RVS TECHNICAL CAMPUS - COIMBATORE
An Autonomous Institution, Affiliated to Anna University, Chennai
(Approved by AICTE New Delhi, Accredited with 'A' Grade by NAAC)
Kumaran Kottam Campus, Kannampalayam Post, Coimbatore – 641 402
Ph: 89039 49934, 0422-2681123, Fax: 0422-2680047, 2687200
E-mail: rvsetgi@rvsgroup.com Website: www.rvstcc.ac.in

Department of Mechatronics Engineering

MR3452 - CONTROL SYSTEMS ENGINEERING MANUAL

Prepared By

Dr.K.Amuthambigaiyin Sundari,

Associate Professor/MCS

VISION AND MISSION OF THE INSTITUTION

VISION
To become a world-class institution for engineering education by imparting quality education that meets global standards, cultivating students who are academically excellent, possess a positive attitude, are ethically sound and socially committed.
MISSION
❖ To enhance academic excellence by educating students in engineering principles, thereby upgrading their knowledge and skills
❖ To empower students with a positive attitude and creative thinking to develop innovative techniques in engineering
❖ To educate and cultivate dedicated learners with ethical values who are socially responsible

VISION AND MISSION OF MECHATRONICS ENGINEERING DEPARTMENT

VISION
To achieve excellence in Mechatronics Engineering through quality education and industry collaboration by integrating mechanical, electronics, computer, and control systems, thereby developing professionally competent, innovative, and socially responsible engineers to contribute in the field of automation, robotics, and smart manufacturing..
MISSION
❖ To strengthen a learner-centric and industry-integrated environment by imparting technical knowledge and practical skills in Mechatronics Engineering, including automation, robotics, IoT, and intelligent systems
❖ To equip learners with industry-relevant tools, problem-solving abilities, and innovative thinking to address real-world engineering challenges across interdisciplinary domains
❖ To nurture learners with professional ethics, leadership qualities, communication skills, and entrepreneurial skills ensuring lifelong learning and adaptability to evolving industrial demands.

MR3452 - CONTROL SYSTEMS ENGINEERING

LIST OF EXPERIMENTS

1. **Mathematical Modeling and Simulation of a Physical Systems and Simulation and Reduction of Cascade and Parallel, and Closed Loop Sub-System.**
2. **Simulation and Analysis of First and Second Order System Equations in Time and Frequency Domain.**
3. **Simulation and Analysis of System using Root-Locus and Bode Plot.**
4. **Simulation and Implementation of PID Combination for First Order Systems.**
5. **Simulation and Implementation of PID Combination for Second Order Systems.**
6. **Auto tuning of PID parameters and analysis of PID Control.**

TOTAL : 30 PERIODS

Ex.No. 1 Mathematical Modelling and Simulation of a Physical Systems and Simulation and Reduction of Cascade and Parallel, and Closed Loop Sub-System.

AIM:

To simulate and analyze the step, ramp, and impulse responses of first order and second order systems and block diagram reduction technique using MATLAB.

TOOLS REQUIRED:

MATLAB Software.

PROCEDURE:

First order and Second order systems:

1. Define the first-order and second order transfer functions, where is the gain and is the time constant.
2. Simulate the responses to step, ramp, and impulse inputs using MATLAB commands.
3. Plot and observe the behavior for different inputs.

Block diagram reduction technique :

1. Write the transfer functions of individual blocks.
2. Use series, parallel and feedback reduction rules to simplify the system.
3. Utilize MATLAB's built-in functions (series, parallel, feedback) to calculate the overall transfer function.
4. Simulate and analyze the system's response.

MATLAB Code:

First Order System

```
%Parameters
K=5;                    %Gain
tau = 2;                %Time constant
num = K;
den = [tau, 1];         %Transfer function:tau*s+1
G = tf(num, den);

%Time vector
 t = 0:0.01:10;

%Step Response figure;
subplot(3,1,1);
step(G,t);
title ('Step Response');
 grid on;
```

```
%Ramp Response
 subplot(3,1,2);
ramp_input=t;                    %Ramp input
lsim(G, ramp_input, t);          title('Ramp Response');
grid on;

%Impulse Response
 subplot(3,1,3);
Impulse (G, t);
title('Impulse Response');
grid on;
```

Second Order System

```
%Parameters
omega_n=5; %Natural frequency
t = 0:0.01:5;
%Damping ratios
zeta_values=[0.2,1,2];           %Under damped, critically damped, over damped

figure;
fori=1:                          length(zeta_values)
zeta = zeta_values(i);
num = [omega_n^2];
den=[1,2*zeta*omega_n,omega_n^2];
 G = tf(num, den);
subplot(3,1,i);
step(G,t);
title(['Step Response(Damping Ratio=',num2str(zeta),')']);
 grid on;
end
```

Block diagram reduction technique

```
% Define individual blocks
G1=tf([1],[1,2]);                %Block1
G2=tf([2],[1,3]);                %Block2
H=tf([1], [1]);                  %Feedbackpath
% Combine G1 and G2 in series
G_series = series(G1, G2);
```

```
% Apply feedback
G_total=feedback(G_series,H);

% Display overall transfer function
disp('Overall Transfer Function:');
 G_total
% Step response of the reduced system
 t = 0:0.01:10;
figure;
step(G_total,t);
title('Step Response of Reduced Block Diagram');
xlabel('Time (seconds)');
ylabel('Response');
grid on;
```

RESULT:

The step, ramp, and impulse responses of first-order and second order systems and block diagram reduction technique using MATLAB are analyzed and simulated.

Ex.No.2.Simulation and Analysis of First and Second Order System Equations in Time and Frequency Domain

AIM: To simulate and analyse First and Second Order System Equations in Time and Frequency Domain using matlab.

APPARATUS: Computer with MATLAB software.

PROCEDURE:

Time Domain

1. Define the transfer function.
2. Simulate its step response.
3. Use the step info function to calculate time-domain parameters.

Frequency Domain

1. Define the transfer function.
2. Use MATLAB's freq resp function to compute the frequency response.
3. Plot the magnitude and phase responses.

MATLAB Code:

Time Domain

```
%Transfer function
%First-order system
 num1 = [1];
 den1=[1,3];
 G1=tf(num1,den1);

%Second-order system
num = [25];
den=[1,6,25];
G2=tf(num,den);

%Step response
[y, t] = step(G1);
%Time-domain parameters
 info = step info(G1);
disp (info);                    %Displays rise time, settling time, peak time, etc.
```

```
 [y, t] = step(G2);
%Time-domain parameters
 info = step info(G2);
  disp (info);                          %Displays rise time, settling time, peak time, etc.

  %Plot the step response
  figure;
   step(G1);
  title ('Step Response with Time Metrics');
  grid on;
  step(G2);
  title ('Step Response with Time Metrics');
  grid on;
```

Frequency Domain

```
  % Define transfer function num = [1];
  den=[1,5,6];
  G=tf(num,den);
  % Frequency range
  omega=log space(-1,2,500);
  [mag,phase]=bode(G,omega);
  % Plot magnitude and phase
  figure;
  subplot(2,1,1);
  semilog x(omega,squeeze(mag));
  title('Magnitude Response');
   xlabel('Frequency (rad/s)');
  ylabel('Magnitude (dB)');
  grid on;
```

```
subplot(2,1,2);
semilog x(omega,squeeze(phase));
title('Phase Response');
 xlabel('Frequency (rad/s)');
 ylabel('Phase (degrees)');
grid on;
```

RESULT:

First and Second Order System Equations in Time and Frequency Domain are simulated and analysed using MATLAB software.

Ex. No.3. Simulation and Analysis of System using Root-Locus and Bode Plot.

AIM: To plot bode plot and root locus using matlab software.

APPARATUS: Computer with MATLAB software.

PROCEDURE:

Bode plot

1. Define the transfer function.
2. Use MATLAB's bode function to plot the Bode diagram.
3. Use the margin function to find gain margin, phase margin, and cutoff frequencies.
4. Analyze the stability and frequency characteristics of the system.

Root locus

1. Define the open-loop transfer function.
2. Use MATLAB's rlocus function to plot the root locus.
3. Observe pole movements as the gain varies.

MATLAB Code:

```
% Define transfer function
 num = [1];
den=[1,2,1];
G=tf(num,den);

% Generate Bode plot
figure;
bode(G); grid on;
title('Bode Plot');

% Calculate gain and phase margins
[Gm,Pm,Wcg,Wcp]=margin(G);
disp('Gain Margin (dB):');
 disp(20*log10(Gm));
disp('Phase Margin (degrees):');
 disp(Pm);
disp('Gain Crossover Frequency (rad/s):');
```

```
 disp(Wcg);
 disp('PhaseCrossoverFrequency(rad/s):');
disp(Wcp);

% Generate root locus plot
figure;
rlocus(G);
title('RootLocus');
grid on;
```

RESULT:

The bode plot and root locus of a transfer function were plotted using MATLAB software.

Ex No. 4. Simulation and Implementation of PID Combination for First Order System

AIM: To Simulate and Implement the PID Combination for First Order System.

APPARATUS: Computer with MATLAB software.

PROCEDURE:

1. Define the plant.
2. Implement the PID controller.
3. Simulate step responses for different values.

MATLAB Code:

```
% Plant transfer function
num = [1];
den=[1,2];
G=tf(num,den);

% PID controller parameters
 Kp = 5; Ki = 2; Kd = 1;
C=pid(Kp,Ki,Kd);
% Closed-loop system
Gc=feedback(C*G,1);
%Step response
 t = 0:0.01:10;
figure;
step(Gc,t);
title('Step Response with PID Controller');
xlabel('Time (seconds)');
ylabel('Response');
grid on;
```

RESULT:

Simulation and Implementation of PID Combination for First Order System is carried out using MATLAB software.

Ex No. 5. Simulation and Implementation of PID Combination for Second Order System

AIM: To Simulate and Implement the PID Combination for Second Order System.

APPARATUS: Computer with MATLAB software.

PROCEDURE:

4. Define the plant.
5. Implement the PID controller.
6. Simulate step responses for different values.

MATLAB Code:

```
% Plant transfer function
num = [1];
den=[1,3,2];
G=tf(num,den);

% PID controller parameters
 Kp = 5; Ki = 2; Kd = 1;
C=pid(Kp,Ki,Kd);
% Closed-loop system
Gc=feedback(C*G,1);
%Step response
 t = 0:0.01:10;
figure;
step(Gc,t);
title('Step Response with PID Controller');
xlabel('Time (seconds)');
ylabel('Response');
grid on;
```

RESULT:

Simulation and Implementation of PID Combination for Second Order System is carried out using MATLAB software.

Ex. NO. 6. Auto tuning of PID parameters and analysis of PID Control.

AIM: To automatically tune and analyse the PID parameters of a given system.

APPARATUS: Computer with MATLAB software.

PROCEDURE :

1. Open MATLAB's PID Tuner using the command pid Tuner.
2. Load the plant transfer function.
3. Adjust tuning parameters and observe the real-time response.
4. Apply the optimized parameters to the system.

MATLAB Code:

```
num=[1];
den=[1,10,20];
%Example:
 G(s)=1/(s^2+10s+20) ;
sys = tf(num, den);

% Open PID Tuner for Automatic Tuning
 Pid Tuner(sys, 'pid')

% Alternatively, use pid tune for direct tuning

[C, info] = pid tune(sys, 'PID');

% Display tuned PID parameters

 disp('Tuned PID Controller:') :
disp(C)

% Closed-loop system analysis
 cl_sys = feedback(C * sys, 1);
step(cl_sys)
%Step response plot
grid on
title('Closed-Loop Step Response')
```

RESULT:

Auto tuning of PID parameters and analysis of PID Control is simulated and implementated.