AE332 – Modeling and Analysis Lab

Name: Shingala Vaidik ID : SC21B054

1. Compound Pendulum:

- **Compound Pendulum, Simulate the pendulum using the single differential equation for θ.**

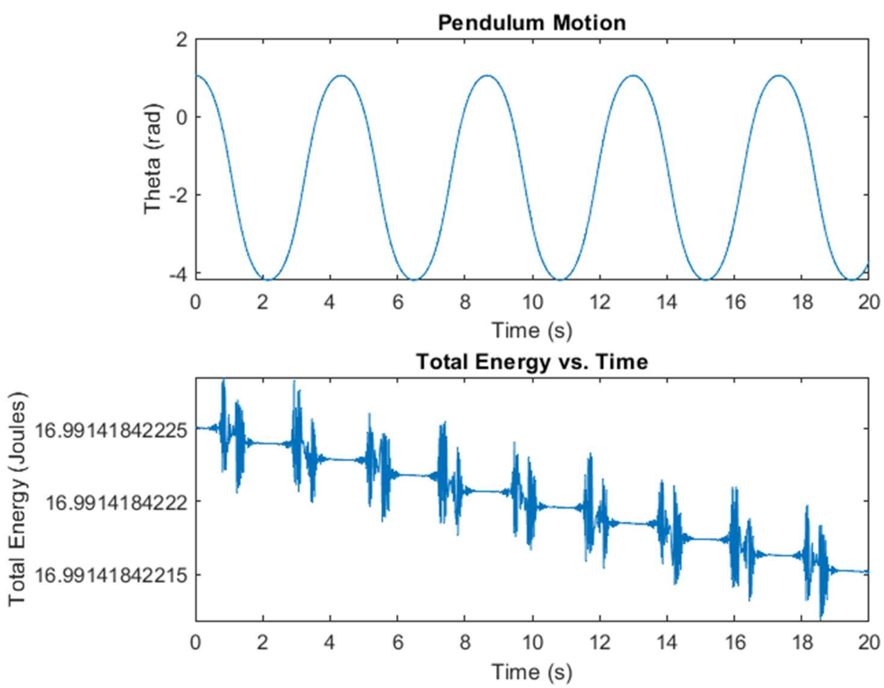
Matlab code: session_04_a_1

```
t = 0:0.01:20;
z0 = [pi/3; 0];
tol1 = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
[t, z] = ode45(@(t, z) pendulum(z), t, z0, tol1);
theta = z(:, 1);
theta_dot = z(:, 2);
PE = m * g * l * sin(theta); % potential energy
KE = 0.5 * (I + m * l^2) * (theta_dot).^2; % kinetic energy
TE = PE + KE; % total energy


% Calculate the energy variation
energy_variation = abs(max(TE) - min(TE));
% Display the energy variation
disp(['Energy Variation (Joules): ', num2str(energy_variation)]);


% Plot theta vs. time
figure;
subplot(2, 1, 1); % Create a subplot with 2 rows and 1 column, and select the
first subplot
plot(t, theta);
xlabel('Time (s)');
ylabel('Theta (rad)');
title('Pendulum Motion');


% Plot total energy vs. time
subplot(2, 1, 2); % Select the second subplot
plot(t, TE);
xlabel('Time (s)');
ylabel('Total Energy (Joules)');
title('Total Energy vs. Time');
```

```
function zd1 = pendulum(z)
    m = 2;
    l = 1;
    I = 1;
    g = 9.81; % fixed variables


    % equations of motion
    zd1 = zeros(2, 1);
    zd1(1) = z(2);
    zd1(2) = -m * g * l * cos(z(1)) / (I + m * l^2);
end
```

Output:

```
Energy Variation (Joules): 1.6718e-10
```



- **Compound Pendulum, Simulate the pendulum using x, y, θ as the configuration variables.**

Matlab code: session_04_a_2

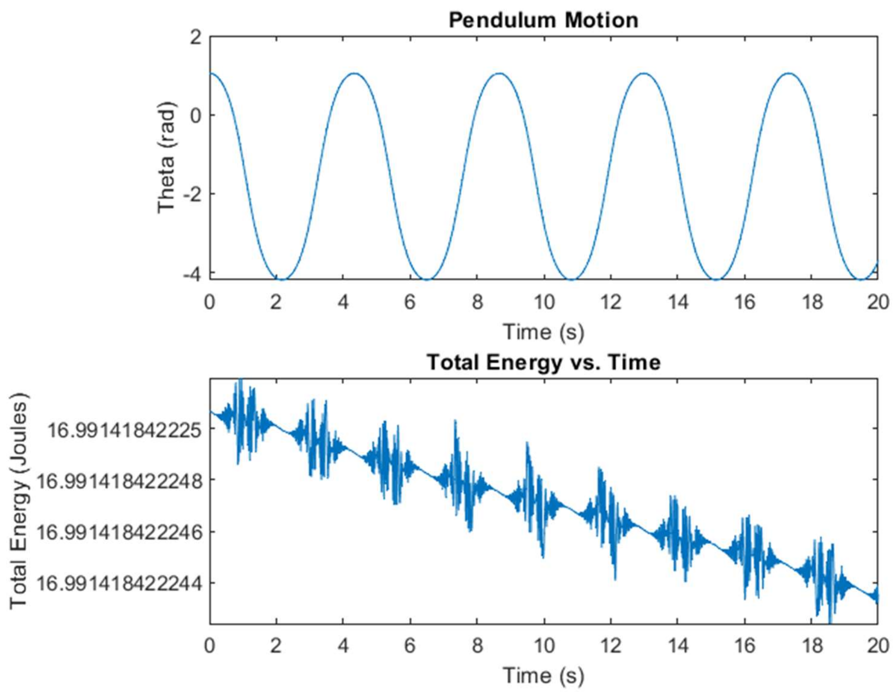
```
t = 0:0.01:20;
z0 = [cos(pi/3);sin(pi/3);pi/3;0;0;0]; %inital conditions vector
tol1 = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
[t, z] = ode45(@(t, z) pendulum_mod(z), t, z0, tol1);
theta = z(:, 3);
theta_dot = z(:, 6);
PE = m * g * l * sin(theta); % potential energy
KE = 0.5 * (I + m * l^2) * (theta_dot).^2; % kinetic energy
```

```
TE = PE + KE; % total energy


% Calculate the energy variation
energy_variation = abs(max(TE) - min(TE));
% Display the energy variation
disp(['Energy Variation (Joules): ', num2str(energy_variation)]);


% Plot theta vs. time
figure;
subplot(2, 1, 1); % Create a subplot with 2 rows and 1 column, and select the
first subplot
plot(t, theta);
xlabel('Time (s)');
ylabel('Theta (rad)');
title('Pendulum Motion');

% Plot total energy vs. time
subplot(2, 1, 2); % Select the second subplot
plot(t, TE);
xlabel('Time (s)');
ylabel('Total Energy (Joules)');
title('Total Energy vs. Time');
```

```
function zd2 = pendulum_mod(z)
     m = 2;
    l = 1;
    I = 1;
    g = 9.81; % fixed variables
%  finding accelerations and reaction forces
    A1 = [0 -m*g 0 -l*cos(z(3))*z(6)^2 -l*sin(z(3))*z(6)^2]';
    B1 = [m 0 0 -1 0
          0 m 0 0 -1
          0 0 I -l*sin(z(3)) l*cos(z(3))
          1 0 l*sin(z(3)) 0 0
          0 1 -l*cos(z(3)) 0 0];
    C1 = B1\A1;
    zd2(1,1) = z(4);
    zd2(2,1) = z(5);
    zd2(3,1) = z(6);
    zd2(6,1) = C1(3,1);
    zd2(4,1) = C1(1,1);
    zd2(5,1) = C1(2,1);
end
```

Output:

Energy Variation (Joules): 9.5817e-12



By comparing the solution, as per above both graphs of **θ** as the configuration variable and with x, y, **θ** as configuration variables, we can see that there is not much change in solution of theta and total energy, also we can verify that energy variation is negligibly small.

2. Compound Pendulum with Spring and Viscous Friction:

- Compound Pendulum with Spring and Viscous Friction, Write the equations of motion and simulate the pendulum using only **θ** as the configuration variable.

Matlab code: session_04_b_1

```
m=2; l=1; I=1; g=9.81; r=0.25; xc=1; yc=1; d0 = 0.75; k=5; b=2; %fixed
parameters
t= 0:0.01:20;
z0 = [pi/3; 0; 0];
tol1 = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
[t, z] = ode45(@(t, z) pendulum_spring(z), t, z0, tol1);
```

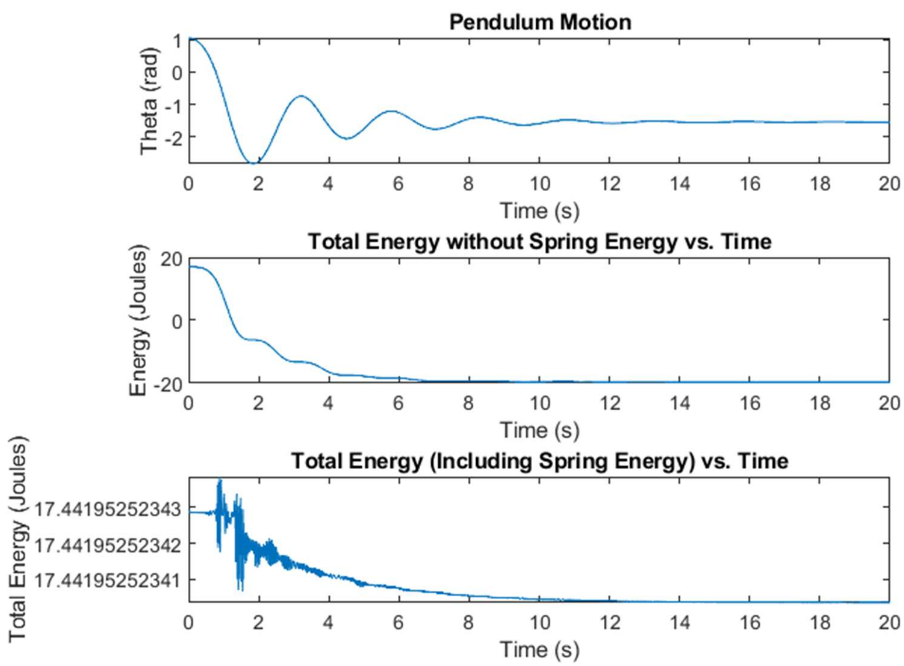
```
PE2 = m.*g.*l.*sin(z(:,1)) + 0.5.*k.*(sqrt((xc-r.*cos(z(:,1))).^2+(yc-
r.*sin(z(:,1))).^2)-d0).^2; %potential energy
KE2 = 0.5.*(I+m*l^2).*z(:,2).^2; %kinetic energy
DE2 = z(:,3); %energy dissipated due to damper
TE2 = PE2+KE2+DE2; %total energy


% Calculate the energy variation
energy_variation = abs(max(TE2) - min(TE2));
% Display the energy variation
disp(['Energy Variation (Joules): ', num2str(energy_variation)]);


% Plot theta vs. time
figure;
subplot(3, 1, 1); % Create a subplot with 2 rows and 1 column, and select the
first subplot
plot(t, theta);
xlabel('Time (s)');
ylabel('Theta (rad)');
title('Pendulum Motion');


% Plot total energy vs. time (without spring energy)
subplot(3, 1, 2);
plot(t, TE);
xlabel('Time (s)');
ylabel('Energy (Joules)');
title('Total Energy without Spring Energy vs. Time');


% Plot total energy vs. time (including spring energy)
subplot(3, 1, 3);
plot(t, TE2);
xlabel('Time (s)');
ylabel('Total Energy (Joules)');
title('Total Energy (Including Spring Energy) vs. Time');
```

```
function zd3 = pendulum_spring(z)
m=2; l=1; I=1; g=9.81; r=0.25; xc=1; yc=1; d0 = 0.75; k=5; b=2; %fixed
parameters
     % Calculate spring-related variables
    phi = atan2(yc - r * sin(z(1)), xc - r * cos(z(1))); % Angle made by
spring force with horizontal
    p = phi - z(1); % Angle between line joining A and G and spring force
    d = sqrt((xc - r * cos(z(1)))^2 + (yc - r * sin(z(1)))^2); % Length of
spring
    Fs = k * (d - d0); % Spring force


    % Equations of motion
    zd3 = zeros(3, 1);
```

```
    zd3(1) = z(2);
    zd3(2) = (-b * z(2) - m * g * l * cos(z(1)) + Fs * r * sin(p)) / (I + m *
l^2);
    zd3(3) = b * z(2)^2;
end
```

Output:

```
Energy Variation (Joules): 3.4653e-11
```



- **Compound Pendulum with Spring and Viscous Friction, Write the equations of motion and simulate the pendulum using x, y, θ as configuration variables.**

Matlab code: session_04_b_2

```
m=2; l=1; I=1; g=9.81; r=0.25; xc=1; yc=1; d0 = 0.75; k=5; b=2; %fixed
parameters
t= 0:0.01:20;
z0 = [cos(pi/3);sin(pi/3);pi/3;0;0;0;0];
tol1 = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
[t, z] = ode45(@(t, z) pendulum_spring_mod(z), t, z0, tol1);


PE3 = m.*g.*z(:,2) + 0.5.*k.*(sqrt((xc-r.*cos(z(:,3))).^2+(yc-
r.*sin(z(:,3))).^2)-d0).^2; %potential energy
KE3 = 0.5.*(I+m*l^2).*z(:,6).^2; %kinetic energy
DE3 = z(:,7); %energy dissipated due to damper
TE3 = PE3+KE3+DE3; %total energy
```

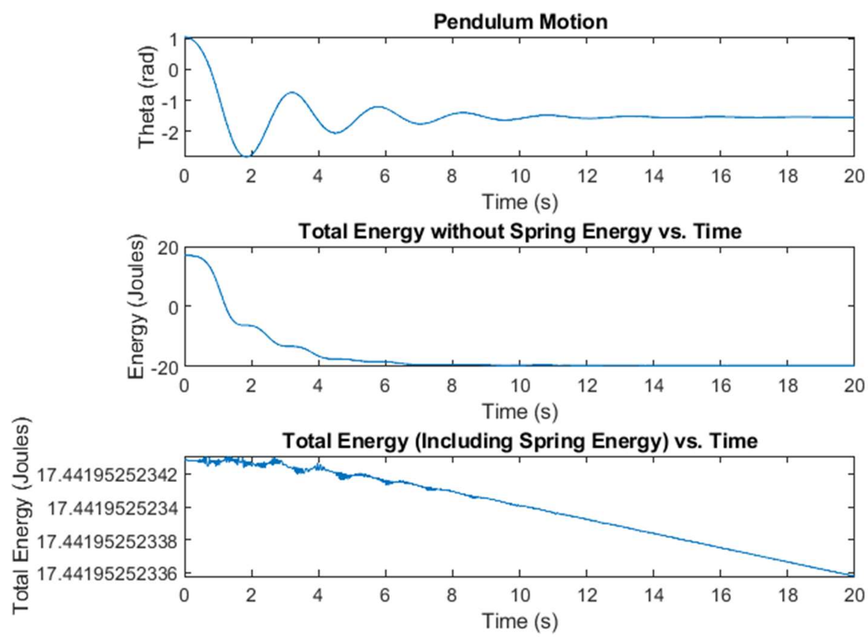
```
% Calculate the energy variation
energy_variation = abs(max(TE3) - min(TE3));
% Display the energy variation
disp(['Energy Variation (Joules): ', num2str(energy_variation)]);


% Plot theta vs. time
figure;
subplot(3, 1, 1); % Create a subplot with 2 rows and 1 column, and select the
first subplot
plot(t, theta);
xlabel('Time (s)');
ylabel('Theta (rad)');
title('Pendulum Motion');
% Plot total energy vs. time (without spring energy)
subplot(3, 1, 2);
plot(t, TE);
xlabel('Time (s)');
ylabel('Energy (Joules)');
title('Total Energy without Spring Energy vs. Time');
% Plot total energy vs. time (including spring energy)
subplot(3, 1, 3);
plot(t, TE3);
xlabel('Time (s)');
ylabel('Total Energy (Joules)');
title('Total Energy (Including Spring Energy) vs. Time');
```

```
function zd4 = pendulum_spring_mod(z)
m=2; l=1; I=1; g=9.81; r=0.25; xc=1; yc=1; d0 = 0.75; k=5; b=2; %fixed
parameters
    xb = r*cos(z(3)); yb = r*sin(z(3)); %coordiantes of point B
    phi=atan2(yc-yb,xc-xb); %angle made by spring force with horizontal
    p=phi-z(3); %angle between line joining A and G and spring force
    d=sqrt((xc-xb)^2+(yc-yb)^2); %length of spring
    Fs=k*(d-d0); %spring force
    Fs2 = Fs*sin(phi); Fs1 = Fs*cos(phi); %components of spring force in y and
x directions

    %finding accelerations and reaction forces
    A = [m 0 0 -1 0
         0 m 0 0 -1
         0 0 I -l*sin(z(3)) l*cos(z(3))
         1 0 l*sin(z(3)) 0 0
         0 1 -l*cos(z(3)) 0 0];
    B = [Fs1 Fs2-m*g (-b*z(6)-Fs*(l-r)*sin(p)) -l*cos(z(3))*(z(6)^2) -
l*sin(z(3))*(z(6)^2)]';
    C = A\B;

    zd4(1,1) = z(4);
    zd4(2,1) = z(5);
    zd4(3,1) = z(6);
    zd4(6,1) = C(3,1);
```

```
    zd4(4,1) = C(1,1);
    zd4(5,1) = C(2,1);
    zd4(7,1) = b*z(6)^2;

end
```

Output:

```
Energy Variation (Joules): 7.2969e-11
```



By comparing the solution, as per above both graphs of **θ** as the configuration variable and with x, y, **θ** as configuration variables, we can see that there is not much change in solution of theta, Total Energy without Spring Energy and Total Energy (Including Spring Energy), also we can verify that energy variation is negligibly small.

3. Slider Crank Mechanism:

- **Slider Crank Mechanism, choose an independent position coordinate (θ_2) as the dependent variable for ode and simulate the slider-crank.**

Matlab code: session_04_c_1

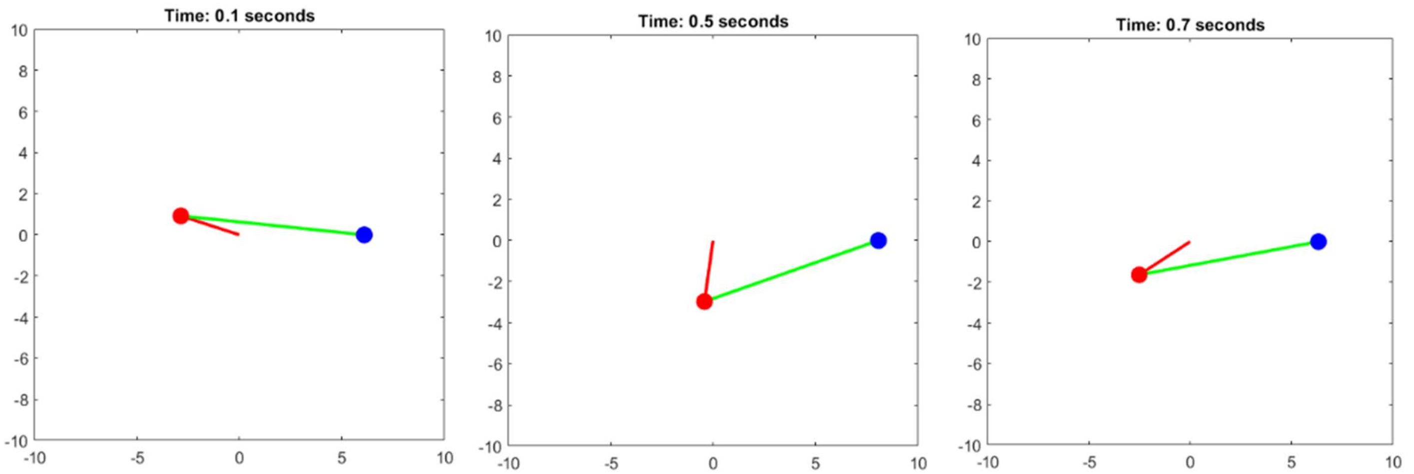
```
m2=1; I2=0.1; l2=3; r2=0.5*l2; %crank
m3=3; I3=0.3; l3=9; r3=0.5*l3; %connecting rod
m4=1.5; I4=0.05; a=0.01; %piston
g = 9.81;
```

```
t= 0:0.1:25;
z0 = [pi/6;0];
tol1 = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
[t, z] = ode45(@(t, z) sldr_crnk(z), t, z0, tol1);


t2 = z(:,1); t2d = z(:,2); %theta2 and theta dot 2
t3 = asin(-l2.*sin(t2)/l3); t3d = t2d.*l2.*cos(t2)./(l3.*cos(t3)); %theta3 and
theta dot 3


%position coordiantes
xg2 = r2.*cos(t2); yg2 = r2.*sin(t2); %position of center of mass of crank
xg3 = l2.*cos(t2)+r3.*cos(t3); yg3 = l2.*sin(t2)+r3.*sin(t3); %position of
center of connecting rod
xg4 = l2.*cos(t2)+l3.*cos(t3)+a; yg4 = l2.*sin(t2)+l3.*sin(t3); %position of
piston
x2 = l2.*cos(t2); y2 = l2.*sin(t2); %position of point B
x3 = l2.*cos(t2)+l3.*cos(t3); y3 = l2.*sin(t2)+l3.*sin(t3); %position of point
C


% Create a figure for the animation
figure;


% Define functions to interpolate positions
x_2 = @(i) interp1(t, l2.*cos(z(:,1)), i); %+ (l2 - r2) * cos(interp1(t, z(:,
1), i));
y_2 = @(i) interp1(t, l2.*sin(z(:,1)), i); %+ (l2 - r2) * sin(interp1(t, z(:,
1), i));


x_3 = @(i) interp1(t, l2.*cos(z(:,1))+l3.*cos(asin(-l2.*sin(z(:, 1))/l3)), i);
%+ (l3 - r3) * cos(interp1(t, asin(-l2.*sin(z(:, 1))/l3), i));
y_3 = @(i) interp1(t, l2.*sin(z(:,1))+l3.*sin(asin(-l2.*sin(z(:, 1))/l3)), i);
%+ (l3 - r3) * sin(interp1(t, asin(-l2.*sin(z(:, 1))/l3), i));


x_4 = @(i) interp1(t, l2.*cos(z(:,1))+l3.*cos(asin(-l2.*sin(z(:, 1))/l3)), i)
+ a;
y_4 = @(i) 0;


% Define colors for the links
link_colors = ['r', 'g', 'b'];


% Create an animation
for i = 1:length(t)
    % Clear the figure to update the animation
    clf;
```

```
    % Plot links
    plot([0, x_2(i)], [0, y_2(i)], 'Color', link_colors(1), 'LineWidth', 2);
    hold on;
    plot([x_2(i), x_3(i)], [y_2(i), y_3(i)], 'Color', link_colors(2),
'LineWidth', 2);
    plot([x_3(i), x_4(i)], [y_3(i), y_4(i)], 'Color', link_colors(3),
'LineWidth', 2);


    % Plot joints
    plot(x_2(i), y_2(i), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');
    plot(x_3(i), y_3(i), 'go', 'MarkerSize', 10, 'MarkerFaceColor', 'g');
    plot(x_4(i), y_4(i), 'bo', 'MarkerSize', 10, 'MarkerFaceColor', 'b');


    % Set axis limits
    axis equal;
    xlim([-10, 10]);
    ylim([-10, 10]);


    % Add a title
    title(['Time: ', num2str(t(i)), ' seconds']);


    % Pause to control the animation speed
    pause(1);


    hold off;
end
```

```
function zd5 = sldr_crnk(z)
    m2=1; I2=0.1; g=9.81; r2=0.5*3; % fixed variables for link 2
    zd5(1,1)=z(2);
    zd5(2,1)=-m2*g*r2*cos(z(1))/(I2 + m2*r2^2);
end
```

Output:



Here we can see position of Slider Crank Mechanism at different time instant from that we can verify the solution of Slider Crank Mechanism.

- **Slider Crank Mechanism, choosing all the nine configuration variables as dependent variables of time for ode and simulate the slider-crank.**

Matlab code: session_04_a_1

```
    m2=1; I2=0.1; l2=3; r2=0.5*l2; %crank
    m3=3; I3=0.3; l3=9; r3=0.5*l3; %connecting rod
    m4=1.5; I4=0.05; a=0.01; %piston
    g = 9.81;
%initial conditions
theta2 = pi/6; theta2d = 0;
theta3 = asin(-l2*sin(theta2)/l3); theta3d =
theta2d*l2*cos(theta2)/(l3*cos(theta3));
xg2i = r2*cos(theta2); yg2i = r2*sin(theta2);
xg3i = l2*cos(theta2)+r3*cos(theta3); yg3i = l2*sin(theta2)+r3*sin(theta3);
xg4i = l2*cos(theta2)+l3*cos(theta3)+a; yg4i = l2*sin(theta2)+l3*sin(theta3);
theta4 = 0; theta4d = 0;
xg2d = -r2*sin(theta2)*theta2d; yg2d = r2*cos(theta2)*theta2d;
xg3d = -l2*sin(theta2)*theta2d-r3*sin(theta3)*theta3d; yg3d =
l2*cos(theta2)*theta2d +r3*cos(theta3)*theta3d;
xg4d = -l2*sin(theta2)*theta2d-l3*sin(theta3)*theta3d; yg4d =
l2*cos(theta2)*theta2d +l3*cos(theta3)*theta3d;


t= 0:0.2:25;
```

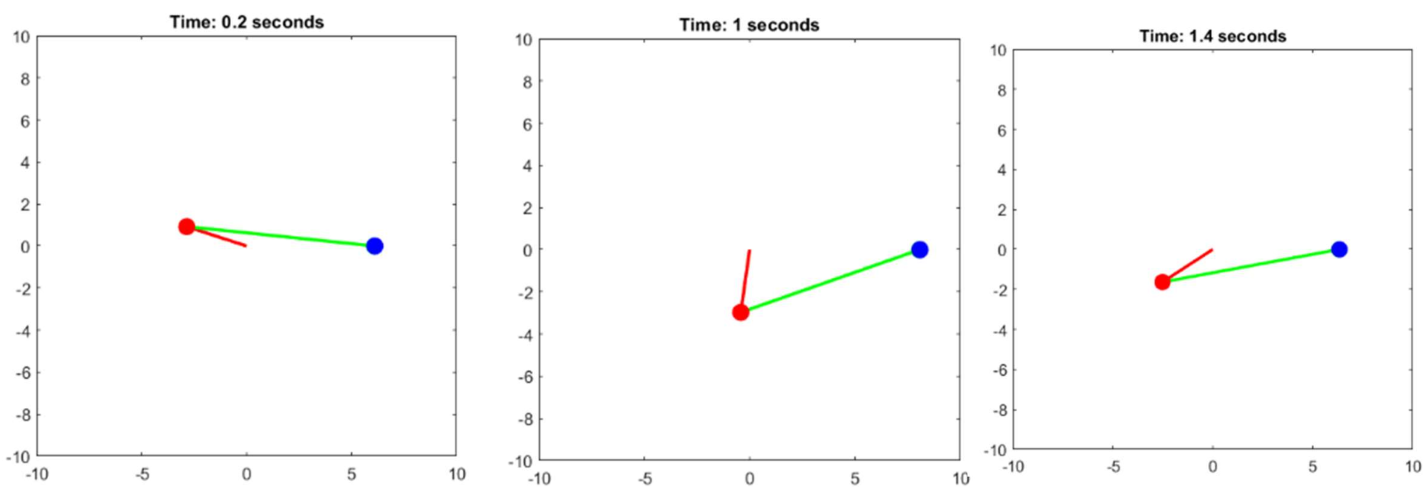
```
z0 =
[xg2i,yg2i,theta2,xg3i,yg3i,theta3,xg4i,yg4i,theta4,xg2d,yg2d,theta2d,xg3d,yg3
d,theta3d,xg4d,yg4d,theta4d]'; %initial conditions vector
tol1 = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
[t, z] = ode45(@(t, z) sldr_crnk_mod(z), t, z0, tol1);


% Create a figure for the animation
figure;


% Define functions to interpolate positions
x_2 = @(i) interp1(t, z(:, 1), i) + (l2 - r2) * cos(interp1(t, z(:, 3), i));
y_2 = @(i) interp1(t, z(:, 2), i) + (l2 - r2) * sin(interp1(t, z(:, 3), i));


x_3 = @(i) interp1(t, z(:, 4), i) + (l3 - r3) * cos(interp1(t, z(:, 6), i));
y_3 = @(i) interp1(t, z(:, 5), i) + (l3 - r3) * sin(interp1(t, z(:, 6), i));


x_4 = @(i) interp1(t, z(:, 7), i) + a;
y_4 = @(i) 0;


% Define colors for the links
link_colors = ['r', 'g', 'b'];


% Create an animation
for i = 1:length(t)
    % Clear the figure to update the animation
    clf;


    % Plot links
    plot([0, x_2(i)], [0, y_2(i)], 'Color', link_colors(1), 'LineWidth', 2);
    hold on;
    plot([x_2(i), x_3(i)], [y_2(i), y_3(i)], 'Color', link_colors(2),
'LineWidth', 2);
    plot([x_3(i), x_4(i)], [y_3(i), y_4(i)], 'Color', link_colors(3),
'LineWidth', 2);


    % Plot joints
    plot(x_2(i), y_2(i), 'ro', 'MarkerSize', 10, 'MarkerFaceColor', 'r');
    plot(x_3(i), y_3(i), 'go', 'MarkerSize', 10, 'MarkerFaceColor', 'g');
    plot(x_4(i), y_4(i), 'bo', 'MarkerSize', 10, 'MarkerFaceColor', 'b');


    % Set axis limits
    axis equal;
    xlim([-10, 10]);
    ylim([-10, 10]);
```

```
    % Add a title
    title(['Time: ', num2str(t(i)), ' seconds']);


    % Pause to control the animation speed
    pause(1);


    hold off;
end
```

```
function zd6 = sldr_crnk_mod(z)
    m2=1; I2=0.1; l2=3; r2=0.5*l2; %crank
    m3=3; I3=0.3; l3=9; r3=0.5*l3; %connecting rod
    m4=1.5; I4=0.05; a=0.01; %piston
    g = 9.81;
    t2=z(3); t3=z(6); t2d=z(12); t3d=z(15);
    %position coordiantes
    xg2 = r2*cos(t2); yg2 = r2*sin(t2); %position of center of mass of crank
    xg3 = l2*cos(t2)+r3*cos(t3); yg3 = l2*sin(t2)+r3*sin(t3); %position of
center of connecting rod
    xg4 = l2*cos(t2)+l3*cos(t3)+a; yg4 = l2*sin(t2)+l3*sin(t3); %position of
piston
    x2 = l2*cos(t2); y2 = l2*sin(t2); %position of point B
    x3 = l2*cos(t2)+l3*cos(t3); y3 = l2*sin(t2)+l3*sin(t3); %position of point
C


    A = zeros(17,17); B = zeros(17,1);

    %finding reaction forces and accelerations

    A(1,1) = m2; A(1,10) = -1; A(1,12) = -1;
    A(2,2) = m2; A(2,11) = -1; A(2,13) = -1; B(2,1) = -m2*g;
    A(3,3) = I2; A(3,10) = -yg2; A(3,11) = xg2; A(3,12) = yg2-y2; A(3,13) =
x2-xg2;
    A(4,4) = m3; A(4,12) = 1; A(4,14) = -1;
    A(5,5) = m3; A(5,13) = 1; A(5,15) = -1; B(5,1) = -m3*g;
    A(6,6) = I3; A(6,12) = yg3-y2; A(6,13) = x2-xg3; A(6,14) = y3-yg3; A(6,15)
= xg3-x3;
    A(7,7) = m4; A(7,14) = 1;
    A(8,8) = m4; A(8,15) = -1; A(8,16) = -1; B(8,1) = -m4*g;
    A(9,9) = I4; A(9,15) = -a; A(9,17) = -1;
    A(10,1) = 1; A(10,3) = r2*sin(t2); B(10,1) = -r2*cos(t2)*(t2d)^2;
    A(11,2) = 1; A(11,3) = -r2*cos(t2); B(11,1) = -r2*sin(t2)*(t2d)^2;
    A(12,4) = 1; A(12,3) = l2*sin(t2); A(12,6) = r3*sin(t3); B(12,1) = -
l2*cos(t2)*(t2d)^2 -r3*cos(t3)*(t3d)^2;
    A(13,5) = 1; A(13,3) = -l2*cos(t2); A(13,6) = -r3*cos(t3); B(13,1) = -
l2*sin(t2)*(t2d)^2 -r3*sin(t3)*(t3d)^2;
    A(14,7) = 1; A(14,3) = l2*sin(t2); A(14,6) = l3*sin(t3);B(14,1) = -
l2*cos(t2)*(t2d)^2 -l3*cos(t3)*(t3d)^2;
```

```
    A(15,8) = 1; A(15,3) = -l2*cos(t2); A(15,6) = -l3*cos(t3); B(15,1) = -
l2*sin(t2)*(t2d)^2 -l3*sin(t3)*(t3d)^2;
    A(16,8) = 1;
    A(17,9) = 1;


    C = A\B;

    zd6(1,1)=z(10);
    zd6(2,1)=z(11);
    zd6(3,1)=z(12);
    zd6(4,1)=z(13);
    zd6(5,1)=z(14);
    zd6(6,1)=z(15);
    zd6(7,1)=z(16);
    zd6(8,1)=z(17);
    zd6(9,1)=z(18);
    zd6(10:18,1)= C(1:9,1); %accelerations

end
```

Output:



Here we can see position of Slider Crank Mechanism at different time instant from that we can verify the solution of Slider Crank Mechanism.

By comparing the solution, as per above both graphs of **θ** as the configuration variable and with x, y, **θ** as configuration variables, we can see that there is not much change in solution.