Multibody Dynamics B - Assignment 6

July 4, 2019

Short problem statement
We are working with a quick-return mechanism that can be seen in figure 1.

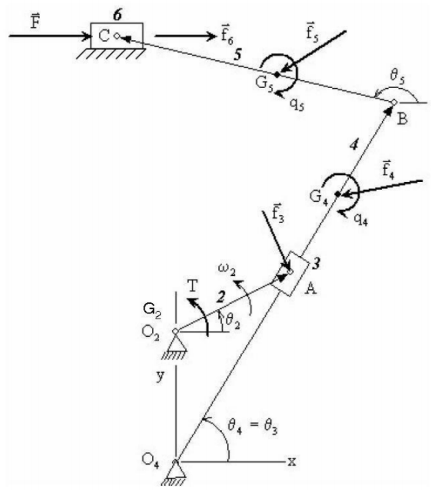

Figure 1 A Quick-Return Mechanism

Figure 1: Quick-return mechanism that we are working with

Crank 2 drives via slider 3 the rocker 4 and bar 5. The centre of the mass of each bar is located at G_i. Now the specification of the system is $O_2A = 0.2m$, $O_4B = 0.7m$, $BC = 0.6m$, $O_4O_2 = 0.3m$, $O_4G_4 = 0.4m$, $BG_5 = 0.3m$, $y_c = 0.9m$, $m_3 = 0.5kg$, $m_4 = 6kg$, $m_5 = 4kg$, $m_6 = 2kg$, $J_4 = 10kgm^2$, $J_5 = 6kgm^2$, $J_3 = 0kgm^2$, $J_2 = 100kgm^2$, $F = 1000N$ and $T = 0$. The initial velocity and angle of θ_2 are $75rpm$ and 0 respectively, there is assumed no friction and no gravity. Now our object is to determine the motion of the mechanism with numerical integration after deriving the equation of motion. We derive the equation of motion in terms of the generalized coordinates and we use the cut loop method on the closed system. Furthermore, we are supposed to use the coordinate projection method to make the solutions fulfil the constraints. Also show:

- The angular speed of crank 2, rocker 4 and bar 5
- The sliding speed of slider 3 with respect to rocker 4
- The horizontal position, speed and acceleration of slider 6
- The normal force exerted by slider 3 on rocker 4
- The normal exerted by slider 6 on the ground

a)

1 Equation of motion

We started by finding the equation of motion of the system. Looking closely at the system it can be observed that the system can be described by only one generalized coordinate, however a complex system like this, a closed system, can be more easily represented if we cut the closed system apart and add additional generalized coordinates. We used this to our advantage when deriving the equation of motion and cut the loop in C. Furthermore, a constrained was defined to link crank 2 with rocker 4, via slider in A, so another 'cut' was made there. This means that three generalized coordinates must be defined and we define those as:

$$q = \begin{bmatrix}\alpha\\ \beta\\ \gamma\end{bmatrix} = \begin{bmatrix}\theta_2\\ \theta_4\\ \theta_5\end{bmatrix}; \quad \dot{q} = \begin{bmatrix}\dot{\alpha}\\ \dot{\beta}\\ \dot{\gamma}\end{bmatrix} = \begin{bmatrix}\dot{\theta_2}\\ \dot{\theta_4}\\ \dot{\theta_5}\end{bmatrix}; \quad \ddot{q} = \begin{bmatrix}\ddot{\alpha}\\ \ddot{\beta}\\ \ddot{\gamma}\end{bmatrix} = \begin{bmatrix}\ddot{\theta_2}\\ \ddot{\theta_4}\\ \ddot{\theta_5}\end{bmatrix}$$

Now we define the CoM coordinates as

$$x_A = O_2Acos(\alpha)$$

$$y_A == O_4O_2 + O_2Asin(\alpha)$$

$$x_4 = O_4G_4cos(\beta)$$

$$y_4 = O_4G_4sin(\beta)$$

$$x_5 = O_4Bcos(\beta) + Bg_5cos(\gamma)$$

$$y_5 = O_4Bsin(\beta) + Bg_5sin(\gamma)$$

$$x_6 = O_4Bcos(\beta) + BCcos(\gamma)$$

We then used the TMT method, with extra constraints due to the loop cutting method. Our equation of motion in matrix form then looked like this, this can be seen in chapter 14.4 or 16.1.1. in the book.

$$\begin{bmatrix}\ddot{q}_k\\ \lambda_c\end{bmatrix} = \begin{bmatrix}T_{i,j}M_{ik}T_{k,l} & C_{c,j}\\ C_{c,l} & 0_{cc}\end{bmatrix}^{-1} \begin{bmatrix}Q_j + T_{i,j}(F_i - Mikg_k\\ -C_{c,jl}\dot{q}_j\dot{q}_l\end{bmatrix}$$

Where we defined

$$T_i = \begin{bmatrix}\alpha & x_A & y_A & \beta & x_4 & y_4 & \beta & x_5 & y_5 & \gamma & x_6\end{bmatrix}^T$$

Notice that $\theta_3 = \theta_4 = \beta$

$$T_{i,j} = \frac{\partial T_i}{\partial q_j}$$

i.e. $T_{i,j}$ is the jacobian, and

$$M = diag(\begin{bmatrix}J_2 & m_3 & m_3 & J_3 & m_4 & m_4 & J_4 & m_5 & m_5 & J_5 & m_6\end{bmatrix})$$

and g_k are the convective acceleration terms. Now we have two constraints, due to the loop cutting, which are

$$C_6 = O_4Bsin(\beta) + BCsin(\gamma) - y_c = 0$$

$$c_A = arctan\frac{x_A}{y_A} - arctan\frac{x_4}{y_4} = 0$$

which gives us

$$c_A = \frac{x_A}{y_A} - \frac{x_4}{y_4} = 0$$

Now C_6 is due to the C slide not being able to move vertically, i.e. it always has to stay y_c length away from the origin. The second constraint C_A is due to the fact that $\theta_3 = \theta_4$ and thus C_A must apply. now we can define

$$C_c = \begin{bmatrix} C_6 \\ C_A \end{bmatrix}; \quad C_{c,j} = \frac{\partial C_c}{\partial q_j}; \quad C_{c,jl} = \frac{\partial C_{c,j}}{\partial q_l}$$

We now have two lagrange multipliers, which tell us the reaction force where we cut the loop, i.e the force which slider 3 exerts on rocker 4 and the force which slider 6 exerts on the ground.
Now our force matrix is defined as, notice that there is no gravitational field and only one external force F that acts on slider 6 in the horizontal direction

$$F_i = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix}^T$$

2 Numerical integration method

The numerical integration method used here was the Runge-Kutta 4th order method, which follows the scheme

$$\begin{gathered} k_1 = f(t_n, y_n) \\ k_2 = f(t_n + h/2, y_n + h/2 * k_1) \\ k_3 = ff(t_n + h/2, y_n + h/2 * k_2) \\ k_4 = ff(t_n + h, y_n + h * k_3) \\ y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{gathered}$$

3 Coordinate Projection Method

By introducing the constraints to close the loops that we have cut open, we have created a problem of drift in the constraints and in the coordinates during numerical integration. To remove these drifts we introduce the coordinate projection method. We use the Gauss-Newton method to implement our coordinate projection. It says that

$$q_{n+1} = \bar{q}_{n+1} + \delta q_{n+1}$$

where

$$y_{n+1} = \begin{bmatrix} \bar{q}_{n+1} \\ \dot{\bar{q}}_{n+1} \end{bmatrix}$$

where we use

$$C_c(\bar{q}_{n+1}) + C_{c,k}(\bar{q}_{n+1})\delta q_{n+1} = 0$$

This gives us

$$\delta q_{n+1} = -C_k^T (C_k C_k^T)^{-1} C$$

where

$$\begin{gathered} C_k = C_{c,k}(\bar{q}_{n+1}) \\ C = C_c(\bar{q}_{n+1}) \end{gathered}$$

This is done until the constraint equations are less then the defined tolerance, we iterate this in a while loop, or until you hit the maximum iteration count, here it is 10^{-12} and 10 respectively.
For velocities this is similar except we don't need to iterate that process since these are linear equations, so we have a linear least square problem, we define

$$\dot{q}_{n+1} = \dot{\bar{q}}_{n+1} + \delta\dot{q}_{n+1}$$

where

$$C_{c,k}(\bar{q}_{n+1})\dot{\bar{q}}_{n+1} + C_{c,k}(\bar{q}_{n+1})\delta\dot{q}_{n+1} = 0$$

which gives

$$\delta\dot{q}_{n+1} = -C_k^T(C_kC_k^T)^{-1}C_k * \dot{\bar{q}}_{n+1}$$

where

$$C_k = C_{c,k}(\bar{q}_{n+1})$$

4 Results

b)

We first look at angular speeds of crank 2, rocker 4 and bar 5, this can be seen in figure 2 below:

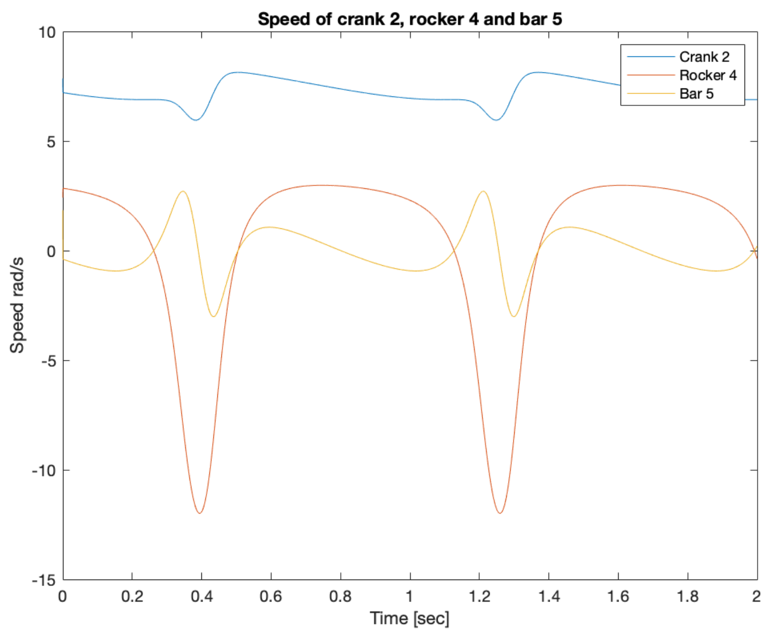

Figure 2: Angular speeds of crank 2, rocker 4 and bar 5

Looking at this picture we see that crank 2 starts at it's initial velocity and it gradually drops, due to the external force acting on slider 6, i.e. the force decelerates the crank when $theta_2$ is on the region of $0 - \pi$. The bump that is noticeable for all three velocities, after the gradual drop in velocity for crank 2, is when the quick-return mechanism kicks in, i.e. when θ_2 is on the range of $\pi/$ to $2\pi/$, because then crank 2 is not rotating against the force in slider 6 but rotates in the same direction of it. The mechanism kind of shoots back, i.e. the quick return, and then stabilizes again when we reach a full circle, like is expected, this then repeats itself every rotation.

c)

We now look at the velocity of slider 3 with regard to rocker 4. To do this we calculate the velocity of slider 3 and rocker 4 in the x and y direction, from the CoM coordinates where we find the velocity by taking the jacobian, i.e.

$$\dot{x_A} = \frac{\partial x_A}{\partial q}\dot{q}$$

and the velocity of slider 3 and rocker 4 is then

$$v_A = \sqrt{\dot{x_A}^2 + \dot{y_A}^2}; V_4 = \sqrt{\dot{x_4}^2 + \dot{y_4}^2}$$

and thus the velocity of slider 3 with respect to rocker 4 is

$$v_{34} = v_3 - v_4$$

Which gives

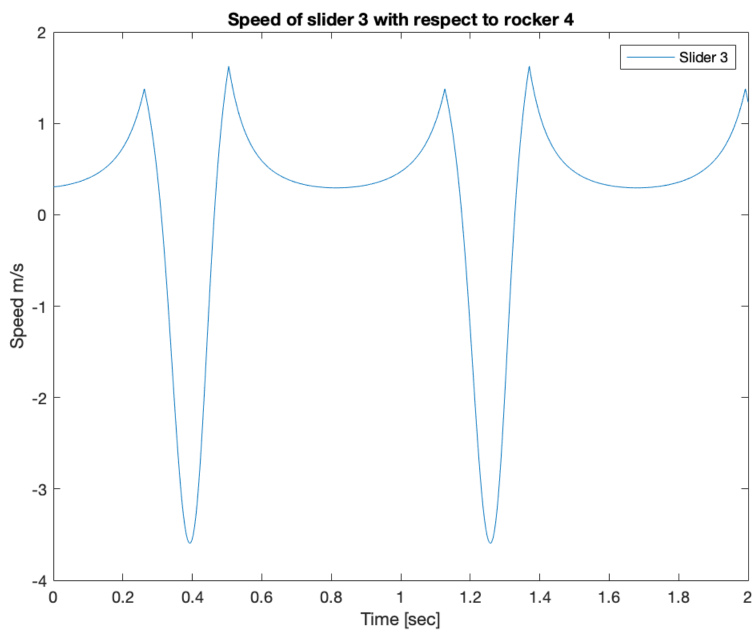

Figure 3: Speed of slider 3 with respect to rocker 4.

Now what is interesting here are the quick changes in velocity, i.e. when it goes from positive to negative back to positive. This is due to the nature of the slider when it moves with crank 2. When crank 2 moves to $\pi/2$ the slider is moving towards the center of mass of bar 4, i.e. the rocker, and when it goes from $\pi/2$ to $3\pi/2$ then the slider is moving away from the center of mass of bar 4, i.e. the rocker, resulting in the negative velocity and when it is again moving to $\pi/2$ the velociti goes back to positive. The high acceleration, deceleration that happens is the quick return mechanism, i.e. when we are moving from π to $3\pi/2$ the velocity becomes negative really quick, and when it is again moving from $3\pi/2$ to 2π the veloicty becomes positive really quick, these are the steep valleys that we observe.

d)

Now we observe the horizontal position, velocity and acceleration of slider 6. Again we calculate the horizontal velocity like we calculated the y and x velocities for slider 3 and rocker 4, i.e. with the jacobian. This was similarly done to find the acceleration where we calculated the acceleration as

$$\ddot{x}_6 = \frac{\partial \dot{x}_6}{\partial q}\dot{q} + \frac{\partial \dot{x}_6}{\partial \dot{q}}\ddot{q}$$

This gave us the following plots:

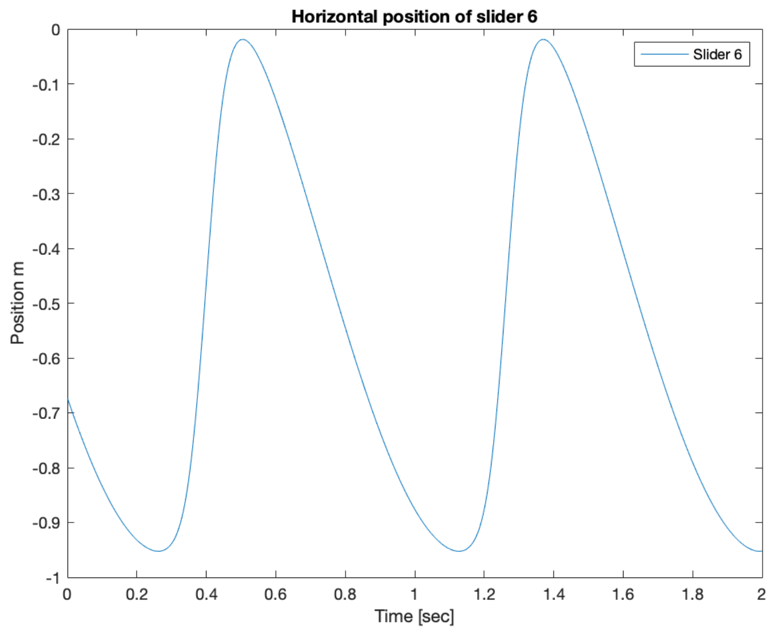

Figure 4: Horizontal position of slider 6.

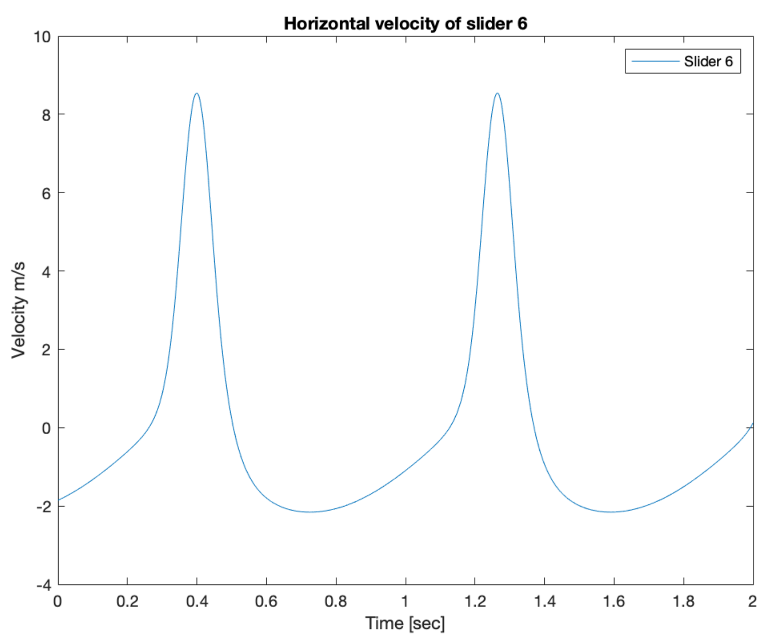

Figure 5: Horizontal velocity of slider 6.

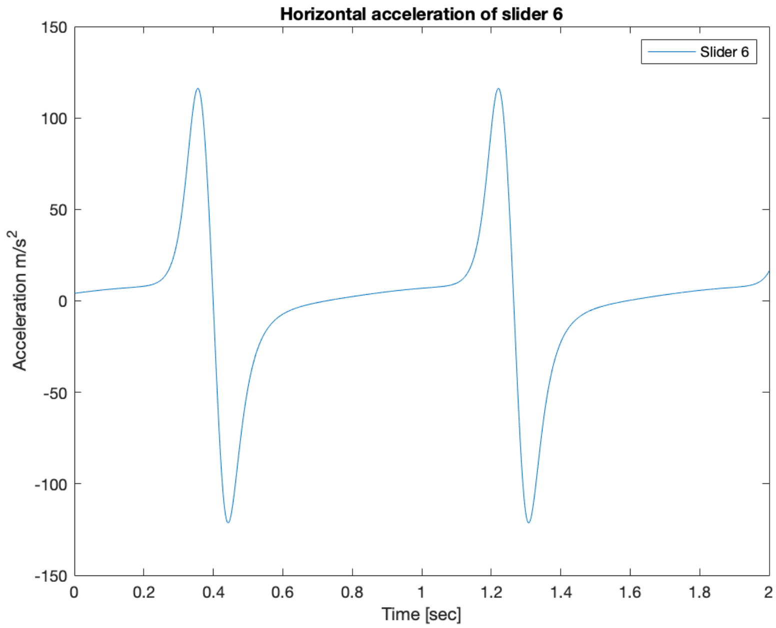

Figure 6: Horizontal acceleration of slider 6.

Here we observe the rapid change in velocity and acceleration is the same what we have observed in previous figures and again is due to how the system reacts when crank 2 is in certain position, i.e. whether it is working against the horizontal force on slider 6 or working with. This again results in the fast changes from acceleration to deceleration, which leads to fast changes in velocity. However we notice that the position is similar to a sinusoidal, however we see that the curve towards the origin is steeper, i.e. we get there faster

e) and f)

Now lastly we will look at the reaction forces.

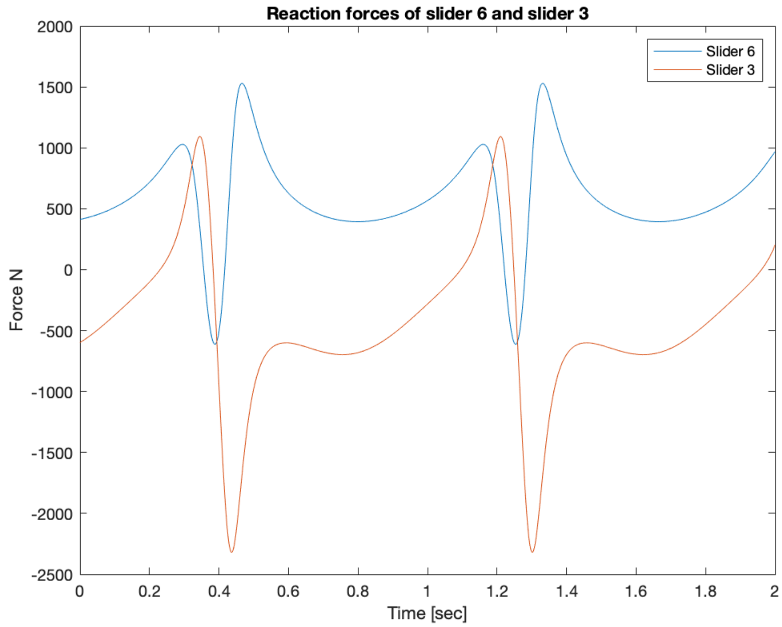

Figure 7: Reaction forces of slider 3 and slider 6.

We see how the normal force from slider 6 on the ground reacts, first, when crank to is moving towards $\pi/2$ the ground reaction force is increasing, meaning that the slider would push up if it wasn't attatched to the ground, then when crank 2 is moving downwards,i.e. from $\pi/$ to $3\pi/2$, the ground force becomes negative and positive again fast when we pass $3\pi/2$. This is expected, since when we move downwards with slider 3 the ground pushes upwards to the ground and we then move the slider 3 back up which means that the ground has to hold on to the slider 6. Similarly with slider the reaction force is just reaction to how slider 3 moves with regard to CoM of bar 4, i.e. when it moves towards it the reaction force is positive, i.e. it pushes slider 3 away, and when slider 3 moves away from CoM of bar 4 the reaction force pulls slider 3 towards it.

g)

To determine the period of motion T we used the `find` function in Matlab and looked when θ_2 became 2π. Then we used that index to find at what time that became true. This can be done in Matlab like the code below shows, where q_n are the positions and velocities of crank 2, rocker 4 and bar 5.

```
%% Find the period of the system

ff = find(q_n(:,1)>2*pi,1,'First');
period = tt(ff);                              % tt gives us the time at each time step
```

This gives us $T = 0.8780$.

h)

To see if my EoM was correct I made it also with the lagrange equations and compared it to the TMT method, this comparison showed that teh EoM was correct. To see if the coordinate projection method was working I checked to see if the constraints are actually zero, which was true and thus confirming that the results were correct. The results can also be checked by using the Coordinate Partitioning Method, so instead of cutting the closed loop we just use one generalized coordinate. This method allows us to find the acceleration and velocity mappings and from them calculate the CoM coordinates and compare to what we had with the coordinate projection method.

Appendix A

Matlab code

```
%% Set up EOM
syms alpha beta gamma
syms alphad betad gammad
syms alphadd betadd gammadd

% alpha = theta2; beta = theta4 = theta3; gamma = theta5

q=[alpha; beta; gamma];
qd=[alphad; betad; gammad];
qdd=[alphadd; betadd; gammadd];

% Given values
% Distance
O2A = 0.2;
O4B = 0.7;
BC = 0.6;
O4O2 = 0.3;
O4G4 = 0.4;
BG5 = 0.3;
yc = 0.9;

% Masses
m3 = 0.5;
m4 = 6;
m5 = 4;
m6 = 2;

% Mass moment inertia
J4 = 10;
J5 = 6;
J2 = 100;
J3 = 0;

% Forces
F = 1000;
T = 0;

% Inital velocities
omega = 75;

% Kinematics
xA = O2A*cos(alpha);
yA = O4O2+O2A*sin(alpha);
x4 = O4G4*cos(beta);
y4 = O4G4*sin(beta);
x5 = O4B*cos(beta)+BG5*cos(gamma);
y5 = O4B*sin(beta)+BG5*sin(gamma);
x6 = O4B*cos(beta)+BC*cos(gamma);

% Transformation matrix
Ti = [alpha; xA; yA; beta; x4; y4; beta; x5; y5; gamma; x6];   % note that ...
    theta3=theta4 so beta is used for both xA, yA and x4,y4

% Find Tij where xd = Tij*qd
Tij = jacobian(Ti,q);

% Find velocity
```

```
Tiv = Tij*qd;

% Find Tijk where xdd = Tij*qdd + Tijk*qd*qd
Tijk=zeros(11,3);

for i=1:3
    Tijk=Tijk+jacobian(Tij(:,i),q);
end

Tacc = jacobian(Tiv,qd)*qdd + jacobian(Tiv,q)*qd;

% Convective acceleration terms
gk = Tijk*(qd.*qd);



% Mass matrix
Mij = diag([J2 m3 m3 J3 m4 m4 J4 m5 m5 J5 m6]);

% Constraints
C6 = O4G4*sin(beta) + BC*sin(gamma) - yc;
CA = xA/yA - x4/y4;

C = [C6; CA];

Cd = jacobian(C,q);
Cd = simplify(Cd);
Cdd = jacobian(Cd*qd,q)*qd;
Cdd = simplify(Cdd);

% Applied forces, no gravity and external forces
Fi = zeros(11,1);
Fi(1) = T;
Fi(11) = F;

% Find reduced mass matrix
M = simplify(Tij.'*Mij*Tij);

Mbar = simplify([M Cd.';Cd zeros(2,2)]);

% Combined force matrix
Q = simplify(Tij.'*(Fi-Mij*gk));

Qbar = simplify([Q;-Cdd]);

Acc = Mbar\Qbar;

matlabFunction(Acc,'File','acchw7');
matlabFunction(C,'File','Chw7');
matlabFunction(Cd,'File','Cdhw7');

x6d = jacobian(x6,q)*qd;
x6dd = jacobian(x6d,qd)*qdd + jacobian(x6d,q)*qd;

matlabFunction(x6d,'File','x6dhw7');
matlabFunction(x6dd,'File','x6ddhw7');
matlabFunction(x6,'File','x6hw7');

%% Setup
time=2;
nn=13;
```

```
N=2.^nn;
h=time./N;

% Calculate initial angles and velocities
alpha = 0;
alphad = omega*2*pi/60;

beta = atan((O4O2+O2A*sin(alpha))/O2A*cos(alpha));
betad = O2A*alphad*cos(beta-alpha)/sqrt(O2A^2+O4O2^2);

gamma = pi-asin((yc-O4B*sin(beta))/BC);
%gammad = O4B*cos(beta)/(BC*sqrt(1-(yc-O4B*sin(beta))^2/BC^2));
gammad = 1.8433;

y0 = [alpha; beta; gamma; alphad; betad; gammad];
ang = [alpha; beta; gamma; alphad; betad; gammad];


% Setup for Gauss-Newton method
tol = 1e-12;
maxit = 10;
%% RK4 method with coordinate projection

for j=1:N
    [k1, force] = qa(y0);
    k2=qa(y0 + h/2*k1);
    k3=qa(y0 + h/2*k2);
    k4=qa(y0 + h*k3);
    qn= y0+1/6*h*(k1+2*k2+2*k3+k4);


%     % C(q_n+1)
%     tap = GNP(qn(1:3));
%     Dc = tap(4:5);      % Set C(q_n+1)

%     %Find corrected positions
%     i = 0;
%     while (max(abs(Dc) > tol) || (i < maxit))
%         tap = GNP(qn(1:3));
%         dp = tap(1:3);                       % Find error
%         qn(1:3) = qn(1:3) + dp;         % Corrected position
%         tap = GNP(qn(1:3));
%         Dc = tap(4:5);                  % Next C(q_n+1)
%         i = i+1;                         %counter
%     end

    % Find corrected velocities
    tep = GNV(qn);
    dcd = tep(4:5);
    dv = tep(1:3);
    qn(4:6) = qn(4:6) + dv;

    y0 = qn;
    q_n(j,:)=qn;
    force_all(j,:)=force;

    acceleration(j,:) = [k1(4);k1(5);k1(6)];

    % Velocity of slider 3 and rocker 4
    vx3 = -0.2*sin(qn(1))*qn(4);
    yx3 = 0.2*cos(qn(1))*qn(4);
```

```
    v3(j,:) = sqrt(vx3^2+yx3^2);

    vx4 = -0.4*sin(qn(2))*qn(5);
    vy4 = 0.4*cos(qn(2))*qn(5);
    v4(j,:) = sqrt(vx4^2+vy4^2);
    v3tov4 = v3 - v4;



end

%% Plot the data
% b) Plot the speeds of the crank, rocker and bar
% q_plot=[ang';q_n];
% figure(1);
% tt=0:h:time;
% plot(tt,q_plot(:,4)); hold on
% plot(tt,q_plot(:,5));
% plot(tt,q_plot(:,6));
% title('Speed of crank 2, rocker 4 and bar 5')
% xlabel('Time [sec]')
% ylabel('Speed rad/s')
% legend('Crank 2', 'Rocker 4', 'Bar 5')

% c)
% figure(2);
% plot(tt(2:end),v3tov4);
% title('Speed of slider 3 with respect to rocker 4')
% xlabel('Time [sec]')
% ylabel('Speed m/s')
% legend('Slider 3')



% % d)
% x6_pos = x6hw7(q_n(:,2),q_n(:,3));
% x6_vel = x6dhw7(q_n(:,5),q_n(:,2),q_n(:,6),q_n(:,3));
% x6_acc = ...
    x6ddhw7(q_n(:,5),acceleration(:,2),q_n(:,2),q_n(:,6),acceleration(:,3),q_n(:,3));
% figure(3);
% %plot(tt(2:end),x6_pos); hold on
% %plot(tt(2:end),x6_vel);
% plot(tt(2:end),x6_acc);
% title('Horizontal acceleration of slider 6')
% xlabel('Time [sec]')
% ylabel('Acceleration m/s^2')
% legend('Slider 6')


% e) and f)
figure(4)
plot(tt(2:end),force_all(:,1)); hold on
plot(tt(2:end),force_all(:,2));
title('Reaction forces of slider 6 and slider 3')
xlabel('Time [sec]')
ylabel('Force N')
legend('Slider 6', 'Slider 3')

%% Find the period of the system

ff = find(q_n(:,1)>2*pi,1,'First');
```

```
period = tt(ff);                          % 0.8780

%% Standard First-Order Form
function [acc, lambda] = qa(y)
alpha=y(1);
beta=y(2);
gamma=y(3);
alphad=y(4);
betad=y(5);
gammad=y(6);

acc = acchw7(alpha,alphad,betad,beta, gammad, gamma);

lambda = acc(4:5,1);

acc = [alphad; betad; gammad; acc(1,1); acc(2,1); acc(3,1)];


end

%% Gauus-Newton method for position coordinate projection
function ∆ = GNP(q)

alpha = q(1);
beta = q(2);
gamma = q(3);


D = Chw7(alpha,beta,gamma);
Dq = Cdhw7(alpha,beta,gamma);

∆ = [-Dq'*inv(Dq*Dq')*D; D];
end

%% Gauus-Newton method for velocity coordinate projection
function ∆_v = GNV(q)

alpha = q(1);
beta = q(2);
gamma = q(3);

Dq = Cdhw7(alpha,beta,gamma);

∆_v = [-Dq'*inv(Dq*Dq')*Dq*q(4:6); Dq*q(4:6)];
end
```