1) Program your constraints and Jacobian for the web cutter into a Newton-Raphson iterative algorithm to solve position kinematics for any desired crank angle. To check your work, when = 90 degrees, = 65.3 degrees and = -69.0 degrees.

2) Develop the velocity right-hand-side vector  for your constraints, and program them to solve for the generalized coordinate velocities. To check your work, when = 90 degrees, = 0.88 rad/s CW and = 0.72 rad/s CCW.

3) Develop the acceleration right-hand-side vector  for your constraints, and program them to solve for the generalized coordinate accelerations . To check your work, when = 90 degrees, = 14.2 rad/s/s CCW and = 10.1 rad/s/s CCW.

4) Place your position, velocity and acceleration algorithms within an outer loop to drive the crank through one complete revolution, and plot as a function of crank angle. Also plot the determinant of your Jacobian as a function of crank angle. Start the crank at = 0 degrees in your plots.

5) Determine the speed of the web.

VWEB \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**EXTRA CREDIT**

Modify your code to simulate an in-line slider crank with R = 4.00 cm and L = 14.23 cm operating at constant 60 rpm CCW crank speed. Validate your results by plotting piston acceleration as a function of crank angle compared to simple geometric equations.

% sc\_geometric.m - in-line slider crank geometric equations

% HJSIII, 14.02.22

clear

% constants

d2r = pi / 180;

% geometry and driver

R = 4; % crank length [inch]

L = 14.23; % connecting rod length [inch[

w = 60 \* 2\*pi / 60; % constant crank speed - 60 rpm CCW converted to rad/sec

% constant speed rotation driver motion

theta\_deg = ( 0 : 6 : 360 )';

theta = theta\_deg \* d2r;

st = sin(theta);

ct = cos(theta);

thetad = w \* ones( size( theta ) );

thetadd = zeros( size( theta ) );

% position

phi = asin( R \* sin(theta) / L );

sp = sin(phi);

cp = cos(phi);

phid = R\*thetad.\*ct ./ (L\*cp);

sd = -R\*thetad.\*st -L\*phid.\*sp;

phidd = ( R\*thetadd.\*ct -R\*thetad.\*thetad.\*st +L\*phid.\*phid.\*sp ) ./ (L\*cp);

sdd = -R\*thetadd.\*st -R\*thetad.\*thetad.\*ct -L\*phidd.\*sp -L\*phid.\*phid.\*cp;

figure( 1 )

 plot( theta\_deg, sdd, 'r' )

 axis( [ 0 360 -250 150 ] )

 title( 'Slider crank geometric solution - 60 rpm CCW' )

 xlabel( 'Crank angle [deg]' )

 ylabel( 'Piston acceleration [ipss]' )

% bottom of sc\_geometric