ME 581 – H03 **Name** _____

1) Download and view the web cutter video "wc.mov" from the class web page. Run MATLAB video analysis code "wc_dig.m" provided below. It will save the first video frame as image "wc_first_frame.jpg" and digitize x-y pixel locations for red and green dots on revolutes B and C respectively. Output will be provided in text file "wc_keep.txt".

Use Microsoft Paint to digitize column-row locations for the centers of revolutes A and D from "wc_first_frame.jpg". Remember to convert column-row locations to x-y pixel locations as shown at the bottom of the MATLAB code.

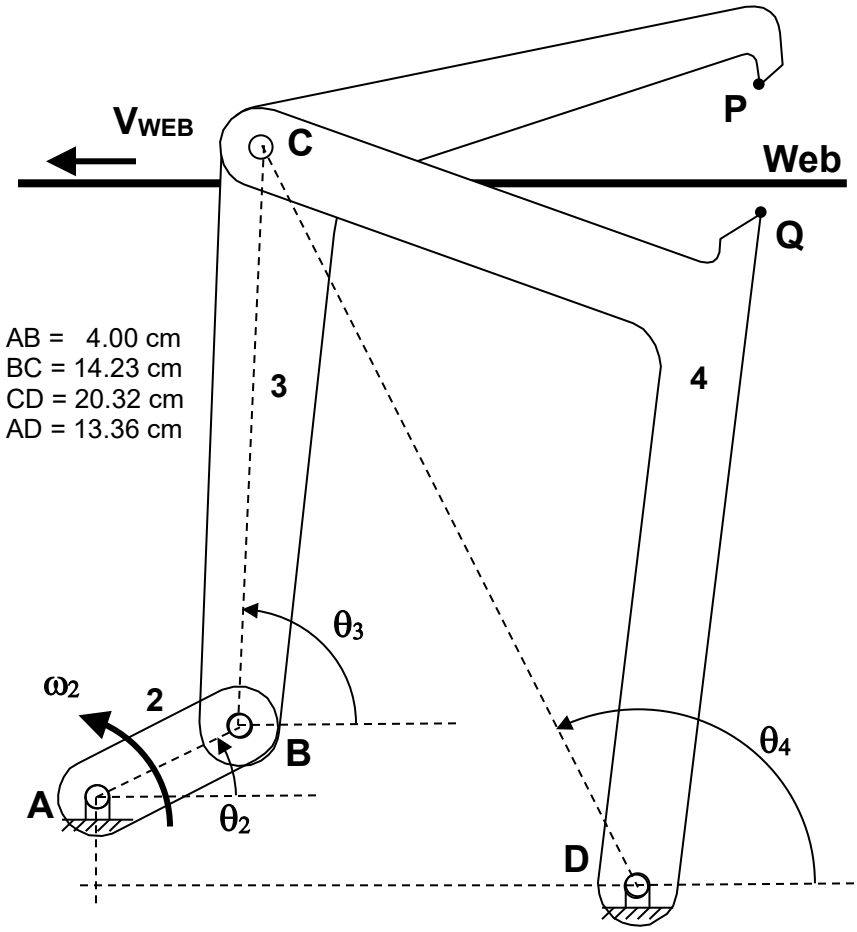
2) Plot angles θ_3 and θ_4 as functions of angle θ_2 as defined below.

3) Plot ω_2 as a function of time using simple finite difference and using Savitsky-Golay floating interpolants. Be certain to use units of rad/sec. Determine the mean value of ω_2 and root-mean-square (RMS) **about the mean value.**

mean ω_2 _____ RMS ω_2 about mean ω_2 _____

EXTRA CREDIT
Fit a circle to x-y data for revolute B and compare the center to your manual measurement.

Ax,Ay manual _____ Ax,Ay circle _____



```
% wc_dig.m - digitize red/green dots on web cutter from MOV video
% HJSIII, 19.12.03

clear

% video file name
fn_input = [ 'wc.mov' ];

% create video file reader object
VR_obj = VideoReader( fn_input );

% get video information
video_fps = VR_obj.FrameRate;          % frames per second
video_duration = VR_obj.Duration;      % sec
%video_frames = VR_obj.NumberOfFrames;  % must recreate object to rewind after using
NumberofFrames
%video_width = VR_obj.Width;
%video_height = VR_obj.Height;

% expected number of frames
nframe = round( video_duration * video_fps );
fps = round( video_fps );
disp( ' ' )
disp( [ 'reading ' num2str(nframe) ' frames recorded at ' num2str(fps) ' frames per second' ] )
disp( ' ' )
disp( 'see Figure 1' )
disp( ' ' )

% step through video
iframe = 0;
keep = [];
while hasFrame( VR_obj )
  a_rgb = readFrame( VR_obj );  % "readFrame" returns class uint8
  [ nr, nc, nk ] = size( a_rgb );
  iframe = iframe + 1;

% save first image
  if iframe==1,
    imwrite( a_rgb, 'wc_first_frame.jpg' )
  end

% convert to CIE L*a*b*
% L* intensity 0=dark, 100=bright - a_lab(:,:,1)
% a* green<0, red>0 - a_lab(:,:,2)
% b* blue<0, yellow>0 - a_lab(:,:,3)
a_lab = rgb2lab( a_rgb );  % size (nr,nc,3) - class double

% find red pixels
bw_r = ( a_lab(:,:,2) > 40 );  % size (nr,nc) - class logical

% find green pixels
bw_g = ( a_lab(:,:,2) < -30 );  % size (nr,nc) - class logical

% find centroid of one object in each black/white image
s_r = regionprops( bw_r, 'Centroid' );  % class structure
s_g = regionprops( bw_g, 'Centroid' );

% column and row stored in structure.Centroid
cr_r = s_r.Centroid;  % size (1,2) - class double
cr_g = s_g.Centroid;

% new figure
figure( 1 )
  clf
  warning( 'OFF', 'images:initSize:adjustingMag' )  % disbale warning for large images

% RGB image in UL
  subplot( 2, 2, 1 )
  imshow( a_rgb )
  title( [ 'frame ' num2str(iframe) ] )

% BW image for red in LL
  subplot( 2, 2, 3 )
  imshow( bw_r )
  title( 'red a*>40' )
```

```
  hold on
  plot( [ 0 cr_r(1) ], [ 0 cr_r(2) ], 'r' )  % line from origin to centroid

% BW image for green in LR
  subplot( 2, 2, 4 )
  imshow( bw_g )
  title( 'green a*<-30' )
  hold on
  plot( [ 0 cr_g(1) ], [ 0 cr_g(2) ], 'g' )  % line from origin to centroid

% update graphics
  drawnow

% save centroids
  keep = [ keep ; [ cr_r  cr_g ] ];

end  % bottom - while hasFrame

% row number increases in negative y direction
keep(:,2) = nr - keep(:,2);
keep(:,4) = nr - keep(:,4);

% show x-y results
figure( 2 )
  clf
  plot( keep(:,1),keep(:,2),'r', keep(:,3),keep(:,4),'g' )
  axis equal

% save to TXT file - x_red  y_red  x_green  y_green
save( 'wc_keep.txt', 'keep', '-ascii' )

% bottom - wc_dig
```

++

circle fit for unknown radius

$$(x-a)^2+(y-b)^2=r^2 \qquad (x^2+y^2)=\begin{bmatrix}1 & x & y\end{bmatrix}\begin{Bmatrix}r^2-a^2-b^2\\ 2a\\ 2b\end{Bmatrix}$$

$$\underset{\text{nobs x 1}}{\{Y\}}=\begin{Bmatrix}(x_1^2+y_1^2)\\ (x_2^2+y_2^2)\\ \vdots\\ (x_{nobs}^2+y_{nobs}^2)\end{Bmatrix} \qquad \underset{\text{nobs x 3}}{[X]}=\begin{bmatrix}1 & x_1 & y_1\\ 1 & x_2 & y_2\\ \vdots & \vdots & \vdots\\ 1 & x_{nobs} & y_{nobs}\end{bmatrix} \qquad \underset{\text{3 x 1}}{\{\beta\}}=\begin{Bmatrix}r^2-a^2-b^2\\ 2a\\ 2b\end{Bmatrix}$$

$$\{\beta\}=\left([X]^T[X]\right)^{-1}[X]^T\{Y\}$$

circle fit for known radius

$$\underset{\text{nobs x 1}}{\{Y\}}=\begin{Bmatrix}(x_1^2+y_1^2-r^2)\\ (x_2^2+y_2^2-r^2)\\ \vdots\\ (x_{nobs}^2+y_{nobs}^2-r^2)\end{Bmatrix} \qquad \underset{\text{nobs x 3}}{[X]}=\begin{bmatrix}1 & x_1 & y_1\\ 1 & x_2 & y_2\\ \vdots & \vdots & \vdots\\ 1 & x_{nobs} & y_{nobs}\end{bmatrix} \qquad \underset{\text{3 x 1}}{\{\beta\}}=\begin{Bmatrix}-a^2-b^2\\ 2a\\ 2b\end{Bmatrix}$$