HW 2 - ME 6406 Machine Vision

Table of Contents

by Cody Houff 9/28/21

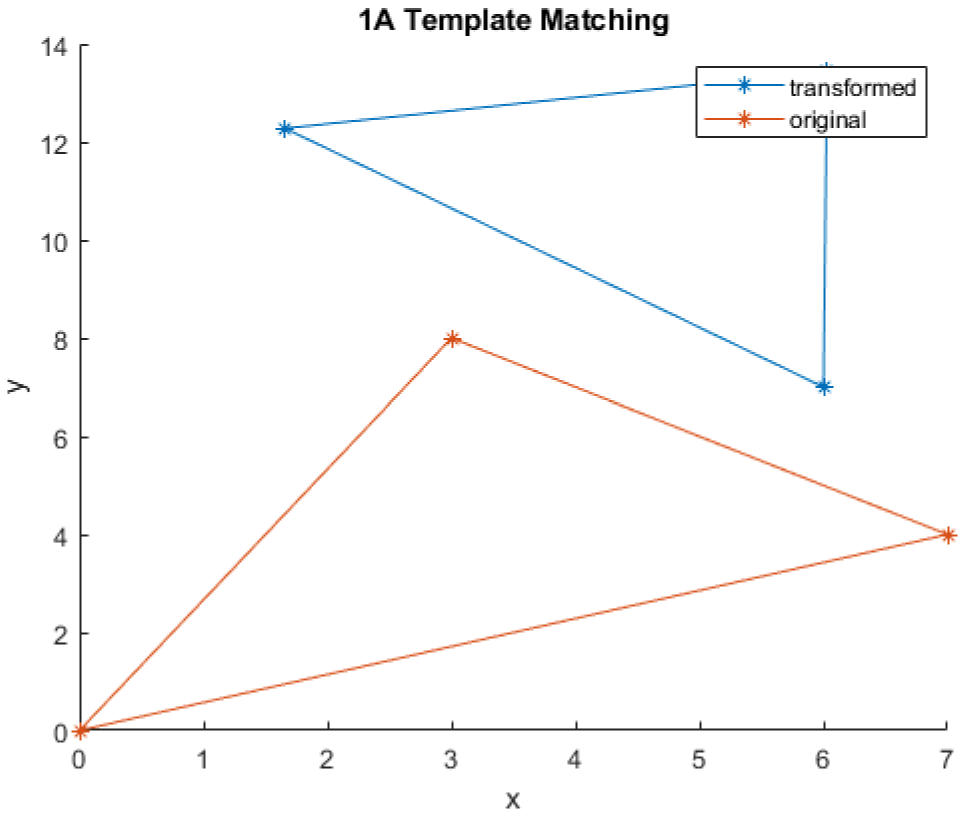
Problem 1a

Forward Transformation

```
clc
clear
close all

%Given
x_old = [0,7,3,0]
y_old = [0,4,8,0]

theta = (60 *pi) /180;
k = 0.8;
xd = 6;
yd = 7;
xc =0;
yc = 0;


x_new = k* cos(theta)* x_old - k*sin(theta)* y_old + xc +xd
y_new = k* sin(theta)* x_old + k*cos(theta)* y_old + yc +yd

figure()
hold on
title('1A Template Matching')
xlabel('x'); ylabel('y');
plot(x_new(:), y_new(:),'*-')
plot(x_old(:), y_old(:),'*-')
legend('transformed', 'original')
```

```
x_old =

     0     7     3     0
```

```
y_old =

     0     4     8     0


x_new =

    6.0000    6.0287    1.6574    6.0000


y_new =

    7.0000   13.4497   12.2785    7.0000
```



Problem 1b

```
%Find k, theta, xd, yd
clc
clear
close all

xy_old = [0,7,3;0,4,8]'
xy_new =[6.0000,6.0287,1.6574; 7.0000,13.4497,12.2785]'
```

```
xc = 0;
yc = 0;

m =1;
for n= 1:3:3


A = [xy_old(n,m),    -xy_old(n,m+1),   1, 0;
     xy_old(n,m+1),   xy_old(n,m),     0, 1;
     xy_old(n+1,m),  -xy_old(n+1,m+1), 1, 0;
     xy_old(n+1,m+1), xy_old(n+1,m),   0, 1];

R = [xy_new(n,m),xy_new(n,m+1),xy_new(n+1,m),xy_new(n+1,m+1)]';

Q = inv(A'*A)*A'*R;

k = sqrt(Q(1)^2+Q(2)^2)
theta = atand(Q(2)/Q(1))
xd = Q(3) - xc
yd = Q(4) - yc
end


xy_old =

     0     0
     7     4
     3     8


xy_new =

    6.0000    7.0000
    6.0287   13.4497
    1.6574   12.2785


k =

    0.8000


theta =

   60.0002


xd =

     6


yd =
        7
```

Problem 1c

```
%triangles
clc
clear
close all


points = nchoosek([1 2 3 4 5],3)';
t_points = points(:);
x_old_list = [2 6 8 5 -3]';
y_old_list = [0 2 6 8 5]';

xy_old = [x_old_list(t_points),y_old_list(t_points)];
x_old = x_old_list(t_points);
y_old = y_old_list(t_points);

%align the triangle points correctly, in the right order
for n = 1:3:size(t_points,1)-2

    t_point1 = t_points(n);
    t_point2 = t_points(n+1);
    t_point3 = t_points(n+2);

    L_12 = sqrt((x_old(n)-x_old(n+1))^2+((y_old(n)-y_old(n+1))^2));
    L_23 = sqrt((x_old(n+1)-x_old(n+2))^2+((y_old(n+1)-y_old(n
+2))^2));
    L_13 = sqrt((x_old(n)-x_old(n+2))^2+((y_old(n)-y_old(n+2))^2));
    L_sort = sort([L_12, L_23, L_13],'descend');

        if (L_sort(1) == L_12)
            %disp("length_12")
            new_t_point3 = t_point3; %let 3 <- 3

            a = xy_old(n+2,:); %outside point
            b = xy_old(n,:);
            c = xy_old(n+1,:);
            C = cross([b-a,0],[c-a,0]);
            %C(3)
            if (C(3)>0)
                new_t_point1 = t_point1; %let 1 <- 1, b
                new_t_point2 = t_point2; %let 2 <- 2, c
            else
                new_t_point1 = t_point2; %let 1 <- 2, c
                new_t_point2 = t_point1; %let 2 <- 1, b
            end

        elseif (L_sort(1) == L_23)
            %disp("length_23")
            new_t_point3 = t_point1; %let 3 <- 1

            a = xy_old(n,:); %outside point
            b = xy_old(n+1,:);
            c = xy_old(n+2,:);
            C = cross([b-a,0],[c-a,0]);
            %C(3)
            if (C(3)>0)
                new_t_point1 = t_point2; %let 1 <- 2, b
                new_t_point2 = t_point3; %let 2 <- 3, c
            else
                new_t_point2 = t_point3; %let 2 <- 3, c
                new_t_point1 = t_point2; %let 1 <- 2, b
            end

        else
            %disp("length_13")
            new_t_point3 = t_point2; %let 3 <- 2

            a = xy_old(n+1,:); %outside point
            b = xy_old(n,:);
            c = xy_old(n+2,:);
            C = cross([b-a,0],[c-a,0]);
            %C(3)
            if (C(3)>0)
                new_t_point1 = t_point1; %let 1 <- 1, b
                new_t_point2 = t_point3; %let 2 <- 3, c
            else
                new_t_point1 = t_point3; %let 1 <- 3, c
                new_t_point2 = t_point1; %let 2 <- 1, b
            end
        end

    new_t_points(n) = new_t_point1;
    new_t_points(n+1) = new_t_point2;
    new_t_points(n+2) = new_t_point3;
end

new_t_points = new_t_points'; %correctly sorted triangle points

xy_old_sorted = [x_old_list(new_t_points),y_old_list(new_t_points)];
x_old_sorted = x_old_list(new_t_points);
y_old_sorted = y_old_list(new_t_points);


x_goal = [2.28, 10.621, 9.545]';
y_goal = [16.28, 10.318, 15.576]';
xy_goal = [x_goal,y_goal];

xc = 0;
yc = 0;
z=1;
for n= 1:3:28
```

```
A = [x_old_sorted(n),    -y_old_sorted(n),   1, 0;
     y_old_sorted(n),   x_old_sorted(n),     0, 1;
     x_old_sorted(n+1),  -y_old_sorted(n+1), 1, 0;
     y_old_sorted(n+1), x_old_sorted(n+1),   0, 1];


    R = [x_goal(1),y_goal(1),x_goal(2),y_goal(2)]';

    Q = inv(A'*A)*A'*R;

    k = sqrt(Q(1)^2+Q(2)^2);
    theta = atan2d(Q(2),Q(1));
    xd = Q(3) - xc;
    yd = Q(4) - yc;

    x_old_changed(n) = k* cosd(theta)* x_old_sorted(n) -
 k*sind(theta)* y_old_sorted(n) + xc +xd;
    x_old_changed(n+1) = k* cosd(theta)* x_old_sorted(n+1) -
 k*sind(theta)* y_old_sorted(n+1) + xc +xd;
    x_old_changed(n+2) = k* cosd(theta)* x_old_sorted(n+2) -
 k*sind(theta)* y_old_sorted(n+2) + xc +xd;

    y_old_changed(n) = k* sind(theta)* x_old_sorted(n) +
 k*cosd(theta)* y_old_sorted(n) + yc +yd;
    y_old_changed(n+1) = k* sind(theta)* x_old_sorted(n+1) +
 k*cosd(theta)* y_old_sorted(n+1) + yc +yd;
    y_old_changed(n+2) = k* sind(theta)* x_old_sorted(n+2) +
 k*cosd(theta)* y_old_sorted(n+2) + yc +yd;

    E1 = sqrt((x_old_changed(n)-x_goal(1))^2+(y_old_changed(n)-
y_goal(1))^2);
    E2 = sqrt((x_old_changed(n+1)-x_goal(2))^2+(y_old_changed(n+1)-
y_goal(2))^2);
    E3 = sqrt((x_old_changed(n+2)-x_goal(3))^2+(y_old_changed(n+2)-
y_goal(3))^2);
    E = E1 + E2 + E3;

   table(z,:)= [E,xd,yd,k,theta,new_t_points(n),new_t_points(n
+1),new_t_points(n
+2)];%,x_old_changed(n),y_old_changed(n),x_old_changed(n
+1),y_old_changed(n+1),x_old_changed(n+2),y_old_changed(n+2)];
   z = z+1;
end

sorted_table = sortrows(table);
Latency_Table = array2table(sorted_table);
Latency_Table.Properties.VariableNames =
 ["Error","xd","yd","k","theta","#1","#2","#3"] %,"x1f","y1f","x2f","y2f","x3f","y

best_triangle = array2table(sorted_table(1,:));
best_triangle.Properties.VariableNames =
 ["Error","xd","yd","k","theta","#1","#2","#3"] %,"x1f","y1f","x2f","y2f","x3f","y

'4 1 2, triangle is the best match'
```

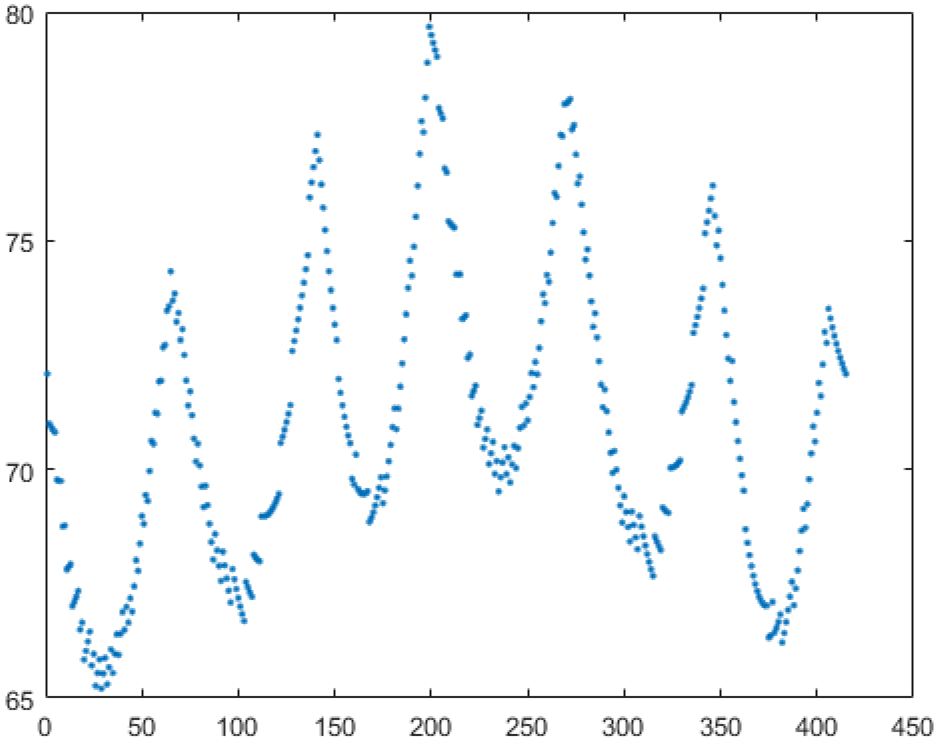
```
Latency_Table =

  10×8 table

      Error          xd          yd           k          theta      #1     #2
    #3
    __________    ______    _______    _______    _______    __     __
    __

    0.00046816    9.9998     7.9998        1.2         75     4      1
    2
        1.4162    11.541     15.652    0.92824     139.25     3      5
    1
        2.0416    1.3599     10.946    0.92824    -40.751     5      3
    4
        2.2803    11.018     7.9342     1.2083     99.443     3      1
    2
        2.3002    5.8505    0.78476     1.6855     44.981     4      2
    3
        2.3219    3.7877     10.161     1.0807    -17.122     5      2
    4
        2.9116    9.1133     16.437     1.0807     162.88     2      5
    1
        4.1366    11.541     15.652    0.92824     139.25     3      5
    2
        4.9472    2.9012     18.598        1.2       -105     1      4
    5
        5.3661    9.9998     7.9998        1.2         75     4      1
    3


best_triangle =

  1×8 table

      Error          xd          yd        k      theta     #1     #2     #3
    __________    ______    ______    ___    _____    __     __     __

    0.00046816    9.9998    7.9998    1.2     75       4      1      2


ans =

    '4 1 2, triangle is the best match'
```

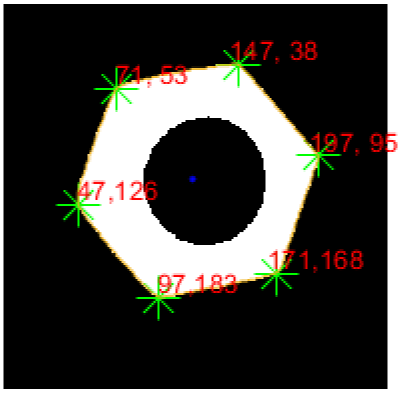
Problem 2a

rho-theta signature

```
clc
clear
```

```
close all

image = imread('HW2.png');
bw_image = rgb2gray(image); %grayscale

threshold = 50/255; %255 because it is an 8 bit image

bi_image = ~imbinarize(bw_image, threshold); %this makes a binary
 image and the "~" creates a white object with black background

store_centroid = regionprops(bi_image,'centroid'); %calculates
 centroid

centroid = cat(1,store_centroid.Centroid); %obtaining the centroid
 values

cx = centroid(1);
cy = centroid(2);

bound = bwboundaries(bi_image, 8, 'noholes');

rho = [];
theta = [];
for j = bound{1}(:,1)
    for i = bound{1}(:,2)
        theta = atan2d(j-cx,i-cy);
        rho = sqrt((i-cx).^2 + (j-cy).^2);
    end
end


figure()
plot(1:length(rho), rho, '.')

figure()
[Peaks,Loc] = findpeaks(rho,'MinPeakHeight', 73, 'MinPeakDistance',
 20);

imshow(bi_image);
hold on
plot(centroid(:,1),centroid(:,2),'b.'); %plot the centroids

y = bound{1}(:,1);
x = bound{1}(:,2);
scatter(x(Loc), y(Loc),250,'g*');
plot(x,y);
text(x(Loc)-5, y(Loc)-10, strcat(int2str(x(Loc)), ',',
 int2str(y(Loc))),'Color','red');
hold off
```





Problem 2b

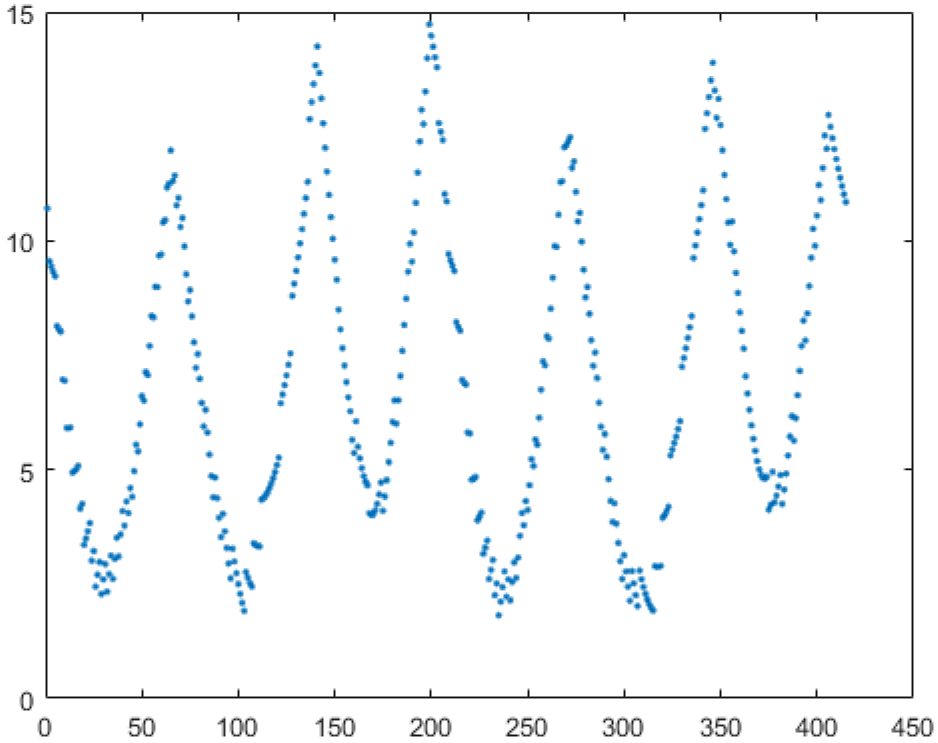
median length

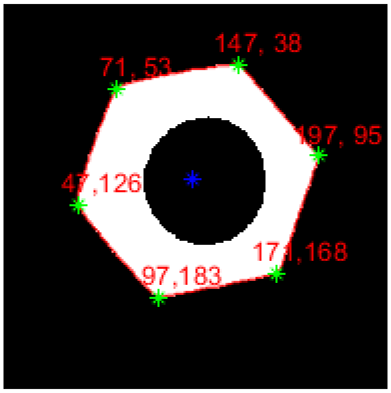
```
clc
clear
close all
image = imread('HW2.png');
```

```
bw_image = rgb2gray(image); % converting the image to greyscale

threshold = 50/255;
bin_image = imcomplement(im2bw(bw_image,threshold)); % binarizing the
 image

store_centroid2 = regionprops(bin_image, 'centroid'); % find the
 centroid

centroid2 = cat(1,store_centroid2.Centroid);
cx = centroid2(1);
cy = centroid2(2);

bound = bwboundaries(bin_image, 8, 'noholes');
y = bound{1}(:,1);
x = bound{1}(:,2);

edge = [x y];

G = zeros(length(x),1); % an empty vector for storing values

dist = 50; % the average distance of points on each side

for n = 1:length(edge)

    % a matrix of the indices that to be compared
    comp = n-dist:1:n+dist;
    % wrapping around indices
    comp(comp<1) = comp(comp<1)+length(edge);
    comp(comp>length(edge)) = comp(comp>length(edge))-length(edge);

    % mean position on each side
    point_ax = mean(x(comp(1:dist)));
    point_ay = mean(y(comp(1:dist)));

    point_bx = mean(x(comp(dist+2:end)));
    point_by = mean(y(comp(dist+2:end)));

    point_x = x(n);
    point_y = y(n);

    % the main equations for the median of a triangle
    c = sqrt((point_ax-point_bx)^2 + (point_ay-point_by)^2);
    b = sqrt((point_x-point_bx)^2 + (point_y-point_by)^2);
    a = sqrt((point_ax-point_x)^2 + (point_ay-point_y)^2);

    G(n) = 1/2*sqrt(2*a^2 + 2*b^2 - c^2);

end
figure()
plot(1:length(G), G, '.')

% the 6 most prominent peaks
```

```
[pks, locs, w, p] = findpeaks(G);
pknum = size(pks,1);
[pksnew, sortlocs] = sort(p, 'descend');

locs = locs(sortlocs);
locs = locs(1:6);

figure()
imshow(bin_image)

hold on
plot(cx,cy,'b*')
plot(x, y, 'r')
scatter(x(locs), y(locs), 'g*')
text(x(locs)-15, y(locs)-15, strcat(int2str(x(locs)), ', ',
 int2str(y(locs))), 'Color', 'red');

corners = edge(locs,:)
```

```
corners =

   197    95
   171   168
   147    38
    97   183
    71    53
    47   126
```





Problem 2c

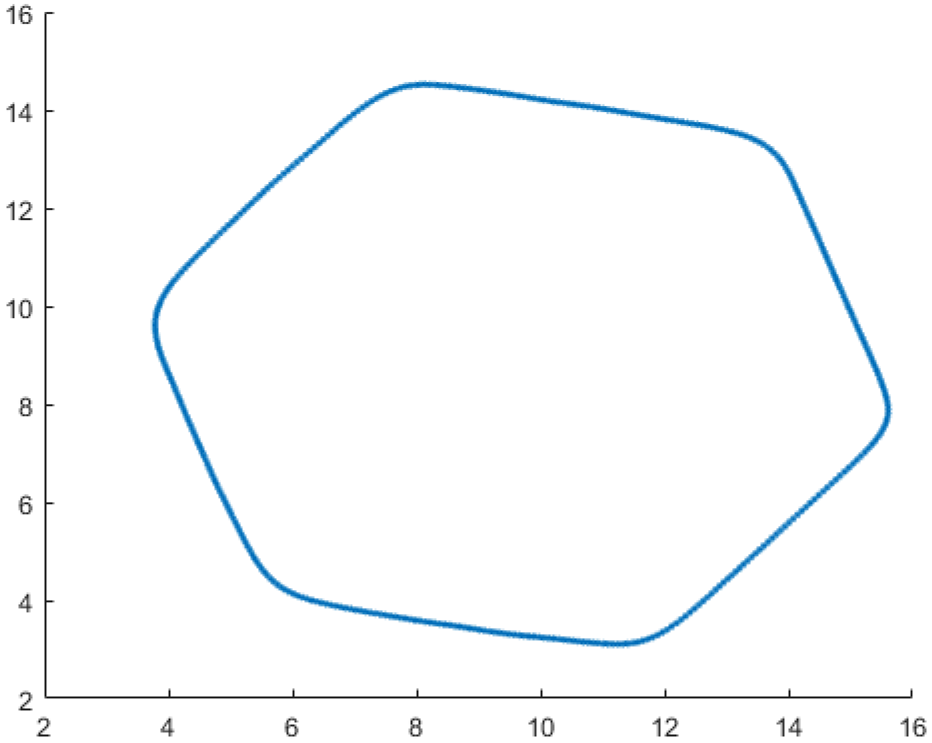
curvature

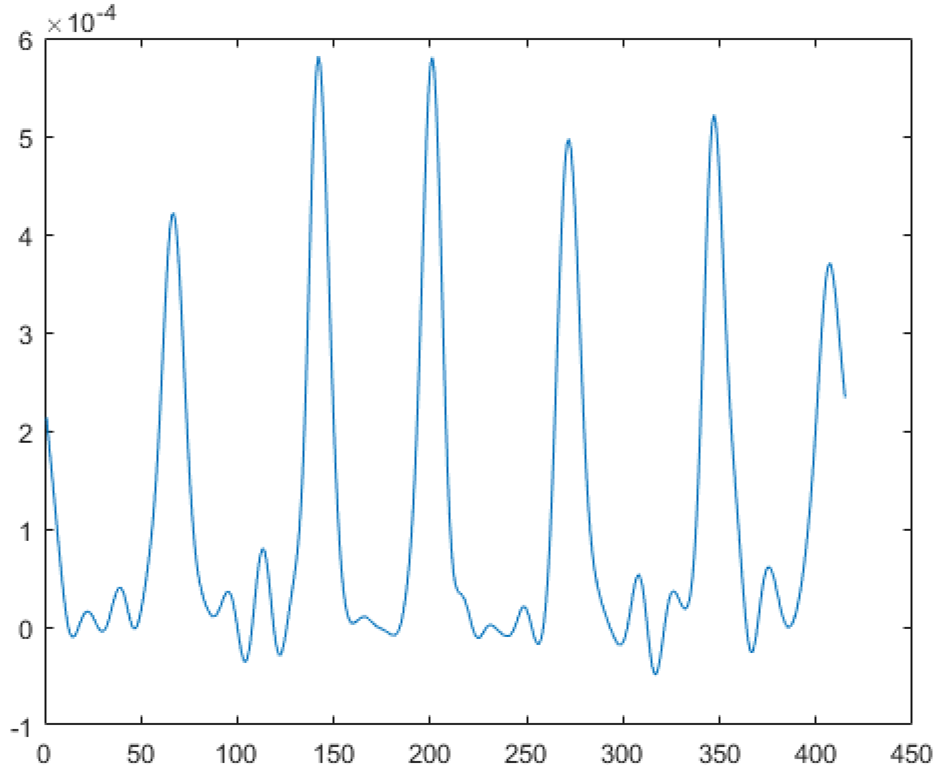
```
clc
clear
close all
```

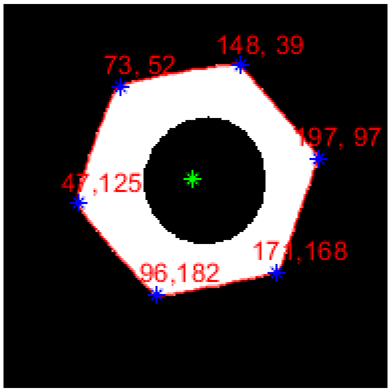
```
image = imread('HW2.png');

image_bw = rgb2gray(image); % convert the image to greyscale

threshold = 50/255;
image_bin = imcomplement(im2bw(image_bw,threshold)); % binarize the
 image

store_centroid3 = regionprops(image_bin, 'centroid'); %find the
 centroid

centroid = cat(1,store_centroid3.Centroid);
cx = centroid(1);
cy = centroid(2);

bound = bwboundaries(image_bin, 8, 'noholes');
y = bound{1}(:,1);
x = bound{1}(:,2);

edge = [x y];

% sigma value for the gaussian filter
sigma = 5; % sigma value
mask_size = 2*3*sigma + 1; % mask size

% gaussian kernel
store_centroid3 = -mask_size:mask_size;
gauss = (1/(2*pi*sigma^2))*exp(-store_centroid3.^2/(2*sigma^2));

% adding extra to the vectors with values from the opposite end
x_extra = [x(end-mask_size:end); x; x(1:mask_size)];
y_extra = [y(end-mask_size:end); y; y(1:mask_size)];

% convolute with the gaussian kernel
convolute_x = conv(x_extra, gauss, 'same');
convolute_y = conv(y_extra, gauss, 'same');
figure()

% subtract the extra
x_conv_values = convolute_x(mask_size:end-mask_size);
y_conv_values = convolute_y(mask_size:end-mask_size);
scatter(x_conv_values, y_conv_values, '.')

% calculate the curvature
xdot = diff(x_conv_values);
ydot = diff(y_conv_values);
xddot = diff(xdot);
yddot = diff(ydot);

c = xdot(1:end-1).*yddot - ydot(1:end-1).*xddot;

figure()
plot(1:length(c), c)
```

```
% find the 6 most prominent peaks
[peaks, locations, w, p] = findpeaks(c);
pknum = size(peaks,1);
[newpeaks, locations_sorted] = sort(p, 'descend');

locations = locations(locations_sorted);
locations = locations(1:6);
corners = edge(locations,:)

figure()
imshow(image_bin)

hold on
plot(cx,cy,'g*')
plot(x, y, 'r')
scatter(x(locations), y(locations), 'b*')
text(x(locations)-15, y(locations)-15,
 strcat(int2str(x(locations)), ', ',
 int2str(y(locations))), 'Color', 'red');
```

```
corners =

   148    39
   197    97
    96   182
   171   168
    73    52
    47   125
```


16
14
12
10
8
6
4
2
2
4
6
8
10
12
14
16


$\times 10^{-4}$
6
5
4
3
2
1
0
-1
0
50
100
150
200
250
300
350
400
450



Problem 3a

Equation See attached paper

Problem 3b

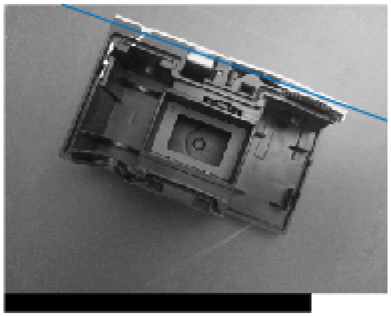
Hough Transform

```
clc
clear
clear tabs

img = imread('Camera.png');

img = im2double(img);

I = rgb2gray(img);

BW=edge(I,'log',[],3);


hx = [-1,-2,-1;
      0, 0, 0;
      1, 2, 1];

hy = [-1,0, 1;
      -2, 0, 2;
      -1, 0, 1];

Gx = imfilter(I,hx);
Gy = imfilter(I,hy);

%round
```

```
%figure(2)
%imshow(Gx)

%figure(3)
%imshow(Gy)

%figure(4)
%imshow(BW)

k=1;
G = zeros(700);

[length, width] = size(BW);
for i = 1:length
 for j = 1:width
  if BW(i,j) == 1


            v  = (i*Gx(i,j)+j*Gy(i,j))/(Gx(i,j)^2+Gy(i,j)^2);

            x0 = v*Gx(i,j);
            y0 = v*Gy(i,j);

            theta = atand(y0/x0);
            p = sqrt(x0^2+y0^2);
            a = -x0/y0;
            b = p/sind(theta);

            table(k,:)= [x0,y0,a,b,p,theta]; %


            y0 = round(y0);
            x0 = round(x0);

            G(x0+300,y0+300) = G(x0+300,y0+300)+1;

            k = k+1;
        end
    end
end



%Latency_Table = array2table(table);
%Latency_Table.Properties.VariableNames =
 ["x0","y0","a","b","p","theta"]

% figure(5)
% imshow(G)
% figure(6)
% surf(G)
%shading interp
maxval = max(max(G));
```

```
[x0_max,y0_max] = find(G==maxval);

x0_max = x0_max-300;
y0_max = y0_max-300;

A = -x0_max/y0_max;
theta = atan2d(y0_max,x0_max);
P = sqrt(x0_max^2+y0_max^2);
b = P/sind(theta);


hold on
imshow(I);
xline = [0, width]
yline = A*xline+b
line(yline,xline)
xlim([0 width])
%ylim([0 height])
```

```
xline =

     0   300


yline =

   44.2000  884.2000
```



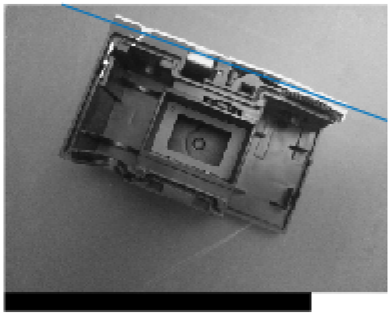
Problem 3c

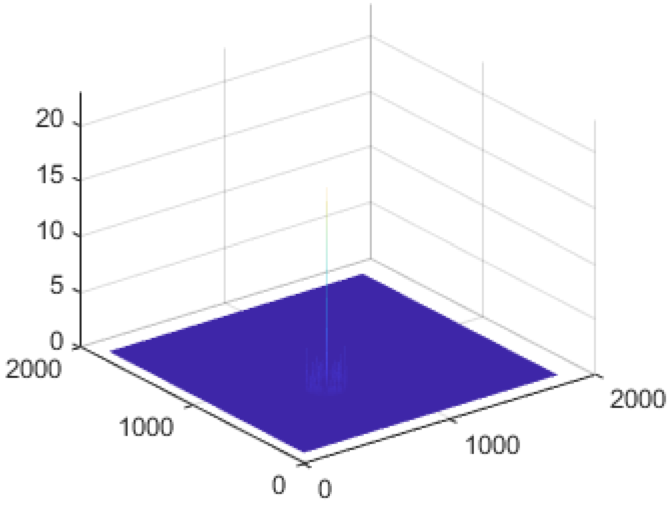
Hough Transform

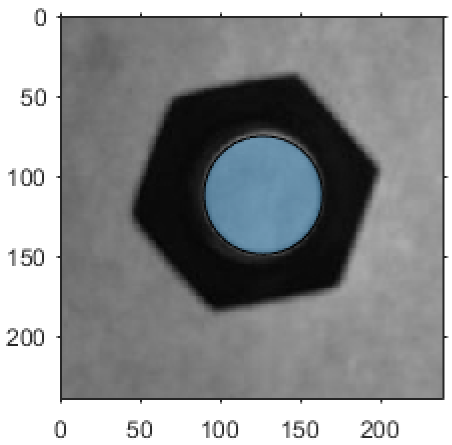
```
clc
clear
clear tabs

image = imread('HW2.png');

image = im2double(image);

image_grey = rgb2gray(image);


hx = [-1,-2,-1;
      0, 0, 0;
      1, 2, 1];

hy = [-1,0, 1;
      -2, 0, 2;
      -1, 0, 1];

figure()
edge = edge(image_grey,'log',[],3);
imshow(edge)

gx = imfilter(image_grey, hx);
gy = imfilter(image_grey, hy);

Gmag = sqrt(gx.^2 + gy.^2);

[height, width] = size(image_grey);

disp("a")
r_range = 1:2:width;
offset = 500;
accum = (zeros ( width+3*offset, width+3*offset, r_range(end))+1 );
for i = 1:height
    for j = 1:width
        if (edge(i,j)~=0)
            for r = r_range

            c_theta = gx(i,j)/Gmag(i,j);
            s_theta = gy(i,j)/Gmag(i,j);

            x0 = round(i + r*c_theta); % main equations
            y0 = round(j + r*s_theta); % main equations

            accum(x0+offset, y0+offset, r) =
 accum(x0+offset,y0+offset, r)+1;

            end
        end
    end
end
```

```
[max_accum, idx_accum] = max(accum(:))
[x0, y0, r0] = ind2sub(size(accum), idx_accum);

x0 = x0 - offset
y0 = y0 - offset
r0

surf(accum(:,:,r0(1)))
shading interp

figure()
imshow(image_grey)
hold on
axis on

p = nsidedpoly(1000, 'Center', [y0 x0], 'Radius', r0); % circle
 plotting function
plot(p)
xlim([0 width])
ylim([0 height])
```

```
a

max_accum =

    23


idx_accum =

   110083452


x0 =

   112


y0 =

   127


r0 =

    37
```




20
15
10
5
0
2000
1000
0
0
1000
2000



Published with MATLAB® R2021a