CSE 5524 – Homework #6
10/07/2013
Manjari Akella

Acknowledgement

- ellipse.m file(used in Q2) was taken from File Exchange at the MATLAB Central http://www.mathworks.com/matlabcentral/fileexchange/289-ellipse-m by David Long
  This function was used to plot the circular patch (when length of major and minor axes are same, ellipse=circle)

1) Use the covariance matching technique to find the correct match in the color image given on the WWW site (target.jpg). The model covariance matrix (of <col,row,R,G,B> features) is given below.
modelCovMatrix =
[47.917 0 -146.636 -141.572 -123.269;
0 408.250 68.487 69.828 53.479;
-146.636 68.487 2654.285 2621.672 2440.381;
-141.572 69.828 2621.672 2597.818 2435.368;
-123.269 53.479 2440.381 2435.368 2404.923];
Test all possible overlapping windows (each of size 70 rows by 24 columns, with the upperleft-corner as the window origin) in the image with the given model. Save the match distance for each box location in the image at each pixel location (for the origin of the window). Plot/display the match-distance-image. Provide the (row,col) location of the best match distance for the best candidate.

- The best score (minimum) was 0.706076. This minima can be seen in Figure 1 (the valley in blue)
- The (r,c) location of the best candidate was (26,255)

Output

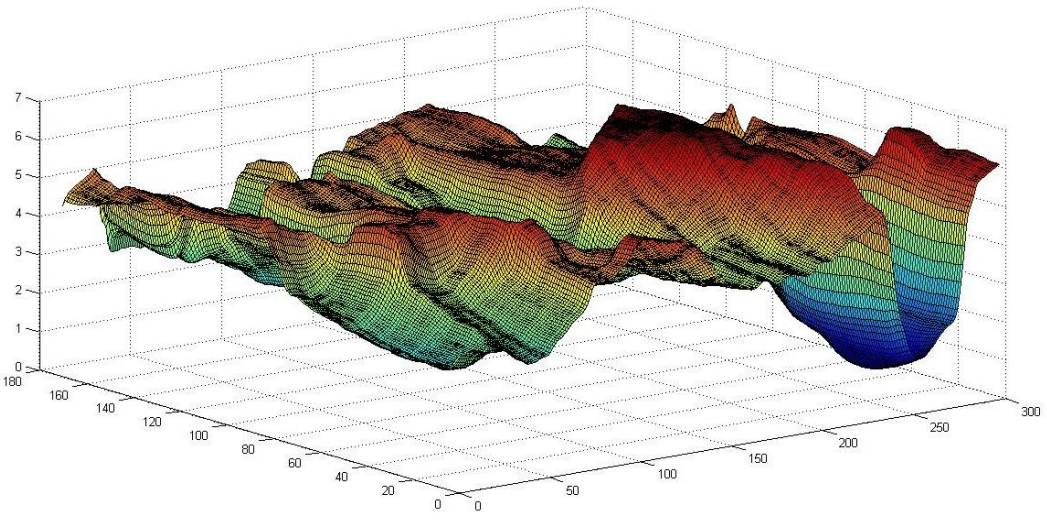

Figure 1: Score Matrix surface plot

Figure 2: Best Match Window

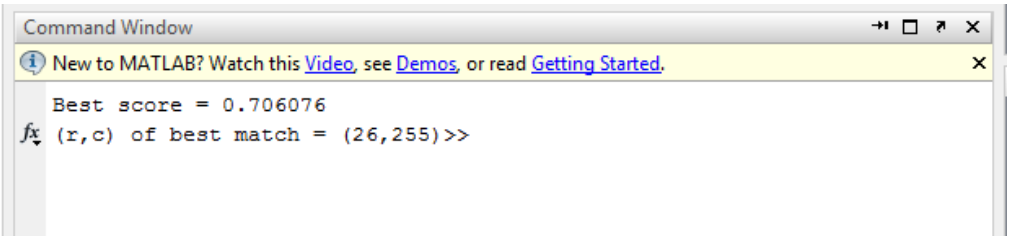

Figure 3: Best match score and coordinates

2) Load the images img1.jpg and img2.jpg and perform mean-shift tracking. Build a model from img1 using a circular neighborhood with a radius of 25 pixels centered at (x0,y0) = (150,175) and a color histogram of size 16x16x16=4096 bins (cube). Quantize each color channel into 16 bins of {0-15, 16-31, …, 240-255}. Build the weighted "cube" histogram using an Epanechnikov kernel with bandwidth h = 25 and a neighborhood of all pixels inside a circle with a radius of 25 pixels. Run 25 iterations of mean-shift tracking. Only round coordinates when indexing into images to build the models (histograms).Report the final (x,y) location and Euclidean distance between the last two iterations (see Step 4 on the Algorithm slide).

- We start at [150,175] in the first image. This corresponds to the red car
- The car moves a few pixels to the left in the new image while there is *almost* no movement in the y direction. So we would expect the final coordinates of y to remain *almost* the same, which is exactly what was observed
- The final x coordinate decreased by about 8 pixels while there was a *very small* change in the y coordinate
- If we consider the double precision (6 decimal points), after about 16-17 iterations, Euclidean distance between consecutive iterations was of the order 10^{-4}, which is very small. So we could probably have used this as a threshold and needn't have performed 25 iterations
- If we consider the precision upto 10 decimal points, the final Euclidean distance was of the order 10^{-7}

Output

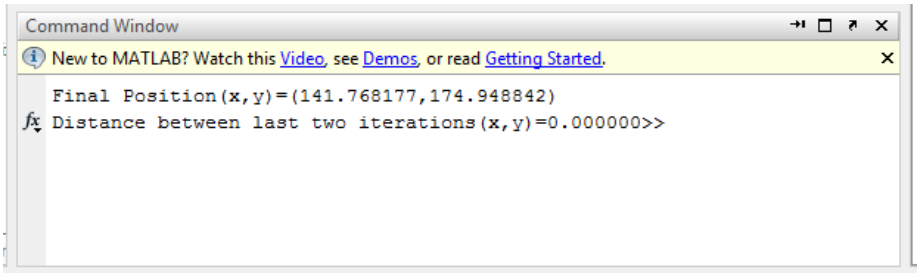

Figure 4: Final Position and Euclidean distance (precision=%f)

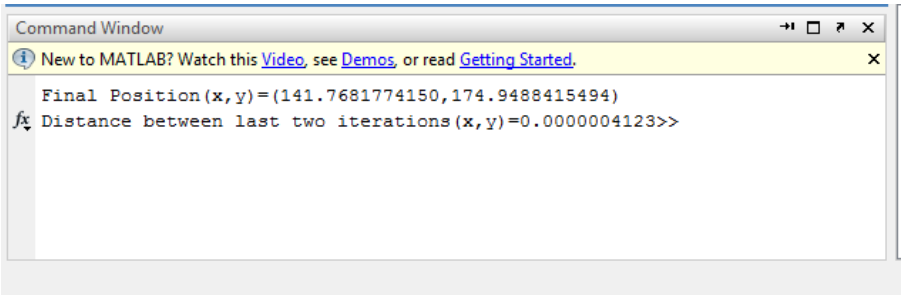

Figure 5: Final Position and Euclidean distance (precision=%.10f)

Figure 6: Frame 1 target patch

Figure 7: Frame 2 initial patch (blue), final patch (red)

CODE

1). compare.m (for Q1)

```
function [ score ] = compare( modelCovMatrix,candCovMatrix )
% Takes the model and candidate covariance matrices and returns
% a score value representing the similarity between the two
[~,Eval] = eig(modelCovMatrix,candCovMatrix);
for i=1:size(Eval,1)
    for j=1:size(Eval,2)
        if(Eval(i,j)==0)
            lEvalsq(i,j)=0;
        else
            lEvalsq(i,j) = (log(Eval(i,j)))^2;
        end
    end
```

```
end
lEvalsum = sum(sum(lEvalsq));
score = sqrt(lEvalsum);
end
```

2). circRegion25.m (for Q2)

```
function [ info ] = circRegion25( I,center )
% Takes an image and center coordinates of terget patch as input
% and returns a matrix info having (c,r,R,G,B)values of pixels in
% a 25 radius neighbourhood
n=1;
for i=1:size(I,1)
    for j=1:size(I,1)
        if(sqrt((i-center(1,2))^2+(j-center(1,1))^2)<=25)
            info(1,n) = j;
            info(2,n) = i;
            info(3,n) = I(i,j,1);
            info(4,n) = I(i,j,2);
            info(5,n) = I(i,j,3);
            n=n+1;
        end
    end
end

end
```

3). computeHist.m (for Q2)

```
function [ q ] = computeHist( info,center)
% Takes the info matrix and center coordinates of the patch
% and returns a histogram distribution cube (16x16x16) of the
% patch under consideration
q_u = zeros(16,16,16);
countR = zeros(1,16);
countG = zeros(1,16);
countB = zeros(1,16);
% Model q_u for n channel
for i=1:size(info,2)
    j1 = ceil(info(3,i)/16);
    j2 = ceil(info(4,i)/16);
    j3 = ceil(info(5,i)/16);
    countR(1,j1) = countR(1,j1)+1;
    countG(1,j2) = countG(1,j2)+1;
    countB(1,j3) = countB(1,j3)+1;
    x = [info(1,i),info(2,i)];
    % Unrounded center
    t = ((center-x)./25);
    t1 = (sqrt(t(1,1)^2+t(1,2)^2))^2;
    q_u(j1,j2,j3) = q_u(j1,j2,j3)+functionK(t1);
end
% Normalize
q = q_u./(sum(sum(sum(q_u))));
end
```

4). computeWeight.m (for Q2)

```
function [ w ] = computeWeight( q_u,p_u,info )
% Takes 2 probability histograms and the info matrix as inputs
% Returns a weight value for each pixel in info
for i=1:size(info,2)
    j1 = ceil(info(3,i)/16);
    j2 = ceil(info(4,i)/16);
    j3 = ceil(info(5,i)/16);
    w(1,i) = sqrt(q_u(j1,j2,j3)/p_u(j1,j2,j3));
end
end
```

5). functionK.m (for Q2)

```
function [ k_r ] = functionK( r )
% The Epanechnikov kernel with r(normalized radius) as input
% Returns k(r)
    if (r<=1)
        k_r = 1-r;
    else
        k_r = 0;
    end
end
```

6). HW6.m script

```
% Manjari Akella
% CSE5524 - HW6
% 10/07/2013

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 1
clear all;
close all;
clc;
% Covariance matrix of model
modelCovMatrix = [47.917 0 -146.636 -141.572 -123.269;
0 408.250 68.487 69.828 53.479;
-146.636 68.487 2654.285 2621.672 2440.381;
-141.572 69.828 2621.672 2597.818 2435.368;
-123.269 53.479 2440.381 2435.368 2404.923];
% Read image
I = double(imread('given_pics/target.jpg'));
r=1;
cfk=zeros(5,5);
while ((r+69)<=240)
    c=1;
    while ((c+23)<=320)
        % Window of 70x24
        Im = I(r:r+69,c:c+23,1:3);
        n=1;
        % Compute feature vector for window
        for i=1:size(Im,1)
```

```
            for j=1:size(Im,2)
                fk(1,n) = j;
                fk(2,n) = i;
                fk(3,n) = Im(i,j,1);
                fk(4,n) = Im(i,j,2);
                fk(5,n) = Im(i,j,3);
                n=n+1;
            end
        end
        % Compute covariance
        candCovMatrix  = cov(fk');
        % Compute score with model matrix
        score = compare(modelCovMatrix,candCovMatrix);
        % Store score value
        scoreMatrix(r,c) = score;
        c=c+1;
    end
    r=r+1;
end
bestMatch = min(min(scoreMatrix));
[m,ind] = min(scoreMatrix(:));
[x,y] = ind2sub(size(scoreMatrix),ind);
figure('Name','Q1: Score Matrix Surface
Plot','NumberTitle','off'),surf(scoreMatrix);
figure('Name','Q1: Best Match Window','NumberTitle','off'),imagesc(uint8(I));
axis('image');
hold on;
rectangle('Position',[y,x,24,70],'LineWidth',2,'EdgeColor',[0.498039, 1, 0]);
hold off;
fprintf('Best score = %f',bestMatch);
fprintf('\n(r,c) of best match = (%d,%d)',x,y);
pause;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 2
clear all;
close all;
clc;
% Read first frame image
I1 = double(imread('given_pics/img1.jpg'));
center = [150,175];
figure('Name','Q2: Frame 1, Target
Model','NumberTitle','off'),imagesc(uint8(I1));
axis('image');
hold on;
ellipse(25,25,0,150,175,'b');
hold off;
[info] = circRegion25(I1,center);
% Compute histogram for R,G,B channels
[q_u] = computeHist(info,center);
% New frame image
I2 = double(imread('given_pics/img2.jpg'));
center1  = center;
figure('Name','Q2: Frame 2, Target Candidate','NumberTitle','off');
imagesc(uint8(I2));
axis('image');
hold on;
```

```
ellipse(25,25,0,center(1,1),center(1,2),'b');
num=0;
di = zeros(25,1);
for i=1:25
    % Compute weights
    [info] = circRegion25(I2,round(center1));
    [p_y] = computeHist(info,center1);
    [w] = computeWeight(q_u,p_y,info);

    % Find next best location
    num(1,1) = sum(info(1,:).*w);
    num(1,2) = sum(info(2,:).*w);
    den = sum(w);
    new_center = num./den;
    % Distance between consecutive iterations
    di(i,1) = sqrt((center1(1,1)-new_center(1,1))^2 + (center1(1,2)-
new_center(1,2))^2);
    % Set center1 = new_center
    center1 = new_center;
    % Plot the new patch
    h = ellipse(25,25,0,round(center1(1,1)),round(center1(1,2)),'r');
    pause(0.5);
    % If not the last iteration, delete circle plot
    if(i~=25)
        delete(h);
    end
end
hold off;
% Final position
fprintf('Final Position(x,y)=(%.10f,%.10f)',center1(1,1),center1(1,2));
% fprintf('Final Position(x,y)=(%f,%f)',center1(1,1),center1(1,2));

% Distance between last 2 iterations
fprintf('\nDistance between last two
iterations(x,y)=%.10f',di(size(di,1),1));
% fprintf('\nDistance between last two iterations(x,y)=%f',di(size(di,1),1));

%%%%%%%%%%%%%%%%%%%%%% END %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```