assignmentSegmentation2

by Mayank Singhal

Submission date: 02-Sep-2018 11:53PM (UTC+0800)
Submission ID: 995961305
File name: allthecode.txt (5.76K)
Word count: 800
Character count: 4619

```
function [I, Ix, Iy, eig1, eig2, corner] = myHarrisCornerDetector(inputImage,
smoothsigma, weightsigma, k)
    % constants
    window_size = 5;
    % Rescaled Image
    I  = inputImage.imageOrig;
    I  = I./255;
    % size
    sizex = size(I,1);
    sizey = size(I,2);
    % Sobel operators
    sobelx = [-1 0 1; -2 0 2; -1 0 1];
    sobely = [1 2 1; 0 0 0; -1 -2 -1];
```

```
% Image Derivatives

Ix = filter2(sobelx, I);

Iy = filter2(sobely, I);

% Masks

smooth = fspecial('gaussian', window_size, smoothsigma);

weights = fspecial('gaussian', window_size, weightsigma);

% smooth images

sIx = filter2(smooth, Ix);

sIy = filter2(smooth, Iy);

sIx2 = sIx.*sIx;

sIxIy = sIx.*sIy;

sIy2 = sIy.*sIy;

% Padded smooth Images
```

```
psIx2 = padarray(sIx2, [2, 2], 0, 'both');

psIy2 = padarray(sIy2, [2, 2], 0, 'both');

psIxIy = padarray(sIxIy, [2, 2], 0, 'both');

% eigen values matrix

eig1 = zeros(size(I));

eig2 = zeros(size(I));

% Cornerness measure

corner = zeros(size(I));

% Implementation

for i = 1:sizex

    for j = 1:sizey

        % Structure Tensor

        M = zeros(2,2);
```

```
            M(1,1) = sum(sum(weights.*psIx2(i:i+4, j:j+4)));

            M(1,2) = sum(sum(weights.*psIxIy(i:i+4, j:j+4)));

            M(2,1) = sum(sum(weights.*psIxIy(i:i+4, j:j+4)));

            M(2,2) = sum(sum(weights.*psIy2(i:i+4, j:j+4)));

            determinant = det(M);

            corner(i,j) = determinant - k*trace(M)*trace(M);

            eigen = eig(M);

            eig1(i,j) = max(eigen);

            eig2(i,j) = min(eigen);

        end

    end

    corner = corner > 0.01;

endtic;
```

```
%% MEAN SHIFT SEGMENTATION

% * First, as required by the question, the image is convolved with a

% gaussian filter of standard deviation = 1

% * Then the image is resized to half the original size

% * Then Mean Shift Segmentation is performed on the image

% * Then the image is Linear Contrast Stretched to the range [0,255]

% * The given function returns the number of segments

%% PARAMETERS FOR THE FUNCTION

% * Number of iterations used in the algorithm = iterations

% * The bandwidth of the spatial kernel = hr

% * The bandwidth of the color kernel = hs

% * image = I

%% FIXED PARAMTERS IN THE ALGORITHM
```

```
% * The size of the gaussain mask for filtering is 7

% * The number of Nearest Neighbours chosen is 100

%% baboonColor.png

% * hr = 8

% * hs = 16

% * iterations = 25

image = imread('../data/baboonColor.png');

[segments, FI] = myMeanShiftSegmentation(image,25,8,16);

%%

imshow(image), colorbar;

title('Original');

%%

imshow(FI), colorbar;
```

```
title('After Mean Shift Segmentation');

fprintf('Number of Segments = %d',segments);

toc;tic;

%% Assignment 3:    Question 1:

% This script performs harris corner detection

I = load('../data/boat.mat');

k = 0.04;

smoothingSigma = 1.33;

weightsSigma = 1.33;

[I, Ix, Iy, eig1, eig2, corner] = myHarrisCornerDetector(I, smoothingSigma, weightsSigma,
k);

%%

imagesc(I);
```

```
title('Original Image');

colormap gray;

colorbar;

%%

imagesc(Ix);

title('Derivative of image along x axis');

colormap gray;

colorbar;

%%

imagesc(Iy);

title('Derivative of image along y axis');

colormap gray;

colorbar;
```

```
%%
imagesc(eig1);
title('Max Eigen values');
colormap gray;
colorbar;
%%
imagesc(eig2);
title('Min Eigen Values')
colormap gray;
colorbar;
%%
imagesc(corner);
title('Harris Cornerness Measure');
```

```
colormap gray;

colorbar;

%% Optimal Values:

fprintf('%s%f₩n%s%f₩n%s%f₩n','Weights gaussian standard deviation: ',weightsSigma,

'Gaussian for smoothing standard deviation: ', smoothingSigma, 'Tuned value of k is ',k);

toc;function [sz_unir , FI] = myMeanShiftSegmentation(I,iterations,hr,hs)

% first we need to create a 5d mapping of the matrix so that we can do the

% mean shift segmentation on those vectors. Each vector is i,j,r,g,b. For

% that we need to do something similar to row order storage of the indexes.

% create a 2d matrix of 5 rows and each column represents a 5d vecor in

% space.

gf = fspecial('gaussian',7,1);

filtered_img = imfilter(I,gf,'conv'); % image after convolution
```

```
RI = filtered_img(1:2:end,1:2:end,:); % this is the resized image

[m, n, d] = size(RI);

row_order_stored = zeros(5,m*n);

for i = 1:m

    for j = 1:n

        row_order_stored(:,((m-1)*i) + j) = [i;j;RI(i,j,1);RI(i,j,2);RI(i,j,3)];

    end

end

row_order_stored(1:2,:) = bsxfun(@rdivide,row_order_stored(1:2,:),hr);

row_order_stored(3:5,:) = bsxfun(@rdivide,row_order_stored(3:5,:),hs);

row_trans = row_order_stored';

updated_vectors = row_trans; % this is the initial value. This gets modified after ecah

iteration
```

```
for i = 1:iterations

    [indices, distances] = knnsearch(row_trans,row_trans,'k',100);

    for v = 1:m*n

        w = exp((-1/2)*(distances(v,:).^2));% this is a vector of the weights fom 1 to n(here
my n = 100)

        denom_w = sum(w); % this is the denominator of the derivative of log(f(x))

        % for the numerator take dot product of this weight vector with

        % each of the n(=100) closest vectors

        w = w';

        w = repmat(w,1,5);

        numerator_w = sum(w.*row_trans(indices(v,:),:));

        updated_vectors(v,:) = numerator_w/denom_w;

    end
```

```
    row_trans = updated_vectors; % sir mentioned in the class to update only after ech
iteration
                              % and not after update of each vector
end
% now to get the output back in the 2d form for each colour
% so map back the row ordered indices into its corresponding 2d indices
for i = 1:m
    for j= 1:n
        segmented_img(i,j,1:3) = row_trans(((m-1)*i) + j,3:5);
    end
end
%converting into uint8
segmented_img = uint8(segmented_img);
```

```
% contrast stretching the image to [0,255]

for i=1:d

    mx = max(max(segmented_img(:,:,i)));

    mn = min(min(segmented_img(:,:,i)));

    FI(:,:,i) = (segmented_img(:,:,i)-mn)*(255/(mx- mn));

end

uni = unique(FI);

[sz_unir, ~] = size(uni);

end
```

assignmentSegmentation2

ORIGINALITY REPORT

7%	5%	3%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Sheffield Student Paper	2%
2	www.osapublishing.org Internet Source	2%
3	Submitted to Rochester Institute of Technology Student Paper	1%
4	mostazafin.tv Internet Source	1%
5	www.it.uu.se Internet Source	1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

assignmentSegmentation2

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14