16D100012_16D070001_16D170005_Assignment3_Segmentation

by Sarthak Consul

Submission date: 03-Sep-2018 02:24AM (UTC+0800)

Submission ID: 995994991

File name: segmentaiton_report.pdf (114.84K)

Word count: 1087

Character count: 5861

Question 1

```
%% MyMainScript for Harris Corner Detection
% By Sarthak Consul (16D100012)
%
% -- Parthsarathi Khirwadkar (16D070001)
%
% -- Bhishma Dedhia (16D170005)
tic;
%% Loading Input
storedStructure = load('../data/boat.mat');
input = storedStructure.imageOrig;
input=im2double(input)/255;
figure
imagesc(input);
colormap(gray);
title('Input');
daspect([1,1,1]);
axis tight;
colorbar;
%% Parameters
k = 0.06;
%
% Gaussian Filter applied before computing gradients
sigma_pre = 1.2; % (so filter size of 4x4)
%
patch_size = 15; % so, gaussian weights  = patch_size/6
%% Harris Corner Detection
output = myHarrisCornerDetector(input, k,patch_size,sigma_pre);

my_imshow(output,{'Output Image',strcat('k=',num2str(k),' patch\_size=',
num2str(patch_size),' sigma\_pre=', num2str(sigma_pre))});

%% Note on tuning
% - Tuning og sigma\_pre affects on how accurate gradients are found. Too
% small results in peaky gradients due to noise and too large removes edges
% entirely
%
% - k must lie between 0 and 0.25. k is chosen to result in positive
cornerness at corner points, negative
% cornerness at edge points and zero cornerness at uniform regions
% - Non max suppression is essential to pick the true corner from a local
% cluster
%
% - Patch\_size decides how large a area must the corner point satify
% 'cornerness' property. If it is too small, errant points would be
% identified as corners. If it is too large, notion of a corner is lost to
% the patch
toc;



function output = myHarrisCornerDetector(input, k, patch_size,sigma_pre)
% Input arguements
```

```
% input (of range 0-1)
% k (should be between 0 and 0.25) - Tunable
% patch_size = size of patch about which structure tensor is computed
% sigma_pre = std. deviation of Gaussian Filter applied before computing
gradients
    [row, col] = size(input(:,:,1));
    %Same padding to ensure gradients at edge of image don't blow up and
    %rescale the colourbars
    input =  padarray(input, [patch_size,patch_size],'replicate', 'both');
    %smoothen image with gaussian to reduce noise before computing
    %gradients
    sm_input = imgaussfilt(input,sigma_pre);

    %Finding Ix
    filtx1 = [1;2;1];
    filtx2 = [-1 0 1];
    Ix = imfilter(sm_input,filtx2,'conv');
    Ix = imfilter(Ix,filtx1,'conv');

    %Finding Iy
    filty1 = [1;0;-1];
    filty2 = [1 2 1];
    Iy = imfilter(sm_input,filty2,'conv');
    Iy = imfilter(Iy,filty1,'conv');

    % Convolve about isometric patch to get structure tensor elements
    iso_G = fspecial('gaussian',patch_size, patch_size/6);
    Ix2 = imfilter(Ix.*Ix,iso_G,'conv');
    Iy2 = imfilter(Iy.*Iy,iso_G,'conv');
    Ixy = imfilter(Ix.*Iy,iso_G,'conv');
    st = patch_size+1;
    en = size(sm_input,1)-patch_size;

    input = input(st:en,st:en);

    a = cat(3, Ix2(st:en,st:en), Ixy(st:en,st:en), Iy2(st:en,st:en) );
    % detA = a1.a3-a2^2, trace(A) = a1+a3
    P = a(:,:,1).*a(:,:,3) - a(:,:,2).*a(:,:,2); %detA is product of eigvals
    S = a(:,:,1) + a(:,:,3); %trace(A) is sum of eigvals

    eigval = zeros(row,col,2);
    eigval(:,:,1) = (S + sqrt(S.*S - 4*P) )/2;
    eigval(:,:,2) = (S - sqrt(S.*S - 4*P) )/2;
    my_imshow(eigval(:,:,1),{'Eigen Value-1',strcat('k= ',num2str(k),'
patch\_size=', num2str(patch_size),' sigma\_pre=', num2str(sigma_pre))});
    my_imshow(eigval(:,:,2),{'Eigen Value-2',strcat('k= ',num2str(k),'
patch\_size=', num2str(patch_size),' sigma\_pre=', num2str(sigma_pre))});

   cornerness = P - k*S.*S;
   corners = cornerness == ordfilt2(cornerness,25,ones(5,5)) & cornerness >
1e-4; %small margin to ensure robustness
   outputR = 0.3*input;
   output = 0.3*input(:,:,[1,1]);
   outputR(corners) = 1;
   output = cat(3,outputR,output);
```

```
   my_imshow(cornerness,{'Cornerness',strcat('k= ',num2str(k),'
patch\_size=', num2str(patch_size),' sigma\_pre=', num2str(sigma_pre))});

  end

function my_imshow(varargin)

    numberColours = 200;
%     colorScale = [[0:1/(numberColours-1):1]',[0:1/(numberColours-
1):1]',[0:1/(numberColours-1):1]'];
    figure('units','normalized','outerposition',[0 0 1 1])

    num = nargin/2;
    for k = 1:num
        subplot(1,num,k);
        imagesc(varargin{2*k-1});
        title(varargin{2*k}, 'Fontsize', 12);
        % truesize;
        colormap(jet);
%         caxis([zMin, zMax]);
        daspect([1,1,1]);
        axis tight;
        colorbar;

    end
end
```

Question 2

```
%% MyMainScript for Mean Shift Segmentation
% By Sarthak Consul (16D100012)
%
% -- Parthsarathi Khirwadkar (16D070001)
%
% -- Bhishma Dedhia (16D170005)
%% Optimum Parameters:
%   Intensity bandwidth = 60
%   Spatial bandwidth = 5
%   Iterations = 20
%   K in K-NN = 300 to 350
input = imread('../data/baboonColor.png');
H_i = 60;
H_s = 5;

output = myMeanShiftSegmentation(input,H_i(i),H_s(j),20);
 %% Results
figure(1)
% title('Input Image')
input = imresize(input,0.5);
my_imshow(input,'Input Image');

figure(2)
my_imshow(output,'Output Image');
imshow(output);
```

```
%%  Plotting scatter plot
figure(3)
title('Scatter plot of input image')
[row_ip, col_ip] = size(input(:,:,1));
li = double(reshape(input,row_ip*col_ip,3))/255;
scatter3(li(:,1), li(:,2), li(:,3),ones(row_ip*col_ip,1), li);

figure(4)
title('Scatter plot of output image')
[row_op, col_op] = size(output(:,:,1));
lo = double(reshape(output,row_op*col_op,3))/255;
scatter3(lo(:,1), lo(:,2), lo(:,3),ones(row_op*col_op,1), lo);

%% Notes on Implementation
% Following the seminal paper of Mean Shift Segmentation, we realised that
% the update rule is not strictly Newton-Raphson in the sense


function output = myMeanShiftSegmentation(ip, h_i, h_s, iter)
    %Mean Shift segmentation on downsampled image
        ip = imgaussfilt(double(ip),1.5);  % apply gaussian smoothening to
the image to reduce noise
        ip = imresize(ip,0.5);            % downsampling image for faster
computation
        [row, col] = size(ip(:,:,1));
        indexX = repmat((1:1:row)' ,1,col);
        indexY = repmat(1:1:col ,row, 1);
        space = cat(3, ip, indexX,indexY); %RGBXY space RxCx5

        for i = 1:iter
%              converting to list for finding k nearest neighbours
            list = reshape(space,row*col,5); %Nx5

%              randomly generating k
            rng(i)
            k = (randi([300,350],1));
%              indices of k-nn for each pixel
            [I,~] = knnsearch(list,list,'k',k);

%              calculating difference between pixel value and its neighbours
            diff_nn = permute(reshape(list(I',:),
k,row*col,5),[1,3,2]);%kx5xN
            diff_nn = diff_nn - repmat(diff_nn(1,:,:),k,1); % diff_nn = x-x_i

%              square of difference
            dist_nn = diff_nn.*diff_nn;

%              calculating gaussian kernel at each pixel for intensity and
%              space
            exp_nn_i = exp(-sum(dist_nn(:,1:3,:),2)/ h_i^2); %kx1xN
            exp_nn_s = exp(-sum(dist_nn(:,4:5,:),2)/ h_s^2); %kx1xN

%              calculating denominator
            den = repmat(exp_nn_i.*exp_nn_s,1,5,1); %kx5xN
%              calculating numerator
```

```
            grad = diff_nn.*den; %kx5xN

            den(1,:,:) = 0;
            den = sum(den,1); %1x5xN
%             calculating final normalised gradient
            grad = sum(grad,1)./ den; %1x5xN
            grad = permute(grad, [3,2,1]); %Nx5
            grad = reshape(grad, row, col, 5); %RxCx5
%             updating the pixel values
            space = space + grad;

        end
    output = double(space(:,:,1:3))/255;
end
```

16D100012_16D070001_16D170005_Assignment3_Segmenta...

ORIGINALITY REPORT

7%	1%	2%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Indian Institute of Technology, Bombay Student Paper	5%
2	He, Harry. "Algorithms", Coding Interviews, 2012. Publication	1%
3	Enrico Negrisolo, Massimiliano Babbucci, Tomaso Patarnello. "The mitochondrial genome of the ascalaphid owlfly Libelloides macaronius and comparative evolutionary mitochondriomics of neuropterid insects", BMC Genomics, 2011 Publication	1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

16D100012_16D070001_16D170005_Assignment3_Segmentatic