Toonification of grayscale image (14uec080,14uec109)

Table of Contents

Anisotropic Diffusion

```
clear all;
close all;
clc;
im=imread('img.jpg');
im=rgb2gray(im);

%          im     - input image
%          niter  - number of iterations.
%          kappa  - conduction coefficient 20-100 ?
%          lambda - max value of .25 for stability
%          option - 1 Perona Malik diffusion equation No 1
%                   2 Perona Malik diffusion equation No 2
niter=6;
kappa=100;
lambda=0.2;
option =2;
diff=im;
diff = anis_diff(im, niter, kappa, lambda, option);

figure;
imagesc(diff), colormap(gray), axis image% axis none
ax = gca
ax.Visible = 'off'
colormap(gray);
f=getframe; imwrite(f.cdata,'test.png');
%saveas(gcf,'figgur.png')
%figure;
%e = canny(diff);
%imshow(e);
close all;
```

```
ax =

  Axes with properties:
```

```
               XLim: [0.5000 220.5000]
               YLim: [0.5000 299.5000]
             XScale: 'linear'
             YScale: 'linear'
      GridLineStyle: '-'
           Position: [0.1300 0.1100 0.7750 0.8150]
              Units: 'normalized'

  Use GET to show all properties


ax =

  Axes with properties:

               XLim: [0.5000 220.5000]
               YLim: [0.5000 299.5000]
             XScale: 'linear'
             YScale: 'linear'
      GridLineStyle: '-'
           Position: [0.1300 0.1100 0.7750 0.8150]
              Units: 'normalized'

  Use GET to show all properties
```

Edge Detection

```
% Input Image
clear all;
clc;
A=[];
piA=[];

%Using 16 fuzzy edge templets that show the possible direction of the
 edges
%in the image and then calculating the divergence between the original
%image and the 16 fuzzy templets.

III = rgb2gray(imread('test.png'));%name of the image
II = imcrop(III,[5 5 560 450]);
I = double(II);
[r,k] = size(I);

% Selection of the 16 fuzzy templets
a=0.3; b=0.8;
t1 = [a a a; 0 0 0; b b b];
t2 = [a a b; a b 0; b 0 0];
t3 = [b b b; 0 0 0; a a a];
t4 = [b a a; 0 b a; 0 0 b];
t5 = [b a 0;b a 0; b a 0];
t6 = [a 0 b;a 0 b; a 0 b];
```

```
t7 = [0 0 0; b b b; a a a];
t8 = [0 b a; 0 b a; 0 b a];
t9 = [a a a; b b b;0 0 0];
t10 = [a b 0; a b 0;a b 0];
t11 = [0 0 0; a a a;b b b];
t12 = [0 a b; 0 a b; 0 a b];
t13 = [b b b; a a a; 0 0 0];
t14 = [b 0 a; b 0 a; b 0 a];
t15 = [b 0 0; b 0 a; a a b];
t16 = [0 0 b; 0 b a; b a a];

% Initization of algorithm
xmax = max(max(max(I))); %maximum pixel/element of the image;
%converting into the fuzzy domain from the original image;
fim = I/xmax;%fim is the image data of the input image in the fuzzy
 domain
%all values of fim are in the interval of [0 1];
%initializing the edge image as zeros matrix
fedgeim = zeros(r,k);%in fuzzy domain
%Increaing the border line of the image i.e to increase the row and
 column
%by 2 in the first and last by taking the mirror image of the
 immediate
%existing rows and columns respectively
r1 = fim(2,:);%Copy of all element in the 2nd row of fim
r2 = fim(r-1,:);
c1 = fim(:,2);
c2 = fim(:,k-1);
b1 = [0 r1 0];
b2 = [0 r2 0];
b3 = [c1 fim c2];
bfim = [b1;b3;b2];%bfim = Border fuzzy image matix
bfim(1,1) = fim(1,1);
bfim(r+2, k+2) = fim(r,k);
bfim(1,k+2) = fim(1,k);
bfim(r+2,1) = fim(r,1);


%finding Hesitation degree or intuitionstic fuzzy index
%c = input("Enter the value of pi  ");
c= 0.2;
pibfim = c*(1-bfim);
pit1 = c*(1-t1);pit2 = c*(1-t2);pit3 = c*(1-t3);pit4 = c*(1-t4);pit5 =
 c*(1-t5);pit6 = c*(1-t6);pit7 = c*(1-t7);
pit8 = c*(1-t8);pit9 = c*(1-t9);pit10 = c*(1-t10);pit11 = c*(1-
t11);pit12 = c*(1-t12);pit13 = c*(1-t13);
pit14 = c*(1-t14);pit15 = c*(1-t15);pit16 = c*(1-t16);

%Calculation of the maximum of the divergance value between the 16
 templates
%and the original image of the same size let the original image
 denoted by
%A this A arew formed by taking the 3x3 matrix in the border matix i.e
 from
```

```
%bfim
%Considering the fuzzy templats as mask of size 3x3 and then we will
 slide
%this matix  in the fuzzy matrix i.e in the fim not inj the bfim
for i = 2:r+1
    for j = 2:k+1
        A = [bfim(i-1,j-1) bfim(i,j-1) bfim(i+1,j-1) ; bfim(i-1,j)
 bfim(i,j) bfim(i+1,j) ; bfim(i-1,j+1) bfim(i,j+1) bfim(i+1,j+1)];
        piA = [pibfim(i-1,j-1) pibfim(i,j-1) pibfim(i+1,j-1) ;
 pibfim(i-1,j) pibfim(i,j) pibfim(i+1,j) ; pibfim(i-1,j+1) pibfim(i,j
+1) pibfim(i+1,j+1)];

        %3x3 matrix for determining the divergence with the tempelets
 t1,
        %t2...15,16.
        %we calculate the divergence of 3x3 matrix at a time and then
        %taking the minimun element of the matrix for all 16 fuzzy
        %tempelets;
        %d1 is a matrix of 3x3 = divergence with original matix and
        %fuzzy templets 1
        d1 = 2 - (1-A+t1).*exp(A-t1)-(1-t1+A).*exp(t1-A)+ 2- (1-(A-
t1)+pit1-piA).*exp(A-t1-(pit1-piA))-(1-(pit1-piA)+A-t1).*exp(pit1-piA-
(A-t1));
        min1 =min(min(d1));
        %d2 is the matix of 3x3 = divergence matix with orinigal
 matrix and fuzzy tempelts 2.
        d2 = 2 - (1-A+t2).*exp(A-t2)-(1-t2+A).*exp(t2-A)+2-(1-(A-
t2)+pit2-piA).*exp(A-t2-(pit2-piA))-(1-(pit2-piA)+A-t2).*exp(pit2-piA-
(A-t2));
        min2 =min(min(d2));
        d3 = 2 - (1-A+t3).*exp(A-t3)-(1-t3+A).*exp(t3-A)+2-(1-(A-
t3)+pit3-piA).*exp(A-t3-(pit3-piA))-(1-(pit3-piA)+A-t3).*exp(pit3-piA-
(A-t3));
        min3 =min(min(d3));
        d4 = 2 - (1-A+t4).*exp(A-t4)-(1-t4+A).*exp(t4-A)+2-(1-(A-
t4)+pit4-piA).*exp(A-t4-(pit4-piA))-(1-(pit4-piA)+A-t4).*exp(pit4-piA-
(A-t4));
        min4 =min(min(d4));
        d5 = 2 - (1-A+t5).*exp(A-t5)-(1-t5+A).*exp(t5-A)+2-(1-(A-
t5)+pit5-piA).*exp(A-t5-(pit5-piA))-(1-(pit5-piA)+A-t5).*exp(pit5-piA-
(A-t5));
        min5 =min(min(d5));
        d6 = 2 - (1-A+t6).*exp(A-t6)-(1-t6+A).*exp(t6-A)+2-(1-(A-
t6)+pit6-piA).*exp(A-t6-(pit6-piA))-(1-(pit6-piA)+A-t6).*exp(pit6-piA-
(A-t6));
        min6 =min(min(d6));
        d7 = 2 - (1-A+t7).*exp(A-t7)-(1-t7+A).*exp(t7-A)+2-(1-(A-
t7)+pit7-piA).*exp(A-t7-(pit7-piA))-(1-(pit7-piA)+A-t7).*exp(pit7-piA-
(A-t7));
        min7 =min(min(d7));
        d8 = 2 - (1-A+t8).*exp(A-t8)-(1-t8+A).*exp(t8-A)+2-(1-(A-
t8)+pit8-piA).*exp(A-t8-(pit8-piA))-(1-(pit8-piA)+A-t8).*exp(pit8-piA-
(A-t8));
        min8 =min(min(d8));
        d9 = 2 - (1-A+t9).*exp(A-t9)-(1-t9+A).*exp(t9-A)+2-(1-(A-
t9)+pit9-piA).*exp(A-t9-(pit9-piA))-(1-(pit9-piA)+A-t9).*exp(pit9-piA-
(A-t9));
        min9 =min(min(d9));
        d10 = 2 - (1-A+t10).*exp(A-t10)-(1-t10+A).*exp(t10-A)+2-
(1-(A-t10)+pit10-piA).*exp(A-t10-(pit10-piA))-(1-(pit10-piA)+A-
t10).*exp(pit10-piA-(A-t10));
        min10 =min(min(d10));
        d11 = 2 - (1-A+t11).*exp(A-t11)-(1-t11+A).*exp(t11-A)+2-
(1-(A-t11)+pit11-piA).*exp(A-t11-(pit11-piA))-(1-(pit11-piA)+A-
t11).*exp(pit11-piA-(A-t11));
        min11 =min(min(d11));
        d12 = 2 - (1-A+t12).*exp(A-t12)-(1-t12+A).*exp(t12-A)+2-
(1-(A-t12)+pit12-piA).*exp(A-t12-(pit12-piA))-(1-(pit12-piA)+A-
t12).*exp(pit12-piA-(A-t12));
        min12 =min(min(d12));
        d13 = 2 - (1-A+t13).*exp(A-t13)-(1-t13+A).*exp(t13-A)+2-
(1-(A-t13)+pit13-piA).*exp(A-t13-(pit13-piA))-(1-(pit13-piA)+A-
t13).*exp(pit13-piA-(A-t13));
        min13 =min(min(d13));
        d14 = 2 - (1-A+t14).*exp(A-t14)-(1-t14+A).*exp(t14-A)+2-
(1-(A-t14)+pit14-piA).*exp(A-t14-(pit14-piA))-(1-(pit14-piA)+A-
t14).*exp(pit14-piA-(A-t14));
        min14 =min(min(d14));
        d15 = 2 - (1-A+t15).*exp(A-t15)-(1-t15+A).*exp(t15-A)+2-
(1-(A-t15)+pit15-piA).*exp(A-t15-(pit15-piA))-(1-(pit15-piA)+A-
t15).*exp(pit15-piA-(A-t15));
        min15 =min(min(d15));
        %d16 is the matix of 3x3 = divergence matix with orinigal
 matrix and
        %fuzzy tempelts 16.
        d16 = 2 - (1-A+t16).*exp(A-t16)-(1-t16+A).*exp(t16-A)+2-
(1-(A-t16)+pit16-piA).*exp(A-t16-(pit16-piA))-(1-(pit16-piA)+A-
t16).*exp(pit16-piA-(A-t16));
        min16 =min(min(d16));
        %Selecting the minimun divergence among the 16 divergence
 values
        %and is positioned at the center of the templets position for
 the
        %edge iamge i.e in edgeim.
        dd = [min1 min2 min3 min4 min5 min6 min7 min8 min9 min10 min11
 min12 min13 min14 min15 min16];
        fedgeim(i-1,j-1) = max(dd);
    end
end
% We wil get the edge image in the fuzzy doamin as edgeim matrix So we
 have
% to transform back in the image pixel domain i.e in the interval [1
 255]
fedgeimmax = max(max(fedgeim));
edgeim = double((1/fedgeimmax)*(fedgeim));

% Output
tt = 255*edgeim;
```

```
ttt = uint8(tt);
addedim=imadd(uint8(I),ttt);

im1=imread('img.jpg');
im2=imread('test.png');
figure; clf;
set(gcf,'Name','Original Image');
imshow(rgb2gray(im1));
title('Original Image');
figure;
set(gcf,'Name','After anisotropic diffusion');
imshow(rgb2gray(im2));
title('Image after anisotropic diffusion');
figure;
set(gcf,'Name','Final Cartoon after edge detection');
imshow(addedim);
title('Final Cartoon after edge detection');
```

Original Image

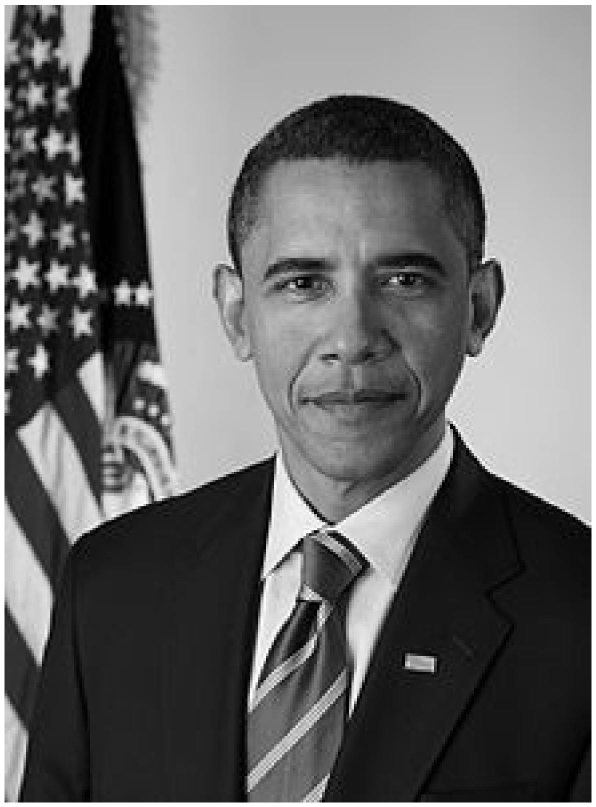
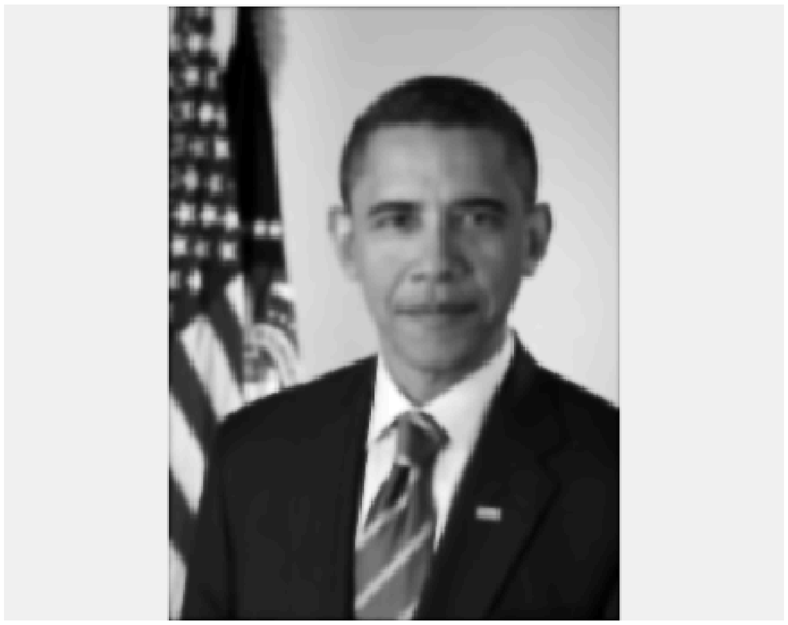
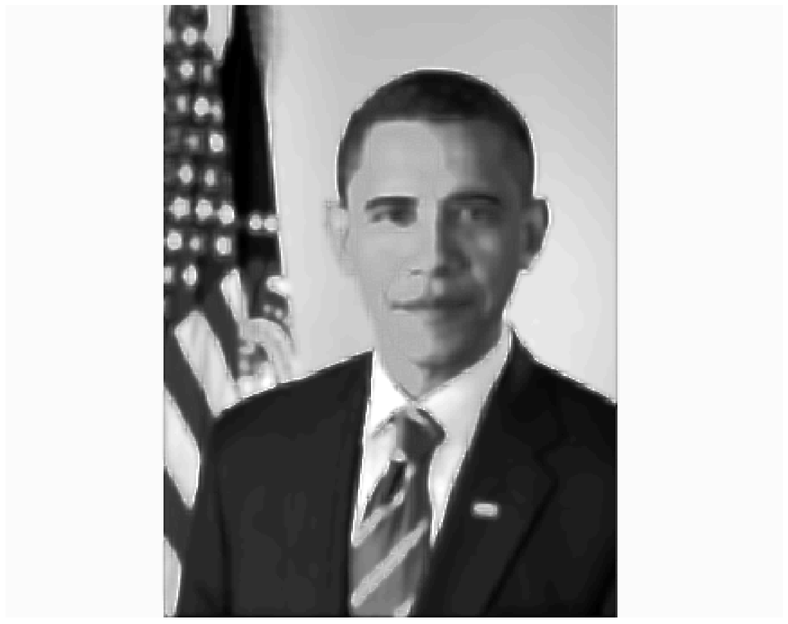


Image after anisotropic diffusion



Final Cartoon after edge detection



Published with MATLAB® R2015b