HW2 EDGE DETECTION

YUKUI YE

1:Implement the following three steps of the Canny edge detection algorithm as we discussed in the class: Relative Code At The End Of The Report;Specific code check at "\*.m" files

2: [15%] Test your algorithm on images ôFlowers.jpgö, ôSyracuse 01.jpgö and ôSyracuse 02.jpgö. Try different values of s (the standard deviation of the Gaussian), tl (lower threshold) and th (higher threshold). Compare and evaluate your results.

3:[10%] Test your algorithm on a favorite image of yours.

a:Input Flowers image. Apply Gaussian smoothing by using deviation=1 with withd = [-1,0,1]; firstTime:deviation=1;secondeTime:deviation=3

aftSmoothingImg= gaussianSmoothing(deviation,Flowers);

[es,eo] = cannyEnhancer (aftSmoothingImg);

eo = qualifyeo(eo,Flowers);

aftNmsuppress = nonmax\_suppress(eo,es,Flowers);

finalmatrix= hysteresisThresh(aftNmsuppress,eo,80,20,Flowers);

imshow(uint8(finalmatrix));

WITH DEVIATION=1

WITH DEVIATION=3

CONCLUSIONT: edges will disappear with bigger deviation

b:Input syracuse\_01 image which size is <590\*900>, Apply Gaussian smoothing by using deviation=1 with withds 3\*3= [-1,0,1];

firstTime:deviation=1;secondeTime:deviation=4

aftSmoothingImg= gaussianSmoothing(1,Syracuse\_01);

[es,eo] = cannyEnhancer (aftSmoothingImg);

eo = qualifyeo(eo,Syracuse\_01);

aftNmsuppress = nonmax\_suppress(eo,es,Syracuse\_01);

finalmatrix= hysteresisThresh(aftNmsuppress,eo,80,20,Syracuse\_01);

imshow(uint8(finalmatrix));

WITH DEVIATION=1

WITH DEVIATION=4

CONCLUSIONT: bigger deviation, more blur image.

c:Input Syracuse\_02 image which size is <571\*862>, Apply Gaussian firstTime: deviation=1 highThreshold=80,lowThreshold=20;

secondTime: deviation=1 highThreshold=50,lowThreshold=10;

aftSmoothingImg= gaussianSmoothing(1,Syracuse\_02);

[es,eo] = cannyEnhancer (aftSmoothingImg);

eo = qualifyeo(eo,Syracuse\_02);

aftNmsuppress = nonmax\_suppress(eo,es,Syracuse\_02);

finalmatrix= hysteresisThresh(aftNmsuppress,eo,80,20,Syracuse\_02);

imshow(uint8(finalmatrix));

high=80,low=20

high=50,low=10

CONCLUSIOION: lower higherThreshold and lowerThreshold, clearer edges will show up.

d: Input my own photo named”hello”,which is a three dimention array<637\*960\*3>

firstTime:deviation=1;high=30,low=10

secondTime:deviation=1;high=80,low=30

aftSmoothingImg= gaussianSmoothing(1,hello);

[es,eo] = cannyEnhancer (aftSmoothingImg);

eo = qualifyeo(eo,hello);

aftNmsuppress = nonmax\_suppress(eoq,es,hello);

finalmatrix= hysteresisThresh(aftNmsuppress,eo,30,10 ,hello);

imshow(uint8(finalmatrix));

FirstTime

SecondTime

conclusion ： 3d image is weird. It comes out three times.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Function Code Used In This Assignment. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

GaussianSmoothing

function [aftSmoothImg] = gaussianSmoothing(deviation,originalImage)

[x\_size,y\_size]=size(originalImage);

w=5\*deviation;

mp=(w+1)/2;

for i=1:w

G0(i)=exp(-((i-mp)^2)/(2\*(deviation^2)))/deviation\*sqrt(2\*pi);

end

G1=G0/G0(1);

G2=floor(G1);

G=G2/sum(G2);

matrix=zeros(x\_size,y\_size);

for i=1:x\_size

holdmatrix=conv2(originalImage(i,:),G,'same');

matrix(i,:)=holdmatrix;

end

for j=1:y\_size

midmatrix=conv2(matrix(:,j),transpose(G),'same');

matrix(:,j)=midmatrix;

end

aftSmoothImg=matrix;

end

%i)CANNYENHANCER

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

function [ es,eo ] = cannyEnhancer( aftSmoothingImg )

[x\_size,y\_size]=size(aftSmoothingImg);

w=[-1,0,1];

for i=1:x\_size

holdmatrix=conv2(aftSmoothingImg(i,:),w,'same');

jx(i,:)=holdmatrix;

end

for j=1:y\_size

midmatrix=conv2(aftSmoothingImg(:,j),transpose(w),'same');

jy(:,j)=midmatrix;

end

es=zeros(x\_size,y\_size);

eo=zeros(x\_size,y\_size);

for row=1:x\_size

for colum=1:y\_size

es(row,colum)=sqrt(jx(row,colum).^2 + jy(row,colum).^2);

eo(row,colum)=(180/pi)\*atan(jy(row,colum)/jx(row,colum));

end

end

end

%ii) [25%] QUALIFYEO

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

function[ eo ] = qualifyeo( eo,originalImage )

[x\_size,y\_size]=size(originalImage);

eo=uint8(eo);

for row=1:x\_size

for colum=1:y\_size

if((0<eo(row,colum) && eo(row,colum)< 22.5 )|| (157.5 <eo(row,colum) && eo(row,colum)< 180))

eo(row,colum)=0;

elseif( 22.5< eo(row,colum) && eo(row,colum) < 67.5)

eo(row,colum)=45;

elseif( 67.5 <eo(row,colum) && eo(row,colum) < 112.5)

eo(row,colum)=90;

elseif(112.5 <eo(row,colum) && eo(row,colum) < 157.5)

eo(row,colum)=135;

end

end

end

%ii) [25%] NONMAX SUPPRESSION

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

function [aftNmsuppress ] = nonmax\_suppress( eo,es,originalImage )

[x\_size,y\_size]=size(originalImage);

aftNmsuppress=zeros(x\_size,y\_size);

for x=2:(x\_size-1)

for y=2:(y\_size-1)

if(eo(x,y) == 0)

if(es(x,y+1) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x,y+1) > es(x,y))

aftNmsuppress(x,y)=0;

end

if(es(x,y-1) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x,y-1) > es(x,y))

aftNmsuppress(x,y)=0;

end

end

if(eo(x,y) == 45)

if(es(x-1,y+1) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x-1,y+1) > es(x,y))

aftNmsuppress(x,y)=0;

end

if(es(x+1,y-1) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x+1,y-1) > es(x,y))

aftNmsuppress(x,y)=0;

end

end

if(eo(x,y) == 90)

if(es(x+1,y) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x+1,y) > es(x,y))

aftNmsuppress(x,y)=0;

end

if(es(x-1,y) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x-1,y) > es(x,y))

aftNmsuppress(x,y)=0;

end

end

if(eo(x,y) == 135)

if(es(x+1,y+1) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x+1,y+1) > es(x,y))

aftNmsuppress(x,y)=0;

end

if(es(x-1,y-1) < es(x,y))

aftNmsuppress(x,y)=es(x,y);

elseif(es(x-1,y-1) > es(x,y))

aftNmsuppress(x,y)=0;

end

end

end

end

end

%iii) [25%] HYSTERESIS THRESH \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

function [ finalmatrix ] =hysteresisThresh( aftNmsuppress,eo,highThresh,lowThresh,originalImage)

es=aftNmsuppress;

[x\_size,y\_size]= size(originalImage)

for x=1:x\_size

for y=1:y\_size

if((es(x,y) > highThresh)&&es(x,y)~=255)

es(x,y)=255;

if(eo(x,y) == 0)

if(es(x+1,y) < lowThresh)

es(x+1,y)=0;

elseif(es(x+1,y)> lowThresh)

es(x+1,y)=255;

end

if(es(x-1,y) < lowThresh)

es(x-1,y)=0;

elseif(es(x-1,y)> lowThresh)

es(x-1,y)=255;

end

end

if(eo(x,y) == 45)

if(es(x+1,y+1) < lowThresh)

es(x+1,y+1)=0;

elseif(es(x+1,y+1)> lowThresh)

es(x+1,y+1)=255;

end

if(es(x-1,y-1) < lowThresh)

es(x-1,y)=0;

elseif(es(x-1,y-1)> lowThresh)

es(x-1,y-1)=255;

end

end

if(eo(x,y) == 90)

if(es(x,y+1) < lowThresh)

es(x,y+1)=0;

elseif(es(x,y+1)> lowThresh)

es(x,y+1)=255;

end

if(es(x,y-1) < lowThresh)

es(x,y-1)=0;

elseif(es(x,y-1)> lowThresh)

es(x,y-1)=255;

end

end

if(eo(x,y) == 135)

if(es(x-1,y+1) < lowThresh)

es(x-1,y+1)=0;

elseif(es(x-1,y+1)> lowThresh)

es(x-1,y+1)=255;

end

if(es(x+1,y-1) < lowThresh)

es(x+1,y-1)=0;

elseif(es(x+1,y-1)> lowThresh)

es(x+1,y-1)=255;

end

end

end

end

end

finalmatrix=es;

end