Assignment III CORNER DETECTION

YUKUI YE SUID:439644268

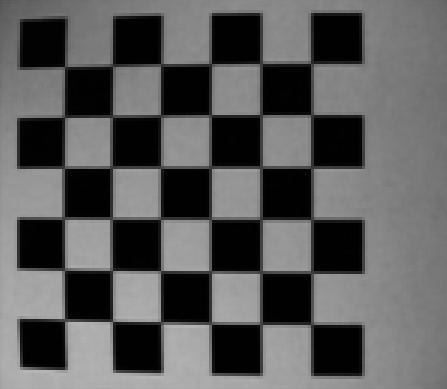
1) [10%] First, apply Gaussian smoothing (with standard deviation σ) to an input image I, to obtain Is,

Is\_Building = gaussianSmoothing(1,Building1);

imshow(uint8(Is\_Building));

Is\_CheckBoard = gaussianSmoothing(1,CheckerBoard);

imshow(uint8(Is\_CheckBoard));

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

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

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

2) [70%] Implement the corner detection algorithm (CORNERS), by using Is as input, as described in class and also in the textbook,

%i)CORNER ALGORITHM

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

function [ sortList ] = getSortedList( aftSmoothingImg,threshold )

[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

n=x\_size\*y\_size

list=zeros(n,3);

cnt=0;

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

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

ex(1:3,1:3)=jx(i-1:i+1,j-1:j+1);

ey(1:3,1:3)=jy(i-1:i+1,j-1:j+1);

ex2=ex^2;

Ex2=sum(sum(ex2));

exy=ex\*ey;

Exy=sum(sum(exy));

eyx=ey\*ex;

Eyx=sum(sum(eyx));

ey2=ey^2;

Ey2=sum(sum(ey2));

minValue=abs(min(eig([Ex2 Exy; Eyx Ey2])));

if(minValue>threshold)

cnt=cnt+1;

list(cnt,1)=minValue;

list(cnt,2)=i;

list(cnt,3)=j;

end

end

end

sortList=flipud(sortrows(list,1));

end

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

getfinalList which will delete some value

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

function [ finalList ] = getFinal(sortList,wth)

s=size(sortList);

for m=1:s-1

i=sortList(m,2);

j=sortList(m,3);

for n=m+1:s

if( sortList(n,2)>=(i-2\*wth-2) && sortList(n,2)<=(i+2\*wth+2) && sortList(n,3)>=(j-2\*wth-2) && sortList(n,3)<=(j+2\*wth+2) )

sortList(n,1)=0;

end

end

end

finalList=flipud(sortrows(sortList,1));

end

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

getfinalImg which will show the corner of the image with value 225

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

function [ ] = getFinalImg(Is,finalList,N )

s=size(finalList);

for a=1:s

if(finalList(a,1)~=0)

ii=finalList(a,2);

jj=finalList;

for q= jj-N:jj+N

Is(ii-N,q)=225;

end

for q=jj-N:jj+N

Is(ii+N,q)=225;

end

for q= ii-N:ii+N

Is(q,jj-N)=225;

end

for q=ii-N:ii+N

Is(q,jj+N)=225;

end

end

end

imshow(uint8(Is));

end

3) [20%] Test your corner detection algorithm on images “Building1.jpg” and “CheckerBoard.jpg”. Try diﬀerent values of the σ, the neighborhood size, and the threshold (τ) on λ2. Compare and evaluate your results.

A:deviation=1; windows size N=6 ;threshold=200;

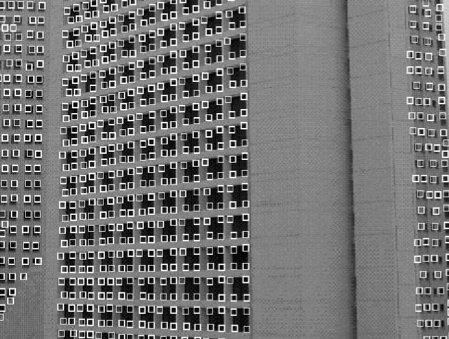
B: deviation=6; windows size N=3 ;threshold=800;

Is\_Building = gaussianSmoothing(deviation,Is\_Building);

[ sortList ] = getSortedList( Is\_Building,threshold );

[ finalList ] = getFinal(sortList,N);

getFinalImg(Is\_Building,finalList );

A B

Conclusion:

Larger deviation, more blur image will appear. Because deviation make a huge impact on Gaussian smoothing, which in turn will smooth the image, thus in the second image , we cannot see corner clearly.

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

A:deviation=1; windows size N=6 ;threshold=800;

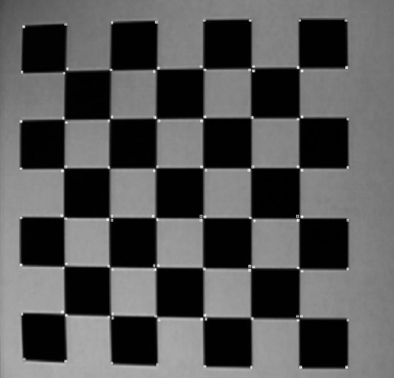
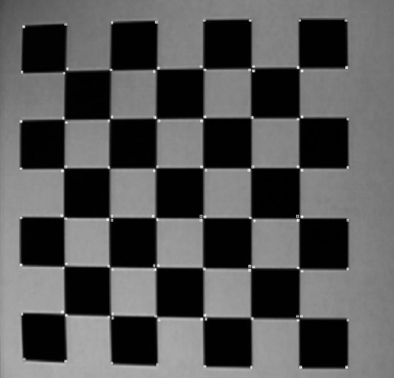
B: deviation=1; windows size N=3 ;threshold=800;

Is\_CheckerBoard = gaussianSmoothing(deviation,Is\_CheckerBoard);

[ sortList ] = getSortedList( Is\_CheckerBoard,threshold );

[ finalList ] = getFinal(sortList,N);

getFinalImg(Is\_CheckerBoard,finalList );

A B 

Conclusion:

If enlarge windows size(neighborhood), but remain threshold, will appear more corners since the corner points become more bright. But if we check carefully, some of the points are not corner. So, we probably will enlarge window’s size and threshold at the same time.

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

\*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)CORNER ALGORITHM

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

function [ sortList ] = getSortedList( aftSmoothingImg,threshold )

[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

n=x\_size\*y\_size

list=zeros(n,3);

cnt=0;

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

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

ex(1:3,1:3)=jx(i-1:i+1,j-1:j+1);

ey(1:3,1:3)=jy(i-1:i+1,j-1:j+1);

ex2=ex^2;

Ex2=sum(sum(ex2));

exy=ex\*ey;

Exy=sum(sum(exy));

eyx=ey\*ex;

Eyx=sum(sum(eyx));

ey2=ey^2;

Ey2=sum(sum(ey2));

minValue=abs(min(eig([Ex2 Exy; Eyx Ey2])));

if(minValue>threshold)

cnt=cnt+1;

list(cnt,1)=minValue;

list(cnt,2)=i;

list(cnt,3)=j;

end

end

end

sortList=flipud(sortrows(list,1));

end

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

getfinalList which will delete some value

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

function [ finalList ] = getFinal(sortList,wth)

s=size(sortList);

for m=1:s-1

i=sortList(m,2);

j=sortList(m,3);

for n=m+1:s

if( sortList(n,2)>=(i-2\*wth-2) && sortList(n,2)<=(i+2\*wth+2) && sortList(n,3)>=(j-2\*wth-2) && sortList(n,3)<=(j+2\*wth+2) )

sortList(n,1)=0;

end

end

end

finalList=flipud(sortrows(sortList,1));

end

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

getfinalImg which will show the corner of the image with value 225

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

function [ ] = getFinalImg(Is,finalList,N )

s=size(finalList);

for a=1:s

if(finalList(a,1)~=0)

ii=finalList(a,2);

jj=finalList;

for q= jj-N:jj+N

Is(ii-N,q)=225;

end

for q=jj-N:jj+N

Is(ii+N,q)=225;

end

for q= ii-N:ii+N

Is(q,jj-N)=225;

end

for q=ii-N:ii+N

Is(q,jj+N)=225;

end

end

end

imshow(uint8(Is));

end