ECE 5256

Project 3: Correlation and Convolution in the Spatial Domain

Keefe Kamp

10 February 2023

Part 1

For this part of the project I looked at the difference between correlations and convolution. In order to do this I created a image of 20 random points and then preformed a convolution and correlation using a smaller image of a ramp. The notable difference with convolution (Fig. 2) and correlation (Fig. 1) is the direction that the imparted ramps are facing. This is due to the fact that with with Convolution the kernel is rotated 90° before the process is began. This results in the imparted graphic begin flipped both in x and y between the two processes.

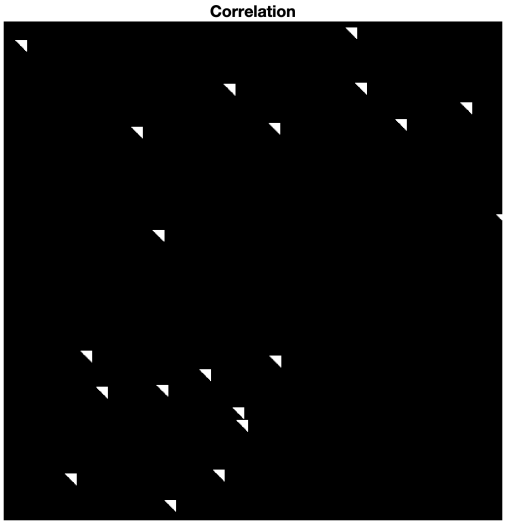

Figure 1: Correlation result from using a set of random points and a "ramp"

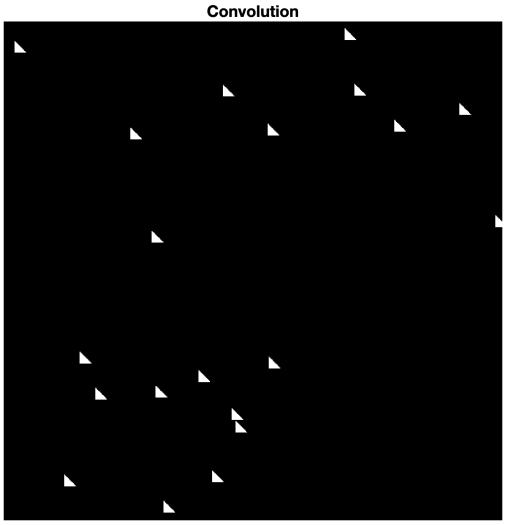

Figure 2: Convolution result from using a set of random points and a "ramp"

Part 2

Figure 3: Input image for the text correlation. The w in white was selected for the correlation kernel.

Using the data cursor in MATLAB I selected two points as my correlation peak, this is where the w is in the text and but next highest point which appeared at a 'y' the ratio between these two in the non-normalized image (Fig. 4) is 1.7856 whereas in the normalized image (Fig. 5) it is 1.3537. This makes sense as with the normalization the "letter interactions" are more visible not just the w interactions that are the bight spots in Fig. 4. It is of note that there is also a bright peak on either side of the 'w' this is most likely due to the the opposite 'v's line up here.

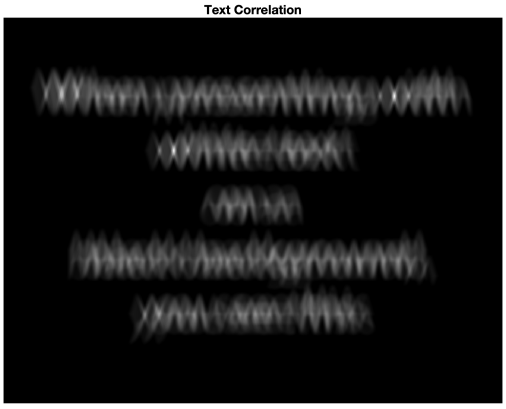

Figure 4: Results of the correlation using the w kernal

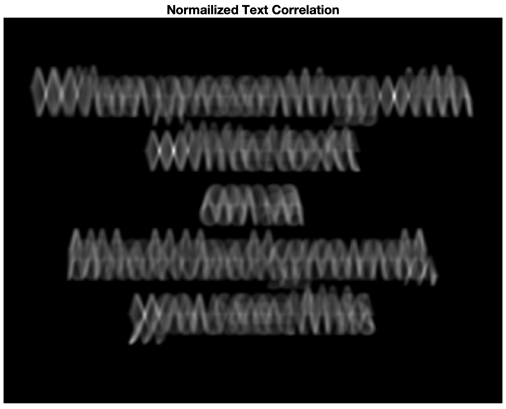

Figure 5: The result of the correlation after it has been normalized

Part 3

In this portion of the project I investigated sobel edge detection. First I took and image and added noise to it (Fig. 6). I then ran a sobel edge detection, $\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$, over the image. The results are in Fig. 7.

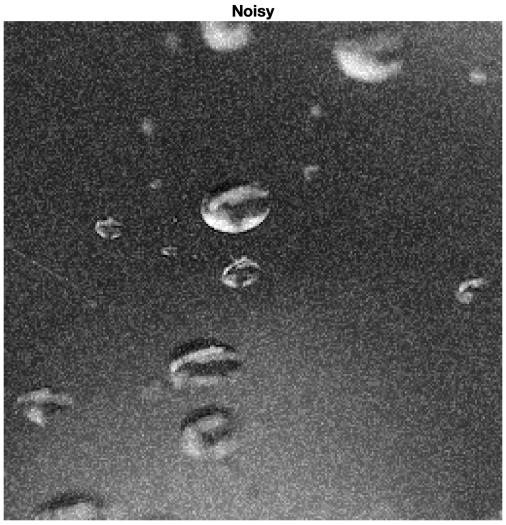

Figure 6: Original Image

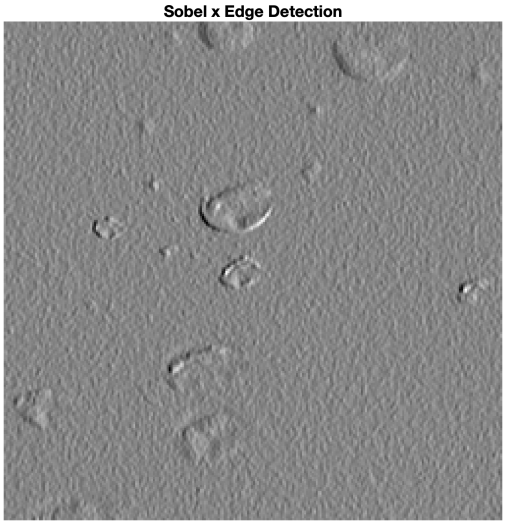

Figure 7: results of running a Sobel edge detection in the x direction

I then applied two different filters to the original image (Fig. 6) before applying the edge detection. The first filter I applied was a uniform filter $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$. Fig. 8 shows the results of the edge detection after the uniform filter was applied. Finally I used a Gaussian filter, $\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$. Fig. 9 shows the results of the edge detection. We can see that after these filters the edges aren't as clear. This is due to these filters being low pass filters. These filterers did remove the noise from the image.

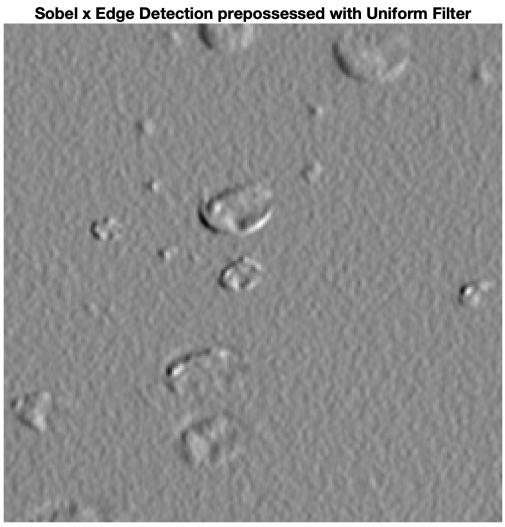

Figure 8: Sobel edge detection on the noisy image after using a uniform low pass filter.

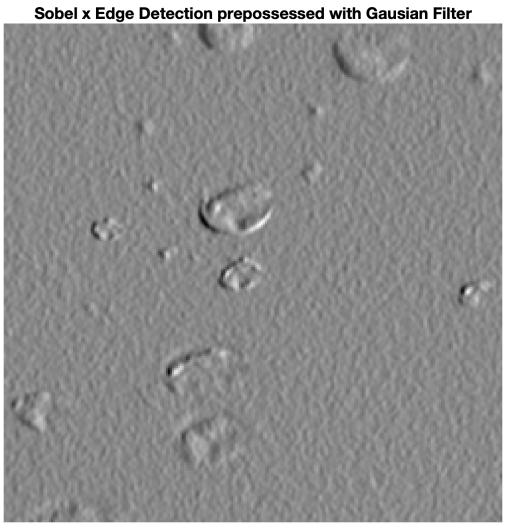

Figure 9: Sobel edge detection on the noisy image after using a Gaussian low pass filter.

A MATLAB Code

This is the code used for the coding portions of this project.

Skip to content

Search or jump to

Pull requests

Issues

Codespaces

Marketplace

Explore

@KKamp2015

KKamp2015

/

DigitalImageProcessing

Public

Cannot fork because you own this repository and are not a member of any organizations.

Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

DigitalImageProcessing/Project3/Project3Code.m

Cannot retrieve latest commit at this time.

0 contributors

118 lines (104 sloc) 3.12 KB

```
%Housekeeping commands
clear all
close all

%Part 1
x=randi([1,512],1,20); %Picking 20 random x values
y=randi([1,512],1,20); %Picking 20 random y values
points=zeros([512,512]); %memory allocation
for i=1:20
    points(x(i),y(i))=255; %assigning the random x,y to 255
end
```

```
%Displaying image
imagesc(points)
colormap('gray')
title('Points')
axis off image
exportgraphics(gcf,'Points.png','Resolution',300)

%reading in ramp
ramp=imread('ramp.png','png'); %reading in image
ramp=im2gray(ramp); %converting to grayscale

bCorr=filter2(ramp,points); %performing correlation
%displaying Correlation result
figure
imagesc(bCorr)
colormap('gray')
title('Correlation')
axis off image
exportgraphics(gcf,'Correlation.png','Resolution',300)


cConv=conv2(points,ramp, 'same'); %perfroming convolution
%Displaying Convolution result
figure
imagesc(cConv)
colormap('gray')
title('Convolution')
```

```
axis off image
exportgraphics(gcf,'Convolution.png','Resolution',300)

%Part2
words=im2gray(imread("Text.png"));%reading in text image

w=im2gray(imread('w.png'));%reading in w image

textcorr=filter2(w,words); %performing correlation
%displaying Correlation Result
figure
imagesc(textcorr)
colormap('gray')
axis off image
title('Text Correlation')
datacursormode('on')
exportgraphics(gcf,'TextCorr.png','Resolution',300)

%Perfroming Normalzination
NormFilter=ones(size(w)); %memory allocation
NormFilter=1/sum(NormFilter,'all').*NormFilter;%making normalization kernal
NormConv=conv2(textcorr,NormFilter,'same');%performing filter normztion
Norm=textcorr./NormConv; %dividing by normailed filter image
%displaying normalziation resutl
figure
imagesc(Norm)
colormap('gray')
```

```
axis off image
title('Normailized Text Correlation')
datacursormode('on')
exportgraphics(gcf,'NormTextCorr.png','Resolution',300)



%Part3
a=imread('Drops.jpeg','jpeg'); %reading in image
a=im2gray(a); %Converting to Greyscale image

noise=uint8(floor(randn(256).*20+20)); %making noise array
b=a+noise; %adding noise to original image
%
figure
imagesc(b)
colormap('gray')
axis off image
title('Noisy')
exportgraphics(gcf,'Noisy.png','Resolution',300)

sobx=[1 0 -1;2 0 -2;1 0 -1]; %Creating Sobel opperator
edges=filter2(sobx,b,"valid"); %Perfroming sobel edge finder
%Displaying results
figure
imagesc(edges)
colormap('gray')
axis off image
```

```
title('Sobel x Edge Detection')
exportgraphics(gcf,'SobelXDrops.png','Resolution',300)

%creating unifomr filter
uni=ones(3);
uni=1/sum(uni,'all') * uni;
uniDrops=filter2(uni,b,'valid'); %Performing uniform filter correlation
uniDropsEdge=filter2(sobx,uniDrops,'valid'); %performing edge dectection
%Displaying results
figure
imagesc(uniDropsEdge)
colormap('gray')
axis off image
title('Sobel x Edge Detection prepossessed with Uniform Filter')
exportgraphics(gcf,'UniSobelXDrops.png','Resolution',300)
%creating Gaussian kernal
Gauss=1/16 .* [1 2 1;2 4 2;1 2 1];
GaussDrops=filter2(Gauss,b,'valid');%performing gaussian
GaussDropsEdge=filter2(sobx,GaussDrops,'valid'); %edge detection
%Displaying Results
figure
imagesc(GaussDropsEdge)
colormap('gray')
axis off image
title('Sobel x Edge Detection prepossessed with Gausian Filter')
exportgraphics(gcf,'GaussSobelXDrops.png','Resolution',300)
Footer
```

2023 GitHub, Inc.

Footer navigation

Terms

Privacy

Security

Status

Docs

Contact GitHub

Pricing

API

Training

Blog

About

DigitalImageProcessing/Project3Code.m at main KKamp2015/DigitalImageProcessing