~~Digital Signal Proc.~~

Digital Image Processing Lecture 6

Spatial Filters

SPATIAL FILTERING (Important)

1-D Signal Processing (Filter in time domain or the frequency domain)

$Y = x * h$

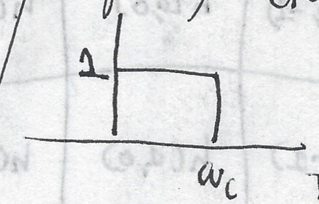
[. . .] $h[n-k]$

[. . . . .] $X[k]$

$$Y[n] = \sum_{k=-\infty}^{\infty} X[k]\, h[n-k]$$

recall wiener filters, FIR, adaptive filters.

frequency domain



Today → 2D analog of time domain

Small filter, slide across image for an output

IN 2-D the Analogue to "TIME Domain"

"SPATIAL Domain"

1D filter

$[-1 \ \underline{2} \ -1]$ 1-D

underline zero zero element

Grid of filter coefficients

h(-1,-1)	h(-1,0)	h(-1,1)
h(-1,0)	h(0,0)	h(0,1)
h(-1,1)	h(0,+1)	h(1,1)

2-D

→ All of this is a lot more tedious than it looks

x ↓ \ y →	-1	0	1
-1	h(-1,-1)	h(-1,0)	h(-1,1)
0	h(0,-1)	h(0,0)	h(0,1)
1	h(1,-1)	h(1,0)	h(1,1)

We wont run into indexing problems in Image Processing,

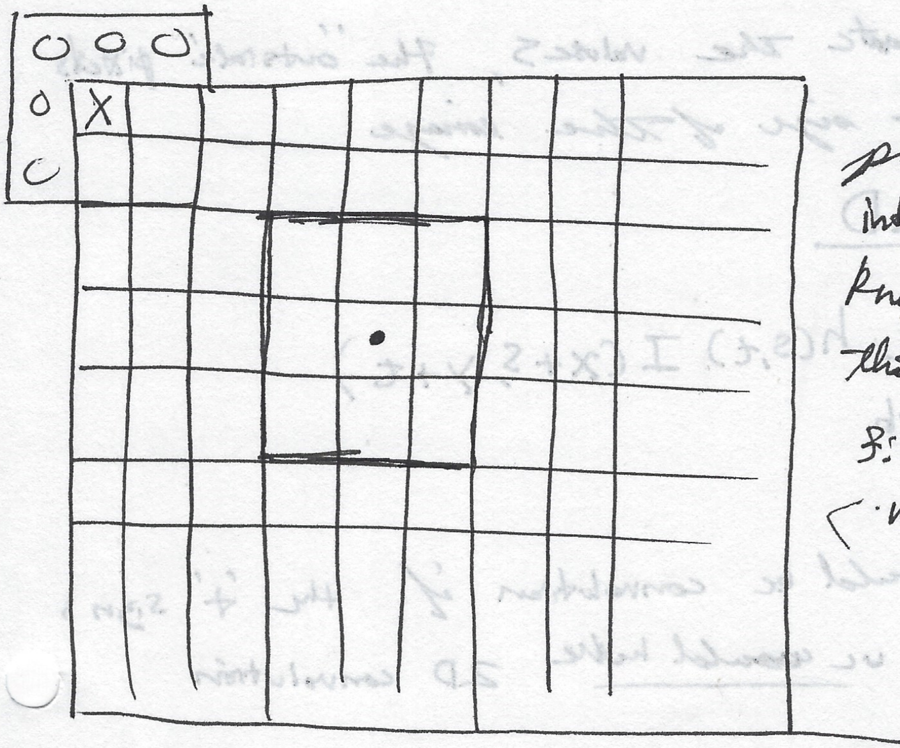
our filters are easy numbers and symmetric around the middle. we think of the middle element as the 0,0 element

we usually deal with odd filters

3x3 5x5 → symmetry around the middle

0	-1	0
-1	5	-1
0	-1	0

we take the block of #5 and ~~slide~~ slide it <u>across the image</u>: Way to apply the filter to the image



Say that the dotted pixel is ~~the~~ pixel of interest, so we want to know the filter response at that pixel, we center the filter on that pixel, we multiply corresponding elements then add up. recall?

use the felter

$$\begin{matrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{matrix}$$

"Filter"
"Mask"

what pixels of the original image does that <u>Mask cover</u>?

$$I(x,y) \rightarrow J(x,y)$$

$$J(x,y) = 5I(x,y) - I(x+1,y) - I(x-1,y) - I(x,y+1) - I(x,y-1)$$

Subtract or add filters today we see different <u>kinds of filters</u>.

4) Say we have a pixel at the upmost left most part of the image, we may discount that pixel, no response but we change the size of the image in which the pipeline may not like.

or we could hallucinate the values, the "outside" pixels are zero. we retain the size of the image

Convolution in 2D

$$J(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} h(s,t)\, I(x+s, y+t)$$

This would be convolution if the '+' signs were '−' sign, we ~~would~~ have 2D convolution

We have Correlation. The difference being to flip the mask around

What are the useful kinds of spatial filters?

SMOOThing FILTER (AKA Low pass filter)
Moving average filter

IDEA: To re place each pixel by a weighted average of its neighbors

PRO: Removes / Reduces Noise
CON: Blurs the IMAGE, Removes detail

unity filter

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

~~MatLab example~~
~~16:43~~

2 ways To filter in MatLab!

filter2(h, im)
→ Get actual, floating point Results
→ Not scaled/clipped To [0, 255] (Image range)

imfilter(im, h) If we have an Image going in, we have the right Image going out
→ uint8 → Uint8
→ works with color
→ May lose important values < 0 or > 255

6 Mat Lab 19:00

im = imread( );

Try out on my own

This is a long one

Highpass ~ image edges

get rid of noise, get rid of edges

Low pass filters

33:24

$$\frac{\text{ones}(n)}{n^2}$$

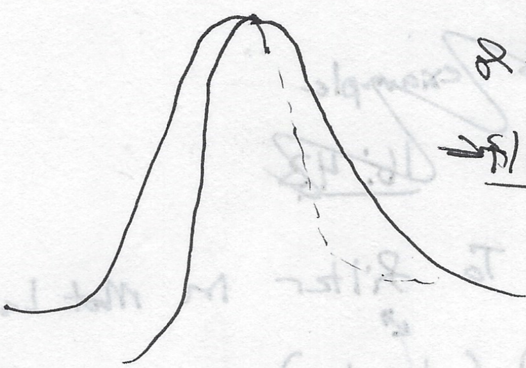
variance

by tuning σ of the Gaussian, flat or skinny filter

What about

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

or



Sharpening filters

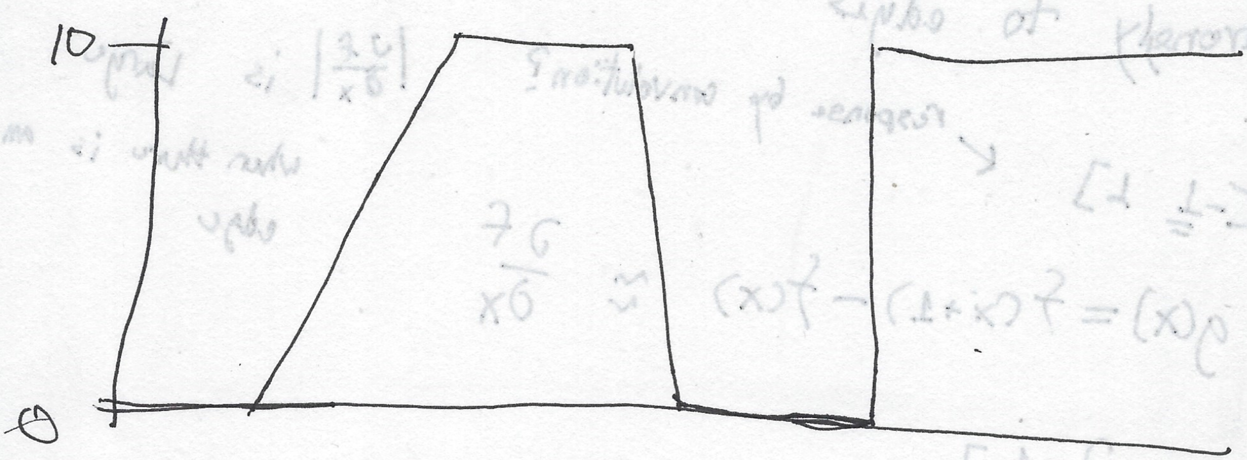
Image averaging ~ integration
(adding stuff up)

For sharpening pixels we want to look at the differences of pixels, (like differentiation)

IMAGE Differencing (Related to Differentiation)

Results in sharpening

IMAGE Edge (1-D)



000 2 4 6 8 10 10 5 0 0 0 10 10 10 10 ← f(x)

← Grey Scale intensity values along edge

Say f(x) is our original function

Discrete world, ~~Difference~~ derivative is the difference between adjacent values

$f'(x)$ 0 0 2 2 2 2 2 0 -5 -5 0 0 10 0 0 0 0 ← First derivative for the first difference

$f''(x)$ 0 2 0 0 0 0 -2 -5 0 5 0 +10 -10 0 0 0   Large values correspond when the image is changing quickly

sign change

We see values at the onset of change

(2) $f'(x)$ and $f''(x)$ are important to find the edges in Images, the edges are important to Sharpening Images

1st step in sharpening, find a filter that reacts strongly to edges

response by convolution?

$\left|\frac{\partial f}{\partial x}\right|$ is Large when there is an edge

1-D $[-1 \; 1]$

$$g(x) = f(x+1) - f(x) \approx \frac{\partial f}{\partial x}$$

$[1 \; -2 \; 1]$

$$g(x) = f(x+1) + f(x-1) - 2f(x) \approx \frac{\partial^2 f}{\partial x^2}$$

$\frac{\partial^2 f}{\partial x^2}$ shows sign changes near edges But is 0 elsewhere

derivative

→ A more Local response

2-D Versions?

Look at $\frac{\partial^2 f}{\partial x^2}$ → to find edges in both x and y Directions

we use an approximation of the Laplacian

$$\nabla^2 f = \frac{\partial^2}{\partial x^2} f + \frac{\partial}{\partial y^2} f$$

9

approximate $\frac{\partial^2 f}{\partial x^2}$

$f(x+1,y)+f(x-1,y)-2f(x,y)$ $\frac{\partial^2}{\partial y^2} f$

$f(x,y+1)+f(x,y-1)-2f(x,y)$

So we have

$$f(x+1,y)+f(x-1,y)+f(x,y+1)+f(x,y-1)$$
$$-4f(x,y)$$

So use the filter

$$\boxed{\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix}}$$

if the middle value is ~~larger~~ larger than its neighbors, we get a large positive or negative response

react strongly to edges

MatLab, 45:56

10

How to Enhance / Sharpen Images?

IDea: strengthen Edges of the original Image by adding a multiple of the edge map to it.

$$J(x,y) = I(x,y) - \nabla^2 I(x,y)$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

We get

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Matlab : 55:20 ish

Note, by enhancing the Image, we have enhanced the noise

filter was too enhanced

→ We can just add a fraction of the edges back in, for a more subtle effect.

Unsharp Masking

Original Image: f

Low-Pass Image: $\bar{f}$

high-Pass Image: $f - \bar{f} = f_{hp}$

$OUTPUT = f + K \cdot f_{hp}$

↖ tunable parameter

$\boxed{\text{Matlab : 1:00:36}}$

Filters need not be symmetric 1:06:01

-1	-2	-1
0	0	0
1	2	1

SoBel Horizontal EDGE Detector

-1	0	1
-2	0	2
-1	0	1

SoBel verticle EDGE detector

$\boxed{\text{Matlab 1:07:00}}$

LTI Systems, 2D LTI systems

Non linear system can be good.

Not all image filters are linear

e.g. Median filtering

$J(x,y) = \text{median}\{3\times3$ pixel neighborhood around $I(x,y)\}$

"my output pixel"

50	48	46	42
52	0	50	48
46	47	255	40
51	48	46	42

← mostly dark-Grey blob

outliers like 0, and 255

"Salt and pepper noise"

$\boxed{\text{matLab} \quad 1:11:33}$

Take a region of pixels, center around 0, (a problem pixel)

→

50	48	46
52	0	50
46	47	255

Give a list ordered from lowest to highest of values

↳ {0 46 46 47 48 50 50 52 255}

median → replaces the center pixel, 0 in our case pixel dependent choice filter (page 12)

$\boxed{\text{Matlab} \quad 1:13:30}$

Gaussian Noise → LPF

salt and Pepper → medium filter
impulsive noise

$\boxed{\text{Mat lab} \quad 1:15:35}$