Image Filter Design

Dr. Daniel B. Rowe
Professor of Computational Statistics
Department of Mathematical and Statistical Sciences
Marquette University

Outline

Filters

Smoothing Filters

Sharpening Filters

Hybrid Filters

Discussion

Homework

Filters

Designing a filter is both a science and an art.

The first characteristic of filters is the determination of filter size.

General heuristics:

The larger the image, the larger the filter.
-Detail not as critical.

The more homogeneous the image, the larger the filter.
-Average over a larger region to reduce noise.

Filters

There are many within image filters that we can apply.

Each has it’s own properties and scenarios when it should be applied.

As we’ve seen, there are two basic types.

Smoothing (Local Averaging-Integration). Low pass.

Sharpening (Local differences-Differentiation). High pass.

Combinations of Smoothing and Sharpening. High boost, band pass.

Smoothing Filters

Gaussian filters can be designed by specifying the area over which the filter is to operate over.

We create Gaussian filters using the bivariate Gaussian (normal) distribution. We need to specify variance σ^2, standard deviation σ, or full-width-at-half-max (FWHM), $\sigma^2=8*\ln(2)*(fwhm)^2$.

The filter can be radially symmetric if we have no knowledge of the image, or have a preferential direction if we want to average more in one direction than the other.

Smoothing Filters

The zero mean Gaussian distribution with common variance σ^2 is

$$g(x,y)=e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

(Note that we have neglected the normalizing constant.)

We can form a 5×5 Gaussian filter with $\sigma^2=0.5$.

Calculate the unnormalized weights.

0.000335462627903

.0003	.0067	.0183	.0067	.0003
.0067	.1353	.3679	.1353	.0067
.0183	.3679	1.0000	.3679	.0183
.0067	.1353	.3679	.1353	.0067
.0003	.0067	.0183	.0067	.0003

Divide all the values by the corners value.

Round to the nearest integer.

Divide by the sum of the integers.

1	20	55	20	1
20	403	1097	403	20
55	1097	2981	1097	55
20	403	1097	403	20
1	20	55	20	1

/9365

Smoothing Filters

Matlab code for generating a Gaussian Kernel

```
% set the kernel array size
k=5;
% set the spread of the kernel
sigma2=0.5
%sigma2=8*log(2)*fwhm^2;

% form the unweighted kernel
x=(-(k-1)/2:(k-1)/2);
y=(-(k-1)/2:(k-1)/2);
[X,Y]=meshgrid(x,y);
gk=exp(-X.^2/(2*sigma2)).*exp(-Y.^2/(2*sigma2))

% form unweighted integerized kernel
gk=round(gk/gk(1,1))

% integer normalizing constant
c=sum(sum(gk))

% normalized final kernel
gk=gk/c
```

Smoothing Filters

Note that we could have used different variances for differential L-R/U-D

$$g(x,y) = \exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right)\right]$$

or incorporated correlation for an oblique angle

$$g(x,y) = \exp\left[-\frac{1}{2(1-\rho^2)}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} - 2\rho\frac{xy}{\sigma_x\sigma_y}\right)\right] \quad .$$

Smoothing Filters

Quite often the Gaussian filter is approximated with a Binomial filter.

$$g(x,y)=\frac{n!}{(n-x)!x!}p_x^x(1-p_x)^{n-x}\frac{m!}{(m-y)!y!}p_y^y(1-p_y)^{m-y}$$ with $p_x=p_y=p=1/2$.

We can form a 5×5 Binomial filter with $p=1/2$.

Calculate the normalized weights.

.0039	.0156	.0234	.0156	.0039
.0156	.0625	.0938	.0625	.0156
.0234	.0938	.1406	.0938	.0234
.0156	.0625	.0938	.0625	.0156
.0039	.0156	.0234	.0156	.0039

Divide all the values by the corners value.

Round to the nearest integer.

Divide by the sum of the integers.

1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1

/256

Smoothing Filters

Matlab code for generating a Binomial Kernel

```
% set the kernel array size
k=5;
% set the probabilities of the kernel
p=0.5 % note that sigma2=n*p*(1-p);

% form the unweighted kernel
n=k-1; m=n;
x=(0:n);
y=(0:n);
[X,Y]=meshgrid(x,y);
% form weighted kernel
gk=(p.^X).*((1-p).^(n-X)).*(factorial(n)./(factorial(n-X).
    .*(p.^Y.*(1-p).^(m-Y)).*(factorial(m)./(factorial(m-Y).*factorial(Y)))

% form unweighted integerized kernel
gk=round(gk/gk(1,1))
% integer normalizing constant
c=sum(sum(gk))
% normalized final kernel
gk=gk/c
```

Image Smoothing

Gaussian

1	20	55	20	1
20	403	1097	403	20
55	1097	2981	1097	55
20	403	1097	403	20
1	20	55	20	1

/9365

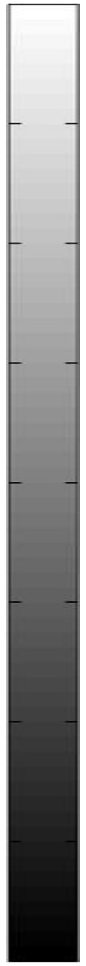
Apply to whole image and examine the difference

Original

Smoothed

Difference

200,20



0,-20

Image Smoothing

Binomial

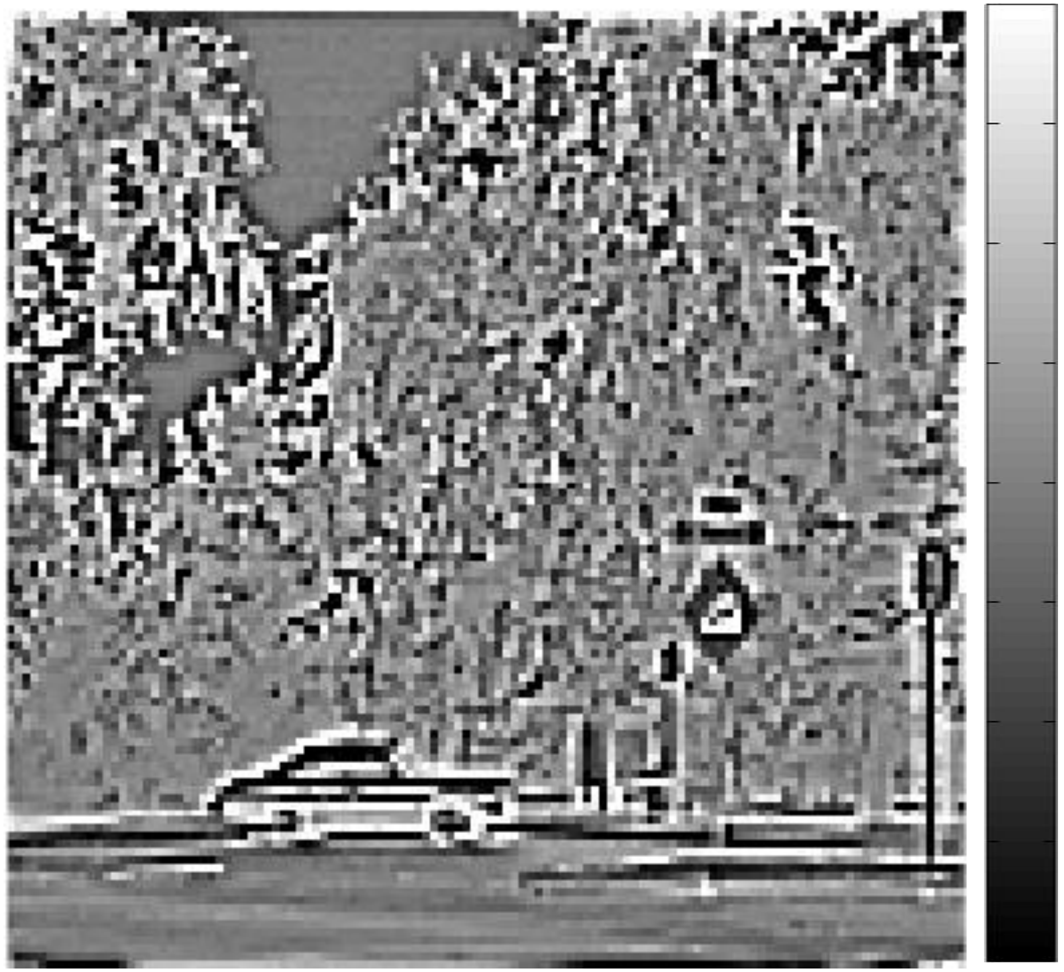
1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1

/256

Apply to whole image and examine the difference

Original

Smoothed



Difference

Sharpening Filters

The discrete version of the first derivatives $\frac{\partial}{\partial x} f(x,y)$ and $\frac{\partial}{\partial y} f(x,y)$

are $D_x = f(x,y) - f(x-1,y)$ and $D_y = f(x,y) - f(x,y-1)$ which in terms of kernels are

-1	1

and

-1
1

but may also be expressed as

-1	1
-1	1

and

-1	-1
1	1

or even with larger kernels.

Prewitt's 3×3 derivative kernels are

-1	-1	-1
0	0	0
1	1	1

and

-1	0	1
-1	0	1
-1	0	1

.

Derivative at center pixel.

Roberts cross gradient at 45° are

-1	0
0	1

and

0	-1
1	0

.

Sharpening Filters

Another version of the first derivatives are $\frac{\partial}{\partial x} f(x,y)$ and $\frac{\partial}{\partial y} f(x,y)$,

called Sobel operators, which in terms of kernels are

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

and

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

.

Derivative at center pixel.

These derivative operators are also said to have a small smoothing effect.

Sharpening Filters

Implementing the gradient $\nabla f(x,y)$ is a little more complicated because

$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial}{\partial x} f(x,y) \\ \frac{\partial}{\partial y} f(x,y) \end{bmatrix}$$ is a vector-valued 2×1 quantity.

So typically its magnitude is computed

$$|\nabla f(x,y)| = \left[\left(\frac{\partial}{\partial x} f(x,y) \right)^2 + \left(\frac{\partial}{\partial y} f(x,y) \right)^2 \right]^{1/2}$$

or approximated by

$$|\nabla f(x,y)| \approx \left| \frac{\partial}{\partial x} f(x,y) \right| + \left| \frac{\partial}{\partial y} f(x,y) \right| \quad .$$

x derivative

-1	1

y derivative

-1
1

Sharpening Filters

The gradient is implemented by applying the x and y derivative kernels,

squaring the resulting x and y derivative images,

summing the squared resulting x and y derivative images,

taking the square root of the summed squared derivative images.

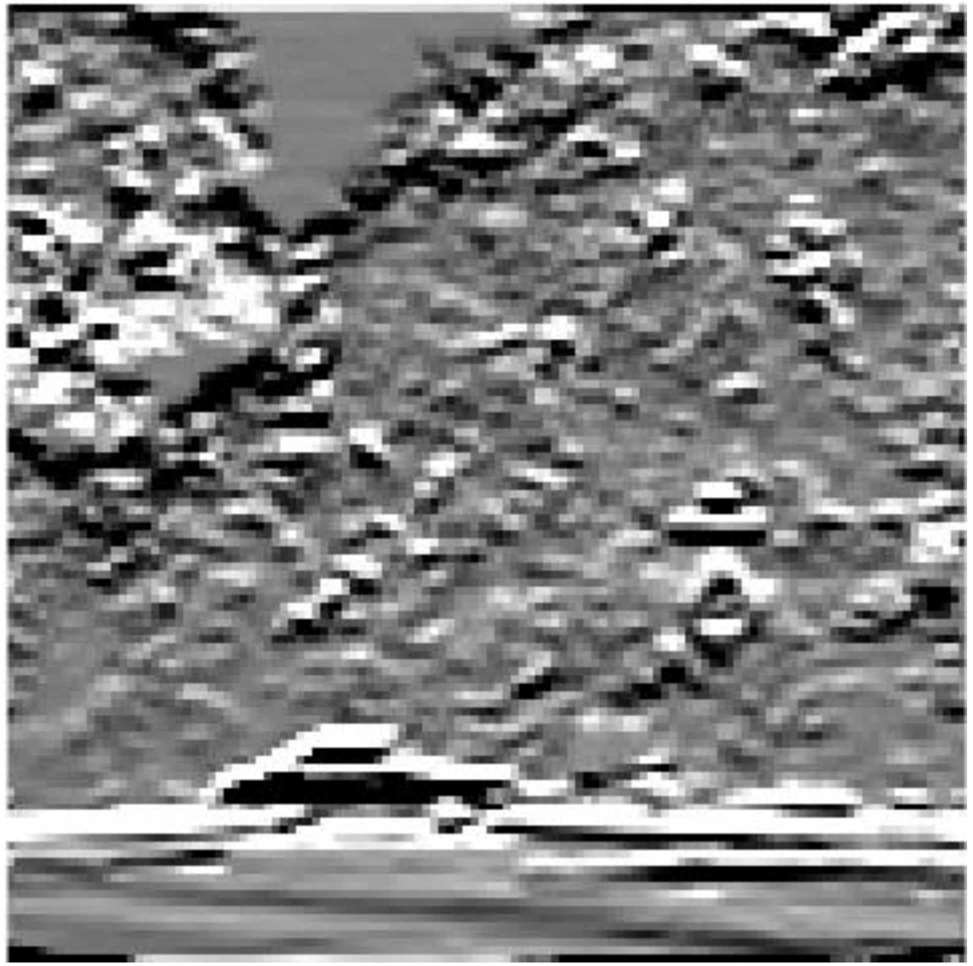
This is the magnitude of the gradient.

Sharpening Filters

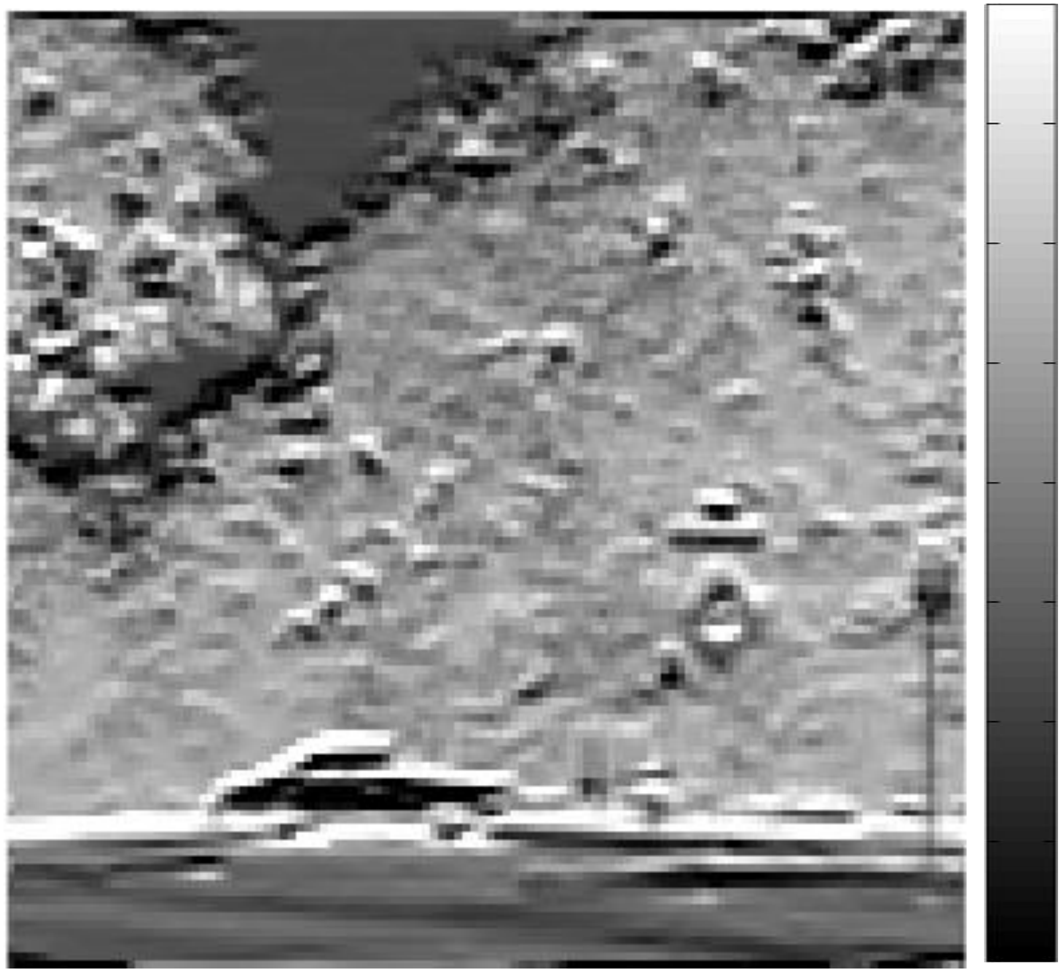
Derivative at center pixel.

-1	-1	-1
0	0	0
1	1	1

Original



U-D Gradient

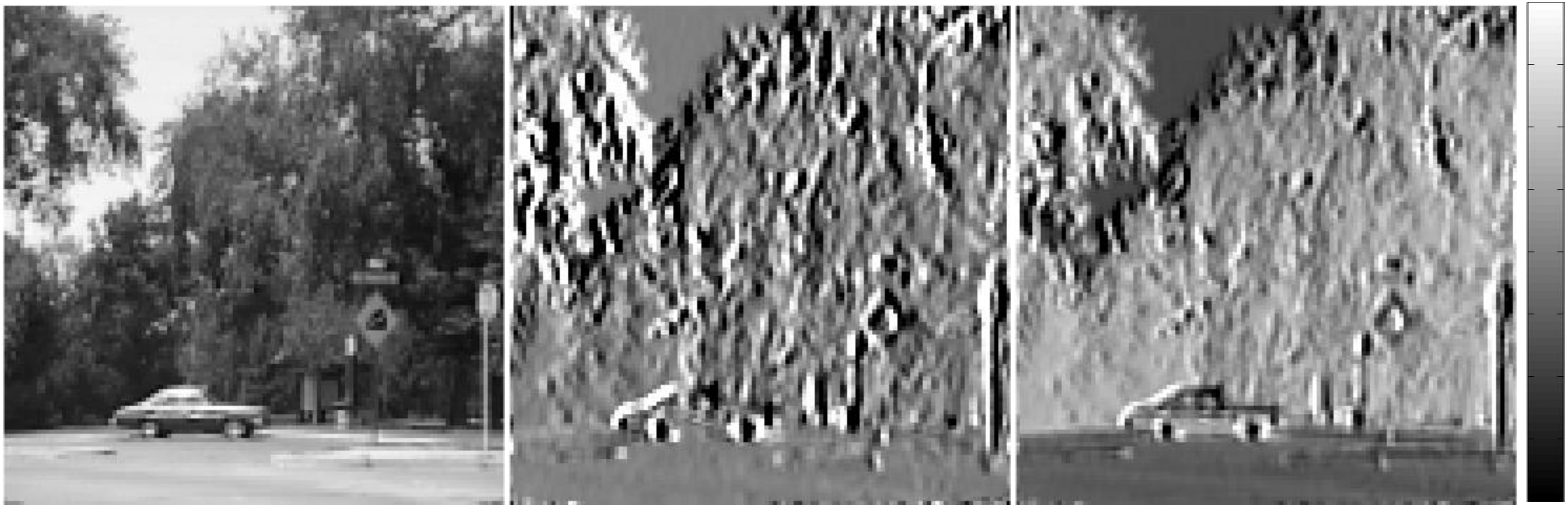


Difference

Sharpening Filters

Derivative at center pixel.

-1	0	1
-1	0	1
-1	0	1

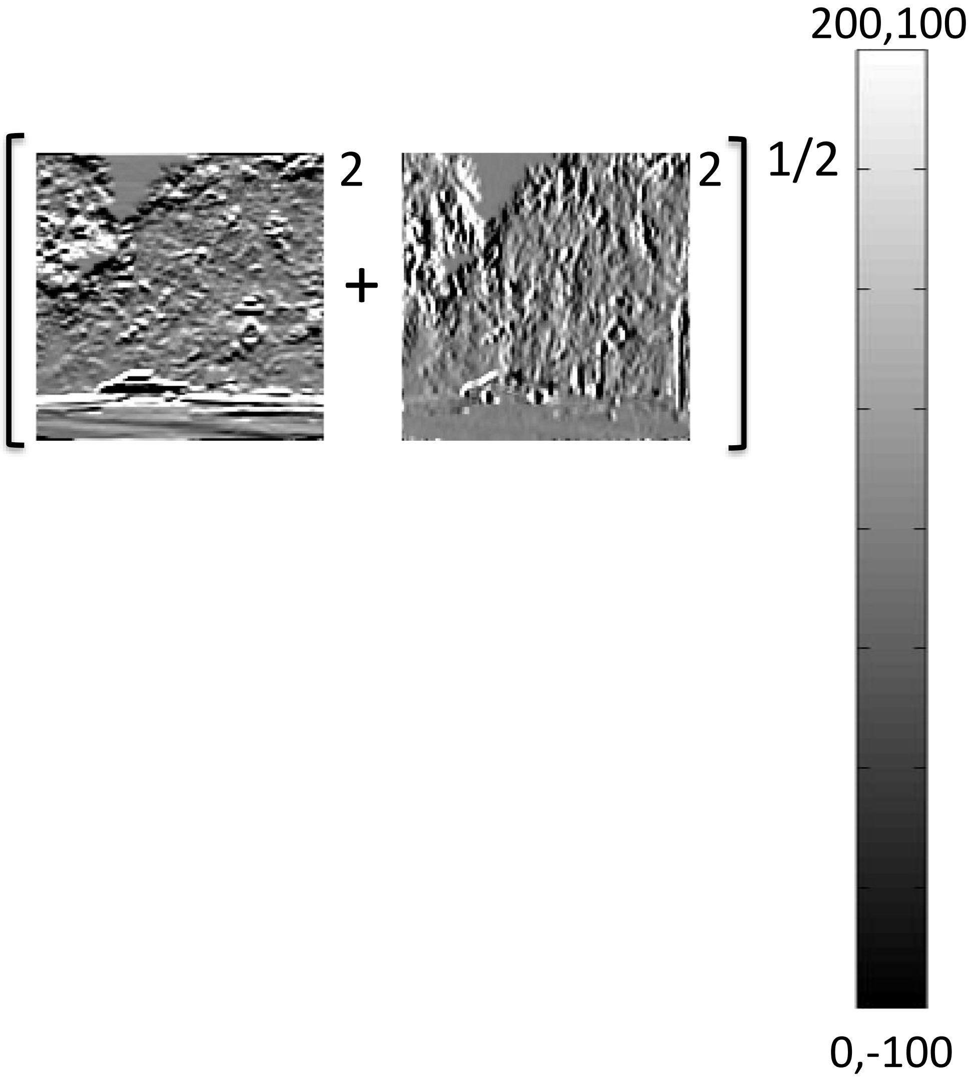


Original L-R Gradient Difference

Sharpening Filters

Original

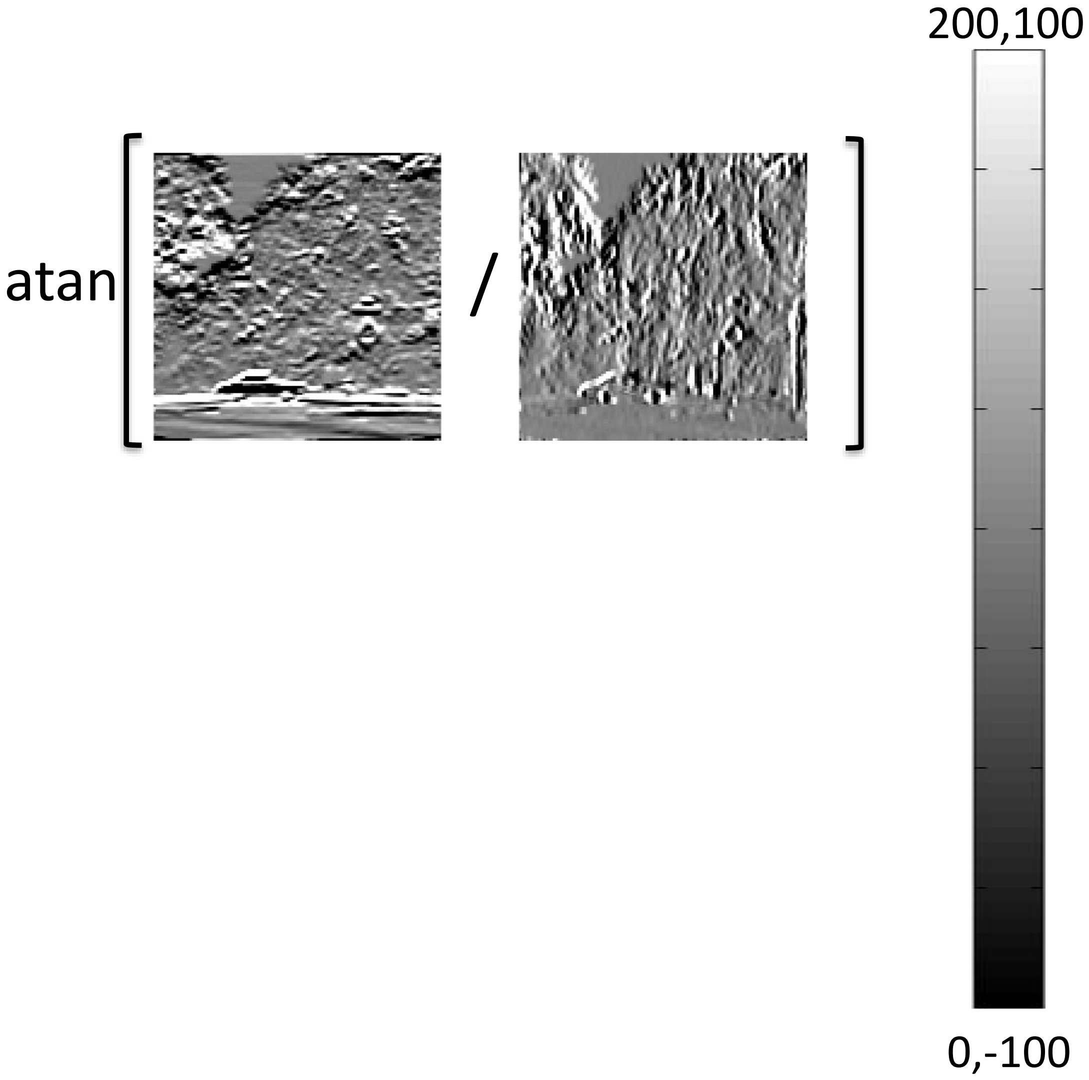
Magnitude Gradient



Sharpening Filters

Original

Direction Gradient



Sharpening Filters

The second derivative or Laplacian of an image $\nabla^2 f(x,y)$ is

$$\nabla^2 f(x,y) = \frac{\partial^2}{\partial x^2} f(x,y) + \frac{\partial^2}{\partial y^2} f(x,y)$$ a scalar.

$$\begin{aligned}\frac{\partial^2}{\partial x^2} f(x,y) &= \frac{\partial}{\partial x} D_x \\ &= \frac{\partial f(x,y)}{\partial x} - \frac{\partial f(x-1,y)}{\partial x} \\ &= (f(x,y) - f(x-1,y)) - (f(x-1,y) - f(x-2,y)) \\ &= f(x,y) - 2f(x-1,y) + f(x-2,y)\end{aligned}$$

$D_x = f(x,y) - f(x-1,y)$
$D_y = f(x,y) - f(x,y-1)$

Similarly

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

Sharpening Filters

Combining the two second derivatives

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

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

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

0	0	0
1	-2	1
0	0	0

+

0	1	0
0	-2	0
0	1	0

leads to

$\nabla^2 \approx$

0	1	0
1	-4	1
0	1	0

Sometimes more weight is given to the center pixel and the Laplacian is approximated by

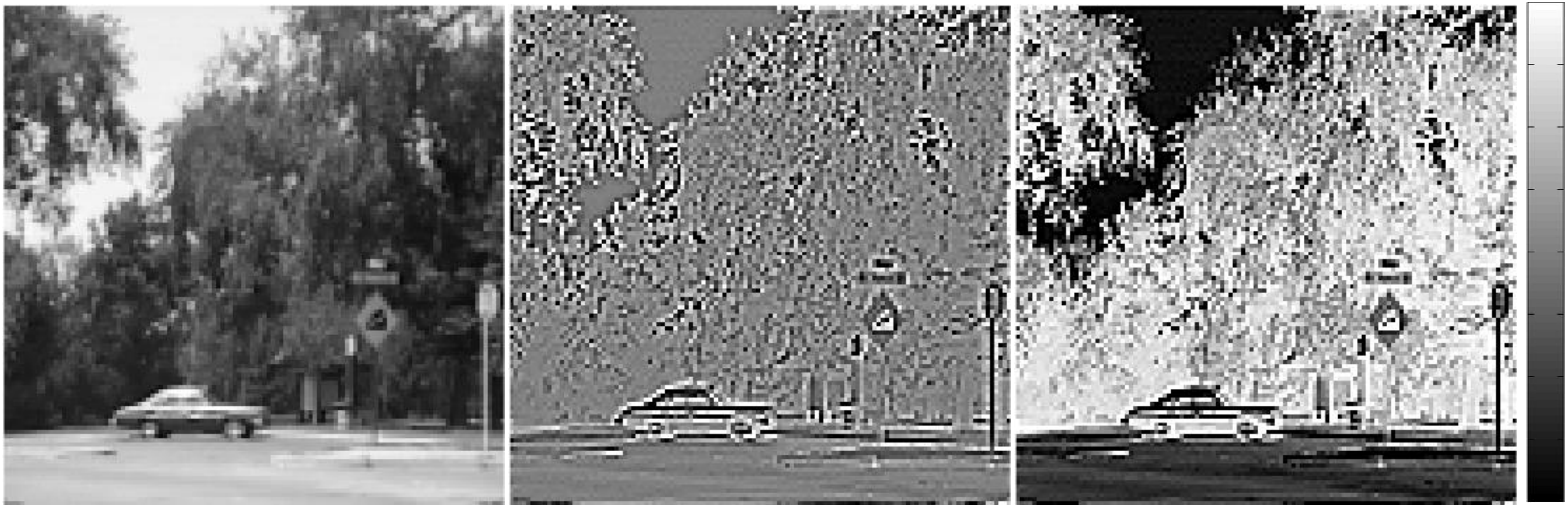
1	4	1
4	-20	4
1	4	1

.

Sharpening Filters

0	1	0
1	-4	1
0	1	0

Example of Laplacian filter applied.



Original Laplacian Difference

Sharpening Filters

Another common filter applied is the Laplacian of the Gaussian.

$$\nabla^2 g(x,y) = \frac{\partial^2}{\partial x^2} g(x,y) + \frac{\partial^2}{\partial y^2} g(x,y) \quad \text{where} \quad g(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

$$\nabla^2 g(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

0	0	1	0	0
0	1	2	1	0
1	2	-16	2	1
0	1	2	1	0
0	0	1	0	0

The result is zero in homogeneous regions.

Hybrid Filters

0	0	1	0	0
0	1	2	1	0
1	2	-16	2	1
0	1	2	1	0
0	0	1	0	0

Example of Laplacian of Gaussian filter applied.

Original — Laplacian of Gaussian — Difference

Hybrid Filters

We can enhance the edges or high frequencies.

We do this by subtracting the low from increased the original
to boost the high frequency edges.

High Boost=A*(Original)-(Low)

or alternatively

High Boost = (A-1)*(Original) + (High)

Hybrid Filters

We can boost the edges in an image with a “high boost” filter.

=

+

Original

Smoothed

High

This can be thought of as subtracting the low fom the original image.

Hybrid Filters

We add back part of the high to the original to boost the high.

=2

-

High Boost Original Low

HighBoost=A*(Original)-(Low)

HighBoost =255(HighBoost) /max(max(HighBoost)); %renormalize

Hybrid Filters

We add back part of the high to the original to boost the high.

=3

-

High Boost

Original

Low

HighBoost=A*(Original)-(Low)

HighBoost =255(HighBoost) /max(max(HighBoost)); %renormalize

Discussion

Each of the filters, accentuates a different aspect of the image.

Smoothing filters compute local averages to decrease noise in the image.

Derivative filters compute rates of change in the image.

Gradient filter computes magnitude of change at a pixel.

Laplacian filter computes sum of x-y second derivatives.

High Boost can accentuate edges and detail.

Discussion

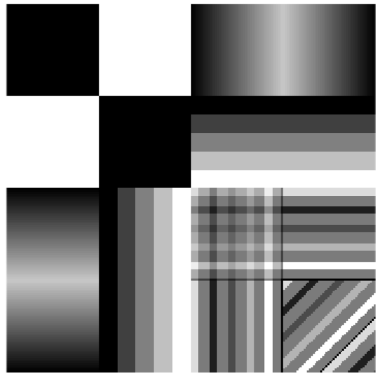
Questions?

Homework 3

1. Use the described process to make an 11×11 Gaussian Filter with $\sigma^2=2$. Show YOUR steps, apply to your own image, compare to the 5×5 Gaussian filter with $\sigma^2=0.5$.

2. Apply the Laplace 3×3 kernel to your image to produce a filtered image.

3. Apply one smoothing, one sharpening, and another filter to test image.



```
I = imread('MyTest.tif');
I=double(I);

figure;
imagesc(I,[0,255])
axis image, axis off
colormap(gray)
```

```
J = imread('MyTest.jpg');
J=double(J);

figure;
imagesc(J,[0,255])
axis image, axis off
colormap(gray)
```

*Note difference between I and J.

4*.Describe the effects of 5 different filters in the test image.

*For students in MSSC 5770.

Homework 3