

## Image filtering



Hybrid Images, Oliva et al., <http://cvcl.mit.edu/hybridimage.htm>

1

## Image filtering



Hybrid Images, Oliva et al., <http://cvcl.mit.edu/hybridimage.htm>

2

## Reading

Szeliski, Chapter 3.1-3.2

3

## What is an image?

4

## Images as functions

We can think of an **image** as a function,  $f$ , from  $\mathbb{R}^2$  to  $\mathbb{R}$ :

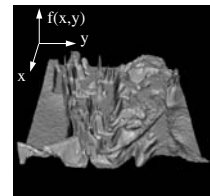
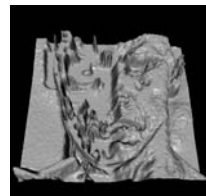
- $f(x, y)$  gives the **intensity** at position  $(x, y)$
- Realistically, we expect the image only to be defined over a rectangle, with a finite range:
  - $f: [a,b] \times [c,d] \rightarrow [0,1]$

A color image is just three functions pasted together. We can write this as a "vector-valued" function:

$$f(x, y) = \begin{bmatrix} r(x, y) \\ g(x, y) \\ b(x, y) \end{bmatrix}$$

5

## Images as functions



6

## What is a digital image?

We usually work with **digital (discrete)** images:

- **Sample** the 2D space on a regular grid
- **Quantize** each sample (round to nearest integer)

If our samples are  $\Delta$  apart, we can write this as:

$$f[i, j] = \text{Quantize}\{ f(i \Delta, j \Delta) \}$$

The image can now be represented as a matrix of integer values

|                |                 |     |     |     |     |     |    |     |
|----------------|-----------------|-----|-----|-----|-----|-----|----|-----|
|                | $j \rightarrow$ |     |     |     |     |     |    |     |
| $i \downarrow$ | 62              | 79  | 23  | 119 | 120 | 105 | 4  | 0   |
|                | 10              | 10  | 9   | 62  | 12  | 78  | 34 | 0   |
|                | 10              | 58  | 197 | 46  | 46  | 0   | 0  | 48  |
|                | 176             | 135 | 5   | 188 | 191 | 68  | 0  | 49  |
|                | 2               | 1   | 1   | 29  | 26  | 37  | 0  | 77  |
|                | 0               | 89  | 144 | 147 | 187 | 102 | 62 | 208 |
|                | 255             | 252 | 0   | 166 | 123 | 62  | 0  | 31  |
|                | 166             | 63  | 127 | 17  | 1   | 0   | 99 | 30  |

7

## Filtering noise

How can we "smooth" away noise in an image?

|   |   |   |     |     |     |     |     |   |   |
|---|---|---|-----|-----|-----|-----|-----|---|---|
| 0 | 0 | 0 | 0   | 0   | 0   | 0   | 0   | 0 | 0 |
| 0 | 0 | 0 | 0   | 0   | 0   | 0   | 0   | 0 | 0 |
| 0 | 0 | 0 | 100 | 130 | 110 | 120 | 110 | 0 | 0 |
| 0 | 0 | 0 | 110 | 90  | 100 | 90  | 100 | 0 | 0 |
| 0 | 0 | 0 | 130 | 100 | 90  | 130 | 110 | 0 | 0 |
| 0 | 0 | 0 | 120 | 100 | 130 | 110 | 120 | 0 | 0 |
| 0 | 0 | 0 | 90  | 110 | 80  | 120 | 100 | 0 | 0 |
| 0 | 0 | 0 | 0   | 0   | 0   | 0   | 0   | 0 | 0 |
| 0 | 0 | 0 | 0   | 0   | 0   | 0   | 0   | 0 | 0 |
| 0 | 0 | 0 | 0   | 0   | 0   | 0   | 0   | 0 | 0 |

8

### Mean filtering

$F[x, y]$

|   |   |    |    |    |    |    |    |   |   |
|---|---|----|----|----|----|----|----|---|---|
| 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 90 | 0  | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 90 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 |

$G[x, y]$

|  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

9



10

### Mean filtering

$F[x, y]$

|   |   |   |    |    |    |    |    |   |   |
|---|---|---|----|----|----|----|----|---|---|
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0  | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 90 | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |

$G[x, y]$

|  |    |    |    |    |    |    |    |    |  |
|--|----|----|----|----|----|----|----|----|--|
|  |    |    |    |    |    |    |    |    |  |
|  | 0  | 10 | 20 | 30 | 30 | 30 | 20 | 10 |  |
|  | 0  | 20 | 40 | 60 | 60 | 60 | 40 | 20 |  |
|  | 0  | 30 | 60 | 90 | 90 | 90 | 60 | 30 |  |
|  | 0  | 30 | 50 | 80 | 80 | 80 | 60 | 30 |  |
|  | 0  | 30 | 50 | 80 | 80 | 90 | 60 | 30 |  |
|  | 0  | 20 | 30 | 50 | 50 | 60 | 40 | 20 |  |
|  | 10 | 20 | 30 | 30 | 30 | 30 | 20 | 10 |  |
|  | 10 | 20 | 10 | 0  | 0  | 0  | 0  | 0  |  |
|  |    |    |    |    |    |    |    |    |  |

11

### Cross-correlation filtering

Let's write this down as an equation. Assume the averaging window is  $(2k+1) \times (2k+1)$ :

$$G[i, j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^k \sum_{v=-k}^k F[i+u, j+v]$$

We can generalize this idea by allowing different weights for different neighboring pixels:

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v] F[i+u, j+v]$$

This is called a **cross-correlation** operation and written:

$$G = H \otimes F$$

H is called the "filter," "kernel," or "mask."

The above allows negative filter indices. When you implement need to use:  $H[u+k, v+k]$  instead of  $H[u, v]$

12

## Mean kernel

What's the kernel for a 3x3 mean filter?

|   |   |   |    |    |    |    |    |   |   |
|---|---|---|----|----|----|----|----|---|---|
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0  | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |

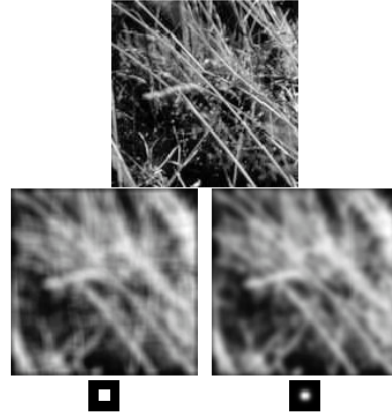
$F[x, y]$

|  |  |  |
|--|--|--|
|  |  |  |
|  |  |  |
|  |  |  |

$H[u, v]$

13

## Mean vs. Gaussian filtering



14

## Gaussian filtering

A Gaussian kernel gives less weight to pixels further from the center of the window

|   |   |   |    |    |    |    |    |   |   |
|---|---|---|----|----|----|----|----|---|---|
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 0  | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |
| 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |

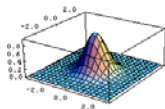
$F[x, y]$

|   |   |   |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

$H[u, v]$

This kernel is an approximation of a Gaussian function:

$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



What happens if you increase  $\sigma$ ?

Photoshop demo

15

## Image gradient

How can we differentiate a *digital* image  $F[x, y]$ ?

- Option 1: reconstruct a continuous image,  $f$ , then take gradient
- Option 2: take discrete derivative (finite difference)

$$\frac{\partial f}{\partial x}[x, y] \approx F[x + 1, y] - F[x, y]$$

How would you implement this as a cross-correlation?

|  |  |  |
|--|--|--|
|  |  |  |
|  |  |  |
|  |  |  |

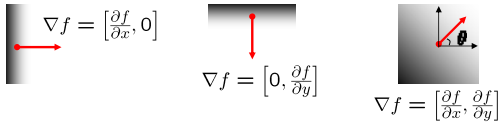
$H$

16

## Image gradient

$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

It points in the direction of most rapid change in intensity



The gradient direction is given by:

$$\theta = \tan^{-1} \left( \frac{\partial f / \partial y}{\partial f / \partial x} \right)$$

- how does this relate to the direction of the edge?

The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

17



original

smoothed (5x5 Gaussian)

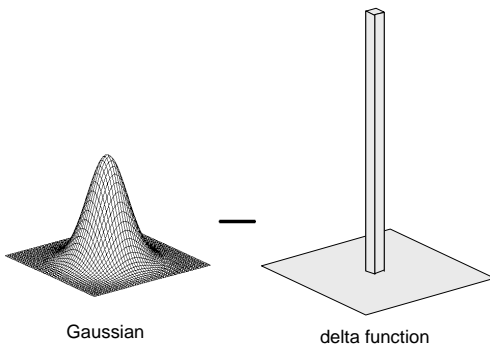


smoothed - original

(scaled by 4, offset +128)

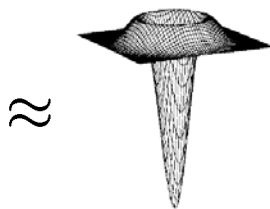
Why does this work?

18



Gaussian

delta function



≈

Laplacian of Gaussian  
(2<sup>nd</sup> derivative operator)

19

## Convolution

A **convolution** operation is a cross-correlation where the filter is flipped both horizontally and vertically before being applied to the image:

$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v] F[i - u, j - v]$$

It is written:  $G = H \star F$

Suppose H is a Gaussian or mean kernel. How does convolution differ from cross-correlation?

Suppose F is an impulse function (previous slide) What will G look like?

20

## Continuous Filters

We can also apply filters to *continuous* images.

In the case of cross correlation:  $g = h \otimes f$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(u, v) f(x + u, y + v) du dv$$

In the case of convolution:  $g = h \star f$

$$g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(u, v) f(x - u, y - v) du dv$$

Note that the image and filter are infinite.

21

## More on filters...

Cross-correlation/convolution is useful for, e.g.,

- ♦ Blurring
- ♦ Sharpening
- ♦ Edge Detection
- ♦ Interpolation

Convolution has a number of nice properties

- ♦ Commutative, associative
- ♦ Convolution corresponds to product in the Fourier domain

More sophisticated filtering techniques can often yield superior results for these and other tasks:

- ♦ Polynomial (e.g., bicubic) filters
- ♦ Steerable filters
- ♦ Median filters
- ♦ Bilateral Filters
- ♦ ...

(see text, web for more details on these)

22