Images & Image Filtering

Abe Davis,
Jan 27, 2020
CS5670: Introduction to Computer Vision
Today’s Lecture

• What are images?
  • How do they form?
  • How can we represent them mathematically?

• What is image filtering?
  • Why do we care?
  • How do we perform it mathematically?
Today’s Lecture

- What are images?
- How do they form?
- How can we represent them mathematically?
- What is image filtering?
- Why do we care?
- How do we perform it mathematically?

Side Note:

- Standing in for Noah today
- Slides are a mix of his slides from previous years and slides I made over the weekend
- If anything seems out of place, please don’t hesitate to ask about it
Reading

• Szeliski, Chapter 3.1-3.2
Announcements

• You should have been invited to Piazza
• We will add students to CMS this week
Announcements

• Project 1 (Hybrid Images) will be released tomorrow
  • This project will be done solo
  • Other projects planned to be done in groups of 2

• More on what hybrid images are toward the end of this lecture
Announcements

• We provide a walkthrough for setting up a python environment for the project

• As a backup, we also have a course virtual machine (VM) for you to run the assignments

• The assignment also works on lab machines
What is an image?
What is an image?

What do they represent?

How do they represent it?
What is an image?

What do they represent?

How do they represent it?

Images from A. Efros
How are Images Formed?

Observer

Image Plane
How are Images Formed?

Observer

Image Plane

Light
How are Images Formed?
How are Images Formed?

Observer

Image Plane

Light
Thinking About Images as Functions

Pixel Location

Pixel Brightness

f(x)
What is an image?

What do they represent?

How do they represent it?
What is an image?

- A grid (matrix) of intensity values

(common to use one byte per value: 0 = black, 255 = white)
What is an image?

• We can think of a (grayscale) image as a function, $f$, from $\mathbb{R}^2$ to $\mathbb{R}$:
  
  – $f(x,y)$ gives the intensity at position $(x,y)$

  – A digital image is a discrete (sampled, quantized) version of this function
Image transformations

• As with any function, we can apply operators to an image

\[ g(x,y) = f(x,y) + 20 \]

\[ g(x,y) = f(-x,y) \]

• Today we’ll talk about a special kind of operator, *convolution* (linear filtering)
Filters

• Filtering
  – Form a new image whose pixels are a combination of the original pixels

• Why?
  – To get useful information from images
    • E.g., extract edges or contours (to understand shape)
  – To enhance the image
    • E.g., to remove noise
    • E.g., to sharpen and “enhance image” a la CSI (sort of...)
Examples of Image Processing problems

• Image Restoration
  – denoising
  – deblurring

• Image Compression
  – JPEG, JPEG2000, MPEG..

• Computing Field Properties
  – optical flow
  – disparity

• Locating Structural Features
  – corners
  – edges
Question: Noise reduction

• Given a camera and a still scene, how can you reduce noise?

Take lots of images and average them!

What’s the next best thing?

Source: S. Seitz
Image Filtering: 
Thinking About Areas Instead of Just Points

Abe Davis

CS5670: Intro to Computer Vision
Putting Pixels in Context

A single pixel doesn’t tell us much out of context...

How do we represent this context mathematically?
Image Filtering: Operations on Image Regions

- Transforms each pixel into some function of the neighborhood around it

\[ p' = f_p(x_1, x_2, \ldots, x_9) \]
Image Filtering: Operations on Image Regions

- Transforms each pixel into some function of the neighborhood around it

\[ p' = f_p(x_1, x_2, \ldots, x_9) \]
Image Filtering: Operations on Image Regions

- Transforms each pixel into some function of the neighborhood around it

\[ p' = f_p(x_1, x_2, \ldots, x_9) \]
Linear Filtering

- Filters where the function \( p' = f_p(x_1, x_2, \ldots, x_9) \) is just a linear combination
Linear filtering

• One simple version of filtering: linear filtering (cross-correlation, convolution)
  – Replace each pixel by a linear combination (a weighted sum) of its neighbors

• The prescription for the linear combination is called the “kernel” (or “mask”, “filter”)

<table>
<thead>
<tr>
<th>10</th>
<th>5</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>8</td>
</tr>
</tbody>
</table>

Local image data

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Kernel

| 8 |

Modified image data

Source: L. Zhang
Cross-correlation

Let $F$ be the image, $H$ be the kernel (of size $2k+1 \times 2k+1$), and $G$ be the output image

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

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

$$G = H \otimes F$$

- Can think of as a “dot product” between local neighborhood and kernel for each pixel
Convolution

• Same as cross-correlation, except that the kernel is “flipped” (horizontally and vertically)

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

This is called a convolution operation:

\[ G = H \ast F \]

• Convolution is commutative and associative
Convolution

• Same as cross-correlation, except that the kernel is “flipped” (both horizontally and vertically)

This is called a convolution operation:

\[
G[i, j] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} A[m, n] \cdot B[i-m, j-n]
\]

The convolution operation is defined as a sum of products of the kernel and the input signal.

**Side Note:**

- **Associativity:** \((A*B)*C = A*(B*C)\)
- **Commutativity:** \((A*B) = (B*A)\)

• Convolution is **commutative** and **associative**
Why Correlation is *not* Commutative

• What does it mean for filtering to be commutative?
  • \( f(A,B) = f(B,A) \)

**Question:**
How do we make the same parts of A and B match up regardless of order?
Why Convolution *is* Commutative

• What does it mean for filtering to be commutative?
  • \( f(A,B) = f(B,A) \)

Answer:
Flip one of them
Convolution

Adapted from F. Durand
Mean filtering

\[ H \ast F = G \]
Mean filtering/Moving average

\[ F[x, y] \]

\[ G[x, y] \]
Mean filtering/Moving average

\[ F[x, y] \]

\[ G[x, y] \]
Mean filtering/Moving average

\[ F[x, y] \]

\[ G[x, y] \]
Mean filtering/Moving average

\[ F[x, y] \]

\[ G[x, y] \]
Mean filtering/Moving average

\[ F[x, y] \]

\[ G[x, y] \]
Mean filtering/Moving average

\[ F[x, y] \]

\[
\begin{array}{cccccccccccc}
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 60 & 90 & 90 & 90 & 90 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 90 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

\[ G[x, y] \]

\[
\begin{array}{cccccccccccc}
 0 & 10 & 20 & 30 & 30 & 30 & 20 & 10 & 0 & 0 & 0 & 0 \\
 0 & 20 & 40 & 60 & 60 & 60 & 40 & 20 & 0 & 0 & 0 & 0 \\
 0 & 30 & 60 & 90 & 90 & 90 & 60 & 30 & 0 & 0 & 0 & 0 \\
 0 & 30 & 60 & 80 & 80 & 80 & 60 & 30 & 0 & 0 & 0 & 0 \\
 0 & 30 & 60 & 80 & 80 & 80 & 60 & 30 & 0 & 0 & 0 & 0 \\
 0 & 20 & 30 & 50 & 50 & 50 & 40 & 20 & 0 & 0 & 0 & 0 \\
 10 & 20 & 30 & 30 & 30 & 30 & 20 & 10 & 0 & 0 & 0 & 0 \\
 10 & 10 & 10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]
Linear filters: examples

Original \* \[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{array}
\] = Identical image

Source: D. Lowe
Linear filters: examples

Original

\[ \begin{array}{ccc}
0 & 0 & 0 \\
1 & 0 & 0 \\
0 & 0 & 0 \\
\end{array} \]

= Shifted left By 1 pixel

Source: D. Lowe
Linear filters: examples

Original

\[ \ast \quad \frac{1}{9} \]

Blur (with a mean filter)

Source: D. Lowe
Linear filters: examples

Can anyone guess a filter we might use to sharpen an image?

Source: D. Lowe
Linear filters: examples

Original

\[ \begin{bmatrix}
0 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0
\end{bmatrix} \ast \frac{1}{9} \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1
\end{bmatrix} = \]

Sharpening filter
(accentuates edges)

Source: D. Lowe
Sharpening

before

after

Source: D. Lowe
Smoothing with box filter revisited

Source: D. Forsyth
Smoothing with box filter revisited

Can anyone think of a better smoothing kernel?

Source: D. Forsyth
Gaussian Kernel

\[ G_{\sigma} = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2 + y^2)}{2\sigma^2}} \]

Source: C. Rasmussen
Gaussian filters
Mean vs. Gaussian filtering
Gaussian filter

- Removes “high-frequency” components from the image (low-pass filter)
- Convolution with self is another Gaussian

- Convolving twice with Gaussian kernel of width $\sigma$
  $\ast$ = convolving once with kernel of width $\sigma\sqrt{2}$

Source: K. Grauman
Sharpening revisited

• What does blurring take away?

Let’s add it back:

(This “detail extraction” operation is also called a high-pass filter)

Photo credit: https://www.flickr.com/photos/geezaweezer/16089096376/
Sharpen filter

\[ F + \alpha (F - F \ast H) = (1 + \alpha) F - \alpha (F \ast H) = F \ast ([1 + \alpha] e - \alpha H) \]
Sharpen filter

\[ F + \alpha (F - F \ast H) = (1 + \alpha) F - \alpha (F \ast H) = F \ast ([1 + \alpha] e - \alpha H) \]

Multiplying out alpha and collecting like terms
Multiplying out alpha and collecting like terms

\[ F + \alpha (F - F * H) = (1 + \alpha) F - \alpha (F * H) = F * \left( [1 + \alpha] e - \alpha H \right) \]

Distribute to represent as convolution with a single kernel

(a unit impulse)
Sharpen filter

In other words:
Boosting the detail layer of an image (i.e., sharpening) can be represented as a single convolution.
Sharpen filter

\[ F + \alpha (F - F \ast H) = (1 + \alpha) F - \alpha (F \ast H) = F \ast ([1 + \alpha] e - \alpha H) \]
Sharpen filter

unfiltered

filtered
“Optical” Convolution

Camera shake

\[ \begin{array}{c}
\text{Source: Fergus, et al. “Removing Camera Shake from a Single Photograph”, SIGGRAPH 2006}
\end{array} \]

**Bokeh:** Blur in out-of-focus regions of an image.

Source: http://lullaby.homepage.dk/diy-camera/bokeh.html
Filters: Thresholding

\[ g(m, n) = \begin{cases} 
255, & f(m, n) > A \\
0, & \text{otherwise} 
\end{cases} \]
Question:
Is thresholding a linear filter?

\[ g(m, n) = \begin{cases} 255, & f(m, n) > A \\ 0, & \text{otherwise} \end{cases} \]
**Why is it Called Filtering?**

Filtering lets us reason about images at different scales, e.g.:

- Mean filtering an image removes fine-scale detail and leaves only coarse-scale information
- Sharpening an image amplifies fine-scale details
Hybrid Images: Do These People Look Happy or Sad?

Hybrid Images, Oliva et al., http://cvcl.mit.edu/hybridimage.htm
Hybrid Images: Do These People Look Happy or Sad?

Hybrid Images, Oliva et al., http://cvcl.mit.edu/hybridimage.htm
Hybrid Images: Do These People Look Happy or Sad?

Hybrid Images, Oliva et al., http://cvcl.mit.edu/hybridimage.htm
Side Note: Remember Yanny and Laurel?
One Final Note: Non-Linear Filtering?

• Q: What’s the most popular way to extend filtering to non-linear functions?

• A: Convolutional Neural Networks
  • Implemented as a series of convolutions separated by nonlinearities
  • More on this later in the course

**One more reason why we care about filtering and convolution**
Questions?