Image Processing
Reading

Course Reader:
Jain et. Al. *Machine Vision*
   Chapter 4 and 5
Definitions

• Many graphics techniques that operate only on images

• **Image processing**: operations that take images as input, produce images as output

• In its most general form, an image is a function $f$ from $\mathbb{R}^2$ to $\mathbb{R}$
  
  – $f(x, y)$ gives the intensity of a channel at position $(x, y)$ 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:

                             \[
                             f(x, y) = (f_r(x, y), f_g(x, y), f_b(x, y))
                             \]
Images as Functions
What is a digital image?

• In computer graphics, we usually operate on digital (discrete) images:
  – Sample the space on a regular grid
  – Quantize each sample (round to nearest integer)

• If our samples are $d$ apart, we can write this as:

\[ f'[i, j] = \text{Quantize}(f(i \cdot d, j \cdot d)) \]
Sampled digital image
Image processing

- An image processing operation typically defines a new image \( g \) in terms of an existing image \( f \).
- The simplest operations are those that transform each pixel in isolation. These pixel-to-pixel operations can be written:

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

- Example: threshold, RGB → grayscale
Pixel Movement

- Some operations preserve intensities, but move pixels around in the image

\[ g(x, y) = f(u(x, y), v(x, y)) \]

- Examples: many amusing warps of images
Multiple input images

- Some operations define a new image $g$ in terms of $n$ existing images ($f_1, f_2, \ldots, f_n$), where $n$ is greater than 1

- Example: cross-dissolve between 2 input images

$$g(x, y) = \sum_i w_i f_i(x, y)$$
Noise

- Common types of noise:
  - Salt and pepper noise: contains random occurrences of black and white pixels
  - Impulse noise: contains random occurrences of white pixels
  - Gaussian noise: variations in intensity drawn from a Gaussian normal distribution
Noise Examples

Original
Salt and pepper noise
Impulse noise
Gaussian noise
Ideal noise reduction
Ideal noise reduction
Practical noise reduction

- How can we “smooth” away noise in a single image?

<p>| | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>130</td>
<td>110</td>
<td>120</td>
<td>110</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>110</td>
<td>90</td>
<td>100</td>
<td>90</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>130</td>
<td>100</td>
<td>90</td>
<td>130</td>
<td>110</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>120</td>
<td>100</td>
<td>130</td>
<td>110</td>
<td>120</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>110</td>
<td>80</td>
<td>120</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
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 \(H[u,v]\)
Mean kernel

- What’s the kernel for a 3x3 mean filter?

\[ H[u, v] \]

\[ F[x, y] \]
Mean Filters

3x3

5x5

7x7

Gaussian noise

Salt and pepper noise
Gaussian Filtering

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

\[
h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}
\]
Gaussian Filters

- Gaussian filters weigh pixels based on their distance to the location of convolution.

\[ h[i, j] = e^{-\left(\frac{i^2 + j^2}{2\sigma^2}\right)} \]

- Blurring noise while preserving features of the image
- Smoothing the same in all directions
- More significance to neighboring pixels
- Width parameterized by \( s \)
- Gaussian functions are separable
- Convolving with multiple Gaussian filters results in a single Gaussian filter
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: \[ G = H \ast F \]

• Suppose H is a Gaussian or mean kernel. How does convolution differ from cross-correlation?
Gaussian Filters
Mean vs. Gaussian filtering
Median Filters

• A Median Filter operates over a $k \leq k$ region by selecting the median intensity in the region.

• What advantage does a median filter have over a mean filter?

• Is a median filter a kind of convolution?
Median Filters
Sampling theorem

• This result is known as the Sampling Theorem and is due to Claude Shannon who first discovered it in 1949:
  A signal can be reconstructed from its samples without loss of information, if the original signal has no frequencies above $\frac{1}{2}$ the sampling frequency.
• For a given bandlimited function, the minimum rate at which it must be sampled is the Nyquist frequency.
Reconstruction filters

• The sinc filter, while “ideal”, has two drawbacks:
  – It has large support (slow to compute)
  – It introduces ringing in practice

• We can choose from many other filters…
Cubic filters

• Mitchell and Netravali (1988) experimented with cubic filters, reducing them all to the following form:

\[
r(x) = \begin{cases} 
\frac{1}{6} \left( (12 - 9B - 6C)|x|^3 + (-18 + 12B + 6C)|x|^2 + (6 - 2B)(x^3) \right) & |x| < 1 \\
0 & 1 \leq |x| < 2 \\
\text{otherwise} & \end{cases}
\]

• The choice of B or C trades off between being too blurry or having too much ringing. B=C=1/3 was their “visually best” choice: “Mitchell filter.”
Reconstruction filters in 2D

\[ \text{sinc}(x) \text{sinc}(y) \]

\[ \Pi(x) \Pi(y) \Rightarrow \]

\[ \Lambda(x) \Lambda(y) \Rightarrow \]

\[ \text{mitchell}(x) \text{mitchell}(y) \Rightarrow \]
Edge detection

• One of the most important uses of image processing is edge detection:
  – Really easy for humans
  – Really difficult for computers
  – Fundamental in computer vision
  – Important in many graphics applications

• How to tell if a pixel is on an edge?
Edge Detection

• One of the most important uses of image processing is **edge detection**
  – Really easy for humans
  – Really difficult for computers
  – Fundamental in computer vision
  – Important in many graphics applications

• What defines an edge?
Gradient

- The gradient is the 2D equivalent of the derivative:

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

- Properties of the gradient
  - It’s a vector
  - Points in the direction of maximum increase of \( f \)
  - Magnitude is rate of increase

- How can we approximate the gradient in a discrete image?
Less than ideal edges
Edge Detection Algorithms

Edge detection algorithms typically proceed in three or four steps:

- Filtering: cut down on noise
- Enhancement: amplify the difference between edges and non-edges
- Detection: use a threshold operation
- Localization (optional): estimate geometry of edges beyond pixels
Edge Enhancement

• A popular gradient magnitude computation is the **Sobel operator**:

\[ s_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \]

\[ s_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \]

• We can then compute the magnitude of the vector \((s_x, s_y)\)
Sobel Operator
Second derivative operators

• The Sobel operator can produce thick edges. Ideally, we’re looking for infinitely thin boundaries.

• An alternative approach is to look for local extrema in the first derivative: places where the change in the gradient is highest.

• Q: A peak in the first derivative corresponds to what in the second derivative?
Localization with the Laplacian

- An equivalent measure of the second derivative in 2D is the Laplacian:
  \[ \nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \]

- Using the same arguments we used to compute the gradient filters, we can derive a Laplacian filter to be:
  \[ \Delta^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \]

- Zero crossings of this filter correspond to positions of maximum gradient. These zero crossings can be used to localize edges.
Laplacian alternatives

\[
\begin{array}{ccc}
0 & 1 & 0 \\
1 & -4 & 1 \\
0 & 1 & 0 \\
\end{array} \\
\begin{array}{ccc}
1 & 1 & 1 \\
1 & -8 & 1 \\
1 & 1 & 1 \\
\end{array} \\
\begin{array}{ccc}
-1 & 2 & -1 \\
2 & -4 & 2 \\
-1 & 2 & -1 \\
\end{array}
\]
Localization with the Laplacian

Original

Smoothed

Laplacian (+128)
Marching squares

- We can convert these signed values into edge contours using a “marching squares” technique:
Sharpening with the Laplacian

Original

Laplacian (+128)

Original + Laplacian

Original - Laplacian
Laplacian of Gaussian
Summary

• Formal definitions of image and image processing
• Kinds of image processing: pixel-to-pixel, pixel movement, convolution, others
• Types of noise and strategies for noise reduction
• Definition of convolution and how discrete convolution works
• The effects of mean, median and Gaussian filtering
• How edge detection is done
• Gradients and discrete approximations