Image Formation

Digital Camera

The Eye

Film

Scene element

Imaging system

Illumination (energy) source

(Internal) image plane
What is an image?

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:

\[
\begin{bmatrix}
  r(x, y) \\
  g(x, y) \\
  b(x, y)
\end{bmatrix}
\]
Images as functions
What’s in the “pixel intensity”?

\[ f(x, y) = \text{reflectance}(x, y) \times \text{illumination}(x, y) \]

Reflectance in \([0,1]\), illumination in \([0, \infty]\)

**FIGURE 2.15** An example of the digital image acquisition process. (a) Energy (“illumination”) source, (b) An element of a scene, (c) Imaging system, (d) Projection of the scene onto the image plane, (e) Digitized image.
Problem: Dynamic Range

The real world is High dynamic range
Long Exposure

Real world

10^{-6} — High dynamic range — 10^{6}

Picture

10^{-6} — 0 to 255 — 10^{6}

• What does pixel value 255 mean?
Short Exposure

Real world: $10^{-6}$ to $10^6$

High dynamic range

Picture: $10^{-6}$ to $10^6$

0 to 255

- What does pixel value 0 mean?
Is Camera a photometer?

Pixel \((312, 284) = 42\)

42 photos?
Scene radiance (W/sr/m²)

Sensor irradiance

Exposure (Δt)

Analog voltages

Digital values

Pixel values

Camera is NOT a photometer!
Simple Point Processing: Enhancement

FIGURE 3.9
(a) Aerial image.
(b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0, 4.0, \text{ and } 5.0$, respectively.
(Original image for this example courtesy of NASA.)
Point Processing

The simplest kind of range transformations are these independent of position $x,y$:

$$ g = T(f) $$

This is called point processing.

e.g. Gain and Bias transform:

$$ g(x,y) = a \cdot f(x,y) + b $$

**Important:** every pixel for himself – spatial information completely lost!
Power-law transformations

\[ s = cr^\gamma \]
FIGURE 3.3 Some basic gray-level transformation functions used for image enhancement.
Negative

FIGURE 3.4
(a) Original digital mammogram.
(b) Negative image obtained using the negative transformation in Eq. (3.2-1).
(Courtesy of G.E. Medical Systems.)
FIGURE 3.5
(a) Fourier spectrum.
(b) Result of applying the log transformation given in Eq. (3.2-2) with $c = 1$. 
Contrast Stretching

**FIGURE 3.10**
Contrast stretching. (a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)
Image Histograms

s = T(r)

FIGURE 3.15 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)
Histogram Equalization

FIGURE 3.17 (a) Images from Fig. 3.15. (b) Results of histogram equalization. (c) Corresponding histograms.
Color Transfer [Reinhard, et al, 2001]

Limitations of Point Processing

Q: What happens if I reshuffle all pixels within the image?

A: It’s histogram won’t change. No point processing will be affected...
Sampling and Reconstruction
Sampling and Quantization

FIGURE 2.16 Generating a digital image: (a) Continuous image, (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization, (c) Sampling and quantization, (d) Digital scan line.
Sampled representations

• How to store and compute with continuous functions?
• Common scheme for representation: samples
  – write down the function’s values at many points
Reconstruction

- Making samples back into a continuous function
  - for output (need realizable method)
  - for analysis or processing (need mathematical method)
  - amounts to “guessing” what the function did in between

© 2006 Steve Marschner • 25
1D Example: Audio

- Low frequencies
- High frequencies
Sampling in digital audio

- Recording: sound to analog to samples to disc
- Playback: disc to samples to analog to sound again
  - how can we be sure we are filling in the gaps correctly?
Sampling and Reconstruction

- Simple example: a sign wave
Undersampling

• What if we “missed” things between the samples?
• Simple example: undersampling a sine wave
  – unsurprising result: information is lost
Undersampling

• What if we “missed” things between the samples?
• Simple example: undersampling a sine wave
  – unsurprising result: information is lost
  – surprising result: indistinguishable from lower frequency
Undersampling

- What if we “missed” things between the samples?
- Simple example: undersampling a sine wave
  - unsurprising result: information is lost
  - surprising result: indistinguishable from lower frequency
  - also, was always indistinguishable from higher frequencies
  - aliasing: signals “traveling in disguise” as other frequencies
Aliasing in video

Imagine a spoked wheel moving to the right (rotating clockwise). Mark wheel with dot so we can see what’s happening.

If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):

Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)
Aliasing in images

Disintegrating textures
Antialiasing

What can we do about aliasing?

Sample more often
- Join the Mega-Pixel craze of the photo industry
- But this can’t go on forever

Make the signal less “wiggly”
- Get rid of some high frequencies
- Will loose information
- But it’s better than aliasing
Preventing aliasing

- Introduce lowpass filters:
  - remove high frequencies leaving only safe, low frequencies
  - choose lowest frequency in reconstruction (disambiguate)
Linear filtering: a key idea

• Transformations on signals; e.g.:
  – bass/treble controls on stereo
  – blurring/sharpening operations in image editing
  – smoothing/noise reduction in tracking

• Key properties
  – linearity: \( \text{filter}(f + g) = \text{filter}(f) + \text{filter}(g) \)
  – shift invariance: behavior invariant to shifting the input
    • delaying an audio signal
    • sliding an image around

• Can be modeled mathematically by \textit{convolution}
Moving Average

• basic idea: define a new function by averaging over a sliding window

• a simple example to start off: smoothing
Moving Average

- Can add weights to our moving average
- $Weights \ [\ldots, 0, 1, 1, 1, 1, 1, 0, \ldots] / 5$
In 2D: box filter

\[
h[\cdot, \cdot] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}
\]
Image filtering

\[
g[m, n] = \sum_{k,l} h[k, l] f[m + k, n + l]
\]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ \begin{array}{ccccccccccc} 
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 0 & 90 & 0 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 90 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array} \]

\[ g[\ldots] \]

\[ g[m,n] = \sum_{k,l} h[k,l] \ f[m+k,n+l] \]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]  

\[ g[\ldots] \]  

Credit: S. Seitz
Image filtering

\[ f[\ldots] \quad g[\ldots] \]

\[ h[\ldots] \frac{1}{9} \]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ g[\ldots] \]

Credit: S. Seitz
Image filtering

$f[\cdot,\cdot]$  

$g[\cdot,\cdot]$

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

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ g[\ldots] \]

\[ h[\ldots] \]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ g[\ldots] \]

\[ g[m, n] = \sum_{k,l} h[k, l] f[m + k, n + l] \]

Credit: S. Seitz
Box Filter

What does it do?

- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

$h[\cdot, \cdot]$
Linear filters: examples

Original

Blur (with a mean filter)

Source: D. Lowe
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
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

Filtered (no change)

Source: D. Lowe
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

Shifted left
By 1 pixel

Source: D. Lowe
Other filters

Sobel

<table>
<thead>
<tr>
<th>1</th>
<th>0</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
</tbody>
</table>

Vertical Edge (absolute value)
Other filters

Sobel

Horizontal Edge (absolute value)
Back to the box filter
Moving Average

• Can add weights to our moving average

• \textit{Weights} \ [\ldots, 0, 1, 1, 1, 1, 1, 0, \ldots] / 5
Weighted Moving Average

- bell curve (gaussian-like) weights [..., 1, 4, 6, 4, 1, ...]
Moving Average In 2D

What are the weights $H$?

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</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>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</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>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</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>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
</tr>
</tbody>
</table>

$H[u, v]$

$F[x, y]$
Gaussian filtering

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

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

This kernel is an approximation of a Gaussian function:

\[ H[u, v] \]

\[ F[x, y] \]
Mean vs. Gaussian filtering
Important filter: Gaussian

Weight contributions of neighboring pixels by nearness

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

5 x 5, $\sigma = 1$

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.003</td>
<td>0.013</td>
<td>0.022</td>
<td>0.013</td>
<td>0.003</td>
<td></td>
</tr>
<tr>
<td>0.013</td>
<td>0.059</td>
<td>0.097</td>
<td>0.059</td>
<td>0.013</td>
<td></td>
</tr>
<tr>
<td>0.022</td>
<td>0.097</td>
<td>0.159</td>
<td>0.097</td>
<td>0.022</td>
<td></td>
</tr>
<tr>
<td>0.013</td>
<td>0.059</td>
<td>0.097</td>
<td>0.059</td>
<td>0.013</td>
<td></td>
</tr>
<tr>
<td>0.003</td>
<td>0.013</td>
<td>0.022</td>
<td>0.013</td>
<td>0.003</td>
<td></td>
</tr>
</tbody>
</table>
Gaussian Kernel

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

- Standard deviation \( \sigma \): determines extent of smoothing

Source: K. Grauman
Gaussian filters

\[ \sigma = 1 \text{ pixel} \]
\[ \sigma = 5 \text{ pixels} \]
\[ \sigma = 10 \text{ pixels} \]
\[ \sigma = 30 \text{ pixels} \]
Choosing kernel width

- The Gaussian function has infinite support, but discrete filters use finite kernels

Source: K. Grauman
Practical matters

How big should the filter be?

Values at edges should be near zero

Rule of thumb for Gaussian: set filter half-width to about $3 \sigma$
Cross-correlation vs. Convolution

cross-correlation: \[ G = H \otimes F \]

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

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:

\[ G = H \ast F \]

Convolution is commutative and associative
Convolution

Adapted from F. Durand
Convolution is nice!

- Notation: \( b = c \ast a \)

- Convolution is a multiplication-like operation
  - commutative: \( a \ast b = b \ast a \)
  - associative: \( a \ast (b \ast c) = (a \ast b) \ast c \)
  - distributes over addition: \( a \ast (b + c) = a \ast b + a \ast c \)
  - scalars factor out: \( \alpha a \ast b = a \ast \alpha b = \alpha(a \ast b) \)
  - identity: unit impulse \( e = [\ldots, 0, 0, 1, 0, 0, \ldots] \)
    \[
    a \ast e = a
    \]

- Conceptually no distinction between filter and signal

- Usefulness of associativity
  - often apply several filters one after another: \(((a \ast b_1) \ast b_2) \ast b_3\)
  - this is equivalent to applying one filter: \(a \ast (b_1 \ast b_2 \ast b_3)\)
Gaussian and convolution

• Removes “high-frequency” components from the image (low-pass filter)

• Convolution with self is another Gaussian

\[ \ast \quad = \]

– Convolving twice with Gaussian kernel of width \( \sigma \) = convolving once with kernel of width \( \sigma \sqrt{2} \)

Source: K. Grauman
The Convolution Theorem

The greatest thing since sliced (banana) bread!

- The Fourier transform of the convolution of two functions is the product of their Fourier transforms
  \[ F[g * h] = F[g]F[h] \]

- The inverse Fourier transform of the product of two Fourier transforms is the convolution of the two inverse Fourier transforms
  \[ F^{-1}[gh] = F^{-1}[g] \ast F^{-1}[h] \]

- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!
2D convolution theorem example

\[ f(x,y) \quad * \quad h(x,y) \quad \rightarrow \quad g(x,y) \]

\[ |F(s_x,s_y)| \quad \times \quad |H(s_x,s_y)| \quad \rightarrow \quad |G(s_x,s_y)| \]
Filtering

Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?
Fourier Transform pairs

Spatial domain

\[ f(x) \]

box(x)

Frequency domain

\[ F(s) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi sx} \, dx \]

sinc(s)

gauss(x; \sigma)

gauss(s; 1/\sigma)

sinc(s)

box(x)
Practical matters

What is the size of the output?

MATLAB: `filter2(g, f, shape)` or `conv2(g, f, shape)`

- `shape = 'full'`: output size is sum of sizes of `f` and `g`
- `shape = 'same'`: output size is same as `f`
- `shape = 'valid'`: output size is difference of sizes of `f` and `g`

Source: S. Lazebnik
Image half-sizing

This image is too big to fit on the screen. How can we reduce it?

How to generate a half-sized version?
Image sub-sampling

Throw away every other row and column to create a 1/2 size image - called *image sub-sampling*
Image sub-sampling

1/2  1/4  (2x zoom)  1/8  (4x zoom)

Aliasing! What do we do?
Sampling an image

Examples of GOOD sampling
Undersampling

Examples of BAD sampling -> Aliasing
Gaussian (lowpass) pre-filtering

Solution: filter the image, \textit{then} subsample

- Filter size should double for each \( \frac{1}{2} \) size reduction. Why?
Subsampling with Gaussian pre-filtering

Gaussian 1/2  G 1/4  G 1/8
Compare with...

1/2

1/4  (2x zoom)

1/8  (4x zoom)

Slide by Steve Seitz
Gaussian (lowpass) pre-filtering

Solution: filter the image, \textit{then} subsample

- Filter size should double for each \( \frac{1}{2} \) size reduction. Why?
- How can we speed this up?
Image Pyramids

Idea: Represent N x N image as a “pyramid” of 1 x 1, 2 x 2, 4 x 4, ..., 2^k x 2^k images (assuming N = 2^k)

Known as a **Gaussian Pyramid** [Burt and Adelson, 1983]

- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*
A bar in the big images is a hair on the zebra’s nose; in smaller images, a stripe; in the smallest, the animal’s nose.
Gaussian pyramid construction

Repeat
• Filter
• Subsample
Until minimum resolution reached
• can specify desired number of levels (e.g., 3-level pyramid)

The whole pyramid is only 4/3 the size of the original image!
What are they good for?

Improve Search

• Search over translations
  – Classic coarse-to-fine strategy
• Search over scale
  – Template matching
  – E.g. find a face at different scales
Sharpening

What does blurring take away?

original \hspace{1cm} \text{smoothed (5x5)} \hspace{1cm} \text{detail}

Let’s add it back:

original \hspace{1cm} \text{detail} \hspace{1cm} \text{sharpened}
Unsharp mask filter

\[ f + \alpha(f - f * g) = (1 + \alpha)f - \alpha f * g = f * ((1 + \alpha)e - \alpha g) \]