Image Processing

Computational Photography
CSE 273
Lecture 3
Announcements

• Assignment 1 is due Jan 12, 11:59 PM
• Assignment 2 will be released Jan 12
  – Due Jan 19, 11:59 PM
Image sensing pipeline

Scene Radiance → Optics → Aperture → Shutter → Camera Body

Sensor (CCD/CMOS) → Gain (ISO) → ADC → RAW

Sensor chip

Denoise and sharpen → Demosaic → White Balance → Gamma/curve → Compress → JPEG

Image Signal Processor (ISP)
Image processing

• A discipline in which both the input and output of a process are images
  – There are usually other input parameters to the process
Demosaicing

Color filter array (CFA)

Image sensor

Bayer pattern

Interpolated (lower case) pixel values

CFA

\[
\begin{array}{cccc}
G & R & G & R \\
B & G & B & G \\
G & R & G & R \\
B & G & B & G \\
\end{array}
\]

\[
\begin{array}{cccc}
rGb & Rgb & rGb & Rgb \\
rGb & Rgb & rGb & Rgb \\
rGb & Rgb & rGb & Rgb \\
rGb & Rgb & rGb & Rgb \\
\end{array}
\]

ORIGINAL IMAGE

CCD ARRAY WITH BAYER PATTERN SHOWING LOCATION OF WHITE/BLACK TRANSITION

ALIASED IMAGE
Image processing

- Color spaces
- Gamut mapping
- White balancing and color balancing
Image processing

- Dehazing
- Denoising
- Deconvolution

Motion blur and additive noise

Degraded image  Inverse filtering  Wiener filtering  Constrained least squares filtering

Less noise

Much less noise
Spatial filtering: correlation and convolution (1D)

**Figure 3.35**
Illustration of 1-D correlation and convolution of a kernel, \( w \), with a function \( f \) consisting of a discrete unit impulse. Note that correlation and convolution are functions of the variable \( x \), which acts to displace one function with respect to the other. For the extended correlation and convolution results, the starting configuration places the rightmost element of the kernel to be coincident with the origin of \( f \). Additional padding must be used.
Spatial filtering (2D)

2D correlation

\[
w(x, y) \star f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)
\]

2D convolution

\[
w(x, y) \bullet f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x - s, y - t)
\]
Correlation and convolution (2D)

FIGURE 3.36
Correlation (middle row) and convolution (last row) of a 2-D kernel with an image consisting of a discrete unit impulse. The 0's are shown in gray to simplify visual analysis. Note that correlation and convolution are functions of $x$ and $y$. As these variable change, they displace one function with respect to the other. See the discussion of Eqs. (3-45) and (3-46) regarding full correlation and convolution.
Correlation and convolution

- Convolution is commutative and associative, correlation is not.

**TABLE 3.5**
Some fundamental properties of convolution and correlation. A dash means that the property does not hold.

<table>
<thead>
<tr>
<th>Property</th>
<th>Convolution</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commutative</td>
<td>$f \ast g = g \ast f$</td>
<td>$-$</td>
</tr>
<tr>
<td>Associative</td>
<td>$f \ast (g \ast h) = (f \ast g) \ast h$</td>
<td>$-$</td>
</tr>
<tr>
<td>Distributive</td>
<td>$f \ast (g + h) = (f \ast g) + (f \ast h)$</td>
<td>$f \ast (g + h) = (f \ast g) + (f \ast h)$</td>
</tr>
</tbody>
</table>
Smoothing kernels

Average
(box kernel)

Average
(weighted average)
(Gaussian kernel)
Smoothing with box kernel

Input image

3x3

11x11

21x21
Smoothing with Gaussian kernel

**Figure 3.41**
(a) Sampling a Gaussian function to obtain a discrete Gaussian kernel. The values shown are for $K = 1$ and $\sigma = 1$. (b) Resulting $3 \times 3$ kernel [this is the same as Fig. 3.37(b)].

<table>
<thead>
<tr>
<th>Standard deviation $\sigma$</th>
<th>Percent of total volume under surface</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>39.35</td>
</tr>
<tr>
<td>2</td>
<td>86.47</td>
</tr>
<tr>
<td>3</td>
<td>98.89</td>
</tr>
</tbody>
</table>

Volume under surface greater than $3\sigma$ is negligible
Smoothing with Gaussian kernel

\[ \sigma = 7 \]
43x43

\[ \sigma = 7 \]
85x85

Difference
Smoothing with Gaussian kernel

Input image

$\sigma = 3.5$

$21 \times 21$

$\sigma = 7$

$43 \times 43$
### Border padding

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>9</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>14</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>16</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>19</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Zero padding when v = 0**

### Constant padding

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>20</td>
</tr>
</tbody>
</table>

### Replicate padding

<table>
<thead>
<tr>
<th>13</th>
<th>12</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>14</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>18</td>
<td>17</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>19</td>
<td>18</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>14</td>
<td>13</td>
</tr>
</tbody>
</table>

### Mirror padding
Border padding

- Zero padding
- Mirror padding
- Replicate padding
Derivatives

FIGURE 3.50
(a) A section of a horizontal scan line from an image, showing ramp and step edges, as well as constant segments.
(b) Values of the scan line and its derivatives.
(c) Plot of the derivatives, showing a zero crossing. In (a) and (c) points were joined by dashed lines as a visual aid.
Sharpening filters

**Figure 3.52**
(a) Blurred image of the North Pole of the moon.
(b) Laplacian image obtained using the kernel in Fig. 3.51(a).
(c) Image sharpened using Eq. (3-63) with $c = -1$.
(d) Image sharpened using the same procedure, but with the kernel in Fig. 3.51(b).
(Original image courtesy of NASA.)

**Figure 3.53**
The Laplacian image from Fig. 3.52(b), scaled to the full [0, 255] range of intensity values. Black pixels correspond to the most negative value in the unscaled Laplacian image, grays are intermediate values, and white pixels correspond to the highest positive value.
Gradient (first derivatives)

![Gradient Diagram](image)

**Figure 3.56**
(a) A $3 \times 3$ region of an image, where the $z$s are intensity values. (b)–(c) Roberts cross-gradient operators. (d)–(e) Sobel operators. All the kernel coefficients sum to zero, as expected of a derivative operator.
Dehazing

• “Single Image Haze Removal Using Dark Channel Prior”, Kaiming He, Jian Sun, and Xiaoou Tang
  – Best paper, CVPR 2009
Dehazing

- “Single Image Dehazing via Multi-Scale Convolutional Neural Networks”, Wenqi Ren, Si Liu, Hua Zhang, Jinshan Pan, Xiaochun Cao, and Ming-Hsuan Yang

ECCV 2016
Model of image degradation

• Spatial domain

\[ g(x, y) = h(x, y) \ast f(x, y) + \eta(x, y) \]

Degraded image  Degradation function  Original image  Noise image
Model of image degradation, then restoration

\[ g(x, y) = h(x, y) \star f(x, y) + \eta(x, y) \]
Image restoration

1. Remove noise
2. Estimate original image
   - Deconvolution
Noise modeled as different probability density functions

\[ p(z) = \frac{1}{\sqrt{2\pi}\sigma} \]

Gaussian

\[ p(z) = 0.607 \frac{1}{\sqrt{2b}} \]

Rayleigh

\[ K = \frac{a(b-1)^b}{(b-1)!} e^{-(b-1)} \]

Erlang (Gamma)

\[ p(z) = \frac{1}{b-a} \]

Exponential

\[ p(z) = 1 - (P_s + P_p) \]

Uniform

\[ p(z) = P_s \]

Salt-and-pepper

CSE 273, Winter 2022
Adding noise from different models

Free of noise

Gaussian

Rayleigh

Gamma
Adding noise from different models

Free of noise

Exponential

Uniform

Salt and pepper
Histograms of sample patches

Sample “flat” patches from images with noise

Identify closest probability density function (pdf) match:

- Gaussian
- Rayleigh
- Uniform
Mean filters

X-ray image

Additive Gaussian noise

Arithmetic mean filtered

Geometric mean filtered
Mean filters

Additive pepper noise

Additive salt noise

Contraharmonic mean filtered

Contraharmonic mean filtered
Order-statistic filters

Additive salt and pepper noise

1x median filtered

2x median filtered

3x median filtered
Order-statistic filters

Max filtered

Min filtered
Comparing filters

- Additive uniform + salt and pepper noise
- Arithmetic mean filtered
- Median filtered
- Geometric mean filtered
- Alpha-trimmed mean filtered
Adaptive filters

Additive Gaussian noise

Geometric mean filtered

Arithmetic mean filtered

Adaptive noise reduction filtered
Adaptive filters

Additive salt and pepper noise

Median filtered

Adaptive median filtered
Bilateral filter

• An edge-preserving low pass filter

\[ g(i, j) = \frac{\sum_{k,l} f(k, l) w(i, j, k, l)}{\sum_{k,l} w(i, j, k, l)} \]

– Bilateral weight function

\[ w(i, j, k, l) = \exp \left( -\frac{(i - k)^2 + (j - l)^2}{2\sigma_d^2} - \frac{\|f(i, j) - f(k, l)\|^2}{2\sigma_r^2} \right) \]

Domain kernel  Range kernel
Bilateral filter

Figure 3.20 Bilateral filtering (Durand and Dorsey 2002) © 2002 ACM: (a) noisy step edge input; (b) domain filter (Gaussian); (c) range filter (similarity to center pixel value); (d) bilateral filter; (e) filtered step edge output; (f) 3D distance between pixels.
Model of image degradation

• Spatial domain

\[ g(x, y) = h(x, y) \star f(x, y) + \eta(x, y) \]

- Degraded image
- Degradation function
- Original image
- Noise image

• Frequency domain

\[ G(u, v) = H(u, v)F(u, v) + N(u, v) \]

- Degraded image
- Degradation function
- Original image
- Noise image
Image processing in the frequency domain

Image in spatial domain $f(x, y)$

Fourier transform

Image in frequency domain $F(u, v)$

Frequency domain processing

Image in frequency domain $G(u, v)$

Inverse Fourier transform

Image in spatial domain $g(x, y)$

Jean-Baptiste Joseph Fourier 1768-1830

CSE 273, Winter 2022
Continuous Fourier transform

Example: box function

1D

2D
Unit discrete impulse

1D

2D
Impulse train

1D

\[ s_{\Delta x}(x) \]

\[ \cdots -3\Delta X - 2\Delta X - \Delta X 0 \Delta X 2\Delta X 3\Delta X \cdots \]

2D

\[ s_{\Delta t\Delta z}(t, z) \]

\[ \cdots \Delta T \Delta Z \cdots \]
Fourier transform of sampled function and extracting one period

**Over-sampled**

1D

- Over-sampled function
- Frequency spectrum
- Ideal lowpass filter

2D

- Recovered function
- Frequency spectrum
- Footprint of a 2-D ideal lowpass (box) filter

**Under-sampled**

1D

- Under-sampled function
- Frequency spectrum
- Ideal lowpass filter

2D

- Imperfect recovery
- Frequency spectrum
- Due to interference

CSE 273, Winter 2022
Aliasing

1D

2D

Original

Aliasing
Aliasing in real images

Original  Aliasing  No aliasing

Figure 4.19 Illustration of aliasing on resampled natural images. (a) A digital image of size 772 × 548 pixels with visually negligible aliasing. (b) Result of resizing the image to 33% of its original size by pixel deletion and then restoring it to its original size by pixel replication. Aliasing is clearly visible. (c) Result of blurring the image in (a) with an averaging filter prior to resizing. The image is slightly more blurred than (b), but aliasing is no longer objectionable. (Original image courtesy of the Signal Compression Laboratory, University of California, Santa Barbara.)
2D discrete Fourier transform (DFT)

- **(Forward) Fourier transform**

\[ F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)} \]

\( u = 0, 1, 2, \ldots, M-1 \) and \( v = 0, 1, 2, \ldots, N-1 \)

- **Inverse Fourier transform**

\[ f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi \left( \frac{ux}{M} + \frac{vy}{N} \right)} \]

\( x = 0, 1, 2, \ldots, M-1 \) and \( y = 0, 1, 2, \ldots, N-1 \)
Centering the DFT

Figure 4.22
Centering the Fourier transform. (a) A 1-D DFT showing an infinite number of periods. (b) Shifted DFT obtained by multiplying \( f(x) \) by \((-1)^x\) before computing \( F(u) \). (c) A 2-D DFT showing an infinite number of periods. The area within the dashed rectangle is the data array, \( F(u,v) \), obtained with Eq. (4-67) with an image \( f(x,y) \) as the input. This array consists of four quarter periods. (d) Shifted array obtained by multiplying \( f(x,y) \) by \((-1)^{x+y}\) before computing \( F(u,v) \). The data now contains one complete, centered period, as in (b).
Centering the DFT

Original

Shifted DFT

DFT (look at corners)

Log of shifted DFT
DFT magnitude of geometrically transformed images

Translated

Rotated about center

Same magnitude as original (invariant to translation)
DFT phase of geometrically transformed images

Original  Translated  Rotated about center
Contributions of magnitude and phase to image formation

- **Phase**
  - IDFT: Phase only (zero magnitude)
  - IDFT: Magnitude only (zero phase)
  - IDFT: Boy magnitude and rectangle phase
  - IDFT: Rectangle magnitude and boy phase
2D convolution theorem

• 2D discrete (circular) convolution

\[ f(x, y) \star h(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x-m, y-n) \]

\[ x = 0, 1, 2, \ldots, M-1 \text{ and } y = 0, 1, 2, \ldots, N-1 \]

• 2D convolution theorem

\[ f(x, y) \star h(x, y) \iff F(u, v)H(u, v) \]

\[ f(x, y) h(x, y) \iff F(u, v) \star H(u, v) \]
Filtering using convolution theorem

Filtering in the spatial domain using convolution

Expected result

Filtering in the frequency domain using product without zero-padding

Wraparound error
Filtering using convolution theorem

Filtering in frequency domain using product with zero-padding

no wraparound error

Fourier transform

Product

Gaussian lowpass filter in frequency domain

Inverse Fourier transform

Zero padding
Filtering using convolution theorem

Filtering in spatial domain using convolution

Filtering in frequency domain using product

\[ f(x, y) \star h(x, y) = \mathcal{F}^{-1} \{ F(u, v)H(u, v) \} \]

Identical results
Filtering in the frequency domain

• Ideal lowpass filter (LPF)
  – Frequency domain
Filtering in the frequency domain

- Ideal lowpass filter (LPF)
  - Spatial domain
Filtering in the frequency domain

• Gaussian lowpass filter (LPF)
Filtering in the frequency domain

- Butterworth lowpass filter (LPF)
Filtering in the frequency domain

Ideal LPF

Gaussian LPF

Butterworth LPF
Highpass filter (HPF)  
Frequency domain

- Ideal HPF
- Gaussian HPF
- Butterworth HPF
Highpass filter (HPF)
Spatial domain

Ideal HPF  Gaussian HPF  Butterworth HPF
Filtering in the frequency domain

- Ideal HPF
- Gaussian HPF
- Butterworth HPF
Filtering in the frequency domain

1D

Lowpass filter

Sharpening filter

Frequency domain

Spatial domain
Filtering in the frequency domain

Lowpass filter  Highpass filter  Offset highpass filter

2D
Bandreject filters

Ideal

Gaussian

Butterworth
Filtering in the frequency domain

• Sharpening filter
Image processing in the frequency domain

Image in spatial domain \( f(x,y) \)

\[ \text{Fourier transform} \]

Image in frequency domain \( F(u,v) \)

Frequency domain processing

\[ \text{Inverse Fourier transform} \]

Image in spatial domain \( g(x,y) \)

\[ \text{Frequency domain} \]

Jean-Baptiste Joseph Fourier
1768-1830
Periodic noise

Additive sinusoidal noise

Conjugate impulses

DFT magnitude
Notch reject filters
Notch reject filter

- Degraded image
- Filter in frequency domain
- Conjugate impulses
- DFT magnitude
- Estimate of original image
Notch reject filter

Degraded image

Filter in frequency domain

DFT magnitude

Estimate of original image

Filter in frequency domain

Estimate of original image
Estimating the degradation function

• Methods
  – Observation
  – Experimentation
  – Mathematical modeling
Estimation of degradation function by experimentation

Impulse of light

Imaged (degraded) impulse
Estimation of degradation function by mathematical modeling
Estimation of degradation function by mathematical modeling

Motion blur model
Image restoration

• Inverse filtering

\[ G(u, v) = H(u, v) \hat{F}(u, v) \]
\[ \hat{F}(u, v) = \frac{G(u, v)}{H(u, v)} \]

– Mitigate divide by zero
  • Threshold \( H(u, v) \)
  • Ideal lowpass filter \( H(u, v) \)
Image restoration, inverse filtering

Full

Limited to radius of 40

Limited to radius of 70

Limited to radius of 85
Image restoration, Wiener filtering

Inverse filtering

Wiener filtering

Full

Radially limited
Image restoration, constrained least squares filtering

- Degraded image
- Inverse filtering
- Wiener filtering
- Constrained least squares filtering

Motion blur and additive noise

Less noise

Much less noise