Lecture 4: Pixels and Filters

Professor Fei-Fei Li
Stanford Vision Lab
What we will learn today?

- Images as functions
- Linear systems (filters)
- Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7
Images as functions

• An image contains discrete number of pixels
  – A simple example
  – Pixel value:
    • “grayscale” (or “intensity”): [0,255]
Images as functions

• An image contains discrete number of pixels
  – A simple example
  – Pixel value:
    • “grayscale” (or “intensity”): [0,255]
    • “color”
      – RGB: [R, G, B]
      – Lab: [L, a, b]
      – HSV: [H, S, V]
Images as functions

• An Image as a function $f$ from $\mathbb{R}^2$ to $\mathbb{R}^M$:
  • $f(x, y)$ gives the **intensity** at position $(x, y)$
  • Defined over a rectangle, with a finite range:
    
    $$f: [a,b] \times [c,d] \to [0,255]$$

\[ \text{Domain support} \quad \text{range} \]
Images as functions

- **An Image** as a function $f$ from $\mathbb{R}^2$ to $\mathbb{R}^M$:
  - $f(x, y)$ gives the **intensity** at position $(x, y)$
  - Defined over a rectangle, with a finite range:
    $$f: [a,b] \times [c,d] \rightarrow [0,255]$$

- A color image: $f(x, y) = \begin{bmatrix} r(x, y) \\ g(x, y) \\ b(x, y) \end{bmatrix}$
Images as discrete functions

• Images are usually **digital** (discrete):
  – **Sample** the 2D space on a regular grid

• Represented as a matrix of integer values

![Image table]

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>62</td>
<td>79</td>
<td>23</td>
<td>119</td>
<td>120</td>
<td>05</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>9</td>
<td>62</td>
<td>12</td>
<td>78</td>
<td>34</td>
</tr>
<tr>
<td>10</td>
<td>58</td>
<td>197</td>
<td>46</td>
<td>46</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>176</td>
<td>135</td>
<td>5</td>
<td>188</td>
<td>191</td>
<td>68</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>29</td>
<td>26</td>
<td>37</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>89</td>
<td>144</td>
<td>147</td>
<td>187</td>
<td>102</td>
<td>62</td>
</tr>
<tr>
<td>255</td>
<td>252</td>
<td>0</td>
<td>166</td>
<td>123</td>
<td>62</td>
<td>0</td>
</tr>
<tr>
<td>166</td>
<td>63</td>
<td>127</td>
<td>17</td>
<td>1</td>
<td>0</td>
<td>99</td>
</tr>
</tbody>
</table>

**pixel**
Images as discrete functions

Cartesian coordinates

\[ f[n, m] = \begin{bmatrix}
\cdots & \cdot & \cdot \\
\cdot & f[-1, 1] & f[0, 1] & f[1, 1] \\
\cdot & f[-1, 0] & f[0, 0] & f[1, 0] & \cdots \\
f[-1, -1] & f[0, -1] & f[1, -1] & \cdots & \cdots
\end{bmatrix} \]

Notation for discrete functions
What we will learn today?

• Images as functions
• Linear systems (filters)
• Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7
Systems and Filters

• Filtering:
  – Form a new image whose pixels are a combination of original pixel values

Goals:
- Extract useful information from the images
  • Features (edges, corners, blobs...)

- Modify or enhance image properties:
  • super-resolution; in-painting; de-noising
De-noising

Salt and pepper noise

Super-resolution

In-painting

Bertamio et al.
2D discrete-space systems (filters)

\[ f[n, m] \rightarrow \text{System } S \rightarrow g[n, m] \]

\[ g = S[f], \quad g[n, m] = S\{f[n, m]\} \]

\[ f[n, m] \xrightarrow{S} g[n, m] \]
Filter example #1: Moving Average

- 2D DS moving average over a $3 \times 3$ window of neighborhood

\[
g[n, m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k, l]
\]

\[
= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]
\]

\[
(f \ast h)[m, n] = \frac{1}{9} \sum_{k,l} f[k, l] h[m-k, n-l]
\]
Filter example #1: Moving Average

\[
(f * h)[m,n] = \sum_{k,l} f[k,l] h[m-k,n-l]
\]
Filter example #1: Moving Average

\[ F[x, y] \]

\[ G[x, y] \]

\[(f \ast h)[m,n] = \sum_{k,l} f[k,l] h[m-k,n-l] \]
Filter example #1: Moving Average

\[ F[x, y] \]

\[
(f \ast h)[m,n] = \sum_{k,l} f[k,l] h[m-k, n-l]
\]
Filter example #1: Moving Average

\[ F[x, y] \]

\[ G[x, y] \]

\[(f * h)[m,n] = \sum_{k,l} f[k,l] h[m-k,n-l] \]
Filter example #1: Moving Average

\[ F[x, y] \]

\[ (f \ast h)[m,n] = \sum_{k,l} f[k,l] h[m - k, n - l] \]
Filter example #1: Moving Average

\[
F[x, y] = \sum_{k,l} f[k,l] h[m-k,n-l]
\]

\[
G[x, y]
\]

Source: S. Seitz
Filter example #1: Moving Average

In summary:

• Replaces each pixel with an average of its neighborhood.

• Achieve smoothing effect (remove sharp features)

\[
h[\cdot, \cdot] = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}
\]
Filter example #1: Moving Average
Filter example #2: Image Segmentation

- Image segmentation based on a simple threshold:

\[
g[n, m] = \begin{cases} 
255, & f[n, m] > 100 \\
0, & \text{otherwise}.
\end{cases}
\]
Classification of systems

• Amplitude properties
  • Linearity
  • Stability
  • Invertibility

• Spatial properties
  • Causality
  • Separability
  • Memory
  • Shift invariance
  • Rotation invariance
Shift-invariance

If \( f[n, m] \xrightarrow{S} g[n, m] \) then

\[
f[n - n_0, m - m_0] \xrightarrow{S} g[n - n_0, m - m_0]
\]

for every input image \( f[n,m] \) and shifts \( n_0,m_0 \)
Is the moving average system is shift invariant?

\[ F[x, y] \]

\[ G[x, y] \]
Is the moving average system is shift invariant?

\[ f[n, m] \xrightarrow{S} g[n, m] = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n - k, m - l] \]

\[ f[n - n_0, m - m_0] \]

\[ \xrightarrow{S} g[n, m] = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n - k, m - l] \]

\[ = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[(n - n_0) - k, (m - m_0) - l] \]

\[ = g[n - n_0, m - m_0] \quad \text{Yes!} \]
Linear Systems (filters)

\[ f(x, y) \rightarrow S \rightarrow g(x, y) \]

• Linear filtering:
  – Form a new image whose pixels are a weighted sum of original pixel values
  – Use the same set of weights at each point

• \( S \) is a linear system (function) iff it \( S \) satisfies

\[
S[\alpha f_1 + \beta f_2] = \alpha S[f_1] + \beta S[f_2]
\]

superposition property
Linear Systems (filters)

\[ f(x, y) \rightarrow S \rightarrow g(x, y) \]

- Is the moving average a linear system?

- Is thresholding a linear system?
  - \( f_1[n,m] + f_2[n,m] > T \)
  - \( f_1[n,m] < T \)
  - \( f_2[n,m]<T \quad \text{No!} \)
LSI (linear shift invariant) systems

Impulse response

\[ \delta[n, m] \rightarrow S \rightarrow h[n, m] \]

\[ \delta[n - k, m - l] \rightarrow S(SI) \rightarrow h[n - k, m - l] \]
LSI (linear shift invariant) systems

**Example:** impulse response of the 3 by 3 moving average filter:

\[ h[n, m] = \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n - k, m - l] \]

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

\[
\frac{1}{9} \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1
\end{bmatrix}
\]
LSI (linear *shift invariant*) systems

An LSI system is completely specified by its impulse response.

\[
f[n, m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k, l] \delta_2[n - k, m - l]
\]

\[
\rightarrow \boxed{S \text{ LSI}} \rightarrow \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k, l] h[n - k, m - l]
\]

\[
\delta_2[n, m] \rightarrow \boxed{S} \rightarrow h[n, m]
\]

Discrete convolution

\[
= f[n, m] \ast \ast h[n, m]
\]
What we will learn today?

- Images as functions
- Linear systems (filters)
- Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7
Discrete convolution (symbol: $\ast$)

- Fold $h[n,m]$ about origin to form $h[-k,-l]$
- Shift the folded results by $n,m$ to form $h[n-k,m-l]$
- Multiply $h[n-k,m-l]$ by $f[k,l]$
- Sum over all $k,l$
- Repeat for every $n,m$
Discrete convolution (symbol: \(**)

- Fold \(h[n,m]\) about origin to form \(h[-k,-l]\)
- Shift the folded results by \(n,m\) to form \(h[n - k, m - l]\)
- Multiply \(h[n - k, m - l]\) by \(f[k, l]\)
- Sum over all \(k, l\)
- Repeat for every \(n,m\)
Discrete convolution (symbol: $\ast$)

- Fold $h[n,m]$ about origin to form $h[-k,-l]$
- Shift the folded results by $n,m$ to form $h[n - k,m - l]$
- Multiply $h[n - k,m - l]$ by $f[k,l]$
- Sum over all $k,l$
- Repeat for every $n,m$
Convolution in 2D - examples

Original

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

= ?

Courtesy of D Lowe
Convolution in 2D - examples

Original

Filtered (no change)

 Courtesy of D Lowe
Convolution in 2D - examples

Original

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

\[=\]

?
Convolution in 2D - examples

Original

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

Shifted right
By 1 pixel

Courtesy of D Lowe
Convolution in 2D - examples
Convolution in 2D - examples

Original

Blur (with a box filter)

Courtesy of D Lowe
Convolution in 2D - examples

Original

(Note that filter sums to 1)

“details of the image”

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0 \\
\end{array}
\] - \frac{1}{9}

\begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
= ?

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

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0 \\
\end{array}
\] - \frac{1}{9}

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

Courtesy of D Lowe
• What does blurring take away?

original

–

smoothed (5x5)

= 

detail

• Let’s add it back:

original

+ a

detail

= 

sharpened
Convolution in 2D – Sharpening filter

Sharpening filter: Accentuates differences with local average
Image support and edge effect

• A computer will only convolve finite support signals.
  • That is: images that are zero for n,m outside some rectangular region
• MATLAB’s conv2 performs 2D DS convolution of finite-support signals.

\[
\begin{align*}
\text{N1} \times \text{M1} & \ast \text{N2} \times \text{M2} = (\text{N1} + \text{N2} - 1) \times (\text{M1} + \text{M2} - 1)
\end{align*}
\]
A computer will only convolve finite support signals.

What happens at the edge?

- zero “padding”
- edge value replication
- mirror extension
- more (beyond the scope of this class)

-> Matlab conv2 uses zero-padding
What we will learn today?

- Images as functions
- Linear systems (filters)
- Convolution and correlation

Some background reading:
Forsyth and Ponce, Computer Vision, Chapter 7
(Cross) correlation (symbol: $\ast\ast$)

Cross correlation of two 2D signals $f[n,m]$ and $g[n,m]$

$$r_{fg}[k, l] \triangleq \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f[n, m] g^*[n - k, m - l]$$

$$= \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f[n + k, m + l] g^*[n, m], \quad k, l \in \mathbb{Z}.$$  

$(k, l)$ is called the lag

• Equivalent to a convolution without the flip

$$r_{fg}[n, m] = f[n, m] \ast\ast g^*[-n, -m]$$

$(g^*$ is defined as the complex conjugate of $g$. In this class, $g(n,m)$ are real numbers, hence $g^*=g.$)
(Cross) correlation – example

MATLAB’s xcorr2

g = f + noise

r > 0.5

Courtesy of J. Fessler
(Cross) correlation – example

\[ dc(y_1, y_2) = \frac{y_1^T y_2}{|y_1||y_2|} \]
Convolution vs. (Cross) Correlation

Convolution

Cross-correlation

Autocorrelation
**Convolution vs. (Cross) Correlation**

- **Convolution** is an integral that expresses the amount of overlap of one function as it is shifted over another function.
  - convolution is a filtering operation

- **Correlation** compares the *similarity of two sets of data*. Correlation computes a measure of similarity of two input signals as they are shifted by one another. The correlation result reaches a maximum at the time when the two signals match best.
  - correlation is a measure of relatedness of two signals
Cross Correlation Application:
Vision system for TV remote control
- uses template matching

properties

• Commutative property:

\[ f \star \star h = h \star \star f \]

• Associative property:

\[ (f \star \star h_1) \star \star h_2 = f \star \star (h_1 \star \star h_2) \]

• Distributive property:

\[ f \star \star (h_1 + h_2) = (f \star \star h_1) + (f \star \star h_2) \]

The order doesn’t matter! \( h_1 \star \star h_2 = h_2 \star \star h_1 \)
properties

• Shift property:

\[ f[n, m] \ast \ast \delta_2[n - n_0, m - m_0] = f[n - n_0, m - m_0] \]

• Shift-invariance:

\[ g[n, m] = f[n, m] \ast \ast h[n, m] \]

\[ \implies f[n - l_1, m - l_1] \ast \ast h[n - l_2, m - l_2] \]

\[ = g[n - l_1 - l_2, m - l_1 - l_2] \]
What we have learned today?

• Images as functions
• Linear systems (filters)
• Convolution and correlation