CPSC 425: Computer Vision

Lecture 4: Image Filtering

(unless otherwise stated slides are taken or adopted from Bob Woodham, Jim Little, Leon Sigal and Fred Tung)
Menu for Today (September 17, 2021)

Topics:
— Linear filters
— Linear filter properties
— Correlation / Convolution
— Filter examples: Box, Gaussian, …

Readings:
— Today’s Lecture: Forsyth & Ponce (2nd ed.) 4.1, 4.5; Szel. 3.1.1-3.2.2
— Next Lecture: none

Reminders:
— Assignment 0 – not due…but complete it ASAP so you can do
Assignment 1: Image Filtering and Hybrid Images - out now
Today’s “fun” Example:

Developed by the French company **Varioptic**, the lenses consist of an oil-based and a water-based fluid sandwiched between glass discs. The electric charge causes the boundary between oil and water to change shape, altering the lens geometry and therefore the lens focal length.

The intended applications are: **auto-focus** and **image stabilization**. No moving parts. Fast response. Minimal power consumption.

**Video Source:** [https://www.youtube.com/watch?v=2c6ICdDFOY8](https://www.youtube.com/watch?v=2c6ICdDFOY8)
Today’s “fun” Example:

**Electrostatic** field between the column of water and the electron (other side of power supply attached to the pipe) — see full video for complete explanation

Video Source: [https://www.youtube.com/watch?v=NjLJ77IuBdM](https://www.youtube.com/watch?v=NjLJ77IuBdM)
Today’s **“fun”** Example:

As one example, in 2010, **Cognex** signed a licence agreement with Varioptic to add auto-focus capability to its DataMan line of industrial ID readers (press release May 29, 2012)

*Video Source:* [https://www.youtube.com/watch?v=EU8LXxip1NM](https://www.youtube.com/watch?v=EU8LXxip1NM)
Today’s “fun” Example:

https://youtu.be/R2mC-NUAmMk
We take a “physics-based” approach to image formation
— Treat the camera as an instrument that takes measurements of the 3D world

Basic abstraction is the pinhole camera

Lenses overcome limitations of the pinhole model while trying to preserve it as a useful abstraction. When maximum accuracy required, it is necessary to model additional details of each particular camera (and camera setting)
— Aside: This is called camera calibration
Lecture 3: Re-cap Snell’s Law

\[ n_1 \sin \alpha_1 = n_2 \sin \alpha_2 \]
Lecture 3: Re-cap Lenses

Thin lens equation

\[
\frac{1}{z'} - \frac{1}{z} = \frac{1}{f}
\]

characterizes the relationship between \( f \), \( z \) and \( z' \)

Some “aberrations and distortions” persist. For example:

— the index of refraction depends on wavelength, \( \lambda \), of light

— vignetting reduces image brightness (gradually) away from the image center

The human eye functions much like a camera
Lecture 4: Image as a 2D Function

A (greyscale) image is a 2D function

What is the range of the image function?

$I(X, Y) \in [0, 255] \in \mathbb{Z}$

**domain**: $(X, Y) \in ([1, \text{width}], [1, \text{height}])$

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
Adding two Images

Since images are functions, we can perform operations on them, e.g., average

\[
\frac{I(X, Y)}{2} + \frac{G(X, Y)}{2}
\]
Adding two Images

\[ a = \frac{I(X,Y)}{2} + \frac{G(X,Y)}{2} \]

\[ b = \frac{I(X,Y) + G(X,Y)}{2} \]

Question:

- \( a = b \)
- \( a > b \)
- \( a < b \)
Adding two Images

Red pixel in camera man image = 98
Red pixel in moon image = 200

\[
\frac{98}{2} + \frac{200}{2} = 49 + 100 = 149
\]

\[
\frac{98 + 200}{2} = \frac{298}{2} = \frac{255}{2} = 127
\]

Question:

\[a = b\]

\[a > b\]

\[a < b\]
Adding two Images

It is often convenient to convert images to doubles when doing processing

In Python

```python
from PIL import Image
img = Image.open('cameraman.png')
import numpy as np
imgArr = np.asfarray(img)

# Or do this
import matplotlib.pyplot as plt
camera = plt.imread('cameraman.png');```
What types of **transformations** can we do?

- **Filtering**
  - $I(X,Y) \rightarrow I'(X,Y)$
  - changes range of image function

- **Warping**
  - $I(X,Y) \rightarrow I'(X,Y)$
  - changes domain of image function

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
What types of **filtering** can we do?

- **Point** Operation
- **Neighborhood** Operation

Slide Credit: Ioannis (Yannis) Gkioulakis (CMU)
Examples of **Point Processing**

- **original**
- **darken**  
  \[ I(X, Y) - 128 \]
- **lower contrast**  
  \[ \frac{I(X, Y)}{2} \]
- **non-linear lower contrast**
  \[ \left( \frac{I(X, Y)}{255} \right)^{1/3} \times 255 \]

- **invert**  
  \[ 255 - I(X, Y) \]
- **lighten**  
  \[ I(X, Y) + 128 \]
- **raise contrast**  
  \[ I(X, Y) \times 2 \]
- **non-linear raise contrast**
  \[ \left( \frac{I(X, Y)}{255} \right)^{2} \times 255 \]

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
Examples of **Point Processing**

- **Original**
  - $I(X, Y)$
- **Darken**
  - $I(X, Y) - 128$
  - $255 - I(X, Y)$
- **Lower Contrast**
  - $I(X, Y) / 2$
  - $I(X, Y) \times 2$
- **Non-linear Lower Contrast**
  - $\left( \frac{I(X, Y)}{255} \right)^{1/3} \times 255$
  - $\left( \frac{I(X, Y)}{255} \right)^2 \times 255$

*Slide Credit: Ioannis (Yannis) Gkioulkas (CMU)*
Lecture 3b: Introduction to Filtering

**Point** Operation

(point processing)

**Neighborhood** Operation

“filtering”
**Linear Filters**

Let $I(X, Y)$ be an $n \times n$ digital image (for convenience we let width = height)

Let $F(X, Y)$ be another $m \times m$ digital image (our “filter” or “kernel”)

For convenience we will assume $m$ is odd (Here, $m = 5$)
### Linear Filters

Let $k = \left\lfloor \frac{m}{2} \right\rfloor$

Compute a new image, $I'(X, Y)$, as follows

$$I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j)$$

**Intuition:** each pixel in the output image is a linear combination of the central pixel and its neighboring pixels in the original image.
Linear Filters

For a given $X$ and $Y$, superimpose the filter on the image centered at $(X, Y)$
Linear Filters

For a given $X$ and $Y$, superimpose the filter on the image centered at $(X, Y)$.

Compute the new pixel value, $I'(X, Y)$, as the sum of $m \times m$ values, where each value is the product of the original pixel value in $I(X, Y)$ and the corresponding values in the filter.
Linear Filters

The computation is repeated for each $(X, Y)$
### Linear Filter Example

The formula for the output image $I'(X, Y)$ is given by:

$$I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j)$$

- **$F(X, Y)$** is the filter with values in a $3 \times 3$ matrix.
- **$I(X, Y)$** is the input image.
- **$I'(X, Y)$** is the output image.

**Slide Credit:** Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ F(X, Y) \]

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

\[ I(X, Y) \]

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

\[ I'(X, Y) \]

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

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j) \]

output

Filter $I(X, Y)$

Image (signal) $I(X+i, Y+j)$

Slide Credit: Ioannis (Yannis) Gkioulkas (CMU)
Linear Filter Example

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X + i, Y + j)
\]

Filter \( F(X, Y) \)

Output \( I'(X, Y) \)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j) \]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ F(X, Y) \]

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

\[ I(X, Y) \]

\[ I'(X, Y) \]

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j) \]

*Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)*
Linear Filter Example

\[ I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I,J) I(X+i,Y+j) \]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[
F(X, Y) = \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{bmatrix}
\]

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

\[
I(X, Y)
\]

image

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

\[
I'(X, Y)
\]

output

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

\[
\begin{array}{cccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X + i, Y + j)
\]

output

filter

image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[
F(X, Y) = \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{bmatrix}
\]

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

\[
I(X, Y) = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 0 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\
0 & 0 & 90 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

\[
I'(X, Y) = \begin{bmatrix}
0 & 10 & 20 & 30 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X + i, Y + j)
\]

Output

Filter

Image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j) \]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) \cdot I(X+i, Y+j) \]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

$$F(X, Y)$$

$${1 \over 9}$$

$$I(X, Y)$$

$$I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X + i, Y + j)$$

Output

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X + i, Y + j) \]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X + i, Y + j) \]

Slide Credit: Ioannis (Yannis) Gkioulakas (CMU)
**Linear Filter Example**

\[ F(X, Y) \]

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

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

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j) \]

**Slide Credit:** Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ F(X, Y) \]

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

\[ I(X, Y) \]

image

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

output

\[ I'(X, Y) \]

\[
\begin{array}{cccccccc}
0 & 10 & 20 & 30 & 30 & 20 & 10 \\
0 & 20 & 40 & \quad & \quad & \quad & \quad \\
\end{array}
\]

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j)
\]

output

\[
\begin{array}{cccccccc}
\text{filter} & \text{image (signal)} \\
\end{array}
\]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j) \]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X+i, Y+j) \]

Slide Credit: Ioannis (Yannis) Gkioulkekas (CMU)
### Linear Filter Example

![Linear Filter Example Diagram](image)

The output of a linear filter can be calculated as:

$$ I'(X, Y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} F(I, J) I(X+i, Y+j) $$

- **Filter** $F(X, Y)$
- **Input Image** $I(X, Y)$
- **Output Image** $I'(X, Y)$

**Slide Credit**: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ F(X, Y) \]

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

\[ I(X, Y) \]

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(I, J) I(X + i, Y + j) \]

output

filter

image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filter Example

\[ F(X, Y) \]

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

\[ I(X, Y) \]

\[ I'(X, Y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} F(I, J) I(X+i, Y+j) \]

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
For a given $X$ and $Y$, superimpose the filter on the image centered at $(X, Y)$

Compute the new pixel value, $I'(X, Y)$, as the sum of $m \times m$ values, where each value is the product of the original pixel value in $I(X, Y)$ and the corresponding values in the filter
Let’s do some accounting ...

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j)
\]

At each pixel \((X, Y)\), there are \(m \times m\) multiplications.

There are \(n \times n\) pixels in \((X, Y)\).

**Total:** \(m^2 \times n^2\) multiplications.

When \(m\) is fixed, small constant, this is \(O(n^2)\). But when \(m \approx n\) this is \(O(m^4)\).
Linear Filters: **Boundary Effects**
Linear Filters: **Boundary Effects**

Four standard ways to deal with boundaries:

1. **Ignore these locations:** Make the computation undefined for the top and bottom $k$ rows and the leftmost and rightmost $k$ columns
Linear Filters: **Boundary Effects**

Four standard ways to deal with boundaries:

1. **Ignore these locations**: Make the computation undefined for the top and bottom \( k \) rows and the leftmost and rightmost \( k \) columns

2. **Pad the image with zeros**: Return zero whenever a value of \( I \) is required at some position outside the defined limits of \( X \) and \( Y \)
Linear Filters: **Boundary** Effects
Linear Filters: **Boundary Effects**

Four standard ways to deal with boundaries:

1. **Ignore these locations**: Make the computation undefined for the top and bottom $k$ rows and the leftmost and rightmost $k$ columns

2. **Pad the image with zeros**: Return zero whenever a value of $I$ is required at some position outside the defined limits of $X$ and $Y$

3. **Assume periodicity**: The top row wraps around to the bottom row; the leftmost column wraps around to the rightmost column
Linear Filters: **Boundary Effects**
Linear Filters: **Boundary** Effects
Linear Filters: **Boundary Effects**

Four standard ways to deal with boundaries:

1. **Ignore these locations:** Make the computation undefined for the top and bottom $k$ rows and the leftmost and rightmost $k$ columns.

2. **Pad the image with zeros:** Return zero whenever a value of $I$ is required at some position outside the defined limits of $X$ and $Y$.

3. **Assume periodicity:** The top row wraps around to the bottom row; the leftmost column wraps around to the rightmost column.

4. **Reflect border:** Copy rows/columns locally by reflecting over the edge.
Linear Filters: **Boundary Effects**
Linear Filters: **Boundary Effects**

Four standard ways to deal with boundaries:

1. **Ignore these locations**: Make the computation undefined for the top and bottom $k$ rows and the leftmost and rightmost $k$ columns

2. **Pad the image with zeros**: Return zero whenever a value of $I$ is required at some position outside the defined limits of $X$ and $Y$

3. **Assume periodicity**: The top row wraps around to the bottom row; the leftmost column wraps around to the rightmost column

4. **Reflect border**: Copy rows/columns locally by reflecting over the edge
A short exercise ...
Example 1: Warm up
Example 1: Warm up

Original

Filter

Result (copy)
Example 2:

Original

Filter

Result
Example 2:

Original

Filter

Result
(shift left by 1 pixel)
Example 3:

Original

Filter
(filter sums to 1)

Result
Example 3:

Original

Filter
(filter sums to 1)

Result
(blur with a box filter)
Example 4:

Original

Filter
(filter sums to 1)

Result
Example 4:

Original

Filter
(filter sums to 1)

Result
(sharpening)
Example 4: Sharpening

Before

After
Example 4: Sharpening

Before

After

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)
Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j) \]
Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j)
\]

Definition: **Convolution**

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X - i, Y - j)
\]
Definition: **Correlation**

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j)
\]

\[
\begin{array}{ccc}
  a & b & c \\
  d & e & f \\
  g & h & i \\
\end{array}
\]

\[
\begin{array}{ccc}
  1 & 2 & 3 \\
  4 & 5 & 6 \\
  7 & 8 & 9 \\
\end{array}
\]

Filter | Image | Output
---|---|---
\[
= 1a + 2b + 3c + 4d + 5e + 6f + 7g + 8h + 9i
\]
**Linear Filters: Correlation vs. Convolution**

**Definition: Correlation**

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j)
\]

**Definition: Convolution**

\[
I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X - i, Y - j)
\]

\[
\begin{array}{ccc}
\text{Filter} & \text{Image} & \text{Output} \\
\hline
a & b & c \\
d & e & f \\
g & h & i \\
\hline
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9 \\
\end{array}
\]

\[
= 9a + 8b + 7c + 6d + 5e + 4f + 3g + 2h + 1i
\]
**Linear Filters:** Correlation vs. Convolution

**Definition: Correlation**

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j) \]

**Definition: Convolution**

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X - i, Y - j) \]

<table>
<thead>
<tr>
<th>Filter</th>
<th>Image</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Filter" /></td>
<td><img src="image" alt="Image" /></td>
<td><img src="image" alt="Output" /></td>
</tr>
</tbody>
</table>

- \[ = 9a + 8b + 7c + 6d + 5e + 4f + 3g + 2h + 1i \]
**Linear Filters: Correlation vs. Convolution**

**Definition: Correlation**

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X + i, Y + j) \]

**Definition: Convolution**

\[ I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j) I(X - i, Y - j) \]

\[ = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(-i, -j) I(X + i, Y + j) \]

**Note:** if \( F(X, Y) = F(-X, -Y) \) then correlation = convolution.
Preview: Why convolutions are important?

Who has heard of **Convolutional Neural Networks** (CNNs)?

What about **Deep Learning**?

Basic operations in CNNs are convolutions (with learned linear filters) followed by non-linear functions.

**Note:** This results in non-linear filters.
Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X,Y)$ be a digital image.

**Superposition**: Let $F_1$ and $F_2$ be digital filters.

\[
(F_1 + F_2) \otimes I(X,Y) = F_1 \otimes I(X,Y) + F_2 \otimes I(X,Y)
\]
Linear Filters: **Properties**

Let \( \otimes \) denote convolution. Let \( I(X, Y) \) be a digital image.

**Superposition:** Let \( F_1 \) and \( F_2 \) be digital filters.

\[
(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)
\]

**Scaling:** Let \( F \) be a digital filter and let \( k \) be a scalar.

\[
(kF) \otimes I(X, Y) = F \otimes (kI(X, Y)) = k(F \otimes I(X, Y))
\]
Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X, Y)$ be a digital image.

**Superposition:** Let $F_1$ and $F_2$ be digital filters

\[
(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)
\]

**Scaling:** Let $F$ be digital filter and let $k$ be a scalar

\[
(kF) \otimes I(X, Y) = F \otimes (kI(X, Y)) = k(F \otimes I(X, Y))
\]

**Shift Invariance:** Output is local (i.e., no dependence on absolute position)
Linear Filters: Shift Invariance

Output does **not** depend on absolute position
Linear Filters: Properties

Let \( \otimes \) denote convolution. Let \( I(X, Y) \) be a digital image.

**Superposition:** Let \( F_1 \) and \( F_2 \) be digital filters.

\[
(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)
\]

**Scaling:** Let \( F \) be a digital filter and let \( k \) be a scalar.

\[
(kF) \otimes I(X, Y) = F \otimes (kI(X, Y)) = k(F \otimes I(X, Y))
\]

**Shift Invariance:** Output is local (i.e., no dependence on absolute position).

An operation is linear if it satisfies both superposition and scaling.
Linear Systems: Characterization Theorem

Any linear, shift invariant operation can be expressed as convolution.
Example 5: Smoothing with a Box Filter

Filter has equal positive values that sum up to 1

Replaces each pixel with the average of itself and its local neighborhood

— Box filter is also referred to as **average filter** or **mean filter**
Example 5: Smoothing with a Box Filter

Forsyth & Ponce (2nd ed.) Figure 4.1 (left and middle)
Example 5: Smoothing with a Box Filter

What happens if we increase the width (size) of the box filter?
Example 5: Smoothing with a Box Filter

Gonzales & Woods (3rd ed.) Figure 3.3
Smoothing

Smoothing with a box *doesn’t model lens defocus* well
— Smoothing with a box filter depends on direction
— Image in which the center point is 1 and every other point is 0
the output will be a square of the same size as the box filter
Lecture 2: Re-cap

* image credit: https://catlikecoding.com/unity/tutorials/advanced-rendering/depth-of-field/circle-of-confusion/lens-camera.png
Example 6:

Smoothing with a box **doesn’t model lens defocus** well
— result of smoothing with a box filter depends on direction
— Image in which the center point is 1 and every other point is 0

Filter

<table>
<thead>
<tr>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
</tr>
<tr>
<td>1 1 1</td>
</tr>
<tr>
<td>1 1 1</td>
</tr>
<tr>
<td>1 1 1</td>
</tr>
</tbody>
</table>

Image

```
0 0 0 0 0 0
0 0 0 0 0 0
0 0 1 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```
Example 6: Smoothing with a Gaussian

Smoothing with a box **doesn’t model lens defocus** well

— Smoothing with a box filter depends on direction

— Image in which the center point is 1 and every other point is 0

<table>
<thead>
<tr>
<th>Filter</th>
<th>Image</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Filter" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Result" /></td>
</tr>
</tbody>
</table>
Smoothing

Smoothing with a box **doesn’t model lens defocus** well
— Smoothing with a box filter depends on direction
— Image in which the center point is 1 and every other point is 0

Smoothing with a (circular) **pillbox** is a better model for defocus *(in geometric optics)*

The **Gaussian** is a good general smoothing model
— for phenomena (that are the sum of other small effects)
— whenever the Central Limit Theorem applies
Summary

— The correlation of $F(X, Y)$ and $I(X, Y)$ is:

$$I'(X, Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i, j)I(X + i, Y + j)$$

— Visual interpretation: Superimpose the filter $F$ on the image $I$ at $(X, Y)$, perform an element-wise multiply, and sum up the values

— Convolution is like correlation except filter “flipped” (rotated by 180)

$$F(X, Y) = F(-X, -Y)$$

then correlation = convolution.

— Characterization Theorem: Any linear, spatially invariant operation can be expressed as a convolution
Menu for Today (September 15, 2021)

Topics: Image Filtering (also topic for next week)

— Image as a function
— Linear filters

— Correlation / Convolution
— Filter examples: Box, Gaussian

Readings:

— Today’s Lecture: Forsyth & Ponce (2nd ed.) 4.1, 4.5; Szel. 3.1.1-3.2.2
— Next Lecture: none

Reminders:

— Assignment 0 (ungraded) due Friday, September 17
— Assignment 1: Image Filtering and Hybrid Images (out September 14)