Machine Learning in Computer Vision: Image Filters and Affine Functions

Lecturer: Oren Freifeld
TA: Ron Shapira Weber

Computer Science, Ben-Gurion University

Apr 25, 2018
1 Image Filters

2 Affine Functions
Examples of using Image Filters

1. Image processing (e.g., enhancement, sharpening, smoothing, etc.)
2. Spatial derivatives (e.g., see optical flow)
3. One way to define high-order clique functions in MRFs (e.g.: Field of Experts; Steerable Random Filters)
4. Modeling image degradation (e.g., blurring or motion artifacts)
5. Convolutional Neural Networks (later in this class)
6. Object Detection via template matching (AKA match filter)
\( I \): an image; \( h \): a filter (another, usually-smaller) image

**Definition (The correlation operator)**

The **correlation operator**, \( \otimes \), creates a new image, \( I \otimes h \), via

\[
(I \otimes h)(i, j) = \sum_{k,l} I(i + k, j + l)h(k, l) = \sum_{k',l'} I(k', l')h(k' - i, l' - j)
\]

**Definition (The convolution operator)**

The **convolution operator**, \( * \), creates a new image, \( I * h \), via

\[
(I * h)(i, j) = \sum_{k,l} I(i - k, j - l)h(k, l) = \sum_{k',l'} I(k', l')h(i - k', j - l')
\]

\[
= (h * I)(i, j) = (I \otimes h_{\text{flipped}})(i, j)
\]
Since $I * h = I \otimes h_{\text{flipped}}$, both operators coincide when the filter is symmetric around each of its axes.

Example

$$h = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \Rightarrow (I * h)(i, j) = (I \otimes h)(i, j)$$ is the average value of $I$ in a $3 \times 3$ neighborhood around pixel $(i, j)$:

$$\frac{1}{9} [I(i - 1, j - 1) + I(i - 1, j) + I(i - 1, j + 1) + I(i, j - 1)$$

$$+ I(i, j) + I(i, j + 1) + I(i + 1, j - 1) + I(i + 1, j) + I(i + 1, j + 1)]$$
Another Example

The filter below is symmetric so convolution = correlation.

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

Figure taken from Szelisk’s book, 2010
Remarks

- There are standard ways to handle pixels near the image boundaries (e.g., set “outside pixels” to zero)
- Internally, efficient implementations usually use the Fast Fourier Transform (FFT) instead of the formulas we saw in the definitions.
- In continuous domains, we have similar definitions, with integrals instead of sums.
Exercise

Show that convolution is linear. In effect, show that if $I_1$ and $I_2$ are 2D digital arrays, and $c_1, c_2 \in \mathbb{R}$, then

$$(c_1I_1 + c_2I_2) * h = c_1(I_1 * h) + c_2(I_2 * h) \quad (1)$$

Solution

Let $g_1 = I_1 * h$, $g_2 = I_2 * h$, and $g_3 = (c_1I_1 + c_2I_2) * h$.

$$g_3(i, j) =$$

$$\sum_{k,l} (c_1I_1(i - k, j - l) + c_2I_2(i - k, j - l))h(k, l)$$

$$= \left(\sum_{k,l} c_1I_1(i - k, j - l)h(k, l)\right) + \left(\sum_{k,l} c_2I_2(i - k, j - l)h(k, l)\right)$$

$$= c_1 \left(\sum_{k,l} I_1(i - k, j - l)h(k, l)\right) + c_2 \left(\sum_{k,l} I_2(i - k, j - l)h(k, l)\right)$$

$$= c_1I_1(i, j) + c_2I_2(i, j)$$
Exercise

Show that convolution is linear. In effect, show that if $I_1$ and $I_2$ are 2D digital arrays, and $c_1, c_2 \in \mathbb{R}$, then

$$(c_1 I_1 + c_2 I_2) \ast h = c_1 (I_1 \ast h) + c_2 (I_2 \ast h) \tag{1}$$

Solution

Let $g_1 = I_1 \ast h$, $g_2 = I_2 \ast h$, and $g_3 = (c_1 I_1 + c_2 I_2) \ast h$.

$$g_3(i, j) =$$
$$\sum_{k,l} (c_1 I_1(i - k, j - l) + c_2 I_2(i - k, j - l))h(k, l)$$
$$= \left( \sum_{k,l} c_1 I_1(i - k, j - l)h(k, l) \right) + \left( \sum_{k,l} c_2 I_2(i - k, j - l)h(k, l) \right)$$
$$= c_1 \left( \sum_{k,l} I_1(i - k, j - l)h(k, l) \right) + c_2 \left( \sum_{k,l} I_2(i - k, j - l)h(k, l) \right)$$
$$= c_1 I_1(i, j) + c_2 I_2(i, j)$$
Different Notations for Convolution

\[ g(i, j) = \sum_{k,l} I(i-k, j-l) h(k,l) = \sum_{k,l} h(i-k, j-l) I(k, l). \quad (2) \]

The summation is done over all relevant pixels (i.e., where \( h \) is defined). A slightly different way of writing the same thing is as

\[ g(\mathbf{x}) = \sum_{\mathbf{x}_i} I(\mathbf{x} - \mathbf{x}_i) h(\mathbf{x}_i) = \sum_{\mathbf{x}_i} h(\mathbf{x} - \mathbf{x}_i) I(\mathbf{x}_i) \quad (3) \]

where \( \mathbf{x} \) denotes the location of the pixel of interest, and the \( \mathbf{x}_i \)'s denote the locations of the pixels where \( h \) is defined.

Example (When \( h \) is Gaussian)

\[ g(\mathbf{x}) = \frac{1}{c} \sum_{\mathbf{x}_i} \exp \left( -\frac{1}{2} \frac{||\mathbf{x} - \mathbf{x}_i||^2}{\sigma^2} \right) I(\mathbf{x}_i) \quad (4) \]

where \( c \) typically is taken as a normalizer: \( c = \sum_{\mathbf{x}_i} \exp \left( -\frac{1}{2} \frac{||\mathbf{x} - \mathbf{x}_i||^2}{\sigma^2} \right) \).
Impulse Response

Definition (2D discrete-domain impulse signal)

\[ \delta(i, j) = \begin{cases} 
1 & \text{if } i = 0 \text{ and } j = 0 \\
0 & \text{otherwise}
\end{cases} \] (5)

- \( h \) is also called the “impulse response”, since its convolution with an impulse signal is \( h \ast \delta = \delta \ast h = h \)
A 2D filter, $K$, is called separable if it can be written as the outer product of two 1D filters; namely:

$$K = vh^T$$  \hspace{1cm} (6)

where $v$ and $h$ are two column vectors, typically of the same length.
Convolution with a Separable Filter

For a separable filter, $K = vh^T$, the 2D convolution, $g = I * K$, can be done more efficiently via a 1D horizontal convolution followed by a 1D vertical convolution; i.e., if $g = I * K$, then it can be computed as:

- Convolve each row $i$ with $h$

$$
\tilde{g}(i,:) = I(i,:) * h
$$

$$
\tilde{g}(i,j) = \sum_l I(i,j-l)h(l) \tag{7}
$$

- Convolve each column $j$ of the result with $v$:

$$
g(:,j) = \tilde{g}(:,j) * v
$$

$$
g(i,j) = \sum_k \tilde{g}(i-k,j)v(k) \tag{8}
$$
Example (An isotropic “Gaussian”)

A $5 \times 5$ discrete and truncated approximation of an isotropic Gaussian:

$$h = v = \frac{1}{16} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

$$K = vh^T = \frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Example (horizontal Sobel filter – approximates $\frac{\partial}{\partial x}$)

$$h = \frac{1}{2} \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad v = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

$$K = vh^T = \frac{1}{8} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$
Example (An isotropic “Gaussian”)

A $5 \times 5$ discrete and truncated approximation of an isotropic Gaussian:

$$h = v = \frac{1}{16} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

$$K = vh^T = \frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Example (horizontal Sobel filter – approximates $\frac{\partial}{\partial x}$)

$$h = \frac{1}{2} \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad v = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

$$K = vh^T = \frac{1}{8} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$
Laplacian of Gaussian (LoG)

In a continuous domain:
- A normalized and isotropic Gaussian filter is given by:

\[ G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \]

- The Laplacian of \( I \) is

\[ \nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) I \]
Laplacian of Gaussian (LoG)

- Blurring $I$ with $G(x, y, \sigma)$ followed up by applying $\nabla^2$ can be done in a single operation, by convolving $I$ with the LoG filter:

$$\nabla^2 G(x, y, \sigma) = \left( \frac{x^2 + y^2}{\sigma^4} - \frac{2}{\sigma^2} \right) G(x, y, \sigma)$$

- In practice, we approximate this using a finite filter, e.g., $5 \times 5$.
- LoG is an example of an undirected filter: at a given point, it is invariant to the rotation of $I$ around that point.
Laplacian of Gaussian (LoG)

\[ \nabla^2 G(x, y, \sigma) = \left( \frac{x^2 + y^2}{\sigma^4} - \frac{2}{\sigma^2} \right) G(x, y, \sigma) \]
**Definition (directional derivative)**

The directional derivative of a function \( f(x, y) \), in the direction of the unit vector \( \mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix} \), denoted by \( \nabla_{\mathbf{u}} = \frac{\partial}{\partial \mathbf{u}} \), is the scalar given by

\[
\nabla_{\mathbf{u}} f \triangleq (\nabla_x f) \mathbf{u} = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix} \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix} = \cos \theta \frac{\partial f}{\partial x} + \sin \theta \frac{\partial f}{\partial y}
\]

where \( \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \) and \( \nabla_x f = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix} \).
Directed/Oriented Filters

- An example for an oriented filter is obtained via a Gaussian blurring followed by taking the directional derivative,

\[
\nabla (G \ast I) \ast u = \nabla u (G \ast I) = (\nabla u G) \ast I
\]

(9)

where \( u = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix} \) and

\[
\nabla u G = u \frac{\partial G}{\partial x} + v \frac{\partial G}{\partial y}
\]

(10)

- The Sobel filter, \( \frac{1}{8} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \), approximates this for \( u = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^T \).

- Remark: we briefly discussed the Steerable Random Field model: an MRF whose clique functions are responses to directed filters which were “steered” according to the spatial gradient.
Sobel and Laplacian Example

Horizontal Sobel $\approx \frac{\partial}{\partial x}$
Vertical Sobel $\approx \frac{\partial}{\partial y}$
Laplacian $\approx \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$
LoG Example

Image Filters

$I$

Gaussian blur

Laplacian of $I$

LoG of $I$
Median Filtering

- Replace pixel \((i, j)\) with the median of the value in, say, a 5 by 5, neighborhood around it. This is a nonlinear operation.

- This is an example of a nonlinear filtering (e.g., usually \(\text{median}(\text{array1}) + (\text{array2}) \neq \text{median}(\text{array1}) + \text{median}(\text{array2})\))

Images taken from Wikipedia
A linear map from $\mathbb{R}^n$ to $\mathbb{R}^n$, $f : x \mapsto Ax$, is invertible if and only if $A$ satisfies a condition. What is this condition?

Solution

$\det A \neq 0$. 
Exercise

A linear map from $\mathbb{R}^n$ to $\mathbb{R}^n$, $f : x \mapsto Ax$, is invertible if and only if $A$ satisfies a condition. What is this condition?

Solution

det $A \neq 0$. 
Exercise

Let \( f : \mathbb{R} \to \mathbb{R} \) be defined by \( f : x \mapsto ax + b \) for some \( a, b \in \mathbb{R} \). In other words, \( f \) describes a straight line. Show that if \( b \neq 0 \) then \( f \) is a nonlinear function. Remark: people often refer to such functions as linear, and occasionally we may also do it ourselves, but, strictly speaking, this is incorrect.

Solution

Note \( f(0) = b \). If \( b \neq 0 \) then \( f \) is nonlinear since a linear function must “take zero to zero”. If \( b = 0 \) then \( f \) is linear since \( x \mapsto ax \) is linear.
Exercise

Let $f : \mathbb{R} \to \mathbb{R}$ be defined by $f : x \mapsto ax + b$ for some $a, b \in \mathbb{R}$. In other words, $f$ describes a straight line. Show that if $b \neq 0$ then $f$ is a nonlinear function. Remark: people often refer to such functions as linear, and occasionally we may also do it ourselves, but, strictly speaking, this is incorrect.

Solution

Note $f(0) = b$. If $b \neq 0$ then $f$ is nonlinear since a linear function must “take zero to zero”. If $b = 0$ then $f$ is linear since $x \mapsto ax$ is linear.
Exercise

Under what condition(s) an affine map, \( f : x \mapsto Ax + b \), is invertible?

Solution

\( \det A \neq 0 \).
Exercise

Under what condition(s) an affine map, \( f : x \mapsto Ax + b \), is invertible?

Solution

\( \det A \neq 0 \).
Exercise (Composition of affine maps)

Let \( h = g \circ f \) where \( f : \mathbb{R}^n \to \mathbb{R}^m \) and \( g : \mathbb{R}^m \to \mathbb{R}^k \) are two affine maps:

\[
\begin{align*}
    f : x &\mapsto A_1 x + b_1 \\
    g : x &\mapsto A_2 x + b_2
\end{align*}
\]

\[ A_1 \in \mathbb{R}^{m \times n} \quad b_1 \in \mathbb{R}^m ; \]

\[ A_2 \in \mathbb{R}^{k \times m} \quad b_2 \in \mathbb{R}^k . \]  \hspace{1cm} (11)

Show that \( h : \mathbb{R}^n \to \mathbb{R}^k \) is affine.

Solution

\[
h(x) = g(f(x)) = A_2 (A_1 x + b_1) + b_2 = A_2 A_1 x + A_2 b_1 + b_2 .
\]

Thus, \( h \) is affine where the linear part of \( h \) is given by the \( A_2 A_1 \) matrix while its offset part is given by \( A_2 b_1 + b_2 \).
Exercise (Composition of affine maps)

Let \( h = g \circ f \) where \( f : \mathbb{R}^n \to \mathbb{R}^m \) and \( g : \mathbb{R}^m \to \mathbb{R}^k \) are two affine maps:

\[
\begin{align*}
f : x &\mapsto A_1 x + b_1 \\
g : x &\mapsto A_2 x + b_2
\end{align*}
\]

\( A_1 \in \mathbb{R}^{m \times n} \quad b_1 \in \mathbb{R}^m ; \)

\( A_2 \in \mathbb{R}^{k \times m} \quad b_2 \in \mathbb{R}^k \). \quad (11)

Show that \( h : \mathbb{R}^n \to \mathbb{R}^k \) is affine.

Solution

\[
\begin{align*}
h(x) = g(f(x)) &= A_2(A_1 x + b_1) + b_2 \\
&= A_2 A_1 x + A_2 b_1 + b_2
\end{align*}
\]

Thus, \( h \) is affine where the linear part of \( h \) is given by the \( A_2 A_1 \) matrix while its offset part is given by \( A_2 b_1 + b_2 \).
Exercise (Composition of invertible affine maps)

Let $h = g \circ f$ where $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^n$ are two invertible affine maps. Show that $h$ is also an invertible affine map.

Solution

The linear part of $h$ is $x \mapsto A_2 A_1 x$. The matrix $A_2 A_1$ is invertible since the product of two $n \times n$ invertible matrices is invertible.
Exercise (Composition of invertible affine maps)

Let \( h = g \circ f \) where \( f : \mathbb{R}^n \to \mathbb{R}^n \) and \( g : \mathbb{R}^n \to \mathbb{R}^n \) are two invertible affine maps. Show that \( h \) is also an invertible affine map.

Solution

The linear part of \( h \) is \( x \mapsto A_2 A_1 x \). The matrix \( A_2 A_1 \) is invertible since the product of two \( n \times n \) invertible matrices is invertible.