

CS 331

Computer Vision

09 / 17 / 2026

Instructor: Michael Eckmann

Today's Topics

- Questions? / Comments?
- Review of Cross-correlation
- Define Convolution related to Cross-correlation
- Determining angle and magnitude of edge
- in-class exercises

Convolution

118

Computer Vision: Algorithms and Applications (September 20, 2020 draft)

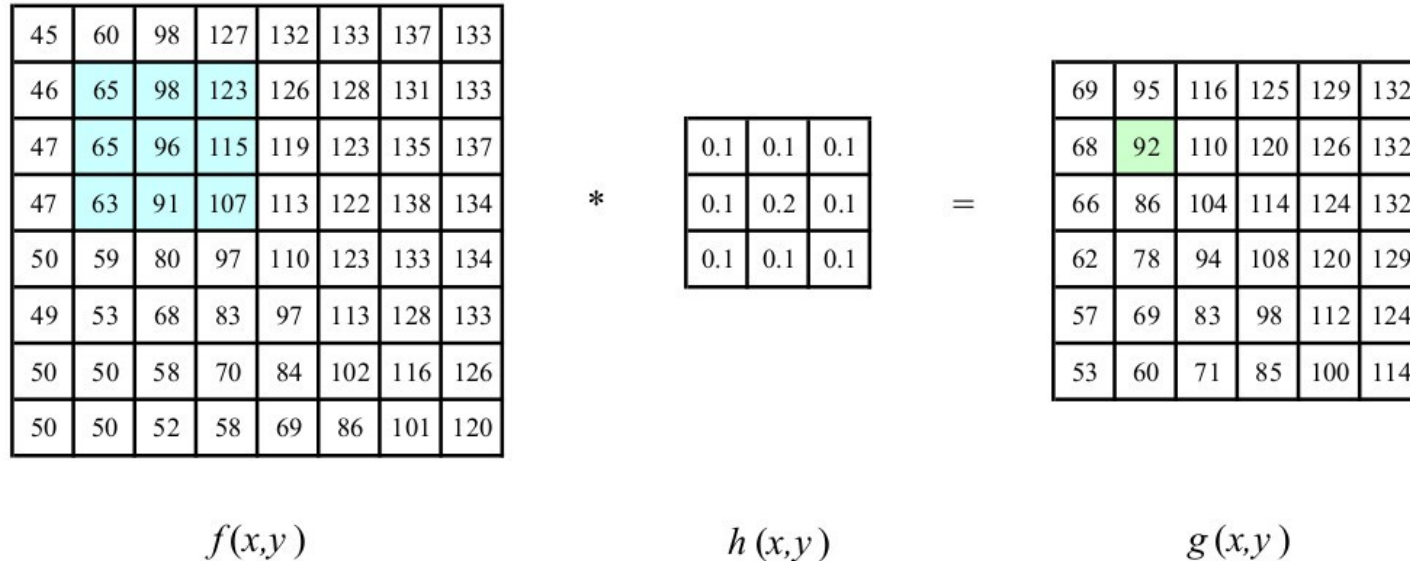


Figure 3.10 *Neighborhood filtering (convolution): The image on the left is convolved with the filter in the middle to yield the image on the right. The light blue pixels indicate the source neighborhood for the light green destination pixel.*

$$.1*65 + .1*98 + .1*123 + .1*65 + .2*96 + .1*115 + .1*63 + .1*91 + .1*107 = 91.9$$

Derivative and Smoothing masks

- Derivative masks
 - values have opposite signs in order to obtain a high response in signal regions of high contrast
 - sum of values is 0, so a 0 results from constant regions
 - first derivative masks produce high abs values at points of high contrast
 - second derivative masks produce zero-crossings at points of high contrast
- Smoothing masks
 - values are positive and sum to 1, so resulting value is same as input in constant regions, but has a blurring effect on non-constant regions

Convolution

- Convolution
 - is similar to cross-correlation except the upper-left value in the mask is multiplied by the lower-right value in the image neighborhood and so on until the lower-right value in the mask is multiplied by the upper-left value in the image.
 - think of the mask being flipped along the horizontal axis and then again along the vertical axis AND THEN doing cross-correlation with the flipped mask
- If the mask's values are symmetrical both vertically and horizontally, then cross-correlation and convolution yield the same result.
- The two terms are often confused and sometimes used interchangeably.

Computational considerations

- consider the computation involved in convolution
 - if mask is size $n \times n$, then for each pixel in the input image, we do n^2 multiplies and n^2-1 additions.
- if a mask is separable into two vectors that, when we take the outer product of them, we get the original mask then we can reduce the computational complexity by first convolving the image with the column vector and then the resulting image can be convolved with the row vector to get the output image (which would result in the same image as convolving or correlating the $n \times n$ mask with the original image).
 - Example on the board of filters that are separable (vertical Sobel, 3x3 Gaussian, or 3x3 averaging filter)
 - We would convolve the image with one vector and then convolve that result with the other vector
 - how many multiplies and additions now?

Computational considerations

- how many multiplies and additions now?
- ($2n$ multiplies and $2(n-1)$ additions) per pixel

(figure from Shapiro's Computer Vision book)

$$\text{mask } \mathbf{M} = [-1, 2, -1]$$

S_1			12	12	12	12	12	24	24	24	24	24
S_1	$\otimes$	M	0	0	0	0	-12	12	0	0	0	0

(a) S_1 is an upward step edge

S_2			24	24	24	24	24	12	12	12	12	12
S_2	$\otimes$	M	0	0	0	0	12	-12	0	0	0	0

(b) S_2 is a downward step edge

S_3			12	12	12	12	15	18	21	24	24	24
S_3	$\otimes$	M	0	0	0	-3	0	0	0	3	0	0

(c) S_3 is an upward ramp

S_4			12	12	12	12	24	12	12	12	12	12
S_4	$\otimes$	M	0	0	0	-12	24	-12	0	0	0	0

(d) S_4 is a bright impulse or "line"

Figure 5.12: Cross correlation of four special signals with second derivative edge detecting mask $\mathbf{M} = [-1, 2, -1]$; (a) upward step edge, (b) downward step edge, (c) upward ramp, and (d) bright impulse. Since the coordinates of $\mathbf{M}$ sum to zero, response on constant regions is zero. Note how a *zero-crossing* appears at an output position where different trends in the input signal join.

Gradient

- You all recall a derivative
 - Rate of change of a single variable function
- Since images are functions of two variables we talk about the gradient instead. The gradient is a vector of partial derivatives.
 - represents the direction of steepest ascent and its magnitude is the rate of increase
- Furthermore, in calculus you mostly dealt with continuous functions whereas a digital image is a discrete function

Estimating the gradient

- Often we want to find boundaries in images and know how sharp the boundaries are and in which direction the boundaries are.
- Given $f(x,y)$ as our image function, we would like to compute the gradient at pixel $[x,y]$.
- The magnitude of the gradient will give us an indication of how sharp a boundary is, and the angle of the gradient will give us the direction.
- If we want to estimate the magnitude and direction of the gradient (where the maximum change occurs) do the following:
 - the partial derivative of f w.r.t. x and the partial derivative of f w.r.t. y make up the gradient.

Estimating the gradient

- The partial derivative of f w.r.t. x (f_x) and the partial derivative of f w.r.t y (f_y) make up the gradient.
- We can estimate these values in a discrete image at pixel x,y :
 - for f_x : take the difference between the pixel at $x-1, y$ and $x+1, y$ and divide by the change in x (2 pixels)
 - for f_y : take the difference between the pixel at $x, y-1$ and $x, y+1$ and divide by the change in y (2 pixels)
- We can get a better estimate if we average these calculations above and below and to the right and left of the pixel in question.
- The magnitude of the gradient is the square root of the sum of the squares of the partials
- The direction of the gradient is the angle which is $\arctan(f_y / f_x)$

Estimating the gradient (figure from Shapiro & Stockman – Computer Vision book)

Shapiro and Stockman

163

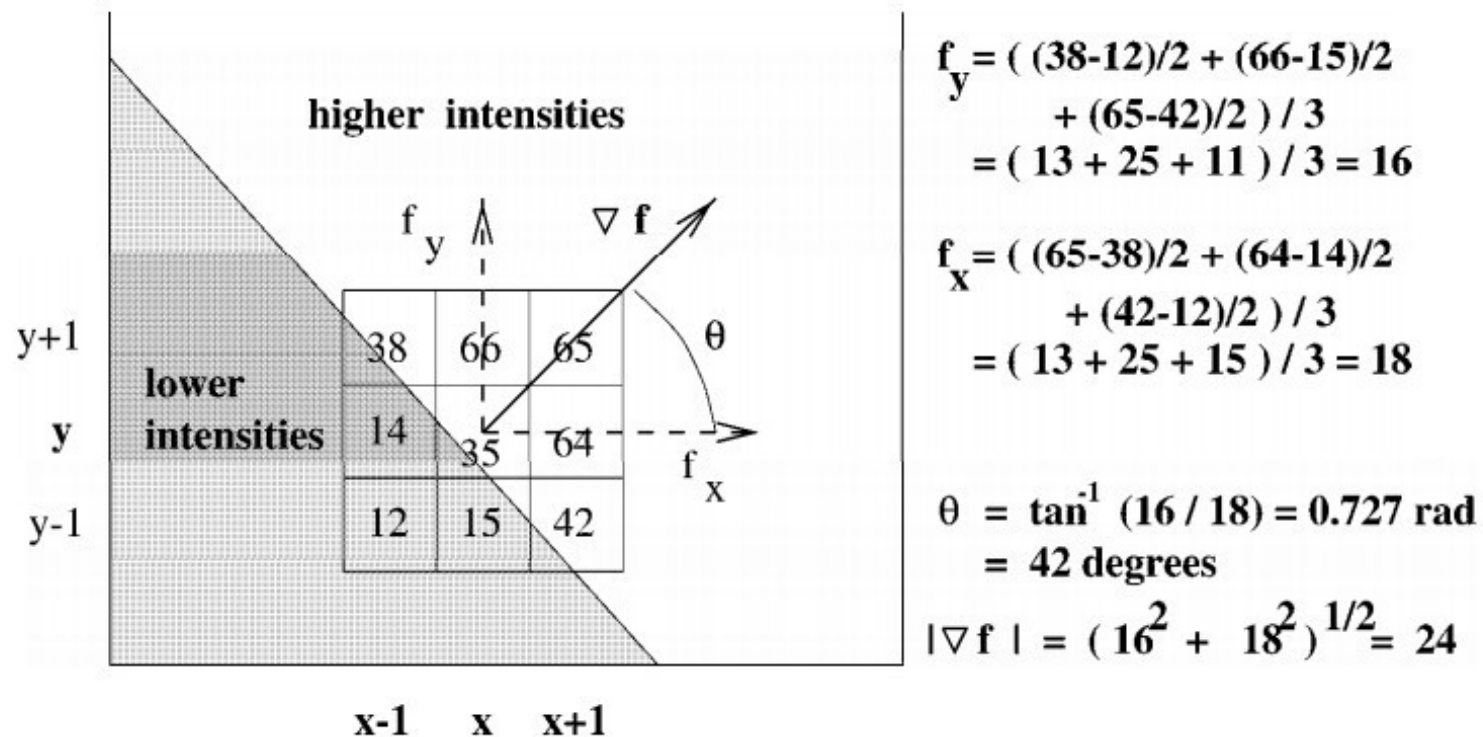


Figure 5.14: Estimating the magnitude and direction of contrast at $I[x, y]$ by estimating the gradient magnitude and direction of picture function $f(x, y)$ using the discrete samples in the image array.

Estimating the gradient

- These lead to the Prewitt masks

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} * \mathbf{A}$$

- The Sobel masks come from the same idea but with twice the weight given to the row/col the pixel is in.

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

In class exercises

Cross-correlation with horizontal Prewitt mask

Determining angle and magnitude of an edge

Cross-correlation with a smoothing mask