Computer Vision

Course 9

Image Segmentation

Segmentation subdivides an image into its constituent regions and objects. The level of detail to which the subdivision is carried depends on the problem being solved. Segmentation should stop when the objects or regions of interest in an application have been detected. For example, in the automated inspection of electronic assemblies, interest lies in analysing images of products with the objective of determining the presence or absence of specific anomalies, such as missing components or broken connection paths. There is no point in carrying segmentation past the level of detail required to identify those elements.

Segmentation of nontrivial images is one of the most difficult tasks in image processing. Segmentation accuracy determines the eventual success or failure of computerized analysis procedure.

https://darwin-public.s3.eu-west-1.amazonaws.com/splash_page/v7-vs-other-tools.mp4

Image segmentation tasks:

(https://www.v7labs.com/blog/image-segmentation-guide)

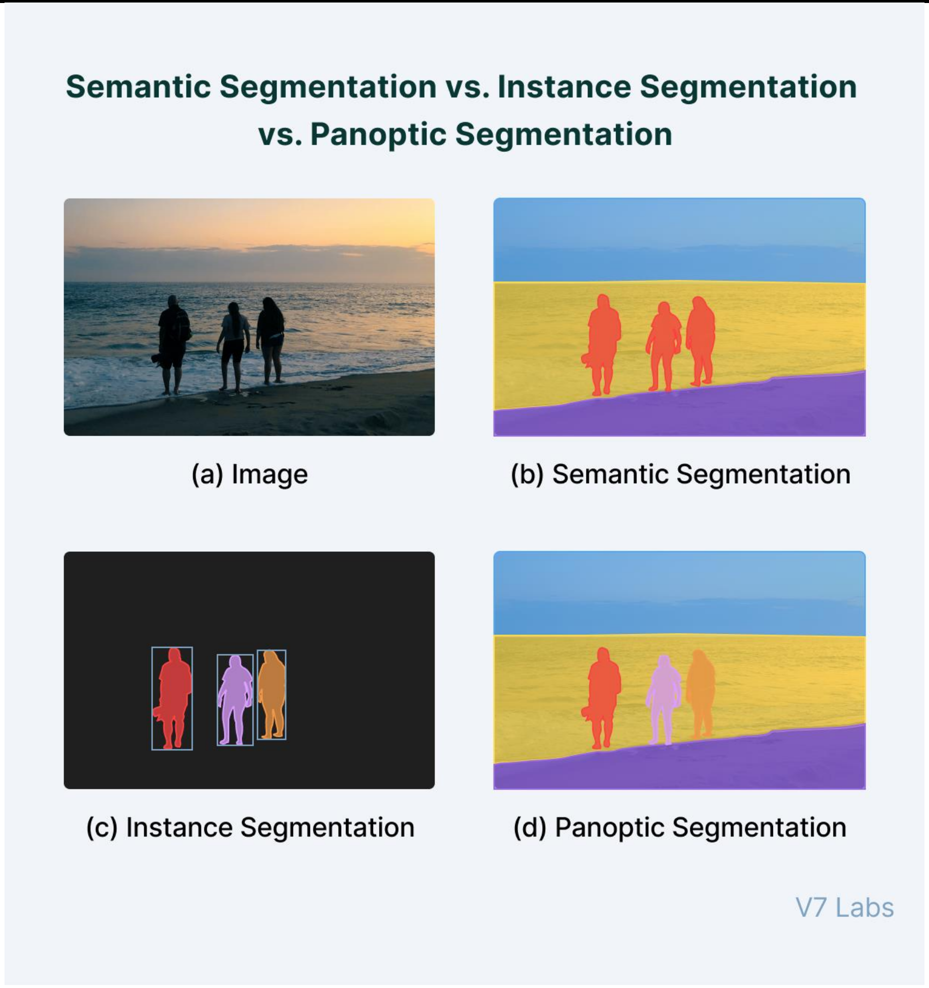
1. **Semantic segmentation** segments out a broad boundary of objects belonging to a particular class. It classifies pixels in an image into semantic classes. Pixels belonging to a particular class are simply classified to that class with no other information or context taken into consideration.

2. **Instance segmentation** provides a segment map for each object it views in the image, without any idea of the class the object belongs to. Pixels are classified into categories on the basis

of “instances” rather than classes. Overlapping or very similar object regions are distinguished on the basis of their boundaries.

3. **Panoptic segmentation** is the conjugation of instance and semantic segmentation tasks. Panoptic segmentation gives us the segment maps of all the objects of any particular class present in the image.

Course 9


Semantic Segmentation vs. Instance Segmentation vs. Panoptic Segmentation
(a) Image
(b) Semantic Segmentation
(c) Instance Segmentation
(d) Panoptic Segmentation
V7 Labs

Course 9

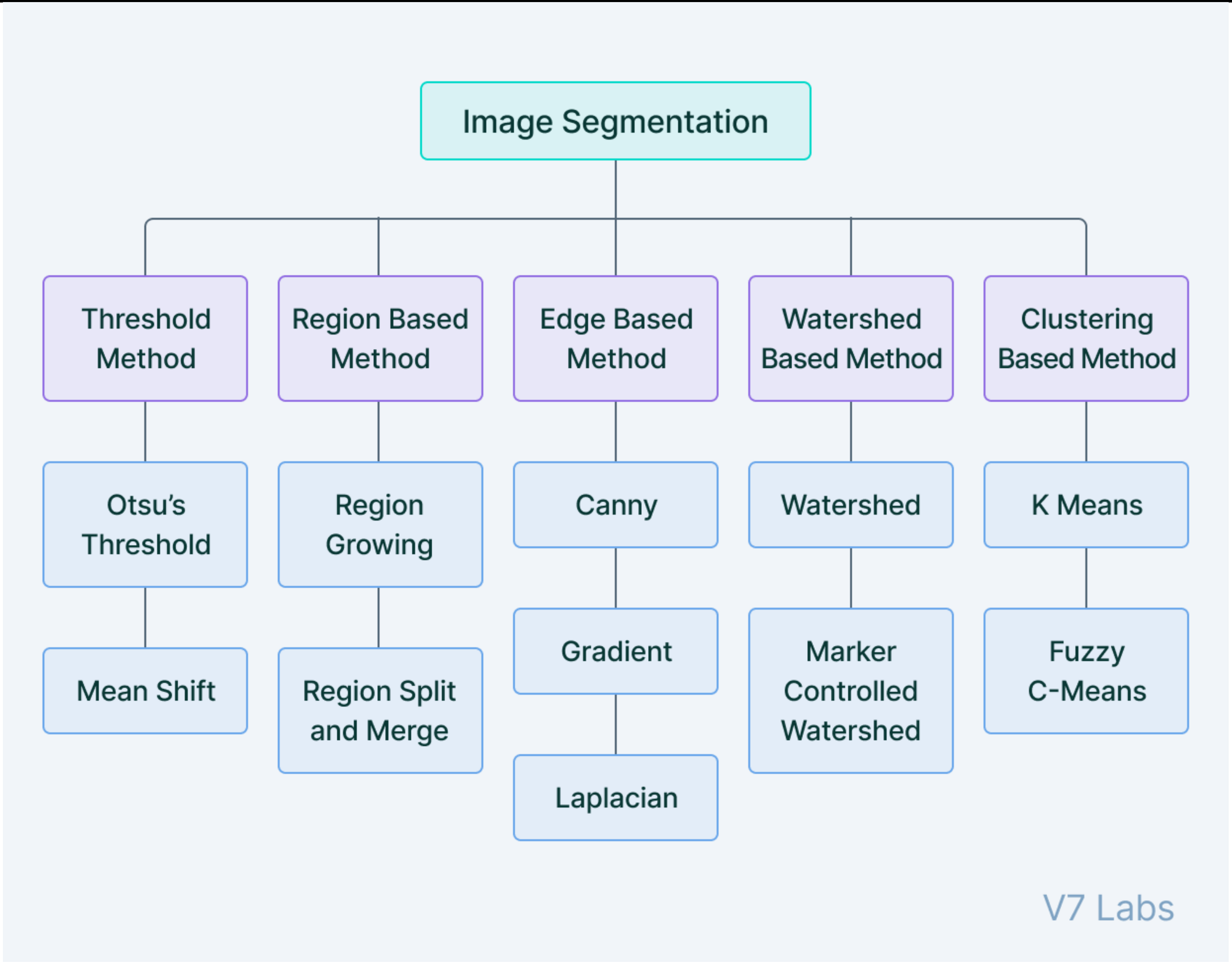


Image Segmentation
Threshold Method
Region Based Method
Edge Based Method
Watershed Based Method
Clustering Based Method
Otsu's Threshold
Region Growing
Canny
Watershed
K Means
Mean Shift
Region Split and Merge
Gradient
Marker Controlled Watershed
Fuzzy C-Means
Laplacian
V7 Labs

Fundamentals

Let ***R*** represent the entire spatial region occupied by an image. Image segmentation can be viewed as a process that partitions ***R*** into ***n*** subregions $\boldsymbol{R_1, R_2, \ldots, R_n}$ such that:

(a) $\bigcup_{i=1}^{n} R_i = R$

(b) R_i is a connected set, $i = 1,2,\ldots,n$

(c) $R_i \cap R_j = \varnothing$ for all i and j, $i \neq j$

(d) $Q(R_i) = TRUE$ for $i = 1,2,\ldots,n$

(e) $Q(R_i \cup R_j) = FALSE$ for any adjacent regions R_i , R_j

$Q(R)$ is a logical predicate defined over the points in set R.

Condition (a) indicates that the segmentation must be complete; that every pixel must be in a region. Condition (b) requires that points in a region be connected in some predefined sense (e.g. the points must be ***4***- or ***8***-connected). Condition (c) indicates that the regions must be disjoint. Condition (d) deals with the properties that must be satisfied by the pixels in a segmented region (for example $Q(R_i)$ is true if all pixels have the same intensity level). Finally, condition (e) indicates that two adjacent regions $\boldsymbol{R_i}$ and $\boldsymbol{R_j}$ must be different in the sense of predicate $\boldsymbol{Q}$.

The fundamental problem in segmentation is to partition an image into regions that satisfy the preceding conditions. Segmentation algorithms for monochrome images generally are based on one of two basic categories dealing with properties of intensity values: discontinuity and similarity. In the first category, the assumption is that boundaries of regions are sufficiently different from each other and from the background to allow boundary detection based on local discontinuities in intensity.

Edge-based segmentation is the principal approach used in this category. *Region-based segmentation* approaches from the second category are based on partitioning an image into regions that are similar according to a set of predefined criteria.

Course 9

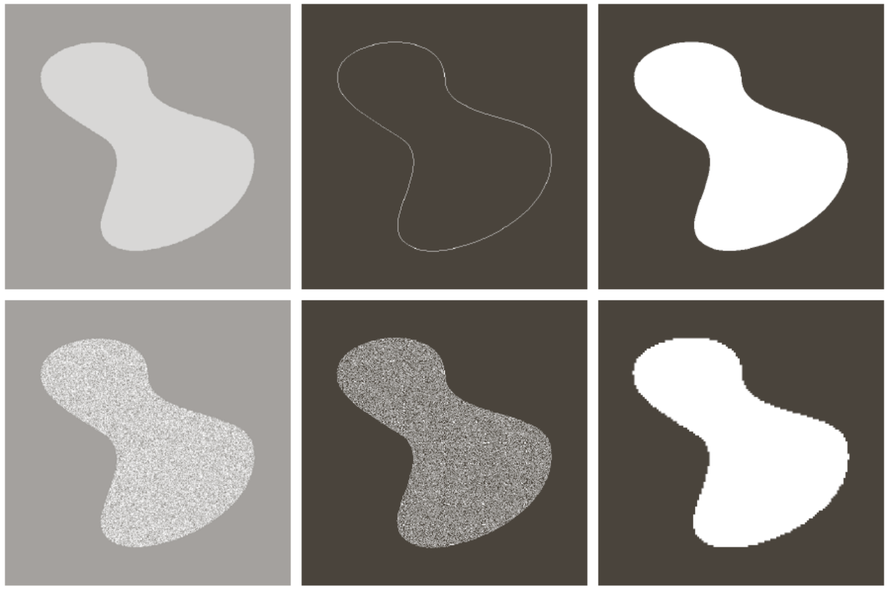


FIGURE 10.1 (a) Image containing a region of constant intensity. (b) Image showing the boundary of the inner region, obtained from intensity discontinuities. (c) Result of segmenting the image into two regions. (d) Image containing a textured region. (e) Result of edge computations. Note the large number of small edges that are connected to the original boundary, making it difficult to find a unique boundary using only edge information. (f) Result of segmentation based on region properties.

Point, Line, and Edge Detection

Edge pixels are pixels at which the intensity of an image function changes abruptly, and *edges* (or *edge segments)* are sets of connected edge pixels. *Edge detectors* are local image processing methods designed to detect edge pixels. A line may be viewed as an edge segment in which the intensity of the background on either side of the line is either much higher or much lower than the intensity of the line pixels.

Background

Abrupt, local changes in intensity can be detected using derivatives, usually first- and second-order derivatives which are defined in terms of differences.

Any approximation for a first derivative must be:

(1) zero in areas of constant intensity

(2) non-zero at the onset of an intensity step or ramp

(3) non-zero at points along an intensity ramp.

An approximation for a second derivative must be:

(1) zero in areas of constant intensity

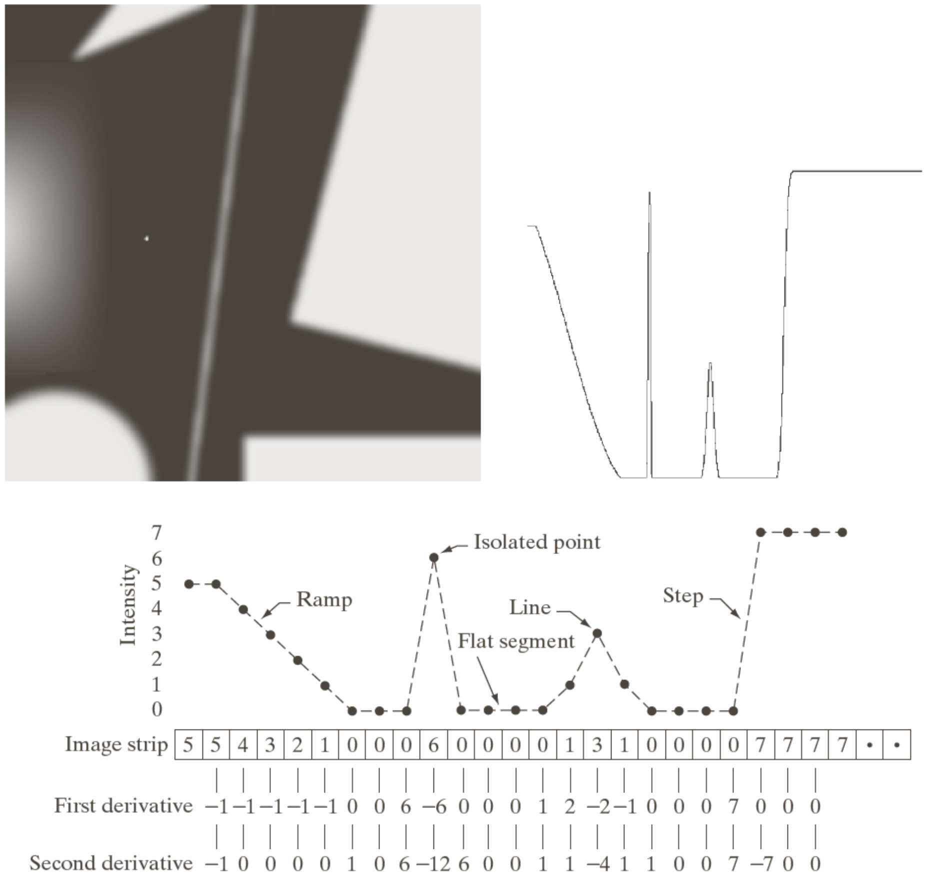
(2) nonzero at the onset and end of an intensity step or ramp

(3) zero along intensity ramps.

$$\frac{\partial f}{\partial x} \approx f(x+1,y) - f(x,y)$$

$$\frac{\partial^2 f}{\partial x^2} \approx \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial x}(x+1,y) - \frac{\partial f}{\partial x}(x,y)\right) \approx$$

$$f(x+2) - 2f(x+1) + f(x)$$



a b
c

FIGURE 10.2 (a) Image. (b) Horizontal intensity profile through the center of the image, including the isolated noise point. (c) Simplified profile (the points are joined by dashes for clarity). The image strip corresponds to the intensity profile, and the numbers in the boxes are the intensity values of the dots shown in the profile. The derivatives were obtained using Eqs. (10.2-1) and (10.2-2).

(a) First-order derivatives generally produce thicker edges in an image

(b) Second-order derivatives have a stronger response to fine detail, such as thin lines, isolated points, and noise

(c) Second-order derivatives produce a double-edge response at ramp and step transitions in intensity

(d) The sign of the second-order derivative can be used to determine whether a transition into an edge is from light to dark or dark to light.

Computing first and second derivatives at every pixel location is done using spatial filters.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

FIGURE 10.3 A general 3×3 spatial filter mask.

The *response* of the mask at the center point of the region is:

$$\boldsymbol{R = w_1 z_1 + w_2 z_2 + \cdots + w_9 z_9 = \sum_{k=1}^{9} w_k z_k} \qquad (1)$$

where $\boldsymbol{z_k}$ is the intensity of the pixel whose spatial location corresponds to the location of the ***k***-th coefficient in the mask.

Detection of isolated points

Point detection is based on computation of the second derivative of the image.

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad \text{the Laplacian}$$

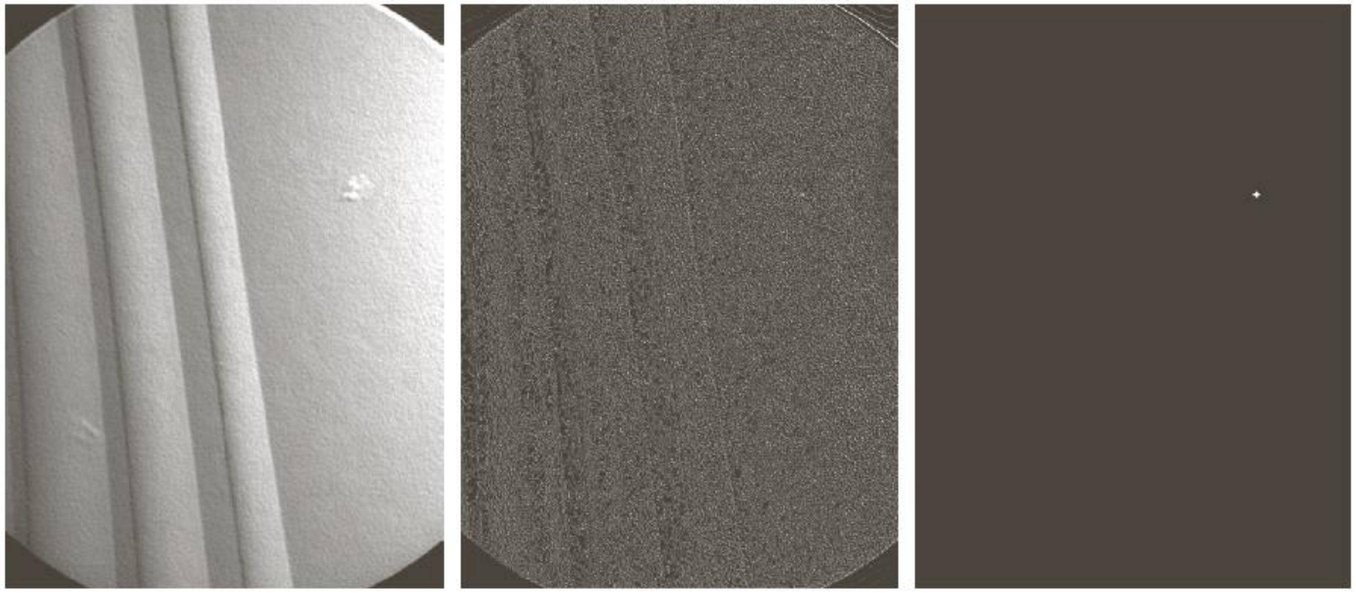
$$\frac{\partial^2 f}{\partial x^2} \approx f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f}{\partial y^2} \approx f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f(x,y) \approx f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

Course 9

1	1	1
1	−8	1
1	1	1



a
b c d

FIGURE 10.4 (a) Point detection (Laplacian) mask. (b) X-ray image of turbine blade with a porosity. The porosity contains a single black pixel. (c) Result of convolving the mask with the image. (d) Result of using Eq. (10.2-8) showing a single point (the point was enlarged to make it easier to see). (Original image courtesy of X-TEK Systems, Ltd.)

Using the Laplacian mask in Figure 10.4(a) we say that a point has been detected at the location ***(x, y)*** on which the mask is centred if the absolute value of the response of the mask at that point exceeds a specified threshold. Such points are labelled ***1*** in the output image and all others are labelled ***0***, thus producing a binary image. The output is obtained using the following expression:

$$\boldsymbol{g(x,y)} = \begin{cases} \mathbf{1} & \text{if } \left|\boldsymbol{R(x,y)}\right| \geq \boldsymbol{T} \\ \mathbf{0} & \text{otherwise} \end{cases}$$

where ***g*** is the output image, ***T*** > ***0*** is the threshold, and ***R*** is given by (1). This formulation measures the weighted difference between a pixel and its ***8***-neighbors. The idea is that the intensity of an isolated point will be quite different from its surroundings and thus will be easily detectable by this type of mask. The only differences in intensity that are considered of interest are those large enough (as determined by ***T)*** to be considered isolated points. The sum of the coefficients of the mask is zero, indicating that the mask response will be zero in areas of constant intensity.

Line Detection

For line detection, we can expect second derivatives to result in a stronger response and to produce thinner lines than first derivatives. We can use the Laplacian mask in Figure 10.4(a) for line detection also, taking care of the double-line effect of the second order derivative.

Figure 10.5(a) shows a $\boldsymbol{486 \times 486}$ (binary) portion of a wire-bond mask for an electronic circuit and Figure 10.5(b) shows its Laplacian. Scaling is necessary in this case (the Laplacian image contains negative values). Mid grey represents ***0***, darker shades of grey represent negative values, and lighter shades are positive. It might appear that negative values can be handled simply by taking the absolute value of the Laplacian image. Figure 10.5(c) shows that this approach doubles the thickness of the lines. A more suitable approach is to use only the positive values of the Laplacian (Figure 10.5(d)).

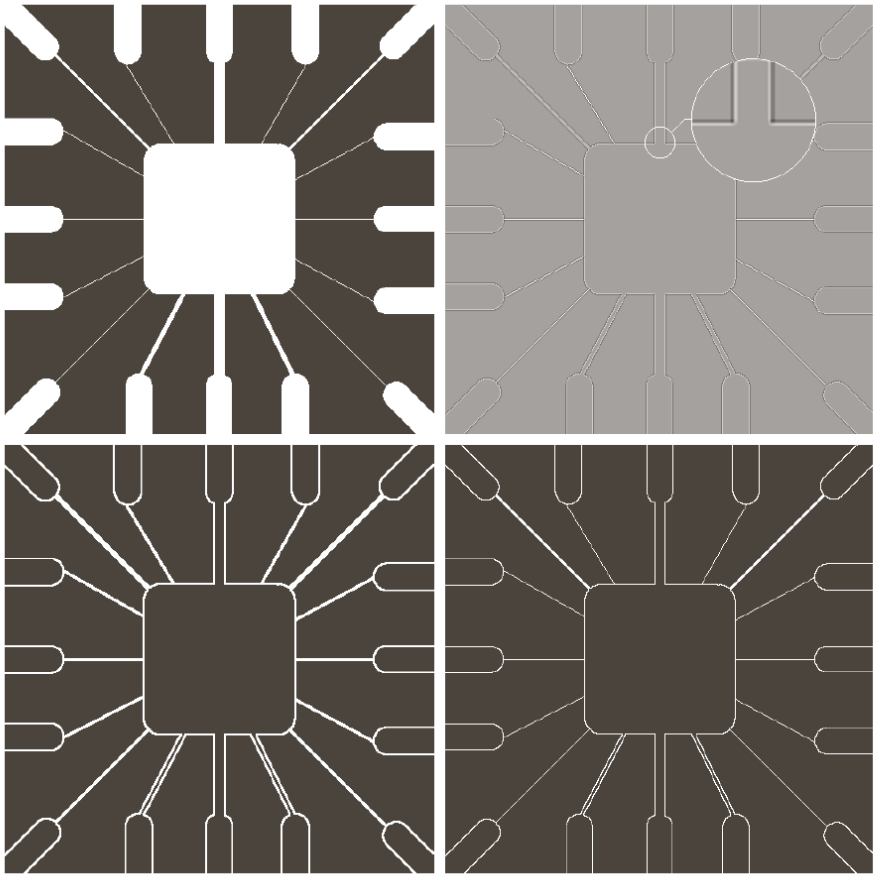


FIGURE 10.5
(a) Original image. (b) Laplacian image; the magnified section shows the positive/negative double-line effect characteristic of the Laplacian. (c) Absolute value of the Laplacian. (d) Positive values of the Laplacian.

The Laplacian detector in Figure 10.4(a) is isotropic, so its response is independent of the direction (with respect to the four directions of the $\boldsymbol{3 \times 3}$ Laplacian mask: vertical, horizontal, and two diagonals). Often, interest lies in detecting lines in *specified* directions.

Consider the masks in Figure 10.6. Suppose that an image with a constant background and containing various lines (oriented at $\boldsymbol{0^o}$, $\boldsymbol{\pm 45^o}$ and $\boldsymbol{90^o}$) is filtered with the first mask. The maximum responses would occur at image locations in which horizontal lines passed through the middle row of the mask.

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

2	−1	−1
−1	2	−1
−1	−1	2

+45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

−1	−1	2
−1	2	−1
2	−1	−1

−45°

FIGURE 10.6 Line detection masks. Angles are with respect to the axis system in Fig. 2.18(b).

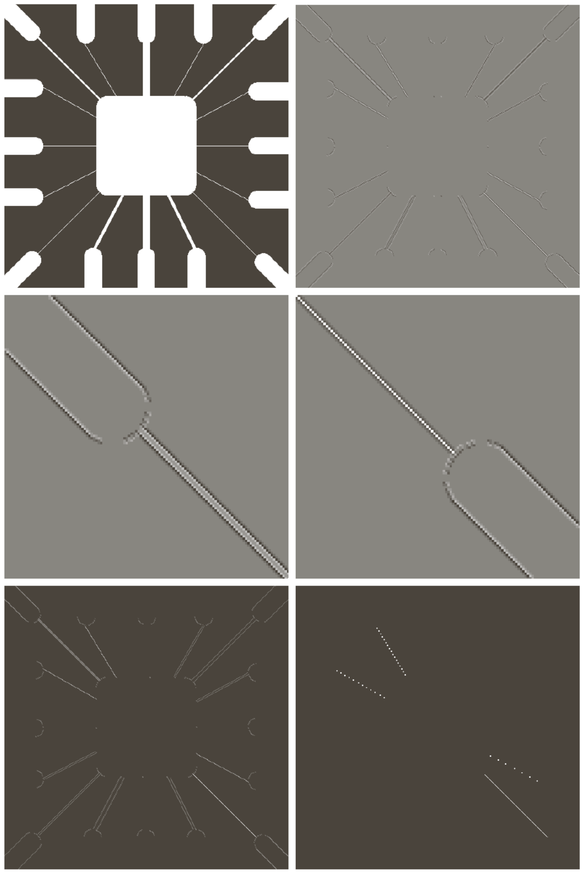
A similar experiment would reveal that the second mask in Figure 10.6 responds best to lines oriented ***+45º***; the third mask to vertical lines; and the fourth mask to lines in the ***-45º*** direction.

Let $\boldsymbol{R_1}$, $\boldsymbol{R_2}$, $\boldsymbol{R_3}$ and $\boldsymbol{R_4}$ denote the response of the masks in Figure 10.6 from left to right, where the ***R***s are given by (1). Suppose that an image is filtered (individually) with the four masks. If at a given point in the image $|\boldsymbol{R_k}| > |\boldsymbol{R_j}|$, for all $\boldsymbol{j \neq k}$, that point is said to be more likely associated with a line in the direction of mask ***k***.

If we are interested in detecting all the lines in an image in the direction defined by a given mask, we simply run the mask through the image and threshold the absolute value of the result. The points that are left are the strongest responses which, for line ***1*** pixels thick, correspond closest to the direction defined by the mask.

Course 9

In Figure 10.7(a) image, we are interested in lines oriented at ***+45º***. We use the second mask, the result is in Figure 10.7(b).



a b
c d
e f

FIGURE 10.7 (a) Image of a wire-bond template. (b) Result of processing with the +45° line detector mask in Fig. 10.6. (c) Zoomed view of the top left region of (b). (d) Zoomed view of the bottom right region of (b). (e) The image in (b) with all negative values set to zero. (f) All points (in white) whose values satisfied the condition $g \geq T$, where g is the image in (e). (The points in (f) were enlarged to make them easier to see.)

Edge Models

Edge detection is the approach used most frequently for segmenting images based on abrupt (local) changes in intensity.

Edge models are classified according to their intensity profiles. A *step edge* involves a transition between two intensity levels occurring ideally over the distance of ***1*** pixel. Figure 10.8(a) shows a section of a vertical step edge and a horizontal intensity profile through the edge.

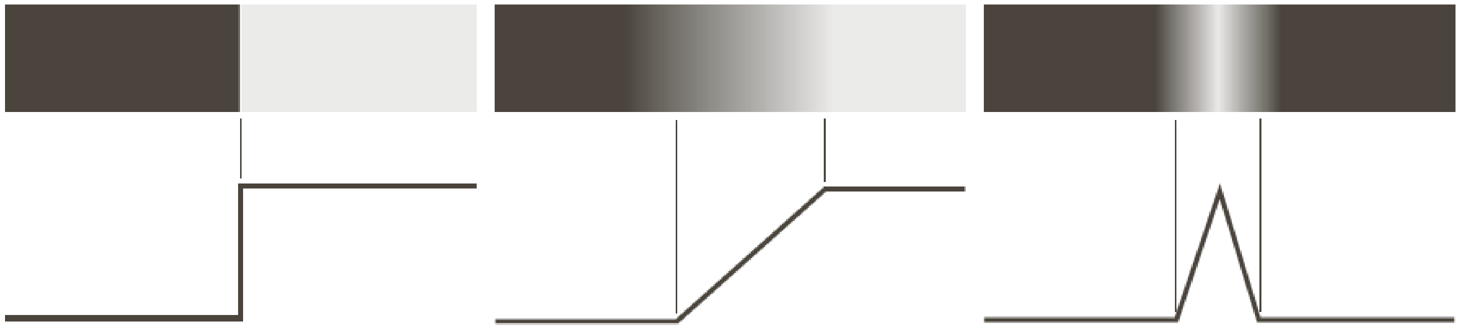


FIGURE 10.8 From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

In practice, digital images have edges that are blurred and noisy, with the degree of blurring determined principally by limitations in the focusing mechanism, and the noise level determined principally by the electronic components of the imaging system. In such situations, edges are more closely modelled as having an intensity *ramp* profile, such as the edge in Figure 10.8(b). The slope of the ramp is inversely proportional to the degree of blurring in the edge. In this model, we no longer have a thin (***1*** pixel thick) path. An edge point now is any point contained in the ramp and an edge segment would then be a set of such points that are connected.

A third model of an edge is the so-called *roof edge*, having the characteristics illustrated in Figure 10.8(c). Roof edges are models of lines through a region, with the base (width) of a roof edge being determined by the thickness and sharpness of the line.

It is not unusual to find images that contain all three types of edges.

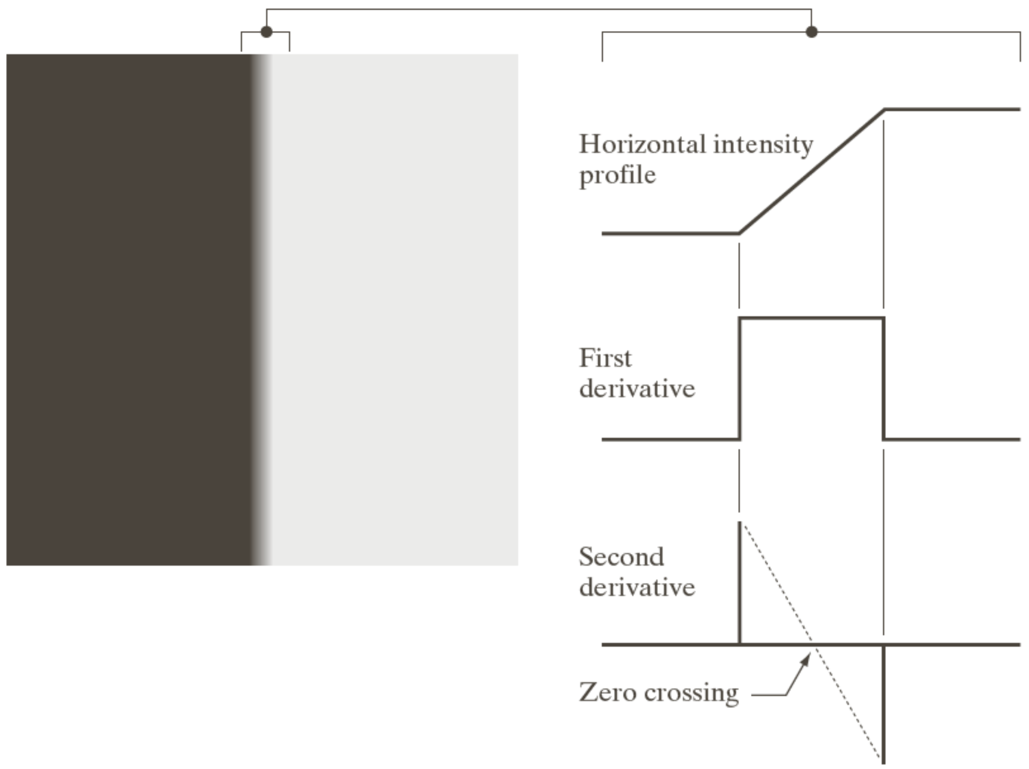
The *magnitude* of the first derivative can be used to detect the presence of an edge at a point in an image. Similarly, the *sign* of the second derivative can be used to determine whether an edge pixel lies on the dark or light side of an edge. The second derivative has the following properties:

(1) it produces two values for every edge in an image (an undesirable feature)

(2) its zero crossing can be used for locating the centres of thick edges

The *zero crossing* of the second derivative is the intersection between the zero-intensity axis and a line extending between the extrema of the second derivative.

Course 9



a b

FIGURE 10.10 (a) Two regions of constant intensity separated by an ideal vertical ramp edge. (b) Detail near the edge, showing a horizontal intensity profile, together with its first and second derivatives.

There are three fundamental steps performed in edge detection:

1. *Image smoothing for noise reduction*
2. *Detection of edge points* – this is a local operation that extracts from an image all points that are potential candidates to become edge points
3. *Edge localization* – the objective of this step is to select from the candidate points only the points that are true members of the set of points comprising an edge.

Basic Edge Detection

The image gradient and its properties

The gradient of an image is the tool for finding edge strength and direction at location ***(x, y)***:

$$\nabla f \equiv \mathrm{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

This vector has the important geometrical property that it points in the direction of the greatest rate of change for ***f*** at location ***(x,y)***.

The *magnitude (length)* of vector ∇f

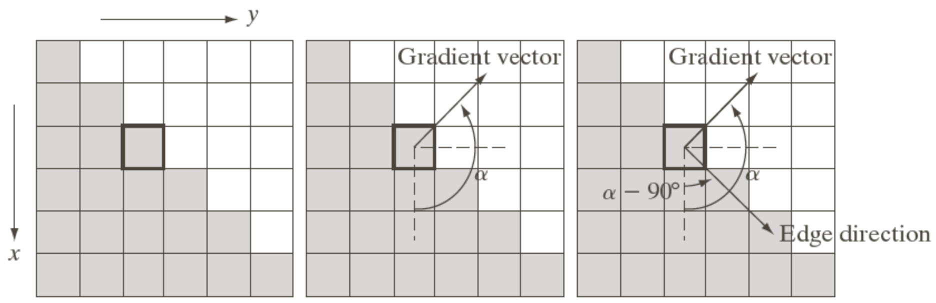
$$M(x,y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

is the value of the rate of change in the direction of the gradient vector.

The *direction* of the gradient vector is given by the angle:

$$\alpha(x,y) = \arctan\left[\frac{g_x}{g_y}\right]$$

measured with respect to the ***x***-axis. The direction of an edge at any arbitrary point ***(x, y)*** is *orthogonal* to the direction, $\alpha(x,y)$, of the gradient vector at the point.



The gradient vector sometimes is called the *edge normal*. When the vector is normalized to unit length (by dividing it by its magnitude) the resulting vector is commonly referred to as the *edge unit normal*.

Gradient operators

$$g_x = \frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y)$$

$$g_y = \frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y)$$

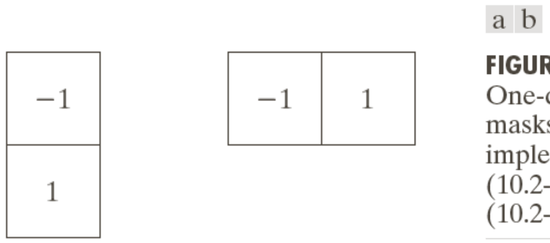


FIGURE 10.13 One-dimensional masks used to implement Eqs. (10.2-12) and (10.2-13).

When diagonal edge direction is of interest, we need a 2-D mask. The *Roberts cross-gradient operators* are one of the earliest attempts to use 2-D masks with a diagonal preference. Consider the ***3×3*** region in Figure 10.14(a). The Roberts operators are based on implementing the diagonal differences.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0
0	1

0	−1
1	0

Roberts

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

	a
b	c
d	e
f	g

FIGURE 10.14
A 3 × 3 region of an image (the z's are intensity values) and various masks used to compute the gradient at the point labeled z_5.

$$g_x = \frac{\partial f}{\partial x} = z_9 - z_5 = f(x+1, y+1) - f(x, y)$$

$$g_y = \frac{\partial f}{\partial y} = z_8 - z_6 = f(x+1, y) - f(x, y+1)$$

Masks of size 2×2 are simple conceptually, but they are not as useful for computing edge direction as masks that are symmetric about the centre point, the smallest of which are of size 3×3.

Prewitt operators

$$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$

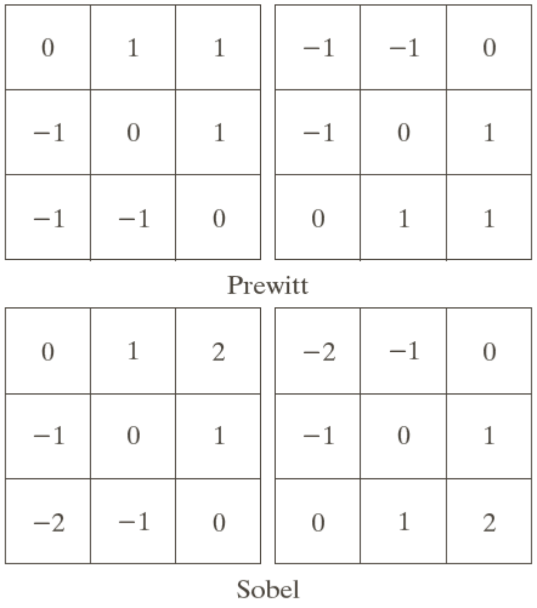
$$g_y = \frac{\partial f}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

Sobel operators

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

The Sobel masks have better noise-suppression (smoothing) effects than the Prewitt masks.



0	1	1
−1	0	1
−1	−1	0

−1	−1	0
−1	0	1
0	1	1

Prewitt

0	1	2
−1	0	1
−2	−1	0

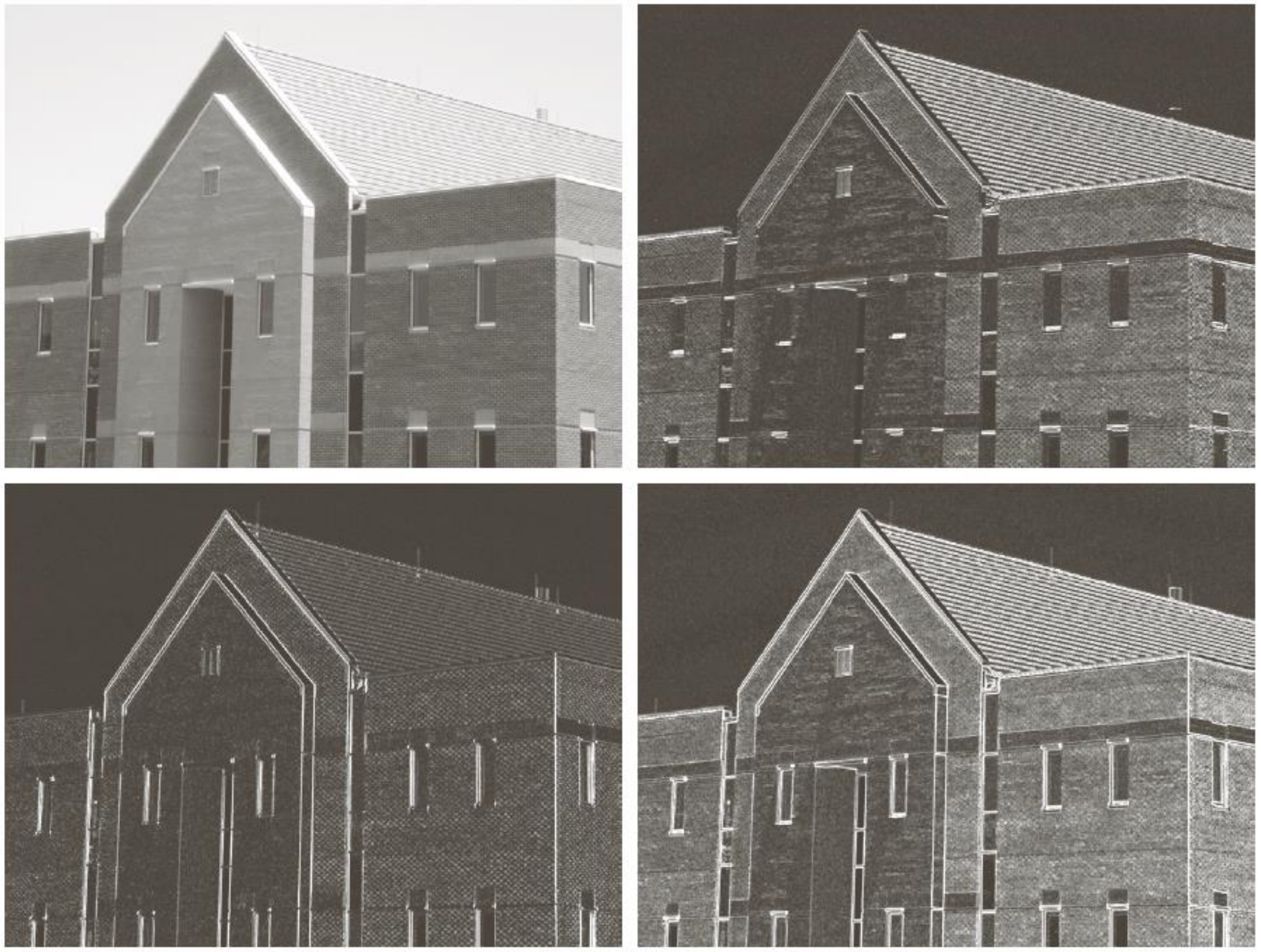
−2	−1	0
−1	0	1
0	1	2

Sobel

a b
c d

FIGURE 10.15 Prewitt and Sobel masks for detecting diagonal edges.

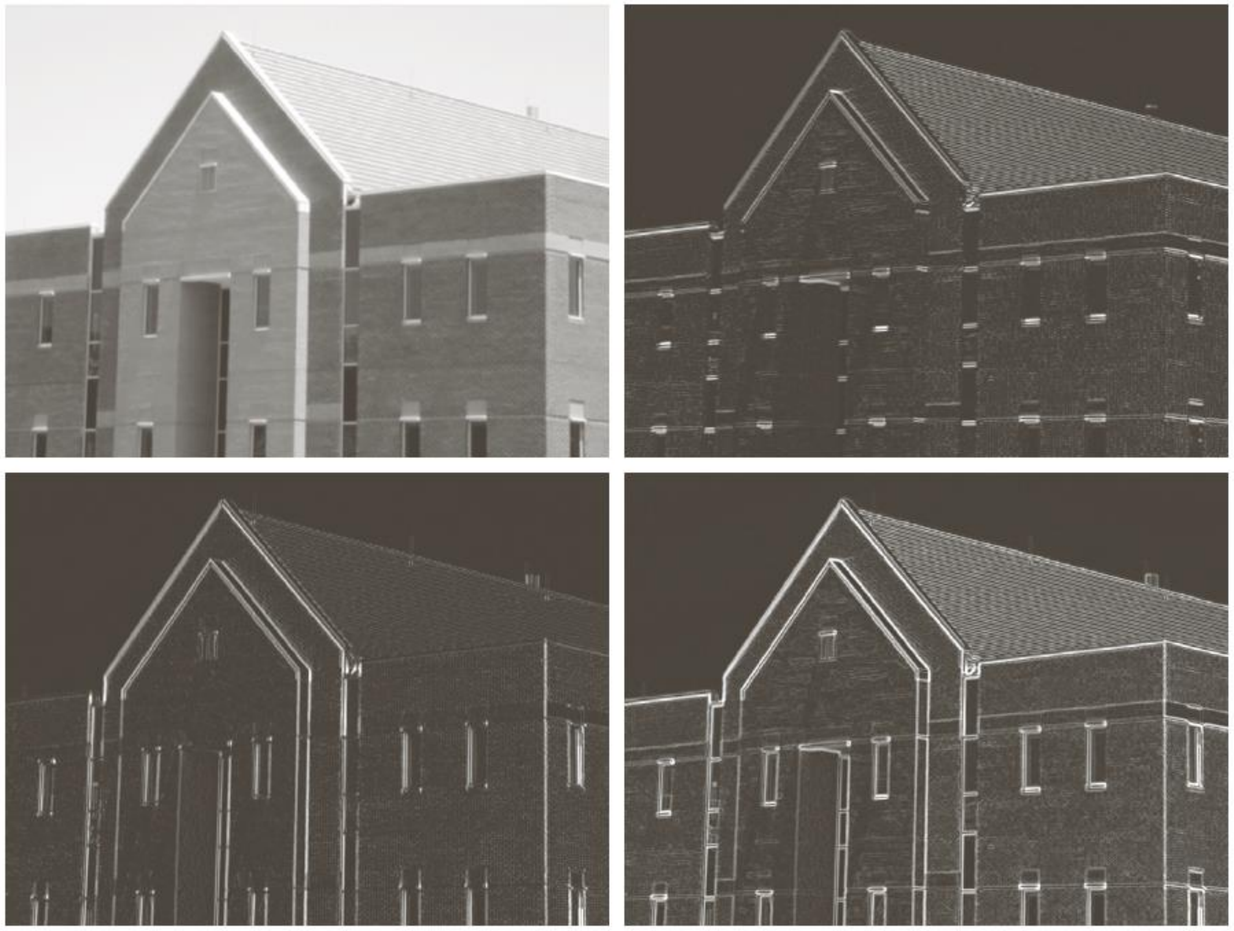
Course 9



a b
c d

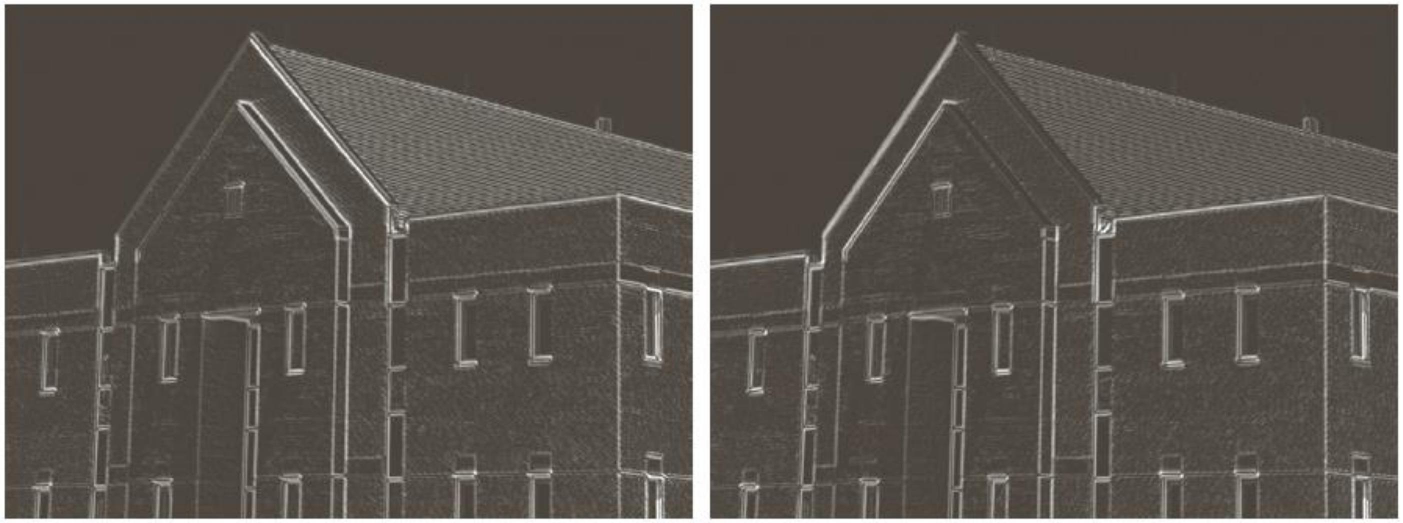
FIGURE 10.16 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) $|g_x|$, the component of the gradient in the x-direction, obtained using the Sobel mask in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the mask in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Course 9



a b
c d

FIGURE 10.18 Same sequence as in Fig. 10.16, but with the original image smoothed using a 5 × 5 averaging filter prior to edge detection.

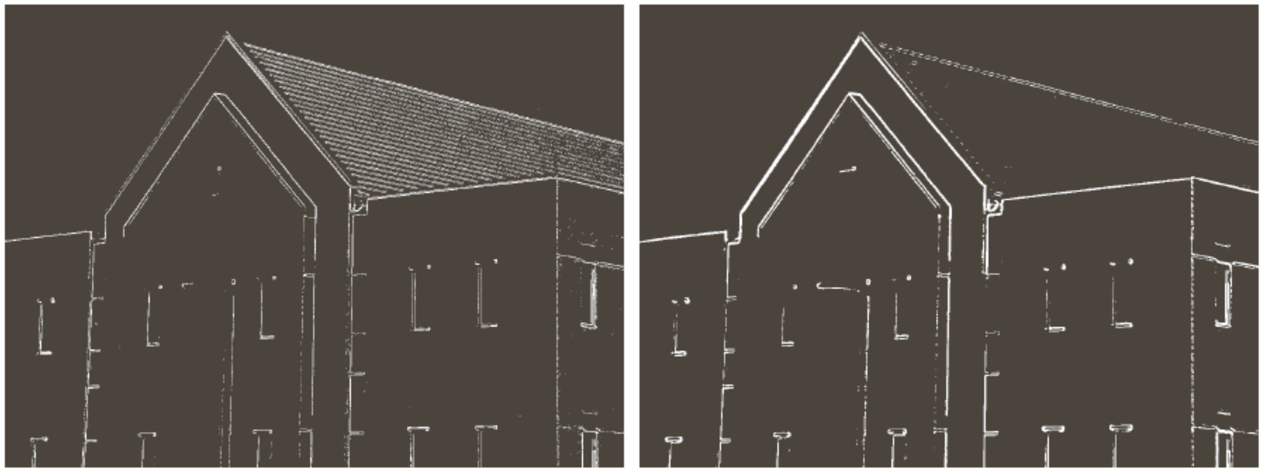


a b

FIGURE 10.19 Diagonal edge detection. (a) Result of using the mask in Fig. 10.15(c). (b) Result of using the mask in Fig. 10.15(d). The input image in both cases was Fig. 10.18(a).

When interest lies both in highlighting the principal edges and on maintaining as much connectivity as possible, it is common practice to use both smoothing and thresholding.

Course 9



a b

FIGURE 10.20 (a) Thresholded version of the image in Fig. 10.16(d), with the threshold selected as 33% of the highest value in the image; this threshold was just high enough to eliminate most of the brick edges in the gradient image. (b) Thresholded version of the image in Fig. 10.18(d), obtained using a threshold equal to 33% of the highest value in that image.

More Advanced Techniques for Edge Detection

The edge-detection methods described until now are based on filtering an image with one or more masks, without approaching the edge characteristics or the noise content of the image. In this section, the noise and the nature of the edges are considered in more advanced edge-detection techniques.

The Marr-Hildreth edge detector

Marr and Hildreth noticed that:

(1) intensity changes are not independent of image scale and so their detection requires the use of operators of different sizes;

(2) a sudden intensity change will give rise to a peak or trough in the first derivative or, equivalently, to a zero crossing in the second derivative.

These ideas suggest that an operator used for edge detection should have two features:

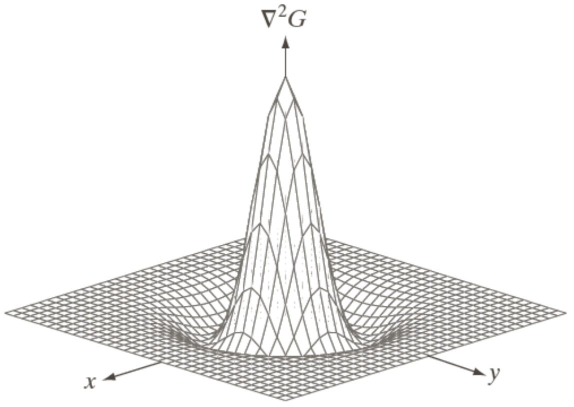
1) it should be a differential operator capable of computing a digital approximation of the first or second derivative at every point in the image
2) it should be capable of being “tuned” to act at any desired scale, so that large operators can be used to detect blurry edges and small operators to detect sharply focused fine detail.

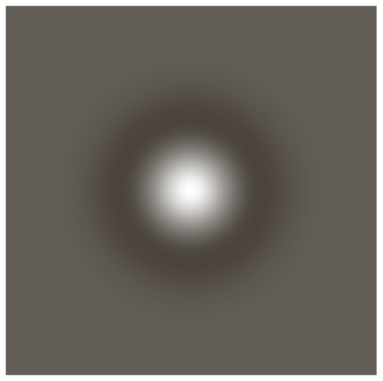
Marr and Hildreth argued that the most satisfactory operator fulfilling these conditions is the filter $\nabla^2 G$, the Laplacian of G, the 2-D Gaussian function with standard deviation σ:

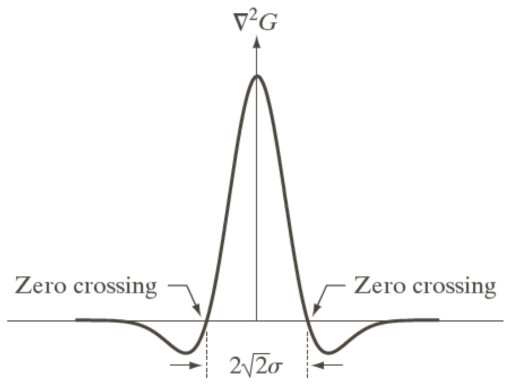
$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (2)$$

$$\nabla^2 G(x,y) = \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (3)$$

The last expression is called the *Laplacian of a Gaussian* (LoG).







0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

a	b
c	d

FIGURE 10.21 (a) Three-dimensional plot of the *negative* of the LoG. (b) Negative of the LoG displayed as an image. (c) Cross section of (a) showing zero crossings. (d) 5×5 mask approximation to the shape in (a). The negative of this mask would be used in practice.

Because of the shape illustrated in Figure 10.21(a), the LoG function sometimes is called the *Mexican hat* operator. Figure 10.21(d) shows a 5×5 mask that approximates the shape in Figure 10.21(a) (in practice, the *negative* of this mask is used). This approximation is not unique. Its purpose is to capture the essential *shape* of the LoG function.

Masks of arbitrary size can be generated by sampling equation (3) and scaling the coefficients so that they sum to zero. A more effective approach for generating LoG filters is to sample equation (2) to the desired ***n × n*** size and then convolve the resulting array with a Laplacian mask, such as the mask in Figure 10.4 (a).

The Marr-Hildreth algorithm consists of convolving the LoG filter with an input image ***f(x,y)***

$$\boldsymbol{g(x,y)=\left[\nabla^2 G(x,y)\right]\blacklozenge f(x,y)=\nabla^2\left[G(x,y)\blacklozenge f(x,y)\right]}$$

The Marr-Hildreth edge-detection algorithm may be summarized as follows:

1. Filter the input image with an $\boldsymbol{n \times n}$ Gaussian lowpass filter obtained by sampling equation (2)
2. Compute the Laplacian of the image resulting in Step 1
3. Find the zero crossing of the image from Step 2.

The size of an $\boldsymbol{n \times n}$ LoG discrete filter should be such that $\boldsymbol{n}$ is the smallest odd integer greater than or equal to $\boldsymbol{6\sigma}$. Choosing a filter mask smaller than this will tend to "truncate" the LoG function, with the degree of truncation being inversely proportional to the size of the mask; using a larger mask would make little difference in the result.

One approach for finding the zero crossing at any pixel ***p*** of the filtered image, ***g(x,y)***, is based on using a $\mathbf{3 \times 3}$ neighbourhood centred at ***p***. A zero crossing at ***p*** implies that the *signs* of at least two of its opposing neighbouring pixel must differ. There are 4 cases to test: left/right, up/down, and the two diagonals.

Course 9

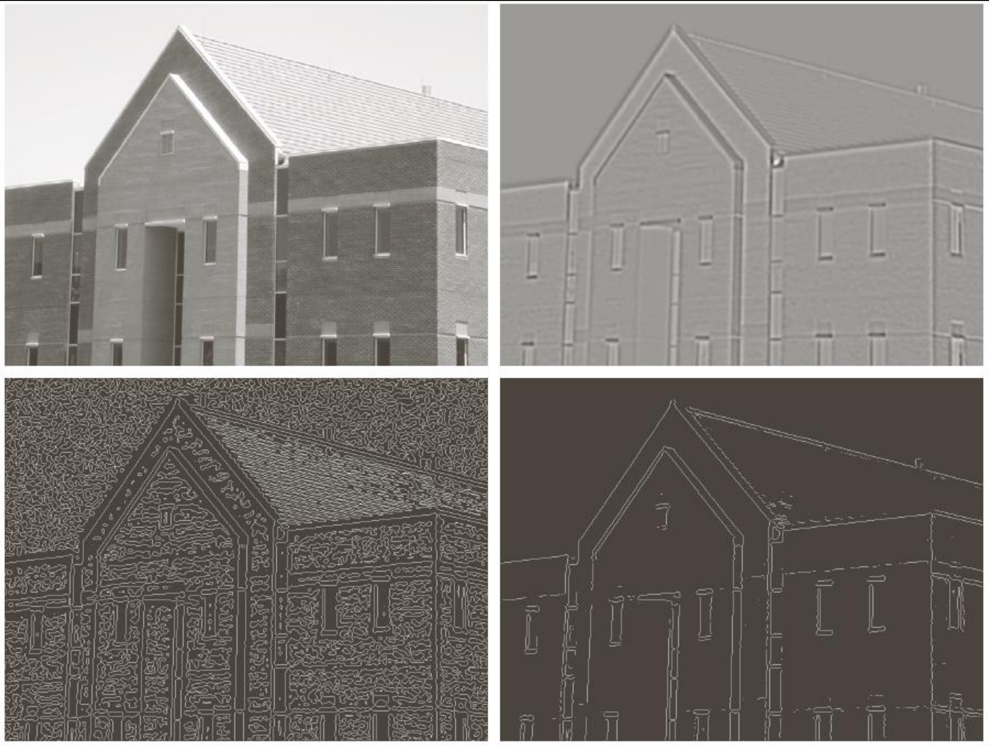


FIGURE 10.22 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) Results of Steps 1 and 2 of the Marr-Hildreth algorithm using $\sigma = 4$ and $n = 25$. (c) Zero crossings of (b) using a threshold of 0 (note the closed-loop edges). (d) Zero crossings found using a threshold equal to 4% of the maximum value of the image in (b). Note the thin edges.

The Canny edge detector

Canny's approach is based on three basic objectives:

1. *Low error rate.* All edges should be found, and there should be no false responses. The detected edges must be as close as possible to the true edges
2. *Edge points should be well localized.* The edges located must be as close as possible to the true edges, that is, the distance between a point marked as an edge by the detector and the centre of the true edge should be minim.

3. *Single edge response*. The detector should return only one point for each true edge point. That is, the number of local maxima around the true edge should be minim. This means that the detector should not identify multiple edge pixels where only a single edge point exists.

In general, it is difficult (or impossible) to find a closed form solution that satisfy all the preceding objectives. However, using numerical optimization with 1-D step edges corrupted by additive white Gaussian noise led to the conclusion that a good approximation to the optimal step edge detector is the *first derivative of a Gaussian*:

$$\frac{d}{dx} e^{-\frac{x^2}{2\sigma^2}} = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}.$$

Let ***f(x, y)*** denote the input image and ***G(x, y)*** denote the Gaussian function:

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}.$$

Form a smoothed, $f_s(x, y)$, by convolving G and f:

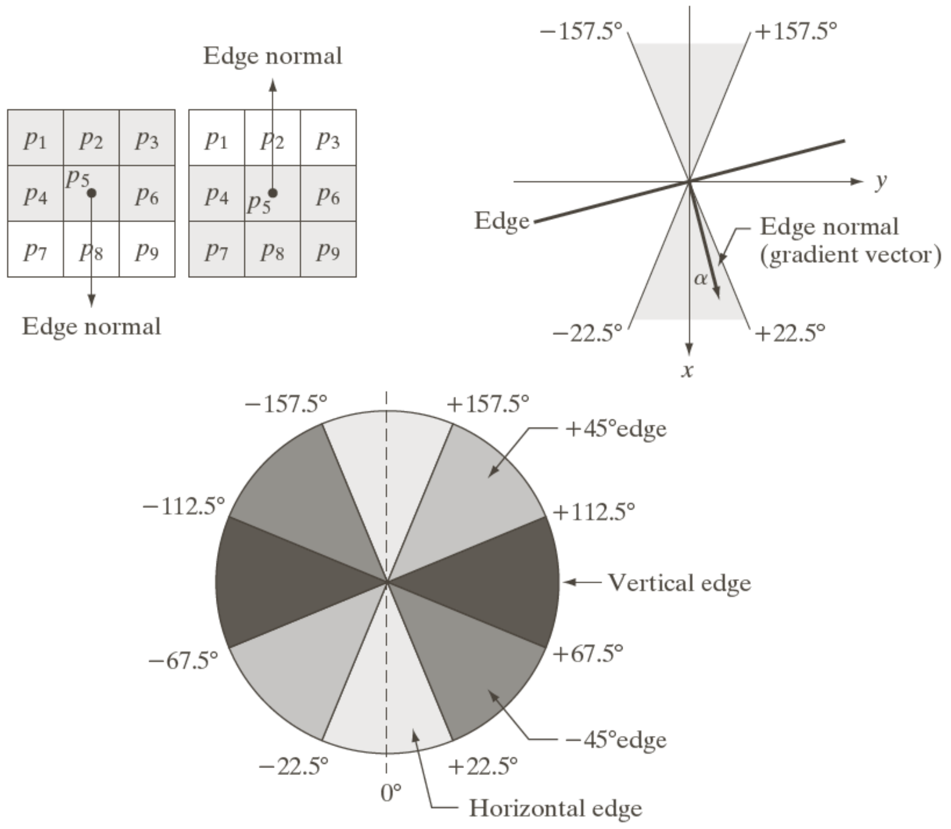
$$f_s(x,y) = G(x,y) \blacklozenge f(x,y).$$

We compute the gradient magnitude and the angle for f_s

$$M(x,y) = \sqrt{g_x^2 + g_y^2}\ , \quad g_x = \frac{\partial f_s}{\partial x}\ , \quad g_y = \frac{\partial f_s}{\partial y}$$

$$\alpha(x,y) = \arctan\left[\frac{g_x}{g_y}\right]$$

$M(x, y)$ contains ridges around local maxima. The next step is to thin those ridges. One approach is to use *non-maxima* *suppression*. This can be done in several ways, but the essence of this approach is to specify a number of discrete orientations of the edge normal (gradient vector). For example, in a $\boldsymbol{3\times 3}$ region we can define four orientations for an edge passing through the centre point of the image: horizontal, vertical, $\boldsymbol{+45^o}$ and $\boldsymbol{-45^o}$.



a b
c

FIGURE 10.24
(a) Two possible orientations of a horizontal edge (in gray) in a 3 × 3 neighborhood. (b) Range of values (in gray) of α, the direction angle of the *edge normal*, for a horizontal edge. (c) The angle ranges of the edge normals for the four types of edge directions in a 3 × 3 neighborhood. Each edge direction has two ranges, shown in corresponding shades of gray.

Let $\boldsymbol{d_1}$, $\boldsymbol{d_2}$, $\boldsymbol{d_3}$ and $\boldsymbol{d_4}$ denote the four basic edge directions for a ***3×3*** region: horizontal, ***-45º***, vertical, and ***+45º***, respectively.

We can formulate the following non-maxima suppression scheme for a 3×3 region centred at every point (x,y) in $\alpha(x,y)$:

1. Find the direction d_k that is closest to $\alpha(x,y)$.
2. If the value of $M(x,y)$ is less than at least one of its two neighbours along d_k, let $g_N(x,y)=0$ (suppression); otherwise, let $g_N(x,y)=M(x,y)$, where $g_N(x,y)$ is the non-maxima-supressed image.

The final operation is to threshold $\boldsymbol{g_N(x, y)}$ to reduce false edge points. If we set the threshold to low, there will still be some false edges (called *false positive*). If the threshold is too high, then actual valid edge points will be eliminated (*false negative*). Canny's algorithm attempts to improve on this situation by using *hysteresis thresholding*, which uses two thresholds: a low threshold $\boldsymbol{T_L}$ and a high threshold $\boldsymbol{T_H}$. Canny suggested that the ratio of the high to low threshold should be two or three to one.

We can visualize the thresholding operation as creating two additional images

$$g_{NH}(x,y) = g_N(x,y) \geq T_H$$
$$g_{NL}(x,y) = g_N(x,y) \geq T_L$$
$$g_{NH} = g_{NL} = 0 - \text{initially}$$

After thresholding $g_{NH}(x,y)$ will have fewer nonzero pixels than $g_{NL}(x,y)$ in general, but all the nonzero pixels in $g_{NH}(x,y)$ will be contained in $g_{NL}(x,y)$ because the later image is formed with a lower threshold. We eliminate from $\boldsymbol{g_{NL}(x,y)}$ all the nonzero pixels from $\boldsymbol{g_{NH}(x,y)}$ by letting:

$$\boldsymbol{g_{NL}(x,y) = g_{NL}(x,y) - g_{NH}(x,y)}$$

The nonzero pixels in $\boldsymbol{g_{NH}(x,y)}$ and $\boldsymbol{g_{NL}(x,y)}$ may be viewed as being “strong” and “weak” edge pixels.

After the thresholding operation, all strong pixels in $\boldsymbol{g_{NH}(x,y)}$ are assumed to be valid edge pixels and are so marked immediately. Depending on the value of $\boldsymbol{T_H}$, the edges in $g_{NH}(x,y)$ typically have gaps. Longer edges are formed using the following procedure:

(a) Locate the next unvisited edge pixel, p, in $g_{NH}(x,y)$.

(b) Mark as valid edge pixels all the pixels in $g_{NL}(x,y)$ that are connected to p (using 8-connectivity, for example)

(c) If all nonzero pixels in $g_{NH}(x,y)$ have been visited go to Step (d). Else return to Step (a).

(d) Set to zero all pixels in $g_{NL}(x,y)$ that were not marked as valid edge pixels.

At the end of this procedure, the final image output by the Canny algorithm is formed by appending to $\boldsymbol{g_{NH}(x,y)}$ all the nonzero pixels from $\boldsymbol{g_{NL}(x,y)}$.

In practice, hysteresis thresholding can be implemented directly during non-maxima suppression, and thresholding can be implemented directly on $\boldsymbol{g_N(x,y)}$ by forming a list of strong pixels and the weak pixels connected to them.

Canny edge detection algorithm consists of the following basic steps:

1. Smooth the input image with a Gaussian filter.
2. Compute the gradient magnitude and angle images.
3. Apply non-maxima suppression to the gradient magnitude image.
4. Use double thresholding and connectivity analysis to detect and link edges.