Errata

for

Burger, Burge: Principles of Digital Image Processing – Core Algorithms

www.imagingbook.com

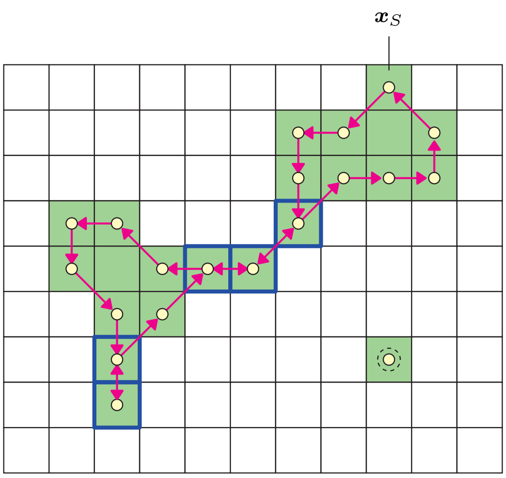


Figure 2.8 The path along a contour as an ordered sequence of pixel coordinates with a given start point $\boldsymbol{x}_S$. Individual pixels may occur (be visited) more than once within the path, and a region consisting of a single isolated pixel will also have a contour (bottom right).

2.2.2 Combining Region Labeling and Contour Finding

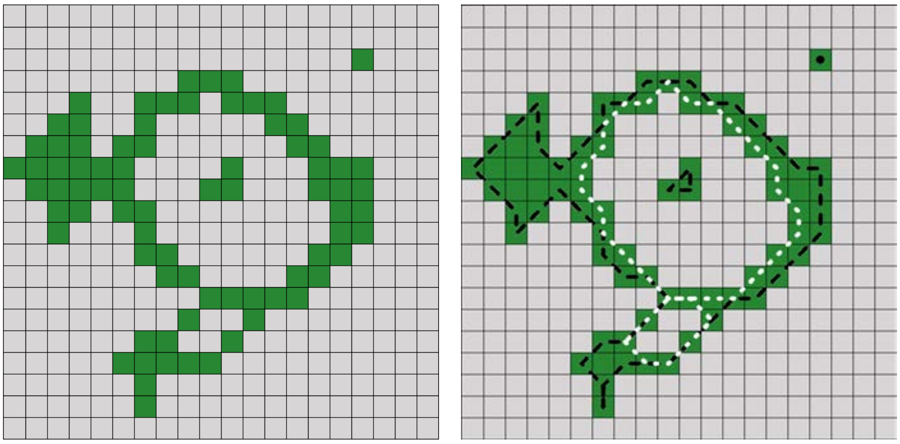
This method, based on [18], combines the concepts of sequential region labeling (Sec. 2.1) and traditional contour tracing into a single algorithm able to perform both tasks simultaneously during a single pass through the image. It identifies and labels regions and at the same time traces both their inner and outer contours. The algorithm does not require any complicated data structures and is very efficient when compared with other methods with similar capabilities. The key steps of this method are described below and illustrated in Fig. 2.9:

1. As in the sequential region labeling (Alg. 2.3), the binary image I is traversed from the top left to the bottom right. Such a traversal ensures that all pixels in the image are eventually examined and assigned an appropriate label.

2. At a given position in the image, the following cases may occur:

   Case A: The transition from a background pixel to a previously unmarked foreground pixel (A in Fig. 2.9 (a)) means that this pixel lies on the outer edge of a new region. A new *label* is assigned and the associated *outer* contour is traversed and marked by calling the method TraceContour() (see Fig. 2.9 (a) and Alg. 2.5 (line 19)). Furthermore, all background pixels directly bordering the region are marked with the special label -1.

   Case B: The transition from a foreground pixel (B in Fig. 2.9 (b)) to an



(a) (b)

Figure 2.10 Combined contour and region marking: original image in gray (a), located contours (b) with black lines for out and white lines for inner contours. Outer contours of one-pixel regions (for example, in the upper-right of (b)) are marked by a single dot.

running trough the centers of the contour pixels, and inner contours are drawn white. Contours of single-pixel regions are marked by small circles filled with the corresponding color. Figure 2.11 shows the results for a larger section taken from a real image (Vol. 1 [14, Fig. 7.12]).

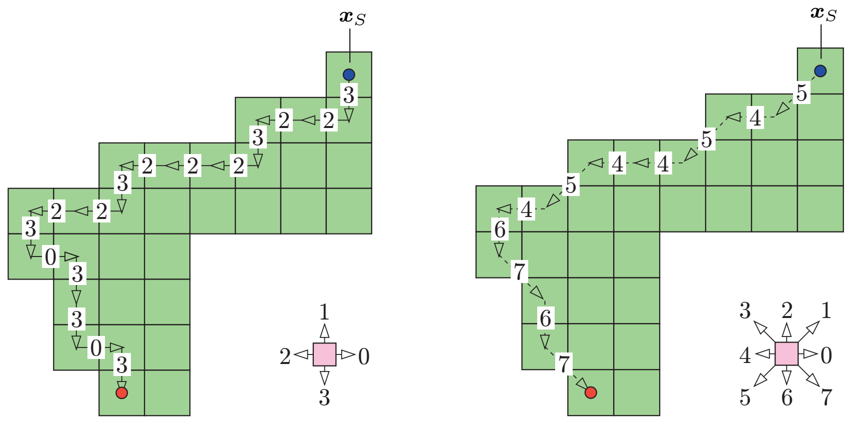
2.3 Representing Image Regions

2.3.1 Matrix Representation

A natural representation for images is a matrix (that is, a two-dimensional array) in which elements represent the intensity or the color at a corresponding position in the image. This representation lends itself, in most programming languages, to a simple and elegant mapping onto two-dimensional arrays, which makes possible a very natural way to work with raster images. One possible disadvantage with this representation is that it does not depend on the content of the image. In other words, it makes no difference whether the image contains only a pair of lines or is of a complex scene because the amount of memory required is constant and depends only on the dimensions of the image.

Regions in an image can be represented using a logical mask in which the area within the region is assigned the value *true* and the area without the value *false* (Fig. 2.12). Since Boolean values can be represented by a single bit, such a matrix is often referred to as a "bitmap".[5]

[5] In Java, variables of the type `boolean` are represented internally within the Java



4-Chain Code
3223222322303303...111
length = 28

8-Chain Code
54544546767...222
length = $16 + 6\sqrt{2} \approx 24.5$

Figure 2.14 Chain codes with 4- and 8-connected neighborhoods. To compute a chain code, begin traversing the contour from a given starting point $\boldsymbol{x}_S$. Encode the relative position between adjacent contour points using the directional code for either 4-connected (left) or 8-connected (right) neighborhoods. The length of the resulting path, calculated as the sum of the individual segments, can be used to approximate the true length of the contour.

at a given start point $\boldsymbol{x}_S$ is represented by the sequence of directional changes it describes on the discrete image raster (Fig. 2.14).

Absolute chain code

For a closed contour of a region $\mathcal{R}$, described by the sequence of points $\boldsymbol{c}_\mathcal{R} = [\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots \boldsymbol{x}_{M-1}]$ with $\boldsymbol{x}_i = \langle u_i, v_i \rangle$, we create the elements of its chain code sequence $\boldsymbol{c}'_\mathcal{R} = [c'_0, c'_1, \ldots c'_{M-1}]$ by

$$c'_i = \textsc{Code}(\Delta u_i, \Delta v_i), \tag{2.1}$$

$$\text{where} \quad (\Delta u_i, \Delta v_i) = \begin{cases} (u_{i+1} - u_i, v_{i+1} - v_i) & \text{for } 0 \le i < M-1 \\ (u_0 - u_i, v_0 - v_i) & \text{for } i = M-1, \end{cases}$$

and $\textsc{Code}(\Delta u, \Delta v)$ being defined by the following table:[6]

Δu	1	1	0	−1	−1	−1	0	1
Δv	0	1	1	1	0	−1	−1	−1
$\textsc{Code}(\Delta u, \Delta v)$	0	1	2	3	4	5	6	7

[6] Assuming an 8-connected neighborhood.

4-connected contour) and the numeric value

$$\begin{aligned}\textsc{Val}(\boldsymbol{c}''_{\mathcal{R}}) &= c''_0 \cdot b^0 + c''_1 \cdot b^1 + \;\ldots\; + c''_{M-1} \cdot b^{M-1} \\ &= \sum_{i=0}^{M-1} c''_i \cdot b^i. \end{aligned} \tag{2.3}$$

Then the sequence $\boldsymbol{c}''_{\mathcal{R}}$ is shifted cyclically until the numeric value of the corresponding number reaches a maximum. We use the expression $\boldsymbol{c}''_{\mathcal{R}} \triangleright k$ to denote the sequence $\boldsymbol{c}''_{\mathcal{R}}$ being cyclically shifted by k positions to the right,[8] such as (for $k = 2$)

$$\begin{aligned}\boldsymbol{c}''_{\mathcal{R}} &= [\,0, 1, 3, 2, \ldots 5, 3, 7, 4\,] \\ \boldsymbol{c}''_{\mathcal{R}} \triangleright 2 &= [\,7, 4, 0, 1, 3, 2, \ldots 5, 3\,]\end{aligned}$$

and

$$k_{\max} = \arg \max_{0 \le k < M} \textsc{Val}(\boldsymbol{c}''_{\mathcal{R}} \triangleright k) \tag{2.4}$$

to denote the shift required to maximize the corresponding arithmetic value. The resulting code sequence or *shape number*,

$$\boldsymbol{s}_{\mathcal{R}} = \boldsymbol{c}''_{\mathcal{R}} \triangleright k_{\max}, \tag{2.5}$$

is *normalized* with respect to the starting point and can thus be directly compared element by element with other normalized code sequences. Since the function $\textsc{Val}()$ in Eqn. (2.3) produces values that are in general too large to be actually computed, in practice the relation

$$\textsc{Val}(\boldsymbol{c}''_1) > \textsc{Val}(\boldsymbol{c}''_2)$$

is determined by comparing the *lexicographic ordering* between the sequences $\boldsymbol{c}''_1$ and $\boldsymbol{c}''_2$ so that the arithmetic values need not be computed at all.

Unfortunately, comparisons based on chain codes are generally not very useful for determining the similarity between regions simply because rotations at arbitrary angles ($\neq 90°$) have too great of an impact (change) on a region's code. In addition, chain codes are not capable of handling changes in size (scaling) or other distortions. Section 2.4 presents a number of tools that are more appropriate in these types of cases.

---

[8] $(\boldsymbol{c}''_{\mathcal{R}} \triangleright k)[\,i\,] = \boldsymbol{c}''_{\mathcal{R}}[(i-k) \bmod M]$.

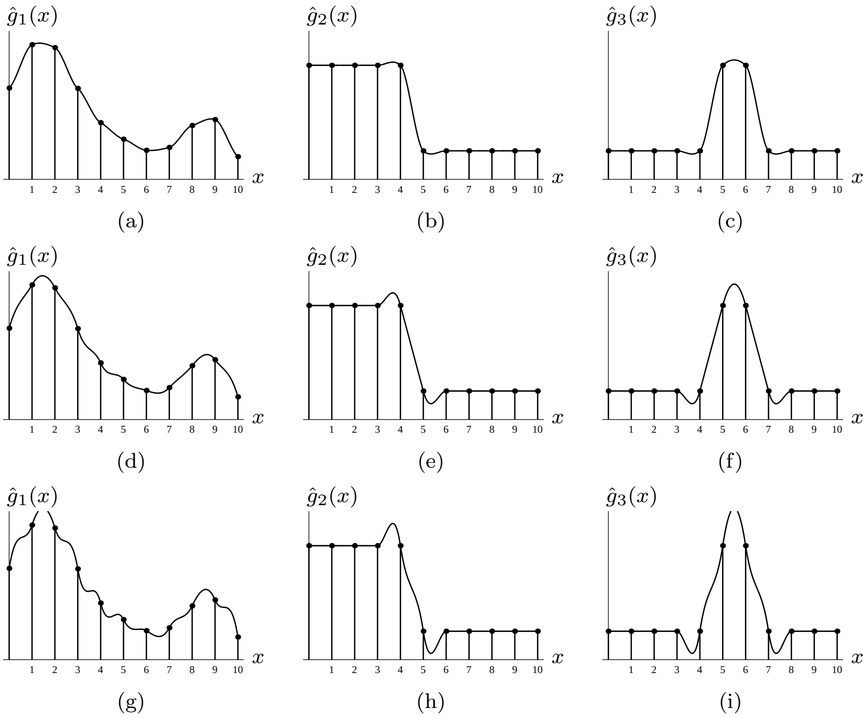


Figure 10.20 Cubic interpolation examples. Parameter a in Eqn. (10.57) controls the amount of signal overshoot or perceived sharpness: $a = 0.25$ (a–c), standard setting $a = 1$ (d–f), $a = 1.75$ (g–i). Notice in (d) the ripple effects incurred by interpolating with the standard settings in smooth signal regions.

control parameters (a, b) [54],[6]

$$w_{cs}(x, a, b) = \frac{1}{6} \cdot \begin{cases} (-6a - 9b + 12) \cdot |x|^3 \\ \quad + (6a + 12b - 18) \cdot |x|^2 - 2b + 6 & \text{for } 0 \le |x| < 1 \\ (-6a - b) \cdot |x|^3 + (30a + 6b) \cdot |x|^2 \\ \quad + (-48a - 12b) \cdot |x| + 24a + 8b & \text{for } 1 \le |x| < 2 \\ 0 & \text{for } |x| \ge 2\,. \end{cases} \tag{10.60}$$

Equation (10.60) describes a family of C1-continuous functions; i. e., the functions and their first derivatives are continuous everywhere and thus their trajectories exhibit no discontinuities or corners. For $b = 0$, the function $w_{cs}(x, a, b)$ specifies a one-parameter family of so-called *cardinal splines* equivalent to the

[6] In [54], the parameters a and b were originally named C and B, respectively, with $B \equiv b$ and $C \equiv a$.