Lecture Notes in Computational Vision and Biomechanics 2

Valentin E. Brimkov
Reneta P. Barneva *Editors*

Digital Geometry Algorithms

Theoretical Foundations and Applications to Computational Imaging

Springer

Valentin E. Brimkov • Reneta P. Barneva
Editors

Digital Geometry Algorithms

Theoretical Foundations and Applications to Computational Imaging

Springer

Chapter 1
Digital Geometry in Image-Based Metrology

Alfred M. Bruckstein

Abstract Interesting issues in digital geometry arise due to the need to perform accurate automated measurements on objects that are "seen through the eyes" of modern imaging devices. These devices are typically regular arrays of light sensors and they yield matrices of quantized probings of the objects being looked at. In this setting, the natural questions that may be posed are: how can we locate and recognize instances from classes of possible objects, and how precisely can we measure various geometric properties of the objects of interest, how accurately can we locate them given the limitations imposed upon us by the geometry of the sensor lattices and the quantization and noise omnipresent in the sensing process. Another interesting area of investigation is the design of classes of objects that enable optimal exploitation of the imaging device capabilities, in the sense of yielding the most accurate measurements possible.

1.1 Introduction

Scanned character recognition systems are by now working quite well, several companies emerged based on the need to do image based inspection for quality control in the semiconductor industry and, in general, automated visual inspection systems are by now widely used in many areas of manufacturing. In these important applications one often needs to perform precise geometric measurements based on images of various types of planar objects or shapes. Images of these shapes are provided by sensors with limited capabilities. These sensors are spatially arranged in regular planar arrays providing matrices of quantized pixel-values that need to be processed by automated metrology systems to extract information on the location, identity, size and orientation, texture and color of the objects being looked at. The geometry, spatial resolution and sensitivity of the sensor array are crucial factors in the measurement performances that are possible. When sensor arrays are regular planar grids, we have to deal with a wealth of issues involving geometry on the integer grid,

A.M. Bruckstein (✉)
Ollendorff Professor of Science, Computer Science Department, Technion, IIT, 32000 Haifa, Israel
e-mail: freddy@cs.technion.ac.il

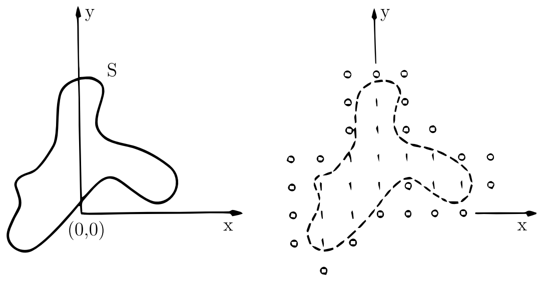


Fig. 1.1 Image digitization by point sampling on the unit grid $\mathbb{Z}^2$

hence digital geometry problems enter the picture in industrial metrology tasks in very fundamental ways.

1.2 The Digitization Model and the Metrology Tasks

We assume that planar shapes, the objects we are interested to locate, measure and recognize are binary (black on a white background) and live in the real plane, $\mathbb{R}^2$. Hence their full description can be given via an indicator function $\xi(x, y)$ which is 1 (black) if (x, y) is inside the shape and 0 (white) if (x, y) is in the background. The digitization process assumed will be point sampling on the integer grid, $\mathbb{Z}^2$, hence the result of digitization will be a discrete indicator function on the integer grid: a discrete binary image, or a zero/one matrix of picture elements, or pixels, see Fig. 1.1. The "generic problem" we deal with is: given the discretized shape, i.e.,

$$\xi_D(i, j) = \begin{cases} 1 & \text{if } \xi_D(i, j) = 1 \\ 0 & \text{if } \xi_D(i, j) = 0 \end{cases}$$

recover as much information as possible on the "pre-image", i.e., on the original binary shape that lives on the continuous real plane. The information on the pre-image shape that one needs might be its location and orientation, area, perimeter, etc. In order to solve the particular problem at hand we shall also need to exploit whatever prior information we may have on the continuous pre-images. This prior information sometimes defines the objects or shapes we digitize as members of parameterized sets of possible pre-images. For example, we might know that the shapes we are called upon to measure are circular with varying locations and sizes. In this case the parameter defining the particular object instance being analyzed from its digitization is a vector comprising three numbers: two coordinates pointing out the center of the disk and a positive number providing its radius. The digitized shape $\xi_D(i, j)$ then provides some information on the center and radius of the disk and we may ask how good an estimate can we get for these quantities given the data.

1.3 Self Similarity of Digital Lines

Digital lines result from point-sampling half-plane pre-images. More is known about the jagged boundaries obtained in this process topic than anyone can possibly know, but the basic facts are both simple and beautiful. Half-planes are not very interesting or practically useful objects, however they already pose the following metrology problem: given the digital image of a half-plane, locate it (i.e., its boundary line) as precisely as possible. Of course, we must ask ourselves whether and how our location estimation improves as we see more and more of the digitized boundary. We can think about the location estimation problem as a problem of determining the half-plane pre-images that satisfy all the constraints that the digitized image provides. Indeed every grid-point pixel that is 0 (white) will tell us that the half-plane does not cover that location while every black (1) pixel will indicate that the half-plane covers its position. It should come as no surprise that the boundary pixels, i.e., the locations where white pixels are neighboring black ones, carry all the information. The constraint that a certain location in the plane belongs, or does not belong to the half-plane that is being probed translates into a condition that the boundary line has a slope and intercept pair in a half-plane defined in the dual representation space (which is called in pattern recognition circles the Hough parameter plane). Therefore, as we collect progressively more data in the “image-plane” we have to intersect more and more half-planes in the Hough plane to get the so called “locale”, or the uncertainty region in parameter space where the boundary line parameters lie, see [12, 18, 25]. Looking at the grid geometry and analyzing the lines that correspond to grid-points in the dual plane one quickly realizes that only the boundary points contribute to setting the limits of the locale of interest, and a careful analysis reveals that, due to the regularity of the sampling grid, the locales are always polygons of at most four sides, see [12, 25]. Hence as more and more consecutive boundary points are added to the pool of information on the digitized half plane, we have to perform half-plane intersections with at most four sided polygonal locales to update them. Clearly the locales generally strictly decrease in size as the number of points increases, and we can get exact estimates on the uncertainty behavior as the jagged boundary is progressively revealed. This idea, combining the geometry of locales for digital straight lines with the process of successively performing the half-plane intersections for each new data point while walking along the jagged digitized boundary, led to the simplest, and only recursive $O(\text{length})$ algorithm for detecting straight edge segments. A complete description of this algorithm is the subject of the next section of this paper.

The jagged edges that result from discretizing half-planes have a beautiful, self-similar structure, intimately related to the continued fraction representation of the real number that defines the slope of their boundary line. It is clear that at various sampling resolutions the boundary maintains its jaggedness in a fractal manner, but here we mean a different type of self-similarity, inherent in the jagged boundaries at any given resolution! A wealth of interesting and beautiful properties that were described over many years of research on digital straight lines follow from a very simple unifying principle: invariance of the linear separability property under re-encoding with respect to regular grids embedded into the integer lattice. Not only does this principle help in re-discovering and proving in a very straightforward manner digital straight edge properties that were often arrived at and proved in sinuous ways, but it also points out all the self-similarity type properties that are possible, making nice connections to number-theoretic issues that arise in this context and the general linear group $GL(2, Z)$ that describes all integer lattice isomorphisms. Following [6], we next present the basic self-similarity results.

A digitized straight line is defined as the boundary of a linearly separable dichotomy of the set of points with integer coordinates, $\mathbb{Z}^2 = \{(i, j) | i, j \in \mathbb{Z}\}$, in the plane. The boundary points of the dichotomy induced by a line with slope m and intercept n, $y = mx + n$, are

$$L(m, n) = \{(i, h_i) | i \in \mathbb{Z}, h_i = \lfloor mi + n \rfloor\}.$$

Without loss of generality let us assume that $m > 0$, so that the sequence h_i is a nondecreasing sequence of integers. Associate to the set of boundary points $L(m, n)$ a string of two symbols, 0 and 1, coding the sequence of differences $h_{i+1} - h_i$, as follows

$$C(m, n) = \cdots C_{-2}C_{-1}C_0C_1C_2 \cdots = \prod_i C_j(m, n)$$

where

$$C_i(m, n) = \begin{cases} 0, & \text{if } h_{i+1} - h_i = 0, \\ 01^k, & \text{if } h_{i+1} - h_i = k, \end{cases}$$

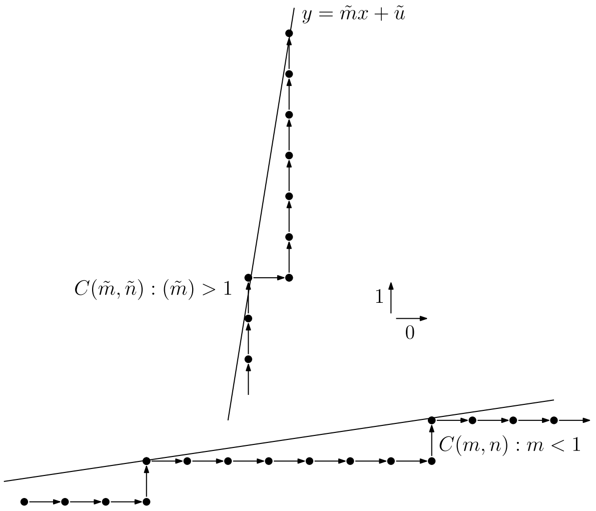
and 1^k means $1\ 1\ \cdots\ 1$ with k 1's. $C(m, n)$ is called the *chain-code* of the line $L(m, n)$. Note that the sequence $C(m, n)$ can be uniquely parsed into its components $C_i(m, n)$, since a separator must precede every 01 string and follow every run of 1's and each of the remaining 0's must be a single component, as well. Clearly, given some h_{i_0}-value and $C(m, n)$, the entire sequence h_i can be recovered. The graphical meaning of the chain-code associated to $L(m, n)$ is depicted in Fig. 1.2.

The set of points $L(m, n)$ uniquely determines the slope of the line m. Indeed, $L(m, n) \equiv L(m', n')$ implies $m = m'$, since otherwise the vertical distance between $y = mx + n$ and $y = m'x + n'$ would become unbounded at $x \to \pm\infty$, and their h_i sequences would differ starting at some large enough i. Furthermore, if m is irrational we have, by a classical result, that the vertical intercepts of $y = mx + n$ modulo 1 are dense in [0, 1]. For every $\varepsilon > 0$ there exist i_0 and j_0 such that

$$mi_0 + n - \lfloor mi_0 + n \rfloor < \varepsilon,$$

$$mj_0 + n - \lfloor mj_0 + n \rfloor > 1 - \varepsilon,$$

and changing n by ε would result in a change in $L(m, n)$. Therefore for irrational m, $L(m, n)$ uniquely determines both m and n. If m is rational there exist only a finite set of distinct vertical intercepts of $y = mx + n$ modulo 1, therefore n is determined only up to an interval and the length of the worst interval of uncertainly for n depends on the minimal p/q representation of m. This also proves that the chain-code $C(m, n)$ determines m uniquely, and if m is irrational, n is also determined by it modulo 1, since we clearly have $C(m, n) \equiv C(m, n + 1)$.



$y = \tilde{m}x + \tilde{u}$
$C(\tilde{m}, \tilde{n}) : (\tilde{m}) > 1$
1
0
$C(m, n) : m < 1$

Fig. 1.2 Chain-codes of $L(m, n)$ for $m < 1$ and $\tilde{m} > 1$

From the definition of chain-codes $C(m, n)$ we obtain several immediate and basic properties a sequences of zeros and ones must have in order to be the chain-code of a straight edge. In the case of $m < 1$, the difference

$$h_{i+1} - h_i = \lfloor m(i + 1) + n \rfloor - \lfloor mi + n \rfloor \tag{1.1}$$

can only be either 0 or 1. In this case the chain-code of a digitized line has runs of 0's separated by single 1's, and the 0's occur in runs with length determined by the number on integer coordinates that fall within the intervals determined on the x-axis by the points defined by

$$mx_i + n = i \in \mathbb{Z}, \quad \text{i.e.,} \quad x_i = \frac{i}{m} + \frac{n}{m}. \tag{1.2}$$

The intervals $[x_i, x_{i+1})$ have a constant length of $1/m$ and therefore the number of integer coordinates covered can be (see Fig. 1.3a) either $\rho_i = \lfloor 1/m \rfloor$ or $\rho_i = \lfloor 1/m \rfloor + 1$. Therefore, if $m < 1$, $C(m, n)$ is of the form

$$C(m, n) = \cdots 10^{\rho_1} 10^{\rho_2} 10^{\rho_3} 1 \cdots \tag{1.3}$$

where $\rho \in \{\lfloor 1/m \rfloor, \lfloor 1/m \rfloor + 1\}$. For the case $m > 1$, the difference $hi + 1 - h_i = \lfloor m(i + 1) + n \rfloor - \lfloor mi + n \rfloor$ is always greater than 1, and therefore the chain-code $C(m, n)$ has runs of 1's separated by single 0's. Since $\lfloor m + mi + n \rfloor - \lfloor mi + n \rfloor$ equals the number of integer coordinates between the values $m(i+1)+n$ and $mi+n$ the run of 1's has length determined by the number of integral values in consecutive intervals of length m, see Fig. 1.3b. This shows that the run-lengths ρ_i in this case, will be either $\rho = \lfloor m \rfloor$ or $\lfloor m \rfloor + 1$. Therefore if $m > 1$, the chain-code $C(m, n)$ has the form

$$C(m, n) = \cdots 01^{\rho_1} 01^{\rho_2} 01^{\rho_3} 0 \cdots \tag{1.4}$$

with $\rho \in \{\lfloor m \rfloor, \lfloor m \rfloor + 1\}$.

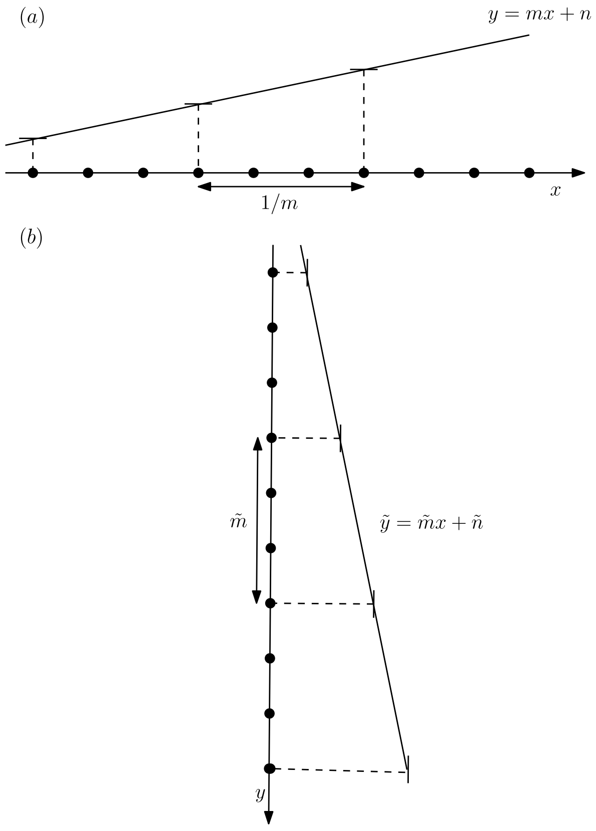


Fig. 1.3 Basic properties of chain-codes

The question that immediately arises is the following: is there any order or *pattern* in the appearance of the two values for the run length of the symbols 0 or 1, i.e., in the sequences $\{\rho_i\}$ that arise from "chain coding" digitized straight lines? The classical results on digital straight edges are focused on "uniformity properties" of the appearance of the separator symbol in the chain-codes sequences, see [14, 20]. We here briefly present a very high level and general uniformity result via self-similarity, as was first defined in [6].

Suppose we are given the chain-code of a digitized straight boundary $C(m,n)$. We know that $C(m,n)$ is a sequence composed of two symbols, 0 and 1, and that it looks either like (1.3) or (1.4), thus it has the general form

$$\cdots \Delta \square^{\rho_i} \Delta \square^{\rho_{i+1}} \Delta \square^{\rho_{i+2}} \Delta \cdots \tag{1.5}$$

where $\rho_i \in \{p, p+1\}$, $p \in \mathbb{Z}$, and Δ, $\square$ stand for either 0, 1 or 1, 0, respectively.

We can define several transformation rules on two symbol, or $\Delta/\square$, sequences of the type (1.5), transformations that yield new $\Delta/\square$ sequences.

RULE X. Interchange the symbols Δ and $\square$ (i.e., $\Delta \to \square$ and $\square \to \Delta$).

Application of **X** to a chain-code $C(m,n)$ yields a new sequence of symbols, with 0's replacing the 1's and 1's replacing the 0's of the original sequence.

RULE S. Replace every $\Box\varDelta$ sequence by $\varDelta$.

Application of the **S**-transformation to a chain-code of the form (1.5) yields a sequence of the same type with the run length ρ_i replaced by $\rho_i - 1$. Applying Rule **S**, p times yields the next transformation rule.

RULE S^p. Replace $\Box^p\varDelta$ by $\varDelta$ and $\Box^{p+1}\varDelta$ by $\Box\varDelta$.

Notice that, in contrast to the transformation rules **X** and **S**, this rule depends on the $\{\rho_i\}$ sequence, i.e., it is adapted to the given pattern (1.5). Indeed we can apply the **S**-transformation successively at most p times where p is the minimal value of ρ_i's. After that, we need to do an **X** transformation in order to bring the sequence of symbols to the form (1.5).

RULE R. Replace $\Box^p\varDelta$ by $\varDelta$ and $\Box^{p+1}\varDelta$ by $\Box$.

We may view the action of **R** as a result of applying **S**^{p} first, then replacing $\Box\varDelta$ by $\Box$. This rule is also adapted to the sequence on which it operates.

The next transformation rule is somewhat different, since it replaces symbols in a way that depends on the neighborhood or the "context".

RULE T. Replace $\Box\varDelta$ by $\Box$ and the $\Box$'s followed by a $\Box$ by $\Box\varDelta$.

Application of rule **T** has the effect of putting a $\varDelta$ between every consecutive pair of $\Box$'s and removing all the $\varDelta$'s appearing in the original sequence. For example the sequence

$$\cdots\Box\varDelta\Box\Box\Box\Box\varDelta\Box\Box\Box\varDelta\Box\Box\Box\varDelta\Box\cdots$$

will be mapped under **T**, to

$$\cdots\Box\Box\varDelta\Box\varDelta\Box\varDelta\Box\Box\varDelta\Box\varDelta\Box\Box\varDelta\Box\varDelta\Box\Box\cdots$$

Up to this point the transformation rules were completely specified by rather simple local symbol replacement rules. The next two classes of transformation rules, require the setting of an initial position and a bilateral parsing for the generation of the transformed sequences.

V-RULES. Given the sequence of $\varDelta\Box$, choose a $\varDelta$ symbol as an initial position, then to the right and to the left of the chosen $\varDelta$ delete batches of $Q-1$ consecutive $\varDelta$'s.

This transformation has the effect of joining together (from the starting position) Q consecutive $\Box$-runs. The sequence

$$\varDelta\Box^{\rho_{i-Q}}\cdots\varDelta\Box^{\rho_{i-1}}\underset{\uparrow}{\varDelta}\Box^{\rho_i}\varDelta\Box^{\rho_{i-1}}\varDelta\cdots\Box^{\rho_{i+Q-1}}\varDelta$$

will be mapped to

$$\cdots\varDelta\Box^{\rho_{i-Q}+\cdots+\rho_{i-1}}\varDelta\Box^{\rho_i+\rho_{i+1}+\cdots+\rho_{i+Q-1}}\varDelta\cdots$$

if the $\varDelta$ preceding $\Box^{\rho_i}$ is chosen as the initial position. Therefore if a $\varDelta/\Box$-chain-code sequence of type (1.5) is specified by the $\Box$-run length sequence $\{\rho_i\}_{i\in N}$ a **V**-transformation as defined above will produce a sequence of type (1.5) specified by $\{\rho_{i_0+nQ}+\rho_{i_0+nQ+1}+\cdots+\rho_{i_0+(n+1)Q-1}\}_{n\in N}$ for a given i_0 and a given integer $Q\geq 1$.

H-RULES. Given the sequence of $\varDelta\Box$ symbols, choose a starting point between two consecutive symbols and parse the sequence to the right and left of the starting point, counting the number of $\Box$'s seen. After seeing P $\Box$'s, replace the subsequence by one $\Box$ followed by the number of $\varDelta$'s encountered while accumulating the P $\Box$'s. In counting the $\varDelta$'s encountered, apply the following rules: (1) when parsing to the right: if the P-th $\Box$ symbol is followed by a $\varDelta$ count this $\varDelta$ as well and start accumulating the next batch of P symbols after it and (2) when parsing to the left: if the P-th $\Box$ symbol is preceded by $\varDelta$ do not count this $\varDelta$ and start accumulating the next batch of P $\Box$'s immediately.

As an example, consider applying an **H**-transformation to the sequence below, with the indicated initial position,

$$\cdots\Box\varDelta\Box\Box\Box\varDelta\Box\Box\Box\varDelta\uparrow\Box\Box\Box\Box\varDelta\Box\Box\Box\varDelta\Box\Box\Box\varDelta\Box\Box\Box\varDelta\Box\cdots$$

and parameter $P=7$. We obtain the parsing

$$\cdots\uparrow\Box\varDelta\Box\Box\Box\varDelta\Box\Box\Box\varDelta\uparrow\Box\Box\Box\Box\varDelta\Box\Box\Box\varDelta\uparrow\Box\Box\Box\varDelta\Box\Box\Box\varDelta\Box\uparrow\cdots$$

that yields the output

$$\cdots\uparrow\Box\varDelta\varDelta\varDelta\uparrow\Box\varDelta\varDelta\uparrow\Box\varDelta\varDelta\uparrow\cdots$$

The same initial conditions with parameter $P=3$ provide the parsing

$$\cdots\Box\varDelta\uparrow\Box\Box\Box\varDelta\uparrow\Box\Box\Box|\varDelta\uparrow\Box\Box\Box\uparrow\Box\varDelta\Box\Box\uparrow\Box\varDelta\Box\Box\uparrow\Box\varDelta\Box\Box\uparrow\Box\varDelta\Box\cdots$$

and an output sequence

$$\cdots\uparrow\Box\varDelta\uparrow\Box\varDelta\uparrow\Box\uparrow\Box\varDelta\uparrow\Box\varDelta\uparrow\Box\varDelta\uparrow\cdots$$

So far we have defined seven rules for transforming $\varDelta/\Box$ sequences into new $\varDelta/\Box$ sequences. The first five of them are uniquely specified in terms of local string replacement rules, the last two being classes of transformations that require the choice of an initial positions for parsing and are further specified by an arbitrarily chosen integer (Q or P). The main self similarity results are, [6]:

The Self-similarity Theorem

1. *Given a $\varDelta\Box$ sequence of type* (1.5), *the new sequence produced by applying to it any of the transformations* **X**, **S**, $\mathbf{S}^p$, **R**, *or* **T**, *is the chain-code of a digitized straight line if and only if the original sequence was the chain-code of a digitized straight line.*
2. *If a $\varDelta\Box$ sequence is the chain-code of a digitized straight line, then the sequences obtained from it by applying any transformation according to the* **H**-*rules, or* **V**-*rules, are also chain-codes of digitized straight lines.*

Note that, for the **X**-, **S**-, $\mathbf{S}^p$-, **R**-, or **T**-transformation rules we have stronger claims than for the classes of **H**- and **R**-rules. The reason for this will become obvious from the proof. The digital line properties stated above are *self-similarity* results since what we have is that a given chain-code pattern generates, under repeated applications of various transformation rules, new patterns in the same class: chain-codes of digitized straight lines.

Proof We argue that the chain-code transformations defined above are simply re-encodings of digitized straight lines on regular lattices of points, embedded into the integer lattice $\mathbb{Z}^2$. This observation, combined with the fact that the embedded lattices are generated by affine coordinate transformations, readily yield the results claimed. Indeed, choose any two linearly independent basis vectors B_1 and B_2 with integer entries and a lattice point (i_0, j_0) for the origin Ω_0. Define a regular embedded lattice of points as follows

$$\mathbf{E}^2 = \left\{(i_0, j_0) + iB_1 + jB_2 | (i, j) \in \mathbb{Z}^2\right\}.$$

A given straight line $y = mx + n$ defines a dichotomy of the points of $\mathbb{Z}^2$, but also of the points of $\mathbf{E}^2 \subset \mathbb{Z}^2$! If B_1 and B_2 are basis vectors, there exists an affine transformation that maps lattice $\mathbf{E}^2$ the embedding back into $\mathbb{Z}^2$, i.e., the point $(i_0, j_0) + iB_1 + jB_2 \in \mathbf{E}^2$ into $(i, j) \in \mathbb{Z}^2$, and the *same* transformation maps the line $y = mx + n$ into some new line $Y = MX + N$, on the transformed plane. The points $(i_0, j_0) + iB_1 + jB_2$ from the original (x, y)-plane map into (i, j), hence the transformation from (X, Y) into (x, y) is

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} i_0 \\ j_0 \end{pmatrix} + \left[B_1^T B_2^T\right] \begin{pmatrix} X \\ Y \end{pmatrix}$$

and therefore the inverse mapping from (x, y) to (X, Y) is

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \left[B_1^T B_2^T\right]^{-1} \left[\begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} i_0 \\ j_0 \end{pmatrix}\right] = \mathbf{M} \begin{pmatrix} i_0 \\ j_0 \end{pmatrix}. \tag{1.6}$$

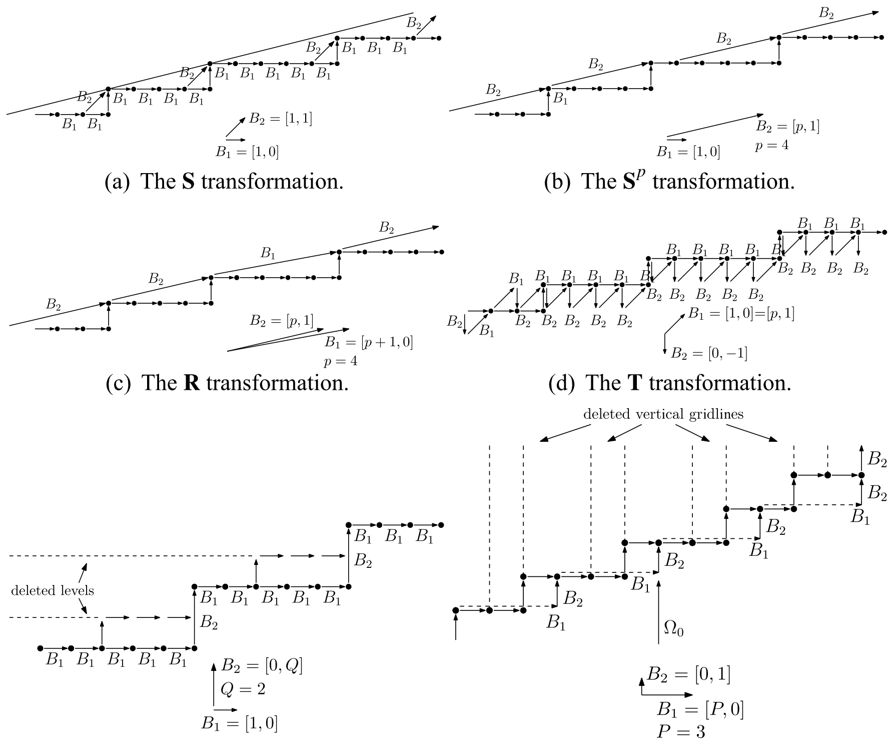
From these transformations the mapping of the line parameters (m, n) into (M, N) can also be readily obtained in terms of $B_1 B_2$ and Ω_0.

After performing the transformation (1.6) the line $Y = MX + N$ can be chain-coded with respect to the lattice $\mathbb{Z}^2$ (which is now the image of $\mathbf{E}^2$) and the resulting chain-code will somehow be related to the chain-code of $y = mx + n$ defined on the original grid $\mathbb{Z}^2$. The key observation, proving the results stated, is that the transformations introduced in the previous section represent straightforward re-encodings of digitized lines with respect to suitably chosen embedded lattices $\mathbf{E}^2$. The choices of basis vectors that lead to each of the sequence transformations we are concentrating on are shown in Fig. 1.4 and are analyzed in detail below:

1. The $\mathbf{X}$ transformation rule, the interchange of Δ and $\square$ symbols, is clearly accomplished by the coordinate-change mapping that takes (i, j) into (j, i). Here $B_1 = [0, 1]$ and $B_2 = [1, 0]$ and we have that $y = mx + n$ maps into $Y = (1/m)X - (n/m)$ under the transformation matrix $\mathbf{M}_X = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$.
2. The $\mathbf{S}$-rule which reduces every integer of the $\{\rho_i\}$ sequence by 1 is induced by the mapping that considers a $\square$ step as a step in the $B_1 = [1, 0]$ direction, but a combined $\square\Delta$-step as the unit step in the $B_2 = [1, 1]$-direction (see Fig. 1.4a). Therefore, the $\mathbf{S}$-transformation matrix is

$$\mathbf{M}_S = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

and $y = mx + n$ maps into $Y = Xm/(1 - m) + n/(1 - m)$.



(a) The **S** transformation. (b) The $\mathbf{S}^p$ transformation.

(c) The **R** transformation. (d) The **T** transformation.

(e) The **V** transformation. (f) The **H** transformation.

Fig. 1.4 The **S**, $\mathbf{S}^p$, **R**, **T**, **V** and **H** transformations

3. The adaptive $\mathbf{S}^p$-transformation rule which replaces $\square^p \varDelta$ by $\varDelta$, and $\square^{p+1}\varDelta$ by $\square\varDelta$, corresponds to choosing $B_1 = [1,0]$ and $B_2 = [p,1]$ (see Fig. 1.4b). The transformation matrix is

$$\mathbf{M}_{S^p} = \begin{bmatrix} 1 & p \\ 0 & 1 \end{bmatrix}^{-1} = \left(\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \right)^{-1}$$
$$= \begin{bmatrix} 1 & -p \\ 0 & 1 \end{bmatrix}.$$

The line $y = mx + n$ is transformed into $Y = Xm/(1-pm) + n/(1-pm)$. Note that, if $m < 1$, $p = \lfloor 1/m \rfloor$ and we denote the fractional part of $1/m$ by $\alpha(= 1/m - \lfloor 1/m \rfloor)$, we have $m/(1-mp) = 1/\alpha > 1$. This shows that one $\mathbf{S}^p$-transformation, that is adapted to the run-length of the $\square$-symbols replaces the slope m with $(1/m - \lfloor 1/m \rfloor)^{-1}$. Therefore, repeated application of this adapted transformation followed by an X-transformation will produce a sequence of slopes recursively given by $m_k = 1/m_{k-1} - \lfloor 1/m_{k-1} \rfloor$, $m_0 = m$. Hence, the sequence of adapted "exponents" of the corresponding $\mathbf{S}^p$-transformations $p_k =$

$\lfloor 1/m_k \rfloor$, is the sequence of integers of the continued fraction representation of m_0,

$$m_0 = \cfrac{1}{p_0 + \cfrac{1}{p_1 + \cfrac{1}{p_2 + \frac{1}{\cdots}}}}$$

4. The transformation rule **R** maps $\square^p \varDelta$ into $\varDelta$ and $\square^{p+1} \varDelta$ into $\square$. Therefore $B_1 = [p+1, 1]$ and $B_2 = [p, 1]$ (Fig. 1.4c). The transformation matrix is

$$\mathbf{M}_R = \begin{bmatrix} p+1 & p \\ 1 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & -p \\ -1 & p+1 \end{bmatrix}$$

and an original line $y = mx + n$ is mapped into $Y = X[(p+1)m - 1]/(1 - pm) + m/(1 - pm)$. Note here that in terms of $\alpha = 1/m - \lfloor 1/m \rfloor$ the new slope is $(1-\alpha)/\alpha$, when $m < 1$.

5. The last of this class of transformations, Rule **T**, replaces $\square\varDelta$ by $\square$'s, and $\square$'s followed by a $\square$ by $\square\varDelta$'s. We may view this transformation as a sequence of two maps: the first one replacing $\square^{p+1}\varDelta$ by $\square$, and $\square^p \varDelta$ by $\varDelta$ by the adapted rule **R**, the second replacing $\square$ by $\square\varDelta\square\varDelta\cdots\varDelta\square$ with $(p+1)\square$'s, and $\varDelta$ by $\square\varDelta\square\varDelta\cdots\varDelta\square$ with $p\square$'s. This would imply that we first do an **R**-transformation via the matrix

$$\mathbf{M}_{RT} = \begin{bmatrix} p+1 & p \\ p & p-1 \end{bmatrix}.$$

Concatenating the two transformations we obtain

$$\mathbf{M}_T = \mathbf{M}_{RT}\mathbf{M}_R = \begin{bmatrix} 1 & 0 \\ 1 & -1 \end{bmatrix},$$

which is not surprising. Indeed, $\square\varDelta$ is mapped by $B_1 = [1\ 1]$ into one $\square$ step, but a $\square$ followed by another $\square$ will have to be mapped into a sequence of two steps, B_2B_1, the first one being $B_2 = [0, -1]$ (see Fig. 1.4d). We readily see from the $\mathbf{M}_T$ transformation that $y = mx + n$ maps into $Y = (1-m)X - n$. Therefore the slopes of the two lines add to 1. Indeed, "summing up" the two sequences in the sense of placing a $\varDelta$ whenever there exists a $\varDelta$ in either the original, or the **T**-transformed chain-code, we get the sequence $\cdots\square\varDelta\square\varDelta\square\varDelta\square\cdots$, which represents the lines of the type $y = x + n$.

Up to this point, all the transformation matrices, whether adapted to the chain-code parameter p or not, were matrices with integer entries and had the property that $\det(\mathbf{M}) = \pm 1$. This implied that the matrices had inverses with integer entries and, as a consequence, the embedded lattice $\mathbf{E}^2$ was simply a "reorganization" or "relabeling" of the entire integer lattice $\mathbb{Z}^2$. In mathematical terms, unimodular lattice transformations are isomorphisms of the two dimensional lattice. The 2×2 integer matrices with determinant ± 1 (called unimodular matrices) form a well known group called $GL(2, \mathbb{Z})$ and this group is finitely generated by the matrices $\left[\begin{smallmatrix}0 & 1\\ +1 & 0\end{smallmatrix}\right]\left[\begin{smallmatrix}1 & 1\\ 0 & 1\end{smallmatrix}\right]\left[\begin{smallmatrix}-1 & 0\\ 0 & 1\end{smallmatrix}\right]$. For all such transformations (that are invertible within $GL(2,\mathbb{Z})$) the corresponding chain-code modification rules will yield chain-codes of linearly separable dichotomies, simply because the transformed line $Y = MX + N$ induces a linearly separable dichotomy of $\mathbf{E}^2$. Therefore the self-similarity results may be regarded as two different ways of stating that the points of the lattice $\mathbb{Z}^2$ are linearly separated by a given line $y = mx + n$. The first class of results presented above becomes obvious in this setting. Furthermore, from the fact that the group $GL(2,\mathbb{Z})$ is finitely generated, it follows that we have countably many sequence transformations, having the property that they yield chain-codes of straight lines if and only if the original chain-code is a digitized straight line, and they are expressible as products of sequences of basic transformations of the type **X**, **S**, and, say **T** (or one other transformation).

The situation is somewhat different for the remaining classes of transformation rules, the **V** and **H**-rules.

6. In the embedded lattice setting it is easy to see that a **V**-rule implies choosing some origin point Ω_0 and basis vectors of the form $B_1 = [1, 0]$, $B_2 = [0, Q]$ (see Fig. 1.4e). In this case, the set $\mathbf{E}^2$ is properly contained in $\mathbb{Z}^2$, i.e., $\mathbf{E}^2 \subset \mathbb{Z}^2$ and the mapping of Eq. (1.6) has fractional entries. Since **V**-rules imply a decimation of the horizontal grid lines, the fact that a chain-code of a digitized line provides a new digitized line, is obvious. However, due to the proper embedding of $\mathbf{E}^2$ into $\mathbb{Z}^2$ these results are not "if and only if" results any more. Indeed, we could start with a sequence like

$$\cdots \varDelta\Box\varDelta\Box^p\varDelta\Box\varDelta\Box^p\varDelta\Box\varDelta\cdots$$

and any **V**-transformation with $Q = 2$ will provide the transformed sequence

$$\varDelta\Box^{p+1}\varDelta\Box^{p+1}\varDelta\Box^{p+1}\cdots$$

This sequence is obviously a digitized straight line while the original one is obviously not, for any $p > 2$. Hence, the proper embedding of $\mathbf{E}^2$ in $\mathbb{Z}^2$ implies that digital lines, but not only digital lines, map into digital lines. Note also that for a **V**-rule determined by an integer Q, the line $y = mx + n$ is mapped into a line with slope m/Q.
7. The **H**-rules defined imply choosing some origin point Ω_0 and decimating this time the vertical grid lines, by removing batches of P consecutive vertical lines. The basis vectors are in this case $B_1 = [P, 0]$ and $B_2 = [0, 1]$ (see Fig. 1.4f). In this case too $\mathbf{E}^2$ is properly contained in $\mathbb{Z}^2$ and again the mapping (1.6) has fractional entries, the determinant of $[B_1 B_2]$ being P. Clearly applying an **H**-rule to the chain-code of a digital straight line will yield the chain-code of a new line with slope mP, however this too is only an one-directional implication, not an "if and only if" result.

   We can clearly combine **V** and **H**-transformations to yield new and more complicated sequence mapping rules. For example, applying a **V** and an **H**-transformation with the same parameter, i.e., $P = Q$ is equivalent to re-encoding the digitized straight line at a reduced resolution. Note that if the line passes through the origin, i.e., we have $y = mx$, and we apply a chain-code transformation rule that has the effect of reducing resolution with any $P = Q$, we must always obtain exactly the same chain-code since the new slope will be the same, $(m \cdot P)/Q = m$. This is a rather nice invariance property of chain-codes of lines passing through the origin and it is not entirely obvious in a nongeometric context. $\square$

The fact that a digitized straight line has the above discussed series of invariance, or "self-similarity" properties, has many immediate consequences.

The result that an **R**-transformation on a sequence of symbols yields the chain-code of digitized straight line if and only if the original sequence was itself a straight line, constrains the run patterns of the symbol occurring in runs. We may have runs of equal-length runs but one of the run-length must always occur in isolation (otherwise the **R**-transformation would yield a sequence in which both symbols occur in runs longer than 1). Furthermore, this must also be the case at further levels of run-length encoding of the run-length sequences.

Consider the chain-code of a digitized straight line $C(m, n)$. Performing an **S**-transformation on it we get a new chain-code with the property that every symbol in the new sequence of symbols corresponds to, or "contains", exactly one $\square$ symbol from the original chain-code. Therefore parsing the **S**-transformed code into subsequences of equal length is equivalent to performing an **H**-transformation on the original chain-code. This shows that in any two equal length subsequences of a straight line chain-code the number of Δ's (and consequently also $\square$'s) may differ by at most 1. This property shows that self-similarity is in fact a description of uniformity in the distribution of the separator symbols (Δ) in the chain-code sequence. Indeed the slope of the line m sets the density of these symbols, and the digitization process ensures that this density will be achieved with a distortion as uniform as possible. This interpretation of digital straight lines, as well as their connections to Euclid's division algorithm (via continued fraction representations) and to a wealth of other areas as diverse as music [16], billiard trajectories [4], abstract sequence analysis [3], combinatorics on words [24], and quasicrystals [30, 32], make this area of research essentially inexhaustible.

From among many interesting consequences of the self-similarity results we have chosen to mention the above two properties because such results have been obtained before, using different proofs, in the context of testing whether a finite sequence of two symbols could be the chain-code of a digitized straight line segment, see, e.g., [20].

What we have in fact shown is that one can obtain chain-code transformation rules that characterize linearity, via the group $GL(2, \mathbb{Z})$ of unimodular lattice transformations. As a consequence we can readily "produce" countably many interesting and new self-similarity properties of linear chain-code patterns.

Using the properties of digital straight lines, we can not only solve the somewhat theoretical issue of locating a half-plane object of infinite extent but we can also address some very practical issues like measuring the perimeters of general planar shapes from their versions digitized on regular grids of pixels. Indeed, analyzing the properties of digitized lines made possible the rational design of some very simple and accurate perimeter estimators, based on classifications of the boundary pixels into different classes according to the jaggedness of their neighborhoods. Building upon earlier work of Proffit and Rosen [31], Koplowitz and Bruckstein proposed a general methodology for the design of simple and accurate perimeter estimation algorithms that are based on minimizing the maximum error that would be incurred for infinitely long digitized straight edges over all orientations [21]. This methodology enables predictions of the expected performance for shapes having arbitrary, but bounded curvature boundaries.

1.4 Digital Straight Segments: Their Characterization and Recognition

The previous section focused on the properties of digitized half planes of infinite extent but we often discretize polygonal shapes that have finite length straight segments as boundaries. Such shapes which will yield finite sequences of chain-code symbols that will be called Digitally Straight Segments (DSS's). In general it is of interest to describe a general discretized boundary by partitioning it into a sequence of digital straight segments, effectively producing a "polygonal pre-image" of the boundary on which a variety of measurements can be performed. In order to describe a very efficient and easy to grasp algorithm for partitioning a chain-code sequence into discrete straight portions we need to formalize the Hough-domain, or dual-space "pre-image" concept. It is well known that a point in the plane defines a pencil of lines that pass through it, i.e., $(x_0, y_0) \in \mathbb{R}^2$ corresponds to the lines $y = mx + n$ that obey $y_0 = mx_0 + n$, and this is an equation defining a line in the Hough-space where the coordinates are (m, n). When a straight Black/White boundary is digitized by point sampling, the grid points that are on the border between the black region ($\xi_D(i, j) = 1$) and the white one ($\xi_D(k, l) = 0$) correspond to lines in the Hough-plane that delineate the (m, n) domain to which the straight line of the boundary belongs. Considering Fig. 1.5 we have that the lines corresponding to a vertical border of the discretized line, i.e., the pixels (i, j), $(i, j + 1)$ for which: $\{\xi_D(i, j) = 1, \xi_D(i, j + 1) = 0\}$ are the parallel lines in the (m, n)-plane defined by

$$\begin{cases} (i + j) = mi + n & \Rightarrow \quad n = -im + (j + 1) \\ j \qquad\;\; = mi + n & \Rightarrow \quad n = -im + j \end{cases}$$

Similarly, the next vertical pair of border pixels for which $\{\xi_D(i + 1, j) = 1, \xi_D(i + 1, j + 1) = 0\}$ correspond to the (m, n)-plane lines given by the parallel lines:

$$\begin{cases} (j + 1) = m(i + 1) + n & \Rightarrow \quad n = -(i + 1)m + (j + 1) \\ j \qquad\quad\;\; = m(i + 1) + n & \Rightarrow \quad n = -(i + 1)m + j \end{cases}$$

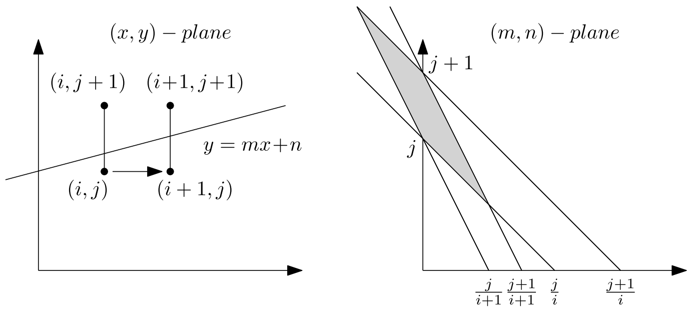


Fig. 1.5 Chain-code step and the corresponding Hough-plane geometry

Since clearly the pre-image border line intersects the segments $[(i, j), (i, j+1)]$, and $[(i+1, j), (i+1, j+1)]$ the (m, n) parameters of the pre-image line belong to the intersection of the bands defined by the two pairs of parallels corresponding to the border pixels. The "locale" for the pre-image has been therefore restricted to a parallelogram by the discrete data that corresponds to one step of the digitized boundary's chain-code. Since this process can be repeated for each chain-code symbol in the description of the discretized boundary, we have that each new chain-code symbol requires the intersection of the previously delineated "locale" for the "pre-image line" with a pair of parallel lines in the (m, n)-plane. We therefore have the following recursive algorithm for determining the "pre-image line locale", which is also, in fact, a process for determining digital straight segment portions of a chain-code:

Digital Straight Segment Detection Process

1. For each symbol of the 4-directional chain-code intersect the uncertainty region or locale in the (m, n)-plane with the corresponding band in the Hough(dual)-plane.
2. While the result is not empty there exists a linear-pre-image for the chain-coded portion of the boundary, hence the chain-code portion is a digital straight segment.

A careful analysis of how the intersections of chain-code bands look in the Hough plane reveals a miraculous fact: the locales are always regions defined by at most 4 boundary lines. This is a marvelous result due to Leo Dorst [12], which was given a simple proof by Douglas McIlroy in [25]. The result is indeed marvelous because it means that the recursive intersection process that the above described algorithm for detecting digital straight segment will only take $O(1)$-time, requiring the intersection of a four-sided polygon with two parallel lines. And the situation is even better: the points defining the locale polygon have rational coordinates hence the DSS detection process involves updating 8 integers for each chain-code symbol parsed, see [23]. Therefore we have a beautifully simple $O(1)/(\text{chain-code-step})$

recursive digital straight segment detection process. Furthermore, starting the algorithm on an arbitrary chain-coded border we can very efficiently parse it into DSS-segments. Hence, given a shape digitized on $\mathbb{Z}^2$, we can determine a polygonal approximation for the shape by parsing the digitized boundary into DSS segments, and for each of these we have position and slope estimates readily provided by their Hough-plane [(m, n)-plane] "locales". In particular the recursive $O(1)$-per boundary pixel algorithm for detecting digital straightness described above, due to Lindenbaum and Bruckstein [23], enables parsing general, curved object boundaries into digitally straight segments in order to estimate the pre-image object's perimeter as a sum of the lengths of the line-segments so-detected. In terms of the methodology discussed in [21] this algorithm yields zero error for digital straight edges of infinite extent at all orientations, and hence should be the best perimeter estimator ever, if the criterion would be performance on straight boundaries.

1.5 Digital Disks, Convex and Star-Shaped Objects

From the realm of half-plane objects and digital straight lines we could move to either infinite extent regions that have more complex boundaries (say parabolas, hyperbolas or some periodic functions along a principal direction) or to the analysis of finite extent objects like polygons, disks and other interesting shapes. Some work has indeed been done on detecting polygonal preimages from their digitized versions, and, as we have seen, a good algorithm for parsing a jagged boundary into digital straight segments turns out to be a crucial ingredient in solving various issues regarding the metrology of such objects.

Suppose next that we have the prior information that the objects discretized are disks of various locations and sizes. Then the metrology question arising naturally is: how precisely can we determine the location of a disk and its radius. Considering the digitization by point sampling, as discussed above, given a digitized image of black and white pixels, we know that if a certain point in the plane is the center of a disk of unknown radius, this point will necessarily be closer to all black grid points than to any white grid point. Hence the locus of all possible points in the plane closer to all black points than to any white points is the locale of possible disk centers, and its size will quantify our uncertainty in locating the object in the preimage plane. It is interesting to note that this locale can be found without knowledge on the radius, which will still need to be estimated. It turns out that the locale as defined above is a well-known concept in computational geometry, and it is known that it is a convex region in the plane. Efrat and Gotsman have done a careful analysis of the problem and produced an $O(R \log R)$ algorithm to determine the locale, where R is the radius of the disk. We refer the interested reader to the paper [13] for details. Note again that the locale we are talking about is independent of the radius parameter. Had we prior knowledge on the exact radius, the location of the disk center could be determined by intersecting all disks of radius R around the black grid points with all the complements of disks or radius R around the white (uncovered) grid points. The resulting intersection locale is generally not a convex shape, due to the precise knowledge of the radius.

For general convex shapes the question of determining the location, area and perimeter cannot be addressed in any generality. The digitized version of a convex shape is a set of black grid points on a background of white ones. As a union of square pixels the digitized shape will not be convex. Hence much work was done addressing the question whether there is a good definition of convexity for discrete objects [20, 37]. A variety of proposals were made and can be found in the literature. The metrology questions however, in all cases remain: determine with best precision the location (first order moments), orientation (second order moments) and other metric properties, like area (zeroth order moment) and perimeter of the shape. These questions, too have received some attention. It turns out that computing the moments of the black grid points yields good estimates for the corresponding continuous quantities, and more refined, boundary estimation procedures (say, based on polygonalization of the jagged boundary via an efficient digital straight segment detection, as discussed above) do indeed provide improved estimates but the improvement needs to be carefully weighed against the increased complexity involved.

Among the many procedures that propose polygonal approximations to preimages based on the discrete grid points that were covered by the shape, and also based on the ones that were not covered, one stands out in elegance and usefulness: the *minimum perimeter polygon* that is enclosing all black (covered) points and excludes all white (uncovered) ones. This minimum perimeter polygon turns out to be the relative convex hull of the connected graph of sampled black points with respect to the white ones. Here we assume that sampling is dense enough so that a connected preimage shape ensures that the black pixels form a 4-connected shape! The relative convex hull can be computed easily and may serve as a good approximation for preimages for all metrology purposes.

So far we talked about disks and convex objects. The next level of complexity in planar shapes are the so called star-shaped objects. These are defined as the shapes that have a “kernel region” inside them so that from any point in the kernel the entire boundary of the shape can be “seen”, i.e., a line from the chosen point to any boundary point will lie entirely inside the shape. It is easy to see that this definition generalizes convexity in a rather natural way and that the kernels must be convex regions. Determining star-shapedness of a planar shape is not a too difficult task for polygons and for spline-gons, and the algorithms for doing this rely on locating and using the inflection points on the boundary, and intersecting the regions in the plane from where the convex boundary regions are seen, see [5]. As with the notion of convexity, determining digital star-shapedness posed a series of special problems that needed careful analysis. This was the topic of a paper by Shaked, Koplowitz and Bruckstein, and there it was shown that the relative convex hull, or minimal perimeter polygon of the grid points covered by the shape with respect to the ones that remained uncovered, provides a convenient computational way to define and algorithmically determine digital star-shapedness, see [33].

1.6 Shape Designs for Good Metrology

Up to this point we have discussed ways to analyze and measure planar shapes when seen through the looking glass of grid probing, or point-sampling discretization. The classes of shapes were assumed given in some, perhaps parameterized form, and we dealt with questions about recovering their various features and parameters, or about measuring their size and perimeter and determining their location with the highest precision possible.

When considering such issues, a further question that can be posed is the following: design planar shapes or collections of shapes that will interact with the discretization process in such a way that the quantities we need to measure will be very easily read out in the discretized images we get. Could we design an object in the plane (that can be a union of continuous binary shapes), so that digitization of this object translated to various locations, will yield black and white patterns on the (discretization) grid that clearly exhibit, say in a binary representation, the X and Y translation values up to a certain desired precision?

Interestingly, recently a pen-like device was invented and advertised, that has the following feature: it automatically computes with very high precision the location of its tip on any of the pages of a paper pad by looking at a faint pattern of dots that is printed on these sheets of paper. The pattern of these dots is so designed that the image obtained on any small region as seen by the pen near it's tip (with the help of a tiny light detector array) uniquely and easily locates the pen-tip's position on any of the pages of the pad, see [1].

This example shows that it is good practice to think about designing shapes to have such "self-advertising" properties and this approach could provide us surprisingly efficient and precise metrology devices. This problem was posed by Bruckstein, O'Gorman, and Orlitsky, at Bell Laboratories, already in 1989, with the aim of designing planar patterns that will serve as location marks, or fiducials on printed circuit boards. The need for location or registration fiducials in printing circuit boards and in processing VLSI devices is quite obvious. When layers of printing and processing are needed in the manufacturing operation, the precision in performing the desired processes in perfect registration with previously processed layers is indeed imperative. The work described in [7] proves that there exists an *information theoretic bound* that limits the location precision for any shape that has an spatial extent of say $A \times A$ in pixel-size. Such a shape, when digitized will provide for us about A^2 meaningful bits of information, via the pattern of black and white pixels in the digitized image. This number of bits can only effectively encode 2^{A^2-1} different locales, and hence the precision to which we can refine a region one pixel-square in size has a maximal area that must exceed $1/(2^{A^2-1})$. If we want balanced X and Y axis precision, we can only locate the pattern to a subpixel precision of $1/[2^{(A^2-1)/2}]$. This is the best precision possible assuming optimal exploitation of the real estate of an $A \times A$ area, assigned to the location mark. The important issue that was further settled in [7] is the existence of a fiducial pattern that indeed achieves this precision. The pattern is so cute that we exhibit it in Fig. 1.6.

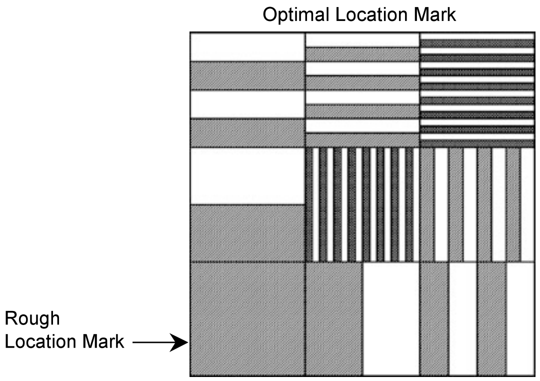
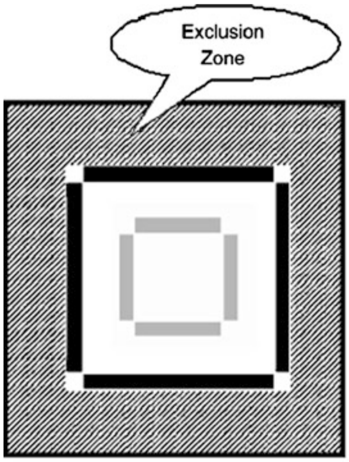


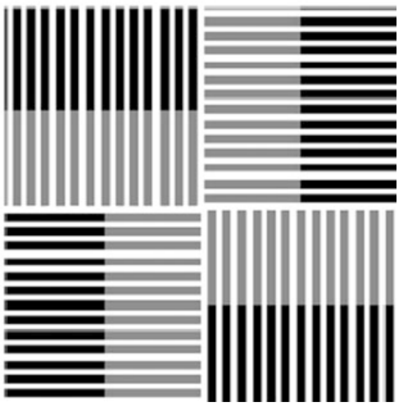
Fig. 1.6 An optimal 2D fiducial of area 3×3 pixels

Looking at this fiducial pattern it becomes obvious what it does. It is indeed a continuous 2D (analog) input that employs the point sampling discretization process to compute its X and Y displacement by providing a binary read-out of the subpixel location of the fiducial within the one pixel to which it can readily be located using the left lowest grid-point (the “rough location” mark) covered by the shape. This leftmost bit of information is also the reason we can only use $A^2 - 1$ bits for subpixel precision, i.e., for cutting the one pixel precision (provided by the “rough location” bit) into locale slices. This process turns the fiducial and the discretization process into a nice “analog computer” that yields the displacements in the X and Y direction easily, and achieves the highest precision in this task that is possible based on the available data. The analysis provided in [7] goes even further. The optimal fiducials turn out to require highly precise etchings on the VLSI or circuit board devices and hence might be difficult to realize in practice. Hence there is a need to analyze other types of fiducial shapes that achieve suboptimal exploitation of the area, however can provide good location accuracies. For rotational invariance, circularly symmetric shapes turn out to be necessary, and therefore bull-eye fiducials were also proposed in [7] and further analyzed in [13, 34]. The main message of the theoretical analysis provided in [7] was that a self location fiducial should have lots of edges that carry information on their location when seen through a digitization camera. Recently, the semiconductor industry used this insight in re-designing the standard registration fiducials. This was the result of a detailed study of novel, robust grating mark fiducials, which greatly increased precision and repeatability. The study, done by us in conjunction with a team of design engineers at KLA-Tencor, a leading manufacturer of vision based process inspection machines for semiconductor industry, proposed fiducial marks as shown in Fig. 1.7b to replace the traditional box in a box mark shown in Fig. 1.7a. The traditional fiducial was clearly not optimal in terms of exploiting the wafer area allocated to it. For a detailed description of the optimized overlay metrology marks that were adopted by industry and the theoretical analysis that led to their design, see [2].

The most interesting question that remains to be addressed here is the following: can we invent shapes that provide other metrological measures as easily as the above discussed example advertised its location?



(a) Box in Box fiducial.



(b) The new standard fiducial.

Fig. 1.7 Overlay metrology fiducials (from [2])

1.7 The Importance of Being Gray

So far we have discussed the case of binary continuous images being point-sampled into matrices of zeros and ones, or Black and White pixels. However the real world is far richer in possibilities and complications.

First of all, point sampling is not a good model of the imaging process as performed by real life cameras. Those carry out, at each sensor level, a weighted integration of the incoming light from the continuous input pattern. This integration happens around each grid point, and the pixel influence region may be assumed circular. The integration yields, at each grid point, values that continuously vary from a lowest value for white (no object) input over the pixel influence region to highest value that corresponds to having the input object cover the entire area of integration. The result of this integration is then transformed into a discrete value encoded by several bits, via quantization. Therefore even for binary preimages, we get at each grid point a pixel value that is the quantization of a continuous variable proportional to the fraction of the pixel influence region that is covered by the input object.

Furthermore we may also consider the advantages of using non-binary, grayscale of color pre-images. The combination or more realistic sampling and quantization processes with the use of gray levels in preimages open for us a great variety of further possibilities. As an example, Kiryati and Bruckstein have analyzed, following a question posed by Theo Pavlidis, the trade-off between spatial resolution and number of gray levels when the aim is to get as much information as possible on a class of binary pre-images that comprise polygonal shapes. The conclusion of this research was that "Gray Levels Can Improve the Performance of Binary Image Digitizers", see [19]. The paper introduces a measure of digitization-induced ambiguity in recovering the binary preimage, hence it is quite relevant to metrology under such sampling conditions. It is then proved that, if the sampling grid is sufficiently dense (i.e., the sampling rate is high!) and if the pixels would provide us exact gray-levels rather than quantized values, error-free reconstruction of the binary pre-image is possible. This is not too surprising, however, when the total bit budget for the digitized image representation is limited (i.e., the sampling rate and the quantization depth are related, both being finite) the bit allocation problem that arises shows that the best resource allocation policy is to increase the gray level quantization accuracy as much as possible, once a sufficiently dense spatial sampling resolution has been reached. Therefore once we have a grid dense enough to ensure that all linear borders of the binary input image polygonal shapes can adequately be "seen" in the sampled image, all the remaining bit resources should go towards finer gray level quantization. The question, which prompted this research asked to explain why gray-level fax machines at low resolution yield nicer images than fax machines at higher resolution, even for binary document images. It was clear that some sort of anti-aliasing effect is in place, however [19] proved quantitatively that even in terms of a well-defined metrology error measure, the gray-levels help considerably more than increased spatial resolution.

Imagine next that we allow gray level input images too. In this case we shall certainly have, in conjunction with multilevel quantizations at each pixel much more information for location and various other measurements. A gradual boundary in the input image, or equivalently an area integration sensor providing a quantized multilevel pixel value at each grid-point, will transform the issue of locating a half plane into a problem of locating precisely several parallel digital straight edges, when they are simultaneously sampled. Such richness of detail will certainly dramatically reduce the size of the uncertainty locales, and enable us to design a wealth of improved location and orientation fiducials in the future.

The conclusion therefore is that gray levels matter, they are good for us! And the last word on these issues certainly has not been said yet. For some very nice recent work along these lines see [35].

1.8 Some Further Open Questions

As was mentioned in the previous sections, there are still many interesting and open digital geometry and metrology problems. Although digital straight edges did receive a lot of attention from digital geometry researchers we still expect to see complete theories pertaining to the sampling and quantization of linear non-step borders in gray level preimages. If a straight border with sigmoidal gray level profile is sampled by some type of area sampling (with pixels with circularly symmetric integration regions) the result will be a border-line with quantized gray levels that will look like a nicely anti-aliased line produced by a computer graphics algorithm. There are interesting digital line properties of the type we discussed in Sect. 1.3 embedded in the resulting image and these will surely be carefully studied sometime in the future.

Along these lines one could also study a class of location fiducials based on shapes with multiple parallel edges, or edges with an a priori known pattern. Such robust fiducials should enable "the design" of desired uncertainly locales for high precision registration, may even be insensitive to pixel size variations.

Another interesting question on self-location that may be subject to further research is the design of binary self-location patterns in the plane. This problem was partially addressed in the paper [15], the pattern proposed being a separable bit-pattern that is generated as the outer (binary) product of two one-dimensional de Bruijn sequences [28] that have the one-dimensional self-location property. Such a pattern can be shown to be robust to some read-out errors but clearly it has a bit too much redundancy built into it. The planar pattern used by the Anoto pen we mentioned before, [1], is an "analog" point pattern that is based on encoding location in geometric constellations of points near grid locations that carry the information on the absolute coordinates of the grid point. It seems that a binary array version of the problem has not been discussed before the work reported in [15].

The problem of length estimation of discretized boundaries was the subject of many papers, as seen in [21, 35] and the references therein. However even this topic was not yet completely exhausted. It is an interesting challenge to design perimeter estimators that will work in conjunction with corner detectors and curvature estimators, perhaps based on digital circle detectors [10], to yield more and more precise length measurements. The design here should not be aimed to get precise results on digital straight lines but rather on various types of continuous curves with breakpoints and corners, and the ranges of curvatures that are expected to appear in practice.

As we discussed in the previous section, subject of bit allocation tradeoff's between resolution and quantization has only been superficially touched upon so far [17, 19, 35]. Although the initial conclusions are that multilevel quantization provides quite a lot of information in binary preimage digitization, a similar study should be made for the case of gray level shape boundaries and gray level images of various types. In this context one might even ask what should be the design of the gray-scale profile of planar shape edges to enhance the edge location and length estimation performance.

We have not discussed in this paper important questions of shape comparison and recognition, of shape decompositions and isoperimetric inequities for digitized shapes. These topics all rise very interesting research questions that are recently beginning to be addressed, see, e.g., [8, 36]. Therefore we may expect the area of digital geometry to remain an active and exciting subject of research in the future.

1.9 Concluding Remarks

This paper surveys research that dealt with digital geometry and metrology issues. As is clear form the topics discussed above and the list of references below, metrology tasks require deep and interesting excursions into discrete geometry, motivating the study of the pixelized world and importing from it important insights and results. More on the vast subject of discrete geometry can be found in several books [9, 11, 20, 22, 26, 27, 29].

Acknowledgement Many thanks to Ms. Yana Katz for preparing this paper for publication.

References

1. http://www.anoto.com/the_paper_3.aspx
2. Adel, M., Ghinovker, M., Golovanevsky, B., Izikson, P., Kassel, E., Yaffe, D., Bruckstein, A.M., Goldenberg, R., Rubner, Y., Rudzsky, M.: Optimized overlay metrology marks: theory and experiment. IEEE Trans. Semicond. Manuf. **17**(2), 166–179 (2004)
3. Allouche, J.P., Shallit, J.: Automatic Sequences. Cambridge University Press, Cambridge (2003)
4. Baryshnikov, Y.: Complexity of trajectories in rectangular billiards. Commun. Math. Phys. **174**(1), 43–56 (1995)
5. Bornstein, R., Bruckstein, A.M.: Finding the kernel of planar shapes. Pattern Recognit. **24**(11), 1019–1035 (1991)
6. Bruckstein, A.M.: Self-similarity properties of digitized straight lines. Contemp. Math. **119**, 1–20 (1991)
7. Bruckstein, A.M., O'Gorman, L., Orlitsky, A.: Design of shapes for precise image registration. IEEE Trans. Inf. Theory **44**(7), 3156–3162 (1998). AT&T Bell Laboratories Technical Memorandum, 1989
8. Bruckstein, A.M., Shaked, D.: Crazy-cuts: dissecting planar shapes into two identical parts. In: IMA Mathematics of Surfaces XIII Conference, The University of York, UK, September 7–9, 2009
9. Chassery, J.M., Montanvert, A.: Géométrie Discréte en Analyse d'Images. Hermes, Paris (1991)
10. Coeurjolly, D., Gérard, Y., Reveillès, J.P., Tougne, L.: An elementary algorithm for digital arc segmentation. Discrete Appl. Math. **139**, 31–50 (2004)
11. Davis, L.S. (ed.): Foundations on Image Understanding. Kluwer Academic, Dordrecht (2011). (The Azriel Rosenfeld Book)
12. Dorst, L.: Discrete straight line segments: parameters, primitives and properties. Ph.D. thesis, Technological University Delft (1986)
13. Efrat, A., Gotsman, C.: Subpixel image registration using circular fiducials. Int. J. Comput. Geom. Appl. **4**, 403–422 (1994)
14. Freeman, H.: On the encoding of arbitrary geometric configurations. IRE Trans. Electron. Comput. **EC-10**, 260–268 (1961)
15. Giryes, R., Shuldiner, D., Gordon, N., Holt, R.J., Bruckstein, A.M.: Simple and robust binary self-location patterns. IEEE Trans. Inf. Theory (2012). doi:10.1109/TIT.2012.2191699
16. Gómez-Martín, F., Taslakian, P., Toussaint, G.: Structural properties of euclidean rhythms. J. Math. Music **3**(1), 1–14 (2009)
17. Gustavson, S., Strand, R.: Anti-aliased euclidean distance transform. Pattern Recognit. Lett. **32**, 252–257 (2011)
18. Havelock, D.I.: The topology of locales and its effects on position uncertainty. IEEE Trans. Pattern Anal. Mach. Intell. **13**(4), 380–386 (1991)
19. Kiryati, N., Bruckstein, A.M.: Gray levels can improve the performance of binary image digitizers. CVGIP, Graph. Models Image Process. **53**(1), 31–39 (1991)
20. Klette, R., Rosenfeld, A.: Digital Geometry—Geometric Methods for Digital Picture Analysis. Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. Morgan Kaufmann, San Francisco (2004)

21. Koplowitz, J., Bruckstein, A.M.: Design of perimeter estimators for digitized planar shapes. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-11**(6), 611–622 (1989)
22. Latecki, L.J.: Discrete Representation of Spatial Objects in Computer Vision. Springer, Berlin (1998)
23. Lindenbaum, M., Bruckstein, A.M.: On recursive, $O(N)$ partitioning of a digitized curve into digital straight segments. IEEE Trans. Pattern Anal. Mach. Intell. **15**(9), 949–953 (1993)
24. Lothaire, M.: Algebraic Combinatorics on Words. Cambridge University Press, Cambridge (2002)
25. McIlroy, M.: A note on discrete representation of lines. AT&T Bell Labs Tech. J. **64**(2), 481–490 (1984)
26. McIlroy, M.D.: Number theory in computer graphics. Proc. Symp. Appl. Math. **46**, 105–121 (1992)
27. Melter, R.A., Rosenfeld, A., Bhattacharya, P. (eds.): Vision Geometry. Contemporary Mathematics, vol. 119. AMS, Providence (1991)
28. Mitchell, C., Etzion, T., Paterson, K.G.: A method for constructing decodable de Bruijn sequences. IEEE Trans. Inf. Theory **42**(5), 1472–1478 (1996)
29. Pavlidis, T.: Algorithms for Graphics and Image Processing. Comput. Sci. Press, Rockville (1982)
30. Pleasants, P.A.B.: Quasicrystallography: some interesting new patterns. In: Elementary and Analytic Theory of Numbers, 2nd edn. Banach Center Publications, vol. 17, pp. 439–461. PWN, Warsaw (1985)
31. Proffit, D., Rosen, D.: Metrication errors and coding efficiency of chain coding schemes for the representation of lines and edges. Comput. Graph. Image Process. **10**, 318–332 (1979)
32. Senechal, M., Taylor, J.: Quasicrystals: the view from Les Houches. Math. Intell. **12**(2), 54–64 (1990)
33. Shaked, D., Koplowitz, J., Bruckstein, A.M.: Star-shapedness of digitized planar shapes. Contemp. Math. **119**, 137–158 (1991)
34. Shih, S.W., Yu, T.Y.: On designing an isotropic fiducial mark. IEEE Trans. Image Process. **12**(9), 1054–1066 (2003)
35. Sladoje, N., Lindblad, J.: High-precision boundary length estimation by utilizing gray-level information. IEEE Trans. Pattern Anal. Mach. Intell. **31**(2), 357–363 (2009)
36. Vainsencher, D., Bruckstein, A.: On isoperimetrically optimal polyforms. Theor. Comput. Sci. **406**, 146–159 (2008)
37. Voss, K.: Discrete Images, Objects, and Functions in Z^n. Springer, Berlin (1991)