DIGITAL SCALING OF BINARY IMAGES

by

ROBERT A. ULICHNEY

B.S., University of Dayton
(1976)

Submitted in Partial Fulfillment
of the Requirements for the Degree of

MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January, 1979

Signature redacted

Signature of Author.....
Department of Electrical Engineering
and Computer Science, January 19, 1979

Signature redacted

Certified by.....
Thesis Supervisor

Signature redacted

Accepted by.....
Chairman, Departmental Committee on Graduate Students

DIGITAL SCALING OF BINARY IMAGES

by

ROBERT A. ULICHNEY

Submitted to the Department of Electrical Engineering and Computer Science on January 19, 1979, in partial fulfillment of the requirements for the Degree of Master of Science.

ABSTRACT

The importance of enlarging and reducing two-level images such as graphical and documentary matter by digital means continues to grow as the phototypesetting industry converts to digital storage of type masters. Besides the unattractive alternative of scaling by changing resolution, relatively no work has been done on this problem.

A mathematical examination is made from the linear processing point of view. The coarse process of binary thresholding, however, constitutes a very appreciable irreversible non-linear contribution. The resulting unnaturally abrupt edges which are the essence of binary images along with other reasons make scaling by linear means impractical.

A non-linear scaling scheme is devised which exploits the simplicity of this binary nature treating it logically instead of mathematically. A priori knowledge of general contour characteristics that can occur within a given image "window" are stored in a "Telescoping Template" and aid in the internal continuous reconstruction of the original image to be resampled at a different spatial frequency. To enhance the quality of reductions a method of non-linear "Binary Convolution" was developed. Another non-linear scaling scheme based on resampling original samples is simulated for comparison. The method of the Telescoping Template yields high fidelity digital scaling and meets the objectives of being fast, conducive to hardware realization, and void of special pre-encoding requirements.

Thesis Supervisor: Donald E. Troxel

Title: Associate Professor of Electrical Engineering

To the letter

for the many embarrassing
contortions it endured throughout
the course of this endeavor.

TABLE OF CONTENTS

CHAPTER 1

PROLOGUE

This investigation deals with the problem of taking a given set of image samples and producing another set of samples which represents the original image at a desired scale within the domain of binary images. For this document a BINARY IMAGE is defined as a two-dimensional signal whose amplitude is precisely either Black (numerically and logically represented as 1) or White (numerically and logically represented as 0). Further, it is assumed that such an image is perceived visually as binary. This excludes high frequency distributions of black and white which produce the illusion of a grey or intermediate area as in half-tone pictures.

DIGITAL SCALING is the process performed on the digital input image resulting in a digital output image of a different size at a fixed resolution. It is the intention of this study to devise such a process which preserves the integrity of the original image while meeting the following objectives:

1. High speed

2. Does not require special pre-encoding of source images or a full two-dimensional intermediate memory space

3. Conducive to hardware realization.

Figure 1-1 graphically defines some important conventions that will be maintained throughout this work. A digital image may be thought of as a two-dimensional array or matrix. The positive direction of horizontal and vertical index incrementation is shown along with index variables which will be reserved for each. The I/O scanning direction is also suggested in the image area as bottom-to-top vertical lines moving from left to right. Each picture element or pel of this matrix can be thought of as a 1 by 1 area with an amplitude corresponding to its color (black or white) which at times will simply be represented as a dot.

The input parameters are:

s_x the Horizontal Scale Factor

s_y the Vertical Scale Factor

G[m,n] the Given M by N Binary Image Array

where m = 1,...,M

n = 1,...,N

The output is

S[k,ℓ] the Scaled K by L Binary Image Array

where k = 1,...,K

ℓ = 1,...,L

and K = Integer(s_yM),

the number of vertical pels

$$L = \text{Integer}(s_x N),$$

the number of horizontal pels.

These and other parameters defined throughout the text are arranged in alphabetical order for easy reference in Appendix A.

In all mathematical expressions, square brackets "[]" will be used to contain the arguments of a discrete function while parentheses "()" will be used for continuous functions. The standard symbol, "*", will be used to indicate the convolution operation. To aid in the symbolic translation from discrete-space to continuous-space signals, the Dirac delta function

$$\delta(x-x_o)$$

will prove useful. It is assumed to be zero everywhere except at $x = x_o$ and possesses the integral property

$$\int_{-\infty}^{\infty} f(x)\ \delta(x-x_o)\ dx = f(x_o)$$

Readers interested only in the development of the Telescoping Template method, the principle result of this research, should skip directly to Chapter 5. It is recommended that at least Chapter 4 be read first to dispel any temptations one might have to perform digital scaling of binary images by means of linear processing. This chapter

also suggests ideas for future study in the area of digital scaling of multi-level images.

The evolution and classification of the solution to this problem primarily in the commercial realm is the theme of the second chapter. The third chapter is presented to help better understand the nature of the special class of images which we are dealing with.

As an editorial comment, I would like to apologize if the variable "ℓ" used to represent lower case "L" appears awkward. The IBM Courier type font used in this document does not distinguish between the number 1 and the letter l. By itself the symbol "l" will always designate the number one.

When viewing the computer generated images in the illustrations it is important to consider the resolution at which they are presented. Some images are displayed at a lower resolution than others to better illustrate the result of some digital process. Figure 4-4 was created on a Laserphoto Receiver, a laser printer developed at M.I.T. for the Associated Press. All other computer generated images were produced on a Verian plotter used in raster mode. Although the resolution indicated for eash image is expressed in terms of lines/inch, it is implied to be both the horizontal and vertical resolutions.

CHAPTER 2

THE SCALING PROBLEM

Long before Gutenberg's invention of the type mold in the 1400's, the Chinese and Koreans printed binary images (text and pictures) from cut wood blocks. When an enlargement was desired, an enlarged image was cut on a larger wood block. This may seem ludicrous to mention but it was this philosophy which constituted the solution to the scaling problem in all phases of printing until very recent years! This of course includes the method of printing known as "hot type" where binary images were produced by inking metal-cast raised letters. The scaling process, if it is to be referred to as such at this level, was performed by the type designer in a type foundry. A different size character was an independent piece of hardware. This form of binary image reproduction is still in existence today but in rapidly diminishing numbers.

The big (and virtually only) turning point in this craft occurred in the early 1950's when two Frenchmen, Rene' Higonnet and Louis Moyroud, introduced phototypesetting with their invention of the "PHOTON". This was the dawn of a

method known as cold type or direct impression composition. The principle difference was the means by which the sources were stored. Instead of a physical piece of metal, the image of a character on a revolving glass matrix disk was the media. A stroboscopic lamp passed light through the desired character into an optical system. This magnified image was then projected onto the proper location of a sensitized film later used to create the actual printing plate [1].

Thus, from one master set of typeface images a character could be optically scaled to any of a variety of selected sizes. The important advantages spawned by such an invention are evident. Besides being faster and less costly, photo-composition provides a greater variety of typographical fonts without the bulk and difficulty of moving heavy casting forms.

The evolution of this industry, from the photon to the current state of the art, is conventionally segmented into three stages or "generations" as referred to in [2,3,4]. Products resembling the opto-mechanical "photon" comprise the so called first generation phototypesetters. These vary only in the uses of photomatrix type: the physical means by which the character images are stored. Besides the rotating disk, some use film masters mounted on rotating drums.

Second generation machines eliminate the optical system linking the source image to the film. The source in this case is a glass grid of photographic character masters. A

selected image is either (1) scanned by a vidicon tube, or, (2) illuminated by an indexing CRT tube and sensed by a photomultiplier as shown in Figure 2-1. The image is then displayed on an output CRT used to expose film.

And finally, as is inevitable for most any trade involved with accuracy and speed, digitization invaded phototypesetting to comprise the third generation. As in the second generation typesetting, a CRT is used to display the output but the source image masters are stored digitally instead of on a glass grid. The commercial leaders in this area are Autologic, MGD, Harris, and Mergenthaler.

I find it useful to re-segment this evolutionary catorization of phototypesetting into the following four levels (subdividing the third generation) and discuss this development from the perspective of the scaling problem:

1. Optical lens system (1st generation)
2. Source scanning resolution variation (2nd generation)
3. Output resolution variation (3rd generation)
4. Digital Scaling (also 3rd generation)

In the first level an opto-mechanical arrangement of lenses in a revolving turret permitted a finite number of magnifications. This was very inhibitive in that scaling at a desired factor required the inclusion of the correct lens in the turret, and the associated mechanical adjustment of the optical system was relatively time consuming. Also,

horizontal and vertical scaling were not independent.

The second level, which also happens to be called second generation, phototypesetters maintain a constant resolution on the output CRT. Scaling is achieved by controlling the way in which the source character on the glass grid is scanned. For example, close placement of adjacent scanning strokes produces a horizontal enlargement. A vertical reduction would be due to increased source scanning speed.

When the source images are stored digitally scaling becomes a nontrivial issue. Level three includes those digital CRT typesetters that achieve scaling by varying the resolution on the output CRT. This approach invites an array of problems. It is expensive and complex in the analog circuit sense. The scanning density variations are achieved by adjusting the range of voltages sent to the horizontal and vertical deflection plates within the CRT. Of course the intensity of the electron beam must be automatically controlled to compensate for changes in scanning density so as to maintain a constant apparent brightness. There are practical limits in the magnitude in which enlargements and reductions can be performed in this fashion. Typically the scan line separation and scan length can be independently or simultaneously varied from .57 to 2.5 times some standard scanning density. Thus, several different size digital masters are required to cover a complete range of output sizes, necessitating additional storage.

Level four represents the frontier of the state of the art; those digital CRT typesetters which have invariant output resolution and perform enlargements and reductions digitally. Commercially only two major products fall into this category: the Mergenthaler Linotron 202 and Rockwell MGD Metro-set; each of which shall be focused on.

Mergenthaler's former prized top of the line product, the Linotron 606, was typical of most level three digital CRT typesetters in that it stored its fonts in some one-dimensional redundance reduction technique (as outlined in section 3.2). And because of the inherent limitations in the analog method of varying resolution to achieve scanning, four master sizes were required to cover the 4 to 72 point range for a particular type style.

However, on June 5, 1978, a new product void of these arduous necessities was surprisingly unveiled at the American Newspaper Press Association (ANPA). In the words of a recent Seybold Report [6, page 1]: "The introduction of the Linotron 202 has changed the nature of the typesetting market." Except for a slight degradation in scaled image quality and output speed, this new product is quite comparable to the Linotron 606 at nearly one-fourth the price. The reason for the tremendous reduction in cost: Digital Scaling! Only <u>one</u> master font contour coded as straight line segments is stored for a given style. Of course higher order curves could have been used but a premium was placed on storage economy since

only two floppy disks are used.

The model size after which the masters were encoded was 48 points (about .67 inch). Although scaling in the range between 4 and 72 points yielded acceptable output for newspaper requirements, degradations become apparent for enlargements over 72 points.

Encoding type masters by fitting curves to their contours is a natural solution to the scaling problem since its nature is insensitive to scaling. The quality of an enlargement is limited only by the care in which the contour was defined.

The MGD Metro-set employs a patented [5] contour coding scheme consisting of circular arc and straight line segments. Typically only one character master is sufficient for all sizes, however, for reasons described in Section 3.3 some styles really required two or three masters since it changes form when imaged at different sizes. The precision of MGD's encoding procedure is higher than the Linotron 202 but requires three to four times more storage [6, page 8] and is available for about twice the price. Also unlike the 202, the model size of a Metro-set master character is its maximum displayable size.

Outside of the digital phototypesetters themselves, the only industrial utilization of digital scaling of binary images occurs in some video terminals used to proof or "soft

typeset" text. Methods used here are quite crude since the objective of such output devices is recognizability while conveying true character heights, widths, and locations on the page.

One such example is Raytheon's Raycomp-100 video page composition terminal [7]. Regardless of the type style used, all symbols displayed are scaled versions of one specially designed font called "Raytheon Gothic", each character of which is stored in the form of a 12 by 16 dot matrix. The terminal does generate the true dimensions of the character but they are "rather strange looking" [8]. Scaling seems to involve a sample-and-hold for enlargements and a simple deletion of points (which I refer to as "Coincident Resampling" in Section 5.1) for reductions yielding less than attractive results.

Clearly the world of typesetting is a very dynamic one and the above is merely a snapshot of the industry at the time of this writing. It is also evident that digital scaling of binary images is a very critical issue at least in this discipline. It is the intention of this research to thoroughly examine and contribute to its solution.

CHAPTER 3

PROPERTIES OF DIGITAL BINARY IMAGES

3.1 RESOLUTION

Probably the most critical property contributing to the quality of a binary image is the fineness of its spatial resolution. Besides being the most dominant feature of image appearance, it alone often determines the optimum redundance reduction scheme (in terms of compression ratios) to use for a given class of images. And most importantly for this work, it establishes how one should best approach the scaling problem. Binary images are of course not band limited. But the human eye can only resolve a certain upper spatial frequency limit; That is, there exists a sampling grid fine enough so that sampling at any higher rate will produce no observable improvement in image quality. In the case of text, a lower bound on resolution has been experimentally determined [9] for reliable (97.5%) identification or legibility of characters of certain sizes.

A grid density is typically chosen between these bounds. One wishes to minimize cost in terms of storage and time by

using the lowest resolution tolerable for a specific quality requirement. The most stringent demand on resolution is found in the printing of text books where in some cases no compromises in quality are permitted; no trace of digital quantization will be accepted. In other cases where mere recognizability is all that is needed the demands on resolution are considerably more lenient.

A list of such requirements has been reported by Walter [10, page 27]:

	End Use	Approximate No. Strokes/Inch
a)	Text books (incl. Dictionaries & Encyclopedias)	>1000
b)	Books (except pulp publications)	900
c)	Magazines & Directories (including Catalogs)	700
d)	Pulp publications (Pocket Books & Magazines on Newsprint semi-glazed paper)	600
e)	Newspapers (produced by off-set printing)	550
f)	Newspapers (produced by rotary letterpress)	500
g)	Proofing Mode for a) through d)	300
h)	Proofing Mode for e) through f)	250
i)	Computer Printout (using stylized font)	175

Thus, depending on the particular application the resolution may be low enough to allow perception of quantization noise but sufficient enough to provide tolerable recognition of information. As for scanning direction (horizontal or vertical) one has not been found preferable to the other, at least in the case of printed matter [9].

In this study we are not concerned with the problem of amplitude quantization (aside from determining the threshold level of the quantizer). It has been an underlying assumption that assigning one bit of color information to the samples of graphical and documentary sources would be sufficiently exact. It is important to note that for high resolution work, this assumption has been empirically verified. Sampled images have been compared using continuous and binary tone scales at various resolutions. It has been determined that for grid densities over 200 lines/inch in both dimensions the results become indistinguishable [11].

At this point I would like to differentiate low and high resolution binary image scaling philosophies using the figure 200 lines/inch as the threshold for this segmentation. For the high resolution case the image outline established by the black to white edges is of primary interest. In this investigation it is assumed that this indeed is the case.

In the case of low resolution images, the precise location of each individual picture element contributes quite appreciably to the appearance of the output. Also, the

additional variables of shape and size of a picture element are open to optimization. An example where low resolution scaling occurs is in the resampling of a high resolution type design for use on low resolution dot matrix printers for some commercial terminals. The Research & Development Group at Digital Equipment Corporation, with whom I am currently associated, is involved in such an effort. It is found that the phase with which the new low resolution sampling grid is placed in relation to the source image is a very important parameter. Optimizing this along with the determination of color (black or white) for each new picture element seems to be very much in the realm of art necessitating human intervention. Automation of such a process remains an unsolved problem. One thing is certain, however, and that is that high resolution resampling is of an entirely different nature.

3.2 REDUNDANCY REDUCTION CODING

In the last section the idea of spatial quantization of continuous images was considered. This kind of compression has been described as an irreversible "information reduction" [12]. Except for some contour coding schemes, "redundancy reduction" techniques are noiseless in that the original digital image can be fully recovered after decompression. The nature of binary images (as defined hereinbefore) is low in entropy; there is a high correlation between neighboring pels. Because of the high cost in terms of amount of storage space used or time needed to transmit such information over a finite channel, much effort has been directed towards exploiting this fact to devise redundancy reducing codes. The purpose of this section is to review the various facsimile coding methods currently practiced and studied.

This is an internationally active area of research with excellent comparisons and summaries of various methods published by Musmann [13], Takagi [14], Arena [15], Huang [16], and Preuss [17]. Most literature on the subject makes reference to the theoretical work of Huffman [18] in 1952. His code employs variable length code words where the code word lengths are related to the message frequencies; the more frequent messages being assigned the shorter code words. Although this scheme provides very high compression ratios,

its utility is inhibited by the extreme complexity in its implementation, as well as the need to have a very large coding table. It is thus used only as a theoretical bound.

Many codes have been developed which achieve bit rates which are nearly as low as the Huffman code but are easy to implement. As pointed out by Huang [16], the majority of codes currently used are based directly or indirectly on the concept of transmitting only boundary points. The only unique information in binary facsimile is contained in the black to white transition areas. This is especially true for higher resolution sampling grids as the redundancy in the image increases. The most popular method involving the concept of storing or transmitting only boundary points is runlength coding.

Runlength Coding

The simplest method of runlength coding is one which uses a fixes length code word. The codeword length, N, is the minimum integer which satisfies the inequality

$$N \geq \log_2 M$$

where M is the maximum possible run length; a run length being the number of consecutive pels of the same color along a scan line. Typically the color need not be transmitted since black and white runs always alternate. Fixed length codes are extremely simple to encode and decode at the price of a relatively high bit rate. This method is thus not used

today.

Many varieties of variable length codes have been developed. A well known publication on "optimum" run length codes by Meyr, et.al. [19] considers and compares two such codes referred to as an older "A-code" (as referred to in his text) and newly proposed "B-code". These codes are optimum for the mathematical models used to estimate the statistics of run lengths. The A-code was adaptive in that the lengths of the code blocks which make up a code word were allowed to vary. It was based on an exponential model which resulted in code word lengths roughly proportional to run lengths. The B-code was found preferable over the A-code because while providing comparably low bit rates its block length does not need to be adaptively varied, thus being easier to implement. The overall code word length increases roughly as the logarithm of the run length, more suited for long runs which are more frequent than previously predicted.

Renelt [20] recently suggested a one-dimensional runlength code which is easy to implement yet yields a 15% improvement over the above B-code in redundancy reduction. Other variable length codes have been presented by Takagi and Tsuda [21] and Weber [22]. The uniqueness of Weber's method is that it takes advantage of line to line correlation by simultaneously encoding two adjacent scan lines. Huang also exploits the vertical correlation with his method basically transmits the differences between black and white transition

edges from one line to the next.

Besides runlength coding, Schreiber, et. al. [24] reviews three other contour coding schemes: Direct Postion Transmission, Direct Contour Tracing, and Fitting Curves to Contours. The compression ratios are compared for various scanning densities.

Direct Position Transmission

Direct position transmission involves retaining the coordinates of all the contour points. Although this coding yields a relatively high bit rate (low compression ratio) its advantage over the other two methods is that it does not require storage for encoding and decoding.

Direct Contour Tracing

This process exploits the fact that all contour points must be adjacent to each other. Thus, after the starting point of a contour has been specified, the contour can be "traced" by merely indicating the direction of each succeeding point. For a rectangular array of samples, only 8 directions are possible (requiring only 3 bits).

A complete algorithm and detailed study of this idea has been presented by Morris [25, 26] who refers to it as "chain-link" coding. He also provides a smoothing procedure which removes the "kinks" caused by scanner quantization noise.

A very primitive means for digital scaling has been presented by Freeman [27] who used this encoding idea to represent outlines or geometric shapes. Enlargements and reductions were confined to integral steps.

Fitting Curves To Contours

Of all the codes thus considered as well as block coding discussed below this scheme is the only non-exact method. The explicit position of each contour point is only approximated; the higher the order of polynomial used to fit the curves, the better the approximation. Parameterizing the contours in this way usually results in a very low bit rate at the expense of a "degradation" in image quality. As mentioned in Chapter 2, the only two attempts at digital scaling in the phototypesetting world have used curve fitting to offer the contours of type masters. In the non-commercial realm, algorithms for representing contours of type faces by cubic spline fitting has been studied by Coueignoux [28], and Knuth [29].

Block Coding

A counter example to this trend of transmitting only boundary points is that of skipping white [30] or block coding. The idea involves first considering a one-dimensional (1 x m) or two-dimensional (k x m) block of pels and assigning the one bit code "0" to the block if it

contains all white pels. If it is not all white the prefix "1" followed by a pel by pel binary pattern is used to represent the block. DeCoulon and Johnson further suggest an adaptive block scheme [31] involving a hierarchy of block sizes in which an initial block is divided into several sub-blocks. If the initial block is not all white then each sub-block may be individually "skipped" if the all white condition is met. In any case the size and shape of the blocks used are optimizable characteristics very dependent on the nature of the document being coded. If an image contains a large amount of white, then block coding may perform as well as the Huffman code while being much simpler to implement [32].

3.3 AESTHETIC CONSIDERATIONS

It has been indicated that the quality of a binary image can certainly be improved (to the upper limit of human vision) by increasing resolution and reducing spatial quantization noise. The assessment of the integrety of a scaling procedure depends on those factors which are independent of resolution. For example, if a source image was contour coded by a series of straight line segments, a gross enlargement would suffer from this degradation while subsequent increases in resolution would provide no cure. This section considers those other factors.

The most common form assumed by binary images are type faces. When changing the size of a character it is not a simple photographic reproduction of a large size type of the same style [33]. If this were the case, the smaller type design would appear structurally incorrect. Hairlines and thin sarifs (finishing-off strokes) may tend to disappear, counters (white spaces inside letters) may close up, and ascenders and descenders on lower case letters appear to be the wrong length.

However, it appears that documented guidelines on the rules governing such sizings simply do not exist. In fact, it appears that the world of type design resides firmly in the domain of the artist and "quantitative" details of this

trade are passed on by word-of-mouth. In a letter describing a procedure used to evaluate what "reduction (of a large drawing) did to curves and things" a type designer in 1937 writes:

> "I have a diminishing glass that reduces the letters to something like 12 point size when I put the drawing on the floor and squint at it through the glass held belt-high." [34]

I fortunately had the opportunity to talk to a modern day expert, Raymond Pell [35], who manages the design of type faces for Rockwell MGD, a company recognized for high quality digital images all of which are copied from their own artwork. He contends that there are three classes of type sizes:

1. Classified (under 7 points)
2. Text (7 - 18 points)
3. Display (over 18 points)

Although some forms of type may require a separate design for each of these classes, many require only one or two to cover the full range of sizes. The number of masters needed is thus highly dependent on the nature of the style. The most common adjustment in design occurs for classified type. Counters should be extended and their aspect ratio should be made more symmetric to increase "readability". Also, the weights of stems and hairlines should be increased. Besides appearance improvement, there is a practical reason for the latter; there is a limit in the thinness of a line

which will hold ink on a printing plate.

Figure 3-1 displays two type styles which demonstrate some of these subtle differences. It is noted that type size or height is measured in terms of "points" while "picas" are used to measure type width. In my attempt to find the precise definition of the point I found that all texts on printing agree that it is about 1/72 inch but disagree on the exact value; again, an indication of the lack of quantitative rigor in this art. I believe the definition of the American Point System as established by the U.S. Type Founders Association: 83 picas are equal to 35 centimeters and one pica equals twelve points [10].

CHAPTER 4

THE LINEAR APPROACH

The linear theory developed for digital processing of continuous images has always been concerned primarily with the issue of spatial quantization assuming that quantization of amplitude was sufficiently accurate so as to contribute unappreciable error. The case of binary images is quite different. Digital representation can be considered from two different points of view, each of which presents an unfriendly situation.

First, our sampling process can be modeled as follows. Imagine the sampler being able to retain exact unquantized real amplitude values. Now also assume that the continuous image to be sampled contains precisely only white (0) or black (1) values and nothing in between. Because of the infinitely sharp edges present between white and black values, such a continuous binary image would contain rippling high spatial frequency components which do not die out quickly. Thus, aliasing is unavoidable and one must settle for some tolerance as tabulated in Section 3.1.

The other point-of-view, which is the true model of the

digitization process, does not assume continuous binary images with infinitely sharp edges. Even if such images could physically exist, it is impossible for the point spread response of the sampler to be an ideal impulse. Quantization of the sampled amplitude is reduced to a binary one. This hard thresholding process is what distinguishes this class of digital images from multilevel images. Whether a binary digital image was produced by sampling in this way or if it was created by direct human assignment of picture elements, it does not represent a truly binary continuous image in the linear processing sense.

This is illustrated as a one-dimensional example in Figure 4-1 (which can be mentally extrapolated to two dimensions). A one-dimensional digital binary image is mathematically interpolated in an "ideal" way by convolution with a sinc function to reconstruct from its spatial samples the continuous image it represented.

4.1 SPATIAL DOMAIN

All linear methods of digital scaling essentially do so in two phases (at least internally). The first is to reconstruct or "repaint" the original continuous image from the given samples. The resulting scaled image is then produced by sampling this reconstruction. I refer to this procedure as RETROSPECTIVE RESAMPLING; that is, looking back to the continuous image as it existed before the original sampling, then resampling it.

The first phase involves a two-dimensional reconstruction or interpolation function which shall be designated as h(x,y). Recalling that our given digital image is G[m,n] with defined values for

$$1 \leq m \leq M$$

$$1 \leq n \leq N$$

and zero otherwise. The reconstructed continuous image is thus

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} G[m,n]\ \delta(x'-n,y'-m)\ h(x-x',y-y')\ dx'\ dy'$$

$$= \sum_{m=1}^{M} \sum_{n=1}^{K} G[m,n]\ h(x-n,y-m) \qquad \text{Eq. 4-1}$$

for $1 \leq x \leq N$

$1 \leq y \leq M$

$\delta(x,y)$ is the two-dimensional Dirac delta function used here to include the discrete-space signal G[m,n] in the continuous convolution integral.

If the Given Signal, G[m,n], is band limited and sampled at a spatial rate sufficient to avoid spectral overlap then exact reconstruction can be performed with h(x,y) being an appropriate interpolation waveform of which many exist [36, page 98]. All such functions, as the two-dimensional sinc function for example, are infinite in space, and thus must be evaluated for at least the size of the given image for exact reconstruction.

The meaning of "exact" reconstructon" should be clarified. All physical sampling systems possess an inherent point spread response. Thus, the sampling process can be described as taking place in two steps. First, the original image is convolved with the point spread response or "blurring functon" of the sampler, and secondly, this "blurred" image is sampled (in the ideal sense). It is this blurred image which can be exactly reconstructed.

Whatever the choice of h(x,y), the second phase of Retrospective Resampling involves the sampling of this reconstructed image with a finer (for enlargement) or coarser (for reduction) grid. As defined in Chapter 1, the scaled image, S[k,ℓ] is a K by L matrix. To help describe how a sampled grid of this size is placed relative to the reconstructed image, we will define two arrays.

The HORIZONTAL RELATIVE POSITION ARRAY, $X[\ell]$, is a set of real numbers that map the index ℓ to a number describing its position relative to the index n. So that the set of new samples are equally spaced with endpoints

$$X[1] = 1 \text{ and } X[L] = N:$$

$$X[\ell] = 1 + (\ell-1)\,\Delta x \qquad \text{Eq. 4-2}$$

where $\Delta x = (N-1)/(L-1)$

and $\ell = 1,\ldots,L.$

For example, if $X[5] = 2.31$ then it is known that the new horizontal coordinate $\ell = 5$ fell between $n = 2$ and $n = 3$. Likewise, the VERTICAL RELATIVE POSITION ARRAY, $Y[k]$, is defined as:

$$Y[k] = 1 + (k-1)\,\Delta y \qquad \text{Eq. 4-3}$$

where $\Delta y = (M-1)/(K-1)$

and $k = 1,\ldots,K$

Now the resampling can be mathematically represented by multiplying equation 4-1 by

$$\sum_{k=1}^{K} \sum_{\ell=1}^{L} \delta(x-X[\ell],y-Y[k])$$

This yields the sampled continuous image:

$$\sum_{k=1}^{K} \sum_{\ell=1}^{L} \sum_{m=1}^{M} \sum_{n=1}^{N} G[m,n]\; h(X[\ell]-n,Y[k]-m)\; \delta(x-X[\ell],y-Y[k])$$

Or simply as a discrete-space signal:

$$\sum_{m=1}^{M} \sum_{n=1}^{N} G[m,n]\, h(X[\ell]-n, Y[k]-m)$$

Eq. 4-4

for $\quad 1 \leq k \leq K$

$\quad 1 \leq \ell \leq L$

The amplitude of the samples is equation 4-4 are still assumed to be continuous. Applying an appropriate Binary Threshold finally results in the desired scaled digital image:

$$S[k,\ell] = \underset{\text{THRESHOLD}}{\text{BINARY}} \left\{ \sum_{m=1}^{M} \sum_{n=1}^{N} G[m,n]\, h(X[\ell]-n, Y[k]-m) \right\}$$

Eq. 4-5

In practical systems, it is desired to have a convolution kernel, h(x,y), with as small an area as possible. A trade-off exists between reduction of computation time and reconstruction fidelity.

For ease in implementation it is also desirable to have a kernel which is separable, i.e. $h(x,y) = h(x)h(y)$. Figure 4-2 displays two common one-dimensional interpolation waveforms. The simplest is the "Gate" or "Pulse" function for sample-and-hold or zero order interpolation. First order or linear interpolation is realized with the Triangle function. The Cubic Spline Function of Figure 4-3 representing a 16

point weighted interpolation was developed for image resampling by TRW's Defense and Space Systems Group [37]. It used cubic splines to approximate the sinc waveform and assure zero slope on its end points. Other interpolation functions along with a discussion of interpolation error is presented by Pratt [36].

It should be noted that the implementation of Eq. 4-5 would not be as monstrous as it appears. In general, if the given image had multi-level amplitudes and h(x,y) was an infinite area impulse response filter (for exact reconstruction), for each output point M x N multiplications, additions, and evaluations of h(x,y) would be necessary. However, since G[m,n] can only have values of one or zero, the multiplications would not be necessary. Also, when finite impulse response interpolating filters are used, the number of computations would be much less than M x N since h(x,y) is zero over most of the image.

The Cognitive Information Processing Group's Image Processing System at M.I.T. employs a scheme developed for the Associated Press Wire Photo System for scaling multilevel monochromatic images as described by Troxel [38] with additional illustrations in [39]. Any separable interpolation function covering 2 by 2 picture elements can be used by prestoring it in table form. Separability permits serial processing. First, the scan lines in one direction are enlarged or reduced line by line, then using this new

semi-scaled set of picture elements as a source the same procedure is repeated in the other direction. The results are good for small changes in size. For gross enlargements however, the quality of the scaled image suffers from an apparent "blockiness". This undesirable artifact follows from the fact that the span of the unit impulse response of the effective two-dimensional filter is only 2 by 2 pels. The results are much more severe when this method is applied to binary images. This is illustrated in Figure 4-4 where Eq. 4-5 was effectively implemented on this system with h(x,y) being the separable triangle function (Figure 4-2(b)) to achieve bilinear interpolation. Note the blocky structure and rounding of corners.

Another approach to scaling multilevel digital images was presented by Jones [40]. It involves a method of obtaining an "optimum" interpolation function which minimizes the mean square error energy. A serious drawback of this procedure was that the optimal solution was peculiar to the image and scaling parameters for which it was designed.

4.2 TRANSFORM DOMAIN

For the sake of clarity, let us first consider the case of "enlarging" a one-dimensional discrete-space signal f[n] of finite duration N. The Fourier Transform of a discrete space signal is of course periodic, since it is represented by those values on the unit circle of the Z-Transform of the discrete signal. The Discrete Fourier Transform (DFT) of f[n] can be thought of as either:

1. One period of the Discrete Fourier Series (DFS) of the periodic signal

$$f_p[n] = \sum_{r=-\infty}^{\infty} f[n+rN]$$

where f[n] is one period.

The DFS of $f_p[n]$ is

$$F[v] = \sum_{n=1}^{N} f[n]\, e^{-j(2\pi/N)nv}$$

Eq. 4-6

The DFT is then those values of F[v] for v = 0,...,N-1. It should be noted that there can only be N distinct coefficients in the Fourier Series representation of such a periodic

sequence [41, page 89] since the exponential in the above expression is only unique for N values of v.

2. The N equispaced samples of the Z-Transform of the finite duration signal f[n] on the unit circle. (Or, N samples of one period of the Fourier Transform of f[n]).

In either case, use of the DFT implies both periodicity in space and in frequency.

It is of interest to determine exactly the continuous space signal represented by N DFT samples. Let us assume in the following one-dimensional discussion that

1. the number of samples, N, is odd
2. the distance between samples in continuous space is one unit.

Reference to the illustrations in Figure 4-5 should prove useful. An example of a seven sample signal f[n], is shown in (a) along with its DFT, F[v], in (b). It is desirable to translate this discrete representation to the continuous one shown in (c) and (d). This is accomplished by utilizing the delta function and the following formulas:

$$f_c(x) = \sum_{n=-\infty}^{\infty} f[n]\ \delta(x-n)$$

$$F_c(\omega) = \sum_{v=-\infty}^{\infty} \frac{2\pi}{N}\ F[v]\ \delta[v-\omega N/2\pi]$$

The gain factor of $2\pi/N$ in the latter expression can be verified by evaluating the average power in one period of $F_c(w)$ and F[v] by means of Parseval's Relation.

Since we assumed a sampling rate of one sample/unit, we apply an Ideal Low Pass Filter with a cutoff frequency at π to $F_c(w)$.

$$F_{cl}(\omega) = F_c(\omega) \cdot \text{ILPF} = \sum_{v=-(N-1)/2}^{(N-1)/2} \frac{2\pi}{N} F[v] \; \delta(v-\omega N/2\pi)$$

Eq. 4-7

which corresponds to an exact reconstruction in the spatial domain:

$$f_{cl}(x) = f_c(x) * \text{sinc}(\pi x)$$

Eq. 4-8

The $f_{cl}(x)$, $F_{cl}(w)$ transform pair are shown in Figure 4-5 (e) and (f).

The reconstruction expression in Eq. 4-8 is not a conceptually convenient one since it involves the convolution of two signals of infinite duration. We thus seek the ideal interpolation function, Q(x), that will yield $f_{cl}(x)$ when operated on the finite signal, designated as $f_{cN}(x)$ which is one period of $f_c(x)$ in the range $0 < x \leq N$. $f_{cl}(x)$ is the continuous Inverse Fourier Transform of $F_{cl}(w)$:

$$f_{cl}(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F_{cl}(\omega) \; e^{j\omega x} \; d\omega$$

Using the expression for $F_{cl}(w)$ in Eq. 4-7 this becomes

$$f_{cl}(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \sum_{v=-(N-1)/2}^{(N-1)/2} \frac{2\pi}{N} F[v]\ \delta(v-\omega N/2\pi)\ e^{j\omega x}\ d\omega$$

Eq. 4-9

F[v] can be expressed in terms of N samples of f[n] from Eq. 4-6. Thus $f_{cl}(x)$ can be completely determined from these samples:

$f_{cl}(x)$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty} \sum_{v=-(N-1)/2}^{(N-1)/2} \frac{N}{2\pi}\left(\sum_{n=1}^{N} f[n]e^{-j(2\pi/N)nv}\right)\delta(v-\omega N/2\pi)e^{j\omega x}\ d\omega$$

$$= \frac{1}{N}\sum_{v=-(N-1)/2}^{(N-1)/2}\sum_{n=1}^{N} f[n]\ e^{-j(2\pi v/N)n}\ e^{j(2\pi v/N)x}$$

Changing the order of summation,

$$= \frac{1}{N}\sum_{n=1}^{N} f[n] \sum_{v=-(N-1)/2}^{(N-1)/2} e^{j(2\pi v/N)(x-n)}$$

Eq. 4-10

With a change of variables

$$q = v + (N-1)/2$$

the last sum in Eq. 4-10 becomes

$$e^{-j\pi((N-1)/N)(x-n)} \sum_{q=0}^{N-1} e^{j(2\pi q/N)(x-n)}$$

$$= e^{-j\pi((N-1)/N)(x-n)} \frac{1-e^{j2\pi(x-n)}}{1-e^{j(2\pi/N)(x-n)}}$$

$$= \frac{e^{-j\pi(x-n)} - e^{j\pi(x-n)}}{e^{-j(\pi/N)(x-n)} - e^{j(\pi/N)(x-n)}}$$

$$= \frac{\sin(\pi(x-n))}{\sin((\pi/N)(x-n))}$$

Substituting back into Eq. 4-10, we have

$$f_{cl}(x) = \sum_{n=1}^{N} f[n] \frac{\sin(\pi(x-n))}{N\sin((\pi/N)(x-n))}$$

or, $$f_{cl}(x) = f_{cN}(x) * Q_N(x)$$

where $$f_{cN}(x) = \sum_{n=1}^{N} f[n]\, \delta(x-n)$$

and $$Q_N(x) = \frac{\sin(\pi x)}{N\sin((\pi/N)x)}$$

These functions are illustrated in Figure 4-6. Note that $Q_N(x)$ has an amplitude of one at the origin and zero at the other sample locations from 1 to N-1, as one would expect from an ideal interpolation function. Also note that it is periodic; this feature is responsible for potential "wrap around" error at the period boundaries of $f_{c1}(x)$. One such period, representing a complete one-dimensional image for $0<x\leq N$, is illustrated in Figure 4-6(c) to emphasize this problem.

The purpose of this exercise was to point out that sampling $f_{c1}(x)$, the image produced by convolving $Q_N(x)$ with the original samples, at any higher spatial rate than N samples/period (say L samples/period) will still yield only N samples in the DFT! This is because $f_{c1}(x)$ as derived from Eq. 4-9 only contains N frequency coeficients. However, the period of the DFT would now be L. This means that there would be L-N zeros between each old period of N values.

Let us now extrapolate to two dimensions. If one had available the DFT of an M by N image, an enlargement to a K by L image could be made by a very trivial manipulation. As pictured in Figure 4-7 simply augmenting the high frequency coeficients of the DFT with zeros would represent the transform of a finer sampled image. The inverse transform would yield K by L samples of a continuous image derived from the original samples convolved with the two-dimensional

interpolation function $Q_{MN}(x,y) = Q_N(x)Q_M(y)$. An illustration of this from our one-dimensional example is the transform pair given in Figure 4-8. To avoid wrap around error in these new samples, the perimeter of the original image should be padded with zeros before evaluating the DFT.

For reductions, one is confronted with increased aliasing. By a similar process, deleting the high frequency components of the DFT to obtain a smaller K by L DFT would represent samples of a reduced image.

Turning now from the Fourier Transform, it has been observed that the Mellin Transform is very well suited for scaling. G.C. Huang, et.al. [42] developed a method as a pattern recognition tool to eliminate the characteristic of size.

Allowing only imaginary frequencies, the two-dimensional Mellin Transform of f(x,y) is defined as follows:

$$F(j\omega_1, j\omega_2) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\, x^{j\omega_1 - 1}\, y^{j\omega_2 - 1}\, dx\, dy$$

If $f_2(x,y)$ is to be the scaled version of $f_1(x,y)$ where $a = s_x$ and $b = s_y$, the relation can be expressed

$$f_2(x,y) = f_1(ax,by)$$

Taking the Mellin Transform of both sides this becomes

$$F_2(j\omega_1, j\omega_2) = a^{-j\omega_1} b^{-j\omega_2} F_1(j\omega_1, j\omega_2)$$

It is important to note that the Magnitude of both sides are indistinquishable. The scaling information then is controlled only by the phase $a^{-j\omega_1} b^{-j\omega_2}$! However, to date there does not exist a fast efficient algorithm to perform this tranform digitally.

For reasons outlined in the next chapter, one should not approach the problem of scaling binary images from the linear processing point-of-view. Nonetheless, it is strongly suggested that future endeavors in digital scaling of multilevel images include investigations of ideas presented here.

In view of the non-linear scaling process introduced in this study, the purpose of this chapter was a heuristic one.

CHAPTER 5

NON-LINEAR DIGITAL IMAGE RESAMPLING

The techniques presented in the last chapter are (and can be) quite useful for scaling multilevel images. Inherent limitations in the linear approach, on the other hand, exist for binary images. The extremely coarse process of binary thresholding constitutes a very appreciable irreversible non-linear contribution. This is evidenced by the rippling present in even the "ideal" reconstructions as displayed in the examples of Figures 4-1(b) and 4-5(e). This again due to the unnaturally abrupt edges which are the essence of binary images.

Further degradations can be assumed upon the execution of Eq. 4-5 when a finite interpolation filter is used. Even when a moderately sized filter is facilitated considerable time is spent to algebraically determine the state of each new sample.

The Transform techniques suggested in section 4.2 would yield results close to the "ideal" at the untolerable price of extravagance in computation and storage. In storing a

discrete Transform even though one-half of the values are redundant and need not be stored, the values are complex. The big loss would be the fact that many bits of precision would be needed to represent the coeficient of a very dynamic transform, as opposed to the wealth of compressions presented in section 3.2 which yielded fractions of a bit per pel. The time required to take a large two-dimensional Transform is also inhibitive.

However, rather than a burden, this investigation considers this binary nature a blessing. For instead of values being treated numerically, they can be treated logically; a feature which is much more conducive to special hardware design. The non-linear solutions to the binary digital scaling problem in this chapter are faster and better than linear solutions.

As was pointed out in Chapter 2, the only attempts at high fidelity digital scaling of binary images required that the source images be subjected to an extensive preprocessing stage where approximating curves are fitted to the contours. Scaling then, is a trivial matter. This is so because all the real labor in such a process occurs in the pre-translation to a scale-insensitive code. Output quality is controlled only by the quality of the encoding processes.

Such a pre-encoding philosophy may be perfectly adequate for most phototypesetting applications, however, when a new logo or symbol is to be added to the image set the slow

pre-encoding processes must be performed. For applications such as facsimile transmission, such a scheme is not at all practical.

An important feature of the scaling process introduced in this chapter is the lack of need for special pre-encoding procedure.

5.1 COINCIDENT RESAMPLING

First the easier of two methods to be discussed is considered.

Coincident Resampling is defined to be a process of digital scaling in which samples on the new sampling grid, representing the scaled image, are coincident with original given samples. Reduced images then consist only of selected original samples while enlargements contain all of the original samples plus additional new samples in between. The states of these "in between" samples are determined by the neighborhood of original samples surrounding them.

Processing is broken up into the following phases:

1. Determination of the sequence of included, deleted, or added lines as dictated by the Horizontal Scale Factor, s_x.
2. Adjustment of included line lengths as dictated by the Vertical Scale Factor, s_y.
3. Horizontal enlargements only: The determination of interpolated (added) line edges.

In executing the first phase of this process it is useful to define a HORIZONTAL RESAMPLING SEQUENCE, $R_{sx}[j]$. This is an indexed sequence starting with j = 1 where the odd positions correspond to the number of original lines to be used. The meaning of the even positions is determined by

whether s_x is greater to or less than 1. For $s_x < 1$ and j odd, $R_{sx}[j]$ is negative and represents the number of original lines to be skipped. For $s_x > 1$ and j odd, $R_{sx}[j]$ is positive and is equal to the number and is equal to the number of new lines that need to be added (interpolated). The series is computed so that the average realized scale factor is as close to s_x as possible.

The sequence terminates when

$$\sum_j R_{sx}[j] = L$$

Consider the application of this sequencing phase where G[n,m] is the letter "A" shown in Figure 5-1. The first few values of $R_{sx}[j]$ are given below for four various horizontal scaling factors with the effective scale factor in parenthesis to their right.

j	$s_x = .19$	$s_x = .85$	$s_x = 1.6$	$s_x = 4.3$
1	1 (1.00)	1 (1.00)	1 (1.00)	1 (1.00)
2	-5 (.17)	-1 (.50)	1 (2.00)	4 (5.00)
3	1 (.29)	5 (.86)	1 (1.50)	1 (3.00)
4	-4 (.18)	-1 (.75)	1 (2.00)	3 (4.50)
5	1 (.25)	6 (.86)	2 (1.50)	1 (3.33)
6	-4 (.19)	-1 (.80)	1 (1.75)	3 (4.33)
7	1 (.24)	6 (.86)	2 (1.50)	1 (3.50)
8	-5 (.18)	-1 (.82)	1 (1.67)	4 (4.50)

The result of the above processes is pictured in Figure 5-2(a).

The next phase of Coincident Resampling involves stretching or shrinking the runs in each line. This process can be easily performed electronically by translating the one-dimensional spatial signals constituting each line to a time signal of binary amplitude, then sampling this signal at a rate determined by s_y. This controlled sampling rate can be realized digitally by a simple binary rate multiplier scheme.

Whether this vertical scaling (shown in Figure 5-2(b) for $s_y = s_x$) is performed by special purpose hardware or by multiplying the run lengths in software (as was the case for all simulations in this investigation), the scaling process is complete if $s_x \leq 1$. For the case of horizontal enlargements there still exists the task of "filling in the gaps". Here lies the third and only non-straight forward phase of Coincident Resampling.

In general, enlarging seems to be inherently more difficult than reduction. Reduction is simply a matter of seletively throwing away data whereas the enlarging process must attempt to extrapolate limited infromation.

The information to be extrapolated from, in this case, is arbitrarily chosen to be the endpoints of black runs in the already vertically scaled lines. Since all black runs

contribute two edge points, such pairs shall be designated $E_B[n,i]$, $E_T[n,i]$ as the Bottom and Top edge pel number of the i th black run in the n th line. I[n] will designate the number of runs in line n.

It was decided that the edges of the new lines falling in the gap between two original lines would be computed based only on those two lines. This choice was not made merely as a savings in complexity. Figure 5-3 demonstrates the limitation in using simple end point information, where the center of each pel of the bottom left foot of our digitized A represented as a dot. For our case (a) in which vertical end points are used it should be noted that the best possible salvage that can be performed when the magnitude of the slope of a contour region is less than one is by linear interpolation between end points. Higher order interpolation in such cases would be fruitless. Admittedly, in regions where the magnitude of the slope is greater than one, improvements could be made.

Fitting straight line segments between the endpoints of two images lines is a trivial matter when the two adjacent lines contain the same number of runs, i.e. I[n] = I[n+1]. The process then simply consists of creating the inbetween lines with I[n] black runs where end points are determined by linearly interpolating between bottoms, $E_B[n,i]$ and $E_B[n+1,i]$, and tops, $E_T[n,i]$ and $E_T[n+1,i]$, for i = 1,...,I[n]. Figure 5-4 illustrates this process.

Since pre-encoding to establish which endpoints are "connected" with which other endpoints is not permitted for reasons of time economy, the difficulty occurs when $I[n] \neq I[n+1]$. In such a case it becomes necessary to selectively eliminate edge points from the greater to equal the lesser. Figure 5-5 shows two simple examples of desired matching of endpoints resulting in an (a) effective elimination of a white run, and (b) effective elimination of a black run.

To realize this desired matching it was found to be convenient to segment the process into two. First, $I[n+1]-I[n]$ top edge points are deleted from the line with more runs by failing to include a top edge point of the line with less runs in their "decision region". The boundaries of these decision regions are equidistant between each top edgepoint in the line with more runs. Figure 5-6(a) illustrates this matching. The process is then repeated in a completely analogous fashion for matching bottom edge point as shown in Figure 5-6(b).

This algorithm works perfectly well for almost all cases where $I[n] \neq I[n+1]$. Figure 5-7(a) however, displays an example of where this approach fails. In this anomalous situation, two top edge points in the line of lesser runs fall in the same decision region. When this is detected, a correction is made by effectively raising the lower boundary of the decision region until the two points in question are separated yielding the desired matching of Figure 5-7(b).

Similarly, if two bottom edge points of the line with lesser runs fell into the same decision region, the upper bound of that region would be lowered so as to separate the two ponts.

It is possible to conceive of wild situations where these rules would fail to produce acceptable results, but this is highly unlikely in view of our definition of a binary image where low entropy is implied.

The output of this third phase of Coincident Resampling is displayed in Figure 5-2(c) completing our original example.

It should be pointed out that this division into three distinct phases (two for the case $s_x \leq 1$) as illustrated, does not imply that execution of the algorithm should be carried out in a similarly segmented manner. It is much more efficient to proceed in a comprehensive manner by performing the operation defined by the next element of $R_{sx}[j]$ one at a time. If j is odd, $R_{sx}[j]$ input lines are read in, scaled in the y direction, and released as output lines. For j even, $R_{sx}[j]$ lines are skipped or interpolated depending on its sign, and so forth. Other examples with various combinations of scaling factors are given in Figure 5-8.

5.2 RETROSPECTIVE RESAMPLING; THE TELESCOPING TEMPLATE

Clearly, the results of the latter method are not appealing for many choices of s_x and s_y. This is primarily due to the lack of breadth or scope in the information used in this asymmetric algorithm. Once again, turning to Figure 5-3, the limitations in using only endpoints of black runs for enlargements are apparent. By comparing (a) and (b) it can be seen that it makes a difference what one calls an endpoint--whether runs are recorded vertically or horizontally. The advantage or disadvantage of each case depends on whether the magnitude of the slope of the local contour considered is less than or greater than one. Also, although Coincident Resampling may be conceptually simple, the process of horizontal enlargements as described would be very difficult to realize with special purpose hardware.

It is for these and other shortcomings that the Telescoping Template method was devised. Retrospective Resampling was described in section 4.1 as the method of figuratively looking back to the continuous image as it existed before the original sampling, then sampling it. The term TELESCOPING is chosen because the bits in the template are arranged as telescoping squares or levels of increasing size, the number of which, p, corresponds to the order of scaling.

Because of differences in executing this method for enlargement and reductions, each will be addressed separately.

5.2.1 ENLARGEMENT

The first phase of a retrospective resampling procedure involves reconstructing the original continuous image. This is performed piece by piece in the most critical signal feature of this scaling scheme, the Assignment Area, A_{mn}. This can be thought of as a 1 by 1 continuous-space area associated with a given sample at G[m,n] as pictured in Figure 5-9. The idea is to select an "assignment rule" describing how this one block of the continuous image is to be painted in based on the neighborhood of pels surrounding A_{mn}. Exploiting the fact that our binary images as defined are low in entropy, within a finite neighborhood of pels a relatively small set of arrangements will occur. Exploiting the fact that our given pels are binary, they can be treated as bits of a code used to map a given arrangement to one of a finite set of assignment rules. Thus, non-linear reconstruction can be performed by storing a priori knowledge of the continuous contours and regions represented by discrete binary pels.

Decoding The Binary "Window"

The most important part of this surrounding "code" is the four samples comprising the corners of A_{mn}. This code segment, designated C_1^W, constitutes the first level or shell

in the center of a window of information. In general W_p (as shown in Figure 5-10) is a p th order Discrete-Space Binary Window about A_{mn} with 2p by 2p samples defined by:

$$W_p[i,j] = G[m-p+i,\ n-p+j]$$
$$\text{for } i = 1,\ldots,2p$$
$$j = 1,\ldots,2p,$$

and C_a^W is a Window Concentric Code, level a. Graphically, the composition of C_a^W can be described as the 8a-4 samples starting with that at $W_p[p-a+1,\ p-a+1]$, (or $G[m-a+1,\ n-a+1]$) and describing a concentric "shell" about A_{mn} in a counter-clockwise direction. Explicitly:

$$C_a^W[q] = \begin{cases} W_p[(p-a)+q,\ (p-a)+1] & \text{for } q = 1,\ldots,\ 2a-1 \\ W_p[(p-a)+2a,\ (p-a)-2a+1+q] & \text{for } q = 2a,\ldots,2(2a-1) \\ Wp[(p-a)+6a-1-q,\ (p-a)+2a] & \text{for } q = 4a-1,\ldots,3(2a-1) \\ W_p[(p-a)+1,\ (p-a)+8a-2-q] & \text{for } q = 6a-2,\ldots,4(2a-1) \end{cases}$$

where $\quad 1 \leq a \leq p$

A third order example of this "unwrapping" process is described visually in Figure 5-11. This is simply a convenient way to translate two-dimensional information into a meaningful one-dimensional form.

A very important short cut in this process stems from the assumption that the majority of a given binary image is

either solid black or solid white and that the occurrence of an edge is comparatively seldom. Thus, before any other processing is performed, the C_1^W code is individually unwrapped and tested for the case of four one bits or four zero bits. If all four bits are logical one or black, A_{mn} is assumed to be solid black and all new samples falling without area are assigned to one. A C_1^W code consisting of four zeros is assumed to correspond to an all white assignment area and this block is quickly skipped leaving its new samples unassigned or zero.

A resulting consequence of the latter is that thin lines with a thickness of one picture element will not be recognized for enlargement. (This will be remedied by techniques developed in the next section). Thus, for now, the following arrangements will be ignored (i.e. treated like all white blocks):

```
0 1      1 0
1 0      0 1
```

The remaining 12 of 16 possible arrangements of 4 bits correspond to potential edge locations necessitating further decoding.

A valuable aid in simplifying this decoding process lies in the recognition and removal of symmetric redundancy. For a given square window of picture elements, eight redundant permutations involving rotations and reflections are possible. This is illustrated in Figure 5-12 for the case of

a 2nd order window.

Let's first consider the removal of rotational redundancy. Derotating a given window can be completely directed by interpretation of the C_1^W code. Figure 5-13 summarizes all 16 possible C_1^W codes with their octal equivalent in parenthesis on their left. The 4 codes for which further decoding is not necessary are collected in part (a) of this illustration, while (b) categorizes the remaining 12 by pictorially defining 3 to be in "STANDARD FORM" of which the others are rotational permutations. Each permutation is associated with a Rotational Index, "RL", which indicates the number of Rotational shifts to the Left that need be executed to de-rotate the window, of which C_1^W is the first level, to STANDARD FORM. Other Window Concentric Codes, C_a^W, are circularly shifted to the left in quantum steps of (2a-1). It should now be clear that the reason the "unwrapping" processes shown in Figure 5-11 in the counter-clockwise direction was so that a rotation of a window W_p to the left corresponded to a circular shift of the Concentric Codes to the left.

Let us designate D_p as the p th ORDER DE-ROTATED WINDOW which is simply a permutation of W_p from which the rotational redundancy is removed. The mapping of W_p to D_p can be expressed

$$D_p[i,j] = W_p[r_I[i,j], r_J[i,j]]$$

$$\text{for } i,j = 1,\ldots,2p$$

where $r_I[i,j]$ and $r_J[i,j]$ represent some non-trivial operation on the indices i and j needed to describe this mapping to Standard Form. Also, let C_a^D be the De-rotated Concentric Code, level a. The positional relationship between C_a^D and D_p is exactly analogous to that between C_a^W and W_p. The mapping of C_a^W to C_a^D can be expressed

$$C_a^D[q] = C_a^W[r_Q[q]]$$

$$\text{for } q = 1,\ldots,8a-4$$

$$a = 1,\ldots,p$$

where the one to one function $r_Q[q]$ represents a trivial a operation on the index q, that of a circular shift to the left in RL steps of 2a-1 to remove permutation due to rotational redundancy. A 2nd order example of this de-rotation is illustrated in Figure 5-14.

An illustration of the mapping relationships between W_p, D_p, C_a^W, and C_a^D, is given in Figure 5-15. Vertical arrows designate the unwrapping process while horizontal arrows designate de-rotation. The dotted arrow represents a direct decoding from the given window W_p to the desired de-rotated concentric code, C_a^D. This dotted path is the one which can easily be performed in a parallel fashion in hardware as will be discussed in the next chapter.

The problem of redundancy due to reflection is a completely different issue. After a window has been de-rotated to Standard Form there still exists two possible types of reflection to deal with. These two types were first

elusively introduced in the two examples of Figure 5-12. Figure 5-16 explicitly illustrates (a) Horizontal Reflective Redundancy and the associated level 1 De-rotated Concentric Codes. Figure 5-16(c) demonstrates the fact that an arrangement can exist where the reflection is equivalent to itself.

It appears that there does not exist a fast convenient way to "de-reflect" a window nor a convenient way to define a "standard form". It is thus necessary to add reflective redundancy to the set of Templates except for those Templates which are reflectively symmetric.

A p th order Template shall be designated $T_p[h,i,j]$

$$\text{for } i = 1,\ldots,2p$$

$$j = 1,\ldots,2p$$

$$\text{where } 1 \leq h \leq H$$

Here, h is used to specify a particular Template out of a repertoire of H Discrete-Space Binary Templates to be compared with a given De-rotated Window D_p for a possible match. Associated with each of the H Templates is an assignment rule dictating how the Assignment area A_{mn} is to be partitioned. This repertoire of Templates, then, is the mechanism by which a priori knowledge of the association between continuous contours and discrete binary samples can be stored.

Analogous to the positional relationship that exists between W_p and C_a^W, and between D_p and C_a^D, T_p is also

unwrapped and stored as p Concentric Codes. C_a^T thus designates the TEMPLATE CONCENTRIC CODE, level a.

The Assignment Rule

We are now ready to closely investigate the nature of the ASSIGNMENT RULE, the selection of which for a particular Assignment Area A_{mn} was the goal of the above.

In general, for interpolation of degree t, the assignment rule describing which values of x and y are black for a given A_{mn} will be of the form

$$y \leq b_0+b_1x+b_2x^2+\ldots+b_tx^t \qquad \text{Eq. 5-1}$$

where $0 \leq x < 1$

$0 \leq y < 1$

Recalling that eight possible redundant permutations involving rotation and reflection can occur for a given arrangement of pels, the same must be true for an associated Assignment Rule. There are two ways to proceed.

First, all eight permuted assignment rules, as shown in Figure 5-17, could be stored. Or second (the much more efficient approach), the relative location of the new samples to be tested in A_{mn} by the given assignment rule could simply be appropriately rearranged. Figure 5-18 shows such a rearrangement in (a) for a given sample with localized coordinates (to be explicitly described later) at x_0, y_0. Figure 5-18(b) tabulates the effective values of these coordinates for various rotational and reflective

permutations. It is very important to note here that since these localized coordinates are between 0 and 1, the value $(1-x_0)$ and $(1-y_0)$ can simply be represented to an excellent approximation in hardware by the compliment of x_0 and y_0!

Figure 5-19 collectively illustrates how the continuous image to be reconstructed is segmented into MN 1 by 1 Assignment Areas. The corners of these subareas are the center points of the given picture elements. A dot at these locations will indicate a value of 1 or black while the absence of a dot implies a white picture element with a 0 value.

With the assumption that we are dealing primarily with black objects on a white background let us consider the consequences associated with two choices of boundary placement for some very simple straight edge examples. Figure 5-20(a) illustrates these two choices for a thin, 3 pel wide vertical bar. Since each picture element is considered to be 1 unit square, the boundaries labelled "IDEAL" are so named because they are separated by 3 units. The alternative of placing the boundaries through the center of edge pels are designated "PRACTICAL" and are the ones used in this study for four reasons:

1. Such a choice has the effect of reducing the weight or thinning black strokes; an idea which is consistent with the aesthetic considerations of section 3.3 for enlargement.

2. Upon observing Figure 5-20(b) it is clear that choosing the "IDEAL" boundary would in general necessitate a second equation:

$$x \leq b_0+b_1y+b_2y^2+\ldots+b_ty^t$$

to be intersected with Eq. 5-1 to provide Assignment Rules with cusps such as the corner as shown. In using the "Practical" boundary approach, cusps <u>always</u> occur on original pel centers eliminating the need for a second equation in the assignment rule.

3. Recall that a tremendous savings in processing time is gained with the assumption that an all zero C_1^W code corresponds to an empty or all white Assignment Area. Figure 5-20(c) shows that this assumption could not be taken advantage of if the "IDEAL" boundary is chosen.
4. The convenience of using the C_1^W code to direct de-rotation would not be possible for $C_1^W = 00$.

Resampling The Reconstruction

Before specific templates and assignment rules are presented for various orders of scaling, p, the development of non-linear retrospective resampling should be completed. As in the linear case, the second phase consists of overlaying a new sampling grid over the reconstructed image. The two real arrays X[ℓ] and Y[k] establishing the Horizontal

and Vertical Relative Positions of these new samples as defined by Eqs. 4-2 and 4-3 serve the same purpose here.

For a particular Assignment Area A_{mn}, it is desired to know exactly what discrete matrix of new samples, i.e. what subset of the output matrix $S[k,\ell]$, reside in it. This matrix is designated Z_{mn}. To aid in expressing Z_{mn} in terms of $S[k,\ell]$, it is useful to define two more parameters. V[n], the HORIZONTAL OCCUPATION NUMBER is equal to the number of elements $X[\ell]$ that satisfy the condition

$$n \leq X[\ell] < n+1$$

Similarly, U[m] is the VERTICAL OCCUPATION NUMBER equal to the number of elements Y[k] that satisfy

$$m \leq Y[k] < m+1$$

Thus, the matrix of new samples falling in A_{mn} is

$$Z_{mn}[u,v] = S\left[\left(\sum_{q=1}^{m-1} U[q]\right)+u, \left(\sum_{q=1}^{n-1} V[q]\right)+v\right]$$

$$\text{for } u = 1,\ldots,U[m]$$

$$v = 1,\ldots,V[u]$$

As was pointed out earlier, these samples are assigned normalized LOCAL COORDINATES within the continuous area A_{mn}. They can now be defined:

$$x_n[v] = X\left[\left(\sum_{q=1}^{n-1} V[q]\right) + v\right] - n$$

for $v = 1,\ldots,V[n]$

and

$$y_m[u] = Y\left[\left(\sum_{q=1}^{m-1} U[q]\right) + u\right] - m$$

for $u = 1,\ldots,U[m]$

A typical arrangement of samples Z_{mn}, along with associated Local Coordinates x_n and y_m in an Assignment Area A_{mn} is pictured in Figure 5-21.

Software Simulations

The simple special case of p = 0, or 0th order enlarging corresponding to interpolation with the gate function of Figure 4-2(a), can be simulated by this model. The parameters of concern can be expressed

$$W_0 = D_0 = C_0^W = C_0^D = G[m,n]$$

That is, the Window consists of only the principle original sample. There are only two Templates. $T_0 = C_0^T = 0$ is associated with an all white assignment Area; $T_0 = C_0^T = 1$ an all black assignment area.

As expected, the results of these "sample and hold" processes illustrated in Figure 5-22(b) are quite

unacceptable. This type of scaling is comparable to scaling by change of resolution. Figure 5-22(a) shows the relationship between the assignment rules and the picture element centers.

For all higher orders of scaling, $p \geq 1$, this investigation utilized only 1st degree assignment rules, i.e. Eq. 5-1 with t = 1. There are two reasons for this:

1. First degree polynominals are easy to realize in hardware
2. Compatabilities between adjacent assignment areas are manageable.

A complete graphic listing of the set of Assignment Rules is given in Appendix B. Notice that each successively higher order scaling procedure includes all rules of lesser orders.

For p = 1, the window only consists of the four center bits comprising the C_1^W code. Appendix C lists the Templates covering all possible cases. Figure 5-23 shows the results of first order enlarging. The Templates of Appendix C are especially important because they establish the default assignments for all higher order scaling.

Appendix D and E list the second and third order Template repertoires respectively. Note that the reflected templates (also a part of the stored list) are not given, and the use of the "Don't Care" bit designated as a dot in these templates. This allows for the recognition of many cases without the need of explicitly enumerating the associated

templates. Cases for which a template match is not found assumes the default assignment of the first order case, again reducing the number of pre-recoded templates needed. Results are displayed in Figure 5-24 for second order scaling and Figure 5-25 for third order scaling.

Other examples are given in Figure 5-26. Notice as the order increases the gentle curves smooth out while the corners and cusps remain crisp.

5.2.2 REDUCTION

Reduction is fundamentally "easier" than enlargement because information is being thrown out instead of extrapolated. The principle difference in the way the problem of reduction is approached stems from the fact that at most only one new sample will fall into a given Assignment Area. The management of boundaries within each assignment area is not really necessary; it is sufficient to simply decide whether an A_{mn} is "on" or "off".

This suggests 0th order scaling presented in the last section as one possible solution. This may be adequate for some applications but in general a more acceptable solution is sought for the following two reasons:

1. The weight of black strokes should be increased or thickened proportionately for the aesthetic and practical considerations of section 3.3. This is not the case with 0th order scaling.
2. The chance of a thin line being lost or unrecognized exists and increases as the scale factors decrease.

To clarify the latter point, consider a given image with a thin, one pel wide vertical line and a horizontal scale factor, $s_x = 1/3$. The probability of such a line being absent in the resulting scaled image is $1-s_x$ or 2/3.

Both of the above shortcomings would be alleviated by employing the linear process of Eq. 4-5 repeated here for convenience

$$S[k,\ell] = \underset{\text{THRESHOLD}}{\text{BINARY}} \left\{ \sum_{m=1}^{M} \sum_{n=1}^{N} G[m,n]\ h(X[\ell]-n, Y[k]-m) \right\}$$

with the convolution kernel, h(x,y), being an extended gate function shown in Figure 5-27. This kernel can be thought of as a new picture element. It also has the effect of a wider sample point spread response. The quantity in the brackets would always be some non-negative integer, and the BINARY THRESHOLD would be anywhere between 0 and 1 exclusively.

The effect of executing Eq. 4-5 can be easily achieved non-linearly in a process I call "BINARY CONVOLUTION" which does not require a single multiplication or addition. The process uses the same tools developed in the last section with a few modifications:

1. The decoding of a given window is skipped if the associated A_{mn} is not occupied by a new sample.
2. De-rotating is not performed.
3. Instead of searching the repertoire of templates for a match, the state of the new sample in A_{mn}, is simply equal to the logical OR of the bit by bit AND-ing of the associated window with a Master Reduction Template.

This Reduction Template is determined by the scale factors and can be thought of as samples of the Convolution Kernel in Figure 5-27. It is a d_H by d_V rectangle of black samples where

$$d_H = \text{Integer}(1/s_x + .5)$$

$$d_V = \text{Integer}(1/s_y + .5)$$

For example, if $s_x = 1/2$ and $s_y = 1/5$ the Master Reduction Template for a third order system would be

```
0 0 0 0 0 0
0 0 1 1 0 0          (d_H = 2, d_V = 5)
0 0 1 1 0 0
0 0 1 1 0 0
0 0 1 1 0 0
0 0 1 1 0 0
```

For $s_x = .31$, $s_y = .95$, and $p = 3$, it would be

```
0 0 0 0 0 0
0 0 0 0 0 0          (d_H = 3, d_V = 1)
0 0 0 0 0 0
0 1 1 1 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```

As can be seen, for a p th order system, the Reduction Template will saturate when s_x or s_y is less than $1/(2p+.5)$. In most cases the ill effects of using a saturated template are not appreciable. For those other cases either a step-and-repeat approach can be used or higher order scaling.

As with enlarging, the two reasons justifying this process in the beginning of this section are based on the

assumption that we are dealing with black objects on white background. When this is not the case or when special thickening effects are desired it is useful at times to directly define d_H and d_V and overide their above definition.

Software Simulations

To best understand the use of Binary Convolution to enhance the quality of digital reduction first consider the process of thickening a binary image with $s_x = s_y = 1.0$. The integers d_H and d_V describing the "Convolution Kernel" will be directly assigned and the output image will be resampled with the same grid as the original image.

Consider the thin line transistor symbol in Figure 5-28. Setting $d_H = d_V = 6$ yields the symmetrically thickened result of Figure 5-29. Asymmetric thickening will occur if the dimensions of the kernel are not equal; Figure 5-30 shows this for $d_H = 1$, $d_V = 6$, and figure 5-31 for $d_H = 6$, $d_V = 1$. Corresponding to these "Convolution Kernels" reductions with (a) $s_x = s_y = .1667$, (b) $s_x = .1667$, $s_y = 1$, and (c) $s_x = 1$, $s_y = .1667$ are shown in Figure 5-32. Other examples are given in Figure 5-33.

CHAPTER 6

IMPLEMENTATION, APPLICATION, and FINAL REMARKS

An original goal of this investigation was to devise a scaling scheme which is fast. This implies a method which is conducive to hardware realization. In developing the Telescoping Template as discussed in the last Chapter, a premium was placed on this objective. As will be seen, the decoding and resample assigning processes are easily translated into parallel hardware.

Figure 6-1 illustrates the design of the decoding phase for a third order system. The window, W_3, is assumed to move from the image bottom to top across 6 lines at a time. Six 6 bit shift registers would serve this purpose. For reduction, the 36 bit contents are simply logically ANDed bit for bit with the Master Convolution Kernel in a parallel fashion setting the ORed result to a new sample if it happens to occupy the current assignment area. For enlargements the process is much more involved.

The two bit Rotation Code, RL, indicating the one of four possible rotations as outlined in Figure 5-13(b) is decoded via combinational logic from the C_1^W code as shown in Figure 6-1. Also derived from this is one of three "ACTION"

Codes: All white, all black, or edge (further decoding needed). If the third non-trivial code is encountered, the decoding proceeds by de-rotating the window. Thirty-six 4-to-1 selectors are needed to realize this De-Rotated Window, D_3, as seen in the bottom of Figure 6-1. The explicit selection wiring is shown only for the $C_3^D[15]$ bit.

Once the De-rotated Window is achieved, the repertoire of Templates must be searched for a possible match. I found that a very efficient way of doing this is to subdivide the set of Templates into many very small linearly linked lists. Each list contains pointers to those templates which have the same bit values in the 9 locations in D_3 indicated by stars in Figure 6-1. Thus, these 9 bits can be used as an address to obtain the first pointer of such a list. These particular 9 locations were chosen because they most uniquely distinguish template arrangements and 2^9 or 512 is a very manageable number of addresses.

The Assignment Area A_{mn}, defined to be continuous-space need not be so since the local coordinates of a new sample within it have a finite precision. Assume for simplicity that only 3 bits of precision is used. The Assignment Area then would only need 2^3 by 2^3 or 64 discrete locations as illustrated in Figure 6-2. A 64 bit Random Access Memory could be used for such a purpose. The Assignment Rule obtained by decoding the current window could be loaded into this matrix by means of a binary rate multiplier scheme

similar to those used to draw straight line vectors on CRT tubes. An example of this for Assignment Rule #3 is shown in the illustration. The coordinates of new samples can then be used to address this memory to determine the state (1 or 0) of the elements in Zmn, the new sample matrix.

As was pointed out in the last chapter, since these local coordinates within A_{mn} are between 0 and 1, 1-x and 1-y can be approximated by $\bar{x}$ and $\bar{y}$. The table of de-rotated and de-reflected values in Figure 5-18(b) can thus be realized by a design shown in Figure 6-3.

Although the method of Coincident Resampling could also be easily implemented in hardware for reductions, the enlargement process is a different issue--particularly when the number of black runs in two adjacent lines are not equal. In general it could be said that the results of scaling by this means is inferior to those of a 3rd order Telescoping Template process. This is not to say that Coincident Resampling has no worth.

For slight enlargements and reductions, almost any method will suffice. In fact even sample-and-hold is tolerable as is evidenced by the many commercial digital phototypesetters which scale by changing resolution. Coincident Resampling is certainly an improvement over 0th order scaling. Since we have defined the direction of scan lines to be vertical, the interpolation scheme for

enlargements is that of part (a) of Figure 5-3. Upon observing the enlarged "Q" of Figure 5-8(a) or the enlarged "A" of Figure 5-2(c) it can be seen that for those edges were the magnitude of the slope is large, the quality of interpolation is extremely good. On the other hand, when the magnitude of the slope is small the resulting interpolation is terrible. This is primarily due to the fact that for a symmetric enlargement connecting vertical endpoints yields a slope which is always an integer. Thus for large slopes the calibration is fine and for small slopes very coarse --especially between 0 and 1.

Asymetric enlargement ($s_x \neq s_y$) is another story. In this case the array of possible slopes is $(s_y/s_x)i$, where i is an integer. For cases where s_x is considerably greater than s_y the problem of coarse calibration of small slopes is alleviated. This suggests a unique scheme for data compression in facsimile transmission or storage. Practically all redundancy reduction methods as summarized in section 3.2 operate in one dimension. One such method could be used to compress the scan lines of a given image. Additional compression by a factor of α could be gained by setting $s_x = \alpha < 1.0$ and $s_y = 1.0$ and reducing the image. Upon retrieving the image from storage or transmission Coincident Resampling could be used to enlarge the compressed image with $s_x = 1/\alpha$, $s_y = 1.0$. Since the enlargement process only operates on the immediate adjacent scan lines,

de-compression would be fast and simple. Figure 6-4 illustrates this process for $\alpha = 1/3$.

The Telescoping Template method does not have a "preferred direction" and does not benefit from asymmetric scaling. By viewing the Template repertoires in the Appendices, it can be seen that for a p th order system the magnitude of the largest slope attainable is p (besides vertical lines) while the magnitude of the smallest slope is 1/p (besides horizontal lines). For asymmetric scaling one limit would be improved while the other would worsen. So the Telescoping Template would not be well suited for a compression scheme as outlined above.

A major advantage of the Telescoping Template method is its ability to recognize and preserve discontinuities. An unfortunate consequence of this feature is its unforgiving nature with respect to image noise; a characteristic unaccommodating to facsimile transmissoin.

In section 5.2.1 it was stated that enlarging thin-line drawings would have to be handled later because our templates do not recognize lines of one pel thickness. This problem is solved by using Binary Convolution to first thicken the image. Figure 6-5 shows an example of this technique.

As for reduction, the value of Binary Convolution can be greater appreciated by closely examining the tiny images in Figures 5-8(e) and 5-32(a). In the case of Coincident Resampling where the process of selecting a given scan line

is a stochastic one, the vertical leads of the transistor symbol were completely missed. The process of effectively convolving an image with the filter of Figure 5-27 tends to remove high frequency content and reduces the chance of spurious noise in the resampling. Close comparison of the .15 reduction in Figure 5-33(b) to that of Figure 5-8(c) at 20 lines/inch reveals a more balanced scaling.

We have seen the utility of digital scaling of binary images in the realm of data compression and most certainly in phototypesetting. Other applications would include an aid in translation of images from a given I/O device to another with picture elements of a different aspect ratio and/or size.

In this study the details of the Telescoping Template method were worked out only up to a 3rd order system. I leave it as a challenge to develope a 4th order system. This should prove to be an uneasy task since the complexity and number of templates appears to grow exponentially with the order. While still preserving the corners and cusps, a 4th order system would yeild even more satisfying smoothing of curves. The beauty of the Telescoping Template is that while increasing the order of scaling may be complex in design, when implemented with the parallel hardware plan above there is essentially no loss of processing speed.

I L L U S T R A T I O N S

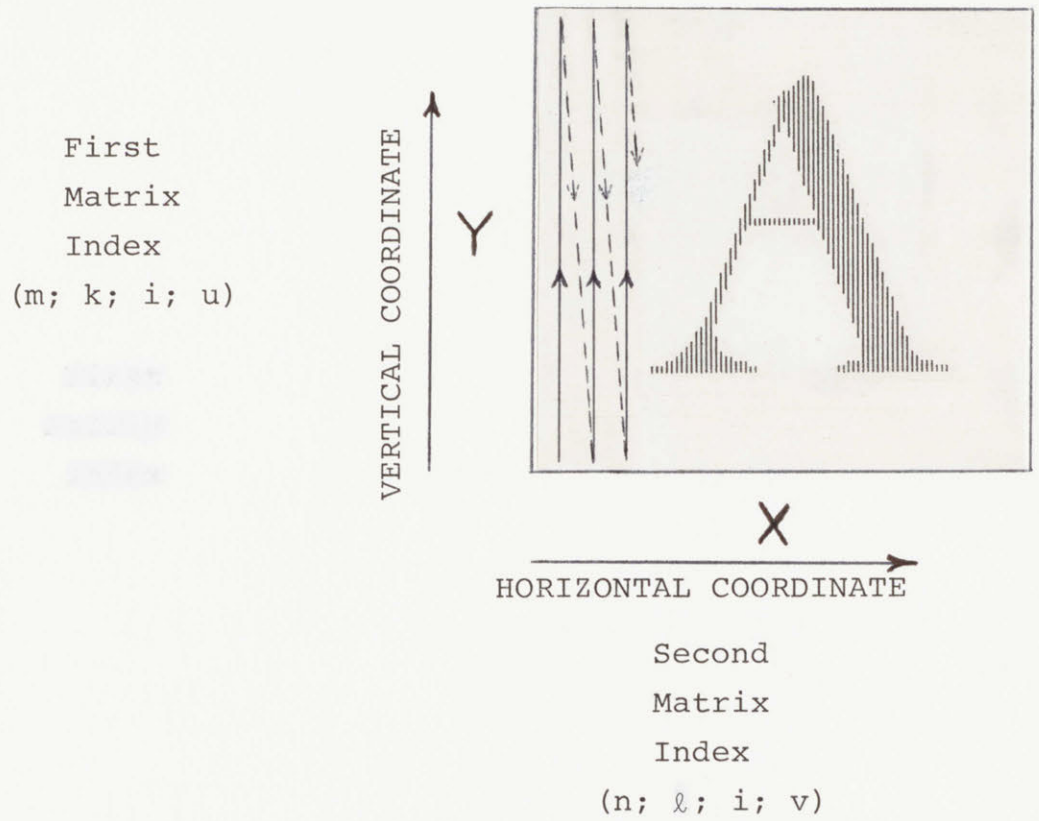


Figure 1-1. Graphical definition of indexing conventions.

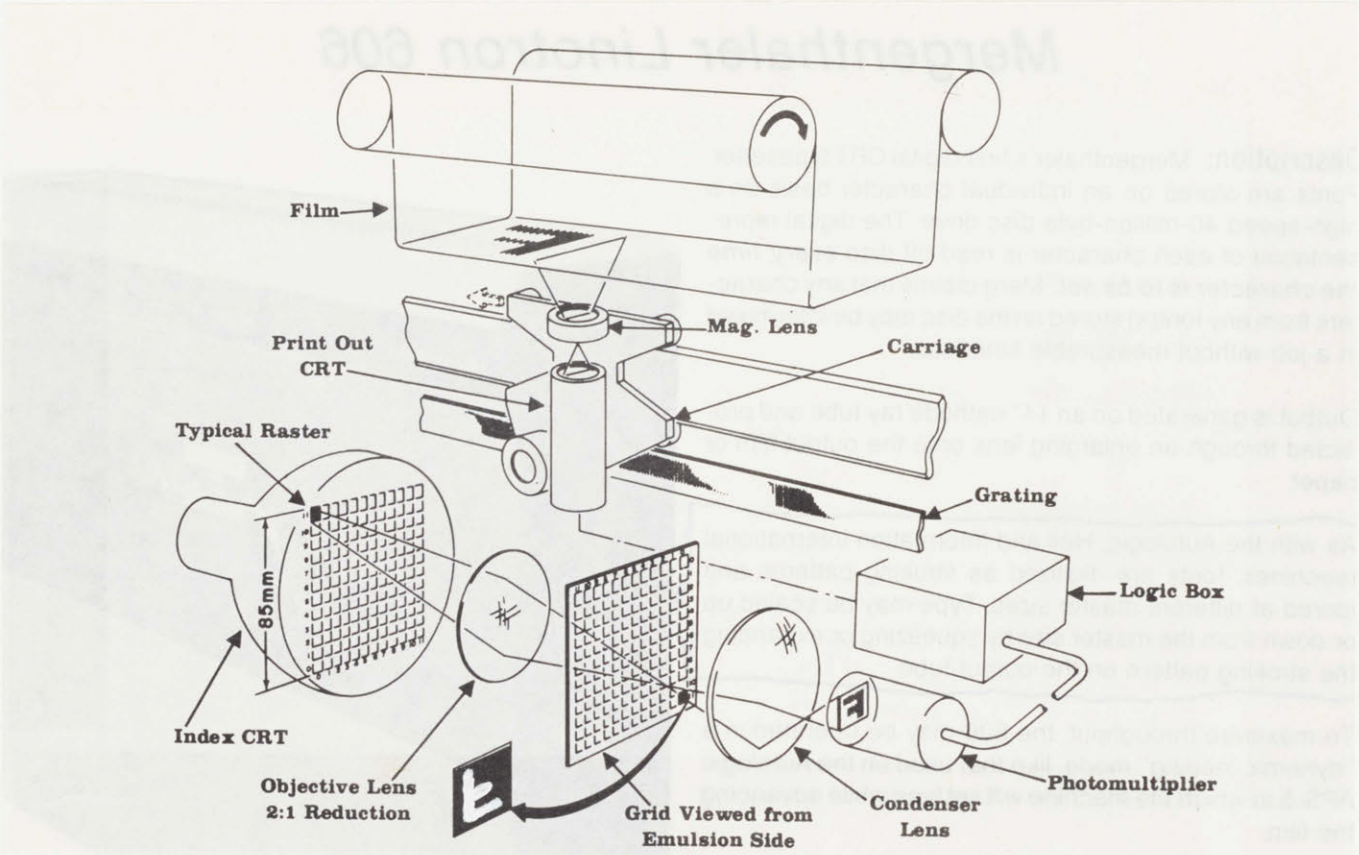
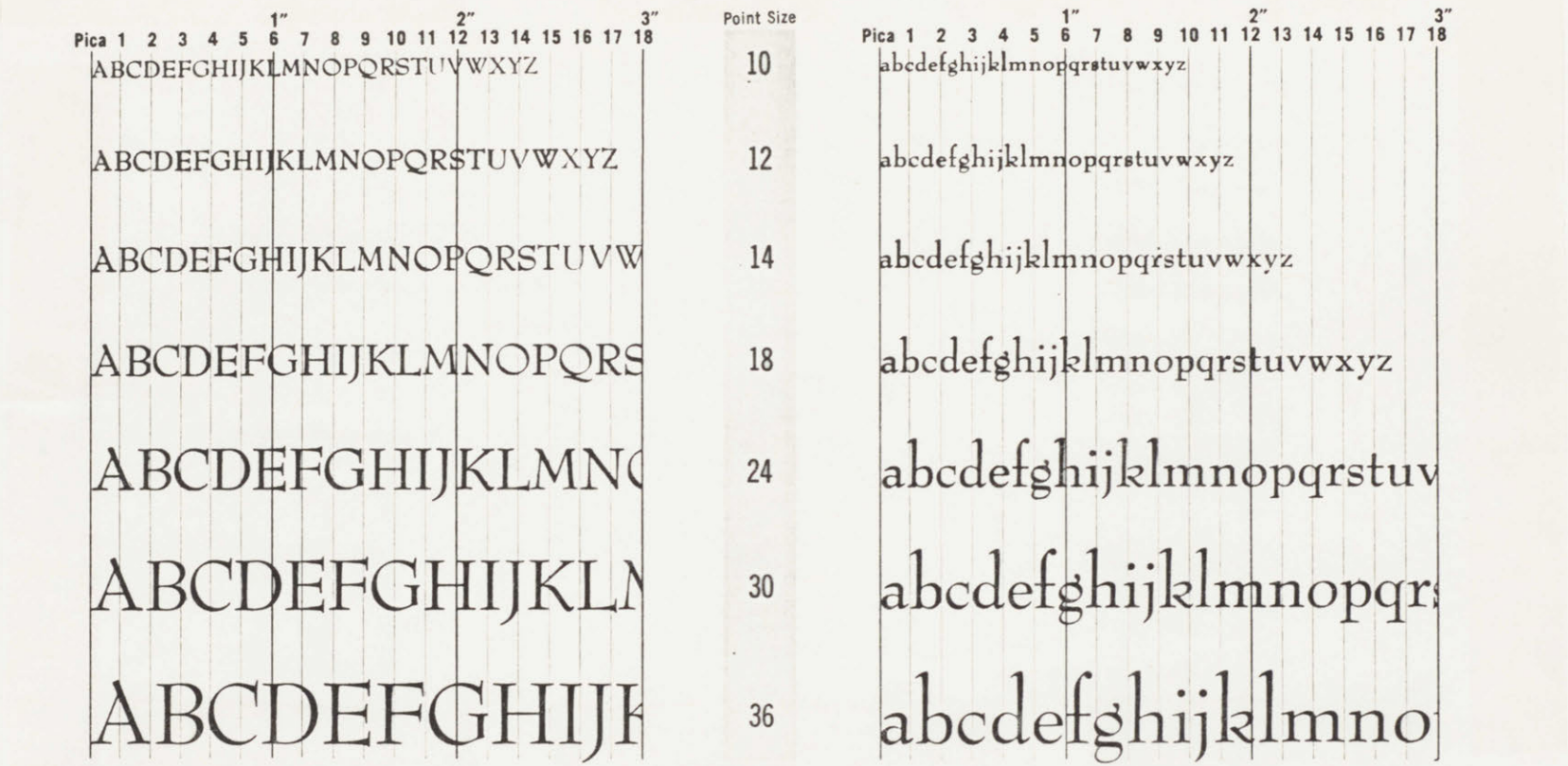


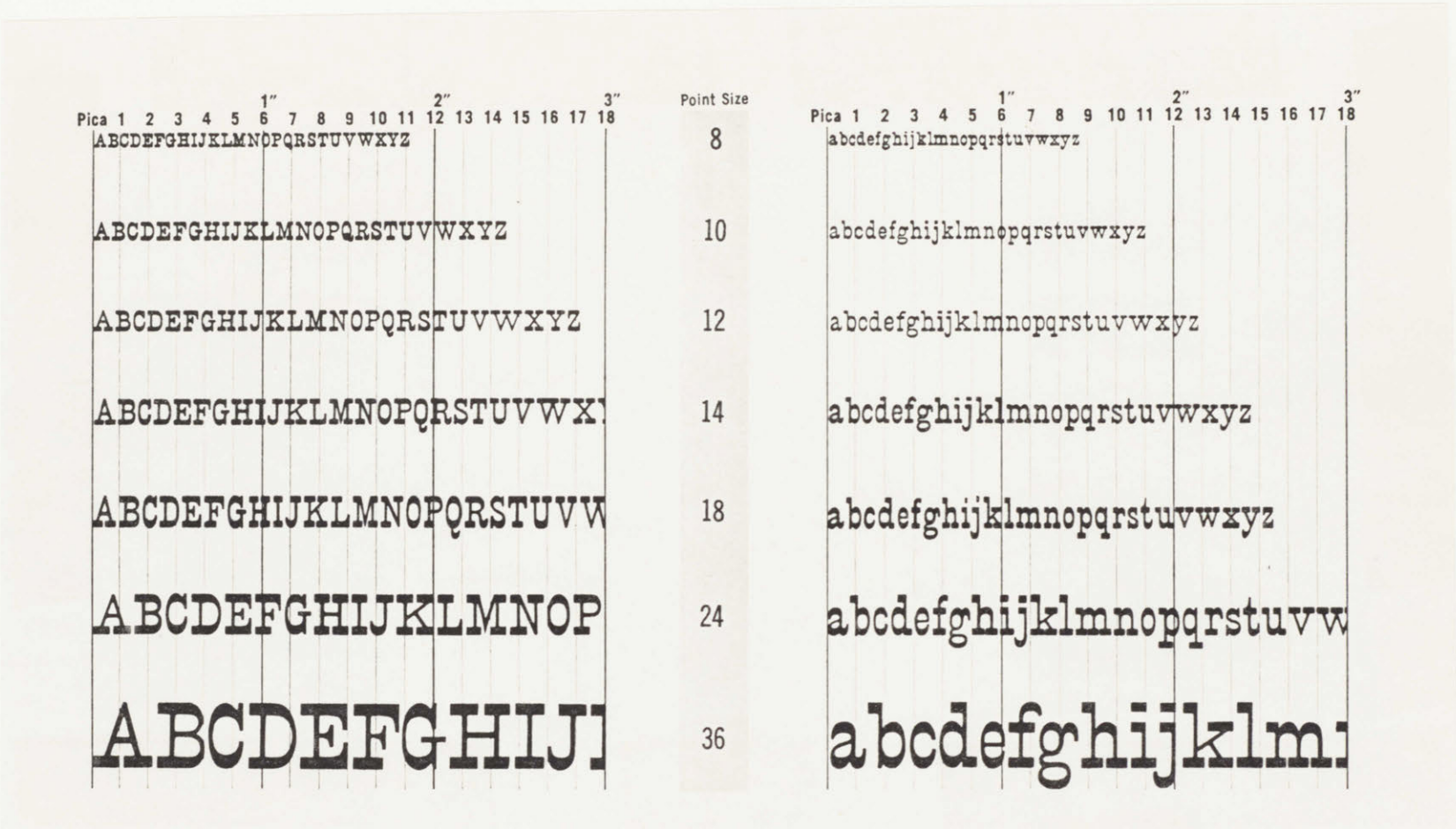
Figure 2-1. Typical Second Generation Phototypesetter: The Linotron 303. (Copied from [3, page 19]).



(a) Bernard Modern Roman

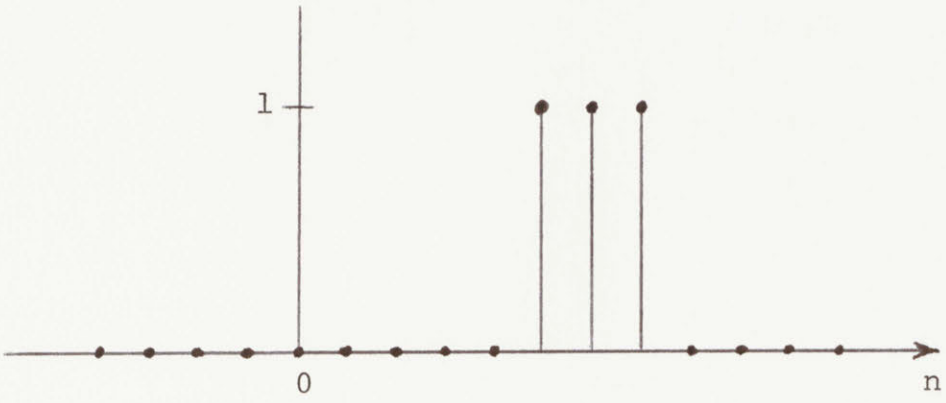
(Copied from [33, page 44])

Figure 3-1. Type design adjustments for various sizes.

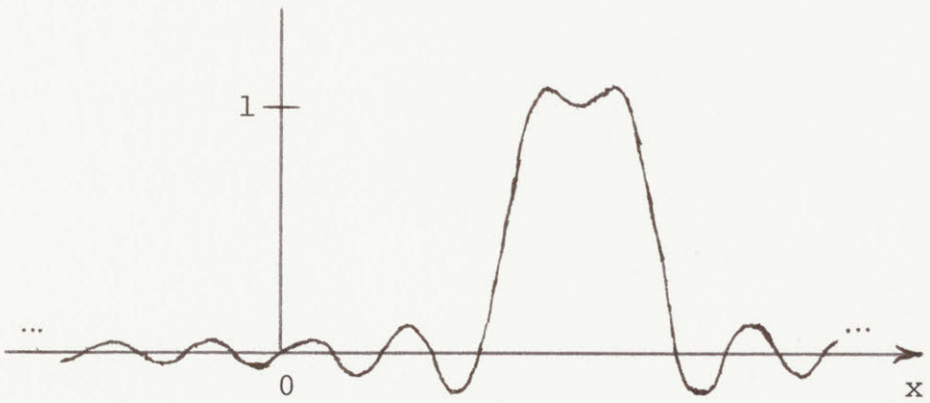


3-1(b). P.T. Barnum

(Copied from [33, page 38])

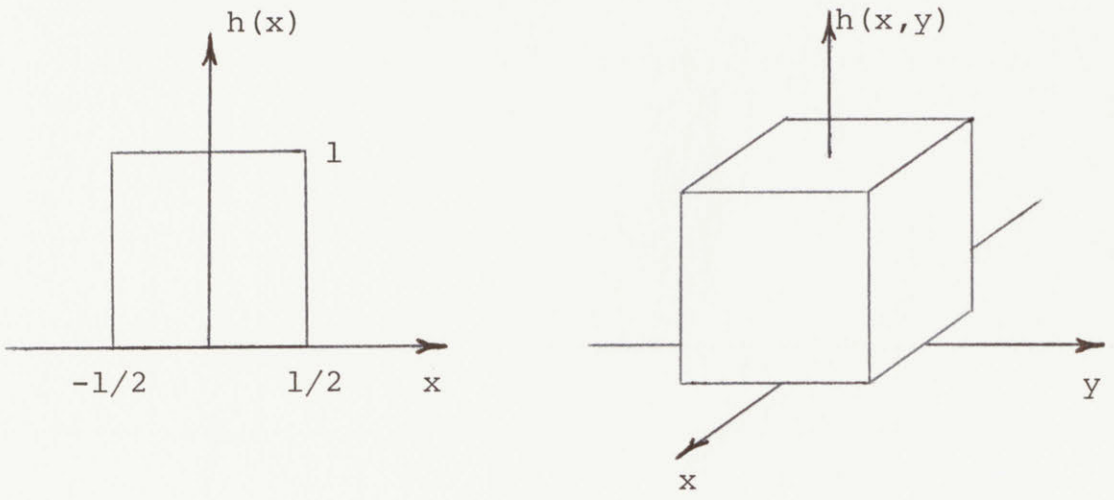


(a) One-dimensional binary image

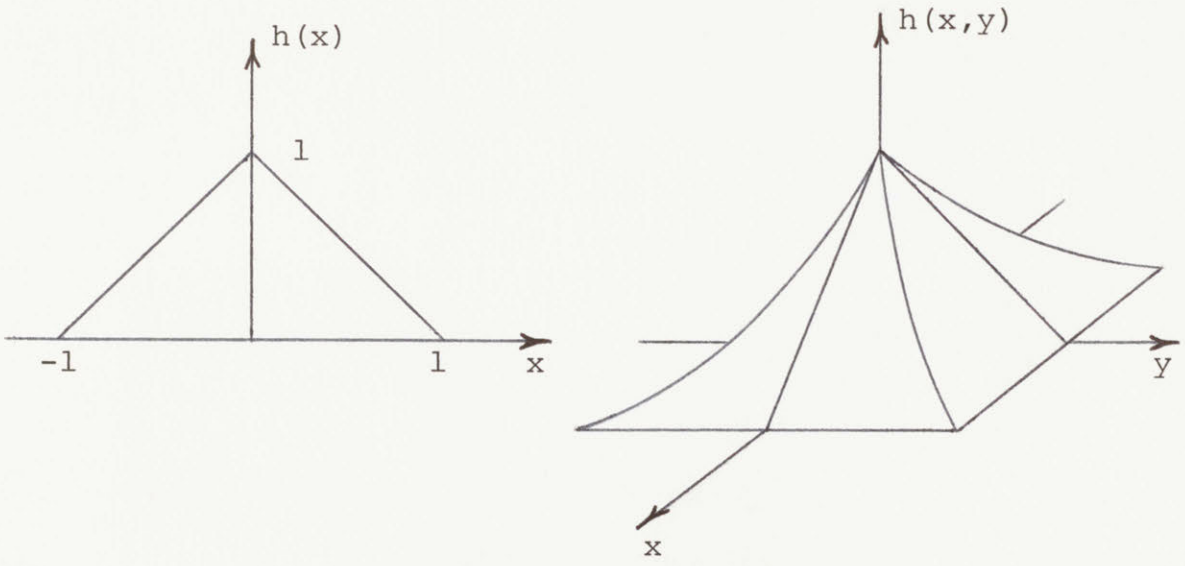


(b) Ideal continuous reconstruction from the spatial samples of (a)

Figure 4-1. Example of rippling due to the sharp edges inherent in a binary signal.

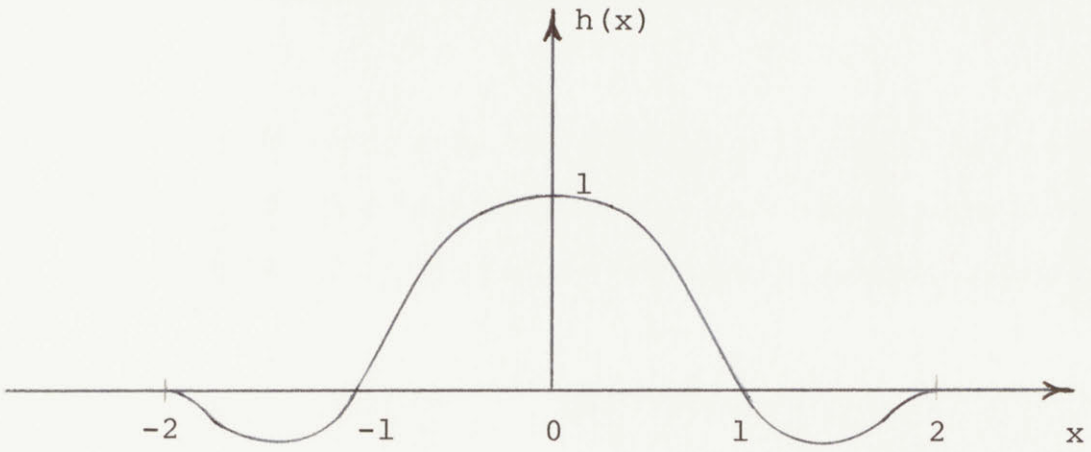


(a) Gate function



(b) Triangle function

Figure 4-2. Common separable finite interpolation waveforms.



$$h(x) = \begin{cases} 1 - 2|x|^2 + |x|^3 & \text{for } |x|<1 \\ 4 - 8|x| + 5|x|^2 - |x|^3 & \text{for } 1\leq|x|<2 \\ 0 & \text{otherwise} \end{cases}$$

Figure 4-3. TRW's Cubic Spline interpolation waveform.

(a) Example 1

Figure 4-4. Enlargement via eq. 4-5 with the triangle function of Figure 4-2(b). $s_x = s_y = 5.5$, resolution = 100 lines/inch

4-4(b) Example 2

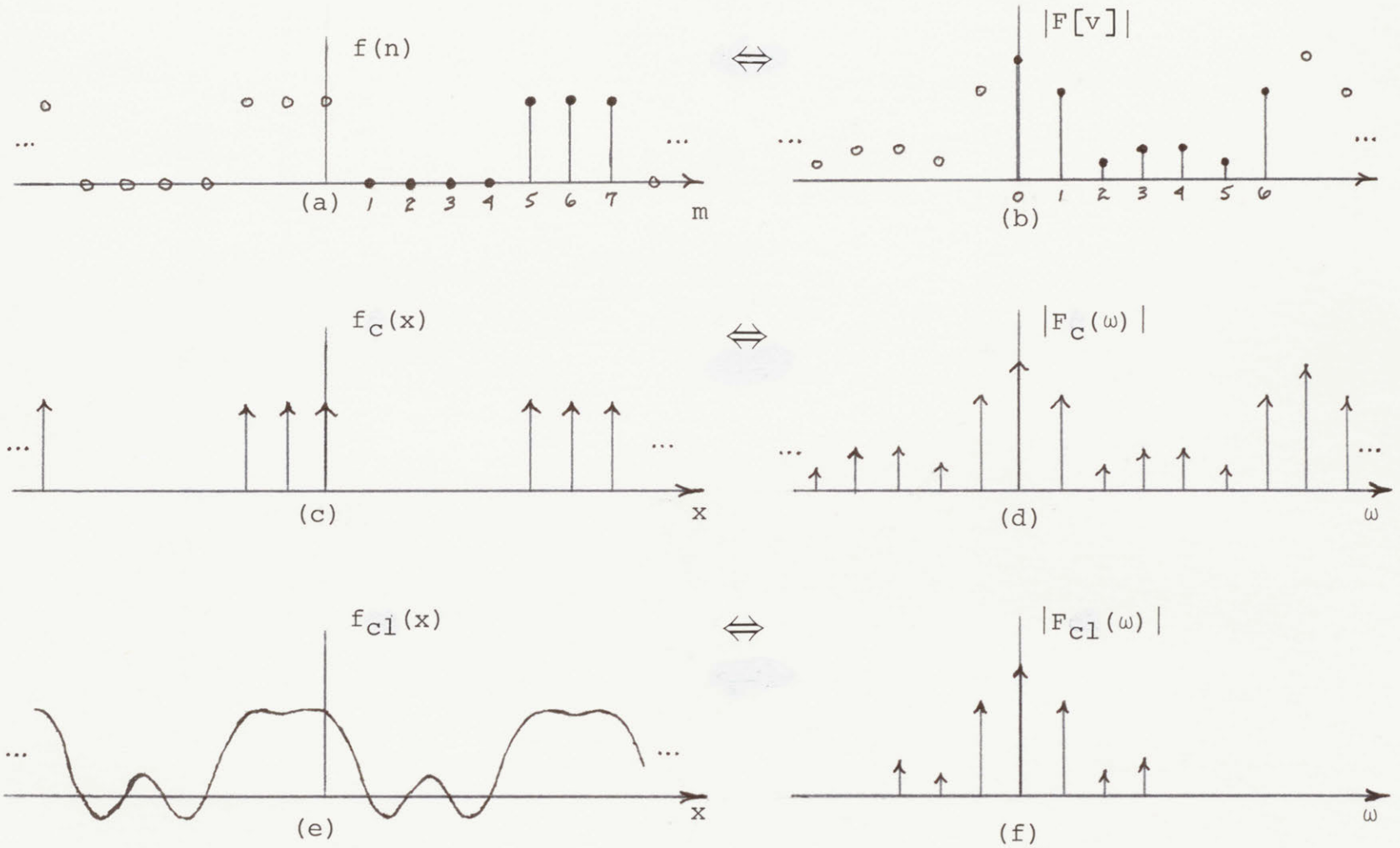


Figure 4-5. Transform pairs illustrating the ideal reconstruction of a continuous signal from its DFT samples.

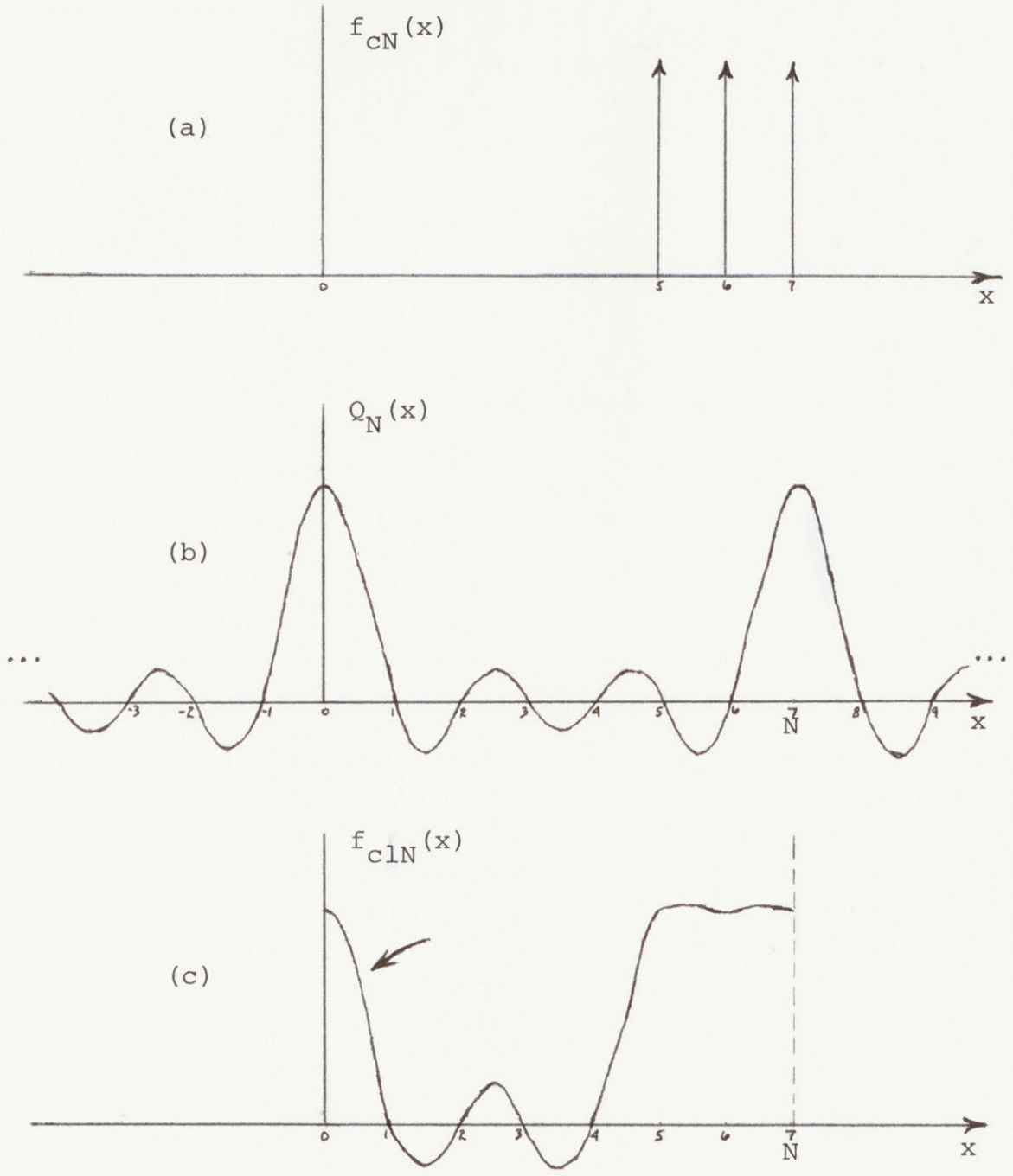
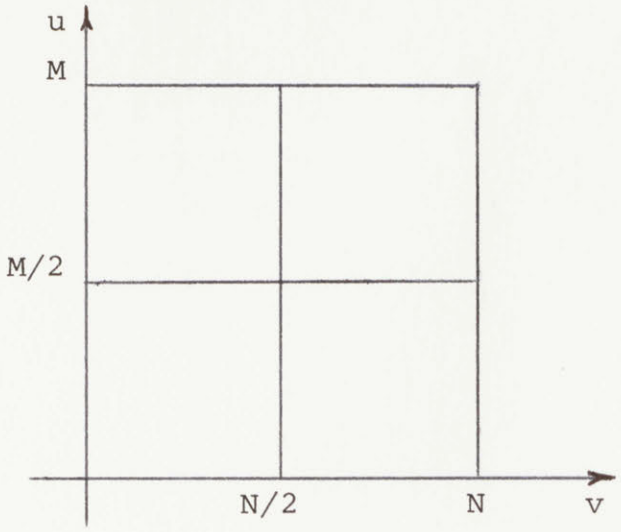
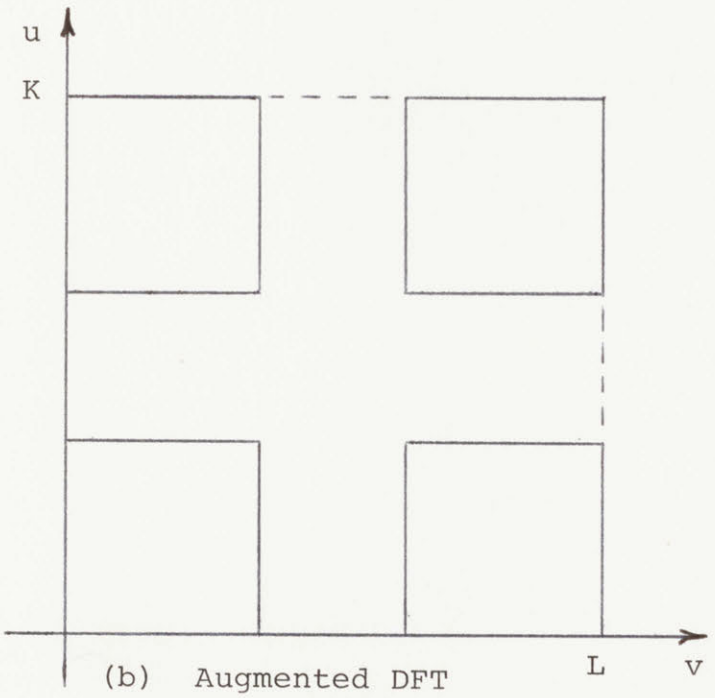


Figure 4-6. One-dimensional example of wrap around error.

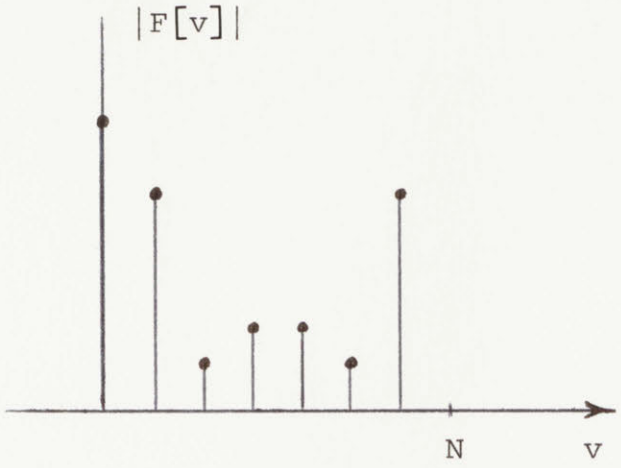


(a) Original DFT

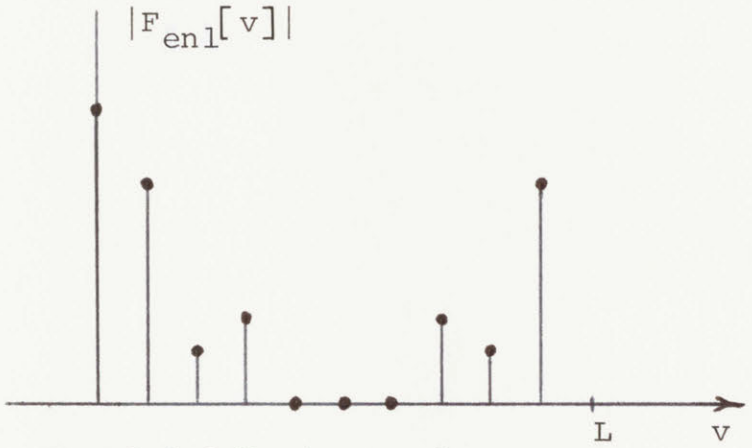


(b) Augmented DFT

Figure 4-7. Enlargement by Transform manipulation.



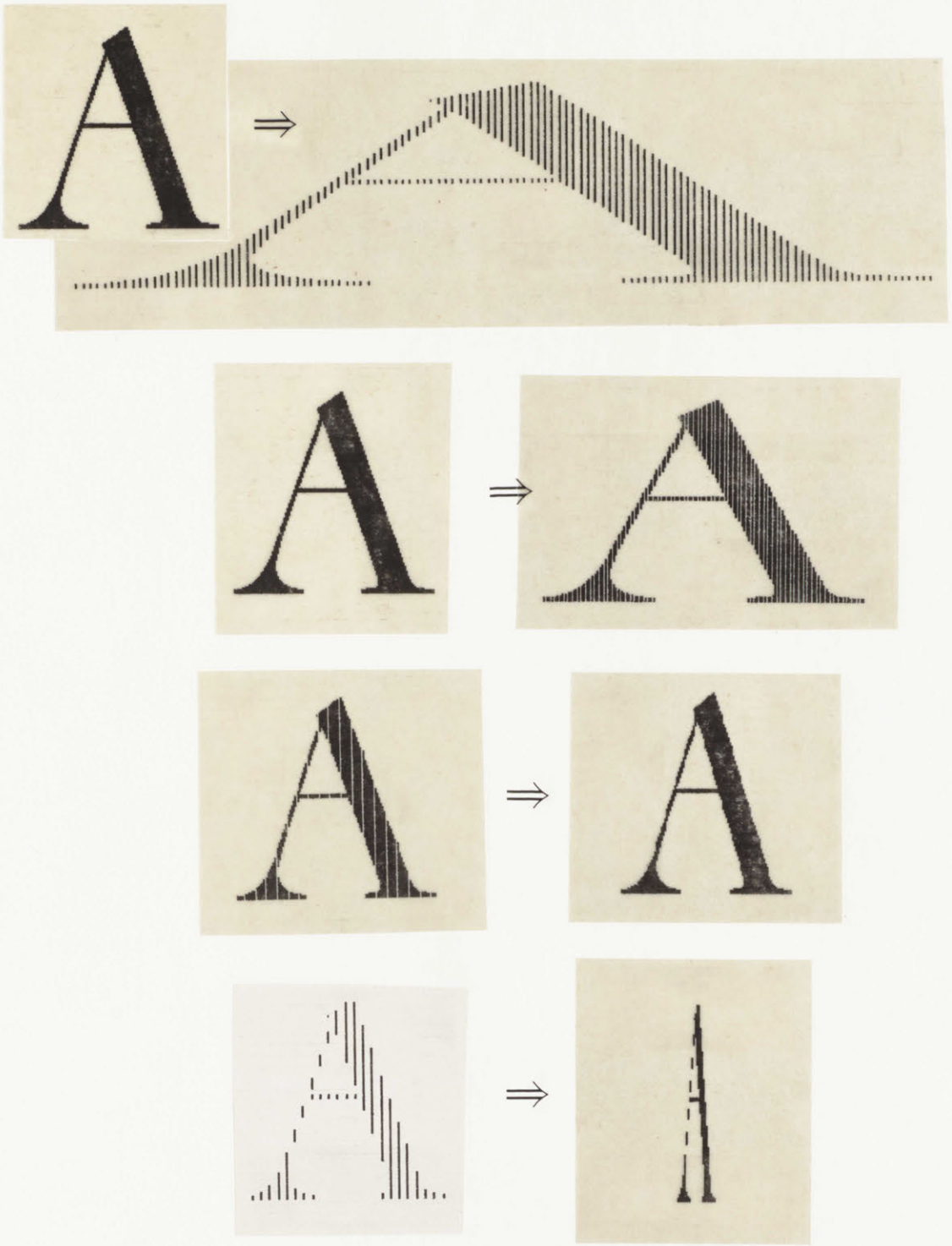
(a) Original DFT



(b) DFT of 1 3/7 times enlargement

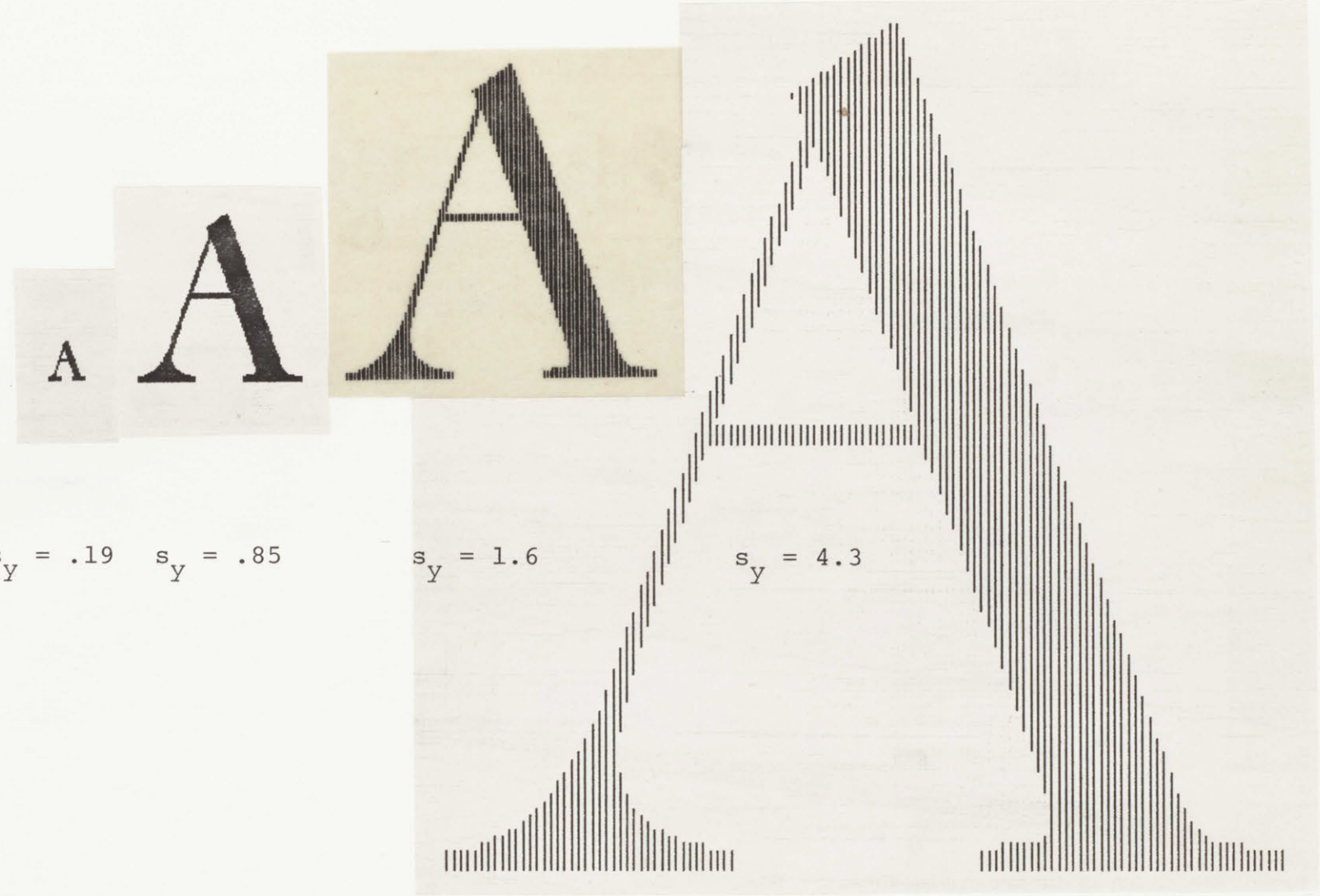
Figure 4-8. One-dimensional example of Transform manipulation.

Figure 5-1. Given Image at various resolutions.



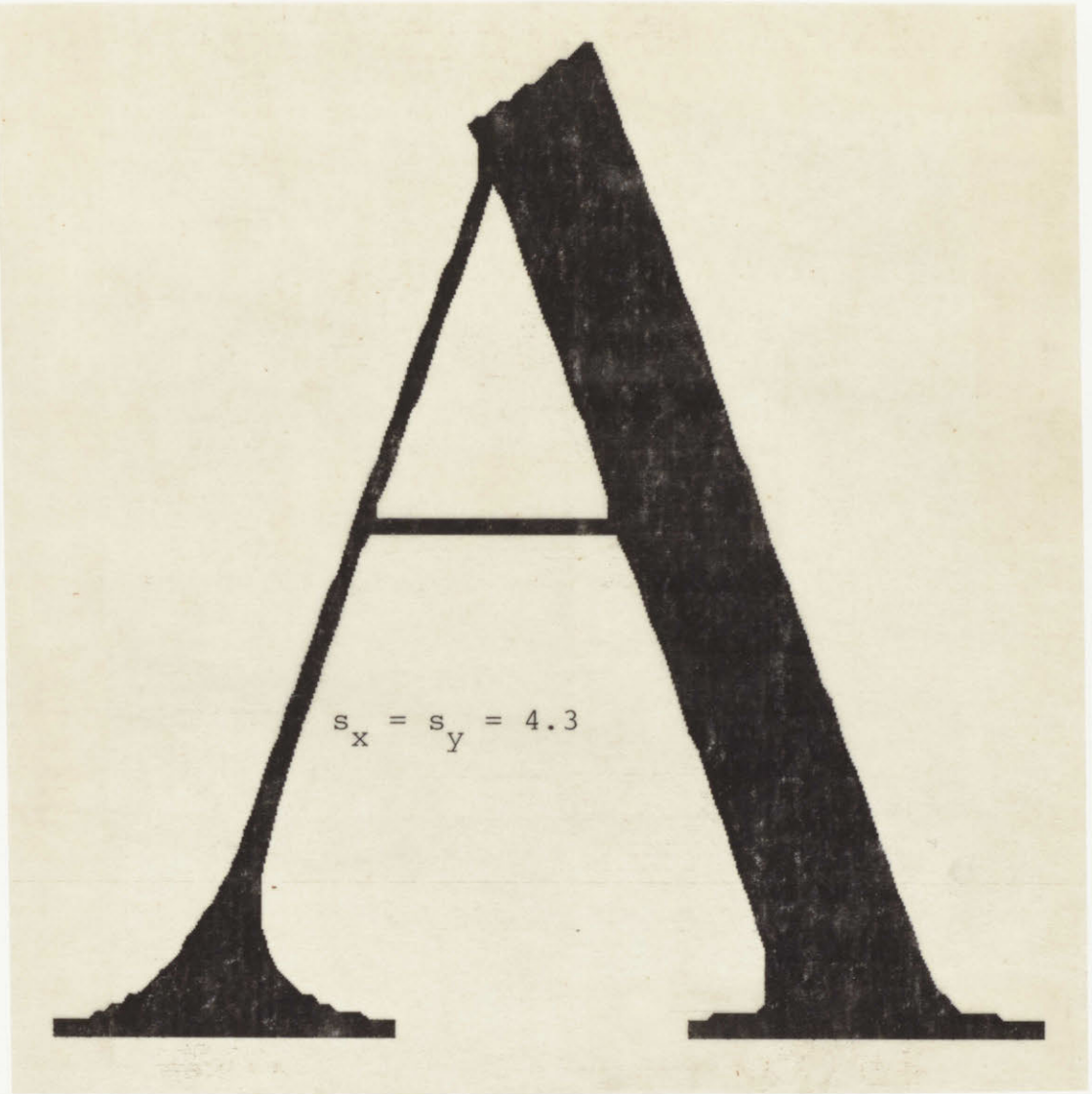
(a) Phase 1, Horizontal Sequencing

Figure 5-2. The Coincident Resampling Process.
resolution = 100 lines/inch



5-2(b). Phase 2, Vertical Scaling ($s_y = s_x$ in this example)

$s_x = s_y = 1.6$



$s_x = s_y = 4.3$

5-2(c). Endpoint Interpolation (s_x >1 only).

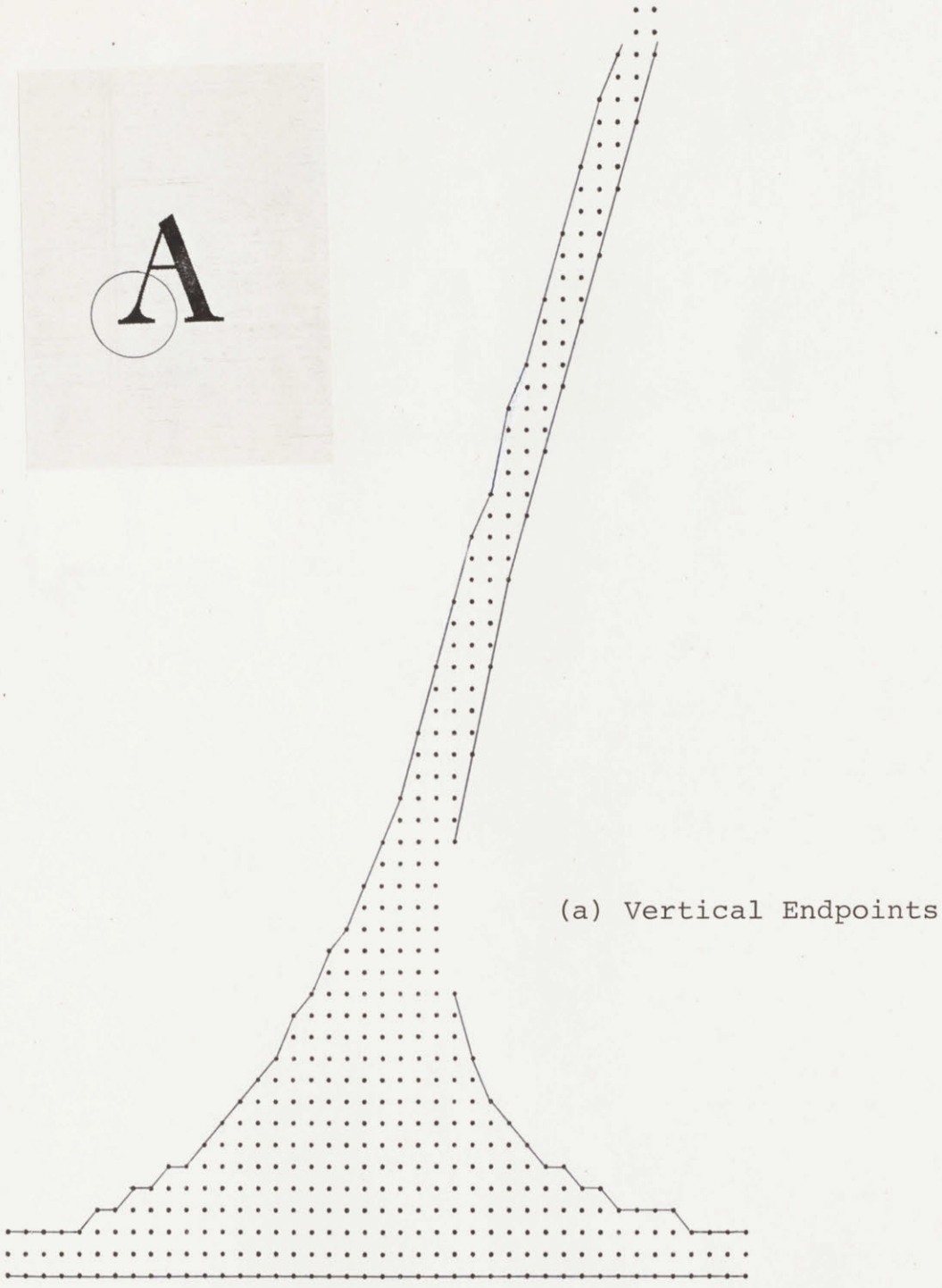
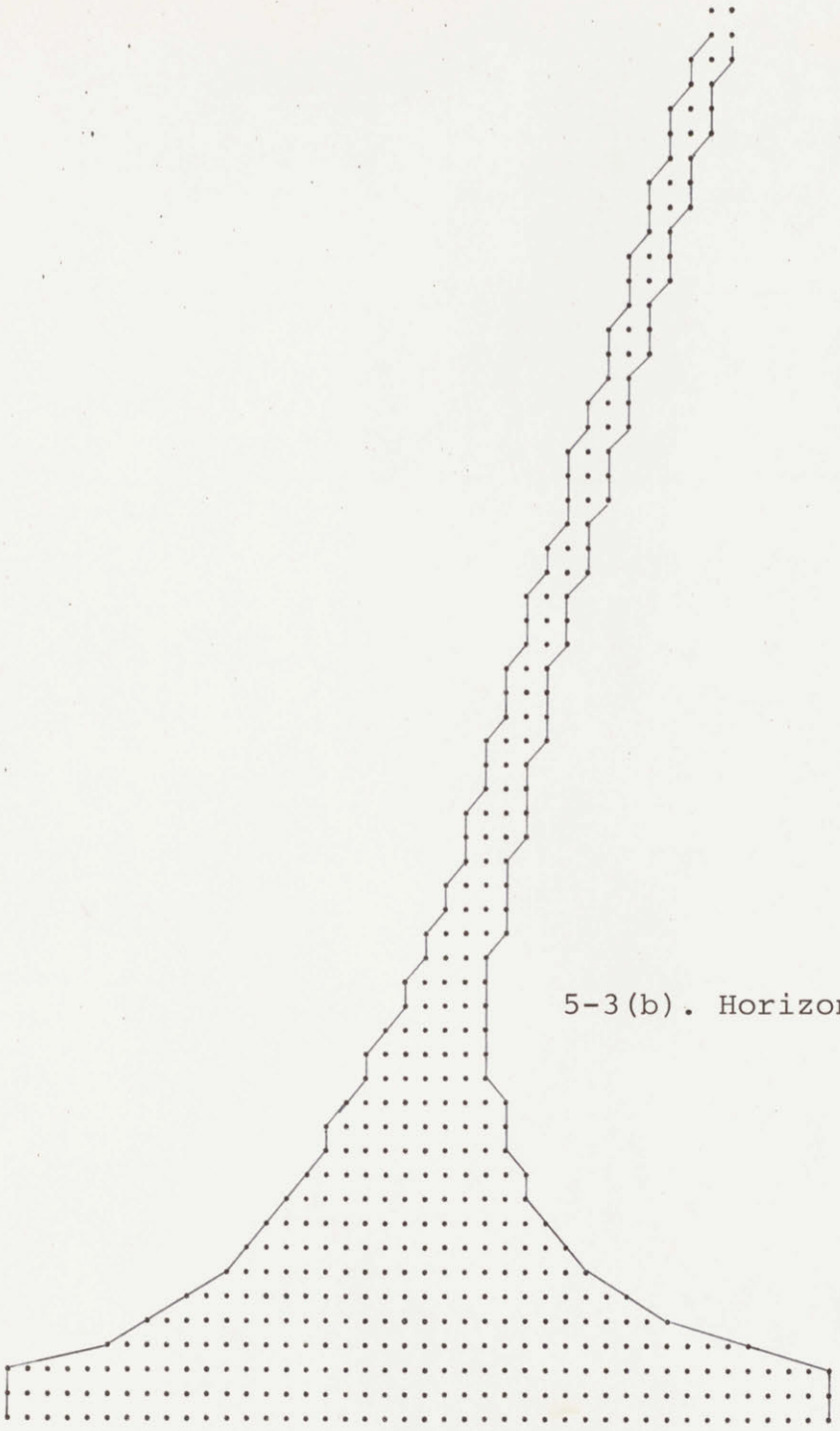


Figure 5-3. The limitations in using only the endpoints of black runs.



5-3(b). Horizontal Endpoint.

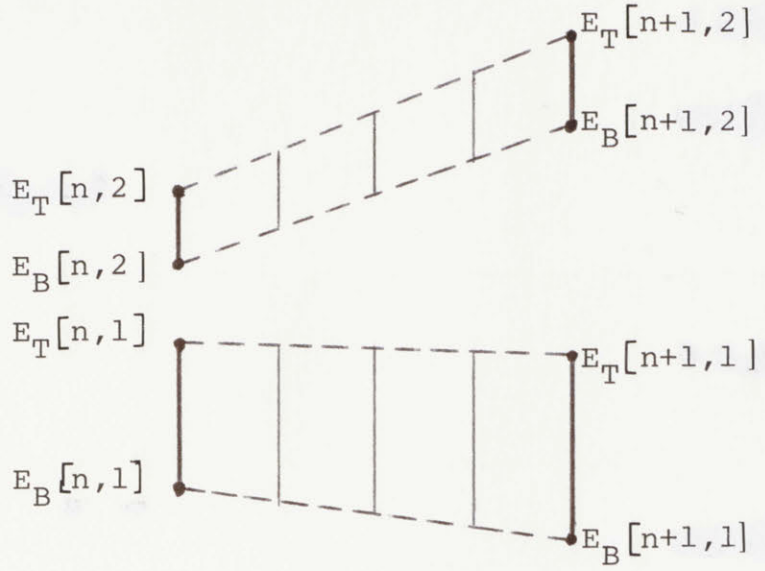
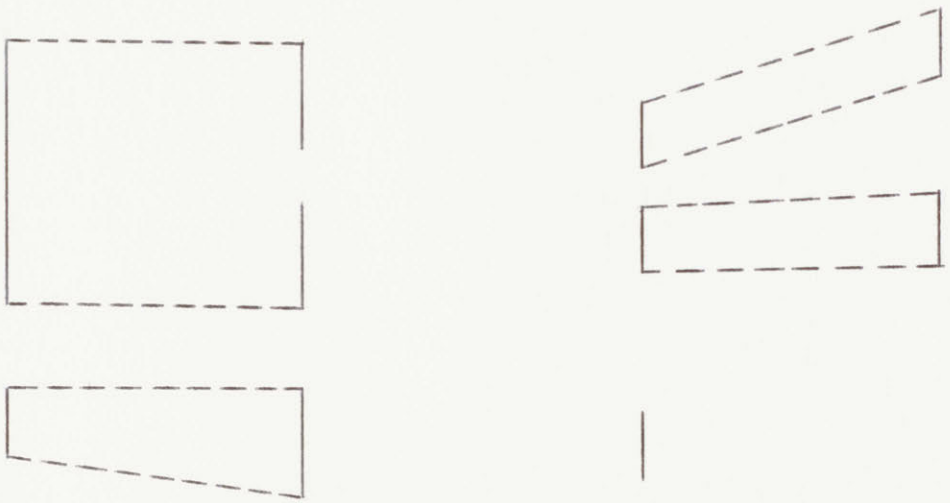


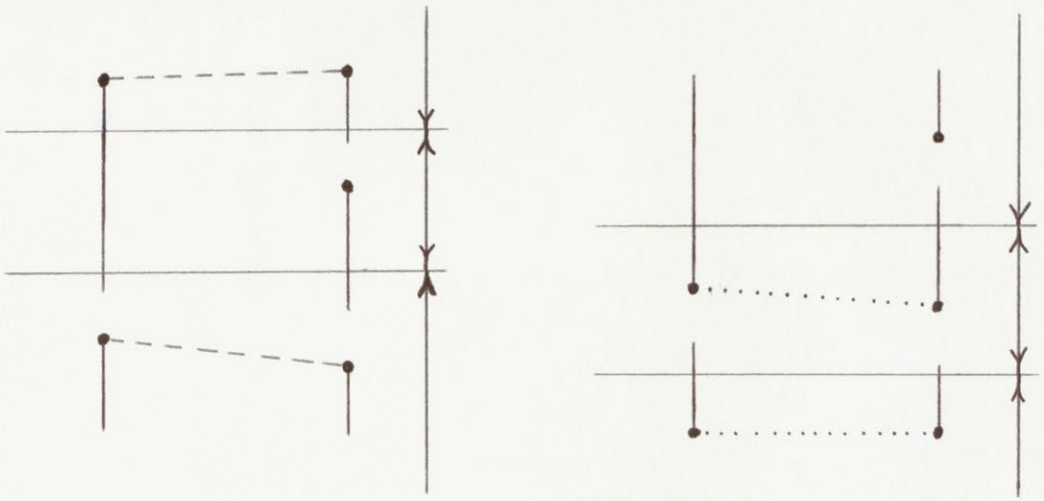
Figure 5-4. Interpolating three new image lines.



(a) Elimination of a white run.

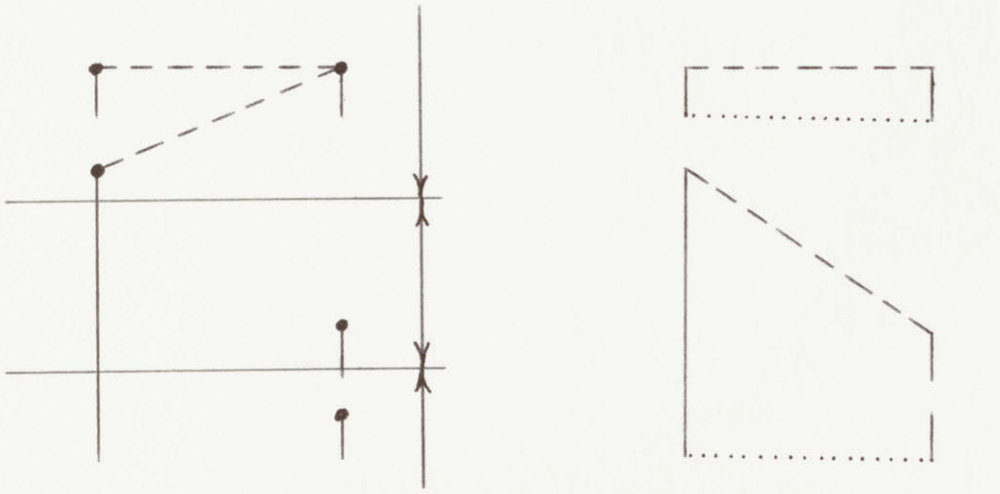
(b) Elimination of a black run.

Figure 5-5. Examples of desired connection of endpoints where $I[n] \neq I[n+1]$.



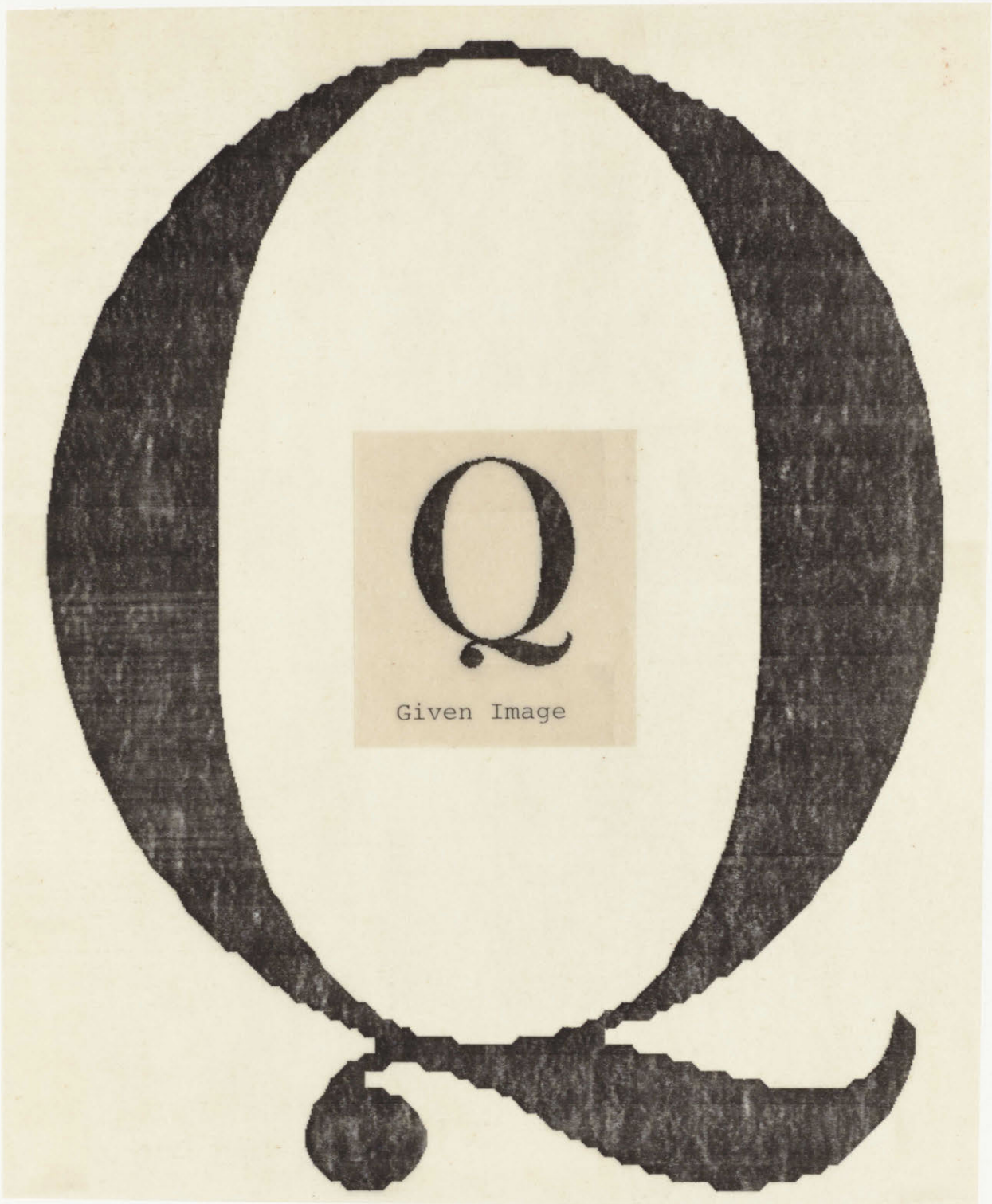
(a) Matching tops (b) Matching bottoms

Figure 5-6. Use of "Decision Regions"



(a) Erroneous matching of tops. (b) Corrected matching

Figure 5-7. Example where Decision Regions fail.



$s_x = s_y = 5.5$

(a) Example 1, @100 lines/inch

Figure 5-8. Examples of Coincident Resampling.

Given Image

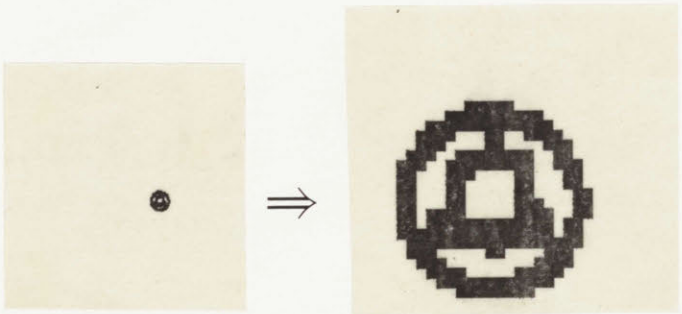
$s_x = 7.0,\ s_y = 5.0$

5-8(b). Coincident Resampling, example 2.

resolution = 200 lines/inch

Given Image

$s_x = s_y = .4$



(@ 20 lines/inch)

$s_x = s_y = .15$

$s_x = s_y = 6.0$

5-8(c). Coincident Resampling, example 3.
resolution = 200 lines/inch

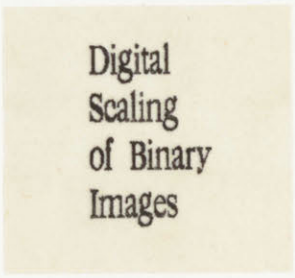
Given Image

$s_x = .33$

$s_y = .25$

$s_x = .24$

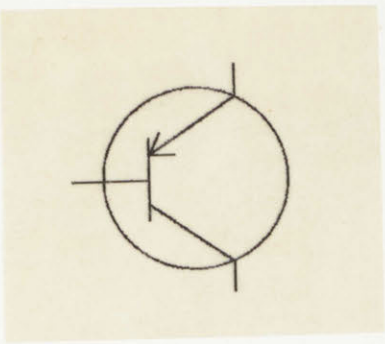
$s_y = .18$



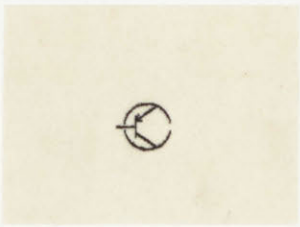
$s_x = .18$

$s_y = .24$

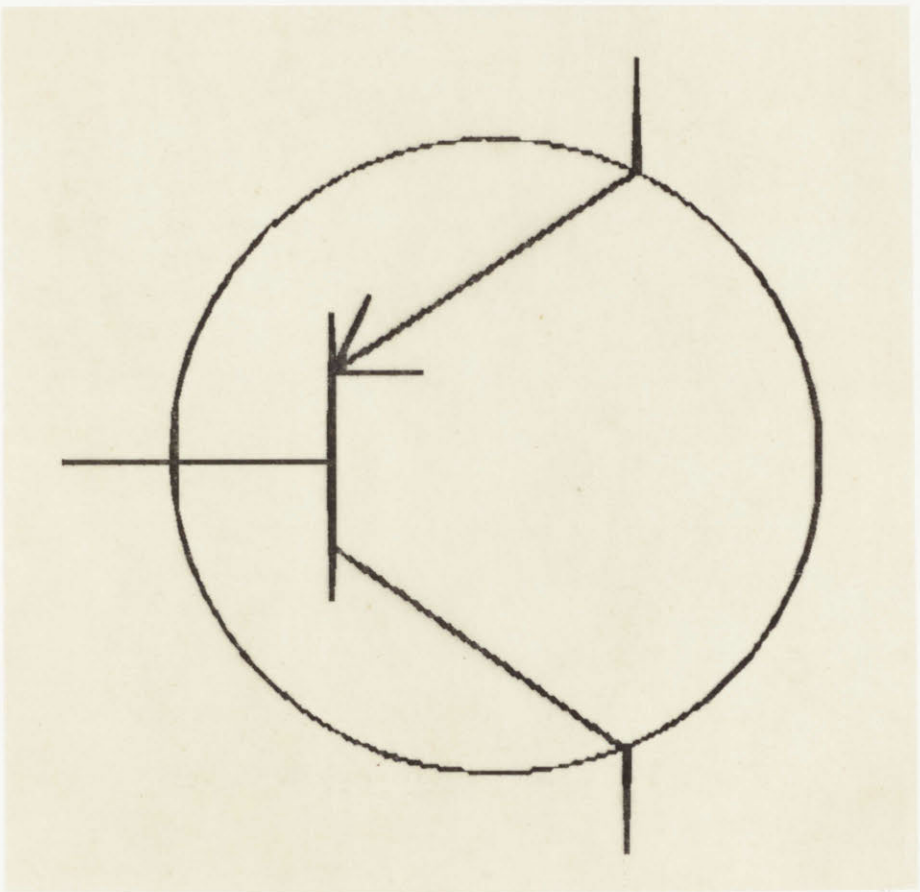
5-8(d). Coincident Resampling, example 4.
resolution = 200 lines/inch



Given Image



$s_x = s_y = .25$



$s_x = s_y = 3.5$

5-8(e). Coincident Resampling, example 5.
resolution = 200 lines/inch

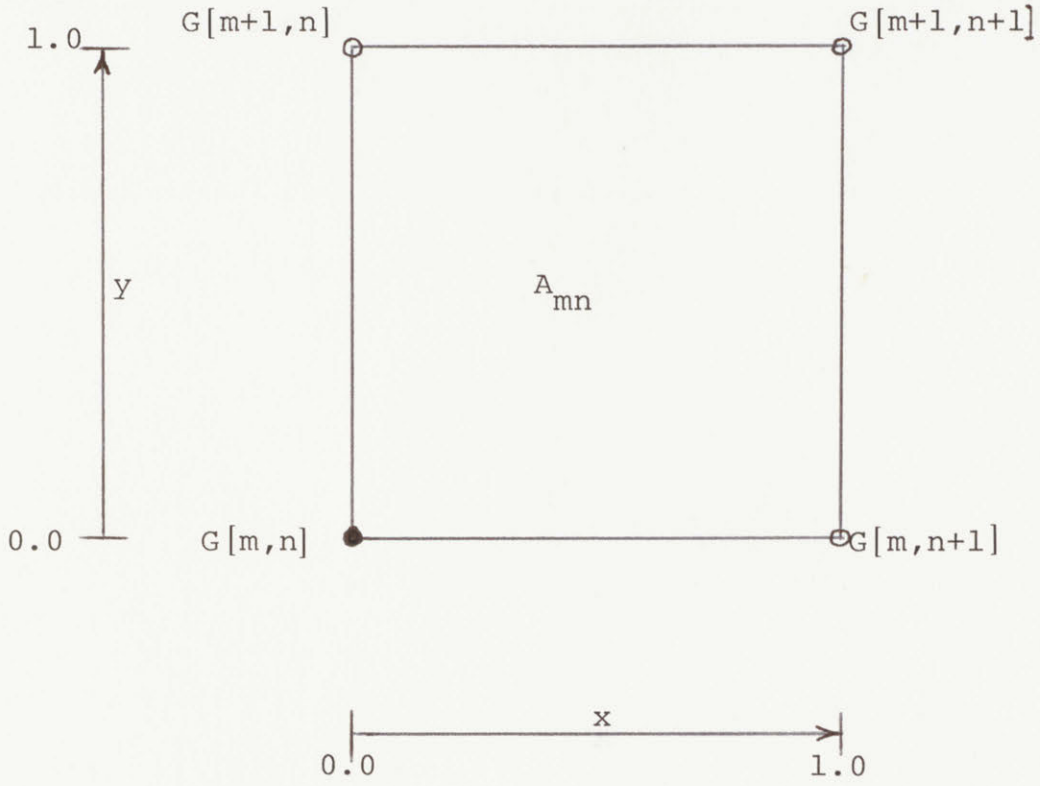


Figure 5-9. A_{mn}, the 1 by 1 continuous assignment area associated with a given sample at G[m,n].

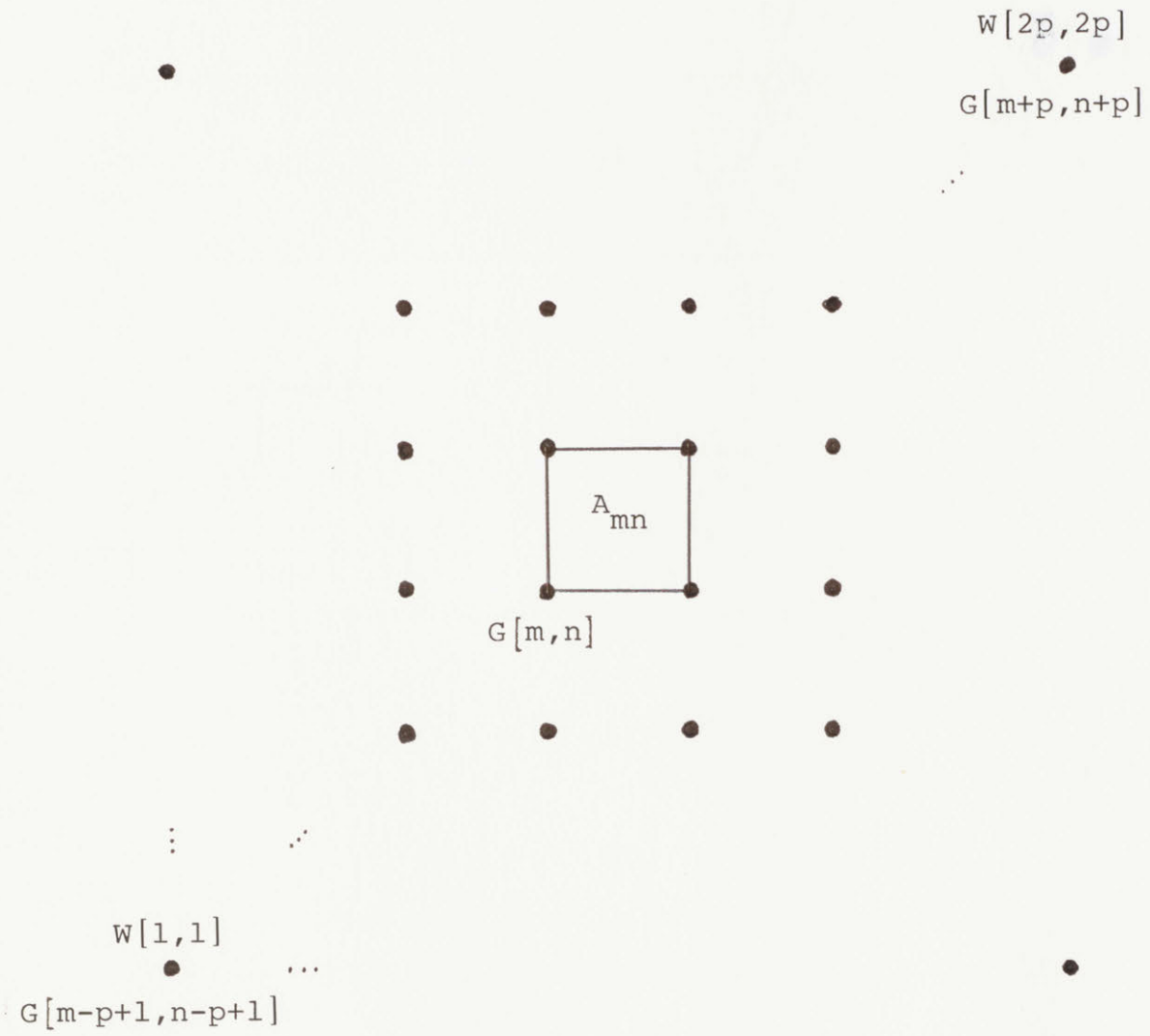


Figure 5-10. W_p, the pth order discrete space binary WINDOW about A_{mn}.

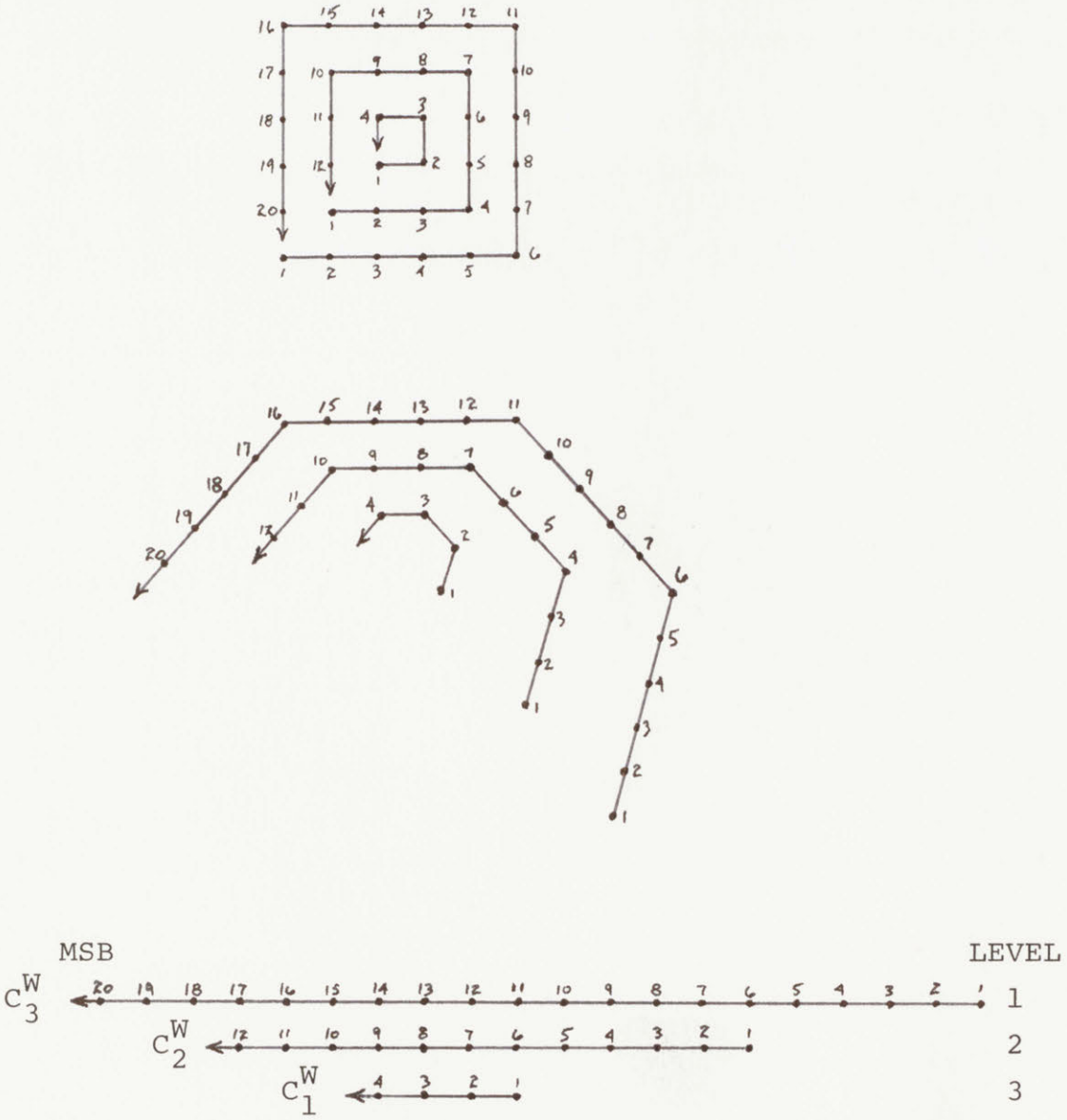


Figure 5-11. 3rd order example of the window "unwrapping" process.

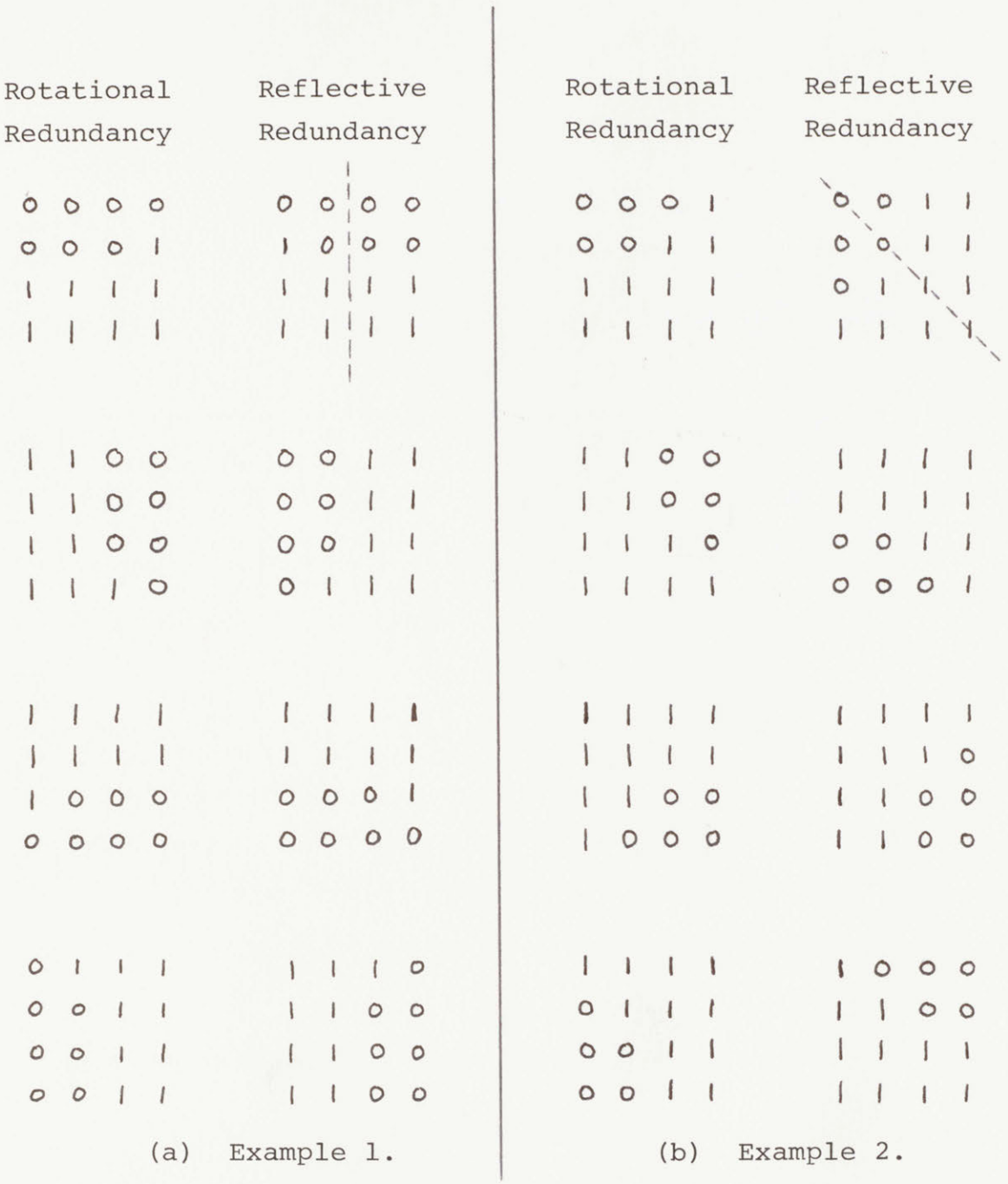


Figure 5-12. Examples of redundancy due to rotations and reflections of a 2nd order window.

(00) $\begin{matrix}0&0\\0&0\end{matrix}$ $\Longrightarrow$ all white

(05) $\begin{matrix}0&1\\1&0\end{matrix}$ (12) $\begin{matrix}1&0\\0&1\end{matrix}$ $\Longrightarrow$ ignore

(17) $\begin{matrix}1&1\\1&1\end{matrix}$ $\Longrightarrow$ all black

(a) Codes for which further decoding is not necessary.

(02) $\begin{matrix}0&0\\0&1\end{matrix}$	(03) $\begin{matrix}0&0\\1&1\end{matrix}$	(07) $\begin{matrix}0&1\\1&1\end{matrix}$	RL=0 (STANDARD FORM)
(01) $\begin{matrix}0&0\\1&0\end{matrix}$	(11) $\begin{matrix}1&0\\1&0\end{matrix}$	(13) $\begin{matrix}1&0\\1&1\end{matrix}$	RL=1
(10) $\begin{matrix}1&0\\0&0\end{matrix}$	(14) $\begin{matrix}1&1\\0&0\end{matrix}$	(15) $\begin{matrix}1&1\\1&0\end{matrix}$	RL=2
(04) $\begin{matrix}0&1\\0&0\end{matrix}$	(06) $\begin{matrix}0&1\\0&1\end{matrix}$	(16) $\begin{matrix}1&1\\0&1\end{matrix}$	RL=3

(b) Rotational permutations of "STANDARD FORM".

Figure 5-13. The 16 C_1^W codes.

(Octal values in parentheses)

W_2, Given Window

$C_1^W = 13 \Longrightarrow RL = 1$

$C_2^W = 7417$

D_2, Derotated Window

$C_1^D = 07$

$C_2^D = 4177$

Figure 5-14. 2nd order example of Derotation (Concentric Codes in octal).

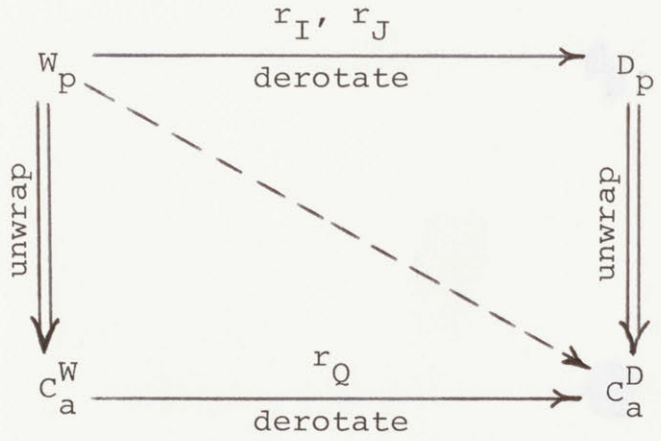
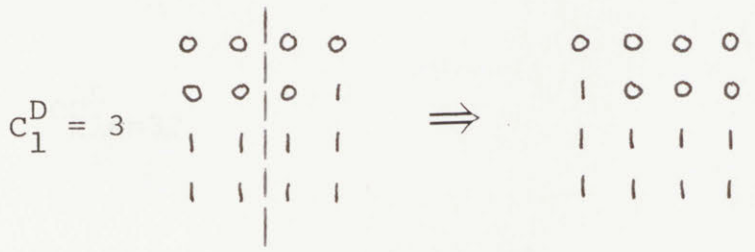
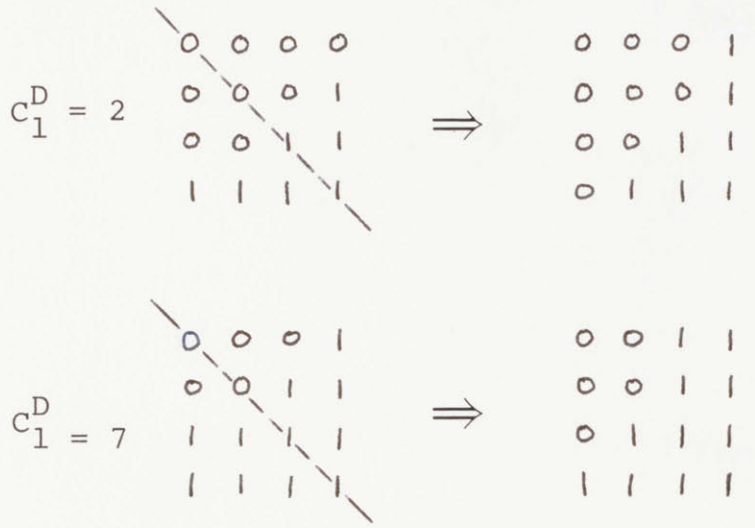


Figure 5-15. Mapping of a given window to derotated concentric codes.



(a) Horizontal Reflective Redundancy (RFL=1)



(b) Transposal Reflective Redundancy (RFL=2)

(c) Examples of Reflectively Symmetric windows

Figure 5-16. Distinguishing reflection types.

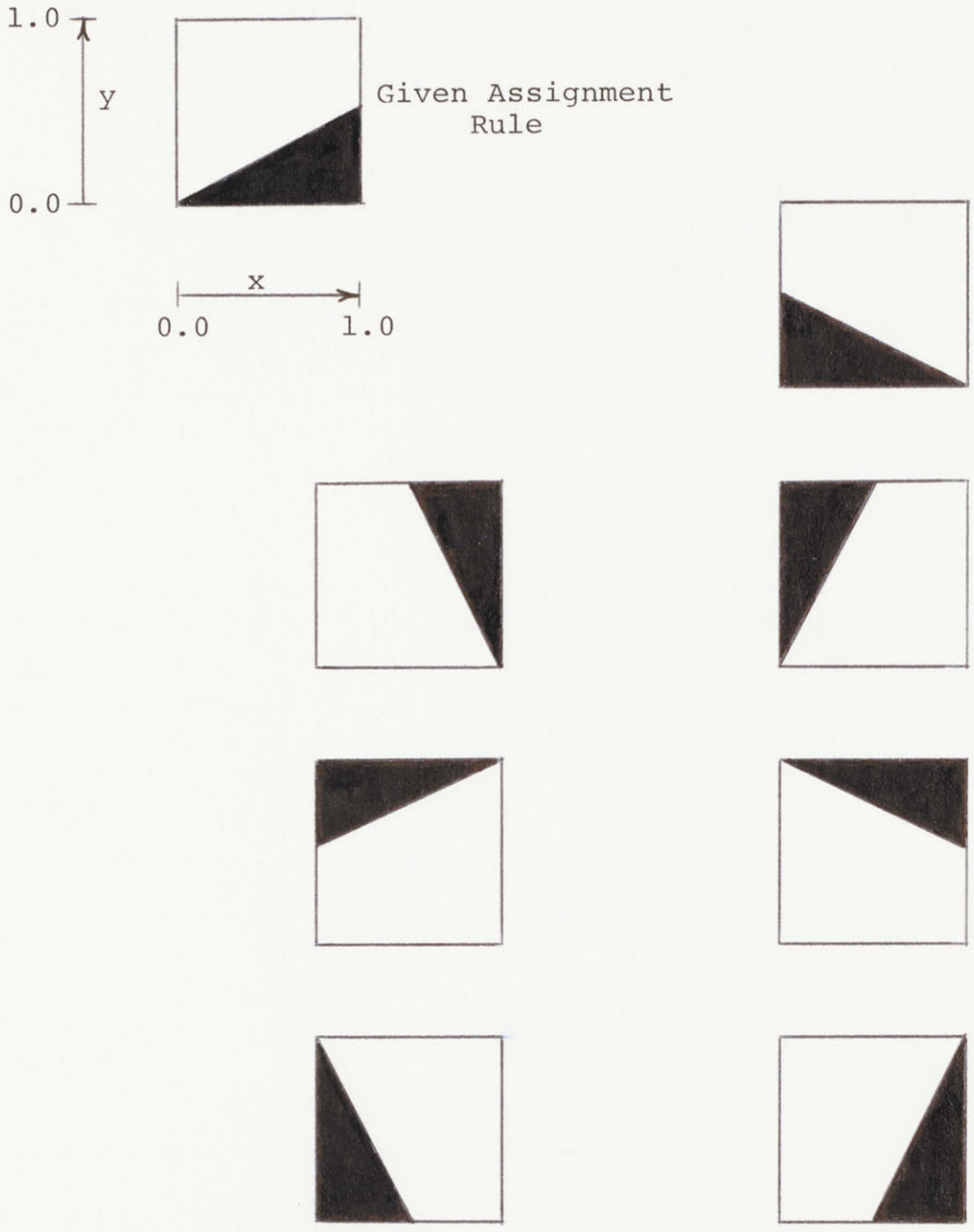
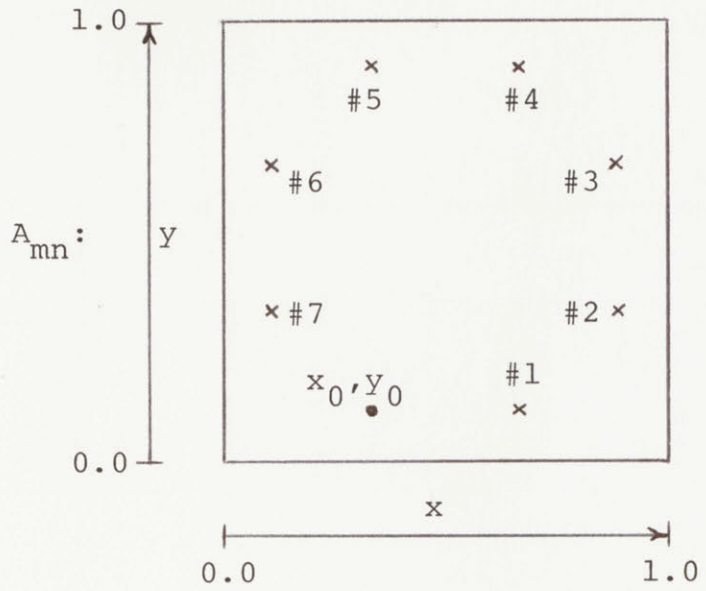


Figure 5-17. Redundant Permutations of a given Assignment Rule



(a)

		Effective x value	Effective y value	Location in (a)
Unreflected Template (RFL=0)	RL=0	x_0	y_0	
	RL=1	$1-y_0$	x_0	#2
	RL=2	$1-x_0$	$1-y_0$	#4
	RL=3	y_0	$1-x_0$	#6
Horizontal Reflected Template (RFL=1)	RL=0	$1-x_0$	y_0	#1
	RL=1	y_0	x_0	#7
	RL-2	x_0	$1-y_0$	#5
	RL=3	$1-y_0$	$1-x_0$	#3
Transposal Reflected Template (RFL=2)	RL=0	$1-y_0$	$1-x_0$	#3
	RL=1	$1-x_0$	y_0	#1
	RL=2	y_0	x_0	#7
	RL=3	x_0	$1-y_0$	#5

(b)

Figure 5-18. Rearrangement of new sample coordinates due to rotational and reflective permutations.

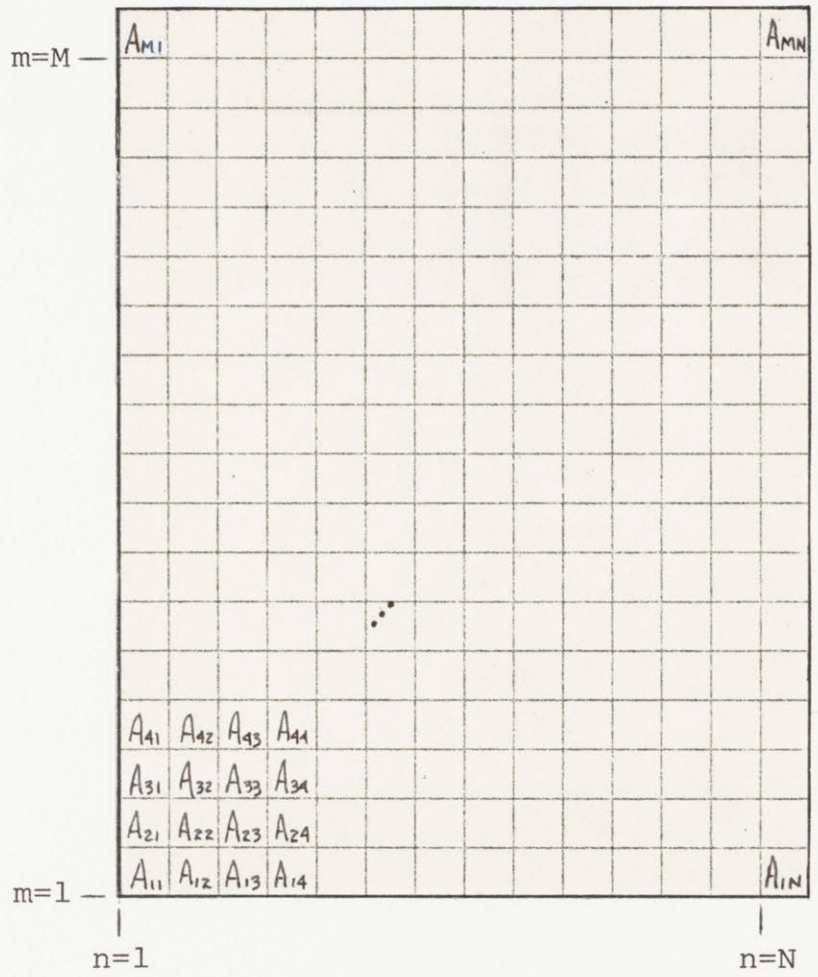
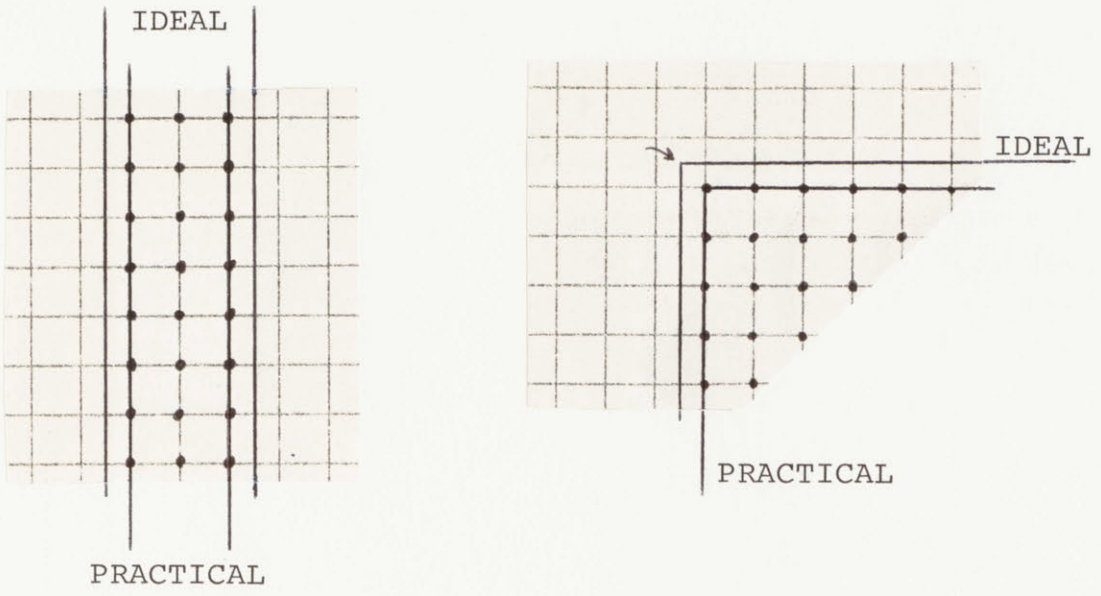
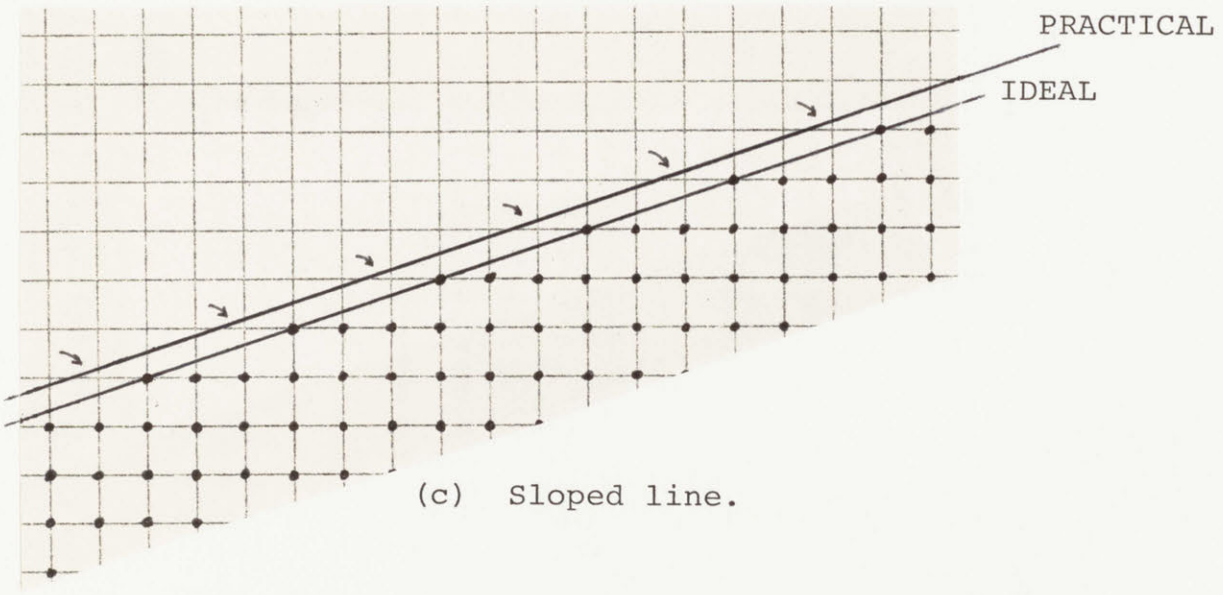


Figure 5-19. Subdividing the continuous reconstruction into Assignment Areas.



(a) Vertical bar

(b) Corner



(c) Sloped line.

Figure 5-20. Selection of Image Boundary.

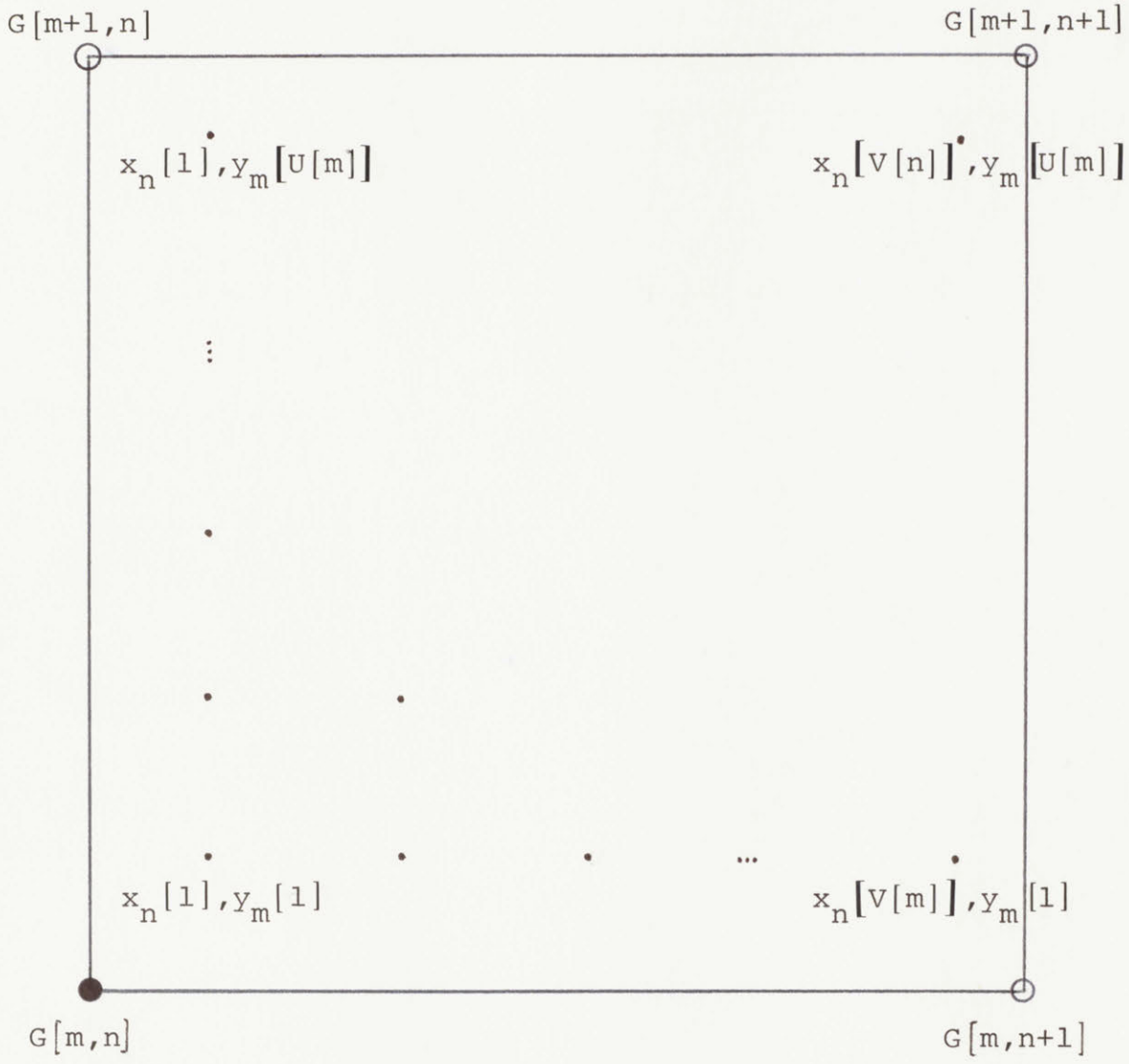
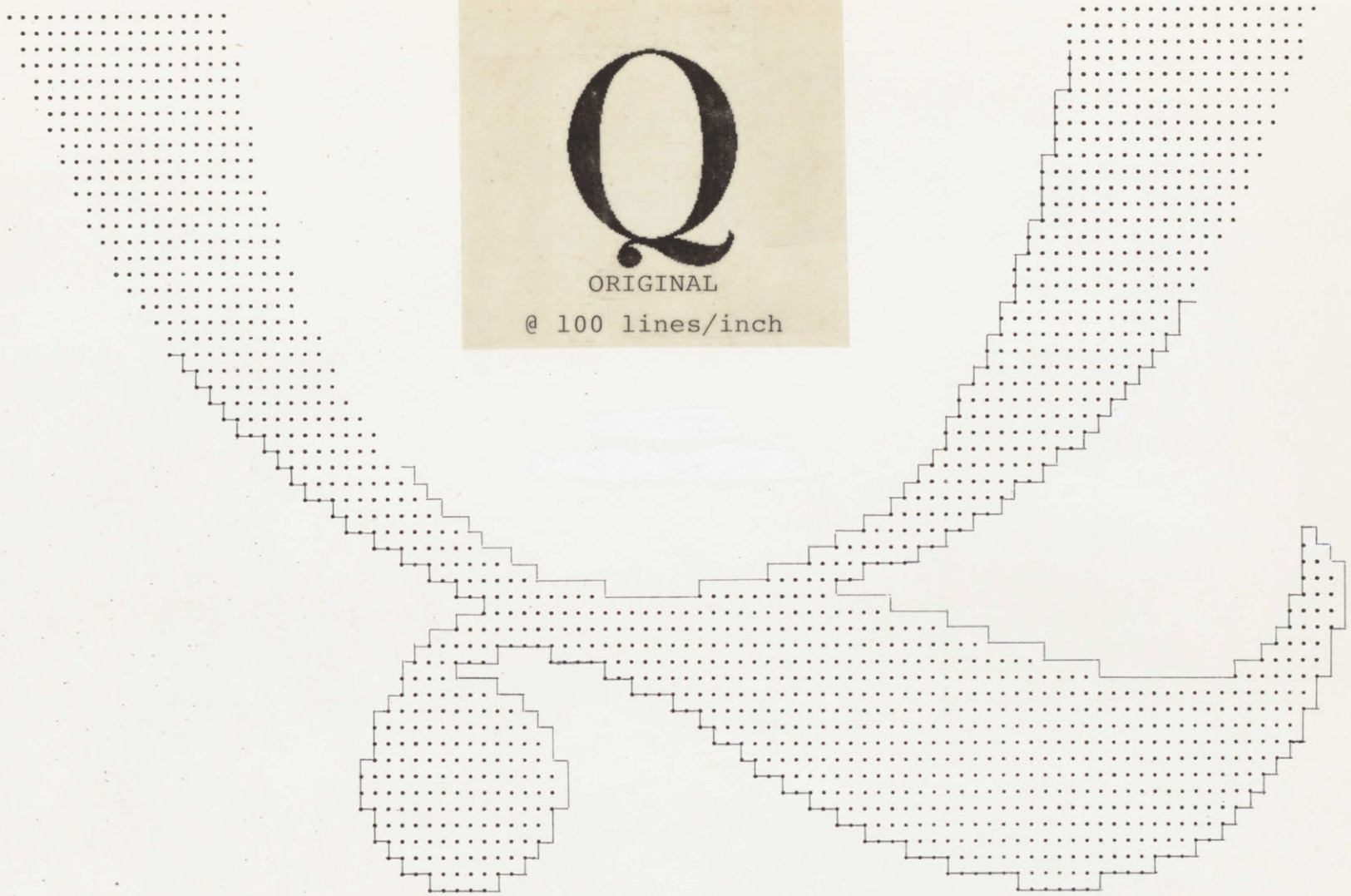


Figure 5-21. The matrix of new samples Z_{mn} within A_{mn}.

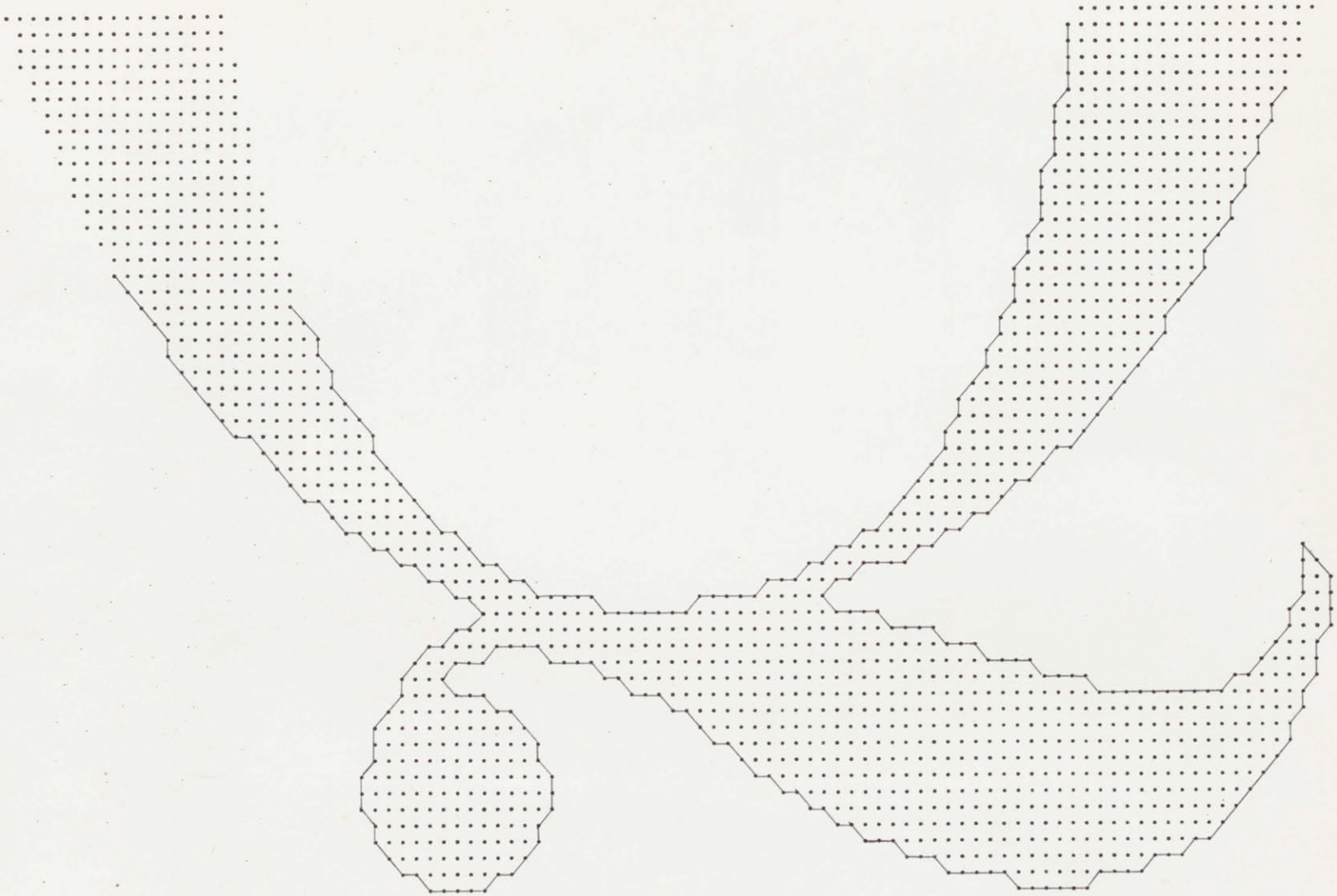


(a) Reconstruction Process

Figure 5-22. 0th Order Scaling.

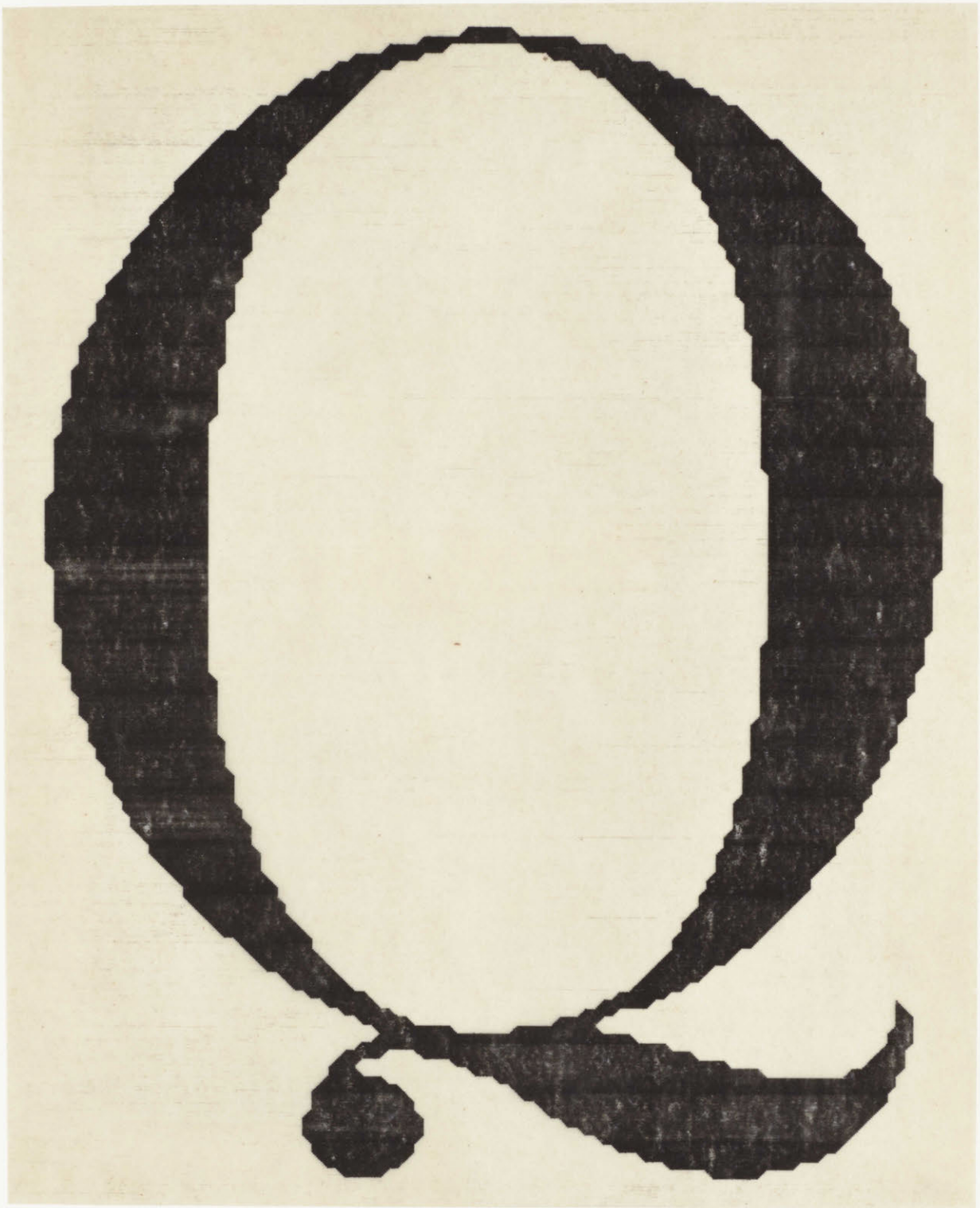
5-22(b). Result of 0th Order Scaling.

$s_x = s_y = 5.5$; resolution = 100 lines/inch

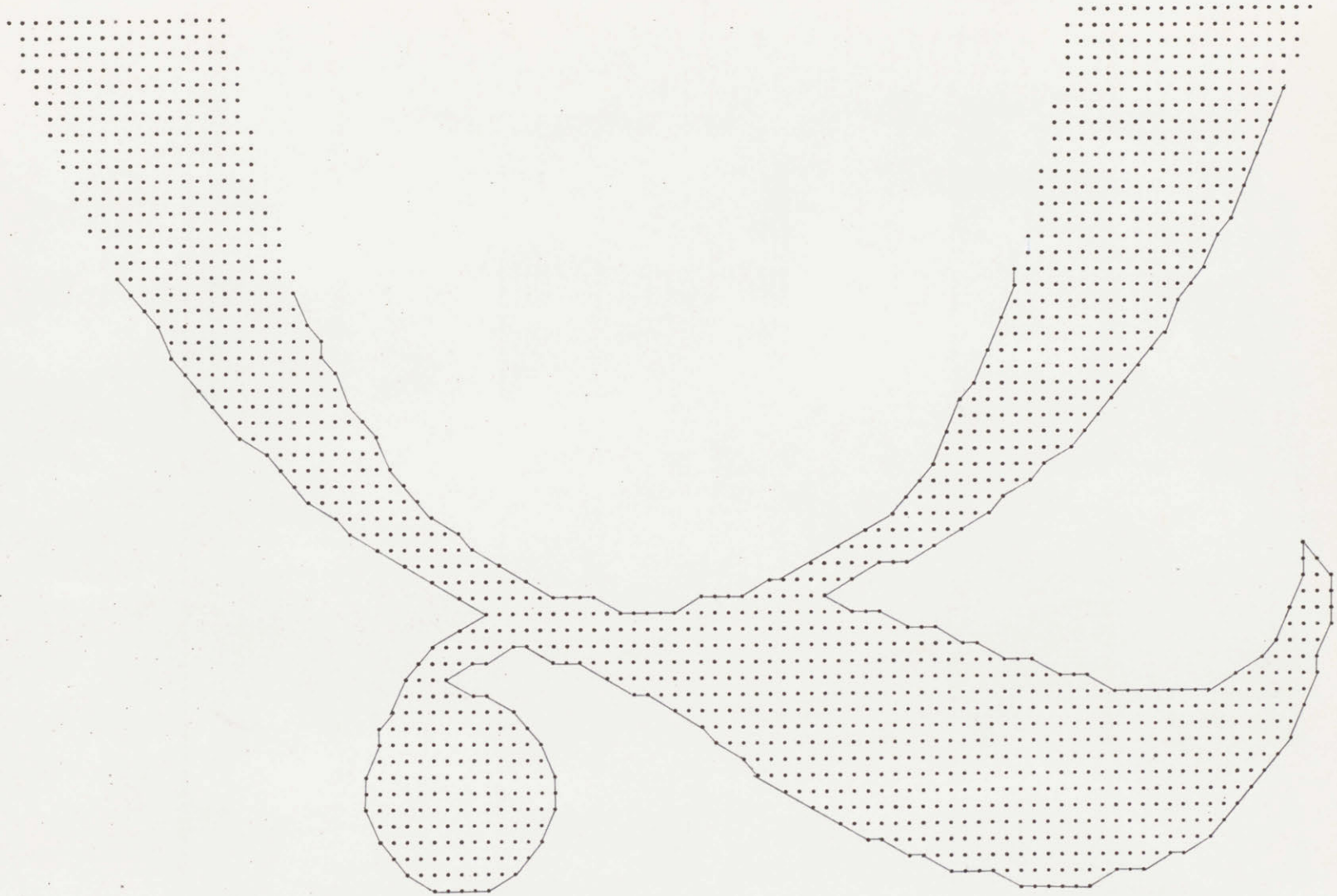


(a) Reconstruction Process

Figure 5-23. 1st Order Scaling.

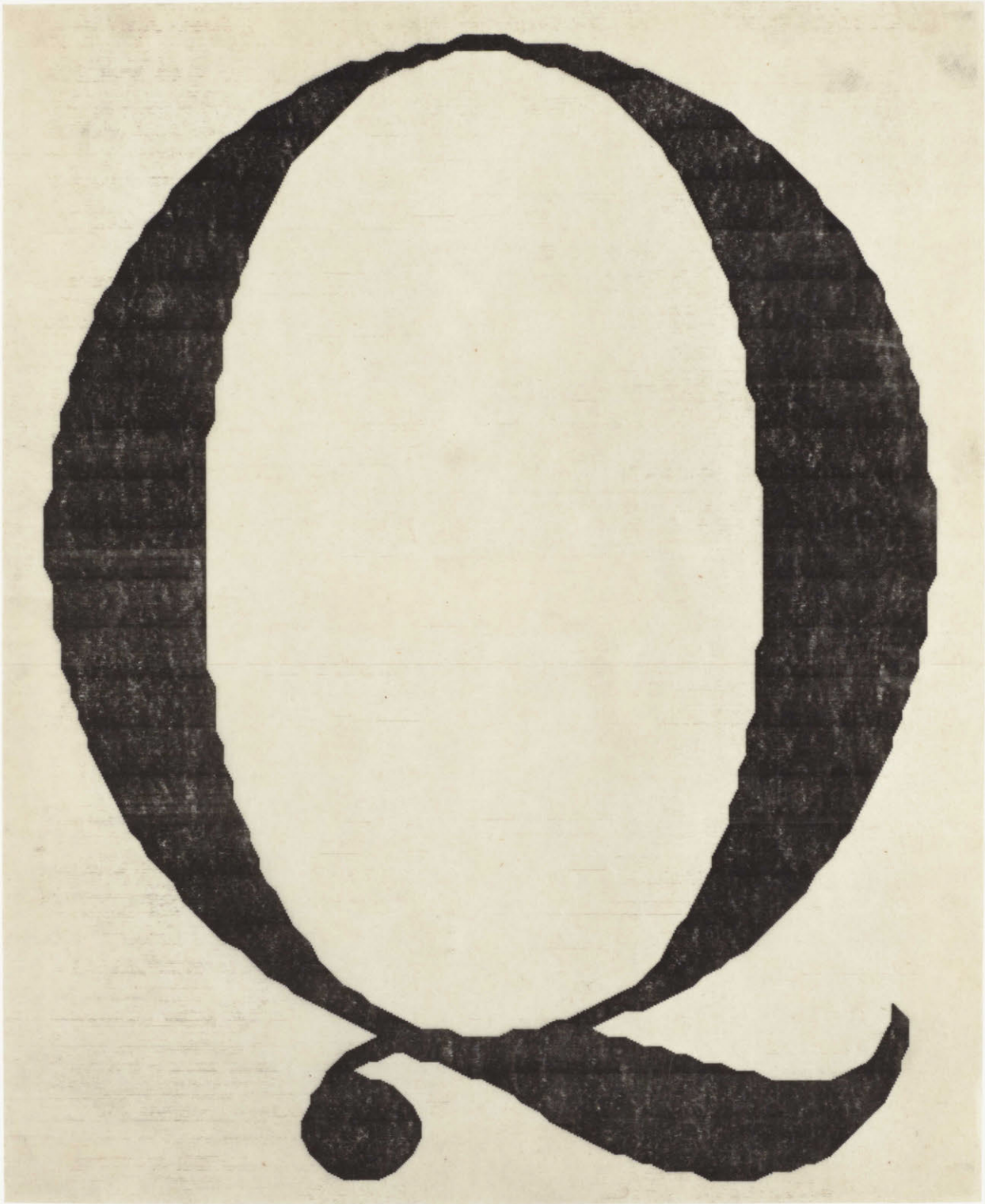


5-23(b). Result of 1st Order Scaling.
$s_x = s_y = 5.5$; resolution = 100 lines/inch

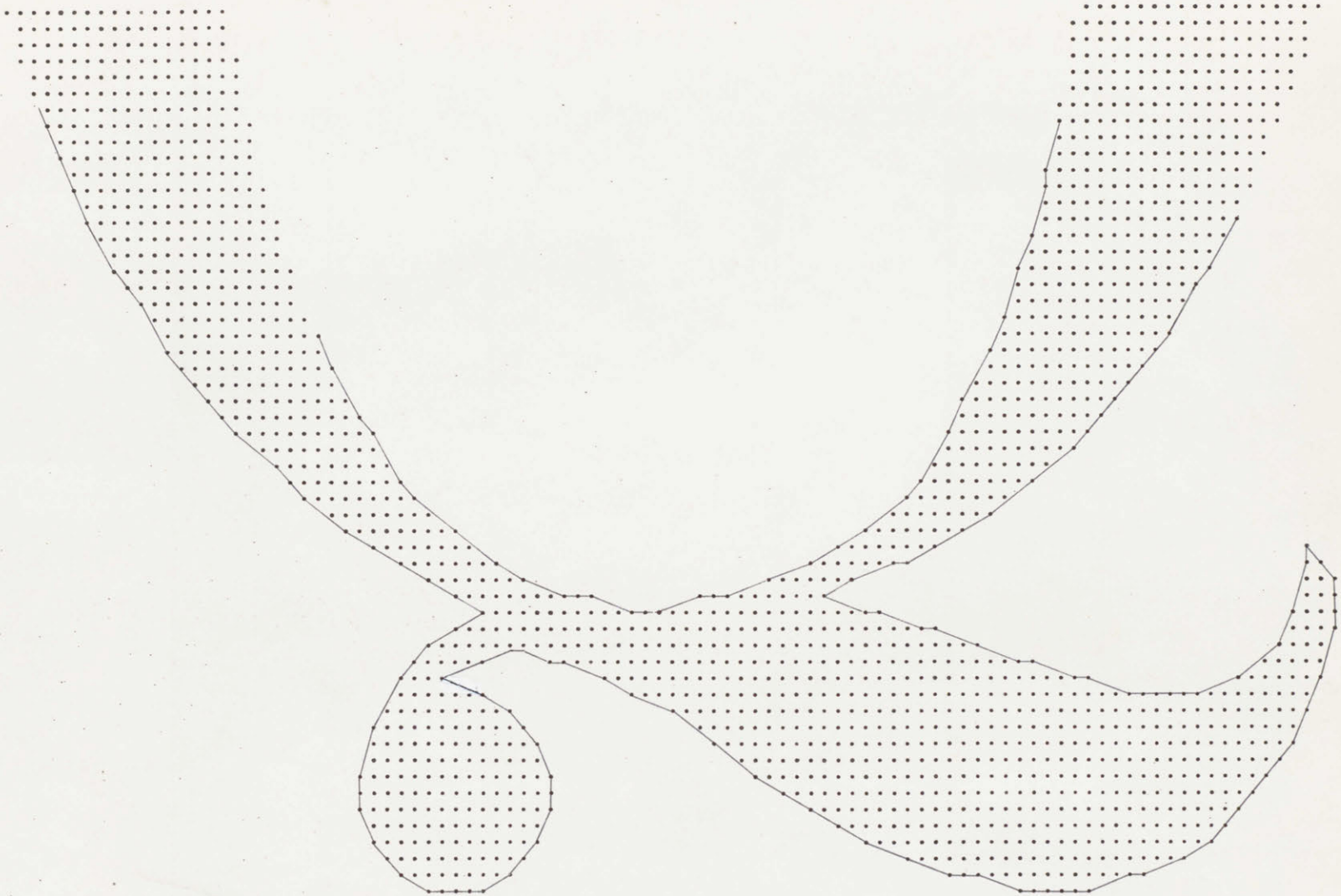


(a) Reconstruction Process

Figure 5-24. 2nd Order Scaling.

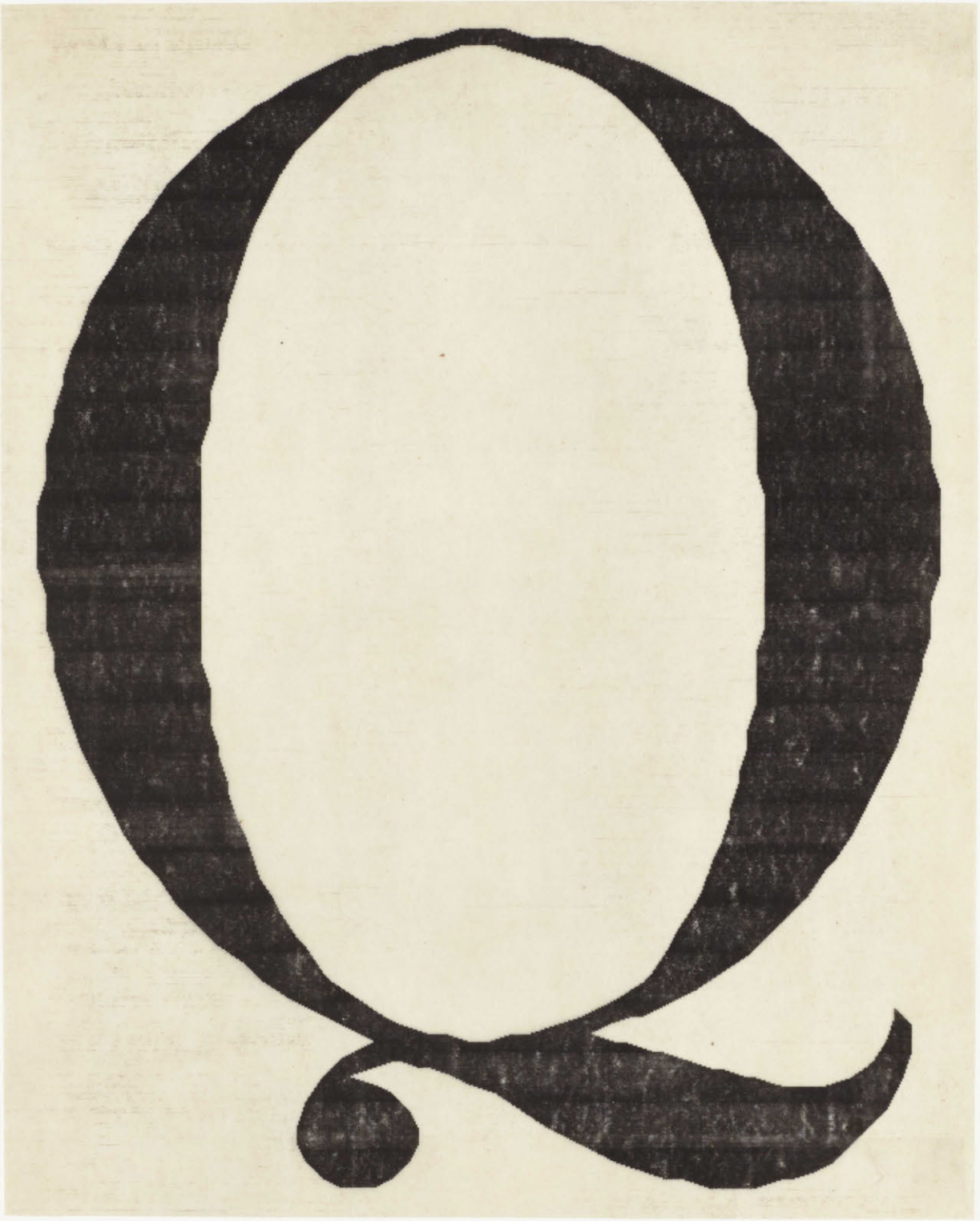


5-24(b). Result of 2nd Order Scaling.
$s_x = s_y = 5.5$; resolution = 100 lines/inch



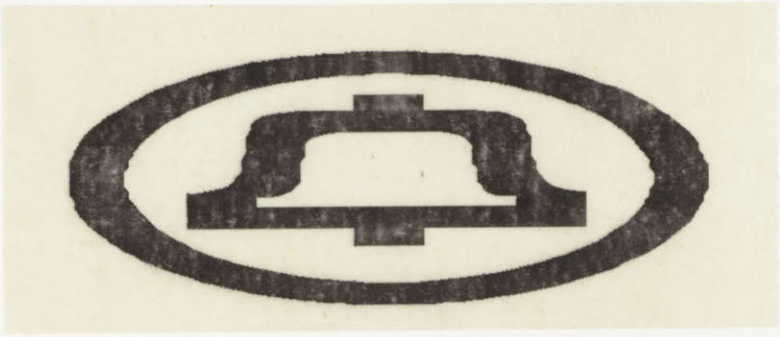
(a) Reconstruction Process

Figure 5-25. 3rd Order Scaling.



5-25(b). Result of 3rd Order Scaling.
$s_x = s_y = 5.5$; resolution = 100 lines/inch

Given Image



$s_x = 5.0,\ s_y = 2.0$

$s_x = s_y = 5.1$

(a) Example 1. p=3, resolution = 200 lines/inch

Figure 5-26. Telescoping Template enlargement examples.

Given Image

p=0

p=1

5-26(b). Telescoping Template example 2.
$s_x = s_y = 5.5$, resolution = 200 lines/inch

p=2

p=3

(5-26(b) continued)

Given
Image

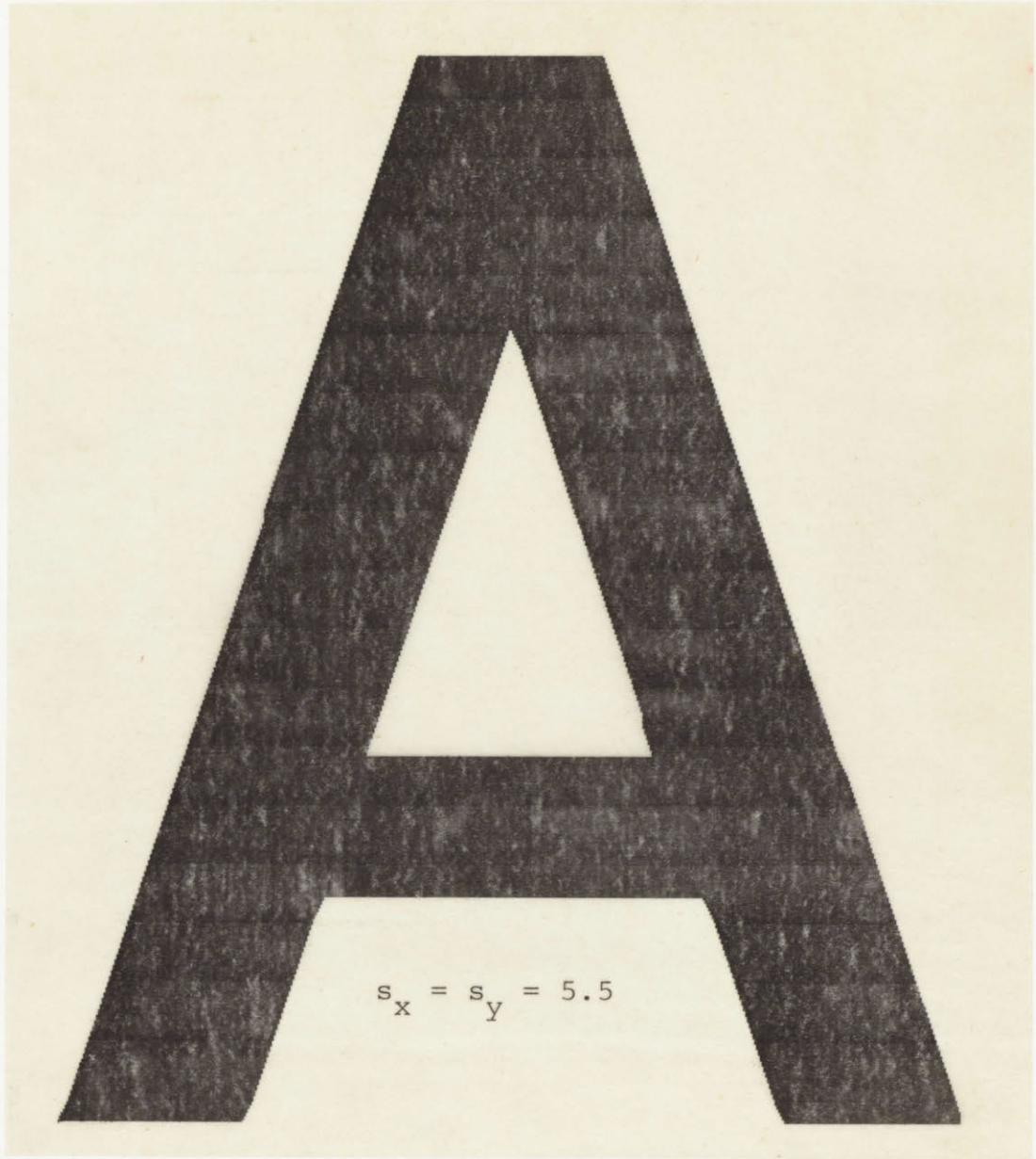
$s_x = s_y = 5.0$

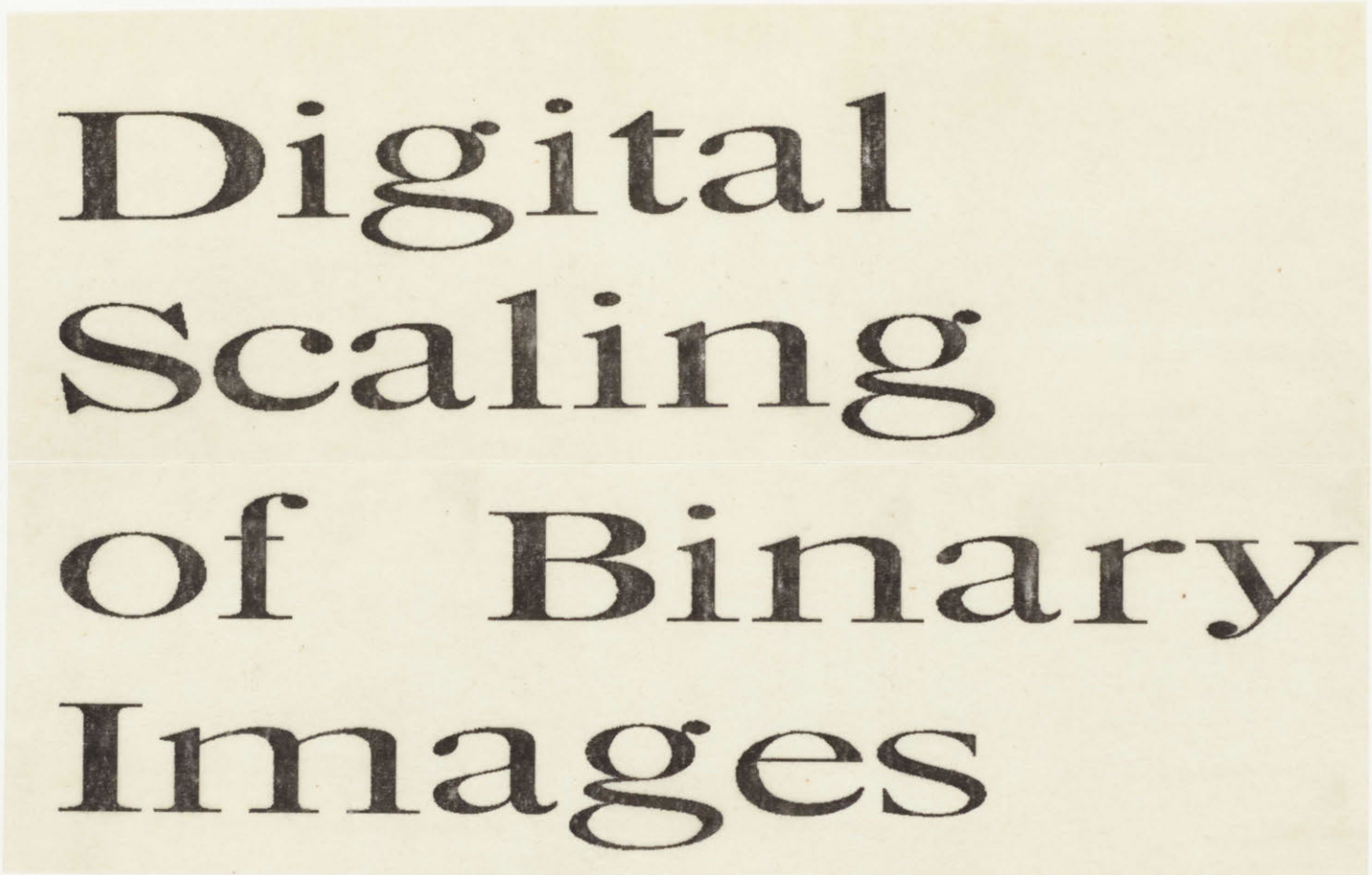
$s_x = 7.0,\ s_y = 5.0$

5-26(c). Telescoping Template example 3.
p=3, resolution = 200 lines/inch.

Given Image

5-26(d). Telescoping Template example 4. p = 3, resolution = 100 lines/inch





$s_x = 2.0$, $s_y = 1.0$

5-26(e). Telescoping Template example 5.
p = 3, resolution = 200 lines/inch
(Given Image shown in Figure 5-8(e))

$s_x = 1.0$, $s_y = 2.0$

(5-26(e) continued)

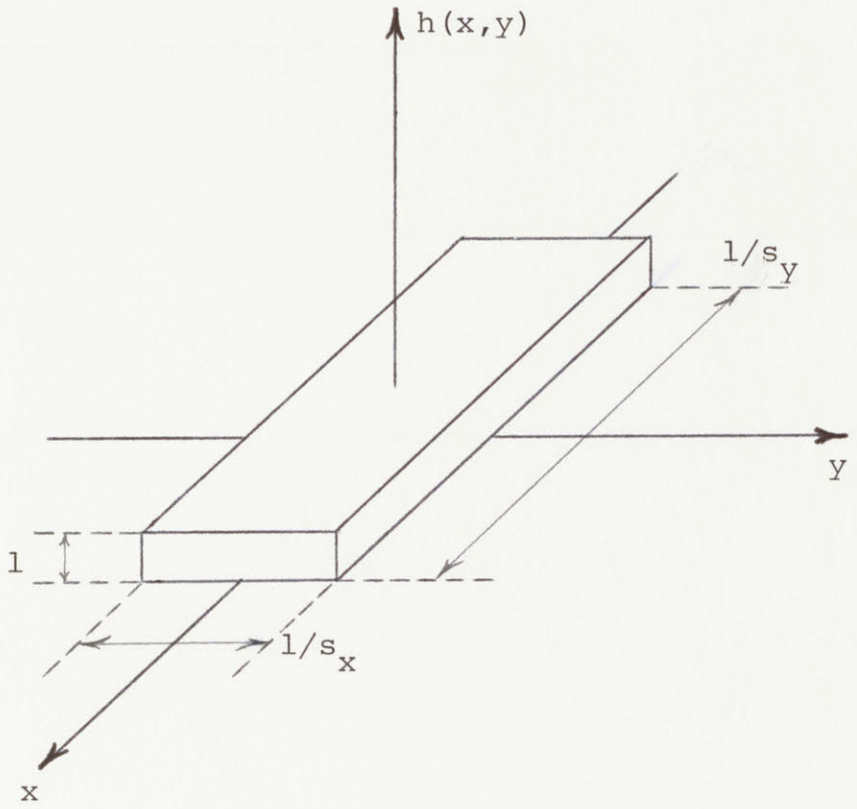
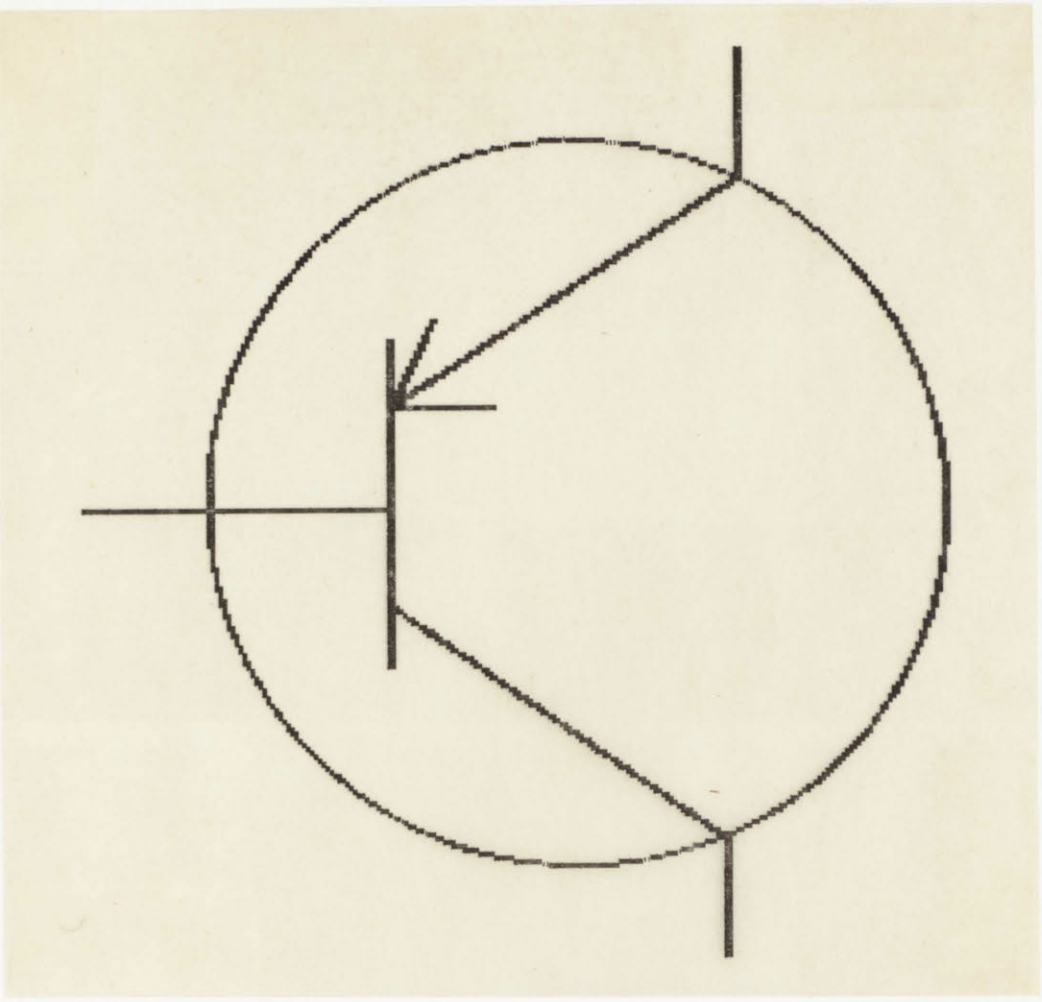
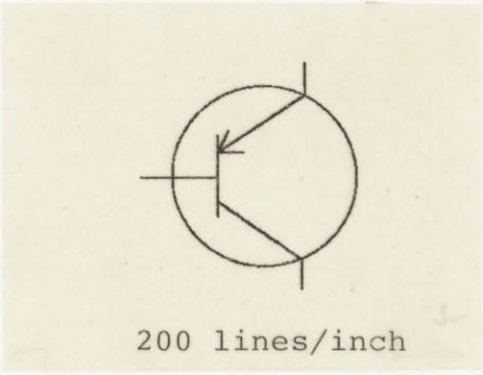


Figure 5-27. Reduction Convolution Kernel.

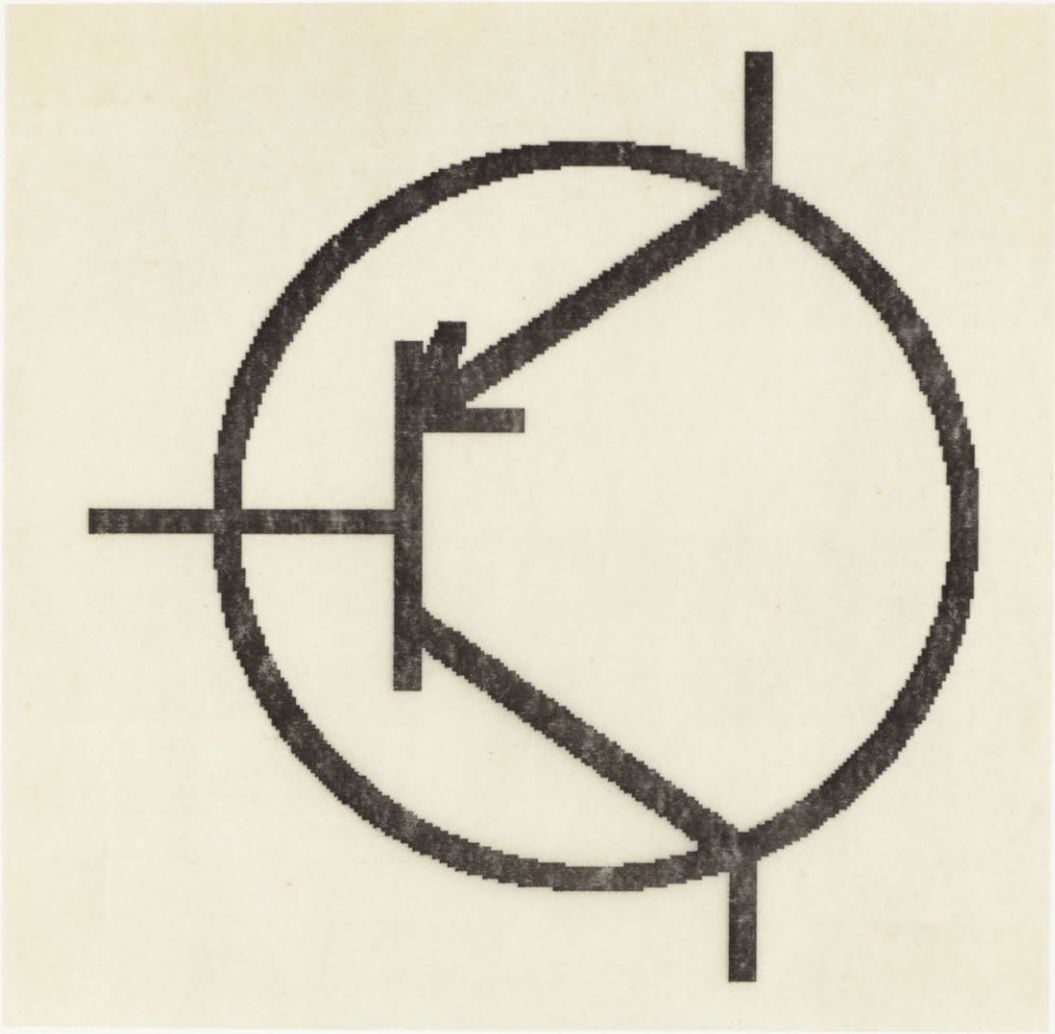


50 lines/inch

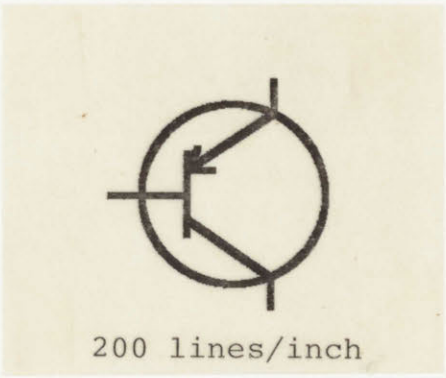


200 lines/inch

Figure 5-28. Original thin line image.

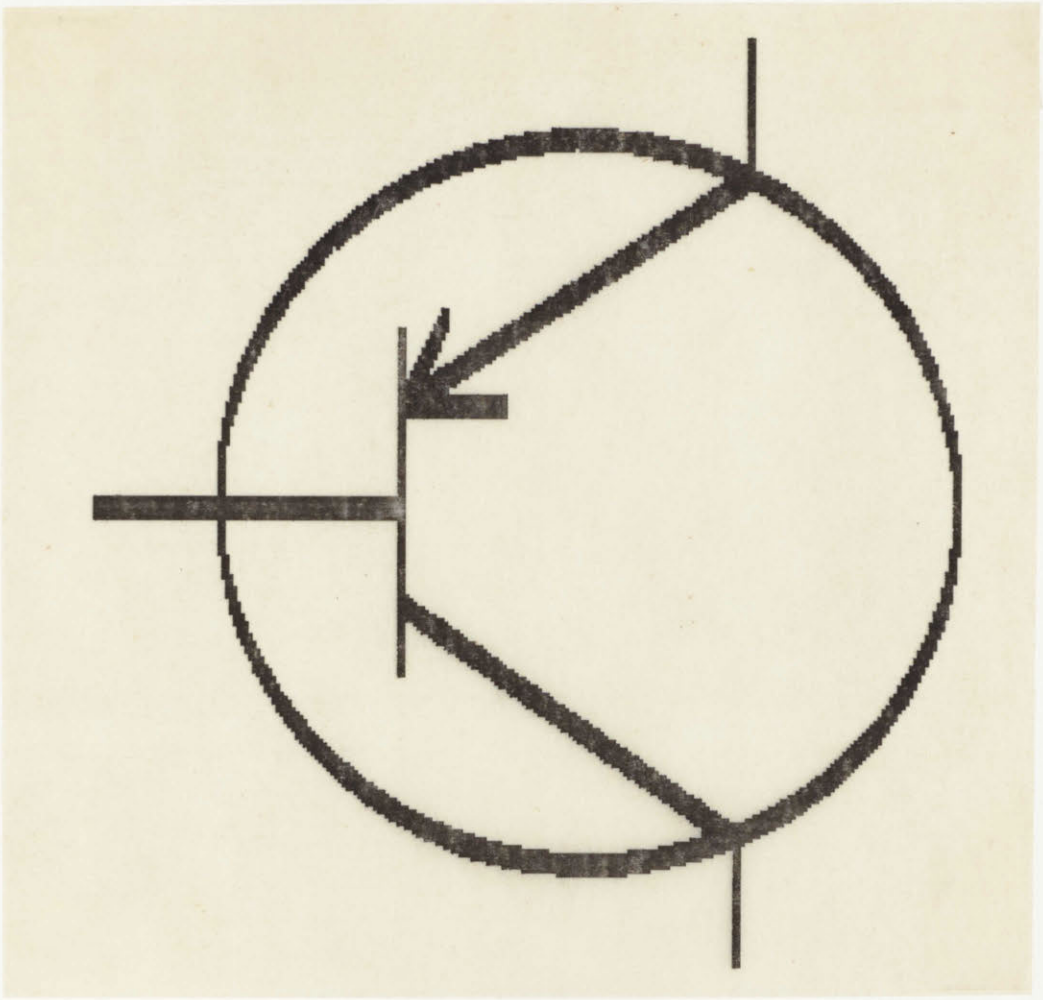


50 lines/inch

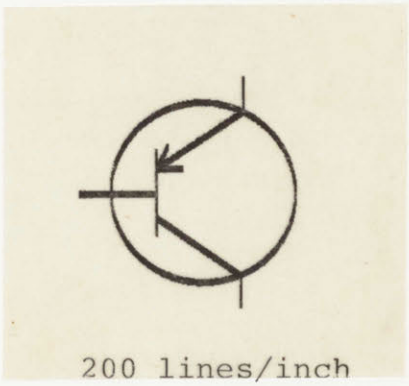


200 lines/inch

Figure 5-29. "Binary Convolution" with $s_x = s_y = 1.0$, $d_H = d_V = 6$

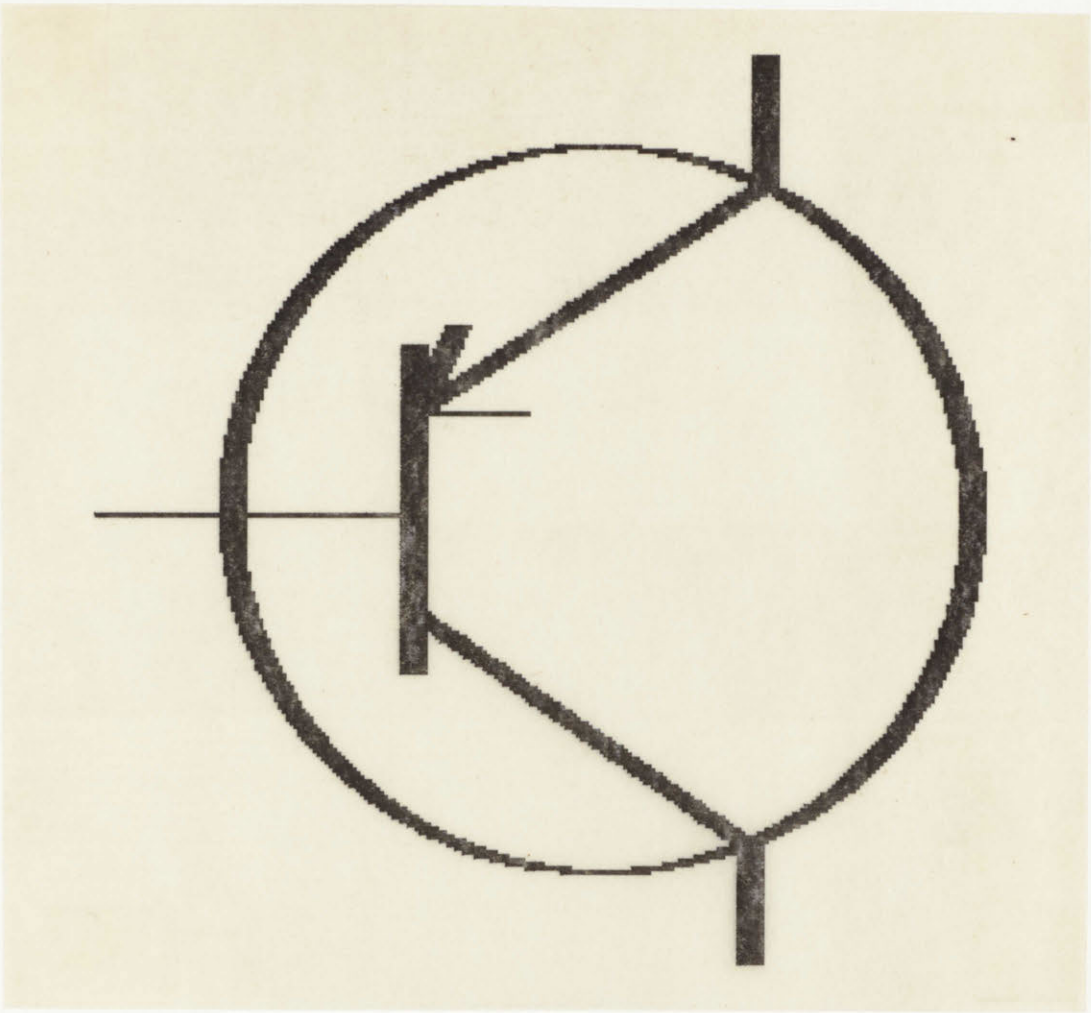


50 lines/inch

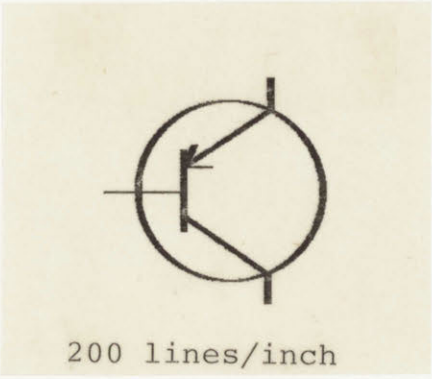


200 lines/inch

Figure 5-30. "Binary Convolution" with $s_x = s_y = 1.0$, $d_H = 1$, $d_V = 6$

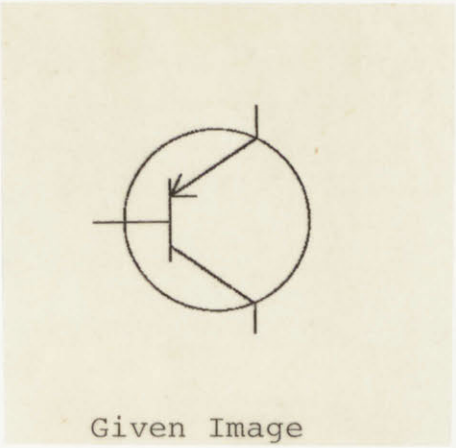


50 lines/inch

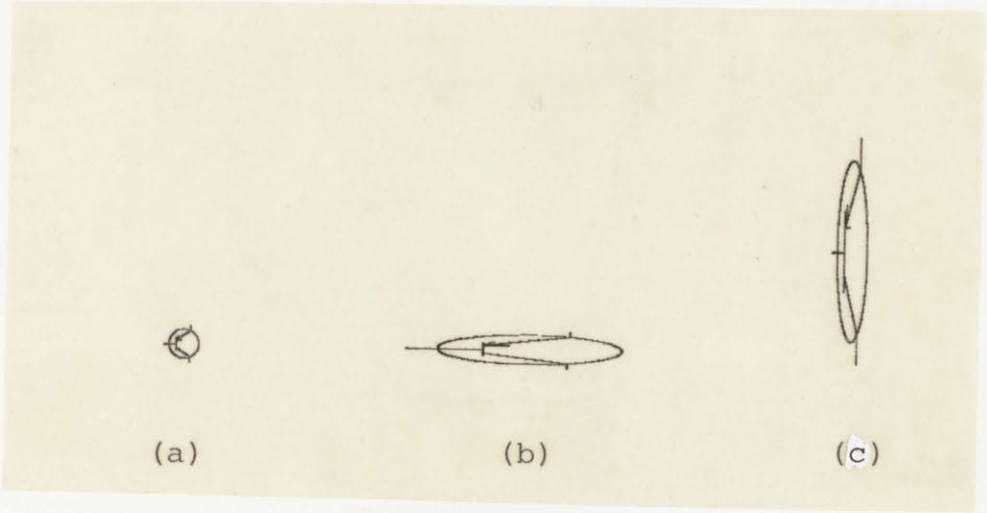


200 lines/inch

Figure 5-31. "Binary Convolution" with $s_x = s_y = 1.0$, $d_H = 6$, $d_V = 1$



Given Image



(a) (b) (c)

s_x	.1667	.1667	1.0
s_y	.1667	1.0	.1667
d_H	6	6	1
d_V	6	1	6

Figure 5-32. Reductions enhanced by Binary Convolution.

resolution = 200 lines/inch

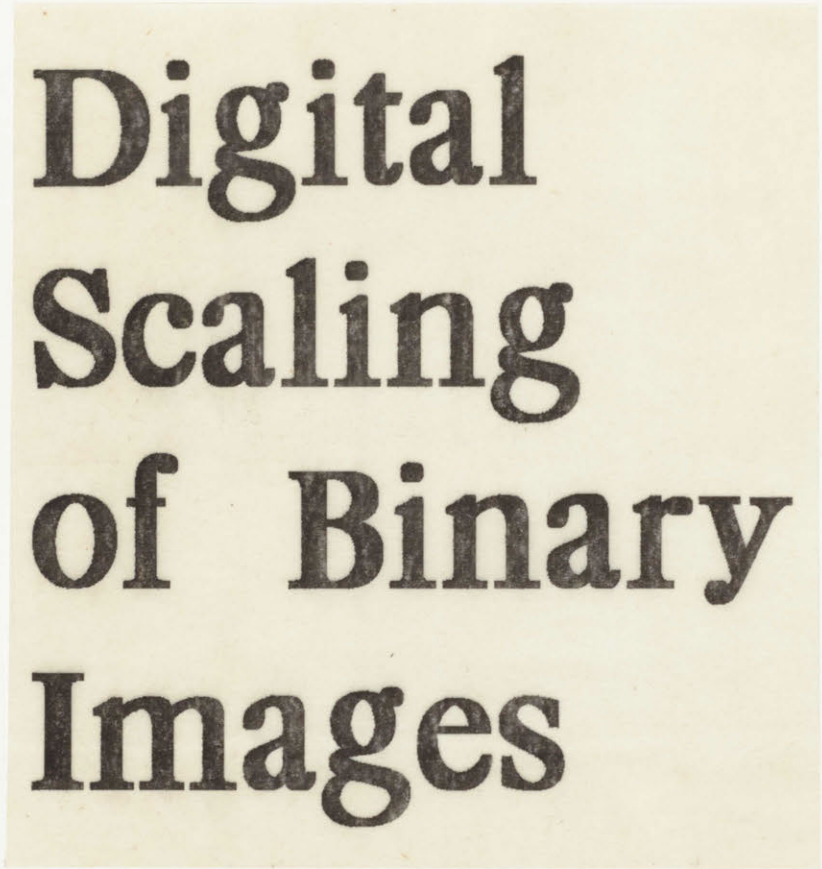
Given Image

$s_x = s_y = 1.0$

$d_H = 6, d_V = 1$

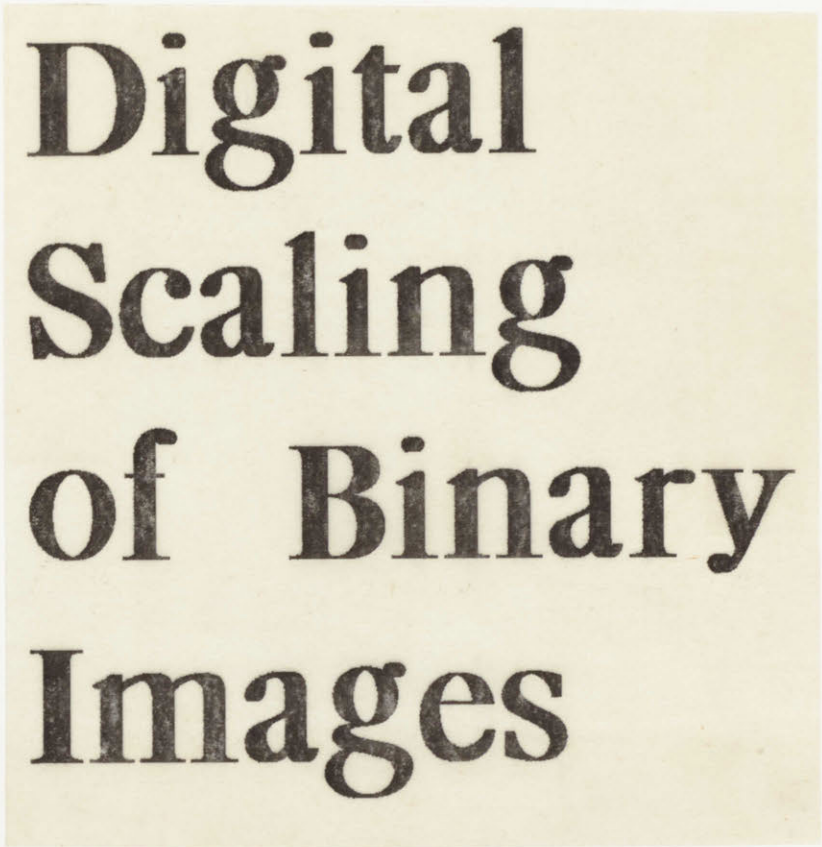
(a) Example 1. resolution = 200 lines/inch

Figure 5-33. Other examples of Binary Convolution



$s_x = s_y = 1.0$

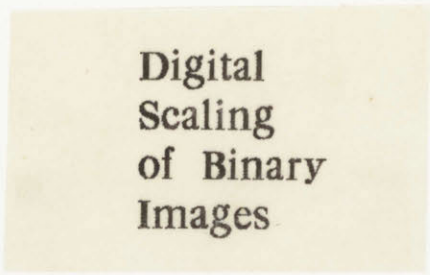
$d_H = 10, d_V = 6$



$s_x = s_y = 1.0$

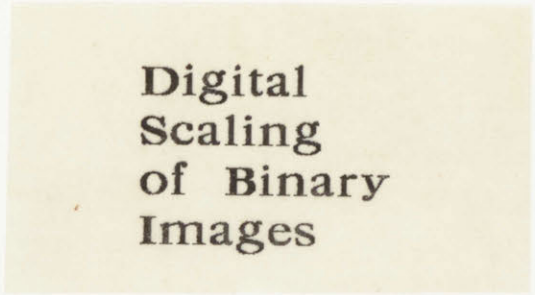
$d_H = 10, d_V = 1$

(5-33(a) continued)



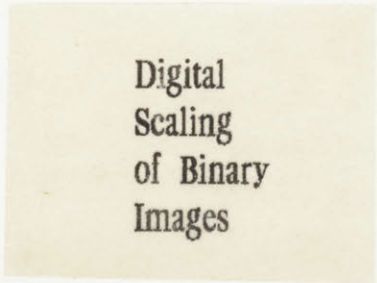
$s_x = s_y = .25$

$d_H = d_V = 4$



$s_x = .33,\ s_y = .25$

$d_H = 3,\ d_V = 4$



$s_x = .18,\ s_y = .24$

$d_H = 6,\ d_V = 4$

$s_x = .24,\ s_y = .18$

$d_H = 4,\ d_V = 6$

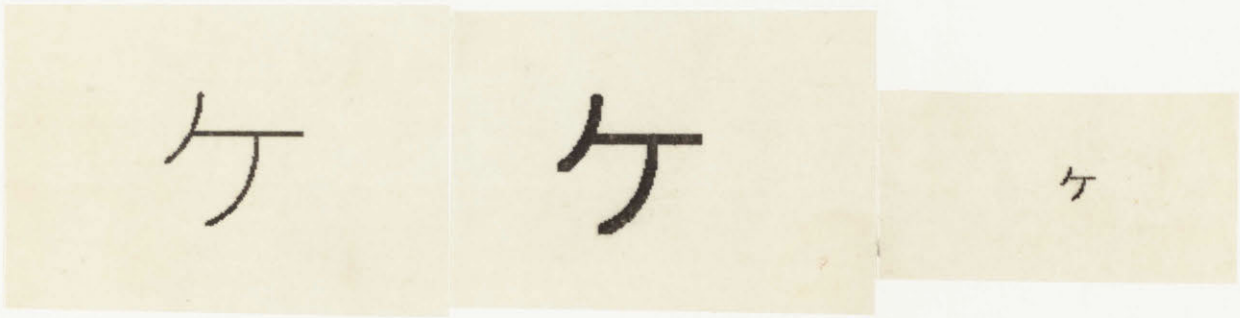
(5-33(a) continued)

Given Image (@ 20 lines/inch)

$s_x = s_y = .15$

$d_H = d_V = 6$

5-33(b). Example 2. resolution = 200 lines/inch



Given Image

$s_x = s_y = 1.0$

$d_H = d_V = 4$

$s_x = s_y = .25$

$d_H = d_V = 4$

5-33(c). Example 3. resolution = 100 lines/inch

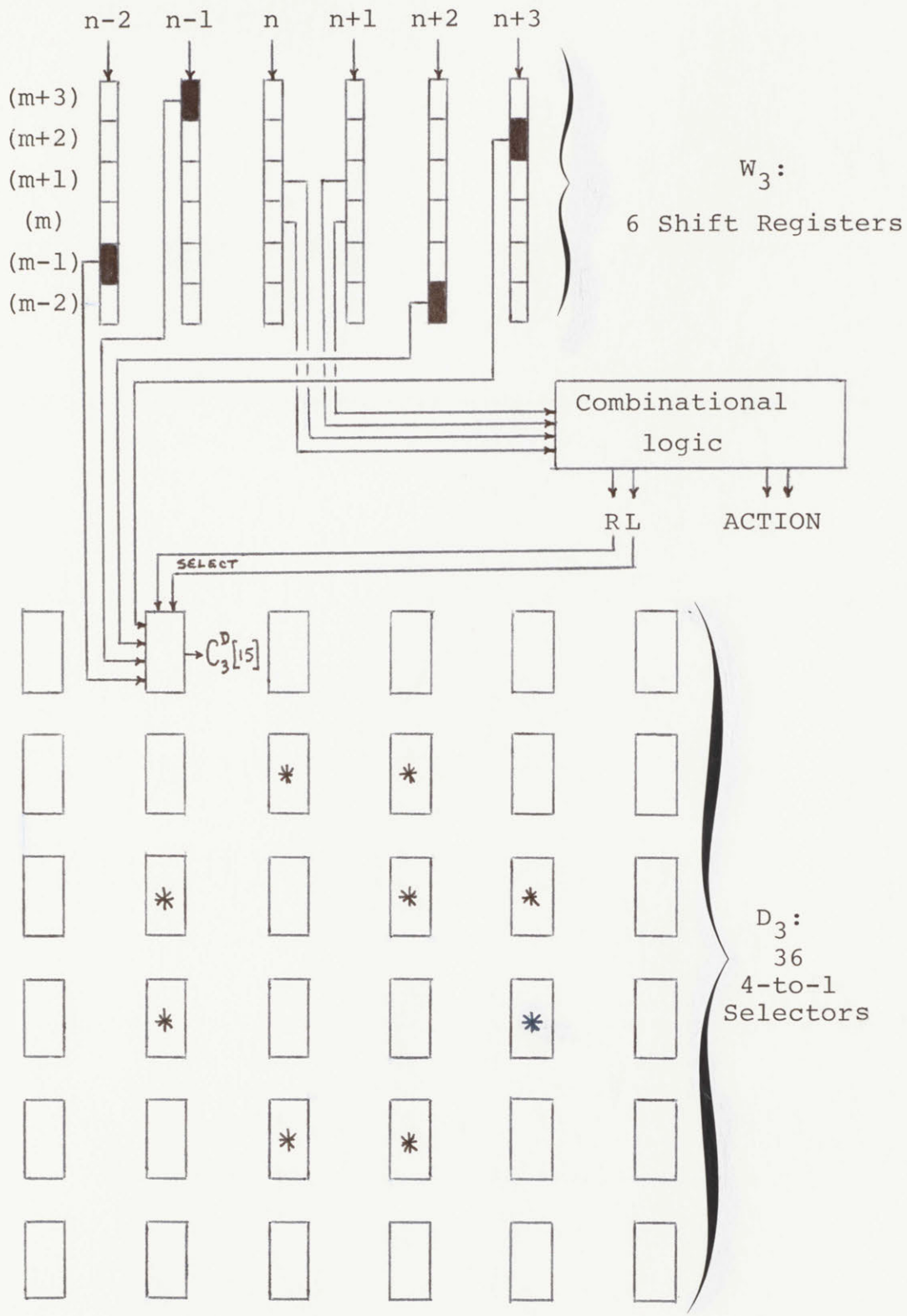


Figure 6-1: Parallel hardware for decoding a third order system.

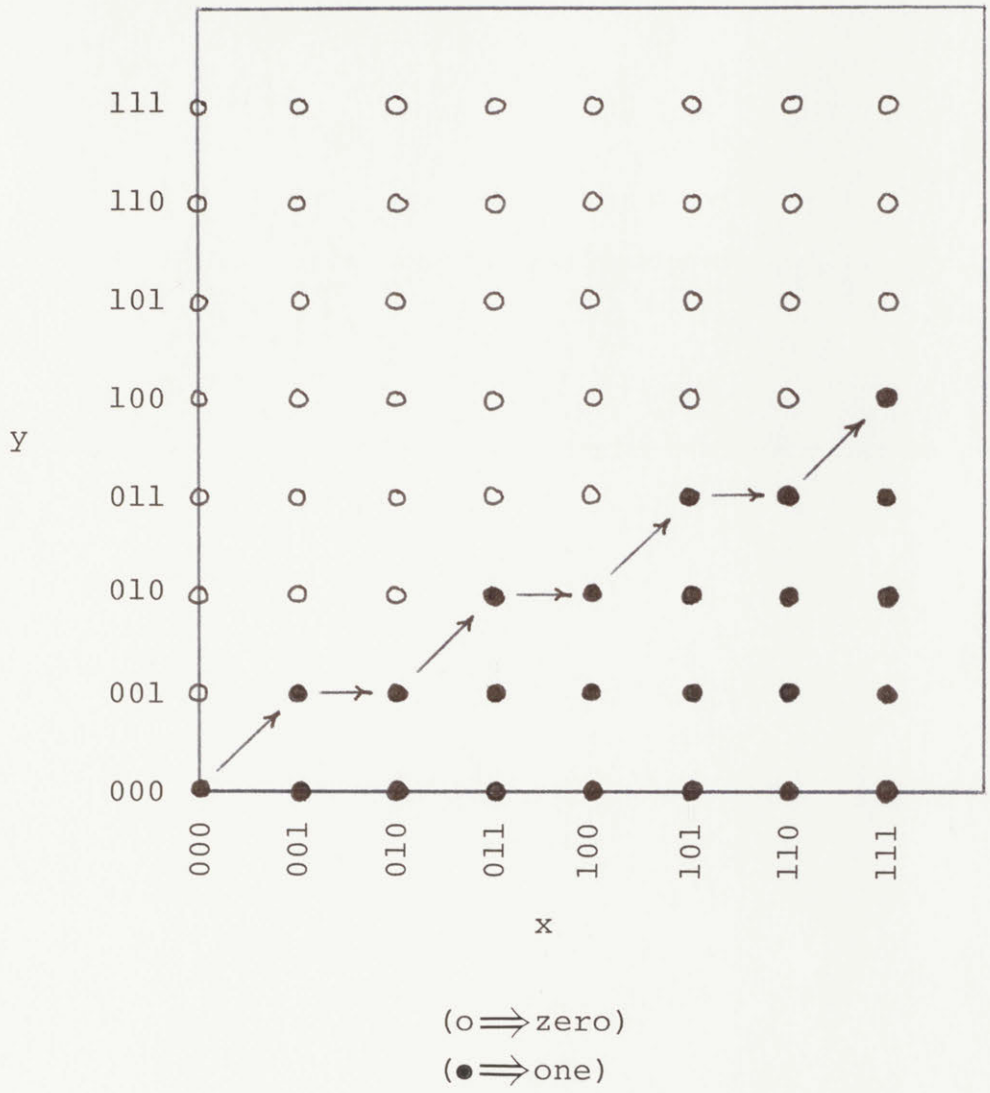


Figure 6-2. The Assignment Area RAM for a system with 3 bits of precision.

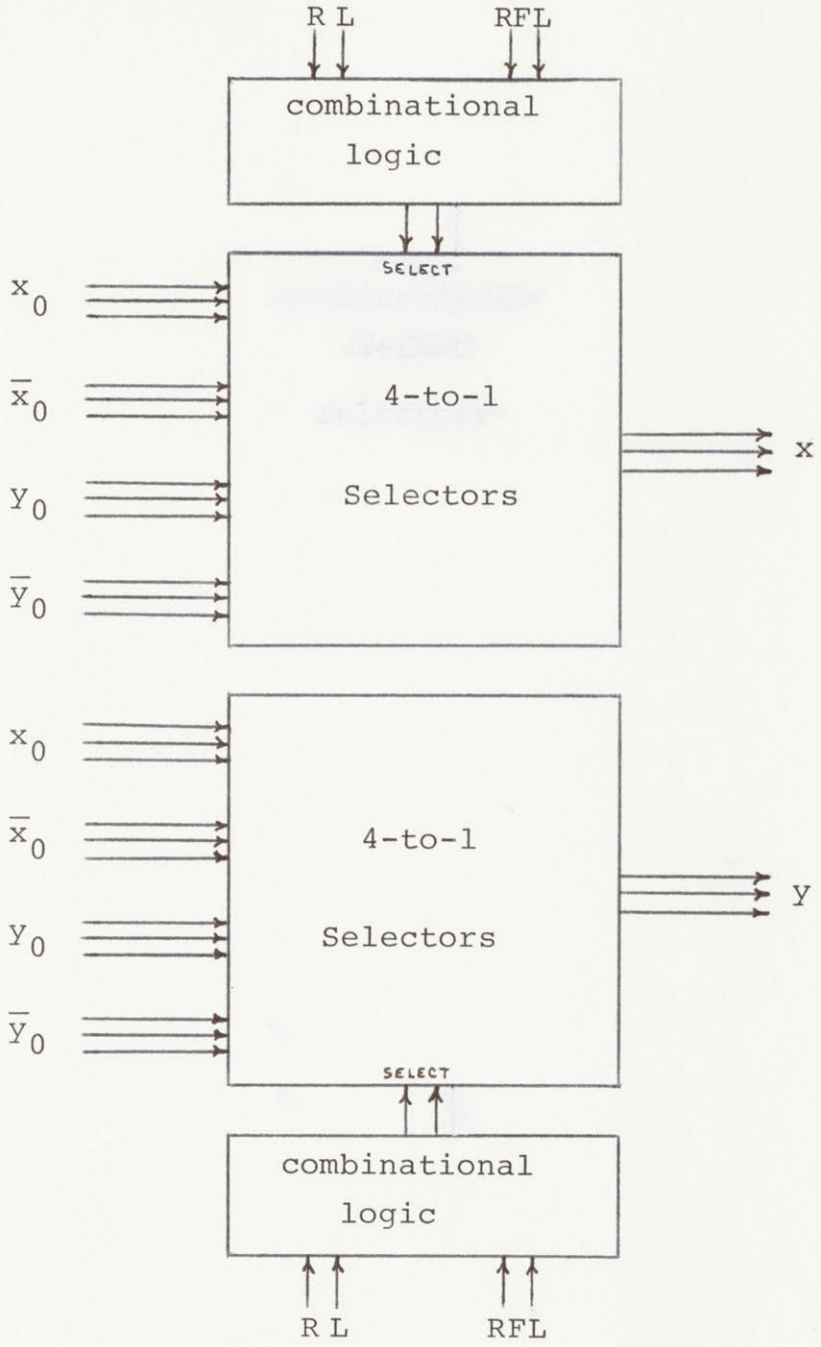


Figure 6-3. Derotating and Dereflecting Local Coordinates.

Given Image

Compressed Image

$s_x = \alpha < 1.0$

$s_y = 1.0$

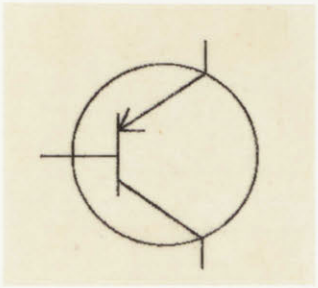
De-compressed Image

$s_x = 1/\alpha$

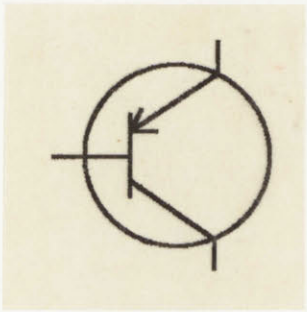
$s_y = 1.0$

Figure 6-4. Use of Coincident Resampling as a data compression tool.

resolution = 200 lines/inch

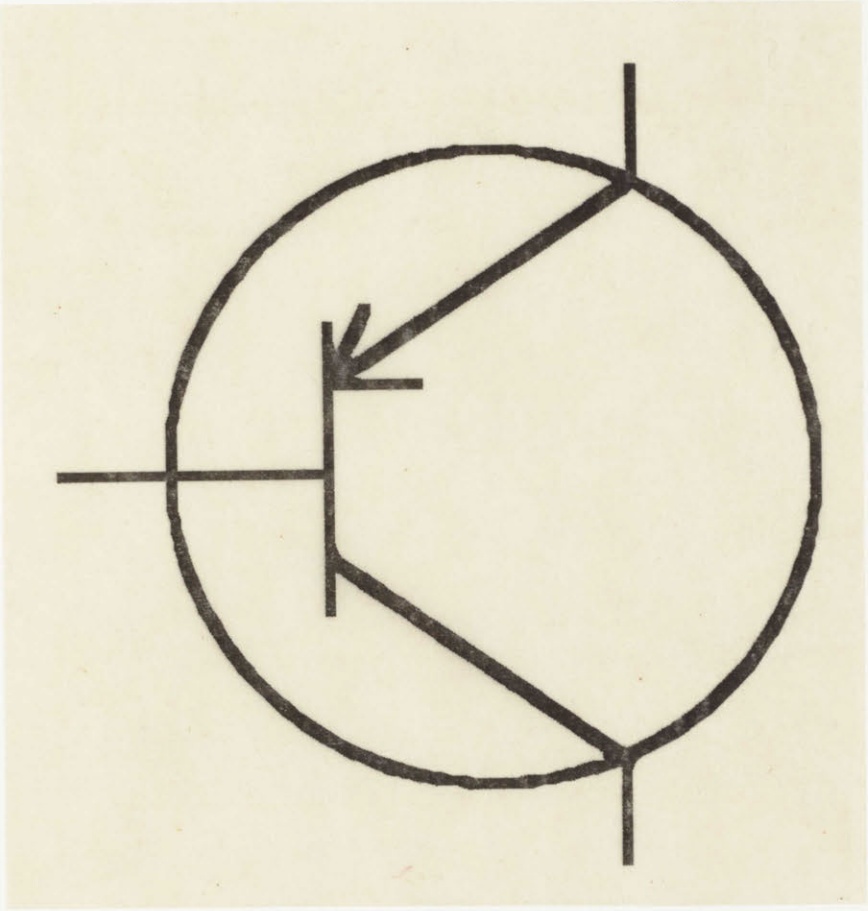


Given Image



$s_x = s_y = 1.0$

$d_H = d_V = 3$



$s_x = s_y = 3.5$

Figure 6-5. Enlarging a thin-line drawing by means of the Telescoping Template.

REFERENCES

1. H.P. Barnett, Computer Typesetting, M.I.T. Press, Cambridge, Mass., 1965.

2. "Photo Typesetter Comparison Charts of Second Generation Machines", The Seybold Report, Vol. 6, No. 6, Nov. 22, 1976.

3. "Beyond the Second Generation: CRT (and LASER) Typesetters", The Seybold Report, Vol. 7, No. 9, Jan. 9, 1978.

4. G.B. Levine, H. Jelke, "Specialized Hardware for Newspaper Processing", Electronic Progress, Vol. 18, No. 2, pp. 9-14, Summer, 1976.

5. G.W. Evans, R.L. Coswell, (Rockwell Int. Corp.), U.S. Patent No. 4029947, June 14, 1977.

6. "The Linotron 202: Better Than Anybody Anticipated!" The Seybold Report, Vol. 7, No. 21, July 17, 1978.

7. A.C. Bordogan, A. Kutemplon, "The RAYCOMP-100: A Video Page Composition Terminal", Electronic Progress, Vol. 18, No. 2, pp. 15-22, Summer, 1976.

8. "Raytheon, The CAM and the RAYCOMP-100", The Seybold Report, Vol. 4, No. 23, pp. 23-1/16, Aug. 18, 1975.

9. R.B. Arps, R.L. Erdmann, A.S. Neal, C.E. Schlaepfer, "Character Legibility vs. Resolution in Image Processing of Printer Matter", IEEE Trans. Man-Machine Systems, Vol. MMS-10, No. 3, pp. 66-71, Sept., 1969.

10. G.O. Walter, "Typographical Fonts for the RCA Video-Comp System", RCA Engineer, Vol. 13, No. 6, pp. 22-24, April/May, 1968.

11. W.F. Schreiber, "Reproduction of Graphical Data by Facsimile", M.I.T., Quarterly Progress Report 84, pp. 291-293, Jan., 1967.

12. R.B. Arps, "An Introduction and Digital Facsimile Compression Review", IEEE Int. Conf. on Communication, Vol. 1, pp. 7-1/3, June 16-18, 1975.

13. H.G. Musmann, D. Preuss, "Comparison of Redundancy Reducing Codes for Facsimile Transmission of Documents", IEEE Trans. Communications, Vol. COM-25, No. 11, pp. 1425-1433, Nov., 1977.

14. M. Takagi, T. Tsudo, "Comparison of Facsimile Bandwidth Compression Using Two-Dimensional Prediction and Signal Modification", IEEE Int. Conf. on Communications, pp. 47-26/31, June, 1976.

15. L. Arena, G. Zarone, "Analysis of Some Systems for the Redundancy Reduction of the Facsimile Signal", Alta Frequenza, Vol. 45, No. 6, pp. 348-357, June, 1976.

16. T.S. Huang, "Coding of Two-Tone Images", IEEE Trans. Communications, Vol. COM-25, No. 11, pp. 1406-1424, Nov. 1977.

17. D. Preuss, "Comparison of Two-Dimensional Facsimile Coding Schemes", IEEE Int. Conf. on Communications, Vol. 1, pp. 7-12/16, June 16-18, 1975.

18. D.A Huffman, "A Method for the Construction of Minimum-Redundancy Codes," Proc. I.R.E., Vol. 40, pp. 1098-1101, Sept. 1952.

19. H. Meyr, H.G. Rosdolsky, T.S. Huang, "Optimum Run Length Codes," IEEE Trans. Communications, Vol. COM-22, No. 6, pp. 826-835, June 1974.

20. G. Renelt, "Easily Decodable Runlength Code (EDRC) for Source Endcoding of Black-and-White Facsimile Pictures," Electronics Letters, Vol 12, No. 23, Nov. 11, 1976.

21. M. Takagi, T. Tsuda, "A Highly Efficient Run-Length Coding Scheme for Facsimile Transmission," Electronics and Communications in Japan, Vol. 58-A, No. 2, pp. 30-38, 1975.

22. D.R. Weber, "An Adaptive Run Length Encoding Algorithm," IEEE Int. Conf. on Communications, Vol. 1, pp. 7-8/11, June 16-18, 1975.

23. T.S. Huang, "Runlength Coding and Its Extentions," in Picture Bandwidth Compression, ed. by T.S. Huang and O.J. Tretiak, New York: Gordon and Beach, 1972.

24. W.F. Schreiber, T.S. Huang, O.J. Tretiak, "Contour Coding of Images," 1968 Wescon Rec., Sess. 8.3.1-8.3.5.

25. T.H. Morrin, II, "Recursive Contour Coding of Nested Objects in Black/White Images," IEEE Int. Conf. on Communications, Vol. 1, pp. 7-17/21, June 16-18, 1975.

26. T. H. Morrin, II, "Chain-Link Compression of Arbitrary Black/White Images," Computer Graphics and Image Processing, Vol. 5, No. 2, pp. 172-189, June 1976.

27. H. Freeman, "On the Encoding of Arbitrary Geometric Configurations," IRE Trans. on Electronic Computers, Vol. EC-10, No. 2, pp. 260-268, June 1961.

28. P.J.M. Coueignoux, "Compression of Type Faces by Contour Coding," M.S. Thesis, M.I.T., Cambridge, MA, Jan. 1973.

29. D.E. Knuth, (Computer Science Dept., Stanford Univ.) "Mathematical Typography", STAN-CS-78-648, Feb. 1978.

30. T.S. Huang, A.B.S. Hussian, "Facsimile Coding by Skipping White," IEEE Trans. on Communications, Vol. COM-23, No. 12, pp. 1452-1466, Dec. 1975.

31. F. DeCoulon, O. Johnson, "Adaptive Block Scheme for Source Ecoding of Black-and-White Facsimile," Electronic Letters, Vol. 12, No. 3, Feb. 5, 1976.

32. T.S. Huang, "Coding of Two Tone Images," Proc. of the Society of Photo-Optical Instrumentation Engineers, Vol. 66, pp. 5-8, Aug. 21-22, 1975.

33. J.I. Biegeleisen, Art Directors' Workbook of Type Faces, New York: Arco Publishing Co., 1976.

34. "WAD to RR, A Letter About Designing Type," Harvard College Library Department of Printing and Graphic Arts, 1940.

35. Conversation with Raymond Pell, Manager of Typography, Rockwell MGD, Downers Grove, ILL., June 27, 1978.

36. W.K. Pratt, DIGITAL IMAGE PROCESSING, New York: John Wiley and Sons, 1978.

37. G. Kang, "Digital Image Processing," Quest (TRW Defense and Space Systems Group), Vol. 1, No. 2, pp. 3-22, Autumn 1977.

38. D.E. Troxel, "Computer Editing of News Photos," IEEE Trans. Systems, Man, and Cybernetics, Vol. SMC-5, No. 6, pp. 625-629, Nov. 1975.

39. D.E. Troxel, C. Lynn, "Enhancement of News Photos," Comp. Graphics and Image Processing, Vol. 7, No. 2, pp. 266-281, April 1978.

40. S.N. Jones, "Enlarging and Reducing Digital Images," B.S. Thesis, M.I.T., Cambridge, MA., June 1977.

41. A.V. Oppenheim, R.W. Shafer, Digital Signal Processing, Englewood Cliffs, N.J.: Prentice-Hall, Inc., pp. 115-120, 1975.

42. G.C. Huang, F.D. Russell, W.H. Chen (Ford Aerospace), "Pattern Recognition by Mellon Transform," Fifth Annual Symposium on Imagery Pattern Recognition, April 17-18, 1975.

A P P E N D I C E S

APPENDIX A

ALPHABETIC LISTING OF VARIABLES

a — Concentric Code level ($1 \leq a \leq p$).

A_{mn} — 1 by 1 continuous Assignment Area associated with a given sample at G[m,n].

b_i — Coefficients of the interpolating polynomial comprising the assignment rule. i = 0, ..., t.

$C_a^D[q]$ — De-rotated Window Concentric Code, level a.

$C_a^T[h,q]$ — h th Template Concentric Code, level a.

$C_a^W[q]$ — Window Concentric Code, level a.

d_H, d_V — The horizontal and vertical dimensions of a Master Reduction Template used for "Binary Convolution".

$D_p[i,j]$ — p th order De-rotated Window.

$E_B[n,i]$,
$E_T[n,i]$ Bottom and Top Edge pel number of the i th black run in the n th line.

$G[m,n]$ The M by N Given Binary Image.

$h(x,y)$ A general interpolation function.

h,H h is the specific template number from a repertoire of H templates.

$I[n]$ The number of black runs in line n.

k,K Pel number of the Scaled Image $(1 \leq k \leq K)$.

ℓ,L Line number of the Scaled Image $(1 \leq \ell \leq L)$.

m,M Pel number of the Given Image $(1 \leq m \leq M)$.

n,N Line number of the Given Image $(1 \leq n \leq N)$.

p Order of scaling; Size of image window is 2p by 2p.

$Q_N(x)$ The ideal interpolation fucion used to reconstruct the continuous image represented by N DFT samples.

RL — Number of Rotations to the Left needed for a window to become standard form.

RFL — 0 => no reflection; 1 => horizontal reflection; 2 => transposal reflection.

$R_{sx}[j]$ — Horizontal Resampling Sequence used in the Coincident Resampling method.

s_x — Given Horizontal Scale Factor.

s_y — Given Vertical Scale Factor.

$S[k,\ell]$ — K by L Scaled Binary Image.

t — Degree of interpolating polynomial comprising the assignment rule. (Only t = 1 was used in this study.)

$T_p[h,i,j]$ — Set of H p th order Templates.

U[m] — Vertical Occupation Number; number of new sample rows in A_{mn}.

V[n] — Horizontal Occupation Number; number of new sample columns in A_{mn}.

$W_p[i,j]$ p th order discrete-space binary Window.

$x_n[v]$,
$y_m[u]$ Local coordinates within A_{mn}. (Sometimes simply designated x_0, y_0)

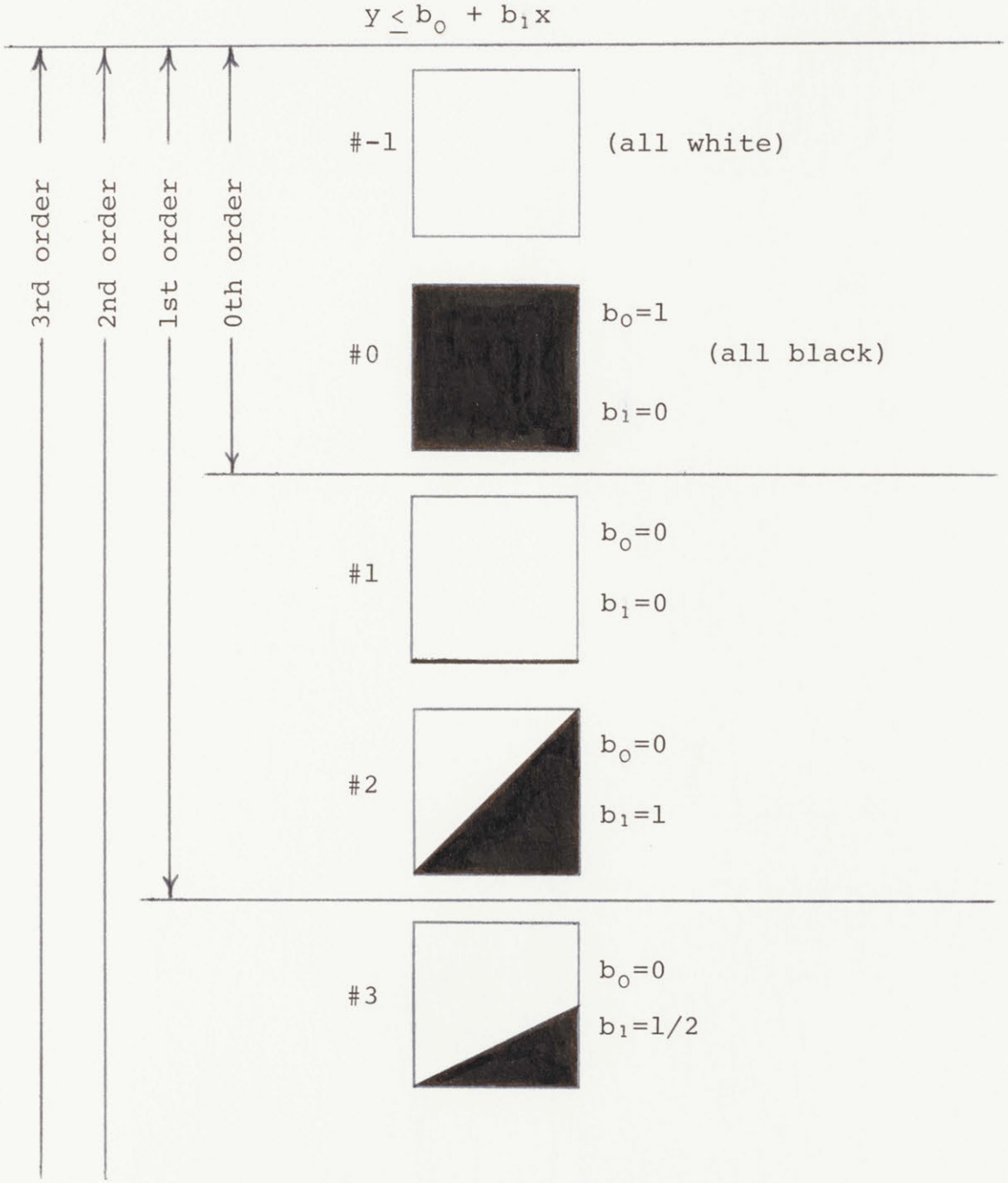
$X[\ell]$ Horizontal Relative Position Array; a set of real numbers between 1 and N that describe where the columns of the new sample grid lie.

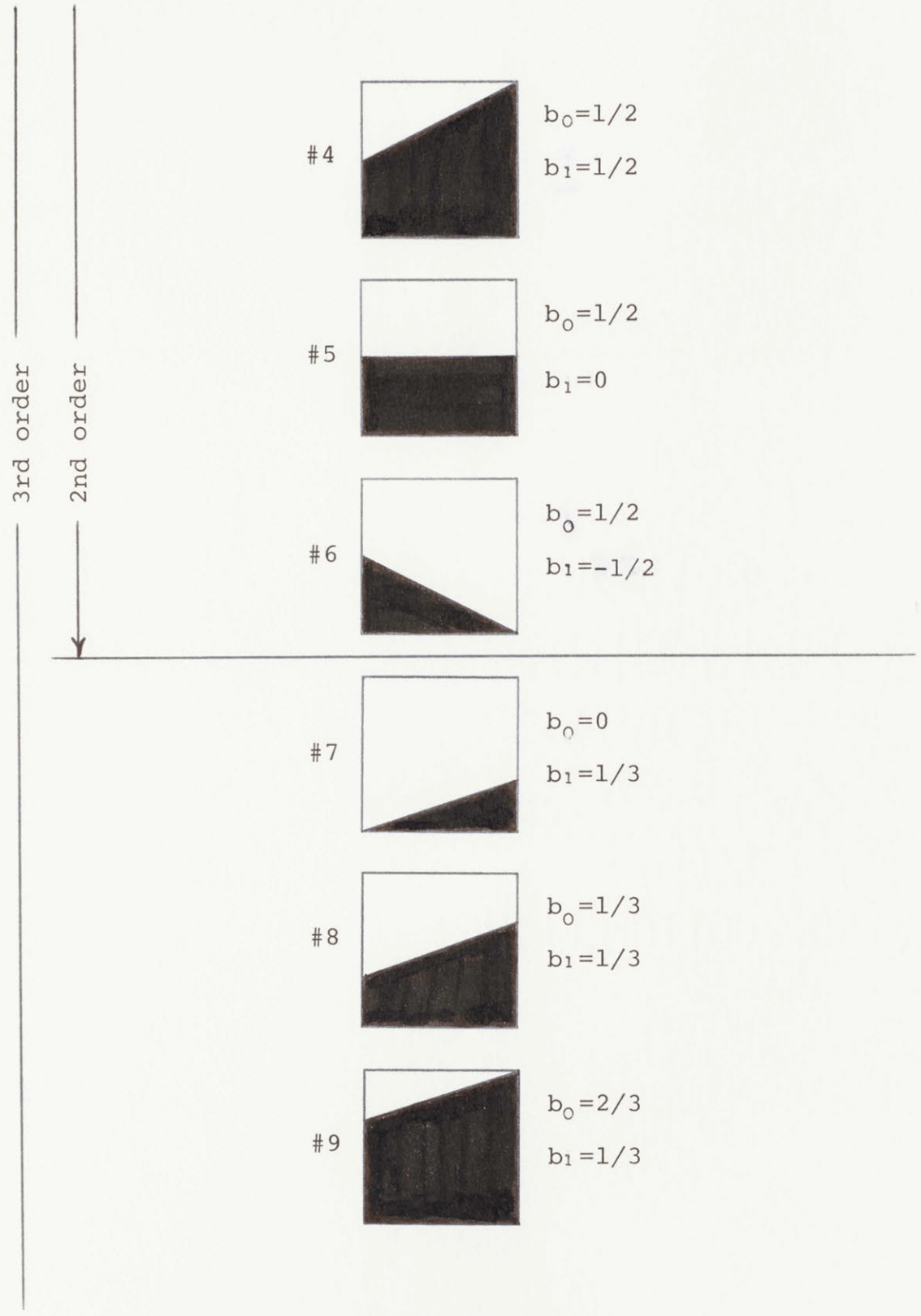
$Y[k]$ Vertical Relative Position Array; a set of real numbers between 1 and M that describe where the rows of the new sample grid lie.

$Z_{mn}[u,v]$ Matrix of New Samples falling in A_{mn}.
$u = 1, \ldots, U[m]; \quad v = 1, \ldots, V[n]$.

APPENDIX B

ASSIGNMENT RULES




3rd order
2nd order
#4
$b_0=1/2$
$b_1=1/2$
#5
$b_0=1/2$
$b_1=0$
#6
$b_0=1/2$
$b_1=-1/2$
#7
$b_0=0$
$b_1=1/3$
#8
$b_0=1/3$
$b_1=1/3$
#9
$b_0=2/3$
$b_1=1/3$

#10

$b_0 = 0$

$b_1 = 2/3$

#11

$b_0 = 2/3$

$b_1 = 2/3$

#12

$b_0 = -1/3$

$b_1 = 2/3$

#13

$b_0 = 1/3$

$b_1 = 2/3$

#14

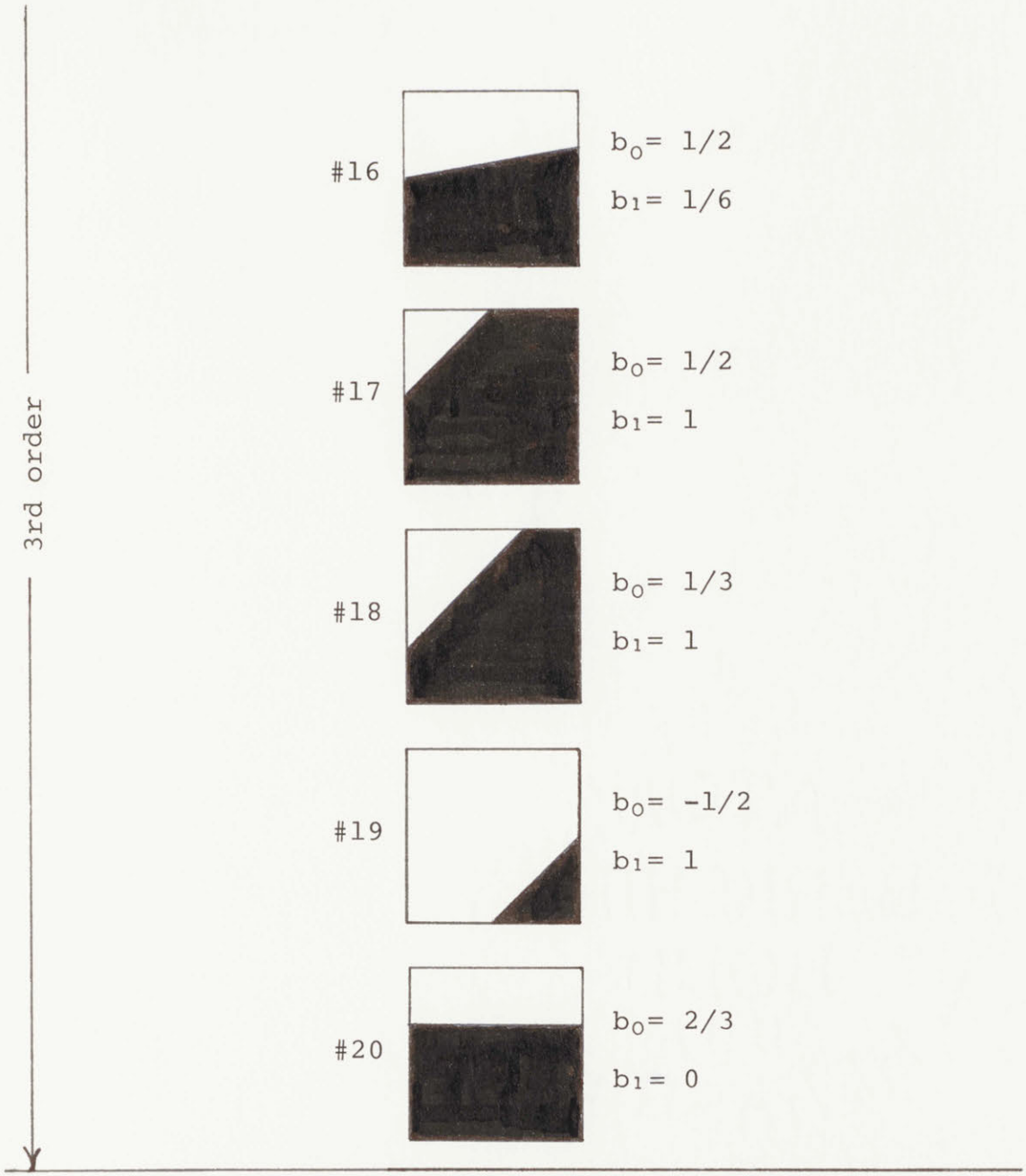
$b_0 = 1/3$

$b_1 = 0$

#15

$b_0 = 1/3$

$b_1 = -1/3$


3rd order
#16
$b_0 = 1/2$
$b_1 = 1/6$
#17
$b_0 = 1/2$
$b_1 = 1$
#18
$b_0 = 1/3$
$b_1 = 1$
#19
$b_0 = -1/2$
$b_1 = 1$
#20
$b_0 = 2/3$
$b_1 = 0$

APPENDIX C

FIRST ORDER TEMPLATE REPERTOIRE

Template #1 — Skip--All White

0 0
0 0

Assignment Rule #-1

Template #2 — All Black

1 1
1 1

Assignment Rule #0

$b_0 = 1$
$b_1 = 0$

Template #3 — Skip--assume all white

0 0
0 1

Assignment Rule #-1

Template #4 — Straight Edge

0 0
1 1

Assignment Rule #1

$b_0 = 0$
$b_1 = 0$

Template #5

Slope = 1

0 1
1 1

Assignment Rule #2

$b_0 = 0$
$b_1 = 1$

(Note: 0 1 and 1 0 are ignored, i.e. Assignment Rule #-1)
1 0 0 1

APPENDIX D

SECOND ORDER TEMPLATE REPERTOIRE

Template #1

Slope = 1/2

```
. 0 0 0
0 0 0 1
0 1 1 1
. 1 1 .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #2

Slope = 1/2

```
. 0 0 0
0 0 0 1
1 1 1 1
. 1 1 .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #3

Slope = 1/2

```
. 0 0 .
0 0 1 1
1 1 1 1
. 1 1 .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #4

Outside Cusp

```
. 0 0 .
0 0 1 0
1 1 1 1
. 1 1 .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #5

Sharp Outside Corner

```
. 0 0 .
0 0 1 0
1 1 1 0
. 1 1 .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #6

Sharp Inside Corner

```
1 0 0 0
1 0 0 1
1 1 1 1
. 1 1 .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #7

Tight Inside Curve

```
0 0 0 0
1 0 0 1
1 1 1 1
. 1 1 .
```

Assignment Rule #5

$b_0 = 1/2$

$b_1 = 0$

Template #8

Inside Cusp

```
. 1 1 .
0 0 0 1
0 1 1 1
. 1 1 .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #9

Inside Cusp

```
. 1 1 .
0 0 0 1
1 1 1 1
. 1 1 .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #10

Inside Cusp

```
1 1 1 .
0 0 1 1
1 1 1 1
. 1 1 .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #11

Tight Inside Corner

```
1 0 0 .
1 0 1 1
1 1 1 1
. 1 1 .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #12

Tight Inside Corner
(Slope = -1/2)

```
0 0 1 .
1 0 1 1
1 1 1 1
. 1 1 .
```

Assignment Rule #6

$b_0 = 1/2$

$b_1 = -1/2$

Template #13

Inside Corner

```
. 0 1 .
0 0 1 1
1 1 1 1
. 1 1 .
```

Assignment Rule #-1

APPENDIX E

THIRD ORDER TEMPLATE REPERTOIRE

Template #1

Slope = 1/2

```
. . . . . .
. . 0 0 0 0
. 0 0 0 1 .
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #2

Slope = 1/2

```
. . . . . .
. . 0 0 0 .
. 0 0 1 1 .
0 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #3

Slope = 1/3

```
. . . . . .
. . 0 0 0 0
0 0 0 0 0 1
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #4

Slope = 1/3

```
. . . . . .
. . 0 0 0 0
. 0 0 0 0 1
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #5

Slope = 1/3

```
. . . . . .
. . 0 0 0 .
. 0 0 0 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #8

$b_0 = 1/3$

$b_1 = 1/3$

Template #6

Slope = 1/3

```
. . . . . .
. . 0 0 0 .
. 0 0 1 1 .
1 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #9

$b_0 = 2/3$

$b_1 = 1/3$

Template #7

Slope = 2/3

```
. . . . . .
. . 0 0 0 1
. 0 0 0 1 .
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #10

$b_0 = 0$

$b_1 = 2/3$

Template #8

Slope = 2/3

```
. . . . . .
. . 0 0 1 .
0 0 0 1 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #11

$b_0 = 2/3$

$b_1 = 2/3$

Template #9

Slope = 2/3

```
. . . . . .
. . 0 0 0 .
. 0 0 0 1 .
0 0 0 1 1 .
. 1 1 1 . .
. . . . . .
```

Assignment Rule #12

$b_0 = -1/3$

$b_1 = 2/3$

Template #10

Slope = 2/3

```
. . . . 0 .
. . 0 0 . .
. 0 0 1 1 .
0 0 1 1 1 .
1 1 1 1 . .
. . . . . .
```

Assignment Rule #13

$b_0 = 1/3$

$b_1 = 2/3$

Template #11

Connecting a 2/3 and 3/2 sloped line; Case 1

```
. . . 0 1 .
. . 0 0 1 .
. 0 0 1 1 .
0 0 1 1 1 .
1 1 1 1 . .
. . . . . .
```

Assignment Rule #18

$b_0 = 1/3$

$b_1 = 1$

Template #12

Connecting a 2/3 and 3/2 sloped line; Case 2

```
. . . . . .
. . 0 0 1 .
. 0 0 0 1 .
. 0 0 1 1 .
. 1 1 1 . .
. . . . . .
```

Assignment Rule #19

$b_0 = -1/2$

$b_1 = 1$

Template # 13

Inside Corner

```
. . 0 1 . .
. . 0 1 . .
0 0 0 1 1 .
1 1 1 1 1 .
. . . . . .
```

Assignment Rule #-1

Template #14

Round Inside Corner

```
. . 0 0 . .
. . 0 1 . .
0 0 0 1 1 .
0 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #17

$b_0 = 1/2$

$b_1 = 1$

Template #15

Wide Inside Corner, Slope = 1/2

```
. . . 1 . .
. . 0 1 . .
. 0 0 1 1 .
0 1 1 1 1 .
. . 1 1 . .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #16

```
. . . . . .
. . 0 0 1 .
. 0 0 0 1 .
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Wide Inside Corner,
Slope = 1/2

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #17

```
0 . . 1 . .
. 0 0 1 . .
. 1 0 1 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Tight Inside Corner,
Slope = -1/2

Assignment Rule #6

$b_0 = 1/2$

$b_1 = -1/2$

Template #18

```
. 1 . . 0 .
. 1 0 0 . .
. 1 0 1 1 .
. 1 1 1 1 .
. . . . . .
```

Tight Inside Corner,
Slope = 1/2

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #19

```
. . . 1 . .
0 0 0 1 . .
1 0 0 1 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Tight Inside Corner,
Slope = -1/3

Assignment Rule #15

$b_0 = 1/3$

$b_1 = -1/3$

Template #20

```
. 1 . . . .
. 1 0 0 0 .
. 1 0 0 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Tight Inside Corner,
Slope = 1/3

Assignment Rule #8

$b_0 = 1/3$

$b_1 = 1/3$

Template #21

```
. 1 . . . .
. 1 0 0 0 0
. 1 0 0 0 1
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

T``ight Inside Corner,
Slope = 1/3

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #22

Outside Cusp, Slope = 1/2

```
. . . . . .
. . 0 0 . .
. 0 0 1 0 .
0 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #23

Outside Cusp, Slope = 1/3

```
. . . . . .
. . 0 0 . .
. 0 0 1 0 .
1 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #9

$b_0 = 2/3$

$b_1 = 1/3$

Template #24

Outside Cusp, Slope = 2/3

```
. . . . . .
. . 0 0 0 .
. 0 0 1 0 .
0 0 1 1 1 .
1 1 1 1 . .
. . . . . .
```

Assignment Rule #13

$b_0 = 1/3$

$b_1 = 2/3$

Template #25

Sharp Outside Corner,
Slope = 2/3

```
. . . . . .
. . 0 0 0 .
. 0 0 1 0 .
0 0 1 1 0 .
1 1 1 1 . .
. . . . . .
```

Assignment Rule #13

$b_0 = 1/3$

$b_1 = 2/3$

Template #26

Sharp Outside Corner,
Slope = 1/3

```
. . . . . .
. . 0 0 . .
. 0 0 1 0 .
1 1 1 1 0 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #9

$b_0 = 2/3$

$b_1 = 1/3$

Template #27

Sharp Outside Corner,
Slope = 1/2

```
. . . . . .
. . 0 0 . .
. 0 0 1 0 .
0 1 1 1 0 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #28

Wide Inside Curve

```
. . . . . .
0 0 0 0 0 0
1 0 0 0 0 1
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #14

$b_0 = 1/3$

$b_1 = 0$

Template #29

Medium Inside Curve

```
. . . . . .
0 0 0 0 0 .
1 0 0 0 1 .
1 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #16

$b_0 = 1/2$

$b_1 = 1/6$

Template #30

Tight Inside Curve

```
. . . . . .
. 0 0 0 0 .
. 1 0 0 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #20

$b_0 = 2/3$

$b_1 = 0$

Template #31

Inside Cusp, Slope = 1/2

```
. . . . . .
. . 1 1 . .
. 0 0 0 1 .
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #32

Inside Cusp, Slope = 1/2

```
. . . . . .
. . 0 1 . .
. 0 0 0 1 .
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #3

$b_0 = 0$

$b_1 = 1/2$

Template #33

Inside Cusp, Slope = 1/2

```
1 1 . . . .
. . 1 1 . .
. 0 0 1 1 .
0 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #4

$b_0 = 1/2$

$b_1 = 1/2$

Template #34

Inside Cusp, Slope = 1/3

```
. . 1 1 . .
. . 0 0 1 .
. 0 0 0 0 1
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #35

Inside Cusp, Slope = 1/3

```
. . . . . .
. . 0 1 1 .
. 0 0 0 0 1
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #36

Inside Cusp, Slope = 1/3

```
. . . . . .
. . 1 1 1 .
. 0 0 0 0 1
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #37

Inside Cusp, Slope = 1/3

```
. . 1 1 . .
. . 0 0 1 .
. 0 0 0 0 1
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #38

Inside Cusp, Slope = 1/3

```
. . . . . .
. . 0 1 1 .
. 0 0 0 0 1
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #39

Inside Cusp, Slope = 1/3

```
. . . . . .
. . 1 1 1 .
. 0 0 0 0 1
. 0 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #7

$b_0 = 0$

$b_1 = 1/3$

Template #40

Inside Cusp, Slope = 1/3

```
. . . . . .
. . 1 1 . .
. 0 0 0 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #8

$b_0 = 1/3$

$b_1 = 1/3$

Template #41

Inside Cusp, Slope = 1/3

```
. 1 1 . . .
. . 0 1 . .
. 0 0 0 1 .
. 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #8

$b_0 = 1/3$

$b_1 = 1/3$

Template #42

Inside Cusp, Slope = 1/3

```
1 1 . . . .
. . 1 1 . .
. 0 0 1 1 .
1 1 1 1 1 .
. . 1 1 . .
. . . . . .
```

Assignment Rule #9

$b_0 = 2/3$

$b_1 = 1/3$

Template #43

Inside Cusp, Slope = 2/3

```
. . . . . .
. . 0 1 1 .
. 0 0 0 1 .
0 0 0 1 1 .
. 1 1 1 . .
. . . . . .
```

Assignment Rule #12

$b_0 = -1/3$

$b_1 = 2/3$

Template #44

Inside Cusp, Slope = 2/3

```
. . . . . .
. . 1 1 1 .
. 0 0 0 1 .
0 0 0 1 1 .
. 1 1 1 . .
. . . . . .
```

Assignment Rule #12

$b_0 = -1/3$

$b_1 = 2/3$

Template #45

Inside Cusp, Slope = 2/3

```
. . . . . .
. . 1 1 . .
. 0 0 1 1 .
0 0 1 1 1 .
1 1 1 1 . .
. . . . . .
```

Assignment Rule #13

$b_0 = 1/3$

$b_1 = 2/3$