

Bi-level Facsimile Compression with Unconstrained Tilings

Kenneth Zeger and Gopal Krishna

Dept. of Electrical Engineering
University of Hawaii
Honolulu, HI 96822

Abstract

A technique for bi-level facsimile compression is presented that extends to two-dimensional sequences the traditional notion of run length coding. The algorithm is relatively efficient and easy to implement and exploits two-dimensional neighboring pixel redundancies, unlike that of many existing systems employing entropy coding on raster scanned images. A bi-level image is spatially predicted and then partitioned into rectangles of arbitrary dimensions that cover either only white pixels or mixtures of black and white pixels. An entropy coded index is used to specify the quantized length and width values of each rectangle from top to bottom and left to right in the image so that the decoder can deduce the rectangle locations using a "water-filling" algorithm. The rectangles containing some black pixels are encoded using entropy coding.

1. Introduction

The compression of bi-level images for facsimile transmission has received considerable attention in the literature (e.g. [1,2,3,4]). Many of the proposed data compression schemes attempt to reduce the redundancy in the image by using statistical codes for encoding the positions of *transitive* pixels [5] (one whose value is different from the previously scanned pixel). Good compression performance can be obtained in these cases by reducing the number of transitive elements prior to coding [6]. Run-length coding [7], predictive coding [8], and READ coding [9] are known techniques which encode the positions of transitive elements. In run-length coding, the length of each white run (consecutive single valued pixels) is often entropy coded using such techniques as Huffman codes [10], Modified Huffman codes, [11] or Wyle codes [12], while scanning a picture sequentially line by line from top to bottom. In predictive coding, each pixel is typically predicted by some function of its causal neighbors. An error sequence is constructed by adding bitwise the input sequence to the prediction sequence. The error sequence is then usually encoded using one of a number of compression techniques, often based on run-length coding. "READ" coding, as well as RAC [13] and EDIC [14], encode the distance between the start position of a run and that of a certain previously scanned run which satisfies some prescribed conditions.

These coding schemes contain two common operations, namely that of selecting the start position of each run from a given picture, and then determining the positions of transitive elements, or equivalently the positions of 1's in the error sequence. Data compression is generally achieved through the latter operation, in which it is common to encode the distances between consecutive transitive elements, or equivalently, the length of runs.

On the other hand, a technique called Selective Element Coding (SECT) [15] attempts to reduce the number of transitive elements to be selected. One limitation with the above schemes is that they use one-dimensional redundancies only, and therefore, lack the capability of exploiting the dependencies of a pixel with its other spatially close neighbors.

Kunt [16] and Johnsen [17] proposed techniques which partition each picture into a set of two-dimensional blocks of size $n \times m$. They assign the codeword 0 to all-white blocks, while

codewords for other configurations are composed of the nm bits of the block preceded by the prefix 1. Cohen, et.al. [18] examined a class of two-dimensional image coding methods based upon recursive hierarchical decomposition of the image into uniform areas, which extends the idea of representing pictures by binary trees [19].

A technique for compression of two-dimensional data using Peano-Hilbert plane-filling curves [20] was investigated by Lempel and Ziv [21]. It is inherently one-dimensional in nature though, since a one-dimensional curve traces out a covering path through 2-dimensional space.

In the present paper we propose a new, easy to implement, technique for the lossless compression of bi-level images, which takes advantage of both transitional point coding, and white-space skipping. This technique, *Unconstrained Block Tiling (UBT)*, divides the problem of encoding for compression into two parts: (1) partitioning an image into "white" and "non-white" rectangles, and (2) compressing the non-white rectangles or regions. Throughout, we use the terminology "non-white rectangles" to denote rectangles containing at least one black pixel.

The basic idea is to first locate in a residual image (after prediction) large contiguous two-dimensional areas of white space and then to efficiently describe and transmit to the receiver enough information to perfectly reconstruct the rectangular partition. The remaining portion of the image is then compressed using entropy, run-length, or block coding. By describing the partition as a disjoint ordered sequence of rectangles with quantized horizontal and vertical dimensions, the compression problem can focus on a small fraction of the total image's pixels.

An alternative description of the rectangular partition is to specify the exact locations of the non-white rectangles. This description is sufficient for coding purposes, since only the locations of the non-white rectangles must be known in order to proceed with the coding process. There are thus two distinct protocols for conveying the partition information: (1) Sequentially transmit quantized lengths and widths of every rectangle in the partition, or (2) Transmit the exact coordinates of only the non-white rectangles. The choice of which method to use depends upon which is more efficient for the particular input image examined.

On one hand, specifying the quantized lengths and widths requires very few bits per rectangle compared to the number of bits required to specify the exact coordinates of a rectangle, whereas on the other hand the number of non-white rectangles might be only a small fraction of all the rectangles in the partition.

An underlying assumption here is that a large percentage of pixels be white, and that they be highly correlated, in the sense that pixels of the same color tend to cluster around each other. A consequence of this assumption is that the proposed technique may have limited utility for images from dithered sources. However, its performance for many typical facsimile images such as business letters and handwritten notes is shown to be superior to many existing techniques. One of the most significant problems to be addressed is that of how to partition the image into white and non-white rectangles. An effective and computationally acceptable algorithm is sought for this task.

The technique described generalizes the idea of run-length coding to two-dimensions in the following sense. One-dimensional run-length coding maximizes at each stage the size of the block (codeword) subject to the constraint that the block contain only white pixels. The two-dimensional scheme presented here maximizes the number of 0's in the block (the area) subject to the same constraint, that the block be all white.

The emphasis in this paper will be on the description of the tiling process rather than the very specific details of how to entropy code the non-white rectangles, since that is a well-studied subject already. Known lossless data compression techniques can readily be applied in that portion of the algorithm.

In Section 2, we describe the initial preprocessing stage of prediction, prior to blocking and compression. Section 3 presents the unconstrained tiling concept and discusses the various different methods available. In Section 4, certain critical implementation issues are examined, dealing primarily with the desire to keep the computational complexity low. Finally, in Section 5, experimental results are presented and analyzed.

2. Prediction Techniques

Bi-level prediction techniques exploit the inter-dependencies of neighboring pixels to create a *predicted image*, based on the original image. The value of each pixel in the predicted image is computed using a *prediction function*, a mapping of particular neighboring pixels forming a *prediction window* to a predicted value of either zero or one. A *residual image* is obtained by subtracting pixel by pixel (or adding, since arithmetic is binary) the predicted image from the original image. The residual image is then processed for compression, in our scheme by tiling followed by selective entropy coding. The use of prediction and residual coding here is analogous to lossy differential source coding techniques such as Differential Pulse-Code Modulation (DPCM).

If the original image contains large amounts of spatial redundancy, then a good predictor will yield a residual image with many clustered zeros. An important constraint in this sliding window approach is that the window be chosen so as to maintain spatial causality. That is, the pixels in a prediction window can only consist of those that have been previously predicted (and subsequently residually encoded).

There are many different possible approaches to designing bi-level predictors. For quantization systems with analog inputs, linear prediction techniques are usually used, though computationally feasible mean-square optimal non-linear prediction can be achieved if a small prediction window is used and the input range is restricted to a small finite set of values [22]. For bi-level images, however, it is quite easy to achieve optimal nonlinear prediction. *Optimal* in this situation refers to maximizing the expected number of correct predictions. For a given prediction window containing n pixels, the optimal predictor is simply a lookup table with 2^n entries. Each entry is a 0 or a 1, and is addressed by an n -bit word formed from the pixels currently in the prediction window. That is, if the prediction window contains the elements x_1, \dots, x_n , and the pixel to be predicted is y , then the optimal predicted value is the Maximum Likelihood estimate

$$f(x_1, \dots, x_n) = \begin{cases} 0 & \text{if } Pr[y=0 | x_1, \dots, x_n] > 1/2 \\ 1 & \text{if } Pr[y=1 | x_1, \dots, x_n] > 1/2 \end{cases}$$

That is, a 1 is predicted if the conditional probability of a 1 given the window pattern is greater than 1/2, otherwise a 0 is predicted.

3. Tiling the Residual Image

Following prediction, the residual image typically contains many white pixels and relatively few black pixels. We propose to partition the residual image with rectangles, some designated as *white* if they cover only white pixels, and the others designated as *non-white* if they cover at least one black pixel. The rectangles available for covering are to be chosen from a limited set, whose vertical and horizontal dimensions are quantized values. More traditional block oriented approaches partition, or tile, the residual image with equal sized squares, as opposed to our unconstrained tiling.

As an example of unconstrained tiling, a typical high resolution scanned facsimile image might contain 5856 rows and 4096 columns. To specify every possible rectangle that could fit inside such an image with a fixed length binary word would require 50 bits per rectangle; that is, 25 bits each to specify the upper left and lower right corner locations for each rectangle. We might typically quantize the dimensions to allow, say, 8 lengths and 8 different widths yielding a total of 6 bits to specify each rectangle's dimensions. In addition one extra bit would be sent for each rectangle to specify whether it is white or non-white. Furthermore, more savings can be achieved by Huffman coding the 2^6 different rectangle (length, width) pairs, whereas this would be quite impractical or useless for the rectangles' corner locations. Thus, in this example, it would be more efficient to transmit the rectangles quantized and entropy coded lengths and widths provided that the number of non-white rectangles is no more than roughly 10% of the total number of rectangles in the partition.

3.1. Rectangular Partition Transmission and Decoding

Once a residual image is partitioned into rectangles, the exact partition must be conveyed to the receiver. We present an efficient technique for accomplishing this task. The main idea is that an entropy coded index describing each rectangle is transmitted precisely when that rectangle's top left pixel becomes the first pixel in the image not to have yet been covered by a transmitted rectangle when the image's pixels are searched from the top row to the bottom and left to right. This algorithm can be described by the following pseudo-code. It is assumed that the functions $x\text{dim}(i,j)$ and $y\text{dim}(i,j)$ output the quantized length and width of a rectangle in the partition whose upper left pixel has coordinates (i,j) . It should be reminded that in addition to the length and width information, one additional bit is transmitted per rectangle to specify if it is all white.

TRANSMIT_PARTITION:

```

For i=1 to N
  For j=1 to N
    If COVERED(i,j) == 0
      For k=1 to xdim(i,j)
        For l=1 to ydim(i,j)
          COVERED(k,l)=1
        SEND(i,j)
      Next l
    Next k
  Endif
Next j
Next i

```

The receiver takes each received entropy coded index and decodes the length and width of the next rectangle in the formation of the partition. In a similar manner as in the encoder, the first uncovered pixel found while scanning top to bottom and left to right is taken as the top left pixel of a rectangle whose dimensions are those of the decoded entropy index. The appropriate pixels corresponding to the added rectangle are then marked as "covered" and the process repeats itself as new indices arrive, until the entire image is covered. A nice feature of this transmission scheme is that it works correctly and efficiently no matter how the rectangular partition was constructed. The receiver also decodes a single bit per rectangle that describes whether it is a white or non-white rectangle. This informs the receiver at a later stage about how to reconstruct the bit patterns inside the non-white rectangles from transmitted entropy coded information.

While the decoding and partition reconstruction process are essentially straightforward, the most effective method of constructing the partition at the encoder is not as obvious. In the following section we discuss some techniques for this process.

3.2. Rectangular Partition Construction

Different rectangular partitions of the residual image yield different performance. This is because of the fact that one partition may cover more white space than another partition, or may select non-white rectangles in a manner very amenable to compression. One effective guideline to follow in designing the partition is to attempt to maximize the total area covered using white rectangles using as few transmitted bits as possible. A similar *greedy* approach is to simply attempt to lay down rectangles one at a time with the goal of maximizing the total white space covered by that rectangle. A more optimal "global" approach would be to jointly lay down rectangles to optimize the total number of bits needed; this however is difficult to accomplish and is computationally burdensome. Throughout, we take the greedy approach, primarily for the purpose of computational convenience. In the following sections we present two distinct methods for tiling the residual image with rectangles. Both methods are greedy in nature, and work well in practice.

3.3. Sequential Tiling

One method of greedily tiling an image with rectangles is to do so in the same order as when the partition is transmitted. That is, at a given iteration the next rectangle to be included in the partition has its upper left corner in the position of the first uncovered white pixel seen in the image when scanning from top to bottom and left to right. Anchoring this pixel as the top left corner, we examine the set of all rectangles emanating from this point which contain only white pixels and do not intersect previously defined rectangles. Of these candidate rectangles, the one with the largest area is chosen as the next rectangle in the partition. As there may be many such rectangles, the computational burden of determining the most voluminous can be great. As a result we present a complexity reduction technique for finding the rectangle with the most area at each stage. The algorithm is based on the observation that for a given width W , if the largest length that can yield an all-white rectangle is L , then for widths larger than W , the largest admissible length is no longer than L . The sequential tiling technique is illustrated in Figure 1, where each rectangle is numbered in the order it is laid down.

We state below the Maximum Area algorithm. The variable N is the maximum allowable dimension. It is assumed for clarity that all possible lengths and widths are allowed in the algorithmic description below, whereas in reality only a small set of quantized lengths and widths are permissible. The program is easily modified by substituting the quantized dimensions for each L and W below.

MAXIMUM_AREA:

```

LMAX = N
For W=1 to N
  While (!White(L,W) && L>0)
    LMAX = L
    If (L*W > MAXAREA)
      MAXAREA = L*W
      BESTL = L
      BESTW = W
    Endif
    L = L - 1
  Next W

```

Let $R(L,W)$ denote the rectangle determined by the current starting point with dimensions L by W , and N denote the distance to the right hand boundary of the image. The function $White(L,W)$ returns a true value if $R(L,W)$ is completely white and does not intersect any previously chosen rectangles. This, however, can have two distinct interpretations. The function $White(L,W)$ can be efficiently computed by noting that if W increases by one then each rectangle $R(l,W+1)$ intersects the previously transmitted rectangle $R(L,W)$ as

$$R(l,W+1) \cap R(L,W) = R(\min(l,n),W) .$$

The intersection is known to be all white so that it need not be examined again. Only the pixels outside of $R(l,W)$ need to be checked. We omit the details of this procedure as they are rather straightforward.

One problem that arises with the described sequential tiling technique is that the constrained nature of the tiling order limits the variety of possibilities for laying down the rectangles. It might thus produce significantly suboptimal partitions of the image and often can miss opportunities to cover large areas of white space with rectangles. In fact, it has been observed experimentally, that frequently long, thin, vertical or horizontal "strips" are chosen for many of the tiling rectangles by the greedy selection process, whereas this seems to harm the overall partitioning process as it prevents future rectangles with very large areas.

To alleviate this problem a more suitable, technique was investigated, based on the idea of growing rectangles at arbitrary positions inside the image until the full image is partitioned.

3.4. Block Growing

In this technique an untiled white point in the image is chosen uniformly at random. The chosen pixel is designated the *birth location* and around it is grown a white rectangle of maximum possible area that is disjoint from previously grown rectangles. An exhaustive approach checks all possible white rectangles containing the birth location to find the one of maximum area. A more efficient, but suboptimal, approach grows the rectangle one pixel's width at a time in each direction until it bumps into a nonwhite pixel (or a non-white rectangle). That is, the "walls" of the growing rectangle spread out in all directions as far as possible. In practice, a good technique for doing this was observed to be cyclically growing the north, east, south, and then west walls in that order. Whenever a wall bumps into something non-white, its growth is halted and the remaining walls keep expanding until all growth is terminated. The resulting grown rectangle is fixed as part of the partition and the process repeats with a new birth location. The same growing technique can be used for determining the non-white rectangles as well. The entire partitioning process terminates when the image is completely covered with rectangles; that is when a complete partition is formed.

One of the main advantages of block growing over sequential tiling is that much more flexibility exists in terms of where to lay down rectangles with large areas. In fact, one ad hoc method that was found to be quite useful was to set a certain area threshold for a block to be grown. If in the growing process that threshold is not attained then the birth location is not accepted and a new one is chosen instead. The threshold is gradually reduced as the tiling progresses so that eventually all pixels can be tiled.

Several different techniques for choosing birth locations were investigated. The sequential tiling method previously described corresponds to choosing the birth locations as the first white uncovered pixel as seen scanning from top to bottom and left to right. A more effective technique is to choose birth locations randomly among the as yet uncovered white pixels. Choosing random birth locations together with using the thresholding criterion essentially amounted to performing several *passes* through the image, each time tiling it with smaller and smaller rectangles. The largest possible rectangles are laid down during the early passes and during the later passes only small rectangles were able to fit in the tiling. Sometimes the rectangles were observed to cover quite smaller areas if the black pixels followed curvilinear contours. As the curvature of the contour increases, the rectangles became increasingly thinner and thus less effective, something that could be handled by carefully controlling the threshold values. An example of block growing with random birth locations and thresholding is shown in Figure 2.

3.5. Covering of Black Pixels: Non-white Regions

The pixels covered by the non-white rectangles are encoded using some entropy coding technique such as arithmetic, run-length, or Huffman coding. This compression might be done on one-dimensional scans or on two-dimensional blocks. High compression ratios can be achieved when an image is tiled with a small number of rectangles each with large areas.

A trade-off arises between the number of bits used to describe the allowable quantization levels for the rectangles' lengths and widths, and the number of rectangles used to tile the image. As the number of quantization levels increases, the rectangles can be packed together more efficiently and generally less rectangles are needed to tile the image. The price that must be paid for this reduction in rectangle count is that more bits need to be transmitted per rectangle. This implies that some compromise must be achieved through experimentation. Also if very few quantization levels are used for the dimensions of the non-white rectangles then often much white space can be covered inside the non-white regions, implying poorer compression ratios from the entropy coding stage.

3.6. Tiling of White Space

Since block growing with several passes can overcome the limitations of sequential tiling, it was chosen that the white space first be tiled by growing rectangles. The number of passes and the area threshold were chosen according to their observed performances. The efficiency of tiling generally improved as the number of passes increases; however, the passes can add to the delay. For instance, in block growing with 7 passes on a variety of business letters, some of the grown rectangles typically cover as many as 700,000 pixels each. On the average, block growing yielded

a compression ratio of around of 1500 within the grown regions, when the minimum threshold area allowed is 100 for these images.

4. Results and Discussion

The proposed technique was simulated on a SUN SPARC 2 workstation to demonstrate its applicability and to evaluate its performance. Figure 3 shows a portion of the original of one of the images tested. It consists of a typed business letter plus a handwritten signature and is fairly typical of a large percentage of facsimile transmissions. In Figure 4, the same image is shown after prediction and after an unconstrained block tiling using random birth locations and 7 area threshold levels. The prediction process can be seen to have removed many of the black pixels from the original image, and the tiling has removed much of the white space.

For this full image, only 9.22% of the original images are covered by non-white pixels after tiling. In other images encoded, it was typical for about 5-20% of the pixels to be covered by non-white rectangles. In almost all business type letters encoded, about 94-99% of all transmitted bits came from the entropy coding within the non-white rectangles. This indicates that the elaborate white rectangle tiling procedure does not increase the overall bit rate by any significant amount. It does however, allow the entropy coder to concentrate on a small portion of the image.

The final compression ratios of the proposed scheme varied widely depending upon which image was coded. We compared the proposed scheme with that of several other techniques. The compression ratios achieved for these on the image shown in Figure 3 were 9 for ordinary one-dimensional entropy coding, (e.g. arithmetic coding), 22 for prediction followed by entropy coding, and 51 for a more elaborate one-dimensional run-length coding scheme. Our unconstrained tiling scheme yielded a ratio of 80. Similar relative compression ratios were observed for many other different source images.

5. Conclusion

A simple-to-implement technique for the compression of high-resolution bi-level images for facsimile transmission and storage was presented. The proposed technique divides the problem of coding for compression into two parts: tiling of the image with white and non-white rectangles, and coding of pixels covered by non-white rectangles. The idea of partitioning the image into rectangles of predefined dimensions extends the notion of run-length coding to two dimensions.

Two new algorithms for tiling the white space, namely sequential tiling and block growing were presented. In block growing, an omni-dimensional growing method was found to be an efficient way of tiling the white space. It was shown that block tiling within the tiled regions yields very high compression ratios.

A new technique which combines the concepts of block and run-length coding was also introduced. It was found that of all the simulated coding techniques, run-length coding and run-length block coding gave the best compression ratios.

It can be concluded that the concepts of tiling of the white space with rectangles and that of WRRC yield very high compression ratios. Although the covering of black pixels with non-white rectangles takes up a negligible part of the total number of transmitted bits, it results in a higher compression, and reduces the problem of compression to that of coding of pixels in well defined regions which cover a relatively very small area of a typical facsimile document.

Although the primary aim of this thesis was compression of bi-level images, it should, however, be emphasized that the proposed technique can effectively be used for multi-level image by simply encoding each bit plane independently as though it were itself a bi-level image.

Acknowledgements

The research was supported in part, by Hewlett-Packard Co., and the National Science Foundation under Grants No. NCR-90-09766 and NCR-91-57770.

REFERENCES

- [1] I.H. Witten, R.M. Neal, and J.G. Cleary, "Arithmetic coding for data compression", *Communications of the ACM*, vol. 30, no. 6, pp. 520-540, June 1987.

- [2] V.R. Algazi, P.L. Kelly, and R.R. Estes, "Compression of binary facsimile images by preprocessing and color shrinking" ,," *IEEE Trans. Commun.*, vol. COM-38, no. 9, pp. 1592-1598, Sept. 1990.
- [3] Y.Yasuda, "Overview of digital facsimile coding techniques in Japan" ,," *Proc. IEEE*, vol. 68, no. 7, pp. 830-845, July 1980.
- [4] H.G. Musmann and D. Preuss, "Comparison of redundancy reducing codes for facsimile transmission of documents" ,," *IEEE Trans. Commun.*, vol. COM-25, no. 11, pp. 1425-1433, Nov. 1977.
- [5] D.C.V. Voorhis, "An extended run-length encoder and decoder for compression of black/white images" ,," *IEEE Trans. Inform. Theory*, vol. IT-22, no. 2, pp. 190-199, Mar. 1976.
- [6] F.W. Mounts, E.G. Bowen, and A.N. Netravali, "An ordering scheme for facsimile coding" ,," *Bell Syst. Tech. J.*, vol. 58, no. 9, pp. 2113-2128, Nov. 1979.
- [7] H. Meyr, H.B. Rodolsky, and T.S. Huang, "Optimum run-length codes" ,," *IEEE Trans. Commun.*, vol. COM-22, no. 6, pp. 826-835, June 1974.
- [8] M. Takagi and T. Tsuda, "Bandwidth compression for facsimile using two-dimensional prediction" ,," *Syst., Comput., Contr.*, vol. 4, no. 2, pp. 16-23, 1973.
- [9] "Proposal for draft recommendation of two-dimensional coding scheme" ,," *CCITT SG XIV Contribution*, no. 42, Nov. 1978.
- [10] D.A. Huffman, "A method for the construction of minimum redundancy codes" ,," *Proc. IRE*, vol. 40, pp. 1098-1101, Sept. 1952.
- [11] A.K. Jain, "Fundamentals of digital image processing," *Prentice Hall*, New Jersey, 1989.
- [12] H. Wyle, T. Erb, and R. Barow, "Reduced time facsimile transmission by digital coding" ,," *IRE Trans. on Commun. Syst.*, vol. CS-9, no. 3, pp. 215-222, Sept. 1961.
- [13] Y. Wakahara, Y. Yamazaki, H. Teramura, and Y. Nakagome, "Data compression factors of relative address coding scheme for facsimile signals",," *J. IEE Jap.*, vol. 5, no. 3, pp. 92-100, Oct. 1976.
- [14] T. Yamada, "Edge-difference coding -- A new, efficient redundancy reduction technique for facsimile signals" ,," *IEEE Trans. Commun.*, vol. COM-27, no. 8, pp. 1210-1217, Aug. 1979.
- [15] H. Morita and S. Arimoto, "SECT -- A coding technique for black/white graphics" ,," *IEEE Trans. Inform. Theory*, vol. IT-29, no. 4, pp. 559-570, July 1983.
- [16] M. Kunt, "Comparaison de techniques d'encodage pour la reduction de redondance d'images facsimile a deux niveaux" ,," *Thesis*, no. 183, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland, , 1974..
- [17] O. Johnsen, "Etude de strategies adaptatives pour la transmission d'images facsimile a deux niveaux" ,," *AGEN Mitteil.*, no. 20, pp. 41-53, June 1976.
- [18] Y. Cohen, M.S. Landy, and M. Pavel, "Hierarchical coding of binary images" ,," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-7, no. 3, pp. 284-298, May 1985.
- [19] K. Knowlton, "Progressive transmission of grey scale and B/W pictures by simple, efficient and lossless encoding schemes" ,," *Proc. IEEE*, vol. 68, pp. 885-896, 1980.
- [20] M. Gardner, "Mathematical games" ,," *Sci. Amer.*, pp. 124-133, Dec. 1976 .
- [21] A. Lempel and J. Ziv, "Compression of two-dimensional data" ,," *IEEE Trans. Info. Theory*, vol. IT-32, no. 1, pp. 2-8, Jan. 1986 .
- [22] A. Gersho, "Optimal Nonlinear Interpolative Vector Quantization" ,," *IEEE Trans. Commun.*, vol. COM-38, no. 9, pp. 1285-1287, September 1990.

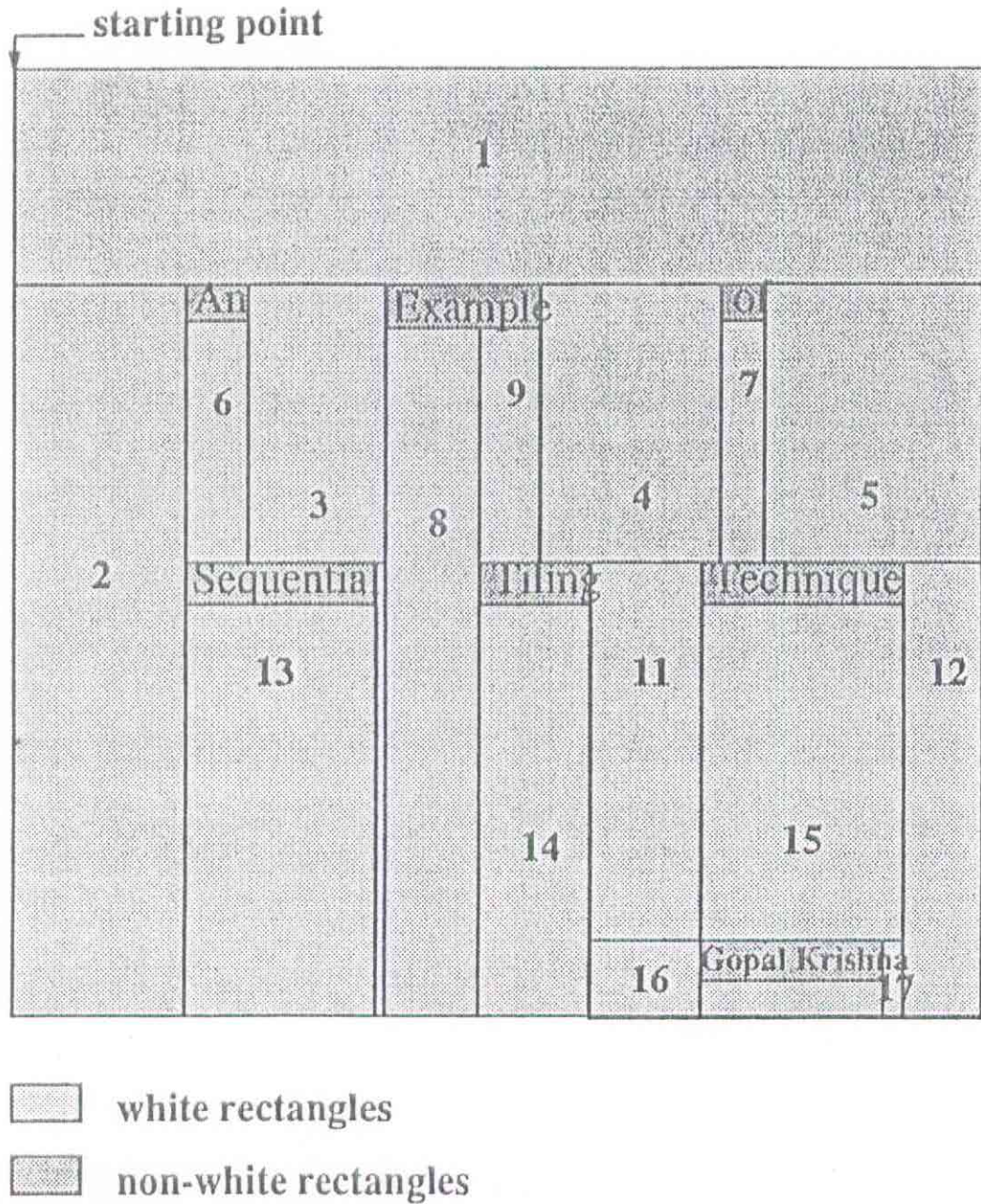
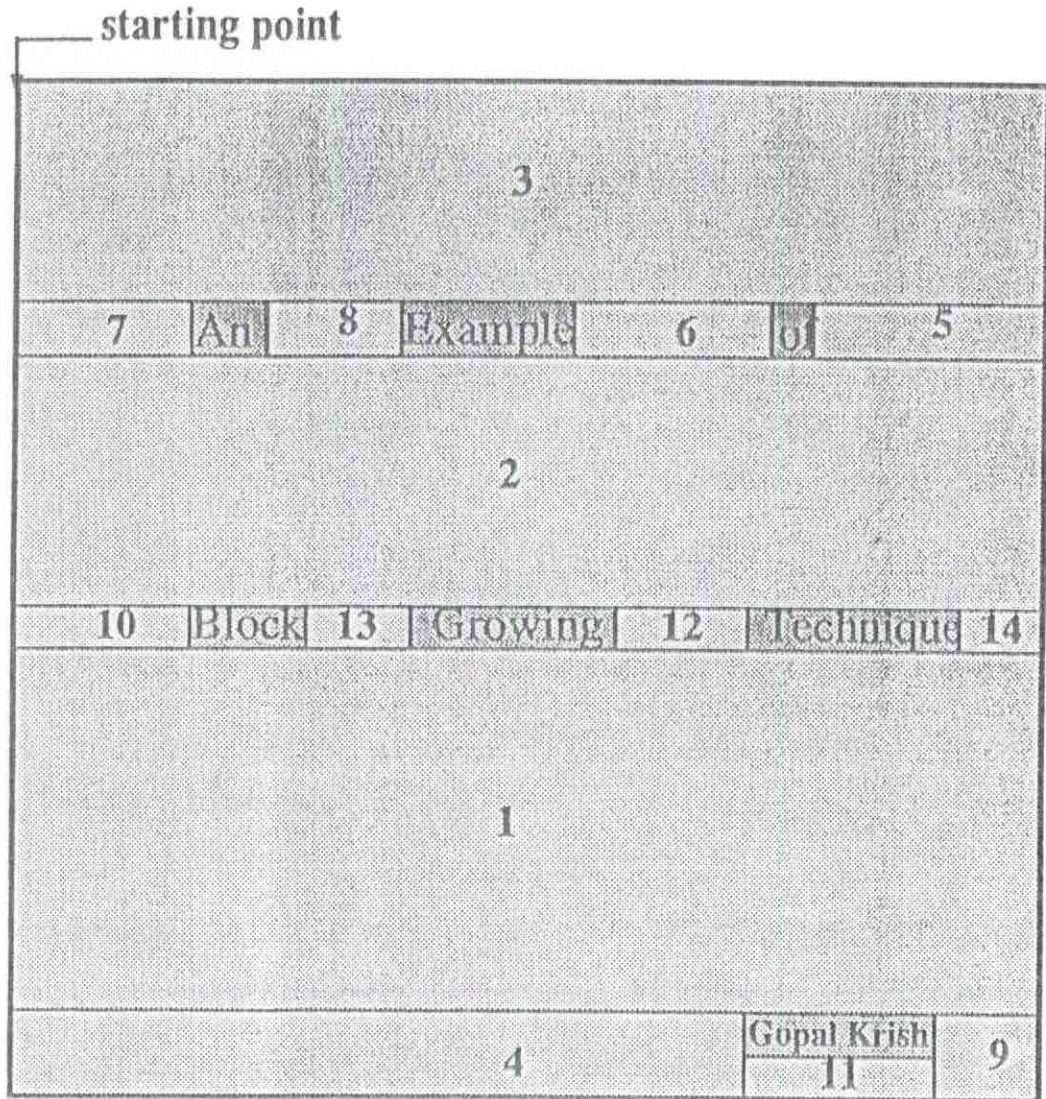


Figure 1: An example of Sequential Tiling of an Image. The rectangles are numbered in the order in which they were constructed, from a left to right, top to bottom scan.





-  grown rectangles
-  non-white rectangles

Figure 2: An example of Block Growing with random birth locations. Seven different threshold levels were used in order to encourage large blocks to be constructed first. The rectangles are numbered in the order in which they were constructed.

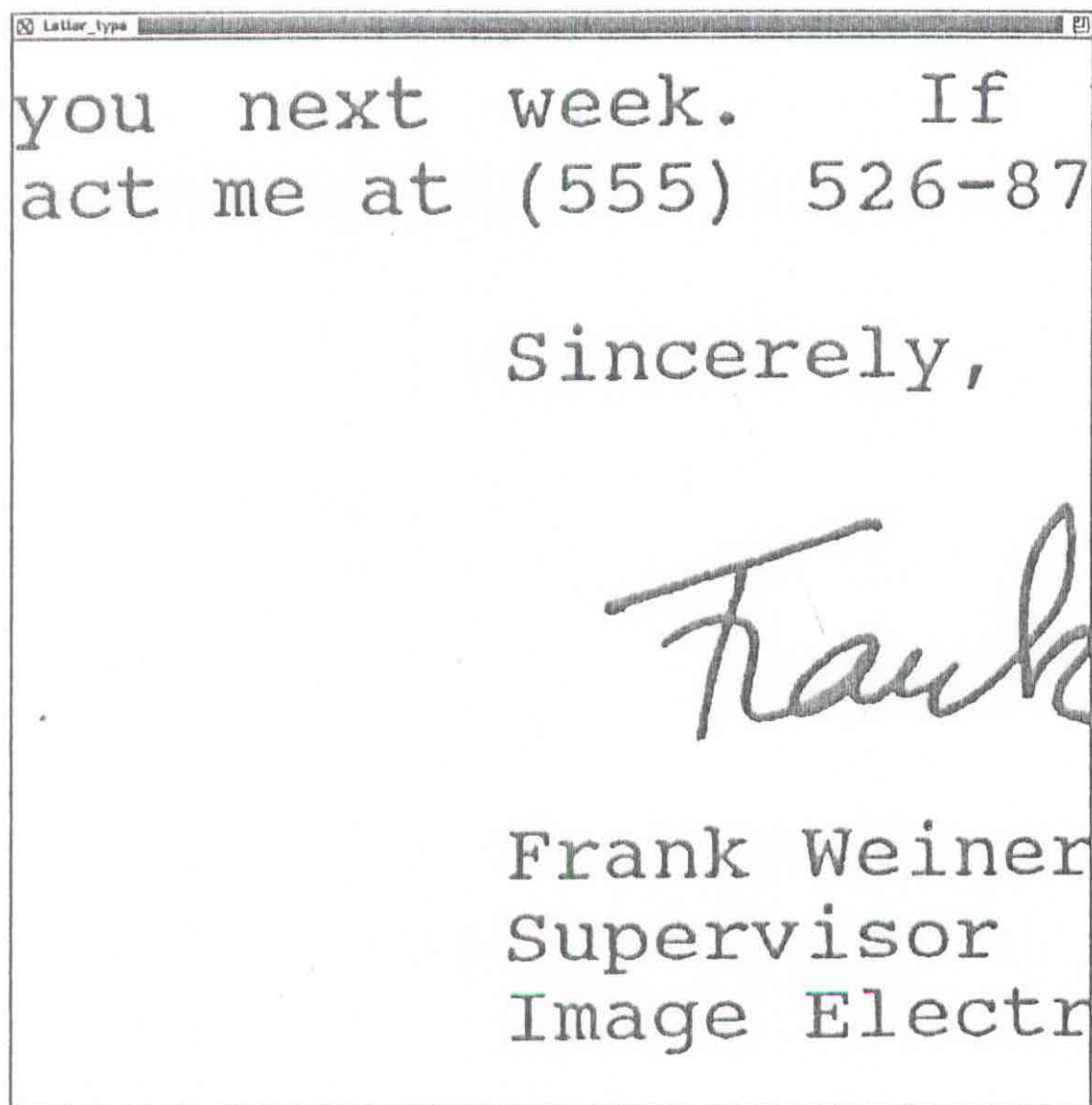


Figure 3: A portion of an original business letter facsimile image coded.

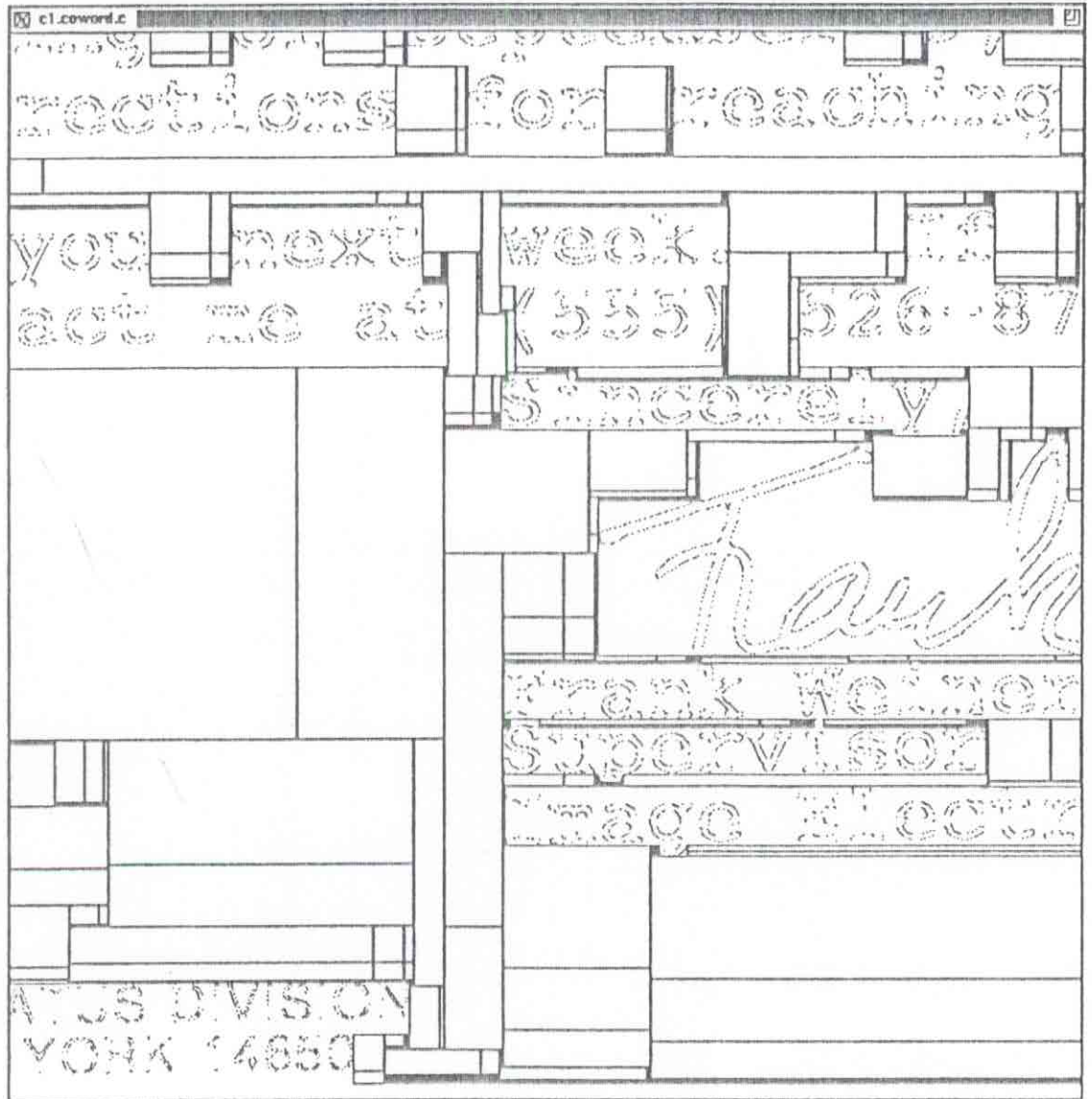


Figure 4: Tiling of the residual image by Block Growing with random birth locations and quantized rectangle lengths and widths.