Presented at SPIE Conf. Image Algebra and Morphological Image Processing II Conference 1568, pp. 38-52, July 23-24, 1991, San Diego, CA.

IMAGE ANALYSIS USING THRESHOLD REDUCTION

Dan S. Bloomberg

Xerox Corporation
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304

Abstract

A class of shift-variant reduction operations is introduced, that is useful for performing efficient and controllable shape and texture transformations between resolution levels. In their most general form, the operations proceed in three steps: (a) convolve a binary image with a kernel of arbitrary size; (b) threshold the result; (c) subsample to produce the reduced image. Taking a binary structuring element for the kernel, the threshold convolution on a binary image is equivalent to a rank order filter, and the full reduction operation is a *threshold reduction*. Threshold reductions that use convolution filters and subsample tiles of equal size are optimized by combining the three operations, using only logical raster operations and producing threshold convolution values only at the sampling points. For 2x reduction, the four possible threshold values (1, 2, 3, and 4) refer to the minimum number of ON pixels within each 2x2 tile for which a pixel in the reduced image will be ON. Algorithms for boolean raster operations are given for 2x, 3x, and 4x threshold reduction, and lookup tables that efficiently implement column raster operations are provided. Threshold reduction rates of 2.5x10^7 pixel/second can be achieved with a Sun SparcStation2^{TM}. A mask-forming image analysis cycle of threshold reduction, augmented by morphology and followed by replicative expansion to full resolution, is described, and some general properties of the cycle are derived. A simple application of threshold reduction to document image analysis, the extraction of halftone regions from scanned images that also contain text and line graphics, is illustrated. A sequence of 2x reductions with first low and then high thresholds is used to create a reduced image consisting of a mask over the halftone regions. In this way, the extraction occurs as a natural consequence of the reductions.

Keywords: image processing, image analysis, morphology, multiresolution, threshold reduction, threshold convolution, segmentation, document, texture

1 Introduction

The intent of this paper is to describe an effective and computationally efficient multiresolution technique for the analysis of shape and texture properties of large binary document images. Morphological image processing can be used to extract both shape and textural information from images. Without special hardware, however, use of morphological image processing at high resolution has a large computational cost, particularly in time, and especially when large-scale features are to be identified. The amount of computation varies approximately inversely as the third power of the reduction factor. Two powers are due to the relative number of pixels, and the third power comes from the size of the structuring elements or the number of iterations required to cover a given feature. Thus, efficiency dictates that images be analyzed at the minimum resolution required for characterizing the requisite shape and texture structure.

Multiresolution methods for image analysis require construction of representations of the original image at many scales. The simplest method for constructing a multiresolution pyramid is successively to subsample the image, at each reduction step taking only one pixel from a r x r tile of pixels in the original. This typically (although not always) preserves the average density of the original image. Haralick et al.[6, 7] have emphasized the importance of using a low-pass filter prior to subsampling, to prevent aliasing of high frequency components. They investigated a morphological analog of the sampling theorem, with a view toward a best effort for reconstructing a high resolution filtered image from a subsampled version. Burt[4] has shown that multiresolution methods with small filters are capable of accurately characterizing texture at multiple scales. Preservation of such image qualities may be useful for measurement, compression, reconstruction, and rendering, where fidelity to the original is paramount.

However, we are not explicitly concerned with either the spatial or the detailed textural fidelity of the subsampled image. Instead, for most purposes of image analysis, we search for methods that give maximum discrimination between regions with differing shape and texture properties. The methods given here are *image-based*, in that discrimination takes place almost entirely in the image domain. Image texture, which is a set of statistical properties of relations between ON and OFF pixels within a region whose size is much larger than the measures used for gathering the statistics, plays a central role. Separation of regions with different texture is accomplished by operations that either differentially transform texture with scale change, or differentially project texture components at constant scale. Ideally, a sequence of operations is obtained that projects out or labels all pixels in any chosen region, and does not mis-classify pixels in other regions.

Our multiresolution pyramid building operations use a *rank order filter* before subsampling. This filter is in fact a *threshold convolution*, and for extreme values of the threshold parameter it is equivalent to morphological dilation and erosion. We call the multiresolution operations *threshold reduction*. Similar approaches have been taken. Tanimoto[10, 11] described construction of binary image pyramids where a pixel at a given level is computed from a threshold of the sum of ON pixels of its children. Bones et al.[3] recently presented an approach for segmenting document images based on morphological image processing and image analysis at various levels of reduction.

The plan of the paper is as follows. We first describe the elements of multiresolution morphology, and introduce the threshold reduction operation. The optimization of the algorithm for general purpose computers is described, and the effect of threshold reductions on texture is explained. Some properties of the mask-forming, pixel-labelling cycle of reduction/expansion are given. Then the method is illustrated by the problem of separating halftone image regions from text and line art. We conclude with a short discussion of the approach. Two appendices provide details on the efficient implementation of threshold reduction by logical raster operations and lookup tables. All algorithms have been implemented in C. CPU timings are given for the Sun Sparcstation2^{TM}.

2 Multiresolution morphology

The filtering operations used in threshold reduction are rank order filters, which are a generalization of the morphological operations erosion and dilation[9, 5]. We first describe these filters, and then show how they are used to implement threshold reduction.

2.1 Morphology and threshold convolution

The fundamental morphological operations, erosion and dilation, are most efficiently implemented by translating the image and either ANDing or ORing it with itself. Specifically, letting X represent the binary image and the (usually) small set A represent the *structuring element* (SE), the *erosion* $\ominus$ and *dilation* $\oplus$ of X by A are defined as

$$X \ominus A = \bigcap_{z \in A} X_{-z} \tag{1}$$

$$X \oplus A = \bigcup_{z \in A} X_{z} \tag{2}$$

where X_z is the *translation* of X along the pixel vector z, and the set intersection and union operations represent bitwise AND and OR, respectively. These operations can be implemented as raster operations to take advantage of the word-parallel representation of the pixels within a computer.

Other morphological operations can be built from these two. Of most importance in image analysis are the *opening*, *closing*, *hit-miss transform*, and *generalized opening*[2]. The latter two operations allow explicit pattern matching to background pixels as well as foreground pixels, and all but the hit-miss transform are idempotent and center-independent.

These morphological operations all require exact matches to the SEs. An imperfect match, called a *rank order filter* or, equivalently, a *threshold convolution*, is a generalization of the erosion and dilation operations of morphology. The m-th rank order transformation of a binary image X by a SE A is the set of pixel positions to which the translated SE covers at least m pixels in the image:

$$X \,\square_m\, A = \{z : |X \cap A_z| \geq m\} \tag{3}$$

If the threshold $m = 1$, $X \square_m A$ is the dilation $X \oplus \check{A}$.[1] Let A be an r x r square SE of "hits". Then at the other extreme, where the threshold m equals the cardinality r^2 of A, $X \square_m A$ becomes the erosion $X \ominus A$. For the relationship between rank order and morphological operations see [8].

[1] $\check{A}$ is the spatial inversion of A about its center. Note the location of the center points for the filters in Figure 1.

2.2 Threshold reduction

To motivate the use of threshold reduction, consider the problem of segmenting a scanned image into text and halftone image regions. We look more closely at this problem in Section 4. A brute-force morphological approach might be to close the image with sufficiently large SEs to solidify the halftone parts, and then open the image with even larger SEs to remove the (somewhat blocked up but smaller) text parts. The opening would not affect the solid halftone regions, and the result would be a separation mask covering only the halftone areas. The closing removes OFF pixels that are "near" ON pixels, and the opening removes ON pixels that are "near" OFF pixels, with the scale of "near" given by the size of the respective SEs. Both operations can be viewed as alteration of short-range image texture.

This suggests that filtering operations before subsampling should be chosen to change the image texture so as to mimic operations at full scale. To solidify pixels within halftone regions, use a closing or dilation operation before subsampling; then at reduced scale, use an opening or erosion before further subsampling. Because it is expensive to use large SEs at high resolution, we can effect an arbitrary and efficient 2^n reduction by a cascade of n 2-fold reductions, pre-filtering with 2x2 SEs at each step.

Thus we tile the image into 2x2 squares and subsample the upper-left pixel of each tile. Consider the 2x2 SE whose reference position is located in the lower-right corner (Figure 1a). Dilation with this SE prior to subsampling is equivalent to setting a threshold of 1 ON pixel in the 2x2 pixel tile: if at least 1 pixel is ON, after dilation the pixel to be subsampled will surely be ON. Likewise, use of an erosion by the SE shown in Figure 1b prior to subsampling is equivalent to setting a threshold of 4 ON pixels in the tile: all four pixels in the tile must be ON if the subsampled pixel is to be ON. Clearly, flexibility is gained if we generalize to allow filters that threshold on 2 and 3 ON pixels within the tile. The requisite filtering operations are threshold convolution (or, equivalently, rank order filters) mentioned previously.

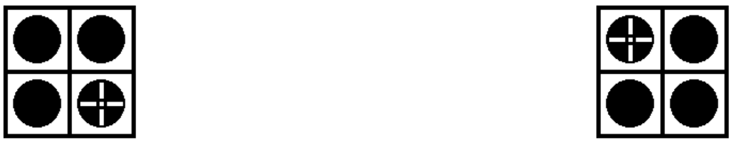

(a) (b)

Figure 1. (a) Dilation filter for threshold 1; (b) erosion filter for threshold 4

We call the combination of a threshold convolution followed by subsampling a *threshold reduction*. The halftone segmentation problem is efficiently addressed by a sequence of 2x reductions using a small threshold, say 1, to consolidate the halftone textured region, followed by further 2x reductions with a large threshold, say 4, to remove the text regions. Larger atomic reductions with more threshold levels for pre-filtering can also be used, but in practice we have found that the four different 2x threshold reductions provide sufficient flexibility.

2.3 Optimization of the threshold reduction algorithm

It is only necessary to apply threshold convolution to those pixels that will be subsampled. Further, it is desirable to use logical operations instead of arithmetic, in order to take advantage of word parallel instructions in the computer. Threshold reduction is efficiently implemented by first forming a half-height, full-width image using a logical operation (OR or AND) between each odd row and the even row below it. This is followed by reduction to a half-height, half-width image using a logical operation (OR or AND) between each odd column and the even column following it. If both row and column operations are OR, each ON pixel in the reduced image is ON if any of the four pixels in the corresponding tile of the original image were ON. This is a threshold 1 reduction. Likewise, if both operations are AND, we get a threshold 4 reduction.

It takes approximately twice as much work to reduce images with threshold values of 2 and 3. To do this, form two intermediate half-height, half-width images, using OR-AND and AND-OR for the row and column operations. The threshold 2 and threshold 3 reduced images are then found by taking the *union* and *intersection* of these intermediate reduced images, respectively. The operations are summarized in Table 1. Along with generalization to 3x and 4x reductions, they are derived in Appendix I. Appendix I also describes symmetry properties of the threshold reduction operators.

Threshold	Row/Column Operations
1	OR/OR
2	(OR/AND) $\cup$ (AND/OR)
3	(OR/AND) $\cap$ (AND/OR)
4	AND/AND

Table 1. Implementation of 2x threshold reduction with boolean operations.

The final optimization is to replace the the column-wise logical operations, which are very slow on a computer that stores the pixels sequentially in raster order, by a set of lookup tables that emulate these column operations. Algorithms for constructing various sized OR and AND lookup tables are given in Appendix II.

With these optimizations, a 2x threshold reduction using either threshold $m = 1$ or $m = 4$ is not much slower than simple subsampling, and proceeds at about 25 Mbit/sec. Reductions using intermediate threshold values $m = 2$ and $m = 3$ operate at about half this speed.

2.4 Rules of thumb for threshold reduction

The effect on texture from a sequence of threshold reductions is fairly predictable. For example, a set of four sequential threshold 1 reductions is approximately equal to a dilation with a 16x16 brick SE, followed by subsampling. Pairs of ON pixels separated by less than about 16 pixels will typically be joined. Likewise, four sequential threshold 4 reductions are roughly equivalent to an erosion with a 16x16 brick SE, followed by subsampling. Regions of ON pixels smaller than such a brick will typically vanish in the reduced image. Just as dilation and erosion expand and shrink regions of ON pixels, threshold reduction using thresholds of 1 and 4 tend to expand and shrink solid regions, respectively, and some compensation may be required. The subsampling operation is not translationally invariant; consequently, some variation is to be expected due to the positioning of the 2x2 tiles on the image.

3 Properties of Threshold Reduction/Expansion Cycles

Here, we consider set properties of threshold reduction, cascades of such reductions, cascades augmented by shift-invariant morphological operations, and *cycles* composed of threshold reduction cascades followed by *replicative expansion* to the initial resolution.

For r-fold reduction, at each reduction stage an r x r tile is reduced to one pixel. For a cascade of k such reductions, a *tile set* is the set of r^k x r^k pixels that are reduced to a single pixel. In the replicative expansion step, that pixel is then expanded back to a r^k x r^k tile set, with each pixel assigned the same value as the single pixel from which it was replicated. The result of a reduction/expansion cycle is the creation of a new binary image at full resolution, that can be used as an extraction mask. The value of each pixel in this image can be viewed as a label, on the corresponding pixel in the original image, that describes some set of neighborhood properties of the original pixel.

PROPERTY 1 . *A cycle composed of k threshold reductions, each with threshold $m = 1$, and followed by replicative expansion, is extensive.*

Proof. If any pixel is initially ON in an r x r tile, then after a reduction/expansion cycle with $k = 1$ reduction, all pixels in that tile will be ON. For k reductions, the result applies to each tile set of r^k x r^k pixels in the image. □

PROPERTY 2 . *Threshold reduction followed by replicative expansion to the original resolution is idempotent and increasing. For minimum threshold it is extensive, for maximum threshold it is anti-extensive, and for intermediate thresholds it is neither extensive nor anti-extensive.*

Proof. After a reduction/expansion cycle, all pixels in a tile are either ON or OFF. Any threshold reduction of a tile with all pixels ON results in an ON pixel; likewise, any threshold reduction of a tile with all pixels OFF results in an OFF pixel. Thus, the second cycle does not change the image, and the cycle is idempotent. The cycle is increasing because increasing the number of ON pixels in a tile can never cause the thresholded tile to change from ON to OFF, and v.v. For minimum threshold, if any pixel is originally ON in a tile, after the cycle all pixels will be ON, so the cycle is extensive. For maximum threshold, if any pixel is originally OFF, after the cycle all pixels will be OFF, so the cycle is anti-extensive. For an intermediate threshold of m, where $1 < m < n = r^2$, all pixels in the tile will be turned ON if at least m pixels are initially ON. If less than m pixels are initially ON, they will be turned OFF, so the cycle is not extensive. Likewise if not more than $n - m$ pixels are OFF, all such pixels will be turned ON, so the cycle is not antiextensive. □

PROPERTY 3 . *A cascade of threshold reductions followed by replicative expansion to the original resolution is idempotent and increasing.*

Proof. Suppose there are k threshold reductions in the cascade. Expansion replicates each reduced pixel to a r^k x r^k set of pixels. Any cascade of k threshold reductions on such a set produces a single pixel with the same polarity as the set; hence, the cycle is idempotent. The cycle is increasing because turning any pixel from OFF to ON in a r^k x r^k set can never cause the pixel produced by a cascade of k threshold reductions to change from ON to OFF, and v.v. □

Consider a cascade of threshold reductions, augmented at each stage of reduction by a set of morphological operations, and ending with replicative expansion to the original resolution. Under what conditions is the result idempotent and increasing? First, if center-dependent operations such as dilations and erosions are permitted, then the cycle will not in general be idempotent, because an arbitrary shift of the image is possible. Second, if non-increasing operations like the hit-miss transform and the generalized opening are permitted, then the cycle can not in general be increasing. Third, if non-increasing operations are permitted, then the cycle will not generally be idempotent because the replicative expansion step results in solid regions that may give no match to a generalized opening. This leaves the following property of an augmented cascade:

PROPERTY 4 . *A cascade of threshold reductions, augmented at each stage of reduction by arbitrary increasing, anti-extensive, idempotent morphological operations (such as opening), and followed by replicative expansion to the initial resolution, is idempotent and increasing.*

Proof. The increasing property of the augmented cascade follows because each atomic operation (threshold reduction or opening) is increasing. Because tile sets do not interact under threshold reduction, and because the opening is anti-extensive, any tile set that begins the second cycle with only OFF pixels will remain in that condition. Further, because (a) both threshold reduction and opening are increasing and (b) at the beginning of the second cycle a tile set is composed of uniform pixel values (ON or OFF), any pixel set that reduced to an ON pixel in the first cycle will surely reduce to the same value in the second cycle, and the augmented cascade is idempotent. □

This result does not apply when *closings* augment the threshold reductions. Figure 2 shows a simple example containing four 2x2 tiles, where the result of the second cycle is different from that of the first. The cycle consists of a closing with a diagonal SE of length 3, a 2x reduction with a threshold of 1, and a 2x2 expansion to original scale. In the first cycle the closing has no effect, but in the second cycle, the closing turns one pixel ON in each of the empty tiles.

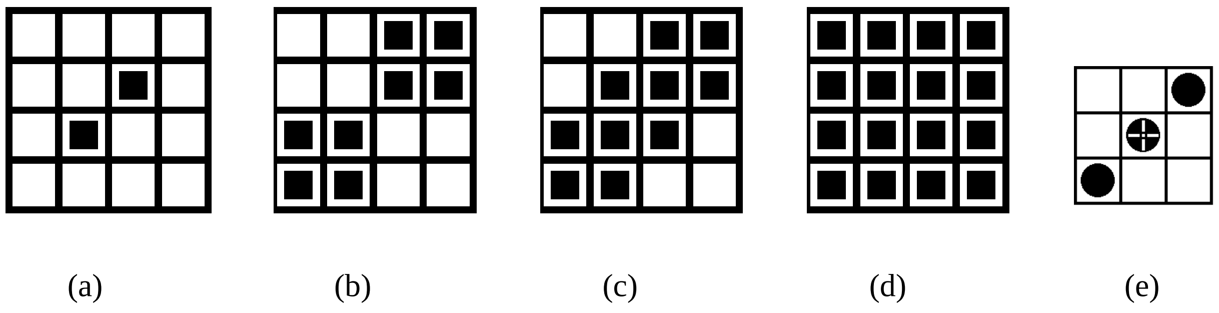


Figure 2. (a) initial image, (b) image after first cycle,
(c) image in second cycle after closing with 3x3 SE in (e),
(d) image after second cycle.

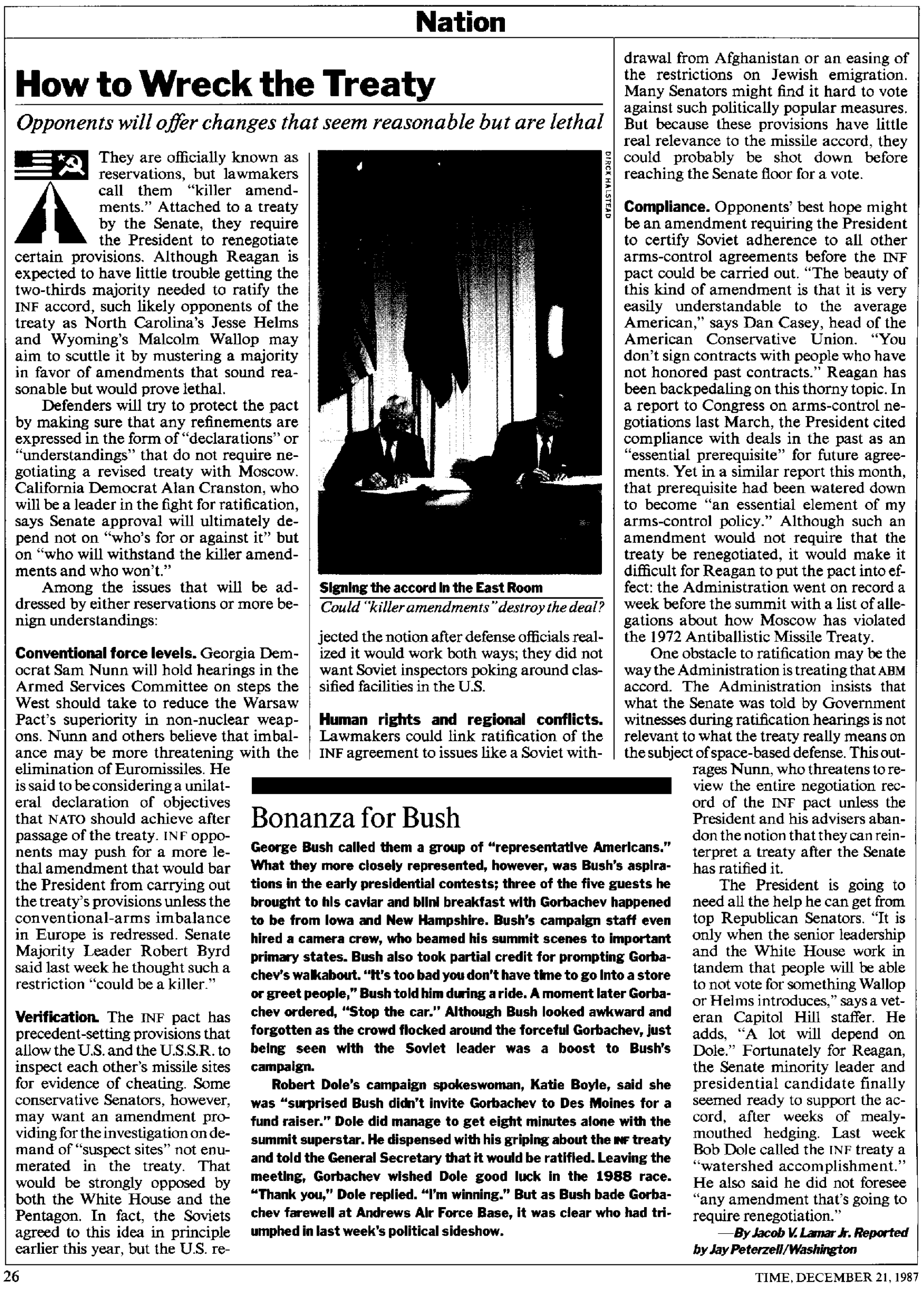
4 Example: segmentation of halftone image areas

Image regions in scanned documents have a variety of short-range textural properties, due to both the method of construction and the scanning and thresholding conditions. Stippled regions are regular and periodic; for all pixels, the minimum distance to the closest ON pixel is sharply bounded. Halftone regions constructed with error diffusion algorithms can have an anisotropic distribution of this measure, and an upper bound for the distance is not guaranteed. The scanner thresholding can cause isolated foreground or background pixels to disappear (i.e., regions of light halftoning become lighter, and dark regions solidify). As a result, the textural statistics of such image regions is highly variable.

The multiresolution method described here correctly labels all pixels that do not belong to image areas. When applied to regular stipple patterns and large dark regions, it also correctly labels all pixels within such image regions. However, it may miss some pixels that are in very light and extensive image regions. Thus, depending on the set of morphological operations that augment the threshold reduction cycle, the segmentation mask may not cover all pixels in some image areas.

The resolution of images displayed here is given in units that are independent of the resolution of the physical rendering device. All images are labelled with both the *sampling resolution*, in pixels/inch, and the *rendering resolution*, also in pixels/inch. The sampling resolution gives the size of the sampled (or subsampled) pixels in the image relative to the original, whereas the rendering resolution gives the size of these sampled pixels as rendered on the page. The *magnification* is the ratio of sampling to rendering resolution.

As an example of a typical problem, Figure 3 shows a scanned image where the contrast in the halftone region has been increased by the scanner: the dark parts have solidified and the light parts have opened up. Figure 4a shows the image after two 2x $m = 1$ threshold reductions, and Figure 4b shows the result of a subsequent closing by a 3x3 brick SE. The halftone region is not entirely solidified, and the text is somewhat blocked up. Figure 4c shows the result of two more 2x threshold reductions, this time at the other extreme with $m = 4$. Remaining pixels in the text/line regions are much less dense, and can be eliminated entirely with an opening by the 3x3 brick SE, as shown in Figure 4d. The entire process on this 8 million pixel image takes about 0.5 sec.



Nation

How to Wreck the Treaty

Opponents will offer changes that seem reasonable but are lethal

They are officially known as reservations, but lawmakers call them "killer amendments." Attached to a treaty by the Senate, they require the President to renegotiate certain provisions. Although Reagan is expected to have little trouble getting the two-thirds majority needed to ratify the INF accord, such likely opponents of the treaty as North Carolina's Jesse Helms and Wyoming's Malcolm Wallop may aim to scuttle it by mustering a majority in favor of amendments that sound reasonable but would prove lethal.

Defenders will try to protect the pact by making sure that any refinements are expressed in the form of "declarations" or "understandings" that do not require negotiating a revised treaty with Moscow. California Democrat Alan Cranston, who will be a leader in the fight for ratification, says Senate approval will ultimately depend not on "who's for or against it" but on "who will withstand the killer amendments and who won't."

Among the issues that will be addressed by either reservations or more benign understandings:

Conventional force levels. Georgia Democrat Sam Nunn will hold hearings in the Armed Services Committee on steps the West should take to reduce the Warsaw Pact's superiority in non-nuclear weapons. Nunn and others believe that imbalance may be more threatening with the elimination of Euromissiles. He is said to be considering a unilateral declaration of objectives that NATO should achieve after passage of the treaty. INF opponents may push for a more lethal amendment that would bar the President from carrying out the treaty's provisions unless the conventional-arms imbalance in Europe is redressed. Senate Majority Leader Robert Byrd said last week he thought such a restriction "could be a killer."

Verification. The INF pact has precedent-setting provisions that allow the U.S. and the U.S.S.R. to inspect each other's missile sites for evidence of cheating. Some conservative Senators, however, may want an amendment providing for the investigation on demand of "suspect sites" not enumerated in the treaty. That would be strongly opposed by both the White House and the Pentagon. In fact, the Soviets agreed to this idea in principle earlier this year, but the U.S. rejected the notion after defense officials realized it would work both ways; they did not want Soviet inspectors poking around classified facilities in the U.S.

DIRCK HALSTEAD

Signing the accord in the East Room
Could "killer amendments" destroy the deal?

Human rights and regional conflicts. Lawmakers could link ratification of the INF agreement to issues like a Soviet withdrawal from Afghanistan or an easing of the restrictions on Jewish emigration. Many Senators might find it hard to vote against such politically popular measures. But because these provisions have little real relevance to the missile accord, they could probably be shot down before reaching the Senate floor for a vote.

Compliance. Opponents' best hope might be an amendment requiring the President to certify Soviet adherence to all other arms-control agreements before the INF pact could be carried out. "The beauty of this kind of amendment is that it is very easily understandable to the average American," says Dan Casey, head of the American Conservative Union. "You don't sign contracts with people who have not honored past contracts." Reagan has been backpedaling on this thorny topic. In a report to Congress on arms-control negotiations last March, the President cited compliance with deals in the past as an "essential prerequisite" for future agreements. Yet in a similar report this month, that prerequisite had been watered down to become "an essential element of my arms-control policy." Although such an amendment would not require that the treaty be renegotiated, it would make it difficult for Reagan to put the pact into effect: the Administration went on record a week before the summit with a list of allegations about how Moscow has violated the 1972 Antiballistic Missile Treaty.

One obstacle to ratification may be the way the Administration is treating that ABM accord. The Administration insists that what the Senate was told by Government witnesses during ratification hearings is not relevant to what the treaty really means on the subject of space-based defense. This outrages Nunn, who threatens to review the entire negotiation record of the INF pact unless the President and his advisers abandon the notion that they can reinterpret a treaty after the Senate has ratified it.

The President is going to need all the help he can get from top Republican Senators. "It is only when the senior leadership and the White House work in tandem that people will be able to not vote for something Wallop or Helms introduces," says a veteran Capitol Hill staffer. He adds, "A lot will depend on Dole." Fortunately for Reagan, the Senate minority leader and presidential candidate finally seemed ready to support the accord, after weeks of mealy-mouthed hedging. Last week Bob Dole called the INF treaty a "watershed accomplishment." He also said he did not foresee "any amendment that's going to require renegotiation."

—***By Jacob V. Lamar Jr. Reported by Jay Peterzell/Washington***

Bonanza for Bush

George Bush called them a group of "representative Americans." What they more closely represented, however, was Bush's aspirations in the early presidential contests; three of the five guests he brought to his caviar and blini breakfast with Gorbachev happened to be from Iowa and New Hampshire. Bush's campaign staff even hired a camera crew, who beamed his summit scenes to important primary states. Bush also took partial credit for prompting Gorbachev's walkabout. "It's too bad you don't have time to go into a store or greet people," Bush told him during a ride. A moment later Gorbachev ordered, "Stop the car." Although Bush looked awkward and forgotten as the crowd flocked around the forceful Gorbachev, just being seen with the Soviet leader was a boost to Bush's campaign.

Robert Dole's campaign spokeswoman, Katie Boyle, said she was "surprised Bush didn't invite Gorbachev to Des Moines for a fund raiser." Dole did manage to get eight minutes alone with the summit superstar. He dispensed with his griping about the INF treaty and told the General Secretary that it would be ratified. Leaving the meeting, Gorbachev wished Dole good luck in the 1988 race. "Thank you," Dole replied. "I'm winning." But as Bush bade Gorbachev farewell at Andrews Air Force Base, it was clear who had triumphed in last week's political sideshow.

26 TIME, DECEMBER 21, 1987

Figure 3. Scanned image containing halftone image area(s). Sampling resolution is 300/in; rendering resolution is 375/in.

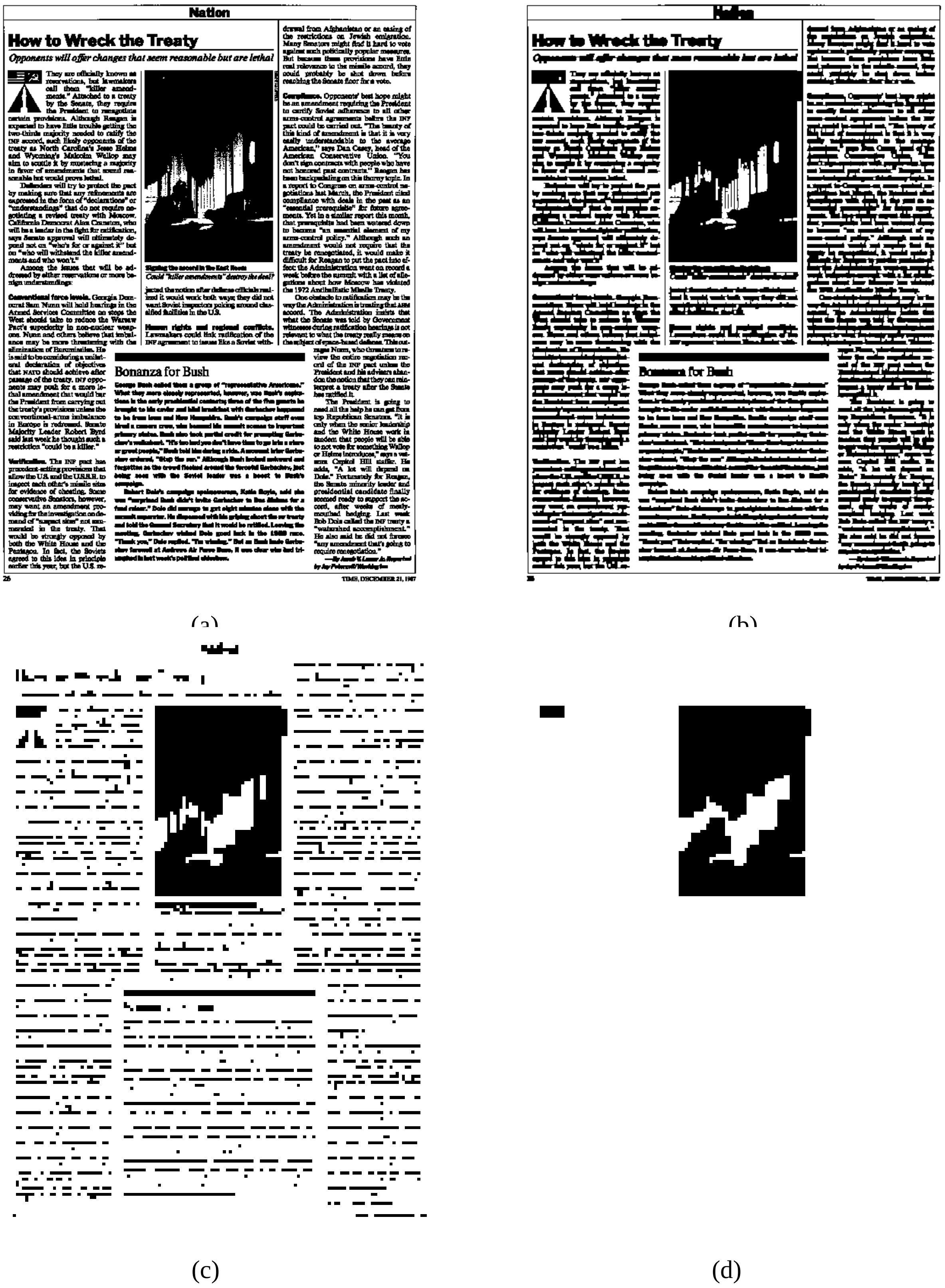


Figure 4. (a) 4x reduction with $m = 1$ for each stage. Resolution: sampling (75/in), rendering (196/in). (b) Closing with 3x3 SE. Resolution: same as (a). (c) Further 4x reduction with $m = 4$ for each stage. Resolution: sampling (19/in), rendering (49/in). (d) Opening with 3x3 SE. Resolution: same as (c).

In addition to the mask region over the halftone image, a small mask region covers a graphic figure in the upper-left corner. This was left after the final opening with a 3x3 brick SE. If a 4x4 brick SE had been used, or if the image had been further reduced 2x (to 32x reduction from the initial sampling resolution) and then opened with a 2x2 brick SE, this region would have been removed. For complicated images, small graphics and halftone regions can be sieved by opening with a set of SEs.

In part because of the final opening, the mask in Figure 4d, if expanded to full resolution, would not not cover all the pixels in the light image areas. This can be rectified in a number of ways. A simple approach, that is effective for images with rectangular image regions (which constitute the majority of cases), is to identify the image regions by computing the bounding box of connected components in the mask. This is nearly instantaneous at the low resolution of 19 pixels/inch. To accomodate non-rectangular image regions, the mask can either be closed with a rectangular or square SE, or it can be subjected to some number of iterations of a morphological bounding box filling algorithm[1]. Either way, the mask regions will be solidified to some extent, but without expansion of the boundaries.

5 Discussion

The search for fast and effective methods for characterizing some short-range texture properties of binary images led directly to the use of threshold reduction as a key technique for multiresolution image analysis. Threshold reduction is closely related to image morphology, and morphological operations are useful in conjunction with threshold reduction for performing image analysis within the image domain.

We have shown how the basic operations can be optimized to enable rapid segmentation of binary images. The example given for motivating the use of threshold reduction, halftone image segmentation, is only one of a large set of segmentation tasks on document images for which threshold reduction and morphology are well-suited. For example, word boundaries or word masks are easily found using threshold reduction, with a small threshold to close intra-word spaces and augmented by morphological operations such as closing. For this case, reduction can typically be carried down to a sampling resolution of about 40/in, which is still high enough to keep words from joining.

Some image analysis can be performed entirely in the image domain, with only image processing operations. For example, reduction/morphology/expansion cycles can be used to create full resolution separation masks for selected regions. This can be understood as a multiresolution pixel labelling process, and some properties of such cycles have been derived. Often it is useful to extract information *about* the image, in a form that does not directly label individual pixels. A typical example is the determination of bounding boxes for regions that are computed at low resolution after a cascade of threshold reductions augmented by morphology.

6 Appendix I: Implementation of threshold reduction by logical operations

In the procedure outlined above for $r = 2$, a threshold reduction is effected by a sequence of logical operations between rows and columns of an r x r tile. This appendix gives a general method for finding a set of m = r^2 boolean operators, implementable by row and column raster operations, that map an r x r tile to a single pixel, such that for $i = 1, 2...m$, there exists an operator with the following property: for all r x r bit arrays with fewer than i ON pixels, the result is a single OFF pixel, and for all arrays with i or more ON pixels, the result is a single ON pixel.

The solution is not unique, even with the following constraints:

1. Row operations are done before column operations.

2. Only "and" and "or" operations are allowed between rows and columns.

For a 2x2 reduction, there are only 2^4 configurations, and the minimum mapping operators can easily be found by considering these 16 cases. However, the number of configurations grows exponentially with the power r^2: for $r = 3$ there are $2^9 = 512$ configurations; for $r = 4$ there are $2^{16} = 65,536$; etc. Operators that work over such large sets can be found by decomposition into products of particular row and column operators that exploit symmetries. We choose one-dimensional row and column operators that are threshold operators for 1xr and rx1 arrays. Their product can then be used to form a basis set for the two-dimensional rxr operators, from which the threshold operators are formed by boolean combinations.

6.1 Threshold operators for $r = 2$ reduction

The requisite thresholding operators on either rows or columns are

a:	**or**	1 or more ON bits
b:	**and**	2 ON bits

The four products of these operators form a basis set of operators on 2x2 bit arrays, where the first operation is between rows and the second is between columns:

aa:	**or/or**	1 or more ON bits
ab:	**or/and**	at least 1 ON bit in each column
ba:	**and/or**	at least one column with 2 ON bits
bb:	**and/and**	all 4 ON bits

The text on the right describes those 2x2 arrays for which the basis operator returns an ON bit. The operators aa and bb are clearly the required ones for thresholding at 1 and 4 ON bits, respectively. To find thresholding operators for 2 and 3 ON bits, we must form boolean combinations of these basis operators.

From the 16 possible 2x2 bit arrays, choose a *canonical* subset by applying the following two reduction rules, which follow directly from the use of one-dimensional threshold operators:

1. *The position of ON bits within a column does not matter. Thus, put the ON bits in each column in the uppermost rows.*
2. *The columns can be permuted. Thus, arrange the columns to have the number of ON bits in each column decreasing to the right.*

Using the notation that an rxr canonical array with $e(j)$ entries in the j^{th} column is denoted

$$r\{e(1),\ ...\ , e(r)\}$$

and grouping them into sets that have exactly one, two, three, and four ON bits, the set of 2x2 canonical arrays is

$2\{1, 0\}$	[one ON bit]
$2\{1, 1\}$ and $2\{2, 0\}$	[two ON bits]
$2\{2, 1\}$	[three ON bits]
$2\{2, 2\}$	[four ON bits]

These canonical arrays should be visualized as 2-dimensional binary objects, onto which the basis operators map as:

$2\{1, 0\}$	<==>	aa
$2\{1, 1\}$	<==>	ab
$2\{2, 0\}$	<==>	ba
$2\{2, 1\}$	<==>	ab and ba together
$2\{2, 2\}$	<==>	bb

The threshold operators are then the union of operators specific to each canonical array, which in general are intersections of the basis operators. The 2x2 threshold operators II_1, II_2, II_3, and II_4 are seen to be

II_1:	aa	all with 1 or more ON bits
II_2:	$ab \cup ba$	all with 2 or more ON bits
II_3:	$ab \cap ba$	all with 3 or more ON bits
II_4:	bb	all 4 ON bits

This is identical to Table 1. Note that the union of basis operators ab and ba projects all arrays represented by $2\{1, 1\}$ and $2\{2, 0\}$, the two canonical arrays for 2 ON bits. Hence, operator II_2 is the *union* of these two basis operators. Also, there is only one canonical array for 3 ON bits, and the threshold operator II_3 corresponding to this array requires the *intersection* of the basis operators ab and ba.

6.2 Threshold operators for $r = 3$ reduction

We outline the extension to $3x3 \Rightarrow 1x1$ reduction of the method described above. As before, start with one-dimensional threshold operators for either rows or columns, which are now composed of more than one operation. Denote an operator

$$\mathbf{or}_{i,j}$$

to mean the **or** of i and j, where $i, j = \{1, 2, 3\}$ and $i \neq j$. Then one way to write the three one-dimensional threshold operators for rows or columns is

a:	$\mathbf{or}_{1,2} \cup \mathbf{or}_{1,3}$	[1 or more ON bits]
b:	$\mathbf{and}_{1,2} \cup \mathbf{and}_{1,3} \cup \mathbf{and}_{2,3}$	[2 or more ON bits]
c:	$\mathbf{and}_{1,2} \cap \mathbf{and}_{1,3}$	[all 3 ON bits]

As before, form a basis set of nine operators on the 3x3 bit arrays from products of these row and column operators. When these operators act on the 3x3 bit arrays, they give an ON bit for the stated subset of arrays:

aa:	1 or more ON bits
ab:	at least 1 ON bit in 2 columns
ac:	at least 1 ON bit in 3 columns
ba:	at least 2 ON bits in 1 column
bb:	at least 2 ON bits in 2 columns
bc:	at least 2 ON bits in 3 columns
ca:	at least 3 ON bits in 1 column
cb:	at least 3 ON bits in 2 columns
cc:	all 9 bits ON

To form the threshold operators as boolean combinations of these basis operators, construct all canonical 3x3 bit arrays that are distinct in the sense of the reduction rules given above. They are

$3\{1, 0, 0\}$	[one ON bit]
$3\{1, 1, 0\}$, $3\{2, 0, 0\}$	[two ON bits]
$3\{1, 1, 1\}$, $3\{2, 1, 0\}$, $3\{3, 0, 0\}$	[three ON bits]
$3\{2, 1, 1\}$, $3\{2, 2, 0\}$, $3\{3, 1, 0\}$	[four ON bits]
$3\{2, 2, 1\}$, $3\{3, 1, 1\}$, $3\{3, 2, 0\}$	[five ON bits]
$3\{2, 2, 2\}$, $3\{3, 2, 1\}$, $3\{3, 3, 0\}$	[six ON bits]
$3\{3, 2, 2\}$, $3\{3, 3, 1\}$	[seven ON bits]
$3\{3, 3, 2\}$	[eight ON bits]
$3\{3, 3, 3\}$	[nine ON bits]

The action of the nine basis operators given above leads to an expression (in general, as an intersection of basis operators) for each canonical array. For example, the canonical array $3\{2, 1, 1\}$ is projected by the intersection $ba \cap ac$, the array $3\{2, 2, 0\}$ is projected by bb, and $3\{3, 1, 0\}$ is projected by $ab \cap ca$. The threshold operators III_1 - III_9 are found as the union of these generally composite operators corresponding to the canonical arrays. Thus, for example,

$$III_4 = (ba \cap ac) \cup bb \cup (ab \cap ca)$$

Terms can often be reduced, and sometimes they can be expressed using operators for lower order reduction; e.g.,

$$III_3 = ac \cup II_3 \cup ca$$

Note, however, that the a and b operators in II_3 are the threshold operators for a 3x1 (or 1x3) bit array, not those for a 2x1 array.

6.3 Threshold operators for $r = 4$ reduction

The four one-dimensional 4x1 or 1x4 threshold reduction operators can be written in a number of ways; e.g.,

a:	$\mathbf{or}_{1,2} \cup \mathbf{or}_{3,4}$	[1 or more ON bits]
b:	$(\mathbf{or}_{1,2} \cap \mathbf{or}_{3,4}) \cup (\mathbf{or}_{1,4} \cap \mathbf{or}_{2,3})$	[2 or more ON bits]
c:	$(\mathbf{and}_{1,2} \cup \mathbf{and}_{3,4}) \cap (\mathbf{and}_{1,4} \cup \mathbf{and}_{2,3})$	[3 or more ON bits]
d:	$\mathbf{and}_{1,2} \cap \mathbf{and}_{3,4}$	[all 4 ON bits]

Construction of the 16 two-dimensional $r = 4$ threshold operators proceeds as before.

6.4 Symmetry properties of threshold reduction operators

From the foregoing examples, it is apparent that the threshold reduction operators satisfy the following symmetries:

1. The two-dimensional operators are symmetric with respect to the order of row and column operations.

2. For any dimensionality, if n is the cardinality of the tile, so that $1 \leq m \leq n$, then the threshold $n - m$ operator can be found from the threshold m operator by interchanging all bit unions with intersections (this includes swapping **and**s with **or**s, as written above).

The second symmetry relation follows from an underlying rank order duality between foreground and background pixels: a filter of cardinality n that projects for m or greater foreground pixels is identical to one that projects for $n - m$ or less background pixels.

7 Appendix II: Lookup tables for 2x2 reduction

Efficient implementation of column reduction operations for the 2x2 operators II_1–II_4 requires table lookup. We give two 2^{16}-entry tables, one for $\mathbf{or}$ and one for $\mathbf{and}$, that are indexed by sixteen bits in the intermediate image (generated by raster operations on the rows), and contain eight bits of the reduced image that correspond to the pair-wise $\mathbf{or}$-ing or $\mathbf{and}$-ing of the index bits, respectively.

These lookup tables are generated from an iterative algorithm that does not require bit masking. For the OR table with a 16-bit index i, the 8-bit table values $t(i)$ are generated by

```
$t(0) = 0, i = 1;$
**for** $(d = 0 ... 7)$
    $i_0 = 2^{2d};$            /* start value of i */
    $t_0 = 2^d;$               /* increment value of t */
    **for** $(r = 1 ... 3)$
        **for** $(k = 1 ... i_0 - 1)$
            $t(i) = t(k) + t_0;$
            $i = i + 1;$
```

Similarly, a 16-bit index AND table is generated by

```
$t(0) = 0, i = 1;$
**for** $(d = 0 ... 7)$
    $i_0 = 2^{2d};$            /* start value of i */
    $t_0 = 2^d;$               /* increment value of t */
    **for** $(r = 1 ... 3)$
        **if** $(r < 3)$  $tInc = 0;$
            **else**  $tInc = t_0;$
        **for** $(k = 1 ... i_0 - 1)$
            $t(i) = t(k) + tInc;$
            $i = i + 1;$
```

The size of each lookup table is determined by the maximum value of the parameter d. Specifically, the number of index bits is $2(d_{max} + 1)$. For example, an 8-bit index (2^8-entry) set of tables can be generated by letting d run from 0 to 3.

With this hybrid (rasterop/lookup) implementation, the computation time is divided nearly equally between the row logical operations and the subsequent column lookup operations.

References

[1] D. S. Bloomberg, "Multiresolution Morphological Approach to Document Image Analysis", submitted to *Int. Conf. on Document Analysis and Recognition,* Saint-Malo, France, Sept-Oct 1991.

[2] D. S. Bloomberg and P. Maragos, "Generalized Hit-Miss Operations," in *SPIE Conf. on Image Algebra and Morphological Image Processing,* Vol. 1350, pp. 116-128, July 1990.

[3] P. J. Bones, T. C. Griffin, and C. M. Carey-Smith, "Segmentation of document images," *SPIE Symp. on Electronic Imaging Science and Technology,* Vol. 1258, Feb. 1990.

[4] P. J. Burt, “The Pyramid as a Structure For Efficient Computation”, *Multiresolution Image Processing and Analysis*, pp. 6-35, Berlin: Springer, 1984

[5] R. M. Haralick, S. R. Sternberg and X. Zhuang, “Image Algebra Using Mathematical Morphology,” *IEEE Trans. PAMI*, Vol. 9, pp. 532-550, July 1987.

[6] R. M. Haralick, C. Lin, J. Lee, X. Zhuang, “Multi-resolution morphology,” *Int. Conf. on Computer Vision, London*, pp. 516-520, June 1987.

[7] R. M. Haralick, X. Zhuang, C. Lin and J. Lee, “Binary Morphology: Working in the Sampled Domain,” *CVPR ’88, Ann Arbor, MI*, pp. 780-791, June 1988.

[8] P. Maragos and R. W. Schafer, “Morphological Filters - Part II: Their Relations to Median, Order-Statistic, and Stack Filters,” *IEEE Trans. Acoust. Speech Signal Process.*, ASSP-35, pp. 1170-1184, Aug. 1987; *ibid.*, ASSP-37, p. 597, Apr. 1989.

[9] J. Serra, *Image Analysis and Mathematical Morphology*, Acad. Press, 1982.

[10] S. L. Tanimoto, “A hierarchical cellular logic for pyramid computers”, *J. Parallel and Distributed Computing*, Vol 1, pp. 105-132, 1984.

[11] S. L. Tanimoto, “Paradigms for pyramid machine algorithms”, in *Pyramidal Systems for Computer Vision*, ed. V. Cantoni and S. Levialdi, NATO ASI Series, Vol. F25, pp. 173-194, Springer Verlag, 1986.