tolerate (detect) $2^{m-i} - 1/2(2^{m-i} - 1)$ faults in it (in the absence of faults elsewhere). Hence, FS is not "hard core."

IV. REMARKS

1) The carry-save adders as presented in this section form complete arithmetic and logic units by themselves since the elementary logical functions on two vectors can be obtained at the outputs of decoder C by a suitable combination of inputs to the adder. For example, to obtain AND(OR) function on two vectors X and Y we may have X and Y and all 0(1) as imputs to the carry-save adders.

2) The encoders for RMC like any other linear codes are synthesized by using EX-OR gates. One of the attractive features of RMC's is that they are majority logic decodable codes which require fewer complex decoders than the conventional multiple error correcting codes. A decoder for a t error correcting majority logic decodable code is synthesized by using EX-OR gates and $2t$-input majority gates (which can be synthesized as an AND-OR network or as a threshold gate). Furthermore, the distance 4 RMC is the same as the SEC–DED Hamming code and therefore, it will require a much simpler decoder.

It may be again noted that if only error detection is desired, it can be accomplished by a simple EX-OR network to compute the syndrome and an OR gate to check whether it is zero or not.

In this case, the complexity of the checker may compare favorably to that in the replication scheme. For the replication scheme, the checker will consist of an array of $t + 1$-input gates capable of detecting any dissimilarity in its inputs.

3) In the following, we will discuss some broader application of the realization presented in this paper. RMC's are parity check codes and they can be well used to protect the memory. Consider a hypothetical computer. We can encode the words in the momery by a ith-order RMC. We can also synthesize the arithmetic and logic unit (ALU) as a fault-tolerant carry-save adder as outlined before. Since the decoder circuit can be expensive, we can use a single $2i$th-order RMC decoder and time share it between memory and ALU (a $2i$th-order decoder can decode the ith-order code with reduced error correction capability). Any word that is fetched from the memory can be first decoded by the decoder for correcting any errors. If a logic (arithmetic) operation is desired, then an appropriate combination of inputs is provided at the inputs to the carry-save adders. The results of computation can then be decoded by the decoder in one or few cycles as the case may be. These kinds of operations can be controlled easily by a microprogrammed control unit. Thus, at least theoretically we can use one kind of code in the memory as well as in the entire arithmetic and logic unit.

V. CONCLUSIONS

We have presented techniques for designing fault-tolerant adders by using RMC's and also have indicated some broader application of these codes. With the results presented in this correspondence, we have now one class of codes applicable to the entire ALU. In the earlier paper [2], it has been established that the use of RMC's yields redundancy of the order of duplication (for the conventional replication scheme, the complexity of the redundant scheme is N times the original scheme where $N \geq 3$) in designing large sacle fault-tolerant logic processors. Thus, the techniques presented in this correspondence may be entirely justifiable in large systems.

ACKNOWLEDGMENT

I am thankful to Prof. S. M. Reddy for his many helpful suggestions during the course of this work.

REFERENCES

[1] J. L. Massey and O. N. Garcia, "Error-correcting codes in computer arithmetic," in *Advances in Information Systems Science*, vol. IV. New York: Plenum, 1972, ch. 5.
[2] D. K. Pradhan and S. M. Reddy, "Error control techniques for logic processors," *IEEE Trans. Comput.*, vol. C-21, pp. 1331–1336, Dec. 1972.
[3] I. Flores, *The Logic of Computer Arithmetic.* Englewood Cliffs, N. J.: Prentice-Hall, 1963.
[4] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in *Automata Studies* (Annals of Mathematical Studies, vol. 34), C. E. Shannon and J. McCarthy, Eds. Princeton, N. J.: Princeton Univ. Press, 1956.
[5] W. W. Peterson, *Error Correcting Codes.* Cambridge, Mass.: MIT Press, 1961.
[6] D. K. Pradhan and S. M. Reddy, "A design technique for synthesis of fault-tolerant adders," in *Dig. Papers, 1972 Int. Symp. Fault-Tolerant Comput.*, Newton, Mass., pp. 20–24.

A Threshold Selection Technique

J. S. WESZKA, R. N. NAGEL, AND A. ROSENFELD

***Abstract*—Threshold selection for picture segmentation is relatively easy when the frequency distribution of gray levels in the picture is strongly bimodal, with the two peaks comparable in size and separated by a deep valley. This report describes a method of handling cases in which the peaks are very unequal in size and the valley is broad. A Laplacian operation is applied to the picture to determine points that lie on or near the edges of objects. Threshold selection becomes easier when the frequency distribution of gray levels *of these points* is used.**

***Index Terms*—Laplacian operator, picture processing, scene analysis, segmentation, thresholding.**

A variety of techniques have been proposed for determining thresholds at which to slice a picture in order to extract objects from their background. If the proportion of the picture area occupied by the objects is known, one can slice at a level such that just the desired fraction of the picture points are above threshold [1]. If the gray level ranges occupied by objects and background are sufficiently well separated (e.g., the picture contains dark objects on a light background, or vice versa) there will be a valley in the gray level histogram between the two peaks corresponding to these ranges, and one can slice at a threshold located at the bottom of this valley [2]. Other methods based on combinations of these ideas have been proposed (e.g., [3]). Still other methods attempt to define thresholds that vary from one part of the picture to another, since no single threshold may give good results for the entire picture; for an interesting recent example see [4], where local thresholds for text are chosen so as to obtain a specified linewidth after thresholding.

A limitation of the "valley" method described above arises when the objects occupy only a small portion of the picture. Here the peaks will be very different in size, making it difficult to locate the valley uniquely. This note describes a technique useful in such cases. Basically, in this technique, the threshold is selected not from the histogram of the entire picture, but from a histogram of only those points which take on high values under a digital "Laplacian" operation.

As an example, consider the picture of a signature shown in Fig. 1, in which the signature points occupy only a small percentage of the total picture. The resulting gray level histogram (Fig. 2) exhibits two peaks of very unequal amplitude separated by a broad valley. Notice that the valley contains nearly half as many points as the small peak. This makes the selection of a gray level for thresholding more difficult than in the case of a histogram containing two nearly equal peaks. Thresholding at a level in the valley which is very close to the large peak (e.g., gray level 11) may result in the inclusion of unwanted background points, while slicing at a level so as to eliminate all valley points (e.g., gray level 45) may cause the exclusion of many signature points. The results of thresholding Fig. 1 at these gray levels (11 and 45) are shown in Fig. 3.

Suppose now that we filter the given picture by applying a local edge detector, e.g., a digital "Laplacian" operator (the absolute difference between the gray level at each point and the average of its eight neighbors' gray levels). The result of this for Fig. 1 is shown in Fig. 4. The points where the filtered picture has high values tend to lie in the gray level transition areas near the border of the signature. In consequence, about equal numbers of these points will be light and dark. Thus a gray level histogram *of only these points* should be more symmetrical than the histogram of the entire picture.

Manuscript received April 1, 1974; revised July 8, 1974. This work was supported by the Air Force Office of Scientific Research under Contract F-44620-72C-0062. The work herein is an extension of ideas developed in connection with R. N. Nagel's Ph.D. dissertation.

The authors are with the Computer Science Center, University of Maryland, College Park, Md.

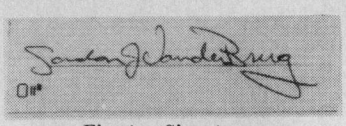


Fig. 1. Signature.

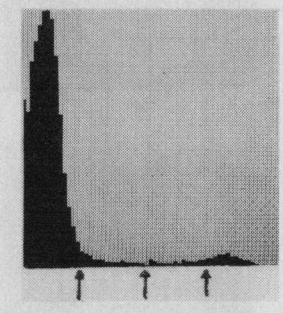
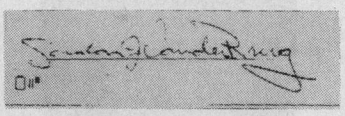


Fig. 2. Gray level histogram for Fig. 1; the thresholds used in Figs. 3 and 7 are marked.

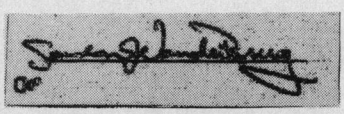
(a)



(b)

Fig. 3. Results of thresholding Fig. 1 at (a) 11, (b) 45.

Fig. 4. Laplacian values for Fig. 1; only the 6 most significant bits are displayed.



(a)

(b)

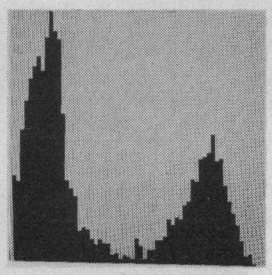
(c)

Fig. 5. Points where the Laplacian has values above its 80th, 85th, and 90th percentiles.

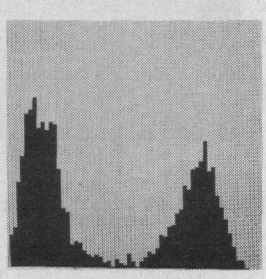
Our choice of the Laplacian as a border detector should not only increase the symmetry of the histogram, but should also tend to deepen and sharpen the valley that separates the peaks. This is because the Laplacian, as a second-difference operator, tends to produce low values at points on "ramps," i.e., points of nearly constant slope. Suppose that an ideal (blurred) "edge" consists of two plateaus of different heights separated by such a ramp. The filtered picture will then have low values on the plateaus themselves, and low values in the middle of the ramp, but it will have high values in the areas of high second derivative at the edges of the plateaus, and on the "shoulders" where the ramp merges with the plateaus. Hence, thresholding the filtered picture can be used to suppress gray levels that lie midway between those of the plateaus (objects and background) while retaining those at either end. This deepens the valley of the resulting histogram at its center, providing a natural place for choosing a threshold for the original picture.

The points where the Laplacian picture has values above its 80th, 85th, and 90th percentiles are shown in Fig. 5. (One would not want to use *too* few points of high Laplacian value, since the histogram for these points might no longer be reliable.) The gray level histograms for these sets of points in the original picture are shown in Fig. 6. It is seen that these three histograms resemble each other strongly, and that all three have two major peaks separated by a deep valley. For all of these histograms, gray level 28 is a reasonable choice for the bottom of the valley. The picture resulting from thresholding at this level is shown in Fig. 7.

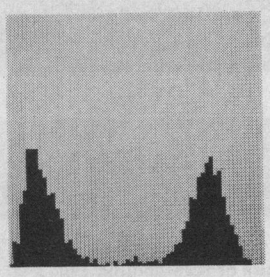
Similar results can also be obtained for other types of pictures. One example is the chromosome spread shown in Fig. 8; its histogram is shown in Fig. 9. Results of two bad slices, at 25 and 52, are shown in Fig. 10. The Laplacian technique, in this case, leads to a slice level of 37; Fig. 11 shows the results of slicing at this level. (For brevity, the intermediate steps are omitted here; see [5] for the details.) In the chromosome case it was found necessary to use "coarse" Laplacians, based on averages over a 7×7 or 11×11 square centered at the given point, rather than the 3×3 square of the ordinary Laplacian. It appears that the chromosome edges are too gradual to be detected adequately by the fine Laplacian; most of the



(a)



(b)



(c)

Fig. 6. Histograms of the gray levels of Fig. 1, using only the sets of points shown in Fig. 5.

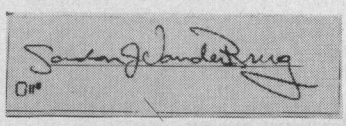


Fig. 7. Result of thresholding Fig. 1 at 28 (as suggested by the histograms in Fig. 6).

Fig. 8. Chromosomes.

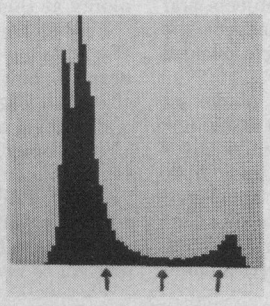
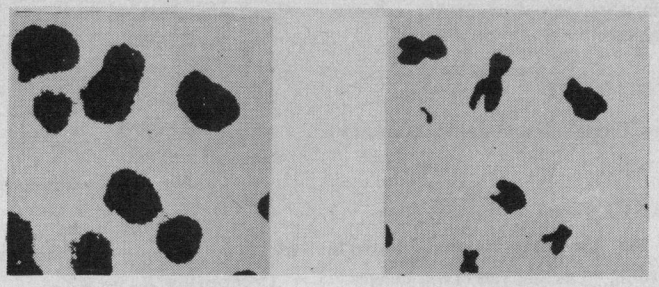


Fig. 9. Gray level histogram for Fig. 8; the thresholds used in Figs. 10 and 11 are marked.



(a) (b)

Fig. 10. Results of thresholding Fig. 8 at (a) 25; (b) 52.

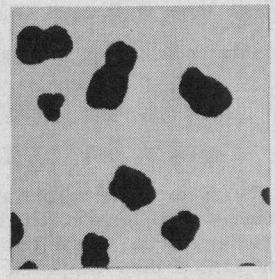


Fig. 11. Results of thresholding Fig. 8 at 37.

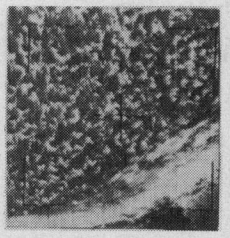


Fig. 12. Cloud cover.

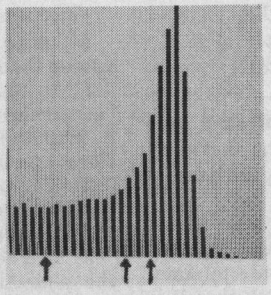
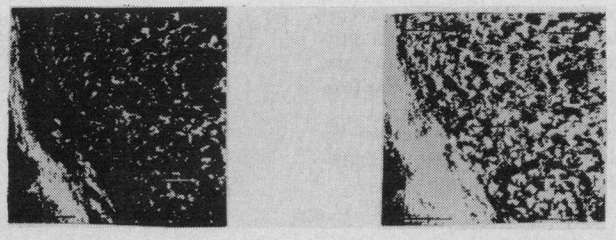


Fig. 13. Gray level histogram for Fig. 12; the thresholds used in Figs. 14 and 15 are marked. (There are no odd-numbered gray levels, because an original 32-level picture has been stretched over 64 levels.)



(a) (b)

Fig. 14. Results of thresholding Fig. 12 at (a) 9; (b) 35.

high Laplacian values originate from background noise rather than from points near edges.[1]

Analogous results for a cloud cover picture (Fig. 12), whose histogram contained only one peak and a "shoulder" (Fig. 13), are shown in Fig. 14 (bad slices, at 9 and 35) and Fig. 15 (good slice, at 24). Here Laplacians of all three sizes (3×3, 7×7, 11×11) yielded the same preferred slice level.

This correspondence has not addressed the problem of automatically choosing the threshold from the original or "enhanced" histogram, nor has it considered the much deeper problem of evaluating the "goodness" of a selected threshold. There is no guarantee that the thresholds found by this method are really appropriate for segmentation of the given scene. However, it is felt that the sub-

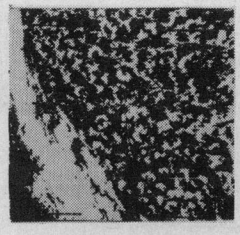


Fig. 15. Results of thresholding Fig. 12 at 29.

[1] One could probably determine the appropriate size of Laplacian automatically, by analyzing the spatial frequency content of the given picture (perhaps assuming some knowledge about the noise spectrum); but this was not attempted in the present study. It should also be pointed out that the results here may depend on the particular digital approximation to the Laplacian that was used; it would be of interest to try other approximations (see, e.g., [6, pp. 106–113]).

jectively pleasing results obtained in a variety of examples indicate that the method may be of practical use.

ACKNOWLEDGMENT

The authors wish to thank Andrew Pilipchuk and Janet James for their help in preparing this report.

REFERENCES

[1] W. Doyle, "Operations useful for similarity-invariant pattern recognition," *J. Ass. Comput. Mach.*, vol. 9, pp. 259–267, 1962.
[2] J. M. S. Prewitt and M. L. Mendelsohn, "The analysis of cell images," *Ann. N. Y. Acad. Sci.*, vol. 128, pp. 1035–1053, 1966.
[3] T. Sakai, M. Nagao, and S. Fujibayashi, "Line extraction and pattern detection in a photograph," *Pattern Recognition*, vol. 1, pp. 233–248, 1969.
[4] A. K. Griffith, "The GRAFIX I image processing system," in *Proc. Comput. Conf.*, 1974, pp. 267–272.
[5] J. S. Weszka, R. N. Nagel, and A. Rosenfeld, "A technique for facilitating threshold selection for object extraction from digital pictures," Comput. Sci. Ctr., Univ. of Maryland, College Park, Md., Tech. Rep. 243, May 1973.
[6] J. M. S. Prewitt, "Object enhancement and extraction," in *Picture Processing and Psychopictorics*, B. S. Lipkin and A. Rosenfeld, Ed. New York: Academic, 1970, pp. 75–149.

Comments on "Cause-Effect Analysis for Multiple Fault Detection in Combinational Networks"

CARL D. LATINO AND JON G. BREDESON

***Abstract*—Bosson and Hong[1] have developed an effective procedure for multiple fault detection. However, their claim that their procedure gives near-minimal results is not necessarily the case.**

Poage [1] has considered the problem of generating test patterns for multiple fault detection. His method can be used to find a minimal set of fault-detection tests; however, it has the drawback that it is computationally complex.

Bossen and Hong[1] claim to have a considerably simpler procedure which gives near-minimal results. To illustrate their procedure, they worked out an example which generates six test patterns for a circuit whose known minimum is five. By their nonredundant (NR) procedure, the example should have given eight test patterns, not six. Since eight is considerably more than five, the ability of the procedure to generate near-minimal results is questionable.

In order to demonstrate the errors, Example 2 considered by Bossen and Hong will be worked out in detail using their own procedures (General (G) and NR). The circuit is given in Fig. 1.

By algorithm G,

$$F' = \mathfrak{F} = wx\bar{y} + wx\bar{z} + \bar{w}yz + \bar{x}yz$$

$$\bar{F}' = \overline{wx\bar{y} + wx\bar{z} + \bar{w}yz + \bar{x}yz}$$

$$\bar{\mathfrak{F}}' = wxyz + y\bar{y}z + yz\bar{z} + w\bar{w}x + \bar{w}\bar{y} + \bar{w}\bar{z} + wx\bar{x} + \bar{x}\bar{y} + \bar{x}\bar{z}$$

$$\bar{\mathfrak{F}} = wxyz + \bar{w}\bar{y} + \bar{w}\bar{z} + \bar{x}\bar{y} + \bar{x}\bar{z}$$

$$X_j = \{wx, w\bar{y}, x\bar{y}, x\bar{z}, w\bar{z}, \bar{w}y, \bar{w}z, yz, \bar{x}y, \bar{x}z\}$$

$$X_m = \{wxy, wxz, wyz, xyz, \bar{y}z, yz, y\bar{z}, \bar{w}x, wx, w\bar{x}, \bar{w}, \bar{x}, \bar{y}, \bar{z}, w, x, y, z\}.$$

Computing $\hat{X}_j = \bar{\mathfrak{F}}X_j$,

$$\hat{X}_j = \{wxyz, \bar{w}xy\bar{z}. \bar{w}x\bar{y}\bar{z}, \bar{w}\bar{y}z, \bar{w}\bar{x}\bar{y}z, \bar{w}x\bar{y}\bar{z}, \bar{w}x\bar{z}, \bar{w}\bar{x}yz, \bar{w}y\bar{z}, w\bar{x}\bar{y}, w\bar{x}\bar{y}\bar{z},$$
$$\bar{w}\bar{x}\bar{y}z, \bar{x}\bar{y}z, w\bar{x}\bar{y}\bar{z}, w\bar{x}\bar{z}, \bar{w}\bar{x}yz, \bar{x}yz\}.$$

Abnormal true tests are therefore

$$\{wxyz, \bar{w}x\bar{y}\bar{z}, w\bar{x}\bar{y}\bar{z}, \bar{w}\bar{x}yz, \bar{w}\bar{x}\bar{y}z\}$$

Manuscript received January 15, 1974; revised April 17, 1974.
The authors are with the Department of Electrical Engineering, Pennsylvania State University, University Park, Pa. 16802.
[1] D. C. Bossen and S. J. Hong, *IEEE Trans. Comput. (Special Issue on Fault-Tolerant Computing)*, vol. C-20, pp. 1252–1257, Nov. 1971.

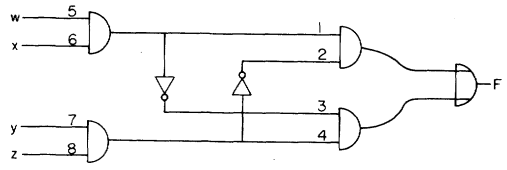


Fig. 1. Circuit considered by Bossen and Hong.

which is the same as in the paper.

Computing $\hat{X}_m = \mathfrak{F}X_m$,

$$\hat{X}_m = \{wxy\bar{z}, wx\bar{y}z, w\bar{x}yz, \bar{w}xyz, wx\bar{y}\bar{z}, \bar{w}\bar{x}yz, \cdots\}.$$

Abnormal false tests are therefore

$$\{wxy\bar{z}, wx\bar{y}z, w\bar{x}yz, \bar{w}xyz, wx\bar{y}\bar{z}, \bar{w}\bar{x}yz\}.$$

The terms $wx\bar{y}\bar{z}$ and $\bar{w}\bar{x}yz$ *were not* included in the paper. These terms represent

$$wx\bar{y}\cdot\bar{z}, \qquad wx\bar{y} \in \mathfrak{F} \text{ and } \bar{z} \in X_m$$

$$\bar{w}yz\cdot\bar{x}, \qquad \bar{w}yz \in \mathfrak{F} \text{ and } \bar{x} \in X_m.$$

The preceding implies eleven test patterns, not nine as Bossen and Hong claim. The aforementioned omissions cited also increase the NR result by two test patterns as can be easily shown. The solution can be reduced to seven terms if the F simplified cause-effect equation is used.

CONCLUSION

Although Bossen and Hong's procedure is considerably less involved than Poage's procedure, it does suffer three drawbacks.

1) In order to be assured that the best attainable solution has been achieved, the NR procedure must be performed on both F and $\bar{F}$. This approximately doubles the amount of work required to find the solution.

2) The conclusion that the results obtained are near minimal is questionable. In the example cited, the minimal result is five, whereas the best solution found required seven terms.

3) If a test set is found for a circuit without prior knowledge of the minimal solution, there is no way of telling how close to minimal the solution really is.

REFERENCES

[1] J. F. Poage, "Derivation of optimum tests to detect faults in combinational circuits," in *Mathematical Theory of Automata*. New York: Polytechnic Press, 1963, pp. 483–528.

Authors' Reply[2]

D. C. BOSSEN AND S. J. HONG

The comment is valid in that the authors followed the given algorithm G in footnote 1 to the letter to come to their conclusion. However, the spirit of the algorithm was somewhat misrepresented in the original paper. A clearer version for the algorithm G follows. The two additional tests claimed by the commentors do not arise if the following version is carried to the letter. And therefore our original assertion about the near minimality still seems valid. Similarly, the additional tests claimed by the commentors using procedure NR will be eliminated if the following version of algorithm G is used in procedure NR. This answers the commentors conclusion 2) in [1].

ALGORITHM G

$G1$) Derive the simplified network equations F' and $\bar{F}'$.

$G2$) Obtain $\mathfrak{F}$ and $\bar{\mathfrak{F}}$ by further simplifying F' and $\bar{F}'$ using all

[2] Manuscript received March 20, 1974.
The authors are with IBM Systems Development Division, Poughkeepsie, N. Y. 12602.