ECMCS'99
EURASIP Conference
DSP for Multimedia Communications and Services
Kraków, 24-26 June

VECTOR SD-ROM FILTER FOR REMOVAL OF IMPULSE NOISE FROM COLOR IMAGES

Michael S. Moore[1], *Moncef Gabbouj*[2], *and Sanjit K. Mitra*[1]
[1]ECE Department, University of California, Santa Barbara, CA 93106,
e-mail: msmoore@iplab.ece.ucsb.edu
[2]Tampere International Center for Signal Processing
Tampere University of Technology, FIN-33 101 Tampere, Finland
e-mail: moncef@cs.tut.fi

ABSTRACT

One well-studied image processing task is the removal of impulse noise from images. Impulse noise can be introduced during image capture, during transmission, or during storage. The signal-dependent rank order mean (SD-ROM) filter has been shown to be effective at removing impulses from 2-D scalar-valued signals while preserving many details and other features. The algorithm is based on a state-estimation approach where first the state of each sample is determined and then an appropriate replacement is calculated, if required. Excellent results have been obtained for both two-state and multi-state versions of the filter. However, many images are not scalar-valued, but rather vector-valued. For example, color images have three values associated with each sample location. The scalar SD-ROM algorithm uses sorted window values to estimate the state of a sample. There are many ways to sort vectors. In this paper, we describe the strengths of the scalar SD-ROM algorithm and the challenges inherent in developing a vector SD-ROM algorithm. We will also present simulation results obtained using corrupted color images for a proposed vector impulse-removing algorithm.

1 INTRODUCTION

One of the most common image processing tasks involves the removal of noise from images. Noise can be introduced during image capture, during transmission, or during storage. However, most images share characteristics with noise sources to a greater or lesser degree. Therefore, at its core, this task requires a balance between the improvement gained by a particular filter as noise is removed from an image and the degradation introduced by a particular filter as the noise-like components of the original image are also removed.

As a result, many different types of filters have been developed to handle different kinds of noise sources[2]. One common noise model corrupts a signal by introducing impulse noise. In this case, most of the original samples are unaltered, but the few samples that are changed can vary drastically. One group of filters, collectively called decision-based filters[2, 3] or state-conditioned filters[4], estimates the state of the sample in question. If the sample is determined to be uncorrupted, it is passed through the filter unchanged. If the sample is corrupted, an appropriate estimate is chosen to replace it.

The signal-dependent rank order mean (SD-ROM) filter has been shown to be effective at removing impulses from 2-D scalar-valued signals[4]. Excellent results were presented for both a two-state and a multi-state version of the filter. However, the proposed algorithms were only applied to grayscale images. Most images available today are in color. If you consider each pixel in a color image as a vector with three components, grayscale detection techniques can no longer be directly applied.

In this paper, we discuss several algorithms that attempt to extend the grayscale SD-ROM algorithm into a vector-valued environment. In the next section, we describe the strengths of the SD-ROM detection scheme that we would like to preserve in a vector algorithm. We also discuss the challenges inherent in working in a vector-valued domain. In the third section, we describe an SD-ROM-like vector algorithms. The fourth section presents some simulation results for the various algorithms. Finally, the last section sums up our conclusions.

2 BACKGROUND

This section introduces the grayscale SD-ROM algorithm and discusses some of the considerations involved in using the algorithm in color images.

2.1 Grayscale SD-ROM

The SD-ROM algorithm was originally proposed as an alternative to the median filter. The filter has been applied in several applications, such as removing impulses[4], streaks[5], and scratches from images and impulse noise from audio signals[6]. The detection algorithm is fully described in many of these papers[4, 5, 6]. For brevity, we will just summarize the major steps in the two-state algorithm here:

1. A three-by-three window centered around the pixel-of-interest is extracted.
2. The surrounding eight pixels values are sorted.
3. Signed differences are calculated between the center pixel and the sorted values. The differences are calculated differently for the smaller four values and the larger four values.
4. The signed differences are compared to positive thresholds. Four thresholds are applied with mirror symmetry to the four smallest differences and the four highest differences.
5. If any threshold was exceeded, the center pixel is replaced. If no thresholds were exceeded, the center pixel is passed unchanged.

The SD-ROM algorithm is tailored for impulse removal. Impulse noise, as modeled, corrupts a subset of pixels in an image completely. This is in contrast to Gaussian noise sources which corrupt all pixels slightly. Therefore, the algorithm tries to determine whether a pixel is corrupted or not before applying correction. The decision is based on a comparison between the pixel-of-interest and the distribution of surrounding values, some of which may also be corrupted. The challenge for impulse detection algorithms is to distinguish between impulses and image features with wide distributions. In the SD-ROM algorithm, the thresholds allow the response to both impulses and impulse-like image features to be tuned.

Many existing impulse-removal filters rely on a sorted window of surrounding values. However, few use the difference between the center pixel and its local neighborhood. The signed differences represent the degree to which the center pixel is an outlier compared to the other pixels. If the center value is outside either the smallest or largest surrounding values, it is probably an impulse. Therefore, a small threshold is applied to these differences. If the center value is inside the extremes, there is a smaller chance that it is an impulse. Thus, the thresholds increase for the differences closer to the middle of the sorted neighborhood. Figure 1 demonstrates this procedure graphically for one example.

2.2 Color Issues

Color images are much more common than grayscale images and can be just as susceptible to impulse noise. Our goal is to design an algorithm similar to SD-ROM for color images. Ideally, the algorithm would have the following characteristics:

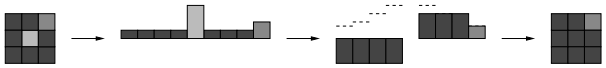


Figure 1. Graphical SD-ROM example. From left to right, you see the original window, the sorted values, the differences with thresholds, and the result. The second threshold was exceeded, so the center pixel was replaced.

1. An order-based detection/estimation approach to impulse noise removal.
2. A method for tuning the algorithm performance so that original image features can be preserved to a greater or lesser degree.
3. An output equal to one of the inputs.

The first two characteristics contain the strengths of the original SD-ROM algorithm. The third item is necessary to avoid the generation of false colors in the filter output.

If the colors associated with each pixel are considered to be the elements of a vector, then the algorithm will have to sort vector-valued elements. There are many ways to sort vectors. Several algorithms have been proposed for removing impulses from color images. These algorithms take various approaches to sorting the vectors. Some common design choices are:

1. Distance versus direction. To order vectors, you need to characterize each vector in terms of a single number. The numbers can then be sorted. Various algorithms use the Euclidean distance between vectors [7], the angle between vectors [8], or both [9].
2. Minimum total distance versus reference vectors. Both the distance and direction measures are applied to just two vectors. You still need some method of choosing which pairs of vectors to compare. One method is to sum the measure for every possible pair of vectors. The totals are then used to sort the vectors [7]. Another approach is to choose a reference vector and compute the measures for the other vectors relative to the reference.
3. Color space transformations. Most color images are transmitted in a red-green-blue (RGB) format. Although convenient for capture, transmission, and display, this format does not reflect the ability of the human visual system to distinguish colors. Some methods use a nonlinear transformation, such as the CIE-LUV color space conversion, to work in a space that is more uniform with respect to human color perception.

3 VECTOR SD-ROM

In our initial implementation of a vector SD-ROM algorithm, we chose to use the total distance between vectors for sorting and we left the vectors in the RGB color space. Specifically, the algorithm:

1. Calculates the distance between all possible pairs of the eight surrounding pixel vectors in a three-by-three window.
2. Sorts the vectors using the sum of the distances to all the other vectors. The pixel with the smallest total distance is the vector median of the surrounding pixels.
3. Compares the distances between the center pixel and the four smallest of the sorted vectors to a set of four increasing thresholds.
4. If any of the thresholds was exceeded, the output of the filter is the vector median of the surrounding values. If no threshold was exceeded, the center pixel is passed unchanged.

This algorithm satisfies all of the requirements in section 2.2. With small thresholds, the filter response is similar to that of a vector median filter. However, the vector median filter removes many small details from a color image. By increasing the thresholds, more detail may be preserved. Because the vector median of the surrounding values is used as the replacement value, no false colors are generated by the algorithm.

4 SIMULATION RESULTS

The proposed algorithm was tested using images with artificially injected impulse noise. All of the original images were 24-bit, RGB color images. A fixed probability of an impulse was assumed for every byte in the image. If a byte was replaced by an impulse, the impulse could take any value in the range [0,255] with uniform probability. Because each byte was considered independently, a given pixel color vector could contain between zero and three impulses. Figure 2 shows an original image. Figure 3(a) is the image corrupted with impulse noise.

Each test image was filtered using the algorithm described in the last section. The algorithm requires the selection of four thresholds. Two methods were used to find the optimum thresholds for each image. The first approach was to change the threshold manually using a trial-and-error technique to find the best-looking result. The second method was to use a search algorithm to find the thresholds with the minimum mean-square-error between the filtered and original images.

Figure 3(b) shows the filter output using hand-tuned thresholds for the Mandrill image with 20 percent corruption. The output of the vector median filter for the same input is shown in figure 3(c). Table 1 summarizes the results, in terms of PSNR, for several images at several levels of corruption using the optimization program. For comparison, the results using the vector median filter are also included. The vector SD-ROM algorithm outperforms the vector median algorithm, especially at low levels of corruption.

Image	% Corrupt	VMedian	VSD-ROM
Mandrill	5	22.65	27.38
	10	22.43	25.66
	20	21.84	23.70
	30	21.05	22.15
	40	19.85	20.46
Fighter	5	33.37	36.92
	10	32.50	35.11
	20	30.28	31.89
	30	26.79	27.70
	40	22.98	23.54
Tulips	5	34.70	39.06
	10	33.39	36.44
	20	30.23	31.95
	30	26.17	27.14
	40	22.09	22.72
Lena	5	33.43	39.26
	10	32.91	37.15
	20	31.62	34.28
	30	29.32	30.67
	40	26.12	26.71

Table 1. Optimization results in PSNR. The vector SD-ROM results were calculated using the MSE optimized thresholds for each example.

5 CONCLUSION

The proposed vector SD-ROM algorithm satisfies the requirements we set for it. It is a vector median-like algorithm, but with more freedom to pass some image details. It incorporates a detection scheme strongly tuned to find impulses in images.

As pointed out in section 2.2, there are several design options that can be used to create variations on this basic algorithm. However, these choices should be made with a particular noise source and application in mind. In future work, we plan to investigate the application of a color SD-ROM algorithm to bad pixel removal in CMOS sensor arrays with a color filter matrix.

ACKNOWLEDGMENTS

This work was supported in part by a University of California MICRO grant with matching support from Optronix Engineering, Lucent Technologies, Xerox Corporation, and Tektronix Corporation. Part of this research was also done as a visiting researcher in the Digital Media Institute at the Technical University of Tampere, in Tampere, Finland.

REFERENCES

[1] A. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, Englewood Cliffs, NJ, 1989.

Figure 2. Original Mandrill image.

[2] J. Astola and P. Kuosmanen, *Fundamentals of Nonlinear Digital Filtering*, CRC Press, New York, 1997.

[3] T. Sun and Y. Neuvo, "Detail-preserving median based filter in image processing," *Pattern Recognition Letters* **15**, pp. 341–347, April 1994.

[4] E. Abreu, M. Lightstone, S. Mitra, and K. Arakawa, "A new efficient approach for the removal of impulse noise from highly corrupted images," *IEEE Trans. on Image Processing* **5**, pp. 1012–1025, June 1996.

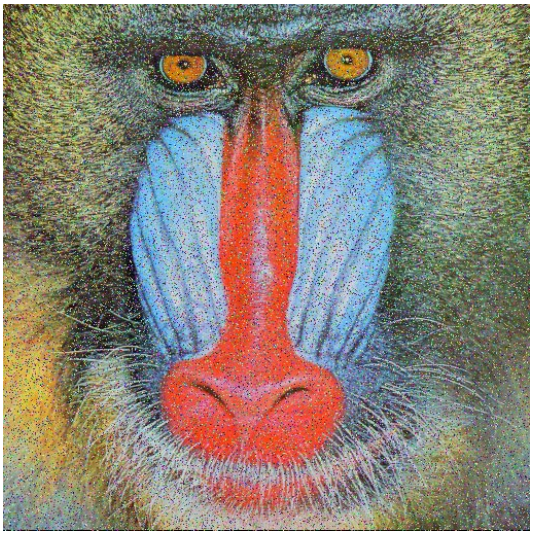
[5] E. Abreu and S. Mitra, "A simple algorithm for restoration of image corrupted by streaks," in *Proceedings of the IEEE Symposium on Circuits and Systems*, vol. 2, pp. 730–733, 1996.

[6] C. Chandra, M. Moore, and S. Mitra, "An efficient method for the removal of impulses from speech and audio signals," in *Proceedings of the IEEE Symposium on Circuits and Systems*, vol. 4, pp. 206–208, 1998.

[7] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proceedings of the IEEE* **78**, pp. 678–689, April 1990.

[8] P. Trahanias and A. Venetsanopoulos, "Vector directional filters - a new class of multichannel image process filters," *IEEE Trans. on Image Processing* **2**(4), pp. 528–534, 1993.

[9] D. Karakos and P. Trahanias, "Combining vector median and vector directional filters: the directional-distance filters," in *Proceedings of ICIP*, vol. 1, pp. 171–174, 1995.



(a)

(b)

(c)

Figure 3. Example images: (a) after 20%corruption, (b) after vector SD-ROM filtering with the thresholds 93, 150, 226, and 234, and (c) after vector median filtering.