Performance Analysis of the Two-State Signal-Dependent Rank Order Mean Filter

Michael S. Moore[*] and Sanjit K. Mitra

Department of Electrical and Computer Engineering
University of California, Santa Barbara
Santa Barbara, CA 93106 USA

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. Excellent results were presented for both a two-state and a multi-state version of the filter. The two-state SD-ROM filter relies on the selection of a set of threshold values. In this paper, we examine the performance of the algorithm with respect to the thresholds. We take three different approaches. First, we discuss the performance of the algorithm with respect to its root signals. Second, we present a probabilistic model for the SD-ROM filter. This model characterizes the performance of the algorithm in terms of the probability of detecting a corrupted pixel while avoiding uncorrupted pixels. Finally, we apply the insight gained from the root signal analysis and the statistical model to optimized thresholds found using a computerized search algorithm for a large number of images and noise conditions.

Keywords: Impulse noise, nonlinear filters, SD-ROM filter, probabilistic model, thresholds

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.[1,2] 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. Although a specific design method was developed for the multi-state SD-ROM, no method was proposed for the simpler two-state algorithm. The two-state SD-ROM relies upon four threshold values. Although guidance based on experience with the filter and several test images was given for selecting threshold values, the thresholds were not directly related to the algorithm performance.

In this paper, we look at the two-state SD-ROM filter in greater detail. First, we discuss the performance of the algorithm with respect to its root signals. The SD-ROM algorithm passes many signals that would normally be altered by a median filter. Second, we present a probabilistic model for the SD-ROM filter. This model characterizes the performance of the algorithm in terms of the probability of correct detection of a corrupted pixel. The model depends on assumed distributions of the background image and the noise source. By numerically maximizing the probability of a correct result, thresholds can be found for given background and noise distributions. Finally, we

[*]Correspondence: Email: msmoore@iplab.ece.ucsb..edu, WWW: http://iplserv.ece.ucsb.edu/users/msmoore

apply the insight gained from the root signal analysis and the statistical model to optimized thresholds found using a computerized search algorithm for a large number of images and noise conditions.

Overall, the approaches presented in this paper are useful for explaining the performance of the SD-ROM filter under various conditions. Although the optimal thresholds were rarely found using indirect techniques, the results support several generalizations that can be made about threshold selection.

2. BACKGROUND

2.1. Median Filter

A key characteristic of impulse noise is the high probability of a large deviation from the original signal value. For well-behaved input signals, the impulses will appear as outliers in the distribution of a local neighborhood of the noisy signal. The process of removing impulses from a corrupted signal can be thought of as estimation of the value of each sample given the noisy signal. A traditional method of estimating the center of a distribution, in the presence of outliers, is to use the median filtering approach attributed to Tukey.[5,6]

In image processing, the median filter operates on a relatively small local region centered on the current pixel of interest. A typical filter uses a three by three or five by five window. Larger windows remove more impulses, but take longer to sort and distort the image in easily apparent ways.[2,7]

The median filter has several desirable characteristics. The median of a set of numbers is the value with the minimum total distance to all of the other values in the set.[8] Therefore the median filter is robust with respect to the outliers. The output of the filter is always one of the input values, so the set of values in an image does not grow with median filtering. The filter also passes several common image structures without change. Signals that are not changed by a filter are called root signals. A signal repeatedly filtered by a median filter will converge to a root signal.[7]

The median filter has root signals that encompass several common image region characteristics. As shown in Figure 1, constant signals, ramp signal, and step edges are passed unchanged (in the absence of impulse noise) by the median filter. However, several other common structures, such as the ends of lines, one pixel wide lines, and corners, are removed by a median filter. Figure 1 shows several examples.

2.2. SD-ROM Filter

The signal-dependent rank order mean (SD-ROM) algorithm has been proposed as an alternative to the median filter. The filter has been applied in several applications, such as removing impulses,[4] streaks,[9] and scratches from images and impulse noise from audio signals.[10]

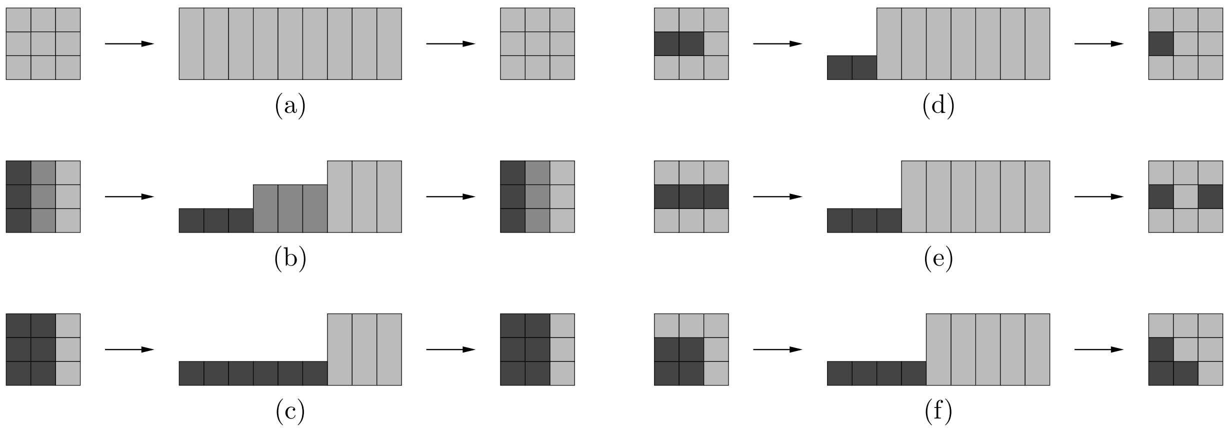


Figure 1. Median filtering examples. Examples (a) through (c) demonstrate root signals. Examples (d) through (f) show structures that are removed by the median filter. In each diagram, the left square is the input, the middle row represents the sorted values, and the right square is the output.

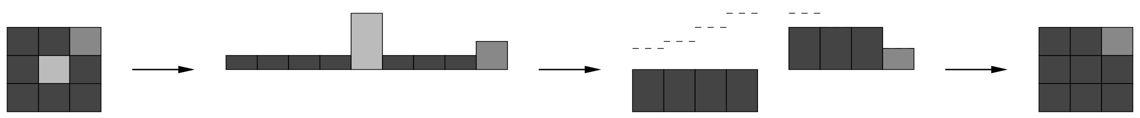


Figure 2. Graphical SD-ROM example. The left square represents the input image. The next row show the sorted surrounding values with the unsorted center value in the middle. The third section shows the result of the difference operation with respect to the thresholds. The right square is the result. The second threshold was exceeded, so the center pixel was replaced.

The SD-ROM algorithm works in two steps. First, a small neighborhood around a sample is used to determine if the central sample is corrupted or not. This step is called detection or state-determination. The two-state algorithm chooses between an uncorrupted state and a corrupted state. If the central sample is corrupted, a new value is estimated for the central sample using the surrounding window. If the central sample is uncorrupted, the sample is not changed.

Although the surrounding window can be of arbitrary size and shape, in practice a three by three window is used in most image applications. The center pixel within the window is designated $x(\mathbf{n})$. The surrounding eight values are labeled $x_1(\mathbf{n}) \ldots x_8(\mathbf{n})$ where $x_1(\mathbf{n})$ is the upper left value in the window and $x_8(\mathbf{n})$ is the lower right value. The remaining values are labeled by scanning across rows in the window, skipping the center value. The algorithm can then be summarized as follows:

1. Order pixel values in surrounding window from smallest to biggest:

$$x_1(\mathbf{n}), x_2(\mathbf{n}), \ldots, x_8(\mathbf{n}) \rightarrow r_1(\mathbf{n}) \leq r_2(\mathbf{n}) \leq \ldots \leq r_8(\mathbf{n}) \tag{1}$$

2. Compute rank-order differences, $i = 1, \ldots, 8$:

$$d_i(\mathbf{n}) = \begin{cases} r_i(\mathbf{n}) - x(\mathbf{n}), & i = 1, \ldots, 4 \\ x(\mathbf{n}) - r_i(\mathbf{n}), & i = 5, \ldots, 8, \end{cases} \tag{2}$$

3. Threshold and replace, if necessary:

$$y(\mathbf{n}) \equiv \begin{cases} m(\mathbf{n}), & d_i(\mathbf{n}) > T_i \text{ or } d_{9-i}(\mathbf{n}) > T_i, \quad i = 1, \ldots, 4, \\ x(\mathbf{n}), & \text{otherwise.} \end{cases} \tag{3}$$

The ordered window values are labeled $r_1(\mathbf{n}) \ldots r_8(\mathbf{n})$. The *rank ordered differences* are labeled $d_1(\mathbf{n}) \ldots d_8(\mathbf{n})$. $m(\mathbf{n})$ is the *rank order mean* and is the average of $r_4(\mathbf{n})$ and $r_5(\mathbf{n})$. This value replaces the center pixel if any of the thresholds $T_1 \ldots T_4$ are exceeded. Because the difference calculation results in a signed difference and not an absolute difference, the thresholds are restricted to values greater than or equal to zero.

Fig. 2 graphically illustrates the basic steps in the SD-ROM algorithm and provides a simple example.

2.3. Noise Model

Although both the median and SD-ROM filters can effectively remove many types of corruption from images, in this paper we only consider a particular kind of impulse noise. Impulse noise is defined as

$$x(\mathbf{n}) = \begin{cases} v(\mathbf{n}), & \text{with probability } 1 - p \\ \eta(\mathbf{n}), & \text{with probability } p, \end{cases} \tag{4}$$

where $v(\mathbf{n})$ is the original image value, $\eta(\mathbf{n})$ is a sample of an identically distributed, independent random process with a uniform probability density function, and p is the probability of corruption.

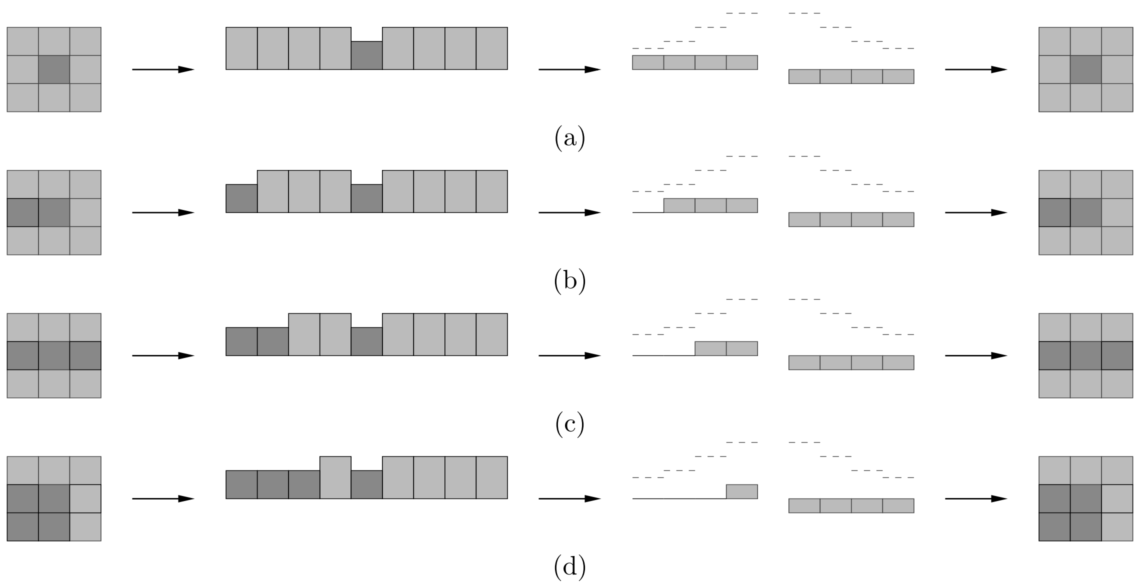


Figure 3. SD-ROM root examples. A median filter would replace the center in each of these examples. However, as no threshold is exceeded, the SD-ROM filter does not alter the signal.

3. SD-ROM ROOT SIGNALS

As discussed in Section 2.1, the median filter passes certain signals without change. These signals are called root signals. A root signal analysis is one method of characterizing a filter. Both the median filter and the SD-ROM filter are nonlinear algorithms. Unlike linear filters, they cannot be represented in the frequency domain in terms of passbands or stopbands. The best approximation one can make to a passband characterization is to flesh out as fully as possible the set of signals that are unchanged by a filter.

Similar to the median filter, the two-state SD-ROM filter has a set of root signals. In fact, any signal that is a root signal for a median filter is also a root signal for any SD-ROM filter. In a median filter, each sample is replaced with the median of the sample window. Therefore, every sample in a root signal is already the median of the sample window values. In a two-state SD-ROM filter, the center value is subtracted from the ordered surrounding values in accordance with Eq. (2) and compared to the thresholds. If the center value is the median, all of the computed differences will be less than or equal to zero. By design, the SD-ROM thresholds are non-negative. Therefore the center sample will never be detected as an impulse and will never be replaced. Thus any median root signal is also a root signal for the two-state SD-ROM.

However, the SD-ROM algorithm also has other root signals. These signals depend on the thresholds selected for the filter. Fig. 3 provides examples of cases where the SD-ROM filter will pass impulses, the ends of lines, one pixel wide lines, and corners. All of these structures are removed by a median filter. In the SD-ROM filter, each structure is keyed to one of the four thresholds. An impulse will be passed if the absolute difference between the impulse and the background is no bigger than T_1. Similarly, the end of a line will be passed if the difference from background is not bigger than T_2, a line will be passed if the difference from background is not bigger than T_3, and a corner will be passed if the difference from background is not bigger than T_4.

Because $T_4 \geq T_3 \geq T_2 \geq T_1$, the SD-ROM filter will pass more corners than lines, more lines than ends, and more ends than impulses. This trait makes the filter a good candidate for removing impulses while still preserving image details. The higher the thresholds, the more detail that will be preserved.

However, the SD-ROM algorithm is order-based. It does not consider any kind of structural information. Therefore, as far as the filter is concerned, the end of a line is identical to a center impulse and an impulse in the same direction in the surrounding window. As the probability of corruption increases, the probability of multiple impulses within a window increases. To remove these impulses, the thresholds must be decreased.

Thus, as usual, the selection of the SD-ROM thresholds involves a balance between the number of impulses that are detected and the number of image details that are removed. To properly design the filter, a quality criterion must be selected. In the original work,[4] the thresholds were chosen on the basis of minimizing the mean square error (MSE) and maximizing subjective quality for simulated results. In the following section, an approach will be outlined for maximizing the probability that impulses will be correctly detected.

4. PROBABILISTIC MODEL

Another way to characterize the performance of a filter is through a statistical model. In this methodology, probability distributions are assumed for the input signal and noise sources and a distribution for the output signal is derived. For a median filter, the output distribution depends only on the size of the window chosen and the distributions of the input and noise signals. For the SD-ROM filter, the output distribution also depends on the thresholds chosen for a particular application. In that way, a probabilistic model may help illustrate the influence of the thresholds on the performance of the SD-ROM algorithm.

The output distribution for a median filter can be found in the existing literature.[11] It is simply the probability that a given value will be at the center of an ordered set of values. More generally, the probability that the n^{th} order statistic for a set of numbers will be less than or equal to x is

$$\Psi_{(n)}(x) = \sum_{i=0}^{N-n} \left(\begin{array}{c} N \\ i \end{array} \right) (1-\Phi(x))^i \Phi(x)^{N-i}\,, \tag{5}$$

where N is the number of samples of input distribution function $\Phi(t)$ and n is the desired order statistic.[12] For a three by three median filter, N equals nine and n equals five.

The probability distribution function in Eq. (5) assumes that the input distribution is continuous. In our application, the input function is discrete. Only valid grayscale values are allowed. Therefore, the distribution function must specifically account for the probability two samples may be equal. In addition, we will eventually be interested in the distribution of the difference between each order statistic and the central sample. To that end, the probability that a given order statistic equals a value x will be useful. The expression for that probability is

$$f_{(n)}(x) = \sum_{i=1}^{N} \sum_{k=max(0,n-i)}^{min(n-1,N-i)} \left(\begin{array}{c} N \\ i \end{array} \right) \left(\begin{array}{c} N-i \\ k \end{array} \right) \mathrm{P}_e^i \mathrm{P}_l^k \mathrm{P}_g^{N-i-k}\,, \tag{6}$$

where P_e is the probability of the value x, P_l is the probability of values less than x, and P_g is the probability of values greater than x. In the summations, the index i is the number of samples equal to x and the index k is the number of samples less than x.

Eq. (6) calculates the probability density function for a single input distribution. However, the input to the SD-ROM filter actually consists of two distribution functions, one for the original image and one for the impulse noise. Given the number of impulses, N_i, a new density function can be derived

$$f_{(n)}(x|N_i) = \sum_{i=1}^{N} \sum_{j=\alpha}^{A} \sum_{k=\beta}^{B} \sum_{l=\gamma}^{\Gamma} \left(\begin{array}{c} N_i \\ j\,,l \end{array} \right) \left(\begin{array}{c} N_b \\ (i-j)\,,(k-l) \end{array} \right) \mathrm{P}_{ie}^{j} \mathrm{P}_{be}^{(i-j)} \mathrm{P}_{il}^{l} \mathrm{P}_{bl}^{(k-l)} \mathrm{P}_{ig}^{(N_i-j-l)} \mathrm{P}_{bg}^{(N_b-(i-j)-(k-l))}\,, \tag{7}$$

where

$$\begin{array}{rcllrcl} \alpha &=& \max(0, i-N_b), & & A &=& \min(i, N_i), \\ \beta &=& \max(0, n-i), & & B &=& \min(n-1, N-i), \\ \gamma &=& \max(0, k-N_b+i-j), & & \Gamma &=& \min(k, N_i-j), \text{ and} \\ N_b &=& N-N_i. & & & & \end{array}$$

As in Eq. (6), the probabilities that an impulse or background pixel value will be equal to, less than, or greater than the value of x are required. For the impulse noise distribution, these probabilities are P_{ie}, P_{il}, and P_{ig}, respectively. For the original image or backgound, these probabilities are P_{be}, P_{bl}, and P_{bg}. The indices i and k have the same meanings as in Eq. (6). The index j is the number of impulses that equal x and the index l is the number of impulses less than x.

The SD-ROM algorithm thresholds the difference between the sorted values from the surrounding window and the center pixel. The center pixel is either an impulse or part of the image. Thus, we need the probability that the difference between two random distributions will exceed a certain value. Using the probability density function (pdf) in Eq. (7) and the probability mass function (PMF) of the center pixel, we can derive the probability that the difference will be less than or equal to a certain threshold value. Two such distributions are useful,

$$F_{c,(n)}(z|N_i) = \sum_{x=0}^{255} F_i(x+z) f_{(n)}(x|N_i), \tag{8}$$

and

$$F_{u,(n)}(z|N_i) = \sum_{x=0}^{255} F_b(x+z) f_{(n)}(x|N_i). \tag{9}$$

where $F_i(x-z)$ is the PMF of the impulse noise and $F_b(x-z)$ is the PMF of the background image. $F_{c,(n)}(z|N_i)$ is the PMF of the difference for order statistic n given that the center pixel is an impulse and the surrounding window contains N_i impulses. $F_{u,(n)}(z|N_i)$ is similar, but the center pixel is uncorrupted. The range of the summation is equal to the assumed range of the values, in this case 8-bit grayscale values. The range is only included here for convenience.

For the basic three by three window version of the SD-ROM filter, there are four thresholds. Each threshold applies to two differences. For example, the first threshold, T_1, is applied to the difference $r_1 - x$ and also to the difference $x - r_8$. If either difference exceeds T_1, the center pixel is considered corrupted. Therefore, we need to calculate two probabilities for each threshold, as follows

$$\text{Prob}(r_i - x > T_i) \quad = \quad F_{c,(i)}((-T_i - 1)|N_i), \tag{10}$$

and

$$\text{Prob}(x - r_{9-i} > T_i) \quad = \quad 1 - F_{c,(9-i)}(T_i|N_i). \tag{11}$$

Because the thresholds are restricted to being non-negative, the probability of exceeding both thresholds at the same time is zero. Therefore, the results of Eqs. (10) and (11) can simply be added. In addition, the probability that a certain number of impulses exists in each window, $P_c(j)$, can be easily calculated using the probability of corruption, p. By summing over the number of impulses in the window, we get the total probability that a single threshold is exceeded given that the center pixel is corrupted

$$\text{Prob}_{\text{detect}}(T_i|\text{corrupted}) = \sum_{j=0}^{8} P_c(j)[F_{c,(i)}((-T_i - 1)|j) + 1 - F_{c,(9-i)}(T_i|j)] \tag{12}$$

Determining the probability that a corrupted pixel will be detected is not enough to optimize the thresholds. As discussed in Section 3, one can maximize detection by simply setting all the thresholds to zero. However, many of the features of the image will also be removed. Therefore we also need the probability that a threshold will *not* be exceeded when the center value is uncorrupted. Using the PMF from Eq. (9), an expression similar to Eq. (12) is derived,

$$\text{Prob}_{\text{pass}}(T_i|\text{uncorrupted}) = \sum_{j=0}^{8} P_c(j)[F_{u,(9-i)}(T_i|j) - F_{u,i}((-T_i - 1)|j)]. \tag{13}$$

Finally, by combining the probability of detecting corrupted pixels with the probability of passing uncorrupted pixels, the total probability of correct operation is found,

$$\text{Prob}_{\text{correct}}(T_i) = p * \text{Prob}_{\text{detect}}(T_i|\text{corrupt}) + (1-p) * \text{Prob}_{\text{pass}}(T_i|\text{uncorrupt}). \tag{14}$$

Using Eq. (14), the uniform impulse noise model, and an assumed image model, we can calculate the probability of correct performance for each threshold, $T_1 \ldots T_4$, for all possible values, $0 \ldots 255$. Figure 4 shows an example input distribution function. The example consists of two equally weighted Gaussian functions, one with a mean of 80 and

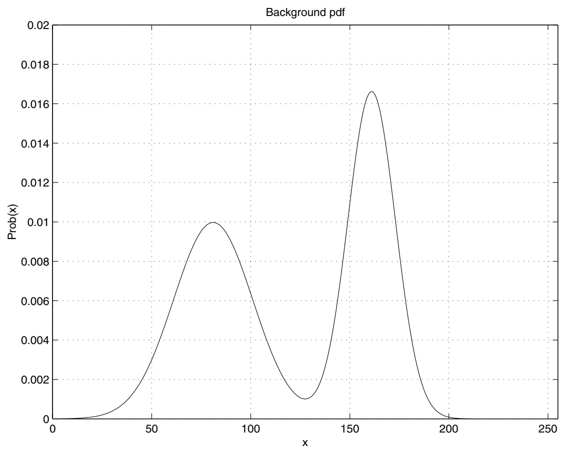


Figure 4. Background probability density function. The curve was created by combining two Gaussian pdfs ($\mu = 80$,$\sigma = 20$) and ($\mu = 160$,$\sigma = 12$).

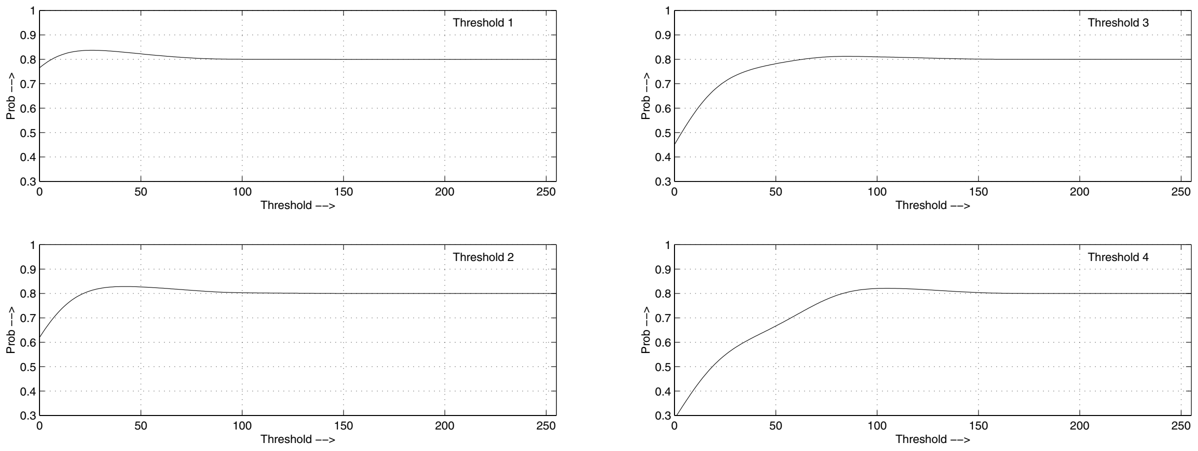


Figure 5. Probability of 'correct' detection versus threshold value for each of the four SD-ROM thresholds.

a standard deviation of 20 and the other with a mean of 160 and a standard deviation of 12. This probability density function was chosen to illustrate certain features in the threshold performance curves.

Figure 5 shows the threshold curves for the input distribution and a corruption probability of twenty percent. The curves have several interesting features. For all four thresholds, the performance converges to 80% correct as the thresholds increase. This trend is expected, as increasing thresholds decrease impulse detections until the corrupted image is unchanged. The curves also show that choosing low thresholds severely reduces the probability of correct detection, especially for the higher order thresholds. Setting T_1 to zero simply replaces the center value if it is outside the most extreme values in the surrounding window. Setting T_4 to zero forces the filter to replace the center pixel unless it is between the innermost window values.

Figure 5 also demonstrates the tendency for the optimal thresholds to be grouped higher for the inner thresholds, like T_4, than for the external thresholds, like T_1. This facts reflects the low probability of large numbers of impulses in the surrounding window. The width of the peaks is also greater for the internal thresholds. This effect becomes more pronounced as the level of corruption is increased. For very high error rates, the external threshold curves become very peaked and the internal thresholds start to show some peaking. However, at corruption rates less than 40%, the performance of the filter is not very sensitive to the inner thresholds, as long as they are large.

Fig. 5 also indicates that the filter does not provide much improvement over the corrupted image. Even with the optimum thresholds, the probability of correct detection only increases slightly over 80%. However, as we shall see in the next section, the actual performance of the SD-ROM algorithm is much better.

The maximum probability of correct detection for this input distribution occurs when $T_1 = 26$, $T_2 = 42$, $T_3 = 85$, and $T_4 = 105$. Choosing these values will ensure that each individual threshold performs optimally. Unfortunately, optimum performance of each individual threshold does not guarantee optimum performance of the overall filter. Because the center value is rejected if any threshold is exceeded, the thresholds are not independent. To some extent, the effect of lowering a threshold can be offset by raising the other thresholds. However, these interaction effects are not considered in this approach. To evaluate the overall filter performance requires the individual threshold probabilities to be combined into a single function of all four thresholds. This work has not yet been completed.

In addition, this approach only results in an estimate for the probability that the filter correctly estimates the state of the center pixel. Although it may seem like the best performance will result from the best estimation of the state, this is not in fact true. A filter which is maximally correct but preserves imperceptible details while passing large impulses will not have good perceptual quality. Ideally, the results of this analysis should be used to find the actual output distribution. With the output distribution, we can estimate the performance of the algorithm with respect to a distance metric, such as the mean square error.

The probabilistic approach to modeling a filter inherently depends on the model of the input signal. This is one of its strengths, as it applies to all specific inputs that match the assumed model. This is also one of its weaknesses, as most input models are of necessity simplified by making a large number of assumptions. Note that the model for the SD-ROM filter assumed that each individual sample is independent of its neighbors. While this is true of our noise source, by our definition, it is not true for images. Therefore, even with an overall model combining all thresholds and incorporating an error criterion, the final results may not match actual performance.

5. SIMULATION RESULTS

In contrast to the probabilistic approach, a simulation approach will find the optimum thresholds relative to a specific criterion, for a specific input image, and with a specific corruption pattern. By optimizing the thresholds for multiple images with varying levels of corruption, it may be possible to generalize the results and guide threshold design in new applications.

The SD-ROM algorithm operates on the sorted set of surrounding pixel values for each pixel in an image. The sorting operation is the most time consuming portion of the algorithm. A straightforward optimization approach would be to search through the sets of possible thresholds to find the set with the best performance. However, even for a three by three window and 8-bit grayscale image, the number of possible thresholds is very large. Although a single image with a single set of thresholds can be processed with little delay, simply processing the entire set of thresholds would take years.

To reduce the optimization time, two simplifications were made. First, the non-recursive implementation of the two-state SD-ROM algorithm was chosen. The SD-ROM algorithm thresholds eight differences calculated per Eq. (2). In the non-recursive implementation, these differences only depend on the input image and do not depend on previous filter results. Therefore, it is possible to pre-scan the image and calculate the differences at every location in the image. By storing the differences at each location as well as the penalties associated with both possible states, the performance for any set of thresholds can be calculated directly from the list.

This approach was taken in our optimization program. In addition, because the thresholds are restricted to being non-negative, all negative differences were set to zero. Then the list of differences was sorted and identical entries were combined. The end result was a greatly reduced list of differences. Frequently, lists with over 200,000 initial entries were reduced in size by one to two orders of magnitude, depending on the original image and the level of corruption. By using the difference lists, it was possible to evaluate the algorithm performance for thousands of threshold sets in less than an hour.

However, the number of possible threshold sets still made an exhaustive search prohibitive. The second simplification was an assumption that the optimum set of thresholds could be found in close proximity to other thresholds with near optimum performance. The assumed optimum thresholds were then found in two steps. First, initial thresholds were found by setting all four thresholds to their maximum values. Then T_1 was decreased until the best performance was achieved. Next, T_2 was reduced with T_1 held constant at its best value. Then, T_3 was scanned with

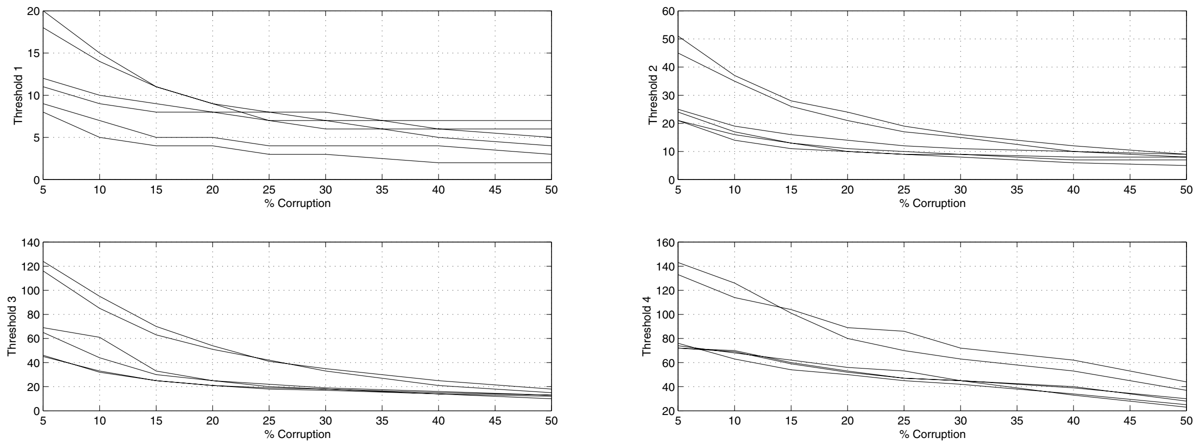


Figure 6. Simulation threshold results. The average of the optimum thresholds found at corruption percentages of 5, 10, 15, 20, 25, 30, 40, and 50 are plotted.

the others held constant and finally T_4 was scanned with all three other thresholds at their best values. The second step was to comprehensively search a range of thresholds around the starting thresholds. The range was allowed to change if better performance was found near the edge of the range of values searched.

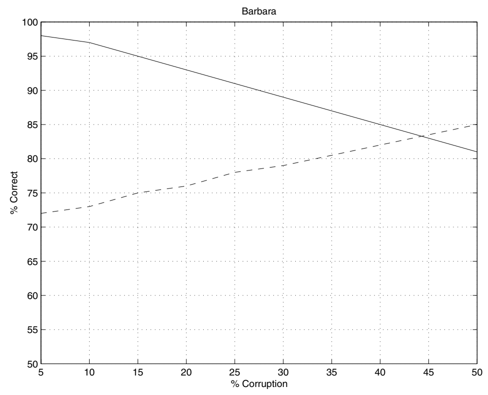
This approach seemed to work well. Although the initial thresholds obtained by setting each threshold independently were rarely the optimal thresholds, the performance of the optimal thresholds was insignificantly better than the performance of the original thresholds. A list of the twenty best sets of thresholds was kept for each image and specific instance of corruption. Examination of the lists did not reveal multiple pockets of optimum performance. Instead, a single best set of thresholds was found. The other sets of thresholds were minor variations of the best set. In general, the upper thresholds T_3 and T_4 varied over a wide range while T_1 and T_2 stayed relatively constant.

Figure 6 shows the simulation results for six images. All of the images were 512 by 512 pixel, 8-bit grayscale images. The images were *Goldhill*, *Lena*, *Barbara*, *Crowd*, *Pyramid*, and *Mill*. They were corrupted with 5, 10, 15, 20, 25, 30, 40, and 50 percent error rates. Five specific noise realizations were simulated for each image at each corruption percentage. The thresholds plotted in Fig. 6 are the averages of all the thresholds found (100 sets per image and corruption level). A much larger set of images has been tested. These images were selected because most of them are commonly available and they illustrate certain traits that are common for all of the images studied.

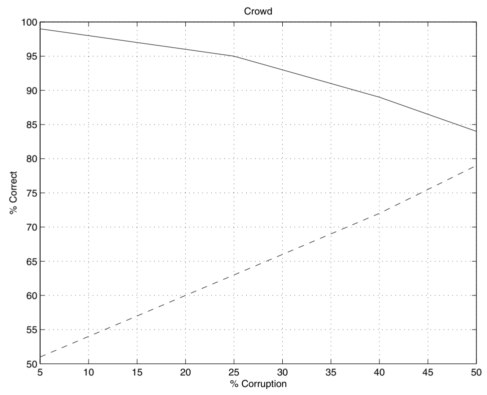
Figure 7 provides another interesting measure of the SD-ROM performance relative to the percentage corruption. The figure plots the percentage of correct detections versus the level of corruption for four of the images. The median filter results for the same corruption levels are also shown.

For most images, the thresholds decrease as the percentage of corruption increases. As the thresholds decrease, the filter becomes more median-like and removes outliers to a greater extent. However, even at 50% corruption, the thresholds are non-zero. When the error rate approaches fifty percent, the performance of the SD-ROM filter and the median filter are comparable. At low levels of corruption, the thresholds are highly image dependent. However, as corruption increases the thresholds begin to converge and become noise-driven as opposed to image-driven.

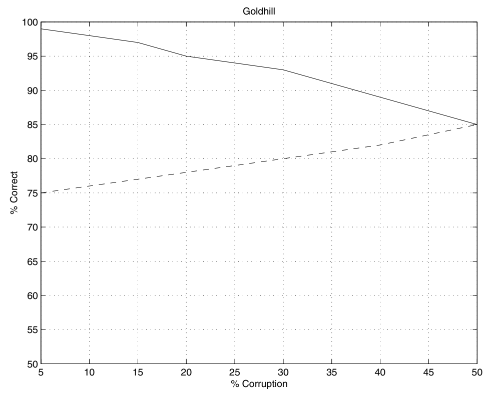
The plots in Fig. 6 show two groupings of image thresholds. The upper two curves in each plot correspond to the *Barbara* and *Mill* images. These images have strong edge and texture features. Preserving these features at low corruption levels drive the thresholds up higher than in the other images. The thresholds are applied to differences. In a typical grayscale image, there are very few areas where adjacent pixels vary by 100 levels or more. As a consequence, setting thresholds T_3 and T_4 to values above 100 effectively disable them. For low levels of corruption, the optimization program appears to push the inner thresholds up until they reach the point where they exceed the maximum local contrast in the original image. For some images with high contrast, like *Barbara* and *Mill*, this is much higher than other images.



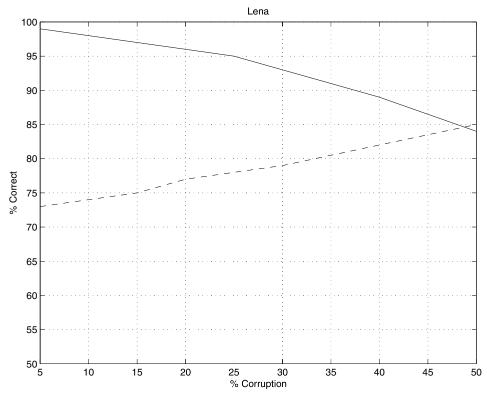
(a)



(b)



(c)



(d)

Figure 7. Simulation performance results. The percentage of the time that the state was correctly estimated for four images are shown. The solid lines are the SD-ROM results. The dashed lines are the median filter results. The images are (a) Barbara, (b) Crowd, (c) Goldhill, and (d) Lena.

Although it can not be seen from the plots, the importance of the thresholds also changes as the corruption level changes. For corruption levels less than approximately twenty percent, the third and fourth thresholds are not very tightly restricted. From the simulation results, it is apparent that the range of thresholds giving nearly optimal performance is very wide. The optimum values for these thresholds also vary significantly between noisy images with the same error rate but with different noise realizations. Because the initial thresholds give near optimum performance for low levels of corruption (within one percent of the best detection performance), the optimization is sensitive to the specific noise patterns, and the optimum ranges are very broad, it is probably best to simply use the values for the inner thresholds given by the initial search procedure.

Similar to the detection probability approach, this optimization procedure only optimizes the detection rate. It may let significant impulses pass while preserving perceptually insignificant image details. However, it is easy to generalize the search algorithm to use any criterion that depends only on the difference between the filter output and the original image. Another implementation of the optimization algorithm has been used to find the best thresholds in an MSE sense. These thresholds tend to agree with subjective evaluations much better than the pure detection thresholds. In fact, the MSE thresholds tend to overcorrect slightly. The best subjective results were obtained with thresholds between those predicted by the detection approach and those predicted by the MSE approach.

6. CONCLUSIONS

In this paper, three different approaches were taken to characterize the two-state SD-ROM filter in terms of its thresholds. None of the approaches provided a simple method for choosing thresholds given a test image or two. However, all three did provide some insights into the selection process. These include:

1. Thresholds should start high and decrease as corruption levels increase. Large thresholds preserve original image features. Low thresholds reject multiple impulses.
2. The likely number of impulses within each window is a good measure of the importance of each threshold. For low levels of corruption, thresholds T_3 and T_4 are not likely to see impulses. As long as they are set to a safely large value, they will not affect performance significantly. However, the values of thresholds T_1 and T_2 are fairly critical and should be set carefully. But because these thresholds are depend more on the noise characteristics, the optimal values are fairly constant between images.
3. Images with high contrast lines or textures require higher thresholds than other images.

If example images and a noise model are available, the approach taken in the simulation section is a good way to find initial thresholds. In fact, the initial thresholds themselves can be computed quickly and provide performance negligibly less than the optimum thresholds.

Although the probabilistic model is not yet practically useful, it can be improved by combining thresholds, calculating the output distribution, and optimizing the thresholds with respect to an error criterion. Future work will involve these modifications. With an output distribution, it should be possible to directly find the influence of the thresholds on the filter output. In addition, a comparison of the predicted thresholds with the simulation results for the same error criterion may results in a practical method for designing thresholds without a comprehensive search.

ACKNOWLEDGMENTS

This work was supported in part by a University of California MICRO grant with matching support from Optronix Engineering, Lucent Technologies, Raytheon Missile Systems, Xerox Corporation, and Tektronix Corporation. Part of this research was 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.
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. J. Tukey, “Nonlinear (nonsuperimposable) methods for smoothing data,” in *Congr. Res. EASCON*, p. 673, 1974.
6. J. Tukey, *Exploratory Data Analysis*, Addison-Wesley, Menlo Park, CA, 1977.
7. J. Fitch, E. Coyle, and J. N.C. Gallagher, “Root properties and convergence rates of median filters,” *IEEE Transactions on Acoustics, Speech, and Signal Processing* **33**, pp. 230–240, April 1985.
8. J. Astola, P. Haavisto, and Y. Neuvo, “Vector median filters,” *Proceedings of the IEEE* **78**, pp. 678–689, April 1990.
9. 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.
10. 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.
11. T. Nodes and J. N.C. Gallagher, “The output distribution of median-type filters,” *IEEE Transactions on Communications* **32**, pp. 532–541, May 1984.
12. J. Astola and Y. Neuvo, “An efficient tool for analyzing weighted median filters,” *IEEE Transactions on Circuits and Systems* **41**, pp. 487–489, July 1994.