3-1994

Rank Conditioned Rank Selection Filters for Signal Restoration

Russell C. Hardie
University of Dayton, rhardie1@udayton.edu

Kenneth E. Barner
University of Delaware

Rank Conditioned Rank Selection Filters for Signal Restoration

Russell C. Hardie † *and Kenneth E. Barner* ‡

† Department of Electrical Engineering
University of Dayton
300 College Park Avenue
Dayton, OH 45469
(513) 229-3611

‡ Applied Science and Engineering Laboratories
A. I. duPont Institute/University of Delaware
1600 Rockland Road, P.O. Box 269
Wilmington, DE 19899
(302) 651-6830

Abstract

A class of nonlinear filters called *rank conditioned rank selection* (RCRS) filters is developed and analyzed in this paper. The RCRS filters are developed within the general framework of *rank selection* (RS) filters, which are filters constrained to output an order statistic from the observation set. Many previously proposed rank order based filters can be formulated as RS filters. The only difference between such filters is in the information used in deciding which order statistic to output. The information used by RCRS filters is the ranks of selected input samples, hence the name rank conditioned rank selection filters. The number of input sample ranks used is referred to as the order of the RCRS filter. Low order filters can give good performance and are relatively simple to optimize and implement. If improved performance is demanded, the order can be increased but at the expense of filter simplicity. In this paper, many statistical and deterministic properties of the RCRS filters are presented. Also presented is a procedure for optimizing over the class of RCRS filters. Finally, extensive computer simulation results are presented which illustrate the performance of RCRS filters in comparison to other techniques in image restoration applications.

EDICS Number: 1.2
Permission to publish this abstract separately is granted.

Submitted to the IEEE Transactions on Image Processing, September 1992.

List of Figures

List of Tables

1 Introduction

A class of nonlinear filters, which we refer to as *rank conditioned rank selection* (RCRS) filters, is presented and analyzed in this paper. The RCRS filters are developed in the general framework of *rank selection* (RS) filters. RS filters are those filters constrained to output an order statistic from the set of input samples. Many rank order based filters which have been proposed can be cast into the RS filter framework. The difference between such filters is in the information used to select an order statistic to output. The information used by RCRS filters is the ranks of selected input samples, hence the name rank conditioned rank selection filters. The number of input sample ranks used in this decision is referred to as the order of the RCRS filter. The order can range from zero to the number of samples in the specified observation window. This control gives the filters valuable flexibility. Low order filters can give good performance and are relatively simple to optimize and implement. If improved performance is demanded the order can be increased, but, at the expense of filter simplicity. Simulation results presented in this paper show that the RCRS filters offer improved performance over many other previously proposed techniques. Thus, we believe that they represent a powerful and useful class of nonlinear filters.

Signal restoration and filtering have traditionally been approached from a linear framework. Linear methods, however, tend to be sensitive to heavy tailed noise. They also tend to be sensitive to non-stationarities, which are prevalent in signals such as images. Such shortcomings have spurred the development of nonlinear filters. One of the earliest nonlinear filters proposed is the median filter [20]. The median is well known for its ability to suppress impulsive type noise while preserving edges [3, 9, 13, 20]. For this reason the median filter is widely used in image processing applications. To gain improved performance, many generalizations of the median have been proposed. These include *multistage median* filters [1, 4, 17], *center weighted median* (CWM) filters [11, 12, 14, 19], general *weighted median* (WM) and *weighted order statistic* (WOS) filters [13, 22, 24], *stack* filters [7, 8, 15, 21, 25], and *permutation* filters [5, 6]. All of these filters can be formulated as RS filters. The filters differ, however, in the information they use to select an output order statistic. These more sophisticated RS filters tend to have better detail preserving characterists than the median.

The main advantages of the RS filtering approach over linear methods are: (1) RS filters tend to preserve edges well, and (2) the effect of outliers is minimized. Edge preservation results from the fact that RS filters always output one of the samples in the observation window. Thus, no new intermediate or transition points are introduced by the filtering process. This tends to keep edges sharp and crisp. The ability of the RS filters to limit the effects of outliers derives from the nature of rank ordered data. In heavy tailed noise, outliers tend to be located in the extreme ranks of the sorted data. By not selecting output samples from the extreme ranks, RS filters can give a robust estimate that is insensitive to even high levels of heavy tailed noise.

We show that the RCRS filters, which use the ranks of selected observation samples as the basis for selecting an output rank, have a number of very useful properties. Furthermore, extensive computer simulations reveal that the RCRS filters perform extremely well in comparison to other techniques in image restoration applications. In particular, they offer superior performance to the simple median and CWM filter, which are subclasses of RCRS filters. In addition, the RCRS filters outperform WOS filters and stack filters in some applications. Finally, optimizing and implementing low order RCRS filters is relatively simple.

This paper is organized as follows. In Section 2, we formally define RS filters and show how several previously proposed rank order based filters can be cast into this framework. We then define the RCRS filters in Section 3 and examine the relationship between them and other filter classes. Also, a procedure for optimizing over the class RCRS filters is described in Section 3. In Section 4, many statistical and deterministic properties of the RCRS filters are presented. Extensive computer simulation results are presented in Section 5. These results illustrate the performance of the RCRS filters in comparison to other techniques in image restoration applications. A thorough quantitative analysis is presented and several images are shown for subjective evaluation. Finally, some conclusions are drawn in Section 6.

2 Rank Selection Filters

In this section, the RS filter structure is defined and discussed. We also examine how several previously proposed rank order based filters can be formulated within this framework. By doing so, these different filtering methods can be better related.

Before the RS filter structure is presented, the notation used in this paper is defined. Consider the discrete sequences $\{d(\mathbf{n})\}$ and $\{x(\mathbf{n})\}$, representing the desired and corrupted versions of a signal respectively. The index $\mathbf{n}$ is a d element vector such that $\mathbf{n} = [n_1, n_2, \ldots, n_d]$, and both $\{d(\mathbf{n})\}$ and $\{x(\mathbf{n})\}$ are d–dimensional sequences. Also, consider a d–dimensional window function that spans N samples and passes over the corrupted sequence in some predetermined fashion. At each location $\mathbf{n}$, the N observation samples spanned by the window can be indexed and written as a vector, yielding

$$\mathbf{x}(\mathbf{n}) = [x_1(\mathbf{n}), x_2(\mathbf{n}), \ldots, x_N(\mathbf{n})]. \tag{1}$$

We define the vector $\mathbf{x}^r(\mathbf{n})$ to be the vector containing the N observation samples arranged in increasing order, such that

$$\mathbf{x}^r(\mathbf{n}) = [x_{(1)}(\mathbf{n}), x_{(2)}(\mathbf{n}), \ldots, x_{(N)}(\mathbf{n})], \tag{2}$$

where $x_{(1)}(\mathbf{n}) \leq x_{(2)}(\mathbf{n}) \leq \cdots \leq x_{(N)}(\mathbf{n})$. To relate the rank of a sample to its location within the window, we define $r_i(\mathbf{n})$ to be the rank of the sample in window location i, i.e., $x_i(\mathbf{n}) \equiv x_{(r_i(\mathbf{n}))}(\mathbf{n})$. Also, let $\mathbf{r}(\mathbf{n}) = [r_1(\mathbf{n}), r_2(\mathbf{n}), \ldots, r_N(\mathbf{n})]$.

From the set of observation samples, we wish to form an estimate of the desired sample at location δ within the window. This estimate is denoted as $\hat{d}_\delta(\mathbf{n})$, where $1 \leq \delta \leq N$. By definition, the output $\hat{d}_\delta(\mathbf{n})$ of an RS filter is constrained to be an order statistic from the observation vector. For notational simplicity, the index $\mathbf{n}$ is assumed, and used explicitly only when necessary for clarity. RS filters are formally defined as follows.

Definition 2.1 *The output of a window size N rank selection filter is given by*

$$F_{RS}(\mathbf{x}) = x_{(\mathcal{S}(\mathbf{z}))}, \tag{3}$$

where $\mathbf{z}$ is a feature vector that lies in the feature space Z, and $\mathcal{S} : Z \mapsto \{1, 2, \ldots, N\}$.

The decision as to which sample from $\mathbf{x}^r$ to take as the output is based on the feature vector $\mathbf{z}$. This feature vector represents a subset of the information contained in $\mathbf{x}$. In general, the full information contained in the observation vector $\mathbf{x}$ is not used as the basis for determining which ranked sample to output. Optimizing over a class of filters that utilizes all the information in $\mathbf{x}$ is impractical, if not impossible, even for small observation windows. Therefore, it is necessary to extract the information from $\mathbf{x}$ that is most relevant for the application at hand. If a feature space with low enough dimensionality is selected, the optimization becomes feasible.

A block diagram of the RS filter structure is shown in Fig. 1. The diagram shows the observation vector $\mathbf{x}$ being fed into two functional blocks. One block extracts the feature vector $\mathbf{z}$, and the other produces the sorted vector $\mathbf{x}^r$. The output rank selector chooses the appropriate sample from $\mathbf{x}^r$ to be the estimate. The choice as to which sample to output is based on the feature vector $\mathbf{z}$, and the rank selection rule $\mathcal{S}(\cdot)$.

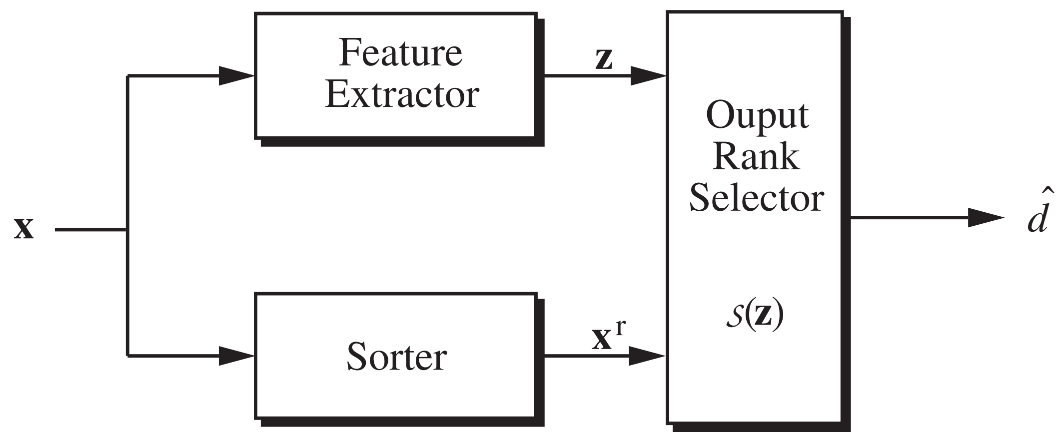


Figure 1: Block diagram showing the rank selection filter structure.

From the definition it is clear that $\mathcal{S}(\cdot)$ can be considered a classifier that partitions the feature space Z into N regions. Each region in the partitioned space corresponds to a specific rank being selected to be the output. Whenever the feature vector lies in i^{th} partition, the filter output is the i^{th} order statistic. Given a feature space Z, the function $\mathcal{S}(\cdot)$ can be found using a variety of traditional classification techniques. For certain feature spaces, optimization techniques can be derived that minimize the filter estimate error under specific quantitative distortion measures. The performance of RS filters depends, in large part, on the choice of feature space. Thus, the feature space must be appropriate for the job at hand.

Many previously proposed filters can be formulated as RS filters. The difference between these filters is in the choice of the feature space and classifier. Perhaps the simplest filter that can be described in the RS framework is a filter that outputs a single constant order statistic. In this case, the classifier is simply a constant and is not a function of the feature space Z. That is $\mathcal{S}(\cdot) = k$. Such a filter class includes the median filter. The median filter has been shown to be effective at suppressing heavy tailed noise while preserving edges [3, 9, 13, 20]. In many applications, however, the median removes signal structure. Moreover, the median offers little flexibility in the tradeoff between detail preservation and noise smoothing.

A second class of filters that can be formulated in the RS framework is the class of CWM filters. This class of filters allows for greater control in the tradeoff between detail preservation and noise smoothing. A similar class of filters, the LUM filters [10, 11, 12], includes all rank order and CWM filters as a subset. It is shown shortly that the feature vector for the CWM filter contains partial rank of the center sample in the observation window. While the CWM filter has been shown to perform well in image restoration applications [11, 12, 14], we show here that RCRS filters, which use complete rank information of selected observation samples, give a significant improvement in performance.

Stack filters can also be cast into the RS framework. This large class of filters contains all order statistic and weighted order statistic operators, as well as all compositions of such operators, as a subset [7]. For stack filters, the determination of which sample to output is based strictly on the level crossing information [7, 21]. The feature space for a size N stack filter can be interpreted as the set of $N \times N$ binary arrays in which each row and column contains, at minimum, a single one. In addition, the rows of the arrays in the feature space are constrained to obey the stacking property. Through an appropriate partitioning of the space consisting of such arrays, where the allowable partitions are governed by the stacking constraint [15], any stack filter can be realized as an RS filter. By operating on level crossing information only, stack filter estimates are robust and effective at smoothing heavy tailed noise. However, the feature space of the class of stack filters grows rapidly with the window size. This rapid growth makes large window size stack filters, e.g., $N > 21$, impractical. Thus, it is useful to explore other RS filters which can be implemented with larger window sizes.

As a final example, consider permutation filters [5, 6]. This class of filters naturally lends itself to the RS framework. For this class of filters, the feature vector is $\mathbf{r}$, the vector relating the rank order and temporal order of each sample in the window. By relating the rank and temporal order of each sample, the permutation filter feature vector allows for the design of highly specialized filters. This choice of feature vector has shown to be particularly effective in applications where frequency selection is required. The drawback of using $\mathbf{r}$ as the feature vector is the rapid growth of the feature space as a function of window size. The cardinality of the permutation filter feature space grows as $N!$, making the use of windows that contain more than nine samples currently impractical. The proposed class of RCRS filters is an effective link between simple order statistic filters (0^{th} order RCRS filter), and permutation filters (N^{th} order RCRS filter). By using a feature vector of lower dimension, the window size of RCRS filters can be increased beyond that of permutation filters. The focus of the presentation here is on the lower order RCRS filters. Through computer simulations we show that lower order RCRS filter have performance superior to many previously proposed filtering methods. While the simulations focus on the lower order cases, the properties of RCRS filters are derived under the general case.

3 Rank Conditioned Rank Selection Filters

In this section, the RCRS filters are defined. The remainder of this paper focuses on these filters. The relationship between RCRS filters and other filter classes is also examined in this section. Finally, optimization methods are discussed.

A Filter Definition

The feature vector for the RCRS filters consists of the the ranks of selected samples in the observation vector $\mathbf{x}$. The selected samples can be placed in a vector, yielding $\mathbf{x}^* = [x_{\gamma_1}, x_{\gamma_2}, \ldots, x_{\gamma_M}]$, where M is referred to as the order of the RCRS filter, and $0 \leq M \leq N$. The respective ranks of these samples comprise the feature vector, yielding $\mathbf{z} = \mathbf{r}^* = [r_{\gamma_1}, r_{\gamma_2}, \ldots, r_{\gamma_M}]$. The feature space is given by $Z = \Omega_M$, where $\Omega_M = \{\ [i_1, i_2, \ldots, i_M] : i_j \in \{1, 2, \ldots, N\}$ and $i_j \neq i_k \ \forall\ j \neq k\}$. Thus, the feature space contains all combinations of ranks excluding those in which any two are equal, since those combinations can not occur. The RCRS filters are defined specifically as follows.

Definition 3.1 *The output of an M^{th}–order RCRS filter with window size N is given by*

$$F_{RCRS}(\mathbf{x}) = x_{(\mathcal{S}(\mathbf{r}^*))}, \tag{4}$$

where $\mathbf{r}^* = [r_{\gamma_1}, r_{\gamma_2}, \ldots, r_{\gamma_M}]$, $0 \leq M \leq N$ *and* $\mathcal{S} : \Omega_M \mapsto \{1, 2, \ldots, N\}$.

The order of the filter and location of the samples chosen for $\mathbf{x}^*$ depend on the application. For $M = 1$, the filter operation is relatively simple to implement and optimize. We show that for many applications, this filter does a good job. If greater performance is demanded, then additional rank information can be added to the feature vector. The following theorem specifies the number of unique filters in the RCRS filter class.

Theorem 3.1: *The cardinality of the M^{th}–order RCRS filter class with window size N is*

$$N^{(N!/(N-M)!)}, \tag{5}$$

where $0 \leq M \leq N$.

Proof: First note that the cardinality of the feature space Ω_M, denoted as $|\Omega_M|$, is $|\Omega_M| = N(N-1)(N-2)\cdots(N-M+1) = N!/(N-M)!$. For each $\mathbf{r}^* \in \Omega_M$, the domain of $\mathcal{S}(\mathbf{r}^*)$ is $\{1, 2, \ldots, N\}$. Thus, for each $\mathbf{r}^* \in \Omega_M$, there are N distinct choices for $\mathcal{S}(\mathbf{r}^*)$. Consequently, the number of distinct filters is $N^{|\Omega_M|}$, which is equivalent to (5). □

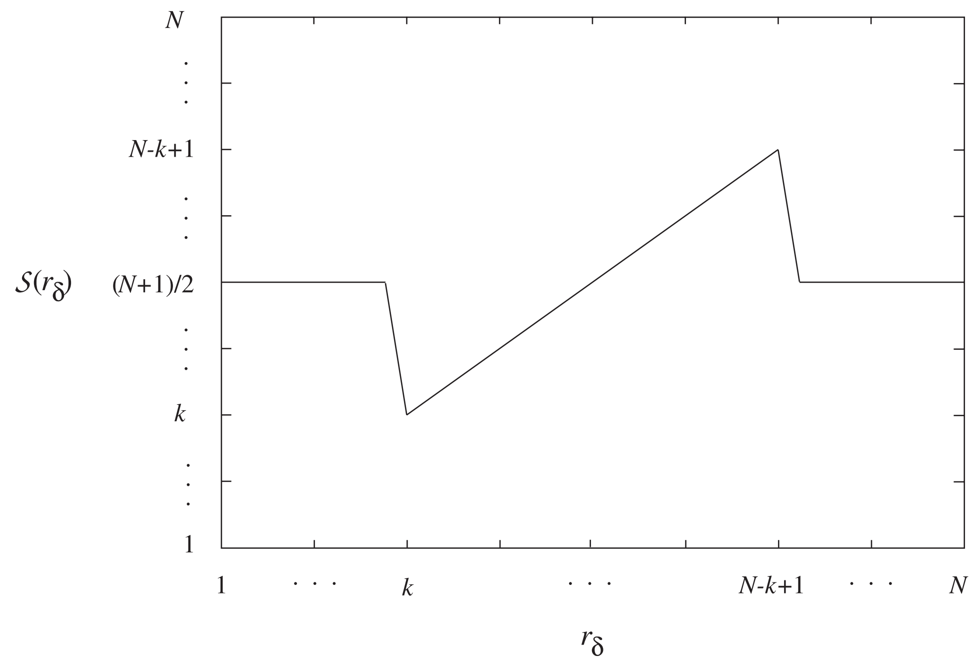


Figure 2: The function $\mathcal{S}(r_\delta)$ corresponding to the RCM filter with parameter k.

For $M = 1$ there are N^N RCRS filters. This number grows rapidly with the filter order. If $M = N$, the RCRS filters are equivalent to permutation filters, and the number of filters within the class grows to $N^{N!}$ [5, 6]. It is currently impractical to implement permutation filters with large window sizes. In this paper, we focus on the simpler RCRS filters where M is small compared to N.

Although it is not necessary, it is generally useful to include x_δ as an element in $\mathbf{x}^*$ since this is the sample at the location of the estimate. The input sample x_δ generally provides the most relevant information with which to form the estimate $\hat{d}_\delta$. Let us consider an example of an RCRS filter where $M = 1$ and $\mathbf{x}^* = x_\delta$. The feature vector is then $\mathbf{r}^* = r_\delta$ and the filter is characterized by the function $\mathcal{S}(r_\delta)$. Figure 2 shows an example of one such function. This particular filter outputs the sample x_δ if $k \leq r_\delta \leq N - k + 1$. Otherwise the filter outputs the median. Thus, we refer to this filter as a *rank conditioned median* (RCM) filter.

Plotting the function $\mathcal{S}(\mathbf{r}^*)$ for $M = 1$ and $M = 2$ can be a powerful aid in analyzing the operation of RCRS filters. For example, from Fig. 2 it is clear that the RCM filter is effective at suppressing heavy tailed noise. If x_δ lies in the middle ranks, it is unaltered. However if it lies in the extreme ranks, the RCM filter outputs the median. Finding the optimal $\mathcal{S}(\cdot)$ under a specified quantitative error measure is discussed later in this section.

B Relationship Between RCRS Filters and Other Filter Classes

Let us now examine the relationship between RCRS filters and several other rank order based filters. First, we consider their relationship to the CWM filter. The output of the CWM filter is defined to be the median over an extended set containing multiple center samples. This operation can be written as

$$F_{CWM}(\mathbf{x}) = median\{x_1, \ldots, w_\delta \diamond x_\delta, \ldots, x_N\}, \tag{6}$$

where $\diamond$ is a replication operator, x_δ represents the center sample in the window and N is assumed to be odd. The center sample is repeated w_δ times, where w_δ is non–zero odd positive integer. When $w_\delta = 1$, the operator is a median filter, and for $w_\delta \geq N$, the CWM reduces to an identity operation. It has been shown in [14] that the CWM filter operation is equivalent to

$$F_{CWM}(\mathbf{x}) = median\{x_{(k)}, x_\delta, x_{(N-}$$

Simply weighted order statistic filters are a subclass of RCRS filters and are characterized by the function

$$\begin{cases} & if \\ & if \\ & if \end{cases}$$

where .

δ

$\square$

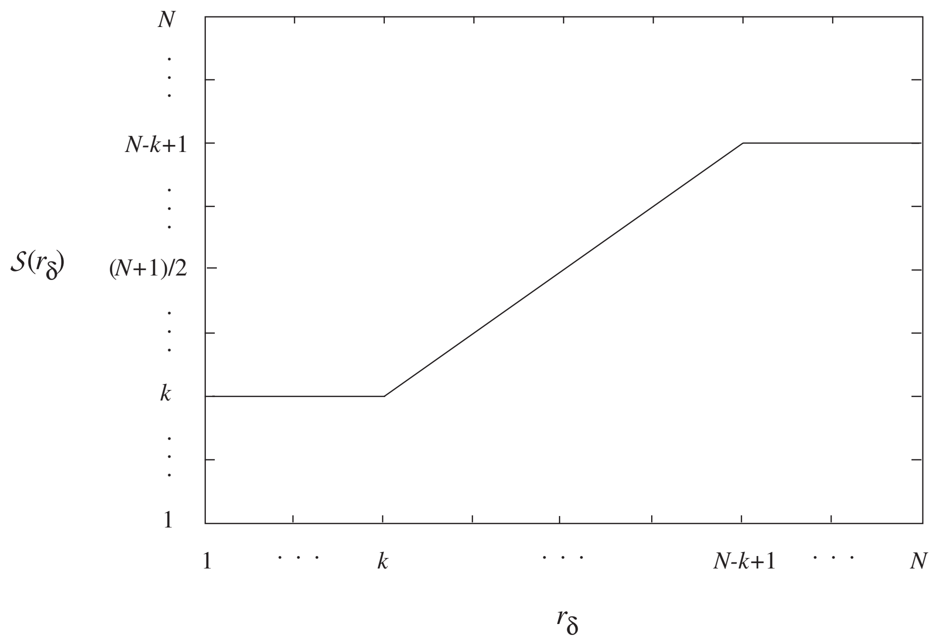


Figure 3: The function $\mathcal{S}(r_\delta)$ corresponding to the CWM filter with parameter k.

From Theorem 3.2 it is clear that SWOS filters use partial rank information about x_δ. The output rank is determined by which rank range x_δ lies in. The performance of the filter can be improved by utilizing the full rank information contained in r_δ. The function $\mathcal{S}(r_\delta)$ corresponding to the CWM filter with parameter k is shown in Fig. 3. Like the RCM filter, the CWM does not alter x_δ if $k \leq r_\delta \leq N-k+1$. If $r_\delta < k$, the output of the CWM is $x_{(k)}$. Similarly, if $N-k+1 \leq r_\delta$, the output is $x_{(N-k+1)}$.

The class of weighted order statistic (WOS) filters is defined by relaxing the SWOS constraint that only a single sample be weighted. The output of a WOS filter is given by

$$F_{WOS}(\mathbf{x}) = rank(v)\{\mathbf{w} \diamond \mathbf{x}\}, \tag{11}$$

where $\mathbf{w}$ is a N element vector of weights [24]. The i^{th} element in this vector, w_i, is the weight applied to the sample x_i. As in the SWOS filter, the output is the v^{th} ranked element in the expanded set. By weighting each sample in the window, WOS filters can emphasize certain observation samples while deemphasizing others. Although this weighting scheme offers flexibility in filter design, and some overlap exists between the WOS and RCRS filter classes, many of the most interesting RCRS filters can not be realized as WOS filters. For instance, the RCM filter can not be realized as a WOS filter. This is proved in the following discussion on stack filters, which contain WOS filters as a subset. But first, since RCRS and WOS filters are compared through computer simulations in Section 5, the number of operations required to form an estimate for each filter type is given.

Both RCRS and WOS filters require the observation data to be rank ordered. In addition to the operations required to rank the data, a WOS filter performs, on average, $N/2$ additions and comparisons to form an estimate. The additional operations required to form a RCRS filter estimate depend on the order of the filter, not the window size. For an order M RCRS filter, an additional $M-1$ multiplies and $M(M-1)/2$ comparisons are required. These operations are necessary, in the RCRS filter case, to generate the appropriate index in the $N!/(N-M)!$ entry look–up table that stores $\mathcal{S}(\cdot)$. Thus, low order ($M = 1, 2$) RCRS filters are, in general, simpler to implement than WOS filters. As the filter order increases, however, the number of operations required to form an RCRS filter estimate grows beyond that of the WOS filter, and the number of entries in the look–up table can become prohibitively large.

The next class of filters we consider is the class of stack filters. While stack filters are a large class of filters containing many other rank order based filters as subclasses, they do not contain RCRS filters as a subclass. Stack filters, however, are contained in the order N RCRS filters (which are equivalent to permutation filters) [5, 6]. The following property specifies filters common to both stack filters and order one RCRS filters.

Theorem 3.3: *Any stack filter $F_S(\cdot)$ that can be expressed as an order one RCRS filter is of the form*

$$F_S(\mathbf{x}) = median\left\{x_{(k)}, x_\delta, x_{(l)}\right\}, \tag{12}$$

where $1 \leq k \leq l \leq N$, or

$$F_S(\mathbf{x}) = x'_{(j)} \tag{13}$$

where $1 \leq j \leq N-1$. The order statistic $x'_{(j)}$ is the j^{th} ranked sample from the vector $\mathbf{x}'$, which contains all the samples in $\mathbf{x}$ excluding x_δ.

Proof: Any stack filter $F_S(\cdot)$ is uniquely defined by a positive Boolean function [21]. This unique positive Boolean function can be expressed in a sum of products form as

$$f(\cdot) = \sum_{i=1}^{m} \pi_i, \tag{14}$$

where each π_i is a product term $\pi_i = x_{i1}x_{i2}\cdots x_{im_i}$. The sum of products expression can be split into two sums, one over the π_i terms containing x_δ, and the other over the remaining product terms,

$$\begin{aligned} f(\cdot) &= \sum_{i:x_\delta \notin \pi_i} \pi_i + \sum_{i:x_\delta \in \pi_i} \pi_i && (15)\\ &= \sum_{i:x_\delta \notin \pi_i} \pi_i + x_\delta \sum_{i:x_\delta \in \pi_i} \pi'_i, && (16)\end{aligned}$$

where the π_i''s in (16) indicate that x_δ has been factored out. An order one RCRS filter makes no distinction between temporal locations of samples other than that indexed by δ. That is, all temporal locations other than δ are considered equivalent (permuting them has no effect on the output). Since the product terms in the summations in (16) do not contain x_δ, they are not functions of temporal location. Thus, the two sum of products must realize rank order operations over the $N-1$ samples x_i $i \neq \delta$. Let $\mathbf{x}'$ be a vector containing the $N-1$ observation samples from $\mathbf{x}$ excluding x_δ, and take $x'_{(k-1)}$ and $x'_{(l)}$ to be the order statistics defined by $\sum_{i:x_\delta \notin \pi_i} \pi_i$ and $\sum_{i:x_\delta \in \pi_i} \pi_i'$ respectively. The function realized by the stack filter can now be written as

$$F_S(\mathbf{x}) = max\left\{x'_{(k-1)}, min\{x_\delta, x'_{(l)}\}\right\}. \tag{17}$$

Consider first the case $k > l$. Then $F_S(\mathbf{x}) = x'_{(k-1)}$, which is independent of x_δ, and equal to (13) for $j = k-1$. Now consider the case $l \leq k$. An examination of the three possibilities $x_\delta < x'_{(k-1)} \leq x'_{(l)}$, $x'_{(k-1)} \leq x_\delta \leq x'_{(l)}$, and $x'_{(k-1)} \leq x'_{(l)} < x_\delta$, shows that in terms of the order statistics from $\mathbf{x}$, $F_S(\mathbf{x}) = max\left\{x_{(k)}, min\{x_\delta, x_{(l)}\}\right\}$, which is equivalent to (12). □

The order one RCRS function corresponding to (12) is that of an SWOS filter, and is given by (10). The order one RCRS function corresponding to (13) is given by

$$\mathcal{S}(r_\delta) = \begin{cases} j & \text{if } j \leq r_\delta \\ j-1 & \text{if } j > r_\delta \end{cases}. \tag{18}$$

Thus, the only order one RCRS filters that can be described as stack filter are SWOS filters, and those defined by (18). As is demonstrated shortly, these are not optimal RCRS filters in many cases. The following example illustrates the fact that RCRS filters (order one or greater) are not a subset of stack filters. The particular RCRS filter used in this example is the RCM filter, which is not a SWOS filter nor described by (18).

Example 3.1: Consider the window size 5 RCM filter with $k = 2$. This filter outputs the median observation sample if the center sample is either the minimum or maximum sample in the window. In all other cases the output is the center sample. To illustrate that this is not a stack filter, let $\mathbf{x} = (2, 3, 1, 4, 5)$. The output of the RCM filter operating on $\mathbf{x}$ is 3 since the center sample is the minimum sample in the observed set. Using threshold decomposition and stable sorting to find the filter output at each threshold level and then adding the results yields:

$$\begin{array}{ccccc|c} 2 & 3 & 1 & 4 & 5 & 2 \neq 3 \\ \hline \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \uparrow \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{array}, \tag{19}$$

which produces a result different than that obtained by operating on the multi–level data. Thus, the RCM filter does not possess the threshold decomposition property and consequently is not contained in the class of stack filters. This result, of course, is predicted by the previous theorem. □

C Optimization

Optimization over the class of RCRS filters is now addressed. The procedure described here closely follows that described in [5, 6]. There, optimization under the mean absolute error (MAE) and the least L_η normed error (LNE) were detailed for the permutation filter. While both the MAE and LNE methods can be modified to perform the optimization over the class of RCRS filters, we detail only the deterministic LNE method here.

In order to implement LNE optimization method, the feature vectors comprising the feature space must be indexed. By doing so, the feature space can be expressed as

$$\Omega_M = \{\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_{|\Omega_M|}\}. \tag{20}$$

In the foregoing development it is useful to write the observation vectors as sequence, indexed in the order that they are utilized. Also, let the indexed sequences exclude all partial observation vectors resulting from border effects. In this fashion, the observation vectors can be written as $\mathbf{x}(\mathbf{n}_1), \mathbf{x}(\mathbf{n}_2), \ldots, \mathbf{x}(\mathbf{n}_K)$, and the corresponding desired estimates as $d(\mathbf{n}_1), d(\mathbf{n}_2), \ldots, d(\mathbf{n}_K)$. For the RCRS filter defined by $\mathcal{S}(\cdot)$, the LNE over the K element training sequence is

$$\sum_{i=1}^{K} \mid d(\mathbf{n}_i) - F_{RCRS}(\mathbf{x}(\mathbf{n}_i)) \mid^{\eta} = \sum_{i=1}^{K} \mid d(\mathbf{n}_i) - x_{(\mathcal{S}(\mathbf{r}^*(\mathbf{n}_i)))} \mid^{\eta}. \tag{21}$$

The classifier[1] that minimizes (21) is referred to as the optimal classifier and is denoted as $\mathcal{S}_{opt}(\cdot)$.

[1] In instances where more than one classifier satisfies the optimality criteria, a tie breaking rule must be employed to define a single optimal classifier.

The LNE in (21) can be partitioned according to the observation feature vectors. Let α_i be the index of the feature vector in Ω_M corresponding to observation vector $\mathbf{x}(\mathbf{n}_i)$, such that $\mathbf{r}_{\alpha_i} = \mathbf{r}^*(\mathbf{n}_i)$, and define $\Gamma_{j,K} = \{i \in \{1, 2, \ldots, K\} : \alpha_i = j\}$. The total LNE incurred over the training sequence by estimating the desired signal with the k^{th} order statistic, given that the feature vector $\mathbf{r}_j$ is observed, can be written as

$$\mathcal{E}_j(k) = \sum_{i \in \Gamma_{j,K}} \mid d(\mathbf{n}_i) - x_{(k)}(\mathbf{n}_i) \mid^{\eta}. \tag{22}$$

If for some $j \in \{1, 2, \ldots, |\Omega_M|\}$ $\Gamma_{j,K} = \emptyset$, then define $\mathcal{E}_j(k) = 0$ for $k = 1, 2, \ldots, N$. The LNE of the RCRS filter defined by $\mathcal{S}(\cdot)$ can now be written as a sum of errors, partitioned according to feature vector, yielding

$$\sum_{i=1}^{K} \mid d(\mathbf{n}_i) - F_{RCRS}(\mathbf{x}(\mathbf{n}_i)) \mid^{\eta} = \sum_{j=1}^{|\Omega_M|} \mathcal{E}_j(\mathcal{S}(\mathbf{r}_j)). \tag{23}$$

It is easy to show that the LNE in (23) is minimized if and only if each of the $\mathcal{E}_j(\mathcal{S}(\mathbf{r}_j))$ error sums is minimized. Thus, the optimal RCRS filter classifier is given by

$$\mathcal{S}_{opt}(\mathbf{r}_j) = k \ : \ \mathcal{E}_j(k) \leq \mathcal{E}_j(l) \ \forall \ l \neq k \tag{24}$$

for $j = 1, 2, \ldots, |\Omega_M|$. If there is not a unique minimum error for some j, then a tie breaking rule must be employed. For example, a tie between two values satisfying (24) may be broken by choosing the order statistic corresponding to one of the minimum errors that is closest in rank to the median. In most practical cases, however, ties are unlikely given a sufficient number of training samples.

The optimization can also be performed recursively. The function $\mathcal{S}_{opt}(\cdot)$ can be updated as new training vectors become available. To do so, define the cumulative partitioned error as

$$R_{j,k}(m) \ = \sum_{i \in \Gamma_{j,m}} \mid d(\mathbf{n}_i) - x_{(k)}(\mathbf{n}_i) \mid^{\eta}. \tag{25}$$

The cumulative partitioned error term $R_{j,k}(m)$ contains the total error incurred by outputting the k^{th} order statistic, given the feature vector j is observed, up to index m in the training sequence. These cumulative error terms can be written as a vector yielding

$$\mathbf{R}_j(m) = \begin{bmatrix} R_{j,1}(m) \\ R_{j,2}(m) \\ \vdots \\ R_{j,N}(m) \end{bmatrix}. \tag{26}$$

The optimal function at index m in the training sequence is determined by the minimum element in $\mathbf{R}_j(m)$, and is given by

$$\mathcal{S}_{opt}^m(\mathbf{r}_j) = k \ : \ R_{j,k}(m) = min\ (R_{j,1}(m), R_{j,2}(m), \ldots, R_{j,N}(m)), \tag{27}$$

for $j = 1, 2, \ldots, |\Omega_M|$. Again, if there is not a unique minimum element in the vector $\mathbf{R}_j(m)$, then a tie breaking rule must be employed.

The iterative optimization procedure goes as follows. The optimal function is set to some initial value, such as the median yielding $\mathcal{S}_{opt}^0(\mathbf{r}_j) = (N+1)/2$, and $R_{j,k}(0) = 0$ for $j = 1, 2, \ldots, |\Omega_M|$ and $k = 1, 2, \ldots, N$. The index m is set to one and the feature vector index α_m is determined. Then, the cumulative error vector $\mathbf{R}_{\alpha_m}(m)$ is updated according to

$$\mathbf{R}_{\alpha_m}(m) = \mathbf{R}_{\alpha_m}(m-1) + \mathbf{P}(m), \tag{28}$$

where $\mathbf{P}(m)$ is a vector that contains the L_η normed difference between the desired signal $d(\mathbf{n}_m)$ and each of the order statistics in the observation vector. Specifically, $\mathbf{P}(m)$ is given by

$$\mathbf{P}(m) = \begin{bmatrix} |d(\mathbf{n}_m) - x_{(1)}(\mathbf{n}_m)|^\eta \\ |d(\mathbf{n}_m) - x_{(2)}(\mathbf{n}_m)|^\eta \\ \vdots \\ |d(\mathbf{n}_m) - x_{(N)}(\mathbf{n}_m)|^\eta \end{bmatrix}. \tag{29}$$

The optimal function $\mathcal{S}_{opt}^m(\mathbf{r}_{\alpha_m})$ is updated according to

$$\mathcal{S}_{opt}^m(\mathbf{r}_{\alpha_m}) = k \ : \ R_{\alpha_m,k}(m) = min\ (R_{\alpha_m,1}(m), R_{\alpha_m,2}(m), \ldots, R_{\alpha_m,N}(m)). \tag{30}$$

The index m is incremented, and the procedure repeats until the end of the training sequence is reached, or such a time that the filter has been determined to be sufficiently trained.

The recursive training algorithm is summarized in Table 1. Advantages of this deterministic training procedure are: (1) the training process always returns the globally optimal filter for the training set and (2) there is freedom to choose an error norm. In addition, an exponential "forgetting" factor can easily be added to the sum of L_η normed estimate errors to accommodate training data with changing statistics [5, 6].

Several examples of optimized functions are shown in Figs. 4-7. The training data used is the 512×512 image "Lena," which is shown in Fig. 13(a). For all of the optimized functions, a 9×9 window is used and δ is the index of the center sample. Figure 4 shows optimal first order filter functions $\mathcal{S}_{opt}(r_\delta)$ for the image corrupted by impulsive noise with various impulse probabilities. Notice that each of the functions has a linear region in which

Table 1: Recursive least L_η normed error training algorithm.

1.	Set $m = 1$, $\mathcal{S}^0_{opt}(\mathbf{r}_j) = \frac{N+1}{2}$ and $\mathbf{R}_{j,k}(0) = 0$, for $j = 1, 2, \ldots, \vert\Omega_M\vert$ and $k = 1, 2, \ldots, N$.
2.	Determine the feature vector index α_m.
3.	Update $\mathbf{R}_{\alpha_m}(m)$ according to $\mathbf{R}_{\alpha_m}(m) = \mathbf{R}_{\alpha_m}(m-1) + \mathbf{P}(m)$.
4.	Set $\mathcal{S}^m_{opt}(\mathbf{r}_{\alpha_m}) = k \;:\; R_{\alpha_m,k}(m) = min\ (R_{\alpha_m,1}(m), R_{\alpha_m,2}(m), \ldots, R_{\alpha_m,N}(m))$.
5.	If $m = K$ or filter is sufficiently trained, stop; else increment m and go to 2.

the input rank equals the output rank. However, when r_δ is in the extreme ranks, the output is a rank closer to the median. This provides the impulse rejection. Note that the break point moves in as the impulse probability increases.

Figure 5 shows several $\mathcal{S}_{opt}(r_\delta)$ functions for "Lena" corrupted by various levels of additive Gaussian noise. In this case, the optimal functions are approximately linear, with slope inversely proportional to the noise level. For no noise, the optimal function has a slope of one, representing an identity filter. For very high noise levels, the slope of the optimal function approaches zero, reducing the filter to a median.

Figure 6 shows several optimal functions for "Lena" corrupted by additive contaminated Gaussian noise. We denote the contaminated Gaussian noise probability density function as $\Phi(\sigma_1, \sigma_2, \varepsilon)$. With probability $1 - \varepsilon$, a noise sample is normally distributed with zero mean and variance σ_1^2, and with probability ε, a noise sample is normally distributed with zero mean and variance σ_2^2. In general, $\sigma_1 < \sigma_2$ and ε represents the "contamination" probability. Figure 6(a) shows several $\mathcal{S}_{opt}(r_\delta)$ functions for the image corrupted by $\Phi(5, 100, \varepsilon)$ contaminated Gaussian noise. Notice that there is a linear region like in the Gaussian noise case and a cut off region like in the impulsive noise case. Figure 6(b) shows several $\mathcal{S}_{opt}(r_\delta)$ functions for "Lena" corrupted by $\Phi(10, 100, \varepsilon)$ contaminated Gaussian noise. Here the slope of the linear regions are lower due to the higher level of background noise.

Figure 7 shows two optimal second order functions for a 9×9 window, where $\mathbf{r}^* = [r_\delta, r_{\delta+1}]$. The index δ represents that of the center sample and $\delta + 1$ is the index of the sample immediately to the right of center. In these plots, the height of the mesh represents

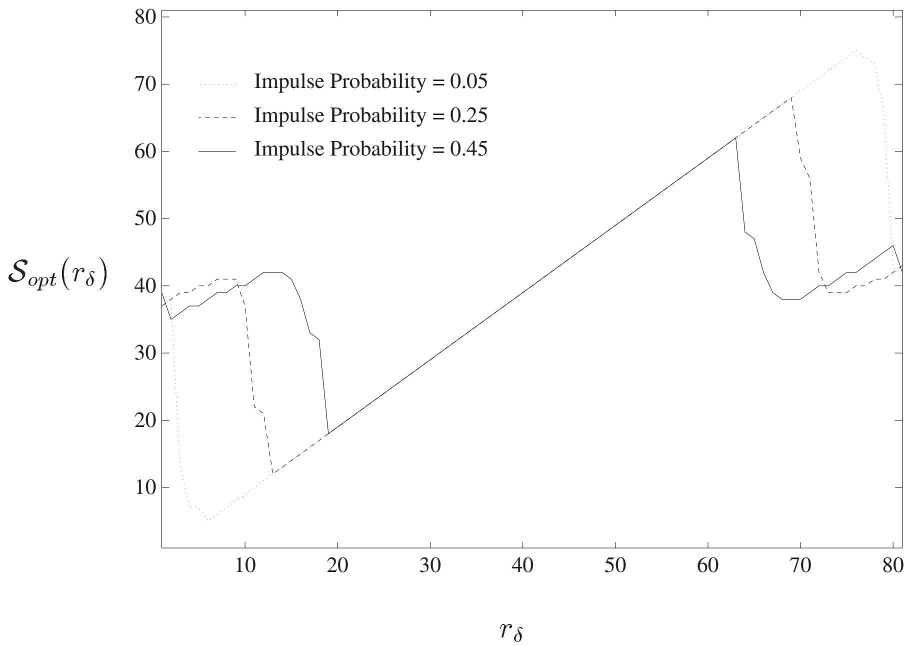


Figure 4: The optimal functions $\mathcal{S}_{opt}(r_\delta)$ for 9×9 RCRS filters trained on "Lena" with impulsive noise.

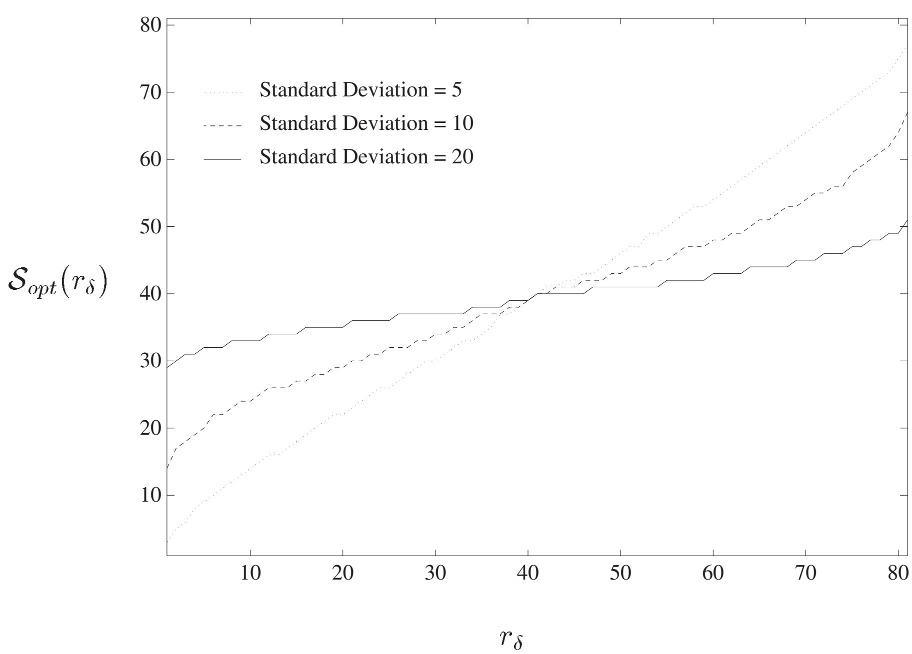
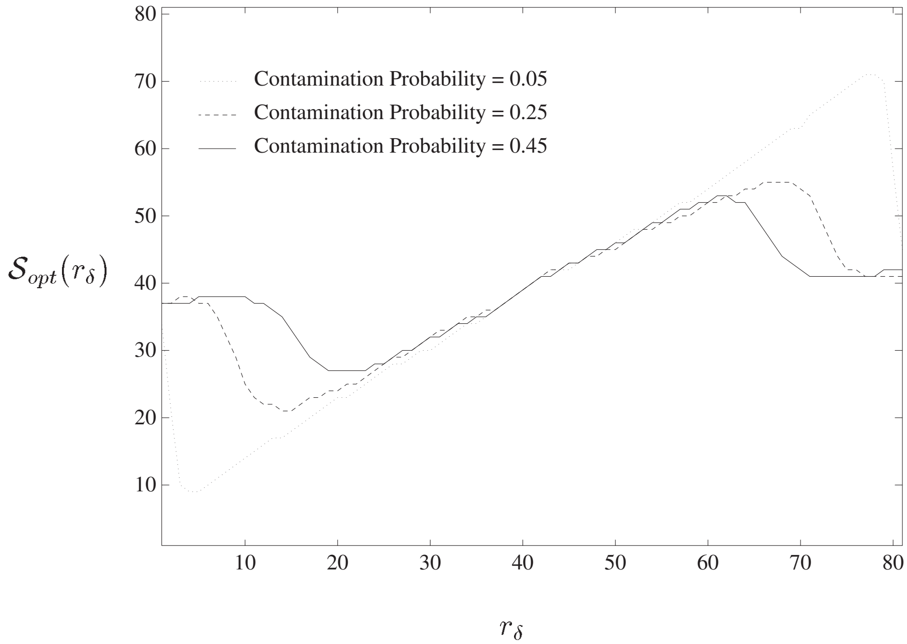
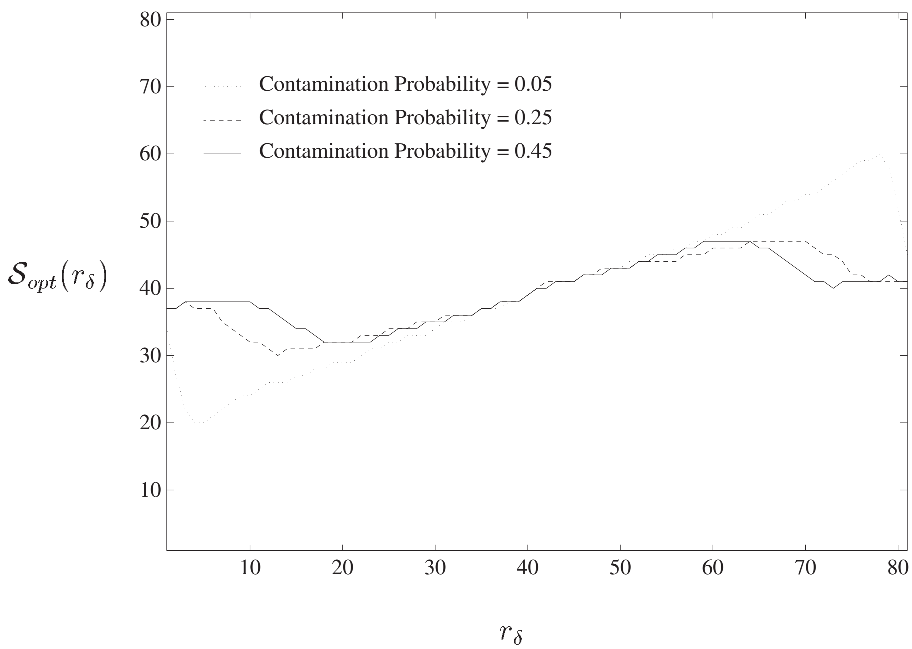


Figure 5: The optimal functions $\mathcal{S}_{opt}(r_\delta)$ for 9×9 RCRS filters trained on "Lena" with additive Gaussian noise.

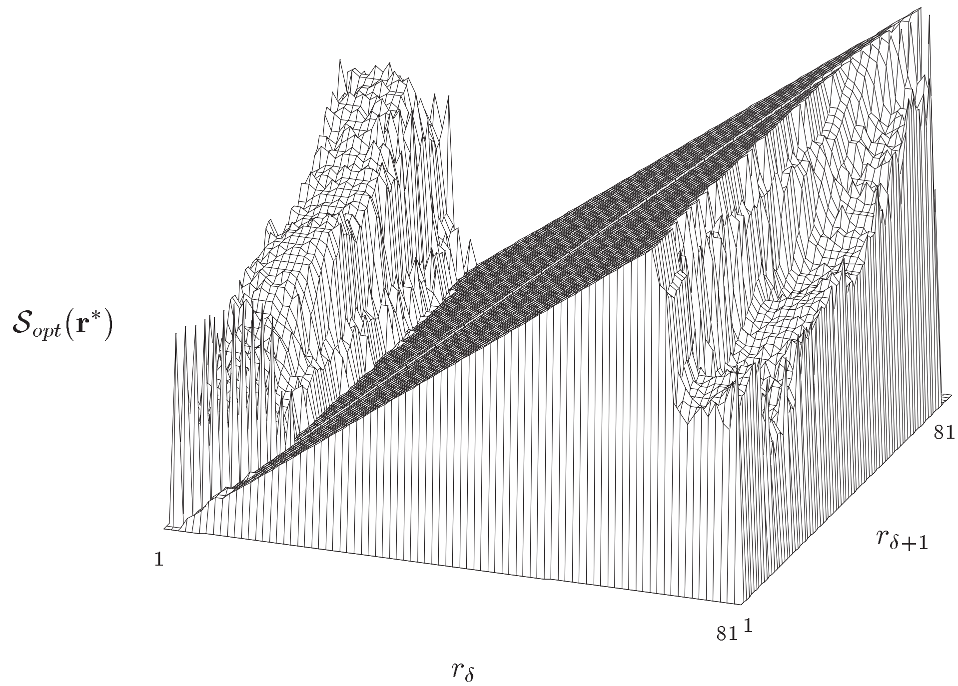


(a)

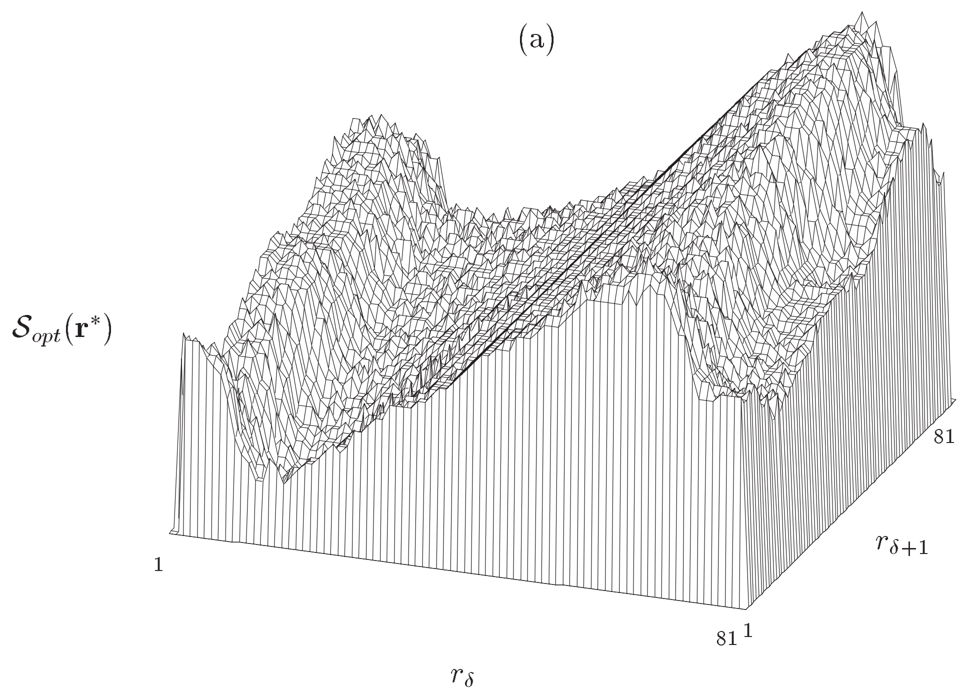


(b)

Figure 6: The optimal functions $\mathcal{S}_{opt}(r_\delta)$ for 9×9 RCRS filters trained on "Lena" with (a) $\Phi(5, 100, \varepsilon)$ additive contaminated Gaussian noise and (b) $\Phi(10, 100, \varepsilon)$ additive contaminated Gaussian noise.



(a)



(b)

Figure 7: The optimal functions $\mathcal{S}_{opt}(\mathbf{r}^*)$ for 9×9 RCRS filters where $\mathbf{r}^* = [r_\delta, r_{\delta+1}]$ trained on "Lena" with (a) impulsive noise ($p = 0.25$) and (b) $\Phi(5, 100, 0.25)$ additive contaminated Gaussian noise.

the output rank for the given feature vector. Figure 7(a) shows $\mathcal{S}_{opt}(\mathbf{r}^*)$ for "Lena" corrupted by impulsive noise. Note that for a fixed $r_{\delta+1}$, the shape of the curve along the r_δ axis is similar to that of the first order function. However, as $r_{\delta+1}$ is increased, the function is biased to output higher order statistics. For lower values of $r_{\delta+1}$, the function is biased to output lower order statistics. The additional information provided by $r_{\delta+1}$ allows the second order filters to have more sophisticated decision rules and gives them improved performance over the order one filters. Figure 7(b) shows $\mathcal{S}_{opt}(\mathbf{r}^*)$ for the image corrupted by $\Phi(5, 100, 0.25)$ contaminated Gaussian noise. Again, the effect of both ranks r_δ and $r_{\delta+1}$ in selecting the output rank can be seen.

4 Properties of the RCRS Filter

In this section, statistical and deterministic properties of the RCRS filters are developed. All properties are derived for the general case of order M RCRS filters. Through the study of these properties, the design and analysis of RCRS filters is aided.

A Statistical Properties

The first statistical property considered is the impulsive noise breakdown probability, introduced in [16]. The breakdown probability is the probability of a filter outputting an impulse, given a certain probability of impulses appearing in the observed signal.

Consider the case of an i.i.d. signal corrupted by independent impulsive noise where a signal sample is replaced by $\pm\infty$ with probability p, otherwise it is unaltered. Let the probability of negative impulse be $p/2$ and the probability of a positive impulse be $p/2$. Given this, the breakdown probability for RCRS filters is defined in the following property.

Property 4.1 (Breakdown probability) *The breakdown probability for an RCRS filter characterized by $\mathcal{S}(\cdot)$ is given by*

$$\Pr(F_{RCRS}(\mathbf{x}) = \pm\infty) = \frac{(N-M)!}{N!}\sum_{\mathbf{i}\in\Omega_M}\left(\sum_{l=\mathcal{S}(\mathbf{i})}^{N}\binom{N}{l}\left(\frac{p}{2}\right)^l\left(1-\frac{p}{2}\right)^{N-l}+\right. \tag{31}$$

$$\left.\sum_{m=N-\mathcal{S}(\mathbf{i})+1}^{N}\binom{N}{m}\left(\frac{p}{2}\right)^m\left(1-\frac{p}{2}\right)^{N-m}\right). \tag{32}$$

Proof: The probability of an RCRS filter outputting an impulse is given by

$$\Pr(F_{RCRS}(\mathbf{x}) = \pm\infty) = \sum_{\mathbf{i}\in\Omega_M}\Pr(x_{(\mathcal{S}(\mathbf{r}^*))} = \pm\infty \mid \mathbf{r}^* = \mathbf{i})\Pr(\mathbf{r}^* = \mathbf{i}) \tag{33}$$

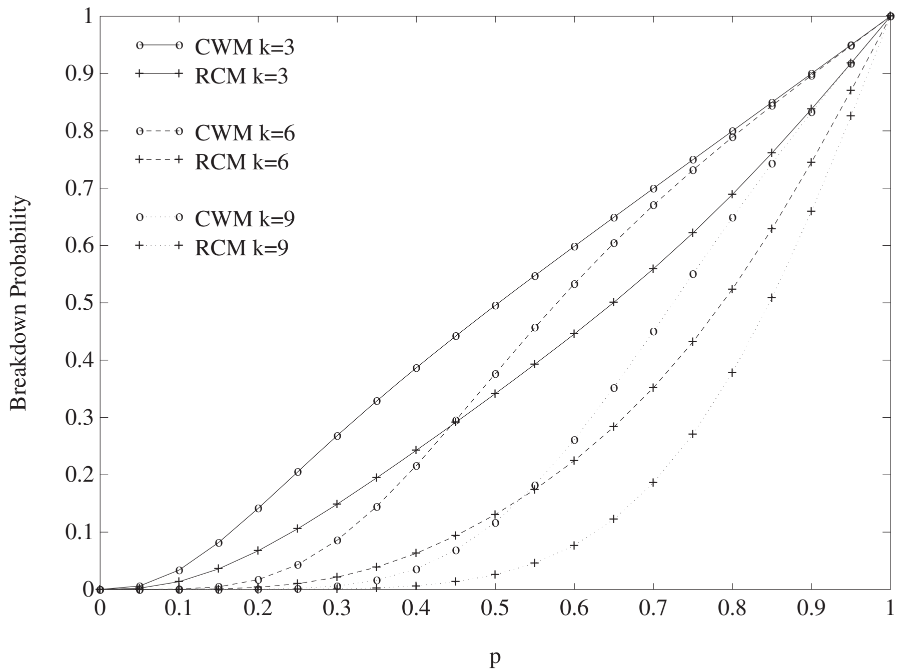


Figure 8: Breakdown probabilities for the RCM and CWM filters where $N = 25$.

$$= \sum_{\mathbf{i}\in\Omega_M} \Pr(x_{(\mathcal{S}(\mathbf{i}))} = \pm\infty) \Pr(\mathbf{r}^* = \mathbf{i}) \tag{34}$$

$$= \frac{(N-M)!}{N!} \sum_{\mathbf{i}\in\Omega_M} \Pr(x_{(\mathcal{S}(\mathbf{i})} = \pm\infty) \tag{35}$$

$$= \frac{(N-M)!}{N!} \sum_{\mathbf{i}\in\Omega_M} (\Pr(\text{at least } \mathcal{S}(\mathbf{i}) \text{ samples out of } N = -\infty) \;+$$
$$\Pr(\text{at least } N - \mathcal{S}(\mathbf{i}) + 1 \text{ samples out of } N = +\infty)). \tag{36}$$

Summing these sets of binomial probabilities yields (31). □

Selected breakdown probabilities are plotted in Fig. 8 for the RCM and CWM filters. Notice that the breakdown probabilities for the RCM filter are less than those of the CWM filter with the same parameter. This is because the RCM filter outputs the median rather than rank k and $N-k+1$ if r_δ is outside the range of k and $N-k+1$. The median is less likely to be a corrupted sample than the samples with rank k and $N-k+1$ for $k < (N+1)/2$.

Another important statistical property of the RCRS filters is the probability that the output is x_δ. In other words, the probability that the RCRS filter performs the identity operation. The probability of identity operation can be written as the ratio of set cardinalities when r_δ is included in the RCRS feature vecor, which is the case in most practical applications.

Property 4.2 (Probability of identity operation) *For an RCRS filter where r_δ is contained in $\mathbf{r}^*$ and the input samples are i.i.d. with a continuous distribution, the probability that the output is equal to x_δ is given by*

$$\Pr(F_{RCRS}(\mathbf{x}) = x_\delta) = |\Lambda| \frac{N!}{(N-M)!}, \tag{37}$$

where $\Lambda = \{\mathbf{i} \in \Omega_M : \mathcal{S}(\mathbf{i}) = r_\delta\}$.

Proof : For an RCRS filter operating on input samples from a continuous distribution,

$$\Pr\left(F_{RCRS}(\mathbf{x}) = x_\delta\right) = \begin{cases} 1 & \text{if } \mathbf{r}^* \in \Lambda \\ 0 & \text{otherwise} \end{cases}. \tag{38}$$

If the input samples are also i.i.d., then $\Pr(\mathbf{r}^* \in \Lambda) = |\Lambda|/|\Omega_M|$. Substituting for the cardinality of Ω_M yields (37). □

It is also informative to know the cumulative distribution function (cdf) of RCRS filter output samples. Let the cdf of the input and output samples be denoted by $\Psi_X(\cdot)$ and $\Psi_Y(\cdot)$ respectively. Assuming i.i.d. input samples and given $\mathcal{S}(\cdot)$, the cdf of the output samples is given in the following property.

Property 4.3 (Output distribution) *The cdf of the output samples of an RCRS filter characterized by $\mathcal{S}(\cdot)$ in the case of i.i.d. input samples is given by*

$$\Psi_Y(y) = \frac{(N-M)!}{N!} \sum_{\mathbf{i} \in \Omega_M} \sum_{j=\mathcal{S}(\mathbf{i})}^{N} \binom{N}{j} \Psi_X^j(y)(1-\Psi_X(y))^{N-j}. \tag{39}$$

Proof: The cdf of the output samples of an RCRS filter is given by

$$\begin{aligned}
\Psi_Y(y) &= \Pr(F_{RCRS}(\mathbf{x}) < y) & (40)\\
&= \sum_{\mathbf{i} \in \Omega_M} \Pr(x_{(\mathcal{S}(\mathbf{r}^*))} < y \mid \mathbf{r}^* = \mathbf{i}) \Pr(\mathbf{r}^* = \mathbf{i}) & (41)\\
&= \sum_{\mathbf{i} \in \Omega_M} \Pr(x_{(\mathcal{S}(\mathbf{i}))} < y) \Pr(\mathbf{r}^* = \mathbf{i}) & (42)\\
&= \frac{(N-M)!}{N!} \sum_{\mathbf{i} \in \Omega_M} \Pr(x_{(\mathcal{S}(\mathbf{i}))} < y) & (43)\\
&= \frac{(N-M)!}{N!} \sum_{\mathbf{i} \in \Omega_M} \Pr(\text{ At least } \mathcal{S}(\mathbf{i}) \text{ samples out of } N \text{ are } < y). & (44)
\end{aligned}$$

These probabilities can be found as a sum of binomial probabilities yielding (39) □

The probability density functions (pdfs) can be found from (39) by means of differentiation. Selected pdfs are plotted in Fig. 9 for the RCM and CWM filters. From the plots, it is clear that the variance of the output samples of both filters decreases as the parameter k is increased. Notice that some of the pdfs for the CWM filter are bimodal. This is because the output of the CWM is often $x_{(k)}$ or $x_{(N-k+1)}$, which lie on different sides of the median. This does not occur with the RCM filter.

B Deterministic Properties

In this subsection, deterministic properties of the RCRS filters are presented. The first deterministic property, which relates to the generalizability of a filter class, is scale and bias invariance.

Property 4.4 (Scale and bias invariance) *RCRS filters have the property of scale and bias invariance. Specifically if* $\mathbf{y} = a\mathbf{x} + b\mathbf{1}$, *where* $\mathbf{1}$ *is an* N*-vector of ones, then*

$$F_{RCRS}(\mathbf{y}) = aF_{RCRS}(\mathbf{x}) + b \tag{45}$$

for $a \geq 0$ *and* $-\infty < b < \infty$. *If the function* $\mathcal{S}(\mathbf{r}^*)$ *has the symmetry* $\mathcal{S}(\mathbf{r}^*) = N - \mathcal{S}(N\mathbf{1} - \mathbf{r}^* + \mathbf{1}) + 1$, *then (45) is valid for* $-\infty < a, b < \infty$.

Proof: The sorted elements from the vector $\mathbf{y}$, where $a > 0$, are

$$ax_{(1)} + b \leq ax_{(2)} + b \leq \cdots \leq ax_{(N)} + b. \tag{46}$$

Each element in $\mathbf{x}$ remains in the same relative rank in the linearly transformed vector $\mathbf{y}$. Thus, if x_{γ_i} has rank r_{γ_i} in $\mathbf{x}$, then y_{γ_i} has rank r_{γ_i} in $\mathbf{y}$. Consequently,

$$F_{RCRS}(\mathbf{y}) = y_{(\mathcal{S}(\mathbf{r}^*))} = ax_{(\mathcal{S}(\mathbf{r}^*))} + b = aF_{RCRS}(\mathbf{x}) + b. \tag{47}$$

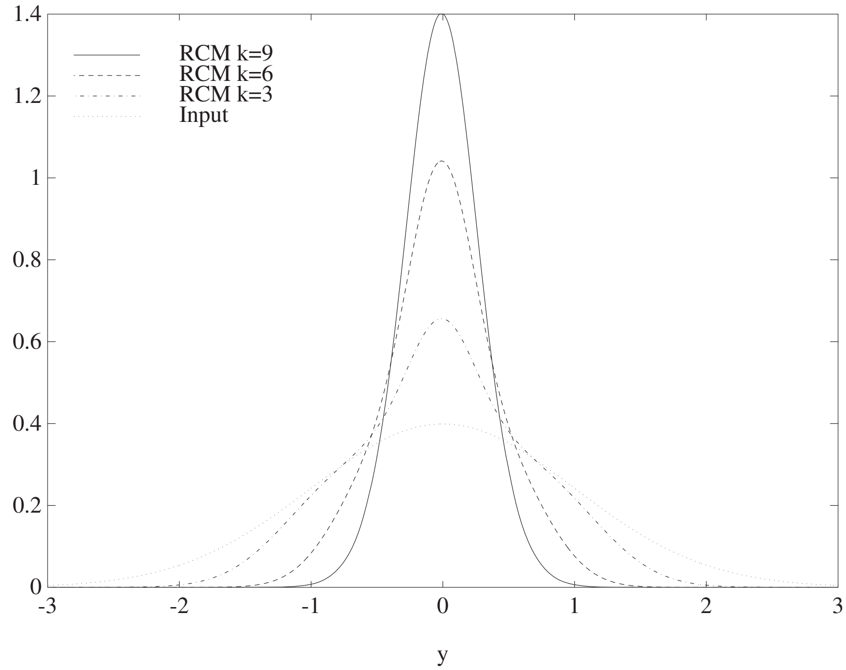
If $a < 0$, then the sorted elements in the vector $\mathbf{y}$ are

$$ax_{(N)} + b \leq ax_{(N-1)} + b \leq \cdots \leq ax_{(1)} + b. \tag{48}$$

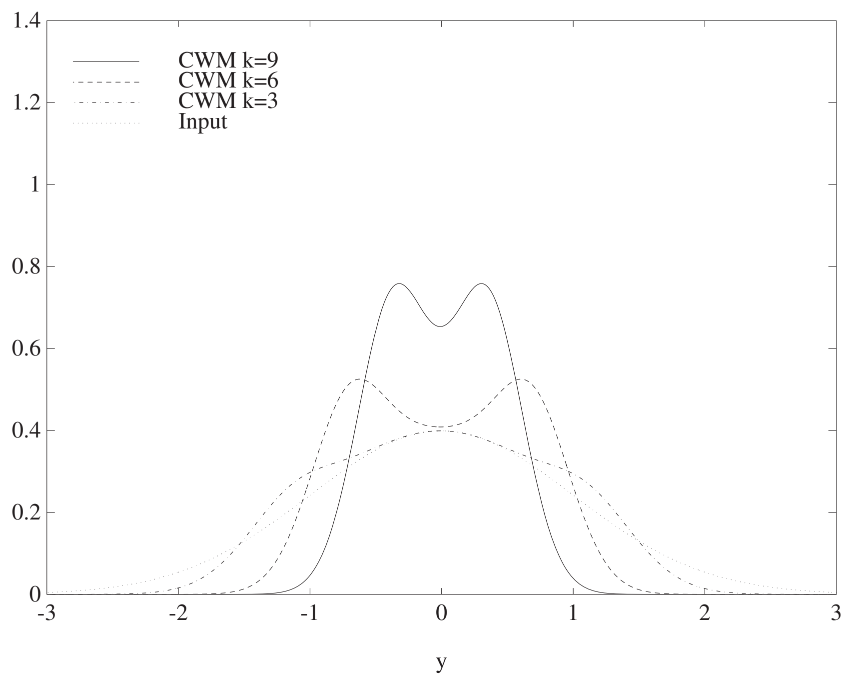
In this case, if x_{γ_i} has rank r_{γ_i} in $\mathbf{x}$, then y_{γ_i} has rank $N - r_{\gamma_i} + 1$ in $\mathbf{y}$. So if $\mathcal{S}(\mathbf{r}^*) = N - \mathcal{S}(N\mathbf{1} - \mathbf{r}^* + \mathbf{1}) + 1$ then

$$F_{RCRS}(\mathbf{y}) = y_{(\mathcal{S}(N\mathbf{1}-\mathbf{r}^*+\mathbf{1}))} = y_{(N-\mathcal{S}(\mathbf{r}^*)+1)} = ax_{(\mathcal{S}(\mathbf{r}^*))} + b = aF_{RCRS}(\mathbf{x}) + b. \tag{49}$$

The case where $a = 0$ is trivial. □



(a)



(b)

Figure 9: Output probability density functions for (a) size $N = 25$ RCM filters and (b) size $N = 25$ CWM filters. The input is normally distributed with zero mean and unit variance.

Thus, the RCRS filters are not be sensitive to changes in scale and bias. This is improtant because these parameters often vary from image to image.

We now focus attention on root signal analysis. A root signal of a filter is one which is unchanged by the filtering operation. Root signal analysis has proven to be a useful tool for evaluating nonlinear filters [2, 9, 18]. By investigating root signals, one can gain insight into the performance of a given filtering algorithm. Root signals can also aid in filter design.

Property 4.5 (Root signals: sufficient conditions) *A signal is a root of an RCRS filter characterized by* $\mathcal{S}(\cdot)$ *if* $\mathbf{r}^*(\mathbf{n}) \in \{\mathbf{i} \in \Omega_M : \mathcal{S}(\mathbf{i}) = r_\delta\}$ *for all* $\mathbf{n}$.

Proof: For a signal such that $\mathbf{r}^*(\mathbf{n}) \in \{\mathbf{i} \in \Omega_M : \mathcal{S}(\mathbf{i}) = r_\delta\}$ for all $\mathbf{n}$, the output of an RCRS filter characterized by $\mathcal{S}(\cdot)$ is given by

$$F_{RCRS}(\mathbf{x}(\mathbf{n})) = x_{(\mathcal{S}(\mathbf{r}^*(\mathbf{n})))}(\mathbf{n}) = x_{(r_\delta)}(\mathbf{n}) = x_\delta(\mathbf{n}) \tag{50}$$

for all $\mathbf{n}$. Thus, such a signal is root of the RCRS filter. □

Necsessary conditions for signals to be roots of an RCRS filter are defined in the following property.

Property 4.6 (Root signals: necessary conditions) *Let the signal* $\{x(\mathbf{n})\}$ *be a root of the RCRS filter characterized by* $\mathcal{S}(\cdot)$. *If for all* $\mathbf{n}$ $x_k(\mathbf{n}) \neq x_\delta(\mathbf{n})$ *for* $k \in \{1, 2, \ldots, N\}, k \neq \delta$, *then* $\mathbf{r}^*(\mathbf{n}) \in \{\mathbf{i} \in \Omega_M : \mathcal{S}(\mathbf{i}) = r_\delta\}$ *for all* $\mathbf{n}$.

Proof: If $\{x(\mathbf{n})\}$ is a root signal, then for all $\mathbf{n}$

$$F_{RCRS}(\mathbf{x}(\mathbf{n})) = x_{(\mathcal{S}(\mathbf{r}^*(\mathbf{n})))} = x_\delta(\mathbf{n}). \tag{51}$$

Also, if for all $\mathbf{n}$ $x_k(\mathbf{n}) \neq x_\delta(\mathbf{n})$ for $k \in \{1, 2, \ldots, N\}, k \neq \delta$, then

$$F_{RCRS}(\mathbf{x}(\mathbf{n})) = x_{(\mathcal{S}(\mathbf{r}^*(\mathbf{n})))} \equiv x_\delta(\mathbf{n}) \equiv x_{(r_\delta)}(\mathbf{n}) \tag{52}$$

for all $\mathbf{n}$. This can only occur if $\mathbf{r}^*(\mathbf{n}) \in \{\mathbf{i} \in \Omega_M : \mathcal{S}(\mathbf{i}) = r_\delta\}$ for all $\mathbf{n}$. □

From Property 4.5 and 4.6, it follows that a signal with samples that derive from a continuous distribution is a root of an RCRS filter characterized by $\mathcal{S}(\cdot)$ iff $\mathbf{r}^*(\mathbf{n}) \in \{\mathbf{i} \in \Omega_M : \mathcal{S}(\mathbf{i}) = r_\delta\}$ with probability 1. The following property gives sufficient conditions for a signal to be a root of two RCRS filters.

Property 4.7 (Shared root signals) *Let* $\{x(\ )\}$ $x_k(\ ) \neq x_\delta(\ )$ $k \in \{1, 2, \ldots, N\}, k \neq \delta$ N M $\mathcal{S}_1(\cdot)$ N M $\mathcal{S}_2(\cdot)$ $\{\ \in \Omega_M : \mathcal{S}_1(\ ) = r_\delta\} = \{\ \in \Omega_M : \mathcal{S}_2(\ ) = r_\delta\}$

If a signal $\{x(\ )\}$ is such that for all $x_k(\ ) \neq x_\delta(\ )$ for $k \in \{1, 2, \ldots, N\}, k \neq \delta$ and is a root of a size N order M RCRS filter characterized by $\mathcal{S}_1(\cdot)$, then by Property 4.6 $^*(\ ) \in \{\ \in \Omega_M : \mathcal{S}_1(\ ) = r_\delta\}$ for all . If a second size N order M RCRS filter characterized by $\mathcal{S}_2(\cdot)$ is such that

$$\{\ \in \Omega_M : \mathcal{S}_2(\ ) = r_\delta\} = \{\ \in \Omega_M : \mathcal{S}_1(\ ) = r_\delta\}, \tag{53}$$

then $^*(\ ) \in \{\ \in \Omega_M : \mathcal{S}_2(\ ) = r_\delta\}$ for all . Thus, by Property 4.5, $\{x(\ )\}$ is a root of the RCRS filters characterized by $\mathcal{S}_2(\cdot)$. □

For example, consider a signal $\{x(\ )\}$ in which for all $x_k(\ ) \neq x_\delta(\ )$ for $k \in \{1, 2, \ldots, N\}, k \neq \delta$. If this signal is a root of the CWM filter of size N with parameter k, then by Property 4.7 this signal is also a root of the RCM filter of size N with parameter k.

5 Experimental Results

The proposed filters can be used in a variety of signal restoration applications. Here we consider the application of these filters to the restoration of an image corrupted by impulsive noise and contaminated Gaussian noise. Quantitative error results are presented and several filtered images are shown for subjective evaluation. The RCRS filter are compared to the median, CWM, WOS, and (where possible) stack filters.

The RCRS filters discussed in this section have the following parameters: for $M = 1$, $\gamma_1 = \delta$ where δ is the index of the center sample in the window; for $M = 2$, $\gamma_1 = \delta$ and $\gamma_2 = \delta + 1$ where $\delta + 1$ is the index of the sample to the right of center; for $M = 3$, $\gamma_1 = \delta$, $\gamma_2 = \delta + 1$ and $\gamma_3 = \delta - 1$ where $\delta - 1$ is the index of the sample to the left of center. The training procedure used to obtain the following simulation results for the RCRS filters is the LNE (L_1) algorithm presented in Section 3C. The optimal CWM filter is found by means of an exhaustive search over the parameter k as defined in (7). The WOS and stack filter training procedures used are those described in [15] and [23].

In the following results, the filters are operating on the image "Lena" and have been trained using the image "Albert." Both images "Lena" and "Albert" are shown in Fig. 13.

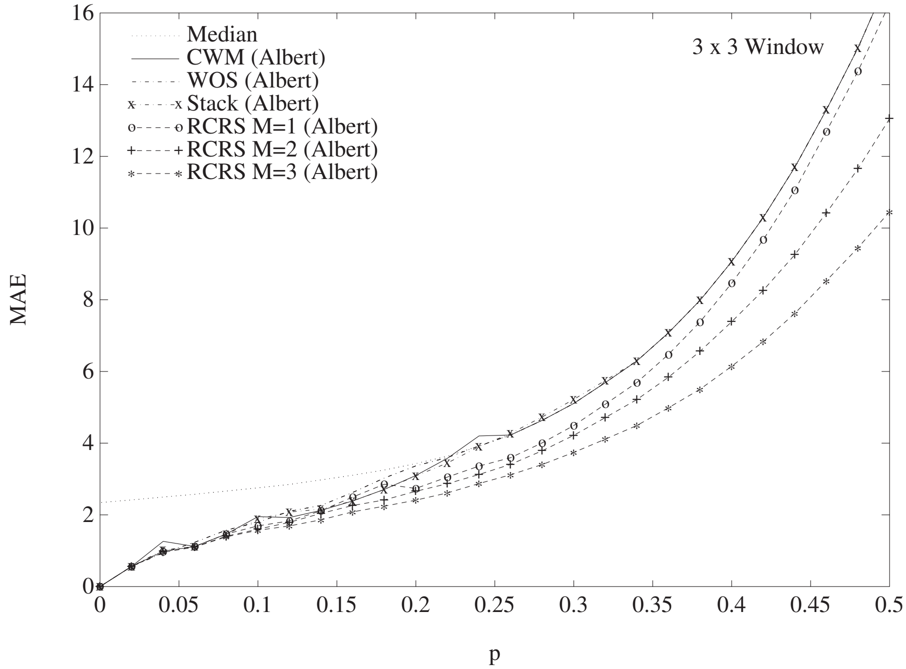
Notice that the two images are quite different in structure. The reason the filters have been trained using an image which is different from the one being filtered, is to present a more realistic scenerio. Using training data which has statistics that are very similar to that of the data being filtered generally gives the best results. We illustrate this idea later in this section.

Figure 10 shows the mean absolute errors (MAE) for RCRS filters operating on the image "Lena" corrupted by impulsive noise with impulse probability p. The impulses take on positive and negative values with equal probability ($p/2$). Figure 10(a) shows the MAE for filters with a 3×3 window. In this plot, the CWM, WOS filter and stack filter estimate errors are approximately equal. The reason the WOS filter and stack filter produces a higher estimate error in a few cases is due to the fact that each filter was optimized for the image "Albert," not "Lena." As the figure shows, the RCRS filters give the best results. The order three filter gives the lowest error followed by the order two and order one filter respectively. Also note that for high noise probabilities, the CWM, WOS filter and stack filter errors are approximately equal to that of the median, while the RCRS filters give significantly lower errors.

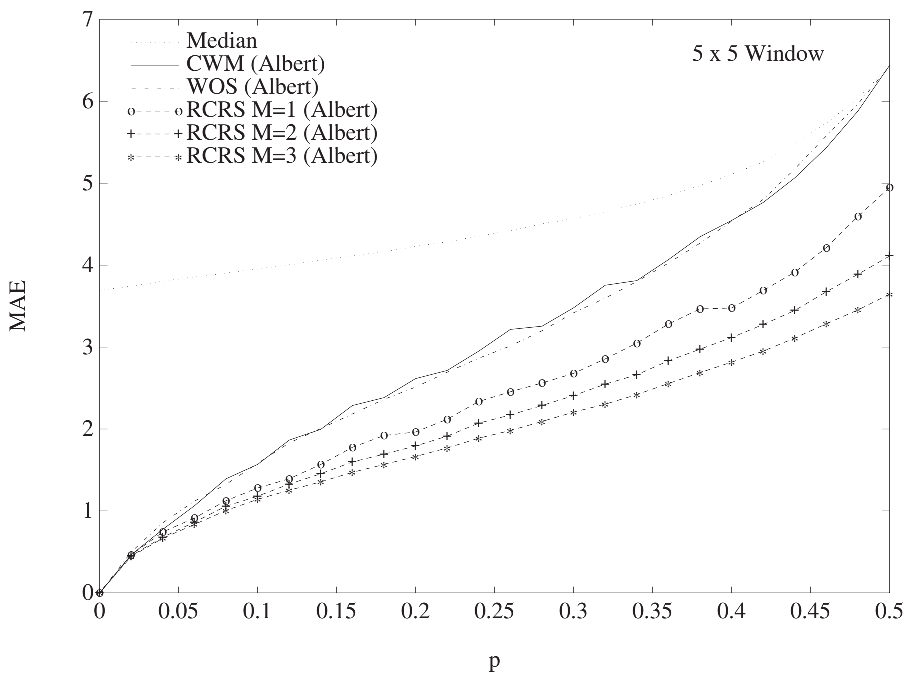
Figure 10(b) shows the MAEs for filters with a 5×5 window. For this window size, stack filters are impractical to implement, thus, these results can not be shown. Figure 10(b) shows that the RCRS filters give significantly lower errors than the CWM and WOS filters. Also, note the the errors for the 5×5 RCRS filters are lower than those of the 3×3 RCRS filters.

Figures 11 and 12 show pairs of MAE curves for each filter type. One of the error curves corresponds to the specified filter trained on "Lena" operating on "Lena." The other corresponds to the specified filter trained on "Albert" operating on "Lena." These results illustrate the generalizability of the filters. Figure 11 shows the MAEs for the filters operating on "Lena" corrupted with impulsive noise. Figure 11(a) shows the results for filters with a 7×7 window. Figure 11(b) shows the results for filters with a 9×9 window. In both cases, the RCRS filters give the best results. The filters which have been trained on "Lena" have a slightly lower error than those trained on "Albert," as would be expected. The loss in performance due to training on a "Albert," as opposed to "Lena," is comparable for the CWM, WOS and RCRS filters. The fact that the difference in performance for the filters trained "Albert" and "Lena" is very small, indicates that the filters generalize extremely well for this type of corruptive process.

Figure 12 shows the MAEs for the filters operating on "Lena" corrupted by $\Phi(5, 100, \varepsilon)$



(a)



(b)

Figure 10: MAE for the RCRS filters and others operating on the image "Lena" corrupted by impulsive noise with impulse probability p. Each filter was optimized using the image "Albert." The results using a 3×3 window are shown in (a) and the results using a 5×5 window are shown in (b).

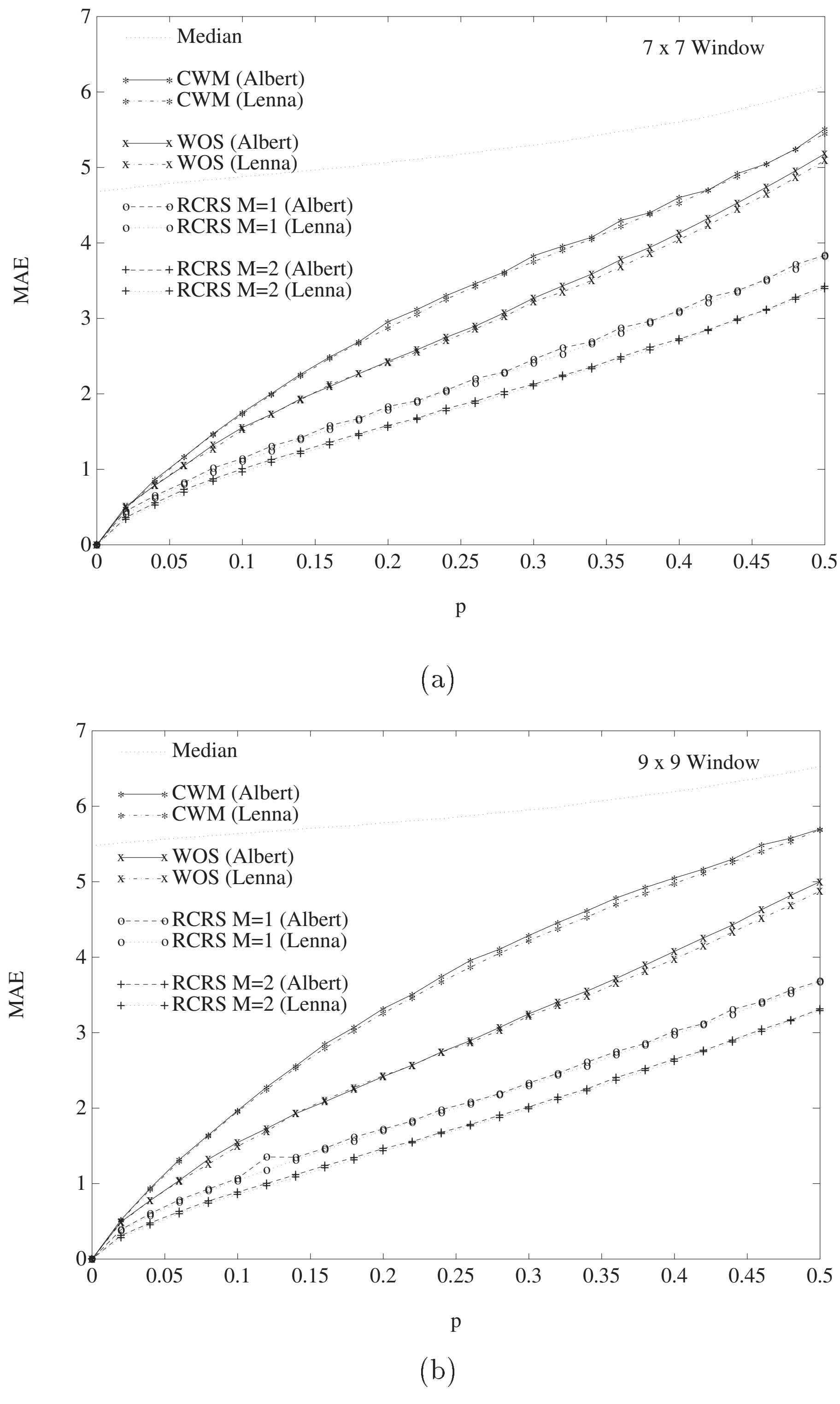
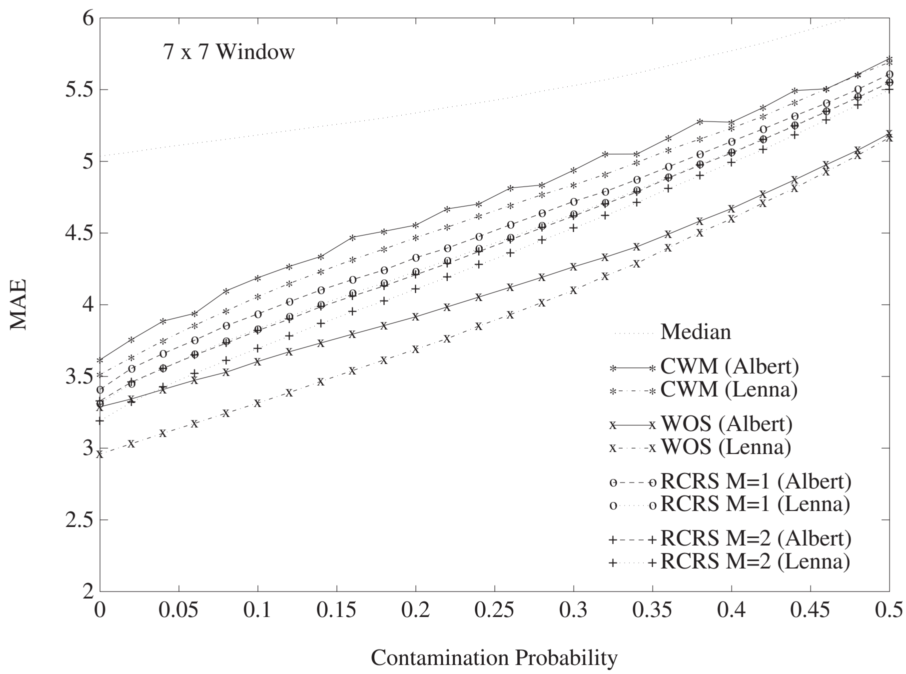
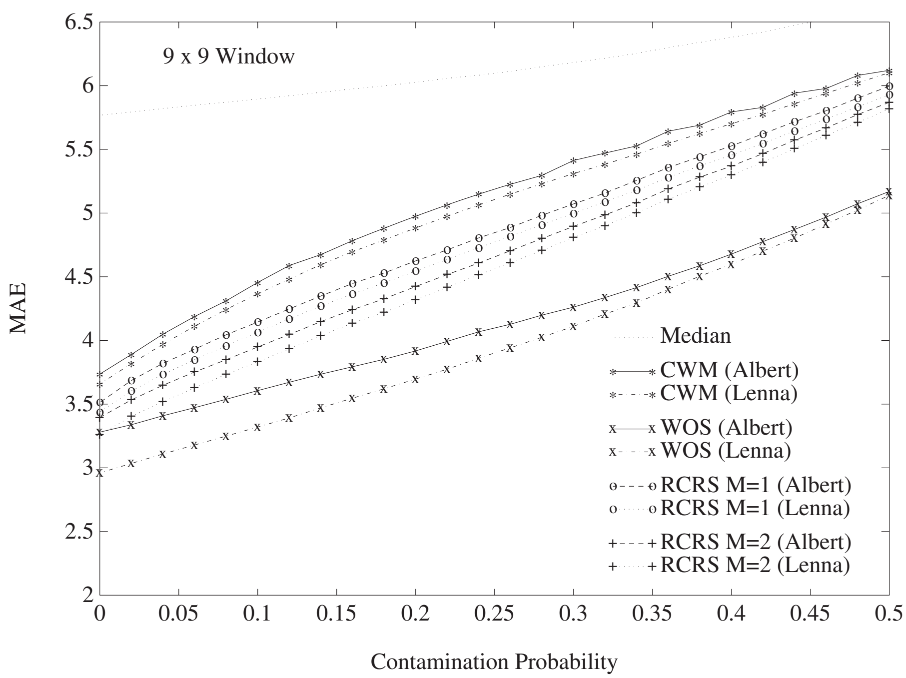


Figure 11: MAE for the RCRS filters and others operating on the image "Lena" corrupted by impulsive noise with impulse probability p. Each filter was optimized using the image specified in the key (either "Lena" or "Albert"). The results using a 7×7 window are shown in (a) and the results using a 9×9 window are shown in (b).



(a)



(b)

Figure 12: MAE for the RCRS filters and others operating on the image "Lena" corrupted by $\Phi(5, 100, \varepsilon)$ contaminated Gaussian noise. Each filter was optimized using the image specified in the key (either "Lena" or "Albert"). The results using a 7×7 window are shown in (a) and the results using a 9×9 window are shown in (b).

contaminated Gaussian noise. Figure 12(a) shows the results for filters with a 7×7 window. Figure 12(b) shows the results for filters with a 9×9 window. Here the WOS filters give the lowest error. This is probably due to the fact that WOS filters use some information about all the samples in the window, whereas low order RCRS filters, use more detailed informaion about fewer samples. Using spatial information which is spread out within the window appears to give better results in this application. Improved results can be obtained with the RCRS filter by increasing the order and using information from more samples. Again, note that those filters trained on "Lena" have a lower error than those trained on "Albert." The WOS filter appears to have a greater loss in performance as a result of training on "Albert" for the low contamination probability cases than the other filters.

Since it is difficult to judge the performance of image processing algorithms based solely on quantitative analysis, we show several filtered images for subjective evaluation. The original images "Lena" and "Albert" are shown in Figs. 13(a) and 13(b) respectively. Figure 13(c) shows the image "Lena" corrupted with impulsive noise where $p = 0.20$. The image restored using a 5×5 CWM filter trained on "Albert" is shown in Fig. 14(a). The image restored using a 5×5 WOS filter trained on "Albert" is shown in Fig. 14(b). Figures 14(c) and 14(d) show the image restored using first and second order RCRS filters respectively. Both RCRS filters use a 5×5 window and were trained on "Albert."

While the CWM and WOS filters do a fairly good job, the edges appear to be cleaner on the images filtered with the RCRS filters. The second order RCRS filter appears to have removed more of the impulses than the order one filter, resulting in a high quality restoration. Note that with the L_1 norm, there tends to be less penalty for allowing impulses to pass than with higher normed error measures. If an L_2 norm is used, the resulting CWM and RCRS filters would tend to suppress the impulses better, but at the expense of image detail.

Next, images restored using a larger window size are shown. Figure 15(a) shows the image restored using a 9×9 CWM filter trained on "Albert." Figure 15(b) shows the image restored using a 9×9 WOS filter trained on "Albert." Figures 15(c) and 15(d) show the image restored using first and second order RCRS filters respectively. Both RCRS filters use a 9×9 window and were trained on "Albert." Here, the CWM filter suppresses the impulses well, but causes significant distortion at edges. On the other hand, the WOS filter preserves detail fairly well, but leaves many impulses. The RCRS filters appear to both preserve image detail and remove most of the impulses. Again, the second order filter appears to have removed more impulses than the first order filter while providing the same level of detail preservation.

6 Conclusions

A new class of nonlinear filters, RCRS filters, has been introduced. These filters are developed under the framework of RS filters. RS filters are all filters constrained to output a sample from the set of rank ordered input samples. Many previously proposed rank order based filters can be formulated as RS filters including the median filter, CWM filter, stack filter and permutation filter. Each of these filters, however, uses different information in selecting an output order statistic. The RCRS filters use the rank of selected input samples as the basis for the output rank selection. A deterministic optimization procedure is described here. This optimization guarantees the optimal filter for the given training data with any L_η normed error. Also, the properties developed here can aid in the design and analysis of the RCRS filters. The simulation results show that for some image restoration applications, the RCRS filters outperform the median filter, CWM filter, and stack filter (under the MAE criteria). In addition, the low order RCRS filters can be implemented with much larger window sizes than stack filters or permutation filters. Finally, the operation of the low order filters is straight forward and intuitive.

References

[1] G. R. Arce and R. Foster, "Detail-Preserving Ranked-Order Based Filters for Image Processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-37, No. 1, January, 1989.

[2] G. R. Arce and N. C. Gallagher, Jr., "State Descriptions of the Root Signal Set of Median Filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-30, No. 6, December, 1982.

[3] G. R.Arce, N. C. Gallagher, Jr. and T. A. Nodes, "Median Filters: Theory for One and Two Dimensional Filters," *Image Enhancement and Restoration*, Chap. 3, pp. 90-166. *Advances in Computer Vision and Image Processing*, Vol. 2, T.S. Huang, ed., Greenwich, CN: JAI Press, 1986.

[4] G. R. Arce and M. P. Mcloughlin, "Theoretical Analysis of Max/Median Filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* Vol. ASSP-35, No. 1, January, 1987.

[5] K. Barner, "Permutation Filters: A Group Theoretic Class of Nonlinear Filters," Ph.D. Dissertation, University of Delaware, May 1992.

[6] K. Barner and G. R. Arce, "Permutation Filters: A Class of Non-Linear Filters Based on Set Peremutations," *IEEE Transactions on Signal Processing,* Vol. ASSP-42, No. 4, April, 1994.

[7] E. J. Coyle and J.-H. Lin, "Stack Filters and the Mean Absolute Error Criterion," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-36, No. 8, pp. 1244-1254, August, 1988.

[8] E. J. Coyle, J.-H. Lin, and M. Gabbouj, "Optimal Stack Filtering and the Estimation and Structural Approaches to Image Processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-37, No. 12, pp. 2037-2066, December, 1989.

[9] N. C. Gallagher, Jr. and G. L. Wise, "A Theoretical Analysis of the Properties of Median Filters", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. ASSP-29, No. 6, December, 1981.

[10] R. C. Hardie, "Nonlinear filters for Signal Restoration and Enhnacement," Ph.D. Dissertation, University of Delaware, May 1992.

[11] R. C. Hardie and C. G. Boncelet, Jr., "LUM Filters: A Class Rank Order Based Filters for Smoothing and Sharpening," in *IEEE Transactions on Signal Processing*, May, 1993.

[12] R. C. Hardie and C. G. Boncelet, Jr., "A New Class of Order Statistic Based Filters for Smoothing and Sharpening," *Proceedings of the 1991 Conference on Information Sciences and Systems*, Baltimore, Maryland, March, 1991.

[13] B. I. Justusson, "Median Filtering: Statistical Properties," *Two-Dimensional Digital Signal Processing, II: Transforms and Median Filters*, Chap. 4, Vol. 42, pp. 161-196. *Topics in Applied Physics,* T. S. Huang, ed., New York: Springer Verlag, 1981.

[14] S.-J. Ko and Y. H. Lee, "Center Weighted Median Filters and Their Applications to Image Enhancement," *IEEE Transactions on Circuits and Systems*, Vol. 38, No. 9, pp. 984-993, September, 1991.

[15] J.-H. Lin, T. M. Sellke, and E. J. Coyle, "Adaptive Stack Filtering Under the Mean Absolute Error Criterion," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-38, No. 6, June, 1990.

[16] C. L. Mallows, “Some Theory of Nonlinear Smoothers,” *The Annals of Statistics*, Vol. 8, No. 4, pp. 695-715, 1980.

[17] A. Nieminen, P. Heinonen, Y. Neuvo, “A New Class of Detail-Preserving Filters for Image Processing,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No. 1, pp. 74–90, January, 1987.

[18] T. A. Nodes, N. C. Gallagher, Jr., “Two-Dimensional Root Structures and Convergence Properties of the Separable Median Filter,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol., 31, pp. 1350–1365, December, 1983.

[19] T. Sun, M. Gabbouj and Y. Neuvo, “Deterministic Properties of Center Weighted Median Filters,” *Proceedings of the 1992 IEEE International Conference on Communication Technology,* Tsinghua Campus, Beijing, P.R. China, September, 1992.

[20] J. W. Tukey, “Nonseparable (nonsuperposable) Methods for Smoothing Data,” in *Cong. Rec. EASCON*, pp. 673, 1974; and, *Exploratory Data Analysis*, Reading, Massachusetts: Addison Wesley, 1977.

[21] P. D. Wendt, E. J. Coyle, and N. C. Gallagher, Jr., “Stack Filters, ” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, No. 8, August, 1986.

[22] L. Yin, J. Astola and Y. Neuvo, “Adaptive Stack Filtering with Applications to Image Processing,” Submitted to *IEEE Transactions on Signal Processing*.

[23] L. Yin, Y. Neuvo, “Fast Adaptation and Performance Characteristics of FIR–WOS Hybrid Filters,” submitted to the *IEEE Transactions on Signal Processing*, December 1992.

[24] O. Yli-Harja, J. Astola and Y. Neuvo, “Analysis of the Properties of Median and Weighted Median Filters Using Threshold Logic and Stack Filter Representation,” *IEEE Transactions on Signal Processing*, Vol. 39, No. 2, February, 1991.

[25] J. Yoo, “Stack Filters: Design Algororithms and Applications,” Ph.D. Dissertation, Purdue University West Lafayette, IN, May 1993.

(a)

(b)

Figure 13: Original 512×512 8 bit/pixel grey scale images. The image "Lena" is shown in (a) and the image "Albert" is shown in (b). The image "Lena" corrupted by impulsive noise where $p = 0.20$ is shown in (c).

(c)

(a)

(b)

Figure 14: The impulse corrupted image restored using an (a) 5×5 CWM filter (b) 5×5 WOS filter (c) 5×5 RCRS filter with $M = 1$ and (d) a 5×5 RCRS filter with $M = 2$. Each of the filters has been optimized using "Albert."

(c)

(d)

(a)

(b)

Figure 15: The impulse corrupted image restored using an (a) 9×9 CWM filter (b) 9×9 WOS filter (c) RCRS filter where $M = 1$ and (d) an RCRS filter where $M = 2$. Each filter has been optimized using "Albert."

(c)

(d)