
Cost-Effective Implementation of Order-Statistics Based Vector Filters Using Minimax Approximations

M. Emre Celebi^{1,*}, Hassan A. Kingravi², Rastislav Lukac³, and Fatih Celiker⁴

¹*Dept. of Computer Science, Louisiana State University, Shreveport, LA, USA*

²*Dept. of Computer Science, Georgia Institute of Technology, Atlanta, GA, USA*

³*Epson Edge, Epson Canada Ltd., Toronto, Ontario, Canada*

⁴*Dept. of Mathematics, Wayne State University, Detroit, MI, USA*

**Corresponding author: ecelebi@lsus.edu*

Vector operators based on robust order statistics have proved successful in digital multichannel imaging applications, particularly color image filtering and enhancement, in dealing with impulsive noise while preserving edges and fine image details. These operators often have very high computational requirements which limits their use in time-critical applications. This paper introduces techniques to speed up vector filters using the minimax approximation theory. Extensive experiments on a large and diverse set of color images show that proposed approximations achieve an excellent balance among ease of implementation, accuracy, and computational speed.

© 2010 Optical Society of America

OCIS codes: 100.2000,100.2980,100.3010

1. Introduction

Image noise filtering - the process of estimating the original image information from noisy data - is a common preprocessing step in image processing and analysis applications, as the presence of noise in images not only lowers their perceptual quality, but also makes subsequent tasks such as edge detection and segmentation more difficult [1]. With the recent shift from traditional grayscale imaging to color imaging, numerous filters have been proposed for removing noise from color images. An extensive overview of color image filtering solutions and their applications can be found in [2], with detailed performance analysis presented in [3].

An important class of filters for noise reduction in color images is the one based on robust vector order statistics [4, 5]. A typical natural image exhibits strong correlation between its red, green, and blue color channels; therefore, treating the pixels of the image as vectors avoids color shifts and artifacts in the output of the filter. Since images are nonstationary due to the presence of edges as well as noise and blur introduced during the image formation, vector filters usually operate on pixels inside a supporting window that slides over the image. Desired noise filtering characteristics can be obtained by using vectors with certain ranks in the ordered set of pixel values inside the supporting window, as an ordering operation performed according to a distance or similarity criterion distinguishes outliers from noise-free samples [4].

Many researchers have noted the high computational requirements of order-statistics based vector filters; however, relatively few studies [6, 7] have focused on alleviating this problem. Furthermore, the scope of these studies is limited to the vector median filter [8], which has been considered as the gold standard of performance in color image filtering due to its robustness and excellent impulsive noise suppression capability [9].

This paper introduces techniques to speed up popular vector filters which use vector ordering criteria other than the Euclidean distance. In particular, the filtering solutions from [10–12] involve, respectively, computationally expensive inverse cosine, exponential, and logarithmic functions that are evaluated during the filtering process typically millions of times. In order to allow the use such filters in time-critical imaging applications, we utilize the minimax approximation theory to substitute the abovementioned elementary functions with computationally efficient polynomials. Extensive experiments on a large and diverse image set show that the presented approximations achieve an excellent balance among ease of implementation, accuracy, and computational speed.

The rest of the paper is organized as follows. Section 2 gives background on minimax approximation theory. Section 3 introduces the use of the minimax approximation theory in speeding up order-statistics based vector filters. Motivation and design characteristics are discussed in detail. Section 4 describes the image set, noise models, filtering performance criteria, and the experimental setup. Finally, conclusions are given in Section 5.

2. Overview of Minimax Approximation Theory

Given a function f , we would like to approximate it by another function g so that the error (ε) between them over a given interval is arbitrarily small. The existence of such approximations is stated by the following theorem:

Theorem 2.1 (Weierstrass) *Let f be a continuous real-valued function defined on $[a, b]$, i.e. $f \in C[a, b]$. Then $\forall \varepsilon > 0$ there exists a polynomial P such that $\|f - P\| < \varepsilon$, i.e. $\forall z \in [a, b], |f(z) - P(z)| < \varepsilon$.*

This is commonly known as the minimax approximation to a function. It differs from other methods, e.g. least squares approximations, in that it minimizes the maximum error (ε) rather than the average error:

$$\varepsilon = \max_{z \in [a,b]} |f(z) - P(z)|. \quad (1)$$

A similar theorem establishes the existence of a rational variant of this method [13]. Let $n \geq 0$ be a natural number and let

$$P_n([a, b]) = \{a_0 + a_1 z + \dots + a_n z^n : z \in [a, b], \ a_i \in \mathbb{R}, \ i = 0, 1, \dots, n\} \quad (2)$$

be the set of all polynomials of degree less than or equal to n . The set of irreducible rational functions, $R_m^n([a, b])$, is defined as

$$R_m^n([a, b]) = \left\{ \frac{p(z)}{q(z)} : p(z) \in P_n([a, b]), \ q(z) \in P_m([a, b]) \right\} \quad (3)$$

where p and q have no common factors. Then [13]:

Theorem 2.2 *For each function $f \in C[a, b]$, there exists at least one best rational approximation from the class $R_m^n([a, b])$.*

This theorem states the existence of a rational approximation $r^* \in R_m^n([a, b])$ to a function $f \in C[a, b]$ that is optimal in the Chebyshev sense:

$$\max_{z \in [a,b]} |f(z) - r^*(z)| = \text{dist}(f, R_m^n) \quad (4)$$

where $\text{dist}(f, R_m^n)$ denotes the distance between f and $R_m^n([a, b])$ with respect to some norm, in our case, the Chebyshev (maximum) norm. Regarding the choice between a polynomial and a rational approximant, it can be said that certain functions can be approximated more accurately by rationals than by polynomials. Jean-Michel Muller explains this phenomenon as follows “It seems quite difficult to predict if a given function will be much better approximated by rational functions than by polynomials. It makes sense to think that functions that have a behavior that is ‘highly nonpolynomial’ (finite limits at $\pm\infty$, poles, infinite derivatives, ...) will be poorly approximated by polynomials.” [14].

In this study the Remez Exchange Algorithm, an iterative method that uses Lagrangian interpolation to systematically minimize the maximum absolute difference between the given function and its polynomial approximation, was used to calculate the polynomials. The reader is referred to [13, 14] for more information on the minimax approximation theory and [15] for the implementation details of the Remez algorithm.

3. Proposed Implementations of Vector Filters

Consider an $M \times N$ red-green-blue (RGB) input image \mathbf{X} that represents a two-dimensional array of three-component vectors $\mathbf{x}(r, c) = [x_1(r, c), x_2(r, c), x_3(r, c)]$ occupying the spatial location (r, c) , with the row and column indices $r = \{1, \dots, M\}$ and $c = \{1, \dots, N\}$, respectively. In the pixel $\mathbf{x}(r, c)$, the $x_k(r, c)$ values denote the red ($k = 1$), green ($k = 2$), and blue ($k = 3$) components. In order to isolate small image regions, each of which can be treated as stationary, and reduce processing errors by operating in such a localized area of the input image, an $\sqrt{n} \times \sqrt{n}$ supporting window $W(r, c)$ centered on pixel $\mathbf{x}(r, c)$ is used. The window slides over the entire image \mathbf{X} in a raster fashion and the procedure replaces the input vector $\mathbf{x}(r, c)$ with the output vector $\mathbf{y}(r, c) = F(W(r, c))$ of a filter function $F(\cdot)$ that operates over the samples inside $W(r, c)$. Repeating the procedure for each pair (r, c) , with $r = \{1, \dots, M\}$ and $c = \{1, \dots, N\}$, produces the output vector $\mathbf{y}(r, c)$ of the $M \times N$ filtered image \mathbf{Y} . For notational simplicity, the input vectors inside $W(r, c)$ are re-indexed as a set, i.e. $W(r, c) = \{\mathbf{x}_i : i = 1, \dots, n\}$ (see Figure 1), as commonly seen in the related literature [2, 3]. In this notation, the center pixel in W is given by $\mathbf{x}_C = \mathbf{x}_{(n+1)/2}$ and in the vector $\mathbf{x}_i = [x_{i1}, x_{i2}, x_{i3}]$ with components x_{ik} , the $i \in \{1, \dots, n\}$ and $k \in \{1, 2, 3\}$ indices denote the position of the vector inside the window and the color channel, respectively.

$$\begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 \\ \mathbf{x}_4 & \mathbf{x}_5 & \mathbf{x}_6 \\ \mathbf{x}_7 & \mathbf{x}_8 & \mathbf{x}_9 \end{bmatrix}$$

Fig. 1. Indexing convention inside a 3×3 window

3.A. Vector Directional Filters

The vector directional filter (VDF) family [10] operates on the direction of the input vectors with the aim of eliminating the vectors with atypical directions. This family utilizes the angle between the input vectors to order the vectors inside the supporting window. For example, the output of the basic vector directional filter (BVDF), the most well-known member of the VDF class, is the input vector inside the supporting window whose direction is the maximum

likelihood estimate of the directions of the input vectors [16]:

$$\mathbf{y}(r, c) = \underset{\mathbf{x}_i \in W(r, c)}{\operatorname{argmin}} \left(\sum_{j=1}^n A(\mathbf{x}_i, \mathbf{x}_j) \right) \quad (5)$$

$$A(\mathbf{x}_i, \mathbf{x}_j) = \arccos \left(\frac{x_{i1}x_{j1} + x_{i2}x_{j2} + x_{i3}x_{j3}}{\|\mathbf{x}_i\|_2 \|\mathbf{x}_j\|_2} \right)$$

where $A(\mathbf{x}_i, \mathbf{x}_j)$ denotes the angle between the two input vectors \mathbf{x}_i and \mathbf{x}_j and $\|\cdot\|_2$ is the L_2 (Euclidean) norm. Note that in addition to BVDF, the angular function $A(\cdot, \cdot)$ was used in the design of a number of other filters including the generalized VDF [17], directional distance filter [18], hybrid vector filters [19], weighted VDFs [20], data-adaptive VDFs [21], and switching VDFs [22].

The computational requirements of these filters can be reduced by speeding up the inverse cosine (ARCCOS) function, whose argument falls into the interval $[0, 1]$ (see Figure 2). Unfortunately, approximating the ARCCOS function in this interval is not easy because of its behavior near 1. This can be circumvented using the following numerically more stable identity for $z \geq 0.5$:

$$\arccos(z) = 2 \arcsin \left(\sqrt{0.5(1-z)} \right) \quad (6)$$

where the inverse sine function (ARCSIN) receives its arguments from the interval $[0, 0.5]$ (see Figure 3). Instead of plugging the value of $\sqrt{0.5(1-z)}$ into a minimax approximation for the ARCSIN function and then multiplying the result by 2, two multiplication operations can be avoided if the following function is approximated:

$$\tau = \sqrt{1-z}$$

$$\arccos(z) = 2 \arcsin(\tau/\sqrt{2}) \quad (7)$$

where the argument τ falls into the interval $[0, 1/\sqrt{2}]$.

Table 1 lists the coefficients of the fourth degree minimax polynomials that approximate the ARCSIN and ARCCOS functions. Since both functions exhibit strong linearity in their respective intervals, they can be accurately approximated by polynomials, as indicated by the small error values listed in the table.

Table 1. Fourth degree minimax polynomials for the ARCSIN and ARCCOS functions

Function	ε	a_0	a_1	a_2	a_3	a_4
ARCSIN	2.097814e-05	2.097797e-05	1.412840	1.429881e-02	6.704361e-02	6.909677e-02
ARCCOS	1.048949e-05	1.570786	-9.990285e-01	-1.429899e-02	-9.481335e-02	-1.381942e-01

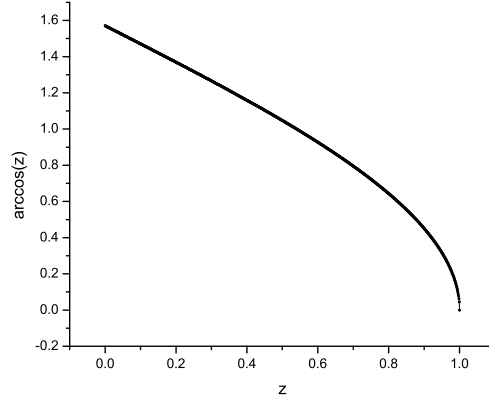


Fig. 2. Function $\arccos(z)$ in the interval $[0, 1]$

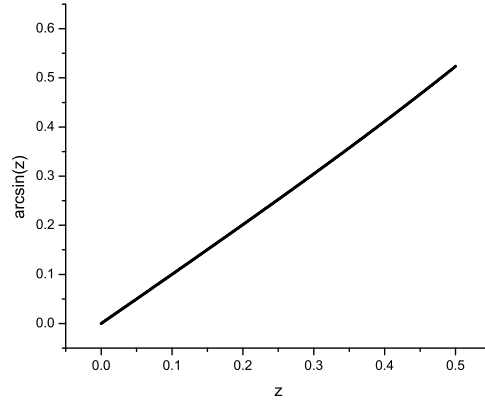


Fig. 3. Function $\arcsin(z)$ in the interval $[0, 0.5]$

3.B. Adaptive Multichannel Non-Parametric Filters

Adaptive Multichannel Non-Parametric Filters (AMNFs) [11] approach the filtering problem from an estimation theoretic perspective. Specifically, these filters employ non-parametric kernel density estimators to determine the pixels in the filtered image as follows:

$$\mathbf{y}(r, c) = \sum_{i=1}^n \mathbf{x}_i \left(\frac{h_i^{-3} K((\mathbf{x}_C - \mathbf{x}_i)/h_i)}{\sum_{j=1}^n h_j^{-3} K((\mathbf{x}_C - \mathbf{x}_j)/h_j)} \right) \quad (8)$$

$$h_i = n^{-\kappa/3} \sum_{j=1}^n \|\mathbf{x}_i - \mathbf{x}_j\|_1$$

where $\| \cdot \|_1$ denotes the L_1 (City-Block) norm. Two possible choices for the kernel function are the multivariate exponential $K(\mathbf{x}) = e^{-\|\mathbf{x}\|_1}$ (AMNFE) and the multivariate gaussian $K(\mathbf{x}) = e^{-0.5 \|\mathbf{x}\|_2^2}$ (AMNFG) functions. The scaling factor κ in the kernel width calculation is set to the author recommended value of 0.33 [11]. The computational requirements of (8) can be reduced by speeding up the kernel computation. Both kernels involve the exponential (EXP) function which can be accurately approximated by polynomials. Note that in addition to AMNFs, the EXP function was used in the design of a number of other filters including the fuzzy vector median filter [23], fuzzy vector median-rational hybrid filter [19], kernel vector median filter [24], fast adaptive noise reduction filter [25], and self-adaptive noise reduction filter [26].

The argument of the EXP function in (8) depends on the κ value, and the size and contents of a particular window. However, to obtain an accurate approximation, this argument needs to be constrained to a preferably small interval. Fortunately, for most practical purposes, we can set a cutoff point at $T = 10.0$ ($e^{-T} = 4.539993\text{e-}05$) and return 0 for arguments outside the interval $[0, T]$. Figure 4 shows a plot of the function in this interval.

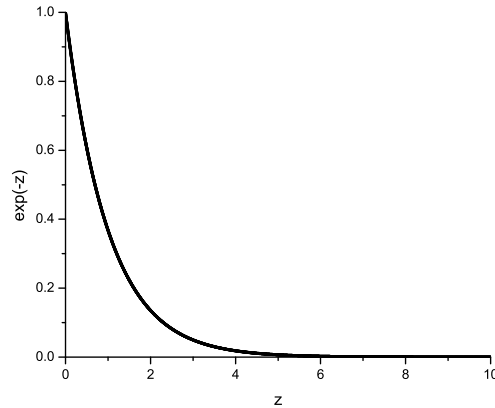


Fig. 4. Function $\exp(-z)$ in the interval $[0, 10]$

Table 2 shows the coefficients of the minimax polynomials of various degrees. Here, p and ε represent the degree of the polynomial and the error of minimax approximation, respectively. It can be seen that the error values are relatively high, and as the approximation degree is increased, the accuracy doesn't improve significantly. This suggests that rational functions might be better suited for this approximation. Table 3 lists the coefficients of a minimax rational that approximates the EXP function with an error of $\varepsilon = 2.227050\text{e-}06$.

Table 2. Minimax polynomials for the EXP function

p	ε	a_0	a_1	a_2	a_3	a_4
2	1.785517e-01	8.214528e-01	-3.186948e-01	2.544088e-02		
3	8.259345e-02	9.174126e-01	-5.631179e-01	1.015041e-01	-5.519183e-03	
4	3.337085e-02	9.666313e-01	-7.620584e-01	2.145386e-01	-2.509526e-02	1.032877e-03

Table 3. Minimax rational for the EXP function

Term	a_0	a_1	a_2	a_3	a_4
Numerator	3.206619e-02	-1.195191e-02	1.756974e-03	-1.199261e-04	3.182685e-06
Denominator	3.206627e-02	2.011147e-02	5.853684e-03	9.780143e-04	1.251598e-04

3.C. Entropy Vector Filters

Entropy vector median filter (EVMF) introduced in [12] adaptively switches between the identity operation and a noise filtering mode to improve signal-detail preserving characteristics of standard filters such as the vector median filter, which performs fixed amount of smoothing in all pixel locations. Noise filtering is performed only in pixel locations which are identified as noisy by a switching operator. This is realized by comparing an adaptive threshold β_C expressed in the form of normalized entropy to a measure of normalized local contrast P_C as follows:

$$\mathbf{y}(r, c) = \begin{cases} \underset{\mathbf{x}_i \in W(r, c)}{\operatorname{argmin}} \left(\sum_{j=1}^n \|\mathbf{x}_i - \mathbf{x}_j\|_2 \right) & P_C > \beta_C \\ \mathbf{x}(r, c) & \text{otherwise} \end{cases} \quad (9)$$

$$P_i = \frac{\|\mathbf{x}_i - \bar{\mathbf{x}}\|_2}{\sum_{j=1}^n \|\mathbf{x}_j - \bar{\mathbf{x}}\|_2}; \quad \beta_i = \frac{-P_i \log P_i}{-\sum_{j=1}^n P_j \log P_j}$$

where $C = (n + 1)/2$ and $\bar{\mathbf{x}}$ denote the linear index of the center pixel (see Figure 1) and the mean vector inside $W(r, c)$, i.e. $\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$, respectively.

Note that within the so-called generalized entropy vector filter (EVF) class [27], new filters can be designed by replacing the Euclidean distance function in (9) with some other distance or similarity measure.

The computational requirements of EVFs can be reduced by speeding up the entropy (ENT) function, whose argument falls into the interval $[0, 1]$. Although, in theory, as the argument approaches 0, the function value approaches 0, in practice, this doesn't hold as the value of the logarithm function approaches negative infinity. Therefore, as in the case of the EXP function, we set a cutoff point at $T = 0.05$ and return 0 for arguments less than T . Figure 5 shows a plot of the function in the interval $[0.05, 1]$. It can be seen that this function

is highly nonpolynomial [14], i.e. its derivatives are infinite at $z = 0$, and therefore using rational functions is more appropriate. Table 4 lists the coefficients of a minimax rational that approximates the ENT function with an error of $\varepsilon = 7.342477\text{e-}07$.

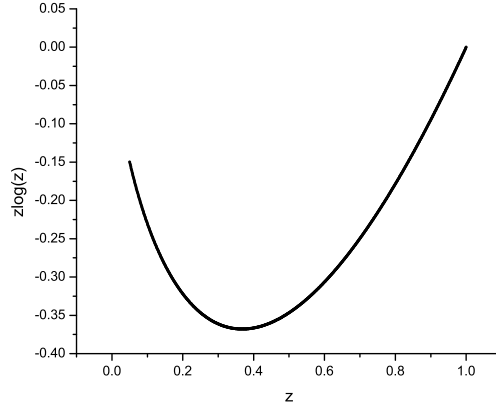


Fig. 5. Function $z\log(z)$ in the interval $[0.05, 1]$

Table 4. Minimax rational for the ENT function

Term	a_0	a_1	a_2	a_3	a_4
Numerator	-1.519742e-04	-6.835769e-02	-8.856923e-01	-5.369609e-01	1.491165
Denominator	1.532270e-02	3.987796e-01	1.461793	6.827004e-01	-4.469776e-02

4. Experimental Results

In order to evaluate the performance and robustness of the presented approximations, a set of 100 high quality RGB images was collected from the Internet. The set included images of people, animals, plants, buildings, aerial maps, man-made objects, natural scenery, paintings, sketches, as well as scientific, biomedical, synthetic, and test images commonly used in the color image processing literature.

The corruption in the test images was simulated using three noise models [28]: uncorrelated impulsive noise model, correlated impulsive noise model, and mixed noise model (Gaussian

Noise + Correlated Impulsive Noise):

Uncorrelated Impulsive Noise

$$\mathbf{x} = \{x_1, x_2, x_3\}$$

$$x_k = \begin{cases} o_k & \text{with probability } 1 - \varphi_k, \\ r_k & \text{with probability } \varphi_k \end{cases}$$

Correlated Impulsive Noise

$$\mathbf{x} = \begin{cases} \mathbf{o} & \text{with probability } 1 - \varphi, \\ \{r_1, o_2, o_3\} & \text{with probability } \varphi_1 \cdot \varphi, \\ \{o_1, r_2, o_3\} & \text{with probability } \varphi_2 \cdot \varphi, \\ \{o_1, o_2, r_3\} & \text{with probability } \varphi_3 \cdot \varphi, \\ \{r_1, r_2, r_3\} & \text{with probability } (1 - (\varphi_1 + \varphi_2 + \varphi_3)) \cdot \varphi \end{cases} \quad (10)$$

where $\mathbf{o} = \{o_1, o_2, o_3\}$ and $\mathbf{x} = \{x_1, x_2, x_3\}$ represent the original and noisy color vectors, respectively, $\mathbf{r} = \{r_1, r_2, r_3\}$ is a random vector that represents the impulsive noise, φ is the sample corruption probability, and φ_1 , φ_2 , and φ_3 are the corruption probabilities for the red, green, and blue channels, respectively. In the experiments, the channel corruption probabilities were set to 0.25.

Filtering performance was evaluated by three effectiveness criteria [3]:

1. Mean Absolute Error: $\text{MAE}(\mathbf{X}, \mathbf{Y}) = \frac{1}{3MN} \sum_{r=1}^M \sum_{c=1}^N \|\mathbf{x}(r, c) - \mathbf{y}(r, c)\|_1$
where, \mathbf{X} and \mathbf{Y} denote respectively the $M \times N$ original and filtered images in the RGB color space. MAE measures the detail preservation capability of a filter.
2. Mean Squared Error: $\text{MSE}(\mathbf{X}, \mathbf{Y}) = \frac{1}{3MN} \sum_{r=1}^M \sum_{c=1}^N \|\mathbf{x}(r, c) - \mathbf{y}(r, c)\|_2^2$
MSE measures the noise suppression capability of a filter.
3. Normalized Color Difference: $\text{NCD}(\mathbf{X}, \mathbf{Y}) = \frac{\sum_{r=1}^M \sum_{c=1}^N \|\mathbf{x}^{Lab}(r, c) - \mathbf{y}^{Lab}(r, c)\|_2}{\sum_{r=1}^M \sum_{c=1}^N \|\mathbf{x}^{Lab}(r, c)\|_2}$
where, $\mathbf{x}^{Lab}(r, c)$ and $\mathbf{y}^{Lab}(r, c)$ denote the CIEL*a*b* coordinates [1] of the pixel (r, c) in the original and filtered images, respectively. NCD measures the color preservation capability of a filter.

The efficiency of a filter was measured by execution time in seconds (Programming Language: C, Compiler: gcc 3.4.4, CPU: Intel Pentium D 2.66Ghz).

Table 5 shows the performance statistics for the three noise models. The test images were first corrupted using one of the noise models and then filtered using the exact and approximate versions of each filter. In the 'Mean' column, negative values and positive values for the MAE, MSE, and NCD indicate the percentage of filtering quality degradation and

improvement, respectively. For example, for 10% correlated impulsive noise, with respect to the MAE criterion, the approximate version of BVDF performs on the average 0.926% worse than the exact version, whereas with respect to the MSE criterion, the former performs 0.171% better than the latter. On the other hand, for the execution time criterion, positive values indicate reduction in filtering time due to the use of the presented approximations. For example, the approximate version of BVDF is on the average 1371% (or 13.71 times) faster than the exact version.

Table 5. Performance statistics at 10% noise level

		Uncorrelated Impulsive		Correlated Impulsive		Mixed	
Filter	Measure	Mean(%)	Stdev(%)	Mean(%)	Stdev(%)	Mean(%)	Stdev(%)
BVDF	MAE	-1.000	1.648	-0.926	1.558	-0.022	0.310
	MSE	-0.571	1.924	0.171	2.426	-0.131	1.103
	NCD	-0.783	1.283	-0.759	1.154	0.007	0.193
	Time	1381.783	16.576	1371.314	16.479	1408.964	11.964
AMNFE	MAE	-0.025	0.137	-0.016	0.136	-0.001	0.137
	MSE	-0.070	0.526	-0.064	0.754	-0.006	0.317
	NCD	-0.020	0.177	-0.031	0.195	-0.004	0.176
	Time	142.700	0.225	143.496	0.214	140.814	0.187
EVMF	MAE	-0.141	0.524	-0.108	0.489	0.000	0.133
	MSE	-0.376	1.732	-0.203	1.496	0.013	0.241
	NCD	-0.147	0.837	-0.149	0.625	-0.004	0.138
	Time	236.582	0.672	236.105	0.646	215.000	0.449

It can be seen that in most cases the exact filters slightly outperform their respective approximate versions. This was expected since the approximate filters necessarily involve small amounts of computational error. Nevertheless, the difference between the approximate and exact versions for each filter is negligible for most practical purposes, which demonstrates the accuracy of the presented approximations. In addition, the low standard deviation values indicate the robustness of the approximations.

The discrepancies in the speed up factors for the three filters can be attributed to the relative computational cost of the elementary functions involved. In other words, the speed up in BVDF is much greater than the other two filters because the ARCCOS function is computationally much more expensive than the EXP and ENT functions.

Since the filters presented in Section 3 are primarily intended for the removal of impulsive noise, we conducted further experiments with the most commonly used impulsive noise model [2, 3], i.e. the correlated impulsive noise model [28]. Table 6 shows the performance statistics at 20%, 30%, and 40% noise levels. It can be seen that the performance of the approximate filters does not change significantly as the noise level is increased.

Figure 6 compares the exact and approximate versions of each filter on the Lenna image. It can be seen that the presented approximations achieve substantial computational savings

Table 6. Performance statistics at higher noise levels

		20%		30%		40%	
Filter	Measure	Mean(%)	Stdev(%)	Mean(%)	Stdev(%)	Mean(%)	Stdev(%)
BVDF	MAE	-0.753	1.311	-0.626	1.070	-0.419	0.861
	MSE	0.193	2.994	0.027	2.630	-0.073	1.855
	NCD	-0.607	1.016	-0.530	0.881	-0.424	0.802
	Time	1342.074	15.480	1328.939	14.784	1301.070	14.026
AMNFE	MAE	-0.032	0.231	0.030	0.342	-0.001	0.315
	MSE	-0.125	0.945	0.021	1.172	0.006	0.812
	NCD	-0.045	0.349	0.034	0.478	0.027	0.395
	Time	143.460	0.241	145.513	0.226	145.362	0.249
EVMF	MAE	-0.030	0.676	0.013	0.534	0.067	0.476
	MSE	-0.056	1.539	0.046	1.105	0.065	0.867
	NCD	-0.004	0.829	0.026	0.633	0.088	0.547
	Time	228.346	0.551	222.356	0.431	216.354	0.423

without introducing any perceivable artifacts on the filtering results. In addition, the MAE and MSE values indicate that the filtering effectiveness of the exact and approximate filters are virtually the same.

5. Conclusions

In this article, we proposed a novel approach to speed up popular vector filters using minimax approximations. Advantages of this approach include ease of implementation, extremely good accuracy, and high computational speed. The presented approach can be adapted to other noise removal filters that involve computationally expensive mathematical functions. Finally, the given approximations have applications that go beyond color image filtering including computer graphics and computational geometry.

Acknowledgments

This publication was made possible by a grant from The Louisiana Board of Regents (LEQSF2008-11-RD-A-12).

References

1. K.N. Plataniotis and A.N. Venetsanopoulos, *Color Image Processing and Applications* (Springer, 2000).
2. R. Lukac and K.N. Plataniotis, “A Taxonomy of Color Image Filtering and Enhancement Solutions,” in *Advances in Imaging & Electron Physics (Vol. 140)*, P.W. Hawkes, ed. (Academic Press, 2006), pp. 187–264.



Fig. 6. Comparison of the exact and approximate filters on the Lenna image

3. M.E. Celebi, H.A. Kingravi, and Y.A. Aslandogan, “Nonlinear Vector Filtering for Impulsive Noise Removal from Color Images,” *Journal of Electronic Imaging* **16**(3), 033008

(2007).

4. R. Lukac, B. Smolka, K. Martin, K.N. Plataniotis, and Venetsanopoulos A.N., “Vector Filtering for Color Imaging,” *IEEE Signal Processing Magazine* **22**(1), 74–86 (2005).
5. M.E. Celebi and Y.A. Aslandogan, “Robust Switching Vector Median Filter for Impulsive Noise Removal,” *Journal of Electronic Imaging* **17**(4), 043006 (2008).
6. M. Barni, “A Fast Algorithm for 1-Norm Vector Median Filtering,” *IEEE Trans. on Image Processing* **6**(10), 1452–1455 (1997).
7. M. Barni, F. Buti, F. Bartolini, and V. Cappellini, “A Quasi-Euclidean Norm to Speed up Vector Median Filtering,” *IEEE Trans. on Image Processing* **9**(10), 1704–1709 (2000).
8. J. Astola, P. Haavisto, and Y. Neuvo, “Vector Median Filters,” *Proc. of the IEEE* **78**(4), 678–689 (1990).
9. M.E. Celebi, H.A. Kingravi, B. Uddin, and Y.A. Aslandogan, “Fast Switching Filter for Impulsive Noise Removal from Color Images,” *Journal of Imaging Science and Technology* **51**(2): 155–165 (2007).
10. P.E. Trahanias and A.N. Venetsanopoulos, “Vector Directional Filters: A New Class of Multichannel Image Processing Filters,” *IEEE Trans. on Image Processing* **2**(4), 528–534 (1993).
11. K.N. Plataniotis, D. Androutsos, S. Vinayagamoorthy, and A.N. Venetsanopoulos, “Color Image Processing Using Adaptive Multichannel Filters,” *IEEE Trans. on Image Processing* **6**(7), 933–949 (1997).
12. R. Lukac, B. Smolka, K.N. Plataniotis, and A.N. Venetsanopoulos, “Entropy Vector Median Filter,” in *Proc. of the IbPRIA Conf.*, Lecture Notes in Computer Science **2652**, pp. 1117–1125 (2003).
13. E.W. Cheney, *Introduction to Approximation Theory* (AMS, 2000).
14. J.-M. Muller, *Elementary Functions: Algorithms and Implementation* (Birkhäuser, 2006).
15. W. Fraser, “A Survey of Methods of Computing Minimax and Near-Minimax Polynomial Approximations for Functions of a Single Independent Variable,” *Journal of the ACM* **12**(3), 295–314 (1965).
16. N. Nikolaidis and I. Pitas, “Nonlinear Processing and Analysis of Angular Signals,” *IEEE Trans. on Signal Processing*, **46**(12), 3181–3194 (1998).
17. P.E. Trahanias, D. Karakos, and A.N. Venetsanopoulos, “Directional Processing of Color Images: Theory and Experimental Results,” *IEEE Trans. on Image Processing* **5**(6), 868–880 (1996).
18. D.G. Karakos and P.E. Trahanias, “Generalized Multichannel Image Filtering Structures,” *IEEE Trans. Image Processing* **6**(7), 1038–1045 (1997).
19. L. Khriji and M. Gabbouj, “Adaptive Fuzzy Order Statistics-Rational Hybrid Filters for Color Image Processing,” *Fuzzy Sets and Systems* **128**(1), 35–46 (2002).

20. R. Lukac, “Adaptive Color Image Filtering Based on Center-Weighted Vector Directional Filters,” *Multidimensional Systems and Signal Processing* **15**(2), 169–196 (2004).
21. K.N. Plataniotis, D. Androutsos, and A.N. Venetsanopoulos, “Color Image Processing Using Adaptive Vector Directional Filters,” *IEEE Trans. on Circuits and Systems-II* **45**(10), 1414–1419 (1998).
22. R. Lukac, B. Smolka, K.N. Plataniotis, and A.N. Venetsanopoulos, “Vector Sigma Filters for Noise Detection and Removal in Color Images,” *Journal of Visual Communication and Image Representation* **17**(1), 1–26 (2006).
23. K.N. Plataniotis, D. Androutsos, and A.N. Venetsanopoulos, “Adaptive Fuzzy Systems for Multichannel Signal Processing,” *Proceedings of the IEEE* **87**(9), 1601–1622 (1999).
24. B. Smolka, K.N. Plataniotis, R. Lukac, and A.N. Venetsanopoulos, “Kernel Density Estimation Based Impulsive Noise Reduction Filter,” in *Proc. of the IEEE Int. Conf. on Image Processing (ICIP’03)* **2**, pp. 137–140 (2003).
25. B. Smolka, R. Lukac, A. Chydzinski, K.N. Plataniotis, and K. Wojciechowski, “Fast Adaptive Similarity Based Impulsive Noise Reduction Filter,” *Real-Time Imaging* **9**(4), 261–276 (2003).
26. B. Smolka, K.N. Plataniotis, A. Chydzinski, M. Szczepanski, A.N. Venetsanopoulos, K. Wojciechowski, “Self-Adaptive Algorithm of Impulsive Noise Reduction in Color Images,” *Pattern Recognition* **35**(8), 1771–1784 (2002).
27. R. Lukac, B. Smolka, K.N. Plataniotis, and A.N. Venetsanopoulos, “Generalized Entropy Vector Filters,” in *Proc. of the 4th EURASIP EC-VIP-MC, Video, Image Processing and Multimedia Communications Conf.*, pp. 239–244 (2003).
28. T. Viero, K. Oistamo, and Y. Neuvo, “Three-Dimensional Median-Related Filters for Color Image Sequence Filtering,” *IEEE Trans. on Circuits and Systems for Video Technology* **4**(2), 129–142 (1994).