Impulse noise removal using polynomial approximation

Dapeng Zhang
Zhou Wang
City University of Hong Kong
Department of Computer Science
Kowloon, Hong Kong
China
E-mail: dapeng@cs.cityu.edu.hk

Abstract. A novel filtering algorithm is presented to restore images corrupted by impulsive noise. As a preprocessing procedure of the noise cancellation filter, an improved impulse detector is used to generate a binary flag image, which gives each pixel a flag indicating whether it is an impulse. This flag image has two uses: (1) a pixel is modified only when it is considered as an impulse; otherwise, it is left unchanged, and (2) only the values of the good pixels are employed as useful information by the noise cancellation filter. To remove noises from the corrupted image, we propose a new filter called a polynomial approximation (PA) filter, which is developed by modeling a local region with a polynomial that can best approximate the region under the condition of least squared error. Furthermore, an adaptive approach is introduced to automatically determine the orders of the polynomials. The proposed two kinds of PA filters, fixed-order and adaptive-order PA filters, are tested on images corrupted by both fixed-valued and random-valued impulsive noise. Major improvements are obtained in comparison with other state-of-the-art algorithms.
 [S0091-3286(98)02904-3]

Subject terms: image enhancement; image filter; impulse noise; noise detection; noise filtering; polynomial approximation.

Paper 32057 received May 30, 1997; revised manuscript received Oct. 23, 1997; accepted for publication Oct. 25, 1997.

1 Introduction

Images are often contaminated by impulse noise during transmission through communication channels. It is important to eliminate noise in the images before subsequent processing such as edge detection, image segmentation, and object recognition. A large variety of filtering algorithms have been proposed to perform effective noise cancellation while preserving the image structure.[1–12] Typical examples include the neighborhood mean filter and the median filter. Nevertheless, because these filters are implemented uniformly across the image, they tend to modify both noise pixels and undisturbed good pixels, resulting in blurring of the image. To avoid the damage of good pixels, the switching strategy was introduced by some recently published work,[1–3,9–12] where impulse detectors are employed to preselect the pixels that should be modified. This kind of technique has proved to be more effective than uniformly applied methods.

The approach we are proposing also uses an impulse detector to categorize all the pixels in the image into two classes—noise pixels and good pixels, i.e., noise-free pixels. These classification results are used in two ways during the filtering procedure. First, only pixels indexed as impulses are modified. Second, the filter uses merely the information of noise-free pixels as the source to estimate the corrupted pixel values.

To replace an impulse pixel with a new value, various methods can be developed to make an estimation. However, almost all previously published algorithms, as far as we know, replace the corrupted value with one from a local window or some linear combination of local samples. In other words, only ranking of or statistical information concerning neighboring pixel values is employed. We consider these kinds of information to be insufficient to represent the real structure, such as direction and curvature, of the local region. Substantially different from previous algorithms, our new approach is developed by modeling the local region using some function approximation algorithms, so that it can tell the structure of the local window. The center corrupted pixel is then replaced with a value that can best comply with the structural information. In particular, the polynomial approximation technique is used in the paper, which is easy to operate and is effective in modeling local regions.

This paper is organized as follows. Section 2 describes the impulse noise model assumed by our experiments and presents an iterative impulse detection algorithm that provides us with more accurate detection results than those of previously proposed methods. In Sec. 3, the polynomial approximation (PA) filter is introduced. Section 4 studies the influence of the polynomial order on the PA filter and proposes an adaptive-order polynomial approximation (AOPA) filter that can automatically determine the polynomial order by investigating some statistical features of local regions. Some further simulation and comparison results are provided in Sec. 5. Finally, a brief conclusion is drawn in Sec. 6.

2 Noise Model and Impulse Detector

2.1 *Impulse Noise Model*

Images may be contaminated by various sorts of noises. The type of noise considered by our algorithm is only im-

pulsive noise whose value is generally independent of the strength of the image signal. Let o_{ij} and x_{ij} denote the gray levels at location (i,j) of the original image and the noisy image, respectively. Then for an impulse noise model with noise probability p_n, we have

$$x_{ij}=\begin{cases} o_{ij} & \text{with probability } 1-p_n \\ n_{ij} & \text{with probability } p_n \end{cases}, \tag{1}$$

where n_{ij} is a noise value independent from o_{ij}. In this paper, two cases of noise distributions are considered. In the first case, the values of the corrupted pixels are equal to the maximum or minimum of the allowed dynamic range with equal probability. This kind of noise is commonly referred to as salt-and-pepper noise. In the other case, the corrupted pixel values are uniformly distributed between the maximum and minimum allowed dynamic range. For 8-bpp (bits/pixel) gray-level images, the noise luminance of the first case corresponds to a fixed value of 0 or 255 with equal probability, while that of the second case corresponds to a random value uniformly distributed between 0 and 255.

2.2 *Impulse Detector*

Our impulse detection algorithm is developed based on two assumptions: (1) a noise-free image should be locally smoothly varying and is separated by edges[3] and (2) a noise pixel takes a gray value substantially larger than or smaller than those of its neighbors.

Sun and Neuvo introduced a simple and effective method to detect noises in their switch I scheme.[3] We briefly describe it as follows. Let x_{ij} represents the pixel values at position (i,j) in the corrupted image. To judge whether x_{ij} is an impulse pixel, we find the median value of the samples in the $(2L_d+1)\times(2L_d+1)$ window centered about it, i.e.,

$$m_{ij}=\text{med}\,\{x_{i-L_d,j-L_d},\ldots,x_{ij},\ldots,x_{i+L_d,j+L_d}\}. \tag{2}$$

The difference between m_{ij} and x_{ij} is used to detect impulses:

$$f_{ij}=\begin{cases} 1 & \text{if } |x_{ij}-m_{ij}|\geq T_d \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

Here T_d is a predefined threshold value. The binary value f_{ij} indicates whether (i,j) is considered as an impulse, i.e., $f_{ij}=1$ means (i,j) is a corrupted pixel; otherwise, (i,j) is noise free.

This detection method is adopted by our algorithm. In addition, we also developed a modified version to improve detection accuracy. The modified version is implemented in an iterative manner where the noisy pixels detected in the current iteration are modified and used to help the detection of other noisy pixels in the subsequent iterations. A main advantage of such an iterative procedure is that some impulse pixels located in the middle of large noise blotches can also be properly detected and filtered. Let $x_{ij}^{(0)}$ denote the pixel value at position (i,j) in the initial noisy image and let $x_{ij}^{(n)}$ represent the pixel value at position (i,j) in the image after the n'th iteration. In the n'th iteration ($n=1,2,\ldots$), for each pixel $x_{ij}^{(n-1)}$, we also first find the median value of the samples in a $(2L_d+1)\times(2L_d+1)$ window centered about it:

$$m_{ij}^{(n-1)}=\text{med}\,\{x_{i-L_d,j-L_d}^{(n-1)},\ldots,x_{ij}^{(n-1)},\ldots,x_{i+L_d,j+L_d}^{(n-1)}\}. \tag{4}$$

The difference between $m_{ij}^{(n-1)}$ and $x_{ij}^{(n-1)}$ is the criterion to determine whether $x_{ij}^{(n-1)}$ should be changed:

$$x_{ij}^{(n)}=\begin{cases} m_{ij}^{(n-1)} & \text{if } |x_{ij}^{(n-1)}-m_{ij}^{(n-1)}|\geq T_d \\ x_{ij}^{(n-1)} & \text{otherwise} \end{cases}. \tag{5}$$

Suppose the iteration stops after the N_d'th iteration. The flag value f_{ij} is then given as:

$$f_{ij}=\begin{cases} 1 & \text{if } x_{ij}^{(0)}\neq x_{ij}^{(N_d)} \\ 0 & \text{otherwise} \end{cases}. \tag{6}$$

Before the real implementation of our impulse detector, the parameters L_d, T_d, and N_d should be predetermined. In our simulations, a 3×3 sized window always gives better results than larger windows, so we choose $L_d=1$. However, it is not as easy to select T_d and N_d, because they are sensitive to the type of the noise. Figures 1 and 2 show typical examples of the correct detection rates as functions of T_d and N_d for fixed-valued and random-valued impulsive noise, respectively. Here, if E is the number of good pixels that are correctly detected as good pixels, F is the number of noise pixels that are correctly detected as noise pixels, and G is the total number of pixels in the image, then the detection correct rate is calculated as $(E+F)/G$. In Figs. 1 and 2, it appears that the best results can be obtained from two or three iterations when the images are corrupted by fixed-valued noise, while for the case of random-valued noise, no iteration is needed, i.e., $N_d=1$ is the best. It can be also observed that the best T_d for fixed-valued noise is larger than that for random-valued noise. According to these facts, we always use $T_d=35$ and $N_d=3$ for fixed-valued impulse noise and $T_d=20$ and $N_d=1$ (i.e., no iteration) for random-valued impulse noise in the remaining part of this paper.

3 PA Filter

Consider a $(2L_f+1)\times(2L_f+1)$ square region centered about an impulse pixel (i_0,j_0) with $f_{i_0,j_0}=1$. The pixels in the region can be categorized into two classes—good pixels $(f_{ij}=0)$ and impulse pixels $(f_{ij}=1)$. Only good pixels are used by the PA filter, which models the local region using a 2-D polynomial. For simplicity, we shift the position of the center pixel (i_0,j_0) to (0,0) and normalize the square region so that its top-left and bottom-right corners are located at $(-1,-1)$ and $(+1,+1)$, respectively:

$$y_{pq}=x_{ij}, \tag{7}$$

where y_{pq} is the pixel value under the new coordinate and we have

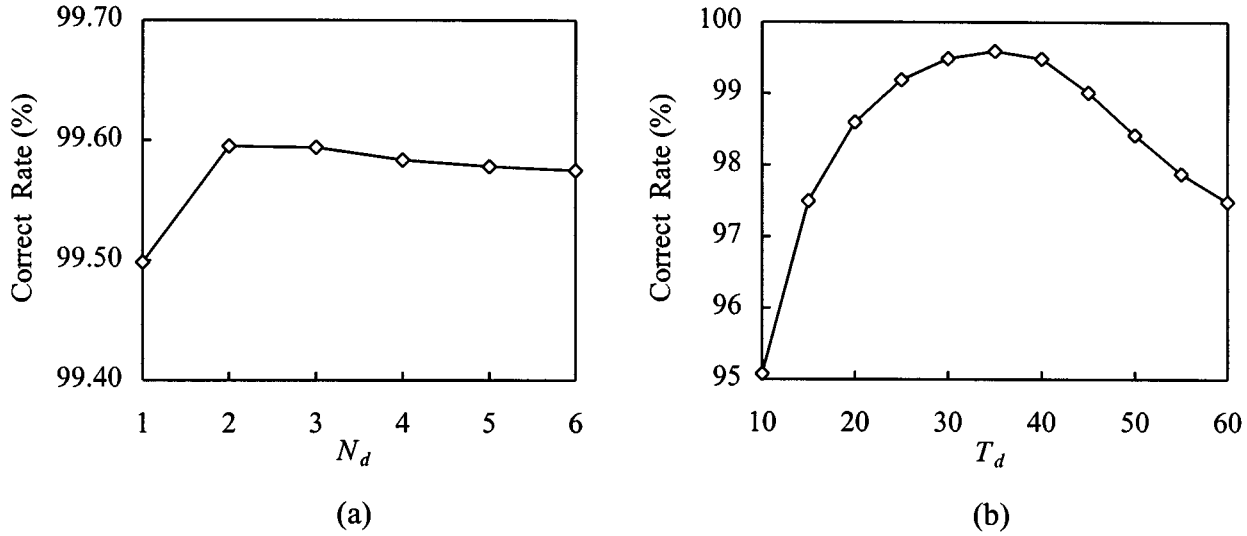


Fig. 1 Example of the influence of detection parameters on the detection correct rate for fixed-valued impulse noise. The test image is ''Lena'' corrupted by 20% noise. (a) Correct rate as a function of N_d, where T_d is a fixed value of 35, and (b) correct rate as a function of T_d, where N_d is a fixed value of 3.

$$p=(i-i_0)/L_f \quad q=(j-j_0)/L_f. \tag{8}$$

Then the approximation polynomial can be denoted as

$$\hat{y}_{pq}=\sum_{m=0}^{K}\sum_{n=0}^{K-m} c_{mn}p^m q^n, \tag{9}$$

where $\{c_{mn}|m=0,\ldots,K;\ n=0,\ldots,m-K\}$ is a set of coefficients, and K is the order of the polynomial. For example, a two-order polynomial can be written as:

$$\hat{y}_{pq}=c_{00}+c_{10}p+c_{01}q+c_{20}p^2+c_{02}q^2+c_{11}pq. \tag{10}$$

The calculated value of $\hat{y}_{pq}$ can be used to estimate the value of x_{ij}:

$$\hat{x}_{ij}=\hat{y}_{pq}, \tag{11}$$

where

$$i=i_0+pL_f \quad j=j_0+qL_f. \tag{12}$$

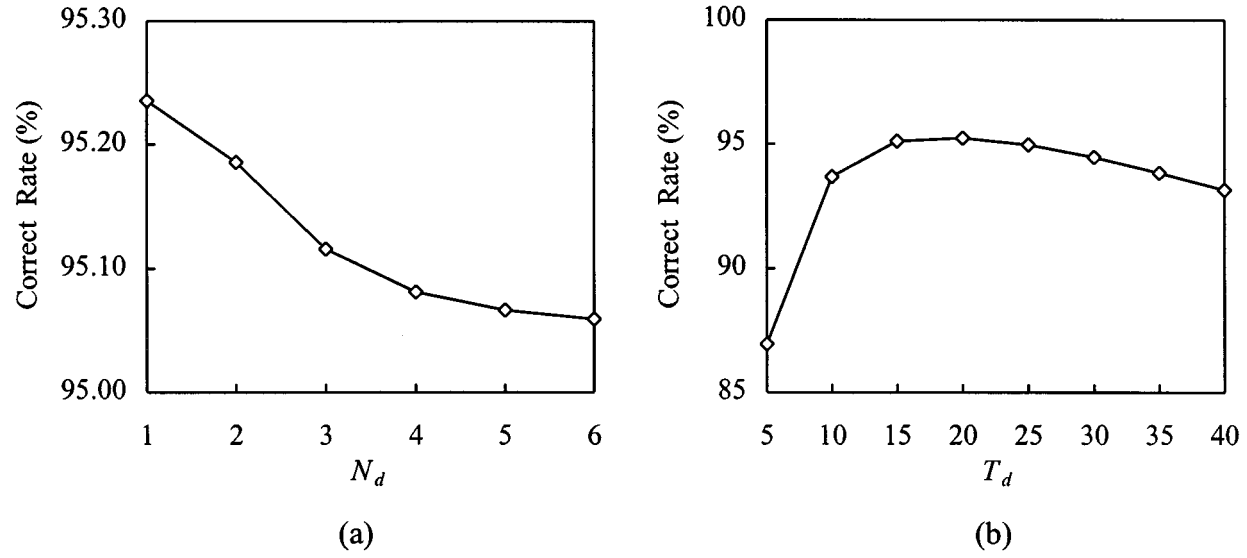


Fig. 2 Example of the influence of detection parameters on the detection correct rate for random-valued impulse noise. The test image is ''Lena'' corrupted by 20% noise. (a) Correct rate as a function of N_d, where T_d is a fixed value of 20, and (b) correct rate as a function of T_d, where N_d is a fixed value of 1.

Table 1 Filtering results in PSNR for ''Lena'' corrupted with fixed-valued impulse noise.

	Percentage of Fixed-Valued Impulse Noise				
Filtering Algorithm	10%	15%	20%	25%	30%
Zero-order PA filter	36.60 dB	35.64 dB	34.65 dB	33.66 dB	32.31 dB
One-order PA filter	36.64 dB	35.65 dB	34.71 dB	33.69 dB	32.36 dB
Two-order PA filter	39.48 dB	38.61 dB	37.44 dB	**36.26** dB	**34.22** dB
Three-order PA filter	**39.63** dB	**38.79** dB	**37.56** dB	35.74 dB	33.03 dB
AOPA filter	39.60 dB	38.79 dB	37.57 dB	36.27 dB	34.21 dB
	−0.03 dB	0.00 dB	+0.01 dB	+0.01 dB	−0.01 dB

The set of coefficients $\{c_{mn}\}$ should be well selected so that the good pixels in the local region are best approximated under the condition of the least squared error. The squared error is calculated as

$$E=\sum_{i=-L_f}^{L_f}\sum_{j=-L_f}^{L_f}(1-f_{ij})(x_{ij}-\hat{x}_{ij})^2. \tag{13}$$

The minimum of E is achieved when the deviation of E with respect to every c_{mn} are all 0:

$$\partial E/\partial c_{mn}=0 \quad (m=0,\ldots,K; \quad n=0,\ldots,K-m). \tag{14}$$

By solving these equations, the value of each c_{mn} can be obtained. Finally, the center pixel $x_{i_0j_0}$ is replaced by the estimated value:

$$\hat{x}_{i_0j_0}=\hat{y}_{00}=c_{00}. \tag{15}$$

As a special case, the zero-order PA filter has only one coefficient c_{00} and the solution of Eq. (14) is simply

$$c_{00}=\frac{\Sigma_{i=-L_f}^{L_f}\Sigma_{j=-L_f}^{L_f}(1-f_{ij})x_{ij}}{\Sigma_{i=-L_f}^{L_f}\Sigma_{j=-L_f}^{L_f}(1-f_{ij})}, \tag{16}$$

that is, c_{00} equals the average value of the good pixels in the square region. Therefore, the zero-order PA filter can also be viewed as a noise-free neighborhood mean filter.

4 AOPA Filter

4.1 *Study of Polynomial Order*

Only two parameters, L_f and K, should be predefined before the application of the PA filter. Our simulations indicate that in most cases, a 5×5 sized square region is the best for the performance of the polynomial approximation algorithm, so we set $L_f=2$. Determining the polynomial order K is not an easy task. The proper value of K depends on how many details are embraced in the local region and how much the region is corrupted. In this subsection, we investigate the performance of a fixed-order PA (FOPA) filter, where the value of K is invariable throughout the whole image. Then in the next subsection, an AOPA filter is introduced, where K is adjusted to comply with some local statistical features. In all the experiments, the test images are 512×512, 8-bpp gray-level images. Peak signal-to-noise ratio (PSNR) is used to assess the filtering results:

$$\mathrm{PSNR}=10\log_{10}\frac{255^2}{(1/r^2)\Sigma_{i=1}^{r}\Sigma_{j=1}^{r}(o_{ij}-t_{ij})^2}, \tag{17}$$

where r is the size of the image ($r=512$), and o_{ij} and t_{ij} are the pixel values at position (i,j) within the original image and the test image, respectively.

In Tables 1 and 2, we list the filtering results of zero-, one-, two-, and three-order PA filters, where the original image ''Lena'' is corrupted with 10 to 30% fixed-valued and random-valued impulse noise, respectively. Basically, higher polynomial order, such as two- or three-order, leads

Table 2 Filtering results in PSNR for ''Lena'' corrupted with random-valued impulse noise.

	Percentage of Random-Valued Impulse Noise				
Filtering Algorithm	10%	15%	20%	25%	30%
Zero-order PA filter	34.41 dB	33.34 dB	32.39 dB	31.41 dB	30.25 dB
One-order PA Filter	34.44 dB	33.38 dB	32.38 dB	31.41 dB	30.23 dB
Two-order PA filter	36.68 dB	35.30 dB	**33.99** dB	**32.59** dB	**30.87** dB
Three-order PA filter	**36.80** dB	**35.32** dB	33.75 dB	32.12 dB	29.67 dB
AOPA filter	36.79 dB	35.36 dB	34.00 dB	32.57 dB	30.86 dB
	−0.01 dB	+0.04 dB	+0.01 dB	−0.02 dB	−0.01 dB

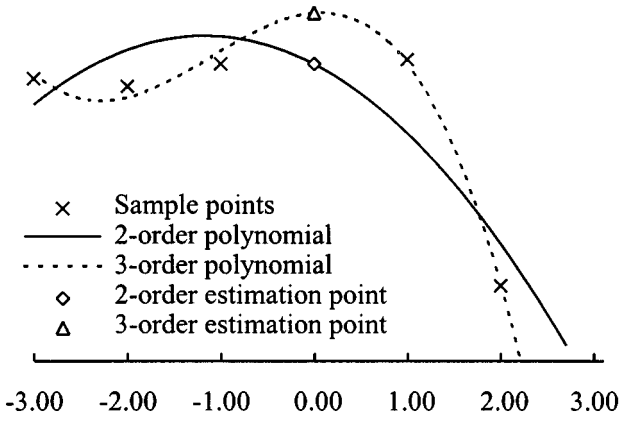


Fig. 3 Example that demonstrates the approximation and estimation functions of two- and three-order polynomials.

to higher PSNR. The reason is simply because higher order polynomials can reflect more complex characteristics, such as direction and curvature, of local regions. However, this is not always true, especially when the noise rates are high. For instance, the PSNR performance of the two-order PA filter for 30% random-valued impulse noise is 30.87 dB, which is substantially higher than 29.67 dB of the three-order PA filter. We think the main reason is that low-order polynomials are more robust than high-order ones. To demonstrate it clearly, we use a simple 1-D example (see Fig. 3), where both a two- and a three-order polynomial are used to estimate the point at 0.00 using the five sample points provided. Both of the polynomials are obtained under the condition of least squared error. It can be observed that although the three-order polynomial curve performs better in approximating the sample points, i.e., the curve is closer to the sample points, the estimated value for the point at 0.00 seems too high, not as good as that estimated by the two-order polynomial. This phenomenon is sometimes called overmatching. Another drawback of a high-order PA is that it leads to a high computation burden, which is not justified, especially when dealing with very smooth regions. For such regions, a zero- or one-order polynomial is enough to give a good approximation within much less time.

4.2 *AOPA Filter*

For a certain region in the image, an appropriate choice of the polynomial order is determined mainly by two factors. The first is how many details the region contains and the second is how much the region is contaminated. Our AOPA filter is developed according to some quantitative evaluations of these two factors.

Before the description of the AOPA filter, however, we first define a detail estimation value $d_{i_0j_0}$ for each good pixel in the image. At (i_0,j_0) with $f_{i_0j_0}=0$, we have

$$d_{i_0j_0}=x_{i_0j_0}-\frac{1}{M}\sum_{i=i_0-1}^{i_0+1}\sum_{j=j_0-1}^{j_0+1}(1-f_{ij})(1-\delta_{i_0j_0})x_{ij}, \quad (18)$$

where

$$\delta_{i_0j_0}=\begin{cases}1 & \text{if } i=i_0 \text{ and } j=j_0\\ 0 & \text{otherwise}\end{cases}. \quad (19)$$

The calculation of $d_{i_0j_0}$ is equivalent to applying the following 3×3 mask to the image:

$$\begin{bmatrix} -\frac{1-f_{i_0-1,j_0-1}}{M} & -\frac{1-f_{i_0-1,j_0}}{M} & -\frac{1-f_{i_0-1,j_0+1}}{M}\\ -\frac{1-f_{i_0,j_0-1}}{M} & 1 & -\frac{1-f_{i_0,j_0+1}}{M}\\ -\frac{1-f_{i_0+1,j_0-1}}{M} & \frac{1-f_{i_0+1,j_0}}{M} & -\frac{1-f_{i_0+1,j_0+1}}{M}\end{bmatrix}, \quad (20)$$

where

$$M=8-\sum_{i=i_0-1}^{i_0+1}\sum_{j=j_0-1}^{j_0+1}f_{ij}. \quad (21)$$

Actually, such a mask can be viewed as a revised version of one of the Laplacian masks[1] that are useful in detecting lines, line ends, and points over edges. The only difference is that it considers merely good neighboring pixels with $f_{ij}=0$. In other words, the typical Laplacian mask is a special case of our revised version, where all $f_{ij}=0$. Our new mask, which we call a noise-free Laplacian filter, is capable of evaluating how many details a certain area of the image contains even under the condition that impulse noises exist in the image.

Now we describe how our AOPA filter determines the polynomial order. For an impulse pixel at position (i_0,j_0), we still use the information within a $(2L_f+1)\times(2L_f+1)$ window centered about it as the source to give an estimation on the pixel. First, we count how many good pixels are in the window:

$$R=\sum_{i=i_0-L_f}^{i_0+L_f}\sum_{j=j_0-L_f}^{j_0+L_f}(1-f_{ij}). \quad (22)$$

Second, a quantitative estimation of how much the local region is corrupted is given as

$$u=\frac{R}{(2L_f+1)^2}. \quad (23)$$

Next, a rough estimation of how many details the local region contains is calculated as follows:

$$v=\frac{1}{R}\sum_{i=i_0-L_f}^{i_0+L_f}\sum_{j=j_0-L_f}^{j_0+L_f}(1-f_{ij})|d_{ij}|. \quad (24)$$

Note that only good pixels in the window are considered. Our adaptive algorithm decides the order of the polynomial by combining the influences of both of these two factors u and v. The synthesized value is defined as

$$s=u^a v^b, \quad (25)$$

where the use of a and b is to offer us with the flexibility to adjust the weights of u and v. Finally, the polynomial order is provided as

$$\hat{y}_a = \begin{cases} \hat{y}_0 & \text{if } 0 \leqslant s \leqslant T_{s0} \\ \hat{y}_1 & \text{if } T_{s0} < s \leqslant T_{s1} \\ \hat{y}_2 & \text{if } T_{s1} < s \leqslant T_{s2} \\ \hat{y}_3 & \text{if } T_{s2} < s \end{cases}, \tag{26}$$

where $\hat{y}_a$, $\hat{y}_0$, $\hat{y}_1$, $\hat{y}_2$, and $\hat{y}_3$ denote the estimated value of the AOPA, zero-order PA, one-order PA, two-order PA, and three-order PA techniques, respectively, and T_{s0}, T_{s1}, and T_{s2} are threshold values that categorize $\hat{y}_a$ into one of four classes.

5 Simulations and Comparisons

To assess the performance of our algorithms, some computer simulations are carried out on several standard gray-level images. In all the experiments concerning the AOPA filter, the parameters are selected as follows:

$$a=3,\ b=1,\quad T_{s0}=0.5,\quad T_{s1}=0.8,\quad \text{and}\quad T_{s2}=3.0. \tag{27}$$

Tables 1 and 2 give the PSNR performance of zero-order PA, one-order PA, two-order PA, three-order PA, and AOPA filters, where the original image ''Lena'' is corrupted with 10 to 30% fixed-valued and random-valued impulse noise, respectively. In each case, the best results among the zero-, one-, and three-order PA filters are printed in bold. When the noise rate is low, the best result is usually obtained by the three-order PA filter. However, with the increase of noise rate, the two-order PA filter gradually outperforms three-order PA filter. The performance of AOPA filter traces the best result of the four FOPA filters. Its PSNR is sometimes slightly higher or sometimes slightly lower than the best of the four. The last rows of the two tables present the differences of the PSNR performance between the AOPA filter and the best of the four FOPA filters. The absolute values of the differences are always very close to 0, ranging from 0.00 to 0.04 dB. Obviously, we prefer the AOPA filter in real applications because for a given corrupted image, it is difficult for us to predetermine which of the four FOPA filters is the best. In addition, in comparison with high-order FOPA filters, the AOPA filter may, in some way, save computing time because some of the regions are approximated by low-order polynomials. For example, for the case that the image ''Lena'' is corrupted by 30% fixed-valued impulse noise, 32.19% of the noise pixels are filtered by the zero-order PA method, 23.05% by the one-order PA method, 38.71% by the two-order PA method, and only 6.04% are filtered by the three-order PA method. Since the zero- and one-order PA methods perform much faster than the two- and three-order PA techniques, a lot of time can be saved by the AOPA filter in comparison with pure two- or three-order PA filters. Note that even though most of the noise pixels are not filtered by the two-order PA technique, the PSNR performance of the AOPA filter is very close to that of the two-order PA filter, which is significantly better than the other three FOPA filters. In Fig. 4, we compare the performance speed of different PA filters, where the AOPA filter uses some extra time to classify each impulse pixel to be filtered by one of the four PA techniques. It appears that a three-order PA filter always requires much more time than other PA filters. The computing time of AOPA filter is close to that of the two-order PA filter. When the noise rate is low, the pure two-order PA filter spends less time. We think that such a small amount of extra time is worthwhile because the PSNR performance of AOPA filter is better than that of the pure two-order PA filter (see Table 2). When noise rate is high, the two-order PA filter requires more time than the AOPA filter. In such a case, the AOPA and the pure two-order PA filters provide almost the same PSNR results, while AOPA filter is better at saving time.

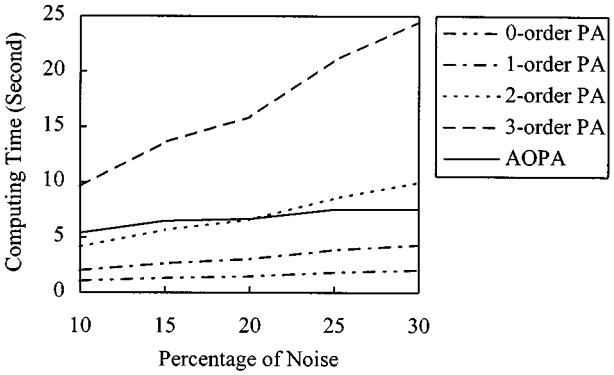


Fig. 4 Comparison of computing time for various PA filters. The test image ''Lena'' is corrupted by fixed-valued impulse noise with a noise rate ranging from 10 to 30%. The simulations are conducted on a Pentium 166M PC computer.

To demonstrate the visual quality of the filtering results, we show an enlarged area of ''Lena'' in Fig. 5, where the image is corrupted by 20% fixed-valued impulse noise. It

Fig. 5 (a) Enlarged area of ''Lena'' corrupted by 20% fixed-valued impulse noise, (b) restoration result by the one-order PA filter, (c) restoration result by the AOPA filter, and (d) the original image area.

Table 3 Comparative restoration results in PSNR for 20% impulse noise for image ''Lena.''

Filtering Algorithm	Fixed-Value Impulses (dB)	Random-Valued Impulses (dB)
Median filter (3×3)	28.57	29.76
Median filter (5×5)	28.78	28.59
Median filter with adaptive length[2]	30.57	31.18
Rank-conditioned rank selection filter[5]	31.36	30.78
Switch I median filter[3]	31.97	31.34
Switch II median filter[3]	29.96	32.04
Abreu et al.[10] ($M=1296$) (inside training set)	35.70	33.37
Fuzzy approach[11]	36.47	33.78
Zero-order PA filter	34.65	32.39
One-order PA filter	34.71	32.38
Two-order PA filter	37.44	33.99
Three-order PA filter	37.56	33.75
AOPA filter	37.57	34.00

For fixed-valued impulse noise, impulses take on only the values 0 or 255 with equal probability. For random-valued impulse noise, impulse values are uniformly distributed between 0 and 255. See Refs. 10 and 11 for parameter selection schemes.

can be observed that the one-order PA filter can give only a rough approximation of the corrupted pixels, while the AOPA filter provides much more detail.

In Table 3, we compare our PA filter with other state-of-the-art algorithms. Abreu et al. reported many restoration results in PSNR for images corrupted by both 20% fixed-valued and random-valued impulse noises.[10] Some filtering results of a fuzzy approach developed by us are also provided.[11] We list some of these data and add the PSNR performance of our FOPA and AOPA filters to the table. It can be observed that for the case of fixed-valued impulse noise, the two-order PA, three-order PA, and AOPA filters provide significant improvement over all the other approaches, while for the case of random-valued impulse noise, the two-order PA, three-order PA, and AOPA filters are also the best and only the fuzzy approach[11] can compete with them. In Figs. 6 and 7, we show some restored images obtained by different filtering methods which are the typical 3×3 and 5×5 median filter, the switch I median filter,[3] and our AOPA filter. In Fig. 6, the test image ''Bridge'' is corrupted by 30% random-valued impulse noise, while in Fig. 7, the test image ''Peppers'' is corrupted by 30% fixed-valued impulse noise. With a small window size of 3×3, the typical median filter misses many impulse pixels remaining in the image. When larger window size such as 5×5 is applied, almost all the impulses are removed, but many good pixels are also modified, resulting in blurring of the image. The switch I median filter can well preserve good pixels while eliminating noise pixels, but still many impulses remained unaltered. Dramatic restoration results are obtained by the AOPA filter. It can remove almost all of the noise pixels while preserve image details very well. Note also that although the parameters are optimized for the ''Lena'' image, good restoration results are still obtained for different types of images such as ''Peppers'' and ''Bridge'' under different occurrence rates of the impulse noise.

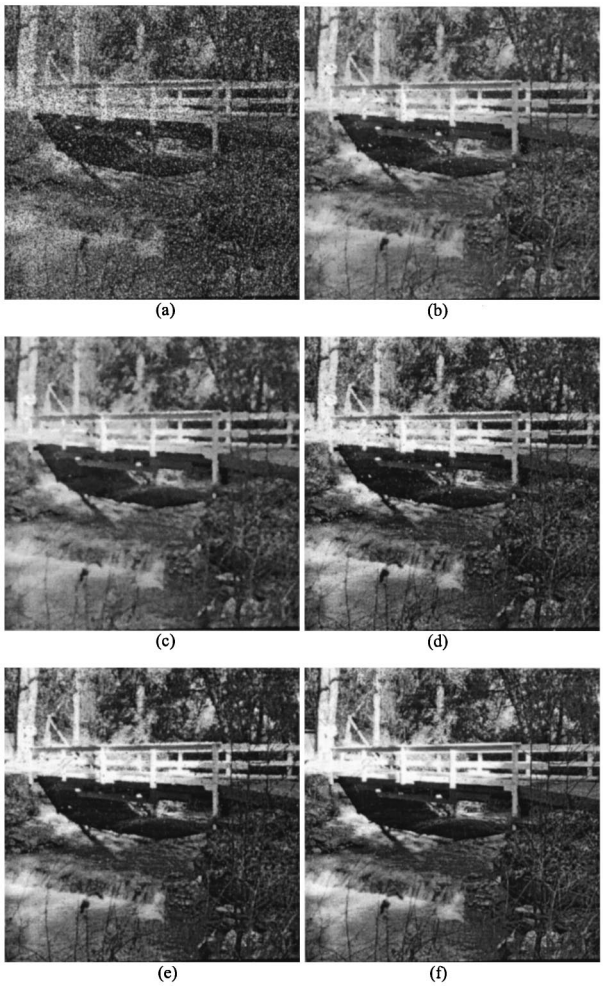


Fig. 6 Comparative restoration results for ''Bridge'' corrupted by 30% random-valued impulse noise: (a) the corrupted image, (b) image restored by the 3×3 median filter, (c) image restored by 5×5 median filter, (d) image restored by the switch I scheme, (e) image restored by the AOPA filter, and (f) the original image of ''Bridge.''

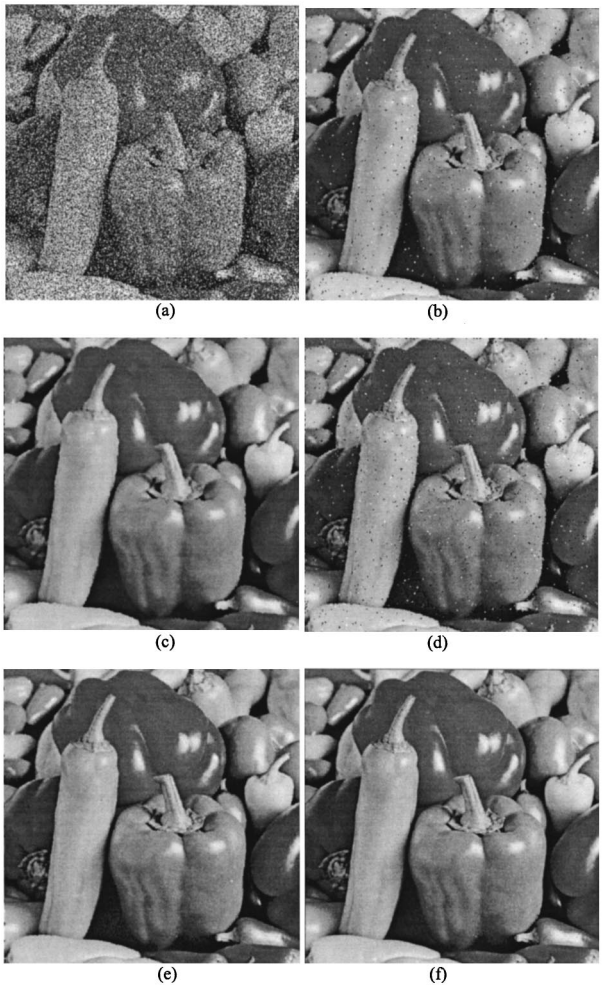


Fig. 7 Comparative restoration results for ''Peppers'' corrupted by 30% fixed-valued impulse noise: (a) the corrupted image, (b) image restored by the 3×3 median filter, (c) image restored by the 5×5 median filter, (d) image restored by the switch I scheme, (e) image restored by the AOPA filter, and (f) the original image of ''Peppers.''

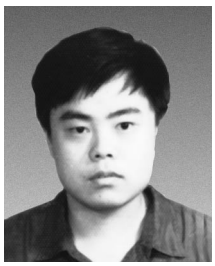
6 Conclusion

In this paper, a new impulse noise removal approach is introduced that is developed using impulse detection and polynomial approximation techniques. The proposed impulse replacement method breaks through the traditional framework in that it is not designed by employing merely the ranking or statistical information and replacing the corrupted value with one from the local window or some linear combination of local samples, but is implemented by modeling the local region using a polynomial, which is more powerful in representing the real structure of the region. An adaptive method is also presented that can automatically give an appropriate polynomial order for a local region. It has been proved to be very successful in obtaining good restoration results while saving computation time. Simulation results show that the proposed approach significantly outperforms many well-known techniques.

References

1. S. T. Bow, *Pattern Recognition and Image Processing*, Marcel Dekker, New York (1992).
2. H. Lin and A. N. Willson, ''Median filter with adaptive length,'' *IEEE Trans. Circ. Syst.* **35**(6), 675–690 (1988).
3. T. Sun and Y. Neuvo, ''Detail-preserving median based filters in image processing,'' *Pattern Recog. Lett.* **15**, 341–347 (1994).
4. R. C. Hardie and C. G. Boncelet, ''LUM filters: a class of rank-order-based filters for smoothing and sharpening,'' *IEEE Trans. Signal Process.* **41**(3), 1061–1076 (1993).
5. R. C. Hardie and K. E. Barner, ''Rank conditioned rank selection filters for signal restoration,'' *IEEE Trans. Image Process.* **3**(2), 192–206 (1994).
6. T. Sun, M. Gabbouj, and Y. Neuvo, ''Center weighted median filters: some properties and their applications in image processing,'' *Signal Process.* **35**, 213–229 (1994).
7. G. Ramponi, ''The rational filter for image smoothing,'' *IEEE Signal Process. Lett.* **3**(3), 63–65 (1996).
8. F. Russo and G. Ramponi, ''A fuzzy filter for images corrupted by impulse noise,'' *IEEE Signal Process. Lett.* **3**(6), 168–170 (1996).
9. L. García-Cabrera, M. J. García-Salinas, P. L. Luque-Escamilla, J. Martínez-Aroza, J. F. Gómez-Lopera, and R. Román-Roldán, ''Median-type filters with model-based preselection masks,'' *Image Vis. Comput.* **14**, 741–752 (1996).
10. E. Abreu, M. Lightstone, S. K. Mitra, and K. Arakawa, ''A new efficient approach for the removal of impulse noise from highly corrupted images,'' *IEEE Trans. Image Process.* **5**(6), 1012–1025 (1996).
11. D. Zhang and Z. Wang, ''Impulse noise detection and removal using fuzzy techniques,'' *Electron. Lett.* **33**, 378–379 (1997).
12. Z. Wang and D. Zhang, ''Restoration of impulse noise corrupted images using long-range correlation,'' *IEEE Signal Process. Lett.* **5**(1), 4–6 (1998).

Dapeng Zhang graduated in computer science from Peking University in 1974 and received his MSc and PhD degrees in computer science and engineering from Harbin Institute of Technology in 1983 and 1985, respectively. From 1986 to 1988, he was a postdoctoral fellow at Tsinghua University and became an associate professor at Academia Sinica, Beijing, China. In 1988, he joined the University of Windsor, Ontario, Canada, as a visiting research fellow in electrical engineering. He received his second PhD in electrical and computer engineering from the University of Waterloo, Ontario, Canada, in 1994. Currently, he is an associate professor with the City University of Hong Kong. His research interests include automated biometric-based identification, neural systems and applications, very large scale integration system design methodology, parallel computer architecture, image processing, and pattern recognition. He has authored and coauthored about 100 papers and two books and has received several project awards. Dr. Zhang is a senior member of the IEEE.



Zhou Wang received his BE in information engineering with a minor in applied mathematics from Huazhong University of Science and Technology, Wuhan, China, in 1993 and his MS in communication and electronic engineering from South China University of Technology, Guangzhou, China in 1995. He is currently with the Department of Electronic and Communication Engineering, South China University of Technology, Guangzhou, and the Department of Computer Science, City University of Hong Kong. His current interests include image coding and image processing.