Title	Performance evaluation of a feature-preserving filtering algorithm for removing additive random noise in digital images
Author(s)	Yung, NHC; Lai, AHS
Citation	Optical Engineering, 1996, v. 35 n. 7, p. 1871-1885
Issued Date	1996
URL	http://hdl.handle.net/10722/44730
Rights	Creative Commons: Attribution 3.0 Hong Kong License

Performance evaluation of a feature-preserving filtering algorithm for removing additive random noise in digital images

Nelson H. C. Yung
Andrew H. S. Lai, MEMBER SPIE
The University of Hong Kong
Department of Electrical
and Electronic Engineering
Pokfulam Road
Hong Kong
E-mail: nyung@hkueee.hku.hk

Abstract. We evaluate the performance of a feature-preserving filtering algorithm over a range of images corrupted by typical additive random noise against three common spatial filter algorithms: median, sigma and averaging. The concept of the new algorithm is based on a corrupted-pixel identification methodology over a variable subimage size. Rather than processing every pixel indiscriminately in a digital image, this corrupted-pixel identification algorithm interrogates the image in variable-sized subimage regions to determine which are the corrupted pixels and which are not. As a result, only the corrupted pixels are being filtered, whereas the uncorrupted pixels are untouched. Extensive evaluation of the algorithm over a large number of noisy images shows that the corrupted-pixel identification algorithm exhibits three major characteristics. First, its ability in removing additive random noise is better visually (subjective) and has the smallest mean-square errors (objective) in all cases compared with the median filter, averaging filter and sigma filter. Second, the effect of smoothing introduced by the new filter is minimal. In other words, most edge and line sharpness is preserved. Third, the corrupted-pixel identification algorithm is consistently faster than the median and sigma filters in all our test cases.

Subject terms: noise removal; feature preserving filtering; corrupted-pixel identification; impulse noise; Gaussian white noise; mean-square error.

Paper 24075 received July 19, 1995; revised manuscript received Nov. 28, 1995; accepted for publication Nov. 28, 1995.

1 Introduction

1.1 *Additive Random Noise Removal*

Practical digital images are often degraded in some manner to some extent that requires algorithmic steps to reduce or eliminate the degradation effect. The objective of removing this degradation effects is to determine the output image $\hat{f}(x,y)$ such that it resembles the input image $f(x,y)$ as closely as possible.[1,2] To effectively achieve this objective, image restoration algorithms must be appropriately chosen to handle the type of degradation introduced, which could be due to the input channels, transmission medium, sensor and/or digitizer.[3]

In general, image restoration algorithms are mathematical and complex in terms of realization. Under this category, there is a subset of algorithms that are simple and heuristic, and has been widely employed to perform deblurring and noise removal in both the spatial and frequency domains. Most of these algorithms are designed to filter the more popular and practical types of signal-independent noise, e.g., Gaussian white noise, impulse noise, burst channel errors and noise with a geometric structure.[4] As the noise content in a digital image can be generally considered as spatially uncorrelated, these algorithms have been widely used in many applications.

Mathematically, an image degraded by such additive random noise can be represented by the following equation:

$$g(x,y)=f(x,y)+\eta(x,y), \tag{1}$$

where $g(x,y)$ is a degraded image consisting of the summation of $f(x,y)$, the input image and an additive noise term $\eta(x,y)$. From equation (1), the process of noise removal may be interpreted as given $g(x,y)$ and *a priori* knowledge of the statistical nature of $\eta(x,y)$, an approximation, $\hat{f}(x,y)$ of the input image can be determined by the filter transformation as given by the following equation

$$\hat{f}(x,y)=T[g(x,y)]. \tag{2}$$

If $f(x,y)$ is also known, the difference between $f(x,y)$ and $\hat{f}(x,y)$ may be optimized in terms of minimum mean-square error (MSE) as given by equation (3) for an image size of $M\times M$:

$$\mathrm{MSE}=\frac{1}{M^2}\sum_{x=0}^{M-1}\sum_{y=0}^{M-1}[\hat{f}(x,y)-f(x,y)]^2. \tag{3}$$

An ideal noise removal filter would of course remove the additive noise distributions exactly, restoring the original image from the noisy image completely. In reality, no mat-

ter how well a noise removal filter is designed, the restored image always exhibits a certain degree of deviation in its pixel values from the original image, $f(x,y)$. Excessive deviation often renders the restored image useless. Therefore, the problem of practical image restoration in the case of additive random noise is reduced to minimizing a chosen error function such that the restored image resembles as closely as possible to the original image objectively and subjectively.

Over the years, many filtering algorithms have been developed in both the spatial and frequency domains mainly for removing spikelike and/or Gaussian white noise distributions. Typical examples in the spatial domain are the neighborhood averaging filter, median filter, maximum filter, minimum filter, sigma filter and box filter. The details of some of three commonly used spatial filters are presented in Section 2.

1.2 *Equality versus Discrimination*

In summary, the filter algorithms discussed in Section 2 all share one thing in common, which is every single pixel in the image is subjected to the same filtering process disregarding the nature of the pixels. This philosophy of ''processing without discrimination'' is commonly employed in spatial filtering and other enhancement operations and has been proven effective in removing additive random noise but is also capable of introducing a smoothing or blurring effect to the restored image. The reason is that these algorithms do not consider which high spatial frequency component is noise and which is not. All pixels are considered equally and treated in exactly the same way. This effect is not entirely undesirable if fine details in the image are to be removed before feature extraction and segmentation, or small gaps in lines or curves are to be filled. However, such distortion may be unacceptable as it can reduce the sharpness of lines, edges and boundaries. Furthermore, indiscriminately processing the whole image wastes a significant amount of computing resources and may become critical in real-time applications. Obviously, if the corrupted pixels can be identified and only this selected subset of pixels is processed, then there would be at least two advantages: image features will not be subjected to filtering and considerable saving in computation would be expected given the algorithmic overhead for identifying the selected subset is smaller than processing all the other uncorrupted pixels. Based on a similar argument, a generalized mean filter algorithm was developed for removing impulse noise using the concepts of thresholding and complementation.[5] This particular algorithm was shown to perform at least as satisfactorily as median filters and yielded better results in some cases. However, due to the nature of the algorithm, which interrogates every single pixel in the image, its computing overhead required for identifying pixel types was high, and its overall delay is expected to be longer than median filter.

In this paper, we present the concept and algorithmic details of a feature preserving filtering algorithm focused on how it selects a subset of pixels for filtering, which is conceptually similar to the algorithm mentioned.[5] The major difference between the two approaches is that our algorithm is based on a corrupted-pixel identification (CPI) methodology over a variable subimage size instead of every pixel. This CPI algorithm interrogates the image in variable-sized subimages to determine which are the corrupted pixels and which are not. As a result, only the corrupted pixels are being filtered, whereas the uncorrupted pixels are untouched. The principle of the CPI algorithm evolves from three observations.

- First, noise effect is best considered at a subimage level.
- Second, each subimage region should be governed by a set of criteria applicable to this subimage only for differentiating the noise corrupted pixels and uncorrupted pixels.
- Third, the subimage size should not be fixed.

After extensive performance evaluation over more than one hundred images, it was found that the CPI algorithm exhibits its four major characteristics. First, its ability in removing additive random noise is better visually (subjective) and has the smallest MSEs (objective) in all cases compared with the median, averaging and sigma filters. Second, the effect of smoothing introduced by the CPI filter is minimal. In other words, almost all edge and line sharpness is preserved. Third, the CPI algorithm is consistently faster in all cases. In theory, there is no speed advantage in the worst case comparison between the CPI filter and the median filter. In practice, the CPI algorithm is around 1.6 times faster than the median and sigma filters. Furthermore, the CPI algorithm can be applied iteratively to remove noise residuals, whereas other noise filters would end up with severe edge/line degradation as a result of iterative application.

1.3 *Organization of This Paper*

This paper is organized in the following manner: Section 2 gives details of three of the common spatial filters. Section 3 overviews the concept and philosophy of the algorithm. Section 4 depicts the assumptions, algorithmic steps and realization of the CPI algorithm. Section 5 presents the performance evaluation of the CPI algorithm in terms of removing impulse noise and Gaussian white noise from lightly to heavily degraded images. Comparison is made with the average filter, the median filter and the sigma filter. The effect of varying the two major parameters—maximum intensity spread and minimum subimage size—are investigated. The computing resource requirement of the CPI algorithm is also studied and the effect of iteratively apply the CPI algorithm is determined. Section 6 concludes this paper by commenting on the performance figures and outlining some future research directions for the CPI algorithm.

2 Some Common Spatial Filter Algorithms

2.1 *Median Filter*

Median filter has the reputation of being a successful and effective technique for removing spikelike components in a noisy image.[6–8] Apart from being able to remove impulse noise effectively, this nonlinear filter algorithm has a known advantage of preserving most of the edge. In fact, although noise suppression is mostly achieved, a degree of signal distortion is still apparent.[9] This manifests itself as a

small degree of edge blurring in the restored image. As the blurring effect is rather small, it is often tolerated.

There are many forms of median filters.[8,9] In general, a 2-D $2N+1\times 2N+1$ median filter is defined as

$$\hat{f}(x,y)=\text{median}\ \{g(x+i,y+j)|i=-N,\ldots,-1,0,1,\ldots,N,\ j=-N,\ldots,-1,0,1,\ldots,N\}, \quad (4)$$

where $\hat{f}(x,y)$ is the restored pixel at (x,y), which is defined as the median of the pixel values enclosed in a 2-D window of size $2N+1\times 2N+1$ centred at $g(x,y)$. The restored image is obtained by applying Eq. (4) to all image pixels.

2.2 *Averaging Filter*

The principle of an averaging filter is that the gray level of a pixel in the restored image is the average of the pixel values of its neighbors within a defined window size in the degraded image. This algorithm assumes that noise pixels are uncorrelated with the original image and have zero average value. Equation (5) defines the averaging filter algorithm for a neighborhood size of $2N+1\times 2N+1$

$$\hat{f}(x,y)=\frac{1}{(2N+1)^2}\sum_{j=-N}^{N}\sum_{i=-N}^{N} g(x+i,y+j), \quad (5)$$

and it follows that

$$E[\hat{f}(x,y)]=f(x,y) \quad \text{and} \quad \sigma^2_{\hat{f}(x,y)}=\frac{1}{(2N+1)^2}\,\sigma^2_{\eta(x,y)}, \quad (6)$$

where $\sigma^2_{\hat{f}(x,y)}$ is the variance of $\hat{f}(x,y)$, and $\sigma^2_{\eta(x,y)}$ is the variance of $\eta(x,y)$. As N increases, the variability of the pixel values at each location (x,y) decreases. In practice, N is taken to be 1, 2 or 3.

2.3 *Sigma Filter*

Based on an averaging concept, the principle of sigma filtering is to average pixels having values fall within the two-sigma probability, which is defined as the probability of a random variable within a range of four times the standard deviations centred at its mean.[4] This probability is 0.955 in the 1-D Gaussian distribution implying that 95.5% of random samples fall within the range. For additive Gaussian white noise removal, any pixel value that falls outside the two-sigma range can be assumed to come from a different population and not considered when calculating the average. Therefore, the pixel value at (x,y) of the restored image $\hat{f}(x,y)$ is the average value calculated from the neighborhood pixel values that are within the two-sigma range.

The characteristic of this algorithm is that most of the high spatial frequency components are considered to come from a different distribution, and therefore not included in the calculation. The major drawback of this approach is that the algorithm is unable to remove very sharp noise clusters spread over 1 or 2 pixels. This problem can be resolved by choosing the neighborhood average instead of the two-sigma average if the total number of pixels (Q) within the intensity range is less than a prespecified value K. The result is of course dependent on the selection of K. Mathematically, a 2-D $2N+1\times 2N+1$ sigma filter is defined by the following equation

$$\hat{f}(x,y)=\frac{1}{Q}\sum_{j=-N}^{N}\sum_{i=-N}^{N} g(x+i,\ y+j)w(x+i,\ y+j), \quad (7)$$

where

$$w(x+i,\ y+j)=\begin{cases}1, & \text{if } g(x,y)-2\sigma_{\eta(x,y)}\leqslant g(x+i,\ y+j)\\ & \leqslant g(x,y)+2\sigma_{\eta(x,y)}\\ 0, & \text{otherwise}\end{cases},$$

$$Q=\begin{cases}(2N+1)^2, & \text{if } \sum w(x+i,y+j)<K\\ \sum w(x+i,\ y+j), & \text{otherwise}\end{cases}.$$

In practice, if $\sigma_{\eta(x,y)}$ is not known, then it will be estimated by considering a smooth region in the image or a sub-image window of $2N+1\times 2N+1$. If this is the case, the sigma calculated will be biased as the $f(x,y)$ distribution will be included in the calculation. Because of this, the two $\sigma_{\eta(x,y)}$ condition may not be applicable, instead, a one $\sigma_{\eta(x,y)}$ or asymmetric condition may be more appropriate.

3 Overview of the Feature Preserving Filtering Algorithm[10,11]

The motivation of this research is based on the observations mentioned in the Section 1.2 and two further hypotheses. Our first hypothesis is that if we can identify the corrupted pixels, then we no longer need to process every pixel in the image. By not processing the uncorrupted pixels, useful information will be preserved. The second hypothesis is that if the corrupted pixels are in minority and the algorithmic/computing overhead needed to determine the *a priori* knowledge of the pixel nature is not more than the processing time required for filtering the uncorrupted pixels, then we are likely to have a reduction in computing delay.

Based on these two hypotheses, a number of strategies were studied. Since the noise distributions we are aiming to remove are likely to be uncorrelated with the original image, the simplest method for identifying a corrupted pixel is perhaps to threshold an image into a binary image by choosing an appropriate global fixed threshold value. Broadly, pixels that are white (black) are classified as corrupted and pixels of the other value are classified uncorrupted. Of course, this method assumes the noise distribution is mostly at one end of the gray level spectrum and the original image distribution is mostly at the other end of the spectrum, and there is a clear distinction between the two. In reality, noisy images seldom behave like this. In addition, if salt-and-pepper noise is considered instead of a one-sided spectrum, global thresholding can only remove at best half of the noise content.

After careful consideration of the preceding argument, our algorithm is formulated broadly as a two stage process involving a CPI stage followed by a filtering stage. Figure 1

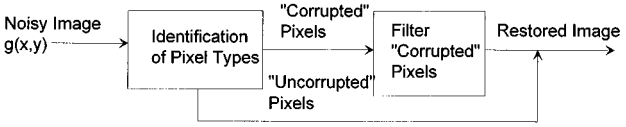


Fig. 1 Conceptual diagram of the CPI algorithm.

depicts the conceptual diagram of the CPI algorithm. At its first stage, pixels in $g(x,y)$ would be interrogated and classified into two classes: ''corrupted'' and ''uncorrupted.'' The interrogation and classification are conducted on groups of pixels instead of single pixel as it will save valuable computing resources. The second stage comprises of a filter that processes the pixels that are classified corrupted. At the conclusion of the second stage, $\hat{f}(x,y)=g(x,y)$ for uncorrupted pixels and $\hat{f}(x,y)=T[g(x,y)]$ for corrupted pixels.

Clearly, the performance of the algorithm relies on how the pixel identification is performed, and how the corrupted pixels are filtered. In the former, the identification process is independent of the filter operator used, and incurs a constant computing overhead. In the latter, the computing overhead is proportional to the number of corrupted pixels identified and the requirement of the type of filter operator used.

4 Identification of Pixel Types

Assume that the corrupted pixels are in minority, let us define the following terms:

MIS = maximum intensity spread, the maximum allowable intensity spread within a subimage region
$S_i(m,n)$ = subimage i of size $m\times n$, where both m and n are integers greater than 1
$I_i(m,n)$ = intensity spread within $S_i(m,n)$
$S(m_0,n_0)$ = subimage of minimum allowable subimage size $m_0\times n_0$ (This value should not be smaller than the mask size of the filter)
$M_i(m,n)$ = mean intensity of $S_i(m,n)$.

The two parameters MIS and $S(m_0,n_0)$ are chosen initially, and for a subimage $S_i(m,n)$ with origin at (x_i,y_i), $I_i(m,n)$ is given by

$$I_i(m,n)=\max_{\substack{x=0\\y=0}}^{\substack{x=m-1\\y=n-1}}[g(x+x_i,\,y+y_i)]-\min_{\substack{x=0\\y=0}}^{\substack{x=m-1\\y=n-1}}[g(x+x_i,y+y_i)]. \tag{8}$$

If $I_i(m,n)$ is greater than MIS and $S_i(m,n)$ is greater than $S(m_0,n_0)$ then divide $S_i(m,n)$ into two equal but smaller subimages using the following criterion: if $m\geqslant n$, then $S_{i+1}(m/2,n)$ else $S_{i+1}(m,n/2)$.

Equation (8) determines the intensity spread of $S_i(m,n)$ to consider whether a further subdivision is necessary. The argument is that if $I_i(m,n)$ is larger than a chosen MIS, such large variation in pixel values implies that the subimage consists of major features from both $\eta(x,y)$ and $f(x,y)$. As it is, it will not enable a correct decision of which are the corrupted pixels to be made. On the other hand, if $I_i(m,n)$ is smaller than or equal to MIS, then the possibility is that the subimage consists of major features of one distribution and similar features from the other. By having a dominant distribution, a decision can be made using the minority rule with regard to which pixels are corrupted and which pixels are not. The lower bound is set at $S(m_0,n_0)$ such that no further subdivision to the pixel level is allowed. The size of $S(m_0,n_0)$ corresponds to the filter size in the next processing stage simply because of the nature of the filtering operation.

From the preceding argument, if $I_i(m,n)$ is either less than MIS or $S_i(m,n)$ is equal to $S(m_0,n_0)$ then the algorithm proceeds to search for the corrupted pixels. This is achieved by determining the mean intensity $M_i(m,n)$ of $S_i(m,n)$ by equation (9), the threshold $\overline{g(}x+x_i,y+y_i)$ of $S_i(m,n)$ by equation (10) and the corrupted pixels by equation (11).

$$M_i(m,n)=\tfrac{1}{2}\Big\{\max_{\substack{x=0\\y=0}}^{\substack{x=m-1\\y=n-1}}[g(x+x_i,\,y+y_i)]+\min_{\substack{x=0\\y=0}}^{\substack{x=m-1\\y=n-1}}[g(x+x_i,\,y+y_i)]\Big\}, \tag{9}$$

$$\overline{g(}x+x_i,\,y+y_i)=\begin{cases}1 & g(x+x_i,\,y+y_i)>M_i(m,n)\\ 0 & g(x+x_i,\,y+y_i)\leqslant M_i(m,n)\end{cases}$$
$$\text{for } x=0,\ldots,m-1 \quad \text{and} \quad y=0,\ldots,n-1, \tag{10}$$

Corrupted pixels

$$=\begin{cases} g(x+x_i,\,y+y_i) & \text{where } \overline{g(}x+x_i,\,y+y_i)=0 \\ & \text{when } \displaystyle\sum_{y=0}^{n-1}\sum_{x=0}^{m-1}\overline{g(}x+x_i,\,y+y_i)\geqslant\frac{mn}{2} \\ g(x+x_i,\,y+y_i) & \text{where } \overline{g(}x+x_i,\,y+y_i)=1 \\ & \text{when } \displaystyle\sum_{y=0}^{n-1}\sum_{x=0}^{m-1}\overline{g(}x+x_i,\,y+y_i)<\frac{mn}{2}. \end{cases} \tag{11}$$

The thresholding of the subimage region by equations (9) and (10) is a standard technique[12] except that equation (9) is not the true mean of the subimage. For convenience, the maximum and minimum values that have been calculated in equation (8) are being used instead. Equation (11) aims to identify the corrupted pixels that could be black or white determined by which type of pixel is the minority in the subimage. For most common filters, this assumption applies. However, we have also evaluated the algorithm with images that are so heavily corrupted that the noise pixels are no longer in minority. The CPI algorithm performed reasonably well even under this condition, except for some subimage region where noise pixel concentration was high, unwanted ''bright'' clusters appeared to remain in the restored image. Note that under this extreme condition, all

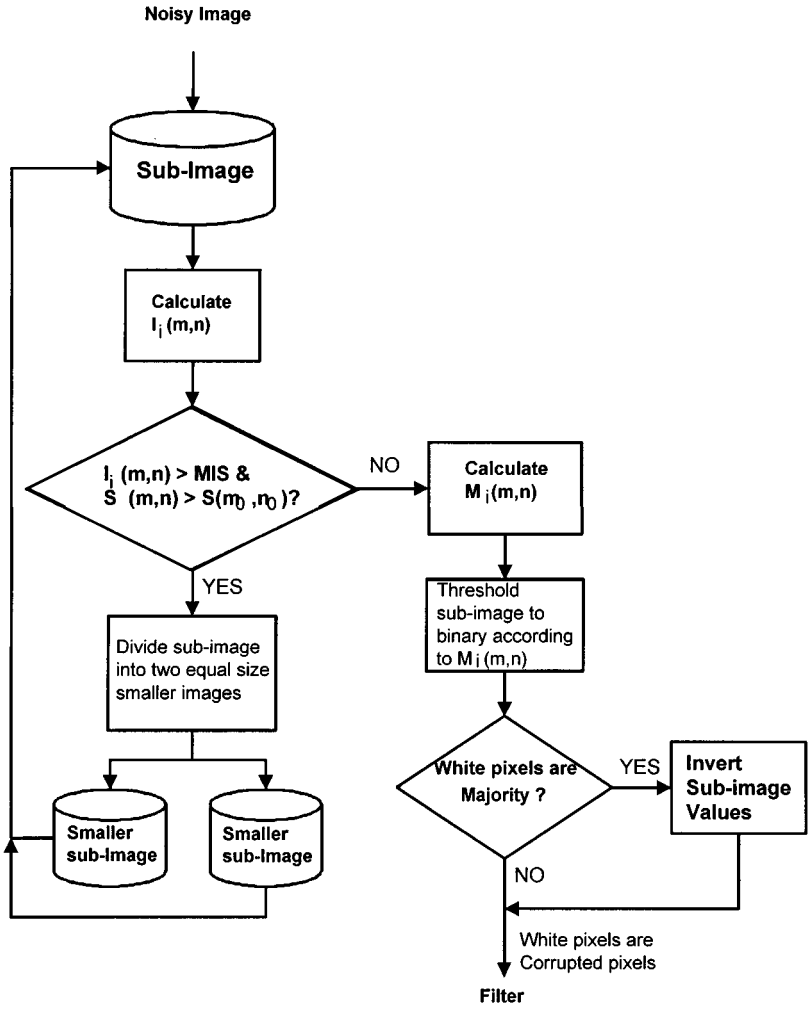


Fig. 2 Signal flow graph of the CPI algorithm.

the other filters concerned failed to remove such ''bright'' noise clusters, and the severity of these noise clusters varies from algorithm to algorithm. The signal flow of the CPI algorithm is depicted in Figure 2.

In this section, two examples are given in Figures 3 and 4 to illustrate the capability of the algorithm. Figure 3 depicts an image lightly degraded by impulse noise [Figure 3(b)] and Gaussian white noise [Figure 3(c)], and the results of filtering these noisy images by the CPI algorithm [Figures 3(d) and 3(e)]. As the degradation is not severe, the restored images resemble the original closely. Apart from the fact that most noise pixels are removed, the loss of spatial details such as blurring in the images is minimal. However, loss of contrast is evident when comparing the restored images with the original.

Figure 4 depicts an image heavily degraded by impulse noise [Figure 4(b)] and Gaussian white noise [Figure 4(c)], and the results of filtering these noisy images by the CPI algorithm. This example demonstrates that when the SNR is reduced, the CPI algorithm performs more or less as expected. The majority of the noise components are removed in the impulse case [Figure 4(d)], and the restored image closely resembles the original. In the case of Gaussian white noise [Figure 4(e)], there are noise components still remaining in the restored image especially in the heavily cluttered regions. In spite of this, the restored image is of good visual quality, clearly showing all the important features without much blurring.

5 Performance Evaluation

The evaluation of algorithmic performance discussed in this section is based on measuring the MSE between the restored image, $\hat{f}(x,y)$ and the original image, $f(x,y)$. All the images evaluated are the image of a ''Mickey mouse'' key-ring having 256 gray levels ranged from 0 (black) to 255 (white); and a spatial dimension of 205 by 441. The characteristics of this image are that the key-ring itself has sharp lines and edges, and well-defined regions against a relatively smooth background. In essence, the evaluation is focused on the aspects of

- filtering images that are degraded by impulse noise and Gaussian white noise
- smoothing effect of the four filter algorithms
- varying MIS, the maximum intensity spread and $S(m_0,n_0)$, the minimum subimage size

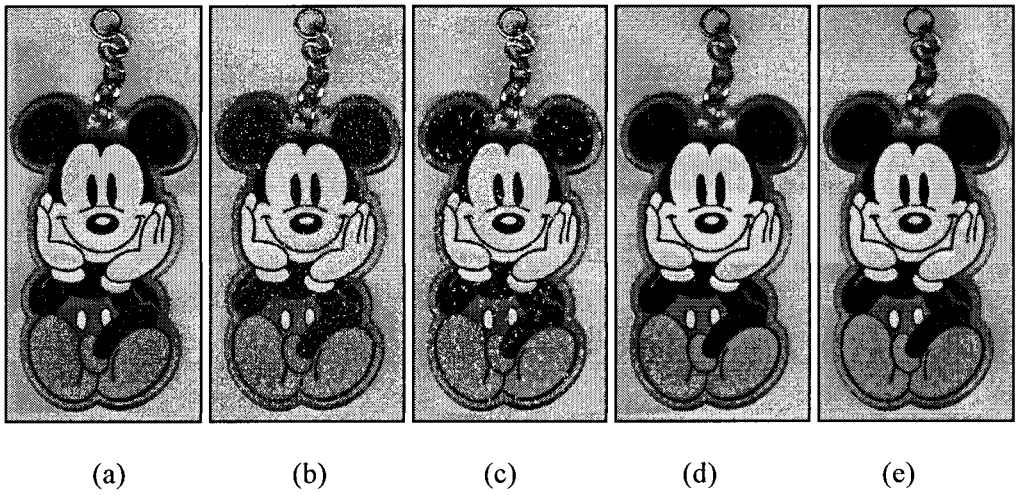


Fig. 3 Results using the CPI algorithm: (a) original image, (b) degraded by impulse noise (50 dB), (c) degraded by Gaussian white noise (50 dB), (d) image (b) restored by CPI, and (e) image (c) restored by CPI.

- theoretical and measured computational resource requirement
- and applying the CPI algorithm iteratively to a noisy image.

Comparisons are made between the CPI algorithm, median, average and sigma filters. A window size of 5×5 is used in all cases, and a 5×5 median filter is used as the filter core of the CPI algorithm. For the CPI algorithm, an MIS$=32$ is used throughout. The reason for choosing these values will be explained in Section 5.3.

5.1 *Images Degraded by Impulse Noise and Gaussian White Noise*

5.1.1 *Impulse noise*

Table 1 depicts the MSEs of the impulse noise corrupted images restored by the various filter algorithms with respect to the original image at different SNRs. These data are plotted in Figure 5 with MSE versus SNR. As can be seen, the range of SNR under investigation represents a reasonable cross-section from lightly degraded to heavily degraded cases. The corresponding MSE of $g(x,y)$ calculated for this range are also given. Broadly, all the filtering algorithms fulfill the goal of removing noise to some extent at varying degrees.

When comparing the MSE of the four filter algorithms, a number of observations can be made. First, all the error functions are monotonic increasing with decreasing SNR. Second, the averaging filter consistently scores the largest MSE throughout the SNR range. Third, the new CPI algorithm consistently scores the lowest MSE throughout the SNR range. Fourth, the median and sigma filters perform more or less the same at SNR above 10 dB, which is about twice the MSE of the CPI algorithm. Fifth, when the SNR

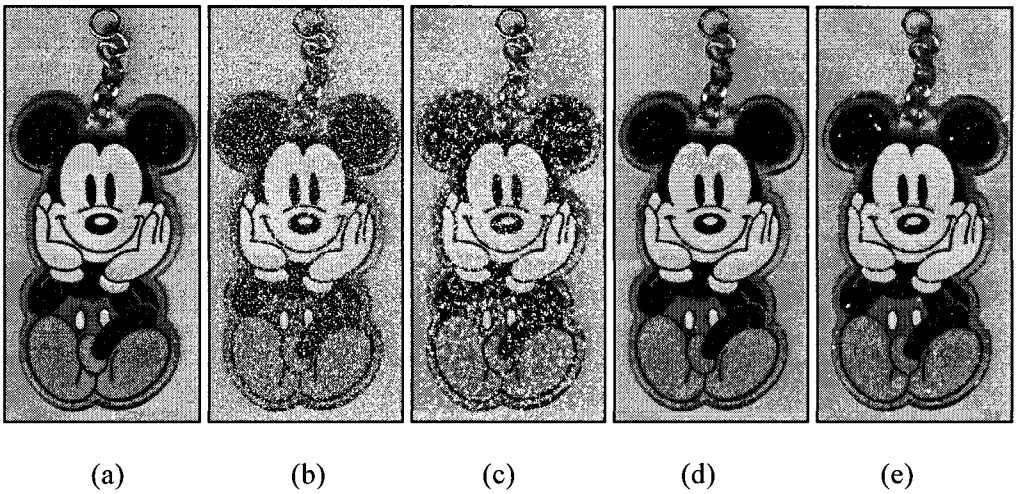


Fig. 4 Results using the CPI algorithm: (a) original image, (b) image degraded by impulse noise (0 dB), (c) image degraded by Gaussian white noise (0 dB), (d) image (b) restored by CPI, and (e) image (c) restored by CPI.

Table 1 MSE results of "Mickey" image degraded by impulse noise.

dB	$g(x,y)$	Averaging	Median	Sigma	CPI
50	791.30	457.05	302.22	305.32	138.34
40	1204.37	488.05	311.16	311.59	136.16
30	1783.16	529.86	322.29	322.86	136.41
20	2485.87	588.12	342.26	334.19	134.11
10	3329.35	673.27	365.99	351.52	144.35
0	4216.14	795.42	436.90	391.44	174.02
−10	4915.31	900.92	496.96	427.67	198.33
−20	5504.86	997.64	613.90	508.95	261.88
−30	5829.13	1053.96	722.63	582.75	334.60
−40	6119.74	1119.02	879.55	688.78	434.28
−50	6255.05	1145.52	939.31	727.98	470.65

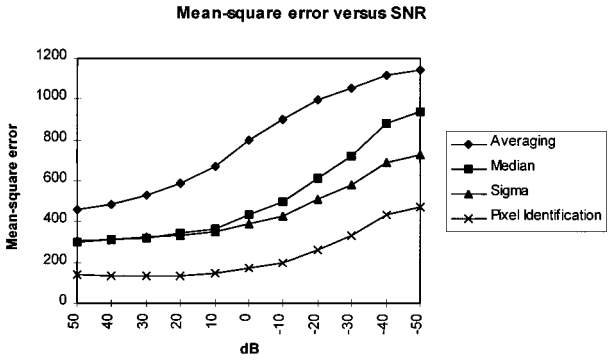


Fig. 5 MSE results of "Mickey" image degraded by impulse noise.

drops below 10 dB, the difference in MSE performance between the median and sigma algorithms become more distinct. The MSE of the median filter increases at a rate faster than the sigma filter. This result is interesting as the median filter is designed to filter impulse noise, whereas the sigma filter is designed to remove Gaussian noise, but the MSE results indicate otherwise.

Figure 6 depicts the noisy image at SNR equal to 50 dB and the restored images for visual inspection. In this case, the noisy image is corrupted by sparsely populated impulses. The fine details of the original are still visually evident. In terms of the restored images, the averaged image is slightly blurred but can still be considered as acceptable as most of the lines and edges are identifiable. The other three images [Figures 6(c) to 6(e)] are all of good visual quality, in which edge and line sharpness have been preserved and blurring is minimal with the exception of perhaps the image restored by the CPI algorithm. It is slightly sharper than the other two.

On the other hand, Figure 7 illustrates the case where the original is heavily corrupted (at SNR$=-50$ dB). The noisy image shown in Figure 7(a) is so badly corrupted that the line and edge sharpness in the original are partially lost and the shape of the key-ring for example, is barely identifiable with most of the details are no longer visually detectable. Figures 7(b) to 7(e) present the restored images by the four filter algorithms concerned.

In the case of the averaged image, the restored image is severely distorted [Figure 7(b)]. Although most of the noise components have been removed, for instance the top left-hand region is reasonably smooth after filtering, the edge and line sharpness is reduced at the same time to such an extent that edges and lines in the bottom half of the image are almost indistinguishable from its background. This result agrees with the high MSE of averaging as given in Table 1 and Figure 5.

In the case of the image restored by the median filter [Figure 7(c)], the restored image is almost acceptable except that a large number of noise clusters remains in the image. These noise clusters correspond spatially to the high concentration of noise pixels in the noisy image. Such result can be explained as when the median filter calculates the median pixel value of a 5×5 window, the resultant median of the local window is a noise pixel. This could be due to two reasons: first, unevenly high local concentration

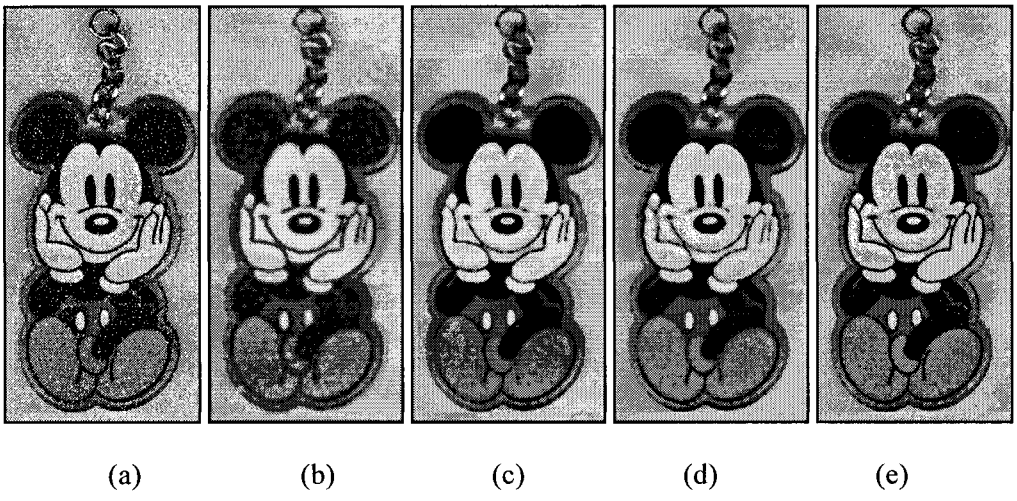


Fig. 6 Lightly degraded "Mickey" image restored by different filters: (a) image degraded by impulse noise at SNR$=$50 dB, (b) restored by the averaging filter, (c) restored by the median filter, (d) restored by the sigma filter, and (e) restored by the CPI filter.

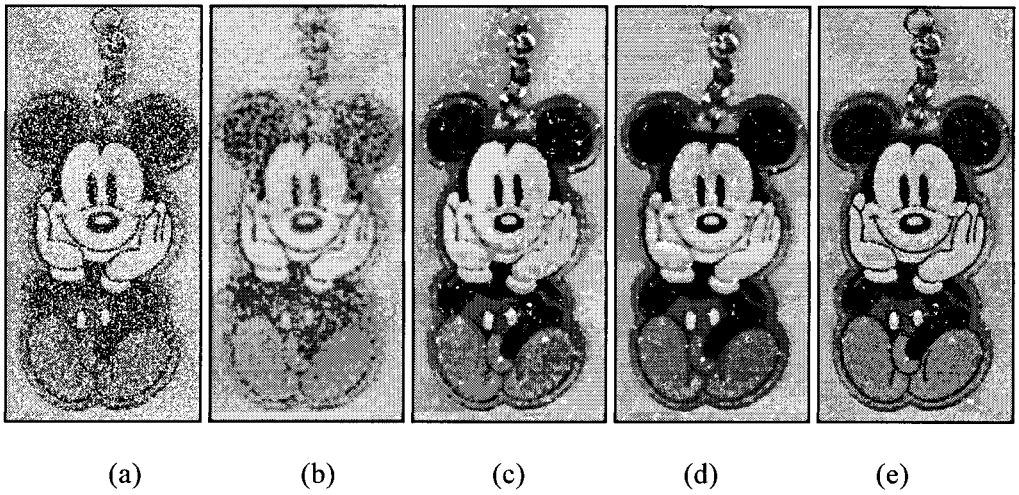


Fig. 7 Heavily degraded "Mickey" image restored by different filters: (a) image degraded by impulse noise at SNR=50 dB, (b) restored by the averaging filter, (c) restored by the median filter, (d) restored by the sigma filter, and (e) restored by the CPI filter.

of noise pixels, and second, the overall noise pixel is in majority. Although the noise pixels are normally distributed, it is more likely that the former is the case. These noise clusters are also believed to be responsible for the high MSE shown in Table 1 and Figure 5.

For the image restored by the sigma filter [Figure 7(d)], the overall visual appearance is acceptable except that a number of noise clusters again remains in the image. However, when this is compared with the median filter case in Figure 7(c), the degree is not as severe. The ability of the sigma filter to remove clustered noise components probably accounts for the reason why the sigma filter has a lower MSE than the median filter. Apart from that, both the median and sigma filters caused a small degree of edge/line distortion.

In the case of the CPI filter, an even smaller number of noise clusters remains. In addition, the edge/line distortion is the smallest, which can be detected visually on close comparison with the original in Figure 3(a). This corresponds to the fact that the CPI algorithm gives the smallest MSE.

Table 2 MSE results of "Mickey" image degraded by Gaussian noise.

dB	$g(x,y)$	Averaging	Median	Sigma	CPI
50	695.18	488.31	302.39	302.43	142.92
40	1148.57	548.25	321.92	321.87	149.04
30	1655.70	619.55	338.45	329.56	156.12
20	2374.66	713.95	366.92	344.84	163.56
10	3180.66	854.10	454.08	399.13	216.67
0	3938.94	979.59	583.48	468.95	288.86
−10	4680.28	1109.74	789.97	613.32	420.44
−20	5152.10	1211.51	1107.57	796.08	628.67
−30	5581.58	1314.06	1510.47	1003.42	923.63
−40	5852.77	1370.45	1788.80	1159.46	1119.18
−50	6062.83	1417.04	1989.91	1367.38	1237.19

5.1.2 *Gaussian white noise*

Table 2 depicts the MSEs of the images corrupted by Gaussian white noise and the images restored by the filter algorithms concerned with respect to the original image at an SNR range between 50 dB and −50 dB, and these data are plotted in Figure 8. As in Table 1, the corresponding MSE of $g(x,y)$ calculated for this range are also given. In general, this set of data shows that all four filter algorithms fulfill the goal of noise removal to some extent at varying degrees.

When making detailed comparisons, a number of observations can be made. First, all the error functions are still monotonic increasing with decreasing SNR. Second, the CPI algorithm still has the lowest MSE among the four algorithms. However, the difference in MSE between the averaging filter, sigma filter and the CPI algorithm becomes less obvious when the SNR is below −30 dB. Third, both the median and sigma filters have similar performance at SNRs above 20 dB. Their difference begins to show below this point and the gap widens as the SNR decreases. This seems to agree with the expectation that the sigma filter is more suited to remove Gaussian noise than the median filter. Fourth, the MSE of the median filter deteriorates rap-

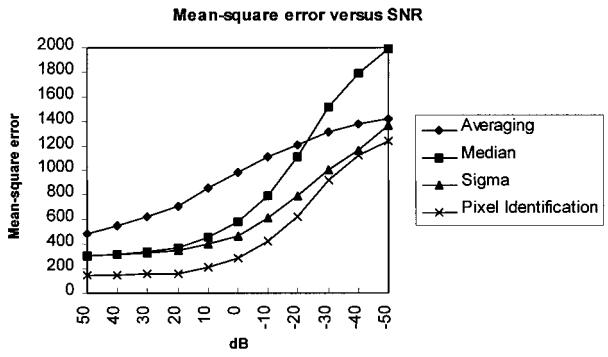


Fig. 8 MSE results of "Mickey" image degraded by Gaussian white noise.

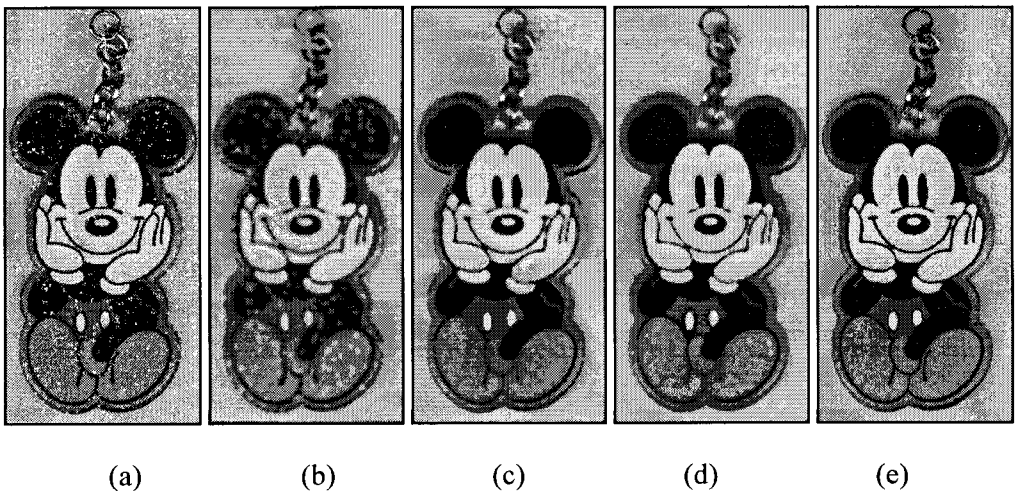


Fig. 9 Lightly degraded ''Mickey'' image restored by different filters: (a) image degraded by Gaussian white noise at SNR=50 dB, (b) restored by the averaging filter, (c) restored by the median filter, (d) restored by the sigma filter, and (e) restored by the CPI filter.

idly below −20 dB and becomes worse than the averaging filter below −30 dB. Figure 8 clearly identifies these trends.

When comparing the two graphs given in Figures 5 and 8, note that the two graphs are very similar at an SNR range between 0 and 50 dB, with the Gaussian case having larger MSE figures than the impulse case. As SNR falls below 0 dB, the MSE figures for the Gaussian case are a lot larger than the impulse case.

For the purpose of subjective measurement, Figures 9 and 10 display the images restored from the lightly (50 dB) and heavily (−50 dB) corrupted noisy images. In Figure 9, the restored images are all considered visually acceptable. On close inspection, the averaged image [Figure 9(b)] looks more grainy than the other three restored images [Figures 9(c) to 9(e)], especially over the smooth regions. Edge and line distortion in the averaged image is also evident, whereas the rest have minimal distortion and most details are preserved.

Figure 10 depicts the worst scenario among all our tests. The original image is severely corrupted and none of the restored images can be considered as acceptable. In the case of averaging [Figure 10(b)] the image quality is so poor that edges and lines are completely distorted and blurred, and the grainy effect is extensive. The quality of the images restored by the median filter, the sigma filter and the CPI algorithm are all rather poor as well. Of the three algorithms, the number of noise clusters that remains in the image is worst in the median case, which reflects the high MSE given in Table 2 and Figure 8. Both the sigma and CPI algorithms have a similar number of noise clusters left in the restored images. A small difference perhaps is

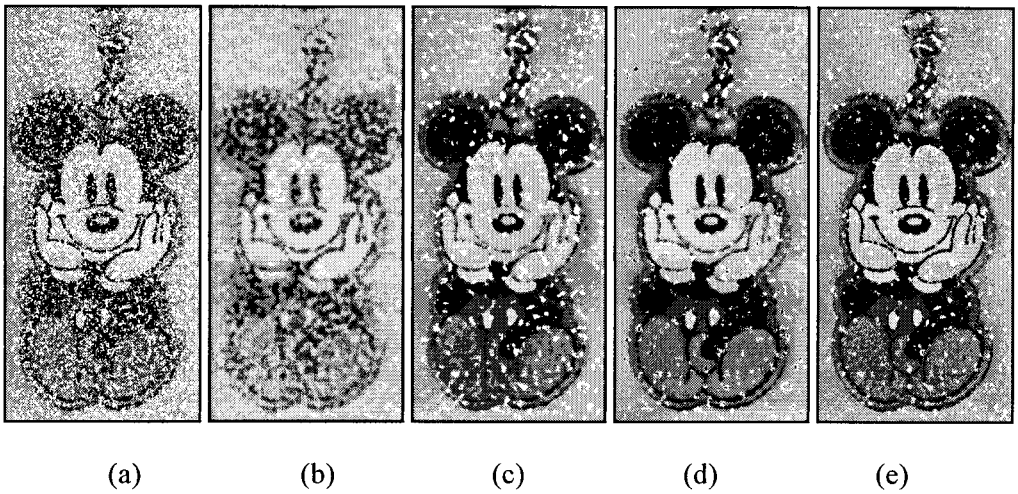


Fig. 10 Heavily degraded ''Mickey'' image restored by different filters: (a) image degraded by Gaussian white noise at SNR=50 dB, (b) restored by the averaging filter, (c) restored by the median filter, (d) restored by the sigma filter, and (e) restored by the CPI filter.

Table 3 Mean-square errors of filtering the original ''Mickey'' image by different filters.

Averaging	Median	Sigma	CPI
406.62	291.45	293.49	144.03
1	0.71	0.72	0.35

that the sigma filter introduces larger distortion on the edges and lines than the CPI algorithm. The small difference in the MSEs seems to agree with this reasonably.

5.2 *Smoothing Effect*

This evaluation aims to identify how much distortion or smoothing a filter algorithm will introduce when undertaking the noise filtering process. The image used for this evaluation purpose is the original ''Mickey mouse'' key-ring image without noise degradation. The absolute and relative MSEs for the filter algorithms used are given in Table 3 and the resultant images are depicted in Figure 11.

From Table 3, we can observe that all the filter algorithms evaluated introduce a certain degree of smoothing to the original image. When the smoothing effect of each filter algorithm is evaluated objectively, the new filter scores the lowest MSE. This indicates the smoothing effect of the CPI algorithm is the least. The small difference between the median and sigma filters ranks them equal in this case. Their actual MSEs are twice that of the CPI algorithm as given by the relative figures, whereas the MSE of the averaging filter is almost three times higher than the CPI algorithm.

Subjective evaluation of the smoothing effect seems to agree with the preceding results. From Figure 11, the smoothing effect is rather obvious in the averaging case. The edge and line sharpness is mostly degraded, but other areas appear to be smoother than before. However, trying to differentiate the smoothing effect caused by the other three algorithms is a slightly more difficult task. The smoothing effect of the median and sigma filters is minor but detectable. Both filtered images show a small degree of smoothing, but it is not severe. In the case of the CPI algorithm, the visual difference between the original image and the filtered image is minor. The difference in edge sharpness can only be detected on close inspection. This is apparent in features such as Mickey's face and hands. Such a result can be easily explained by the inherent nature of the algorithm of which only corrupted pixels are processed, and as such, a majority of the pixels are not filtered and therefore major features preserved.

5.3 *Varying $S(m_0,n_0)$ and MIS*

This evaluation aims to determine the effect of varying the minimum subimage size $S(m_0,n_0)$ and the MIS on the restored image using the CPI algorithm. The image used is the ''Mickey mouse'' key-ring image degraded by Gaussian white noise at 0 dB SNR. The MSEs are tabulated in Table 4 at various $S(m_0,n_0)$ and MIS.

From Table 4, a number of points can be observed. First, MSE in general increases with increasing minimum subimage size. For example, at $S(m_0,n_0)=3$, most MSEs are around 300, whereas at $S(m_0,n_0)=50$, all MSEs are close to 500. Second, the variation of MIS indicates that it can be optimized. Taking the case of $S(m_0,n_0)=3$, the minimum is at MIS=192. The same applies in the other cases of $S(m_0,n_0)$. Third, when combining the two parameters, it is noted that there are minima at $S(m_0,n_0)=3$ and MIS=192 and $S(m_0,n_0)=5$ and MIS=4, 8, 16, 32, 64, 128 and 160. As the difference between these minima is small, it can be deduced that if these values of $S(m_0,n_0)$ and MIS are chosen, the CPI algorithm would be optimum or near optimum in the MSE sense. In our previous evaluation, we chose $S(m_0,n_0)=5$ and MIS=32, which correspond to a near optimum MSE.

5.4 *Computing Resource Evaluation*

For a conventional spatial filter, the total number of computations is proportional to the size of the image disregarding the nature and degree of noise degradation. For an $M\times M$ image, this can be written as

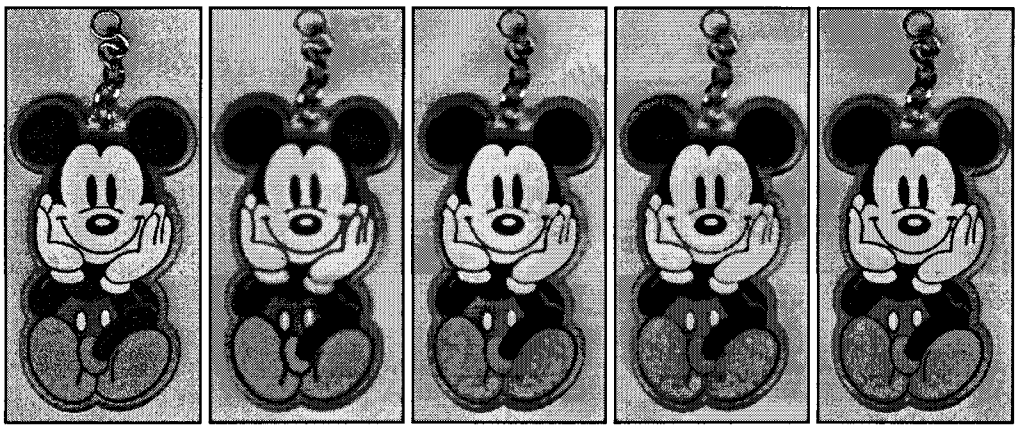


Fig. 11 Smoothing effect of different filters: (a) original image, (b) processed by the averaging filter, (c) processed by the median filter, (d) processed by the sigma filter, and (e) processed by the CPI filter.

Table 4 Effect of varying $S(m_0,n_0)$ and MIS.

$S(m_0,n_0)$	MIS	MSE	$S(m_0,n_0)$	MIS	MSE
3	32	314.70	10	2	399.09
3	64	312.87	10	16	399.09
3	128	299.15	10	32	399.09
3	192	287.38	10	64	399.09
3	254	481.01	10	128	399.11
5	2	306.86	10	192	399.70
5	4	288.85	10	254	485.51
5	8	288.85	20	32	463.12
5	16	288.85	20	64	463.12
5	32	288.86	20	128	463.12
5	64	288.87	20	192	463.12
5	128	288.45	20	254	487.81
5	160	291.72	50	32	485.57
5	192	301.62	50	64	485.57
5	224	325.63	50	128	485.57
5	254	482.16	50	192	485.57
			50	254	493.78

$$C_{\text{filter}}(M,M)=M^2\{C_{\text{filter}}[(2N+1),(2N+1)]\}, \tag{12}$$

where $C_{\text{filter}}(M,M)$ is the number of computations required to completely process the $M\times M$ image and $C_{\text{filter}}[(2N+1),(2N+1)]$ is the number of computations required to filter a single pixel, assuming the window size is $2N+1\times 2N+1$. For the median filter, $C_{\text{filter}}[(2N+1),(2N+1)]$ is the average number of comparisons required using quick sort and is equal to $(2N+1)\ln[(2N+1)]$. Therefore, equation (12) could be written for the median case as

$$C_{\text{median}}(M,M)=M^2[(2N+1)\ln(2N+1)]. \tag{13}$$

From Eqs. (12) and (13), it is obvious that the computing time required by these filters is determined by M^2 and a function of $(2N+1)$, depending on which filtering algorithm is selected.

In the case of the CPI algorithm, the number of computations is determined by the SNR and the overall size of the image. A lightly degraded image would have less noise pixels overall, and hence less number of corrupted pixels identified and processed. A heavily degraded image, on the other hand, would have a larger number of corrupted pixels for processing. In addition, it has been assumed that the noise pixels are in minority. In the worst case, this number can only be as large as $1/2M^2$.

However, the CPI algorithm has a computing overhead in the identification of pixel types. If this computing time is less than $1/2M^2\{C_{\text{filter}}[(2N+1),(2N+1)]\}$, the algorithm will have a computing advantage over the other common filter algorithms, otherwise, the performance gain will be offset by the slower identification time.

In considering the CPI algorithm, there are at least two methods for calculating $I_i(m,n)$, the intensity spread for the i'th subimage. The first is to calculate the corresponding $I_i(m,n)$ each time when a subimage is divided. If it is smaller than MIS, no further subdivision is needed. However, calculating $I_i(m,n)$ at this level involves the sorting of mn pixels at an average of $2m\ln(m)+2$ comparisons if $n>m$. This could be as large as $M\ln(M/2)+2$ if the first subdivision satisfies the preceding condition. The second method is to divide the subimage until $S_i(m,n)$ equals $S(m_0,n_0)$. This involves sorting $(2N+1)^2$ pixels at an average of $(2N+1)\ln(2N+1)$ comparisons per subimage, and 2^k subimages at the k'th level of subdivision. Maximum and minimum pixel values of larger subimages are determined by comparing the pixel maxima and minima of two identical smaller subimages. These maxima and minima can also be used for calculating $M_i(m,n)$ at a later stage.

As the second method covers all the possibilities in the manipulation of subimages, the following discussion is based on this particular method. From the preceding argument, the number of iterations (k) is determined by the inequality given in equation (14). The number of comparisons (C_k) required to determine that is given by Eq. (15).

$$\frac{M^2}{2^k}\leqslant(2N+1)^2, \tag{14}$$

$$C_k=2\sum_{i=1}^{k}2^i=2(2^{k+1}-2). \tag{15}$$

Using the boundary condition of equation (14), the number of comparisons (C_{2N+1}) required to determine $I_i(m,n)$ for all the 2^k regions is

$$C_{2N+1}=2^k[(2N+1)\ln(2N+1)]. \tag{16}$$

To compute Eq. (11) we need mn additions for each subimage $S_i(m,n)$ and a total of M^2 additions for the whole image. Since comparisons are usually slower than additions, let us assume these M^2 additions can be completed within the time taken to perform M^2 comparisons. Therefore the total number of comparisons (C_t) required to calculate $I_i(m,n), M_i(m,n)$, and identify the corrupted pixels in the worst case is

$$\begin{aligned}C_t&=C_k+C_{2N+1}+M^2\\&=2(2^{k+1}-2)+2^k[(2N+1)\ln(2N+1)]+M^2.\end{aligned} \tag{17}$$

For the CPI algorithm to perform faster than the median filter, we must have $C_t\leqslant 1/2M^2[(2N+1)\ln(2N+1)]$, or

$$\begin{aligned}&2(2^{k+1}-2)+2^k[(2N+1)\ln(2N+1)]+M^2\\&\quad\leqslant\tfrac{1}{2}M^2[(2N+1)\ln(2N+1)].\end{aligned} \tag{18}$$

Equation (18) can be expressed as

$$\begin{aligned}f(2N+1)=M^2\{&\tfrac{1}{2}[(2N+1)\ln(2N+1)]-1\}\\&-2^{k+1}\{\tfrac{1}{2}[(2N+1)\ln(2N+1)]+2\}+4\geqslant 0.\end{aligned} \tag{19}$$

By differentiating $f(2N+1)$ with respect to $(2N+1)$, we have

Table 5 Computing speed of the four filter algorithms on Impulse noise corrupted images.

dB	Averaging	Median	Sigma	CPI
50	6	27	27	17
40	7	27	27	15
30	6	27	28	15
20	6	28	27	15
10	6	28	27	14
0	6	28	28	15
−10	7	28	27	15
−20	7	28	27	15
−30	7	27	27	16
−40	7	27	27	17
−50	6	28	27	17
Average	6.45	27.55	27.18	15.55
Relative	1	4.27	4.21	2.41

Table 6 Computing speed of the four filter algorithms on Gaussian white noise corrupted images.

dB	Averaging	Median	Sigma	CPI
50	6	27	28	17
40	6	26	28	17
30	7	27	27	17
20	6	28	27	16
10	6	27	27	16
0	7	28	27	16
−10	7	27	27	17
−20	7	28	27	17
−30	6	28	27	18
−40	6	28	27	17
−50	7	28	27	17
Average	6.45	27.45	27.18	16.82

$$\frac{\mathrm{d}[f(2N+1)]}{\mathrm{d}(2N+1)}=\frac{M^2}{2}[1+\ln(2N+1)]-\frac{1-\ln(2N+1)}{(2N+1)^2}, \tag{20}$$

which is always positive, implying $f(2N+1)$ is monotonic increasing for all $2N+1\geqslant 1$. Considering equation (19) and using the boundary condition of $M^2=2^k(2N+1)^2$, we can rewrite the equation as

$$\begin{aligned} f(2N+1)&=(M^2-2^{k+1})\{\tfrac{1}{2}[(2N+1)\ln(2N+1)]-1\}\\ &\quad -3.2^{k+1}+4\\ &=2^k(\{\tfrac{1}{2}[(2N+1)\ln(2N+1)]-1\}[(2N+1)^2\\ &\quad -2]-6)+4. \end{aligned} \tag{21}$$

Note that Eq. (21) is always positive, as long as $(2N+1)$ is larger than 3, implying the worst case computing requirement of the CPI algorithm in theory is always better than the median filter for $(2N+1)>3$.

In our performance evaluation, the computing time for each SNR case and for each filtering algorithm is recorded and an average is taken for each algorithm for comparison. These data are tabulated in Tables 5 and 6. The hardware platform for this test is a 486 DX2/66 and the measurement is in seconds. From Table 5, we can observe that, first, the computing requirement for the averaging, median and sigma filters is independent of the SNR of the image. Second, median and sigma filters have very similar performance. Third, the computing requirement for the CPI algorithm is not monotonic. There is a minimum at around 10 dB, at which the computing time is shortest. For the case of smaller SNR, this can be explained as the number of corrupted pixels identified is more, therefore the time required to identify and process them is longer. On the other hand, it is also found that the number of corrupted pixels identified by the algorithm for the larger SNR cases increases with increasing SNR beyond 10 dB. According to our measurement, there is a difference of around 6000 pixels between 50 dB (29,705) and 10 dB (24,090). This can be explained as some of the uncorrupted pixels are being identified as corrupted because of the condition for the decision is based on smaller than or equal to the MIS value, as explained in Section 4. This is undesirable in terms of noise identification, but the results do not significantly vary the MSE or the appearance of the image. Fourth, averaging is the fastest among the four algorithms, with the CPI algorithm 2.41 times slower, and the median and sigma filters 4.27 and 4.21 times slower, respectively, in the case of impulse noise removal. The results are almost identical for the Gaussian white case. From these measured data, it can be deduced that equation (21) is a reasonable representation of the actual computing requirement.

5.5 *Iterative Application of the CPI Algorithm*

In practice, conventional filter algorithms are seldom used iteratively in processing a noisy image even when the result of the restored image is far from satisfactory. This is not unexpected as these filters remove noisy components as well as distort the line and edge information contained in the image. If these filter algorithms are applied to a noisy image iteratively, the resultant distortion may become more intolerable.

Because the CPI algorithm exhibits the desirable property of feature preservation, it is likely that even if the algorithm is applied iteratively to a noisy image, the resultant line and edge distortion will be kept to a minimum. Therefore, it is the purpose of this section to study the effect of applying the CPI and other filtering algorithms iteratively and to identify the optimal number of iterations, if one exists, with respect to noise removal and feature preservation.

Table 7 depicts the MSE of iteratively processing the ''Mickey'' image degraded by Gaussian noise at 0 and −50 dB using the four algorithms, and these data are plotted in Figures 12(a) and 12(b). From Table 7 and Figure 12, a number of points can be observed. First, at both moderate (0 dB) and severe (−50 dB) degradation, the application of averaging filter iteratively to the image resulted in an increase in MSE. This result shows that the averaging filter is unable to remove noise components effectively, even if it is

Table 7 MSE errors of iteratively processing the "Mickey" image degraded by Gaussian noise at 0 dB and −50 dB using the three filter algorithms and the CPI algorithm.

Iteration	Averaging	Median	Sigma	CPI
		SNR=0 dB		
1	979.59	583.48	468.95	288.86
2	1021.11	513.74	448.61	236.76
3	1135.51	549.92	497.83	257.15
4	1233.61	585.21	539.14	279.56
5	1328.45	613.39	575.41	294.95
		SNR=−50 dB		
1	1417.04	1989.91	1367.38	1237.19
2	1420.34	1154.84	707.07	405.06
3	1526.24	925.46	668.65	327.22
4	1631.73	848.39	677.76	331.10
5	1732.16	819.95	696.07	343.65

applied again, and will introduce more distortion to the image after each iteration.

Second, for the rest of the algorithms including the CPI, the MSEs decrease initially and increase after a number of iterations, in the case of moderate degradation. This minimum is at the second iteration for all three algorithms, with the CPI algorithm having the smallest MSE, followed by the sigma filter and then the median filter. The MSE of the sigma and median algorithms are very similar while the MSE of the CPI algorithm is consistently substantially smaller. This can be explained as when the filter algorithms are only applied once, the effect of removing the noise components dominates the effect of image distortion. On the other hand, after the second iteration, the reverse is true. The optimum is to apply the algorithms twice in this circumstance.

Third, in the case of severe degradation, the effect of iteratively applying the three filter algorithms is more noticeable as the MSEs are reduced almost by half in the cases of the median and sigma algorithms, and three times in the case of the CPI algorithm after the second iteration. Further reduction in MSE can also be seen with median filtering, whereas both the sigma and CPI algorithms exhibit a minimum at the third iteration. This can be explained as for the median filter, the effect of removing the noise components and residuals dominates the effect of image distortion up to five iterations. It would not be unreasonable to expect a minimum MSE for the median filter at a higher number of iterations. For the sigma and CPI algorithms, the reduction in MSE after the first iteration is more substantial, and as a result, their respective minimums correspond to the third iteration.

Figure 13 depicts the filtered images of an original image at SNR=−50 dB for subjective evaluation. It is clear that the averaged images are not acceptable visually, whereas in both the median and sigma cases, more noise residuals are removed as a result of iterative filtering. Unfortunately, the line and edge distortion has also worsened visually. For the CPI algorithm, the visual appearance of the processed images is most acceptable with the least amount of noise residuals and distortion.

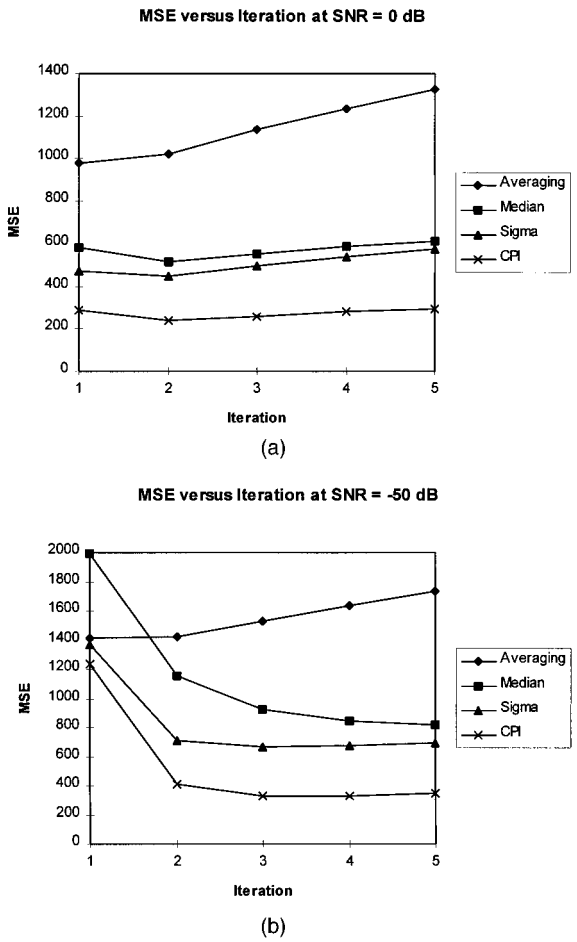
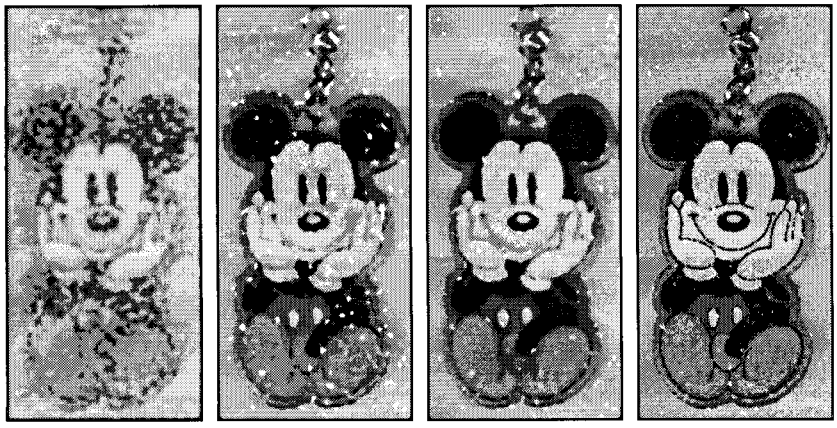


Fig. 12 MSE results of "Mickey" image filtered iteratively at SNRs of (a) 0 dB and (b) −50 dB.

In terms of computational requirement, the average, median and sigma filters require the same amount of computation in each iteration, therefore if L is the number of iterations performed, the total number of computations required is $LC_{\text{filter}}(M,M)$, where $C_{\text{filter}}(M,M)$ is given by Eq. (12). This is also true for the CPI algorithm, as explained in Section 5.4.

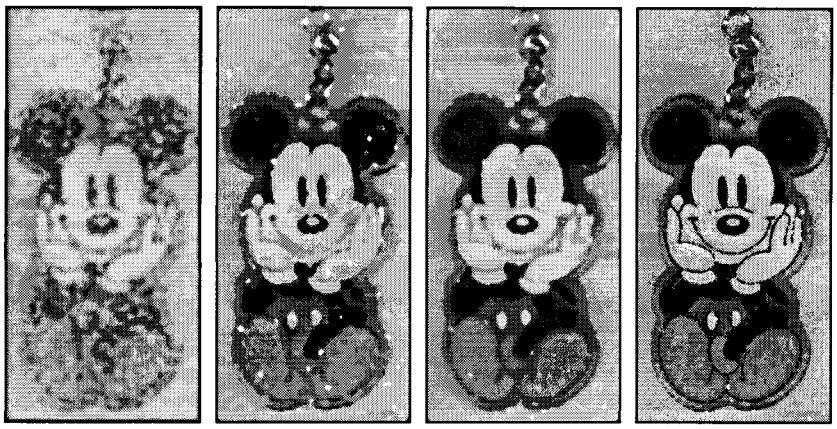
6 Conclusion

From the evaluation given in Section 5, the CPI algorithm is the best performing filter algorithm among all those considered in this paper. In general, we can conclude that the CPI algorithm is more capable of removing impulse noise and Gaussian white noise than the averaging, median and sigma algorithms. The CPI algorithm consistently scores the lowest MSE among the group of algorithms concerned. Subjective evaluation also shows that the CPI algorithm has a better visual appearance than the others due to the fact that noise pixels are more effectively removed, and most image features are preserved. In terms of computing resource requirement, the computing speed of the CPI algorithm is determined by the SNR rather than just the overall image size. In theory, the total computing resource required for identifying pixel types and filtering the corrupted pixels is always less than that of median filter for $(2N+1)>3$.



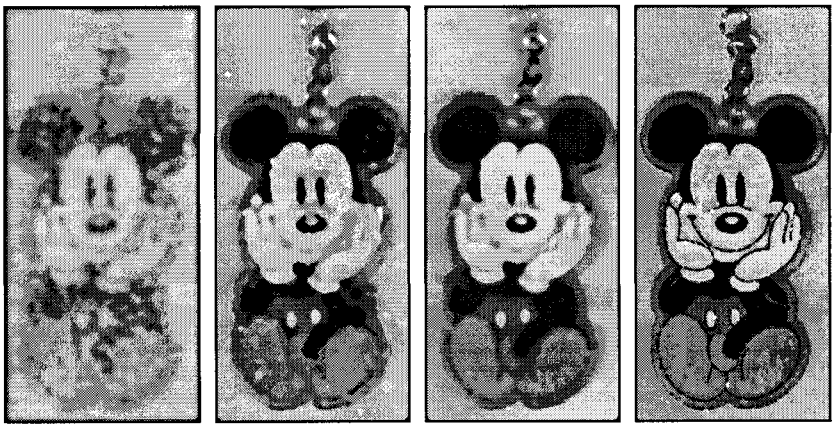
averaging median sigma CPI

(a)



averaging median sigma CPI

(b)



averaging median sigma CPI

(c)

Fig. 13 Iteratively processed heavily degraded image at SNR = −50 dB (a) twice, (b) three times, and (c) five times.

This is under the condition that the filtering core in the CPI algorithm is also a median filter. Extensive evaluation of digital images at different SNR values shows that in the worst case, the CPI algorithm is about 1.6 times faster than the median filter. Although the averaging algorithm is the fastest in all cases, its poor filtering accuracy excludes it from being useful practically. Furthermore, the CPI algorithm has a unique characteristic of being able to preserve edge and line sharpness while the others are unable to do so. This is due to the inherent selective property of the CPI algorithm. As uncorrupted pixels are not processed at all, the CPI algorithm gives the least distorted restored image in all our evaluation cases. Unfortunately, when the image is heavily corrupted, or no longer in minority, the CPI algorithm fails to produce good enough results. But it must be stressed that the same applies to all the other filters evaluated here. However, if the restored image is being filtered by the same algorithm iteratively, the image restored by the CPI algorithm through a number of iterations will be visually better than the restored images produced by the other filter algorithms. This is chiefly due to the fact that the edge and line sharpness of the image is preserved by the CPI algorithm, whereas this is not so in other cases. In the moderately degraded case, the optimum number of iterations for the CPI algorithm is two, and in the heavily degraded case, the optimum number of iterations for the CPI algorithm is three. In other words, if computing resource permits, an optimum number of iterations based on the CPI algorithm can be applied to a noisy image for the best result.

Regarding future directions, research effort will be spent on investigating the effect of varying the decision process described by Eqs. (9), (10) and (11), and employing a different decision function altogether. A detailed study of the number of corrupted pixels identified and whether they correspond to actual noise pixels has been conducted. The effectiveness of the algorithm against other types of noise distributions will also be investigated. At present, the CPI algorithm together with the other filter algorithms concerned are being implemented in a multiprocessing environment, to study the ease of implementing such a selective algorithm compared with the more traditional algorithms.

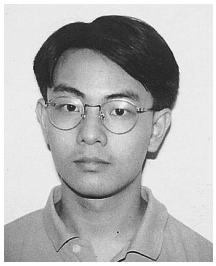
Acknowledgments

The authors gratefully acknowledge the financial support of the Committee on Research and Conference Grants of The University of Hong Kong under grant number 337/062/0016.

References

1. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, Massachusetts (1992).
2. J. S. Lim, *Two-Dimensional Signal & Image Processing*, pp. 524–588, Prentice-Hall, Englewood Cliffs, NJ (1990).
3. N. D. Sidiropoulos, J. S. Baras and C. A. Berenstein, ''Optimal filtering of digital binary images corrupted by union/intersection noise,'' *IEEE Trans. Image Process.* **3**, 382–403 (1994).
4. J. S. Lee, ''The sigma filter and its application to speckle smoothing of synthetic aperture radar image,'' in *Statistical Signal Processing*, I. W. Eward and J. G. Smith, Eds., pp. 445–459, Marcel Dekker, New York (1984).
5. A. Kundu, S. K. Mitra and P. P. Vaidyanathan, ''Application of two-dimensional generalized mean filtering for removal of impulse noises from images,'' *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-32** (3), 600–609 (1984).
6. G. R. Arce and M. P. McLoughlin, ''Theoretical analysis of the max/median filter,'' *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-35** (1), 60–69 (1987).
7. Z. Yi, ''Image analysis, modeling, enhancement, restoration, feature extraction and their application in nondestructive evaluation and radio astronomy,'' PhD Thesis, Iowa State Univ. (1987).
8. T. S. Huang, Ed., *Two Dimensional Digital Signal Processing II*, Topics in Applied Physics, vol. 43, Springer-Verlag, Berlin (1981).
9. J. E. Hall and J. D. Awtrey, ''Real-time image enhancement using 3×3 pixel neighbourhood operation functions,'' in *Selected Papers on Digital Image Processing*, vol. MS17, Mohan M. Trivedi, Ed., pp. 595–598, SPIE Optical Engineering Press, Bellingham, WA (1990).
10. H. C. Yung and H. S. Lai, ''An intelligent spatial filtering algorithm for removing additive random noise in digital images,'' Research Report EEE94001, Dept of E. & E. Eng., The University of Hong Kong (1994).
11. H. C. Yung and H. S. Lai, ''Novel filter algorithm for removing impulse noise in digital images,'' in *Visual Communications and Image Processing '95, Proc. SPIE* **2501**, 210–220 (1995).
12. C. R. Allen and N. H. C. Yung, *Vision Assistant Software—A Practical Introduction to Image Processing and Pattern Classifiers*, Chapman & Hall, London (1995).

Nelson H. C. Yung received his BSc and PhD degrees in 1982 and 1985, respectively, from the University of Newcastle Upon Tyne, England, where he was a lecturer from 1985 to 1990. During this period, he led a number of large-scale research projects on image processing. From 1990 to 1993, Dr. Yung was employed by the Defence Science and Technology Organisation, Australia. He was chiefly responsible for the theoretical research and algorithmic development of military-grade signal analysis systems. He became a lecturer with the University of Hong Kong in late 1993. Dr. Yung has coauthored a computer vision book and a number of book chapters and has published over 50 journal and conference papers in the areas of system methodology and image processing. He acted as referee for the IEE proceedings, HKIE transactions and a number of international conferences. He is a chartered electrical engineer and a member of IEE and IEEE.



Andrew H. S. Lai received a BEng degree in computer engineering from the University of Hong Kong in 1994, and MSc degree from the University of Surrey, UK, and was awarded the Cable & Wireless Prize for being the best overall student on the course in 1995. He is currently a PhD candidate in the Department of Electrical and Electronic Engineering, the University of Hong Kong. His research interests are digital image processing and machine navigation.