A mixed noise image filtering method using weighted-linking PCNNs

Luping Ji*, Zhang Yi

Computational Intelligence Laboratory, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610054, PR China

Received 25 January 2007; received in revised form 21 March 2007; accepted 24 April 2007

Communicated by R.W. Newcomb

Abstract

Image is often degraded by more than one type of noise. In order to design an efficient filter to remove mixed noise from image, this paper proposes a weighted-linking pulse coupled neural network (PCNN) model so as to construct a two-channel parallel noise filter using four PCNNs of this model. This filter detects noise using the pulses generated by neurons, and iteratively removes noise by the pixel signal variation of pulse neurons. The filtering parameters and the iteration stopping conditions are discussed. Experiments show that the proposed PCNN-based filtering method is fast and effective for removing single impulse noise, additional Gaussian noise, as well as the mixed noise of them.

Keywords: Noise image filtering; Weighted-linking PCNN; NMSE; PSNR

1. Introduction

Causing noise is often inevitable at image acquisition, transmission and processing stages. Little noise may often severely damage image quality. Usually, noise may make image understanding and recognizing more difficult. As a result, some serious error may be caused, even a spurious conclusion may be drawn. Thus, noise should be filtered as completely as possible from image. Generally, noise filtering technology consists of two main steps: noise detection and noise removal. Furthermore, during noise filtering, these original image features, such as edges, size and shape, should be kept unchanged.

Generally, two main models are used for characterizing most noise in digital images. They are additive Gaussian noise model and impulse noise model [6]. From the viewpoint of probability, the distribution function of Gaussian noise conforms to

\[ g(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}, \]

where \(-\infty \leq x \leq -\infty\), \(\sigma\) is the standard deviation, and \(\mu\) is the mean value of noise signal. If the mean value \(\mu = 0\), noise signal follows exactly the additive Gaussian noise model [6], which can be formulated as

\[ I_{\text{ni}}^{i,j} = I_{\text{o}}^{i,j} + n_{ij}, \]

where \(I_{\text{ni}}^{i,j}\) indicates the intensity of noise pixel \((i,j)\), \(I_{\text{o}}^{i,j}\) indicates the original pixel intensity and \(n_{ij}\) is the added noise intensity. Such a type of noise is usually generated during image acquisition. Some classical filters, such as Gaussian filter, recursive two-dimensional (2D) filter [29], adaptive Wiener filter [33], neighbors mean filter, can efficiently remove it, except that they may degrade the sharpness of object edges. To overcome this weakness, some nonlinear methods are developed [31,35,21,15,32,17]. These methods can effectively remove Gaussian noise, however they may bring a new weakness, namely some thin lines and small objects of original image cannot be well preserved.

The other one is impulse noise model. This type of noise may be characterized by replacing the portion of pixel intensity with a random value, while keeping the remainder unchanged [6]. Generally, this impulse noise model can be
formulated as

\[
P_{ij}^n = \begin{cases} 
  n_{ij} & \text{with probability } p, \\
  I_{ij} & \text{with probability } (1 - p).
\end{cases}
\]  

(3)

where \(P_{ij}^n\) indicates the total intensity of pixel \((i,j)\), \(n_{ij}\) indicates the original pixel intensity and \(I_{ij}\) indicates the added noise intensity. Such a noise is usually caused by transmission error, and it may make pixel intensity abnormally fluctuate. Such a fluctuation is often utilized by many filtering methods.

As we know, linear filtering cannot effectively remove impulse noise because it may mistakenly detect object edge as noise region. Just for overcoming this weakness, some nonlinear technologies [15,2,24,10,3,16] and neural filtering method [37] are specially developed. Like most methods, these also consist of two basic steps, namely noise detection and pixel replacement with an estimated value. However, these methods are only effective for removing impulse noise. For the removal of additional Gaussian noise, they seldom work well.

In addition to these above, wavelet transform (WT) [18] and partial differential equations (PDEs) with partial derivative estimation are also powerful tools for removing noise. The WT-based methods are often categorized into two classes, namely attenuation and selection. Many WT-based ones are developed for removing white Gaussian noise [19,25,26]. Moreover, for PDEs, the classical heat equation [30] and some nonlinear diffusion algorithms [23,36,12,7] are also frequently used for the removal of additional Gaussian noise. However, the two tools have a same application limitation, namely like linear technologies, they are also inefficient for the removal of impulse noise.

Seldom, an image only contains single noise (Gaussian or impulse noise), and it is often contaminated by more than one type of noise, such as mixed noise. The mixed noise is often caused in some complex environment, for example, an image with Gaussian noise is transmitted through a disturbed communication channel. For the existing noise image filtering technologies, most of them are specially proposed for removing single noise [29,31,2,10,3,16,37,17], and some others are presented for the mixed noise removal, such as the local image statistic filtering approach [6], the median-based SD-ROM filter [1], and the fuzzy noise filter [22]. These technologies mentioned above can only remove low-intensity mixed noise, and when they are applied to high-intensity noise, the filtering performances often heavily decline.

This paper gives a study on the removal of mixed noise using pulse coupled neural networks (PCNNs). The initial PCNN model proposed by Echorn originates from the observation of synchronous pulses in cat visual cortex [4]. Lindblad discusses image processing technologies of PCNN [13], and Johnson systematically describes some classical PCNN models and their typical applications in [9]. Moreover, Ranganath and Kuntimad develop several image processing methods [28] using modified PCNNs, such as object detection [27] and image segmentation [11]. In addition, Lindblad also discusses the pinpoint differences as well as similarities between PCNN and wavelet transform in 2D data processing [14]. Recently, various applications of PCNN are proposed, such as the pattern recognition method [20] by Muresan, the image shadow removal approach [8] by Gu, the noise filtering technology [17] by Ma, the image thinning algorithm [34] by Shang, and the intersecting cortical model (ICM) for special image processing [5] by Ekblada.

However, these traditional PCNNs are only useful for removing single noise [37,17]. To develop the mixed noise removal of PCNN, this paper constructs a two-channel parallel filter by proposing a weighted-linking PCNN model that originates from [9,27,20,8,34]. Like [27,17], this filter also needs to iteratively run, so we design a special stopping condition for determining when the two-channel PCNN filtering finishes. In this filter, each channel consists of two serial PCNNs, and each PCNN iteratively detects and removes noise using neuron pulses and pixel intensity variation, until the given stopping condition is satisfied. In addition to these, to obtain good filtering performance, these channel outputs are fused together using an optimization strategy.

The remainder of this paper is organized as follows. Section 2 introduces the pulse coupled neuron (PCN) model of the proposed network. Section 3 presents how to construct a two-channel filter using four PCNNs, and choose suitable parameters for them. Section 4 exhibits experimental results and comparisons. Finally, some conclusions are drawn in Section 5.

2. Weighted-linking PCNN model

PCNN, a well-known class of neural networks, has many advantages, such as fast parallel processing capability and one-to-one relationship between neurons and image pixels. It can conveniently process image by manipulating its neurons. These advantages make it quite attractive to image processing, as a result, many classical PCNN-based applications have been found. For noise removal, some initial algorithms [28,27,17] have been proposed, but limited by the adopted PCNN models, they are only efficient for removing single type of noise, such as impulse noise or Gaussian noise. How to remove mixed noise using PCNN is a challenging problem that is waiting for being settled. To improve the noise removal ability of PCNN and develop a new method for mixed noise removal, some modifications are introduced into existing PCNNs so as to propose a weighted-linking PCNN model. For explaining this model in a detailed way, the PCN structure of such a model and an eight-neuron linking pattern are illustrated in Fig. 1(a) and (b), respectively.

Before the PCN is introduced, some notations are defined as follows. \(N\) denotes a PCNN, and \(N_{m,n}\) denotes neuron \((m,n)\). \(H(N_{m,n})\) denotes the linking-neuron set of
$N_{m,n}$, which consists of $h$ linking neurons, $h = H^q(N_{m,n})$. $N_j$ is a brief expression of $N_{m,n}$, and $\phi(x_i,y_j)$, where $1 \leq i \leq h$, indicates the mixed input from the $i$th linking neuron, $x_i$ and $y_j$ are pixel and pulse signal of the $i$th linking neuron, respectively. $w_{ij}$ is linking weight between $N_j$ and its $i$th linking neuron. Coefficient $\gamma$ is an adjustment value, $\beta$ is linking strength, $u_j$ is total activity strength, $\theta_j$ is pulse threshold, $f$ is pulse generator function, $y_j$ is output signal and $d$ is the linking distance between two neurons. Based on these defined notations, the behavior of $N_j$ can be formulated as

$$
\begin{align*}
  u_j(t) &= [x_j(t)w(j,j) + \gamma] \times \left[ 1 + \beta \sum_{i=1}^{h} w(i,j)\phi(x_i(t),y_j(t)) \right], \\
  y_j(t) &= f(u_j(t) - \theta_j(t)),
\end{align*}
$$

where $t$ is iteration times, $t = 0, 1, \ldots, +\infty$, $\gamma > 0$, $\beta > 0$ and $1 \leq i \leq h$. The mixed input $\phi(x_i,y_j)$ follows $\phi(x_i,y_j) = |x_i - x_i| \cdot e^{x_i}$, where $| \cdot |$ is absolute value function. In Eq. (4), the pulse function $f(\phi)$ is defined as

$$
f(\phi) = \begin{cases} 
1 & \text{if } u \geq \theta, \\
0 & \text{if } u < \theta 
\end{cases}
$$

and linking weight $w_{ij}$ is the output of $w(i,j)$, which is calculated by

$$
w_{ij} = w(i,j) = \begin{cases} 
1 & \text{if } d_{ij} = 0, \\
e^{1-d_{ij}} & \text{if } d_{ij} > 0,
\end{cases}
$$

where $d_{ij}$ is the linking distance between $N_j$ and $i$th linking neuron, $N_i$. The output curves of function $w(\cdot)$ and $f(\cdot)$ are plotted in Fig. 2(a) and (b), respectively. Eq. (4) gives the rules of computing total activity strength and generating pulse signal. Eq. (5) means that if total activity reaches the pulse threshold, neuron will generate a pulse, otherwise it will not. Eq. (6) denotes that linking weight is related to the linking distance between two neurons, and it decreases exponentially with respect to linking distance $d_{ij}$.

Moreover, in Eq. (4), pulse threshold $\theta_j(t)$ of neuron $N_j$ (or $N_{m,n}$) is real-time computed from the linking-neuron set $H(N_{m,n})$, and it does not exponentially decrease with respect to $t$. Contrarily, in [37,9,27,20,8], the pulse threshold is initialized experientially, then it will decrease exponentially with respect to iteration times $t$. At the $t$th iteration, for the weighted-linking PCN, $N_j$, the real-time pulse threshold is computed by

$$
\theta_j(t) = \frac{\tau(t)}{h^2} \sum_{i=1}^{h} \sum_{i=t}^{h} \nabla x_i(t),
$$

where $h$ ($h = H^q(N_j)$) denotes the total number of linking neurons around central neuron $N_j$, and $\tau(t)$ is the mean pixel value of all linking neurons. $\nabla x_i(t)$ indicates the pixel signal difference between two neurons. This equation denotes that pulse threshold is related to these linking-neurons. For instance, for a $3 \times 3$ linking-neuron set, $h = 8$ and there are totally 25 neurons to be used for computing real-time pulse threshold $\theta_j(t)$ of the neuron.

During the weighted-linking PCNN running, if $u_j(t) \geq \theta_j(t)$, neuron $N_j$ will fire (namely it generates a pulse), and cause $y_j(t) = 1$. So in the next iteration (namely $T = t + 1$), neuron can update its pixel signal $x_j(t)$ using

$$
x_j(t+1) = \begin{cases} 
x_j(t) & \text{if } y_j(t) = 0, \\
\xi \tau(t) + (x_j(t) - \tau(t)) & \text{if } y_j(t) = 1,
\end{cases}
$$

where $\tau(t)$ indicates the mean pixel intensity of all $h + 1$ neurons, including $N_j$, and $\xi$ ($\xi > 0$) is an adjustable coefficient, which should be embodied in application. Furthermore, this rule implies that if neuron does not fire, $y_j(t) = 0$, so its pixel signal is kept unchanged, that is $x_j(t+1) = x_j(t)$. Contrarily, if neuron fires, $y_j(t) = 1$, then the pixel signal of $N_j$ is changed.

Compared with these conventional PCNNs [37,9,8], the proposed one shows some novelties: (i) linking input is extended to the mixture of pixel and pulse signals, (ii) weighted-linking weight is used to differentiatize the linking stimulus related to the linking distance, (iii) pulse threshold is computed real-timely, (iv) total activity strength is redefined using linking weight and (v) neuron can update pixel signal if it generates a pulse (fires).

3. Noise filtering using weighted-linking PCNNs

For image filtering, how to detect noise pixel is a crucial problem. Once it is determined, how to remove it is another problem. Since additional Gaussian noise and impulse noise conform to different distributions, we have to design...
a new filter, consisting of four weighted-linking PCNNs, to process mixed noise. Fig. 3 shows the coarse structure of such a proposed filter.

As shown in Fig. 3, this filter consists of four PCNNs, two for Gaussian noise, and the others for impulse noise. Networks \(N^2\) and \(N^3\) make up of the first filtering channel, \(F_1\), and \(N^1\) and \(N^4\) make up of the second filtering channel, \(F_2\). Parameters \(e\) and \(\eta\) are two weight coefficients for \(F_1\) and \(F_2\), respectively. Just like the conventional models [37,9,8], for any network \(N^k\) \((1 \leq k \leq 4)\), there exists strictly a one-to-one relationship between neuron and pixel, and this network runs iteratively with respect to iteration \(t\), until the stopping condition is satisfied or it is terminated compulsively.

To describe the proposed two-channel parallel filtering algorithm, we give two algorithm flowcharts, which are illustrated in Fig. 4. In the figure, (a) is the uniform working flowchart for the four PCNNs, which coarsely presents how to detect noise and when to switch to direct-pass model, and (b) is the systematical flowchart of the constructed filter, which presents when to terminate the filtering algorithm. Next, we will describe how to determine filtering parameters and how to define the stopping condition of PCNN.

Next in this paper, \(I_{m \times n}\) indicates a \(m \times n\) noise image, and \(I'_{m \times n}\) denotes the ultimate filtering output. PCNN is expressed as \(N^k_{m \times n}\), where \(k (1 \leq k \leq 4)\) is a serial number. The input and output of \(N^k_{m \times n}\) are denoted as \(I(N^k_{m \times n})\) and \(O(N^k_{m \times n})\), respectively. Some other filtering parameters are determined as follows. Firstly, the linking matrix (or filter window) is defined as a \(\lambda \times \lambda\) matrix, where \(\lambda\) is an odd integer. For example, if \(\lambda = 3\), the linking distance matrix,
$D_{3\times3}$, and the weighted-linking matrix, $W_{3\times3}$, are computed according to Eq. (6), and then the computation results can be expressed as

$$
D_{3\times3} = \begin{pmatrix}
\sqrt{2} & 1 & \sqrt{2} \\
1 & 0 & 1 \\
\sqrt{2} & 1 & \sqrt{2}
\end{pmatrix},
$$

and

$$
W_{3\times3} = \begin{pmatrix}
0.6609 & 1 & 0.6609 \\
1 & 1 & 1 \\
0.6609 & 1 & 0.6609
\end{pmatrix},
$$

if $\lambda$ is equal to other odd integer, the distance matrix, $D_{\lambda\times\lambda}$, and the weighted-linking matrix, $W_{\lambda\times\lambda}$, are still computed similarly to when $\lambda = 3$.

At any iteration times $t$, according to Eq. (4), for neural network $N_{m,n}^k$, where $k \in \{1, 2, 3, 4\}$, the total internal activity strength, $u_{m,n}^k(t)$, of neuron $N_{m,n}^k$ can be computed by

$$
u_{m,n}^k(t) = [x_i(t)w(j,j) + \gamma] 
\times \left[1 + \beta \sum_{i=1}^{h} w(i,j)\phi(x_i(t), y_i(t))\right] 
\times \left[1 + \beta \sum_{a,b \neq \lambda'} W_{a,b} |a_k^{m,n}(t) - x_{a,b}(t)| \exp(x_{a,b}(t))\right],$$

where $(m',n')$ are neuron coordinates (1 $\leq m' \leq m$ and 1 $\leq n' \leq n$), and $(\lambda', \lambda')$ are the coordinates of the median element in $W_{\lambda\times\lambda}$, $\lambda = (\lambda \mod 2) + 1$. $W_{a,b}$ indicates the element $(a,b)$ of $W_{\lambda\times\lambda}$, $x_{a,b}$ and $y_{a,b}$ (1 $\leq a,b \leq \lambda$ and $a,b \neq \lambda'$) denote the pixel signal and output signal of linking neuron $N_{d,b}$, where $d = m' + a - \lambda'$. Linking strength factor follows $\beta = 1/(\lambda^2 - 1)x_{m,n}^k(t)$, symbol $| \cdot |$ denotes an absolute value function, and coefficient $\gamma$ is determined experimentally. At $t = 0$ (prior to the first iteration of PCNN), PCNN begins to initialize itself, and then, at $t \geq 1$, it starts to iteratively remove noise until the stopping condition (to be defined) is satisfied.

Moreover, it should be noted that in a 2D PCNN, there are three possible locations for neuron: namely the common location (Fig. 5(a)), the edge location (Fig. 5(b)) and the corner location (Fig. 5(c)). In the illustrations, these dashed circle means that no neuron exists on the location. For the neuron $N_3$ in Fig. 5(a), the eight-neuron linking set $H(N_3)$ is full, however the eight-neuron linking set $H(N_3)$ is not full in either Fig. 5(b) or (c). Generally, for a given linking matrix $W_{\lambda\times\lambda}$, the proposed algorithm need not require that all the $\lambda^2 - 1$ linking neurons must exist, and it only considers those real-existing linking neurons within a given neuron linking set.

In addition to these, according to Eq. (7), at the $r$th iteration, real-time pulse threshold $\theta_{m,n}^r$ of neuron $N_{m,n}^r$ should be embodied by

$$
\theta_{m,n}^r(t) = \tau(t) \times \left[1 + \frac{1}{h^2} \sum_{i=1}^{h} \sum_{j=1}^{h} \nabla x_{p-r-f}(t)\right] 
\times \sum_{a,b \neq \lambda} W_{a,b} \left[\theta_{a,b}(t) - x_{a,b}(t)\right],
$$

where $\nabla x_{p-r-f}$ is the pixel gradient between two neurons, $x_{a,b}$ indicates pixel signal of neuron $N_{d,b}$ within neuron linking set $H(N_{d,b})$, and $N_{a,b}$ indicates the neuron $(a,b)$, which is within the linking set $H(N_{m,n}^r)$ of $N_{m,n}^r$.

As explained in Eq. (10), to calculate pulse threshold, the direct-linking neuron $N_{a,b}$ and the indirect-linking neuron $N_{d,b}$ within linking set $H(N_{a,b})$ must be referred to. To explain this calculation rule, Fig. 6 gives an example. For neuron $N_{m,n}^r$, the direct-linking set $H(N_{m,n}^r) = \{N_{a,b}, N_{d,b}\}_{m' - 1 \leq a' \leq m' + 1, n' - 1 \leq b' \leq n' + 1}$, and the indirect-linking set $H(N_{m,n}^r) = \{N_{d,b}\}_{m' - 2 \leq a' \leq m', n' \leq b' \leq n' + 2}$, noting $N_{m,n} \notin H(N_{m,n}^r)$ for any neuron. According to Eq. (10) and Fig. 6, if $\lambda = 3$, there will be 25 neurons to be involved in the pulse threshold calculation of $N_{m,n}^r$.

For each network of this filter, the weighted-linking matrix $W_{\lambda\times\lambda}$ always conforms to the same definition as Eq. (6). Threshold $\theta(t)$ and total activity $u(t)$ are calculated according to Eqs. (9) and (10), respectively. Furthermore, according to Eq. (8), for both the filtering channels $F_1$ and $F_2$.  

Fig. 5. Three possible locations of neuron $N_3$ appearing in network: (a) the common location; (b) the edge location; (c) the corner location.
L. Ji, Z. Yi / Neurocomputing 00 (2007) 000–000

Fig. 6. An example for computing the pulse threshold, \( \lambda = 3 \).

\( F_2 \), how to update the pixel signal of neuron is completely different from each other. Totally, the pixel signal of a pulse neuron should be updated by

\[
x_{m',n'}(t+1) = \begin{cases} 
  x_{m',n'}(t) & \text{if } y_{m',n'}(t) = 0, \\
  \xi(x_{m',n'}(t) - x(t)) & \text{if } y_{m',n'}(t) = 1,
\end{cases}
\]

(11)

where the adjustable coefficient \( \xi \) is determined by noise model, additional Gaussian noise or impulse noise. In PCNN \( \mathbb{N}^3_{m,n} \) (1 \( \leq k \leq 4 \)), as for different noise models, the coefficient \( \xi \) can be determined by

\[
\xi = \begin{cases} 
  \frac{\sigma_{m',n'}^2}{(\sigma_{m',n'}^2 + \sigma_{m,n}^2)} & \text{for } \mathbb{N}^3_{m,n}, \\
  \frac{1}{\bar{h} \cdot x_{m',n'}(t)} \sum |x_{m',n'}(t) - x_{a,b}(t)| & \text{for } \mathbb{N}^4_{m,n},
\end{cases}
\]

(12)

where \( \sigma_{m',n'} \) and \( \sigma_{m,n} \) denote the estimated local variance and mean variance, respectively. Herein, \( x_{m',n'} \) and \( x_{a,b} \) are the pixel intensities of \( N_{m',n'} \) and \( N_{a,b} \), respectively. Positive integers \( a, b, a', b' \) and \( \lambda \) conform to \( a' = m'+(a-\lambda), b' = m'+(b-\lambda) \) and \( 1 \leq a, b \leq \lambda \). The first option of coefficient \( \xi \) in Eq. (12) is designed for removing additional Gaussian noise, and the other is designed for removing impulse noise. In fact, such options of \( \xi \) are two modified forms of the definitions in the literature [24]. Generally, insufficient filtering or excessive filtering may result in a nonoptimal filtering output. Therefore, a certain stopping condition should be defined for our filter. Before we discuss the stopping condition, we firstly introduce normalized mean square error (NMSE), which is given as

\[
NMSE(I_{m,n}, O_{m,n}) = \frac{\sum_{m'=1}^{m} \sum_{n'=1}^{n} (I_{m',n'} - O_{m',n'})^2}{\sum_{m'=1}^{m} \sum_{n'=1}^{n} I_{m',n'}^2},
\]

(13)

where \( I_{m,n} \) and \( O_{m,n} \) denote filtering input and output, \( I_{m',n'} \) and \( O_{m',n'} \) denote the pixel sets of images \( I_{m,n} \) and \( O_{m,n} \), respectively.

Using NMSE, the stopping condition for PCNNs is determined as follows. Firstly, define \( e(t) = NMSE(I(N^k_{m,n}(t)), O(N^k_{m,n}(t))) \), where \( k \) is still the serial number, where \( 1 \leq k \leq 4 \). \( I(N^k_{m,n}(t)) \) and \( O(N^k_{m,n}(t)) \) are input and output of \( N^k_{m,n} \) at \( t = t \), respectively. Generally, larger \( e(t) \) implies stronger filtering response to current input. And then, define a threshold \( \Omega (\Omega > 0) \) for \( e(t) \), and experiments show that 0.0005 is a suitable value for \( \Omega \). Finally, the stopping condition of \( N^k_{m,n} \) (1 \( \leq k \leq 4 \)) is given as

\[
e(t) = NMSE(I(N^k_{m,n}(t)), O(N^k_{m,n}(t))) < \Omega.
\]

(14)

This condition means that during noise filtering stage, if \( e(t) \) decreases under the threshold \( \Omega \), the network immediately stops. Eq. (14) also implies when to stop filtering can be determined by both \( e(t) \) and \( \Omega \) together. If it satisfies \( e(t) < \Omega \) at \( t = t' \), PCNN will stop immediately, otherwise it continues to run until Eq. (14) is satisfied. Noting, PCNN determines when to stop iteration only by itself, in other words, such a stop is irrelevant to other PCNNs. Once a PCNN stops at \( t = t' \), it will switch to a direct-pass mode, and this mode results in \( O(N^k_{m,n}(t)) = I(N^k_{m,n}(t)) \) for any \( t > t' \).

As mentioned above, the two-channel filter consists of four PCNNs, and it will keep on running until all PCNNs have stopped. Once the filter stops, both \( F_1 \) and \( F_2 \) will output stable filtering signal. Suppose \( O(F_1) \) and \( O(F_2) \) indicate the output signals of channels \( F_1 \) and \( F_2 \), respectively. For obtaining better result, the ultimate filtering output \( I_{m,n} \) is fused by

\[
I_{m,n} = \varepsilon \cdot O(F_1) + \eta \cdot O(F_2),
\]

(15)

where the coefficients \( \varepsilon \) and \( \eta \) conform to \( \varepsilon + \eta \equiv 1 \) and \( 0 \leq \varepsilon, \eta \leq 1 \). They are determined as follows. Firstly, calculating the maximal value of NMSE for \( N^k_{m,n} \); \( e_k = \max_{0 \leq t \leq \varepsilon} e(t) \), where \( 1 \leq k \leq 4 \). Then coefficient \( \varepsilon \) is calculated by \( \varepsilon = (e_1 + e_2)/(e_1 + e_2 + e_3 + e_4) \), and then \( \eta \) is calculated by \( \eta = 1 - \varepsilon \). This fusion rule indicates that this channel coefficient is closely related to the PCNN with the maximal filtering response, and the channel with larger response accounts for the larger percent of final filtering output.

By now, the proposed noise filtering algorithm has been completely described. It absorbs some basic ideas of
Ranganath’s noise removal algorithm [28,27], such as iteration, detecting noise using pulses, removing noise by adjusting pixel intensity and termination condition. In addition to these, we adopt a different PCNN model from Ranganath’s, we detect noise by judging whether a neuron has generated pulse and we use a NMSE threshold instead of the unidirectional termination condition by Ranganath to determine when these PCNNs stop. Moreover, we use two parallel filtering channels to detect and remove noise, and a channel fusion strategy is also applied to this filter. These new modifications make our algorithm effective for mixed noise removal, and also bring a possibility to further improve filtering performance.

4. Experiments of testing and evaluation

In this section, two classical images and a PC with 800 Hz CPU are used in the experiments. These experiments mainly consist of two parts, namely testing the proposed algorithm and comparing it with some others. Throughout this section, \( p (0 \leq p \leq 1) \) denotes the intensity of impulse noise and \( \sigma (0 \leq \sigma \leq 1) \) denotes the standard variance of additional Gaussian noise.

Generally, filtering performance can be evaluated by NMSE (Eq. (13)) and peak signal-to-noise ratio (PSNR). Herein, PSNR is formulated by

\[
\begin{align*}
PSNR(I_{m \times n}, O_{m \times n}) &= 10 \log_{10} \left( \frac{255^2 \times m \times n}{\sum_{m'=1}^{m} \sum_{n'=1}^{n} (I_{m',n'} - O_{m',n'})^2} \right) \text{dB}, \tag{16}
\end{align*}
\]

where \( I_{m \times n} \) and \( O_{m \times n} \) denote an original image and a filtered image, respectively, \( m \times n \) denotes image size and \( (m',n') \) denotes pixel coordinates of input or output image. Just as discussed in [6], a better noise image filtering method can often result in the less NMSE and larger PSNR values.

4.1. Filtering performance testing

The filtering performance of the proposed method is closely related to two important parameters, namely the linking matrix \( W_{l \times l} \) (filtering window) and the iteration times \( t \). With respect to different parameters choice, our method often achieves different filtering performance. In this section, the performance is tested quantitatively, and these experiments are based on some noise images with different mixture \((p, \sigma)\), linking matrix and iteration \( t \).

To test filtering performance, firstly, the “Lina” image is added with three types of noise: impulse noise \( (p = 0.1) \), additional Gaussian noise \( (\sigma = 0.1) \) and mixed noise \( (p = 0.1 \text{ and } \sigma = 0.1) \), see these noise samples in Fig. 7. Moreover, the threshold \( \Omega \) is set to \(-1\), parameter \( \gamma \) is set to 0.0001, and \( \lambda = 3, 5 \) and 7. During the iterative filtering by the filter, according to Eqs. (13) and (16), at each iteration, NMSE and PSNR of filtering images can be calculated. Figs. 8–10 give these NMSE or PSNR curves with respect to \( t \).

As shown in Fig. 8, the proposed method can efficiently remove single impulse noise and achieve satisfying NMSE and PSNR values. However, with respect to different parameters, it often generates different results. Especially, too large \( t \) often worsens the performance. As a whole, \( t = 2 \) can make the filtering performance reach a peak. If the image is added with the Gaussian noise of \( \sigma = 0.1 \), our method obtains the best filtering result at \( \lambda = 3 \), see Fig. 9. Even if this mixed noise \( (p = 0.1 \text{ and } \sigma = 0.1) \) is added, there is a slight influence of \( \lambda \) on the filtering result. As a whole, the filtering performance can often reach a performance peak under the conditions of \( t = 2 \) and \( \lambda = 7 \).

As discussed before, our method is sensitive to two parameters: \( t \) and \( \lambda \). Table 1 gives the numeric sensitivity to these two parameters. These experiments are carried out on the image with the mixed noise \( (p = 0.1 \text{ and } \sigma = 0.1) \). Table 1 shows that NMSE decreases while PSNR gradually increases with respect to the iteration \( t \) and any \( \lambda \). This table also shows that the increase or decrease in magnitude always gets less and less with respect to \( t \). Once iteration satisfies \( t > 8 \), NMSE and PSNR are almost kept invariant.

Since these experiments above are based on threshold \( \Omega = -1 \), according to Eq. (13), \( 0 < e(t) < 1 \) holds, so \( e(t) > \Omega \) always holds, in other words, the stopping condition cannot be satisfied at any \( t \). Thus, the filter itself never stops unless it is terminated compulsively. These experiments also show that it can quickly stops under the conditions of \( \gamma = 0.0001, \lambda = 5 \) and \( \Omega = 0.0005 \). As for the samples in Fig. 7, their filtering results at iteration \( t = 2 \) are exhibited in Fig. 11(b)–(d), respectively. In terms of vision, they show that this filter can significantly improve image quality by a filtering technology.

Fig. 7. (a) Original image; (b) image with impulse noise, \( p = 0.1 \); (c) image with Gaussian noise, \( \sigma = 0.1 \); (d) image with mixed noise, \( p = 0.1 \) and \( \sigma = 0.1 \).
Since the proposed filtering method removes noise by updating noise pixel intensity, for a filtered image, the pixel intensity is inevitably changed during the filtering stage. To analyze these pixel intensity variations caused by the filter, we choose the pixels on the 128th row of image for some comparison experiments. The pixel intensity curve of the 128th row in the noise image Fig. 7(d) is exhibited in Fig. 12(a), and the similar curve of the image in Fig. 11(d) (namely the filtering image) is exhibited in Fig. 12(b).

As shown in Fig. 12, the original intensity curve has become a fine curve after noise filtering. Such a comparison shows that filtering does not change the curve shape,
contrarily, it makes the curve more smoother and harmonious than before. Not only the 128th row, but also any other rows conform to the same pixel variation trend as Fig. 12. From another aspect, such a variation trend also shows the usefulness of the proposed method to noise removal.

### 4.2. Filtering performance comparison

In this subsection, the proposed method is compared with some well-known algorithms: trilateral filter [6], switching median filter [21], vector median filtering method [15], order statistics filter [2], PCNN-based method [37] and the median-based SD-ROM [1]. In these experiments, some evaluation criterions, such as NMSE, PSNR and filtering time expense, are used to evaluate filtering performance of each algorithm with respect to noise mixture \((p, \sigma)\).

To obtain enough data for the performance comparison, we test the seven typical filtering methods using some “Lina” and “Man” images added with eight types of mixed noise. The parameter choice for the proposed PCNN-based filtering method is given in Table 2. As in Section 4.1, we

![Fig. 11. Filtering results, iteration \(t = 2\): (a) original image; (b) filtering result, \(p = 0.1\); (c) filtering result, \(\sigma = 0.1\); (d) filtering result, mixture \((p = 0.1, \sigma = 0.1)\).](image)

![Fig. 12. Intensity curves of the 128th row: (a) before filtering, (b) after filtering.](image)

<table>
<thead>
<tr>
<th>Times (t = \lambda)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\lambda = 3)</td>
<td>NMSE</td>
<td>0.0043</td>
<td>0.0039</td>
<td>0.0023</td>
<td>0.0017</td>
<td>0.0013</td>
<td>0.0011</td>
<td>0.0009</td>
<td>0.0008</td>
<td>0.0009</td>
</tr>
<tr>
<td>(\lambda = 5)</td>
<td>NMSE</td>
<td>0.0035</td>
<td>0.0033</td>
<td>0.0021</td>
<td>0.0016</td>
<td>0.0013</td>
<td>0.0010</td>
<td>0.0009</td>
<td>0.0008</td>
<td>0.0007</td>
</tr>
<tr>
<td>(\lambda = 7)</td>
<td>NMSE</td>
<td>0.0033</td>
<td>0.0033</td>
<td>0.0021</td>
<td>0.0016</td>
<td>0.0013</td>
<td>0.0011</td>
<td>0.0009</td>
<td>0.0008</td>
<td>0.0007</td>
</tr>
</tbody>
</table>

#### Table 2

Parameters choice for the proposed filtering method

<table>
<thead>
<tr>
<th>Distribution of mixed noise</th>
<th>Filtering window, (\lambda \times \gamma)</th>
<th>(\gamma)</th>
<th>(\Omega)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.0 \leq p, \sigma \leq 0.1)</td>
<td>(3 \times 3)</td>
<td>0.0001</td>
<td>0.0005</td>
</tr>
<tr>
<td>(0.1 &lt; p, \sigma \leq 0.3)</td>
<td>(5 \times 5)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(p, \sigma &gt; 0.3)</td>
<td>(7 \times 7)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please cite this article as: L. Ji, Z. Yi, A mixed noise image filtering method using weighted-linking PCNNs, Neurocomputing (2007), doi:10.1016/j.neucom.2007.04.015
still use NMSE (Eq. (13)) and PSNR (Eq. (16)) to evaluate the filtering performance of each filtering method with respect to different noise. These compared filtering methods, some added noise mixtures and the numeric experimental results (NMSE and PSNR) are exhibited in Tables 3–8.

According to these tables, as for each algorithm, the performance always falls with respect to noise intensity. Some methods are quite effective for the single Gaussian noise, but they are not so for the impulse noise. For example, for the switching median filter [21], vector median filter [15], order statistics filter [2] and the SD-ROM [1], these PSNR values in Table 3 are less than 27 dB, but they exceed 34 dB in Table 4. Similarly, they are almost under 22 dB in Table 6, but they are beyond 30 dB in Table 7. Moreover, some methods work well for removing single noise (Gaussian or impulse noise), but their performances fall severely for mixed noise removal. For example, for the method in [37], each PSNR is more than 28 dB in Tables 3 and 4; however, it falls below 16 dB in Tables 5 and 8. Each decrease rate of PSNR almost exceeds 40%. As a whole, only the trilateral filter [6] and the proposed one show...

Table 3
Filtering comparison of single Gaussian noise, $\sigma = 0.2$

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lina</td>
<td>0.00314</td>
<td>0.00784</td>
<td>0.00612</td>
<td>0.02860</td>
<td>0.00445</td>
<td>0.04735</td>
<td>0.00211</td>
</tr>
<tr>
<td>Man</td>
<td>0.00301</td>
<td>0.00712</td>
<td>0.00585</td>
<td>0.02842</td>
<td>0.00423</td>
<td>0.04712</td>
<td>0.00203</td>
</tr>
<tr>
<td>PSNR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4
Filtering comparison of single impulse noise, $p = 0.2$

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lina</td>
<td>0.00117</td>
<td>0.00103</td>
<td>0.00141</td>
<td>0.00135</td>
<td>0.00142</td>
<td>0.00184</td>
<td>0.00101</td>
</tr>
<tr>
<td>Man</td>
<td>0.00106</td>
<td>0.00092</td>
<td>0.00135</td>
<td>0.00128</td>
<td>0.00133</td>
<td>0.00172</td>
<td>0.00093</td>
</tr>
<tr>
<td>PSNR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lina</td>
<td>35.030</td>
<td>37.258</td>
<td>34.823</td>
<td>34.892</td>
<td>31.417</td>
<td>34.230</td>
<td>37.635</td>
</tr>
<tr>
<td>Man</td>
<td>35.051</td>
<td>37.281</td>
<td>34.875</td>
<td>34.926</td>
<td>31.438</td>
<td>34.251</td>
<td>37.669</td>
</tr>
</tbody>
</table>

Table 5
Filtering comparison of the mixed noise, $p = 0.2$ and $\sigma = 0.2$

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lina</td>
<td>0.00713</td>
<td>0.09569</td>
<td>0.13114</td>
<td>0.10452</td>
<td>0.12887</td>
<td>0.18112</td>
<td>0.00513</td>
</tr>
<tr>
<td>Man</td>
<td>0.00706</td>
<td>0.09442</td>
<td>0.13003</td>
<td>0.10337</td>
<td>0.12759</td>
<td>0.18005</td>
<td>0.00501</td>
</tr>
<tr>
<td>PSNR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lina</td>
<td>35.030</td>
<td>37.258</td>
<td>34.823</td>
<td>34.892</td>
<td>31.417</td>
<td>34.230</td>
<td>37.635</td>
</tr>
<tr>
<td>Man</td>
<td>35.082</td>
<td>37.271</td>
<td>34.835</td>
<td>34.911</td>
<td>31.434</td>
<td>34.245</td>
<td>37.648</td>
</tr>
</tbody>
</table>

Table 6
Filtering comparison of single Gaussian noise, $\sigma = 0.4$

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lina</td>
<td>0.00426</td>
<td>0.02844</td>
<td>0.03056</td>
<td>0.03148</td>
<td>0.02612</td>
<td>0.11253</td>
<td>0.00408</td>
</tr>
<tr>
<td>Man</td>
<td>0.00415</td>
<td>0.02831</td>
<td>0.03042</td>
<td>0.03131</td>
<td>0.02594</td>
<td>0.11239</td>
<td>0.00391</td>
</tr>
<tr>
<td>PSNR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please cite this article as: L. Ji, Z. Yi, A mixed noise image filtering method using weighted-linking PCNNs, Neurocomputing (2007), doi:10.1016/j.neucom.2007.04.015
perspective good performance for either single noise or mixed noise removal. Furthermore, the proposed one often achieves higher PSNR and less NMSE than the trilateral filter. Even for mixed noise ($p = 0.4$, $s = 0.4$), statistically, each NMSE is $0.017$ lower than that by the trilateral filter and PSNR is $2.1$ dB higher than that by the trilateral filter (see Table 8).

Figs. 13–18 exhibit filtering comparisons between four methods: trilateral filter [6], vector median filter [15], PCNN-based method [37] and the proposed one. The added noise mixture is ($p = 0.1$, $s = 0.1$) for the images in Figs. 13 and 14, ($p = 0.4$, $s = 0.4$) for the images in Figs. 15 and 16 and ($p = 0.7$, $s = 0.7$) for the images in Figs. 17 and 18. From these figures, for the mixed noise ($p \leq 0.4$, $s \leq 0.4$), all methods can effectively remove noise of images. However, for higher noise mix ($p > 0.4$, $s > 0.4$), filtering performance of each method gradually falls, and many pixel speckles are produced in these results, see Figs. 17 and 18. Even so, for these low-intensity noise images, the filtering output of our method is better than those of the
other three methods, and for the high-intensity noise, the proposed method is also better than the other three methods, from the viewpoint of vision.

Generally, the time expense of noise filtering algorithm is also an important evaluation criterion. In these experiments, the filtering time of each method is also tested using noise images, and these experimental results are listed in Table 9. These time expense comparisons show that the proposed method is the one that has the lowest filtering time expense. Statistically, it runs 22 ms faster than [37]. Based on Table 9 and Figs. 13–18, we see that the proposed one produces the best filtering result while consuming the least time. Though the proposed method is iterative, it can quickly reach the convergent times $t'$ (usually, $t' = 2$) under the condition $\Omega = 0.0005$. This is a primary reason why the proposed filtering algorithm has the lowest time expense.
5. Conclusions

To remove mixed noise in images, this paper proposes a weighted-linking PCNN model by (i) extending the linking input, (ii) defining an universal weighted-linking rule, (iii) redefining the real-time pulse threshold, (iv) modifying the computation rule of activity strength and (v) designing the pixel variation rule of neuron. Based on such a model, a two-channel parallel filter is constructed. By choosing suitable parameters, it can effectively remove single impulse noise \( (p > 0.0, \sigma = 0.0) \) and additional Gaussian noise \( (p = 0.0, \sigma > 0.0) \), as well as the mixed noise of them \( (p > 0.0, \sigma > 0.0) \). Experiments show that this method can generate good noise filtering results and has low filtering time expense.

Fig. 16. Filtering results on the image “Man” with mixed noise \( (p = 0.4, \sigma = 0.4) \): (a) original image; (b) noise image; (c) the proposed method; (d) the vector median filter \[15\]; (e) the PCNN-based method \[37\]; (f) the trilateral filter \[6\].

Fig. 17. Filtering results on the image “Lina” with mixed noise \( (p = 0.7, \sigma = 0.7) \): (a) original image; (b) noise image; (c) the proposed method; (d) the vector median filter \[15\]; (e) the PCNN-based method \[37\]; (f) the trilateral filter \[6\].
Fig. 18. Filtering results on the image “Man” with mixed noise ($p = 0.7, \sigma = 0.7$): (a) original image; (b) noise image; (c) the proposed method; (d) the vector median filter [15]; (e) the PCNN-based method [37]; (f) the trilateral filter [6].

Table 9
Time expense comparisons of noise filtering (unit: ms)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Time (ms)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$p = 0.1, \sigma = 0.1$</td>
<td>Lina</td>
<td>51</td>
<td>59</td>
<td>44</td>
<td>62</td>
<td>38</td>
</tr>
<tr>
<td></td>
<td>Man</td>
<td>49</td>
<td>59</td>
<td>43</td>
<td>61</td>
<td>36</td>
</tr>
<tr>
<td>$p = 0.4, \sigma = 0.4$</td>
<td>Lina</td>
<td>49</td>
<td>58</td>
<td>46</td>
<td>63</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>Man</td>
<td>50</td>
<td>56</td>
<td>46</td>
<td>63</td>
<td>41</td>
</tr>
<tr>
<td>$p = 0.7, \sigma = 0.7$</td>
<td>Lina</td>
<td>51</td>
<td>58</td>
<td>45</td>
<td>67</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>Man</td>
<td>52</td>
<td>57</td>
<td>47</td>
<td>66</td>
<td>40</td>
</tr>
</tbody>
</table>

References


Please cite this article as: L. Ji, Z. Yi, A mixed noise image filtering method using weighted-linking PCNNs, Neurocomputing (2007), doi:10.1016/j.neucom.2007.04.015


Mr. Luping Ji received his B.S. degree from Beijing Institute of Technology, Beijing, China, in July 1999 and M.S. degree from University of Electronic Science and Technology of China, Chengdu, China, in April 2005. Currently, he is a Ph.D. student in the Computational Intelligence Laboratory of University of Electronic Science and Technology of China, Chengdu, China. His research interests include Neural Networks and Pattern Recognition.

Dr. Zhang Yi received his Ph.D. degree in mathematics from the Institute of Mathematics, The Chinese Academy of Science, Beijing, China, in 1994. Currently, he is a Professor in the Computational Intelligence Laboratory, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China. He is the author of the book Convergence Analysis of Recurrent Neural Networks and coauthor of the book Neural Networks: Computational Models and Applications. His current research interests include Neural Networks and Data Mining.