

Filtrado de ruido Gaussiano mediante redes neuronales pulso-acopladas

Estela Ortiz Rangel¹, Manuel Mejía-Lavalle¹, Humberto Sossa²

¹ Centro Nacional de Investigación y Desarrollo Tecnológico,
Departamento de Ciencias Computacionales, Cuernavaca, Morelos,
México

² Instituto Politécnico Nacional,
Centro de investigación en Computación, Ciudad de México,
México

{estela_or, mlavalle}@cenidet.edu.mx, humbertosossa@gmail.com

Resumen. Se describe un algoritmo llamado ICM-TM para reducir el efecto del ruido Gaussiano en imágenes monocromáticas. La operación del algoritmo se basa en el Modelo de Intersección Cortical (ICM) que es un tipo de Red Neuronal Artificial tipo Pulso-Acoplado. Una matriz de tiempos (TM) proporciona la información correspondiente a la iteración cuando la neurona correspondiente se activa por primera vez. Se establece un criterio de filtrado selectivo que combina el operador de mediana y promedio tomando como base el tiempo de activación de las neuronas. El desempeño del algoritmo propuesto se evaluó experimentalmente con ruido Gaussiano a varios niveles. Los resultados muestran la efectividad de la propuesta con respecto a los filtros mediana, Gaussiano, Sigma, Wiener y las Redes Neuronales Pulso-Acopladas tipo PCNNNI. Los resultados son representados principalmente a través del Cociente Pico Señal a Ruido (CPSR).

Palabras clave. Modelo de Intersección Cortical (ICM), ruido Gaussiano, filtro Wiener, relación pico señal a ruido (PSNR).

Using Pulse Coupled Neural Networks to Improve Image Filtering Contaminated with Gaussian Noise

Abstract. An algorithm called ICM-TM to reduce the effect of Gaussian noise in grayscale images is proposed. It is based on the operation of the well-

known Intersection Cortical Model (ICM), a kind of Pulse-Coupled Artificial Neural Network. A Time Matrix (TM) provides information about the iteration when the neuron fires for first time. Each neuron corresponds to a pixel. A selective filtering criteria that combines the median and average operators using the neuron's activation time is established. The performance of the proposed algorithm is evaluated experimentally with varying degrees of Gaussian noise. Simulation results show that the effectiveness of the method is superior to the median filter, Gaussian filter, Sigma filter, Wiener filter and to the Pulse-Coupled Neural Networks with the Null Interconnections (PCNNNI). Results are mainly provided by the parameter Peak Signal to Noise Ratio (PSNR).

Keywords. Intersection Cortical Model (ICM), Gaussian noise, Wiener filter, Peak Signal to Noise Ratio (PSNR).

1. Introducción

El ruido aditivo Gaussiano q es un modelo que permite simular la afectación aleatoria de los píxeles de una imagen con valores uniformemente distribuidos. Su función de densidad de probabilidad $p_q(x)$ viene dada en términos de la media μ y la varianza σ^2 de una variable aleatoria x . Se expresa en la siguiente ecuación [1]:

$$p_q(x) = (2\pi\sigma^2)^{-1/2} e^{-(x-\mu)^2/2\sigma^2}. \quad (1)$$

Este tipo de ruido es muy común en las imágenes digitales. Debido a sus características es difícil de eliminar por completo.

El filtro Wiener $H(u, v)$ es un filtro frecuencial como se expresa en la ecuación 2 [2]:

$$H(u, v) = \frac{D^*(u, v)}{|D(u, v)|^2 + \Gamma} = \frac{1}{1 + \frac{S_w(u, v)}{S_f(u, v)}} \quad (2)$$

Se basa en la reducción del error cuadrático medio; mejora sustancialmente la calidad de una imagen cuando ésta es afectada con ruido Gaussiano. Requiere, sin embargo del cálculo del espectro de energía de la imagen ideal $S_w(u, v)$, del espectro de energía del ruido $S_f(u, v)$, cuyo cociente se aproxima por un coeficiente Γ , y de la estimación de una función de degradación $D(u, v)$ y su conjugado $D^*(u, v)$. Algunas modificaciones a este filtro han sido propuestas con el fin de mejorar su desempeño y generalizar su utilización [3,4]. El cálculo aún depende de las estimaciones de dichos coeficientes, lo cual dificulta su aplicación.

El filtro Sigma también es efectivo en la eliminación del ruido Gaussiano; es ampliamente utilizado por su simplicidad, no obstante, su capacidad de preservar bordes aún se sigue mejorando, por ejemplo, por medio del pre-procesamiento de la imagen dividiéndola en dos componentes a las que se aplica el filtro Sigma por separado [5], lo cual mejora la preservación de bordes pero aumenta el tiempo de procesamiento del algoritmo y depende de un estimador del ruido para cada imagen.

Se han propuesto técnicas que utilizan la información de los bordes de la imagen contaminada, donde se aplican los principios de similitud [6]. No obstante, su proceso computacional es largo; su implementación en tiempo real en sistemas embebidos es difícil.

Entre las técnicas más recientes de filtrado de ruido Gaussiano se encuentran los algoritmos basados en onditas (wavelets) multidireccionales [7], que pueden remover el ruido y preservar los detalles como bordes y texturas. Tal es el caso del método de filtrado basado en la Transformada Directionlet que permite la reducción multidireccional al construir una matriz con las principales direcciones encontradas en la imagen para evitar eliminar bordes. Este algoritmo es

laborioso y no toma en cuenta la información de similitud espacial entre los píxeles por lo que puede llevar a falsas detecciones.

Otras técnicas típicas de filtrado son los métodos de representación sobre-completa basados en bloques, como son el algoritmo de promedio ponderado óptimo [8], el BM3D [9] y el K-SVD [10]. Recientemente se desarrolló el SM3D-DCTNS [11] que consiste en la agrupación de bloques coincidentes con filtrado por medio de la transformada discreta coseno, el umbralado, la segmentación y un procesamiento posterior de restauración de bordes.

La técnica de filtrado mediante la integral fraccional de Alexander [9] trabaja mediante la construcción de máscaras fraccionales a partir del llamado polinomio de Alexander. Por medio de esta técnica la imagen es descompuesta a través de wavelets de los cuales sólo se filtran los componentes diagonal, vertical y horizontal. Dos de las principales desventajas de este método son la descomposición de la imagen en sub-ondas así como el cálculo de coeficientes de forma experimental.

Dentro de las técnicas de inteligencia artificial también se han hecho propuestas para mejorar el filtrado de ruido Gaussiano, las más prominentes son las que emplean el aprendizaje para máquinas como el Ridgelet Support Vector Filter (RSVM) [12] que se basan en la generación de Diccionarios Multiescala generados mediante aprendizaje a partir de ejemplos (MRSVF y NCSR) y que permiten extraer las características sobresalientes asociadas con singularidades lineales y así evitar su desaparición. Dicho método consiste en el cálculo de una serie de matrices, la extracción de vectores columna centrales y la comparación con los diccionarios de las características estadísticas de las imágenes con ruido para después minimizar el error en un laborioso e iterativo proceso.

Una forma distinta de enfrentar el problema de eliminación del ruido Gaussiano en imágenes digitales ha surgido de la exploración experimental, tal es el caso de las Redes Neuronales Artificiales de tercera generación llamadas Redes Neuronales Pulso-Acopladas o por sus siglas en inglés PCNN (Pulse Coupled Neural Networks), las cuales han sido empleadas de modo eficiente para el procesamiento de

imágenes en diversas tareas como la segmentación, la clasificación, la identificación de imágenes, entre otras [13].

Las redes tipo PCNN son un modelo matemático, propuesto por Eckhorn. Se basa en la frecuencia de activación de las neuronas de la corteza visual de los mamíferos [14,15]. El tiempo y frecuencia de activación de las neuronas y han sido utilizados para procesar imágenes gracias al modelo computacional simplificado de PCNN propuesto por Ranganath y Kuntimad [16].

Las propiedades del modelo PCNN permiten a cada neurona procesar un pixel y relacionar su nivel de gris con el de sus vecinos. Al iterar la Red Neuronal el tiempo de activación de cada neurona se registra en una matriz. Esta información puede ser utilizada para la detección de los pixeles con ruido y la aplicación de una técnica selectiva de filtrado de ruido Gaussiano [17, 18], para la reducción del ruido Gaussiano y de sal y pimienta mezclados [19, 20].

Diversos modelos simplificados de PCNN han sido propuestos para trabajar computacionalmente con imágenes. Dos de las principales variaciones son el Intersection Cortical Model (ICM) y el Pulse-Coupled Neural Networks with the Null Interconnections (PCNNNI) [21]. Al combinarlos con el operador de mediana, promedio y morfológico o con el filtro Wiener y la matriz de tiempos (TM) se logra reducir el ruido de Sal y Pimienta y el ruido Gaussiano en imágenes digitales, no obstante la restauración de la imagen es deficiente.

Dentro de los métodos de filtrado basados en las propiedades de las PCNN que pretenden mejorar la calidad de la imagen restaurada se encuentra la propuesta de Zhang [22]. Este método utiliza un algoritmo genético adaptativo y un filtro de difusión anisotrópica para disminuir el ruido Gaussiano. No obstante sólo es útil para valores de varianza del ruido menores a 0.009 e implica gran cantidad de cálculos para completar el filtrado.

El método ICM-TM que se propone consiste en utilizar una red ICM para generar la matriz de tiempos y aplicar selectivamente los operadores mediana y promedio para suprimir el ruido Gaussiano en imágenes digitales en escala de grises, de forma que se logre una reducción significativa del ruido con un método

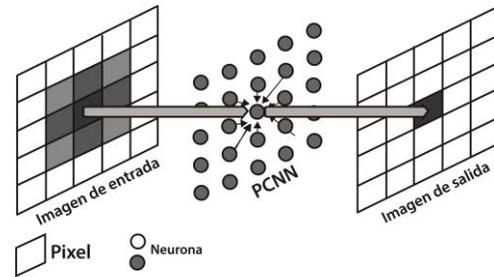


Fig. 1. Diagrama de mapeo de la red PCNN

computacional sencillo que puede ser implementado para aplicaciones con sistemas embebidos.

2. Los modelos PCNN e ICM

Las Redes Neuronales Pulso-Acopladas son un sistema que permite emular efectivamente a las neuronas biológicas de la corteza visual de los mamíferos. Han sido aplicadas en variedad de dominios, especialmente en el procesamiento de imágenes para la remoción de ruido, reconocimiento de objetos, optimización, adelgazamiento, segmentación, fusión, identificación y remoción de sombras [14, 23].

En el ámbito del procesamiento de imágenes las diferencias entre las Redes Neuronales Artificiales tradicionales y las Redes Neuronales de tercera generación son evidentes tanto en su configuración como en su operación. Las principales diferencias son que a cada neurona corresponde un pixel y la imagen de salida es binaria (Fig. 1).

Las redes tipo PCNN (Fig. 2) no requieren de un entrenamiento. Su función es clasificar los píxeles por sus niveles de intensidad. En este modelo, cada pixel de la imagen se corresponde con una neurona. Su valor S_{ij} es introducido a la red por medio de una señal llamada "feeding" F_{ij} . El umbral de activación de las neuronas T_{ij} es dinámico. Además, cada neurona recibe información de sus vecinas a través de una sinapsis, lo cual se conoce como "linking" L_{ij} .

La señal *feeding* y la de *linking* ponderada por un factor de unión β se combinan para formar el

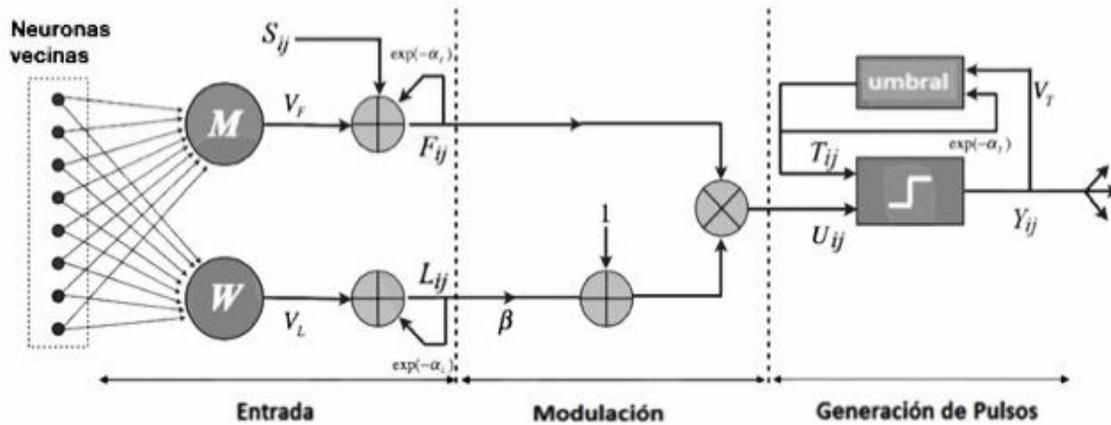


Fig. 2. Diagrama general de interconexiones de la red PCNN [14]

potencial interno de la neurona U_{ij} , que al ser comparado con el umbral produce salidas binarias Y_{ij} . Dichas señales se ven afectadas por factores de atenuación V_F, V_L, V_T y de amplificación $\alpha_F, \alpha_L, \alpha_T$; la imagen se va procesando de acuerdo al tamaño de las matrices M y W .

Estas características hacen que las neuronas correspondientes a pixeles vecinos con valores de intensidad similares se activen al mismo tiempo en ciertas regiones, a lo que se denomina activación de pulsos síncrona [23].

El modelo original de PCNN presenta algunas limitantes en la práctica debido al gran número de interconexiones entre neuronas y al establecimiento de parámetros de operación. Por este motivo se obtuvo un modelo iterativo simplificado de PCNN que permite extender el uso de estas redes en el procesamiento de imágenes. El modelo computacional iterativo ICM se describe como sigue (3)-(5) [21]:

$$F_{ij}[n] = fF_{ij}[n-1] + \sum w_{ijkl}Y_{kl}[n-1] + S_{ij} \quad (3)$$

$$T_{ij}[n] = gT_{ij}[n-1] + hY_{ij}[n-1] \quad (4)$$

$$Y_{ij}[n] = \begin{cases} 1 & \text{if } F_{ij}[n] > T_{ij}[n] \\ 0 & \text{en otro caso} \end{cases} \quad (5)$$

donde n es la iteración actual, w_{ijkl} es la matriz de pesos sinápticos que liga una neurona con sus

vecinas y puede tener alguna de las siguientes formas:

$$w_{ijkl}(5 \times 5) = \begin{bmatrix} 0.12 & 0.25 & 0.50 & 0.25 & 0.12 \\ 0.25 & 0.50 & 1.00 & 0.50 & 0.25 \\ 0.50 & 1.00 & 0 & 1.00 & 0.25 \\ 0.25 & 0.50 & 1.00 & 0.50 & 0.25 \\ 0.12 & 0.25 & 0.50 & 0.25 & 0.12 \end{bmatrix}$$

$$w_{ijkl}(3 \times 3) = \begin{bmatrix} 0.50 & 1.00 & 0.50 \\ 1.00 & 0 & 1.00 \\ 0.50 & 1.00 & 0.50 \end{bmatrix}$$

y finalmente f, g y h son coeficientes de ajuste, donde típicamente $g < 1.0$, $f < g$ y h es un valor grande, típicamente de entre 20 y 40.

ICM es un tipo de PCNN más simple (Fig. 3), donde no se considera *linking* en las neuronas y donde el *feeding* F_{ij} mantiene su salida con un factor de decaimiento dado por f .

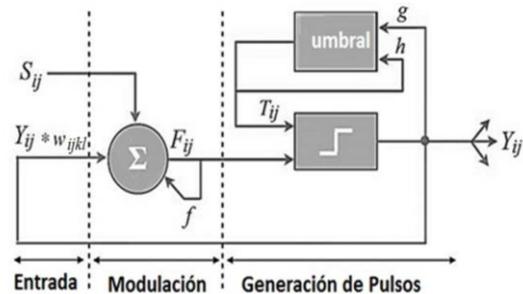


Fig. 3. Diagrama general de la red ICM [14]

La señal F_{ij} se compone por la última salida del vecindario de neuronas Y_{ij} ponderada por una matriz w_{ijkl} (generalmente Gaussiana) y por el estímulo externo de entrada S_{ij} (nivel de gris de cada pixel normalizado entre 0 y 1).

El umbral dinámico para cada pixel T_{ij} crece obedeciendo a h cuando su salida se activa y mantiene su estado previo con una atenuación dada por g , lo cual da origen a la formación de pulsos.

3. La matriz de tiempos (TM)

Las redes tipo PCNN pueden ser utilizadas para determinar la posición de los pixeles ruidosos con base en las neuronas activadas a la salida de la red por iteración. Al aplicar un operador de mediana que elimine los valores más altos y más bajos, es posible eliminar el ruido de Sal y Pimienta.

Para el filtrado del ruido Gaussiano es necesario emplear otra técnica ya que todos los pixeles han sido afectados en algún grado, por lo que se introduce la matriz de tiempos de las mismas dimensiones que la imagen a tratar. Esta matriz contiene información relacionada con la estructura espacial de la imagen, es decir, realiza un mapeo de la información espacial en una secuencia temporal [16].

La matriz de tiempos obtenida a partir de una red neuronal con interconexión nula PCNNNI [21] ha sido utilizada para detectar los pixeles ruidosos y procesada para reducir el ruido Gaussiano combinándola con otros métodos [17]. El modelo PCNNNI no considera la señal de *linking* y elimina la influencia de la matriz de pesos sinápticos, el potencial interno, umbral y salida de una neurona dependen únicamente de la intensidad del pixel correspondiente.

El método propuesto ICM-TM consiste en obtener la matriz de tiempos conservando la información de la relación espacial entre pixeles, de modo que las diferencias en la intensidad de los pixeles de la imagen originen diferencias en la secuencia de activación de sus respectivas neuronas (Fig. 4). Para almacenar el tiempo de activación de cada neurona se define una matriz M_{ij} que puede ser descrita como (6) [18].

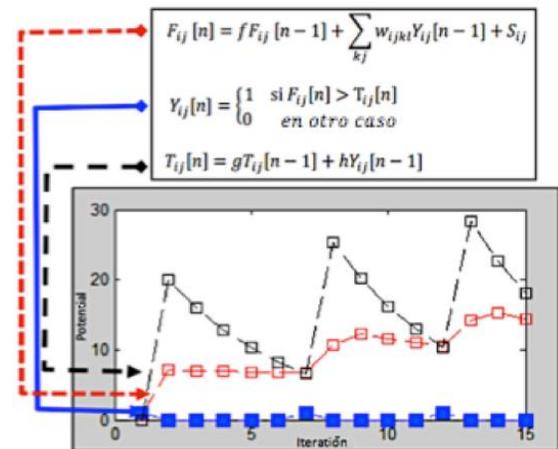


Fig. 4. Secuencias de activación de una neurona de la red ICM [14]



Fig. 5. Diagrama del algoritmo propuesto

$$M_{ij}[n] = \begin{cases} n & \text{if } Y_{ij}[n] = 1, \\ M_{ij}[n-1] & \text{en otro caso.} \end{cases} \quad (6)$$

El algoritmo ICM-TM mantiene el rango dinámico de la imagen y complementa el proceso de filtrado obteniendo la información de la matriz de tiempos para cualquier número de iteraciones, suavizando regiones de 5×5 pixeles con el filtro promedio para suavizar la imagen o bien conservando información de la imagen pero suprimiendo valores extremos mediante el operador de mediana.

4. Descripción del algoritmo ICM-TM propuesto

En nuestro método todas las neuronas están ligadas mutuamente del mismo modo y sus salidas Y_{ij} tienen dos estados posibles, activado

(1) y no activado (0). Este modelo es más rápido que PCNN debido a que implica menos ecuaciones y es más adaptable ya que tiene menos parámetros que ajustar. El proceso principal del algoritmo ICM-TM se lleva a cabo de manera secuencial en bloques (Fig. 5) y el algoritmo desglosado se describe en la Tabla 1.

Tabla 1. Algoritmo del método propuesto

Algoritmo ICM-TM

Inicialización:

Paso 1. Normalizar entre 0 y 1 los valores de gris de la imagen original con ruido O_{ij} .

Paso 2. Inicializar los valores de las matrices Y_{ij} , S_{ij} , T_{ij} , F_{ij} y M_{ij} en 0, sus dimensiones son las mismas que las de la imagen de entrada. Pasar los valores normalizados a la red por medio de S_{ij} .

Llenado de la matriz de tiempos:

Paso 3. Calcular el valor de las matrices F_{ij} , T_{ij} , Y_{ij} de acuerdo a las ecuaciones 3, 4 y 5 en dicho orden.

Paso 3.a Cuando una neurona en la posición (i, j) tome el valor de activada por primera vez ($Y_{ij}[n] = 1$) registrar el número de iteración actual n en la matriz de tiempos $M_{ij}[n]$ en el lugar correspondiente al pixel procesado, omitiendo la primera iteración.

Paso 3.b Continuar iterando desde el Paso 3 hasta que todos los pixeles se hayan activado por lo menos una vez, es decir cuando todos los elementos en $M_{ij}[n]$ sean distintos de cero.

Filtrado de la imagen:

Paso 4. Recorrer la imagen O_{ij} y la matriz $M_{ij}[n]$ con una ventana cuadrada W de 5×5 pixeles, calcular la mediana m_{ij} dentro de W en $M_{ij}[n]$ y aplicar el siguiente criterio:

Paso 4.a Si el tiempo de activación n de la neurona (i, j) es igual a m_{ij} , el valor del pixel (i, j) en la imagen filtrada Sdn_{ij} será el valor resultante de aplicar el operador promedio p_{ij} sobre esta vecindad en O_{ij} .

Paso 4.b Si no el valor del pixel (i, j) de Sdn_{ij} será el resultante de aplicar el operador de mediana m_{ij} en O_{ij} .

5. Experimentos y resultados

Las pruebas experimentales fueron realizadas con imágenes estándar en escala de grises de 255 valores de la base de datos SIPI [24], con tamaño de 512×512 pixeles.

Durante el proceso de experimentación, se introdujo a las imágenes ruido Gaussiano de media 0 y varianza entre 0.01 y 0.09.

Los otros parámetros fueron empíricamente seleccionados para la simulación.

La matriz interna de pesos formada con valores Gaussianos en función de la distancia seleccionada fue $w_{ijkl}(3 \times 3)$. Las constantes: $f = 0.9$, $g = 0.8$, $h = 20$.

En el paso 4, el operador de mediana y el operador promedio están definidos de acuerdo a lo siguiente: si x_{ij} denota al pixel con coordenadas (i, j) en la imagen con ruido y X_{ij} denota el conjunto de pixeles en la ventana W con el vecindario $(2K + 1) \times (2K + 1)$ centrada en x_{ij} , entonces:

$$X_{ij} = \{x_{i-K, j-K}, \dots, x_{ij}, \dots, x_{i+K, j+K}\}. \quad (7)$$

La mediana de la ventana de la imagen se define como

$$m_{ij} = \text{mediana}(X_{ij}). \quad (8)$$

El promedio de la ventana de la imagen se define como

$$p_{ij} = \text{promedio}(X_{ij}). \quad (9)$$

El algoritmo ICM-TM se comparó contra filtros clásicos, el filtro de mediana de 3×3 , el filtro Gaussiano, el filtro Sigma y el filtro Wiener de 3×3 , y contra el PCNNNI basado en el algoritmo de Ma [21].

Se realizó la medición de los resultados obtenidos con tres métricas, las cuales están formuladas como sigue [19]:

a) *PSNR* (Peak Signal to Noise Ratio, dB), el cual es utilizado para medir la habilidad de supresión del ruido. Mientras más grande sea su valor mejor es el efecto del filtrado:

$$PSNR = 10 \log_{10} \left(\frac{f(m,n)^2}{\frac{1}{MN} \sum_{m=1}^M \sum_{n=1}^N [f(m,n) - f'(m,n)]^2} \right). \quad (10)$$

b) *MAE* (Mean Absolute Error), indica la calidad del filtrado como la preservación de detalles finos, para lo que cual de ser minimizado:

$$MAE = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N |f(m,n) - f'(m,n)|, \quad (11)$$

donde M y N denotan las filas y columnas de la imagen, $f(m,n)$ denota la imagen sin ruido y $f'(m,n)$ es la imagen resultante del proceso de filtrado.

c) *NMSE* (Normalized Mean Square Error), un mejor método de filtrado debe generar un menor valor resultante de *NMSE*:

$$NMSE = \frac{\sum_{m=1}^M \sum_{n=1}^N [f(m,n) - f'(m,n)]^2}{\sum_{m=1}^M \sum_{n=1}^N [f(m,n)]^2}. \quad (12)$$

De manera adicional también se midió el tiempo de cómputo, utilizando Matlab 2014 en Windows 7 Ultimate SP1, con un procesador Intel Core i7 a 3.40 GHz y 8GB de RAM.

6. Desempeño del método propuesto

Para la red ICM cada salida es distinta en cada iteración y la activación de las neuronas obedece a la relación entre los niveles de gris de la imagen con ruido Gaussiano.

En este caso de media 0 y varianza 0.01 (Fig. 6). La red puede iterarse cuantas veces se requiera. Para el caso de filtrado de ruido Gaussiano se requiere encontrar información sobre el grado de contaminación de cada pixel, por lo que se analizó la matriz de tiempos.

La información de esta matriz es numérica, por lo que se pueden distinguir los pixeles que se activan por primera vez en cada iteración y se extraen los pixeles que guardan mayor

información de la imagen, discriminando a los otros para eliminar la mayor cantidad de ruido no deseado en la imagen.

Para dar un ejemplo numérico del procedimiento de filtrado considérese una matriz correspondiente a una imagen en escala de grises, contaminada con ruido Gaussiano de media 0 y varianza 0.08, de la que se tomó una ventana de 6x6 pixeles O_{ij} y su correspondiente imagen normalizada S_{ij} :

$$O_{ij} = \begin{pmatrix} 238 & 139 & 255 & 239 & 112 & 182 \\ 68 & 235 & 59 & 255 & 178 & 42 \\ 179 & 52 & 138 & 227 & 116 & 81 \\ 238 & 45 & 114 & 67 & 205 & 51 \\ 147 & 54 & 148 & 128 & 37 & 255 \\ 180 & 0 & 141 & 25 & 110 & 56 \end{pmatrix},$$

$$S_{ij} = \begin{pmatrix} 0.93 & 0.54 & 1.00 & 0.93 & 0.43 & 0.71 \\ 0.26 & 0.92 & 0.23 & 1.00 & 0.69 & 0.16 \\ 0.70 & 0.20 & 0.54 & 0.89 & 0.45 & 0.31 \\ 0.93 & 0.17 & 0.44 & 0.26 & 0.80 & 0.20 \\ 0.57 & 0.21 & 0.58 & 0.50 & 0.14 & 1.00 \\ 0.70 & 0 & 0.55 & 0.09 & 0.43 & 0.21 \end{pmatrix}.$$

Se introduce la imagen normalizada en la red ICM y se itera 10 veces, obteniéndose el umbral dinámico de dicha iteración T_{ij} , los valores en x son indistintos para este ejemplo, tal es el caso de los bordes de la imagen:

$$T_{ij} = \begin{pmatrix} 16.15 & 19.35 & 16.15 & 16.15 & 19.35 & x \\ 19.35 & 13.59 & 16.15 & 13.59 & 16.15 & x \\ 19.35 & 19.35 & 16.15 & 16.15 & 19.35 & x \\ 16.15 & 19.35 & 19.35 & 19.35 & 16.15 & x \\ 19.35 & 23.35 & 19.35 & 19.35 & 23.35 & x \\ x & x & x & x & x & x \end{pmatrix}.$$

Luego se obtiene el potencial interno de la matriz F_{ij} :

$$F_{ij} = \begin{pmatrix} 9.31 & 8.47 & 11.48 & 11.07 & 5.92 & x \\ 6.79 & 13.76 & 8.87 & 14.04 & 9.60 & x \\ 9.71 & 8.96 & 11.15 & 13.38 & 7.92 & x \\ 11.45 & 9.09 & 10.80 & 9.51 & 10.61 & x \\ 7.09 & 6.65 & 9.20 & 8.64 & 4.18 & x \\ x & x & x & x & x & x \end{pmatrix}.$$

Si el potencial interno de la neurona (ij) supera al valor de umbral, entonces se produce una salida en 1 para esa neurona.

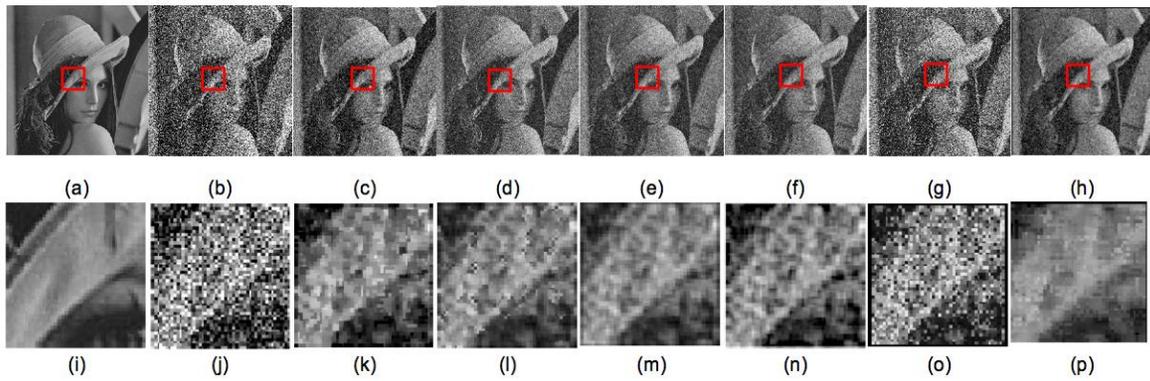


Fig. 7. Comparación entre técnicas de filtrado a) Lena original sin ruido, b) con ruido Gaussiano var. 0.08, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

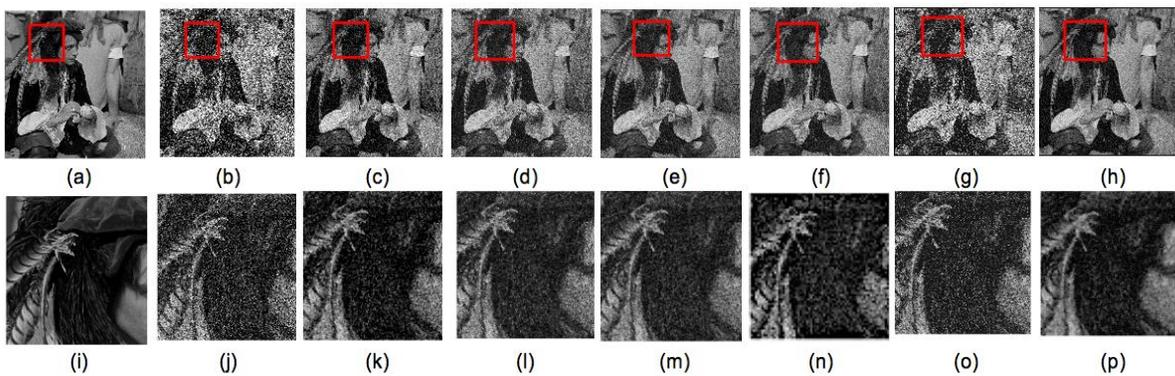


Fig. 8. Comparación entre técnicas de filtrado a) Man original sin ruido, b) con ruido Gaussiano var. 0.08, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

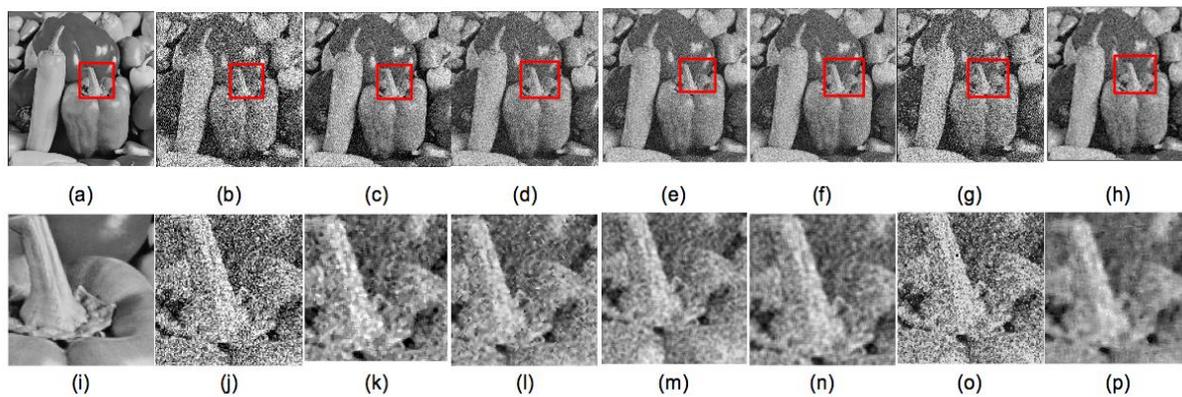


Fig. 9. Comparación entre técnicas de filtrado a) Peppers original sin ruido, b) con ruido Gaussiano var. 0.09, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

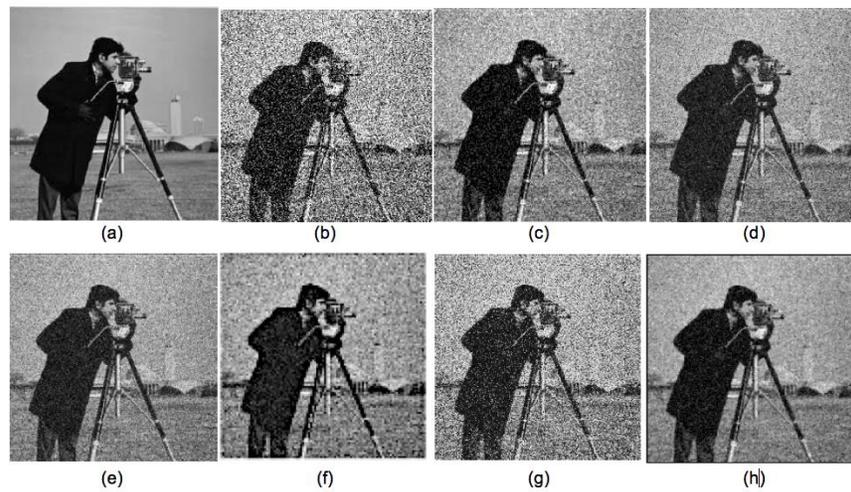


Fig. 10. Comparación entre técnicas de filtrado a) cameraman original sin ruido, b) con ruido Gaussiano var. 0.08, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

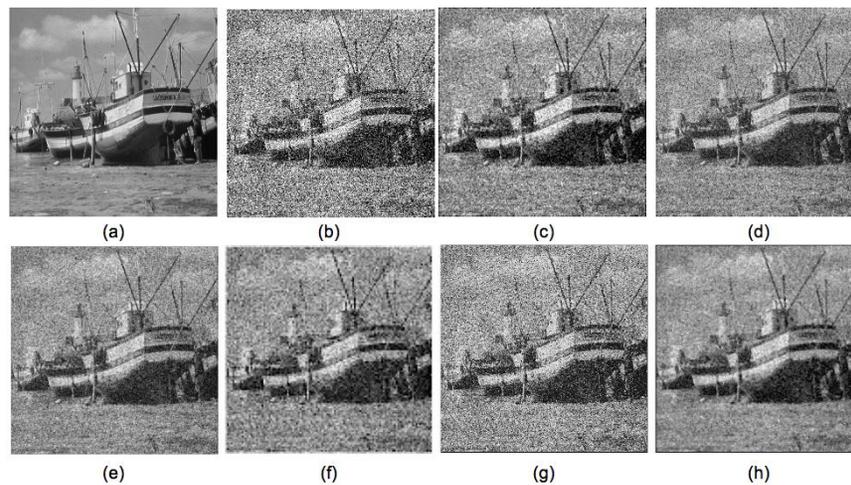


Fig. 11. Comparación entre técnicas de filtrado a) boat original sin ruido, b) con ruido Gaussiano var. 0.09, c) mediana, d) Wiener, e) Gaussiano, f) Sigma, g) PCNNNI y h) ICM-TM

Al terminar de procesar la imagen se tiene una matriz binaria de salidas Y_{ij} .

$$Y_{ij} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

El número de iteración en el que sucede la primera activación de cada neurona se guarda en la matriz de tiempos final M_{ij} :

$$M_{ij} = \begin{pmatrix} 7 & 8 & 7 & 7 & 8 & x \\ 8 & 6 & 7 & 6 & 7 & x \\ 8 & 8 & 7 & 7 & 8 & x \\ 7 & 8 & 8 & 8 & 7 & x \\ 8 & 9 & 8 & 8 & 9 & x \\ x & x & x & x & x & x \end{pmatrix}.$$

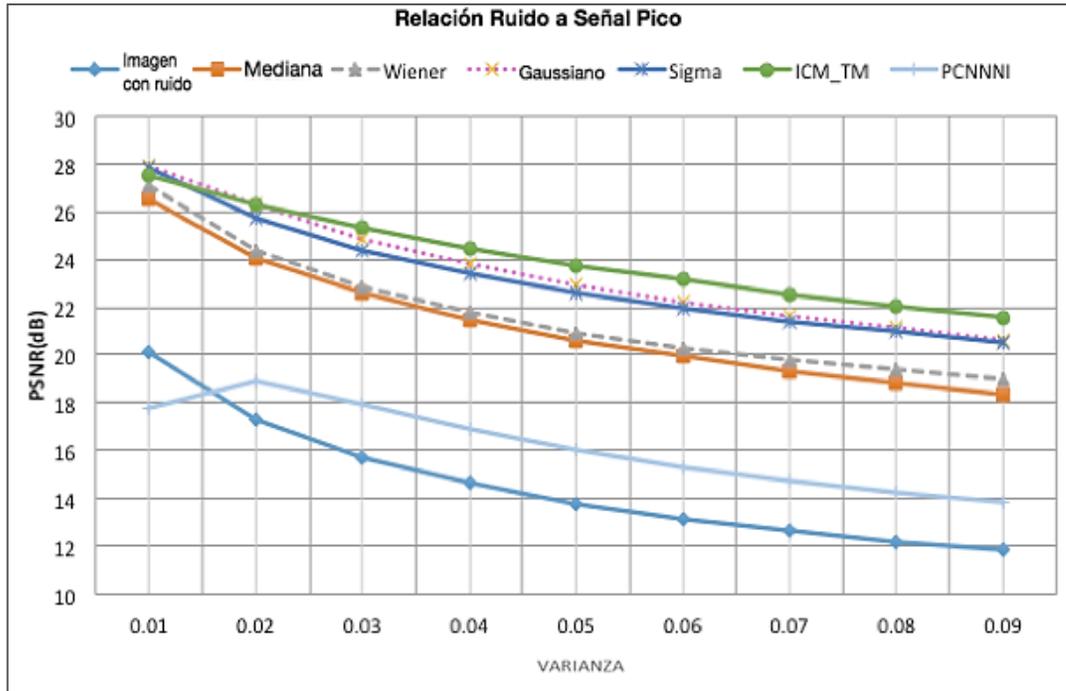


Fig. 12. Curvas de desempeño en PSNR de los filtros de ruido Gaussiano

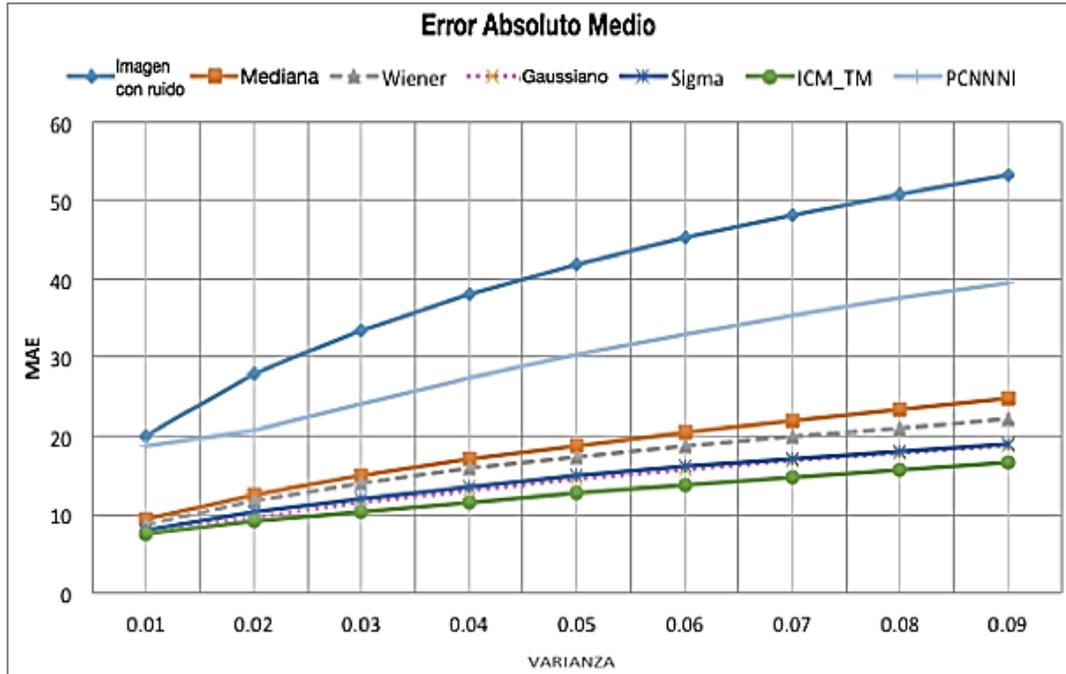


Fig. 13. Curvas de desempeño en MAE de los filtros de ruido Gaussiano

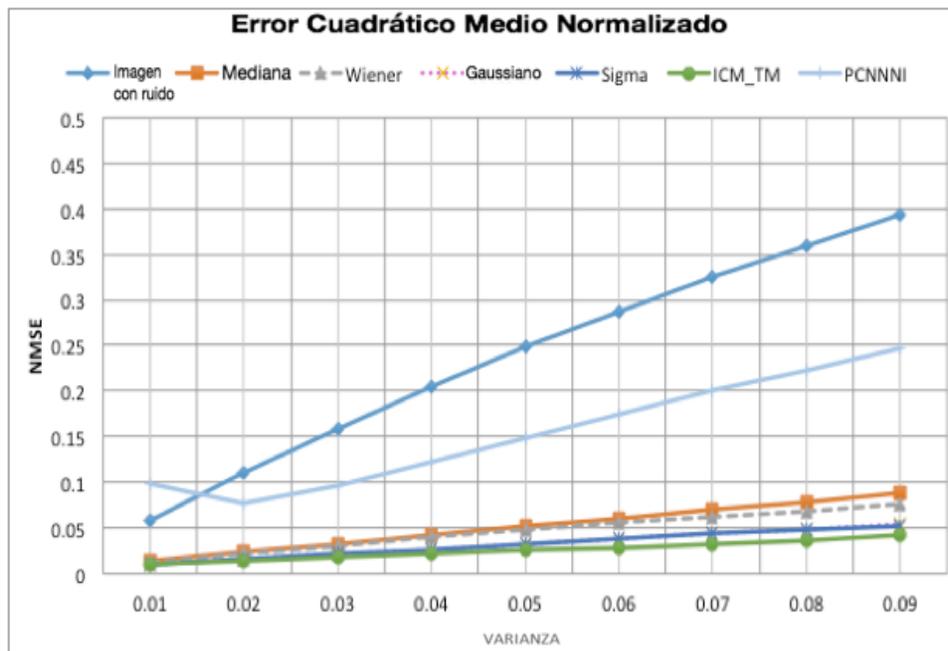


Fig. 14. Curvas de desempeño en NMSE de los filtros de ruido Gaussiano

Tabla 2. PSNR para imágenes con ruido Gaussiano de varianza 0.08

PSNR (dB)	Con ruido	Mediana	Wiener	Gaussiano	Sigma	PCNNNI	ICM-TM
Lena	12.20	18.81	19.39	21.15	20.96	14.27	22.07
Baboon	11.96	17.42	18.54	19.42	19.14	13.91	19.05
Camán.	12.24	19.07	19.02	20.78	20.60	14.27	22.34
Peppers	12.16	18.79	19.32	21.05	20.86	14.21	22.30
Barbara	13.66	20.11	18.58	19.96	19.76	15.62	22.05
Cerebro	12.97	20.10	17.88	19.64	19.45	14.66	22.27
5.1.13	13.96	18.30	16.88	16.25	15.94	8.24	15.80
Lunetas	12.34	18.78	19.70	21.43	21.24	14.59	22.37
Man	12.37	19.02	18.70	20.41	20.21	14.32	21.72
Boat	12.02	18.52	19.18	20.73	20.51	14.04	21.44
Couple	13.36	19.92	18.40	19.94	19.74	15.24	21.60
Grass	12.54	15.53	16.25	16.25	15.91	13.92	13.93
House	12.07	18.83	19.57	21.24	21.04	14.14	22.12
Mountain	13.61	19.92	18.59	19.95	19.75	15.58	21.92

La etapa de filtrado selectivo consiste en recorrer las matrices calculando la mediana de los valores de la matriz de tiempos en una ventana de 5×5 denotado como

mediana $M_{2,2}(5 \times 5)$ y comparar el valor con el correspondiente al pixel central $M_{2,2}$ de la ventana considerada, cuyos valores respectivos son:

Tabla 3. PSNR para imágenes con ruido Gaussiano de varianza 0.09

PSNR (dB)	Con ruido	Mediana	Wiener	Gaussiano	Sigma	PCNNNI	ICM-TM
Lena	11.81	18.32	18.97	20.63	20.49	13.84	21.59
Baboon	11.60	17.08	18.26	19.13	18.91	13.54	18.89
Caman.	11.89	18.63	18.67	20.36	20.22	13.93	21.93
Peppers	11.79	18.32	18.96	20.61	20.47	13.84	21.85
Barbara	13.17	19.66	18.03	19.32	19.18	15.07	21.62
Cerebro	12.57	19.65	17.49	19.13	18.99	14.24	21.77
5.1.13	13.44	17.99	16.39	15.85	15.60	8.70	15.59
Lunetas	12.00	18.42	19.38	20.98	20.84	14.20	21.94
Man	11.98	18.56	18.32	19.93	19.78	13.91	21.29
Boat	11.64	18.09	18.89	20.34	20.18	13.67	21.11
Couple	12.88	19.43	17.90	19.30	19.15	14.68	21.11
Grass	12.14	15.28	16.01	16.01	15.72	13.59	13.84
House	11.73	18.26	19.18	20.76	20.62	13.81	21.66
Mountain	13.13	19.55	18.04	19.35	19.20	15.04	21.51

Tabla 4. PSNR de los filtros de ruido Gaussiano en *Lena* 512x512

Desviación estándar	Con ruido	Mediana	Wiener	Gaussiano	Sigma	PCNNNI	BM3D [25]	ICM-TM
5	34.00	32.86	35.00	32.48	32.22	10.37	38.72	29.36
10	28.28	31.35	33.04	31.63	31.41	11.83	35.93	29.01
15	24.62	29.64	30.82	30.42	30.26	13.83	34.27	28.60
20	22.16	28.09	28.89	29.21	29.10	15.82	33.05	28.11
25	20.34	26.69	27.23	28.06	27.98	17.60	32.08	27.60
30	18.81	25.47	25.85	26.96	26.91	18.80	31.26	27.00
35	17.55	24.29	24.62	25.96	25.93	18.93	30.56	26.40
40	16.49	23.31	23.57	25.04	25.02	18.54	29.86	25.79
45	15.57	22.45	22.71	24.30	24.28	17.81	29.05	25.24
50	14.78	21.64	21.94	23.55	23.54	17.05	28.27	24.59

$$\text{mediana } M_{2,2}(5 \times 5) = 8,$$

$$M_{2,2} = 7.$$

Si estas cifras son iguales, el valor del pixel correspondiente en la imagen filtrada $Sdn_{2,2}$ será el promedio de los valores en esta ventana en la imagen ruidosa $O_{2,2}(5 \times 5)$, en caso contrario el pixel correspondiente tomará el valor de la mediana de la vecindad de 5×5 de la imagen con ruido. Se procede a calcular la mediana de la ventana en la imagen con ruido $O_{2,2}(5 \times 5)$ y asignarlo al pixel de la imagen filtrada $Sdn_{2,2}$. El proceso termina cuando se ha procesado la imagen completa:

$$\text{mediana } O_{2,2}(5 \times 5) = 138,$$

$$Sdn_{2,2} = 138.$$

Se realizó la comparación del método propuesto con los filtros de mediana, Wiener, Gaussiano, Sigma y el basado en PCNNNI para la imagen de *Lena* (Fig. 7) a la que se agregó ruido Gaussiano con varianza 0.08; se observó que la capacidad de suavizado del filtro de mediana puede ser empleado para disminuir el ruido Gaussiano que genera valores extremos en los pixeles, no obstante, no se debe aplicar este filtro de manera uniforme pues la imagen se verá afectada en detalles y bordes. Para otras regiones se debe tomar en cuenta la información presente en los pixeles por medio de un suavizado más fino como es el generado por el operador promedio.

El análisis cualitativo de la imagen ampliada hace evidente que el filtro Wiener mantiene un alto grado de ruido aunque preserva los bordes.

El filtro Gaussiano preserva bordes, disminuye el ruido al suavizar la imagen pero la imagen no es nítida.

El filtro Sigma recupera la imagen en buen grado preservando los bordes de modo claro pero el ruido aún es notorio, mientras que el método PCNNNI no elimina la mayor parte del ruido presente en la imagen. Finalmente el método propuesto muestra un mayor efecto de suavizado, los bordes mejor delimitados y el ruido menos presente.

Se repitió la prueba con distintas imágenes obteniéndose resultados similares como puede observarse para la imagen de *Man* (Fig. 8) con ruido Gaussiano de varianza 0.08 y para la imagen *Peppers* (Fig. 9) con varianza de 0.09, donde se ve en la imagen completa y el detalle ampliado que el método propuesto ICM-TM genera una mejor recuperación de la imagen con ruido.

Otros experimentos con la imagen de *cameraman* (Fig. 10) con ruido Gaussiano de varianza 0.08 y con la imagen de *boat* (Fig. 11) con varianza de 0.09 muestran que el método propuesto ICM-TM genera imágenes recuperadas visualmente con mayor calidad que otras técnicas tradicionales de filtrado y técnicas basadas en redes neuronales pulsantes.

El análisis cuantitativo de la gráfica de PSNR de las distintas técnicas comparadas cuando la varianza va de 0.01 y 0.09 (Fig. 12) permite mostrar que la supresión del ruido del método propuesto ICM-TM recupera la imagen en promedio por 9.6dB y es superior en promedio por 1.2dB a los métodos tradicionales de filtrado Sigma y Gaussiano, por 2.4dB al filtro de mediana y Wiener, y por 7.8dB a la técnica basada en PCNNNI, técnica que se distingue por su desempeño no lineal.

El método propuesto muestra un buen desempeño para varianza de ruido Gaussiano de 0.02 y hasta 0.09. Si la varianza es menor a 0.02 el método propuesto recupera la imagen por 7.6dB, aunque ya no logra superar a los otros filtros.

Se determinó el MAE de las distintas técnicas (Fig. 13), la imagen con menor error es la que se

obtiene por medio del método propuesto, la disminución del error es de 28 puntos.

El NMSE (Fig. 14) permite observar que el método propuesto ICM-TM tiende a minimizar esta métrica por debajo de los otros métodos. Conforme la varianza del ruido aumenta el error de la imagen contaminada y la capacidad de recuperación de los filtros disminuye.

Los experimentos con distintas imágenes a las que se agregó ruido Gaussiano de varianza 0.08, muestran la capacidad de los métodos para recuperar la imagen del ruido en valores de PSNR (Tabla 2).

Se repitió el experimento con ruido Gaussiano de varianza 0.09 (Tabla 3). Los resultados en PSNR permiten observar la recuperación para cada método donde destaca la técnica ICM-TM.

Se realizaron experimentos para varianzas de ruido Gaussiano de 0.003 a 0.04 que corresponden a una desviación estándar de entre 5 y 50 (Tabla 4). Se muestra que cuando la varianza es baja el filtro Wiener es el que obtiene un mejor resultado y cuando la varianza el método propuesto ICM-TM es el de mejor desempeño; no obstante la técnica BM3D muestra superioridad en PSNR para este rango de ruido filtrado.

7. Conclusiones y trabajo futuro

Los resultados obtenidos muestran que el método de filtrado ICM-TM propuesto supera en desempeño a los filtros tradicionales como filtro Gaussiano, Sigma y Wiener en promedio por 1.2 dB según PSNR a 0.02 y por 28 puntos en MAE y por 0.2 en NMSE para varianzas superiores. Permite relacionar vecindades de píxeles y no requiere cálculos estadísticos ni complejos, del mismo modo aprovecha las características de los filtros promedio y mediana al aplicarlos selectivamente; mientras que el método BM3D supera a los métodos comparados en PSNR, para el rango de varianza dado, y es de mayor complejidad.

Por otro lado ICM-TM supera a PCNNNI [18] en promedio por 7.8dB, y requiere 80% menos tiempo de procesamiento, además no se realiza un cambio del rango dinámico de la imagen, se

considera la relación entre neuronas vecinas y se tienen que ajustar menos parámetros.

En cuanto al tiempo de procesamiento el promedio para PCNNI fue de 160s, para el filtro de Mediana de 0.028s, para el filtro Wiener de 0.036s, para el método BM3D de 5.8s, para ICM-TM de 20s, por lo que la complejidad del método es acorde con el tiempo de procesamiento aunque éste puede ser mejorado mediante procesamiento en paralelo.

El trabajo futuro se centrará en la exploración de PCNN con variantes y la elección de los coeficientes de ajuste por medio de algoritmos evolutivos, así como en la búsqueda de un operador que permita un mejor ajuste de los valores de la imagen para preservar los detalles, el estudio profundo de la matriz de tiempos y tomando en cuenta las frecuencias para la localización de los píxeles que guardan mayor información de la imagen discriminando aquellos que presentan mayor grado de ruido.

También se pretende estudiar el comportamiento del filtro como sistema dinámico y su desempeño sobre algunas imágenes artificiales que se enfoca a características específicas como bordes y líneas. Finalmente una vez que se haya mejorado el algoritmo se llevarán a cabo comparaciones con otras técnicas de filtrado y por medio de criterios como la similitud estructural (*ssim*).

Agradecimientos

Los autores agradecen al CENIDET y al IPN por el apoyo económico para la realización de la presente investigación con ayuda de los siguientes fondos: SIP-IPN 20170693, y CONACYT en el marco del proyecto 65 en el marco de la convocatoria de Investigación Fronteras de la Ciencia 2015. Estela Ortiz agradece al CONACYT por la beca concedida para la realización de sus estudios de maestría.

Referencias

1. Bovik, A. (2009). *The Essential Guide to Image Processing*. Elsevier.
2. González, R. C. (2002). *Digital Image Processing*. Prentice Hall.
3. Pratt, W. (1972). Generalized Wiener Filtering Computation Techniques. *IEEE Transactions on Computers*, Vol. C-21, pp. 636–641.
4. Kaur, D. (2015). Remove Noise Effects From Degraded Document Images Using Matlab Algorithm. *International Journal of Engineering Sciences & Research Technology*, Vol. 4, No. 9, pp. 544–549.
5. Kang, D., & Lim, H. (2013). *Efficient noise reduction in images using directional modified sigma filter*. Springer Science Business Media New York, pp. 580–592.
6. Panetta, K., Bao, L. & Agaian, S. (2016). Sequence-to-Sequence Similarity Based Filter for Image Denoising. *IEEE Sensors Journal*.
7. Liu, J., Wang, Y., Su, K., & He, W. (2016). Image denoising with multidirectional shrinkage in directionlet domain. *Signal Processing*, Vol. 125, pp. 64–78.
8. Guleryuz, O. G. (2007). Weighted averaging for denoising with overcomplete dictionaries. *IEEE Trans. Image Process*, Vol. 16, pp. 3020–3034.
9. Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. (2007). Image denoising by sparse 3D transform-domain collaborative filtering. *IEEE Trans. Image Process*, Vol. 16, pp. 2080–2095.
10. Elad, M. & Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. on Image Process*, Vol. 15, pp. 3736–3745.
11. Palacios, A. E. & Ponomaryov, V. (2016). Image Denoising using Block Matching and Discrete Cosine Transform with Edge Restoring. *IEEE Conference Proceedings*, pp. 140–147.
12. Kumar, A. & Singh, B. (2015). Alexander Fractional Integral Filtering Of Wavelet Coefficients for Image Denoising. *Signal & Image Processing: An International Journal (SIPIJ)*, Vol. 6, No. 3, pp. 43–54.
13. Yang, S. & Min, W. (2013). Image Noise Reduction via Geometric Multiscale Ridgelet Support Vector Transform and Dictionary Learning. *IEEE Transactions on Image Processing*, Vol. 22, pp. 4161–4169.
14. Lindblad, T. & Kinser, J.M. (2005). *Image processing using Pulse-Coupled Neural Networks*. Springer.

15. Eckhorn, R., Reitboeck, H.J., & Arndt, M. (1990). Feature linking via synchronization among distributed assemblies: simulation of results from cat cortex. *Neural Computation*, pp. 293–307.
16. Ranganath, H.S., Kuntimad, G., & Johnson, J.L. (1995). Pulse coupled neural networks for image processing. *Proceedings of IEEE Southeast Conference*, Raleigh, pp. 26–29.
17. Ma, Y.D., Shi, F., & Li, L. (2003). Gaussian noise filter based on PCNN. *Proceedings of International Conference on Neural Networks and Signal Processing*, Nanjing.
18. Ma, Y.D., Lin, D.M., & Zhang, B.D. (2007). A novel algorithm of image Gaussian noise filtering based on PCNN time matrix. *Proceedings of IEEE International Conference on Signal Processing and Communication*, Dubai.
19. Lui, C. & Zhang, Z. (2008). Sonar images denoising based on pulse coupled neural networks. *Congress on Image and Signal Processing*, pp. 403–406.
20. Yuan-yuan, C. & Hai-yan, L. (2011). A new method of denoising mixed noise using Limited Grayscale Pulsed Couple Neural Network. *Cross Strait Quad-Regional Radio Science and Wireless Technology Conference*, pp. 1410–1413.
21. Ma, Y.D., Zhan, K., & Wang, Z. (2010). *Applications of pulse-coupled neural networks*. Springer.
22. Zhang, D. & Nishimura, T. (2009). Pulse Coupled Neural Network Based Anisotropic Diffusion Method for I/f Noise Reduction. *IEEE Transactions on Neural Networks*.
23. Johnson, J. L. & Padgett M. L. (1999). PCNN model and applications. *IEEE Transactions on Neural Networks*, Vol. 10, 480–498.
24. <http://sipi.usc.edu/database>.
25. Matlab toolbox para BM3D. Disponible en <http://www.cs.tut.fi/~foi/GCF-BM3D/>

Artículo recibido el 06/12/2016; aceptado el 26/04/2017.
 Autor de correspondencia es Estela Ortiz Rangel.