ALGORITMOS PARA REDUCCIÓN DE RUIDO GAUSSIANO Y SAL Y PIMIENTA EN IMÁGENES DIGITALES

TESIS

Que para obtener el grado de

DOCTOR EN CIENCIAS EN INGENIERÍA ELÉCTRICA

presenta

CARLOS ALBERTO JÚNEZ FERREIRA

DR. FERNANDO AUGUSTO VELASCO ÁVALOS

Director de Tesis

DR. FÉLIX CALDERÓN SOLORIO

Co-Director de Tesis

Universidad Michoacana de San Nicolás de Hidalgo

AGOSTO 2011

Al Todopoderoso, dador de vida, fuerza y esperanza

A Claudia y Santiago, por quienes yo seguiré despertando cada día con la voluntad de dar pasos adelante. En compañía de nuestro amor, siempre andaremos por el sendero de la felicidad.

A mis padres, que han entregado su vida a todos sus hijos, por todo su cariño, fe, apoyo y desvelos. Yo también los quiero y estoy orgulloso de ustedes. Son los mejores padres que pude haber tenido y un ejemplo para mí.

A mis hermanos, por sus palabras de aliento. Siempre los cuidaré.

Agradecimientos

Quiero manifestar mi más sincero agradecimiento a todas las personas que han contribuido para que mis estudios de doctorado y el desarrollo de esta tesis hayan sido posibles.

A mis asesores, el Dr. Fernando Velasco Avalos y el Dr. Félix Calderón Solorio, por compartirme una pequeña parte de sus conocimientos, sus grandes ideas y dirección que aportaron en este y otros trabajos, sus consejos, tiempo, paciencia y guía. Por su gran disposición para atenderme en cualquier momento, siempre con amabilidad. Son académicos ejemplares y mejores personas.

A mis sinodales y profesores Dr. Juan José Flores Romero, Dr. Leonardo Romero Muñoz y Dr. Francisco Javier Domínguez Mota. Por sus valiosas críticas, aportaciones y enseñanzas durante el proceso de revisión de este documento. También, agradezco sus atenciones y disposición.

A los directivos, personal administrativo, profesores y compañeros del programa de doctorado de la Facultad de Ingeniería Eléctrica, por las facilidades y atenciones prestadas.

Al Dr. Nelio Pastor Gómez, por sus aportaciones y consejos.

A directivos y compañeros de la Facultad de Ingeniería Civil y de la Universidad Michoacana de San Nicolás de Hidalgo, por su respaldo y facilidades prestadas.

A todos aquellos que omití y que formaron parte de mi formación en algún momento.

Lista de Publicaciones

"A simple algorithm for image denoising based on non-local means and preliminary segmentation"
Júnez-Ferreira, C.A., Velasco-Avalos, F.A.
Publicado en Proceedings of the 2009 Electronics, Robotics and Automotive Mechanics Conference, CERMA 09.

"Un algoritmo iterativo para la segmentación de imágenes con ruido"
Júnez-Ferreira, C.A., Velasco-Avalos, F.A., Pastor, N.
Publicado en VII Congreso Internacional en Innovación y Desarrollo Tecnológico, CIINDET 2009.

"A salt and pepper noise removal and restoration refinement algorithm"
Júnez-Ferreira, C.A., Velasco, F.A., Pastor, N.
Publicado en Proceedings of International Multi-conference on Complexity, Informatics and Cybernetics, IMCIC 2010.

"Salt and pepper noise detection based on non-local means"
Júnez-Ferreira, C.A., Velasco, F.A., Pastor, N.
Publicado en Proceedings of 7th International Conference on Informatics in Control, Automation and Robotics, ICINCO 2010.

"Regularization with adaptive neighborhood condition for image denoising"
Calderón, F., Júnez-Ferreira, C.A., Velasco, F.
Enviado a Journal of Signal Processing Systems, Springer.

Resumen

En este documento se presentan algoritmos para la reducción de ruido en imágenes digitales; de manera específica se aborda la reducción de ruido gaussiano y ruido del tipo sal y pimienta. Los algoritmos se basan en el empleo de información relacionada con las estructuras presentes en la imagen.

La primera contribución presentada en este trabajo consiste en un algoritmo que permite mejorar los tiempos de ejecución del algoritmo de medias no-locales, el cual elimina ruido gaussiano. Esta reducción en tiempo se realiza mediante el filtrado de la imagen submuestreada, la cual se obtiene con base en la segmentación de la imagen. El sobremuestreo se lleva a cabo de tal manera que permite la preservación de los bordes.

La segunda contribución es el diseño de un algoritmo orientado a la reducción de ruido sal y pimienta, el cual se realiza en dos fases. La primera fase consiste en el empleo de un filtro mediana, que adapta el tamaño de su ventana de acuerdo con la presencia de pixeles considerados como ruido. Posteriormente se aplica un filtro de mediana ponderada, donde se hace una propuesta de distribución de los coeficientes, lo que mejora sensiblemente los resultados.

La tercera contribución que se presenta en este trabajo consiste en una función basada en las características de los vecindarios, cuya minimización permite restaurar imágenes con ruido gaussiano y sal y pimienta, o una combinación de ambos. La función mencionada consiste de dos términos, uno de fidelidad que, a diferencia de los presentados en otros trabajos dentro de la literatura especializada, utiliza una estimación de la imagen restaurada. El otro término es el de regularización, el cual toma en cuenta la diferencia entre vecindarios con la finalidad de restaurar el valor de intensidad de un pixel.

Los resultados obtenidos de la aplicación de los algoritmos diseñados son robustos, en términos de la semejanza con la imagen original.

Abstract

This document presents algorithms for noise reduction in digital images, specifically, it addresses reduction of gaussian noise and salt and pepper noise. The proposed algorithms are based on the use of information related to the structures in the image.

The first contribution presented in this work is an algorithm that improves the execution time of non-local means algorithm, which removes gaussian noise. This time reduction is performed by filtering the subsampled image, which is obtained based on image segmentation. Oversampling is done in such a way that allows edge preservation.

The second contribution is the design of an algorithm aimed to suppress salt and pepper noise. This algorithm is divided in two phases: the first phase involves the use of a median filter which adapts its window size according to the presence of pixels considered as noise. The second phase consists of applying a weighted median filter, where a distribution of coefficients is proposed, which improves the results considerably.

The third contribution presented in this document is a function based on the characteristics of neighborhoods, whose minimization can restore images with gaussian noise, salt and pepper noise, or a mixture of both. The mentioned function is composed by two terms, a fidelity term that, unlike those reported in other works of the specialized literature, uses an estimate of the restored image. The other is a regularization term, which takes into account the difference between neighborhoods in order to restore a pixel intensity value.

The obtained results of applying the designed algorithms are robust in terms of similarity to the original image.

Contenido

Lista de Figuras

Lista de Tablas

Lista de Algoritmos

Lista de Símbolos

$\mathbf{I_0}$	Imagen libre de ruido.
Ω	Conjunto de todas las posiciones en el dominio de la imagen.
$I_0\big(x, y\big)$	Pixel de la imagen libre de ruido en la posición $\big(x, y\big)$.
η	Ruido.
$\mathbf{I}$	Imagen con ruido.
$I\big(x, y\big)$	Pixel de la imagen con ruido en la posición $\big(x, y\big)$.
$\mathbf{I_R}$	Imagen restaurada.
$I_R\big(x, y\big)$	Pixel de la imagen restaurada en la posición $\big(x, y\big)$.
$\big[I_{min}, I_{max}\big]$	Rango dinámico de valores en la imagen $\mathbf{I}$.
$N\big(x, y\big)$	Vecindario de tamaño $W \times W$, centrado en la posición $\big(x, y\big)$.
$\overline{N}(x, y)$	Valor de la media del vecindario $N(x, y)$.
$\widetilde{N}(x, y)$	Valor de la mediana del vecindario $N(x, y)$.
$\sigma^2(\phi)$	Varianza del conjunto de datos ϕ.
$\sigma(\phi)$	Desviación estándar del conjunto de datos ϕ.
δ	Densidad de ruido impulsivo.
∇	Operador gradiente.
Δ	Operador laplaciano.
$*$	Operador convolución.
$r\Diamond x$	Operador repetición, x repetido r veces.
$MAE(\mathbf{I_0}, \mathbf{I})$	Error Medio Absoluto.
$MSE(\mathbf{I_0}, \mathbf{I})$	Error Medio Cuadrático.
$SNR(\mathbf{I_0}, \eta)$	Relación Señal-Ruido.
$PSNR(\mathbf{I_0}, \mathbf{I})$	Relación Señal-Ruido Pico.
$UIQI(\mathbf{I_0}, \mathbf{I})$	Índice Universal de Calidad en Imágenes.

Capítulo 1

Introducción

El objetivo de la visión computacional es desarrollar sistemas que puedan comprender o interpretar la información presente en una imagen, en ocasiones tratando de simular el modo en que un humano lo haría, describiendo las estructuras y propiedades de una escena del mundo real a través del uso de imágenes. Por lo tanto, la visión computacional puede ser definida como el proceso de extraer información relevante del mundo físico a partir de imágenes digitales utilizando un sistema de cómputo para obtener esta información [Muñoz02].

El procesamiento de imágenes es una de las herramientas con que cuenta la visión computacional y consiste en producir nuevas imágenes a partir de las imágenes originales, resaltando las características más deseables para ciertos fines y, a su vez, minimizando las no deseables. La relevancia del procesamiento de imágenes en diversos campos ha estado creciendo de manera importante hasta nuestros días, teniendo numerosas aplicaciones en áreas como la industria, la militar, las ciencias biomédicas, las ingenierías, etc. Uno de los objetivos de sus aplicaciones consiste en auxiliar en el análisis de imágenes y el reconocimiento de objetos, tratando de obtener información y, a partir de ésta, formular conclusiones relacionadas con los fines perseguidos para, posteriormente, incidir en la toma de decisiones y constituir parte de la base de un plan de acción para la resolución de un problema determinado a partir de la información proporcionada por la imagen.

Con frecuencia, las imágenes digitales son alteradas por variaciones indeseables

en sus valores de intensidad, las cuales se conocen como ruido. La presencia de ruido se debe a diversos factores entre los que se encuentran los procesos de adquisición, compresión y transmisión de los datos, así como la presencia de fenómenos en la escena de interés [R. González01]. Entre los tipos más comunes de ruido se encuentran el gaussiano y el conocido como sal y pimienta. El primero, generado principalmente por problemas en los sensores o en la digitalización, produce pequeñas variaciones en los valores de intensidad [R. Jain95]. El segundo, generado principalmente durante la compresión y transmisión de los datos. En este tipo de ruido los valores de intensidad de algunos pixeles cambian aleatoriamente a los valores máximo o mínimo del rango de intensidades dentro de la imagen [R. Jain95].

En lo que resta de este capítulo se presentan tanto la motivación como los objetivos y el planteamiento de una propuesta que aborda la supresión de ruido gaussiano, sal y pimienta y la presencia de ambos en imágenes digitales.

1.1. Motivación

La remoción del ruido, en general, es una tarea fundamental dentro del área del procesamiento de imágenes digitales, ya que permite la preparación de la imagen para otros procesos tales como la segmentación y el reconocimiento de objetos.

Hasta el momento, se han presentado una gran cantidad de trabajos que abordan el problema de la detección y supresión de ruido gaussiano [A. Buades05b] y sal y pimienta [J. Astola97]. En lo que respecta a la supresión de ruido sal y pimienta, desafortunadamente no existen algoritmos que utilicen la información proporcionada por las estructuras presentes en la imagen para su remoción. En el presente trabajo se define una estructura como el arreglo de pixeles pertenecientes a un vecindario centrado en un pixel determinado, en el cual es relevante la posición de cada uno de los pixeles en el vecindario con respecto a la posición del pixel central. También, es necesario agregar que la mayoría de los métodos más populares para la supresión de ruido impulsivo no brindan buenos resultados en imágenes con altos niveles del mismo.

Con la finalidad de contribuir al desarrollo del área del preprocesamiento de imágenes, se deben proponer algoritmos para la remoción de ruido gaussiano y sal y pimien-

ta de una manera sencilla, rápida y con calidad en sus resultados, aún en imágenes con altos niveles de ruido, ya que, en la actualidad, los métodos que proporcionan los mejores resultados, se pueden considerar como costosos en su tiempo de ejecución y con resultados de baja calidad en el problema de borde, sobre todo en imágenes con altos niveles de ruido [R.H. Chan05].

1.2. Hipótesis

Las imágenes contienen estructuras que se repiten, es decir, áreas de apariencia similar y, con base en la información que proporcionan estas estructuras o características de vecindario, se puede mejorar la calidad de estas imágenes, incluso si se encuentran alteradas con ruido de tipo impulsivo.

De manera particular, es posible restaurar una imagen con ruido gaussiano y sal y pimienta haciendo uso de información estructural presente en ésta. En imágenes con altos niveles de ruido, es necesaria la obtención de una estimación preliminar, para entonces poder hacer uso de las características estructurales presentes en la imagen y de esta manera mejorar la calidad de la misma.

1.3. Planteamiento del Problema

El objetivo de cualquier algoritmo de supresión de ruido es restaurar el valor de intensidad original de un pixel que ha sido alterado o modificado por ruido. Sin embargo, esta tarea no es simple ya que si no se cuenta con la imagen no alterada (si la tuviéramos no habría problema que resolver), es imposible saber con certeza el valor original de un pixel alterado, lo que dificulta la resolución del problema. Lo único que se puede hacer es obtener una estimación, la mejor que se pueda, de la imagen restaurada con base en información extraída a partir de la imagen con ruido. Es evidente que, mientras se tengan más pixeles alterados, más complicada será esta restauración.

Con la finalidad de tener una idea de la magnitud del problema, ésta se puede dimensionar partiendo de una imagen con *256* tonos de gris (pertenecientes al intervalo

$[0, 255]$) que consta de $n_1 \times n_2$ pixeles, la cual ha sido alterada aleatoriamente por ruido sal y pimienta. Este tipo de ruido modifica aleatoriamente los valores de intensidad de algunos pixeles al valor máximo o valor mínimo del rango de intensidades dentro de la imagen. El nivel de daño en la imagen se puede medir por medio de un valor de densidad δ con $0 \leq \delta \leq 1$, esto es, que aproximadamente el $(100\delta)\,\%$ de los pixeles ha sido sustituido por el valor mínimo o por el valor máximo en la imagen. Ahora, con la finalidad de dar una idea de la magnitud del problema de detección de ruido impulsivo, se plantea determinar la cantidad de imágenes diferentes con ruido impulsivo de densidad δ que se pueden tener. Considerando solamente los pixeles alterados por ruido que puede haber (sin tomar en cuenta si su intensidad es la mínima o la máxima), la cantidad mencionada estará dada por la expresión

$$\frac{(n_1 n_2)!}{(\delta n_1 n_2)!((1-\delta)n_1 n_2)!}$$

por ejemplo, con una imagen pequeña en tonos de gris de 4×4 pixeles y considerando una densidad de ruido $\delta = 0.2$ (que se puede considerar como baja), se pueden tener aproximadamente $\frac{(16)!}{(3)!(13)!} = 560$ diferentes imágenes modificadas por ruido. La cantidad anterior representa solamente la cantidad de imágenes modificadas para una sola densidad de ruido, ahora, hay que imaginar la cantidad de imágenes alteradas que se pueden tener para todas las densidades de ruido en su conjunto, lo que brinda una idea de la dimensión del problema.

En lo que respecta a la restauración de la imagen, una vez detectados los pixeles que pueden ser considerados como alterados, el valor buscado para cada uno de ellos puede tomar 256 valores diferentes lo que conduce a tener aproximadamente $256^{\delta n_1 n_2}$ posibles imágenes que podrían resolver el problema, por ejemplo, para una imagen de 256×256 pixeles y una densidad de ruido $\delta = 0.2$, se tendrían 256^{13107} opciones o soluciones de restauración, de las cuales no se puede saber con certeza cual es la correcta. Lo único que se puede hacer es obtener una estimación del valor de intensidad de cada pixel que toma en cuenta los valores de intensidad de pixeles vecinos mediante operaciones de filtrado.

Entre los métodos más empleados para suprimir ruido, se encuentran las operaciones de filtrado pasa-bajas, sin embargo, en lo que respecta a la supresión de ruido sal y pimienta, este tipo de métodos no brindan buenos resultados, debido a la dificultad que representa el suavizado de impulsos. Además, la remoción del ruido por medio de estos

algoritmos es proporcional a la alteración de detalles o bordes [P. Perona90]. Normalmente, para realizar esta tarea se emplean filtros no lineales [J. Astola97], e.g. el filtro mediana, sin embargo, en imágenes con altas densidades de ruido, este tipo de filtros no producen resultados satisfactorios.

1.4. Objetivos de la Tesis

1.4.1. Objetivo general

El objetivo de esta tesis es desarrollar algoritmos para suprimir ruido gaussiano y sal y pimienta en imágenes digitales, cuando se presentan por separado o combinados. Los algoritmos propuestos deberán ser capaces de restaurar una imagen dañada por estos tipos de ruido de manera robusta, es decir, que la imagen resultado de la aplicación de los algoritmos diseñados, sea lo más cercano a la imagen original, ante diferentes condiciones o niveles de ruido, en un tiempo aceptable.

1.4.2. Objetivos particulares

- Mejorar la eliminación de ruido gaussiano reduciendo los tiempos de ejecución de un algoritmo existente.

- Diseñar un algoritmo que suprima ruido impulsivo mediante un filtrado que se adapte a las características de ruido en la imagen.

- Diseñar una función de costos cuya optimización restaure una imagen ruidosa, alterada con ruido gaussiano, sal y pimienta, o una combinación de ambos, en forma robusta.

- Comparar el desempeño de los algoritmos propuestos con algunos existentes en la literatura.

1.5. Descripción de Capítulos

El contenido de esta tesis se ha organizado en seis capítulos. En el presente capítulo se ha hecho una introducción, se ha planteado el problema, y se especificaron la motivación, la hipótesis y los objetivos de este trabajo. En el Capítulo 2 se tratan aspectos básicos de las imágenes digitales, así como del ruido que se puede presentar en ellas. Se hace, además, una revisión del estado del arte de los métodos más comunes de supresión de ruido, tanto gaussiano como sal y pimienta. El Capítulo 3 presenta una propuesta para reducir los tiempos de ejecución de un algoritmo orientado a la remoción de ruido gaussiano en imágenes digitales. En el Capítulo 4 se presenta un algoritmo diseñado para la supresión de ruido sal y pimienta. En el Capítulo 5 se ha diseñado una función de costos que puede ser empleada en la reducción de ruido gaussiano, sal y pimienta o una combinación de ambos. Finalmente, en el Capítulo 6 se presentan conclusiones derivadas de esta investigación y se plantean los trabajos para futuras investigaciones.

Capítulo 2

Revisión del estado del arte

2.1. Imágenes digitales

Una imagen es la representación que reproduce la apariencia de un objeto o escena real. En el presente trabajo se define a una imagen como la representación visual en un cierto dominio de una señal física (escena) adquirida mediante un dispositivo. Por ejemplo, en el dominio bidimensional, una imagen puede ser modelada mediante una función de dos variables constituida por una distribución espacial de valores de intensidad en un plano [R. Jain95].

Para fines de procesamiento y almacenamiento en una computadora, es necesario transformar la información continua en discreta, asumiendo que la imagen se puede representar mediante una matriz de coordenadas espaciales enteras mediante un proceso de digitalización. Digitalizar una imagen involucra dos pasos: muestreo y cuantización. Muestrear una imagen es el proceso de representar la imagen original continua y bidimensional con una rejilla que consta de pequeñas localidades. Un punto en la rejilla es llamado pixel. Cuantización es el proceso de asignar un valor entero a cada localidad (para una imagen en tonos de gris estos valores enteros se encuentran dentro del intervalo $[0, 255]$).

Entonces, una imagen $\mathbf{I}$ de tamaño $n_1 \times n_2$, se representa como un arreglo bidimensional de puntos, donde n_1 es el número de renglones del arreglo y n_2 es el número de columnas. Una imagen que ha sido muestreada y cuantizada es llamada imagen digital.

Un pixel $I(x, y)$ tiene asociado un valor de intensidad con la posición correspondiente en el arreglo, en otras palabras, un pixel es una muestra de la intensidad de la imagen cuantizada a un valor entero. Los índices (x, y) de un pixel son valores enteros que especifican el renglón y la columna en el arreglo de pixeles, respectivamente. El par ordenado (x, y) es un elemento del conjunto de todas las posiciones de los pixeles en el dominio de la imagen $\Omega = \{1, \ldots, n_1\} \times \{1, \ldots, n_2\}$. El pixel en la posición $(1, 1)$ se localiza en la esquina superior izquierda de la imagen y a partir de ahí hacia abajo se encuentra el índice x y hacia la derecha el índice y.

En lo que respecta a las imágenes a color, existen diversos espacios o modelos de color, siendo el más empleado el modelo RGB (del inglés Red, Green, Blue) [Russ99]. Con este modelo es posible representar un color mediante la combinación por adición de los colores rojo, verde y azul. Por lo tanto una imagen digital a color estará representada por tres matrices bidimensionales con el mismo tamaño cada una. En otras palabras, un pixel en la posición (x, y) tendrá asociado un vector de tres elementos, donde a cada elemento le corresponderá un valor entero de intensidad dentro del intervalo de valores enteros $[0, 255]$ de rojo, verde y azul respectivamente. Los algoritmos propuestos en el presente trabajo son aplicados en imágenes en tonos de gris, en caso de que se tratara de una imagen a color con modelo RGB se emplea el mismo algoritmo para cada uno de los tres colores por separado.

2.2. Ruido en imágenes

Como se mencionó anteriormente, las imágenes digitales son alteradas frecuentemente por variaciones aleatorias indeseables en los valores de intensidad, llamadas ruido. La presencia de ruido se debe a diversos factores tales como el proceso de adquisición, compresión y transmisión de los datos, así como la ocurrencia de fenómenos en la escena de interés [R. González01].

De manera general, un pixel de una imagen con ruido aditivo e independiente de la misma puede ser representado como

$$I(x, y) = I_0(x, y) + \eta \tag{2.1}$$

donde $I(x, y)$ y $I_0(x, y)$ son los pixeles en la posición (x, y) de la imagen observada $\mathbf{I}$ y de la imagen original sin alterar $\mathbf{I_0}$, respectivamente, y η es el ruido agregado cuyo valor es aleatorio. Existen diversos tipos de ruido, y entre los más comunes se encuentran el ruido gaussiano y el impulsivo [J. Astola97].

El ruido gaussiano consiste en variaciones de la intensidad descritas por medio de una distribución gaussiana, con cierta desviación estándar σ_η o varianza y comúnmente con media igual a cero [R. Jain95]. Es decir, en la expresión 2.1, cada muestra de ruido proviene de una distribución normal $\eta \sim N_G(0, \sigma_\eta)$. El ruido gaussiano es un buen modelo para muchos tipos de ruido aleatorios debido a la superposición de muchos procesos individuales, lo cual se puede relacionar con el teorema del límite central [Jahne97], con lo que se puede considerar que la suma de diferentes tipos de ruido tiende a aproximarse a una distribución gaussiana. Usualmente, con el ruido gaussiano, la mayoría de los pixeles en la imagen sufren pequeños cambios en su valor de intensidad. Las Figuras 2.1(b) y 2.1(c) muestran a la imagen de la Figura 2.1(a) contaminada con ruido gaussiano de desviaciones estándar de $\sigma_\eta = 20$ y $\sigma_\eta = 60$, respectivamente.

Como se ha mencionado, otro tipo común de ruido es el de tipo impulsivo, de manera particular el conocido como sal y pimienta, el cual es llamado así debido a su apariencia de puntos blancos y negros en la imagen. Este tipo de ruido se produce principalmente debido a fallas en sensores de los dispositivos de adquisición, ya sea por su calidad, o por la presencia de condiciones ambientales específicas tales como el nivel de luz o la temperatura, también se puede producir debido a fallas en localidades de memoria y durante la transmisión de la imagen debido a interferencias en los canales usados para la transmisión, finalmente, otra de las causas que lo pueden generar, aunque no tan común, es la presencia de partículas de polvo al interior del dispositivo de captura [A. Shrivastava07]. El ruido sal y pimienta puede ser encontrado en diferentes tipos de imágenes tales como las satelitales y las médicas, entre otras. En este tipo de ruido los valores de intensidad de algunos pixeles cambian aleatoriamente a los valores máximo o mínimo del rango de intensidades dentro de la imagen [R.H. Chan05]. De lo anterior se puede considerar que, por lo general, un pixel alterado en la imagen, difiere drásticamente con respecto a la intensidad de los pixeles vecinos, es decir, no conservan ningún tipo de relación y generalmente se cuantifica por el

porcentaje o densidad (δ) de pixeles alterados.

El ruido sal y pimienta puede ser modelado o reproducido como se describe a continuación [R.H. Chan05]:

Sea $I_0(x, y)$, el valor de intensidad de un pixel en la posición (x, y) de una imagen $\mathbf{I_0}$ de tamaño $n_1 \times n_2$, y sea $[I_{min}, I_{max}]$, i.e. $I_{min} \leq I_0(x, y) \leq I_{max}$, para toda $(x, y) \in \Omega$, el rango dinámico de $\mathbf{I_0}$. Se define $\mathbf{I}$ como la versión alterada con ruido sal y pimienta de $\mathbf{I_0}$, entonces, el valor de intensidad observado en la posición (x, y) de la imagen $\mathbf{I}$ está dado por

$$I(x,y) = \begin{cases} I_{min} & con\, probabilidad\, \delta_1 \\ I_{max} & con\, probabilidad\, \delta - \delta_1 \\ I_0(x,y) & con\, probabilidad\, 1 - \delta \end{cases} \tag{2.2}$$

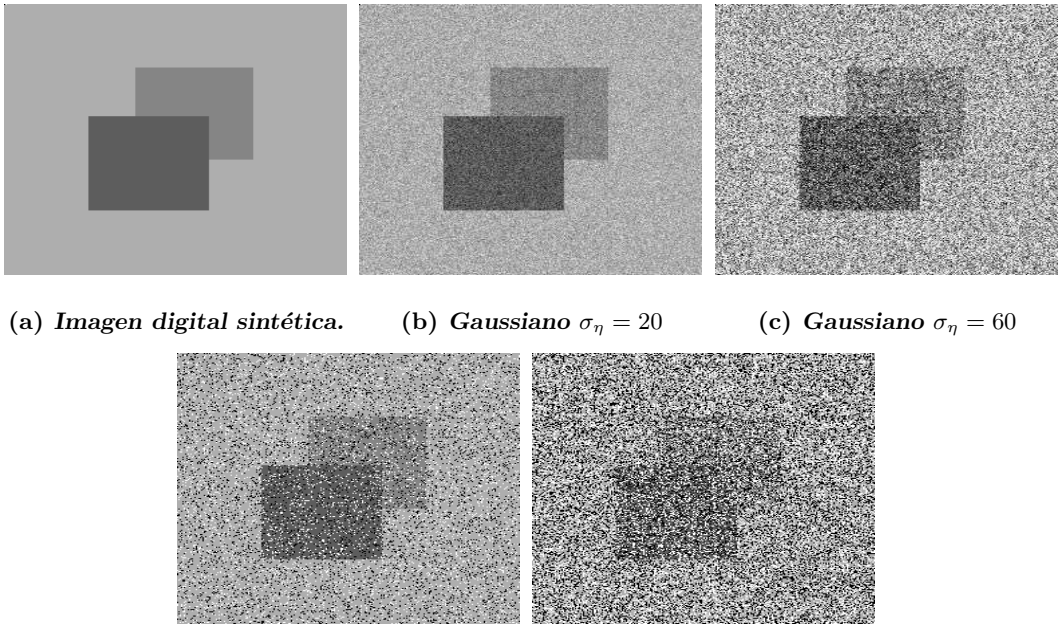
donde δ define la densidad de ruido total en toda la imagen, δ_1 es la densidad de pixeles ruidosos con valor de intensidad mínimo y $\delta - \delta_1$ es la densidad de pixeles ruidosos con valor de intensidad máximo.

Las Figuras 2.1(d) y 2.1(e) muestran a la imagen de la Figura 2.1(a) contaminada con ruido sal y pimienta de densidades $\delta = 20$ y $\delta = 50$, respectivamente.

2.3. Medición del ruido

Con la finalidad de cuantificar el daño que produce el ruido en una imagen se han definido diferentes maneras de medirlo, entre las que se encuentran el error absoluto medio (MAE) [Chacón07], el error cuadrático medio (MSE) [Chacón07], la relación señal a ruido (SNR) [Horn01], la relación señal a ruido de pico ($PSNR$) [Chacón07] y el índice universal de calidad en imágenes ($UIQI$) [Z. Wang02], los cuales se definen en esta sección.

Existen otros tipos de métodos de medición basados en el sistema de visión humana, los cuales quedan fueran del alcance del presente trabajo, además de que no presentan hasta el momento ninguna ventaja significativa con respecto a las medidas definidas matemáticamente, debido a su naturaleza subjetiva [T.N. Pappas00].



(a) ***Imagen digital sintética.*** **(b)** ***Gaussiano*** $\sigma_\eta = 20$ **(c)** ***Gaussiano*** $\sigma_\eta = 60$

(d) ***Sal y pimienta*** $\delta = 20$ **(e)** ***Sal y pimienta*** $\delta = 50$

Figura 2.1: Imagen sintética contaminada con diferentes tipos de ruido.

2.3.1. Error medio absoluto

Este parámetro es ampliamente utilizado y mide la media de los valores absolutos de las diferencias entre los pixeles de dos imágenes [Chacón07] y se expresa como

$$MAE(\mathbf{I_0}, \mathbf{I}) = \frac{1}{n_1 n_2} \sum_{x=1}^{n_1} \sum_{y=1}^{n_2} |I_0(x,y) - I(x,y)| \tag{2.3}$$

2.3.2. Error medio cuadrático

Este valor mide la media de los cuadrados de las diferencias entre los pixeles de las imágenes $\mathbf{I_0}$ e $\mathbf{I}$ [Chacón07], y se expresa como

$$MSE(\mathbf{I_0}, \mathbf{I}) = \frac{1}{n_1 n_2} \sum_{x=1}^{n_1} \sum_{y=1}^{n_2} (I_0(x,y) - I(x,y))^2 \tag{2.4}$$

2.3.3. Relación señal-ruido

La relación señal a ruido se puede expresar como [Horn01]

$$SNR(\mathbf{I_0}, \eta) = \frac{\sigma_{\mathbf{I_0}}}{\sigma_\eta} \tag{2.5}$$

donde $\sigma_{\mathbf{I_0}}$ denota la deviación estándar de la imagen $\mathbf{I_0}$ de manera empírica,

$$\sigma_{\mathbf{I_0}} = \sqrt{\frac{1}{n_1 n_2} \sum_{x=1}^{n_1} \sum_{y=1}^{n_2} (I_0(x,y) - \overline{\mathbf{I_0}})^2} \tag{2.6}$$

y a su vez

$$\overline{\mathbf{I_0}} = \frac{1}{n_1 n_2} \sum_{x=1}^{n_1} \sum_{y=1}^{n_2} I_0(x,y) \tag{2.7}$$

La desviación estándar del ruido σ_η puede ser obtenida a partir de los parámetros del modelo del ruido conocidos.

2.3.4. Relación señal-ruido de pico

La relación señal a ruido de pico define la relación entre la máxima energía posible de una señal y el ruido que afecta a su representación fidedigna [Chacón07]. Se expresa generalmente en escala logarítimica, su unidad es el decibel (dB) y se puede expresar como

$$PSNR(\mathbf{I_0}, \mathbf{I}) = 10 log_{10} \left(\frac{(max(\mathbf{I_0}))^2}{MSE(\mathbf{I_0}, \mathbf{I})} \right) \tag{2.8}$$

donde $max(\mathbf{I_0})$ corresponde al máximo valor de intensidad presente en la imagen y $MSE(\mathbf{I_0}, \mathbf{I})$ es el error cuadrático medio que se calcula con la ecuación (2.4).

2.3.5. Índice universal de calidad en imágenes

Este índice [Z. Wang02] modela la distorsión de una imagen como una combinación de tres factores: pérdida de correlación, distorsión de la luminancia y distorsión del contraste. El índice se define como

$$UIQI(\mathbf{I_0}, \mathbf{I}) = \frac{4\sigma_{\mathbf{I_0},\mathbf{I}} \overline{\mathbf{I_0}}\, \overline{\mathbf{I}}}{(\sigma^2_{\mathbf{I_0}} + \sigma^2_{\mathbf{I}})(\overline{\mathbf{I_0}}^2 + \overline{\mathbf{I}}^2)} \tag{2.9}$$

donde

$$\sigma_{\mathbf{I_0},\mathbf{I}} = \frac{1}{n_1 n_2} \sum_{x=1}^{n_1} \sum_{y=1}^{n_2} (I_0(x,y) - \overline{\mathbf{I_0}})(I(x,y) - \overline{\mathbf{I}}) \tag{2.10}$$

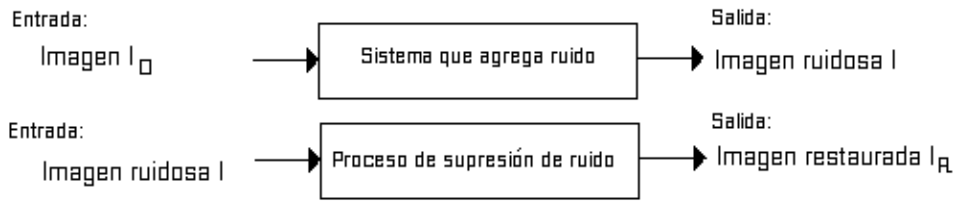


Figura 2.2: Sistemas de adición y supresión de ruido.

y $\overline{\mathbf{I_0}}$ e $\bar{\mathbf{I}}$ son la media y $\sigma^2_{\mathbf{I_0}}$ y $\sigma^2_{\mathbf{I}}$ son la varianza de las imágenes $\mathbf{I_0}$ e $\mathbf{I}$ respectivamente. El rango dinámico de este índice es $[-1, 1]$, alcanzándose el máximo valor de 1 si y sólo si $\mathbf{I_0} = \mathbf{I}$. La expresión (2.9) puede ser escrita como

$$UIQI(\mathbf{I_0}, \mathbf{I}) = \left(\frac{\sigma_{\mathbf{I_0},\mathbf{I}}}{\sigma_{\mathbf{I_0}}\sigma_{\mathbf{I}}}\right)\left(\frac{2\overline{\mathbf{I_0}}\,\bar{\mathbf{I}}}{\overline{\mathbf{I_0}}^2 + \bar{\mathbf{I}}^2}\right)\left(\frac{2\sigma_{\mathbf{I_0}}\sigma_{\mathbf{I}}}{\sigma^2_{\mathbf{I_0}} + \sigma^2_{\mathbf{I}}}\right) \tag{2.11}$$

el primer componente es el coeficiente de correlación entre $\mathbf{I_0}$ e $\mathbf{I}$, el segundo componente mide qué tan parecida es la intensidad media entre $\mathbf{I_0}$ y $\mathbf{I}$, finalmente, el tercer componente mide qué tan similar es el contraste entre las dos imágenes, es decir compara la dispersión entre las dos imágenes.

2.4. Métodos de reducción de ruido

En el caso particular de las imágenes digitales, el objetivo de los métodos de eliminación de ruido es recuperar la imagen de la escena original (o la mejor aproximación posible a ésta) a partir de su versión con ruido, tratando de conservar o restaurar lo mejor posible los detalles existentes en la escena. En la Figura 2.2 se esquematizan los procesos de adición y supresión de ruido. Después del proceso de supresión, la imagen restaurada $\mathbf{I_R}$ tiene que ser lo más parecida a la imagen original $\mathbf{I_0}$.

Hasta la fecha, de la mano con la evolución de los sistemas de cómputo, una gran cantidad de métodos han sido propuestos con la finalidad de suprimir ruido en imágenes digitales y, por consiguiente, tratar de restaurar de la mejor manera posible la imagen original. Cabe mencionar que, hasta el momento, no existe un método capaz de resolver el problema de la supresión de ruido de manera general para todo tipo de ruido en tiempos

cortos debido principalmente a la estructura propia de los diferentes tipos de imágenes y del ruido presente en ellas para los cuáles estas técnicas están orientadas y a las bases teóricas y modelos a partir de los cuales fueron desarrolladas. Por los motivos anteriores, así como la combinación de técnicas diversas, tampoco se puede hablar de una clasificación generalizada de las metodologías seguidas para suprimir ruido. Sin embargo, en [A. Buades05b] se hace un análisis muy completo de los métodos más generales para la supresión de ruido y en [J. Astola97] y [E. Dougherty99] se hacen amplias revisiones de los métodos más comunes que emplean el filtrado no lineal, principalmente aplicado en la eliminación de ruido impulsivo.

En esta sección se revisarán algunos aspectos teóricos de los métodos más comunes para reducir ruido gaussiano y sal y pimienta. Los resultados experimentales y su análisis, después de implementar estos métodos en los tipos de ruido citados, se muestran en una sección posterior.

2.4.1. Filtrado gaussiano

Una de las formas más comunes para remover el ruido es convolucionando la imagen con un kernel que represente un filtro, efectuando una operación de suavizado a través de un promedio pesado de los valores de intensidad de los pixeles vecinos, en el cual, usualmente, los pesos varían con la distancia existente entre estos y el pixel central de la ventana del filtro.

Los filtros gaussianos son una clase de filtros lineales de suavizado, cuyos pesos se definen de acuerdo con una función gaussiana [R. Jain95]. El kernel gaussiano en dos dimensiones, con media igual a cero y desviación estándar σ_F se puede expresar de la forma

$$G(x, y, \sigma_F) = \frac{1}{2\pi\sigma_F^2} e^{\frac{-(x^2+y^2)}{2\sigma_F^2}} \tag{2.12}$$

Entonces, la obtención de la imagen suavizada se puede expresar como

$$\mathbf{I_1} = \mathbf{I} * \mathbf{G} \tag{2.13}$$

donde el símbolo * representa la operación de convolución y que, de manera discreta, mediante un operador de ventana de tamaño $W \times W$ con $W = 2m_1 + 1$, se reemplaza el valor

del pixel en la posición (x, y) por medio de la operación de convolución [Jahne97]:

$$I_1(x,y) = \sum_{p=1}^{W} \sum_{q=1}^{W} G(p,q,\sigma_F) I(x-p, y-q) \tag{2.14}$$

Las funciones gaussianas tienen propiedades que pueden ser de utilidad en el procesamiento de imágenes [R. Jain95]. Cuando se trabaja en dos dimensiones, los filtros gaussianos pueden separarse, aplicando un kernel gaussiano en una dimensión, primero en una dirección y después en la otra. Esto representa un ahorro en tiempo de cómputo comparado con la aplicación de un kernel para dos dimensiones. Entre otras propiedades, también se encuentra que este tipo de filtros son isotrópicos, esto significa que el suavizado será el mismo en cualquier dirección (lo cual puede ser un problema en los bordes); otra característica es su unimodalidad, lo que significa que los pesos en el vecindario decrecen monotónicamente con la distancia a partir del pixel central. Otras propiedades y detalles sobre su implementación se pueden consultar en [R. Jain95] y en [Jahne97].

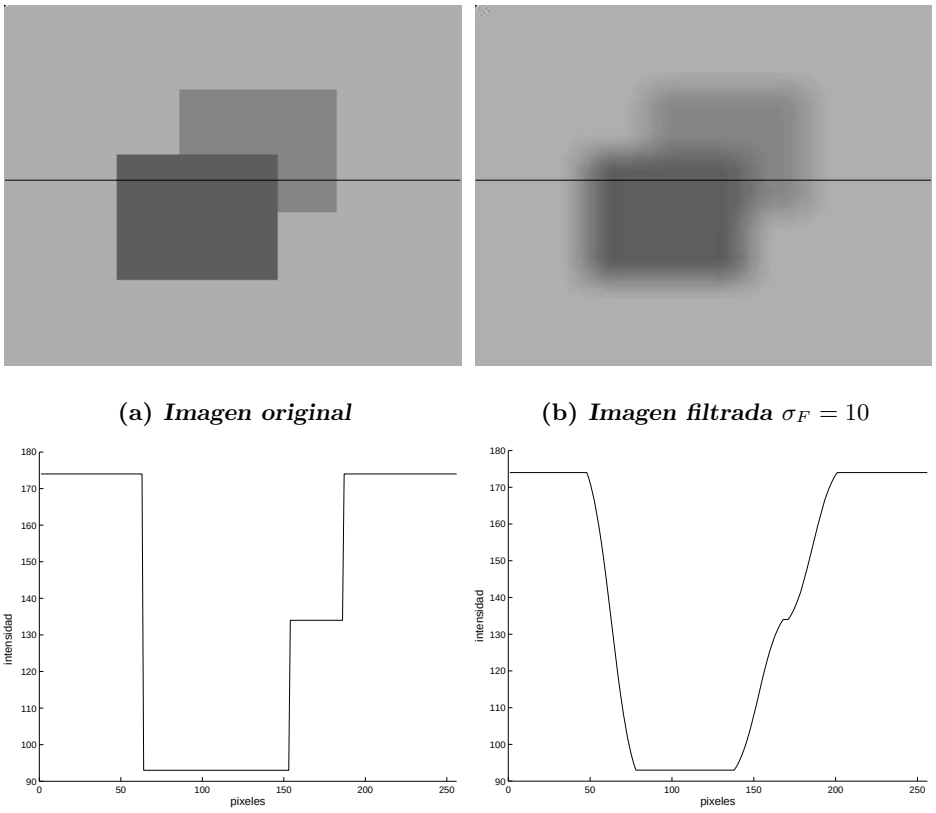
En la Figura 2.3(b) se muestra el resultado de aplicar un filtro gaussiano en la Figura 2.3(a), el filtro tiene una desviación estándar de $\sigma_F = 10$. Las Figuras 2.3(d) y 2.3(c) muestran los perfiles respectivos, donde se puede observar el efecto de borroneo o desvanecimiento que se produce en los bordes al aplicar este tipo de filtros.

El filtro gaussiano es considerado como un buen reductor de ruido, sin embargo, como se puede observar en la Figura 2.3 el empleo de filtros pasa-bajas provoca borroneo de los bordes y detalles de la imagen, lo que conduce al problema de la selección de la escala (parametrizada por la varianza o desviación estándar). A mayor escala, mayor atenuación del ruido, pero también mayor borroneo de los bordes y detalles.

2.4.2. Difusión anisotrópica

Un trabajo orientado a solucionar el problema de borroneo o desvanecimiento de los bordes fue presentado por Perona y Malik [P. Perona90], cuya propuesta consiste en procesar la imagen con una ecuación diferencial parcial de suavizado similar a la ecuación de calor, conocida como difusión anisotrópica.

La idea principal consiste en que dado un coeficiente de difusión diseñado para detectar bordes, el ruido puede ser removido sin el borroneo de estos. Para esto, se parte



(a) ***Imagen original*** **(b)** ***Imagen filtrada*** $\sigma_F = 10$

(c) ***Perfil de imagen original*** **(d)** ***Perfil de imagen filtrada***

Figura 2.3: Resultados de filtrado gaussiano.

de la definición de la ecuación de difusión anisotrópica como

$$I_1(x, y, t) = \nabla(c(x, y, t)\nabla I) = c(x, y, t)\Delta I + \nabla c \cdot \nabla I \tag{2.15}$$

donde $c(x, y, t)$ es el coeficiente de difusión, t es la escala, así como ∇ el gradiente, y Δ, el Laplaciano, con respecto a las variables espaciales x y y.

El gradiente en el pixel $I(x, y)$ es la función vectorial [Chacón07]

$$\nabla I(x, y) = \left[\frac{\partial I(x, y)}{\partial x} \quad \frac{\partial I(x, y)}{\partial y}\right]^T \tag{2.16}$$

donde $\frac{\partial I(x,y)}{\partial x}$ y $\frac{\partial I(x,y)}{\partial y}$, pueden ser obtenidos mediante $\frac{\partial I(x,y)}{\partial x} \equiv I(x, y) - I(x + 1, y)$ y $\frac{\partial I(x,y)}{\partial y} \equiv I(x, y) - I(x, y + 1)$, respectivamente.

La magnitud del gradiente está dada por

$$\| \nabla I(x,y) \| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2} \tag{2.17}$$

El Laplaciano se puede considerar como el equivalente bidimensional de la segunda derivada [R. Jain95] [R. González01]. La fórmula para el Laplaciano en el pixel $I(x,y)$ es

$$\Delta I(x,y) = \nabla^2 I(x,y) = \frac{\partial^2 I(x,y)}{\partial x^2} + \frac{\partial^2 I(x,y)}{\partial y^2} \tag{2.18}$$

El Laplaciano puede ser utilizado para la detección de bordes. Los filtros basados en el Laplaciano tienen una respuesta cero o muy pequeña en regiones uniformes o con poca variación y mayor en regiones no uniformes. Lo anterior significa que este tipo de filtros modifican el valor promedio de las regiones uniformes a un valor próximo a cero y cerca de las regiones no uniformes habrá valores positivos y negativos. Así, la segunda derivada proporciona los máximos locales de los valores de gradiente, lo que permite detectar bordes. Esto quiere decir que en los puntos de bordes hay un pico en la primera derivada y por lo tanto hay un cruce por cero de la segunda derivada, condiciones que se buscan en la solución del problema de detección de bordes.

La segunda derivada con respecto a x se aproxima usando ecuaciones de diferencias

$$\frac{\partial^2 I(x,y)}{\partial x^2} \equiv I(x-1,y) - 2I(x,y) + I(x+1,y) \tag{2.19}$$

De manera similar, la aproximación a la segunda derivada con respecto a y puede ser obtenida mediante

$$\frac{\partial^2 I(x,y)}{\partial y^2} \equiv I(x,y-1) - 2I(x,y) + I(x,y+1) \tag{2.20}$$

Aquí es necesario mencionar que el empleo de las expresiones (2.16)-(2.20) en imágenes con ruido no es lo mejor, ya que son muy sensibles a éste. En este caso es recomendable utilizar aproximaciones más robustas como las derivadas de gaussianas, así, por ejemplo, la primera derivada estará dada por [Canny86]

$$\nabla(I(x,y) * G(x,y)) = \left[I(x,y) * \frac{\partial G(x,y)}{\partial x} \quad I(x,y) * \frac{\partial G(x,y)}{\partial y} \right]^T \tag{2.21}$$

y como la función gaussiana en dos dimensiones es separable

$$\nabla(I(x,y) * G(x,y)) = \left[I(x,y) * \frac{dG(x)}{dx} * G(y) \quad I(x,y) * G(x) * \frac{dG(y)}{dy} \right]^T \tag{2.22}$$

Ahora, se supone que a la escala t se conocen las localidades de los bordes para tal escala, mediante una estimación **E**, y lo que se desea es suavizar dentro de la región y no en los bordes. Esto sería posible haciendo igual a 1 el coeficiente c al interior de cada región y 0 en los bordes. Entonces, la estimación **E** tendría las siguientes propiedades:

- $E(x,y,t) = 0$ al interior de cada región.
- $E(x,y,t) = kE_u(x,y,t)$ en cada punto de borde, donde E_u es un vector unitario normal al borde en el punto (x,y), y k es una constante que representa el contraste local en el borde.

Tomando lo anterior, el coeficiente de difusión $c(x,y,t)$ se puede definir como función de la magnitud de la estimación

$$c(x,y,t) = g(\parallel E(x,y,t) \parallel) \tag{2.23}$$

donde la función g tiene que ser monotónicamente decreciente y no negativa con $g(0) = 1$.

Perona y Malik propusieron dos funciones para el coeficiente de difusión, considerando al gradiente de la imagen como una buena estimación para **E**:

$$g(\nabla I) = e^{-\left(\frac{\|\nabla I\|}{k}\right)^2} \tag{2.24}$$

y

$$g(\nabla I) = \frac{1}{1 + \left(\frac{\|\nabla I\|}{k}\right)^2} \tag{2.25}$$

La función (2.24) privilegia los bordes de alto contraste y la función (2.25) privilegia las regiones homogéneas extensas.

2.4.3. Filtrado Bilateral

Tomasi y Manduchi [C. Tomasi98], presentaron el filtrado bilateral, el cual se ha convertido en un método popular que proporciona muy buenos resultados para la supresión

de ruido. Esta propuesta realiza la combinación de valores de pixeles basada en la cercanía geométrica o distancia y en la similitud de intensidades, prefiriendo valores cercanos a aquellos distantes y promediando valores de intensidad con pesos que se encuentran en función de su similitud.

Se define un filtro en el dominio espacial, aplicado a una imagen **I**, como

$$I_1(x,y) = \frac{1}{Z_d} \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} C(x,y,x+p,y+q) I(x+p,y+q) \tag{2.26}$$

donde (x, y) es el centro del vecindario de tamaño $W \times W$ con $W = 2m_1 + 1$, el par de va-

lores $(x + p, y + q)$ son las coordenadas de cualquier pixel en el vecindario y $C(x_0, y_0, x_1, y_1)$ mide la cercanía o distancia geométrica entre un pixel en la posición (x_0, y_0) y un pixel en la posición (x_1, y_1). La constante de normalización Z_d se define como

$$Z_d = \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} C(x,y,x+p,y+q) \tag{2.27}$$

Por otro lado, se define un filtro en el rango de intensidades como

$$I_1(x,y) = \frac{1}{Z_r} \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} S(I(x,y), I(x+p,y+q)) I(x+p,y+q) \tag{2.28}$$

donde $S(a, b)$ mide la similitud de intensidades entre un pixel con intensidad a y un pixel con intensidad b. Entonces, la constante de normalización está dada por

$$Z_r = \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} S(I(x,y), I(x+p,y+q)) \tag{2.29}$$

Resumiendo lo anterior, la función de similitud S opera en el rango de intensidades de la imagen, mientras que la función de cercanía o distancia C opera en el dominio espacial de la imagen. Tomasi y Manduchi proponen ambas funciones como gaussianas, que se pueden expresar de la siguiente manera:

$$C(x_0, y_0, x_1, y_1) = e^{\left(-\frac{1}{2}\frac{d_1}{\sigma_d^2}\right)} \tag{2.30}$$

con $d_1 = (x_0 - x_1)^2 + (y_0 - y_1)^2$, y la ecuación

$$S(a,b) = e^{\left(-\frac{1}{2}\left(\frac{a-b}{\sigma_r}\right)^2\right)} \tag{2.31}$$

donde σ_d y σ_r controlan el nivel de suavizado.

Combinar intensidades de pixeles en la imagen para reemplazar el valor de intensidad de un pixel (x, y), solo tiene sentido si se encuentran lo suficientemente cerca del pixel. Por tal motivo, se introdujo el filtrado bilateral que resulta de la combinación del filtrado basado en la función de cercanía C con la de similitud S. Así, el filtrado bilateral se resume en la ecuación 2.32:

$$I_1(x,y) = \frac{1}{Z} \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} C(x,y,x+p,y+q) S(I(x,y), I(x+p,y+q)) I(x+p,y+q) \quad (2.32)$$

con la normalización

$$Z = \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} C(x,y,x+p,y+q) S(I(x,y), I(x+p,y+q)) \quad (2.33)$$

En otras palabras, se reemplaza el valor de intensidad del pixel (x, y) con un promedio de intensidades similares de pixeles cercanos. Lo anterior se basa en la consideración de que en regiones relativamente homogéneas, los valores de intensidad en pequeños vecindarios son similares entre sí, es decir, la función de normalización es casi la unidad. Este tipo de filtrado asigna pesos pequeños a los pixeles con poca relación al pixel central, los cuales se consideran como ruido.

Una propuesta muy parecida a la de Tomasi y Manduchi fue presentada por Smith y Brady [S.M. Smith97], con el algoritmo SUSAN (Segmento de valor único más pequeño que asimila el núcleo, *Smallest Univalue Segment Assimilating Nucleus*), el cual utiliza medidas locales para reducir ruido tratando de preservar estructuras en la imagen.

El filtro SUSAN lleva a cabo un promedio sobre todos los pixeles del vecindario, cuyos pesos toman en cuenta tanto la cercanía espacial como la similitud de intensidades, que a diferencia del filtrado bilateral no se incluye el pixel central con la finalidad de tener una mejor restauración. Para el pixel en la posición (x, y) y un vecindario de tamaño $W \times W$ con $W = 2m_1 + 1$, este filtro se puede expresar como

$$I_1(x,y) = \frac{1}{Z} \sum_{\substack{p=-m_1 \\ (p,q)\neq(0,0)}}^{m_1} \sum_{\substack{q=-m_1 \\ (p,q)\neq(0,0)}}^{m_1} I(x+p,y+q) e^{-\frac{r^2}{2\sigma_F^2}} e^{-\frac{(I(x,y)-I(x+p,y+q))^2}{t_h^2}} \quad (2.34)$$

donde Z es la constante de normalización

$$Z = \sum_{\substack{p=-m_1 \\ (p,q)\neq(0,0)}}^{m_1} \sum_{\substack{q=-m_1 \\ (p,q)\neq(0,0)}}^{m_1} e^{-\frac{r^2}{2\sigma_F^2}} e^{-\frac{(I(x,y)-I(x+p,y+q))^2}{t_h^2}} \tag{2.35}$$

donde $r = \sqrt{p^2 + q^2}$, σ_F controla la escala del suavizado espacial y t_h es un umbral de intensidad que controla la escala en el dominio de intensidades.

2.4.4. Medias no locales

Buades *et al.* en [A. Buades05a], propusieron el algoritmo de medias no-locales, soportado en la idea de que las imágenes contienen estructuras repetidas, y que promediando estas estructuras se puede reducir el ruido. Esto es, en lugar de usar promedios de valores de intensidad similares, este método promedia vecinos con vecindarios similares. Se define a un vecindario $N(i,j)$ centrado en el pixel con la posición (i,j), de tamaño $W \times W$ con $W = 2m_1 + 1$, como el conjunto

$$N(i,j) = \{I(i+p, j+q) : -m_1 \leq p \leq m_1, -m_1 \leq q \leq m_1\} \tag{2.36}$$

El valor restaurado $I_1(x,y)$, para el pixel en la posición (x,y), es calculado como el promedio pesado de todos los pixeles de la imagen, es decir,

$$I_1(x,y) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w(N(x,y), N(i,j)) I(i,j) \tag{2.37}$$

donde el peso $w(N(x,y), N(i,j))$ depende de la similitud $d(N(x,y), N(i,j))$ entre los vecindarios centrados en las posiciones (x,y) y (i,j), y satisface las condiciones
$0 \leq w(N(x,y), N(i,j)) \leq 1$ y $\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w(N(x,y), N(i,j)) = 1$.

La similitud entre los vecindarios centrados en las posiciones (x,y) e (i,j), es medida a través de la expresión

$$d(N(x,y), N(i,j)) = \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} \left(I(x+p, y+q) - I(i+p, j+q)\right)^2 \tag{2.38}$$

Los pixeles con vecindario similar a $N(x,y)$ tienen pesos más grandes dentro del promedio. Estos pesos se definen como

$$w(N(x,y), N(i,j)) = \frac{1}{Z} e^{-\frac{d(N(x,y),N(i,j))}{\sigma_F^2}} \tag{2.39}$$

donde Z es la constante de normalización

$$Z = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} e^{-\frac{d(N(x,y),N(i,j))}{\sigma_F^2}} \tag{2.40}$$

El parámetro σ_F^2 es la varianza y actúa para definir el nivel de suavizado, es decir, controla el decaimiento de los pesos como función de las distancias.

La implementación de este algoritmo proporciona excelentes resultados, sin embargo, puede llegar a tener un alto costo computacional ya que para cada pixel en la imagen se calculan $n = n_1 \times n_2$ pesos, lo que conduce a una complejidad cuadrática por pixel. Para reducir este problema, los autores propusieron la utilización de tamaños de vecindarios limitados, tanto para el cálculo del promedio como para la comparación de las estructuras en el vecindario. En este mismo sentido, existen trabajos que presentan variantes del método original para mejorar el costo computacional, como en [M. Mahmoudi05], donde se reduce el número de pesos calculados, rechazando vecindarios donde se esperan pesos pequeños, por medio del uso de filtros preliminares basados en promedios, para preclasificar bloques en la imagen. Solo los bloques con características similares se usan para calcular los pesos.

2.4.5. Filtro mediana

El filtro mediana es un filtro no lineal que consiste en reemplazar el valor de intensidad del pixel en la posición (x, y) por el valor de la mediana de las intensidades del conjunto de pixeles de un vecindario $N(x, y)$ centrado en el pixel con la posición (x, y), de tamaño $W \times W$ con $W = 2m_1 + 1$ [J. Astola97]. Lo anterior se puede resumir como

$$I_1(x, y) = \widetilde{N}(x, y) \tag{2.41}$$

donde $\widetilde{N}(x, y)$ es la mediana del vector que contiene todos los valores de intensidad de los pixeles del vecindario $N(x, y)$ centrado en la posición (x, y).

Los filtros basados en la mediana han sido ampliamente utilizados, debido a su capacidad de preservar bordes al suprimir ruido impulsivo. Sin embargo, esto no siempre sucede en imágenes con alta densidad de ruido, ya que la gran mayoría de los pixeles, sobre

los cuales se basa el cálculo de la mediana, están contaminados. Otra desventaja que puede presentar su empleo es que en ocasiones llegan a degradar la imagen en bordes curvos, esquinas o líneas delgadas.

2.4.6. Filtro de mediana ponderada

Al emplear el filtro mediana simple, cada muestra de la ventana tiene la misma influencia en la salida. Sin embargo, en ocasiones es deseable dar mayor importancia a algunos pixeles en posiciones específicas dentro de la ventana, es decir, aquellos que por alguna razón puedan ser considerados más confiables, por ejemplo, aquellos que han sido detectados mediante un proceso preliminar como no alterados. Esta idea condujo al desarrollo de los filtros mediana ponderada [Brownrigg84].

Se define una operación de repetición del valor a, w veces (w entero) por medio del símbolo $\Diamond$, esto es,

$$w \Diamond a = \overbrace{a, \cdots, a}^{w \; veces} \tag{2.42}$$

Por ejemplo, en el caso del vector $A = [a \;\; b \;\; c]^T$, se puede tener asociado un vector de pesos $w = [2 \;\; 1 \;\; 3]^T$, entonces, aplicando el operador de repetición se obtiene un nuevo vector $B = [2 \Diamond a \;\; 1 \Diamond b \;\; 3 \Diamond c]^T = [a \;\; a \;\; b \;\; c \;\; c \;\; c]^T$.

Sea $w(i, j)$ el peso para el pixel en la posición (i, j), entonces la salida del filtro de mediana ponderada para el pixel en la posición (x, y) se puede obtener por medio de la operación

$$I_1(x, y) = \widetilde{\phi} \tag{2.43}$$

donde $\widetilde{\phi}$ es el valor mediana del vector
$\phi = \{w(x + p, y + q) \Diamond I(x + p, y + q) | - m_1 \leq p \leq m_1, -m_1 \leq q \leq m_1\}$.

2.4.7. Filtros dependientes de los datos

Este tipo de filtros se basan en estadísticas locales. Kuan et al. en [D.T. Kuan85] propusieron un método basado en la mediana y la desviación estándar obtenida en una

ventana. De esta manera, la fórmula para el filtrado de ruido impulsivo es

$$I_1(x,y) = \frac{\hat{\sigma}^2_{\mathbf{I_0}}}{\hat{\sigma}^2_{\mathbf{I_0}} + \sigma^2_\eta} I(x,y) + \left(1 - \frac{\hat{\sigma}^2_{\mathbf{I_0}}}{\hat{\sigma}^2_{\mathbf{I_0}} + \sigma^2_\eta}\right) \widetilde{N}(x,y) \tag{2.44}$$

donde $\widetilde{N}(x,y)$ es el valor de la mediana en la ventana de tamaño $W \times W$ con $W = 2m_1 + 1$, σ_η es la desviación estándar del ruido y $\hat{\sigma}_{\mathbf{I_0}}$ es una estimación de la desviación estándar de la imagen original, el cuál está dado por la mediana de las desviaciones absolutas con respecto a la mediana

$$\hat{\sigma}_{\mathbf{I_0}} = \widetilde{\phi_2} \tag{2.45}$$

con

$$\phi_2 = \left\{ |I(x+p, y+q) - \widetilde{N}(x,y)| | -m_1 \leq p \leq m_1, -m_1 \leq q \leq m_1 \right\} \tag{2.46}$$

2.4.8. Filtros adaptables

Hwang y Haddad [H. Hwang95] presentaron un filtro mediana adaptable, mejorando el desempeño del filtro mediana variando o adaptando el tamaño de la ventana. El algoritmo consiste en dos niveles. El primer nivel busca la presencia de impulsos en la salida del filtro mediana. Si el primer nivel identifica que no se obtiene un impulso en la salida del filtro, el segundo nivel prueba que el pixel central está alterado por un impulso o no. Si el pixel central es clasificado como no alterado, entonces se deja como si no se le hubiera aplicado el filtro. Si no, la salida del filtro es reemplazada por la salida del filtro mediana del primer nivel. Del otro lado, si el primer nivel identifica que se obtiene un impulso en la salida del filtro mediana, entonces se incrementa el tamaño del la ventana y se repite el primer nivel.

En el primer nivel, la salida del filtro mediana puede tener tres posibles valores

$$I_1(x,y) = \begin{cases} I_{min} \\ I_{max} \\ \widetilde{N}(x,y) \end{cases} \tag{2.47}$$

donde $\widetilde{N}(x,y)$ es la mediana del vecindario centrado en la posición (x,y) y cuyo valor se encuentra entre I_{min} y I_{max}, que a su vez son el valor mínimo y el valor máximo respectivamente dentro de la ventana.

Si $I_1(x,y) = I_{min}$ o $I_1(x,y) = I_{max}$ se cumple, entonces se incrementa el tamaño de la ventana y se repite el primer nivel. Esta operación se realiza iterativamente hasta que $I_{min} < I_1(x,y) < I_{max}$ o se alcance un tamaño de ventana máximo establecido previamente.

Dentro del segundo nivel, si $I(x,y) = I_{min}$ o $I(x,y) = I_{max}$, el pixel es considerado corrupto y su valor es reemplazado por la salida $I_2(x,y) = I_1(x,y)$ de la última iteración del nivel 1. Si $I_{min} < I(x,y) < I_{max}$ la salida final será $I_2(x,y) = I(x,y)$.

Una propuesta más reciente donde se utilizan filtros adaptables fue hecha por Yuan y Tan [S.Q. Yuan06], donde como primer paso se detecta el ruido, primero realizando una estimación $\mathbf{I_1}$ de la imagen original obtenida de la aplicación del filtro mediana simple en la imagen $\mathbf{I}$ mediante el empleo de la ecuación (2.41). Posteriormente se determina una matriz de diferencias

$$\mathbf{I_2} = \mathbf{I} - \mathbf{I_1} \tag{2.48}$$

La matriz $\mathbf{I_2}$ se divide en bloques de tamaño fijo $W \times W$, siendo $I_{2blo}(x,y)$ un elemento de un bloque cualquiera. Entonces, se definen un par de límites

$$L_{sup} = \sqrt{\frac{1}{n_{blo1}} \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} (I_{2blo}(x+p, y+q))^2}, \quad \forall I_{2blo}(x+p, y+q) \geq 0 \tag{2.49}$$

siendo n_{blo1} el número de elementos mayores o iguales a cero en el bloque, y

$$L_{inf} = -\sqrt{\frac{1}{n_{blo2}} \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} (I_{2blo}(x+p, y+q))^2}, \quad \forall I_{2blo}(x+p, y+q) \leq 0 \tag{2.50}$$

siendo n_{blo2} el número de elementos menores o iguales a cero en el bloque.

Una vez determinados estos límites, si $I_{2blo}(x,y) > L_{sup}$ o $I_{2blo}(x,y) < L_{inf}$, el pixel en la posición (x,y) es clasificado como corrupto.

Una vez detectados los pixeles corruptos, se elige un tamaño de ventana mínimo. Si el número de pixeles no corruptos es al menos un valor establecido, el pixel corrupto es sustituido por la mediana de los pixeles no corruptos. De otra manera, si esta cantidad no llega a este valor se incrementa el tamaño de la ventana hasta que el criterio se cumpla o se alcance un tamaño máximo de ventana.

2.4.9. Filtros iterativos

Wang y Zhang [Z. Wang99] presentaron un método basado en iteraciones para detectar y remover ruido impulsivo, donde se propone una secuencia de imágenes en tonos de gris

$$\mathbf{I_t} = \{\mathbf{I_1}, \mathbf{I_2}, \cdots, \mathbf{I_k}, \cdots\},$$

con $\mathbf{I_1} = \mathbf{I}$, así como una secuencia de imágenes binarias

$$\mathbf{B_t} = \{\mathbf{B_1}, \mathbf{B_2}, \cdots, \mathbf{B_k}, \cdots\},$$

donde si $B_k(x, y) = 0$, significa que el pixel en la posición (x, y) no ha sido modificado y si $B_k(x, y) = 1$ significa que el pixel ha sido alterado. En la primera iteración se asume que todos los pixeles de la imagen se consideran inicialmente no alterados.

Para cada pixel en la posición (x, y) de la *k-ésima* iteración se calcula el valor de la mediana $\widetilde{N}(x, y)$ de su vecindario correspondiente en la imagen $\mathbf{I_{k-1}}$, de tal forma que se pueden detectar los impulsos de la manera

$$B_k(x,y) = \begin{cases} B_{k-1}(x,y) & si\,|I_{k-1}(x,y) - \widetilde{N}(x,y)| < T_D \\ 1 & de\,otra\,manera \end{cases} \tag{2.51}$$

donde T_D es un valor de umbralización definido. Entonces, el valor del pixel $I_k(x, y)$ será

$$I_k(x,y) = \begin{cases} \widetilde{N}(x,y) & si\,B_k(x,y) \neq B_{k-1}(x,y) \\ I_{k-1}(x,y) & si\,B_k(x,y) = B_{k-1}(x,y) \end{cases} \tag{2.52}$$

Este procedimiento de detección de impulsos se detiene hasta que ya no haya modificaciones en la imagen binaria $\mathbf{B_k}$ o se alcance un número máximo de iteraciones N_{iter}.

Para el procedimiento de filtrado se parte nuevamente de $\mathbf{I_1} = \mathbf{I}$ y se asigna a la imagen binaria inicial la última obtenida en la fase anterior, es decir, $\mathbf{B_1} = \mathbf{B_{N_{iter}}}$.

Partiendo de lo anterior, en la *k-ésima* iteración, para el pixel en la posición (x, y) se calcula el valor de la mediana del vecindario centrado en esta posición de la imagen $\mathbf{I_{k-1}}$, pero ahora, sólo se toman en cuenta los pixeles clasificados al momento como no alterados, es decir, aquellos que cumplan con $B_{k-1}(x + p, y + q) = 0$ para $-m_1 \leq p \leq m_1$

y $-m_1 \leq q \leq m_1$. Entonces, el nuevo valor del pixel $I_k(x, y)$ estará dado por

$$I_k(x,y) = \begin{cases} \widetilde{N}(x,y) & si\, B_{k-1}(x,y) = 1 \\ I_{k-1}(x,y) & de\, otra\, manera \end{cases} \tag{2.53}$$

Una vez que el pixel ha sido modificado, entonces el valor de $B_k(x, y)$ será

$$B_k(x,y) = \begin{cases} B_{k-1}(x,y) & si\, I_k(x,y) = I_{k-1}(x,y) \\ 0 & si\, I_k(x,y) = \widetilde{N}(x,y) \end{cases} \tag{2.54}$$

Este procedimiento se detiene cuando todos los pixeles inicialmente considerados como contaminados, han sido modificados.

2.4.10. Filtros cascada

Balasubramanian et al. en [S. Balasubramanian09] proponen un filtrado cascada para la supresión de ruido impulsivo, esto es, que después de haber procesado una imagen por medio de un filtro, se procesa la imagen resultante por otro filtro, y así sucesivamente.

En este trabajo, los autores proponen dos filtrados. El primero consiste en un filtro mediana basado en decisiones, el cuál utilizan para identificar pixeles alterados y reemplazarlos por el valor mediana. El segundo paso consiste en aplicar un filtrado truncado no simétrico, el cuál es usado para remover los pixeles alterados remanentes. La motivación de usar de manera sucesiva estos dos filtros, consiste en que el nivel de supresión de ruido con el filtro de la primera etapa no es tan bueno en altas densidades de ruido, la segunda etapa permite incrementar la supresión de éste.

La primera etapa consiste en emplear un filtro mediana basado en decisiones. Mientras que en un filtro mediana simple cada valor de todos los pixeles es reemplazado por la mediana de los valores en su vecindario sin importar si ha sido modificado o no, uno basado en decisiones trata de identificar si un pixel ha sido alterado o no. Así, si el pixel es identificado como alterado, su valor es reemplazado por el valor mediana de sus vecinos. Si es identificado como no alterado, su valor no es modificado.

Este filtrado se puede resumir, para el pixel en la posición (x, y) mediante la

siguiente expresión

$$I_1(x,y) = \begin{cases} I(x,y) & si\, I_{min} < I(x,y) < I_{max} \\ \widetilde{N}(x,y) & de\, otra\, manera \end{cases} \tag{2.55}$$

donde $\widetilde{N}(x,y)$ es el valor mediana del vector que contiene todos los pixeles del vecindario con centro en la posición (x,y).

La idea detrás de un filtro truncado consiste en rechazar los valores extremos en una ventana. El procedimiento seguido por los autores se puede resumir mediante los siguientes pasos:

1. Los valores de los pixeles del vecindario centrado en la posición (x,y) se ordenan de manera ascendente y se almacenan en un vector.

2. Los pixeles con valores iguales a I_{min} ó I_{max} son eliminados del vector.

3. El pixel en la posición (x,y) es reemplazado por la mediana de los pixeles restantes en el vector.

4. Si todos los valores del vector son eliminados, es decir, si el vector se queda sin elementos, entonces el valor del pixel en la posición (x,y) es reemplazado por el valor del pixel ya procesado más cercano.

2.5. Análisis y Experimentos

Con la finalidad de analizar imágenes restauradas por los métodos revisados en la sección anterior, su implementación se aplicó en imágenes con ruido gaussiano con $\sigma_\eta = 20$ y $\sigma_\eta = 60$, así como en imágenes con ruido sal y pimienta con $\delta = 20$ y $\delta = 50$.

En las Figuras 2.4 y 2.5 se muestran los resultados de aplicar algunos filtros, en la imagen con ruido gaussiano con desviación estándar de $\sigma_\eta = 20$ y $\sigma_\eta = 60$, respectivamente. Las desviaciones estándar utilizadas para el filtrado Gaussiano fueron de $\sigma_F = 1$ y de $\sigma_F = 10$ y el valor de varianza empleado para el filtro bilateral y de medias no locales es de $\sigma_F^2 = 20$, en cuanto a la difusión anisotrópica, el coeficiente que controla la difusión en función del gradiente le fue asignado un valor de $k = 0.25$.

Con base en las Figuras 2.4 y 2.5 se puede observar que todos los filtros empleados brindan mejores resultados en la imagen con bajo nivel de ruido ($\sigma_\eta = 20$) que con la de alto ($\sigma_\eta = 60$). En lo que respecta al filtro gaussiano, es evidente que cuando se utiliza un mayor nivel de suavizado, es decir una mayor desviación estándar del filtro (Figuras 2.4(c) y 2.5(c)), se obtiene una mejor eliminación del ruido, sin embargo, se presenta un mayor borroneo en los bordes. La difusión anisotrópica (Figuras 2.4(d) y 2.5(d)) preserva muy bien los bordes, pero la restauración en general no es muy buena, sobre todo en altos niveles de ruido. En cuanto al filtro bilateral (Figuras 2.4(e) y 2.5(e)), se tiene una mejor restauración, junto con un leve borroneo en los bordes. Finalmente, el algoritmo de medias no locales proporciona excelentes resultados en la imagen con bajo nivel de ruido (Figura 2.4(f) y en la imagen con alto nivel de ruido (Figura 2.5(f)) se preservan de manera aceptable los bordes, pero no hay una buena reducción del ruido.

En cuanto a la restauración de la imagen con ruido sal y pimienta, con $\delta = 50$, ninguno de los métodos anteriormente empleados da resultados satisfactorios, como se muestra en la Figura 2.6 debido a la dificultad de suavizar impulsos o grandes diferencias entre pixeles de una región.

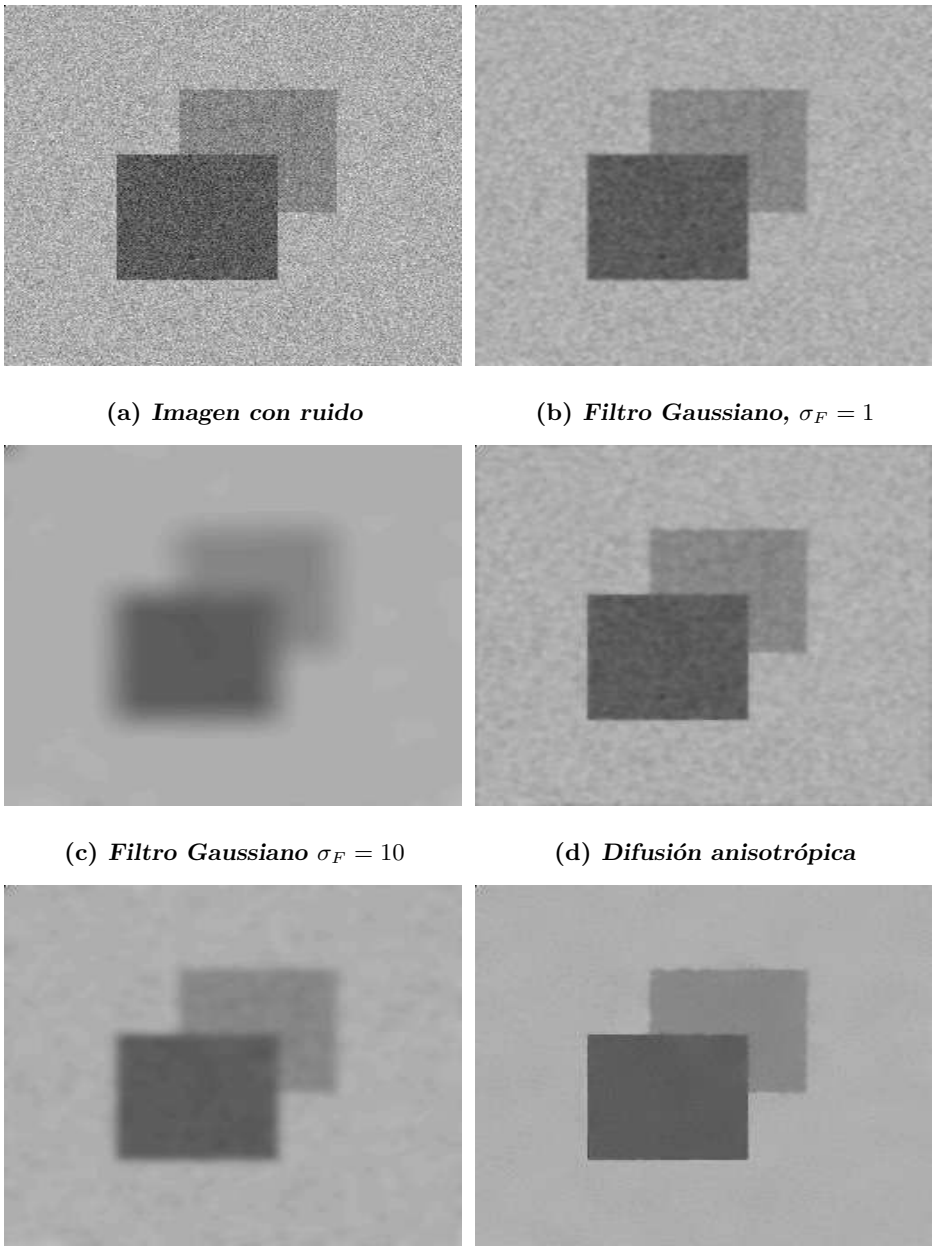
En las Figuras 2.7 y 2.8 se muestran los resultados de aplicar algunos filtros, en la imagen con ruido sal y pimienta con densidades de $\delta = 20$ y $\delta = 50$, respectivamente. Para el filtro mediana simple, mediana ponderada, de Kuan, iterativo y cascada se utilizaron ventanas de 5×5; para el filtro de mediana ponderada se usó un peso de 4 en el pixel central y 1 en los restantes. En lo que respecta al filtro iterativo se usó un umbral $T_D = 20$.

Con base en las Figuras 2.7 y 2.8 se puede evidenciar que todos los filtros empleados brindan mejores resultados en la imagen con bajo nivel de ruido ($\delta = 20$) que con la de mediano ($\delta = 50$). Para los dos niveles de densidad de ruido, las mejores restauraciones son obtenidas, en cuanto a la supresión de ruido y preservación de los bordes, con el filtro mediana ponderado y el filtro de Yuan (Figuras 2.7(c), 2.8(c), 2.7(f) y 2.8(f)).

2.6. Conclusiones

En este capítulo se han revisado algunos trabajos interesantes en lo referente a la supresión de ruido. En lo que respecta a la supresión de ruido gaussiano, la mayoría de los métodos de filtrado más comunes utilizan operaciones de promedios ponderados. Un problema aún no totalmente resuelto que presentan este tipo de métodos es que el grado de reducción de ruido es proporcional a la preservación de detalles o bordes. Es decir, que a mayor supresión de ruido, mayor será también la eliminación de detalles en la imagen original, así como el borroneo en los bordes. También es necesario mencionar que este tipo de filtros no proporcionan buenos resultados cuando la imagen se encuentra dañada con ruido sal y pimienta, debido a la dificultad de suavizar impulsos. Uno de los métodos que proporciona muy buenos resultados en la supresión de ruido gaussiano, considerado como parte del estado del arte, es el de medias no-locales [A. Buades05a], el cuál se basa en las características estructurales existentes en las imágenes para remover el ruido, lo que permite la reducción de ruido con una buena preservación de bordes y detalles. Sin embargo, una de las desventajas de su empleo es su costo computacional. En cuanto a la supresión de ruido impulsivo, es muy común el empleo de filtros basados en la mediana, debido a su efectividad y a la buena preservación de bordes que brinda, tanto para fines de detección como de remoción. Sin embargo, cuando se tienen altas densidades de ruido impulsivo, la restauración de estas imágenes deja de ser una tarea fácil.

De lo anterior se puede concluir la necesidad de contar con métodos que permitan restaurar imágenes dañadas con ruido gaussiano o sal y pimienta o, inclusive, una combinación de ambos, con buenos resultados (al menos similares a los reportados) y, de ser posible, en tiempos más cortos a los existentes.

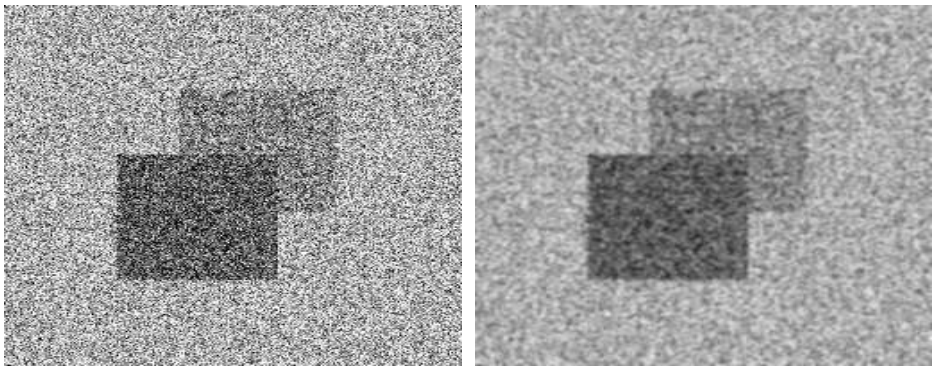


(a) ***Imagen con ruido*** **(b)** ***Filtro Gaussiano,*** $\sigma_F = 1$

(c) ***Filtro Gaussiano*** $\sigma_F = 10$ **(d)** ***Difusión anisotrópica***

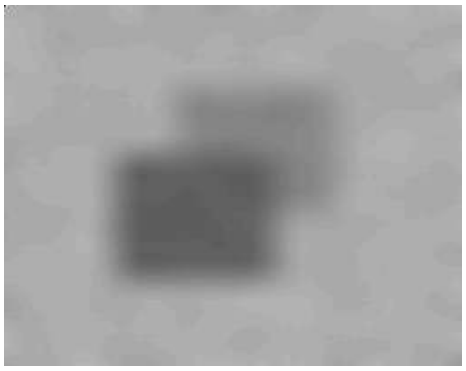
(e) ***Filtro bilateral*** **(f)** ***Medias no locales***

Figura 2.4: Resultados de diferentes filtros en una imagen con ruido gaussiano con $\sigma_\eta = 20$.

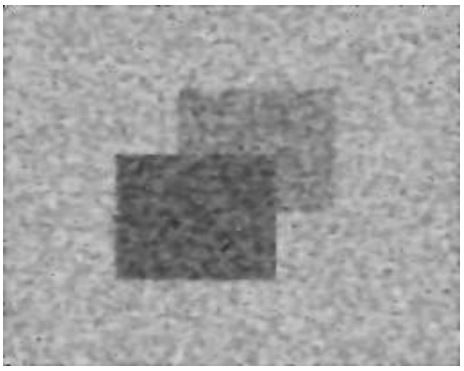


(a) ***Imagen con ruido***

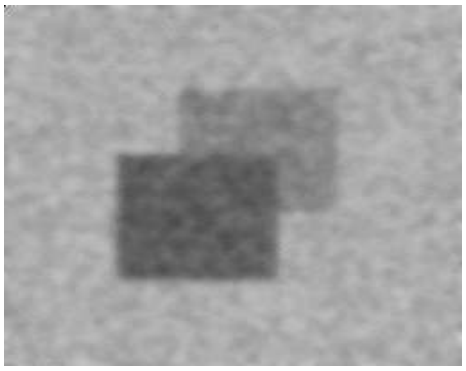
(b) ***Filtro Gaussiano,*** $\sigma_F = 1$



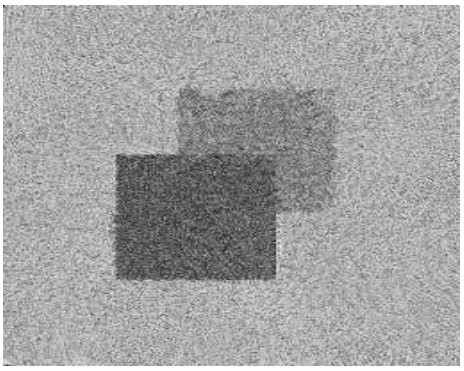
(c) ***Filtro Gaussiano,*** $\sigma_F = 10$



(d) ***Difusión anisotrópica***

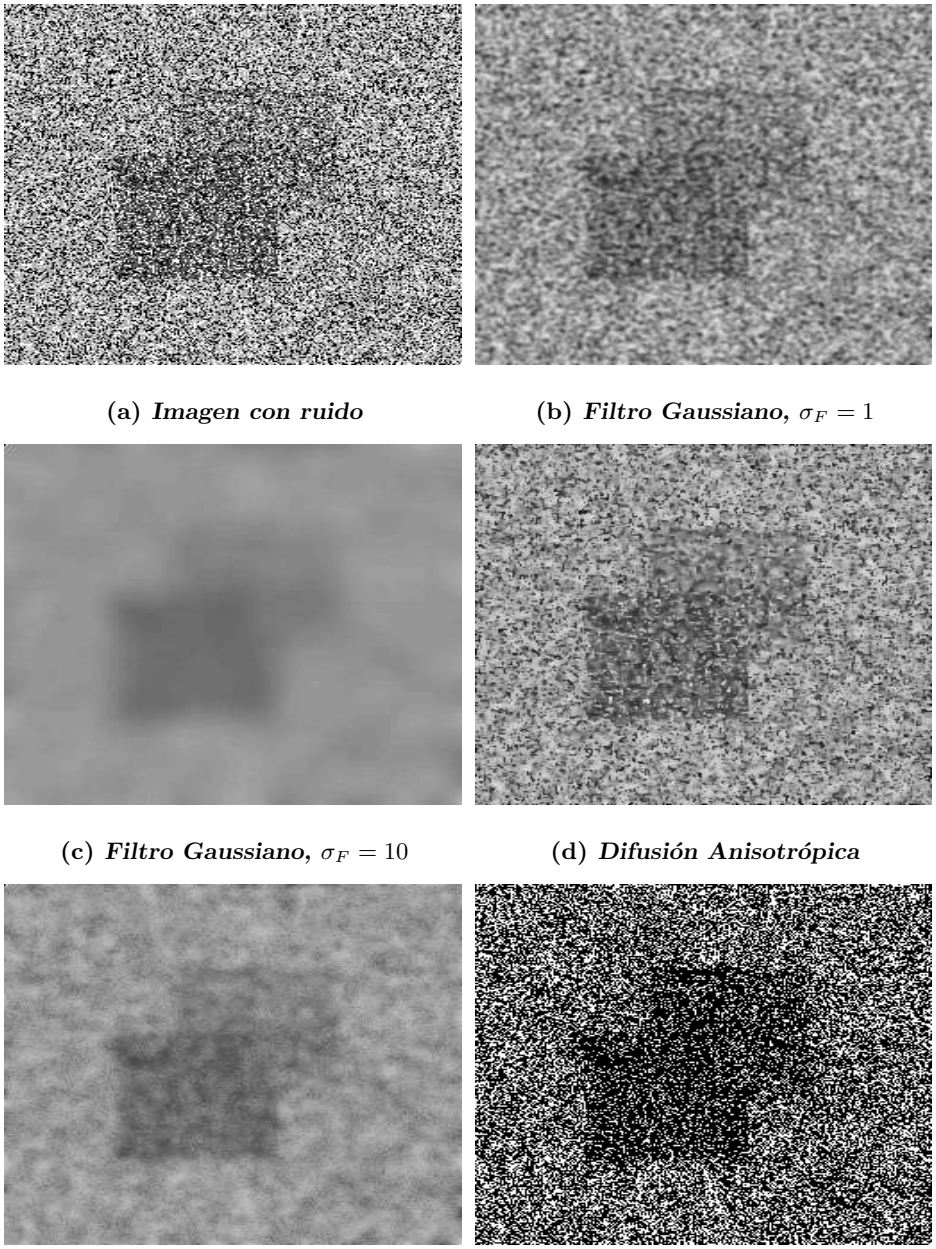


(e) ***Filtro bilateral***



(f) ***Medias no locales***

Figura 2.5: Resultados de diferentes filtros en una imagen con ruido gaussiano con $\sigma_\eta = 60$.

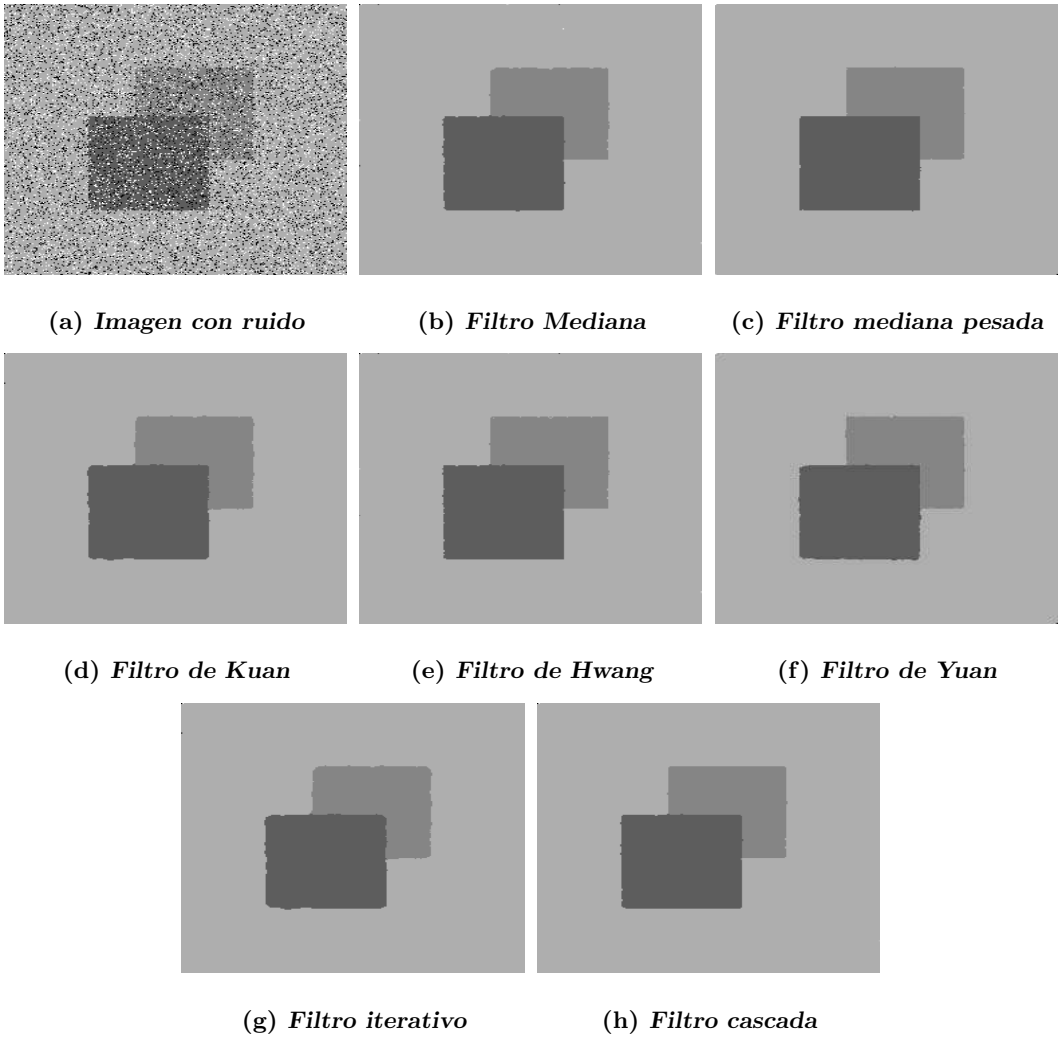


(a) ***Imagen con ruido*** **(b)** ***Filtro Gaussiano,*** $\sigma_F = 1$

(c) ***Filtro Gaussiano,*** $\sigma_F = 10$ **(d)** ***Difusión Anisotrópica***

(e) ***Filtro Bilateral*** **(f)** ***Medias No Locales***

Figura 2.6: Resultados de diferentes filtros en una imagen con ruido sal y pimienta con $\delta = 50$.

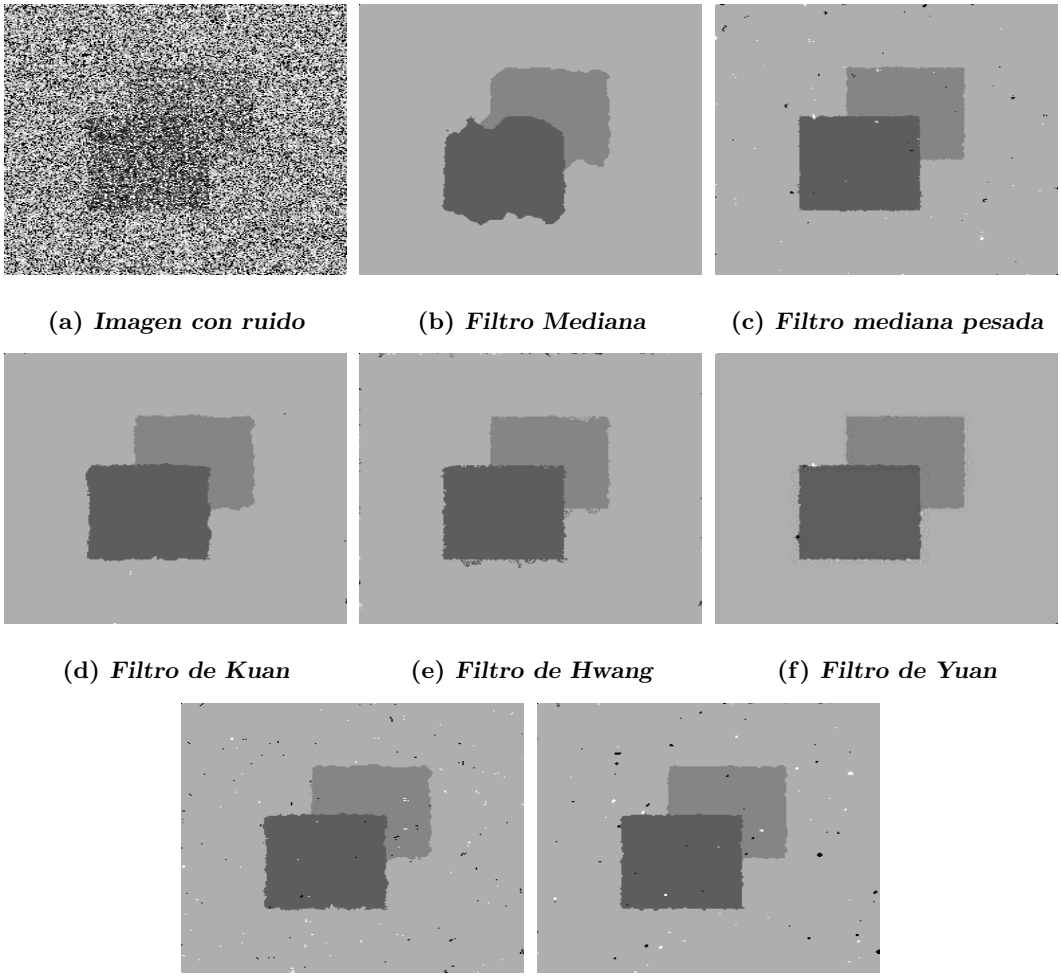


(a) ***Imagen con ruido*** **(b)** ***Filtro Mediana*** **(c)** ***Filtro mediana pesada***

(d) ***Filtro de Kuan*** **(e)** ***Filtro de Hwang*** **(f)** ***Filtro de Yuan***

(g) ***Filtro iterativo*** **(h)** ***Filtro cascada***

Figura 2.7: Resultados de diferentes filtros en una imagen con ruido sal y pimienta con $\delta = 20$.



(a) ***Imagen con ruido*** (b) ***Filtro Mediana*** (c) ***Filtro mediana pesada***

(d) ***Filtro de Kuan*** (e) ***Filtro de Hwang*** (f) ***Filtro de Yuan***

(g) ***Filtro iterativo*** (h) ***Filtro cascada***

Figura 2.8: Resultados de diferentes filtros en una imagen con ruido sal y pimienta con $\delta = 50$.

Capítulo 3

Reducción de ruido gaussiano

Como se mencionó en el Capítulo 2, diversos métodos han sido propuestos para la remoción de ruido gaussiano, y de entre ellos el método de medias no-locales [A. Buades05a] proporciona muy buenos resultados, sin embargo su implementación puede implicar un alto costo computacional, debido a que en su versión original tiene una complejidad cuadrática.

Con la finalidad de mejorar los tiempos de ejecución del método de medias no-locales, Mahmoudi y Sapiro en [M. Mahmoudi05] proponen la reducción de la cantidad de pesos por ser calculados, rechazando vecindarios donde se esperan pesos pequeños por medio del uso de filtros para preclasificar bloques en la imagen. Para llevar a cabo lo anterior, se emplea el valor de la media en el vecindario como una medida de similitud entre pixeles. Aquí, la idea consiste en considerar que vecindarios similares deben tener valores de media también similares. Entonces el cálculo del peso correspondiente a los pixeles en las posiciones (x, y) y (i, j) se realiza de la siguiente manera:

$$w(N(x,y),N(i,j)) = \begin{cases} \frac{1}{Z} e^{-\frac{d(N(x,y),N(i,j))}{\sigma_F^2}} & si\, T_{h1} < \frac{\overline{N}(x,y)}{\overline{N}(i,j)} < T_{h2} \\ 0 & de\, otra\, manera \end{cases} \tag{3.1}$$

donde la parte superior corresponde a la expresión (2.39) propuesta por Buades *et al*, $\overline{N}(x, y)$ y $\overline{N}(i, j)$ son la media de los valores de intensidad de los pixeles pertenecientes a los vecindarios centrados en las posiciones (x, y) y (i, j), respectivamente. Los valores T_{h1} y T_{h2} son los umbrales que permiten decidir si dos vecindarios pueden ser considerados similares con base en la razón de sus medias, teniendo en cuenta que $T_{h1} < 1 < T_{h2}$. En otras palabras,

se considera que dos vecindarios son similares si $\frac{\overline{N}(x,y)}{\overline{N}(i,j)} \cong 1$. Ambos umbrales son obtenidos experimentalmente en [M. Mahmoudi05].

Resulta claro que, mediante el procesamiento solamente de los pixeles seleccionados a través del criterio mencionado, el número de cálculos para determinar los pesos se reduce notablemente. Ésta es una de las operaciones que consume tiempo en el método de Medias No-Locales.

En lo que resta de este capítulo se presentan algunos algoritmos, para diferentes niveles de ruido, que aproximan de una manera más rápida los resultados obtenidos al emplear el método de medias no-locales tal y como se propone en [A. Buades05a], también reduciendo el número de pixeles que se procesan mediante este método, como se hizo en [M. Mahmoudi05]. A diferencia de la propuesta de Mahmoudi y Sapiro, se emplea un submuestreo de la imagen para reducir la cantidad de pixeles por ser procesados.

3.1. Submuestreo

El principal objetivo de realizar un submuestreo en una imagen es el de reducir el tiempo de ejecución de un proceso, mediante la disminución del tamaño de la imagen, i.e. la cantidad de pixeles por ser procesados.

Dada una imagen $\mathbf{I}^{(l)}$, siendo $\mathbf{I}^{(0)}$ la imagen original, la construcción de una imagen submuestreada [R. Marfil06]

$$\mathbf{I}^{(l+1)} = R(\mathbf{I}^{(l)}) \tag{3.2}$$

se obtiene a través de establecer una relación de dependencia entre cada pixel del nivel $l+1$, llamado padre, y un conjunto de pixeles del nivel l, al cual se le llama ventana de reducción, $W_r^{(l)}(x,y)$ para el pixel padre en la posición (x,y). A los pixeles pertenecientes a una ventana de reducción se les llama hijos. El valor de cada pixel padre es calculado con base en el conjunto de valores de intensidad de los pixeles hijos usando una función de reducción $R()$. El superíndice indica la posición de la imagen en una pirámide.

Una de las formas más comunes de realizar el submuestreo de imágenes, es seleccionando aleatoriamente un pixel de la ventana de reducción. Otra manera de realizarlo es

mediante el cálculo de un valor representativo de la ventana de reducción, por ejemplo, la media o la mediana de la ventana de reducción. Para ilustrar esta idea [terHaar Romeny94], por simplicidad, se considera que el tamaño de la imagen original $\mathbf{I}^{(0)}$ es $2^K \times 2^K$, entonces la función de reducción, con un factor de dos que equivale a usar una ventana de reducción $W_r^{(l)}$ de tamaño 2×2, con la que se obtiene la imagen $\mathbf{I}^{(1)}$ del nivel $l = 1$ y tamaño $2^{K-1} \times 2^{K-1}$ está determinada por

$$I^{(1)}(x,y) = R(W_r^{(0)}(x,y)) = \sum_{m=0}^{1}\sum_{n=0}^{1} c(2-m, 2-n) I^{(0)}(2x-m, 2y-n) \tag{3.3}$$

donde $c(i,j)$ con $(i,j) \in \{(1,1),(1,2),(2,1),(2,2)\}$ es un kernel de pesos que representa la contribución de cada pixel hijo en la formación del pixel padre. Si se repite la operación indicada en la expresión 3.3 de manera iterativa con los niveles subsecuentes se obtiene lo que se conoce como una pirámide pasabajas.

En este trabajo, como en [C.A. Júnez09a] y en [C.A. Júnez09b], se propone realizar un submuestreo basado en la segmentación de la imagen. Como primer paso, se lleva a cabo una segmentación de la imagen con ruido, mediante el agrupamiento de pixeles vecinos inmediatos con valores de intensidad similares, con base en un umbral T_I, procedimiento seguido en el trabajo de Adams y Bischof [R. Adams94]. Un pixel en la posición (x,y) es similar a un pixel vecino inmediato en la posición (i,j) si

$$|I(x,y) - I(i,j)| \leq T_I \tag{3.4}$$

donde $(i,j) \in \{\{x-1, x, x+1\} \times \{y-1, y, y+1\}\} - \{(x,y)\}$.
Un pixel no asignado a una región es considerado como la semilla de una nueva región, entonces, para este pixel semilla, se revisan sus vecinos inmediatos y se detectan aquellos que no tienen asignada región alguna. Hecho esto, aquellos cuya diferencia absoluta de intensidades con respecto al pixel semilla sea menor o igual que un umbral establecido, es agregado al agrupamiento pasando a formar parte de la región iniciada por el pixel semilla. Este proceso se repite para cada nuevo pixel agregado a la región (los cuales, ahora, se consideran como semilla) hasta que ya no haya posibilidades de crecimiento y hasta que toda la imagen sea segmentada.

El propósito de llevar a cabo esta segmentación es encontrar una partición de la

imagen $\mathbf{I}$ en un conjunto de n_Λ regiones Λ_s, de tal manera que

$$\bigcup_{s=1}^{n_\Lambda} \Lambda_s = \mathbf{I} \tag{3.5}$$

donde $\Lambda_s \cap \Lambda_k = \{\,\}, s \neq k$.

Entonces, se define

$$\Lambda_G = \{\Lambda_s | \upsilon(\Lambda_s) > P_I\} \tag{3.6}$$

y

$$\Lambda_P = \{\Lambda_s | \upsilon(\Lambda_s) \leq P_I\} \tag{3.7}$$

de tal manera que

$$\Lambda_G \cup \Lambda_P = \mathbf{I} \tag{3.8}$$

donde Λ_G es el conjunto de regiones consideradas como objetos relevantes; esto es, son aquellas regiones cuyo número de pixeles representan un porcentaje de la imagen mayor que un umbral P_I. De manera obvia, Λ_P es el conjunto de regiones que son, por su tamaño, consideradas como detalles o mayormente afectadas por ruido ya que representan un pequeño porcentaje de la imagen. La función $\upsilon(\Lambda_s)$ proporciona el porcentaje de la imagen que corresponde a la región Λ_s.

Así, con base en la segmentación descrita se puede discriminar entre pixeles que pueden ser considerados como levemente (aquellos que pertenecen a una región relativamente grande) o fuertemente alterados (aquellos que pertenecen a una región relativamente pequeña), para así realizar el submuestreo.

Se definen los conjuntos

$$\Upsilon_G = \{I^{(l)}(i,j) | I^{(l)}(i,j) \in W_r^{(l)}(x,y),\ I^{(l)}(i,j) \in \Lambda_G\} \tag{3.9}$$

y

$$\Upsilon_P = \{I^{(l)}(i,j) | I^{(l)}(i,j) \in W_r^{(l)}(x,y),\ I^{(l)}(i,j) \in \Lambda_P\} \tag{3.10}$$

entonces, el submuestreo se puede llevar a cabo como sigue

$$I^{(l+1)}(x,y) = \begin{cases} \overline{\Upsilon}_G & \Upsilon_G \neq \{\,\} \\ \overline{\Upsilon}_P & \Upsilon_G = \{\,\} \end{cases} \tag{3.11}$$

donde $\overline{\Upsilon}_G$ y $\overline{\Upsilon}_P$ son la media de los conjuntos Υ_G y Υ_P, respectivamente.

Este tipo de submuestreo reduce la cantidad de pixeles, eliminando algo de ruido al mismo tiempo, ya que privilegia los pixeles que pertenecen a una región grande (que puede ser un objeto) con respecto a una pequeña que pueden ser pixeles fuertemente alterados o detalles que se pierden en escalas mayores.

Una alternativa a la propuesta anterior, la cual se presenta en este trabajo, consiste en que el valor del pixel padre será igual al valor del pixel hijo que tenga la mayor cantidad de pixeles parecidos a él, aunque no se encuentren ligados a la misma región, y que se encuentren en un vecindario centrado al pixel mencionado. El criterio anterior pretende priorizar el valor de intensidad que posiblemente sea el menos afectado por el ruido, dicho en otras palabras, se pretende reducir la influencia de un pixel fuertemente alterado, a partir de la consideración de que un pixel con estas características no tendrá vecinos similares a él. Además, este criterio no genera importante borroneo en los bordes. Lo anterior se puede representar como

$$I^{(l+1)}(x,y) = \underset{I^{(l)}(i,j)}{\arg\max}\, f(I^{(l)}(i,j)) \tag{3.12}$$

donde $I^{(l)}(i,j) \in W_r^{(l)}(x,y)$ y la función $f(I^{(l)}(i,j))$ proporciona el número de pixeles similares en intensidad (con respecto a un umbral establecido) dentro de un vecindario de tamaño $W \times W$ centrado en la posición (i,j).

A continuación se presentan dos algoritmos para llevar a cabo la remoción de ruido gaussiano en la imagen submuestreada y el posterior sobremuestreo, el primero de ellos aplica el algoritmo de Medias No Locales simple con submuestreo (NLMS) y el segundo utiliza una variante robusta (NLMR).

3.2. Reducción de Ruido con NLMS

Una vez que se tiene la imagen submuestreada, el paso que sigue consiste en aplicarle a ésta el método de medias no-locales de acuerdo con las expresiones (2.37-2.40), obteniéndose, para el nivel l, la imagen restaurada $\hat{\mathbf{I}}^{(l)}$. Como se trata de una imagen más pequeña, es posible utilizar tamaños de ventanas menores que aquellos que se emplearían en la imagen de tamaño original, lo cual reduce el tiempo de ejecución.

Una de las principales ventajas del método presentado por Buades *et al* es que preserva de muy buena manera los bordes, sin embargo, al emplear una técnica de submuestreo y sobremuestreo. Esta ventaja desaparece ya que en la etapa de sobremuestreo es común utilizar una interpolación con el objetivo de asignar un valor a los pixeles que no fueron tomados en cuenta en la etapa de submuestreo, provocando con frecuencia cierto borroneo en los bordes. Para evitar lo anterior, se propone el procedimiento que se describe a continuación.

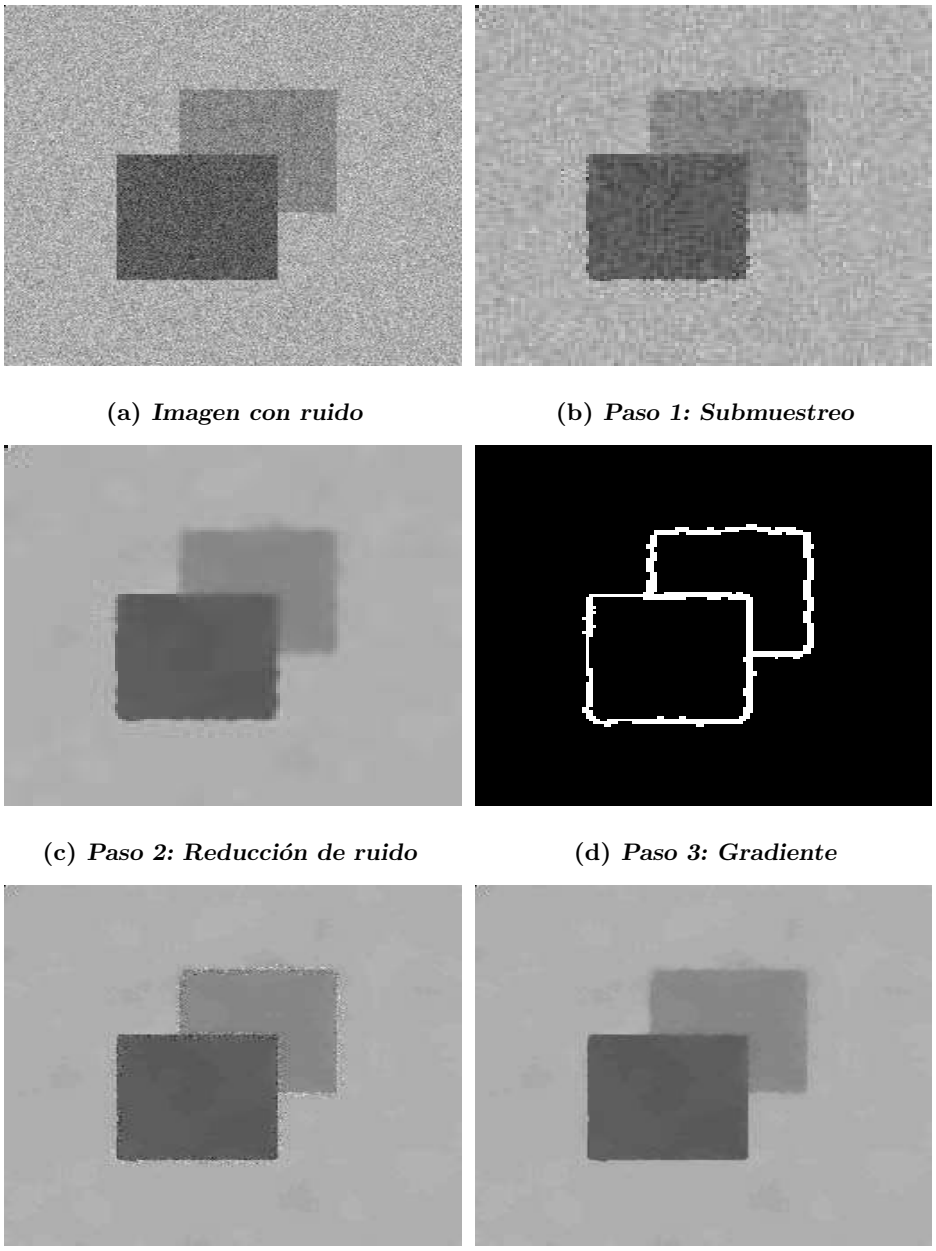
Ya que se ha restaurado la imagen submuestreada y obtenido $\hat{\mathbf{I}}^{(l)}$, se procede a la detección de los posibles bordes presentes en ésta por medio del gradiente de la imagen, calculado con la expresión (2.16). El siguiente paso es el de llevar acabo el sobremuestreo bajo el criterio de que, si el pixel padre $I^{l+1}(x, y)$ tiene una magnitud de gradiente baja con respecto a un umbral establecido, es poco probable que sea un borde y entonces todos los pixeles hijos de la imagen sobremuestreada $\hat{\mathbf{I}}^{(l)}$, cuyas posiciones pertenezcan a la ventana de reducción $W_r^{(l)}(x, y)$, toman el valor del pixel padre, es decir,

$$\hat{I}^{l}(i, j) = \hat{I}^{(l+1)}(x, y), \;\; I^{(l)}(i, j) \in W_r^{(l)}(x, y) \tag{3.13}$$

En el otro sentido, si el pixel padre $I^{(l+1)}(x, y)$ tiene una magnitud de gradiente alta, entonces es posible que pertenezca a un borde, por lo cual los pixeles hijos de la imagen sobremuestreada conservan el valor de intensidad no restaurado de la imagen con ruido en su posición correspondiente. Esto es,

$$\hat{I}^{(l)}(i, j) = I^{(l)}(i, j), \;\; I^{(l)}(i, j) \in W_r^{(l)}(x, y) \tag{3.14}$$

De esta manera, la imagen restaurada tendrá su tamaño original, pero aún con algunos pixeles alterados (precisamente en los bordes). Solamente a estos pixeles se le aplica el método de Medias No Locales, como se ha propuesto en [A. Buades05a], obteniéndose la imagen $\hat{\mathbf{I}}$ correspondiente a la restauración de la imagen original $\mathbf{I}^{(0)}$. Al proceso anterior se le denomina Medias No-Locales con Submuestreo (NLMS). En la Figura 3.1 se muestra el procedimiento general seguido para la restauración de una imagen con ruido.



(a) ***Imagen con ruido*** (b) ***Paso 1: Submuestreo***

(c) ***Paso 2: Reducción de ruido*** (d) ***Paso 3: Gradiente***

(e) ***Paso 4: Sobremuestreo*** (f) ***Paso 5: Restauración final***

Figura 3.1: Procedimiento propuesto de restauración.

3.3. Reducción de Ruido con NLMR

Para imágenes con mayores niveles de ruido, el procedimiento descrito en la sección anterior es el mismo, salvo una diferencia en la expresión para determinar los pesos en el

método de medias no locales.

Es evidente que la restauración de imágenes con altos niveles de ruido es más complicada de realizarse, debido a que para restaurar un pixel alterado, solo se cuenta con información alterada. Lo mejor que se puede hacer es obtener un estimado del valor de intensidad que pudiera tener un pixel perteneciente a una zona de la imagen. La propuesta de este trabajo es agregar un término a la expresión (2.39) que aumente o reduzca la influencia de un pixel de acuerdo con su grado de coherencia espacial. Por coherencia espacial se entiende al grado de pertenencia de un pixel a una región de acuerdo con las características de ésta. En otras palabras, determina si es lógico que un pixel pertenezca a una región de acuerdo con las características de ambos. Por ejemplo, si un pixel tiene un valor de intensidad de 250, es poco lógico que se encuentre en una región donde casi todos tengan un valor de intensidad de 100 (a menos de que estemos hablando de una textura).

En este trabajo, la coherencia espacial se mide mediante la diferencia del valor de intensidad del pixel y la media de una región. Esto es, mide la similitud entre un pixel que se va a utilizar en la restauración de otro pixel y la media de la región a la que pertenece este último. Lo anterior conduce a la expresión

$$w(N(x,y),N(i,j)) = \frac{1}{Z} e^{-\frac{d(N(x,y),N(i,j))}{\sigma_F^2}} e^{-\frac{(I(i,j)-\overline{N}(x,y))^2}{H_F^2}} \tag{3.15}$$

donde Z es la constante de normalización

$$Z = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} e^{-\frac{d(N(x,y),N(i,j))}{\sigma_F^2}} e^{-\frac{(I(i,j)-\overline{N}(x,y))^2}{H_F^2}} \tag{3.16}$$

donde $\overline{N}(x,y)$ es la media de los valores de intensidad de los pixeles pertenecientes al vecindario $N(x,y)$, centrado en la posición (x,y). El parámetro H_F controla el nivel de participación del término de coherencia espacial.

Al proceso anterior se le denomina Medias No-Locales Robusta (NLMR).

3.4. Experimentos

La implementación de los algoritmos propuestos se aplicó en las imágenes que se muestran en las Figuras 3.2, 3.3 y 3.4, las cuales fueron alteradas con ruido gaussiano con diferentes desviaciones estándar. Con la finalidad de evaluar el desempeño de los algoritmos propuestos (NLMS) y (NLMR), éstos se comparan con los presentados por Buades *et al.* (NLM) y por Mahmoudi y Sapiro (MAH). Todos los algoritmos mencionados fueron ejecutados con Matlab R2007b en una PC con procesador Intel Core2 Duo, 2.20 Ghz.

Los parámetros empleados en el algoritmo de NLM son los reportados por Buades et al. en [A. Buades05a]: un tamaño de ventana de vecindario de búsqueda $W_N = 21$ (lo que significa una ventana de 21×21), un tamaño de ventana de vecindario de similitud $W_S = 7$ y el grado de suavizado $\sigma_F = 2\sigma_\eta$. De igual manera, los utilizados en MAH son los reportados en [M. Mahmoudi05]: $W_N = 21$, $W_S = 7$ y $\sigma_F = 2\sigma_\eta$, la discriminación de pixeles con respecto a los valores de la media de vecindarios se hizo usando los umbrales $T_{h1} = 0.9$ y $T_{h2} = 1.1$.

Los valores de los parámetros empleados tanto en NLMS y NLMR fueron $W_N = 11$ y $W_N = 5$ en la imagen submuestreada y $W_N = 21$ y $W_N = 7$ en la imagen sobremuestreada, el grado de suavizado usado fue $\sigma_F = 2\sigma_\eta$, el submuestreo se realizó con una ventana de reducción $W_r^{(l)}$ de tamaño 2×2, para la segmentación se utilizó un valor de umbral $T_I = 20$, el umbral para la discriminación de gradientes fue de 30. En el caso del algoritmo NLMR se utilizó $H_F = 6$.

En las Tablas 3.1-3.12 se presentan los resultados obtenidos con los cuatro algoritmos mencionados. Se compara tanto la relación señal a ruido de pico (PSNR), el error medio absoluto (MAE) y el índice universal de calidad en imágenes (UIQI). Para el PSNR un valor mayor, significa una mayor calidad en la restauración, caso contrario con el error medio absoluto, donde un menor valor corresponde a una mejor calidad de restauración. En el caso del UIQI, la mejor calidad de restauración tiende a 1. También se presentan los tiempos de restauración en segundos.

Analizando los resultados de las tablas, se puede observar que se obtienen, por lo general para todos los algoritmos probados, valores similares en la calidad de restauración en

todos los indicadores, y la gran mayoría de las veces, los mejores resultados fueron obtenidos con el algoritmo (NLMS). En lo que respecta a los tiempos de ejecución, el algoritmo MAH reduce los tiempos y la calidad de restauración en imágenes con niveles bajos de ruido; sin embargo, en niveles medios y altos, también reduce los tiempos, pero con calidad de restauración menor comparado con NLM. Por otro lado, tanto NLMS como NLMR presentan menores tiempos de ejecución con calidades de restauración similares (en la mayoría de los casos, mejores) a los obtenidos con NLM.

Las Figuras 3.5-3.16 presentan los resultados experimentales para cuatro imágenes (Sintética, Lena, Mandril y Peppers) alteradas con diferentes niveles de ruido. Se puede observar en las imágenes, por lo general, que la menor calidad en la restauración la proporciona el algoritmo MAH, así como una mejor definición en los bordes que se obtiene con NLMS y NLMR, comparados con NLM.

La Figura 3.17 muestra una comparación entre los resultados obtenidos con los cuatro algoritmos para las mismas cuatro imágenes. El algoritmo MAH brinda los mejores resultados en niveles bajos de ruido, en cambio, cuando este nivel aumenta, tanto NLMS como NLMR son los de mejor desempeño, mejorando un poco la calidad obtenida con NLM, en un tiempo menor.

Tabla 3.1: Resultados NLM, ruido $\sigma_\eta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	33.86	2	0.05	149
Lena	29.17	6	0.49	612
Mandril	22.73	12	0.43	613
Peppers	24.92	9	0.54	148
Bee	28.67	6	0.37	596
Butterfly	24.98	8	0.52	604
Cat	26.41	8	0.26	596
Duck1	24.48	10	0.37	600
Duck2	25.50	8	0.34	602
Fish	23.44	12	0.41	596
Owl	23.62	8	0.42	599
Bike	24.62	9	0.54	601
Plane	27.28	6	0.39	595
Boat	26.09	9	0.38	603
Lake1	24.81	9	0.34	600
Lake2	25.43	9	0.47	605
Land1	18.59	26	0.57	599
Land2	23.55	8	0.52	600
Cameraman	27.70	6	0.27	610
Actor	25.18	10	0.38	599
Barbara	25.79	9	0.55	616
Elaine	28.60	7	0.43	596
Tvset	25.27	10	0.39	597
Village	25.72	9	0.37	609
House1	25.33	9	0.41	598
House2	27.77	10	0.35	147
House3	24.56	9	0.29	604
Moon	27.21	8	0.25	147
Partenon	23.53	8	0.31	603
Pentagon	25.21	35	0.33	2476

Tabla 3.2: Resultados NLM, ruido $\sigma_\eta = 40$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	29.23	4	0.04	147
Lena	25.02	10	0.36	601
Mandril	20.60	16	0.23	599
Peppers	20.48	15	0.43	147
Bee	26.03	8	0.29	603
Butterfly	21.95	12	0.38	598
Cat	23.77	12	0.16	605
Duck1	21.91	14	0.21	604
Duck2	23.67	9	0.23	601
Fish	20.14	18	0.26	599
Owl	21.30	26	0.28	603
Bike	18.53	26	0.38	599
Plane	22.93	10	0.26	599
Boat	23.08	12	0.25	601
Lake1	22.54	26	0.26	601
Lake2	22.11	26	0.34	603
Land1	16.47	26	0.22	611
Land2	20.52	26	0.25	602
Cameraman	23.40	11	0.18	602
Actor	21.92	16	0.24	597
Barbara	22.17	14	0.37	601
Elaine	25.86	9	0.35	598
Tvset	22.68	13	0.23	607
Village	23.47	12	0.24	602
House1	21.98	13	0.27	602
House2	23.48	28	0.21	147
House3	23.38	27	0.12	603
Moon	25.72	9	0.19	147
Partenon	21.37	25	0.25	602
Pentagon	23.52	18	0.23	2453

Tabla 3.3: Resultados NLM, ruido $\sigma_\eta = 60$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	27.32	5	0.03	150
Lena	22.70	13	0.28	600
Mandril	19.88	17	0.15	633
Peppers	17.37	23	0.32	146
Bee	24.38	11	0.23	606
Butterfly	20.64	14	0.30	600
Cat	21.79	17	0.12	610
Duck1	20.89	16	0.14	597
Duck2	22.69	11	0.18	599
Fish	18.24	23	0.18	611
Owl	20.13	26	0.22	600
Bike	16.20	26	0.24	594
Plane	20.82	15	0.18	594
Boat	21.68	15	0.19	603
Lake1	21.36	26	0.16	601
Lake2	20.24	26	0.28	602
Land1	15.64	26	0.16	615
Land2	19.28	26	0.13	598
Cameraman	21.01	15	0.12	604
Actor	20.21	19	0.17	599
Barbara	20.65	17	0.28	601
Elaine	23.88	12	0.29	607
Tvset	21.50	15	0.16	613
Village	22.36	14	0.19	600
House1	20.37	16	0.19	596
House2	21.68	27	0.11	146
House3	22.64	29	0.10	609
Moon	24.89	10	0.16	146
Partenon	20.54	26	0.16	616
Pentagon	22.81	26	0.25	2431

Tabla 3.4: Resultados MAH, ruido $\sigma_\eta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	28.39	7	0.06	145
Lena	27.82	8	0.45	349
Mandril	25.77	10	0.70	380
Peppers	26.82	9	0.63	67
Bee	27.60	8	0.36	423
Butterfly	26.61	9	0.57	343
Cat	27.54	8	0.44	292
Duck1	26.81	9	0.55	363
Duck2	27.73	8	0.51	350
Fish	26.24	10	0.63	269
Owl	26.46	8	0.78	239
Bike	27.23	8	0.68	377
Plane	28.45	7	0.41	473
Boat	27.06	9	0.54	408
Lake1	27.11	8	0.67	327
Lake2	27.32	8	0.56	348
Land1	22.23	26	0.86	338
Land2	26.74	8	0.73	420
Cameraman	27.81	8	0.35	425
Actor	26.75	9	0.57	262
Barbara	27.00	9	0.61	369
Elaine	28.32	8	0.52	415
Tvset	27.61	8	0.61	387
Village	27.55	8	0.57	359
House1	27.60	8	0.53	424
House2	28.83	8	0.44	111
House3	26.85	8	0.61	381
Moon	27.85	8	0.49	122
Partenon	25.99	8	0.57	382
Pentagon	27.67	32	0.61	1801

Tabla 3.5: Resultados MAH, ruido $\sigma_\eta = 40$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	19.73	21	0.04	137
Lena	19.79	20	0.23	322
Mandril	18.96	23	0.39	362
Peppers	20.46	19	0.41	63
Bee	19.61	21	0.16	378
Butterfly	19.29	22	0.28	322
Cat	20.46	18	0.22	269
Duck1	19.86	20	0.34	318
Duck2	20.18	20	0.20	320
Fish	20.10	20	0.41	268
Owl	20.05	20	0.44	234
Bike	21.24	21	0.50	371
Plane	20.84	18	0.24	464
Boat	19.44	21	0.28	385
Lake1	20.48	20	0.35	297
Lake2	20.36	20	0.37	334
Land1	18.18	26	0.73	324
Land2	20.13	20	0.41	418
Cameraman	19.90	20	0.20	403
Actor	19.91	20	0.32	258
Barbara	19.45	21	0.35	336
Elaine	20.31	19	0.22	391
Tvset	19.92	20	0.32	354
Village	20.13	20	0.27	337
House1	20.24	20	0.30	415
House2	20.34	20	0.21	104
House3	20.07	20	0.34	339
Moon	19.96	20	0.20	110
Partenon	19.85	19	0.27	357
Pentagon	19.82	28	0.38	1668

Tabla 3.6: Resultados MAH, ruido $\sigma_\eta = 60$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	15.91	33	0.02	128
Lena	16.18	31	0.14	298
Mandril	15.52	34	0.25	359
Peppers	17.02	28	0.31	62
Bee	16.07	32	0.10	346
Butterfly	15.77	33	0.18	309
Cat	16.84	27	0.13	266
Duck1	16.30	31	0.23	295
Duck2	16.34	31	0.11	298
Fish	16.67	29	0.28	265
Owl	16.56	31	0.35	235
Bike	17.73	33	0.45	371
Plane	17.16	28	0.16	448
Boat	15.75	33	0.18	364
Lake1	16.87	31	0.29	283
Lake2	16.78	31	0.28	325
Land1	15.34	26	0.58	306
Land2	16.74	31	0.36	411
Cameraman	16.19	31	0.14	379
Actor	16.39	30	0.21	249
Barbara	15.99	32	0.24	318
Elaine	16.35	31	0.12	371
Tvset	16.17	32	0.20	333
Village	16.41	31	0.16	320
House1	16.62	30	0.20	399
House2	16.37	30	0.16	97
House3	16.43	31	0.24	310
Moon	16.05	32	0.11	98
Partenon	16.24	30	0.11	338
Pentagon	16.01	27	0.24	1548

Tabla 3.7: Resultados NLMS, ruido $\sigma_\eta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	37.60	1	0.06	55
Lena	30.49	5	0.51	305
Mandril	23.84	11	0.49	420
Peppers	24.88	8	0.57	81
Bee	29.47	5	0.36	274
Butterfly	26.41	7	0.54	335
Cat	27.39	7	0.28	293
Duck1	26.09	9	0.43	351
Duck2	27.01	7	0.38	278
Fish	25.00	10	0.46	343
Owl	25.38	7	0.59	385
Bike	24.81	7	0.57	306
Plane	27.92	5	0.40	288
Boat	27.57	7	0.41	337
Lake1	26.07	8	0.31	320
Lake2	26.86	8	0.59	325
Land1	20.28	26	0.62	523
Land2	25.29	7	0.52	378
Cameraman	29.75	5	0.31	277
Actor	26.48	9	0.41	319
Barbara	26.85	8	0.56	391
Elaine	29.35	6	0.43	278
Tvset	26.83	8	0.45	338
Village	26.98	8	0.41	317
House1	25.66	8	0.45	330
House2	29.33	9	0.35	69
House3	25.59	7	0.36	312
Moon	27.74	7	0.26	67
Partenon	24.71	7	0.47	306
Pentagon	26.42	10	0.43	1244

Tabla 3.8: Resultados NLMS, ruido $\sigma_\eta = 40$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	30.78	4	0.05	142
Lena	27.53	7	0.42	552
Mandril	21.79	14	0.36	585
Peppers	23.84	10	0.53	126
Bee	27.46	7	0.31	565
Butterfly	24.05	10	0.45	572
Cat	25.30	10	0.23	432
Duck1	24.35	11	0.38	541
Duck2	25.77	9	0.34	528
Fish	23.58	12	0.45	452
Owl	23.59	9	0.49	453
Bike	22.78	9	0.57	364
Plane	26.15	8	0.33	545
Boat	24.98	10	0.34	563
Lake1	24.37	9	0.35	404
Lake2	24.61	10	0.43	457
Land1	18.76	26	0.56	528
Land2	23.17	9	0.54	475
Cameraman	26.67	8	0.25	498
Actor	24.27	11	0.35	468
Barbara	24.26	11	0.47	574
Elaine	27.33	8	0.39	528
Tvset	24.95	11	0.40	532
Village	25.54	10	0.38	533
House1	24.69	10	0.39	547
House2	26.94	10	0.33	107
House3	24.13	9	0.37	442
Moon	26.55	9	0.29	129
Partenon	22.80	9	0.38	438
Pentagon	24.73	10	0.35	1744

Tabla 3.9: Resultados NLMS, ruido $\sigma_\eta = 60$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	28.39	5	0.04	154
Lena	25.16	10	0.34	586
Mandril	20.73	16	0.27	636
Peppers	21.93	14	0.45	127
Bee	25.36	9	0.23	641
Butterfly	22.92	12	0.36	639
Cat	24.03	12	0.19	441
Duck1	22.45	14	0.31	583
Duck2	23.78	12	0.23	589
Fish	22.05	14	0.38	482
Owl	21.66	12	0.47	460
Bike	20.31	11	0.45	355
Plane	23.37	12	0.26	568
Boat	23.26	12	0.27	613
Lake1	22.53	13	0.36	1459
Lake2	22.27	13	0.31	486
Land1	17.77	26	0.46	540
Land2	21.15	12	0.42	494
Cameraman	24.28	10	0.19	548
Actor	22.69	13	0.28	494
Barbara	22.33	14	0.36	608
Elaine	24.56	12	0.29	581
Tvset	22.95	14	0.31	583
Village	23.45	13	0.29	585
House1	22.75	14	0.32	580
House2	23.83	13	0.23	122
House3	22.78	13	0.23	488
Moon	24.23	12	0.23	145
Partenon	21.46	12	0.39	485
Pentagon	23.42	13	0.35	1983

Tabla 3.10: Resultados NLMR, ruido $\sigma_\eta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	37.48	2	0.06	51
Lena	30.10	6	0.50	296
Mandril	23.81	11	0.49	406
Peppers	24.75	8	0.56	83
Bee	29.33	5	0.36	264
Butterfly	26.27	8	0.54	320
Cat	27.24	7	0.28	281
Duck1	25.84	9	0.42	358
Duck2	26.50	7	0.36	293
Fish	24.84	11	0.45	346
Owl	25.02	8	0.54	393
Bike	24.70	8	0.57	314
Plane	27.68	6	0.39	294
Boat	27.44	8	0.41	322
Lake1	25.94	8	0.32	323
Lake2	26.55	8	0.43	338
Land1	20.25	26	0.64	534
Land2	25.00	8	0.58	385
Cameraman	29.53	5	0.30	270
Actor	26.38	9	0.41	305
Barbara	26.67	8	0.56	358
Elaine	28.94	7	0.42	281
Tvset	26.54	9	0.43	345
Village	26.77	8	0.44	324
House1	25.46	8	0.44	333
House2	29.03	9	0.37	70
House3	25.37	8	0.25	318
Moon	27.56	7	0.25	68
Partenon	24.48	7	0.47	313
Pentagon	26.10	12	0.42	1269

Tabla 3.11: Resultados NLMR, ruido $\sigma_\eta = 40$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	30.46	4	0.05	102
Lena	26.65	8	0.40	432
Mandril	21.44	14	0.33	471
Peppers	22.64	11	0.50	107
Bee	26.94	7	0.29	433
Butterfly	23.25	11	0.43	446
Cat	24.20	13	0.21	330
Duck1	23.15	12	0.30	459
Duck2	24.27	9	0.28	444
Fish	22.38	12	0.37	396
Owl	22.46	10	0.46	453
Bike	21.55	10	0.44	372
Plane	25.13	9	0.31	463
Boat	24.53	11	0.32	440
Lake1	23.59	11	0.37	406
Lake2	23.94	11	0.41	466
Land1	17.94	26	0.44	530
Land2	22.16	10	0.43	482
Cameraman	26.09	9	0.23	393
Actor	23.75	11	0.33	373
Barbara	23.41	12	0.43	455
Elaine	26.85	9	0.37	430
Tvset	23.94	12	0.32	453
Village	24.72	11	0.36	443
House1	23.72	11	0.33	461
House2	25.88	12	0.35	108
House3	23.72	11	0.22	450
Moon	26.22	9	0.22	104
Partenon	22.21	10	0.39	440
Pentagon	24.13	14	0.34	1766

Tabla 3.12: Resultados NLMR, ruido $\sigma_\eta = 60$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	28.57	6	0.04	122
Lena	23.57	12	0.32	471
Mandril	20.87	16	0.25	510
Peppers	20.30	17	0.43	112
Bee	23.94	11	0.23	490
Butterfly	22.81	13	0.35	503
Cat	22.41	16	0.17	332
Duck1	21.55	15	0.23	504
Duck2	23.50	11	0.22	497
Fish	20.81	15	0.31	418
Owl	20.66	13	0.38	456
Bike	18.43	11	0.34	360
Plane	21.22	16	0.23	475
Boat	22.55	13	0.25	481
Lake1	21.58	13	0.29	430
Lake2	21.45	13	0.33	489
Land1	16.71	26	0.31	549
Land2	19.84	12	0.37	498
Cameraman	23.73	11	0.19	426
Actor	21.63	15	0.26	383
Barbara	21.42	16	0.33	485
Elaine	24.90	11	0.31	486
Tvset	22.40	14	0.25	497
Village	22.91	14	0.25	484
House1	21.52	15	0.26	491
House2	23.26	14	0.22	123
House3	21.97	13	0.25	501
Moon	25.17	10	0.20	120
Partenon	20.69	13	0.20	493
Pentagon	23.57	15	0.21	2000

3.5. Conclusiones

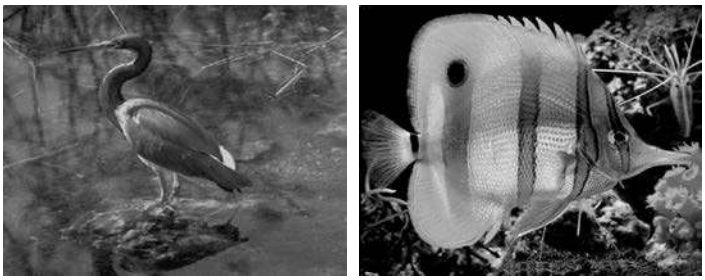
En este capítulo se han propuesto dos algoritmos para la reducción de ruido gaussiano. Los dos algoritmos tienen su fundamento en la idea presentada en [A. Buades05a], el cual es considerado como parte del estado del arte. La diferencia de ambas propuestas con respecto al método propuesto por Buades *et al*, es que llevan a cabo un submuestreo para reducir el tiempo de ejecución. Con base en los experimentos realizados, se puede hacer el comentario de que los algoritmos NLMS y NLMR proporcionan resultados similares a NLM y MAH para imágenes con ruido gaussiano con $\sigma_\eta = 20$, de acuerdo con los parámetros de la calidad de restauración considerados, consiguiéndose una reducción en el tiempo de ejecución aproximadamente a la mitad con respecto a NLM y un *10* % con respecto a MAH. Para imágenes con ruido gaussiano con $\sigma_\eta = 40$, se tiene una mejora media en la calidad de restauración de 1.1 veces. Con respecto a los tiempos, estos se reducen aproximadamente un *20* % comparándolos con NLM y son más altos 1.2 veces que con MAH. En imágenes con ruido gaussiano con $\sigma_\eta = 60$, se tiene una mejora media en la calidad de restauración de 1.2 veces. Con respecto a los tiempos, estos se reducen aproximadamente un *10* % comparándolos con NLM y son más altos 1.4 veces que con MAH.

(a) ***Sintética*** (b) ***Lena***

(c) ***Mandril*** (d) ***Peppers***

(e) ***Bee*** (f) ***Butterfly***

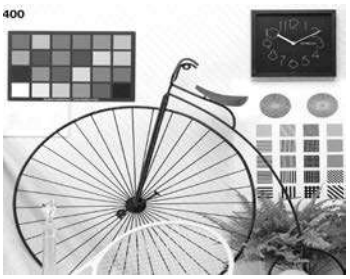
(g) ***Cat*** (h) ***Duck1***



(i) ***Duck2*** (j) ***Fish***

Figura 3.2: Imágenes empleadas en experimentos.

(a) ***Owl***



(b) ***Bike***

(c) ***Plane***

(d) ***Boat***

(e) ***Lake1***

(f) ***Lake2***

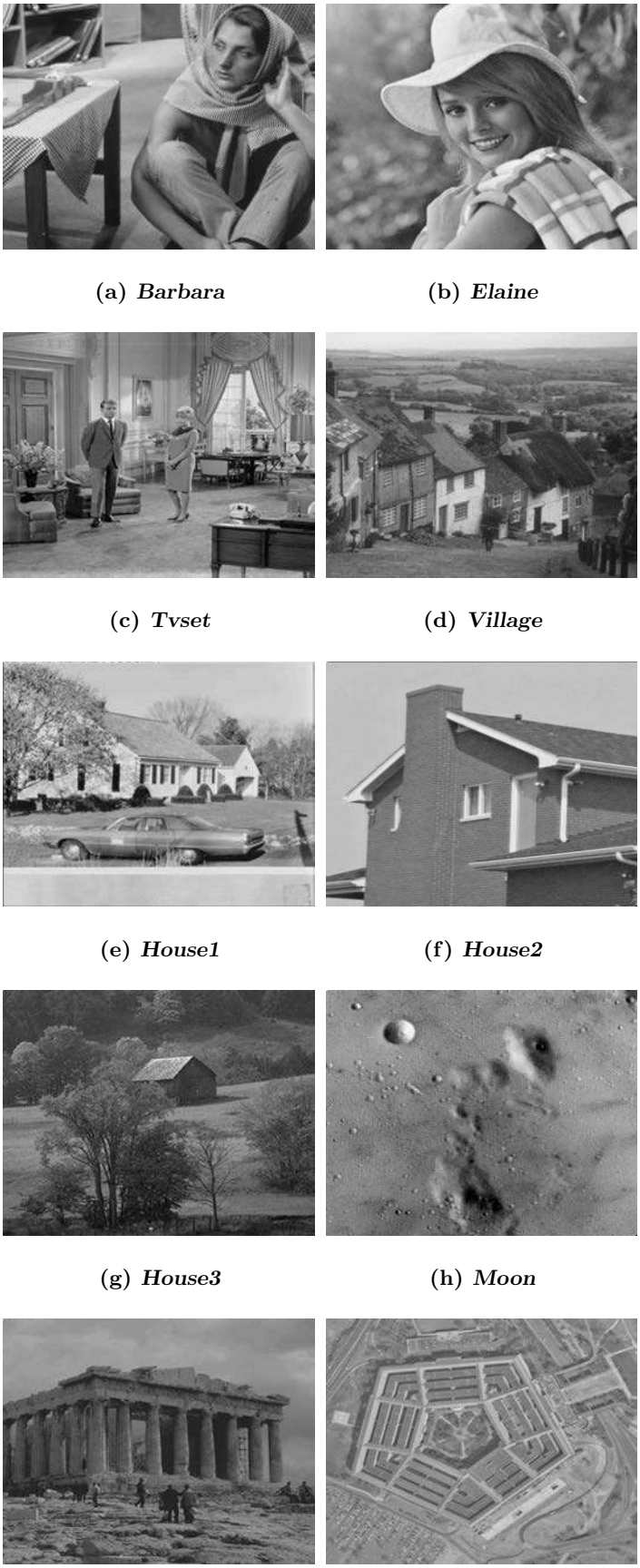
(g) ***Land1***

(h) ***Land2***

(i) ***Cameraman***

(j) ***Actor***

Figura 3.3: Imágenes empleadas en experimentos.



(a) ***Barbara*** **(b)** ***Elaine***

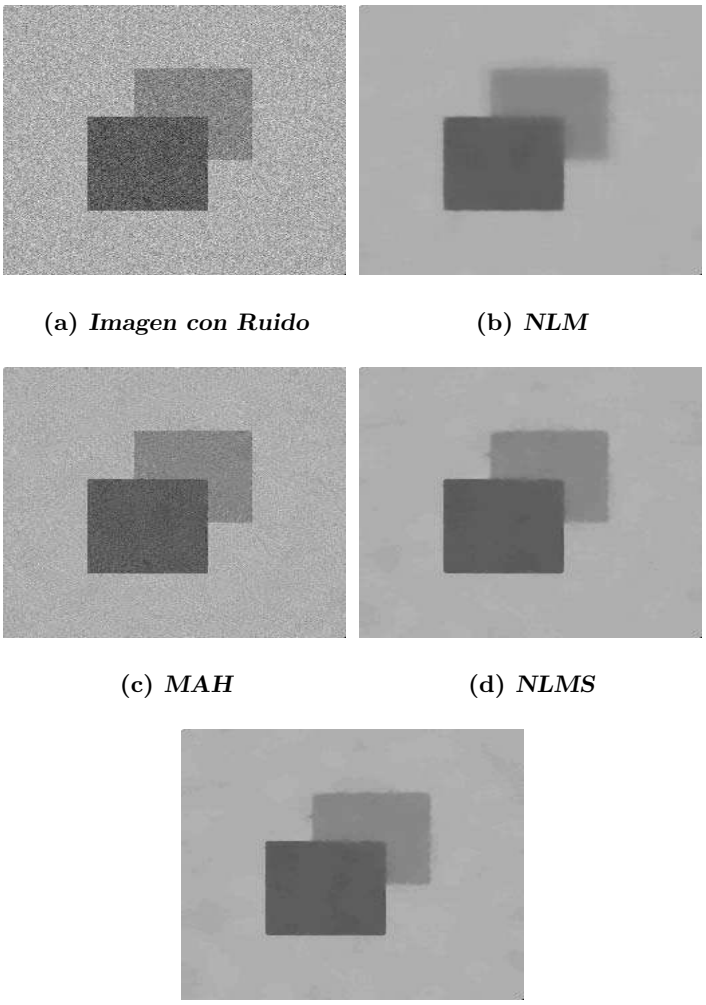
(c) ***Tvset*** **(d)** ***Village***

(e) ***House1*** **(f)** ***House2***

(g) ***House3*** **(h)** ***Moon***

(i) ***Partenon*** **(j)** ***Pentagon***

Figura 3.4: Imágenes empleadas en experimentos.

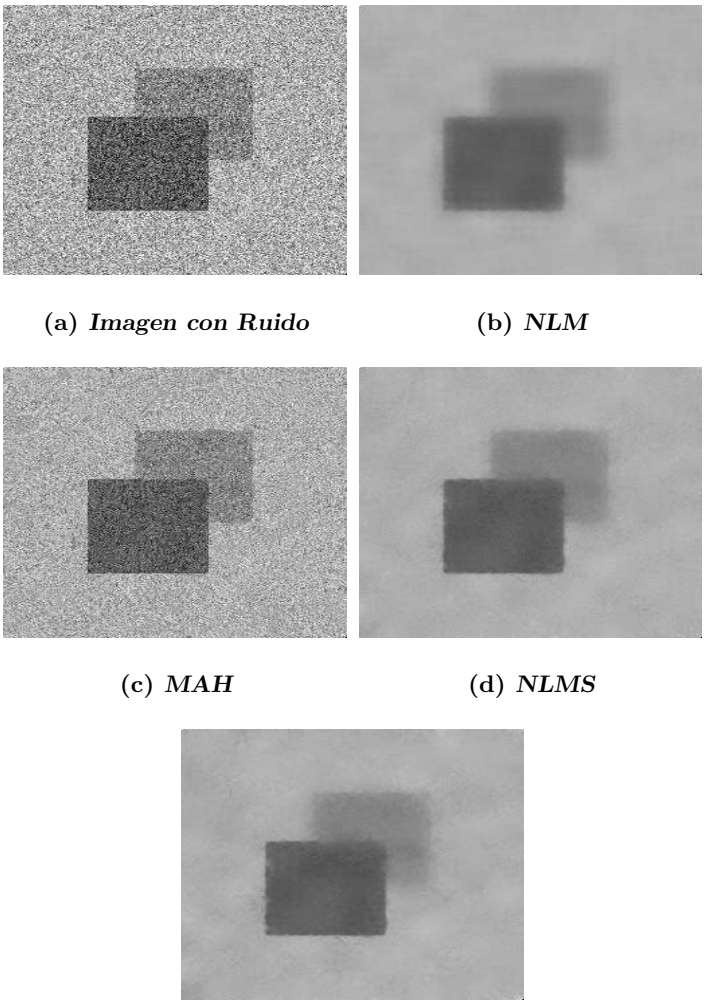


(a) ***Imagen con Ruido*** **(b)** ***NLM***

(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.5: Restauración de imagen sintética con ruido gaussiano $\sigma_\eta = 20$.

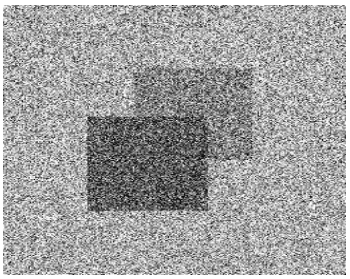


(a) ***Imagen con Ruido*** **(b)** ***NLM***

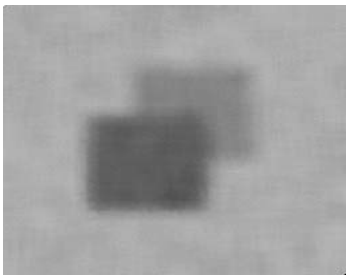
(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

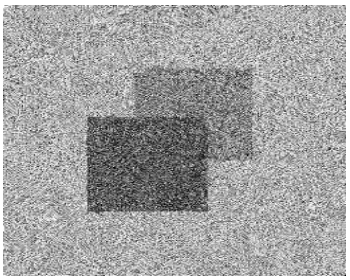
Figura 3.6: Restauración de imagen sintética con ruido gaussiano $\sigma_\eta = 40$.



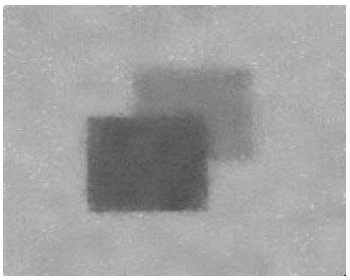
(a) ***Imagen con Ruido***



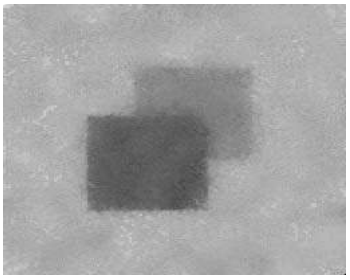
(b) ***NLM***



(c) ***MAH***



(d) ***NLMS***



(e) ***NLMR***

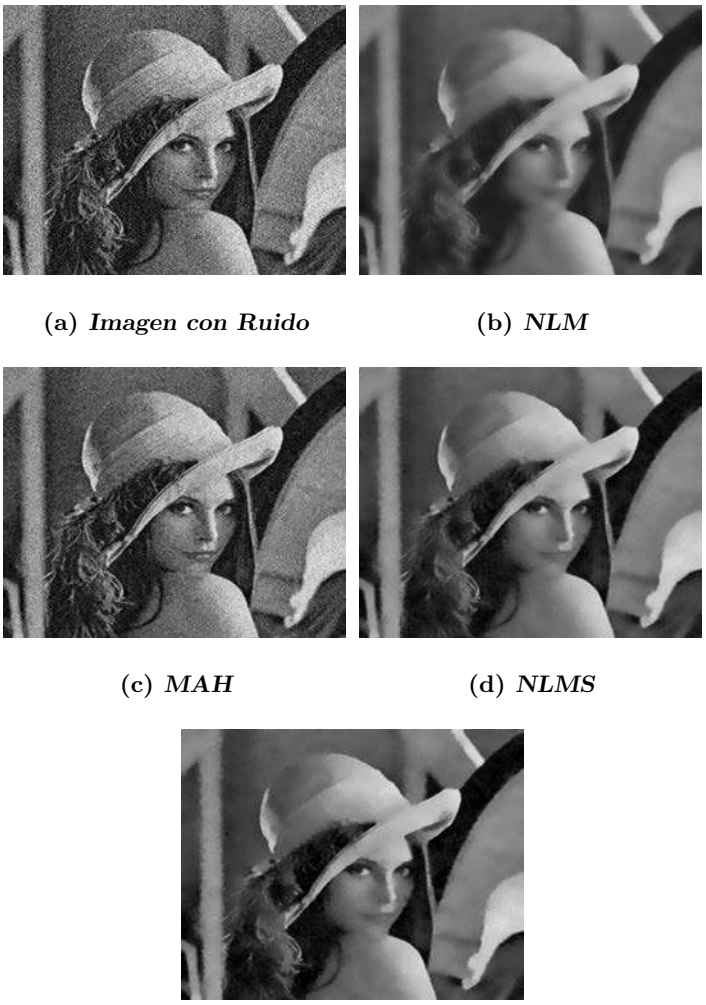
Figura 3.7: Restauración de imagen sintética con ruido gaussiano $\sigma_\eta = 60$.

(a) ***Imagen con Ruido*** **(b)** ***NLM***

(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.8: Restauración de imagen Lena con ruido gaussiano $\sigma_\eta = 20$.

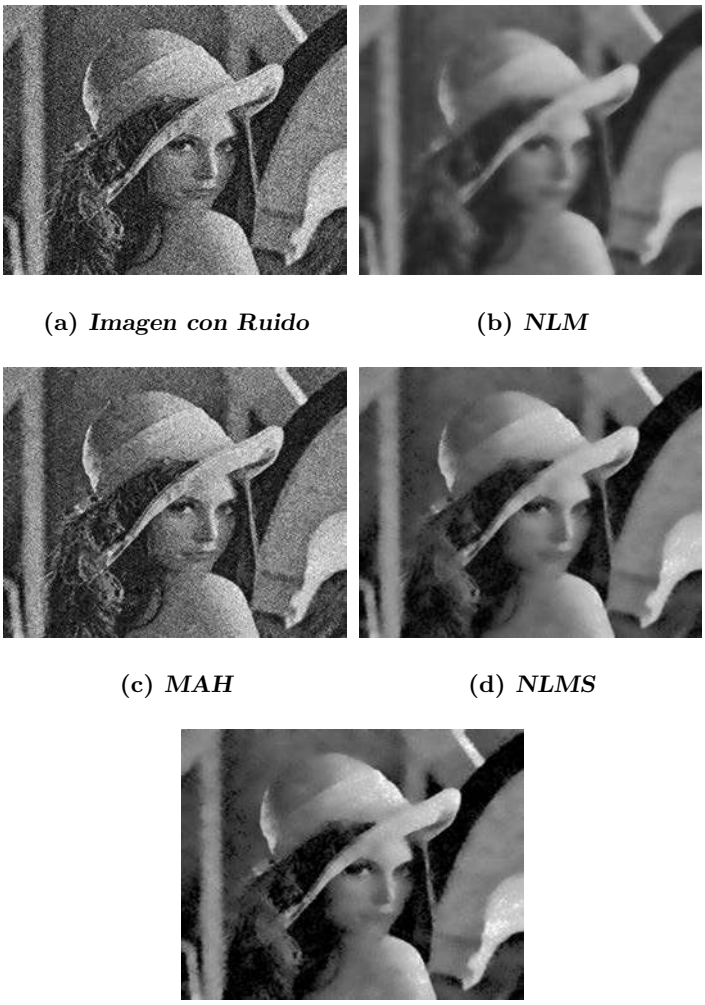


(a) ***Imagen con Ruido*** **(b)** ***NLM***

(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.9: Restauración de imagen Lena con ruido gaussiano $\sigma_\eta = 40$.

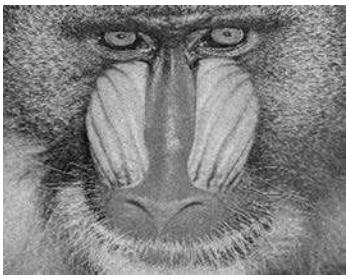


(a) ***Imagen con Ruido*** **(b)** ***NLM***

(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.10: Restauración de imagen Lena con ruido gaussiano $\sigma_\eta = 60$.



(a) ***Imagen con Ruido***

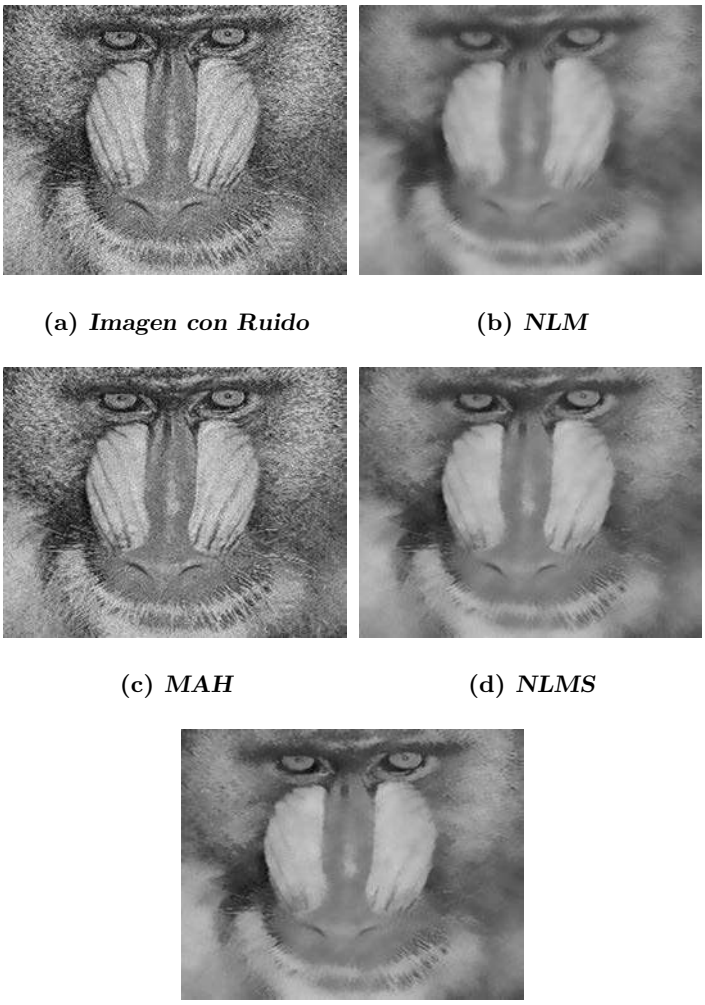
(b) ***NLM***

(c) ***MAH***

(d) ***NLMS***

(e) ***NLMR***

Figura 3.11: Restauración de imagen Mandril con ruido gaussiano $\sigma_\eta = 20$.

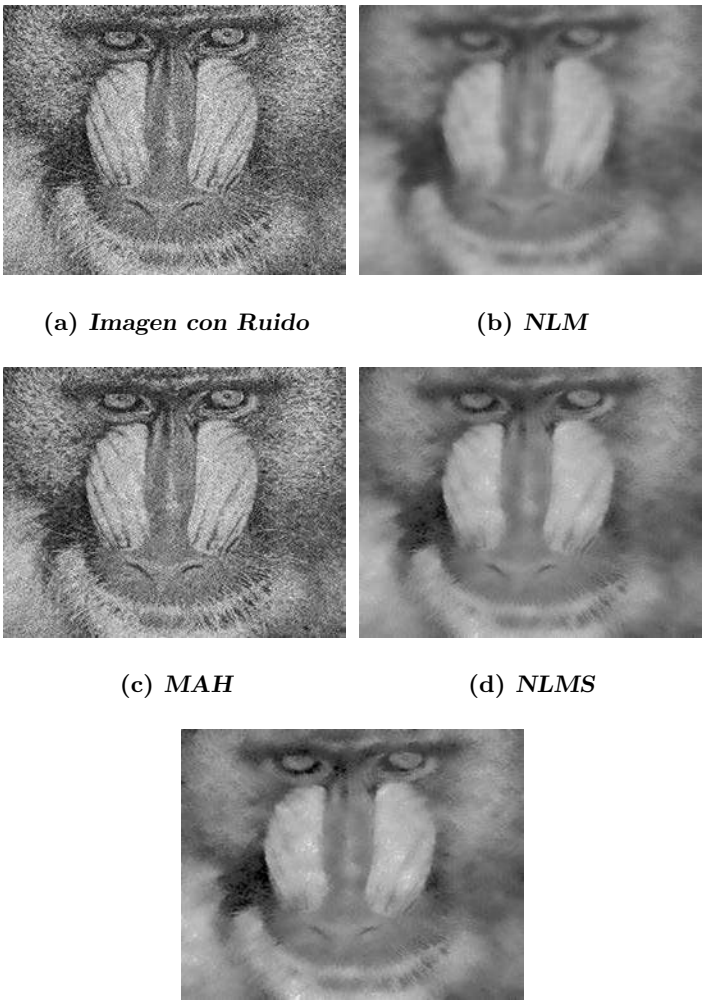


(a) ***Imagen con Ruido*** **(b)** ***NLM***

(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.12: Restauración de imagen Mandril con ruido gaussiano $\sigma_\eta = 40$.

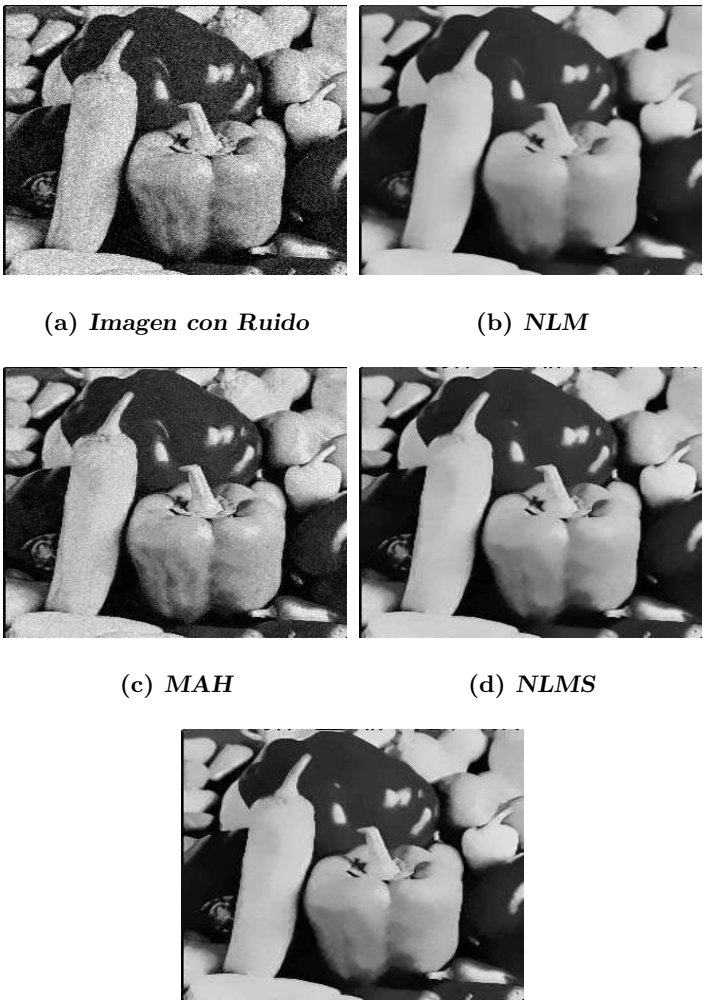


(a) ***Imagen con Ruido*** **(b)** ***NLM***

(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.13: Restauración de imagen Mandril con ruido gaussiano $\sigma_\eta = 60$.

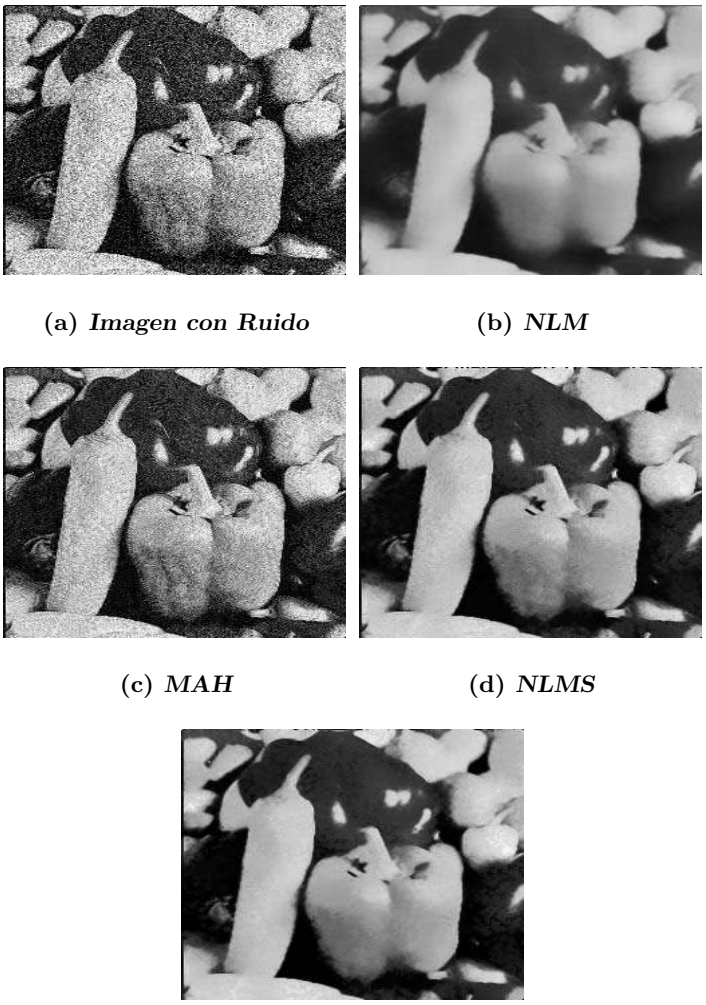


(a) ***Imagen con Ruido*** **(b)** ***NLM***

(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.14: Restauración de imagen Peppers con ruido gaussiano $\sigma_\eta = 20$.

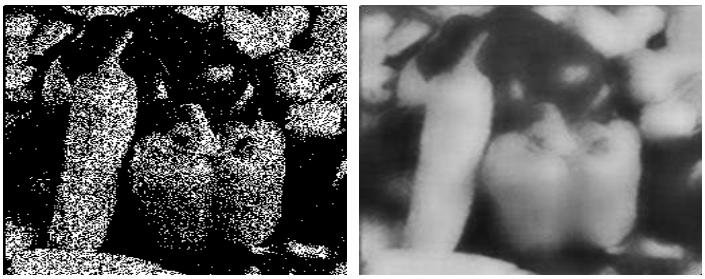


(a) ***Imagen con Ruido*** **(b)** ***NLM***

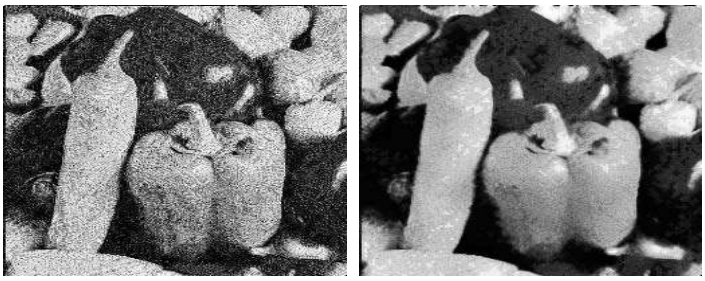
(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.15: Restauración de imagen Peppers con ruido gaussiano $\sigma_\eta = 40$.



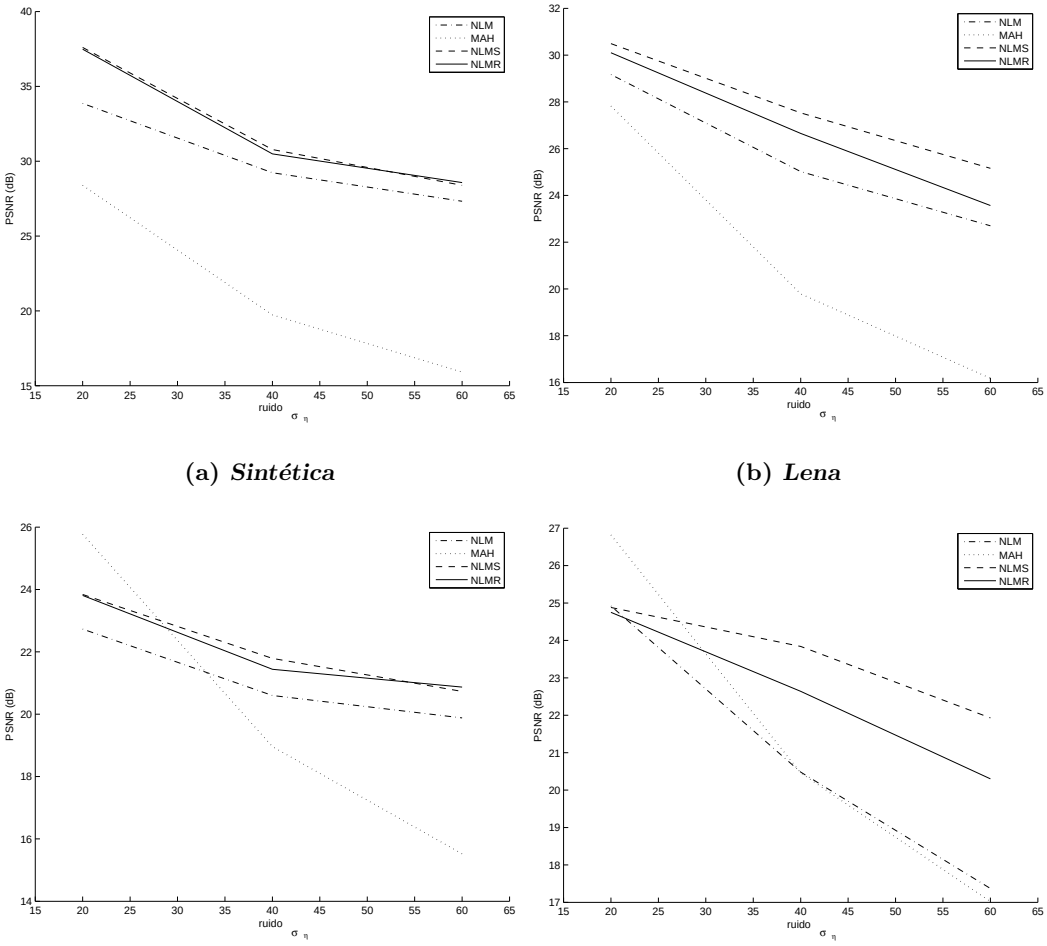
(a) ***Imagen con Ruido*** **(b)** ***NLM***



(c) ***MAH*** **(d)** ***NLMS***

(e) ***NLMR***

Figura 3.16: Restauración de imagen Peppers con ruido gaussiano $\sigma_\eta = 60$.



(a) *Sintética* **(b) *Lena***

(c) *Mandril* **(d) *Peppers***

Figura 3.17: Comparación de los resultados de PSNR para imágenes alteradas con ruido gaussiano.

Capítulo 4

Reducción de ruido sal y pimienta

Un impulso es un pico estrecho y alto con área finita, comúnmente igual a 1 [Ambardar02]. En señales discretas un impulso unitario es una muestra unitaria definida como

$$u(r) = \begin{cases} 0, & r \neq 0 \\ 1, & r = 0 \end{cases} \tag{4.1}$$

En el área de procesamiento de imágenes, el ruido impulsivo es aquel que modifica solamente algunos pixeles con un cierto valor de intensidad, dejando los pixeles restantes inalterados. De esta manera, un impulso en una imagen digital también puede ser definido como aquella observación que es inconsistente con los datos pertenecientes a su región o vecindario.
Las técnicas que utilizan borroneo son poco eficaces para la eliminación de ruido impulsivo y, como alternativa para abordar este problema, usualmente se emplean metodologías basadas en filtros no lineales. La gran mayoría de los algoritmos orientados a la eliminación de ruido impulsivo se encuentran basados en el filtrado de mediana debido a su robustez y a su capacidad de preservar bordes. Sin embargo, cuando los niveles de ruido son altos la imagen restaurada no presenta la calidad deseada, por lo que es necesario algún tipo de refinamiento a la misma.

En el presente capítulo se presenta el algoritmo propuesto en [C.A. Júnez10a] y [C.A. Júnez10b], que remueve ruido sal y pimienta en imágenes con diferentes niveles de ruido.

4.1. Filtrado

El algoritmo que se propone consiste de dos fases. Como primer paso, el algoritmo obtiene una estimación I_1 de la imagen restaurada usando un filtrado de mediana adaptable, debido a que el tamaño de la ventana usada en el filtrado se ajusta a la información que se obtiene, concepto introducido en [H. Hwang95]. Lo anterior puede reducir el tiempo de ejecución y mejorar la calidad de restauración.

Se sabe que, de acuerdo con el modelo definido en la ecuación (2.2), los pixeles ruidosos toman sus valores del conjunto $\{I_{min}, I_{max}\}$. Entonces, el filtrado de mediana adaptable se aplicará unicamente a los pixeles en las posiciones del conjunto

$$\Gamma = \{(x, y) | I(x, y) = I_{min} \; o \; I(x, y) = I_{max}\} \tag{4.2}$$

i.e. Γ es el conjunto de los candidatos a ser pixeles ruidosos. Así, también se define a Γ^c como el conjunto de los pixeles no considerados como ruido, de esta manera los valores de intensidad de estos pixeles permanecen inalterados.

Comenzando con un tamaño de ventana mínimo $W = W_{min}$, se define un vecindario $N(x, y)$ para el pixel ruidoso localizado en la posición (x, y), posteriormente, todos los pixeles pertenecientes a este vecindario y considerados como no alterados son almacenados en un vector $\mathbf{V_{\Gamma^c}}$. Si este vector contiene al menos un elemento, entonces el pixel en la posición (x, y) es reemplazado por el valor de la mediana de los elementos del vector, obteniéndose el valor de intensidad estimado $I_1(x, y)$. Si el vector $\mathbf{V_{\Gamma^c}}$ está vacío, el tamaño de la ventana W se incrementa, repitiéndose el procedimiento antes descrito. Con este tipo de filtrado se considera la cercanía geométrica para la restauración, es decir, que es más probable que el valor de intensidad real del pixel ruidoso, sea parecido a alguno de los pixeles más cercanos.

Si el tamaño se incrementa hasta un tamaño de ventana máximo $W = W_{max}$ y el vector $\mathbf{V_{\Gamma^c}}$ continúa vacío, es posible que los pixeles del vecindario pertenezcan a una región cuyo valor de intensidad sea igual al valor máximo o al mínimo en la imagen y no sea ruido. En este caso, si el número de pixeles en el vecindario, con intensidad igual al máximo (mínimo) es mayor que el número de pixeles con valor de intensidad mínimo (máximo), entonces el valor de intensidad del pixel central con posición (x, y) es reemplazado por el

valor de intensidad máximo (mínimo). El algoritmo 1 muestra este proceso para el pixel en la posicion (x, y).

La segunda fase consiste en aplicar un filtrado de mediana ponderada [Brownrigg84] a la estimación I_1. Este filtrado se aplica solamente a los pixeles del vecindario perteneciente a la imagen I_1 y con centro en un pixel cuya posición (x, y) pertenece al conjunto Γ. El filtrado mediana ponderada se lleva a cabo usando como pesos los elementos de una máscara gaussiana $\mathbf{G}$, así, se construye un vector V agregando a éste $G(i, j)$ veces el elemento en la posición (i, j). El algoritmo 2 muestra este proceso para el pixel en la posicion (x, y).

4.2. Experimentos

La implementación de los algoritmos propuestos se aplicó en las imágenes que se muestran en las Figuras 3.2, 3.3 y 3.4, las cuales fueron alteradas con ruido sal y pimienta con diferentes densidades. Los algoritmos que se comparan son el Filtro Mediana Simple (FMS) [J. Astola97], el Filtro Mediana Adaptable (FMA) [H. Hwang95], el Filtro Mediana Ponderada (WMF) [Brownrigg84], el Filtro creciente de Yuan (FYU) [S.Q. Yuan06] la primera fase del algoritmo propuesto en este capítulo, al cual se le denominará Filtro Mediana de Ventana Creciente (FMC) [C.A. Júnez10a], y un posterior refinamiento al resultado obtenido con este algoritmo, mediante un Filtro Mediana Ponderado (FMCW). Todos los algoritmos mencionados fueron ejecutados con Matlab R2007b en una PC con procesador Intel Core2 Duo, 2.20 Ghz.

El tamaño de ventana empleado para FMS, FMA y la primera fase de FYU fue de 5×5, 11×11 y 21×21, para ruido con $\delta = 20\,\%$, $\delta = 50\,\%$ y $\delta = 80\,\%$, respectivamente. En lo que respecta al WMF, los pesos están dados por una máscara con valor de 4 en el pixel central y 1 en los restantes. Para FMC y la segunda fase de FYU y FMCW, éstos inician con un tamaño de ventana de 3×3, considerando un tamaño de ventana máximo de 21×21. Para el refinamiento con el algoritmo FMCW, también se considera una máscara gaussiana de 7×7, con desviación estándar $\sigma_F = 1.5$ para la definición de los pesos.

En las Tablas 4.1-4.18, se comparan los valores de PSNR, MAE, UIQI y el tiempo

Algoritmo 1 Cálculo de estimación del valor de intensidad de un pixel

```
ESTIMACION(I, x, y, I_min, I_max, W_min, W_max)
 1  if (I(x,y) = I_min) or (I(x,y) = I_max)
 2     W ← W_min
 3     testigo ← 0
 4     do
 5        V_Γc ← {}
 6        m ← (W − 1)/2
 7        p ← −m
 8        do
 9           q ← −m
10           do
11              if (I(x + p, y + q) ≠ I_min) and (I(x + p, y + q) ≠ I_max)
12                 V_Γc ← V_Γc ∪ I(x + p, y + q)
13              q ← q + 1
14           while q ≤ m
15           p ← p + 1
16        while p ≤ m
17        if V_Γc ≠ {}
18           I_1(x, y) ← ~V_Γc~
19           testigo ← 1
20        else
21           W ← W + 2
22     while (W ≤ W_max) and (testigo = 0)
23     if testigo = 0 and W > W_max
24        if ContarI_min > ContarI_max
25           I_1(x, y) ← I_min
26        else
27           I_1(x, y) ← I_max
28  else
29     I_1(x, y) ← I(x, y)
30  regresar I_1(x, y)
```

Algoritmo 2 Filtro de mediana ponderada

MEDPOND$(I, I_1, x, y, I_{min}, I_{max}, G, W)$

1 $if\ \ (I(x,y) = I_{min})\ \ or\ \ (I(x,y) = I_{max})$
2 $\quad V \leftarrow \{\}$
3 $\quad m \leftarrow (W-1)/2$
4 $\quad p \leftarrow -m$
5 $\quad do$
6 $\quad\quad q \leftarrow -m$
7 $\quad\quad do$
8 $\quad\quad\quad r \leftarrow 1$
9 $\quad\quad\quad do$
10 $\quad\quad\quad\quad V \leftarrow V \cup I_1(x+p, y+q)$
11 $\quad\quad\quad\quad r \leftarrow r+1$
12 $\quad\quad\quad while\ \ r \leq G(p,q)$
13 $\quad\quad\quad q \leftarrow q+1$
14 $\quad\quad while\ \ q \leq m$
15 $\quad\quad p \leftarrow p+1$
16 $\quad while\ \ p \leq m$
17 $\quad I_2(x,y) \leftarrow \widetilde{V}$
18 $else$
19 $\quad I_2(x,y) \leftarrow I(x,y)$
20 **regresar** $I_2(x,y)$

de ejecución. En éstas se puede observar que se obtienen los mejores resultados, para la imagen sintética, con WMF, FMC y FMCW. Para las imágenes reales, los resultados son muy similares para los algoritmos FMA, WMF, FYU, FMC y FMCW, si el ruido es de baja densidad ($\delta = 20\,\%$) o de densidad media ($\delta = 50\,\%$). En lo que respecta a imágenes con ruido de alta densidad ($\delta = 80\,\%$), los algoritmos FMC y FMCW presentan los mejores resultados de restauración, además de obtenerlos en los tiempos más cortos.

Las Figuras 4.1-4.12 presentan los resultados experimentales para cuatro imágenes (Sintetica, Lena, Mandril y Peppers) alteradas con diferentes niveles de ruido. Se puede observar en las imágenes una calidad de restauración muy similar entre FMA, WMF, FYU, FMC y FMCW para imágenes con ruido de baja densidad ($\delta = 20\,\%$). En las imágenes con ruido de densidad media ($\delta = 50\,\%$), se puede notar que no todo el ruido es eliminado para WMF y FYU, lo cual no sucede con FMA, FMC y FMCW. En lo referente al ruido de alta densidad ($\delta = 80\,\%$), se puede observar la aparición de grupos de pixeles con los valores máximos y mínimos con el algoritmo WMF; en el caso de FYU, se puede notar que el problema de la supresión de ruido se presenta principalmente en los bordes; los algoritmos FMA, FMC y FMCW son muy efectivos para suprimir el ruido, siendo los dos últimos los que mejor preservan los bordes.

La Figura 4.13 muestra una comparación entre los resultados obtenidos con los seis algoritmos para las mismas cuatro imágenes. Los algoritmos WMF, FMC y FMCW (excepto en la imagen peppers, tanto FMC como FMCW) brindan los mejores resultados en niveles bajos y medio de ruido. En niveles altos los algoritmos FMC y FMCW son los que mejor desempeño tienen.

Tabla 4.1: Resultados FMS, ruido SP $\delta = 20\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	35.44	0	0.99	3
Lena	32.04	3	0.79	11
Mandril	23.26	10	0.63	11
Peppers	22.34	8	0.68	3
Bee	30.22	3	0.56	11
Butterfly	26.15	6	0.67	11
Cat	27.98	5	0.65	11
Duck1	24.84	8	0.52	11
Duck2	28.43	4	0.68	11
Fish	23.51	12	0.58	11
Owl	25.15	9	0.64	11
Bike	18.03	12	0.38	11
Plane	26.27	5	0.64	11
Boat	26.48	7	0.55	11
Lake1	24.85	9	0.40	11
Lake2	25.42	7	0.58	11
Land1	16.81	27	0.33	11
Land2	23.41	9	0.63	11
Cameraman	25.29	5	0.64	11
Actor	25.93	7	0.58	11
Barbara	22.53	10	0.53	11
Elaine	30.67	5	0.57	11
Tvset	26.15	7	0.58	11
Village	27.61	6	0.60	11
House1	24.53	7	0.59	11
House2	27.99	4	0.43	3
House3	25.39	8	0.41	11
Moon	28.67	6	0.39	3
Partenon	25.30	6	0.50	11
Pentagon	26.45	7	0.49	44

Tabla 4.2: Resultados FMS, ruido SP $\delta = 50\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	31.06	0	0.97	8
Lena	25.94	7	0.48	32
Mandril	20.24	15	0.21	32
Peppers	18.29	14	0.46	8
Bee	27.14	5	0.40	32
Butterfly	22.57	9	0.45	32
Cat	24.65	8	0.42	32
Duck1	21.48	12	0.22	32
Duck2	24.70	7	0.35	32
Fish	19.63	18	0.34	32
Owl	21.77	13	0.30	32
Bike	15.67	18	0.19	32
Plane	22.12	8	0.34	32
Boat	22.69	11	0.26	32
Lake1	22.40	12	0.15	32
Lake2	21.32	11	0.32	32
Land1	15.20	33	0.08	32
Land2	19.64	16	0.24	32
Cameraman	21.59	9	0.27	32
Actor	22.42	11	0.30	32
Barbara	21.82	12	0.39	32
Elaine	27.47	7	0.44	32
Tvset	22.75	11	0.27	32
Village	24.31	9	0.32	32
House1	21.18	11	0.30	32
House2	23.44	7	0.25	8
House3	23.38	10	0.14	32
Moon	26.78	7	0.23	8
Partenon	22.07	9	0.26	32
Pentagon	24.26	9	0.27	128

Tabla 4.3: Resultados FMS, ruido SP $\delta = 80\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	26.53	1	0.93	27
Lena	21.61	11	0.27	108
Mandril	18.99	18	0.08	108
Peppers	15.37	23	0.22	26
Bee	24.13	8	0.24	108
Butterfly	20.83	14	0.24	108
Cat	22.67	10	0.31	109
Duck1	19.91	15	0.11	108
Duck2	22.40	10	0.15	108
Fish	17.03	24	0.19	108
Owl	19.38	18	0.10	108
Bike	14.13	24	0.08	108
Plane	19.84	13	0.14	108
Boat	20.45	14	0.12	108
Lake1	20.91	14	0.07	107
Lake2	19.09	16	0.13	108
Land1	14.31	37	0.02	108
Land2	17.58	21	0.06	108
Cameraman	19.40	12	0.10	108
Actor	19.62	16	0.12	108
Barbara	19.79	17	0.22	108
Elaine	23.29	11	0.26	108
Tvset	20.44	15	0.11	108
Village	22.20	13	0.14	108
House1	18.68	15	0.15	108
House2	19.73	11	0.10	26
House3	23.04	12	0.06	108
Moon	24.22	10	0.11	27
Partenon	22.79	12	0.12	108
Pentagon	22.38	13	0.11	436

Tabla 4.4: Resultados FMA, ruido SP $\delta = 20\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	40.01	0	0.99	1
Lena	37.87	1	0.96	3
Mandril	29.78	2	0.94	3
Peppers	25.11	3	0.94	1
Bee	36.73	1	0.93	3
Butterfly	32.46	1	0.94	3
Cat	34.32	1	0.94	3
Duck1	30.81	2	0.92	3
Duck2	34.74	1	0.94	3
Fish	29.40	3	0.93	5
Owl	31.59	2	0.94	3
Bike	24.45	3	0.90	3
Plane	32.61	1	0.94	3
Boat	32.70	2	0.93	3
Lake1	31.27	2	0.91	3
Lake2	31.78	2	0.93	3
Land1	21.26	8	0.86	3
Land2	29.16	2	0.94	3
Cameraman	31.25	1	0.94	3
Actor	32.13	2	0.93	3
Barbara	28.92	2	0.93	3
Elaine	36.96	1	0.93	3
Tvset	32.57	2	0.93	3
Village	33.92	1	0.93	3
House1	30.90	2	0.93	3
House2	34.18	1	0.91	1
House3	31.94	2	0.91	3
Moon	35.18	1	0.91	1
Partenon	31.71	1	0.92	3
Pentagon	32.92	2	0.92	11

Tabla 4.5: Resultados FMA, ruido SP $\delta = 50\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	33.75	0	0.98	4
Lena	28.74	3	0.78	17
Mandril	23.14	8	0.70	17
Peppers	19.98	8	0.75	4
Bee	29.98	3	0.75	16
Butterfly	25.10	5	0.77	17
Cat	27.59	4	0.77	17
Duck1	24.26	6	0.68	17
Duck2	27.62	4	0.73	17
Fish	22.21	10	0.72	20
Owl	24.71	7	0.72	17
Bike	18.53	9	0.67	17
Plane	24.94	4	0.73	17
Boat	25.64	5	0.71	17
Lake1	25.32	6	0.67	17
Lake2	24.19	6	0.73	17
Land1	17.13	19	0.60	17
Land2	22.46	8	0.70	17
Cameraman	24.54	4	0.71	17
Actor	25.24	6	0.71	17
Barbara	24.70	6	0.75	17
Elaine	30.29	3	0.77	17
Tvset	25.68	6	0.71	17
Village	27.17	5	0.73	17
House1	24.07	6	0.72	17
House2	26.37	4	0.70	4
House3	26.30	5	0.68	17
Moon	29.75	4	0.71	4
Partenon	24.97	4	0.70	17
Pentagon	27.21	5	0.72	67

Tabla 4.6: Resultados FMA, ruido SP $\delta = 80\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	27.42	1	0.94	21
Lena	22.52	9	0.44	88
Mandril	19.94	15	0.34	282
Peppers	16.08	19	0.39	22
Bee	25.05	6	0.42	88
Butterfly	21.75	11	0.42	88
Cat	23.60	8	0.50	88
Duck1	20.83	12	0.35	88
Duck2	23.38	8	0.37	88
Fish	17.83	20	0.39	94
Owl	20.32	15	0.35	87
Bike	15.07	19	0.32	87
Plane	20.75	10	0.36	87
Boat	21.39	12	0.36	87
Lake1	21.85	11	0.33	87
Lake2	19.41	13	0.36	87
Land1	14.96	31	0.27	88
Land2	18.51	17	0.32	87
Cameraman	20.34	10	0.34	88
Actor	20.53	13	0.34	88
Barbara	20.73	14	0.41	88
Elaine	24.24	9	0.44	88
Tvset	21.37	12	0.35	87
Village	22.43	10	0.37	88
House1	19.61	12	0.37	88
House2	20.62	9	0.34	22
House3	23.99	10	0.33	88
Moon	25.16	8	0.36	22
Partenon	23.73	9	0.35	88
Pentagon	23.33	10	0.35	353

Tabla 4.7: Resultados WMF, ruido SP $\delta = 20\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	45.57	0	1.00	3
Lena	38.35	1	0.96	11
Mandril	30.27	2	0.95	11
Peppers	28.97	2	0.94	3
Bee	37.16	1	0.93	11
Butterfly	32.85	1	0.94	11
Cat	34.81	1	0.95	11
Duck1	31.15	2	0.92	12
Duck2	35.09	1	0.95	11
Fish	29.79	3	0.93	19
Owl	31.90	2	0.94	11
Bike	24.79	3	0.90	11
Plane	34.28	1	0.95	11
Boat	33.38	1	0.93	11
Lake1	31.99	2	0.92	11
Lake2	32.69	1	0.94	11
Land1	21.33	8	0.86	13
Land2	30.13	2	0.94	11
Cameraman	33.05	1	0.95	11
Actor	32.70	2	0.93	13
Barbara	29.97	2	0.94	11
Elaine	37.20	1	0.93	11
Tvset	33.48	1	0.94	11
Village	34.66	1	0.94	11
House1	32.65	1	0.94	11
House2	35.75	1	0.92	3
House3	32.39	2	0.92	11
Moon	35.61	1	0.91	3
Partenon	32.52	1	0.93	11
Pentagon	33.62	1	0.93	44

Tabla 4.8: Resultados WMF, ruido SP $\delta = 50\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	31.67	0	0.93	7
Lena	28.93	3	0.83	26
Mandril	25.19	6	0.79	27
Peppers	23.43	5	0.79	7
Bee	29.31	2	0.75	27
Butterfly	27.51	4	0.79	26
Cat	27.51	3	0.79	27
Duck1	25.32	5	0.74	27
Duck2	29.66	3	0.80	26
Fish	23.82	8	0.77	32
Owl	25.86	5	0.79	26
Bike	19.68	8	0.69	26
Plane	27.03	3	0.78	26
Boat	26.75	4	0.76	26
Lake1	25.95	5	0.72	26
Lake2	26.22	5	0.77	26
Land1	18.00	17	0.67	28
Land2	23.87	6	0.78	27
Cameraman	25.77	3	0.78	26
Actor	25.86	5	0.76	27
Barbara	24.43	6	0.76	26
Elaine	29.11	3	0.76	26
Tvset	26.52	4	0.77	26
Village	27.80	4	0.78	26
House1	26.21	4	0.77	26
House2	27.54	3	0.71	7
House3	27.15	5	0.72	27
Moon	28.73	4	0.71	7
Partenon	28.04	3	0.74	27
Pentagon	27.37	4	0.74	104

Tabla 4.9: Resultados WMF, ruido SP $\delta = 80\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	12.38	13	0.02	10
Lena	11.83	24	0.10	42
Mandril	12.03	27	0.13	42
Peppers	11.08	28	0.18	11
Bee	12.08	22	0.07	41
Butterfly	12.23	24	0.11	41
Cat	11.27	28	0.09	42
Duck1	11.78	27	0.11	42
Duck2	12.25	25	0.07	41
Fish	10.68	39	0.15	44
Owl	11.72	30	0.14	42
Bike	10.22	23	0.17	42
Plane	11.65	16	0.10	42
Boat	12.10	23	0.10	41
Lake1	11.40	29	0.10	41
Lake2	11.61	24	0.13	42
Land1	10.52	41	0.18	42
Land2	11.43	20	0.16	42
Cameraman	11.68	23	0.08	41
Actor	11.37	29	0.13	42
Barbara	11.65	27	0.11	42
Elaine	12.36	20	0.08	42
Tvset	12.14	24	0.11	42
Village	12.09	24	0.09	42
House1	11.73	20	0.12	42
House2	12.08	20	0.08	10
House3	11.95	29	0.07	43
Moon	12.63	21	0.06	11
Partenon	12.12	25	0.08	42
Pentagon	12.41	22	0.09	166

Tabla 4.10: Resultados FYU, ruido SP $\delta = 20\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	37.82	0	0.99	6
Lena	37.39	1	0.97	24
Mandril	31.48	2	0.96	24
Peppers	23.56	3	0.92	6
Bee	36.58	1	0.93	24
Butterfly	31.40	1	0.94	24
Cat	34.15	1	0.94	24
Duck1	28.68	2	0.91	25
Duck2	35.92	1	0.95	24
Fish	27.79	4	0.91	24
Owl	30.96	2	0.93	25
Bike	18.88	8	0.72	25
Plane	30.47	1	0.95	24
Boat	31.03	2	0.92	24
Lake1	29.54	2	0.88	24
Lake2	30.37	2	0.93	24
Land1	18.78	13	0.72	26
Land2	27.29	3	0.93	25
Cameraman	26.26	2	0.94	24
Actor	29.88	2	0.91	25
Barbara	27.76	3	0.92	25
Elaine	36.92	1	0.93	24
Tvset	30.86	2	0.93	24
Village	34.78	1	0.95	24
House1	28.14	2	0.94	24
House2	33.09	1	0.91	6
House3	32.46	2	0.91	24
Moon	34.35	1	0.91	6
Partenon	32.11	1	0.93	24
Pentagon	32.31	2	0.92	97

Tabla 4.11: Resultados FYU, ruido SP $\delta = 50\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	36.49	0	0.98	12
Lena	28.07	2	0.87	47
Mandril	27.69	5	0.89	47
Peppers	19.42	8	0.74	12
Bee	30.74	2	0.79	47
Butterfly	28.23	3	0.83	47
Cat	29.48	3	0.85	47
Duck1	25.76	4	0.78	47
Duck2	32.37	2	0.87	47
Fish	23.17	7	0.78	46
Owl	27.09	5	0.84	47
Bike	16.87	11	0.59	47
Plane	25.77	3	0.82	47
Boat	26.28	4	0.78	47
Lake1	26.58	5	0.76	47
Lake2	24.68	4	0.78	47
Land1	17.48	18	0.64	47
Land2	23.55	6	0.83	47
Cameraman	22.86	3	0.81	47
Actor	25.26	4	0.77	47
Barbara	24.11	6	0.77	47
Elaine	28.92	3	0.78	47
Tvset	26.89	4	0.81	47
Village	28.98	3	0.83	47
House1	25.32	4	0.81	47
House2	27.42	3	0.73	12
House3	29.08	4	0.79	47
Moon	30.85	3	0.76	11
Partenon	30.28	3	0.80	47
Pentagon	28.83	4	0.79	190

Tabla 4.12: Resultados FYU, ruido SP $\delta = 80\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	21.46	3	0.86	35
Lena	15.03	14	0.40	138
Mandril	15.78	16	0.44	140
Peppers	12.16	29	0.27	33
Bee	17.61	9	0.41	140
Butterfly	15.12	14	0.38	139
Cat	18.05	9	0.50	139
Duck1	15.74	14	0.38	139
Duck2	21.28	6	0.56	142
Fish	13.47	27	0.34	136
Owl	15.42	17	0.35	138
Bike	12.41	21	0.26	137
Plane	15.10	13	0.41	139
Boat	15.88	14	0.38	139
Lake1	16.46	13	0.31	139
Lake2	14.36	17	0.32	137
Land1	11.10	37	0.22	138
Land2	14.44	19	0.40	138
Cameraman	15.46	13	0.45	140
Actor	14.86	18	0.32	140
Barbara	13.59	23	0.29	137
Elaine	16.63	12	0.38	139
Tvset	15.47	15	0.39	139
Village	16.76	12	0.40	140
House1	15.02	16	0.38	138
House2	15.32	12	0.30	34
House3	18.92	10	0.38	140
Moon	21.48	7	0.43	35
Partenon	17.73	9	0.38	140
Pentagon	17.26	12	0.40	569

Tabla 4.13: Resultados FMC, ruido SP $\delta = 20\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	45.29	0	1.00	1
Lena	41.49	1	0.97	3
Mandril	33.54	1	0.97	3
Peppers	21.92	4	0.91	1
Bee	38.07	1	0.94	3
Butterfly	34.49	1	0.95	3
Cat	36.45	1	0.96	3
Duck1	30.57	2	0.92	4
Duck2	36.73	1	0.96	3
Fish	26.34	4	0.85	28
Owl	33.39	2	0.96	3
Bike	26.09	2	0.91	3
Plane	34.97	1	0.96	3
Boat	35.26	1	0.94	3
Lake1	33.73	1	0.94	3
Lake2	34.07	1	0.94	3
Land1	20.80	9	0.86	4
Land2	30.40	2	0.96	3
Cameraman	34.61	1	0.96	3
Actor	33.66	1	0.92	4
Barbara	30.11	2	0.94	3
Elaine	37.66	1	0.93	3
Tvset	35.24	1	0.95	3
Village	35.78	1	0.95	3
House1	33.23	1	0.95	3
House2	36.43	1	0.92	1
House3	33.77	1	0.94	3
Moon	36.48	1	0.92	1
Partenon	34.38	1	0.94	3
Pentagon	34.60	1	0.94	13

Tabla 4.14: Resultados FMC, ruido SP $\delta = 50\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	37.04	0	0.98	2
Lena	34.86	2	0.91	7
Mandril	27.85	4	0.91	7
Peppers	20.71	7	0.81	2
Bee	32.87	2	0.81	7
Butterfly	29.17	3	0.85	7
Cat	31.24	3	0.87	7
Duck1	27.00	4	0.80	8
Duck2	31.59	2	0.88	7
Fish	23.28	8	0.74	31
Owl	28.17	4	0.86	7
Bike	21.12	7	0.74	7
Plane	29.55	2	0.86	7
Boat	29.70	3	0.81	7
Lake1	28.50	4	0.80	7
Lake2	28.66	4	0.83	7
Land1	18.24	17	0.69	7
Land2	26.49	5	0.88	7
Cameraman	28.49	2	0.86	7
Actor	28.63	4	0.81	7
Barbara	25.35	5	0.82	7
Elaine	32.59	3	0.80	7
Tvset	29.38	4	0.84	7
Village	30.48	3	0.84	7
House1	28.02	3	0.85	7
House2	30.64	2	0.75	2
House3	28.82	4	0.80	7
Moon	31.83	3	0.77	2
Partenon	28.74	3	0.82	7
Pentagon	29.45	4	0.80	28

Tabla 4.15: Resultados FMC, ruido SP $\delta = 80\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	32.24	0	0.97	3
Lena	29.42	4	0.78	12
Mandril	23.17	9	0.73	12
Peppers	18.58	11	0.63	3
Bee	29.00	4	0.62	12
Butterfly	24.89	6	0.68	12
Cat	27.15	5	0.71	12
Duck1	23.45	9	0.58	13
Duck2	27.61	4	0.71	12
Fish	20.71	14	0.55	36
Owl	24.15	9	0.68	12
Bike	17.63	13	0.49	12
Plane	25.26	5	0.68	12
Boat	25.36	7	0.60	12
Lake1	24.45	8	0.55	12
Lake2	24.49	7	0.63	12
Land1	15.66	29	0.41	12
Land2	22.58	10	0.69	12
Cameraman	24.15	5	0.68	12
Actor	24.46	8	0.61	12
Barbara	22.14	10	0.61	12
Elaine	28.70	6	0.59	12
Tvset	25.19	7	0.64	12
Village	26.49	6	0.65	12
House1	23.94	7	0.66	12
House2	26.02	5	0.50	3
House3	24.96	8	0.55	12
Moon	27.81	6	0.51	3
Partenon	24.50	6	0.60	12
Pentagon	25.60	7	0.56	47

Tabla 4.16: Resultados FMCW, ruido SP $\delta = 20\,\%$

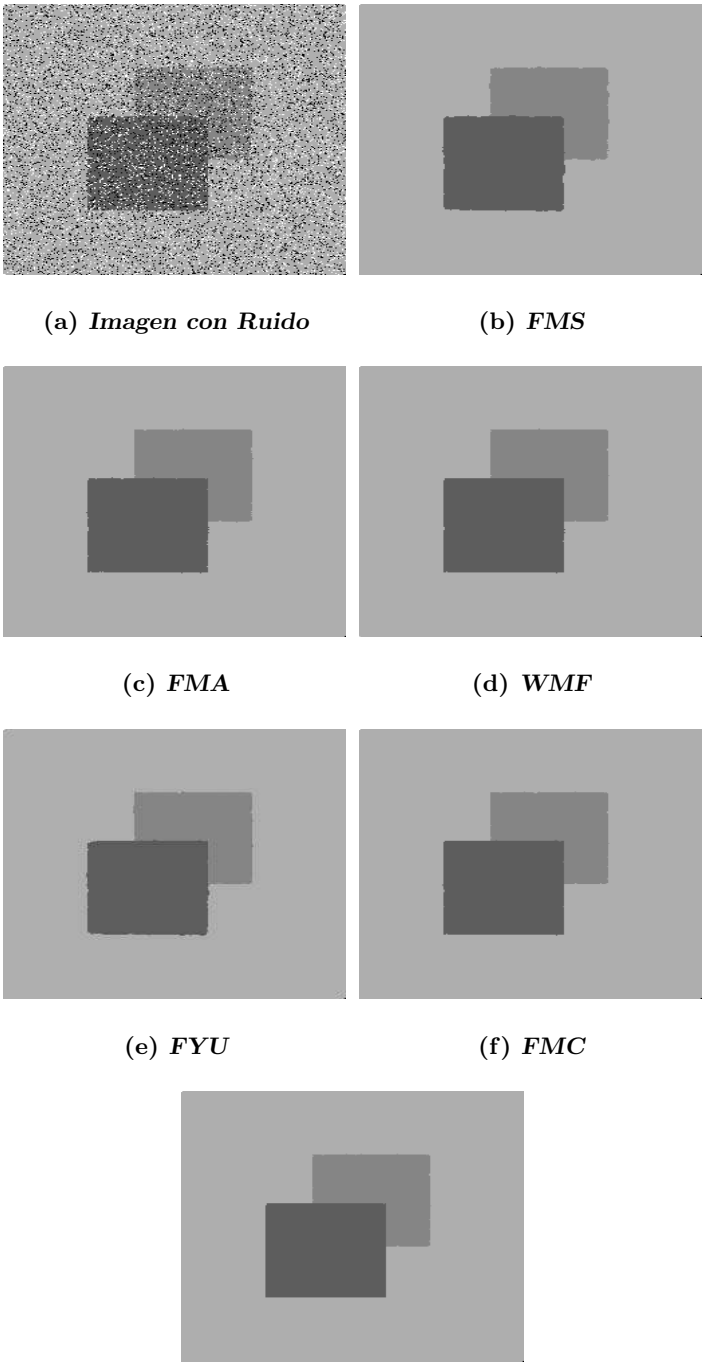
Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	48.03	0	1.00	1
Lena	40.13	1	0.97	5
Mandril	31.22	2	0.96	7
Peppers	21.29	4	0.90	2
Bee	37.70	1	0.94	5
Butterfly	33.38	1	0.95	5
Cat	35.75	1	0.96	5
Duck1	29.90	2	0.92	7
Duck2	35.95	1	0.96	5
Fish	25.71	5	0.84	45
Owl	32.56	2	0.95	7
Bike	24.57	3	0.90	6
Plane	34.14	1	0.96	5
Boat	34.02	1	0.94	5
Lake1	32.54	2	0.93	6
Lake2	33.00	1	0.94	6
Land1	19.73	10	0.83	13
Land2	29.26	2	0.95	7
Cameraman	33.81	1	0.96	5
Actor	32.78	2	0.92	6
Barbara	30.31	2	0.94	7
Elaine	37.56	1	0.93	4
Tvset	34.10	1	0.95	6
Village	35.35	1	0.95	5
House1	32.72	1	0.95	6
House2	35.91	1	0.92	1
House3	33.13	1	0.93	6
Moon	35.85	1	0.92	1
Partenon	33.45	1	0.94	5
Pentagon	33.65	1	0.93	22

Tabla 4.17: Resultados FMCW, ruido SP $\delta = 50\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	40.31	0	0.99	2
Lena	34.42	2	0.91	10
Mandril	26.60	5	0.88	14
Peppers	20.33	7	0.80	3
Bee	32.95	2	0.81	10
Butterfly	28.80	3	0.84	10
Cat	31.14	3	0.87	11
Duck1	26.77	5	0.79	13
Duck2	31.37	2	0.88	9
Fish	23.63	8	0.73	51
Owl	28.01	4	0.85	13
Bike	20.60	7	0.73	13
Plane	29.29	2	0.86	10
Boat	29.50	4	0.81	11
Lake1	28.05	4	0.79	12
Lake2	28.21	4	0.82	11
Land1	17.82	18	0.66	22
Land2	25.75	5	0.86	13
Cameraman	28.96	2	0.87	10
Actor	28.41	4	0.81	12
Barbara	25.80	5	0.82	13
Elaine	32.70	3	0.80	9
Tvset	29.26	4	0.84	12
Village	30.64	3	0.85	10
House1	28.03	3	0.84	12
House2	30.81	2	0.76	2
House3	28.69	4	0.80	11
Moon	31.54	3	0.76	2
Partenon	28.63	3	0.82	11
Pentagon	29.25	4	0.79	43

Tabla 4.18: Resultados FMCW, ruido SP $\delta = 80\,\%$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	32.53	0	0.97	3
Lena	29.56	4	0.78	15
Mandril	24.00	10	0.70	19
Peppers	19.25	11	0.64	5
Bee	29.38	3	0.62	15
Butterfly	24.94	6	0.68	16
Cat	27.36	5	0.71	16
Duck1	23.54	8	0.58	19
Duck2	27.74	4	0.72	14
Fish	20.40	14	0.55	57
Owl	24.98	9	0.67	18
Bike	17.85	13	0.48	18
Plane	26.16	5	0.68	15
Boat	25.58	7	0.60	17
Lake1	24.68	8	0.55	17
Lake2	25.14	7	0.62	17
Land1	15.90	27	0.40	27
Land2	22.54	10	0.67	19
Cameraman	24.67	5	0.69	15
Actor	24.72	8	0.61	17
Barbara	22.91	9	0.61	18
Elaine	28.93	5	0.60	14
Tvset	25.41	7	0.64	17
Village	26.87	6	0.66	16
House1	24.80	7	0.65	17
House2	26.35	5	0.51	4
House3	26.24	8	0.55	16
Moon	28.05	6	0.51	4
Partenon	27.66	6	0.60	16
Pentagon	26.25	7	0.56	65



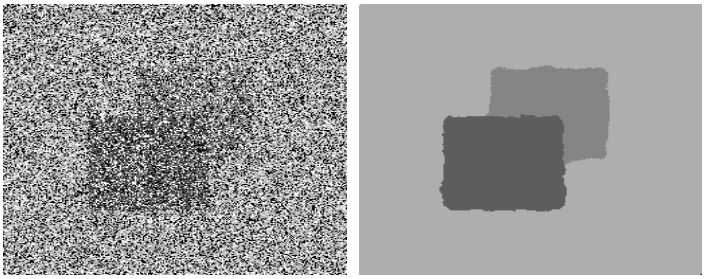
(a) ***Imagen con Ruido*** **(b)** ***FMS***

(c) ***FMA*** **(d)** ***WMF***

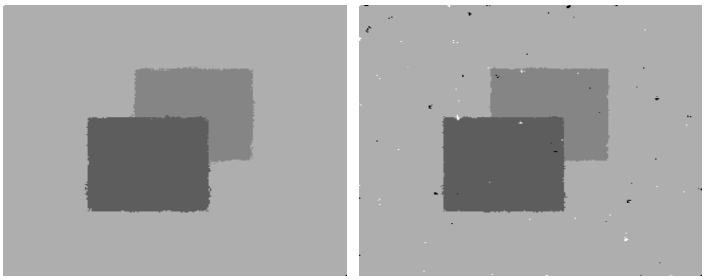
(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

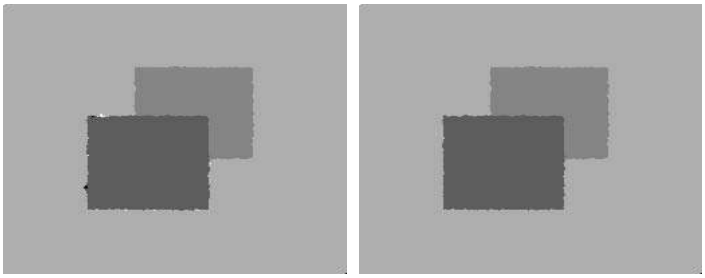
Figura 4.1: Restauración de imagen sintética con ruido sal y pimienta $\delta = 20\,\%$.



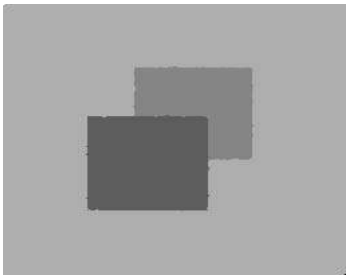
(a) ***Imagen con Ruido*** **(b)** ***FMS***



(c) ***FMA*** **(d)** ***WMF***

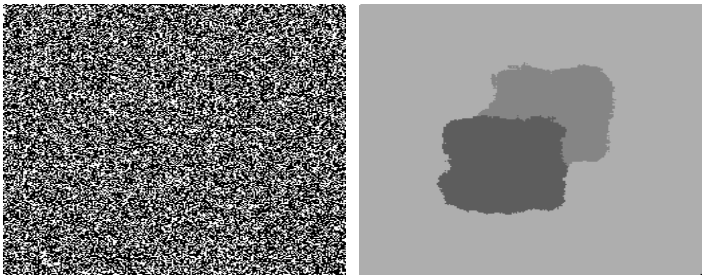


(e) ***FYU*** **(f)** ***FMC***

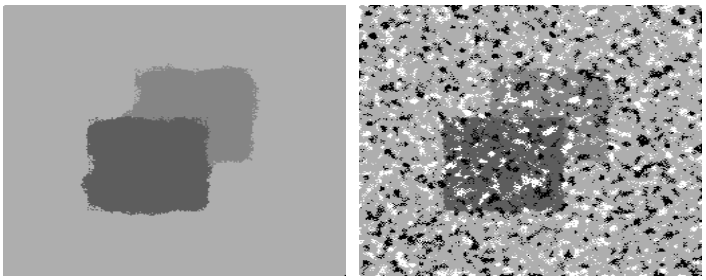


(g) ***FMCW***

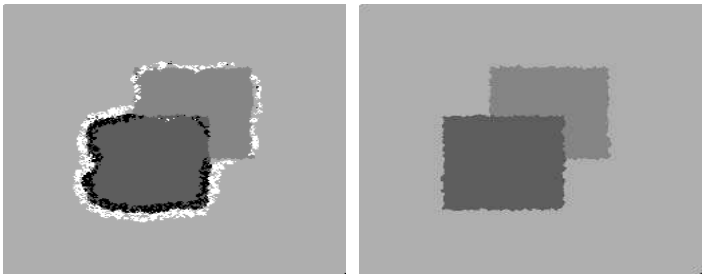
Figura 4.2: Restauración de imagen sintética con ruido sal y pimienta $\delta = 50\,\%$.



(a) ***Imagen con Ruido*** **(b)** ***FMS***



(c) ***FMA*** **(d)** ***WMF***



(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.3: Restauración de imagen sintética con ruido sal y pimienta $\delta = 80\,\%$.

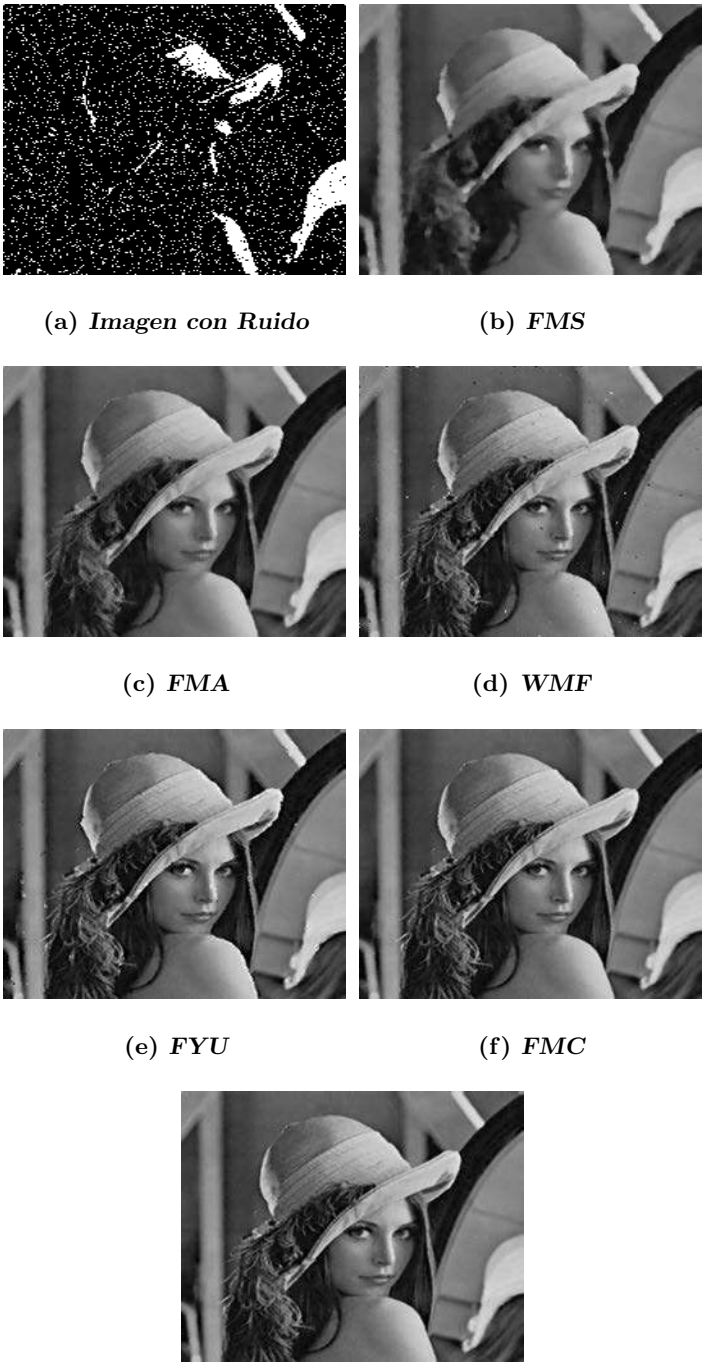
(a) ***Imagen con Ruido*** **(b)** ***FMS***

(c) ***FMA*** **(d)** ***WMF***

(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.4: Restauración de imagen Lena con ruido sal y pimienta $\delta = 20\,\%$.



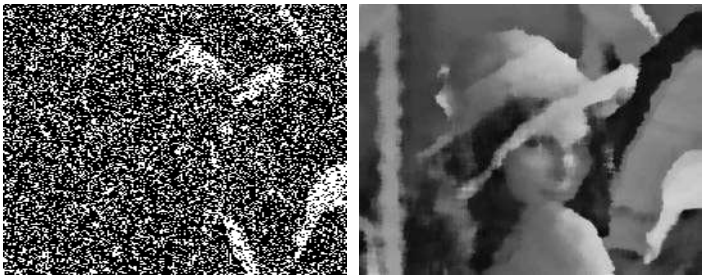
(a) ***Imagen con Ruido*** **(b)** ***FMS***

(c) ***FMA*** **(d)** ***WMF***

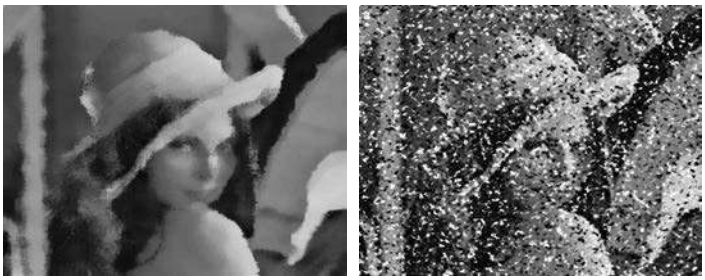
(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.5: Restauración de imagen Lena con ruido sal y pimienta $\delta = 50\,\%$.



(a) ***Imagen con Ruido*** **(b)** ***FMS***



(c) ***FMA*** **(d)** ***WMF***

(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.6: Restauración de imagen Lena con ruido sal y pimienta $\delta = 80\,\%$.

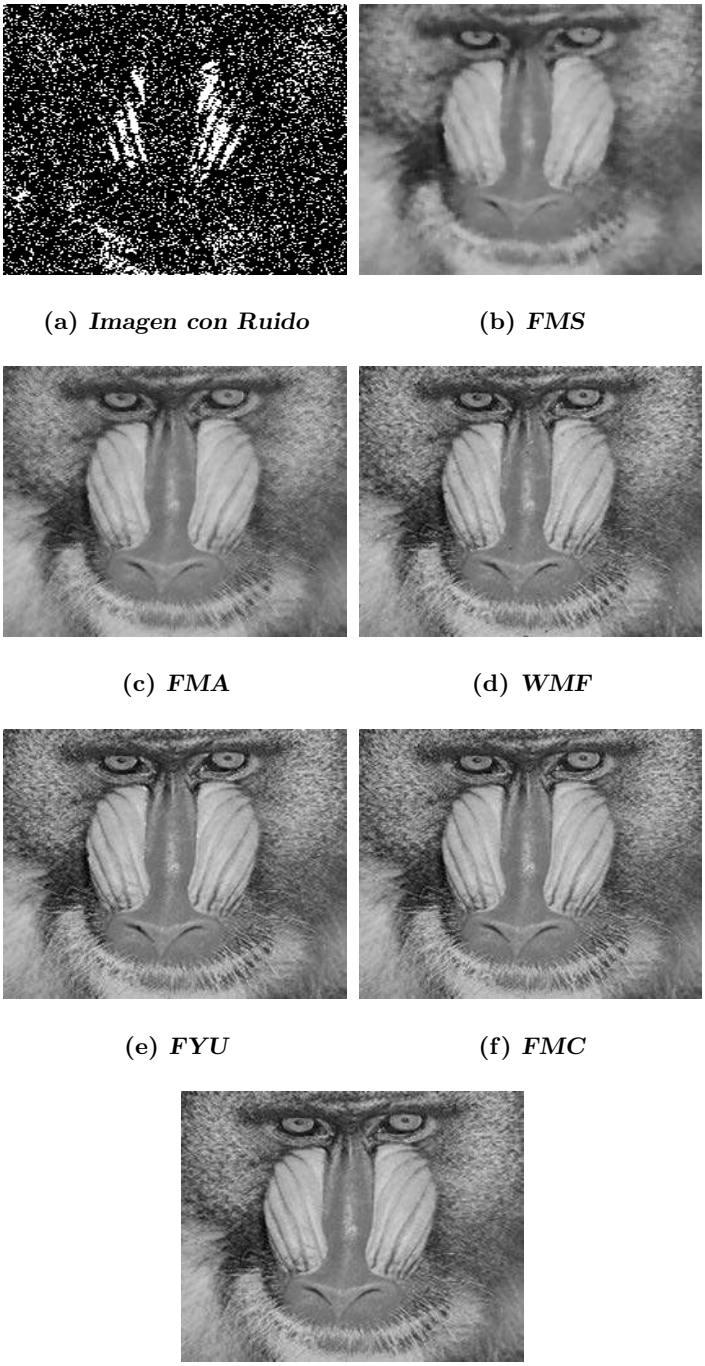
(a) ***Imagen con Ruido*** **(b)** ***FMS***

(c) ***FMA*** **(d)** ***WMF***

(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.7: Restauración de imagen Mandril con ruido sal y pimienta $\delta = 20\,\%$.



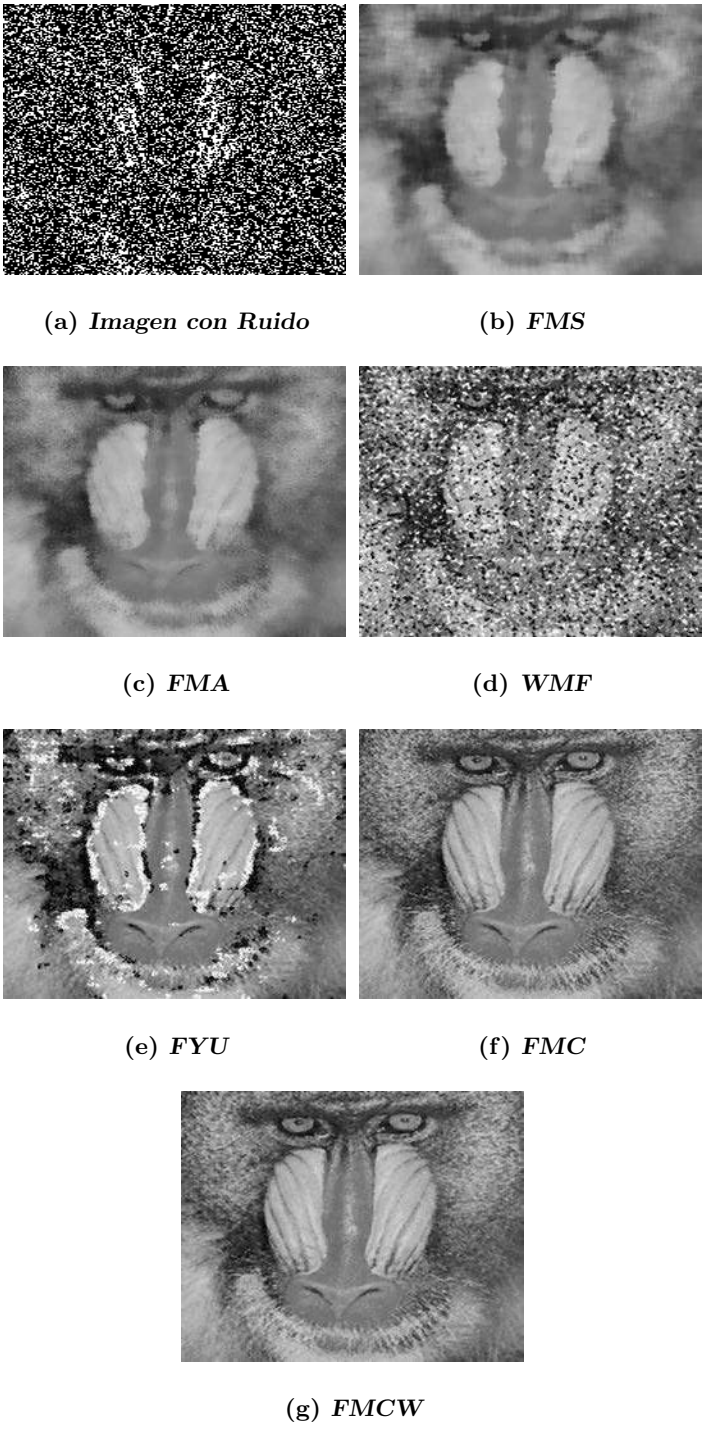
(a) ***Imagen con Ruido*** **(b)** ***FMS***

(c) ***FMA*** **(d)** ***WMF***

(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.8: Restauración de imagen Mandril con ruido sal y pimienta $\delta = 50\,\%$.



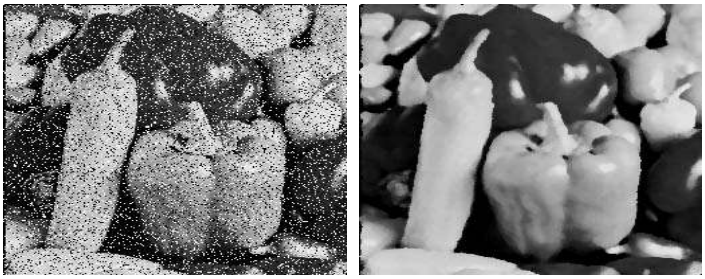
(a) ***Imagen con Ruido*** **(b)** ***FMS***

(c) ***FMA*** **(d)** ***WMF***

(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.9: Restauración de imagen Mandril con ruido sal y pimienta $\delta = 80\,\%$.



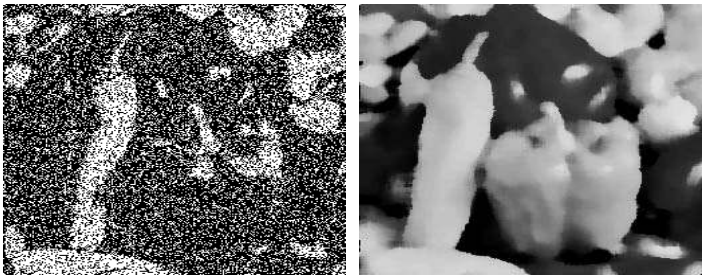
(a) ***Imagen con Ruido*** **(b)** ***FMS***

(c) ***FMA*** **(d)** ***WMF***

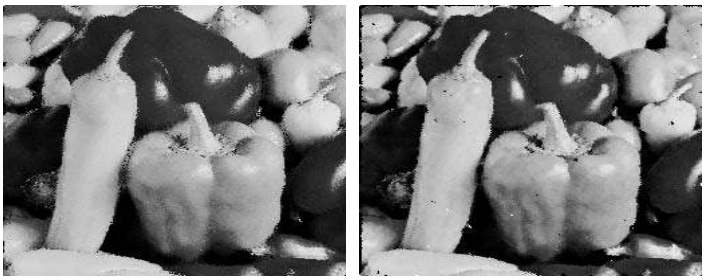
(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.10: Restauración de imagen Peppers con ruido sal y pimienta $\delta = 20\,\%$.



(a) ***Imagen con Ruido*** **(b)** ***FMS***

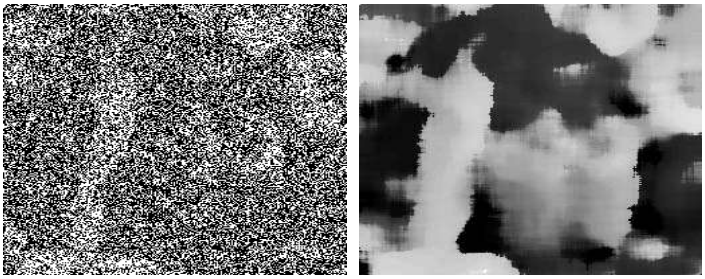


(c) ***FMA*** **(d)** ***WMF***

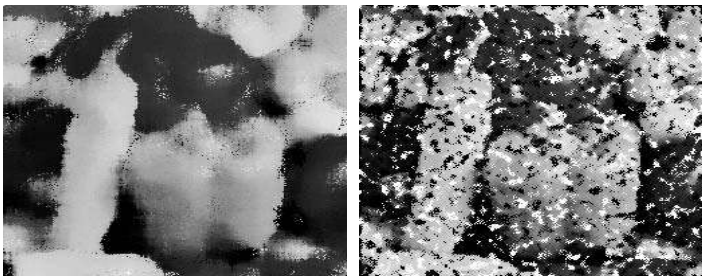
(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

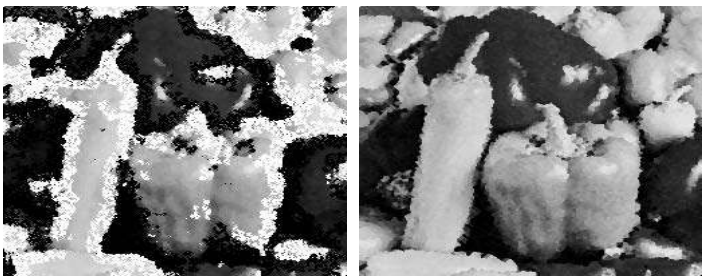
Figura 4.11: Restauración de imagen Peppers con ruido sal y pimienta $\delta = 50\,\%$.



(a) ***Imagen con Ruido*** **(b)** ***FMS***



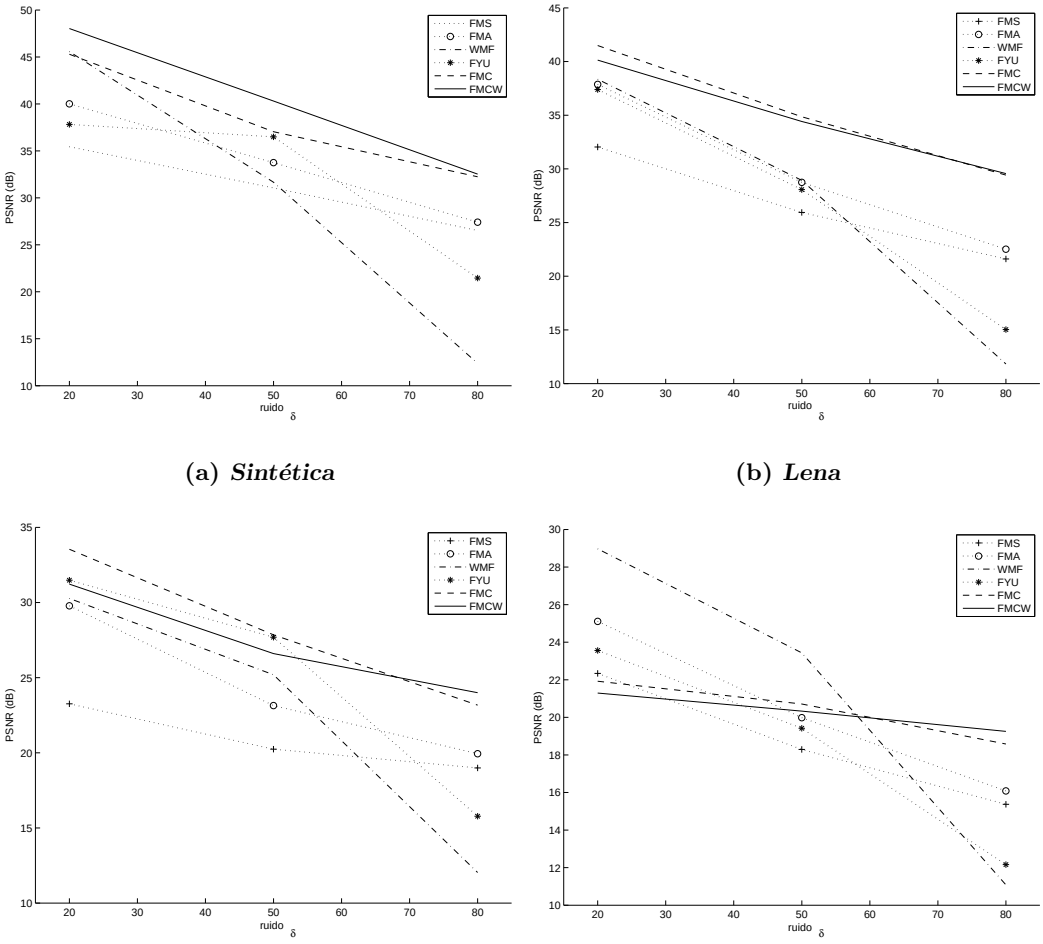
(c) ***FMA*** **(d)** ***WMF***



(e) ***FYU*** **(f)** ***FMC***

(g) ***FMCW***

Figura 4.12: Restauración de imagen Peppers con ruido sal y pimienta $\delta = 80\,\%$.



(a) *Sintética* **(b) *Lena***

(c) *Mandril* **(d) *Peppers***

Figura 4.13: Comparación de los resultados de PSNR para imágenes alteradas con ruido sal y pimienta.

4.3. Conclusiones

En este capítulo se ha propuesto un algoritmo en dos fases para la remoción de ruido sal y pimienta, el cual contempla en su primera fase, el crecimiento de la ventana con base en la información presente en ésta. En la segunda fase se propone un refinamiento de la imagen obtenida, mediante un filtro mediana ponderado. Con base en los experimentos realizados, se puede observar que el algoritmo propuesto proporciona restauraciones de

buena calidad, en tiempos relativamente cortos. En lo que respecta a ruido de baja densidad $\delta = 0.20$, el algoritmo propuesto FMCW mejora la calidad de restauración 1.2 veces la proporcionada por el filtro mediana simple. Con respecto a los algoritmos FMA, WMF y FYU los resultados son prácticamente iguales. En imágenes con ruido sal y pimienta con $\delta = 0.50$ el algoritmo propuesto mejora la calidad de restauración 1.2 veces la proporcionada por el FMS, 1.1 veces la de FMA, y con WMF y FYU los resultados son prácticamente iguales. Finalmente, con ruido de densidad $\delta = 0.80$, el algoritmo FMCW mejora la calidad en la restauración 1.2 veces las correspondientes a los filtros FMS y FMA; 2.1 veces los obtenidos con WMF y 1.5 veces con respecto a FYU. Con base en lo anterior, se puede considerar que los mejores resultados son obtenidos en imágenes con altas densidades de ruido sal y pimienta.

Capítulo 5

Reducción de ruido gaussiano y sal y pimienta de forma robusta

La mayoría de los métodos existentes de reducción de ruido se encuentran orientados a la supresión de un solo tipo de ruido. Sin embargo, debido a que la presencia de ruido puede deberse a diversos factores; es posible encontrar imágenes que contengan diferentes tipos de ruido de manera simultánea, lo que dificulta su restauración. Por ejemplo, los algoritmos desarrollados para eliminar ruido gaussiano son poco eficaces para suprimir ruido de tipo impulsivo. Por lo tanto, es necesario diseñar algoritmos que sean capaces de eliminar ruido, ya sea de un solo tipo o la combinación de tipos diferentes. En este capítulo, se presenta una función diseñada para restaurar una imagen ruidosa y cuya optimización permite la eliminación de ruido gaussiano, de ruido sal y pimienta, así como una combinación de ambos.

La reducción de ruido en imágenes digitales mediante la minimizacion de una función de costos ha sido abordada por diversos autores [L. Rudin92], [P. Charbonnier97], [R.H. Chan05]. De entre ellos, uno de los primeros esfuerzos en esta área es la propuesta presentada por Rudin *et al.* [L. Rudin92] llamada Variación Total (TV), cuya minimización está sujeta a condiciones que involucran estadísticas relacionadas con el ruido, proponiendo

la minimización de la magnitud del gradiente

$$F(\mathbf{I_0}) = \sum_{x=1}^{n_1}\sum_{y=1}^{n_2}\sqrt{\left(\frac{\partial I_0(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I_0(x,y)}{\partial y}\right)^2} \tag{5.1}$$

con respecto a $\mathbf{I_0}$, tomando en cuenta las condiciones

$$\sum_{x=1}^{n_1}\sum_{y=1}^{n_2} I_0(x,y) = \sum_{x=1}^{n_1}\sum_{y=1}^{n_2} I(x,y) \tag{5.2}$$

y

$$\sum_{x=1}^{n_1}\sum_{y=1}^{n_2}\frac{1}{2}\left(I_0(x,y) - I(x,y)\right)^2 = \sigma_\eta^2 \tag{5.3}$$

La condición 5.2 implica que el ruido tiene media cero y la condición 5.3 usa la información *a priori* de que la desviación estándar del ruido es σ_η.

De acuerdo con Charbonnier *et al.* [P. Charbonnier97], la imagen regularizada $\hat{I}_0$ se puede obtener mediante la expresión:

$$\hat{I}_0 = \arg\min_{I_0} F\left(I_0\right) \tag{5.4}$$

donde

$$F\left(I_0\right) = D\left(I_0\right) + \lambda R\left(I_0\right) \tag{5.5}$$

En la ecuación 5.5, el término de fidelidad de los datos D establece que la reconstrucción de la imagen tiene que ser consistente con los datos observados en la imagen I y es de la forma

$$D\left(I_0\right) = \sum_{x=1}^{n_1}\sum_{y=1}^{n_2}\left(I_0(x,y) - I(x,y)\right)^2 \tag{5.6}$$

El término de regularización R en la expresión 5.5, usualmente es una función de las derivadas en la imagen y promueve la formación de regiones planas en la solución. Este término es penalizado por el parámetro λ, el cual controla la contribución del término que hace la consideración *a priori* de que las regiones son suaves. Para el cálculo de las derivadas se consideran diferencias de primer orden entre los pixeles pertenecientes a los cliques horizontales y verticales de tamaño 2 (vecinos inmediatos). Lo anterior conduce a la expresión:

$$R\left(I_0\right) = \sum_{x=1}^{n_1}\sum_{y=1}^{n_2}\phi\left(\frac{\partial I_0(x,y)}{\partial x}\right) + \phi\left(\frac{\partial I_0(x,y)}{\partial y}\right) \tag{5.7}$$

La función $\phi()$ es una función potencial que asigna un costo a cada valor del gradiente de la imagen con el objetivo de preservar los bordes. Según Charbonnier *et al.* [P. Charbonnier97] dicha función, para preservar bordes, debe cumplir con las condiciones siguientes:

- $\frac{\phi'(t)}{2t}$ debe ser continua y decreciente en $[0, +\infty)$.
- $\lim_{t\to+\infty} \frac{\phi'(t)}{2t} = 0$.
- $\lim_{t\to 0^+} \frac{\phi'(t)}{2t} = M, 0 < M < +\infty$.

Este modelo implica que los gradientes pequeños deben ser suavizados, mientras que los gradientes grandes (interpretados como bordes) deben conservarse. Sin embargo, ante la presencia de ruido fuerte como el sal y pimienta, que produce gradientes grandes, el modelo puede no ser adecuado.

Pocos han sido los trabajos relacionados con la optimización de funciones orientados a la remoción de ruido impulsivo, como el propuesto por Chan et al. [R.H. Chan05], quienes presentaron un esquema dividido en dos etapas para la supresión de ruido sal y pimienta.

En la primera etapa, utilizan un filtro mediana adaptable para identificar los pixeles corruptos. En la segunda fase, la imagen es restaurada empleando un método de regularización mediante la minimización de una función, lo cual se aplica solamente a los pixeles identificados como corruptos.

El conjunto de las posiciones de los pixeles corruptos puede ser definido como

$$\Psi = \{(x, y)|I_1(x, y) \neq I(x, y),\ I(x, y) \in \{I_{min}, I_{max}\}\} \tag{5.8}$$

donde $I_1(x, y)$ es obtenido de la siguiente manera

1. Inicializar el tamaño de la ventana $W = W_{min}$.
2. Calcular I_{min}, I_{max} y $\widetilde{N}(x, y)$ que son los valores mínimo, máximo y la mediana del vecindario de tamaño W, respectivamente.

3. Si $I_{min} < \widetilde{N}(x,y) < I_{max}$, entonces ir al paso 5. De otra manera, incrementar el tamaño de la ventana W. Esto es necesario si en la ventana hay únicamente pixeles con intensidades máxima o mínima.

4. Si $W \leq W_{max}$, ir al paso 2. De otra manera, $I_1(x,y) = \widetilde{N}(x,y)$.

5. Si $I_{min} < I(x,y) < I_{max}$, entonces $I_1(x,y) = I(x,y)$, si no, $I_1(x,y) = \widetilde{N}(x,y)$

El conjunto de los pixeles no corruptos estará representado por Ψ^c.

Con la finalidad de restaurar los pixeles corruptos, Chan *et al.* proponen minimizar la siguiente función con respecto a I_0

$$F(\mathbf{I_0}) = \sum_{(x,y)\in\Psi} \left[|I_0(x,y) - I(x,y)| + \frac{\beta}{2}(S_1 + S_2) \right] \tag{5.9}$$

donde

$$S_1 = \sum_{(p,q)\in N\cap\Psi^c} 2\varphi\left(I_0(x,y) - I(p,q)\right)$$

$$S_2 = \sum_{(p,q)\in N\cap\Psi} \varphi\left(I_0(x,y) - I_0(p,q)\right)$$

y N es el vecindario centrado en la posición (x,y). El término de fidelidad de los datos $|I_0(x,y) - I(x,y)|$ restringe que un pixel mal clasificado como corrupto sufra modificaciones importantes. El resto de la función, controlada por el factor β, es un término de regularización que favorece la preservación de los bordes empleando la función potencial

$$\varphi(t) = |t|^\alpha \tag{5.10}$$

para $1 < \alpha \leq 2$.

Recientemente, Cai et al. [J.F. Cai10] presentan un esquema similar al anterior, pero proponiendo una función de la forma

$$F(\mathbf{I_0}) = \sum_{(x,y)\in\Psi} |I_0(x,y) - I(x,y)| + \beta \sum_{(x,y)\in\Omega} \sum_{(p,q)\in N_1} |I_0(x,y) - I_0(p,q)| \tag{5.11}$$

donde $N_1(x,y)$ es el conjunto de los cuatro vecinos cercanos a la posición (x,y) y Ω son todas las posiciones en la imagen.

En este capítulo se presenta una función de costo orientada a la remoción de ruido

tanto gaussiano como impulsivo, ya sea de manera independiente o como una combinación de estos. Esta función de costo está basada en la diferencia entre vecindarios y consta de un término de fidelidad de los datos así como de un término de suavizado, el cual es controlado por un coeficiente. Este coeficiente de peso mide la similitud entre el pixel en proceso y los existentes dentro de su vecindario y es penalizado mediante la anexión de un término adicional. A diferencia de la mayoría de los trabajos en el área de reducción de ruido, el peso de cada pixel dentro del vecindario no está definido mediante una función gaussiana.

5.1. Función de costos basada en vecindarios (RANC)

En la sección anterior se ha mencionado que la función de costos se encuentra conformada por un término de fidelidad de los datos y por un término de regularización. Con respecto al término de fidelidad de los datos, éste puede tener una influencia negativa en la restauración si el pixel $I(x, y)$ se encuentra altamente alterado, ya que este término, precisamente trata de conservar este valor. Por tal motivo, es necesario usar un valor de intensidad más apropiado que el valor observado de $I(x, y)$, i.e. usar una estimación $H(I(x, y))$ del valor de intensidad que probablemente el pixel en la posición (x, y) pudiera tener. El operador $H()$ realiza un filtrado en la imagen, eliminando las intensidades con poca relación con sus vecinos. Para llevar a cabo esto, por ejemplo, se pueden utilizar filtros basados en el valor de la mediana de las intensidades de los pixeles pertenecientes al vecindario centrado en (x, y). La elección del uso del valor de la mediana se debe a su capacidad de preservación de los bordes. Así, el término de fidelidad de los datos se puede expresar como

$$D\left(I_0\right) = \sum_{r \in \Omega} \left(I_0(r) - H(I(r))\right)^2 \tag{5.12}$$

donde $r = (x, y)$ representa la posición de un pixel en el conjunto de posiciones Ω de la imagen.

En lo que respecta al término de regularización, usualmente se considera *a priori* que las regiones son suaves. En este trabajo la consideración que se hace explota la idea de la existencia de estructuras repetitivas en la imagen, propuesta hecha por Buades et al.

[A. Buades05a], donde la remoción de ruido se lleva a cabo mediante promedios pesados. De esta manera, el término de regularización puede ser expresado en términos de la similitud o distancia $d(N(r), N(s))$ entre los vecindarios centrados en las posiciones $r = (x, y)$ y $s = (i, j)$. Tal similitud está determinada por la expresión

$$d(N(r), N(s)) = d(N(x, y), N(i, j)) = \sum_{p=-m_1}^{m_1} \sum_{q=-m_1}^{m_1} \left(I_0(x + p, y + q) - I_0(i + p, j + q)\right)^2 \tag{5.13}$$

Entonces, incorporando las ideas anteriores, se propone la restauración de una imagen mediante la minimización de una función, a la que se denominará RANC (Regularización Adaptable a la Condición de Vecindario), y que es de la forma:

$$F\left(I_0, w\right) = \sum_{r \in \Omega} \left(I_0(r) - H(I(r))\right)^2 + \lambda \sum_{r \in \Omega} \sum_{s \in N(r)} \left(1 - w(r, s)\right)^2 d\left(N(r), N(s)\right) + \mu w(r, s)^2 \tag{5.14}$$

donde el parámetro λ controla la contribución del término de regularización. En otras palabras controla el nivel de suavidad en la restauración de una región, $w(r, s)$ actúa como un peso a la similitud o distancia entre los vecindarios $N(r)$ y $N(s)$ y el parámetro μ penaliza a $w(r, s)$ evitando que la optimización conduzca a que toda la imagen tenga un solo valor de intensidad. Es decir, permite que la restauración sea suave a pedazos. Los valores de los parámetros λ y μ se obtienen de manera experimental.

La solución de la función propuesta puede ser obtenida considerando las condiciones

$$\frac{\partial F}{\partial w\left(r, s\right)} = 0 \tag{5.15}$$

y

$$\frac{\partial F}{\partial I_0(r)} = 0 \tag{5.16}$$

El sistema resultante es diagonal dominante, característica suficiente para garantizar la convergencia al ser resuelto usando un algoritmo de Gauss-Seidel, donde la iteración t-ésima está dada por las ecuaciones:

$$w(r, s)^{(t)} = \frac{d(N(r), N(s))^{(t)}}{\mu + d(N(r), N(s))^{(t)}} \tag{5.17}$$

y

$$I_0(r)^{(t+1)} = \frac{H(I(r)) + \lambda \sum_{s \in N(r)} \left(1 - w(r,s)^{(t)}\right)^2 I_0(s)^{(t)}}{1 + \lambda \sum_{s \in N(r)} \left(1 - w(r,s)^{(t)}\right)^2} \tag{5.18}$$

donde $w(r,s) \in [0,1]$. El peso $w(r,s)$ tiende a 1 cuando $d(N(r), N(s))$ es alto y $w(r,s)$ tiende a 0 cuando $d(N(r), N(s))$ también tiende a 0.

Cabe hacer mención que la expresión 5.17 representa el peso de la contribución que el pixel en la posición $s = (i,j)$ hace a la restauración del pixel en la posición $r = (x,y)$, en función de la similitud o distancia entre sus vecindarios. A diferencia de los métodos tradicionales de reducción de ruido gaussiano como [A. Buades05a], [C. Tomasi98] y [S.M. Smith97], estos pesos no se encuentran definidos por una función gaussiana.

5.2. Experimentos

La implementación del algoritmo RANC se aplicó en las imágenes de las Figuras 3.2, 3.3 y 3.4, las cuales fueron alteradas con ruido gaussiano con desviaciones estándar $\sigma_\eta = 20$, $\sigma_\eta = 40$ y $\sigma_\eta = 60$ y ruido sal y pimienta con densidades $\delta = 20$, $\delta = 50$ y $\delta = 80$. También se realizaron experimentos con ruido gaussiano con desviación estándar $\sigma_\eta = 20$ combinado con ruido sal y pimienta con densidad $\delta = 20$. Con la finalidad de evaluar el desempeño del algoritmo propuesto, éste se compara con el algoritmo de Variación Total (TV) [L. Rudin92] y con el algoritmo propuesto por Cai et al. [J.F. Cai10]. Todos los algoritmos mencionados fueron ejecutados con Matlab R2007b en una PC con procesador Intel Core2 Duo, 2.20 Ghz.

Los parámetros empleados en el algoritmo RANC son $\lambda = 30$ y $\mu = 500$, un tamaño de ventana de vecindario de búsqueda $W_N = 3$ y un tamaño de ventana de vecindario de similitud $W_S = 3$.

En las Tablas 5.1-5.17 se presentan los resultados obtenidos con los algoritmos mencionados. Se compara tanto la relación señal a ruido de pico (PSNR), el error medio absoluto (MAE) y el índice universal de calidad en imágenes (UIQI). También se presentan los tiempos de restauración en segundos.

Las Tablas 5.1-5.6 presentan los resultados de restauración de las imágenes alteradas con ruido gaussiano con diferentes desviaciones estándar, tanto para TV como para

RANC. El operador $H()$ empleado para RANC es el operador identidad, lo que significa que $H(I) = I$. A partir de las tablas como de las Figuras 5.1-5.12, se puede observar que los resultados obtenidos son muy similares para ambos algoritmos, siendo lo tiempos de ejecución mayores para RANC.

La Figura 5.13 muestra una comparación entre los resultados obtenidos con TV y RANC para las imágenes Sintética, Lena, Mandril y Peppers. Se pueden observar mejores resultados con RANC, sobre todo en imágenes con bajo y medio nivel de ruido.

Las Tablas 5.7-5.15 presentan los resultados de restauración de las imágenes alteradas con ruido sal y pimienta con diferentes densidades, para los algoritmos TV, Cai y RANC. El operador $H()$ empleado para RANC es el algoritmo FMCW propuesto en el capítulo anterior. Tanto en las Tablas 5.7-5.15 como en las Figuras 5.14-5.25 se puede notar una mejor calidad de restauración con el algoritmo RANC.

La Figura 5.26 muestra una comparación entre los resultados obtenidos con TV, Cai y RANC para las mismas cuatro imágenes. Se pueden observar mejores resultados con RANC, excepto en la imagen Peppers con bajo nivel de ruido.

Las Tablas 5.16-5.17 presentan los resultados de restauración de las imágenes alteradas con ruido gaussiano con desviación estándar $\sigma_\eta = 20$ y ruido sal y pimienta con densidad $\delta = 20$, para los algoritmos Cai y RANC. El operador $H()$ empleado para RANC es el algoritmo FMCW propuesto en el capítulo anterior. Tanto en las Tablas 5.16-5.17 como en las Figuras 5.27-5.30 se puede notar una mejor calidad de restauración con el algoritmo RANC.

La Figura 5.31 muestra una comparación entre los resultados obtenidos con Cai y RANC para las imágenes Sintética, Lena, Mandril y Peppers. Se pueden observar mejores resultados con RANC para todas las imágenes.

Tabla 5.1: Resultados TV, ruido $\sigma_\eta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	41.66	1	0.07	18
Lena	30.06	5	0.53	85
Mandril	25.33	9	0.67	86
Peppers	26.57	8	0.59	18
Bee	30.31	5	0.43	86
Butterfly	27.81	6	0.61	86
Cat	27.50	7	0.31	85
Duck1	27.21	7	0.52	86
Duck2	28.34	6	0.50	86
Fish	25.14	10	0.49	86
Owl	26.48	9	0.63	86
Bike	25.12	9	0.55	85
Plane	29.47	5	0.45	85
Boat	28.50	7	0.51	85
Lake1	26.97	8	0.44	85
Lake2	27.63	7	0.53	85
Land1	22.54	16	0.80	74
Land2	26.61	8	0.67	86
Cameraman	29.32	5	0.34	85
Actor	26.68	9	0.45	85
Barbara	25.62	8	0.60	85
Elaine	29.99	6	0.48	85
Tvset	28.16	7	0.56	85
Village	28.23	7	0.52	85
House1	27.90	7	0.52	86
House2	30.61	4	0.37	18
House3	27.53	8	0.47	87
Moon	29.09	6	0.39	19
Partenon	26.23	6	0.49	87
Pentagon	28.05	7	0.52	459

Tabla 5.2: Resultados TV, ruido $\sigma_\eta = 40$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	33.47	3	0.06	18
Lena	27.94	7	0.46	85
Mandril	22.69	12	0.48	86
Peppers	23.84	11	0.51	18
Bee	28.44	6	0.35	86
Butterfly	25.65	8	0.52	86
Cat	25.21	10	0.19	86
Duck1	24.42	10	0.38	86
Duck2	26.82	7	0.39	86
Fish	22.53	15	0.35	86
Owl	23.56	12	0.42	85
Bike	20.62	15	0.41	85
Plane	26.56	7	0.36	86
Boat	26.02	9	0.39	85
Lake1	24.20	11	0.25	85
Lake2	24.91	9	0.40	86
Land1	18.89	23	0.56	86
Land2	23.38	12	0.46	86
Cameraman	26.87	8	0.27	85
Actor	24.16	12	0.33	85
Barbara	23.26	11	0.44	85
Elaine	28.30	7	0.42	85
Tvset	25.63	9	0.43	86
Village	26.03	9	0.40	87
House1	25.53	9	0.40	86
House2	28.10	6	0.31	18
House3	24.82	10	0.27	86
Moon	27.19	8	0.29	19
Partenon	23.98	8	0.37	86
Pentagon	25.78	9	0.39	470

Tabla 5.3: Resultados TV, ruido $\sigma_\eta = 60$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	29.86	5	0.05	18
Lena	26.30	9	0.39	86
Mandril	21.50	14	0.38	86
Peppers	22.56	13	0.48	18
Bee	26.99	8	0.30	86
Butterfly	23.99	10	0.45	86
Cat	24.34	12	0.19	86
Duck1	23.14	13	0.31	86
Duck2	26.20	9	0.33	86
Fish	21.33	17	0.32	86
Owl	22.53	14	0.36	85
Bike	19.24	19	0.35	85
Plane	24.96	9	0.32	85
Boat	24.62	11	0.35	86
Lake1	23.22	13	0.21	86
Lake2	23.52	11	0.36	86
Land1	17.40	27	0.38	86
Land2	22.10	14	0.37	86
Cameraman	25.16	10	0.23	85
Actor	23.14	14	0.30	86
Barbara	22.33	13	0.38	86
Elaine	26.72	9	0.37	86
Tvset	24.16	12	0.36	86
Village	24.83	10	0.35	87
House1	23.94	11	0.35	87
House2	25.60	8	0.27	18
House3	23.86	11	0.20	87
Moon	25.87	9	0.25	19
Partenon	23.32	10	0.30	86
Pentagon	24.42	11	0.32	471

Tabla 5.4: Resultados RANC, ruido $\sigma_\eta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	42.10	1	0.07	156
Lena	32.10	5	0.63	34
Mandril	27.31	8	0.76	22
Peppers	28.70	7	0.68	5
Bee	31.02	5	0.47	33
Butterfly	29.16	6	0.67	32
Cat	29.44	6	0.46	22
Duck1	27.87	7	0.57	22
Duck2	30.18	5	0.62	33
Fish	26.94	9	0.62	22
Owl	27.56	8	0.72	22
Bike	28.03	7	0.60	32
Plane	30.73	5	0.52	32
Boat	29.44	6	0.58	32
Lake1	27.94	8	0.56	21
Lake2	28.71	7	0.59	21
Land1	23.19	14	0.86	22
Land2	27.90	7	0.75	22
Cameraman	30.24	5	0.43	32
Actor	28.47	7	0.61	22
Barbara	27.95	7	0.66	32
Elaine	30.36	6	0.53	61
Tvset	29.03	7	0.65	30
Village	29.23	7	0.62	22
House1	28.95	7	0.57	21
House2	31.25	5	0.40	8
House3	27.91	8	0.56	21
Moon	29.20	6	0.42	8
Partenon	28.46	6	0.54	21
Pentagon	28.66	7	0.60	86

Tabla 5.5: Resultados RANC, ruido $\sigma_\eta = 40$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	36.88	2	0.06	274
Lena	28.31	7	0.50	54
Mandril	23.44	12	0.57	22
Peppers	24.58	11	0.57	5
Bee	28.09	6	0.35	165
Butterfly	25.87	9	0.52	142
Cat	25.52	10	0.26	130
Duck1	23.96	11	0.38	120
Duck2	27.27	8	0.40	143
Fish	22.78	14	0.41	121
Owl	23.80	12	0.46	118
Bike	23.40	13	0.50	129
Plane	26.94	7	0.39	128
Boat	25.58	9	0.39	139
Lake1	24.10	12	0.30	118
Lake2	25.15	9	0.43	118
Land1	18.63	24	0.62	107
Land2	23.45	12	0.51	107
Cameraman	26.36	8	0.28	150
Actor	24.30	12	0.38	129
Barbara	23.57	11	0.46	186
Elaine	28.05	7	0.42	257
Tvset	25.01	10	0.41	201
Village	25.87	9	0.39	139
House1	25.07	9	0.40	127
House2	27.03	7	0.31	35
House3	24.60	10	0.26	117
Moon	26.78	8	0.23	42
Partenon	25.58	9	0.33	139
Pentagon	25.16	9	0.35	560

Tabla 5.6: Resultados RANC, ruido $\sigma_\eta = 60$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	33.30	3	0.05	207
Lena	26.03	9	0.41	97
Mandril	22.02	14	0.44	43
Peppers	22.41	14	0.49	11
Bee	26.39	8	0.27	454
Butterfly	23.57	11	0.39	381
Cat	23.16	14	0.15	406
Duck1	21.96	14	0.22	381
Duck2	25.20	9	0.23	426
Fish	20.60	18	0.28	348
Owl	21.67	15	0.28	343
Bike	20.64	17	0.42	322
Plane	24.16	10	0.28	351
Boat	23.32	12	0.25	394
Lake1	22.22	15	0.14	385
Lake2	22.78	12	0.31	345
Land1	16.74	29	0.39	310
Land2	21.24	15	0.32	311
Cameraman	24.16	11	0.20	418
Actor	22.12	15	0.25	375
Barbara	21.74	14	0.32	618
Elaine	25.98	9	0.33	634
Tvset	23.01	13	0.25	3192
Village	23.86	11	0.26	396
House1	22.91	12	0.28	353
House2	24.39	8	0.22	109
House3	23.59	12	0.11	416
Moon	25.53	9	0.16	125
Partenon	24.19	11	0.21	407
Pentagon	23.78	11	0.21	19452

Tabla 5.7: Resultados TV, ruido SP $\delta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	14.02	21	0.02	18
Lena	13.87	21	0.10	85
Mandril	14.68	22	0.23	45
Peppers	13.38	22	0.23	18
Bee	13.84	21	0.07	85
Butterfly	14.30	21	0.15	85
Cat	12.75	22	0.09	85
Duck1	13.96	22	0.17	86
Duck2	14.13	21	0.09	88
Fish	12.79	23	0.20	37
Owl	13.99	22	0.20	86
Bike	12.65	23	0.28	85
Plane	13.40	22	0.12	85
Boat	14.21	22	0.15	86
Lake1	13.71	22	0.14	87
Lake2	13.72	22	0.17	86
Land1	15.11	25	0.49	37
Land2	13.92	22	0.24	86
Cameraman	13.73	22	0.11	86
Actor	13.21	22	0.16	86
Barbara	13.99	22	0.18	85
Elaine	14.35	22	0.10	85
Tvset	14.36	21	0.17	86
Village	14.21	22	0.13	86
House1	13.95	22	0.15	86
House2	13.98	21	0.11	19
House3	14.23	22	0.13	87
Moon	14.68	22	0.09	19
Partenon	14.35	21	0.11	86
Pentagon	14.84	21	0.14	475

Tabla 5.8: Resultados TV, ruido SP $\delta = 50$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	11.78	48	0.01	18
Lena	11.63	50	0.05	85
Mandril	11.83	49	0.10	86
Peppers	11.00	52	0.12	18
Bee	11.88	49	0.03	85
Butterfly	12.06	49	0.07	86
Cat	10.51	53	0.04	86
Duck1	11.93	50	0.08	85
Duck2	11.85	49	0.03	88
Fish	10.45	55	0.09	87
Owl	11.04	51	0.08	86
Bike	9.94	54	0.14	85
Plane	11.03	50	0.06	86
Boat	11.69	50	0.07	85
Lake1	11.14	52	0.06	87
Lake2	11.29	50	0.08	85
Land1	11.91	53	0.23	85
Land2	11.41	51	0.11	86
Cameraman	11.31	51	0.06	86
Actor	10.44	53	0.06	86
Barbara	11.66	50	0.08	85
Elaine	11.88	49	0.04	85
Tvset	12.08	49	0.07	87
Village	11.98	49	0.05	87
House1	11.66	50	0.07	87
House2	11.65	49	0.06	19
House3	12.06	49	0.05	87
Moon	12.19	48	0.04	19
Partenon	11.84	49	0.05	87
Pentagon	12.64	48	0.06	475

Tabla 5.9: Resultados TV, ruido SP $\delta = 80$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	10.14	71	0.00	18
Lena	9.88	74	0.02	85
Mandril	10.63	71	0.03	85
Peppers	9.30	77	0.05	18
Bee	10.17	72	0.01	86
Butterfly	10.71	71	0.02	86
Cat	8.99	79	0.01	86
Duck1	10.05	73	0.03	87
Duck2	10.23	72	0.01	88
Fish	9.05	79	0.04	87
Owl	9.72	74	0.03	86
Bike	8.53	82	0.06	85
Plane	9.63	74	0.02	85
Boat	10.38	72	0.02	86
Lake1	9.48	77	0.02	87
Lake2	9.77	74	0.03	86
Land1	10.11	74	0.07	86
Land2	9.65	74	0.04	86
Cameraman	9.69	75	0.02	86
Actor	9.14	78	0.02	86
Barbara	10.04	73	0.02	86
Elaine	10.47	71	0.01	85
Tvset	10.49	71	0.02	86
Village	10.27	72	0.02	86
House1	10.02	73	0.03	86
House2	10.06	71	0.02	19
House3	10.13	73	0.01	87
Moon	10.67	70	0.01	19
Partenon	10.37	71	0.01	86
Pentagon	11.14	70	0.02	474

Tabla 5.10: Resultados Cai, ruido SP $\delta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	39.63	0	0.07	24
Lena	31.94	2	0.82	146
Mandril	26.96	3	0.86	148
Peppers	25.85	4	0.83	24
Bee	33.10	2	0.77	145
Butterfly	28.48	3	0.83	146
Cat	30.81	2	0.63	145
Duck1	28.13	3	0.79	147
Duck2	30.02	2	0.83	148
Fish	25.82	5	0.74	147
Owl	27.94	3	0.86	147
Bike	24.93	4	0.83	148
Plane	30.04	2	0.80	146
Boat	29.95	2	0.85	146
Lake1	29.46	3	0.86	148
Lake2	28.69	3	0.84	148
Land1	19.92	10	0.82	147
Land2	27.01	4	0.85	146
Cameraman	30.54	2	0.81	143
Actor	28.95	3	0.80	147
Barbara	28.48	3	0.83	147
Elaine	32.07	2	0.84	148
Tvset	29.79	3	0.85	147
Village	30.52	2	0.84	143
House1	28.80	3	0.82	146
House2	31.47	2	0.84	24
House3	29.92	2	0.87	147
Moon	31.97	2	0.86	24
Partenon	28.34	2	0.83	145
Pentagon	30.19	2	0.85	582

Tabla 5.11: Resultados Cai, ruido SP $\delta = 50$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	27.51	4	0.05	52
Lena	23.68	9	0.46	317
Mandril	21.61	10	0.58	317
Peppers	17.93	17	0.45	52
Bee	25.01	8	0.40	314
Butterfly	21.57	10	0.51	314
Cat	23.25	10	0.36	317
Duck1	22.02	11	0.43	318
Duck2	23.56	7	0.51	318
Fish	18.88	17	0.46	316
Owl	21.39	12	0.53	317
Bike	18.23	17	0.44	318
Plane	22.00	10	0.35	315
Boat	23.18	9	0.54	317
Lake1	22.96	10	0.50	317
Lake2	21.30	11	0.46	314
Land1	16.55	21	0.56	314
Land2	20.61	13	0.51	312
Cameraman	22.93	9	0.38	309
Actor	21.52	12	0.47	315
Barbara	21.80	13	0.50	315
Elaine	24.10	12	0.50	316
Tvset	23.00	14	0.50	314
Village	23.40	12	0.50	311
House1	21.70	13	0.40	317
House2	23.20	11	0.40	52
House3	23.90	12	0.50	317
Moon	26.00	10	0.60	52
Partenon	21.70	13	0.50	312
Pentagon	24.20	14	0.60	1274

Tabla 5.12: Resultados Cai, ruido SP $\delta = 80$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	16.76	21	0.02	73
Lena	16.26	28	0.16	424
Mandril	17.67	22	0.25	423
Peppers	12.54	45	0.16	70
Bee	17.19	26	0.14	417
Butterfly	16.78	23	0.22	419
Cat	14.08	38	0.14	425
Duck1	16.92	28	0.17	426
Duck2	16.55	23	0.17	422
Fish	13.06	41	0.18	423
Owl	15.33	31	0.20	424
Bike	12.28	50	0.17	426
Plane	14.88	34	0.12	421
Boat	17.88	22	0.22	423
Lake1	15.24	33	0.17	424
Lake2	15.11	34	0.15	418
Land1	14.75	33	0.26	418
Land2	15.44	34	0.19	416
Cameraman	15.99	28	0.13	414
Actor	14.58	35	0.17	422
Barbara	16.10	26	0.20	421
Elaine	18.00	28	0.20	422
Tvset	18.20	26	0.20	419
Village	17.30	24	0.20	417
House1	15.70	28	0.20	423
House2	17.50	26	0.20	70
House3	16.90	26	0.20	420
Moon	21.40	22	0.20	70
Partenon	17.40	26	0.20	419
Pentagon	20.60	31	0.30	1683

Tabla 5.13: Resultados RANC, ruido SP $\delta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	54.84	0	0.97	17
Lena	45.84	0	0.98	25
Mandril	36.11	1	0.98	27
Peppers	22.68	3	0.93	21
Bee	39.07	1	0.94	66
Butterfly	36.18	1	0.96	67
Cat	37.94	1	0.95	49
Duck1	32.06	2	0.93	72
Duck2	37.95	1	0.97	67
Fish	28.15	4	0.81	119
Owl	34.36	1	0.96	112
Bike	28.49	2	0.93	112
Plane	38.54	1	0.96	112
Boat	38.31	1	0.96	113
Lake1	35.65	1	0.95	112
Lake2	35.83	1	0.95	112
Land1	21.45	8	0.88	134
Land2	31.55	1	0.97	112
Cameraman	36.01	1	0.96	113
Actor	35.63	1	0.94	129
Barbara	34.36	1	0.96	112
Elaine	28.49	2	0.93	112
Tvset	38.54	1	0.96	112
Village	38.31	1	0.96	113
House1	35.65	1	0.95	112
House2	35.83	1	0.95	112
House3	21.45	8	0.88	134
Moon	31.55	1	0.97	112
Partenon	36.01	1	0.96	113
Pentagon	35.63	1	0.94	129

Tabla 5.14: Resultados RANC, ruido SP $\delta = 50$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	49.29	0	0.90	40
Lena	38.94	1	0.93	54
Mandril	29.03	4	0.92	22
Peppers	21.71	6	0.84	43
Bee	34.07	2	0.81	22
Butterfly	30.81	3	0.88	76
Cat	32.49	2	0.80	43
Duck1	28.29	4	0.80	165
Duck2	32.80	2	0.89	22
Fish	24.59	7	0.71	195
Owl	29.22	4	0.88	22
Bike	22.92	5	0.79	272
Plane	31.59	2	0.88	268
Boat	32.17	3	0.85	272
Lake1	29.84	4	0.82	92
Lake2	30.02	3	0.85	43
Land1	18.62	16	0.70	34
Land2	27.62	4	0.90	60
Cameraman	30.78	2	0.87	270
Actor	30.18	3	0.84	146
Barbara	27.27	4	0.86	434
Elaine	33.93	3	0.82	38
Tvset	32.15	3	0.88	269
Village	32.39	3	0.87	419
House1	29.92	3	0.87	433
House2	34.34	2	0.80	39
House3	29.59	4	0.81	27
Moon	32.55	3	0.78	4
Partenon	30.63	2	0.84	226
Pentagon	30.70	3	0.83	172

Tabla 5.15: Resultados RANC, ruido SP $\delta = 80$

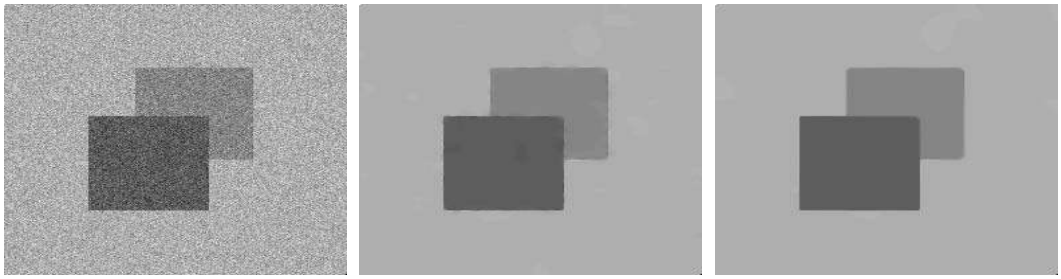
Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	37.44	0	0.65	64
Lena	31.56	3	0.80	77
Mandril	23.84	9	0.72	17
Peppers	19.55	10	0.67	66
Bee	30.53	3	0.63	52
Butterfly	26.18	5	0.72	44
Cat	28.21	5	0.62	34
Duck1	24.38	8	0.58	52
Duck2	28.78	4	0.73	25
Fish	21.06	12	0.51	270
Owl	25.25	8	0.70	26
Bike	18.59	11	0.52	429
Plane	26.43	4	0.69	68
Boat	26.53	6	0.63	60
Lake1	25.38	8	0.56	34
Lake2	25.64	6	0.66	44
Land1	16.26	27	0.42	61
Land2	23.44	9	0.69	43
Cameraman	25.42	5	0.66	430
Actor	25.62	7	0.62	78
Barbara	23.25	8	0.65	112
Elaine	30.75	5	0.65	113
Tvset	26.47	7	0.66	59
Village	27.83	5	0.67	43
House1	24.90	6	0.67	60
House2	28.13	4	0.56	43
House3	25.88	7	0.54	35
Moon	29.01	5	0.53	11
Partenon	25.72	5	0.62	34
Pentagon	26.83	6	0.59	172

Tabla 5.16: Resultados Cai, ruido $G\sigma_\eta = 20 + SP\delta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	32.66	2	0.05	24
Lena	25.24	9	0.32	148
Mandril	19.97	17	0.11	146
Peppers	20.61	15	0.40	24
Bee	26.21	8	0.25	143
Butterfly	21.65	13	0.30	149
Cat	24.05	10	0.12	148
Duck1	22.09	14	0.15	148
Duck2	23.19	10	0.15	147
Fish	20.37	17	0.23	149
Owl	21.06	16	0.18	147
Bike	18.45	19	0.25	147
Plane	23.70	10	0.22	147
Boat	23.19	12	0.20	147
Lake1	22.56	13	0.12	147
Lake2	21.92	13	0.23	148
Land1	15.80	32	0.11	148
Land2	20.37	17	0.17	148
Cameraman	24.34	9	0.17	146
Actor	22.52	14	0.21	148
Barbara	21.62	15	0.29	147
Elaine	25.44	10	0.27	148
Tvset	22.88	13	0.20	147
Village	23.56	12	0.21	148
House1	22.11	12	0.22	148
House2	24.57	8	0.18	24
House3	23.04	12	0.08	148
Moon	25.31	9	0.11	24
Partenon	21.41	11	0.17	147
Pentagon	23.26	12	0.14	593

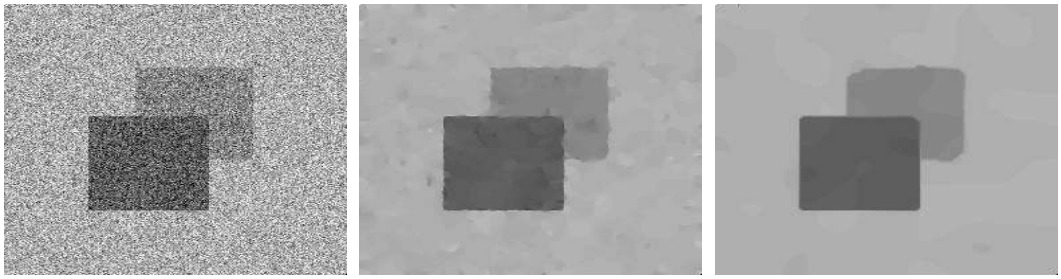
Tabla 5.17: Resultados RANC, ruido $G\sigma_\eta = 20 + SP\delta = 20$

Imagen	PSNR(db)	MAE	UIQI	tiempo (s)
Sintética	35.94	2	0.06	31
Lena	31.10	5	0.61	33
Mandril	26.32	9	0.73	22
Peppers	24.11	9	0.64	5
Bee	30.21	5	0.44	33
Butterfly	28.41	7	0.64	22
Cat	27.64	8	0.42	22
Duck1	26.59	8	0.54	22
Duck2	29.29	6	0.60	22
Fish	24.62	12	0.56	22
Owl	26.73	9	0.68	22
Bike	22.73	12	0.54	33
Plane	29.53	6	0.49	22
Boat	28.43	7	0.57	22
Lake1	26.58	9	0.50	22
Lake2	27.89	7	0.57	22
Land1	20.24	18	0.78	11
Land2	26.38	8	0.71	22
Cameraman	28.13	7	0.39	33
Actor	26.43	9	0.55	22
Barbara	26.16	8	0.61	22
Elaine	30.00	6	0.51	43
Tvset	28.06	7	0.62	22
Village	28.44	7	0.59	22
House1	27.80	7	0.55	22
House2	29.77	5	0.38	8
House3	26.99	8	0.51	22
Moon	28.72	7	0.39	8
Partenon	27.89	7	0.52	22
Pentagon	27.89	7	0.56	87



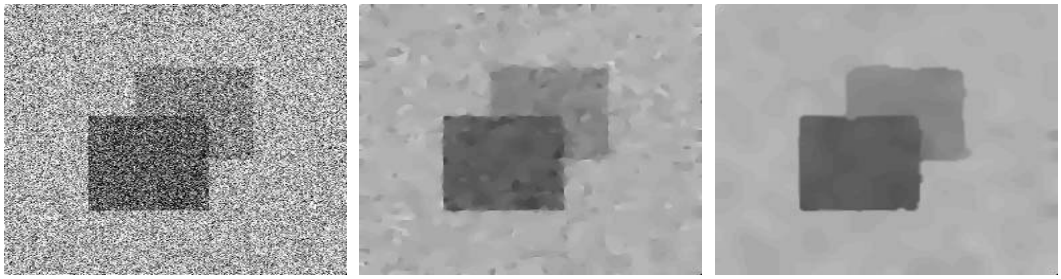
(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.1: Restauración de imagen sintética con ruido gaussiano $\sigma_\eta = 20$.



(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.2: Restauración de imagen sintética con ruido gaussiano $\sigma_\eta = 40$.



(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.3: Restauración de imagen sintética con ruido gaussiano $\sigma_\eta = 60$.

(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.4: Restauración de imagen Lena con ruido gaussiano $\sigma_\eta = 20$.

(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.5: Restauración de imagen Lena con ruido gaussiano $\sigma_\eta = 40$.

(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.6: Restauración de imagen Lena con ruido gaussiano $\sigma_\eta = 60$.

(a) ***Imagen con Ruido*** **(b)** ***TV*** **(c)** ***RANC***

Figura 5.7: Restauración de imagen Mandril con ruido gaussiano $\sigma_\eta = 20$.

(a) ***Imagen con Ruido*** **(b)** ***TV*** **(c)** ***RANC***

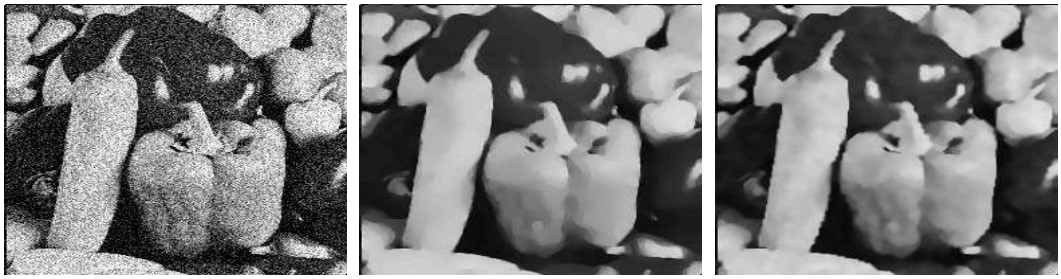
Figura 5.8: Restauración de imagen Mandril con ruido gaussiano $\sigma_\eta = 40$.

(a) ***Imagen con Ruido*** **(b)** ***TV*** **(c)** ***RANC***

Figura 5.9: Restauración de imagen Mandril con ruido gaussiano $\sigma_\eta = 60$.

(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.10: Restauración de imagen Peppers con ruido gaussiano $\sigma_\eta = 20$.

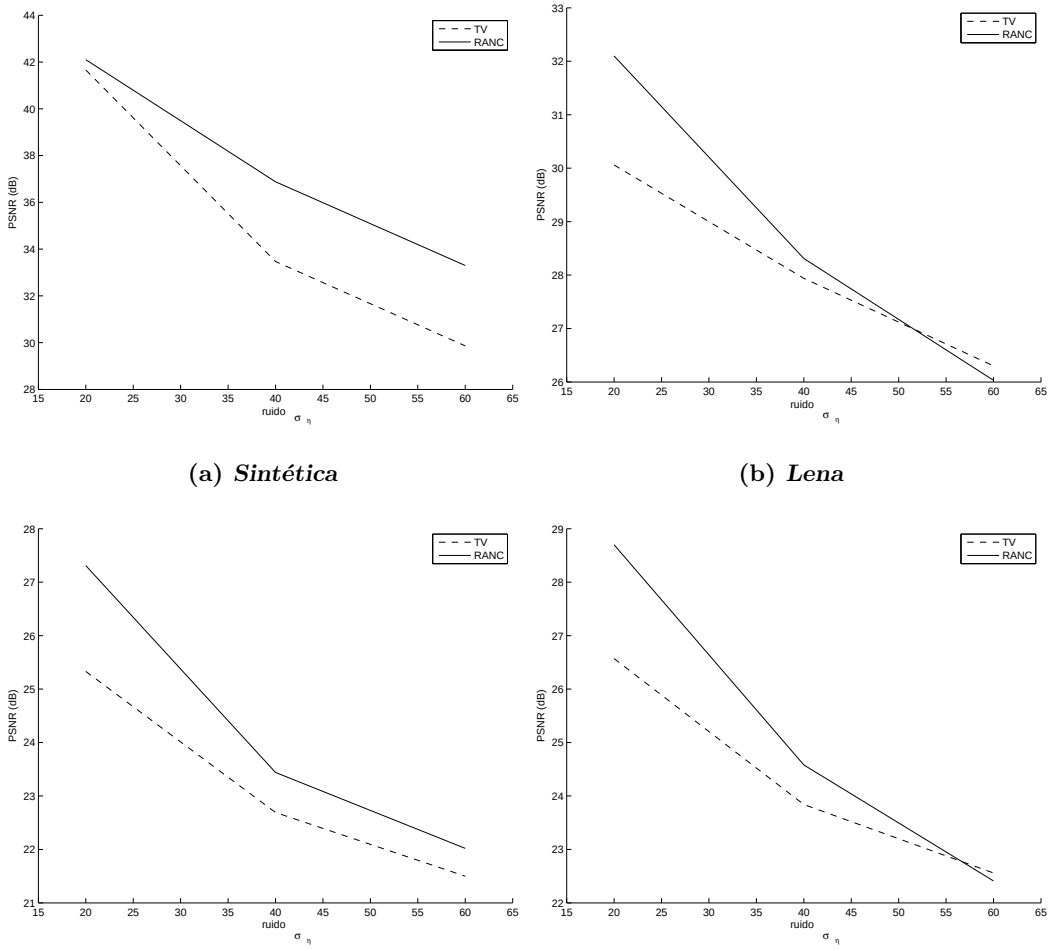


(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

Figura 5.11: Restauración de imagen Peppers con ruido gaussiano $\sigma_\eta = 40$.

(a) *Imagen con Ruido* **(b) *TV*** **(c) *RANC***

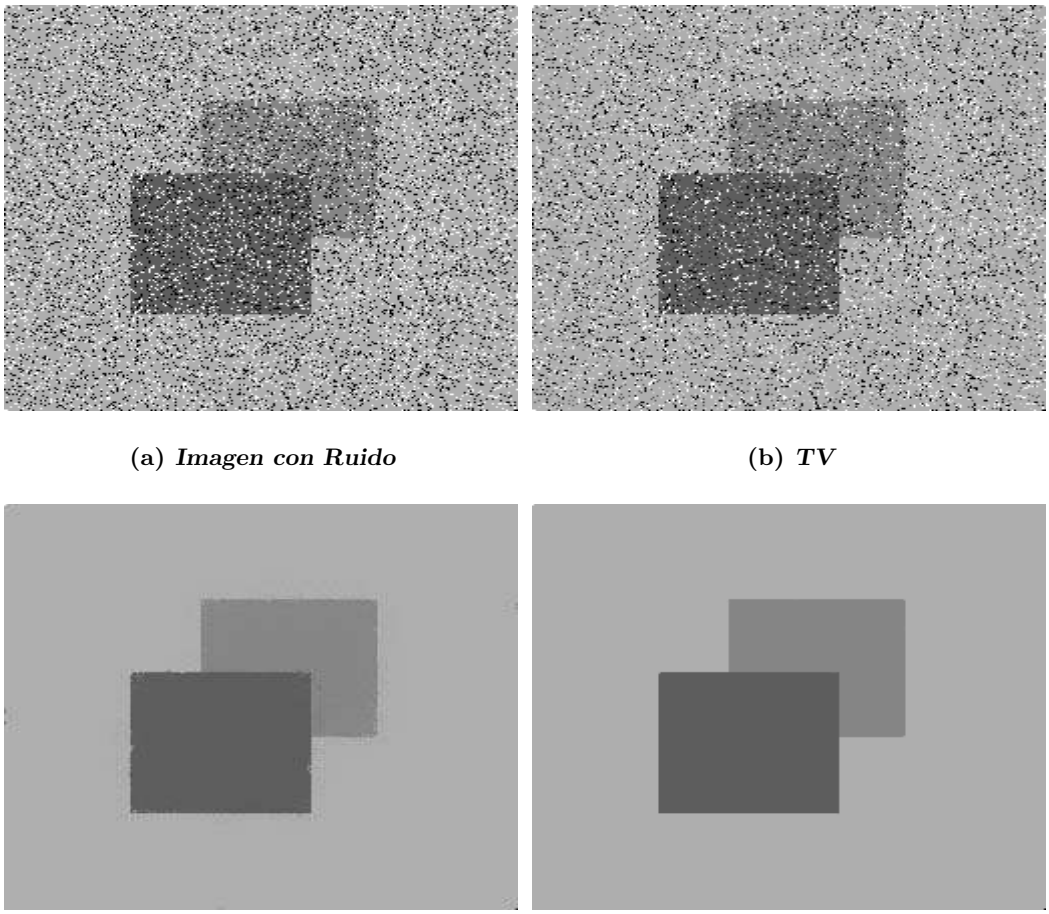
Figura 5.12: Restauración de imagen Peppers con ruido gaussiano $\sigma_\eta = 60$.



(a) *Sintética* **(b) *Lena***

(c) *Mandril* **(d) *Peppers***

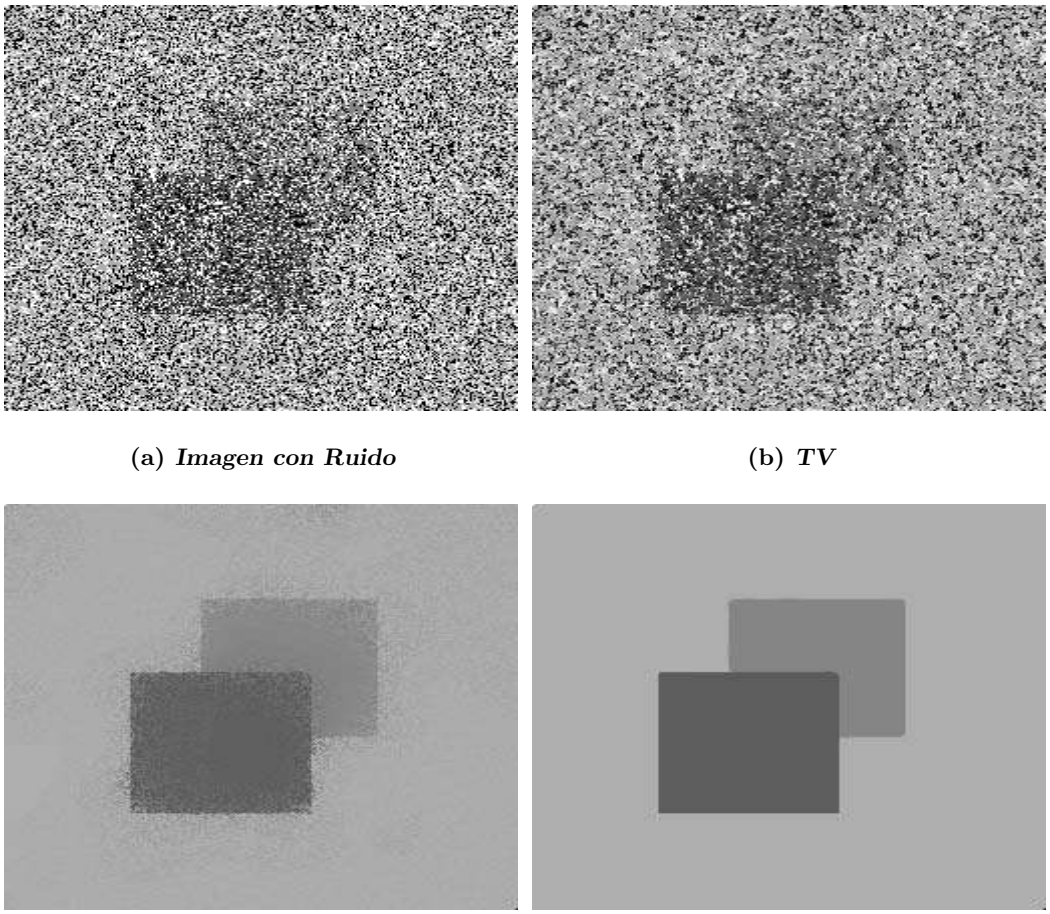
Figura 5.13: Comparación de los resultados de PSNR para imágenes alteradas con ruido gaussiano.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

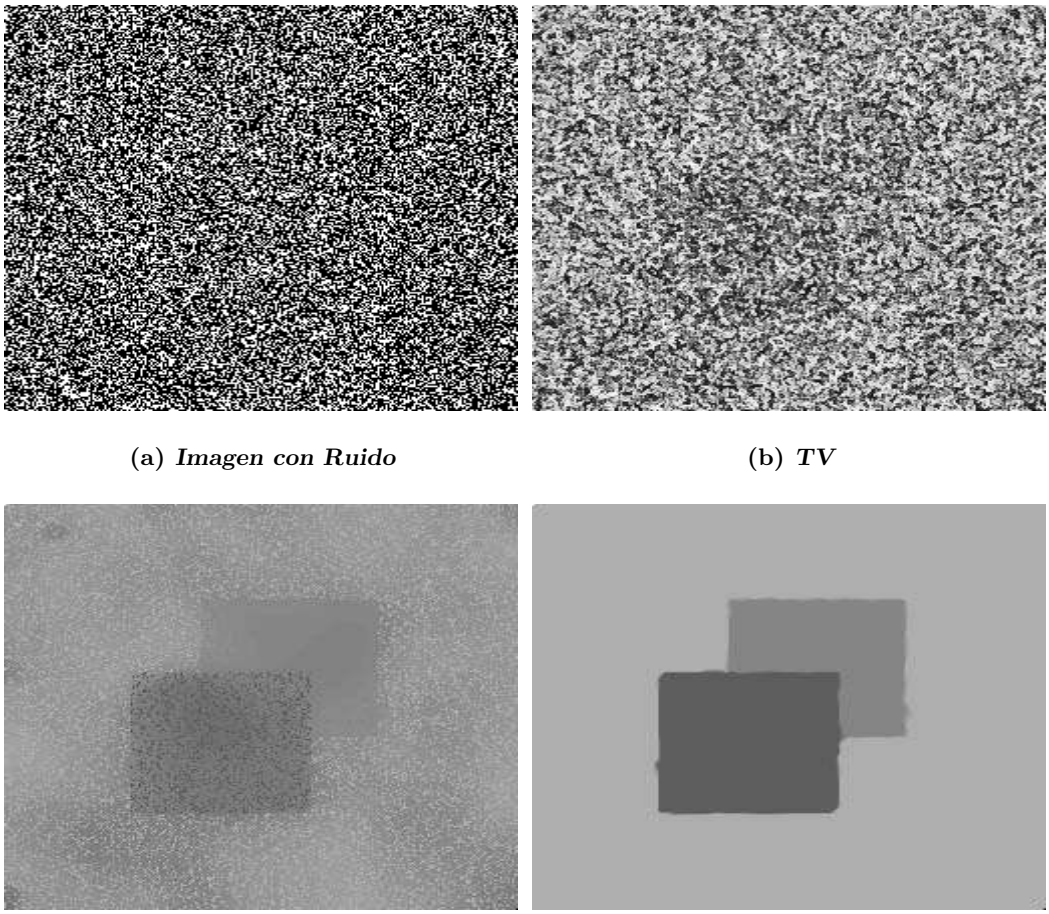
Figura 5.14: Restauración de imagen sintética con ruido sal y pimienta $\delta = 20$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

Figura 5.15: Restauración de imagen sintética con ruido sal y pimienta $\delta = 50$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

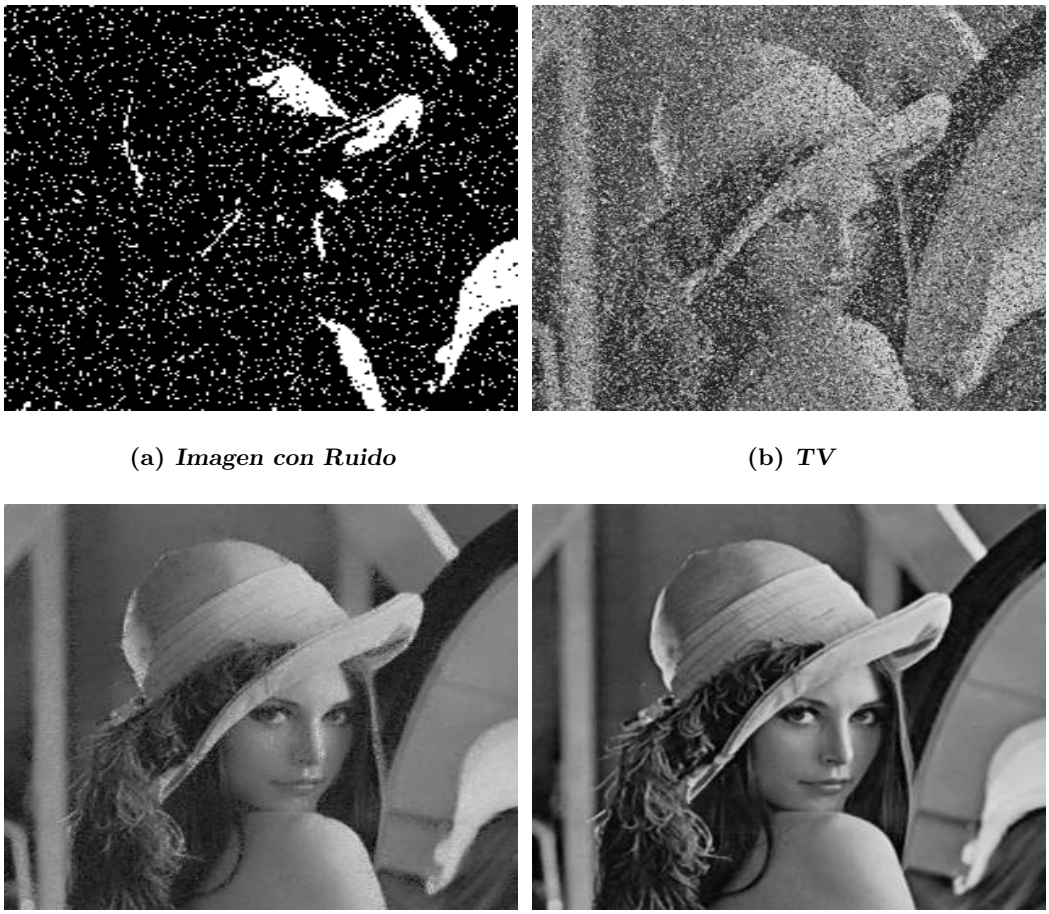
(c) ***Cai*** **(d)** ***RANC***

Figura 5.16: Restauración de imagen sintética con ruido sal y pimienta $\delta = 80$.

(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

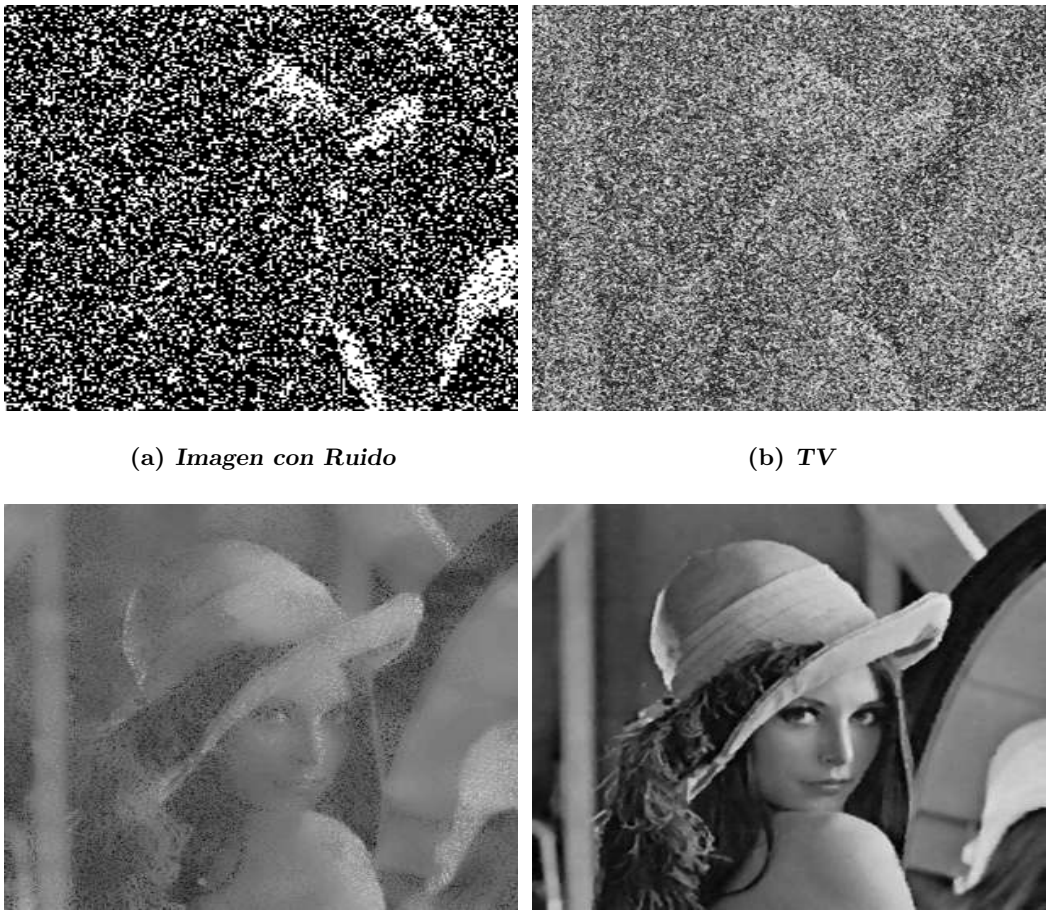
Figura 5.17: Restauración de imagen Lena con ruido sal y pimienta $\delta = 20$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

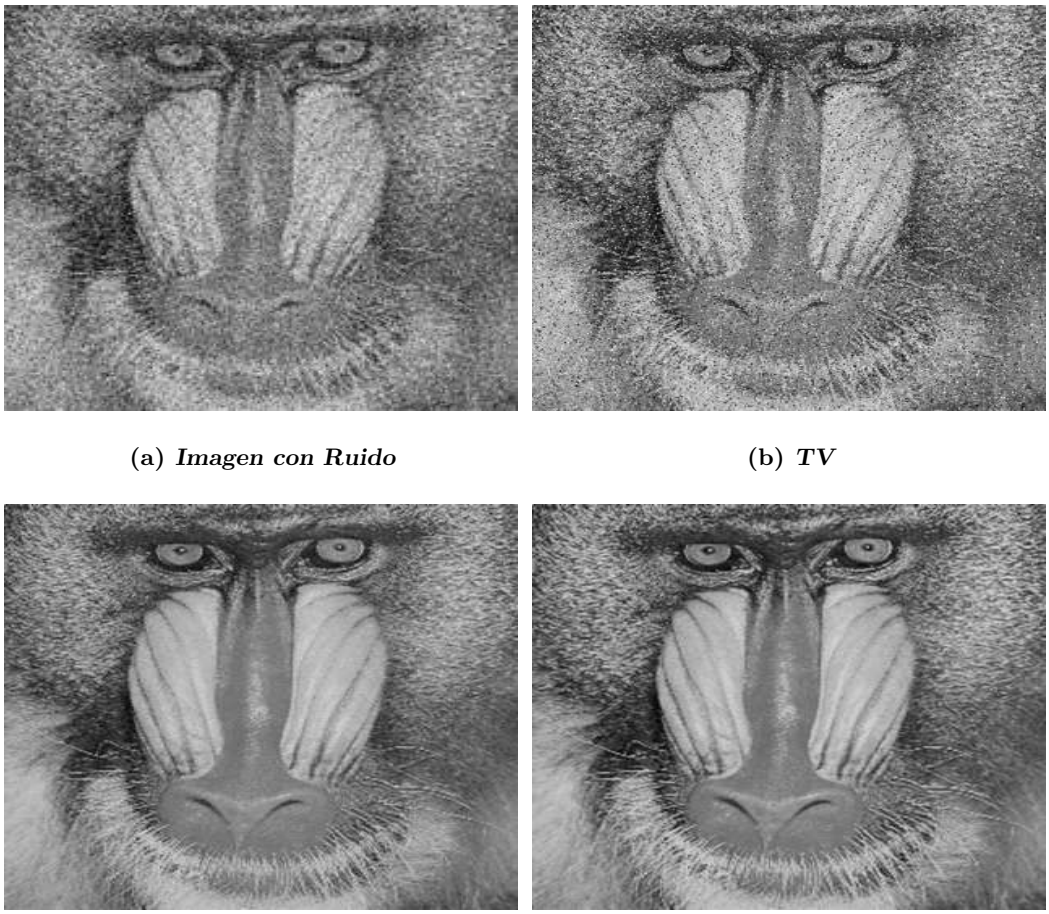
Figura 5.18: Restauración de imagen Lena con ruido sal y pimienta $\delta = 50$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

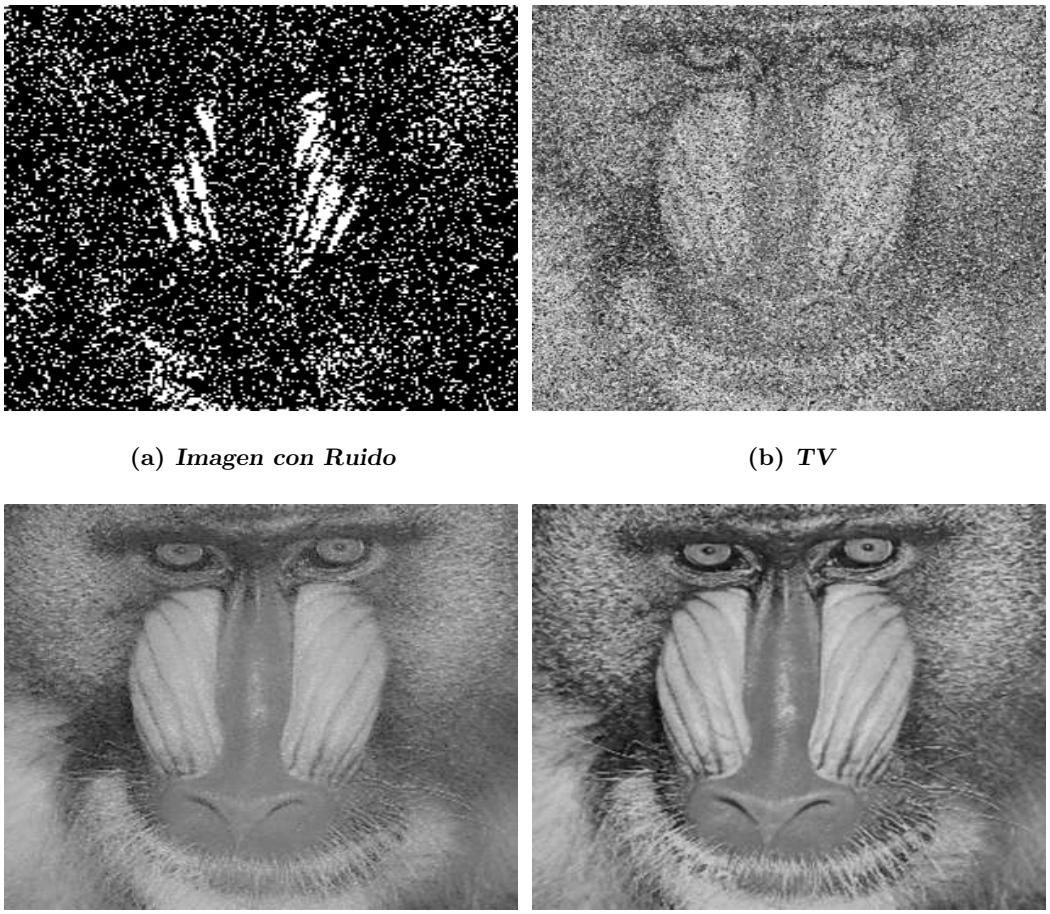
Figura 5.19: Restauración de imagen Lena con ruido sal y pimienta $\delta = 80$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

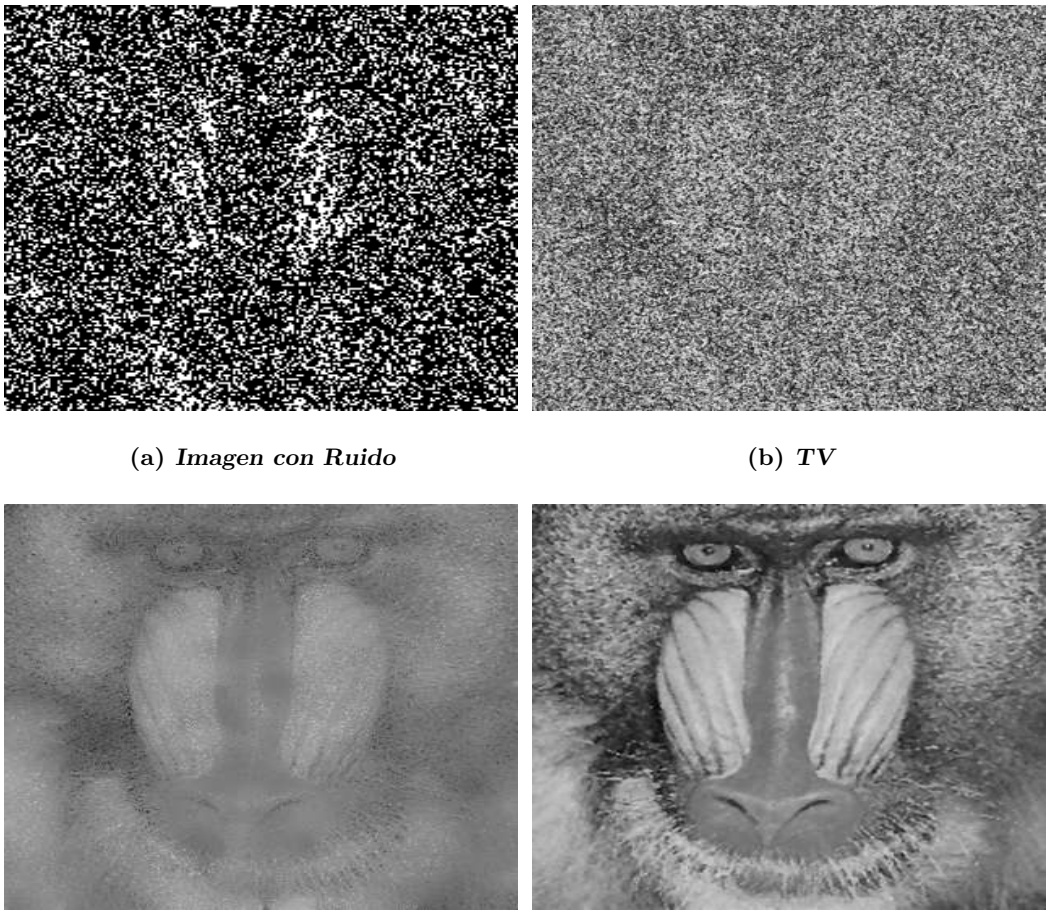
Figura 5.20: Restauración de imagen Mandril con ruido sal y pimienta $\delta = 20$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

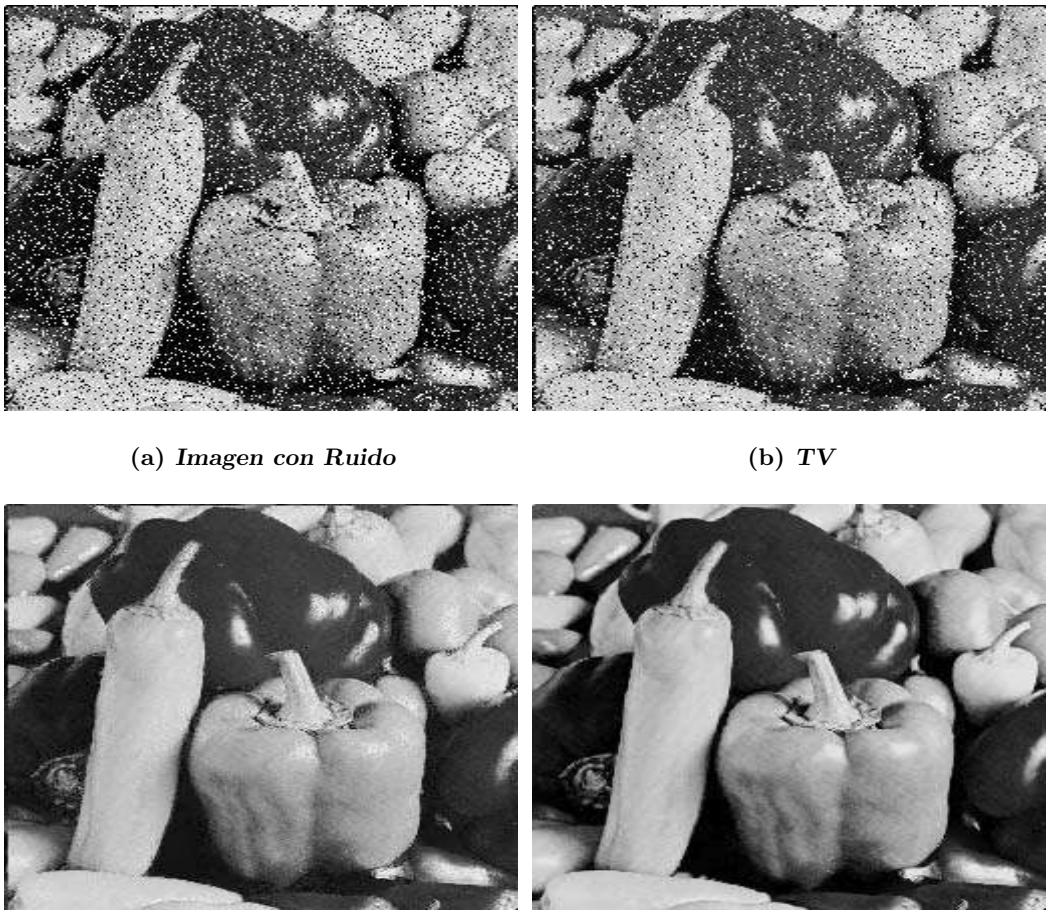
Figura 5.21: Restauración de imagen Mandril con ruido sal y pimienta $\delta = 50$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

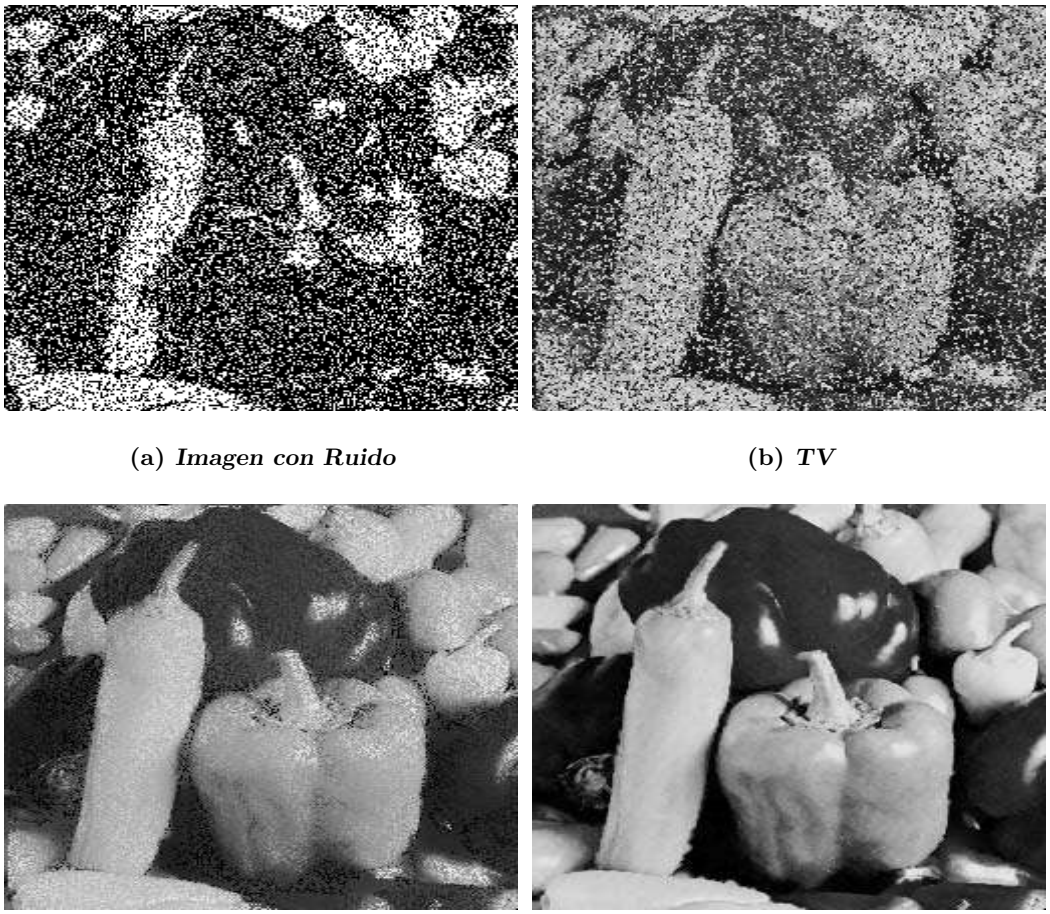
Figura 5.22: Restauración de imagen Mandril con ruido sal y pimienta $\delta = 80$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

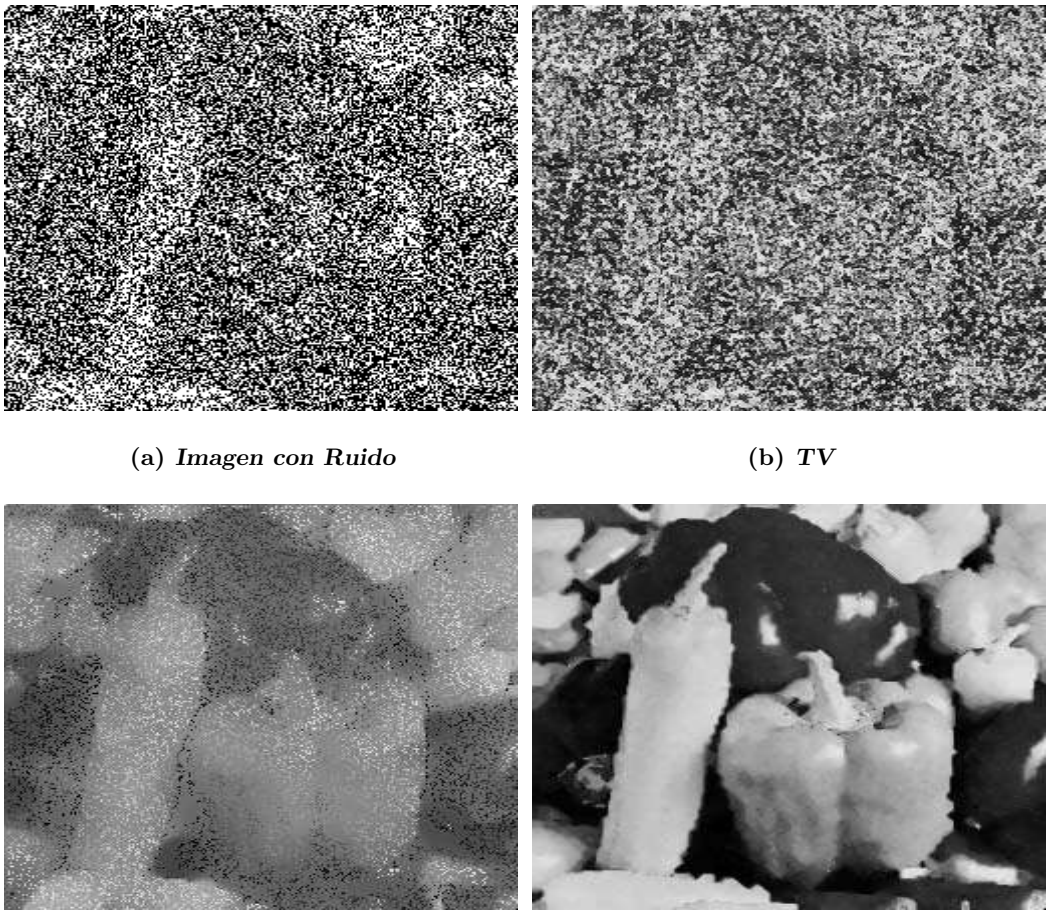
Figura 5.23: Restauración de imagen Peppers con ruido sal y pimienta $\delta = 20$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

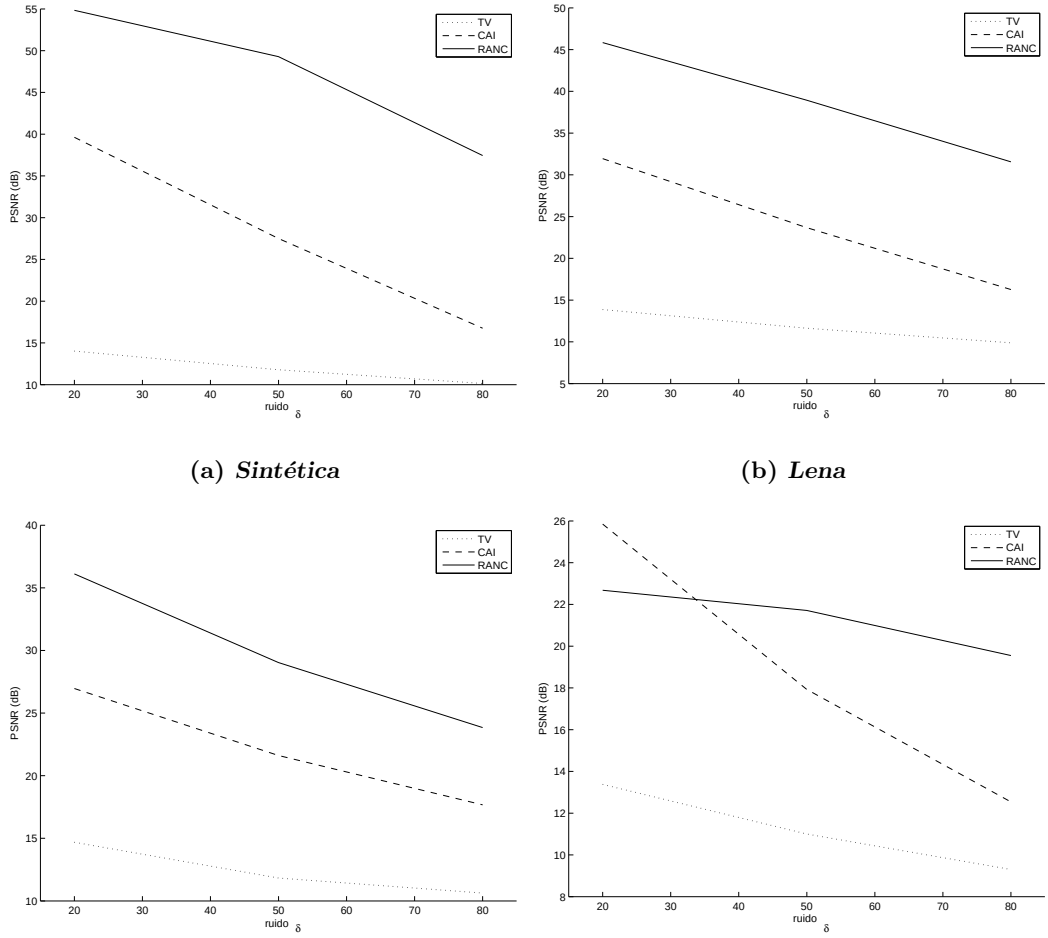
Figura 5.24: Restauración de imagen Peppers con ruido sal y pimienta $\delta = 50$.



(a) ***Imagen con Ruido*** **(b)** ***TV***

(c) ***Cai*** **(d)** ***RANC***

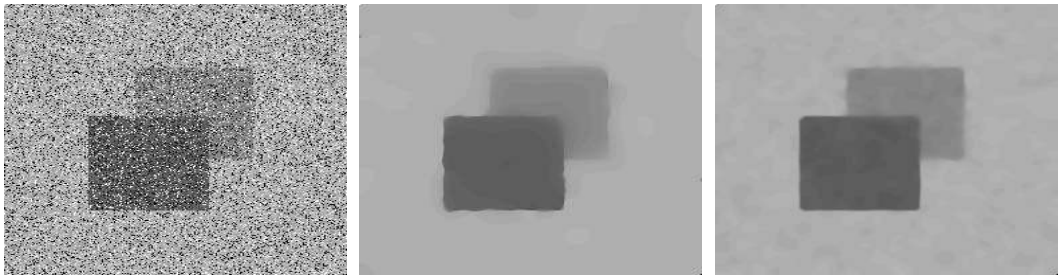
Figura 5.25: Restauración de imagen Peppers con ruido sal y pimienta $\delta = 80$.



(a) *Sintética* **(b) *Lena***

(c) *Mandril* **(d) *Peppers***

Figura 5.26: Comparación de los resultados de PSNR para imágenes alteradas con ruido sal y pimienta.



(a) ***Imagen con Ruido*** **(b)** ***Cai*** **(c)** ***RANC***

Figura 5.27: Restauración de imagen sintética con ruido gaussiano $\sigma_\eta = 20$ + sal y pimienta $\delta = 20$.

(a) ***Imagen con Ruido*** **(b)** ***Cai*** **(c)** ***RANC***

Figura 5.28: Restauración de imagen Lena con ruido gaussiano $\sigma_\eta = 20$ + sal y pimienta $\delta = 20$.

(a) ***Imagen con Ruido*** **(b)** ***Cai*** **(c)** ***RANC***

Figura 5.29: Restauración de imagen Mandril con ruido gaussiano $\sigma_\eta = 20$ + sal y pimienta $\delta = 20$.

(a) *Imagen con Ruido* **(b) *Cai*** **(c) *RANC***

Figura 5.30: Restauración de imagen Peppers con ruido gaussiano $\sigma_\eta = 20$ + sal y pimienta $\delta = 20$.

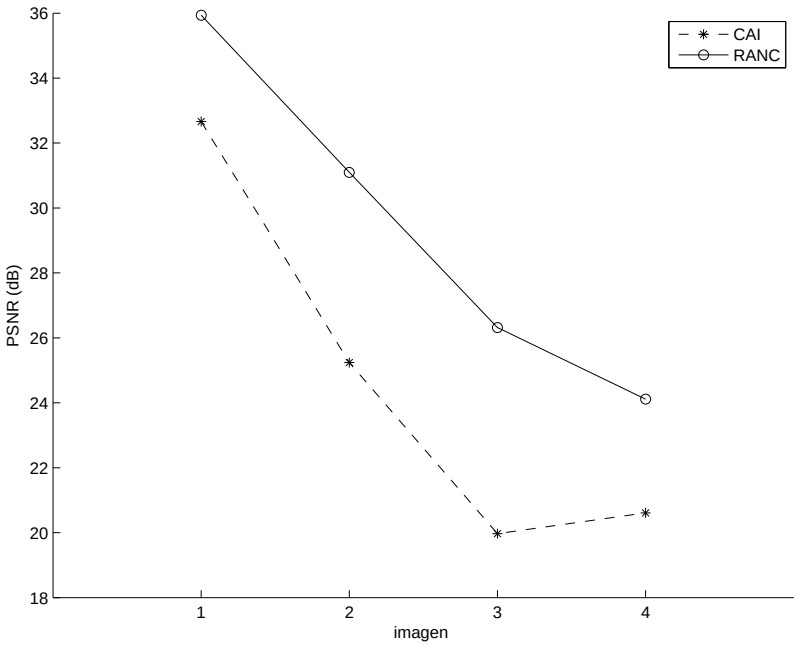


Figura 5.31: Comparación de los resultados de PSNR para imágenes alteradas con ruido gaussiano $\sigma_\eta = 20$ + sal y pimienta $\delta = 20$: 1) Sintética; 2) Lena; 3) Mandril; 4) Peppers.

5.3. Conclusiones

En este capítulo se ha propuesto un algoritmo de regularización adaptable a la condición de vecindario (RANC) para la reducción de ruido tanto gaussiano como sal y

pimienta, además de una combinación de ambos. El algoritmo RANC realiza una regularización de la imagen basada en la similitud entre vecindarios. Con fines de una mejor calidad en la restauración se emplea una estimación obtenida mediante la aplicación de un operador $H()$ en la imagen con ruido. El algoritmo RANC fue comparado con otros algoritmos como Variación Total (TV) y el propuesto por Cai et al. Los resultados muestran que para ruido gaussiano, tanto TV como RANC, presentan resultados similares de restauración. En lo que respecta a imágenes con ruido sal y pimienta, el algoritmo TV falla completamente mientras que RANC mejora la calidad de restauración con respecto a Cai 1.1, 1.3 y 1.6 veces para densidades de ruido de *20 %*, *50 %* y *80 %*, respectivamente. En cuanto a la combinación de ruido gaussiano con sal y pimienta, el algoritmo RANC también presenta los mejores resultados, alrededor de 1.2 veces los obtenidos con el método propuesto por Cai. Lo anterior muestra experimentalmente que el algoritmo es robusto ya que, a diferencia de los métodos comparados en este trabajo, puede restaurar una gran variedad de imágenes ante la presencia de dos tipos de ruido, ya sea por separado o una combinación de ambos.

Capítulo 6

Conclusiones

6.1. Conclusiones Generales

La reducción de ruido es importante como parte del pre-proceso de una imagen, preparando la imagen para procesos posteriores como la segmentación o la identificación de objetos.

Se presentaron tres propuestas para la reducción de los tipos de ruido gaussiano y sal y pimienta, que con frecuencia se pueden encontrar en imágenes digitales.

1. En el caso del ruido gaussiano, existen eficientes algoritmos orientados en su reducción. Uno de los más recientes y ampliamente utilizado es el de medias no-locales, basado en la existencia de estructuras repetitivas en la imagen. Sin embargo, sus tiempos de ejecución pueden ser considerados altos. Por tal motivo, en este trabajo se presentaron algoritmos que permiten reducir los tiempos de ejecución, obteniéndose resultados similares en cuanto a la calidad obtenida en las restauraciones.

2. En lo que respecta al ruido sal y pimienta, se ha diseñado y propuesto un algoritmo basado en el cálculo del valor de la mediana del conjunto de elementos que se encuentran en una ventana. El tamaño de estas ventana se adapta a las características de los pixeles pertenecientes a la misma. Los resultados muestran una buena calidad de

restauración, similar (y en ocasiones mejor) a otros métodos existentes, inclusive ante la presencia de altos niveles de ruido.

3. También se ha diseñado y presentado una función de costos basada en las características de los vecindarios de la imagen, denominada RANC. Esta función de costos consiste en dos partes, una relacionada a un término de fidelidad de los datos (el cual considera una estimación obtenida mediante un operador), y la otra un término de regularización que preserva estructuras. Este último es controlado por un coeficiente que mide la similitud entre dos pixeles. Este coeficiente es penalizado agregando un término adicional. A diferencia de muchos de los trabajos en el área de la remoción de ruido, el peso de cada pixel del vecindario no se encuentra definido por una función gaussiana.

En todos los casos, los resultados obtenidos muestran un buen desempeño de esta propuesta, comparada con otros algoritmos. El algoritmo RANC fue probado con ruido gaussiano, sal y pimienta y una combinación de ambos, lo que muestra su robustez de manera experimental ya que puede restaurar diferentes imágenes ante la presencia de diferentes tipos de ruido, ya sea por separado o combinados. Los coeficientes utilizados en el algoritmo RANC, fueron determinados de manera experimental y garantizan robustez dentro de amplio rango cada uno de ellos.

6.2. Trabajos Futuros

La investigación reportada en esta tesis, puede continuarse mediante las siguientes sugerencias de trabajo futuro:

1. La modificación de los algoritmos presentados, con la finalidad de hacerlos óptimos en cuanto a la calidad de la restauración y con mejores tiempos de ejecución.

2. La aplicación de los algoritmos propuestos en la restauración de diferentes tipos de imágenes como las imágenes médicas (tomografías, etc), industriales, entre otros.

3. Ejecutar los algoritmos en imágenes diferentes a imágenes de 8 bits.

4. La restauración de imágenes con tipos de ruido diferentes al gaussiano y sal y pimienta.

5. Explorar la utilización de otras alternativas para la minimización de la función de regularización, como pueden ser las técnicas que aporta la computación evolutiva.

6. La determinación de los rangos de valores más adecuados para los parámetros presentes en los algoritmos para los diferentes tipos de ruido.

Referencias

[A. Buades05a] A. Buades, J. M., B. Coll. A non-local algorithm for image denoising. *in proc. IEEE Computer society Conference on computer vision and pattern recognition*, 2:60–65, 2005.

[A. Buades05b] A. Buades, J. M., B. Coll. A review of image denoising algorithms, with a new one. *SIAM, multiscale modeling simulation*, 4(2):490–530, 2005.

[A. Shrivastava07] A. Shrivastava, S. G. P. L., M. Shinde. An approach-effect of an exponential distribution on different medical images. *International journal of computer science and network security*, 7(9):235–241, september 2007.

[Ambardar02] Ambardar, A. Procesamiento de señales analógicas y digitales. *Thomson*, 2002.

[Brownrigg84] Brownrigg, D. The weighted median filter. *Communications of the ACM*, 27(8):1984, 1984.

[C. Tomasi98] C. Tomasi, R. M. Bilateral filtering for gray and color images. *in proc. IEEE international conference on computer vision*, págs. 839–846, 1998.

[C.A. Júnez09a] C.A. Júnez, F. V. A simple algorithm for image denoising based on non-local means and preliminary segmentation. *2009 Electronics,*

Robotics and Automotive Mechanics Conference, CERMA, págs. 204–208, 2009.

[C.A. Júnez09b] C.A. Júnez, F. V. Un algoritmo iterativo para la segmentación de imágenes con ruido. *7° Congreso Internacional sobre Innovación y Desarrollo Tecnológico CIINDET*, 2009.

[C.A. Júnez10a] C.A. Júnez, F. V. A salt and pepper noise removal and restoration refinement algorithm. *International Multiconference on Complexity, Informatics and Cybernetics, IMCIC*, 2010.

[C.A. Júnez10b] C.A. Júnez, N. P., F. Velasco. Salt and pepper noise detection based on non-local means. *7th International Conference on Informatics in Control, Automation and Robotics ICINCO*, 2010.

[Canny86] Canny, J. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, PAMI-8(6):679–698, november 1986.

[Chacón07] Chacón, M. Procesamiento digital de imágenes. *Editorial Trillas*, 2007.

[D.T. Kuan85] D.T. Kuan, T. S. P. C., A.A. Sawchuk. Adaptive noise smoothing filter for images with signal-dependent noise. *IEEE Transactions on pattern analysis and machine intelligence*, 7:165–177, march 1985.

[E. Dougherty99] E. Dougherty, J. A. Non linear filters for image processing. *IEEE Press*, 1999.

[H. Hwang95] H. Hwang, R. A. H. Adaptive median filters: New algorithms and results. *IEEE Transactions on image processing*, 4(4):499–502, april 1995.

[Horn01] Horn, B. Robot vision. *MIT Press, McGraw-Hill*, 1rst edition, 2001.

[J. Astola97] J. Astola, P. K. Fundamentals of nonlinear digital filtering. *CRC Press*, 1997.

[Jahne97] Jahne, B. Practical handbook on image processing for scientific applications. *CRC Press*, 1997.

[J.F. Cai10] J.F. Cai, M. N., R.H. Chan. Fast two-phase image deblurring under impulsive noise. *J. Math Imaging Vis*, 36:46–53, 2010.

[L. Rudin92] L. Rudin, E. F., S. Osher. Nonlinear total variation based noise removal algorithms. *Physica D*, (60):259–268, 1992.

[M. Mahmoudi05] M. Mahmoudi, G. S. Fast image and video denoising via nonlocal means of similar neighborhoods. *IEEE Signal processing letters*, 12(12):839–842, december 2005.

[Muñoz02] Muñoz, X. Image segmentation integrating colour, texture and boundary information. *PhD. Thesis, Universitat de Girona*, 2002.

[P. Charbonnier97] P. Charbonnier, G. A., L. Blanc-Feraud y Barlaud, M. Deterministic edge-preserving regularization in computed imaging. *IEEE Transactions on Image Processing*, (vol. 6(2)):pp.298–311, february 1997.

[P. Perona90] P. Perona, S. M. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[R. Adams94] R. Adams, L. B. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6 (16)):641–794, june 1994.

[R. González01] R. González, R. W. Digital image processing. *Prentice-Hall*, 2001.

[R. Jain95] R. Jain, B. S., R. Kasturi. Machine vision. *McGraw-Hill*, 1995.

[R. Marfil06] R. Marfil, A. B. J. R. F. S., L. Molina-Tanco. Pyramid segmentation algorithms revisited. *Pattern recognition*, 39:1430–1451, 2006.

[R.H. Chan05] R.H. Chan, M. N., C.W. Ho. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *IEEE Transactions on image processing*, 14(10):1479–1485, october 2005.

[Russ99] Russ, J. The image processing handbook. *CRC Press IEEE Press*, 3rd edition, 1999.

[S. Balasubramanian09] S. Balasubramanian, R. M. D. E. V. J., S. Kalishwaran. An efficient non-linear cascade filtering algorithm for removal of high density salt and pepper noise in image and video sequence. *International conference on control, automation, communication and energy conservation 2009*, págs. 1–6, june 2009.

[S.M. Smith97] S.M. Smith, J. B. Susan - a new approach to low level image processing. *International journal of computer vision*, 23(1):45–78, may 1997.

[S.Q. Yuan06] S.Q. Yuan, Y. T. Difference-type noise detector for adaptive median filter. *IEEE Electronic letters*, 42(8), april 2006.

[terHaar Romeny94] ter Haar Romeny, B. M. Geometry-driven diffusion in computer vision. *Kluwer Academic Publishers*, 1994.

[T.N. Pappas00] T.N. Pappas, R. S. Perceptual criteria for image quality evaluation. *Handbook of Image and Video Processing, Academic Press*, May 2000.

[Z. Wang99] Z. Wang, D. Z. Progressive switching median filter for the removal of impulse noise from highly corrupted images. *IEEE Transactions on circuits and systems-II: analog and digital signal processing*, 46(1):78–80, january 1999.

[Z. Wang02] Z. Wang, A. C. B. A universal image quality index. *IEEE Signal processing letters*, march 2002.