Compresión de Imágenes con Preservación de Características

Osslan Osiris Vergara Villegas[1], Raúl Pinto Elías[1] y Vianey Guadalupe Cruz Sánchez[1]

[1] Centro Nacional de Investigación y Desarrollo Tecnológico (cenidet)
Interior Internado Palmira s/n. Col. Palmira, P. C. 62490
Cuernavaca Morelos México
{osslan, rpinto, vianey}@cenidet.edu.mx

Resumen. En el presente artículo se muestra una solución al problema de inspección automática de piezas industriales utilizando imágenes provenientes de un proceso de compresión de imágenes con pérdidas con preservación de bordes. El codificador propuesto ofrece como ventaja adicional a la del ahorro de espacio de almacenamiento, una forma de manejar y preservar las características importantes de una imagen definidas por los bordes. La preservación de características se realiza con una modificación del algoritmo SPIHT. Dicha modificación se obtiene al cambiar el concepto de significancia de un píxel, donde la selección se realiza no sólo por magnitud sino por la pertenencia del píxel a un Mapa de Características de Interés (MCI). El mapa es obtenido con el detector SUSAN y después se realiza un mapeo de coeficientes al dominio wavelet. Finalmente, se muestran experimentos donde se demuestra la efectividad del codificador en tareas de inspección utilizando el software comercial "Vision Builder".

Palabras clave: Compresión de imágenes, preservación de bordes, transformación wavelet, SPIHT, inspección automática.

1 Introducción

En los últimos años se han realizado una gran cantidad de investigaciones para el estudio y diseño de algoritmos de compresión de imágenes, que básicamente se dividen en algoritmos con pérdida y sin pérdida de información. El principal interés de los investigadores está en el diseño de codificadores que den como resultado un gran factor de compresión y una buena calidad de las imágenes reconstruidas [1].

Una de las razones principales para diseñar dichos codificadores es el incremento en el uso de Internet y los dispositivos de comunicación móvil, en los que la entrega de la información necesita ser eficiente en el sentido de usar menos información para una buena representación con el objetivo de evitar saturación del ancho de banda y poder percibir y reconocer las imágenes lo más temprano posible.

Una rama importante de la compresión son los codificadores progresivos que buscan obtener la mejor calidad de la imagen a diferentes tasas de compresión [2].

J. C. Zavala-Díaz y M. A. Cruz-Chávez (Eds.): AGECOMP 2006, ISBN: 968-878-273-4, pp. 16-26, 2006.

El codificador progresivo ideal es aquel que permite reconstruir la imagen mostrando primero, las características importantes (preservadas) para una aplicación en particular como puede ser el entendimiento o reconocimiento de imágenes. Algunas de las características más importantes de las imágenes son los bordes, las texturas, etc. que son la clave para el éxito de diversas tareas de visión artificial.

La preservación de características se vuelve imperativa en áreas como la medicina, la industria textil, el sensado remoto, vigilancia en tiempo real y sistemas de identificación de huellas y rostros por citar algunos ejemplos. En los que el uso de las imágenes originales está reglamentado y restringido por lo que no se recomienda la pérdida de información relevante [3].

Preservar características, significa que la colocación, fuerza y forma de las características de una imagen no cambien aún después de la aplicación de un filtro general, por supuesto, pueden ocurrir diferencias naturales debido a cambios en el manejo de la resolución [4].

En la literatura existen algunos trabajos que abordan el problema de Compresión de Imágenes con Preservación de Características (CPIC). Los primeros intentos utilizaban esquemas llamados codificadores de regiones de interés (ROI´s) en los que la cantidad de bits era en gran parte usada en las partes definidas por las ROI y en menor cantidad en otras partes de la imagen [5]. Después, se propusieron transformaciones de dominio diseñadas para no actuar sobre las características deseadas (con el fin de conservarlas) como por ejemplo la Transformada Wavelet Discreta Basada en Regiones (TWDBR), en la que las diversas características son procesadas y codificadas de forma distinta para conservarlas [6].

El trabajo presentado en [2] se presentan dos codificadores progresivos diseñados para mejorar la claridad visual de los bordes de una imagen. Las posiciones de los bordes son capturadas con un detector de bordes y codificadas y transmitidas al receptor como parte de la cabecera de la imagen. Finalmente en [3] se muestra una metodología para preservar bordes usando un vector de características y para la etapa de cuantificación se utiliza EZW.

En este artículo se presenta una nueva metodología para el diseño de un codificador de imágenes con pérdidas que permite la preservación de los bordes basada en una modificación del algoritmo SPIHT.

2 Modelo Propuesto

Una imagen contiene diferentes características tales como bordes, texturas, detalles asociados a los bordes, etc. que exhiben la habilidad de la imagen para ofrecer información a las personas de los objetos contenidos en ella.

Para poder diseñar el codificador de imágenes con preservación de características es importante identificar las características que se quieren preservar, diseñar la tarea de procesamiento digital para extraer exitosamente las características de la imagen, y una vez obtenidas, se puede gastar una mayor cantidad de bits en las características de interés y sacrificar fidelidad o calidad en otras regiones de la imagen [7].

Las etapas para diseñar el compresor propuesto son las siguientes: a) Extracción de las Características De Interés (CDI) con SUSAN, b) Transformación de dominio con

una wavelet reversible, c) Mapeo de las CDI al dominio wavelet, d) Codificación de imágenes con el algoritmo SPIHT modificado y e) Decodificación de imágenes.

2.1 Extracción de las Características De Interés (CDI) con SUSAN

La primera etapa consiste en seleccionar las imágenes a comprimir y obtener el mapa de CDI, para esto, se utilizó el detector de bordes conocido como SUSAN (Smallest Univalue Segment Assimilating Nucleus) que es más robusto y efectivo que el detector Canny [8] en el sentido de que proporciona mejor localización de los bordes y mejor conectividad.

SUSAN utiliza una máscara predeterminada centrada en cada píxel de la imagen, con la que se aplica un conjunto de reglas locales para proporcionar las respuestas de los bordes. Dicha respuesta es procesada para ofrecer como salida el conjunto de bordes [9]. En resumen SUSAN realiza los siguientes tres procedimientos en cada píxel de la imagen:

1. Coloca una máscara circular alrededor del píxel en cuestión (el núcleo).
2. Calcula el número de píxeles dentro de la máscara circular que tienen un valor similar al brillo del núcleo.
3. Substrae el tamaño de USAN del umbral geométrico para producir una imagen de bordes.

La figura 1 muestra el ejemplo de las imágenes seleccionadas y sus correspondientes mapas de bordes.

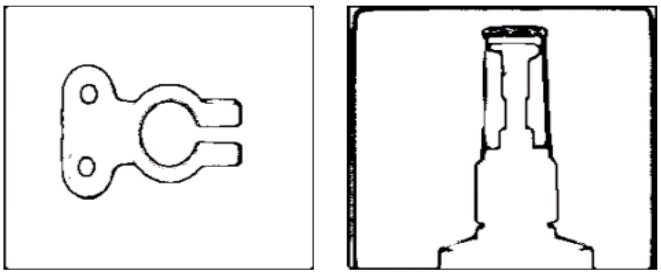

Fig. 1. Mapas de CDI (bordes). a) Abrazadera de una batería y b) Bote de Spray.

2.2 Transformación de Dominio con una Wavelet Reversible

Después de la extracción del mapa de CDI se aplica una transformación de dominio a la imagen original que ofrece una representación alternativa que puede revelar características de la imagen que en el dominio original eran difíciles de detectar. La transformación, persigue dos objetivos: a) reducir la correlación entre los coeficientes transformados y b) tomar ventaja de la compactación de energía para codificar solo una fracción de los coeficientes transformados sin producir demasiada distorsión [10].

La Transformada Wavelet Discreta (TWD) permite descomponer jerárquicamente una señal de entrada en una serie de señales de referencia de menor resolución y sus

señales de detalle asociadas [11]. La TWD de una imagen se obtiene convolucionando en los renglones y en las columnas un filtro pasa bajos (función de escalamiento _ o wavelet padre) y un filtro pasa altos (función wavelet _ o wavelet madre).

Para el presente artículo las imágenes son descompuestas un número log_2 *(tamaño de la imagen) – 1* de niveles como es definido en [12] y el filtro wavelet utilizado es el Biortogonal 2.2. En la figura 2 se muestra un ejemplo de una transformación wavelet a tres niveles aplicada sobre la imagen de la batería y sobre el bote de spray.

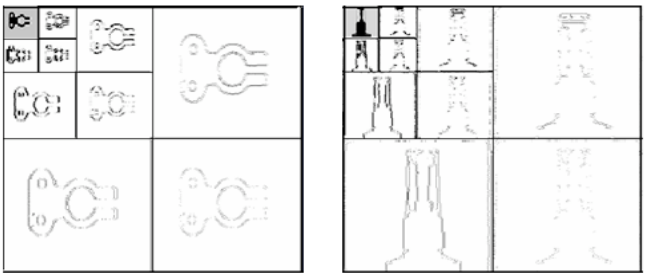

Fig. 2. Transformada wavelet discreta a tres niveles**.** a) Abrazadera de una batería y b) Bote de Spray.

2.3 Mapeo de las CDI al Dominio Wavelet

Es esta etapa se realiza un proceso de mapeo (correspondencia) de las posiciones de los píxeles del mapa de CDI obtenido con SUSAN a las posiciones de los píxeles en el dominio transformado. En una descomposición wavelet un coeficiente de escala *i* afecta un área de *2i x 2i* posiciones de la imagen original, por lo que existe una relación jerárquica entre los coeficientes que permite definir una estructura conocida como árbol de orientación espacial.

Un píxel en la subbanda wavelet más burda es padre de tres coeficientes en la misma posición en las subbandas de alta frecuencia en la misma escala. Cualquier otro coeficiente que no pertenece a la banda más baja tiene cuatro hijos, definidos con la ecuación 1.

$$hijos(x,y), s \neq 1 = \begin{cases} (2x-1, 2y-1) \\ (2x-1, 2y) \\ (2x, 2y-1) \\ (2x, 2y) \end{cases} \quad (1)$$

En resumen el proceso de mapeo es: a) Calcule el mapa de bordes con el detector SUSAN, b) Duplique las posición de los píxeles pertenecientes al mapa de CDI (2 x 2 como en la transformada wavelet), c) Realice un proceso de submuestreo, d) repita *b* y *c* un número *n* determinado de niveles y e) Marque los coeficientes descendientes como parte del mapa de CDI hasta que se alcanza el tamaño de la imagen original.

2.4 Codificación de Imágenes con el Algoritmo SPIHT Modificado

En 1996 Said y Pearlman presentaron una versión alternativa de los principios de operación del algoritmo Embedded Zerotree Wavelet (EZW) en la cual proponían una estructura de árbol diferente, a dicho algoritmo le llamaron SPIHT [13]. SPIHT permite definir la significancia de un píxel si su valor es mayor o igual a un umbral dado con lo que el coeficiente puede ser codificado.

Para poder comprimir una imagen preservando características se realizó una modificación al algoritmo SPIHT. Es decir, la significancia de un píxel no es solamente definida por su magnitud sino también por la posición del píxel con respecto al mapa obtenido en el dominio transformado. Para explicar como funciona la modificación propuesta se utiliza como entrada las imágenes de la figura 3a que representa una porción de una imagen (4 x 4) en el dominio wavelet y su respectivo mapa en la figura 3b.

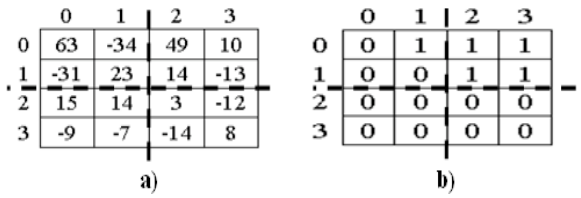

Fig. 3. Imágenes ejemplo. a) Imagen en el dominio wavelet, b) Mapa de características.

Para la modificación, primero, se deben inicializar los conjuntos básicos SPIHT definidos por: *O (i, j)* es el conjunto de descendientes directos de un nodo del árbol definido por la colocación de un píxel *(i, j)*. *D (i, j)* es el conjunto de descendientes de un nodo definidos por la colocación de un píxel. *L (i, j)* es el conjunto definido por *D (i, j) _ O (i, j)*. La siguiente explicación corresponde a los números en las entradas de la tabla 1 para el SPIHT modificado.

1. Sintonización inicial. El umbral inicial toma el valor de 32 obtenido del logaritmo base dos de 64. Las listas *LIS* (List of Insignificant Sets), *LIP* (List of Insignificant Pixels) y *LSP* (List of Significant Pixels) son inicializadas.
2. SPITH comienza codificando la significancia de los píxeles en *LIP*. La posición (0, 1) es significativa por que es mayor que el umbral y pertenece a una coordenada del mapa de referencia (marcada con un uno). A diferencia del SPITH original en el que los coeficientes significativos son (0, 0) y (0, 1).
3. Después de realizar la verificación a nivel de píxel con la lista *LIP*, SPIHT comienza a buscar en los diferentes conjuntos de píxeles siguiendo las entradas de *LIS*. La primera búsqueda la define *D (0, 1)* que representa un conjunto de cuatro coeficientes *{*(0, 2), (0, 3), (1, 2), (1, 3)*}*. Dado que *D (0, 1)* es significativo, entonces se debe verificar la significancia de los cuatro hijos y se encuentra que en su conjunto existen píxeles significativos y además pertenecientes al mapa. Finalmente (0, 1) es removido de la lista.
4. El mismo procedimiento se realiza sobre *D (1, 0)*, y dado que no es significativo no se realiza ninguna acción y se verifica el siguiente elemento.

5. *D (1,1)* no es significativo, no se realiza ninguna acción, por lo que la primera pasada para la etapa de ordenamiento termina.

Tabla 1. Proceso de codificación con SPITH modificado.

Comentario	Píxel o conjunto a buscar	Bit de salida	Acción	Lista de control
(1)				LIS = {(0,1)A,(1,0)A,(1,1)A} LIP = {(0,0),(0,1),(1,0),(1,1)} LSP = ϕ
(2)	(0,0)	0	ninguna	
	(0, 1)	1-	(0, 1) a LSP	LIP = {(0,0),(1,0),(1,1)} LSP = {(0,1)}
	(1, 0)	0	ninguna	
	(1,1)	0	ninguna	
(3)	D(0,1)	1	Buscar en descendientes	LIS = {(0,1)A,(1,0)A,(1,1)A}
	(0,2)	1+	(0,2) a LSP	LSP = {(0,1), (0,2)}
	(0,3)	0	(0,3) a LIP	LIP = {(0,0),(1,0),(1,1),(0,3)}
	(1,2)	0	(1,2) a LIP	LIP = {(0,0),(1,0),(1,1),(0,3),(1,2)}
	(1,3)	0	(1,3) a LIP	LIP = {(0,0),(1,0),(1,1),(0,3),(1,2), (1,3)}
				LIS = {(1,0)A,(1,1)A}
(4)	D(1,0)	0	ninguna	
(5)	D(1,1)	0	ninguna	

La etapa de refinamiento continua de la misma manera que lo hace SPITH en su versión original. Al final de la primera pasada el SPITH modificado gasta 13 bits comparado con los 14 bits gastados por el SPIHT original, el estado de la lista *LSP* es: *{*(0, 1), (0, 2)*}*, y para el SPIHT original *LSP* es: *{*(0, 0), (0, 1), (0, 2)*}*. Si se utiliza una tasa de compresión por ejemplo de 5 sobre la figura 3a entonces la lista *LSP* final con el SPITH original es: *{*(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (0, 3), (1, 2), (1, 3), (2, 0), (2, 1), (3, 0), (2, 3), (3, 2), (3, 3), (3, 1)*}* y el número de bits gastados es igual a 55. Mientras que con SPIHT modificado *LSP* = *{*(0, 1), (0, 2), (0, 3), (1, 2), (1, 3)*}* que corresponde a las posiciones preservadas y el número de bits gastados es de 56.

Al final el flujo de bits obtenido del algoritmo SPIHT modificado es codificado por entropía para obtener el archivo final, al flujo se le añade información acerca del tamaño de la imagen, nivel wavelet y filtro usado.

2.5 Decodificación de Imágenes

En esta etapa se aplica la decodificación por entropía seguida de la decodificación SPIHT y la Transformada Wavelet Discreta Inversa (TWDI) para obtener la imagen reconstruida final.

El resultado obtenido de la decodificación sobre la imagen 3a se muestra en la figura 4a para SPITH original y en 4b para SPITH modificado. En la imagen 4b se puede observar que a diferencia de la imagen 4a la cantidad de bits solamente fue gastada en las partes definidas por el mapa de la figura 3b, lo que demuestra que la modificación propuesta funciona de manera exitosa.

El error cuadrático medio (MSE) obtenido en la imagen con SPITH original es de 8.3125 y la relación señal a ruido pico (PSNR) 38.9335. Para el caso del SPITH modificado el MSE es de 403.0625 y el PSNR de 22.0771.

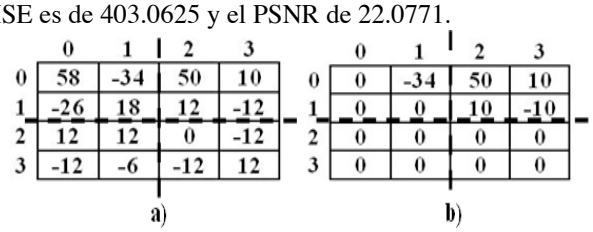

Fig. 4. Imágenes decodificadas. a) SPIHT Original, b) SPIHT modificado.

3 Experimentación y Resultados

Para la etapa de pruebas se realizó un proceso de inspección automática utilizando las imágenes provenientes del proceso de CIPC, el cuál se obtiene con el software comercial de National Instruments llamado "Vision Builder".

Vision Builder para inspección automática (IA) es un ambiente de desarrollo configurable para visión por computadora que no requiere programación [14]. El software puede ser usado para probar visualmente cuando un producto es ensamblado y manufacturado correctamente. De dicho software se utilizaron dos módulos de inspección y clasificación de piezas industriales que verifica criterios de calidad para clasificar piezas correspondientes a la clase de buena o mala calidad.

Se obtuvieron las imágenes del proceso de compresión/descompresión con una tasa de compresión de 0.1 y 0.5.

La primera prueba consiste en la verificación de la calidad de la abrazadera de una batería, donde lo primero que se realiza es la detección de la abrazadera en la imagen, después, se determina un sistema de referencia basado en la localización de la parte, se buscan los orificios circulares y se verifica el radio de la abrazadera, por último se mide la distancia (apertura) que existe entre los dos brazos.

La tabla 2 muestra los resultados obtenidos de la verificación de la abrazadera de batería. El primer renglón muestra el resultado con la imagen original, el segundo el resultado con la imagen comprimida a 0.5, el tercero con la imagen comprimida a 0.1 y el último renglón el resultado con una imagen a la que se le añadió una gran cantidad de ruido y a 0.5. En la figura 5 se muestran las imágenes reconstruidas.

Tabla 2. Resultados para la verificación de la abrazadera de batería.

Imagen	*Detección de la pieza*	*Sistema de coordenadas*	*Orificios circulares*	*Objeto detectado*	*Objeto detectado*	*Apertura*
Batería	PASA	PASA	Radio 59.22. PASA	# Objetos 1 PASA	# Objetos 2 PASA	Distancia: 28.32 pix
Batería 0.5	PASA	PASA	Radio 59.18. PASA	# Objetos 1 PASA	# Objetos 2 PASA	Distancia: 29.25 pix
Batería 0.1	PASA	PASA	Radio 59.47. PASA	# Objetos 1 PASA	# Objetos 2 PASA	Distancia: 29.41 pix
Batería con ruido	PASA	PASA	Radio 56.87. PASA	# Objetos 1 PASA	# Objetos 2 PASA	Distancia: 18.24 pix Fallo, distancia muy pequeña

De los resultados mostrados en la tabla 2 se puede observar que tanto con la imagen original como con las reconstruidas el proceso de inspección es exitoso excepto con la imagen comprimida con ruido. El fracaso se debe a la gran cantidad de ruido, el proceso falla en la última etapa de inspección donde la distancia entre los brazos es muy pequeña puesto que lo que mide como borde es una parte del ruido de la imagen, lo anterior sucede aún cuando los bordes fueron preservados exitosamente.

La segunda prueba consiste en la inspección de un envase plástico de spray que verifica lo siguiente: primero localiza el borde izquierdo de la imagen, enseguida se crea un sistema de referencia para después localizar los otros dos bordes del envase. Después mide la distancia hacia la izquierda y localiza los bordes de la derecha con su respectiva distancia, por último verifica la existencia del atomizador y de la tapa del envase.

La tabla 3 muestra los resultados obtenidos de la verificación del envase plástico de spray con la imagen original, y las imágenes comprimidas. El primer renglón muestra el resultado obtenido con la imagen original, el segundo el resultado con la imagen comprimida a 0.5, el tercero con la imagen comprimida a 0.1 y el último renglón muestra el resultado de una imagen a la que se le añadió una gran cantidad de ruido y a 0.5. En la figura 5 se muestran las imágenes reconstruidas.

Tabla 3. Resultados para la verificación del envase plástico de spray.

Imagen	*Localiza borde izquierdo*	*Sistema de referencia*	*Borde izquierdo*	*Dist. izquierda*	*Borde derecho*	*Dist. derecha*	*Presencia de atomizador*	*Presencia de tapa*
Spray	139.29 PASA	PASA	#Bordes = 2	10.53 PASA	# Bordes = 2	6.52 PASA	54.61 pix PASA	Intensidad = 22
Spray 0.5	142.20 PASA	PASA	# Bordes = 2	10.32 PASA	# Bordes = 2	7.39 PASA	55.70 pix PASA	Intensidad = 12
Spray 0.1	138.66 PASA	PASA	# Bordes = 2	12.32 PASA	# Bordes = 2	7.51 PASA	55.61 pix PASA	Intensidad = 16
Spray con ruido	Fallo, muchos bordes	Fallo, punto no disponible	Fallo, sistema de coordenadas	Fallo, punto no disponible	Fallo, sistema de coordenadas	Fallo, punto no disponible	Fallo, sistema de coordenadas	Fallo, sistema de coordenadas

En esta prueba otra vez se obtiene una falla en la inspección con la imagen en la que se añadió ruido. El proceso de inspección fracasa desde la primera etapa por lo que las restantes etapas también fallan. Con el resto de las imágenes el proceso es exitoso al igual que en la prueba de la abrazadera de la batería, con lo que queda demostrada la efectividad del proceso de compresión de imágenes con preservación de características, en este caso aplicado a dos procesos de inspección de calidad.

Para medir de manera objetiva el desempeño del codificador se proponen cinco medidas: Error Cuadrático Medio (Mean square Error MSE), Relación Señal a Ruido Pico (Peak Signal-to-Noise Ratio PSNR), Norma 2, Norma de Frobenius y el factor de compresión (FC).

La tabla 4 muestra las medidas de error obtenidas para todas las imágenes de prueba utilizadas para el proceso de inspección.

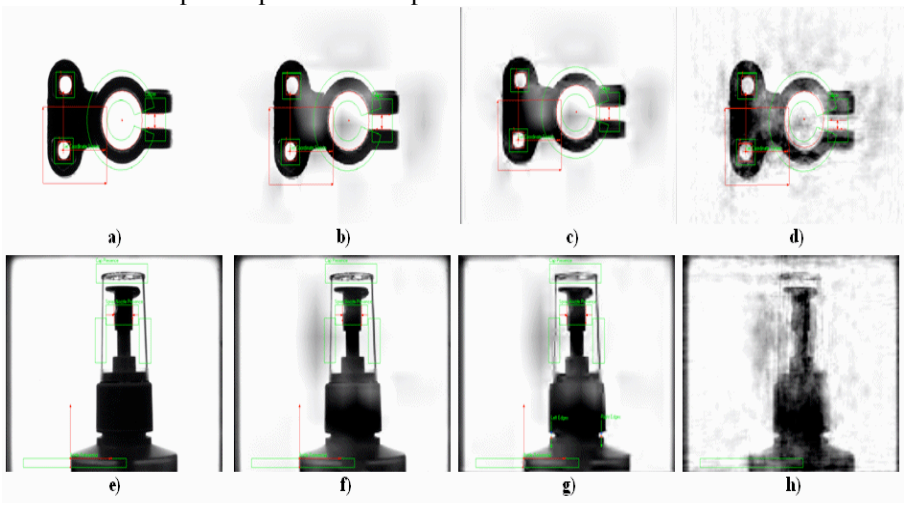

Fig. 5. Imágenes originales y reconstruidas. a) Abrazadera de batería original, b) Batería a 0.5, c) Batería a 0.1, d) Batería con ruido, e) Envase de spray original, f) Spray a 0.5, g) Spray a 0.1 y f) Spray con ruido.

Tabla 4. Medidas de error para la abrazadera de batería y el envase de spray.

Imagen	*FC*	*MSE*	*PSNR*	*F*	*N2*
Batería 0.5	18.179:1	615.2355	20.2404	0.1055	0.0914
Batería 0.1	85.780:1	740.257	19.437	0.1157	0.0923
Batería con ruido	18.719:1	1041.962	17.9523	0.1373	0.09558
Spray 0.5	17.296:1	353.8215	22.64	0.0845	0.0697
Spray 0.1	86.459:1	467.2724	21.4351	0.0971	0.0724
Spray con ruido	19.793:1	1091.2189	17.7517	0.1485	0.0980

Como se puede observar en la tabla 4 las medidas de error pueden ser malas, pero como se demostró en la pruebas el proceso de inspección se realiza de forma satisfactoria. Los factores de compresión son buenos comparados con otros algoritmo similares con la ventaja de que aquí no se envía ningún tipo de información del lado del codificador. La importancia del método propuesto no es mejorar las medidas de error sino poder realizar el proceso de inspección de forma exitosa.

4 Conclusiones y Trabajos Futuros

Se mostró un codificador de imágenes con pérdidas basado en la transformada wavelet que permite la preservación de bordes; el codificador puede ser usado si se necesita un reconocimiento temprano de las imágenes o en ambientes tales como la medicina en la que existen leyes para el uso de las imágenes y la preservación de las características es un proceso imperativo.

Para diseñar el codificador se obtiene la imagen de bordes usando SUSAN, después se realiza una transformación de dominio y un proceso de mapeo de los puntos del dominio original al dominio wavelet. Finalmente, las imágenes son codificadas y decodificadas con el algoritmo SPIHT modificado.

Con las pruebas y los resultados presentados se demostró la habilidad del codificador para reconstruir imágenes gastando más bits y dando más calidad a aquellas partes importantes de la imagen. Las medidas objetivas no ofrecen una buena información acerca de la bondad del método, y se debe recordar que el objetivo es la preservación de características aún a tasas de compresión muy bajas.

A futuro se espera trabajar en la búsqueda de un método para reducir los artefactos presentes en las imágenes y probar otras transformaciones como por ejemplo la transformada contourlet para resolver el problema de direccionalidad de los bordes y la geometría de la imagen.

Referencias

1. A. Mertins, Image Compression Via Edge-Based Wavelet Transform, *Optical Engineering*, vol. 38, no. 6, pp. 991 – 100, junio de 1999.
2. D. Schilling y P. C. Cosman, Preserving Step Edges in Low Bit Rate Progressive Image Compression, *IEEE Transactions on Image Processing*, vol. 12, no. 12, pp. 1473 – 1484, deciembre de 2003.
3. K. R. Namuduri y V. N. Ramaswamy, Feature Preserving Image Compression, *Pattern Recognition Letters*, vol. 24, no. 15, pp. 2767 – 2776, noviembre de 2003.
4. G. Craciun G., M. Jiang, D. Thompson y R. Machiraju, Spatial Domain Wavelet Design for Feature Preservation in Computational Data Sets, *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 2, pp. 149 – 159, abril de 2005.
5. M. Kunt, A. Ikonomopoulos y M. Kocher, Second Generation Image Coding Techniques, *Proceedings of the IEEE*, vol. 73, no. 4, pp. 549 – 574, abril de 1985.
6. H. J. Barnard, Image and Video Coding Using a Wavelet Decomposition, Tesis doctoral, Universidad Tecnológica de Delft, Departamento de ingeniería eléctrica, Grupo de teoría de la información, Holanda, 1994.
7. D. Schilling y P. C. Cosman, Feature-Preserving Image Coding for Very Low Bit Rates, *Proceedings of the IEEE Data Compression Conference (DCC)*, Snowbird, Utah, USA, pp. 103 – 112, marzo de 2001.
8. J. F. Canny, A Computational Approach to Edge Detection, I*EEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679 – 698, noviembre de 1986.
9. S. M. Smith y J. M. Brady, SUSAN - A New Approach to Low Level Image Processing, *International Journal of Computer Vision*, vol. 23, no 1, pp. 45 – 78, mayo de 1997.
10. Ôktem R., Transform Domain Algorithms for Image Compression and Denoising, Tesis de maestría, Universida Tecnológica de Tampere, Finlandia, mayo de 2000.
11. J. Villaseñor, B. Belzer y J. Lia, Wavelet Filter Evaluation for Image Compression, *IEEE Transactions on Image processing*, vol. 4, no. 8, pp. 1053 – 1060, agosto de 1995.
12. Information and Communication Theory Group, Image and Video Compression Learning Tool, VcDemo 5.03, Universidad Tecnológica de Delft, Holanda, septiembre de 2004.
13 A. Said y W. Pearlman, A New Fast and Efficient Image Codec Based on Set Partitioning In Hierarchical Trees, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243 – 250, junio de 1996.
14. National Instruments, Vision Builder for Automated Inspection, available [on line]: http://www.ni.com/vision/vbai.htm, U.S.A., 2005.