Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática industrial 15 (2018) 46-57

www.revista-riai.org

Sistema de Visión Subacuático Inalámbrico Usando un Algoritmo de Compresión Progresivo con Región de Interés

Eduardo M.Rubino[a,*], D. Centelles, Jorge Sales, José V.Martí, Raúl Marín, Alberto J. Alvares, Pedro J.Sanz

[a]*Departamento de Ingeniería y Ciencia de los Computadores, Universidad Jaume I, Castellón de la Plana, España.*

Resumen

La creciente demanda en todo el mundo de sistemas de intervención submarina en diversos dominios de aplicación requiere sistemas más versátiles y económicos. Empleando un sistema de comunicación inalámbrica, los robots semiautónomos supervisados disponen de libertad de movimientos y, al mismo tiempo, permiten al operador obtener información visual y supervisar la intervención. Por otro lado, la velocidad de transmisión de datos típica de los canales inalámbricos submarinos es, en general, muy limitada, siendo necesaria la aplicación de técnicas de compresión avanzadas. En este artículo se presenta principalmente el algoritmo DEBT (Depth Embedded Block Tree) para la compresión progresiva de imágenes con región de interés (ROI). Los resultados demuestran ventajas con respecto a otros algoritmos de compresión, y la posibilidad de ejecución del algoritmo en tiempo real en ordenadores embebidos de bajo consumo basados en ARM.

Palabras Clave:
Compresión de Imagen, Transmisión de Datos, Sistemas de Comunicación, Teleoperación

Underwater Wireless Vision System Using Progressive Image Compression and Region of Interest

Abstract

The increasing demand for underwater robotic intervention systems around the world in several application domains requires more versatile and inexpensive systems. By using a wireless communication system, supervised semi-autonomous robots have freedom of movement and, at the same time, allows the operator to get camera feedback and supervise the intervention. On the other hand, the typical data rate of the wireless submarine channels is generally very limited, requiring the application of advanced compression techniques. In this paper we present the DEBT (Depth Embedded Block Tree) algorithm for the progressive compression of images with region of interest (ROI). The results demonstrate advantages with other compression algorithms, and the possibility of executing the algorithm in real time on ARM-based embedded low-power computers.

Keywords:
Image Compression, Data Transmission, Communication Systems, Teleoperation

1. Introducción

En el contexto del proyecto de investigación MERBOTS (DPI2014-57746-C3) (http://www.irs.uji.es/merbots/), un proyecto coordinado de tres años financiado por el Gobierno Español a lo largo del periodo 2015-2017 (Sanz et al., 2015), uno de los objetivos consiste en la obtención de un sistema de comunicaciones inalámbricas que proporcione libertad de movimientos a un robot submarino y, al mismo tiempo, permita al operador obtener realimentación y supervisar la intervención (ver Fig. 1). El sistema robótico en desarrollo ayudará a los arqueólogos en tareas detalladas de monitorización, caracterización, estudio, reconstrucción y preservación de emplazamientos arqueológicos, siempre bajo la continua supervisión de un experto humano.

*Autor en correspondencia: rubino@uji.es
To cite this article: Eduardo M.Rubino, D.Centelles, Jorge Sales, José V.Martí, Raúl Marín, Alberto J. Alvares, Pedro J.Sanz. 2018. Underwater wire-less Vision System using progressive Image Compression and Region of Interest. RIAI: Revista Iberoamericana de Automática e Informática industrial 15 (2018) 46-57. https://doi.org/10.4995/riai.2017.8953

Como puede apreciarse en la imagen conceptual del proyecto, Fig. 1, el sistema conectará la superficie con el vehículo de intervención I-AUV, permitiendo acceso a la red a través de un canal WiFi de 5 GHz (<10Km).

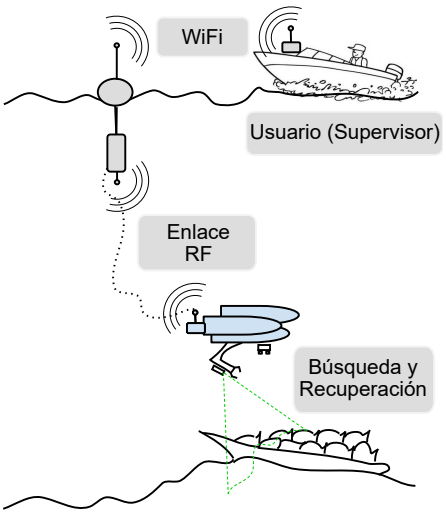

Figura 1: Imagen conceptual de búsqueda y recuperación en el contexto de la arqueología submarina. Un enlace RF proporciona realimentación al usuario que supervisa la intervención.

MERBOTS representa la continuación natural de los proyectos de investigación nacionales e internacionales previos en el campo de la intervención robótica submarina (i.e. RAUVI (Sanz et al., 2010), TRITON (Sanz et al., 2013a), EU FP7 TRIDENT (Sanz et al., 2013b)). Uno de los objetivos del proyecto MERBOTS consiste en proporcionar tecnologías de comunicación diferentes que puedan emplearse para la operación de un vehículo sin ninguna conexión física con los operadores de la superficie, quienes supervisan y controlan las tareas de la intervención.

El presente artículo describe el estado actual del sistema inalámbrico de visión submarina, que es capaz de transmitir tanto datos de telemetría como imágenes comprimidas de baja resolución, permitiendo realizar misiones de intervención cooperativas.

El uso de enlaces RF submarinos es un ejemplo destacado de escenario con ancho de banda reducido. Así mismo, las técnicas aquí descritas pueden también ser aplicadas a otros escenarios similares, como por ejemplo rescate en entornos peligrosos (e.g. radioactividad). También pueden servir para mejorar considerablemente la usabilidad en escenarios con ancho de banda normal, especialmente cuando se trata de búsqueda remota mediante imagen, usando versiones de baja calidad o baja resolución de la imagen original y solicitando mayor nivel de detalle únicamente cuando el objetivo ha sido localizado.

En este artículo se presenta el sistema de compresión de imágenes progresivo Depth Embedded Block Tree (DEBT), el cual ha sido específicamente diseñado para enlaces de bajo ancho de banda, especialmente el control supervisado de vehículos submarinos de forma inalámbrica. El sistema de compresión se caracteriza por los siguientes aspectos:

- Usa una técnica de compresión progresiva diseñada específicamente para reducir la latencia, teniendo en cuenta que reducir el retardo es fundamental en el control remoto de un sistema robótico, especialmente cuando dispone de un brazo manipulador. Igualmente, se pueden almacenar localmente (i.e. en el propio robot) imágenes de alta definición, incluso sin pérdidas de detalle, de forma que las imágenes originales pueden ser recuperadas posteriormente para ser archivadas y estudiadas.
- El algoritmo de compresión se ha implementado con el objetivo de ser utilizado en escenarios con un ancho de banda muy reducido. Es aplicable tanto a comunicaciones robóticas submarinas como a otras aplicaciones robóticas.
- Se ha realizado una implementación del algoritmo de compresión capaz de comprimir imágenes a un ritmo mayor que 30 fps en computadores empotrados de bajo consumo (e.g. Raspberry Pi 2 B).
- El sistema es capaz de enviar imágenes utilizables empleando solamente unos pocos cientos de bytes por fotograma. El flujo de datos puede simplemente truncarse para enviar una versión de la imagen de menor calidad.
- El usuario puede seleccionar una o más Regiones de Interés (ROI) en las que se obtendrá mayor calidad de imagen. El presente artículo explica cómo se ha implementado esta técnica.
- El algoritmo de compresión fue comprobado con una batería de imágenes submarinas con el objetivo de ajustar los parámetros de compresión para misiones de intervención submarina.

2. El Dominio de la Intervención Robótica

En las aplicaciones robóticas y, en particular, en los Vehículos Subacuáticos Autónomos de Intervención (I-AUV), una de las principales fuentes de información son las cámaras que proporcionan imágenes de la escena en tiempo real (Carreras et al., 2012). En un sistema supervisado, estas imágenes deben llegar al operador con la menor latencia y con la máxima calidad posible para que permitan interactuar con el sistema y ajustar la ejecución de la tarea de forma apropiada. A modo de ejemplo, este tipo de control ha sido experimentado en el proyecto FP7 TRIDENT, para realizar un agarre autónomo guiado visualmente en el mar (Prats et al., 2013).

Además, las comunicaciones son un subsistema crucial en cualquier aplicación robótica, especialmente las que permiten al usuario interactuar remotamente con el sistema. Por ello, la compresión y transmisión de imágenes es necesaria para enviar la información requerida con la menor latencia y sin comprometer la red y todo el sistema.

Aunque estudios recientes demuestran que, utilizando los métodos de modulación más eficientes, es posible transmitir vídeo a través de un canal subacuático usando señales acústicas (Pelekanakis et al., 2003; Ribas et al., 2010) y *Blue Light* (Farr et al., 2010), ninguno de los dos tipos de señal es capaz de pasar a través de objetos sólidos que podrían estar en la línea

de visión de los transceptores inalámbricos. Además, el rendimiento de estos métodos depende en gran medida de las características del escenario submarino y del tipo de canal. Por un lado, los sistemas acústicos se ven muy afectados debido a la multi-trayectoria si el enlace es horizontal, y también por el ruido acústico originado por la actividad humana o el ruido de las olas del mar, animales, etc. Las frecuencias utilizadas en sistemas acústicos están entre 8 y 155 KHz (Stojanovic and Preisig, 2009), lo que hace muy difícil alcanzar altas velocidades de datos. Por otro lado, los métodos de comunicación basados en la luz visible solo funcionan bien en aguas muy claras y se ven muy afectados por la dispersión, sufren atenuación por absorción y por lo general necesitan una alineación precisa.

Por otro lado, las soluciones basadas en RF no se ven tan afectadas por los problemas típicos de los métodos acústicos (multi-path, variabilidad en el retardo, etc.) y ópticos (alineación, visibilidad), siendo mucho más económicas, y ofreciendo un retardo constante, el cual es necesario para aplicaciones futuras, como por ejemplo robótica cooperativa. Además, las señales RF pueden propagarse más fácilmente de un medio a otro, permitiendo el establecimiento de un enlace de comunicación con un transductor submarino desde la superficie. Por otro lado, resulta también más sencillo implementar un enlace *full-duplex* utilizando varios canales de RF que empleando la técnica acústica. No obstante, el principal problema del uso de RF es la alta atenuación que sufre en el ambiente acuático, especialmente en agua salada. Sin embargo, diferentes estudios (Zhang and Meng, 2012; Siegel and King, 1973; Shaw et al., 2006) y empresas como WFS (*Wireless For Subsea*) indican que, con las antenas necesarias, a frecuencias más bajas, y utilizando los mejores métodos de codificación y modulación, es posible establecer un enlace de comunicación hasta varias decenas de metros a través del agua (incluso en agua salada).

Es necesario aclarar que en este trabajo se ha decidido utilizar, para un primer prototipo, la técnica basada en RF por ser la más sencilla y económica de implementar. Además, no sería viable la utilización de un modem acústico en un tanque de agua dulce como el que se dispone en un laboratorio pequeño, debido a los problemas que ocasionarían en el enlace los efectos por el multi-path. El presente trabajo no pretende mejorar los medios físicos de comunicación para mejores enlaces inalámbricos submarinos, sino el diseño de un sistema unificado que utiliza módulos bien integrados (es decir, compresión y transmisión), diseñando un protocolo de nivel superior (i.e. basado en el sistema ROS) para la compresión y transmisión de imágenes con sistemas robóticos submarinos utilizando para ello cualquiera de las tres técnicas de transmisión mencionadas: sonido, luz o RF. Es decir, el objetivo de este trabajo es mostrar las ventajas del algoritmo de compresión DEBT y la viabilidad de un protocolo de transporte de imágenes para sistemas robóticos submarinos que haga buen uso de las ventajas de este algoritmo.

La aplicación de los algoritmos de compresión progresivos más avanzados permite velocidades de transmisión de imágenes de varios fotogramas por segundo con una baja latencia.

En el sistema propuesto se demuestra una técnica progresiva de compresión de imágenes parametrizable dinámicamente en tamaño, calidad, y Región de Interés (ROI), la cual está resultando de gran auxilio en la implementación de controles remotos submarinos de forma inalámbrica. El disponer del diseño propio del algoritmo de compresión permite una mejor adaptabilidad a la necesidad de la aplicación. Así mismo, se están estudiando mejoras del algoritmo para descubrir, de forma automática y sin aumentar el retardo, escenarios y objetos de interés. Los esfuerzos actuales se están desarrollando en esta línea de trabajo.

3. Descripción General del Sistema

El sistema de transmisión de RF se encuentra en una fase temprana de desarrollo y, para fines de evaluación del algoritmo de compresión y del protocolo de transporte, se han utilizado módulos UHF de baja potencia en un entorno controlado de agua dulce. Para ser fieles a las velocidades de transmisión típicas en canales subacuáticos, la velocidad de transmisión configurada en los módulos de RF utilizados en los experimentos no supera los 9600 baudios. Los trabajos adicionales se concentrarán en conseguir distancias más largas, incluso en agua salada, utilizando módems acústicos avanzados, como los S2CR 18/34 de EvoLogics, capaces de transmitir hasta 13.9 kbps, o módulos de RF de muy baja frecuencia, y a potencias más altas, como los Seatooth S100 de WFS (Wireless For Subsea), capaces de establecer un enlace a distancias de hasta 7 m y a velocidades de hasta 2.4 kbps en agua salada.

La Fig. 2 muestra un prototipo del sistema basado en RF que podría ser usado en una plataforma tipo OpenROV. Como se puede observar, el sistema tiene dos partes principales: (1) el lado del sensor, y (2) el operador. En el lado del vehículo, los circuitos desarrollados se instalan en una versión de mayor tamaño del cilindro estanco usado en la plataforma OpenROV. Incluye una tarjeta Raspberry PI que ejecuta Linux, una cámara de adquisición de vídeo, la compresión, el ROI y los módulos de capa de transporte. Todo ello se conecta a una placa Arduino que controla un transmisor RF de la empresa Radiometrix. La Raspberry Pi ejecuta una versión adaptada del sistema de control usado en OpenROV. Aunque en la figura sólo se muestra la electrónica correspondiente al sistema de comunicación, para el control de los motores usados en OpenROV sería suficiente con conectar una Arduino extra tipo Mega a la Raspberry Pi.

En el lado del usuario, el receptor RF submarino se conecta al ordenador del usuario a través de un puerto USB y proporciona una interfaz de usuario que permite al operador obtener las imágenes comprimidas, seleccionar la región de interés correspondiente para una inspección adicional y configurar el grado de compresión de las imágenes. La interfaz de usuario está basada en el software de la plataforma OpenROV, donde el servidor Web se ejecuta en el propio ordenador embebido.

4. Compresión de imágenes

La compresión es una transformación que se aplica a una imagen con el objetivo de reducir su tamaño tanto como sea posible y, de esta manera, poder almacenar mayor número de imágenes y transmitirlas con una mayor tasa de transferencia. Hay una clara distinción entre compresión sin pérdidas y compresión con pérdidas. En la compresión sin pérdidas, la imagen descomprimida será una copia exacta a la imagen original, mientras que una compresión con pérdidas ofrecerá solo una aproximación de la imagen original al descomprimir. Las

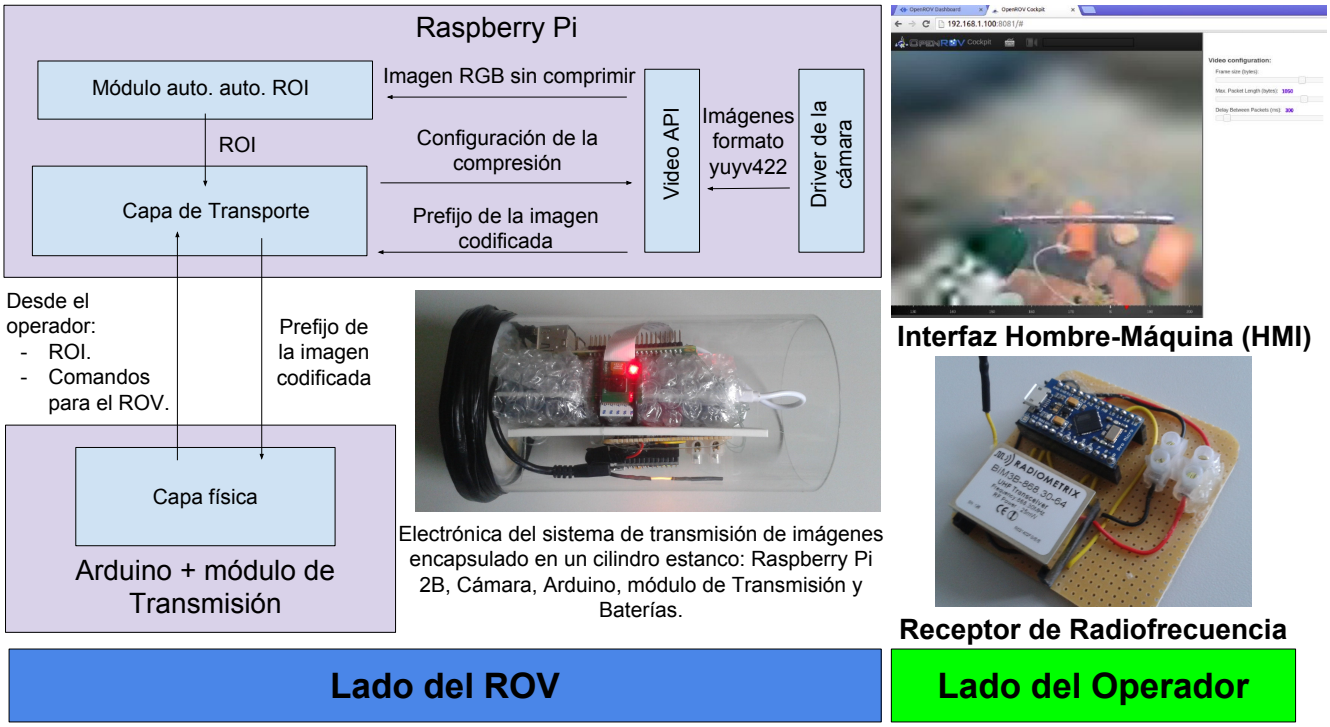

Figura 2: Arquitectura general del sistema: (izquierda) Lado del sensor instalado en la plataforma OpenROV, incluye la Raspberry Pi para el módulo de compresión, el ROI y el protocolo de red, así como el transceptor de RF físico controlado a través de una placa Arduino. (Derecha) Interfaz de usuario que permite al usuario monitorizar la información comprimida de la cámara, así como la especificación de regiones de interés por parte del usuario.

imágenes digitales tienen 3 componentes de color, lo cual significa que lo que nosotros percibimos como una imagen de un color está, de hecho, (sin pérdida de generalidad) compuesto por un canal de luminancia (una versión de blanco y negro de la imagen de color) y dos canales de diferencia de color (que normalmente pueden ser submuestreados sin demasiada pérdida).

En contraposición con la compresión de vídeo, la cual tiene la ventaja de tener una alta correlación temporal entre *frames* adyacentes en una secuencia de vídeo, y de crear *intra-frames* que son dependientes de *frames* previos, la compresión de imagen crea exclusivamente *intra-frames*, los cuales son *frames* comprimidos de forma independiente. Los *intra-frames* tienen la ventaja de ser capaces de adaptarse rápidamente a condiciones cambiantes en el canal de comunicación, así como una mayor flexibilidad para cambiar dinámicamente los parámetros de velocidad de fotogramas y calidad, que son de gran importancia en comunicaciones de bajo ancho de banda y baja latencia.

La compresión en general y la compresión de imágenes en particular es una tarea muy específica de la aplicación, con muchas compensaciones disponibles y muchos algoritmos diferentes que tratan de maximizar (o minimizar) algunos criterios de diseño. La mayoría de los algoritmos de compresión de imágenes son algoritmos con pérdidas diseñados con el único propósito de minimizar el tamaño resultante de la imagen con consideraciones mínimas a otras restricciones, además de que, normalmente, es necesaria la totalidad de los datos comprimidos para ser capaz de descomprimir la imagen. Un ejemplo primordial de esta clase de algoritmos es el algoritmo JPEG (*Joint Photographic Experts Group*) que es un estándar de facto pero que funciona muy mal cuando se requiere un nivel alto de compresión.

La compresión de imagen progresiva o integrada es tal que es trivial y barato en términos de potencia de procesamiento (no hay necesidad de descomprimir la imagen) suministrar una imagen que presente una resolución inferior o constituya una aproximación de calidad inferior de la imagen original. Preferiblemente, la imagen comprimida podría ser simplemente truncada en cualquier punto, produciendo una imagen con una resolución más baja o una versión de calidad inferior de la imagen original (en este sentido, los *streams* sin pérdidas progresivos pueden llegar a ser con pérdidas al truncar la imagen comprimida). En el caso de las imágenes en color, también podrían ser preparadas de tal manera que se pudiese obtener una versión monocromática de la misma con las mismas características progresivas que se han citado anteriormente.

5. El algoritmo Depth Embedded Block Tree (DEBT)

El algoritmo de compresión ha sido diseñado con el objetivo de mejorar la velocidad de compresión, así como permitir el uso de canales de comunicación de bajo ancho de banda, como los que podemos encontrar en entornos submarinos. En términos de velocidad, se toma como patrón de referencia el JPEG original (W. B. Pennebaker and J. L. Mitchell, 1992). Así mismo, el alto grado de compresión necesario para este tipo de aplicaciones se inspira en el más reciente algoritmo JPEG-2000 (Taubman and Marcellin, 2001).

Disponer de semejante algoritmo nos permite hacer un diseño mas afinado de sistemas telerobóticos submarinos, espe-

cialmente cuando se trata de comunicaciones inalámbricas con módems acústicos y/o radio frecuencia.

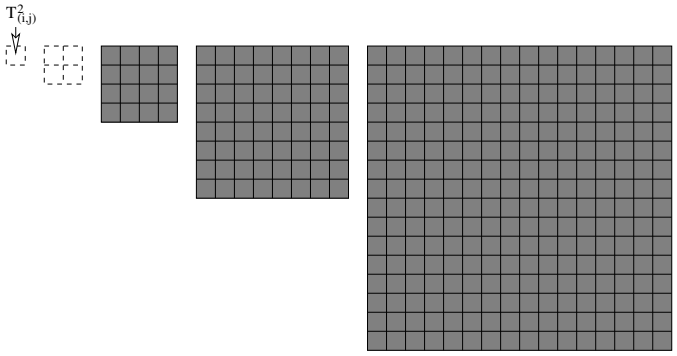

Figura 3: Árbol de profundidad 2

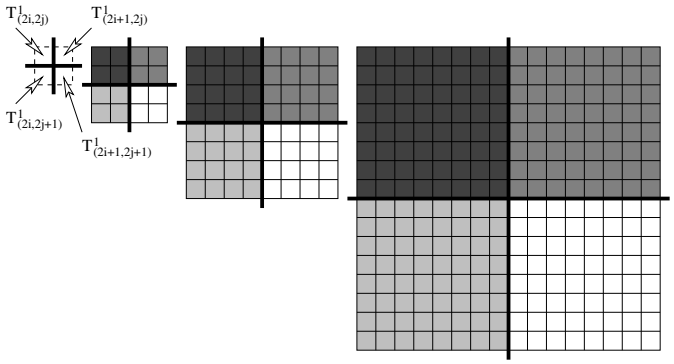

Figura 4: Partición de un árbol de profundidad 2 en cuatro árboles de profundidad 1

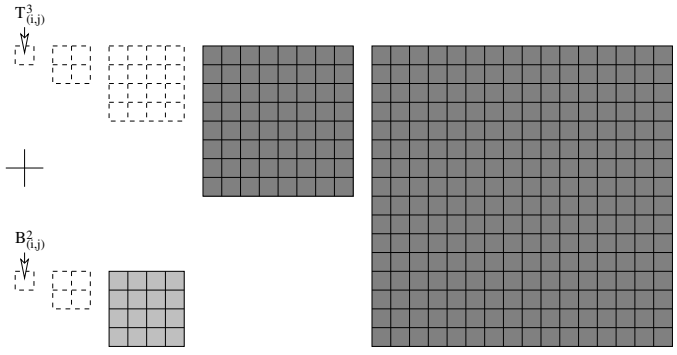

Figura 5: Descomposición de un árbol de profundidad 2 en un bloque de profundidad 2 y un árbol de profundidad 3

Este nuevo algoritmo de compresión usa como principales estructuras de datos árboles y bloques de profundidad variable, las cuales agrupan coeficientes de magnitud similar, de manera que la información es transmitida de una manera más eficiente con el estudio de las correspondientes correlaciones de coeficientes.

Los árboles y bloques tienen asociado un valor de profundidad, el cual indica el primer nivel piramidal donde se encuentran dichos coeficientes. Este tipo de estructura de datos permite que el algoritmo sea adecuado para escenarios con bajo ancho de banda. Las figuras 4 y 5 muestran la partición y descomposición, respectivamente, de un árbol de profundidad 2 (ver Fig. 3). De hecho, el algoritmo desarrollado unifica en el mismo sistema aspectos localizados de manera discreta en sistemas de compresión previos, como por ejemplo SPIHT (Set Partitioning in Hierarchical Trees) (Said and Pearlman, 1996), SPECK (Set Partioning Embedded Block Coder) (Pearlman et al., 2004), HBC (Hybrid Block Coder) (Wheeler and Pearlman, 2000), y WBTC (Wavelet Block Tree Coding) (Moinuddin and Khan, 2006), permitiendo, de una forma paramétrica, ajustar las condiciones de trabajo y adaptarse mejor al medio de comunicación existente.

Los pasos que sigue el algoritmo de compresión son los siguientes:

1. Transformada Wavelet - Modifica la imagen utilizando una transformada wavelet de N niveles. En el momento de la escritura del presente artículo se dispone de la implementación de las transformadas enteras biortogonales simétricas 5/3, 9/3, 9/7, y 13/7, construidas a partir de las funciones de escala interpoladas de Deslauries-Dubuc (Calderbank et al., 1998), así como la popular transformada biortogonal simétrica real CDF-9/7 Calderbank et al. (1998).
2. Agrupación de coeficientes en árboles, los cuales o bien se particionan en otros árboles de menor profundidad (ver Fig. 4), o bien se descomponen en bloques de la misma profundidad y un árbol de mayor profundidad (ver Fig. 5). Esta organización de las estructuras de coeficientes permite explorar de una forma más eficiente y rápida las correlaciones dentro y entre bandas. Los bloques siempre se particionan en bloques de menor profundidad.
3. Se usa una distribución laplaciana para los coeficientes de alta frecuencia (HL, LH y HH) y así obtenemos los valores de decrecimiento de distorsión para los bits de significancia y de refinamiento. Futuros trabajos se centrarán en buscar una PDF más ajustada a los coeficientes, y así obtener una tabla de pesos para los bitplanes más optimizada.
4. La asignación del decrecimiento de distorsión a cada par (*bitplane*, *subband*) para todos los coeficientes de significancia y de refinamiento indica cuál es el más importante para ser enviado, sirviendo asimismo como mecanismo de ubicación de bits. En general, la mayoría de DWT (Discrete Wavelet Transform) no son conservativas respecto de la energía (no son ortogonales), cada *subband* contribuye en diferente medida a la distorsión total. La ganancia de cada *subband* deberá aplicarse como factor de ajuste a todos los valores de significancia y refinamiento de cada *subband*. Dichos factores de ajuste pueden ser calculados como una función de su respectivo filtro de reconstrucción de acuerdo con (Usevitch, 1996).
5. Se recorre la lista anterior en orden decreciente de valores, produciendo como salida la información de significancia y refinamiento y manteniendo la información de particionamiento y descomposición (direccionamiento) de los conjuntos (árboles y bloques). Todo el almacenamiento se realiza en una zona de memoria de tamaño

fijo, cuya dimensión depende del tamaño de la imagen. En general, para una imagen de $N \times M$ pixels de 8 bits, el algoritmo DEBT necesita una matriz de $N \times M$ elementos de 16 bits para los coeficientes de la DWT, otra matriz de $N \times M$ elementos de 16 bits para la gestión de los coeficientes y una matriz de $N/2 \times M/2$ elementos de 32 bits para la gestión de los conjuntos (árboles y bloques).

6. Imágenes de prueba bajo el agua

Con el fin de probar y ajustar algunos parámetros del algoritmo para imágenes submarinas se ha utilizado un conjunto de 1258 imágenes submarinas en escala de grises del *Australian Center for Field Robotics* (`http://marine.acfr.usyd.edu.au/datasets/data/TasCPC/TasCPC\_RM.tar.gz`). Se ha realizado una comparativa del rendimiento por *bitrate* con el algoritmo JPEG2000 usando la transformada wavelet CDT-9/7 y 6 niveles de descomposición. Se ha mostrado gráficamente la diferencia del PSNR (Peak Signal-to-Noise Ratio) entre los algoritmos DEBT y JPEG2000 para 4 tasas diferentes (0.0005, 0.001, 0.005, y 0.01). Todas las imágenes tienen una resolución de 1360×1024 píxeles y fueron numeradas en órden léxico desde 1 hasta 1258.

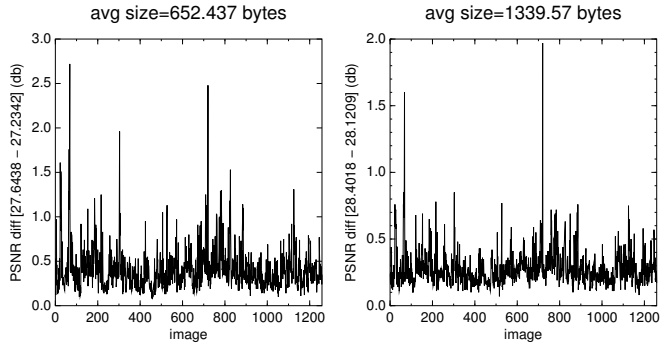

Figura 6: Imágenes TasCPC para ratios 0.0005 y 0.001

La Fig. 6 muestra la diferencia de la compresión en dB para todas las imágenes para las tasas 0.0005 y 0.001. Claramente el algoritmo DEBT funciona mejor para todas las imágenes sin excepción, para estas tasas más bajas por un margen bastante significativo, alcanzando una diferencia de casi 3 dB en el caso 0.0005 y casi 2 dB en el caso 0.001 para algunas imágenes en este *dataset*. En promedio, la ganancia de DEBT sobre JPEG2000 es de alrededor de 0.40 y 0.28 dB para las tasas de 0.0005 y 0.001, respectivamente.

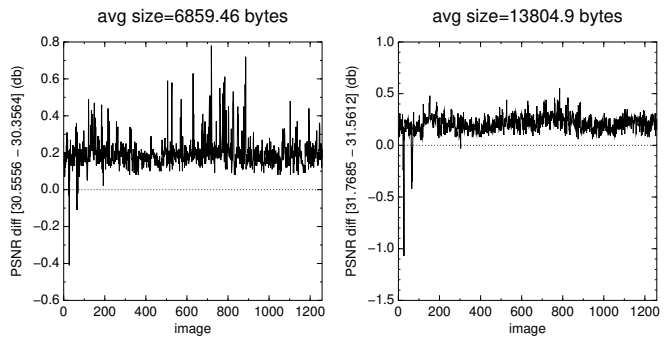

Figura 7: Imágenes TasCPC para ratios 0.005 y 0.01

A medida que aumenta la tasa (menos compresión y más calidad), el algoritmo DEBT sigue siendo superior al JPEG2000 para todas las imágenes de este conjunto de datos, excepto para 6 en el caso 0.005 y 13 en el caso 0.01. En todos estos casos, la calidad de la compresión fue muy alta (las imágenes eran muy oscuras), cerca de 40 dB o más para ambos algoritmos. La Fig. 7 muestra los resultados para estas tasas. En promedio, la ganancia de DEBT sobre JPEG2000 es de alrededor de 0.20 dB y 0.21 dB para las tasas de 0.005 y 0.01, respectivamente.

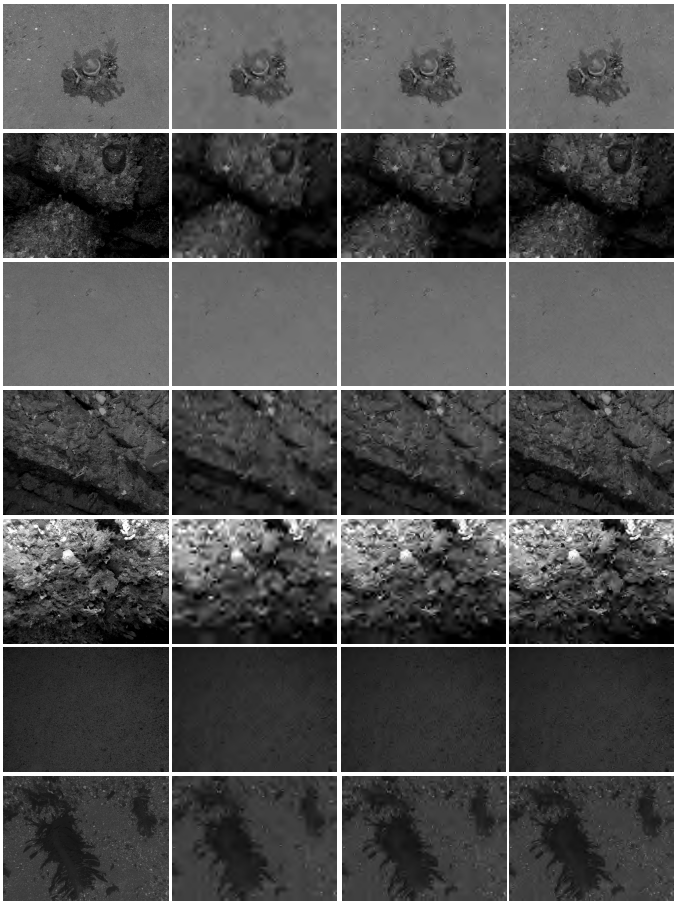

Figura 8: Imágenes submarinas 8, 19, 35, 256, 317, 432, y 834 comprimidas en original, 500, 1000, y 2000 bytes

De entre el conjunto de datos considerado, se seleccionaron aleatoriamente algunas imágenes que son presentadas en la Fig. 8. Cada fila muestra la imagen original (primera columna) y la comprimida con el algoritmo DEBT a exactamente 500, 1000, y 2000 bytes en las columnas 2ª, 3ª y 4ª, respectivamente.

Un punto importante a destacar es que el algoritmo DEBT funciona de forma muy sencilla, simplemente cambiando los pesos de cada celda mostrada en la Fig. 10, para crear un *stream* que no es óptimo en el sentido del MSE (Mean Square Error), pero que podría ser más adecuado para resaltar otras características de la imagen a *bitrates* muy bajos. Estas métricas alternativas deberían ser investigadas más a fondo.

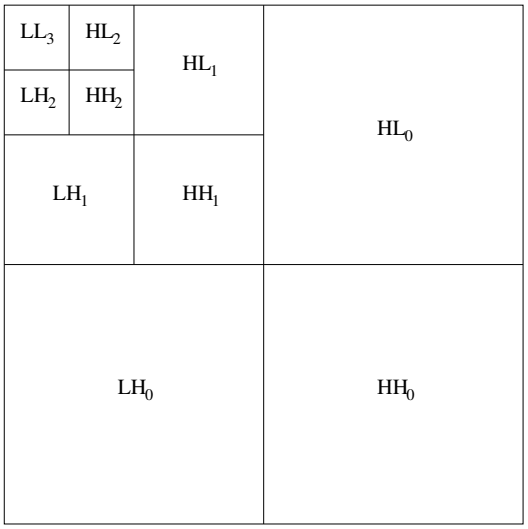

Figura 9: Descomposición wavelet de tres niveles.

7. Región de Interés (ROI)

En escenarios de bajo ancho de banda, o cuando las imágenes son altamente comprimidas, es posible que haya circunstancias donde las imágenes no tengan la suficiente calidad como para que un operador sea capaz de distinguir los detalles necesarios. En este caso, el uso de una ROI (*Region Of Interest*) es una solución elegante al problema de ser capaz de ver los detalles en una parte de la imagen manteniendo un nivel alto de compresión, a expensas de disminuir los detalles en las demás áreas de la imagen. La ROI ha sido ampliamente utilizada en conjunción con imágenes médicas y es parte integral del estándar JPEG2000. La mayoría de técnicas de ROI son utilizadas normalmente en conjunción con las técnicas de codificación de imágenes basadas en *wavelets* (Rehna and Kumar, 2012).

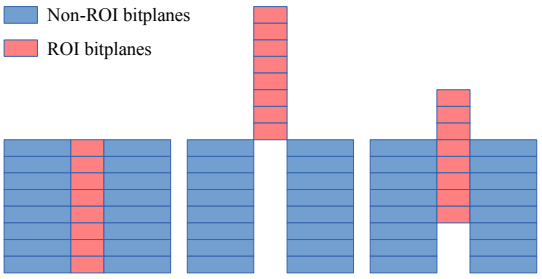

Figura 11: Métodos de codificación de ROI: (a) Compresión uniforme, (b) *bitplane shifts* para el método *maxshift*, y (c) los *bitplane shifts* para el *scaling-based method*.

En el procesamiento de una ROI, debe existir una forma de dar a conocer al decodificador qué regiones fueron codificadas con mayor prioridad que otras. Un método común conocido como MAXSHIFT (Subedar et al., 2004) (Delaunay et al., 2010) se utiliza comúnmente para que los *bitplanes* de la región ROI se codifiquen en su totalidad antes que cualquier plano de bits del resto de la imagen (fondo) (Ver Fig. 11). Esto tiene la ventaja de que casi no hay sobrecarga (solo el número de *bitplanes* extra es enviado para que el decodificador sepa que, después de llegar a este número, debería descalificar los coeficientes recibidos por la cantidad de *bitplanes* restantes), pero tiene la desventaja de tener que enviar la totalidad del ROI, con todos sus detalles, antes de recibir un solo bit del resto de la imagen.

Un método más útil conocido como SCALING (Subedar et al., 2004) (Delaunay et al., 2010) consiste simplemente en desplazar los coeficientes de ROI un cierto número de bits para que engañen al algoritmo de asignación de bits a pensar que son más importantes de lo que realmente son y codificándolos antes que otros coeficientes que se hicieron más pequeños debido al escalado (véase Fig. 11). De hecho, esto combina eficazmente los coeficientes de ROI con coeficientes de fondo que también son importantes (del mismo orden de magnitud) de tal manera que los resultados se ven con buena calidad en un fondo de menor calidad. La principal desventaja de este método es que un mapa de ROI debe ser enviado como información adicional al decodificador (sobrecarga), aumentando la cantidad mínima de bits necesaria para recuperar una aproximación adecuada a la imagen original.

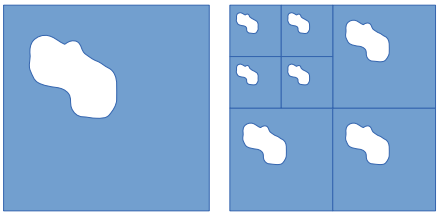

Figura 12: (a) Máscara ROI (b) Máscara Wavelet para 2 niveles de DWT.

Esta información del mapa puede consistir en coordenadas de objeto (rectángulos, elipses o polígonos arbitrarios) o, en caso de una región arbitraria, un mapa de bits del ROI. En este último caso, con el fin de reducir la cantidad de sobrecarga, este mapa podría ser el mapa resultante en la última sub-banda de descomposición (LL), disminuyendo así significativamente el tamaño del mapa de bits, pero teniendo el inconveniente de usar una escala más gruesa, dependiendo del número de descomposiciones diádicas (para una descomposición diádica de nivel n la cuadrícula sería $2^n \times 2^n$ píxeles). Este mapa de bits consta normalmente de una pequeña región y, por lo tanto, es un buen candidato para alguna forma de codificación *run-length* (Fig. 12).

También se han ideado otros métodos que entrelazan los coeficientes de ROI con los coeficientes de fondo, en un orden predeterminado y alternativo, con el fin de minimizar el *overhead* en la transmisión, pero la mayoría de ellos requieren cambios algorítmicos fundamentales para que tanto el codificador como el decodificador exploren los *bitplanes* en el mismo orden y son más complejos que el método MAXSHIFT y SCALING.

Los ejemplos utilizados en este documento fueron preparados con el algoritmo SCALING con regiones gruesas y arbitrarias, tal y como se describió anteriormente.

Debe observarse que, en general, el uso de la ROI impactará negativamente en el PSNR de toda la imagen reconstruida, pero mejorará significativamente la fidelidad en la propia ROI. Para ejemplificar este hecho se ha utilizado una imagen monocromática 1024×768 en la que se ha establecido una ROI rectangular de dimensiones 64×192 que contiene leyenda TRIDENT con la esquina superior izquierda en $(x_0, y_0) = (417, 65)$ y esquina inferior derecha en $(x_1, y_1) = (480, 256)$ (ver Fig. 13). La Fig. 14 muestra la diferencia resultante al codificar la región

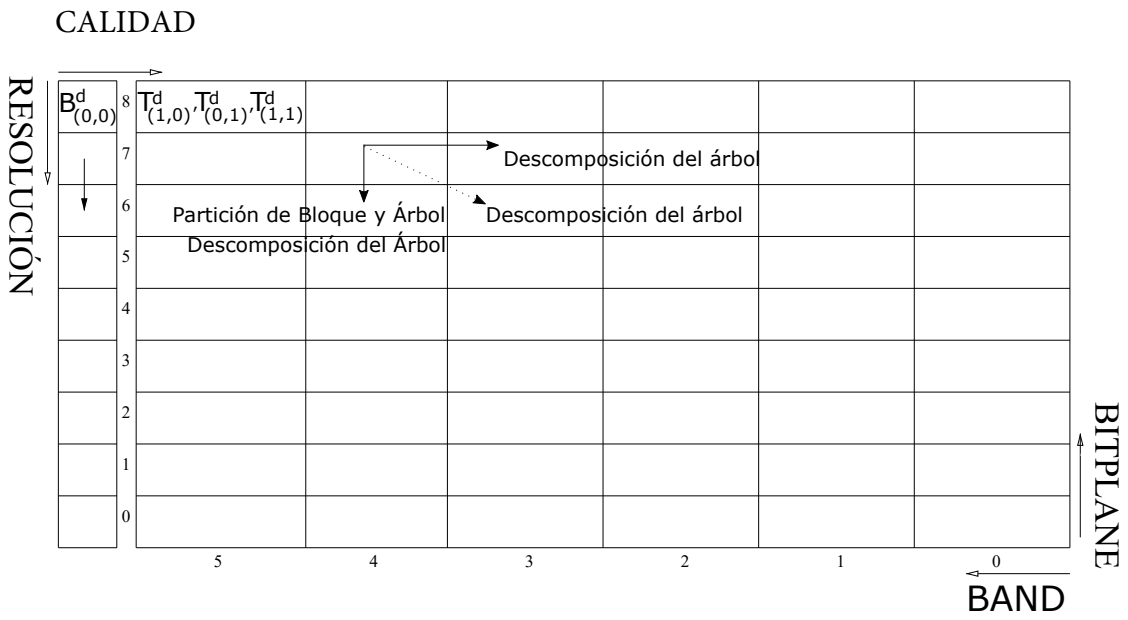

Figura 10: Orden de escaneado de bits.

alrededor de la leyenda con 4 bits de desplazamiento en comparación con la codificación de la imagen sin ROI.

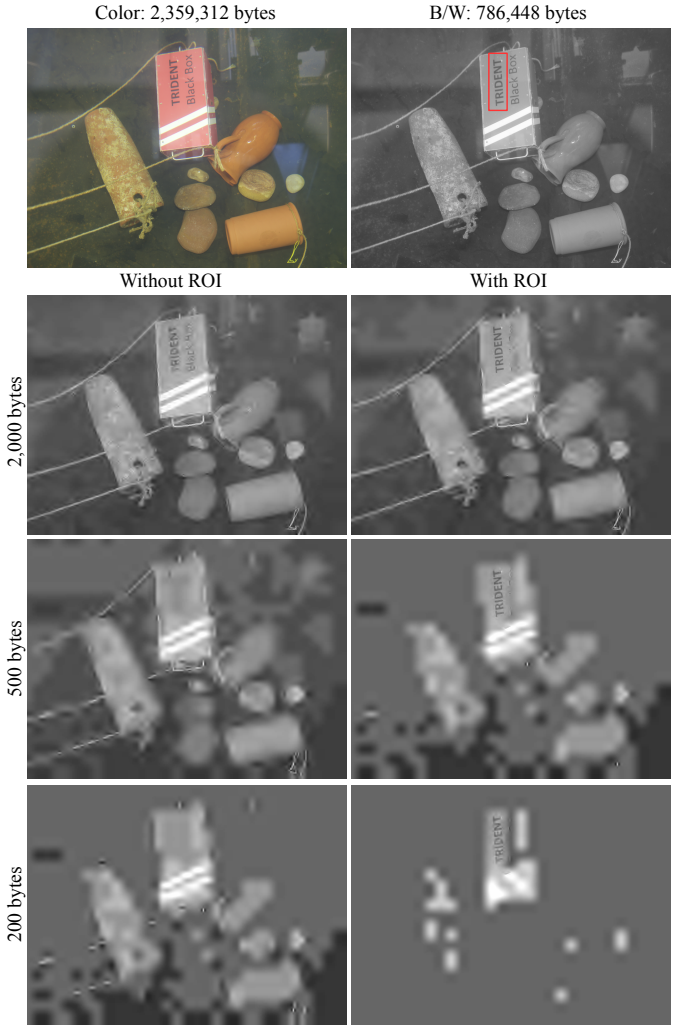

Figura 13: Comparación de la imagen comprimida sin y con ROI.

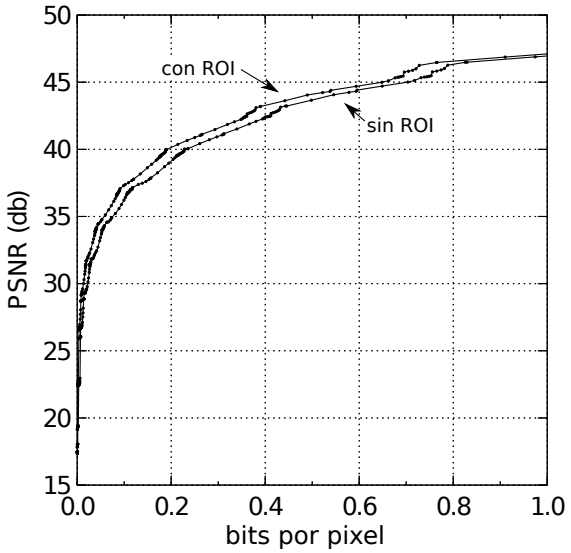

Figura 14: Efecto de la ROI en el PSNR.

8. Implementación de DEBT

Se ha desarrollado una implementación en el lenguaje C con código vectorial específico para la arquirectura de procesadores ARM e Intel, con el objetivo de acelerar el bucle interno de la rutina de la transformada wavelet. Se han implementado algunas transformadas wavelet - las transformadas wavelet enteras (CDF (2,2) [5/3], CDF (2,4) [9/3], CDF (4,2) [9/7] y CDF (4,4) [13/7]) y la transformada real CDF-9/7. La transformada usada para el ejemplo en este documento fue la CDF (4,4) [13/7], la cual tiene un filtro pasa-alta con 7 *taps* y un filtro pasa-baja con 13 *taps*. Esta wavelet permite la compresión sin pérdida y puede además ser truncada en cualquier punto dejando el rendimiento dentro de los 0.5 dB de la transformada real CDF-9/7, aunque siendo mucho más rápida debido a toda la aritmética de enteros.

Para las imágenes de 1024×768 píxeles, en las que se utilizaron niveles de gris de 8 bpp (esquina superior derecha ima-

gen en la Fig.13) ejecutamos el algoritmo de compresión tanto en una Raspberry Pi 2 Model B (RPi2B) como en una Beaglebone SBC (Single Board Computer). La RPi2B se basa en un procesador ARM de cuatro núcleos de 900 MHz, fabricado por Broadcom (BCM2836 SoC), mientras que el Beaglebone se basa en un ARM de un solo núcleo de 720 MHz (AM335x 720MHz ARM Cortex-A8) fabricado por Texas Instruments.

Los tiempos de ejecución en cada dispositivo para la compresión de la imagen que se muestra en la esquina superior derecha de la Fig. 13 sin y con ROI pueden verse en las Tablas 1, 2, 3, y 4. El algoritmo tiene un parámetro que especifica un tamaño máximo o un factor de "calidad" (que presenta una relación inversa con el PSNR). El uso normal (si no se requiere compresión sin pérdida) es usar un parámetro de calidad "bueno" para el almacenamiento local y la transmisión de cualquier prefijo de este fichero para versiones de baja calidad de la imagen comprimida. La "calidad" utilizada para cada línea de las tablas fue 0, 4, 8, 12, 16 y 24. La wavelet utilizada fue la CDF(4,4) (Cohen-Daubechies-Feauveau) b-spline entera, también conocida como la transformada 13/7-T, con 5 niveles de descomposición. La primera línea de cada tabla (donde el parámetro de "calidad" es 0, es decir, el PSNR presenta un valor infinito) es para el caso sin pérdidas, donde MSE = 0 (PSNR = ∞).

La columna etiquetada como "pre" es el tiempo (en milisegundos) para la transformación wavelet y todas las demás tareas necesarias para comenzar realmente a ejecutar el algoritmo de compresión (extracción media, conversión a *sign-magnitude*, etc.). Este paso es independiente de la cantidad de bits de salida y es, de hecho, un límite inferior para imágenes de este tamaño (es casi independiente del contenido de la imagen en sí y dependiente principalmente de las dimensiones de la imagen).

La columna etiquetada como "code" es el tiempo (en milisegundos) del algoritmo de compresión real y es directamente proporcional a la cantidad de bits de salida. Por lo tanto, la especificación de un factor de calidad o de un tamaño máximo tendrá un gran impacto en esta parte y en el tiempo de ejecución total para la compresión de la imagen.

Tabla 1: Tiempos RPi2B

PSNR (dB)	Size (bytes)	Pre (ms)	Code (ms)	Total (ms)
∞	277346	25.0	110.5	135.5
44.04	48111	25.0	24.5	49.5
40.66	22998	25.0	12.3	37.3
39.48	17356	25.0	9.0	34.0
37.38	10055	25.0	5.5	30.5
36.66	8366	25.0	4.5	29.5

Tabla 2: Tiempos Beaglebone

PSNR (dB)	Size (bytes)	Pre (ms)	Code (ms)	Total (ms)
∞	277346	58.0	172.0	230.0
44.04	48111	58.0	40.5	98.5
40.66	22998	58.0	20.8	78.8
39.48	17356	58.0	15.0	73.0
37.38	10055	58.0	9.4	67.4
36.66	8366	58.0	7.5	65.5

La implementación del bucle interno de la transformación wavelet se realizó utilizando las instrucciones vectoriales NEON para el ARM con la ayuda de las denominadas *intrinsics* del compilador GCC (GNU Compiler Collection). Una implementación sencilla en C tiene un coste de 50 ms para la RPi2B y 90 ms para la Beaglebone, para la columna "pre".

Tabla 3: Tiempos RPi2B (ROI)

PSNR (dB)	Size (bytes)	Pre (ms)	Code (ms)	Total (ms)
∞	283292	26.0	111.5	137.5
44.07	53667	26.0	26.5	52.5
40.72	27648	26.0	14.0	40.0
39.56	21074	26.0	10.7	36.7
37.48	13633	26.0	7.2	33.2
36.76	10590	26.0	5.5	31.5

Tabla 4: Tiempos Beaglebone (ROI)

PSNR (dB)	Size (bytes)	Pre (ms)	Code (ms)	Total (ms)
∞	283292	59.5	174.0	233.5
44.07	53667	59.5	44.0	103.5
40.72	27648	59.5	24.0	83.5
39.56	21074	59.5	17.8	77.3
37.48	13633	59.5	12.2	71.7
36.76	10590	59.5	9.5	69.0

La Tabla 5 compara el algoritmo actual con JPEG-2000. La implementación JPEG-2000 utilizada fue la del programa "JasPer" (Adams and Kossentini, 2000) con 6 niveles de descomposición y la transformada wavelet CDF-9/7. Para hacer una comparación "justa", también hemos incluido una ejecución de nuestro algoritmo con el mismo número de descomposiciones (6) y transformada wavelet (CDF-9/7) usadas para el caso JPEG-2000, junto con los 5 niveles de descomposición previamente utilizados y la transformada entera CDF (4,4), que es lo que hemos utilizado para los resultados de tiempos en las tablas anteriores.

Tabla 5: Comparación con JPEG2000 (sin ROI)

Rate	Size (bytes)	PSNR (dB)		
		JPEG2000	DEBT CDF(4,4)	DEBT CDF-9/7
0.001	758	28.37	28.68	29.03
0.002	1559	30.82	30.66	31.23
0.005	3898	33.99	34.03	34.22
0.010	7810	36.43	36.25	36.86
0.020	15724	39.20	38.97	39.86
0.050	39305	43.32	43.31	44.04
0.100	78637	46.65	46.62	47.73

Nuestra implementación del CDF-9/7 en la RPi2B tiene un coste de 59 ms para la columna "pre" y actualmente utiliza una implementación de punto flotante de 32 bits. Se está trabajando en una implementación que utiliza precisión de 16 bits en

punto flotante para el almacenamiento haciendo que la implementación de CDF-9/7 use la misma cantidad de memoria que las transformaciones de wavelet de enteros, de lo cual se esperan ganancias de velocidad considerables.

Es importante tener en cuenta que la cantidad de bytes utilizados en la comparación fue dada por el tamaño resultante del archivo JPEG-2000 mediante el siguiente comando: `jasper --input tank.pgm --output tank.jpc -output-format jpc -O rate=xxx`, donde `xxx` es la velocidad (primera columna) para la compresión. El tamaño del archivo resultante se utilizó para comprimir la misma imagen usando nuestro algoritmo para exactamente este tamaño, una vez con nuestros parámetros actuales (5 niveles de descomposición y la transformada wavelet entera CDF (4,4)) y otra con los mismos parámetros que los utilizados en el caso JPEG-2000 (6 niveles de descomposición y la transformada wavelet de punto flotante CDF-9/7).

Los resultados muestran que, para la imagen de ejemplo utilizada, nuestro algoritmo es bastante competitivo con el actual codec JPEG-2000, incluso cuando se utiliza la transformada wavelet entera CDF (4,4). En el ejemplo, nuestro algoritmo siempre supera al JPEG-2000 al utilizar el mismo número de descomposiciones (6) y transformación wavelet (CDF-9/7), siendo a la vez mucho más rápido.

Los tiempos de ejecución de cada algoritmo (incluyendo la entrada y salida) para una tasa de 0.1 son de 792 ms para el algoritmo JPEG-2000, 114 ms para el algoritmo propuesto (i.e. DEBT) con los mismos parámetros y, solo para una comparación simple, 66 ms para el antiguo algoritmo JPEG (W. B. Pennebaker and J. L. Mitchell, 1992) basado en DCT con calidad=86, lo que da un tamaño de archivo comprimido resultante de 78216 bytes.

Cabe señalar que, mientras que el algoritmo JPEG (DCT) es más rápido y utiliza mucha menos memoria, la comparación no es buena, ya que la imagen resultante tiene un PSNR de 45.62 dB, que es más de 2 dB peor que en el caso de DEBT. Tal vez podría realizarse una mejor comparación comparando los tiempos para llegar a una imagen que tenga la misma "calidad", la cual ocuparía muchos más bits en el caso de JPEG.

Para una tasa de 0,02, que es un valor más realista para ser usado en un escenario real, los tiempos alcanzados fueron 725 ms para JPEG-2000, 80 ms para nuestro algoritmo con los mismos parámetros y 60 ms para el algoritmo JPEG basado en DCT con calidad=12 (15310 bytes de tamaño del archivo resultante). El PSNR del archivo JPEG (DCT) resultante fue 35.02 dB, lo cual es 4.84 dB menos de lo que se logró con nuestro algoritmo.

9. Transmisión subacuática por RF de imágenes comprimidas con DEBT

Los dispositivos de RF utilizados para este experimento son los módulos comerciales UHF de baja potencia BiM3B (25 mW), conectados a antenas de 868,3 MHz de 1/4 de onda. Estos transceptores utilizan una modulación simple, FSK, para transmitir los bits. En el lado del transmisor la electrónica implicada es la RPi2B (Raspberry Pi), la RaspiCam, una Arduino Pro Micro y un módulo RF. En el lado del receptor, una Arduino y un módulo RF. Toda la electrónica ha sido encapsulada correctamente dentro de recipientes herméticos. Los contenedores de los transmisores y receptores han sido fijados a un palo de madera que ha sido sumergido aproximadamente a una profundidad de 15 cm en una piscina de agua dulce. Algunos trabajos han demostrado la posibilidad de establecer enlaces de RF directos entre un punto en el aire a otro punto sumergido en agua dulce (Shan Jiang, 2011) Teniendo en cuenta esto, la profundidad donde se fijó el experimento podría haber tenido algún efecto en los resultados. Por lo tanto, la variable de la profundidad se tendrá en cuenta para futuros experimentos.

Para este experimento, se han transmitido 100 imágenes comprimidas con DEBT en diferentes distancias, a 20, 40, 60, 70, 80 y 100 cm. Para enviar cada imagen se ha transmitido un prefijo de 400 bytes de la imagen comprimida, cada uno dentro de una sola PDU. El prefijo se corresponde con la entrada completa del algoritmo de descompresión, que se ajusta a un tamaño determinado (p.e. 400 bytes). Por cada distancia, se ha medido el número de errores en la recepción (el número de paquetes perdidos más el de paquetes con errores) y el RSSI. El receptor (Arduino + módulo de RF) se comunicaba con el PC a través de un cable USB, donde un proceso decodificaba y mostraba cada imagen, además de contabilizar el número de errores. El enlace de comunicación por RF se estableció a 9600 baudios para ser fieles a las velocidades típicas (generalmente bajas), de los enlaces inalámbricos submarinos. Cada paquete tenía un *overhead* de un tamaño de 15 bytes, y contenía en su *payload*, no solo los 400 bytes de la imagen codificada, sino también otros 26 bytes extra sin ningún significado establecido, pero que podrían representar información de otros sensores de un posible ROV.

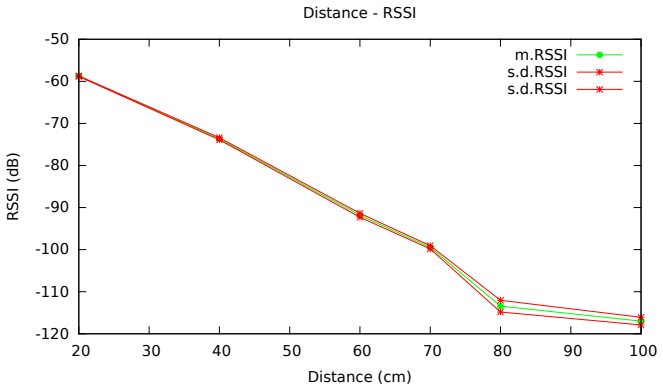

Figura 15: RSSI en una piscina de agua dulce de 8x4 usando los BiM3B (868.3 MHz a 25 mW). La media del RSSI se muestra en la línea verde, mientras que las líneas rojas representan la desviación típica. Cada punto es la distancia donde ha tenido lugar un muestreo

Con esta implementación preliminar ha sido posible transmitir cada una de las 100 imágenes sin errores, a una distancia máxima de 60 cm, en agua dulce. A partir de los 70 cm, solo 36 de las 100 imágenes fueron recibidas sin ningún error. A mayores distancias (> 70 cm), fue imposible transmitir una imagen con la configuración del enlace, los transceptores y antenas utilizados en este experimento en particular. La Fig. 15 contiene el muestreo del RSSI en cada posición. La Fig. 16 muestra los FPS obtenidos con la misma configuración del protocolo que la usada en los experimentos en agua dulce, para diferentes tamaños del prefijo de la imagen comprimida. Como se muestra

en la imagen, con un tamaño igual o menor a 800 bytes es posible realizar una transmisión, descompresión y visualización de las imágenes a tasa mayor que 1 FPS y a una velocidad de 9600 baudios. Normalmente, con el algoritmo DEBT, un tamaño de prefijo superior a 400 bytes correspondería a imágenes con una calidad suficiente que permitiría a un operador en la superfície monitorizar la entrada de la cámara.

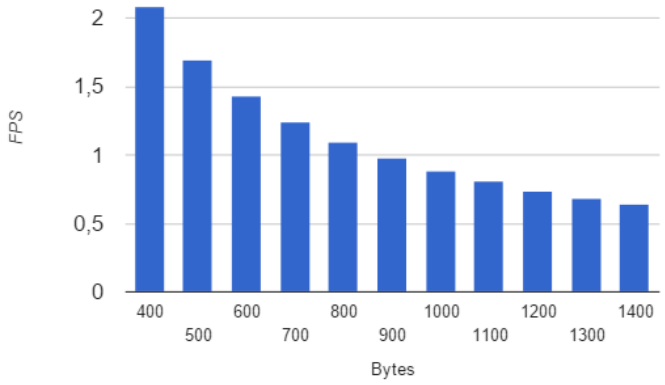

Figura 16: FPS obtenidos para cada tamaño del prefijo con la configuración experimental del protocolo y del enlace RF establecido a 9600 baudios.

10. Conclusiones y trabajo futuro

Este trabajo describe la implementación y uso de un algoritmo de compresión de imagen progresiva y la región de interés (ROI) para un sensor de imagen subacuático de RF pensado para ser instalado en un ROV. El operador puede decidir dinámicamente el tamaño, la calidad, la velocidad de fotogramas y la resolución de las imágenes recibidas, de modo que el ancho de banda disponible se utilice a su máximo potencial y con la latencia mínima requerida.

El sistema es capaz de ajustar de forma dinámica y precisa la compresión de imagen a un tamaño o calidad predefinidos y demostró ser capaz de enviar imágenes de buena calidad utilizando 400-800 bytes. La velocidad de fotogramas es directamente proporcional a la capacidad del canal y puede establecerse con precisión variando el tamaño de las imágenes comprimidas. La calidad se mejora al permitir al operador especificar una región de interés en la entrada de la cámara, lo que significa que se pueden observar más detalles en esta parte específica de la vista, manteniendo el tamaño de la imagen y el canal de red consumido.

Relacionados con el algoritmo de compresión, que se explica detalladamente en las secciones anteriores, los resultados muestran que un núcleo de un RPi2 puede comprimir imágenes de alta resolución (1024×768) de muy alta calidad. Se puede ver a partir de la tabla 1 que puede procesar cerca de 30 fotogramas por segundo con un PSNR de alrededor de 40 dB y todavía hay 3 núcleos más para ser utilizados por otros procesos.

El Beaglebone es un dispositivo menos potente para las imágenes de esta dimensión, pero todavía es capaz de comprimir alrededor de 12,5 fotogramas por segundo en el mismo 40 dB PSNR. En este caso, la calidad, el tamaño de trama o la velocidad de fotogramas podrían ajustarse para que la compresión utilizara una parte predefinida de la CPU, permitiendo que otras tareas pudieran ejecutarse en la misma tarjeta.

Siempre y cuando el ROI sea una región simple, su uso no producirá una diferencia destacable en los tiempos de codificación, aunque hay una pequeña penalidad a pagar en la eficiencia de compresión para toda la imagen (alrededor del 2 % en nuestro ejemplo).

En resumen, se ha implementado un algoritmo de compresión progresiva especialmente diseñado para que posea escalabilidad de calidad y resolución, ROI, y es bastante simple para poder ser usado en ordenadores de recursos limitados. Los resultados muestran que es bastante competitivo con los algoritmos de compresión de última generación como JPEG2000, además de ser mucho más rápido.

Los trabajos de investigación actuales se centran en mejorar la capa física de la comunicación para alcanzar distancias del orden de los 5 metros, por un lado mejorando el diseño de la antena y el circuito de adaptación de impedancia, y por otro mediante el uso de módems comerciales RF, luz y acústicos, de acuerdo con los objetivos del proyecto. Además, en el momento de la escritura de este artículo, se está mejorando el protocolo de transporte de nivel superior, utilizando técnicas de congestión que obtengan el mejor uso del ancho de banda disponible. También, en estos momentos se está trabajando en el aprovechamiento de las estructuras de datos del sistema de compresión para realizar reconocimiento de patrones globales (e.g. escenarios, tuberías, etc.).

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio de Economía y competitividad, código de proyecto DPI2014-57746-C3 (proyecto MERBOTS), por la Generalitat Valenciana GVA, con el código de proyecto PROMETEO/2016/066 y por la Universidad Jaume I, proyecto MASUMIA (P1-1B2015-68), becas PREDOC/2012/47, PREDOC/2013/46, y por el CNPq del Brasil.

El conjunto de imágenes submarinas utilizadas en este trabajo ha sido fruto del esfuerzo del *Australian Center for Field Robotics* (`http://marine.acfr.usyd.edu.au/datasets/data/TasCPC/TasCPC\_RM.tar.gz`)

Referencias

Adams, M., Kossentini, F., Sept 2000. Jasper: a software-based JPEG-2000 codec implementation. In: Image Processing, 2000. Proceedings. 2000 International Conference on. Vol. 2. pp. 53–56.

Calderbank, A., Daubechies, I., Sweldens, W., Yeo, B.-L., 1998. Wavelet transforms that map integers to integers. Applied and Computational Harmonic Analysis 5 (3), 332 – 369.
URL: `http://www.sciencedirect.com/science/article/pii/S1063520397902384`
DOI: `http://dx.doi.org/10.1006/acha.1997.0238`

Carreras, M., Ridao, P., García, R., Ribas, D., Palomeras, N., 2012. Inspección visual subacuática mediante robótica submarina. Revista Iberoamericana de Automática e Informática Industrial RIAI 9 (1), 34–45.

Delaunay, X., Thiebaut, C., Chabert, M., Charvillat, V., Morin, G., Oct. 2010. Progressive coding of satellite images with regions of interest. In: On-Board Payload Data Compression Workshop. Toulouse, France.

Farr, N., Bowen, A., Ware, J., Pontbriand, C., Tivey, M., May 2010. An integrated, underwater optical /acoustic communications system. In: OCEANS 2010 IEEE - Sydney. pp. 1–6.
DOI: `10.1109/OCEANSSYD.2010.5603510`

Moinuddin, A., Khan, E., May 2006. Wavelet based embedded image coding using unified zero-block-zero-tree approach. In: Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on. Vol. 2. pp. II–II.
DOI: 10.1109/ICASSP.2006.1660377

Pearlman, W., Islam, A., Nagaraj, N., Said, A., Nov 2004. Efficient, low-complexity image coding with a set-partitioning embedded block coder. Circuits and Systems for Video Technology, IEEE Transactions on 14 (11), 1219–1235.
DOI: 10.1109/TCSVT.2004.835150

Pelekanakis, C., Stojanovic, M., Freitag, L., Sept 2003. High rate acoustic link for underwater video transmission. In: OCEANS 2003. Proceedings. Vol. 2. pp. 1091–1097 Vol.2.
DOI: 10.1109/OCEANS.2003.178494

Prats, M., del Pobil, A. P., Sanz, P. J., 2013. Robot physical interaction through the combination of vision, tactile and force feedback. Applications to assistive robotics. Springer Tracts in Advanced Robotics, Volume 84. Springer Publishing Company, Incorporated.

Rehna, V. J., Kumar, M. K. J., 2012. Wavelet based image coding schemes: A recent survey. CoRR abs/1209.2515.
URL: http://arxiv.org/abs/1209.2515

Ribas, J., Sura, D., Stojanovic, M., Sept 2010. Underwater wireless video transmission for supervisory control and inspection using acoustic OFDM. In: OCEANS 2010. pp. 1–9.
DOI: 10.1109/OCEANS.2010.5663839

Said, A., Pearlman, W., Jun 1996. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. Circuits and Systems for Video Technology, IEEE Transactions on 6 (3), 243–250.
DOI: 10.1109/76.499834

Sanz, P. J., Peñalver, A., Sales, J., Fornas, D., Fernández, J. J., Perez, J., Bernabé, J. A., Oct 2013a. GRASPER: A multisensory based manipulation system for underwater operations. In: 2013 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, Manchester, UK.

Sanz, P. J., Peñalver, A., Sales, J., Fernández, J. J., Pérez, J., Fornas, D., García, J., Marin, R., Sep 2015. Multipurpose underwater manipulation for archaeological intervention. In: Sixth International Workshop on Marine Technology (MARTECH'15). Cartagena, Spain.

Sanz, P. J., Prats, M., Ridao, P., Ribas, D., Oliver, G., Orti, A., September 2010. Recent progress in the RAUVI project. A reconfigurable autonomous underwater vehicle for intervention. In: 52-th International Symphosium ELMAR-2010. Zadar, Croatia, pp. 471–474.

Sanz, P. J., Ridao, P., Oliver, G., Casalino, G., Petillot, Y., Silvestre, C., Melchiorri, C., Turetta, A., Sept 2013b. TRIDENT: An european project targeted to increase the autonomy levels for underwater intervention missions. In: OCEANS'13 MTS/IEEE conference. San Diego, CA, pp. 1–10.

Shan Jiang, S. G., 2011. Electromagnetic wave propagation into fresh water. Journal of Electromagnetic Analysis and Applications 3 (7), 261–266.
DOI: 10.4236/jemaa.2011.37042

Shaw, A., Al-Shamma'a, A., Wylie, S., Toal, D., Sept 2006. Experimental investigations of electromagnetic wave propagation in seawater. In: Microwave Conference, 2006. 36th European. pp. 572–575.

Siegel, M., King, R. W. P., Jul 1973. Electromagnetic propagation between antennas submerged in the ocean. Antennas and Propagation, IEEE Transactions on 21 (4), 507–513.
DOI: 10.1109/TAP.1973.1140525

Stojanovic, M., Preisig, J., January 2009. Underwater acoustic communication channels: Propagation models and statistical characterization. Communications Magazine, IEEE 47 (1), 84–89.
DOI: 10.1109/MCOM.2009.4752682

Subedar, M., Karam, L., Abousleman, G., May 2004. An embedded scaling-based arbitrary shape region-of-interest coding method for JPEG2000. In: Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on. Vol. 3. pp. iii–681–4.
DOI: 10.1109/ICASSP.2004.1326636

Taubman, D. S., Marcellin, M. W., 2001. JPEG 2000: Image Compression Fundamentals, Standards and Practice. Kluwer Academic Publishers, Norwell, MA, USA.

Usevitch, B., Mar 1996. Optimal bit allocation for biorthogonal wavelet coding. In: Data Compression Conference, 1996. DCC '96. Proceedings. pp. 387–395.
DOI: 10.1109/DCC.1996.488344

W. B. Pennebaker and J. L. Mitchell, 1992. JPEG still image data compression standard. New York: Van Nostrand Reinhold, 1992.

Wheeler, F. W., Pearlman, W., 2000. Combined spatial and subband block coding of images. In: Image Processing, 2000. Proceedings. 2000 International Conference on. Vol. 3. pp. 861–864 vol.3.
DOI: 10.1109/ICIP.2000.899592

Zhang, H., Meng, F., Aug 2012. Exploiting the skin effect using radio frequency communication in underwater communication. In: Industrial Control and Electronics Engineering (ICICEE), 2012 International Conference on. pp. 1150–1153.
DOI: 10.1109/ICICEE.2012.305