# ESCALABILIDAD DE UNA IMAGEN
## CALIDAD
### TRANSMISIÓN PROGRESIVA

## OBJETIVO

Aplicar el algoritmo de codificación por transmisión progresiva para modificar la calidad de una imagen mediante indexación con el fractal de Hilbert y transformada wavelet.

## DESARROLLO

**ESTRUCTURA DEL SISTEMA DE CODIFICACIÓN POR TRANSMISIÓN PROGRESIVA.**

El sistema que permite realizar la codificación por transmisión progresiva de una imagen se representa mediante el modelo cibernético de primer orden que se observa en la Figura 1. Asimismo, la Tabla 1 presenta las variables de entrada, que se requieren durante el proceso y la salida que se entrega.


**Figura 1**.- Modelo cibernético de primer orden del sistema de codificación por transmisión progresiva.

| Nombre | Tipo | Descripción |
|---|---|---|
| $\boldsymbol{I}$ | Entero sin signo de 8 bits | Arreglo de dimensiones $m \times n \times 3$ que almacena la información de una imagen considerada como original. |
| $\boldsymbol{bpp_{ideal}}$ | Entero sin signo de 8 bits | Valor ideal de la tasa de bits a la que se debe comprimir la imagen original. |
| $\boldsymbol{\hat{I}}$ | Entero sin signo de 8 bits | Arreglo de dimensiones $m \times n \times 3$ que almacena la información de la imagen original recuperada con un cambio de calidad. |

| | | |
|---|---|---|
| $bpp_{real}$ | Entero sin signo de 8 bits | Valor real de la tasa de bits a la que se comprime la imagen original con el sistema. |
| $PSNR$ | Punto flotante de doble precisión | Valor de la relación señal a ruido de pico en dB para medir la calidad de la imagen recuperada. |

**Tabla 1**.- Entradas requeridas y salidas entregadas después de aplicar la codificación por transmisión progresiva.

Internamente, el sistema de la Figura 1 se divide en los diferentes subprocesos que se observa en la Figura 2. Asimismo, en la Tabla 2 se muestran las variables que intervienen dentro del sistema y permiten modificar la calidad de la imagen a partir de realizar una codificación por transmisión progresiva. Cabe mencionar que el contenido de esta tabla se rellena solo con aquellas variables que no se consideran en la Tabla 1 y que faltan por describirse.

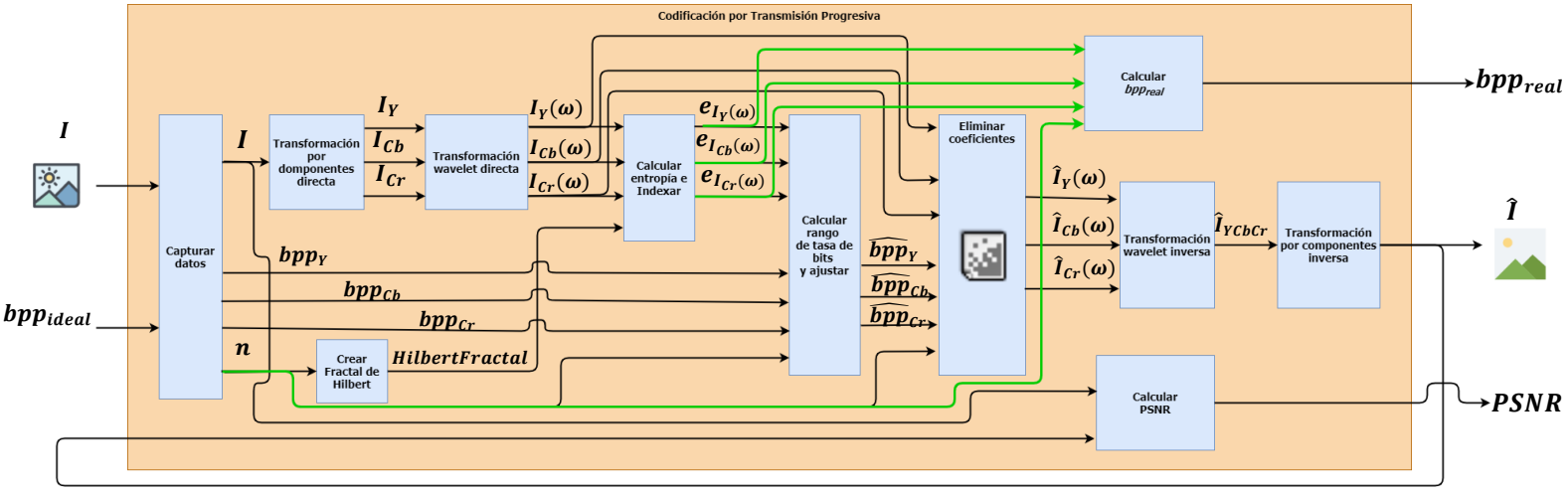


**Figura 2**.- Bloques internos del sistema de codificación por transmisión progresiva.

| Nombre | Tipo | Descripción |
|---|---|---|
| $n$ | Entero sin signo de 8 bits | Variable que almacena la potencia de 2 que genera el lienzo $2^n \times 2^n$ de la imagen original. |
| $bpp_Y$ | Punto flotante de doble precisión | Variable que almacena el porcentaje de la tasa de bits total correspondiente al canal de intensidad, $Y$. |
| $bpp_{Cb}$ | Punto flotante de doble precisión | Variable que almacena el porcentaje de la tasa de bits total correspondiente al canal que guarda la información de las cromaticidades azul y amarilla, $Cb$. |

| | | |
|---|---|---|
| $\boldsymbol{bpp_{Cr}}$ | Punto flotante de doble precisión | Variable que almacena el porcentaje de la tasa de bits total correspondiente al canal que guarda la información de las cromaticidades roja y verde, $\boldsymbol{Cr}$. |
| $\boldsymbol{HilbertFractal}$ | Entero sin signo de 8 bits | Arreglo con las dimensiones $2^n \times 2^n$ de la imagen original que almacena la secuencia de números correspondiente al Fractal de Hilbert. |
| $\boldsymbol{I_Y}$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena la información del canal de intensidad de la imagen, $\boldsymbol{Y}$. |
| $\boldsymbol{I_{Cb}}$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena la información del canal con la información de las cromaticidades azul y amarilla de la imagen, $\boldsymbol{Cb}$. |
| $\boldsymbol{I_{Cr}}$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena la información del canal con la información de las cromaticidades roja y verde de la imagen, $\boldsymbol{Cr}$. |
| $\boldsymbol{I_Y(\omega)}$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena al canal $\boldsymbol{Y}$ transformado a la frecuencia. |
| $\boldsymbol{I_{Cb}(\omega)}$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena al canal $\boldsymbol{Cb}$ transformado a la frecuencia. |
| $\boldsymbol{I_{Cr}(\omega)}$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena al canal $\boldsymbol{Cr}$ transformado a la frecuencia. |
| $\boldsymbol{e_{I_Y(\omega)}}$ | Punto flotante de doble precisión | Arreglo de dimensiones $2^n \times 2^n$ que almacena la entropía de los elementos del canal $\boldsymbol{Y}$ transformado a la frecuencia. |
| $\boldsymbol{e_{I_{Cb}(\omega)}}$ | Punto flotante de doble precisión | Arreglo de dimensiones $2^n \times 2^n$ que almacena la entropía de los elementos del canal $\boldsymbol{Cb}$ transformado a la frecuencia. |
| $\boldsymbol{e_{I_{Cr}(\omega)}}$ | Punto flotante de doble precisión | Arreglo de dimensiones $2^n \times 2^n$ que almacena la entropía de los elementos del canal $\boldsymbol{Cr}$ transformado a la frecuencia. |
| $\boldsymbol{\widehat{bpp}_Y}$ | Punto flotante de doble precisión | Variable que almacena la tasa de bits correspondiente al canal $\boldsymbol{Y}$ ajustada dentro de los límites posibles. |
| $\boldsymbol{\widehat{bpp}_{Cb}}$ | Punto flotante de doble precisión | Variable que almacena la tasa de bits correspondiente al canal $\boldsymbol{Cb}$ ajustada dentro de los límites posibles |

| | | |
|---|---|---|
| $\widehat{\boldsymbol{bpp}_{\boldsymbol{Cr}}}$ | Punto flotante de doble precisión | Variable que almacena la tasa de bits correspondiente al canal $\boldsymbol{Cr}$ ajustada dentro de los límites posibles |
| $\hat{\boldsymbol{I}}_{\boldsymbol{Y}}(\boldsymbol{\omega})$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena al canal $\boldsymbol{Y}$ transformado a la frecuencia y con información eliminada en las bandas horizontal, diagonal o vertical. |
| $\hat{\boldsymbol{I}}_{\boldsymbol{Cb}}(\boldsymbol{\omega})$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena al canal $\boldsymbol{Cb}$ transformado a la frecuencia y con información eliminada en las bandas horizontal, diagonal o vertical. |
| $\hat{\boldsymbol{I}}_{\boldsymbol{Cr}}(\boldsymbol{\omega})$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena al canal $\boldsymbol{Cr}$ transformado a la frecuencia y con información eliminada en las bandas horizontal, diagonal o vertical. |
| $\hat{\boldsymbol{I}}_{\boldsymbol{YCbCr}}$ | Punto flotante de doble precisión. | Arreglo de dimensiones $2^n \times 2^n$ que almacena a la imagen con la calidad modificada en el espacio post recepcional $\boldsymbol{YCbCr}$. |

**Tabla 2**.- Variables que intervienen internamente en el proceso de escalabilidad de calidad mediante codificación por transmisión progresiva.

## METODOLOGÍA

De acuerdo con la estructura del sistema que se plantea en la Figura 2 se aplican los siguientes pasos para recuperar la imagen original con la calidad modificada:

**1.** Capturar datos de entrada.

1.1. Leer la imagen original, $\boldsymbol{I}$.
1.2. Leer la potencia $\boldsymbol{n}$**,** que genera el lienzo de la imagen original $\boldsymbol{I}$.
1.3. Leer $\boldsymbol{bpp_{ideal}}$ y distribuirlo entre los canales del espacio post recepcional YCbCr.

$$bpp_Y = 0.5bpp_{ideal} \quad bpp_{Cb} = 0.25bpp_{ideal} \quad bpp_{Cr} = 0.25bpp_{ideal}$$

**2.** Crear el fractal de Hilbert de orden $\boldsymbol{n}$.

$$HilbertFractal_{2^n \times 2^n}$$

**3.** Aplicar una transformación por componentes directa a la imagen original $\boldsymbol{I}$, es decir, pasar del espacio recepcional RGB al espacio post recepcional YCbCr.

$$I_{RGB} \rightarrow I_{YCbCr}$$

**4.** Realizar una transformación wavelet directa de 3 niveles**.**

$$I_{YCbCr}(t) \rightarrow I_{YCbCr}(\omega)$$

**5.** Calcular entropía para cada canal

$$e = \lceil \log_2 |I_{YCbCr}(\omega) + 1| \rceil$$

5.1. Ordenar la entropía de cada canal con una indexación basada en el Fractal de Hilbert, es decir, realizar una conversión de 2 dimensiones a 1 dimensión.

$$e_{indexada}[HilbertFractal] = e$$

**6.** Calcular tasa de bits:

6.1. Tasa máxima de cada canal.

$$bpp_{máx_{CH}} = \frac{e_{indexada_{CH}}}{4^n}$$

6.2. Tasa mínima de cada canal.

$$bpp_{mín_{CH}} = \frac{e_{residual_{indexada_{CH}}}}{4^n}$$

6.3. Verificar que el bpp ideal de cada canal esta dentro de sus respectivos límites y ajustar dentro de su respectivo rango si es necesario.

$$bpp_{CH} = \begin{cases} bpp_{máx_{CH}} & ;\ si\ \ bpp_{CH} > bpp_{máx_{CH}} \\ bpp_{mín_{CH}} & ;\ si\ \ bpp_{CH} < bpp_{mín_{CH}} \\ bpp_{CH} & ;\ si\ \ bpp_{mín_{CH}} < bpp_{CH} < bpp_{máx_{CH}} \end{cases}$$

**7.** Eliminar coeficientes utilizando una técnica de *bit allocation*:

7.1.1. Calcular los bits para el $\boldsymbol{bpp}_{ideal}$ de cada canal.

$$bits_{CH} = bpp_{CH} * 4^n$$

7.1.2. Determinar la cantidad de valores que se transmitirán en cada canal sin rebasar la tasa de bits $\boldsymbol{bpp_{ideal}}$.

$$indice_{CH} = \left\lfloor \frac{bits_{CH}}{\bar{e}_{CH}} \right\rfloor$$

7.1.3. Convertir cada canal de la imagen transformada de 2 dimensiones a 1 dimensión, es decir, ordenar sus elementos indexando con el Fractal de Hilbert.

$$I(\omega)_{indexada}[HilbertFractal] = I(\omega)_{m \times n \times CH}$$

7.1.4. Recuperar solo la cantidad de pixeles y coeficientes que señala el índice de cada canal de $\boldsymbol{I(\omega)_{indexada}}$ y convertir el resto de elementos en ceros.

$$I(\omega)_{indexada_{recuperada}}[i] = \begin{cases} I(\omega)_{indexada}[i] & ;\ \ i \leq indice \\ 0 & ;\ \ i > indice \end{cases}_{\forall\, i=0\ \ldots\ n;}$$

7.1.5. Reconstruir cada canal pasando de 1 dimensión a 2 dimensiones con el Fractal de Hilbert.

$$I(\omega) = I(\omega)_{indexada}[HilbertFractal]$$

**8.** Realizar una transformación wavelet inversa de 3 niveles**.**

$$I(\omega) \rightarrow I_{YCbCr}$$

**9.** Aplicar una transformación por componentes inversa a la imagen $\boldsymbol{I_{YCbCr}}$, es decir, pasar del espacio recepcional RGB al espacio post recepcional YCbCr.

$$I_{RGB} \rightarrow I_{YCbCr}$$

**10.** Entregar la imagen recuperada $\hat{\boldsymbol{I}}$.

# PRUEBA DE FUNCIONAMIENTO

La metodología planteada en este trabajo se implementa en el Entorno de Desarrollo Integrado (*IDE*, por sus siglas en inglés) de *MATLAB* con el lenguaje de programación *M*. Para verificar que el sistema es funcional se utiliza la una imagen de prueba que se observa en la Figura 3 cuyas dimensiones son 512x512 pixeles.

**Figura 3.-** Imagen de prueba a la que se aplica la codificación por transmisión progresiva y se modifica su calidad.

Posteriormente se aplica el algoritmo de codificación por transmisión progresiva con dos tasas de bits ideales diferentes, la primera de 0.3 bpp y la segunda de 5.5 bpp. De esta manera, el sistema entrega las imágenes de la Figura 4 mismas que se han modificado en calidad con respecto a la imagen original.


a) $bpp_{ideal} = 0.3\ bpp$ b) $bpp_{ideal} = 5.5\ bpp$

**Figura 4.-** Imágenes recuperadas con la calidad modificada de acuerdo con una tasa de bits ideal específica.

# EXPERIMENTOS Y RESULTADOS

Para observar como varía la calidad cuando se modifica la tasa de bits se aplica la metodología en repetidas ocasiones con la imagen de prueba de la Figura 3 para diferentes valores ideales de bits por pixel. Los límites inferior y superior en este experimento son de 0.3 bpp y 5.5 bpp, respectivamente. En cada ensayo se captura el valor de la relación señal a ruido de pico (*PSNR*, por sus siglas en inglés) que es una de las métricas más comunes para medir la calidad de una imagen. Asimismo, se almacena su correspondiente tasa de bpp real y el conjunto de datos se presenta en la gráfica de la Figura 5 en la cual se muestra la relación entre el bpp real y el PSNR de la imagen.

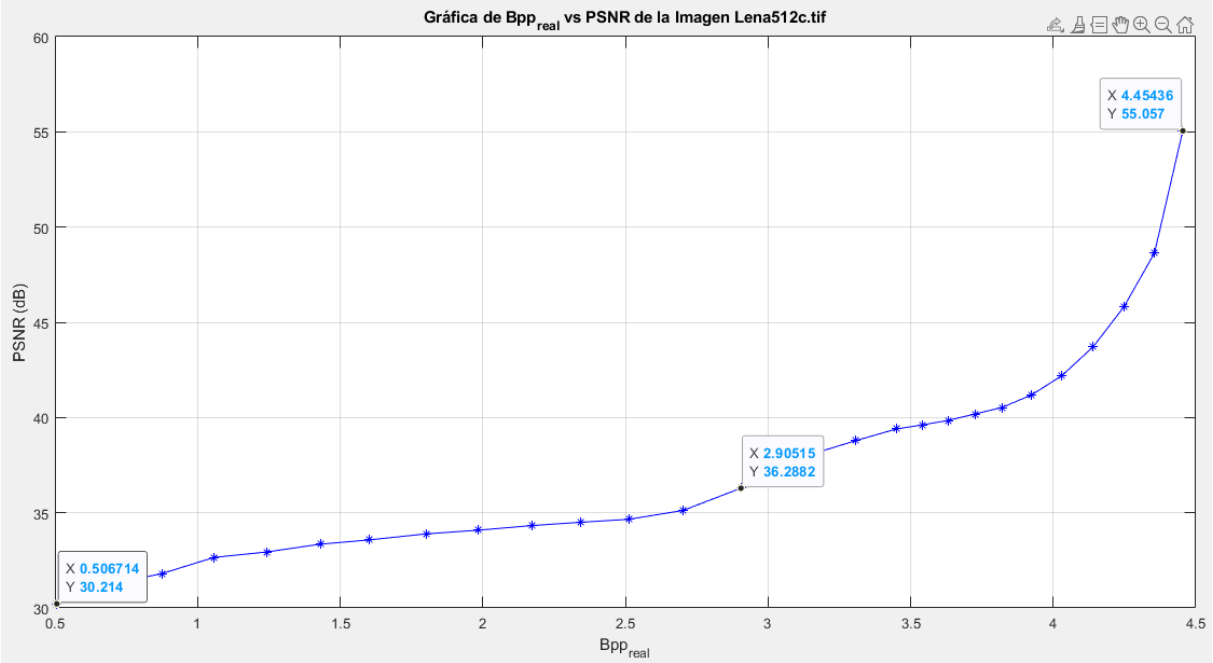


**Figura 5.-** Variación de la calidad de la imagen de prueba con respecto a la tasa de bits por pixel con la que se recupera.

Así, se tiene que para una tasa de 0.506714 bpp la calidad de la imagen es de 30.214 dB mientras que para 2.90515 bpp aumenta el valor del PSNR a 36.2882 dB y finalmente en 4.45436 bpp ocurre que la calidad es de 55.057 dB. De esta manera, se tiene que la calidad crece de manera exponencial conforme aumenta la cantidad de bits por pixel que permite transmitir el canal.

Por otro lado, para la ejecución del procedimiento experimental se utilizan dos paradigmas de programación diferentes: monoprogramación y multiprogramación. En el primer caso se repite la metodología con sus respectivos datos de forma secuencial. Esto quiere decir que primero se aplica la codificación por transmisión progresiva con un valor específico de bpp y hasta que finaliza este proceso comienza de nuevo la metodología con el siguiente dato. En el caso del segundo paradigma, se ejecutan los procesos de forma paralela. Lo anterior implica que la metodología se ejecute con un determinado valor de bpp al mismo tiempo que se realiza el mismo proceso con otra tasa de bits por pixel, es decir, que ocurra más de un ensayo al mismo tiempo.

Para verificar cuál de los dos paradigmas es más óptimo se cuantifica el tiempo que le toma a cada uno en completar el proceso experimental y generar la gráfica. Así, en la monoprogramación que se ejecuta de manera secuencial se tiene un tiempo de 8.528913 segundos. Por otra parte, la multiprogramación que se ejecuta de forma paralela se requiere de un tiempo de 4.332831 segundos para completar la tarea.

# CONCLUSIONES

Se aplicó el algoritmo de codificación por transmisión progresiva el cual se utilizó con la finalidad de modificar la calidad de una imagen, este proceso se realizó gracias a herramientas y técnicas matemáticas como la indexación con el Fractal de Hilbert y la transformada wavelet. La importancia de dicho algoritmo es que siempre intenta recuperar la imagen con la mejor calidad posible que permita el canal sin que se pierda la estructura original que esta tiene. Lo anterior ocurre gracias a que la transmisión progresiva siempre se asegura de mandar las bajas frecuencias de la imagen y poco a poco se agregan detalles de alta frecuencia pertenecientes a los bordes. Esto solo sucede si el ancho de banda del canal es suficiente para enviar más información. Así, la calidad se va modificando de manera directamente proporcional con respecto a las capacidades de transmisión del canal.

La relación entre la tasa de bits y la calidad se comprueba en la Sección de Experimentos y Resultados donde se codificó con transmisión progresiva a una imagen utilizando diferentes tasas de bits desde un límite inferior hasta un límite superior. De esta manera, se obtuvo que el PSNR aumenta exponencialmente demostrando así la funcionalidad de la codificación por transmisión progresiva para modificar la calidad de una imagen original. Sin embargo, aun existen aspectos que mejorar en el algoritmo presentado en este trabajo, sobre todo en la técnica de *bit allocation* utilizada para determinar la cantidad de pixeles y coeficientes que se enviaran por cada canal.

Por otro lado, la computación paralela es una herramienta fundamental para la realización de experimentos en los que se realiza el tratamiento de una gran cantidad de datos. Uno de los principales beneficios que ofrece es el ahorro de tiempo al momento de realizar los cálculos necesarios. De igual manera, esto se comprobó en la Sección de Experimentos y Resultados. Así, se obtuvo que después de ejecutar el algoritmo de forma totalmente secuencial la computadora se tardó 8.528913 segundos. Por otro lado, al configurar el equipo de cómputo para trabajar de forma paralela, el tiempo se redujo hasta 4.332831 segundos.