UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

INTERPOLACIÓN DE IMÁGENES DIGITALES

Enrique Luque Lanza
Tutor: Jesús Bescós Cano

MAYO DE 2020

INTERPOLACIÓN DE IMÁGENES DIGITALES

AUTOR: Enrique Luque Lanza
TUTOR: Jesús Bescós Cano

Video Processing and Understanding Lab
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Mayo de 2020

Resumen (Castellano)

Este Trabajo Fin de Grado ha sido realizado con objeto de profundizar en las diferentes técnicas de interpolación aplicadas sobre imágenes digitales, las cuales son de gran utilidad en campos de diversas índoles tales como: medicina, estudios científicos, aplicaciones comerciales, biometría y seguridad, etc.

Dadas unas primeras nociones básicas sobre imagen digital, y una vez fundamentadas las bases y herramientas principales empleadas en esta y en su procesado, nos adentraremos de lleno en qué es y para qué sirve la interpolación digital.

Haremos un largo y bastante detallado recorrido sobre cada una de las técnicas de interpolación de imágenes digitales ya existentes, realizando un estudio sobre todas éstas y analizando de forma teórica y práctica cada una de ellas.

Someteremos cada una de estas técnicas a estudio, siendo capaces de ver así cómo son los resultados y de esta forma ser capaces de establecer una crítica comparativa entre cada una de ellas. Todo esto nos permitirá tener diferentes resultados visuales y por lo tanto una primera aproximación a los pros y contras del empleo de una técnica u otra sobre distintas imágenes de entrenamiento.

Una vez realizado este trabajo, habremos conseguido todos los ingredientes necesarios para el diseño de nuevos algoritmos que se nos ofrezcan como alternativas a los ya establecidos; este será uno de los principales objetivos.

Por último, evaluaremos mediante técnicas de calidad de imagen, tanto objetivas como subjetivas, los resultados de aplicar en diferentes bases de datos de imágenes cada uno de los algoritmos presentados en este trabajo. Mediante diferentes representaciones gráficas podremos ver qué tan mejores o peores son los algoritmos propuestos en relación con los ya existentes.

Palabras clave (Castellano)

Imagen, Digital, Procesamiento, Señal, Técnicas, Interpolación, Sistemas, Algoritmos, Desarrollo, Teórico, Práctico, Análisis espectral, Contenido espacial, 2D, Señales, Procesado, Espacio, Pixel, Color, Grises, Aliasing, Filtros, Filtrado, Frecuencias, Transformada, Fourier, Vecino, Bilineal, Bicúbica, DFT, DCT, Zero-Padding, Espectro, Densidad, Energía, Detalle, Fino, Grueso, Diezmado, Muestreo, Aliasing, Corrección, Error, Capas, Artefactos, Distorsión, Suavizado, Nitidez, Convolución, Sistema, Correlación, Características espaciales, Replicación, Espejo, Parámetros, Calidad, Medida, MSE, PSNR, SSIM, BRISQUE, NIQE, PIQUE.

Abstract (English)

This Bachelor Thesis has been carried out in order to delve into the different interpolation techniques applied to digital images, which are very useful in different areas such as: medicine, scientific studies, commercial applications, biometrics and security…

Given a few basic notions about digital image, and once the most basic bases and tools used in his processing were established, we will fully explore what digital interpolation is and what it is for.

We will take a long, fairly detailed tour of each of the existing digital image interpolation techniques, carrying out a study on all of these and analyzing each of them in a theoretical and practical way.

We will submit each of these techniques to study, being able to see how the results are and thus being able to establish a comparative criticism between each of them. All this will allow us to have different visual results and therefore a first approach to the pros and cons of using one technique or another on different training images.

Once this work is done, we will have obtained all the necessary ingredients for the design of different new algorithms that are offered to us as alternatives to those already established; this will be our principal objective.

Finally, we will evaluate, using both objective and subjective image quality techniques, the results of applying each of the algorithms presented in this work in different image datasets. Through different graphic representations we will be able to see how better or worse the proposed algorithms are in relation to those already invented.

Keywords (English)

Image, Digital, Processing, Signal, Techniques, Interpolation, Systems, Algorithms, Development, Theoretical, Practical, Spectral Analysis, Spatial content, 2D, Signals, Processed, Space, Pixel, Color, Gray, Aliasing, Filters, Filtering, Frequencies, Transformed, Fourier, Neighbor, Bilinear, Bicubic, DFT, DCT, Zero-Padding, Spectrum, Density, Energy, Detail, Fine, Coarse, Decimated, Sampling, Aliasing, Correction, Error, Layers, Artifacts, Distortion, Smoothing, Sharpness, Convolution, System, Correlation, Spatial Characteristics, Replication, Mirror, Parameters, Quality, Measure, MSE, PSNR, SSIM, BRISQUE, NIQE, PIQUE.

Agradecimientos

En primer lugar, quiero agradecer a toda la Comunidad de Investigadores su trabajo e interés por conocer el mundo que nos rodea y crear así nuevas invenciones con el objeto de hacernos la vida más fácil.

En segundo lugar, al Equipo de Dirección de la Escuela Politécnica Superior de Madrid por las instalaciones que deja a disposición de sus alumnos, así como todas las herramientas y equipos informáticos y el contenido bibliográfico de su biblioteca. Todos estos recursos han sido imprescindibles a la hora de realizar este trabajo.

A nuestros profesores, por sus capacidades y su interés en la docencia, por todo el trabajo que hay detrás del material docente con el que nos imparten clases y nos permite desarrollarnos más fácilmente como estudiantes.

Quiero señalar un especial agradecimiento a mi tutor, Jesús Bescós Cano, un excelente profesional que no ha dudado en prestarme todo el tiempo que he necesitado para resolver problemas y dudas surgidas durante la creación de este trabajo. Gracias por haber confiado en mí y en mis capacidades para la confección de este tema por ti propuesto.

No menos importante es el agradecimiento de aquel apoyo que me han brindado tanto familiares como amigos durante todo este tiempo. Todas las dificultades que se me presentaban a lo largo de mi carrera académica dejaban de ser dificultades en el momento que todos ellos me ofrecían la ayuda y fuerzas para superarlas.

Gracias a mi padre y a mi madre por ofrecerme la posibilidad de obtener una formación universitaria, por permitirme disfrutar de muchas facilidades a las que no todo el mundo tiene acceso, así como por los constantes ánimos y, en definitiva, por ayudarme a convertirme en la persona que soy.

A todos mis compañeros, con los que he compartido esta etapa tan maravillosa de la vida. Ya sea entre apuntes, ordenadores o cañas están algunos de los mejores recuerdos que llevaré siempre conmigo.

Por último, quiero agradecer a mi pareja toda la paciencia que ha demostrado tener conmigo a lo largo de todos estos años. Desde mi entrada en la universidad ha sido un apoyo fundamental para mí. Ella ha sido con quien he compartido todas mis lágrimas y logros, quien en incontables ocasiones me ha hecho espabilar y superar muchos muros. No tengo la forma de agradecerle todo esto.

De nuevo gracias, muchas gracias a todos.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 Introducción

1.1 Motivación

Vivimos en un mundo donde el almacenamiento y consumo de información digital supera con creces cualquier tipo de información analógica. Ya sean libros de tapa blanda o dura, periódicos de hoja blanca o de color salmón, imágenes en negativos o instantáneas, incluso música en vinilo o casete… tan solo un reducido grupo de nostálgicos y otros aferrados a las costumbres son los que disfrutan de estos formatos.

Lo cierto es que este hecho ha dado paso a una nueva era para la información, la cual nos abre un mundo de importantes posibilidades gracias al procesamiento digital, algo de especial importancia en la imagen [1].
Si prestamos atención, nos damos cuenta del amplio y variado uso de la imagen digital:

- Educativo, utilizando las imágenes como medio de aprendizaje visual.
- Social, compartiendo contenido para el entretenimiento o en nuestras redes sociales (RRSS).
- Científico, mediante el uso de imágenes para su posterior análisis en aplicaciones, por ejemplo, médicas o de labores de investigación.
- Comercial, pues creamos productos basados en imágenes y basamos campañas de marketing en fotografías de producto.
- Otros, como la seguridad biométrica basada en la captación de imágenes de nuestro rostro (por ejemplo) para el desbloqueo de terminales y dispositivos móviles.

Es innegable el empapamiento constante que tenemos hoy día con la información digital y su procesamiento. Cuando hace unos pocos años disfrutábamos de unas pocas fotografías en familia en momentos puntuales, ahora redes sociales como Facebook manejan un orden superior a los 350 millones de imágenes nuevas cada día [2].
Un gesto tan sencillo como ampliar una imagen para verla con detalle es posible gracias al procesamiento digital, más concretamente gracias a las operaciones de ampliación, rotación, traslación, proyección, etc.… que se aplican mediante técnicas de interpolación de imágenes digitales.
Las capacidades de procesamiento que tienen nuestros dispositivos electrónicos hoy día son muy superiores a las capacidades que estos tenían años atrás, pero típicamente se siguen empleando las mismas técnicas cuando los recursos son muy superiores.

La motivación de este trabajo consiste en explorar los distintos algoritmos y proponer nuevas alternativas a los métodos ya establecidos en interpolación de los datos contenidos en imágenes digitales.

1.2 Objetivos

Esta memoria de TFG pretende aunar las diferentes técnicas de interpolación existentes más extendidas, con el fin de comprender el funcionamiento de cada una y, en cada caso, conocer los resultados que estas nos ofrecen. Además del entendimiento de técnicas ya existentes, uno de los objetivos principales que se pretende, es la creación de técnicas alternativas basadas en el estudio detallado del proceso de interpolación digital que, en la medida de lo posible, nos ofrezcan resultados mejores a los de las más utilizadas.

La mayoría de las operaciones empleadas en procesamiento de imagen básico (ya sea, por ejemplo, el hacer zoom sobre una imagen en un dispositivo móvil) suelen basarse en interpoladores muy simples como es el caso del interpolador por vecino más próximo (Nearest-Neighbor) o interpoladores bilineales y bicúbicos en el mejor de los casos [3].
Estos nos ofrecen unos buenos pero mejorables resultados (tanto en calidad como en coste computacional y tiempo de procesado).

Así pues, los objetivos los reuniremos en cuatro grandes bloques:

- El entendimiento de la imagen digital y las operaciones de interpolación, así como sus características más básicas y esenciales para el entendimiento total de este documento.
- La revisión y explicación de diferentes interpoladores conocidos, su desarrollo e implementación en programación con Matlab, y una aplicación práctica a imágenes de entrenamiento de cara a obtener resultados visuales comparables.
- El diseño, desarrollo y explicación de nuevos interpoladores, su implementación en programación Matlab y la aplicación práctica en imágenes de prueba. En este punto desarrollaremos varias técnicas, pero nos centraremos principalmente en dos propuestas de interpoladores: el primero basado en tratamiento por capas espectrales y el segundo basado en el análisis del espectro local de la imagen.
- Por último, el análisis y comparación de los resultados de medidas de calidad para diferentes conjuntos de BBDD al verse aplicadas cada una de las técnicas (tanto conocidas como propuestas), con el fin de conocer qué tan buena es cada una de ellas.

La implementación en MATLAB de todo lo expuesto en este trabajo se encuentra disponible como material descargable en el Anexo 1.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Capítulo I: Fundamentos y técnicas de base.**

En este primer capítulo entenderemos qué es y cómo funciona la imagen digital y, por otro lado, hablaremos sobre las técnicas de interpolación existentes, tanto espaciales como frecuenciales.

- **Capítulo II: Diseño, desarrollo y explicación de nuevas técnicas.**

En el segundo capítulo se ofrece el diseño y desarrollo de nuevas técnicas: las primeras aproximaciones (que podemos entender como técnicas más simples) y las propuestas finales (las dos técnicas propuestas principales de este trabajo).

- **Capitulo III: Pruebas, resultados y conclusiones.**

El último capítulo recopilará todas las herramientas de pruebas, así como el conjunto de resultados de medidas de calidad de cada interpolador tratado en este trabajo y su interpretación.

2 Fundamentos y técnicas de base

2.1 Conceptos básicos

2.1.1 La imagen digital

Un poco de historia:
La primera imagen digital de la historia surge en 1957 de mano del estadounidense Russell A. Kirsch, y no, no fue a través del uso de una cámara digital pues esta no sería inventada hasta 1975 (figura 2-1).

La obtención de la primera imagen digital fue posible gracias a la invención de lo que sería el primer escáner de tambor. Este dispositivo permitía captar las variaciones de intensidad sobre la superficie de una fotografía analógica, guardando esta información como un conjunto de valores numéricos. Esta idea sería tomada como base para lo que hoy día conocemos como imagen digital [4].

Figura 2-1: Russell A. K. con escáner

Entonces, ¿qué es la imagen digital?:
Una imagen digital no es más que un conjunto de datos numéricos discretos y finitos que nos permiten la representación y reproducción de una figura, objeto, paisaje, etc.… real o no. Así, podemos entender una imagen digital como un conjunto matricial donde cada elemento numérico se corresponde a un valor de intensidad de pixel.

De esta forma, una imagen en escala de grises podría ser representada como, por ejemplo, una única matriz de dimensiones MxN cuyos valores denotasen los valores de gris de cada uno de los pixeles.
Para una imagen a color su representación es análoga, siendo necesarias esta vez tres matrices de iguales dimensiones MxN, donde cada una de ellas vendría representando una capa de color diferente.

Formatos:
Algo muy importante a tener en cuenta dentro del mundo de la imagen digital es el formato en el que vamos a almacenar la información [5]. En este escenario se nos van a presentar conceptos muy diferentes:

- Formato de imagen: Es el medio principal que tenemos para organizar y almacenar la imagen digital y que nos permite posteriormente rasterizarla en un medio de reproducción como una pantalla, proyector o impresora. Existen formatos de imagen que nos permiten almacenar la imagen sin compresión de la información (RAW), con compresión de la información sin/con pérdidas (TIFF/JPEG) o en formato vectorial (SVG).

- Formato de magnitud representada: Consiste en el modo que establecemos para interpretar el color de una imagen. Dicho de otra forma, es el modo que tenemos de

organizar los distintos colores (o grises) con los que representaremos una imagen. Típicamente nos encontramos con formatos de espacio como GreyScale, o de espacio de color como RGB, YCbCr o HSV.

- Formato del tipo de dato: Este es de los más importantes para el manejo de la información que contiene la imagen, pues establece el rango dinámico de los valores de la imagen. Así pues, las imágenes de tipo *double* recogerán valores de punto flotante entre 0 y 1, mientras que las imágenes de tipo *uint8* recogerán valores enteros entre 0 y 255. Por lo general, se recomienda emplear el formato *double* para el procesamiento de información, pues aporta mayor precisión en los cálculos.

Utilidad del procesamiento de imágenes digitales:
Ahora que sabemos cómo es una imagen digital, podemos entender que el hecho de poder tratar imágenes reales como números supone una revolución sin precedentes en el mundo del procesamiento de señales visuales.

El procesamiento de imágenes digitales consiste en la manipulación de la información que estas contienen. La finalidad de esto es muy amplia: ya sea para mejorarla, modificarla con intenciones artísticas o extraer información útil de esta.

Bases de procesado de señales visuales:
El procesamiento o tratamiento digital de imágenes típicamente se realiza [6]:

- Mediante la descomposición de una imagen (señal 2D) en señales 1D. Para una imagen MxN esto lo podemos conseguir tratando la imagen por columnas (obteniendo N funciones unidimensionales discretas de M muestras) o bien tratando la imagen como una función unidimensional donde cada valor de esta corresponde a una posición de pixel que varía en el tiempo (obteniendo 1 función unidimensional discreta de MxN muestras).

- Mediante el análisis frecuencial de la imagen, ya sea mediante la aplicación de transformadas DFT o DCT típicamente, las cuales nos ofrecen una forma distinta de entender la información de la imagen.

- Mediante el uso de sistemas, que no son nada más y nada menos que algoritmos complejos que engloban diversas operaciones funcionales sobre una señal de entrada y ofrecen una salida diferente.

Análisis frecuencial de imágenes:
Estamos acostumbrados a creer que la representación de una imagen únicamente atiende al dominio espacial, donde tendremos muestras repartidas en diferentes posiciones y donde cada valor se corresponderá a una intensidad de color o gris.
Pero lo cierto es que podemos representar las imágenes de otra forma a la que nuestros ojos no están acostumbrados, el dominio frecuencial.

Mediante este tipo de dominio y según Fourier, seríamos capaces de representar una imagen como la suma de series de sinusoides bidimensionales.
Para entenderlo de forma más sencilla, el dominio frecuencial nos permite representar la tasa de cambio de los pixeles de nuestra imagen, esto es, qué tan brusco es o no el cambio de intensidad de uno respecto de sus pixeles vecinos.

Llegados a este punto, debemos diferenciar dos tipos de operaciones que nos permitirán dar el paso de dominio espacial al frecuencial y serán ampliamente empleadas en este trabajo: La transformada discreta de Fourier (DFT) y la transformada discreta del coseno (DCT) [7].

La Transformada discreta de Fourier aplicada sobre señales bidimensionales descompone la señal como un conjunto o suma de funciones exponenciales complejas relacionadas armónicamente. Esta operación es de gran utilidad y ampliamente empleada en aplicaciones donde se requiere un análisis espectral, y la segmentación de información de la imagen en detalle grueso, detalle medio y detalle fino (ver figura 2-2).

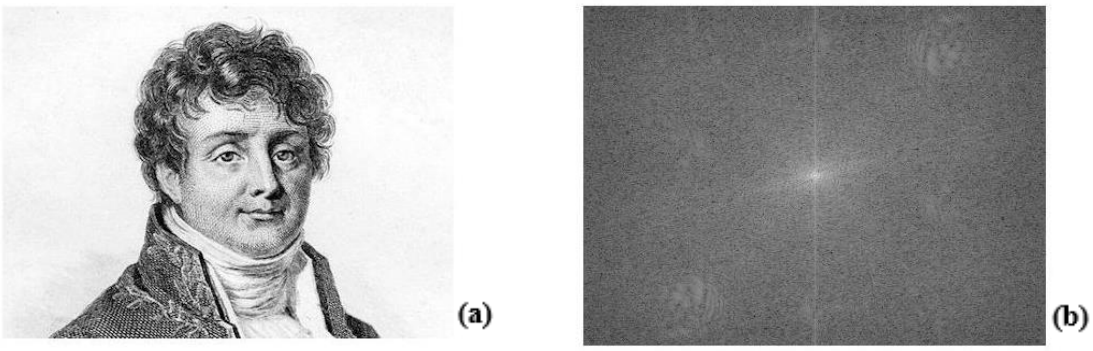


Figura 2-2: Imagen Joseph Fourier (a) y su transformada DFT (b)

Esto va a ser clave para entender muchos de los conceptos tratados en este escrito en los que se involucra el dominio frecuencial:

- El detalle grueso viene determinado por las frecuencias bajas de nuestra imagen, es la parte principal de la que se compone la imagen, da lugar a las formas y fondos más básicos de esta.

- El detalle medio viene determinado por las frecuencias medias de la imagen, y nos da lugar a los contornos y bordes más marcados de la imagen.

- El detalle fino viene determinado por las frecuencias altas de la imagen y nos aporta la información más sutil de esta, es decir, los detalles más pequeños que aportan nitidez a la imagen.

Si filtramos una imagen dando lugar a la eliminación de parte del detalle medio y el detalle fino obtendríamos como resultado una imagen que parece visualmente ‘desenfocada’, mientras que si reforzamos en esta las frecuencias medias y altas obtendríamos una imagen mucho más nítida y aparentemente de mejor calidad.

En cambio, la Transformada discreta del coseno aplicada sobre señales bidimensionales nos descompone la señal como una suma de funciones coseno de valor real. Su operativa nos ofrece un resultado de alta compactación espectral, lo cual hace que se concentre la energía en menos coeficientes. Es muy útil para compresión y para aplicaciones donde queremos acelerar la velocidad de procesamiento.

Su apariencia es diferente a la de la DFT (ver figura 2-3), en este caso las frecuencias más bajas son las que se encuentran dispuestas en la esquina superior izquierda de la imagen y, según nos acercamos a la esquina inferior derecha, nos encontramos con las frecuencias medias y altas.

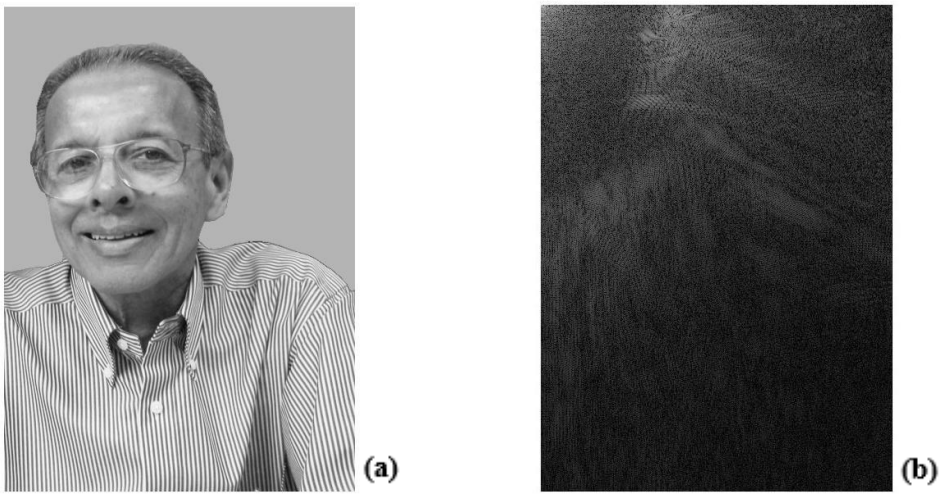


Figura 2-3: Imagen N. Ahmed (a) y su transformada DCT (b)

2.1.2 La interpolación digital

Conceptos básicos:

Si tratamos de dar una definición a interpolación, podríamos decir que esta es la acción de estimar el valor de una determinada magnitud a partir de otras, siguiendo un orden dentro del conjunto que estas forman (ver figura 2-4).

Matemáticamente podemos entenderlo como un método que nos permite la obtención de nuevos puntos de datos dentro de un conjunto discreto de puntos de datos ya conocidos [8].

Si trasladamos este concepto al de imagen digital, la interpolación de imágenes digitales pretende la creación u obtención de nuevos pixeles dado el conjunto original de pixeles que forman la fotografía. Así pues, el objetivo de estas técnicas es tomar una imagen y ofrecer el valor de pixeles de una imagen que ha sido transformada (transformaciones como, por ejemplo, aumentar el tamaño, rotarla, proyectarla…).

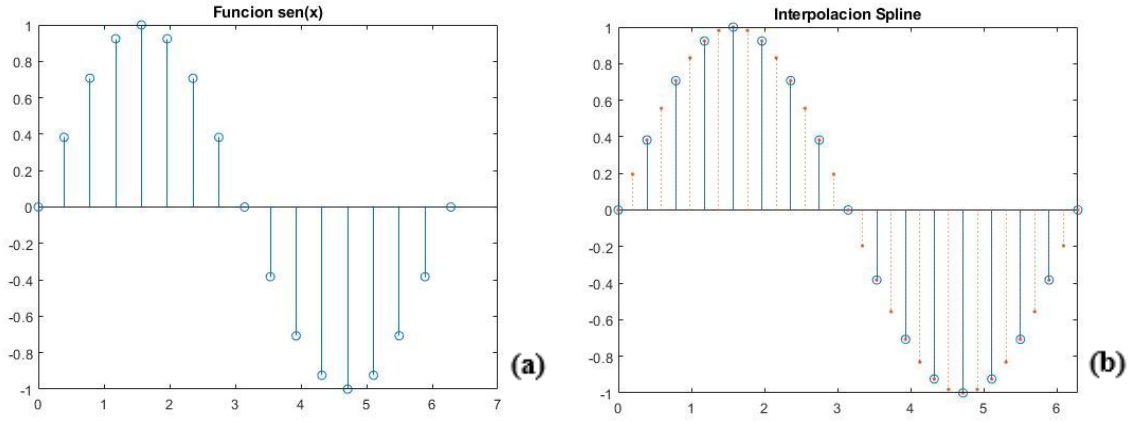


Figura 2-4: Función sen(x) (a) y versión interpolada con Spline (b)

La operación opuesta a esta es el diezmado, la cual pretende ser homóloga a la operación de muestreo. Mientras que un muestreo consiste en el proceso de transformación de una señal analógica a una digital a partir de la toma de muestras, el diezmado consiste en el proceso de la eliminación de muestras de una señal ya discreta con el fin de obtener una versión más pequeña que esta [9]. El concepto de diezmado será empleado de forma puntual a lo largo de este escrito, es por ello que debemos tenerlo también en cuenta.

Aplicaciones de la interpolación

La interpolación como tal nos permite por lo tanto estimar y predecir datos, siendo esto aplicable a cualquier aspecto de la vida real (estudios de negocio, gestión de recursos, incluso en predicciones meteorológicas).

La interpolación aplicada a señales visuales nos permite una gran cantidad de cosas, algunos ejemplos son:

- Las operaciones de transformaciones lineales de imágenes tales como rotación, proyección, estirado y ampliado se consiguen gracias a la interpolación.

- No solo nos interesa la ampliación de imágenes, sino un aumento de la resolución o detalle que esta tiene, esto también podemos tratar de conseguirlo mediante interpolación.

- Nos permite también la restauración de partes 'dañadas' o 'deterioradas' de una imagen digital (inpainting).

- La deformación no lineal de una imagen (conocida como warping) puede ser conseguida también mediante esta.

- La creación de imágenes panorámicas a través de la 'fusión' de varias imágenes.

Comparación entre interpolación 1D y 2D

Digamos que la interpolación es un operador aplicable a cualquier tipo de señal.
El ejemplo mostrado anteriormente en las figuras 2-6 y 2-7 nos muestra la aplicación muy sencilla de una técnica de interpolación sobre una señal unidimensional (1D).
Algunas técnicas 1D típicamente utilizadas son la de Nearest-Neighbor, Lineal o la Cúbica.

Sin embargo, la naturaleza de una imagen es la de una señal bidimensional (2D), algo a tener en cuenta pero que no supone ningún tipo de dificultad. Esto nos obligará a utilizar algunas de las técnicas citadas en el párrafo anterior, pero en su versión extendida al plano bidimensional: 2D Nearest-Neighbor, bilineal y bicúbica (ver figura 2-5).

Esta naturaleza bidimensional de las imágenes en parte entorpece el uso de cierto tipo de interpoladores (haciendo inevitable el aumento del tiempo de procesado), pues estos tendrán que trabajar sobre la imagen teniendo en cuenta las muestras en ambas direcciones.

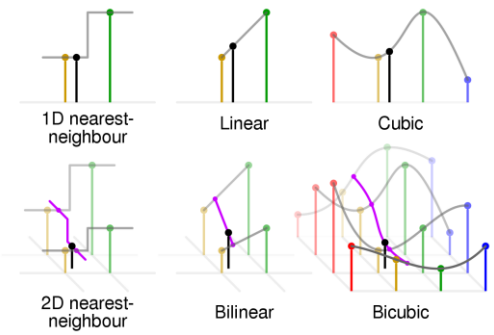


Figura 2-5: Interpoladores 1D vs 2D

Clasificación de técnicas de interpolación
Vamos a establecer dos tipos de clasificaciones diferentes que caracterizan a los interpoladores: según su dominio y según si se adaptan o no a la señal que interpolan.

Según su dominio podemos diferenciar interpoladores espaciales de interpoladores frecuenciales: los primeros trabajarán con los valores de intensidad asociados a cada pixel repartido en la imagen digital, mientras que los segundos trabajarán con los diferentes valores de amplitud asociados a las frecuencias de la imagen.

- Ejemplos de interpoladores espaciales: Nearest-Neighbor, bilineal, bicúbico…
- Ejemplos de interpoladores frecuenciales: interpolación por retículos de celda unidad, Zero-Padding DFT y Zero-Padding DCT…

Según su adaptación podemos diferenciar interpoladores no adaptativos de interpoladores adaptativos [10]: los interpoladores no adaptativos funcionarán de la misma forma para toda la imagen, tratando todos los pixeles por igual, mientras que los interpoladores adaptativos variarán su forma de actuar según la detección de bordes o regiones planas en la imagen. Estos segundos son más complejos y su implementación no suele encontrarse de forma gratuita, pero ofrecen mejores resultados.

- Ejemplos de interpoladores no adaptativos: Spline, Block-Based DCT, Lanczos…
- Ejemplos de interpoladores adaptativos: Qimage, Genuine Fractals… [4]

Artefactos producidos por interpolación
Sin embargo, la utilización de estas técnicas no siempre trae consigo resultados realmente óptimos y siempre distan de ser perfectos. No debemos olvidarnos que estamos 'inventando' nueva información en base a suposiciones y aproximaciones de cálculos matemáticos. La naturaleza de una imagen es completamente abstracta y aleatoria, con diferentes formas, figuras y colores y esto es algo que afectará a los algoritmos no adaptativos de cara a sus resultados.

Así bien, unas técnicas u otras funcionarán mejor o peor sobre un cierto tipo de imágenes diferentes.

Dicho esto, comentaremos tres artefactos (defectos u errores introducidos) típicos de la interpolación [11]:

- Aliasing: Es el efecto más típicamente observado, este consiste en la introducción de bordes con apariencia serrada y patrones de Moiré. Este efecto se produce en las operaciones que implican reducción de muestras (diezmado). En estos casos bastaría con la aplicación del teorema de Shannon Nyquist, o con la aplicación de un filtrado anti-aliasing previo que nos elimine altas frecuencias a fin de evitar este solapamiento espectral).

- Suavizado: Este efecto proviene de ampliar una señal en tiempo o espacio, lo cual consecuentemente produce compresión de información en frecuencia (desaparecen las altas frecuencias, lo que produce este suavizado). Asimismo, la no idealidad de un filtro interpolador que se aplica sobre una imagen, inevitablemente filtrará las altas frecuencias de esta, suavizándola en cualquiera de los casos.

- Halo: Este efecto se produce al tratar de dar solución al problema de suavizado, reforzando las altas frecuencias de una imagen y llevando a cabo un excesivo afilado de la imagen.

2.2 Técnicas de interpolación en el espacio

En este apartado sentaremos las ideas básicas para cada algoritmo, una explicación textual de cómo pueden ser implementados en un programa y, dependiendo del grado de dificultad que el algoritmo tenga, un breve desarrollo matemático de su versión 1D que puede ser extendido a la bidimensionalidad de la imagen. Finalmente, trataremos de realizar un análisis visual de los resultados que cada técnica nos ofrece.

Recordamos que las técnicas aquí expuestas han sido implementadas personalmente, desde cero, en MATLAB e incluidas en el Anexo 1.

2.2.1 Interpolación Nearest-Neighbor (Vecino más próximo)

Definición e ideas básicas para su implementación práctica

Se trata del algoritmo no adaptativo más sencillo, pues solo considera en la interpolación un único pixel para la asignación del valor desconocido. Éste toma el pixel más cercano, dando lugar a la replicación del mismo [12].

La idea básica del algoritmo consiste en crear una imagen de valores nulos con las dimensiones deseadas, una vez conseguida, rellenamos equitativamente esta nueva imagen con los valores originales, obteniendo una imagen 'mallada'.
Finalmente damos valores a los pixeles nulos de malla según su valor vecino conocido no nulo más cercano (ver figura 2-6).

De esta forma y siguiendo la figura, suponemos que (i, j), (i+1, j), (i, j+1) e (i+1, j+1) son los 4 vecinos conocidos al pixel objetivo (u, v). Sus valores vienen determinados por f(i, j), f(i+1, j), f(i, j+1) y f(i+1, j+1). Así, el valor de f(u, v) vendrá determinado por el pixel más cercano, siendo en este caso f(u, v) = f(i, j).

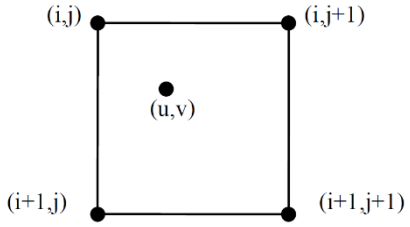


Figura 2-6: Nearest-Neighbor

Desarrollo matemático

Visto desde un enfoque matemático y dado que nos centraremos en operaciones de 'zoom' o ampliación digital para aplicar los interpoladores, desarrollaríamos el siguiente procedimiento:

Siendo el tamaño de la imagen original, $I: RxC$
Siendo el tamaño de la imagen escalada, $J: R'xC'$

El factor de escala para filas (entrada a salida) viene determinado por:

$$S_r = \begin{cases} R/R', & \mathrm{R} > \mathrm{R}' \\ (\mathrm{R}-1)/\mathrm{R}', & \mathrm{R} < \mathrm{R}' \end{cases} \qquad (2.2.1 - 1)$$

El factor de escala para columnas (entrada a salida) viene determinador por:

$$S_c = \begin{cases} C/C', & \mathrm{C} > \mathrm{C}' \\ (\mathrm{C}-1)/\mathrm{C}', & \mathrm{C} < \mathrm{C}' \end{cases} \qquad (2.2.1 - 2)$$

Para cada (r', c') en J, la correspondiente posición fraccional de pixel (r_f, c_f), en I es:

$$(r_f, c_f) = (S_r \cdot r', S_c \cdot c') \qquad (2.2.1 - 3)$$

La posición de pixel entera más cercana (r, c), en I es:

$$(\bar{r}, \bar{c}) = round(r_f, c_f) \qquad (2.2.1 - 4)$$

Y de esta forma obtenemos que nuestra imagen escalada (objetivo) es:

$$J(r', c') = I(\bar{r}, \bar{c}) \qquad (2.2.1 - 5)$$

Dada la sencillez del algoritmo (pues no requiere de la aplicación de procedimientos matemáticos como el cálculo de medias o pesos en función de distancias a varios valores conocidos), es el que menor carga computacional tiene y por lo tanto un menor tiempo de procesamiento.

Análisis de la técnica según resultados visuales

Sometiendo a pruebas imágenes de entrenamiento (`rice.png`) con características diferentes obtenemos los siguientes resultados mostrados en la figura 2-7:

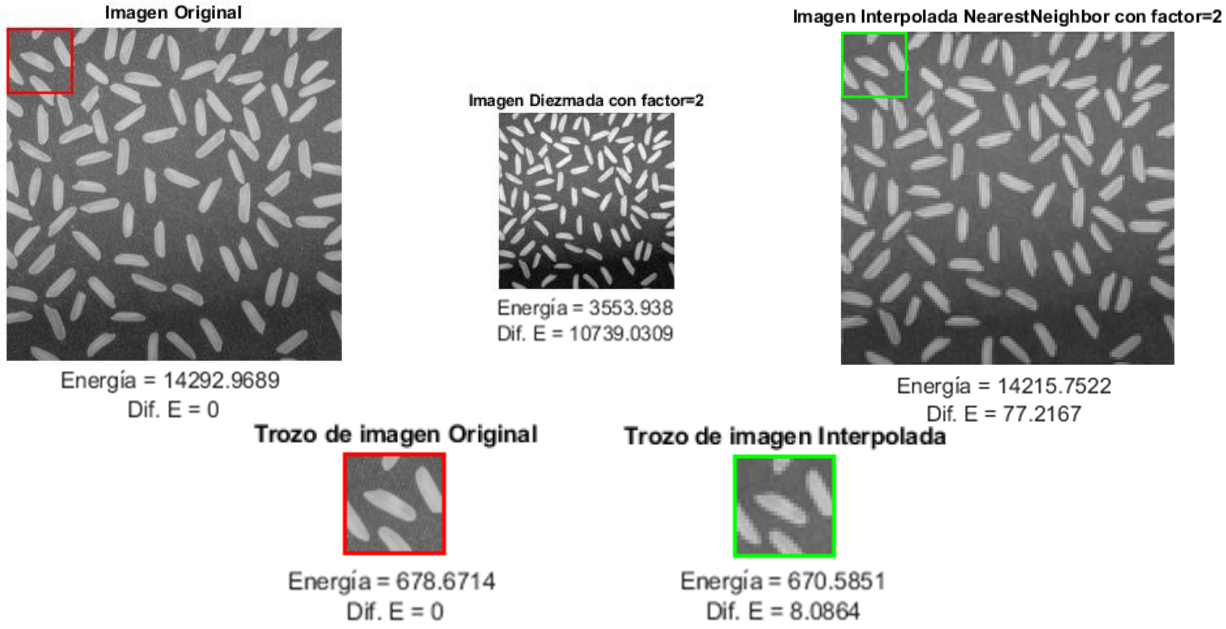


Figura 2-7: Resultado Visual 1 Nearest-Neighbor

Si prestamos atención a las porciones ampliadas, mediante vecino más cercano obtenemos un resultado visualmente pobre. Esta técnica nos da lugar a un resultado con efecto sierra o de pixelado (algo que no debemos confundir con el introducido por aliasing, que en todo caso eliminamos con el filtrado previo al diezmar la imagen).

Obtenemos este efecto en aquellas zonas de la imagen donde existan frecuencias más altas (esto es, donde la imagen tenga mayores variaciones como bordes, regiones ruidosas…). Esto se debe a que, a la hora de asignar un valor a un pixel desconocido, simplemente iteramos el más cercano, obviando el valor de los demás vecinos e introduciendo en algunas ocasiones frecuencias altas irreales.

Sin embargo, que este interpolador sea tan rudimentario y de una aplicación computacional tan sencilla, hace que el tiempo que emplea en procesar la imagen sea muy bajo. Además, si la imagen a procesar es una imagen muy plana, con pocas variaciones o muy correladas y por lo tanto poca información de alta frecuencia, puede ser una muy buena opción.

De este modo, podemos decir que esta técnica funciona muy bien en aplicaciones donde se pretende una alta velocidad de procesado y la imagen es plana.
También puede ser útil si la imagen ya presenta ese efecto de 'pixelado' (como imágenes retro que sean puramente 8bit).

2.2.2 Interpolación Bilineal

Definición e ideas básicas para su implementación práctica
Este interpolador hace uso de un algoritmo no adaptativo más complejo que la técnica interpoladora Nearest-Neighbor, considerando esta vez los cuatro pixeles vecinos más cercanos para la asignación del valor desconocido [13].
Ahora, en vez de replicar el valor conocido más cercano, aplicaremos la operativa de interpolación bilineal, que podemos entender como una interpolación lineal simple en las filas y otra en las columnas de la imagen (como comentamos anteriormente, la naturaleza 2D de la imagen hace que sea necesario tener en cuenta las muestras en ambas direcciones, esto hace que la técnica sobre imágenes sea conocida como Interpolación bi-lineal).

La idea básica del algoritmo es de mayor complejidad que la anterior técnica, pues para crear nuestra imagen interpolada debemos determinar las posiciones que ocuparán los centros de los pixeles de nuestra imagen transformada en la imagen original (ver figura 2-8).

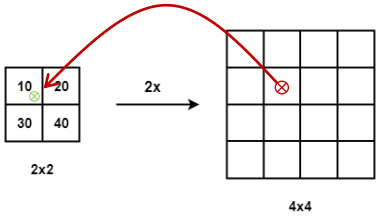


Figura 2-8: Explicación Bil.

Para cada pixel, una vez tenemos la posición de su centro en la imagen original, aplicamos la interpolación bilineal mediante los cuatro pixeles vecinos más cercanos a ese centro.

Esta interpolación bilineal tiene en cuenta la influencia de los cuatro vecinos sobre este centro y la distancia que tiene el punto objetivo con cada uno de ellos, teniendo así mayor influencia sobre la determinación del nivel o intensidad del pixel aquel que más cerca se encuentre en el plano (x,y).

Desarrollo matemático
Visto desde un enfoque matemático, una vez determinada la posición de los centros (esto simplemente consistiría en recalcular las coordenadas al hacer la transformación inversa de zoom, rotación, proyección, etc…) el siguiente paso consiste en aplicar la transformación lineal doblemente:

Así, por ejemplo, para el siguiente caso (ver figura 2-9) en el que tenemos una imagen en escala de grises y queremos calcular el nivel del punto P donde sabemos que:

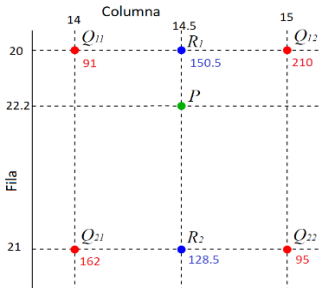


Figura 2-9: Explicación distancias

$$Q_{11} = (20, 14) \qquad Q_{12} = (20, 15) \qquad R_1 = (20, 14.5)$$

$$Q_{21} = (21, 14) \qquad Q_{22} = (21, 15) \qquad R_2 = (21, 14.5)$$

$$P = (22.2, 14.5)$$

Donde hemos definido que $Q_{ij} = (y_q, x_q)$ (2.2.2 - 1)

La ecuación de la interpolacion lineal, considerando $Q = x_q$ un punto de coordenadas cualquiera comprendido en una recta por dos puntos $[x_1, x_2]$:

$$f(x_q) = \frac{x_2 - x_q}{x_2 - x_1} \cdot f(x_1) + \frac{x_q - x_1}{x_2 - x_1} \cdot f(x_2) \qquad \text{(2.2.2 - 2)}$$

En este ejemplo, los valores asociados a cada uno de los pixels (teniendo en cuenta dimensionalidad solo en filas) $Q_{11}, Q_{12}, Q_{21}, Q_{22}$ son:

Primer conjunto $y_q = 20$: $f(Q_{11}) = f(20,14) = 91$ $f(Q_{12}) = f(20,15) = 210$
Segundo conjunto $y_q = 21$: $f(Q_{21}) = f(21,14) = 162$ $f(Q_{22}) = f(21,15) = 95$

De esta forma, aplicando interpolación lineal en filas obtendríamos lo siguiente:

Para el primer conjunto: $f(R_1) = \frac{15 - 14.5}{15 - 14} \cdot 91 + \frac{14.5 - 14}{15 - 14} \cdot 210 = 150.5$

Para el segundo conjunto: $f(R_2) = \frac{15 - 14.5}{15 - 14} \cdot 162 + \frac{14.5 - 14}{15 - 14} \cdot 95 = 128.5$

Ahora si realizamos nuevamente una interpolación lineal en columnas entre ambos valores obtenemos:

$$f(P) = \frac{21 - 20.2}{21 - 20} \cdot 150.5 + \frac{20.2 - 20}{21 - 20} \cdot 128.5 = 146.1$$

Como podemos observar, este algoritmo tiene una carga computacional algo costosa, aunque no preocupante incluso si queremos emplearlo en imágenes de grandes dimensiones, pues los cálculos se limitan a simples operaciones de multiplicaciones como sumas y restas.
Esta carga computacional se debe al hecho de emplear 4 pixeles cercanos, pero gracias a este hecho obtendremos unos resultados mucho más correlados y acordes a la realidad.

Análisis de la técnica según resultados visuales
Sometiendo a pruebas imágenes de entrenamiento con características diferentes obtenemos los siguientes resultados (ver figura 2-10), muy diferentes a los de la anterior técnica:

En un primer caso, hemos tomado la porción de una imagen con un fondo más plano donde las muestras se encuentran más correladas, `threads.png`. Al diezmarla e interpolarla, podemos ver que sí se ajusta mucho el resultado y las energías son casi las mismas.

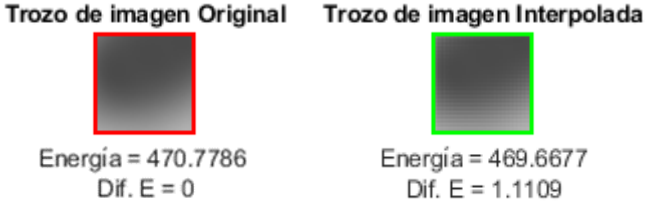


Figura 2-10: Resultado Visual 2 Bilineal

Ahora, probaremos de la imagen de prueba `rice.png`, la cual tiene un contenido muy variado y no tan correlado: la imagen obtenida muestra un notable suavizado, esto se debe a la asignación de valores correlados sobre aquellos pixeles 'inventados' (ver figura 2-11).
De esta forma, la progresión de los valores de la imagen es mucho más 'suave', haciendo que las zonas de mayor variación, como bordes, no sean tan abruptos.

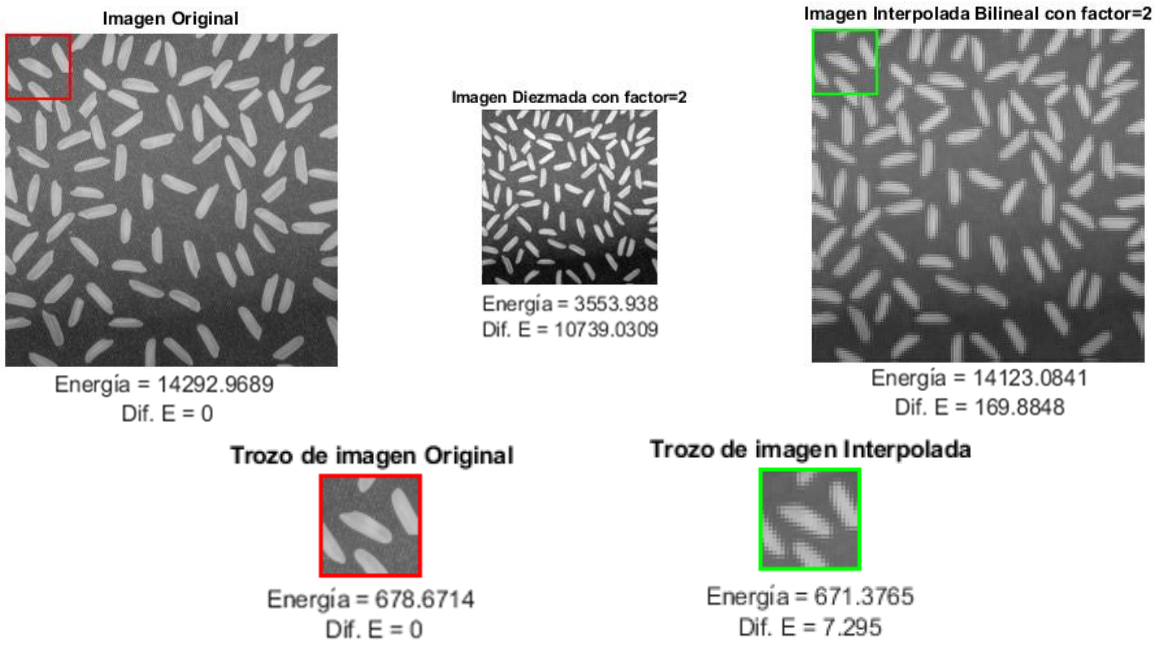


Figura 2-11: Resultado Visual 1 Bilineal

En contraposición a la interpolación por vecino más próximo, vamos a obtener una nueva imagen con menos 'falsas' medias-altas frecuencias, aunque se sigue apreciando cierto error en los bordes.
Esta interpolación podría ser muy útil en su uso sobre imágenes que presenten variaciones muy progresivas y poco marcadas.

Junto con Nearest-Neighbor, esta técnica es ampliamente utilizada por la gran mayoría de aplicaciones de visualización y edición de imagen, reservando así las técnicas más complejas (veremos más adelante) a programas más avanzados como Adobe Photoshop, Adobe Lightroom, Gimp o Helicon Focus. Ofrece un resultado 'aceptable' si lo que queremos es un buen rendimiento computacional, pero con un suavizado general y con contornos todavía 'serrados'.

2.2.3 Interpolación Bicúbica

Definición e ideas básicas para su implementación práctica
Si bien antes dijimos que podíamos ver la interpolación bilineal como una extensión 'bidimensional' de lo que sería una interpolación lineal, en este caso no es diferente. La interpolación bicúbica no es nada más que una extensión 2D de la interpolación cúbica.
Es un algoritmo notablemente más complejo que el anterior, en el que vamos a considerar esta vez hasta 16 pixeles (un conjunto de 4x4 vecinos) para obtener nuestro valor interpolado [14].

Este algoritmo no requiere de una explicación mucho más extensa que el anterior, pues el proceso es exactamente el mismo:

Primero hay que determinar la posición del centro del pixel de interés sobre nuestra imagen original, y una vez tenemos esto, debemos aplicar la interpolación cúbica hasta cuatro veces para las filas, y una más para los resultados obtenidos (ver figura 2-12).

Si comparamos este algoritmo con el de la interpolación bilineal, el bicúbico extenderá la influencia con más puntos y con funciones curvilíneas dando lugar a unos resultados aún más suavizados y mejores, lo cual puede ser muy interesante en aplicaciones que no requieran un uso a tiempo real.

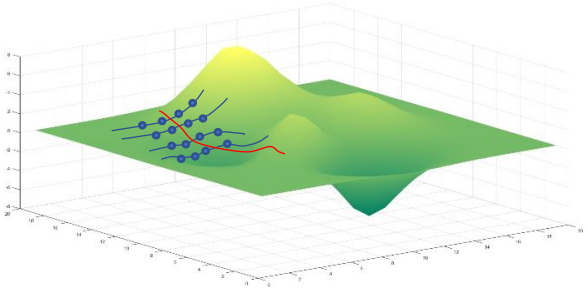


Figura 2-12: Método Bicúbico

Desarrollo matemático

En este caso, el enfoque matemático puede resultar tedioso, así que explicaremos el proceso para calcular una interpolación cúbica (1D) y extenderemos este concepto a la bidimensionalidad de la imagen (esto es, divide y vencerás, igual que en el de la técnica bilineal) [15].

En este caso, si queremos interpolar de forma cúbica un unico valor perteneciente a una curva, necesitaremos 4 valores (4 pixeles) para lo cual necesitaremos esta otra funcion:

Definición de polinomio cúbico: $f(x) = ax^3 + bx^2 + cx + d = Q$ (2.2.3 - 1)

Definición de su derivada: $f(x) = 3ax^2 + 2bx + c$ (2.2.3 - 2)

Resultado para $x = 0$ y $x = 1$ obtenemos:

$$f(0) = d$$
$$f(1) = a + b + c + d$$
$$f'(0) = c$$
$$f'(1) = 3a + 2b + c$$

Lo cual también se puede definir como:

$$a = 2f(0) - 2f(1) + f'(0) + f'(1)$$
$$b = -3f(0) + 3f(1) - 2f'(0) - f'(1)$$
$$c = f'(0)$$
$$d = f(0)$$ (2.2.3 - 3)

Dado que no conocemos la derivada de la función pues solo dispondremos de una lista de valores, suponemos que tenemos los siguientes valores (los vecinos):

$Q_1 = f(x = -1)$ $Q_2 = f(x = 0)$ $Q_3 = f(x = 1)$ $Q_4 = f(x = 2)$

Si queremos interpolar entre Q_2 y Q_3 podemos decir que:

$f(0) = Q_2$ $f(1) = Q_3$

Y para las derivadas a izquierda y derecha, los valores medios:

$$f'(0) = \frac{Q_3 - Q_1}{2} \qquad f'(1) = \frac{Q_4 - Q_2}{2}$$

De esta manera, y despejando según la fórmula, obtenemos los coeficientes:

$$a = -\frac{1}{2}Q_1 + \frac{3}{2}Q_2 - \frac{3}{2}Q_3 + \frac{1}{2}Q_4$$
$$b = Q_1 - \frac{5}{2}Q_2 + 2Q_3 - \frac{1}{2}Q_4$$
$$c = -\frac{1}{2}Q_1 + \frac{1}{2}Q_3$$
$$d = Q_2$$ (2.2.3 - 4)

Sustituyendo valores de estos coeficientes en la ecuación polinómica, obtenemos la función que nos permite determinar el valor de cualquier punto $R = f(x_r)$ sobre la curva:

$$R = f(x_r) = \left(-\frac{1}{2}Q_1 + \frac{3}{2}Q_2 - \frac{3}{2}Q_3 + \frac{1}{2}Q_4\right)x_r^3 + \left(Q_1 - \frac{5}{2}Q_2 + 2Q_3 - \frac{1}{2}Q_4\right)x_r^2 + \left(-\frac{1}{2}Q_1 + \frac{1}{2}Q_3\right)x_r + Q_2 \quad (2.2.3 - 5)$$

Repitiendo este procedimiento en 4 de las filas de nuestra imagen, y de nuevo reaplicando la interpolación cúbica utilizando los resultados obtenidos como datos, obtendríamos el valor objetivo.

La realización de toda esta operativa puede resultar bastante engorrosa y compleja, aunque existen métodos convolucionales con los que conseguimos el mismo resultado, e incluso la posibilidad de realizar el mismo proceso mediante operativas matriciales que nos permiten agilizar los cálculos.

Análisis de la técnica según resultados visuales

Realizaremos pruebas para nuestra imagen de entrenamiento `rice.png` (ver figura 2-13):

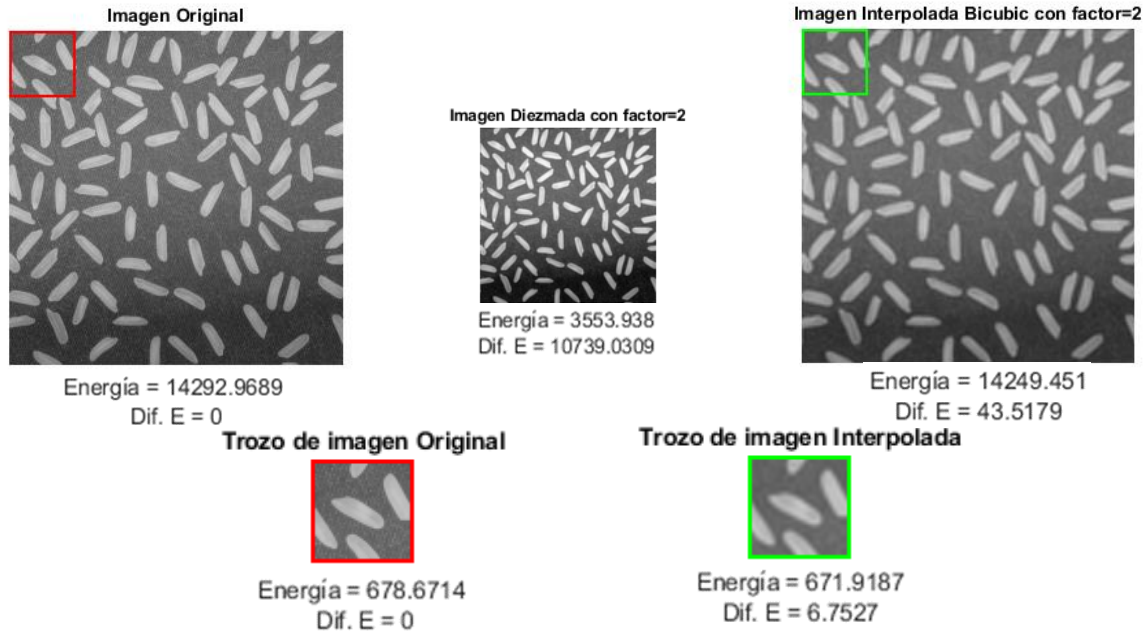


Figura 2-13: Resultado Visual 1 Bicúbico

El resultado es óptimo en lo que a calidad se refiere pues obtenemos una máxima correlación entre pixeles y, a pesar de haber perdido hasta 3/4 partes de la información, somos capaces de reinventarla con bastante proximidad al resultado original. Existe todavía un suavizado de la imagen (algo lógico, una máxima correlación entre muestras hace que perdamos las grandes variaciones y los cambios bruscos, perdiendo así detalle fino), pero los bordes y contornos de la imagen se presentan mucho más ajustados que en las otras técnicas (ver figura 2-14).

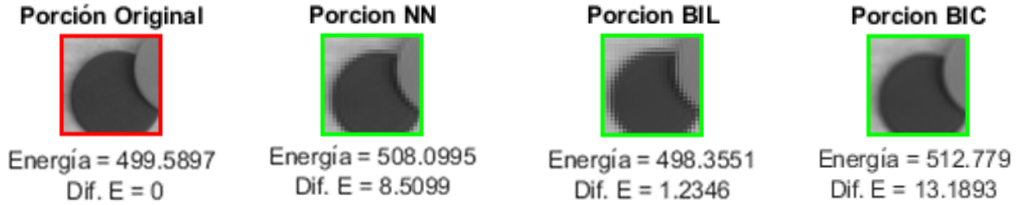


Figura 2-14: Resultado Visual 2 Bicúbico

2.3 Técnicas de interpolación en la frecuencia

Las siguientes técnicas que se van a presentar son técnicas mucho menos extendidas dada su naturaleza frecuencial. Algunas de ellas son muy fáciles de implementar y muy rápidas en cómputo. De nuevo, daremos ideas básicas para cada algoritmo, una explicación textual de cómo implementarlos y, en este caso (ya que no es preciso el desarrollo matemático y este puede ser muy laborioso), un esquema de diagrama de bloques del mismo. Por último, analizaremos visualmente los resultados de cada técnica.

Recordamos nuevamente que las técnicas aquí expuestas han sido implementadas personalmente, desde cero, en MATLAB e incluidas en el Anexo 1.

2.3.1 Interpolación por retículos de celda unidad

Definición e ideas básicas para su implementación práctica

Podemos pensar que trabajar en el dominio espacial con imágenes es mucho más intuitivo y sencillo, pues no nos es difícil entender la relación existente entre un valor de intensidad de pixel y el color con el que se representa visualmente.
Sin embargo, entender la relación entre los valores en frecuencia con su representación visual no es tan trivial, pero lo cierto es que la operativa de diseño de estas técnicas es mucho más sencilla y menos tediosa.

Esta técnica a la que nos referimos como interpolación por retículo de celda unidad (*Unitary Cell Reticle*) se basa principalmente en la creación de una nueva imagen 'desmuestreada' (esto es, una imagen con las dimensiones objetivo con los pixeles originales y otros nulos/ceros dispuestos de forma mayada o en ajedrez según un retículo unitario) y la aplicación de un filtrado normalizado posterior que nos elimine las componentes de frecuencias más altas introducidas por este 'desmuestreo' [16].

Las únicas complicaciones de este método pueden ser dos:

- Este tipo de interpolación funciona según un filtrado frecuencial, luego tendremos que aplicar la DFT sobre nuestra imagen.

- En una DFT, las máximas frecuencias representables tanto horizontal como vertical irán desde -1/2 a +1/2. Este concepto tendremos que tenerlo en cuenta a la hora de filtrar, pues tendremos que aplicar un filtro circular con fc = 1/(2*I) que mostrará una banda de paso plana y que estará centrada en el origen con ancho y alto igual a la mitad de la DFT de la imagen).

Diagrama de bloques explicativo

A partir de ahora, entenderemos la señal de entrada $\psi[n,m]$ (dominio espacial) al sistema como nuestra imagen original, de tamaño MxN.
Nuestro factor de interpolación vendrá definido en el sistema como I, y la señal de salida será definida como $\psi_s[n,m]$, que será la imagen ya interpolada con dimensiones IMxIN.
Debido a que estaremos manejando en prácticamente todo el proceso señales transformadas al dominio frecuencial, estableceremos que:

$$\psi[n,m] \xrightarrow{DFT} \Psi[u,v] \xrightarrow{DFT^{-1}} \psi[n,m]$$

$$\psi_s[n,m] \xrightarrow{DFT} \Psi_s[u,v] \xrightarrow{DFT^{-1}} \psi_s[n,m] \qquad (2.3.1 - 1)$$

Presentamos el diagrama de bloque y lo explicamos paso a paso (figura 2-15):

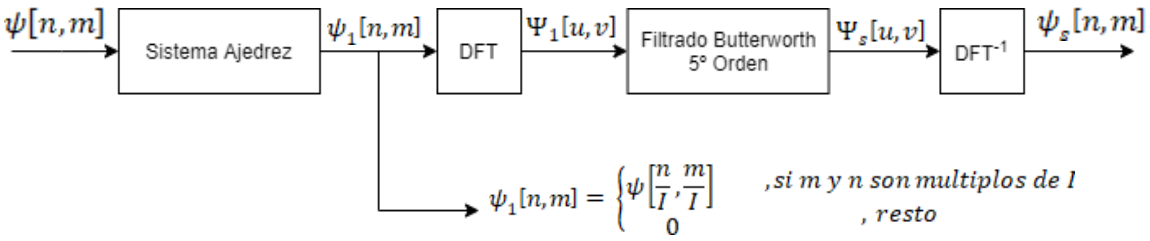


Figura 2-15: Diagrama Bloques Interpolador UCELLRET

- Nuestra imagen de entrada pasa por el 'Sistema Ajedrez', que aplica un aumento de información según el parámetro I en ambas direcciones de la imagen. Esta nueva información serán pixeles de intensidad '0', por lo que si realizamos una representación visual de la imagen obtendremos ese aspecto 'de ajedrez'.

- A esta señal ofrecida le aplicamos la transformada discreta de Fourier, de esta forma podremos aplicar, con una simple multiplicación, el siguiente filtro a nuestra señal.

- El filtro a aplicar será un Butterworth lo más próximo al ideal, por ello emplearemos uno de 5º orden como el mostrado en la figura 2-16, así obtendremos una eliminación progresiva de las altas frecuencias introducidas por el sistema ajedrez (el resultado será mucho mejor que con un paso bajo circular ideal).

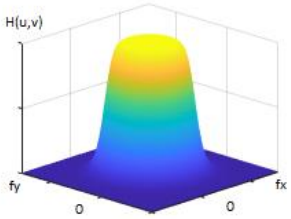


Figura 2-16: Filtro Butt.

- Por último, devolvemos la señal al dominio espacial, obteniendo nuestra imagen de salida lista para ser representada o almacenada.

Análisis de la técnica según resultados visuales

Vamos a someter a análisis tanto el proceso de obtención de la imagen interpolada como el resultado (a continuación, en la figura 2-17, resultados sobre `rice.png`):

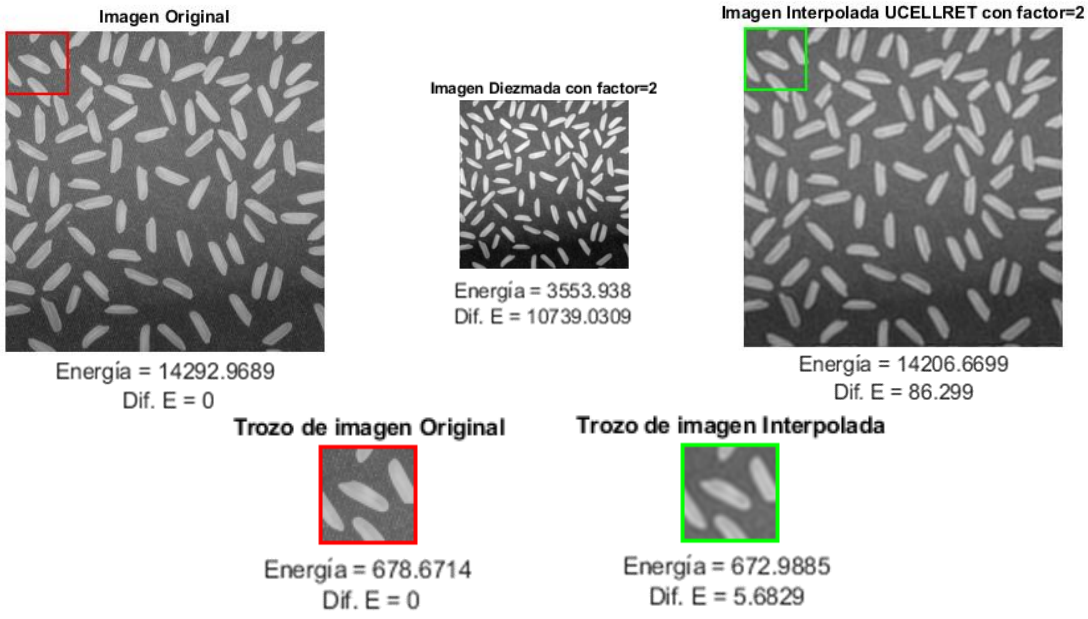


Figura 2-17: Resultado Visual 1 UCELLRET

El resultado se ajusta mucho al ofrecido por la técnica bicúbica, siendo el ofrecido por UCELLRET ligeramente superior en calidad y mucho más eficiente en tiempos de procesado. Por otro lado, tiene su principal hándicap implícito en su funcionamiento: la eliminación total de las altas frecuencias con el fin de borrar ese efecto 'ajedrez' introducido artificialmente; además, esta interpolación no estima altas frecuencias.
Al obtener nuestra imagen ajedrez (insertar ceros), lo que estamos haciendo es comprimir el espectro de la imagen y, al ser la DFT periódica, se produce esta inclusión de réplicas (ver Anexo 2). Esto nos introduce en el espectro de la imagen unas altas frecuencias falsas que eliminamos posteriormente con el filtrado, borrando este efecto ajedrez generado artificialmente.

2.3.2 Interpolación Zero-Padding DFT

Definición e ideas básicas para su implementación práctica
Hagamos la siguiente reflexión sobre la anterior técnica: Al realizar la versión 'ajedrez' de la imagen que queremos interpolar, obtenemos una imagen cuya DFT no es más que la DFT de la imagen diezmada centrada y comprimida. Si posteriormente eliminamos esta información dispuesta de forma periódica, simplemente nos estamos quedando con la misma información contenida en la DFT de nuestra imagen original pero centrada en un plano de las dimensiones objetivo y con ciertos residuos del filtrado.

Esta técnica se fundamenta en esa reflexión: Tomar la DFT de la imagen que queremos interpolar, centrarla en un plano nulo de las dimensiones objetivo y devolver al dominio espacial el resultado (ver Anexo 3) [17].

Sin duda alguna, esta técnica será incluso más rápida que la anterior y podemos adelantar que los resultados serán iguales o mejores.

Diagrama de bloques explicativo
En este caso el diagrama de bloques es muy sencillo (ver figura 2-18):

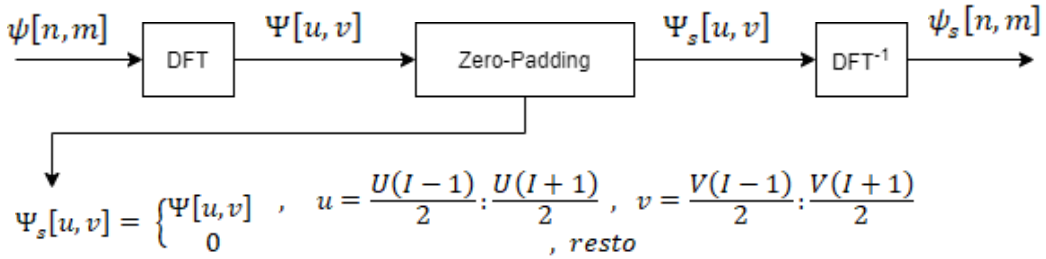


Figura 2-18: Diagrama Bloques Interpolador ZP-DFT

- La señal que queremos interpolar es directamente transformada al dominio espectral mediante el cálculo de su DFT.

- Aprovechamos las características de la DFT para hacer zero-padding: Creamos una nueva imagen espectral de valores nulos y tamaño IUxIV = IMxIN y centramos sobre ella la DFT de la imagen de entrada.

- Por último, deshacemos la DFT y esta señal sería nuestra salida.

Análisis de la técnica según resultados visuales
En la obtención de la imagen de salida, lo primero en llamarnos la atención es la rapidez del algoritmo en su ejecución dada la sencillez que este tiene.
Los resultados de la figura 2-19, como era de esperar, son incluso algo mejores que en las demás técnicas ya tratadas, aunque es necesario prestar una gran atención al detalle.[1]

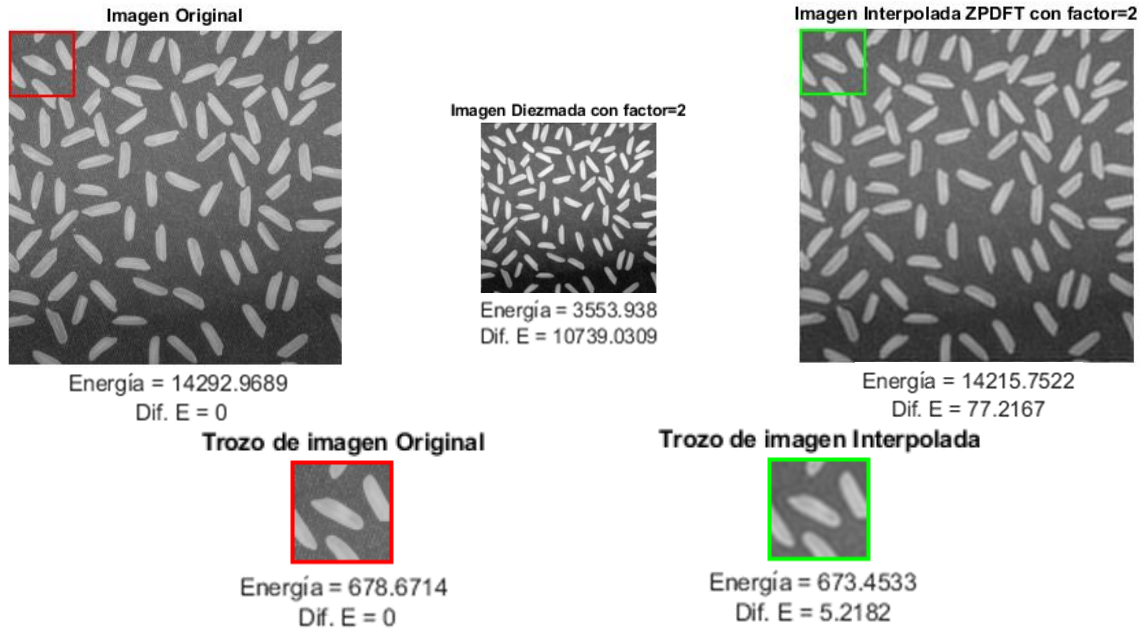


Figura 2-19: Resultado Visual 1 Interpolador ZP-DFT

Siendo estas dos últimas técnicas similares en cuanto al tratamiento del contenido espectral, podemos confirmar que esta técnica tiene un resultado mejor a la anterior y lo podemos confirmar ya que la energía contenida se ajusta (poco más) a la de la imagen original. Esto se debe a que aprovechamos al máximo la información espectral contenida por la imagen de entrada y no empleamos ningún tipo de operación o filtrado que la desvirtúe.
Sin embargo, seguimos teniendo el efecto de suavizado, ya que no conseguimos aportar detalle fino a la interpolación.

2.3.1 Interpolación Zero-Padding DCT y Block-Based DCT

Definición e ideas básicas para su implementación práctica
Estas dos técnicas las tratamos en conjunto ya que ambas trabajan con la transformada discreta del coseno de la imagen de entrada y la segunda técnica es extensión de la primera.

El interpolador Zero-Padding DCT sigue exactamente un esquema de funcionamiento análogo al de DFT: Aplicamos transformada, esta vez discreta del coseno, aplicamos Zero-Padding, y devolvemos la señal al dominio espacial.
En el caso de la DCT, el contenido espectral se dispone de forma diferente (no simétrica): para hacer el Zero-Padding no hace falta centrarlo, simplemente debemos situar en la

[1] Puede consultar la representación visual de la diferencia entre los trozos de imagen para cualquier técnica tratada ejecutando el código MATLAB aportado.

esquina superior izquierda de la imagen nula (de dimensiones IUxIV) la DCT de la imagen de entrada (ver Anexo 4) [18].

El interpolador Block-Based DCT añade dos pasos más: se dividirá la imagen de entrada en bloques o porciones, cada uno de estos bloques será trasformado por DCT, se les aplicará Zero-Padding de forma independiente, los devolveremos al dominio espacial mediante DCT inversa y por último concatenaremos los resultados obteniendo así la salida [19].

Diagrama de bloques explicativo

Dada la sencillez del interpolador Zero-Padding DCT (y su analogía con la Zero-Padding DFT), así como que Block-Based DCT la incluye enteramente[2], pasaremos a explicar directamente esta última técnica (ver figura 2-20):

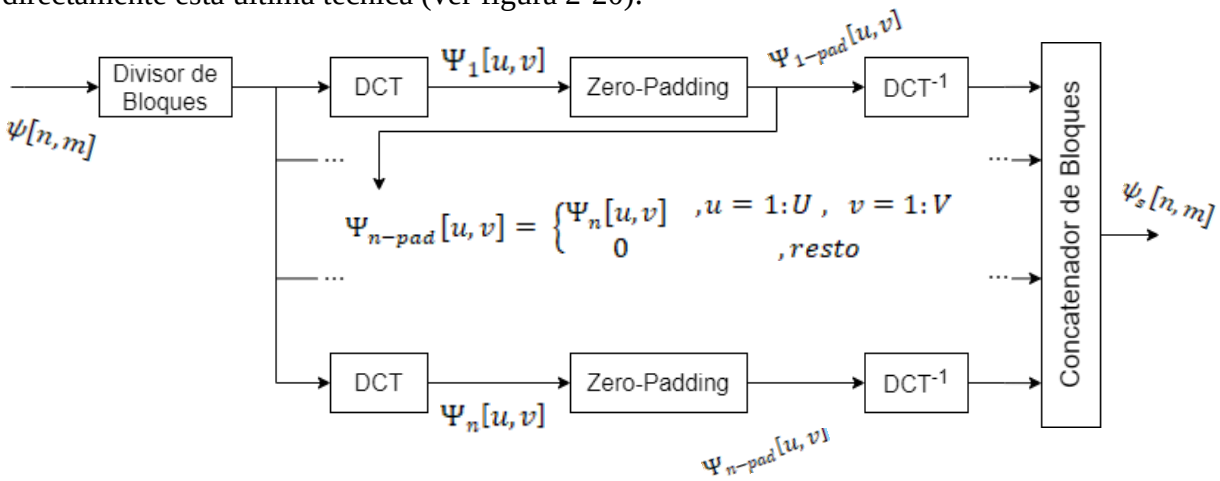


Figura 2-20: Diagrama Bloques Interpolador Block-Based DCT

- El divisor de bloques funciona aplicando ventanas no solapadas de 8x8 muestras de la imagen; cuando las dimensiones de la imagen de entrada no son múltiplos de 8, la información sobrante no la descartamos, sino que la enventanamos con bloques asimétricos.

- Aplicamos DCT sobre cada bloque (asimétrico o no) y zero-padding: Creamos nuevos bloques de valores nulos de tamaño IFxIC, donde F es el número de filas y C el de columnas del bloque, y colocamos sobre ella (empezando desde la esquina superior izquierda) la DCT del bloque.

- Por último, deshacemos la DFT de cada bloque y los concatenamos con el fin de recuperar la señal de salida.

La ventaja de tratar la imagen en bloques es la reducción de la complejidad computacional, así como la capacidad de realizar un análisis de contenido espectral y la naturaleza de contenido del bloque (por ejemplo, si un bloque encierra un fondo o un borde presente en la imagen, esto lo trataremos más adelante).
Como los estándares de codificación de imagen emplean DCTs de dimensión 8x8, hemos establecido este valor para el interpolador, pero puede ser otro: cuanto más pequeña sea la ventana mayor resolución espacial pero menor resolución frecuencial y viceversa.

[2] Entendamos la primera línea de la figura 2-24 como el interpolador sencillo Zero-Padding DCT.

Análisis de la técnica según resultados visuales

En la introducción de la imagen digital (al inicio de esta memoria) dimos una breve explicación de las características de la transformada discreta del coseno. Entre otras cosas, se hablaba de su gran utilidad en compresión de información, dado que compactaba en gran medida la información frecuencial.

Es de esperar entonces que ambas técnicas basadas en DCT (tanto zero-padding y block-based) no nos ofrezcan unos grandes resultados si lo que pretendemos es obtener grandes resoluciones espaciales en la imagen interpolada, pues esta técnica nos servirá únicamente para preservar bien bajas frecuencias.

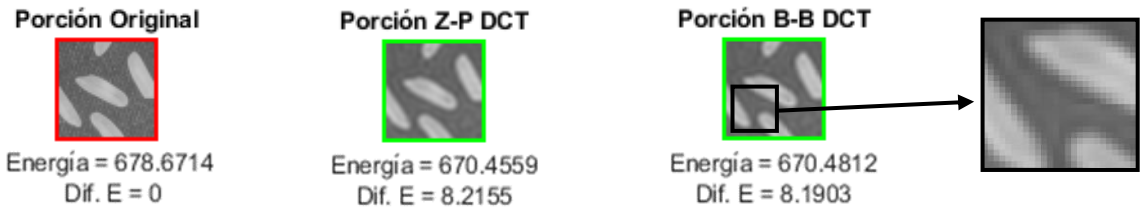


Figura 2-21: Resultado Visual Comparativo 1 DCT

El principal artefacto visible que podemos encontrar en block-based DCT es ese 'mayado' entre bloques adyacentes (conocido como efecto de bloques), introducido debido a su tratamiento independiente (ver figura 2-21). Es un hándicap que, aunque en ocasiones puede parecer desapercibido a simple vista, debemos tener en cuenta. Debemos decidir, por lo tanto, en qué momento nos es interesante disminuir la complejidad computacional sacrificando el resultado con este artefacto.

Veamos cómo se comporta con imágenes ya suavizadas (con muy poca energía en altas frecuencias):

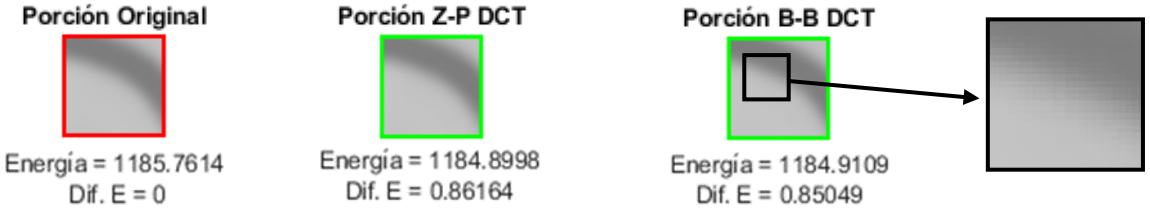


Figura 2-22: Resultado Visual Comparativo 2 DCT

En este segundo caso (figura 2-22), con una imagen relativamente suave, los resultados se ajustan visualmente mucho más a la imagen original y con muy parecida proporción de energía. Destaquemos que, para este tipo de imagen, el efecto 'mayado' entre bloques adyacentes aunque aún existente, es muchísimo menos apreciable.

3 Diseño de técnicas de interpolación

3.1 Aproximación y pruebas iniciales

Hasta ahora nos hemos limitado a explicar algoritmos de interpolación ya existentes y típicamente empleados (en mayor o menor medida) en procesado de imagen digital. Hemos visto su funcionamiento y analizado los resultados que ofrecen, y hemos sido capaces de detectar sus puntos fuertes y débiles. Empleando todo ello, vamos a hablar del diseño de nuevas técnicas propuestas que hemos desarrollado e implementado en MATLAB (recordamos que pueden acceder a ellas en el Anexo 1).
Puede consultar resultados de calidad sobre imágenes de prueba en el (Anexo 5).

3.1.1 Interpolador por combinación Nearest-Bicúbica

Fundamentos de la técnica simple
Esta técnica se ofrece como una nueva alternativa a los interpoladores espaciales existentes.

Anteriormente llegamos a la conclusión de la gran utilidad que tenía Nearest-Neighbor sobre imágenes típicamente planas, con pixeles de valores muy correlados y muy poca variación entre ellos y por lo tanto pocas altas frecuencias. También vimos que esta técnica producía un efecto de 'sierra' indeseado en los bordes de la imagen, dando l un efecto de suavizado que pasa más desapercibido debido a las altas frecuencias que se 'cuelan'.

Por otro lado, la técnica bicúbica obtenía un resultado excelente para imágenes o regiones planas, y un resultado bastante aceptable en bordes (aquí el problema es que incluso los bordes más marcados pasaban a suavizarse debido a que los pixeles resultantes de interpolar siempre iban a estar muy correlados entre sí).

El diseño propuesto consiste en realizar el valor medio entre ambos resultados ofrecidos por los dos interpoladores (figura 3-1):

- De esta forma, las zonas planas seguirán conservándose igual de buenas en el resultado final, ya que los resultados de estas regiones en ambas técnicas son muy ajustados.

- Por otro lado, las regiones con bordes y detalles se verán más corregidas (consiguiendo un punto intermedio entre borde 'serrado' y borde suavizado).

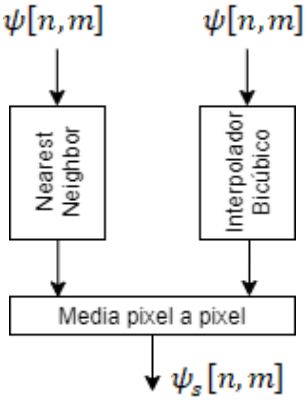


Figura 3-1: Diagrama NN-BIC

Esta técnica, aunque pueda parecer una aproximación al interpolador bilineal, no lo es, y da lugar a un resultado algo más nítido a todos estos interpoladores espaciales (ver figura 3-2).

Figura 3-2: Resultado Visual Comparativo NN-BIC

3.1.2 Interpolador Nearest-Neighbor y corrección Cross-Butterworth

Fundamentos de la técnica simple
Antes intentábamos corregir el efecto de bordes serrados introducido en Nearest-Neighbor por combinación espacial con el resultado de otro interpolador que opera en este dominio. También sabemos que toda aquella adición de información en el dominio espacial supone una variación del contenido frecuencial.

Analicemos la imagen de prueba y su interpolación Nearest-Neighbor, junto con sus DFTs:

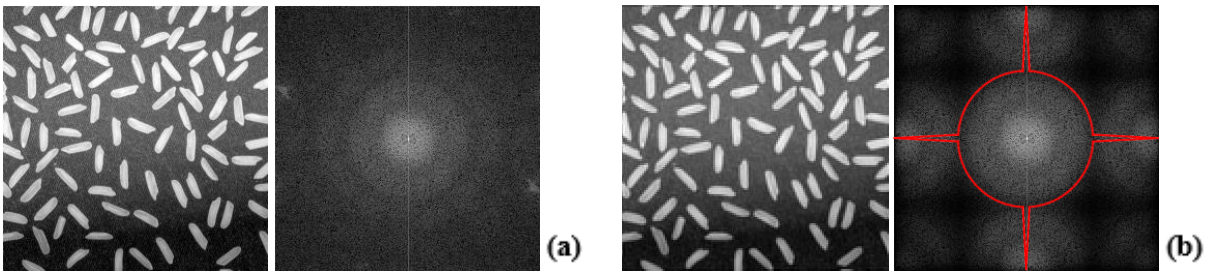


Figura 3-3: Imagen Original y DFT (a), Interpolación NN y DFT (b)

La información que más se asemeja entre la DFT de la imagen original y la DFT de la imagen interpolada por Nearest-Neighbor es aquella que se encuentra dentro de la zona marcada en rojo en la figura 3-3. Esta propiedad se repetirá para cualquier imagen debido a la propiedad de simetría que presentan las DFTs de las imágenes. La información central y la contenida justo alrededor de un eje de ordenadas y abscisas dentro de la DFT suele sernos de gran utilidad.

En este caso hemos diseñado en MATLAB un filtro (al que referenciaremos a partir de ahora Cross-Butterworth) sencillo a partir de un Butterworth de 4º orden: Basta con crear un paso alto a partir de este y luego reorganizar los cuadrantes (ver figura 3-4).

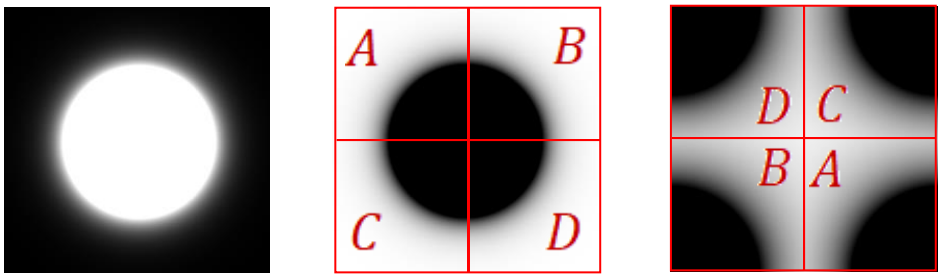


Figura 3-4: Creación del filtro Cross-Butterworth paso a paso

Al aplicar este filtro, nos quedamos con las bajas frecuencias de la imagen, y una parte de las altas frecuencias no redundantes (ver figura 3-5). Esta técnica aún necesita arreglos y el diseño de un filtro mejor.

Figura 3-5: Interpolación Nearest-Neighbor con corrección Cross-Butterworth

3.1.3 Interpolación por composición aleatoria

Fundamentos de la técnica simple
Esta técnica es una primera aproximación, meramente experimental, a uno de los interpoladores complejos que veremos más adelante, siendo así de poca utilidad práctica.

Hemos visto un total de siete interpoladores diferentes ya existentes (tres de ellos que operan en el dominio espacial y otros cuatro en el frecuencial), cada uno de ellos con sus ventajas y desventajas, funcionando así unos mejor que otros en cierto tipo de imágenes.
Así pues, una imagen digital tomada por una cámara fotográfica típicamente suele contener información de todo tipo: regiones planas como el cielo, regiones con bordes como una montaña, regiones con más detalle fino como las hojas de los árboles…

Llegados a este punto pensamos lo siguiente:
¿Qué resultado se nos podría ofrecer si combinamos cada uno de los resultados ofrecidos por estos interpoladores? ¿Por qué no generamos una imagen de salida resultante de realizar una composición aleatoria de los pixeles resultados de cada técnica?[3]

En esta primera aproximación no buscamos realizar ningún tipo de decisión sobre qué pixel es el más conveniente para cada posición de la imagen de salida, simplemente lo estableceremos de manera aleatoria para así saber a qué nos enfrentamos.

La carga computacional de este algoritmo también será uno de los principales contras, ya que procesaremos la misma imagen tantas veces como interpoladores dispongamos (en este caso seis) con sus respectivos tiempos y consumo de recursos (figura 3-6).

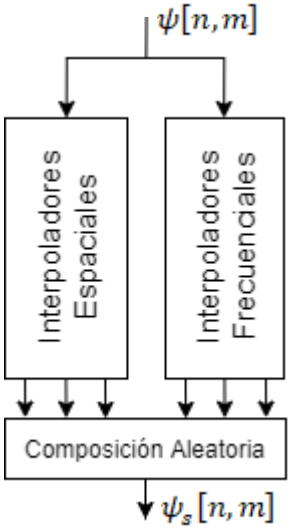


Figura 3-6: Diagrama Composición Aleatoria

Veamos cómo se comportan los resultados:

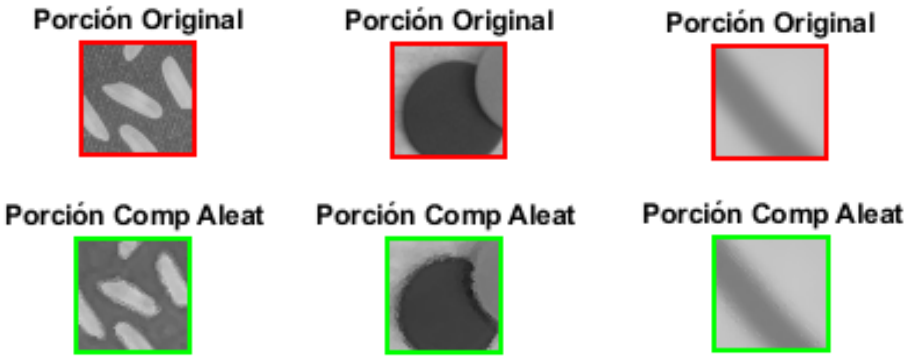


Figura 3-7: Resultado Visual Comparativo Composición Aleatoria

Era previsible que el resultado no fuese óptimo: como era de esperar se produce un efecto granulado sobre la imagen (muy perceptible sobre todo en los bordes, ver figura 3-7).
Aun así, creemos que puede ser conveniente seguir desarrollando este diseño a uno más complejo y con decisión de elección de pixeles, lo veremos a continuación.

3 Dado que Zero-Padding DCT y Block-Based DCT tienen resultados análogos excepto en términos de complejidad computacional, emplearemos Zero-Padding DCT.

3.1.4 Interpolación por decisión binaria

Fundamentos de la técnica simple
Viendo que la composición aleatoria de pixeles da lugar a unos resultados no tan 'desastrosos', vemos que puede ser factible tomar la siguiente vía de diseño: la combinación de resultados de técnicas, pero esta vez condicionadas según decisiones.

En este caso planteamos un diseño más complejo que el anterior, pero de no tan difícil implementación:

- Paso 1: Interpolamos la imagen de entrada mediante dos técnicas dispares.
    - Un interpolador que aprovecha toda la información frecuencial de la imagen de entrada (Zero-Padding DFT) y nos da lugar unos excelentes resultados para las regiones planas o bajas frecuencias.
    - Un interpolador que funcione muy bien en las regiones donde hay contornos o altas frecuencias (Interpolador Bicúbico, o como alternativa, el interpolador propuesto combinación Nearest-Bicúbica que aporta unos bordes más sólidos).

- Paso 2: Aplicaremos a la imagen de entrada una máscara que nos ofrezca una extracción de los contornos más marcados de esta (filtro de desviación estándar) (ver figura 3-8).

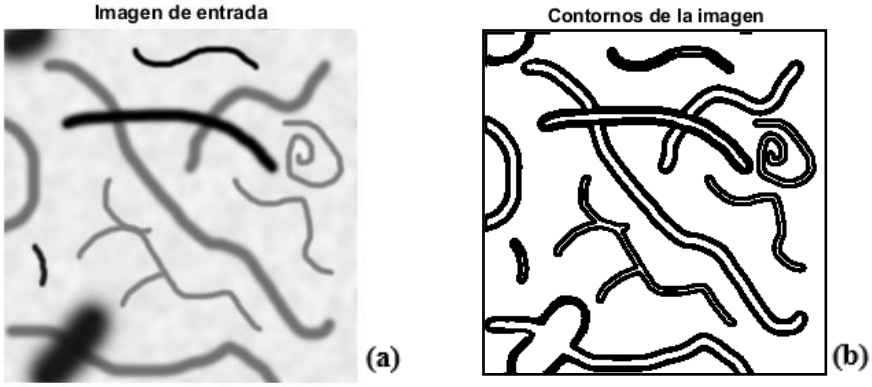


Figura 3-8: Imagen (a) y extracción de contornos (b)

- Paso 3: Crearemos una imagen interpolada con valores nulos de las dimensiones objetivo (IMxIN).

- Paso 4: Recorreremos en un bucle pixel a pixel la imagen vacía y, para cada pixel, realizaremos un mapeo (o cálculo) de la posición de este sobre la imagen original.

- Paso 5: Nos situamos en esta posición para la imagen de entrada enmascarada y, en función de encontrarnos con un valor '1' (blanco - región plana) o '0' (negro - región de contorno), sabremos cual es la técnica más óptima a aplicar, ofreciendo como salida el correspondiente pixel de una de las dos versiones obtenidas en el paso 1.

- Paso 6: Realizamos una corrección de ruido de la imagen mediante un filtro de mediana (cada bloque adyacente IxI puede corresponderse a dos interpoladores

diferentes, dando lugar a una poca correlación entre ellos que en ocasiones puede ser muy fuerte; este filtrado intenta corregir este hecho).

- Paso 7: Aplicamos una máscara de aumento de nitidez con la que intentamos reforzar las altas frecuencias (inevitablemente las altas frecuencias siempre se desvirtúan en todo proceso de interpolación).

Ambos pasos 6 y 7 son, aunque recomendables, totalmente opcionales. Durante los diferentes ensayos realizados en la creación de esta técnica se han obtenido mejores resultados, pero NO han sido tenidos en cuenta a la hora de obtener los resultados (visuales y numéricos) expuestos en esta memoria.

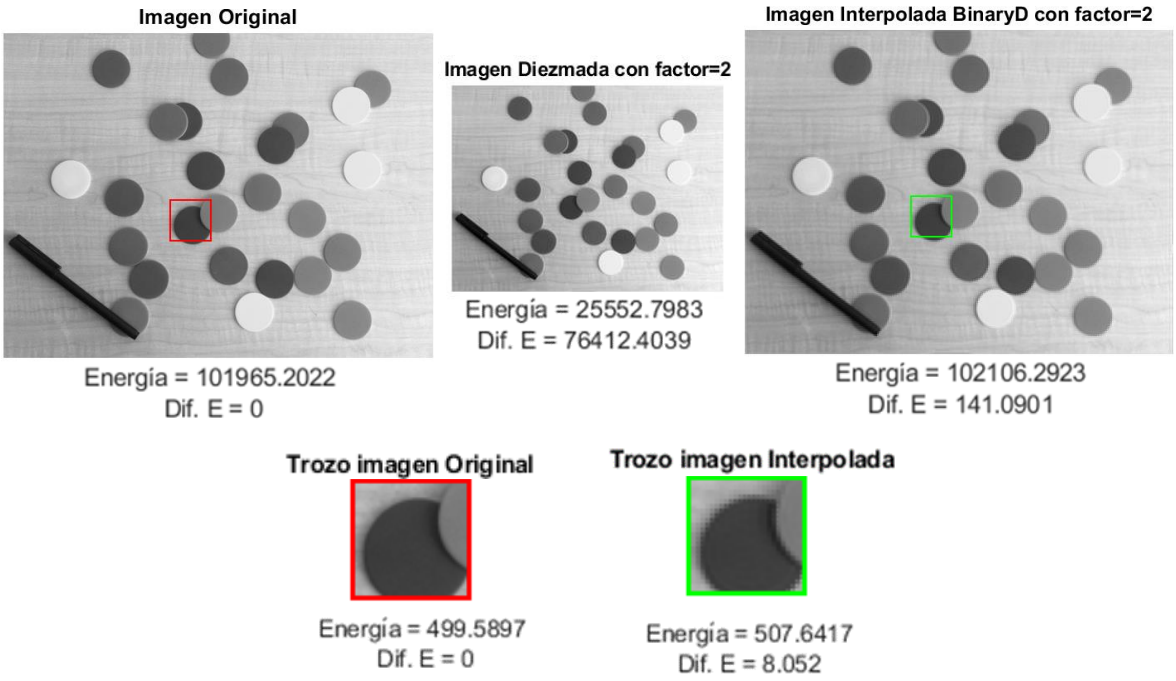


Figura 3-9: Resultado Visual 1 Interpolador por decisión binaria (ZP-DFT y BIL)

A efectos visuales (figura 3-9), los resultados son más que satisfactorios, sobre todo si tenemos en cuenta el funcionamiento binario de la técnica. Además, a pesar de introducir dos técnicas que aportan resultados notablemente diferentes, hemos conseguido combinarlas consiguiendo cierta correlación entre muestras y aun así reforzando bordes.

Acabamos de diseñar una técnica adaptativa (para nada compleja) que funciona de forma satisfactoria a pesar de establecer un condicional que únicamente distingue zonas planas de contornos.
Hasta ahora hemos estado tomando la información espacial y frecuencial de la imagen de entrada con el único fin de interpolarla, pero a estas alturas debemos hacernos la siguiente pregunta: ¿por qué no aprovechar esta información para también analizarla y así determinar características de la imagen? con estas características de la imagen seríamos capaces de establecer aún más condicionales, y con más condicionales podríamos diseñar un sistema hecho a medida del contenido de la información de entrada.

3.2 Propuestas y diseño

El diseño de los anteriores sistemas no han sido más que pequeñas ideas puestas en práctica y desarrolladas de forma paralela a lo que sería uno de los objetivos principales de este trabajo: el diseño y desarrollo de un método complejo de interpolación.
Anteriormente, hablábamos de sistemas simples pues estos tenían un funcionamiento trivial, siendo al fin y al cabo tres técnicas no-adaptativas y una cuarta adaptativa, pero de decisión binaria. Ahora, observaremos el diseño de dos sistemas nuevos algo más complejos (siendo el primero pseudo-adaptativo y el segundo adaptativo multi-decisivo).

Aquí aportaremos ideas de diseño, siendo en el siguiente capítulo cuando los desarrollaremos ampliamente de forma descriptiva y con resultados visuales.

3.2.1 Interpolación por capas espectrales

Fundamentos de la técnica compleja
Como bien explicábamos al inicio de esta memoria, si tratábamos de relacionar el contenido espacial y frecuencial de una imagen observábamos que el detalle grueso (formas y fondos), el detalle medio (bordes y contornos) y el detalle fino (particularidades o sutilezas) se concentraba en las bajas, medias y altas frecuencias de su DFT respectivamente [20] (ver figuras 3-10/11).

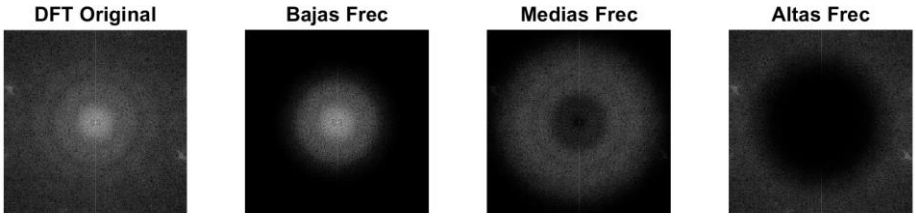


Figura 3-10: DFT y descomposición en bandas

De esta forma, si calculamos la DFT a una imagen de entrada y separamos su información frecuencial mediante un banco de filtros (L-M-H), obtendríamos tres capas de diferentes bandas frecuenciales cuya suma es la imagen original, cuya representación espacial sería:

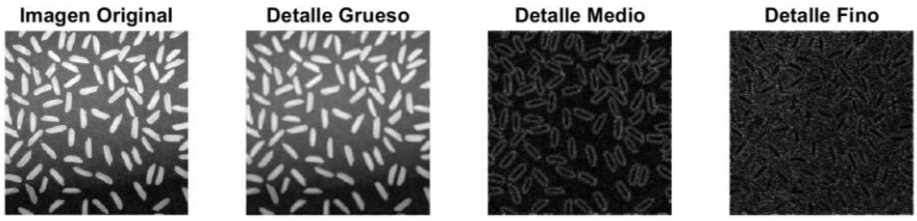


Figura 3-11: Imagen y descomposición en bandas

Si disponemos de diferentes algoritmos que nos ofrecen mejores o peores resultados según el contenido ¿Por qué no interpolar cada capa de información con una técnica diferente?
Si casi todos los interpoladores presentan el mismo problema, que es el suavizado general y la pérdida del detalle, ¿por qué no interpolar con la mejor técnica la imagen, y a este resultado sumarle la interpolación de sus bandas media y alta? Así nos aseguramos un buen resultado general con un refuerzo realista de las medias y altas frecuencias.
Además, la descomposición de la imagen en bandas nos permitirá añadir ganancias sobre estas de una forma muy sencilla si lo vemos conveniente.

3.2.2 Interpolación según análisis de espectro

Fundamentos de la técnica compleja

En esta coyuntura tenemos suficientes herramientas para construir otro algoritmo muy potente:

- Disponemos de una gran batería de diferentes técnicas de interpolación (que operan tanto en el dominio espacial como frecuencial) los cuales nos ofrecen diferentes resultados.
- Además, podemos hacernos una idea de qué tan bien o mal funcionan según el tipo de información que tiene la imagen de entrada.
- Hemos visto cómo se comporta el resultado de interpolar una imagen tratándola como un conjunto de bloques (Block-Based DCT).
- También hemos sido capaces de procesar una imagen pixel a pixel recorriendo una máscara y comprobando valores, dando como salida pixeles interpolados según la decisión de una técnica u otra (Interpolación por decisión binaria).
- Por otro lado, acabamos de conocer la gran practicidad que nos ofrecen los bancos de filtros aplicados en imágenes para, por ejemplo, reforzar bandas. Estos también pueden ser una herramienta muy potente para analizar frecuencias ¿Por qué no usarlos también para saber cómo se distribuyen las densidades espectrales de una imagen?

El siguiente diseño tiene en cuenta el uso de todas estas herramientas, siendo así el algoritmo más sustancial presentado en este trabajo de fin de grado.

Su funcionamiento estará fundamentado en un procesado pixel a pixel de la imagen de entrada con los consiguientes pasos:

1. El objetivo es establecer una ventana de dimensiones impares con su centro en el pixel de análisis (más adelante veremos cómo decidimos cuál es este pixel de análisis).
2. Es importante establecer el tamaño de esta ventana, pues debe ser un valor tal que se abarquen los valores de pixel vecinos suficientes para analizar qué tipo de información rodea al pixel de análisis.
3. Una vez extraída esta ventana, debemos proceder al cálculo de su densidad espectral, la cual someteremos a análisis.
4. Separaremos esta densidad espectral en bandas de frecuencia, con el objetivo de conocer el porcentaje de energía existente en cada una de ellas.
5. En función de estos porcentajes tendremos una idea de cómo es esa ventana y por extensión, qué tipo de información rodea a este pixel.
6. Una vez sabemos si el pixel de análisis pertenece a un fondo, un contorno o una región donde hay muchas variaciones, podremos determinar qué tipo de técnica es la más conveniente.
7. No interpolaremos la ventana completa, pues deseamos obtener resultados correlados a nivel de muestra y no de bloque, además interpolar ventana a ventana supondría una gran carga computacional.
8. Para ello, tendremos varias versiones de la imagen previamente interpoladas con cada técnica típica (Nearest-Neighbor, bilineal, bicúbica, Zero-Padding DFT…), y utilizaremos como salidas a cada pixel los valores de pixel de estas versiones.

4 Desarrollo

4.1 Explicación, detección y corrección de errores en técnicas complejas

En este capítulo explicaremos a fondo las dos anteriores técnicas introducidas en diseño: cómo han sido planteadas desde el inicio de su desarrollo, la evolución de las mismas según la detección y corrección de errores que fueron surgiendo en su implementación, el análisis visual de los resultados y por último su diagrama de bloques. Puede acceder a su implementación en MATLAB consultando el Anexo 1. También puede consultar resultados de calidad sobre imágenes de prueba en el Anexo 6.

4.1.1 A fondo: Interpolación por capas espectrales

Un primer intento (Versión 1)

Desde el inicio de este escrito dejamos constancia de la gran importancia que tiene el dominio frecuencial en el procesado de imágenes. Es una forma alternativa de entender la información que esta tiene y ya hemos visto que nos puede facilitar mucho los cálculos y la operativa.

Los cimientos de esta técnica serán por tanto la descomposición de la imagen en bandas de frecuencia para su procesado independiente.

Para ello, será necesaria la creación de un banco de tres filtros que se repartan respectivamente las frecuencias bajas, medias y altas de una imagen. Este será nuestro primer reto:

Nuestro banco de filtros estará formado por tres filtros circulares Butterworth (paso bajo, paso banda y paso alto), caracterizados por tener una dureza de corte $\beta = 0.75$ (cuanto más se acerque a 1 más ideal es el corte) y distancia entre frecuencias de corte $dFc = 50$.

Siendo la DFT de una imagen $\Psi[u,v]$ (de dimensiones UxV) estableceremos los filtros:

$$H_{LP}[u,v] = 1/(1 + e^{\frac{beta}{4} * Rdist[u,v] - dFc}) \quad (4.1.1 - 1)$$

$$H_{HP}[u,v] = 1/(1 + e^{-\frac{beta}{4} * Rdist[u,v] - dFc * 2}) \quad (4.1.1 - 2)$$

$$H_{BP}[u,v] = H_{HP}[u,v] - 1/(1 + e^{-\frac{beta}{4} * Rdist[u,v] - dFc}) \quad (4.1.1 - 3)$$

Donde previamente hemos definido $Rdist[u,v]$[4] como la transformación de distancias euclídeas sobre una función delta tal que:

$$\delta[u,v] = \begin{cases} 1 & , [u,v] = [\frac{U}{2}, \frac{V}{2}] \\ 0 & , resto \end{cases} \quad (4.1.1 - 4)$$

[4] Este tipo de transformación de distancias se realizará en MATBLAB de una forma muy sencilla mediante la función `bwdist()`.

Una vez creado el banco de filtros, podremos extraer las diferentes capas L-M-H de cualquier imagen que deseemos (ver figuras 3-10 y 3-11). Para aplicarlos basta con una simple multiplicación entre estos y la DFT de la imagen de entrada.
Los resultados son devueltos al dominio espacial mediante DFT^{-1} pudiendo así aplicar cualquier técnica de interpolación de la batería disponible sobre cada capa de forma independiente.

Ahora, vamos a ver qué ocurre si interpolamos cada capa de forma independiente:

- Para la capa de detalle grueso nos encontraremos con información muy correlada y suavizada. En esta capa cualquier técnica puede ser útil y nos va a ampliar muy bien esta información, por lo que escogeremos el algoritmo con menor complejidad y mayor rapidez: la técnica de Nearest-Neighbor.

- En la capa de detalle medio nos vamos a encontrar con los bordes y contornos. Los pixeles que denoten estos contornos se presentarán muy descorrelados con respecto del resto de la información de la capa. Aquí queremos una técnica que nos aproveche al máximo la información de esta capa, ya que queremos unos bordes lo más nítidos posibles, por ello emplearemos Zero-Padding DFT.

- Por último, para el detalle fino (la información más débil) deberíamos llevar a cabo la aplicación de una ganancia y emplear una interpolación lo más ajustada y correlada a esos detalles para reforzarlos, por ello emplearemos la técnica Bicúbica.

Sin embargo, si analizamos los resultados, nos daremos cuenta de un nuevo problema:
Al tratar las imágenes en bandas frecuenciales e interpolarlas por separado en el dominio espacial, se nos producirá un efecto de superposición de información el cual nos provocará un emborronamiento de la información (ver figura 4-1).

Figura 4-1: Resultado Visual 1 Interpolador por capas espectrales (Versión 1)

Aprovechando sus puntos fuertes, propuesta final (Versión 2)
Acabamos de ver cómo la interpolación de bandas frecuenciales en una imagen de forma independiente puede no ser lo más apropiado, pero todavía podemos utilizar la extracción de bandas con otro fin: la corrección o el aporte de ganancia sobre bandas más afectadas.

La técnica Zero-Padding DFT es la que más aprovecha toda la información espectral de la imagen de entrada, pero recordemos que el resultado era suavizado debido al poco peso de las medias y altas frecuencias.

Entonces esta vez, aplicaremos Zero-Padding DFT enteramente sobre la imagen de entrada.
Por otro lado (y de forma paralela), extraeremos las bandas medias y altas de la imagen de entrada (que son las más afectadas en el proceso de interpolación). Nuevamente estas bandas las interpolaremos como en la primera versión (ZP-DFT y Bicúbica).

En el proceso de recomposición, sumaremos todos los resultados, pero aplicando una atenuación sobre la banda media interpolada (no queremos exceder esta componente frecuencial y producir efecto halo); la banda alta tiene una carga tan baja de energía que no es necesario atenuarla.

Figura 4-2: Resultado Visual 2 Interpolador por capas espectrales (Versión 2)

Trabajar de esta forma nos garantiza un resultado visiblemente mejor que el ofrecido por la otra versión (ver figura 4-2).

Diagrama de bloques final

La estructura de esta técnica se puede ver simplificada según el siguiente diagrama mostrado en siguiente figura 4-3:

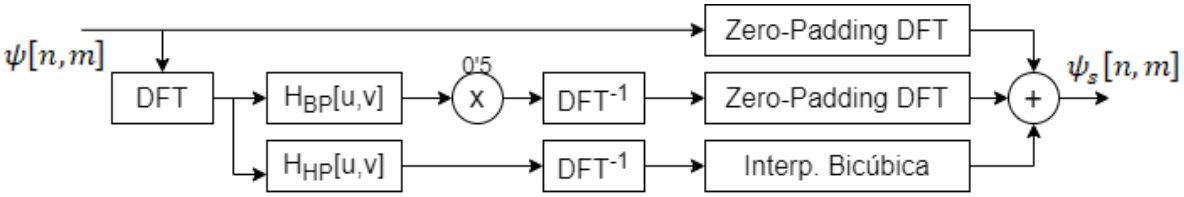


Figura 4-3:Diagrama Bloques Interpolador por capas espectrales (Versión 2)

En este caso los filtros paso banda y paso alto son los mismos que diseñamos anteriormente para nuestro banco de filtros (fórmulas 4.1.1 – 1,2,3,4).

La atenuación aplicada sobre la señal filtrada en su banda de paso ha sido definida como un factor de 0’5, pero este valor puede variar siempre y cuando se encuentre entre los valores [0,1]. Cuanto mayor sea, más visible será el efecto halo, luego tendremos que tener este hecho en cuenta.

4.1.2 A fondo: Interpolación según análisis de espectro

Un paso a paso de la evolución del sistema (Versiones 1 y 2)
El principio de funcionamiento de esta nueva técnica es simple: establecer una ventana de análisis que recorra pixel a pixel la imagen y nos ofrezca una interpolación diferente para cada uno de ellos según el contenido espectral enventanado.

I) Lo primero de todo es diseñar nuestra ventana de análisis, escogimos una ventana de dimensiones impares (de esta forma podremos utilizar como centro de esta el pixel de análisis).
En este caso la dimensión de la ventana cuadrada puede ser variable, pero debemos tener en cuenta que su tamaño condicionará el funcionamiento del interpolador:
Debemos escoger una ventana que nos permita conocer con precisión el contenido espacial de cada zona de la imagen y para ello conviene que sea pequeña (más resolución espacial), pero una ventana pequeña implica menos frecuencias discretas para representar y realizar cálculos. Escogimos una dimensión 19x19 pues nos permite una sintonía espacio-frecuencia, así como establecer un buen número de bandas de análisis.

II) Ahora bien, una vez diseñada la ventana, la centramos sobre cada pixel [n, m] de nuestra imagen de entrada. Esto nos da lugar a un nuevo problema: para los pixeles situados en los bordes de nuestra imagen, la ventana recogerá una parte de valores nulos (ver figura 4-4).

En una primera versión consideraremos la creación de un margen negro para esos valores de pixel en los que la ventana 'sobresalga' de la imagen. Esto tiene un contra que puede afectar en gran medida al correcto funcionamiento de la técnica: el uso de ventanas tiene como objetivo analizar el contenido frecuencial de esta, esto quiere decir que al introducir muestras con valores nulos estamos añadiendo componentes frecuenciales, para nada realistas, con el contenido frecuencial real [21].

Por ello, en segunda versión, nos decantamos por una imagen con un marco de información en 'espejo', donde cada borde del marco es el reflejo de ella misma, de esta forma nos aseguramos mantener mucho más estables las componentes frecuenciales reales cuando la ventana se sitúe en los márgenes de la imagen. Ahora sí, centramos la ventana sobre cada pixel [n, m] de la imagen de entrada con marco espejo.

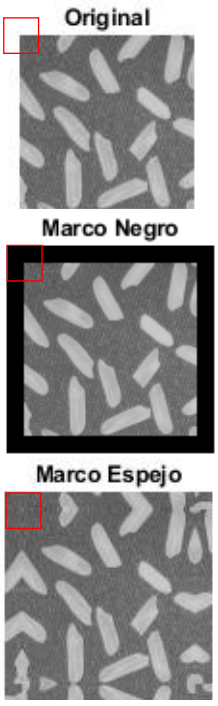


Figura 4-4: Bordes

III) Debido a que queremos realizar un análisis del contenido frecuencial de una ventana, será necesario crear un banco de hasta 6 filtros que se repartan el conjunto de las frecuencias a analizar.
Como disponemos de una ventana de pequeñas dimensiones (19x19), no nos podemos permitir el diseño de filtros circulares de corte no ideal, por lo que estableceremos filtros cuadrados ideales con el aspecto mostrado en la figura 4-5:

Figura 4-5: Banco de filtros análisis

Hemos escogido este diseño debido a la distribución de las energías, pues siempre son más marcadas en la zona media y baja del espectro.

IV) Una vez tenemos los filtros, procederemos a analizar el espectro: para ello calcularemos la densidad espectral de la ventana completa (a partir de ahora nos referiremos a ella como WSD, la cual expresa cómo se encuentra distribuida la energía de la ventana en términos de frecuencia) [22].

Siendo $\Psi_w[u,v]$ la DFT de la ventana de análisis para un pixel, su densidad espectral en dB puede ser calculada como:

$$WSD[u,v] = log_{10}(|\Psi_w[u,v]|^2) \qquad (4.1.2 - 1)$$

V) Con la WSD calculada, la filtramos con el banco definido según la figura 4-5, obteniendo así la densidad espectral de la ventana en cada una de las bandas.

En la primera versión calcularemos de forma muy simple el valor medio de esta densidad espectral en cada banda y comprobaremos cuál es la más alta, conociendo así si es predominante el detalle grueso, medio o fino, y decidiendo la técnica que más se ajuste a este contenido.
Al actuar de esta forma, estamos errando respecto al análisis de la WSD de cada banda, pues no tenemos en cuenta:

- La normalización de valores en amplitud.
- La distribución del peso de las bandas según el ancho de cada una.
- La alta concentración de energías en las bajas frecuencias en imágenes no sintéticas (el detalle grueso siempre está presente en mayor medida que el resto).

Por ello, para la segunda versión NO calcularemos el valor medio de cada banda de forma directa, sino que calcularemos el porcentaje de WSD en cada banda estando este normalizado y con una distribución de pesos según el ancho del filtro en cada banda. En función del valor de cada porcentaje (ver Anexo 7), y teniendo en cuenta que en toda imagen no sintética las frecuencias bajas agrupan porcentajes altos de energía, estableceremos qué técnica emplear:

- Cuando el porcentaje en bajas es muy alto (mucho detalle grueso) empleamos Nearest-Neighbor y Zero-Padding DCT.
- Si el porcentaje en bajas no es tan alto y el de altas sí (mucho detalle fino), empleamos Zero-Padding DFT.
- Si estamos en otra situación (predomina detalle medio) aplicamos bicúbica o bilineal.

VI) En un primer diseño optábamos por aplicar la interpolación a la ventana de análisis completa quedándonos con los IxI pixeles centrales de esta salida como resultado para ese pixel de análisis.
Respecto a esta interpolación de la ventana de análisis, nos vamos a encontrar con varios problemas:

- Aplicaríamos un total de MxN interpolaciones sobre ventanas de 19x19 pixeles, lo que supone una altísima carga computacional.
- Una ventana tan pequeña se traduce en una resolución frecuencial muy escasa, por lo que una interpolación de tipo frecuencial no va a ofrecer un buen resultado en ninguno de los casos.

Así pues, la solución empleada en el segundo diseño, es disponer de hasta 6 versiones diferentes de la imagen de entrada interpolada (una para cada técnica), y escoger como salida al pixel de análisis el correspondiente conjunto de IxI pixeles de una de estas versiones.

VII) Una vez hemos analizado todos los pixeles de la imagen y generado la imagen interpolada es necesario realizar una serie de correcciones sobre el resultado:

Dado que la imagen de salida se obtiene a partir de la concatenación de bloques IxI de distinta naturaleza (fruto de diferentes interpoladores), se produce una descorrelación a nivel de bloque. Para eliminar este efecto, así como cualquier ruido adherido, aplicaremos un filtrado de mediana, el cual mantendrá una mayor correlación a nivel de pixel y de bloque.
Por otro lado, la anterior operación reducirá en cierta medida las altas frecuencias reales que teníamos, por ello es necesario reforzarlas. Para ello, extraeremos una copia de las altas frecuencias de la imagen de entrada mediante un filtro paso alto (definido según la fórmula 4.1.1 - 2), las interpolaremos mediante el interpolador bicúbico y las sumaremos a la DFT de la imagen resultado. Por último, aumentaremos la nitidez del resultado total mediante una máscara de enfoque.

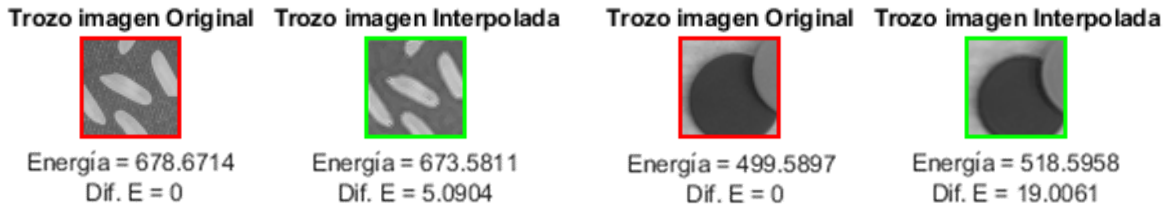


Figura 4-6: Resultado Visual Interpolador según análisis de espectro (Versión 2)

Aparentemente esta técnica funciona muy bien (figura 4-6), aunque es necesario depurarla y perfeccionarla ya que en algunas zonas concretas de la imagen se pueden producir artefactos que no consiguen eliminarse completamente, más adelante veremos qué solución final ofrecemos.

Diagramas de bloques (Versiones 1 y 2)
Debido a la alta complejidad de su transcripción en términos matemáticos, aportamos diagramas más sinópticos que resumen el funcionamiento de los sistemas anteriores.

Diagrama explicativo para la versión 1:

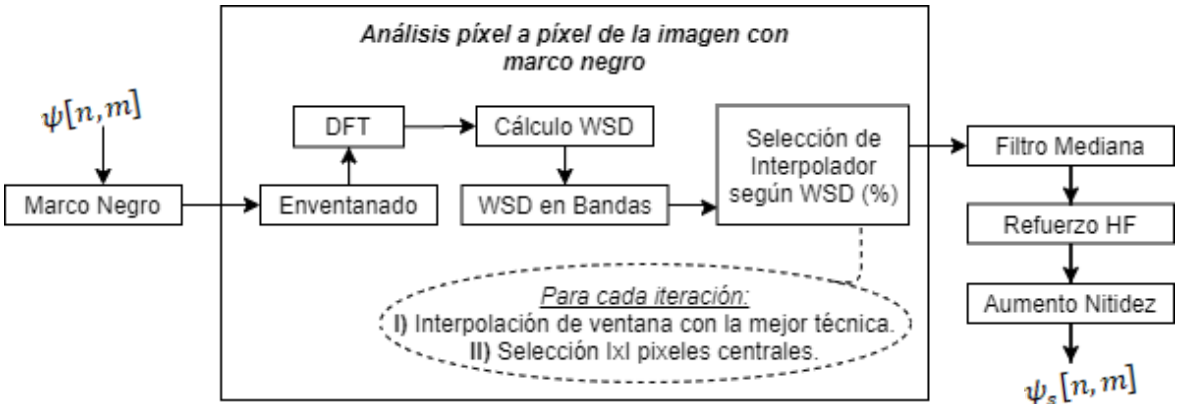


Figura 4-7: Diagrama Interpolador según análisis de espectro (Versión 1)

Diagrama explicativo para la versión 2:

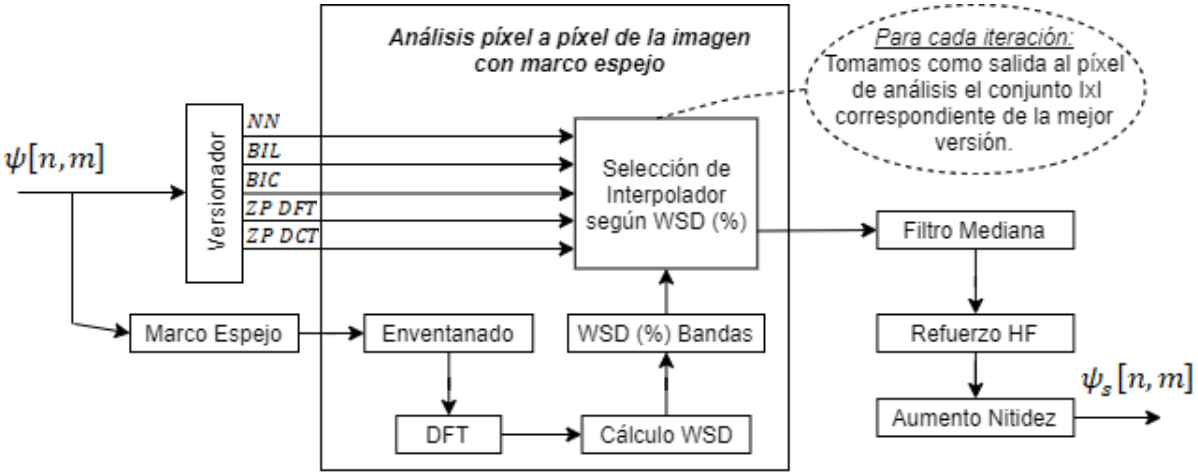


Figura 4-8: Diagrama Interpolador según análisis de espectro (Versión 2)

Esta segunda versión, aunque a primera vista mucho más compleja, nos reduce hasta 16 veces el tiempo de procesado para la misma imagen, obteniendo incluso mejores resultados.

Propuesta final (Versión 3)

Llegados a este punto nos damos cuenta de algo fundamental y básico en un interpolador: su posibilidad de uso en diferentes operaciones (como por ejemplo traslación, rotación, proyección…). Esto se debe a que todo proceso de interpolación actúa de la siguiente manera sobre una imagen (ver figura 4-9):

La operación sobre la imagen, que no es más que una transformación cualquiera (en nuestro caso una ampliación), sitúa los pixeles de la imagen original a unas posiciones que pueden ser no enteras (en verde).

El objetivo de la interpolación es determinar el valor de los pixeles en posiciones enteras (en rojo). Para ello, debemos recorrer <u>nuestra imagen transformada</u> pixel a pixel. Para cada pixel, debemos conocer qué posición le corresponde en la imagen original (esto lo conseguimos deshaciendo la transformación), y una vez sabemos esta posición interpolar el valor.

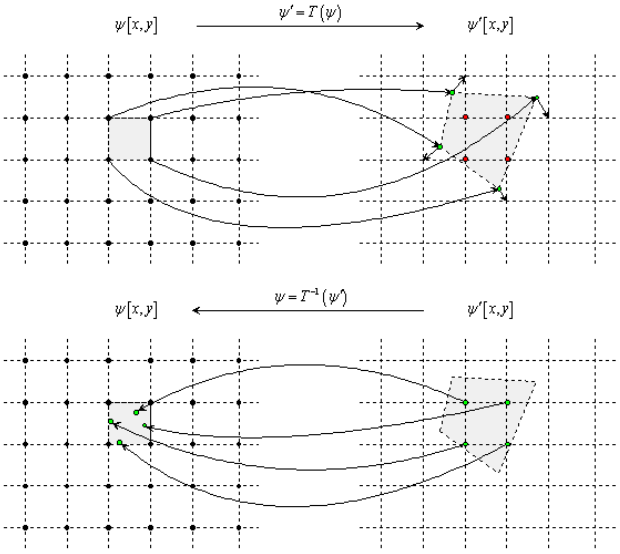


Figura 4-9: Funcionamiento general interpoladores

Nuestra función propuesta ofrece una interpolación sujeta exclusivamente a la ampliación de la imagen (pues directamente recorremos la imagen de entrada y no las posiciones correspondientes en esta según la imagen trasformada, y además ofrecemos como salidas bloques IxI que en ningún caso nos servirán para una interpolación según un factor no entero).

Por ello, es necesario establecer el siguiente diseño final:

I) Escogemos nuestra ventana de análisis, de 19x19, igual que en las anteriores versiones.

II) Creamos la imagen con marco espejo a partir de la de entrada, de la misma forma que en la anterior versión. Antes de centrar la ventana sobre esta imagen y empezar a recorrerla debemos añadir unos nuevos pasos.

III) Creamos una imagen de salida de retículos de celda unidad (imagen ajedrez, esto es hacer un mapeo de la información original sobre las posiciones enteras de la imagen salida según la transformación), la cual recorreremos pixel a pixel (en iteraciones [i, j]). Cuando nos encontremos con un pixel sin valor asignado, deshacemos la transformación (en nuestro caso la ampliación por ese factor de interpolación), obteniendo la posición que le corresponde en la imagen original.

IV) Ahora sí, centramos la ventana de análisis en esta posición de la imagen con marco espejo. Calculamos el WSD de la imagen.

V) Lo separamos en bandas con el banco de filtros y hallamos el % WSD en cada banda de la misma forma que en la anterior versión; una vez sabemos esto, elegimos la interpolación más óptima según ese valor.

VI) Por último, para esa iteración [i, j] escogemos como salida el pixel [i, j] de una de las versiones previamente interpoladas.

Este método, a diferencia de las versiones 1 y 2, y al seguir el esquema de la figura 4-9, nos ofrece un interpolador funcional con cualquier tipo de operación y permite la interpolación con factores no enteros. Otra ventaja es que no ofrece efecto bloque ya que las salidas son pixel a pixel, por lo que no es necesario llevar a cabo correcciones sobre el resultado, siendo ese superior (ver figura 4-10).

Figura 4-10: Resultado Visual Interpolador según análisis de espectro (Versión 3)

Diagrama de bloques final (Versión 3)

El diagrama de implementación final tiene el siguiente aspecto (figura 4-11):

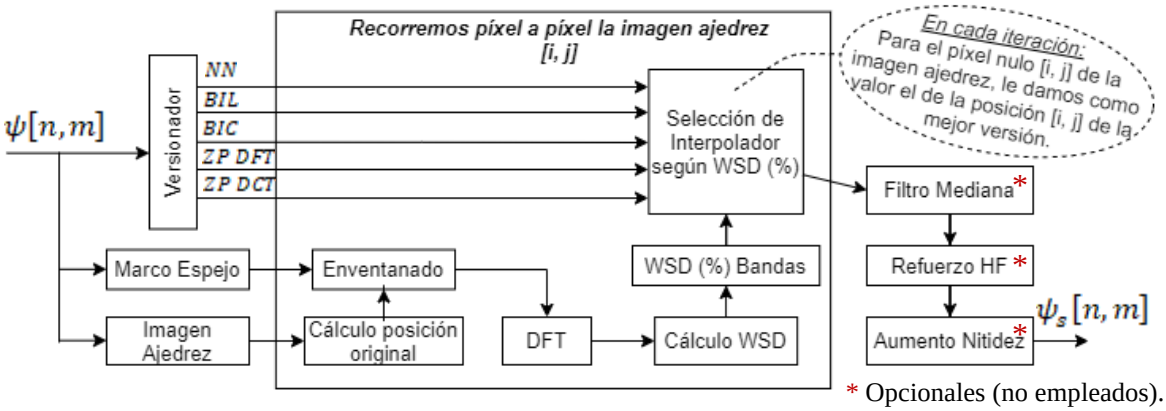


Figura 4-11: Diagrama Interpolador según análisis de espectro (Versión 3)

5 Integración, pruebas y resultados

5.1 Banco de pruebas, dataset de imágenes y medidas de calidad.

Diseño e implementación fichero de pruebas

En la creación de todos estos operadores sobre imágenes digitales hemos utilizado MATLAB como herramienta de programación.

El banco de pruebas que hemos implementado para obtener todos los resultados presentados en este apartado tiene una triple funcionalidad a selección del usuario:

- La lectura de las imágenes de un dataset y la generación de sus versiones diezmadas (parámetro de diezmado fijado en 2, puede ser modificado en el código) con filtrado aliasing previo o no.
- La generación de la versión interpolada de estas imágenes diezmadas a elección del usuario.
- La generación de las medidas de calidad (según medidas de referencia completa o sin referencia) sobre cada técnica a elección del usuario.

En su ejecución se desplegará un pequeño menú donde podremos navegar de forma muy sencilla entre las diferentes funcionalidades. Tras la selección de la funcionalidad, todo resultado (versiones diezmadas, interpoladas o medidas de calidad) será almacenado en diferentes ficheros de forma automática dentro de la carpeta contenedora del código. Todo código MATLAB se encuentra incluido en el Anexo 1 y puede descargarse para su uso.

Generación de un dataset de imágenes de prueba

La selección de una gran batería de imágenes es algo esencial para obtener resultados representativos. Debido a la gran extensión de código que disponemos (hasta un total de 7 técnicas existentes y 6 técnicas propuestas), hemos decidido realizar un análisis sobre imágenes en escala de grises por simplicidad de procesado (ya que la aplicación de estas técnicas sobre imágenes a color es típicamente por capas, multiplicando hasta en tres veces el número de operaciones totales a realizar).

Hemos escogido una selección de la base de imágenes empleada en el libro 'Digital Image Processing, 3rd Edition' de Rafael C. Gonzalez y Richard E. Woods [23] la cual es de libre uso en propósitos de investigación y educativos. Esta presenta hasta 320 imágenes con contenido de gran diversidad (fotografías, sintéticas, con/sin ruido adherido…).

Medidas de calidad de imagen

Para conocer qué tan buenos o malos son los resultados obtenidos a la salida de nuestros sistemas de interpolación es necesario realizar el cálculo de parámetros de calidad sobre imagen (hasta ahora solamente hemos empleado diferencia de energías, algo útil pero no suficiente), utilizaremos los siguientes (más información sobre cada uno en el Anexo 8):

- Medidas de calidad de referencia completa: para calcularlos estos comparan dos imágenes (la original y su versión diezmada-interpolada). Emplearemos las medidas de MSE, SSIM y pSNR [24].
- Medidas de calidad sin referencia: estas determinan la calidad de la imagen según su propio contenido (versión diezmada-interpolada). Emplearemos las medidas BRISQUE, NIQE, PIQE [25] [26].

5.2 Análisis de los resultados de calidad obtenidos

Llevar a cabo un control exhaustivo sobre los resultados obtenidos a partir de distintos medidores de calidad es algo crucial para saber si el funcionamiento de una técnica u otra es correcto o tiene deficiencias. Aquí se presentan resultados sobre el dataset anteriormente mencionado[5], aunque se aconseja, para trabajos futuros, el uso de otros datasets más amplios con el fin de contrastar nuevos resultados. Puede consultar más resultados en el Anexo 9.

5.2.1 Análisis de resultados: medidas de referencia completa

Técnicas existentes

El valor medio de MSE obtenido para la base de datos de imágenes con cada técnica es el mostrado en la tabla 5-1.
Para las técnicas existentes, la que ofrece mejores resultados (un menor error) es la técnica Zero-Padding DFT con un valor de 225,989. Muy seguida a esta se encuentra el interpolador por retículo de celda unidad, algo que cobra sentido debido a la gran similitud en su funcionamiento.

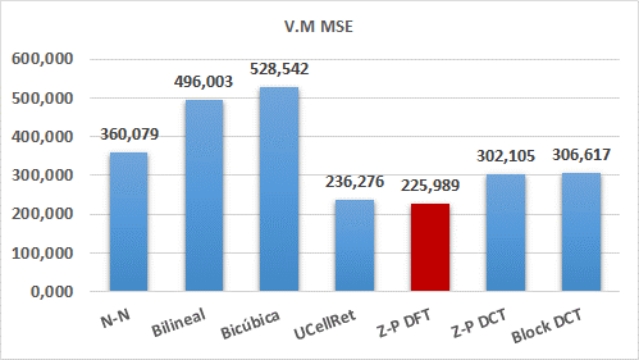


Tabla 5-1: V. M. Error Cuadrático Medio (E)

En el caso del valor medio del SSIM obtenido y mostrado en la tabla 5-2, tenemos mejores resultados (un mayor índice) para la técnica de interpolación por retículo de celda unidad, con un valor de 0,87447, y en esta ocasión le sigue Zero-Padding DFT.
También podemos ver cómo, en general, los resultados de las técnicas frecuenciales se ajustan mucho (a tan solo unas pocas centésimas las unas de las otras).

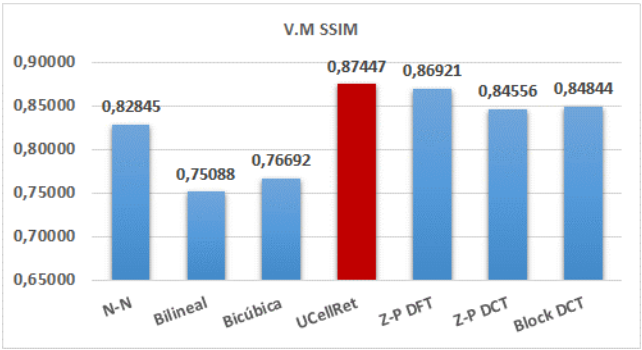


Tabla 5-28: V. M. Índice de Similitud Estructural

Los valores más tipicos para una buena relación señal-ruido pico suelen pertenecer al rango de los 30 y 50 dB.
En este último caso, las técnicas con mejores resultados son los de Zero-Padding DFT e interpolación por retículos de celda unidad (tabla 5-3), algo que era de esperar ya que estos valores dependen directamente del valor MSE.

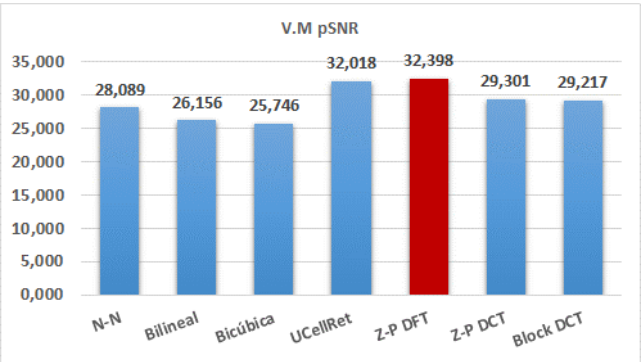


Tabla 5-3: V. M. Relación Señal-Ruido Pico (E)

[5] Empleada en 'Digital Image Porcessing, 3rd Edition', Rafael C. Gonzalez y Richard E. Woods.

Técnicas propuestas

Si ahora analizamos los resultados para las técnicas propuestas, nos damos cuenta de aquella que arroja el mejor valor medio de MSE (tabla 5-4) es el interpolador según análisis espectral y el interpolador por capas espectrales (ambos en sus versiones finales). Además, estos resultados se ajustan mucho a los ofrecidos por las mejores técnicas existentes, algo que es una muy buena señal de cara a la calidad obtenida.

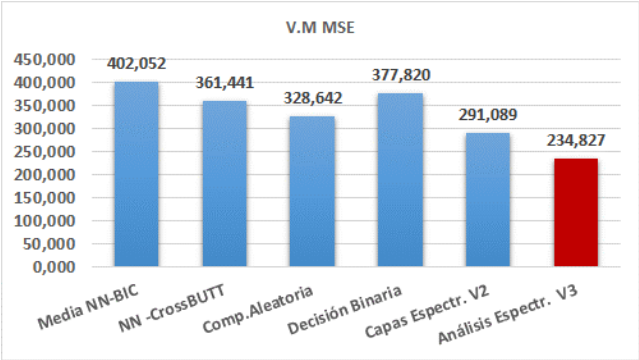


Tabla 5-4: V. M. Error Cuadrático Medio (P)

Nos encontramos con una situación muy similar para el SSIM (tabla 5-5), ya que el interpolador según análisis espectral ofrece un muy buen resultado respecto a las demás técnicas, seguida del interpolador por capas espectrales (obviamos el resultado aportado por la técnica de composición aleatoria pues este resultado es meramente fortuito y diferente en cada ejecución). Todas presentan valores por encima de 0,81, algo muy positivo si lo comparamos con los bilineales y bicúbicos.

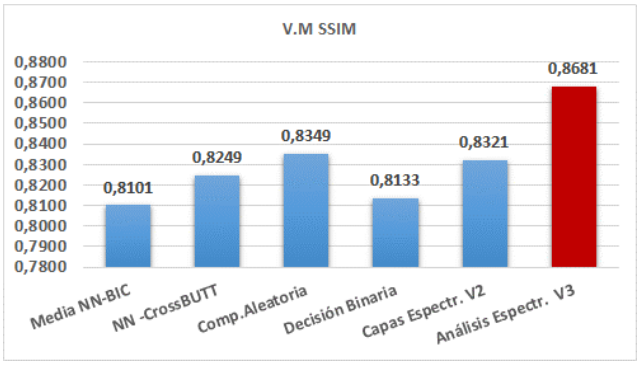


Tabla 5-5: V. M. Índice de Similitud Estructural (P)

Para la última medición, la relación señal-ruido pico (tabla 5-6), arroja sus mejores resultados sobre la técnica de interpolación según análisis espectral en primer lugar y en segundo, sobre el interpolador por capas espectrales.

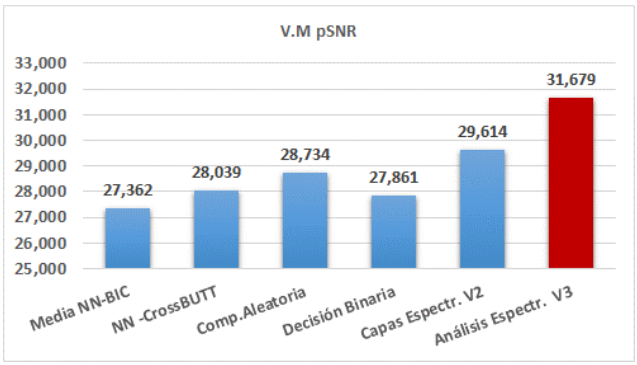


Tabla 5-6: V. M. Relación Señal-Ruido Pico (P)

De entre las técnicas propuestas, las principales (y a su vez las más complejas) son sin lugar a duda las que mejores resultados ofrecen. Esto se debe a que su diseño ha sido mucho más elaborado, estudiado y desarrollado de forma metódica mientras que las otras técnicas han surgido de ideas simples que se dieron durante el transcurso de este trabajo.

Sin embargo, para todas estas técnicas propuestas (principales o no), nos encontramos resultados bastante positivos con respecto a las técnicas existentes en las que se basan. También podemos afirmar que incluso las técnicas principales de este trabajo (tanto interpolador por capas espectrales como interpolador según análisis de espectro[6]) dan lugar a resultados muy a la altura de las mejores técnicas ya existentes.

[6] Recordamos que las ideas de diseño de estas técnicas se exponen los capítulos 3.2.1 y 3.2.2 y su desarrollo en los capítulos 4.1 y 4.2.

5.2.2 Análisis de resultados: medidas sin referencia

Las medidas de calidad de imagen sin referencia establecen un valor o score en función de la información que contiene la imagen. Por ello en estas medidas, es de gran importancia el tipo de imágenes sobre las que se emplea el evaluador de calidad (siendo muy relevantes las imágenes con resolución, buena nitidez, correcta iluminación, contraste…).
Para todos estos medidores, un valor pequeño indica mayor calidad para la imagen.

Técnicas existentes

Al aplicar sobre nuestra base de 320 imágenes cada operador existente para después emplear las medidas de calidad sin referencia y calcular sus valores medios obtenemos los siguientes resultados (tabla 5-7):

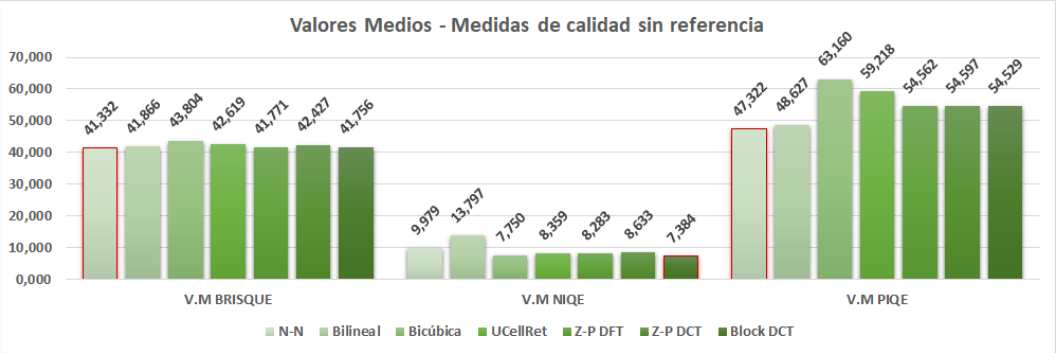


Tabla 5-7: V. M. Medidas de calidad sin referencia (E)

Para la medida BRISQUE y NIQUE, los valores medios apenas muestran unas pocas unidades o decenas de diferencia, siendo los mejores resultados los ofrecidos por Nearest-Neighbor y Block-Based DCT respectivamente. Para la medida PIQE nuevamente nos encontramos con Nearest-Neighbor como la técnica que ofrece los resultados mejores.

Técnicas propuestas

Ahora, si calculamos estos mismos resultados para los sistemas propuestos obtenemos lo siguiente (tabla 5-8):

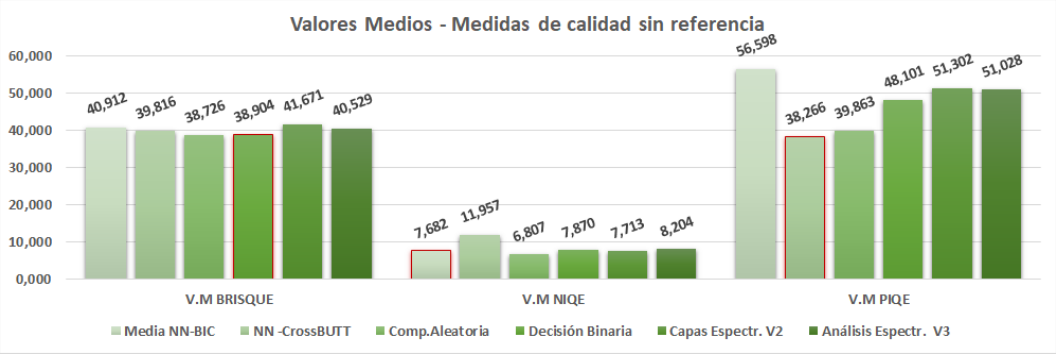


Tabla 5-8: V. M. Medidas de calidad sin referencia (P)

Si comparamos la tabla de resultados 5-8 con la 5-7 anteriormente comentada, podemos observar cómo hemos conseguido proponer unas técnicas que ofrecen, en prácticamente todos los casos, unos resultados mucho mejores que los ofrecidos por las técnicas existentes en las que se basan. Excluyendo los resultados ofrecidos por el interpolador de composición aleatoria, obtenemos como mejores valores para BRISQUE, NIQE y PIQE, los ofrecidos por los interpoladores de decisión binaria, de media (o combinación) Nearest-Neighbor y Bicúbica y de Nearest-Neighbor con corrección Cross-Butterworth.

6 Conclusiones y trabajo futuro

6.1 Conclusiones

Vivimos en una realidad actualmente sujeta al intercambio de información a lo largo de todo el mundo. La era digital se abre paso entre millones de usuarios que comparten mensajes, audios, imágenes y videos… en definitiva, señales de todo tipo, y todo gracias a esa capacidad de 'estar conectados' en todo momento.
La facilidad de manejo de información entre personas abre un nuevo frente: su procesado.

Ya lo decíamos al introducir este trabajo: a día de hoy ¿quién no ha aplicado un zoom o un giro a una imagen desde su terminal móvil? Estos gestos tan sencillos son posibles gracias al procesamiento de esa imagen que indirectamente queremos 'manipular', y más concretamente, son las técnicas de interpolación las que nos permiten disfrutar del resultado, pues son las que estiman los valores que tendrán los pixeles de nuestra imagen ampliada o rotada.

Lo cierto es que la interpolación de datos (y no únicamente de imágenes digitales), presenta una utilidad sin mesura. La clave está en lo que implícitamente pretende: la estimación de nuevos datos.

Durante todo este recorrido hemos discutido sobre diferentes técnicas y algoritmos de interpolación de imágenes digitales. El estudio directo de las ya existentes nos ha permitido conocer su funcionamiento, y entender cómo se comporta una imagen a la salida de un interpolador, tanto en espacio como en frecuencia.
La implementación en MATLAB de todos ellos (a pesar de existir varios modelos ya programados y de fácil acceso en la red) ha sido un pilar fundamental en la etapa de aprendizaje y vital para entender los puntos fuertes y débiles de cada técnica.

En definitiva, con este trabajo hemos comprendido su funcionamiento tanto teórico como práctico y hemos analizado cómo se comportaban con diferentes conjuntos de imágenes de prueba. Todo ello nos ha permitido desarrollar (o mejorar) algunas técnicas simples y elaborar unas nuevas técnicas más complejas con buenas bases analíticas justificadas y unos resultados en calidad favorables.

Sin embargo, debemos considerar que todavía estamos muy lejos de obtener resultados perfectos. La estimación de nueva información sobre un conjunto de datos que se presentan de forma incierta (como es el caso de una imagen), no nos asegura que esta sea objetivamente correcta con la información real en ninguno de los casos.
La interpolación de señales multidimensionales es un tema muy interesante que, investigado en profundidad, podría adquirir grandísimas extensiones; más adelante comentaremos algunas barreras que presenta a nivel de resultados (en imágenes a color), su interpretación (¿diezmar-interpolar es realmente un buen procedimiento?), o herramientas con mucho potencial de cara a obtener muy buenos resultados (machine learning).

Por último, hemos comprendido las cargas computacionales de cada sistema. Hemos descubierto cómo estas cargas no suponen una barrera dadas las capacidades tecnológicas de los dispositivos actuales y, por lo tanto, podríamos actualizar las típicamente implementadas.

6.2 Trabajo futuro

La realización de este trabajo de fin de grado ha tenido en cuenta el uso de imágenes en escala de grises por la simplicidad de procesado que tiene este formato de color.
La aplicación de interpoladores sobre imágenes digitales a color no es por el contrario mucho más compleja, pues típicamente está extendido el llevar a cabo la misma técnica de procesado, pero esta vez repitiendo el algoritmo en cada capa de color.
Por lo tanto, disponiendo de una imagen en formato por ejemplo RGB, para interpolarla bastaría con aplicar la técnica de forma independientemente la capa R, la G y la B.
Sin embargo, dada la no linealidad entre las capas de un mismo espacio de color, es frecuente la aparición de ciertos artefactos o saltos de color en la imagen resultante.
La solución a este complejo problema suele ser el procesado de pixeles de forma vectorial, lo que supondría el diseño de nuevas técnicas adaptadas a atajar este problema, algo que podría ser un interesante trabajo de cara al futuro.

Por otra parte, para la extracción de las diferentes medidas de calidad hemos empleado una base de datos de imágenes que previamente diezmábamos para luego devolverla a sus dimensiones iniciales y así comparar este resultado con su original.
Este efecto de diezmar la imagen nos va a producir en la mayor parte de los casos una gran eliminación de altas frecuencias, por lo que la interpolación no estará trabajando sobre un conjunto de datos íntegramente ‘realista’.
La solución a esto es muy interesante, por ello se propone como trabajo futuro lo siguiente: el uso de una base de datos de imágenes con y sin zoom analógico.
Las medidas de calidad serían obtenidas a partir de la comparación directa entre la imagen con zoom analógico y la imagen sin zoom interpolada.
Dado que todas las lentes añaden una ligera distorsión sobre la imagen al interpolarla, podríamos simular este zoom analógico acercando de forma manual la cámara manteniendo el mismo centro focal (mediante un sistema de guía).

Por último, una herramienta muy potente en el tratamiento de datos con gran precisión son las redes neuronales artificiales, siendo éstas muy interesantes para su uso en la interpolación de información de todo tipo.

Referencias

[1] K. Pavithradevi, M. Pavithra, S. Mangayarkani, “Digital Image Processing and its applications”, IJRASET - International Journal for Research in Applied Science & Engineering Technology, Volume 5 Issue VI, June 2017.

[2] Rose Eveleth, “How Many Photographs of You Are Out There in the World?”, The Atlantic Magazine (online journal article), November 2015.

[3] Dianyuan Han, “Comparison of Commonly Used Image Interpolation Methods”, Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering, (ICCSEE 2013).

[4] Russell A. Kirsch, “SEAC and the Start of Image Processing at the National Bureau of Standards”, IEEE Annals of the History of Computing, Vol. 20, No. 2, 1998.

[5] James D. Murray and William vanRyper, “Encyclopedia of graphics file formats”, O'Reilly & Associates, April 1996.

[6] John Woods, “Multidimensional Signal, Image, and Video Processing and Coding”, Academic Press, June 2011.

[7] Anil K. Jain, “Fundamentals of Digital Image Processing”, Prentice Hall, 1989.

[8] William Fleetwood Sheppard, "Interpolation", In Chisholm, Hugh (ed.). Encyclopædia Britannica. 14 (11th ed.). Cambridge University Press. pp. 706–710.

[9] Alan V. Oppenheim, Ronald W. Schafer, “Discrete-Time Signal Processing” (2nd ed.), Prentice Hall. p. 168.

[10] Mrs. Amruta Savagave, Prof. A. P. Patil, “Study of Image Interpolation”, IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 10, December 2014.

[11] Philippe Thévenaz, Thierry Blu and Michael Unser, “Image Interpolation and Resampling”, Extracted from Swiss Federal Institute of Technology—Lausanne, pp. 7-10.

[12] Faisal A Khan, S. P Bhosale, “Image Interpolation Techniques in Digital Image Processing: An Overview”, IJSR - International Journal of Science and Research, Volume 4 Issue 7, July 2015.

[13] Alasdair McAndrew, “A Computational Introduction to Digital Image Processing” (2nd ed), Chapman and Hall/CRC, September 2015, pp. 128-130.

[14] Prachi R Rajarapollu, Vijay R Mankar, “Bicubic Interpolation Algorithm Implementation for Image Appearance Enhancement”, IJCST - International Journal of Computer Science and Technology, Vol. 8, Issue 2, April - June 2017.

[15] Paul Breeuwsma, “Cubic interpolation”, Paulinternet, Recuperado de https://www.paulinternet.nl/?page=bicubic

[16] Jesús Bescós Cano, “Interpolación y diezmado”, Material extraído de la asignatura “Temas Avanzados en Proceso de Señales – TAPS” (2009), Recuperado de http://arantxa.ii.uam.es/~jms/taps/teoria/Interpolacion.pdf

[17] Chi-Wah Kok, Wing-Shan Tam, “Digital Image Interpolation in MATLAB”, Wiley IEEE PRESS, December 2018, pp. 125-132.

[18] Chi-Wah Kok, Wing-Shan Tam, “Digital Image Interpolation in MATLAB”, Wiley IEEE PRESS, December 2018, pp. 132-138.

[19] Chi-Wah Kok, Wing-Shan Tam, “Digital Image Interpolation in MATLAB”, Wiley IEEE PRESS, December 2018, pp. 138-144.

[20] Alasdair McAndrew, “Introduction to digital image processing with Matlab”, Brooks/Cole, May 2004, pp. 148-158.

[21] Richard Szeliski, “Computer Vision: Algorithms and Applications” (1st ed.), Springer, October 2010, pp. 101-102.

[22] John G. Proakis, Dimitris Manolakis, “Digital Signal Processing: Principles, Algorithms, and Applications” (3rd ed.), Prentice Hall, October 1995, pp. 896-902.

[23] Rafael C. Gonzalez, Richard E. Woods, “Digital Image Processing” (3rd ed.), Prentice Hall, 2008.

[24] K.Silpa, Dr.S.Aruna Mastani, “Comparison of image quality metrics”, IJERT - International Journal of Engineering Research & Technology, Vol. 1, Issue 4, June – 2012.

[25] MathWorks Support Team, “Image Quality”, MathWorks, Recuperado de https://es.mathworks.com/help/images/image-quality-metrics.html?lang=en

[26] MathWorks Support Team, “Train and Use No-Reference Quality Assessment Model”, MathWorks, Recuperado de https://es.mathworks.com/help/images/train-and-use-a-no-reference-quality-assessment-model.html?lang=en

Glosario

BBDD	Base de datos
BRISQUE	Blind/Referenceless Image Spatial Quality Evaluator
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
Fc	Frecuencia de corte
L-M-H	Low-Mid-High
MSE	Mean-squared error
NIQE	Naturalness Image Quality Evaluator
pSNR	Peak Signal-to-Noise Ratio
PIQE	Perception based Image Quality Evaluator
RRSS	Redes Sociales
SSIM	Structural Similarity
TFG	Trabajo de Fin de Grado
UCELLRET	Unitary Cell Reticle
WSD	Windowed Spectral Density
ZP-DCT	Zero-Padding Discrete Cosine Transform
ZP-DFT	Zero-Padding Discrete Fourier Transform

Anexos

Anexo 1: Implementación MATLAB de técnicas de interpolación y métricas de calidad.
En este anexo facilitamos un enlace de descarga al fichero comprimido .zip que incluye todas y cada una de las interpolaciones tratadas en el trabajo y que han sido implementadas a mano[7].

Los ficheros PRUEBAS_xxx son los ejecutables principales, mientras que el resto de ficheros TSM_xxx son funciones que implementan las diferentes técnicas de interpolación.
El fichero comprimido consta de lo siguiente:

- Carpeta "Capítulo 1 - Técnicas Espaciales".
- Carpeta "Capítulo 2 - Técnicas Frecuenciales".
- Carpeta "Capítulo 3 - Técnicas Propuestas".
- Carpeta "Capítulo 4 - Pruebas con Imágenes".
- Carpeta "Otras Funciones".

Este material ha sido confeccionado para la realización escrita de este trabajo, todas las ilustraciones y tablas presentadas han sido generadas mediante estos archivos.

Puede descargarlo mediante el siguiente enlace de Google Drive[8]:
https://drive.google.com/file/d/1Q8BHgg2zqaN0-AWiZbhLqzg6snQfPElK/view?usp=sharing

También puede hacer uso del siguiente código QR:

Notas:
Es de especial importancia mantener los nombres de todos los archivos y carpetas para el correcto funcionamiento de los ficheros ejecutables.

La carpeta "Capítulo 4 - Pruebas con Imágenes" contiene la BBDD de imágenes empleadas para el cálculo de las medidas de calidad, puede sustituirse por otra siempre y cuando se conserve el formato de nombre de imágenes y de la carpeta contenedora.

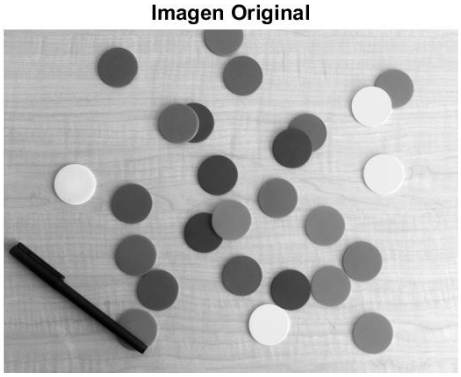
En la ejecución del fichero "PRUEBAS_MedidasCalidad.m" es muy importante generar (en su primera ejecución) las imágenes diezmadas para poder generar las versiones interpoladas. Las carpetas con las imágenes resultado se generarán de forma automática.

[7] Códigos implementados por Enrique Luque Lanza, basados en algunos modelos establecidos de prácticas de la asignatura de Tratamiento de Señales Multimedia.

[8] Si el enlace no funciona puede mandar una solicitud por e-mail a: enrique.luque@estudiante.uam.es
enrique.luque.lanza@gmail.com

Anexo 2: Efecto mirroring frecuencial introducido en la interpolación por retículos de celda unidad.

Imagen Original



DFT Imagen Original

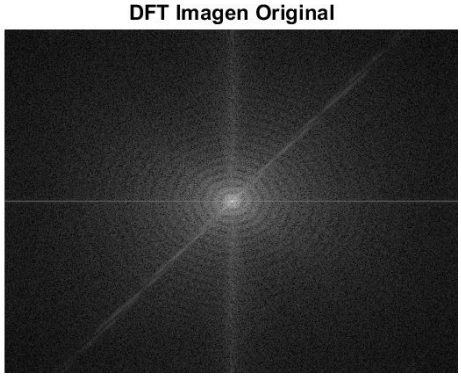
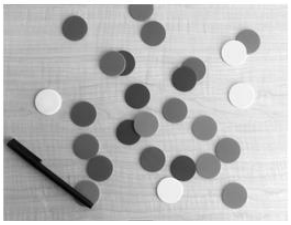


Imagen Diezmada



DFT Imagen Diezmada

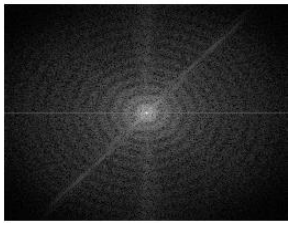
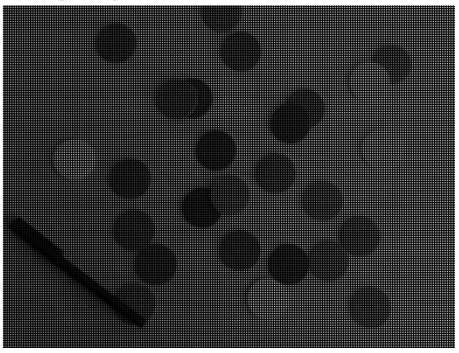


Imagen Ajedrez antes de aplicar filtro Butt



DFT Imagen Ajedrez ANTES de aplicar filtro Butt

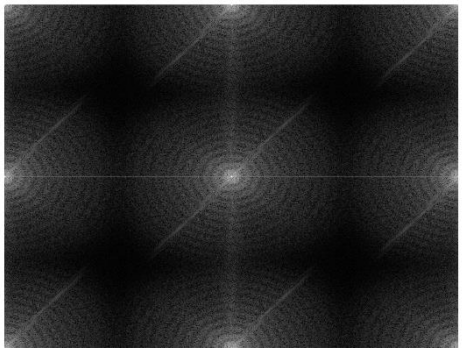
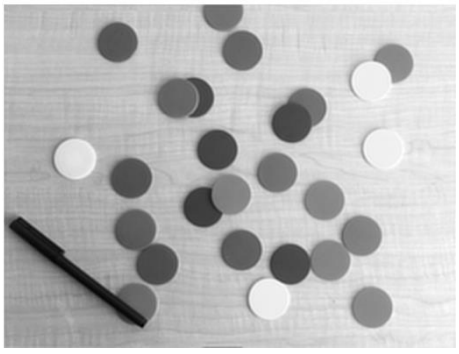
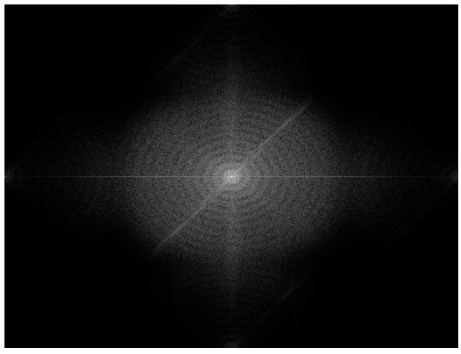


Imagen Interpolada Final



DFT Imagen interpolada DESPUES de aplicar filtro Butt



Anexo 3: Efecto Padding frecuencial como interpolador (Zero-Padding DFT).

Imagen Original

DFT Imagen Original

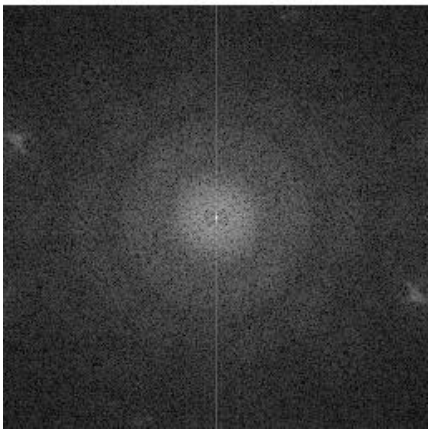


Imagen Diezmada

DFT Imagen Diezmada

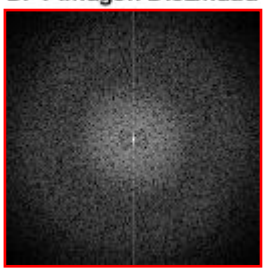
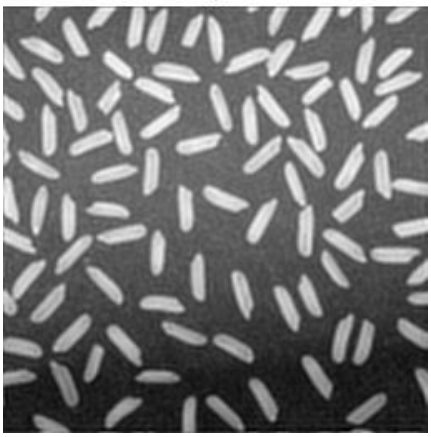
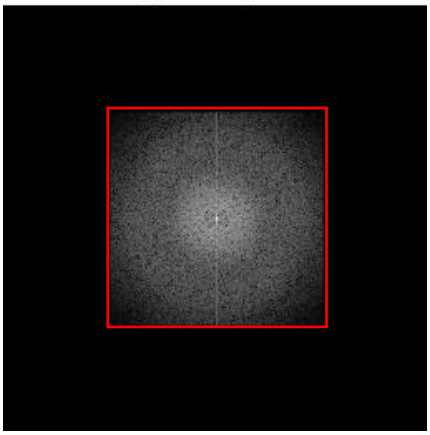


Imagen Intepolada Final

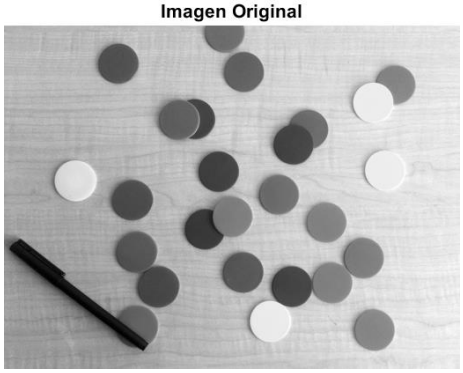


DFT Imagen Interpolada Final



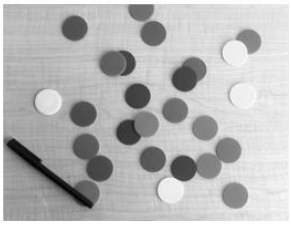
Anexo 4: Efecto Padding frecuencial como interpolador (Zero-Padding DCT).

Imagen Original



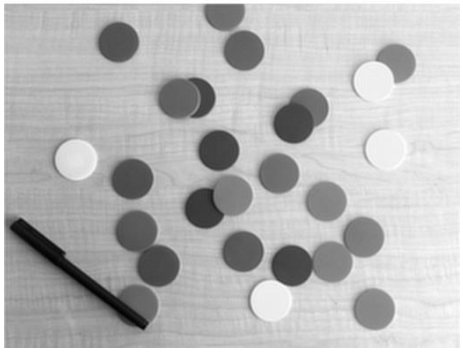
DFT Imagen Original

Imagen Diezmada



DFT Imagen Diezmada

Imagen Intepolada Final



DFT Imagen Interpolada Final

Anexo 5: Resultados de calidad sobre imágenes de prueba (capítulo 3.1):
A lo largo del capítulo 3.1 introducimos el diseño de diferentes técnicas simples que son una primera aproximación al objetivo de este trabajo.

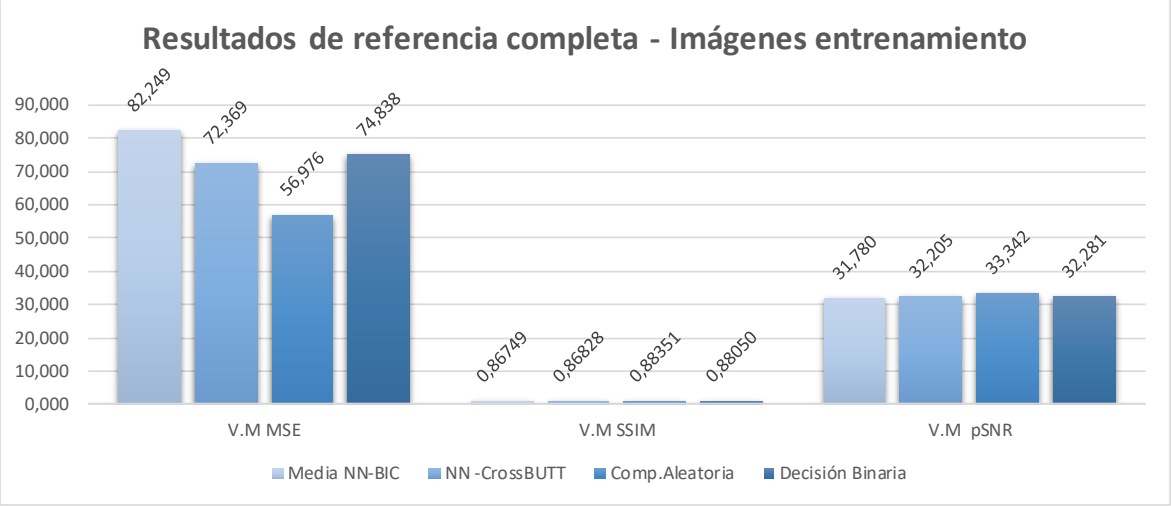
Una vez finalizado el proceso de desarrollo (explicado en ese mismo punto), sometimos a pruebas de calidad los resultados de tratar tres imágenes de prueba con cada sistema. Las imágenes de entrenamiento son: `rice.png`, `coloredChips.png` y `threads.png` (todas ellas disponibles en el toolbox de procesamiento de imágenes en MATLAB).

Los resultados obtenidos según medidas de referencia completa son los siguientes:

	Error cuadrático medio (MSE)			
ImageName	**Media NN-BIC**	**NN -CrossBUTT**	**Comp.Aleatoria**	**Decisión Binaria**
rice.png	197,300	173,811	137,548	181,738
coloredChips.png	39,001	32,772	25,629	33,409
threads.png	10,447	10,523	7,751	9,368
V.M MSE	82,249	72,369	56,976	74,838

	Índice de Similitud Estructural (SSIM)			
ImageName	**Media NN-BIC**	**NN-CrossBUTT**	**Comp.Aleatoria**	**Decisión Binaria**
rice.png	0,72738	0,73491	0,76503	0,74497
coloredChips.png	0,88935	0,89144	0,90372	0,91471
threads.png	0,98573	0,97848	0,98179	0,98181
V.M SSIM	0,86749	0,86828	0,88351	0,88050

	Relación Señal-Ruido pico (pSNR)			
ImageName	**Media NN-BIC**	**NN-CrossBUTT**	**Comp.Aleatoria**	**Decisión Binaria**
rice.png	25,180	25,730	26,746	25,536
coloredChips.png	32,220	32,976	34,043	32,892
threads.png	37,941	37,909	39,237	38,414
V.M pSNR	31,780	32,205	33,342	32,281

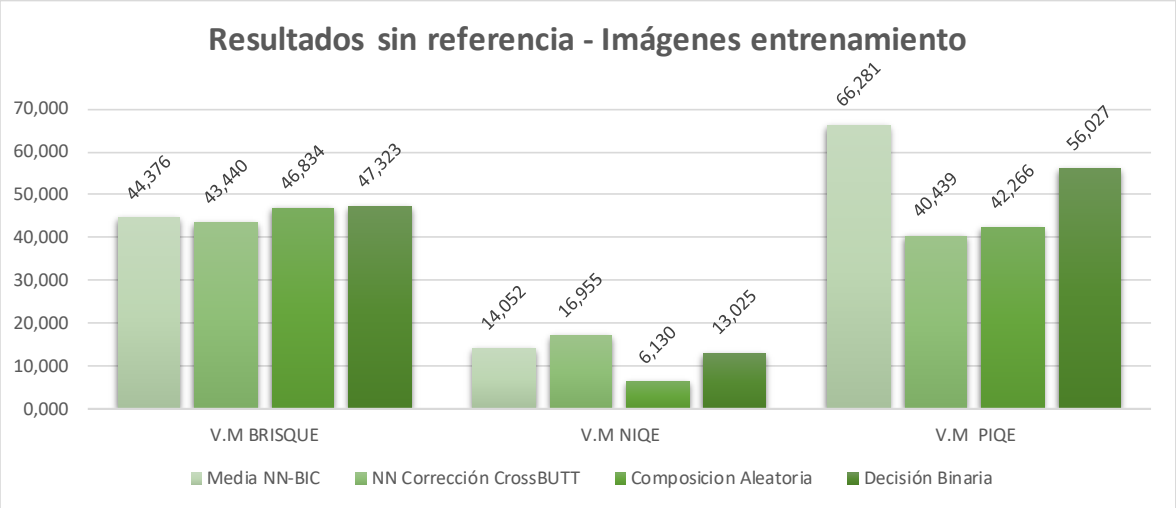


En este caso obviamos los resultados ofrecidos por el interpolador de composición aleatoria ya que los resultados son fortuitos y se presentan de forma aleatoria en cada ejecución. La técnica con mejor error cuadrático medio es Nearest-Neighbor con corrección Cross-Butterworth, la que presenta mejor índice de similitud estructural y relación señal-ruido pico es el interpolador por decisión binaria.
Los resultados obtenidos según medidas sin referencia son los siguientes:

	Evaluador de calidad espacial de imagen ciega/sin referencia (BRISQUE)			
ImageName	**Media NN-BIC**	**NN Corrección CrossBUTT**	**Composicion Aleatoria**	**Decisión Binaria**
rice.png	43,458	43,458	43,458	43,458
coloredChips.png	41,554	42,034	39,270	41,407
threads.png	48,114	44,827	57,773	57,102
V.M BRISQUE	44,376	43,440	46,834	47,323

	Evaluador de calidad de imagen basado en naturalidad (NIQE)			
ImageName	**Media NN-BIC**	**NN Corrección CrossBUTT**	**Composicion Aleatoria**	**Decisión Binaria**
rice.png	30,653	28,995	8,622	26,977
coloredChips.png	5,765	15,572	5,029	6,628
threads.png	5,739	6,299	4,741	5,472
V.M NIQE	14,052	16,955	6,130	13,025

	Evaluador de calidad de imagen basado en percepción (PIQE)			
ImageName	**Media NN-BIC**	**NN Corrección CrossBUTT**	**Composicion Aleatoria**	**Decisión Binaria**
rice.png	53,324	16,515	27,295	39,086
coloredChips.png	55,988	30,746	33,235	45,952
threads.png	89,533	74,055	66,268	83,042
V.M PIQE	66,281	40,439	42,266	56,027



Nuevamente, en la interpretación de los resultados descartamos los resultados ofrecidos por el interpolador de composición aleatoria.

Así, la técnica que arroja mejores resultados en calidad espacial de imagen ciega (BRISQUE) y de percepción (PIQE) es el interpolador Nearest-Neighbor con corrección Cross-Butterworth.

El interpolador por decisión binaria adquiere los mejores resultados de calidad de imagen basados en naturalidad (NIQE).

Anexo 6: Resultados de calidad sobre imágenes de prueba (capítulo 4.1):
A lo largo del capítulo 4.1 hablamos sobre el desarrollo e implementación de las dos técnicas principales de este trabajo.

En este caso, hemos desarrollado hasta dos versiones para la interpolación por capas espectrales, y hasta tres versiones para la interpolación según análisis de espectro.

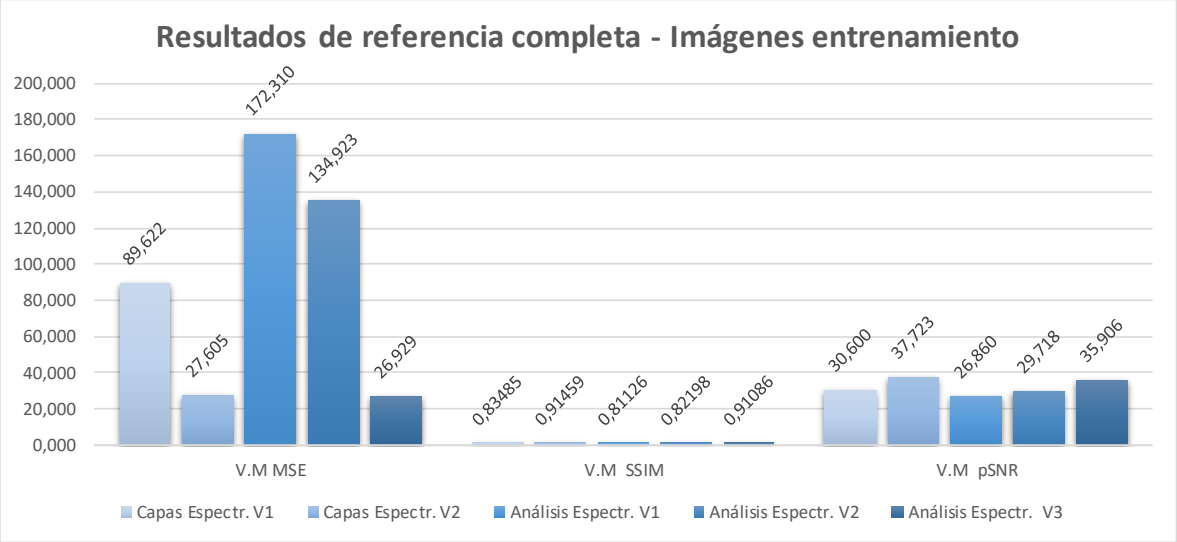
Hemos sometido a pruebas de calidad los resultados de tratar tres imágenes de prueba con cada una de estas versiones. Las imágenes de entrenamiento son: `rice.png`, `coloredChips.png` y `threads.png` (todas ellas disponibles en el toolbox de procesamiento de imágenes en MATLAB).

Los resultados obtenidos según medidas de referencia completa son los siguientes:

	Error Cuadrático Medio (MSE)				
ImageName	**Capas Espectr. V1**	**Capas Espectr. V2**	**Análisis Espectr. V1**	**Análisis Espectr. V2**	**Análisis Espectr. V3**
rice.png	189,869	66,693	346,269	325,834	63,151
coloredChips.png	64,063	14,775	103,562	62,546	10,851
threads.png	14,935	1,345	67,099	16,389	6,785
V.M MSE	89,622	27,605	172,310	134,923	26,929

	Índice de Similitud Estructural (SSIM)				
ImageName	**Capas Espectr. V1**	**Capas Espectr. V2**	**Análisis Espectr. V1**	**Análisis Espectr. V2**	**Análisis Espectr. V3**
rice.png	0,71636	0,82326	0,66464	0,65285	0,82419
coloredChips.png	0,81867	0,92627	0,82696	0,84717	0,92414
threads.png	0,96953	0,99423	0,94218	0,96593	0,98426
V.M SSIM	0,83485	0,91459	0,81126	0,82198	0,91086

	Relación Señal-Ruido pico (pSNR)				
ImageName	**Capas Espectr. V1**	**Capas Espectr. V2**	**Análisis Espectr. V1**	**Análisis Espectr. V2**	**Análisis Espectr. V3**
rice.png	25,346	29,890	22,737	23,001	30,127
coloredChips.png	30,065	36,435	27,979	30,169	37,776
threads.png	36,389	46,843	29,864	35,985	39,815
V.M pSNR	30,600	37,723	26,860	29,718	35,906



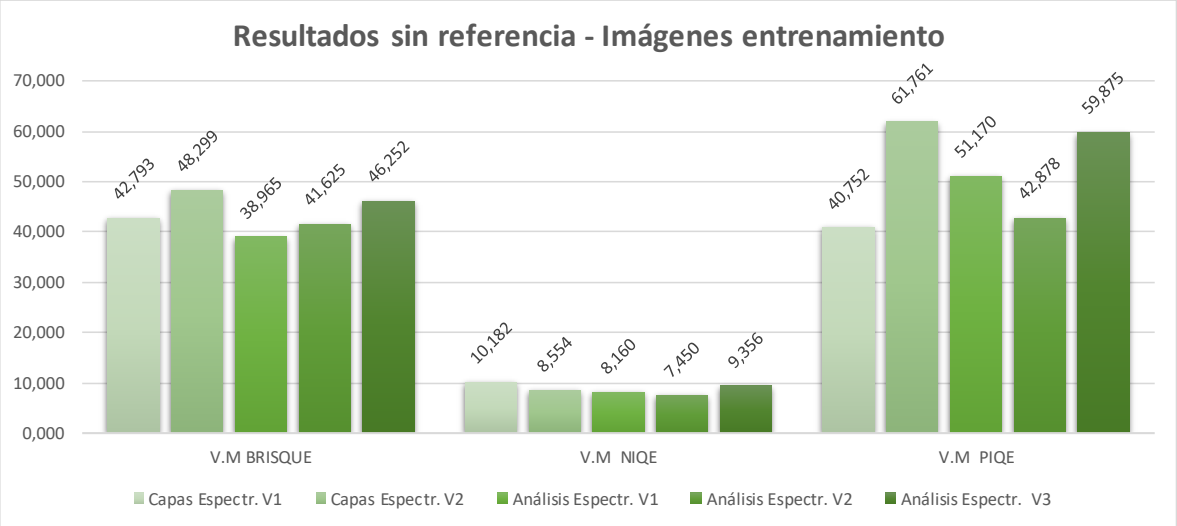
Para cada versión, todos los resultados han evolucionado de forma muy favorable, consiguiendo disminuir el MSE y aumentando el SSIM y pSNR. Ambas técnicas tienen resultados muy similares y buenos en su versión final para las imágenes de entrenamiento.

Los resultados obtenidos según medidas sin referencia son los siguientes:

	Evaluador de calidad espacial de imagen ciega/sin referencia (BRISQUE)				
ImageName	**Capas Espectr. V1**	**Capas Espectr. V2**	**Análisis Espectr. V1**	**Análisis Espectr. V2**	**Análisis Espectr. V3**
rice.png	43,458	43,458	43,458	43,458	43,458
coloredChips.png	41,333	42,286	30,398	34,231	43,264
threads.png	43,588	59,154	43,039	47,185	52,033
V.M BRISQUE	42,793	48,299	38,965	41,625	46,252

	Evaluador de calidad de imagen basado en naturalidad (NIQE)				
ImageName	**Capas Espectr. V1**	**Capas Espectr. V2**	**Análisis Espectr. V1**	**Análisis Espectr. V2**	**Análisis Espectr. V3**
rice.png	16,868	14,651	16,150	13,205	16,308
coloredChips.png	6,456	6,000	3,475	4,195	6,426
threads.png	7,222	5,010	4,856	4,950	5,334
V.M NIQE	10,182	8,554	8,160	7,450	9,356

	Evaluador de calidad de imagen basado en percepción (PIQE)				
ImageName	**Capas Espectr. V1**	**Capas Espectr. V2**	**Análisis Espectr. V1**	**Análisis Espectr. V2**	**Análisis Espectr. V3**
rice.png	30,596	42,316	39,112	28,807	44,716
coloredChips.png	30,105	54,397	36,093	29,583	50,997
threads.png	61,554	88,572	78,304	70,245	83,911
V.M PIQE	40,752	61,761	51,170	42,878	59,875



Nuevamente, nos encontramos en la misma situación que la expuesta en las técnicas de referencia completa. Los resultados en este caso empeoran en cada versión, pero esto tiene una sencilla explicación: las versiones iniciales introducían una serie de artefactos o distorsiones sobre la imagen que corregíamos mediante filtrados de mediana, refuerzo de altas frecuencias y aumento de la nitidez. La aplicación de estas correcciones se traduce, en definitiva, en mejores resultados para los medidores sin referencia.
En la versión final de las técnicas de interpolación por capas espectrales y por análisis espectral, estos artefactos no se introducían en el resultado, por lo que no es necesario aplicar las correcciones, y por lo tanto obtenemos unos resultados 'peores'.

Anexo 7: Selección de condicionales según WSD en el interpolador según análisis de espectro (capítulo 4.1.2):

La elección de los porcentajes de densidad espectral por bandas se ha realizado de forma empírica sobre imágenes de entrenamiento. Para ello, hemos extraído porciones de una imagen con contenidos espaciales diferentes (contenidos planos, con más o menos bordes, detalles…). Sobre cada porción hemos aplicado el porcentaje de densidad espectral por banda (WSD_xxx), conociendo así cómo se distribuye, estableciendo una relación numérico-visual. Así, por ejemplo, para los siguientes pedazos de imagen:

Porción entrenamiento 1 (Región Plana sin contornos)

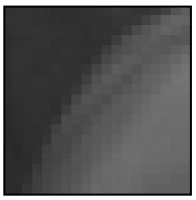
WSD_LP1 = 34.26254 (%)
WSD_BP1 = 15.18423 (%)
WSD_BP2 = 12.09134 (%)
WSD_BP3 = 11.68720 (%)
WSD_BP4 = 10.03663 (%)
WSD_BP5 = 8.733907 (%)
WSD_HP1 = 8.004129 (%)

WSD_LowFreqs → 34.2625 (%)
WSD_MidFreqs → 48.9994 (%)
WSD_HighFreqs → 16.73803 (%)

Porción entrenamiento 2 (Zona con borde algo marcado)

WSD_LP1 = 26.35306 (%)
WSD_BP1 = 18.90477 (%)
WSD_BP2 = 14.33921 (%)
WSD_BP3 = 11.88475 (%)
WSD_BP4 = 10.62553 (%)
WSD_BP5 = 9.499067 (%)
WSD_HP1 = 8.393590 (%)

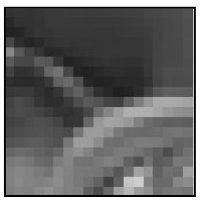
WSD_LowFreqs → 26.353062 (%)
WSD_MidFreqs → 55.754282 (%)
WSD_HighFreqs → 17.892656 (%)



Porción entrenamiento 3 (Zona con mucha variación)

WSD_LP1 = 22.61094 (%)
WSD_BP1 = 16.42485 (%)
WSD_BP2 = 14.60939 (%)
WSD_BP3 = 13.36669 (%)
WSD_BP4 = 12.03338 (%)
WSD_BP5 = 10.98340 (%)
WSD_HP1 = 9.971327 (%)

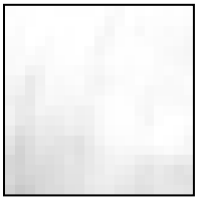
WSD_LowFreqs → 22.610942 (%)
WSD_MidFreqs → 56.434323 (%)
WSD_HighFreqs → 20.954735 (%)



Porción entrenamiento 4 (Región Plana con poco contorno)

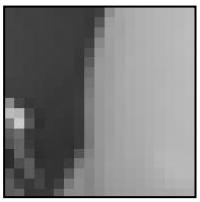
WSD_LP1 = 30.63499 (%)
WSD_BP1 = 18.49695 (%)
WSD_BP2 = 12.97369 (%)
WSD_BP3 = 11.42483 (%)
WSD_BP4 = 9.659676 (%)
WSD_BP5 = 8.918606 (%)
WSD HP1 = 7.891234 (%)

WSD_LowFreqs → 30.634995 (%)
WSD_MidFreqs → 52.555164 (%)
WSD_HighFreqs → 16.809841 (%)



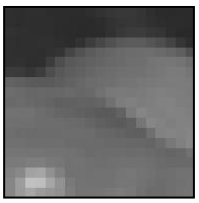
Porción entrenamiento 5 (Zona con borde muy marcado)

WSD_LP1 = 23.79902 (%)
WSD_BP1 = 15.97401 (%) WSD_LowFreqs → 23.799026 (%)
WSD_BP2 = 14.61396 (%)
WSD_BP3 = 12.72947 (%) WSD_MidFreqs → 55.26011 (%)
WSD_BP4 = 11.94265 (%)
WSD_BP5 = 11.07598 (%) WSD_HighFreqs → 20.940856 (%)
WSD_HP1 = 9.864868 (%)



Porción entrenamiento 6 (Zona con borde marcado)

WSD_LP1 = 25.364074 (%)
WSD_BP1 = 18.423257 (%) WSD_LowFreqs → 25.364074 (%)
WSD_BP2 = 14.475191 (%)
WSD_BP3 = 12.505445 (%) WSD_MidFreqs → 56.525587 (%)
WSD_BP4 = 11.121694 (%)
WSD_BP5 = 9.568042 (%) WSD_HighFreqs → 18.110339 (%)
WSD_HP1 = 8.542298 (%)



En este caso todas las porciones tienen un tamaño de 19x19 pixeles, a las cuales hemos dividido en bandas según el banco de filtros definido en el capítulo 4.1.2 (figura 4-5), obteniendo esos resultados. Hemos definido que:

- Como el detalle grueso se concentra en las frecuencias más bajas y con un gran nivel de energía, este detalle grueso toma el valor: WSD_LowFreqs = WSD_LP1.
- El detalle medio o la información de contornos y bordes, viene definida por la región media de frecuencias, por lo que esta información tomará el valor: WSD_MidFreqs = WSD_BP1 + WSD_BP2 + WSD_BP3 + WSD_BP4.
- El detalle fino viene denotado por la región frecuencial más alta, así que vendrá denotado según: WSD_HighFreqs = WSD_BP5 + WSD_HP1.

Si analizamos los resultados, según este conjunto de pruebas podemos definir:

- Un bloque con poca variación (región plana), presenta WSD_LowFreqs >= 28.5.
- Un bloque de mucha variación (bordes solidos), muestra WSD_HighFreqs >= 19.
- Un bloque con algo de variación (situación intermedia), no cumple ninguna de las anteriores condiciones.

Estas serán las bases de selección de nuestro analizador de frecuencias, mediante las cuales decidiremos qué tipo de interpolador aplicar sobre el pixel de análisis.

Anexo 8: Resultados de calidad sobre subconjunto de imágenes de la base de datos:

Error cuadrático medio - MSE

Esta es una medida de referencia completa que nos permite calcular la cantidad de error existente entre dos conjuntos de información diferentes (donde uno es el original y otro su versionado o predicho). Esta herramienta es muy útil ya que en imágenes es capaz de cuantificar las variaciones existentes entre pixeles. Cuanto más pequeño es este valor, más se asemejan las imágenes a comparar.

Se puede calcular como: $$MSE = \frac{1}{M\,N}\sum_{n=0}^{M-1}\sum_{n=0}^{N-1}(\psi[n,m] - \psi_p[n,m])$$

Índice de similitud estructural - SSIM

El SSIM consiste en una medida de calidad perceptual capaz de cuantificar la degradación de una imagen causada por el procesamiento. Esta medida considera la degradación de la imagen como un cambio percibido en la información estructural, y no solo basándose en los errores absolutos (como MSE). Este valor oscila entre [0, 1] y cuanto más alto es, mejor es la calidad.

Se puede calcular como: $$SSIM\,(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma}$$

Definiendo las funciones de luminancia, contraste y estructura respectivamente como:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \qquad c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \qquad s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}$$

Los valores μ_x, μ_y, σ_x, σ_y, σ_{xy} se corresponden a los promedios, varianzas y covarianza de x e y. Típicamente en el desarrollo de la formula de SSIM se suele emplear $\alpha, \beta, \gamma = 1$

Relación señal-ruido pico – pSNR

Esta es una medida de calidad de referencia completa que muestra la relación existente entre la potencia máxima de una señal y la potencia de ruido que afecta la fidelidad de su representación.

Se puede calcular como: $$pSNR = 10 \cdot log_{10}\left(\frac{MAX_{\psi}^2}{MSE}\right) = 20 \cdot log_{10}\left(\frac{MAX_{\psi}}{\sqrt{MSE}}\right)$$

Evaluador de calidad de imagen espacial ciego - BRISQUE

Esta técnica es una medida no referencial algo más compleja, que utiliza estadísticas de escena de coeficientes de luminancia localmente normalizados, con el fin de cuantificar las posibles pérdidas de "naturalidad" debido a la presencia de distorsiones.[9]

Evaluador de calidad de imagen de naturalidad - NIQE

Similar al anterior, esta medida no referencial ofrece un resultado en función de la utilización de desviaciones medibles de regularidades estadísticas observadas en imágenes puramente naturales.

Evaluador de calidad de imagen de percepción - PIQE

Este algoritmo no referencial nos permite calcular un valor de calidad de forma similar a la medida BRISQUE, pero estimando la distorsión en bloque y midiendo la varianza local de estos bloques distorsionados.[10]

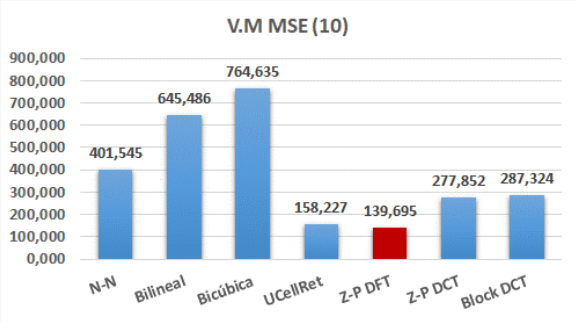
[9] Enlace de interés: https://live.ece.utexas.edu/research/Quality/nrqa.htm

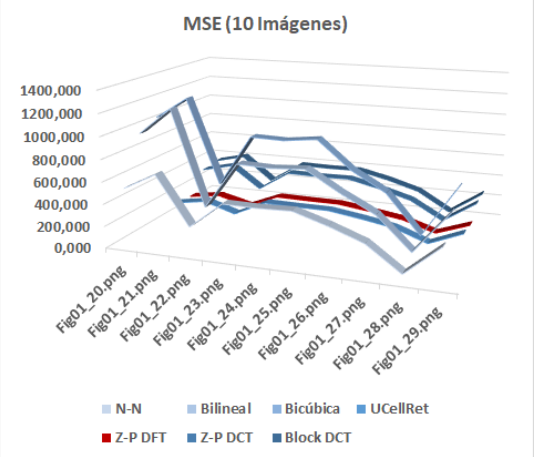
[10] Enlace de interés: https://es.mathworks.com/help/images/ref/piqe.html?lang=en

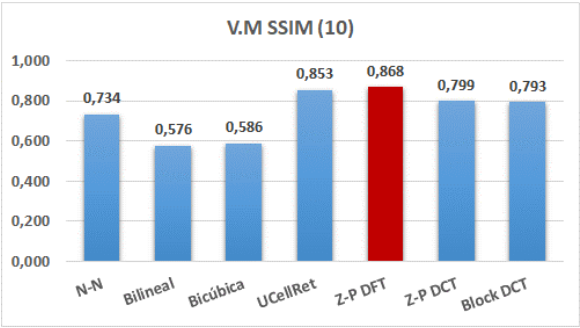
Anexo 9: Resultados de calidad sobre subconjunto de imágenes de la base de datos:

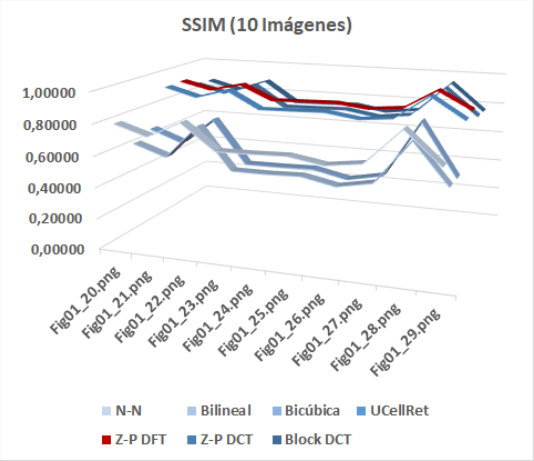
En este anexo se muestran resultados más detallados sobre dos conjuntos de 10 imágenes, las cuales han sido escogidas de forma aleatoria de la BBDD de imágenes empleada en este trabajo.

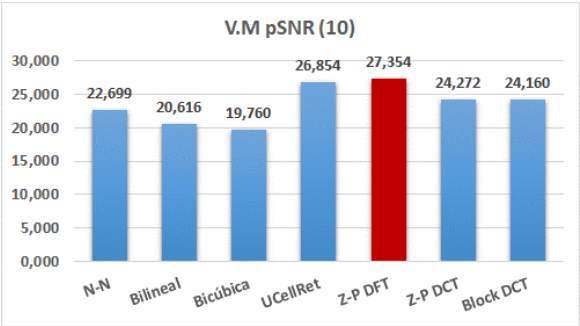
I) Medidas de calidad de referencia completa, técnicas existentes:

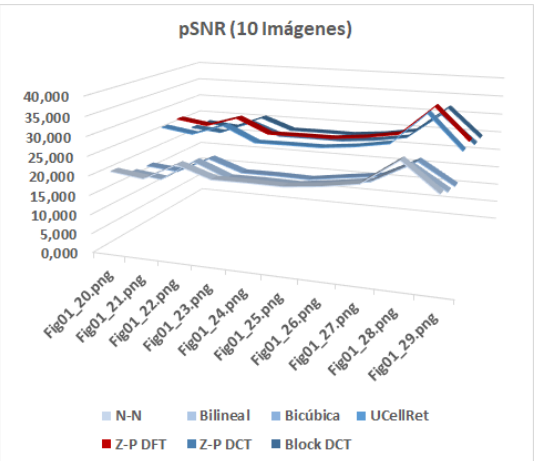




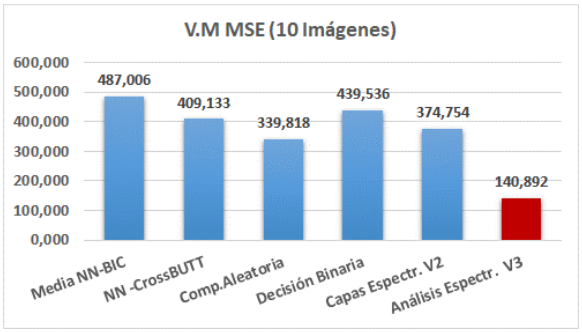


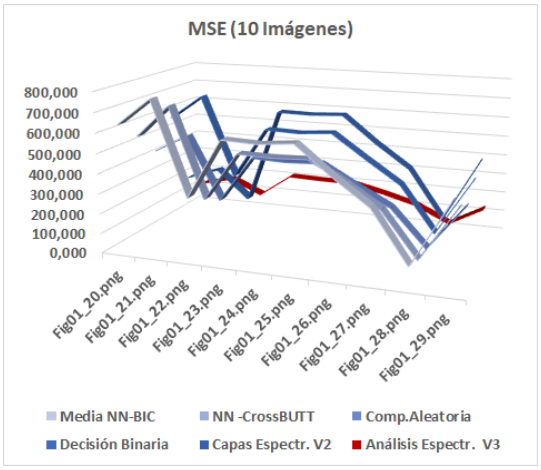


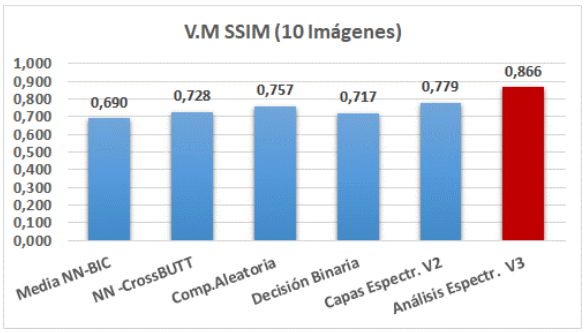


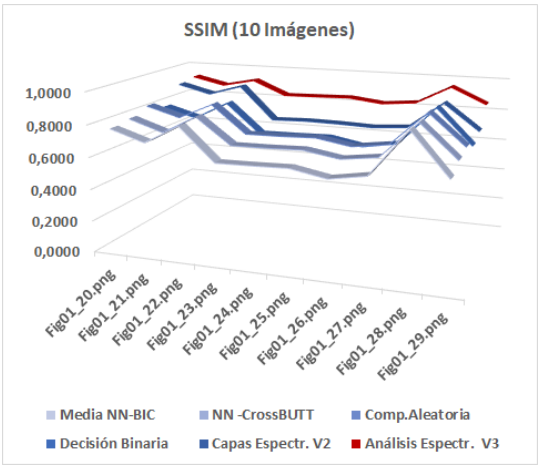


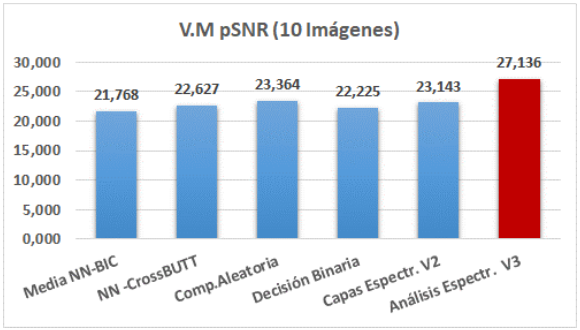
II) Medidas de calidad de referencia completa, técnicas propuestas:

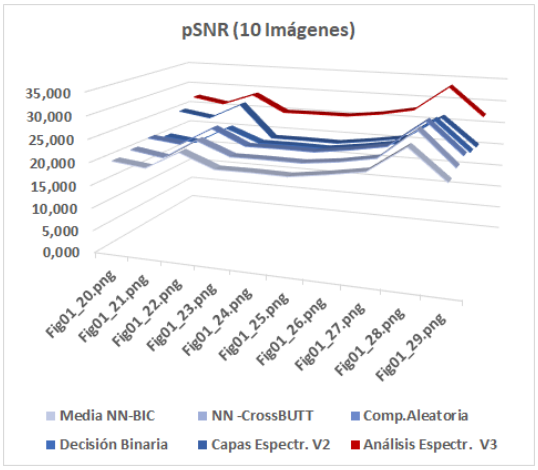




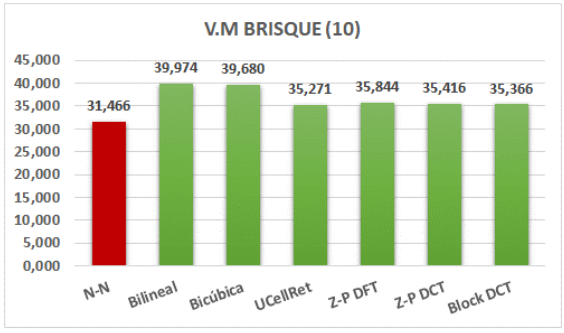


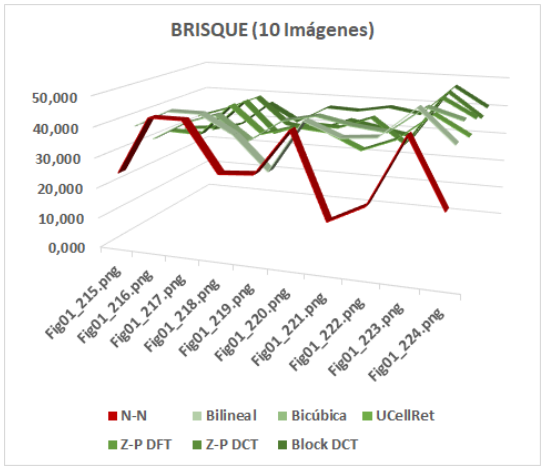


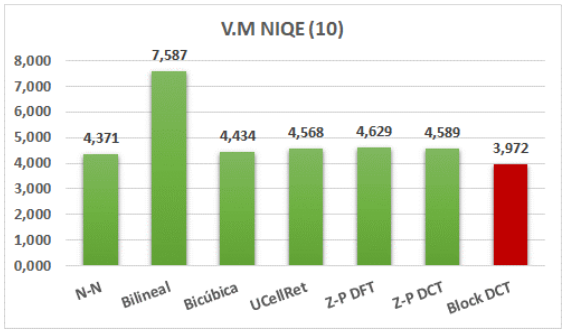


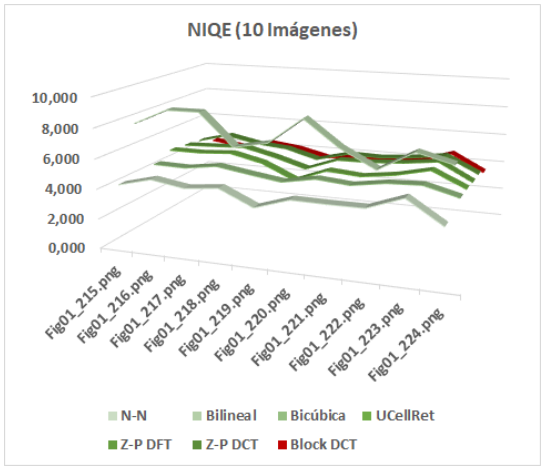


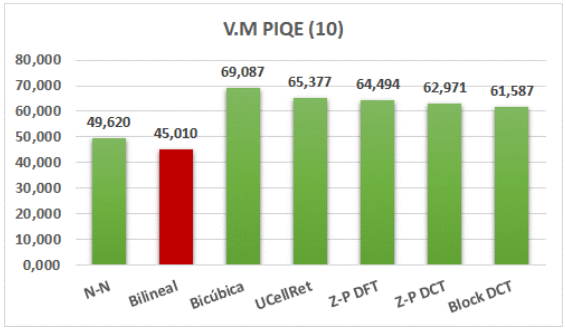
III) Medidas de calidad sin referencia, técnicas existentes:

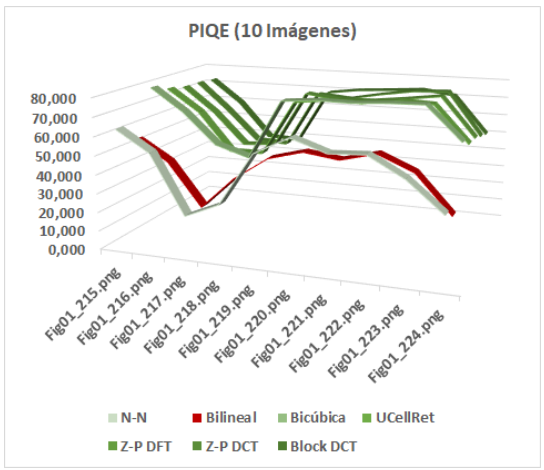












IV) Medidas de calidad sin referencia, técnicas propuestas: