Implementación y evaluación de algoritmos de fusión de imagen en el contexto de la imagen médica

Autor: Antonio Marín García
Director: Dr. Jorge Larrey Ruiz

Septiembre 2013

Autor	Antonio Marín García
E-mail del Autor	amg456@hotmail.com
Director	Jorge Larrey Ruiz
E-mail del Director	jorge.larrey@upct.es
Título del PFC	Implementación y evaluación de algoritmos de fusión de imagen en el contexto de la imagen médica
Resumen	Estudio comparativo de varias reglas de fusión de imagen sobre pares de imágenes procedentes de tecnologías radiológicas distintas y relativas a un mismo paciente. Extracción de conclusiones en base a los resultados obtenidos, pretendiendo colaborar así en el camino hacia la definición del mejor algoritmo de fusión posible que pueda definir la más óptima detección de patologías en pacientes. El objetivo principal sería pues determinar qué técnica de fusión de imagen presenta el mejor compromiso entre unos valores de los parámetros que definen la calidad de la imagen considerados como aceptables o buenos incluso, junto con una apariencia de la imagen fusionada lo suficientemente buena como para permitir a los radiólogos una mejor definición de la patología presentada.
Titulación	Ingeniero Técnico de Telecomunicaciones, Especialidad Telemática
Departamento	Tecnologías de la información y las comunicaciones. Grupo de teoría y tratamiento de la señal.
Fecha de Presentación	Septiembre 2013

A los enfermos y familiares de enfermos del Hospital de Día Médico Morales Meseguer de Murcia; a los que esperan con esperanza, a los que esperan luchando por mantenerla, a todos ellos.....

A todo el equipo de Doctores, enfermeros/as y auxiliares de dicho centro, en especial y con gran estima al Doctor José Javier Sánchez Blanco, los cuales desarrollan a diario una labor profesional y humana muy necesaria y seguramente nunca lo suficientemente reconocida.

Decía el célebre Einstein que sólo hay dos cosas infinitas, el Universo y la estupidez humana. Yo añadiría una tercera. La paciencia de mis padres. Sin ella no habría llegado hasta aquí, no hubiera realizado este trabajo, no hubiera conseguido ninguna meta de las que he alcanzado hasta hoy, y sin duda sería un ser aun más imperfecto de lo que soy. Si la vida les hubiera ofrecido las oportunidades de las que yo he dispuesto, seguro las habrían aprovechado mucho mejor. El mérito de este trabajo es mucho más suyo que mío. Uno de mis objetivos es poder algun día parecerme a ellos.

De igual forma agradecer al director de mi proyecto, el Doctor Jorge Larrey Ruiz, su imprescindible aportación al presente trabajo. Muy especialmente resaltar su flexibilidad y comprensión en los momentos en los que mi actitud no fue la que cabría esperar.

Por último nombrar a dos personas muy importantes en mi vida como son mi novia Paqui y mi hermano Miguel Ángel. La primera ha sido mi inseparable compañera de fatigas durante toda mi época universitaria y perfecta conocedora de los sinsabores y menos numerosos éxitos vividos en dicho período. Mi hermano por otro lado no es sino aquello a lo que siempre me quise parecer desde que alcanzan mis recuerdos. Mi perfil personal está estrechamente vinculado a su influencia sobre mi.

¡Gracias a todos!

Índice de figuras

Índice general

Capítulo 1

Introducción

A raíz de diversos estudios que revelan importantes avances en el diagnóstico médico por imagen provocados por la presencia de imágenes médicas que resultaron del proceso de fusión de un par de imágenes utilizadas de partida, se consideró la conveniencia de acometer un estudio exhaustivo dentro del campo de la fusión de imágenes en contextos de imagen médica. Este proyecto pues tiene como objetivo prioritario la implementación y posterior evaluación de distintos algoritmos de fusión de imágenes médicas.

Para alcanzar dicho fin se ha partido de conjuntos de pares de imágenes para proceder a su fusión mediante la implementación de varios algoritmos distintos, con el objetivo de alcanzar unas conclusiones tanto objetivas, en función de unos datos cuantitativos concluyentes, como pudieran ser la relación señal a ruido o el coeficiente de correlación, entre otros, como subjetivas, a través de la opinión de un facultativo especialista en Radiología, el cual ha vertido su opinión sobre qué imágenes definitivas ya fusionadas le son más amigables para un mejor diagnóstico de la posible dolencia del paciente.

Los algoritmos que se han implementado son los que se conocen como algoritmos multirresolución, los cuales pertenecen a la categoría de nivel de píxel, y cuyo funcionamiento se basa en la descomposición piramidal de las imágenes de partida, obteniendo réplicas de las mismas a diferentes resoluciones.

El proceso completo de fusión que se ha desarrollado en este trabajo consiste en tomar un par de imágenes originales procedentes de tecnologías radiológicas distintas y relativas a un mismo corte anatómico en un paciente dado, y una vez registradas ambas pasar cada una de ellas por separado a un dominio transformado, en concreto al dominio wavelet para el caso que nos ocupa, por medio de la transformada wavelet discreta bidimensional (DWT), y aplicar posteriormente una de las reglas de fusión que se han desarrollado a dichas imágenes transformadas. Por último se obtiene la imagen final resultado de la fusión mediante la transformada wavelet discreta inversa (IDWT).

La memoria se compone de un repaso del Estado del Arte en el que se hace un breve recorrido por los inicios de la Radiología y se comentan las principales tecnologías de imagen médica. También se comenta de manera introductoria el proceso de registro de imágenes, puesto que para poder fusionar un par de imágenes

médicas éstas han de estar previamente registradas. A continuación se puede encontrar el trabajo en sí, el cual consiste en implementar dentro del entorno MATLAB los algoritmos de fusión de imagen citados. El siguiente apartado relata cómo se ha desarrollado una interfaz gráfica de usuario también en MATLAB con la que poder interactuar a la hora de formar los pares a fusionar y de elegir el método de fusión deseado. Seguidamente se desglosa la toma de datos de las medidas para clasificar las reglas de fusión y se relata la opinión del radiólogo que se ha ofrecido a hacer una valoración. Por último se extraen las conclusiones finales de este trabajo.

Los tres tipos de imágenes médicas que han servido como objeto de estudio han sido:

- Imagen por resonancia magnética (MRI).
- Tomografía axial computarizada (CT o TAC).
- Tomografía por emisión de positrones (PET).

A partir de ellas se han realizado tres tipos de fusión distintos: PET-TAC, MR-PET y MR-TAC.

Figura 1.1: Imagen por resonancia magnética.

La fusión de imagen médica es una técnica de reciente aparición y que tiene por ello unas posibilidades de desarrollo y perfeccionamiento enormes, con lo que puede colaborar activamente y de hecho ya lo hace en la mejor y más concreta detección de patologías en pacientes en el subrango del diagnóstico por imagen, pues estudios de diversa índole revelan que dicha técnica permite determinar de manera más precisa la presencia o ausencia de malignidad sobre partes anatómicas que aparentan un cambio morfológico, colaborando en la disminución de falsos positivos, y en definitiva aportando claros avances en el camino hacia un diagnóstico radioclínico lo más óptimo posible.

Existen evidencias que hacen pensar que en la fusión de imágenes encontramos ciertas ventajas. Gracias a la imagen PET-TAC fusionada, pudiera ser que se accediera a una mejor localización de las lesiones del paciente o a un menor número de falsos positivos, disminuyendo los resultados equívocos o no concluyentes y discriminando entre tejido maligno y tejido inflamado, respecto al estudio de las imágenes TAC y PET por separado. La imagen PET-TAC colabora también en la simulación en radioterapia mediante la combinación de información a nivel funcional o

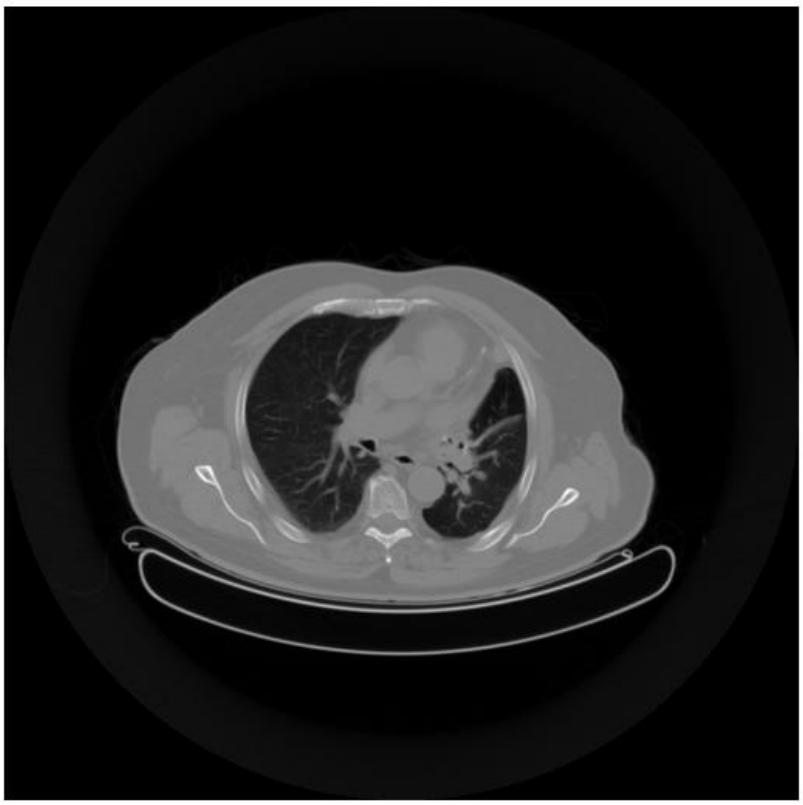


Figura 1.2: Tomografía axial computarizada.

metabólico que proporciona la imagen PET y la información a nivel anatómico o morfológico que facilita la imagen TAC. La fusión MRI-TAC por su parte es muy útil en el campo de la radiocirugía, permitiendo aprovechar lo mejor de ambas modalidades, aportando la imagen TAC rigurosos requisitos de exactitud geométrica y la MRI una óptima diferenciación de tejidos. En los inicios del estudio de la fusión de imágenes médicas, se optó como una de las posibles vías por el corregistro de dos imágenes independientes, es decir, se obtienen ambas imágenes en sesiones y máquinas diferentes y después se procede a su fusión mediante herramientas software, lo cual presentaba como principal inconveniente que al tomarse las dos imágenes en diferentes sesiones y máquinas es normal que cambie la postura del paciente y que las imágenes no se correspondan con una alineación exacta.

Por contra, existe otro método de fusión más reciente mediante unos sistemas híbridos, los cuales integran en la misma máquina dos escáneres distintos situados en línea y que permiten la adquisición de imágenes combinadas mediante una sola sesión. El ejemplo más claro de este último es la cámara PET-TAC. En la actualidad diversas corporaciones como puedan ser Siemens, Philips o General Electric ofrecen este tipo de equipamientos, cada uno con su mecanismo implementado de fusión de imagen. Pero, ¿es ése algoritmo el más óptimo que se podría implementar?

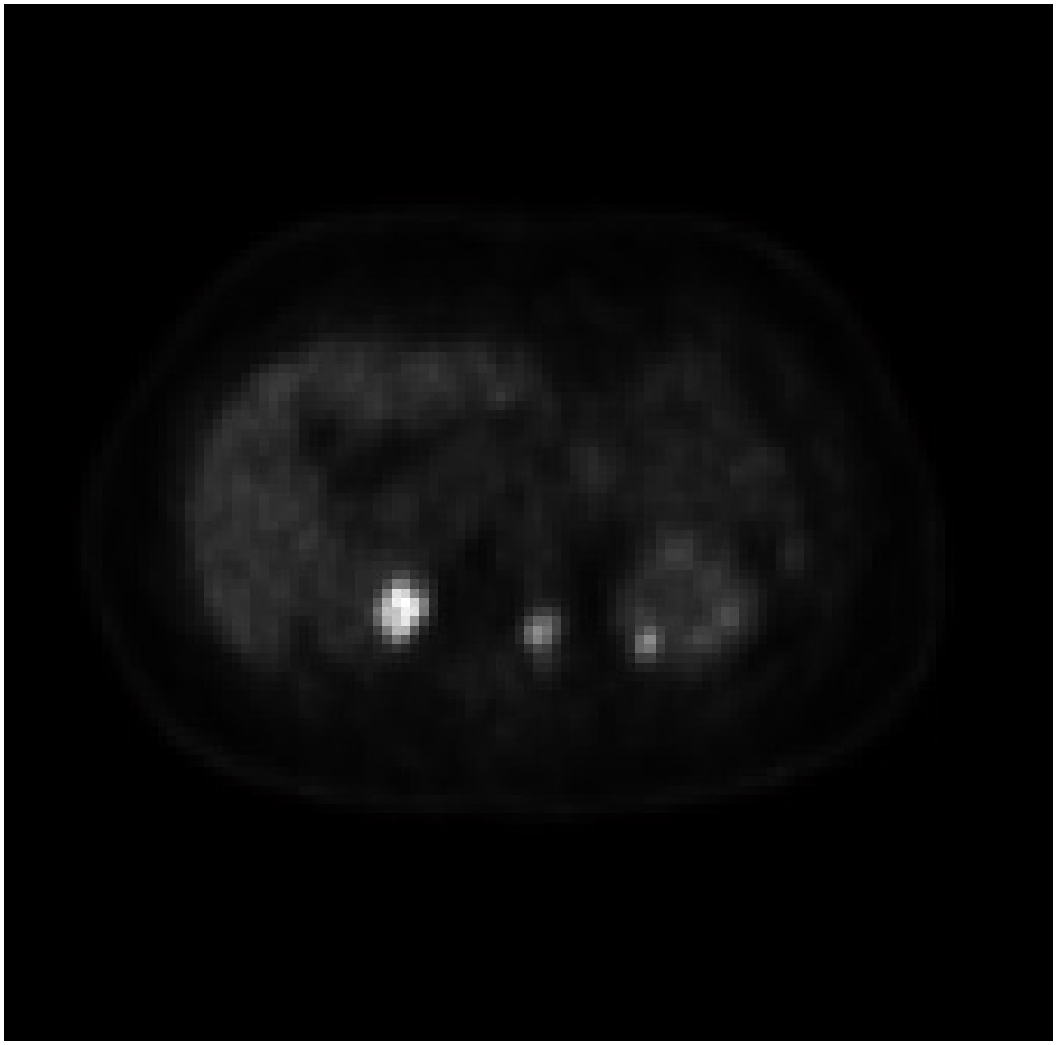


Figura 1.3: Tomografía por emisión de positrones.

Se considera pues, a raíz de ésto último, de una vital importancia el tener constancia de un estudio en profundidad de diversos algoritmos de fusión sobre un conjunto de parejas de imágenes lo suficientemente significativo como para poder inferir unas conclusiones que pudieran colaborar en el camino hacia un mecanismo de fusión lo más óptimo posible.

Se cree por tanto que este trabajo pudiera aportar un resultado esclarecedor sobre qué algoritmo de fusión de imagen prevalece como más eficaz en términos de presentación simultánea de información complementaria frente al resto, aportando así una ayuda para la mejor y más concreta deteccion de patologías en pacientes.

Capítulo 2

Estado del Arte

2.1. Inicios y evolución de la imagen médica. Introducción a la Radiología

La imagen médica puede ser considerada como la técnica en la cual se apoya el campo de la medicina para crear imágenes de la anatomía humana, ya sea de forma parcial o global, con el objetivo de utilizarla bien en el área de la investigación médica, para poder definir el funcionamiento del cuerpo humano y sus patologías de forma más rigurosa, o bien como mera herramienta complementaria de diagnóstico clínico.

Los primeros pasos que se dieron fueron, al menos que se tenga constancia, allá por el siglo XVI, cuando aparecieron las primeras publicaciones en torno a imágenes anatómicas humanas, tales como fueron el *Tabulae Anatomicae Sex* o más tarde el *De Humani Corporis Fabrica*, ambos publicados por Andreas Vesalio y que supusieron un importante avance, ya sea sólo por el hecho de que hasta ese momento era considerado un tema cuanto menos controvertido, el cual era mirado con recelo por Instituciones y parte de la sociedad.

Pero no fue ya hasta el año 1895 cuando sucedió quizá el hecho más relevante concerniente a este campo, el cual marcó el devenir y trazó las directrices sobre las que se apoyó el campo de la imagen médica para su desarrollo y perfeccionamiento y llegar así a ser la potente herramienta que es a día de hoy en el marco de la medicina. En el otoño de aquél año el alemán Wilhem Conrad Röntgen, movido por la curiosidad de unos experimentos realizados con un tubo de rayos catódicos por Hittorf y Crookes, descubrió que una pieza de cartón cubierta con cristal de platinocianuro de bario, se volvía fluorescente cuando un haz de rayos catódicos pasaba a través de un tubo de Hittorf. Paulatinamente pudo demostrar que esta fluorescencia ocurría a pesar de colocar diferentes objetos entre el tubo y la superficie fluorescente, mientras que el plomo sin embargo no permitía el paso de estos nuevos rayos.

Posteriormente registró la existencia de esta radiación sustituyendo la pantalla fluorescente por una placa fotográfica. Al colocar un trozo de platino sobre dicha placa y exponerla a la fuente de radiación, apareció una zona clara en la placa dibujando el área en la que el platino había absorbido la radiación. Este hallazgo inesperado lo confirmó Röntgen colocando la mano de su mujer sobre una funda que

contenía una placa fotográfica y haciendo una exposición de nada menos que quince minutos observó que los huesos se veían blancos y la carne de una tonalidad más oscura sobre la placa revelada.

Figura 2.1: Primera imagen tomada por rayos X. Mano de la mujer de Röntgen.

El 28 de diciembre de 1895 presenta en la Sociedad de Física Médica de Würzburg la publicación *Sobre una nueva clase de rayos: Comunicación preliminar*, y en 1901 recibe a raíz de su hallazgo el Premio Nobel de Física.

Röntgen denominó a estos rayos *rayos X*, y desde su descubrimiento y junto con otros hechos relevantes como fuera el descubrimiento de la radiactividad natural por Bequerel en 1898, el campo de la Radiología ha experimentado un progreso exponencial, facilitando la aparición de nuevas técnicas de imagen médica. Los rayos X pertenecen al espectro de radiaciones electromagnéticas, y se originan cuando los electrones inciden con muy alta velocidad sobre la materia y son frenados repentinamente, estando la radiación X así producida formada por muchas y variadas longitudes de onda constituyendo lo que se conoce como *espectro continuo*. Poseen diversas propiedades importantes como son las capacidades de penetrar en la materia, de producir luz sobre las superficies en las que inciden, de producir cambios en las emulsiones fotográficas, de ionizar los gases o de producir cambios en los tejidos vivos.

A partir de dicho fenómeno aparecen distintos tipos de tecnologías radiológicas como son por ejemplo la radiografía, la fluoroscopia o la tomografía, tanto en su vertiente convencional como computarizada, las cuales se apoyan en la técnica de los rayos X. De forma paralela se desarrollaron otras técnicas no apoyadas en los rayos X pero no por ello menos relevantes como la resonancia magnética o la ecografía. La inmensa mayoría de los distintos tipos de imágenes médicas no podrían haberse desarrollado sin la ayuda de la tecnología de procesamiento digital de imágenes.

Todos estos tipos de imágenes médicas contienen gran cantidad de información, por lo que la mayoría de procesos de tratamiento de imágenes exigen la utilización

de un sistema informático complejo, capaz de convertir una imagen en datos digitales inteligibles para dicho sistema con el fin de ser sometidos a diversos algoritmos englobados dentro del procesamiento y tratamiento de imágenes.

Este hecho provoca pues la distinción entre Radiología convencional, que sigue siendo una herramienta muy útil utilizada sobre todo en la evaluación inicial del paciente, y Radiología computarizada, utilizada para obtener información diagnóstica adicional y, en algunos casos, ha reemplazado a la Radiología convencional en el manejo inicial del paciente. La Radiología computarizada permite entre otras ventajas una mayor calidad de imagen y una menor dosis de radiación para el paciente.

2.2. Principios básicos de la imagen.

Una imagen como tal es una distribución monocromática en dos dimensiones, generalmente en blanco y negro, la cual aporta algún tipo de información a quien la observa. Sus dos características principales son su tamaño y contorno, tiene una amplitud horizontal y una vertical y el conjunto de ambas determina la dimensión de la imagen.

Si la imagen resulta agradable a la persona que la observa de acuerdo a criterios subjetivos, se dice que la imagen es estética. Si simplemente debe aportar un mensaje sin especiales cualidades métricas, se dice que es informativa. Si por contra los detalles de la imagen deben tener un valor cuantitativo específico, se le denomina imagen métrica.

En la literatura de los procesos electrónicos, la anchura viene representada en el eje X, la altura en el eje Y y el nivel de brillo de cada punto en el eje Z, pasando a tener la imagen tres dimensiones.

Cuando la imagen tiene una distribución continua de brillos, delimitada por un valor máximo o mínimo el cual está distribuido de manera continua en el espacio pero sus límites sólo estarán en los márgenes de la imagen, se dice que es una *imagen analógica*.

Si la distribución de brillos tiene una representación de sus valores máximo y mínimo, con unos límites concretos, con los extremos de dichos valores bien establecidos, formando una escala de grises, que no es sino el número de tonalidades que se pueden representar, y contando cada valor de gris con un valor discreto, se conoce la imagen como *imagen digital*, y a cada uno de sus elementos se le denomina *píxel*. Al digitalizar una imagen analógica, se perderá algo de información, pero a cambio la información se puede cuantificar y por tanto se puede actuar sobre ella modificándola si fuera necesario para obtener así ciertas mejoras.

El conjunto de celdas elementales o píxeles de la imagen digitalizada forman el total de la superficie de la imagen, y el brillo de cada celda es igual al brillo medio de la superficie correspondiente en la imagen original. La imagen digital será pues en esencia una secuencia ordenada de bits, entendiendo por bit la unidad mínima de información, que toman valores discretos, bien 0 ó bien 1.

Es difícil poder definir una imagen de forma objetiva. Sin embargo, existen parámetros que permiten clasificar la calidad de la misma, la cual es un elemento primordial en el rendimiento diagnóstico. Esos parámetros se conocen como factores de calidad y, aunque no permiten una *combinación milagrosa*, sí que pueden clasificar una imagen con un estándar de calidad aceptable mediante un compromiso óptimo de todos ellos. No obstante, dicho compromiso se verá condicionado por la duración de la adquisición y el tipo de patología estudiada.

Dichos parámetros responden a medidas físicas tales como la relación señal/ruido, el contraste o la resolución espacial. La relación señal/ruido es quizá el factor que más condiciona la calidad de la imagen, y tendrá una influencia directa sobre el

contraste y la resolución espacial. Se busca siempre la imagen con la mejor relación señal/ruido que permite la mejor resolución espacial posible.

El ruido es considerado como el conjunto de señales indeseables que degradan la información de la imagen. Es aleatorio y proviene tanto del sistema electrónico de tratamiento de la señal como de los movimientos moleculares y de los artefactos, ya estén dichos artefactos relacionados con el equipo de toma de imágenes o bien relacionados con el paciente.

El contraste se define como la variación de la intensidad de la señal entre dos estructuras adyacentes. La resolución espacial por su parte permite determinar la dimensión del mínimo volumen observable.

Para cada imagen existe un compromiso ideal entre estos tres parámetros, si bien todos ellos están influidos por los parámetros que se tienen en cuenta para el cálculo del tiempo de adquisición. En resumen, el objetivo máximo de una imagen radiológica es obtener una reproducción lo más cercana posible a la realidad, que en este caso no es otra que los órganos y anatomia interna en general del paciente.

2.3. Principales tipos de imágenes médicas

2.3.1. Radiografía

La radiografía convencional obtiene una imagen sobre una película radiográfica en la que el plano elegido de la anatomía se mantiene enfocado sometiéndose a un haz de rayos X en donde la absorción que sufre el haz de rayos X depende del espesor de la sustancia atravesada y de los átomos que la constituyen, mientras que las estructuras situadas delante y detrás de este plano aparecen intencionadamente borrosas. Para conseguir esto, el tubo de rayos X y la película se desplazan simultáneamente en sentidos opuestos durante la exposición de la imagen tomográfica, cubriendo un arco de 8 a 40 grados. Cuanto mayor sea este arco, más fino es el grosor del corte obtenido.

Además de movimientos lineales simples, los tubos de rayos X pueden realizar movimientos más complejos, que pueden aumentar la borrosidad de los tejidos fuera del plano y por tanto mejorar la calidad de imagen del corte tomográfico. Debido al escaso contraste natural entre estructuras adyacentes de densidad radiológica similar, en ocasiones los estudios se ven obligados a ir acompañados de un estudio con contraste.

Se puede decir por tanto que en esencia una radiografía es la sombra producida por el cuerpo humano cuando se le ilumina con un haz de rayos X, quedando ésta reflejada sobre la película radiológica anteriormente comentada.. Aunque ha sido sustituida por la tomografía computarizada en la mayoría de los casos de estudio, sigue siendo de gran utilidad como herramienta de diagnóstico.

2.3.2. Tomografía computarizada

Esta nueva técnica radiológica fue descrita en 1972 por Godfrey Hounsfield, recibiendo por ello el Premio Nobel de Medicina en 1979 compartido con A.M. Cormack, quien también había trabajado intensamente en sus principios básicos.

Se podría decir muy esencialmente que un estudio de tomografía computarizada consiste en imágenes de cortes transversales de un objeto obtenidas con el conjunto de imágenes de proyección del objeto en el plano a estudiar. Dichas imágenes se generan en un ordenador, lo que permitirá su posterior tratamiento digital, y se obtienen mediante un haz de rayos X rotatorio y un sistema de detectores que rodean al paciente en forma radial, razón por la cual se coloca al paciente en una máquina con forma de aro en el momento de realizar la exploración. Avances recientes han permitido adquirir simultáneamente mútliples imágenes en cada rotación del tubo de rayos X.

La mayor resolución de contraste de la tomografía computarizada respecto a la convencional permite mostrar las imágenes capturadas con diferentes valores de *ventana* y *nivel* para ver mejor las diferencias de densidad o atenuación entre tejidos. No obstante, es necesario con cierta frecuencia administrar vía intravenosa un contraste yodado al paciente.

Según sea la naturaleza del plano a estudiar, se distingue entre tres tipos de cortes:

- Axial.
- Coronal.
- Sagital.

Si el corte es axial, se le denomina *tomografía axial computarizada* ó TAC, y es una técnica muy utilizada en la actualidad como herramienta de diagnóstico.

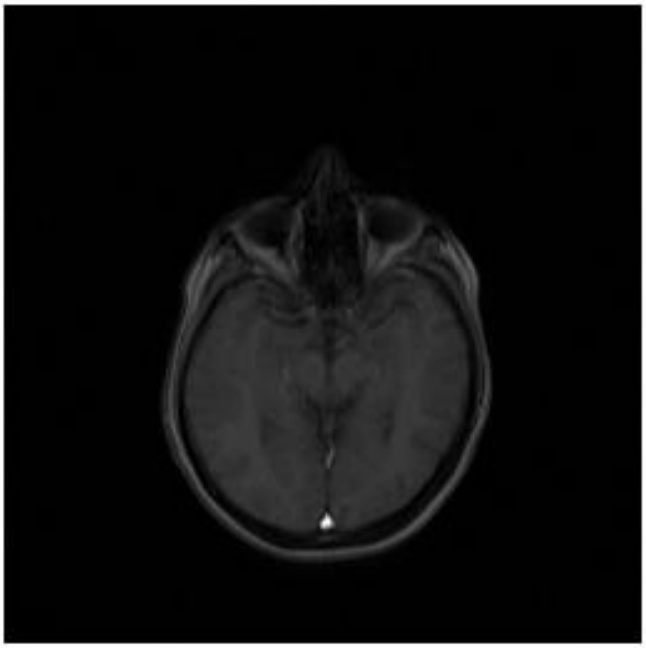


Figura 2.2: Corte axial.

Según sea el método concreto de toma de imágenes, se distingue entre otros tres modos:

- Transmisivo.
- Emisivo.
- Reflectivo.

La tomografía computarizada aporta como principales ventajas el ser una técnica económica y fácilmente disponible, la cual puede ser usada en pacientes con implantes no compatibles con la resonancia magnética. Sus imágenes resultantes, al contrario que las radiografías simples, muestran de forma clara los tejidos blandos y permiten diferenciar unos de otros. Por contra, somete a los pacientes a radiaciones ionizantes y sus imágenes serán de muy baja calidad si el paciente se mueve, si bien en ese caso se pueden repetir rápidamente. En ocasiones la presencia de objetos metálicos puede provocar el ocultamiento de lesiones.

Medicina nuclear

Los estudios de medicina nuclear pertenecen a la familia de la tomografía por modo emisivo. Es una técnica especial, en la que para obtener las imágenes se administra un radiofármaco al paciente para a posteriori registrar su distribución por el

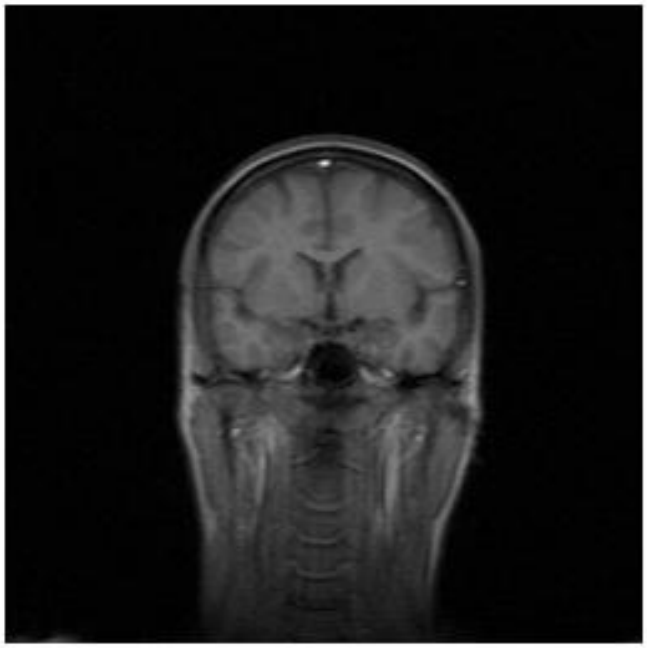


Figura 2.3: Corte coronal.

cuerpo a lo largo de un determinado período de tiempo. Es una técnica muy sensible aunque relativamente inespecífica en la detección de patologías, por lo tanto es conveniente acompañarla del historial clínico, exploración física, datos de laboratorio y otras técnicas de imagen para interpretar de forma más precisa la patología que presenta el paciente, maximizando así los beneficios clínicos que de ello se pudieran derivar.

Los radiofármacos son compuestos radioactivos que típicamente no presentan respuesta fisiológica alguna, y cuya captación selectiva por partes de distintos órganos constituye la base de la imagen nuclear. De hecho el radiofármaco debe estar involucrado en el metabolismo fisiológico del órgano para obtener una imagen satisfactoria. Están formados por el propio compuesto que se distribuye en determinados órganos debido a distintos mecanismos fisiológicos, y un radionúclido pegado a dicho compuesto que emite rayos gamma permitiendo la detección del compuesto en el cuerpo. Proporcionan información a nivel metabólico, si bien su resolución espacial es relativamente pobre.

Presentan como principales ventajas la capacidad de localizar la distribución de un radiofármaco en tres dimensiones junto con la posibilidad de cuantificar la captación del radiofármaco y generar imágenes dinámicas de dicha captación por el órgano. Desgraciadamente es a día de hoy una tecnología muy costosa, y que necesita además paralelamente de un ciclotrón capaz de generar el radionúclido o

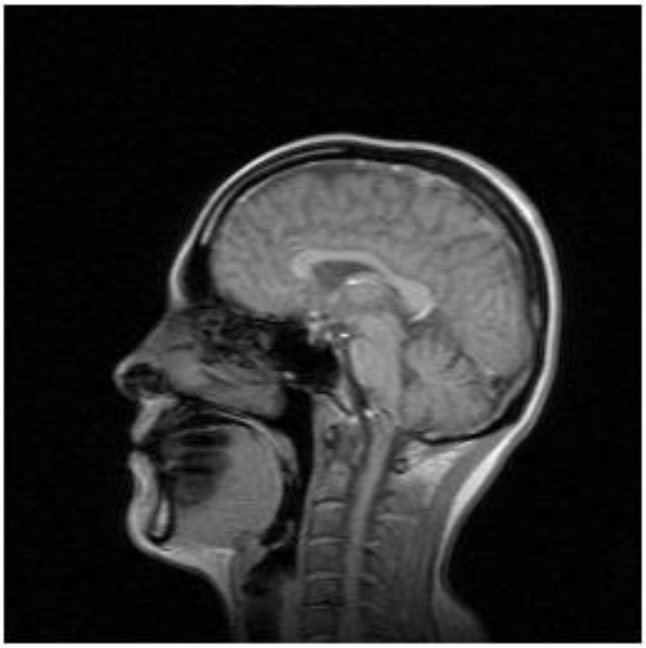


Figura 2.4: Corte sagital.

isótopo radiactivo, de un laboratorio químico de radiofarmacia que acabe de sintetizar el radiofármaco, así como evidentemente de un potente sistema computacional con capacidad para procesar las imágenes y realizar la reconstrucción tomográfica.

Los estudios de medicina nuclear se pueden realizan con gammacámaras, las cuales convierten los fotones emitidos por el radionúclido inyectado al paciente en pulsos luminosos, para pasar éstos ultimos a convertirse en señales eléctricas que serán utilizadas para producir una imagen de la distribución del radionúclido.

Cuando la tecnología de imagen es de tipo tomográfico y se basa en la emisión de un fotón único, utilizando una gammacámara rotatoria, hablamos de tomografía computarizada por emisión de un fotón único, SPECT.

Cuando el radiofármaco administrado para generar imágenes tomográficas seccionales y volumétricas del cuerpo entero mediante la detección de rayos gamma es una 18-Fluorodesoxiglucosa, compuesta del radionúclido 18Flúor y de la molécula que actúa de trazador desoxiglucosa, se está hablando concretamente de la tecnología de medicina nuclear conocida como *tomografía por emisión de positrones* con 18FDG o FDG-PET.

Los rayos gamma se generan cuando los positrones (rayos beta) emitidos por el radionúclido interaccionan con los electrones, momento en el cual se emiten dos

fotones gamma que se propagan en direcciones opuestas y que son capatados de forma simultánea por el anillo con detectores que rodea al paciente, permitiendo así precisar la localización donde se produjo el choque de partículas.

Esta técnica es muy sensible a la hora de detectar alteraciones metabólicas en las células tumorales. De hecho es actualmente la tecnología de imagen médica utilizada para el diagnóstico y estadificación de la inmensa mayoría de tumores, y aunque es una técnica muy sensible para este tipo de diagnóstico, hay que contar con la posibilidad de que no detecte masas tumorales residuales, por lo que es conveniente acompañar el estudio diagnóstico de otras técnicas al alcance del facultativo. Es por tanto una herramienta muy potente para el campo de la Oncología. También es utilizada con menor frecuencia en la Neurología y Cardiología.

2.3.3. Resonancia magnética

El fenómeno cuya denominación original fue de resonancia magnética nuclear, fue descubierto de forma paralela e independiente por Félix Bloch y Edward Purcell en 1946, hecho por el cual recibieron el Premio Nobel de Física en 1952. Entre los años 1950 y 1970 se utilizó para análisis molecular físico-químico, y a partir de esa fecha se empezó a demostrar que podía ayudar en el diagnóstico de enfermedades. Ya en 1975 Richard Ernst empleó una codificación de fase y frecuencia junto a una transformada de Fourier para constituir lo que iba a ser la base para la obtención de imágenes por resonancia magnética en la actualidad.

Dicha técnica utiliza un haz pulsado de radiofrecuencia en presencia de un intenso campo magnético, para generar imágenes de alta calidad del cuerpo humano en cualquier tipo de plano. Los núcleos de hidrógeno son los elegidos para este tipo de imagen. Así, cuando el paciente se coloca en el equipo de resonancia magnética, los átomos de hidrógeno, previamente orientados al azar, se alinean con el campo magnético estático. Para detectar la señal se aplica transitoriamente un pulso de radiofrecuencia, que produce una modificación neta en la alineación de dichos núcleos atómicos. Cuando dicho pulso de radiofrecuencia cesa, los núcleos vuelven a su estado de equilibrio, y a este período de reorientación se le denomina tiempo de relajación, y engloba a los tiempos de relajación longitudinal y transversal. El primero representa la recuperación de la magnetización longitudinal, en la dirección del campo magnético principal, y el segundo hace referencia a la pérdida de la magnetización en el plano transversal, perpendicular al eje del campo.

Al igual que la tomografía computarizada está formada por diversas imágenes tomográficas reconstruidas en un ordenador, aunque aquí por contra la información que se recoge no es la atenuación de un haz de rayos X, sino una manifestación gráfica de los datos de resonancia magnética nuclear que se recogen de los núcleos atómicos de los tejidos orgánicos. Dicho de otro modo, para generar las imágenes utiliza la información proporcionada por la distribución del hidrógeno presente en las moléculas de agua en un organismo humano. Dependiendo de cómo se combinan esas moléculas de agua el tiempo de relajación aumenta o disminuye, permitiendo así obtener una óptima diferenciación entre distintos tipos de tejidos y entre tejidos sanos y patológicos.

La imagen por resonancia magnética se ha erigido como la técnica de imagen más sensible para el estudio de lesiones espinales, y presenta como principales ventajas una magnífica resolución de contraste, una elevada resolución espacial y la ausencia de radiaciones ionizantes. Por contra, sus equipamientos tienen un coste elevado y es una técnica lenta.

Conclusiones y objetivos a alcanzar

Con el desarrollo de nuevos métodos en el diagnóstico médico por imagen surge la necesidad de combinar con criterio y *de forma espacialmente correcta* todos los conjuntos de datos de imágenes disponibles.

En ocasiones, podría ocurrir que la información facilitada por las distintas técnicas de imagen médica fuera confusa o no concluyente. A menudo una imagen funcional no ofrece el nivel de detalle anatómico suficiente como para poder determinar la posición exacta de un tumor u otra lesión.

Generalmente, las imágenes funcionales tienen baja resolución espacial mientras que las imágenes anatómicas tiene una resolución espacial alta. Así pues, con las imágenes anatómicas se puede detectar una lesión con exactitud de milímetros, mientras que por el contrario con las imágenes funcionales esto no es posible pero sin embargo posibilita la pronta detección de lesiones previa a la afectación anatómica general.

La combinación de ambos tipos de imagen pretende ofrecer una información clínica adicional no facilitada aparentemente en las imágenes por separado, evitando así efectos indeseados.

2.4. El estándar DICOM

A raíz de la inmersión de las computadoras en el ámbito clínico y la consecuente digitalización de las imágenes médicas, en el año 1983 se reunen el Colegio Estadounidense de Radiología (ACR) y la Asociación Nacional de Fabricantes Eléctricos (NEMA) con el fin de desarrollar un estándar que definiera un formato de imagen médica, así como la compatibilidad entre equipos de distintos fabricantes que permitiera la transferencia de esas imágenes médicas y la conexión de dichos dispositivos con otros (los que hoy se conocen como periféricos).

De esa reunión salió constituido un comité que más tarde definió la primera versión del estándar, el ACR-NEMA Standars Publication No. 300-1985 y más tarde una segunda versión ACR-NEMA Standars Publication No. 300-1988.

En 1993 se presenta una tercera versión del formato, preparado para trabajar de forma segura con dispositivos conectados en red y con un lenguaje para las imágenes definido que era entendible por los distintos equipos médicos. Ese estándar se conoce como DICOM (Digital Imaging and Communication in Medicine) y es el formato de imagen médica aceptado en la actualidad.

DICOM define unos protocolos para el manejo, almacenamiento, impresión e intercambio de imágenes médicas dentro de una red compuesta de escáneres, servidores, estaciones de trabajo e impresoras que funcionan dentro de un sistema de almacenamiento y comunicación de imágenes.

Dicho formato consta de la propia imagen (Data Set) junto con una cabecera de datos (Header) donde se almacenan datos del paciente, del dispositivo que obtuvo la imagen y de cómo estaba configurado dicho equipo para realizar el estudio. En dicha cabecera aparecen entre otros, campos como puedan ser:

- *Filename*. Nombre del archivo.
- *Rows*. Número de filas.
- *Columns*. Número de columnas.
- *Modality*. Modalidad de la imagen.
- *Slicelocation*. Número de corte tomado durante la adquisición de imágenes sobre el paciente.

El nombre de los archivos aparece como *nombre del archivo.dcm*, aunque también existe la extensión *nombre del archivo.v2* para una versión más reciente del estándar.

En MATLAB existen una serie de funciones que permiten tratar con imágenes en formato DICOM, como por ejemplo la función $dicomread('*.dcm')$ que permite leer la imagen, la función $dicomwrite(X,'*.dcm')$ que a partir de la variable X genera una imagen en formato DICOM de nombre $*.dcm$, o la función $dicominfo('*.dcm')$ que muestra la información contenida en la cabecera.

```
Command Window
>> dicomread('TAC-004.v2');
>> dicominfo('TAC-004.v2')

ans =

                          Filename: [1x73 char]
                       FileModDate: '17-may.-2006 12:17:54'
                          FileSize: 527606
                            Format: 'DICOM'
                     FormatVersion: 3
                             Width: 512
                            Height: 512
                          BitDepth: 16
                         ColorType: 'grayscale'
    FileMetaInformationGroupLength: 196
        FileMetaInformationVersion: [2x1 uint8]
           MediaStorageSOPClassUID: '1.2.840.10008.5.1.4.1.1.2'
        MediaStorageSOPInstanceUID: [1x54 char]
                  TransferSyntaxUID: '1.2.840.10008.1.2'
             ImplementationClassUID: '1.2.276.0.7230010.3.0.3.5.3'
          ImplementationVersionName: 'OFFIS_DCMTK_353'
               SpecificCharacterSet: 'ISO_IR 100'
                          ImageType: 'ORIGINAL\PRIMARY\AXIAL'
                 InstanceCreationDate: '20060404'
                 InstanceCreationTime: '134406'
                        SOPClassUID: '1.2.840.10008.5.1.4.1.1.2'
                    SOPInstanceUID: [1x54 char]
```

Figura 2.5: Salida por pantalla de la función dicominfo de MATLAB.

Existen también multitud de visores DICOM a nivel de usuario, ya sea del ámbito clínico o no, que permiten tanto visualizar la imagen como acceder a los datos de la cabecera entre otras diversas opciones.

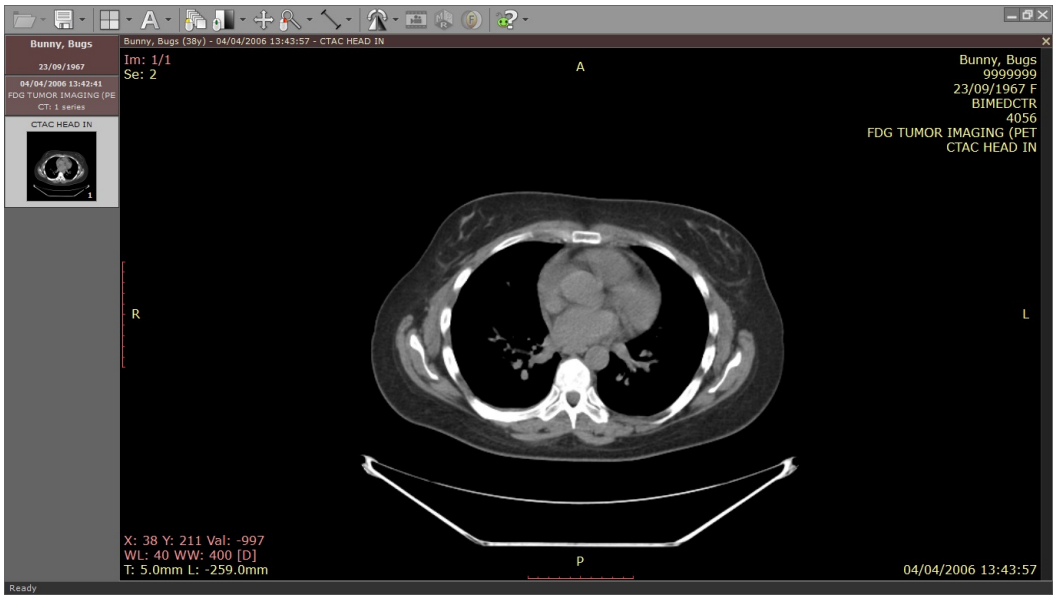
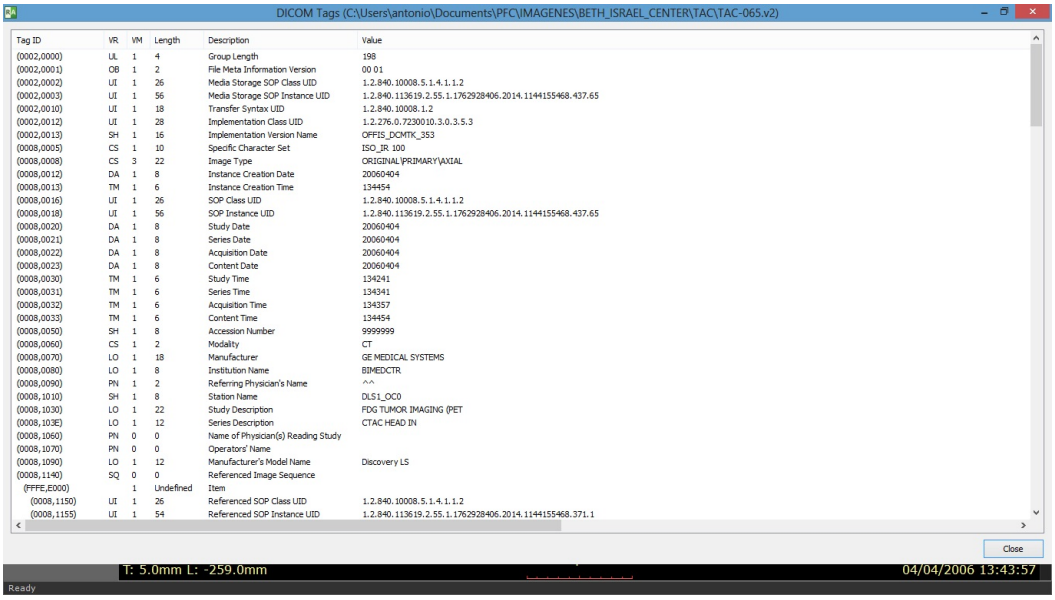


Figura 2.6: Imagen TAC mostrada por un visor DICOM.



Tag ID	VR	VM	Length	Description	Value
(0002,0000)	UL	1	4	Group Length	198
(0002,0001)	OB	1	2	File Meta Information Version	00 01
(0002,0002)	UI	1	26	Media Storage SOP Class UID	1.2.840.10008.5.1.4.1.1.2
(0002,0003)	UI	1	56	Media Storage SOP Instance UID	1.2.840.113619.2.55.1.1762928406.2014.1144155468.437.65
(0002,0010)	UI	1	18	Transfer Syntax UID	1.2.840.10008.1.2
(0002,0012)	UI	1	28	Implementation Class UID	1.2.276.0.7230010.3.0.3.5.3
(0002,0013)	SH	1	16	Implementation Version Name	OFFIS_DCMTK_353
(0008,0005)	CS	1	10	Specific Character Set	ISO_IR 100
(0008,0008)	CS	3	22	Image Type	ORIGINAL\PRIMARY\AXIAL
(0008,0012)	DA	1	8	Instance Creation Date	20060404
(0008,0013)	TM	1	6	Instance Creation Time	134454
(0008,0016)	UI	1	26	SOP Class UID	1.2.840.10008.5.1.4.1.1.2
(0008,0018)	UI	1	56	SOP Instance UID	1.2.840.113619.2.55.1.1762928406.2014.1144155468.437.65
(0008,0020)	DA	1	8	Study Date	20060404
(0008,0021)	DA	1	8	Series Date	20060404
(0008,0022)	DA	1	8	Acquisition Date	20060404
(0008,0023)	DA	1	8	Content Date	20060404
(0008,0030)	TM	1	6	Study Time	134241
(0008,0031)	TM	1	6	Series Time	134341
(0008,0032)	TM	1	6	Acquisition Time	134357
(0008,0033)	TM	1	6	Content Time	134454
(0008,0050)	SH	1	8	Accession Number	9999999
(0008,0060)	CS	1	2	Modality	CT
(0008,0070)	LO	1	18	Manufacturer	GE MEDICAL SYSTEMS
(0008,0080)	LO	1	8	Institution Name	BIMEDCTR
(0008,0090)	PN	1	2	Referring Physician's Name	^^
(0008,1010)	SH	1	8	Station Name	DLS1_OC0
(0008,1030)	LO	1	22	Study Description	FDG TUMOR IMAGING (PET
(0008,103E)	LO	1	12	Series Description	CTAC HEAD IN
(0008,1060)	PN	0	0	Name of Physician(s) Reading Study	
(0008,1070)	PN	0	0	Operators' Name	
(0008,1090)	LO	1	12	Manufacturer's Model Name	Discovery LS
(0008,1140)	SQ	0	0	Referenced Image Sequence	
(FFFE,E000)		1	Undefined	Item	
(0008,1150)	UI	1	26	Referenced SOP Class UID	1.2.840.10008.5.1.4.1.1.2
(0008,1155)	UI	1	54	Referenced SOP Instance UID	1.2.840.113619.2.55.1.1762928406.2014.1144155468.371.1

Figura 2.7: Información de la cabecera de un TAC mostrada por un visor DICOM.

2.5. Equipamientos médicos

2.5.1. Cámara TAC

Los equipos de captación de imágenes por tomografía computarizada aparecen a mediados de los años setenta y siguen en constante evolución. La cámara TAC posee un tubo de rayos X que gira para tomar imágenes a partir de cortes transverales. Las imágenes se obtienen tras someter al paciente a un haz de rayos X, midiendo el equipo la atenuación de la energía del haz emitido y en función de cuanta de esa energía ha sido absorvida por los distintos tejidos del paciente se puede reconstruir la imagen. Una masa ósea absorverá gran parte de la energía y provocará una gran atenuación de la señal, lo que se traduce a la imagen adquirida como parte blanca. Los distintos órganos y tejido blandos absorven mucha menos energía, apareciendo en la imagen con diversos tonos de gris, mientras que las masas de aire dejan pasar el haz sin atenuarlo generando en la imagen una zona de color negro que se corresponde a esa masa de aire.

Para los tratamientos en radioterapia, a la cámara TAC se añade un acelerador de electrones capaz de generar fotones de alta energía capaces de destruir las cadenas de ADN de las células tumorales.

Son varias las corporaciones que ofrecen estos equipos actualmente, como puedan ser entre otras Siemens, General Electric o Philips. A continuación se muestra uno de los equipos de tomografía computarizada de que dispone Siemens:

Figura 2.8: Equipo de tomografía computarizada.

2.5.2. Cámara PET

Al contrario del TAC que es una imagen de transmisión, el PET en una imagen de emisión. Sus equipos disponen de un anillo detector (gammacámara) que registran la emisión de dos positrones que se han emitido en direcciones opuestas, detectando así dónde queda situado el compuesto formado por glucosa y un isótopo radioactivo que se inyectó al paciente vía intravenosa.

2.5.3. Resonancia magnética

Los equipos de resonancia magnética generan un campo magnético que alinea los núcleos de hidrógeno presentes en el agua dentro del cuerpo humano en la dirección del campo. Al cesar el campo, los átomos de hidrógeno vuelven a su estado de reposo en lo que se conoce como tiempo de relajación. A partir de un pulso de radiofrecuencia se mide ese intervalo de tiempo y se genera la imagen con dicha información de la distribución del hidrógeno. Son por tanto maquinarias altamente complejas.

Figura 2.9: Equipo de resonancia magnética de Philips.

2.5.4. Sistemas híbridos

Cámara PET-TAC

Es un equipo híbrido que toma ambos conjuntos de imágenes TAC y PET en una misma sesión sin que el paciente tenga que cambiar de equipo y sin ni siquiera moverse de la camilla, y que además reduce considerablemente los tiempos de adquisición de las imágenes respecto a la cámara PET convencional.

Se están implantando cada vez más y empieza a ser interesante para los centros hospitalarios con necesidades de radiodiagnóstico en oncología, el adquirirlas en detrimento de los equipos TAC y PET por separado.

General Electric por ejemplo dispone de este tipo de equipamiento:

Figura 2.10: Equipo híbrido PET-TAC.

Cámara MR-PET

Es seguramente el equipo híbrido más novedoso en el mercado. De hecho, actualmente tiene un coste muy elevado y no es fácil encontrarlo en cualquier hospital. Durante su desarrollo, la mayor problemática que se encontró fue el hecho de que el campo magnético generado por la resonancia magética dañaba la maquinaria de la parte del PET.

En el caso de la cámara MR-PET, son menos los fabricantes que disponen de ella y la ofrecen en la actualidad.

Figura 2.11: Equipo híbrido MR-PET.

2.6. Registro de imágenes

Para poder implementar la fusión de pares de imágenes tal y como se pretende en el presente estudio, éstas han de estar previamente registradas.

A continuación se va realizar un pequeño recorrido sobre la técnica del registro de imagen, si bien se hará de una forma ciertamente introductoria y sin entrar en detalle ya que no es objetivo principal de este proyecto.

Se considera al registro de imagen como el proceso que permite corresponder geométricamente dos o más conjuntos de datos, entendidos en este caso como imágenes, que representan una misma escena o escenas similares, ya sea bien por estar tomadas en diferentes instantes de tiempo, desde distintos puntos de vista o bien captadas por múltiples sensores. Para alcanzar dicha correspondencia geométrica se harán relacionar los puntos correspondientes de los conjuntos de datos, transformando uno de ellos, que se conoce como imagen objetivo, y haciéndolo corresponder con el otro conjunto denominado imagen de referencia.

Una de las posibles aplicaciones que tiene el registro de imágenes puede ser la medicina, siendo el paso previo al proceso de fusión de dichas imágenes procedentes de modalidades distintas.

Presenta ciertas limitaciones, como cuando por ejemplo la imagen objetivo presenta una información que no se da en la imagen de referencia o viceversa, o bien el simple hecho de que no exista un mecanismo de registro que prevalezca sobre el resto, debido entre otros motivos a la mera dificultad para comparar distintas me-

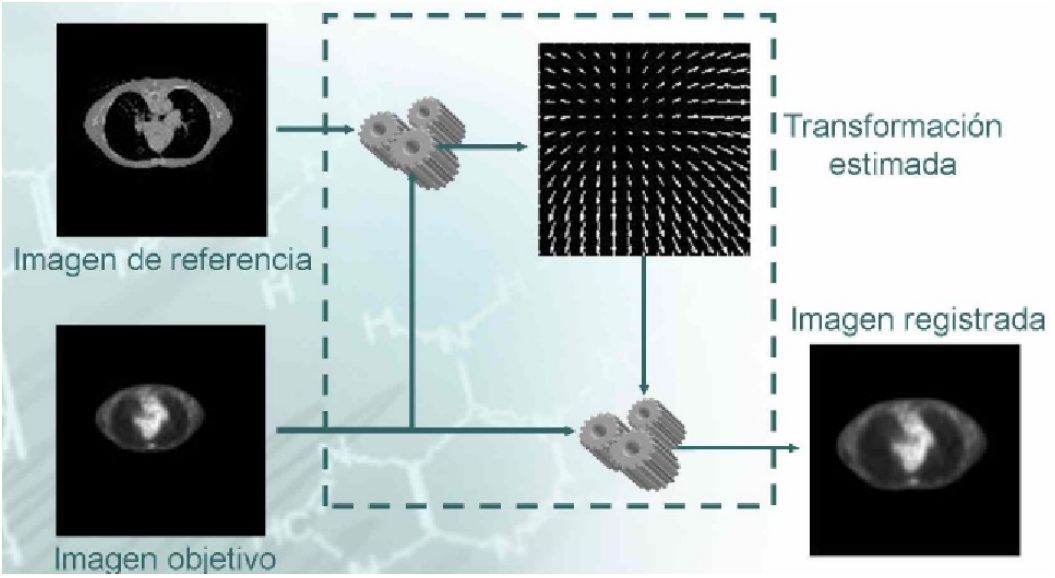


Figura 2.12: Ejemplo de registro de imagen.

todologías de registro entre sí.

Los diferentes métodos de registro se clasifican siguiendo una serie de criterios, algunos de ellos comunes a la mayoría de clasificaciones. Si bien cabe destacar que todas las clasificaciones presentadas hasta la fecha resultan discutibles y hasta cierto punto poco rigurosas.

De entre las distintas clasificaciones de métodos de registro, se podrían destacar los métodos basados en características geométricas, los basados en intensidades, en características icónicas o una familia aparte de métodos de registro conocidos como métodos variacionales que incluyen los siguientes métodos para registrar:

- Elástico.
- Fluido.
- Por difusión.
- Por curvatura.
- Híbrido.

Por otro lado, cabe destacar que la última version 2013 de MATLAB en la que se ha desarrollado este trabajo ofrece la posibilidad de registrar imágenes mediante la función *imregister*, la cual calcula el registro a partir de la imagen objetivo *moving* y de referencia *fixed*.

2.7. Fusión de imágenes

El término fusión alude a la combinación de la información de dos o más conjuntos de datos relativos a una misma escena, procedentes éstos de fuentes distintas, con el objetivo de obtener un conjunto de datos de dicha escena que proporcione un conocimiento nuevo o más exhaustivo de la misma.

El objetivo máximo pues del nuevo conjunto de datos ya fusionado es proporcionar más o mejor información de la que proporcionaban los dos conjuntos de datos primitivos por separado. El conjunto de datos será típicamente una imagen de una escena determinada, y por tanto la fusión de dos imágenes de una misma escena proporcionará una nueva imagen optimizada, conocida como la nueva imagen fusionada y que viene a complementar o mejorar el conocimiento que se tenía de dicha escena en el momento previo a la fusión. El método que la ha proporcionado se llama esquema de fusión. Los datos aparecen en forma de arrays o matrices de números que representan diversas propiedades intrínsecas a una escena.

Gracias a la capacidad de fusionar imágenes de distintas naturalezas, la técnica de fusión ha emergido como una nueva y prometedora área de investigación. Aporta una serie de ventajas, como pudiera ser por ejemplo el almacenamiento más eficiente de la información, reduciendo la dimensión de los datos, al integrar los datos de los dos conjuntos en uno solo, eliminando por tanto la información redundante, pero complementando a su vez la información procedente de los dos conjuntos primitivos, y aportando una mayor cobertura espacial y temporal. La adquisición de datos es a su vez más rápida y económica, y la robustez y rendimiento del sistema se ven incrementados.

Una fusión de imagen satisfactoria reduce bastante los datos a procesar sin que ésto implique una reducción importante de la información de relevancia.

Cuando se utiliza la técnica de fusión de imagen, se deben considerar algunos requerimientos:

. Los algoritmos de fusión no deberían descartar la información contenida en las imágenes de partida.

. Los algoritmos de fusión no deberían introducir artefactos o inconsistencias que pudieran inducir a error a la persona que observa o interferir en cualquiera de los pasos del procesamiento de imagen.

. Los algoritmos deben ser realizables, robustos y deben tener en la medida de lo posible, la capacidad de tolerar imperfecciones como ruido o registros erróneos.

En el contexto de la imagen médica, una imagen fusionada debe proporcionar información nueva o adicional al facultativo respecto a la que le proporcionaban las dos imágenes médicas originales previas a la fusión obtenidas por separado mediante sensores pertenecientes a tecnologías distintas. Esta técnica nos permite pues obtener imágenes médicas multimodales, las cuales presentarán a los facultativos simultáneamente información de la anatomía del paciente a nivel anatómico y me-

tabólico, en el caso de una fusión PET-TAC, o informaciones conjuntas de masas óseas y tejidos blandos en la fusión MR-TAC.

Así por ejemplo, se puede obtener una tomografía por emisión de positrones y una resonancia magnética de un cerebro humano en una imagen. La primera proporciona información funcional de la actividad del cerebro en detrimento de información anatómica alguna, mientras que con la resonancia magnética ocurre justo al contrario. Además, aunque ambas imágenes conciernen en torno a la misma área del cerebro, la imagen PET tiene menos píxeles que la resonancia, por eso se dice que dicha tomografía posee una resolución espacial menor.

El objetivo de cualquier esquema de fusión aplicable para este caso sería unificar las citadas informaciones funcionales y anatómicas en una única imagen con la mejor resolución posible.

Otro caso similar a éste podría ser la presentación simultánea de la imagen PET como fuente de información diagnóstica a nivel metabólico junto con la imagen TAC referida a nivel morfológico, lo que puede resultar de gran interés para el radiólogo.

El principal motor de motivación para la realización de este trabajo es eminentemente la serie de ventajas que las imágenes médicas fusionadas parecen presentan en radiodiagnóstico según diversos estudios realizados, si bien al ser una tecnología de relativa creciente aparición, los estudios comparativos respecto a las tecnologías radiológicas convencionales por separado no son demasiado abundantes y se refieren a tamaños muestrales reducidos.

Estudios publicados que comparan la tecnología híbrida PET-TAC con PET y/o TAC muestran que la principal ventaja de la primera respecto a las otras radica en la mejor o más exacta localización de las lesiones. Así, PET-TAC detecta lesiones no identificadas en PET o TAC, detectando la malignidad que pudiera presentar una lesión de forma más precoz, discriminando entre tejido maligno o tejido inflamado. Aplicándose con el radiofármaco FDG favorece la planificación quirúrgica y disminuye los resultados equívocos o no concluyentes, aportando globalmente una mayor exactitud diagnóstica, modificando el diagnóstico y en ocasiones el tratamiento a administrar al paciente, con los evidentes beneficios que ésto aporta. Permite también hacer una planificación de Radioterapia.

La fusión MR-TAC por su parte es muy útil en el campo de la radiocirugía, permitiendo aprovechar lo mejor de ambas modalidades, aportando la imagen TAC rigurosos requisitos de exactitud geométrica y la MRI una óptima diferenciación de tejidos.

A día de hoy que se sepa no existe un estándar universalmente aceptado para la evaluación del desarrollo de fusión de imágenes.

Reseña histórica

Los primeros trabajos referentes a la fusión de imagen aparecieron a mediados de los ochenta. Así por ejemplo, Peter J. Burt enunció técnicas piramidales laplacianas,

y junto a Edward H. Adelson introdujeron más tarde un nuevo estudio de fusión de imagen basado en descomposición de imagen jerárquica.

También sobre esa época, Adelson reveló el uso de una técnica laplaciana en la construcción de una imagen con un estudio más en profundidad sobre un conjunto de imágenes tomadas con una cámara fija pero con varias longitudes focales distintas. Posteriormente, Alexander Toet y sus colaboradores utilizaron diferentes esquemas piramidales en la fusión de imagen con propósitos de vigilancia en sistemas de defensa.

Otros trabajos más recientes propusieron la composición de imágenes visibles y térmico-infrarrojos, la implementación de fusión de imágenes infrarrojas pasivas y LADAR o el análisis integral de imágenes visuales y térmicas para la interpretación de escenas.

Seguidamente y hasta el dia de hoy se continuan desarrollando múltiples técnicas de fusión de imagen las cuales son aplicadas a campos tan dispares como la medicina, industria, vigilancia o sistemas de defensa que requieren del uso de múltiples imágenes de una misma escena. Se han utilizado así mismo de forma extensa en sensores remotos para interpretación y clasificación de imágenes aéreas y por satélite.

TÉCNICAS DE FUSIÓN DE IMAGEN SOBRE DOMINIOS WAVELET

A la hora de realizar la fusión, se puede trabajar sobre tres niveles distintos de abstracción:

- Nivel de píxel (alto).
- Nivel de característica (medio).
- Nivel de decisión (bajo).

El primero es el que proporciona mayor calidad en el resultado de la fusión y los algoritmos de fusión de imagen más significativos pertenecen a dicho nivel de abstracción. Si bien el nivel de píxel requiere que las dos imágenes de partida estén ya registradas.

Los algoritmos de fusión de nivel de característica asumen que se conoce la correspondencia entre características en las imágenes. Hasta ahora se ha realizado un trabajo considerable en correspondencia de características, aunque sin embargo los métodos desarrollados dependen en mayor medida de la aplicación y la imagen. También se han descrito métodos para determinar la correspondencia entre características de punto, segmentos de línea y regiones, así como técnicas para alineamiento espacio-tiempo en secuencias de imagen.

Los algoritmos de fusión de nivel de decisión asumen la correspondencia entre píxeles en las imágenes de entrada. Bajo ciertas condiciones, es posible capturar imágenes registradas. Si los parámetros intrínsecos y extrínsecos a la cámara no se cambian y solo cambian los parámetros del entorno, las imágenes obtenidas estarán registradas espacialmente. Algunos sensores son capaces de capturar imágenes multimodales registradas. Cuando las imágenes son capturadas con cámaras donde los parámetros intrínsecos y/o extrínsecos son diferentes, se necesita de registrar primero las imágenes.

Hasta ahora se ha trabajado bastante en la fusión para nivel de píxel, pero sin embargo no se ha insistido tanto para los otros dos niveles de abstracción.

Para llevar a cabo la fusión, se ha optado por desarrollar diversos algoritmos pertenecientes a la familia de los algoritmos piramidales o algoritmos multirresolución, los cuales forman parte de la categoría de nivel de píxel, y que a partir de las imágenes tomadas como base realizan transformaciones de las mismas a resoluciones más bajas, en lo que se conoce como descomposición piramidal, para su posterior implementación.

Si se modifica el tamaño de la imagen al variar la resolución, se habla de algoritmos multirresolución basados en estructura piramidal; en caso contrario se trata de algoritmos multirresolución con estructura en paralelepípedo, donde en cada etapa de la descomposición o recontrucción de la imagen se conserva su tamaño a pesar de los cambios en la resolución.

Los algoritmos irán así propagando a resoluciones mayores las transformaciones obtenidas tras realizar las operaciones oportunas a las imágenes de resolución menor. Se pueden fusionar imágenes con igual o distinto nivel de resolución, y surgen a raíz de la necesidad de disminuir el coste computacional que requiere el trabajar con imágenes de muy altas resoluciones. Presentan por contra el inconveniente de arrastrar en jerarquía ascendente los errores obtenidos durante las operaciones.

Las técnicas de análisis multirresolución aparecen en los años 80 con idea de introducir una transformada que respondiera a ciertos problemas para los que la transformada de Fourier no resulta una herramienta del todo satisfactoria: **la transformada wavelet.**

Los algoritmos de nivel de píxel pueden trabajar tanto en el dominio wavelet como en el dominio espacial, si bien sobre escenarios de imagen médica sirve de gran ayuda el trabajar con imágenes definidas en el dominio wavelet.

Aunque la fusión de nivel de píxel es una operación local, los algoritmos en el dominio transformado crean la imagen fusionada globalmente. Cambiando un solo coeficiente en la imagen fusionada transformada, todos los valores de la imagen o todo un conjunto cercano en el dominio espacial también cambiarán.

Como resultado, en el proceso de mejora de propiedades en algunas áreas de la imagen, se pueden crean artefactos o efectos no deseados en otras áreas. Los algoritmos que trabajan en el dominio espacial tienen la capacidad de centrarse en las áreas de la imagen deseadas, limitando los cambios en otras áreas sobre las que no se quiere influir.

La descomposición piramidal se llevará a cabo por tanto en dicho dominio transformado, haciendo uso de la transformada wavelet, la cual permite una representación equivalente de la información de una imagen.

Para el caso particular de este proyecto, la herramienta utilizada será la transformada wavelet en dos dimensiones dicreta, DWT, para pasar las imágenes al dominio transformado. A continuación se procederá a realizar las operaciones pertinentes según cada caso de estudio sobre los coeficientes de la transformada. Por último se implementará la transformada wavelet en dos dimensiones discreta inversa, IDWT, para reconstruir dicha imagen al dominio original.

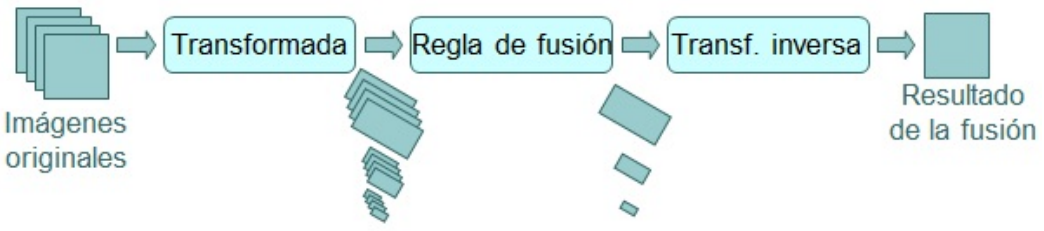


Figura 2.13: Esquema de procedimiento para fusionar.

El enfoque basado en coeficientes wavelet es apropiado para realizar tareas de fusión por las siguientes razones:

1. Es un enfoque multirresolución muy indicado para cuando se trabaja con distintas resoluciones de una misma imagen. En los últimos años, algunos estudios realizados con representaciones multiescala (descomposición piramidal) de una señal han establecido que la información multiescala puede ser muy útil en aplicaciones de procesamiento de imagen, incluyendo la fusión de imagen.
2. La transformada wavelet discreta permite descomponer la imagen en diferentes tipos de coeficientes sin perder la información de la misma.
3. Los coeficientes procedentes de distintas imágenes se pueden combinar para obtener unos nuevos coeficientes de forma que la información de las distintas imágenes de origen queda perfectamente integrada en estos nuevos coeficientes.
4. Una vez se han integrado los coeficientes, se obtiene la imagen final fusionada mediante la transformada wavelet discreta inversa IDWT, manteniendo la información de los coeficientes manipulados.

TRANSFORMADA WAVELET

La herramienta matemática conocida como transformada de Fourier permite transformar señales definidas en el dominio del tiempo al dominio de la frecuencia. El problema radica en la imposibilidad de determinar ambas variables simultáneamente según lo enunciado en el principio de incertidumbre de Heisemberg, que alude a la imposibilidad de conocer de forma simultánea un par de magnitudes físicas relativas a un mismo fenómeno.

La transformada wavelet permite hacer análisis en el conjunto tiempo-frecuencia, descomponiendo así la información contenida en la imagen para los dominios frecuencial y espacial.

Transformada wavelet continua

Una función wavelet hace referencia a una señal del mismo nombre de duración limitada cuyo valor medio es cero. Se representa por ψ, y está contenida en el espacio de señales de energía finita $L^2(\Re)$ y centrada en $t = 0$ mediante la imposición de la condición

$$\int_{-\infty}^{\infty} \psi(t)dt = 0$$

con valor medio nulo.

Aplicando unas operaciones de desplazamiento temporal en $t = b$ y cambio frecuencial por un factor de escalado a, la expresión de la función wavelet queda expresada así:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a})$$

A partir de ella, y obviando el desglose de su desarrollo por razones de extensión, se define la transformada wavelet continua asociada a f como

$$W(f)(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(t)\psi(\frac{t-b}{a})dt$$

así como la transformada wavelet inversa continua como

$$f(t) = \frac{1}{C_\psi}\frac{1}{a^2}\int_0^\infty \int_{-\infty}^{\infty} W(f)(a,b)\frac{1}{\sqrt{|a|}}\psi(\frac{t-b}{a})dadb$$

siendo

$$C_\psi = \int_0^\infty \frac{|\widehat{\psi}(W)|^2}{W}dW < +\infty$$

con $\psi, f \epsilon L^2(\Re)$.

La función wavelet madre es aquella particularizada para el caso $a = 1, b = 0$.

Transformada wavelet discreta

La transformada wavelet discreta se obtiene mediante un proceso de discretización basado en el muestreo de los dos parámetros reales a y b presentes para el caso continuo. Así, en la transformada wavelet discreta se elimina la representación bidimensional de una señal que es de naturaleza unidimensional, tal y como ocurre para el caso continuo. Desaparece de este modo información que resultaba redundante. Tomando como base la función wavelet particularizada para el caso a = 1 y b = 0, se realiza una descomposición en series de funciones

$$f(x) = \sum_k \alpha_k \varphi_k(x)$$

tomando para ello los parámetros a y b valores discretos, tales como $a = 2^{-j}$ y $b = 2^{-j}k$, siendo j el escalado y k el desplazamiento. Se toman potencias de dos por motivos de ahorro computacional.

Se pasa así de un sistema de la forma

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a})$$

a uno de la forma

$$\psi j, k(t) = \frac{1}{\sqrt{2^{-j}}}\psi(\frac{t-2^{-j}k}{2^{-j}})$$

Realizando una expansión en series de funciones wavelets

$$f(x) = \sum_k C_{j_0}(k)\varphi_{j_0,k}(x) + \sum_{j=j_0}^{\infty}\sum_k d_j(k)\psi_{j,k}(x)$$

se obtienen los coeficientes de aproximación

$$c_{j_0}(k) = \langle f(x), \varphi_{j_0,k}(x) \rangle = \int f(x)\varphi_{j_0,k}(x)dx$$

y los coeficientes de detalle

$$d_j(k) = \langle f(x), \psi_{j,k}(x) \rangle = \int f(x)\psi_{j,k}(x)dx$$

La función wavelet más sencilla es la wavelet de Haar

$$\psi_{j,k}(t) = 2^{\frac{j}{2}}\psi(2^j t - k)$$

Se pretende ahora encontrar las funciones ψ que conduzcan a sistemas ortonormales y completos en el espacio $L^2(\Re)$. Para ese caso, se tiene $f \epsilon L^2(\Re)$, tal que

$$f = \sum_j \sum_k \langle f, \psi_{j,k} \rangle \psi_{j,k}$$

La transformada wavelet discreta unidimensional se define mediante la expresión

$$\langle f, \psi_{j,k} \rangle = 2^{\frac{j}{2}} \int_{-\infty}^{\infty} f(t)\psi(2^j t - k)dt$$

Transformada wavelet bidimensional discreta

La descomposición wavelet multirresolución descrita en la transformada wavelet unidimensional puede extenderse para el caso de la transformada bidimensional, introduciendo para ello un escalado 2-D y haciendo uso de las funciones wavelet descritas para el caso de una única dimension. Ésto permite cierto ahorro de coste computacional.

La transformada wavelet bidimensional discreta de una imagen f(x,y) se calcula mediante

$$W_\varphi(j_0, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)\varphi_{j_0,m,n}(x, y)$$

$$W_\psi^i(j, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)\psi_{j,m,n}^i(x, y)$$

Transformada wavelet bidimensional discreta inversa

$$f(x, y) = \frac{1}{\sqrt{MN}} \sum_m \sum_n W_\varphi(j_0, m, n)\varphi_{j_0,m,n}(x, y) + \frac{1}{\sqrt{MN}} \sum_{i=H,V,D} \sum_{j=j_0}^{\infty} \sum_m \sum_n W_\psi^i(j, m, n)\psi_{j,m,n}^i(x, y)$$

Traduciendo todo lo anterior a lenguaje de teoría de la señal, se podría decir que la descomposición wavelet es un proceso en el cual a una imagen $X(m, n)$, donde m es el número de filas y n el número de columnas, se le aplican sendos filtrados paso bajo y paso alto y tras ellos una reducción de muestras o down-sampling a través del recorrido de las filas de la matriz de la imagen donde se elima una de

cada dos muestras, y a su vez cada uno de los dos resultados obtenidos tras el filtrado y el down-sampling se filtran con un filtro paso bajo por un lado y paso alto por otro, para por último realizar un nuevo eliminado de muestras, en este caso eliminando una de cada dos muestras durante el recorrido de las columnas de la matriz.

Se obtienen así por tanto a partir de la matriz de la imagen cuatro submatrices $a(m,n)$, $d^V(m,n)$, $d^H(m,n)$ y $d^D(m,n)$ correspondientes a la matriz de bajas frecuencias o matriz de aproximación, y las matrices de altas frecuencias o matrices de detalles verticales, horizontales y diagonales. El tamaño de todas ellas se habrá reducido a la mitad respecto de la imagen original, lo que a efectos prácticos se corresponde con un diezmado por un factor $M = 2$. Gráficamente se puede ver lo anteriormente descrito de una forma más clara:

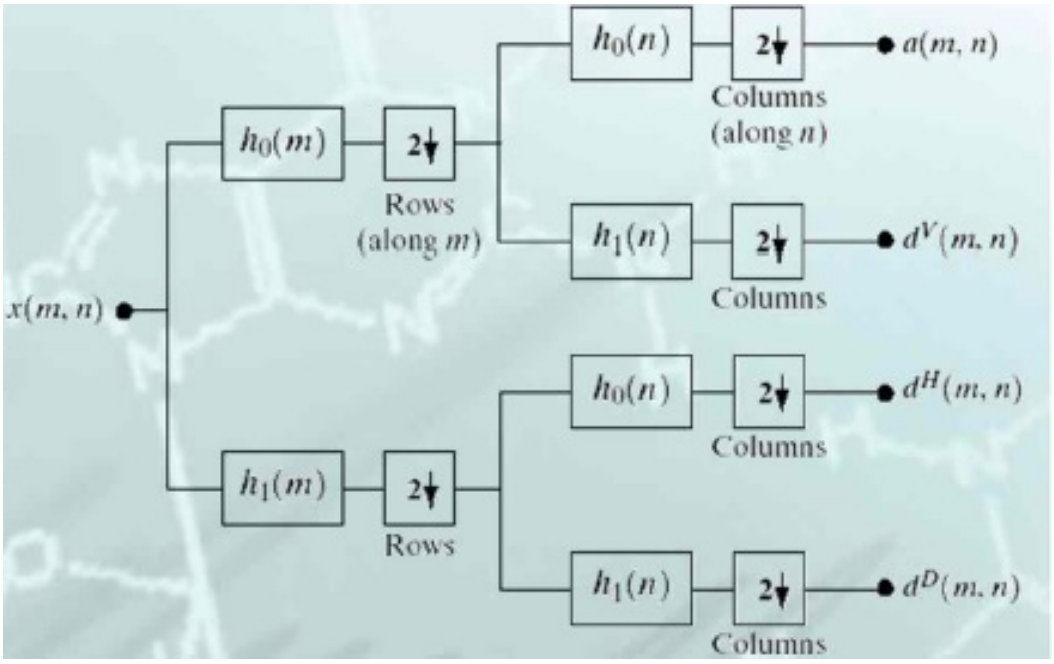
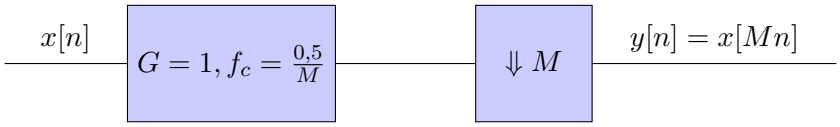


Figura 2.14: Gráfica del proceso de descomposición de la imagen.

DIEZMADO

Durante la explicación de la descomposición wavelet se ha hablado de un diezmado por un factor de $M = 2$. El diezmado es el proceso en el cual un sistema discreto aumenta el período de muestreo (o lo que es lo mismo, disminuye la frecuencia de muestreo) de una señal por un factor entero M.

Dicho sistema discreto se conoce con el nombre de *compresor*, y para realizar dicha compresión lo que hace es de cada M muestras que entran al sistema sale una. Si al sistema compresor se le añade previamente un filtro paso bajo (de ganancia $G = 1$ y frecuencia de corte $f_c = \frac{0,5}{M}$) con el objetivo de que no haya solapamieno espectral, al conjunto se le conoce como *diezmador*.

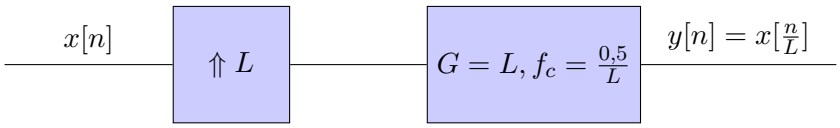


INTERPOLACIÓN

Por otro lado, al proceder a fusionar las imágenes ha aparecido en según qué tipo de fusión el problema de tener dos imágenes a fusionar de distinto tamaño. Para solventar ésto último se optó por aumentar una de las imágenes mediante una interpolación.

En la práctica la interpolación se ha llevado a cabo mediante interpolación lineal a través de la función de MATLAB *interp*2. Otros tipos de interpolación aumentaban el coste computacional sin obtener beneficio.

La interpolación es el proceso en el que se reduce el período de muestreo (lo que equivale a aumentar la frecuencia de muestreo) por un factor entero L mediante un sistema discreto compuesto de un bloque *expansor* que inserta $L-1$ ceros entre cada dos muestras de la señal de entrada, y de un filtro paso bajo de ganancia $G = L$ y frecuencia de corte $f_c = \frac{0,5}{L}$.



Capítulo 3

Algoritmos de fusión de imagen

En este capítulo se procede a desglosar cómo son los distintos algoritmos de fusión de imagen que se han ido extrayendo de las literaturas analizadas concernientes al tema a tratar, analizando su base teórica y procediendo a su implementación en el entorno MATLAB.

NOTA: Los códigos MATLAB de programación de los algoritmos se encuentran todos al completo al final de este documento en el Anexo.

Este trabajo se ha centrado en reglas de fusión pertenecientes a dos familias distintas:

- Reglas basadas en la combinación de coeficientes.
  - Media aritmética.
  - Media ponderada gaussiana.
  - Media ponderada laplaciana.
  - Media de imágenes multirresolución.

- Reglas basadas en la selección de coeficientes.
  - Varianza sobre una ventana de coeficientes.
  - Coeficiente de variación sobre una ventana de coeficientes.
  - Medición del nivel de actividad basado en coeficientes.
  - Medición del nivel de actividad basado en ventana.

Si se realiza una combinación de coeficientes, ésta ha de hacerse de la forma más apropiada para obtener la mejor calidad de imagen posible en el resultado de la fusión. La regla más simple consistiría en obtener la media de los coeficientes de las dos imágenes en una misma posición en la matriz. Promediando, se preserva la presencia de las características de cada imagen de entrada en la imagen fusionada; sin embargo, el contraste de las características podría verse significantemente reducido, especialmente para los detalles. Existen otras reglas más potentes que parecen ofrecen mejores resultados.

Si se opta por reglas de selección de coeficientes, éstas se pueden aplicar a toda la matriz o a una pequeña ventana de coeficientes. Una vez comparadas las matrices o las ventanas, según el caso, de ambas imágenes, se selecciona el coeficiente en cuestión de la imagen que más interese, según lo descrito por la regla de fusión que se esté aplicando.

Así pues, salvo el primer algoritmo de la media aritmética que se calcula sobre el dominio espacial, el resto sigue el esquema ya comentado en el capítulo anterior, en el que las imágenes de partida se descomponen mediante la transformada wavelet (ver Figura 3.1), la cual se aplicará una o más veces según el nivel de descomposición al que se pretenda llegar antes de proceder a la fusión (en este caso se ha llegado como máximo hasta el tercer nivel), y una vez obtenidas las matrices de aproximación y de detalles horizontales, verticales y diagonales de cada imagen, se procederá siempre a calcular la media aritmética de las matrices de aproximación de cada imagen, mientras que para las matrices de detalles se procederá según dicte cada esquema de fusión (ver Figura 3.2).

Figura 3.1: Proceso de descomposición de la imagen. Aquí se puede ver cómo llega hasta el segundo nivel de descomposición.

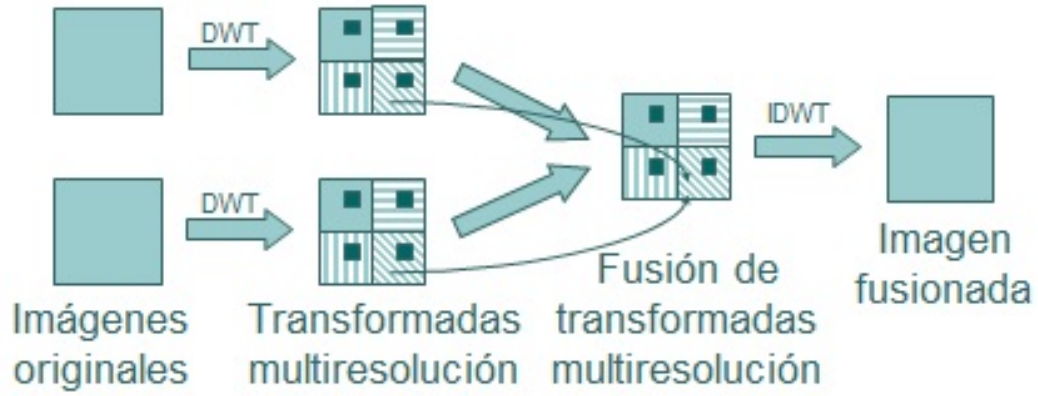


Figura 3.2: Esquema básico de fusión.

Mención especial merece el esquema seguido por el algoritmo *media aritmética de imágenes multirresolución*, en el que únicamente se descompone una de las imágenes originales, para proceder al cálculo de la media entre la matriz de aproximación de la imagen que se ha descompuesto en el nivel deseado y la matriz original de niveles de gris de la otra imagen:

Una vez dicho ésto, y trabajando ya dentro del entorno MATLAB, se pasa a explicar qué pasos previos se han llevado a cabo en todos los casos para una correcta

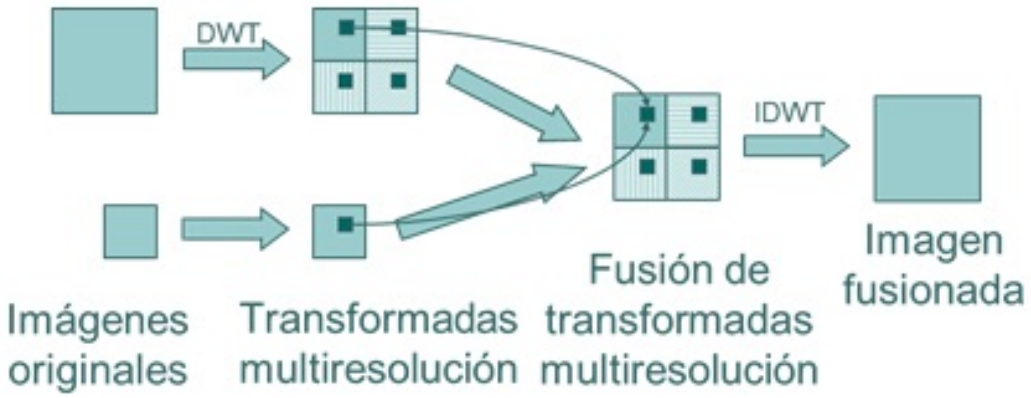


Figura 3.3: Esquema de fusión con DWT sólo sobre una imagen.

implementación del mecanismo elegido. La nomenclatura utilizada en MATLAB común a todos los algoritmos ha sido la siguiente:

- $X(m,n)$ e $Y(p,q)$ para las imágenes de partida.
- $Z(r,s)$ para la imagen fusionada.

En primer lugar se procede a leer las imágenes de entrada haciendo uso de la función *dicomread*, para a continuación convertir los coeficientes de la imagen al tipo de dato que se requiere (*double* en este caso) y normalizar toda la matriz, pasando todos los valores a positivo y acotándolos entre 0 y 255.

```
X=dicomread(fullfile(get(handles.textoInfotac,'UserData'),get(handles.editInfotac,'UserData')));
Y=dicomread(fullfile(get(handles.textoInfopet,'UserData'),get(handles.editInfopet,'UserData')));

X=double(X);
X=X-min(X(:));
X=X/max(X(:))*255;

Y=double(Y);
Y=Y-min(Y(:));
Y=Y/max(Y(:))*255;
```

Figura 3.4: Código MATLAB para la lectura de las imágenes.

Lo siguiente es comprobar si ambas imágenes no son del mismo tamaño, para en caso afirmativo aumentar el tamaño de la más pequeña interpolando por un factor L. Se fuerza además a que las matrices sean cuadradas en caso de que no lo fueran.

```
%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

end

%Se comprueba si las matrices de las imágenes a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end
```

Figura 3.5: Código MATLAB de comprobación del tamaño de las matrices.

Una vez superados estos pasos previos, ya se dispone de las matrices de las imágenes sobre las que se va a proceder con el tamaño adecuado y dentro del rango de valores deseado.

Lo siguiente sería descomponer las imágenes mediante la transformada wavelet discreta para obtener de cada una de ellas las submatrices de aproximación y de detalles horizontales, verticales y diagonales.

```
%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a la
%matriz de coeficientes de aproximación, y matrices de coeficientes de
%detalles horizontales, verticales y diagonales.

[xA,xH,xV,xD]=dwt2(X,'haar');

[yA,yH,yV,yD]=dwt2(Y,'haar');
```

Figura 3.6: Descomposición de las imágenes mediante la DWT2.

Una vez se llega aquí, ya se pueden aplicar los algoritmos de fusión. Por último, a partir de la variable que contiene la matriz resultante de las operaciones realizadas por el mecanismo de fusión, se genera una nueva imagen DICOM con la función

dicomwrite, que será la imagen final resultante.

3.1. Algoritmos en el dominio espacial

3.1.1. *Media aritmética*

Como ya se ha comentado anteriormente, éste es el único algoritmo en el que no se ha procedido a descomponer la imagen, y la media se ha calculado directamente sobre los niveles de gris. Se calcula promediando los coeficientes de las imágenes en posiciones correspondientes.

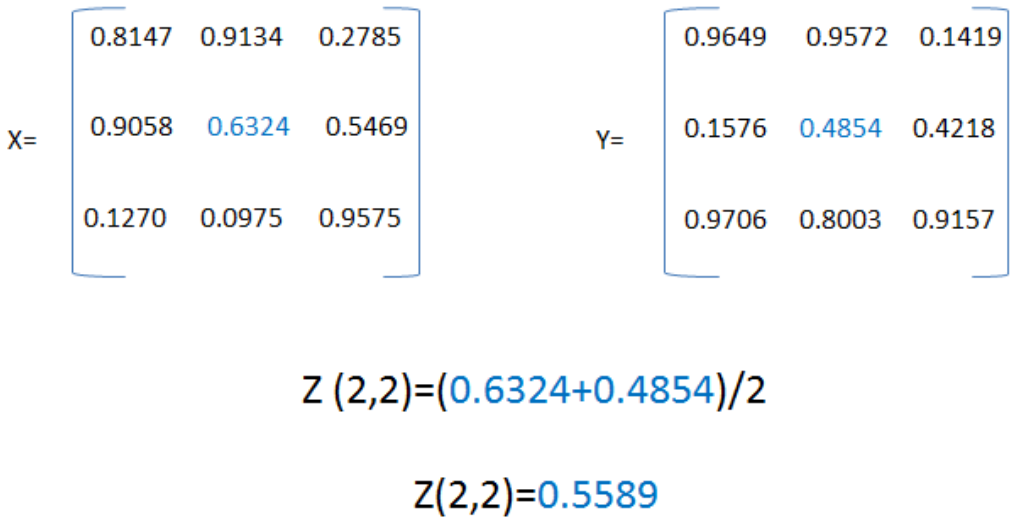
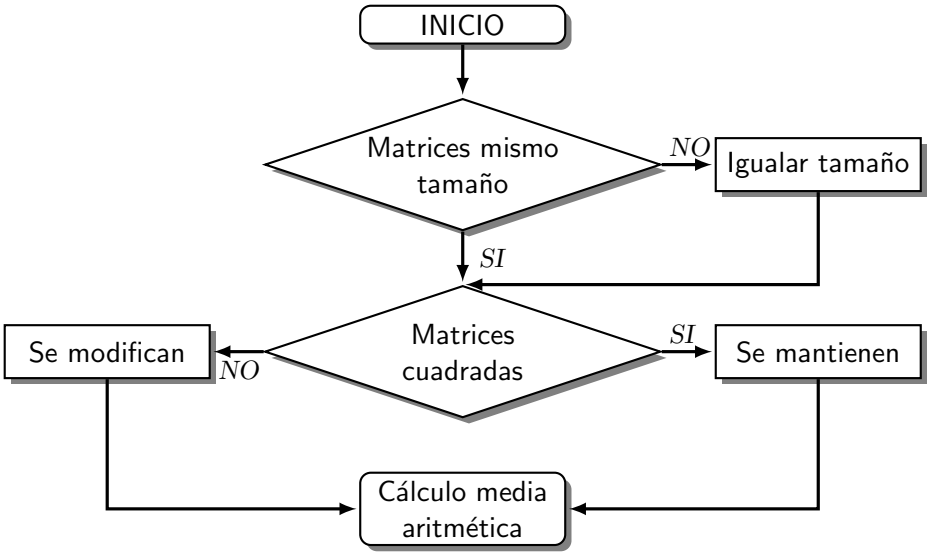


Figura 3.7: Ejemplo de cálculo del coeficiente de la posición (2,2) en la imagen fusionada Z, a partir de X e Y.

FLUJOGRAMA



3.2. Algoritmos sobre dominios wavelet

3.2.1. *Media aritmética*

En este caso se procederá de igual forma que en el caso del apartado 3.1.1, con la salvedad de que la media se calcula una vez se ha realizado la descomposición de las imágenes, promediando las submatrices de aproximación y de detalles de cada imagen.

Ejemplo de resultado de fusión con este método:

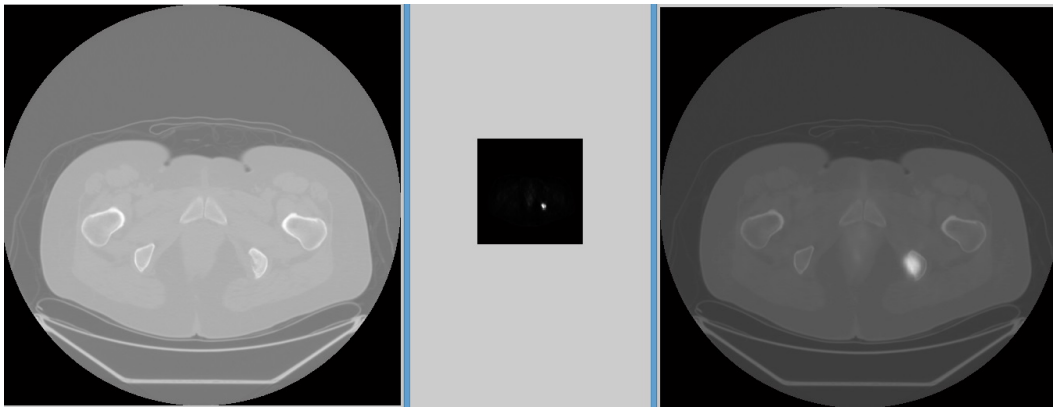
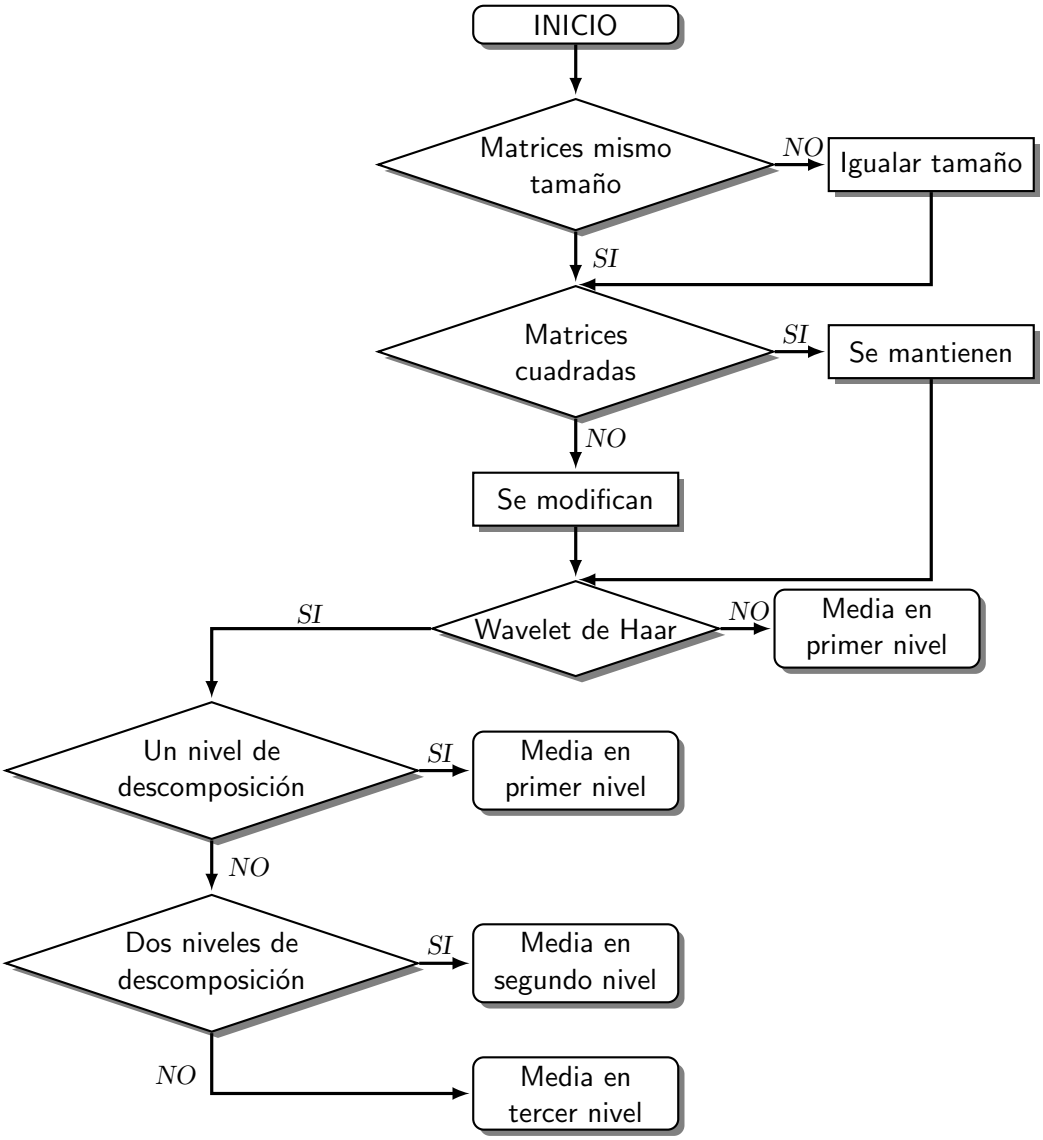
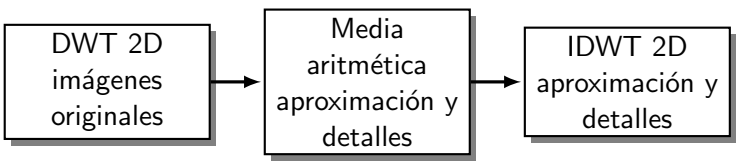


Figura 3.8: Fusión mediante media aritmética.

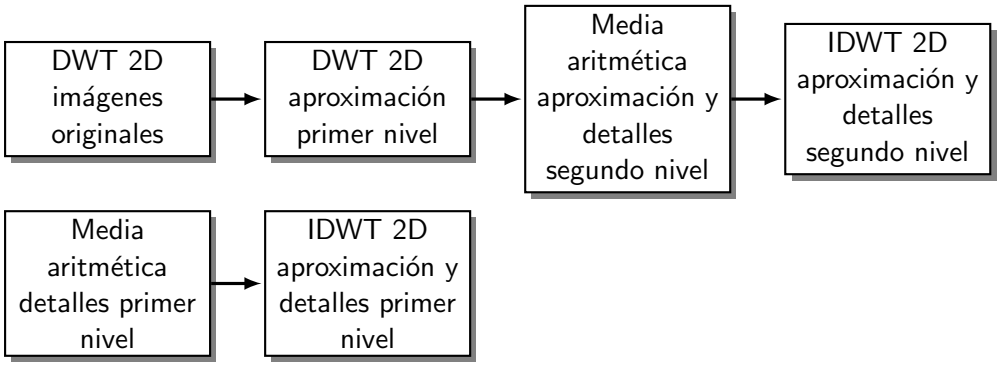
FLUJOGRAMA



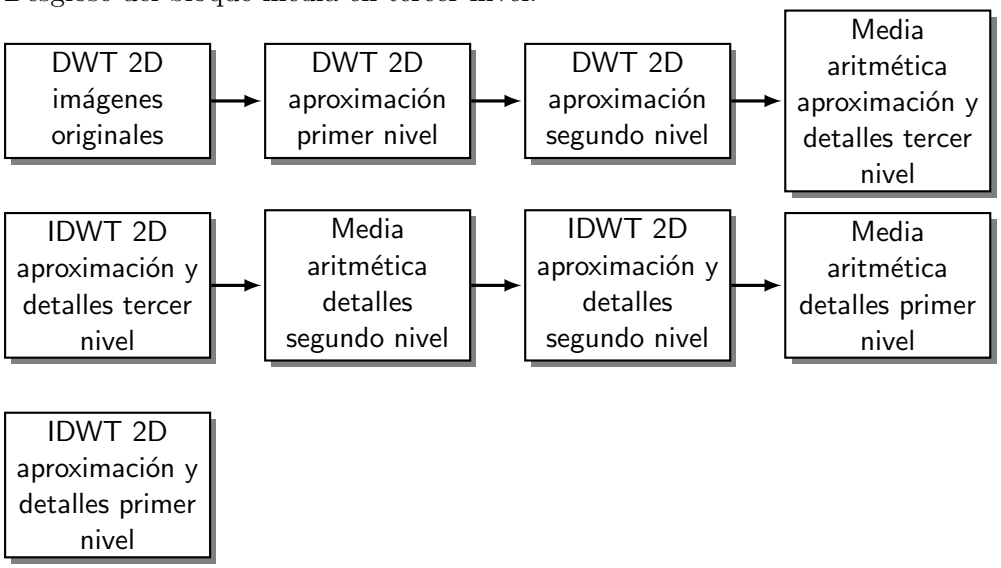
Desglose del bloque *media en primer nivel:*



Desglose del bloque *media en segundo nivel:*



Desglose del bloque *media en tercer nivel:*



3.2.2. *Media ponderada gaussiana*

Este mecanismo de fusión calcula la media aritmética de las submatrices de aproximación, mientras que para las submatrices de detalles realiza una ponderación mediante una matriz o máscara de pesos gaussiana W previa al promediado. Dicha matriz de pesos tendrá un tamaño S a elegir entre 3, 5, 11 y 21, y se caracteriza por dar mayor ponderación al píxel central de la ventana, actuando como filtro paso bajo. La matriz de pesos se genera mediante

```
%Se genera una matriz gaussiana del mismo tamaño de la ventana que
%va a ir rodeando a cada coeficiente durante el recorrido de la matriz.
W=gausswin(S)*gausswin(S)';
```

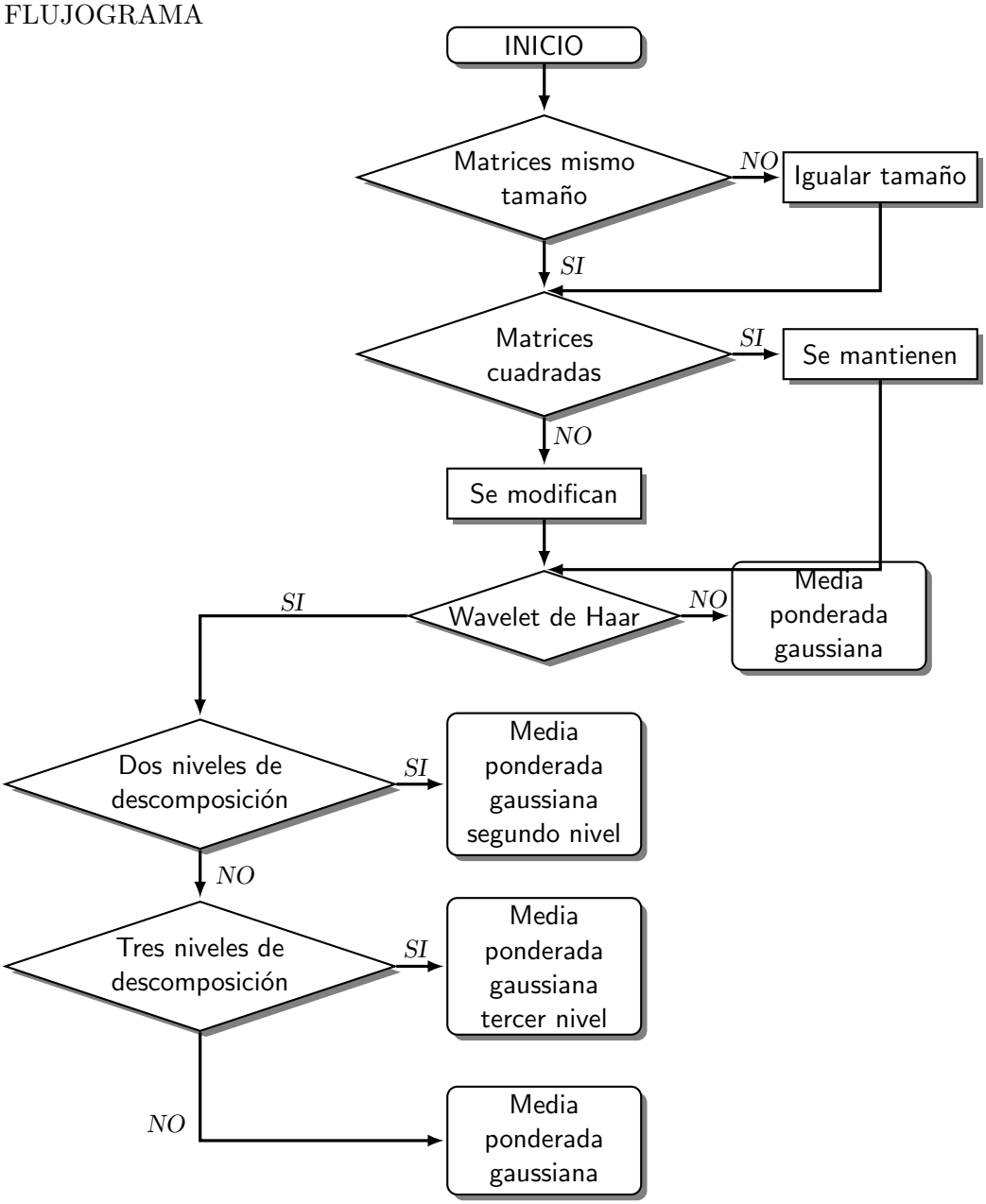
Figura 3.9: Creación de una matriz de pesos gaussiana.

lo que da como resultado para un caso práctico de tamaño de ventana $S = 3$ la siguiente matriz:

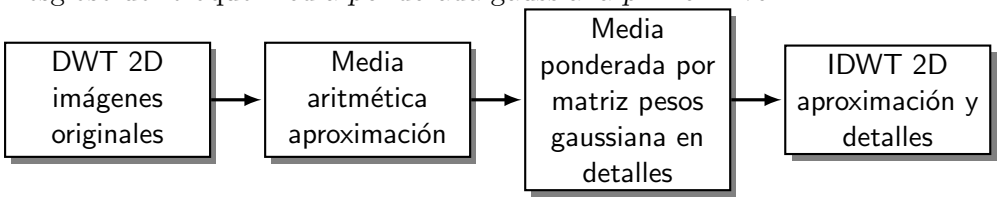
$$W = \begin{pmatrix} 0.0019 & 0.0439 & 0.0019 \\ 0.0439 & 1.0000 & 0.0439 \\ 0.0019 & 0.0439 & 0.0019 \end{pmatrix}$$

Con esa matriz W se ponderan y promedian las ventanas de tamaño S que rodean al coeficiente (m, n) de interés en cada una de las submatrices de detalles, quedando una única matriz de tamaño S. En esta matriz todos sus coeficientes se dividen por la suma de todos los coeficientes de la matriz gaussiana, y los coeficientes que resultan se suman todos para obtener un único coeficiente que será el que ocupe la posición (m, n) en la imagen fusionada.

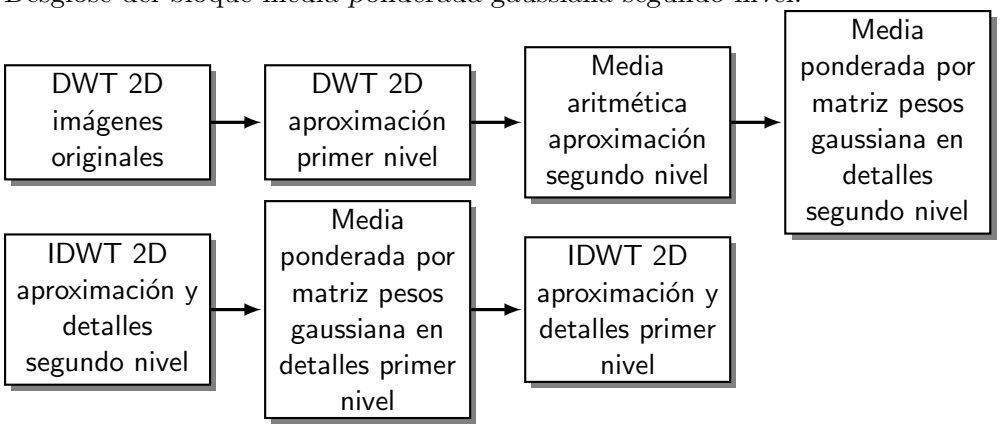
FLUJOGRAMA



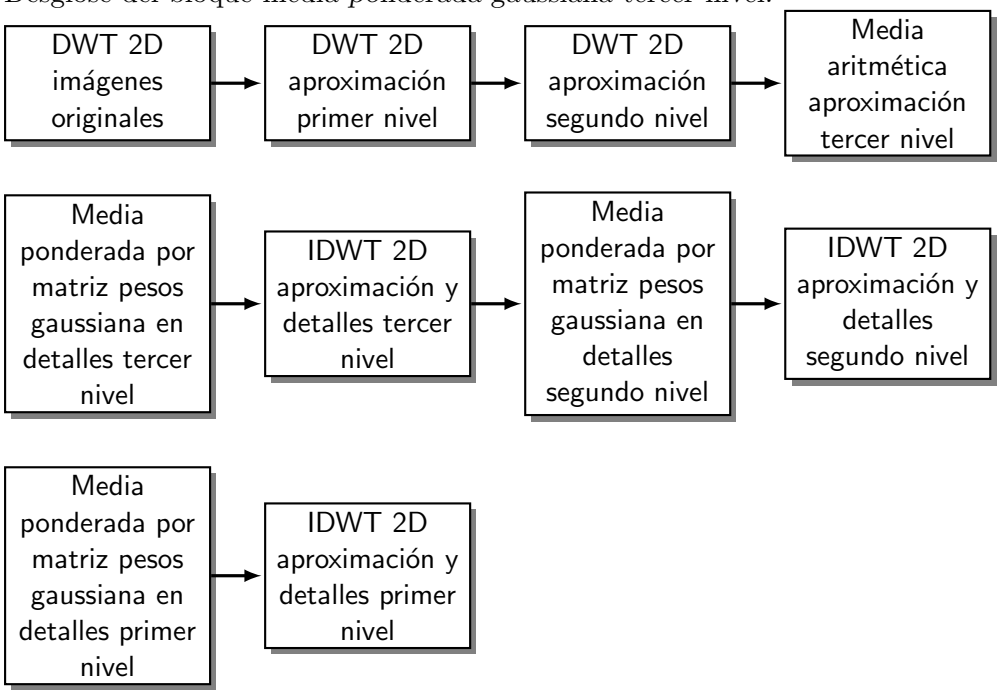
Desglose del bloque *media ponderada gaussiana primer nivel:*



Desglose del bloque *media ponderada gaussiana segundo nivel:*



Desglose del bloque *media ponderada gaussiana tercer nivel:*



3.2.3. *Media ponderada laplaciana*

Este algoritmo calculará la media aritmética de las submatrices de aproximación, mientras que para las submatrices de detalles realiza una ponderación mediante una matriz o máscara de pesos laplaciana L previa al promediado. Dicha matriz de pesos tendrá un tamaño de $3x3$, y se caracteriza por dar una ponderación mucho mayor al píxel central de la ventana, mientras que la suma de todos los coeficientes de dicha máscara suma cero. Actúa como filtro paso alto.

La matriz L es de la forma

$$L = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

La suma de los coeficientes de la matriz obtenida tras la ponderación y el promediado será el coeficiente que irá en la posición (m,n) en la imagen final fusionada.

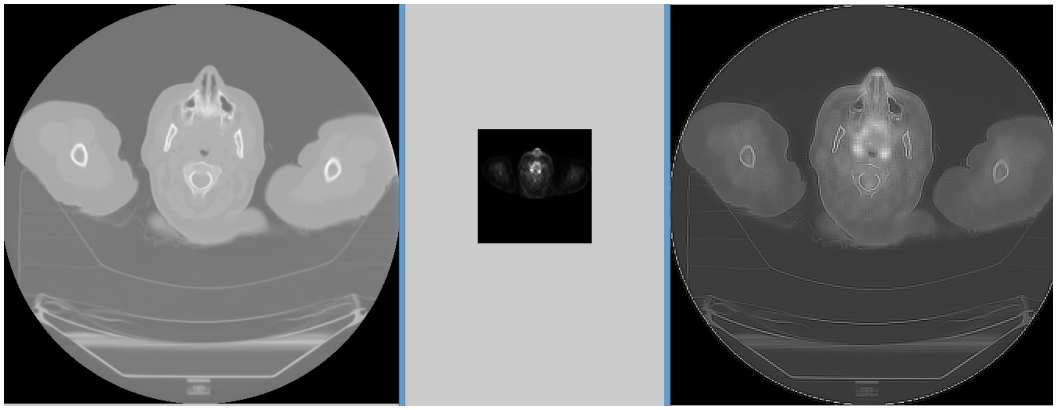
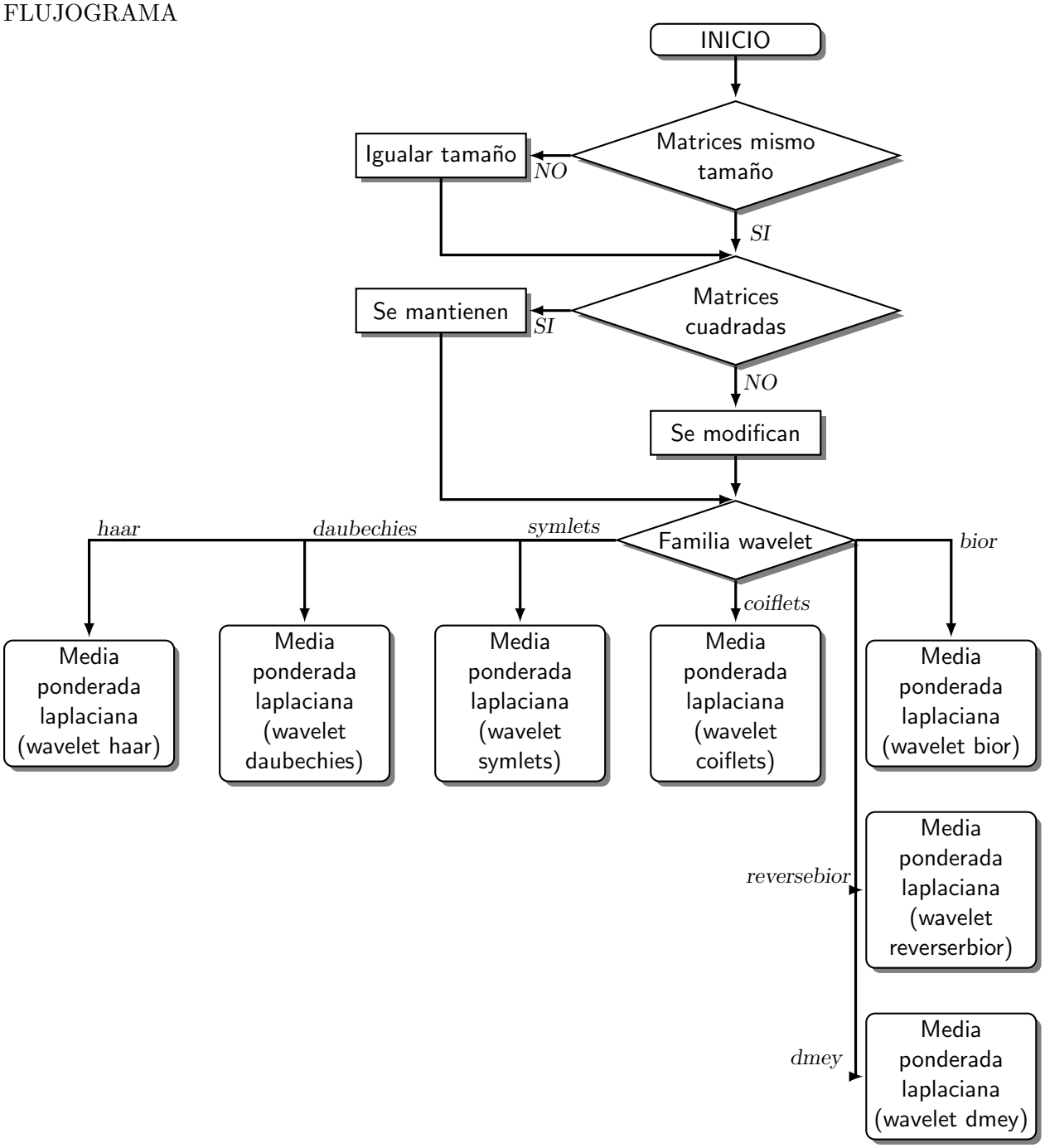
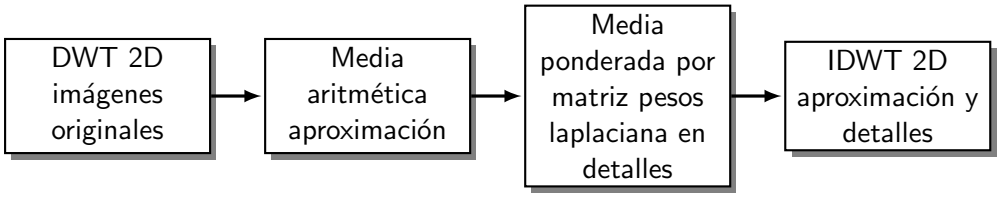


Figura 3.10: Fusión mediante media ponderada laplaciana.

FLUJOGRAMA



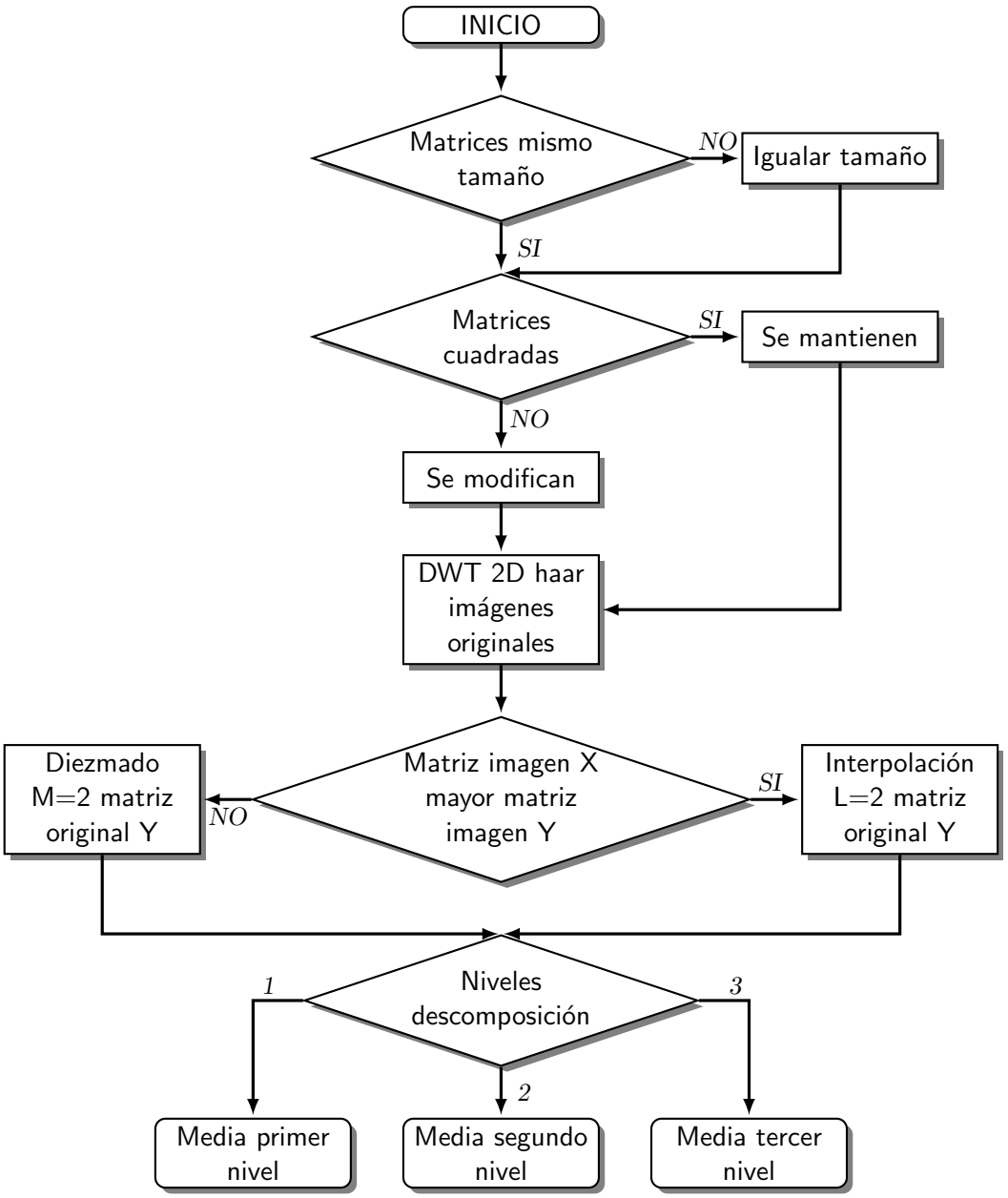
Desglose del bloque *media ponderada laplaciana (para cualquiera de las wavelet):*



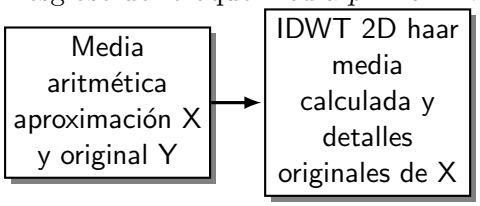
3.2.4. *Media aritmética de imágenes multirresolución*

Este algoritmo calcula la media aritmética entre la submatriz de aproximación obtenida al descomponer una de las imágenes y la matriz original de la otra imagen en el dominio espacial. El único inconveniente encontrado para esta implementación es el hecho de que una de las matrices a fusionar está formada por coeficientes wavelet mientras los coeficientes de la otra son directamente los niveles de intensidad en la escala de grises, luego sus rangos de valores difieren. Para ello, se ha tenido que "forzar" a que los rangos de valores de los coeficientes sean similares.

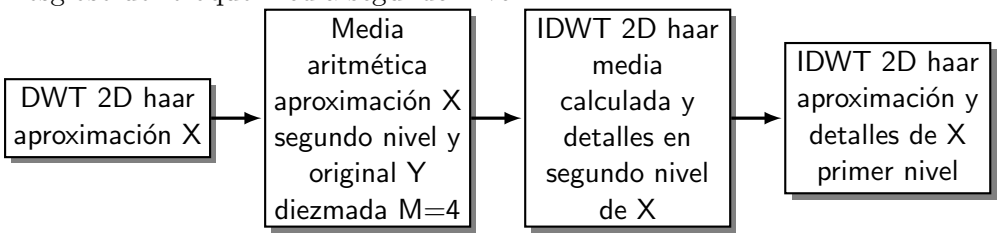
FLUJOGRAMA



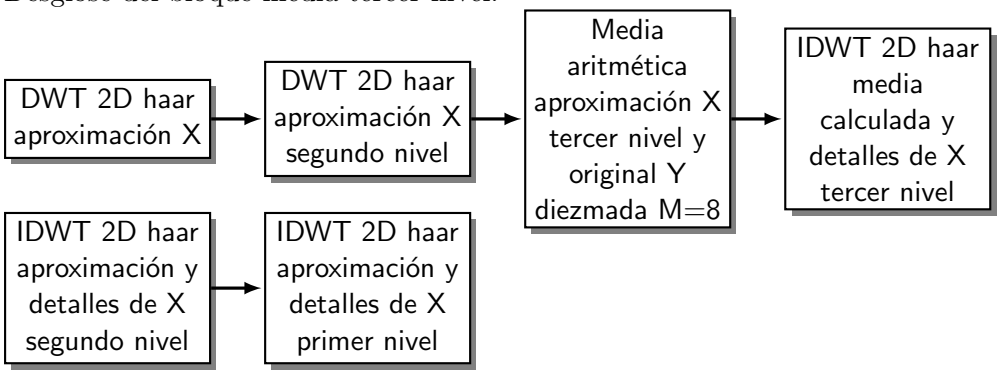
Desglose del bloque *media primer nivel:*



Desglose del bloque *media segundo nivel:*



Desglose del bloque *media tercer nivel:*



3.2.5. *Varianza de una ventana de coeficientes*

Mientras que la media aritmética es una medida de centralización, la varianza es una medida de dispersión que se define como

$$Var(X) = E[(X - \bar{X})^2]$$

Como su nombre hace ver, este algoritmo se basa en una ventana de coeficientes de tamaño S que puede tomar valores $3, 5, 11, 21$, la cual va recorriendo las respectivas submatrices de detalles, y sobre la que se calculará la varianza en ambos casos. A continuación, se comparan los valores de ambas varianzas, y si su diferencia queda por debajo del valor de un umbral U a elegir entre $1, 5, 15, 30$, se calculará la media aritmética entre los coeficientes correspondientes al píxel central de cada ventana. En caso que la diferencia entre varianzas quede por encima del umbral, se seleccionará el coeficiente del píxel central de la ventana con mayor varianza.

Para un caso práctico, se tienen dos ventanas a y b que rodean a sendos píxeles en la posición central (m, n):

$$a = \begin{pmatrix} -0.3034 & 0.8884 & -0.8095 \\ 0.2939 & -1.1471 & -2.9443 \\ -0.7873 & -1.0689 & 1.4384 \end{pmatrix} b = \begin{pmatrix} 0.3252 & -1.7115 & 0.3192 \\ -0.7549 & -0.1022 & 0.3129 \\ 1.3703 & -0.2414 & -0.8649 \end{pmatrix}$$

Se calculan las varianzas de ambas ventanas, que resultan ser
$var(a) = 1.6561$ $var(b) = 0.7872$

Si la diferencia entre ellas, $|1.6561 - 0.7872| = 0.8689$ es menor que el umbral fijado de los disponibles, en la matriz fusionada resultante irá en la posición central (m, n) el promedio de los coeficientes $a(m, n)$ y $b(m, n)$. Si por el contrario esa diferencia es mayor que el umbral, en la posición central (m, n) de la matriz final irá el coeficiente de esa posición de la ventana con mayor varianza.

En este caso la varianza de la ventana a es mayor, luego se elige el coeficiente $a(m, n) = -1.1471$.

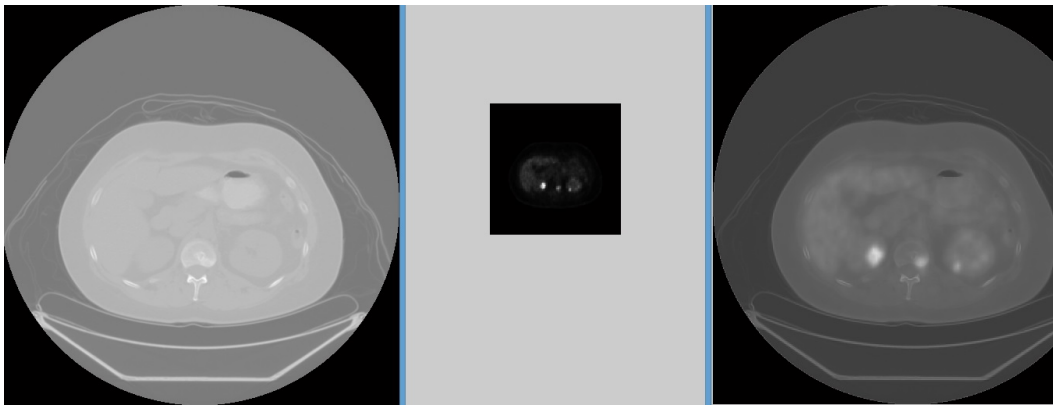
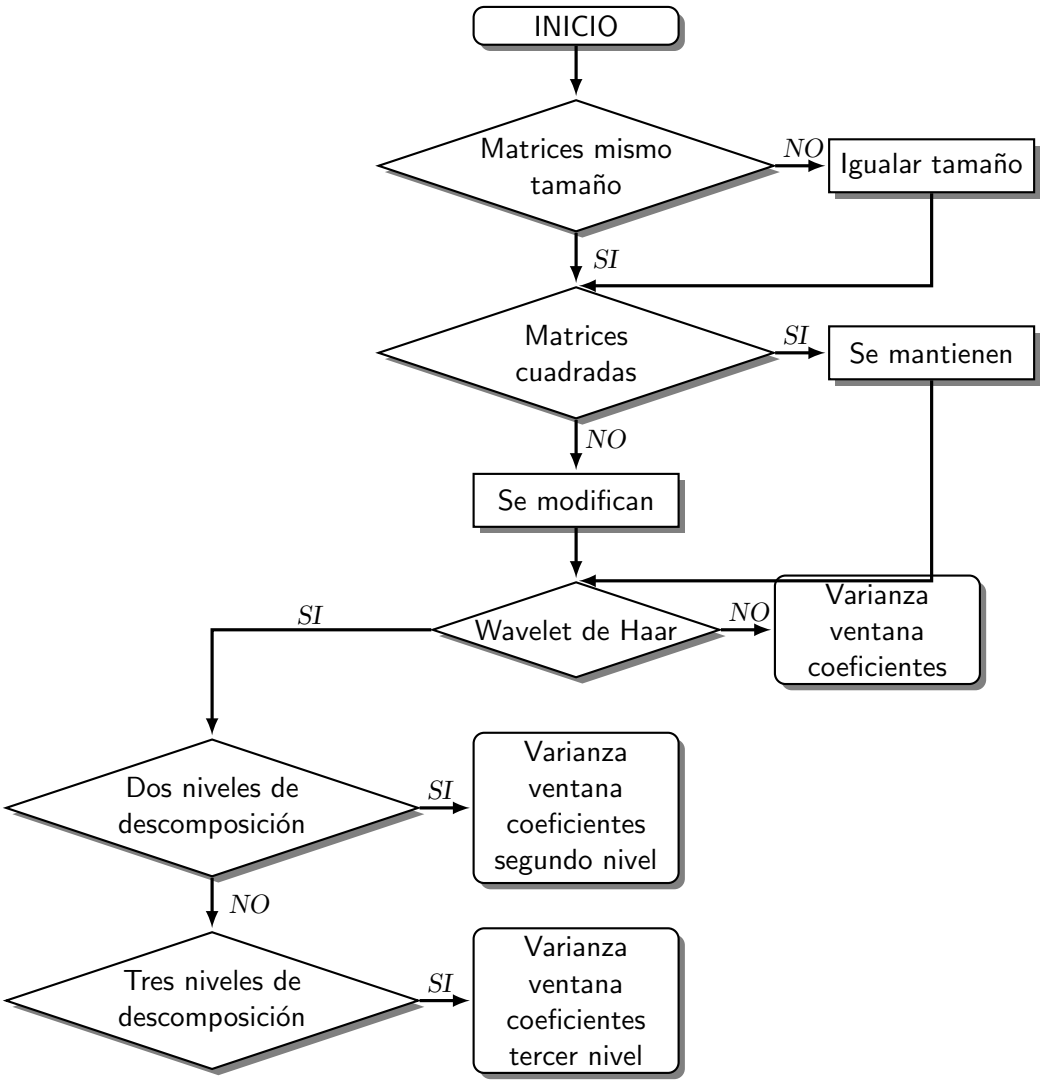
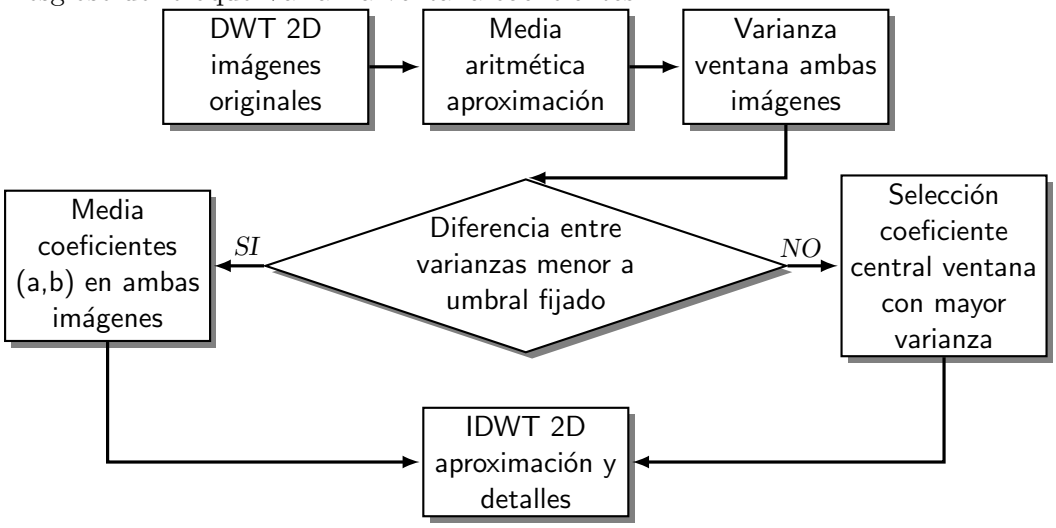


Figura 3.11: Fusión mediante varianza sobre ventana de coeficientes.

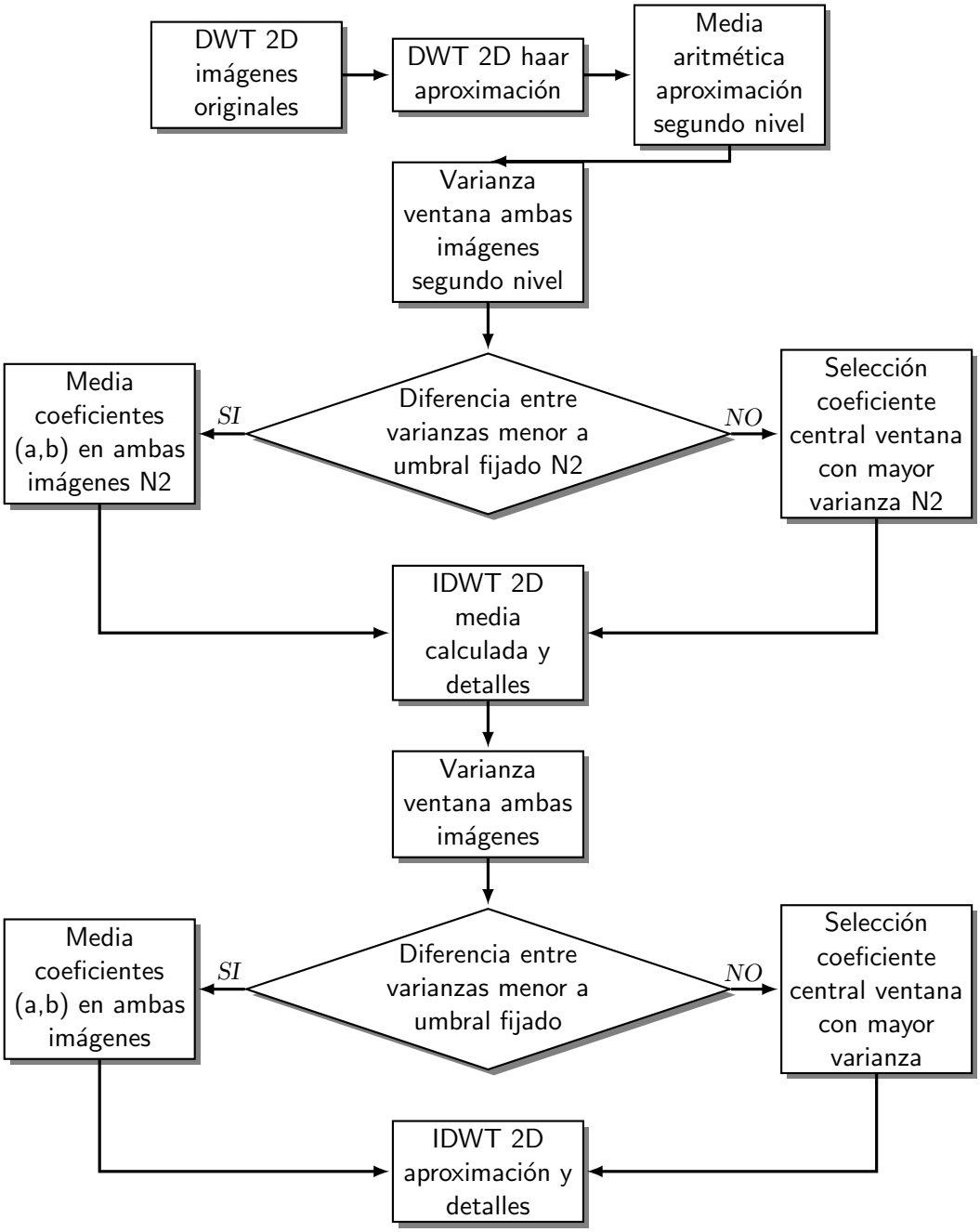
FLUJOGRAMA



Desglose del bloque *Varianza ventana coeficientes:*



Desglose del bloque *Varianza ventana coeficientes segundo nivel:*



3.2.6. *Coeficiente de variación de una ventana de coeficientes*

Para este algoritmo se procederá de idéntica forma que para el caso del cálculo de la varianza, salvo que en este caso en lugar de la varianza se calculará sobre cada ventana lo que se conoce como coeficiente de variación, el cual es un parámetro estadístico resultante del cociente entre la desviación típica y la media de la muestra a estudiar.

$$C_v = \frac{\sigma}{|\bar{X}|}$$

Es un parámetro que mide lo dispersos que están los datos dentro de una muestra dada. En este caso, al calcular el coeficiente de variación dentro de una ventana de coeficientes, si da un valor alto significa que hay mucha variabilidad entre los valores de los coeficientes, mientras que si resulta un valor pequeño los coeficientes son más homogéneos.

A la hora de realizar la selección de coeficientes para la imagen fusionada, se siguen los pasos del algoritmo anterior de la varianza, sólo que en este caso lo que se comparan son las diferencias entre coeficientes de variación.

Ejemplo de fusión empleando este algoritmo:

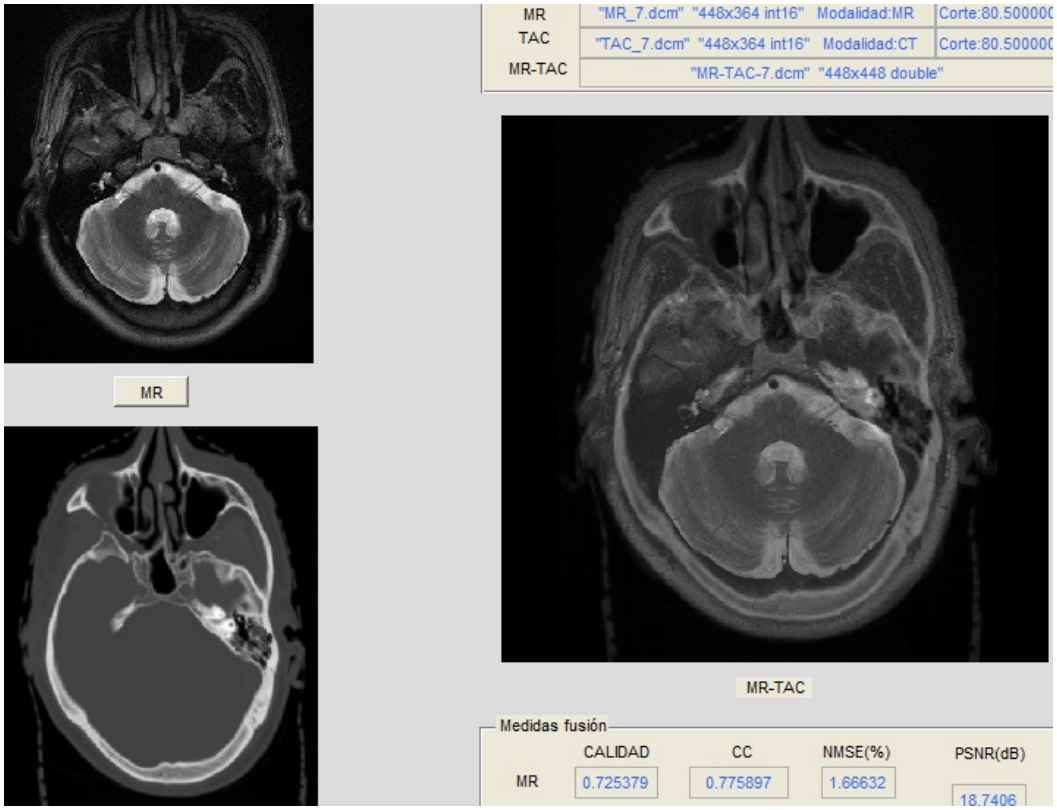


Figura 3.12: Fusión mediante coeficiente de variación sobre ventana de coeficientes.

El flujograma de la varianza es perfectamente aplicable a este caso, con la salvedad de los bloques en los que se calcula la varianza. Para este algoritmo, en esos bloques se calcularía el coeficiente de variación. Todo lo demás es exactamente igual.

3.2.7. *Medición del nivel de actividad*

El nivel de actividad de un coeficiente refleja la energía local en el espacio delimitado por dicho coeficiente y viene descrito por el valor absoluto o el cuadrado del coeficiente (refiriéndose siempre a coeficientes wavelet). Este algoritmo viene a comparar los niveles de actividad entre coeficientes (compara los coeficientes en valor absoluto) en posiciones correspondientes en las submatrices de detalles, seleccionando el de mayor valor y colocándolo en la posición correspondiente en la matriz resultante con su signo original. Para las submatrices de aproximación se calcula la media.

Como ejemplo ilustrativo se tienen dos matrices X e Y que tienen valores positivos y negativos:

$$X = \begin{pmatrix} -0.3034 & 0.8884 & -0.8095 \\ 0.2939 & -1.1471 & -2.9443 \\ -0.7873 & -1.0689 & 1.4384 \end{pmatrix} Y = \begin{pmatrix} 0.3252 & -1.7115 & 0.3192 \\ -0.7549 & -0.1022 & 0.3129 \\ 1.3703 & -0.2414 & -0.8649 \end{pmatrix}$$

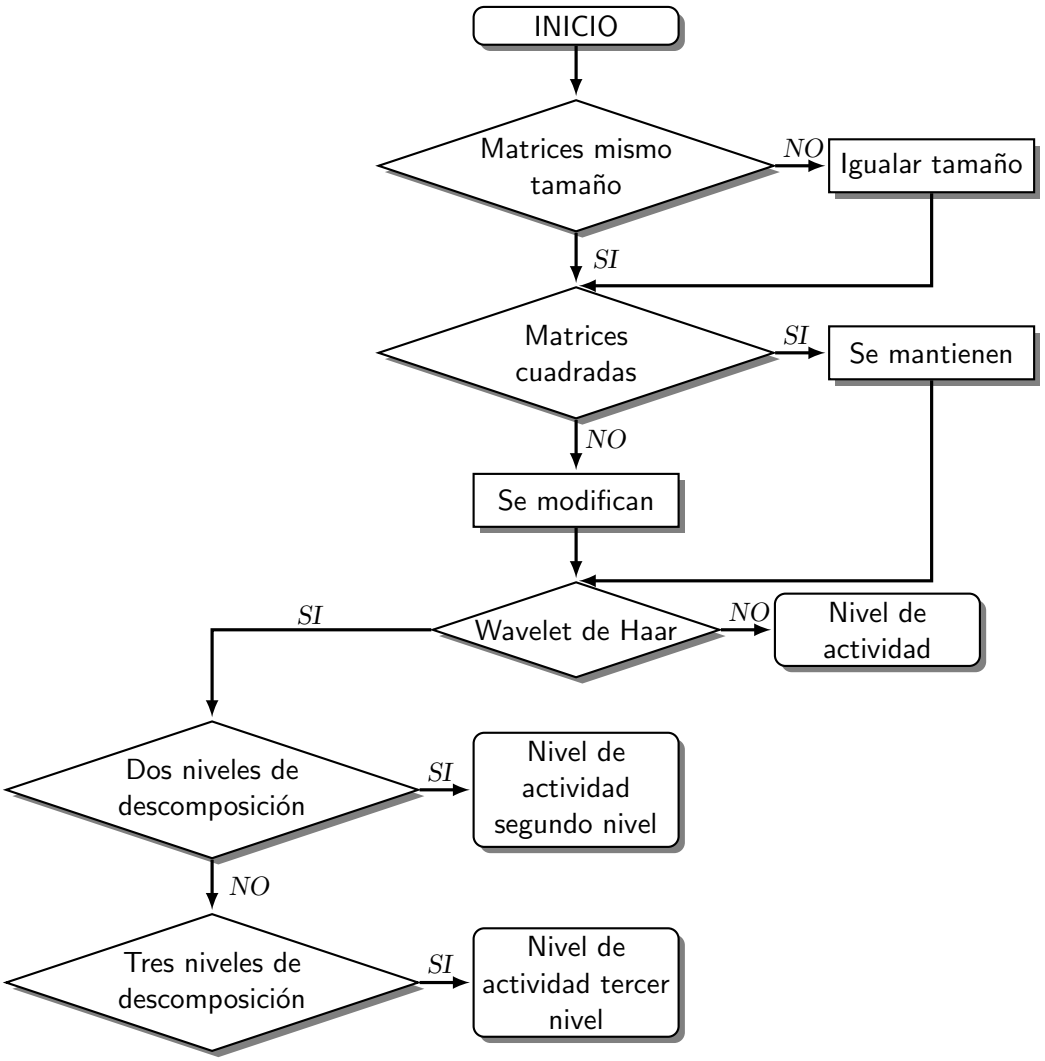
Tras calcular el valor absoluto quedan todos los coeficientes en positivo:

$$|X| = \begin{pmatrix} 0.3034 & 0.8884 & 0.8095 \\ 0.2939 & 1.1471 & 2.9443 \\ 0.7873 & 1.0689 & 1.4384 \end{pmatrix} |Y| = \begin{pmatrix} 0.3252 & 1.7115 & 0.3192 \\ 0.7549 & 0.1022 & 0.3129 \\ 1.3703 & 0.2414 & 0.8649 \end{pmatrix}$$

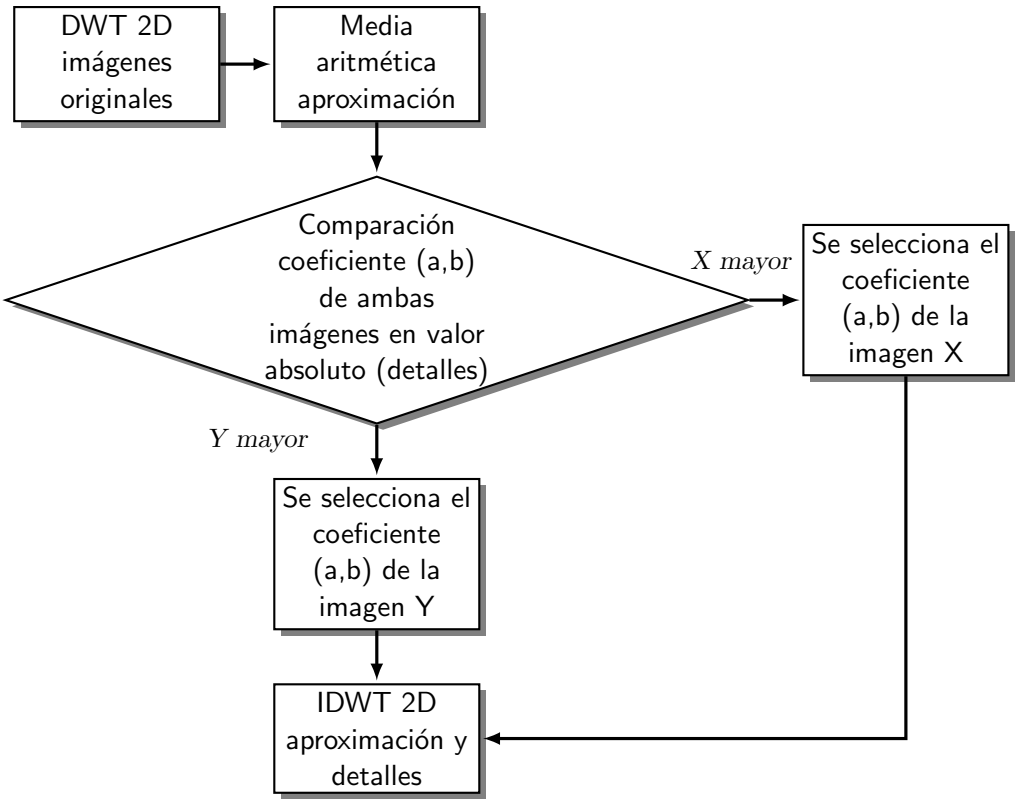
Y al compararlos uno a uno y seleccionar el de mayor valor queda la matriz resultante

$$Z = \begin{pmatrix} 0.3252 & -1.7115 & -0.8095 \\ -0.7549 & -1.1471 & -2.9443 \\ 1.3703 & -1.0689 & 1.4384 \end{pmatrix}$$

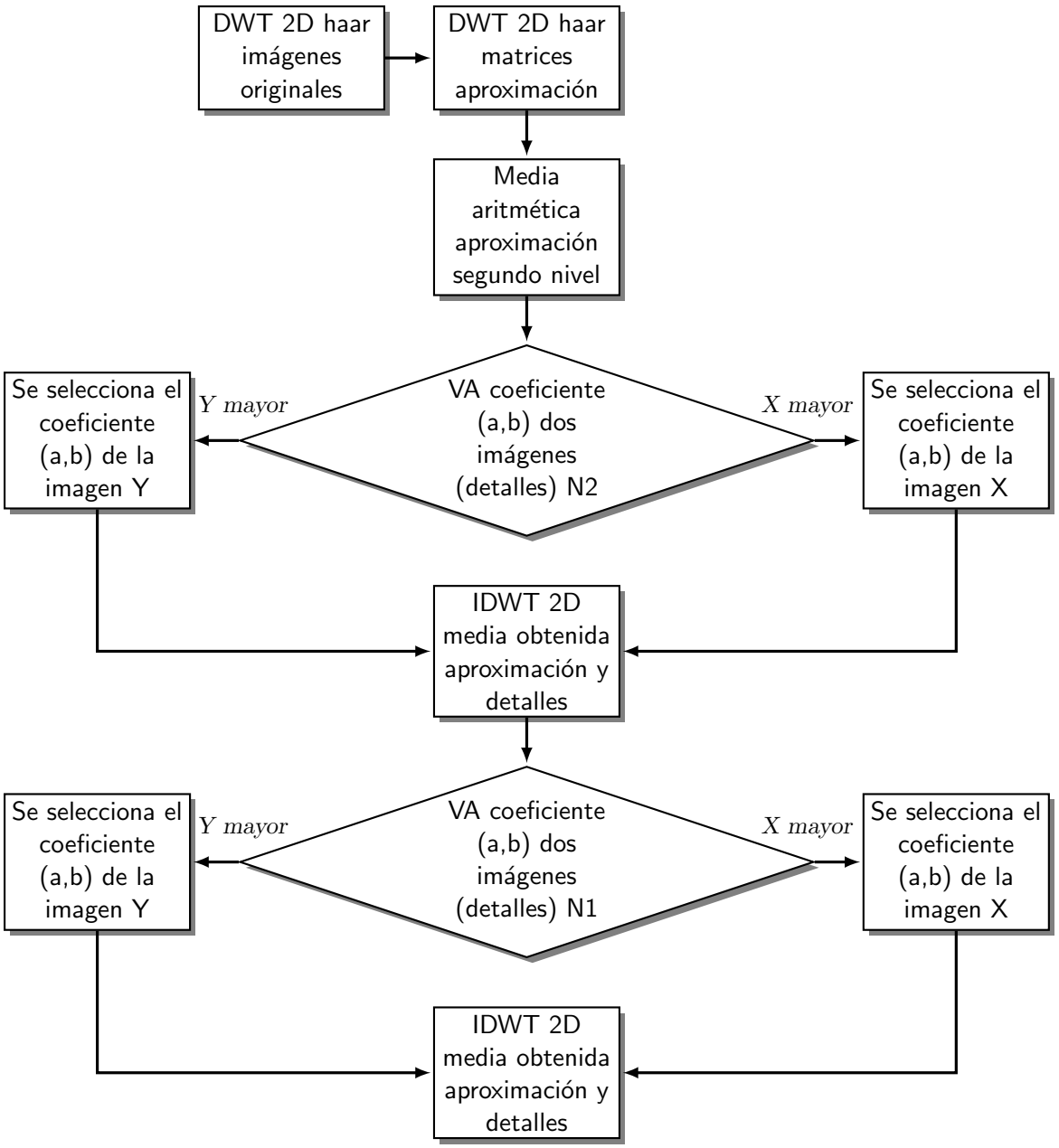
FLUJOGRAMA


INICIO
Matrices mismo tamaño
NO
Igualar tamaño
SI
Matrices cuadradas
SI
Se mantienen
NO
Se modifican
SI
Wavelet de Haar
NO
Nivel de actividad
Dos niveles de descomposición
SI
Nivel de actividad segundo nivel
NO
Tres niveles de descomposición
SI
Nivel de actividad tercer nivel

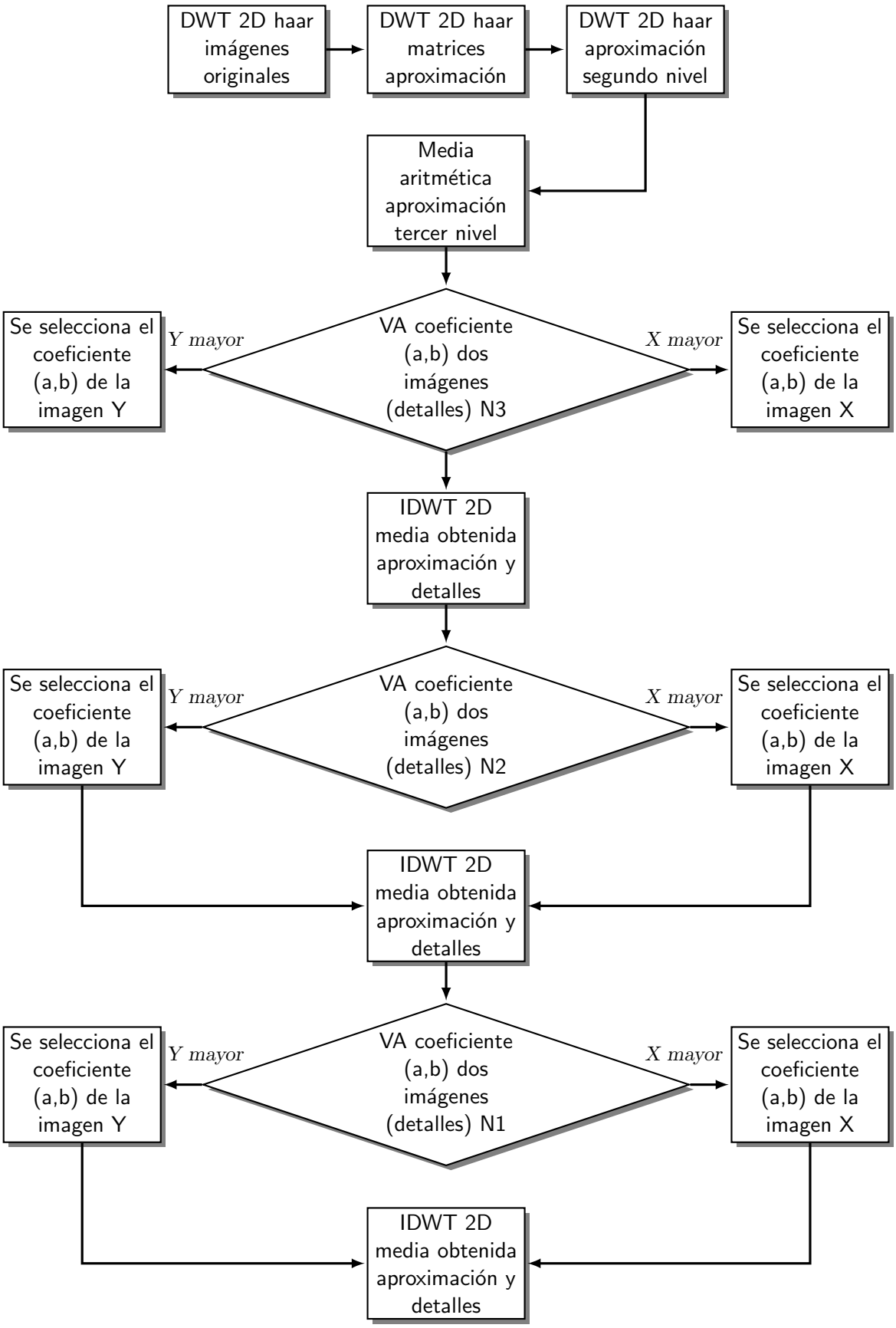
Desglose del bloque *Nivel de actividad:*



Desglose del bloque *Nivel de actividad segundo nivel:*



Desglose del bloque *Nivel de actividad tercer nivel:*



3.2.8. *Medición del nivel de actividad sobre una ventana de coeficientes*

Este algoritmo funciona de forma similar al anterior, con la diferencia que en este caso no se comparan los valores absolutos de las submatrices de detalle coeficiente a coeficiente, sino que se comparan dentro de una ventana de tamaño S que puede ser 3, 5, 11 o 21 y que rodea a un coeficiente en la posición determinada (m, n), y basa su regla de decisión en seleccionar el coeficiente en la posición (m, n) con su signo original de la ventana que gane por mayoría.
Para las submatrices de aproximación se calcula la media.

Aprovechando el ejemplo ilustrativo del algoritmo anterior, se supone el caso ficticio de dos ventanas a y b de tamaño $S = 3$

$$a = \begin{pmatrix} -0.3034 & 0.8884 & -0.8095 \\ 0.2939 & -1.1471 & -2.9443 \\ -0.7873 & -1.0689 & 1.4384 \end{pmatrix} b = \begin{pmatrix} 0.3252 & -1.7115 & 0.3192 \\ -0.7549 & -0.1022 & 0.3129 \\ 1.3703 & -0.2414 & -0.8649 \end{pmatrix}$$

que rodean a los respectivos coeficientes de la posición (m, n) (en este caso posición $(2, 2)$) en cada una de unas supuestas submatrices de detalle.

Tras calcular el valor absoluto quedan todos los coeficientes de la ventana en positivo:

$$|a| = \begin{pmatrix} 0.3034 & 0.8884 & 0.8095 \\ 0.2939 & 1.1471 & 2.9443 \\ 0.7873 & 1.0689 & 1.4384 \end{pmatrix} |b| = \begin{pmatrix} 0.3252 & 1.7115 & 0.3192 \\ 0.7549 & 0.1022 & 0.3129 \\ 1.3703 & 0.2414 & 0.8649 \end{pmatrix}$$

A continuación se comparan los coeficientes de las ventanas en posiciones correspondientes uno a uno, y se selecciona el coeficiente de la posición (m, n) de la ventana que gana por mayoría, que será el coeficiente que quede en la posición (m, n) de la matriz de la imagen fusionada.

Para el caso del ejemplo, al comparar las ventanas se ve que la ventana a gana por cinco a cuatro. Luego en la matriz fusionada, en la posición (m, n) iría el coeficiente $a(m, n) = -1.1471$.

El flujograma del método anterior es perfectamente aplicable a este caso, luego no se detalla con objeto de no ser redundante.

Ejemplo de fusión empleando este algoritmo:

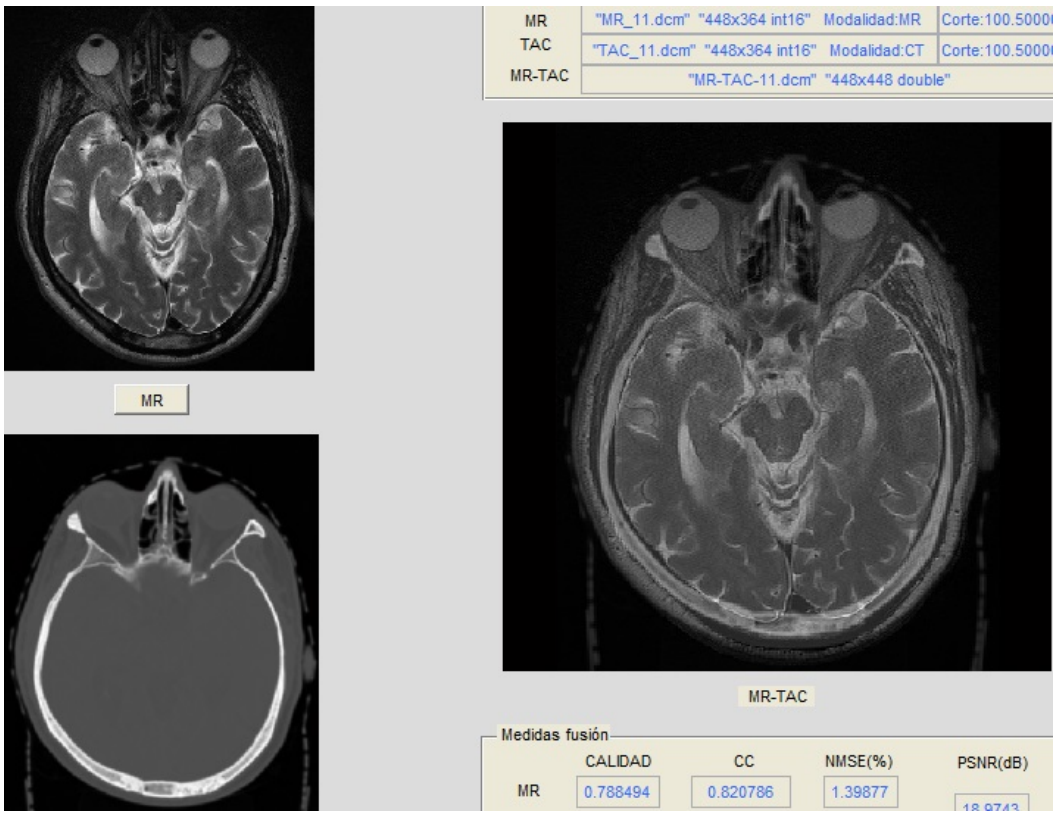


Figura 3.13: Fusión mediante nivel de actividad basado en ventana.

Capítulo 4

Interfaz gráfica

Si bien al comienzo de este proyecto no se había contemplado, durante el desarrollo del mismo surgió la idea de realizar una interfaz gráfica de usuario (GUI en lo sucesivo) en la que poder seleccionar tanto los pares de imágenes a fusionar como los métodos de fusión junto con los diferentes parámetros que algunos de éstos requieren para su implementación, así como unas opciones extra añadidas a dicha interfaz que aportan algo más de información al usuario y que se comentarán más adelante. Con todo ésto se hace mucho más amigable la ejecución de los algoritmos tanto para el propio creador de los mismos como para un posible usuario externo.

NOTA: El código MATLAB de programación de la interfaz gráfica se encuentra al final de este documento en el Anexo.

4.1. Creación de la interfaz gráfica

La interfaz gráfica se ha implementado en el mismo entorno MATLAB donde se ha realizado el resto de este trabajo, haciendo uso del asistente para la creación de GUI del que dispone dicho entorno y aprovechando las bondades del asistente para crear aplicaciones gráficas sin necesidad de ser un experto en dicha materia.

Una vez se ejecuta el asistente de creación de GUI en MATLAB se abre una ventana como la que se muestra en la figura que se puede ver más abajo. Se podría desglosar paso a paso cómo se ha creado la interfaz, pero se extendería en exceso y no es objetivo prioritario de este trabajo. Por tanto, símplemente se comenta que a través del menú gráfico que aparece en la parte izquierda, se puede pinchar y arrastrar para añadir los elementos que se desea que aparezcan en la aplicación, tales como pudieran ser entre otros:

. Unos ejes donde representar una imagen (Axes).

. Botones de distinto tipo, como puedan ser los que al clicar con el cursor realizan una operación determinada (Push Button), ó los que disponen de una casilla donde al marcar/desmarcar la misma se habilita/deshabilita alguna función (Check Box, Radio Button).

. Menús desplegables (Pop-up Menu, Listbox).

. Paneles que ayudan a disponer los botones de la aplicación de la forma que interese al creador de la misma (Panel).

. Cajas donde insertar texto (Static Text).

Sobre cada uno de los elementos se podrá acceder a sus propiedades (Property Inspector) y modificarlas. Las más importantes serán String, que indica el texto que aparecerá en el elemento en cuestión, y la propiedad Tag, que será el nombre que tome la función llamada por el elemento referido para que ejecute las operaciones mandadas por su parte de código.

Al guardar la interfaz con el nombre deseado, se genera una figura de MATLAB **.fig* junto con un archivo **.m* asociado, en el que se puede ir modificando el código a merced del creador para ampliar o modificar el comportamiento de cada componente de la interfaz gráfica. En este caso la aplicación se llama *fusion imagenes medicas*.

Nótese que todos los archivos de MATLAB utilizados en este trabajo, ya sean **.fig* ó **.m*, se nombran en esta memoria tal y como se han nombrado en el entorno MATLAB, cambiando únicamente los guiones bajos por espacios.

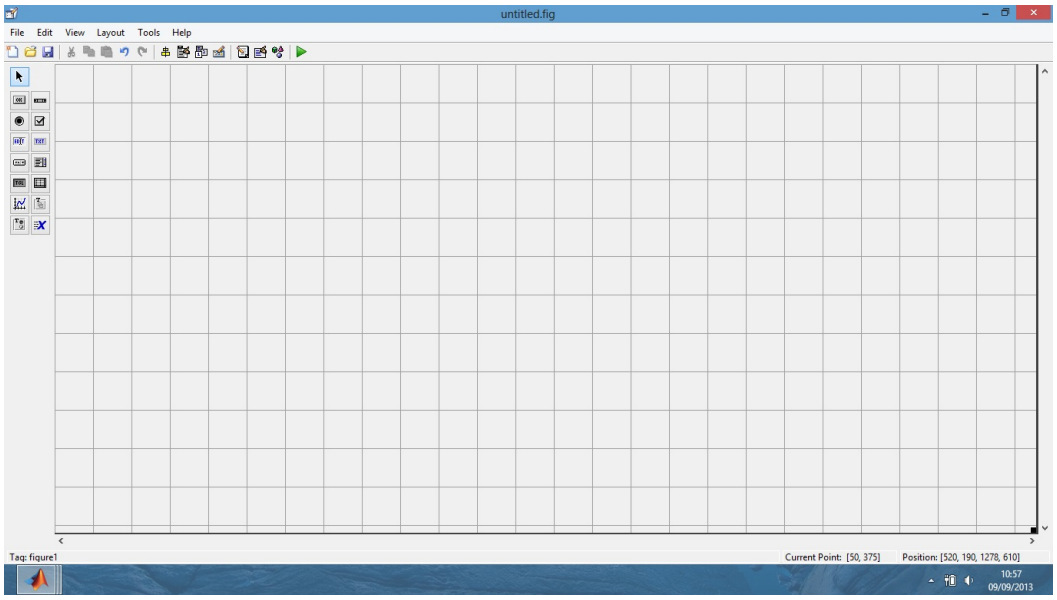


Figura 4.1: Asistente de creación de interfaces gráficas en MATLAB (GUIDE).

4.2. Guía de uso de la aplicación

Para ejecutar la interfaz gráfica, se debe introducir en el intérprete de comandos de MATLAB *fusion imagenes medicas* ó clicar sobre Run en el menú de MATLAB una vez abierto el archivo *fusion imagenes medicas.m*

La interfaz gráfica tiene en su origen el aspecto que se muestra a continuación:

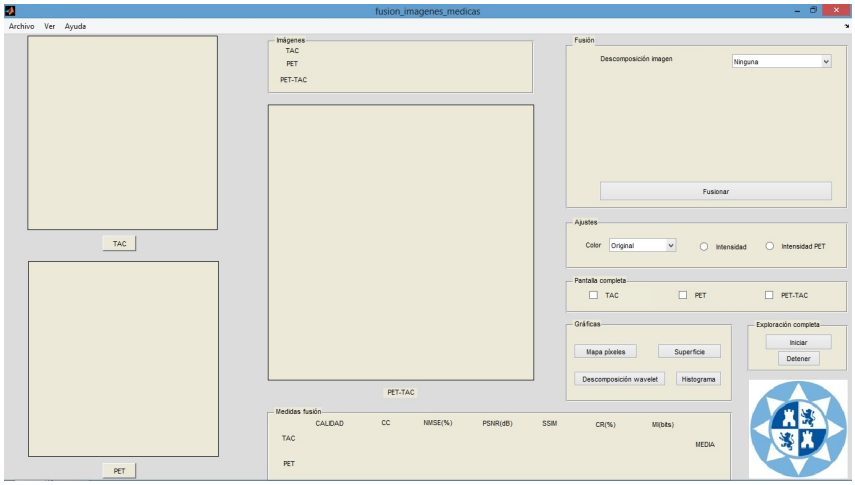


Figura 4.2: Interfaz gráfica de usuario *fusión imágenes médicas*

En ella aparecen tres recuadros donde se mostrarán las dos imágenes a fusionar y la fusionada (ésta útima con un tamaño algo mayor), y los paneles Imágenes, Fusión, Ajustes, Pantalla completa, Gráficas, Exploración completa y Medidas fusión. La aplicación cuenta en su parte superior con un menú en el que aparecen las etiquetas *Archivo*, *Ver* y *Ayuda*. En *Archivo-Cargar* se selecciona qué tipo de fusión se desea realizar entre PET-TAC, MR-PET ó MR-TAC, donde para éste último caso se dispone de dos estudios de pacientes distintos, Paciente 1 y Paciente 2. Si se selecciona la fusión PET-TAC se dispone de un estudio de 205 cortes correspondientes de un mismo paciente. Si por el contrario se elige fusionar resonancia magnética y PET se dispone de cuatro cortes, y para la fusión MR-TAC se dispone de once cortes en el primer paciente y cinco en el segundo. Para cada estudio elegido cambian los textos que aparecen en algunos botones y otros textos fijos de la interfaz.

Siguiendo dentro del menú superior, en *Ver-Idioma* se puede elegir el idioma que se quiere utilizar en la interfaz entre español, inglés y alemán. Al pinchar en *Ayuda-Guía* se muestra una ventana con una guía básica de uso de la interfaz (ver Figura 4.3), mientras que tras clicar en *Ayuda-Acerca de* aparece una pequeña ventana con el nombre de la aplicación y de los autores que la realizaron. (Figura 4.4)

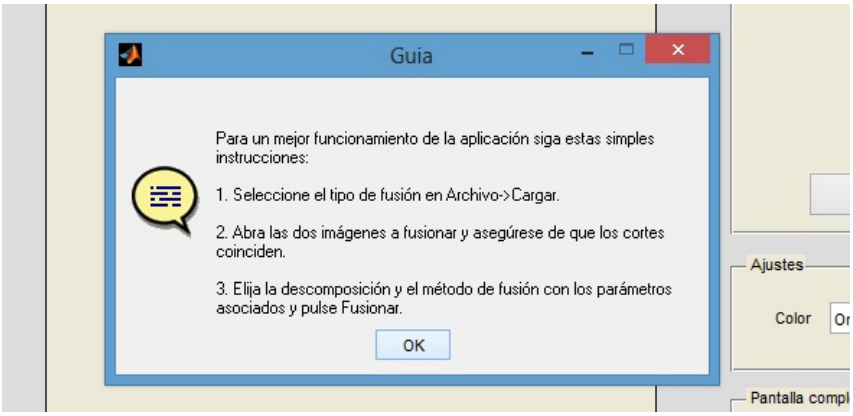


Figura 4.3: Guía básica de apoyo en el uso de la aplicación.

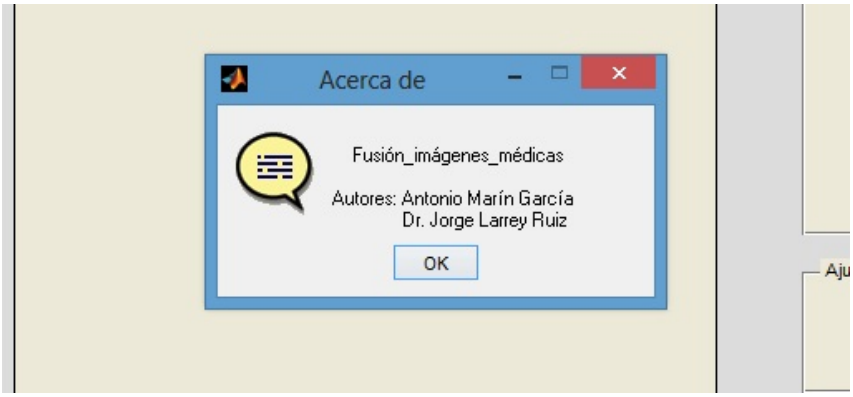


Figura 4.4: Nombre de la aplicación y autores.

Luego una vez seleccionado en *Archivo* el tipo de fusión a realizar, se clica en los botones que aparecen debajo de los dos recuadros del lado izuierdo para seleccionar las imágenes a fusionar. Así por ejemplo, para un caso práctico de una fusión MR-TAC, se clica en el botón *MR* y se abrirá una ventana con las imágenes por resonancia magnética disponibles para ese paciente. Pinchando en el botón *TAC* aparecerá la ventana con los TAC que se corresponden con dichas resonancias.

Se ha procurado facilitar la selección de las imágenes haciendo que los botones que se ven debajo de las mismas enlacen inequívocamente con el directorio donde se hallan, así como se ha hecho coincidir la nomenclatura de los distintos cortes, que sigue la forma *formato de imagen-número de corte.extensión del archivo*. Es decir, en una fusión MR-TAC, si se ha seleccionado la resonancia magnética *MR-7.dcm*, se deberá elegir el TAC *TAC-7.dcm*

No obstante, si por cualquier motivo se seleccionara un tipo de imagen distinta a la que espera la aplicación, se mostraría un mensaje de error.

Así por ejemplo, si al clicar sobre el botón *MR* se selecciona una imagen que no es una resonancia magnética aparece el siguiente mensaje:

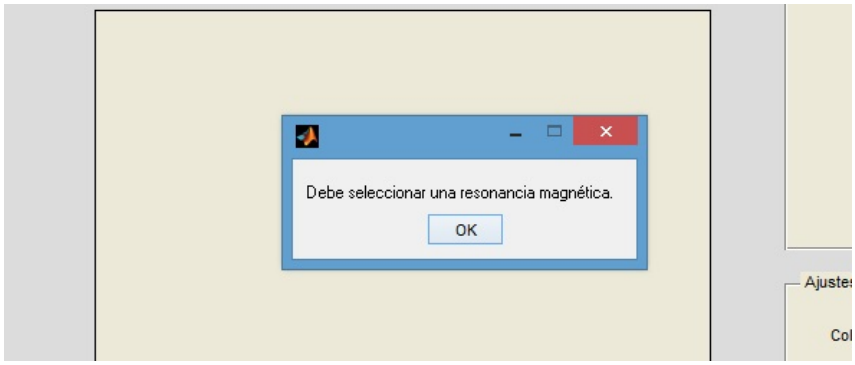


Figura 4.5: Mensaje de error MR.

Cuando se selecciona una imagen que no es un TAC al clicar sobre el botón *TAC* se muestra el mensaje:

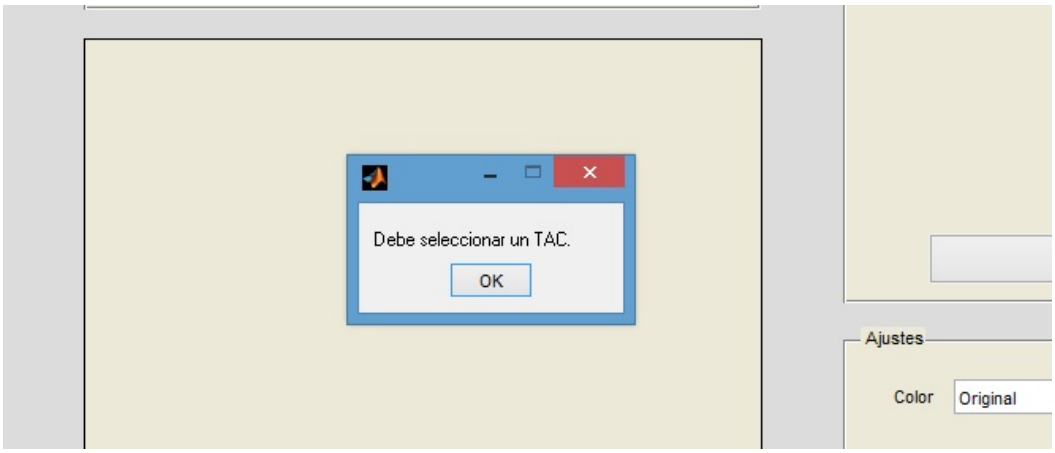


Figura 4.6: Mensaje de error TAC

Si al clicar sobre PET se selecciona una imagen que no se corresponden con una tomografía por emisión de positrones se leerá el mensaje:

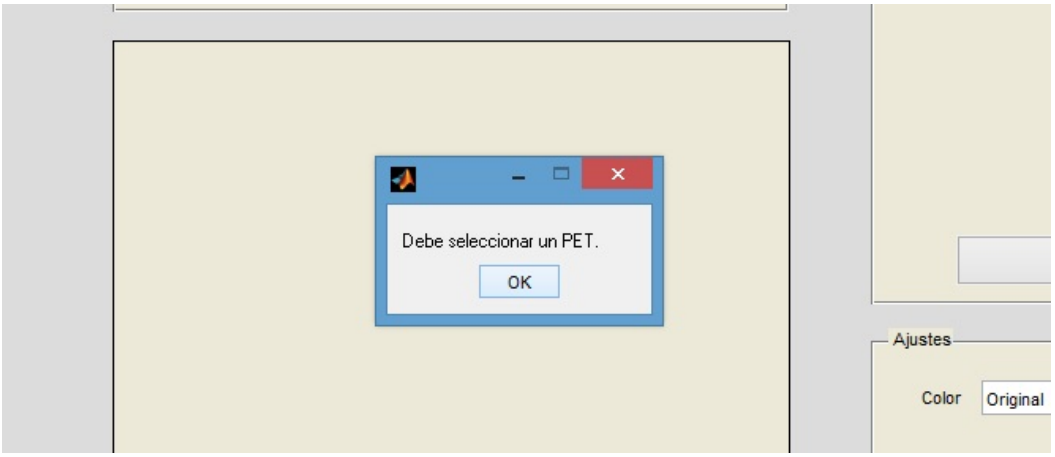


Figura 4.7: Mensaje de error PET

Una vez se han seleccionado las dos imágenes a fusionar, la interfaz gráfica muestra el siguiente aspecto:

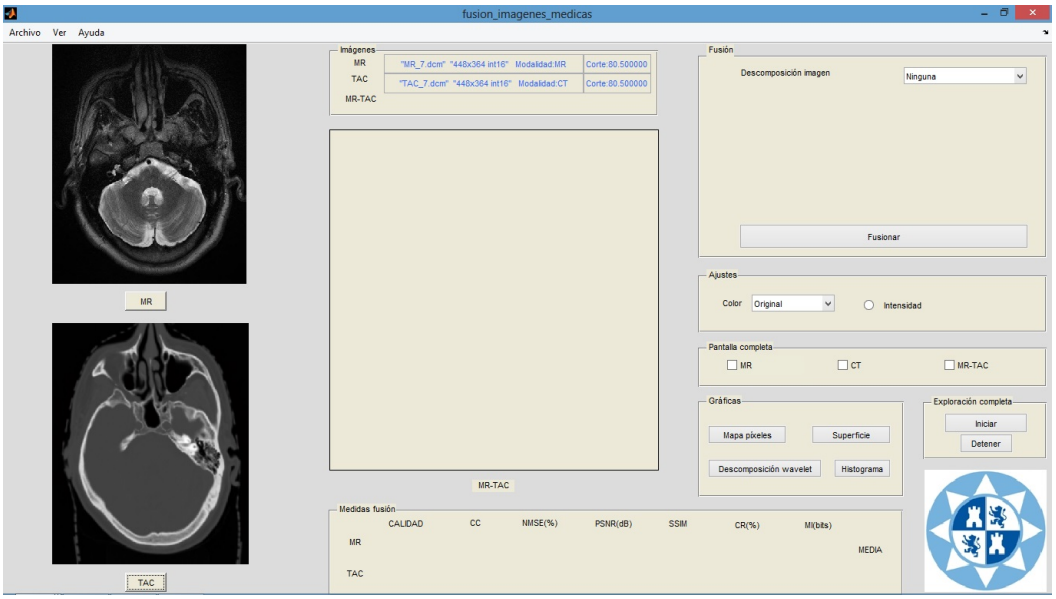


Figura 4.8: Imágenes a fusionar ya seleccionadas

A continuación en el panel Fusión ya se puede seleccionar la configuración de la fusión que se desea realizar. Aparece un primer menú desplegable en el que elegir si se quiere descomponer la imagen con la transformada wavelet o no. Si no se descompone la imagen, las demás opciones del panel Fusión no se harán visibles y el único método de fusión será en este caso la media aritmética y por tanto ya se puede clicar sobre el botón Fusionar.

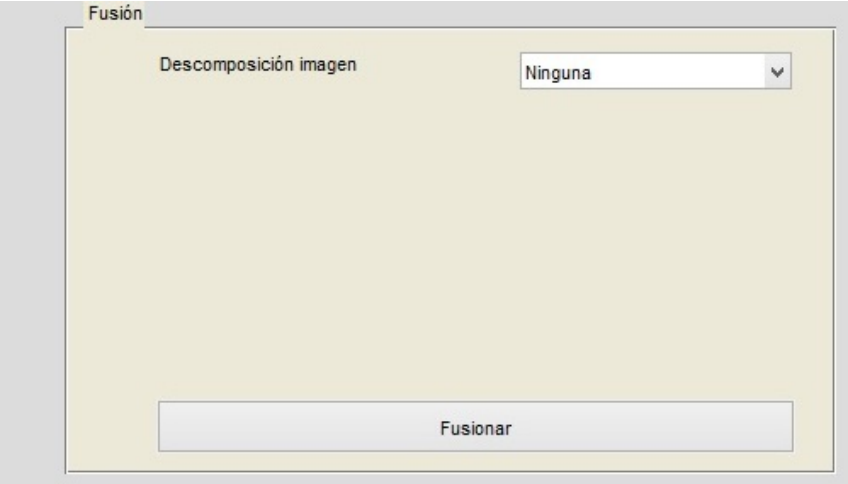


Figura 4.9: Seleccionada la opción de no descomponer las imágenes antes de la fusión

Al elegir descomponer las imágenes aparecen otra serie de opciones que permiten configurar los parámetros con los que se hará la fusión.

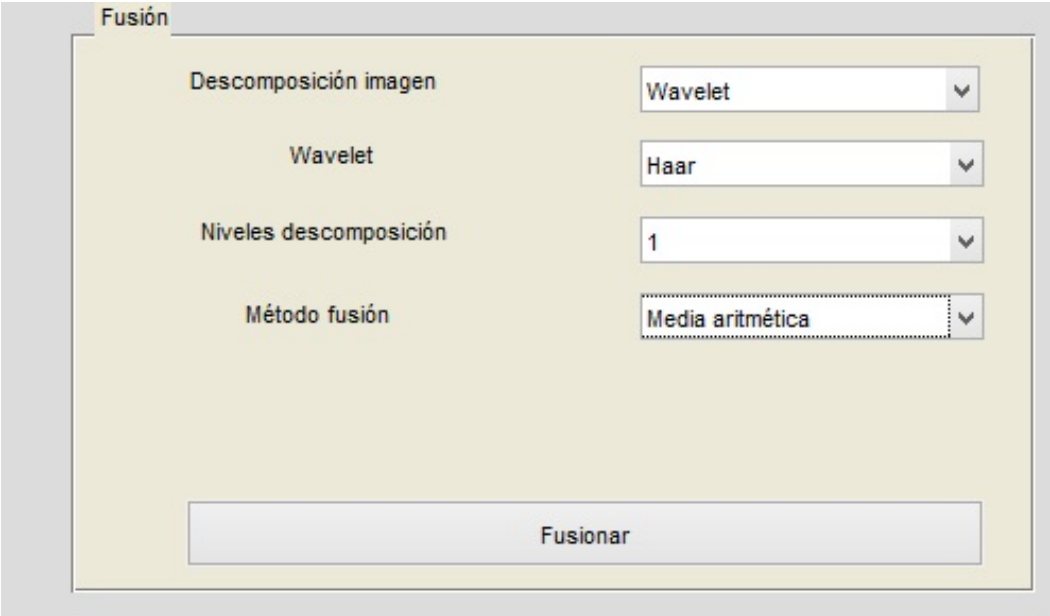


Figura 4.10: Seleccionada la opción de descomponer las imágenes antes de la fusión con la transformada wavelet

En Wavelet se puede elegir la familia wavelet con la que se quiere calcular la transformada.

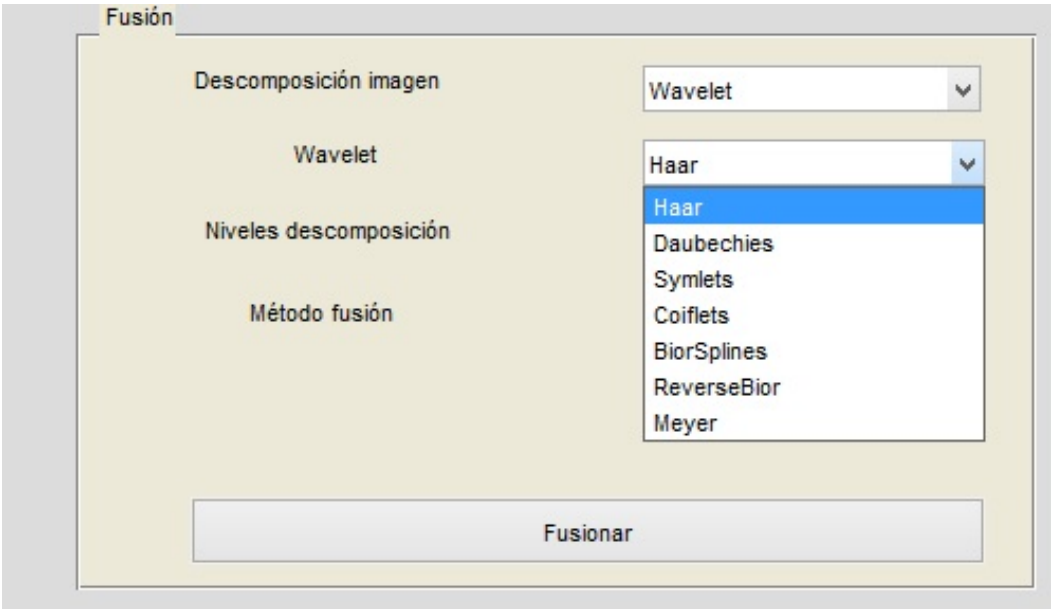


Figura 4.11: Menú desplegable para elegir la familia wavelet

Niveles permite elegir hasta qué nivel de descomposición se quiere llegar.

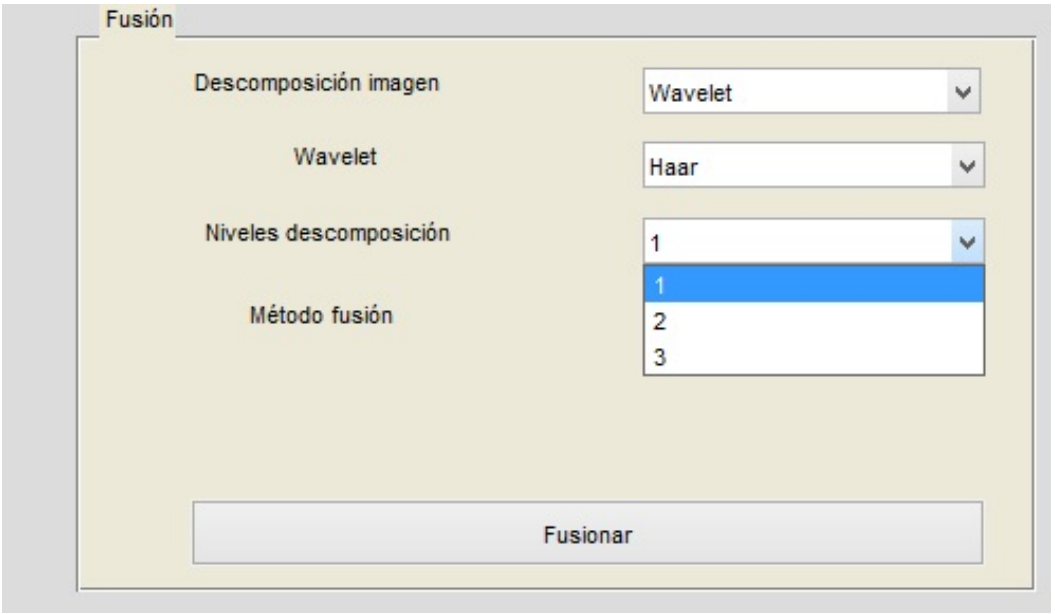


Figura 4.12: Niveles de descomposición

Y por último en Método fusión se puede seleccionar de los algoritmos que se han implementado, el que se pretende aplicar a la fusión de las dos imágenes médicas.

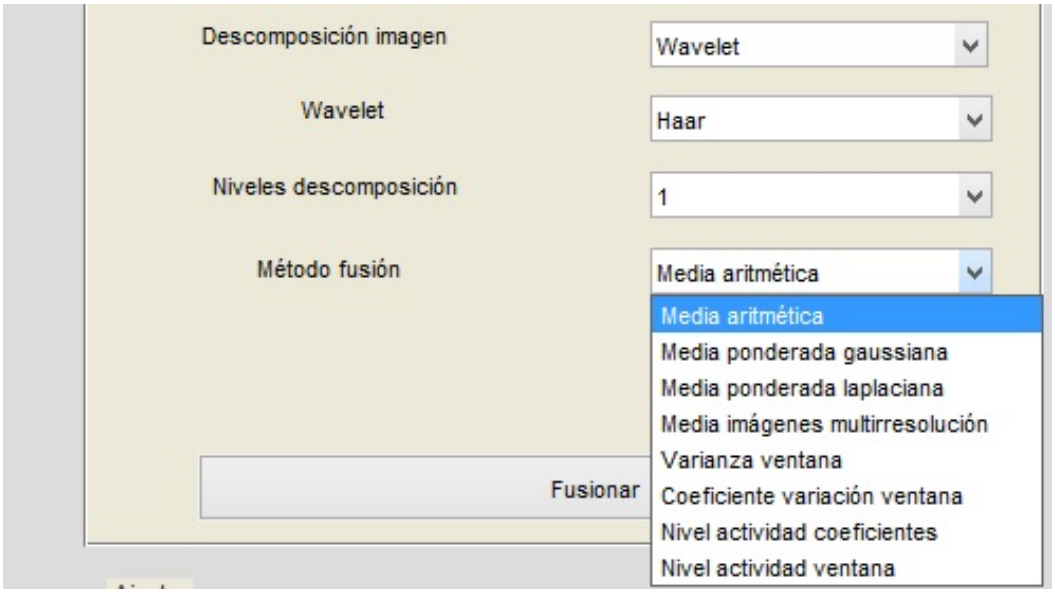


Figura 4.13: Algoritmos de fusión de imagen disponibles

Algunos de los algoritmos necesitan de unos parámetros para realizar la fusión, como son en este caso Ventana y Umbral. Cuando se seleccione por tanto un método de fusión que requiera de alguno de estos parámetros ó de los dos, se harán visibles unos menús desplegables extra donde poder concretar los parámetros deseados para calcular la fusión.

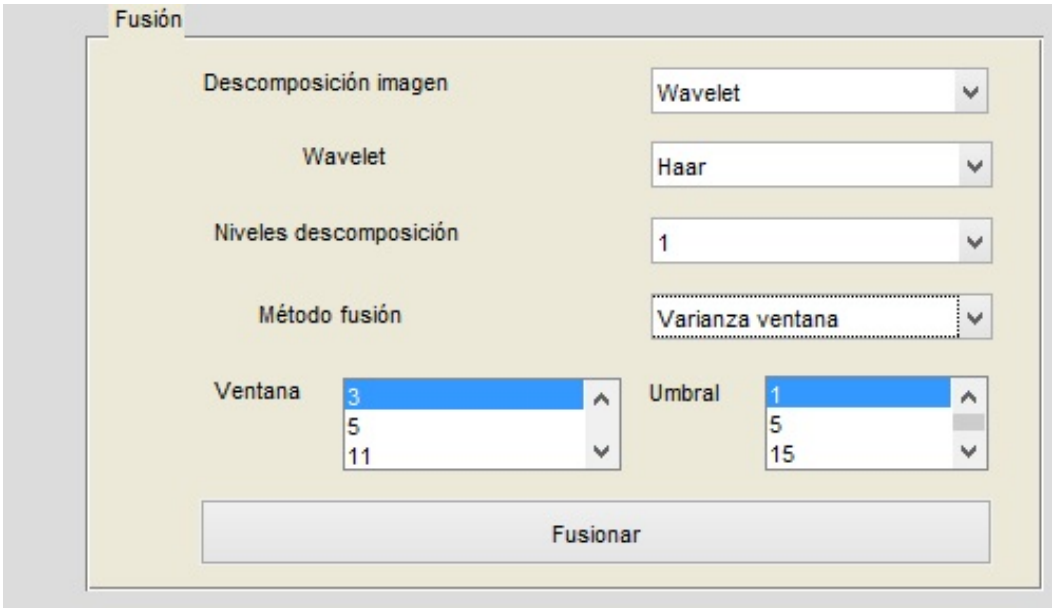


Figura 4.14: Parámetros Ventana y Umbral

Con todas las opciones ya seleccionadas se pulsa el botón Fusionar y aparece la imagen fusionada en el centro de la interfaz.

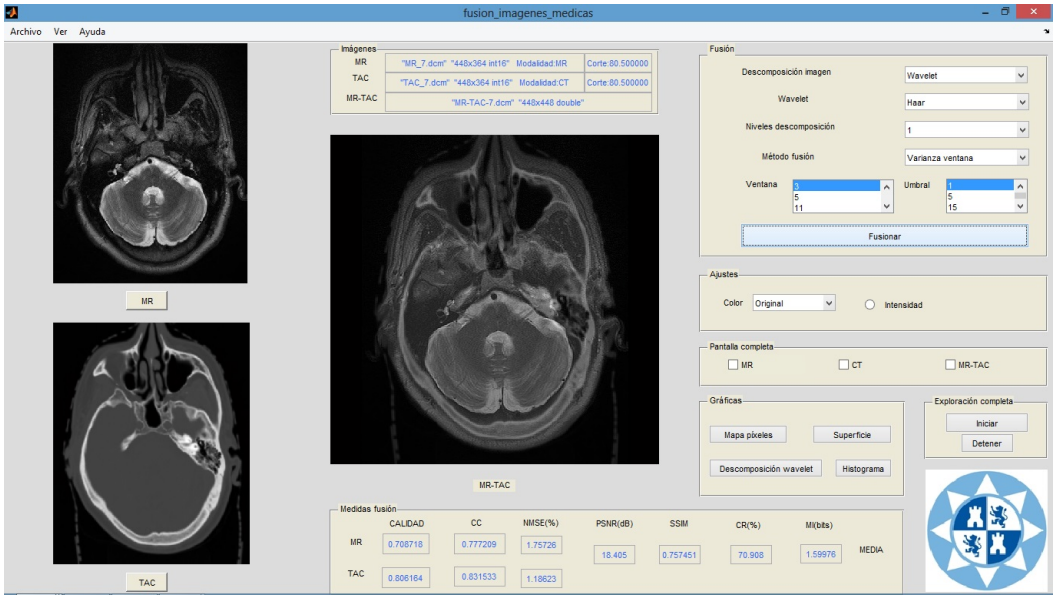


Figura 4.15: La interfaz muestra tanto las imágenes originales como la fusionada.

Una vez mostrada la imagen fusionada se muestran a su vez dos paneles informativos como son Imágenes y Medidas fusión. En el primero se puede leer información de las tres imágenes. De las imágenes originales se puede ver el nombre del archivo, el tamaño de la imagen junto con el tipo de dato, la modalidad de imagen médica y el número de corte axial. De la imagen fusionada se da información del nombre, tamaño de la misma y tipo de dato en que está almacenada.

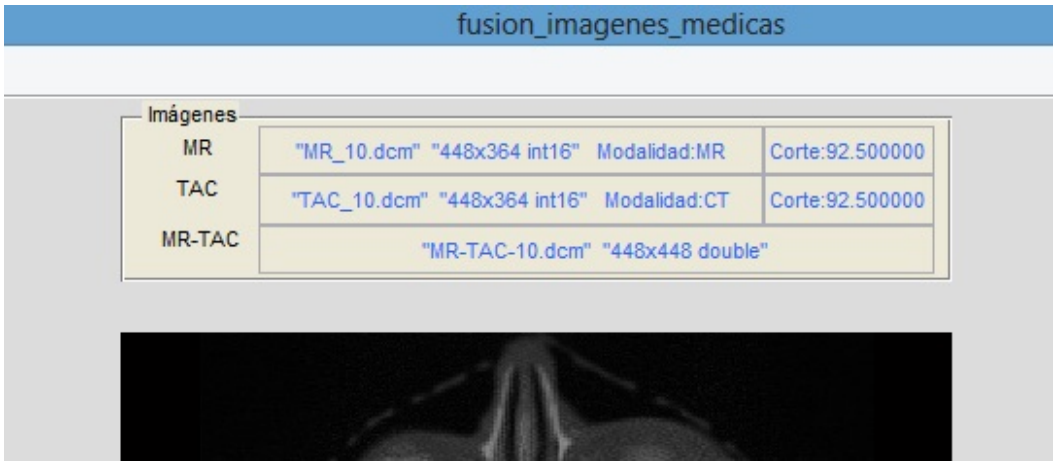


Figura 4.16: Panel donde se muestra la información más relevante de las tres imágenes

En la parte baja de la ventana se muestran las medidas numéricas que se han calculado para poder clasificar los distintos algoritmos.

Figura 4.17: Panel donde se muestra la información de las medidas calculadas tras cada fusión

Si se hubiera pulsado el botón Fusionar sin haber elegido las dos imágenes de partida aparecería el siguiente mensaje de error:

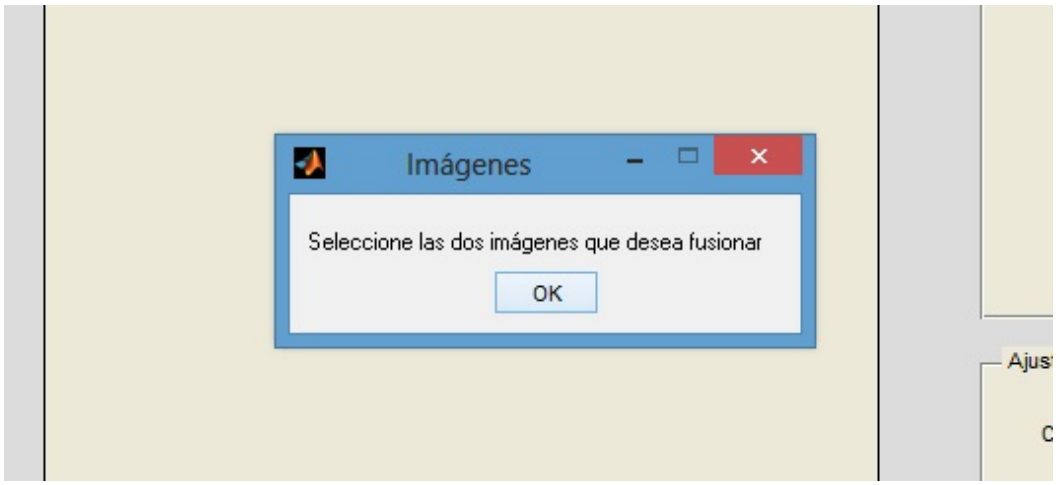


Figura 4.18: Se ha pulsado el botón Fusionar sin haber elegido las imágenes de partida.

Si al pulsar el botón Fusionar sólo hubiera una de las dos imágenes de partida se leería un mensaje de error como muestra la siguiente figura:

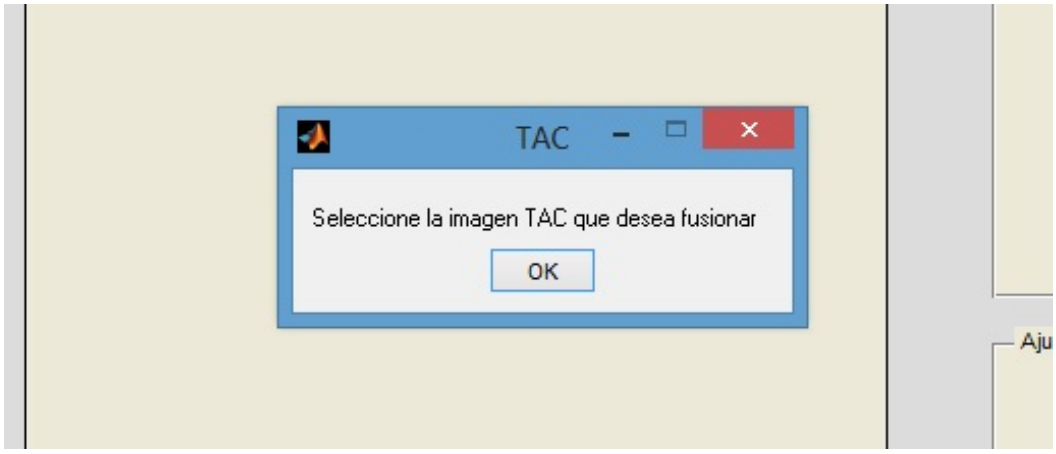


Figura 4.19: Se ha seleccionado únicamente una de las imágenes de partida. En este caso por ejemplo falta seleccionar el TAC

Si se seleccionan dos imágenes de cortes anatómicos del paciente que no coinciden, no se calcula la imagen fusionada y a cambio se muestra un mensaje de aviso de selección de cortes incorrecta.

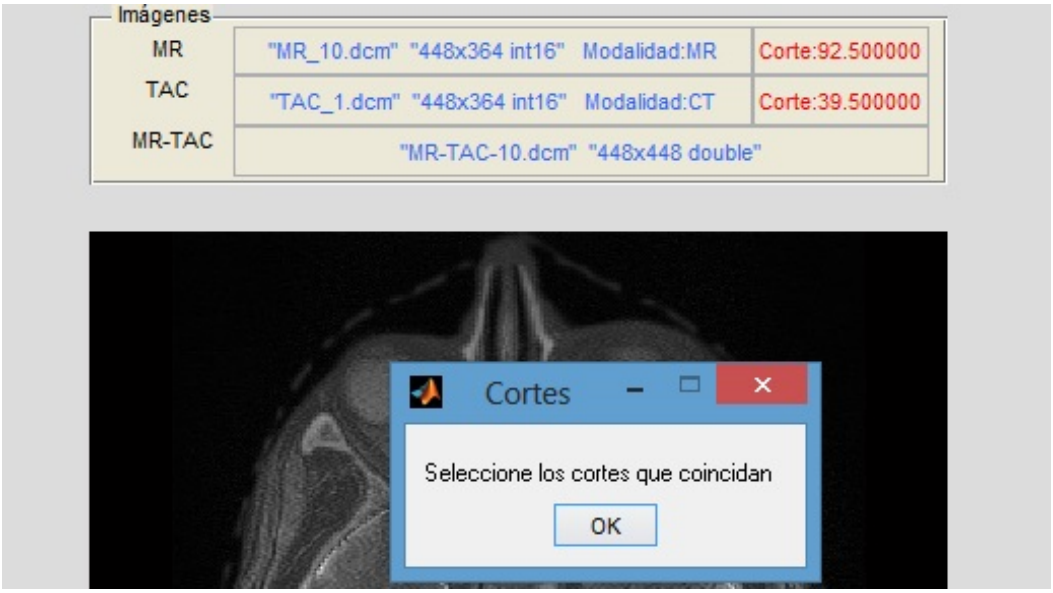


Figura 4.20: Las imágenes a fusionar son de cortes que no se corresponden.

En el panel Ajustes aparece un menú deplegable Color en el que se puede modificar el color de las tres imágenes, dando opción a elegir algunos de los *colormap* de MATLAB. Están contenidos también en dicho panel los botones con casilla de seleccionar/deseleccionar Intensidad e Intensidas PET (éste último sólo disponible para fusión PET-TAC). El primero realiza un ajuste al alza de los niveles de intensidad de gris en toda la imagen fusionada, mientras que Intensidad PET eleva selectivamente los niveles intensidad únicamente en la zona de captación que ha mostrado el PET.

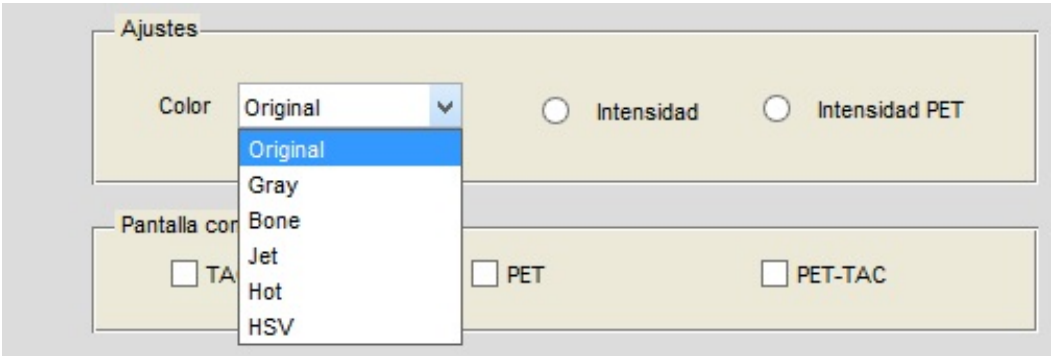


Figura 4.21: Panel donde se pueden modificar los colores ó los niveles de intensidad de las imágenes.

Así por ejemplo, si se elige el color Hot el resultado visual es el que se muestra a continuación:

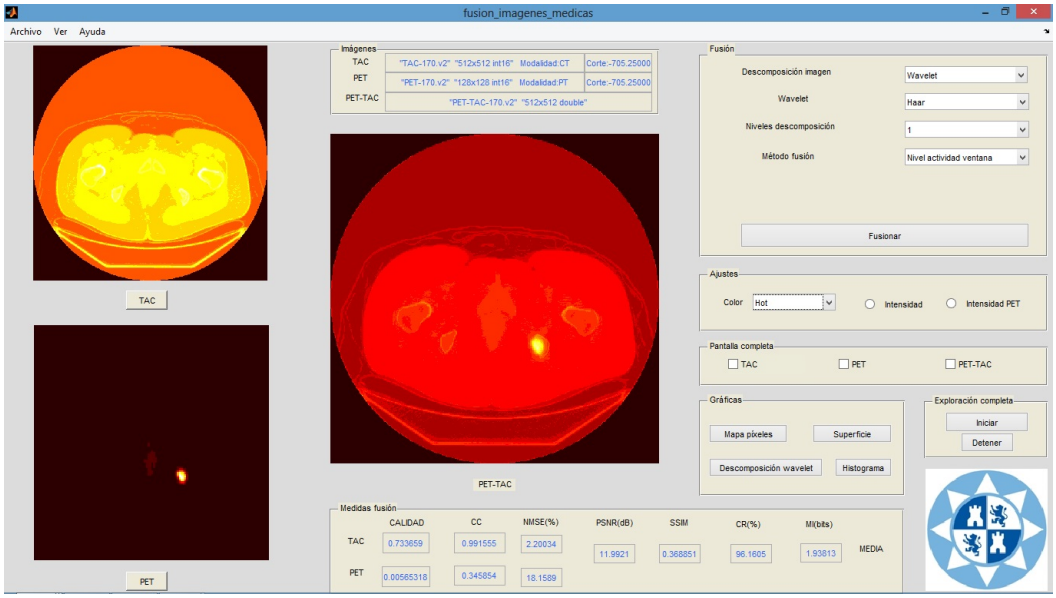


Figura 4.22: Seleccionado el *colormap* Hot.

Si por el contrario se opta por el color Jet las imágenes cambian al siguiente estado:

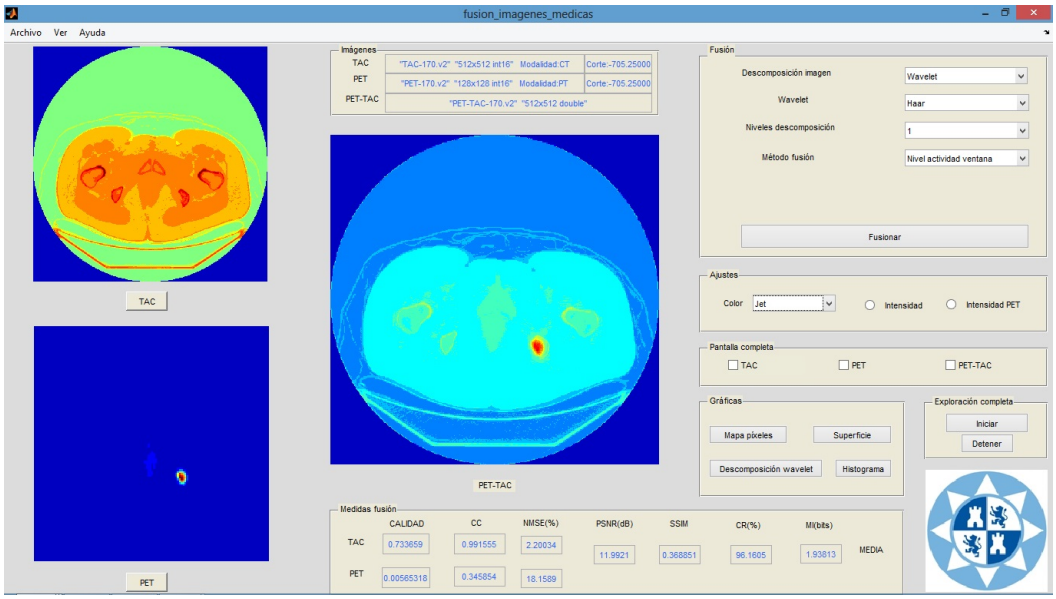


Figura 4.23: Seleccionado el *colormap* Jet.

Cuando se marca por ejemplo la casilla Intensidad PET se resalta la zona de actividad metabólica del PET:

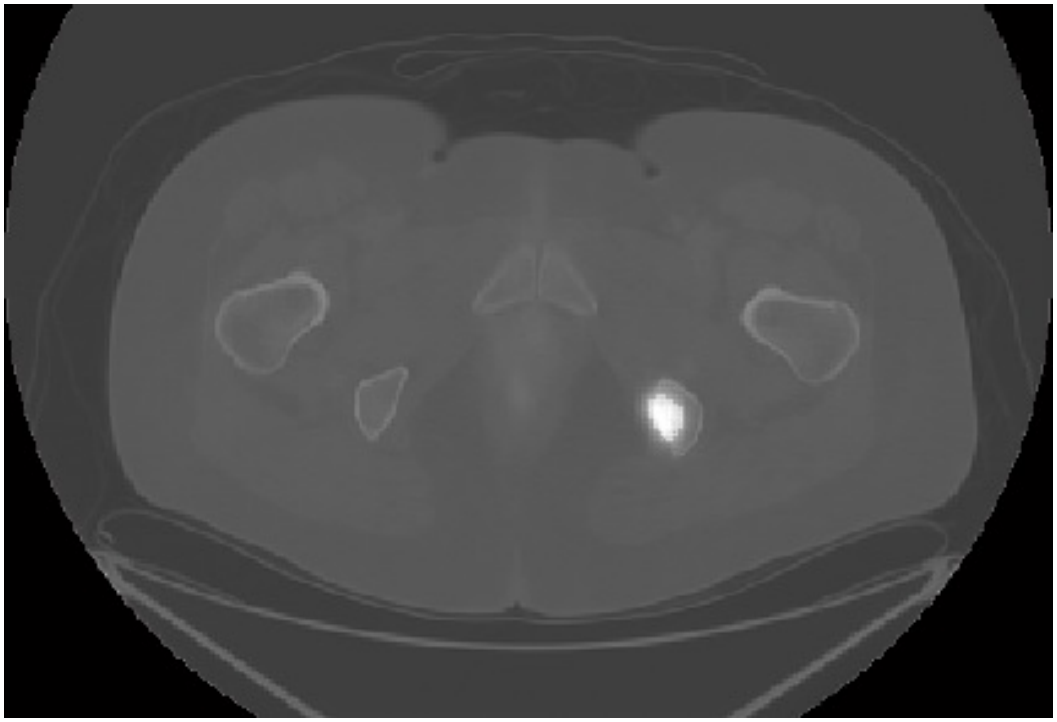


Figura 4.24: Imagen fusionada PET-TAC en la que se resalta la zona interesada.

En el panel Pantalla completa, se muestra en un tamaño mayor presidiendo la interfaz la imagen correspondiente a la casilla que haya marcado el usuario. Para volver al estado original de la aplicación basta con desmarcar la casilla marcada.

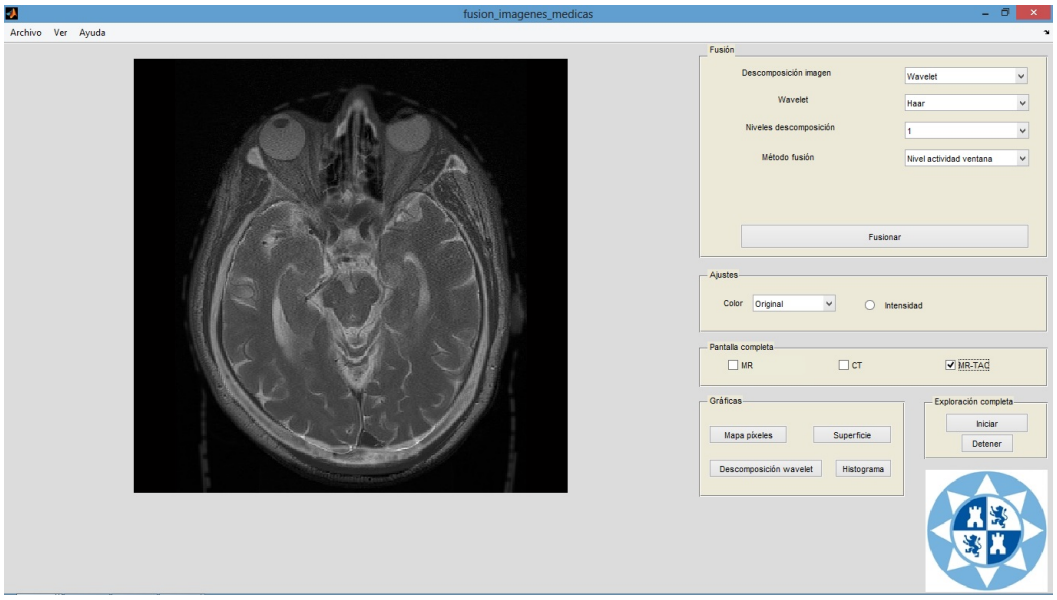


Figura 4.25: Imagen fusionada MR-TAC a pantalla completa.

El panel Exploración completa queda disponible únicamente para el estudio PET-TAC. Al pulsar sobre el botón Iniciar, comienza un estudio completo desde el primer corte hasta el último de los disponibles para dicho estudio, mostrando au-

tomáticamente los dos cortes a fusionar y la imagen fusionada, y haciendo una pausa de un segundo antes de pasar al siguiente corte. La exploración se puede pausar si se desea para examinar un corte en profundidad pulsando sobre Pausar (en ese caso se debe pinchar sobre Continuar cuando se quiera proseguir), y se puede terminar de forma voluntaria clicando sobre Detener.

Se muestra una captura de un momento de la exploración en la que se hizo una pausa en el corte número veinte:

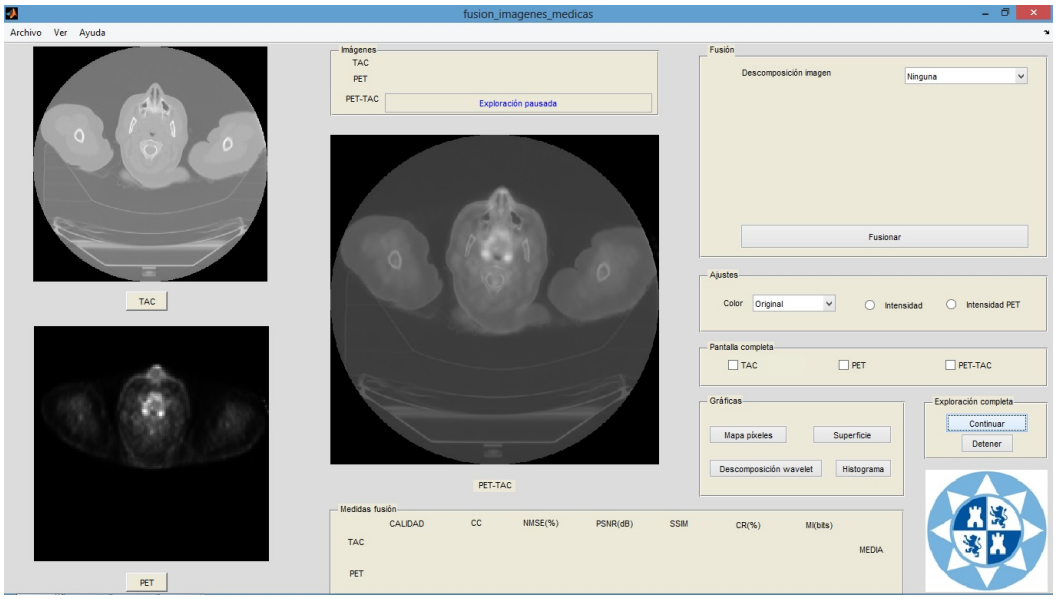


Figura 4.26: Pausa durante una exploración completa.

Por último, el panel Gráficas contiene cuatro botones:

Mapa píxeles

Ofrece la posibilidad de observar con gran detalle la distribución de píxeles en la imagen. Ampliando mucho el zoom se pueden ver los valores numéricos de intensidad, acotados entre cero y uno. Puede ser de especial utilidad cuando se quiere ver con gran detalle por ejemplo la distribución de la zona de actividad metabólica del PET en la imagen fusionada PET-TAC. (Figura 4.27)

Superficie

Genera una gráfica en tres dimensiones donde la altura representa los niveles de intensidad de gris de los píxeles de la imagen fusionada. Dicho nivel de intensidad es proporcional a la altura de la superficie (los valores quedan en el rango 0-255). Los ejes X e Y quedan delimitados por el tamaño de la imagen. (Figura 4.28)

Descomposición wavelet

Estas gráficas muestran cómo quedan las tres imágenes cuando se les aplica la transformada wavelet discreta en dos dimensiones. En ellas se pueden ver para cada una de las imágenes las matrices de aproximación y las de detalles horizontales, verticales y diagonales. (Figura 4.29)

Histograma

La gráfica crea un histograma de los datos de la imagen, en la que se puede apreciar cuántos píxeles hay en cada valor de intensidad de gris desde 0 a 255. (Figura 4.30)

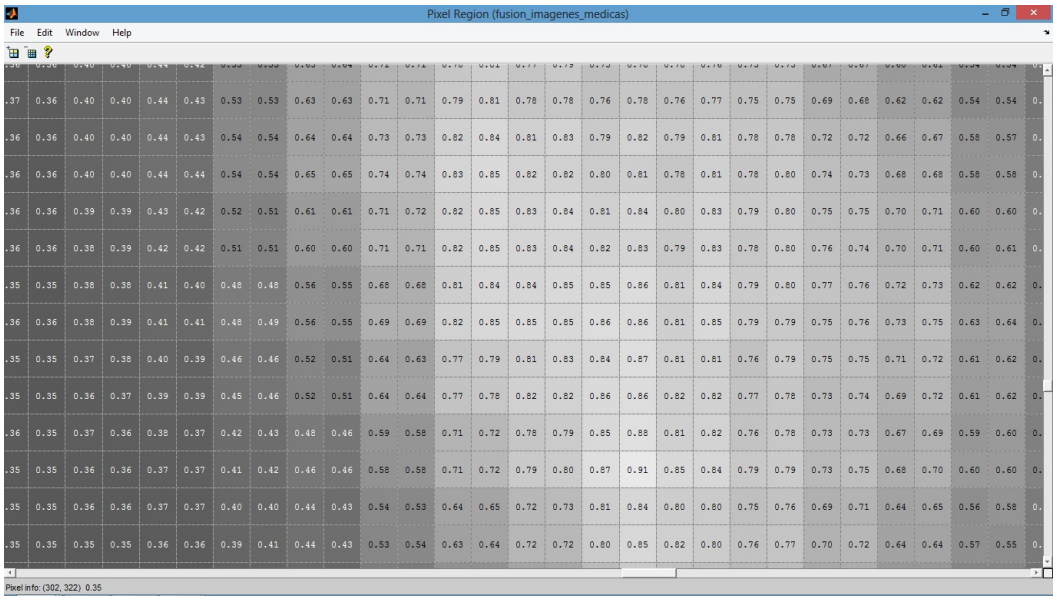


Figura 4.27: Mapa de píxeles de la imagen fusionada.

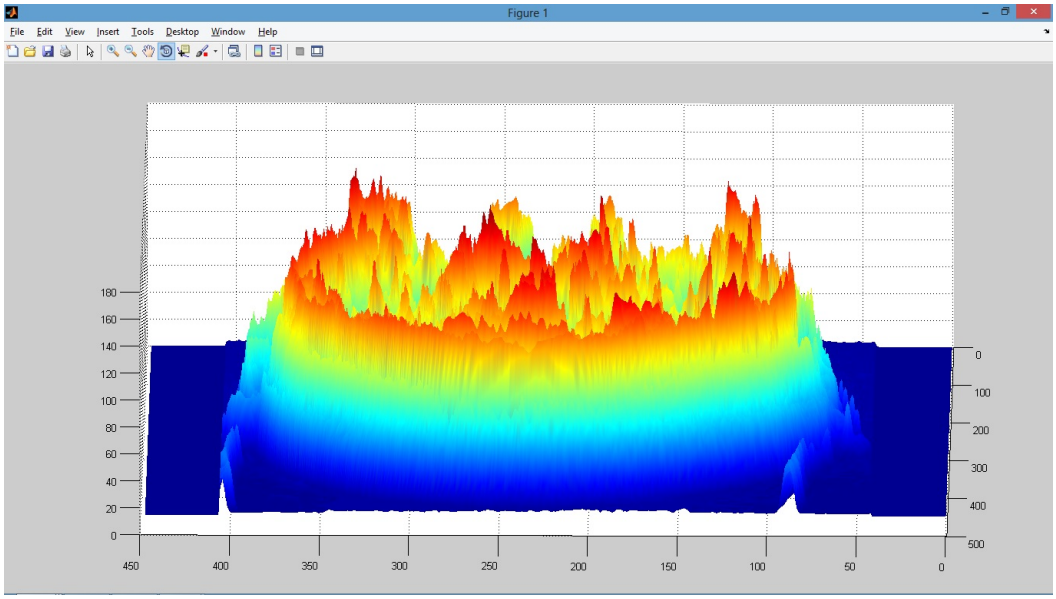


Figura 4.28: Gráfica de superficie de la imagen fusionada.

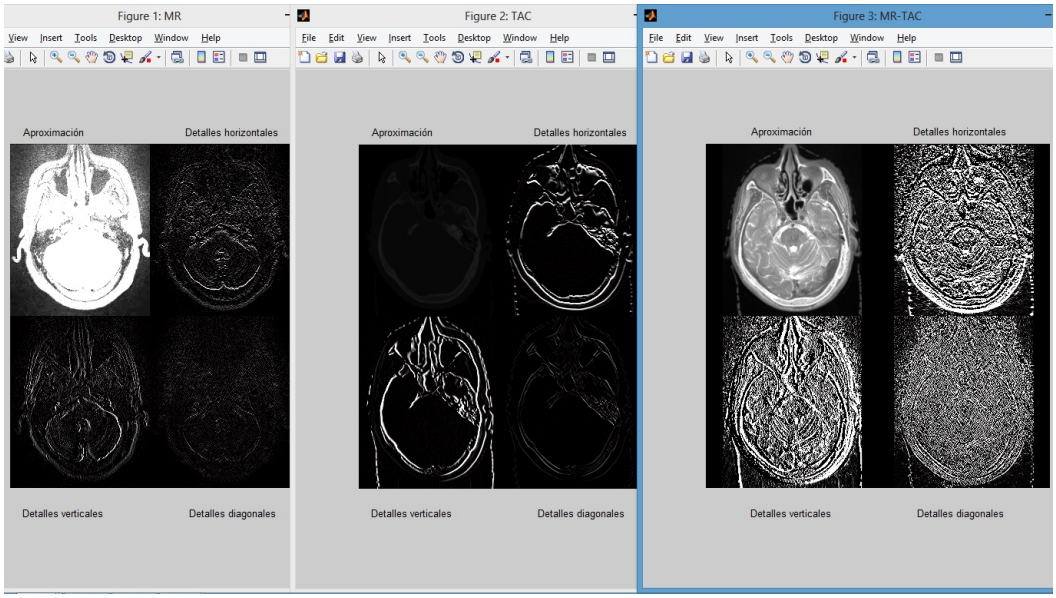


Figura 4.29: Gráficas de la descomposición wavelet de las tres imágenes.

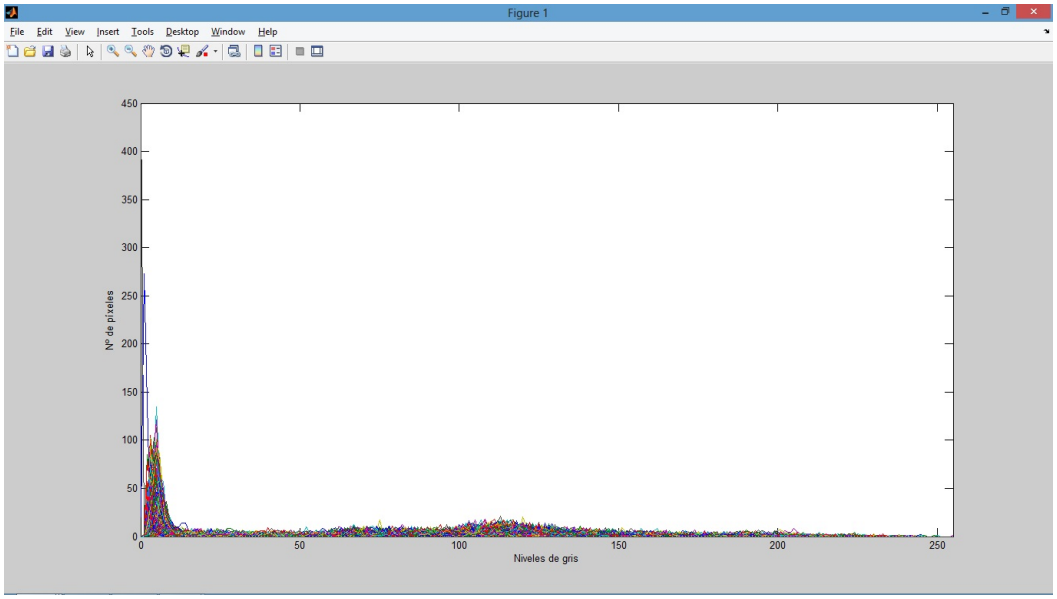


Figura 4.30: Gráfica del histograma de la imagen fusionada.

Cuando se quiera cerrar la aplicación basta con clicar sobre el aspa como en cualquier aplicación conocida o en *Archivo-Salir*. Ambas opciones mostrarán un mismo mensaje de confirmación de abandono de la aplicación. Basta con confirmar y se habrá cerrado satisfactoriamente.

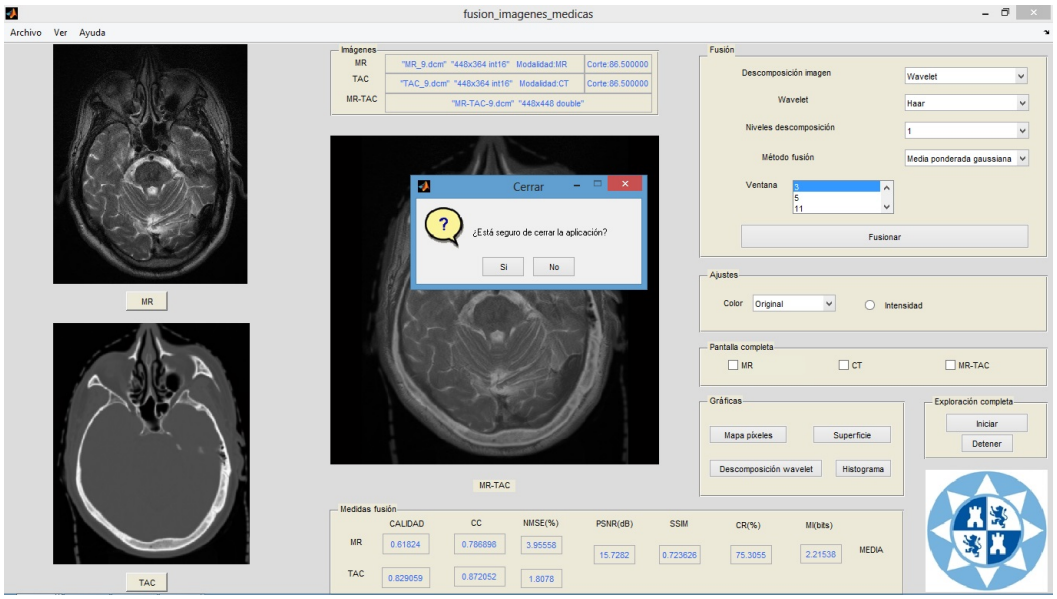


Figura 4.31: Mensaje de confirmación de cierre de la aplicación.

Capítulo 5

Evaluación y clasificación de los algoritmos

5.1. Valoración de un profesional en diagnóstico por imagen

Para poder hacer una valoración de este trabajo desde un punto de vista clínico, se solicitó la colaboración de un médico especialista en la materia, y fue el Doctor Antonio Lozano el que tuvo la amabilidad de dar su opinión sobre los resultados de los algoritmos en los distintos tipos de fusión. El Dr. Lozano es oncólogo y actualmente trabaja en el servicio de Radioterapia del Hospital Virgen de la Arrixaca de Murcia. Su trabajo consiste básicamente en planificar los tratamientos de Radioterapia que se administran a los pacientes, y para ello se apoya en imágenes por resonancia magnética, TAC y PET, y por tanto se considera su opinión de gran importancia en las valoraciones finales de este trabajo.

El Dr. Lozano considera de gran importancia el tratamiento de imágenes fusionadas, y en concreto y siempre que se hable desde el punto de vista del diagnóstico clínico, estima de gran utilidad las imágenes PET-TAC y MR-PET. Por el contrario, considera de poca o nula utilidad la fusión MR-TAC siempre que se busque elaborar un diagnóstico sobre la dolencia de un paciente. En su caso particular, utiliza a diario imágenes fusionadas MR-TAC únicamente para comprobar la correspondencia de los cortes MR y TAC que se obtuvieron por separado. Resalta asimismo la utilidad de las imágenes MR-TAC para la colaboración en radiocirugía.

De la fusión PET-TAC destaca su imprescindible aportación actual para el diagnóstico en oncología. Expuso un ejemplo práctico en el que se examina en primer lugar un TAC en el que se aprecian ganglios con un tamaño por encima de un centímetro de diámetro, y que por tanto se podrían considerar patológicos, si bien quizá no se atrevería a hacer un diagnóstico final. Es al observar la imagen conjunta PET-TAC y comprobar que en el ganglio sospechoso de malignidad hay captación de glucosa contenida en el radiofármaco FDG del PET, y por tanto existe actividad metabólica, cuando se puede realizar un diagnóstico certero y concluir que existe una masa tumoral en dicho ganglio. El Dr. Lozano considera que el PET-TAC tiene un potencial de desarrollo enorme en la actualidad, y de hecho es algo de lo que se habla mucho en los congresos a los que él asiste.

Por otro lado cree que la imagen MR-PET también es una gran herramienta de diagnóstico y resalta que ya existen equipos médicos híbridos MR-PET, aunque de momento no disponen de ellos. Advierte que en buena parte de los cortes realizados a nivel cerebral, la MR-PET sirve de escasa ayuda puesto que a ese nivel anatómico existen múltiples captaciones de glucosa y por tanto el PET muestra gran cantidad de pequeñas masas en las que considera que existe actividad metabólica. Para otras partes anatómicas sí que considera que es una gran herramienta de diagnóstico y a la que augura también una capacidad de desarrollo importante.

Una vez ya delante de la aplicación y preguntado por los algoritmos de fusión desarrollados, el Dr. responde:

En la fusión MR-TAC, no aprecia diferencias visuales relevantes en la imagen fusionada obtenida por los algoritmos *media aritmética* (ya sea en el dominio espacial o wavelet) y *media ponderada gaussiana*. Resalta una mejoría evidente para el caso de la *varianza* y el *coeficiente de variación* (ambos dan para él resultados similares) así como para el *nivel de actividad* calculado tanto sobre una ventana de coeficientes como sobre toda la imagen, provocada dicha mejoría por una ganancia en el nivel de detalle. La *media de imágenes multirresolución* también gana en nivel de detalle aunque parece perder en cuanto a resolución, pues aparece la imagen fusionada como más pixelada. Destaca que si bien se gana en nivel de detalle con los últimos cinco algoritmos, el contraste que hay entre colores hipo-densos e híper-densos, que es lo que se busca en MR-TAC, cambian mínimamente. Aunque no se decanta por uno en concreto, parece que se fija más en el caso de *varianza* y *nivel de actividad ventana*que en el resto.

Para la fusión MR-PET se repiten las valoraciones, aunque en este caso resalta que los tres primeros cortes disponibles en la aplicación no servirían para hacer un diagnóstico por lo anteriormente comentado de la múltiple captación de glucosa. El último corte por el contrario sí que serviría, de hecho comenta la existencia clara de un tumor cerebral. En la resonancia aparece una gran masa con dos tonalidades, una blanca y otra de un tono más grisáceo. En el PET se aprecia una zona de actividad metabólica que rodea a una masa de color negro. Dicha masa corresponde a una necrosis, es decir, tejido muerto que el propio desarrollo del tumor ha destruido, luego se intuye que el comienzo de la actividad tumoral no es reciente. La masa negra del PET coincide con la masa de tono blanco de la resonancia (la citada necrosis), y la zona de captación de glucosa que muestra el PET coincide con la masa gris de la resonancia, que correspondería con la actual masa tumoral.

La imagen final fusionada MR-PET en este corte ayudaría en la correcta delimitación de la zona tumoral para su posterior sometimiento a radioterapia si es que se precisara, descartando lo que en un principio en la resonancia se podría confundir con masa tumoral y que posteriormente el PET y la MR-PET lo confirman como tejido muerto.

Se muestra a continuación el corte MR-PET al que se hace referencia:

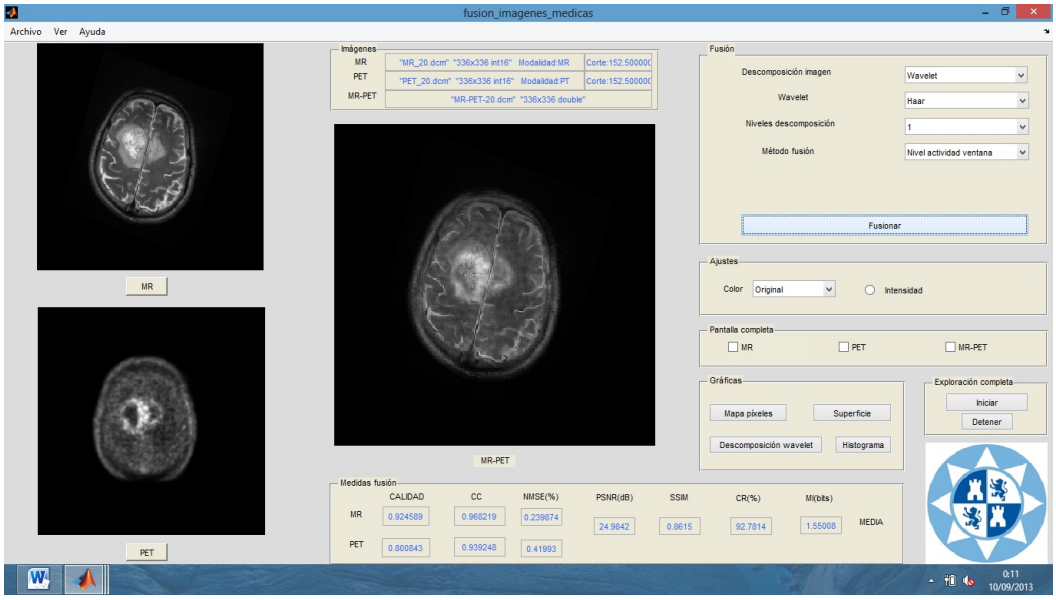


Figura 5.1: Fusión del cuarto corte disponible en MR-PET.

Para la fusión PET-TAC, el Dr. dictamina un empate de nuevo entre la *media aritmética* y la *media ponderada gaussiana*, y una mejora leve en los siguientes algoritmos, los cuales delimitan a su entender mejor en la imagen fusionada la zona anatómica que rodea a la masa donde según el PET hay actividad metabólica. Por el contrario, apenas hay cambios de unos algoritmos a otros en la delimitación de la zona de captación del PET. El Doctor afirma para terminar que donde menos se aprecian los cambios visuales entre los distintos algoritmos es en la fusión PET-TAC.

5.2. Obtención de medidas numéricas

Tal como se comenta al comienzo de este documento, para poder evaluar los distintos algoritmos de fusión de una manera cuantitativa, se han calculado una serie de medidas numéricas que en función de distintas evaluaciones pretenden hacer ver qué técnicas de fusión ofrecen unos mejores resultados frente a otras.

Se han calculado a partir de unos parámetros sobre los que conviene especificar qué indican en cada caso.

- X y Z serían las imágenes original y fusionada.
- σ_{XZ} es la covarianza entre X y Z.
- m_X y m_Z son los valores medios de los píxeles de ambas imágenes.
- σ_X^2 y σ_Z^2 son las varianzas de X y Z respectivamente.
- X_i y Z_i son los píxeles i-ésimos de las imágenes original X y fusionada Z respectivamente.
- p es el número de píxeles totales de la imagen.
- MAX es el valor máximo que puede tomar un píxel de la imagen.

CALIDAD

Si se habla de fusión PET-TAC, cuando se compara la imagen fusionada con el TAC se habla de calidad espacial y cuando se compara con el PET de calidad espectral. Lo mismo se podría decir en el caso de la fusión MR-PET. Si la comparación es fusionada con resonancia se trata de calidad espacial, mientras que la comparativa fusionada-PET habla de calidad espectral. En el caso de fusión MR-TAC no se puede decir lo mismo, puesto que tanto la resonancia magnética como el TAC son imágenes de buena resolución espacial.

Para calcular su valor se utiliza la siguiente expresión:

$$Q(X,Z) = \frac{4\sigma_{XZ} m_X m_Z}{(\sigma_X^2 + \sigma_Z^2)(m_X^2 + m_X^2)}$$

El rango de posibles valores que puede tomar va desde 0 hasta 1.

CC

Correlation Coefficient o coeficiente de correlación. Es una medida estadística que cuantifica la relación entre dos señales.

Obedece a la expresión:

$$CC(X,Z) = \frac{\sum_{i=1}^{p}(X_i - m_X)(Z_i - m_Z)}{(\sum_{i=1}^{p}(X_i - m_X)^2 \sum_{i=1}^{p}(Z_i - m_Z)^2)^{1/2}}$$

Los resultados serán mejores cuanto más se aproximen a 1.

NMSE

Normalised Mean Square Error o error cuadrático medio normalizado. Cuantifica la diferencia existente entre la señal deseada o esperada y la obtenida. A partir de la expresión del error cuadrático medio

$$MSE = \sum_{i=1}^{p} |Z_i - X_i|^2$$

se normaliza y se expresa en tanto por ciento. Será mejor cuanto menor sea su valor.

PSNR

Peak Signal-to-Noise Ratio o Relación Señal a Ruido. Define la relación existente entre la energía máxima de una señal y el ruido que afecta a su correcta representación. Su unidad de medida es el decibelio, siendo mejor cuanto mayor es su valor.

Se calcula a partir de la expresión:

$$PSNR = 20 \lg_{10} \frac{MAX}{\sqrt{MSE}}$$

SSIM

Structural Similitary o similitud estructural. Mide el grado de similitud existente entre dos imágenes en base a la calidad de la imagen obtenida respecto de la imagen de partida.

El rango de posibles valores que puede tomar oscila entre 0 y 1.

CR

Correlation Ratio o ratio de correlación. Mide la relación existente entre la dispersión estadística dentro de un conjunto local de muestras y la dispersión existente en el conjunto global de todas las muestra de la imagen. Expresado en tanto por ciento.

MI

Mutual Information o información mutua. Mide la dependencia mutua entre las dos imágenes comparadas. Queda expresada en bits.

Para las tres últimas medidas no se detalla el modo de calcularlas, puesto que requieren de una explicación extensa.

Una vez descritos brevemente los parámetros que se van a calcular, se muestra a continuación la adquisición de los datos mediante unas tablas. La calidad, el coeficiente de correlación y el error cuadrático medio normalizado se han calculado sobre la imagen fusionada respecto de cada una de las imágenes de partida por separado, mientras que las otras medidas se han calculado haciendo un promedio. Las medidas aquí expuestas son un resumen que se ha hecho comparando los diversos algoritmos sólo respecto a un corte de cada uno de los distintos tipo de fusión a realizar, puesto que en caso contrario se haría tedioso de leer y alargaría innecesariamente en exceso la extensión de esta memoria.

Para la toma de datos se han obviado el segundo y tercer nivel en la descomposición wavelet, puesto que ofrecen resultados claramente peores que el primer nivel, tanto visualmente para un examen diagnóstico como en las meras medidas numéricas. Para la descomposición wavelet se ha optado por la de Haar, puesto que el resto no ofrece variaciones numéricas.

Para las medidas de la varianza y el coeficiente de variación sobre una ventana de coeficientes, y tras comprobar que los datos fluctúan más al variar el tamaño de ventana que al modificar el umbral, se exponen los resultados para los distintos tamaños de ventana disponibles pero todos sobre un mismo umbral, con el objetivo de no alargar el número de tablas con resultados en exceso.

MEDIDAS SOBRE LA FUSIÓN PET-TAC

Para el número de corte -67.75, y a partir de las imágenes TAC-020.v2, PET-020.v2 y PET-TAC-020.v2, se obtienen las siguientes medidas:

Media en el dominio espacial

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.825001	0.967457	3.1807	13.1131	0.473971	96.412	2.08727
PET	0.123846	0.64595	7.49653				

Media wavelet

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.825001	0.967457	3.1807	13.1131	0.473971	96.412	2.08727
PET	0.123846	0.64595	7.49653				

Media ponderada gaussiana (tamaño de ventana 3)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.833821	0.96726	2.88228	13.1897	0.476504	96.4429	2.04869
PET	0.120345	0.646335	7.98591				

Media ponderada gaussiana (tamaño de ventana 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.846563	0.965458	2.30205	13.3661	0.479153	96.4043	1.93618
PET	0.113202	0.647216	9.21862				

Media ponderada gaussiana (tamaño de ventana 11)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.840919	0.964316	2.62653	13.2553	0.478174	96.3929	1.90157
PET	0.11675	0.647347	8.50267				

Media ponderada gaussiana (tamaño de ventana 21)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.835902	0.964037	2.83267	13.1922	0.476769	96.3748	1.8758
PET	0.11904	0.647266	8.11646				

Media ponderada laplaciana

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.278262	0.878486	4.93439	9.71423	0.168133	83.0495	1.38924
PET	0.0607727	0.565358	23.1164				

Media imágenes multirresolución

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.811346	0.967979	1.49469	12.723	0.447251	95.3149	1.87541
PET	0.0842547	0.634557	19.0919				

Varianza sobre una ventana de coeficientes (tamaño de ventana 3 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.796767	0.96602	1.51874	12.8693	0.443632	95.5571	1.89608
PET	0.0897114	0.641017	17.5654				

Varianza sobre una ventana de coeficientes (tamaño de ventana 5 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.796765	0.966006	1.51877	12.8692	0.443632	95.5512	1.88998
PET	0.0897111	0.641021	17.5655				

Varianza sobre una ventana de coeficientes (tamaño de ventana 11 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.796761	0.965998	1.5188	12.8692	0.443631	95.5504	1.8831
PET	0.0897095	0.641011	17.5655				

Varianza sobre una ventana de coeficientes (tamaño de ventana 21 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.79676	0.966017	1.51879	12.8692	0.443629	95.5635	1.87952
PET	0.0897065	0.64098	17.5655				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 3 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.829268	0.966978	1.33345	13.4516	0.461537	96.2703	1.97127
PET	0.09379	0.645671	15.3005				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 5 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.830254	0.9665	1.43204	13.529	0.464084	96.2009	1.94049
PET	0.0983441	0.645461	13.7481				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 11 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.827462	0.965754	1.41613	13.4939	0.461965	96.0584	1.91267
PET	0.0972798	0.645005	14.1294				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 21 y umbral 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.804732	0.966274	1.5077	12.7688	0.445524	95.9849	1.89002
PET	0.0872644	0.644771	18.532				

Nivel de actividad sobre los coeficientes

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.796198	0.965908	1.52101	12.8721	0.443385	95.5498	1.87957
PET	0.0898941	0.641193	17.5165				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 3)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.795018	0.965497	1.52852	12.8621	0.442915	95.5216	1.88132
PET	0.0901314	0.642367	17.5106				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.794669	0.965387	1.53072	12.8592	0.442777	95.5332	1.88192
PET	0.0901976	0.64268	17.5089				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 11)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.815547	0.965039	1.41553	13.3142	0.455272	95.5811	1.89959
PET	0.0944422	0.643686	15.3547				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 21)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
TAC	0.795284	0.965619	1.52679	12.8644	0.443016	95.798	1.88088
PET	0.0900671	0.642011	17.512				

MEDIDAS SOBRE LA FUSIÓN MR-PET

En este caso, para los algoritmos de la varianza y el coeficente de variación se sube el valor del umbral a 30, puesto que parece que ofrece resultados ligeramente mejores. Para el número de corte 152.5, y a partir de las imágenes MR-20.dcm, PET-20.dcm y MR-PET-20.dcm, se obtienen las siguientes medidas:

Media en el dominio espacial

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.966519	0.967665	0.157503	26.5915	0.930257	93.6434	1.68705
PET	0.894246	0.94723	0.305082				

Media wavelet

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.966519	0.967665	0.157503	26.5915	0.930257	93.6434	1.68705
PET	0.894246	0.94723	0.305082				

Media ponderada gaussiana (tamaño de ventana 3)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.966337	0.967108	0.159775	26.5216	0.928976	93.6663	1.70384
PET	0.891566	0.947792	0.310572				

Media ponderada gaussiana (tamaño de ventana 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.963316	0.963645	0.177165	26.1115	0.921809	93.4225	1.65395
PET	0.879819	0.948859	0.338312				

Media ponderada gaussiana (tamaño de ventana 11)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.961185	0.961519	0.188229	25.9221	0.919511	93.1837	1.634
PET	0.877279	0.948593	0.347454				

Media ponderada gaussiana (tamaño de ventana 21)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.960975	0.961134	0.190639	25.8663	0.919522	93.1108	1.61488
PET	0.877878	0.948329	0.351987				

Media ponderada laplaciana

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.199528	0.878784	12.0258	9.05	0.181928	81.5525	1.06253
PET	0.166158	0.815667	12.8791				

Media imágenes multirresolución

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.927127	0.969694	0.226509	24.9069	0.855989	92.5593	1.53351
PET	0.790753	0.936022	0.460822				

Varianza sobre una ventana de coeficientes (tamaño de ventana 3 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.922525	0.969029	0.239259	24.8213	0.856114	92.8024	1.64528
PET	0.78983	0.938956	0.453813				

Varianza sobre una ventana de coeficientes (tamaño de ventana 5 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.922588	0.969082	0.238986	24.8203	0.856058	92.8007	1.63942
PET	0.78966	0.938765	0.454535				

Varianza sobre una ventana de coeficientes (tamaño de ventana 11 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.920733	0.969315	0.240918	24.6931	0.852077	92.7882	1.66778
PET	0.782467	0.938358	0.478094				

Varianza sobre una ventana de coeficientes (tamaño de ventana 21 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.943247	0.969593	0.194337	25.5562	0.879347	92.7332	1.60252
PET	0.819868	0.9378	0.3983				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 3 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.966326	0.967366	0.158781	26.4714	0.924047	93.5352	1.65554
PET	0.882318	0.946848	0.319826				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 5 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.964913	0.96711	0.161962	26.561	0.925297	93.479	1.68686
PET	0.885064	0.946531	0.300869				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 11 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.962935	0.966782	0.165904	26.4061	0.91884	93.3504	1.70791
PET	0.873864	0.945964	0.315452				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 21 y umbral 30)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.960732	0.966132	0.172185	26.3948	0.916091	93.3105	1.60388
PET	0.873745	0.946203	0.305521				

Nivel de actividad sobre los coeficientes

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.921881	0.968179	0.24242	24.7944	0.85583	92.7453	1.56388
PET	0.790003	0.939317	0.453481				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 3)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.918588	0.968219	0.247506	24.593	0.849725	92.7926	1.62771
PET	0.779574	0.939248	0.487326				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.915496	0.968519	0.252649	24.4049	0.843562	92.796	1.63728
PET	0.769955	0.938864	0.520589				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 11)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.922525	0.968982	0.239316	24.8084	0.855771	92.7317	1.56659
PET	0.789217	0.938269	0.456412				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 21)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.921566	0.969029	0.24049	24.7464	0.853888	92.7245	1.57885
PET	0.785814	0.938205	0.467326				

MEDIDAS SOBRE LA FUSIÓN MR-TAC

Se seleccionará aquí un valor de umbral intermedio de 15. En el siguiente apartado se profundiza más sobre la valoración de resultados en función del umbral. Para el número de corte 80.5, y a partir de las imágenes MR-7.dcm, TAC-7.dcm y MR-TAC-7.dcm, se obtienen las siguientes medidas:

Media en el dominio espacial

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.729379	0.776431	1.65382	18.7634	0.783644	72.5841	2.1117
TAC	0.837955	0.842315	1.06864				

Media wavelet

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.729379	0.776431	1.65382	18.7634	0.783644	72.5841	2.1117
TAC	0.837955	0.842315	1.06864				

Media ponderada gaussiana (tamaño de ventana 3)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.729068	0.775205	1.64888	18.786	0.783868	72.6016	2.00876
TAC	0.838723	0.843151	1.06073				

Media ponderada gaussiana (tamaño de ventana 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.725176	0.768924	1.6464	18.8185	0.781959	71.9899	1.68435
TAC	0.838775	0.844106	1.04655				

Media ponderada gaussiana (tamaño de ventana 11)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.719484	0.766217	1.70369	18.7096	0.778494	71.6425	1.60401
TAC	0.837516	0.842706	1.06337				

Media ponderada gaussiana (tamaño de ventana 21)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.714513	0.765552	1.77632	18.5777	0.775971	71.5909	1.59107
TAC	0.837449	0.842048	1.08375				

Media ponderada laplaciana

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.289514	0.678423	9.41038	10.7458	0.300942	51.4578	0.840213
TAC	0.312163	0.664626	7.53765				

Media imágenes multirresolución

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.715964	0.783939	1.69924	18.3961	0.756359	70.9096	1.5784
TAC	0.796727	0.822691	1.23176				

Varianza sobre una ventana de coeficientes (tamaño de ventana 3 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.708743	0.777266	1.7564	18.4117	0.757664	71.0049	1.65012
TAC	0.806575	0.832078	1.18317				

Varianza sobre una ventana de coeficientes (tamaño de ventana 5 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.70813	0.776598	1.75962	18.4138	0.757635	71.0463	1.65191
TAC	0.807123	0.83266	1.17982				

Varianza sobre una ventana de coeficientes (tamaño de ventana 11 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.708262	0.776792	1.75772	18.4141	0.757524	70.9902	1.6505
TAC	0.806771	0.832493	1.18094				

Varianza sobre una ventana de coeficientes (tamaño de ventana 21 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.708897	0.777547	1.7529	18.4145	0.757488	71.1136	1.64859
TAC	0.806065	0.831997	1.18399				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 3 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.72334	0.775437	1.68552	18.7019	0.778293	72.2688	1.81937
TAC	0.833298	0.841228	1.07866				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 5 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.717385	0.774814	1.70846	18.6447	0.772131	71.9893	1.74113
TAC	0.826927	0.840605	1.09261				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 11 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.694373	0.77363	2.01051	18.0759	0.756278	71.5376	1.67276
TAC	0.818214	0.838964	1.20644				

Coeficiente de variación sobre una ventana de coeficientes (tamaño de ventana 21 y umbral 15)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.690237	0.773	1.93985	18.1128	0.745969	71.0291	1.63102
TAC	0.801741	0.837452	1.22936				

Nivel de actividad sobre los coeficientes

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.708206	0.776575	1.76185	18.4057	0.757611	70.7679	1.60182
TAC	0.807004	0.832103	1.18272				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 3)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.719503	0.776053	1.54427	18.7735	0.760559	70.8941	1.59554
TAC	0.801647	0.832514	1.13916				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 5)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.720131	0.776797	1.54035	18.7715	0.76047	70.9372	1.59486
TAC	0.800845	0.83185	1.14308				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 11)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.711138	0.779971	1.74155	18.3978	0.75725	70.9481	1.59355
TAC	0.803338	0.829049	1.2009				

Nivel de actividad sobre una ventana de coeficientes (tamaño de ventana 21)

	CALIDAD	CC	NMSE	PSNR	SSIM	CR	MI
MR	0.712913	0.781978	1.73057	18.393	0.757174	71.1145	1.5901
TAC	0.801407	0.827291	1.21121				

5.3. Observaciones y ranking clasificatorio final

A la vista de los resultados se pueden ir extrayendo unas primeras conclusiones sobre el trabajo que se ha desarrollado:

- Cuando se habla de fusionar imágenes mediante algoritmos wavelet en el contexto de la imagen médica, no interesa profundizar más allá del primer nivel en la descomposición wavelet, puesto que los resultados empeoran significativamente según se va pasando al siguiente nivel. En este estudio se ha llegado hasta el tercer nivel de descomposición, si bien sólo en el primero se han obtenido unos resultados útiles para un exámen diagnóstico.

- Aplicar la transformada wavelet discreta en dos dimensiones a la hora de descomponer la imagen con familias wavelet distintas a la de Haar no aporta ni perjuicio ni beneficio alguno en los resultados.

- Calcular la media aritmética sobre coeficientes wavelet no aporta ninguna ventaja respecto de dicho promedio calculado en el dominio espacial directamente sobre los niveles de intensidad de gris. A pesar de ser un cálculo sencillo y que pudiera parecer un algoritmo muy básico, los resultados tanto numéricos como visuales no han sido malos en absoluto.

- La media ponderada gaussiana se calcula ponderando con una ventana de pesos gaussiana, lo que deja paso a las bajas frecuencias y da pues como resultado una imagen visualmente "suavizada", por decirlo de alguna manera. Es el algoritmo que parece ofrecer menos nivel de detalle, si bien los resultados numéricos han sido bastante buenos para los tres tipos de fusión.

- La media ponderada laplaciana arroja unos resultados muy pobres, de hecho son los peores del grupo con mucha diferencia. Ésto se debe a que al actuar la matriz de pesos laplaciana como un filtro paso alto, respeta las altas frecuencias, que no son otra cosa que cambios abruptos de los niveles de intensidad, lo que provoca por tanto un realce de los bordes de la imagen. Para otros campos del procesado de imagen puede ser de gran utilidad, pero desde luego no para un estudio de fusión de imágenes médicas.

- Calcular la media entre la matriz de aproximación en el dominio wavelet de una de las imágenes junto con la matriz original de la otra imagen en el dominio espacial ofrece valores peores que el cálculo de la media aritmética con las dos imágenes en un mismo dominio.

- Se ha comprobado cómo los valores de las varianzas y por consiguiente las diferencias entre ellas varían mucho dependiendo de dónde esté situada la ventana que va recorriendo la matriz, puesto que si por ejemplo la ventana se encuentra en una de las imágenes en una zona donde todos los píxeles tienen el mismo valor (y por tanto la varianza es nula), mientras que en la otra imagen hay cambios bruscos de intensidad (lo que da una varianza elevada), la diferencia entre varianzas ofrece un resultado alto y que queda muy por encima del umbral. Luego las diferencias entre varianzas fluctúan de una manera muy acusada y en general este método de fusión es poco sensible al umbral establecido, sea cual sea su valor.

- El coeficiente de variación en cambio ofrece valores más próximos entre ambas imágenes, y sobre todo más homogéneos durante el recorrido de la ventana, y es mucho más sensible a los cambios de umbral. Los mejores valores numéricos se obtienen para valores de umbral altos, si bien cabe destacar que aumentando el nivel del umbral se está provocando un mayor número de cálculos de la media aritmética entre los coeficientes centrales de la ventana en detrimento de la selección del píxel central de la ventana con mayor coeficiente de variación.
- No existen variaciones significativas en los datos obtenidos cuando se comparan los cálculos del nivel de actividad sobre la imagen completa o sobre una ventana de coeficientes.

FUSIÓN PET-TAC

La gran diferencia de este tipo de fusión respecto de las otras radica en el salto existente entre los valores de calidad espacial y espectral, siendo muy buena la primera y muy baja la segunda.

Para la fusión PET-TAC se puede apreciar cómo la media aritmética tanto para el dominio espacial como wavelet arroja exactamente los mismos resultados. La media ponderada gaussiana mejora en una pequeña proporción los valores de la media aritmética, y parece que los valores mejoran en general muy ligeramente para un tamaño de ventana intermedio, justo donde parece aumentar la calidad espacial mientras disminuye la espectral. La media de imágenes multirresolución empeora mínimamente los resultados de la media aritmética, con un NMSE respecto al PET particularmente por encima del resto.

El cálculo de la varianza sobre una ventana de coeficientes se aproxima mucho a los resultados dados por la media de imágenes multirresolución, destacando además una afectación casi nula a los cambios de ventana y umbral. El coeficiente de variación mejora claramente los resultados para el mayor valor de umbral de los disponibles y un tamaño de ventana intermedio, colocándose bastante parejo a la media aritmética, pues la mejora en algunos datos y la empeora en otros.

El nivel de actividad sobre los coeficientes quedaría empatado junto con la varianza, mientras que el nivel de actividad basado en ventana quedaría justo por debajo del coeficiente de variación. Luego para un algoritmo de fusión de imagen que se base en cálculos del nivel de actividad, mejor hacerlo sobre una ventana de coeficientes que sobre toda la matriz.

Así pues, se pueden listar los algoritmos por orden de mejor a peor para la fusión PET-TAC según la siguiente relación:

1. Media ponderada gaussiana (en particular para tamaños de ventana 5 y 11).
2. Media aritmética (espacial y wavelet) junto con el coeficiente de variación (para ventana 5 y umbral 30).
3. Nivel de actividad basado en ventana (mejor con ventana 11).
4. Media de imágenes multirresolución, varianza y nivel de actividad basado en coeficientes.

5. Media ponderada laplaciana.

FUSIÓN MR-PET

Los mejores resultados numéricos en su conjunto se han obtenido claramente para este tipo de fusión, puesto que se alcanza un gran compromiso entre las calidades espacial y espectral y los coeficientes de correlación, el error cuadrático medio normalizado es muy bajo, mientras que la similitud estructural y la PSNR ofrecen los mejores valores cuando se habla de fusión MR-PET.

El ranking obtenido para la fusión MR-PET ordenado de mejor a peor es el siguiente:

1. Coeficiente de variación (ventana 3 y 5, umbral 30).
2. Media aritmética (espacial y wavelet) y media ponderada gaussiana(ventana 3).
3. Varianza sobre ventana de coeficientes (ventana 21).
4. Media imágenes multirresolución.
5. Nivel de actividad basado en coeficientes y basado en ventana (ventana 11 y 21).
6. Media ponderada laplaciana.

FUSIÓN MR-TAC

En este caso no se puede hablar de calidad espacial y espectral puesto que ambas imágenes atienden a una buena resolución espacial. El parámetro calidad ofrece mejores resultados cuando se compara la imagen fusionada con el TAC respecto a cuando se compara con la resonancia magnética.

La media ponderada gaussiana ofrece unos valores muy similares a la media aritmética, y es muy poco sensible a los cambios en el tamaño de ventana. Si acaso los valores son algo mejores para la ventana más pequeña. La media en imágenes multirresolución empeora los datos de las dos últimas, aunque queda justo por delante de la varianza, que para el caso de la combinación MR-TAC muestra una insensibilidad a la variación del tamaño de la ventana. El coeficiente de variación empeora sus resultados según se aumenta el tamaño de ventana y se disminuye el umbral, dando sus mejores resultados para una ventana de tamaño 3 y un umbral de 30. En esa configuración de parámetros, queda justo por debajo de las medias aritmética y ponderada gaussiana.

La lista de algoritmos ordenada de mejor a peor para una fusión MR-TAC queda como sigue:

1. Media aritmética (espacial y wavelet) y media ponderada gaussiana (tamaño de ventana 3).
2. Coeficiente de variación (ventana 3, umbral 30).
3. Nivel de actividad sobre una ventana de coeficientes (ventanas 3 y 5).

4. Media en imágenes multirresolución.
5. Varianza sobre una ventana de coeficientes.
6. Nivel de actividad basado en coeficientes.
7. Media ponderada laplaciana.

Capítulo 6

Conclusiones

Una vez extraídas unas conclusiones tanto subjetivas como objetivas durante la evaluación de los distintos algoritmos de fusión de imagen implementados en este trabajo, se puede hacer un juicio diagnóstico de los mismos de una forma ciertamente fiable.

Si bien el Doctor especializado en oncología que colaboró dando su opinión lo valoró positivamente, el algoritmo que basa sus decisiones en base al nivel de actividad de los coeficientes de las imágenes no ofrece resultados numéricos especialmente buenos. Los algoritmos de fusión que calculan la varianza y el coeficiente de variación daban unos resultados asimismo buenos a su entender, mientras que en las medidas es el segundo el que queda mejor colocado.

La media de imágenes mutirresolución ofrece algo más de nivel de detalle que la media aritmética, sobre todo en la fusión MR-TAC, aunque ambas las colocara el facultativo bastante parejas y algo por detrás de las anteriormente comentadas. En obtención de medidas queda por delante sin embargo la media aritmética respecto de la media de imágenes multirresolución.

La media ponderada gaussiana ofrece buenos resultados numéricos, aunque visualmente en general pierde nivel de detalle. La media ponderada laplaciana queda totalmente descartada y no es una buena herramienta para este tipo de fusión de imagen.

Luego por tanto, en cómputo global y atendiendo a la valoración del experto y en virtud de los resultados numéricos obtenidos, y siempre bajo unas diferencias mínimas, se puede elegir por su equilibrado compromiso entre resultados objetivos y subjetivos, el algoritmo que calcula el coeficiente de variación sobre una ventana de coeficientes para el caso particular de un tamaño de ventana de 3 y un umbral de 30.

Por otra parte, dentro del ámbito clínico, cuando se habla de diagnóstico por imagen, interesa que se profundice en los estudios de fusiones PET-TAC y MR-PET. Más concretamente, interesaría perfeccionar al máximo la fusión PET-TAC de forma que permita realizar el mejor diagnóstico posible en base a la información conjunta anatómico-metabólica. Mientras que en la fusión MR-PET un objetivo cla-

ro sería definir lo mejor posible en la imagen fusionada la masa tumoral mostrada por la resonancia magnética junto con la zona de actividad metabólica captada por el PET dentro de dicha masa y sus alrededores, de forma que permitiera distinguir entre masa tumoral actual, zona de expansión del tumor y tejido muerto si es que lo hubiera. Ésto ultimo permite al facultativo tanto elaborar un diagnóstico como delimitar la zona a tratar en la planificación de radioterapia. De la fusión MR-TAC no se habla de momento en el ámbito del radiodiagnóstico, mientras que sí parece de utilidad para los trabajos en radiocirugía.

Volviendo a la temática de la fusión de imágenes, sería interesante el abordar un estudio similar al que aquí nos ocupa, pero sobre una familia de algoritmos de más reciente aparición denominados curvelet, los cuales realizan una descomposición de la imagen más minuciosa y parece que según algún estudio publicado en el que se calcula la fusión a partir de la medición del nivel de actividad, mejora los resultados que se obtienen con algoritmos de fusión wavelet.

Bibliografía

[1] Michael Y. M. Chen, Thomas L. Pope Jr., David J.Ott, *Radiología básica. Editorial McGraw Hill Interamericana.*

[2] B. Kastler, D. Vetter, A. Gangi, *Principios de RM, manual de autoaprendizaje. Editorial Masson.*

[3] Miguel Alcaraz Baños, *Bases físicas y biológicas del radiodiagnóstico médico. Universidad de Murcia.*

[4] César S. Pedrosa, Diagnóstico por imagen. Tratado de radiología clínica. Editorial McGraw Hill Interamericana.

[5] Agencia de Evaluación de Tecnologías Sanitarias, *Tomografía por emisión de positrones con 18FDG en oncología clínica. I.E.T.S Nº 30, Instituto de Salud Carlos III.*

[6] Paul Gerhardt, Walter Frommhold , *Atlas de correlaciones anatómicas en TAC y RMN. Salvat.*

[7] María Moncayo y Juan Francisco Reinoso, Análisis multirresolución y fusión de imágenes. Métodos numéricos y su relación con modelos matemáticos. Universidad Politécnica de Cartagena. Universidad de Granada.

[8] Lázaro José Guillén Fernández, María Moncayo y Juan Francisco Reinoso, *Fusión de imágenes mediante algoritmos lineales de multirresolución en paralelepípedo. Universidad Politécnica de Cartagena [proyecto fin de carrera].*

[9] Francisca María Lucas Martínez, Juan Carlos Trillo Moya, *Sistemas de funciones iteradas y aplicaciones. Universidad Politécnica de Cartagena [proyecto fin de carrera].*

[10] Rafael Verdú Monedero, Jorge Larrey Ruíz, Bonifacio Tobarra González, Antonio González López *Procesado de Señales e Imágenes Biomédicas y Aplicaciones. Grupo de Teoría y Tratamiento de la Señal. Universidad Politécnica de Cartagena. Hospital universitario Virgen de la Arrixaca.*

[11] Rafael Verdú Monedero, Juan Morales Sánchez, *Tratamiento Digital de la Señal. Universidad Politécnica de Cartagena.*

[12] Rafael Verdú Monedero, José Luis Gómez Tornero, *Desarrollo de aplicaciones con interfaces gráficas en Matlab. Universidad Politécnica de Cartagena.*

[13] Gonzalo Pajares y Jesús Manuel de la Cruz, *A wavelet-based image fusion tutorial. Departamento de Arquitectura de Computadores y Automática. Facultad de Ciencias Físicas. Universidad Complutense de Madrid.*

[14] C.Y. Wen, J.K. Chen, *Multi-resolution image fusion technique and its application to forensic science . Department of Forensic Science, Central Police University Taiwan.*

[15] Vinay K. Ingle, John G. Proakis, *Digital Signal Processing using MATLAB. Second Edition. Editorial Thomson.*

Códigos MATLAB

----------Interfaz gráfica de usuario----------

fusion_imagenes_medicas.m

```
function varargout = fusion_imagenes_medicas(varargin)
% FUSION_IMAGENES_MEDICAS MATLAB code for fusion_imagenes_medicas.fig
%      FUSION_IMAGENES_MEDICAS, by itself, creates a new
FUSION_IMAGENES_MEDICAS or raises the existing
%      singleton*.
%
%      H = FUSION_IMAGENES_MEDICAS returns the handle to a new
FUSION_IMAGENES_MEDICAS or the handle to
%      the existing singleton*.
%
%      FUSION_IMAGENES_MEDICAS('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in FUSION_IMAGENES_MEDICAS.M with the
given input arguments.
%
%      FUSION_IMAGENES_MEDICAS('Property','Value',...) creates a new
FUSION_IMAGENES_MEDICAS or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before fusion_imagenes_medicas_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to fusion_imagenes_medicas_OpeningFcn
via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
fusion_imagenes_medicas

% Last Modified by GUIDE v2.5 12-Jul-2013 02:00:14

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @fusion_imagenes_medicas_OpeningFcn,
...
```

```
                   'gui_OutputFcn',  @fusion_imagenes_medicas_OutputFcn,
...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before fusion_imagenes_medicas is made visible.
function fusion_imagenes_medicas_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to fusion_imagenes_medicas (see
VARARGIN)

% Choose default command line output for fusion_imagenes_medicas
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

%Visualizar logo UPCT
img=imread('background.jpg');
imshow(img,'Parent',handles.axes4);

%Idioma por defecto español
if isempty(get(handles.English,'UserData')) &&
isempty(get(handles.Deutsch,'UserData'))
    set(handles.Espanol,'UserData',1);
end

%Imagen en pantalla completa por defecto no visible
set(handles.axes6,'Visible','off');

%Color de la fuente igual al resto por defecto
set(handles.editInfopettac,'ForegroundColor',[.2 .4 1]);

set(handles.botonIniciarexploracion,'UserData',1);

%Cargar por defecto estudio PET-TAC
if isempty(get(handles.cargarRMTAC,'UserData')) &&
isempty(get(handles.cargarRMPET,'UserData'))

    set(handles.cargarPETTAC,'UserData',1);
```

```
    set(handles.desplegableNiveles,'Visible','off');
    set(handles.desplegableMetodo,'Visible','off');
    set(handles.desplegableTipowavelet,'Visible','off');
    set(handles.listboxVentana,'Visible','off');
    set(handles.listboxUmbral,'Visible','off');
    set(handles.textoWavelet,'Visible','off');
    set(handles.textoNiveles,'Visible','off');
    set(handles.textoMetodofusion,'Visible','off');
    set(handles.textoWavelet,'Visible','off');
    set(handles.textoVentana,'Visible','off');
    set(handles.textoUmbral,'Visible','off');

end

% UIWAIT makes fusion_imagenes_medicas wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = fusion_imagenes_medicas_OutputFcn(hObject,
eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in botonPET.
function botonPET_Callback(hObject, eventdata, handles)
% hObject    handle to botonPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

try
    if get(handles.cargarPETTAC,'UserData')==1
        [filenameY,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\BETH_ISRAEL_CENTER\PET');
    elseif get(handles.cargarRMTAC,'UserData')==1
        if get(handles.Paciente_1,'UserData')==1
            [filenameY,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\TUMOR
CEREBRAL(TAC+PET+RM)\Pares RM-TAC registrados');
        elseif get(handles.Paciente_2,'UserData')==1
            [filenameY,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\ARRIXACA\MR_TAC
registrados');
        end
    elseif get(handles.cargarRMPET,'UserData')==1
```

```
        [filenameY,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\TUMOR
CEREBRAL(TAC+PET+RM)\Pares RM-PET registrados');
    end

    if isequal(filenameY,0)
        %Do nothing yet
    else
        handles.myImage=dicomread(fullfile(pathname,filenameY));

        infoY=dicominfo(filenameY);
        modalidadY=infoY.Modality;

        if get(handles.cargarRMTAC,'UserData')==1
            if strcmp(modalidadY,'CT')~=1
                if get(handles.Espanol,'UserData')==1
                    msgbox('Debe seleccionar un TAC.');
                    return
                elseif get(handles.English,'UserData')==1
                    msgbox('Please open a CT image.');
                    return
                elseif get(handles.Deutsch,'UserData')==1
                    msgbox('Bitte auswählen Sie ein Bild CT.');
                    return
                end
            end
        else
            if strcmp(modalidadY,'PT')~=1
                if get(handles.Espanol,'UserData')==1
                    msgbox('Debe seleccionar un PET.');
                    return
                elseif get(handles.English,'UserData')==1
                    msgbox('Please open a PET image.');
                    return
                elseif get(handles.Deutsch,'UserData')==1
                    msgbox('Bitte auswählen Sie ein Bild PET.');
                    return
                end
            end
        end

        filasY=infoY.Rows;
        columnasY=infoY.Columns;
        bitsY=infoY.BitDepth;
        corteY=infoY.SliceLocation;

        if get(handles.Espanol,'UserData')==1
            infoCortey=sprintf('Corte:%f',corteY);
            infoPET=sprintf('"%s"  "%dx%d int%d"
Modalidad:%s',filenameY,filasY,columnasY,bitsY,modalidadY);
        elseif get(handles.English,'UserData')==1
            infoCortey=sprintf('Slice:%f',corteY);
            infoPET=sprintf('"%s"  "%dx%d int%d"
Modality:%s',filenameY,filasY,columnasY,bitsY,modalidadY);
         elseif get(handles.Deutsch,'UserData')==1
            infoCortey=sprintf('Schnitt:%f',corteY);
```

```
            infoPET=sprintf('"%s"  "%dx%d int%d"
Modalität:%s',filenameY,filasY,columnasY,bitsY,modalidadY);
        end

        Y=double(handles.myImage);
        Y=Y-min(Y(:));
        Y=Y/max(Y(:))*255;

        imshow(Y/255,'Parent',handles.axes2);

        set(handles.editInfopet,'UserData',filenameY);
        set(handles.textoInfopet,'UserData',pathname);

        set(handles.editInfopet,'String',infoPET);
        set(handles.editCortey,'String',infoCortey);

    end
    guidata(hObject,handles);
catch
    msgbox('Error');

end



% --- Executes on button press in botonTAC.
function botonTAC_Callback(hObject, eventdata, handles)
% hObject    handle to botonTAC (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

try
    if get(handles.cargarPETTAC,'UserData')==1
        [filenameX,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\BETH_ISRAEL_CENTER\TAC');
    elseif get(handles.cargarRMTAC,'UserData')==1
        if get(handles.Paciente_1,'UserData')==1
            [filenameX,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\TUMOR
CEREBRAL(TAC+PET+RM)\Pares RM-TAC registrados');
        elseif get(handles.Paciente_2,'UserData')==1
            [filenameX,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\ARRIXACA\MR_TAC
registrados');
        end
    elseif get(handles.cargarRMPET,'UserData')==1
        [filenameX,pathname]=uigetfile('*.*','Selecciona imagen para
abrir','C:\Users\antonio\Documents\PFC\IMAGENES\TUMOR
CEREBRAL(TAC+PET+RM)\Pares RM-PET registrados');
    end

    if isequal(filenameX,0)
        %Do nothing yet
    else
        handles.myImage=dicomread(fullfile(pathname,filenameX));
```

```
        infoX=dicominfo(filenameX);
        modalidadX=infoX.Modality;

        if get(handles.cargarPETTAC,'UserData')==1
            if strcmp(modalidadX,'CT')~=1
                if get(handles.Espanol,'UserData')==1
                    msgbox('Debe seleccionar un TAC.');
                    return
                elseif get(handles.English,'UserData')==1
                    msgbox('Please open a CT image.');
                    return
                elseif get(handles.Deutsch,'UserData')==1
                    msgbox('Bitte auswählen Sie ein Bild CT.');
                    return
                end
            end
        else
            if strcmp(modalidadX,'MR')~=1
                if get(handles.Espanol,'UserData')==1
                    msgbox('Debe seleccionar una resonancia magnética.');
                    return
                elseif get(handles.English,'UserData')==1
                    msgbox('Please open a MR image.');
                    return
                elseif get(handles.Deutsch,'UserData')==1
                    msgbox('Bitte auswählen Sie ein Bild MR.');
                    return
                end
            end
        end

        filasX=infoX.Rows;
        columnasX=infoX.Columns;
        bitsX=infoX.BitDepth;
        corteX=infoX.SliceLocation;

        if get(handles.Espanol,'UserData')==1
            infoCortex=sprintf('Corte:%f',corteX);
            infoTAC=sprintf('"%s"  "%dx%d int%d"
Modalidad:%s',filenameX,filasX,columnasX,bitsX,modalidadX);
        elseif get(handles.English,'UserData')==1
            infoCortex=sprintf('Slice:%f',corteX);
            infoTAC=sprintf('"%s"  "%dx%d int%d"
Modality:%s',filenameX,filasX,columnasX,bitsX,modalidadX);
         elseif get(handles.Deutsch,'UserData')==1
            infoCortex=sprintf('Schnitt:%f',corteX);
            infoTAC=sprintf('"%s"  "%dx%d int%d"
Modalität:%s',filenameX,filasX,columnasX,bitsX,modalidadX);
        end

        X=double(handles.myImage);
        X=X-min(X(:));
        X=X/max(X(:))*255;

        imshow(X/255,'Parent',handles.axes1);
```

```
        set(handles.editInfotac,'UserData',filenameX);
        set(handles.textoInfotac,'UserData',pathname);

        set(handles.editInfotac,'String',infoTAC);

        set(handles.editCortex,'String',infoCortex);


    end
    guidata(hObject,handles);
catch
    msgbox('Error');

end


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in radioIntensidadPET.
function radioIntensidadPET_Callback(hObject, eventdata, handles)
% hObject    handle to radioIntensidadPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radioIntensidadPET



Z=dicomread(get(handles.editInfopettac,'UserData'));
Z=double(Z);
Z=Z/255;

%Se aumentan los niveles de intensidad de la zona afectada
subpZ=Z;
[f,c]=size(subpZ);

for i=1:1:f
    for j=1:1:c
        if subpZ(i,j)>=170.0000
            subpZ(i,j)=1.2*subpZ(i,j);
```

```
        end
    end
end

set(handles.radioIntensidadPET,'UserData',subpZ);

if get(handles.radioIntensidadPET,'Value') == 1

    imshow(subpZ/255,'Parent',handles.axes3);

elseif get(handles.radioIntensidadPET,'Value') == 0

    imshow(Z/255,'Parent',handles.axes3);

end

%----------SELECCIÓN DE COLOR (DESPLEGABLE)------------------------------
--
% --- Executes on selection change in desplegableMenu.
function desplegableMenu_Callback(hObject, eventdata, handles)
% hObject    handle to desplegableMenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

Z=dicomread(get(handles.editInfopettac,'UserData'));
Z=double(Z);
Z=Z/255;

subZ=get(handles.radioIntensidad,'UserData');
subpZ=get(handles.radioIntensidadPET,'UserData');

color=grayslice(Z/255,16);

switch get(handles.desplegableMenu,'Value')
    case 1
        if get(handles.radioIntensidad,'Value')==1
            imshow(subZ/255,'Parent',handles.axes3);
        elseif get(handles.radioIntensidadPET,'Value')==1
            imshow(subpZ/255,'Parent',handles.axes3);
        else
            imshow(Z/255,'Parent',handles.axes3);
        end
    case 2
        imshow(color,gray(16),'Parent',handles.axes3);
    case 3
        imshow(color,bone(16),'Parent',handles.axes3);
    case 4
        imshow(color,jet(16),'Parent',handles.axes3);
    case 5
        imshow(color,hot(16),'Parent',handles.axes3);
    case 6
        imshow(color,hsv(16),'Parent',handles.axes3);
end
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns
desplegableMenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
desplegableMenu


% --- Executes during object creation, after setting all properties.
function desplegableMenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to desplegableMenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in desplegableTipowavelet.
function desplegableTipowavelet_Callback(hObject, eventdata, handles)
% hObject    handle to desplegableTipowavelet (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
desplegableTipowavelet contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
desplegableTipowavelet

switch get(handles.desplegableTipowavelet,'Value');

    case 1
        %Wavelet de Haar
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
    case 2
        %Wavelet de Daubechies
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
    case 3
        %Wavelet Symlets
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
    case 4
        %Wavelet Coiflets
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
    case 5
        %Wavelet Biosplines
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
    case 6
```

```
        %Wavelet Reversebior
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
    case 7
        %Wavelet Meyer
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
end

% --- Executes during object creation, after setting all properties.
function desplegableTipowavelet_CreateFcn(hObject, eventdata, handles)
% hObject    handle to desplegableTipowavelet (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on selection change in desplegableNiveles.
function desplegableNiveles_Callback(hObject, eventdata, handles)
% hObject    handle to desplegableNiveles (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
desplegableNiveles contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
desplegableNiveles



% --- Executes during object creation, after setting all properties.
function desplegableNiveles_CreateFcn(hObject, eventdata, handles)
% hObject    handle to desplegableNiveles (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on selection change in desplegableMetodo.
function desplegableMetodo_Callback(hObject, eventdata, handles)
% hObject    handle to desplegableMetodo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
desplegableMetodo contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
desplegableMetodo

switch get(handles.desplegableMetodo,'Value')
    case 1
        %Media aritmética
        set(handles.listboxVentana,'Visible','off');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoVentana,'Visible','off');
        set(handles.textoUmbral,'Visible','off');
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','on');
        set(handles.textoWavelet,'Visible','on');
    case 2
        %Media ponderada gaussiana
        set(handles.listboxVentana,'Visible','on');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoVentana,'Visible','on');
        set(handles.textoUmbral,'Visible','off');
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','on');
        set(handles.textoWavelet,'Visible','on');
    case 3
        %Media ponderada lapacliana
        set(handles.listboxVentana,'Visible','off');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoVentana,'Visible','off');
        set(handles.textoUmbral,'Visible','off');
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
        set(handles.desplegableTipowavelet,'Visible','on');
        set(handles.textoWavelet,'Visible','on');
    case 4
        %Media imágenes multirresolución
        set(handles.listboxVentana,'Visible','off');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoVentana,'Visible','off');
        set(handles.textoUmbral,'Visible','off');
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','off');
        set(handles.textoWavelet,'Visible','off');
    case 5
        %Varianza ventana
        set(handles.listboxVentana,'Visible','on');
        set(handles.listboxUmbral,'Visible','on');
        set(handles.textoVentana,'Visible','on');
        set(handles.textoUmbral,'Visible','on');
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','on');
```

```
        set(handles.textoWavelet,'Visible','on');
    case 6
        %Coeficiente variación ventana
        set(handles.listboxVentana,'Visible','on');
        set(handles.listboxUmbral,'Visible','on');
        set(handles.textoVentana,'Visible','on');
        set(handles.textoUmbral,'Visible','on');
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','on');
        set(handles.textoWavelet,'Visible','on');
    case 7
        %Nivel actividad coeficientes
        set(handles.listboxVentana,'Visible','off');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoVentana,'Visible','off');
        set(handles.textoUmbral,'Visible','off');
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','on');
        set(handles.textoWavelet,'Visible','on');
    case 8
        %Nivel actividad ventana
        set(handles.listboxVentana,'Visible','on');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoVentana,'Visible','on');
        set(handles.textoUmbral,'Visible','off');
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','on');
        set(handles.textoWavelet,'Visible','on');

end


% --- Executes during object creation, after setting all properties.
function desplegableMetodo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to desplegableMetodo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on selection change in desplegableDescomposicion.
function desplegableDescomposicion_Callback(hObject, eventdata, handles)
% hObject    handle to desplegableDescomposicion (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
desplegableDescomposicion contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
desplegableDescomposicion

switch get(handles.desplegableDescomposicion,'Value')
    case 1
        %Ningún tipo de descomposición seleccionado
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.desplegableMetodo,'Visible','off');
        set(handles.desplegableTipowavelet,'Visible','off');
        set(handles.listboxVentana,'Visible','off');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoWavelet,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
        set(handles.textoMetodofusion,'Visible','off');
        set(handles.textoWavelet,'Visible','off');
        set(handles.textoVentana,'Visible','off');
        set(handles.textoUmbral,'Visible','off');
    case 2
        %Descomposición wavelet seleccionada
        set(handles.desplegableNiveles,'Visible','on');
        set(handles.desplegableMetodo,'Visible','on');
        set(handles.desplegableTipowavelet,'Visible','on');
        set(handles.textoWavelet,'Visible','on');
        set(handles.textoNiveles,'Visible','on');
        set(handles.textoMetodofusion,'Visible','on');
        set(handles.textoWavelet,'Visible','on');
        set(handles.textoVentana,'Visible','off');
        set(handles.textoUmbral,'Visible','off');
    case 3
        %Descomposición curvelet seleccionada
        set(handles.desplegableNiveles,'Visible','off');
        set(handles.desplegableMetodo,'Visible','off');
        set(handles.desplegableTipowavelet,'Visible','off');
        set(handles.listboxVentana,'Visible','off');
        set(handles.listboxUmbral,'Visible','off');
        set(handles.textoWavelet,'Visible','off');
        set(handles.textoNiveles,'Visible','off');
        set(handles.textoMetodofusion,'Visible','off');
        set(handles.textoWavelet,'Visible','off');
        set(handles.textoVentana,'Visible','off');
        set(handles.textoUmbral,'Visible','off');
end




% --- Executes during object creation, after setting all properties.
```

```
function desplegableDescomposicion_CreateFcn(hObject, eventdata, handles)
% hObject    handle to desplegableDescomposicion (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on selection change in listboxVentana.
function listboxVentana_Callback(hObject, eventdata, handles)
% hObject    handle to listboxVentana (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listboxVentana
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
listboxVentana



% --- Executes during object creation, after setting all properties.
function listboxVentana_CreateFcn(hObject, eventdata, handles)
% hObject    handle to listboxVentana (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on selection change in listboxUmbral.
function listboxUmbral_Callback(hObject, eventdata, handles)
% hObject    handle to listboxUmbral (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns listboxUmbral
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
listboxUmbral



% --- Executes during object creation, after setting all properties.
function listboxUmbral_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to listboxUmbral (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: listbox controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton2


% --- Executes on button press in radioIntensidad.
function radioIntensidad_Callback(hObject, eventdata, handles)
% hObject    handle to radioIntensidad (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radioIntensidad
%global Z;
Z=dicomread(get(handles.editInfopettac,'UserData'));
Z=double(Z);
Z=Z/255;
```

```
%Se aumentan los niveles de intensidad
subZ=Z;
[f,c]=size(subZ);

for i=1:1:f
    for j=1:1:c
        if subZ(i,j)>=80.0000
            subZ(i,j)=1.45*subZ(i,j);
        end
    end
end

set(handles.radioIntensidad,'UserData',subZ);

if get(handles.radioIntensidad,'Value') == 1

    imshow(subZ/255,'Parent',handles.axes3);

elseif get(handles.radioIntensidad,'Value') == 0

    imshow(Z/255,'Parent',handles.axes3);

end


% --- Executes on button press in botonGrafica.
function botonGrafica_Callback(hObject, eventdata, handles)
% hObject    handle to botonGrafica (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

Z=dicomread(get(handles.editInfopettac,'UserData'));
Z=double(Z);
Z=Z/255;

figure,plot(0:255,hist(Z,256));

if get(handles.Espanol,'UserData')==1
    xlabel('Niveles de gris')
    ylabel('Nº de píxeles')
elseif get(handles.English,'UserData')==1
    xlabel('Grayscale')
    ylabel('Number of pixels')
elseif get(handles.Deutsch,'UserData')==1
    xlabel('Graustufen')
    ylabel('Pixelnummern')
end

v=zeros(1,4);
v=axis;
v(1,2)=255;
axis(v);
```

```
% --- Executes on button press in botonSuperficie.
function botonSuperficie_Callback(hObject, eventdata, handles)
% hObject    handle to botonSuperficie (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

Z=dicomread(get(handles.editInfopettac,'UserData'));
Z=double(Z);
Z=Z/255;

figure,surf(Z);
shading interp;


% --- Executes on button press in botonMapapixeles.
function botonMapapixeles_Callback(hObject, eventdata, handles)
% hObject    handle to botonMapapixeles (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

impixelregion(handles.axes3);


% --- Executes on button press in botonDescwavelet.
function botonDescwavelet_Callback(hObject, eventdata, handles)
% hObject    handle to botonDescwavelet (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Se muestran las tres imágenes para ver mejor donde se ganan o pierden
%detalles.

%Imagen X
X=dicomread(get(handles.editInfotac,'UserData'));
X=double(X);
X=X/255;

[LL,LH,HL,HH]=dwt2(X,'haar');
imgX=[LL,LH;HL,HH];
if get(handles.cargarPETTAC,'UserData')==1
    if get(handles.Espanol,'UserData')==1
        figure('Name','TAC'),imshow(imgX,'Border','loose');
    else
        figure('Name','CT'),imshow(imgX,'Border','loose');
    end
else
    figure('Name','MR'),imshow(imgX,'Border','loose');
end

if get(handles.Espanol,'UserData')==1
    title(' Aproximación                                    Detalles
horizontales');
```

```
    xlabel('Detalles verticales                                Detalles
diagonales');
elseif get(handles.English,'UserData')==1
    title('Approximation matrix
Horizontal details');
    xlabel('Vertical details                                Diagonal
details');
elseif get(handles.Deutsch,'UserData')==1
    title(' Annäherung Matrize
Horizontaleausschnitte');
    xlabel('Vertikaleausschnitte
Diagonaleausschnitte');
end

%Imagen Y
Y=dicomread(get(handles.editInfopet,'UserData'));
Y=double(Y);
Y=Y/255;

[LL,LH,HL,HH]=dwt2(Y,'haar');
imgY=[LL/255,LH;HL,HH];
if get(handles.cargarRMTAC,'UserData')==1
    if get(handles.Espanol,'UserData')==1
        figure('Name','TAC'),imshow(imgY,'Border','loose');
    else
        figure('Name','CT'),imshow(imgY,'Border','loose');
    end
else
    figure('Name','PET'),imshow(imgY,'Border','loose');
end

if get(handles.Espanol,'UserData')==1
    title(' Aproximación                                Detalles
horizontales');
    xlabel('Detalles verticales                                Detalles
diagonales');
elseif get(handles.English,'UserData')==1
    title('Approximation matrix
Horizontal details');
    xlabel('Vertical details                                Diagonal
details');
elseif get(handles.Deutsch,'UserData')==1
    title(' Annäherung Matrize
Horizontaleausschnitte');
    xlabel('Vertikaleausschnitte
Diagonaleausschnitte');
end

%Imagen Z
Z=dicomread(get(handles.editInfopettac,'UserData'));
Z=double(Z);
Z=Z/255;

[LL,LH,HL,HH]=dwt2(Z,'haar');
imgZ=[LL/255,LH;HL,HH];
if get(handles.cargarPETTAC,'UserData')==1
```

```
    if get(handles.Espanol,'UserData')==1
        figure('Name','PET-TAC'),imshow(imgZ,'Border','loose');
    else
        figure('Name','PET-CT'),imshow(imgZ,'Border','loose');
    end
elseif get(handles.cargarRMTAC,'UserData')==1
    if get(handles.Espanol,'UserData')==1
        figure('Name','MR-TAC'),imshow(imgZ,'Border','loose');
    else
        figure('Name','MR-CT'),imshow(imgZ,'Border','loose');
    end
elseif get(handles.cargarRMPET,'UserData')==1
    figure('Name','MR-PET'),imshow(imgZ,'Border','loose');
end

if get(handles.Espanol,'UserData')==1
    title(' Aproximación                                    Detalles
horizontales');
    xlabel('Detalles verticales                                 Detalles
diagonales');
elseif get(handles.English,'UserData')==1
    title('Approximation matrix
Horizontal details');
    xlabel('Vertical details                                Diagonal
details');
elseif get(handles.Deutsch,'UserData')==1
    title(' Annäherung Matrize
Horizontaleausschnitte');
    xlabel('Vertikaleausschnitte
Diagonaleausschnitte');
end



% --- Executes on button press in checkboxTAC.
function checkboxTAC_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxTAC (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxTAC

X=dicomread(get(handles.editInfotac,'UserData'));

X=double(X);
X=X-min(X(:));
X=X/max(X(:))*255;

if get(handles.checkboxTAC,'Value') == 1
    tac=findobj(handles.axes1,'type','image');
    set(tac,'visible','off');
    pet=findobj(handles.axes2,'type','image');
    set(pet,'visible','off');
    pettac=findobj(handles.axes3,'type','image');
    set(pettac,'visible','off');
    set(handles.panelMedidas,'Visible','off');
    set(handles.panelImagenes,'Visible','off');
```

```
    set(handles.botonTAC,'Visible','off');
    set(handles.botonPET,'Visible','off');
    set(handles.textoPETTAC,'Visible','off');
    set(handles.axes6,'Visible','on');

    colorX=grayslice(X/255,16);

    switch get(handles.desplegableMenu,'Value')

    case 1
        imshow(X/255,'Parent',handles.axes6);
    case 2
        imshow(colorX,gray(16),'Parent',handles.axes6);
    case 3
        imshow(colorX,bone(16),'Parent',handles.axes6);
    case 4
        imshow(colorX,jet(16),'Parent',handles.axes6);
    case 5
        imshow(colorX,hot(16),'Parent',handles.axes6);
    case 6
        imshow(colorX,hsv(16),'Parent',handles.axes6);
    end

elseif get(handles.checkboxTAC,'Value') == 0
    tac=findobj(handles.axes1,'type','image');
    set(tac,'visible','on');
    pet=findobj(handles.axes2,'type','image');
    set(pet,'visible','on');
    pettac=findobj(handles.axes3,'type','image');
    set(pettac,'visible','on');
    grande=findobj(handles.axes6,'type','image');
    set(grande,'visible','off');
    set(handles.axes6,'Visible','off');
    set(handles.panelMedidas,'Visible','on');
    set(handles.panelMedidas,'Visible','on');
    set(handles.panelImagenes,'Visible','on');
    set(handles.botonTAC,'Visible','on');
    set(handles.botonPET,'Visible','on');
    set(handles.textoPETTAC,'Visible','on');

end


% --- Executes on button press in checkboxPET.
function checkboxPET_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxPET

Y=dicomread(get(handles.editInfopet,'UserData'));

Y=double(Y);
Y=Y-min(Y(:));
```

```
Y=Y/max(Y(:))*255;

if get(handles.checkboxPET,'Value') == 1

    tac=findobj(handles.axes1,'type','image');
    set(tac,'visible','off');
    pet=findobj(handles.axes2,'type','image');
    set(pet,'visible','off');
    pettac=findobj(handles.axes3,'type','image');
    set(pettac,'visible','off');
    set(handles.panelMedidas,'Visible','off');
    set(handles.panelImagenes,'Visible','off');
    set(handles.botonTAC,'Visible','off');
    set(handles.botonPET,'Visible','off');
    set(handles.textoPETTAC,'Visible','off');
    set(handles.axes6,'Visible','on');
    imshow(Y/255,'Parent',handles.axes6);

    colorY=grayslice(Y/255,16);

    switch get(handles.desplegableMenu,'Value')

    case 1
        imshow(Y/255,'Parent',handles.axes6);
    case 2
        imshow(colorY,gray(16),'Parent',handles.axes6);
    case 3
        imshow(colorY,bone(16),'Parent',handles.axes6);
    case 4
        imshow(colorY,jet(16),'Parent',handles.axes6);
    case 5
        imshow(colorY,hot(16),'Parent',handles.axes6);
    case 6
        imshow(colorY,hsv(16),'Parent',handles.axes6);
    end

elseif get(handles.checkboxPET,'Value') == 0

    tac=findobj(handles.axes1,'type','image');
    set(tac,'visible','on');
    pet=findobj(handles.axes2,'type','image');
    set(pet,'visible','on');
    pettac=findobj(handles.axes3,'type','image');
    set(pettac,'visible','on');
    grande=findobj(handles.axes6,'type','image');
    set(grande,'visible','off');
    set(handles.panelMedidas,'Visible','on');
    set(handles.panelImagenes,'Visible','on');
    set(handles.botonTAC,'Visible','on');
    set(handles.botonPET,'Visible','on');
    set(handles.textoPETTAC,'Visible','on');


end
```

```
% --- Executes on button press in checkboxPETTAC.
function checkboxPETTAC_Callback(hObject, eventdata, handles)
% hObject    handle to checkboxPETTAC (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of checkboxPETTAC
global Z;
Z=dicomread(get(handles.editInfopettac,'UserData'));
Z=double(Z);
Z=Z/255;

subZ=get(handles.radioIntensidad,'UserData');
subpZ=get(handles.radioIntensidadPET,'UserData');

if get(handles.checkboxPETTAC,'Value') == 1

    tac=findobj(handles.axes1,'type','image');
    set(tac,'visible','off');
    pet=findobj(handles.axes2,'type','image');
    set(pet,'visible','off');
    pettac=findobj(handles.axes3,'type','image');
    set(pettac,'visible','off');
    set(handles.panelMedidas,'Visible','off');
    set(handles.panelImagenes,'Visible','off');
    set(handles.botonTAC,'Visible','off');
    set(handles.botonPET,'Visible','off');
    set(handles.textoPETTAC,'Visible','off');
    set(handles.axes6,'Visible','on');
    if get(handles.radioIntensidad,'Value') == 1
        imshow(subZ/255,'Parent',handles.axes6);
    elseif get(handles.radioIntensidadPET,'Value') == 1
        imshow(subpZ/255,'Parent',handles.axes6);
    else
        imshow(Z/255,'Parent',handles.axes6);
    end

    colorZ=grayslice(Z/255,16);

    switch get(handles.desplegableMenu,'Value')

    case 1
        if get(handles.radioIntensidad,'Value') == 1
            imshow(subZ/255,'Parent',handles.axes6);
        elseif get(handles.radioIntensidadPET,'Value') == 1
            imshow(subpZ/255,'Parent',handles.axes6);
        else
            imshow(Z/255,'Parent',handles.axes6);
        end
    case 2
        imshow(colorZ,gray(16),'Parent',handles.axes6);
    case 3
        imshow(colorZ,bone(16),'Parent',handles.axes6);
    case 4
```

```
        imshow(colorZ,jet(16),'Parent',handles.axes6);
    case 5
        imshow(colorZ,hot(16),'Parent',handles.axes6);
    case 6
        imshow(colorZ,hsv(16),'Parent',handles.axes6);
    end

elseif get(handles.checkboxPETTAC,'Value') == 0

    tac=findobj(handles.axes1,'type','image');
    set(tac,'visible','on');
    pet=findobj(handles.axes2,'type','image');
    set(pet,'visible','on');
    pettac=findobj(handles.axes3,'type','image');
    set(pettac,'visible','on');
    grande=findobj(handles.axes6,'type','image');
    set(grande,'visible','off');
    set(handles.panelMedidas,'Visible','on');
    set(handles.panelMedidas,'Visible','on');
    set(handles.panelImagenes,'Visible','on');
    set(handles.botonTAC,'Visible','on');
    set(handles.botonPET,'Visible','on');
    set(handles.textoPETTAC,'Visible','on');

end


%--------------------------BOTÓN FUSIONAR-------------------------------
--
%-----------------------------------------------------------------------
--
%-----------------------------------------------------------------------
--
% --- Executes on button press in botonFusionar.
function botonFusionar_Callback(hObject, eventdata, handles)
% hObject    handle to botonFusionar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Se maximiza la ventana
maxfig(fusion_imagenes_medicas,0);

%No se ha seleccionado ninguna de las imágenes a fusionar
if isempty(get(handles.editInfotac,'UserData')) &&
isempty(get(handles.editInfopet,'UserData'))

    if get(handles.Espanol,'UserData')==1
        msgbox('Seleccione las dos imágenes que desea
fusionar','Imágenes');
        return
    elseif get(handles.English,'UserData')==1;
        msgbox('Please select both images to fusion','Images');
        return
    elseif get(handles.Deutsch,'UserData')==1;
        msgbox('Bitte auswählen Sie die Bilder','Bilder');
```

```
        return
    end

%Se ha seleccionado sólo la segunda de las imágenes a fusionar
elseif isempty(get(handles.editInfotac,'UserData'))

    if get(handles.Espanol,'UserData')==1
        if get(handles.cargarPETTAC,'UserData')==1
            msgbox('Seleccione la imagen TAC que desea fusionar','TAC');
            return
        else
            msgbox('Seleccione la resonancia magnética que desea
fusionar','MR');
            return
        end
    elseif get(handles.English,'UserData')==1;
        if get(handles.cargarPETTAC,'UserData')==1
            msgbox('Please select the CT image to fusion','CT');
            return
        else
            msgbox('Please select the MR image to fusion','MR');
            return
        end
    elseif get(handles.Deutsch,'UserData')==1;
        if get(handles.cargarPETTAC,'UserData')==1
            msgbox('Bitte auswählen Sie das Bild CT','CT');
            return
        else
            msgbox('Bitte auswählen Sie das Bild MR','MR');
            return
        end
    end

%Se ha seleccionado sólo la primera de las imágenes a fusionar
elseif isempty(get(handles.editInfopet,'UserData'))

    if get(handles.Espanol,'UserData')==1
        if get(handles.cargarRMTAC,'UserData')==1
            msgbox('Seleccione la imagen TAC que desea fusionar','TAC');
            return
        else
            msgbox('Seleccione la imagen PET que desea fusionar','PET');
            return
        end
    elseif get(handles.English,'UserData')==1;
        if get(handles.cargarRMTAC,'UserData')==1
            msgbox('Please select the CT image to fusion','CT');
            return
        else
            msgbox('Please select the PET image to fusion','PET');
            return
        end
    elseif get(handles.Deutsch,'UserData')==1;
        if get(handles.cargarRMTAC,'UserData')==1
            msgbox('Bitte auswählen Sie das Bild CT','CT');
            return
```

```
        else
            msgbox('Bitte auswählen Sie das Bild PET','PET');
            return
        end
    end

else
    %Se han seleccionado ambas imágenes a fusionar

X=dicomread(fullfile(get(handles.textoInfotac,'UserData'),get(handles.edi
tInfotac,'UserData')));

Y=dicomread(fullfile(get(handles.textoInfopet,'UserData'),get(handles.edi
tInfopet,'UserData')));

    X=double(X);
    X=X-min(X(:));
    X=X/max(X(:))*255;

    Y=double(Y);
    Y=Y-min(Y(:));
    Y=Y/max(Y(:))*255;

    set(handles.editCortex,'ForegroundColor',[.2 .4 1]);
    set(handles.editCortey,'ForegroundColor',[.2 .4 1]);

    infoX=dicominfo(get(handles.editInfotac,'UserData'));
    infoY=dicominfo(get(handles.editInfopet,'UserData'));

    corteX=infoX.SliceLocation;
    corteY=infoY.SliceLocation;

    if corteX ~= corteY
        if get(handles.Espanol,'UserData')==1
            msgbox('Seleccione los cortes que coincidan','Cortes');
        elseif get(handles.English,'UserData')==1
            msgbox('Please select matching slices','Slices');
        elseif get(handles.Deutsch,'UserData')==1
            msgbox('Bitte auswählen Sie die passende
Schnitte','Schnitte');
        end

        set(handles.editCortex,'ForegroundColor','red');
        set(handles.editCortey,'ForegroundColor','red');

        return
    end

end

set(fusion_imagenes_medicas, 'pointer', 'watch');
drawnow;

%--------------------SELECCIÓN DE PARÁMETROS--------------------------
```

```
%Se lee el tipo de descomposición elegido por el usuario
descomposicion=get(handles.desplegableDescomposicion,'Value');

%Se lee el tipo de wavelet seleccionada por el usuario
wavelet=get(handles.desplegableTipowavelet,'Value');

%Se lee el nivel de descomposición elegido
nivel=get(handles.desplegableNiveles,'Value');

%Se lee el método de fusión elegido por el usuario
metodo=get(handles.desplegableMetodo,'Value');

%Se lee el tamaño de ventana seleccionado por el usuario
ventana=get(handles.listboxVentana,'Value');

%Comprobamos que el menú para elegir umbral no está visible
umbral=get(handles.listboxUmbral,'Value');

if descomposicion==1

     %------------------NO SE DESCOMPONE LA IMAGEN---------------------
     %Se fusiona con el método de cálculo de la media de los niveles de
     %intensidad.
     Z=media_intensidades(X,Y);

elseif descomposicion==2
    %--------------------DESCOMPOSICIÓN WAVELET-----------------------

    switch metodo

        case 1
            %Media aritmética
            Z=media(X,Y,wavelet,nivel);
        case 2
            %Media ponderada gaussiana
            Z=media_ponderada_gaussiana(X,Y,ventana,wavelet,nivel);
        case 3
            %Media ponderada lapacliana
            Z=media_ponderada_laplaciana(X,Y,wavelet);
        case 4
            %Media imágenes multirresolución
            Z=media_imagenes_multirresolucion(X,Y,nivel);
        case 5
             %Varianza ventana
             Z=varianza_ventana(X,Y,ventana,umbral,wavelet,nivel);
        case 6
            %Coeficiente variación ventana

Z=coeficiente_variacion_ventana(X,Y,ventana,umbral,wavelet,nivel);
        case 7
            %Nivel actividad coeficientes
            Z=nivel_actividad_coeficientes(X,Y,wavelet,nivel);
        case 8
            %Nivel actividad ventana
```

```
            Z=nivel_actividad_ventana(X,Y,ventana,wavelet,nivel);

    end

end

imshow(X/255,'Parent',handles.axes1);
imshow(Y/255,'Parent',handles.axes2);
imshow(Z/255,'Parent',handles.axes3);

[fx,cx]=size(Z);

if get(handles.cargarPETTAC,'UserData')==1

    cortes=zeros(205,1);

    for i=2:1:205
        cortes(1)=13.000;
        cortes(i)=cortes(i-1)-4.25;
    end

    j=find(cortes==corteX);

    filename_fusionada=sprintf('PET-TAC-%d.v2',j);
    set(handles.editInfopettac,'UserData',filename_fusionada);
    dicomwrite(Z/255,filename_fusionada);


    if j<10

    infoCorteZ=sprintf('"PET-TAC-00%d.v2"  "%dx%d double"',j,fx,cx);
    set(handles.editInfopettac,'String',infoCorteZ);

    elseif j<100

    infoCorteZ=sprintf('"PET-TAC-0%d.v2"  "%dx%d double"',j,fx,cx);
    set(handles.editInfopettac,'String',infoCorteZ);

    else

    infoCorteZ=sprintf('"PET-TAC-%d.v2"  "%dx%d double"',j,fx,cx);
    set(handles.editInfopettac,'String',infoCorteZ);

    end

else

    img_name=get(handles.editInfopet,'UserData');
    C=sscanf(img_name,'%s%u');
    j=char(C(5));

    if get(handles.cargarRMTAC,'UserData')==1
```

```
        if length(C)>9
            k=char(C(6));
            set(handles.editInfopettac,'String',sprintf('"MR-TAC-
%c%c.dcm"  "%dx%d double"',j,k,fx,cx));
        else
            set(handles.editInfopettac,'String',sprintf('"MR-TAC-%c.dcm"
"%dx%d double"',j,fx,cx));
        end

        filename_fusionada=sprintf('RM-TAC-%c.dcm',j);
        set(handles.editInfopettac,'UserData',filename_fusionada);
        dicomwrite(Z/255,filename_fusionada);

    elseif get(handles.cargarRMPET,'UserData')==1

        if length(C)>9
            k=char(C(6));
            set(handles.editInfopettac,'String',sprintf('"MR-PET-
%c%c.dcm"  "%dx%d double"',j,k,fx,cx));
        else
            set(handles.editInfopettac,'String',sprintf('"MR-PET-%c.dcm"
"%dx%d double"',j,fx,cx));
        end

        filename_fusionada=sprintf('RM-PET-%c.dcm',j);
        set(handles.editInfopettac,'UserData',filename_fusionada);
        dicomwrite(Z/255,filename_fusionada);

    end

end

%Ajuste previo al cáculo de medidas para el caso de media ponderada
laplaciana
if metodo==3

    %Se acotan los coeficientes entre 0 y 255.
    Z=Z-min(Z(:));
    Z=Z/max(Z(:))*255;

end

[calidadX,calidadY,CC_TAC,CC_PET,MSEx,MSEy,PSNR,SSIM,CR,MI]=medidas(X,Y,Z
);

set(handles.editCalidadx,'String',calidadX);
set(handles.editCalidady,'String',calidadY);
set(handles.editCCTAC,'String',CC_TAC);
set(handles.editCCPET,'String',CC_PET);
set(handles.editMSEx,'String',MSEx);
set(handles.editMSEy,'String',MSEy);
set(handles.editPSNR,'String',PSNR);
set(handles.editSSIM,'String',SSIM);
set(handles.editCR,'String',CR);
set(handles.editMI,'String',MI);
```

```
set(handles.figure1, 'pointer', 'arrow');

if get(handles.cargarPETTAC,'UserData')==1
    set(handles.cargarRMPET,'UserData',0);
    set(handles.cargarPETTAC,'UserData',1);
    set(handles.cargarRMTAC,'UserData',0);
elseif get(handles.cargarRMTAC,'UserData')==1
    set(handles.cargarRMPET,'UserData',0);
    set(handles.cargarPETTAC,'UserData',0);
    set(handles.cargarRMTAC,'UserData',1);
elseif get(handles.cargarRMPET,'UserData')==1
    set(handles.cargarRMPET,'UserData',1);
    set(handles.cargarPETTAC,'UserData',0);
    set(handles.cargarRMTAC,'UserData',0);
end

maxfig(fusion_imagenes_medicas,1);



function editCCTAC_Callback(hObject, eventdata, handles)
% hObject    handle to editCCTAC (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCCTAC as text
%        str2double(get(hObject,'String')) returns contents of editCCTAC
as a double


% --- Executes during object creation, after setting all properties.
function editCCTAC_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCCTAC (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editCCPET_Callback(hObject, eventdata, handles)
% hObject    handle to editCCPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCCPET as text
```

```
%        str2double(get(hObject,'String')) returns contents of editCCPET
as a double


% --- Executes during object creation, after setting all properties.
function editCCPET_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCCPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editPSNR_Callback(hObject, eventdata, handles)
% hObject    handle to editPSNR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editPSNR as text
%        str2double(get(hObject,'String')) returns contents of editPSNR
as a double


% --- Executes during object creation, after setting all properties.
function editPSNR_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editPSNR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editPSNRPET_Callback(hObject, eventdata, handles)
% hObject    handle to editPSNRPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editPSNRPET as text
%        str2double(get(hObject,'String')) returns contents of
editPSNRPET as a double
```

```
% --- Executes during object creation, after setting all properties.
function editPSNRPET_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editPSNRPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editCalidadx_Callback(hObject, eventdata, handles)
% hObject    handle to editCalidadx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCalidadx as text
%        str2double(get(hObject,'String')) returns contents of
editCalidadx as a double


% --- Executes during object creation, after setting all properties.
function editCalidadx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCalidadx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editCalidady_Callback(hObject, eventdata, handles)
% hObject    handle to editCalidady (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCalidady as text
%        str2double(get(hObject,'String')) returns contents of
editCalidady as a double
```

```
% --- Executes during object creation, after setting all properties.
function editCalidady_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCalidady (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editMSEx_Callback(hObject, eventdata, handles)
% hObject    handle to editMSEx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editMSEx as text
%        str2double(get(hObject,'String')) returns contents of editMSEx
as a double


% --- Executes during object creation, after setting all properties.
function editMSEx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editMSEx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editMSEy_Callback(hObject, eventdata, handles)
% hObject    handle to editMSEy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editMSEy as text
%        str2double(get(hObject,'String')) returns contents of editMSEy
as a double


% --- Executes during object creation, after setting all properties.
function editMSEy_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to editMSEy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editSSIM_Callback(hObject, eventdata, handles)
% hObject    handle to editSSIM (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editSSIM as text
%        str2double(get(hObject,'String')) returns contents of editSSIM
as a double


% --- Executes during object creation, after setting all properties.
function editSSIM_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editSSIM (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editSimilitudy_Callback(hObject, eventdata, handles)
% hObject    handle to editSimilitudy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editSimilitudy as text
%        str2double(get(hObject,'String')) returns contents of
editSimilitudy as a double


% --- Executes during object creation, after setting all properties.
function editSimilitudy_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editSimilitudy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editCR_Callback(hObject, eventdata, handles)
% hObject    handle to editCR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCR as text
%        str2double(get(hObject,'String')) returns contents of editCR as
a double


% --- Executes during object creation, after setting all properties.
function editCR_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editCRy_Callback(hObject, eventdata, handles)
% hObject    handle to editCRy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCRy as text
%        str2double(get(hObject,'String')) returns contents of editCRy as
a double


% --- Executes during object creation, after setting all properties.
function editCRy_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCRy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editMI_Callback(hObject, eventdata, handles)
% hObject    handle to editMI (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editMI as text
%        str2double(get(hObject,'String')) returns contents of editMI as
a double


% --- Executes during object creation, after setting all properties.
function editMI_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editMI (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editMIy_Callback(hObject, eventdata, handles)
% hObject    handle to editMIy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editMIy as text
%        str2double(get(hObject,'String')) returns contents of editMIy as
a double


% --- Executes during object creation, after setting all properties.
function editMIy_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editMIy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editInfotac_Callback(hObject, eventdata, handles)
% hObject    handle to editInfotac (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editInfotac as text
%        str2double(get(hObject,'String')) returns contents of
editInfotac as a double


% --- Executes during object creation, after setting all properties.
function editInfotac_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editInfotac (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editInfopet_Callback(hObject, eventdata, handles)
% hObject    handle to editInfopet (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editInfopet as text
%        str2double(get(hObject,'String')) returns contents of
editInfopet as a double


% --- Executes during object creation, after setting all properties.
function editInfopet_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editInfopet (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editInfopettac_Callback(hObject, eventdata, handles)
% hObject    handle to editInfopettac (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editInfopettac as text
%        str2double(get(hObject,'String')) returns contents of
editInfopettac as a double


% --- Executes during object creation, after setting all properties.
function editInfopettac_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editInfopettac (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function editCortex_Callback(hObject, eventdata, handles)
% hObject    handle to editCortex (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCortex as text
%        str2double(get(hObject,'String')) returns contents of editCortex
as a double


% --- Executes during object creation, after setting all properties.
function editCortex_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCortex (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end



function editCortey_Callback(hObject, eventdata, handles)
% hObject    handle to editCortey (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of editCortey as text
%        str2double(get(hObject,'String')) returns contents of editCortey
as a double


% --- Executes during object creation, after setting all properties.
function editCortey_CreateFcn(hObject, eventdata, handles)
% hObject    handle to editCortey (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in botonAnimacion.
function botonAnimacion_Callback(hObject, eventdata, handles)
% hObject    handle to botonAnimacion (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

if get(handles.cargarPETTAC,'UserData')==1

    set(handles.botonAnimacion,'UserData',1);
    secuencia=1;
    set(handles.botonIniciarexploracion,'UserData',secuencia);

    if get(handles.Espanol,'UserData')==1

        set(handles.botonIniciarexploracion,'String','Iniciar');

        fusionada=sprintf('Exploración cancelada');
        set(handles.editInfopettac,'String',fusionada);

    elseif get(handles.English,'UserData')==1

        set(handles.botonIniciarexploracion,'String','Play');

        fusionada=sprintf('Exploration finished');
```

```
        set(handles.editInfopettac,'String',fusionada);

     elseif get(handles.Deutsch,'UserData')==1

        set(handles.botonIniciarexploracion,'String','Abspielen');

        fusionada=sprintf('Exploration beendet');
        set(handles.editInfopettac,'String',fusionada);

    end
else
    if get(handles.Espanol,'UserData')==1
        msgbox('Opción disponible sólo en el estudio PET-TAC.');
    elseif get(handles.English,'UserData')==1
        msgbox('Available only with PET-CT exploration.');
    elseif get(handles.Deutsch,'UserData')==1
        msgbox('Nur mit PET-CT Studie verfügbar.');
    end
end


% --- Executes when user attempts to close figure1.
function figure1_CloseRequestFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: delete(hObject) closes the figure

if get(handles.Espanol,'UserData')==1

    cerrar=questdlg('¿Está seguro de cerrar la
aplicación?','Cerrar','Si','No','Default');

    switch cerrar
        case 'Si'
            set(handles.editInfotac,'UserData',0);
            set(handles.editInfopet,'UserData',0);
            set(handles.editInfopettac,'UserData',0);
            delete(hObject);
        case 'No'
            return
        case 'Default'
            return
    end

elseif get(handles.English,'UserData')==1;

    cerrar=questdlg('Are you sure you want to
quit?','Exit','Yes','No','Default');

    switch cerrar
        case 'Yes'
            set(handles.editInfotac,'UserData',0);
```

```
            set(handles.editInfopet,'UserData',0);
            set(handles.editInfopettac,'UserData',0);
            delete(hObject);
        case 'No'
            return
        case 'Default'
            return
    end

elseif get(handles.Deutsch,'UserData')==1;

    cerrar=questdlg('Möchten Sie das Programm
verlassen?','Schliessen','Ja','Nein','Default');

    switch cerrar
        case 'Ja'
            set(handles.editInfotac,'UserData',0);
            set(handles.editInfopet,'UserData',0);
            set(handles.editInfopettac,'UserData',0);
            delete(hObject);
        case 'Nein'
            return
        case 'Default'
            return
    end
end


% --- Executes when figure1 is resized.
function figure1_ResizeFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on selection change in desplegableFiltrado.
function desplegableFiltrado_Callback(hObject, eventdata, handles)
% hObject    handle to desplegableFiltrado (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
desplegableFiltrado contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
desplegableFiltrado


% --- Executes during object creation, after setting all properties.
function desplegableFiltrado_CreateFcn(hObject, eventdata, handles)
% hObject    handle to desplegableFiltrado (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in botonIniciarexploracion.
function botonIniciarexploracion_Callback(hObject, eventdata, handles)
% hObject    handle to botonIniciarexploracion (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of
botonIniciarexploracion

if get(handles.cargarPETTAC,'UserData')==1

    seguir=get(handles.botonIniciarexploracion,'Value');

    if get(handles.Espanol,'UserData')==1
        set(handles.botonIniciarexploracion,'String','Pausar');
    else
        set(handles.botonIniciarexploracion,'String','Pause');
    end

    set(handles.editInfopettac,'ForegroundColor','blue');

    set(handles.botonAnimacion,'UserData',0);

    secuencia=get(handles.botonIniciarexploracion,'UserData');

    while seguir==1
        for i=secuencia:1:205
            if get(handles.botonAnimacion,'UserData')==1
                return
            end

            if get(handles.Espanol,'UserData')==1
                set(handles.botonIniciarexploracion,'String','Pausar');
            else
                set(handles.botonIniciarexploracion,'String','Pause');
            end
            %set(handles.botonIniciarexploracion,'String','Pausar');

            if i>99
                leer_tac=sprintf('TAC-%d.v2',i);
                leer_pet=sprintf('PET-%d.v2',i);
                fusionada=sprintf('PET-TAC-%d.v2',i);
                set(handles.editInfopettac,'String',fusionada);
            elseif i>9
                leer_tac=sprintf('TAC-0%d.v2',i);
                leer_pet=sprintf('PET-0%d.v2',i);
                fusionada=sprintf('PET-TAC-0%d.v2',i);
                set(handles.editInfopettac,'String',fusionada);
            else
```

```
                leer_tac=sprintf('TAC-00%d.v2',i);
                leer_pet=sprintf('PET-00%d.v2',i);
                fusionada=sprintf('PET-TAC-00%d.v2',i);
                set(handles.editInfopettac,'String',fusionada);
            end

            handles.myImage=dicomread(leer_tac);

            X=double(handles.myImage);
            X=X-min(X(:));
            X=X/max(X(:))*255;

            handles.myImage=dicomread(leer_pet);

            Y=double(handles.myImage);
            Y=Y-min(Y(:));
            Y=Y/max(Y(:))*255;

            imshow(X/255,'Parent',handles.axes1);
            imshow(Y/255,'Parent',handles.axes2);

            Z=media_intensidades(X,Y);

            imshow(Z/255,'Parent',handles.axes3);
            pause(1);
            secuencia=secuencia+1;
            set(handles.botonIniciarexploracion,'UserData',secuencia);
            secuencia=get(handles.botonIniciarexploracion,'UserData');

            seguir=get(handles.botonIniciarexploracion,'Value');

            if seguir==0

                if get(handles.Espanol,'UserData')==1

set(handles.botonIniciarexploracion,'String','Continuar');
                    fusionada=sprintf('Exploración pausada');
                    set(handles.editInfopettac,'String',fusionada);
                    break;
                elseif get(handles.English,'UserData')==1

set(handles.botonIniciarexploracion,'String','Continue');
                    fusionada=sprintf('Exploration paused');
                    set(handles.editInfopettac,'String',fusionada);
                    break;
                elseif get(handles.Deutsch,'UserData')==1

set(handles.botonIniciarexploracion,'String','Weiter');
                    fusionada=sprintf('Explorationspause');
                    set(handles.editInfopettac,'String',fusionada);
                    break;
                end

            end
```

```
        end
    end

else
    if get(handles.Espanol,'UserData')==1
        msgbox('Opción disponible sólo en el estudio PET-TAC.');
    elseif get(handles.English,'UserData')==1
        msgbox('Available just with PET-CT exploration.');
    elseif get(handles.Deutsch,'UserData')==1
        msgbox('Nur mit PET-CT Studie verfügbar.');
    end
end


% --------------------------------------------------------------------
function Archivo_Callback(hObject, eventdata, handles)
% hObject    handle to Archivo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)




% --------------------------------------------------------------------
function Untitled_1_Callback(hObject, eventdata, handles)
% hObject    handle to Archivo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Cargar_Callback(hObject, eventdata, handles)
% hObject    handle to Cargar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Salir_Callback(hObject, eventdata, handles)
% hObject    handle to Salir (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

if get(handles.Espanol,'UserData')==1

    cerrar=questdlg('¿Está seguro de cerrar la
aplicación?','Cerrar','Si','No','Default');

    switch cerrar
        case 'Si'
            set(handles.editInfotac,'UserData',0);
            set(handles.editInfopet,'UserData',0);
            set(handles.editInfopettac,'UserData',0);
            delete(hObject);
```

```
        case 'No'
            return
        case 'Default'
            return
    end

elseif get(handles.English,'UserData')==1;

    cerrar=questdlg('Are you sure you want to
quit?','Exit','Yes','No','Default');

    switch cerrar
        case 'Yes'
            set(handles.editInfotac,'UserData',0);
            set(handles.editInfopet,'UserData',0);
            set(handles.editInfopettac,'UserData',0);
            delete(hObject);
        case 'No'
            return
        case 'Default'
            return
    end

elseif get(handles.Deutsch,'UserData')==1;

    cerrar=questdlg('Möchten Sie das Programm
verlassen?','Schliessen','Ja','Nein','Default');

    switch cerrar
        case 'Ja'
            set(handles.editInfotac,'UserData',0);
            set(handles.editInfopet,'UserData',0);
            set(handles.editInfopettac,'UserData',0);
            delete(hObject);
        case 'Nein'
            return
        case 'Default'
            return
    end
end


% --------------------------------------------------------------------
function cargarRMTAC_Callback(hObject, eventdata, handles)
% hObject    handle to cargarRMTAC (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

set(handles.cargarRMPET,'UserData',0);
set(handles.cargarPETTAC,'UserData',0);
set(handles.cargarRMTAC,'UserData',1);

if get(handles.Espanol,'UserData')==1
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','TAC');
```

```
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','TAC');
    set(handles.textoInfopettac,'String','MR-TAC');
    set(handles.textoPETTAC,'String','MR-TAC');
    set(handles.textoMedidasTAC,'String','MR');
    set(handles.textoMedidasPET,'String','TAC');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','CT');
    set(handles.checkboxPETTAC,'String','MR-TAC');
else
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','CT');
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','CT');
    set(handles.textoInfopettac,'String','MR-CT');
    set(handles.textoPETTAC,'String','MR-CT');
    set(handles.textoMedidasTAC,'String','MR');
    set(handles.textoMedidasPET,'String','CT');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','CT');
    set(handles.checkboxPETTAC,'String','MR-CT');
end

if get(handles.cargarPETTAC,'UserData')==1
    set(handles.radioIntensidadPET,'Visible','on');
else
    set(handles.radioIntensidadPET,'Visible','off');
end

set(handles.desplegableNiveles,'Visible','off');
set(handles.desplegableMetodo,'Visible','off');
set(handles.desplegableTipowavelet,'Visible','off');
set(handles.listboxVentana,'Visible','off');
set(handles.listboxUmbral,'Visible','off');
set(handles.textoWavelet,'Visible','off');
set(handles.textoNiveles,'Visible','off');
set(handles.textoMetodofusion,'Visible','off');
set(handles.textoWavelet,'Visible','off');
set(handles.textoVentana,'Visible','off');
set(handles.textoUmbral,'Visible','off');


% --------------------------------------------------------------------
function Paciente_2_Callback(hObject, eventdata, handles)
% hObject    handle to Untitled_5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%----------PACIENTE 2 RM-TAC-----------------------------------------------
-

set(handles.Paciente_1,'UserData',0);
set(handles.Paciente_2,'UserData',1);

% --------------------------------------------------------------------
function Untitled_5_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to Untitled_5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Untitled_6_Callback(hObject, eventdata, handles)
% hObject    handle to Untitled_6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Untitled_7_Callback(hObject, eventdata, handles)
% hObject    handle to Untitled_7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Ver_Callback(hObject, eventdata, handles)
% hObject    handle to Ver (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --------------------------------------------------------------------
function Ayuda_Callback(hObject, eventdata, handles)
% hObject    handle to Ver (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA

% --------------------------------------------------------------------
function cargarPETTAC_Callback(hObject, eventdata, handles)
% hObject    handle to cargarPETTAC (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

set(handles.cargarRMPET,'UserData',0);
set(handles.cargarPETTAC,'UserData',1);
set(handles.cargarRMTAC,'UserData',0);

if get(handles.Espanol,'UserData')==1
    set(handles.botonTAC,'String','TAC');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','TAC');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','PET-TAC');
    set(handles.textoPETTAC,'String','PET-TAC');
    set(handles.textoMedidasTAC,'String','TAC');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','TAC');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','PET-TAC');
```

```
else
    set(handles.botonTAC,'String','CT');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','CT');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','PET-CT');
    set(handles.textoPETTAC,'String','PET-CT');
    set(handles.textoMedidasTAC,'String','CT');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','TAC');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','PET-CT');
end

if get(handles.cargarPETTAC,'UserData')==1
    set(handles.radioIntensidadPET,'Visible','on');
else
    set(handles.radioIntensidadPET,'Visible','off');
end

set(handles.desplegableNiveles,'Visible','off');
set(handles.desplegableMetodo,'Visible','off');
set(handles.desplegableTipowavelet,'Visible','off');
set(handles.listboxVentana,'Visible','off');
set(handles.listboxUmbral,'Visible','off');
set(handles.textoWavelet,'Visible','off');
set(handles.textoNiveles,'Visible','off');
set(handles.textoMetodofusion,'Visible','off');
set(handles.textoWavelet,'Visible','off');
set(handles.textoVentana,'Visible','off');
set(handles.textoUmbral,'Visible','off');



% --------------------------------------------------------------------
function cargarRMPET_Callback(hObject, eventdata, handles)
% hObject    handle to cargarRMPET (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

set(handles.cargarRMPET,'UserData',1);
set(handles.cargarPETTAC,'UserData',0);
set(handles.cargarRMTAC,'UserData',0);

set(handles.botonTAC,'String','MR');
set(handles.botonPET,'String','PET');
set(handles.textoInfotac,'String','MR');
set(handles.textoInfopet,'String','PET');
set(handles.textoInfopettac,'String','MR-PET');
set(handles.textoPETTAC,'String','MR-PET');
set(handles.textoMedidasTAC,'String','MR');
set(handles.textoMedidasPET,'String','PET');
set(handles.checkboxTAC,'String','MR');
set(handles.checkboxPET,'String','PET');
set(handles.checkboxPETTAC,'String','MR-PET');
```

```
if get(handles.cargarPETTAC,'UserData')==1
    set(handles.radioIntensidadPET,'Visible','on');
else
    set(handles.radioIntensidadPET,'Visible','off');
end

set(handles.desplegableNiveles,'Visible','off');
set(handles.desplegableMetodo,'Visible','off');
set(handles.desplegableTipowavelet,'Visible','off');
set(handles.listboxVentana,'Visible','off');
set(handles.listboxUmbral,'Visible','off');
set(handles.textoWavelet,'Visible','off');
set(handles.textoNiveles,'Visible','off');
set(handles.textoMetodofusion,'Visible','off');
set(handles.textoWavelet,'Visible','off');
set(handles.textoVentana,'Visible','off');
set(handles.textoUmbral,'Visible','off');


% --------------------------------------------------------------------
function Paciente_1_Callback(hObject, eventdata, handles)
% hObject    handle to Paciente_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%----------PACIENTE 1 RM-TAC-----------------------------------------------
-


set(handles.Paciente_1,'UserData',1);
set(handles.Paciente_2,'UserData',0);


% --------------------------------------------------------------------
function Guia_Callback(hObject, eventdata, handles)
% hObject    handle to Guia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

mensaje=sprintf('Para un mejor funcionamiento de la aplicación siga estas
simples instrucciones:\n\n1. Seleccione el tipo de fusión en Archivo-
>Cargar.\n\n2. Abra las dos imágenes a fusionar y asegúrese de que los
cortes coinciden.\n\n3. Elija la descomposición y el método de fusión con
los parámetros asociados y pulse Fusionar.');
helpdlg(mensaje,'Guia');


% --------------------------------------------------------------------
function Acerca_de_Callback(hObject, eventdata, handles)
% hObject    handle to Acerca_de (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

mensaje=sprintf('\t\t\t\t    Fusión_imágenes_médicas\n\nAutores: Antonio
Marín García\n              Dr. Jorge Larrey Ruiz');
```

```
helpdlg(mensaje,'Acerca de');


% --------------------------------------------------------------------
function Idioma_Callback(hObject, eventdata, handles)
% hObject    handle to Idioma (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --------------------------------------------------------------------
function Espanol_Callback(hObject, eventdata, handles)
% hObject    handle to Espanol (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

set(handles.Espanol,'UserData',1);
set(handles.English,'UserData',0);
set(handles.Deutsch,'UserData',0);

%Las etiquetas del menú principal se muestran en español.
set(handles.Archivo,'Label','Archivo');
set(handles.Cargar,'Label','Cargar');
set(handles.Salir,'Label','Salir');
set(handles.cargarPETTAC,'Label','Estudio PET-TAC');
set(handles.cargarRMPET,'Label','Estudio RM-PET');
set(handles.cargarRMTAC,'Label','Estudio RM-TAC');
set(handles.Paciente_1,'Label','Paciente 1');
set(handles.Paciente_2,'Label','Paciente 2');
set(handles.Ver,'Label','Ver');
set(handles.Idioma,'Label','Idioma');
set(handles.Ayuda,'Label','Ayuda');
set(handles.Guia,'Label','Guia');
set(handles.Acerca_de,'Label','Acerca de');

%El resto de etiquetas y textos de la interfaz se muestran en español
if get(handles.cargarPETTAC,'UserData')==1
    set(handles.botonTAC,'String','TAC');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','TAC');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','PET-TAC');
    set(handles.textoPETTAC,'String','PET-TAC');
    set(handles.textoMedidasTAC,'String','TAC');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','TAC');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','PET-TAC');
elseif get(handles.cargarRMTAC,'UserData')==1
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','TAC');
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','TAC');
    set(handles.textoInfopettac,'String','MR-TAC');
    set(handles.textoPETTAC,'String','MR-TAC');
    set(handles.textoMedidasTAC,'String','MR');
```

```
    set(handles.textoMedidasPET,'String','TAC');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','TAC');
    set(handles.checkboxPETTAC,'String','MR-TAC');
elseif get(handles.cargarRMPET,'UserData')==1
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','MR-PET');
    set(handles.textoPETTAC,'String','MR-PET');
    set(handles.textoMedidasTAC,'String','MR');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','MR-PET');
end

set(handles.textoCalidad,'String','CALIDAD');
set(handles.textoMedia,'String','MEDIA');
set(handles.botonDescwavelet,'String','Descomposición wavelet');
set(handles.botonGrafica,'String','Histograma');
set(handles.botonSuperficie,'String','Superficie');
set(handles.botonMapapixeles,'String','Mapa píxeles');
set(handles.botonIniciarexploracion,'String','Iniciar');
set(handles.botonAnimacion,'String','Detener');
set(handles.textoColor,'String','Color');
set(handles.radioIntensidad,'String','Intensidad');
set(handles.radioIntensidadPET,'String','Intensidad PET');
set(handles.botonFusionar,'String','Fusionar');
set(handles.textoVentana,'String','Ventana');
set(handles.textoUmbral,'String','Umbral');
set(handles.textoMetodofusion,'String','Método fusión');
set(handles.textoNiveles,'String','Niveles descomposición');
set(handles.textoDescomposicion,'String','Descomposición');


set(handles.panelImagenes,'Title','Imágenes');
set(handles.panelFusion,'Title','Fusión');
set(handles.panelAjustes,'Title','Ajustes');
set(handles.panelCompleta,'Title','Pantalla completa');
set(handles.panelGraficas,'Title','Gráficas');
set(handles.panelExtras,'Title','Exploración completa');
set(handles.panelMedidas,'Title','Medidas fusión');


% --------------------------------------------------------------------
function English_Callback(hObject, eventdata, handles)
% hObject    handle to English (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

set(handles.Espanol,'UserData',0);
set(handles.English,'UserData',1);
set(handles.Deutsch,'UserData',0);
```

```
%Las etiquetas del menú principal se muestran en inglés.
set(handles.Archivo,'Label','File');
set(handles.Cargar,'Label','Load');
set(handles.Salir,'Label','Exit');
set(handles.cargarPETTAC,'Label','PET-TAC fusion');
set(handles.cargarRMPET,'Label','RM-PET fusion');
set(handles.cargarRMTAC,'Label','RM-TAC fusion');
set(handles.Paciente_1,'Label','Patient 1');
set(handles.Paciente_2,'Label','Patient 2');
set(handles.Ver,'Label','View');
set(handles.Idioma,'Label','Language');
set(handles.Ayuda,'Label','Help');
set(handles.Guia,'Label','Guide');
set(handles.Acerca_de,'Label','About');

%El resto de etiquetas y textos de la interfaz se muestran en inglés.
if get(handles.cargarPETTAC,'UserData')==1
    set(handles.botonTAC,'String','CT');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','CT');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','PET-CT');
    set(handles.textoPETTAC,'String','PET-CT');
    set(handles.textoMedidasTAC,'String','CT');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','CT');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','PET-CT');
elseif get(handles.cargarRMTAC,'UserData')==1
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','CT');
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','CT');
    set(handles.textoInfopettac,'String','MR-CT');
    set(handles.textoPETTAC,'String','MR-CT');
    set(handles.textoMedidasTAC,'String','MR');
    set(handles.textoMedidasPET,'String','CT');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','CT');
    set(handles.checkboxPETTAC,'String','MR-CT');
elseif get(handles.cargarRMPET,'UserData')==1
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','MR-PET');
    set(handles.textoPETTAC,'String','MR-PET');
    set(handles.textoMedidasTAC,'String','MR');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','MR-PET');
end

set(handles.textoCalidad,'String','QUALITY');
set(handles.textoMedia,'String','MEAN');
set(handles.botonDescwavelet,'String','Wavelet descomposition');
```

```
set(handles.botonGrafica,'String','Histogram');
set(handles.botonSuperficie,'String','Surface');
set(handles.botonMapapixeles,'String','Pixel region');
set(handles.botonIniciarexploracion,'String','Play');
set(handles.botonAnimacion,'String','Stop');
set(handles.textoColor,'String','Color');
set(handles.radioIntensidad,'String','Sharpness');
set(handles.radioIntensidadPET,'String','PET sharpness');
set(handles.botonFusionar,'String','Fusion');
set(handles.textoVentana,'String','Window');
set(handles.textoUmbral,'String','Threshold');
set(handles.textoMetodofusion,'String','Fusion scheme');
set(handles.textoNiveles,'String','Descomposition levels');
set(handles.textoDescomposicion,'String','Descomposition');

set(handles.panelImagenes,'Title','Images');
set(handles.panelFusion,'Title','Fusion');
set(handles.panelAjustes,'Title','Settings');
set(handles.panelCompleta,'Title','Full screen');
set(handles.panelGraficas,'Title','Figures');
set(handles.panelExtras,'Title','Full exploration');
set(handles.panelMedidas,'Title','Fusion measurements');


% --------------------------------------------------------------------
function Deutsch_Callback(hObject, eventdata, handles)
% hObject    handle to Deutsch (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

set(handles.Espanol,'UserData',0);
set(handles.English,'UserData',0);
set(handles.Deutsch,'UserData',1);

%Las etiquetas del menú principal se muestran en alemán.
set(handles.Archivo,'Label','Datei');
set(handles.Cargar,'Label','Laden');
set(handles.Salir,'Label','Verlassen');
set(handles.cargarPETTAC,'Label','PET-TAC fusion');
set(handles.cargarRMPET,'Label','RM-PET fusion');
set(handles.cargarRMTAC,'Label','RM-TAC fusion');
set(handles.Paciente_1,'Label','Patient 1');
set(handles.Paciente_2,'Label','Patient 2');
set(handles.Ver,'Label','Sehen');
set(handles.Idioma,'Label','Sprache');
set(handles.Ayuda,'Label','Hilfe');
set(handles.Guia,'Label','Richtschnur');
set(handles.Acerca_de,'Label','Über');

%El resto de etiquetas y textos de la interfaz se muestran en alemán.
if get(handles.cargarPETTAC,'UserData')==1
    set(handles.botonTAC,'String','CT');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','CT');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','PET-CT');
```

```
    set(handles.textoPETTAC,'String','PET-CT');
    set(handles.textoMedidasTAC,'String','CT');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','CT');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','PET-CT');
elseif get(handles.cargarRMTAC,'UserData')==1
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','CT');
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','CT');
    set(handles.textoInfopettac,'String','MR-CT');
    set(handles.textoPETTAC,'String','MR-CT');
    set(handles.textoMedidasTAC,'String','MR');
    set(handles.textoMedidasPET,'String','CT');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','CT');
    set(handles.checkboxPETTAC,'String','MR-CT');
elseif get(handles.cargarRMPET,'UserData')==1
    set(handles.botonTAC,'String','MR');
    set(handles.botonPET,'String','PET');
    set(handles.textoInfotac,'String','MR');
    set(handles.textoInfopet,'String','PET');
    set(handles.textoInfopettac,'String','MR-PET');
    set(handles.textoPETTAC,'String','MR-PET');
    set(handles.textoMedidasTAC,'String','MR');
    set(handles.textoMedidasPET,'String','PET');
    set(handles.checkboxTAC,'String','MR');
    set(handles.checkboxPET,'String','PET');
    set(handles.checkboxPETTAC,'String','MR-PET');
end

set(handles.textoCalidad,'String','BESCHAFFENHEIT');
set(handles.textoMedia,'String','MITTEL');
set(handles.botonDescwavelet,'String','Wavelet Zersetzung');
set(handles.botonGrafica,'String','Histogramm');
set(handles.botonSuperficie,'String','Fläche');
set(handles.botonMapapixeles,'String','Pixelskarte');
set(handles.botonIniciarexploracion,'String','Abspielen');
set(handles.botonAnimacion,'String','Stoppen');
set(handles.textoColor,'String','Farbe');
set(handles.radioIntensidad,'String','Schärfe');
set(handles.radioIntensidadPET,'String','PET Schärfe');
set(handles.botonFusionar,'String','Fusion');
set(handles.textoVentana,'String','Fenster');
set(handles.textoUmbral,'String','Schwelle');
set(handles.textoMetodofusion,'String','Fusionmethode');
set(handles.textoNiveles,'String','Zersetzungstande');
set(handles.textoDescomposicion,'String','Zersetzung');

set(handles.panelImagenes,'Title','Bilder');
set(handles.panelFusion,'Title','Fusion');
set(handles.panelAjustes,'Title','Einstellungen');
set(handles.panelCompleta,'Title','Vollbild');
set(handles.panelGraficas,'Title','Grafiken');
set(handles.panelExtras,'Title','Volluntersuchung');
set(handles.panelMedidas,'Title','Fusionmassnahme');
```

media_intensidades.m

```
function Z=media_intensidades(X,Y)
%La función media_intensidades calcula la media aritmética de los niveles
%de intensidad de gris de ambas imágenes en el dominio espacial.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

end

%Se comprueba si las matrices de las imágenes a fusionar no son
cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end

%Se calcula la media aritmética de los coeficientes en posiciones
%correspondientes en ambas matrices.
Z=(auxX+auxY)/2;

end
```

media.m

```
function Z=media(X,Y,wavelet,nivel)
%La función media calcula la media aritmética de los coeficientes
%wavelet obtenidos tras convertir las imágenes del dominio espacial al
%dominio wavelet.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.
%'wavelet' especifica con qué familia wavelet se hará la descomposición.
%'nivel' indica el nivel de descomposición.

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

end

%Se comprueba si las matrices de las imágenes a fusionar no son
cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end

%Se comprueba qué familia wavelet se quiere utilizar en la
descomposición.
switch wavelet

    case 1
        fam_wavelet=sprintf('haar');
    case 2
        fam_wavelet=sprintf('db2');
    case 3
        fam_wavelet=sprintf('sym2');
    case 4
        fam_wavelet=sprintf('coif2');
```

```
    case 5
        fam_wavelet=sprintf('bior2.2');
    case 6
        fam_wavelet=sprintf('rbio2.2');
    case 7
        fam_wavelet=sprintf('dmey');

end

%-----------------------------MEDIA-----------------------------------------
--

%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a las
%matrices de aproximación y de detalles horizontales, verticales y
%diagonales.

%Se calcula la media aritmética de las parejas de submatrices
%correspondientes.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
%para obtener la matriz final resultante.

%-------------------------------------------------------------------------
--

%NOTA: Para dos y tres niveles de descomposición se implementará
únicamente
%la wavelet de haar.

if nivel==1

    %Un nivel de descomposición.
    [xA,xH,xV,xD]=dwt2(auxX,fam_wavelet);
    [yA,yH,yV,yD]=dwt2(auxY,fam_wavelet);

    zA=(xA+yA)/2;
    zH=(xH+yH)/2;
    zV=(xV+yV)/2;
    zD=(xD+yD)/2;

    Z=idwt2(zA,zH,zV,zD,fam_wavelet);

elseif nivel==2

    %Segundo nivel de descomposición.
    [xA,xH,xV,xD]=dwt2(auxX,'haar');
    [yA,yH,yV,yD]=dwt2(auxY,'haar');

    [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
    [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');
```

```
    z2A=(x2A+y2A)/2;
    z2H=(x2H+y2H)/2;
    z2V=(x2V+y2V)/2;
    z2D=(x2D+y2D)/2;

    zA=idwt2(z2A,z2H,z2V,z2D,'haar');
    zH=(xH+yH)/2;
    zV=(xV+yV)/2;
    zD=(xD+yD)/2;

    Z=idwt2(zA,zH,zV,zD,'haar');

elseif nivel==3

    %Tercer nivel de descomposición.
    [xA,xH,xV,xD]=dwt2(auxX,'haar');
    [yA,yH,yV,yD]=dwt2(auxY,'haar');

    [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
    [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

    [x3A,x3H,x3V,x3D]=dwt2(x2A,'haar');
    [y3A,y3H,y3V,y3D]=dwt2(y2A,'haar');

    z3A=(x3A+y3A)/2;
    z3H=(x3H+y3H)/2;
    z3V=(x3V+y3V)/2;
    z3D=(x3D+y3D)/2;

    z2A=idwt2(z3A,z3H,z3V,z3D,'haar');
    z2H=(x2H+y2H)/2;
    z2V=(x2V+y2V)/2;
    z2D=(x2D+y2D)/2;

    zA=idwt2(z2A,z2H,z2V,z2D,'haar');
    zH=(xH+yH)/2;
    zV=(xV+yV)/2;
    zD=(xD+yD)/2;

    Z=idwt2(zA,zH,zV,zD,'haar');

end

end
```

media_ponderada_gaussiana.m

```
function Z=media_ponderada_gaussiana(X,Y,ventana,wavelet,nivel)
%La función media_ponderada_gaussiana calcula una media ponderada por una
%matriz de pesos gaussiana sobre los coeficientes wavelet obtenidos tras
%convertir las imágenes del dominio espacial al dominio wavelet.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.
%'ventana' es el tamaño de la ventana, queda a opción del usuario. Se da
la
%opción a elegir entre 3, 5, 11 y 21.
%'wavelet' especifica con qué familia wavelet se hará la descomposición
%'nivel' indica el nivel de descomposición

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

    [p,q]=size(Y);

end

%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end

%Se comprueba el tamaño de ventana escogido.
switch ventana

    case 1
        S=3;
    case 2
```

```
        S=5;
    case 3
        S=11;
    case 4
        S=21;

end

%Se comprueba qué familia wavelet se quiere utilizar en la
descomposición.
switch wavelet

    case 1
        fam_wavelet=sprintf('haar');
    case 2
        fam_wavelet=sprintf('db2');
    case 3
        fam_wavelet=sprintf('sym2');
    case 4
        fam_wavelet=sprintf('coif2');
    case 5
        fam_wavelet=sprintf('bior2.2');
    case 6
        fam_wavelet=sprintf('rbio2.2');
    case 7
        fam_wavelet=sprintf('dmey');

end

%--------------------MEDIA PONDERADA GAUSSIANA--------------------------
--

%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a las
%matrices de aproximación y de detalles horizontales, verticales y
%diagonales.

%Se pondera una pequeña ventana de coeficientes en cada una de las
%submatrices de detalles por una matriz de pesos gaussiana y a
continuación
%se calcula un promedio.
%Para las submatrices de aproximación se calcula la media aritmética.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
%para obtener la matriz final resultante.

%-----------------------------------------------------------------------
--

%NOTA: Para dos y tres niveles de descomposición se implementará
únicamente
%la wavelet de haar.
```

```
if nivel==3

    %Tercer nivel de descomposición
    [xA,xH,xV,xD]=dwt2(auxX,'haar');
    [yA,yH,yV,yD]=dwt2(auxY,'haar');

    [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
    [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

    [x3A,x3H,x3V,x3D]=dwt2(x2A,'haar');
    [y3A,y3H,y3V,y3D]=dwt2(y2A,'haar');

    %Media aritmética de las matrices de aproximación en el tercer nivel.
    z3A=(x3A+y3A)/2;

    %Se genera una matriz gaussiana del mismo tamaño de la ventana que
    %va a ir rodeando a cada coeficiente durante el recorrido de la
matriz.
    W=gausswin(S)*gausswin(S)';

    %NOTA: x3H, x3V, x3D, y3H, y3V, y3D son del mismo tamaño.
    [f,c]=size(x3H);

    a3H=zeros(f+(S-1),c+(S-1));
    b3H=a3H;c3H=a3H;a3V=a3H;b3V=a3H;c3V=a3H;a3D=a3H;b3D=a3H;c3D=a3H;

    a3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3H;
    b3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3H;

    a3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3V;
    b3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3V;

    a3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3D;
    b3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3D;

    %Se selecciona una ventana de coeficientes de tamaño SxS centrada en
    %cada coeficiente de las submatrices de detalles horizontales,
    %verticales y diagonales.
    for i=((S-1)/2)+1:1:f+((S-1)/2)
        for j=((S-1)/2)+1:1:c+((S-1)/2)

            %Matrices de detalles horizontales
            suba3H=a3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb3H=b3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

            auxz3H=(suba3H.*W+subb3H.*W)/2;
            auxz3H=auxz3H/sum(W(:));
            c3H(i,j)=sum(auxz3H(:));

            %Matrices de detalles verticales
            suba3V=a3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb3V=b3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
```

```
            auxz3V=(suba3V.*W+subb3V.*W)/2;
            auxz3V=auxz3V/sum(W(:));
            c3V(i,j)=sum(auxz3V(:));

            %Matrices de detalles diagonales
            suba3D=a3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb3D=b3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

            auxz3D=(suba3D.*W+subb3D.*W)/2;
            auxz3D=auxz3D/sum(W(:));
            c3D(i,j)=sum(auxz3D(:));

        end
    end

    z3H=c3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z3V=c3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z3D=c3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

    z2A=idwt2(z3A,z3H,z3V,z3D,'haar');

end

if nivel~=1

    if nivel==2

        %Segundo nivel de descomposición
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        %Media aritmética de las matrices de aproximación en el segundo
nivel.
        z2A=(x2A+y2A)/2;

        %Se genera una matriz Gaussiana del mismo tamaño de la ventana
que
        %va a ir rodeando a cada coeficiente durante el recorrido de la
matriz.
        W=gausswin(S)*gausswin(S)';

    end

    %NOTA: x2H, x2V, x2D, y2H, y2V, y2D son del mismo tamaño.
    [f,c]=size(x2H);

    a2H=zeros(f+(S-1),c+(S-1));
    b2H=a2H;c2H=a2H;a2V=a2H;b2V=a2H;c2V=a2H;a2D=a2H;b2D=a2H;c2D=a2H;

    a2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2H;
    b2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2H;
```

```
    a2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2V;
    b2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2V;

    a2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2D;
    b2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2D;

    %Se selecciona una ventana de coeficientes de tamaño SxS centrada en
    %cada coeficiente de las submatrices de detalles horizontales,
    %verticales y diagonales.
    for i=((S-1)/2)+1:1:f+((S-1)/2)
        for j=((S-1)/2)+1:1:c+((S-1)/2)

            %Matrices de detalles horizontales
            suba2H=a2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb2H=b2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

            auxz2H=(suba2H.*W+subb2H.*W)/2;
            auxz2H=auxz2H/sum(W(:));
            c2H(i,j)=sum(auxz2H(:));

            %Matrices de detalles verticales
            suba2V=a2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb2V=b2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

            auxz2V=(suba2V.*W+subb2V.*W)/2;
            auxz2V=auxz2V/sum(W(:));
            c2V(i,j)=sum(auxz2V(:));

            %Matrices de detalles diagonales
            suba2D=a2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb2D=b2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

            auxz2D=(suba2D.*W+subb2D.*W)/2;
            auxz2D=auxz2D/sum(W(:));
            c2D(i,j)=sum(auxz2D(:));

        end
    end

    z2H=c2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2V=c2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2D=c2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

    zA=idwt2(z2A,z2H,z2V,z2D,'haar');

end

if nivel==1

    %Primer nivel de descomposición
    [xA,xH,xV,xD]=dwt2(auxX,fam_wavelet);
    [yA,yH,yV,yD]=dwt2(auxY,fam_wavelet);
```

```
    %Se calcula la media aritmética de las matrices de aproximación.
    zA=(xA+yA)/2;

end

%Se genera una matriz gaussiana del mismo tamaño de la ventana que va a
%ir rodeando a cada coeficiente durante el recorrido de la matriz.
W=gausswin(S)*gausswin(S)';

%NOTA: xH, xV, xD, yH, yV, yD son del mismo tamaño.
[f,c]=size(xH);

aH=zeros(f+(S-1),c+(S-1));
bH=aH;cH=aH;aV=aH;bV=aH;cV=aH;aD=aH;bD=aH;cD=aH;

aH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xH;
bH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yH;

aV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xV;
bV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yV;

aD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xD;
bD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yD;

%Se selecciona una ventana de coeficientes de tamaño SxS centrada en
%cada coeficiente de las submatrices de detalles horizontales,
%verticales y diagonales.
for i=((S-1)/2)+1:1:f+((S-1)/2)
    for j=((S-1)/2)+1:1:c+((S-1)/2)

        %Matrices de detalles horizontales
        subaH=aH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbH=bH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        auxzH=(subaH.*W+subbH.*W)/2;
        auxzH=auxzH/sum(W(:));
        cH(i,j)=sum(auxzH(:));

        %Matrices de detalles verticales
        subaV=aV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbV=bV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        auxzV=(subaV.*W+subbV.*W)/2;
        auxzV=auxzV/sum(W(:));
        cV(i,j)=sum(auxzV(:));

        %Matrices de detalles diagonales
        subaD=aD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbD=bD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        auxzD=(subaD.*W+subbD.*W)/2;
        auxzD=auxzD/sum(W(:));
        cD(i,j)=sum(auxzD(:));
```

```
    end
end

zH=cH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zV=cV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zD=cD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

if nivel==1

    Z=idwt2(zA,zH,zV,zD,fam_wavelet);

else

    Z=idwt2(zA,zH,zV,zD,'haar');

end

end
```

---

media_ponderada_laplaciana.m

```
function Z=media_ponderada_laplaciana(X,Y,wavelet)
%La función media_ponderada_laplaciana calcula una media ponderada por
una
%matriz de pesos laplaciana sobre los coeficientes wavelet obtenidos tras
%convertir las imágenes del dominio espacial al dominio wavelet.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.
%'wavelet' especifica con qué familia wavelet se hará la descomposición

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

    [p,q]=size(Y);

end
```

```
%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end

%Se comprueba qué familia wavelet se quiere utilizar en la
descomposición.
switch wavelet

    case 1
        fam_wavelet=sprintf('haar');
    case 2
        fam_wavelet=sprintf('db2');
    case 3
        fam_wavelet=sprintf('sym2');
    case 4
        fam_wavelet=sprintf('coif2');
    case 5
        fam_wavelet=sprintf('bior2.2');
    case 6
        fam_wavelet=sprintf('rbio2.2');
    case 7
        fam_wavelet=sprintf('dmey');

end

%--------------------MEDIA PONDERADA LAPLACIANA--------------------------
--

%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a las
%matrices de aproximación y de detalles horizontales, verticales y
%diagonales.

%Se pondera una pequeña ventana de coeficientes en cada una de las
%submatrices de detalles por una matriz de pesos laplaciana y a
continuación
%se calcula un promedio.
%Para las submatrices de aproximación se calcula la media aritmética.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
```

```
%para obtener la matriz final resultante.

%-------------------------------------------------------------------------
--

%NOTA: Se implementará únicamente para un nivel de descomposición.

[xA,xH,xV,xD]=dwt2(auxX,fam_wavelet);
[yA,yH,yV,yD]=dwt2(auxY,fam_wavelet);

%Media aritmética de las matrices de aproximación.
zA=(xA+yA)/2;

%Tamaño de ventana S=3.
S=3;

%Se genera una matriz laplaciana de tamaño 3x3.
L=[-1 -1 -1;-1 8 -1;-1 -1 -1];

%NOTA: xH, xV, xD, yH, yV, yD son del mismo tamaño.
[f,c]=size(xH);

aH=zeros(f+(S-1),c+(S-1));
bH=aH;cH=aH;aV=aH;bV=aH;cV=aH;aD=aH;bD=aH;cD=aH;

aH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xH;
bH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yH;

aV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xV;
bV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yV;

aD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xD;
bD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yD;

%Se selecciona una ventana de coeficientes de tamaño 3x3 centrada en
%cada coeficiente de las submatrices de detalles horizontales,
%verticales y diagonales.
for i=((S-1)/2)+1:1:f+((S-1)/2)
    for j=((S-1)/2)+1:1:c+((S-1)/2)

        %Matrices de detalles horizontales
        subaH=aH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbH=bH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        auxzH=(subaH.*L+subbH.*L)/2;
        cH(i,j)=sum(auxzH(:));

        %Matrices de detalles verticales
        subaV=aV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbV=bV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        auxzV=(subaV.*L+subbV.*L)/2;
        cV(i,j)=sum(auxzV(:));
```

```
        %Matrices de detalles diagonales
        subaD=aD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbD=bD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        auxzD=(subaD.*L+subbD.*L)/2;
        cD(i,j)=sum(auxzD(:));

    end
end

zH=cH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zV=cV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zD=cD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

Z=idwt2(zA,zH,zV,zD,fam_wavelet);

end
```

---

media_imagenes_multirresolucion.m

```
function Z=media_imagenes_multirresolucion(X,Y,nivel)
%La función media_coeficientes_wavelet_wTAC calcula la media de los
%coeficientes de la matriz de aproximación de una de las imágenes (la que
%se considera que más información puede aportar), una vez se ha
convertido
%ésta al dominio wavelet, y los coeficientes de la otra imagen en el
%dominio espacial.
%'nivel' indica el nivel de descomposición.

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    interpY=interp2(Y,x,y);

else

    interpY=Y;
```

```
end

%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=interpY;

else

    auxX=X;
    auxY=interpY;

end

%--------------------MEDIA IMÁGENES MULTIRRESOLUCIÓN----------------------
--

%Se aplica la transformada wavelet discreta en dos dimensiones a la
%matriz X que corresponde a una de las dos imágenes a fusionar (el TAC
para
%la fusión PET-TAC y la resonancia magnética para la fusión MR-TAC y MR-
PET).
%Tras esa descomposición se obtendrán cuatro submatrices xA, xH, xV, xD
%correspondientes a las matrices de aproximación y de detalles
horizontales,
%verticales y diagonales.

%Se calcula la media aritmética de los coeficientes de la submatriz de
%aproximación xA y de la matriz original Y de la otra imagen.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
%a la matriz promediada y las otras tres submatrices de detalles xH, xV y
xD
%de la imagen contenida en la variable X.

%------------------------------------------------------------------------
--

[xA,xH,xV,xD]=dwt2(auxX,'haar');
[yA,yH,yV,yD]=dwt2(auxY,'haar');

%Se hacen coincidir los tamaños de las matrices a promediar.
if size(X)>size(Y)

    %Se interpola Y por un factor L=2.
    [x,y]=meshgrid(linspace(1,p,m/2),linspace(1,p,m/2));
    Y2=interp2(Y,x,y);

else
```

```
    %Se diezma Y por un factor M=2 porque en este caso la matriz de la
    %imagen original Y es más grande que la submatriz de aproximación xA
    %obtenida tras la descomposición wavelet.
    Y2=auxY(1:2:end,1:2:end);

end

maxyA=max(yA(:));

%Se promedia para el nivel de descomposición seleccionado.
%Previamente, y puesto que se van a promediar coeficientes wavelet y
%coeficientes correspondientes a niveles de intensidad de gris en el
%dominio espacial, se realizan las operaciones necesarias para que ambos
%conjuntos de coeficientes estén en un rango de valores similar.
if nivel==1

    %Un nivel de descomposición.
    Y2=Y2*maxyA/255;

    xA=(xA+Y2)/2;

elseif nivel==2

    %Dos niveles de descomposición.
    [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');

    [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

    maxy2A=max(y2A(:));

    %Se diezma Y por un factor M=4
    Y4=auxY(1:4:end,1:4:end);

    Y4=Y4*maxy2A/255;

    x2A=(x2A+Y4)/2;

    xA=idwt2(x2A,x2H,x2V,x2D,'haar');

elseif nivel==3

    %Tres niveles de descomposición.
    [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');

    [x3A,x3H,x3V,x3D]=dwt2(x2A,'haar');

    [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

    [y3A,y3H,y3V,y3D]=dwt2(y2A,'haar');

    maxy3A=max(y3A(:));
```

```
    %Se diezma Y por un factor M=8
    Y8=auxY(1:8:end,1:8:end);

    Y8=Y8*maxy3A/255;

    x3A=(x3A+Y8)/2;

    x2A=idwt2(x3A,x3H,x3V,x3D,'haar');

    xA=idwt2(x2A,x2H,x2V,x2D,'haar');

end

Z=idwt2(xA,xH,xV,xD,'haar');

end
```

```
function Z=varianza_ventana(X,Y,ventana,umb,wavelet,nivel)
%La función varianza_ventana calcula la varianza sobre una pequeña
ventana
%de coeficientes en submatrices correspondientes. Compara ambas
varianzas;
%si son similares (esa similitud queda sujeta a un umbral fijado) calcula
%el promedio de los píxeles centrales de ambas ventanas; en caso
contrario
%se elige el píxel central de la ventana con mayor varianza.
%Todo ello tras convertir previamente las imágenes originales del dominio
%espacial al dominio wavelet.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.
%'ventana' es el tamaño de la ventana, queda a opción del usuario. Se da
la
%opción a elegir entre 3, 5, 11 y 21.
%'umb' será el umbral elegido para comparar las varianzas.
%'wavelet' especifica con qué familia wavelet se hará la descomposición.
%'nivel' indica el nivel de descomposición.

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);
```

```
    [p,q]=size(Y);

end

%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end

%Se comprueba el tamaño de ventana escogido.
switch ventana

    case 1
        S=3;
    case 2
        S=5;
    case 3
        S=11;
    case 4
        S=21;

end

%Se comprueba el umbral escogido para comparar las varianzas.
switch umb

    case 1
        umbral=1;
    case 2
        umbral=5;
    case 3
        umbral=15;
    case 4
        umbral=30;

end

%Se comprueba qué familia wavelet se quiere utilizar en la
descomposición.
switch wavelet

    case 1
        fam_wavelet=sprintf('haar');
```

```
    case 2
        fam_wavelet=sprintf('db2');
    case 3
        fam_wavelet=sprintf('sym2');
    case 4
        fam_wavelet=sprintf('coif2');
    case 5
        fam_wavelet=sprintf('bior2.2');
    case 6
        fam_wavelet=sprintf('rbio2.2');
    case 7
        fam_wavelet=sprintf('dmey');

end

%--------------------VARIANZA VENTANA-----------------------------------
--

%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a las
%matrices de aproximación y de detalles horizontales, verticales y
%diagonales.

%Se calcula la varianza sobre una pequeña ventana de coeficientes en cada
%una de las submatrices de detalles y se elige el coeficiente según
indica
%la regla de fusión.
%Para las submatrices de aproximación se calcula la media aritmética.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
%para obtener la matriz final resultante.

%-------------------------------------------------------------------------
--

%NOTA: Para dos y tres niveles de descomposición se implementará
únicamente
%la wavelet de haar.

if nivel==3

        %Tercer nivel de descomposición.
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        [x3A,x3H,x3V,x3D]=dwt2(x2A,'haar');
        [y3A,y3H,y3V,y3D]=dwt2(y2A,'haar');
```

```
        %Media aritmética de las matrices de aproximación en el tercer
nivel.
        z3A=(x3A+y3A)/2;

        %NOTA: x3H, x3V, x3D, y3H, y3V, y3D son del mismo tamaño.
        [f,c]=size(x3H);

        a3H=zeros(f+(S-1),c+(S-1));
        b3H=a3H;c3H=a3H;a3V=a3H;b3V=a3H;c3V=a3H;a3D=a3H;b3D=a3H;c3D=a3H;

        a3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3H;
        b3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3H;

        a3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3V;
        b3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3V;

        a3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3D;
        b3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3D;

        %Se selecciona una ventana de coeficientes de tamaño SxS centrada
en
        %cada coeficiente de las submatrices de detalles horizontales,
        %verticales y diagonales.
        for i=((S-1)/2)+1:1:f+((S-1)/2)
            for j=((S-1)/2)+1:1:c+((S-1)/2)

                %Matrices de detalles horizontales.
                suba3H=a3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3H=b3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                %Matrices de detalles verticales.
                suba3V=a3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3V=b3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                %Matrices de detalles diagonales.
                suba3D=a3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3D=b3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                varianza_x3H=var(suba3H(:));
                varianza_x3V=var(suba3V(:));
                varianza_x3D=var(suba3D(:));

                varianza_y3H=var(subb3H(:));
                varianza_y3V=var(subb3V(:));
                varianza_y3D=var(subb3D(:));

                %Se comprueba si los valores de las varianzas en
subimágenes
                %correspondientes son similares.
```

```
%Para el par de subimágenes de detalles horizontales.
if (abs(varianza_x3H-varianza_y3H))<=umbral

    %Son similares, se toma el valor medio de los píxeles
    %centrales.
    c3H(i,j)=(a3H(i,j)+b3H(i,j))/2;

else

    %No son similares, se toma el valor del píxel central
    %de la ventana con mayor valor.
    if varianza_x3H>=varianza_y3H

        c3H(i,j)=a3H(i,j);

    else

        c3H(i,j)=b3H(i,j);

    end

end

%Para el par de subimágenes de detalles verticales.
if (abs(varianza_x3V-varianza_y3V))<=umbral

    %Son similares, se toma el valor medio de los píxeles
    %centrales.
    c3V(i,j)=(a3V(i,j)+b3V(i,j))/2;

else

    %No son similares, se toma el valor del píxel central
    %de la ventana con mayor valor.
    if varianza_x3V>=varianza_y3V

        c3V(i,j)=a3V(i,j);

    else

        c3V(i,j)=b3V(i,j);

    end

end

%Para el par de subimágenes de detalles diagonales.
if (abs(varianza_x3D-varianza_y3D))<=umbral

    %Son similares, se toma el valor medio de los píxeles
    %centrales.
    c3D(i,j)=(a3D(i,j)+b3D(i,j))/2;
```

```
                else

                    %No son similares, se toma el valor del píxel central
                    %de la ventana con mayor valor.
                    if varianza_x3D>=varianza_y3D

                        c3D(i,j)=a3D(i,j);

                    else

                        c3D(i,j)=b3D(i,j);

                    end

                end

            end
        end

        z3H=c3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
        z3V=c3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
        z3D=c3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

        z2A=idwt2(z3A,z3H,z3V,z3D,'haar');

end

if nivel~=1

    if nivel==2

        %Segundo nivel de descomposición.
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        %Media aritmética de las matrices de aproximación en el segundo
nivel.
        z2A=(x2A+y2A)/2;

    end

    %NOTA: x2H, x2V, x2D, y2H, y2V, y2D son del mismo tamaño.
    [f,c]=size(x2H);

    a2H=zeros(f+(S-1),c+(S-1));
    b2H=a2H;c2H=a2H;a2V=a2H;b2V=a2H;c2V=a2H;a2D=a2H;b2D=a2H;c2D=a2H;

    a2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2H;
    b2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2H;
```

```
a2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2V;
b2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2V;

a2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2D;
b2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2D;

%Se selecciona una ventana de coeficientes de tamaño SxS centrada
%en cada coeficiente de las submatrices de detalles horizontales,
%verticales y diagonales.
for i=((S-1)/2)+1:1:f+((S-1)/2)
    for j=((S-1)/2)+1:1:c+((S-1)/2)

        %Matrices de detalles horizontales.
        suba2H=a2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subb2H=b2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles verticales.
        suba2V=a2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subb2V=b2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles diagonales.
        suba2D=a2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subb2D=b2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        varianza_x2H=var(suba2H(:));
        varianza_x2V=var(suba2V(:));
        varianza_x2D=var(suba2D(:));

        varianza_y2H=var(subb2H(:));
        varianza_y2V=var(subb2V(:));
        varianza_y2D=var(subb2D(:));

        %Se comprueba si los valores de las varianzas en subimágenes
        %correspondientes son similares.

        %Para el par de subimágenes de detalles horizontales.
        if (abs(varianza_x2H-varianza_y2H))<=umbral

            %Son similares, se toma el valor medio de los píxeles
            %centrales.
            c2H(i,j)=(a2H(i,j)+b2H(i,j))/2;

        else

            %No son similares, se toma el valor del píxel central
            %de la ventana con mayor valor.
            if varianza_x2H>=varianza_y2H

                c2H(i,j)=a2H(i,j);

            else

                c2H(i,j)=b2H(i,j);
```

```
            end

        end

        %Para el par de subimágenes de detalles verticales.
        if (abs(varianza_x2V-varianza_y2V))<=umbral

            %Son similares, se toma el valor medio de los píxeles
            %centrales.
            c2V(i,j)=(a2V(i,j)+b2V(i,j))/2;

        else

            %No son similares, se toma el valor del píxel central
            %de la ventana con mayor valor.
            if varianza_x2V>=varianza_y2V

                c2V(i,j)=a2V(i,j);

            else

                c2V(i,j)=b2V(i,j);

            end

        end

        %Para el par de subimágenes de detalles diagonales.
        if (abs(varianza_x2D-varianza_y2D))<=umbral

            %Son similares, se toma el valor medio de los píxeles
            %centrales.
            c2D(i,j)=(a2D(i,j)+b2D(i,j))/2;

        else

           %No son similares, se toma el valor del píxel central
           %de la ventana con mayor valor.
           if varianza_x2D>=varianza_y2D

               c2D(i,j)=a2D(i,j);

           else

               c2D(i,j)=b2D(i,j);

           end

        end

    end
end
```

```
    z2H=c2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2V=c2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2D=c2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

    zA=idwt2(z2A,z2H,z2V,z2D,'haar');

end

if nivel==1

    %Primer nivel de descomposición
    [xA,xH,xV,xD]=dwt2(auxX,fam_wavelet);
    [yA,yH,yV,yD]=dwt2(auxY,fam_wavelet);

    %Se calcula la media aritmética de las matrices de aproximación.
    zA=(xA+yA)/2;

end

%NOTA: xH, xV, xD, yH, yV, yD son del mismo tamaño.
[f,c]=size(xH);

aH=zeros(f+(S-1),c+(S-1));
bH=aH;cH=aH;aV=aH;bV=aH;cV=aH;aD=aH;bD=aH;cD=aH;

aH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xH;
bH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yH;

aV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xV;
bV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yV;

aD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xD;
bD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yD;

%Se selecciona una ventana de coeficientes de tamaño SxS centrada en
%cada coeficiente de las submatrices de detalles horizontales,
%verticales y diagonales.
for i=((S-1)/2)+1:1:f+((S-1)/2)
    for j=((S-1)/2)+1:1:c+((S-1)/2)

        %Matrices de detalles horizontales
        subaH=aH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbH=bH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles verticales
        subaV=aV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbV=bV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles diagonales
        subaD=aD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbD=bD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Se calcula la varianza de esas pequeñas ventanas de coeficientes
```

```
        varianza_xH=var(subaH(:));
        varianza_xV=var(subaV(:));
        varianza_xD=var(subaD(:));

        varianza_yH=var(subbH(:));
        varianza_yV=var(subbV(:));
        varianza_yD=var(subbD(:));

        %Se comprueba si los valores de las varianzas en subimágenes
        %correspondientes son similares

        %Para el par de subimágenes de detalles horizontales
        if (abs(varianza_xH-varianza_yH))<=umbral

            %Son similares, se toma el valor medio de los píxeles
centrales
            cH(i,j)=(aH(i,j)+bH(i,j))/2;

        else

            %No son similares, se toma el valor del píxel central de la
            %ventana con mayor valor
            if varianza_xH>=varianza_yH

                cH(i,j)=aH(i,j);

            else

                cH(i,j)=bH(i,j);

            end

        end

        %Para el par de subimágenes de detalles verticales
        if (abs(varianza_xV-varianza_yV))<=umbral

            %Son similares, se toma el valor medio de los píxeles
centrales
            cV(i,j)=(aV(i,j)+bV(i,j))/2;

        else

            %No son similares, se toma el valor del píxel central de la
            %ventana con mayor valor
            if varianza_xV>=varianza_yV

                cV(i,j)=aV(i,j);

            else

                cV(i,j)=bV(i,j);
```

```
            end

        end

        %Para el par de subimágenes de detalles diagonales
        if (abs(varianza_xD-varianza_yD))<=umbral

            %Son similares, se toma el valor medio de los píxeles
            %centrales
            cD(i,j)=(aD(i,j)+bD(i,j))/2;

        else

           %No son similares, se toma el valor del píxel central
           %de la ventana con mayor valor
           if varianza_xD>=varianza_yD

               cD(i,j)=aD(i,j);

           else

               cD(i,j)=bD(i,j);

           end

        end

    end
end

zH=cH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zV=cV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zD=cD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

if nivel==1

    Z=idwt2(zA,zH,zV,zD,fam_wavelet);

else

    Z=idwt2(zA,zH,zV,zD,'haar');

end

end
```

coeficiente_variacion_ventana.m

```
function Z=coeficiente_variacion_ventana(X,Y,ventana,umb,wavelet,nivel)
%La función coeficiente_variacion_ventana calcula el coeficiente de
%variación sobre una pequeña ventana de coeficientes en submatrices
%correspondientes. Compara ambos valores; si son similares (esa similitud
%queda sujeta a un umbral fijado) calcula el promedio de los píxeles
%centrales de ambas ventanas; en caso contrario se elige el píxel central
%de la ventana con mayor coeficiente de variación.
%Todo ello tras convertir previamente las imágenes originales del dominio
%espacial al dominio wavelet.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.
%'ventana' es el tamaño de la ventana, queda a opción del usuario. Se da
la
%opción a elegir entre 3, 5, 11 y 21.
%'umb' será el umbral elegido para comparar las varianzas.
%'wavelet' especifica con qué familia wavelet se hará la descomposición.
%'nivel' indica el nivel de descomposición.

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

    [p,q]=size(Y);

end

%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end
```

```
%Se comprueba el tamaño de ventana escogido.
switch ventana

    case 1
        S=3;
    case 2
        S=5;
    case 3
        S=11;
    case 4
        S=21;

end

%Se comprueba el umbral escogido para comparar las varianzas.
switch umb

    case 1
        umbral=1;
    case 2
        umbral=5;
    case 3
        umbral=15;
    case 4
        umbral=30;

end

%Se comprueba qué familia wavelet se quiere utilizar en la
descomposición.
switch wavelet

    case 1
        fam_wavelet=sprintf('haar');
    case 2
        fam_wavelet=sprintf('db2');
    case 3
        fam_wavelet=sprintf('sym2');
    case 4
        fam_wavelet=sprintf('coif2');
    case 5
        fam_wavelet=sprintf('bior2.2');
    case 6
        fam_wavelet=sprintf('rbio2.2');
    case 7
        fam_wavelet=sprintf('dmey');

end

%--------------------COEFICIENTE VARIACIÓN VENTANA----------------------
--

%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
```

```
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a las
%matrices de aproximación y de detalles horizontales, verticales y
%diagonales.

%Se calcula el coeficiente de variación sobre una pequeña ventana de
%coeficientes en cada una de las submatrices de detalles y se elige el
%coeficiente según indica la regla de fusión.
%Para las submatrices de aproximación se calcula la media aritmética.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
%para obtener la matriz final resultante.

%-------------------------------------------------------------------------
--

%NOTA: Para dos y tres niveles de descomposición se implementará
únicamente
%la wavelet de haar.

if nivel==3

        %Tercer nivel de descomposición.
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        [x3A,x3H,x3V,x3D]=dwt2(x2A,'haar');
        [y3A,y3H,y3V,y3D]=dwt2(y2A,'haar');

        %Media aritmética de las matrices de aproximación en el tercer
nivel.
        z3A=(x3A+y3A)/2;

        %NOTA: x3H, x3V, x3D, y3H, y3V, y3D son del mismo tamaño.
        [f,c]=size(x3H);

        a3H=zeros(f+(S-1),c+(S-1));
        b3H=a3H;c3H=a3H;a3V=a3H;b3V=a3H;c3V=a3H;a3D=a3H;b3D=a3H;c3D=a3H;

        a3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3H;
        b3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3H;

        a3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3V;
        b3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3V;

        a3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3D;
        b3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3D;

        %Se selecciona una ventana de coeficientes de tamaño SxS centrada
en
        %cada coeficiente de las submatrices de detalles horizontales,
```

```
        %verticales y diagonales.
        for i=((S-1)/2)+1:1:f+((S-1)/2)
            for j=((S-1)/2)+1:1:c+((S-1)/2)

                %Matrices de detalles horizontales.
                suba3H=a3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3H=b3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                %Matrices de detalles verticales.
                suba3V=a3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3V=b3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                %Matrices de detalles diagonales.
                suba3D=a3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3D=b3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                cv_x3H=std(suba3H(:))/mean(suba3H(:));
                cv_y3H=std(subb3H(:))/mean(subb3H(:));

                cv_x3V=std(suba3V(:))/mean(suba3V(:));
                cv_y3V=std(subb3V(:))/mean(subb3V(:));

                cv_x3D=std(suba3D(:))/mean(suba3D(:));
                cv_y3D=std(subb3D(:))/mean(subb3D(:));

                %Se comprueba si los valores de las varianzas en
subimágenes
                %correspondientes son similares.

                %Para el par de subimágenes de detalles horizontales.
                if (abs(cv_x3H-cv_y3H))<=umbral

                    %Son similares, se toma el valor medio de los píxeles
                    %centrales.
                    c3H(i,j)=(a3H(i,j)+b3H(i,j))/2;

                else

                    %No son similares, se toma el valor del píxel central
                    %de la ventana con mayor valor.
                    if cv_x3H<=cv_y3H

                        c3H(i,j)=a3H(i,j);

                    else

                        c3H(i,j)=b3H(i,j);
```

```
            end

        end

        %Para el par de subimágenes de detalles verticales.
        if (abs(cv_x3V-cv_y3V))<=umbral

            %Son similares, se toma el valor medio de los píxeles
            %centrales.
            c3V(i,j)=(a3V(i,j)+b3V(i,j))/2;

        else

            %No son similares, se toma el valor del píxel central
            %de la ventana con mayor valor.
            if cv_x3V<=cv_y3V

                c3V(i,j)=a3V(i,j);

            else

                c3V(i,j)=b3V(i,j);

            end

        end

        %Para el par de subimágenes de detalles diagonales.
        if (abs(cv_x3D-cv_y3D))<=umbral

            %Son similares, se toma el valor medio de los píxeles
            %centrales.
            c3D(i,j)=(a3D(i,j)+b3D(i,j))/2;

        else

            %No son similares, se toma el valor del píxel central
            %de la ventana con mayor valor
            if cv_x3D<=cv_y3D

                c3D(i,j)=a3D(i,j);

            else

                c3D(i,j)=b3D(i,j);

            end

        end

    end
end
```

```
        z3H=c3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
        z3V=c3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
        z3D=c3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

        z2A=idwt2(z3A,z3H,z3V,z3D,'haar');

end

if nivel~=1

    if nivel==2

        %Segundo nivel de descomposición
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        %Media aritmética de las matrices de aproximación en el segundo
nivel.
        z2A=(x2A+y2A)/2;

    end

    %NOTA: x2H, x2V, x2D, y2H, y2V, y2D son del mismo tamaño.
    [f,c]=size(x2H);

    a2H=zeros(f+(S-1),c+(S-1));
    b2H=a2H;c2H=a2H;a2V=a2H;b2V=a2H;c2V=a2H;a2D=a2H;b2D=a2H;c2D=a2H;

    a2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2H;
    b2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2H;

    a2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2V;
    b2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2V;

    a2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2D;
    b2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2D;

    %Se selecciona una ventana de coeficientes de tamaño SxS centrada
    %en cada coeficiente de las submatrices de detalles horizontales,
    %verticales y diagonales.
    for i=((S-1)/2)+1:1:f+((S-1)/2)
        for j=((S-1)/2)+1:1:c+((S-1)/2)

            %Matrices de detalles horizontales.
            suba2H=a2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb2H=b2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

            %Matrices de detalles verticales.
            suba2V=a2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb2V=b2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
```

```
            %Matrices de detalles diagonales.
            suba2D=a2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
            subb2D=b2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

            cv_x2H=std(suba2H(:))/mean(suba2H(:));
            cv_y2H=std(subb2H(:))/mean(subb2H(:));

            cv_x2V=std(suba2V(:))/mean(suba2V(:));
            cv_y2V=std(subb2V(:))/mean(subb2V(:));

            cv_x2D=std(suba2D(:))/mean(suba2D(:));
            cv_y2D=std(subb2D(:))/mean(subb2D(:));

            %Se comprueba si los valores de las varianzas en subimágenes
            %correspondientes son similares.

            %Para el par de subimágenes de detalles horizontales.
            if (abs(cv_x2H-cv_y2H))<=umbral

                %Son similares, se toma el valor medio de los píxeles
                %centrales.
                c2H(i,j)=(a2H(i,j)+b2H(i,j))/2;

            else

                %No son similares, se toma el valor del píxel central de
la
                %ventana con mayor valor.
                if cv_x2H<=cv_y2H

                    c2H(i,j)=a2H(i,j);

                else

                    c2H(i,j)=b2H(i,j);

                end

            end

            %Para el par de subimágenes de detalles verticales.
            if (abs(cv_x2V-cv_y2V))<=umbral

                %Son similares, se toma el valor medio de los píxeles
                %centrales.
                c2V(i,j)=(a2V(i,j)+b2V(i,j))/2;

            else

                %No son similares, se toma el valor del píxel central de
la
                %ventana con mayor valor
                if cv_x2V<=cv_y2V
```

```
                    c2V(i,j)=a2V(i,j);

                else

                    c2V(i,j)=b2V(i,j);

                end

            end

            %Para el par de subimágenes de detalles diagonales.
            if (abs(cv_x2D-cv_y2D))<=umbral

                %Son similares, se toma el valor medio de los píxeles
                %centrales.
                c2D(i,j)=(a2D(i,j)+b2D(i,j))/2;

            else

                %No son similares, se toma el valor del píxel central
                %de la ventana con mayor valor.
                if cv_x2D<=cv_y2D

                    c2D(i,j)=a2D(i,j);

                else

                    c2D(i,j)=b2D(i,j);

                end

            end

        end
    end

    z2H=c2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2V=c2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2D=c2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

    zA=idwt2(z2A,z2H,z2V,z2D,'haar');

end

if nivel==1

    %Primer nivel de descomposición.
    [xA,xH,xV,xD]=dwt2(auxX,fam_wavelet);
    [yA,yH,yV,yD]=dwt2(auxY,fam_wavelet);

    %Se calcula la media aritmética de las matrices de aproximación.
    zA=(xA+yA)/2;
```

```
end

%NOTA: xH, xV, xD, yH, yV, yD son del mismo tamaño.
[f,c]=size(xH);

aH=zeros(f+(S-1),c+(S-1));
bH=aH;cH=aH;aV=aH;bV=aH;cV=aH;aD=aH;bD=aH;cD=aH;

aH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xH;
bH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yH;

aV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xV;
bV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yV;

aD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xD;
bD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yD;

%Se selecciona una ventana de coeficientes de tamaño SxS centrada en
%cada coeficiente de las submatrices de detalles horizontales,
%verticales y diagonales.
for i=((S-1)/2)+1:1:f+((S-1)/2)
    for j=((S-1)/2)+1:1:c+((S-1)/2)

        %Matrices de detalles horizontales.
        subaH=aH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbH=bH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles verticales.
        subaV=aV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbV=bV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles diagonales.
        subaD=aD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbD=bD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        cv_xH=std(subaH(:))/mean(subaH(:));
        cv_yH=std(subbH(:))/mean(subbH(:));

        cv_xV=std(subaV(:))/mean(subaV(:));
        cv_yV=std(subbV(:))/mean(subbV(:));

        cv_xD=std(subaD(:))/mean(subaD(:));
        cv_yD=std(subbD(:))/mean(subbD(:));

        %Se comprueba si los valores de los coeficientes de variación en
        %subimágenes correspondientes son similares.

        %Para el par de subimágenes de detalles horizontales.
        if (abs(cv_xH-cv_yH))<=umbral

            %Son similares, se toma el valor medio de los píxeles
            %centrales.
            cH(i,j)=(aH(i,j)+bH(i,j))/2;
```

```
else

    %No son similares, se toma el valor del píxel central de la
    %ventana con mayor valor.
    if cv_xH<=cv_yH

        cH(i,j)=aH(i,j);

    else

        cH(i,j)=bH(i,j);

    end

end

%Para el par de subimágenes de detalles verticales.
if (abs(cv_xV-cv_yV))<=umbral

    %Son similares, se toma el valor medio de los píxeles
    %centrales.
    cV(i,j)=(aV(i,j)+bV(i,j))/2;

else

    %No son similares, se toma el valor del píxel central de la
    %ventana con mayor valor.
    if cv_xV<=cv_yV

        cV(i,j)=aV(i,j);

    else

        cV(i,j)=bV(i,j);

    end

end

%Para el par de subimágenes de detalles diagonales.
if (abs(cv_xD-cv_yD))<=umbral

    %Son similares, se toma el valor medio de los píxeles
    %centrales.
    cD(i,j)=(aD(i,j)+bD(i,j))/2;

else

    %No son similares, se toma el valor del píxel central
    %de la ventana con mayor valor.
    if cv_xD<=cv_yD
```

```
                cD(i,j)=aD(i,j);

            else

                cD(i,j)=bD(i,j);

            end

        end

    end
end

zH=cH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zV=cV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zD=cD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

if nivel==1

    Z=idwt2(zA,zH,zV,zD,fam_wavelet);

else

    Z=idwt2(zA,zH,zV,zD,'haar');

end

end
```

---

nivel_actividad_coeficientes.m

```
function Z=nivel_actividad_coeficientes(X,Y,wavelet,nivel)
%La función nivel_actividad_coeficientes mide el nivel de actividad en
cada
%posición de la imagen, calculando para ello el valor absoluto de cada
%coeficiente. Se comparan los valores absolutos y se selecciona el de
%mayor valor.
%Todo ello tras convertir previamente las imágenes originales del dominio
%espacial al dominio wavelet.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.
%'wavelet' especifica con qué familia wavelet se hará la descomposición.
%'nivel' indica el nivel de descomposición.

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
```

```
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

    [p,q]=size(Y);

end

%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);

    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end

%Se comprueba qué familia wavelet se quiere utilizar en la
descomposición.
switch wavelet

    case 1
        fam_wavelet=sprintf('haar');
    case 2
        fam_wavelet=sprintf('db2');
    case 3
        fam_wavelet=sprintf('sym2');
    case 4
        fam_wavelet=sprintf('coif2');
    case 5
        fam_wavelet=sprintf('bior2.2');
    case 6
        fam_wavelet=sprintf('rbio2.2');
    case 7
        fam_wavelet=sprintf('dmey');

end

%--------------------NIVEL ACTIVIDAD COEFICIENTES-----------------------
--
```

```
%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a las
%matrices de aproximación y de detalles horizontales, verticales y
%diagonales.

%Se comparan las submatrices de detalles en valor absoluto y se
selecciona
%el de mayor valor.
%Para las submatrices de aproximación se calcula la media aritmética.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
%para obtener la matriz final resultante.

%-------------------------------------------------------------------------
--

%NOTA: Para dos y tres niveles de descomposición se implementará
únicamente
%la wavelet de haar.

if nivel==3

    %Tercer nivel de descomposición.
    [xA,xH,xV,xD]=dwt2(auxX,'haar');
    [yA,yH,yV,yD]=dwt2(auxY,'haar');

    [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
    [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

    [x3A,x3H,x3V,x3D]=dwt2(x2A,'haar');
    [y3A,y3H,y3V,y3D]=dwt2(y2A,'haar');

    %Media aritmética de las matrices de aproximación en el tercer nivel.
    z3A=(x3A+y3A)/2;

    %Se comparan las matrices de detalles en valor absoluto coeficiente a
    %coeficiente y se selecciona el de mayor valor.
    aux3H=abs(x3H)>=abs(y3H);
    aux3V=abs(x3V)>=abs(y3V);
    aux3D=abs(x3D)>=abs(y3D);

    z3H=(aux3H.*x3H)+((1-aux3H).*y3H);
    z3V=(aux3V.*x3V)+((1-aux3V).*y3V);
    z3D=(aux3D.*x3D)+((1-aux3D).*y3D);

    z2A=idwt2(z3A,z3H,z3V,z3D,'haar');

end

if nivel~=1
```

```
    if nivel==2

        %Segundo nivel de descomposición
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        %Media aritmética de las matrices de aproximación en el segundo
nivel.
        z2A=(x2A+y2A)/2;

    end

    %Se comparan las matrices de detalles en valor absoluto coeficiente a
    %coeficiente y se selecciona el de mayor valor.
    aux2H=abs(x2H)>=abs(y2H);
    aux2V=abs(x2V)>=abs(y2V);
    aux2D=abs(x2D)>=abs(y2D);

    z2H=(aux2H.*x2H)+((1-aux2H).*y2H);
    z2V=(aux2V.*x2V)+((1-aux2V).*y2V);
    z2D=(aux2D.*x2D)+((1-aux2D).*y2D);

    zA=idwt2(z2A,z2H,z2V,z2D,'haar');

end

if nivel==1

    [xA,xH,xV,xD]=dwt2(auxX,fam_wavelet);
    [yA,yH,yV,yD]=dwt2(auxY,fam_wavelet);

    %Se calcula la media aritmética de las matrices de aproximación.
    zA=(xA+yA)/2;

end

%Se comparan las matrices de detalles en valor absoluto coeficiente a
%coeficiente y se selecciona el de mayor valor.
auxH=abs(xH)>=abs(yH);
auxV=abs(xV)>=abs(yV);
auxD=abs(xD)>=abs(yD);

zH=(auxH.*xH)+((1-auxH).*yH);
zV=(auxV.*xV)+((1-auxV).*yV);
zD=(auxD.*xD)+((1-auxD).*yD);

if nivel==1

    Z=idwt2(zA,zH,zV,zD,fam_wavelet);

else
```

```
    Z=idwt2(zA,zH,zV,zD,'haar');

end

end
```

---

nivel_actividad_ventana.m

```
function Z=nivel_actividad_ventana(X,Y,ventana,wavelet,nivel)
%La función nivel_actividad_ventana mide el nivel de actividad de una
%pequeña ventana de coeficientes en cada imagen, calculando para ello el
%valor absoluto de los coeficientes de la ventana. Se comparan las
ventanas
%correspondientes de las submatrices de detalles y se selecciona el píxel
%central de la ventana que gane por mayoría.
%Todo ello tras convertir previamente las imágenes originales del dominio
%espacial al dominio wavelet.
%Las matrices correspondientes a ambas imágenes estarán contenidas en las
%variables X e Y.
%'ventana' es el tamaño de la ventana, queda a opción del usuario. Se da
la
%opción a elegir entre 3, 5, 11 y 21.
%'wavelet' especifica con qué familia wavelet se hará la descomposición.
%'nivel' indica el nivel de descomposición.

%Se obtiene el tamaño de las imágenes a fusionar.
[m,n]=size(X);
[p,q]=size(Y);

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

    [p,q]=size(Y);

end

%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n

    auxX=zeros(m);
    auxY=zeros(p);
```

```
    auxX(:,(((m-n)/2)+1):m-((m-n)/2))=X;
    auxY(:,(((p-q)/2)+1):p-((p-q)/2))=Y;

else

    auxX=X;
    auxY=Y;

end

%Se comprueba el tamaño de ventana escogido.
switch ventana
    case 1
        S=3;
    case 2
        S=5;
    case 3
        S=11;
    case 4
        S=21;
end

%Se comprueba qué familia wavelet se quiere utilizar en la
descomposición.
switch wavelet

    case 1
        fam_wavelet=sprintf('haar');
    case 2
        fam_wavelet=sprintf('db2');
    case 3
        fam_wavelet=sprintf('sym2');
    case 4
        fam_wavelet=sprintf('coif2');
    case 5
        fam_wavelet=sprintf('bior2.2');
    case 6
        fam_wavelet=sprintf('rbio2.2');
    case 7
        fam_wavelet=sprintf('dmey');

end

%--------------------NIVEL ACTIVIDAD VENTANA----------------------------
--

%Se aplica la transformada wavelet discreta en dos dimensiones a las
%matrices X e Y correspondientes a las dos imágenes a fusionar. Para
%cada una de las matrices X e Y se obtendrá una descomposición en
%cuatro submatrices xA, xH, xV, xD (en el caso de la matriz X), e
%yA, yH, yV, yD (para el caso de la matriz Y) correspondientes a las
%matrices de aproximación y de detalles horizontales, verticales y
%diagonales.
```

```
%Se comparan las ventanas en las submatrices de detalles en valor
absoluto
%y se elige el coeficiente según indica la regla de fusión.
%Para las submatrices de aproximación se calcula la media aritmética.

%Se realiza la transformada wavelet discreta inversa en dos dimensiones
%para obtener la matriz final resultante.

%-------------------------------------------------------------------------
--

%NOTA: Para dos y tres niveles de descomposición se implementará
únicamente
%la wavelet de haar.

if nivel==3

        %Tercer nivel de descomposición.
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        [x3A,x3H,x3V,x3D]=dwt2(x2A,'haar');
        [y3A,y3H,y3V,y3D]=dwt2(y2A,'haar');

        %Media aritmética de las matrices de aproximación en el tercer
nivel.
        z3A=(x3A+y3A)/2;

        %NOTA: x3H, x3V, x3D, y3H, y3V, y3D son del mismo tamaño.
        [f,c]=size(x3H);

        a3H=zeros(f+(S-1),c+(S-1));
        b3H=a3H;c3H=a3H;a3V=a3H;b3V=a3H;c3V=a3H;a3D=a3H;b3D=a3H;c3D=a3H;

        a3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3H;
        b3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3H;

        a3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3V;
        b3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3V;

        a3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x3D;
        b3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y3D;

        %Se calcula el valor absoluto de los coeficentes de cada
submatriz.
        absa3H=abs(a3H);
        absa3V=abs(a3V);
        absa3D=abs(a3D);

        absb3H=abs(b3H);
        absb3V=abs(b3V);
```

```
        absb3D=abs(b3D);

        %Se selecciona una ventana de coeficientes de tamaño SxS centrada
en
        %cada coeficiente de las submatrices de detalles horizontales,
        %verticales y diagonales.
        for i=((S-1)/2)+1:1:f+((S-1)/2)
            for j=((S-1)/2)+1:1:c+((S-1)/2)

                %Matrices de detalles horizontales
                suba3H=absa3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3H=absb3H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                %Matrices de detalles verticales
                suba3V=absa3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3V=absb3V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                %Matrices de detalles diagonales
                suba3D=absa3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
                subb3D=absb3D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

                %A continuación se comparan los coeficientes de las
ventanas
                %en posiciones correspondientes y se elige como píxel
                %central el de la ventana que gane por mayoría.
                %En caso de empate o igualdad se elige el píxel central
del
                %TAC en la fusión PET-TAC, y la resonancia magnética en
los
                %casos de fusión MR-PET y MR-TAC.

                %Matrices de detalles horizontales
                if (sum(sum(suba3H>=subb3H))>=((S^2)+1)/2)
                    c3H(i,j)=a3H(i,j);
                else
                    c3H(i,j)=b3H(i,j);
                end

                %Matrices de detalles verticales
                if (sum(sum(suba3V>=subb3V))>=((S^2)+1)/2)
                    c3V(i,j)=a3V(i,j);
                else
                    c3V(i,j)=b3V(i,j);
                end

                %Matrices de detalles diagonales
                if (sum(sum(suba3D>=subb3D))>=((S^2)+1)/2)
                    c3D(i,j)=a3D(i,j);
                else
```

```
                    c3D(i,j)=b3D(i,j);
                end

            end
        end

        z3H=c3H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
        z3V=c3V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
        z3D=c3D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

        z2A=idwt2(z3A,z3H,z3V,z3D,'haar');

end

if nivel~=1

    if nivel==2
        %Segundo nivel de descomposición
        [xA,xH,xV,xD]=dwt2(auxX,'haar');
        [yA,yH,yV,yD]=dwt2(auxY,'haar');

        [x2A,x2H,x2V,x2D]=dwt2(xA,'haar');
        [y2A,y2H,y2V,y2D]=dwt2(yA,'haar');

        %Media aritmética de las matrices de aproximación en el segundo
nivel.
        z2A=(x2A+y2A)/2;

    end

    %NOTA: x2H, x2V, x2D, y2H, y2V, y2D son del mismo tamaño.
    [f,c]=size(x2H);

    a2H=zeros(f+(S-1),c+(S-1));
    b2H=a2H;c2H=a2H;a2V=a2H;b2V=a2H;c2V=a2H;a2D=a2H;b2D=a2H;c2D=a2H;

    a2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2H;
    b2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2H;

    a2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2V;
    b2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2V;

    a2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=x2D;
    b2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=y2D;

    %Se calcula el valor absoluto de los coeficentes de cada submatriz.
    absa2H=abs(a2H);
    absa2V=abs(a2V);
    absa2D=abs(a2D);

    absb2H=abs(b2H);
    absb2V=abs(b2V);
    absb2D=abs(b2D);
```

```
    %Se selecciona una ventana de coeficientes de tamaño SxS centrada en
    %cada coeficiente de las submatrices de detalles horizontales,
    %verticales y diagonales.
    for i=((S-1)/2)+1:1:f+((S-1)/2)
        for j=((S-1)/2)+1:1:c+((S-1)/2)

            %Matrices de detalles horizontales
            suba2H=absa2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
            subb2H=absb2H(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

            %Matrices de detalles verticales
            suba2V=absa2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
            subb2V=absb2V(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

            %Matrices de detalles diagonales
            suba2D=absa2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));
            subb2D=absb2D(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-
1)/2));

            %A continuación se comparan los coeficientes de las ventanas
            %en posiciones correspondientes y se elige como píxel
            %central el de la ventana que gane por mayoría.
            %En caso de empate o igualdad se elige el píxel central del
            %TAC en la fusión PET-TAC, y la resonancia magnética en los
            %casos de fusión MR-PET y MR-TAC.

            %Matrices de detalles horizontales
            if (sum(sum(suba2H>=subb2H))>=((S^2)+1)/2)
                c2H(i,j)=a2H(i,j);
            else
                c2H(i,j)=b2H(i,j);
            end

            %Matrices de detalles verticales
            if (sum(sum(suba2V>=subb2V))>=((S^2)+1)/2)
                c2V(i,j)=a2V(i,j);
            else
                c2V(i,j)=b2V(i,j);
            end

            %Matrices de detalles diagonales
            if (sum(sum(suba2D>=subb2D))>=((S^2)+1)/2)
                c2D(i,j)=a2D(i,j);
            else
                c2D(i,j)=b2D(i,j);
            end

        end
    end
```

```
    z2H=c2H(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2V=c2V(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
    z2D=c2D(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

    zA=idwt2(z2A,z2H,z2V,z2D,'haar');

end

if nivel==1

    %Primer nivel de descomposición.
    [xA,xH,xV,xD]=dwt2(auxX,fam_wavelet);
    [yA,yH,yV,yD]=dwt2(auxY,fam_wavelet);

    %Se calcula la media aritmética de las matrices de aproximación.
    zA=(xA+yA)/2;

end

%NOTA: xH, xV, xD, yH, yV, yD son del mismo tamaño.
[f,c]=size(xH);

aH=zeros(f+(S-1),c+(S-1));
bH=aH;cH=aH;aV=aH;bV=aH;cV=aH;aD=aH;bD=aH;cD=aH;

aH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xH;
bH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yH;

aV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xV;
bV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yV;

aD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=xD;
bD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2))=yD;

%Se calcula el valor absoluto de los coeficentes de cada submatriz.
absaH=abs(aH);
absaV=abs(aV);
absaD=abs(aD);

absbH=abs(bH);
absbV=abs(bV);
absbD=abs(bD);

%Se selecciona una ventana de coeficientes de tamaño SxS centrada en
%cada coeficiente de las submatrices de detalles horizontales,
%verticales y diagonales.
for i=((S-1)/2)+1:1:f+((S-1)/2)
    for j=((S-1)/2)+1:1:c+((S-1)/2)

        %Matrices de detalles horizontales
        subaH=absaH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbH=absbH(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles verticales
```

```
        subaV=absaV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbV=absbV(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %Matrices de detalles diagonales
        subaD=absaD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));
        subbD=absbD(i-((S-1)/2):i+((S-1)/2),j-((S-1)/2):j+((S-1)/2));

        %A continuación se comparan los coeficientes de las ventanas
        %en posiciones correspondientes y se elige como píxel
        %central el de la ventana que gane por mayoría.
        %En caso de empate o igualdad se elige el píxel central del
        %TAC en la fusión PET-TAC, y la resonancia magnética en los
        %casos de fusión MR-PET y MR-TAC.

        %Matrices de detalles horizontales
        if (sum(sum(subaH>=subbH))>=((S^2)+1)/2)
            cH(i,j)=aH(i,j);
        else
            cH(i,j)=bH(i,j);
        end

        %Matrices de detalles verticales
        if (sum(sum(subaV>=subbV))>=((S^2)+1)/2)
            cV(i,j)=aV(i,j);
        else
            cV(i,j)=bV(i,j);
        end

        %Matrices de detalles diagonales
        if (sum(sum(subaD>=subbD))>=((S^2)+1)/2)
            cD(i,j)=aD(i,j);
        else
            cD(i,j)=bD(i,j);
        end

    end
end

zH=cH(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zV=cV(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));
zD=cD(((S-1)/2)+1:c+((S-1)/2),((S-1)/2)+1:c+((S-1)/2));

if nivel==1

    Z=idwt2(zA,zH,zV,zD,fam_wavelet);

else

    Z=idwt2(zA,zH,zV,zD,'haar');

end

end
```

----------Medidas----------

medidas.m

```
function
[calidadX,calidadY,CC_TAC,CC_PET,MSEx,MSEy,PSNR,SSIM,CR,MI]=medidas(X,Y,Z
)
%La función medidas calcula unas medidas numéricas que ayudan a evaluar
de
%forma cuantitativa los distintos algoritmos de fusión.
%X e Y se corresponden con las imágenes de partida y Z es la imagen
%fusionada.

[m,n]=size(X);
[p,q]=size(Y);
[r,s]=size(Z);

%Se comprueba si las matrices a fusionar no son cuadradas.
if m>n || p>q

    %Se eliminarán las columnas de la matriz de la imagen fusionada que
se
    %añadieron durante el proceso de fusión para que las matrices fueran
    %cuadradas.

    Z=Z(:,(((m-n)/2)+1):m-((m-n)/2));
    [r,s]=size(Z);

end

X=double(X);
Y=double(Y);

X=X-min(X(:));
Y=Y-min(Y(:));
Z=Z-min(Z(:));

X=X/max(X(:))*255;
Y=Y/max(Y(:))*255;
Z=Z/max(Z(:))*255;

%Se comprueba si ambas imágenes son del mismo tamaño.
if size(X)>size(Y)

    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,p,p*(m/p)),linspace(1,p,p*(m/p)));
    Y=interp2(Y,x,y);

elseif size(X)<size(Y)
```

```
    %Se aumenta el tamaño de la imagen más pequeña, interpolando por un
    %factor L.
    [x,y]=meshgrid(linspace(1,m,m*(p/m)),linspace(1,m,m*(p/m)));
    X=interp2(X,x,y);

end

aux=ones(r,s);

mediaX=mean(X(:));
mediaY=mean(Y(:));
mediaZ=mean(Z(:));

auxX=mediaX.*aux;
auxY=mediaY.*aux;
auxZ=mediaZ.*aux;

varX=mean2((X-auxX).^2);
varY=mean2((Y-auxY).^2);
varZ=mean2((Z-auxZ).^2);

covarX=mean2((X-auxX).*(Z-auxZ));
covarY=mean2((Y-auxY).*(Z-auxZ));

errorX=abs(Z-X);
errorY=abs(Z-Y);

error2X=errorX.^2;
error2Y=errorY.^2;

%Calidad espacial

calidadX=4*covarX*mediaX*mediaZ/((varX+varZ)*((mediaX^2)+(mediaZ^2)));

%Calidad espectral

calidadY=4*covarY*mediaY*mediaZ/((varY+varZ)*((mediaY^2)+(mediaZ^2)));

%Coeficiente de correlación
CC_TAC=corrcoef([X(:) Z(:)]);
CC_PET=corrcoef([Y(:) Z(:)]);

CC_TAC=CC_TAC(1,2);
CC_PET=CC_PET(1,2);

%MSE

MSEx=((sum(error2X(:)))/(r*s*255^2))*100;

MSEy=((sum(error2Y(:)))/(r*s*255^2))*100;

%PSNR
```

```
e1=errorX/255;
e2=errorY/255;

PSNR_TAC= 20*log10(1/(sqrt(mean(e1(:).^2))));
PSNR_PET= 20*log10(1/(sqrt(mean(e2(:).^2))));

PSNR=(PSNR_TAC+PSNR_PET)/2;

%Similitud SSIM

%TAC
p1Z = Parzen1D(Z);
p1X = Parzen1D(X);
p2 = Parzen2D(Z,X);
i = 0:511;
i2 = i.'*i;
m1 = sum(i.*p1Z.');
m2 = sum(i.*p1X.');
v1 = sum(i.^2.*p1Z.')-m1^2;
v2 = sum(i.^2.*p1X.')-m2^2;
v12 = sum(sum(i2.*p2))-m1*m2;
SSIMx = 4*m1*m2*v12/((m1^2+m2^2)*(v1+v2));

%PET
p1Z = Parzen1D(Z);
p1Y = Parzen1D(Y);
p2 = Parzen2D(Z,Y);
i = 0:511;
i2 = i.'*i;
m1 = sum(i.*p1Z.');
m2 = sum(i.*p1Y.');
v1 = sum(i.^2.*p1Z.')-m1^2;
v2 = sum(i.^2.*p1Y.')-m2^2;
v12 = sum(sum(i2.*p2))-m1*m2;
SSIMy = 4*m1*m2*v12/((m1^2+m2^2)*(v1+v2));

SSIM=(SSIMx+SSIMy)/2;

%CR

%TAC

p1Z = Parzen1D(Z);
p1X = Parzen1D(X);
p2 = Parzen2D(Z,X);
p21 = p2./(repmat(p1Z,1,512)+eps);
i = 0:511;
m2 = sum(i.*p1X.');
v2 = sum(i.^2.*p1X.')-m2^2;
m21 = sum(repmat(i,512,1).*p21,2);
v21 = sum(repmat(i.^2,512,1).*p21,2)-m21.^2;
CRx = 1-1/v2*sum(v21.*p1Z);

%PET
```

```
p1Z = Parzen1D(Z);
p1Y = Parzen1D(Y);
p2 = Parzen2D(Z,Y);
p21 = p2./(repmat(p1Z,1,512)+eps);
i = 0:511;
m2 = sum(i.*p1Y.');
v2 = sum(i.^2.*p1Y.')-m2^2;
m21 = sum(repmat(i,512,1).*p21,2);
v21 = sum(repmat(i.^2,512,1).*p21,2)-m21.^2;
CRy = 1-1/v2*sum(v21.*p1Z);

CR=((CRx+CRy)/2)*100;

%MI

%TAC

p1Z = Parzen1D(Z);
p1X = Parzen1D(X);
p2 = Parzen2D(Z,X);
MIx = sum(sum(p2.*log2(p2./(p1Z*p1X.'+eps)+eps)));

%PET

p1Z = Parzen1D(Z);
p1Y = Parzen1D(Y);
p2 = Parzen2D(Z,Y);
MIy = sum(sum(p2.*log2(p2./(p1Z*p1Y.'+eps)+eps)));

MI=(MIx+MIy)/2;

end
```