INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA
UNIDAD CULHUACAN
INGENIERÍA EN COMUNICACIONES Y ELECTRÓNICA

SEMINARIO DE TITULACIÓN
"PROCESAMIENTO DIGITAL DE SEÑALES"

(PROCESAMIENTO DIGITAL DE SEÑALES BIOMEDICAS CARDIOLOGIA E IMAGENOLOGIA)

T E S I N A

Que para obtener el grado de:

INGENIERO EN COMUNICACIONES Y ELECTRÓNICA.

Presentan:

(Jesús Bernal Núñez)

ASESORES:

M. en C. ORLANDO BELTRÁN NAVARRO.
M. en C. BRAULIO SANCHEZ ZAMORA

México, D. F. Noviembre de 2009.

INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA ELÉCTRICA
UNIDAD CULHUACAN

TESINA

Que para obtener el título de:	**INGENIERO EN COMUNICACIONES Y ELECTRONICA**
Por la opción de titulación:	**SEMINARIO DE TITULACIÓN "PROCESAMIENTO DIGITAL DE SEÑALES"**
Deberán desarrollar:	**(Jesús Bernal Núñez))**

INTRODUCCION

(Las imágenes en la medicina son fundamentales para diagnostico y planeación de acciones a tomar, por lo cual la importancia de su adquisición , representación visual, almacenamiento y transportación juegan un papel importantísimo en la medicina actual. El siguiente trabajo trata de manera general los tres casos planteados, dando a conocer las bases del tratamiento digital de imágenes, la adquisición de las imágenes biomédicas y las aplicaciones en software. **)**

CAPITULO I (TRATEMIENTO DIGITAL DE SEÑALES)
CAPITULO II (TRATAMIETNO DIGITAL DE IMAGEN)
CAPITULO III (IMÁGENES MEDICAS: ADQUISICION, INSTRUENTACION Y GESTION)
CAPITULO IV (IMÁGENES BIOMEDICAS APLICACIÓN CON SOFTWARE)

M. en C. Orlando Beltrán Navarro
Coordinador del seminario

M. en C. Braulio Sánchez Zamora
Asesor

Ing. Ignacio Monroy Ostria
Jefe de carrera de ICE

Intruduccion:

Capitulo 1;En el capitulo esta dedicado a la darnos a conocer la naturaleza de las señales las teorías la terminología y las características de generales de cada una de hallas, posterior nos introduciremos mas a fondo en le análisis de las imágenes, las principales formas de concebir y manipular una imagen de naturaleza digital, abordando también los tipos de modelos tomados de la naturaleza para organizar y poder manipular una imagen, asi como de la matemática especifica para algunos ejemplos de manipulación de imagen con sus distintas características complejidades diferencias, ventajas y desventajas.

Capituilo 2 y 3; En el capitulo dos tratamos el temas de filtros y tipos de filtros una baliose herraminta para poder lograr cualquier tipo de manipulación en una imagen. Ya que las imágenes por su manera de adquirirlas y representarlas visualmente tienen una connotación vectorial, esto es que cada pixel tiene un valor vectorial, posición en un plano, en un tiempo determinado y con un valor numérico conocido adquirido por tres factores, color, tinte y brillantes. Gracias a que se pueden tratar un pixel como un vector también los podemos manipular en conjunto o individual conocido como filtraje.

Capitulo 4; En el capitulo tres damos a conocer las distintas técnicas de conseguir una imagen medica y con esto veremos también la diferencia que hay entre estas y que cada técnica tiene ventajas y desventajas una con respecto a la otra y que algunas son aplicaciones de una misma técnica. Por su complejidad de adquisición y la naturaleza de la imagen ya sea analógica o digita nos muestra que el tratamiento de estas son mucho mayor complejas que las adquiridas por una cámara de video o fotográfica.

Capitulo 5; Daremos ejemplos de algunas manipulaciones de imágenes médicas adquiridas mediante una digitalización previa luego guardadas en un ordenador y manejadas mediante algunos programas de software.

1.1. TIPOS DE SEÑALES

Una primera clasificación importante en tratamiento de señal son los diferentes tipos de señales a tratar, puesto que las aplicaciones, herramientas, y procedimientos a aplicar dependerán en gran medida de cada clase en concreto.

Una posible clasificación es en función del número de variables independientes necesarias para especificar el valor de una determinada señal. Básicamente podemos encontrar:

Señales unidimensionales

Se pueden representar mediante una función que depende de una única variable. En este libro nos centraremos en la señal de voz, aunque son posibles otros tipos de señales, como la temperatura en un determinado observatorio, las cotizaciones en bolsa de una determinada acción, el registro de un sismógrafo, etcétera.

En el caso de la voz, un micrófono captará una tensión variable en función del tiempo. Por tanto, vendrá especificada mediante una función del tiempo f(t). Puesto que los tratamientos (o procesamientos) que describiremos a lo largo del texto se realizarán mediante un ordenador, los valores de la función (t) no estarán disponibles para todo instante de tiempo t, sino sólo cada cierto intervalo de tiempo (periodo de muestreo), y será necesario muestrear la señal.

Por otra parte, el espacio para almacenar datos en un ordenador es limitado y no es posible guardar un margen infinito (analógico) de valores de tensión posibles, tendrá que ser discretizado a un conjunto finito de valores, mediante un cuantificador escalar.

De ahora en adelante, cuando hablemos de señales de voz o imagen supondremos que las señales ya han sido muestreadas y cuantificadas.

No se tratará el tema de formatos de audio y gráficos, puesto que éstos únicamente condicionan la forma de almacenar los datos, y no afectan a las técnicas de tratamiento descritas en este libro. Sin embargo, es cierto que dependiendo del entorno de trabajo y la aplicación en concreto, resultarán más adecuados unos que otros.

Señales bidimensionales

Es el caso de imágenes estáticas. Dispondremos de una única imagen en blanco y negro formada por pixeis (picture elements) cuyo valor o nivel de gris vendrá determinado por sus coordenadas espaciales x e y.

Las dimensiones usuales suelen ser 5 12x5 12 pixels y 8 bits por pixel (256 niveles de gris posibles), para tener una calidad similar a la televisión convencional.

Señales tridimensionales

El paso de imagen estática a secuencias de imágenes supone añadir una variable más. El valor de un pixel dependerá de sus coordenadas espaciales y del instante de tiempo considerado, que sería equivalente a escoger un fotograma en concreto, en el caso de una película. Otra posibilidad es tener una imagen en color. Habitualmente se suele formar el color mediante la combinación de cantidades adecuadas de los colores rojo (R), verde (G) y azul (B). De esta forma es posible reproducir un gran número de colores.

De hecho, es equivalente a tener tres imágenes (una para cada color) y, por tanto, tres dimensiones. La figura 1 . 1 . 1 presenta un ejemplo de los tipos de señales que serán tratadas.

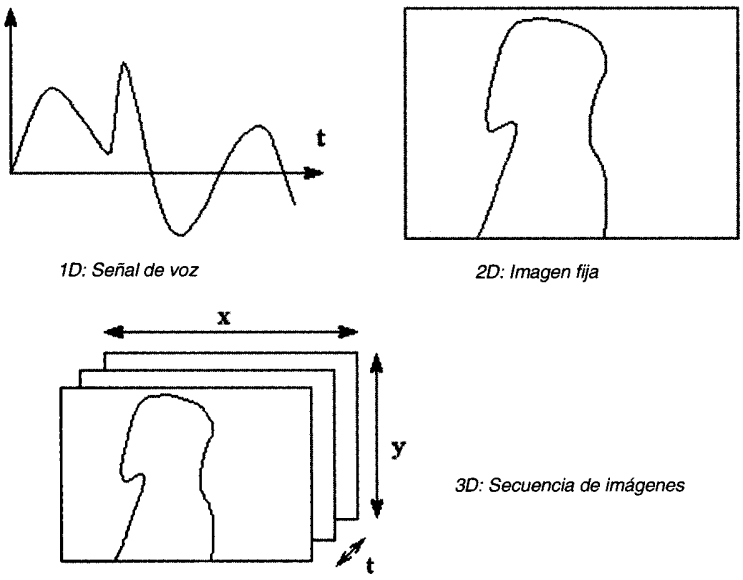


Figura 1.1.1. Ejemplos de tipos de señales

1 .2. CUANTIFICACIÓN VECTORLAL

Una de las herramientas que se utilizan en tratamiento de señal es la cuantificación vectorial. Por ello en este apartado se describe en qué consiste y se presentan algunas aplicaciones típicas para ayudar a comprender el concepto de cuantificación vectorial.

1.2.1. Concepto

Es la generalización de la cuantificación escalar para dos o más componentes (vectores). A continuación se resumen las principales características de la cuantificación escalar.

Mediante la cuantificación escalar se asigna un valor de salida perteneciente a un conjunto discreto de valores, a un cierto margen de valores posibles de entrada. La figura 1 .2. 1 muestra un cuantificador escalar de 3 bits (8 valores de salida posibles).

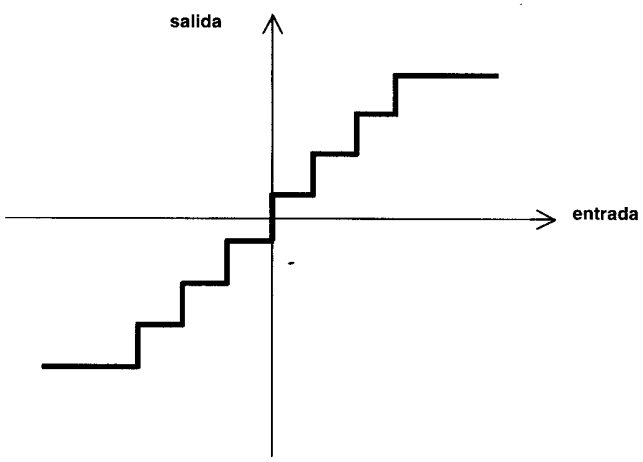


Figura 1.2.1. Cuantificador escalar de 3 bits (8 niveles de salida).

Al aplicar cuantificación a una señal se obtiene compresión, puesto que el conjunto de valores posibles de salida es menor al de entrada. En contrapartida,
introduce un error ya que es un proceso irreversible (a partir de la señal cuantificada no se puede recuperar la señal original).

Existe un compromiso entre la compresión y el error introducido por la cuantificación. Cuanto mayor sea el conjunto posible de valores de salida, más parecidas serán las señales previa y posterior a la cuantificación, pero la compresión será menor, puesto que el número de bits necesarios para representar todos los valores de salida posibles será mayor.

Ejemplo 1.2.1

Si se aplica a una señal de entrada discreta representada con 8 bits/muestra (256 valores posibles) un cuantificador escalar de 3 bits (8 valores posibles), se obtiene una compresión 8 a 3 (2,7:1).

En el diseño de un cuantificador escalar hay que escoger el número de bits del cuantificador o, equivalentemente, el número de valores posibles de salida (2^N bits = número de valores posibles) y, a continuación, buscar cuáles son los valores óptimos de salida para obtener el mínimo error posible y el correspondiente margen de valores de entrada.

La cuantificación vectorial es una generalización de la cuantificación escalar consistente en tratar conjuntamente grupos de N muestras, siendo N la dimensión de los vectores.

Por tanto, mientras que la cuantificación escalar trabaja sobre valores de entrada escalares (o vectores de una componente en el espacio $\mathcal{R}$), la cuantificación vectorial trabaja sobre vectores de entrada N-dimensionales en el espacio $\mathcal{R}$^N Análogamente, de todo el conjunto de vectores N-dimensionales posibles de entrada sólo permite K vectores posibles a la salida.

Mientras que en la cuantificación escalar la relación salida-entrada se representa mediante una función de tipo escalera o un array que contiene los márgenes de entrada y su correspondiente valor de salida, en la cuantificación vectorial se utiliza un libro de códigos, conocido en la terminología inglesa como codebook.

El codebook es una tabla que contiene los vectores posibles a la salida del cuantificador. Cada vector del codebook se llama codevector y su correspondiente índice en la tabla codeword.
Al igual que en la cuantificación escalar, para obtener una representación eficiente, se almacena el índice de la tabla para cada una de las muestras de la señal de entrada y la tabla que permite obtener el correspondiente valor de salida.

En el diseño de un cuantificador vectorial hay que escoger el número de bits del codebook (o equivalentemente el número de codewords = 2^N bits codebook), la dimensión de los vectores de entrada, y los valores de los codevectores para que la cuantificación implique el mínimo error posible de la señal cuantificada respecto a la original.

Ejemplo 1.2.2: Cuantificador vectorial de vectores en $\mathcal{R}$^2

En el caso de tratar con vectores de dimensión dos, resultará posible dibujar los vectores en el plano, en un espacio bidimensional. Independientemente de la dimensión de la señal de entrada hay que coger pares de muestras y obtener el valor de salida. Si la señal es unidimensional y el codebook es de cuatro vectores hay que coger las muestras de dos en dos y escoger el codevector más parecido (el que proporciona una distancia menor respecto al de entrada). Si re- presentamos las dos primeras muestras de x(t) en el espacio $\mathcal{R}$^2 y buscamos el codevector más cercano de los cuatro posibles (representados con cruces) obtenemos como salida el índice del segundo vector (fig. 1.2.2).

Análogamente a la cuantificación escalar, es posible definir particiones en ℜ^2 que indiquen las zonas de entrada a las que corresponde cada uno de los vectores de salida.

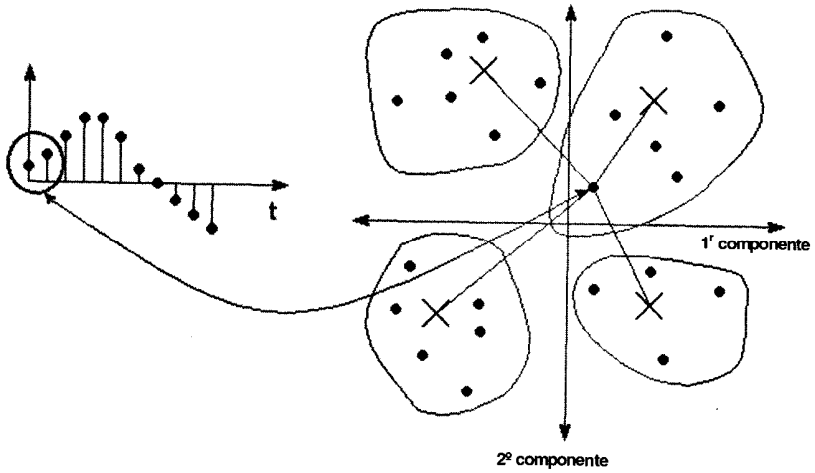


Figura 1.2.2. Señal de voz, vectores en R^2

Ejemplo 1.2.3

Calcular la compresión conseguida al realizar cuantificación vectorial de imagen tomando bloques de 4x4 pixels (N= 16) siendo cada pixel 1 byte (256 niveles de gris) y el codebook de 7 bits.
La compresión es la relación entre el número de bits antes y después de cuantificar. Por tanto,

$$C = \frac{4 \cdot 4 \cdot 8}{7} = 18{,}3$$

Ídem si la cuantificación vectorial se realiza de secuencias de imágenes con bloques cúbicos de 4x4x2 (vectores de dimensión N= 32).

$$C = \frac{4 \cdot 4 \cdot 2 \cdot 8}{7} = 36{,}6$$

1 .2.2. Proceso de cuantificación

Cuantificar un vector de entrada (x̄) consiste en encontrar su "vecino más cercano" (ȳj) dentro del codebook. Para ello hay que calcular su distancia a todos los vectores del codebook y escoger el que proporcione un valor menor.

Es necesario disponer de una medida de distancia. Por ejemplo, la distancia cuclídea entre vectores:

$$d(\vec{x}, \vec{y}_j) = \sum_{i=1}^{N} (x_j - y_{ji})^2 = D_j \qquad \text{con } \vec{x} = [x_1, x_2, ..., x_N] \text{ e } \vec{y}_j = [y_{j1}, y_{j2}, ..., y_{jN}]$$

El algoritmo es el siguiente:

1. Inicializar la distancia mínima (d) igual a la distancia inicial (D_1), y los contadores j e i (índice del codevector más cercano):

$$d = D_1 ; \qquad j = 1 ; \qquad i = 1$$

2.

$$D_{j+1} = d(\vec{x}, \vec{y}_{j+1})$$

3.

$$\text{Si } D_{j+1} < d \Rightarrow \begin{cases} d = D_{j+1} \\ i = j + 1 \end{cases}$$

4.

$$\text{Si } \quad j < K \Rightarrow \begin{cases} j = j + 1 \\ \text{volver al punto 2} \end{cases}$$

$$\text{Si } \quad j = K, \text{ el resultado es el índice } i \text{ y la distancia mínima } d.$$

Existen algoritmos que permiten reducir las operaciones ordenando el codebook de forma que no sea necesario realizar la comparación con todos los codevectores de la tabla. Sin embargo, requieren más memoria para almacenar el codebook.

La obtención del codebook suele realizarse en dos pasos:

- Obtención de un codebook inicial.
- Mejora del codebook inicial mediante el algoritmo de Lloyd.

A continuación se describen estos dos pasos.

1 .2.3. Obtención del codebook inicial

Es importante obtener un bueñ codebook inicial, para conseguir un resultado final (después de aplicar el algoritmo de mejora de Lloyd) satisfactorio. El ejemplo de la cuantificación vectorial de imágenes codificadas con una variante del algoritmo BTC (Block Truncation Coding) ilustra este fenómeno.

La creación del codebook inicial y su mejora se realiza a partir de una secuencia de entrenamiento, formada por un conjunto de vectores pertenecientes al espacio N—dimensional.

Por tanto, el codebook estaría especialmente adaptado a la secuencia de en- trenamiento, puesto que se obtendrá a partir de él. Para conseguir una cuantificación eficiente de vectores no contenidos en la secuencia de entrenamiento, dicha secuencia debe ser lo más representativa posible de la distribución de la señal de entrada. En caso contrario, el codebook se especializará en unos vectores determinados a costa de penalizar otros.

A continuación se describen algunos de los algoritmos más utilizados para obtener un codebook inicial a partir de una secuencia de entrenamiento.

1.2.3. 1. Método aleatorio (random)

Si el codebook debe contener k vectores, tomamos los primeros k vectores
la secuencia de entrenamiento. Si los datos están muy correlacionados (vectores próximos son semejantes) se obtiene un mejor resultado tomándolos es- paciados (por ejemplo, uno de cada cinco vectores consecutivos).

En cualquiera de ambos casos no existe la seguridad de que los vectores del codebook sean distintos entre sí y cubran todo el espacio N-dimensional y en especial las zonas donde existe más probabilidad de encontrar vectores.

Si la distribución fuera uniforme, entonces el cuantificador óptimo es el es- calar. No se obtiene ninguna ventaja usando cuantificación vectorial. Esta es cuando existen zonas del espacio en las que la probabilidad de encontrar un vector es mayor que en otras.

1.2. 3.2. Método de poda (pruning)

Es una sofisticación del método anterior consistente en asegurarse de que los vectores del codebook no cubran una misma zona del espacio dejando de lado otras zonas. El algoritmo es el siguiente:

1. Se introduce el primer vector de la secuencia de entrenamiento en el co- debook.
2. Se calcula la distorsión entre el siguiente vector de la secuencia de entre- namiento y el primer codevector.

- Si es menor a un umbral se prueba con el siguiente vector de la se- cuencia de entrenamiento.
- Si no, se añade al codebook como un nuevo vector.
- Con cada nuevo vector de la secuencia de entrenamiento se busca el vecino más cercano en el codebook. Si la distorsión entre ambos es mayor al umbral, se añade el vector al codebook. Se repite el procedimiento hasta llenar el codebook.

3. Si se llega al final de la secuencia de entrenamiento y todavía no se ha llenado todo el codebook, hay que reducir el umbral y repetir el proceso.

El unibral suele escogerse proporcional a:

$$\text{umbral} \quad \alpha \quad K^{\frac{-2}{N}} = \frac{1}{K^{\frac{2}{N}}}$$

donde N es la dimensión de los vectores y K el número de vectores del codebook.
Obsérvese que si el número de vectores del codebook (K) aumenta, el umbral disminuye puesto que debemos coger más vectores para poder llenar el co- debook, y si la dimensión de los vectores (N) aumenta, el umbral también aumenta puesto que las distancias serán mayores por naturaleza.

1.2. 3. 3. Método de splitting

A diferencia de los métodos anteriores, únicamente permite crear codebooks de tamaños que sean potencia exacta de dos. Ello es debido a que se trata de un proceso iterativo en el que en cada iteración se dobla el tamaño del codebook obtenido en la iteración anterior.

El algoritmo consiste en los siguientes pasos:

1. Se crea un codebook con un único vector $\bar{c}_0$ que es el centroide (codevector) de toda la secuencia de entrenamiento (suma de todos los vectores de la secuencia de entrenamiento dividido por el número de vectores que contiene la secuencia (L).

$$\vec{c}_o = \frac{1}{L}\sum_{i=1}^{L}\vec{v}_i$$

Donde la secuencia de entrenamiento es el conjunto de L vectores:

$$SE = \{\vec{v}_1, \vec{v}_2, ..., \vec{v}_L\}$$

2. Se divide el centroide en dos codevectores, obteniendo un codebook dos vectores:

$$codebook = \{\vec{c}_o, \vec{c}_o + \vec{\varepsilon}\}$$

donde *E* es una pequeña perturbación. Por ejemplo, puede hacerse la componente *i*-ésima de e proporcional a la varianza de la componente *i*-ésima del conjunto de vectores de la secuencia de entrenamiento:

$$\begin{aligned}
\vec{v}_1 &= \{v_{11}, v_{12}, ..., v_{1N}\} \\
\vec{v}_2 &= \{v_{21}, v_{22}, ..., v_{2N}\} \qquad \sigma_1 = varianza\,\{v_{11}, v_{21}, ..., v_{L1}\} \\
&\vdots \qquad\qquad\qquad\qquad \sigma_2 = varianza\,\{v_{12}, v_{22}, ..., v_{L2}\} \\
\vec{v}_L &= \{v_{L1}, v_{L2}, ..., v_{LN}\} \qquad \vdots \\
\vec{\varepsilon} &= \{\sigma_1, \sigma_2, ..., \sigma_N\} \qquad \sigma_N = varianza\,\{v_{1N}, v_{2N}, ..., v_{LN}\}
\end{aligned}$$

con

3. Se aplica el algoritmo de mejora de codebooks.
4. Se vuelve a dividir en dos y mejorar cada uno de los codevectores del codebook hasta llegar al tamaño deseado, que será potencia de dos.

La figura 1 .2.3 muestra un ejemplo en el caso de trabajar con vectores de dimensión dos.

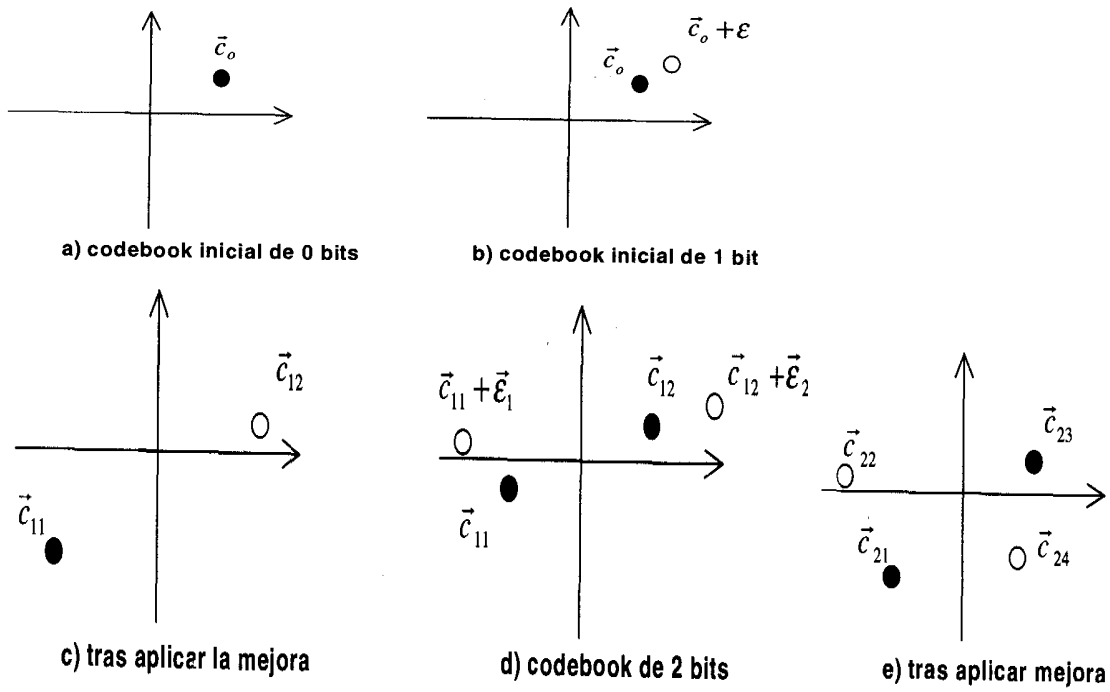


Figura 1.2.3. Obtención del codebook inicial (método de splitting).

El cálculo de la perturbación εi usada en un determinado vector *Ci* se puede realizar a partir del cálculo de las varianzas de las componentes de los vectores de secuencia de entrenamiento asignados a dicho vector *Ci*. De esta forma, cuanto mayor sea la dispersión (o varianza) de la nube de vectores de entrenamiento asignada al codeword *Ci* más separados estarán los vectores resultantes *Ci y Ci*+εi.

La utilización del algoritmo de splitting para la creación del codebook inicial presenta una ventaja respecto a los otros métodos: Si se almacenan en me- mona los diferentes codebooks previos a la división, será posible realizar una búsqueda de codewords en una estructura de árbol binario, la cual requiere log2 k comparaciones de vectores, frente a las k comparaciones requeridas mediante la búsqueda exhaustiva.

1.2.4. Mejora del codebook

1.2.4. 1. Iteración de Lloyd

Dado un codebook inicial (obtenido por cualquier método), puede mejorarse mediante el siguiente proceso:

1. Dividir la secuencia de entrenamiento en sectores (clusters) usando la condición de vecino más cercano. De esta forma, cada uno de los co- dewords tendrá asignado un subconjunto de vectores pertenecientes a la secuencia de entrenamiento.
2. Una vez obtenidas las particiones podemos preguntarnos para cada uno de los codevectores: ¿Es el codevector que mejor representa a los vectores asignados? El mejor codevector será el promedio de todos los vectores que tiene asignados. Por tanto, podemos calcularlo y sustituir el antiguo codevector. Si en el primer paso hemos obtenido una región vacía, habrá que tratarla como caso aparte. Por ejemplo, podremos dividir en dos çl centroide que más contribuya a la distorsión total del codebook.

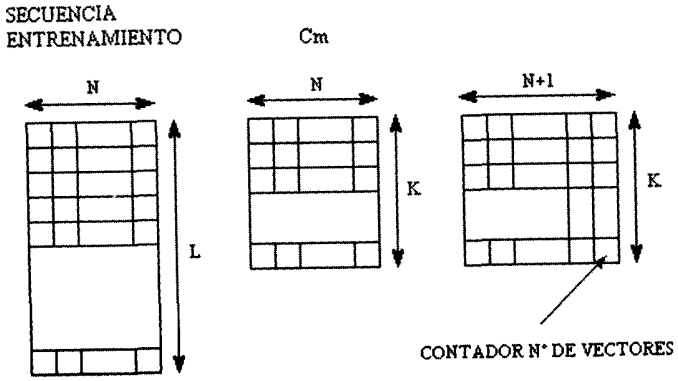


Figura 1.2.4. Implementación práctica de la iteración de Lloyd.

La implementación práctica será la representada en la figura 1.2.4.
Necesitamos:

- La secuencia de entrenamiento formada por L vectores de dimensión N.
- El codebook de K vectores de dimensión N.
- Una tabla intermedia formada por K filas de N+ 1 posiciones.

Procedimiento

Para cada uno de los L vectores de la secuencia de entrenamiento buscamos su vecino más cercano en el codebook y en la correspondiente posición de la tabla intermedia, inicializada previamente a cero en todas sus casillas, sumamos el vector de la secuencia de entrenamiento en las correspondientes N primeras casillas. La posición N+ 1 es un contador con el número de vectores asociados a dicha partición. Por tanto, su contenido debe incrementarse en una unidad.

Se procede de esta forma con todos los vectores de la secuencia de entrenamiento, de manera que al final en la tabla intermedia tenemos el valor acumulado de todos los vectores asociados a una determinada partición y el número de vectores asignados. Por tanto, el paso final consiste en actualizar el codebook sustituyendo cada una de las K filas por las correspondientes N primeras casillas de la tabla intermedia, divididas por la casilla N+ 1.

1.2.4.2. Algoritmo de Lloyd generalizado

Tras recalcular el codebook es posible que las particiones de la secuencia de entrenamiento hayan cambiado, con lo cual es posible recalcularlas y encontrar nuevos codevectores que reduzcan más la distorsión, y nuevamente varíen particiones. Por tanto, es necesario realizar el proceso de forma iterativa, según el algoritmo de la figura 1.2.5.

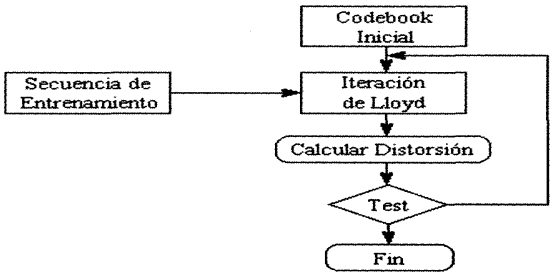


Figura 1.2.5. Diagrama de flujo del algoritmo de Lloyd.

Normalmente, al cabo de unas cuantas iteraciones la situación se estabiliza, la aplicación de la iteración de Lloyd no produce una mejora significativa. tanto, se repite todo el proceso hasta que la reducción de distorsión entre iteraciones consecutivas sea nula o poco significativa.

1.2.5. Variantes de la cuantificación vectorial

El diseño de un codebook es tanto más complicado cuanto mayor sea su tamaño. Por ello, se han propuesto variantes para reducir la complejidad.

1.2.5. 1. Cuantificación vectorial clasificada

Se divide la señal de entrada en varias categorías y se ajusta un codebook a cada una de ellas. Por ejemplo, en el caso de voz, se pueden establecer las siguientes clases: vectores pertenecientes a señal sonora, sorda y transición; y en caso de imagen: vectores pertenecientes a zonas homogéneas, transiciones suaves y transiciones bruscas.

Será necesario disponer de un clasificador que separe los vectores de entrada en las clases fijadas, y el resultado de la cuantificación serán los índices del codevector y del codebook seleccionados.

La figura 1 .2.6 muestra el esquema a utilizar.

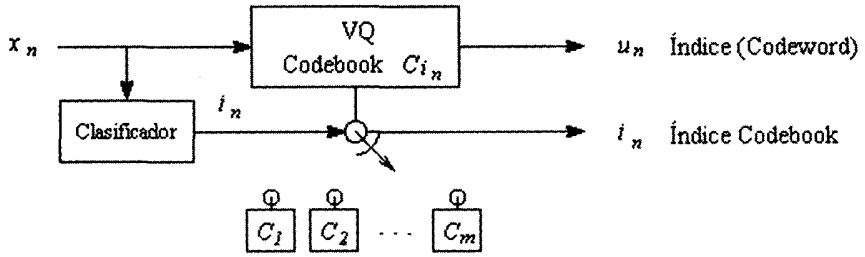


Figura 1.2.6. Cuantificación vectorial clasificada.

Mediante esta alternativa se reduce el problema de diseñar un gran codebook al diseño de diversos codebooks más pequeños especializados en vectores entrada con unas características similares.

1.2.5.2. Cuantificación vectorial de vectores transformados

Reduciendo la dimensionalidad de los vectores de entrada se simplifica el diseño del codebook. Para ello se realiza una transformación de los vectores de entrada al dominio de la frecuencia. En un vector transformado la energía está compactada en las bajas frecuencias, y los coeficientes de alta frecuencia suelen contener valores pequeños que pueden ser despreciados. Por tanto, el codebook se diseñará sobre vectores transformados de dimensión M<N y para recuperar el vector de entrada se añadirán ceros en los coeficientes de alta frecuencia hasta completar la dimensionalidad del vector de entrada (N—M ceros). Poste- riormente se realizará la transformación inversa.

La figura 1 .2.7 muestra el esquema a utilizar.

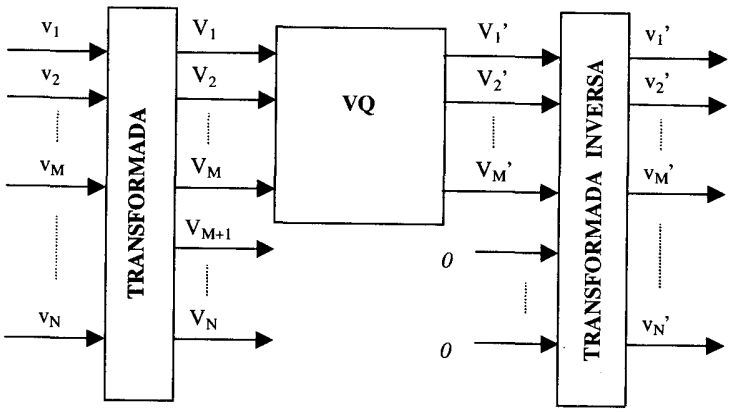


Figura 1.2.7. Cuantificación vectorial de vectores transformados.

1.2.5. 3. Cuantificación vectorial de ganancia y forma

Cuanto más acotada esté la zona del espacio en la que están los vectores a cuantificar, más fácil será obtener un buen codebook. De hecho, pueden existir vectores que únicamente se diferencien por un factor de ganancia. Por ejemplo, el vector { 1 , 1 , 1 } y el vector { 10, 10, 1O} presentan una diferencia elevada entre ambos, pese apuntar a una misma dirección del espacio. Se podrían representar con un mismo codeword si se realiza primero una normalización de su módulo y se codifica por separado la forma del vector, o dirección a la que apunta, y su módulo. La ganancia (g) es un número y, por tanto, puede codificarse mediante un cuantificador escalar.

$$g = |\vec{x}| = \sqrt{\sum_{i=1}^{N} x_i^2}$$

Para normalizar el vector por su ganancia el vector de entrada por la ganancia:

$$\vec{s} = \frac{\vec{x}}{g}$$

Realizando esta operación sobre todos los vectores, la cuantificación vectorial en un espacio N-dimensional se reduce a la cuantificación vectorial de vectores de dimensión N contenidos en una hiperesfera de radio unidad.

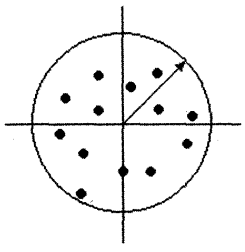


Figura 1.2.8. Hiperesfera de radio unidad.

1.2.6. Aplicaciones

En esta sección se describen diversas aplicaciones reales en las que se utiliza cuantificación vectorial.

1.2.6. 1. Codificación Block Truncation Coding (BTC) de imágenes con cuantificación vectorial

Consiste en dividir las imágenes en bloques más pequeños y realizar la codificación de cada uno de ellos por separado.

La compresión BTC se basa en la conservación de datos estadísticos del bloque en la imagen reconstruida, en concreto la media y la desviación típica de sus niveles de gris. La codificación de cada bloque se lleva a cabo mediante un mapa de bits y 2 niveles de gris. El mapa de bits debe tener tantos bits como pi- xels por bloque utilicemos. En nuestro caso serán 16 bits, puesto que utilizare- fl10S bloques de 4x4 pixels. La misión del mapa de bits es conservar la información de qué pixels del bloque tienen un nivel de gris alto y cuáles lo tienen bajo. Para ello primero calcularemos el nivel de gris medio del bloque y luego compararemos el nivel de cada pixel con el nivel medio. Si el nivel es superior al medio colocaremos un "1" en el bit del mapa correspondiente al pixel, y si es inferior codificaremos un "O". Ahora sólo falta escoger los dos niveles de gris representativos del bloque, de forma que se conserven la media y la desviación. I)urante el proceso de descompresión se reconstruirán con el nivel de gris alto (a) todos los pixeis con un "1" en el mapa de bits, y con el nivel de gris bajo (h) aquellos codificados con un "O". Las fórmulas de cálculo de los niveles son las siguientes:

$$X = \frac{1}{n}\sum_{i=1}^{n} x_i \quad ; \quad X_2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2$$

$$b = X - \sigma\sqrt{\frac{q}{n-q}} \quad ; \quad a = X + \sigma\sqrt{\frac{n-q}{q}} \quad ; \quad \sigma = \sqrt{X^2 - X_2^2}$$

donde x es el nivel de gris del pixel i, X es el nivel de gris medio del bloque, X2 es el momento de segundo orden, n es el número de pixels por bloque y q el número de pixels codificados con un "1" en el mapa de bits.

Utilizando esta técnica se consigue un factor de compresión de 4: 1 si trabajamos con imágenes de 256 niveles de gris (8 bits/pixel).

La codificación BTC básica puede interpretarse como un caso particular de la cuantificación vectorial en la que se obtiene un codebook de 2 elementos para cada uno de los bloques que integran la imagen.

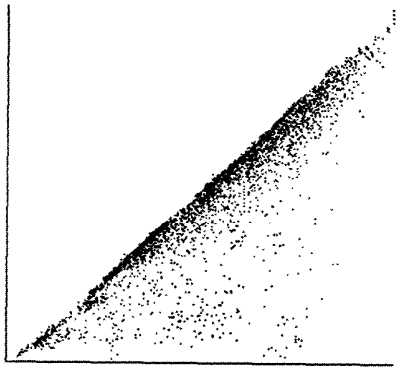


Figura 1.2.9. Secuencia de entrenamiento.

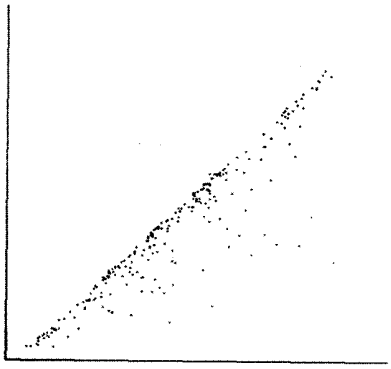


Figura 1.2.10. Codebook inicial con método aleatorio.

En las imágenes utilizadas cada pixel viene representado por su nivel de luminancia, siendo éste un valor positivo comprendido entre O (negro) y 255 (blanco). Los pares de niveles representativos de cada bloque (a y b) se pueden ordenar (mayor, menor) y representar en el espacio R2, quedando ubicados en medio cuadrante. Por otra parte, la mayoría de subimágenes pertenecen a partes homogéneas de la imagen, en las que hay poca variación y sus respectivos pares quedan situados principalmente alrededor de la recta y =x (a b). Dado que los pares (a,b) están concentrados en una zona determinada del espacio (figura 1.2.9), puede obtenerse una representación más eficiente ralizando la cuantificación vectorial de los mismos. De esta forma, se reduce el número de bits necesarios para codificar los niveles a y b, y, por tanto, aumenta la relación de compresión conseguida. El tamaño escogido del codebook es de 8 bits. Por consiguiente, la relación de compresión es 5,3: 1 . Aunque existen diversas variantes para incrementar el factor de compresión, el objetivo de este ejemplo es únicamente ilustrar una aplicación de la cuantificación vectorial. Por tanto, se limitará a visualizar los codebooks obtenidos y sus correspondientes distorciones.

Los pasos realizados en la generación del codebook son los siguientes:

1. Obtención de una secuencia de entrenamiento: Se han utilizado 4750 vectores provenientes de la codificación BTC de diversas imágenes. figura 1 .2.9 representa los vectores de la secuencia de entrenamiento el plano R2.
2. Obtención de un codebook inicial: Se han estudiado dos métodos: (aleatorio) y splitting. Para evaluar cuál de los dos sistemas es más efectivo, se ha cuantificado la secuencia de entrenamiento con codebooks iníciales generados a partir de los dos métodos. Los valores obtenidos de MSE (error cuadrático medio) han sido los siguientes:

Aleatorio: 4,5895
splitting: 2,3966

La figura 1.2.10 muestra el codebook inicial obtenido con el método aleatorio, y la figura 1.2.11 con el splitting.

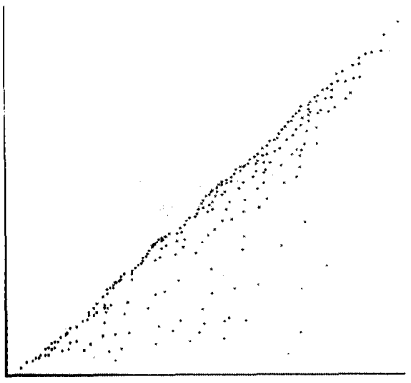


Figura 1.2.11. Codebook inicial splitting.

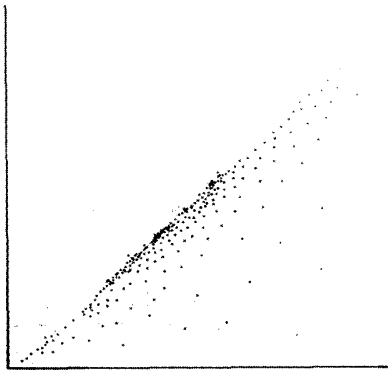


Figura 1.2.12. Codebook inicial método aleatorio mejorado con la iteración de Lloyd.

3. Mejora del codebook inicial: Se ha aplicado la iteración de Lloyd, hasta que el error no ha disminuido sustancialmente respecto a la iteración antenor. Los resultados obtenidos después de aplicar la iteración de Lloyd han sido:

aleatorio: 2,692] (se realizaron 1 2 iteraciones)
splitting: 2,0204 (se realizaron 9 iteraciones)

Es interesante observar que el método splitting obtiene un resultado meje el aleatorio con iteración de Lloyd. Esta observación también puede com probarse evaluando la distribución de puntos mostrada en la figura 1 .2. 1 2 (método aleatorio) y en la figura 1 .2. 1 3 (splitting). En el segundo caso los punto están mejor distribuidos.

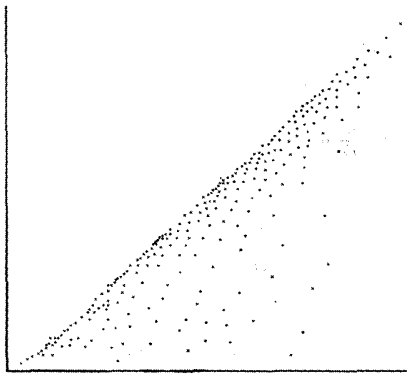


Figura 1.2.13. Codebook inicial método splitting mejorado con la iteración de Lloyd.

1.2.6.2. Cuantificación vectorial de coeficientes LPC (LPC-VQ)

En tratamiento de voz suele utilizarse un modelo de producción del habla consistente en modelar la vibración de las cuerdas vocales y el aire impulsad desde los pulmones mediante una señal de excitación y el tracto vocal como filtro. La parte del filtro típicamente se modela con 10 o 12 coeficientes que s actualizan cada 20 ms y se denominan coeficientes LPC. El conjunto de todo los valores posibles del filtro es muy elevado y por ello puede recurrirse a I cuantificación vectorial para representarlos de forma más eficiente.

La realización de un único codebook que permita cuantificar los coeficien1e LPC de forma transparente (sin distorsión apreciable) es un problema inabordable de forma directa, debido a la gran complejidad de cálculo y memoria necesaria. Por ello, se puede utilizar una de las variantes de la cuantificación vectorial. En concreto, existen estudios que demuestran que con 24 bits y e método split-VQ es posible realizar la cuantificación transparente.

El método split-VQ aplicado a los coeficientes LPC consiste en dividir Io diez coeficientes de cada vector ($\bar{c}$) LPC en dos subvectores ($\bar{c}1$, $\bar{c}2$) de siguiente forma: $\bar{c}$= [$\bar{c}1, \bar{c}2, \bar{c}3,..., \bar{c}10$] se divide en $\bar{c}1$ =[c1, c2, c3, c4] y $\bar{c}2$ =[c5, c6,... , c10] .

Una vez hecha la (livisión, se calculan dos codehook: 1() para los cuatro (primeros coeficientes y otro para los seis siguientes, dedicando igual número de bits a ambos codebooks (si el número de bits es impar se asigna un bit más al primer codebook).

De esta forma, se reduce la complejidad computacional, puesto que se reduce:

- La dimensión de los vectores.
- La dimensión del codebook.

El aspecto más importante es que resulta más sencillo implementar dos codebooks de 212 vectores (4096) que uno de 224 vectores (16.777.216).

La figura 1.2.14 muestra la ganancia de predicción obtenida con un único codebook para tamaños de K= 1 a K=212 codevectores y el resultado con dos codebooks hasta tamaños de 212 cada codebook.

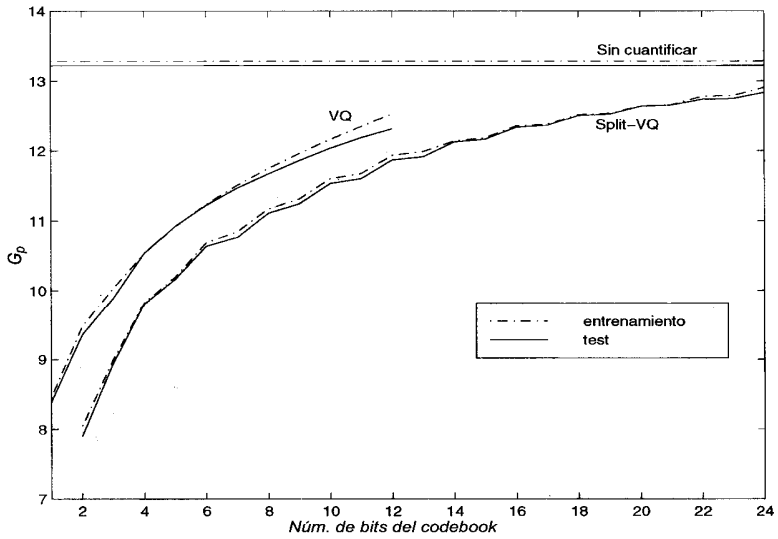


Figura 1.2.14. Ganancias de predicción en función del número de bits del cuantificador.

Se puede observar que el método spiit-VQ es una solución subóptima, puesto que se consiguen ganancias inferiores a igualdad de número total de bits. Sin embargo, para 24 bits se obtienen ganancias superiores a las conseguibles con cuantificación directa y una complejidad abordable (K ≤ 2ˆ12).

La gráfica muestra los resultados obtenidos con dos secuencias, una utilizada en el proceso de creación de los codebooks y otra no utilizada en dicho proceso, así como la ganancia obtenida sin cuantificación. La coincidencia de las ganancias de las secuencias de entrenamiento y de test indica una buena capacidad de generalización del codebook, para cuantificar vectores no utilizados en el entrenamiento. La coincidencia de las ganancias sin cuantificar y cuantificando revela la obtención de cuantificación transparente (ausencia de pérdida de prestaciones al cuantificar).

1 .3. REDES NEURONALES ARTIFICIALES

1.3.1. La neurona biológica

Una neurona biológica es una célula especializada en procesar información. Está compuesta por el cuerpo de la célula (soma) y dos tipos de ramificaciones: el axón y las dendritas (figura 1.3.1). La neurona recibe señales (impulsos) de otras neuronas a través de sus dendritas y transmite señales generadas por el cuerpo de la célula a través del axón.

El córtex cerebral humano contiene alrededor de 1011 neuronas (aproximadamente el mismo número de estrellas que hay en la Vía Láctea). Cada neurona está conectada de forma masiva a un número variable de entre 1000 y 10.000 neuronas. Por tanto, el cerebro contiene de 1014 a 1015 interconexiones

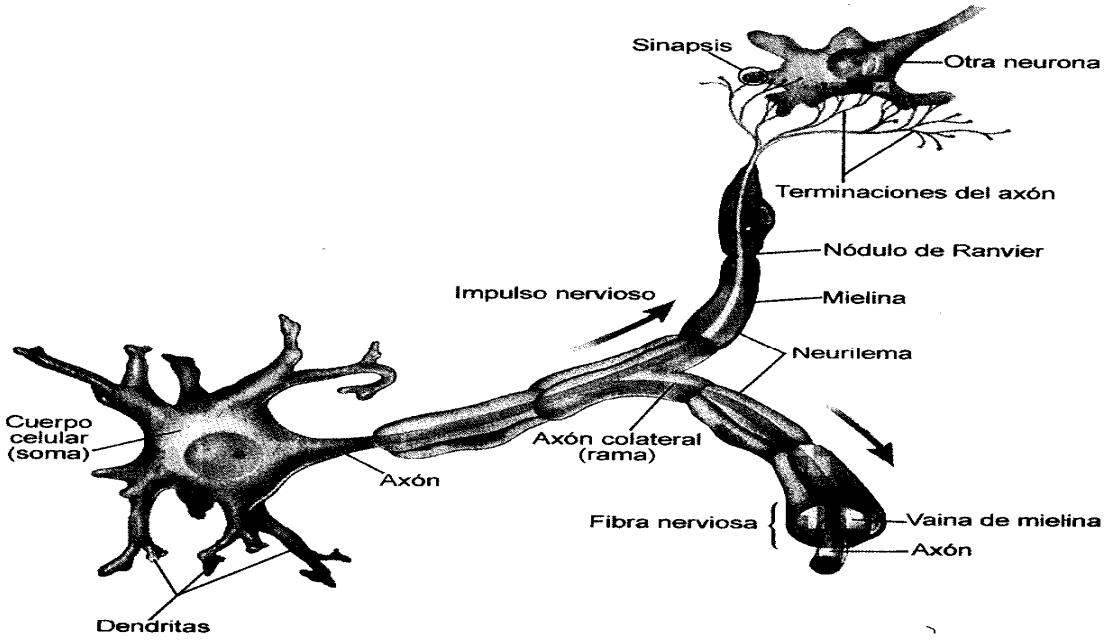


Figura 1.3.1. Neurona biológica.

Las neuronas se comunican entre sí mediante trenes de pulsos de corta duración (del orden de milisegundos). El mensaje, está modulado en la frecuencia transmisión de los pulsos. Esta frecuencia varía sobre los 100 Hz. Por tanto, más de un millón de veces inferior a la velocidad de conmutación de los circuitos electrónicos típicos de nuestros días.

Sin embargo, el ser humano es capaz de realizar tareas complejas en un tiempo inferior y con un porcentaje de aciertos superior al conseguido actualmente mediante ordenadores. Por ejemplo, tareas de reconocimiento del habla, locutor a través de su voz, cara, etc. Ello es debido a la paralelización masiva realiza el cerebro, que está extremadamente lejos de poder ser implementada en un circuito electrónico.

1.3.2. Modelo de neurona artificial

Las redes neuronales artificiales pretenden emular la red neuronal biológica se obtienen interconectando neuronas. Para ello se supone un modelo matemático simplificado de la neurona biológica.

Este modelo es una generalización del propuesto por McCulloch y Pitts en 1943:

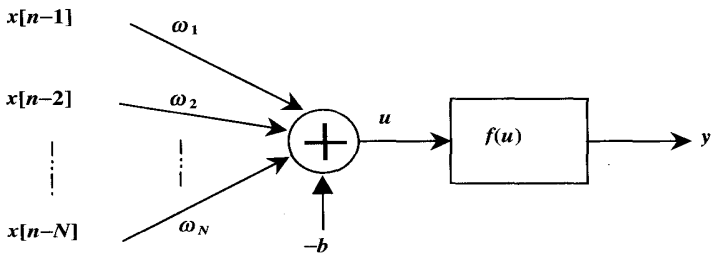


Figura 1.3.2. Modelo de neurona artificial.

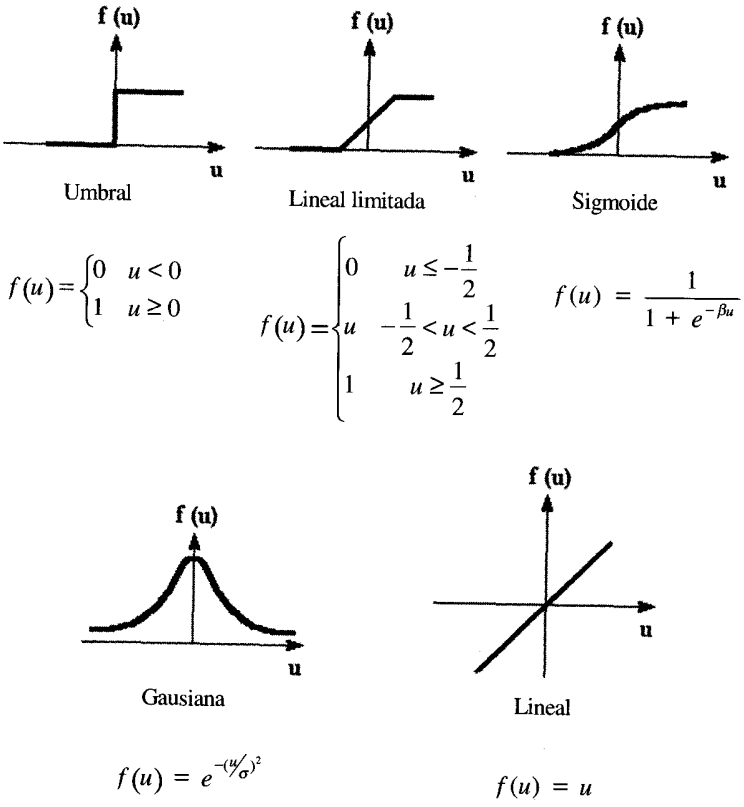
$$y = f\left[\left(\sum_{i=1}^{N} \omega_i \cdot x(n-i)\right) - b\right]$$

La neurona recibe del exterior un umbral b y N entradas a las que asocia conjunto de pesos ú (i=1,..., N). Aplicando el producto de los pesos por entradas respectivas más el umbral (o desplazamiento) a una función de se obtiene la salida y. La figura 1.3.3 muestra las funciones de más comunes.
ū(x̄,ω̄) recibe el nombre de función base, donde los vectores *x̄,ω̄*, son:

$$\vec{x} = \{x[n-1],..., [n-N]\}$$
$$\vec{\omega} = \{\omega_1,..., \omega_N\}$$

Cuando la combinación de las entradas es de tipo lineal, u(., c) es una función de base lineal, pero son posibles otras combinaciones, entre las que destacan las funciones de base radial, consistentes en realizar una operación no lineal sobre las entradas, con simetría radial. Por ejemplo:

$$u(\vec{x},\vec{\omega}) = \sum_{i=1}^{N} [\omega_i - x(n-i)]^2$$



$$f(u) = \begin{cases} 0 & u < 0 \\ 1 & u \geq 0 \end{cases}$$

$$f(u) = \begin{cases} 0 & u \leq -\frac{1}{2} \\ u & -\frac{1}{2} < u < \frac{1}{2} \\ 1 & u \geq \frac{1}{2} \end{cases}$$

$$f(u) = \frac{1}{1 + e^{-\beta u}}$$

$$f(u) = e^{-(u/\sigma)^2}$$

$$f(u) = u$$

Figura 1.3.3. Funciones de activación.

1.3.3. Arquitectura de una Red Neuronal Artificial (RNA)

Las neuronas de una determinada red pertenecerán a una de las siguientes clases:

Neuronas de entrada: Reciben las señales exteriores. Integran la primera capa de la red.

Neuronas ocultas: Producen resultados intermedios.
Neuronas de salida: Su salida es observable desde el exterior. Constituyen la última capa de la red.

Por otra parte, dependiendo de la forma en que se estructuren las neuronas y conexiones, aparecen dos grandes tipos de redes:

- Redes progresivas.
- Redes realimentadas.

1.3.3.1. Redes progresivas

Las conexiones de sus neuronas (y, por lo tanto, el flujo de información) son siempre hacia adelante, de la entrada a la salida. No presentan memoria (la respuesta a una entrada determinada es independiente del estado anterior de la red). Por tanto, son estáticas.

Las arquitecturas progresivas de mayor importancia son el *perceptrón* y las de función de base radial (RBF).

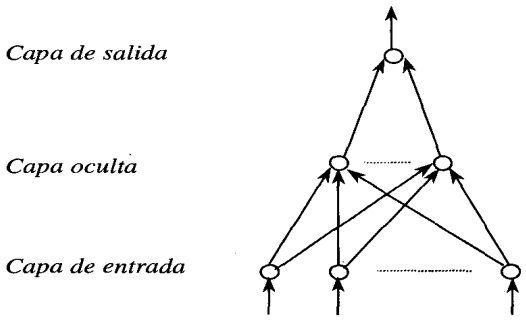


Figura 1.3.4. Red progresiva.

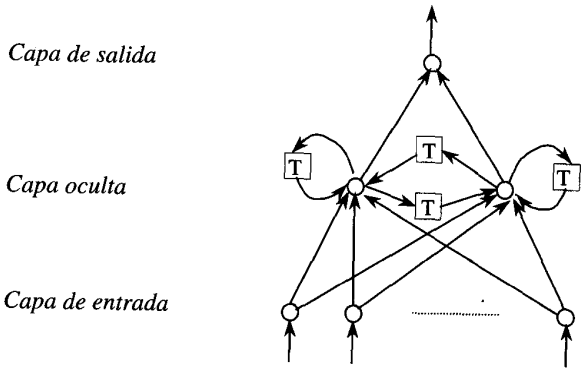


Figura 1.3.5. Red realimentada.

1.3.3.2. Redes realimentadas

A diferencia de las redes progresivas, presentan como mínimo un camino de realimentación, ya sea entre neuronas de una misma capa (conexiones laterales) o corno conexión hacia atrás, de una capa exterior a otra más interior.

Son sistemas dinámicos. Cada vez que se presenta una nueva entrada a la red, se calcula la salida de las neuronas. Debido a los lazos de realimentación, las entradas de las neuronas se modifican y la red entra en un nuevo estado.

Las arquitecturas más destacables son:

- Redes competitivas.
- Mapas autoorganizados de Kohonen.
- Red de Hopfield.
- Red de Elmann.
- Modelos ART.

1.3.4. Aprendizaje

Una vez fijada la arquitectura y el número de neuronas de la red, hay que calcular el valor de los pesos para que realice la función deseada. Usualmente, la red debe aprender los pesos de las conexiones a partir de un conjunto de patrones de entrenamiento. Por tanto, no se calculan los pesos de forma analítica sino que es necesario un proceso de aprendizaje automático a partir de una serie de ejemplos.

En vez de fijar una serie de reglas explícitas, es la red neuronal la que aprende las reglas implícitas que relacionada salida con la entrada, a partir de una colección de ejemplos representativos. Esta es una de las mayores ventajas de las redes neuronales sobre los sistemas tradicionales. Para diseñar un proceso de aprendizaje será necesario disponer de:

- Un conjunto de información disponible para el aprendizaje.
- Un proceso para actualizar los pesos, o algoritmo de aprendizaje.

Existen tres métodos básicos de aprendizaje:

Aprendizaje supervisado

La red conoce la salida correcta para cada patrón de entrada. Por tanto, se ajustan los pesos de forma que las salidas de la red sean lo más parecidas posibles a la salida correcta.

Aprendizaje por esfuerzo

Es una variante del método anterior, en el cual la red únicamente conoce una crítica sobre la exactitud de las salidas, pero no la respuesta correcta.

Utiliza un mecanismo de penalización y recompensa, mediante el cual la red resulta recompensada por una decisión correcta y castigada por una equivocada. De esta forma, el sistema tiende a actuar de la forma favorecida y debilita la tendencia a proporcionar la respuesta penalizada.

Aprendizaje autoorganizado

No requiere el conocimiento de la salida correcta, y debe aprender la estructura implícita en los datos. Generalmente requiere un número mayor de datos de entrenamiento.

También son posibles aprendizajes híbridos, combinando aprendizaje supervisado y no supervisado. En este caso, parte de los pesos se calculan mediante un método y la otra parte mediante el otro método.

En el aprendizaje, existen tres cuestiones importantes:

Capacidad de la red para implementar funciones y aprender patrones. Dependerá de su estructura, tamaño, etc.

Complejidad de los datos. Determina el número de patrones de entrenamiento necesarios para entrenar la red de forma que se garantice una buena capacidad de generalización (capacidad de procesar correctamente datos no utilizados en el aprendizaje). Si el número de patrones es pequeño, la red se comportará bien con aquellos patrones utilizados en el entrenamiento, pero presentará un comportamiento pobre con aquellos datos no utilizados.

Complejidad computacional. Condiciona el tiempo necesario para que el algoritmo de aprendizaje proporcione una solución ajustada.

El conjunto de patrones utilizados en el proceso de aprendizaje recibe el nombre de secuencia de entrenamiento. Cada vez que se ha presentado la totalidad de la secuencia de entrenamiento a la red,

se dice que se ha realizado un “epoch”. Normalmente los algoritmos que consumen poco tiempo en cada epoch requieren muchas epochs y viceversa, con lo cual la evaluación de la complejidad computacional debe considerar el número de epochs necesarios y tiempo transcurrido en cada epoch.

1.3.5. Aplicaciones de las RNA

Las principales aplicaciones de las redes neuronales aplicadas al tratamiento de voz e imagen se pueden dividir en:

Clasificación de patrones

Consiste en asignar un patrón de entrada, representado normalmente por un vector de características (que representan al patrón de entrada) a una de las clases predeterminada. Las principales aplicaciones en las que las redes neuronales actúan de forma satisfactoria como clasificadores son: reconocimiento de caracteres, del habla y de locutor.

Agrupaciones (clustering)

También se conoce como clasificación no supervisada de patrones. El algoritmo examina la semejanza entre patrones, y los agrupa en un cierto número de clases. Tiene aplicación en compresión de datos y como paso preliminar en aplicaciones de reconocimiento.

Aproximación de funciones

A partir de un conjunto finito de patrones (pares entrada-salida) se encuentra una función interpolativa que permite calcular la salida a entradas nuevas. Es una alternativa a los métodos de interpolación clásicos, con la ventaja de obtener fácilmente una función de interpolación no lineal. Es útil en voz e imagen para la interpolación de señal, parámetros, etc.

Predicción

Dado un conjunto de muestras de entrada hasta un determinado instante de tiempo, se pretende predecir la muestra siguiente en un tiempo futuro. La obtención de modelos predictivos para una determinada señal es ampliamente utilizada en tratamiento de voz e imagen en todos sus campos.

Destaca especialmente la predicción no lineal basada en red neurona!, por sus mayores prestaciones respecto a los métodos lineales.

2. FILTROS NO LINEALES

2.1. Concepto, Tipos de filtros

Son filtros no lineales aquellos que no cumplen el principio de superposición. En contrapartida, los filtros lineales sí cumplen dicho principio. Si bien filtros lineales presentan la ventaja de que existen herramientas matemáticas facilitan y simplifican su estudio y diseño, existen multitud de aplicaciones que se obtienen mejores resultados utilizando filtrado no lineal. Esto es a la naturaleza no lineal de un buen número de fenómenos como, por ejemplo, no linealidades debidas al sistema que genera la señal y/o el ruido, no linealidades del canal de transmisión, etc.

Para comprobar si un determinado sistema cumple el principio de superposición, habrá que comprobar si cumple las siguientes dos propiedades:

$$f(x) = y \rightarrow f(\alpha x) = \alpha y$$
$$f(x_1 + x_2) = f(x_1) + f(x_2)$$

La primera establece que, conocida la salida a una determinada entrada, si la entrada se escala por un factor multiplicativo, la salida queda afectada por ese mismo factor. Un ejemplo de función de transferencia que no cumple esta propiedad es la función cuadrática.

La segunda propiedad establece que, conocidas las salidas de un determinado sistema a dos entradas por separado, resulta inmediato conocer la salida entrada igual a la suma de las dos señales anteriores. Si el sistema es lineal , valor es igual a la suma de las salidas de cada una de las entradas por separado. Existen diversas filosofías para implementar filtros no lineales. Las clases utilizadas de filtros no lineales son:

- filtros de estadísticos ordenados,
- filtros morfológicos,
- filtros homomórficos,
- filtros polinómicos,
- filtros basados en redes neuronales.

Cada tipo de filtros no lineales se utiliza en aplicaciones diferentes. El mayor inconveniente en comparación con el procesado lineal de señal es la carencia de una teoría unificadora de las diferentes herramientas de tratamiento no lineal.

En este capítulo nos centraremos en los filtros de estadísticos ordenados. En general, los filtros de estadísticos ordenados resultan más adecuados para eliminar ruido de tipo impulsional y para mantener las transiciones bruscas. En cambio, los filtros lineales resultan más adecuados en zonas homogéneas de señal y ante ruido correlacionado de tipo gausiano.

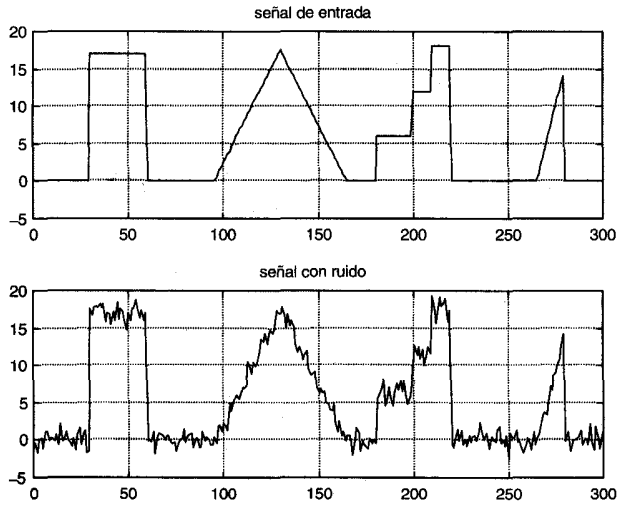


Figura 1.4.1. Señales original y afectada por ruido gausiano.

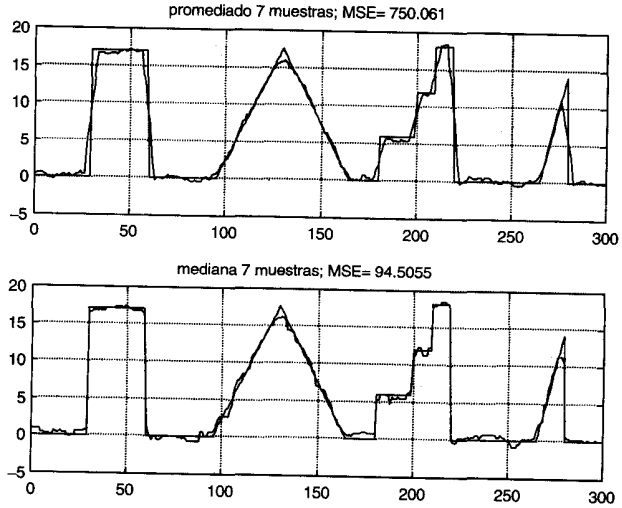


Figura 1.4.1 (continuación). Resultados de filtrar la señal ruidosa con filtros de promediado y mediana. Obsérvese como el filtro de mediana mantiene las transiciones bruscas de la señal original.

La figura 1.4.1 muestra el resultado de aplicar un filtro de promedio y un filtro de mediana de siete muestras. El primero es un filtro lineal, y el segundo estadísticos ordenados. Obsérvese que el error cuadrático medio de la señal filtrada respecto a la original es menor con el filtro no lineal.
En el apartado 3.2 "Mejora de imágenes" se presenta un ejemplo de eliminación de ruido en imágenes que compara diversos tipos de filtros.

A continuación se describen las principales implementaciones y aplicaciones de los filtros de estadísticos ordenados.

2.1.2. Filtros de estadísticos ordenados (OS)

En este tipo de filtros la salida se obtiene después de realizar una ordenación sobre las muestras de entrada al filtro. Si las muestras de entrada son:

$$x_1\, x_2 \cdots x_{2N+1}$$

Y las muestras ordenadas:

$$x_{[1]}\, x_{[2]} \cdots x_{[2N+1]}$$

Se pueden obtener filtros OS recursivos utilizando muestras ya filtradas previamente (es el equivalente a los filtros lineales ¡IR).
Entre las diferentes ordenaciones, destacan:

Ordenación global

La regla de ordenación es independiente de la información contenida en la ventana del filtro.

a) Ordenación temporal

No se realiza ninguna ordenación de la información contenida en el filtro. Sería el caso de los filtros lineales **FIR e IIR.**

Para el ejemplo comparativo de la figura 1.4.1 se utiliza un filtro FIR consistente en promediar 7 muestras (3 anteriores y 3 posteriores a la posición del filtro, así como la muestra central).

b) Ordenación natural o inversa

Se ordenan las muestras según el criterio X[i] <X[i+l] (ordenación natural) o el criterio inverso X[I] > X[i+ 1] .

Después de la ordenación se multiplican las muestras por unos pesos (x*wT). Según los pesos se obtienen, entre otros, los siguientes filtros:

- mediana: se toma el elemento central, por ejemplo: w = [0 0 0 1 0 0 01].
- mm: menor de los elementos, por ejemplo: w = [1 0 0 0 0 0 0].
- máx: máximo de los elementos, por ejemplo: w = [0 0 0 0 0 0 1].
- En general, orden K (rank OS): el que ocupa la posición K. Por ejemplo: w=[00 10000].
- cc-trimmed mean: pondera los elementos centrales, por ejemplo: w = [0 0
- 1/3 1/3 1/3 0 0].
- Otra posibilidad es calcular w según los datos a filtrar: w=[wi w2... W2N+1].

c) Ordenación absoluta

Idéntico al anterior, pero realizando el valor absoluto de las muestras de entrada antes de ordenarlas.

d) Ordenación de nivel

Generalización del caso anterior, ordenando los datos según su distancia respecto a una referencia o nivel arbitrario. (Antes de realizar el valor absoluto, se resta la referencia a todos los números.) Salvo aplicaciones específicas, cabe esperar un mejor comportamiento variando la referencia en función de la señal. Esta idea nos conduce al siguiente punto:

Ordenación local

La regla de ordenación depende de los datos presentes en la ventana.

a) Ordenación de media

Particularización de la ordenación de nivel, donde el nivel se toma igual a la media de los datos contenidos en la ventana.

b) Ordenación de mediana

Igual que el caso anterior, pero sustituyendo la media por la mediana.

c) Ordenación de variación

Las muestras se ordenan según su derivada unilateral, bilateral, diferencia respecto a la salida anterior del filtro, etc.

Filtro LUM smoother

La salida del filtro LUM smoother depende del parámetro K, y se obtiene mediante:

$$y = \text{median}\ \{x_{[K]}, x_{N+1}, x_{[N-K+1]}\}$$

donde el parámetro K puede variar entre:

$$1 \leq K \leq (N+1)/2$$

Y XN+1 es la muestra central de la ventana que queremos filtrar.

- El parámetro K define un margen "normal" de valores. Si la muestra
- filtrar está dentro de este margen, no se modifica. Si está fuera, se reemplaza por una muestra más cercana a la mediana.
- Para K= 1, resulta el filtro identidad.
- Para K máxima la salida es idéntica al filtro de mediana y se obtiene el máximo suavizado posible.
- Es equivalente al filtro center-weighted medians, en el cual se añade a las muestras de la ventana W—1 veces la muestra central, obteniendo una ventana ampliada. Ambos filtros son iguales efectuando:

W=N—2K+2

Filtro LUM sharpener

A diferencia del filtro anterior, se aleja la muestra a filtrar hacia la estadística ordenada extrema más cercana.

Se define el punto medio entre la estadística ordenada inferior y superior, en función de L como: la salida del filtro:

$$t_L = (x_{[L]} + x_{[N-L+1]})/2$$

y la salida del filtro:

$$\begin{array}{ll} y = x_{[L]} & \text{si } x_{[L]} < x_{N+1} \leq t_l \\ y = x_{[N-L+1]} & \text{si } t_l < x_{N+1} < x_{[N-L+1]} \\ y = x_N & \text{en otros casos} \end{array}$$

Si la muestra central está contenida en el margen de valores definido por el parámetro L, se interpreta como una muestra de transición en una zona de pendiente, y es sustituida por el límite más próximo del margen, de forma que se crea una transición más abrupta.

Filtro LUM

Es la combinación de los dos anteriores. Por tanto, se dispone de dos para metros: K y L

3. Tratamiento de imagen

3.1. INTRODUCCIÓN

Son técnicas de tratamiento digital de imagen todas aquellas manipulaciones realizadas sobre la imagen con el objetivo de:

• Mejorar su calidad eliminando degradaciones.
• Conseguir una representación eficiente que permita almacenarla en un determinado soporte con el menor número posible de bits sin una pérdida apreciable de calidad o requerir el menor tiempo posible de transmisión.
• Extraer información relevante de cara a interpretar su contenido, tomar decisiones, etc.

Si bien existe una gran interrelación entre las técnicas utilizadas en cada uno de los campos anteriores, las aplicaciones prácticas son muy diferentes, y cubren un amplio abanico de posibilidades, desde la industria y la medicina a las aplicaciones militares y de ocio.

En este libro pretende darse una visión global del estado del arte en cada una de las ramas del tratamiento de imagen, haciendo hincapié en las aplicaciones y prescindiendo, en lo posible, de la formulación matemática.

Los temas cubiertos son:

• Mejora de imágenes.
• Compresión.
• Tratamiento de secuencias de imágenes en movimiento.
• Morfología matemática.
• Visión artificial.

3.2. MEJORA DE IMÁGENES

3.2.1. Introducción

Son técnicas de mejora de imagen todos los algoritmos destinados a acentuar, resaltar, contrastar, etc., determinados aspectos de la imagen o eliminar efectos no deseados como, por ejemplo, desenfoques y ruidos.

Figura 3.2.1. Ecografía mejorada.

Figura 3.2.31.1, Ecografía Mejorada

Dos ejemplos típicos de aplicaciones en las que es necesario la mejora de imágenes son las imágenes médicas y las obtenidas vía satélite.

En el primer caso, las limitaciones del sistema de adquisición conducen a radiografías, ecografías, etc., de escaso contraste, que dificultan la discriminación de los distintos elementos presentes en la imagen (pulmones, costillas, corazón...). Mediante técnicas de tratamiento de imagen es posible resaltar los distintos elementos de forma que sean más claramente identificables por el ojo humano.
La figura 3.2.1 muestra un ejemplo de ecografía mejorada.

En el segundo caso, la imposibilidad de instalar emisores de potencia elevada en los satélites y la gran distancia que debe recorrer la imagen al ser transmitida, proporcionan imágenes con una relación señal ruido baja, que puede ser mejorada mediante técnicas de eliminación de ruido.

Es importante destacar que mediante la mejora de imágenes:

- No se añade información nueva no presente en la imagen. Simplemente se resalta la existente, para que sea mejor apreciada por el ojo humano.
- No existe un criterio para cuantificar el grado de mejora que introducen las diferentes técnicas. La valoración de los resultados es subjetiva, y a menudo se trata de métodos empíricos que requieren procedimientos de prueba y error hasta obtener el resultado deseado.

Para simplificar el estudio de las diferentes técnicas, las dividiremos en las siguientes categorías: **operaciones puntuales, espaciales y transformadas, restauración de imágenes y pseudocolor.**

3.2.2. Operaciones puntuales

Se procesa cada pixel de la imagen por separado, independientemente del valor de los pixels vecinos. Se Irala, por lo tanto, de operaciones sin memoria, en las que un nivel de gris u =1(1, m) perteneciente al pixel de coordenadas (1, m) dentro de la imagen 1 (x, y) y al margen de valores [02B — 1] (donde B es el número de bits por pixel de la imagen), se convierte en otro nivel de gris u'E [0,2B_1] mediante una determinada función *u'=f(u).*

3.2.2.1. Función definida a intervalos

Un ejemplo es la función definida por tres intervalos representada en la figura 3.2.2.

$$f(u) = \begin{cases} m_1 \cdot u & 0 \le u < a \\ m_2 \cdot (u-a) & a \le u < b \\ m_3 \cdot (u-b) & b \le u < 2^B \end{cases}$$

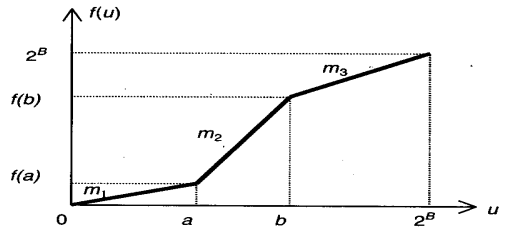


Figura 3.2.2. Función definida por tres intervalos.

Existen dos casos particulares de especial importancia:

- mi=m3=O y a<b

Se utiliza en aquellos casos en los que el dispositivo de captación (cámara, escáner, etc.) no aprovecha todo el margen dinámico disponible, o existe ruido

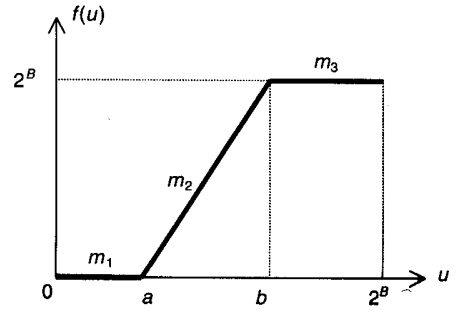


Figura 3.2.3. Caso particular $m_1 = m_3$, $a < b$.

fuera del margen [a, b] que debe ser eliminado. La figura 3.2.3 representa la función a utilizar.

- Simi=m3=Oya=b=umbral

Se utiliza para obtener imágenes monocromas a partir de imágenes en tonos de gris. Todos los pixels por debajo del umbral tomarán el valor cero (negro) y los que estén por encima el nivel más alto (blanco). Si, posteriormente, se almacena la imagen con 1 bit/pixel la función será la mostrada en la figura 3.2.4.

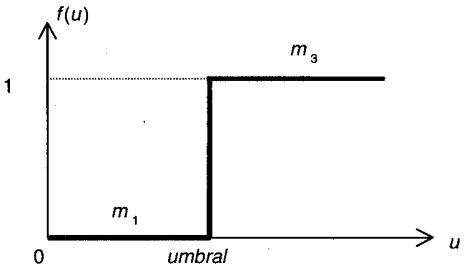


Figura 3.2.4. Función de binarización.

3.2.2.2. Negativo de una imagen

Intercambiando los niveles blanco y negro entre sí mediante la función:

$$f(u)=(2\text{^}B) - u$$

se obtiene el negativo digital de la imagen. La figura 3.2.5 muestra la representación gráfica de la función f(u), y la figura 3.2.6 un ejemplo de negativo de una imagen.

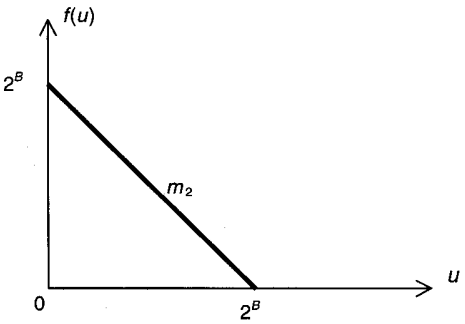


Figura 3.2.5. Función para obtener el negativo.

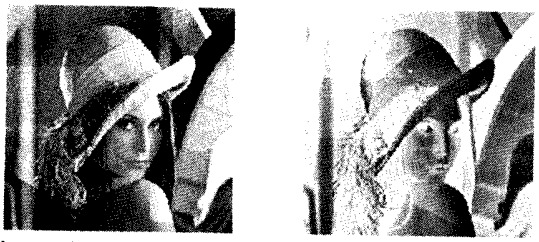


Figura 3.2.6. Imagen **Pamela** *original y su negativo.*

3.2.2.3. Extracción de bits

A partir de una imagen cuantificada a B bits/pixel, podemos extraer y representar cualquiera de los B bits.

Expresando el valor de u en binario:

$$u = b_1 \cdot 2^{B-1} + b_2 \cdot 2^{B-2} + \cdots + b_n \cdot 2^{B-n} + \cdots + b_{B-1} \cdot 2 + b_B$$

la imagen para el bit n-ésimo se formará con la relación:

$$f(u) = \begin{cases} 2^B & \text{si } b_n = 1 \\ 0 & \text{si } b_n = 0 \end{cases}$$

donde b se puede obtener como: $b_n = i_n - 2 \cdot i_{n-1}$

$$\text{con } i_n = \text{int}\left(\frac{u}{2^{B-n}}\right)$$

siendo int(x) la función que retorna la parte entera de x.

Figura 3.2.7. Mapas de los 8 bits de la imagen **Pamela.**

La figura 3.2.7 muestra los ocho mapas de bits de la imagen Pamela ordenados del MSB (bit más significativo) al LSB (bit menos significativo).

Esta operación resulta útil para determinar el número de bits significativos de una imagen. A partir de la figura 3.2.7, se deduce que los tres bits menos significativos no aportan información (sólo ruido), y sería posible almacenar la imagen con únicamente 5 bits/pixel sin una pérdida apreciable de detalle en la imagen.

3.2.2.4. Procedimientos basados en el histograma

El histograma de una imagen es la representación del número de pixels que posee un determinado nivel de gris, para todos los niveles de gris posibles de la imagen.

Las funciones de mejora basadas en el histograma son principalmente la ecualización y la especificación del histograma.

En la ecualización del histograma, el objetivo es obtener en la imagen de salida un histograma uniforme, mientras que en la especificación del histograma se fuerza a la imagen de salida a tener un determinado histograma. En [Jain] se puede obtener más información de las operaciones matemáticas que deben realizarse.

Otra aplicación del histograma es como herramienta de análisis de imágenes, útil para la obtención de soluciones. Los siguientes problemas son un ejemplo.

Ejemplo 3.2.1.

Un texto monocromo escaneado presenta un histograma como el de la figura 3.2.8. En principio, debería presentar únicamente dos picos, uno para el nivel negro del texto y otro para el fondo blanco. Sin embargo, el dispositivo de adquisición es imperfecto y proporciona más de dos niveles de gris. Para binarizar la imagen podemos aplicar la función de umbral de la figura 3.2.4, utilizando el histograma como herramienta de análisis para encontrar un umbral adecuado que nos permita separar ambos picos. Sin utilizar el histograma, se deberían hacer varias pruebas hasta obtener un umbral correcto.

Figura 3.2.8. Imagen monocroma escaneada y su histograma.

La figura 3.2.9 muestra los resultados obtenidos para diversos umbrales Los histogramas resultantes.

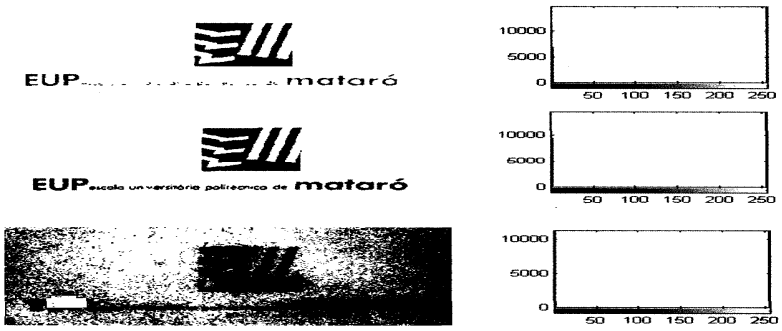


Figura 3.2.9. Tres binarizaciones distintas a niveles 40, lOOyL3O de gris respec1iimente. Como puede observarse en el histograma de la figura 3.2.8, el umbral adecuado es 100.

Ejemplo 3.2.2.

La figura 3.2.10 presenta una imagen y su histograma.

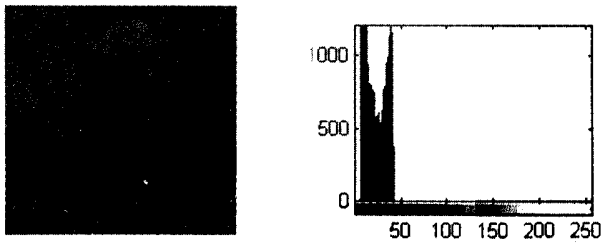


Figura 3.2.10. Imagen oscura y su histograma.

Se puede observar que es una imagen muy oscura en la que es difícil discernir su contenido. A partir del histograma se comprueba que, ciertamente, todos los pixels presentan niveles de bajos grises niveles (cercanos al negro), pero no todos son iguales. Por tanto, aumentando la diferencia entre los diversos niveles de gris y aprovechando mejor el margen dinámico, será posible apreciar mejor el contenido de la imagen. Para ello, se reparten los niveles de gris en todo el margen dinámico y se obtiene la figura 3.2.11, en la cual resulta más fácil interpretar su contenido.

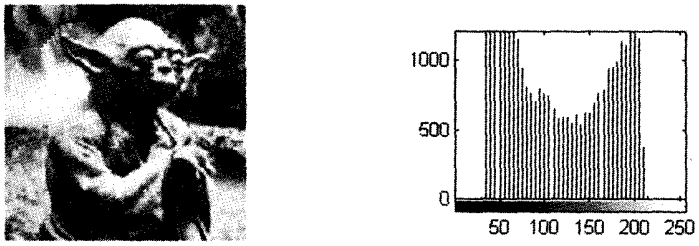


Figura 3.2.11. Imagen ecualizada y su histograma.

3.2.3. Operaciones espaciales

A diferencia de los métodos descritos en el apartado anterior, se modifica el valor de cada pixel en función del valor de un conjunto de pixels vecinos. A continuación se describen los procedimientos básicos

3.2.3.1. Filtrado espacial

Mediante la función *f(I1 (x,y))* se obtiene la imagen ‘2 en la que cada pixel de I se ha substituido por un promedio ponderado de pixels vecinos. La expresión matemática es:

$$f(I_1(x,y)) = \sum_{i,j \in W} \sum \alpha(i,j) I_1(x-i, y-j) \qquad \text{(ec.3.2.1)}$$

En la que:

- x, y hacen referencia a la posición del pixel que se está filtrando.
- W se extiende al conjunto de pixels utilizado en los cálculos (vecindario).
- a(i, j) son los coeficientes de ponderación del filtro.

Una interpretación equivalente es convolucionar la imagen con la respuesta impulsional del filtro, que puede representarse mediante una plantilla.

Un caso particular es considerar todos los coeficientes iguales, con lo cual se está realizando un promediado.

Si W es de 3 x 3, la expresión particularizada es:

$$I_2 = f(I_1(x,y)) = \frac{1}{9} \cdot \sum_{i,j \in W} \sum I_1(x-i, y-j)$$

A la que corresponde a la plantilla de la figura 3.2.1.2.

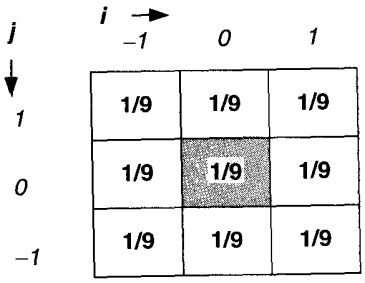
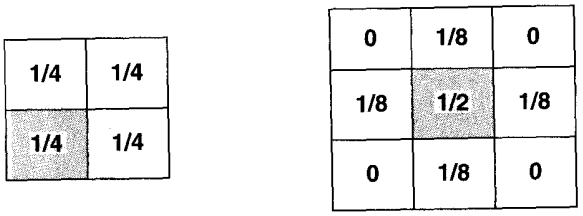


Figura 3.2.12. Plantilla de promediado 3×3.

$$I_2 = \frac{1}{9}\big(I_1(x,y) + I_1(x-1,y-1) + I_1(x,y-1) + I_1(x+1,y-1) +$$
$$+ I_1(x-1,y) + I_1(x+1,y) + I_1(x-1,y+1) + I_1(x,y+1) + I_1(x+1,y+1)\big)$$

Es posible realizar promediados tomando vecindarios de tamaño diferente. Por ejemplo, los mostrados en la figura 3.2.13.



Promediado de 4 puntos *Promediado de 5 puntos*

Figura 3.2.13. Ejemplos de otras plantillas de promediado.

Siempre hay que indicar la posición del pixel central que se está filtrado (i=j=o). En la figura 3.2.13 se ha marcado sombreando la casilla que corresponde a esta posición.

El filtrado de una imagen de M x N pixels consiste en calcular 12(x,y)para todos los M x N pixels mediante la ecuación (ec. 3.2.1), o análogamente plazar la plantilla barriendo todas las posiciones de la imagen, y para de ellas dar como salida la suma del producto de todos los coeficientes valor de los pixels sobre los que están (figura 3.2.14).

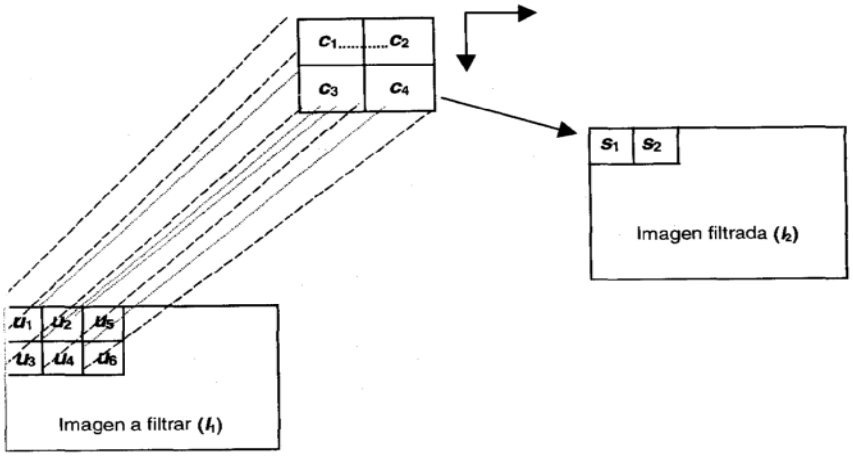


Figura 3.2.14. Proceso de filtrado con plantilla.

$$S_1 = c_1 \cdot u_1 + c_2 \cdot u_2 + c_3 \cdot u_3 + c_4 \cdot u_4$$

$$S_2 = c_1 \cdot u_2 + c_2 \cdot u_5 + c_3 \cdot u_4 + c_4 \cdot u_6$$

La figura 3.2.15 muestra un ejemplo de aplicar un promediado espacial 2x2, 3x3 y 4x4 a una imagen corrompida con ruido gausiano. Obsérvese la reducción de ruido y el aspecto borroso de la imagen resultante causado por el promediado.

Figura 3.2.15. a) Imagen corrompida con ruido gausiano; b) filtrado espacial 2×2; c) filtrado espacial 3×3; d) filtrado espacial 4×4.

Para evaluar objetivamente la mejora obtenida en el proceso de filtrado, puede calcularse el error cuadrático medio (MSE), definido como la suma de la diferencia de los pixels de la imagen original y filtrada, elevada al cuadrado. La tabla 3.2.1 presenta los resultados. Se ha incluido el error MSE entre la

imagen original y la corrompida con ruido, para poder apreciar la mejora introducida en el proceso de filtrado.

Tabla 3.2.1. Errores MSE entre la imagen original y las filtradas.

	Error cuadrático medio
Original ↔ Corrompida	614,4221
Original ↔ Filtrado 2×2	336,6628
Original ↔ Filtrado 3×3	211,8597
Original ↔ Filtrado 4×4	290,0359

A partir de la tabla 3.2.1 se deduce que la imagen que presenta una mayor fidelidad a la original es la e) (filtrado con ventana de 3x3).

Si W es pequeña, la probabilidad de que contenga pixels con valores semejantes es elevada. A medida que aumente su tamaño, esta probabilidad disminuirá y, por tanto, para tamaños grandes se perderá una cantidad notable de información.

Dado que la notación con plantillas es más compacta e intuitiva que la expresión matemática, suele utilizarse ésta con mayor frecuencia.

–1/8	**–1/8**	**–1/8**
–1/8	**1**	**–1/8**
–1/8	**–1/8**	**–1/8**

gura 3.2.16. Plantilla para filtrado de paso alto.

Mediante filtrado espacial, es posible realizar un elevado número de funciones. En el apartado 3.6 (Visión artificial) se explicará la detección de puntos aislados, rectas, transiciones, etc.

La figura 3.2.16 muestra una plantilla para implementar un filtrado de paso alto. Obsérvese en la figura 3.2. 17 que el efecto producido es resaltar las transiciones.

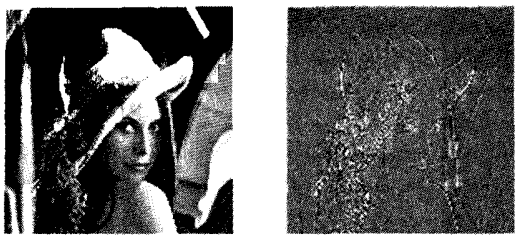


Figura 3.2.17. Imagen original y filtrada de paso alto.

3.2.3.2. Suavizado direccional

El promediado espacial visto en el ejemplo anterior presenta el inconveniente de desenfocar los contornos. Para evitarlo, se puede recurrir a un promediado direccional. Este método consiste en aplicar promediados espaciales en diversas direcciones y escoger la dirección que proporciona una diferencia menor entre el valor filtrado y el original (antes de filtrar). El problema siguiente ilustra su funcionamiento.

Ejemplo 3.2.3.

Aplíquese un filtro de suavizado direccional de cuatro direcciones (figura 3.2.1 8a) para filtrar el pixel central de la porción de imagen representada en la figura 3.2.1 8b. Considérese que el filtro en cada dirección consiste en realizar un promediado de los pixels de la ventana. ¿Cuál es la dirección escogida?

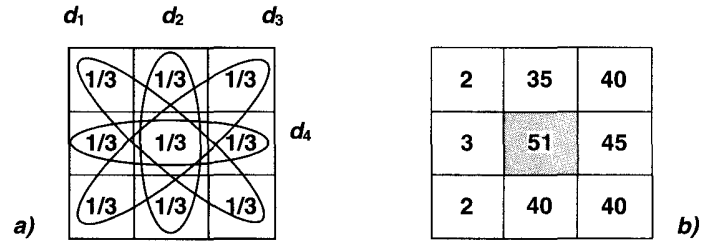


Figura 3.2.18. a) Plantilla para suavizado direccional de cuatro direcciones; b) porción de imagen a filtrar.

Solución

El valor a filtrar es 51 y las salidas en las cuatro direcciones son:

$$d_1 = \frac{1}{3}\cdot[2+51+40] = 31$$

$$d_2 = \frac{1}{3}\cdot[35+51+40] = 42$$

$$d_3 = \frac{1}{3}\cdot[40+51+2] = 3$$

$$d_4 = \frac{1}{3}\cdot[3+51+45] = 3$$

Y las diferencias respecto al valor antes de filtrar:

$$51 - d_1 = 20$$
$$51 - d_2 = 9$$
$$51 - d_3 = 20$$
$$51 - d_4 = 18$$

La dirección de menor diferencia es la 2 y, por tanto, el resultado de filtrar pixel central es 42. Obsérvese que se ha filtrado respetando la dirección de la transición horizontal de la primera columna a las siguientes.

Con un promediado 3x3 se habría obtenido un resultado 28,6=29, que habría modificado considerablemente el contorno.

En la figura 3.2.20 se observa que el suavizado direccional preserva mejor los contornos que el promediado espacial y, a la vez, elimina una cantidad considerable de ruido.

3.2.3.3. Filtrado de mediana

Es análogo al promediado espacial, pero en vez de realizar el promediado d los pixels de la ventana, se calcula la mediana.

La mediana se obtiene ordenando los valores contenidos en la ventana (ordenación natural) y escogiendo el valor central. En el caso en que el número de pixels contenidos en la ventana sea par, se realiza la media entre los dos valore centrales.

Ejemplo 3.2.4.

Calcular el resultado de filtrar el pixel central de la porción de imagen representada en la figura 3.2.19 con un filtro de mediana de 3x3.

10	10	10
10	40	8
8	8	10

Figura 3.2.19. Porción de imagen a filtrar.

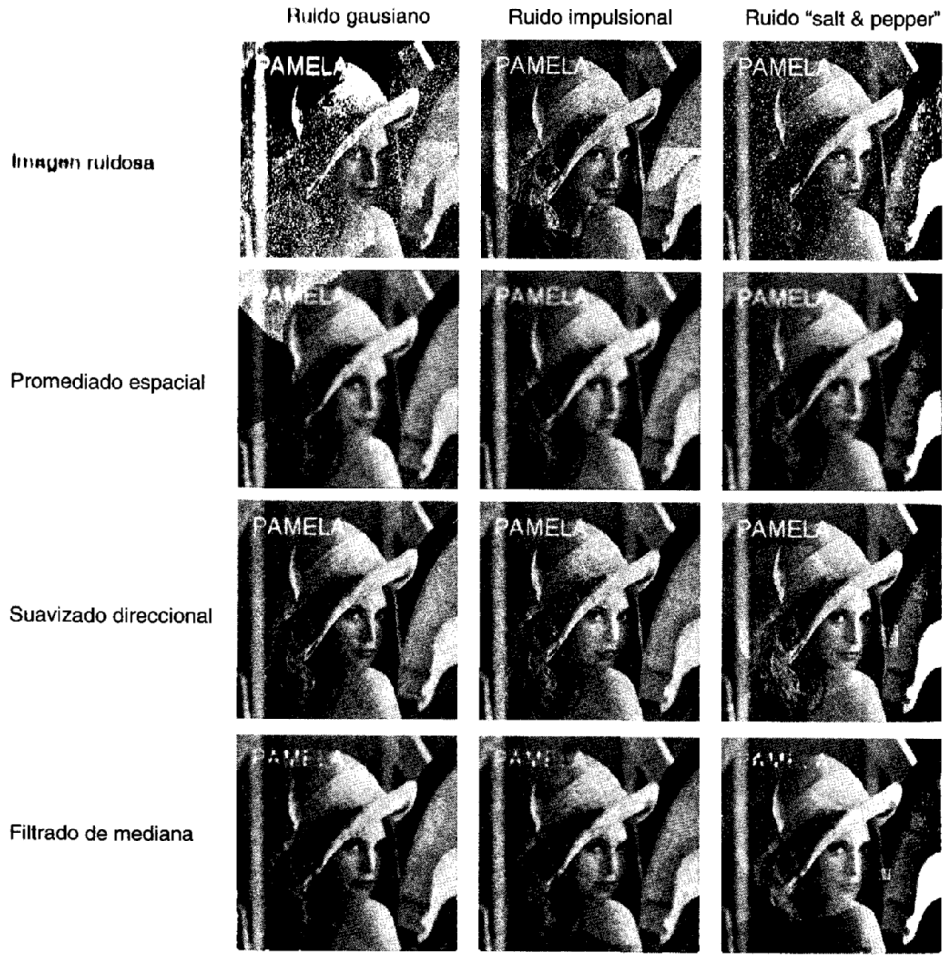


Figura 3.2.20. Comparación de resultados para diferentes métodos y ruidos.

Solución.

Al realizar la ordenación obtenemos el conjunto {8,8,8,10,10,10,10,10,40}. El elemento central es el quinto y su valor es 10. Por tanto, sustituiremos el valor 40 por 10. Este filtro resulta especialmente indicado para aquellos casos en los que pretenda eliminarse ruido impulsional. En el caso de ruido gausiano, es preferible el promediado espacial o el suavizado direccional.

Computacionalmente, es más costoso que el promediado espacial, puesto que requiere una operación de ordenación. Sin embargo, existen algoritmos eficientes para implementar el filtrado de mediana en un tiempo pequeño.

La figura 3.2.20 compara los resultados obtenidos mediante filtrado de mediana, promediado espacial y suavizado direccional al aplicarlo sobre una imagen corrompida con ruido gausiano, impulsional y "salt & pepper" (se denomina ruido de "sal y pimienta" a la aparición de los puntos blancos y negros sobre la imagen).

La tabla 3.2.2 muestra los errores cuadráticos medios obtenidos con cada uno de los métodos. Obsérvese que en el ejemplo de la figura 3.2.20 el suavizado direccional siempre proporciona el mejor resultado numérico, y preserva mejor las letras.

Tabla 3.2.2. Errores MSE entre la imagen original y las filtradas.

	Error cuadrático medio		
Ruido	**Gausiano**	**Impulsional**	**Salt & pepper**
Original ↔ Ruidosa	621,3961	425,3942	980,6522
Original ↔ Promediado espacial	408,4900	401,5712	434,5825
Original ↔ Suavizado direccional	256,4418	208,3129	154,9508
Original ↔ Filtrado de mediana	389,6372	402,2701	364,4770

3.2.3.4. Ampliación de imágenes

En ocasiones es necesario aumentar el tamaño de una imagen, o una parte de ella, con la finalidad de apreciar mejor sus detalles. Para ello existen diversos métodos.

Método 1: Repetición

Para pasar de una imagen I de MxN pixeis a otra imagen ‘2 de kMxkN (k 2) el método más simple consiste en repetir cada pixel de todas las líneas k veces, y a continuación cada línea k veces. Este proceso es equivalente a:

• Intercalar k columnas y k filas entre sus pixels, de valor 0, obteniendo una imagen de kMxkNpixels.
• Convolucionar la imagen del punto 1 con la matriz unitaria H de kxk elementos, todos a uno.

$$H = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \quad \text{para un orden } k = 2$$

Matemáticamente, la función aplicada es:

$$I_2(x, y) = I_1(i, j) \text{ con } i = \text{int}\left(\frac{x}{k}\right), j = \text{int}\left(\frac{y}{k}\right)$$

$$\text{para} \begin{cases} x = 1,\ldots, kM \\ y = 1,\ldots, kN \end{cases}$$

La figura 3.2.2 1 muestra el resultado de amplificar una porción de imagen mediante el método de repetición. Obsérvese que para factores de repetición elevados se aprecia un “efecto bloque” (la imagen parece estar formada por cuadrados de diferentes niveles de gris).

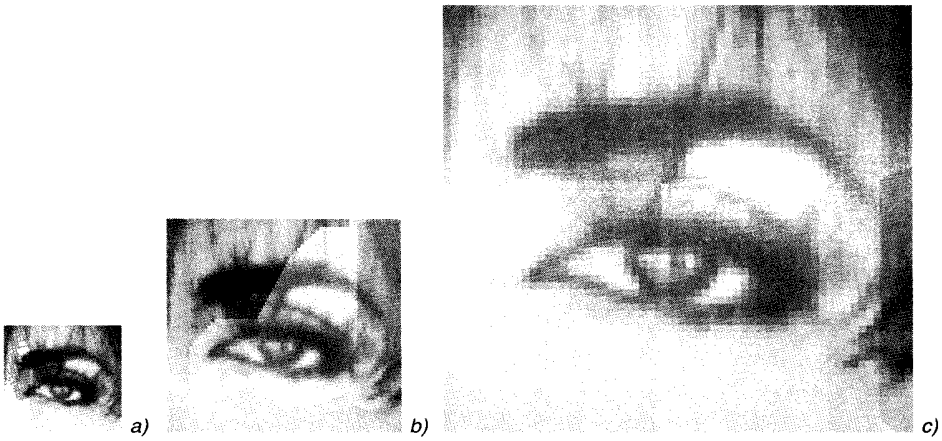


Figura 3.2.21. Ampliación de imágenes por repetición. a) Imagen original; b) ampliación con factor 2; c) ampliación con factor 4.

Método 2: Interpolación lineal

Para evitar el efecto bloque del método anterior, el valor de los pixels añadidos no se calcula por repetición sino por interpolación lineal, de forma que se produce una transición suave entre pixels y, por tanto, no aparecen transiciones bruscas. Sin embargo, para órdenes de ampliación elevados, la imagen resultante queda borrosa.

El procedimiento consiste en ajustar una recta entre los pixels de valor conocido de todas las líneas y después entre las columnas.

Mediante plantillas, las operaciones son semejantes al método de repetición, usando la plantilla:

$$H = \begin{bmatrix} 1/4 & 1/2 & 1/4 \\ 1/2 & \textcircled{1} & 1/2 \\ 1/4 & 1/2 & 1/4 \end{bmatrix} \qquad \text{para un orden } k = 2$$

Para una k genérica, hay que aplicar la plantilla H k veces, obteniendo una interpolación de orden k.

Ejemplo 3.2.5.

Aplicar la imagen definida en la figura 3.2.22 mediante interpolación de orden *k=2.*

5	**15**
11	**25**

Figura 3.2.22.

Solución:

En primer lugar, intercalamos ceros obteniendo la imagen:

$$\begin{bmatrix} 5 & 0 & 15 & 0 \\ 0 & 0 & 0 & 0 \\ 11 & 0 & 25 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

A continuación, convolucionamos el resultado con la matriz H, obteniendo:

$$\begin{bmatrix} 5 & 0 & 15 & 0 \\ 0 & 14 & 20 & 0 \\ 11 & 18 & 25 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Obsérvese que los contornos son un caso crítico, ya que nos faltan datos para poder aplicar la plantilla. Por tanto, deben considerarse aparte, como una excepción. Una posible solución es repetir la última fila y la última columna, e interpolar los valores en la primera fila y la primera columna usando los dos pixels cercanos.
El resultado es:

$$\begin{bmatrix} 5 & 10 & 15 & 15 \\ 8 & 14 & 20 & 20 \\ 11 & 18 & 25 & 25 \\ 11 & 18 & 25 & 25 \end{bmatrix}$$

La figura 3.2.23 muestra distintos factores de ampliación por interpolación al, sobre la misma imagen de la figura 3.2.21.

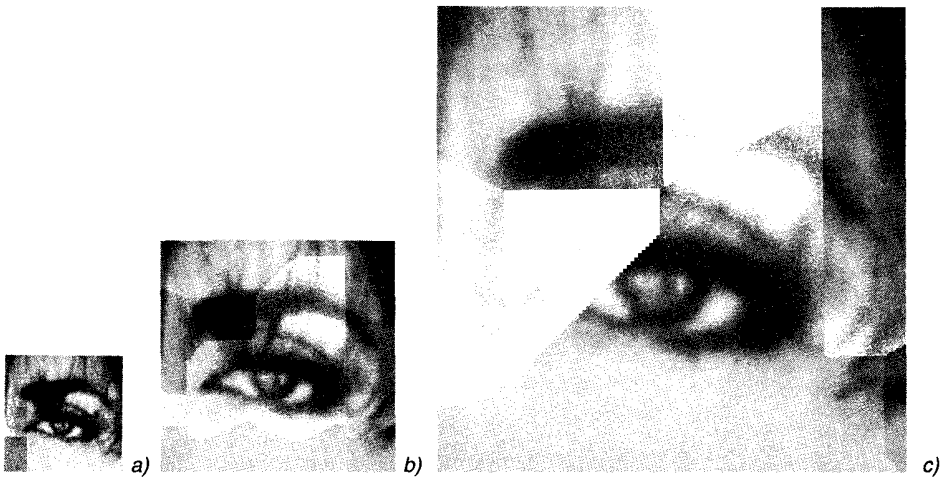


Figura 3.2.23. Ampliación de imágenes por interpolación lineal. a) Imagen original; b) ampliación con factor 2; c) ampliación con factor 4.

3.2.4. Operaciones transformadas

diferencia de los métodos anteriores, no se modifica directamente el valor de los pixeis de la imagen, sino que se realiza una transformación previa. Sobre la imagen transformada resultante se aplican operaciones puntuales y, por último, se calcula la transformada inversa.

Las transformadas más utilizadas son:

La DFT (Transformada de Fourier Discreta):

Transformada Directa:

$$\tilde{I}(\tilde{x},\tilde{y}) = \begin{cases} \sum_{x=0}^{N-1}\sum_{y=0}^{M-1} I(x,y)\cdot e^{-j(2\pi/N)\tilde{x}x}\cdot e^{-j(2\pi/M)\tilde{y}y}, \\ \qquad \text{para} \quad 0 \le \tilde{x} \le N-1\ ,\ 0 \le \tilde{y} \le M-1 \\ 0, \qquad \text{otros casos} \end{cases}$$

Transformada inversa:

$$I(x,y) = \begin{cases} \dfrac{1}{N\cdot M}\sum_{\tilde{x}=0}^{N-1}\sum_{\tilde{y}=0}^{M-1} \tilde{I}(\tilde{x},\tilde{y})\cdot e^{j(2\pi/N)\tilde{x}x}\cdot e^{j(2\pi/M)\tilde{y}y}, \\ \qquad \text{para} \quad 0 \le x \le N-1\ ,\ 0 \le y \le M-1 \\ 0 \qquad \text{otros casos} \end{cases}$$

La DCT (Trasformada Coseno Discreto):

Trasformada Directa:

$$C(\tilde{x},\tilde{y}) = \begin{cases} \sum_{x=0}^{N-1}\sum_{y=0}^{M-1} 4I(x,y)\cos\dfrac{\pi}{2N}\tilde{x}(2x+1)\cos\dfrac{\pi}{2M}\tilde{y}(2y+1), \\ \qquad \text{para} \quad 0 \le \tilde{x} \le N-1,\ 0 \le \tilde{y} \le M-1 \\ 0, \qquad \text{otros casos} \end{cases}$$

Trasformada Inversa:

$$I(x,y) = \begin{cases} \dfrac{1}{NM}\sum_{\tilde{x}=0}^{N-1}\sum_{\tilde{y}=0}^{M-1} w_x(\tilde{x})w_y(\tilde{y})C(\tilde{x},\tilde{y})\cos\dfrac{\pi}{2N}\tilde{x}(2x+1)\cos\dfrac{\pi}{2M}\tilde{y}(2y+1), \\ \qquad \text{para} \quad 0 \le x \le N-1,\ 0 \le y \le M-1 \\ 0, \qquad \text{otros casos} \end{cases}$$

donde

$$w_x(\tilde{x}) = \begin{cases} \frac{1}{2}, & \tilde{x} = 0 \\ 1, & 1 \le \tilde{x} \le N-1 \end{cases}$$
$$w_y(\tilde{y}) = \begin{cases} \frac{1}{2}, & \tilde{y} = 0 \\ 1, & 1 \le \tilde{y} \le M-1 \end{cases}$$

Y la transformada de Hadamard:

Si M=N, se puede calcular la transformada de la imagen mediante el producto de matrices:

$$\tilde{I} = A \cdot I \cdot A^T,$$

Y la trasformada inversa mediante:

$$I = A^{*T} \cdot \tilde{I} \cdot A^{*T},$$

donde:

I es la imagen a transformar,

A es la matriz de transformación,

A^T es la traspuesta de la matriz de transformación, y

A^*T es la traspuesta de la matriz conjugada de la matriz de transformación.

La matriz de transformación de Hadamard de NxN (Hn) puede obtenerse a partir de:

$$H_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Y la siguiente recursión:

$$H_n = \frac{1}{\sqrt{2}} \begin{pmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{pmatrix} \qquad \text{(ec. 3.2.3)}$$

Aplica hasta k *n =N,* de forma que:

$$\tilde{I} = H_n \cdot I \cdot H_n$$
$$I = H_n \cdot \tilde{I} \cdot H_n$$

puesto que H = Hn* = Hn^T.

Obsérvese que los elementos de la matriz siempre valen +1 o —1, con lo cual s posible implementar la transformación de forma que realice menos opera iones que la DFT o la DCT.

Ejemplo 3.2.6.

Hallar la matriz de Hadamard H3.

Aplicando dos veces la iteración de la ecuación (ec. 3.2.3), se obtiene:

$$H_3 = \frac{1}{\sqrt{8}} \cdot \left[\begin{array}{cc|cc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ \hline 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right]$$

Ejemplo 3.2.7.

Calcular la trasformada de Hadamard de la imagen:

$$I = \begin{pmatrix} 105 & 105 \\ 105 & 105 \end{pmatrix}$$

Solución: aplicado la solución (ec. 3.2.2):

$$\tilde{I} = \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \cdot \begin{pmatrix} 105 & 105 \\ 105 & 105 \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \cdot \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$\tilde{I} = \frac{1}{2} \cdot \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \cdot \begin{pmatrix} 210 & 0 \\ 210 & 0 \end{pmatrix} = \frac{1}{2} \cdot \begin{pmatrix} 420 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 210 & 0 \\ 0 & 0 \end{pmatrix}$$

Obsérvese que la imagen transformada sólo presenta baja frecuencia, ya que la imagen original era homogénea (todos los pixels tienen un nivel de gris 105).

Si la transformación escogida es la DFT, será posible observar el contenido frecuencial bidimensional de la imagen. La figura 3.2.24 muestra los distinto genes frecuenciales.

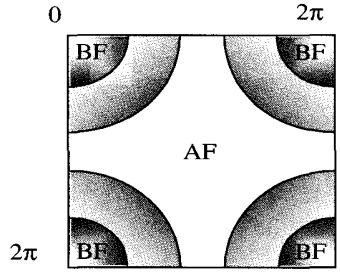
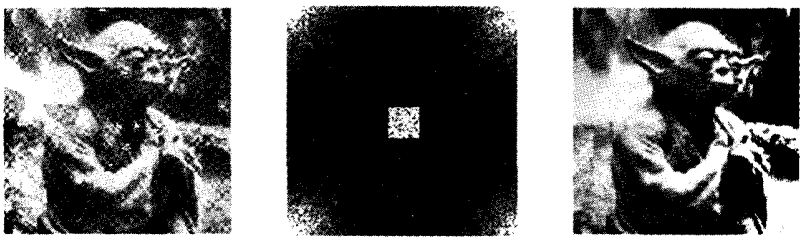


Figura 3.2.24. Márgenes de frecuencia (AF: alta frecuencia, BF: baja frecuencia).

Ejemplo 3.2.8.

La imagen de la figura 3.2.25a presenta un patrón de alta frecuencia difícilmente eliminable mediante los métodos vistos en los apartados anteriores. Sin embargo, realizando la DFT de la imagen se observa un cuadrado en 1 zona de alta frecuencia que corresponde a la interferencia (fig. 3.2.25b).

Si eliminamos esas frecuencias mediante una máscara zonal que valga cero en esa zona y uno en el resto, conseguiremos eliminarla (fig. 3.2.25c).



a) b) c)

'igura 3.2.25. Filtrado de paso bajo. a) Imagen corrompida; b) transformada DFT c) imagen filtrada.

Ejemplo 3.2.9: Filtrado de raíz.

Expresando la imagen transformada en forma módulo y argumento:

$$I_z(x,y) = |I_z(x,y)|\, e^{j\varphi(x,y)}$$

Se puede modificar el módulo manteniendo inalterada la fase.

Examinando el valor de los coeficientes, se observa que son más pequeños los correspondientes a frecuencias altas. Por tanto, si realizamos la raíz n-ésima del módulo, se consiguen enfatizar las altas frecuencias.

La figura 3.2.26 presenta los resultados obtenidos al filtrar la imagen Pamela con a= 0,95, a= 0,99 y a= 1,05.

a) b) c) d)

Figura 3.2.26. Filtrado de raíz. a) Imagen original; b) $\alpha = 0{,}95$; c) $\alpha = 0{,}99$; d) $\alpha = 1{,}05$.

3.2.5. Restauración de imágenes

En determinadas aplicaciones, es posible conocer la causa que ha degradado las imágenes, y disponer de un modelo de la degradación. Entre los casos más habituales cabe destacar: ruido del sensor, imagen desenfocada, imagen movida, etc.

El objetivo será restaurar estas degradaciones y la metodología será distinta de la mejora de imágenes. Estará basada en el conocimiento de la degradación, en vez de ajustarse a criterios subjetivos sobre la imagen en particular.

3.2.6. Pseudocolor

El ojo humano es capaz de distinguir un número mucho mayor de colores que de niveles de gris. Además, resultan mucho más agradables las imágenes en color que las de blanco y negro. Por tanto, se puede utilizar el color para mejorar la calidad subjetiva de las imágenes. Básicamente existen dos grupos de aplicaciones:

Falso color

Consiste en cambiar el color de los objetos con el fin de enfatizarlos. ejemplo, en una imagen en color de una serie de piezas, señalar una de ellas mercado su color a rojo.

Pseudocolor

Consiste en asignar colores a una imagen monocroma o a una serie de i senes monocromas para formar una imagen en color. Lógicamente, las posibles asignaciones son muchas, y normalmente se hace un proceso de prueba y error hasta conseguir el resultado deseado.

Por ejemplo, a partir de una imagen monocroma puede filtrarse en paso bajo, paso de banda y paso alto y asignarle a las imágenes resultantes los colores azul, verde y rojo. La combinación de las tres imágenes proporcionará imagen en color. Los colores resultantes no serán naturales. Sin embargo Imagen coloreada puede ser más fácilmente interpretable (**por ejemplo. Imágenes de rayos X utilizadas para controlar los equipajes en los aeropuertos las imágenes ecográficas**).

3.3. COMPRESIÓN

3.3.1. Introducción. Tipos y aplicaciones

Mediante la compresión de imágenes se pretende reducir el número de necesarios para transmitir o almacenar imágenes.

Según si el proceso de compresión introduce error o no, se puede establecer tina primera clasificación en métodos sin error y métodos con error. Sus principales características son:

Métodos de compresión sin error

El proceso de compresión es reversible. Por tanto, a partir de la representación compactada de la imagen (con menos bits) es posible recuperar la imagen original exacta previa a la compresión.

La principal característica de estos métodos es que los factores de comisión (relación entre el número de bits que ocupa la imagen antes de ser comprimida respecto a la imagen comprimida) no son muy elevados.

Por otra parte, la mayoría de métodos no garantizan que exista compres En determinados casos, el resultado puede ser una expansión del número bits necesarios para representar la imagen.

Métodos de compresión con error

A diferencia del grupo anterior, el proceso de compresión no es reversible. No se recuperará la información de forma exacta. La ventaja es que se consiguen Factores de compresión más elevados, y es posible que no exista una perdida apreciable (subjetiva) de calidad en la imagen descomprimida.

Lógica mente, existirá un compromiso entre el factor de compresión y la calidad de la imagen recuperada, puesto que son objetivos contrapuestos.

Mientras que en la codificación de ficheros de texto únicamente son válidos los métodos del primer grupo, en codificación de imagen y de voz los más frecuentes son los segundos. Sin embargo, como ejemplo de especial relevancia de compresión de imágenes sin error hay que mencionar la codificación de imágenes bitonales que se realiza en el fax, y los formatos .GIF y .TIF. En este libro nos centraremos principalmente en la compresión con error.

En el caso de imagen, la necesidad de utilizar compresión es todavía más relevante que en la voz, puesto que el volumen de información es mayor. Por ejemplo, una secuencia de imágenes en blanco y negro con calidad de televisión convencional requiere un tamaño de 512 x 512 pixels, 8 bits/pixel y 50 imágenes por segundo. Esto conduce a una tasa de bits de:

$$v_T = 512 \times 512 \text{ pixels} \times 8 \frac{\text{bits}}{\text{pixel}} \times 50 \frac{\text{imágenes}}{\text{segundo}} = 105 \text{ Mbits/segundo}$$

Si usáramos imágenes en color y/o una mayor resolución este valor sería mayor, requiriendo un gran ancho de banda en el caso de transmitir imágenes, o un gran espacio de almacenamiento en caso de pretender guardar la información para su posterior uso.

3.3.1.1. Aplicaciones

Las aplicaciones de la compresión de imagen pueden dividirse en dos grandes grupos:

Aplicaciones relacionadas con la transmisión de imágenes: Son aquellos casos en los que se pretende enviar imágenes de un emisor a un receptor ocupando el mínimo ancho de banda posible o minimizando el tiempo de transmisión. Entre otras cabe destacar:

Comunicaciones interpersonales: Por ejemplo, videoconferencia o fax. En el primer caso interesa ajustarse al ancho de banda disponible, mientras que en el segundo se pretende ocupar el mínimo tiempo posible de línea telefónica.

Imágenes vía Internet: Para reducir el tiempo de espera en la recepción, las imágenes están soportadas en formatos gráficos que incorporan compresión, como, por ejemplo, GIF, JPEG, etc.

Aplicaciones relacionadas con el almacenamiento:

Bases de datos gráficas: En determinadas aplicaciones es necesario guardar fotografías de clientes, imágenes médicas de pacientes, fotografías de pisos, planos, mapas, etc. En estos casos, dado que el soporte de almacenamiento es limitado, si se aplica una compresión a las imágenes previa a su almacenamiento, será posible archivar un número mayor de imágenes.

CD-ROM *multimedia:* Para almacenar películas, animaciones, enciclopedias etc., en soporte CD-ROM es habitual usar formatos de imagen que incorporan compresión, tales como JPEG, MPEG, etc.

Antes de analizar los principales algoritmos de compresión con error, es in acotar la máxima compresión conseguible sin error.

3.3.1.2. Máxima compresión conseguible sin error

Según el teorema de Shannon, es posible codificar, sin distorsión, una fuente pía H bits/símbolo usando H + e bits/símbolo, donde e es un número tan pequeño como se desee, siendo

$$H = -\sum_{i=0}^{L-1} P_i \log_2 P_i \quad \text{bits/símbolo}$$

Para una fuente de probabilidad de símbolo i, P, i = 0. .L—1 máxima compresión conseguible es:

$$C = \frac{\textit{tasa de bits media de los datos originales}}{\textit{tasa de bits media de los datos comprimidos}}$$

Sin embargo, calcular H es prácticamente imposible.

Ejemplo 3.3.1: Imagen de MxN pixels y B bits por pixel.

El número de imágenes posibles es: L = 2BMN

Si P es la probabilidad de la i-ésima imagen, se podría calcular la entropía, almacenar las L imágenes posibles y codificar mediante un índice cuál de las L.

Este es un método. Se usarían H bits por imagen (H/(MN) bits por pixel). Este es un método de cuantificación vectorial o codificación de bloque.

Ejemplo numérico: Suponiendo imágenes de 16x16 pixels con B = pixel resulta:

$$\left.\begin{array}{c} B=8 \\ M=N=16 \end{array}\right\} \Rightarrow L = 2^{2048} \approx 10^{614}\,!!! \quad \text{que es inviable.}$$

Una posible implementación práctica sería:

1. Dividir la imagen en bloques y aplicar cuantificación vectorial (VQ) de bloques 4x4 y B = 8, según los procedimientos descritos en el capítulo 1. l)e todos los bloques posibles se guardan sólo L' «

L de forma que sean más representativos posible de los bloques de imagen a codificar, puesto que el número de casos posibles sigue siendo muy elevado e inviable guardarlo todos:

2. De todos los bloques posibles se guardan sólo L`<< L de forma que sean lo más representativos posibles de los bloques de imagen de codificar, puesto que el número de los casos posibles siguen siendo muy elevados e inviable guardarlos todos:

$$\left.\begin{array}{c} B=8 \\ M=N=4 \end{array}\right\} \Rightarrow L=2^{4\cdot4\cdot8}=2^{128}\approx 10^{38}$$

Típicamente se guardan de 32 a 1024 bloques, con lo cual el índice necesario para indicar cuál de los bloques se ha seleccionado es un número de 5 a 10 bits.

La entropía de una imagen también se puede estimar a partir de la entropía condicional. Para un bloque de N pixels: u0, u1, u2, ..., UN_1.

$$H_N \cong -\sum_{u_0}\ldots\sum_{u_{N-1}} P(u_0 \mid u_1,\ldots,u_{N-1})\cdot \log_2 P(u_0 \mid u_1,\ldots,u_{N-1})$$

Si N — 00 HN converge a H. Sin embargo, cuanto mayor es N, más difícil es estimar las probabilidades condicionadas.

Es importante destacar que un método de codificación sin error nunca puede estar por debajo de H bits/pixel.

Para imágenes de tipo televisión a 8 bits/pixel las entropías de orden cero a segundo orden están en el margen 2-6 bits/pixel.

3.3.2. Codificación de pixel

Consiste en procesar cada pixel por separado, ignorando las redundancias existentes entre pixels cercanos. Por tanto, se trata de métodos sin memoria.

Las técnicas más utilizadas son PCM, cuantificación de contraste, dithering, codificación entrópica, RLE y codificación de bit plane. A continuación se describen sus características.

3.3.2.1. PCM

Consiste en muestrear, cuantificar y codificar la señal de vídeo, obteniendo una señal digitalizada. Lógicamente cuanto menos puntos (pixels) se tomen menor será el tamaño ocupado en memoria por la imagen. En contrapartida, la definición también será tanto menor y será más difícil apreciar los detalles de la imagen.

Por otra parte, el número de bits por pixel asignado en el proceso de cuantificación debe ser superior a seis en el caso de imágenes monocromas. En caso contrario, aparecería un efecto de “falsos contornos”. Si el cuantificador utilizado es no uniforme, puede reducirse a 5-6 bits/pixel sin notar una diferencia apreciable.

3.3.2.2. Cuantificación de contraste

Para reducir el número de bits de cuantificación, puede realizarse una transformación de luminancia a contraste y cuantificar el contraste con 6 bits y cuantificación uniforme o usar 4-5 bits si la cuantificación es no uniforme.

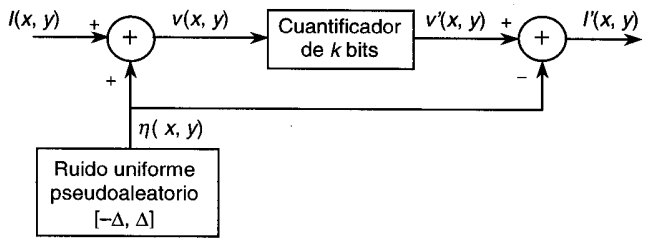


Figura 3.3.1. Diagrama de bloques del codificador de dithering.

3.3.2.3. Dithering

Para eliminar el efecto de los falsos contornos puede añadirse un ruido pseudoaleatorio a los pixels, antes de su cuantificación. Antes de visualizar la imagen habrá que restar el mismo (u otro) ruido.

El ruido debe tener una amplitud tal que afecte únicamente el bit menos significativo del cuantificador.

Usando dithering, pueden obtenerse imágenes de calidad con cuantificado- res de 3 bits.

El efecto producido por el ruido es aumentar o disminuir el nivel de gris original del pixel, de forma que los contornos quedan difuminados y no se aprecian líneas y contornos artificiales (no presentes en la imagen) que delimiten zonas con niveles de gris distintos (por ejemplo, el hombro de la figura 3.3.2c).

La figura 3.3.2 ilustra este fenómeno para cuantificaciones a 1 y a 3 bits.

Figura 3.3.2. Ejemplo de codificación con dithering. a) Codificación a 1 bit sin di thering; b) codificación a 1 bit con dithering; c) codificación a 3 bits sin dithering; d) codificación a 3 bits con dithering.

El ruido deberá ser aleatorio. Si no lo es y presenta una periodicidad marcada, se apreciará su efecto en la imagen resultante. La figura 3.3.3a presenta el resultado de usar como ruido de dither la señal de barras de la figura 3.3.3b. Obsérvese que las barras del dithering son visibles en la figura 3.3.3a.

a)  b)

Figura 3.3.3. Codificación con dithering de ruido no aleatorio. a) Codificación a 3 bits con dithering; b) patrón de ruido utilizado.

3.3.2.4. Codificación entrópica

Se trata de un método de compresión sin error. Si los pixels cuantificados no están distribuidos uniformemente, su entropía será menor a B y se pueden codificar mediante Huffman u otro método asignando códigos más cortos a los códigos más probables. De esta forma, es posible usar en promedio un número menor de bits por pixel que mediante un código de longitud fija. En (Held) puede encontrarse un estudio detallado de los códigos de longitud variable. Sin embargo, en imágenes de TV las probabilidades a largo plazo son iguales (distribución uniforme), aunque la estadística a corto plazo sea altamente variable. Por tanto, la codificación de entropía no es adecuada para imágenes PCM. En cambio, sí es útil en codificación predictiva y transformada, imágenes binarias (FAX, gráficos...), etc.

3.3.2.5. RLE (Run Length Encoding)

Se trata de un método de compresión sin error. Por ejemplo, en imágenes binarias, se puede codificar el número de ceros entre dos unos sucesivos. Es útil cuando hay largas secuencias de ceros.

Se suele trabajar con longitudes máximas de ceros de longitud M de forma que M = — 1, por lo que se necesitan m bits para codificar cada patrón con un código de longitud fija.

En el caso de imágenes multinivel, el RLE consigue eliminar la redundancia de repeticiones consecutivas de un mismo color o nivel de gris. El modelo más sencillo consiste en sustituir todas las repeticiones adyacentes por tres códigos:

- Un carácter especial que indica compresión RLE.
- El carácter que se comprime.
- Un contador que indica el número de repeticiones que se han eliminado.

Ejemplo 3.3.2:

Datos originales:	30	40	40	40	40	40	10
Datos comprimidos:	30	0	40	5	10		

Donde se ha supuesto que el O es el carácter especial que indica compresión. Por lo tanto, si la imagen está codificada a 8 bits por pixel, únicamente serían posibles los niveles de gris comprendidos en el margen [1, 255].

Lino de los modelos de compresión posibles de los formatos TIFF y PCX incorporan versiones del RLE.

Cuando no se dispone de ningún carácter especial que indique compresión, pueden usarse un esquema semejante a la compresión MNP5 que realizan los módems. Consiste en considerar tres repeticiones de un mismo carácter como código especial que indica compresión, de tal manera que cuando se detectan o más caracteres iguales, el siguiente carácter se interpreta como un contar con el número de repeticiones en exceso a las tres primeras. Un ejemplo es siguiente:

Datos originales:	30	30	30	40	40	40	40	40
Datos comprimidos:	30	30	30	0	40	40	40	2

Obsérvese que este método produce una expansión cuando sólo hay tres reparticiones de un mismo carácter, puesto que en este caso hay que añadir un contador a cero, y únicamente se obtiene compresión para cinco o más repeticiones.

3.3.2.6. Codificaciónes de bit plane

Una imagen de 256 niveles de gris se puede considerar como 8 bits planes (o mapas de bits) de 1 bit. Cada bit plane se puede codificar con un método de codificación de imágenes bitonales.
Se consiguen tasas de compresión de 1,5 a 2.

Este método es muy sensible a errores en el canal, a menos que se protejan los bits más significativos (MSB).

3.3.3. Codificación predictiva

Las imágenes habituales suelen presentar una evolución suave entre pixels consecutivos. Únicamente en el caso de los contornos existen variaciones bruscas. Por tanto, es posible conseguir compresiones mayores mediante métodos que eliminen la redundancia existente entre pixels consecutivos y codifiquen únicamente la información nueva.

Los principales métodos son DPCM, modulación delta y las técnicas adaptivas.

3.3.3.1. DPCM (Differential Pulse Code Modulation)

Forma una predicción del pixel a codificar en función de los pixels ya Codificados, y guarda el error de predicción (la parte no predecible es la información nueva) cuantificado (figura 3.3.4).

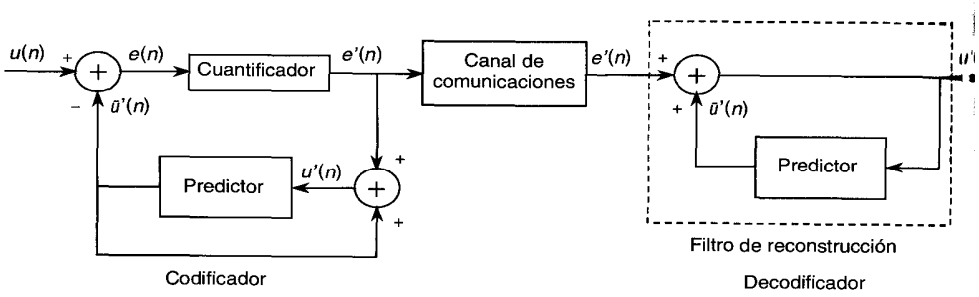


Figura 3.3.4. Codificador y decodificador DPCM.

Usualmente la predicción es buena y el margen dinámico del error de pre dicción es menor, por lo que se necesitan menos bits.

El predictor puede usar pixels de una línea (1D) o de más (2D).

Utilizando una estructura en lazo cerrado o DPCM (figura 3.3.4), el emisor (o codificador) contiene una réplica del receptor (o decodificador), de tal forma que trabaja sobre las mismas muestras que utilizará el receptor. De este modo, se evita la acumulación del error en la señal recuperada.

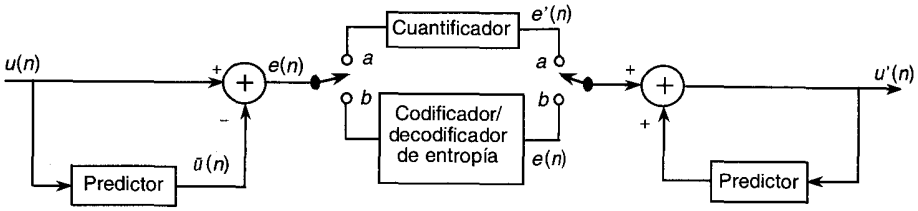


Figura 3.3.5. Codificador feedforward: (a) con distorsión; (b) sin distorsión.

Otra posibilidad es hacer trabajar al predictor sobre las muestras de entrada, no corrompidas por el error de cuantificación. Se conoce como codificación feedforward. La ventaja de este esquema es que permite una codificación sin error: si no se cuantifica el error de predicción (figura 3.3.5b), se recupera la información exactamente. En este caso, debe forzarse al predictor a dar valores enteros. La tasa de bits será la entropía del error de predicción.

3.3.3.2. Modulación delta

Es una simplificación del esquema de la figura 3.3.5. Utiliza un cuantifica1 bit y la predicción es el pixel anterior.

La elección del paso de cuantificación A es un compromiso entre los dos problemas clásicos de la modulación delta:

Saturación de pendiente

Si el paso de cuantificación es demasiado pequeño, la señal de salida no seguir a la entrada cuando ésta presenta una fuerte pendiente. Una solución consiste en filtrar en paso bajo la imagen de forma que se reduzcan las pendientes (contornos). El inconveniente es que la imagen resulta desenfocada.

Ruido granular

Si se aumenta el paso de cuantificación para evitar la saturación de la pendiente problemas en las zonas estacionarias de señal, puesto que la oscila entre dos niveles, proporcionando una salida ruidosa.

Una solución consiste en utilizar la modulación delta tri-state, consistente en utilizar tres posibles niveles de salida en vez de dos, con un cuantificador como la figura 3.3.7.

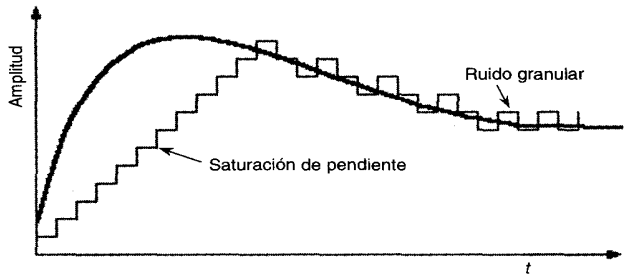


Figura 3.3.6. Modulación delta. Señales típicas de entrada y salida.

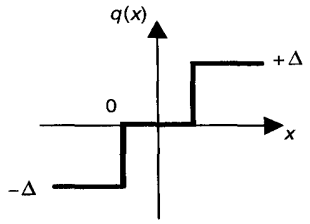


Figura 3.3.7. Cuantificador delta tri-state.

Puesto de pixels (del 65 al 85%) estarán en el nivel 0, se puede obtener una tasa de bits aproximada de 1 bit/pixel, utilizando una como, por ejemplo, la siguiente, o codificando los ceros con RLE (Run Length Encoding):

Nivel	*Codificación*
0	0
+Δ	10
−Δ	11

Obsérvese que para que los códigos sean descifrables de forma unívoca ninguno de ellos es prefijo de los otros.

Otra posibilidad es usar la modulación delta adaptativa de forma que el paso de cuantificación aumente al estar en una zona de pendiente elevada y disminuya en una zona homogénea. Para ello, será necesario evaluar la salida del cuantificador en dos o más instantes anteriores.

Un posible algoritmo de adaptación es el siguiente:

• Si salida(n) = salida(n—1), es posible que exista saturación de la pendiente y habrá que incrementar el paso de cuantificación = z = a.A.

• Si salida(n) salida(n—l), es posible que exista ruido granular y hahi1 que reducir el paso de cuantificación = z = b.A.

Dos valores típicos son a = 1,5 y b = 0,85.

3.3.3.3. Técnicas adaptativas

Se pueden conseguir mejores resultados adaptando cuantificador y/o predictor a la estadística local de la imagen. A continuación se presentan dos métodos para adaptar el cuantificador:

Control adaptativo de ganancia

Se adapta el cuantificador a señales de varianza unidad, se normaliza la señal de entrada a varianza unidad antes de cuantificarla (figura 3.3.8).
Es equivalente a adaptar el cuantificador pero más sencillo de implementar.

Clasificación adaptativa

Cada parte de la imagen se clasifica según su detalle espacial (o actividad), y se utilizan cuantificadores y/o predictores diferentes según la clase.

Por ejemplo, un clasificador puede ser la varianza de los pixels contiguos al pixel que se va a predecir. De esta forma, se pueden diseñar cuantificadores ajustados a zonas estacionarias, a zonas con transiciones abruptas, etc., y escoger en cada momento el más adecuado, siguiendo las indicaciones del medidor de actividad o clasificador.

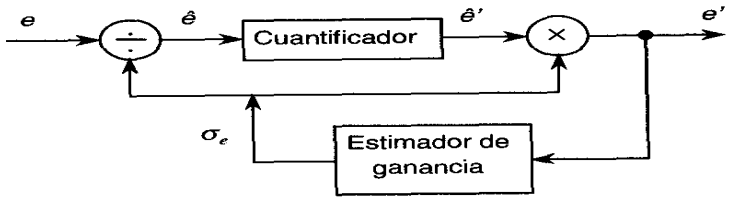


Figura 3.3.8. Control adaptativo de ganancia.

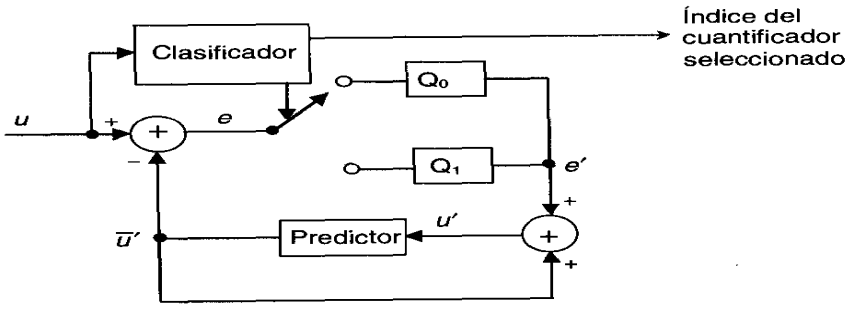


Figura 3.3.9. Clasificación adaptativa.

Mediante este método se explota el hecho de que el ruido es menos visible zonas con más actividad. La figura 3.3.9 muestra un ejemplo con dos clases. Normalmente, hay suficiente con cuatro

clases, lo cual supone 2 bits para indicar la clase. Esta información recibe el nombre de cabecera u "overhead", in la terminología inglesa.

3.3.4. Codificación transformada

Si bien es posible transformar una imagen tal y como se hizo en el capítulo mejora de imágenes, los codificadores por métodos transformados suelen trabajar sobre bloques más pequeños (típicamente 4x4 a 32x32 pixels).

La metodología es la siguiente (figura 3.3.10):

Proceso de compresión

1. Dividir la imagen en bloques y procesar cada bloque por separado.
2. Transformar el bloque a partir de las matrices de transformación escogidas (DCT, DFT, Hadamard, etc.).
3. Cuantificar los coeficientes transformados, asignando más bits a los más importantes. Generalmente, las varianzas de los coeficientes transformados son diferentes, por lo que se utiliza un número diferente de bits para cada coeficiente. Existe un algoritmo para asignar bits a los coeficientes transformados para una tasa de bits determinada, de forma que el resultado sea

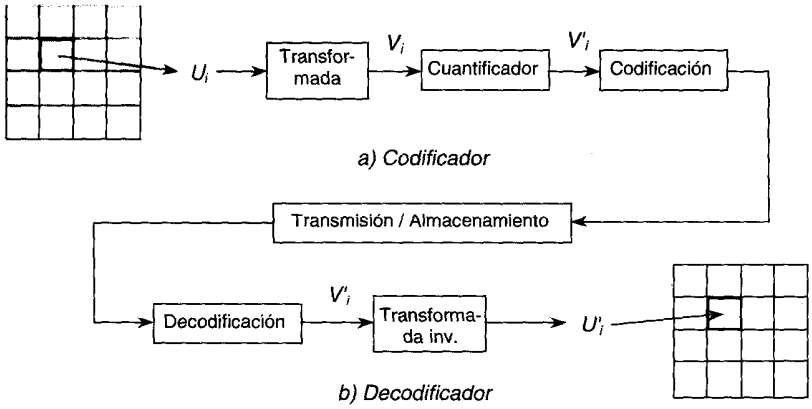


Figura 3.3.10. Codificación transformada bidimensional.

óptimo. La figura 3.3.11 muestra una posible asignación concreta de bits. Obsérvese que la mitad de los coeficientes no se codifican. Ello es debido a que contienen poca información, y el hecho de despreciarlos no supone introducir un error considerable. En vez de aplicar N cuantificadores con un número variable de bits (siguiendo la estructura de la figura 3.3.12), es posible realizar una cuantificación vectorial de todo el vector transformado de /V componentes o únicamente de un vector de dimensión N'<N (despreciando parte de los coeficientes, igual que en la estructura de la figura 3.3.12).

8	7	6	5	3	3	2	2	2	1	1	1	1	1	0	0
7	6	5	4	3	3	2	2	1	1	1	1	1	0	0	0
6	5	4	3	3	2	2	1	1	1	1	1	1	0	0	0
5	4	3	3	3	2	2	1	1	1	1	1	1	0	0	0
3	3	3	3	2	2	2	1	1	1	1	1	1	0	0	0
3	3	2	2	2	2	2	1	1	1	1	1	1	0	0	0
2	2	2	2	2	2	1	1	1	1	1	1	0	0	0	0
2	2	2	2	1	1	1	1	1	1	1	1	0	0	0	0
2	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figura 3.3.11. Asignación de bits.

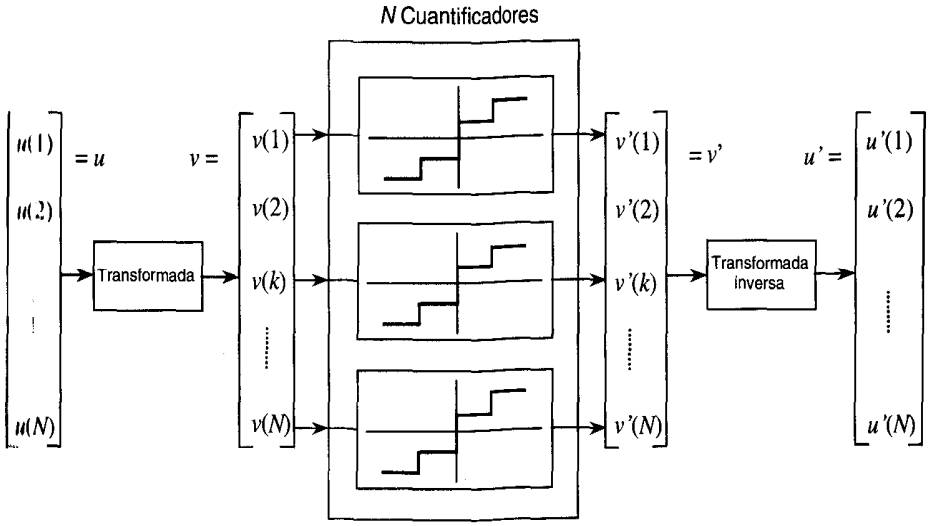


Figura 3.3.12. Codificación transformada unidimensional.

Se realiza la codificación, consistente en asignar a cada coeficiente el nivel de salida más cercano de su correspondiente cuantificador (el que proporciona menor error de cuantificación).

Proceso de descompresión

Debe realizarse la decodificación de los índices del cuantificador, para obtener los niveles de salida de cada coeficiente.

Será necesario restaurar la dimensión del vector transformado con los ceros necesarios hasta completar el tamaño original N.

Se realiza la transformación inversa, para recuperar el bloque descomprimido. Se repite el proceso con todos los bloques hasta completar la imagen descomprimida.

En cuanto al tamaño de los bloques, éstos pueden escogerse de forma unidimensional (1 xN o Nx 1) pero lo óptimo es tomarlos de forma cuadrada. Los valores más típicos son N=8x8 y N= 16xl6.

Se define el codificador transformado óptimo como aquel que minimiza la distorsión cuadrática media de los datos reproducidos. La transformación óptima es la Karhunen Loeve (KL). Sin embargo, es la más costosa de implementar.

3.3.4.1. Codificación zonal y codificación umbral

Para escoger los coeficientes transformados que deben codificarse, existen dos alternativas:

Codificación zonal

Se codifican únicamente los coeficientes que pertenecen a la "zona" a codificar, que es aquella en la que los coeficientes presentan mayor varianza. En la figura 3.3.13a se ha marcado con unos un posible conjunto de coeficientes a codificar.

Codificación umbral

Se comparan los coeficientes con un cierto umbral predeterminado previamente (figura 3.3.13b). Sólo se codifican y transmiten los que están por encima. 1 s un método adaptativo. Para un umbral fijo, el número de coeficientes elegido varía de una subimagen a otra. Por tanto, a diferencia del método anterior, la lasa de compresión es variable en función de las características particulares de cada imagen.

En codificación zonal es posible que se codifiquen coeficientes pequeños y se dejen fuera coeficientes importantes (si están situados fuera de la máscara).

En la codificación umbral se debe enviar de alguna forma la posición de los coeficientes seleccionados (o máscara umbral), lo cual implica aumentar la tasa de bits.

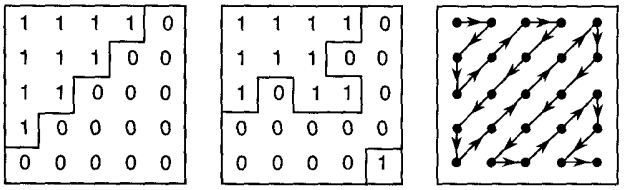


Figura 3.3.13. Máscaras zonal y umbral.

Para codificar la máscara umbral se realiza una ordenación (figura 3.3.1 3c) de forma que se obtiene un array unidimensional de ceros y unos. Este array se codifica mediante un método de compresión sin error.

3.3.4.2. Codificación transformada adaptativa

Existen tres posibilidades de adaptación correspondientes a distintas fases de la codificación transformada:

Adaptación de la transformada

Es costoso computacionalmente. En la KL se obtiene la matriz de transformación en función de los datos. Es óptima, pero para bloques grandes no hay diferencias significativas respecto a la DCT (para N> 64).

Adaptación de la asignación de bits

Se fijan varias clases según la actividad de la subimagen y se escoge una. Por tanto, la tasa de bits es distinta para cada bloque.

Adaptación del cuantificador

Se adaptan los niveles del cuantificador a la estadística de los coeficientes trasformados
Para relaciones seña/ruido (SNR) de 30 a 36 dB, se obtienen las compresiones de la tabla 3.3.1.

Tabla 3.3.1. Relaciones de compresión.

Método	Relaciones de compresión típicas para imágenes
Unidimensional	2-4
Bidimensional	4-8
Bidimensional adaptativa	8-16
Tridimensional	8-16
Tridimensional adaptativa	16-32

3.3.5. Codificación híbrida

Consiste en pasar al dominio transformado mediante una transformación frecuencial (DFT, DCT, etc.) y codificar los coeficientes transformados mediante codificación predictiva (típicamente DPCM).

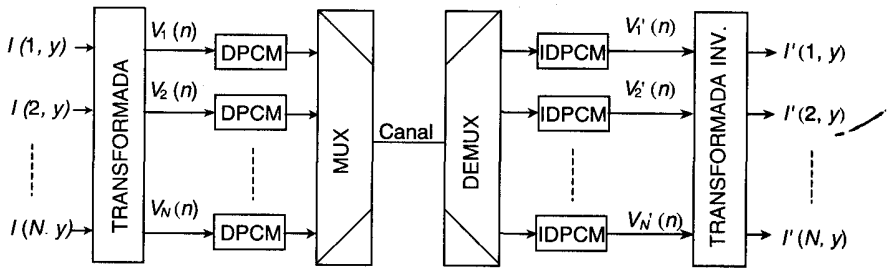


Figura 3.3.14. Método de codificación híbrida.

Una posibilidad es realizar una transformada unidimensional de las columnas de la imagen (I(i, y) i = 1,..., 1V) y codificar la secuencia de valores de un mismo coeficiente transformado, mediante DPCM en una dimensión (codificar, DPCM las filas de la imagen transformada) (figura 3.3.14).

Usualmente no se codifican todos los coeficientes transformados, ya que la mayor parte de la energía se compacta en los primeros, y los otros pueden set despreciados sin una pérdida apreciable de calidad. Lógicamente, de realizar la codificación se deberá restaurar el tamaño original de los vectores trasformados, añadiendo tantos ceros como coeficientes despreciados.

Dando que se utilizan métodos trasformados y productivos, se aplican los mismos criterios que en ellos, y es posible adaptar el predictor, cuantificador etc.

Es interesante destacar que si se hiciera primero DPCM y a continuación la transformación, empeoraría el resultado sustancialmente, puesto que el resultado de la predicción DPCM (la parte no predecible) tiene un comportamiento ruidoso (errático) con lo cual un método transformado no conseguiría compactar la energía en unos pocos coeficientes.

3.3.6. Codificación piramidal

La codificación piramidal de una imagen consiste en la siguiente metodología:

Si partimos de que go(i, j) es la imagen original, y le aplicamos un filtro de paso bajo (LPF) obtenemos gi(i, j). Teniendo en cuenta estas dos imágenes, podemos expresar el error de predicción L0(i, j) mediante:

$$L_0(i,j) = g_0(i,j) - \mathrm{LPF}\left[g_0(i,j)\right] = g_0(i,j) - g_1(i,j)$$

Codificaremos la imagen únicamente con L0 y g por dos motivos:

• gi(i, j) puede ser submuestreada ya que es una imagen de paso bajo. De esta forma, se reduce la densidad de muestras.

• L0(i, j) está concentrada en valores cercanos a cero y, consecuentemente, necesitaremos menos bits para representarla.

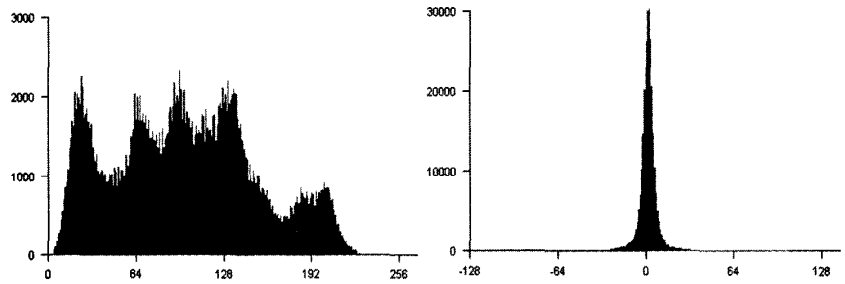


Figura 3.3.15. Histogramas de la imagen original (izquierda) y del primer nivel laplaciano L_0 (derecha).

Para poner de manifiesto esta idea, observemos los histogramas de la imagen original go y del primer nivel laplaciano L0 de la figura 3.3.15. En el histograma de la imagen original se observa que hay 256 niveles diferentes de gris, esto indica que para codificarla se precisan 8 bits; no obstante, el histograma correspondiente a L0 tiene dos características a destacar; en primer lugar, la varianza ha disminuido y, en segundo lugar, la distribución tiende a ser laplaciana, pudiéndose aplicar los códigos de Huffman para reducir aún más el número de bits por muestra. Por otra parte, podemos aumentar más la reducción si consideramos que es una imagen de alta frecuencia y el ojo es menos sensible a estas frecuencias. Por tanto, podemos usar menos bits para cuantificarla.
Esta misma idea se puede aplicar diversas veces, para codificar las imágenes respectivas, obteniéndose los diferentes niveles de la pirámide. De aquí el nombre del método.
Repitiendo este proceso obtenemos dos secuencias:

$$g_0,\ g_1,\ g_2,\ \ldots,\ g_{N-1}$$
$$L_0,\ L_1,\ L_2,\ \ldots,\ L_{N-1}$$

donde {g} es un conjunto de imágenes de paso bajo y {L} un conjunto de imágenes de paso alto. La estructura de datos {g } se denomina pirámide gausiana porque el filtro usado para su construcción es un filtro gausiano; igual- mente {L} recibe el nombre de pirámide laplaciana ya que cada nivel L se obtiene como la diferencia de dos gausianas y puede interpretarse como una función laplaciana.

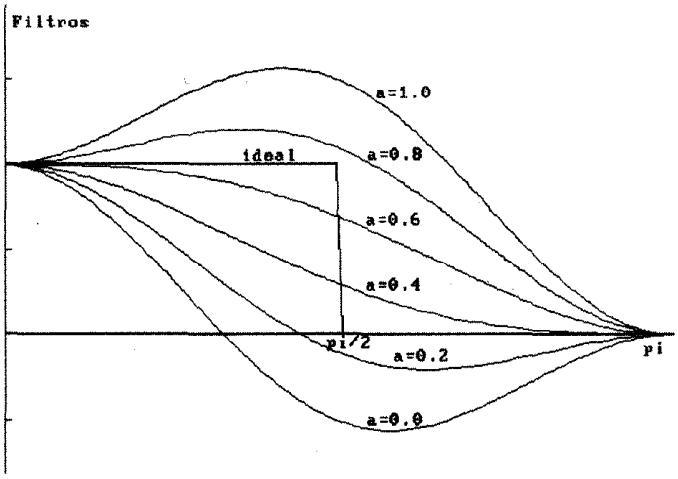


Figura 3.3.16. Filtro gausiano con **d = 2** *para diversos parámetros* **a.**

La figura 3.3.16 muestra diversos filtros de paso bajo, para un factor de diezmado d = 2. La expresión del filtro previo al diezmado es:

$$W(i,j) = W(i) \cdot W(j)$$

donde

$$W(i)=\begin{cases} a & i=0 \\ \frac{1}{4} & i=\pm 1 \\ \frac{1}{4}-\frac{a}{2} & i=\pm 2 \end{cases}$$

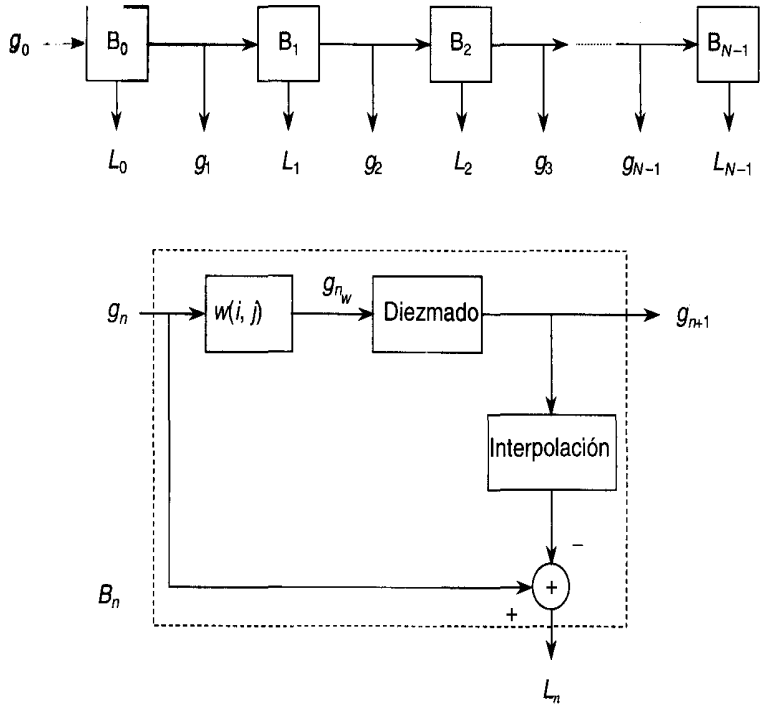


Figura 3.3.17. Diagrama de bloques de la generación de las pirámides gausiana y laplaciana.

Experimentalmente se obtiene que el valor óptimo del parámetro del filtro es a = 0,4.
La figura 3.3.17 representa el proceso iterativo necesario para obtener las secuencias de imágenes gausianas y laplacianas, donde el proceso realizado en cada nivel de la pirámide es el representado en el bloque B.

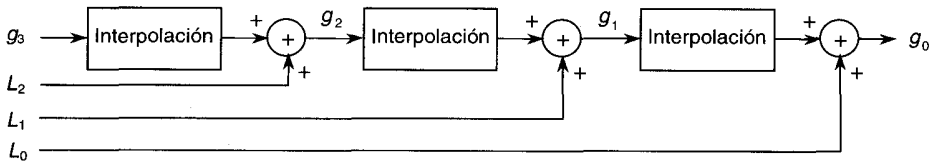


Figura 3.3.18. Diagrama de bloques para la recuperación de una imagen codificada de tres niveles.

Para recuperar la imagen descomprimida es necesario ir interpolando a partir del nivel más alto de la pirámide y sumando su correspondiente imagen de error laplaciana cuantificada. El proceso es el mostrado en la figura 3.3.18.

La figura 3.3.19 muestra las pirámides laplaciana y gausiana obtenidas en la codificación piramidal de la imagen Pamela con cinco niveles.

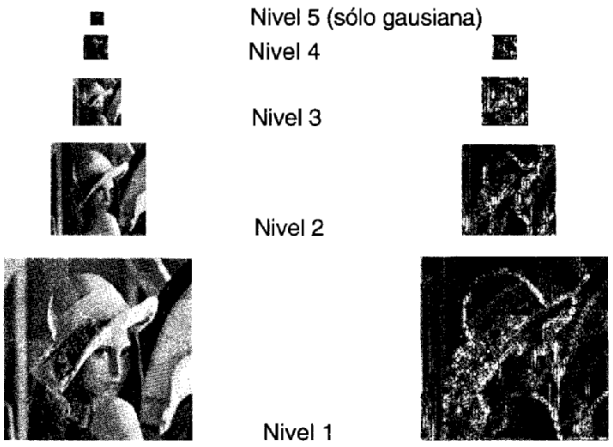


Figura 3.3.19. Pirámides gausiana (izquierda) y laplaciana (derecha).

Figura 3.3.20. Imagen* Pamela *original y codificada piramidalmente.

3.3.7. JPEG (Joint Photographlc Experis Group)

Se trata de un estándar internacional de codificación de imágenes fijas, que es el resultado del trabajo de un grupo de expertos en imágenes estáticas, denominado .JPEG. Fue aprobado en el año 1992 y es válido tanto para imágenes en tonos de gris como para imágenes en color.

Existen cuatro variantes de la codificación JPEG: básico/secuencial, sin pérdidas, progresivo y jerárquico. Sin embargo, el modo básico/secuencial es el más común.

Para imágenes en color y codificación con error, la relación tasa de bits ver- sus calidad es aproximadamente la siguiente:

— 0,25-0,5 bits/pixel: calidad moderada a buena. Suficiente para algunas aplicaciones.

— 0,5-0,75 bits/pixel: calidad buena a muy buena. Suficiente para muchas aplicaciones.

— 1,5-2 bits/pixel: usualmente indistinguible de la original. Suficiente para la mayor parte de las aplicaciones.

A continuación se describen las características principales de los diferentes modos de funcionamiento de JPEG.

Modo básico/secuencial

El proceso de codificación es el mostrado en la figura 3.3.21 y se puede resumir en los siguientes pasos:

• Se divide la imagen de entrada en subimágenes o bloques de 8x8 pixels.

• Se resta la componente continua (DC) del bloque y se cuantifica la diferencia de su valor respecto al término DC del bloque anterior. Por tanto, se realiza una codificación predictiva del término de continua de tipo

DPCM.

• Una vez eliminada la componente continua de cada bloque, se transforma mediante la DCT y se cuantifican los coeficientes transformados mediante un cuantificador escalar uniforme. Los pasos de cuantificación están definidos para cada uno de los 64 coeficientes en una matriz de cuantificación de 8x8. Típicamente se usa una matriz de cuantificación para la luminancia y otra para la crominancia, las cuales están definidas por defecto en el estándar. Sin embargo, en caso necesario es posible usar hasta cuatro tablas de cuantificación. Además, los valores de las tablas de cuantificación se codifican en la cabecera del fichero comprimido. El proceso de cuantificación es irreversible, y es el que proporciona la mayor parte de la compresión del fichero original. El proceso de cuantificación consiste en dividir cada uno de los componentes transformados por su coeficiente respectivo en la tabla, y redondear el resultado al entero más cercano. La tabla 3.3.2 es un ejemplo. Otra posibilidad más sencilla consiste en dividir todos los coeficientes por un mismo número N (por ejemplo, 4 u 8) y ajustar al entero más cercano.

Tabla 3.3.2. Ejemplo de tabla de cuantificación.

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

• Se ordenan los coeficientes transformados en zigzag, de forma que quedan ordenados de menor a mayor frecuencia, y se cuantifican con mayor precisión los coeficientes de baja frecuencia del bloque transformado. Esto es debido a que el ojo humano es más sensible a las bajas frecuencias. Una vez ordenados, se codifican mediante un código run length (RLE) consistente en dos campos: (longitud, valor), de los cuales la longitud indica el número de repeticiones consecutivas de un mismo carácter y el campo valor indica cuál es el carácter que se repite. Al final del bloque se envía el código (0,0).

• La salida del codificador RLE y el término DC se codifican mediante un código de longitud variable tipo Huffman, que asigna códigos más cortos a los valores más probables y más largos a los menos probables. De esta forma, se consigue reducir la longitud media del código. En cuanto a los valores de las tablas de Huffman, es posible usar las que establece por defecto el estándar o usar tablas específicas e incluirlas en la cabecera del fichero comprimido.

El proceso de decodificación consiste en deshacer los pasos realizados en h codificación en orden inverso.

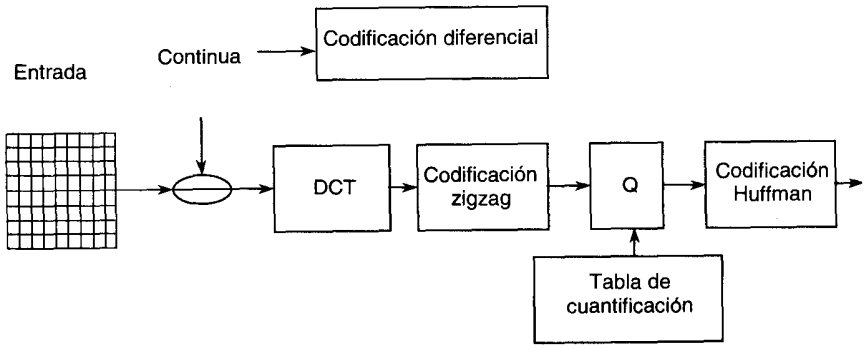


Figura 3.3.21. Diagrama de bloques de la codificación JPEG.

Método sin error (o sin pérdidas)

Para reducir la tasa de bits de la imagen original sin que aparezca error en el proceso de decodificación, se realiza un proceso predictivo. El tipo de predicción consiste en formar una combinación lineal de pixels vecinos ya codificados anteriormente. Para ello, existen siete posibles predictores, que son los mostrados en la figura 3.3.22.

Dado que únicamente se pueden usar pixels anteriores ya codificados, la primera fila de la imagen siempre se codifica con el predictor 2, y la primera columna con el predictor 1. En cuanto a los

resultados, se obtiene mayor calidad usando pixels de diferentes filas y columnas (predictores bidimensionales), como es el caso de los predictores 4, 5, 6 y 7. La diferencia entre la imagen original y la predicción es la parte no predecible que, para conseguir una compresión sin error, tendrá que ser codificada de forma exacta (sin introducir error). Sin embargo, si la predicción es buena, el margen dinámico del error de predicción y su distribución estadística son más adecuados para una codificación eficiente. De hecho, en el proceso predictivo se ha eliminado redundancia. El estándar JPEG en el modo sin error utiliza una codificación Huffman.
Típicamente este método consigue mayores tasas de compresión que otros métodos sin error tales como Lempel Ziv-Welch y Huffman por sí solos.

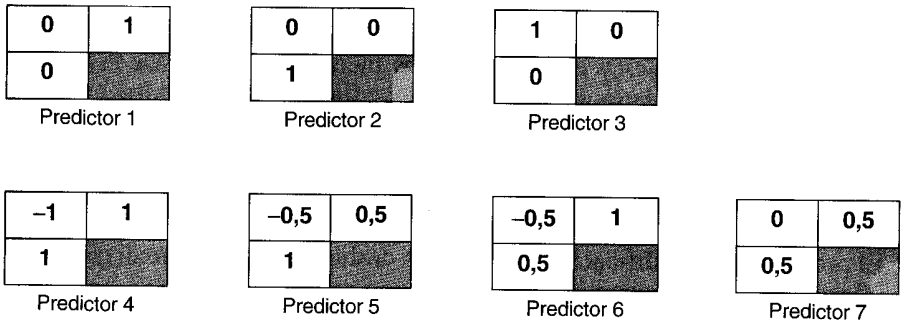


Figura 3.3.22. Esquemas de predicción (el pixel a predecir es el inferior derecho).

Modo progresivo

El objetivo de este modo es visualizar inicialmente la imagen en un modo de baja calidad, para ir incrementándola progresivamente. Una posible aplicación sería en las imágenes que aparecen en los webs. En vez de ir incrementando la porción visible de imagen conforme se van recibiendo más bits, resulta más agradable disponer rápidamente de toda la imagen, aunque sea con poca definición. A partir de la imagen inicial, y conforme se van recibiendo más bits, la calidad irá mejorando.
Existen dos formas para ir incrementando progresivamente la calidad:
Selección de coeficientes transformados: inicialmente se envía el término de continua y unos pocos coeficientes de alterna, y sucesivamente se va refinando la imagen enviando el resto de coeficientes de todos los bloques.

Aproximaciones sucesivas: Se envían inicialmente los bits más significativos de los coeficientes transformados (lo cual supone una aproximación burda a su valor), y sucesivamente se va refinando la información enviando el resto tic bits menos significativos, necesarios para conocer con mayor precisión los valores de los coeficientes transformados.

Modo jerárquico

Se trata de un algoritmo semejante a la codificación piramidal descrita en la sección 3.3.6. Se puede resumir en los siguientes puntos:
• Diezmar la imagen por un factor dos en cada dirección.
• Codificar la imagen resultante usando otro método (progresivo, secuencial o sin pérdidas).

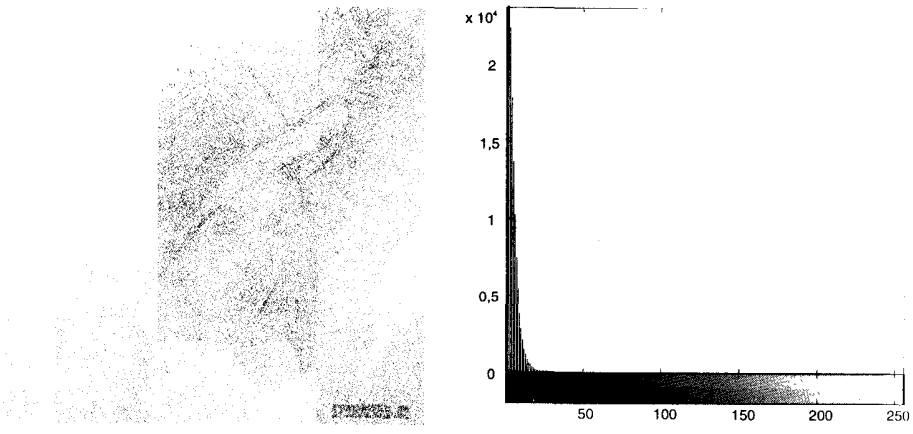
• Decodificar la imagen comprimida y restaurar el tamaño original, interpolándola por el factor de diezmado.
• Codificar la diferencia entre la imagen original y la interpolada en el punto anterior. Nuevamente se usará alguna de las otras variantes de JPEG.

Al igual que en la codificación piramidal, este proceso se puede repetir diversas veces, y presenta la ventaja de facilitar la visualización de imágenes de alta resolución en visualizadores de baja resolución.

En general, los programas comerciales, como por ejemplo Paint Shop Pro o MATLAB presentan la posibilidad de escoger entre varios de los métodos de codificación JPEG, así como un parámetro para controlar la tasa de compresión y. por tanto, la calidad de la imagen recuperada.

a) b)

Figura 3.3.23. Ejemplo de codificación JPEG. a) Imagen original; b) imagen decodificada JPEG (factor 50).



c) d)

Figura 3.3.23 (Continuación). c) negativo imagen error (×8); d) histograma imagen error.

La figura 3.3.23 muestra un ejemplo de imagen codificada JPEG con un programa comercial. En él se ha escogido como parámetro de compresión 50 (este parámetro es ajustable entre 1 y 100), y ha resultado en una compresión de 1:11 respecto a la misma imagen sin comprimir. Para apreciar el error

introducido en el proceso de descompresión, se incluye la imagen diferencia entre la original sin compresión y la descomprimida. A partir del histograma de la imagen diferencia se constata que el error es muy pequeño. Por ello se ha visualizado la imagen diferencia (imagen error) multiplicándola por 8. Por otra parte, se ha realizado el negativo, de tal forma que cuanto más negro es un pixel, mayor es su error.

3.3.8. Codificación de imágenes en color

En las imágenes en color se trata con tres colores primarios: rojo, verde y azul (R, G y B). Mediante ellos es posible formar un amplio margen de colores, en el proceso denominado mezcla aditiva. Por ejemplo, sumando dos de los colores primarios se obtienen los colores complementarios. Sumando rojo y verde se obtiene amarillo. Sumando azul y verde cyan. Azul más rojo produce magenta. Si se varía la proporción de cada uno de los tres colores primarios, se obtienen nuevos colores.

Por otra parte, se usa la luminancia o sensación lumínica. Se trata de un atributo psíquico que depende de la potencia y del matiz (o color). La luminancia de un punto depende de la luminancia de los tres colores que lo forman. Se representa mediante la letra Y y se obtiene mediante la expresión:

$$Y = 0{,}3R + 0{,}59G + 0{,}11B$$

En la televisión analógica en color convencional se envían las señales Y y diferencia de color Cb y C,, obtenidas a partir de:

$$C_b = B - Y$$
$$C_r = R - Y$$

Para reproducir una versión en blanco y negro de la imagen, bastará la información de la luminancia. Cada una de las imágenes se codifica mediante la 1 u luminancia.

Para codificar imágenes en color o imágenes multiespectrales, se pueden generalizar los métodos descritos en este capítulo. Para ello, son posibles diversas representaciones de la imagen en color, entre otras:

• Disponer de tres imágenes, una para el color rojo, otra para el verde y otra para el azul, de forma que la combinación de cantidades adecuadas de los tres colores permita representar un número muy elevado de colores.

• Disponer de la información de luminancia y crominancia al igual que en las imágenes de televisión.

• Disponer de un mapa de bits y un mapa de color. La forma de la codificación más habitual consiste en representar la imagen en un sistema de coordenadas adecuado (mediante una transformación de coordenadas) y representar cada componente por separado mediante un codificador específico.

En general, el proceso de codificación consistirá en comprimir cada uno de los parámetros que representan la imagen por separado, mediante cualquiera de los métodos descritos anteriormente, o codificarlos conjuntamente. Por ejemplo, se puede representar la imagen mediante un modelo RGB y codificar cada uno de los colores por separado, como si se tratara de tres imágenes distintas, o

codificarlas conjuntamente como si fueran una secuencia de tres imágenes (por ejemplo, formando cubos de MxNx3 pixels).

3.4. SECUENCIAS DE IMÁGENES

Cuando se dispone de dos o más imágenes de una misma escena, éstas constituyen una secuencia de imágenes. Es una situación semejante a los fotogramas (imágenes que integran una película). Para obtener sensación de movimiento continuo, se requieren unas 50 imágenes por segundo. Sin embargo, independientemente del número de imágenes por segundo, es posible aplicar a cada imagen por separado todas las técnicas descritas en los capítulos precedentes (técnicas intraframe) o realizar tratamientos teniendo en cuenta la dimensión temporal, usando dos o más imágenes (técnicas interframe).

Dado que los objetos presentes en una imagen experimentan habitualmente movimientos moderados, existirá una gran semejanza entre imágenes sucesivas (salvo en el caso de cambios de plano de la cámara). En aplicaciones de compresión, se puede aprovechar esta característica para reducir el total de información.

En caso de comprensión en una sola imagen, la información redundante la buscábamos en una imagen. Ahora, será posible buscarla entre imágenes consecutivas. La comprensión conseguible será tanto mayor cuanto menor sea el movimiento de los objetivos presentes en la imagen. Si el movimiento aumenta, disminuye la correlación temporal entre pixels en la misma posición especial, y resultan más adecuadas las técnicas intrframe (eliminar las redundacias respecto a los pixels vecinos especialmente).

3.4.1. Aplicaciones

Entre las aplicaciones se pueden citar: videoconferencia, vídeo digital, televisión de alta definición (HDTV), etc., y puesto que el problema de la compresión de imágenes suele implicar un problema de estimación de movimiento, el estudio también es válido para aplicaciones como: interpolación de imágenes, detección y seguimiento de objetos en movimiento, predicciones de movimiento, etc.

Entre las diferentes aplicaciones, las diferencias son sustanciales, tanto a nivel de requerimientos como a nivel de implementación. Por ejemplo, en una aplicación de videoconferencia, se trataría con imágenes más bien estáticas, con un fondo constante, y un orador que realizaría movimientos lentos, lo cual permitiría relajar los requerimientos. En cambio, en una aplicación de televisión de alta definición, el problema es mucho más amplio, puesto que abarca cualquier tipo de imágenes, movimientos rápidos, lentos, etc. Puesto que se trata de un medio de alta calidad, las especificaciones del problema serían mucho más restrictivas.

Evidentemente, al particularizar para una aplicación concreta, se podrán aplicar simplificaciones que reduzcan el problema, como es el caso del ejemplo

Ejemplo 3.4.1: Estimar la velocidad de un móvil

Para estimar la velocidad de un automóvil dada una secuencia de imágenes, bastaría estimar una sola componente, siendo irrelevante la velocidad en sentido perpendicular al carril.

A partir del desplazamiento estimado en pixels, el número de imágenes por segundo, y una relación de distancia en metros a pixels en pantalla (fácilmente conseguible con una referencia y calibración), se podría obtener la velocidad en km/h.

En otros casos, puede interesar conocer simplemente si ha habido movimiento o no, a grosso modo, lo cual permite realizar una estimación aproximada.

Ejemplo 3.4.2: Compresión de imágenes

Uno de los métodos más sencillos para comprimir una secuencia de imágenes es despreciando alguna de las imágenes en el transmisor ("frame skipping"). Por simplicidad, supongamos que se envía una de cada dos imágenes. Sin conocimiento de la trayectoria seguida por los pixels de la imagen S2KI a la S2K, la imagen perdida S2K se puede recuperar de dos formas:

Repitiendo la imagen anterior:

$$S_{2K} = S_{2K-1}$$

Mediante interpolación:

$$S_{2K} = \frac{1}{2}\{ S_{2K-1} + S_{2K+1} \}$$

Ambos métodos, presentan graves efectos en la calidad de reproducción del movimiento. En el primer caso, se aprecia un movimiento a saltos, mientras que en el segundo, se producen zonas borrosas (desenfocadas) en las áreas con movimiento.

Sin embargo, si se conoce el movimiento de los objetos presentes en la imagen puede realizarse una interpolación siguiendo el movimiento de los objetos. De esta forma, la calidad de la imagen interpolada es sensiblemente mayor.

Ejemplo 3.4.3: Televisión de alta definición

En HDMAC, se consigue un factor de compresión igual a cuatro mediante procedimientos de diezmado en el emisor, e interpolación en el receptor, de tipo espacio-temporal, para aprovechar las redundancias presentes en ambas dimensiones.

Dicho submuestreo debe adaptarse al contenido de la secuencia de televisión, mediante la información de movimiento presente en la misma. Si el movimiento (variación temporal) es poco significativo, predominará el diezmado en la dirección temporal (eliminación de campos o incluso imágenes completas); en caso contrario, se realizará un submuestreo espacial más severo. 1)e todo ello, resulta una codificación en tres modos:

• Modo estacionario o de alta definición:

Diezmado temporal 2: 1, se transmite una imagen de cada dos.

Diezmado espacial 2: 1 en cada imagen, submuestreo quincunx.

• Modo (le seguimiento:

Diezmado temporal 2:1, transmisión de un campo de cada imagen. Diezmado espacial 2:1 en cada campo o 4:1 en cada imagen.

• Modo de movimiento rápido:

No hay diezmado temporal (se transmiten muestras de todos los campos). Diezmado espacial 4: 1.

En cada uno de ellos, los tiempos invertidos en la transmisión de una imagen son, respectivamente: 80, 40 y 20 ms.

En las zonas con mayor movimiento, envía menor definición, porque el sistema visual humano presenta menor resolución a estas áreas.

La selección de los distintos modos se efectúa basándose en la información de un estimador de movimiento.

3.4.2. Estimación de movimiento

Existen diversos métodos para estimar el desplazamiento de los objetos presentes en una secuencia de imágenes. En este texto se estudiarán los métodos de block matching (BM) o ajuste de bloques.

Los métodos de block matching consisten en dividir la imagen en pequeños bloques no solapados de MxN pixels (típicamente 16x 16 u 8x8) y estimar el desplazamiento de cada uno de estos bloques, a los que llamaremos subimágenes, asumiendo que todos los pixels de un mismo bloque experimentan el mismo desplazamiento. Esta suposición será tanto más correcta cuantas más pequeñas sean la subimágenes consideradas, lo cual supone que el número de subimágenes de las que hay que calcular su desplazamiento es mayor.

En los métodos de block matching el error es grande si el desplazamiento no es constante en un bloque de pixels, lo cual sucede cuando la imagen contiene múltiples objetos en movimiento, objetos que se deforman, u otros movimientos complicados de objetos.

Se asumirá un modelo de movimiento puramente traslacional. Por tanto, no será posible modelar rotaciones, deformaciones y zoom de objetos, a menos que éstos sean divididos en bloques suficientemente pequeños.

El desplazamiento que experimenta una subimagen de un cuadro al siguiente, se puede representar mediante un vector, donde la dirección del mismo coincide con la dirección del movimiento, y el módulo con el número de pixels que se ha desplazado la subimagen en la dirección del movimiento.

La obtención del desplazamiento de una subimagen dada se obtiene buscando la posición de máxima coincidencia de la subimagen en cuestión dentro de la imagen siguiente. A menos que se utilice refinamiento de subpixel, se obtienen vectores de movimiento con una precisión de ±0,5 pixels.

Para encontrar el desplazamiento de un bloque S de MxN pixeis entre una imagen y la siguiente hay que fijar un desplazamiento máximo (dm), que nos define una zona de búsqueda de tamaño:

$$Z = (M + 2 \cdot d_m) \cdot (N + 2 \cdot d_m)$$

Para obtener el vector de desplazamiento hay que encontrar cuál es el desplazamiento que proporciona el error menor dentro del área de búsqueda Z, lo cual requiere la evaluación de un determinado criterio de error E entre (2dm+ 1) sibimágenes de la imagen K y la subimagen del cuadro K—l para cada vector desplazamiento (i, j):

$$E = \sum_{m=1}^{M} \sum_{n=1}^{N} f\left[S(m,n,t) - S(m+i, n+j, t-\tau)\right]$$

Realizar block matching consiste en desplazar la subimagen S de la imagen (K—1) por toda el área de búsqueda (figura 3.4.1) y escoger la posición de menor error. Es decir, buscar la máxima correlación entre ambas.

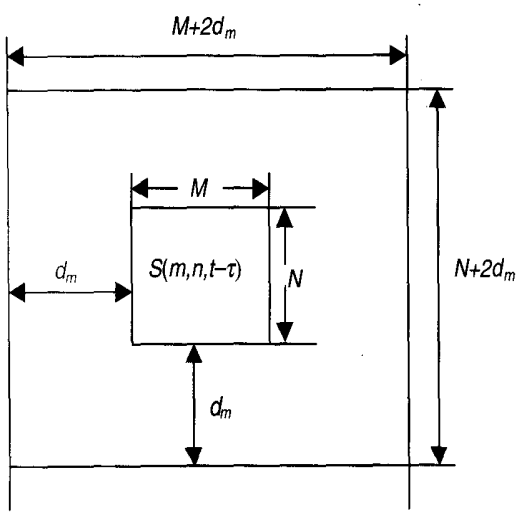


Figura 3.4.1. Área de búsqueda alrededor del bloque **S(m, n, t − τ).**

Esto supone un excesivo número de operaciones que incrementan el tiempo de cálculo. Cabe pensar en un aumento considerable de la rapidez mediante búsqueda inteligente.

Los algoritmos rápidos de block matching suponen una reducción del número de operaciones respecto a la búsqueda total (evaluar todos los posibles va res de (i, f) dentro del área de búsqueda), que es (2dm+ 1)2.

A continuación se describen dos algoritmos de BM: método three siel conjugado modificado.

Se consideran puntos de una misma iteración, aquellos que pueden ser evaluados en paralelo, ya que su ubicación no depende de los otros puntos tic misma iteración. En cambio, los puntos de una iteración tendrán posiciones dependientes del resultado de la iteración anterior, y deberán calcularse secuencialmente.

Como ejemplo, tomaremos subímágenes de 16x 16 pixels y un desplazamiento máximo dm6 pixels. El dibujo muestra los posibles candidatos a ser la dirección de mínima distorsión o diferencia (DMD) examinados en cada iteración marcados con un círculo y un número que indica el número de iteración. Cada punto de la rejilla, tiene coordenadas (x, y) que representan las componentes del vector de desplazamiento que estamos probando al estar en ese punto. El punto que nos hace mínimo el criterio de error, está sombreado. En la siguiente iteración, el proceso se repite pero con centro en este punto.

Este proceso se repite iterativamente, hasta obtener la DMD.

En caso de que el verdadero desplazamiento esté fuera de la zona de búsqueda, no podremos llegar a evaluarlo correctamente, pero sí el resultado que obtendremos, que estará sobre los límites de la

zona de búsqueda e indicará la dirección correcta del movimiento (no el módulo), a menos que los dos cuadros sean totalmente incorrelacionados, y no exista DMD ni dentro ni fuera del área de búsqueda.

3.4.2.1. Método three step

El nombre del método se debe a que la estimación de movimiento se hace en tres iteraciones. Para estimar un desplazamiento máximo de 6 pixels en las dos direcciones, se evalúan los puntos (O, O), (O, m), (m, O), (m, m), (—m, —m), (O, —m), (—m, O), (—m, m) y (m, —m) referidos al vector obtenido en la iteración anterior; m toma los valores sucesivos 3, 2, 1 en las iteraciones 1, 2 y 3 respectivamente.

Si no se abandona la búsqueda cuando se encuentra un punto con error inferior a un cierto umbral prefijado o igual a cero (caso que en imágenes reales, en general, no sucederá), el método three step siempre evalúa el mismo número de puntos, lo cual supone que una de sus características es la independencia entre tiempo de cálculo y tipo de imagen examinada. En los otros métodos de block matching, esta característica no estará presente.

Si el desplazamiento real fuera el vector (2,6), las aproximaciones sucesivas a la estimación y los puntos examinados serían los mostrados en la figura 3.4.2.

En general, el método three step requiere la evaluación de más puntos que la mayoría de métodos de block matching, pero estima el movimiento en tan sólo tres iteraciones, mientras que los otros métodos requieren más. Esta ventaja del método three step, en cuanto a número de iteraciones, lo hace especialmente indicado para ser implementado en hardware en un sistema en tiempo real, con una complejidad menor, ya que todos los puntos de una iteración pueden ser calculados en paralelo (son cálculos independientes) y, en este caso, es más importante el número de iteraciones que el número de puntos a examinar. Para desplazamientos máximos diferentes de 6, el algoritmo debe ser modificado, manteniendo su filosofía:

• La suma de los incrementos de cada iteración debe ser igual al desplazamiento máximo.

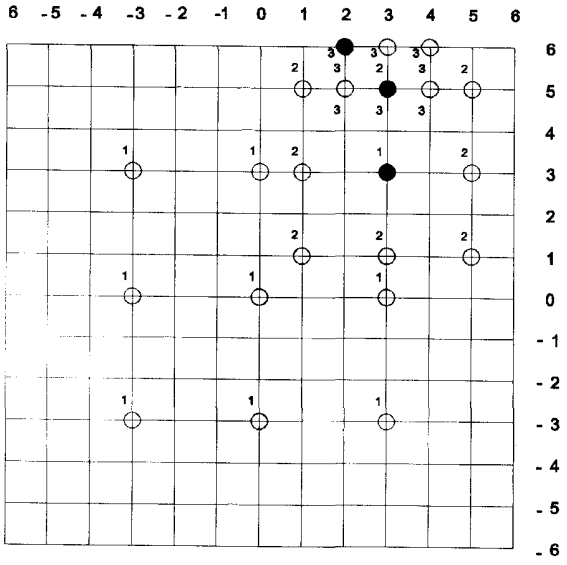


Figura 3.4.2. Método three step para reconocer el vector (2, 6). Encontramos pr viamente los vectores (3, 3) y (3, 5).

• Mediante combinaciones de los diferentes incrementos, debe ser posible obtener todos los números enteros entre 1 y el desplazamiento máximo (d,

Por ejemplo, para un desplazamiento máximo de 15, serían adecuadas cuatro iteraciones, con los incrementos 8, 4, 2, 1.

Además, el número de puntos evaluados n es:

$$n = 1 + 8 * (\text{número de iteraciones})$$

3.4.2.2. Método conjugado modificado

Los requerimientos deseables para un buen algoritmo de block matching son:

1. **Convergencia**. Que converja al punto óptimo en un número finito de pasos para una zona finita de búsqueda.
2. **Pocos puntos de búsqueda**. El número total de puntos de búsqueda necesarios para encontrar el óptimo debe ser tan pequeño como sea posible. l característica puede ser evaluada en el peor caso (que es lo más comúnmente usado en las publicaciones), o en el caso promedio, lo cual es más fácil de analizar, pero se puede simular con una imagen en la que no pre mine ningún tipo de movimiento en especial.
3. Repercute directamente en el tiempo de cálculo, en el caso de una implementación en soltware.
4. **Pocas iteraciones de búsqueda.** El número total de pasos para encontrar el punto óptimo debe ser ll más pequeño posible.
5. **Inmunidad respecto al ruido**. La convergencia debe ser insensible al ruido en los datos.

Además de los criterios expuestos, hay otros aspectos importantes como, por ejemplo, la complejidad del hardware. Un algoritmo que tenga un número pequeño de búsqueda, necesitará una complicada estructura de decisión que requiera más hardware que una estructura regular con más puntos.

Evidentemente, ningún algoritmo podrá cumplir todos los requerimientos simultáneamente, debido a los inherentes conflictos que hay entre ellos. El método conjugado modificado ofrece unas buenas prestaciones.

Para el caso de desplazamiento máximo igual a 6 se obtiene un buen compromiso con una separación de 2 pixels entre puntos examinados.

La búsqueda se inicia evaluando el criterio de error en los puntos (O, O), (2, 0) y (—2, 0). A partir de aquí pueden suceder tres casos:

- Que el punto central sea el mínimo. En este caso, se reduce el incremento a uno y se busca el mínimo. A partir de él se realiza la búsqueda en el eje y con el mismo criterio del eje x, buscando los puntos (0, 2) y (0, —2) referidos al mínimo encontrado en la dirección x. El punto (O, O) no hace falta calcularlo, puesto que ya se obtuvo en la iteración anterior.
- Que el punto de la derecha presente mínimo error. Se continúa añadiendo puntos por la derecha separados 2 pixels hasta que el central dé mínimo error, o hasta llegar a un desplazamiento de 6. En el primer caso, se procede como en el caso anterior, y en el segundo sólo se examina el punto inmediato a la izquierda (para no salirse del área de búsqueda), escogiendo el de mínimo error, y conmutando a la dirección y a partir de ese punto.
- Que el punto de la izquierda dé mínimo error. En este caso se procede corno en el caso anterior, pero añadiendo puntos por la izquierda, y al llegar a -6 se observa el inmediato a la derecha.
- El número de iteraciones necesarias para estimar el desplazamiento varía entre 4 y 8, mientras que el número de puntos examinados está comprendido entre 9 y 13 (el mínimo se reduce a uno en ambos casos si se abandona la búsqueda al encontrar un vector con error nulo o inferior a un cierto umbral prefijado).

En simulaciones prácticas se ha obtenido que el método conjugado modificado es sensiblemente más rápido que los otros métodos de block matching, y tarda casi la mitad que el algoritmo three step.

Gráficamente, los puntos examinados para estimar el vector (2, 6) son los de la figura 3.4.3.

Puede pensarse en reducir todavía más el número de puntos a examinar, cuando el mínimo lo obtenemos en el punto central, mediante el siguiente razonamiento: cuando el mínimo lo obtenemos en el punto central, no hace falta examinar los dos puntos inmediatos adyacentes al central, sino sólo el comprendido entre el central y el que nos dé mínimo error de (—2, 0), (2, 0) (relativos al central y cuando se minimiza en el eje x).

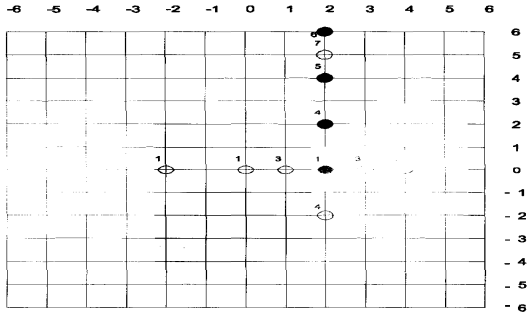


Figura 3.4.3. Método conjugado modificado

Otra posible modificación, es iterar primero conocimientos de 2 pixels hasta obtener mínimo en ambas direcciones, y a continuación reducir el incremento a uno, y ajustar en los ejes x e y Esta

modificación, sin suponer ningún aumento del tiempo de cálculo, permite una mayor libertad respecto al método discreto en principio y, pese a que en muchos casos el resultado será el mismo, estadísticamente el resultado será mejor, puesto que el refinamiento se realiza Final, cuando ya se está sobre la dirección correcta.

Dado que el número de puntos examinados varía poco de una subimagen otra (está comprendido entre 9 y 13), el tiempo de cálculo es aproximadamente mismo independientemente de si el movimiento presente en la imagen es evado o moderado. Esta es otra buena característica del método conjugado modificado.

FI número de puntos a examinar, n, se puede expresar en función del desplazamiento máximo dm como:

$$n = 5 + dm$$

3.4.3. Predicción

En codificación de imagen es posible formar una predicción de la imagen siguiente a partir de la actual, de tal forma que únicamente sea necesario enviar o codificar el error de predicción.

La predicción más sencilla a utilizar es la imagen anterior directamente; sin embargo, es de esperar que se obtenga una mejor predicción compensando el movimiento mediante los métodos a estudio. En este caso, la información a muir serán los vectores de movimiento de cada subimagen y el error.

CAPITULO 4: IMÁGENES MÉDICAS: ADQUISICIÓN, INSTRUMENTACIÓN Y GESTIÓN

4.1 Introducción

La **Ingeniería Biomédica** reúne técnicas y métodos de ingeniería con las ciencias biológicas y la medicina para tender hacia una mejora de la calidad de vida y de atención de la salud.
Existen dos preocupaciones fundamentales:

- entender los **fenómenos biológicos** (modelos, análisis, experimentos)
- **desarrollo de dispositivos y programas** (métodos, algoritmos, materiales, estructuras teóricas)

El resultado debe ser medido en términos de **eficacia de la provisión de cuidados clínicos** y en el aumento del conocimiento.

Imágenes médicas

Ejemplos

- Placa de rayos X
- Tomografía Computada
- Ecografía obstétrica
- Resonancia magnética
- Franjas Moiré
- Tomografía por emisión de positrones

Finalidad de las imágenes

- Estructura interna a evaluar (huesos)
- Función vital a estudiar
- Documentación de situación biológica
- Resumen visual de información compleja
- Integración de imágenes diferentes
- Ayuda para acciones (cirujía, cateterismo, etc.)

Particularidades respecto a otras imágenes

- Dificultad de acceso al objeto
- Variabilidad biológica
- Evolución en el tiempo
- Necesidad de interpretación experta
- Complejidad
- Preservar la privacidad del paciente
- Gravedad de consecuencias de artefactos
- Noción de la diversidad de principios físicos que permiten construir y obtener imágenes para uso médico
- Manejar los principios de algunas modalidades de imágenes
- Conocer los elementos constitutivos de la instrumentación
- Adquirir práctica de manejo de imágenes en ámbito telemático (DICOM)
- Proyecto de interconexión de equipos de imágenes y redes para uso en Hospitales

- Precauciones para pacientes y operadores, blindajes, envergadura y dimensiones del mantenimiento
- Todos estos conceptos cuantificados (tarea del ingeniero)

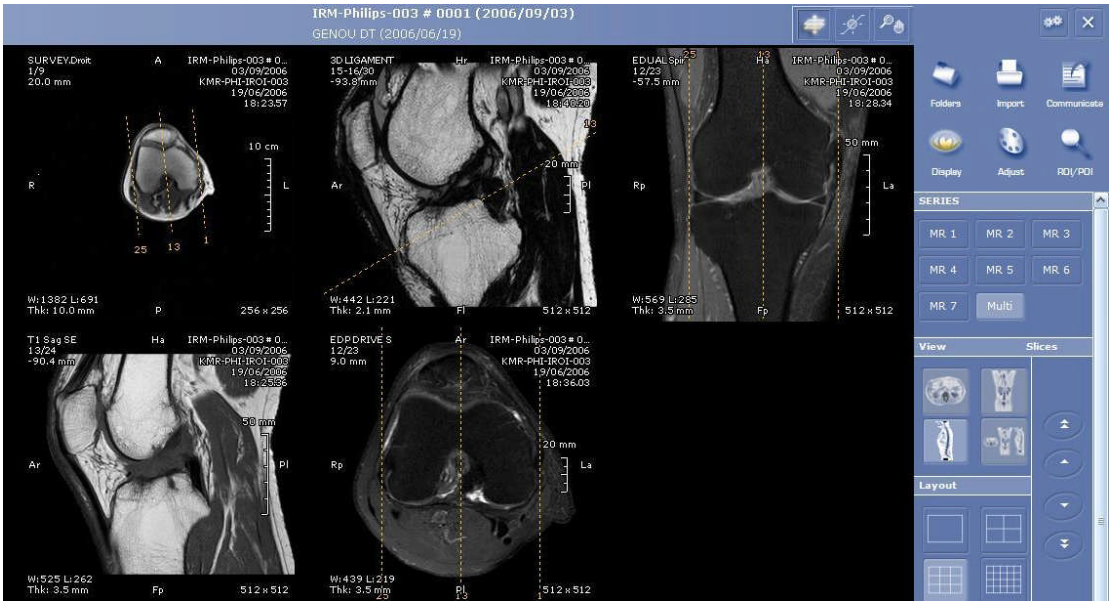


Figura 4.1 Ejemplo de imagen médica: rodilla por RM

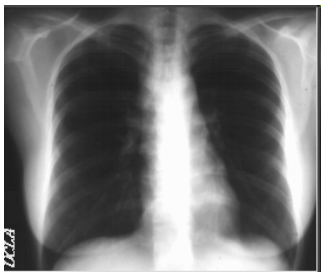


Figura 4.2 Placa de rayos X

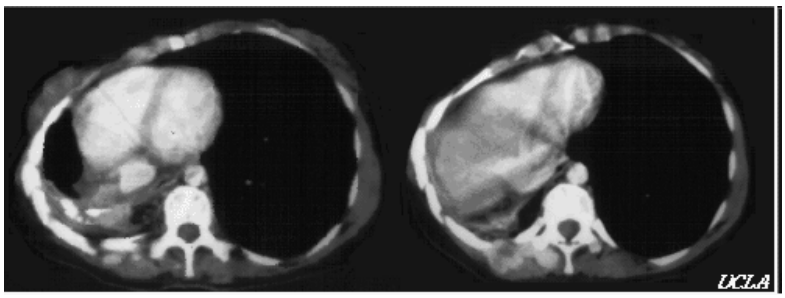


Figura 4.3 Tomografía computada

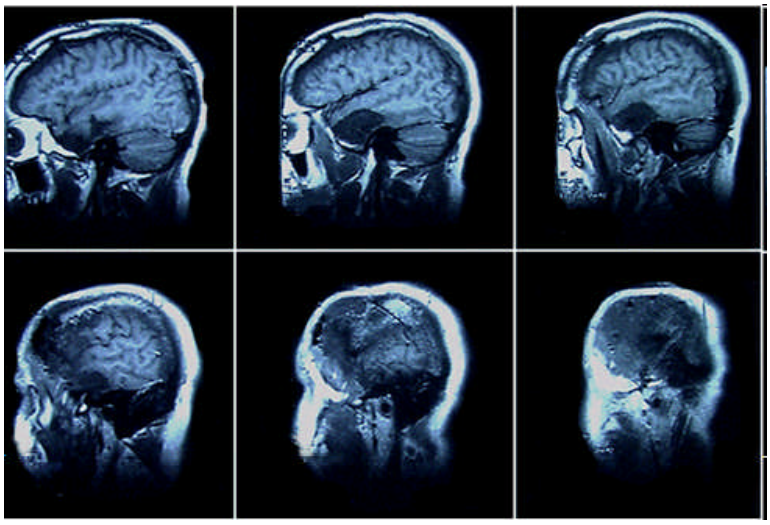


Figura 4.4 Resonancia magnética

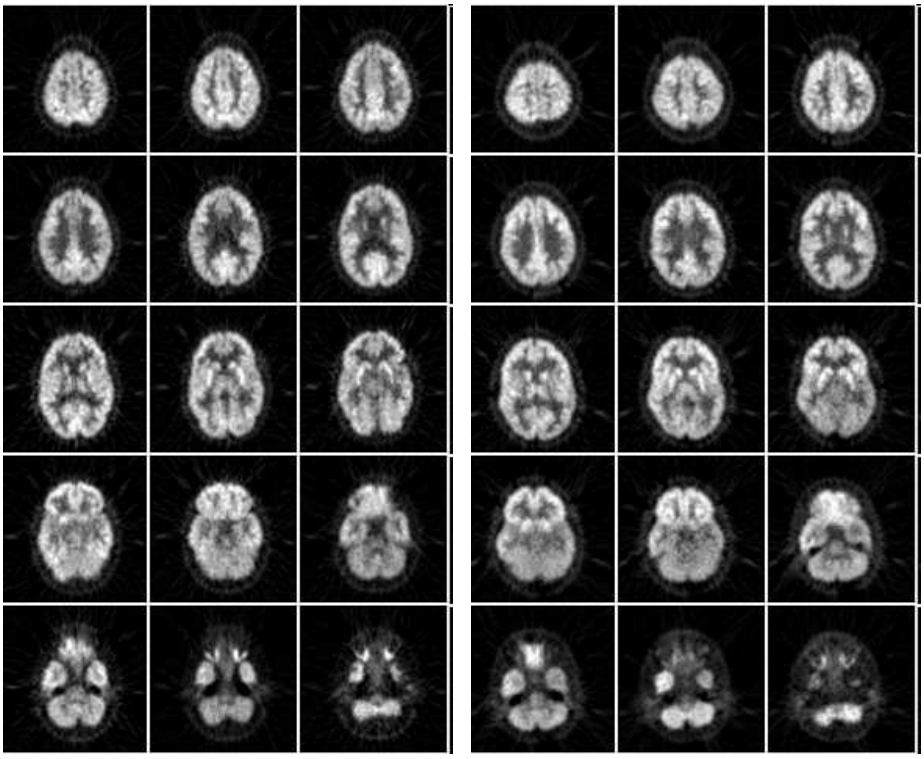


Figura 4.5 Placa de rayos X

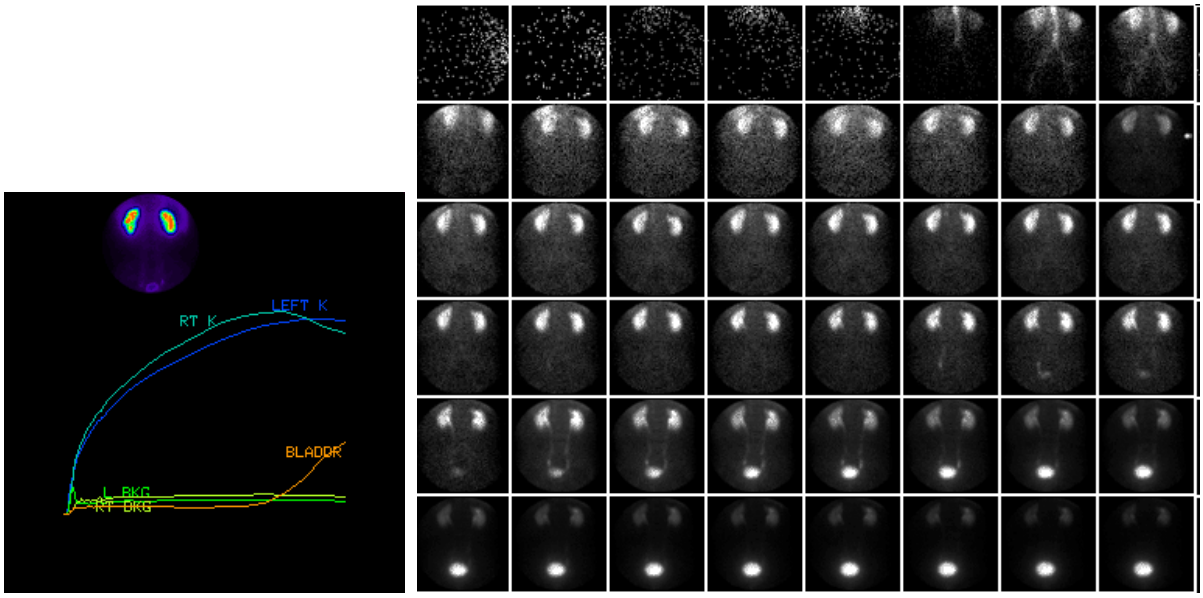


Figura 3.6 Estudio de Medicina Nuclear – riñón

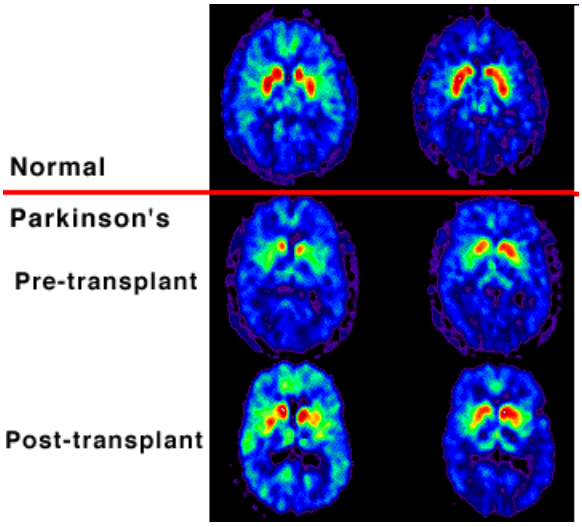


Figura 4.7 Evolución de radioactividad en una Región de Interés (ROI) seleccionada por el operador en las imágenes

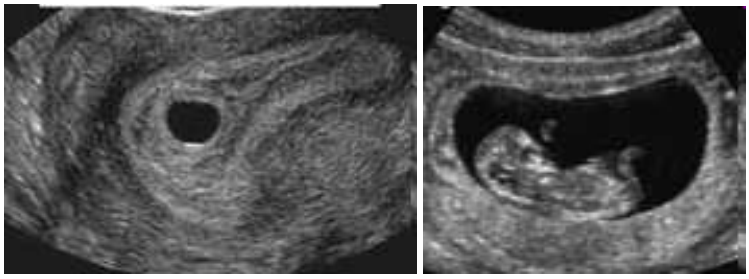


Figura 4.8 Ecografía obstétrica

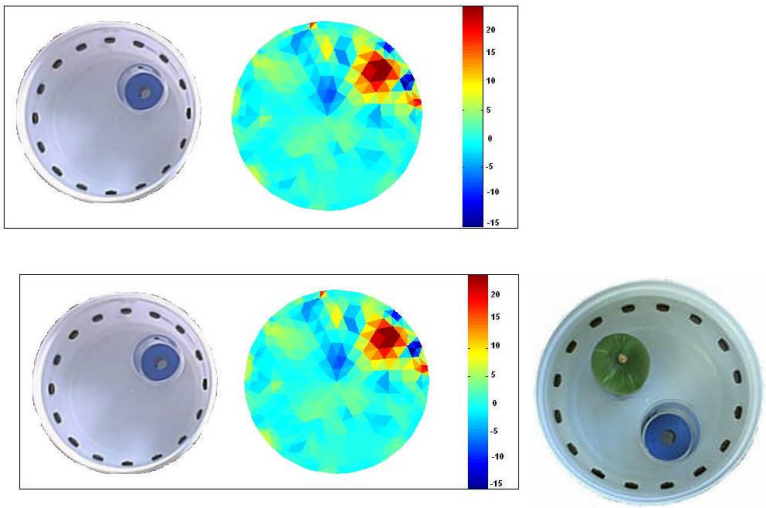


Figura 4.9 Tomografía de impedancia eléctrica

El procesamiento digital de imágenes incluye un conjunto de técnicas que operan sobre la representación digital de una imagen, con el objeto de destacar algunos de los elementos que conforman la escena, de modo que se facilite su posterior análisis, ya sea por parte de un usuario (humano) o un sistema de visión computarizado. En general, las técnicas de procesamiento de imágenes son aplicadas cuando resulta necesario realzar o modificar una imagen para mejorar su apariencia o para destacar algún aspecto de la información contenida en la misma, o cuando se requiere, medir, contrastar o clasificar algún elemento de la misma imagen. También se utilizan técnicas de procesamiento, cuando se requiere combinar imágenes o porciones de las mismas o reorganizar su contenido.

La imagenología médica (conocida así en la labor hospitalaria), por su parte, considera un conjunto de modalidades de adquisición de imágenes médicas, las cuales se diferencian en cuanto a la naturaleza de los principios físicos involucrados en el proceso de adquisición. Adicionalmente existen también diferencias en cuanto a la aplicación médica. Las modalidades más comunes de imagenología médica son los rayos *X*, la tomografía computarizada, la resonancia magnética nuclear, la imagenología nuclear, la imagenología por ultrasonidos, etc. Este campo de las imágenes médicas ha avanzado significativamente en los últimos años, convirtiéndose en una herramienta primordial en la práctica de la medicina y generando, a su vez, nuevas líneas de investigación científica que abarcan múltiples disciplinas. Esta evolución ha sido posible gracias a la difusión de áreas tradicionalmente separadas pero muy relacionadas: desde la visión por computadora y el procesamiento de imágenes, pasando por técnicas gráficas y visualización, hasta los entornos inmersivos e interactivos y dispositivos para la manipulación interactiva de la información. Como resultado de estas innovaciones, las imágenes médicas se están reinventando a si mismas continuamente, fundamentalmente en el modo en que son vistas, en cómo son comunicadas y en la forma de interactuar con la información médica.

4.2. Instrumentación de radiología analógica y digital

4.2.1. Detección de rayos X

Introducción

- Contamos con una fuente que emite RX (tubo de rayox x).
- Sabemos que existen propiedades en la materia que atenúan dichos rayos x en forma diferente según:
    - Su número atómico.
    - Su espesor.
    - Su densidad.
    - Veamos como detectar dichos rayos x atenuados y transformarlos en una imagen en una placa o en un monitor.

- Necesidad de obtener imágenes tanto estáticas (en placa o en un monitor) e imágenes dinámicas (secuencias de video que se visualizan en un monitor).
- Históricamente ambas detecciones eran analógicas (placas reveladas o secuencias de video tomadas con cámaras analógicas).
- Hoy en día se están popularizando las técnicas de detección digitales, teniendo ambos tipos de capturas en un monitor. El mundo digital ofrece múltiples ventajas como veremos mas adelante.

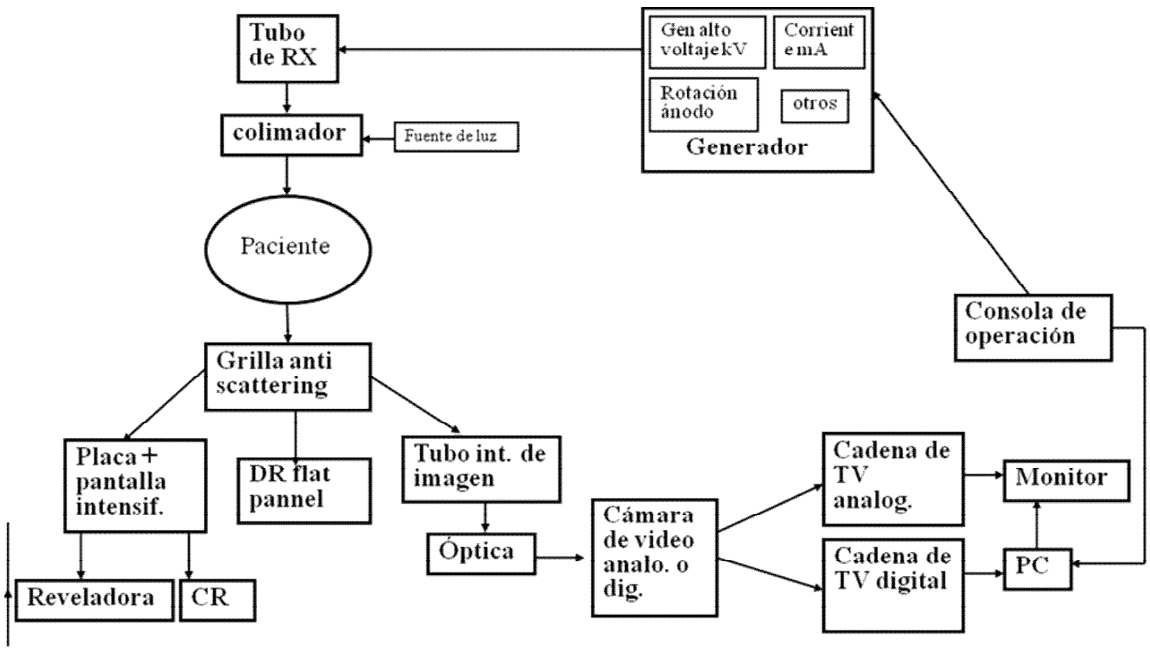


Figura 4.1.1 diagrama de bloques de un equipo de rayos x equipado con computadora

4.2.2. Captura de imágenes estáticas

- El método mas utilizado históricamente ha sido la placa.
- Se trata de proyectar los rayos x absorbidos por el paciente en una placa fotosensible (película, film).
- Luego dicha placa es revelada utilizando productos químicos similar al negativo de una cámara de fotos.

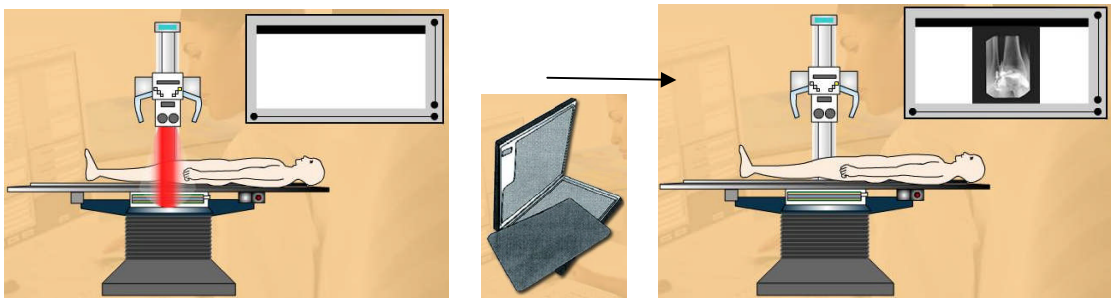


Figura 4.1.2.1 Captura de imágenes estáticas

Pantalla intensificadora

- Por si solo los film serían capaces de detectar los rayos x e imprimirlos en la placa.
- Serían necesarias grandes cantidades de rayos x para producir una imagen con resolución suficiente.
- Para mejorar esto se utilizan pantallas intensificadoras (screen) colocadas en las paredes de un “cassette” donde se coloca la placa.

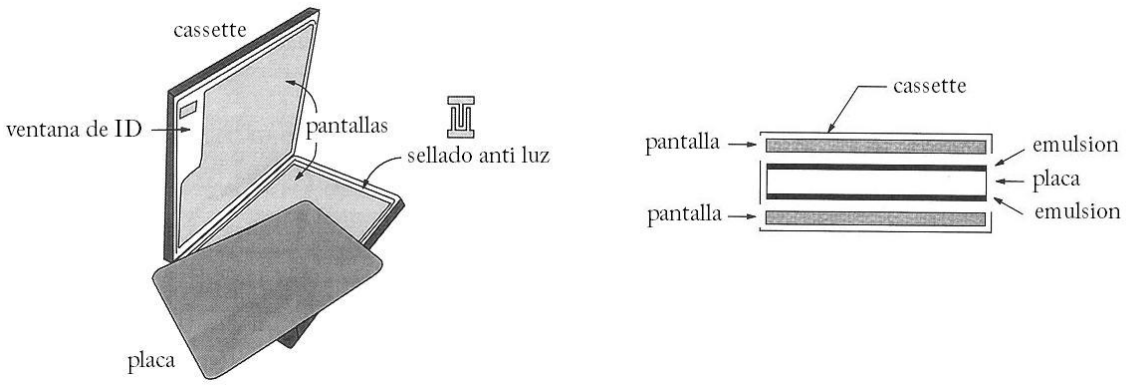


Figura 4.1.3 Pantalla intensificadora

- Fabricadas de un material centellante.
- Emiten fotones de luz al ser golpeadas por los rayos x. Esta luz aumenta muchísimo la eficiencia de la placa (los films son mas sensibles a estas long de onda) y la imagen es impresa con mayor claridad con mínima radiación.

- Existen 2 tipos de materiales utilizados para fabricar las pantallas:
    - Tungstato de calcio (CaWO4).
    - Tierras raras: Gd2O2S, LaOBr, YTaO4, etc.
- Como vemos es muy importante el apareo pantalla-placa.

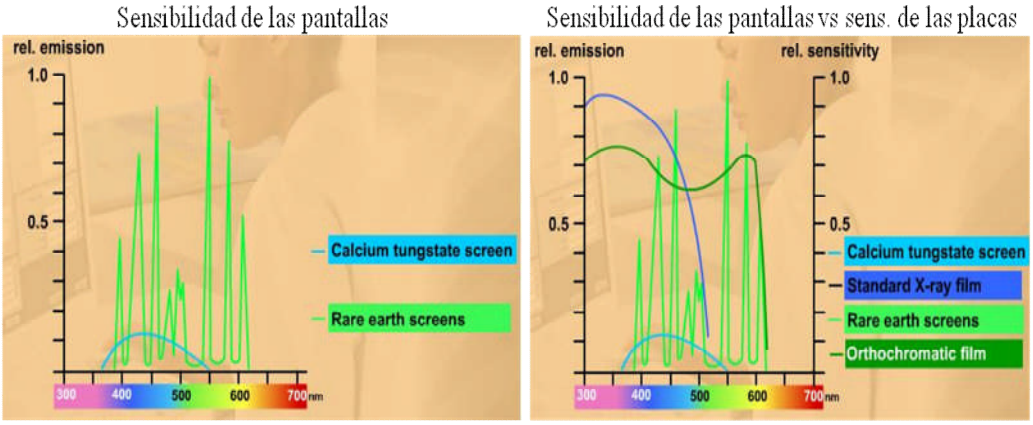


Figura 4.1.4 Curvas de sensibilidad de pantalla

4.2.3 Captura de imágenes dinámicas

- En ciertas aplicaciones es necesario obtener imágenes en movimiento.
- Dependiendo de la aplicación, son necesarios sistemas de TV con una tasa de entre 25 (fluoroscopia) a 100 (cine en angiografía) cuadros/segundo.
- El tiempo de exposición normal de una placa estática es del orden de 100ms o más. Con escenas dinámicas esto se reduciría a 1/25=40ms o menos por cada cuadro. Esta dosis es insuficiente por si sola para producir una imagen de resolución aceptable.
- Es necesario utilizar un sistema de “amplificación” de la señal de rayos x recibida.

Se utiliza un tubo intensificador de imagen.

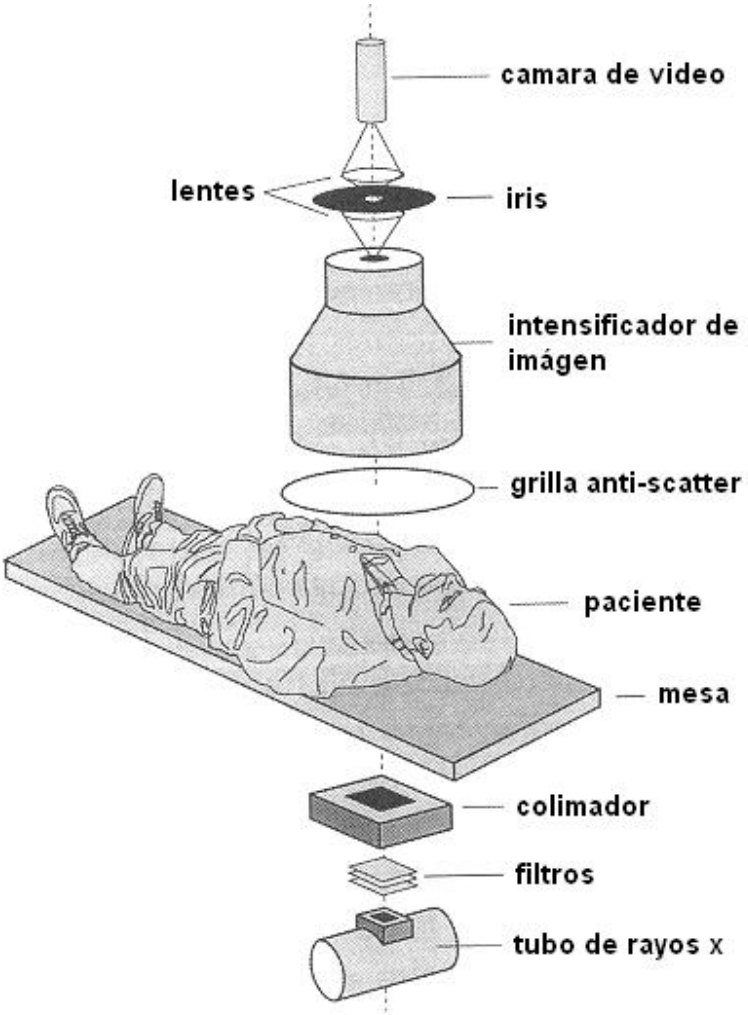


Figura 4.1.5 tubo intensificador de imagen lugar de colocación.

- Posee 4 componentes fundamentales:

  — Un tubo de vacío dentro del cual los electrones son acelerados con alto voltaje.
  — Una pantalla de entrada donde los rayos x se convierten en electrones.
  — Una cadena de lentes electrostáticos que enfocan el haz de electrones.
  — Una pantalla de salida que convierte los electrones en luz visible.

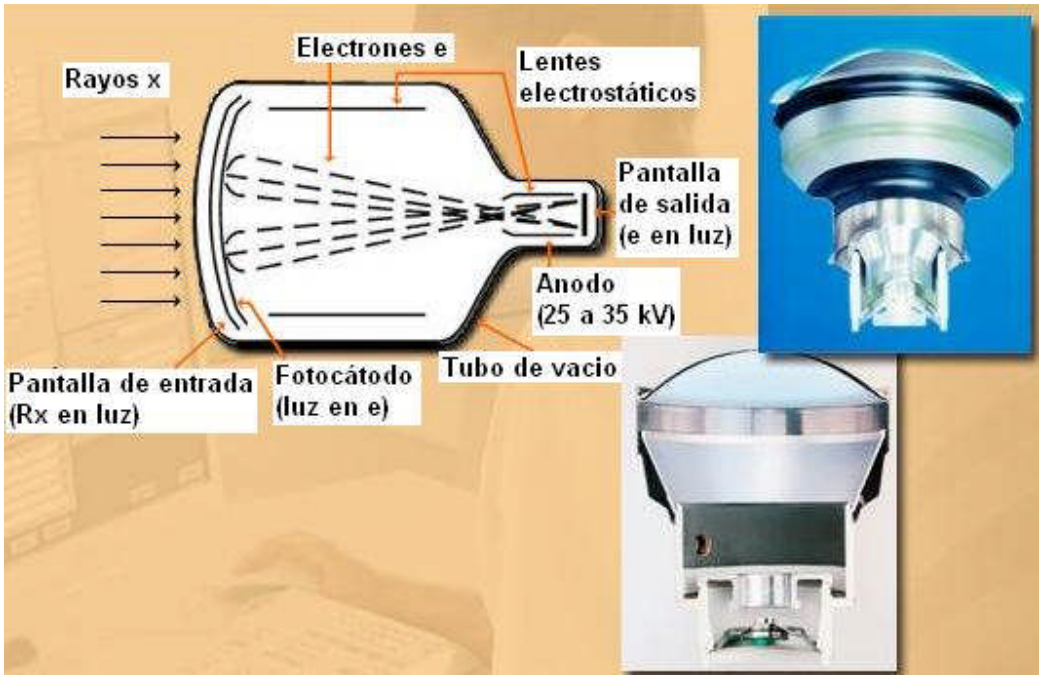


Figura 4.1.6 tubo intensificador de imagen.

- Los electrones llegan a la pantalla de fósforo que se encuentra en la salida, solo el 1% de los electrones incidentes serán convertidos en luz, la cual será luego capturada por cámaras de TV.
- El proceso de aceleración y minificación (reducción de tamaño), logran amplificaciones de la información del orden de 10000 veces.

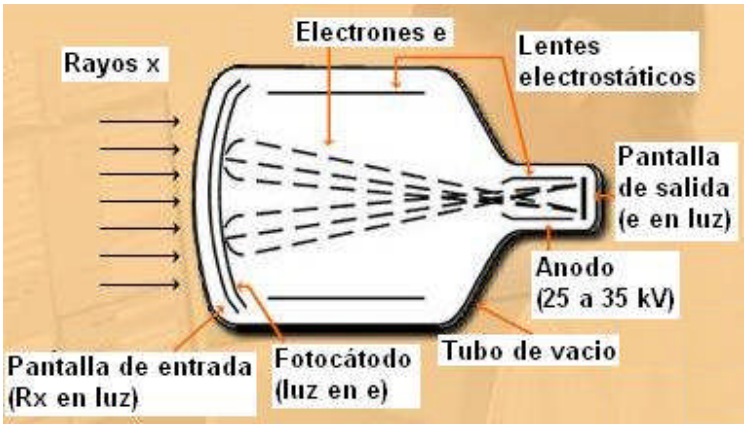


Figura 4.1.6 tubo intensificador de imagen diagrama.

Resumen:

— Contamos con una fuente de RX (tubo).
— Paciente donde dichos rayos son atenuados.

— Sistemas de detección de dicha atenuación (film, pantalla intensificadora, tubo intensificador de imagen, cadena de TV, etc).
— Imagen representativa de dicha atenuación.

4.2.4. Aplicaciones

Surgen así diferentes áreas de aplicación de dichas propiedades que veremos a continuación

4.2.4.1. Radiología convencional

- Es tal vez la técnica más popular, utilizada en ortopedia y traumatología para ver huesos. Se utilizan placas junto con pantallas intensificadoras.

Aplicaciones:

— Identificar fracturas, artrosis, etc.
— Radiología de tórax, etc.

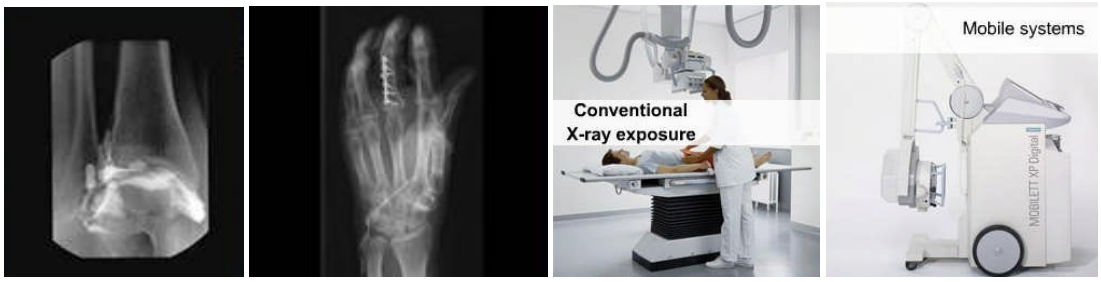


Figura 4.1.2.1 Equipos de Rayos X y placas obtenidas

4.2.4.2. Fluoroscopio

- Fluoroscopía o radioscopía: similar al anterior pero permite estudios dinámicos, es decir, ver secuencias de video en tiempo real. Generalmente con el uso líquidos de contraste. Se utilizan tubos intensificadores de imagen y cadenas de TV convencionales.

Aplicaciones:

— Seguimiento y visualización del tracto gastro-intestinal.
— Esófago, intestino grueso y delgado, etc.

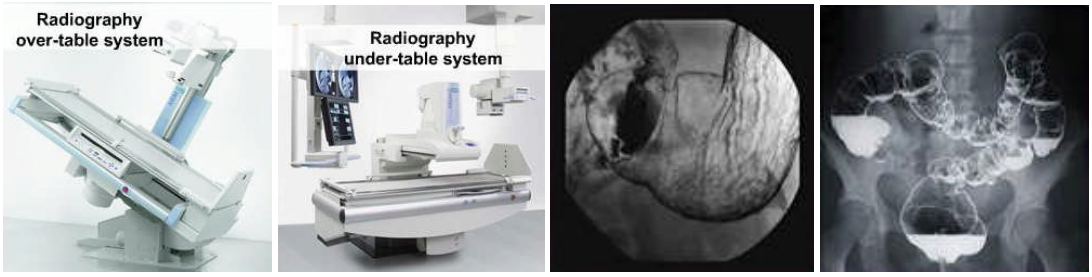


Figura 4.1.2.2, Equipos de Fluoroscopio y placas obtenidas

4.2.4.3. Angiografía

- Técnica dedicada a la visualización de vasos sanguíneos, venas y arterias. Mediante la inyección de contrastes se pueden ver con claridad. Se utilizan tubos intensificadores de imagen y cadenas de TV especiales.

Aplicaciones:

- Estudios de hemodinámica, localización de estenosis o malformaciones de ciertos vasos.
- Vascularización de tumores.
- Estudios coronarios, etc.

Figura 4.1.2.3, Equipos de Angiografía y placas obtenidas

4.2.4.4. Mamografía

Técnica utilizada para ver en detalle el tejido mamario. Poseen una altísima resolución, se pueden ver detalles muy pequeños. Se utilizan placas junto con pantallas intensificadoras.

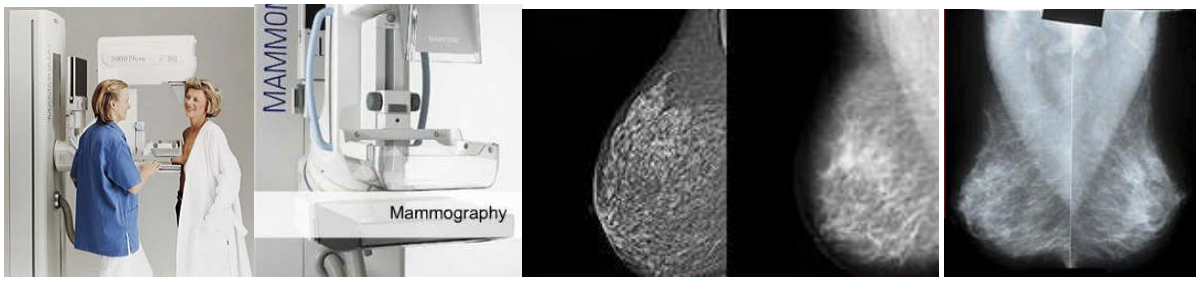


Figura 4.1.2.4, Equipos de Mamografía y placas obtenidas

4.2.4.5 Arco en C

- Similar a un equipo de angiografía pero de menor potencia y mas protatíl. Se utilizan tubos intensificadores de imagen junto con cadenas de TV convencionales.

Aplicaciones:

- Intervenciones quirúrgicas.
- Estudios hemodinámicas, etc.

Figura 4.1.2.5, Equipo de Mamografía y placas obtenidas

4.2.4.6. Litotricia

- Localización de cálculos para litotricia: la litotricia es la técnica que se encarga de la destrucción de cálculos mediante la aplicación de ondas de ultrasonido. La visualización de dichos cálculos y centrado de los disparos se realizan con la ayuda de rayos x. Se utilizan tubos intensificadores de imagen junto con cadenas de TV convencionales.

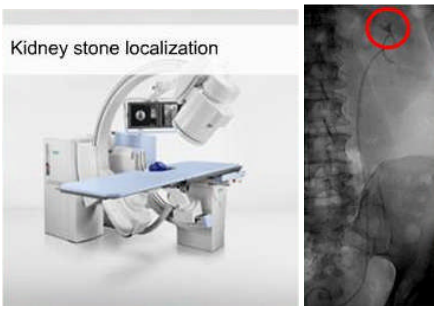


Figura 4.1.2.6, Equipo de Litotricia y placas obtenidas

4.2.4.7. Tomografía computarizada

- Se obtienen imágenes anatómicas del cuerpo humano para el diagnóstico de múltiples patologías, cortes 2D o imágenes 3D. Se utilizan otro tipo de detectores no visto, detectores de gas, cerámicos, estado sólido, etc.

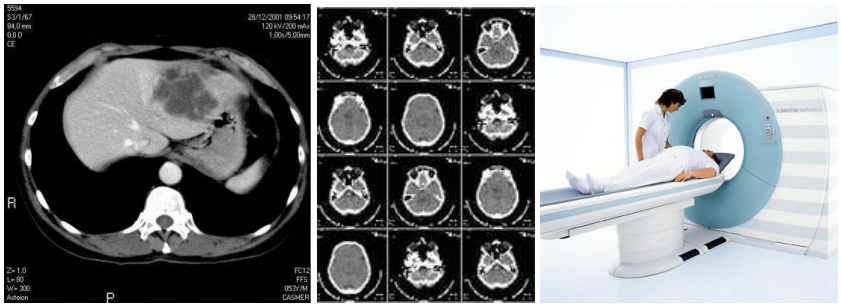


Figura 4.1.2.7, Equipo de Tomografía Computarizada y placas obtenidas

4.3. Radiología digital vs Radiología analógica

Beneficios obtenidos:

- Menor dosis de radiaciones para el paciente y el operador.
- Menor cantidad de material contaminante (Plomo, Químicos de revelador y fijador).
- Ahorros económicos: placas radiográficas y rollos fotográficos, ahorro en la compra de reveladores y fijadores, ahorro en la compra y mantenimiento de procesadoras de placas y equipos de revelado.
- Disminución del espacio físico para guardar las imágenes, uso de archivos digitales.
- Diagnóstico remoto y envío de resultados por intranet hospitalaria o internet, brindando rapidez, practicidad y posibilidad de interconsulta entre profesionales al instante.
- Alto contraste de las imágenes digitales, uso de monitores especiales software con herramientas de procesamiento que ayudan al médico, facilitando y mejorado el diagnóstico.

Técnicas de digitalización

- Ciertos equipos (modalidades), como ser CT, MR, NM, US, DSA es mucho mas común que posean salida digital (aunque no siempre).
- Actualmente hay disponibles equipos de RX con detectores digitales.
- Otros como RX convencional, portátiles, mamografía, radioscopia, etc no es común que la tengan y hay que digitalizarlos.
- Tenemos 2 maneras de hacer esto:

    - Forma directa.
    - Forma indirecta.

Digitalización en forma directa

- **DR (Digital Radiography):**
    - Se utilizan detectores digitales directamente del tipo “flat pannel” quienes convierten los Rx en luz (yoduro de cesio) y son captados por pequeños elementos del estilo TFT.
    - DDR es una variante en la cual no hay conversión a luz, directamente pasan de Rx a señales eléctricas.
- CR (Computed Radiography):
    - Esta en el límite entre ser un método directo o indirecto.
    - Se sustituye la placa convencional por una placa con capacidad de memoria:

DR y DDR

- Son llamados detectores flat pannel.
- Una fina capa de yoduro de cesio que emite luz al incidirle rayos x.
- Matriz de detectores: cada pixel consiste de un transistor, una celda TFT (thin film transistor) y un fotodiodo. El fotodiodo convierte la luz en un voltaje que es almacenado en el condensador y luego leído por los IC con ayuda de cada transistor de la matriz TFT.
- Existe otro tipo de detectores directos, donde se utiliza fotodetectores de celenio y no es necesario el pasaje a luz, los rayos x son directamente convertidos en corrientes eléctricas.

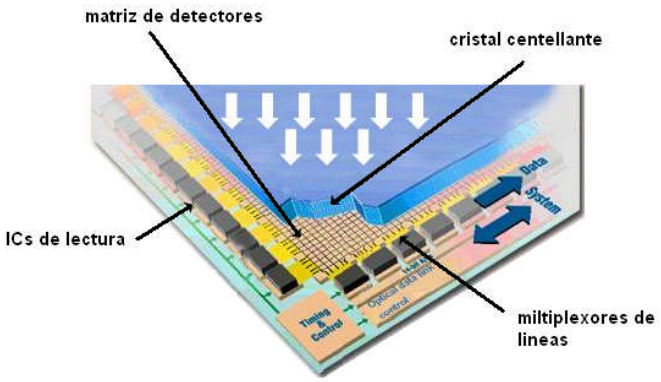


Figura 4.1.3.1 detectores flat panel

CR

- Placa de fluorobromo de bario, los Rx hacen que electrones pasen de un estado de baja energía a uno de mas alta. Al volver a su estado de reposo emitirían luz, pero esto es impedido mediante “trampas” existentes en la placa.
- Dicha placa se coloca en el CR quien realiza un barrido punto a punto con un láser de He-Ne de 633nm, provocando la liberación de las “trampas” y volviendo a su estado de reposo emitiendo luz azul de aprox 400nm. Dicha luz es captada y convertida en una señal eléctrica.
- Luego la placa se borra sometiendola a luz intensa quedando lista para un nuevo uso, llegan a durar alrededor de 3000 reusos.

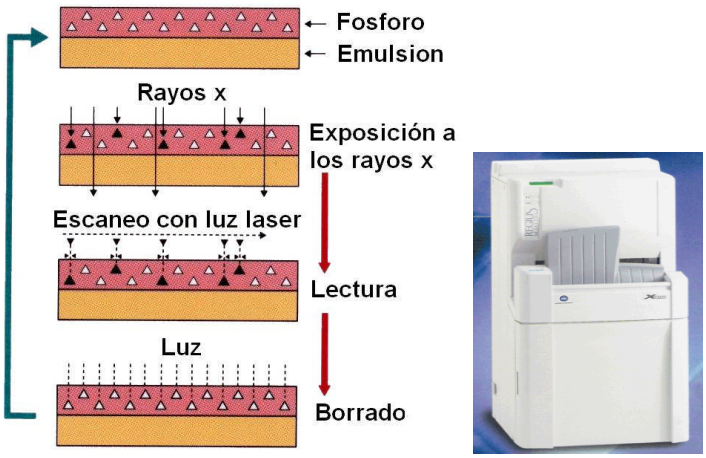


Figura 4.1.3.2, Reveladora de rayos X

4.4. RESONANCIA MAGNÉTICA (MRI)

4.4.1 INTRODUCCIÓN

CT y MR sirven para lo mismo?

- Tomografía Computada es una técnica basada en rayos X y produce imágenes cuyo contraste es determinado principalmente por la densidad de la masa que atraviesan
- La siguiente grafica muestra la densidad de cada uno de los diferentes tejidos y de esta forma la habilidad de CT para diferenciar entre diferentes tejidos y hueso. Ver que los tejidos blandos solo caen en el rango de los 10 a los 60 HU en un rango total de unos 4000.
- Por ello CT no es muy buena para diferenciar tejidos blandos y si lo es para ver hueso. Como veremos MR es lo contrario.

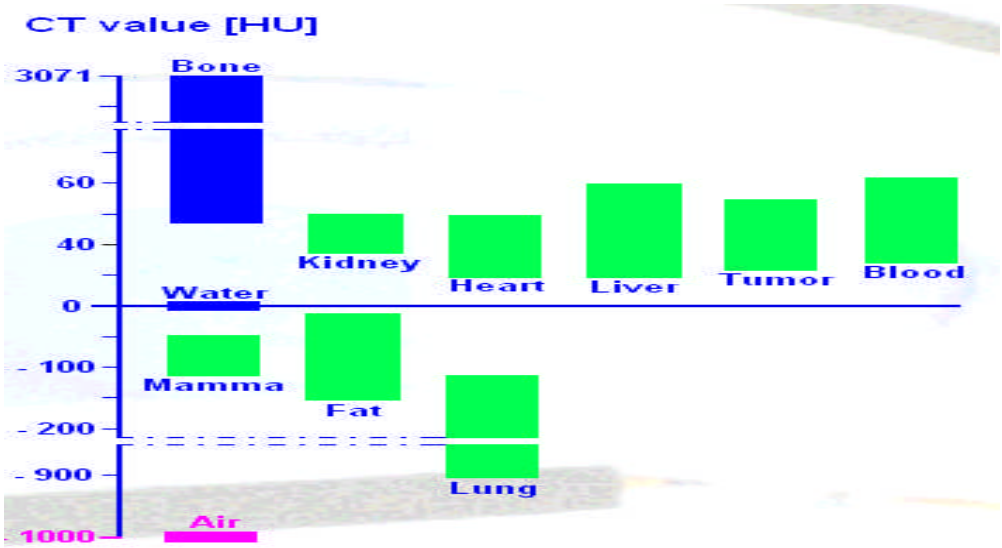


Figura 4.2.1 densidad de cada uno de los diferentes tejidos

- La Resonancia Magnética (MR) es capaz de medir los protones de los átomos de hidrógeno en las moléculas de agua. La gran cantidad de agua existente en los tejidos blandos hacen que MR sea excelente para ver este tipo de tejidos.
- MR tiene ciertaa ventajas sobre CT:
  - Excelente para diferenciar tejidos blandos
  - Las imágenes pueden ser adquiridas directamente en cualquier orientación
  - No se usan radiaciones ionizantes, es inofensivo para el paciente.
  - Los medios de contraste usados en MR son menos agresivos que en CT

Tejido Blando Hueso

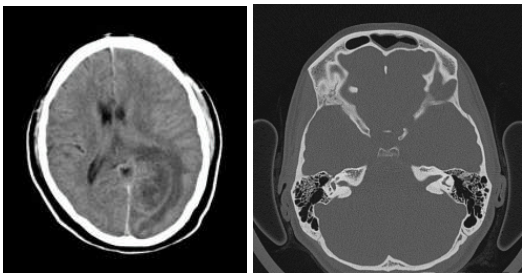


Figura 4.2.2 Ejemplos de imagen de CT

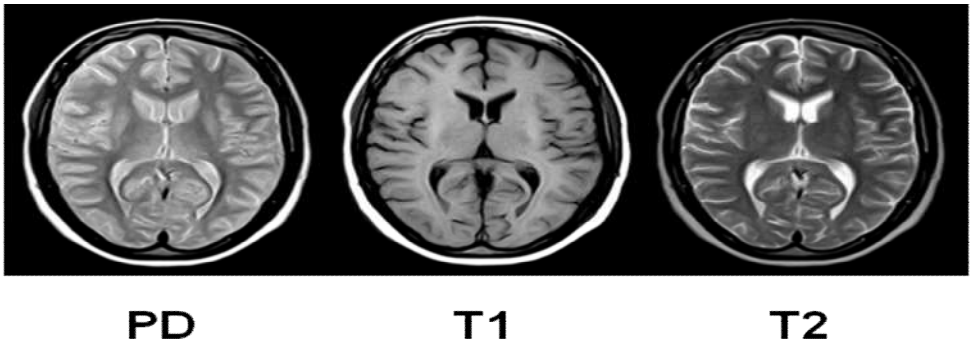


Figura 4.2.3 Ejemplos de imágenes MR

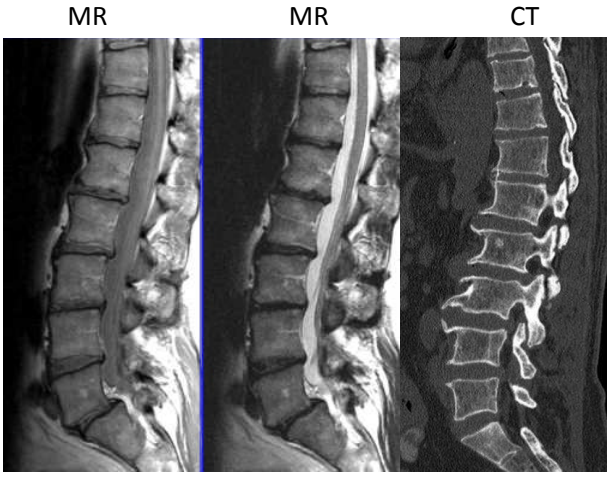


Figura 4.2.3 Imágenes MR vs CT

Figura 4.2.4 Visualización de tejidos blandos MR

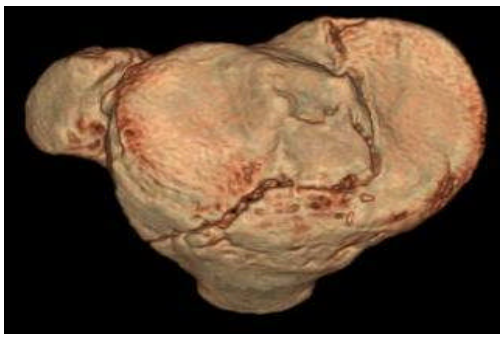


Figura 4.2.5 Visualización de fractura MR

4.4.2. MRI TEORIA

4.4.2.1 Protones y su PIN

- Las moléculas de agua están constituidas por dos moléculas de Hidrógeno y una de Oxígeno.

- El átomo de Hidrógeno posee un protón y un electrón.
- Dicho protón en el núcleo del átomo es quien proveerá la señal de RM

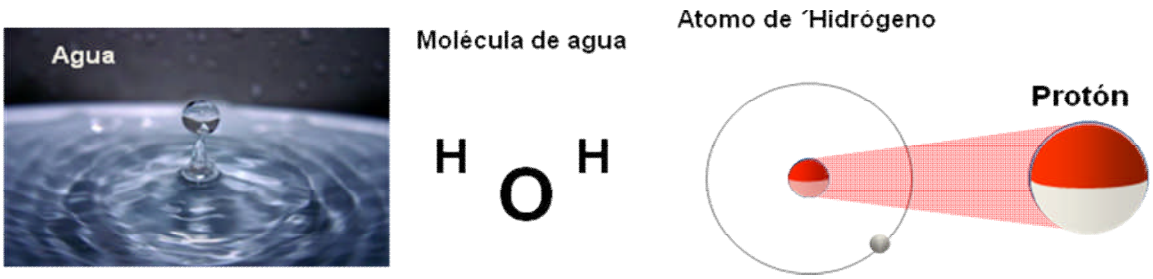


Figura 4.2.1.1.1 Molécula de agua

- Los protones poseen una propiedad llamada **Spin** e indica que tienen un momento angular, están rotando sobre su eje al igual que un trompo.
- El spin se representa mediante un vector que sigue la regla de la mano derecha.
- Adicionalmente poseen un momento magnético, quiere decir que generan un campo magnético, similar a un imán.

Figura 4.2.1.1.2 Momento angular de una protón

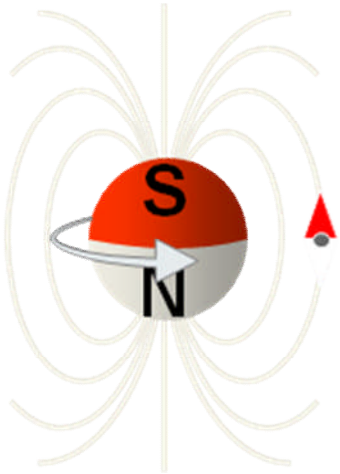


Figura 4.2.1.1.3 Momento magnético de un protón

Presión

- Que sucede cuando dicho protón es sometido a un campo magnético externo uniforme Bo?
- Su Spin hace que el protón comience un movimiento de precesión a una frecuencia w proporcional a la intensidad del campo externo Bo.
- El valor de w viene dado por la ecuación de Larmor que la relaciona con Bo y con la constante gyro-magnética g (constante de proporcionalidad dependiente del átomo en cuestión):

$$\omega = \gamma \, . \, B_0$$

$2\pi f = \gamma \, . \, B_0 \qquad f = \gamma / 2\pi \, . \, B_0$

Para el Hidrógeno ^1H: γ = 42.577MHz/T

f = 42,577 MHz para un campo magnético de 1T

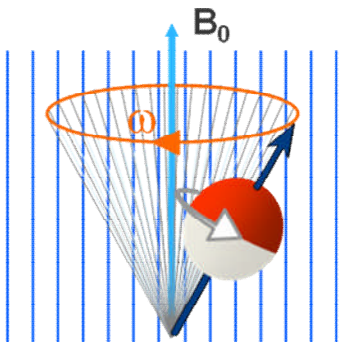


Figura 4.2.1.1.4 protón sometido a un campo magnético externo uniforme Bo

Orientación de los protones

- Cuando el campo magnético externo Bo es nulo, los spines se orientan en forma aleatoria.
- Resultando una magnetización neta M igual a cero.

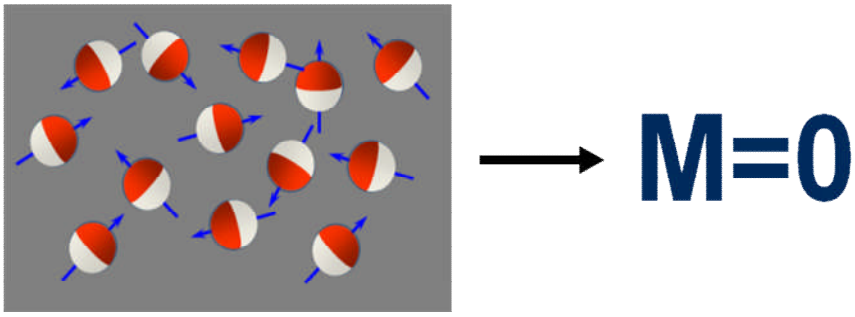


Figura 4.2.1.1.5 espines con orientación aleatorio

- Cuando el campo magnético externo Bo **no** es nulo, los spines se orientan en forma paralela o antiparalela al campo Bo.
- Existe una muy pequeña mayoría de ellos que se orientan en forma paralela a Bo.
- Dicha mayoría crece cuando crece Bo. Es por esto que cuanto mayor sea el campo externo, mayor será la intensidad de la señal recibida de los protones por el equipo de MR.
- Ejemplo: en un campo de 1T, si consideramos 2exp106 protones, solo habrá 7! Capaces de emitir señal.

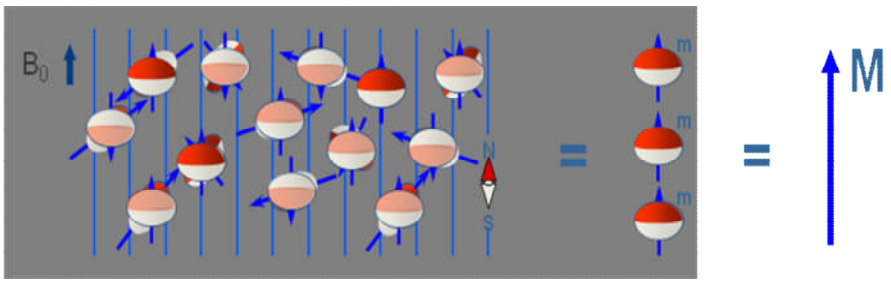


Figura 4.2.1.1.6 espines con orientación aleatorio y algunos orientados en forma paralela a Bo

Excitación

- La idea es hacer que estos protones absorban energía y cambien de nivel (del paralelo al antiparalelo), esto se logra utilizando RF.
- Los pulsos de RF deben ser de una frecuencia f que sea igual a la frecuencia de Larmor, solo así se producirá la absorción de energía. Es a esto que se llama resonancia.

Figura 4.2.1.1.7 Excitación de los espines aplicando un señal de radio frecuencia igual al f de Lermor

- Los spines no solo comenzaran a cambiar al estado antiparalelo sino que también comenzarán a girar en forma coherente, esto es todos con la misma fase.

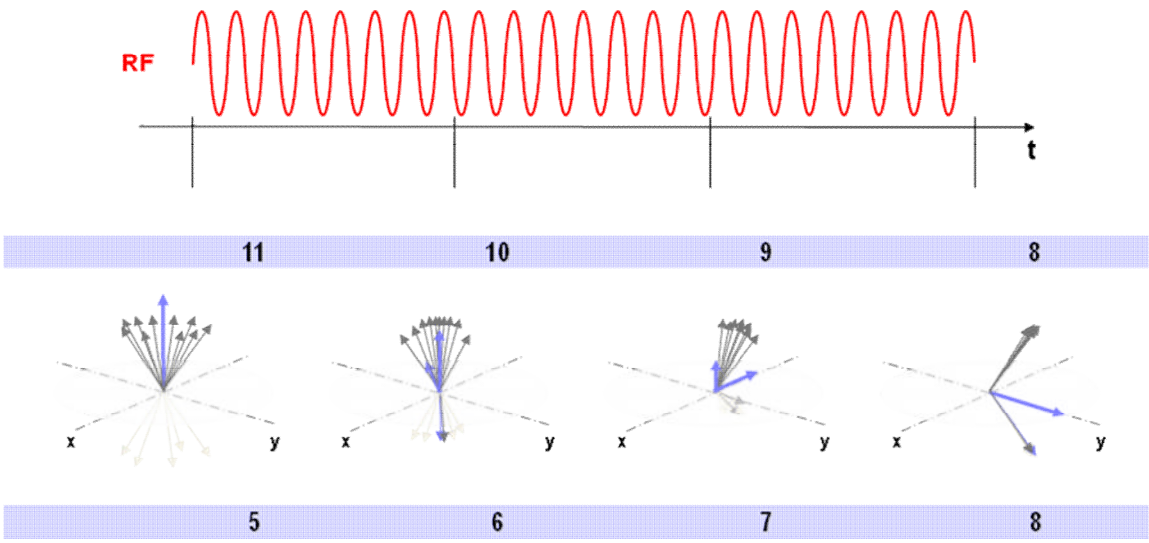


Figura 4.2.1.1.8 Fasores de un espin sometidos a la frecuencia de Lermor

Medición de la señales de MR

- Cuando el pulso de RF es quitado, los protones vuelven a su estado inicial, emitiendo la energía que absorbieron cuando el pulso de RF estaba presente. A este proceso se le llama **relajación**
- Separamos el vector de M en dos componentes, Mz se llama componente longitudinal y Mxy se llama transversal.

- Se dispondrán antenas de tal modo que **solo la componente transversal Mxy sea captada**

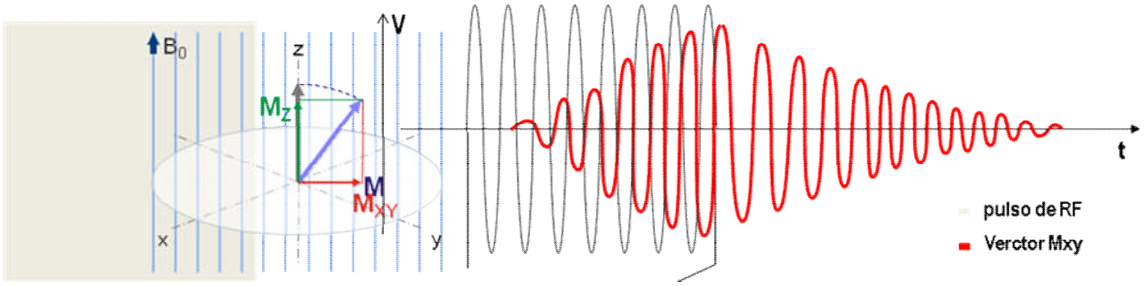


Figura 4.2.1.1.9 Excitación y relajación de un espin

Relajación y contraste

- En MR el contraste de las imágenes quedan determinado por los parámetros de la secuencia utilizada (dependiente del usuario) y por otros 3 parámetros dependientes del tejido en cuestión, estos son:
  - PD: densidad de protones, en este tipo de imágenes cada pixel representa la cantidad de protones que hay.
  - T1: tiempo de relajación T1, en este tipo de imágenes el tiempo de relajación de la componente longitudinal Tz es el que tiene mayor peso en el valor de cada píxel, es usual llamarlas imágenes T1 weighted.
  - T2: tiempo de relajación T2, ídem que T1 pero tomando en cuenta el tiempo de relajación de la componente transversal Txy.
- Los tiempos de relajación son únicos para cada tipo de tejido y son quienes juegan un papel fundamental para obtener el contraste de las imágenes.
- Tiempo de relajación T1

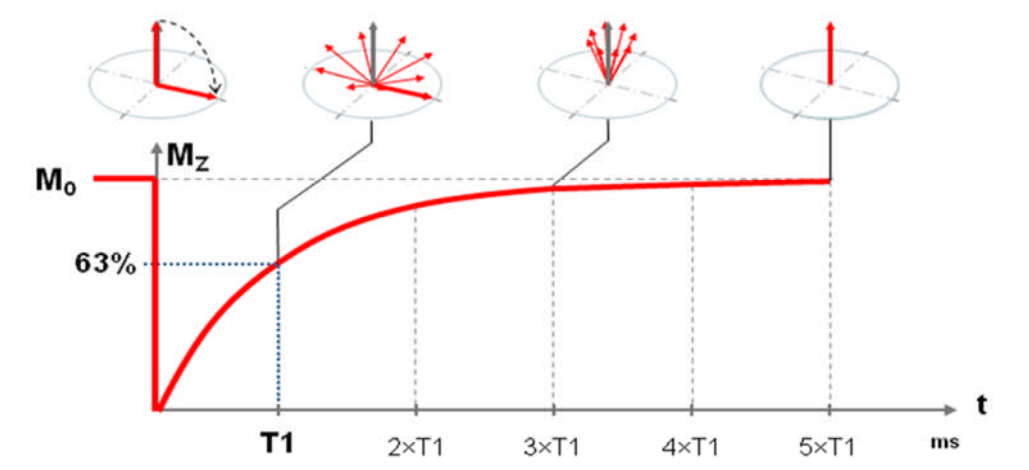


Figura 4.2.1.1.10 Tiempos de relajación de T1 componente Mz

- Este tiempo T1 es dependiente del tipo de tejido en el que se encuentren "inmersos" los protones, por dicha razón es específico del tejido que se esté excitando.

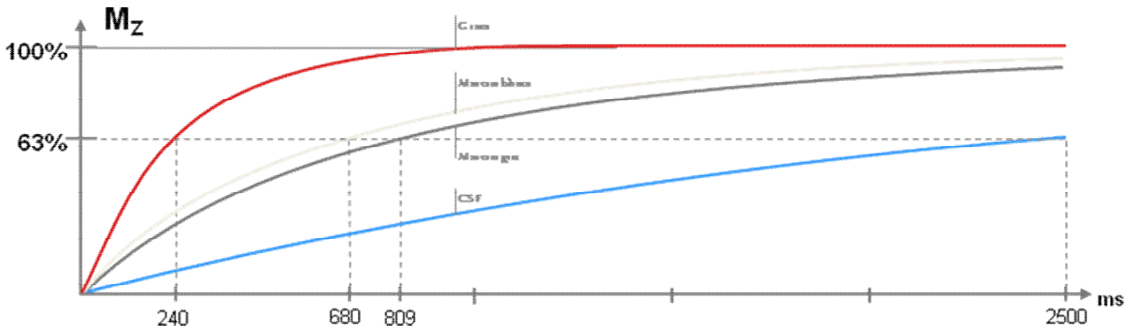


Figura 4.2.1.1.11 Tiempos T1 de relajación de tres distintas tejidos

Tejido	T1 [ms] (a 0.2T)	T1 [ms] (a 1T)	T1 [ms] (a 1.5T)
Grasa	200±60	250±70	260±70
Hígado	228±50	420±92	490±110
Riñón	393±110	587±160	650±180
Vaso	398±75	680±130	778±150
Materia blanca	388±66	680±120	783±130
Músculo esquelético	370±66	730±130	863±160
Músculo cardíaco	416±66	745±120	862±140
Materia gris	492±84	809±140	917±160
CSF	1500±400	2500±500	3000±600

Figura 4.2.1.1.12 Valores de T1 para algunos tejidos

Tiempo de Relajación para T2

- Este es el tiempo de relajación de la componente transversal, esta determinado por la interacción entre protones (los spines se anulan entre ellos al desfasarse). Se llama relajación spin-spin.
- Se define T2como el tiempo en que tarda la componente transversal en decaer al 37% de su valor inicial.

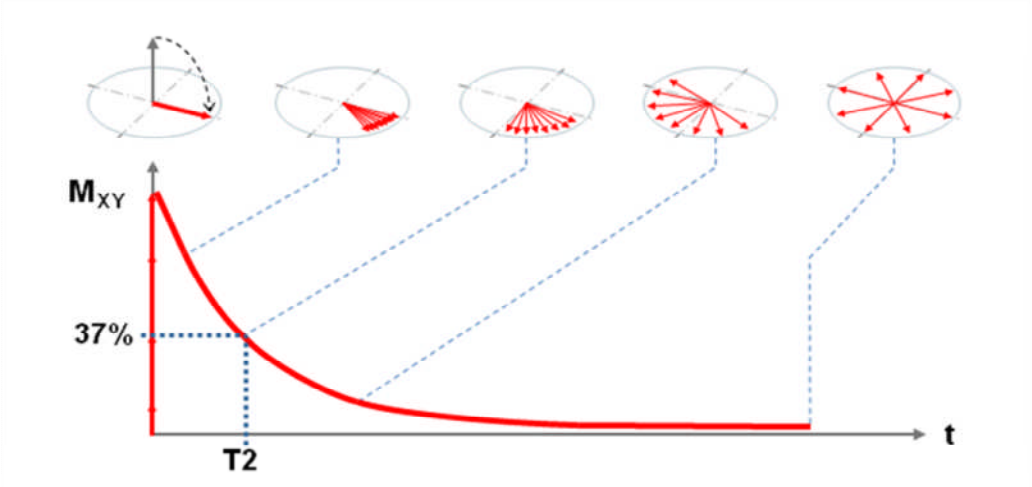


Figura 4.2.1.1.13 Tiempos de relajación de T2 componente Mxy

- Este tiempo T2 también es dependiente del tipo de tejido en el que se encuentren "inmersos" los protones, por dicha razón también es específico del tejido que se esté excitando.

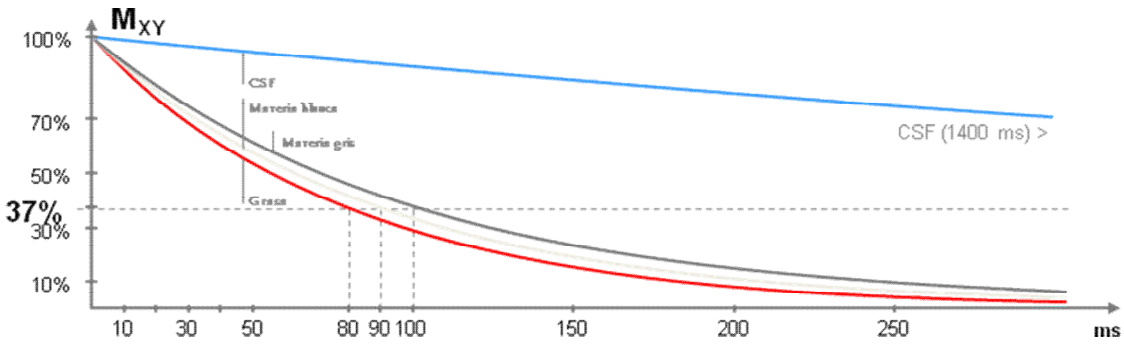


Figura 4.2.1.1.14 Tiempos T2 de relajación de tres distintas tejidos

Tejido	T2 [ms]
Hígado	43 ±6
Músculo esquelético	47 ±6
Músculo cardíaco	57 ±9
Riñones	58 ±8
Vaso	62 ±17
Grasa	80 ±36
Materia blanca	92 ±20
Materia gris	101 ±13
CSF	1400 ±250

Figura 4.2.1.1.15 Valores de T2 para algunos tejidos

4.4.3.. MRI Aplicación

Repaso

- Hemos visto que los protones tienen un momento magnético llamado spin y cuando estos son incluidos en un campo externo Bo se obtiene una magnetización neta M en el sentido de Bo.
- Al excitar estos protones con RF de frecuencia igual a la de presesión (Larmor), estos absorben energía. **Solo** esta frecuencia producira absorción de energía y rotación del vector M.
- Todo el resto de las frecuencias de RF no tendrán efecto sobre los protones.
- Al retirar la RF, se detectan mediante antenas en el plano transversal la señal emitida por los protones.
- La señal medida es la suma de todas las señales de los protones de todo el tejido excitado.
- Debemos diferenciar de donde proviene cada una de las señales, tantas señales diferentes como pixeles en mi imagen. Llamamos a esto **Localización espacial**

Localización Espacial

- Debemos lograr que en cada punto del espacio exista un campo magnético ligeramente diferente a Bo, de esta forma la frecuencia de presesión de los átomos variara en el espacio.
- Esto se logra con el uso de gradientes, hay 3 gradientes, uno para cada uno de las direcciones espaciales x, y, z.

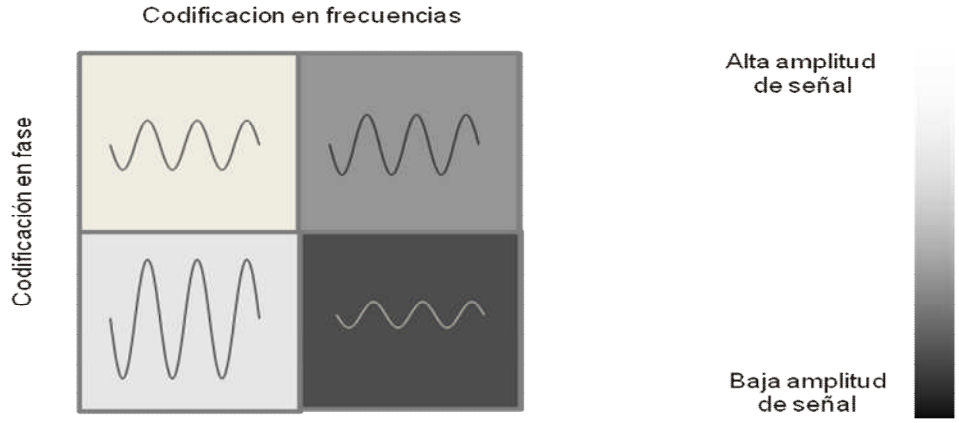


Figura 4.2.1.1.16 Codificación del pixel

Codificación Espacial

- De esta forma utiliza uno de los gradientes (z por ejemplo) para excitar solo una slice (rebanada) de tejido para así formar una imagen en 2D.
- Luego los otros 2 gradientes se utilizan para lograr codificación en frecuencia.
- De esta forma cada punto del espacio posee un único valor de frecuencia. Es decir cada voxel (píxel en mi imagen) va a responder a una frecuencia de resonancia diferente.
- En realidad se utiliza codificación en frecuencia en una dirección y en fase en la otra pero no vamos a entrar en detalle.

- Al recibir la señal de MR, recibimos la suma de todos los protones de todo el slice excitado. Luego utilizando Fourier como sabemos, separamos en componentes de frecuencia y tendremos asi el valor de cada uno de nuestros pixeles.

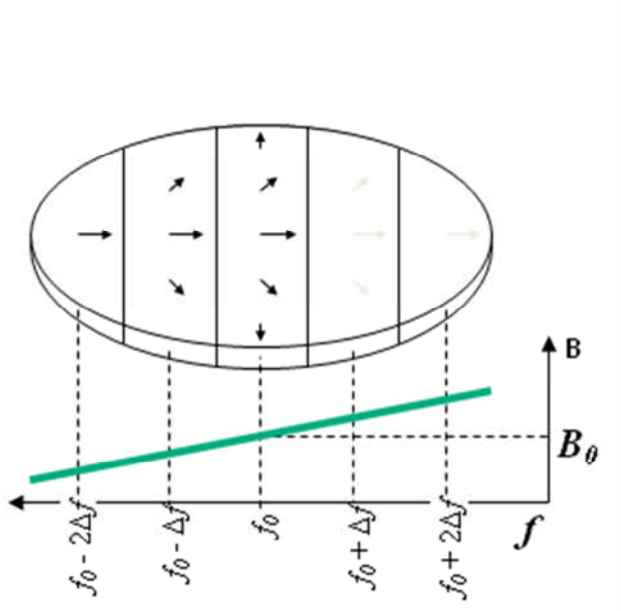


Figura 4.2.1.1.17 Codificación de una imagen en rebanada utilizando Furier sumando las componentes

- La idea es lograr variaciones en el cambo Bo, en cada una de las direcciones.
- Para ello hay 3 gradientes, Gx, Gy, Gz.

Gradiente

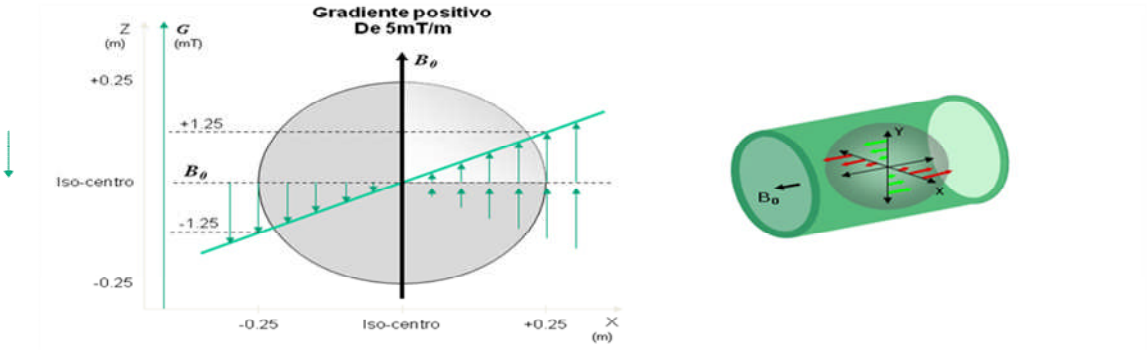


Figura 4.2.1.1.19 Gradiente emitido

Pulsos de RF

- Hemos visto como codificar espacialmente los puntos de un slice (imagen 2D).
- Pero como seleccionamos un slice?, su posición y su espesor?

- Puedo hacerlo de 2 formas, aumentando mi gradiente o variando la frecuencia central de mi pulso de RF.
- Se utilizan pulsos selectivos de RF, esto es funciones sinc en el tiempo.

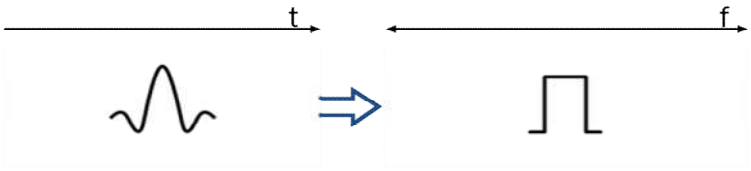


Figura 4.2.1.1.20 Pulsos selectivos de RF

Selección del Slice

- Dependiendo que gradiente utilice para hacer la selección del slice determino la orientación del mismo:

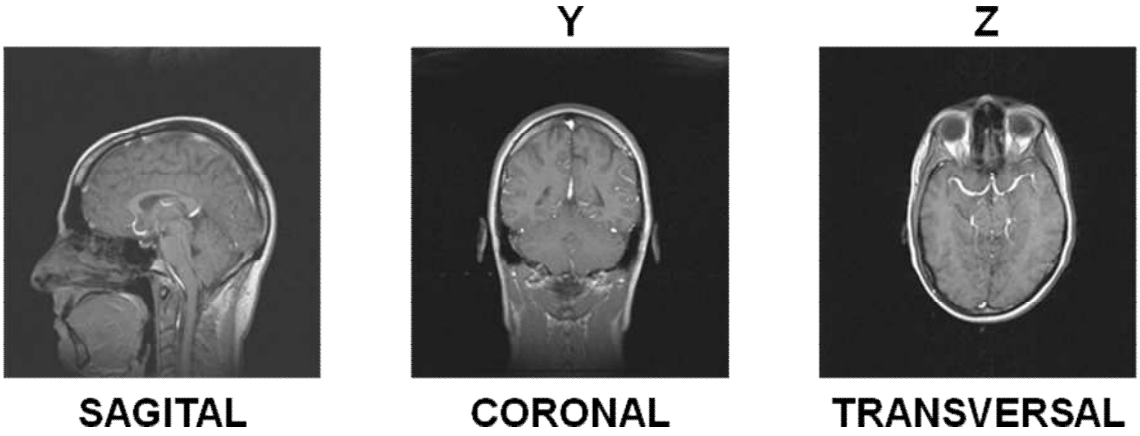


Figura 4.2.1.1.21 Gradiente utilizado para seleccionar el Slice

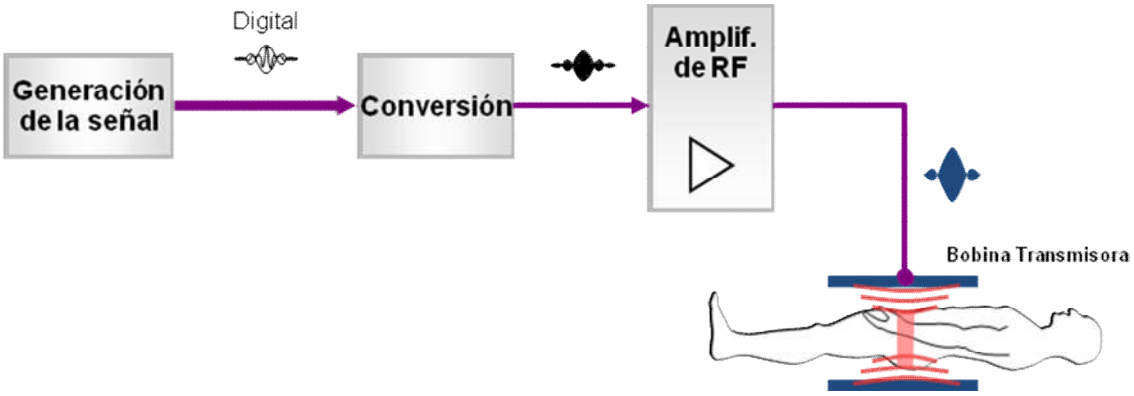
4.2.3 MRI Instrumentación

Figura 4.2.1.1.6 Equipos de resonancia magnetico

Sistema de RF

- **Transmisión:**
  - Generación de pulsos de RF.
  - Amplificación de la señal de RF.

- Transducción V, I a B, E. Uso de antenas
- Adaptación de impedancias en la transmisión (Macheo de impedancias)



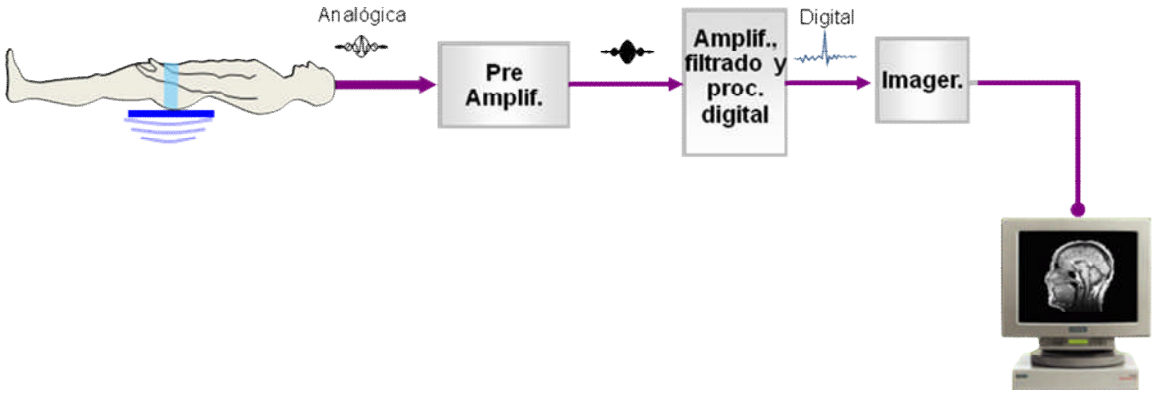
- La bobina transmisora convierte la señal de tensión en campo electromagnético, dicha señal de RF interacciona con los protones como ya vimos.

- El pulso analógico de RF entra al amplificador para incrementar su potencia y lograr la excitación adecuada en los protones.

- El pulso transmisor es calculado y modulado digitalmente utilizando DSPs, luego es enviado al transmisor para convertir dicha señal en analógica a la frecuencia de RF requerida

- El pulso amplificado es aplicado a la bobina transmisora para excitar el slice seleccionado

- **Recepción:**
  - Captación de pulsos de RF. Uso de antenas especiales lo mas cercanas posible al cuerpo del paciente.
  - Amplificación en las propias bobinas (antenas) de la señal recibida.
  - Amplificacion y filtrado en el módulo de recepción de RF.
  - Procesamiento digital y envio al PC de reconstrucción.



- Luego de la excitación de los protones, la señal de eco debe ser leída. La bobina receptora debe estar en la posición correcta para captar la señal de RF emitida por los protones. Las bobinas receptoras pueden ser de varios tipos y diseños, LP, CP, volumétricas, de superficie, etc.

- La señal es procesada digitalmente y enviada al Imager, computadora encargada de hacer los cálculos para la reconstrucción de la imagen

- La señal obtenida es preamplificada en las mismas bobinas ya que es muy pequeña, además se cuenta con electrónica que permite seleccionar múltiplex bobinas (canales).

- La imagen es enviada al Host que la despliega en el monitor

4.5 Ultra Sonidos

Generalidades

En su afán de estudio, la medicina busca desde hace tiempo métodos diversos para poder representar en imágenes el interior del organismo. El avance más espectacular fue conseguido a finales del siglo XIX con el descubrimiento de los rayos X, ya que además de ser un buen método abriría paso a los demás sistemas de imágenes.

Intentando desarrollar otras técnicas se empezaron a utilizar los ultrasonidos con bases similares a las del radar y con equipos semejantes a los utilizados en la industria para detectar fallos en los materiales.

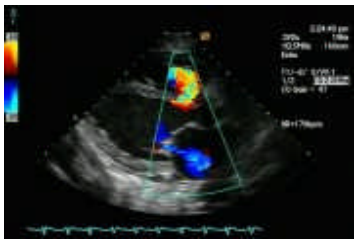


Figura 4.3.1 Imagen adquirida por un equipo de ultrasonido

Actualmente, el uso de ultrasonidos en la medicina ha encontrado un lugar sólido entre la variedad de métodos para obtener imágenes del cuerpo. Las razones para esta popularidad son muchas, pero quizás derivan de la simplicidad y la seguridad en su uso.

Los ultrasonidos se definen como ondas acústicas con frecuencias por encima de aquellas que pueden ser detectadas por el oído humano, desde aproximadamente 20 KHz hasta varios cientos de megahertz. En contraste con las ondas electromagnéticas, estas vibraciones necesitan de un medio físico para su propagación.

Los instrumentos médicos utilizan sólo una porción del espectro de ultrasonidos, entre 1 MHz y 10 MHz, debido a las necesidades combinadas de buena resolución (longitudes de onda pequeñas) y buena penetración en los tejidos (frecuencias no demasiado altas).

4.5.1 Instrumentos biomédicos basados en ultrasonido

El campo de los ultrasonidos es muy amplio, habiendo varias técnicas. El campo más conocido hoy en día es el obstétrico, para el control del feto intraútero. Esto permite comprobar el estado fetal en cuanto a su crecimiento y a su morfología, permitiendo asimismo el estudio de los perímetros pélvicos para conocer la viabilidad del parto normal; un caso especial y muy comentado, pero de poca importancia clínica, es el diagnóstico del sexo del futuro recién nacido.

Además de ésto, los ultrasonidos son utilizados en las siguientes áreas:

- Imágenes por efecto Doppler
- Flujímetros Doppler
- Tomografía computada por ultrasonidos
- Cirugía (destrucción de estructuras por vibración)
- Producción de calor
- Microscopio ultrasónico
- Fisioterapia

En la sección equipos de ultrasonidos se describen estos ítems en detalle.

4.5.1.1. Teoría

La amplia gama de instrumentación biomédica que hace uso de los ultrasonidos se apoya en dos aspectos físicos comunes a todo tipo de ondas:

- Los fenómenos de transmisión, absorción y reflexión.
- El efecto Doppler.

Las bases matemáticas que describen estos procesos (ecuación de ondas, resolución de ecuaciones diferenciales parciales, etc.) son relativamente complicadas y escapan al objetivo de este documento, que yace principalmente en la comprensión cualitativa del fenómeno. No obstante, se realiza el desarrollo matemático de la ecuación que describe el efecto Doppler debido a que es muy útil en la comprensión del fenómeno.

Transmisión, absorción y reflexión de las ondas

Como se dijo en la sección generalidades, los ultrasonidos no son más que ondas acústicas de frecuencia superior a la que puede detectar el oído humano. En contraposición con las ondas electromagnéticas, las ondas sonoras necesitan de un medio material para propagarse.

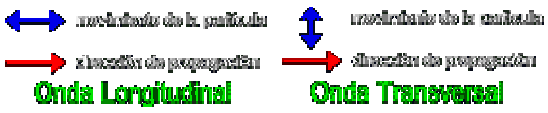


Figura 4.3.3.1 Tipo de ondas

Además, dentro de las ondas acústicas, las más útiles en instrumentación son las ondas longitudinales. Es decir, aquellas en las que las partículas del medio transmisor se mueven en la dirección de propagación de la onda, creando zonas de compresión y descompresión. Las ondas transversales son aquellas en las que la partícula se mueve en la dirección perpendicular a la dirección de propagación de la onda (como el caso de una cuerda atada en un extremo a la que se le suministra un estímulo en la punta opuesta).

La instrumentación basada en la penetración de ultrasonidos en objetos no sería posible si no existiera la reflexión de las ondas que revelase las interfases internas de los objetos. Cada vez que una onda pasa de un tejido a otro (con distinta impedancia acústica) cierta cantidad de la energía incidente es reflejada en la interfase. La energía restante continúa como onda transmitida.

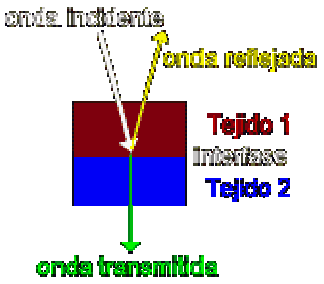


Figura 4.3.3.2 Comportamiento de una onda al pasar por un cuerpo

La onda reflejada sirve como indicador de la posición del límite entre los dos tejidos y de la forma de la interfase, mientras que la porción transmitida sigue buscando interfases más profundas.

Entonces, midiendo los tiempos a los que se producen las reflexiones (ecos) se obtiene información acerca de la profundidad de cada interfase y de la forma de la misma. Por otro lado, la cantidad de energía reflejada depende de los tipos de tejidos que existen a ambos lados de la interfase. Así, se obtienen datos morfológicos e histológicos de la muestra estudiada.

Por último, hay que tener en cuenta que se produce absorción de los ultrasonidos por parte de los tejidos. La energía absorbida es disipada en forma de calor. Cuanto mayor sea la frecuencia de la onda, mayor será la energía absorbida y menor la penetración del ultrasonido.

Recordemos que la resolución óptica de un sistema de imágenes por ultrasonidos será mayor a medida que la frecuencia del ultrasonido también lo sea. El fenómeno de absorción es el que limita la

frecuencia máxima que puede utilizarse en los equipos de imágenes por ultrasonidos. Este límite se encuentra entre 3MHz y 10 MHz. La absorción, que representa una desventaja para la utilización de los ultrasonidos en imágenes, es aprovechada para la producción de calor en músculos y articulaciones, especialmente en el tratamiento de atletas.

Efecto Doppler

En el bloque anterior se describieron las bases físicas involucradas en los procesos de obtención de imágenes por ultrasonidos. Otra aplicación igualmente importante es la medición de la velocidad de los fluidos en movimiento dentro del cuerpo, como la velocidad de la sangre en los vasos o el flujo aéreo en las vías respiratorias.

Por mucho, la forma más popular de medir velocidades a través de ondas acústicas se basa en el efecto Doppler, donde las ondas reflejadas por cuerpos en movimiento adquieren un corrimiento de su frecuencia en una cantidad proporcional a la velocidad del objeto. En la medición transcutánea de flujo sanguíneo, por ejemplo, los glóbulos rojos que fluyen con el plasma son los reflectores de las ondas.

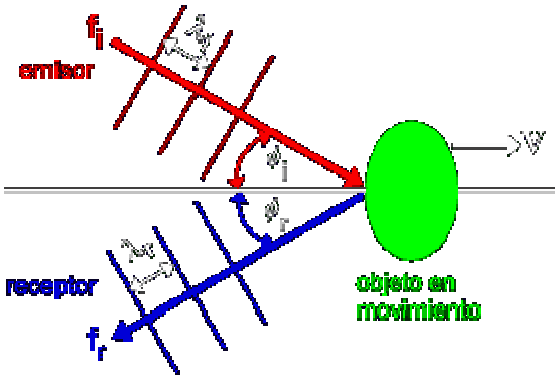


Figura 4.3.2.1 Esquema del efecto dopler

Para obtener la ecuación del corrimiento Doppler y así comprender mejor el fenómeno, consideremos una onda acústica producida por algún emisor estacionario (quieto) con una frecuencia f_i. Esta onda acústica incidente tiene una longitud de onda λ_i. El objeto dispersor se mueve con una velocidad V que forma un ángulo θ_i con respecto a la dirección de propagación. Después de la reflexión en la superficie móvil, la onda reflejada se propaga hacia atrás hacia un receptor, también estacionario, situado a un ángulo θ_r con respecto al movimiento del objeto. La frecuencia f_r de la onda reflejada será distinta de f_i. El valor de este corrimiento Doppler se obtiene en dos pasos: 1) Analizando el camino desde la fuente hasta el objeto. 2) Analizando el recorrido desde el objeto hasta el receptor.

1) Camino desde la fuente hasta el objeto en movimiento

Un observador ubicado sobre el objeto en movimiento puede detectar la onda cuando incide sobre el objeto, pero la velocidad de la onda respecto del objeto no será igual a la velocidad que traía cuando se propagaba en el medio circundante. Como consecuencia, la frecuencia f_m con que las ondas

alcanzan al objeto será diferente de la frecuencia enviada f_i. Si analizamos la geometría de la figura anterior, la onda incidente sobre el objeto tendrá una velocidad efectiva dada por

$$c_{in,ef} = c - V \cos \theta_i \qquad (1)$$

Así, la frecuencia efectiva con la que las ondas chocan contra el objeto se puede encontrar mediante la relación fundamental de las ondas:

$$f_m = c_{in,ef} / \lambda_i = (c - V \cos \theta_i) / \lambda_i \qquad (2)$$

Como en el medio donde se propaga la onda $\lambda_i = c / f_i$, la ecuación (2) queda

$$f_m = f_i \, . \, (1 - V \cos \theta_i / c) \qquad (3)$$

Notar que la diferencia entre f_m y f_i es proprcional a la relación entre la componente de la velocidad del objeto paralela a la dirección de propagación de la onda y la velocidad de la onda. También hay que destacar que si V = 0 y/o θ_i = 90º, las dos frecuencias serán iguales.

2) Camino desde el objeto en movimiento hacia el receptor

Las ondas acústicas incidentes sobre el objeto provocarán en su material oscilaciones a la frecuencia f_m. Estas oscilaciones provocarán una re-emisión de ondas desde el objeto, algunas de las cuales serán detectadas por el receptor estacionario.

Las ondas reflejadas con el ángulo θ_r de la gráfica serán irradiadas a la frecuencia f_m en el medio que se encuentra detrás del objeto en movimiento. Debido a este movimiento, la velocidad efectiva de la onda cuando abandona la superficie será

$$c_{out,ef} = c + V \cos \theta_i \qquad (4)$$

El movimiento relativo del objeto "estirará" las ondas re-irradiadas, si las comparamos con los valores que tendrían si el objeto estuviera quieto. La longitud de onda aumentada puede obtenerse utilizando nuevamente la ecuación fundamental de las ondas:

$$\lambda_r = c_{out,ef} / f_m = (c + V \cos \theta_r) / f_m \qquad (5)$$

Como consecuencia, la frecuencia con la cual la onda es detectada en el receptor será

$$f_r = c / \lambda_r = f_m / (1 + V \cos \theta_r / c) \qquad (6)$$

Notar que de nuevo la diferencia entre las dos frecuencias está referida a la relación entre la velocidad proyectada del objeto y la velocidad de propagación de la onda.

Ahora nos encontramos en condiciones de combinar los corrimientos de frecuencia obtenidos desde el análisis de los dos caminos recorridos por las ondas, eliminando f_m desde las ecuaciones (3) y (6). Esto nos brinda la relación entre las frecuencias del emisor y del receptor:

$$f_r = f_i \, . \, [\, (1 - V \cos \theta_i / c \,) \, / \, (1 + V \cos \theta_r / c \,) \,] \qquad (7)$$

La ecuación (7) puede simplificarse utilizando una aproximación que es válida para la mayoría de los casos prácticos:

$$V << c$$

Reduciéndose la ecuación (7) a

$$f_r = f_i \, . \, [\, 1 \, - \, V \cos \theta_i / c \, - \, V \cos \theta_r / c \,] \qquad (8)$$

El corrimiento Doppler, f_d, se define como la diferencia entre f_r y f_i:

$$f_d = f_r - f_i \qquad (9)$$

De la ecuación (8), obtenemos el resultado final:

$$f_d = - (V / c) \, . \, (\cos \theta_i \, + \, \cos \theta_r) \, . \, f_i \qquad (10)$$

La ecuación (10) para el corrimiento de frecuencia Doppler contiene algunas características interesantes. En particular, notar que el corrimiento es proporcional a f_i, de manera que cuanto mayor sea la frecuencia del emisor, mayor será la diferencia. También hay que advertir que $f_d = 0$ cuando V = 0, como se puede suponer intuitivamente.

Si se invierte la dirección de V, entonces el corrimiento Doppler se invertirá también en signo. En general, se puede decir que si el objeto se aleja de la fuente y del receptor, la onda reflejada será de menor frecuencia. Si el objeto se mueve hacia la fuente y el receptor la frecuencia detectada será mayor.

Propiedades acústicas de los tejidos

Para comprender mejor el estudio de las estructuras biológicas mediante ultrasonidos, es necesario hacer una breve revisión de las propiedades generales de los distintos tipos de tejidos. Este informe no es un intento de cubrir todos los detalles histológicos, más bien se describirán las características importantes para la comprensión de la propagación de las ondas acústicas.

La célula

El bloque constructor universal de todos los tejidos vivientes es la célula. Las células del cuerpo poseen una tremenda variedad de formas y medidas, aunque la mayoría tiene dimensiones entre 10 m y 100 m.

Cada célula es capaz de realizar los procesos vitales en forma independiente, incluyendo la reproducción (mitosis), la respiración y el metabolismo, la excresión de desechos, el crecimiento y la autoreparación.

El efecto de los ultrasonidos sobre las células depende de la potencia de aquellos. A bajos niveles de potencia (utilizados en diagnóstico), la célula experimenta poco trauma mecánico y cambio de temperatura. A niveles muy altos, la pared celular puede ser dañada o aún destruida, volcándose el contenido y destruyéndose la célula. También se pueden producir daños a los organelos. Si se mueren muchas células por la exposición a los ultrasonidos, el tejido puede no ser capaz de repararse suficientemente, ocurriendo un daño tisular severo. Esta situación, desde luego, debe ser evitada.

Las células se organizan en grupos que constituyen los diversos tejidos del cuerpo. Estos tejidos pueden clasificarse según su función en cuatro tipos básicos: epitelial, muscular, nervioso y conectivo.

Tejido epitelial

El tejido epitelial está formado por células que se ubican en capas para cubrir las diversas superficies del cuerpo. Las funciones fundamentales de este tejido son de protección, compartimentación y de regulación de secresiones y del intercambio de sustancias de los órganos que cubre.

Cómo las capas de tejido epitelial son relativamente delgadas, el tejido epitelial, normalmente, no es un factor importante en la determinación de las propiedades acústicas de la mayoría de las regiones del cuerpo. Los tejidos conectivo y muscular conforman un porcentaje mucho mayor del volumen de los órganos y vísceras.

Tejido muscular

Las células del tejido muscular son unidades alargadas llamadas fibras, que usualmente se encuentran organizadas en haces coherentes. La función del músculo es proveer movimiento o control de las partes del cuerpo mediante su contracción. El tejido muscular se clasifica en liso y estriado, dependiendo de su apariencia ante el microscopio óptico.

El músculo liso es controlado involuntariamente y se encuentra en las paredes del tracto digestivo, en los conductos de las glándulas y en las paredes de las arterias y venas. El músculo estriado se controla voluntariamente en el caso del músculo esquelético (utilizado para generar fuerza y locomoción) e involuntariamente en el caso del músculo cardíaco, que posee células musculares estriadas, especializadas para poder llevar a cabo la continua función cardíaca.

Los valores de densidad, velocidad de transmisión de las ondas sonoras, impedancia y atenuación del tejido muscular son todos mayores que los del agua y otros tejidos blandos. Dichos valores dependen un tanto de si la dirección de propagación de la onda es paralela o transversal al eje longitudinal de las fibras musculares. Por ejemplo, la atenuación de los ultrasonidos en la dirección paralela a las fibras es el doble de la encontrada en la dirección perpendicular.

Tejido nervioso

El tejido nervioso se utiliza en el cuerpo para el control y la comunicación. Las células nerviosas, llamadas neuronas, son usualmente largas y actúan como líneas de transmisión recogiendo, transmitendo y distribuyendo los impulsos nerviosos.

Se estiman unos 14 billones de células nerviosas en un ser humano, distribuidas en casi todas las partes del cuerpo para el control local o la comunicación con los centros nerviosos. Sin embargo, sólo en el cerebro y en la médula espinal la densidad de neuronas es suficiente como para afectar la propagación acústica. Por otro lado, los nervios (conjunto de axones; axón: proyección larga del citoplasma de la neurona que conduce los estímulos nerviosos) son delgados y se encuentran diseminados, no modificando apreciablemente las propiedades acústicas de los tejidos básicos (como el muscular) que ellos inervan.

Tejido conectivo

Esta es una amplia categoría encontrada a través de todo el cuerpo. El tejido conectivo llena muchos de los espacios entre los órganos, proveyendo soporte y conección de varias partes del cuerpo, como su nombre lo sugiere. También sirve como material por el cual otras células circulan, como el caso del plasma que transporta los eritrocitos, los glóbulos blancos y el resto de los componentes sanguíneos.

Una manera de clasificar al tejido conectivo es según su densidad y el tipo de material no-vivo encontrado en la sustancia extracelular.

- Tejido conectivo laxo: posee una estructura tipo red de células espaciadas (fibroblastos). Contiene fibras proteínicas, pero no tantas como el tejido conectivo denso. Se lo encuentra a menudo entre órganos y llenando espacios anatómicos.
- Tejido conectivo denso: está caracterizado por una gran abundancia de fibras organizadas para proveer resistencia o elasticidad según las necesidades. Los tendones y las cápsulas de los órganos son ejemplos de este tipo de tejido.
- Hueso: es un tipo de tejido conectivo en el que la sustancia extracelular se solidificó a través de un proceso de calcificación, haciendo al hueso fuerte y rígido. Su función principal es de soporte del esqueleto y de protección. El hueso también provee calcio para la regulación de este ión en la sangre y otros fluidos corporales. El hueso es más denso que los tejidos blandos, pero no demasiado (aproximadamente 1.7 veces) debido a la multitud de canales y espacios en su interior. Sin embargo, es mucho menos compresible que el tejido blando, lo que lleva a mayor velocidad de propagación de las ondas acústicas y mayor impedancia. la gran diferencia de impedancias en la interfase hueso-tejido blando resulta en un elevado coeficiente de reflexión, siéndole muy difícil a los ultrasonidos penetrar áreas óseas (como la cabeza).
- Sangre: es el fluido que lleva los nutrientes y quita los desechos de todas las regiones del cuerpo. Por medio de sus glóbulos rojos, o eritrocitos, la sangre transporta el oxígeno desde los pulmones hacia todos los tejidos para satisfacer los requerimientos metabólicos. Además, la sangre puede llevar calor desde o hacia una región, siendo un contribuyente importante para

la regulación térmica del organismo.

La porción líquida de la sangre, llamada plasma, está compuesta de agua con muchos electrolitos y proteínas disueltos. Una apreciable porción del total del volumen sanguíneo (cerca del 40%) está ocupada por los glóbulos rojos.
Los eritrocitos son células altamente especializadas para el transporte de oxígeno que poseen forma bicóncava, lo cual maximiza su relación superficie-volumen para incrementar el intercambio de gases. Acústicamente, los glóbulos rojos actúan como pequeños dispersores de los ultrasonidos, permitiendo así la medición de la velocidad del flujo sanguíneo a través del efecto Doppler. Debido a que el tamaño de los eritrocitos es mucho menor que la longitud de onda de los ultrasonidos, la dispersión de las ondas acústicas se clasifica como dispersión de Rayleigh (la célula absorbe la energía de la onda acústica y la re-emite como si fuera una antena).

4.3.4 Equipos de ultrasonido

A continuación se muestra una clasificación general de las configuraciones utilizadas en los equipos biomédicos de ultrasonidos. Los instrumentos basados en los modos A, B y C brindan información espacial sobre las regiones en estudio, mientras que el modo M y los dispositivos basados en el efecto Doppler aportan datos sobre movimiento y velocidad.

- Modo A (unidimensional)
- Modo B (bidimensional)
- Modo M (movimiento)
- modo C (utiliza la transmisión de los ultrasonidos, no las reflexiones)
- Doppler (velocidad, movimiento)
- Cirugía con ultrasonidos
- Microscopio ultrasónico
- Fisioterapia

Modo A

Este modo, como todas las otras configuraciones (excepto el modo C y el Doppler), está basado en la técnica de ecos, donde se emite un pulso de ultrasonidos desde un transductor hacia el interior de la región a estudiar. Las reflexiones en cada interfase entre tejidos son recibidas por el mismo transductor. El tiempo total desde el pulso inicial hasta el momento de la recepción del eco es proporcional a la profundidad de la interfase. Esto hace posible un mapeo unidimensional de las interfases entre tejidos a lo largo de la línea de propagación del rayo de ultrasonidos.

Lo que distingue al modo A de los otros métodos es la forma en que se muestran los resultados. En el modo A se grafican las amplitudes de los ecos recibidos. Es decir, un voltaje proporcional a los mismos, proporcionado por el transductor, se muestra en una pantalla similar a la de un osciloscopio.

Figura 4.3.4.1 Grafica obtenida por un transductor ultrasónico tipo A

La ventaja del modo A es que brinda información posicional de una manera rápida con un equipamiento mínimo. Su desventaja es que sólo ofrece información unidimensional.

Se lo utiliza en electroencefalografía para la detección de la línea media cerebral que, en personas sanas, se encuentra ubicada en el centro del cráneo en el plano sagital medio. También se utiliza el modo A en oftalmología para la determinación del tamaño de las estructuras del ojo.

Modo B

El desarrollo de este modo puede comprenderse mejor considerando una línea de modo A modificada de tal manera que la amplitud del eco no causa un desplazamiento vertical del tubo de rayos catódicos del osciloscopio, produciendo en cambio un aumento o decremento proporcional en el brillo (dado por la cantidad de electrones que colisionan contra la pantalla fluorescente del osciloscopio). Es decir, los ecos modulan el brillo de los puntos de la pantalla.

La modulación del brillo libera un eje de la gráfica para la presentación de otra información. El eje a lo largo de la dirección del rayo aún corresponde a la profundidad de penetración o distancia; pero en el modo B, el eje perpendicular al rayo se utiliza para mostrar distancia, relacionando la dirección del rayo de ultrasonidos con la de la línea que se imprime en la pantalla. Esto se realiza con dispositivos llamados transductores de posición.

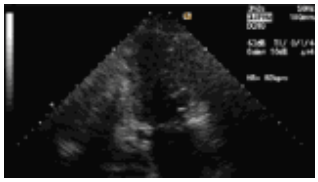


Figura 4.3.4.2 Grafica obtenida por un transductor ultrasónico tipo B

Esta imagen muestra una ecografía por ultrasonidos realizada mediante el modo B. La barra vertical en degradé de la parte superior izquierda indica la escala de grises utilizada en la imagen. En la parte superior derecha aparece información alfanumérica sobre los datos del paciente y los parámetros seteados en el equipo. En el centro queda la imagen de modo B. La parte superior es por donde ingresan los rayos de ultrasonidos al cuerpo. Cada línea radial es equivalente a una línea de modo A (cambiando, como se dijo, amplitud por brillo). La imagen completa se obtiene moviendo el transductor (manual o automáticamente) en todas las direcciones radiales de interés, formando la típica imagen con forma de abanico.

Los instrumentos de modo B representan hoy por hoy la mayoría de los dispositivos utilizados clínicamente. Esto se debe a la gran variedad de regiones anatómicas que pueden ser escaneadas exitosamente y a la fácil interpretación del mapa bidimensional de tejidos que produce.

La principal aplicación es en obstetricia, donde se puede registrar el crecimiento fetal, la orientación y las anormalidades, sin el riesgo que podrían producirle los rayos X al feto. También se pueden estudiar la ubicación de la placenta y los casos de embarazos múltiples.

En el ámbito ginecológico, mediante el modo B se pueden detectar quistes o tumores en los ovarios. A nivel abdominal, se pueden obtener claras imágenes del hígado, del bazo, de la vesícula biliar y de los riñones.

Los pechos femeninos son otra región que puede ser analizada mediante ultrasonidos. Puede revelarse la presencia de quistes o tumores mediante un procedimiento seguro.

También pueden realizarse estudios cardíacos, pero aquí surge un problema debido a que las "observaciones" deben hacerse a través de los espacios entre las costillas, obteniéndose imágenes seccionadas (ventanas) del corazón.

Modo M

Esta configuración se utiliza para analizar cualitativa y cuantitativamente el movimiento de las estructuras del cuerpo, como las válvulas del corazón. Este modo es un híbrido de los modos A y B. Como en el modo B, el brillo de cada línea es modulado de acuerdo a la amplitud de los ecos recibidos. Sin embargo, se parece al modo A en el sentido que los ecos son recogidos en una sola dirección, a lo largo del recorrido del rayo. Estas señales son presentadas en el eje horizontal del monitor.

La deflexión vertical de la pantalla se produce lentamente, de manera que las líneas sucesivas son inscriptas en orden progresivo (como lo hace una impresora). Cualquier movimiento de un objeto a lo largo del camino del rayo presentará un desplazamiento horizontal del eco registrado en las líneas sucesivas.

El tiempo en que se recorre una línea horizontal es el mismo al encontrado en los modos A y B (13 μs por cada centímetro de profundidad). El barrido vertical es mucho más lento, entre 2 y 3 segundos para cubrir toda la pantalla, de manera que pueden mostrarse varios ciclos cardíacos.

Debido a que el eje horizontal de la pantalla está calibrado en términos de profundidad, el desplazamiento espacial neto del objeto móvil se puede medir directamente. Como el eje vertical está dado en unidades de tiempo, se puede medir la velocidad cuantitativamente desde el monitor, en mm/s.

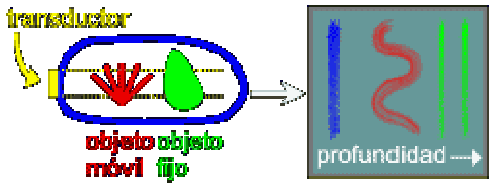


Figura 4.3.4.3 Profundidades en tejido medido con ultrasonido de tipo B

En la figura se muestra un diagrama explicativo del funcionamiento del modo M. El transductor emite las ondas que serán reflejadas por las interfases entre los tejidos. A la derecha se ve una

representación de la ecografía de modo M correspondiente. Los colores se utilizan simplemente para relacionar los ecos con las distintas estructuras. Los equipos clínicos de modo M son monocromáticos.

Modo C

Este método difiere de los anteriormente descriptos en el sentido que no utiliza los ecos de las ondas reflejadas en las interfases entre los tejidos. En cambio, el modo C aprovecha las ondas transmitidas. El emisor de ultrasonidos se coloca sobre el objeto a estudiar y el receptor en el extremo opuesto.

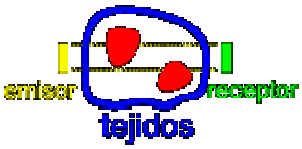


Figura 4.3.4.4 Profundidades en tejidos medidos con ultrasonido tipo C

La exploración va acompañada de una translación de los dos transductores (unidos mecánicamente entre sí). Debido a que el movimiento de barrido se realiza en un plano perpendicular al rayo (en la gráfica anterior, el emisor y el receptor se moverían hacia arriba o hacia abajo), se puede obtener una imagen bidimensional similar a la producida por los rayos X convencionales.

Pueden determinarse dos características de los tejidos mediante los scanners C. En primer lugar, comparando la amplitud del pulso recibido con la amplitud del pulso emitido, se puede lograr una medida de la atenuación total a lo largo del camino recorrido. Esta atenuación no es debida sólo a la absorción de los tejidos, sino también a las reflexiones en las interfases. En segundo lugar, comparando el tiempo entre la emisión del pulso y su recepción (tiempo de vuelo), se obtendrán los datos necesarios para calcular la velocidad de propagación acústica en el tejido.

Como el pulso de ultrasonidos debe atravesar toda la distancia a través del tejido, las regiones de anatomía compleja son difíciles de estudiar, debido a las múltiples pérdidas por reflexión.

El mayor éxito de los scanners C es la obtención de imágenes de partes anatómicas relativamente homogéneas, como los pechos femeninos en la dirección lateral. Además, actualmente se están utilizando tomógrafos por ultrasonidos en base al modo C, obteniendo registros en múltiples direcciones y "fabricando" la imagen a través de un algoritmo de reconstrucción por computadora.

Instrumentos basados en el efecto dopler

En la sección bases físicas se explicó el efecto Doppler y sus posibles aplicaciones, todas relacionadas con el análisis de estructuras en movimiento. Aquí se desarrollará un resumen de los instrumentos basados en dicho principio.

En su forma básica, el transmisor de los flujímetros Doppler es excitado continuamente con un voltaje sinusoidal. La señal del transductor de recepción es mezclada con una porción de la onda transmitida en un dispositivo no lineal (como puede ser un diodo)

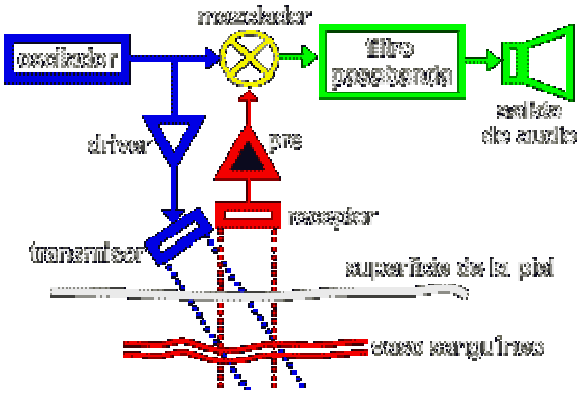


Figura 4.3.5 Diagrama a bloques de instrumentación de un equipo de ultrasonido

El contenido frecuencial a la salida del diodo mezclador puede hallarse considerando la teoría de la mezcla (mixing theory), en la cual dos ondas senoidales de igual o distinta frecuencia son mezcladas y luego pasadas a través de un dispositivo de comportamiento no lineal. La salida del mezclador contendrá varias componentes senoidales que representarán todas las posibles combinaciones de suma y diferencia de las señales originales. Para los propósitos de este instrumento, sólo se retendrá, mediante un filtro pasabanda, la diferencia de frecuencias (f_d).

Si el espectro filtrado simplemente se amplifica y se lo reproduce mediante un parlante o auriculares, la información Doppler se presentará al usuario como un tono audible. Esto se debe a una circunstancia afortunada: el rango de frecuencias Doppler para la mayoría de flujos corporales, cuando la frecuencia incidente está entre 2 MHz y 10 MHz, cae dentro de la porción audible del espectro sonoro, desde 20 Hz hasta 10 KHz, aproximadamente. No obstante, este método sólo ofrece información cualitativa que un operador experimentado podría relacionarla con velocidades. Además, de esta manera no se puede saber la dirección de circulación del flujo.

Existen dos métodos cuantitativos, que además indican la dirección del flujo. Uno trabaja en base a tratamientos analógicos de la señal mezclada. El otro calcula, mediante el algoritmo de la transformada rápida de Fourier (FFT), las componentes espectrales de la señal mezclada y las muestra en un monitor.

Los flujímetros Doppler descriptos hasta aquí proveen sólo información espacial parcial sobre las posiciones de los objetos móviles. Existen varias situaciones, sin embargo, donde una resolución espacial más precisa sería de un importante valor clínico. Por ejemplo, en una rama arterial o en otro sitio donde más de un vaso pueda encontrarse al mismo tiempo en el campo de observación. En estos caso, es importante diferenciar las contribuciones de flujo individuales.

Por otro lado, la frecuencia Doppler es básicamente una medida de la velocidad de la sangre, no de su flujo volumétrico. Para convertir velocidad en flujo, deben conocerse el área transversal de los vasos y el perfil de velocidades.

Se han desarrollado varios instrumentos para proveer información espacial, como los scanners transversales, los flujímetros Doppler pulsados y los scanners dúplex. Ésta es una de las áreas de mayor crecimiento de la instrumentación por ultrasonidos.

Cirugía con ultrasonidos

La acción mecánica de la vibración molecular que acompaña al pasaje de ultrasonidos puede en algunas circunstancias ser utilizada para interrumpir el desarrollo o extraer tejidos o partículas no deseados. Para que el proceso sea efectivo, la potencia emitida debe ser alta, evitándose el daño a los tejidos sanos vecinos.

Por ejemplo, las cataratas en el cristalino del ojo pueden ser fragmentadas y removidas mediante una aguja que vibra a unos 40 KHz. Los fragmentos son aspirados por succión a través del tubo central de la aguja. Las ventajas de esta técnica sobre la cirugía convencional son las pequeñas incisiones y que la córnea y la cámara anterior del ojo son menos perturbadas.

Un enfoque similar ha tenido éxito en la ruptura de cálculos renales y del tracto urinario superior. Cuando las piedras en dichas áreas adquieren el tamaño suficiente para no pasar a través de los uréteres, deben ser removidas quirúrgicamente. Una alternativa a la cirugía abierta es el acceso percutáneo a los cálculos por medio de una sonda a través de la piel. Un endoscopio de fibra óptica (nefroscopio) se inserta en la sonda para permitir la visión de las piedras que son sacadas a través del agujero interior.

Si se encuentra un cálculo demasiado grande para pasar por el túnel de la sonda, debe romperse (proceso llamado litotripsia). Esto se puede llevar a cabo con ultrasonidos, aplicados a través de una aguja hueca, unida a un transductor externo, que se inserta a través de la sonda. En este caso se utilizan vibraciones de unos 25 KHz.

Microscopio ultrasónico

La resolución límite de los instrumentos de imágenes, incluyendo los microscopios, depende de la longitud de onda de la iluminación utilizada. Los microscopios ópticos buenos pueden diferenciar estructuras de poco menos de 1 μm. Para mejores resoluciones, debe utilizarse microscopía electrónica. Con electrones de alta energía puede llegarse a resoluciones del orden de los nanometros (10^{-9} m).

Se ha desarrollado un microscopio que utiliza ondas acústicas. Como las muestras observadas generalmente son pequeñas, la absorción que producen al ser atravesadas por el rayo de ultrasonidos será pequeña. Por ende, se pueden emplear frecuencias ultra-altas para mejorar la resolución del instrumento.

Actualmente se aplican frecuencias de hasta 1500 MHz. La longitud de onda y la resolución aproximada en el agua a esta frecuencia es de 1 μm. Entonces, si la resolución del microscopio ultrasónico no es mejor que la de los microscopios ópticos, ¿Qué espacio podrían tener estos instrumentos en el mercado?

En primer lugar, utiliza el medio acuoso natural que rodea a la muestra para facilitar el acoplamiento acústico, de esta forma las ondas sonoras no degradan la muestra, pudiéndose observar materiales vivos, como las células, para seguir su desplazamiento, y analizar propiedades de los tejidos.

Más importante aún, las imágenes del microscopio acústico muestran características diferentes de las reveladas por los microscopios ópticos y electrónicos. Cuando se lo utiliza en el modo de

reflexión, es sensible a los patrones de fase acústica y a la impedancia de las interfases dentro del objeto. En el modo de transmisión, brinda imágenes de atenuación y refracción. En pocas palabras, el miscroscopio ultrasónico entrega una vista distinta del mundo que nos rodea.

Fisioterapia

La Fisioterapia es el conjunto de métodos y técnicas, que mediante la aplicación de medios físicos, curan, previenen y adaptan a personas discapacitadas o afectadas de disfunciones psicosomáticas, somáticas y orgánicas o a las que desean mantener un nivel adecuado de salud.

Los fisioterapeutas emplean los ultrasonidos en múltiples patologías por su acción fibrolítica, térmica y vasodilatadora local (antiinflamatoria y antiálgica) y simpaticolítica. Es decir, en reumatismos articulares, neuralgias, algias traumáticas, nódulos celulíticos, artrosis, edemas, etc. *(información aportada por la Fisioterapeuta Mónica Borrás)*.

5. IMAGENES BIOMEDICAS APLICACIÓN CON HERRAMIENTAS SOFTWARE

5.1 Segmentación de imágenes abdominales de Resonancia Magnética y Tomografía Axial Computerizada

5.1.1 Las herramientas software para el análisis de imágenes médicas

Las herramientas software para el análisis de imágenes médicas suponen una gran ayuda para el personal médico a la hora del diagnóstico, tratamiento y monitorización de los pacientes. Los dos problemas a los que nos enfrentamos al elaborar estas herramientas son el registro, o el hecho de establecer correspondencias entre las imágenes, y la segmentación, que permite determinar las estructuras
anatómicas en una imagen. Este proyecto se ha centrado principalmente en la segmentación, en un intento de determinar el hígado dentro de imágenes abdominales procedentes de RM o TAC.

Con este fin se ha intentado implementar una aplicación gráfica "Open Source" que de una manera automática asista al médico en la determinación del hígado dentro de una imagen abdominal. Para desarrollar esta aplicación se ha creado un algoritmo que
tiene en cuenta dos aspectos del hígado: por un lado el hecho de que es el órgano más grande presente en una imagen abdominal y por otro la homogeneidad en el nivel de gris de todos los voxels que forman parte de este órgano. Este algoritmo proporciona un contorno inicial del hígado que es mejorado mediante las modernas técnicas de Level Set.

La implementación de este algoritmo se ha realizado en su mayoría con las librerías de ITK, pertenecientes al proyecto "Visible Human". Para la realización del entorno gráfico nos hemos basado en el programa ITK-Snap, realizado a partir de las
librerías FLTK, para el desarrollo de ventanas gráficas, y VTK, para la visualización y renderizado de imágenes. Todas estas librerías son de software libre. Los primeros prototipos de la aplicación han sido llevados a cabo mediante Matlab.
En un futuro próximo se pretende añadir a esta aplicación nuevas funcionalidades que permitan la determinación de enfermedades del hígado y su seguimiento.

5.1.1.1. ITK

Las Insight Toolkit (también conocidas como ITK) son una serie de librerías de software libre para el registro y la segmentación de imágenes. La segmentación es el proceso por el que se identifican y se clasifican los datos encontrados en una representación digital muestreada, típicamente el resultado de un TAC o una resonancia magnética. El registro es la búsqueda de correspondencias entre los datos extraídos. Por ejemplo, en el entorno médico, una resonancia magnética debe ser alineada con un TAC con el fin de obtener información contenida en la combinación de ambos y que no se puede obtener de las imágenes por separado.

Las ITK se encuentran implementadas en C++. Pueden ser empleadas en cualquier plataforma, para

lo que requieren de CMake para configurar sus opciones de compilación. Además mediante aplicaciones como CableSwig podemos convertir automáticamente desde C++ a Tcl, Java o Python y viceversa. Esta característica permite que las aplicaciones basadas en ITK puedan ser desarrolladas en múltiples lenguajes de programación. La implementación en C++ de las ITK se ha realizado lo más genéricamente posible usando clases templatizadas. Este uso de clases templatizadas permite que el código sea muy eficiente y que la mayor parte de los errores sean descubiertos en tiempo de compilación en vez de en tiempo de ejecución.

Como ITK es un proyecto de software libre, los desarrolladores de todo el mundo pueden usarlo, modificarlo y extender sus aplicaciones. ITK usa un modelo de programación conocido como programación extrema. La programación extrema convierte el proceso de programación usual en un proceso iterativo y simultáneo de diseño, implementación, testeo y lanzamiento. Los puntos clave de esta programación extrema deben ser la comunicación y el testeo. La comunicación entre los distintos miembros de la comunidad ITK permite la rápida evolución del software, mientras que el testeo mantiene el software estable.

Estas librerías han sido desarrolladas por seis principales organizaciones, pudiéndose encontrar entre ellas tanto organizaciones comerciales como académicas. La financiación del proyecto ha ido a cargo de la National Library of Medicine del Instituto Nacional de Salud estadounidense.

5.1.1.2 VTK

Las Visualization Toolkit (VTK) son un conjunto de librerías de software libre, desarrolladas por Kitware, para la computación de gráficos 3D, el procesamiento de imágenes y la visualización. Las VTK son librerías implementadas en C++, aunque permiten la programación con Tcl/Tk, Java y Python mediante interfaces. VTK funciona sobre plataformas basadas en UNIX, Windows 95/98/NT/2000/XP y Mac OS X Jaguar y posteriores. El diseño y la implementación de las librerías se ha basado en los principios de la programación orientada a objetos.

El modelo gráfico de las VTK está basado en un nivel de abstracción mucho mayor que otras librerías de renderizado como pueden ser OpenGL o PEX. Esto significa que es mucho más fácil crear aplicaciones gráficas y de visualización. Las aplicaciones sobre las VTK pueden ser escritas directamente en C++, Tcl, Java o Python. De hecho, utilizando Tcl o Python con Tk, o incluso Java con sus librerías GUI, se pueden crear aplicaciones muy rápidas.

Las VTK proporcionan un sistema de verdadera visualización que no permite visualizar la geometría. Soporta una amplia variedad de algoritmos de visualización, incluyendo los métodos escalar, vectorial, tensorial, de textura y volumétrico; y avanzadas técnicas para modelar como mesh smoothing, contouring, polygon reduction y la triangulación de Delaunay. Además, incorpora docenas de

algoritmos para imágenes que permiten mezclar imágenes 2D con algoritmos de gráficos 3D.

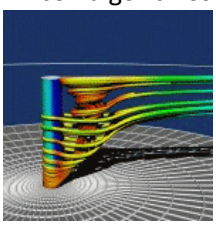


Figura 5.1.1.2, Mezcla de graficas 2D y 3D

El objetivo de estas librerías es obtener un software lo más fácil de utilizar, para que cualquier persona con unos mínimos conocimientos pueda emplearlo. Por último mencionar que aunque las VTK están disponibles gratuitamente, se puede obtener soporte comercial de Kitware.

5.1.1.3. FLTK

Las Fast Light Tool Kit (FLTK) son una serie de librerías multiplataforma programadas en C++ para el desarrollo de aplicaciones GUI. Se encuentran bajo licencia GPL. Funcionan tanto en UNIX/Linux, como sobre Microsoft Windows y Mac OS X. Las FLTK proporcionan funcionalidades GUI modernas, así como, soporte de gráficos 3D a través de OpenGL y las GLUT. Fue originalmente desarrollado por Bill Spitzak y actualmente, cualquier usuario con unos mínimos conocimientos puede dedicarse a su mantenimiento y desarrollo.

5.1.1.4. ITK Snap

ITK-Snap es una aplicación software, bajo licencia GPL, desarrollada por Cognitiva Corporation en colaboración con la Medical Library of Medicine, que permite la segmentación de órganos del cuerpo humano a partir de imágenes médicas en tres dimensiones. Esta aplicación permite segmentación semiautomática a partir de técnicas basadas en contornos activos, así como navegación visual a través de la imagen. Además de estas funciones básicas ITK-Snap también proporciona otra serie de utilidades:

- Cursor para facilitar la navegación en tres dimensiones.
- Segmentación manual simultánea en tres planos ortogonales.
- Interfaz de usuario simplificada para la elección de los parámetros del método de los contornos activos.
- Soporte para diferentes formatos de imágenes 3D.
- Gráfico 3D que muestra la posición de los planos de corte para facilitar el post-procesado de las imágenes segmentadas.
- Tutorial sobre el uso de ITK-Snap.
- LiverSegm

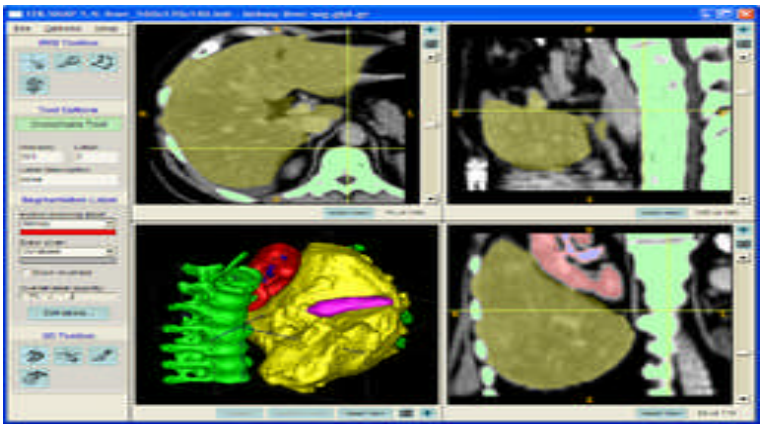


Figura 5.1.1.4, Mezcla de graficas 2D y 3D

Liver-Segm es la aplicación que se ha desarrollado durante la ejecución del proyecto. Se ha basado en el entorno ITK-Snap. Se han mantenido todas las herramientas que ya presentaba éste y se ha añadido una nueva herramienta para la segmentación automática de hígados.

Para su elaboración se ha contado con RM y TAC procedentes del Hospital Clínico San Carlos. Es también en este hospital donde se está trabajando con las primeras versiones del programa.

En próximas ediciones del programa se pretende que éste sea capaz de detectar lesiones hepáticas y permitir un seguimiento de las mismas.

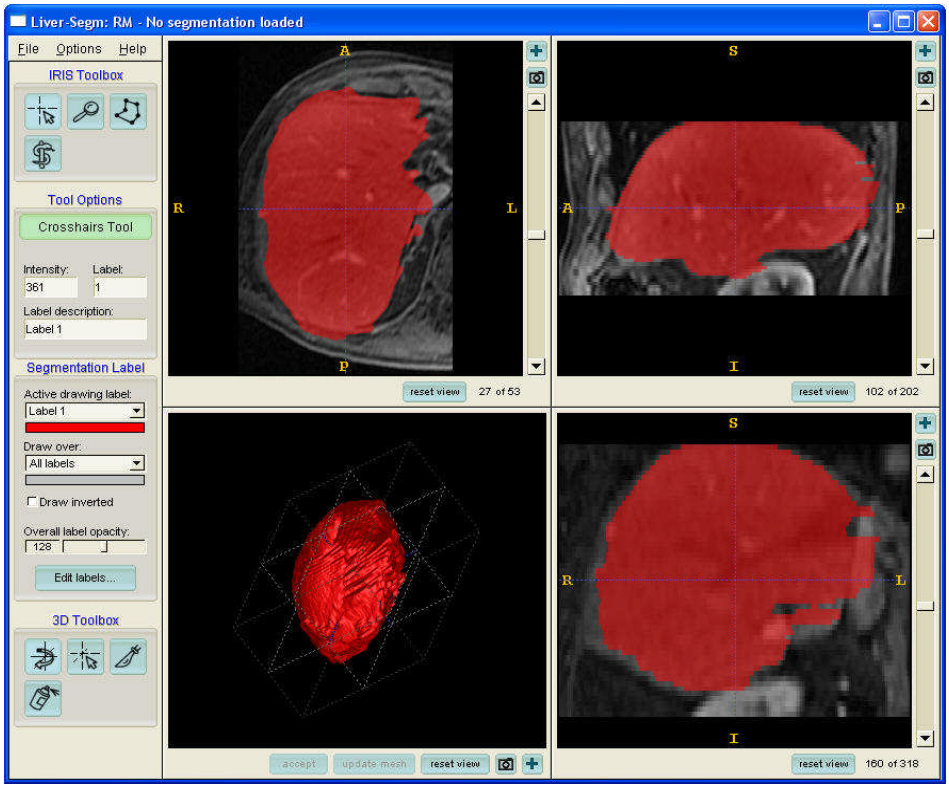


Figura 5.1.2.2, Mezcla de graficas 2D y 3D aplicación Snap

5.2. Integración de librerías ITK en MATLAB mediante objetos MEX para la segmentación del hígado procedente de imágenes de CT en entornos multiplataforma

Resumen

La resonancia magnética y la tomografía axial computarizada son métodos de obtención de imágenes médicas ampliamente usados para el diagnóstico de lesiones hepáticas y el seguimiento de las respuestas terapéuticas de dichas lesiones. La incorporación de herramientas informáticas que faciliten y mejoren el procesamiento de las imágenes médicas supone un gran avance para el diagnóstico y el tratamiento de numerosas patologías.

El objetivo de este proyecto es mejorar los resultados y tiempos de ejecución de Liver-Segm, una aplicación de código libre usada para realizar la segmentación automática del hígado. En este sentido, se ha optado por una arquitectura cliente – servidor que permita agilizar el procesamiento. Para la programación se han utilizado las librerías ITK, pertenecientes al proyecto *Visible Human*, que proporcionan una interfaz altamente optimizada para el tratamiento de imágenes biomédicas. Asimismo, se ha mejorado la portabilidad del programa, permitiendo su uso en diferentes plataformas.

La segmentación propuesta está basada en un procesado de difusión anisotrópica 3D adaptativo y carente de parámetros de control. A la imagen realzada se le aplica una combinación de técnicas de detección de bordes 3D, análisis del histograma, postprocesado morfológico y evolución de un contorno activo 3D. Éste último fusiona información de apariencia y forma del hígado.

Las futuras líneas de trabajo pasan por mejorar los resultados y tiempos de ejecución de la segmentación del hígado. Asimismo, se pretende incorporar una nueva herramienta que permita estudiar la evolución de las lesiones hepáticas.

5.2.1. Liver-Segm en Ubuntu 64 Bits

Ubuntu

Ubuntu es una distribución GNU/Linux que ofrece un sistema operativo orientado principalmente a ordenadores personales, aunque también proporciona soporte para servidores. Es una de las distribuciones de GNU/Linux más importantes a nivel mundial. Se basa en Debian GNU/Linux y sus principales objetivos se basan en la facilidad y libertad de uso, la fluida instalación y los lanzamientos regulares. El principal patrocinador es Canonical Ltd., una empresa privada fundada y financiada por el empresario sudafricano Mark Shuttleworth.

Existen diversas variantes de Ubuntu disponibles, las cuales poseen lanzamientos simultáneos con Ubuntu.

CMake

CMake es un sistema *open-source* que permite administrar el proceso de construcción de una aplicación informática independientemente del sistema operativo y compilador elegido. CMake ha sido diseñado para ser usado en conjunción con el programa nativo con el que se vaya a compilar la aplicación. Una serie de simples ficheros en cada directorio fuente (llamados ficheros CMakeLists)

permiten generar ficheros estándar de construcción del software (makefiles, workspaces de Visual Studio, etc.) que serán tratados como ficheros normales. CMake puede compilar código fuente, crear librerías o construir ejecutables, además soporta construcciones de bibliotecas tanto estática como dinámicamente.

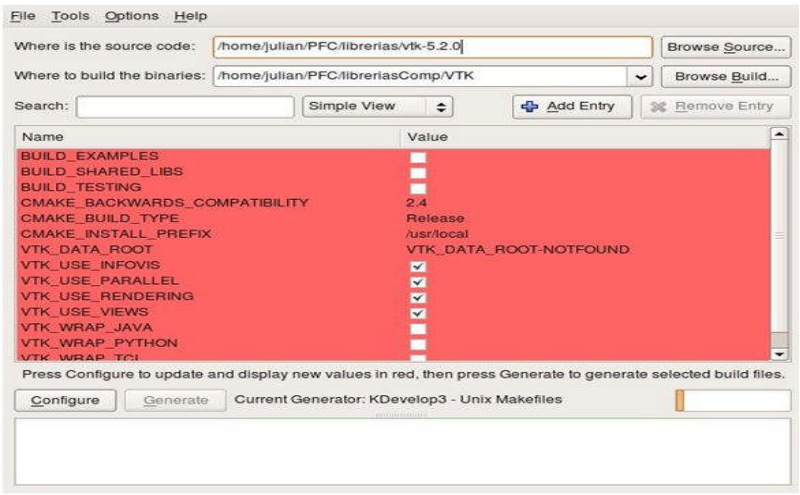
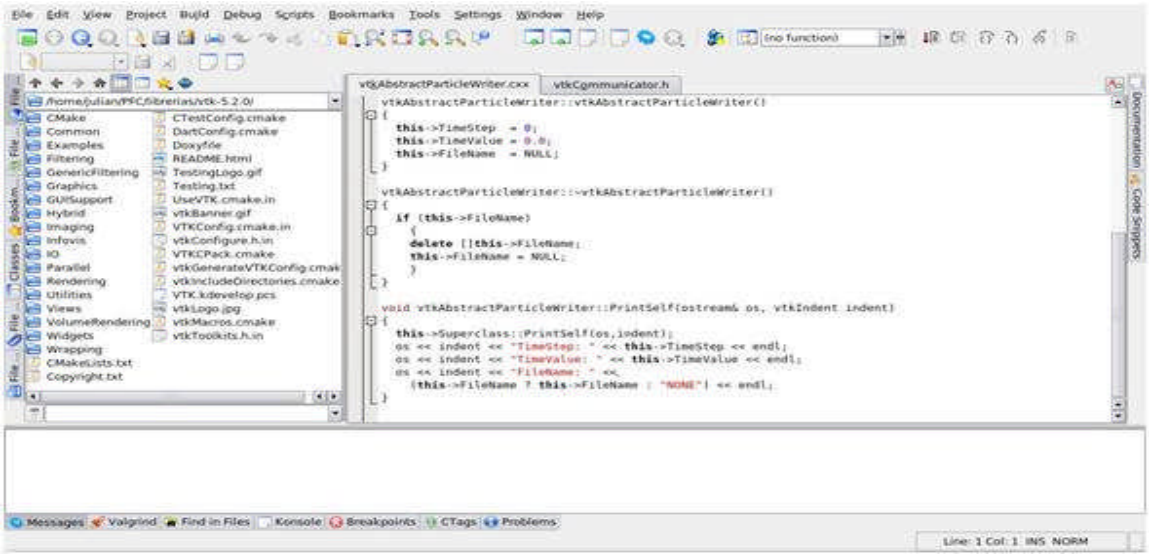


Figura 5.2.1.1, Librerías multiplataforma

KDevelop

KDevelop es un entorno de desarrollo integrado para sistemas GNU/Linux y otros sistemas Unix, publicado bajo licencia GPL, y orientado al uso bajo el entorno gráfico KDE, aunque también funciona con otros entornos, como Gnome. A diferencia de muchas otras interfaces de desarrollo, KDevelop no cuenta con un compilador propio, por lo que depende de gcc para producir código binario.

Su última versión se encuentra actualmente en desarrollo y funciona con distintos lenguajes de programación como C, C++, Java, SQL, Python, Perl y Pascal, así como guiones para el intérprete de comandos Bash.



Librerías necesarias

- Las VTK son un conjunto de librerías de software libre, desarrolladas por Kitware, para la computación de gráficos 3D, el procesamiento de imágenes y la visualización.
- Las FLTK son una serie de librerías multiplataforma programadas en C++ para el desarrollo de aplicaciones GUI.
- Las ITK son una serie de librerías de software libre para el registro y la segmentación de imágenes.

Liver-Segm

Liver-Segm es la aplicación desarrollada por el grupo de investigación de Visión Artificial (GVA). Esta aplicación está basada en ITK-SNAP, la cual es una aplicación software *Open Source* utilizada para segmentar estructuras en 3D de imágenes médicas. Utiliza las librerías ITK para la segmentación de las imágenes, VTK para el renderizado y FLTK para el entorno gráfico.

Se ha modificado la aplicación ITK-SNAP introduciendo los algoritmos para la segmentación del hígado y de las lesiones hepáticas, utilizando las librerías ITK.

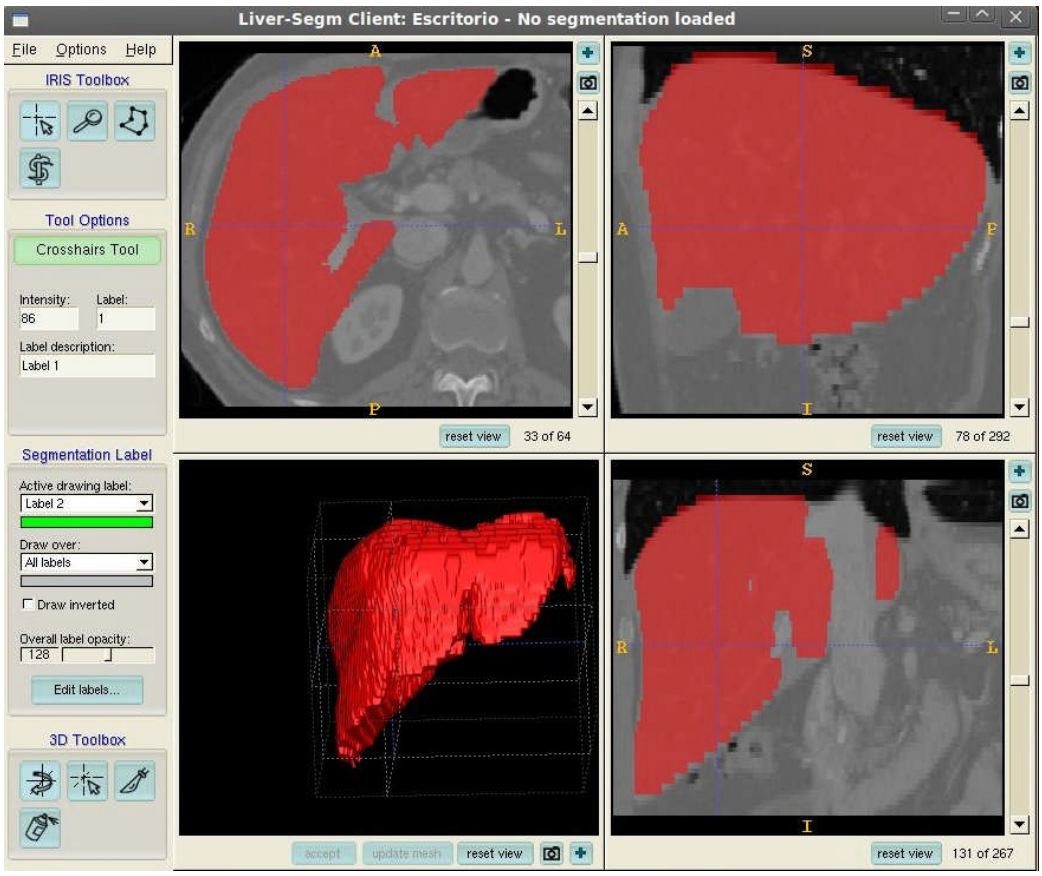


Figura 5.2.3, Mezcla de graficas 2D y 3D con Liver-segm segmento de hígado

5.2.2 ITK desde MATLAB

MATLAB

MATLAB (abreviatura de *MATrix LABoratory*, "laboratorio de matrices") es un software matemático que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M). Está disponible para las plataformas Unix, Windows y Apple Mac OS X.
Entre sus prestaciones básicas se hallan: la manipulación de matrices, la representación de datos y funciones, la implementación de algoritmos, la creación de interfaces de usuario (GUI) y la comunicación con programas en otros lenguajes y con otros dispositivos hardware. El paquete MATLAB dispone de dos herramientas adicionales que aumentan sus prestaciones, a saber: Simulink (plataforma de simulación multidominio) y GUIDE (editor de interfaces de usuario - GUI). Además, se pueden ampliar las capacidades de MATLAB mediante las cajas de herramientas (*toolboxes*) y las de Simulink con los paquetes de bloques (*blocksets*).
Las funcionalidades de MATLAB se agrupan en más de 35 *toolboxes*, clasificadas en las siguientes categorías:

- Matemáticas y optimización: Optimization Toolbox, Symbolic Math Toolbox, Partial Differential, Equation Toolbox y Genetic Algorithm and Direct Search Toolbox.

- Estadística y análisis de datos: Statistics Toolbox, Neural Network Toolbox, Curve Fitting Toolbox, Spline Toolbox y Model-Based Calibration Toolbox.
- Diseño de sistemas de control y análisis: Control System Toolbox, System Identification Toolbox, Fuzzy Logic Toolbox, Robust Control Toolbox, Model Predictive Control Toolbox y Aerospace Toolbox.
- Procesado de señal y comunicaciones: Signal Processing Toolbox, Communications Toolbox, Filter Design Toolbox, Filter Design HDL Coder, Wavelet Toolbox, Fixed-Point Toolbox y RF Toolbox.
- Procesado de imagen: Image Processing Toolbox, Image Acquisition Toolbox y Mapping Toolbox.
- Pruebas y medidas: Data Acquisition Toolbox, Instrument Control Toolbox, Image Acquisition Toolbox, SystemTest, OPC Toolbox y Vehicle Network Toolbox.
- Biología computacional: Bioinformatics Toolbox y SimBiology.
- Modelado y análisis financiero: Financial Toolbox, Financial Derivatives Toolbox, Datafeed Toolbox, Fixed-Income Toolbox y Econometrics Toolbox.
- Desarrollo de aplicaciones: MATLAB Compiler, Spreadsheet Link EX (para Microsoft Excel), MATLAB Builder NE (para Microsoft .NET Framework) y MATLAB Builder JA (para Java).
- Informes y conexión a bases de datos: Database Toolbox y MATLAB Report Generator.

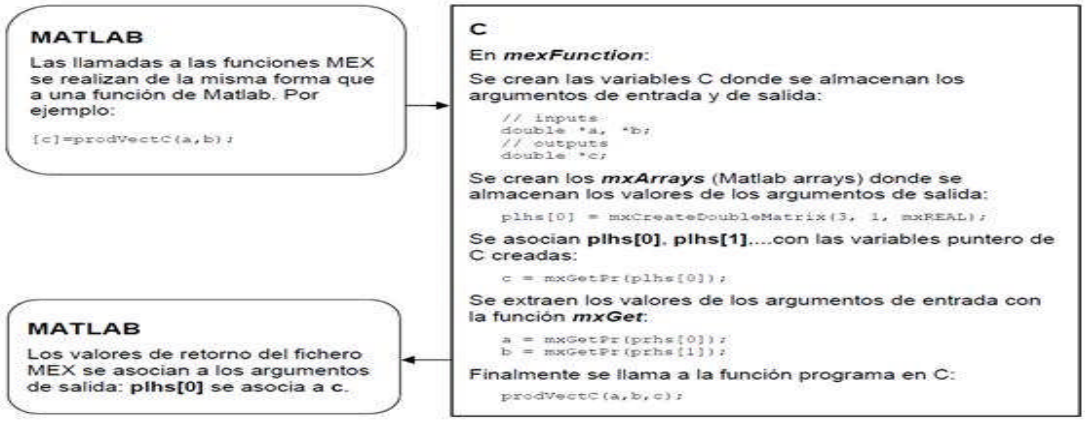
Comunicación entre MATLAB y C++

Una vez se tenga el código de la función de MATLAB escrito en C++, se debe añadir en el archivo la definición de la función que convierte los datos de MATLAB a C++ y viceversa, es decir, se debe rellenar la interfaz entre MATLAB y C++.

La MexFunction es la encargada de traducir los datos de MATLAB a C++. La declaración de MexFunction es la misma para todos los programas y consta de 4 parámetros de entrada:

- nlhs (*left hand side argument*), que es el número de salidas en la llamada desde MATLAB.
- *plhs. Con este puntero se puede acceder a cada salida en MATLAB.
- nrhs (*right hand side argument*), que es el número de entradas en la llamada desde MATLAB.
- *prhs. Con este puntero se puede acceder a cada entrada en MATLAB, es de 'sólo lectura'.

Todos los tipos de variables de MATLAB (escalares, vectores, matrices, cadenas de caracteres, estructuras, vectores de celdas, etc.) son *mxArrays*. Para cada *mxArray*, MATLAB almacena el tipo, las dimensiones, los datos, si es real o complejo (para datos numéricos), el número de campos y sus nombres para las estructuras, etc. MATLAB dispone de un gran número de funciones C para trabajar con *mxArrays*, que pueden encontrarse buscando "*MX Array Manipulation (C)*" en la ayuda. Algunas de estas funciones se deben utilizar para construir la MexFunction.



5.2.3 Liver Segm Server C++

Para la segmentación propuesta, primero se realiza una expansión del histograma de las imágenes tomadas por RM. Después, se realiza un procesado de difusión anisotrópica adaptativo sin parámetros de control en tres dimensiones para la eliminación del ruido de la imagen (*denoising*). A la imagen realzada se le aplica un análisis del histograma y técnicas de detección de bordes en tres dimensiones basadas en el detector de Canny. Posteriormente, se procede a la etapa del post-procesado morfológico y a la de evolución del contorno activo en tres dimensiones. Éste último fusiona información de apariencia y forma del hígado.

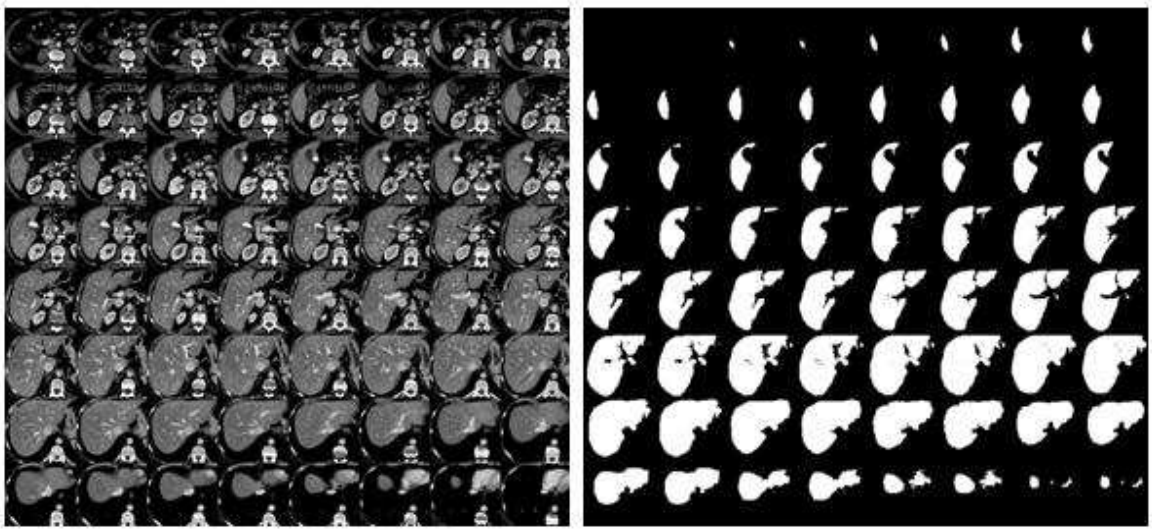


Figura 5.2.3.1, Información de apariencia y forma del hígado imagen con Liver Segm Server C++

Uso de ITK

ITK es un conjunto de librerías de código abierto que proporciona a los programadores una amplia gama de herramientas software para el análisis de imágenes. Las ITK están muy optimizadas, por lo que confieren un marco de trabajo más rápido y efectivo. Están desarrolladas a través de

metodologías de programación extrema, y emplea los principales algoritmos punteros para el registro y la segmentación de datos multidimensionales. Las metas de ITK incluyen:

- Apoyar el proyecto "*Visible Human Project*".
- Establecer de una base para futuras investigaciones.
- Crear un directorio de algoritmos fundamentales.
- Crear una plataforma para el desarrollo de productos avanzados.
- Apoyar a las aplicaciones comerciales de la tecnología.
- Crear convenios para futuros trabajos.
- Incrementar la comunidad auto-sostenible de usuarios y desarrolladores de software.

Supercomputación

La segmentación de imágenes médicas usando Liver-Segm es un proceso relativamente costoso, tanto en tiempo como en memoria. Por ello, se ha probado su uso en Magerit, un supercomputador perteneciente a la Universidad Politécnica de Madrid. En general, en un supercomputador se reduce considerablemente el tiempo de ejecución de un programa, aunque también debe tenerse en cuenta el tiempo que se emplea en el envío y la recepción de la información.

Magerit

Magerit es un clúster de 1204 nodos (1036 nodos eServer BladeCenter JS20 y 168 nodos eServer BladeCenter JS21) que utiliza como sistema operativo una distribución SuSE Linux Enterprise Server 9. Dependiendo del tipo de nodo se tienen dos configuraciones diferentes aunque completamente compatibles:

- JS20: 2 cores en dos procesadores IBM PowerPC single-core 970FX de 64 bits a 2'2 GHz, 4 GB de memoria RAM y 40 GB de disco duro local. Estos nodos alcanzan una potencia de 8'8 GFLOPS.
- JS21: 4 cores en dos procesadores IBM PowerPC dual-core 970MP de 64 bits a 2'3 GHz, 8 GB de memoria RAM y 80 GB de disco duro local. Estos nodos alcanzan una potencia de 9'2 GFLOPS.

El sistema dispone de una capacidad de almacenamiento distribuida de 192 TB en un sistema de ficheros distribuido y tolerante a fallos de IBM denominado GPFS (*Global Parallel File System*). La conexión entre los elementos de cómputo se realiza mediante una red Myrinet de baja latencia. Asimismo, cada nodo dispone de otras dos conexiones Ethernet de 1 Gbps auxiliares para que su gestión no interfiera con los cálculos de usuarios realizados en los nodos. La comunicación con el exterior se realiza a través de enlaces de 1 Gbps y 10 Gbps.

Las herramientas utilizadas para la conexión con Magerit son:

- PuTTY: es un cliente SSH, Telnet y rlogin con licencia libre.
- WinSCP: es un cliente SFTP gráfico para Windows que emplea SSH. Es una aplicación de Software libre. Su función principal es facilitar la transferencia segura de archivos entre dos sistemas informáticos, el local y el remoto, que ofrezca servicios SSH.

- XMing: es una implementación portátil del sistema de ventanas X para sistemas operativos Microsoft Windows XP, 2003 y Vista. El servidor Xming está basado en el servidor X.org

5.3. IMPLEMENTACIÓN MULTIPLATAFORMA DE ALGORITMOS DE PROCESAMIENTO DE IMÁGENES BIOMÉDICAS 2D EN MATLAB Y C++

Este proyecto se desarrolla en el marco de una disciplina creciente en los últimos años en el mundo de la ciencia y tecnología, la Visión Artificial; dentro de este extenso campo el presente proyecto se ocupa del procesamiento y análisis de imágenes biomédicas digitales.
La problemática de la que se ocupa el procesamiento de imágenes biomédicas surge cuando antigüamente los profesionales de la salud elaboraban diagnósticos a raíz de la observación meramente visual de las imágenes resultado de sus experimentos. Dicho diagnóstico es subjetivo, sin ningún fundamento científico y aproximado, con lo cual anteriormente el diagnóstico al paciente era muy precario.
En la actualidad, gracias a la visión por computador, en este proyecto se trata de establecer soporte informático para analizar y procesar imágenes biomédicas que proceden de un sistema de adquisición de imágenes. El objetivo es proporcionar al personal biomédico un software informático que le ayude a elaborar un diagnóstico fiable, exacto y con resultados científicos que se desprenden de las imágenes analizadas por la algoritmia de procesamiento.
Así pues es el campo de la implementación y desarrollo de algoritmos que permiten gestionar imágenes biomédicas, sacando una serie de resultados relevantes de dichas imágenes que permiten mejorar el diagnóstico médico.

5.3.1. Implementación de algoritmos de procesamiento de imágenes biomédicas procedentes de microscopia mediante ITK, IPP y MatLab

El objetivo principal de este proyecto ha sido la implementación de algoritmos de segmentación de espermatozoides humanos usando componentes ITK e IPP para poder comparar estos paquees entre si y con las funciones incluidas en el paquete de MatlLab.

- Instalación de las ITK
- Instalación de las IPP
- Fases del algoritmo
- Comparativa ITK, IPP y Matlab

Instalación de las ITK

Debido a que el usuario final puede estar trabajando sobre cualquiera de las plataformas existentes (Linux, Solaris, Mac, Windows…) y utilizando cualquier suite de programación (Microsoft Visual Studio, Borland C++ Builder…), es necesario que la instalación de este toolkit permita que todos

sus componentes estén disponibles al desarrollador independientemente de la plataforma sobre la que esté trabajando o del compilador que esté usando.
Para garantizar este correcto funcionamiento se tiene el CMake, un sistema de plataforma cruzada (cross-platform) y código abierto que, una vez reconocido el tipo de plataforma e introducido el entorno de compilación a usar, genera los archivos de trabajo necesarios (como puede ser el espacio de trabajo) para el desarrollo de la aplicación que necesitará de las ITK.
En primer lugar se han de descargar el código fuente de las ITK desde la siguiente dirección: http://www.itk.org/HTML/Download.php A continuación se descomprimen en la ubicación deseada dentro del PC.
A continuación se debe descargar CMake desde http://www.cmake.org/HTML/Download.html donde deberemos seleccionar la plataforma sobre la cual trabajaremos. En este caso, al trabajar sobre Windows descargamos el archivo "CMsetup205.exe"
En su ejecución se tiene:

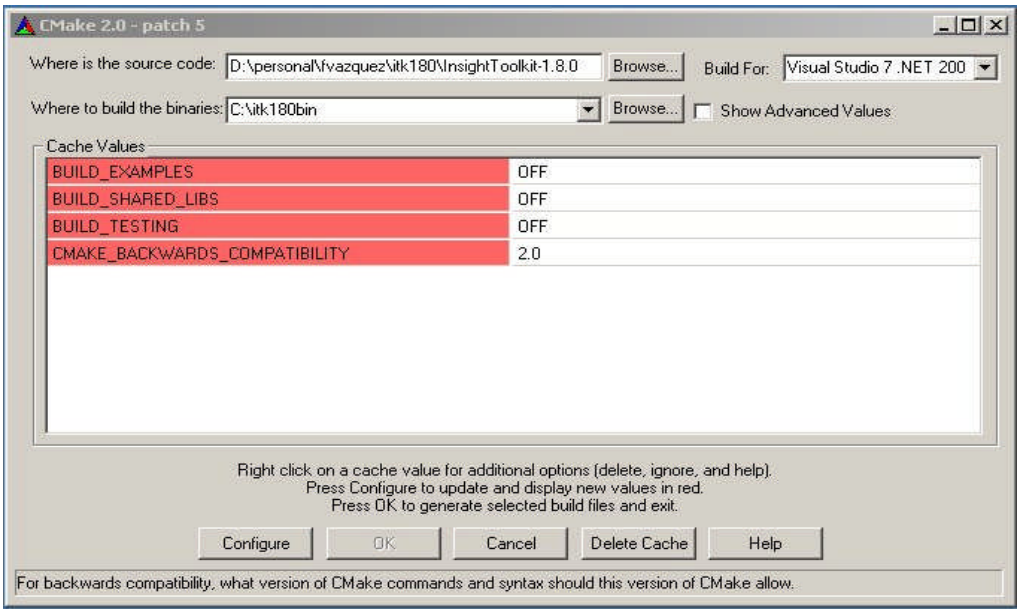


Figura 5.3.1.1, Ventana para Windows descarga de archivos "CMsetup205.exe"

Se rellenan los campos referentes a la ubicación del código fuente y dónde se generarán los archivos binarios. Se activan las opciones de generación de librerías y de ejemplos y se configura el sistema.

CMake genera los proyectos para Visual ya configurados para poder usar los componentes ITK. Se carga el espacio de trabajo "ITK.sln" generado en la ubicación especificada anteriormente y se genera el proyecto. Este proceso suele tardar entre 15 y 30 minutos, pero todo depende de las capacidades del ordenador.
El proceso debe finalizar sin que se haya producido ningún error de compilación, aunque si se pueden producir advertencias u omisiones.

Finalizado el proceso las ITK se hallarán correctamente instaladas en el sistema y listas para ser usadas.

Instalación de las IPP

Las Intel ®IPP forman una biblioteca software de funciones útiles optimizadas para múltiples procesadores y sistemas de operativos.
Intel IPP ofrece el mejor funcionamiento de aplicación a través de núcleos rápidos que usan la optimización manual y juegos de instrucción específicos para cada arquitectura.
Para solicitar un paquete de evaluación de IPP, es necesario el relleno de un formulario en el sitio seguro https://registrationcenter.intel.com/EvalCenter/EvalForm.aspx?ProductID=263 , indicando el tipo de librería (imagen, audio o señales) y del tipo de procesador.
Se ha de indicar una dirección de correo válida donde se recibirá el archivo de licencia y el número de serie para la instalación.

Descargar archivo de instalación w_ipp_ia32_itanium_em64_eval_p_4_1_2_ev05.exe desde http://www.intel.com/software/products/ipp/downloads/ippwin_eval.htm.
Se ejecuta la instalación del paquete y durante a instalación se indica la ubicación del archivo de licencia recibido.
Para poder usar las librerías IPP dentro del entorno de programación Microsoft Visual.NET no es necesario el uso de un programa de plataforma cruzada como ocurría con las ITK y CMake para la generación de proyectos.
Se debe crear un proyecto de la forma habitual. En este caso se va a partir de unos de los proyectos con las ITK ya incluidas a través de CMake.

Los include y librerías necesarias para el proyecto son:
- kernel32.lib user32.lib gdi32.lib (Que suelen estar ya incluidas).
- ippi20.lib ipps20.lib ippcore.lib (Se deben incluir a mano).

Para los include: proyecto -> propiedades ->C/C++ ->General-> directorios de inclusión adicionales. Indicar la ubicación del paquete IPP instalado.

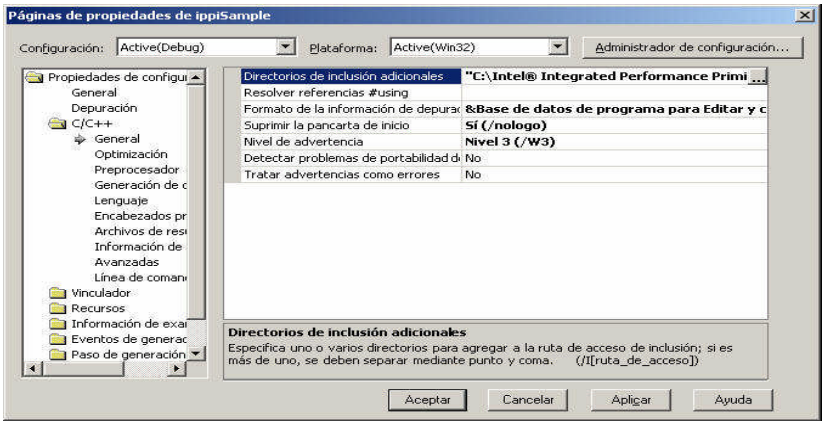


Figura 5.3.1.2,Ventana para Windows descarga de archivos "CMsetup205.exe"

Fases del algoritmo

El funcionamiento del programa de realiza en las siguientes fases:

1. Lectura de la imagen de entrada.
2. Determinación del histograma de la componente verde y azul.
3. Determinación del tamaño de la imagen.
4. Determinación del umbral del núcleo y del halo del espermatozoide.
5. Binarización de la imagen de entrada.
6. Hallar los halos cortados por el borde de imagen.
7. Hallar los halos completos.
8. Hallar los núcleos completos.
9. Escritura de las imágenes de salida de halos y núcleos completos.
10. Eliminar los nucleos sin halos y asignar a cada halo un único nucleo
11. Segmentar la imagen en regiones conteniendo un único halo y un único nucleo.
12. Hallar sus datos de area y centro.

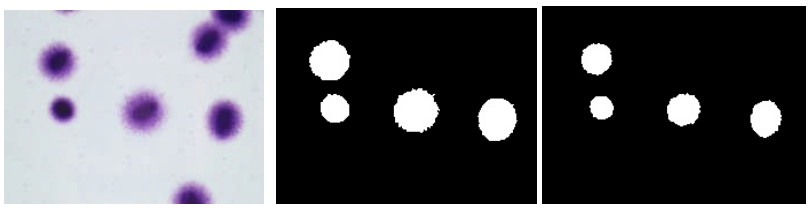


Figura 5.3.1.4. Imagenes de entrada halos y núcleos imágenes manejadas con algoritmos de fases

Obtención de las regiones de la imagen. Asignación del nucleo a su halo tras la eliminación de los halos pequeños por una dilatación

Figura 5.3.1.5. Imagen Geodesica, imágenes manejadas con algoritmos de fases

Comparativa ITK, IPP y Matlab

Para comparar los tiempos de ejecución de los distintos programas, se va a realizar una operación sobre una imagen que es común para todos los paquetes.
En este caso se va a implementar una transformación morfológica, una erosión clásica sobre una imagen ya binarizada. Se a cronometrar el tiempo que se tarda en aplicar el algoritmo, sin tener en cuenta el tiempo empleado en leer la imagen de entrada ni en escribir la imagen de salida.

Salida de la aplicación con IPP:

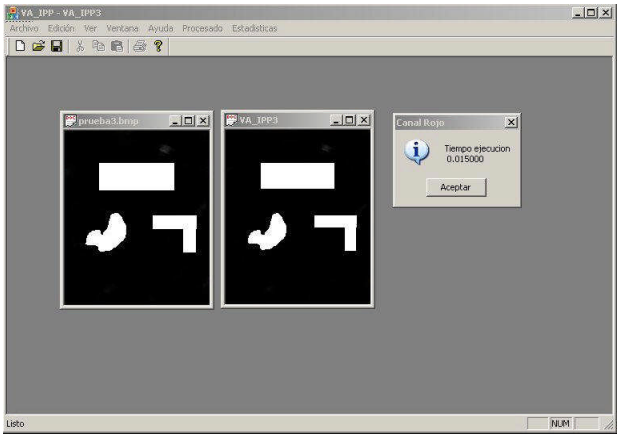


Figura 5.3.1.6, Imagen Geodesica, imágenes manejadas con algoritmos de fases salida con ITK

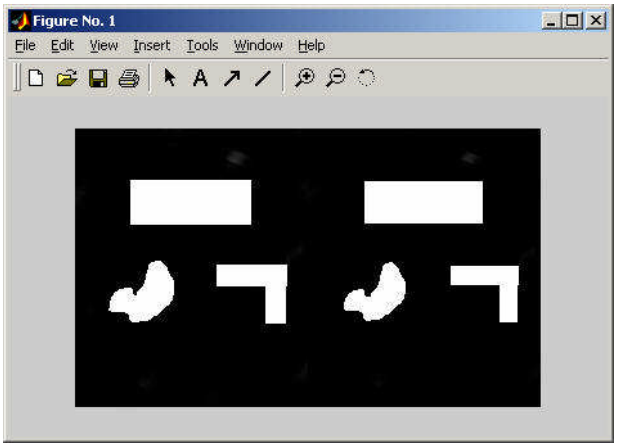


Figura 5.3.1.7. Imagen Geodesica, imágenes manejadas con algoritmos de fases, salida con Matlab

Siendo los tiempos empleados para la misma imagen de 453 X 543:

	IPP	ITK	MATLAB
453x543	0,015	6,49	0,442

La ventaja de las IPP es evidente. Tanto IPP como Matlab aprovechan más la tecnología del microprocesador que ITK. Sin embargo en este caso se ha realizado una erosión clásica. No se ha realizado una erosión geodésica, que seguro aumentaría el tiempo de ejecución. IPP no implementa transformadas geodésicas como ITK y Matlab. Esa es de hecho la principal limitación de IPP ya que las transformadas geodésicas son de gran importancia en este proyecto ya que se utilizan para eliminar los objetos cortados por el borde y la eliminación de los halos pequeños sin nucleo.

5.4. Paralelización de tareas para el análisis de imágenes médicas con aplicaciones a la segmentación del hígado

Resumen

Las imágenes biomédicas se caracterizan fundamentalmente por la dificultad que existe a la hora de generar información válida para ser procesada. Poseen gran cantidad de ruido y una enorme variabilidad en sus propiedades. Debido a esto hay que "preparar" la imagen para su segmentación realizando diferentes filtrados. En especial las imágenes de RM son imágenes muy pesadas porque suelen tener muchas rodajas que segmentar y su procesamiento es relativamente lento.

El objetivo del presente proyecto es optimizar los algoritmos de segmentación del hígado y de las lesiones hepáticas adquiridas por RM, la optimización se ha realizado tanto para mejorar el resultado de la segmentación, como para mejorar los tiempos de ejecución de los algoritmos. Por un lado, para mejorar la segmentación se han incluido algoritmos con apariencia y forma escritos en C++ para añadirlos a la aplicación "Liver-Segm". Por el otro, se han mejorado los tiempos de ejecución de la difusión anisotrópica (filtro de suavizado) utilizando la toolbox de procesamiento en paralelo de Matlab y MPI junto con OpenMP para los algoritmos escritos en C++.

En un futuro se pretende, si es posible, paralelizar la parte de la segmentación que más tarda en su ejecución, esto es, el contorno activo que evoluciona al borde del objeto a segmentar, en nuestro caso el hígado, utilizando las herramientas descritas en el proyecto.

Liver-Segm

Liver-Segm es la aplicación desarrollada por el grupo de investigación de Visión Artificial (GVA). Esta aplicación esta basada en ITK-SNAP, la cual es una aplicación software "Open Source" utilizada para segmentar estructuras en 3D de imágenes médicas. Utiliza las librerías ITK para la segmentación de las imágenes, VTK para el renderizado y FLTK para el entorno gráfico.

Se ha modificado la aplicación ITK-SNAP introduciendo los algoritmos para la segmentación del hígado y de las lesiones hepáticas, utilizando las librerías ITK.

- **Segmentación del higado y de lesiones hepática**

Las etapas en la segmentación del hígado y las lesiones hepáticas son:

- Expansión del histograma.
- Filtrado de difusión anisotrópica 3D.
- Umbralización.
- Detección de bordes.
- Post-Procesado morfológico.
- Contorno activo 3D.

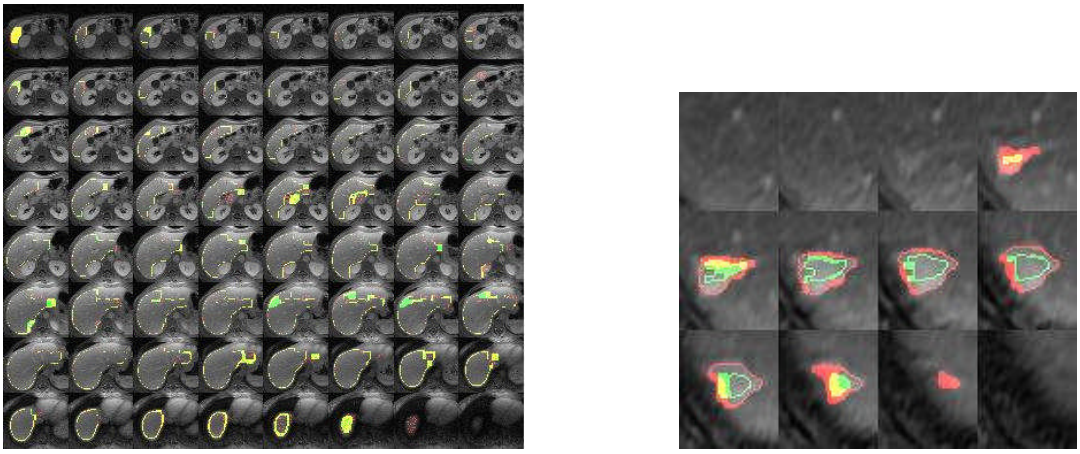


Figura 5.3.1.7,Segmentacion del hígado con lesiones apáticas aplicación con software open sours, base Matlab

Segmentación con apariencia y forma

La segmentación con apariencia y forma se basa en un conocimiento a priori del objeto a segmentar y se incorpora dicho conocimiento a la segmentación. Este método de segmentación es invariante ante rotaciones y traslaciones de la imagen a segmentar.

- Para implementar dicha segmentación se realiza una transformación no rígida de la forma previa conocida al centro de inercia del contorno activo. Para ello primero hay que obtener el modelo que se adapta mejor al objeto a segmentar de los distintos modelos a priori conocidos, los pasos que hay que realizar para cada modelo son:
- Adaptación del modelo a la imagen del contorno activo. En la primera iteración se partirá de la solución inicial calculada en la etapa de post-procesado morfológico.
- Normalización del modelo realizando un análisis en componentes principales (PCA).
- Registro de la imagen realizando una transformación geométrica estadística no rígida al centro de inercia del contorno activo.
- Se halla la distancia entre el modelo transformado y el contorno activo.

Estos pasos se realizan para cada iteración del contorno activo, por lo que el modelo elegido puede cambiar de una iteración a otra. El modelo con menor distancia será el que se utilizará para la segmentación, incorporando su información de forma al contorno activo 3D.

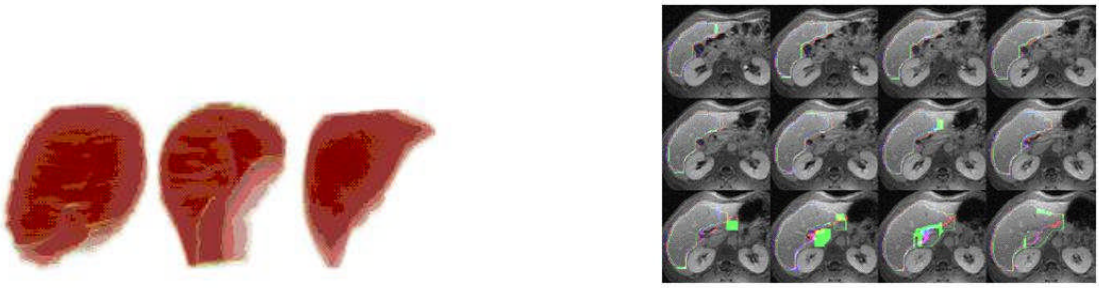


Figura 5.3.1.8,Segmentacion del hígado con lesiones apáticas aplicación con software open sours, base Matlab. Caomputacion paralela

Computación paralela con MPI

MPI ("Message Passing Interface", Interfaz de Paso de Mensajes) es un estándar que define la sintaxis y la semántica de las funciones contenidas en una biblioteca de paso de mensajes diseñada para ser usada en programas que exploten la existencia de múltiples procesadores.

El paso de mensajes es una técnica empleada en programación concurrente para aportar sincronización entre procesos y permitir la exclusión mutua, de manera similar a como se hace con los semáforos, monitores, etc.

Su principal característica es que no precisa de memoria compartida, por lo que es muy importante en la programación para sistemas distribuidos.

La Interfaz de Paso de Mensajes (conocido ampliamente como MPI, siglas en inglés de Message Passing Interface) es un protocolo de comunicación entre computadoras. Es el estándar para la comunicación entre los nodos que ejecutan un programa en un sistema de memoria distribuida. Las implementaciones en MPI consisten en un conjunto de bibliotecas de rutinas que pueden ser utilizadas en programas escritos en los lenguajes de programación C, C++, Fortran y Ada. La ventaja de MPI sobre otras bibliotecas de paso de mensajes, es que los programas que utilizan la biblioteca son portables (dado que MPI ha sido implementado para casi toda arquitectura de memoria distribuida), y rápidos, (porque cada implementación de la biblioteca ha sido optimizada para el hardware en la cual se ejecuta).

Computación paralela en Matlab

Parallel Computing Toolbox™ es una toolbox que ofrece Matlab2008b, esta toolbox te permite:

- Descargar de trabajo a una sesión de MATLAB (el cliente) hacia otras sesiones de MATLAB llamadas trabajadores para aprovechar al máximo los procesadores que tenga nuestro ordenador de forma local.
- Crear versiones paralelizadas de tus aplicaciones con un cambio mínimo de código.
- Ejecutar hasta cuatro trabajadores de forma local, además de la sesión del cliente. Para poder ejecutar mas trabajadores, por ejemplo en un cluster de ordenadores, se utiliza MATLAB® Distributed Computing Server™ que te permite ejecutar tantos trabajadores como nodos tenga el cluster teniendo las correspondientes licencias.

- **Paralelización de bucles for**

El concepto del comando parfor es el mismo que un bucle for normal se ejecutan una serie de sentencias (el cuerpo del bucle) en un rango de valores. Cuando ejecutamos en MATLAB el comando parfor parte del cuerpo del bucle se ejecuta en el cliente (donde se emite el parfor) y parte se ejecuta en paralelo en los trabajadores. Los datos necesarios sobre los que opera parfor se envían desde el cliente a los trabajadores y los resultados se envían al cliente todos juntos.

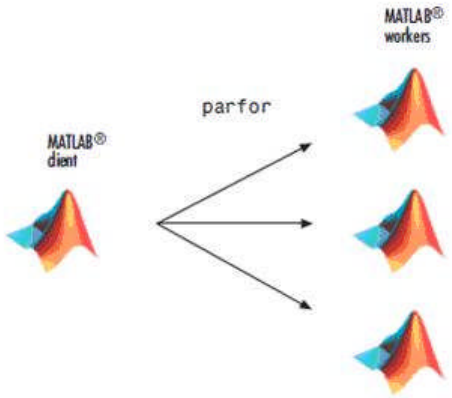


Figura 5.3.1.9, Esquema de computación paralela con Matlab

- **Single Program Multiple Data (spmd)**

El lenguaje spmd permite intercalar programación en serie y en paralelo. Las declaraciones spmd permiten definir un bloque de código que se ejecuta simultáneamente en varios laboratorios o labs (los laboratorios o labs son los diferentes nodos o procesadores donde se ejecuta nuestro programa). Las variables asignadas dentro de las declaraciones smpd permiten acceso directo a ellas por el cliente utilizando objetos Composite.

"Single Program" significa que el código de un programa se ejecuta en varios laboratorios simultáneamente. Si ejecutas un programa en el cliente, aquellas partes que estén etiquetadas como bloques spmd se ejecutan en varios laboratorios. Cuando el bloque spmd finaliza, el programa sigue ejecutándose en el cliente.

"Multiple Data" significa que a pesar de que las declaraciones spmd ejecutan idéntico código en todos los laboratorios, cada laboratorio puede tener diferentes o únicos datos para ese código. Por lo tanto, múltiples conjuntos de datos pueden ser por procesados por múltiples laboratorios.

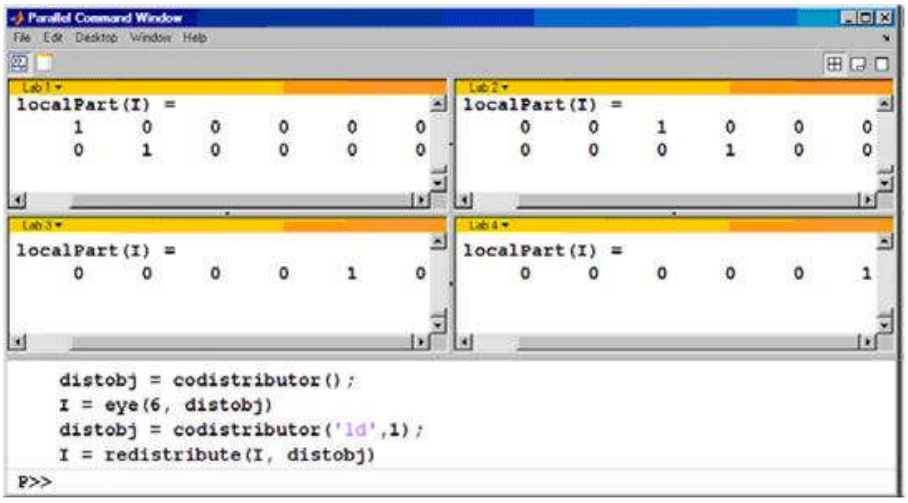
Las aplicaciones típicas con spmd son aquellas que requieren una ejecución simultánea de un programa en múltiples conjuntos de datos, cuando la comunicación o sincronización es necesaria entre los laboratorios. Algunos de los casos son los siguientes:

- Programas que tardan mucho tiempo en ejecutarse: spmd permite a varios laboratorios computar soluciones simultáneamente.
- Programas que operan con grandes conjuntos de datos: spmd permite que los datos se distribuyan a varios laboratorios

- **Computación paralela interactiva con pmode**

pmode te permite trabajar interactivamente con un trabajo paralelo ejecutándose simultáneamente en varios laboratorios. Los comandos que se escriban en el prompt de pmode (Parallel Comand Window) se ejecutan en todos los laboratorios al mismo tiempo. Cada laboratorio ejecuta los comandos en su propio en su propio espacio de trabajo en sus propias variables.

La forma en que los laboratorios permanecen sincronizados es que cada laboratorio esta ocioso cuando completa un comando o declaración, esperando hasta que todos los laboratorios que estén trabajando hayan terminado la misma declaración. Sólo cuando todos estén ociosos, procederán juntos al próximo comando pmode.



- *Figura 5.3.1.10, Esquema de computación paralela con Matlab, interactiva con pmode*

- **Matrices codistribuidas**

Las matrices codistribuidas se dividen en segmentos, y cada uno de ellos reside en un entorno de trabajo de diferentes laboratorios. Cada laboratorio tiene su propio segmento de la matriz para trabajar con él. La reducción del tamaño de la matriz en segmentos almacenados y procesados en cada laboratorio hace que el uso de memoria sea más eficiente y el procesamiento sea más rápido, especialmente para grandes conjuntos de datos.

LAB 1				LAB 2			
1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64
LAB 3				LAB 4			

Figura 5.3.1.11, Esquema de computación paralela con Matlab, matrices codistribuidas

Computación distribuida en Matlab

Parallel Computing Toolbox™ y MATLAB® Distributed Computing Server™ te permiten coordinar y ejecutar diferentes operaciones simultáneamente en un cluster de ordenadores, aumentando la velocidad de ejecución de las aplicaciones.

Un trabajo es una larga ejecución de un programa que quieres ejecutar en una sesión de MATLAB. Este trabajo se puede dividir en segmentos llamados tareas. Esta división en distintas tareas la debes realizar tú, pudiendo dividir el trabajo en tareas idénticas o distintas dependiendo de la naturaleza de tu programa.

La sesión de MATLAB en la que el trabajo y las tareas son definidas se llama la sesión cliente. El cliente usa el Parallel Computing Toolbox para llevar a cabo la definición de los trabajos y las tareas. MATLAB® Distributed Computing Server es el software que lleva a cabo la ejecución del trabajo evaluando cada una de las tareas devolviendo los resultados al cliente.

El job manager (administrador del trabajo) es la parte del software del servidor que coordina la ejecución de los trabajos y la evaluación de sus tareas. job manager distribuye las tareas para su evaluación a cada uno de los trabajadores que son sesiones servidor de MATLAB. El uso de MathWorks™ job manager es opcional pudiéndose usar también Windows® Compute Cluster Server (CCS) o Platform LSF®.

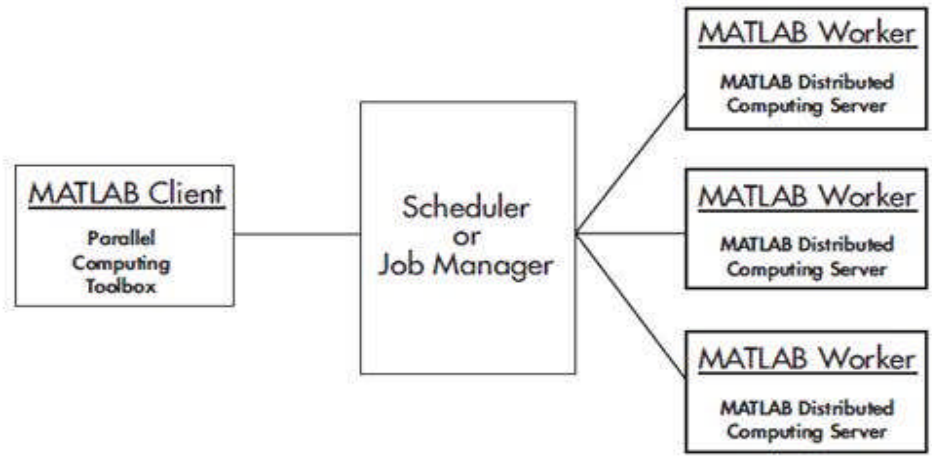


Figura 5.3.1.12, Diagrama abloques computaciondistributiva en matlab

OpenMP

OpenMP (*Open Multi Processing*) es una API (*Application Programming Interface*) que permite añadir concurrencia a las aplicaciones mediante paralelismo con memoria compartida. Se basa en la creación de hilos de ejecución paralelos compartiendo las variables del proceso padre que los crea. Ofrece un modelo portable y escalable para los desarrolladores de aplicaciones paralelas de memoria compartida.

Es portable ya que esta disponible en múltiples plataformas y lenguajes, desde las derivadas de UNIX hasta la plataforma Windows. Existen extensiones para los lenguajes más conocidos como C, C++ y Fortran.

Esta compuesta de los tres principales componentes de una API:

- Directivas de Compilador.
- Bibliotecas de rutinas en tiempo de ejecución.
- Variables de entorno.

Es estándar ya que está definido conjuntamente con el respaldo de los mayores proveedores de hardware y software (*Sun Microsystems*, *IBM*, *HP*, *Intel*, *Microsoft*...). Se espera que se convierta en un estándar ANSI a largo plazo.

- **Modelo de programación de OpenMP**

OpenMP se basa en la existencia de múltiples hilos en el paradigma de programación de memoria compartida. Un proceso de la memoria compartida consta de varios hilos. Es modelo de programación explícito (no automático), ofrece al programador el control total sobre la paralelización.

Utiliza el modelo *fork-join* para la ejecución en paralelo:

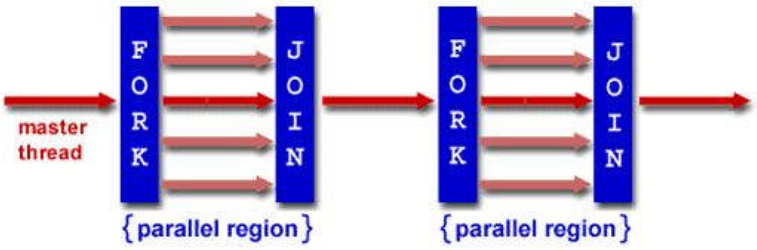


Figura 5.3.1.13, Modelo de programación de OpenMP

- **Características del lenguaje OpenMP**

OpenMP proporciona directivas, funciones de librería, y variables de entorno para crear y controlar la ejecución de programas paralelos. Un gran número de aplicaciones se pueden paralelizar utilizando un número relativamente reducido de constructores y funciones. Algunas rutinas de la librería de OpenMP son:

- Constructor de región paralela (*parallel*).
- Constructores de trabajo compartido:
- Constructor de bucle (*for*).
- Constructor de secciones (*sections*).
- Constructor de región única (*single*).
- Claúsulas *Data-Sharing*, *No Wait*, y *Schedule.*
- Constructor de barrera (*barrier*).
- Constructor de región crítica (*critical*).
- Constructor maestro (*master*).

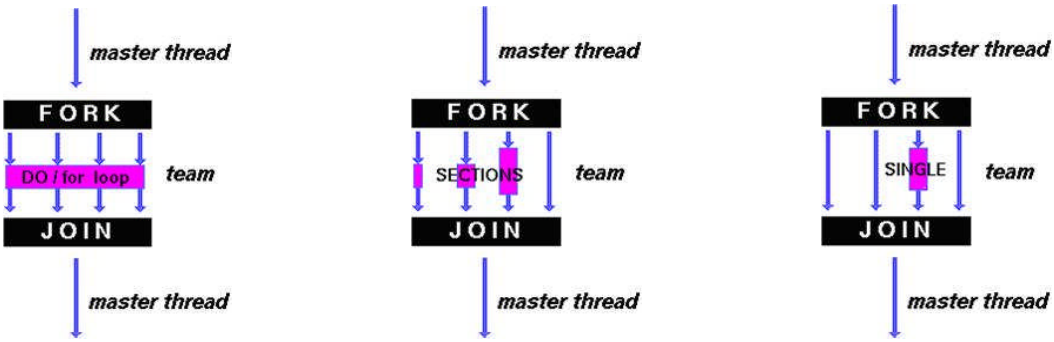


Figura 5.3.1.13, Modelo de programación de OpenMP, Características del lenguaje OpenMP

1. TRATAMIENTO DIGITAL DE SEÑALES

1.1. Tipos de señales
1.2. Cuantificación vectorial
1.2.1. Concepto
1.2.2. Proceso de cuantificación
1.2.3. Obtención del codebook inicial
1.2.3.1. Método aleatorio (random)
1.2.3.2. Método de poda (pruning)
1.2.3.3. Método de splitting
1.2.4. Mejora del codebook
1.2.4.1. Iteración de Lloyd
1.2.4.2. Algoritmo de Lloyd generalizado
1.2.5. Variantes de la cuantificación vectorial
1.2.5.1. Cuantificación vectorial clasificada
1.2.5.2. Cuantificación vectorial de vectores transformados
1.2.5.3. Cuantificación vectorial de ganancia y forma
1.2.6. Aplicaciones 14
1.2.6.1. Codificación Block Truncation Coding (BTC) de imágenes con cuantificación vectorial
1.2.6.2. Cuantificación vectorial de coeficientes LPC (LPC-VQ)
1.3. Redes neuronales artificiales
1.3.1. La neurona biológica
1.3.2. Modelo de neurona artificial
1.3.3. Arquitectura de una Red Neuronal Artificial (RNA)
1.3.3.1. Redes progresivas
1.3.3.2. Redes realimentadas
1.3.4. Aprendizaje
1.3.5. Aplicaciones de las RNA

2. Filtros no lineales

2.1. Concepto. Tipos de filtros
2.1.2. Filtros de estadísticos ordenados (OS)

3. TRATAMIENTO DIGITAL DE IMAGEN

3.1. Introducción
3.2. Mejora de imágenes
3.2.1. Introducción
3.2.2. Operaciones puntuales
3.2.2.1. Función definida a intervalos
3.2.2.2. Negativo de una imagen

- 3.2.2.3. Extracción de bits
- 3.2.2.4. Procedimientos basados en el histograma
- 3.2.3. Operaciones espaciales
  - 3.2.3.1. Filtrado espacial
  - 3.2.3.2. Suavizado direccional
  - 3.2.3.3. Filtrado de mediana
  - 3.2.3.4. Ampliación de imágenes
- 3.2.4. Operaciones transformadas
- 3.2.5. Restauración de imágenes
- 3.2.6. Pseudocolor

3.3. Compresión

- 3.3.1. Introducción. Tipos y aplicaciones
  - 3.3.1.1. Aplicaciones
  - 3.3.1.2. Máxima compresión conseguible sin error
- 3.3.2. Codificación de pixel
  - 3.3.2.1. PCM
  - 3.3.2.2. Cuantificación de contraste
  - 3.3.2.3. Dithering
  - 3.3.2.4. Codificación entrópica
  - 3.3.2.5. RLE (Run Length Encoding)
  - 3.3.2.6. Codificación de bit plane
- 3.3.3. Codificación predictiva
  - 3.3.3.1. DPCM (Differential Pulse Code Modulation)
  - 3.3.3.2. Modulación delta
  - 3.3.3.3. Técnicas adaptativas
- 3.3.4. Codificación transformada
  - 3.3.4.1. Códificación zonal y codificación umbral
  - 3.3.4.2. Codificación transformada adaptativa
- 3.3.5. Codificación híbrida
- 3.3.6. Codificación piramidal
- 3.3.7. JPEG (Joint Photographic Experts Group)
- 3.3.8. Codificación de imágenes en color

3.4. Secuencias de imágenes

- 3.4.1. Aplicaciones
- 3.4.2. Estimación de movimiento
  - 3.4.2.1. Método three step
  - 3.4.2.2. Método conjugado modificado
- 3.4.3. Predicción

3.5. Morfología matemática

- 3.5.1. Definición
- 3.5.2. Operaciones

3.6. Visión artificial

3.6.1. Detección de contornos
3.6.2. Formas de objetos
3.6.2.1. Método de las firmas
3.6.3. Reconocimiento óptico de caracteres
3.6.3.1. Método de los perfile
3.6.3.2. Reconocimiento de caracteres por el método de los perfiles
3.6.3.3. OCR con red neuronal
3.6.4. Reconocimiento de personas mediante características biométricas
3.6.4.1. Aplicaciones y ventajas sobre los métodos tradicionales
3.6.4.2. Precisión del sistema
3.6.4.3. Cuestiones importantes
3.6.4.4. Huellas dactilares
3.6.4.5. Geometría de la mano
3.6.4.6. Geometría de los dedos
3.6.4.7. Venas de las manos
3.6.4.8. Retina
3.6.4.9. fis
3.6.4.10. Firma
3.6.4.11. Reconocimiento de caras
3.6.4.12. Mejoras en las aplicaciones de verificación
3.6.4.13. Sistemas implantados en la actualidad

4. IMÁGENES MEDICAS: ADQUISICION, INSTRUMENTACIÓN Y GESTION

4.1. Introducción
4.2. Instrumentación de radiología analógica y digital
4.2.1 Detección de rayos x
4.2.2. Captura de imágenes estáticas
4.2.3. Captura de imágenes dinámicas
4.2.4 Aplicaciones
4.2.4.1 Radiología convencional
4.2.4.2. Fluoroscopio
4.2.4.3. Angiografía
4.2.4.4. Mamografía
4.2.4.5. Arco en C
4.2.4.6. Litotricia
4.2.4.7. Tomografía Computarizada
4.3. Radiología digital vs radiología analógica
4.4. Resonancia magnética
4.4.1. Introducción
4.4.2. Teoría

Bibliografía

Solucion de problemas de ingeniería con Matlab
Delores M. Etter Ed. Pretntice Hall

Tratamiento digital de voz e imagen
Marco Faúndez Zanuy Ed. Alfaomega

Nonlinear Biomédical Signal Procesing
Ed. By Metin Axay
Biomédical signal procesing and signal Modeling
Lugen N Bruce

Bibliografía electrónica

www.elai.upm.es/spain/investiga
www.verona.fip.unam.mx
www.itzamna.izt.uam.mx
www.bioingenieria.edu.ar
www.cinvestav.mx
www.biblioteca.mty.itesm.mx
www.bloghost.udg.mx
www.wolfram.com
www.ccg.ciens.edu.ue

Conclusiones:

Como vimos la naturaleza de la adquisición de una imagen medica para los diversos estudios, se vuelve compleja tratar de digitalizarla con toda la información requerida para un buen diagnostico medico, mucho mas homogenizar un formato digital para cada una de las distintas imágenes medicas. Por lo cual la digitalización y almacenamiento de estas se vuelve un serio reto para las disciplinas que e intervienen es esto como son la Ingeniería Biomédica, Ingeniería Electrónica y La Ingeniería en computación.

El manipular imágenes con medicas con software matemáticos nos da un alto grado de incertidumbre, ya que estos software introducen datos nuevos a la imagen, con estos datos nuevos se produce una imagen secundaria la cual no fue obtenida con los datos originales si no a completada con los datos introducidos originalmente, digamos si tenemos un paciente con un tumor, y se ingresan los datos de esta imagen a un software matemático, este podría o no representar este tumor, o en caso contrario si no lo hay tal vez aparezca, en este caso se daría un diagnostico equivoco.

Decimos entonces que el primer paso para el desarrollo de un formato digital estándar es primero en la adquisición de la imagen, el primer paso esta en el hardware y después se tendría que trabajar con el software.

También sabemos que la cantidad de la información que se pueda adquirir mediante aplicaciones biomédicas debe ser suficiente para una muy buena calidad de imagen para su buena interpretación.