Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

Procesamiento Digital de Imágenes

Prof. Manuel Martín Ortíz

Enero 2013

Nota.

Quiero dedicar este pequeño trabajo a nuestro Profesor y entrañable Amigo **Dr. Rodolfo Reyes Sánchez**.

Rodolfo es uno de los Primeros y a mi criterio miembro de los verdaderos fundadores del Colegio de Computación de la ECFM de la UAP. Los tiempos han cambiado así como los nombres de las instituciones, pero el espíritu con el cual ésta Carrera se creó en el seno del Colegio de Computación en la década de los años 70's espero se defienda como lo hizo Rodolfo con vehemencia y entrega.

Él ya no está para ver la obra de los pioneros (incluyéndolo a él) de la Computación en la UAP, desde una propuesta realmente visionaria del Ing. Luis Rivera Terrazas hasta la Facultad que ahora en 2013 aloja a miles de estudiantes y cientos de profesores, aunado a esto a los cientos de egresados de la misma.

Como un humilde homenaje a ellos reporto este trabajo producto de la labor diaria como profesor de la Asignatura Correspondiente. Si bien el mismo es incompleto y plagado de pequeños errores, junto a los nuevos compañeros a corto plazo se hará una entrega de un material ampliado, mejorado y de mayor calidad.

Manuel Martín O
Profesor de la FCC – BUAP
Enero de 2013

INDICE

Capítulo I. Introducción.

Capítulo II. Operaciones Orientadas al Punto

Capítulo III. Operaciones Orientadas a la Región

Capítulo IV. Operaciones Geométricas

Capítulo V. Operaciones entre Imágenes

Capítulo VI. Operaciones Especiales

Capítulo VII. Falso Color y Pseudocolor

Capítulo 1

Fundamentos

El manejo de las imágenes digitales se ha convertido en las últimas décadas en un tema de interés extendido en diferentes áreas de las ciencias naturales, las ciencias médicas y las aplicaciones tecnológicas entre otras. El crecimiento en el poder de cómputo, las capacidades de almacenamiento y los nuevos sistemas de desplegado, captura e impresión de bajo costo han facilitado el desarrollo de ésta disciplina.

Hace no mucho las posibilidades de los equipos de captura y procesamiento digital eran bastante limitadas y los costos y tiempo de procesamiento prohibitivos. Ante lo cual en muy pocas áreas se prestaba atención al potencial que las herramientas para el manejo de imágenes digitales ofrecían. La explotación de éstas herramientas se había quedado restringida a algunas secciones de investigación y el desarrollo de aplicaciones de software se orientaba hacia problemas donde el presupuesto era vasto. En la actualidad es posible explotar plataformas de bajo costo y obtener resultados de gran calidad y crear aplicaciones de gran utilidad, versátiles y flexibles, así como aplicaciones de software de propósito específico para atender las diversas necesidades de los especialistas.

Es posible citar gran cantidad de ejemplos donde el procesamiento de imágenes ayuda a analizar, deducir y tomar decisiones. Entre otras áreas en las cuales se han desarrollado herramientas de gran utili-

dad podemos mencionar las siguientes: Medicina, Fisiología, Biometría, Astronomía, Ciencias Ambientales, Robótica, Metalúrgica, Física, Electrónica, Biología y el ROC (Reconocimiento Óptico de Caracteres, OCR=Optic Character Recognition).

1.1. Adquisición

Existen diferentes medios para la obtención de una imagen digital, los más comunes son: los scanners de cama plana y las cámaras digitales. Ambos se basan en un dispositivo llamado CCD (Coupled Charge Device), éste se recibe la luz de la imagen, ya sea por reflexión o por transmisión e integra en un tiempo definido la cantidad de luz que llega a él. Formando un arreglo de CCD's es posible realizar la digitalización de la imagen por renglones o bien entera [1-2].

Se dice que la imagen ha sido *digitalizada*, ya que por cada región e ella se genera un *número* que representa la cantidad de luz que fue registrada. En el siguiente diagrama (fig. 1.1) se muestra un arreglo de 3×3 de digitalización. Para cada celda de registro se genera un número que corresponde a la cantidad

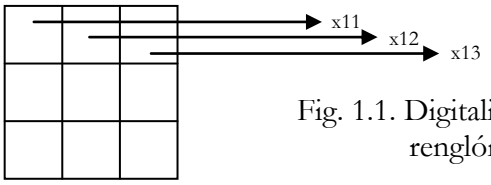


Fig. 1.1. Digitalización del primer renglón de una imagen

de luz que se registro en la zona, en la figura los valores x11, x12 y x13 son éstas cantidades. De igual manera se hace con todos los renglones. En general para un arreglo de n×m celdas se generará una matriz de la forma,

$$\mathbf{I} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}. \qquad (1.1)$$

Cada elemento de matriz representa una propiedad de la imagen. El sistema más simple de digitalización corresponde al llamado *tono de gris*, éste indica la cantidad o intensidad de la luz registrada. Por ejemplo para un sistema de transmisión, como es el negatoscopio utilizado en radiología, se coloca una

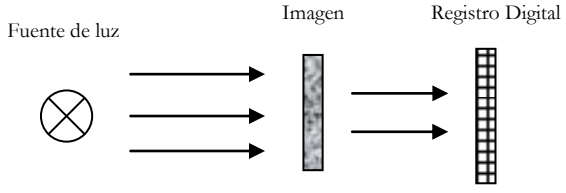


Fig. 1.2. Registro de por transmisión de luz

fuente de luz a continuación el negativo o lámina con la imagen y al final del arreglo el arreglo de CCD's. La figura siguiente (fig. 1.2) muestra el arreglo.

El otro modelo corresponde a los sistemas basados en *reflexión*, éste es el caso de los scanners de cama plana. Éstos dispositivos en vez tener un arreglo bidimensional de CCD's, sólo tienen un arreglo lineal de éstos. Mediante un sistema óptico se envía luz desde una lámpara a la imagen y es recibida en un

arreglo lineal de detectores, mediante un motor de pasos se mueve el sistema un "paso" y se vuelve a realizar otro registro, éste proceso se repite hasta cubrir toda la imagen. Este sistema es más económico, ya que utiliza un arreglo unidimensional de detectores y un subsistema de desplazamiento basado en un motor. La figura siguiente (fig. 1.3) muestra el principio de operación del dispositivo.

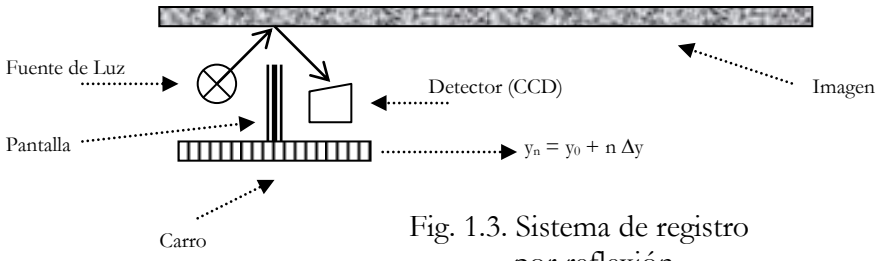
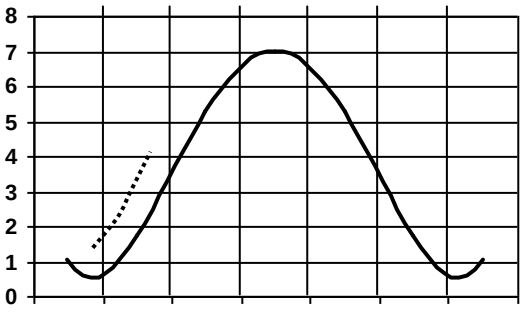


Fig. 1.3. Sistema de registro por reflexión.

Es claro que dependiendo del tipo de luz que se utilice y las propiedades de reflectividad de la imagen el registro variará. El objetivo de la pantalla es evitar que la luz de la fuente de luz llegue directamente al detector, de tal forma que lo que éste registra es la luz que se ha reflejado en la imagen, la cual contiene información de ella. El "carro" se mueve mediante un motor en pasos Δy. El sistema se compone de un cierto número de CCD's en la dirección perpendicular al movimiento del carro, los cuales registran de forma "paralela" la información de la luz reflejada a lo "ancho" de la imagen, éste arreglo de datos se almacena en forma de "renglones" en la matriz de digitalización. Y para cada "paso" del motor se hace el cambio de renglón generándose así las columnas de la matriz de datos (1.1). Cuando se ha recorrido toda la imagen se procede al almacenamiento.

El principio de la cámaras digitales es también la reflexión de la luz, solo que la fuente es externa al dispositivo (la cámara). En general puede haber varias fuentes de luz, pero solo hay un sistema de registro.

En todos los casos antes descritos los datos sufren un proceso de discretización o cuantización. Este proceso se refiere al hecho de que la información registrada no es almacenada de manera exacta como un número real - los cuales son densos -, sino como enteros, ya que el sistema luego de tomar el dato (en general analógico) lo pasa por un "Convertidor Analógico-Digital" (DAC). Este paso ocasiona una pérdida en la precisión de los registros. En la gráfica siguiente (fig. 1.4) se muestra una curva analógica digitalizada a 8 niveles.



N	X_N
1	1
2	**2**
3	5
4	7
5	5
6	2
7	1

Fig. 1.4. Proceso de cuantización de los datos

Puede notarse como por ejemplo para el segundo dato (línea punteada), la curva toma valores desde algunas décimas hasta un poco más de 3, pero el valor medio registrado en la tabla es 2.

1.2. Representación Digital Simple

La representación más simple de una imagen es una colección de puntos en un arreglo bidimensional, donde para cada punto se almacena una serie de parámetros propios de la imagen.

$$\mathbf{I} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}.$$

Para el registro de una imagen se acostumbran usar formatos que están referenciados por el número de bits que son utilizados para la cuantización de la cantidad de luz recibida. Cuando se trabaja solo la intensidad de la luz recibida en el detector se dice que la imagen ha sido adquirida en *tonos de gris*, los tipos más frecuentes son los siguientes:

Número de bits	**Descripción**
1	Imagen Monocromática / Blanco y Negro / Alto contraste
4	Imagen de 16 niveles
8	Imagen de 256 niveles

Actualmente la mayor parte de los dispositivos simples permiten adquirir la imagen con una profundidad de 8 bits, los cuales permiten representar 256 niveles de gris, el rango es el típico [0, 255]. Bajo esta representación los elementos de la matriz que representa a la imagen $x_{ij} \in [0 \ldots 255]$.

Al punto $x_{ij} = I[i, j]$ se le llama "*pixel*" y se debe entender el tono de gris que se ha asociado a la imagen en la coordenada (i, j) de la partición definida por el sistema de digitalización.

Si consideramos a la clase que representa a una imagen en tonos de gris, sus *propiedades* básicas serán:

Clase Imagen
Ancho : entero
Alto: entero
ProfBits : entero
Pixels : Matriz de enteros[0 … Ancho-1, 0 … Alto-1]

Donde Pixels $\in [0 \ldots 2^{\text{ProfBits}} - 1]$.

Muchos de los dispositivos modernos permiten realizar el registro en colores, de tal forma que para cada zona de la imagen se genera un pixel con tres componentes, la representación común es mediante un vector formado por una combinación de los colores básicos utilizados en la electrónica de video. Éstos colores son: el azul (B=Blue), el verde (G=Green) y el rojo(R=Red).

Así un color "cualquiera" se puede expresar como una combinación lineal de los colores básicos, es decir:

$$\mathbf{C} = r\,\mathbf{R} + g\,\mathbf{G} + b\,\mathbf{B}. \qquad (1.2)$$

Donde podemos interpretar un color como una combinación lineal de los vectores unitarios cromáticos (**R, G, B**) y las proyecciones en cada eje cromático o coordenadas del color **C** son (r, g, b). Podemos interpretar la ecuación (1.2) como la mezcla de los colores primarios. Donde se utilizará una cantidad *r* de rojo, una *g* de verde y una *b* de azul. Si r, g y b se manejan a 8 bits cada uno, se dice que la imagen esta representada a 24 bits.

Una representación en términos de clase para un pixel en la base RGB, puede ser la siguiente:

Clase PixelColor
r : entero
g : entero
b : entero

De donde la Clase ImagenenColor, puede quedar como sigue

Clase ImagenenColor
Ancho : entero
Alto: entero
ProfBits : entero
Pixels : Matriz de PixelColor[0 … Ancho-1, 0 … Alto-1]

En principio uno siente que el verde no es primario, sino debe ser el amarillo, pero la electrónica de video tomó el verde como base y seguiremos la convención establecida. Si tomamos una profundidad en bits de 8, los colores puros serán:

rojo = (255, 0, 0) verde = (0, 255, 0) azul = (0, 0, 255)

Definiremos en el espacio RGB un color "gris" aquel que tiene sus tres componentes iguales, de tal forma que todo gris tendrá la forma:

$$Z = (z, z, z). \qquad (1.3)$$

Es decir la proporción de cada uno de los colores básicos es la misma. Por tanto podemos afirmar que los *grises* se ubican en la recta que va del origen (0, 0, 0) al punto (255, 255, 255).

Se denomina *resolución* al número de pixeles por pulgada utilizados en el proceso de adquisición, ésta se indica en **dpi** (dots per inch = puntos por pulgada). Mientras mayor es la resolución mayor es la fidelidad de la imagen, es decir se obtienen más detalles de ella. Los valores de resolución varían de 75 dpi

hasta algunos miles (7200 dpi). Generalmente éste es un parámetro del dispositivo de adquisición que es posible definir durante el proceso de registro.

1.3. Almacenamiento

Para almacenar una imagen en disco o memoria se han desarrollado una diversidad de formatos, los campos básicos a considerarse son los descriptores principales de la imagen que están definidos en la clase ImagenenColor o Imagen (que se relaciona con una imagen en tonos de gris).

Se almacenan mínimamente: el ancho, el alto, la profundidad en bits y la matriz de pixeles. A esta representación se le llama un "Mapa de Bits" o simplemente BitMap.

Por ejemplo para guardar una imagen en tonos de gris adquirida de una fotografía de 2"x3" con una resolución de 300 dpi y una profundidad en bits de 8, se requiere un espacio de almacenamiento de:

$$N_1 = (2\times 300)(3\times 300)\ \text{bytes} = 5.4\times 10^5\ \text{bytes}$$

Que corresponde aproximadamente a ½ MB.

Si tomamos ahora una postal (5"×8")y la digitalizamos a 600 dpi en color a 24 bits, el espacio de almacenamiento requerido para el BitMap será:

$$N_2 = (5\times 600)(8\times 600)\ \text{bytes} = 1.44\times 10^7\ \text{bytes},$$

Que ahora corresponde a 13.73 MB. Puede notarse que el aumento de resolución al doble (300→ 600) y el aumento en el tamaño de la imagen en (5/2, 8/3) ≈ (2.5, 2.66) refleja el carácter no lineal de la transformación, donde $N_2 = 2\times 2\times 2.5\times (8/3)\ N_1 = 26.66\ N_1$, de donde el crecimiento es aproximadamente 26.7 veces.

Debido al gran espacio que se requiere para el proceso, se han desarrollado formatos comprimidos para el almacenamiento de las imágenes [4]. Estos utilizan diferentes técnicas para reducir el espacio de almacenamiento, algunas son: Método de árboles de Huffman, códigos de empacado por repetición de vecinos (RLE), manejo de auto referencias (Métodos LZ77, LZ78 y LZW) y métodos basados en la Transformada de Fourier, éstos últimos son los más eficientes en términos generales.

Algunos formatos son públicos y abiertos, mientras que otros son del tipo privado y su incorporación en las aplicaciones requiere de pago de derechos.

Para el manejo de los formatos de imágenes se requiere conocer la manera en que está almacenada la información y hacer la decodificación hasta tener la imagen en forma de matriz y separada en canales, esto facilitará el procesamiento de la misma.

1.4. Histograma de una Imagen

Dada una imagen es posible contar el número de pixeles que corresponden a cada tono en cada canal, a la representación gráfica de esta característica se le llama (como en estadística descriptiva) el Histograma del canal. Para el caso de imágenes en tonos de gris solo existe un histograma.

El algoritmo para calcular las poblaciones **H[k], k= 0 .. Λ** del canal **c** para una imagen de **n** columnas y **m** renglones es el siguiente:

```
// Algortimo Tabla - Histograma

// Limpia sumas
for (k=0, Λ-1) {H[k] = 0}
// Calcula poblaciones por tono
for (y=0, m-1)
  for (x=0, n-1)
     H[ I[x,y,c] ] = H[ I[x,y,c] ] + 1
```

En la figura (Fig. 1.5) siguiente se muestran tres imágenes y sus histogramas correspondientes.

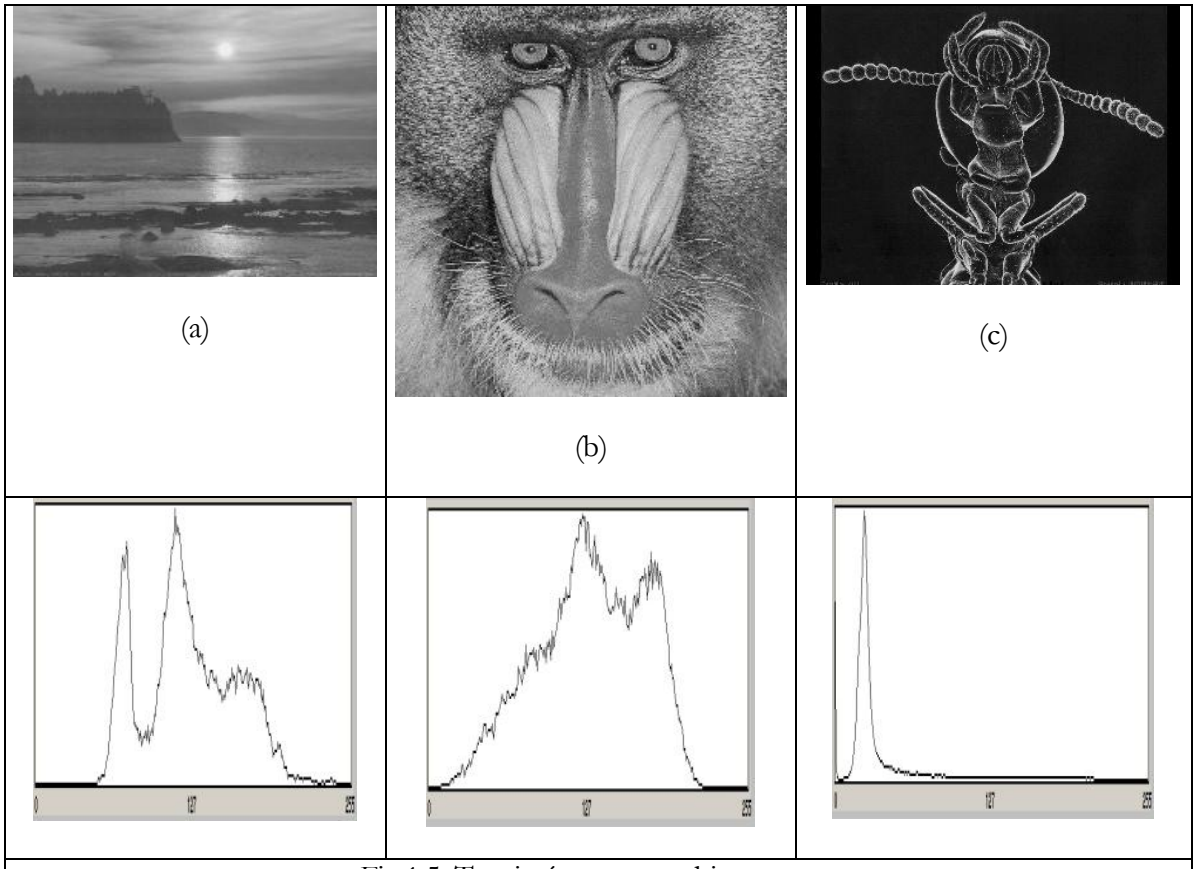


Fig 1.5. Tres imágenes y sus histogramas

En la Figura 1.5, puede notarse que el histograma de la imagen (a) presenta tres picos y se encuentra centrado, mientras que el de la imagen (b) solo tiene dos picos y esta cargado a los tonos claros (la media esta por encima de 127) y el de la imagen (c) tiene un solo pico marcado muy cerca del origen (tonos muy oscuros) y la estructura de los tonos que forman al animal es muy pobre. Es posible refinar los detalles usando una escala **Log(z+1)** para el eje vertical en la Fig. 1.6 se muestra la gráfica con ésta transformación.

Esta técnica permite observar detalles en la estructura de información del histograma que a simple vista no son notorios y ayuda a encontrar detalles que eventualmente son relevantes en las imágenes.

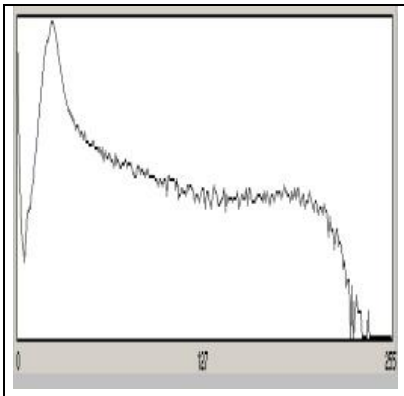
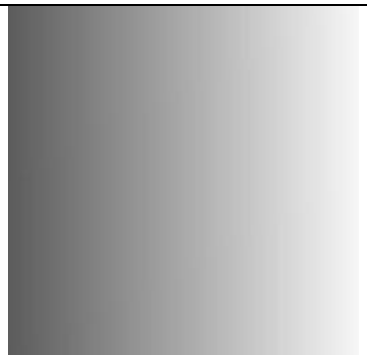
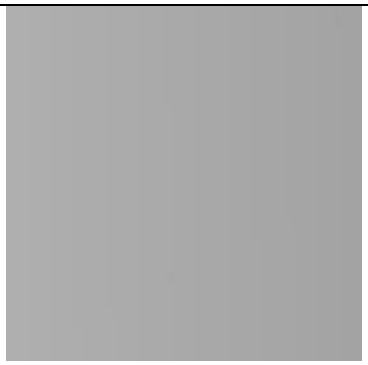


Fig. 1.6. Histograma de la imagen 1.5c usando una escala logarítmica en el eje vertical.

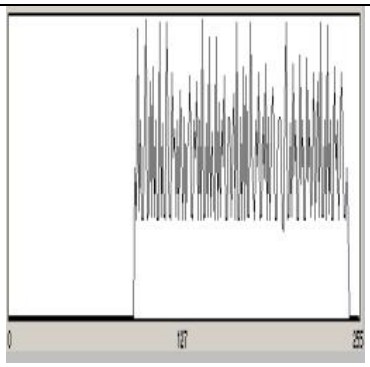
Diremos que una imagen presenta contraste si existe una diferencia entre los tonos que la componen y que su contraste es bajo sí es difícil distinguir entre los elementos que la componen. En la figura 1.7 se muestran dos imágenes y sus histogramas.



(a)



(b)



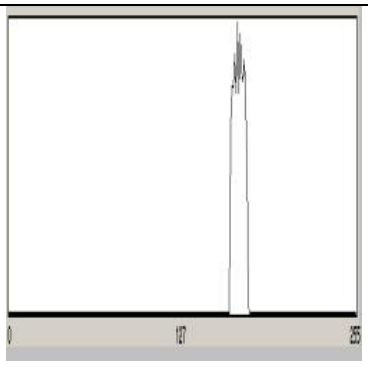


Fig. 1.7. Dos imágenes. La imagen (a) presenta un contraste alto y la (b) uno bajo.

La imagen 1.7 a corresponde a un degradado horizontal lineal con valores que inician en 90 aproximadamente y termina en 245, pude notarse que los tonos son fáciles de discriminar. Mientras que la imagen 1.7 b se forma por un degradado similar que inicia en 175 y termina en 163, dado que los tonos

son muy próximos la separación de ellos por el ojo es difícil y parece que se trata de una imagen de un solo tono, puede verse en su histograma que la dispersión de tonos es muy pequeña, pero no es nula. Este ejemplo ilustra el fenómeno de contraste visual.

1.5. Imágenes y visión

El rango en el cual el ojo humano es capaz de registrar una imagen óptica esta limitado por las capacidades del mismo. La cantidad física que permite observar corresponde a la cantidad de luz que llega al ojo y a la longitud de onda de ella. El ojo humano normal es capaz de percibir señales luminosas entre el rojo y el violeta, es la siguiente gráfica se muestra la posición de los colores y su longitud de onda en nanómetros (10^{-9} m).

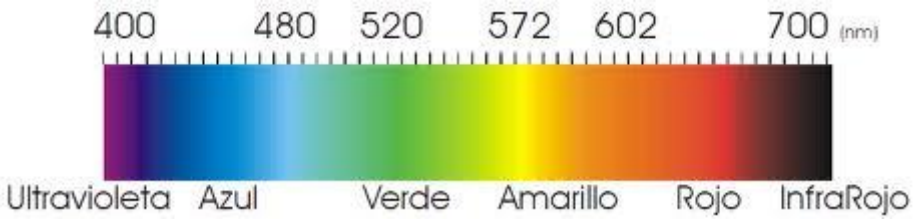


Fig. 1.8 Colores y su posición en el espectro

Este pequeño rango es lo que llamamos espectro visible. En la siguiente figura se muestran otros fenómenos electromagnéticos y sus respectivas longitudes de onda en micras.

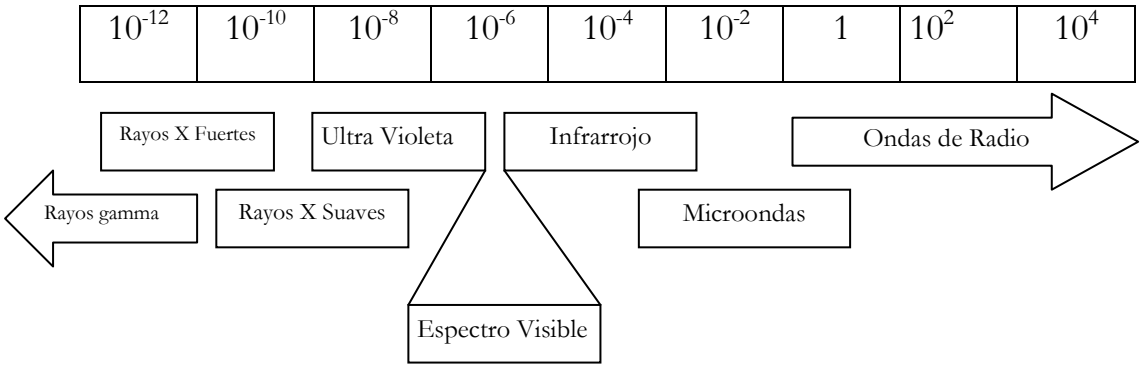


Fig. 1.9. Ubicación de diferentes tipos de radiación en el espectro electromagnético

En general se podrán formar imágenes con radiación en cualquier parte del espectro, o que cambia lógicamente serán los métodos de adquisición y registro. Ejemplos clásicos son las radiografías médicas, que se trabajan en la banda de Rayos X y los sistemas de visión nocturna basados en Luz infrarroja. En astronomía se utiliza la banda de ondas de radio para registrar los fenómenos en estrellas y galaxias lejanas, a ésta rama de la astronomía se le llama Radio astronomía.

Otros sistemas pueden registrar otro tipo de variables y traducirlas a una matriz que se normaliza y se puede presentar la información en forma de imagen visual, más la interpretación se debe hacer según la(s) variable(es) originales. Algunos ejemplos en este caso son:

- Termografía. Registro de perfiles de temperatura en sistemas vivos o inertes.
- Ecosonografía. Registro del movimiento mediante el efecto Doppler usando ultrasonido.

- RMN. Generación de imágenes mediante el fenómeno de Resonancia Magnética Nuclear.

- AFM. Registro de las propiedades microscópicas de las estructuras cristalinas moleculares mediante sistemas de Fuerza Atómica.

- TAC. Reconstrucción del interior de un sistema sin intrusión mediante técnicas de Tomografía Axial.

- XRD. Recolección de información de sistemas cristalinos mediante la difracción de Rayos X.

- Sonar. Construcción de perfiles de profundidad en el mar mediante ondas sonoras y su eco.

- EM. Registro de imágenes submicroscópicas mediante la emisión de electrones. Esta técnica se conoce como microscopía electrónica y permite observar objetos que la luz ya no puede resolver ópticamente.

En todos los casos se requiere de un sistema que haga la medición de la propiedad de interés sobre una malla o cuadrícula del sistema, de tal manera que se forme una matriz de datos de uno o muchos canales. Cuando hay presente mas de un canal en los datos de dice que se tiene una imagen multiespectral. Las combinaciones multiespectrales son útiles en la solución de muchos problemas, ya que permiten obtener información complementaria, la cual tratada de manera adecuada permite revelar comportamientos o características singulares en diversos sistemas.

Capítulo 2.

Operaciones Orientadas al Punto

Fundamentos

Las operaciones orientadas al punto transforman a la imagen modificando un pixel a la vez, en general sin importar el estado de los pixeles vecinos. La transformación se puede aplicar a toda la imagen o a una región de ella.

Sea **x** un pixel de una imagen **I**, es decir $\mathbf{x} \in \mathbf{I}$. Supongamos que la función $\mathbf{y} = f(\mathbf{x})$ transforma a un pixel **x** mediante la regla *f*, generando un nuevo valor para él, digamos **y**. Entonces diremos que la nueva imagen **I'**, donde $\mathbf{y} \in \mathbf{I'}$, es el producto de aplicar *f* sobre **I**. Simbólicamente diremos que,

$$\mathbf{I'} = f(\mathbf{I}). \qquad (2.1)$$

El proceso de transformación en la mayor parte de los casos será tal que

$$\text{sí } \mathbf{x} = \mathbf{I}[i, j] \Rightarrow \mathbf{y} = \mathbf{I'}[i, j], \text{ donde } \mathbf{y} = f(\mathbf{x}).$$

Para que la transformación *f* no ocasione problemas de representación, si el dominio de **x** está en el intervalo $\boldsymbol{D} = [0, L - 1]$, donde $L = 2^p$, donde *p* es la profundidad en bits de la imagen, entonces se va a exigir que $\mathbf{y} \in \boldsymbol{D'}$, donde en general $\boldsymbol{D'} \subseteq \boldsymbol{D}$. Lo cual implica que el mecanismo de representación de la imagen sobre elementos de la clase **x**, seguirá siendo válido para la clase a la que pertenece **y**. Esta condición permite que los métodos desarrollados para la visualización de la imagen **I** se pueden utilizar para **I'**.

El algoritmo básico de transformación bajo *f* para una región rectangular de **I** definida por

$$R = [i1 \ldots i2, j1 \ldots j2],$$

es el siguiente:

```
for i = i1, i2 {
  for j = j1, j2 {
     I'[i, j] = f ( I[i, j] )
  }
}
```

fig. 2.1. Algoritmo básico para transformar una región de una imagen **I** bajo *f*.

En el caso que: i1 = 0, i2 = M (donde M = Imagen.Ancho-1), j1 = 0, N (donde N = Imagen.Alto-1); el proceso modificará a toda la imagen.

Al cambiar *f* la transformación será diferente. Si definimos la composición de transformaciones de la manera habitual, tendremos que:

$$f_1 \circ f_2 (\mathbf{I}) = f_1 (f_2 (\mathbf{I})). \qquad (2.2)$$

En general al aplicar dos transformaciones a una imagen en diferente orden, no se debe esperar que la imagen resultante sea la misma, es decir, la composición de transformaciones no es conmutativa, simbólicamente tendremos que:

$$f_1 \circ f_2 (\mathbf{I}) \neq f_2 \circ f_1 (\mathbf{I}). \qquad (2.3)$$

Definiremos una *batería* o *serie* de transformaciones f_k mediante la composición de ellas. Muchas de las operaciones de mejora de la imagen, detección de bordes, etc., se definen como una batería. El sentido de ésta es similar a la composición de las funciones que generan cada transformación. Sean $f_1, f_2, \ldots, f_n$ las funciones que definen cada proceso sobre la imagen, entonces la transformación compuesta o batería será:

$$F(\mathbf{I}) = f_1 \circ f_2 \circ \ldots \circ f_n(\mathbf{I}) = f_1(f_2(\ldots f_{n-1}(f_n(\mathbf{I}))\ldots)). \qquad (2.3)$$

Gráficamente podemos representar el proceso de transformación múltiple mediante celdas, donde cada celda representa una transformación o *filtro*. La figura siguiente (fig. 2.2) ilustra la situación.

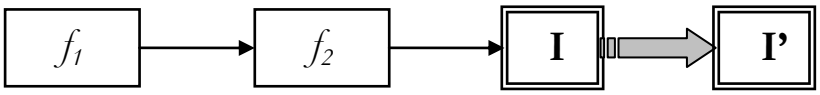


fig. 2.2. Representación gráfica de la composición de procesos.

2.1. Operaciones Elementales

Introduciremos ahora algunas operaciones simples sobre la imagen. Sea **I** una imagen en colores, con un dominio [0, L] para los valores del tono de cada canal para los pixeles, en la representación RGB estándar, de ancho *M* y alto *N*.

La operación más simple es la *Identidad*, ésta deja a la imagen igual. Podemos usar ésta para realizar por ejemplo copias de una imagen. La función correspondiente es: $\mathbf{y} = \mathbf{x}$, de donde $f(\mathbf{x}) = \mathbf{x}$. Si representamos ésta función de manera gráfica visualizaremos una ecuación de mapeo lineal simple (fig. 2.3). Ésta función nos indica que el tono *w* es mapeado al tono *w'*.

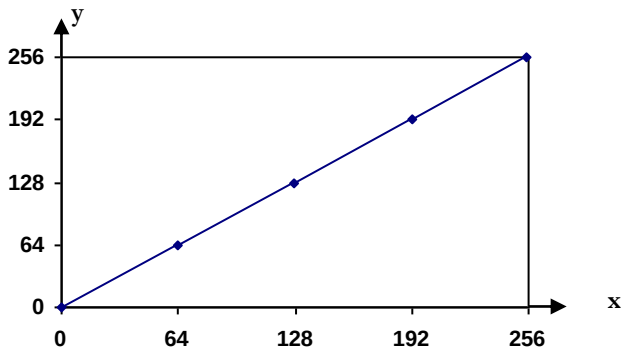


Fig. 2.3. Transformación identidad entre las variables: $\mathbf{y} = f(\mathbf{x}) = \mathbf{x}$, (p=8, L =255).

2.1.1. Negativo de una imagen.

Una transformación muy simple es el ***negativo***, ésta se construye de alguna de las siguientes maneras, sea $\mathbf{x} = (r, g, b)$ un pixel de la imagen **I**, entonces el negativo de **x** se puede hallar simplemente como:

$$\mathbf{x'} = (\sim r, \sim g, \sim b) = (\Lambda - r, \Lambda - g, \Lambda - b), \qquad (2.4)$$

donde $\Lambda = L - 1$ y $L = 2^p$.

Este proceso es muy claro de entender para una imagen monocromática, pues en ella $p = 1$, y entonces L = 2 y en consecuencia Λ= 1, de donde dado que los valores permitidos para un pixel (en cada plano) serán únicamente x = {0, 1}, de donde las transiciones serán: 0 → 1 y 1 → 0. De donde un pixel en 0 (negro) se transformará en 1 (blanco) y viceversa.

De forma gráfica para cada canal el negativo se puede interpretar como una línea de transformación con pendiente negativa (fig. 2.4).

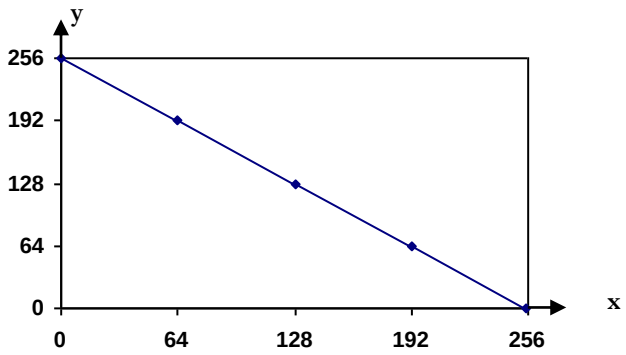


Fig. 2.3. Transformación de la función negativo para un canal x = 255 – x. (p=8)

En el siguiente ejemplo se aplica la transformación a una imagen en tonos de gris (r=g=b=z), donde el negativo de un pixel **x** = (z, z, z) será **x'** = (~z, ~z, ~z). En la figura 2.4. se muestra el proceso sobre dos imágenes de ejemplo una monocroma y otra en tonos de gris.

	Imagen Original	Negativo de la Imagen
Imagen Monocroma		

Imagen en tonos de gris		

Fig. 2.4. Imágenes y sus Negativos

Para observar este fenómeno en color puede realizarse la transformación en la pantalla de una computadora. Un ejemplo de aplicación de éste filtro es el "revelado" de un negativo fotográfico. Si uno escanea el negativo de una foto mediante un escáner abierto auxiliándonos con una lámpara o con un dispositivo especializado, entonces se puede revelar éste usando el filtro de "Negativo" antes descrito sobre los tres canales (fórmula 2.4).

2.1.2. Transformaciones funcionales.

En general, uno puede pensar en una función que transforme cada canal $z' = f_i(z)$, donde z puede ser cualquiera de los tres canales que forman a una imagen en color $z \in \{r, g, b\}$. La transformación más general tendrá la forma:

$$\mathbf{x'} = (r', g', b') = F(\mathbf{x}) = (f_1(r, g, b), f_2(r, g, b), f_3(r, g, b)), \quad (2.5)$$

donde las f_i (i=1, 2, 3) son funciones de salida entera cualesquiera. En general la respuesta en un canal puede depender de los valores de entrada en los demás canales considerándolo a él mismo. Los procesos más simples de interpretar son aquellos en los cuales la respuesta para un canal solo depende de la entrada en él, a estos les llamaremos directos, es decir

$$\mathbf{x'} = (r', g', b') = F(\mathbf{x}) = (f_1(r), f_2(g), f_3(b)). \quad (2.6)$$

Si además las formas funcionales de las f_i son iguales, tendremos una transformación directa simétrica, a estos filtros les llamaremos directos simétricos, su forma es

$$\mathbf{x'} = (r', g', b') = F(\mathbf{x}) = (f(r), f(g), f(b)). \quad (2.7)$$

Para los casos (2.6) y (2.7) se van a analizar algunas situaciones que se producen cuando las funciones f, f_i son monótonas crecientes o decrecientes y además tocan los puntos de referencia (0, 0) y (Λ, Λ).

En general las transformaciones funcionales se denominan "**Transformaciones u operaciones puntuales**", esto se debe a que aplica aun punto de la imagen en la posición (i,j) y generan un nuevo valor que se asigna a la imagen transformada en su coordenada (i,j). A las transformaciones en el lenguaje del procesamiento de imágenes se les denominan **filtros** por su análogo en la óptica.

Una transformación básica en esta familia es el filtro de corrección de luz o corrección gamma (γ), también por su forma se le llama función potencia.. Ésta tiene la forma normalizada

$$z' = \Lambda\left(\frac{z}{\Lambda}\right)^{\gamma}, \qquad (2.8)$$

donde γ es un real positivo y $\mathbf{z} \in [0, \Lambda]$, por lo cual también $\mathbf{z'} \in [0, \Lambda]$. Se pueden identificar dos clases de transformaciones: la primera corresponde al caso cuando $\gamma \in (0, 1]$. Estas transformaciones van a realizar un proceso de "aclarado de la imagen", si observamos las siguientes gráficas (Fig. 2.5), podemos ver que las funciones son monótonas crecientes y están sobre la línea de la identidad, de donde para todos los valores de **z** (excepto el blanco y el negro) **z'** estará arriba de la identidad, es decir el tono será más claro.

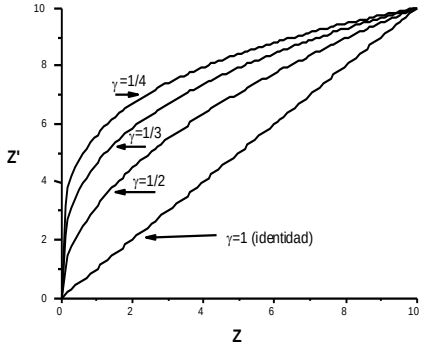


Fig. 2.5. Función potencia $11(z/11)^{\gamma}$

El segundo caso corresponde a la situación cuando $\gamma > 1$. Lo que sucede ahora es que todos los puntos están por debajo de la identidad, por lo tanto la transformación "obscurecerá la imagen" (Fig. 2.6.) en consecuencia.

A esta transformación (para los valores de $\gamma > 0$) se le llama *corrección de gamma*, debido a la letra que se utiliza en la función potencia.

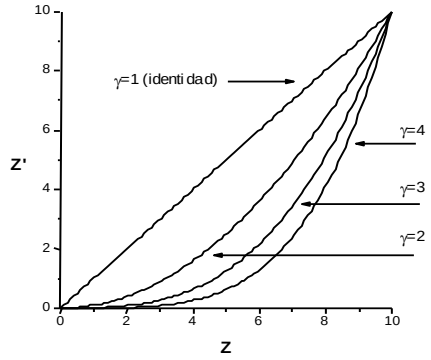


Fig. 2.6. Función potencia $11(z/11)^{\gamma}$

En la siguiente tabla se muestra el resultado de aplicar ésta transformación para una imagen, usando los valores $\gamma = ½$ y $\gamma = 2$.

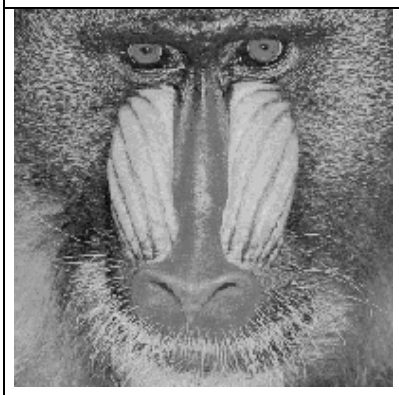
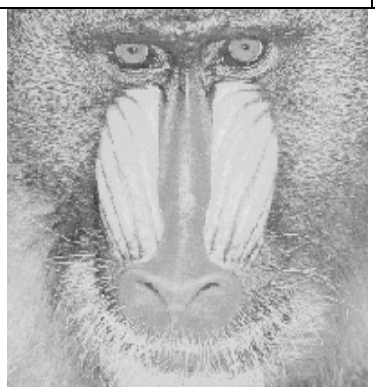
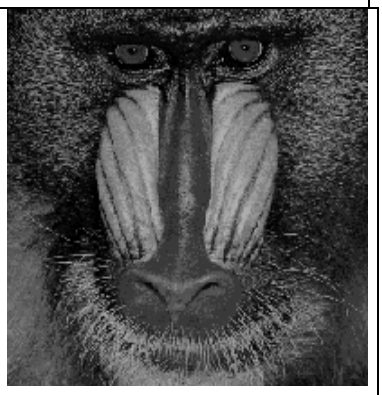
Original	$\gamma = ½$	$\gamma = 2$
		

Fig. 2.7. Aplicación de la Función potencia

A continuación se presentan una serie de filtros clásicos organizados según su efecto sobre la imagen.

Filtros de aclarado

En este grupo se puede construir una amplia familia. El efecto de ésta es el corrimiento de los tonos hacia el blanco usándose diferentes reglas.

En general puede notarse que la curva que las funciones analíticas o definidas en tablas tienen las siguientes propiedades:

$$\left.\begin{aligned} f(0) &= 0 \\ f(z) &> z \\ f(\Lambda) &= \Lambda \end{aligned}\right\} \qquad (2.9)$$

a) *Función Logarítmica.* Se conoce también como transformación de rango dinámico generalizada, introduciendo un parámetro α se puede modificar la curva de respuesta para incrementar el efecto de aclarado para los tonos obscuros. Su forma analítica es la siguiente.

$$z' = A\ln(\alpha z + 1), \ \alpha > 1, z \in [0, \ \Lambda]. \qquad (2.10)$$

Es claro que cuando $z=0 \Rightarrow z' = 0$, para determinar A pediremos que $z'=\Lambda$ cuando $z=\Lambda$, de donde donde $\mathbf{A = \Lambda/\ln(\alpha\Lambda+1)}$. La razón del porque sumar uno a **z** en el argumento radica en el hecho que los valores de **z** comienzan en "cero" y para este valor el logaritmo no está definido.

La curva de respuesta de ésta transformación se muestra a continuación (Fig. 2.8.), se utilizaron los valores de $\alpha=1$, $\Lambda=31$ y $\mu = 31/\ln(32)$. Puede verse que es una curva que aclara mas fuertemente que Función Potencia antes discutida. Ésta función se utiliza para aclarar imágenes obscuras y aumentar el contraste. A ésta transformación también se le llama Función de Corrección de Rango Dinámico.

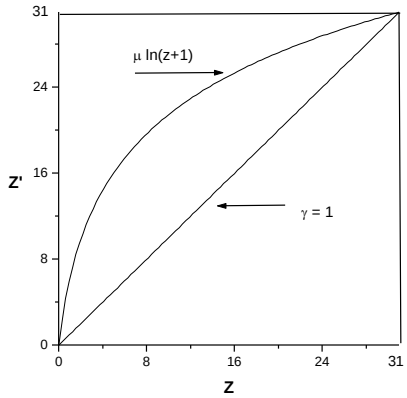


Fig. 2.8. Función logarítmica $\mu \ln(z + 1)$

En la tabla siguiente (Fig. 2.9.) se muestra la aplicación a un par de imágenes.

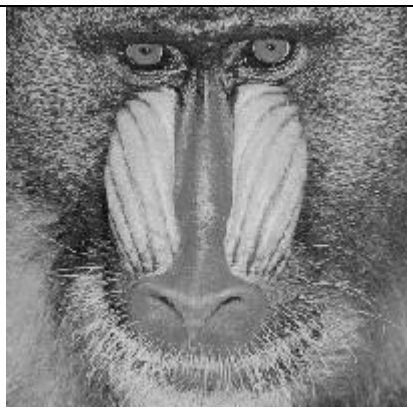
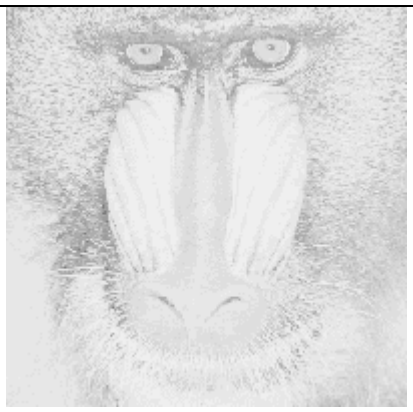
Original	Función logarítmica
	
	

Fig. 2.9. Aplicación de la función de corrección de rango dinámico.

b) *Función seno*. Es posible construir una función que aclare a partir de la función seno en el intervalo $[0, \pi/2]$. Se selecciona este intervalo con el propósito de aprovechar que en él la función es monótona creciente. La transformación tiene la forma general:

$$z' = \mu \, sin(kz), \qquad (2.10)$$

para realizar la normalización, pediremos que para ($\mathbf{z = \Lambda}$) $\Rightarrow$ ($\mathbf{z' = \Lambda}$). Dado que la función es uno en $\pi/2$, entonces $k = \frac{\pi}{2\Lambda}$.

Y $\mu = \Lambda$, por lo tanto la función normalizada en el cuadrado $(0, \Lambda)\times(0, \Lambda)$ tendrá la forma:

$$z' = \Lambda \, sin\left(\frac{\pi \, z}{2\Lambda}\right). \qquad (2.11)$$

c) *Función exponencial.* Otra forma de obtener un filtro de aclarado es tomar la curva de carga de un condensador en un circuito RC en serie. La forma funcional es

$$z' = A\left(1 - e^{-\alpha z/\Lambda}\right), \text{ donde } \alpha > 0, z \in [\,0, \Lambda]\,. \qquad (2.12)$$

Es fácil verificar que en $z = 0 \Rightarrow z' = 0$. La función tiende a A cuando z crece, para determinar el valor de A pediremos que en $z = \Lambda \Rightarrow z' = \Lambda$, de donde $A = \Lambda/(1 - e^{-\alpha})$.

Filtros de oscurecimiento.

La curva que las funciones analíticas o definidas en tablas que oscurecen a una imagen tienen las siguientes propiedades:

$$\left.\begin{aligned} f(0) &= 0 \\ f(z) &< z \\ f(\Lambda) &= \Lambda \end{aligned}\right\} \qquad (2.13)$$

a) *Función cosenoidal.* Utilizando argumentos similares a los utilizados para la función senoidal, se puede construir una función cosenoidal normalizada, dado que en el mismo intervalo el coseno varía de uno a cero producirá un efecto de "negativo", por lo cual es conveniente realizar una inversión vertical de la transformación pura y construir una función monótona creciente. La forma final es:

$$z' = \Lambda\left(1 - \cos\left(\frac{\pi \, z}{2\Lambda}\right)\right), \; z \in [\,0, \Lambda] \qquad (2.12)$$

La curva de transformación se muestra en la figura siguiente (Fig. 2.10.). Puede observarse que la función cosenoidal esta por debajo de la identidad, por lo que tiene un efecto de oscurecimiento. En el ejemplo se ha tomado $\Lambda = 11$.

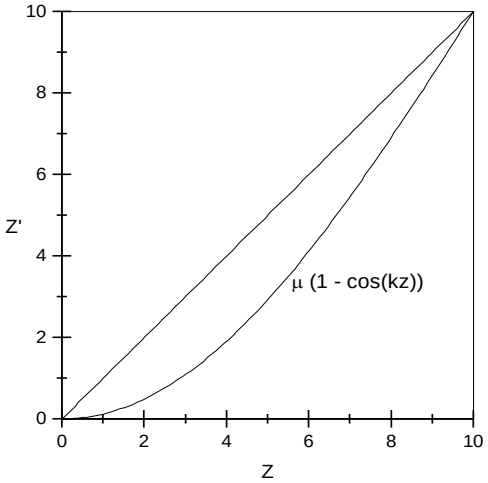


Fig. 2.10. Función coseno normalizada

b) Un filtro de oscurecimiento "fuerte" se puede construir con una exponencial creciente y ajustada, su forma funcional es

$$z' = A\left(e^{\alpha z/\Lambda} - 1\right),\ \alpha > 0,\ z \in [\,0,\ \Lambda] \qquad (2.13)$$

Luego de normalizar se puede mostrar que $A = \Lambda/(e^{\alpha} - 1)$.

Filtros definidos por segmentos lineales.

Esta clase define una interesante variedad de transformaciones. Comenzaremos discutiendo el caso de una función continua definida por un punto móvil en el cuadro de trabajo $(0, \Lambda) \times (0, \Lambda)$. La gráfica de la transformación se muestra en la figura 2.11.

Al cambiar la posición de punto (z_o , z'_o) se podrán producir diferentes efectos. Por ejemplo en la posición en que se encuentra el punto en la figura 2.11., este realizará un proceso de oscurecimiento en general. Si el punto se hallara arriba de la identidad, entonces el resultado de la función será de aclarado. La implementación de la función se puede hacer mediante la construcción de las ecuaciones de las dos rectas que la forman y verificando simplemente si el tono **z** del pixel de entrada es menor o mayor que $\mathbf{z_o}$. Y a continuación evaluar el tono de salida usando la ecuación de la recta correspondiente.

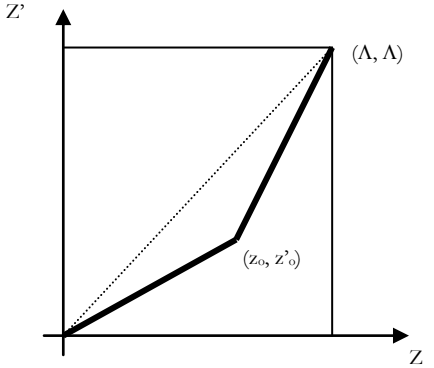


Fig. 2.11. Función Lineal con un punto móvil.

Otro caso corresponde al de tres segmentos de recta definidos por dos puntos (z_1, z'_1) y (z_2, z'_2). La transformación tiene la forma presentada en la figura 2.12. Ésta es mucho más versátil que la de un punto, ya que permite crear varias situaciones. Se pueden definir algunos casos particulares notables.

Compresión de rango. $z'_1 = 0$, $z'_2 = \Lambda$. La transformación ocasiona un estrechamiento en el rango de respuesta de la iluminación. La recta queda definida por los puntos z_1 y z_2. La recta central pasa por los puntos $(z_1, 0)$ y (z_2, Λ), de donde su ecuación será:

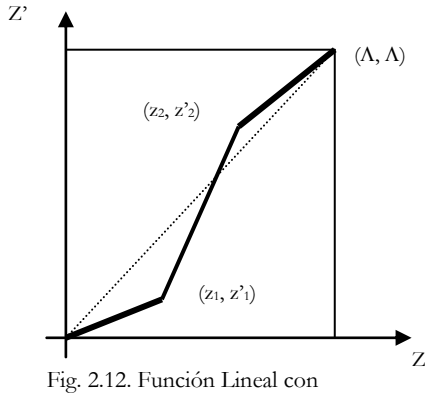


Fig. 2.12. Función Lineal con dos puntos móviles.

$$z'(z) = mz + b.$$

donde $m = \Lambda/(z_2 - z_1)$ y $b = -\Lambda z_1/(z_2 - z_1)$.

Que satisface las condiciones predefinidas: $z'(z_1) = 0$ y $z'(z_2) = \Lambda$. Las cuales se pueden comprobar por simple inspección. Las relaciones que define el algoritmo son:

$$z' = \begin{cases} z < z_1 & \Rightarrow \quad z' = z \\ z_1 \le z \le z_2 & \Rightarrow \quad z' = \Lambda(z - z_1)/(z_2 - z_1) \\ z > z_2 & \Rightarrow \quad z' = z \end{cases} \qquad (2.14)$$

Filtros selectivos claro - oscuro.

Estas transformaciones tienen la característica de tratar a los tonos oscuros de manera inversa que a los tonos claros. Se pueden clasificar en dos grupos: de aumento o disminución del contraste.

Los que aumentan el contraste tienen como caso extremo el filtro de binarización o alto contraste. Este modela una función de clasificación categórica con la forma:

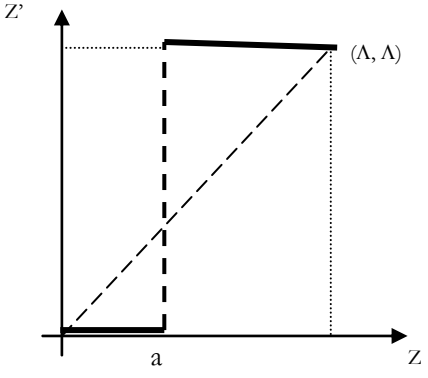


Fig. 2.13. Filtro de Alto Contraste

$$z' = \begin{cases} 0 & z < a \\ \Lambda & z \ge a \end{cases} \qquad (2.15)$$

Como se puede observar, ésta transformación convierte a la imagen en una imagen binaria o bitonal, en la cual aparecerán sólo los tonos **blanco** y **negro**.

Como casos intermedios y menos tajantes, tenemos por ejemplo la transformación **sigmoide**. Su forma gráfica se muestra en la figura 2.14. En la parte derecha de la figura se muestran las propiedades de la función.

Puede notarse que los tonos oscuros ($z < a$) se oscurecerán mas y los claros ($z > a$) se aclararán mas, éste comportamiento lógicamente aumentará el contraste, es decir la separación (distinción) entre claros y oscuros crecerá en la imagen transformada por la función sigmoide..

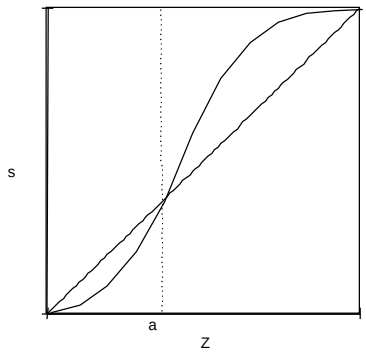


Fig. 2.14. Función Sigmoide

$\sigma(0) = 0$; $\sigma(\Lambda) = \Lambda$

$\sigma(z) < z$ si $z < a$

$\sigma(z) = z$ si $z = a$

$\sigma(z) > z$ si $z > a$

Una primera propuesta para construir una sigmoide es "montar" una función senoidal invertida "sobre" una recta a 45° (es decir la identidad $z' = z$), la forma explícita de la función es

$$z' = z - \lambda \sin(2\pi z/\Lambda), \text{ donde } \lambda > 0.$$

Se puede probar que la transformación cumple con las características que debe cumplir una sigmoide con $a = \Lambda/2$. El efecto sobre el contraste se puede regular variando el valor de λ. Deben vigilarse los valores de λ para evitar que la salida de la función se mantenga en el intervalo $[0, \Lambda]$. En la figura siguiente (Fig 2.15) se presenta una gráfica de la Transformación sigmoide senoidal, donde $\lambda 1 > \lambda 2$. En particular cuando $\Lambda = 255$, los valores para λ que mantienen la salida en el rango correcto deben estar entre (0, 40].

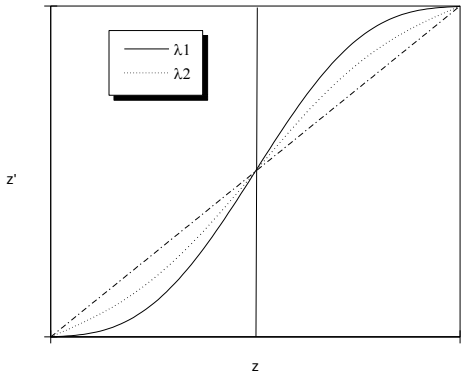
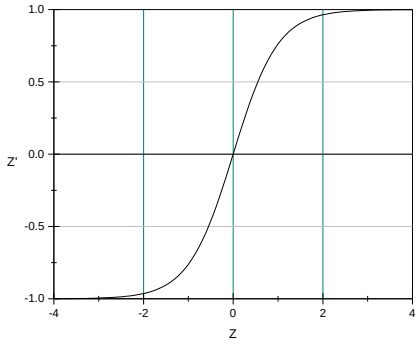


Fig. 2.15. Sigmoide senoidal

Podemos construir otra forma sigmoide basándonos en una función del tipo *tangente hiperbólica*, ésta función en su forma natural se define como

$$z' = tanh(z) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}, \qquad (2.16)$$

Fig. 2.16 Tangente Hiperbólica natural



En la Figura 2.16 se muestra la gráfica de ésta función. Puede notarse que su respuesta está acotada en el intervalo [-1, 1], tiene su cambio de signo en cero y es una función monótona creciente.

Si se hace un desplazamiento vertical de una unidad, otro horizontal de $\Lambda/2$, se escalan z y z' y se introduce un parámetro para controlar la inclinación de la curva, llegamos a la forma

$$z' = \frac{\Lambda}{2}\left[1 + tanh\left[\alpha\left(z - \frac{\Lambda}{2}\right)\right]\right], \alpha > 0 \qquad (2.17)$$

Puede verificarse que con valores adecuados de α

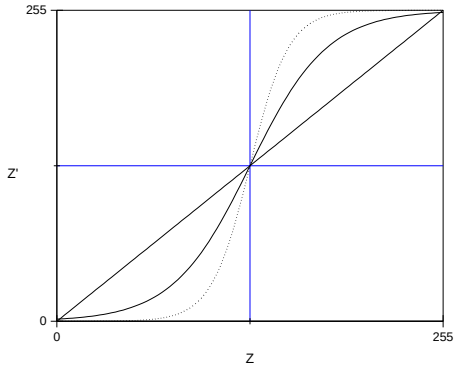
$$z'(z=0) = \frac{\Lambda}{2}\left(1 + tanh\left(-\frac{\alpha\Lambda}{2}\right)\right) \approx 0,$$

$$z'(z=\Lambda/2) = 0,$$

$$z'(z=\Lambda) = \frac{\Lambda}{2}\left(1 + tanh\left(\frac{\alpha\Lambda}{2}\right)\right) \approx \Lambda.$$

Esta transformación se convierte en una función "cuasi - escalón" para valores "grandes" de α, y su forma es prácticamente la del filtro de Alto Contraste. En la figura 2.17 se muestra la transformación

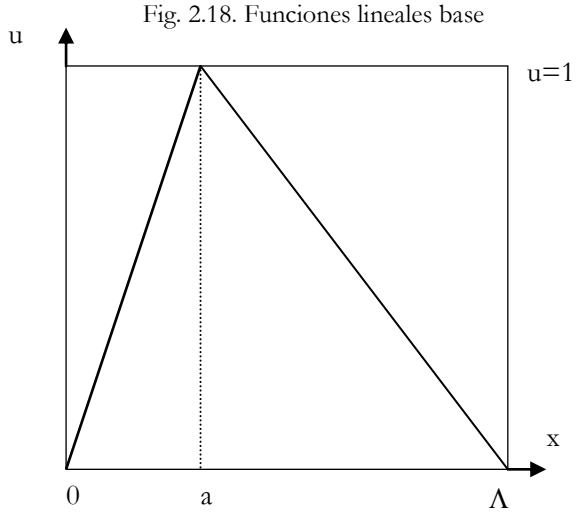
Fig. 2.17 Tangente Hiperbólica Λ=255



para varios valores del parámetro α. La línea sólida corresponde a α=5 y la punteada a α=10.

Puede notarse que los ejemplos anteriores tienen su punto de inflexión respecto a la identidad en $\Lambda/2$, es decir el umbral para transformar a los oscuros y los claros se encuentra en el punto medio del rango de entrada **z**. Un caso que permite parametrizar el punto de umbralización **a** se puede construir mediante una función base lineal definida por pedazos, integrándola y normalizándola. Este modelo permite controlar la posición del punto **a** de la sigmoide. Partiendo del modelo presentado en la figura 2.18,

Fig. 2.18. Funciones lineales base



las funciones base lineales descritas como u=u(x), son

$$u = x/a\ ,\ 0 \le x \le a,$$

$$u = (\Lambda - x)/(\Lambda - a),\ a \le x \le \Lambda.$$

Puede verificarse fácilmente que existe continuidad en la función cuando x=a. Integrando

ésta función, definiremos la sigmoide con control de **a** como

$$y_a(x) = \int_0^x u(v)dv .$$

Puede notarse que la función implícita depende del límite superior de la integral. Para encontrar una forma explícita de la función debemos realizar la integración, ya que la función esta definida por pedazos, entonces será necesario separarla en dos integrales, la primera de **0** a **a** y la segunda de **a** a Λ. Y el resultado dependerá del valor de **x**, es decir si **x** ≤ **a** o bien si **x** > **a**.

En el primer caso (**x** ≤ **a**) tendremos que

$$y = \int_0^x u(v)dv = \int_0^x \frac{u}{a}dv = \frac{x^2}{2a},$$

o sea $y = \frac{x^2}{2a}$, cuando **x** ≤ **a**. Es claro que lo que hemos hallado es una parábola abierta hacia arriba que pasa por el origen es simétrica respecto al eje vertical.

Ahora cuando **x** > **a**, dado que el punto de discontinuidad ha sido superado, tendremos que

$$y = \frac{a}{2} + \int_a^x \frac{\Lambda - v}{a - v}dv = \frac{a}{2} + \frac{x-a}{2(\Lambda - a)}(2\Lambda - x - a).$$

Es simple hallar el área bajo la curva de todo el triángulo (S), basta con hacer x=Λ, haciendo la evaluación tendremos que

$$S = \frac{a}{2} + \frac{\Lambda - a}{2} = \frac{\Lambda}{2}.$$

Para normalizar el área total S a Λ con el propósito de que y(x=Λ)=Λ baste con multiplicar por un factor 2 la función. Con esto hemos completado el ciclo de construcción y tenemos la transformación deseada, ésta se muestra en la figura 2.19 (Λ=10, **a**=4) y a la derecha está el filtro paramétrico analítico.

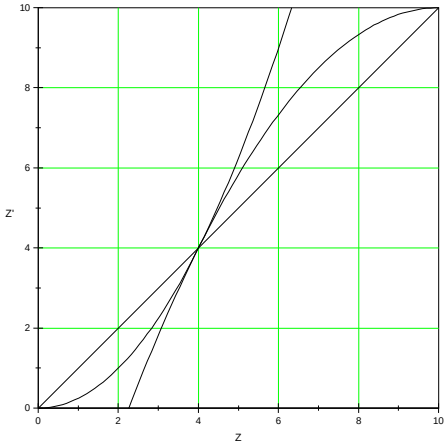


Fig. 2.19 Sigmoide parabólica con control **a**.

$$z' = \begin{cases} \dfrac{z^2}{a} & 0 \le z \le a \\ a + \dfrac{z-a}{\Lambda - a}(2\Lambda - z - a) & a < z \le \Lambda \end{cases}$$

Un ejemplo interesante de aplicación de éste filtro es el caso de las huellas digitales humanas de las yemas de los dedos, en la figura se muestra una huella típica (2.20 a) y su histograma (2.20 b). Puede notarse que el punto de umbralización no corresponde a la mitad del rango de la

escala de grises, se observa que a > 127, tomado el valor de **a** del histograma a mano se tiene que **a** ≈ 197, si aplicamos el Filtro Sigmoide Triangular antes desarrollado con el valor de a en 197, se obtiene una imagen de la huella con mayor contraste entre valles y crestas (2.20 c) a la derecha se muestra el histograma de la imagen filtrada (2.20 d). Es claro que la nueva imagen presenta mejores características para poder realizar la segmentación y reconocimiento de ella.

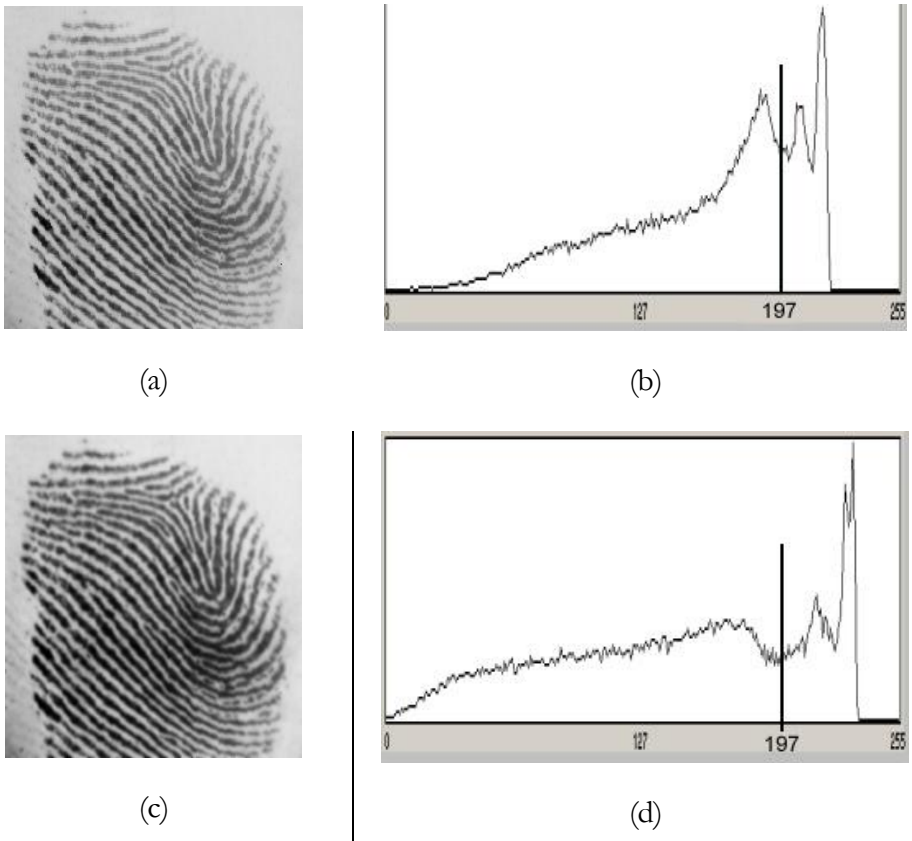


Fig 2.20 Huella Digital Humana, Normal(a) y Filtrada(c).

Otra manera de mejorar la calidad de la imagen para la segmentación consiste en usar un filtro Sigmoide de Tangente Hiperbólica, en la figura 2.21 se muestra la imagen filtrada y su histograma.

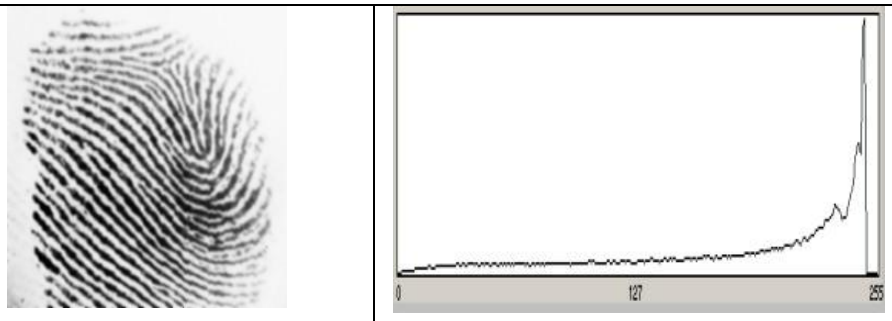
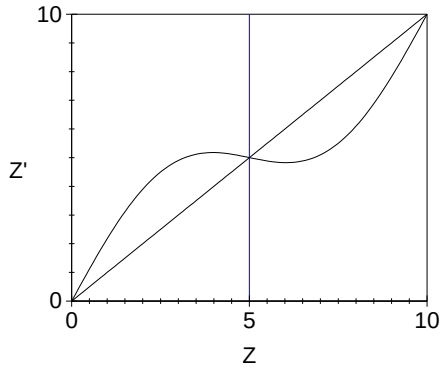
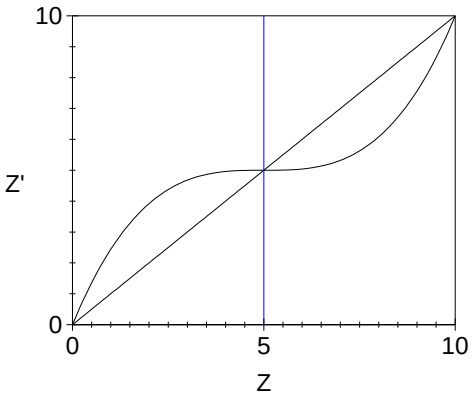


Fig 2.21. Huella Filtrada con una Tangente Hiperbólica con parámetro 5.

A partir de los filtros de aumento de contraste, se puede construir una familia con el efecto inverso, es decir de "reducción de contraste", la curva típica de éstos filtros es la forma mostrada en la figura 2.22. En ésta se muestra una pareja de filtros de la familia.





$z' = z + \lambda \sin(2\pi z/\Lambda)$, $\lambda=2$, $\Lambda=10$	$z' = \frac{\Lambda}{2} + \frac{4}{\Lambda^2}\left(x - \frac{\Lambda}{2}\right)^3$, $\Lambda=10$

Fig 2.22. Funciones típicas de reducción de contraste

En general se pueden construir múltiples funciones analíticas o empíricas para la transformación de una imagen orientada al punto. Estas se pueden interpretar como una tabla de traducción de los tonos de un canal a una nueva curva.

Capítulo 3.

Operaciones Orientadas a la Región

Fundamentos

Las operaciones orientadas a la región transforman a la imagen modificando un pixel a la vez y toman en cuenta para dicha transformación los pixeles vecinos. Y como es natural la transformación se puede aplicar a toda la imagen o a una región de ella.

Diremos que los píxeles vecinos de primer orden son aquellos contiguos a él, en una retícula cartesiana regular un pixel, digamos aquel ubicado en la coordenada (i, j) tiene 8 primeros vecinos, si denotamos por I[i, j] al pixel de referencia en la Figura 3.1 se muestran estos.

I[i -1,j -1]	**I[i, j -1]**	**I[i +1,j -1]**
I[i -1,j]	**I[i, j]**	**I[i +1, j]**
I[i -1 ,j +1]	**I[i, j +1]**	**I[i +1, j + 1]**

Fig 3.1. Primeros Vecinos del Píxel I[i, j]

Muchos filtros regionales utilizan de uno a ocho pixeles vecinos, en particular podemos decir que una transformación regional simple que involucra a los primeros vecinos es una transformación de la forma

$$I'[i, j] = F(I[i+\alpha, j+\beta]),\ \alpha, \beta \in \{-1, 0, 1\}. \qquad (3.1)$$

Existen también transformaciones que consideran vecinos más lejanos.

3.1. Bordes

Las operaciones más simples se refieren a las operaciones de diferencia, las cuales modelan a derivadas bidimensionales discretas. Si consideramos el concepto clásico de derivada unidimensional y consideramos

el hecho que entre dos pixeles la distancia más cercana es de un solo pixel, tendremos por ejemplo que en la dirección horizontal el cambio se puede escribir como:

$$\frac{\delta I}{\delta x} = \frac{I[x+\delta x, y] - I[x, y]}{\delta x}, \qquad (3.2)$$

pero dada la restricción discreta de cercanía entre pixeles, entonces $\delta x = 1$, por lo tanto

$$\frac{\delta I}{\delta x} = I[x+1, y] - I[x, y], \qquad (3.3)$$

mas si quisiéramos hallar los cambios en la imagen mediante la expresión (3.3) debemos tener cuidado ya que si $\delta I/\delta x$ representa una nueva imagen, entonces no puede ser negativa, entonces podemos transformar (3.3) mediante la aplicación de la función valor absoluto, por tanto una transformación que encuentre los cambios en una imagen de manera básica puede escribirse como

$$\frac{\delta I}{\delta x} = \left| I[x+1, y] - I[x, y] \right|. \qquad (3.4)$$

Ahora podemos formular la siguiente pregunta: ¿cuál es el sentido de ésta transformación? La derivada en principio encuentra el cambio local de una función, pero a nivel digital se reduce a medir el cambio entre dos pixeles vecinos como muestra la expresión (3.4). Por lo tanto si aplicamos esta operación a una imagen hallaremos los cambios en la tonalidad de un canal y si en cierta región ésta es constante, entonces el cambio será nulo, más donde haya una transición la derivada digital determinará la tasa cambio. En la figura 3.2 se muestra una imagen y su derivada en la dirección horizontal (x) según la fórmula (3.4), se ha tomado una imagen con pocas tonalidades para hacer claro el ejemplo. Debemos aclarar que en la orilla derecha esta operación no tiene sentido ya que no hay algún pixel con el cual calcular la diferencia.

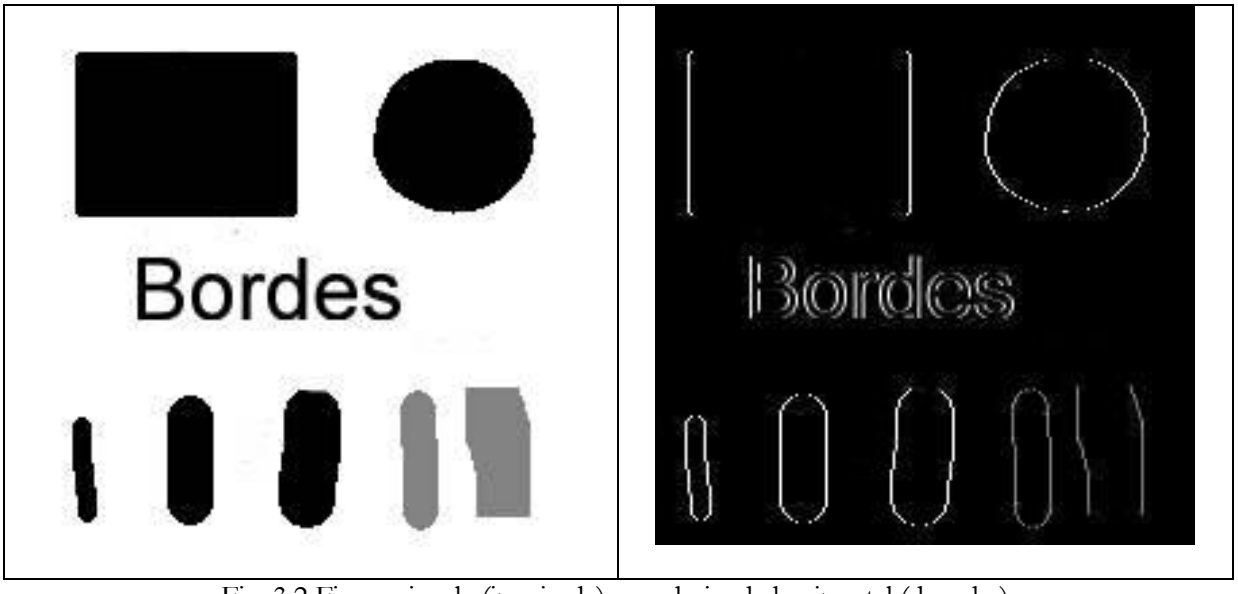


Fig. 3.2 Figura simple (izquierda) y su derivada horizontal (derecha).

Puede notarse que el resultado de calcular la derivada horizontal arroja una figura de fondo negro y se han delineado los bordes de la figura original, es decir la derivada es una operación que extrae los bordes de la figura. Esto se puede entender de la siguiente manera: dado que la derivada haya los cambios, entonces en las zonas uniformes su valor es cero, po lo cual en éstos el fondo es negro y en las transiciones solo queda el perfil de la figura es decir su borde en la dirección horizontal.

Si desarrollamos ahora la misma idea, pero para la dirección vertical, obtendremos la relación

$$\frac{\delta I}{\delta y} = \left| I[x, y+1] - I[x, y] \right| . \qquad (3.5)$$

En la figura 3.3. se muestra la aplicación de ésta fórmula para la figura anterior.

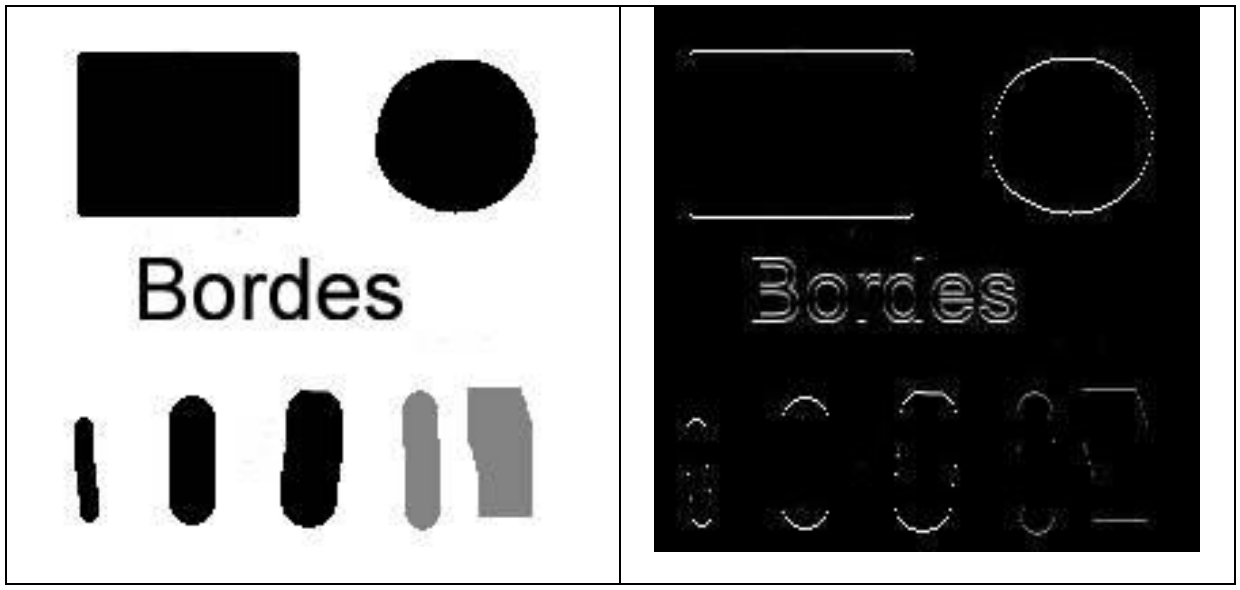


Fig. 3.3 Figura simple (izquierda) y su derivada vertical (derecha).

El efecto es similar a (3.4), pero ahora se han demarcado los bordes verticales. Por lo tanto podemos concluir que en general el operador *derivada* funciona como un detector de bordes y dependiendo de su dirección determina los bordes en esa dirección.

Extendiendo ésta idea podemos crear detectores de bordes en diferentes direcciones o bien detectores para todos los bordes, para esto podemos introducir varios métodos combinando (3.4) y (3.5). Una manera sencilla de hacer la combinación es

$$\delta I = \max\{\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y}\} . \qquad (3.6)$$

Donde **max** indica el mayor de los argumentos. Si aplicamos esta operación obtendremos los bordes o contornos de la imagen como se muestra en la fig. 3.4.

Fig. 3.4. Una imagen (izquierda) y el resultado de aplicar la fórmula 3.6 (derecha).

Puede observarse que ahora se han hallado los bordes en ambas direcciones. La importancia que tiene la determinación de los contornos de una figura dentro de una imagen radica en que muchos algoritmos requieren que se demarquen los objetos que están contenidos en una imagen. Algunas de las aplicaciones más importantes que requieren de ésta operación como paso previo son las siguientes:

- Conteo de objetos
- Localización de objetos
- Medición de las características métricas de los objetos
- Determinación de las características geométricas de los objetos
- Discriminación por tamaño o forma de objetos
- Reconocimiento óptico de caracteres (OCR: Optical Character Recognition).

Es posible formular otros modelos para el cálculo de bordes a partir del concepto de derivada, a continuación se presentan algunas formulaciones típicas.

3.2 Detectores basados en el gradiente.

El gradiente es una operación que determina la dirección de máximo crecimiento de una función, en el caso de dos dimensiones y en su versión continua se define como una operación que se aplica a una función de dos variables **f(x,y)**, su forma matemática es

$$\nabla f(x,y) = \hat{i}\,\frac{\partial f(x,y)}{\partial x} + \hat{j}\,\frac{\partial f(x,y)}{\partial y}. \qquad (3.7)$$

Donde $\hat{i}y\hat{j}$ son los vectores unitarios en las direcciones x e y. Si quisiéramos aplicar esta forma de la derivada para una función bidimensional, como es el caso de una imagen, en particular para una imagen digital nos enfrentamos al problema que los pixeles representan un valor escalar y no vectorial, por lo cual debemos hacer una simplificación de la expresión (3.7), tomando como base las relaciones (3.4) y (3.5) podemos tomar la magnitud de (3.7) en forma digital, por lo tanto usando la norma L2 para la magnitud del vector tendremos que

$$\nabla_{D2} I[x,y] = \sqrt{\left(\frac{\delta I}{\delta x}\right)^2 + \left(\frac{\delta I}{\delta y}\right)^2}\ . \qquad (3.8)$$

Lo cual se puede escribir como

$$\nabla_{D2} I[x,y] = \sqrt{(I[x+1,y]-I[x,y])^2 + (I[x,y+1]-I[x,y])^2}\ . \qquad (3.9)$$

Pero como la diferencia máxima que podemos encontrar en una imagen de **L** bits por pixel en alguno de sus canales es 2^L-1, entonces podemos tener valores fuera de rango hasta por factor de dos, ya que cada término al cuadrado puede tomar ese valor. Para evitar este problema debemos acotar la salida, de donde la forma de (3.9) adaptada a un ambiente digital puede escribirse de la siguiente manera

$$\nabla_{D2} I[x,y] = \max\left\{\sqrt{(I[x+1,y]-I[x,y])^2 + (I[x,y+1]-I[x,y])^2}, 2^L - 1\right\}. \qquad (3.10)$$

Y otro mecanismo puede ser el introducir un factor ½ fuera del radical, de donde otra versión es

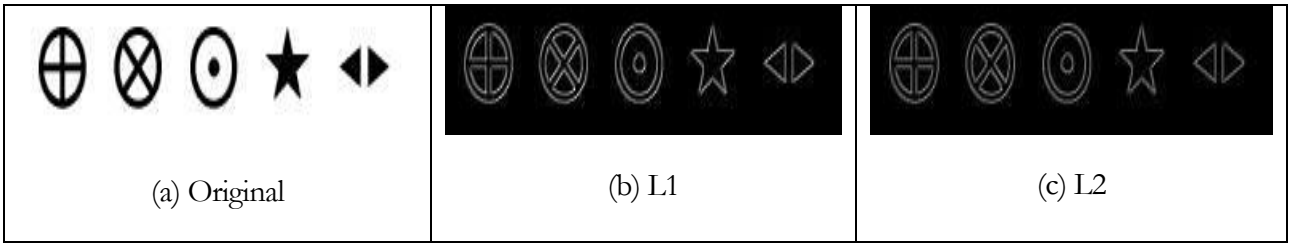
$$\nabla_{D2} I[x,y] = \frac{1}{2}\sqrt{(I[x+1,y]-I[x,y])^2 + (I[x,y+1]-I[x,y])^2}\ . \qquad (3.11)$$

Un método menos ávido de operaciones numéricas de punto flotante es utilizar la norma L1 o "de cuadras", en ésta se toman las diferencias completas en las direcciones x e y como si se caminase en una ciudad con manzanas cuadradas e iguales y la regla es que solo se puede caminar obre las calles sin ingresar a las manzanas en diagonal. Usando ésta norma tendremos que el gradiente digital en norma L1 tomará alguna de las siguientes formas:

$$\nabla_{D1} I[x,y] = \max\left\{|I[x+1,y]-I[x,y]| + |I[x,y+1]-I[x,y]|, 2^L - 1\right\} \qquad (3.12)$$

$$\nabla_{D1} I[x,y] = \frac{1}{2}\left\{|I[x+1,y]-I[x,y]| + |I[x,y+1]-I[x,y]|\right\} \qquad (3.13)$$

En la figura 3.5. se muestra el efecto de (3.11) y (3.13) sobre una imagen.



(a) Original (b) L1 (c) L2

Fig. 3.5. Gradiente digital de una imagen

Podemos notar que la evaluación de (3.11) y (3.13) solo involucran al pixel de referencia, al que está a su derecha y al que está debajo de él. En la figura 3.6a se muestran los pixeles involucrados. Algunos autores introducen un **Gradiente Aproximado** haciendo participar a otro pixel del entorno del pixel de referencia como muestra la figura 3.6b. Las flechas indican las diferencias consideradas. La fórmula resultante es

$$\nabla_{D-Aprox} I[x,y] = F\left\{|I[x+1,y+1]-I[x,y]| + |I[x,y+1]-I[x+1,y]|\right\} \qquad (3.14)$$

Fig. 3.6. Gradiente – puntos relacionados

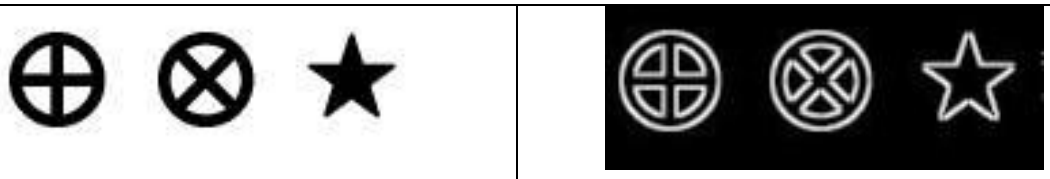


Fig. 3.7 Imagen y su gradiente aproximado

Donde la función **F** se puede escoger como se hizo en (3.12) y (3.13).

Comparación de los modelos de gradiente

La siguiente secuencia de imágenes muestra el resultado de aplicar cada uno de los modelos de gradiente y se presentan algunos comentarios.

Imagen	Comentario
	Original
	**L2**. Puede notarse que el resultado es una imagen oscura y los bordes tienen una intensidad uniforme.
	**L1**. Los bordes tienen una intensidad variable y son más claros.
	**Aproximado**. La intensidad es alta, pero los bordes son muy gruesos y llegan a interceptarse.

Fig. 3.8. Imagen y gradientes digitales.

Debido a que estas operaciones utilizan la primera derivada digital para determinar los bordes, se les conoce como detectores de primer orden.

3.2. Laplaciano y Convolución.

A las operaciones diferenciales digitales basadas en la segunda derivada les llamaremos operaciones de segundo orden. Si evaluamos la segunda derivada en la dirección de **x,** partiendo de (3.4) tendremos que:

$$\begin{aligned}\frac{\delta^2 I}{\delta^2 x} &= \frac{\delta}{\delta x}\left(\frac{\delta I}{\delta x}\right)\\ &= \frac{\delta}{\delta x}(I[x+1,y]-I[x,y])\\ &= (I[x+2,y]-I[x+1,y])-(I[x+1,y]-I[x,y])\end{aligned}$$

Simplificando la última relación, tendremos que

$$\frac{\delta^2 I}{\delta^2 x} = I[x+2,y]-2I[x+1,y]+I[x,y], \qquad (3.15)$$

si centramos las diferencias respecto al pixel ubicado en la coordenada (x,y) haciendo **x** □ **x-1** tendremos que

$$\frac{\delta^2 I}{\delta^2 x} = I[x+1,y]-2I[x,y]+I[x-1,y]. \qquad (3.16)$$

A esta relación se le llama *segunda diferencia central*. Podemos rescribirla usando la siguiente interpretación: "buscar la diferencia entre un pixel y sus vecinos laterales". De donde (3.16) quedará

$$\frac{\delta^2 I}{\delta^2 x} = (I[x,y]-I[x+1,y])+(I[x,y]-I[x-1,y]),$$

que se puede expresar como

$$\frac{\delta^2 I}{\delta^2 x} = 2I[x,y]-I[x+1,y]-I[x-1,y],$$

o bien

$$\frac{\delta^2 I}{\delta^2 x} = -I[x+1,y]+2I[x,y]-I[x-1,y]. \qquad (3.17)$$

Esta relación es simétrica y difiere de (3.16) solo por un cambio de signo en cada uno de los términos de la parte derecha. Es decir la segunda diferencia expresa el cambio entre un pixel y sus vecinos. Desarrollando la expresión equivalente en la dirección **y** obtendremos que

$$\frac{\delta^2 I}{\delta^2 x} = -I[x,y+1]+2I[x,y]-I[x,y-1], \qquad (3.18)$$

Si ahora recordamos la expresión para el Laplaciano (∇^2) de una función de dos variables, tenemos que

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial^2 x}+\frac{\partial^2 f}{\partial^2 y}, \qquad (3.19)$$

Que en su versión digital corresponde a la segunda diferencia combinada en **x** e **y** por tanto

$$\begin{aligned}\nabla^2 I &= \frac{\delta^2 I}{\delta^2 x}+\frac{\delta^2 I}{\delta^2 y}\\ &= -I[x-1,y]+2I[x,y]-I[x+1,y]-I[x,y-1]+2I[x,y]-I[x-1,y]\\ &= -I[x-1,y]-I[x,y-1]+4I[x,y]-I[x+1,y]-I[x,y+1]\end{aligned}$$

si acomodamos los elementos que definen ∇^2I en una cuadrícula según su posición relativa al pixel en (x,y), tendremos que

∇^2I =

	-I[x , y-1]	
-I[x-1 , y]	4 I[x , y]	-I[x+1, y]
	-I[x, y+1]	

(3.20)

En este modelo geométrico podemos ver como el Laplaciano cuantifica la diferencia entre el tono del pixel ubicado en la posición (x, y) y sus vecinos horizontales-verticales laterales. Si formamos una matriz con los pixeles vecinos del pixel centrado en (x, y), es decir un ventana de la imagen **I** de 3x3 alrededor del pixel citado y por otro lado extraemos los coeficientes del arreglo bidimensional en la expresión (3.20), tendremos que

$$\nabla^2 I[x,y]=\begin{pmatrix}0 & -1 & 0\\ -1 & 4 & -1\\ 0 & -1 & 0\end{pmatrix}\otimes\begin{pmatrix}I[x-1,y-1] & I[x,y-1] & I[x+1,y-1]\\ I[x-1,y] & I[x,y] & I[x+1,y]\\ I[x-1,y+1] & I[x,y+1] & I[x+1,y+1]\end{pmatrix}=\tilde{\mathrm{M}}\otimes\tilde{\mathrm{I}}_3[x,y]$$

Definición. *El símbolo ⊗ es una operación entre matrices denominado* ***convolución,*** *éste produce un número real de tal forma que los elementos de las matrices se combinan uno a uno multiplicándose y los productos se suman, analíticamente:*

Sean A *y* B *dos matrices cuadradas en* $\Re^n\times\Re^n$ *con elementos de matriz* (a_{ij}) *y* (b_{ij}) *respectivamente, entonces*

$$A\otimes B=\sum_{i=1}^{n}\sum_{j=1}^{n}a_{ij}b_{ij}=z, \quad | \quad z\in\Re\,. \qquad (3.21)$$

Este producto se puede interpretar como un producto escalar entre matrices cuadradas.

Usando ésta terminología podemos decir que el Laplaciano de un pixel de una imagen digital bidimensional es la convolución entre la matriz $\tilde{M}$ dada por

$$\tilde{M}_{+} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad (3.22)$$

y la ventana de 3×3 centrada en el pixel de referencia $\tilde{I}_3[x, y]$. Este genera un número que se asignará al pixel (x, y) de la nueva imagen $\tilde{I}'$ **[**x,y**]**. Dado que el tono debe estar en el intervalo [0, 2^L-1] donde L es la profundidad en bits de la representación de la imagen en cada canal, es claro que no puede haber tonos negativos y como la convolución de $\mathbf{M}_+$ con la ventana podría producir tales eventos puede usarse alguna de las siguientes reglas o una combinación de ellas para evitar el suceso:

1. Aplicar la función valor absoluto al resultado. *Eliminación directa de valores negativos.*

2. si $z < 0 \Rightarrow z = 0$, si $z > 2^L\text{-}1 \Rightarrow z = 2^L\text{-}1$. *Ajuste al intervalo válido de salida.*

Se ha introducido un índice (+) por la forma en que están ubicados los términos (-1) en la matriz.

La aplicación del Laplaciano debe considerar que el los bordes de la imagen (superior, inferior, derecho e izquierdo) no será posible calcularlo, dado que estos carecen de alguno de los vecinos requeridos para la evaluación, por lo tanto si la imagen tiene dimensiones horizontal- vertical **n** □ **m**, entonces el ciclo de aplicación se debe aumentar o reducir un pixel en cada borde dependiendo del caso, por lo tanto el algoritmo puede quedar de la siguiente manera.

```
for (i = 1 .. n-2 )
  for (j = 1 .. m-2){
     s ← 0
     for (α = -1 .. 1)
       for (β = -1 .. 1)
         s ← s + M[α, β]*I[x+ α, y+ β]
     I'[x,y] ←  s
  }
```

Fig. 3.9. Algoritmo para aplicar una convolución de 3x3.

Donde se han modelado los índices de la matriz de convolución entre -1 y 1 con el objetivo de acceder de manera simple a los elementos de matriz de la ventana de la imagen correspondientes.

3.3. Otros detectores de Bordes de segundo orden

La matriz $\tilde{M}$ de 3x3 que representa la operación asociada la Laplaciano se puede interpretar al ser aplicada a una ventana de la imagen $\tilde{I}[x, y]$como la diferencia entre el pixel central ubicado en [x , y] en la imagen y sus vecinos en el este (E), oeste (O), norte (N) y sur (S) (geográficamente) como indica la relación (3.20). Podríamos introducir otra matriz de convolución similar, pero que tomase las diferencias entre el pixel central y los pixeles vecinos ubicados en las posiciones geográficas NE, NO, SE y SO, ésta matriz de convolución es un Laplaciano rotado un ángulo de 45º (es decir $\pi/4$). Su forma es la siguiente:

$$\tilde{M}_{\times} = \begin{bmatrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{bmatrix} \qquad (3.23)$$

El símbolo × como subíndice representa la ubicación de los términos -1 en la matriz. Este es un buen detector de bordes diagonales, es la siguiente figura se muestra el efecto de las transformaciones regionales $\tilde{M}_{+}$ y $\tilde{M}_{\times}$ sobre una imagen de contornos simples.

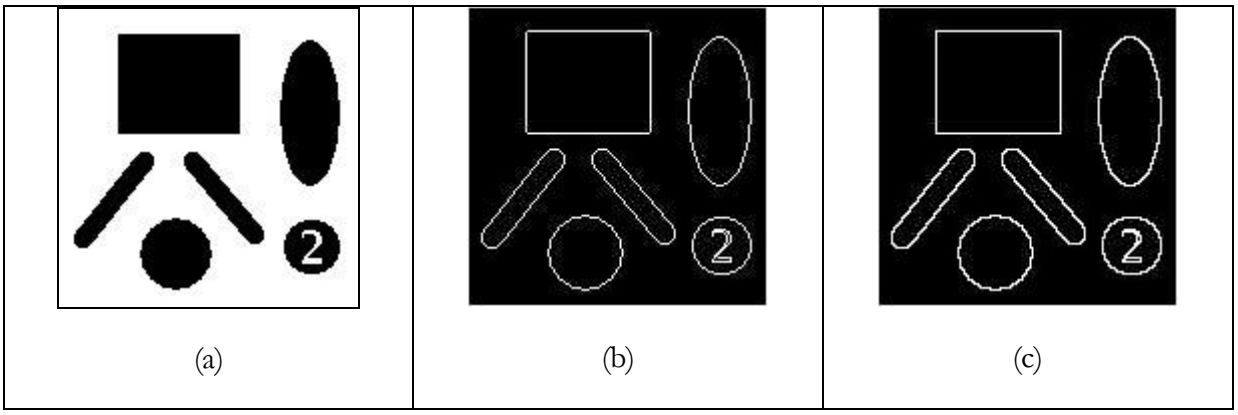


Fig. 3.10. Figura Simple (a) y sus Laplacianos en + (b) y × (c).

Puede notarse que los bordes de la Fig. 3.10c son más gruesos que los de 3.10b.

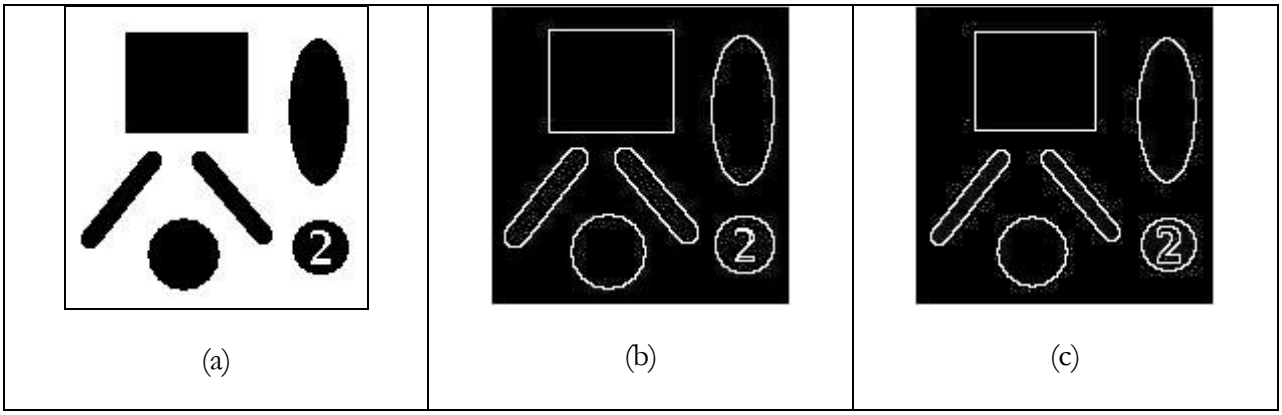
Es posible definir otras formas para el Laplaciano, una inmediata es aquella que considera las diferencias del pixel central a sus ocho primeros vecinos, la matriz de convolución para éste tiene la forma

$$\tilde{M}_{8V+} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad (3.24)$$

Puede notarse que el elemento central ahora es ocho debido a que tenemos ocho diferencias respecto al pixel central. Una variante posible consiste en invertir los signos de los coeficientes de (3.24), resultando la matriz

$$\tilde{M}_{8V-} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad (3.25)$$

En la figura 3.11 se muestra el efecto de (3.24 y (3.25) sobre la imagen anterior.



(a) (b) (c)

Fig. 3.11 Aplicación de los Laplacianos 8V+ (b) y 8V- (c) sobre la imagen (a).

Puede notarse como 8V+ y 8V- detectan los bordes de diferente manera, en particular note en efecto en las líneas inclinadas y el número dos enmarcado.

Existe una gran familia de detectores de bordes modelados mediante matrices de convolución de 3x3, a continuación se presenta una relación de los más importantes.

3.3.1. Operadores de Sobel

El modelo estima las componentes del gradiente mediante las siguientes relaciones:

$$\tilde{G}_x = \frac{\partial}{\partial x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{3.26}$$

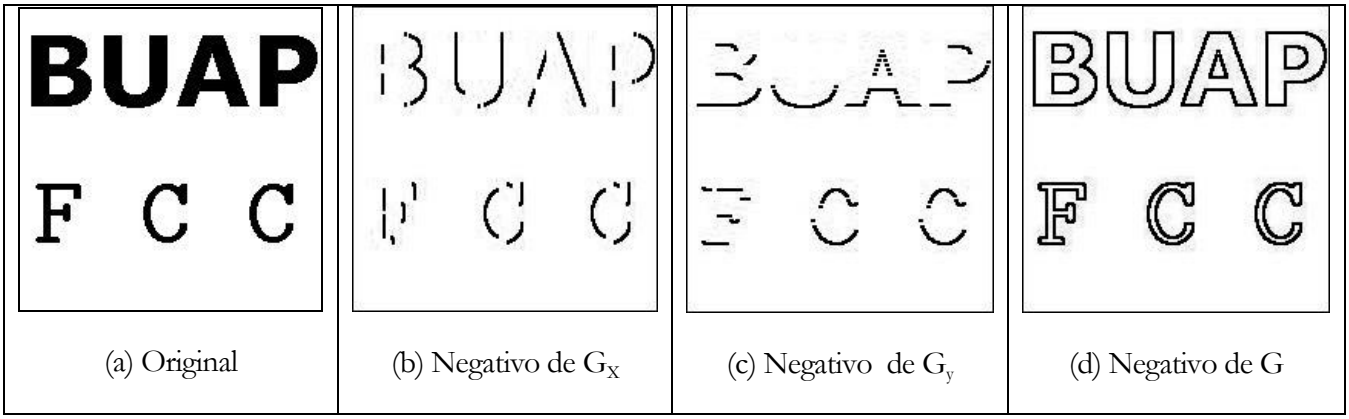
$$\tilde{G}_y = \frac{\partial}{\partial y} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{3.27}$$

Estos permiten hallar las componentes del gradiente y a partir de ellas determinar su magnitud y dirección, éstas se pueden calcular mediante las relaciones típicas

$$G = \frac{\partial^2}{\partial x \partial y} = \left\| \hat{i} G_x + \hat{j} G_y \right\| = \sqrt{G_x^2 + G_y^2} \tag{3.28}$$

$$\tan(\theta) = \frac{G_y}{G_x} \tag{3.29}$$

En la Fig. 3.12 se muestra el efecto de G_x , G_y y G sobre una imagen.



(a) Original (b) Negativo de G_X (c) Negativo de G_y (d) Negativo de G

Fig. 3.12. Aplicación de los operadores de Sobel.

Puede notarse como G_x y G_y se complementan para formar G, detectando cada uno los bordes verticales (G_x) y horizontales (G_y) respectivamente.

3.3.2. Otros operadores de bordes de 3×3

a) Operadores de Prewitt

$$\tilde{P}_x = \frac{\partial}{\partial x} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad \tilde{P}_y = \frac{\partial}{\partial y} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

b) Operadores Simples de diferencia de primer orden

$$\tilde{D}_x = \frac{\partial}{\partial x} = \begin{bmatrix} 0 & 0 & 0 \\ -1/2 & 0 & 1/2 \\ 0 & 0 & 0 \end{bmatrix} \qquad \tilde{D}_y = \frac{\partial}{\partial y} = \begin{bmatrix} 0 & -1/2 & 0 \\ 0 & 0 & 0 \\ 0 & 1/2 & 0 \end{bmatrix}$$

c) Operadores Diferenciales Combinados de segundo orden

$$\tilde{C}_x = \frac{\partial}{\partial x} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad \tilde{C}_y = \frac{\partial}{\partial y} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad \tilde{C}_{xy} = \frac{\partial^2}{\partial x \partial y} = \begin{bmatrix} -1/4 & 0 & 1/4 \\ 0 & 0 & 0 \\ 1/4 & 0 & -1/4 \end{bmatrix}$$

3.4. Filtros de suavizado – Medias

Uno de los problemas que se atacan con filtros de convolución de 3x3 es del suavizar imágenes. Este proceso se relaciona con la reducción de las transiciones fuertes. Un efecto que tiene éste grupo de filtros es la reducción de ruido, a costa de la pérdida de algunos detalles.

3.4.1. Media aritmética

El modelo más simple corresponde a la media aritmética., este considera la media de los pixeles en un entorno de 3x3 centrado en un pixel *(x , y)*. En el siguiente diagrama se muestra que pixeles se consideran:

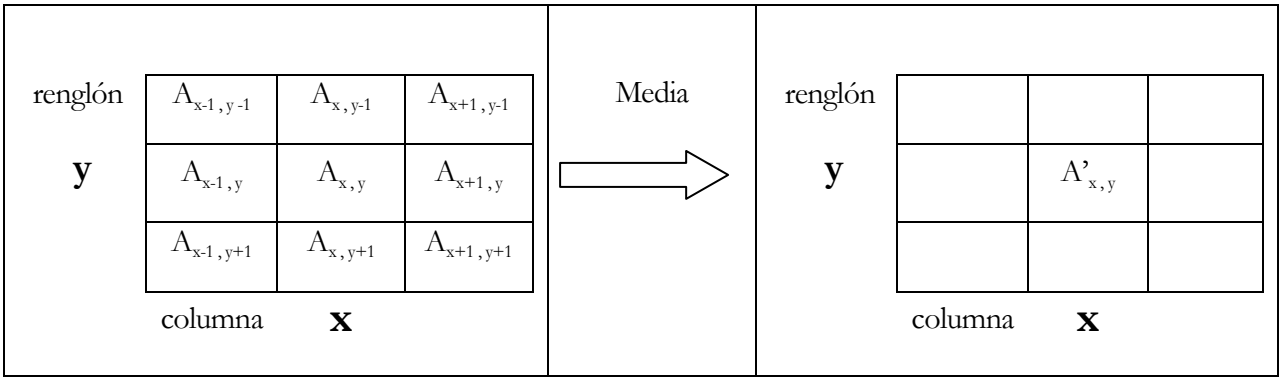


Fig. 3.13. Proceso de la media aritmética

Para determinar el valor del tono $A'_{x,y}$ en el canal **k** se utiliza simplemente la media aritmética de los valores de los pixeles del canal **k** de la imagen original considerando los primeros vecinos del pixel *(x , y)*. Analíticamente tendremos que:

$$I'[x,y]=\frac{1}{9}\sum_{i=-1}^{1}\sum_{j=-1}^{1}I[x+i,y+j] \qquad (3.30)$$

Esto se puede rescribir posicionalmente de la siguiente manera:

$$I'[x,y]=\begin{pmatrix}\frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\end{pmatrix}\otimes\begin{pmatrix}I[x-1,y-1] & I[x,y-1] & I[x+1,y-1]\\ I[x-1,y] & I[x,y] & I[x+1,y]\\ I[x-1,y+1] & I[x,y+1] & I[x+1,y+1]\end{pmatrix}$$

De donde la matriz de convolución que modela a la media aritmética resulta:

$$\tilde{M}_{MA}=\begin{bmatrix}\frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\end{bmatrix} \qquad (3.31)$$

Lógicamente este filtro se puede aplicar en el cuerpo de la imagen, excepto en las orillas, donde se deben usar estructuras reducidas debido a la carencia de todos los vecinos. Se puede entender que el efecto de la *media aritmética* es de suavizamiento (soften), ya que los pixeles vecinos se parecerán debido a la mezcla que se produce entre ellos. Una observación que se puede hacer es que la imagen se hará borrosa y las transiciones fuertes (bordes) se disolverán parcialmente. En la figura siguiente se muestra una imagen y su media, note por ejemplo el efecto en los cabellos de la frente de la chica como se disuelven, así como el resto de detalles.

Fig. 3.14. Imagen y su media aritmética (aplicada 4 veces).

3.4.2. Media ponderada

Es posible modelar otros procesos para la media, como por ejemplo la media ponderada. Ésta considera modificar el peso que tiene el pixel central ubicado en *(x, y)*, en el modelo anterior todos los pesos son iguales a $\frac{1}{9}$, si modificamos el peso del pixel central a un valor arbitrario positivo, digamos **p**, entonces dado que debemos preservar el rango de salida de la matriz de convolución, debemos modificar el peso de los pixeles vecinos. Una manera de garantizar que la salida no sobrepase el rango permitido debido a la representación de cada canal ($[0 \ldots 2^L\text{-}1]$, donde L es la profundidad en bits). Podemos notar que la matriz de convolución de la media aritmética se puede escribir como:

$$\tilde{M}_{MA} = \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

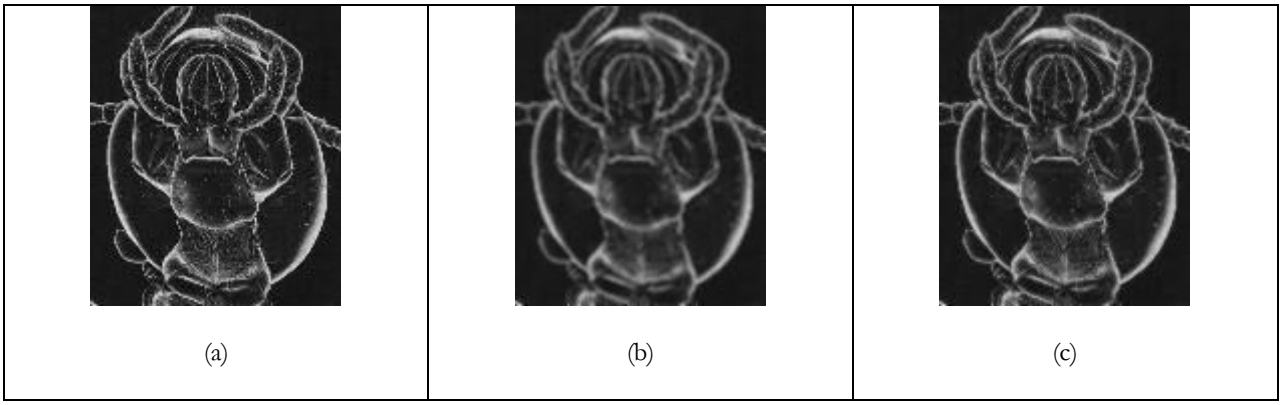
Es decir la suma de los coeficientes de la matriz es la unidad, esto garantiza que no salgamos de rango. Por lo cual si modificamos el peso del pixel central (de referencia), entonces podemos escribir la matriz de convolución para la media ponderada como:

$$\tilde{M}_{MP}(p) = \frac{1}{8+p}\begin{bmatrix} 1 & 1 & 1 \\ 1 & p & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad (3.32)$$

Que se reduce al caso de la media aritmética cuando $p = 1$. En general para éste filtro, vamos a considerar que $p \geq 0$. Podemos afirmar que la media ponderada es una función del peso p. El caso cuando $p = 0$, equivale a sustituir al pixel en (x, y) por el promedio de sus vecinos. Cuando p es grande comparado con la unidad la influencia de los vecinos se ve reducida y cuando $p = 1$ todos los vecinos contribuyen igual que el pixel central al valor de salida.

Algunos autores [5] llaman a éste filtro: Filtro de Paso Bajo (Low Pass Filter). Debido a que incrementa las transiciones suaves y reduce las fuertes, estos etiquetan algunas matrices de convolución con valores de p específicos, por ejemplo cuando $p = 2$ se denomina al filtro: LP1 (Low Pass 1), $p = 4$: LP2 y $p = 12$: LP3.

En la siguiente figura se muestra el efecto de la media ponderada para dos valores de p.

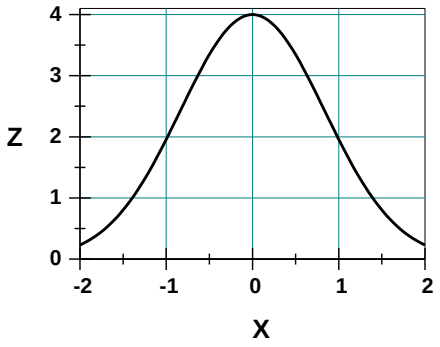


(a) (b) (c)

Fig. 3.15. (a) Imagen de microscopia. (b) LP1. (c) LP3.

3.4.3. Media Gaussiana.

Es muy popular el filtro de la media gaussiana, este se basa en considerar una curva gaussiana de revolución alrededor del pixel central de una matriz de convolución. La curva en dos dimensiones de la función de Gauss es:



(x-1, y-1)	(x, y-1)	(x+1, y-1)
(x-1, y)	(x, y)	(x+1, y)
(x-1, y+1)	(x, y+1)	(x+1, y+1)

Fig.3.16. Curva Gaussiana y Pixeles Vecinos al (x, y).

La función que se propone para modelar la función es la siguiente:

$$z = 4e^{-x^2/\sqrt{2}} \qquad (3.33)$$

De esta manera en x = 0, z = 4. Esto se hace para tener un valor entero cuando x = ±1, z = 2, lo cual corresponde a los pixeles vecinos del (x, y). Y para los pixeles de las esquinas la distancia del central es de $\sqrt{2}$ donde z ≈ 1. Se usan los mismos argumentos para la dirección vertical. Con estas observaciones, la matriz de convolución quedará:

$$\tilde{M}_{Gauss} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \qquad (3.34)$$

En la figura siguiente se muestra el efecto del filtro de la media gaussiana.

(a)

(b)

Fig.3 .17.

(a) Imagen Original

(b) Aplicación del Filtro Gaussiano

Puede notarse la reducción en la luz de la imagen y que al igual que la media aritmética la imagen se hace menos nítida.

A éste filtro también se le llama *Gaussiano Fuerte* y existe una variante llamada *Gaussiano Débil*, el cual utiliza el valor de 0.9 para la desviación estándar de la curva de Gauss. Para éste caso se utiliza la función

$$z = 12e^{-x^2/0.9} \qquad (3.35)$$

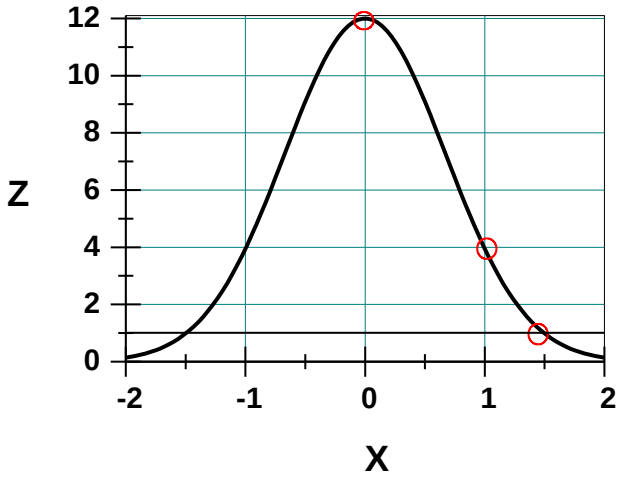


Fig. 3.18. Curva Gaussiana con media cero y desviación estándar 0.9.

Puede notarse en la figura que para x = 1, tenemos que z = 4 y que para x = $\sqrt{2}$, tenemos que z ≈ 1. La matriz de convolución para este filtro de media tiene la forma:

$$\tilde{M}_{Gauss-Débil} = \frac{1}{32}\begin{bmatrix} 1 & 4 & 1 \\ 4 & 12 & 4 \\ 1 & 4 & 1 \end{bmatrix} \qquad (3.36)$$

Puede notarse que el peso del pixel central es bastante alto comparado con sus vecinos. En general se pueden modelar otros filtros de ésta clase eligiendo valores para **A** y **S** en la expresión:

$$Z(x) = Ae^{-x^2/S} \qquad (3.37)$$

Para formar la matriz de convulsión, se deben tomar los valores de z para cuando x = 0, 1, $\sqrt{2}$. El primero valor se asigna al centro de la matriz, el segundo a los valores (N, S, E, O) de la matriz y el tercero para las esquinas de ella (NE, NO, SE, SO). Y finalmente el peso exterior se calcula como

$$P = 1/\left(Z(0) + 4Z(1) + 4Z(\sqrt{2})\right) \qquad (3.38)$$

3.4.4. Media Geométrica.

Una manera no lineal de calcular un promedio consiste en utilizar una norma cuadrática [6]. Se define la media geométrica de un conjunto de datos

$$X = \{x_1, x_2, \ldots, x_n\}, \quad < X >_G = \sqrt[n]{x_1 x_2 \cdots x_n} \qquad (3.39)$$

Este tipo de filtro se puede aplicar a un entorno de 3x3 o (2k+1)x(2k+1) con k un natural. Para el caso de 3x3 tendremos que el filtro se podrá evaluar mediante la expresión

$$I'[x,y] = \sqrt[9]{\prod_{i=-1}^{1}\prod_{j=-1}^{1} I[x+i, y+j]} \qquad (3.40)$$

Donde el símbolo Π indica el producto extendido, es decir:

$$\prod_{k=a}^{b} f(n) = f(a) f(a+1) f(a+2) \cdots f(b-2) f(b-1) f(b)$$

Es claro que la evaluación de la media geométrica es más pesada que el de la media aritmética por el tipo de operaciones que realiza.

3.4.5. Promedio Direccional.

Este considera las cuatro direcciones indicadas en la figura siguiente, se realiza el promedio de tres datos en cada una y se asigna a la salida el mayor de ellos [6].

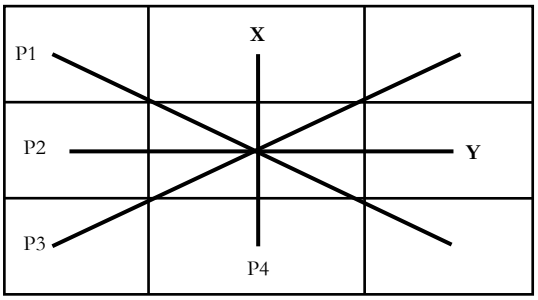


Fig. 3.19. Direcciones base para la media direccional.

Los promedios que se toman alrededor del pixel (x, y) son los siguientes:

$$\begin{aligned} p_1 &= \left(I[x-1,y-1]+I[x,y]+[x+1,y+1]\right)/3 \\ p_2 &= \left(I[x-1,y]+I[x,y]+[x+1,y]\right)/3 \\ p_3 &= \left(I[x-1,y+1]+I[x,y]+[x+1,y-1]\right)/3 \\ p_4 &= \left(I[x,y-1]+I[x,y]+[x,y+1]\right)/3 \\ I'[x,y] &= \max(p_1, p_2, p_3, p_4) \end{aligned} \tag{3.41}$$

3.4.6. Media Armónica y Contra Armónica.

Estos son procedimientos están basados en evaluar la media de los recíprocos en un entorno de 3x3. Para la *Media Armónica* [6] en caso de haber un tono negro en la región la salida se hace cero (negro).

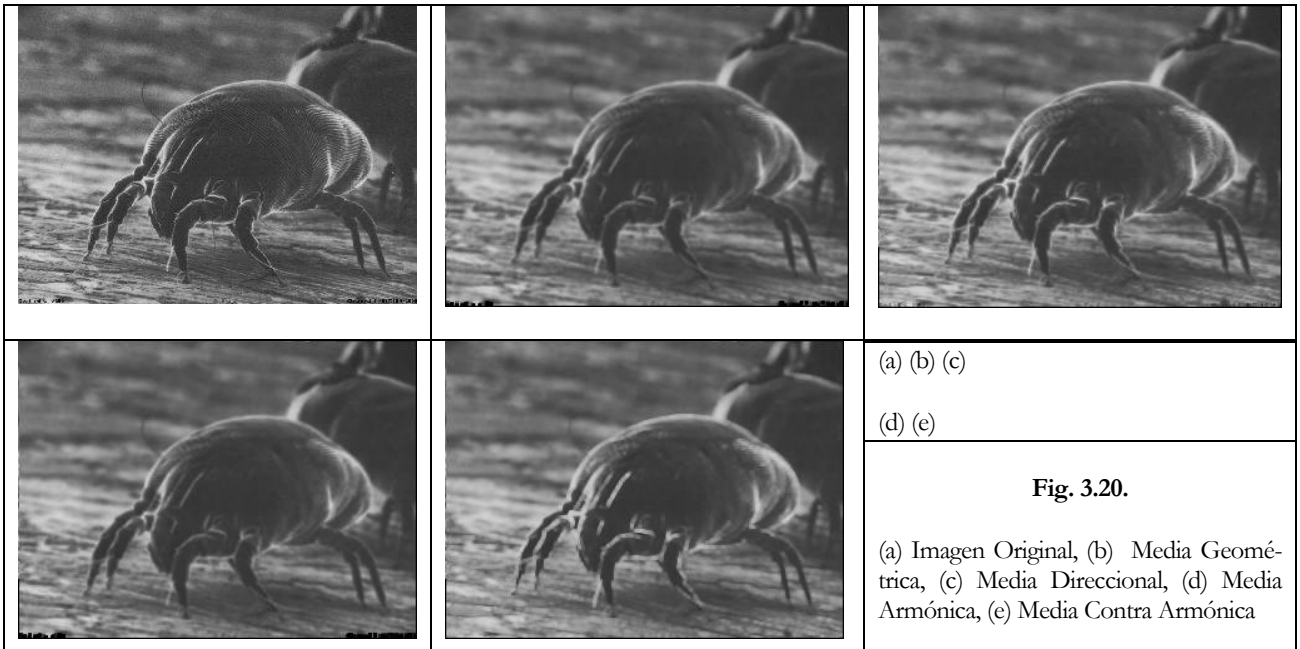
La manera de hacer la evaluación es la siguiente:

$$\begin{aligned} Z_k &= \begin{Bmatrix} I[x-1,y-1], I[x,y-1], I[x+1,y-1], \\ I[x-1,y], I[x,y], I[x+1,y], \\ I[x-1,y+1], I[x,y+1], I[x+1,y+1] \end{Bmatrix}, \\ I'[x,y] &= \begin{cases} 0 \;\; si \;\; a\lg\acute{u}n \;\; Z_k = 0 \\ \dfrac{9}{S}, \;\; con \;\; S = \displaystyle\sum_{k=1}^{9} \frac{1}{Z_k} \end{cases} \end{aligned} \tag{3.42}$$

Para la *Media Contra Armónica* [6], el procedimiento es:

$$I'[x,y] = \begin{cases} 0 \;\; si \;\; todos \;\; los \;\; Z_k = 0 \\ Z_c \;\; si \;\; k = c \;\; es \;\; el \;\; único \;\; no \;\; nulo \\ \dfrac{S_4}{S_3}, \;\; con \;\; S_4 = \displaystyle\sum_{k=1}^{9} Z_k^4, S_3 = \sum_{k=1}^{9} Z_k^3 \end{cases} \tag{3.43}$$

En la figura siguiente se muestra el efecto de los cuatro últimos filtros comentados.



(a) (b) (c)

(d) (e)

Fig. 3.20.

(a) Imagen Original, (b) Media Geométrica, (c) Media Direccional, (d) Media Armónica, (e) Media Contra Armónica

3.4.7. Mediana

Un método que se usa de manera frecuente para eliminar el ruido debido a fallas en los sistemas de registro de imágenes digitales, tal como la "sal" y "pimienta", es el filtro de la mediana. Entenderemos como "sal" puntos blancos (saturados) en una zona obscura y que han producido por una falla del detector y "pimienta" aquellos puntos negros (no registrados) en una zona clara. La figura siguiente muestra imágenes con este tipo de ruido.

(a) (b)

Fig. 3.21. Imágenes con "sal" y "pimienta".

El algoritmo consiste en tomar un entorno alrededor de un pixel *(x, y)*, por ejemplo los 9 elementos en una región de 3x3, ordenar los elementos y elegir el central como valor de salida.

Tomando el conjunto Z_{3x3} del ejemplo anterior (media armónica y contra armónica), el conjunto se organiza en forma de vector, de tal manera que:

$Z_{3x3} = (Z_1, Z_2, Z_3, Z_4, Z_5, Z_6, Z_7, Z_8, Z_9)$,

Luego del ordenamiento el vector quedará:

$Z_{3x3} = (Z'_1, Z'_2, Z'_3, Z'_4, Z'_5, Z'_6, Z'_7, Z'_8, Z'_9)$, donde por ejemplo $Z'_k \leq Z'_{k+1}$; orden creciente.

Y entonces la mediana será:

$$med = med\{Z_{3x3}\} = Z_{3x3}[5].$$

Y finalmente el pixel central se sustituirá por la median, es decir:

$$I'[x, y] = med. \qquad (3.44)$$

En la figura siguiente se muestra el efecto de aplicar éste filtro a las imágenes de la figura 3.21.

(a) (b)

Fig. 3.22. Imágenes con "sal" y "pimienta", luego de aplicar el filtro de la mediana.

Puede notarse la eliminación del ruido del tipo "sal" y "pimienta" de las imágenes luego de aplicarse el filtro de la mediana.

El efecto puede explicarse de la siguiente manera: si en una región hay un parásito blanco y/o negro luego de ordenar el conjunto, estos se colocarán en las orillas del vector ordenado:

$Z_{3x3} = (\mathbf{Z'_1}, Z'_2, Z'_3, Z'_4, Z'_5, Z'_6, Z'_7, Z'_8, \mathbf{Z'_9})$,

Donde Z'_1 será un negro natural o un parásito y Z'_9 será un blanco natural o un parásito. Al tomarse como salida el elemento Z'_5, los parásitos se eliminan. Si los parásitos caen en una zona negra (o bien blanca) no afectan la salida.

3.4.8. Media recortada o alpha – trim (K, m).

Este procedimiento es una combinación de la media aritmética y un ordenamiento [6]. La idea consiste en ordenar un entorno por ejemplo de 3x3 y luego cortar las orillas (un elemento) y a continuación realizar el promedio. A este filtro se le llama alpha-trim (3,1), donde el primer índice indica el tamaño del entorno y el segundo el número de pixeles recortados en cada extremo.

La formula para el filtro es la siguiente:

Dado un entorno de KxK pixeles

$Z_{KxK} = (Z_1, Z_2, \ldots, Z_{KxK-1}, Z_{KxK})$,

Se construye el vector ordenado

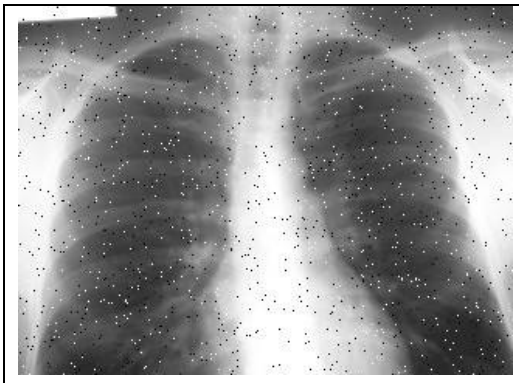
$Z_{KxK} = (Z'_1, Z'_2, \ldots, Z'_{KxK-1}, Z'_{KxK})$,

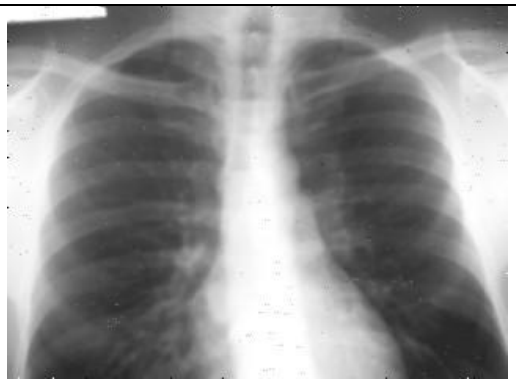
Y se evalúa el promedio eliminando **m** elementos en cada extremo, de tal forma que:

$$I'[x, y] = \alpha t\,(K, m)[x, y] = \sum_{i=m}^{K^2-m} Z'_i \qquad (3.45)$$

Para ordenar se puede usar cualquier método que sea eficiente para conjuntos pequeños de datos.

En la figura siguiente se muestra la aplicación de dicho filtro sobre una imagen ruidosa.





(a) (b)

Fig.3.23.

Imagen ruidosa,

Aplicación del filtro αt(3,2)

Puede notarse que no se eliminan completamente los puntos de ruido, solo se atenúan y aparecen como puntos grises en vez de puntos de sal y pimienta.

Existen otros métodos para la eliminación de ruido tipo sal y pimienta como es el filtrado en el espacio de Fourier o esquemas similares, los procedimientos antes expuestos son ampliamente utilizados dada su simplicidad.

3.5. Repujado – (relieves y luces)

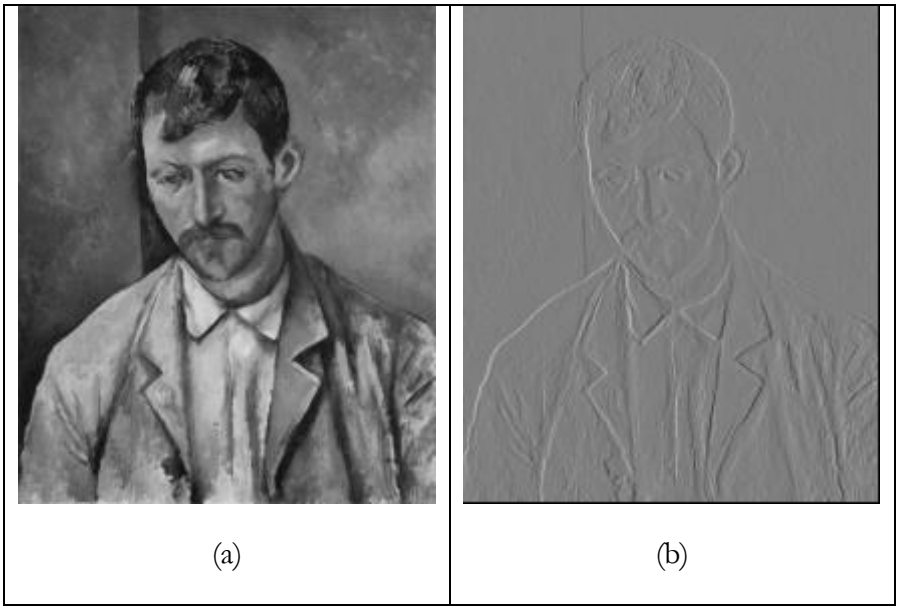
En cartografía, artes gráficas y algunas aplicaciones científicas se requiere visualizar información sobre los cambios en la imagen, estos se relacionan como es natural con derivadas direccionales. Una manera de observar los cambios es modificar la referencia de la derivada, moviéndola de cero a un punto diferente, es común utilizar para esto el punto medio del rango de la gama de tonos.

3.5.1. Repujados simples

Podemos introducir como primer caso el de las derivadas direccionales horizontal y vertical, definiendo la referencia en **Λ/2** en la ecuación (3.4) y eliminando el valor absoluto, luego de una normalización de rango tendremos que:

$$\frac{\delta I}{\delta x} = \frac{\Lambda}{2} + \frac{I[x+1, y] - I[x, y]}{2} \quad , \qquad (3.46)$$

podemos notar que ahora el segundo término puede manejar signo, lo cual ocasiona que en las transiciones donde éste sea positivo se ubicarán a la derecha de **Λ/2** y los de signo negativo a la izquierda de este valor. Es decir los valores donde el cambio sea cero se presentarán como un fondo gris ubicado en la parte central del rango, las zonas con cambio positivo serán más claros y donde la derivada sea negativa más obscuros, en las artes gráficas a éste proceso se le denomina *Repujado*. En la figura. 3.24 se muestra un ejemplo de la situación para el *repujado horizntal*.



(a) (b)

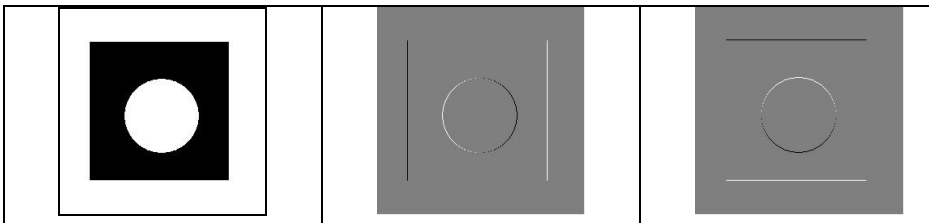
Fig. 3.24. Figura en grises (a) y su repujado X (b)

Puede notarse como la imagen resultante se presenta como una figura en relieve (figura 3.24-b), si observamos por ejemplo el pelo del personaje podemos notar que del lado izquierdo al pasar del fondo (la pared) y el cabello se va de un tono claro a uno obscuro y la derivada resulta negativa, obteniéndose un tono un poco mas obscuro del fondo y del lado derecho el proceso es inverso, el tono obtenido es mas claro.

Para la dirección vertical se puede definir una derivada de manera similar, obteniéndose que:

$$\frac{\delta I}{\delta y} = \frac{\Lambda}{2} + \frac{I[x, y+1] - I[x, y]}{2}, \qquad (3.47)$$

Este proceso remarcará las transiciones en la dirección vertical, al proceso se le denomina *repujado vertical* en la figura 3.25 se presenta una imagen simple para percibir las diferencias entre el repujado horizontal y el vertical.



(a) (b) (c)

Fig. 3.25.

(a) Figura Original

(b) Repujado Horizontal

(c) Repujado Vertical

Puede notarse en la figura 3.25-(b) como se marcan los bordes horizontales y en la 3.25-(c) los verticales, para el cuadro y la circunferencia inscrita en él. En el primer caso se simula una lámpara ubicada a la izquierda de la figura que ilumina los bordes del relieve y en el segundo una luz ubicada en la parte superior de la figura en relieve.

Usando otro tipo de derivada se pueden lograr diferentes efectos de iluminación sobre los objetos en relieve. En las figura 3.26 se utilizó una derivada tipo Robinson con las 8 orientaciones dadas por la rosa de los vientos.

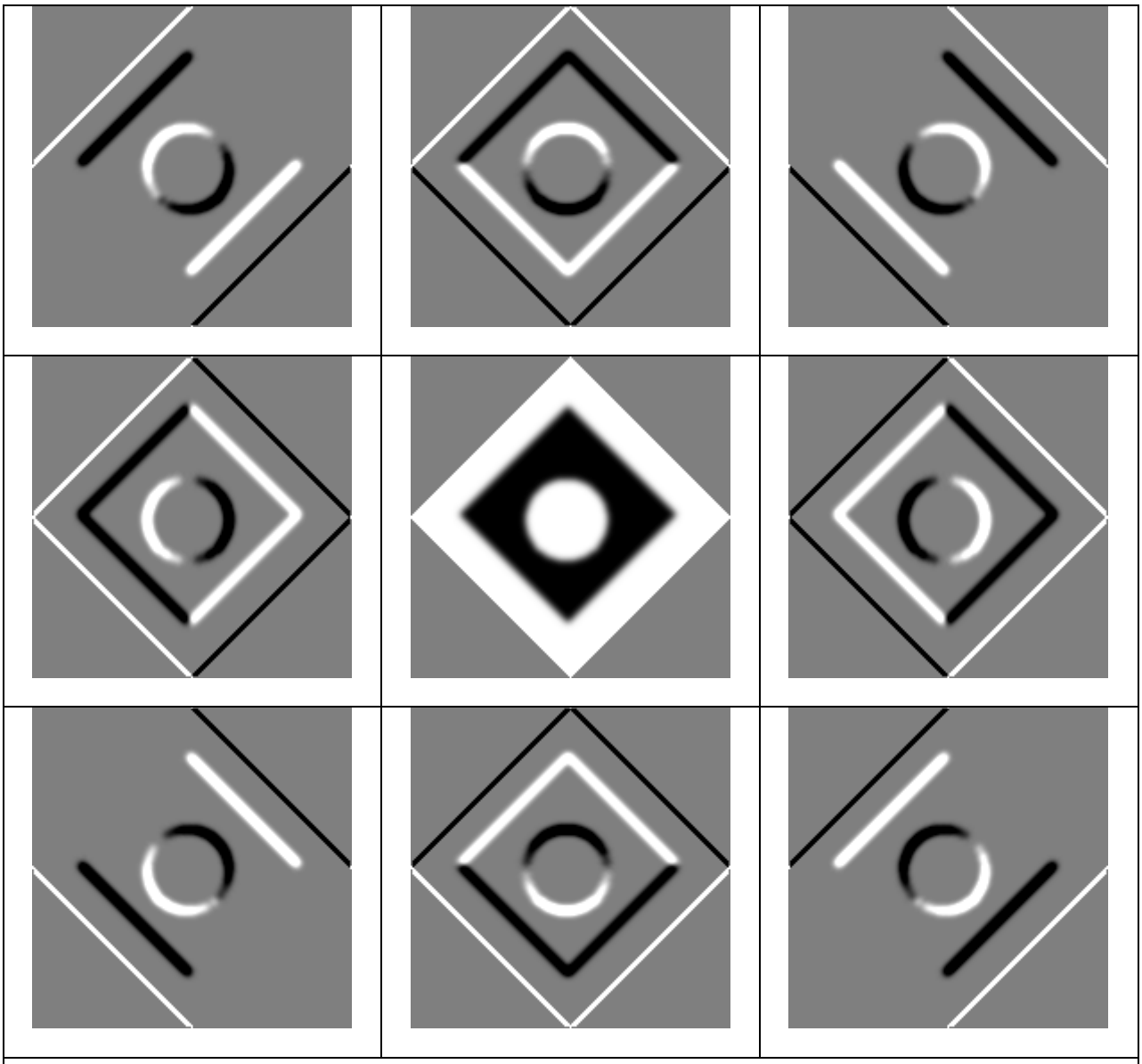


Fig. 3.26. A la imagen central se le ha aplicado un Filtro de convolución de Robinson en las 8 direcciones geográficas.

En la tabla siguiente se muestran las matrices de convolución utilizadas, puede como notarse la simetría y distribución de coeficientes ocasionan los efectos ilustrados en la figura 3.26.

$\begin{pmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$ NO	$\begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$ N	$\begin{pmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix}$ NE
$\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}$ O		$\begin{pmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{pmatrix}$ E
$\begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix}$ SO	$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$ S	$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{pmatrix}$ SE

Fig. 3.26. Matrices de repujado direccional de Robinson

En cada caso la aplicación de éstas matrices se hace mediante la siguiente relación:

$$I'[x,y] = \frac{\Lambda}{2} + \tilde{M}_{Rob}(\theta) \otimes I[x,y] \qquad (3.48)$$

Donde θ indica la dirección deseada, θ = {NO, N, NE, O, E, SO, S, SE}.

Se definen los operadores de segundo orden de Robinson como:

$\begin{pmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{pmatrix}$ NO	$\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$ N	$\begin{pmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{pmatrix}$ NE
$\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}$ O		$\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}$ E
$\begin{pmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{pmatrix}$ SO	$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$ S	$\begin{pmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{pmatrix}$ SE

Fig. 3.27. Matrices de repujado direccional de Robinson de segundo orden

En este caso el mecanismo de aplicación es el mismo que para primer orden y el relieve es más pronunciado.

CAPITULO 4.

INDICE
OPERACIONES GEOMÉTRICAS

Capítulo 4.

Operaciones Geométricas

Fundamentos

Una operación geométrica es aquella que cambia el tamaño, forma u orientación de una imagen. En general no se puede considerar un filtro, pero corresponden a transformaciones útiles en el procesamiento digital de imágenes. Por ejemplo cuando se requiere escalar, orientar, empalmar una imagen se pueden aprovechar dichas operaciones.

En muchos casos al aplicar una operación geométrica cambia el tamaño de las imágenes, esto debe tomarse en cuenta para el desplegado y las operaciones de la interfase de deshacer y rehacer.

Estas operaciones corresponden más al graficado por computadora que al procesamiento digital de imágenes en particular, pero ya que este material trata de dar una visión general de las tareas más importantes del procesamiento se incluyen como tema de importancia.

4.1. Cambios de tamaño.

Las operaciones más frecuentes corresponden a las de cambio de tamaño, a estas muchas veces se les llama operaciones de *zoom* por su analogía con el efecto que hace una lente en una cámara analógica o digital. Vamos a distinguir dos clases de métodos: aquellos que reducen el tamaño y los que lo incrementan. A los primeros se les llama de *reducción* y a los segundos de *ampliación*. En el primer caso es claro que se producirá una pérdida de información en la imagen resultante debido a que el tamaño de la misma es menor y en el segundo caso se tendrán que proponer algoritmos que propongan como estimar el tono o color de los pixeles nuevos que se crean al ampliar la imagen. En este caso debemos estar concientes de que se "inventará" información que no esta presente en la imagen original, lo cual puede producir *artefactos* en la imagen ampliada. Diremos que un artefacto es un elemento que es producido por un algoritmo de transformación que no está presente en la imagen original, pero aparece en la imagen producto de la transformación. En general los artefactos no son deseables pues introducen elementos indeseables que se pueden considerar como ajenos a los datos originales.

Las transformaciones más simples de cambio de tamaño corresponden a la ampliación por 2 en ambas dimensiones y la reducción a la mitad de ellas.

4.1.1. Ampliación al doble simple.

El algoritmo consiste en la duplicación de cada pixel en la imagen resultante, es decir ya que el alto y ancho de la imagen final corresponden al doble del valor de las propiedades correspondientes originales lo mas simple es reproducir el pixel original en los vecinos mapeados.

Si nos enfocamos en el pixel que se encuentra en la coordenada corriente (x, y), entonces en la imagen ampliada le corresponderán cuatro pixeles en la coordenada (2x, 2y) en la imagen ampliada, en la figura 4.1. se muestra el fenómeno.

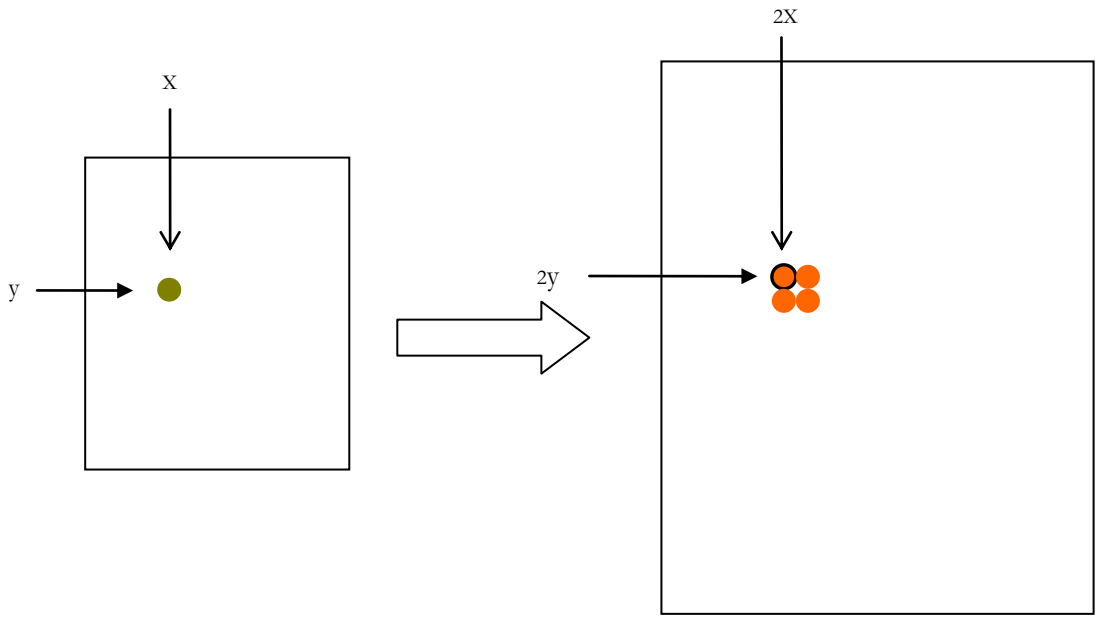


Fig. 4.1. Pixeles correspondientes al mapeo de duplicación del píxel I[i, j].

El algoritmo de ampliación simple basado en la matriz de represtación de la imagen con **C** canales y dimensiones originales en la horizontal (**n)** y en la vertical (**m)** es el siguiente:

```
amplia_2X(Matriz I, Matriz S, n , m, C){
   for (j =0, m-1){ j' = 2j;
     for (i = 0, n-1){ i' = 2i;
        for ( k = 0, C-1){
            v = I[i, j, k];
            S[i'  , j'  , k] = v;
            S[i'+1, j'  , k] = v;
            S[i'  , j'+1, k] = v;
            S[i'+1, j'+1, k] = v;
        }
     }
   }
}
```

Puede verse que el procedimiento corresponde a una copia en los cuatro pixeles de la imagen de destino. Debemos recalcar que las dimensiones de la imagen resultante serán **2n** □ **2m**. Y deberán preparase previamente las dimensiones de la matriz de salida (**S**) y del bitmap que la alojará. A este método se le llama "del flojo", ya que no hay esfuerzo para hacer la ampliación y luego de varias aplicaciones cada pixel crecerá como un cuadro que se duplicará en cada paso, lo cual es un efecto poco agradable, pero no se está creando información nueva.

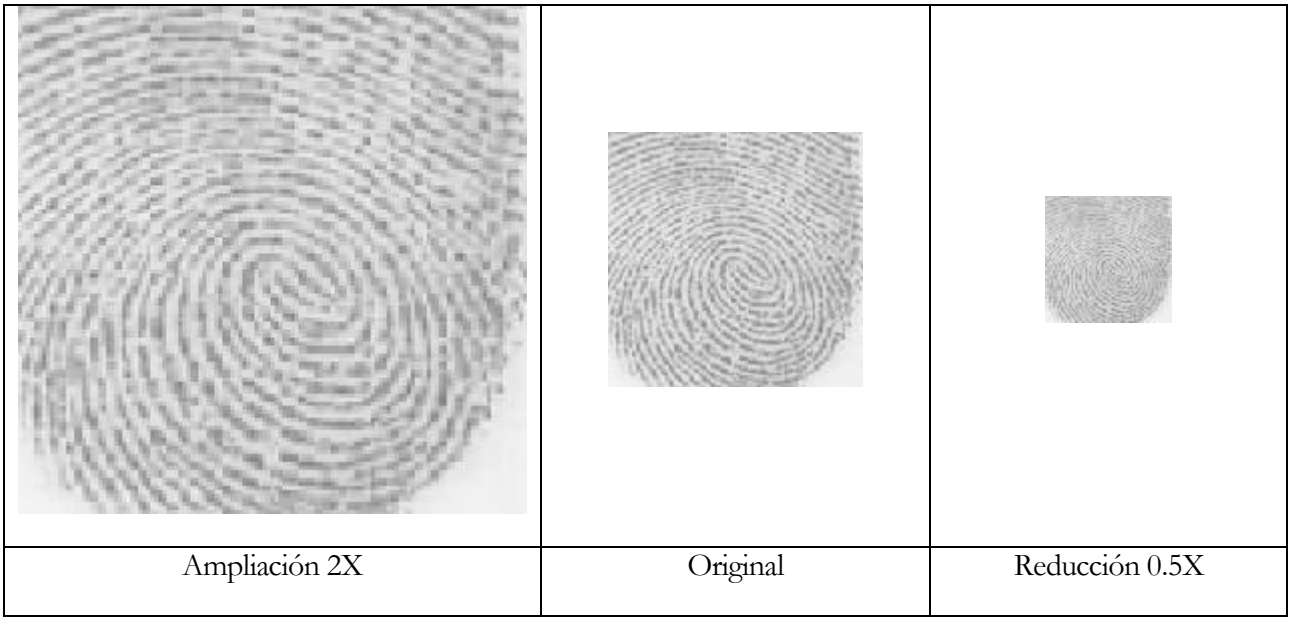
4.1.2. Reducción a la mitad simple.

Considerando que la imagen resultante tendrá la mitad de ancho y alto respecto a la original, el método consiste la inserción de los pixeles pares de la imagen original (0, 2, 4, 6, …) en la imagen resultante y en la eliminación sistemática de pixeles intercalados (1, 3, 5, 7 , …) de la imagen original. El proceso se debe hacer en las dos dimensiones de la imagen original, por tanto en este caso se perderá información al realizarse la reducción.

El algoritmo de reducción simple basado en la matriz de represtación de la imagen con **C** canales y dimensiones originales en la horizontal (**n)** y en la vertical (**m)** es el siguiente:

```
reduce_0_5X( Matriz I, Matriz S, n , m, C){
   n' = [n/2]; // donde[] indica la parte entera
   m' = [m/2];
   for (j =0, m'-1){ j' = 2j;
     for (i = 0, n'-1){ i' = 2i;
        for ( k = 0, C-1){
            S[i, j, k] = I[i', j', k] ;
        }
     }
   }
}
```

Es claro que al realizar este proceso se pierde el 75% de la información que contiene la imagen original y en caso que no se guarde una copia de ella el proceso inverso no recuperará dicha información. En la figura siguiente (Fig. 4.2.) se hace muestra el efecto de éstos dos procesos.



Ampliación 2X	Original	Reducción 0.5X

Fig.4.2. Ejemplo de ampliación y reducción simple.

La imagen original corresponde a una huella digital registrada con una resolución de 127x127 pixeles en tonos de gris, Puede verse como al realizarse la ampliación se produce el fenómeno de pixeleo, es decir los pixeles originales se expanden como un pixel cuadrado.

4.1.3. Ampliación al doble con promedio.

El algoritmo consiste en la estimación del tono de los pixeles a crearse mediante un esquema de interpolación usando los vecinos del pixel original, en general se pueden utilizar diferentes estrategias, se presenta una considerada simple. La idea es promediar los vecinos horizontales y verticales para estimar el tono de los tres pixeles que deben producirse. En el esquema siguiente se muestra de manera esquemática el proceso considerando el pixel de la imagen original en la posición (x, y).

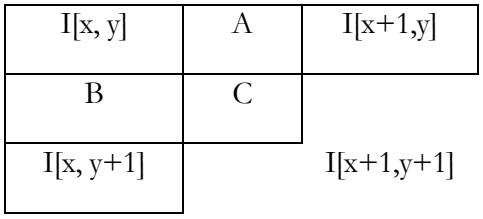


Fig. 4.3. Contribuciones de los pixeles vecinos a los tres estimados

Los tonos asignados a los pixeles A, B y C son los siguientes:

$$A = (I[x, y] + I[x+1, y]) / 2 ,$$

$$B = (I[x, y] + I[x, y+1]) / 2,$$

$$C = (I[x, y]+ I[x+1,y+1]) / 2 .$$

En este caso el algoritmo de ampliación con promedio toma la siguiente forma.

```
amplia_2X_media(Matriz I, Matriz S, n , m, C){
   for (j =0, m-1){ j' = 2j; jp = j+1;
     for (i = 0, n-1){ i' = 2i; ip = i+1;
        for ( k = 0, C-1){
            v = I[i, j, k];
            S[i'  , j'  , k] = v;
            S[i'+1, j'  , k] = (v+I[ip,  j, k])/2;
            S[i'  , j'+1, k] = (v+I[i,  jp, k])/2;
            S[i'+1, j'+1, k] = (v+I[ip, jp, k])/2;
        }
     }
   }
}
```

Este procedimiento no pixelea a la imagen, pero introduce información que no está presente en la imagen original, lo cual para cierto tipo de aplicaciones no es conveniente. El uso del procedimiento se justifica para una visualización donde no es trascendente la información y es más importante la visualización estética de ella. Debe adecuarse el algoritmo en la frontera izquierda e inferior dado que no se cuenta con toda la información necesaria para aplicarlo al pie de la letra.

En la figura siguiente (Fig. 4.3.) se muestra la ampliación simple y la de la media.

Ampliación 2X

Original

Ampliación 2X - media

Fig.4.3. Ampliación simple y por el método de la media.

Puede notarse que la imagen producida usándose la media es más suave y ligeramente borrosa. Esto se debe a que la media como sabemos elimina ciertos detalles. Razón por la cual el pixeleo no es notorio.

4.1.4. Ampliación y reducción por interpolación bilineal.

Una pregunta interesante es, ¿será posible ampliar o reducir en una proporción que no sea el doble o la mitad? La respuesta es afirmativa, esto se logra utilizando un método de interpolación. Vamos a presentar el caso bilineal.

Supongamos que deseamos estimar nuevos valores intermedios de una función - analicemos primero el problema en una variable - dado un conjunto finito de N valores (p_k, q_k) para ella. Supongamos además que la variable independiente se ha evaluado a intervalos uniformes, es decir $p_k = p_o + k\,\Delta p$ (con Δp=cte), de tal suerte que $q_k = f(p_k)$ y k = 0, 1, 2, … N-1. Pensemos que la función ahora deberá ser estimada en M puntos entre el primero de la lista y el último de ella $[p_o, p_N]$.

La nueva partición será ahora $p'_k = p_o + k\,\Delta' p$, donde k = 0, 1, 2…N'-1 y debe cumplirse que $p'_{N'} = p_N$. Una forma ingenua de tratar de hacer la estimación es ir del dominio a la imagen, pero esto es un problema mal planteado. La manera correcta de hacer el *nuevo muestreo* (resampling) es ir de la imagen al dominio.

En la figura siguiente (Fig. 4.4.) se hace una representación gráfica del proceso, algo que podemos esperar es que el primer punto en las variables primadas corresponde al primero de las no primadas y el último de las primadas se va al último de las no primadas, más ¿qué sucede con en k-ésimo de las no primadas? Este en general se va a un punto exacto en la variable no primada o bien a un punto intermedio.

Si asumimos que la evaluación de las variables p y p' se hace en posiciones enteras simples según su índice, entonces podemos decir que el espacio E = [0, N-1] se transforma en el E' = [0,N'-1].

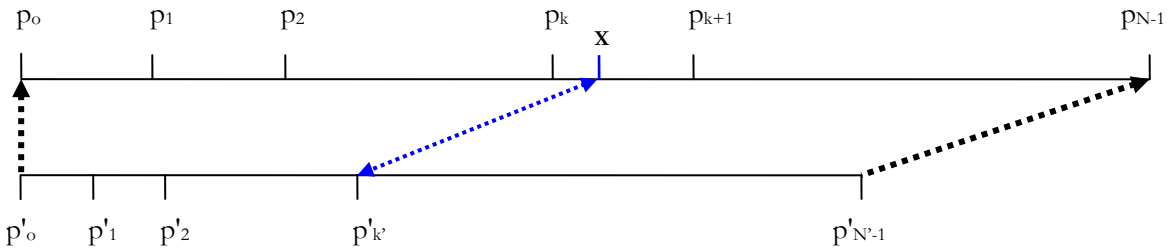


Fig. 4.4. Esquema de interpolación en una dimensión.

Y los valores de la función que son conocidos corresponden a los del espacio E, de tal forma que $p_k=f(k)$ y $k \in E$. Ya que se deben guardar las proporciones entonces:

$$\frac{N}{N'} = \frac{x}{k'}, \qquad (4.1)$$

donde N, N' y k' son enteros, pero x no lo es necesariamente. Ya que lo que queremos calcular es el valor de la función entonces podemos estimar las contribuciones en los extremos enteros que contienen a x, es decir k y k+1. Definiremos la cercanía de x a cada extremo (Fig. 4.5.) como el porcentaje de contribución de cada extremo, de tal forma que mientras mas cerca esté de un extremo la contribución será mayor.

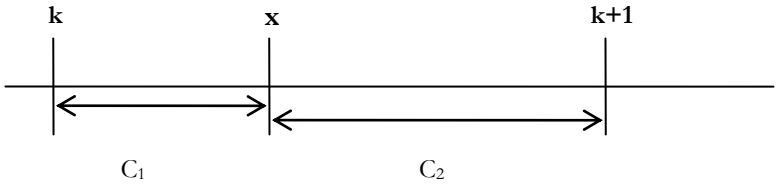


Fig. 4.5. Contribuciones de los extremos a un punto intermedio **x**.

Es claro que la distancia entre **k** y **k+1** es la unidad, por lo tanto $C_1 + C_2 = 1$. Definiremos analíticamente la *cercanía* de **x** al extremo izquierdo como $\alpha = (1-C_1)$ y al extremo derecho como $\beta = (1-C_2)$. Y ya que conocemos los valores de la función en **k** y en **k+1**, entonces el valor estimado en **k'** será:

$$f(k') = \alpha * f(k) + \beta * f(k+1) \qquad (4.2)$$

Así mientras mayor sea la cercanía a un extremo este contribuirá mas a f(k'). Este procedimiento se conoce como *mapeo inverso* o *método de remuestreo*. El algoritmo es el siguiente:

Entrada	: N y las parejas (k, f[k]) con k=0, 1, 2, …N-1. N > 1
	: N , con N'>1 // número de puntos donde se estimará la función.
Salida	: las parejas (k', g[k']).

```
float F, x, d, a ,b
int  k', k

F = N/N'
for (k'=0, N'-1){
  x = F*k'
  k = [x]       // parte entera de x
  d = x-k
  a = 1-d
  b = d

  g[k'] = a*f[k] + b*f[k+1]
}
```

Este método se puede extender a dos dimensiones, ahora debemos considerar que la función será de dos variables, es decir **f** = f(x, y). Donde x e y serán enteros y **f** real en general. Ahora un punto del espacio escalado N'×M' se debe estimar de la función conocida en el espacio N×M.

El método consiste en encontrar el cuadro definido por los puntos (i, j) y (i+1, j+1) que contiene al punto (p, q) que proviene del mapeo de un punto exacto (i', j'), así entonces se deberán calcular las contribuciones por cercanía de los valores conocidos de la función en {(i, j), (i+1,j), (i, j+1), (i+1, j+1)}. En la figura siguiente se esquematiza el proceso.

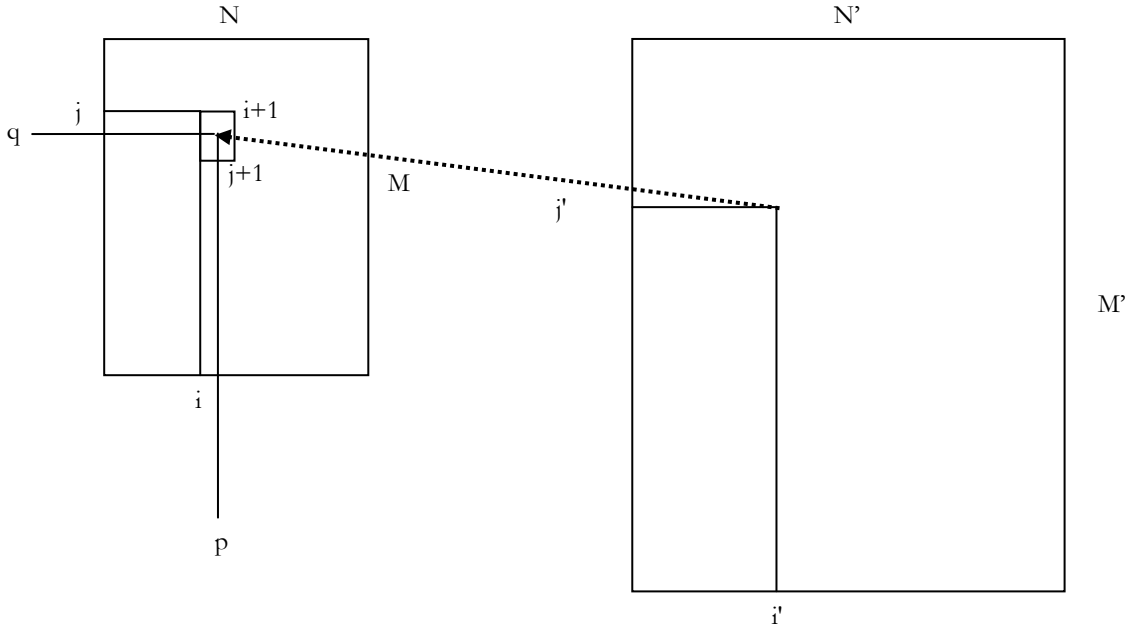


Fig. 4.6. Mapeo inverso bidimensional

Las ecuaciones ahora vendrán en parejas y debemos considerar las contribuciones de cada uno de los cuatro puntos que contienen al punto (p, q). Siguiendo la lógica del mapeo en una dimensión, tendremos que

ahora debemos estimar las contribuciones por cercanía de las cuatro esquinas del "cuadro" donde cae (p,q). Procediendo, tendremos que:

$$p = i' F_1, q = j' F_2 \quad \text{donde} \quad F_1 = \frac{N}{N'}, F_2 = \frac{M}{M'} \tag{4.3}$$

Las cercanías horizontales serán:

$$\alpha = 1 - C_1, \beta = 1 - C_2 \tag{4.4}$$

y las verticales

$$\alpha' = 1 - D_1, \beta' = 1 - D_2 \tag{4.5}$$

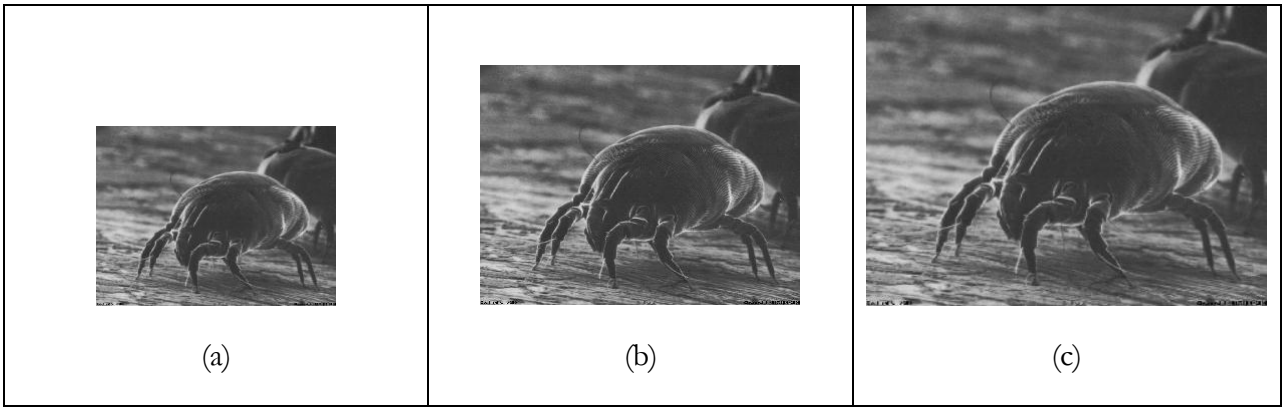
donde

$$C_1 = p - [p], C_2 = 1 - C_1 \quad \text{y} \quad D_1 = q - [q], D_2 = 1 - D_1 \tag{4.6}$$

y las contribuciones se calculan como los productos de las cercanías y los valores de la función conocida, de donde:

$$f'(i', j') = \alpha\alpha' f(i, j) + \beta\alpha' f(i+1, j) + \alpha\beta' f(i, j+1) + \alpha' \beta' f(i+1, j+1) \tag{4.7}$$

Se deja como ejercicio escribir el algoritmo correspondiente al escalado por mapeo inverso bidimensional.



(a) (b) (c)

Fig. 4.7. Se muestra una imagen de Microscopía: (b) Original, (a) reducida al 75%, (c) aumentada al 125%.

En el ejemplo (Fig.4.7) se ha modificado la imagen el mismo porcentaje en ambas dimensiones, a este proceso se le llama *escalado isométrico* debido a que se guardan las proporciones horizontales y verticales simultáneamente.

En general se puede programar el algoritmo de tal manera que los porcentajes de cambio sean diferentes en la dirección horizontal y vertical, a dicho proceso se le llama *escalado libre*.

4.2. Rotaciones.

Otro proceso frecuente sobre las imágenes es el de rotación, éste consiste en girar la imagen un ángulo definido, se pueden desarrollar rotaciones simples sobre ángulos tales como $\pm\pi/2$ ($\pm$ 90º) y π (180º) o bien rotaciones en ángulos arbitrarios θ. El uso de esta transformación se encuentra por ejemplo en la alineación de las imágenes respecto a cierta referencia para realizar una presentación adecuada en la solución de algunos problemas. Una aplicación de la rotación simple de $\pm$ 90º es para ofrecer una vista de una imagen que fue adquirida con el dispositivo de registro perpendicular al modo estándar, por ejemplo por una cámara digital o un escáner.

Un detalle a considerarse en el proceso de rotación es el hecho de que la imagen puede cambiar de tamaño respecto a la original y una zona de la imagen rotada contenida en un lienzo rectangular deberá ser llenada con un color arbitrario.

4.2.1. Rotación simple de $\pm$ 90º.

Este procedimiento se puede considerar como un reacomodo de los pixeles que conforman a la imagen original y el problema consiste en saber ¿dónde se ha de reubicar cada pixel?, consideremos como imagen base la siguiente (Fig. 4.8.), ésta contiene un trazo que nos permite analizar e identificar el destino de cada pixel luego de la rotación. Consideremos las dimensiones horizontal y vertical en pixeles de la imagen como N y M respectivamente.

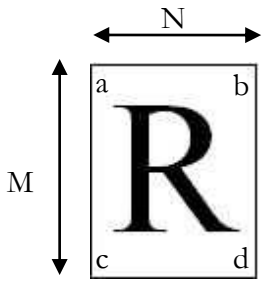


Fig.4.8. Figura base para el estudio de las rotaciones

Al efectuar una rotación de 90º en la dirección contraria a las manecillas del reloj respecto al centro de la imagen, se producirá una transposición de las dimensiones de la imagen de tal forma que el número de columnas será ahora M y el de renglones N.

Los puntos de referencia {a, b, c, d}(esquinas de la imagen)se ubicarán en nuevos lugares como se muestra en la figura 4.9.

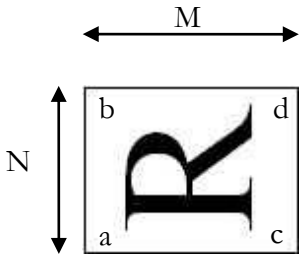


Fig.4.9. Figura base rotada 90º.

Podemos notar que si la información de la imagen original está en una matriz R[i, j], entonces i $\in$ [0, N-1] y j $\in$ [0, M-1]. Si R'[i', j'] contiene a la imagen rotada entonces i' $\in$ [0, M-1] y j' $\in$ [0, N-1]. Esto quiere decir que los índices han intercambiado sus recorridos, para encontrar la relación entre R y R' podemos fijarnos en los puntos de referencia. Puede verse que los renglones {a□b}se han convertido en columnas, pero el recorrido se hace de forma inversa {b□a}. Por otro lado las columnas {a□c}se han convertido en renglones y el recorrido no ha cambiado de dirección en la imagen rotada.

Así entonces la relación entre las matrices se puede establecer como:

$$R'[i, j] = R[j, N-1-i] \qquad (4.8)$$

Se han simplificado la notación para los índices para facilitar la implementación y pude observarse que para cada matriz los rangos son los correctos. De aquí (Ec. 4.8.) se pude extraer el algoritmo de rotación, éste se puede escribir de la siguiente manera:

```
rota_90(mat R,R'; int M,N){
  Np = N-1
  for (i=0,N-1){ip = Np-i
    for (j=1,M-1)
      for (c=0,Nc-1)
        R'[i, j, c] = R[j, ip, c]
  }
}
```

Donde *Nc* indica el número de canales de la imagen. Se deja como ejercicio analizar el problema de la rotación de -90º, es decir 90º en sentido de las manecillas del reloj y escribir el algoritmo respectivo.

4.2.2. Rotación simple de 180º.

Tomando como referencia la Fig. 4.8., es claro que no tiene caso proponer una rotación a la derecha y otra a la izquierda, ya que ambas llevan al mismo resultado. El resultado de dicha rotación se muestra en la Fig. 4.10. En este caso es claro que las dimensiones de la imagen no se han modificado, el número de columnas y reglones sigue siendo el mismo. Y lo que ha sucedido es un cambio en la posición de los pixeles de tal manera que los índices de los renglones y las columnas ahora corren al revés. La relación entre la matriz original R[i, j] y la transformada R'[i'. j'] se puede mostrar que es:

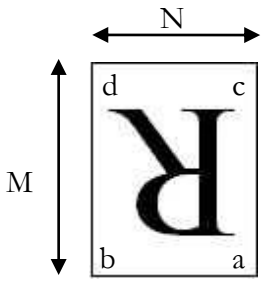


Fig.4.10. Figura base rotada 180º.

$$R'[i, j] = R[N-1-i, M-1-j] \qquad (4.9)$$

De aquí se puede escribir el algoritmo correspondiente a la rotación de 180º.

```
rota_180(mat R,R'; int M,N){
  Np = N-1
  Mp = M-1
  for (i=0,N-1){ip = Np-i
    for (j=1,M-1){jp = Mp-j
      for (c=0,Nc-1)
        R'[i, j, c] = R[ip, jp, c]
  }
}
```

4.2.3. Rotación libre directa.

Una forma sencilla de realizar una rotación libre un ángulo θ es utilizando la matriz de rotación que nos ofrece la geometría plana, esta matriz tiene la forma:

$$\tilde{R}(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \qquad (4.10)$$

Las propiedades más relevantes de ella son:

1. No se modifican las dimensiones de los objetos respecto al centro de rotación.
2. Su determinante es la unidad.

3. Es una transformación lineal, lo cual implica que $\tilde{R}(\alpha+\beta)=\tilde{R}(\alpha)+\tilde{R}(\beta)$.
4. Su inversa es $\tilde{R}^{-1}(\theta)=\tilde{R}(-\theta)$, de tal suerte que
   $\tilde{R}^{-1}(\theta)\tilde{R}(\theta)=\tilde{R}(-\theta)\tilde{R}(\theta)=\tilde{R}(\theta-\theta)=\tilde{R}(0)=\tilde{I}=identidad$

Entre los problemas que debemos atender para aplicar esta transformación están los siguientes:

1. Se debe definir un centro de rotación para aplicar la transformación.
2. Antes de aplicar la transformación será necesario calcular el tamaño de la imagen rotada y de éste las dimensiones de la matriz donde se almacenará.
3. Dado que los lienzos son rectangulares, se deben rellenar las esquinas de la imagen rotada con algún color arbitrario, ya que para la mayoría de los valores de θ la imagen rotada será más grande que la original (las excepciones se dan para θ=±90º y 180º). Esto introducirá información que no esta presente en la imagen original.
4. Como θ es arbitrario en general al transformar la coordenada (x, y) de algún pixel, dado que x e y son enteros, la transformación nos regresará un número real y nos veremos obligados a redondear, esto puede provocar que más de un pixel de la imagen original se mapee en el mismo en la imagen transformada. Esto puede provocar que algunos pixeles no se llenen y se produzcan huecos (que forman patrones de tipo mosaico) en la imagen transformada.

El primer aspecto se puede resolver usando el "centro" de la imagen como eje de rotación, esto implica que se debe aplicar una transformación de translación T(-N/2, -M/2), rotar y luego deshacer la traslación inicial T(N/2, M/2), esta es la regla que las técnicas estándares de graficado indican. Es normal que este proceso genere coordenadas negativas, motivo por el cual se debe introducir una corrección ya que los índices de las matrices no deben ser negativos en general.

El segundo problema se puede resolver mapeando las esquinas de la imagen original, es decir transformar los puntos {(0, 0), (N-1, 0), (0, M-1), (N-1, M-1)}, y a partir de su mapeo encontrar las dimensiones del lienzo que contiene a la imagen rotada.

El tercer asunto se puede resolver eligiendo un color de fondo, se llena la matriz de destino ya dimensionada y luego se mapea la imagen original punto a punto.

Y finalmente el cuarto problema (redondeo) se incluye en el algoritmo al momento de hallar los índices del pixel transformado.

A continuación se irá construyendo de manera modular el método de rotación considerando los problemas y algunas posibles soluciones a los problemas antes enunciados.

Fase 1. Mapeo de las esquinas.
Si aplicamos la traslación al centro de la imagen y aplicamos la matriz de rotación a las esquinas del lienzo que contiene a la imagen obtendremos cuatro puntos, llamemos a las esquinas {x[k], y[k] : k = 0…3} y a los puntos transformados { p[k], q[k] : k = 0…3}. El algoritmo de transformación puede quedar de la siguiente manera:

1. Copiar las esquinas al arreglo.
   x[0] = 0 ; y[0] = 0; x[1] = N-1; y[1] = 0; x[2] = 0; y[2] = M-1; x[3] = N-1; y[3] = M-1;

2. Aplicar la transformación compuesta a cada punto:

$$\begin{pmatrix} p \\ q \end{pmatrix}_K = \tilde{T}(-N/2,-M/2)\tilde{R}(\theta)\tilde{T}(N/2,M/2)\begin{pmatrix} x \\ y \end{pmatrix}_K \qquad (4.11)$$

De forma operativa tendremos:

```
N2 = N/2; M2 = M/2;
ct = cos(θ) ; st = sin(θ);
for (k=0,3){
  xp   = x[k] – N2;      yp   = y[k] - M2;
  xr   =  xp*ct + yp*st; yr   = -xp*st + yp*ct;
  p[k] = xr + N2;        q[k] = yr + M2;
}
```

Puede notarse que el punto central de la imagen y las funciones trigonométricas se han calculado una sola vez. Como se discutió anteriormente p[k] y q[k] pueden ser negativos, en la figura 4.11. se muestra el fenómeno.

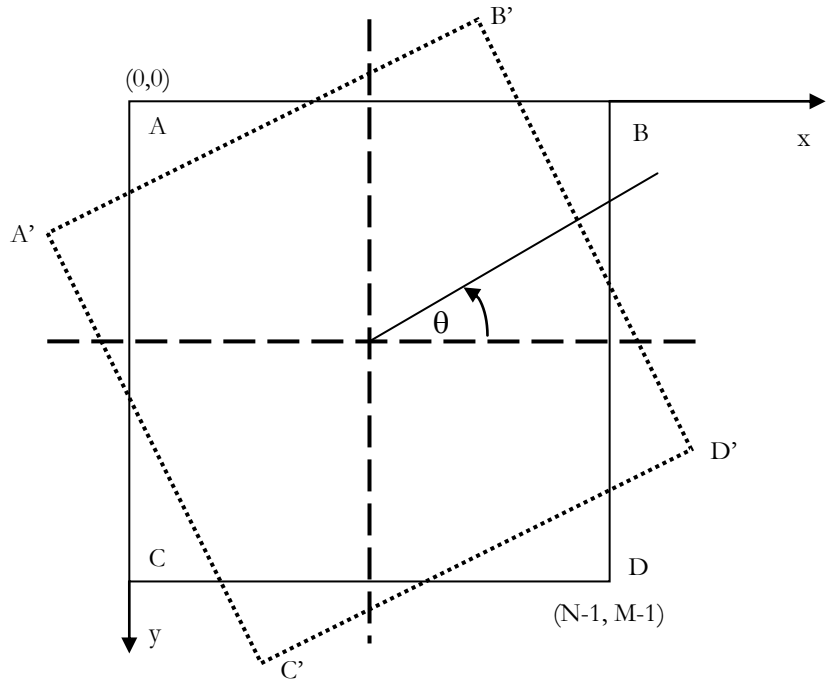


Fig. 4.11. Rotación de las esquinas del lienzo que contiene a la imagen

El rectángulo punteado representa el lienzo rotado y los ejes a trazos el centro de rotación. Respecto al sistema original de coordenadas del lienzo es notorio que el punto A' tendrá su abscisa negativa, el B' su ordenada negativa, D' su abscisa mayor que N-1 y C' su ordenada mayor que M-1. Es claro que pueden presentarse otros casos si cambiamos el ángulo de rotación θ. Para determinar el tamaño del nuevo lienzo podemos calcular las cotas de los puntos p[k] y q[k], de donde podemos definir:

```
p1 = min{p[k]};   p2 = max{p[k]};   q1 = min{q[k]};   q2 = max{q[k]};
```

donde k= 0…3. Con estos valores se tendrá que el tamaño del lienzo que contiene a la imagen rotada será:

$$Np = p2 - p1 + 1;\ Mp = q2 - q1 + 1;$$

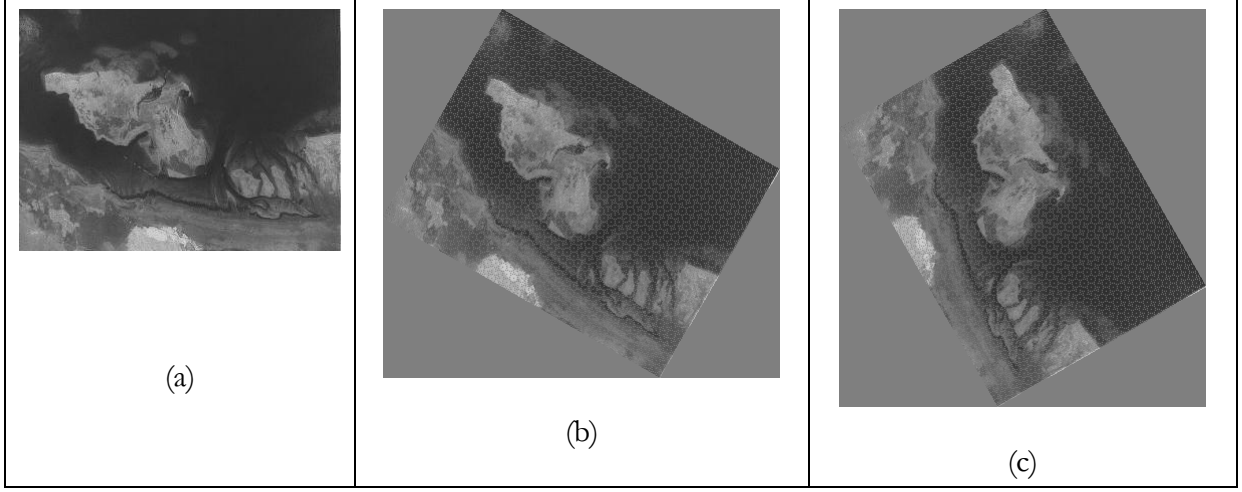
Y se deberá considerar el corrimiento que introducen p1 y q1 para que las coordenadas en el nuevo lienzo sean no negativas.

Fase 2. Calculo de los nuevos índices luego de la rotación.

Con los parámetros antes evaluados podemos rotar la imagen transformando los índices de la matriz que la representa, el algoritmo puede quedar de la siguiente manera:

```
// Redimensionar la matriz de salida R': Np×Mp
// La matriz de entrada R tiene dimensiones N×M

// Llenamos la matriz de salida con un color de fondo arbitrario
for (j = 0, Mp-1)
  for (i = 0, Np-1)
     for (c= 0, Nc)
           R'[i, j, c] = ColFondo;

// sx y sy son los corrimientos acumulados
sx = p1 + N2;
sy = q1 + M2;
for (j = 0,M-1){yp = j - M2;
  for (i = 0, N-1){xp = i – N2;
    xr =  xp*ct + yp*st;
    yr = -xp*st + yp*ct;

    ip = [xr + sx];
    jp = [yr + sy];

    for (c = 0,Nc) R'[ip, jp , c] = R[i, j, c];
  }
}
```

En la figura siguiente se muestra una imagen y un par de rotaciones (30º y 60º) de ella, se utilizó como color de fondo un gris neutro (127, 127, 127).



(a) (b) (c)

Fig. 4.12. Una imagen (a) y dos rotaciones directas de ella (b) 30º (c) 60º

La figura 4.13. muestra una ampliación de la imagen rotada 45º, pueden verse los *artefactos* en forma de patrones periódicos que se producen por la rotación libre directa.

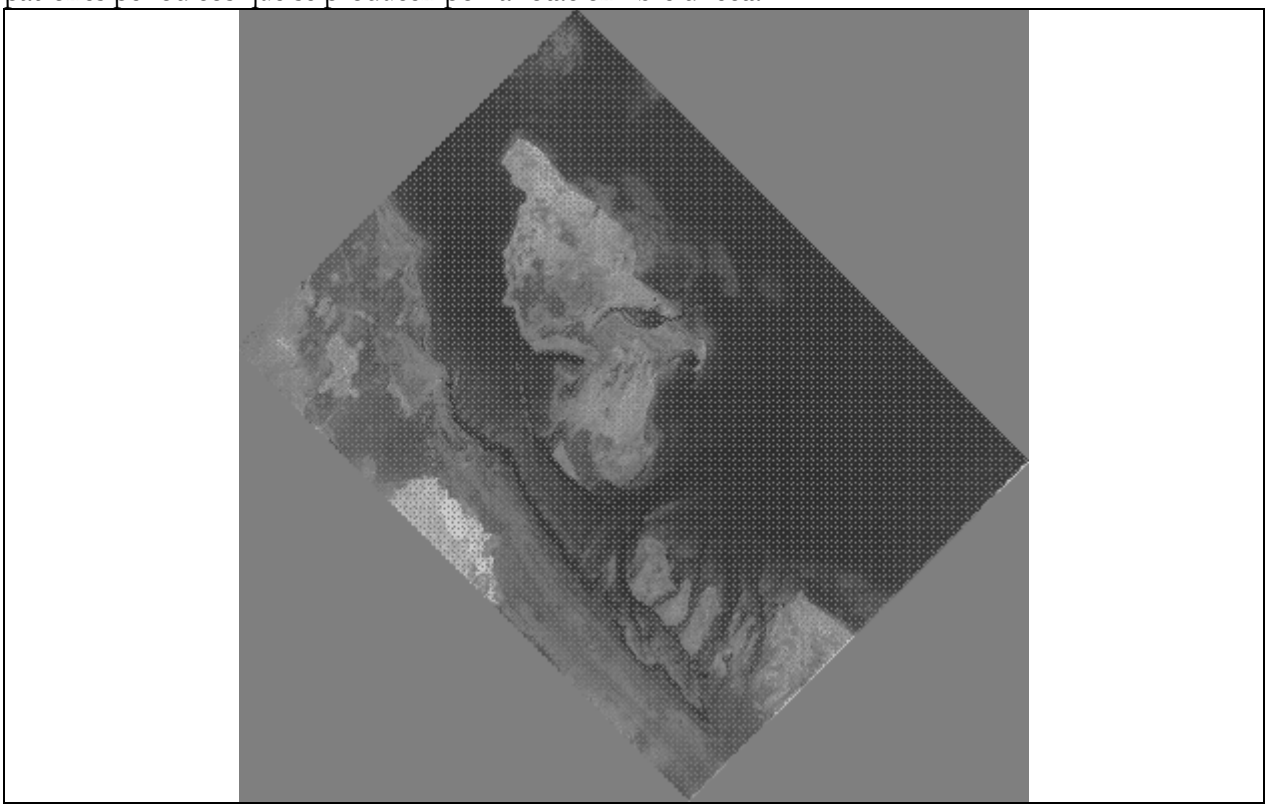


Fig. 4.13. Ampliación de la imagen rotada 45º.

Los artefactos tienen una estructura que depende del ángulo de rotación aplicado. A este fenómeno se le llama *generación de puntos ciegos*, este es introducido por el redondeo que se hace luego de transformar los índices de las matrices que representan a la imagen.

4.2.4. Rotación libre inversa con interpolación lineal.

Una manera de evitar estos artefactos es utilizar la idea que se manejo en el escalamiento libre en la sección 4.1.4., es decir si en vez de enviar los índices de la imagen original a la rotada utilizamos el algoritmo de interpolación lineal inversa y construimos cada pixel de la imagen rotada a partir de las contribuciones de la original entonces no habrá puntos ciegos, pero introduciremos un ligero suavizamiento de la imagen debido a la interpolación.

Los pasos para realizar la rotación son similares hasta la *fase 1* antes propuesta y el cambio esencial se hará en el proceso de mapeo o sea cambiaremos la *fase 2* por un procedimiento de interpolación inversa.

Fase 2a. Interpolación lineal inversa para la rotación.

Dado que ya conocemos el tamaño del lienzo nuevo que recibirá la matriz rotada primero procederemos a llenarlo con algún color de fondo arbitrario. Posteriormente vamos a mapear un punto del lienzo rotado y evaluaremos las contribuciones por cercanía de los pixeles que contienen el punto transformado. La transformación ahora se deberá hacer en *sentido contrario*, esto le da el nombre de ***inversa***, es decir dado un punto con índices enteros en la imagen rotada (i, j) debemos encontrar su correspondiente en la imagen original (x, y) como un real en general. La transformación toma la forma:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \tilde{T}(-N'/2,-M'/2)\tilde{R}(-\theta)\tilde{T}(N'/2,M'/2)\begin{pmatrix} i \\ j \end{pmatrix} \qquad (4.12)$$

El algoritmo puede quedar de la siguiente manera:

```
// note el cambio de signo en st = sin(-θ) = -sin(θ)
//  xr = ct*x' - st*y'
//  yr = st*x' + st*y'
//
Np2 = Np/2; Mp2 = Mp/2;
for (j = 0, Mp-1){yp = j - Mp2;  ys = - st*yp; yc = ct*yp;
  for (i = 0, Np-1){xp = i - Np2;
    xr = ct*xp + ys;
    yr = st*xp + yc;

    x = xr + Np2;
    y = yr + Mp2;

    ip = [x]; // parte entera de x
    jp = [y]; // parte entera de y

    i1 = ip+1; // índice derecho a i
    j1 = jp+1; // índice inferior a j

    c1 = x - ip;
    d1 = y - jp;

    a1 = 1 - c1; // a1 = 1 - c1 = c2
    b1 = c1    ; // b1 = 1 - c2 = c1

    a2 = 1 - d1; // a2 = 1 - d1 = d2
    b2 = d1    ; // b2 = 1 - d2 = d1

    f1 = a1*a2; f2 = b1*a2; f3 = a1*b2; f4 = a2*b2;

    for (c=0,Nc)
      R'[i,j,c] = f1*R[ip,jp,c]+f2*R[i1,jp]+f3*R[ip,j1,c]+f4*R[i1,j1]
  }
}
```

En el código se han introducido algunas optimizaciones para evitar el cómputo repetitivo de algunos factores que participan en las evaluaciones. Se incluyen algunos comentarios explicativos referentes a la factorización de términos en los ciclos sobre **i** y **j** de cómputo.

En las figuras de 4.14 se muestra la aplicación de la rotación libre inversa con interpolación lineal, puede verse en ellas que no aparecen artefactos en la imagen rotada.

En la figura 4.15. se muestra una ampliación de la imagen rotada 47º para poder observar que no hay artefactos importantes presentes.

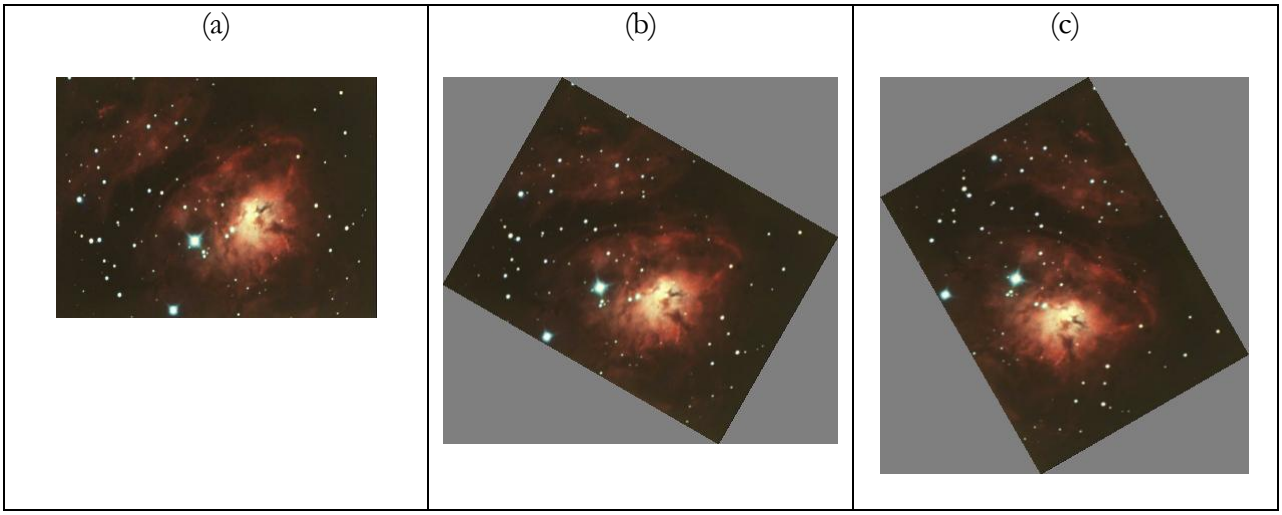


Fig. 4.14. Una imagen (a) y dos vistas con rotación libre inversa de 30º (b) y 60º (c).

Es obvio que este procedimiento de rotación si bien es más complicado ofrece un mejor resultado que la rotación libre directa y se compone de la rotación aunada a la interpolación lineal por cercanía.

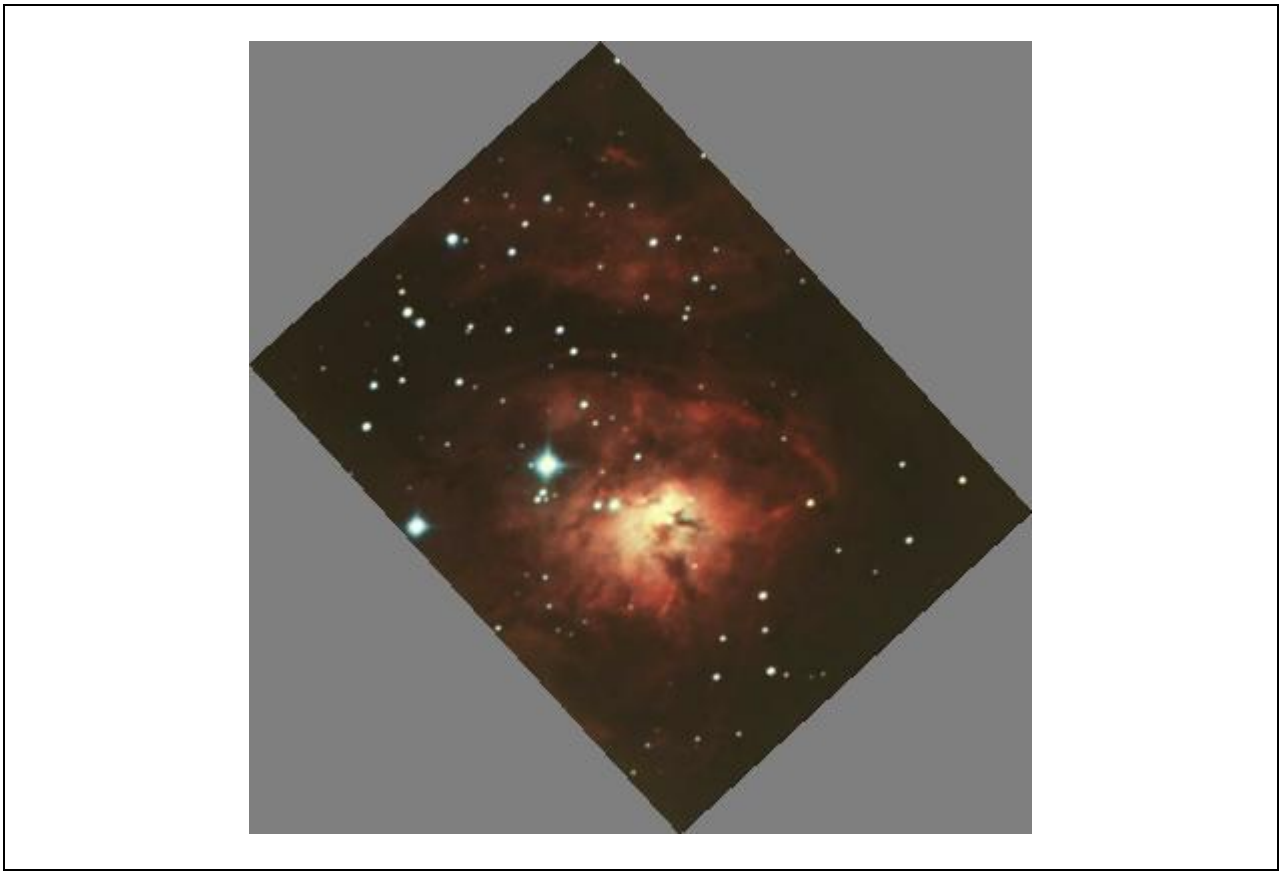


Fig. 4.15. Imagen rotada 47º, note que no hay artefactos notorios.

4.3. Otras operaciones de interés.

Es posible definir otras operaciones geométricas especiales que pueden ser útiles para la correcta presentación de algunas imágenes. A continuación se enuncian algunas, se deja como ejercicio su implementación.

4.3.1. Reflexión horizontal.

Esta consiste en intercambiar las columnas de una imagen. La figura 4.16. ilustra la transformación.

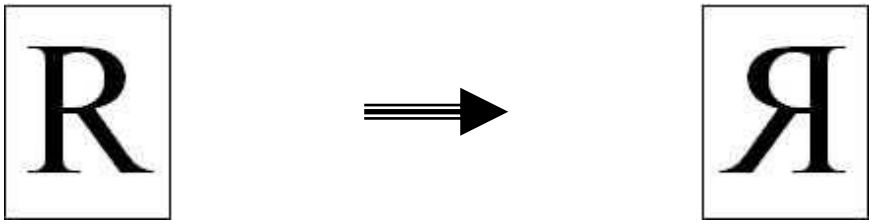


Fig. 4.16. Reflexión horizontal

4.3.2. Reflexión vertical.

Esta consiste en intercambiar los renglones de una imagen. La figura 4.17. enseña el efecto.

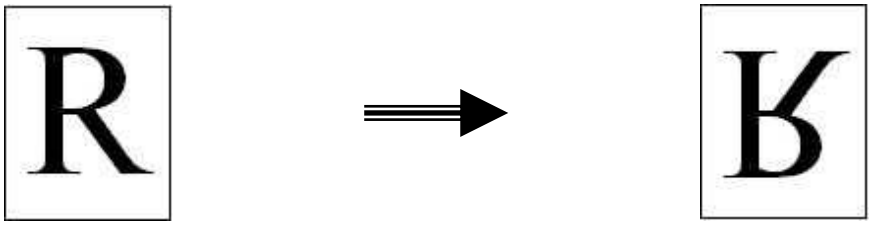


Fig. 4.17. Reflexión vertical

4.3.3. Reflexión doble.

Esta consiste en intercambiar las columnas y renglones de una imagen. La figura 4.18. muestra el cambio.

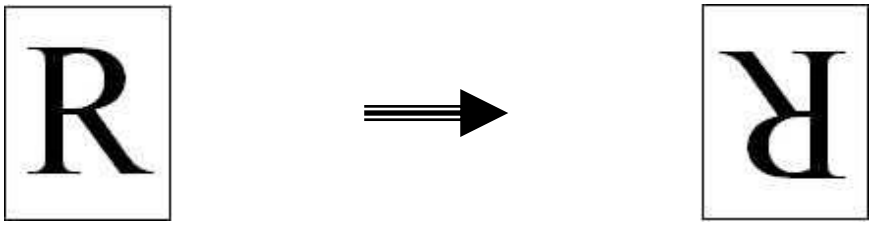


Fig. 4.18. Reflexión doble

4.3.4. Estiramiento horizontal.

Esta consiste en hacer un zoom sólo en la dirección horizontal. La figura 4.19. indica el resultado.

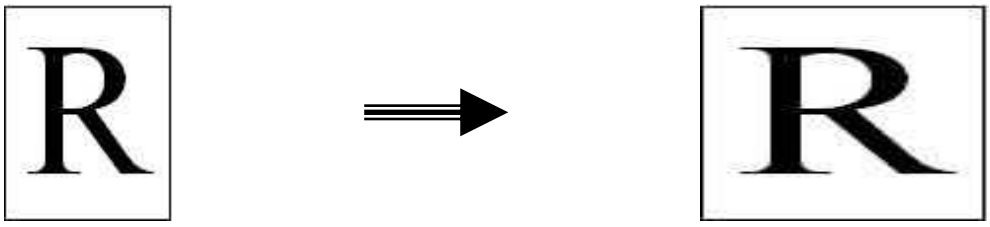


Fig. 4.19. Estiramiento horizontal

4.3.5. Estiramiento vertical.

Esta consiste en intercambiar las columnas de una imagen. La figura 4.20. ilustra el proceso.

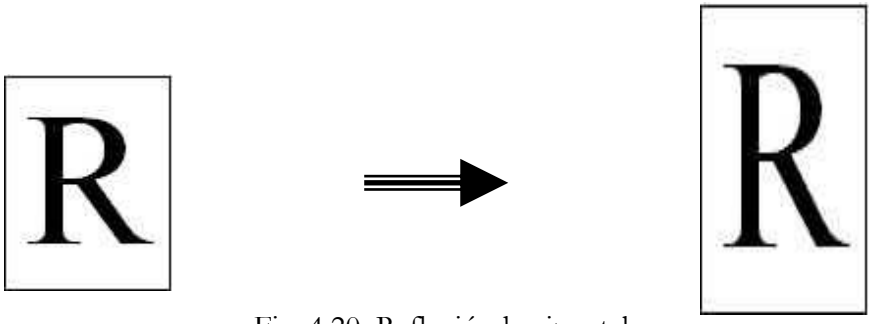


Fig. 4.20. Reflexión horizontal

4.4. Operaciones geométricas complejas.

Para algunos problemas se deben aplicar operaciones geométricas de mayor complejidad. Por ejemplo cuando se adquiere una imagen satelital puede ser muy pronunciado el efecto de la curvatura de la tierra y para que la vista de la imagen mantenga las proporciones usuales se debe hacer una corrección geométrica. Esta debe corregir el efecto de curvatura y producir una imagen plana.

CAPITULO 5.

Operaciones entre Imágenes

Capítulo 5.

Operaciones entre Imágenes

Fundamentos

En muchos problemas el objeto de trabajo no es una imagen aislada sino se ven involucradas una pareja o una serie de imágenes. El tipo de operaciones de éste tipo de mayor uso corresponden a el cálculo de diferencias y la fusión de varias imágenes, más es posible implementar otro tipo de operaciones.

En muchos casos el resultado de la operación entre imágenes genera una imagen de un tamaño distinto a los objetos inicialmente involucrados, razón por la cual de la misma manera que en las operaciones geométricas la situación debe ser tomada en cuenta para el manejo del resultado.

Las problemas base que reciben mayor beneficio de estas operaciones son aquellos en los que se ven involucradas comparaciones entre imágenes. Un problema que utiliza estas rutinas en su forma más primitiva es el correspondiente a la detección de movimiento, de manera general podemos decir que la solución simple se alcanza hallando la diferencia entre dos imágenes obtenidas de manera secuencial en el tiempo y que corresponden a un mismo escenario en el cual uno a varios elementos han cambiado de posición. La diferencia entre los cuadros puede ayudar a encontrar el cambio de posición de algunos objetos.

5.1. Planteamiento General.

Como se ha discutido en este material una imagen se puede manejar a través de su representación matricial, donde *M[i, j, c]* representa el pixel ubicado en la posición *(i,j)* correspondiente al canal *c*, donde el primer índice corre entre 0 y n-1, el segundo entre 0 y m-1, y c entre 0 y nc-1, siendo n el ancho de la imagen, m el alto de ella y nc el número de canales que se han adquirido.

Si ahora consideramos dos imágenes en ésta representación (M_1 y M_2) es posible implementar operaciones *aritméticas* y *lógicas* entre ellas a nivel de pixeles, donde al operar cada pareja obtenemos un resultado que puede ser almacenado en una nueva matriz (M_R), la cual corresponde a la imagen resultante del proceso. Si asumimos que el operador Δ indica una operación binaria bien definida entonces podemos construir la siguiente asociación:

$$M_R[i, j, c] = M_1[i, j, c] \Delta M_2[i, j, c]. \quad (5.1)$$

Esta indica que en la posición (i, j) para cada canal de la imagen resultado asignamos el valor que arroja la operación Δ al operar los pixeles en la posición (i, j) de las imágenes base.

5.1.1. Elección de rangos.

El algoritmo debe correr los índices i y j, pero en general M_1 y M_2 no tendrán las mismas dimensiones, por lo tanto no es evidente decir en que rango han de moverse dichos índices. En la fig. 5.1a se muestra un caso referente a ésta indeterminación. Existen dos estrategias para eliminarla, la primera y mas simple corresponde a alinear las imágenes base en su borde izquierdo superior (fig. 5.1b), que corresponde al índice matricial (0, 0, c) y operar solamente la región de ellas que se traslapa, es decir el área común o intersección, esto definirá el rango de aplicación y a su vez el tamaño de la matriz resultante. Es fácil deducir que los rangos en cada dimensión están definidos como:

$$nn = \max\{n1, n2\}, \; mm = \max\{m1, m2\}. \quad (5.2)$$

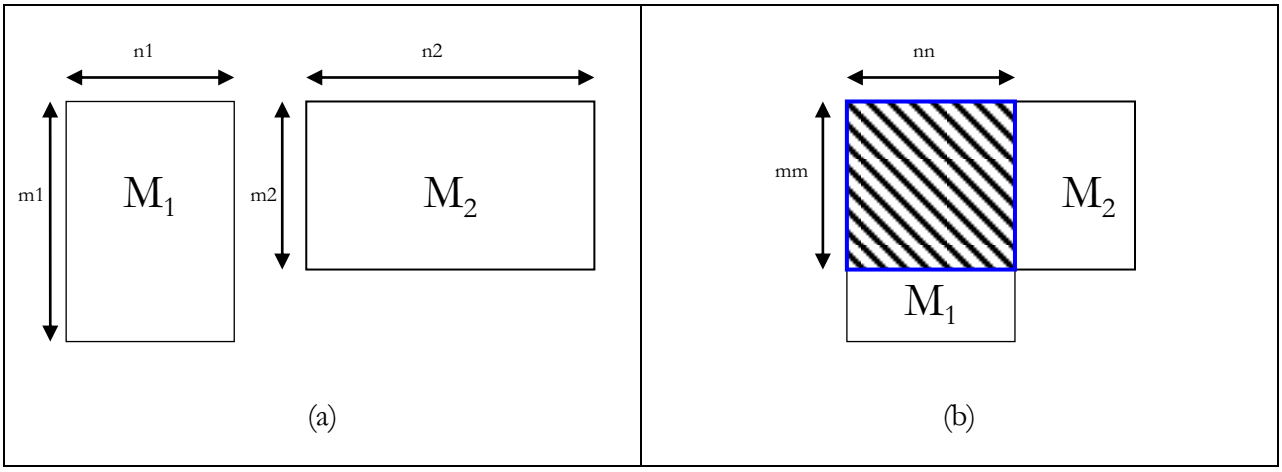


Fig. 5.1. (a) Pareja de imágenes a operar, (b) zona común de las imágenes

Donde n_i corresponde al ancho de cada imagen y m_i a sus altura. Con estas salvedades podemos proponer un algoritmo de operación entre dos imágenes, vamos a definir este como *método de alineación superior*.

Método de alineación superior

```
Entrada  : M1 de n1×m1×nc,  M2 de n2×m2×nc
Salida   : MR, nn, mm.

nn = max(n1, n2)
mm = max(m1, m2)
dimensionar MR a nn×mm×nc,
for (j=0, mm-1)
  for (i=0, nn-1)
     for (c=0, nc)
        MR[i, j, c] = M1[i, j, c]  Δ  M1[i, j, c]
```

En el caso que una imagen sea mas pequeña que la otra en ambas dimensiones (supongamos que M2 es la menor) es posible producir un efecto de montaje, es decir elegir un punto dentro de M1 (X0, Y0) respecto al cual se haga la alineación de M2, y en ese punto se aplica la operación entre pixeles para la región común entre m1 y M2. En la fig. 5.2. se muestra la idea.

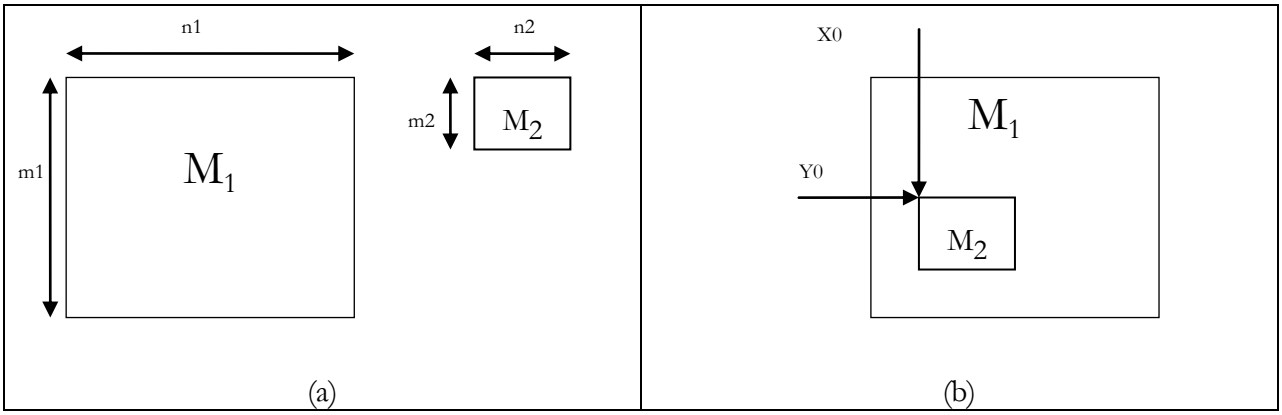


Fig. 5.2. (a) Pareja de imágenes a operar, (b) colocación de M2 dentro de M1 en el punto (X0,Y0)

A esta estrategia definida por las restricción de que M2 este inscrita en M1 se le puede llamar *Método de inscripción*. El algoritmo puede tomar la siguiente forma:

```
Método de inscripción

Entrada  : M1 de n1×m1×nc,  M2 de n2×m2×nc, (X0,Y0)
Salida   : MR.

dimensionar MR a n1×m1×nc
copiar M1 en MR

j = Y0; jj = 0
while (j<m1)&& (jj<m1){
  i = X0; ii = 0
  while (i<n1) && (ii<n2){
    for (c=0, nc) MR[i,j,c] = M1[i,j,c]  Δ  M1[ii,jj,c]
    i++; ii++
  }
  j++; jj++
}
```

En este algoritmo se han cambiado los *for* por *while* con el propósito de evitar operar pixeles fuera de rango en las imágenes de entrada.

5.1.2. Operaciones típicas: suma y resta.

Los operadores más simples corresponden a la suma y resta de imágenes. En el caso de la suma debemos considerar la posible saturación, ya que como los valores de los tonos de cada pixel varían entre [0, Λ-1], entonces la suma puede alcanzar un valor máximo de 2* (Λ-1), lo cual no es admisible por la capacidad de representación tonal en la imagen, por lo tanto debemos realizar una normalización, la típica corresponde a introducir un factor multiplicativo de ½ luego de realizar la operación, esto limitará la salida tonal al rango base [0, Λ-1]. Otra alternativa es utilizar la función de acotación de rango superior (*fars*):

$$fars(x,y) = \begin{cases} x+y & x+y < \Lambda - 1 \\ \Lambda - 1 & x+y \geq \Lambda \end{cases} \qquad (5.3)$$

En la siguiente figura (fig. 5.3.) se muestra el resultado de cada norma.

A	B	(A+B)/2	fars(A, B)

Fig. 5.3. Suma de imágenes: con norma ½ y con norma *fars*

Puede notarse como la norma ½ entrega una imagen mas obscura que la correspondiente a la norma *fars*.

Es claro que las imágenes elegidas son un poco obscuras, en el caso que hayan sido muy claras se hubiera producido en ambos casos saturación (fig.5.4.).

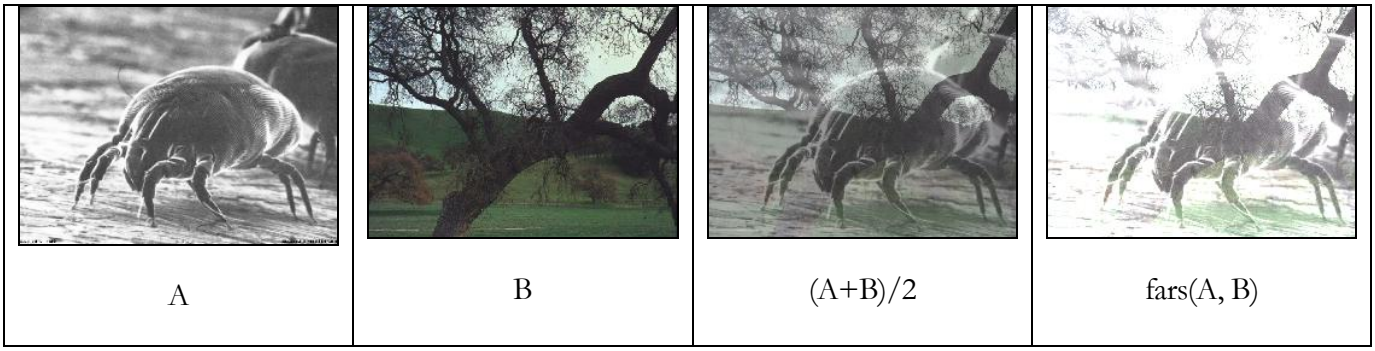


Fig. 5.4. Suma de imágenes: con norma ½ y con norma *fars*. Observe la saturación.

En aquellos casos que se deba realizar la suma y se quiera evitar la saturación, es posible construir una combinación lineal entre los pixeles de las imágenes que no tenga peso (½) para cada uno. Una forma simple es construir una salida es hacer una *combinación lineal* de la forma

$$z = \alpha x + \beta y \text{ , donde } \alpha+\beta=1. \qquad (5.4)$$

Esto permite ponderar la importancia de cada imagen en el resultado. La condición $\alpha+\beta=1$ se puede imponer o bien se pueden dejar libres α y β dependiendo del problema. En la siguiente secuencia se hace una composición mediante combinación lineal con las imágenes de la figura 5.4.

Fig. 5.5. Combinación Lineal de imágenes con diferentes valores de α, usando $\alpha+\beta=1$.

La **resta** entre imágenes se puede utilizarse por ejemplo para encontrar diferencias entre dos cuadros, en particular es posible implementar tres tipos de salida debido al potencial valor negativo que la resta puede producir, en términos generales la salida tendrá la forma $z = x - y$. Que tendrá un valor entre $[-\Lambda+1, \Lambda-1]$, en particular los valores negativos no son representables de manera directa, pero haciendo algunas consideraciones será posible visualizarlos. El caso más simple para obtener el resultado es aplicar la función valor absoluto,

$$z_a = |x - y|. \qquad (5.5)$$

Este esquema no permitirá distinguir entre las diferencias negativas y positivas. Esta resta tiene la propiedad de ser conmutativa.

El segundo modelo se construye usando el ajuste de media o norma shift, su forma analítica es

$$z_s = (\Lambda - 1)/2 + (x - y)/2. \qquad (5.6)$$

Esta operación no es conmutativa y retiene información de la operación (x-y) y es posible distinguir el resultado con (y-x). El fondo es una clase de gris para imágenes en tonos de gris y produce tonos complementarios cuando las imágenes están en colores.

El tercer modelo corresponde a la llamada *resta a cero*, esta elimina los valores negativos, se puede considerar un ajuste por abajo. Su forma es

$$z_n = \begin{cases} x - y, x \geq y \\ 0, x < y \end{cases} \qquad (5.7)$$

Esta función ocasiona que los valores para los cuales la diferencia sea negativa los hace cero (negro).

En la figura siguiente se muestra el resultado de los tres modelos de resta (Se utilizan las figuras base de la Figura 5.3).

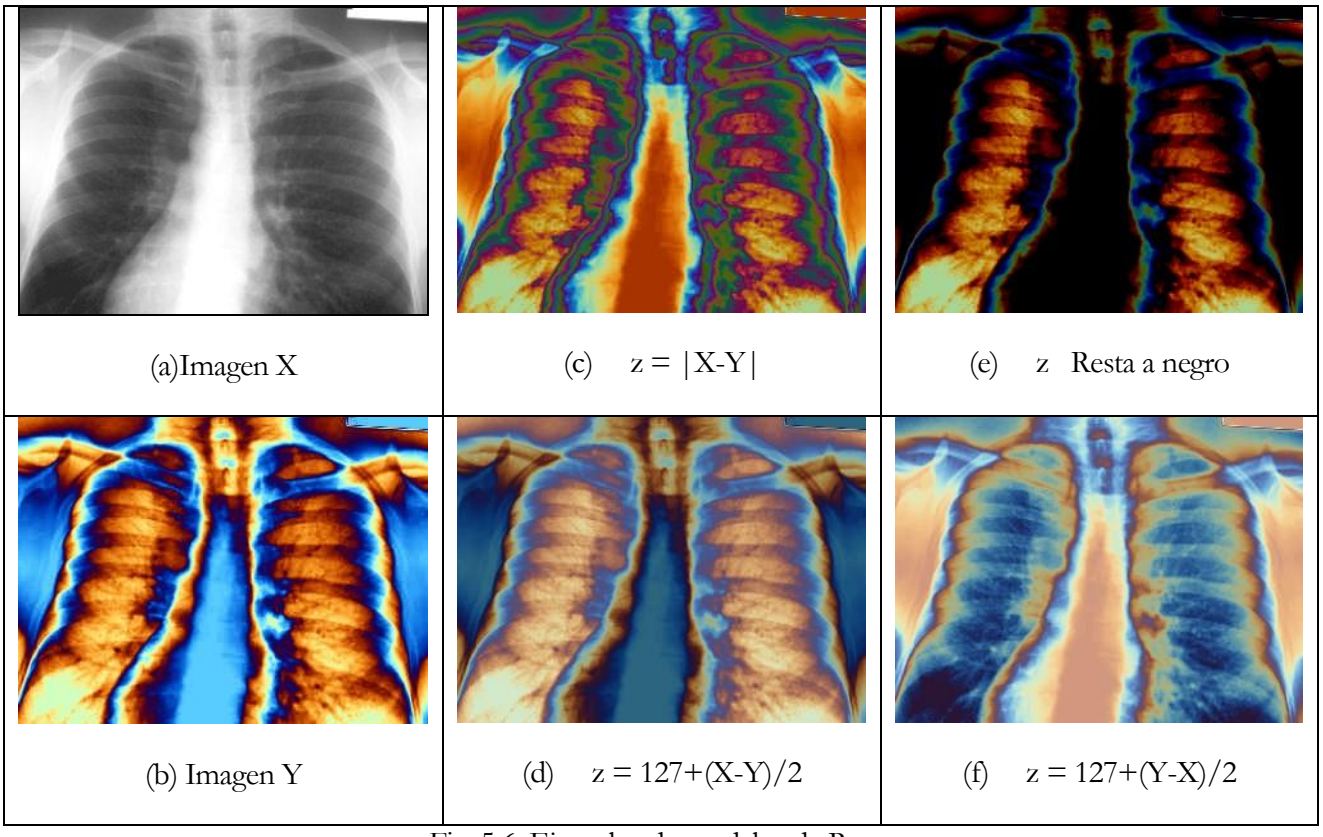
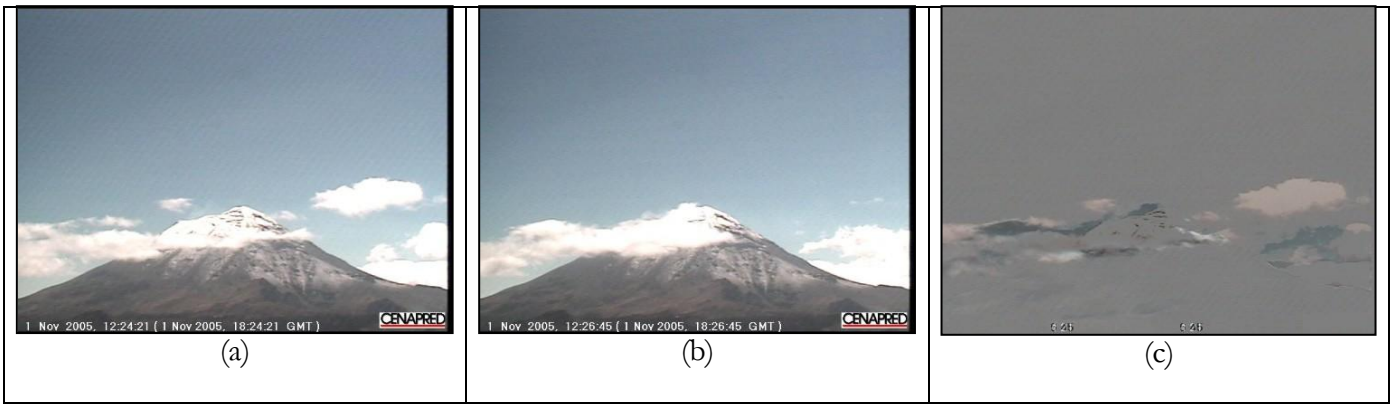


Fig. 5.6. Ejemplos de modelos de Resta

La imagen Y (Fig. 5.6) es una recreación de la X en Falso Color.

Cuando entre dos imágenes se trata de encontrar las diferencias, el método de resta en una buena opción, la siguiente pareja de imágenes corresponden a dos tomas de una cámara fija para vigilancia.



(a) (b) (c)

Fig. 5.7. (a)Imagen del Popocatepetl tomada desde una estación de vigilancia (CENAPRED), (b) Otra foto del volcán tomada 2 min y 24 seg. después, (c) Diferencia entre las imágenes(norma shift)

Puede observarse que la diferencia se pueden observar las nubes alrededor del volcán, también se puede notar parte de la estructura del volcán debido a la diferencia de iluminación en la escena.

5.1.3. Otras operaciones Aritméticas

Es posible implementar otras operaciones simples de tipo aritmético, entre otras están: la multiplicación y el cociente (división). La primera se puede construir mediante la regla:

$$z = k*x*y. \qquad (5.8)$$

donde k se debe ajustar a $1/(\Lambda\text{-}1)$ para evitar la saturación debido a que el dominio de x e y es $[0, \Lambda\text{-}1]$. Este proceso genera una imagen que será clara en las zonas donde los factores sean claros y obscura cuando alguno lo sea. En la siguiente figura se muestra el resultado del producto de dos imágenes.

Fig. 5.8. La imagen derecha es el resultado de multiplicar la primera y la segunda.

Puede notarse el efecto de fusión entre las imágenes (Fig. 5.8.) y además como el resultado es oscuro en general comparado con las imágenes originales. Es posible construir máscaras para realizar montajes, si hacemos una combinación de productos y sumas usando una imagen sintética con un círculo con un relleno de gradiente radial.

En la Figura siguiente se hace el montaje de un rostro y un fondo (pirámide de Kabah, Yucatán, Mex.).

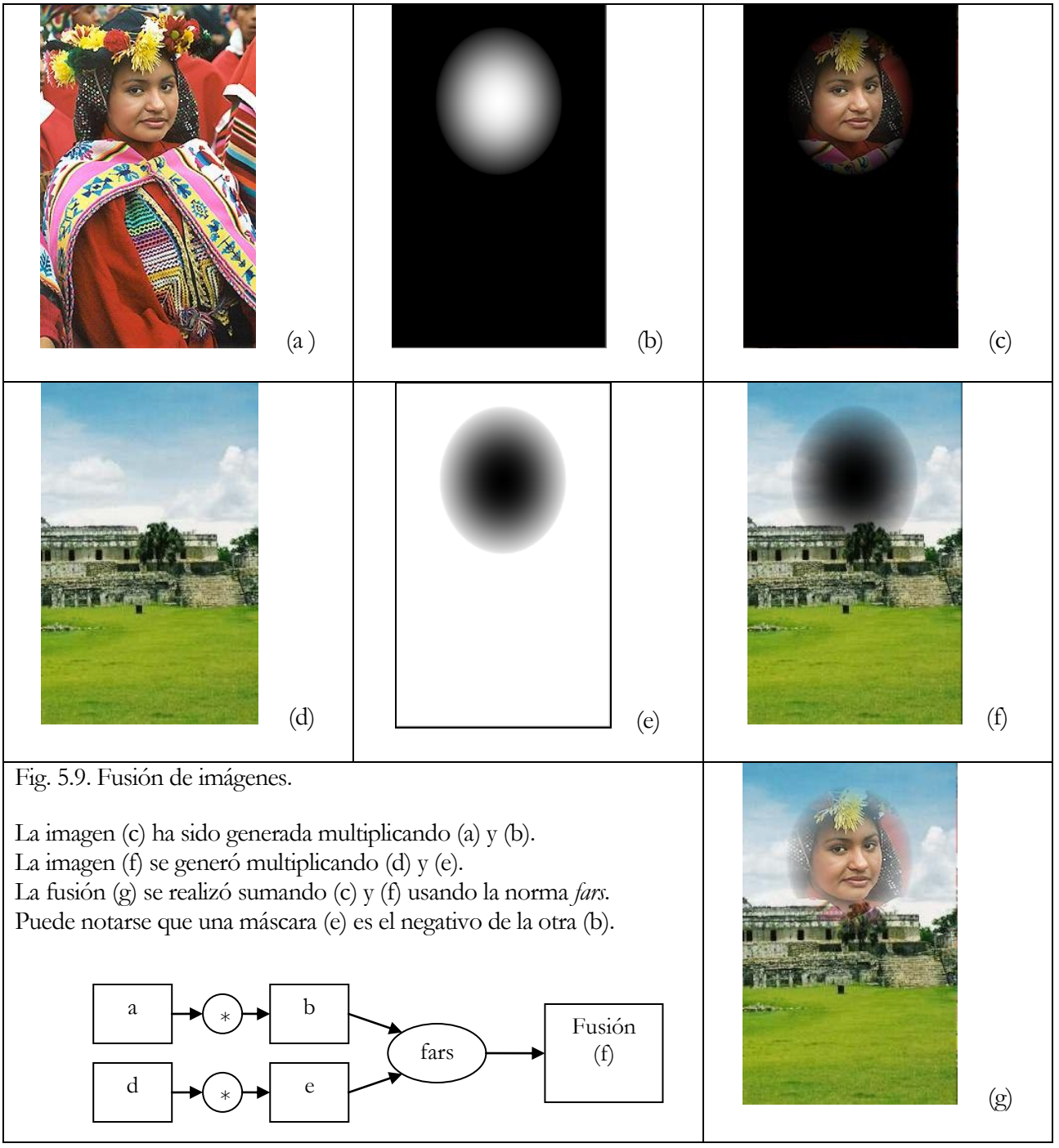


Fig. 5.9. Fusión de imágenes.

La imagen (c) ha sido generada multiplicando (a) y (b).
La imagen (f) se generó multiplicando (d) y (e).
La fusión (g) se realizó sumando (c) y (f) usando la norma *fars*.
Puede notarse que una máscara (e) es el negativo de la otra (b).

En general es posible construir otras operaciones entre imágenes, las cuales pueden utilizarse en problemas concretos. Entre otras están los cocientes, se define un par típico:

$$coc(x, y)_1 = x/y = (\Lambda\text{-}1) - ((\Lambda\text{-}1)*\min(x, y)) \text{ div } (\max(x, y)+1) \qquad (5.9a)$$

$$coc(x, y)_2 = k*\ln(1 + \max(x, y)/(\min(x, y)+1), \text{ donde } k = (\Lambda\text{-}1)/\ln(\Lambda) \qquad (5.9b)$$

La siguiente figura muestra el resultado de estas operaciones.

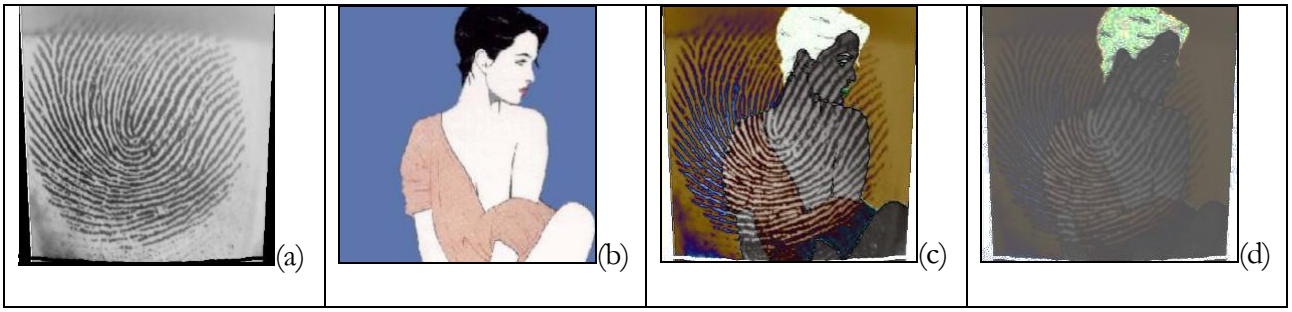


Fig. 5.10. La imagen (c) es el resultado de aplicar $coc(a, b)_1$ y (d)* corresponde a $coc(a, b)_2$.
* Se ha aplicado a (d) un aumento de brillo del 10% para apreciar el resultado.

5.1.4. Operaciones Lógicas y de relación.

Entre otras se pueden implementar una serie de operaciones binarias entre las imágenes, en la siguiente tabla se muestra la operación y el resultado de la misma con base a las imágenes de referencia de la Fig. 5.11.

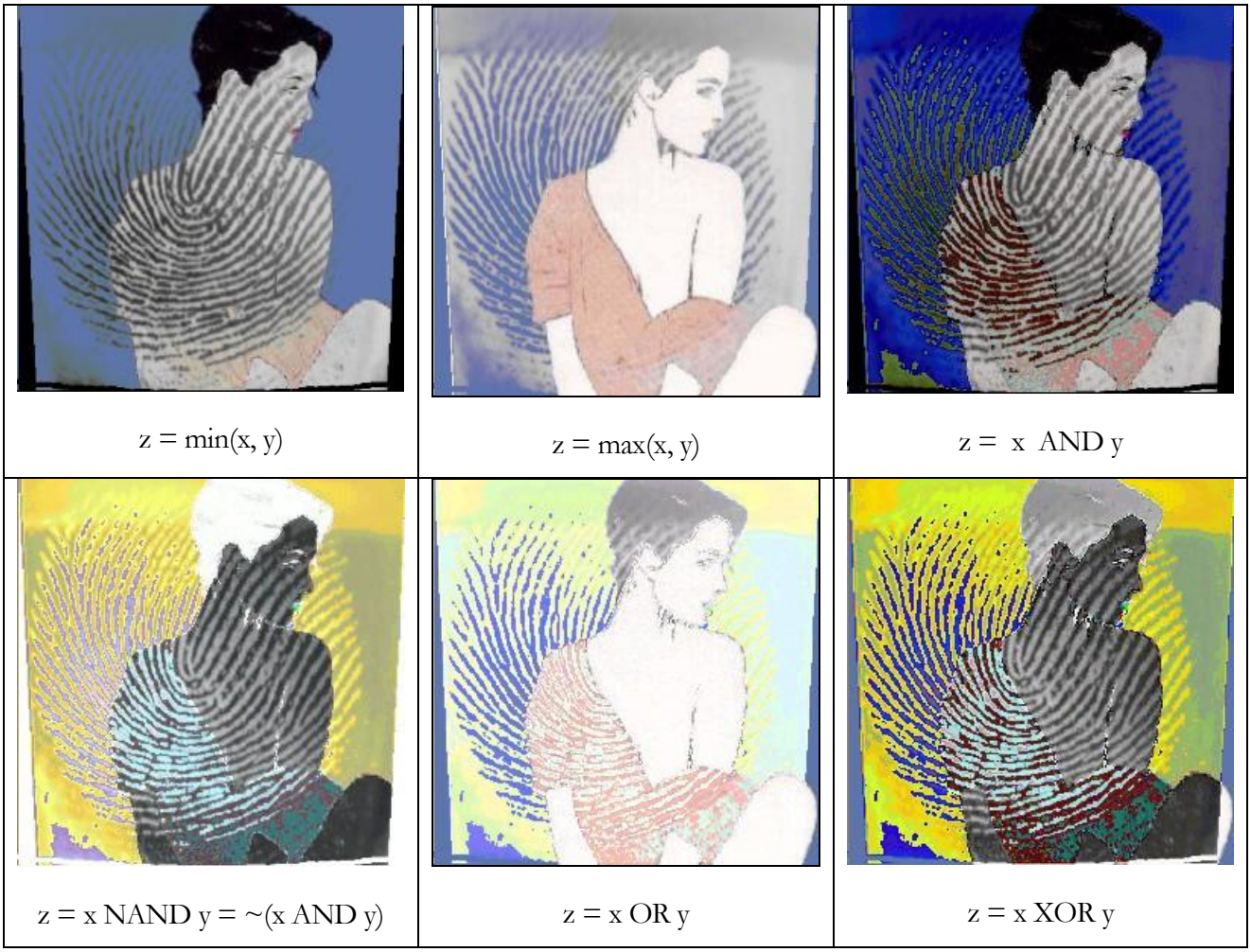
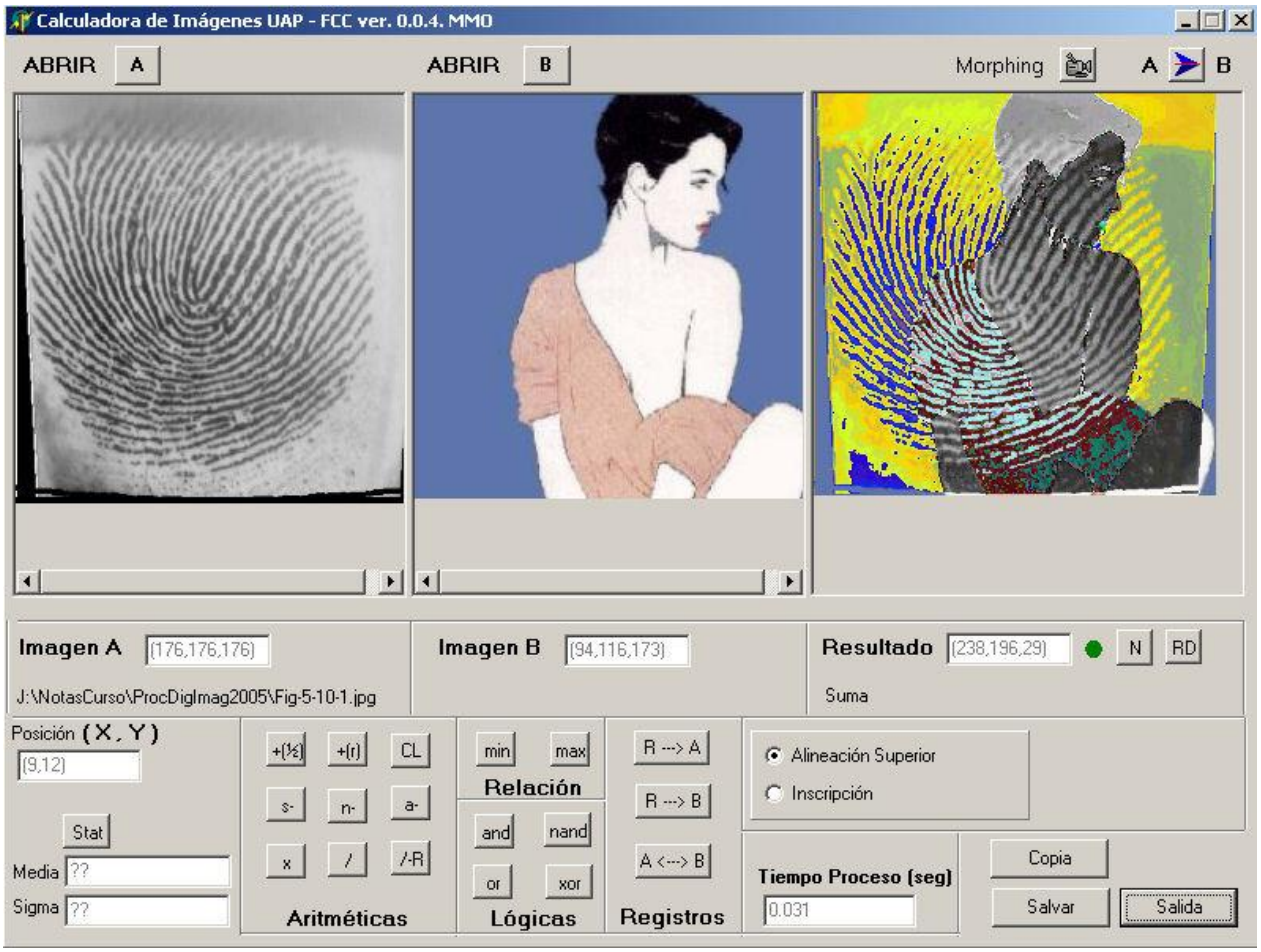


Fig. 5.11. Operaciones varias entre imágenes.

5.1.5. Una interfase para la Calculadora.

La siguiente interfase implementa una calculadora para imágenes.



Los paneles superiores permiten abrir las imágenes de trabajo (A) y (B). Las operaciones se encuentran organizadas en bloques en la parte inferior. Se ofrecen algunas operaciones complementarias como mandar el resultado (R) a la imagen (A) o bien a la (B), también se pueden intercambiar las imágenes (A) y (B) a solicitud del usuario con el propósito de trabajas con operaciones no conmutativas. Se implementan las opciones de inserción con "Alineación Superior" e "Inscripción". Los botones [N] y [RD] aplican el negativo y la función de rango dinámico al resultado.

Es posible ampliar las posibilidades de ésta herramienta de operación según los problemas planteados. En general algunas operaciones pueden automatizarse dentro de las aplicaciones en vez de utilizar una interfase interactiva.

Capítulo 6.

Transformaciones Especiales

En esta sección se presentarán una serie de transformaciones interesantes que no caen en los grupos antes expuestos o bien son derivados o combinaciones de ellos. En la primera sección se presentan procesos basados en el Histograma, los cuales permiten el aumento del contraste en un canal o en todos ellos.

6.1. Ecualización Simple.

Como se indica en el Capítulo I sección 1.4, es posible asociar a cada canal de una imagen una función que describe la distribución de los tonos llamada *Histograma*. Ésta se construye contando para cada tono el número de ocurrencias del mismo sobre una región o bien sobre toda la imagen, es posible asociar el histograma de cada canal a una *Función de Distribución de Probabilidad* (FDF) de hallar un tono específico. En el caso de una imagen obscura su histograma se encuentra sesgado hacia los tonos cercanos a cero. En la figura 6.1. se presenta un caso concreto del fenómeno.

Puede notarse que las distribuciones para cada canal no son iguales ya que en el histograma no se distingue la posición del píxel que contribuye a un tono en particular, en general podemos decir que el histograma deslocaliza el origen del píxel.

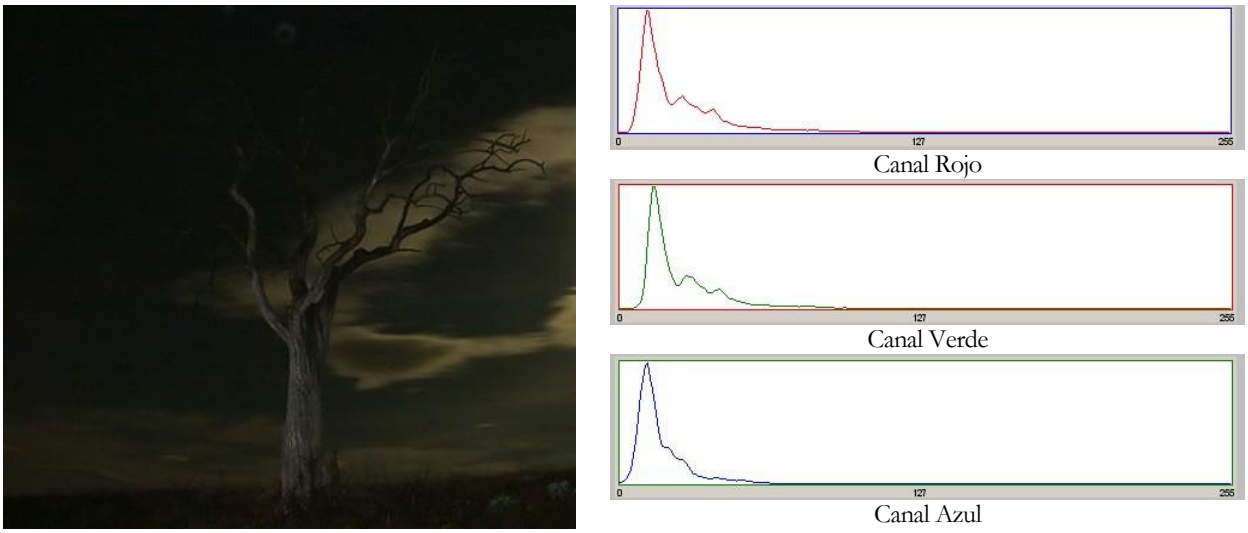


Fig. 6.1. Imagen Obscura y los histogramas para cada Canal.

Puede verse que los tres histogramas están definidos en la zona de los tonos obscuros ($t \rightarrow 0$) para cada canal.

El problema es construir una imagen tal que tenga un conjunto de histogramas mejor distribuidos a lo ancho de la variedad de tonos posibles. Es decir una imagen con mejor contraste. Un método simple se basa en la construcción de una función dependiente del mismo histograma, la idea es dispersar los valores de los tonos a lo ancho del rango posible de salida [0, Λ-1]. El procedimiento es el siguiente:

Asumiendo que *H[k, canal]* es la matriz que contiene los conteos por tono (*k*) y por *canal*, entonces se construye la función acumulativa para cada canal, a saber:

$$A[j, canal] = \sum_{k=0}^{j} H[k, canal] \,, \qquad (6.1)$$

con $j < \Lambda$-1.

En la siguiente ilustración se muestra un histograma y su función acumulativa.

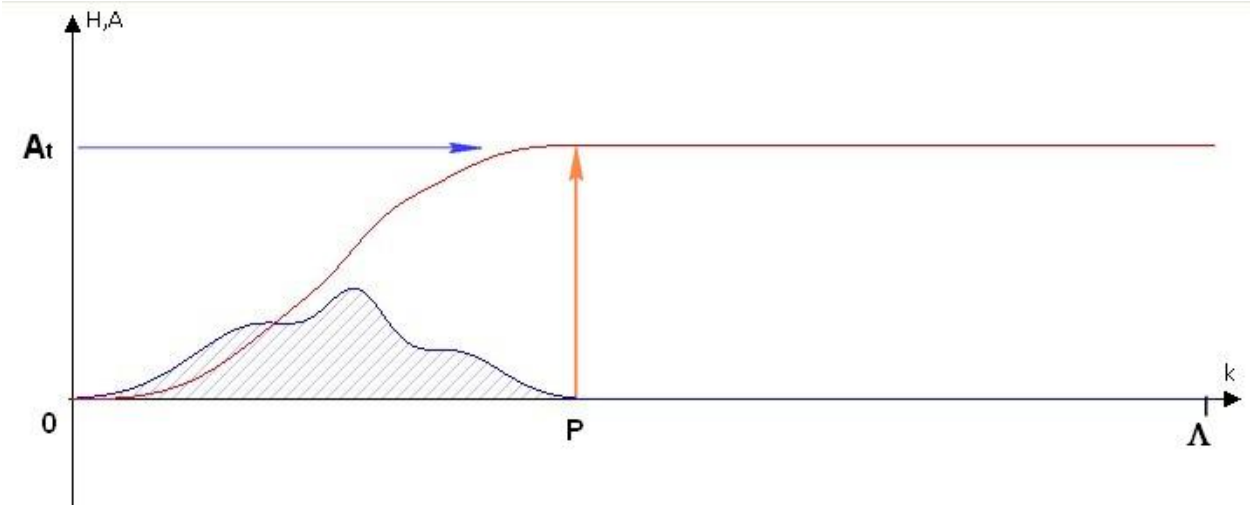


Fig. 6.2. Histograma H[k] y su correspondiente función acumulativa A[k]

Puede observarse que al caer H[k] en el punto **P** a cero deja de haber contribución a la función acumulativa y entonces ésta alcanza su valor máximo ($\mathbf{A}_t$) y se mantiene de forma asintótica en él.

El valor $\mathbf{A}_t$ corresponde obviamente a la suma total del histograma. En el caso de una imagen éste valor será el número total de pixeles de ella, ya que al hacerse el conteo de uno a uno todos son visitados. Es decir para el caso de una imagen,

$$A_t = n \times m, \qquad (6.2)$$

donde **n** corresponde al ancho de la imagen y **m** a su alto.
Si hacemos una normalización de la función acumulativa mediante el factor $(\Lambda\text{-}1)/nm$, entonces el rango de respuesta de la nueva función será [0, Λ-1]. El cual corresponde al intervalo de tonos representables en el dispositivo. Por lo tanto una función de la forma,

$$T[j, canal] = \frac{\Lambda - 1}{nm} \sum_{k=0}^{j} H[k, canal], \quad (6.3)$$

produce una función normalizada de tipo filtro corriente que aumenta el contraste de la imagen en un canal dado.
Dado que el rango de entrada de T[j] va de [0, Λ-1] y el de salida se ha ajustado a [0, Λ-1], entonces tenemos un filtro tradicional. Podemos notar que el intervalo [0, P] se mapea en el [0, Λ-1], de donde es un hecho el aumento en el contraste de la imagen. Al proceso de aplicar la transformación ***T[j, canal]*** se le denomina ecualización.

A continuación se muestran vistas de cada canal ecualizado y el histograma resultante.

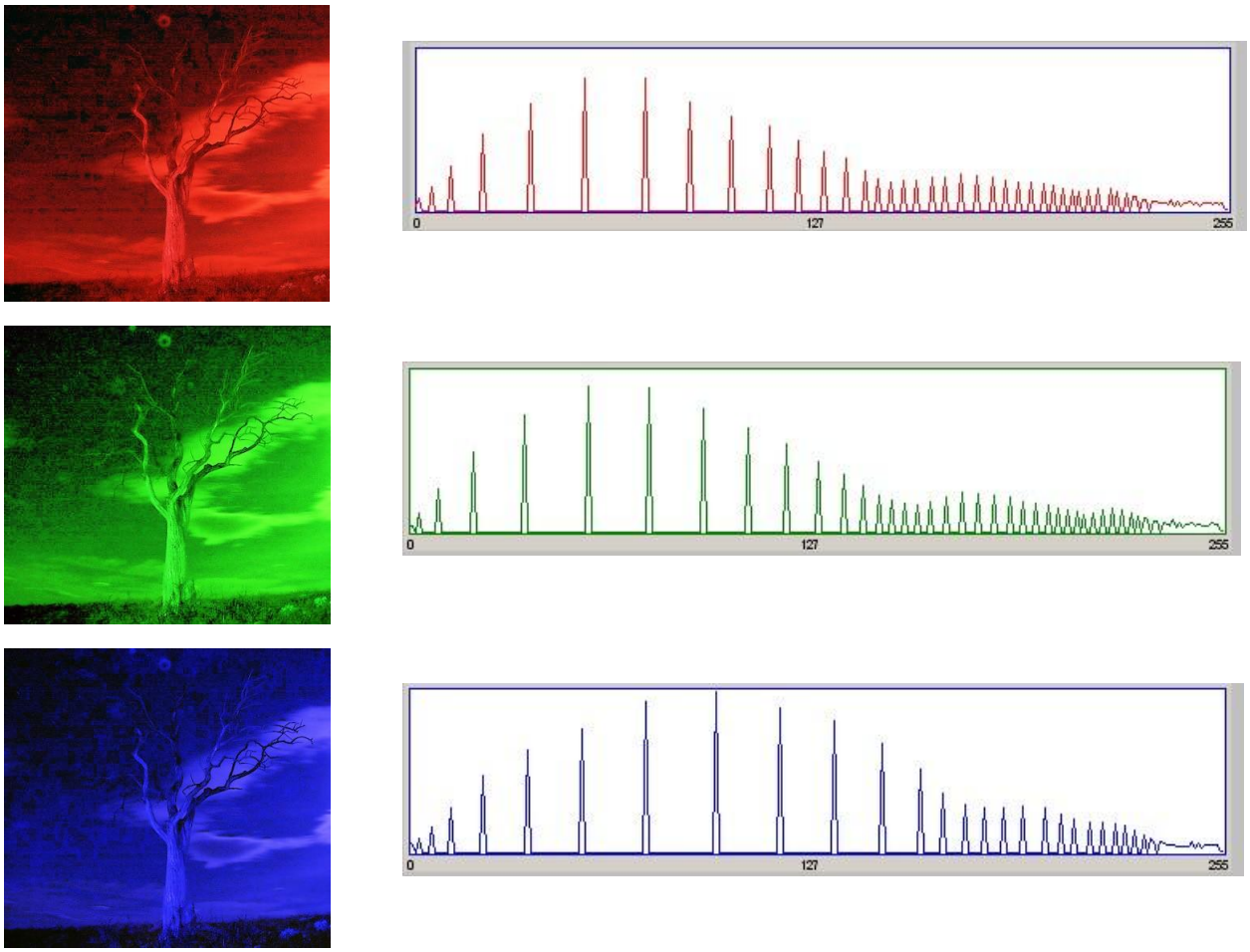


Fig. 6.3. Canales ecualizados y sus histogramas asociados

Puede notarse que los detalles se perciben mejor en el canal verde y que los histogramas ecualizados no son iguales. En la siguiente figura se muestra el traslape de ellos y el histograma de intensidades, así como la imagen conformada por los tres canales ecualizados.

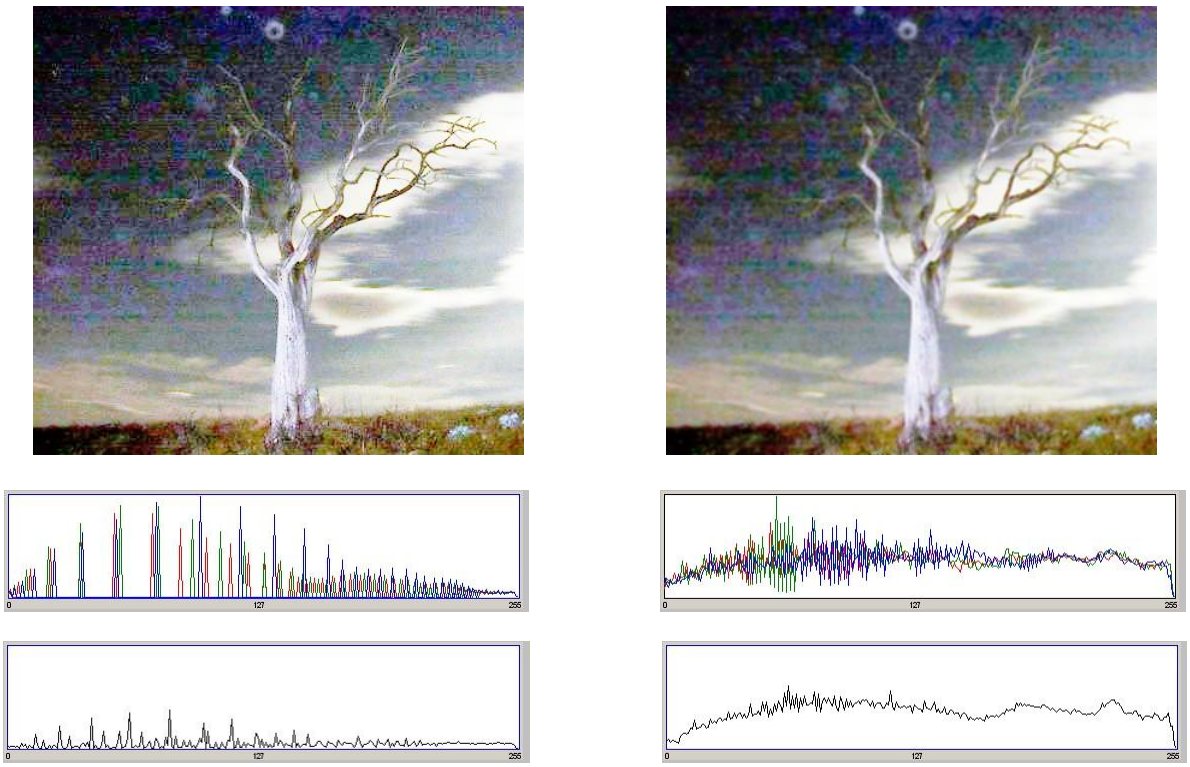


Fig. 6.4. Composición de imágenes ecualizadas (izquierda) y media de la imagen ecualizada (derecha) con sus histogramas RGB (arriba) e histograma de intensidades (abajo).

Pueden notarse los detalles sobre resaltados en la imagen compuesta y la reducción de los mismos luego de la aplicación de una media simple (máscara de 3x3). La calidad de la imagen respecto a la original (Fig. 6.1) es naturalmente mejor a pesar de los artefactos generados. Es clara la cromaticidad escondida en la misma luego del proceso de ecualización.

A continuación se muestra otro ejemplo sobre una imagen en tonos de gris y una versión aclarada.

Fig. 6.5. Imagen obscura y versión aclarada con sus histogramas

A continuación se presentan las imágenes de la Fig. 6.5. ecualizadas.

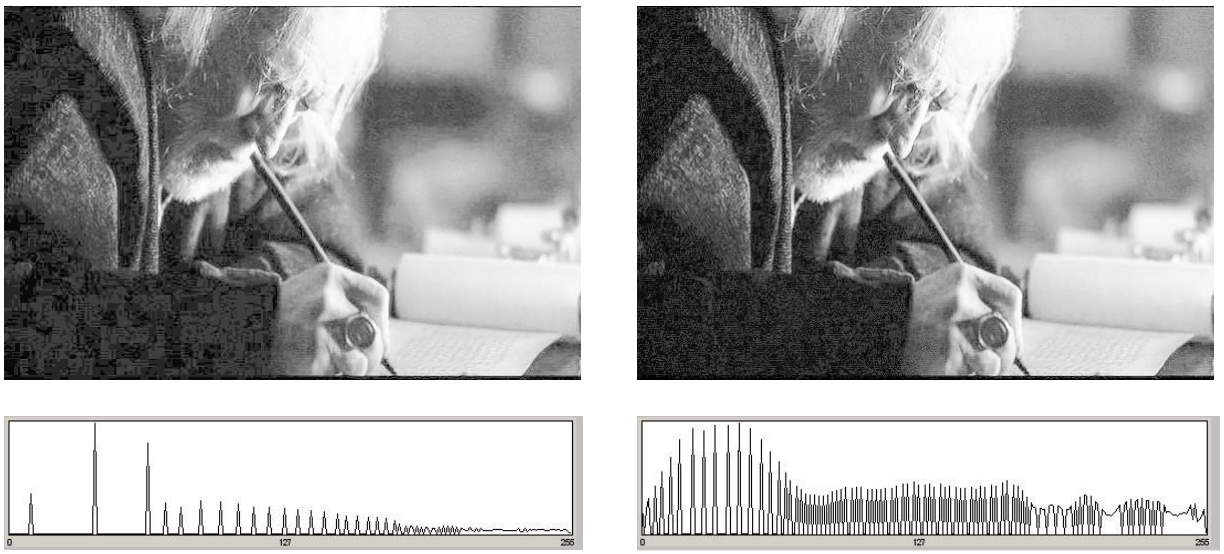


Fig. 6.5. Imagen obscura y aclarada ecualizadas con sus histogramas

Puede notarse que la gama de tonos es mayor para la imagen aclarada (derecha) ecualizada, respecto a la obscura (izquierda) ecualizada. Pueden apreciarse algunos artefactos en la manga del abrigo producidos por la ecualización.

A continuación se resume el algoritmo de ecualización para un canal.

Ecualización de un canal

```
Entrada  : M1 de n×m×c, [x1,y1]×[x2,y2] región a ecualizar, canal a ecualizar
Salida   : M2 de n×m×c con c=canal ecualizado

  base = x2-x1+1
  altu = y2-y1+1

  npix   = base*altu

  // genera el Histograma del canal elegido
  CuentaPobla(M1,x1,x2,y1,y2, H, canal)

  {Construir Arreglo con Función Acumulativa}
  A[canal,0] = H[canal,0]
  for (i = 1,255)
    A[canal,i] = A[canal,i-1] + H[canal,i]

  {Crear tabla de conversión}
  ff = num_niveles/npix;

  for (i = 0,num_niveles){
    indice = ff*A[i];
    j = round(indice)-1
    j = Max(j,0)
    Tabla[i] = j
  }
```

```
// Llena canales no afectados
for (k = 0,2 )
  if (k != canal) Copia_Matriz(n, m, M1, M2, k)

// procesa el canal elegido
for (j = y1, y2-1){
  for (i = x1, x2-1){
    {Procesando el Plano correspondiente}
    p = M1[i, j, canal]
    q = Tabla[p]

    {Componiendo el nuevo Pixel}
    M2[i, j, canal] = q;
  }
}
```

En general éste proceso se puede aplicar a cada uno de los canales de la imagen en colores para tener la imagen ecualizada completa. El resultado es una imagen de mayor contraste.

Índice del Capítulo

Capítulo 7.

Falso Color

Muchos sensores capturan imágenes en Tonos de Gris, éste es el caso de los Sistemas de Infrarrojos, Rayos X y la Termografía. Por otro lado en otros sistemas se transformas matrices de datos no ópticos en una imagen, casos de estos son: Ultrasonido, Cámaras-Z y la Resonancia Magnética Nuclear (RMN). Si

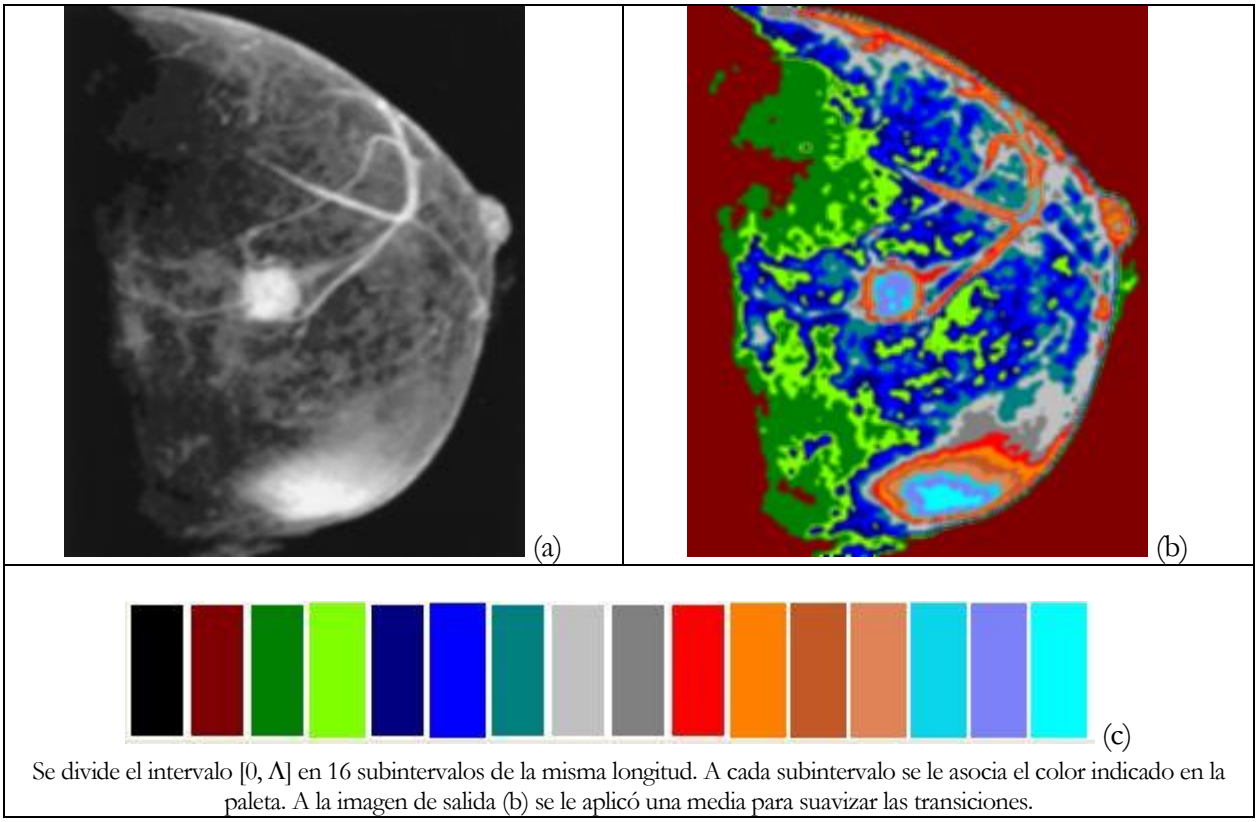
bien es posible analizar la información obtenida en escala de grises, se suele transformar las imágenes obtenidas en arreglos con color. A esta metodología se le llama *Falso Color* debido a que éste se impone mediante tablas construidas a conveniencia, mediante patrones de calibración o mediante funciones generadoras de transformaciones de grises a colores.

7.1. Tablas de Usuario

Esta es una de las técnicas más simples y de uso frecuente. La idea consiste en dividir el intervalo de tonos de gris a rango completo $[0, \Lambda]$ o un subrango en caso de que la imagen no cubra todo el espectro de grises $[\gamma_1, \gamma_2]$ en un conjunto de subintervalos $\{[g_i, g_{i+1}], i = 1, n-1\}$ donde g_1 corresponde al tono más pequeño a considerar $\{0 \text{ o } \gamma_1\}$ y g_n al tono más alto $\{\Lambda \text{ o } \gamma_2\}$ según el caso seleccionado. Donde a cada subintervalo se le asigna un color, $[g_i, g_{i+1}) \rightarrow C_i$. Posteriormente se recorre la imagen y a cada píxel según sea su tono de gris se sustituye por el color que le corresponde a dicho subintervalo, es decir

$$\text{si } T_{gris}(x, y) \in [g_i, g_{i+1}) \Rightarrow M[x][y] = C_i$$

A la secuencia de colores para cada subintervalo de tonos de gris se le denomina Paleta. En la Fig. 7.1 se muestra un ejemplo.



Se divide el intervalo [0, Λ] en 16 subintervalos de la misma longitud. A cada subintervalo se le asocia el color indicado en la paleta. A la imagen de salida (b) se le aplicó una media para suavizar las transiciones.

Fig. 7.1 (a) Imagen de una mastografía en tonos de gris. (b) Imagen en Falso color usando una paleta de 16 colores (c).

Este tipo de Falso color es muy común encontrarlo en la presentación de datos meteorológicos e imágenes de termografía. En la Fig. 7.2 se muestra una imagen satelital GOES del sitio www.noaa.org de la banda de Infrarrojo correspondiente al Mar Atlántico y su correspondiente imagen en falso color (se usó la misma paleta que en la Fig. 7.1).

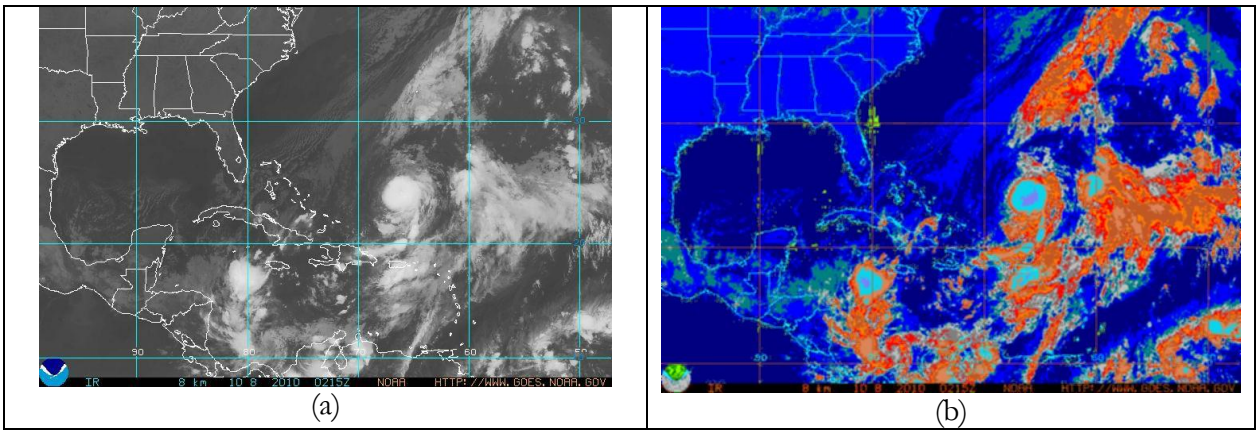


Fig. 7.2 (a) Imagen IR satelital (b) Correspondiente en Falso Color

Es posible crear paletas con intervalos irregulares y con mas subintervalos en general. En el caso de contarse con algún referente o patrón los colores se pueden asociar conforme a éste, en la Fig. 7.3 se muestran unas termografías obtenidas con una cámara basada en microbolómetros (estos permiten mediar la temperatura píxel a píxel con gran precisión). En la parte derecha se muestra la paleta calibrada utilizada.

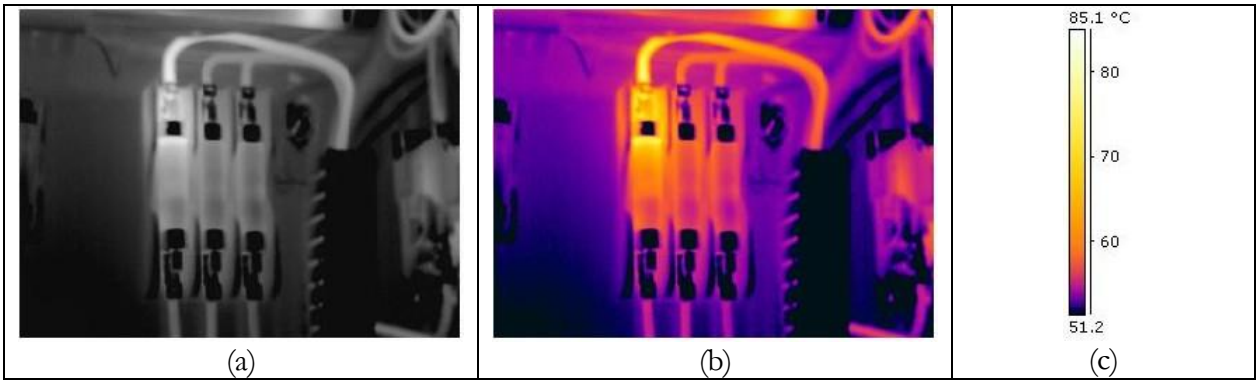


Fig 7.3. (a) Termografía. (b) Su imagen en Falso Color. (c) Paleta Calibrada.

7.4. Cámara FLIR serie-i

En este caso debido a que el dispositivo regresa los valores numéricos de la temperatura con al menos décimas de grado centígrado es posible asociar una paleta asociada a las temperaturas del objeto registrado. Los tonos obscuros corresponden a las partes frías y los claros a las partes calientes, es notorio que en la versión en falso color es más sencillo apreciar las distribuciones y gradientes de temperatura en el sistema eléctrico bajo observación. Para este tipo de problemas existen cámaras tipo pistola que tienen una pantalla LCD en la parte posterior y de forma manual se puede escoger o definir una paleta e inspeccionar in situ los sistemas eléctricos e industriales para controlar su desempeño. El caso de la termografía es singular, ya que de forma directa no es posible cuantificar la temperatura de los objetos mediante los sentidos de los operadores o inspectores. En la Fig. 7.4 se muestra una cámara de ésta clase. Además estos dispositivos tienen como es de esperarse una memoria donde se guardan las imágenes y luego pueden ser transferidas a aplicaciones de software para ser estudiadas y crear reportes (www.flir.com).

Las paletas de usuario pueden ser almacenadas y recargadas como archivos para ser utilizadas sobre otras imágenes y poder comparar las vistas en falso color.

7.2. Paletas Simples

Una forma de construir paletas para un intervalo dado es la explotación de los gradientes de colores. La idea se basa en tomar dos o mas colores y construir una secuencia, por ejemplo lineal, de colores mediante reglas de interpolación en el espacio de colores.

Paleta de dos colores.

Supongamos que se eligen dos colores C_1 y C_2 en el espacio RGB y diferentes; y los queremos mapear al intervalo $[\gamma_1, \gamma_2]$ en tonos de gris de manera lineal. Primero debemos recordar que un color en éste espacio es una terna (r, g, b), entonces podemos utilizar la ecuación paramétrica de la recta en 3 dimensiones. Sea **z** uno de los canales, entonces $z_i = z(C_i)$ corresponderá a la componente cromática z del color C_i . Usando la ecuación de la recta en la manera mencionada, entonces las parejas que forman la recta para el canal z serán (γ_1 , z_1) y (γ_2 , z_2), por tanto para un tono arbitrario $\gamma \in [\gamma_1, \gamma_2]$ tendremos que

$$z = z_1 + (\gamma - \gamma_1)\frac{z_2 - z_1}{\gamma_2 - \gamma_1}, \tag{7.1}$$

Donde para $\boldsymbol{\gamma = \gamma_1 \Rightarrow z = z_1}$; y para $\boldsymbol{\gamma = \gamma_2 \Rightarrow z = z_2}$ **.** En el caso que se haga un barrido completo del intervalo $\boldsymbol{\gamma \in [0, \Lambda]}$ $(\boldsymbol{\gamma_1 = 0, \gamma_2 = \Lambda})$, la fórmula (7.1) se simplifica como

$$z = z_1 + \alpha(z_2 - z_1), \tag{7.2}$$

Donde α correrá de 0 a 1, ya que $\alpha = \gamma/\Lambda$.

En la figura siguiente (Fig. 7.5) se muestra una paleta de ésta clase y los colores base que la generan.

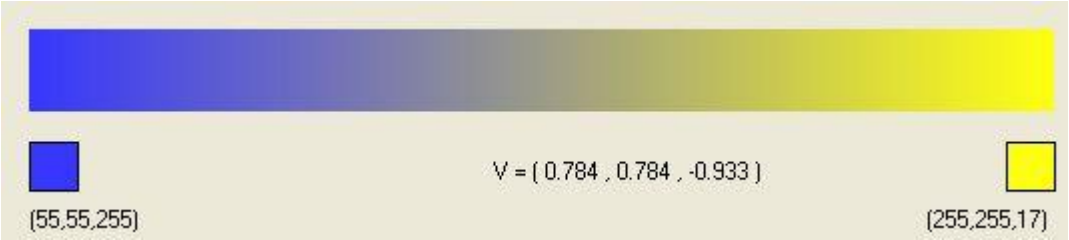


Fig. 7.5. Paleta de dos colores a rango completo [0, Λ], en los cuadros inferiores se muestran los colores base.

De manera similar se puede generar una paleta con tres colores con algunas salvedades. Sean C_1, C_2 y C_3 los colores generadores en el orden dado, entonces para cubrir el rango completo [0, Λ] debemos escalar cada segmento en el espacio cromático C_1C_2 y C_2C_3 de tal forma que la suma de las magnitudes de los segmentos corresponda a Λ. Sean d_1 y d_2 las distancias en el espacio cromático entre C_1 y C_2; y C_2 y C_3 correspondientemente. Entonces

$$d_1 = \sqrt{(r_2 - r_1)^2 + (g_2 - g_1)^2 + (b_2 - b_1)^2}$$

$$d_2 = \sqrt{(r_3 - r_2)^2 + (g_3 - g_2)^2 + (b_3 - b_2)^2}$$

Definiendo $L = d_1 + d_2$, entonces $d_1 /L + d_2/L = 1$, multiplicando por Λ ambos miembros resulta que

$$\Lambda = \Lambda\, d_1 /L + \Lambda\, d_2 /L,$$

entonces los recorridos para cada subintervalo serán:

$$n_1 = \frac{\Lambda d_1}{L}$$

$$n_2 = \frac{\Lambda d_2}{L} = \Lambda - n_1$$

Esto implica que el primer segmento $\mathbf{C_1C_2}$ se debe dividir en n_1 subintervalos y el segundo $\mathbf{C_2C_3}$ en n_2 subintervalos. La siguiente figura (Fig. 7.6) muestra una paleta de 3 colores y su correspondencia con los tonos de gris a 8 bits.

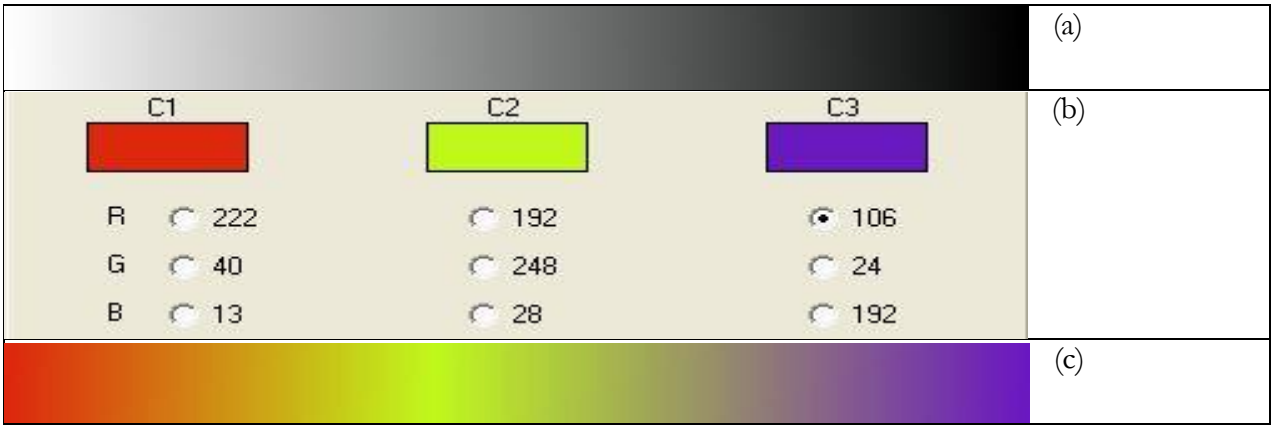


Fig. 7.6. (a) Gradiente de Grises. (b) Colores Base. (c) Paleta de Falso color lineal de tres colores.

En la siguiente imagen (Fig. 7.7) se realiza la aplicación de una paleta de tres colores a una imagen obtenida mediante un telescopio, esta corresponde a la Nebulosa de burbuja NGC7635 (www.noao.edu).

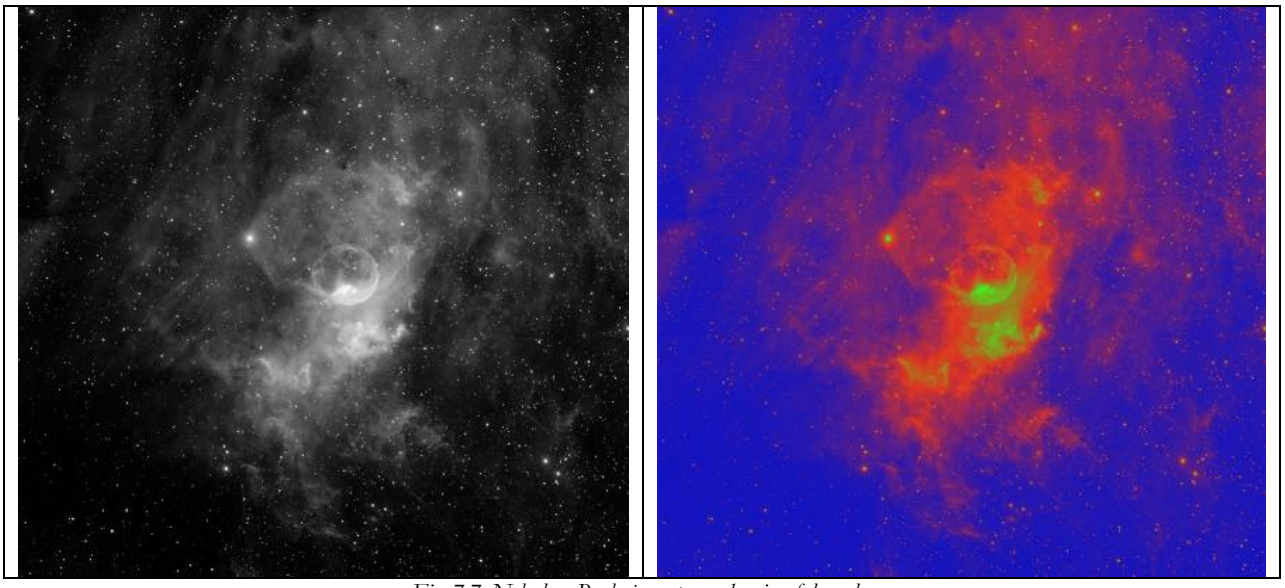


Fig. 7.7. Nebulosa Burbuja en tonos de gris y falso color

Un método que produce imágenes en falso color para el caso de los satélites meteorológicos consiste en combinar las imágenes de los canales que son recibidos y asignar una combinación lineal de estos respecto a las salidas RGB de forma directa o mediante diferencias de canales.

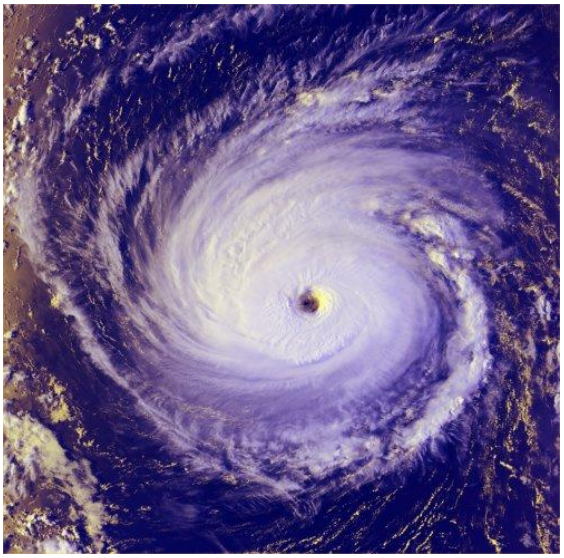


Fig.7.8. Imagen Satelital del Huracán Isaacs

La imagen izquierda (Fig. 7.8) corresponde al Huracán Isaacs en la coordenada (25.6N 53.5W) obtenida del satélite NOAA-14 el 14 de septiembre del año 2000. Estos satélites envían 5 canales de información con las siguientes características:

Canal	Banda (μm)	Uso y análisis de
1	0.58 - 0.68	Nubes y tierra
2	0.725 - 1.00	Frontera tierra-agua
3a	1.58 - 1.64	Detección de hielo y nieve
3b	3.55 - 3.93	Nubes nocturnas y Temperatura del mar
4	10.30 - 11.30	Nubes nocturnas y Temperatura del mar
5	11.50 - 12.50	Temperatura del mar

Se puede encontrar más información en el sitio http://noaasis.noaa.gov/NOAASIS/ml/avhrr.html .

La imagen de falso color se genera mediante una combinación del tipo:

$$\text{Color} = \alpha_1 \text{Can}[u] + \alpha_2 \text{Can}[v] + \alpha_3 \text{Can}[w],$$

o bien

$$\text{Color} = \alpha_1 (\text{Can}[u_1]-\text{Can}[u_2]) + \alpha_2 (\text{Can}[v_1]-\text{Can}[v_2]) + \alpha_3 (\text{Can}[w_1]-\text{Can}[w_2]).$$

En particular en los sitios de Prospección Remota se pueden encontrar diversas combinaciones que permiten resaltar diferentes aspectos de los fenómenos meteorológicos, características de los suelos y formaciones geológicas.

7.3. Paletas Automáticas y Flexibles

Otro mecanismo para la generación de paletas de Falso Color es mediante funciones generadoras. La idea es construir una tercia de funciones tales que a partir del tono de gris de cada píxel se generen los tres canales convencionales, es decir

$$r = f_1(z)$$
$$g = f_2(z)$$
$$b = f_3(z)$$

Donde z es el tono de gris de la imagen original. Si las funciones f_i se parametrizan es posible crear paletas dinámicas, las cuales se pueden almacenar explícitamente o bien almacenar las funciones y sus parámetros para una ulterior aplicación o modificación.

Triple función rampa - escalón

Existen diversas propuestas para las funciones f_i , una de las más simples se hace mediante funciones definidas por pedazos. En el siguiente gráfico (Fig. 7.9) se ilustra un esquema de generación de paleta, cada función se representa por su color.

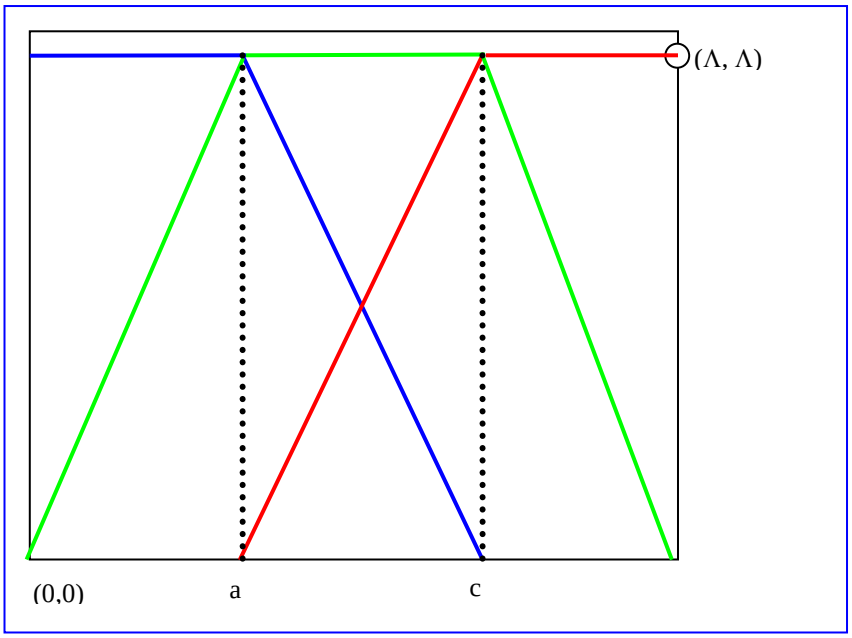


Fig. 7.9. Perfiles de funciones cromáticas r, g y b para funciones rampa

Así entonces las funciones para cada canal están dadas por:

$$r = f_1(z) = \begin{cases} 0 & 0 \le z < a \\ \dfrac{\Lambda}{c-a}(z-a) & a \le z \le c \\ \Lambda & c < z \le \Lambda \end{cases}$$

$$g = f_2(z) = \begin{cases} \dfrac{\Lambda}{a} z & 0 \le z < a \\ \Lambda & a \le z \le c \\ \dfrac{\Lambda}{\Lambda - c}(\Lambda - z) & c < z \le \Lambda \end{cases}$$

Y finalmente

$$b = f_3(z) = \begin{cases} \Lambda & 0 \le z < a \\ \dfrac{\Lambda}{c-a}(c-z) & a < z \le c \\ 0 & c < z \le \Lambda \end{cases}$$

Para este modelo un caso particular interesante es cuando reducimos a un solo parámetro el modelo

$$a = \Lambda - k,$$
$$c = \Lambda + k.$$

En este caso la diferencia (c-a) = 2k; y Λ-c = -k. Y se debe tener cuidado en el caso que k=0, ya que las ecuaciones toman una forma diferente ya que las transformaciones para r y c se convierten en funciones escalón.

Triple seno positivo con corrimiento de fase

Un caso interesante es la utilización de una tercia de funciones seno (o coseno) no negativas con corrimiento de fase. Este modelo es utilizado en la detección de objetos observados bajo scanners de rayos-X en sistemas de seguridad y puntos de control. Las ecuaciones de transformación de la imagen en tonos de gris son las siguientes:

$$r(z) = A_1 \,|\sin(kz + \phi_1)|$$
$$g(z) = A_2 \,|\sin(kz + \phi_2)|$$
$$b(z) = A_2 \,|\sin(kz + \phi_3)|$$

Donde los coeficientes A_i modelan las amplitudes de las componentes cromáticas y los factores ϕ_i los desfases de cada función, en las figuras siguientes (7.10a - 7.10d) se muestran las funciones, un ejemplo de paleta y la aplicación de ella a una imagen en tonos de gris.

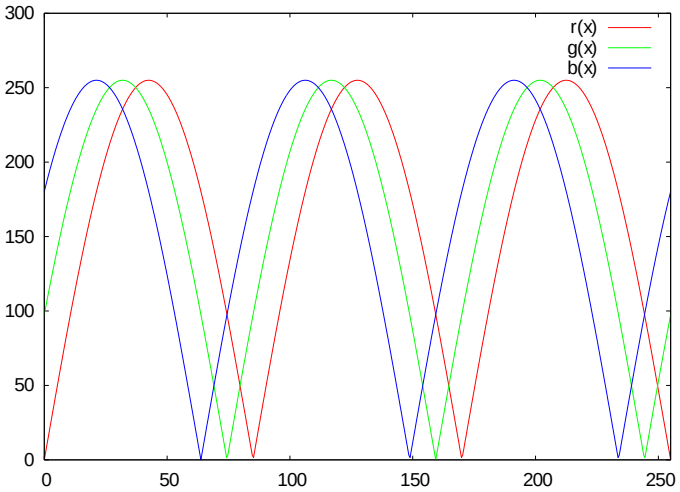


Fig. 7.10a. Funciones con $A_i = A$, $\phi_1=0$, $\phi_2=\pi/8$, $\phi_3= =\pi/4$; y $k=3\pi/A$

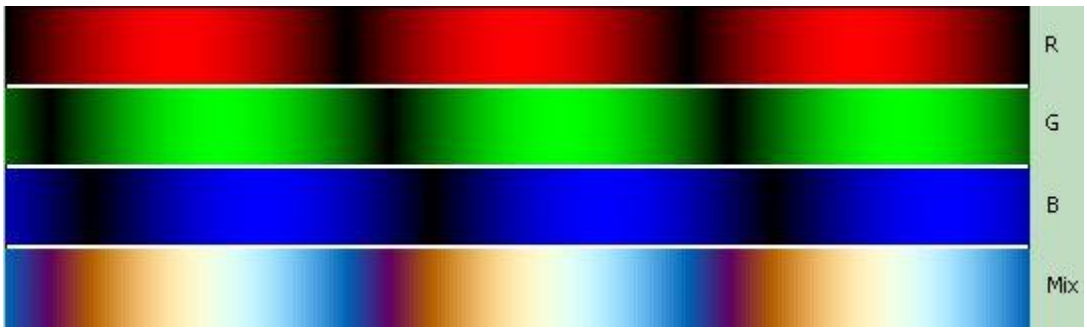


Fig. 7.10b. Composiciones cromáticas (RGB) por canal y paleta mezclada (renglón inferior)

En las figuras siguientes se muestra una imagen termográfica y la aplicación de paleta anterior

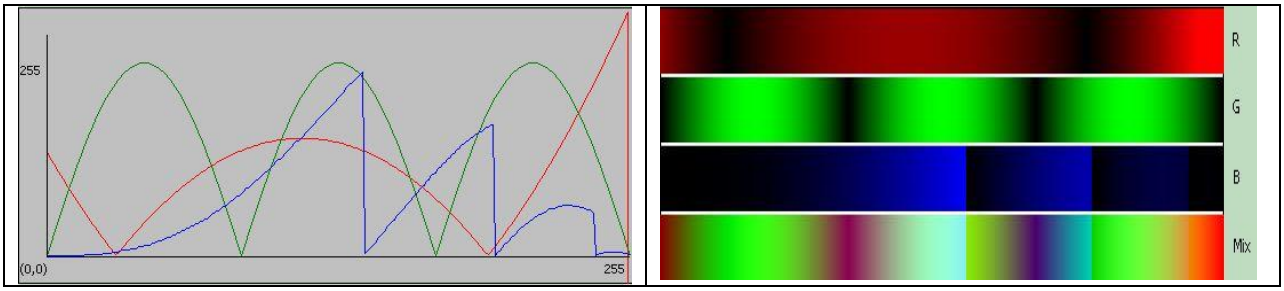
(c) (d)

Fig. 7.10c y 7.10d. Imagen en tonos de gris(izquierda) e imagen en colores en base a la aplicación de la paleta dada en la Fig. 7.10a y 7.10b

Puede notarse en la figura 7.10d el detalle de la escena resaltado por el falso color.

Algunas paletas interesantes

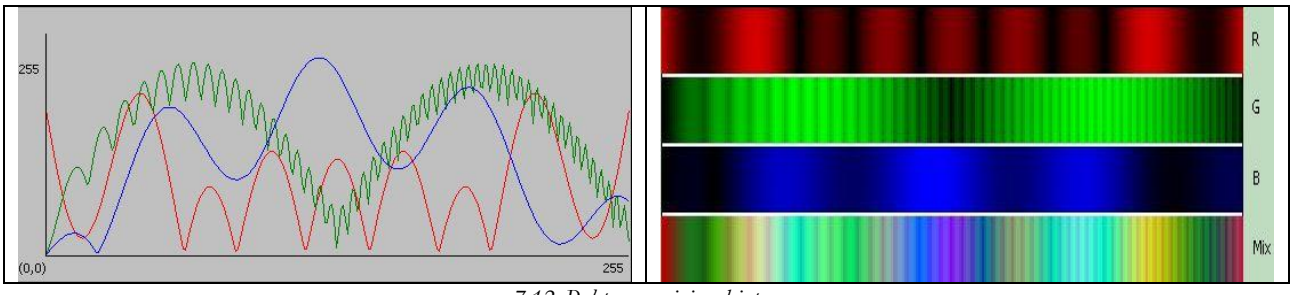
A continuación se muestra una serie de paletas y las funciones que las generan, éstas han sido creadas por un programa que recibe las funciones r(z), g(z) y b(z) y produce la paleta.



7.11. Paleta con funciones módulo

Las funciones generadoras son:
r(z) = 3*abs(Λ-(2*z-Λ/3)*(z-2Λ/3))/ Λ
g(z) = Λ*abs(sin(3*π*z/Λ))
b(z) = 4*((z*z)/ Λ mod 80)*z*(Λ-z)/21000

La siguiente paleta contiene minigradientes que hacen la función de marcas de región.

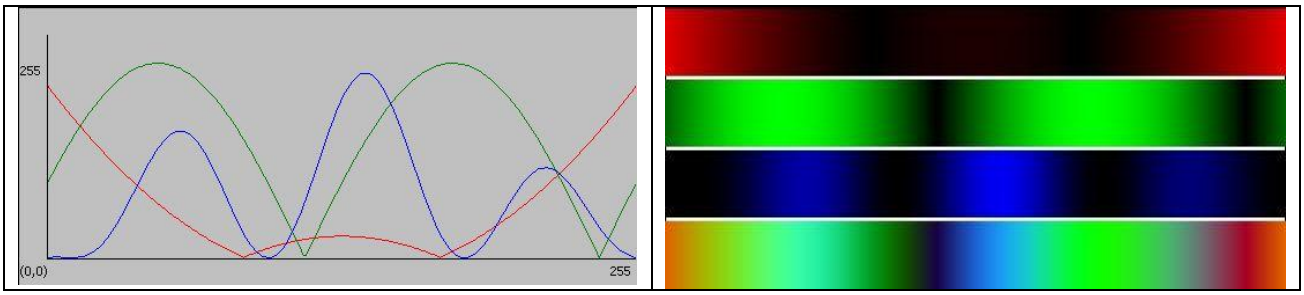


7.12. Paleta con minigradeintes

Las funciones generadoras son:
r(z) = abs (3*(z-Λ/2)*(z-Λ/2)/ Λ - Λ*sin(9*π*z/Λ)/2)
g(z) = 200*abs(sin(2*π*z/Λ)) + 55*abs(sin(3.2*π*z*sqrt(z)/ Λ))

b(z) = abs(3*z*(Λ-z)/ Λ - 70*sin(7.5*π*z/Λ))

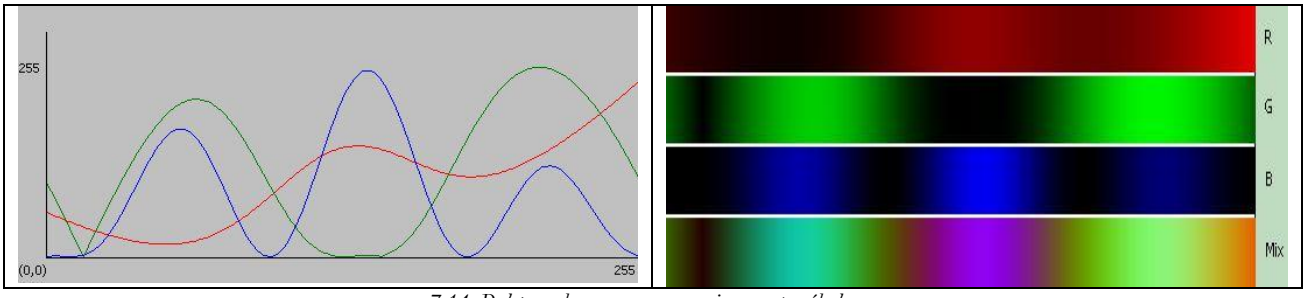
Esta paleta esta generada mediante la combinación de funciones seno y parábolas



7.13. Paleta en base a senos y parábolas

Las funciones generadoras son:
r(z) = 4*abs((z-Λ/3)*(z-2*Λ/3)/ Λ)
g(z) = Λ*abs(sin(2*π*z/Λ+ π/8))
b(z) = 3*sqr(sin(3*π*z/Λ-π/8))*z*(Λ-z)/200

La siguiente paleta incorpora funciones exponenciales.



7.14. Paleta en base a senos, gaussianas y parábolas

Las funciones generadoras son para este caso:
r(z) = Λ*(exp(-sqr(z-Λ/2)/2000) + sqr(1.5*z-Λ/2)/36000)/2
g(z) = Λ*(1-exp(-sqr(z-Λ/2)/2000))*abs(sin(2*π*z/Λ-π/8))
b(z) = 6*sqr(sin(3*π*z/Λ-pi/8))*z*(Λ-z)/400

Bibliografía

[1] González R., Woods R., Digital Image Processing, Prentice Hall, 2004.
[2] Low A., Introductory Computer Vision and Image Processing, McGraww – Hill, 1991.
[3] Parker J. R., Algorithms for Image Processing and Computer Vision, John Wiley & Sons, 1997.
[4] Murray J. D., VanRyper W., The Encyclopedia of Graphics File Formats 2nd Edition, O'Reilly & Associates, CDROM edition, 1996.
[5] Lyon D., Image Processing in Java, Prentice Hall, 1999.
[6] Myler H., Weeks M., The Pocket Handbook of Image Processing Algorithms in C, Prentice Hall, 1993
[7] Russ J. C., The image processing Handbook, CRC Press, 2002
[8] Pajares G., de la Cruz J., Visión por computador – imágenes digitales y aplicaciones, Alfaomega –Rama, 2002.
[9] Krpatsch W., Bischof H., Digital Image Analysis, Springer, 2001
[10] Awcock G.J., Thomas R., Applied Image Processing, McGraw-Hill, 1996
[11] Jain R., Kasturi R., Schunck B., Machine Vision, McGraw-Hill, 1995
[12] de la Escalera A., Visión por computador – Fundamentos y Métodos, Prentice Hall, 2001.

Apéndice 1

Transformaciones en Espacios Alternos

Una *imagen digital* se puede definir como una como una colección de *puntos* definidos en una región **C**, las regiones pueden en general tener cualquier forma y existe una función **I(x, y, z)**, donde alguno de los parámetros es función continua de los otros dos, que nos permite recorrer los puntos que componen la colección. Digamos que **z = h(x, y),** entonces una representación de la imagen digital es,

$$I(x, y, z) = I(x, y, h(x, y)) = J(x, y), \quad (1)$$

donde las propiedades de cada punto o *pixel* que conforma la imagen están contenidas en **J(x, y)**. Estas consideran por ejemplo: el color, el brillo, la reflectividad, etc.

Se define una transformación **T** de la imagen **J(x,y)** como una función tal que,

$$T_I[\, J(x, y)] = J'(x, y), \quad (2)$$

donde $\Gamma \subset \mathbf{C}$, es decir Γ esta contenida en **C** lo que implica que actúa sobre una parte o toda la imagen.

Los efectos de **T** sobre **J** pueden ser diversos, entre otros podemos citar,

- modificación del dominio **C**,
- cambio en una o varias de sus propiedades,
- cambio en la posición de los puntos,
- cambio en la representación de **J.**

Si **T** actúa sobre un punto diremos que la transformación es *puntual*, si actúa sobre una región será *zonal* y si actúa sobre toda la imagen se dice que es *global*.

Diremos que **T** es una *transformación isomórfica* o *mapeo isomórfico* de un pixel (uno a uno) si cada pixel de la imagen de origen nos genera un nuevo pixel en otro *espacio*.

Muchas de éstas transformaciones se pueden definir a través de la relación

$$F(u, v) = < g(x, y) \mid J(x, y) >, \tag{3}$$

donde **g(x, y)** se denomina el *núcleo* o *kernel* de la transformación, *J(x, y)* es un punto de la imagen y < | > define el mecanismo de transformación.

Una clase bastante grande de éstas transformaciones se realiza mediante un *producto escalar*, si la magnitud de *f(x, y)* tiene magnitud uno la transformación se denomina *unitaria*.
Algunas formas usuales del producto escalar están definidas por una integrales definidas o una sumatorias y el kernel de la transformación es típicamente una base ortogonal y puede ser utilizada para la reconstrucción de la imagen original. Si el producto escalar de dos funciones es cero, se dice que éstas son ortogonales.

A.1. Transformada de Fourier

En una dimensión se define la transformada de Fourier como,

$$F(u) = \langle f(x) | e^{-i2\pi ux} \rangle = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx \tag{4}$$

Aquí el kernel se define por las funciones complejas $e^{-i2\pi ux}$, que forman una base ortogonal para las funciones en una dimensión. La versión discreta de la transformada de Fourier es,

$$F(u) = \langle f(x) | e^{-i2\pi ux} \rangle = \sum_{x=x1}^{x=x2} f(x) e^{-i2\pi ux} \Delta x, \tag{5}$$

si ahora consideramos una colección discretizada de datos, los índices serán enteros, por lo cual $x_1=0$ y $x_2=N-1$, donde N es el número de puntos se realiza la transformación. La partición del intervalo (x_1, x_2) es uniforme y se tendrá que $\Delta x = (x_2 - x_1)/N$, considerando que el paso mínimo entre dos pixeles es uno, se puede escribir (5) de la siguiente manera

$$F(u) = \langle f(x) | e^{-i2\pi ux} \rangle = \frac{1}{N} \sum_{x=0}^{x=N-1} f(x) e^{-i2\pi ux}, \tag{6}$$

así dada una imagen definida en una región rectangular *[0 ... N-1, 0 ... M-1]*, la Transformada Discreta de Fourier (DFT) de está dada por

$$F(u,v) = \langle f(x,y) | e^{-2i\pi(ux/N+vy/M)} \rangle , \tag{7}$$

o bien

$$F(u,v) = \frac{1}{NM} \sum_{x=0}^{x=N-1} \sum_{y=0}^{y=M-1} f(x,y) e^{-2i\pi(ux/N+vy/M)} . \tag{8}$$

La *Transformada Inversa de Fourier* se calcula mediante la expresión

$$f(x,y) = \sum_{x=0}^{x=N-1} \sum_{y=0}^{y=M-1} F(u,v) e^{2i\pi(ux/N+vy/M)} . \tag{9}$$

Puede verse que se puede implementar un algoritmo directo para calcular la DFT, cabe aclarar que para cada pixel de una imagen se debe aplicar la relación (8), haciendo una evaluación simple de la complejidad del método, tendremos que para cada punto *(x, y)* se deberán realizar *NM* operaciones y como la imagen esta formada por *NM* elementos, entonces

$$O(M,N) = O(NM\ NM) = O(N^2 M^2) . \tag{10}$$

Así por ejemplo para una imagen de 500 x 500 pixeles, el costo será de $(500)^2$ x $(500)^2$ operaciones, que resulta $6.25\ 10^{10}$ operaciones. En una máquina que realice 1 Mflop (es decir un millón de operaciones de punto flotante por segundo), el tiempo de ejecución del cálculo asumiendo una operación por iteración será de

$$T_{ejec}(N=500,M=500) \cong (\text{Núm. Ops.})/(\text{Velocidad}) = 6.25\ 10^{10} / 10^6 \text{ seg}$$
$$= 6.25\ 10^4 \text{ seg} = 6.25\ 10^4 /(3600) \text{ Hrs} = 17.361 \text{ Hrs}$$

Es claro que éste es un método de alto costo y se han propuesto una gran cantidad de variantes que aceleren el proceso de cómputo [1].

Las mejores alcanzan un rendimiento dado por $O(MN \log_2 N \log_2 M)$, usando algunas restricciones para los valores del tamaño de la imagen , en particular se construye bajo la hipótesis de que *N* y *M* tengan la forma 2^k, donde *k* es un entero no negativo. Así para una imagen con *N=M=512*, que es un poco mayor que la del ejemplo anterior, se tendrá un costo para el tiempo de ejecución para la misma máquina (1 Mflop)de

$$T_{ejec}(N=M=512) \cong (\text{Núm. Ops.})/(\text{Velocidad}) = (512)^2 (\log_2 512)^2 / 10^6 \text{ seg.}$$
$$= 2.62144\ 10^5 \quad 81/10^6 \text{ seg.} = 21.23 \text{ seg.}$$

En términos generales la relación de tiempos de ejecución entre la DFT y la FFT para *M=N* es

$$\tau = (N^2 (\log_2 N^2))/(N^4) = (\log_2 N / N)^2 . \tag{11}$$

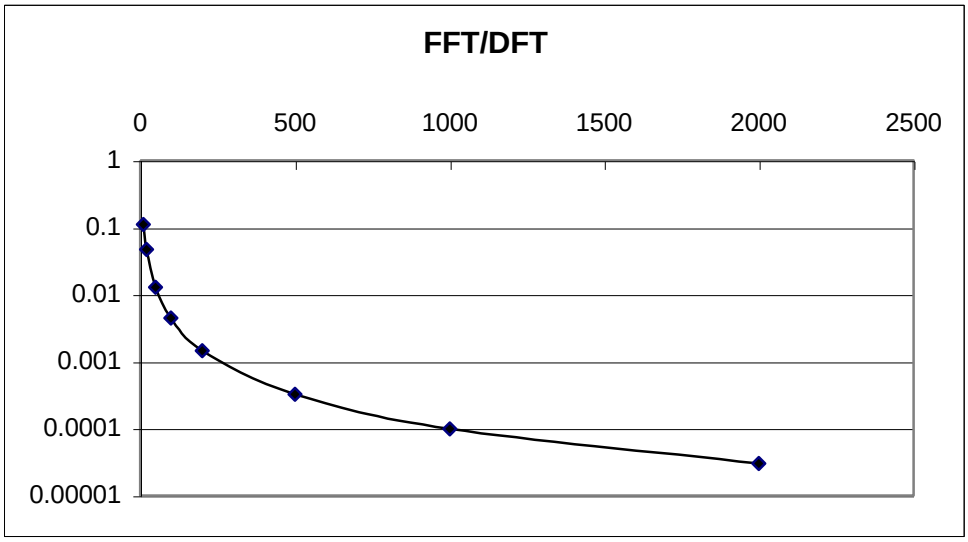


Fig. Relación de velocidad entre la FFT y la DFT cruda.

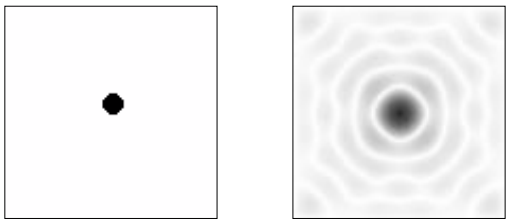
A estos métodos se les llama *Transformadas Rápidas de Fourier.* Quienes construyeron los primeros algoritmos para mejorar de éste cómputo de manera automática fueron J.W. Cooley y J.W. Tukey a mediados de 1960 apoyados por IBM, el trabajo se realizó en el Yorktown Heights Research Center. Una de la implementaciones es la conocida como *Radix-2 FFT*, ésta se basa en una estrategia de *divide y vencerás*, en términos generales lo que se hace es dividir de manera recursiva el problema de tamaño *N* hasta que resulten solo dos muestras, la DFT se calcula en todos los conjuntos de dos muestras y luego se realiza la recombinación. La división de la secuencia se basa en el lema de *Danielson – Lanczos* (1942), el cual establece que la muestra se puede dividir en sus componentes con índice par e impar, los subconjuntos se vuelven a dividir hasta que resultan solo dos elementos (criterio de parada), hay versiones recursivas[2] y no recursivas[3]. Las versiones no recursivas dan una velocidad 6x superior a las versiones recursivas, el costo de implementación de las versiones no recursivas obviamente se ve incrementado.

Se puede consultar una versión no recursiva de *Radix-2 /Cooley y Tukey FFT* en la clase *vsFFT1D* [4]. O bien la rutina ***fourn*** [1] (sección 12.11 FFT en dos a mas dimensiones) que se basa en el trabajo de N.M. Brenner de los "Lincon Laboratories".

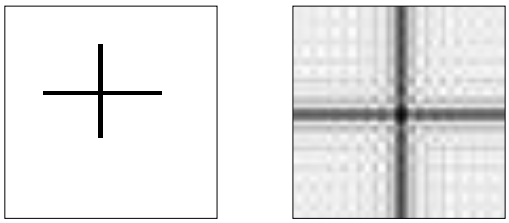
Una variante interesante es la que se basa en la transformada de Hartley respecto al espectro de potencia de la transformada de Fourier [5] (ver cap. 13).

A continuación se presentan algunos ejemplos de transformadas de Fourier de imágenes simples.

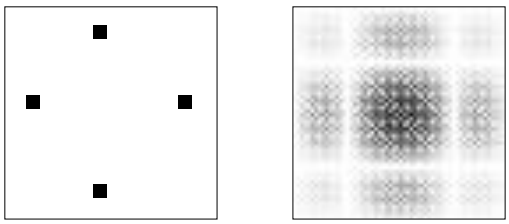
Imagen Original	Transformada de la Imagen



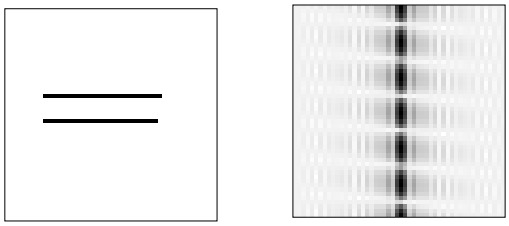
Círculo



Cruz



Cuatro Puntos



Dos líneas horizontales

Las figuras se generaron con el programa Fourier (*hhtp://www.cs.buap.mx/~mmartin*) que se realizó para el Taller de PDI en verano de 2000. Las imágenes originales tienen se construyeron en blanco y negro y tienen 60 x 60 pixeles de tamaño. Se presenta el negativo de las imágenes.

BIBLIOGRAFÍA DEL APÉNDICE 1.

[1] *Press, Flannery, Teukolsky & Vetterling*. **Numerical Recipies in C** (The Art of Scientific Computing), Cambridge University Press, 1990. – Ver Capítulo 12 – *Fourier* Transform, Spectral Methods. Existen versiones en Fortran y Pascal también.
[2] *Wolberg*, **Digital Image Warping**, IEEE Press, 1990.
[3] *Lyon & Rao*, **Java Digital Signal Proccesing**, M&T Press. (1997) <http:// www.DocJava.com>
[4] *Lyon*, **Image Processing in Java**, Prentice Hall (1999).
[5] *Low*, **Introductory Computer Vision and Image Processing**, McGraw Hill, (1991).