Fundamentos de Procesamiento de Imágenes

I. Fundamentos de procesamiento de imágenes digitales

I.1 Definiciones

Visión por computadora.- Consiste en la adquisición, procesamiento, clasificación y reconocimiento de imágenes digitales.

Píxel.- Elemento básico de una imagen (*picture element*).

Imagen.- Arreglo bidimensional de píxeles con diferente intensidad luminosa (escala de gris).

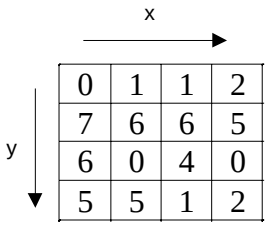


Figura 1. Imagen de 16 píxeles

Si la intensidad luminosa de cada píxel se representa por n bits, entonces existirán 2^n escalas de gris diferentes.

Matemáticamente, una imagen se representa por $r = f(x, y)$, donde r es la intensidad luminosa del píxel cuyas coordenadas son *(x,y)*. Matemáticamente, un sistema para procesar imágenes se representa como $g(x, y) = T[f(x, y)]$.

Color.- El color se forma mediante la combinación de los tres colores básicos rojo, azul y verde (en inglés RGB). A continuación se presentan algunas definiciones básicas para comprender los espacios de color:

Brillo.- Indica si un área está más o menos iluminada.
Tono.- Indica si un área parece similar al rojo, amarillo, verde o azul o a una proporción de ellos.
Luminosidad.- Brillo de una zona respecto a otra zona blanca en la imagen.
Croma.- Indica la coloración de un área respecto al brillo de un blanco de referencia.

Para obtener una imagen a color deben transformarse primero los parámetros cromáticos en eléctricos y representar los colores, lo cual puede realizarse de diferentes maneras, dando lugar a diferentes espacios de colores o mapas de color.

Espacio RGB.- se basa en la combinación de tres señales de luminancia cromática distinta: rojo, verde, azul (*Red, Green, Blue*). La forma más sencilla de obtener un color específico es determinar la cantidad de color rojo, verde y azul que se requiere combinar

para obtener el color deseado, ver la figura 2; para lo cual se realiza la suma aritmética de las componentes: X = R + G + B, gráficamente representada por un cubo.

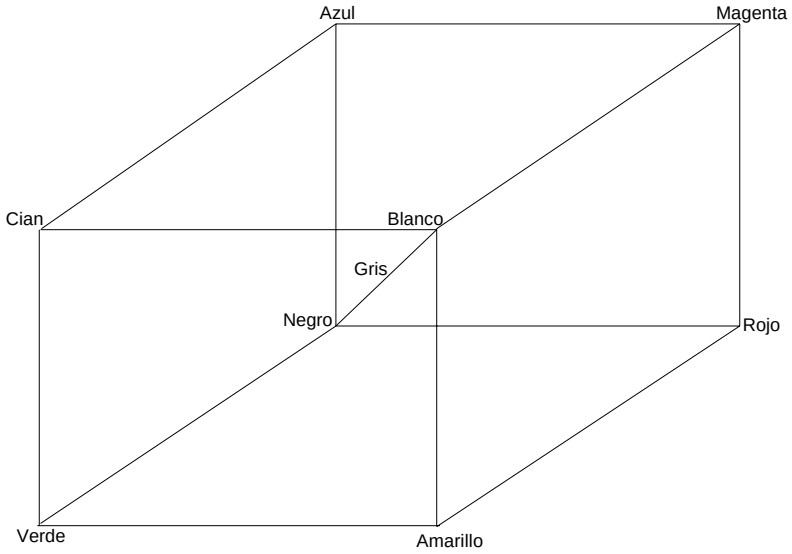


Figura 2. Espacio de colores RGB.

En la recta que une el origen con el valor máximo se encuentran ubicados los grises (escala de gris) debido a que sus tres componentes son iguales. Cuando una cámara adquiere una imagen a color, para cada píxel en color se tienen en realidad 3 componentes, una para cada uno de los colores básicos (rojo, verde y azul); la ganancia máxima para cada componente corresponde a la longitud de onda de los tres colores básicos.

Color

Un color puede definirse como la combinación de tres colores básicos: rojo, verde y azul, y expresarse mediante una tripleta de valores de 0 a 1 (R, G, B), donde R, G y B representan las intensidades de cada uno de los tres colores básicos rojo, verde y azul, respectivamente. En la tabla I se presentan ejemplos de colores definidos mediante estas tripletas.

Tabla I. Colores RGB

Color	R	G	B
Blanco	1	1	1
Rojo	1	0	0
Amarillo	1	1	0
Verde	0	1	0
Turquesa	0	1	1
Gris	0.5	0.5	0.5
Rojo Oscuro	0.5	0	0
Azul	0	0	1
Aguamarina	0.5	1	0.83
Negro	0	0	0

Mapa de color
El mapa de color es una matriz de n x 3, donde cada renglón es una tripleta de colores. El primer renglón corresponde al valor mínimo del eje de color y el último renglón al máximo. Al definir diferentes distribuciones de intensidad de los tres colores básicos, se crean diferentes mapas de color. Algunos de los mapas de color predeterminados en MATLAB son:

```
hsv, cool, hot, jet, gray, flag
```

Histograma de una imagen.
El histograma de una imagen es una representación del número de píxeles de cierto nivel de gris en función de los niveles de gris.

I.2 Relaciones entre píxeles

Un píxel *p* con coordenadas *(x,y)* tiene cuatro vecinos horizontales y verticales, cuyas coordenadas son: *(x+1,y), (x-1,y), (x,y-1), (x,y+1).* A este conjunto de píxeles se llama vecindad 4 o 4 vecinos de *p* y se denota por $N_4(p)$, ver la figura 3. Nótese que para cada uno de estos píxeles hay una distancia de 1 de p y que en los bordes de la imagen algunos de estos píxeles quedarán fuera de la imagen.

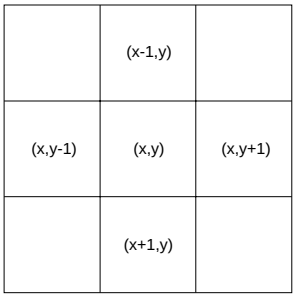


Figura 3. Vecindad $N_4(p)$.

Existen también 4 vecinos diagonales de *p* con coordenadas: *(x+1,y+1), (x+1,y-1), (x-1,y-1), (x-1,y-1)* y se les denota por $N_D(p)$, ver la figura 4. $N_4(p)$ y $N_D(p)$ juntos forman la vecindad 8 denotada por $N_8(p)$.

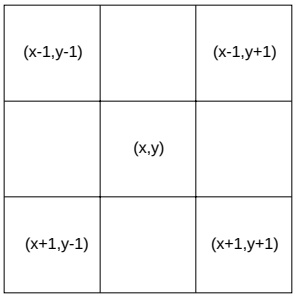


Figura 4. Vecindad $N_D(p)$.

I.2.1 Conectividad

La conectividad es un concepto importante utilizado para establecer los límites de objetos en regiones dentro de una imagen. Para determinar si dos píxeles están conectados se determina si son adyacentes en algún sentido ($N_D(p)$, $N_4(p)$ por ejemplo) y si sus niveles de gris satisfacen un criterio de similaridad (por ejemplo si son iguales). Por ejemplo, en una imagen binaria con valores de 1 y 0, dos píxeles pueden ser vecinos $N_4(p)$, pero se dice que están conectados solo cuando tienen el mismo valor.

I.2.2 Distancia

La distancia o transformada de distancia proporciona una medición de la separación existente entre dos puntos dentro de una imagen. Dados tres píxeles, *p*, *q* y *z*, con coordenadas (*x*,*y*), (*s*,*t*) y (*u*,*v*), respectivamente, se puede definir una función de distancia D si se cumple:

$$D(p,q) \geq 0, (D(p,q) = 0, si\ p = q)$$
$$D(p,q) = D(q,p)$$
$$D(p,z) \leq D(p,q) + D(q,z)$$

Las funciones de distancia comúnmente usadas son: distancia euclidiana, distancia Manhattan o de cuadra y distancia tablero de ajedrez.

Distancia euclidiana entre p y q: $D_E(p,q) = \sqrt{(x-s)^2 + (y-t)}$. En la figura 5 se muestra la distancia euclidiana para una imagen de 5 por 5.

$$\begin{array}{ccccc} \sqrt{8} & \sqrt{5} & 2 & \sqrt{5} & \sqrt{8} \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} & \sqrt{5} \\ 2 & 1 & 0 & 1 & 2 \\ \sqrt{5} & \sqrt{2} & 1 & \sqrt{2} & \sqrt{5} \\ \sqrt{8} & \sqrt{5} & 2 & \sqrt{5} & \sqrt{8} \end{array}$$

Figura 5. Distancia euclidiana para una imagen de 5 por 5.

Distancia Manhattan: se toman solamente en cuenta los vecinos de orden 4, es decir:

$$D = |x-s| + |y-t|$$

En la figura 6 se muestra la distancia Manhattan de una imagen de 5 por 5.

$$\begin{array}{ccccc} 4 & 3 & 2 & 3 & 4 \\ 3 & 2 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 2 & 3 \\ 4 & 3 & 2 & 3 & 4 \end{array}$$

Figura 6. Distancia Manhattan para una imagen de 5 por 5.

Distancia tablero de ajedrez: es similar a la distancia Manhattan, en donde se observa que los 4-vecinos están a una distancia unitaria del píxel central; si se desea que los 8-vecinos estén a la misma distancia se toma:

$$D(p,q) = Max(x-s, y-t)$$

En la figura 7 se muestra la distancia tablero de ajedrez.

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Figura 7. Distancia tablero de ajedrez.

I.3 Ruido en imágenes

Todas las imágenes tienen cierta cantidad de ruido, la cual se puede deber a la cámara o al medio de transmisión de la señal. Generalmente el ruido se manifiesta como píxeles aislados que toman un nivel de gris diferente al de sus vecinos. Los algoritmos de filtrado que se verán más adelante permiten eliminar o disminuir este ruido. El ruido puede clasificarse en los siguientes tipos:

Gaussiano: produce pequeñas variaciones en la imagen; generalmente se debe a diferentes ganancias en la cámara, ruido en los digitalizadores, perturbaciones en la transmisión, etc. Se considera que el valor final del píxel sería el ideal más una cantidad correspondiente al error que puede describirse como una variable aleatoria gaussiana.

Impulsional (sal y pimienta): el valor que toma el píxel no tiene relación con el valor ideal, sino con el valor del ruido que toma valores muy altos o bajos (puntos blancos y/o negros) causados por una saturación del sensor o por un valor mínimo captado, si se ha perdido la señal en ese punto. Se encuentran también al trabajar con objetos a altas temperaturas, ya que las cámaras tienen una ganancia en el infrarrojo que no es detectable por el ojo humano; por ello las partes más calientes de un objeto pueden llegar a saturar un píxel.

Multiplicativo: La imagen obtenida es el resultado de la multiplicación de dos señales.

En la figura 8 se muestran los diferentes ruidos afectando a una imagen.

I.4 Procesamiento espacial

El procesamiento espacial esta formado por aquellas técnicas que operan directamente sobre los valores de los píxeles de la imagen. Las transformaciones son de la siguiente forma:

$$S(x,y) = F(I(x,y))$$

donde *I(x,y)* es la imagen original, *S(x,y)* la imagen resultante y *F* la transformación.

I.4.1 Operaciones aritméticas, lógicas y transformaciones geométricas.

Las operaciones aritméticas más usadas en procesamiento de imágenes son; suma, resta, multiplicación y división. Para que se pueda llevar a cabo una operación aritmética, ambas imágenes deben ser del mismo tamaño. En la figura 9 se muestra la suma de dos imágenes, la cual se realiza de la forma $C(x, y) = A(x, y) + B(x, y)$ mediante el comando `imadd` en Matlab.

Figura 8. Diferentes tipos de ruido afectando a una imagen.

Figura 9. Suma de dos imágenes

También es posible aumentar el brillo a una imagen sumándole un valor constante a cada píxel. En la figura 10 se muestra el efecto de sumar un escalar (50) a una imagen, el cual se realiza de la forma $B(x, y) = A(x, y) + a$.

Figura 10. Aumento del brillo de la imagen usando la suma de un escalar a cada píxel de la imagen, imagen original (izquierda) e imagen modificada (derecha).

La resta de imágenes consiste en restar de una imagen el valor correspondiente de otra imagen. Esta operación es un paso intermedio en algunos procesamientos más complejos, como la detección de movimiento, etc. La resta, al igual que la suma de imágenes requiere que ambas imágenes sean de igual tamaño. En la figura 11 se muestra el efecto de restar una imagen de otra, de la forma $C(x,y) = A(x,y) - B(x,y)$ mediante el comando `imsubtract` en Matlab.

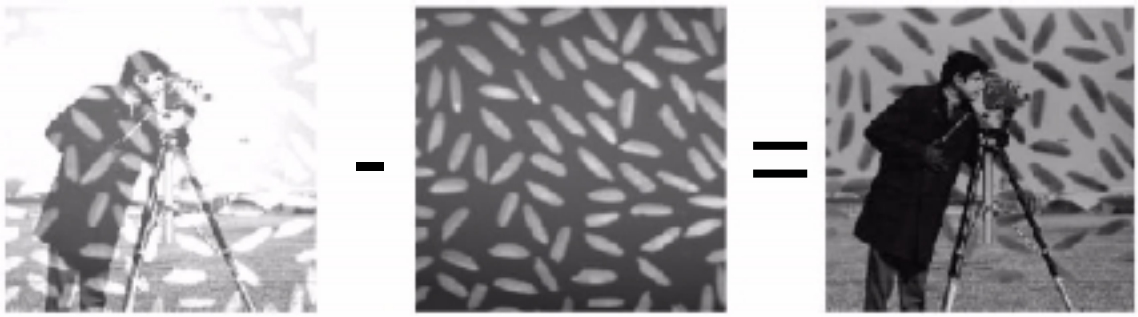


Figura 11. Resta de dos imágenes, imagen original (izquierda), imagen a restar (centro) y resultado (derecha).

En la figura 12 se muestra la resta de un escalar (50) a cada píxel de la imagen original, la cual se lleva a cabo de la forma $B(x,y) = A(x,y) - a$.

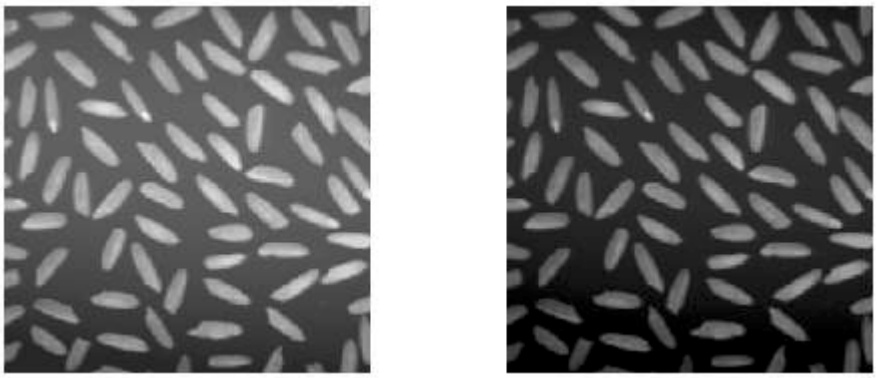


Figura 12. Resta de un escalar (50) a cada píxel de la imagen original (izquierda)

En el campo de las imágenes, la multiplicación se puede llevar a cabo, entre dos imágenes del mismo tamaño, multiplicando elemento a elemento cada uno de los píxeles de la imagen, de la forma $C(x,y) = A(x,y) \cdot B(x,y)$, en Matlab esto se realiza con el comando `immultiply`. En la figura 13 se muestra la multiplicación de dos imágenes.

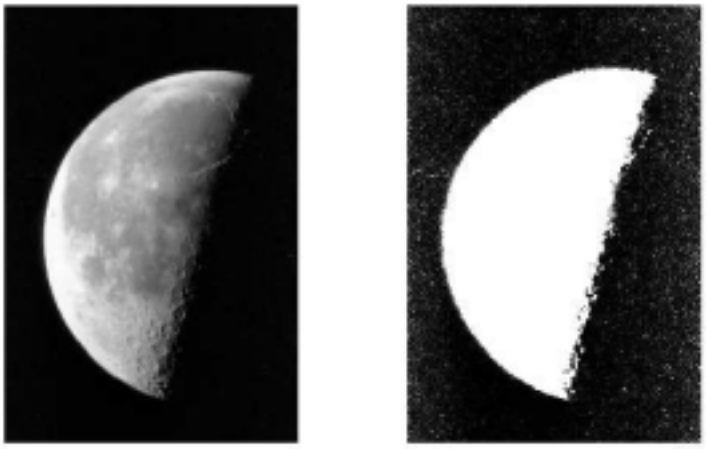


Figura 13 Multiplicación de una imagen por si misma, imagen original (izquierda), imagen resultante (derecha).

Cuando se multiplica cada uno de los píxeles de una imagen por un escalar, se le conoce como escalamiento, el cual se realiza de la siguiente forma $B(x,y) = a \cdot A(x,y)$. Cuando el escalar o constante es menor a 1, se oscurece la imagen y si es mayor a uno aumenta el brillo de la imagen. En la figura 14 se muestra el resultado de multiplicar la imagen original por el escalar 1.2

Figura 14. Multiplicación de una imagen por un escalar, imagen original (izquierda), imagen resultante (derecha).

La división de imágenes consiste en una división de elemento a elemento, como las demás operaciones vistas anteriormente. La división entre imágenes puede utilizarse para detectar cambios en dos imágenes, sin embargo, en lugar de dar el cambio absoluto de cada píxel, la división da el cambio fraccional o razón de cambio entre los valores de dos píxeles correspondientes. A la división de imágenes también se le conoce como

racionalización. En la figura 15 se presenta la división entre imágenes, la cual se realiza de la forma $C(x, y) = A(x, y) \div B(x, y)$.

Figura 15. División de imágenes, se muestra la división de la imagen original (izquierda) entre el fondo (centro) dando como resultado la figura de la derecha.

I.4.2 Operaciones lógicas

Las principales operaciones lógicas utilizadas en el procesamiento de imágenes son: AND, OR, NOT, las cuales se aplican solo a imágenes binarizadas. En la figura 16 se muestran las operaciones lógicas aplicadas a imágenes binarias.

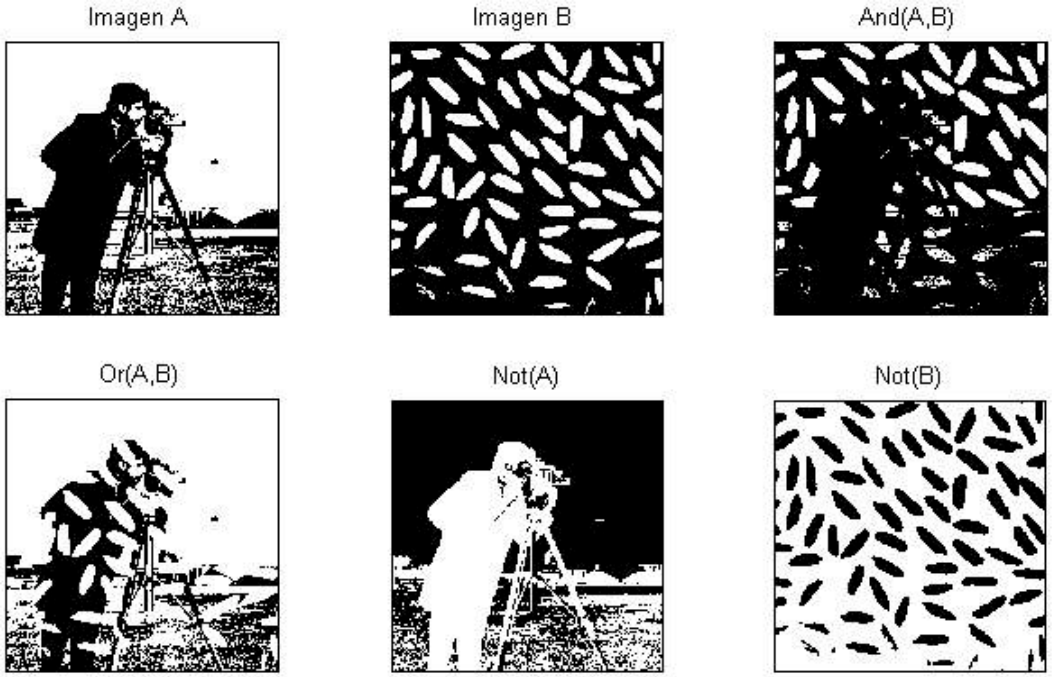


Figura 16. Operaciones lógicas aplicadas a imágenes binarias.

I.4.3 Transformaciones geométricas

Las transformaciones geométricas modifican las relaciones espaciales entre píxeles; a continuación se presentan algunas.

I.4.3.1 Interpolación

La interpolación es el proceso en el cual se estiman los valores de una imagen en una sección específica, cuando por ejemplo, se cambia el tamaño de una imagen y en la nueva imagen existen más píxeles que en la imagen original. Dentro de Matlab los comandos

`imresize` e `imrotate` utilizan interpolación bidimensional como paso intermedio en sus procesos.
De forma general, la interpolación de una imagen se presenta como:

$$f(x, y) = \sum\sum g(i, j)h(x - i, y - j)$$

donde: *g(x,y)* es la imagen original, *f(x,y)* representa la imagen procesada y *h(x,y*) es la interpolación (máscara). En el toolbox de *Image Processing* se encuentran implementados los siguientes métodos de interpolación: interpolación por el vecino más próximo, interpolación bilineal e interpolación bicúbica. Dichos métodos se explicarán más delante de forma breve. Los tres métodos de interpolación funcionan de forma similar, en cada caso para determinar el valor para un píxel interpolado, se encuentra el punto en la imagen original que corresponde a la imagen interpolada. Se asigna el valor del píxel interpolado calculando el promedio ponderado de el conjunto de píxeles hallados en la vecindad de dicho punto. Los tres métodos difieren en el conjunto de píxeles que consideran:

- Vecino más próximo: al píxel interpolado se le asigna el valor del píxel que corresponde
- Interpolación bilineal: el valor del píxel interpolado es el promedio ponderado de los píxeles en la vecindad 2x2 más cercana.
- Interpolación bicúbica: el valor del píxel interpolado es el promedio ponderado de los píxeles presentes en la vecindad 4x4 más cercana.

Nótese que el número de píxeles considerado aumenta la complejidad del cálculo, es por eso que la interpolación bilineal es más lenta que el método del vecino más próximo y el método bicúbico es más lento que el método bilineal. Nótese también que si se considera un mayor número de píxeles, se tendrán mejores resultados.

Para la mayoría de las funciones el método utilizado por omisión es el de vecino más próximo. Este método produce resultados aceptables para todos los tipos de imágenes y es el único método apropiado para imágenes indexadas. Sin embargo, para imágenes de intensidad y RBG generalmente se especifica la interpolación bilineal o bicúbica porque estos métodos proporcionan mejores resultados. Para imágenes RGB, la interpolación se ejecuta en los planos de color rojo, verde y azul de forma individual.

I.4.3.2 Amplificación/Reducción de imágenes

Para el cambio de tamaño de una imagen (amplificación/reducción) se utiliza el comando `imresize`. Este comando permite especificar: el tamaño de la imagen de salida (procesada), el método de interpolación utilizado y el filtro a usar para evitar el efecto alias. El efecto alias se presenta al reducir el tamaño de una imagen. Esto es debido a que se presenta una pérdida de información cuando se reduce el tamaño de una imagen.

En las figuras 17 y 18 se presenta un ejemplo de amplificación de imágenes usando los métodos de interpolación descritos anteriormente. La interpolación mediante Fourier se expone más adelante.

Figura 17. Imagen original

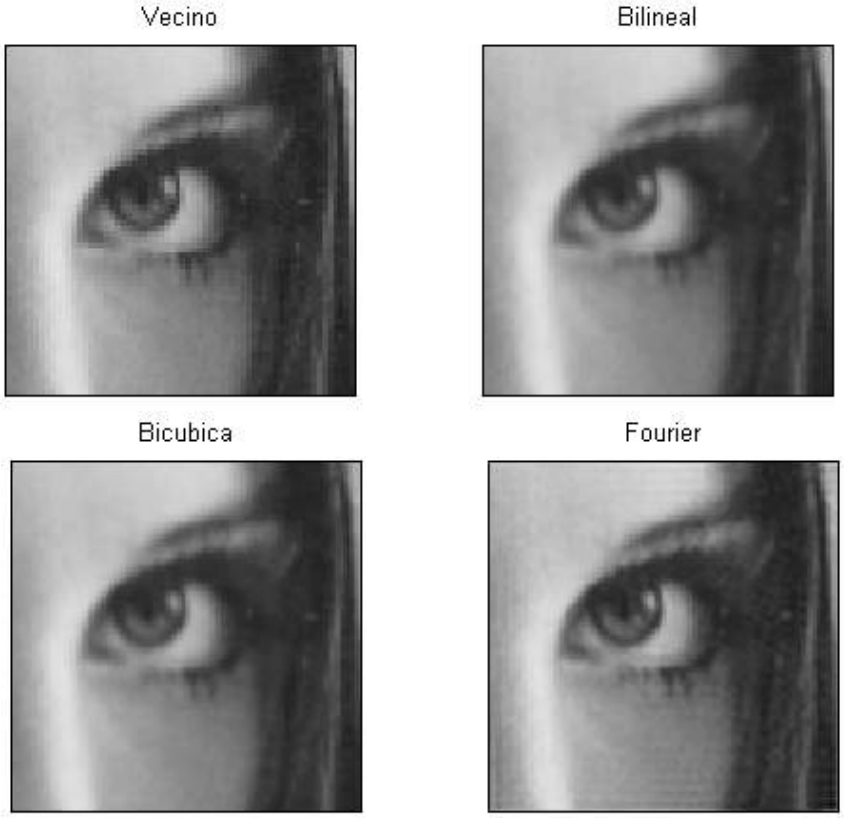


Figura 18. Amplificación de la figura 17 usando diferentes métodos de interpolación.

I.4.3.3 Traslación

Si se requiere trasladar el origen de una imagen se aplican las ecuaciones:

$$x_f = x_i + x_o$$
$$y_f = y_i + y_o$$

Que en coordenadas homogéneas es:

$$\begin{bmatrix} x_f \\ y_f \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_o \\ 0 & 1 & y_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Rotación respecto al origen

$$\begin{bmatrix} x_f \\ y_f \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -sen\theta & 0 \\ sen\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Rotación respecto a un punto cualquiera

$$\begin{bmatrix} x_f \\ y_f \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_o \\ 0 & 1 & y_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -sen\theta & 0 \\ sen\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_o \\ 0 & 1 & -y_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

En la figura 19 se muestra la rotación de la figura 17 usando diferentes métodos de interpolación.

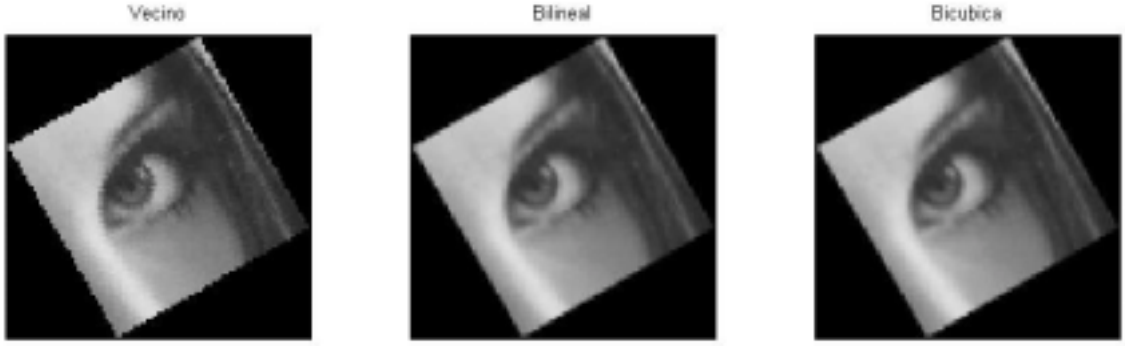


Figura 19. Rotación de la imagen de la figura 17 usando diferentes métodos de interpolación.

I.4.3.4 Convolución bidimensional

La convolución bidimensional discreta es la base de algunos procesamientos comunes, como el filtrado de imágenes. En la convolución, el valor de un píxel de salida se calcula mediante la suma ponderada de los píxeles vecinos. Dentro del campo del procesamiento de imágenes, la convolución se realiza entre la imagen y una matriz (los coeficientes del filtro) llamada máscara para filtrar una imagen. En Matlab la convolución bidimensional (aplicada a imágenes) se encuentra en el comando `conv2`. La convolución de *f(x,y)* y *h(x,y)* está dada por:

$$g(x,y) = h(x,y) * f(x,y) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f(i,j) h(x-i, y-j)$$

Lo más común es usar convoluciones de 3 x 3 elementos; entonces la ecuación anterior se convierte en:

$$g(x,y) = h(x,y) * f(x,y) = \sum_{i=0}^{2} \sum_{j=0}^{2} f(i,j) h(x-i, y-j)$$

que, por ejemplo, para obtener g(2,2) se tiene:

$$g(2,2) = \sum_{i=0}^{2} \sum_{j=0}^{2} f(i,j) h(2-i, 2-j) =$$

$$f(0,0)h(2,2) + f(0,1)h(2,1) + f(0,2)h(2,0) + \cdots$$

$$\cdots + f(1,0)h(1,2) + f(1,1)h(1,1) + f(1,2)h(1,0) + \cdots$$
$$\cdots + f(2,0)h(0,2) + f(2,1)h(0,1) + f(2,2)h(0,0)$$

Considérese que la imagen es la mostrada en la figura 20:

A =

17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

Figura 20. Imagen

y la máscara se muestra en la figura 21:

h =

8	1	6
3	5	7
4	9	2

Figura 21. Máscara de convolución.

En la figura 22 se muestra como calcular el píxel de salida (procesada) mediante los siguientes pasos:

1. Rotar la máscara de convolución 180 grados a partir del elemento del centro. La máscara rotada queda entonces como:

2	9	4
7	5	3
6	1	8

Figura 22. Máscara rotada para la convolución.

2. Sobreponer el elemento central de la máscara de tal forma que quede sobre el elemento de interés, en este caso el elemento (2,4) de A, tal como se muestra en la figura 23.

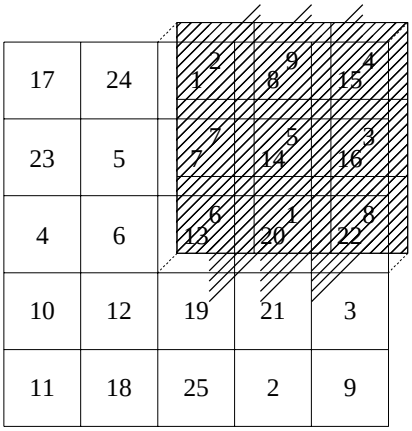


Figura 23. Convolución para obtener el valor de A(2,4)

3. Multiplicar cada peso (valor) de la máscara rotada por el píxel de A que se encuentra “bajo” la máscara.
4. Sumar los productos individuales obtenidos en el paso 3.

Por ejemplo, para el píxel (2,4), el píxel de salida (procesado) es:

$$1 \cdot 2 + 8 \cdot 9 + 15 \cdot 4 + 7 \cdot 7 + 14 \cdot 5 + 16 \cdot 3 + 13 \cdot 6 + 20 \cdot 1 + 22 \cdot 8 = 575$$

Cuando se trabaja en los extremos de la imagen, se acostumbra insertar ceros (zero padding) en los extremos, tal como se muestra en la figura 24.

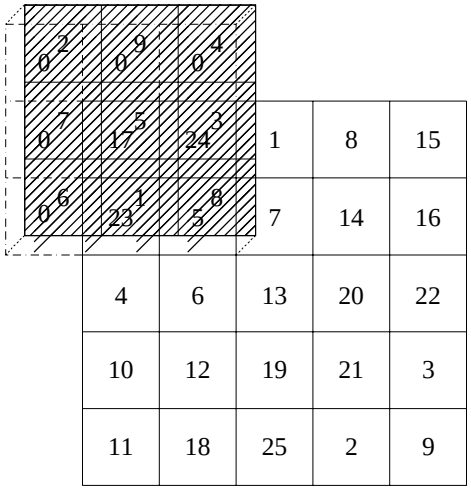


Figura 24. Inserción de ceros (zero padding) en los extremos.

I.4.3.5 Correlación

La correlación es una operación parecida a la convolución, en la cual el valor de un píxel de salida se calcula como la suma ponderada de los píxeles vecinos. La diferencia está en que la matriz de pesos o máscara, en este caso llamada núcleo o kernel de correlación no se rota durante el cálculo. La correlación está dada por:

$$g(x,y) = h(x,y) \circ f(x,y) = \sum_{i=-\infty}^{\infty} \sum_{j=\infty}^{\infty} f^{*}(i,j) h(x+i, y+j)$$

La correlación se utiliza para encontrar el parecido entre píxeles de una imagen. Si los píxeles son iguales o parecidos, se dice que están altamente correlacionados entre si. La correlación permite hallar patrones.
La diferencia entre la correlación y la convolución estriba en que la máscara de correlación no se rota como en la convolución. En la figura 25 se muestra como se calcula la correlación para el píxel (2,4) de la imagen A usando como máscara de correlación a h.

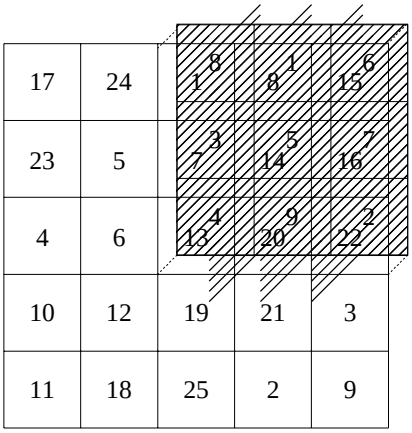


Figura 25. Correlación de A con h.

El algoritmo de la correlación opera de la siguiente forma:

1. Sobreponer el elemento central de la máscara de tal forma que quede sobre el elemento de interés, en este caso el elemento (2,4) de A, tal como se muestra en la figura 22.
2. Multiplicar cada peso (valor) de la máscara rotada por el píxel de A que se encuentra “bajo” la máscara.
3. Sumar los productos individuales obtenidos en el paso 2.

Por ejemplo, para el píxel (2,4), el píxel de salida (procesado) es:

$$1 \cdot 8 + 8 \cdot 1 + 15 \cdot 6 + 7 \cdot 3 + 14 \cdot 5 + 16 \cdot 7 + 13 \cdot 4 + 20 \cdot 9 + 22 \cdot 2 = 585$$

I.5 Procesamiento en el dominio de la frecuencia

En el campo de las imágenes, el dominio de la frecuencia es aquel en el que una imagen se representa como la suma de señales periódicas con diferentes frecuencias. Por ejemplo, la transformada de Fourier de una imagen es la representación de dicha imagen como una suma de exponenciales complejos de diferentes magnitudes, frecuencias y fases. Este tipo de transformaciones frecuenciales se llevan a cabo para una amplia gama de procesamientos, entre los cuales se encuentran: la convolución, el mejoramiento de imágenes, la detección de características, compresión, etc.

I.5.1 Series de Fourier

Dada una función periódica en el dominio del tiempo *f(t)*, es lógico pensar que esta función se puede expresar mediante la suma de otras funciones periódicas. Lo anterior se expresa mediante las series de Fourier, las cuales dan las fórmulas matemáticas que

expresan esta relación con las funciones periódicas seno y coseno. Así, cualquier función *f(t)* con periodo T_o se puede expresar como:

$$f(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos(k\omega_0 t) + b_k sen(k\omega_0 t)$$

Donde:

$$a_0 = \frac{1}{T_0}\int_0^{T_0} f(t)\, ft\,,\ a_k = \frac{2}{T_0}\int_0^{T_0} f(t)\cos(k\omega_0 t)dt\,,\ b_k = \frac{2}{T_0}\int_0^{T_0} f(t)sen(k\omega_0 t)dt\,,\ \omega_0 = \frac{2\pi}{T_0}$$

La serie de Fourier de la señal cuadrada se puede representar como:

$$f(t) = \frac{4}{\pi}\sum_{k=0}^{\infty}\frac{sen((2k+1)t)}{2k+1}$$

En la figura 26 se presenta la señal cuadrada

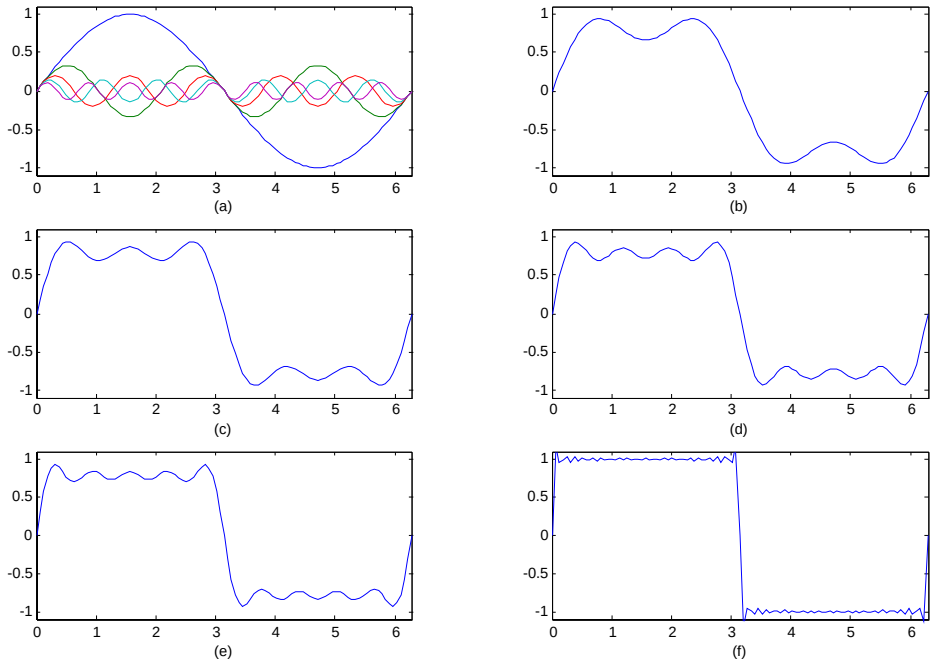


Figura 26. Formación de la señal cuadrada mediante la serie de Fourier. (a) cinco señales senoidales (términos); (b) suma de dos términos; (c) suma de tres términos; (d) suma de cuatro términos; (e) suma de cinco términos; (f) suma de veinte términos.

I.5.2 Transformada de Fourier

La transformada de Fourier es una extensión de las series de Fourier a señales no periódicas. El par transformado analítico de Fourier está dado por:

$$F[x(t)] = X(f) = \int_{-\infty}^{\infty} x(t)e^{-j2\pi ft}dt \ \text{(ec. de análisis)}$$

$$x(t) = F^{-1}[X(f)] = \int_{-\infty}^{\infty} X(f)e^{j2\pi ft}df = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{j\omega t}d\omega \ \text{(ec. de síntesis)}$$

donde $X(\omega) = 2\pi X(f)$.

Para el caso unidimensional discreto, el par transformado de Fourier está dado por:

$$F[x(n)] = X(k) = \sum_{n=0}^{N-1} x(n) e^{-\frac{j2\pi}{N}kn} \quad ; \text{k = 0, 1, ..., N-1 (ec. de análisis)}$$

$$F^{-1}[X(k)] = x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{\frac{j2\pi}{N}kn} \quad ; \text{n = 0, 1, ..., N-1 (ec. de síntesis)}$$

Para el caso bidimensional discreto, se tiene que:

$$F[x(k,l)] = X(k,l) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x(m,n) e^{-\frac{j2\pi}{M}km} e^{-\frac{j2\pi}{N}\ln} \quad ; \text{k = 0, 1, ..., M-1; l = 0, 1, …, N-1}$$

$$F^{-1}[X(k,l)] = x(m,n) = \frac{1}{MN} \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} X(k,l) e^{\frac{j2\pi}{M}km} e^{\frac{j2\pi}{N}\ln} \quad ; \text{m = 0, 1, ..., M-1; n = 0,1,…, N-1}$$

Los valores *X(k,l)* son los coeficientes de la transformada de Fourier de la imagen *x(m,n).* A los coeficientes referentes a la frecuencia cero, *X(0,0)* usualmente se les conoce como la componente de corriente directa. La transformada de Fourier se encuentra implementada en Matlab en el comando `fft` (para el caso unidimensional), `fft2` (para el caso bidimensional) y `fftn` (para el caso N-dimensional). Las transformadas inversas se encuentran en los comandos `ifft` (para el caso unidimensional), `ifft2` para el caso bidimensional e `ifftn` (para el caso N-dimensional).

La fase obtenida mediante la transformada de Fourier contiene información esencial sobre la estructura de la imagen. La amplitud por si sola implica solo que existe una estructura periódica dentro de la imagen, pero no especifica donde se encuentra. Es decir, si no se conoce la fase de la transformada de Fourier, se puede determinar que objetos hay en la imagen, pero no su posición. Por lo tanto, resulta obvio que si se obtiene un espectro de potencia, se tendría muy poca información sobre la imagen debido a que la fase se ha perdido. Si se asocia un nivel de gris con la amplitud de un proceso físico, por ejemplo una oscilación armónica, entonces el espectro de potencia proporciona la distribución de la energía en el dominio de la frecuencia.

I.5.2.1 Aplicaciones de la transformada de Fourier

En esta sección se presentan algunos procesamientos de imágenes relacionados con la transformada de Fourier.

Respuesta a la frecuencia de filtros lineales

La transformada de Fourier de la respuesta al impulso de un filtro lineal proporciona la respuesta a la frecuencia del filtro. Esto puede obtenerse mediante el comando `freqz2`, el cual calcula y despliega la respuesta a la frecuencia de un filtro.

Convolución rápida

Una propiedad clave de la transformada de Fourier es que la multiplicación de dos transformadas de Fourier de dos funciones corresponde a la convolución de las funciones espaciales asociadas. Esta propiedad, junto con la transformada rápida de Fourier forman la base para el algoritmo de la convolución.

Supóngase que A es una matriz de M por N y B es una matriz de *P* por *Q*; la convolución de A y B se obtiene de la siguiente forma:

1. Se agregan ceros a A y a B para que su longitud sea de al menos (*M+P+1*) por (*N+Q-1*). Generalmente se les agrega ceros a A y B para que su tamaño sea una potencia de 2 debido a que el algoritmo de la `fft2` es más rápido para potencias de dos.
2. Se calcula la transformada de Fourier bidimensional mediante `fft2`
3. Se multiplican las dos transformadas
4. Se obtiene la transformada inversa de Fourier bidimensional de la multiplicación mediante el comando `ifft2`.

Localización de características en imágenes

La transformada de Fourier se utiliza también para realizar correlaciones. La correlación se utiliza para localizar algunas características en una imagen. Por ejemplo si se desea encontrar la letra "a" en una imagen que contenga texto, se establece un patrón (sección de la imagen con la característica de búsqueda deseada) con la letra "a" . Posteriormente se obtiene la correlación de la imagen patrón y la imagen original rotando 180º la imagen patrón y se utiliza la técnica de convolución basada en la transformada de Fourier, descrita anteriormente (Nota: la convolución es equivalente a la correlación si se rota el kernel de convolución 180º. Para buscar coincidencias en la imagen se utiliza la transformada de Fourier y la transformada inversa de Fourier.

Amplificación de imágenes

Para realizar la ampliación de una imagen se utiliza una interpolación en el dominio del tiempo (espacial) mediante la transformada discreta de Fourier, siempre y cuando los lados de la imagen tengan un número de píxeles que sea potencia de dos. A continuación se presenta el procedimiento para cuadruplicar el tamaño de una imagen (duplicar cada lado),

1. Transformar la imagen A al dominio de la frecuencia (imagen o matriz B).
2. Dividir la imagen transformada (imagen o matriz B) en cuatro partes iguales, tal como se muestra en la figura 27.

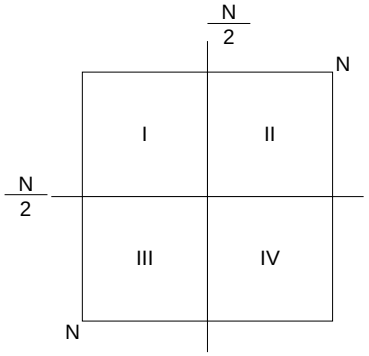


Figura 27. División en cuatro partes iguales de la imagen transformada (B)

3. Insertar N ceros a cada renglón de la matriz B, ver la figura 28 (b); enseguida insertar N ceros a cada columna, tal como se muestra en la figura 28(c) para formar una matriz aumentada, B'.

4. El siguiente paso es antitransformar la matriz aumentada B' para obtener una matriz A' aumentada, de dimensiones 2N x 2N.
5. Dividir cada elemento de la matriz X' entre 64. La relación es $4^{(2n-1)}$, donde n es el número de veces que se amplifica la imagen, en este caso n = 2.

El procedimiento anterior amplifica la imagen original por un factor de 2 (el área original se cuadruplica). Si se requiere un factor de amplificación diferente de dos, por ejemplo un factor α, deberán agregarse a B (α -1) ceros.

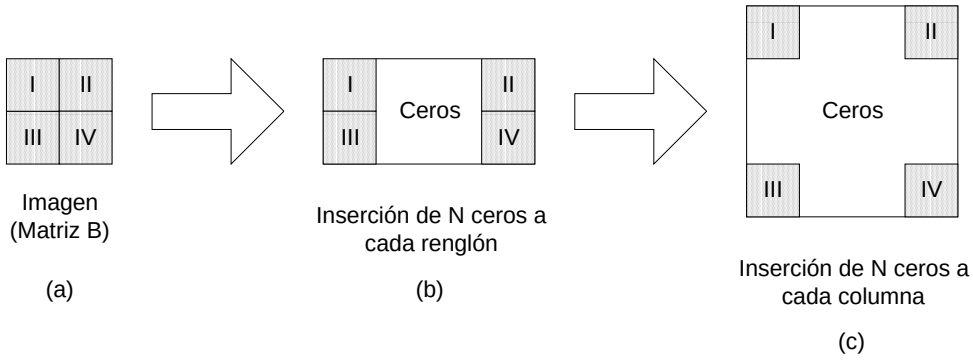


Figura 28. Inserción de ceros para obtener la matriz B aumentada (B').

I.5.3 Transformada discreta coseno

La transformada discreta coseno (TDC) representa una imagen como la suma de senoidales de diferentes amplitudes y frecuencias. La transformada discreta coseno tiene una propiedad tal que para una imagen típica, la mayoría de la información visualmente significativa de una imagen se concentra en solo unos cuantos coeficientes de la DCT. Por esta razón, la TDC es comúnmente usada en aplicaciones de compresión de imágenes. Por ejemplo, la TDC es el corazón del algoritmo estándar de compresión de imágenes conocido como JPEG (Joint Photographic Experts Group).

La transformada discreta coseno de una matriz A de M por N, tal como se define a continuación:

$$B_{pq} = \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} A_{mn} \cos\frac{\pi(2m+1)p}{2M} \cos\frac{\pi(2n+1)q}{2N}, \qquad \begin{matrix} 0 \le p \le M-1 \\ 0 \le q \le N-1 \end{matrix}$$

$$\alpha_p = \begin{cases} \dfrac{1}{\sqrt{M}}, & p = 0 \\ \sqrt{\dfrac{2}{M}} & 1 \le p \le M-1 \end{cases} \qquad \alpha_p = \begin{cases} \dfrac{1}{\sqrt{N}}, & q = 0 \\ \sqrt{\dfrac{2}{N}} & 1 \le q \le N-1 \end{cases}$$

Los valores B_{pq} son los coeficientes de la transformada discreta coseno de A. La TDC es invertible, y su inversa está dada por:

$$A_{mn} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \alpha_p \alpha_q B_{pq} \cos\frac{\pi(2m+1)p}{2M} \cos\frac{\pi(2n+1)q}{2N}, \qquad \begin{matrix} 0 \le m \le M-1 \\ 0 \le n \le N-1 \end{matrix}$$

$$\alpha_p = \begin{cases} \frac{1}{\sqrt{M}}, & p = 0 \\ \sqrt{\frac{2}{M}} & 1 \le p \le M-1 \end{cases} \qquad \alpha_p = \begin{cases} \frac{1}{\sqrt{N}}, & q = 0 \\ \sqrt{\frac{2}{N}} & 1 \le q \le N-1 \end{cases}$$

La ecuación de la TDC inversa se puede interpretar como una matriz A de M por N que puede escribirse como la suma de MN funciones de la forma:

$$\alpha_p \alpha_q \cos\frac{\pi(2m+1)p}{2M} \cos\frac{\pi(2n+1)q}{2N}, \qquad \begin{matrix} 0 \le m \le M-1 \\ 0 \le n \le N-1 \end{matrix}$$

A estas funciones se les llama funciones básicas de la TDC, estos coeficientes B_{pq} se conocen como pesos aplicados a cada función base. La razón por la cual las imágenes pueden ser comprimidas y recuperadas exitosamente con pequeños errores es la gran cantidad de redundancia en las imágenes típicas. El propósito de esta transformada es obtener un conjunto de coeficientes que representen la imagen con valores que no estén correlacionados (es decir, cada valor en el arreglo o imagen proporciona nueva información no dada por ningún otro valor en el arreglo). Algunos valores en el arreglo transformado dan poca o ninguna información acerca de la imagen original y pueden ser descartados. En la figura 29 se presenta un ejemplo de compresión usando la transformada discreta coseno (comando dct2) usado para transformar la imagen mostrada. La imagen transformada muestra de forma logarítmica la distribución de los niveles de gris en la imagen transformada. La imagen cuantizada muestra los niveles de gris transformados representativos de forma logarítmica, los tonos negros pueden desecharse (en este caso se desecharon valores inferiores a 10), con lo que se comprime la imagen y por último se muestra la imagen reconstruida.

I.6 Procesamiento de imágenes básico

En esta sección se presentan y describen algunos procesamientos más comunes.

I.6.1 Binarización de una imagen

La binarización de una imagen consiste en comparar los niveles de gris presentes en la imagen con un valor (umbral) predeterminado. Si el nivel de gris de la imagen es menor que el umbral predeterminado, se le asigna al píxel de la imagen binarizada el valor 0 (negro), y si es mayor, se le asigna un 1 (blanco). De esta forma se obtiene una imagen en blanco y negro. Generalmente se utiliza un umbral de 128 si se trabaja con 255 niveles de gris, pero en algunas aplicaciones se requiere de otro umbral. En la figura 30 se muestra un ejemplo de imagen binarizada.

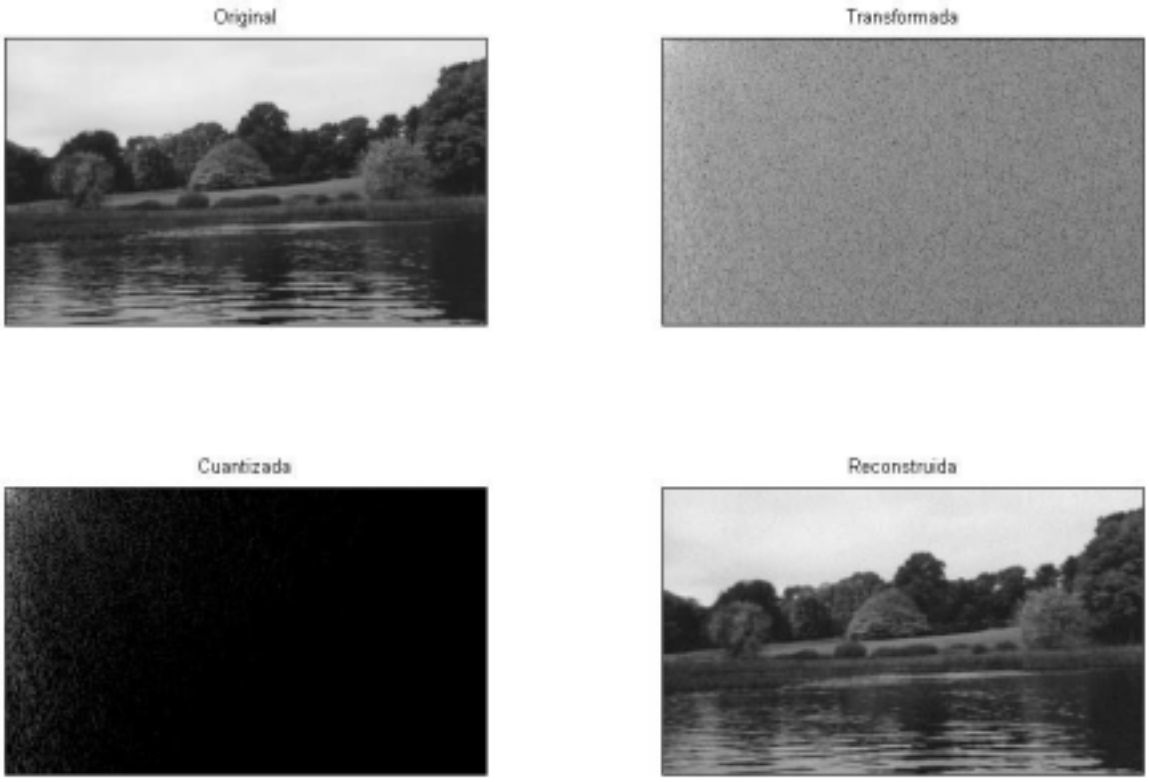


Figura 29. Compresión de imágenes usando la TDC.

Fig. 30. Binarización de una imagen

I.6.2 Manipulación del contraste

El histograma que se muestra en la figura 31 toma valores limitados, por lo que el contraste en la imagen es muy bajo y apenas se aprecian los detalles. Se desea encontrar una función que produzca una nueva imagen que si cubra todo el conjunto de valores posibles de la imagen (niveles de gris). Si a y b son los valores mínimos y máximos, respectivamente, puede definirse la función T(c) que asigna los nuevos valores de gris a partir de los antiguos:

$$y = T(c) = A\frac{c-a}{b-a}$$

donde: a y b son los límites inferior y superior, c es el valor de gris de la imagen original y A es el valor máximo que se desea que tengan los píxeles de la imagen.

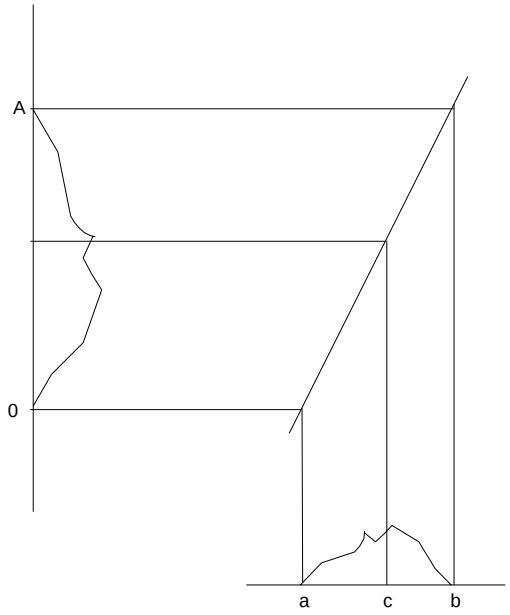


Figura 31. Expansión del histograma de la imagen.

En la figura 32 se muestra el resultado de aplicar a la imagen la modificación del contraste, procesamiento también conocido como ecualización de la imagen. El contraste (separación entre los niveles de gris) ha mejorado y ahora se aprecian mejor los detalles de la imagen. En el nuevo histograma puede observarse como la separación entre los diferentes niveles de gris es mayor. En este caso la separación es igual para todos los niveles de gris debido a que la transformación es lineal. Nótese que aunque la imagen se ve mejor, la información es la misma en ambas imágenes, lo único que se ha hecho es asignar nuevos niveles de gris, pero los píxeles que tenían un nivel de gris determinado en la imagen original diferente a los niveles de gris inferior y superior, son los mismos en la imagen nueva.

Para un caso más general la función buscada tendrá la forma (ver la figura 33)

$$y = T(x) = \begin{cases} \alpha x & 0 \le x \le a \\ \beta(x-a) + y_a & a \le x \le b \\ \gamma(x-b) + y_b & b \le x \le L \end{cases}$$

donde:

y, *x* son los niveles de gris de las imágenes resultante y original

α, β, γ son ganancias de cada tramo

a, *b* y *L* son los intervalos de ganancia

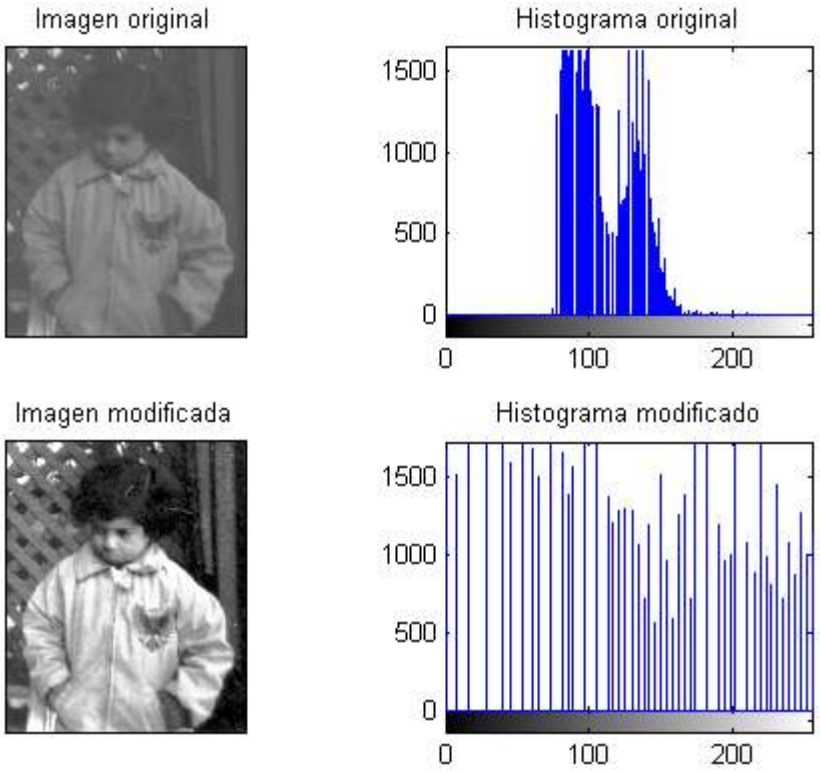


Figura 32. Modificación de contraste

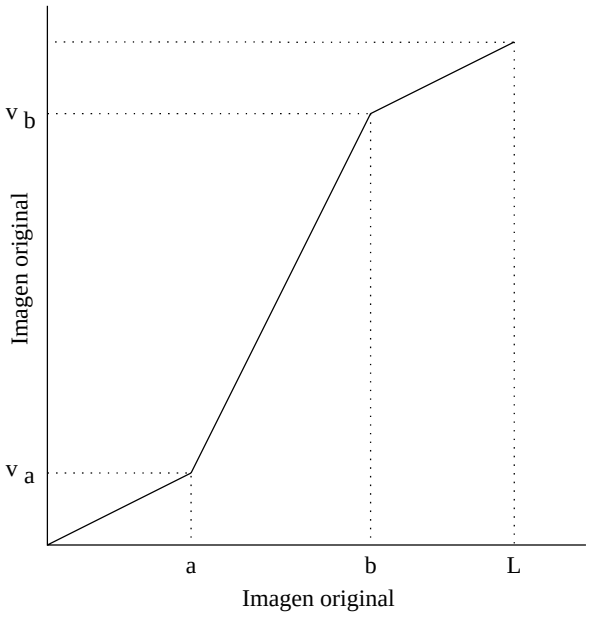


Figura 33. Caso general

I.6.3 Modificación del contraste

La modificación del contraste consiste en aplicar una función a cada uno de los píxeles de la imagen, de la forma: $p = m^a$

donde:

m es el valor de gris de la imagen original
p es el nuevo valor de gris en la imagen resultante
a es la potencia a la que se eleva

Entre las transformaciones más usuales se encuentran:

Función inversa $p = 255 - m$

Función cuadrada $p = \frac{m^2}{255}$

Función cúbica $p = \frac{m^3}{255^2}$

Función raíz cuadrada $p = \sqrt{255m}$

Función raíz cúbica $p = \sqrt[3]{255^2 m}$

Función logarítmica $p = 255\frac{\ln(1+m)}{\ln(1+255)}$

El valor 255 se utiliza para normalizar los valores entre 0 y 255 si se trabaja con imágenes con niveles de gris de 8 bits, de lo contrario se debe remplazar este valor por el valor máximo representable con el número de bits utilizados.

Con la función cuadrada y cúbica se oscurece la imagen resultante. Con las funciones raíz cuadrada, raíz cúbica y logarítmica sucede lo inverso.

I.6.4 Modificación del histograma

Los métodos anteriores modifican cada nivel de gris y dependen únicamente de su valor y por lo tanto, son locales. Si se desea adquirir una información global de la imagen, la forma más fácil de hacerlo es analizar y modificar el histograma. Esto se hace con la idea de que éste se ajuste a una forma predeterminada; la forma más usual se conoce como ecualización del histograma, en la que se pretende que éste sea horizontal, es decir, que para todos los valores de gris se tenga el mismo número de píxeles.

La ecualización del histograma se realiza trabajando sobre el histograma acumulado, el cual está dado por:

$$H(i) = \sum_{k=0}^{i} h(k)$$

Si el histograma fuera totalmente plano, el histograma para cada nivel de gris sería:

$$G(i') = (i+1)\frac{NM}{256}$$

donde N y M son las dimensiones de la imagen y 256 corresponde al número de niveles dado por el número de bits con los que se representan (en este caso 8 bits).
Como se desea que *G(i') = H(i),* se tiene que;

$$(i'+1)\frac{NM}{256} = H(i) \therefore \quad i' = \frac{256}{NM}H(i) - 1$$

Debido a que los niveles de gris son únicamente valores enteros, se realiza un cambio en los niveles de gris de acuerdo a:

$$i_{nuevo} = \text{Parte entera}\left[\frac{256}{NM} H\left(i_{anterior}\right) - 1\right]$$

I.6.5 Filtrado de una imagen

El filtrado es una técnica para modificar o mejorar a una imagen. Por ejemplo, un filtro puede resaltar o atenuar algunas características. El filtrado es una operación de vecindario, en la cual el valor de un píxel dado en la imagen procesada se calcula mediante algún algoritmo que toma en cuenta los valores de los píxeles de la vecindad de la imagen original.

I.6.5.1 Filtros lineales espaciales

El ruido en una imagen es una característica que se desea eliminar, y al ser este variaciones sobre los niveles de gris, le corresponden las frecuencias altas. Si se supone que el ruido es una señal que se suma a la señal (imagen) original, el nivel de gris de un píxel puede definirse como la suma del nivel de gris ideal y el ruido:

$$f(x, y) = f_i(x, y) + r(x, y)$$

Aunque el ruido esta siempre presente, el que afecte más o menos a un píxel determinado es aleatorio. Si se trata de un ruido Gaussiano, este esta definido por una distribución normal de media cero y variancia típica de σ .

I.6.5.2 Filtro pasa bajas espacial

Una forma de eliminar el ruido consiste en hacer pasar la imagen por un filtro pasa bajas que disminuya la ganancia de las componentes de alta frecuencia. El filtro más sencillo e intuitivo es aquel que tiene coeficientes unitarios en todos los elementos, tal como se muestra a continuación.

1	1	1
1	1	1
1	1	1

Si al resultado se le multiplica por un noveno, se obtiene la media de todos los píxeles, por lo que el ruido disminuye. Sin embargo, este filtro presupone que la influencia de todos los píxeles es igual. Otra consideración es que cuanto más alejado esté el píxel del central, su valor será menor y se obtiene la siguiente máscara:

1	1	1
1	2	1
1	1	1

Si se desea dar mayor peso al píxel central que a sus vecinos, y a los vecinos tipo 4 que a los de tipo 8, se tiene

1	2	1
2	4	2
1	2	1

En general, se tiene:

1	b	1
b	b^2	b
1	b	1

debiendo ser la ganancia de todas ellas la unidad para no variar la imagen. El filtrado de imágenes en Matlab está implementado en el comando *filter2*.

I.6.5.3 Filtrado por la mediana

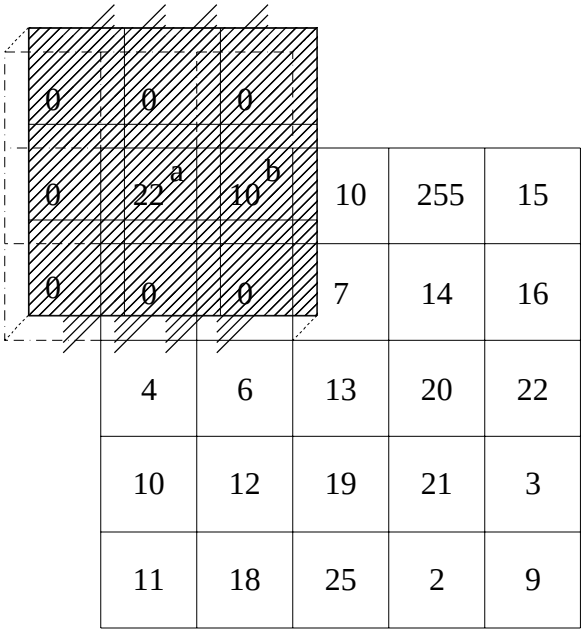
El filtrado por la mediana permite eliminar el ruido tipo sal y pimienta, es decir, elimina puntos blancos y negros presentes en la imagen. En una secuencia de números $x_1, x_2, ...x_N$, la mediana es aquel valor que cumple que (N-1)/2 elementos tienen un valor menor o igual a ella y que (N-1)/2 tiene un valor mayor o menor que la mediana. La mediana se obtiene ordenando las intensidades de los píxeles de menor a mayor, y el píxel que se encuentra en (N-1)/2 es la mediana , tal como se muestra en la figura 34. A continuación se muestra como el filtrado por la mediana puede eliminar un valor.

vecindad de x = [0,0,0,0,255,0,0,0,0]

vecindad de x = [0,0,0,0,0,0,0,0,255]

↑

Mediana



vecindad de a = [0,0,0,0,22,10,0,0,0]

Se ordena de mayor a menor

vecindad de a = [0,0,0,0,0,0,0,10,22]

Mediana

Figura 34. Vecindad de a y obtención de la mediana.

En Matlab este filtro se encuentra implementado en el comando *medfilt2*.

I.6.5.4 Realce de bordes

El realce de bordes en una imagen tiene un efecto opuesto a la eliminación de ruido; consiste en enfatizar o resaltar aquellos píxeles que tienen un valor de gris diferente al de sus vecinos. Cabe resaltar que si la imagen contiene ruido, su efecto se multiplicará, por lo que ser recomienda primero eliminar el ruido. En la figura 35 se muestra un ejemplo de realce de contornos.

Figura 35. Realce de una imagen

En el realce de imágenes consiste en aumentar la ganancia de las altas frecuencia, es decir:

Imagen resultante = (Ganancia)(Imagen Original) – Bajas frecuencias

De forma general, la máscara usada para realzar los bordes es:

-1	-1	-1
-1	A	-1
-1	-1	-1

donde:

$$A = 9 \cdot \text{Ganancia} - 1$$

y todo ello multiplicado por un noveno.

I.6.5.5 Detección de contornos

La detección de contornos es un paso intermedio en el reconocimiento de patrones en imágenes digitales. En una imagen, los contornos corresponden a los limites de los objetos presentes en la imagen. Para hallar los contornos se buscan los lugares en la imagen en los que la intensidad del píxel cambia rápidamente, generalmente usando alguno de los siguientes criterios:

- Lugares donde la primera derivada (gradiente) de la intensidad es de magnitud mayor que la de un umbral predefinido
- Lugares donde la segunda derivada (laplaciano) de la intensidad tiene un cruce por cero.

En el primer caso se buscarán grandes picos y en el segundo cambios de signo, tal como se muestra en la figura 36.

I.6.5.5.1 Técnicas basadas en el gradiente

Estas técnicas se basan en una aproximación al concepto de la derivada para espacios discretos. Esta generalización se basa en el cálculo de diferencias entre píxeles vecinos; estas diferencias, según la relación de píxeles considerados, puede dar lugar a derivadas unidimensionales o bidimensionales, así como aplicarse en una dirección determinada de la imagen o en todas direcciones. Otras aproximaciones diferenciales de gran utilidad son la de Roberts y la de Sobel.

El operador gradiente G aplicado a una imagen f(x,y) esta definido como:

$$\nabla f(x,y) = \begin{bmatrix} G_x & G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix}$$

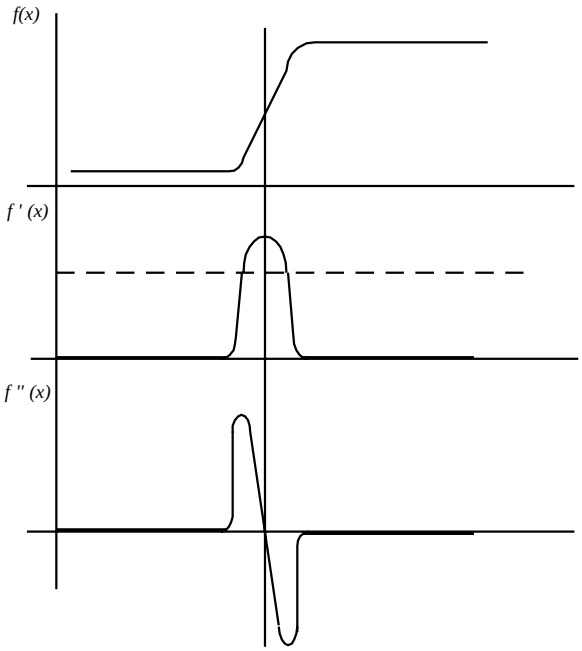


Figura 36. Detección de contornos mediante la primera y segunda derivada.

El vector gradiente representa el cambio máximo de intensidad para el punto (x,y); su magnitud y dirección están dados por:

$$|\nabla f| = \sqrt{G_{xx}^2 + G_y^2}$$

$$\angle \nabla f = \arctan\left(\frac{G_y}{G_x}\right)$$

siendo la dirección del gradiente perpendicular al borde. Para reducir el costo computacional, generalmente se aplica:

$$|\nabla f| = |G_x| + |G_y|$$

Debido a que las imágenes digitales no son señales continuas, se tiene:

$$\nabla f(x,y) = \begin{bmatrix} G_x & G_y \end{bmatrix} = \begin{bmatrix} \frac{\Delta f}{\Delta x} & \frac{\Delta f}{\Delta y} \end{bmatrix}$$

que se puede representar mediante las máscaras:

$$G_x = \frac{\Delta f}{\Delta x} \qquad \begin{array}{|c|c|} \hline -1 & 1 \\ \hline \end{array} \quad * \quad f(x,y)$$

$$G_y = \frac{\Delta f}{\Delta y} \qquad \begin{array}{|c|} \hline -1 \\ \hline 1 \\ \hline \end{array} \quad * \quad f(x,y)$$

Estas máscaras generalmente no se utilizan debido a que son muy poco sensibles al ruido al tomar en cuenta solamente la información de dos píxeles. Entre los filtros (operadores) más usados, que además permiten obtener un gradiente suavizado, se encuentran: Roberts, Prewitt, Sobel e Isotrópico. En la figura 37 se muestran las máscaras referentes a estos operadores.

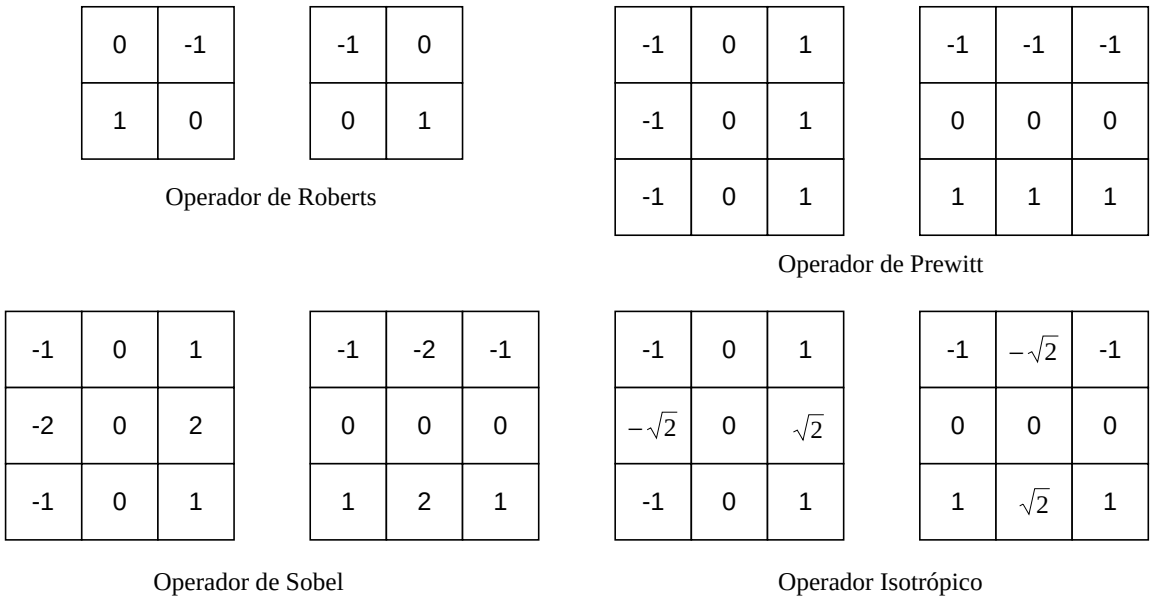


Figura 37. Máscaras para los operadores: Roberts, Prewitt, Sobel e Isotrópico.

I.6.5.5.2 Técnicas basadas en el laplaciano

El laplaciano es la segunda derivada de una función y representa la derivada de esta respecto a todas las direcciones, y esta dado por:

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

Generalmente para el laplaciano se utilizan las máscaras mostradas en la figura 38. Nótese que el píxel central toma el valor negativo de la suma de todos los que lo rodean, de tal forma que la suma aritmética de todos los píxeles sea cero.

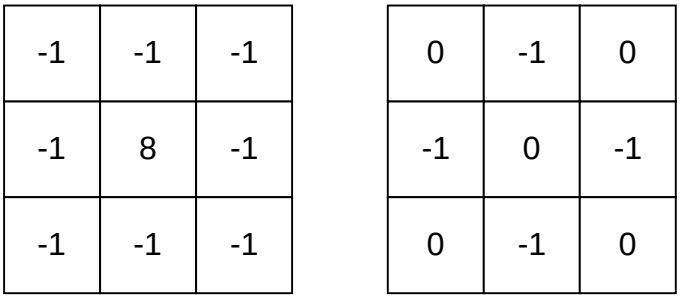


Figura 38. Máscaras utilizadas para el operador laplaciano.

En la figura 39 se presentan algunos ejemplos de detección de contornos.

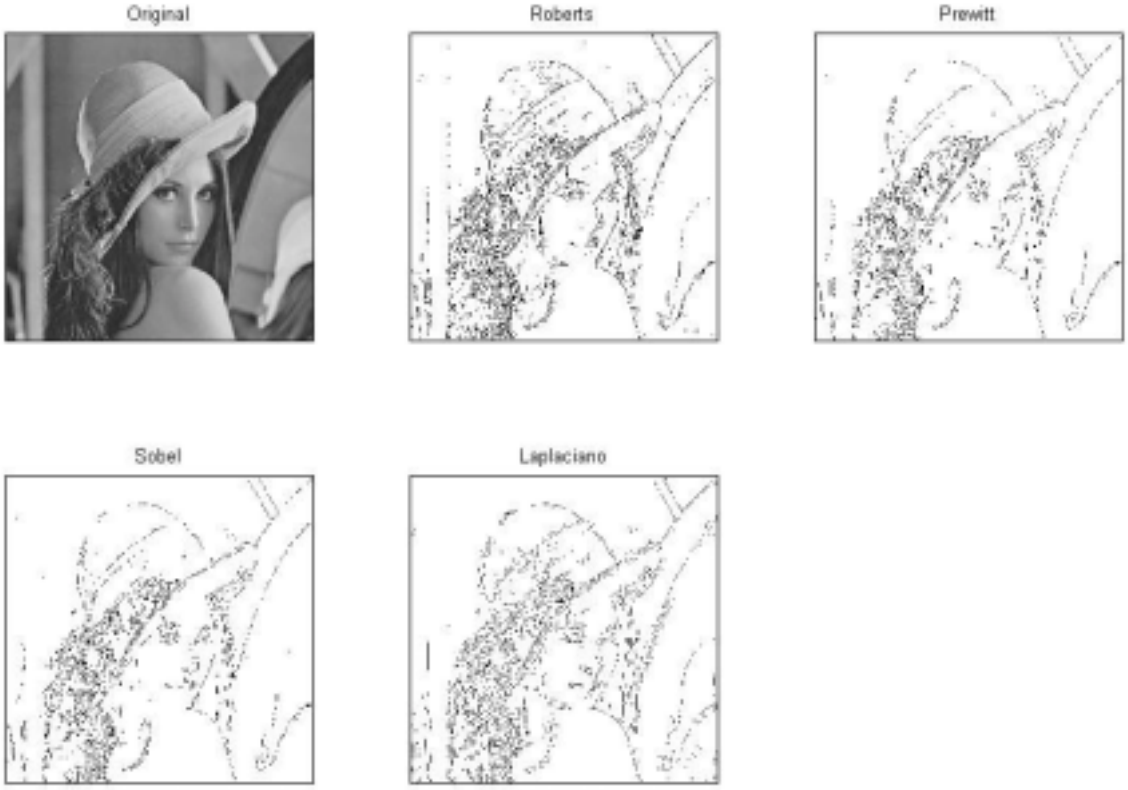


Figura 39. Ejemplos de detección de contornos.

La detección de contornos se encuentra implementada en Matlab en el comando *edge*.