Operaciones Básicas

Rodrigo Javier Herrera García *

Las imágenes digitales, consideradas como arreglos de datos, pueden participar en operaciones sencillas que permitirían obtener nuevas imágenes o imágenes modificadas. Las operaciones básicas contempladas aquí son las aritméticas y las lógicas con las cuales se pueden obtener correcciones o efectos que amplían el espectro de resultados al modificar las intensidades de las imágenes. De hecho, algunas de estas operaciones se utilizan en la modificación de la intensidad cuando la iluminación de la escena no es uniforme en el momento de la captura. O cuando se requiere interactuar sólamente con algunas regiones de una imagen. O cuando se requiere mezclar imágenes para obtener efectos útiles en la visualización de los resultados. En esta sección se tratarán estas operaciones y los efectos que con ellas se pueden lograr.

1. Operaciones Aritméticas

Dos imágenes pueden ser sumadas, restadas, multiplicadas o divididas para obtener una tercera imagen compuesta. Estas operaciones son fundamentales en el procesamiento de imágenes porque pueden ser parte del propósito de crear imágenes compuestas o como parte de un proceso más elaborado. Es, ciertamente, esta última la opción que será utilizada en algunos procesos que se estudiarán más adelante. Aquí solamente se tratarán las operaciones sin considerar aplicaciones específicas.

1.1. Adición de imágenes

La operación de *suma* de las imágenes $f_1(x, y)$ y $f_2(x, y)$ para obtener la imagen $f_s(x, y)$ se define como

$$f_s(x, y) = f_1(x, y) + f_2(x, y) \tag{1}$$

donde x y y corresponden a las coordenadas espaciales horizontal y vertical, respectivamente, de las funciones de imagen. Sin embargo, las imágenes digitales se modelan como arreglos bidimensionales de valores de intensidad o color. La suma de imágenes digitales monocromáticas se reduce a la suma de matrices donde los tamaños de los arreglos deben ser iguales. Entonces, la imagen $f_s[m, n]$ como la suma de las imágenes digitales monocromáticas $f_1[m, n]$ y $f_2[m, n]$ de tamaño $M \times N$ se define como

$$f_s[m, n] = f_1[m, n] + f_2[m, n] \tag{2}$$

donde m y n corresponden a la fila y columna, respectivamente, de cada uno de los elementos (píxeles) de los arreglos (imágenes), que están limitados entre L_{min} y L_{max} que corresponden a los niveles de brillo mínimo y máximo posibles. Sin embargo, al sumar dos imágenes, algunos de los niveles de intensidad de la imagen resultante pueden llegar a ser superiores al nivel máximo L_{max} implicando una saturación hacia el blanco. De hecho, el brillo promedio de la imagen resultante es siempre superior al brillo promedio de cada una de las imágenes de la suma. La **Figura 1** muestra dos imágenes y su suma. Observe cómo la imagen resultante se satura hacia el blanco en aquellas regiones donde las intensidades superan el nivel de brillo máximo L_{max}.

Para evitar que la intensidad de los píxeles llegue a sobrepasar el nivel máximo, una mejor opción es hacer que la suma de las imágenes se haga en forma ponderada. Si se considera un factor de ponderación para cada imagen, denotado como a, y la suma de las ponderaciones es uno, la imagen resultado de la suma no presentará saturación. Además, cuando se incluyen los factores de ponderación, la participación

*Profesor del Proyecto Curricular de Ingeniería Electrónica, Facultad de Ingeniería, Universidad Distrital Francisco José de Caldas, Bogotá, D.C., Colombia.

(a) (b) (c)

Figura 1: Ejemplo de suma de imágenes. Las imágenes (a) y (b) se han sumado para obtener la imagen (c). Puesto que la suma es directa, la imagen resultante tiene varias regiones de intensidad saturada.

de cada imagen en la suma puede variarse para que pueda resaltarse alguna de ellas más que la otra. Entonces, ahora la *suma ponderada* de imágenes puede definirse como

$$f_s[m,n] = a_1 f_1[m,n] + a_2 f_2[m,n] \tag{3}$$

donde $a_1, a_2 > 0$ y $a_1 + a_2 = 1$. De manera general, la suma ponderada de K imágenes digitales monocromáticas de tamaño $M \times N$ se define como

$$\begin{aligned} f_s[m,n] &= a_1 f_1[m,n] + a_2 f_2[m,n] + \cdots + a_K f_K[m,n] \\ &= \sum_{k=1}^{K} a_k f_k[m,n] \end{aligned} \tag{4}$$

donde f_s es la imagen resultado de la suma, f_k es cada imagen sumando, a_k es el factor de ponderación para la imagen k y tiene un valor entre cero y uno y $\sum a_k = 1$. La **Figura 2** muestra las imágenes resultantes de la suma de las imágenes (a) y (b) de la **Figura 1** para tres casos diferentes de ponderación. Observe cómo las diferentes ponderaciones hacen que las imágenes participen más o menos en el resultado.

(a) (b) (c)

Figura 2: Ejemplo de suma ponderada de imágenes. Las imágenes (a) y (b) de la **Figura 1** se han sumado para obtener las imágenes mostradas. En (a) $a_1 = 0{,}3$ y $a_2 = 0{,}7$; en (b) $a_1 = 0{,}5$ y $a_2 = 0{,}5$; en (c) $a_1 = 0{,}7$ y $a_2 = 0{,}3$;.

Cuando las imágenes son en color, cada píxel está definido por un vector de tres elementos que corresponden a la composición de las componentes de color. En consecuencia, cada imagen en color está definida como

$$\mathbf{f}[m,n] = \begin{bmatrix} f_r[m,n] \\ f_s[m,n] \\ f_t[m,n] \end{bmatrix} \tag{5}$$

donde $\mathbf{f}[m,n]$ es un vector de tres elementos del espacio de color, r, s y t corresponden a los parámetros que describen la imagen en color y m y n son las coordenadas del píxel. Por ejemplo, para una imagen

en el espacio de color RGB, los tres parámetros corresponden a la composición de rojo, verde y azul del píxel, es decir $r = R$, $s = G$ y $t = B$.

En general, la suma ponderada de imágenes en color es, por consiguiente, una suma vectorial. Sean $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_K$ K imágenes en color de tamaño $M \times N$ píxeles y sean $a_1, a_2, \ldots, a_K$ los factores de ponderación, la suma ponderada de imágenes en color se define como

$$\mathbf{f}_s[m,n] = a_1\mathbf{f}_1[m,n] + a_2\mathbf{f}_2[m,n] + \cdots + a_K\mathbf{f}_K[m,n] \tag{6}$$

donde $m = 1, 2, \ldots, M$, $n = 1, 2, \ldots, N$, $\mathbf{f}_i[m,n]$ es un vector de tres elementos que caracterizan el color del píxel con coordenadas $[m,n]$, $\sum_{k=1}^{K} a_k = 1$ y $\mathbf{f}_s$ es una imagen en color de tamaño $M \times N$ píxeles. Como ejemplo, en la **Figura 3** se puede ver la suma ponderada de dos imágenes en color RGB. La ponderación para la primera suma es $a_1 = 0,3$ y $a_2 = 0,7$. La ponderación para la segunda suma es $a_1 = 0,7$ y $a_2 = 0,3$. Se observa que los colores de las imágenes originales se conservan y predominan aquellos que tienen un mayor peso en la suma.

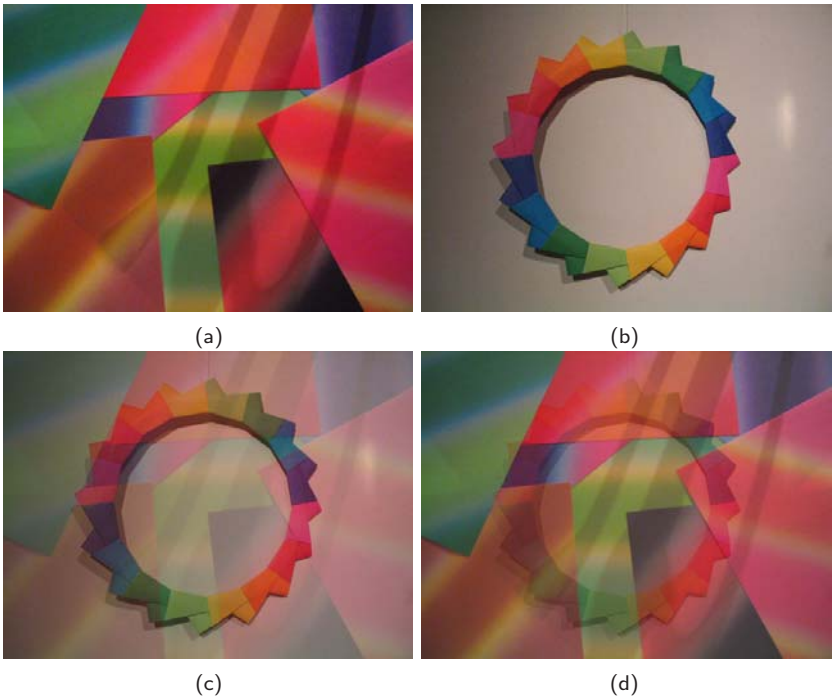


Figura 3: Ejemplo de suma ponderada de imágenes en color. Las imágenes originales en (a) y (b) se han sumado para obtener las imágenes (c) y (d). En (c), la primera imagen se ponderó en 30 % mientras que la segunda en 70 %. En (d) se invirtieron las ponderaciones.

Con la operación de suma se pueden tener diferentes efectos para diferentes espacios de color. En la suma de imágenes, representadas en espacios de color diferentes al espacio RGB, se podrían obtener resultados con colores diferentes al de las imágenes originales. La forma de representación de las imágenes involucran un elemento dentro del vector de cada píxel que representa el color y una modificación implica una variación del color. Un ejemplo es la representación de las imágenes en el espacio HSI, donde el primer elemento del vector que describe el píxel corresponde al color. Si este elemento se modifica, se cambia el color. Luego, el resultado de la suma produciría colores diferentes a los de las imágenes originales.

1.2. Diferencia de imágenes

La operación de *resta* o *diferencia* entre dos imágenes digitales monocromáticas $f_1[m,n]$ y $f_2[m,n]$ de tamaño $M \times N$ está definida como

$$f_d[m,n] = f_1[m,n] - f_2[m,n] \tag{7}$$

donde f_d es una matriz de tamaño $M \times N$ que contiene las diferencias de los píxeles con coordenadas $[m, n]$ ([filas,columnas]) entre las dos imágenes. En contraste con la suma, la resta o diferencia de imágenes no es una operación conmutativa. El resultado de restar dos imágenes $f_1[m, n]$ y $f_2[m, n]$ es diferente si se tiene $f_1[m, n] - f_2[m, n]$ o $f_2[m, n] - f_1[m, n]$.

Por otro lado, en la operación de diferencia también se presenta saturación pero en el otro extremo que el de la suma. La diferencia entre las intensidades de dos píxeles de dos imágenes puede ser negativa lo que indica que no tendría una representación visual[1]. Esto conlleva a considerar todos los valores de cero o inferiores como intensidad de negro. Habría un recorte de intensidades de la imagen diferencia. Estos dos aspectos de la operación de resta o diferencia se pueden ver en la **Figura 4** donde se han considerado como operandos las imágenes de la **Figura 1**.

(a) (b)

Figura 4: Ejemplo de resta o diferencia de imágenes. Las imágenes (a) y (b) de la **Figura 1** se han restado para obtener las imágenes mostradas. En (a) se tiene la diferencia de la primera con la segunda imagen; en (b) se tiene la diferencia de la segunda con la primera. Ambas imágenes presentan saturación al negro.

Para obtener una imagen que muestre la diferencia entre otras dos imágenes, es necesario realizar dos procesos. En primer lugar, se deben trasladar todas los valores de las diferencias de intensidad de los píxeles a valores que se permitan en la visualización (rango de valores de intensidad visible). En segundo lugar, puesto que el rango de valores de intensidad en la diferencia puede ser muy grande o muy pequeño comparado con el rango de intensidades disponible, se debe realizar una escalización de las diferencias para que estas se distribuyan dentro del rango de valores de intensidad visible. Estos procesos corresponden a una transformación lineal de valores, la cual se puede observar gráficamente en la **Figura 5**. Observe cómo los resultados de las diferencias se trasladan al rango de intensidades visibles disponibles: $f_{d_{\min}} \rightarrow L_{\min}$ (negro), $f_{d_{\max}} \rightarrow L_{\max}$ (blanco) y $f_{d_{\min}} < f_d < f_{d_{\max}} \rightarrow L_{\min} < l < L_{\max}$ (niveles de gris).

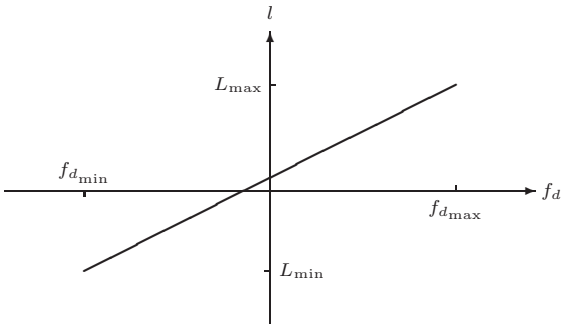


Figura 5: Gráfica de transformación de los valores de una matriz al rango de valores de intensidad para visualización como imagen.

Haciendo la operación de transformación lineal de las imágenes diferencia mostradas en la **Figura 4** se obtienen las imágenes mostradas en la **Figura 6** que han sido trasladadas al rango de intensidades

[1] La representación de las intensidades está dada entre cero y un valor positivo máximo que representan el negro y el blanco, respectivamente. Todos los valores intermedios representan tonalidades de gris.

visibles. Esta presentación permite *ver* el resultado que puede ser una opción importante en el análisis. La transformación de los valores de cada uno de los píxeles puede expresarse en forma generalizada como

$$l = \frac{L_{\max} - L_{\min}}{f_{d_{\max}} - f_{d_{\min}}} (f_d - f_{d_{\min}}) + L_{\min} \tag{8}$$

(a) (b)

Figura 6: Ejemplo de imágenes resta o diferencia con intensidades visibles. Los píxeles de las imágenes (a) y (b) de la **Figura 4** se han trasladado a los valores del rango visible entre negro y blanco.

donde $L_{\max}$ corresponde al valor de intensidad del blanco, $L_{\min}$ corresponde al valor de intensidad del negro, $f_{d_{\max}}$ y $f_{d_{\min}}$ corresponden a los valores máximo y mínimo, respectivamente, de la matriz diferencia y f_d son los valores de cada uno de los elementos de la matriz diferencia.

En algunas ocasiones se está más interesado en cuantificar la diferencia entre las intensidades de los píxeles de dos imágenes que en la imagen diferencia misma. Para visualizar estas diferencias se utiliza la *diferencia absoluta* que es el valor absoluto de la resta. Entonces, si se tienen dos imágenes digitales f_1 y f_2 de tamaño $M \times N$, la diferencia absoluta está dada por

$$f_{da}[m,n] = |f_1[m,n] - f_2[m,n]| \tag{9}$$

para $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$ y donde f_{da} es una matriz de tamaño $M \times N$ que contiene los valores absolutos de las diferencias de cada uno de los píxeles.

Como ejemplo de los resultados de la diferencia absoluta, la **Figura 7** muestra las imágenes de $|f_1[m,n] - f_2[m,n]|$ y $|f_2[m,n] - f_1[m,n]|$ de las imágenes mostradas en la **Figura 1**. Observe que el cambio de orden de los operandos produce el mismo resultado. En efecto, en la magnitud de la diferencia no importa cuál de los operandos va primero o último. La operación es conmutativa.

(a) (b)

Figura 7: Ejemplo de imágenes de resta o diferencia absoluta. En esta operación se hace énfasis en las diferencias sin importar cuál imagen tiene más o menos intensidad o brillo. La imagen (a) corresponde a $|f_1[m,n] - f_2[m,n]|$ mientras que la imagen (b) corresponde a $|f_2[m,n] - f_1[m,n]|$ donde f_1 y f_2 son las imágenes (a) y (b) de la **Figura 1**.

Sin embargo, la utilidad de la diferencia absoluta está en aquellas imágenes a las cuales se les ha hecho alguna modificación y se requiere evaluar dónde se han afectado o en aquellas imágenes que aparentemente

son iguales. Este es el caso, por ejemplo, de las imágenes mostradas en la **Figura 8** que muestra una imagen que ha sido «suavizada» y de la cual se observa dónde afectó más el proceso de suavizado. La diferencia absoluta no tiene en cuenta si los píxeles aumentaron o disminuyeron su brillo sino la magnitud de la variación. Para tener una mejor «observación» de las diferencias, esta última imagen se ha escalado a través de la transformación de intensidades planteada en la ecuación (8).

(a) (b) (c)

Figura 8: Diferencia absoluta entre dos imágenes. La imagen de la escalera (a) se ha «suavizado» a través de un filtro promediador para obtener la imagen (b). La imagen (c) es el negativo de la imagen que muestra la diferencia abosoluta escalada entre las intensidades de las imágenes (a) y (b).

1.3. Multiplicación de imágenes

Si se tienen dos imágenes digitales monocromáticas $f_1[m,n]$ y $f_2[m,n]$ de tamaño $M \times N$, la operación de *multiplicación* se define como

$$f_m[m,n] = f_1[m,n]\, f_2[m,n] \tag{10}$$

para $m = 1,2,\ldots,M$ y $n = 1,2,\ldots,N$. Aunque las imágenes digitales se tienen como arreglos bidimensionales, la operación de multiplicación se hace entre los elementos y no como una multiplicación de matrices. Esta condición no obliga a tener imágenes cuadradas ($M = N$). Las dimensiones pueden ser diferentes ($M \neq N$). No obstante, las imágenes deben tener el mismo tamaño.

En la **Figura 9** se puede ver el producto de las imágenes (mostradas en (a) y (b)). A la imagen resultante se le debe modificar la escala de intensidades debido a que los productos de los brillos pueden superar la intensidad máxima (blanco). Para cambiar la escala de la matriz producto, se utilizó la transformación lineal definida en la ecuación (8). Sin embargo, como se puede observar en la imagen (c), el resultado puede producir una imagen compuesta con brillo menor que el brillo de las imágenes originales. El brillo medio de esta imagen puede modificarse a través de transformaciones lineales o no lineales.

(a) (b) (c)

Figura 9: Ejemplo de multiplicación de imágenes monocromáticas. Los elementos (píxeles) de las imágenes (a) y (b) se multiplicaron para obtener una matriz producto. La imagen (c) se obtuvo después de modificar los valores de los elementos del producto a valores de intensidades.

Cuando las imágenes son en color, la operación de multiplicación se realiza entre los elementos de los vectores que representan los píxeles. Cada elemento dentro del vector puede representar características

diferentes del píxel. Es por esto que la multiplicación de imágenes se restrinja a que el producto se realice entre los elementos del vector y no como un producto vectorial.

Entonces, si $\mathbf{f}_1$ y $\mathbf{f}_2$ representan los arreglos bidimensionales de los vectores (de tres elementos) de dos imágenes en color, de tamaño $M \times N$, y están definidos tal como se hizo en la ecuación (5), la multiplicación de imágenes en color está dada por

$$\begin{aligned}\mathbf{f}_m[m,n] &= \mathbf{f}_1[m,n]\,\mathbf{f}_2[m,n] \\ &= \begin{bmatrix} f_{m_r}[m,n] \\ f_{m_s}[m,n] \\ f_{m_t}[m,n] \end{bmatrix} = \begin{bmatrix} f_{1_r}[m,n]\, f_{2_r}[m,n] \\ f_{1_s}[m,n]\, f_{2_s}[m,n] \\ f_{1_t}[m,n]\, f_{2_t}[m,n] \end{bmatrix}\end{aligned} \tag{11}$$

donde $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$ corresponden a las coordenadas de los elementos de las matrices f_{i_j} cuyos tamaños son de $M \times N$.

En la **Figura 10** se muestran dos ejemplos de multiplicación de dos imágenes en color. El primer ejemplo consiste en la multiplicación en el espacio de color RGB mientras que el segundo consiste en la multiplicación en el espacio de color HSV. Para el primer caso, cada píxel está descrito por un vector que contiene tres elementos que cuantifican la cantidad de rojo, verde y azul, así

$$\begin{aligned}\mathbf{f}_m[m,n] &= [f_{m_R}[m,n], f_{m_G}[m,n], f_{m_B}[m,n]]^T \\ &= [f_{1_R}[m,n]\, f_{2_R}[m,n], f_{1_G}[m,n]\, f_{2_G}[m,n], f_{1_B}[m,n]\, f_{2_B}[m,n]]^T\end{aligned}$$

donde T significa que los vectores son transpuestos. Así mismo, en el segundo caso, cada píxel está descrito por un vector que contiene los valores que cuantifican el tono o matiz, la saturación y la intensidad. El vector de cada píxel es

$$\begin{aligned}\mathbf{f}_m[m,n] &= [f_{m_H}[m,n], f_{m_S}[m,n], f_{m_V}[m,n]]^T \\ &= [f_{1_H}[m,n]\, f_{2_H}[m,n], f_{1_S}[m,n]\, f_{2_S}[m,n], f_{1_V}[m,n]\, f_{2_V}[m,n]]^T\end{aligned}$$

Se puede observar que en las imágenes producto hay variación de los colores originales, especialmente donde intervienen colores con altos brillos. Además, las imágenes producto presentan una reducción en el brillo promedio, lo cual es razonable debido al amplio rango que presenta el producto y a los niveles relativamente bajos que puede dar la multiplicación de los elementos. De hecho, si se considerara que el rango de los elementos fueran valores que estuvieran entre cero y uno, la multiplicación de números con valores menores que uno darán productos más pequeños que los mismos operandos. Esto hace que el brillo promedio de la imagen sea menor.

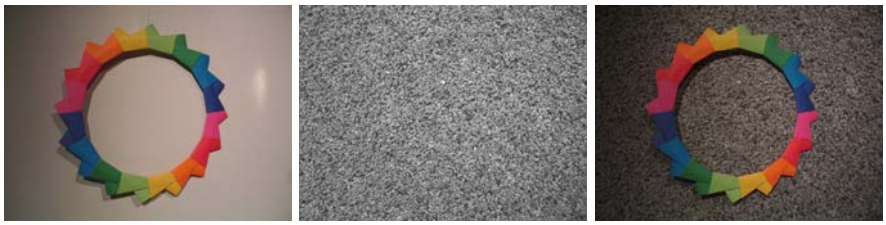
(a) (b)

Figura 10: Ejemplo de multiplicación de imágenes en color. En (a) se observa el resultado de la multiplicación de las imágenes de la **Figura 3** representadas en el espacio de color RGB. En (b) se observa el resultado de la multiplicación de las mismas imágenes representadas en el espacio de color HSV.

Una aplicación común del producto de imágenes es la colocación de texturas en los objetos que componen imágenes sintéticas. Al seleccionarse los objetos apropiadamente a través de máscaras[2], se

[2]La sección 3.1 trata de la generación y utilización de máscaras en las imágenes

pueden colocar diferentes texturas obtenidas a través de otras imágenes. Precisamente la **Figura 11** muestra la imagen de la figura de papel, la imagen de la textura que se va a colocar en el «fondo» y la imagen resultante del producto de las imágenes en las regiones seleccionadas a través de una máscara. Con la adición de la textura, el fondo de la imagen tiene un brillo menor. Sin embargo, por medio de manipulaciones regionales de la imagen producto, se puede modificar el brillo del fondo.



(a) (b) (c)

Figura 11: Ejemplo de adición de texturas. Al fondo de la imagen (a) se le adiciona la textura de la imagen (b) a través de la multiplicación de imágenes. El resultado se puede ver en la imagen (c).

1.4. División entre imágenes

La *división* entre dos imágenes monocromáticas se realiza entre los elementos de los arreglos que las conforman. Entonces, si f_1 y f_2 son los arreglos bidimensionales que representan dos imágenes monocromáticas de tamaño $M \times N$, la división se define como

$$f_r[m,n] = \frac{f_1[m,n]}{f_2[m,n]} \qquad f_2[m,n] \neq 0 \tag{12}$$

donde $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$ que corresponden a las coordenadas de las imágenes. La operación tiene además la restricción de que los elementos del denominador deben ser diferentes de cero. Se produciría un elemento que no puede ser representado numéricamente. Esta condición hace que la operación en la práctica sea poco útil debido a que, en general, la representación del negro se realiza con el valor de cero. Será necesario entonces primero evaluar la imagen del denominador y hacer algún tipo de ajuste para evitar que se presenten ceros en el denominador.

La división entre dos imágenes en color también se realiza entre los elementos de los arreglos que las conforman. Sea entonces $\mathbf{f}_1$ y $\mathbf{f}_2$ los arreglos bidimensionales de los vectores que representan las componentes de color de dos imágenes de tamaño $M \times N$ píxeles, tal como se definió en la ecuación (5). La división entre las dos imágenes es

$$\begin{aligned} \mathbf{f}_r[m,n] &= \mathbf{f}_1[m,n] \div \mathbf{f}_2[m,n] \\ &= \begin{bmatrix} f_{r_r}[m,n] \\ f_{r_s}[m,n] \\ f_{r_t}[m,n] \end{bmatrix} = \begin{bmatrix} f_{1_r}[m,n]/f_{2_r}[m,n] \\ f_{1_s}[m,n]/f_{2_s}[m,n] \\ f_{1_t}[m,n]/f_{2_t}[m,n] \end{bmatrix} \end{aligned} \tag{13}$$

para $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$ y siempre y cuando $f_{2_r}[m,n]$, $f_{2_s}[m,n]$ or $f_{2_t}[m,n]$ sean diferentes de cero. Aquí también, como en el caso de las imágenes monocromáticas, la condición hace que su utilidad sea restringida porque es bastante común que las componentes del espacio de color sean cero.

2. Operaciones lógicas

Las operaciones lógicas básicas son la conjunción, la disyunción y la negación. Se realizan sobre variables binarias que puede tener solo uno de dos estados cuantificados con los valores cero y uno. En consecuencia, las operaciones lógicas son aplicables sólo a imágenes binarias. Las imágenes binarias son aquellas que sus píxeles contienen uno de dos posibles intensidades: negro o blanco. En los sistemas digitales, el estado de una variable puede ser uno o cero pero cada uno de ellos puede estar representado por valores reales

de cantidades físicas dentro de rangos específicos. En las imágenes, cada píxel puede tener uno de dos estados: blanco o negro. Sin embargo, cuál es el estado uno o cero depende de la aplicación. Este tipo de imágenes también se conocen como imágenes en blanco y negro puesto que son las únicas intensidades posibles.

2.1. Negación (NEG)

La negación lógica es la operación más sencilla de las que se aplican sobre variables binarias. Si una variable binaria A puede tener dos estados, estos se representan con los valores cero y uno, donde el valor de uno representa generalmente el valor cierto. Una forma de mostrar la operación es a través de *tablas de verdad*, como la mostrada en la **Tabla 1**. El símbolo que denota la operación de negación es $\neg$.

Tabla 1: Negación lógica NEG: tabla de verdad y equivalente en píxeles.

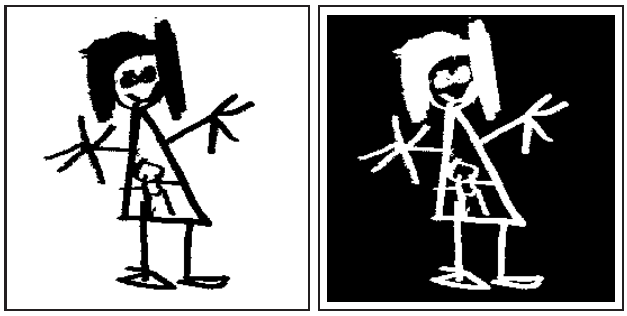
A	$\neg A$
0	1
1	0

$f[m,n]$	$\neg f[m,n]$
blanco	negro
negro	blanco

En las imágenes binarias (están compuestas sólo de dos intensidades: blanco y negro), la operación de negación consiste en cambiar el estado de todos los píxeles. Ese cambio es el que se muestra en la misma **Tabla 1**. Observe que la operación transforma los píxeles blancos en negros y viceversa. Entonces, si a una imagen binaria f, de tamaño $M \times N$, se le aplica la operación de negación, esta se define como

$$f_{\mathsf{NEG}}[m,n] = \neg f[m,n] \tag{14}$$

para $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$. La **Figura 12** muestra un ejemplo sobre una imagen binaria.



(a) (b)

Figura 12: Ejemplo de negación lógica NEG entre imágenes. La imagen binaria (b) es el resultado de la negación lógica NEG de la imagen binaria (a).

2.2. Conjunción lógica (AND)

La conjunción lógica es una operación que se aplica a variables binarias y se conoce como la operación lógica AND. En las imágenes binarias, cada píxel puede tener uno de dos estados: blanco o negro. Si los estados se representan con los números enteros cero o uno, la operación AND (representada a través del operador $\wedge$) entre dos variables binarias A y B se define a través de la **Tabla 2**.

En la misma tabla aparece la definición de la operación cuando se tienen dos imágenes binarias f_1 y f_2. En este caso, la intensidad blanco se ha hecho equivalente con el valor lógico cero y la intensidad negro con el valor lógico uno. Sin embargo, la asignación de cuál es el valor lógico para cada una de las intensidades puede variar dependiendo de la aplicación o de lo que se representa en las imágenes.

Entonces, la operación lógica AND entre dos imágenes binarias f_1 y f_2, cuyos tamaños son de $M \times N$ píxeles, es

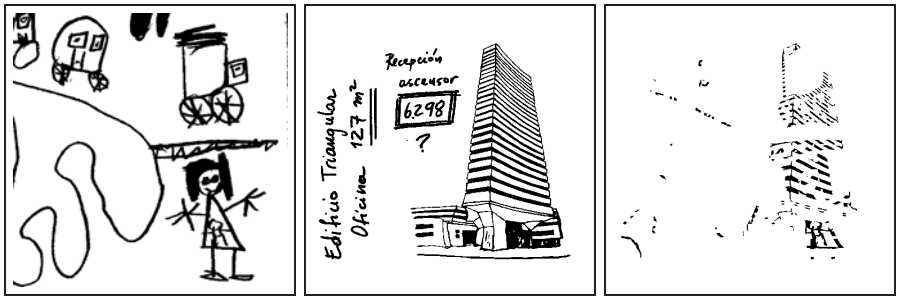
$$f_{\mathsf{AND}}[m,n] = f_1[m,n] \wedge f_2[m,n] \tag{15}$$

Tabla 2: Operación lógica AND: tabla de verdad y equivalente en píxeles.

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

$f_1[m,n]$	$f_2[m,n]$	$f_1[m,n] \wedge f_2[m,n]$
blanco	blanco	blanco
blanco	negro	blanco
negro	blanco	blanco
negro	negro	negro

para $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$. Obsérvese que los tamaños de las imágenes deben ser iguales. La **Figura 13** muestra el resultado de la operación AND entre dos imágenes binarias. La asignación de cada píxel se realizó siguiendo la convención definida en la **Tabla 2**.



(a) (b) (c)

Figura 13: Ejemplo de operación lógica AND entre imágenes. La imagen binaria (c) es el resultado de la operación lógica AND entre las imágenes binarias (a) y (b).

2.3. Disyunción lógica (OR)

La disyunción lógica u operación lógica OR está definida para variables binarias. Para las variables binarias A y B la operación lógica OR está definida como se muestra en la **Tabla 3**.

Tabla 3: Operación lógica OR: tabla de verdad y equivalente en píxeles.

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

$f_1[m,n]$	$f_2[m,n]$	$f_1[m,n] \vee f_2[m,n]$
blanco	blanco	blanco
blanco	negro	negro
negro	blanco	negro
negro	negro	negro

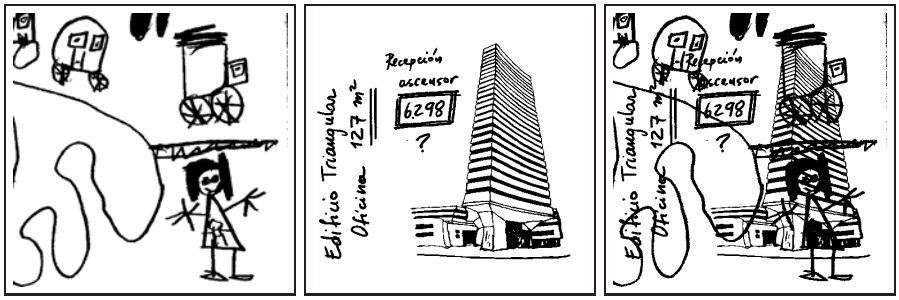
La operación lógica OR (representada a través del operador $\vee$) entre dos imágenes binarias f_1 y f_2, de tamaños $M \times N$, se define como

$$f_{\mathsf{OR}}[m,n] = f_1[m,n] \vee f_2[m,n] \tag{16}$$

para $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$. La **Figura 14** muestra un ejemplo de la operación entre dos imágenes binarias. La operación aplicada a cada par de píxeles sigue la convención mostrada en la **Tabla 3**.

2.4. Disyunción lógica exclusiva (XOR)

La disyunción lógica exclusiva, conocida como la operación lógica XOR, también se realiza sobre variables binarias. Está definida como aquella operación que produce un estado cierto si las variables tienen estados diferentes. Entonces, si se tienen dos variables A y B, que pueden tener uno de dos estados (0 ó 1), la operación lógica XOR (representada con el símbolo $\veebar$) está definida de acuerdo con la **Tabla 4**.



(a) (b) (c)

Figura 14: Ejemplo de operación lógica OR entre imágenes. La imagen binaria (c) es el resultado de la operación lógica OR entre las imágenes binarias (a) y (b).

Tabla 4: Operación lógica XOR: tabla de verdad y equivalente en píxeles.

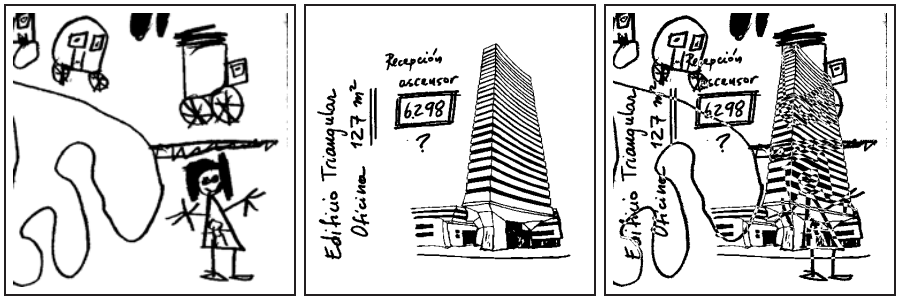
A	B	$A \veebar B$
0	0	0
0	1	1
1	0	1
1	1	0

$f_1[m,n]$	$f_2[m,n]$	$f_1[m,n] \veebar f_2[m,n]$
blanco	blanco	blanco
blanco	negro	negro
negro	blanco	negro
negro	negro	blanco

La **Tabla 4** también define la operación cuando se trata de píxeles de dos imágenes binarias f_1 y f_2 cuyos tamaños son de $M \times N$ píxeles. Entonces, la operación lógica XOR entre dos imágenes binarias está definida como

$$f_{\mathsf{XOR}}[m,n] = f_1[m,n] \veebar f_2[m,n] \tag{17}$$

para $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$. La **Figura 15** muestra un ejemplo de la operación entre las dos imágenes binarias utilizadas anteriormente. La operación aplicada a cada par de píxeles sigue la convención mostrada en la **Tabla 4**. Observe que en este caso, la imagen muestra únicamente las diferencias que existen entre las dos imágenes, no importa si los píxeles pertenecen a algún objeto o al fondo de la imagen.



(a) (b) (c)

Figura 15: Ejemplo de operación lógica XOR entre imágenes. La imagen binaria (c) es el resultado de la operación lógica XOR entre las imágenes binarias (a) y (b).

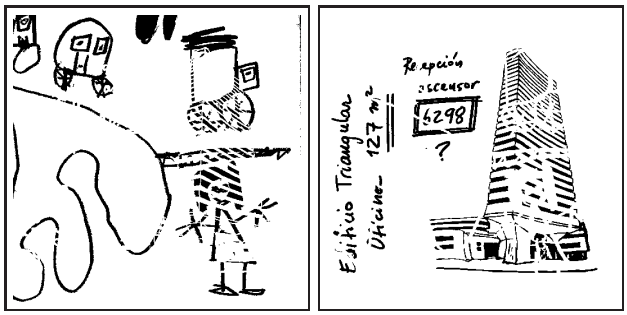
Se debe tener en cuenta que las operaciones lógicas pueden combinarse para obtener operaciones más elaboradas. De hecho, la operación XOR aplicada sobre dos imágenes f_1 y f_2, de $M \times N$ píxeles, puede implementarse con las operaciones NEG, AND y OR de la siguiente manera

$$f_{\mathsf{XOR}}[m,n] = ((\neg f_1[m,n]) \wedge f_2[m,n]) \ \vee \ (f_1[m,n] \wedge (\neg f_2[m,n]))$$

para $m = 1, 2, \ldots, M$ y $n = 1, 2, \ldots, N$.

Las imágenes binarias también pueden verse como conjuntos de píxeles que están distribuidos espacialmente y que representan los objetos en la escena. Por ejemplo, la imagen del dibujo de una niña que

está en la parte (a) de la **Figura 15** contiene objetos como el dibujo de la mujer y los automóviles mientras que en la imagen (b) hay el dibujo de un edificio y texto. Si estos objetos se consideran conjuntos, se puede encontrar el conjunto diferencia entre las dos imágenes. Esta operación se puede ver en la **Figura 16** donde se encontró el conjunto diferencia entre la imagen del dibujo infantil y la imagen del edificio en (a) y al contrario en (b). Observe cómo la primera imagen contiene los píxeles que están en el dibujo infantil pero no están aquellos que también están en el dibujo del edificio. Lo mismo se puede observar en (b) pero invirtiendo el orden de las imágenes. En los dos casos, las imágenes fueron obtenidas a través de operaciones lógicas.



(a) (b)

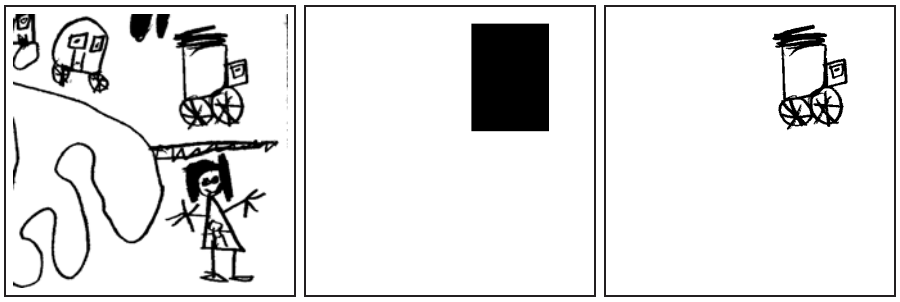
Figura 16: Ejemplo de diferencia de conjuntos en imágenes obtenida a través de operaciones lógicas.

3. Otras operaciones básicas

Las operaciones lógicas tienen el inconveniente que sólo son aplicables a imágenes binarias. Las imágenes monocromáticas y las imágenes en color no podrían participar de sus resultados. No obstante, algunas operaciones sobre imágenes de intensidades o en color pueden asimilarse como operaciones lógicas por su similitud en los resultados. En las siguientes secciones se definirán algunas de estas operaciones.

3.1. Enmascaramiento

La operación de *enmascaramiento* consiste en seleccionar sólo una parte de una imagen. Para realizar la operación es necesario crear una *máscara*, que es una imagen binaria, con valores de cero o uno, donde los píxeles con valor de uno seleccionan el área de interés y los píxeles con valor de cero «enmascaran»el resto. En las imágenes binarias, el enmascaramiento se obtiene sencillamente con la aplicación de la operación lógica AND entre la imagen y la máscara. En la **Figura 17** puede verse un ejemplo donde en (a) se tiene la imagen de la cual se requiere extraer una parte, en (b) se tiene la máscara y en (c) se tiene la imagen enmascarada.



(a) (b) (c)

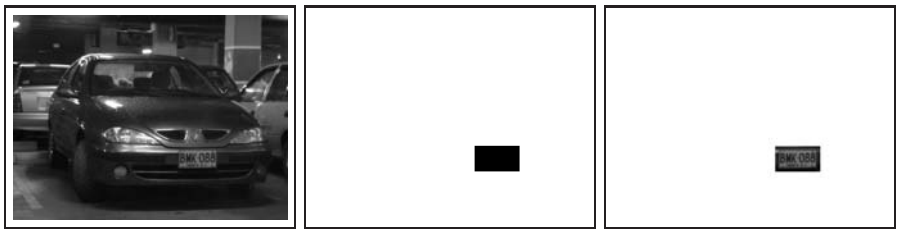
Figura 17: Ejemplo de enmascaramiento de imágenes binarias. De la imagen binaria (a) se obtiene una parte seleccionada por la máscara mostrada en (b). En (c) se tiene la imagen enmascarada.

Donde realmente se obtiene enmascaramientos es en las imágenes de intensidades y en las imágenes en color. Puesto que las imágenes de intensidades se pueden considerar como arreglos bidimensionales, el enmascaramiento de zonas se puede realizar a través de la operación aritmética de multiplicación. La imagen a enmascarar se multiplica por una máscara, que es otra imagen donde los píxeles de las áreas de interés tienen valor de uno y las zonas que se desean eliminar tienen valor de cero. La multiplicación no es entre matrices sino entre los elementos de los arreglos. De aquellos elementos de la imagen que se multiplican por los elementos de la máscara que contienen uno se obtendrán valores de intensidad iguales a los de la imagen; los elementos de la imagen que se multiplican por los elementos de la máscara que contienen cero producirán un valor de cero, eliminando el contenido de intensidad de la imagen. La imagen enmascarada contendrá sólo aquellos píxeles que se seleccionaron con la máscara.

Para el enmascaramiento de una imagen f, de tamaño $M \times N$ píxeles, se crea una máscara h del mismo tamaño con valores de cero en las zonas de la imagen que se requieren enmascarar y valores de uno en el resto. La imagen enmascarada se obtiene por medio de la siguiente operación para cada píxel,

$$f_e[m,n] = f[m,n]\, h[m,n] \qquad \text{para } m = 1,\ldots,M \text{ y } n = 1,\ldots,N \tag{18}$$

La **Figura 18** muestra un ejemplo de enmascaramiento. De la imagen del automóvil se requiere obtener sólo la parte que corresponde a la matrícula que lo identifica. La máscara se crea de manera que seleccione la zona de la imagen que contiene la matrícula de identificación. La imagen enmascarada muestra sólo la parte seleccionada. El resto de la imagen se ha eliminado. (La máscara y la imagen obtenida se han modificado para que el fondo se vea blanco y no negro como resulta originalmente con la intención de resaltar el enmascaramiento.)



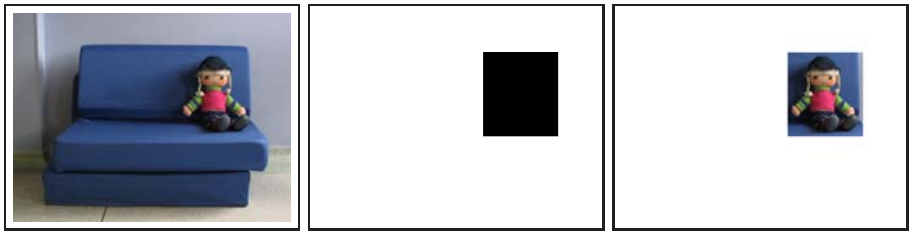
(a) (b) (c)

Figura 18: Ejemplo de enmascaramiento en imágenes de intensidades. En (a) se tiene la imagen original; en (b) se tiene la máscara; en (c) se tiene la imagen enmascarada.

Con las imágenes en color, el procedimiento de enmascaramiento es el mismo que con las imágenes de intensidades pero las operaciones se realizan con los vectores que representan la cromaticidad de cada píxel. Si cada píxel $\mathbf{f}[m,n]$, de una imagen $\mathbf{f}$ de $M \times N$ píxeles, es un vector, se crea una máscara $\mathbf{h}$ cuyos píxeles $\mathbf{h}[m,n]$ también son vectores del mismo tamaño de la imagen a enmascarar. Los elementos de los vectores de la máscara contienen unos o ceros dependiendo de si los píxeles pertenencen o no a las zonas a enmascarar. Después de tener la máscara, se efectúa la operación de multiplicación para cada píxel. La imagen enmascarada se obtiene entonces como

$$\mathbf{f}_e[m,n] = \mathbf{f}[m,n]\, \mathbf{h}[m,n] \qquad \text{para } m = 1,\ldots,M \text{ y } n = 1,\ldots,N \tag{19}$$

La **Figura 19** muestra un ejemplo de enmascaramiento de una imagen en color. La presentación de la máscara se ha modificado a su negativo y a la imagen enmscarada se le ha modificado el fondo para que aparezca blanco. Sin embargo, el efecto del enmascaramiento se puede observar con claridad. Aunque la forma de la máscara utilizada en el ejemplo es rectangular, cualquier forma es posible. Algunas formas pueden depender de los objetos que contiene la imagen misma. De hecho, algunas formas que adquieren las máscaras pueden obtenerse a partir de procesamiento sobre la imagen a enmascarar misma. El enmascaramiento se convierte en una herramienta útil en el análisis de las imágenes tanto para interpretación de su contenido como para medición de parámetros. Se debe entonces considerar diferentes técnicas de obtención de las máscaras que se desarrollarán en los capítulos siguientes.



(a) (b) (c)

Figura 19: Ejemplo de enmascaramiento en imágenes en color. En (a) se tiene la imagen original; en (b) se tiene la máscara (en color); en (c) se tiene la imagen enmascarada.

3.2. Recorte

La operación de *recorte* de imágenes consiste en eliminar aquellas zonas contiguas con los bordes de las imágenes que no son necesarias. Un caso particular es el de las imágenes enmascaradas que pueden contener zonas sin información y que ocupan espacio. En general, en las imágenes que se representan a través de arreglos bidmensionales, la operación de recorte se realiza definiendo las coordenadas del rectángulo que contiene el segmento de imagen que se desea preservar asignando los píxeles a un nuevo arreglo que tiene el mismo tamaño del rectángulo.

Entonces, si f es una imagen representada por un arreglo bidimensional[3] de $M \times N$ píxeles, y se requiere recortar un segmento rectangular de una imagen cuyas coordenadas van de m_1 a m_2 y de n_1 a n_2, se genera el arreglo bidimensional f_r, de tamaño $K \times L$, que representa la imagen recortada de la siguiente forma:

$$f_r[k,l] = f[m_1 + k - 1, n_1 + l - 1] \tag{20}$$

para $k = 1, 2, \ldots, K$ y $l = 1, 2, \ldots, L$ donde $K = m_2 - m_1 + 1$ y $L = n_2 - n_1 + 1$. Como ejemplo, la **Figura 20** muestra las imágenes recortadas de las imágenes obtenidas en la operación de enmascaramiento, mostradas en las **Figuras 17 y 18**. Obsérvese que la resolución de las imágenes recortadas es menor por la reducción de la cantidad de píxeles.

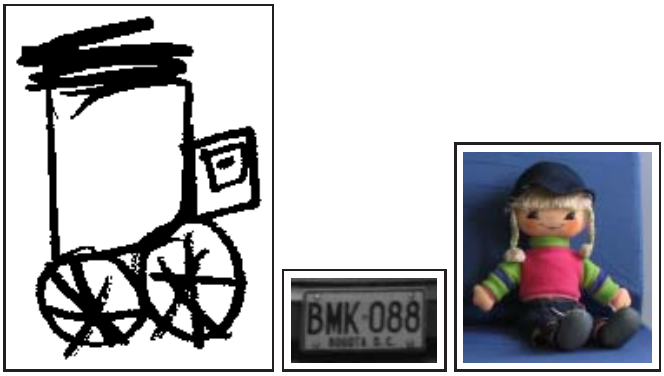


Figura 20: Imágenes recortadas de los resultados del enmascaramiento de las imágenes de las **Figuras 17, 18 y 19**.

Cuando la imagen a recortar es en colores, la operación de recorte es similar al mostrado anteriormente. La diferencia reside en que la asignación de la imagen recortada ahora es de vectores que corresponden a la cromaticidad de los píxeles. Arreglando la ecuación (20) para imágenes en color, se tiene

$$\mathbf{f}_r[k,l] = \mathbf{f}[m_1 + k - 1, n_1 + l - 1] \tag{21}$$

para una imagen $\mathbf{f}$ de $M \times N$ píxeles, una imagen recortada $\mathbf{f}_r$ de $K \times L$, un segmento de imagen que va m_1 a m_2 y de n_1 a n_2, $k = 1, 2, \ldots, K$ y $l = 1, 2, \ldots, L$ donde $K = m_2 - m_1 + 1$ y $L = n_2 - n_1 + 1$. En la **Figura 20**, parte (c), se muestra la imagen recortada de la **Figura 19**.

[3]Tanto las imágenes binarias como las de intensidades se representan como arreglos bidimensionales.

Bibiografía

[1] Gonzalo Pajares Martinsanz y Jesús M. De La Cruz García, *Visión por Computador: Imágenes Digitales y Aplicaciones*, Alfaomega Grupo Editor, S.A. de C.V., México, D.F., 2002.

[2] John C. Russ, *The Image Processing Handbook*, Fourth Edition, CRC Press LLC, Boca Ratón, Florida, USA, 2002.

[3] Marcos Faúndez Zanuy, *Tratamiento Digital de Voz e Imagen y Aplicación a la Multimedia*, Alfaomega Grupo Editor, S.A. de C.V., México, D.F., 2001.

[4] Rafael C. González and Richard E. Woods, *Digital Image Processing*, Second Edition, Prentice-Hall, Inc., New Jersey, USA, 2002.

[5] Steven W. Smith, *The Scientist and Engineers Guide to Digital Signal Processing*, Second Edition, California Technical Publishing, San Diego, California, 1999.

[6] William K. Pratt, *Digital Image Processing*, Second Edition, John Wiley & Sons, Inc., New York, USA, 1991.