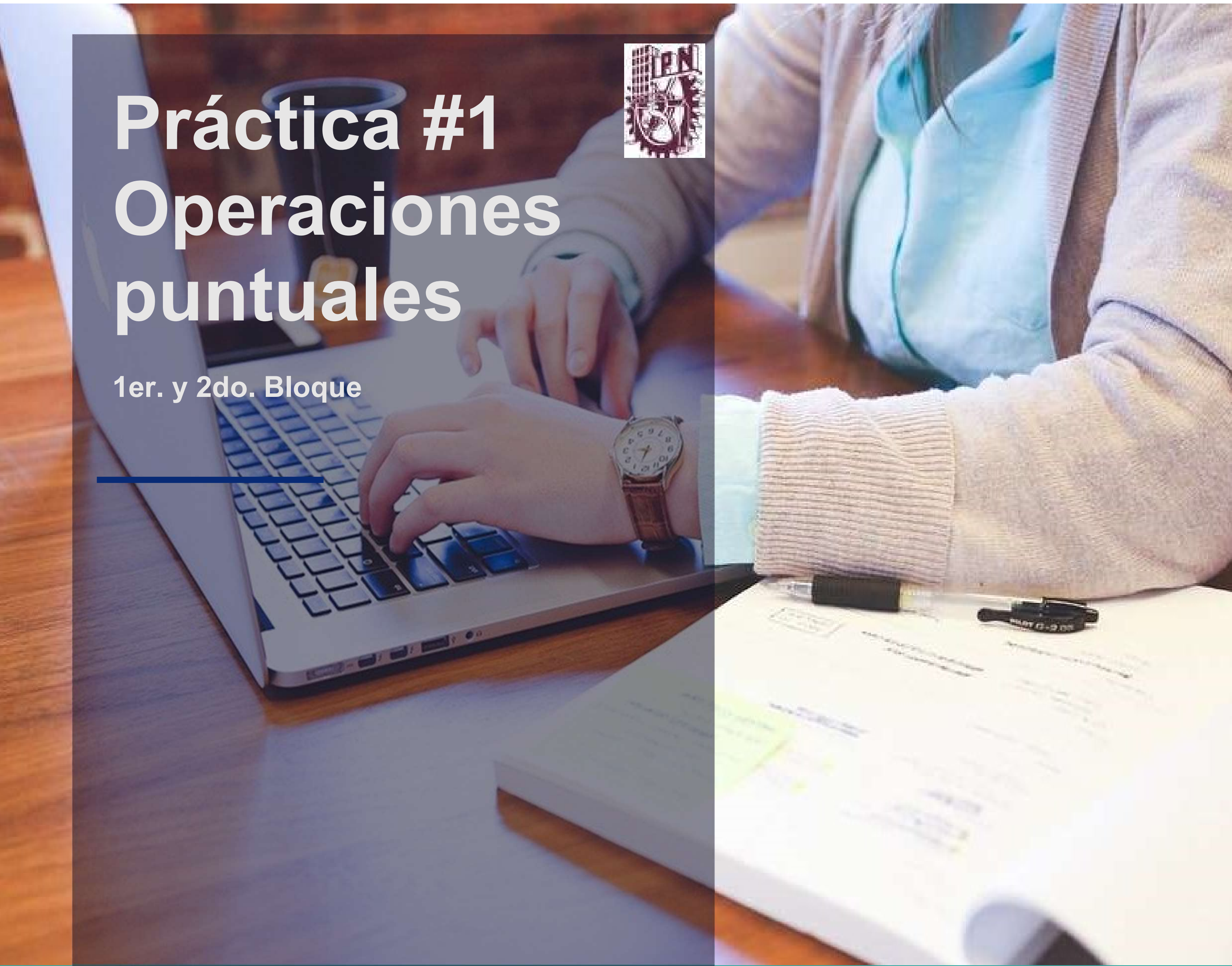

Práctica #1 Operaciones puntuales

1er. y 2do. Bloque

13 NOVIEMBRE

Análisis de Imágenes
María Elena Cruz Meza

ESCOM

Práctica #1 Analisis de Imágenes

Objetivo: Mostrar la habilidad para identificar los parámetros y el manejo e implementación de los algoritmos de operaciones punto a punto a imágenes binarias, en grises y en color.

Tareas del 1er bloque (objetivos específicos):

- ✓ Abrir y cerrar una imagen (operación identidad)
- ✓ Transformación de una imagen en niveles de gris (elegir algún método)
- ✓ Operación negativo o inverso a una imagen (binaria, en grises y en color)

Tareas del 2do bloque:

- ✓ Operaciones relacionales entre dos imágenes (+,-)
- ✓ Operaciones lógicas entre dos imágenes (OR, AND)
- ✓ Operación de binarización y su inverso

Instrucciones

- **Conocimientos previos**
  a) Habilidades en la implementación de vectores y matrices, así como graficar histogramas.
- **Sistema operativo y lenguaje de programación**
  a) Elegir el sistema operativo que vaya a utilizar para desarrollar las prácticas, Windows o algún distro de Linux.
  b) Elegir el lenguaje de programación en el que tengas mayor conocimiento o manejo y que sea sencilla la inclusión y manipulación de librerías para manipular histogramas.
- **Entrega de evidencia**
  a) Medio para su entrega: Entregar la tarea en la plataforma Microsoft Teams una vez terminada toda la práctica (dos algoritmos). Elaborar un informe del desarrollo de esta práctica en un archivo con formato de video (consultar rúbrica).
  b) Crear un archivo en formato de video, elija la herramienta de su preferencia y subirlo a alguna plataforma para compartir el link para visualizarlo (drive, Dropbox, youtube, etc). El archivo debe enviarse con una nomenclatura específica y con el siguiente contenido:
    - Nombre al archivo como: Si se elige una plataforma donde se tiene acceso al archivo, este deberás nombrarlo como: P1B-ABC-.FOR, donde “P1B” indica el número de práctica reportada, ABC corresponderá a iniciales de tu nombre (empezando por el apellido), y “.FOR” es el formato correspondiente al archivo tipo video*, etc.* Si optas por compartirlo en youtube o alguna herramienta similar, entonces el nombre deberá ser: “Practica #1-ABC– Clasificador Bayesiano”, donde ABC corresponde a las iniciales de tu nombre. Como solo es para fines académicos se sugiere compartirlo en forma privada, pero, tú decides su publicación libre.
    - Portada: Es importante agregar algunos datos al inicio del vídeo: identidad politécnica (Nombre o logos de la unidad académica: IPN y Escom), Nombre de la

unidad de aprendizaje, Número de la práctica, Objetivo de la práctica, Nombre de quien presenta, es opcional mostrar el Nombre del(a) facilitador(A) (profesora) y Fecha de entrega.

- Marco teórico: El vídeo debe incluir una breve información formal que sustenta el trabajo que se desarrolla (para este caso puede tomar como base o referencia lo que se anexa al final de este documento, notas del curso facilitadas en la plataforma o consultar algún autor), si consulta alguna página, esta debe ser validada por alguna universidad u organismo educativo (papper o artículos, cursos en línea, presentaciones de docentes o investigadores, libros gratuitos en línea o por alguna editorial, etc.).
- Programa y pruebas: En el vídeo debe explicarse que lenguaje de programación es utilizado para implementar el algoritmo, explicar los segmentos de código principales, y posteriormente el funcionamiento de este con al menos dos ejemplos
  - En caso de no terminar todos o bien crea que aun tenga fallas en la programación, presentar los que se hayan realizado. Al entregar el vídeo también anexa un archivo en formato PDF o TXT, con el código completo para su revisión y ponderación respectiva. El nombre de este archivo deberá nombrarse como el archivo de vídeo
- Créditos: Agregar las referencias consultadas para el desarrollo de la actividad.
- La duración del vídeo es recomendable entre 5 a 10 minutos.

Prácticas relacionadas con las Unidades I y II con los temas: Creación de imágenes digitales y su almacenamiento, y operaciones punto a punto

Fundamentos

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

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

$$\mathbf{I}' = f(\mathbf{I}). \qquad (1)$$

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

sí $x = I[i, j] \in y = I'[i, j]$, donde $y = f(x)$.

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

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

R= [i1…i2, j1 … j2],

es el siguiente:

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

fig. 1. Algoritmo básico para transformar una región de una imagen I bajo *f*.

Consideraciones

Supongamos el caso en que: i1 = 0, i2 = M (donde M = Imagen.Ancho-1), j1 = 0, N (donde N = Imagen.Alto-1); el proceso modificará a toda la imagen. Notemos que al cambiar *f* la transformación será diferente. Es decir, si definimos la composición de transformaciones de la manera habitual, tendremos que:

$$f_1 \circ f_2 (I) = f_1(f_2 (I)). \qquad (2)$$

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

$$f_1 \circ f_2 (I) \neq f_2 \circ f_1 (I). \qquad (3)$$

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

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

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

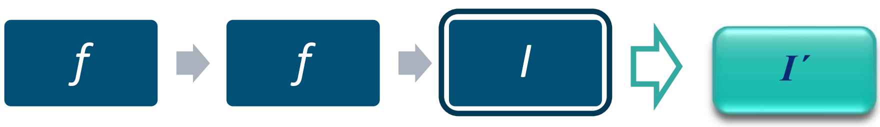

fig. 2. Representación gráfica de la composición de procesos.

Operaciones Elementales

La operación más simple es la *Identidad*, ésta deja a la imagen igual. Esta operación suele usarse para realizar por ejemplo copias de una imagen, notemos que la imagen copiada es la que podemos estar transformando y la imagen original la dejamos intacta, hacer esto es una buena práctica, recordemos que una vez transformada nuestra imagen, no hay transformación inversa que nos permita recuperar la imagen original. Veamos como es el proceso para invertir una imagen.

Sea **I** una imagen en colores, con un dominio [0, L] para los valores del tono de cada canal para los pixeles, en la representación RGB estándar, de ancho *M* y alto *N*. La función correspondiente es: $\mathbf{y} = \mathbf{x}$, de donde $f(\mathbf{x}) = \mathbf{x}$. Si representamos esta función de manera gráfica visualizaremos una ecuación de mapeo lineal simple (fig. 3). Esta función nos indica que el tono *w* es mapeado al tono *w'*.

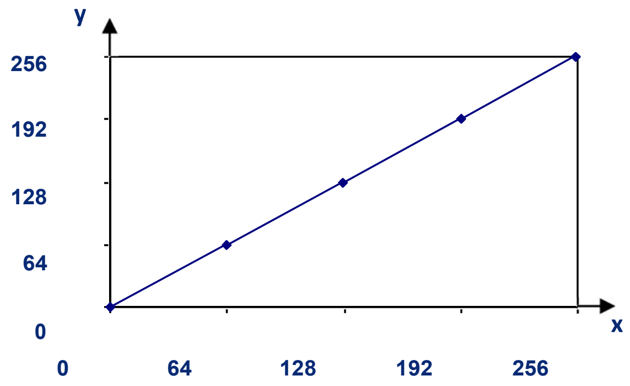

Fig. 3. Transformación identidad entre las variables: $y = f(x) = x$, (p=8, L =255)

Negativo de una imagen

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

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

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

Veamos un ejemplo de esta transformación a una imagen en tonos de gris (r=g=b=z), donde el negativo de un pixel $x = (z, z, z)$ será $\mathbf{x'} = (\sim z, \sim z, \sim z)$, este efecto se muestra en la figura 4-a y 4-b. En la primera se muestra el proceso sobre dos imágenes de ejemplo, una monocroma o binaria y otra en tonos de gris. Por su parte, para obtener el negativo de una imagen en color, debes transformar cada canal RGB.

 a-1 imagen binaria	 a-2 aTransformación inversa de a)

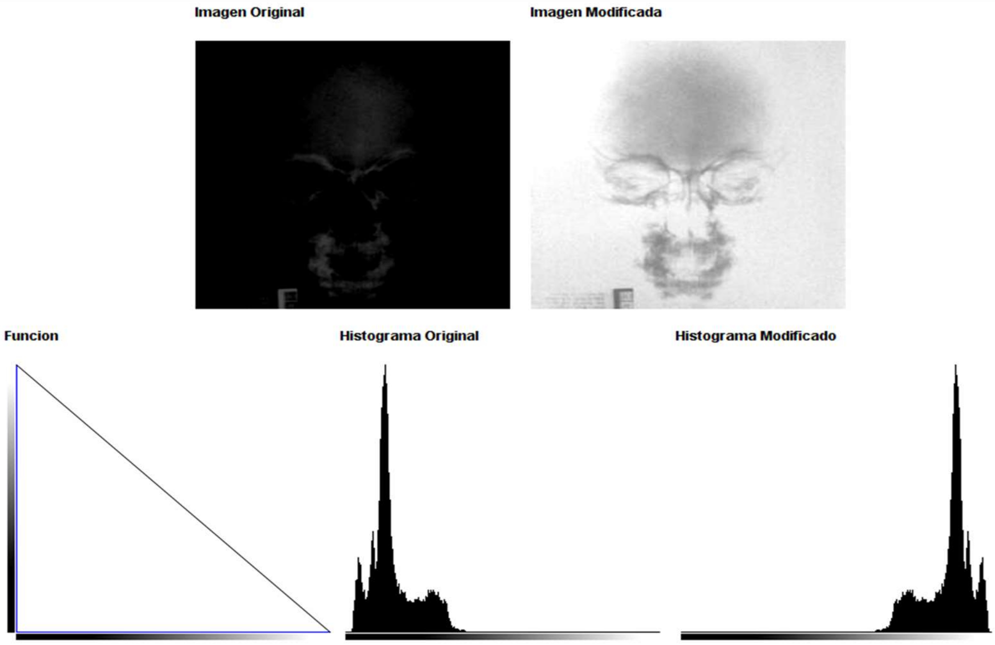

Imagen 4. Imágenes con su respectivos transformación inversa

Conversión de una imagen en color a una imagen en niveles de gris

Una transformación que generalmente es muy útil aplicar a una imagen es la de transformarla de color en niveles de gris (imagen 5), usualmente el método más sencillo es utilizar el esquema de promedio ponderado, es decir, sea x = (r, g, b) un pixel de la imagen I, entonces su nivel de intensidad o brillo (gris) en ese punto se puede hallar simplemente como:

$$X = \frac{R+G+B}{3}$$

Sin embargo hay otras maneras, las estandarizadas por la IEEE, debido a la respuesta del ojo al espectro visible. El ojo reconoce los patrones de iluminación en color en las siguientes proporciones para cada componente: Rojo:30%, Verde:59% y Azul:11%. Notemos que el ojo humano percibe distintas intensidades de luz en función del color que se observe, se dice que es más sensible al verde y al rojo que al azul.

Esto se puede modelar, y la expresión matemática del fenómeno es:

Y = R*0.3+G*0.59+B*0.11
Donde:
Los factores de ponderación de cada componente de color nos indican la sensibilidad del ojo humano a las frecuencias del espectro cercanas al rojo, verde y azul. Así, obtenemos el pixel de color gris con la iluminación adecuada para que nuestro ojo lo perciba como un mejor equivalente a su versión en color

Imagen 5. Imagen en color y su transformación en grises

Operaciones puntuales entre dos imágenes binarias

Ejemplo.
Sean las imágenes A y B que se muestran en la figura 2. Y suponiendo el rango de valores de intensidad (niveles de brillo o gris ideales presentes en ambas imágenes), para este caso el ng [0,1], es decir, el ng es negro o blanco.

$$A = \begin{array}{cc} \begin{array}{ccc} 0 & 1 & 2 \end{array} & \\ \begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 0 & 1 & 0 \\ \hline 1 & 1 & 1 \\ \hline \end{array} & \begin{array}{c} 0 \\ 1 \\ 2 \end{array} \end{array} \qquad B = \begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 0 & 1 \\ \hline \end{array}$$

Figura 2. Imágenes binarias A y B, respectivamente

NOTA: Todos los operadores lógicos tienen sentido cuando al menos una de las imágenes es binaria, recordemos que las imágenes binarias contienen píxeles cuyo valor son:
Negro (0) = FALSE
Blanco (1 ó 255) = TRUE

también notaremos que las imágenes binarias son equivalentes (en su mayoría) a los operadores booleanos.

Objetivo

a) Aplicar las operaciones aritméticas suma y la diferencia entre las imágenes binarias A con B.
b) Aplicar las operaciones lógicas OR y AND entre las imágenes binarias A con B.

Solución

Inciso a) Obtención de los operadores SUMA y DIFERENCIA de A con B

Para obtener la operación SUMA de A con B, consideremos los puntos (x=0, y=0) y (x=1, y=2), se procede operando píxel a píxel, y consideramos el procedimiento efectuado en la suma de imágenes en ng antes visto:

$$Y_{X=0,\ \ y=0} = \left(a_{0,0} +\ b_{\ 0,0}\right)/2 = (1 +\ 1)/2 = 1$$

$$Y_{X=2,\ \ y=0} = \left(a_{2,1}\ or\ \ b_{\ 2,1}\right) = \frac{1+0}{2} = 0.5 \approx 1$$

$$\text{A}=\begin{array}{c|c|c|c} 0 & 1 & 2 & \\ \hline 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 2 \end{array} \qquad \text{B}=\begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 0 & 1 \\ \hline \end{array}$$

La operación para estos dos píxeles queda:

$$\text{S=(A + B)=}\begin{array}{|c|c|c|} \hline 1 & & \\ \hline & & \\ \hline & 0 & \\ \hline \end{array}$$

Para obtener el operador DIFERENCIA de A con B, consideremos los puntos (x=0, y=0) y (x=1, y=2), se procede operando píxel a píxel y se sigue el esquema de la operación diferencia usada en las operaciones entre imágenes en ng antes ya visto

$$Y_{X=0,\ \ y=0} = \left(a_{0,0} -\ \ b_{\ 0,0}\right) = 1 -\ 1 = 0$$

$$Y_{X=1,\ \ y=2} = \left(a_{1,2} -\ \ b_{\ 1.2}\right) = 1 -\ 0 = 1$$

$$Y_{X=0\ \ y=1} = \left(a_{0,1} -\ \ b_{\ 0,1}\right) = 0 -\ 1 = 0$$

$$\text{A}=\begin{array}{c|c|c|c} 0 & 1 & 2 & \\ \hline 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 2 \end{array} \qquad \text{B}=\begin{array}{|c|c|c|} \hline 1 & 0 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 0 & 1 \\ \hline \end{array}$$

La operación para estos dos píxeles queda:

$$\text{S=(A + B)=}\begin{array}{|c|c|c|} \hline 0 & & \\ \hline 0 & & \\ \hline \end{array}$$

	1	

Finalmente, las operaciones aritméticas SUMA y DIFERENCIA quedan de la forma:

S=(A + B)=

1	0	1
1	1	1
1	1	1

R=(A - B)=

0	0	0
0	0	0
0	1	0

Inciso b) Obtención de los operadores OR y AND de A con B

Para obtener la operación OR de A con B, consideremos los puntos (x=0, y=0) y (x=1, y=2), solo que ahora se procede operando píxel a píxel

$$Y_{X=0,\ \ y=0} = \left(a_{0,0}\ or\ \ b_{\ 0,0}\right) = 1\ or\ 1 = 1$$
$$Y_{X=2,\ \ y=0} = \left(a_{2,1}\ or\ \ b_{\ 2,1}\right) = 1\ or\ 0 = 1$$

A=

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

B=

1	0	1
1	1	1
1	0	1

La operación para estos dos píxeles queda:

O=(A OR B)=

1		
	1	

Para obtener el operador AND de A con B, consideremos los puntos (x=0, y=0) y (x=1, y=2), solo que ahora se procede operando píxel a píxel

$$Y_{X=0,\ \ y=0} = \left(a_{0,0}\ and\ \ b_{\ 0,0}\right) = 1\ and\ 1 = 1$$
$$Y_{X=2,\ \ y=0} = \left(a_{2,1}\ and\ \ b_{\ 2,1}\right) = 1\ and\ 0 = 0$$

A=

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

B=

1	0	1
1	1	1
1	0	1

La operación para estos dos píxeles queda:

O=(A AND B)=

1		
	0	

Finalmente, las operaciones lógicas OR y AND quedan de la forma:

O=(A or B)=

1	0	1
1	1	1
1	1	1

Y=(A and B)=

1	0	1
0	1	0
1	0	1

Operaciones puntuales entre dos imágenes en niveles de gris

Sean las imágenes A y B que se muestran en la figura 2. Y suponiendo el rango de valores de intensidad (niveles de brillo o gris ideales presentes en ambas imágenes), para este caso el ng [0-9], es decir, el ng varía entre mi= 0 y Ma= 9.

$$A=\begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{array}{|c|c|c|}\hline 5 & 2 & 3 \\ \hline 4 & 5 & 6 \\ \hline 7 & 8 & 9 \\ \hline \end{array} \end{array}\begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array} \qquad B=\begin{array}{|c|c|c|}\hline 6 & 4 & 4 \\ \hline 6 & 9 & 8 \\ \hline 5 & 8 & 0 \\ \hline \end{array}$$

Figura 2. Imágenes A y B, respectivamente

Objetivo

a) Aplicar las operaciones aritméticas suma y la diferencia entre las imágenes A con B.
b) Aplicar las operaciones lógicas OR y AND entre las imágenes A con B.

Solución

Inciso a) Obtención de los operadores suma y diferencia de A con B

Para obtener la suma de A con B, consideremos dos puntos (x=0, y=0) y (x=2, y=1) como ejemplo:

$$s_{X=0,\ Y=0} = \frac{(a_{0,0} + b_{0,0})}{2} = \frac{(5+6)}{2} = 5.5 \approx 5$$

$$s_{X=2,\ Y=1} = \frac{(a_{2,1} + b_{2,1})}{2} = \frac{(6+8)}{2} = 7$$

Se usa la media porque esto es lo conveniente, evita la sobresaturación, como resultado, las imágenes son semitransparentes (al 50%).

La operación aritmética de la suma para estas coordenadas queda de la forma:

T=

-1	-2	-1
-2	-4	-2
2	0	9

$s_{x=0,y=0} =$

5		
		7

Otra forma de operar la suma entre imágenes es definir la media ponderada. Y normalmente se usa para crear una transición suave entre imágenes (o vídeos).

Media ponderada: R(x,y):= a·A(x,y) + (1-a)·B(x,y). Probar con a = 0,25, 0.50 ó 0.75

- Otra variante común permite adicionar una constante específica a cada píxel:

$$\boldsymbol{f(x,y) = f(x,y) + C}$$

- Para obtener la diferencia de A con B, para ello consideremos los mismos puntos que en el ejemplo utilizado en la suma (x=0, y=0) y (x=2, y=1):

$$\boldsymbol{r_{x,y} = k\left(a_{x,y} - b_{x,y}\right)}$$

Aquí es preciso definir k para que haga este proceso, y en este caso debemos considerar lo siguiente:

1. Obtener la matriz T cuyos elementos están dados por

$$t_{x,y} = \left(a_{x,y} - b_{x,y}\right)$$

2. Calcular el máximo y el mínimo de T: M = max { T } y m = min { T }
3. Obtener $r_{X,Y} = k(t_{x,y})$

$$\boldsymbol{r_{X,Y} = k(t_{x,y}) = round\left[\left(\,(t_{x,y}) - m\right) / \left(M - m\right)\left(M_a - m_i\right)\right]}$$

- Resolviendo estos tres puntos, tenemos:

Paso 1) La matriz T completa queda de la forma,

T=

-1	-2	-1
-2	-4	-2
2	0	9

Paso 2) Los valores máximo y mínimo resultan ser: M=9 y m=-4
Siendo los máximos y mínimos del rango [0,9]: Ma=9 y mi=-0

NOTA: Recordemos que M y m, son los valores máximos y mínimos encontrados una vez que realizamos la transformación para la diferencia entre A con B, (A-B).

Por su parte, Ma y mi, se refiere a los valores máximos y mínimos de gris que podemos manipular, que fue definido al inicio en el problema. Notemos que de forma real manejamos el rango completo que va de [0, L-1] ó [0, 255], y en el caso de nuestro ejemplo, el rango especificado es de [0, 9].

Paso 3) Ahora debemos aplicar a cada píxel de la matriz T, la ecuación:

$$r_{x,y} = k(t_{x,y}) = round\,[(\,(t_{x,y}) - m)\;/\;(M - m)\;(M_a - m_i)]$$

Probemos con (x=0, y=0):

$$r_{x=0,y=0} = k(t_{x=0,y=0}) = round\,[\,((-1-(-4))\;/\;(9-(-4))\,)(9-0)]$$

$$r_{x=0,y=0} = k(t_{x=0,y=0}) = round\,[(\,3\;/\;13)\;(9)] = round\,[(\,0.23076923)\;(9)]$$

$$r_{x=0,y=0} = k(t_{x=0,y=0}) = round\;(2.07692308) = 2$$

La operación aritmética de la resta en la coordenada x=0, y=0, queda de la forma:

T=

-1	-2	-1
-2	-4	-2
2	0	9

$s_{x=0,y=0} =$

2		

Finalmente, las operaciones de la suma y la diferencia entre las dos imágenes quedan de la forma:

S=(A+B)=

5	3	3
5	7	7
6	8	4

R=(A-B)=

2	1	2
1	0	1
4	3	9

Otras forma de operar la resta entre imágenes es una operación similar a la suma, ya hemos notado que la diferencia radica en evitar los valores negativos en el resultado de la operación, veamos:

- $f(x,y) = |x - y|$
- $f(x,y) = \frac{255}{2} + \frac{(x-y)}{2}$
- $f(x,y) = \begin{Bmatrix} x-y, & x-y \geq 0 \\ 0, & en\ otro\ caso \end{Bmatrix}$

- A veces, se requerirá restar dos imágenes, manteniendo el rango de salida:

$$R(x, y) = (A(x, y) - B(x, y))/2 + 128$$

  - En ocasiones también puede ser que lo que interesa es conocer la diferencia entre las imágenes. Una solución es tomar valor absoluto de la resta.

$$R(x, y) = abs(A(x, y) - B(x, y))$$

- Otra variante común permite restar una constante específica a cada píxel:

$$f(x, y) = f(x, y) - C$$

Inciso b) Obtención de los operadores OR y AND de A con B

- Para obtener el OR de A con B, consideremos los puntos (x=1 y=1) y (x=2, y=0), solo que ahora se procede operando los píxeles bit a bit, veamos:

$$Y_{X=0,\ y=0} = (a_{0,0}\ or\ \ b_{\ 0,0}) = 5\ or\ 9 = 0101_{\ 2}\ or\ 1001_{\ 2}\ \ = 1101_{\ 2}\ = 13$$

Aquí notemos que se produce un desbordamiento,ya que nuestro rango de nivel de gris es [0,9],por lo que para este tipo de operación debemos proceder a la transformación lineal, aplicando los pasos 2 y 3,desarrollados en el ejemplo de la resta aritmética.

Aplicando el paso 2)

La matriz T completa queda de la forma:

T=

7	6	7
6	13	14
7	8	9

Los valores máximo y mínimo resultan ser: M=14 y m=6
Siendo los máximos y mínimos del rango [0,9]: Ma=9 y mi=-0

Aplicando el paso 3) a la coordenada x= 1, y=1

$$r_{x=1,y=1} = k(t_{x=1,y=1}) = round\ [(\ (t_{x=1,y=1}) - m)\ /\ (M -\ m)\ (M_a -\ m_i)]$$
$$r_{\ x=0,y=0} = k(t_{x=0,y=0}) = round\ [((\ 13 - 6)\ /\ (14 - 6)\ )\ (9 -\ 0)]$$
$$r_{x=0,y=0} = k(t_{x=0,y=0}) = round\ [(\ 7\ /\ 8\ )(9)] = round\ [(\ 0.875)\ (9)] = 7.875$$
$$r_{x=0,y=0} = k(t_{x=0,y=0}) = round\ (7.875) \approx 7$$

El píxel obtenido queda de la forma:

T=

7	6	7
6	13	14
7	8	9

$Y_{1,1} =$

	7	

- Para obtener el AND de A con B, consideremos los puntos (x=0, y=0) y (x=2, y=0), solo que ahora se procede operando bit a bit

$$\begin{aligned} Y_{X=0,\ y=0} &= (a_{0,0}\ and\ b_{0,0}) = 5\ and\ 6 \\ &= 0101_2\ and\ 0110_2 = 0100_2 = 4 \\ Y_{X=2,\ y=0} &= (a_{2,1}\ and\ b_{2,1}) = 3\ and\ 4 \\ &= 0011_2\ and\ 0100_2 = 0000_2 = 0 \end{aligned}$$

Finalmente, las operaciones lógicas OR y AND quedan de la forma:

O=(A or B)=

1	0	1
0	8	9
1	2	3

Y=(A and B)=

4	0	0
4	1	0
5	8	0

Fuentes consultadas

- Gonzalo Pajares Martinsanz & Jesús M. de la Cruz García. Visión por computadora: imágenes digitales y aplicaciones. Ed. Alfaomega Ra-Ma. 2002
- González, RC y Woods, RE Digital Image Processing. Addision-Weslay, USA, 1992.
- Forsyth y Ponce. Computer Vision: A modern approach. Prentice-Hall, New Jersey, 2003.
- J. Parker. Algorithms for image processing and computer vision. John Wiley & Sons ed. (1997), 116-149.