Tratamiento digital de imágenes.

Concepto:

Consideramos una función bidimensional para definir a una imagen, el par de coordenadas **(x,y)** definen un único punto en la matriz y el valor de la función en ese punto **f(x,y)** da el valor de la intensidad luminica en ese punto, donde:

0 £ f(x,y) £ ¥

Esta es la definición de imagen monocromática, y el valor de **f(x,y)** recibe el nombre de nivel de gris dentro de una escala: escala de grises, donde:

f(x,y) = 0 .... ausencia de luz

f(x,y) = L ..... máxima luminosidad (por razones practicas este valor no es infinito.)

negro £ f(x,y) £ blanco

Entonces para realizar la representación de una imagen usamos matrices en memoria donde tenemos:

m ..... filas

n .....columnas

p .....nº de bits para la cuantificacion de una muestra

Cada punto de la matriz es un pixel , entonces el numero de pixeles es MxN , esto define la resolución, y cada pixel esta cuantificado en un numero determinado de bits definiendo desde 0 a 2^p posibles valores lo que define la calidad de la imagen.
Entonces la matriz queda de un tamaño dado por:

MxNxP..... bits

y por comodidad usaremos matrices cuadradas de M=N

Relaciones básicas

Vecindad :

horizontal:..... (x-1,y) y (x+1,y)

vertical:.....(x,y-1) y (x,y+1)

diagonal.....(x-1,y-1) ; (x+1,y-1) ; (x-1,y+1) y (x+1,y+1)

contorno:

$C_4(p)$: de orden cuatro ,esta dado por los cuatro pixeles de vecindad horizontal y vertical.

$C_8(p)$: de orden ocho , dado por los cuatro horizontales y verticales, y los cuatro de vecindad diagonal

conectividad

Para saber si dos pixeles (p y q) están conectados hay tres criterios:

1) - si existe proximidad física , es decir que la distancia entre ellos no supere un valor umbral.

2) - si existe entre ellos una relación de vecindad de orden 4, u 8 , o M ===> $q \in C_4(p)$ o...

$q \in C_v(p)$ y $N_4(p) \cap N_4(q) = \varnothing$

3) - si sus niveles de gris satisfacen un determinado criterio de gris.

Distancia:

sean los pixeles : p y q de coordenadas: (x,y) y (s,t) respectivamente entonces definimos:

1> distancia euclidea:

$$D_E(p,q) = [(x-s)^2 + (y-t)^2]^{1/2}$$

2> distancia Manhattan o D_4:

$$D_4(p,q) = |x-s| + |x-t|$$

3> distancia checkboard o D_8:

$$D_8(p,q) = \max (|x-s| , |x-t|)$$

D_E

```
        2
   1,4  1  1,4
2   1   0   1   2
   1,4  1  1,4
        2
```

D_4

```
      2
    2 1 2
2 1 0 1 2
    2 1 2
      2
```

D_8

```
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
```

Clasificación de las operaciones:

1. Objetivos
   a) restauración
   b) mejora
   c) segmentación
   d) análisis
   e) comprensión / codificación
2. Dominio de la operación
   a) dominio de coordenadas espaciales
   b) dominio de coordenadas frecuenciales
3. Alcance la operación . Efecto
   a) operaciones de punto
   b) operaciones de área o entorno
   c) modificación del histograma
   d) operaciones geométricas
   e) operaciones de transformación.

1. a) **Restauración:**

El objetivo es contrarrestar la degradación de la imagen por el ruido en la captura , por lo gral. a través de filtros con un objetivo concreto.

1. b) **Mejora**

Acentúa o realza las características de la imagen para su análisis: bordes , texturas , contrastes o brillos.

No agrega información a la imagen.

1. c) **Segmentación:**

Desglosa los componentes de la imagen para extraer de ella aquella que contiene la información relevante, la que se busca.

Segmentación por niveles de grises, utilizada para diferenciar objetos claros de oscuros.

Segmentación por regiones, diferencia regiones cerradas entre si

1. d) **Análisis:**

La fase de análisis implica el estudio de las características de la imagen con el fin de extraer una información adicional contenida de forma poco evidente en la imagen.

1. e) **Compresión / codificación.**

La compresión minimiza el volumen necesario de la imagen a base de eliminar la información redundante.

La codificación transforma la representación de la imagen de acuerdo a un propósito definido. Puede codificarse para enviar una imagen por línea o, para hacerla ilegible a un intruso. (esta

codificación puede aumentar el tamaño de la imagen).

Técnicas de manipulación de pixeles.

Las operaciones de pixeles toman el valor de luminancia de un pixel y se le aplica una transformación deseada. Entonces si teníamos un valor de pixel:

$u(x,y) \in [0, L]$ con $L=255$ en nuestro caso.

y le aplico la función dando como resultado v(x,y), esto es:

$v(x,y) = f(x,y) \in [0,L]$

- usaremos en todas las funciones:

unsigned char origen [ ] [256], destino [ ] [256];
int size, umbral, inf, sup, nivel, back, ventana, matriz [][9], temp, xo, yo, vent, suma, ii, jj , modo, matr[];
float valor, sig, vm, cte, incrx, incry, factor, ejex, ejey, beta;

Complementacion o efecto negativo

Sustituye el valor de luminancia en cada punto por el de su complemento

efecto: Se logra el negativo digital de la imagen.

función:

- v(x,y) = L - u (x,y)
- **void complementa (origen destino , size)**

recibe: la imagen de origen y la matriz donde estará el resultado, y el tamaño de la matriz.

devuelve: en la matriz destino tendremos la imagen complementada.

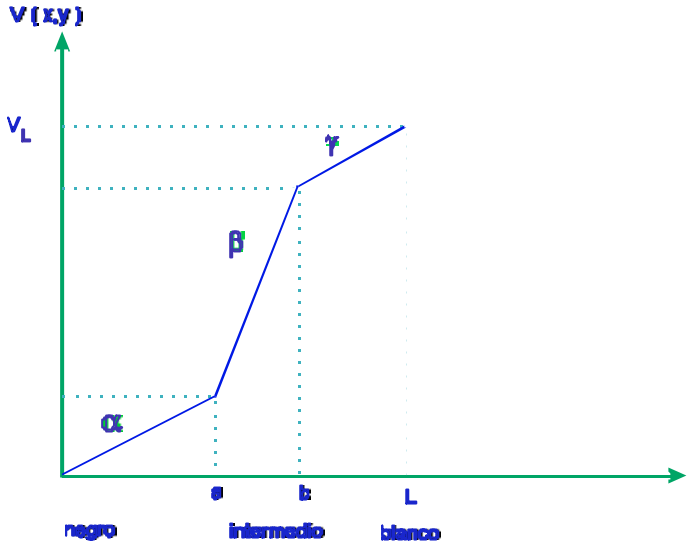
Ampliación de contraste

Aplica una función transferencia con pendiente distinta de 1 y se pueden distinguir tres regiones : pixeles oscuros [0 , a) , pixeles intermedios [a , b) y pixeles brillantes [b , L]

efecto: mejora el aspecto de la imagen con la ampliación del contraste

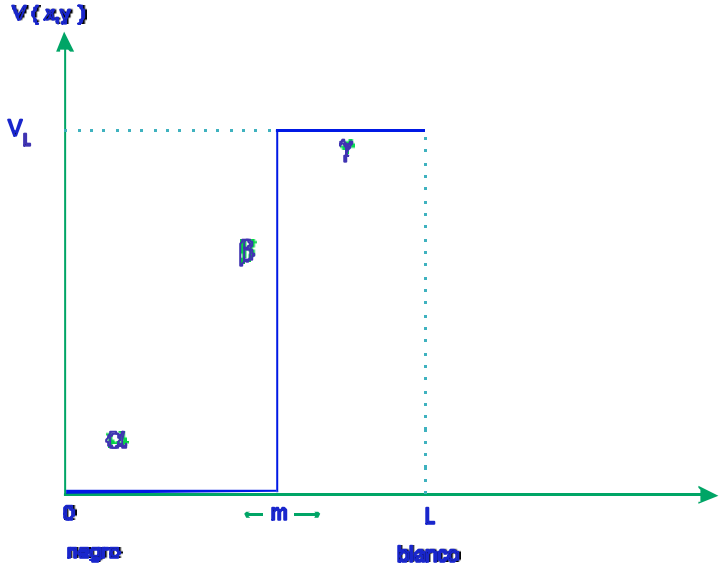
función: Aplica una pendiente de transformación mayor que 1 en la zona de brillo intermedio a costa de reducir la pendiente en la zona de claros y oscuros. Esta puede tener mas de tres regiones.

$V_L = L$



Binarizacion

Genera una imagen en dos tonos (blanco y negro) a partir de otra con múltiples niveles de gris. Es un caso particular de la ampliación de contraste en la que $\alpha=\gamma=0$ y $\beta=\pi/2$



efecto: La imagen resultante queda en dos niveles de gris que por lo gral. es blanco y negro permitiendo luego realizar una segmentación mas fácil, detección de bordes, formas, etc.

función: Si $u(x,y) < m \Rightarrow v(x,y) = 0$
si $u(x,y) > m \Rightarrow v(x,y) = V_L = L$

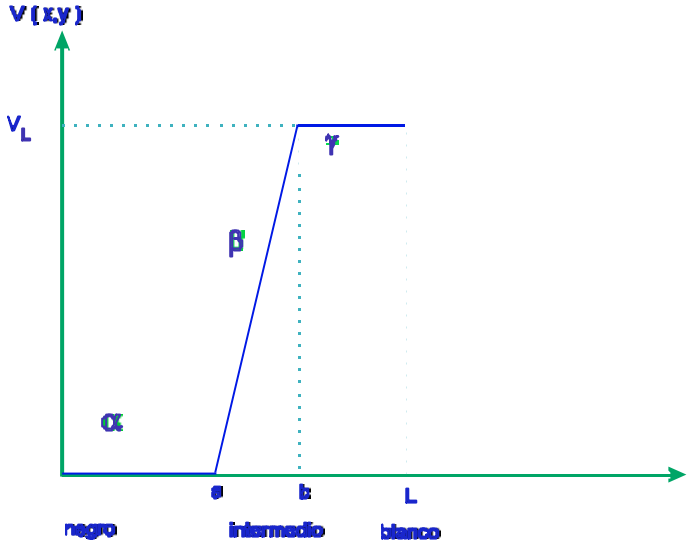
void binariza (origen, destino, size , umbral)

recibe: La matriz imagen de origen y destino, el tamaño y el valor umbral para la binarizacion (m).

devuelve: En la matriz destino queda la imagen en forma binaria.

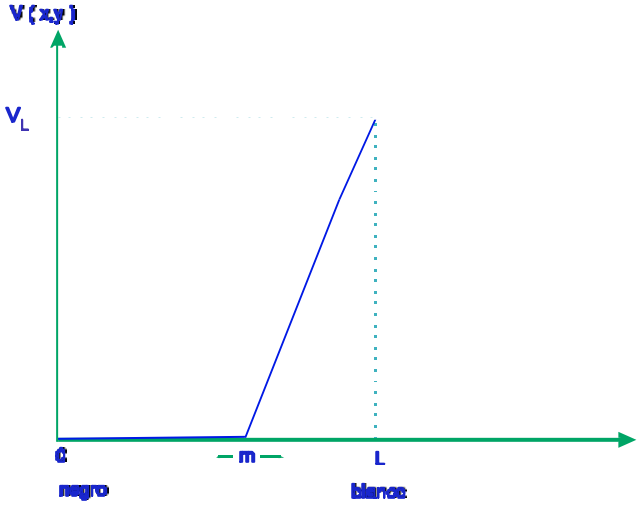
Clipping

Hace como un estilo de binarizacion pero con un β=/= π/2 .
efecto: Amplia drásticamente el contraste



Umbralizacion

Respeta el contenido de la imagen salvo una porción (alta o baja) de los niveles de intensidad que se elimina de la imagen. Dicha región termina o comienza en el valor umbral.
efecto:



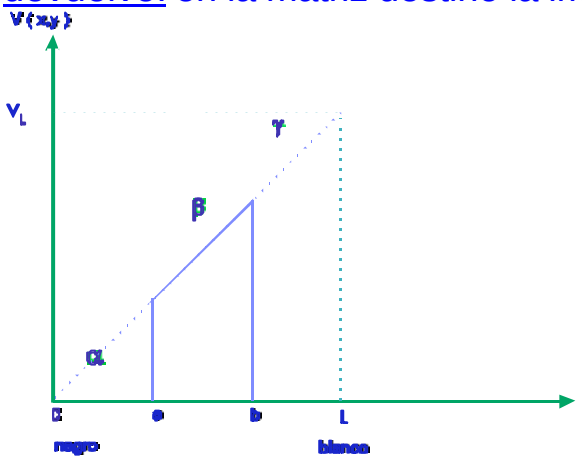
Slice

Resalta una franja de nivel de gris que se deja en su valor primario o a un nivel máximo Ln, mientras que los restantes valores de luminosidad se dejan a cero o a su valor previo respectivamente.
efecto:
función: **void slice (origen , destino , size, inf, sup, nivel, back)**

recibe: las matrices de imagen origen y destino, el tamaño, y:

- **inf** y **sup** son los niveles de gris que delimitan la franja de gris que delimitan la franja de slicing..
- **nivel** corresponde al nivel de gris que tomaran todos los pixeles que estén dentro de la franja.
- **back** decide el nivel de gris de los valores ajenos a la franja, así si **back**=0 se ponen a 0, y si **back**=/=0 conservan su nivel de gris.

devuelve: en la matriz destino la imagen transformada.



Logaritmo

Se aplica el logaritmo a los niveles de gris. transformando la intensidad a contraste.

efecto: Realza el contraste de la zona de niveles bajos de gris frente a un empobrecimiento de la correspondiente a los niveles altos.

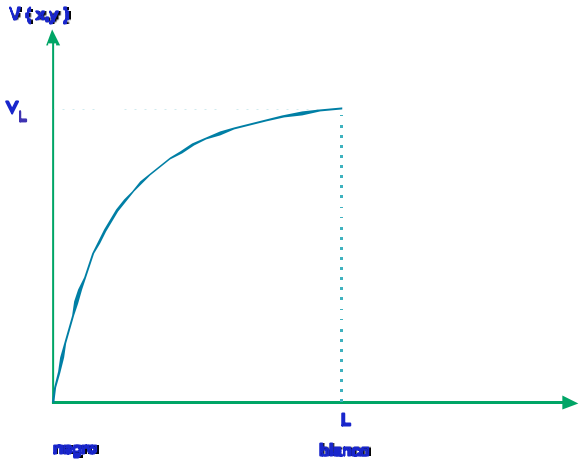
función: $\mathbf{v(x,y) = k * \log_{10}(1+ u(x,y))}$

void logaritmo (origen , destino , size , valor)

recibe: la matriz imagen de **origen** y **destino**, con su tamaño **size**. Y además **valor** representa la constante **k** y este parámetro debe elegirse para que v(x,y) <= L, entonces :

valor = Ln / log (1 + Max (u (x,y))

devuelve: la imagen transformada en **destino**.



Operaciones Aritmético

1. *Suma* **v(x,y) = u (x,y) + k** incrementa linealmente la luminancia de cada pixel
2. *Resta* **v(x,y) = u (x,y) - k** disminuye linealmente la luminancia de cada pixel
3. *Producto* **v(x,y) = u (x,y).k** incrementa proporcional mente la luminancia de cada pixel
4. *Logaritmo* **v(x,y) = K . log (1 + u(x,y))** aumenta el contraste de las zonas oscuras en detrimento de las zonas claras
5. *Exponencial* **v(x,y) = K. exp (u(x,y) -1)** aumenta el contraste en las zonas claras en detrimento de las zonas oscuras

Operaciones lógicas

1. AND **v(x,y) = u(x,y) and (k)** extracción de los bits que en u(x,y) estén en "1"
2. OR / XOR **v(x,y) = u(x,y) OR (k)** borra los bits que se indique en la mascara
3. NOT **v(x,y) = NOT { u (x,y) }** complementa el pixel.

Ruido

Simula el efecto del ruido aleatorio sobre la imagen a procesar para el estudio de métodos que permitan eliminar el efecto indeseable de las interferencias.

efecto: Agrega un ruido aditivo aleatorio .

función: $R_{i,j} = 2 * s * (Rnd + Rnd + Rnd) + V_m - 3 * s$

void ruido (origen , destino , size , sig , vm)

Donde Rnd es una función generadora de números aleatorios entre 0 y 1.

recibe: La matriz de la imagen **origen**, la de **destino** ,el tamaño en **size**. Donde **sig** es la desviación típica gaussiana correspondiente a la distribución de probabilidad del ruido. Y **vm** es su valor medio .

devuelve: en la matriz **destino** devuelve la imagen con el nivel de ruido predefinido en **sig**.

Filtros espaciales

Los métodos de mejora basados en el dominio de las frecuencias modifican indirectamente la luminancia de cada pixel, utilizando como factor de ponderación los valores de los otros pixeles de la imagen o del entorno del punto.

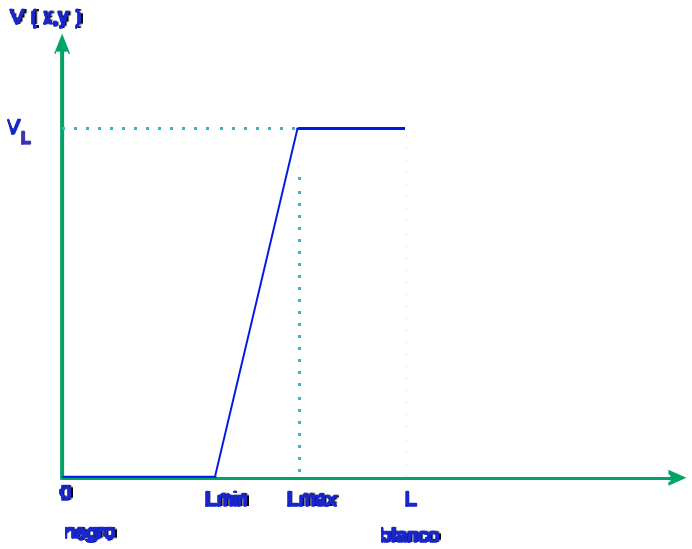
a) *Sin convolucion*

Normalización:

Consiste en aplicar un mapa de transformación de luminancias que haga corresponder a los valores mínimo y máximo de la imagen original los valores mínimo y máximo del rango permisible de lumiinancia (0 y L)

función: $\mathbf{Pix_{ij} = L_x \cdot \frac{Pix_{ij} - L_{min}}{L_{max} - L_{min}}}$

void normaliza (origen , destino , size)



La imagen original tiene como extremos de luminancia (L_{min} , L_{max}) y la idea es transformar sus extremos a (0 , L).

Filtro de media

Se sustituye el pixel central por la media aritmética de los pixeles de su entorno (incluido el mismo) .

Efecto: difuminacion de la imagen, suavizado. Sirve para eliminar los ruidos aleatorios.

función: Toma como entorno una ventana cuadrada de Nv x Nv centrada en el punto y aplica :

$$Pix_{i\,j} = \frac{1}{(2*Sv)^2} \sum_{m*i-Sv}^{m*i+Sv} \sum_{n*j-Sv}^{n*j+Sv} Pix_{m,\,n}$$

Donde **Sv** representa el tamaño en pixeles del lado de la semiventana, definida como la parte entera de la mitad del tamaño de la ventana , en pixeles. (la ventana con un valor impar).

void media (origen , destino , size , ventana)

recibe: la matriz imagen de **origen** y de **destino**, con su tamaño en **size**. Y además el tamaño de la ventana en **ventana**.

Filtro de mediana:

Análogo al filtro de media, sustituyendo el valor de cada pixel por el de la mediana de los valores de su entorno. Donde la mediana es el valor central del array de valores ordenados.

Efecto: Elimina los puntos aislados (del ruido) pero sin difuminar tanto los bordes y otros detalles abruptos (correspondientes a las altas frecuencias de la imagen).

Filtro de moda:

se sustituye el valor de cada pixel por aquel que mas veces aparece en su entorno considerado.

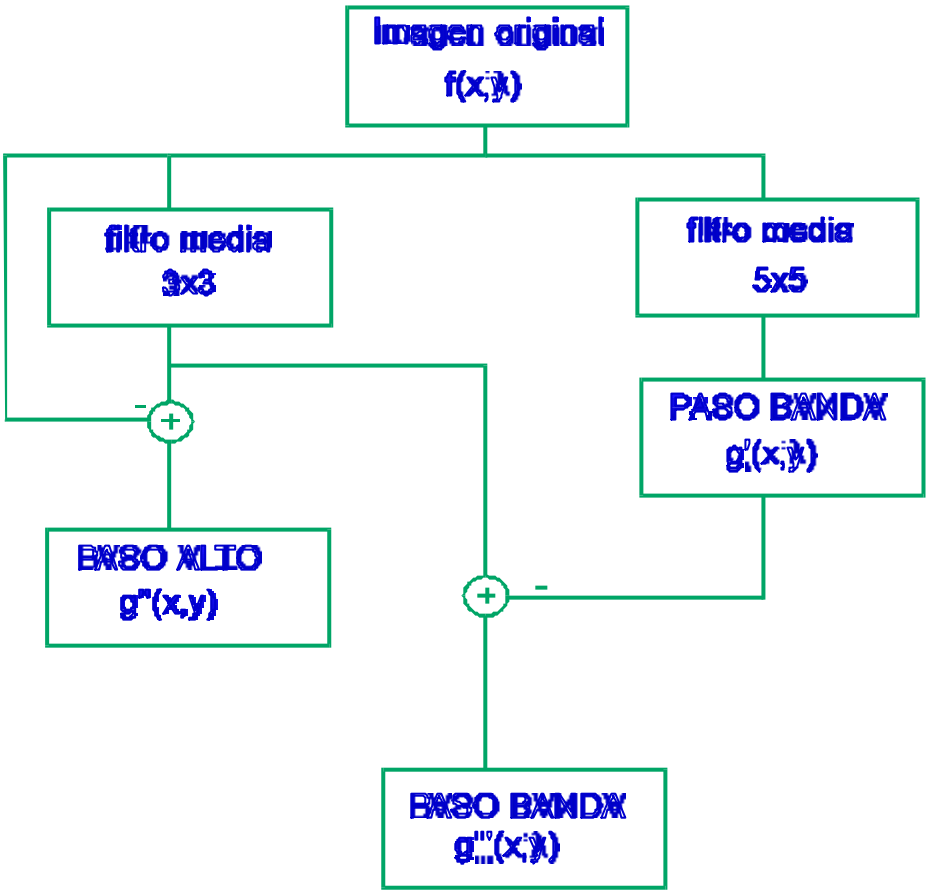
Efecto: Aun en discusión, algunos dicen que difumina y elimina progresivamente el fondo, pero depende de la imagen.

Filtrado de frecuencias

Las frecuencias espaciales determinan la variacion local de la luminosidad en el entorno del pixel. Así las bajas frecuencias corresponden a las zonas con pequeños cambios de luminancia, mientras que las altas frec. tienen que ver con las zonas donde hay cambios bruscos.

Con el teorema de la convolucion podemos obtener la filtracion deseada, al multiplicar la transformada de Fourier por la transf. de fourier del filtro deseado (matriz de dos dimensiones.) O podemos utilizarse el filtro de media (paso bajo). En el filtro de

media la frec. de corte esta determinada por el tamaño de la ventana. Entonces se puede obtener un pasa banda al restar el resultado de un filtro de media de un ventana grande con uno de ventana mas pequeña. Y restando a la imagen real el resultado de un filtro paso bajo obtengo un filtro paso alto.



Unsharp masking: Resta a una imagen una fraccion de la misma luego de un filtro paso bajo.
efecto: Resalta y acentua bordes en procesos litograficos.
funcion: **v(x,y) = u(x,y) - m.[g(x,y) Ã u(x,y)]**
Donde **u** es la imagen original, **g** es la funcion de unsharp y **m** es la constante de suavizado.
void unsharp (origen, destino, size, ventana, cte)
recibe: La imagen original en **origen,** el tamaño de la imagen en **size**, el tamaño de la ventana en **ventana**, y la constante de suavizado (**m**) en **cte**.
devuelve: En **destino** esta el resultado.

b) Con convolucion.

La convolucion

En el caso de las imagenes tenemos que la respuesta a una delta δ(m,n) ,por ser de bidimensional, es la salida que presenta el sistema cuando a la entrada se aplica un punto lineal (de luminancia infinita y anchura nula) . A esta funcion de respuesta se la conoce como PSF (point spread function). Así la convolucion se define para el caso bidimensional como:

$$g(x,y) = h(x,y) \tilde{A} f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x-x',y-y').\ f(x',y').dx'.dy'$$

en el caso discreto, reemplazo las integrales por sumatorias

$$y(m,n) = h(m,n) \tilde{A} x(m,n) = \sum_{m'=-\infty}^{\infty} \sum_{n'=-\infty}^{\infty} h(m-m',n-n').\ x(m',n')$$

efecto: El teorema de convolucion nos indica un filtrado directo en frecuencias, de acuerdo con la equivalencia entre convolucion en el dominio espacial y producto en el dominio de frecuencias :
$g(x,y) = h(x,y) \tilde{A} f(x,y) \iff G(w_1,w_2) = H(w_1,w_2) . F(w_1,w_2)$
Donde **f**, **h**, y **g** representan las señales de entrada, respuesta al impulso y salida, **F**, **G** y **H** sus transformadas de Fourier.
funcion: Realiza un producto ponderado de la matriz de convolucion con el entorno de un pixel, para cada pixel de la imagen.

$$P_{ix} = P'_s = \left(\sum_{i=1}^{9} K_i . P_i \right) / \left(\sum_{i=1}^{9} K_i \right)$$

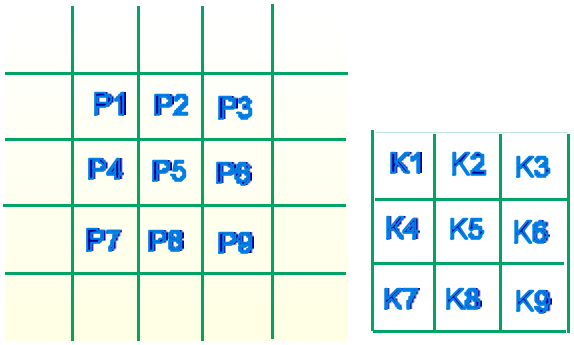


IMAGEN **MASCARA**

void convol (origen, destino, size, matriz, ventana)
recibe: La imagen original en **origen,** el tamaño de la imagen en **size**, el tamaño de la ventana en **ventana**, y la mascara de convolucion en **matriz**.
devuelve: En **destino** se halla el resultado de la convolucion.

Suavizado

El filtro de media calcula la media del entorno de un pixel y sustituye el valor de este por el de la media. Este mismo efecto se logra convolucionando la imagen con la mascara :

Ki = 1 ===>

1	1	1
1	1	1
1	1	1

$$P_{ix} = P'_s = (1 / 9) . \sum_{i=1}^{9} P_i$$

Efecto: Difumina los cambios bruscos de luminancia
Si al pixel central se le da un peso mayor que a los restantes el efecto de suavizado se hace menos brusco.

K5 = m m > 1

Gradientes y derivadas

efecto: Realza los bordes, magnifica los cambios abruptos en la luminancia.

funcion: Para diferenciar una imagen se aplica el gradiente. Sea la funcion bidemensional : f (x,y) se define el gradiente g (x,y) en el punto (x,y) como el vector: $g(x,y) = G [f(x,y)] = \delta f / \delta x \; \hat{i} + \delta f / \delta y \; j$
y en el caso discreto, sacando el modulo obtengo:

$$G[f(x,y)] = \{ [f(x,y) - f(x+1,y)]^2 + [f(x,y) - f(x,y+1)]^2 \}^{1/2}$$

y otra forma es la del gradiente Roberts, con diferencias cruzadas :

$$G[f(x,y)] = \{ [f(x,y) - f(x+1,y+1)]^2 + [f(x+1,y) - f(x,y+1)]^2 \}^{1/2}$$

Como se ve toma valores altos en aquellas zonas de bordes y valores bajos en las zonas con valores de gris casi constantes.

void gradiente (origen, destino, size , modo , umbral)

recibe: la matriz imagen de **origen** y de **destino**, con su tamaño en **size**. Mientras que usa de parametros a **modo** y **umbral** para determinar si, en el caso de que el valor numerico del gradiente en un punto sea menor que cierto umbral, se debe usar el valor del gradiente (modo 0) , el valor de la imagen original (modo 1) , un valor nulo (modo 3 y 4) . O si el gradiente es mayor que el umbral se pone 255 (modo 2 y 4). O sea el modo 4 pone 0 ó 255.

Para aplicar la convolucion a la operacion gradiente se parte del entorno de 3x3, y puede dividirse en dos componentes : **Gx** y **Gy** :

X1	X2	X3
X4	X5	X6
X7	X8	X9

Gx

-1	-2	-1
0	0	0
1	2	1

Gy

-1	0	1
-2	0	2
-1	0	1

$Gx = (X_7 + 2 X_8 + X_9) - (X_1 + 2 X_2 + X_3)$
$Gy = (X_3 + 2 X_6 + X_9) - (X_1 + 2 X_4 + X_7)$

Las mascaras reciben el nombre de Operadores Sobel. Luego para obtener el gradiente original $G= (Gx^2 + Gy^2)^{1/2}$

El operador Laplaciano: Se define con la derivada de segundo orden.

$$L [f(x,y)] = \delta 2f / \delta x2 + \delta 2f / \delta y2$$

que seria : $L [f(x,y)] = x2 + x4 + x6 + x8 - 4x5$

y puede implementarse con la convolucion con la mascara (a) :

a)

0	-1	0
-1	4	-1
0	-1	0

b)

-1	-1	-1
-1	8	-1
-1	-1	-1

La ventaja sobre el gradiente es cuando la imagen no tiene variaciones de intensidad abruptas, pero es mas sensible al ruido, ya que también resalta los puntos aislados. Para atenuar el efecto del ruido puedo usar la (b)

Detección de puntos y líneas

Para la deteccion de puntos se puede usar la mascara de convolucion del operador laplaciano (b) . Solo en el caso de que el pixel central tenga un valor diferente de los de su entorno, la convolucion arrojara un valor sustancial.

Para aislar lineas se deben utilizar operadores sensibles a cambios de intensidad ortogonales a las mismas. Entonces las mascaras de convolucion serian:

-1	-1	-1
2	2	2
-1	-1	-1

-1	2	-1
-1	2	-1
-1	2	-1

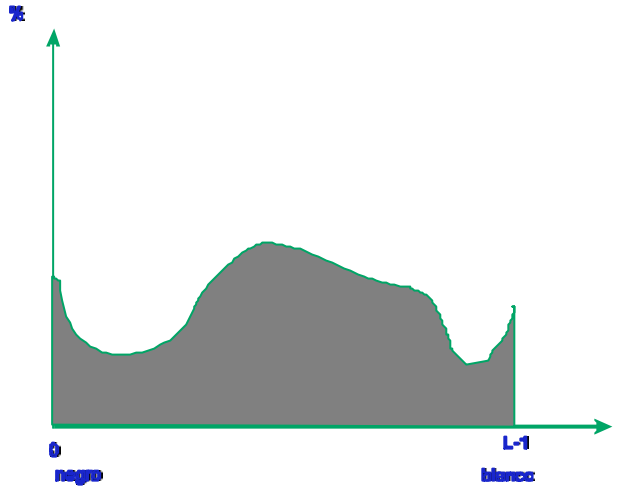
-1	-1	2
-1	2	-1
2	-1	-1

2	-1	-1
-1	2	-1
-1	-1	2

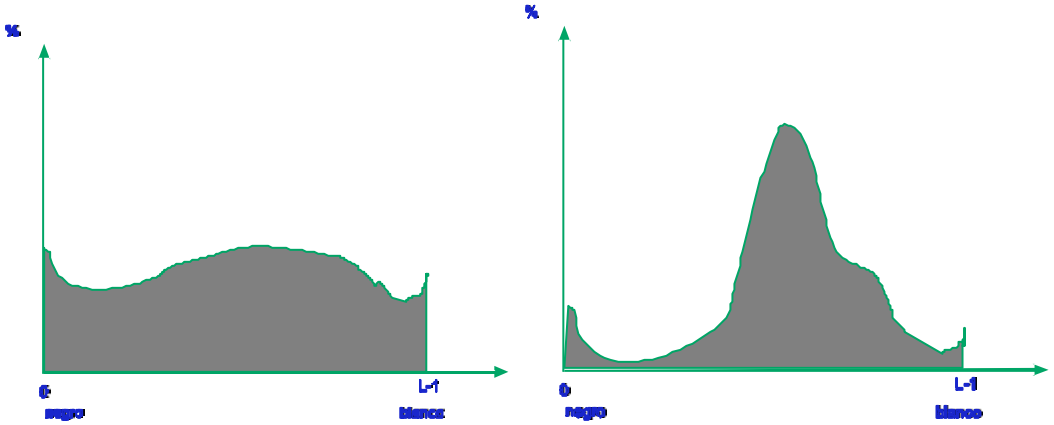
a) E-O b) N-S c) NE-SO d) NO-SE

Histograma

Se define histograma de una imagen cono la curva que en ordenadas representa cada uno de los posibles niveles de gris (0 - L), mostrando en abscisas la frecuencia relativa de aparicion del mismo en la imagen.



Un vistazo al histograma ofrece una idea rápida de cuan distribuidos se encuentran los niveles de gris en la imagen, aspecto relacionado con el contraste. Así por ejemplo podemos ver el histograma de una imagen con los niveles de gris distribuidos en forma uniforme (a), y de otra con una concentración de los niveles de gris entorno a un pequeño rango de valores (b). Por lo que el contraste es mejor en la (a) pues los niveles de gris estan mas distribuidos y es mas facil diferenciar unos de otros.



a) b)

Calculo del histograma:

Cuento el numero de apariciones de cada uno de los niveles de gris y luego saco el porcentaje normalizando con el maximo.

void histograma (imagen, tam , matr)

recibe: la matriz imagen en **imagen** con su tamaño en **tam**.

devuelve: en **matr** un vector que tiene en el subindice el valor de nivel de gris , y para cada subindice la cantidad de veces que aparece ese nivel (en porcentaje).

Aplicacion del histograma:

Ecualizacion: En un histograma muy polarizado en torno a un valor central , y si tiene poco contraste esta desaprovechando el margen dinamico, entonces con la ecualizacion modificamos la luminancia de los pixeles para distribuirlos en forma mas uniforme.

Efecto: Mejora el contraste en los histogramas muy concentrados.
Busca el histograma plano.

Función: llamamos **u** a los valores de luminancia. Podemos asumir que es una variable aleatoria con densidad de probabilidad $\mathbf{p_u(u)}$ y funcion de distribucion de probabilidad $\mathbf{F_u(u)}$. Y **h(u)** es lo que da el histograma para el nivel de gris **u**. Con **u**= x_i = 0, 1, 2, ... , L-1

$$F_u(U) = \int_0^u p_u(U) \, . \, dU$$

$$p_u(x_i) = h(x_i) \; / \; \left(\sum_{k=0}^{L-1} h(x_k) \right)$$

$$v_k = \sum_{i=0}^{k} p_u(x_i)$$

esto ultimo es para que los valores de salida también tengan L-1 niveles.

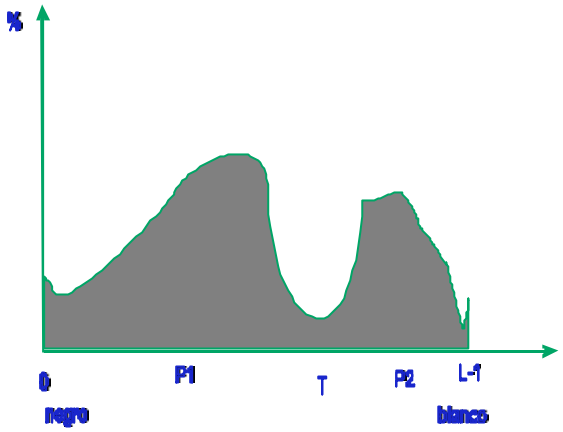
void ecualiza (origen, destino, size)

recibe: la matriz imagen de **origen** y de **destino**, con su tamaño en **size**.

devuelve: en destino la imagen ecualizada.

Segmentacion seudoautomatica

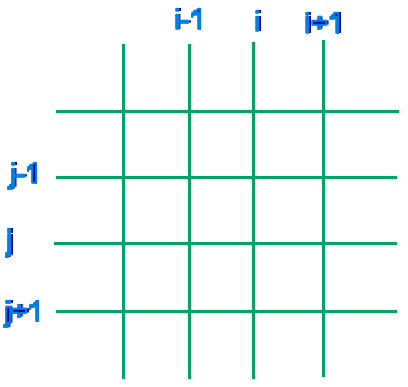
La umbralizacion en el histograma nos da la base para segmentar: separar partes autoconsistentes de la imagen. Segmentar mediante la umbralizacion consiste en separar parte de la imagen que se encuentra entre dos niveles de gris (umbrales). Así podemos por ejemplo apartar las partes oscuras de las brillantes. Pero si queremos segmentar objetos de su entorno , el problema esta en determinar los umbrales. Y para eso podemos usar el histograma



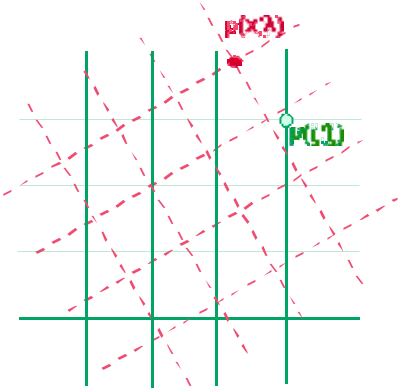
Acá podemos ver dos niveles de gris dominantes en un histograma de una imagen, la idea es encontrar el umbral T para separar los puntos del objeto los que tengan mayor o igual luminancia que T, de los puntos del fondo (todos los que no lo alcancen).

Operaciones geométricas

No modifica la informacion, solo el aspecto visual, como podría observarse desde otro punto de vista. Magnificar o reducir simula acercarse o alejarse, desplazar o rotar es hacer lo mismo con el punto de observacion. Como la imagen en memoria se guarda en forma matricial discreta, pues no hay valores de luminancias entre los valores discretos de coordenadas que corresponden a los pixeles. Como vemos en **a)** la rejilla original donde las intersecciones son los pixeles, al transformar la imagen (en este caso la roto) vemos que los pixeles de la rejilla transformada (punteada) no coinciden con los de la rejilla destino.



a)

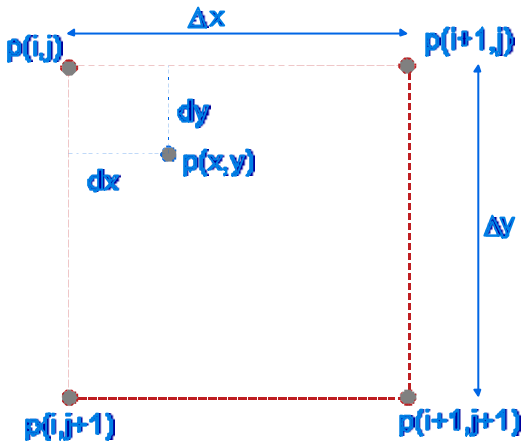


b)

Mejor lo vamos a pensar como que la rejilla destino es la continua y la rejilla de origen es la punteada, entonces lo que tenemos que averiguar es el valor del pixel en la original **p(x,y)** que aplicandole la transformacion obtengo la rejilla continua **p(i',j')**. Para averiguar el valor del pixel original **p(x,y)** uso la interpolacion.

Interpolación

Puede considerarse como el calculo del valor de luminancia de un pixel en una posicion cualquiera, como una funcion de los pixels que le rodean (posiciones enteras de la rejilla origen). Para averiguar p(x,y) podemos hacer:



0£dx£1 ; 0£dy£1 ; Dx = Dy = 1

- Asignarle el valor del pixel mas cercano de entre los cuatro que lo rodean, usando alguna funcion de distancia. En este caso p(x,y)=p(i,j)
- Asignarle la luminancia media asociada a los dos pixels mas cercanos. En este caso tomaria el valor medio de p(i,j) y p(i,j+1)
- Otra manera es realizar la interpolacion bilineal.

Interpolacion bilineal:

Asigna al pixel en cuestion un valor medio ponderado de las luminancias de los cuatro pixels que le rodean. Los factores de ponderacion vienen dados por las distancias entre el pixel y los del entorno.

$a_1 = (1 - dx / \Delta x) . (1 - dy / \Delta y) = (1 - dx) . (1 - dy)$

$a_2 = dx / \Delta x . (1 - dy / \Delta y) = dx . (1 - dy)$

$a_3 = (1 - dx / \Delta x) . dy / \Delta y = (1 - dx) . dy$

$a_4 = dx / \Delta x . dy / \Delta y = dx . dy$

El valor del pixel interpolado es :

$p(x,y) = a_1 . p(x_i,y_j) + a_2 . p(x_{i+1},y_j) + a_3 . p(x_i,y_{j+1}) + a_4 . p(x_{i+1},y_{j+1})$

Por lo gral (x,y) no es un valor entero, entonces aplico la funcion con el valor entero obtenido.
Entonces los pasos a seguir en una transformacion geometrica serian:

1) Al pixel destino hubicado en (x',y') hallo la hubicacion del pixel origen aplicando la funcion inversa, es decir en un zoom me pongo en un pixel de la imagen agrandada y busco el que le corresponde en la imagen reducida.

2) hallo por interpolacion el valor del pixel en p(x,y), donde la parte entera: (ent(x) , ent(y)) sirve para hallar los factores ponderados

3) El valor hallado de p(x,y) es el mismo en p(x',y') del destino.

funcion de interpolado bilineal:

p(x,y) = a_1 . p(ent(x),ent(y) **) + a_2 . p(** ent(x)+1,ent(y) **) + a_3 . p(**ent(x),ent(y)+1 **) + a_4 . p(** ent(x)+1,ent(y)+1 **)**

Desplazamiento

efecto: Se produce el desplazamiento de la imagen en Xd e Yd , conservando el aspecto.

funcion: Reemplaza cada pixel por el correspondiente a sus coordenadas mas el desplazamiento **incrx**, **incry** .

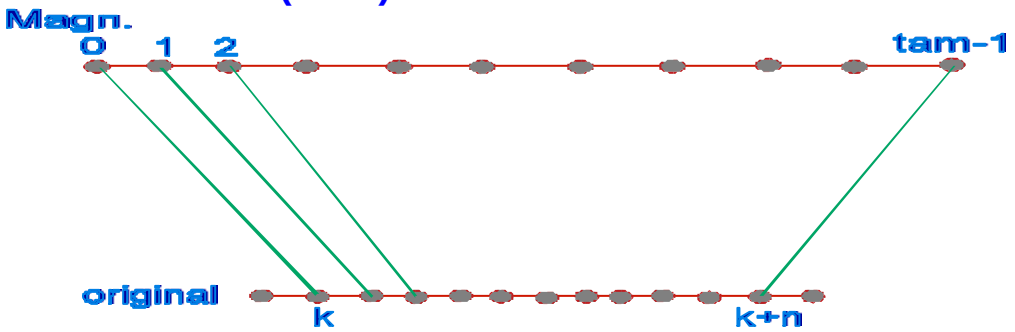
void shift (origen, destino, size , incrx, incry)

recibe: la matriz imagen de **origen** y de **destino**, con su tamaño en **size**. El desplazamiento en incrx , incry que puede tener parte decimal, y en este caso recurrimos a la interpolacion bilineal.

Cambio de escalas

De la imagen original se toma un fragmento (de **k** a **k+n**) y se amplia hasta ocupar el tamaño deseado (**tam** puntos) . Esto corresponde a un facto de aumento

fac= tam / (n+1) .



efecto: Amplia o reduce la imagen.

funcion: Primero calcula la coordenada de la rejila origen para cada uno de los **tam** puntos de la escala, y calculo el valor del pixel **p(x_{orig},y_{orig})** por interpolacion.

$Xo = X_{orig} = X_{mag} . n / (tam - 1) + k_x$
$Yo = Y_{orig} = Y_{mag} . n / (tam - 1) + k_y$
$n = tam / fac - 1$
$k_x = X_{cent} - n / 2$
$K_y = Y_{cent} - n / 2$

void magnifica (origen, destino, size, xo, yo, factor)

recibe: La matriz imagen de **origen** y de **destino**, con su tamaño en **size**. Siendo **xo** e **yo** el centro del cuadrado a ampliar, y **factor** el factor de escalado (mayor a 1 amplia, menor a 1 reduce).

Zoom y unzoom:

Cuando el cambio de escala se hace en un factor de 2, y la imagen resultante tiene el doble de tamaño que la original se duplican el tamaño de todos los pixels de la imagen. Esto evita el interpolado. También en la reduccion de la imagen a la mitad (factor = 0.5) se puede hacer eliminando una de cada dos filas, y de cada fila eliminar uno de cada dos pixels, lo que evita el trabajo del interpolado.

efecto: el zoom duplica el tamaño de la imagen, y el unzoom reduce a la mitad el tamaño de la imagen.

funcion:

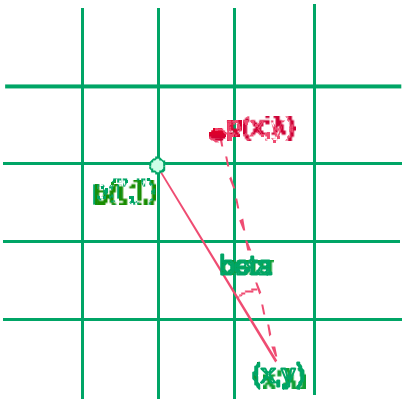
void zoom (origen, destino, size)
void unzoom (origen, destino, size)

recibe: La matriz imagen de **origen** y de **destino**, con su tamaño en **size**.

Giros :

Simula la rotacion de la camara de captura o la rotacion del objeto. Se necesitan los paramentros: centro de rotacion y angulo de giro ó centro de giro , radio de giro y posicion angular inicial.

Para cada pixel de la rejilla destino, calcula el pixel origen que le dio lugar como las coordenadas calculadas, por lo gral. , no serán valores enteros, se calculara el valor de luminancia del pixel por interpolacion (bilineal)



funcion: **void giro (origen, destino, size, beta, ejex, ejey)**
recibe: La matriz imagen de **origen** y de **destino**, con su tamaño en **size**. Siendo **beta** el angulo de giro, y el centro de giro esta dado por **ejex** y **ejey**.