ETSI Telecomunicación

Tratamiento Digital de la Imagen

Apuntes de Pak

Tratamiento Digital de la Imagen

Apuntes de Pak (Fco. J. Rodríguez Fortuño)
ETSI Telecomunicación. Universidad Politécnica de Valencia.
Primer cuatrimestre de 5º curso
Curso 2007/2008

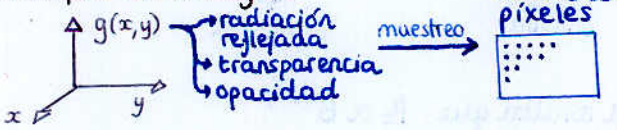
Contenido

- Referencia rápida de la asignatura
- Apuntes extensos (los últimos dos temas sólo transparencias)

Fecha de última actualización: 8 Noviembre 2008

TEMA 1. INTRODUCCIÓN

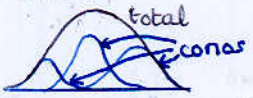
- Concepto de imagen



- Clasificación: tipos de imagen según:
  - Naturaleza: - continuas - discretas - mixtas (ej TV)
  - Nº bandas: - monobanda - multibanda
  - Animación: - estáticas - dinámicas

- Operaciones de TDI
  - Formación/captura
  - Almacenamiento/transmisión
  - Modificación
  - Extraer información
  - Visualización reproducción

El color



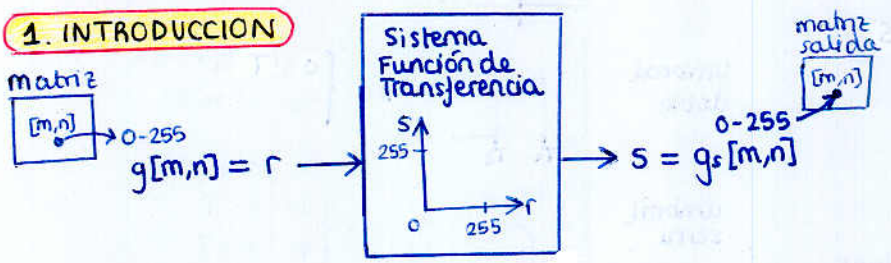
reproducción del color:
- simultáneo
- yuxtaposición espacial
- secuencial (promedio temporal)

Recordar de Imagen y Sonido
- Formación de la imagen: cámara, lente, foco, círculo de difusión, diafragma
  - distancia focal ↑ (zoom) → menor P.C. (prof campo)
  - apertura diafragma ↑ → menor P.C.
  - plano enfocado lejano → mas P.C.
  - → mas P.C. por detrás que por delante

Elementos psicofísicos:
- Retina: 120M bastones → 1000x sensibles; 6M conos → color; mayor concentración en fóvea
- Visión: resolución 0'5-2 minutos de arco; buena resolución en B/N; adaptación al nivel de luz (pupila+retina)

TEMA 2. TRANSFORMACIONES DE INTENSIDAD

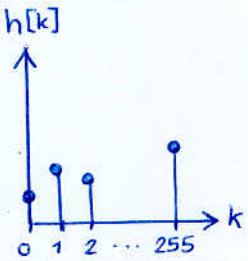
1. INTRODUCCION



- puntual: cada pto del destino DEPENDE SÓLO DE 1 PUNTO en la imagen origen
- sin memoria: cada pto [m,n] del destino DEPENDE DEL MISMO PUNTO [m,n] en el origen

2. HISTOGRAMA

k	h[k] = nº pixeles de valor k
0	h_0
1	h_1
⋮	⋮
255	h_{255}

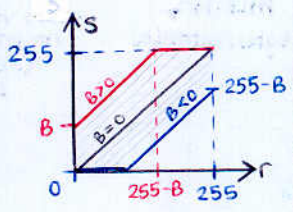
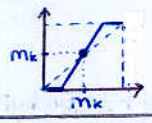
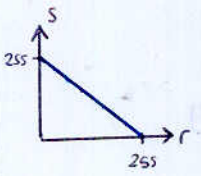
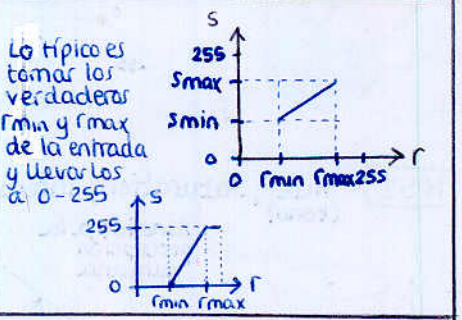
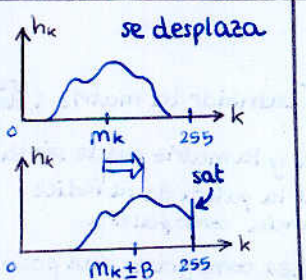
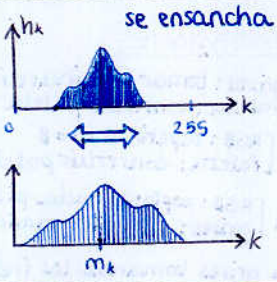
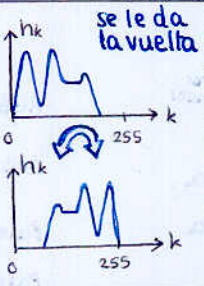
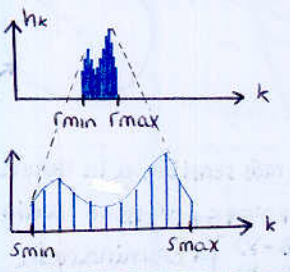


Estima la fdp de la imagen (la cual es un proceso estocástico) $P_k = \frac{h[k]}{N}$

3. REMAPEADO DE GRISES

intensidad media de una imagen $m_k = \frac{\sum^{M}\sum^{N} I[m,n]}{M \cdot N}$

3.1 Transformaciones lineales

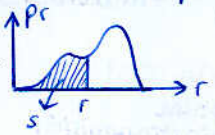
	BRILLO	CONTRASTE	NEGATIVO	CAMBIO DE MD (margen dinámico)
F. de transferencia	$S = r \pm B$ 	$S = (r - m_k)\cdot C + m_k$  • Si sólo hiciese $S = r \cdot C$ cambiaria contraste y brillo • Forma facil: tomar $m_k = 128$	$S = 255 - r$ 	Lo típico es tomar los verdaderos r_{min} y r_{max} de la entrada y llevarlos a 0-255 
Histograma				

3.2 Transformaciones no lineales

Ecualización del histograma

Teoría: Conseguir que la imagen de salida tenga un histograma plano

$$s = \int_0^r p_r(w)\,dw$$ (↳ fdp imagen r)



Dem: si $r \xrightarrow{T[r]\ \text{monótona creciente}} s \Rightarrow p_s(s) = p_r(r)\cdot\frac{dr}{ds}$

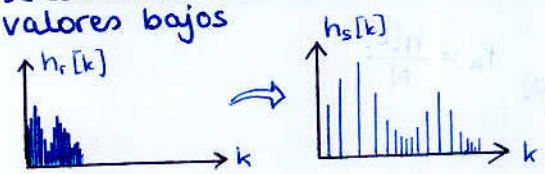
por tanto: si $s = \int_0^r p_r(w)\,dw \rightarrow p_s(s) = p_r(r)\cdot\frac{1}{ds/dr} = p_r(r)/p_r(r) = 1$

Algoritmo práctico — se usa el histograma para estimar la fdp

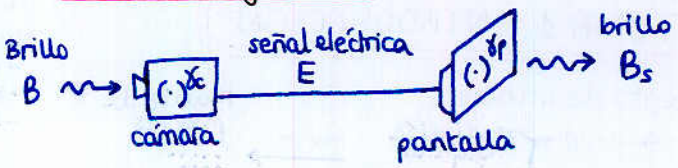
histograma normaliz. → histograma acumulado → redondeo

r	h[r]/N	A[k]	S
0	n_0/N	$(n_0/N)\cdot 255$	
1	n_1/N	$\frac{n_0+n_1}{N}\cdot 255$	
2	n_2/N	$\frac{n_0+n_1+n_2}{N}\cdot 255$	
⋮	⋮	⋮	

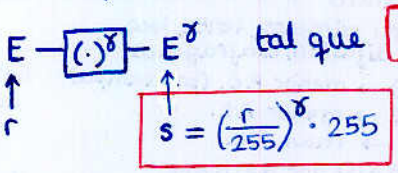
En lugar de quedar el histograma plano, se concentran más líneas allá donde hay valores bajos



Corrección gamma



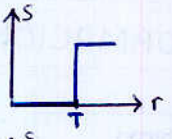
- Al final resulta que $B_s \propto B^{\gamma_c\gamma_p}$
- Para compensar se usa un corrector de gamma



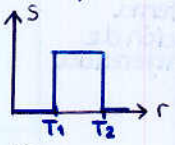
tal que $\gamma_c \cdot \gamma_p \cdot \gamma = 1$

3.3 Umbralización

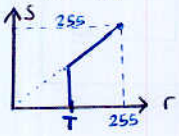
Umbral simple: $s = \begin{cases} 0 & r < T \\ 255 & r > T \end{cases}$



Umbral doble: $s = \begin{cases} 255 & T_1 < r < T_2 \\ 0 & \text{resto} \end{cases}$

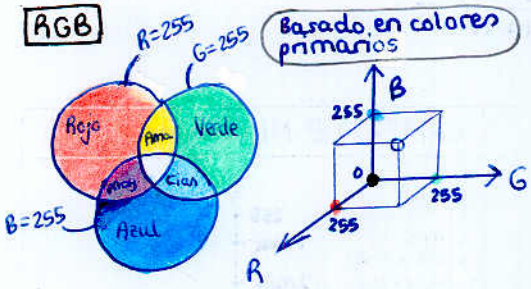


Umbral semi: $s = \begin{cases} 0 & r < T \\ r & r > T \end{cases}$

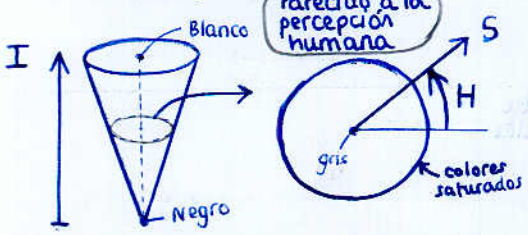


4. ESPACIOS DE COLOR

RGB — Basado en colores primarios



HSI — Hue (tono), Saturation, Intensity — Parecido a la percepción humana



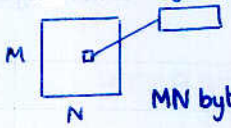
YUV — Ojo más sensible a la luminancia

$Y = 0'3R + 0'59G + 0'11B$ → luminancia

$U = K_1(B-Y)$, $V = K_2(R-Y)$ → cromninancia

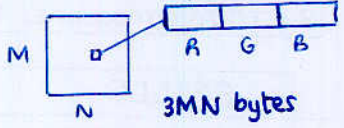
5. TIPOS DE IMÁGENES

- **Grises** — 1 byte/pixel



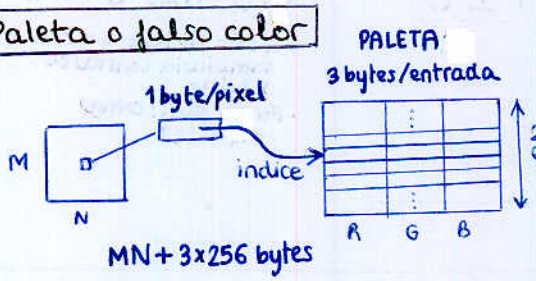
MN bytes

- **RGB** — 3 bytes/pixel



3MN bytes

- **Paleta o falso color**



MN + 3×256 bytes

- Ahorra memoria
- Solo visualizar, no procesar (salvo operaciones simples sobre la paleta ej: brillo)

CONVERSIONES

RGB a [grises: tomar luminancia / paleta: construir paleta y cambiar la matriz (creación de paleta) →

Grises a [RGB: repetir R=G=B / Paleta: construir paleta y la matriz puede ser la misma

Paleta a [RGB: copias en cada pixel la paleta de su índice / grises: tomar la luminancia como paleta

Paleta a grises tomando los índices como grises (un poco raro)

CREACIÓN DE PALETAS

Método producto: (usar una paleta fija)

Necesito rellenar 256 colores

	R	G	B	Total colores
perjudicar al azul →	8 niveles	8 niveles	4 niveles	256
favorecer al verde →	6 niveles	8 niveles	6 niveles	252
igual →	6 niveles	6 niveles	6 niveles	216

métodos adaptativos (la paleta depende de la imagen)

1. Se impone paleta inicial (método producto)
2. Se recorre la imagen RGB rellenando la matriz de índices con el índice asociado al color de la paleta más parecido (más cercano en el espacio RGB)
$$d = \sqrt{(R-R_p)^2 + (G-G_p)^2 + (B-B_p)^2}$$
3. Se actualiza cada color de la paleta con la media de los píxeles RGB asociados a él
4. Volver a 2 hasta que la paleta no cambie

6. OTRAS TRANSFORMACIONES DE INTENSIDAD

- Balance de blancos:
  - captar imagen blanca
  - calcular $factor_R$, $factor_G$, $factor_B$ [m,n] tal que $factor_R \times R = 255$, $factor_G \times G = 255$, $factor_B \times B = 255$ (en cada pixel!)
  - A partir de ahora multiplicar por esos factores (elimina el efecto de la iluminación no uniforme)
- Filtros de color: multiplicar cada componente RGB por un factor diferente

7. ARITMÉTICA DE IMÁGENES

- Suma ej $\frac{A+B}{2}$ ej $\alpha A + (1-\alpha)B$ para hacer fundidas
- Resta útil para ver diferencias (detectar movimiento)
  $|A-B|$ sólo detecta si hay o no
  $A-B+128$ detecta el sentido
- Operaciones lógicas AND, OR, XOR → útil con máscaras

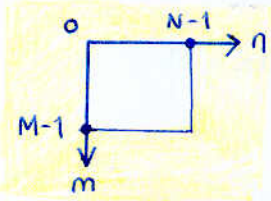
TEMA 3. TRANSFORMACIONES GEOMÉTRICAS

T. Intensidad: cambian el valor pero no la posición

T. Geométricas: cambian la posición pero no el valor

- puntuales
- con memoria

ejes:



$x[m,n] \rightarrow$ T. Geométrica $\begin{cases} m' = f(m,n) \\ n' = g(m,n) \end{cases} \rightarrow y[m,n] = x[m',n'] = x[f(m,n), g(m,n)]$

(las primas son las del origen)

1. TRANSFORMACIONES PERFECTAS

- Espejo: $y[m,n] = x[m, N-1-n]$
- Arriba-abajo: $y[m,n] = x[M-1-m, n]$
- Rotación 90° horario: $y[m,n] = x[M-1-n, m]$
- Rotación 90° antihorario: $y[m,n] = x[n, N-1-m]$
- Rotación 180°: $y[m,n] = x[M-1-m, N-1-n]$

2. GIROS ARBITRARIOS

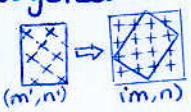
$$\begin{cases} m' = (m-m_0)\cos\theta + (n-n_0)\text{sen}\theta + m_0' \\ n' = (n-n_0)\cos\theta + (m-m_0)\text{sen}\theta + n_0' \end{cases}$$

siendo (m_0', n_0') → centro imagen origen

(m_0, n_0) → centro imagen destino

Problema ①: La rejilla puede no ser ortogonal ((m', n') no ser enteros)

Solución: → vecino más próximo
↳ interpolación



Problema ②: La imagen girada no es un rectángulo ni mide MxN

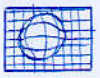
Solución: → agrandar matriz y rellenar con un color
↳ recortar la imagen girada

3. CAMBIOS DE FORMA

- **con ley matemática**

$y[m,n] = x[m',n'] = x[T(m,n)]$

ejemplo: efecto bola de cristal



- **arbitrarios**

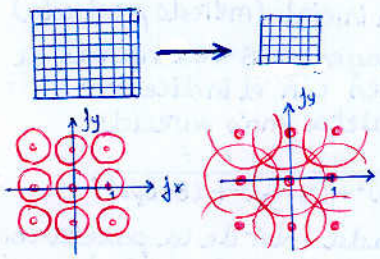
Transformación afín:
$$\begin{pmatrix} m' \\ n' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} m \\ n \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$
Incógnitas: 6 parámetros
Necesita: tres parejas $(m,n) \leftrightarrow (m',n')$

Warping: sólo cambio de forma
Morphing: cambia forma y valor RGB

Procedimiento:
1. Mallar las imágenes con triángulos (cada triángulo es una transfo. diferente)
2. Con los vértices de cada par de triángulos se calcula a, b, c, d, e, f
3. Se aplica dicha transformación a los píxeles del interior del triángulo destino

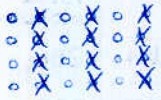
4. DIEZMADO E INTERPOLACIÓN

- **Diezmado**



Cuidado con el aliasing! (F. P. Bajo antes de diezmar)

ej: diezmado por K=2 en horizontal



$y[m,n] = x[m, 2n]$

$0 \le m \le M-1$

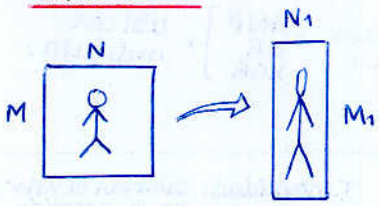
$0 \le n \le \frac{N}{2}-1$

- **Interpolación**

Aumentar el nº de pixeles en horizontal, vertical, o ambas

ej: interpolar en vertical = insertar K-1 filas entre cada 2

- **Remuestreo**

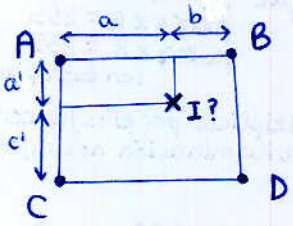


interpolar y/o diezmar adecuadamente en horizontal y vertical

TÉCNICAS PARA INTERPOLAR PIXELES

- **Vecino más próximo**: (interpolador orden cero)
  - tomas el valor del más cercano
  - aparece el efecto de bloques
- **Interpolación bilineal**

Bordes suavizados pero imagen más borrosa



se usan las distancias lejanas, para que si está cerca sean mayores

$$I = B(ac') + C(a'b) + D(a'a) + A(b'c)$$

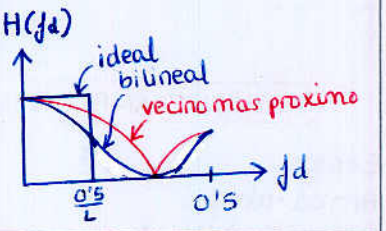
- **Interpolador ideal**

(como en TDs) · insertar ceros · filtro paso bajo $f_c = \frac{1}{2L}$

En realidad según el filtro se tiene un tipo de interpolación u otra, pero siempre cumple

$h[0,0] = 1$

$h[jK, eK] = 0$ para $[j,e] \neq (0,0)$ $\xrightarrow{TF}$ $H(f_d)$



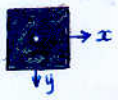
TEMA 4: TRANSFORMADA DE FOURIER BIDIMENSIONAL

1. SEÑALES Y SISTEMAS CONTINUOS 2D

Ejemplos de señales 2D

nota: $\underline{x} \triangleq \begin{pmatrix} x \\ y \end{pmatrix}$

- Impulso unitario: $\delta(x,y) = \delta(\underline{x})$



- Escalón: $u(x,y) = u(\underline{x}) \underset{\text{separable}}{=} u(x)\cdot u(y)$

- Recta: $\begin{cases} \text{eje } x: \delta(y) \\ \text{eje } y: \delta(x) \\ \text{oblicua}: \delta(ax+by+c) \end{cases}$

recta de ecuación $ax+by+c=0$

- Cuadrado: $\Pi(x,y) = \Pi(\underline{x}) \underset{\text{separable}}{=} \Pi(x)\cdot\Pi(y)$

- Círculo: $O(x,y) = O(\underline{x}) = \Pi\left(\sqrt{x^2+y^2}\right)$

- FASORES:

→ Exponencial compleja

$$s(x,y) = e^{j(\underline{\Omega}^t \cdot \underline{x})} = e^{j(\Omega_x x + \Omega_y y)}$$

$$\underline{\Omega} = \begin{pmatrix} \Omega_x \\ \Omega_y \end{pmatrix} = \begin{pmatrix} 2\pi f_x \\ 2\pi f_y \end{pmatrix} \quad \begin{matrix} x\ [cm] \\ f_x\ [cm^{-1}] \end{matrix}$$

→ sinusoides

$$C(x,y) = \cos(2\pi f_x x + 2\pi f_y y)$$
$$S(x,y) = \operatorname{sen}(2\pi f_x x + 2\pi f_y y)$$

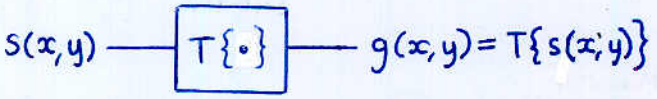
$|\underline{f}|^2 = \sqrt{f_x^2 + f_y^2}$

$\frac{\underline{f}}{|\underline{f}|^2}$ = unitario en dirección de máxima variación

ej: $C(x,y)$ con $\underline{f} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

ej $S(x,y)$ con $\underline{f} = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$

Sistemas 2D continuos



ejemplos:
- Desplazamiento: $T\{s(x,y)\} = s(x-x_0, y-y_0)$
- Giro: $T\{s(x,y)\} = s(x\cos\theta + y\operatorname{sen}\theta,\ y\cos\theta - x\operatorname{sen}\theta)$
- Reflexión sobre eje x: $T\{s(x,y)\} = s(-x,y)$
- Promedio deslizante:

$$T\{s(x,y)\} = \frac{1}{\Delta x \Delta y} \int_{x-\frac{\Delta x}{2}}^{x+\frac{\Delta x}{2}} \int_{y-\frac{\Delta y}{2}}^{y+\frac{\Delta y}{2}} s(x',y')\,dx'\,dy'$$

cada punto hace la media de un cuadradito a su alrededor

Tipos de sistemas

- puntuales: cada punto de salida depende de UN UNICO PUNTO a la entrada
- sin memoria: cada punto de la salida depende sólo del MISMO PUNTO a la entrada
- causales: cada punto depende sólo de los anteriores (poco sentido en imagen)
- estables: BIBO Bounded Input Bounded Output
- lineales: $T\{as+bg\} = aT\{s\} + bT\{g\}$
- invariantes: $T\{s(\underline{x})\} = g(\underline{x}) \Rightarrow T\{s(\underline{x}-\underline{\Delta x})\} = g(\underline{x}-\underline{\Delta x})$

→ Sistemas LSI (linear, spatially invariant)

Fáciles de caracterizar (TF): $s(x,y) \to h(x,y) \to s(x,y)*h(x,y)$; $S(f_x,f_y) \to H(f_x,f_y) \to S(f_x,f_y)\cdot H(f_x,f_y)$

LSI

Los fasores son autofunciones: $e^{j\underline{\omega}^T\underline{x}} \to \boxed{\quad} \to A\cdot e^{j\underline{\omega}^T\underline{x}}$ $\quad A\{|A|, \angle A\}$

- Fase = formas → fase lineal: no deforma, sólo desplaza

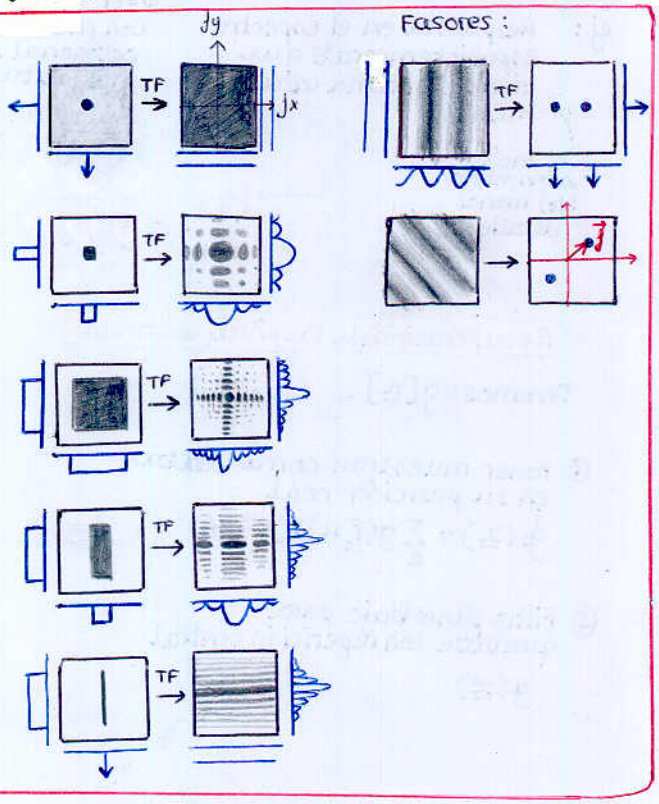
Transformada de Fourier 2D

TF2D Directa: $$S(\Omega_x,\Omega_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} s(x,y)\cdot e^{-j(\Omega_x x + \Omega_y y)}\,dx\,dy$$

TF2D Inversa: $$s(x,y) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} S(\Omega_x,\Omega_y)\, e^{j(\Omega_x x + \Omega_y y)}\,d\Omega_x\,d\Omega_y$$

Propiedades de la TF2D:
- Linealidad: $TF[ax+by] = aTF[x] + bTF[y]$
- Desplazamiento: $s(\underline{x}-\underline{x_0}) \xrightarrow{TF\,2D} S(\underline{\Omega})e^{-j\underline{\Omega}^t\underline{x_0}}$
- Convolución (filtrado): $s(\underline{x})*h(\underline{x}) \xrightarrow{TF\,2D} S(\underline{\Omega})\cdot H(\underline{\Omega})$
- Producto (enventanar): $s(\underline{x})\cdot w(\underline{x}) \xrightarrow{TF2D} S(\underline{\Omega})*W(\underline{\Omega})$
- Simetría: $s(\underline{x})$ real $\Rightarrow S(\underline{\Omega}) = S^*(-\underline{\Omega})$ hermítica

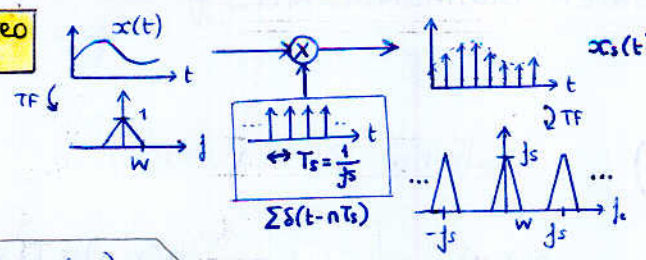
Nuevas propiedades:
- separabilidad: $s(x,y) = a(x)\cdot b(y) \xrightarrow{TF} S(\Omega_x,\Omega_y) = A(\Omega_x)\cdot B(\Omega_y)$
- Rotación: si rotamos una imagen un ángulo θ, su TF rota el mismo ángulo



2. MUESTREO DE SEÑALES 2D

Usamos el modelo de multiplicar por un tren de deltas para obtener el espectro.

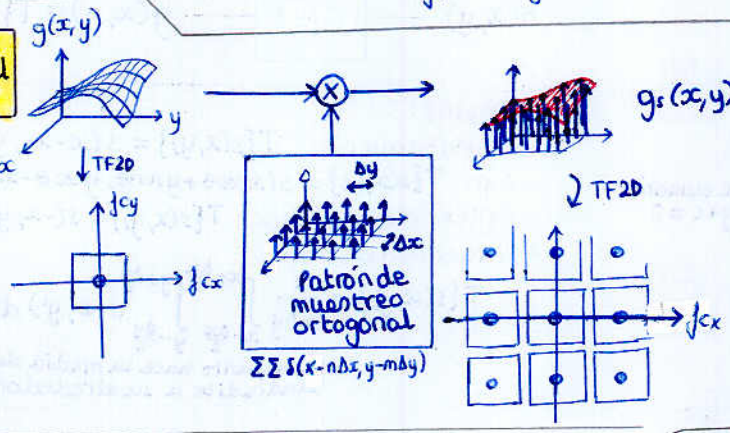
Muestreo en 1D



$$x_s(t) = x(t)\cdot\sum\delta(t-nT_s) = \sum_n \underbrace{x(nT_s)}_{x[n]}\cdot\delta(t-nT_s)$$

$$X_s(f_c) = \frac{1}{T_s}\sum_n X\left(f_c - n\frac{1}{T_s}\right)$$

Muestreo ortogonal 2D



$$g_s(x,y) = g(x,y)\cdot\sum_m\sum_n\delta(x-m\Delta x, y-n\Delta y) = \sum_m\sum_n \underbrace{g(m\Delta x, n\Delta y)}_{g[m,n]}\delta(x-m\Delta x, y-n\Delta y)$$

$$G_s(f_x,f_y) = \frac{1}{\Delta x\Delta y}\sum_m\sum_n G\left(f_x - m\frac{1}{\Delta x}, f_y - n\frac{1}{\Delta y}\right)$$

Muestreo periódico (no necesariamente ortogonal)

Usar como patrón de muestreo $\sum_m\sum_n\delta(\underline{x} - m\underline{T_1} - n\underline{T_2})$

Y entonces se obtiene

$$g_s(x,y) = g(x,y)\cdot\sum_m\sum_n\delta(\underline{x} - m\underline{T_1} - n\underline{T_2})$$

$$g_s(\underline{x}) = \sum_{\underline{n}} \underbrace{g(\underline{P}\,\underline{n})}_{g[\underline{n}]}\cdot\delta(\underline{x} - \underline{P}\,\underline{n}) \qquad \underline{P} = (\vec{T}_1\ \vec{T}_2)$$

$$\text{TF}\downarrow$$

$$G_s(f_x,f_y) = \underbrace{\frac{1}{\det(\underline{P})}}_{\text{densidad de muestreo}}\cdot\sum_{\underline{n}=-\infty}^{\infty} G(\underline{f} - \underline{U}\cdot\underline{n}) \qquad \underline{U} = (\underline{P}^{-1})^t$$

Periodicidad 2D

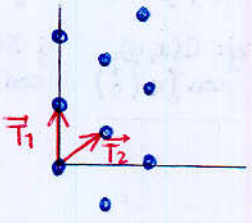
en 1D: $x(t) = x(t-nT) \rightarrow$ periodo T

en 2D:

$$s(\underline{x}) = s(\underline{x} - m\underline{T_1} - n\underline{T_2}) = s(\underline{x} - \underline{P}\,\underline{n}) \Rightarrow \text{periodicidad } \underline{P}$$

Matriz de periodicidad $\underline{P} = (\underline{T_1}\ \underline{T_2}) = \begin{pmatrix} T_{1x} & T_{2x} \\ T_{1y} & T_{2y}\end{pmatrix}$

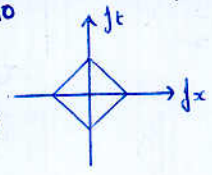
$\underline{n} = \begin{pmatrix} m \\ n\end{pmatrix}$



Caso particular: ortogonal $\underline{P} = \begin{pmatrix} \Delta x & 0 \\ 0 & \Delta y\end{pmatrix}$

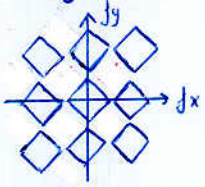
- Eficiencia del muestreo no ortogonal

ej: Respuesta en el espectro espaciotemporal que tiene el sistema visual humano

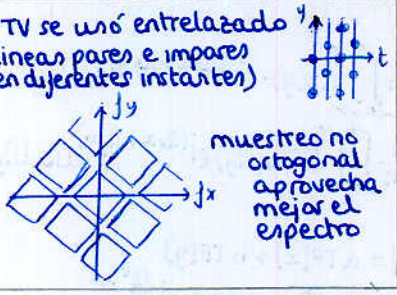


a más movimiento (ft) menos detalle (fx)

Con muestreo ortogonal se malgasta

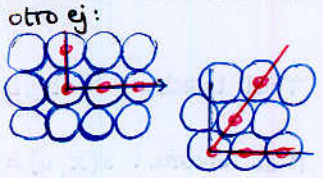


En TV se usó entrelazado (líneas pares e impares en diferentes instantes)



muestreo no ortogonal aprovecha mejor el espectro

otro ej:

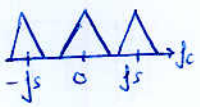


Recuperación de la señal continua

Tenemos $g[\underline{n}]$

① Poner muestras como deltas en su posición real

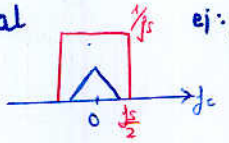
$$g_s(\underline{x}) = \sum_{\underline{n}} g(\underline{P}\,\underline{n})\delta(\underline{x} - \underline{P}\,\underline{n})$$



ej: imprimir puntitas tinta

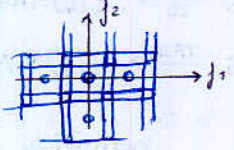
② Filtro paso bajo para quedarse con repetición central

$g(\underline{x})$



ej: el sistema visual humano

Aliasing en señales multidimensionales



Efectos:

ESPACIAL:
- efecto escalera en bordes
- efectos raros (moiré)

TEMPORAL:
- efecto estroboscópico

Solución: F.P. Bajo antes de muestrear

ejemplos:

Espacial:
- ancho del haz de electrones de una cámara de TV
- tamaño del CCD en cámaras digitales

Temporal:
- tiempo de obturación en cámaras

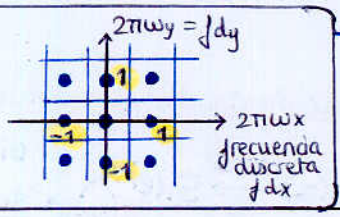
3. SEÑALES Y SISTEMAS 2D DISCRETOS

- señal discreta: $s(x,y) \rightarrow$ C/D $\rightarrow s[m,n]$ (nº pixels/cm)
- señal digital: A/D (H) → C/D (nº pix/cm) → Q (nº bits/pixel) → codif (hay que cuantificar)
- sistemas discretos: puntuales, sin memoria, causales, estables, lineales, invariantes → LSI
  - sistemas LSI: fáciles de caracterizar → $h[m,n]$, $H(e^{j\omega_x}, e^{j\omega_y})$
  - si fase lineal → sólo desplaza no deforma

· Transformada de Fourier 2D continua sobre señal discreta

$$s[m,n] \xrightarrow{TF2D} S(e^{j\omega_x}, e^{j\omega_y}) = \sum_{m}\sum_{n=-\infty}^{\infty} s[m,n]\, e^{-j(\omega_x m + \omega_y n)}$$

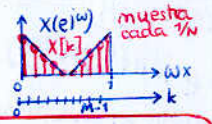
es: · continua · periódica → no se puede representar en un ordenador



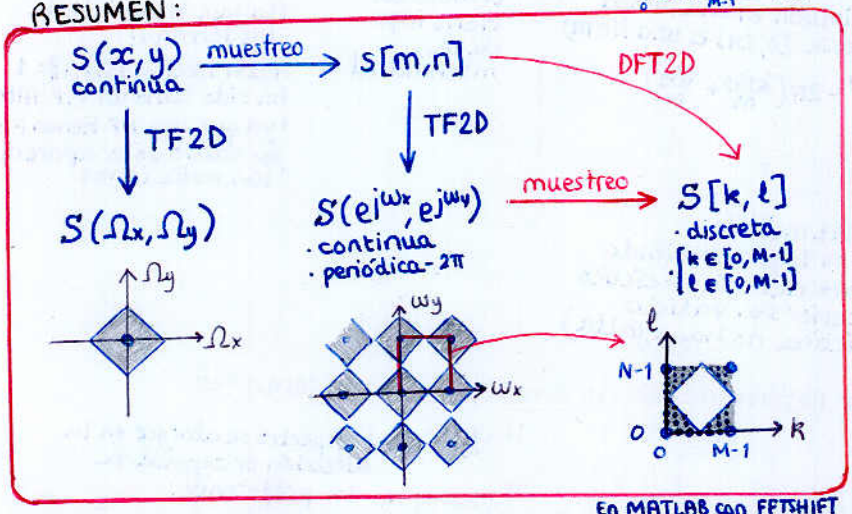
→ mismas propiedades que sobre la continua

4. LA DFT2D (FFT2D)

recuerda 1D



RESUMEN:



En MATLAB con FFTSHIFT se coloca el cero en el centro

DFT2D: Formulación:

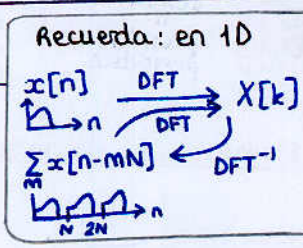
$$S[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} s[m,n]\, e^{-j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)} \qquad \begin{matrix} 0 \le k \le M-1 \\ 0 \le l \le N-1 \end{matrix}$$

$$s[m,n] = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} S[k,l]\, e^{j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)} \qquad \begin{matrix} 0 \le m \le M-1 \\ 0 \le n \le N-1 \end{matrix}$$

Propiedades:
las mismas pero con desplazamientos e inversiones cíclicas

ejemplo: $s[((m-m_0))_M, ((n-n_0))_N] \xrightarrow{DFT} S[k,l]\, e^{j2\pi\left(\frac{km_0}{M}+\frac{ln_0}{N}\right)}$ (desplazamiento; sólo altera la fase)

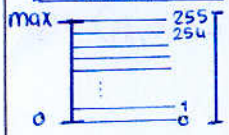
Recuerda: en 1D



Gran margen dinámico: visualización

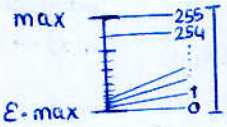
$s[m,n]$ real $\xrightarrow{DFT2D}$ $S[k,l]$ complejo → FASE; MÓDULO: gran margen dinámico

Visualización lineal:



Visualización en dB's

$$\varepsilon = 10^{-MD/20}$$

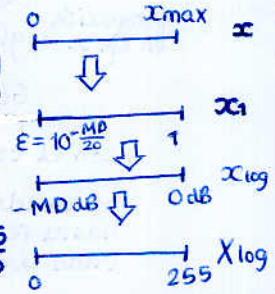


Proceso:

$$x_1 = \left[\frac{x}{x_{max}}(1-\varepsilon)+\varepsilon\right]$$

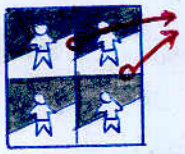
$$x_{log} = 20 \log x_1$$

$$X_{log} = (x_{log}+MD)\frac{255}{MD}$$



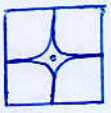
Enventanado

La DFT2D es en realidad el muestreo del espectro de la imagen repetida periódicamente

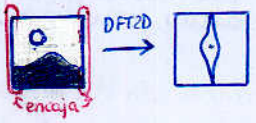


En los límites entre repeticiones pueden haber cambios bruscos que dan lugar a una cruz en el espectro (frecs elevadas)

→ FFT2D



Algunas imágenes pueden no tener uno o ambos palos de la cruz



la solución para quitar la cruz es enventanar

- dejormando la media: $s[m,n]\cdot w[m,n] \xrightarrow{DFT2D} S[k,l] \textcircled{N} W[k,l]$
- sin deformar la media: $(s[m,n]-\text{media})\cdot w[m,n] + \text{media}$
  la frec (0,0) es entonces la media original

Transformada rápida FFT2D (cálculo eficiente de DFT2D)

$x \rightarrow$ FFT2D$_{N,M}$ $\rightarrow X[k,l]$

recursivo hasta tener matriz 2x2

método más utilizado

5. APLICACIONES DE LA FFT

• Filtrado en el dominio de la frecuencia

Multiplicar por una 'ventana' en el dominio frec.

$$G(e^{j\omega}) = S(e^{j\omega}) \cdot H(e^{j\omega})$$

$\downarrow$ IFFT2D

$$g(x) = s(x) \circledast h(x) = s(((x))_N) * h(x)$$

recuerda: convol. circular = convol. lineal de la señal repetida periodicamente

• Determinación de desplazamientos

$s_1(x) \xrightarrow{\text{FFT2D}} S_1(e^{j\omega})$

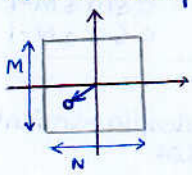
$s_2(x) = s_1(x - n_0) \longrightarrow S_2(e^{j\omega})$

División de las FFT2D

$$\frac{S_1(e^{j\omega})}{S_2(e^{j\omega})} = \frac{S_1(e^{j\omega})}{S_1(e^{j\omega}) \cdot e^{j2\pi\left(\frac{km_0}{M} + \frac{ln_0}{N}\right)}} = 1 \cdot e^{-j2\pi\left(\frac{km_0}{M} + \frac{ln_0}{N}\right)}$$

La FFT inversa de la división de FFTs

$\text{IFFT}\left(\frac{S_1(e^{j\omega})}{S_2(e^{j\omega})}\right)$ es por tanto una delta en el valor del desplaz:



La fase de la división es una rampa (que al enrollarse $[0, 2\pi]$ es una sierra)

$$\angle\left(\frac{S_1}{S_2}\right) = -2\pi\left(\frac{km_0}{M} + \frac{ln_0}{N}\right)$$

Por tanto el máximo desplaz. medible es de $\frac{M}{2}, \frac{N}{2}$ lo cual es lógico sabiendo que el desplazamiento debería ser cíclico para que todo lo anterior sea válido (aunque en la práctica no haga falta)

Nota sobre la fase:

$s[m,n]$ $\downarrow$ FFT2D $S[k,l] \rightarrow |S[k,l]|$, $\angle S[k,l]$

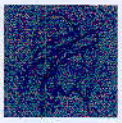
$\angle S = 0$ Tomando sólo el módulo

$\downarrow$ FFT^{-1}

El módulo tiene info de la intensidad

$|S| = 1$ Tomando sólo la fase

$\downarrow$ FFT^{-1}



¡La fase tiene info de las formas!

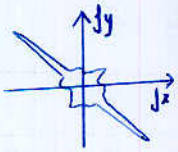
Haber hecho $|S[k,l]| = 1$ ha sido como un F.P. Alto (ya que las AF tienen $1 \cdot 1 < 1$) por tanto sólo aparecen las variaciones

• Análisis de texturas

textura = patrón periódico → Espectro de picos que la caracterizan

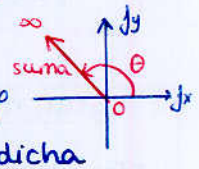
• Determinación de orientaciones preferentes

FFT2D



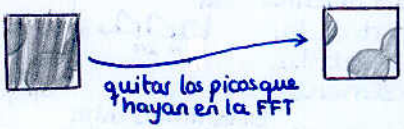
el espectro se alarga en la dirección ortogonal a la preferente.

si definimos $g(\theta) = \int_0^\infty G(\rho, \theta)\, d\rho$



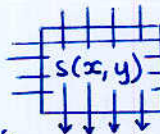
se verá un pico en dicha dirección

• Eliminación de interferencias periódicas



quitar los picos que hayan en la FFT

• Teorema de las proyecciones (se usa en TAC)

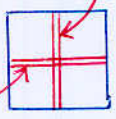


proyección en eje y: $h(y) \rightarrow H(f_y) = S(0, f_y)$

proyección en eje x: $g(x) = \int_{-\infty}^{\infty} s(x,y)\, dy$

$\downarrow$

$G(f_x) = S(f_x, 0)$



con la propiedad de rotación puedo ir tomando más líneas hasta rellenar la DFT2D (tras un remuestreo)

$\downarrow$ remuestreo

$\downarrow$ IFFT2D

$s(x, y)$

• Determinación del enfoque óptimo

Desenfoque ≡ F.P. Bajo

Por tanto se puede ir variando el enfoque hasta que se maximicen las altas frecuencias.

• Determinación de degradaciones

- Degradación se modela como un LSI $H(f)$
- Los nulos de $H(f)$ aparecen en la imagen degradada $G(f)$
- Los nulos caracterizan totalmente la degradación

TEMA 4. ESTIMACIÓN DE MOVIMIENTO EN SECUENCIAS

1. INTRODUCCIÓN

Yo estimaré el vector desplazamiento 2D (únicamente la proyección del movimiento sobre la imagen)

Problemas:

- Cambios de iluminación

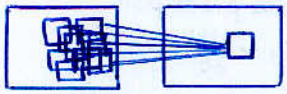
- Oclusión

- Falta de gradiente espacial: si no hay diferencias de intensidad en el objeto no veré el movimiento

ej: disco girando

2. MÉTODOS DE BLOQUES

- Comparar un bloque de la imagen t+1 con bloques de la imagen t



Medidas del parecido entre bloques

- Máxima correlación cruzada: $R = \frac{\sum_m \sum_n b_1(m,n) \cdot b_2(m,n)}{\sqrt{\sum_m \sum_n b_2^2(m,n)}}$
- Mínimo error cuadrático medio: $MSE = \frac{1}{M \cdot N} \sum_m \sum_n [b_1(m,n) - b_2(m,n)]^2$
- Mínima diferencia absoluta media: $MAD = \frac{1}{M \cdot N} \sum_m \sum_n |b_1(m,n) - b_2(m,n)|$

tomando

$b_2 = s(m, n, k+1)$

$b_1 = s(m+d_1, n+d_2, k)$

vector de desplaz $\vec{d} = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$

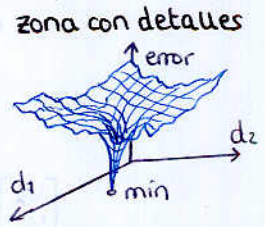
→ Búsqueda exhaustiva

Ir variando el vector desplazamiento PIXEL A PIXEL (en una zona) hasta encontrar el más parecido

- Óptimo
- Lento

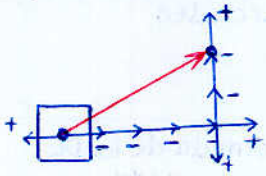
solución: métodos subóptimos

ej búsqueda exhaustiva minimizando el error



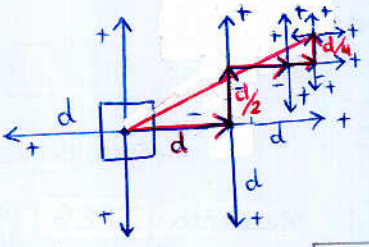
el mínimo del error (o máximo de la correlación) es más brusco en zonas con detalles

→ Búsqueda dirección conjugada



Avanzar en la dirección que reduce el error hasta que deje de reducirse (encuentra un mínimo LOCAL)

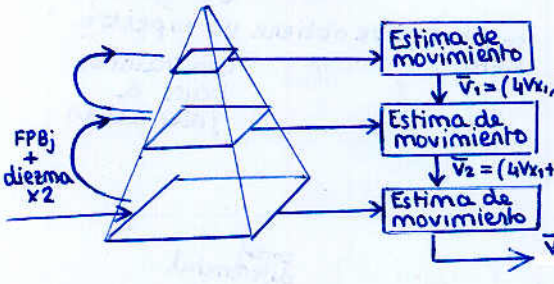
→ Búsqueda logarítmica



Busco el mínimo en las 4 direcciones a d pixeles (típicamente 2).

Cuando el menor me da en el centro, reduzco d a la mitad y repito el proceso

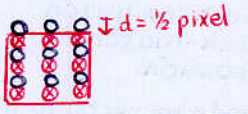
→ Multiresolución



cada nivel, al tener el doble de pixeles que el anterior, puede refinar a la mitad de finura

- Refinado de ½ pixel

Cuando hacemos que el vector desplaz. pueda variar de ½ en ½ pixel (por interpolación)



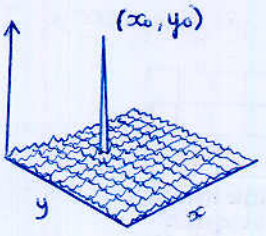
3. MÉTODOS BASADOS EN LA FASE

Imagen k → TF → $S_k(f_x, f_y)$

Imagen k+1 → TF → $S_{k+1}(f_x, f_y) = S_k(f_x, f_y) e^{j2\pi(f_x x_0 + f_y y_0)}$

$\frac{S_k(f_x, f_y)}{S_{k+1}(f_x, f_y)} = e^{j2\pi(f_x x_0 + f_y y_0)}$

$\xrightarrow{TF^{-1}}$



módulo: 1

fase: rampa enrollada de 0 a 2π

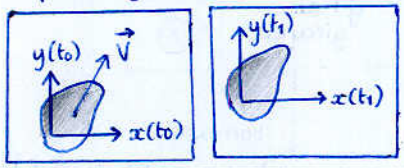
Para detectar el movimiento de objetos (sin necesidad de que se mueva toda la imagen) se divide en bloques y se hace para cada bloque

- si se usa la FFT (lo normal) el desplaz máximo medible es $(\frac{N}{2}, \frac{N}{2})$
- Es inmune a cambios de iluminación

4. MÉTODOS DE FLUJO ÓPTICO

Suposición inicial: todos los píxeles de un objeto en movimiento mantienen la intensidad en espacio y tiempo

$$\frac{d}{dt}[S_c(x(t), y(t), t)] = 0$$



regla de la cadena →

Ecuación de flujo óptico (1)

$$\frac{\partial S_c(x,y,t)}{\partial x} V_x + \frac{\partial S_c(x,y,t)}{\partial y} V_y + \frac{\partial S_c(x,y,t)}{\partial t} = 0$$

($V_x = \frac{\partial x}{\partial t}$, $V_y = \frac{\partial y}{\partial t}$)

Problema:
- 1 ecuación
- 2 incógnitas (V_x, V_y)

→ hay que exigir una condición adicional (una ecuación más)

→ Diferenciales

condición: conservación en el tiempo del gradiente espacial

$$\nabla S_c(x,y,t) = \begin{bmatrix} \partial S_c/\partial x \\ \partial S_c/\partial y \end{bmatrix}$$

$$\frac{d}{dt}(\nabla S_c(x,y,t)) = 0 \quad (2)$$

⇓ (1)+(2)

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 S_c}{\partial x^2} & \frac{\partial^2 S_c}{\partial x \partial y} \\ \frac{\partial^2 S_c}{\partial y \partial x} & \frac{\partial^2 S_c}{\partial y^2} \end{bmatrix}^{-1} \cdot \begin{bmatrix} -\partial^2 S_c/\partial t \partial x \\ -\partial^2 S_c/\partial t \partial y \end{bmatrix}$$

PROBLEMAS:
- No funciona con
  - zoom (cambia módulo de ∇S_c)
  - rotación (cambia dirección de ∇S_c)
- Difícil estimación de derivadas de 2º orden

→ Bloques

En lugar de suponer cada pixel por separado $\frac{\partial S_c}{\partial x} V_x + \frac{\partial S_c}{\partial y} V_y + \frac{\partial S_c}{\partial t} = 0$

supongo que los pixeles se mueven en grupos, por tanto sumo para varios. Además, minimizo la suma en lugar de exigir que sea cero

$$\min\left[\sum_{x,y}\left(\frac{\partial S_c}{\partial x} V_x + \frac{\partial S_c}{\partial y} V_y + \frac{\partial S_c}{\partial t}\right)\right]$$

⇓

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_{x,y} \frac{\partial^2 S_c}{\partial x^2} & \sum_{x,y} \frac{\partial^2 S_c}{\partial x \partial y} \\ \sum_{x,y} \frac{\partial^2 S_c}{\partial x \partial y} & \sum_{x,y} \frac{\partial^2 S_c}{\partial y^2} \end{bmatrix}^{-1} \cdot \begin{bmatrix} -\sum_{x,y} \frac{\partial^2 S_c}{\partial t \partial x} \\ -\sum_{x,y} \frac{\partial^2 S_c}{\partial t \partial y} \end{bmatrix}$$

PROBLEMA: cálculo de derivadas parciales

5. APLICACIONES

· Eliminación de vibración

- Movimiento global de la escena

- multiresolución
- Block-matching
- Votación

Teniendo un vector de mov. para cada bloque, se puede hacer la media (o la moda)

· Compresión de vídeo

MPEG: Moving Pictures Expert Group

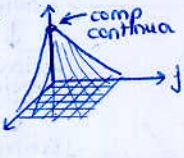
hace uso de
- JPEG para redundancia espacial
- H.261 para redundancia temporal

Fórmula de la DCT

$$F[u,v] = \frac{1}{4} C(u)C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\left(\frac{\pi(2x+1)u}{2N}\right) \cos\left(\frac{\pi(2y+1)v}{2N}\right)$$

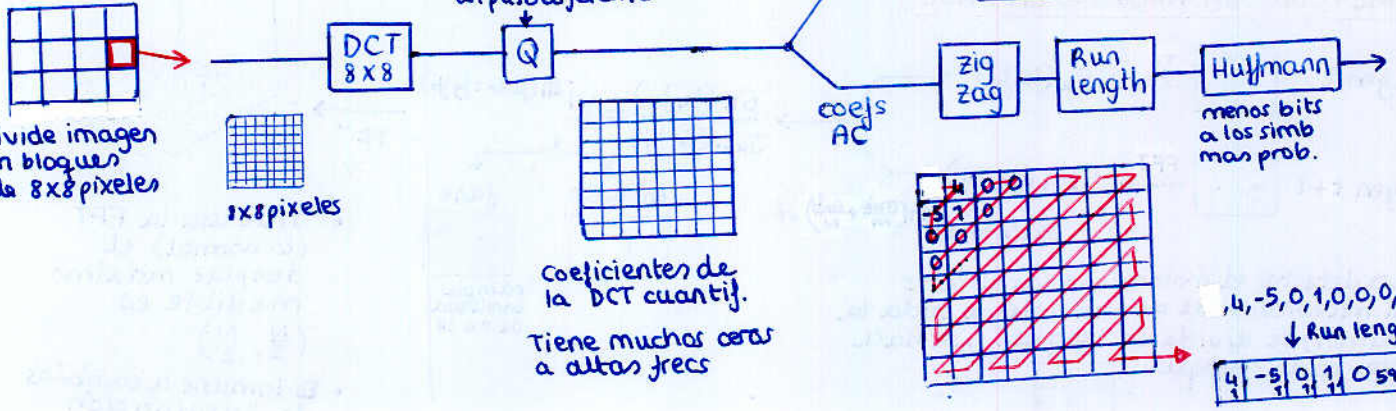
con $C(u), C(v) = \begin{cases} 1/\sqrt{2} & \text{para } u,v = 0 \\ 1 & \text{resto} \end{cases}$

Con la DCT se obtiene un espectro REAL



Con valores bajos a frecs altas

· Esquema JPEG



MPEG

Moving Pictures Expert Group (1982)

Bases { JPEG: Eliminar redundancia espacial
H.261: Estándar de videoconferencia, elimina redundancia temporal (diferencia compensada en movimiento)

Nuevas mejoras: · acceso aleatorio · búsqueda rápida · retroceso

Idea:

- La resta entre fotograma t+1 y fotograma t tiene menor potencia que el propio fotograma (sólo en cambios) y tiene menos potencia aún si antes de restar se compensa el movimiento.
- Además conviene hacer esta resta por bloques, cada uno con su movimiento individual.
- La diferencia compensada resultante de cada bloque puede comprimirse en JPEG con muy pocos bits (alta redundancia)

Tipos de fotograma: I: intra frame
P: predictive
B: bidirectional

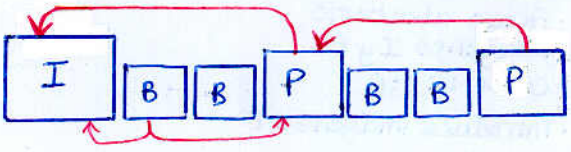
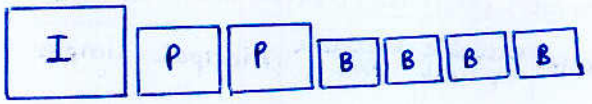


Imagen I: Se codifica íntegramente (keyframe) (conviene que coincida con cambios de escena)

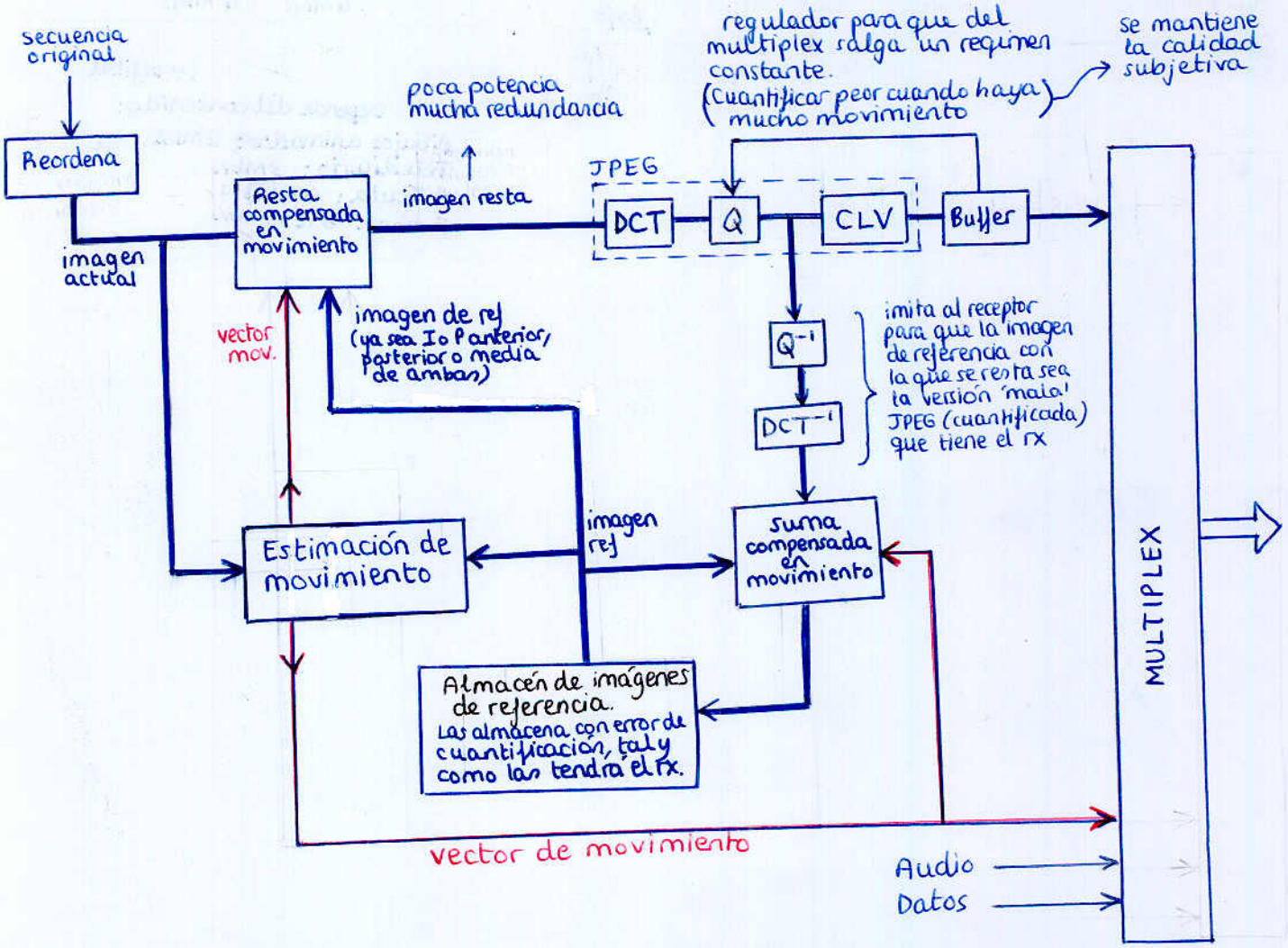
Imagen P: Cada macrobloque se codifica o bien íntegramente o bien diferencialmente respecto a I o P anterior

Imagen B: Cada macrobloque se codifica o bien íntegramente o bien diferencialmente respecto al I o P anterior, posterior, o media de ambas.

nota: un macrobloque (16x16) son cuatro bloques JPEG y es la unidad en la que se compensa el movimiento
(si es en color se usa 4:2:0 por lo que
1 MB = 4 bloques 8x8 luma
1 bloque 8x8 croma azul
1 bloque 8x8 croma roja)



Esquema del codificador



Formatos MPEG

MPEG1 : Vídeo Digital en CD
MPEG2 : Televisión
MPEG4 : Videoconferencia
↳ estimación de mov. de Audio Visual Objects
MPEG7 : Bases de datos audiovisuales (Detección de caras, ...)

MPEG1 vs MPEG2

	MPEG 1	MPEG 2
	1992	1994
	video CDRom	TV digital
	288 x 360	576 x 720
	25-30 fps	
	1'5 Mbps	15 Mbps
calidad:	VHS	NTSC/PAL
compresión:	20-30	30-40

MPEG-2 : Jerarquía de perfiles y niveles

· **Nivel**:

	res horiz.		res vert.		res temp (fps)
- Bajo:	352	x	288	,	30 fps
- Principal:	720	x	576	,	30 fps
- Alto 1440:	1440	x	1152	,	60 fps
- Alto	1920	x	1152	,	60 fps

· **Perfil**: conjunto de algoritmos añadidos

- Simple: · Video entrelazado
  · Acceso aleatorio
  · Imágenes I y P
  · Color 4:2:0
- Principal: · Introduce imágenes B
- Escalable en S/N:
  · permite codificar en función de S/N
- Escalable espacialmente
  · permite codificar en función de la resolución
- Alto · color 4:2:2 (sólo diezma en horizontal)

· **De la intersección perfil/nivel se obtiene una tasa**

Perfil / Nivel	Alto	escalable espacial	escalable S/N	Principal	Simple
Alto	100 Mb/s			80 Mb/s → HDTV	
Alto 1440	80 Mb/s	60 Mb/s		60 Mb/s	
Principal	20 Mb/s		15 Mb/s	15 Mb/s	15 Mb/s
Bajo			4 Mb/s	4 Mb/s	

TV Digital

Depende del contenido:
Dibujos animados: 2 Mb/s
Telediario: 3 Mb/s
Película: 4 Mb/s
Futbol: 6 a 8 Mb/s

TEMA 5: FILTRADO

1. FILTROS LINEALES E INVARIANTES (LSI)

$x[m,n] \longrightarrow \boxed{h[m,n]} \longrightarrow y[m,n] = x[m,n] * h[m,n]$

$$= \sum_{k=-\infty}^{\infty} \sum_{l}^{\infty} x[k,l] \cdot h[m-k, n-l]$$

$$Y(e^{j\omega_x}, e^{j\omega_y}) = X(e^{j\omega_x}, e^{j\omega_y}) \cdot H(e^{j\omega_x}, e^{j\omega_y})$$

Filtros IIR
- implementables con recurrencia
- Difícil verificar estabilidad (polos de H(z) en círculo unidad)
- Nunca fase lineal
- Menos operaciones

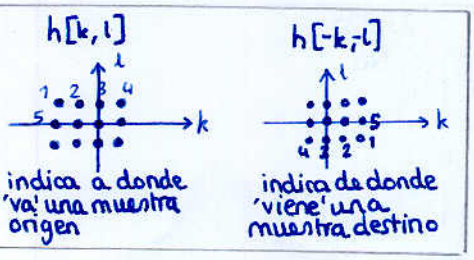
Filtros FIR
- nº finito operaciones/pixel
- estables
- fase lineal si son simétricos

h[m,n] simétrica: fase lineal
la imagen no se deforma, sólo se desplaza

h[m,n] simétrica respecto al origen
- → fase cero
- si N impar: desplaz 0 pixeles
- → si N par: desplaz $\pm\frac{1}{2}$ pixeles (no puede ser simétrico respecto al origen)

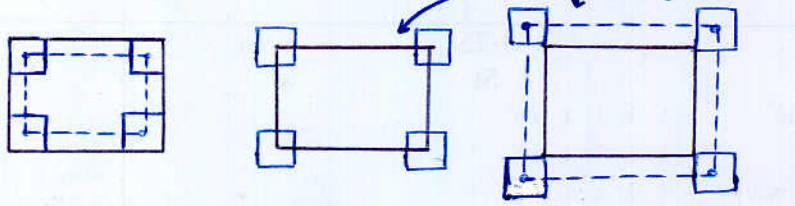
Implementación

$y[m,n] = \sum\sum x[k,l] \cdot h[m-k, n-l]$



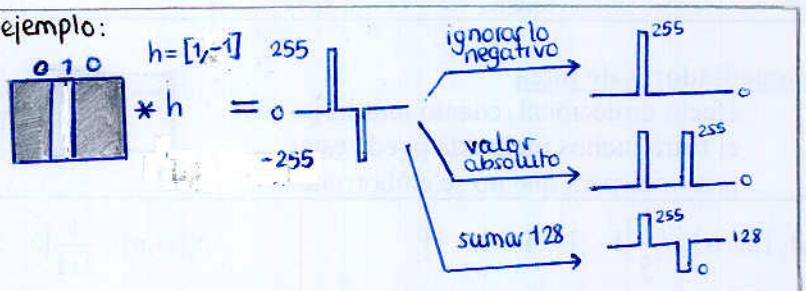
Tratamiento de bordes (3 opciones)

relleno: -espejo -ceros -gris medio



Efectos de precisión finita
- cuantificación de coeficientes
- redondeo en operaciones ($\sigma^2 = \Delta^2/12$) (para mejorar S/N aumentar el MD antes de filtrar)
- saturaciones (resultado < 0 o' > 255)

ejemplo:



Implementación eficiente

- **Simetría** (pixel, coef filtro)
  par: $a \cdot s_1 + a \cdot s_2 = a \cdot (s_1 + s_2)$
  impar: $a \cdot s_1 + (-a) \cdot s_2 = a \cdot (s_1 - s_2)$

ej: $h[m,n] = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$

- **Separabilidad**

$h[m,n] = h_1[m] \cdot h_2[n] \Rightarrow x[m,n] * h[m,n] = (x[m,n] * h_1[m]) * h_2[n]$
separo las dimensiones (filtro 1º en x y luego en y)

$(M_1+M_2-1) \times (N_1+N_2-1)$ ops/pixel

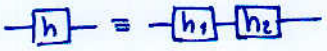
ej $h[m,n] = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \frac{1}{5}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \cdot \frac{1}{5}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}$

- **Descomposición en cascada**

$h[m,n] = h_1[m,n] * h_2[m,n] \Rightarrow x[m,n] * h[m,n] = x[m,n] * h_1[m,n] * h_2[m,n]$

$M_1 N_1 + M_2 N_2$ ops/pixel

($M_1 \times N_1$, $M_2 \times N_2$)



- **Recursividad** — todos los elementos iguales

ej $h = [1\ 1\ 1\ 1\ 1]$

paso 1 = a + b + c + d + e

paso 2 = b + c + d + e + f
= paso 1 − a + f

sumar la intensidad del pixel que entra y restar la del que sale

ejemplo:

$$h[m,n] = \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} * \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Filtros suavizadores (paso bajo)

- suavizan los bordes (emborronan)
- utilidad: reducción del ruido
- condición: conservar brillo medio $\sum_m \sum_n h[m,n] = 1$

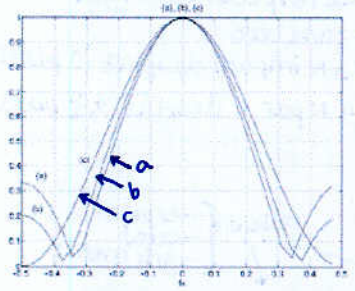
$$\sigma_i^2 \longrightarrow h[m,n] \longrightarrow \sigma_{out}^2$$

$$\sigma_{out}^2 = \sigma_i^2 \underbrace{\sum_m \sum_n |h[m,n]|^2}_{\text{factor de reducción de ruido}}$$

Promediadores de área

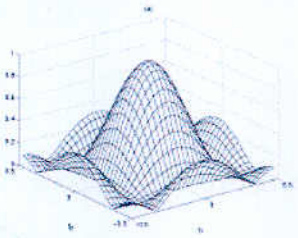
el divisor es la suma de los coeficientes

Reducción de ruido
Separable



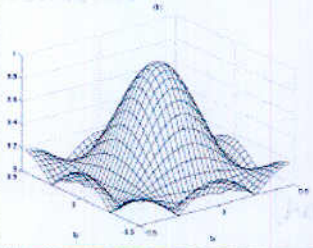
1/9
Si

$$h_a[m,n] = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



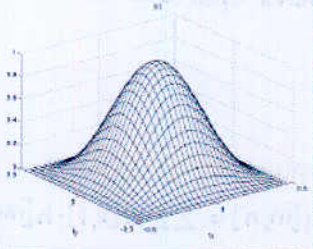
3/25
No

$$h_b[m,n] = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



9/64
Si

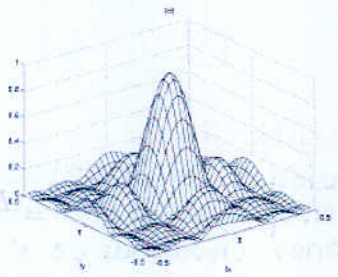
$$h_c[m,n] = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



1/25
Si

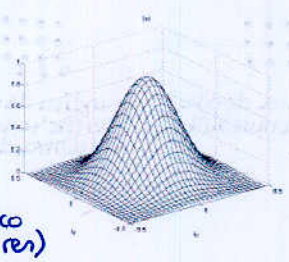
$$h_d[m,n] = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Todo unos: mas estrecho pero más lóbulos secundarios



23/225
No

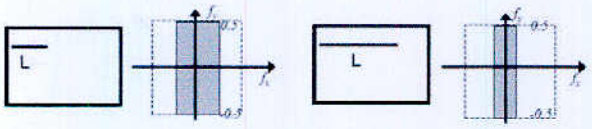
$$h_e[m,n] = \frac{1}{120}\begin{bmatrix} 0 & 1 & 2 & 1 & 0 \\ 1 & 7 & 13 & 7 & 1 \\ 2 & 13 & 24 & 13 & 2 \\ 1 & 7 & 13 & 7 & 1 \\ 0 & 1 & 2 & 1 & 0 \end{bmatrix}$$



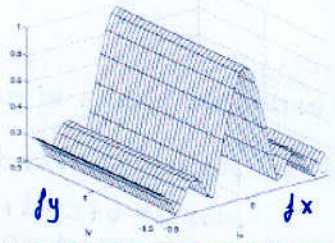
ø como enventanar
→ lóbulo más ancho
→ mejor margen dinámico
(lób. secundarios menores)

Promediadores de línea

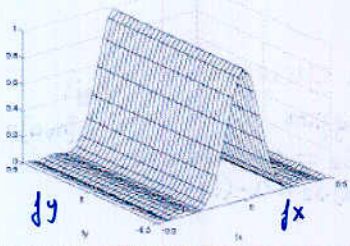
efecto direccional, cuanto más largo sea el filtro menos inclinada puede estar una línea para que no se emborrone



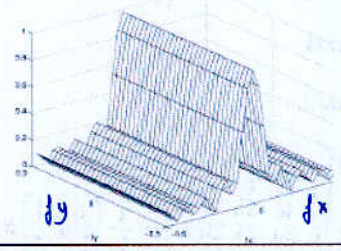
$$h_f[m,n] = \frac{1}{5}[1 \quad 1 \quad 1 \quad 1 \quad 1]$$



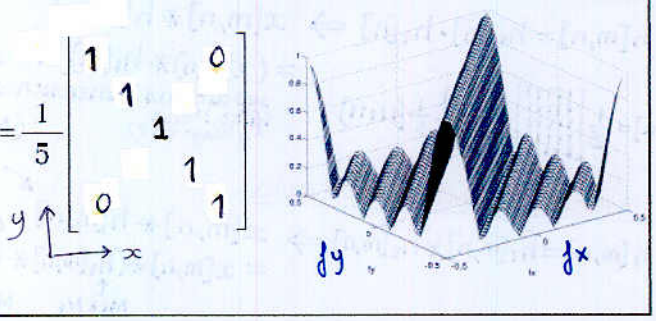
$$h_h[m,n] = \frac{1}{111}[2 \quad 5 \quad 14 \quad 22 \quad 25 \quad 22 \quad 14 \quad 5 \quad 2]$$



$$h_g[m,n] = \frac{1}{9}[1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1]$$



$$h_i[m,n] = \frac{1}{5}\begin{bmatrix} 1 & & 0 \\ & 1 & \\ & 1 & \\ & 1 & \\ 0 & & 1 \end{bmatrix}$$



Filtros realzadores de contorno (paso alto)

- Problema: amplifican también el ruido
- De nuevo conservan el brillo medio

$$\sum_m \sum_n h[m,n] = 1$$

creación de h[m,n]:

$R = I + \alpha\, AF$

$= I + \alpha\,(I - BF)$

$= (1+\alpha)\cdot I - \alpha \cdot BF$

I =imagen original
R = imagen realzada
AF = imagen de altas frecuencias
BF = imagen de bajas frecuencias

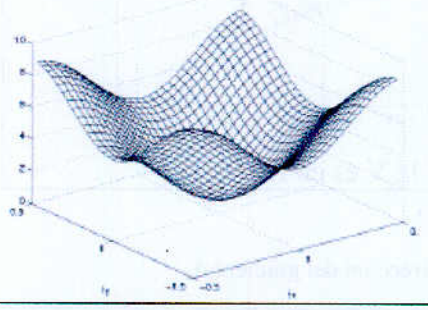
ejemplo $\alpha = 9$

$$h_i[m,n] = 10\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix} - 9\left[\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}\right] = \begin{bmatrix}-1&-1&-1\\-1&9&-1\\-1&-1&-1\end{bmatrix}$$

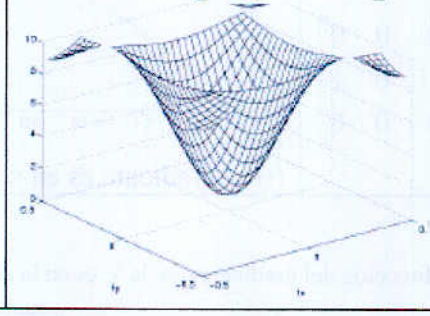
ORIGINAL

REALZADA

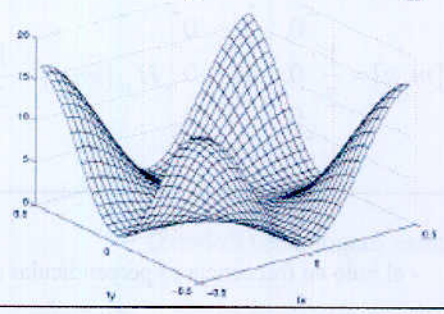
$$h_k[m,n] = \begin{bmatrix}0&-1&0\\-1&5&-1\\0&-1&0\end{bmatrix}$$



$$h_i[m,n] = \begin{bmatrix}-1&-1&-1\\-1&9&-1\\-1&-1&-1\end{bmatrix}$$



$$h_m[m,n] = \begin{bmatrix}1&-2&1\\-2&5&-2\\1&-2&1\end{bmatrix}$$



Filtros direccionales

Para detectar bordes en una dirección

$$H_{90^\circ}[m,n] = \begin{bmatrix}1&0&-1\\1&0&-1\\1&0&-1\end{bmatrix} \qquad H_{45^\circ}[m,n] = \begin{bmatrix}0&-1&-1\\1&0&-1\\1&1&0\end{bmatrix}$$

Filtros gradientes

Sirven para detectar bordes

El gradiente da información en
- dirección
- modulo

$$\nabla f(x,y) = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

$$|\nabla f(x,y)| = \begin{cases} G_1 = \sqrt{g_h^2 + g_v^2} \\ G_2 = |g_h| + |g_v| \end{cases}$$ ← no es exacto pero es rápido

$$TF\{\nabla f(x,y)\} = \left|j\omega_x F(\omega_x,\omega_x), j\omega_y F(\omega_x,\omega_x)\right|$$

Implementación

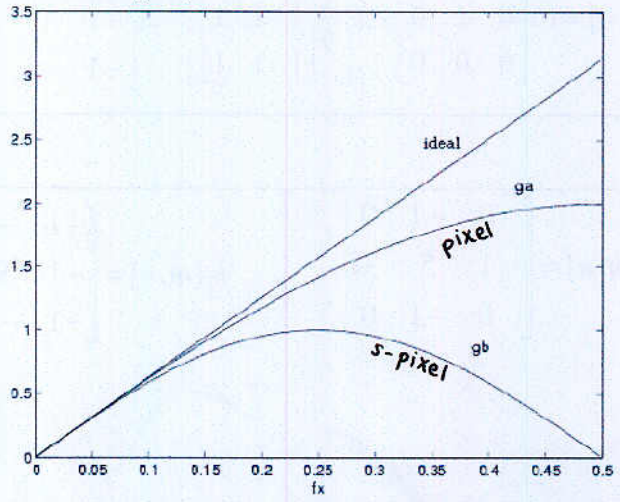
píxel gradient (desplazan 0.5 píxeles)

$$V_{ga}[m,n] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix} \quad H_{ga}[m,n] = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

s-píxel gradient (no desplazan) (el divisor es igual a la separación entre los píxeles que se restan)

$$V_{gb}[m,n] = \frac{1}{2}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \quad H_{gb}[m,n] = \frac{1}{2}\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Ideal = |j2πf|



(si el gradiente es en x, la V es en fx)

Gradientes diagonales (Roberts)

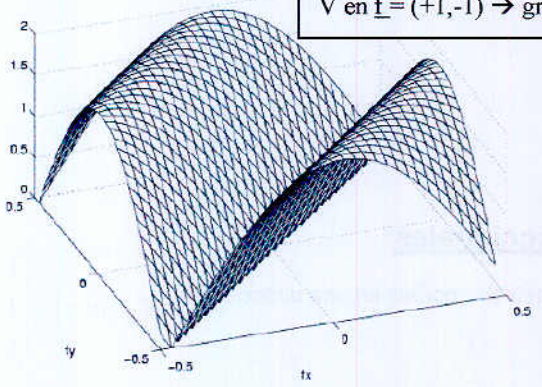
- el nulo en frecuencia es perpendicular a la dirección del gradiente (i.e. la V es en la dirección del gradiente)

$$D_{gc}[m,n] = \frac{1}{\sqrt{2}}\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

grad

y ↑ → x

V en f = (+1,-1) → gradiente en X = (+1,-1)



Gradientes con promediado (Sobel, Prewitt, …)

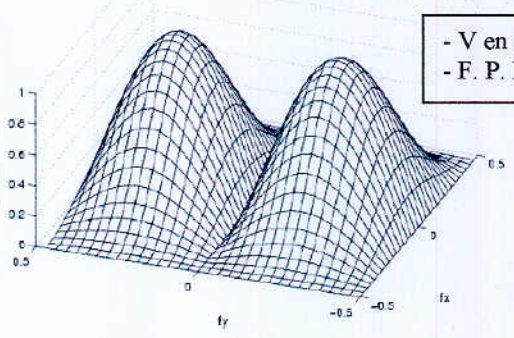
- en una dirección promedian (eliminan ruidos) y en la otra hacen el gradiente (detectan bordes)

$$V_{gd}[m,n] = \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

grad

y ↑ → x

- V en fy → gradiente en y
- F. P. Bajo en fx → promedio en x



¿Divisor?
→ 1+2+1 → ¼ para mantener el brillo medio
→ ½ por la distancia de la resta

Filtrado de imágenes en color

Hacer el proceso para los 3 colores

Filtros no lineales

Soluciona algunos problemas de los filtros lineales

Tipos:

- **reductores de ruido adaptativos**
  (reducen ruido sin emborronar, para ello detectan bordes y quitan ruido sólo en las zonas uniformes- la gente prefiere un borde ruidoso que borroso)
- **filtros de estadísticos ordenados**
  (ejemplos: filtros de mediana, de dilatación, de erosión, …)
- **reductores de ruido estacionario + impulsivo**

ejemplo: filtro de mediana

Coge los píxeles de una zona y los ordena de menor a mayor, y se queda con la mediana.
(si hay nº par tomamos la media de ambos centrales)

Logra eliminar el ruido impulsivo (sal y/o pimienta), ya que siempre se queda en los extremos

11	12	12	13	14
10	11	12	13	15
10	10	14	16	15
12	9	15	8	9
11	9	8	9	9

8 9 10 11 (12) → 12 13 14 15 16

Ruidosa Media 3x3 Mediana 3x3

Este tipo de filtros (ejemplo: cogiendo el máximo -dilatación- o el mínimo -erosión-)son la base de *algoritmos de morfología matemática.*

Ruido uniforme: funciona bien filtro lineal (promediador)
Ruido impulsivo: funciona bien filtro no lineal (ej filtro de mediana)

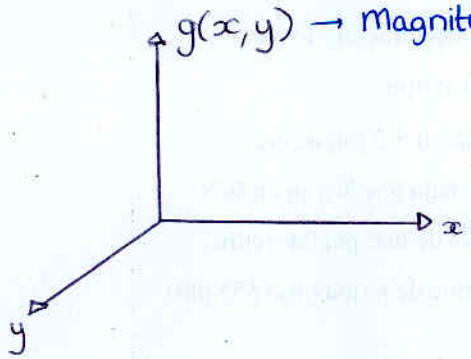
Ruido uniforme + impulsivo : combinar ambos con un algoritmo
ejemplo

11	12	12	13	14
10	11	12	13	15
10	10	14	16	15
12	9	15	8	9
11	9	8	9	9

8 9 (10 11 12 13 14) → media → 12 15 16

Tema 1. Introducción

Concepto de Imagen



$g(x,y)$ → Magnitud representada:
- Radiación reflejada (luz óptica, infrarrojos, ...)
- Transparencia
- Opacidad

En realidad la imagen es continua, pero para trabajar con ella la muestreo, y la imagen se convierte en una matriz de números

cada número suele codificarse con 8 bits (0-255) escala de grises
cada número es un píxel

Tipos de imágenes

Naturaleza:
- continuas
- discretas
- mixtas (ej TV)

Bandas:
- monobanda
- multibanda

Animación:
- estáticas
- dinámicas

Operaciones de TDI

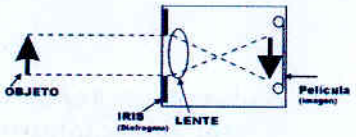
- Formación / Captura
- Almacenamiento / Transmisión
- Modificación
- Extraer información
- Visualización / reproducción

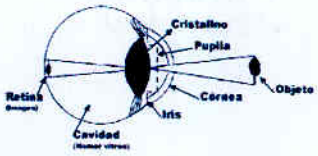
Formación de la imagen óptica

- cámara, lente, foco, círculo de difusión, diafragma, profundidad de campo → ver apuntes de imagen y sonido

recuerda:
- distancia focal mayor (zoom) → menor P.C.
- mayor apertura diafragma → menor P.C.
- plano enfocado más lejano → más P.C.
- mayor P.C. por detrás

Elementos psicofísicos de la visión





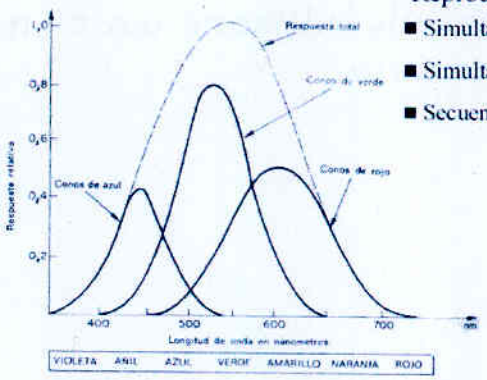
Estructura de la retina

- Bastones: 1000 veces más sensibles, 120 millones
- Conos: sensibles al color, 6 millones
- Zona de mayor concentración: fóvea

Características de la visión

- Resolución espacial: 0.5-2 mn de arco
- Detalle pequeño y baja resolución en B/N
- Adaptación al nivel de luz: pupila+retina
- Sensibilidad depende de λ (máximo 555 nm)

El color



Reproducción del color

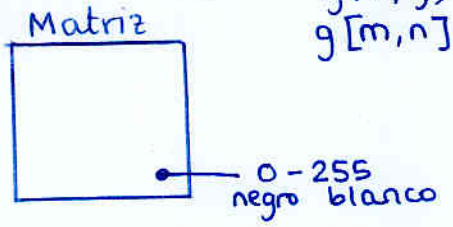
- Simultánea en el mismo punto
- Simultánea yuxtapuesta espacialmente (promediado espacial)
- Secuencial en el mismo punto (promediado temporal)

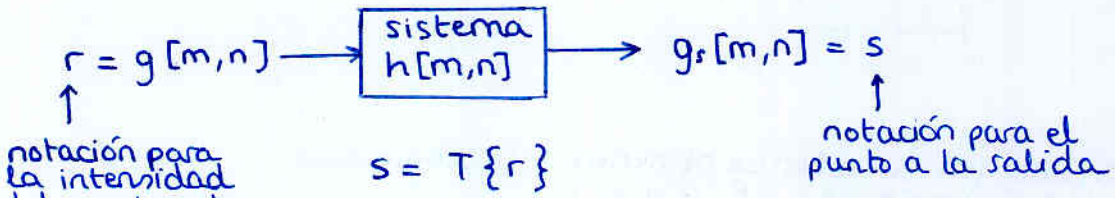
Tema 2. Transformaciones de Intensidad

2.1- Introducción
2.2- Histogramas
2.3- Remapeado de grises
2.4- Espacio de color
2.5- Tipos de imágenes
2.6- Otras transformaciones de intensidad
2.7- Aritmética de imágenes

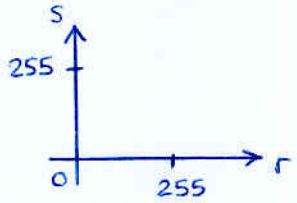
2.1. Introducción

$g(x,y)$ → imagen continua
$g[m,n]$ → imagen discreta





$s = T\{r\}$



Función de transferencia

→ Efecto en la imagen
→ Efecto en el histograma

Las transf. de intensidad son:

- **puntuales**: cada punto de la imagen destino depende sólo de un punto de la imagen origen
- sin memoria: cada punto de la imagen destino depende sólo del mismo punto (misma posición) de la imagen origen.

2.2 Histogramas

Imagen → Proceso estocástico
→ Función de densidad de probabilidad – cómo de probable es que un determinado nivel aparezca

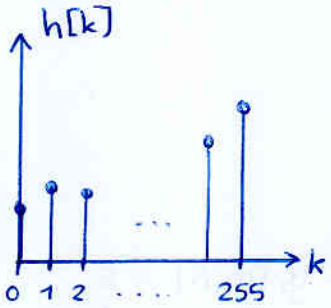
↓
Hay que estimarla a partir de las muestras
↓
Histograma

k	h[k]
0	h_0
1	h_1
⋮	⋮
255	h_{255}

número de píxeles que hay en cada nivel

En forma de probabilidad

$$P_k = \frac{h_k}{N} \quad \in [0,1]$$



Para una imagen en color se obtienen 3 histogramas (para las 3 imágenes R, G y B)

2.3 Remapeado de grises

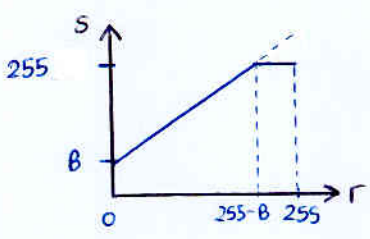
2.3.1 Tr. lineales

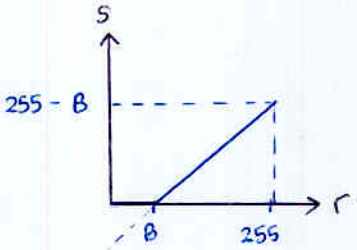
- **Brillo**:
  cambio de la intensidad media de la imagen

$$\text{media} = \frac{\sum_m \sum_n I(m,n)}{M \cdot N}$$

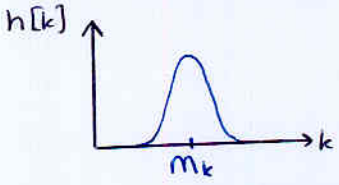
simplemente sumar un valor

$$s = r \pm B$$

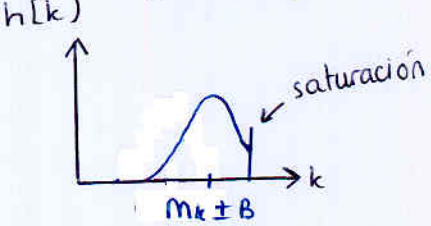




Cambio en el histograma



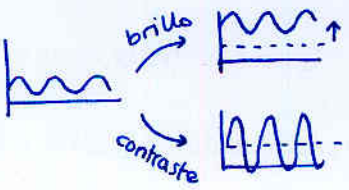
↓ se desplaza el histograma



En la imagen RGB se suman a todos la misma cantidad

$$s_R = r_R + B$$
$$s_G = r_G + B$$
$$s_B = r_B + B$$

· Contraste

ej 

Cambio del margen dinámico de la imagen

$S = C \times r$ cambia constante Y BRILLO!

Para no cambiar la media hay que hacer

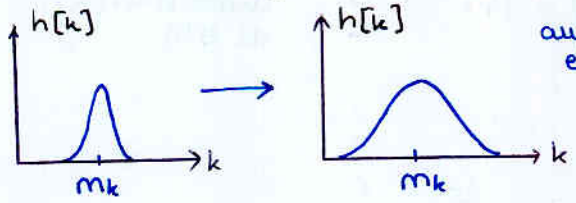
$$\boxed{S = (r - m_k) \cdot C + m_k}$$

o una versión más rápida (aunque funciona peor)

$$s = (r - 128) \cdot C + 128$$

En RGB hay que repetir la operación sobre cada componente

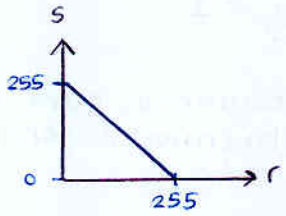
Efecto sobre el histograma:



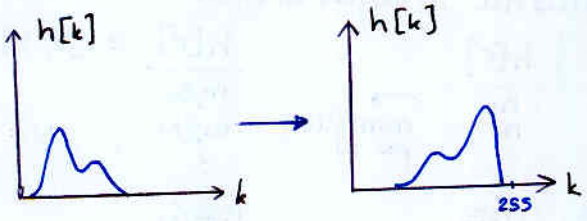
aumenta o disminuye el contraste

· Negativo

$$\boxed{S = 255 - r}$$



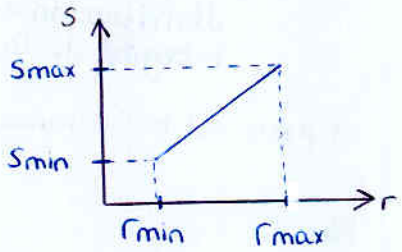
Efecto sobre el histograma



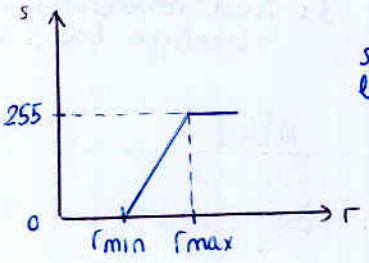
le da la vuelta

· Cambio de margen dinámico

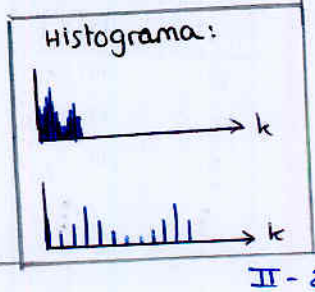
Controlando niveles mínimos y máximos de salida



lo típico es $S_{min} = 0$ y $S_{max} = 255$



y el histograma se expande hasta los límites



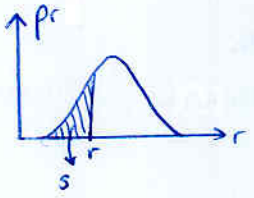
2.3.2. Transformaciones no lineales

· Ecualización del histograma

Se trata de conseguir que la imagen de salida tenga un histograma constante

$$\boxed{s = \int_0^r p_r(w)\, dw}$$

función densidad de probabilidad



¿Porqué consigue ecualizar el histograma?

si $r \xrightarrow{T\{r\}} s$ monótona creciente $\Longrightarrow p_s(s) = p_r(r) \cdot \dfrac{dr}{ds}$

No vemos la demostración de esto

por tanto

$$s = \int_0^r p_r(w)\,dw \rightarrow \frac{ds}{dr} = p_r(r) \rightarrow \frac{dr}{ds} = \frac{1}{p_r(r)}$$

por tanto $p_s(s) = p_r(r) \cdot \dfrac{1}{p_r(r)} = 1$

En la práctica, usamos el histograma para estimar la fdp y no se consigue que sea totalmente constante

Algoritmo práctico:

origen destino

1° Histograma imagen origen

r	h[r]
0	n_0
1	n_1
⋮	
255	n_{255}

normalizo por N

$h[r] \in [0,1]$

n_0/N
n_1/N
⋮
n_{255}/N

N = número total de pixeles

2° Histograma acumulado

$$A[k] = 255 \cdot \sum_{l=0}^{k} h[l]$$

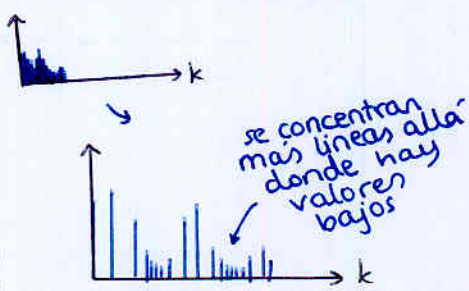
$h[l]$: 0...1

A[k] será el estimador de la función de distribución (recuerda: la función de distribución es la integral de la fdp)

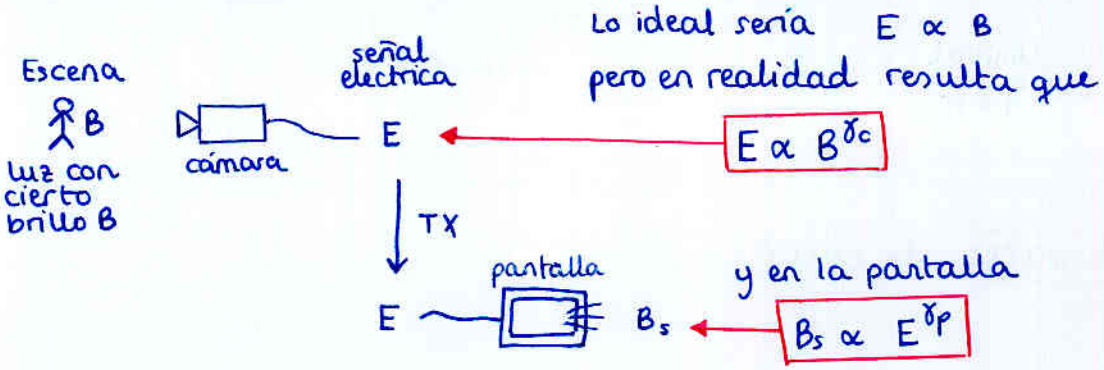
3: Redondeándolo obtengo la s de cada r

r	h[r]	A[k]	s
0	n_0/N	$(n_0/N) \cdot 255$	
1	n_1/N	$\left(\frac{n_0+n_1}{N}\right) \cdot 255$	
2	n_2/N	$\left(\frac{n_0+n_1+n_2}{N}\right) \cdot 255$	
⋮	⋮	⋮	

Aspecto del histograma



· Corrección Gamma



Por tanto en total tenemos que

$$B_s \propto B^{\gamma_c \gamma_p}$$

Interesaría un corrector de gamma

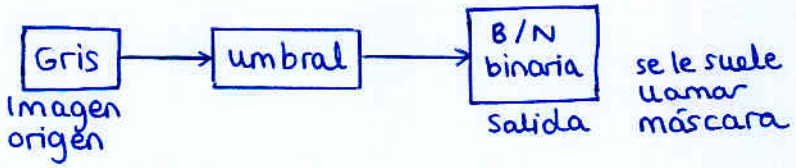
tal que $\gamma_c \cdot \gamma_p \cdot \gamma = 1$

↳ gamma de las pantallas: fijado por ley en un margen

El corrector gamma hace

$$s = \left(\frac{r}{255}\right)^{\gamma} \cdot 255$$

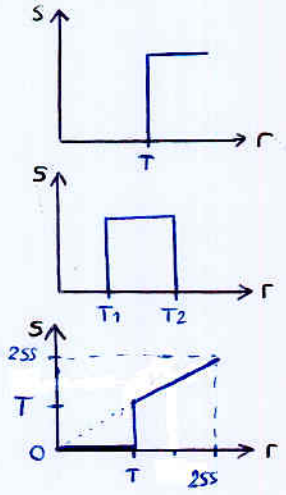
2.3.3. Umbralización



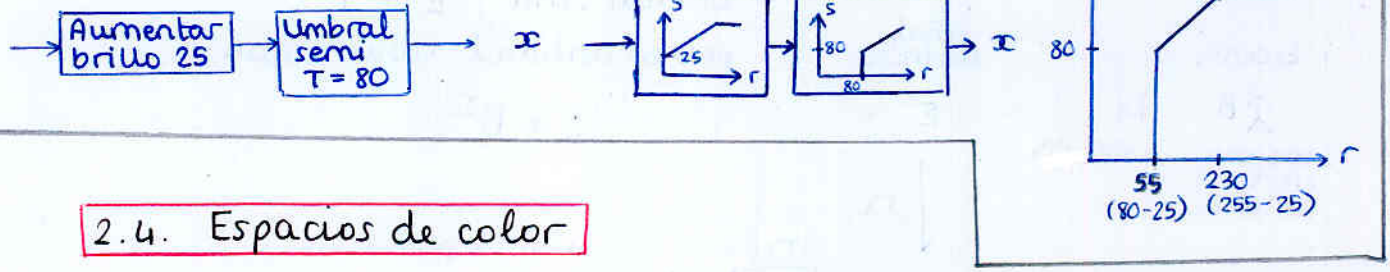
- Umbral simple: $s = \begin{cases} 0 & r < T \\ 255 & r > T \end{cases}$

- Umbral doble: $s = \begin{cases} 0 & r < T_1 \text{ o } r > T_2 \\ 255 & T_1 \leq r \leq T_2 \end{cases}$

- Umbral semi: $s = \begin{cases} 0 & r < T \\ r & r \geq T \end{cases}$



ejercicio

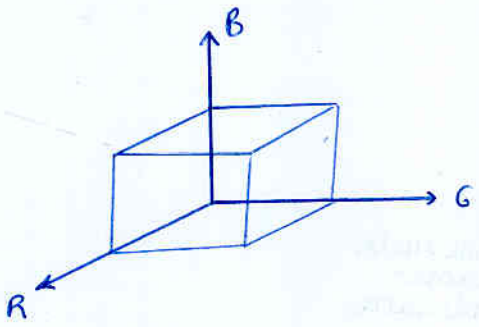


2.4. Espacios de color

2.4.1 RGB

Red, Green, Blue : colores primarios

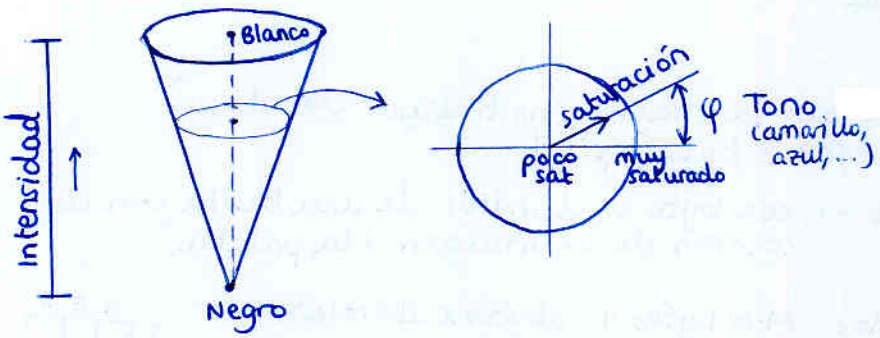
R	G	B		
255	0	0	Rojo	→ Primarios
0	255	0	Verde	
0	0	255	Azul	
0	255	255	Cyan	→ complementarios
255	0	255	Magenta	
255	255	0	Amarillo	
0	0	0	Negro	
255	255	255	Blanco	

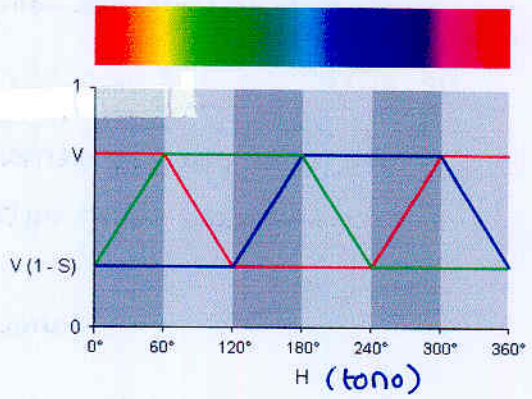


2.4.2 HSI

Hue (Tono), Saturation, Intensity $I = \frac{R+G+B}{3}$

Se parece más a la percepción humana del color





2.4.3. Y U V

El ojo es más sensible a la luminancia

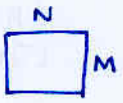
Y : Luminancia
U, V : Crominancias

$Y = 0'3R + 0'59G + 0'11B \longrightarrow$ Luminancia

$U = K_1 \cdot (B - Y)$
$V = K_2 \cdot (R - Y)$

2.5 Tipos de imágenes

- **De grises**
  - 1 byte/pixel
  - tamaño MxN bytes
  - (la binaria también una 1 byte/pixel aunque bastaría con 1 bit)
- **RGB**
  - 3 bytes/pixel
  - tamaño 3MN bytes

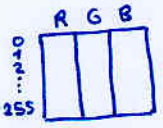


- **De paleta o falso color**

  Si sólo queremos 'visualizar' la imagen (no trabajar con ellas) en color pero sin almacenar 3 bytes/pixel

  cada pixel es un byte → ese byte es el índice de una tabla con los colores de la imagen (la paleta)

  tamaño: M·N bytes + 3 bytes x 256 colores → Paleta de 256 colores



  ej típico: matriz de cámara térmica con paleta de azul a rojo

  Nota: sí se pueden hacer algunos procesos fáciles cambiando únicamente la paleta
  ej → aumentar brillo

Conversiones

- De RGB a
  - Grises → Tomando la luminancia $Y = 0'3R + 0'59G + 0'11B$
  - Paleta →
    - construir una paleta
    - cambiar la matriz por una de índices

- De grises a
  - componentes R,G,B a RGB → trivial
  - Luminancia a RGB → no podré recuperar el color
    - copiar la matriz 3 veces en R, G y B. Se verá gris
  - Paleta →
    - construir paleta de grises
    - la matriz puede seguir siendo la misma

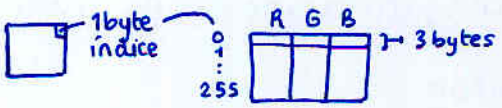
	R	G	B
0	0	0	0
1	1	1	1
2	2	2	2
⋮			
255	255	255	255

- De paleta a
  - RGB: copiar lo que haya en la entrada apropiada en RGB a RGB
  - grises:
    - tomar la luminancia de la paleta
    - ó
    - tomar los índices

Ejercicio: Se calculan las componentes HSI de una imagen RGB
¿Qué significa si en todos los píxeles S = 1?

Significa que todos los pixeles tienen la saturación al máximo, es decir que no tienen ninguna componente de blanco, es decir que al menos una de las 3 componentes RGB es nula en cada pixel.

Creación de paletas



- **Método producto**

1 byte → 8 bits

Uso una paleta fija ej:

8 niveles R	8 niveles G	4 niveles B ↑ perjudica al azul	→ 8×8×4 = 256 colores
6 niveles R	8 niveles G ↑ el ojo ve mejor el verde	6 niveles B	→ 6×8×6 = 252 colores
6 niveles R	6 niveles G	6 niveles B	→ 216 niveles

- **Métodos adaptativos**

La paleta depende de la imagen

① Se pone una paleta inicial → método producto

② Se recorre la imagen RGB y cada pixel se asigna a la entrada de la paleta cuyo color sea más cercano

$$d_c = \sqrt{(R-R_p)^2 + (G-G_p)^2 + (B-B_p)^2}$$

③ Se actualiza cada entrada de la paleta con la media de RGB de los píxeles asignados a esa entrada

④ Volver a 2 hasta que la paleta no cambie

2.6 Otras transformaciones de intensidad

- Filtros de color
  - Balance de blancos
    - Captar imagen blanca
    - Calcular $factor_R$ de forma que $factor_R \times R = 255$
      $factor_G$ $factor_G \times G = 255$
      $factor_B$ $factor_B \times B = 255$
    - A partir de ahora multiplicar siempre por esos factores
  - Filtros de color (efectos)
    - multiplicar cada componente RGB por un factor diferente para cambiar el color de la imagen
  - Eliminar el efecto de la iluminación no uniforme
    - Tomas una imagen blanca
    - A CADA pixel le calculas un factor (o 3 si la imagen es RGB) para conseguir que la imagen sea blanca

2.7 Aritmética de imágenes

- Suma: ejemplo: $\frac{A+B}{2}$ ejemplo2: para hacer fundidos $\alpha A + (1-\alpha)B$
- Resta: útil para ver diferencias (detectar movimiento)
  $|A-B|$ detecta si hay o no movimiento
  $A-B+128$ detecta el SENTIDO del movimiento
- operaciones lógicas: AND, OR, XOR — útil para trabajar con máscaras

Tema 3. Transformaciones geométricas

3.1 Introducción

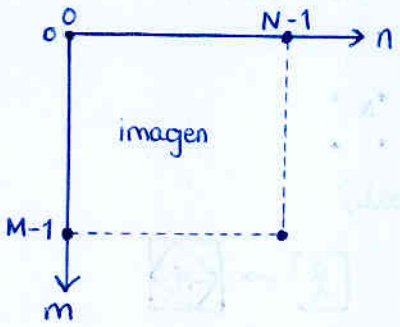
T. Intensidad: cambian el valor pero no la posición ej $s = r + 8$

T. Geométricas: cambian la posición pero no el valor ej $(m,n) = (m', n') + (1,1)$

son también transf. puntuales con memoria

- puntuales: cada pixel sólo depende de un pixel
- con memoria: cada pixel puede depender de un pixel en OTRA posición

Eje de coordenadas:



Típicamente vamos buscando de qué pixel pescar cada posición del destino

$m' = ..f(m,n)..$
$n' = ..g(m,n)..$
$im\_dest[m,n] = im\_orig[m',n']$

3.2 Simetrías

- Espejo
- Arriba-abajo
- Rotación 90° horario
- Rotación 90° antihorario
- Rotación 180°

Éstas son las únicas "transformaciones perfectas"

- Espejo:

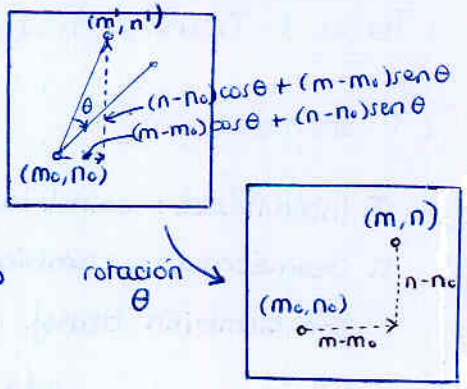
$$\begin{cases} m' = m \\ n' = N-1-n \end{cases} \qquad y[m,n] = x[m',n'] = x[m, N-1-n]$$

- Arriba-abajo: $y[m,n] = x[M-1-m, n]$
- Rotación 90° horario: $y[m,n] = x[M-1-n, m]$
- Rotación 90° antihorario: $y[m,n] = x[n, N-1-m]$
- Rotación 180°: $y[m,n] = x[M-1-m, N-1-n]$

3.3 Giros arbitrarios

$$m' = (m - m_0)\cos\theta + (n - n_0)\operatorname{sen}\theta + m_0'$$
$$n' = (n - n_0)\cos\theta - (m - m_0)\operatorname{sen}\theta + n_0'$$

$(m_0', n_0') \rightarrow$ Centro de la imagen origen
$(m_0, n_0) \rightarrow$ Centro de la imagen destino



Problemas

① • la rejilla de muestreo puede no ser ortogonal (m', n' pueden no ser enteros)

② • la imagen girada no es un rectángulo ni tiene tamaño $M \times N$

Soluciones

① · Redondear $(m', n') \rightarrow$ vecino más próximo

· Algún tipo de interpolación (interpolación bilineal)

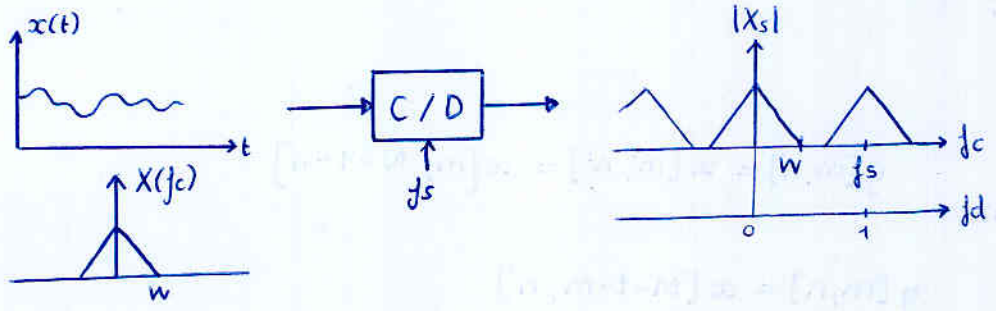
② · Agrandar la imagen y rellenar con un color

· Recortar la imagen girada

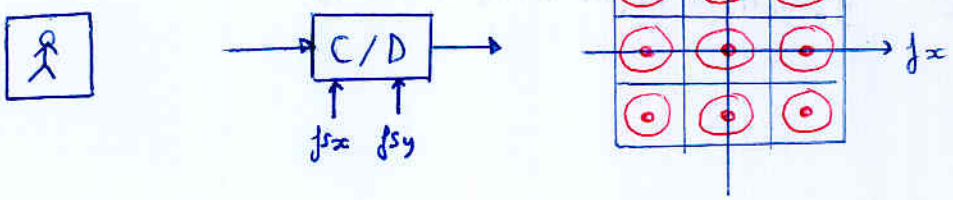
3.4 Diezmado de imágenes

Recuerda:

Muestreo de una señal



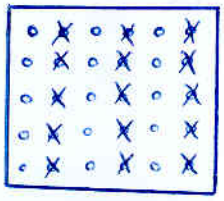
Muestreo de una imagen



Diezmado:

Diezmar por factor K es eliminar K-1 pixeles de cada K pixeles

ejemplo: diezmado K=2 en horizontal



ejemplo: diezmado en las 2 dimensiones

$$y[m,n] = x[2m, 2n] \qquad \frac{M}{2} \times \frac{N}{2} \qquad M \times N$$

$$0 \le m \le \frac{M}{2} - 1$$

$$0 \le n \le \frac{N}{2} - 1$$

Al diezmar el espectro se ensancha → ¡Ojo con el aliasing!

Antes de diezmar hay que filtrar paso bajo

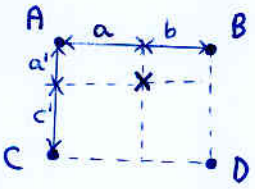
3.5 Interpolación

Aumentar el número de píxeles por un factor K en horizontal, vertical, o ambas.

Interpolar en vertical: insertar K-1 filas entre cada 2
Interpolar en horizontal: insertar K-1 columnas entre cada 2

Interpolar píxeles:

- repetir pixeles (vecino más próximo) (interpolador de orden cero) aparece el efecto de bloque
- interpolación bilineal: bordes suavizados pero imagen más borrosa



$$I = A \cdot b \cdot c' + B \cdot a c' + C \cdot a' b + D a' a$$

uso las distancias lejanas, para que si está cerca de A se pondere más

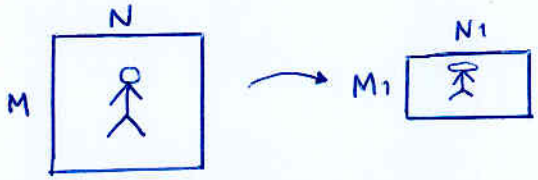
- interpolación ideal
  - insertar ceros
  - filtro paso bajo $f_c = \frac{1}{2L}$ →

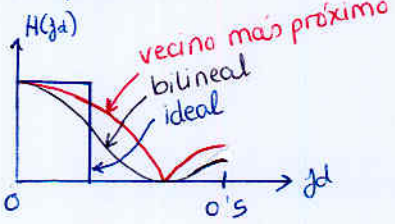
$$h[0,0] = 1$$
$$h[jK, lK] = 0 \qquad [j,l] \neq [0,0]$$

que en [0,0] valga 1
que en [jK, lK] valga 0
los valores intermedios dependen del tipo de interpolación

Al interpolar el espectro se estrecha

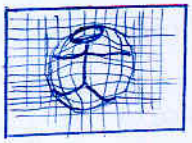
Remuestreo





3.6 Cambios de forma

ej: efecto bola de cristal



- con ley ⟶ $(m', n') = T(m, n)$
  ↑ ley matemática dada

- arbitrarios

transformación afín:

$$\begin{pmatrix} m' \\ n' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} m \\ n \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} \iff \begin{aligned} m' &= am + bn + e \\ n' &= cm + dn + f \end{aligned}$$

Problema: averigüiar 6 parámetros a, b, c, d, e y f

si yo conociera algunas parejas $(m', n') \leftrightarrow (m, n)$
podría calcular a, b, c, d, e y f

Necesito saber a priori 3 puntos del origen que se corresponden con 3 puntos del destino

→ Warping: sólo hay cambio de forma

→ Morphing: cambia la forma y la intensidad de los píxeles

1º Mallar las imagenes con triángulos
2 Cada triangulo es una transf. diferente
3 Con los vertices de cada triángulo se calculan a, b, c, d, e y f para cada triángulo
4 Se aplica la transformación de a, b, c, d, e y f a los pixeles del interior del triángulo destino

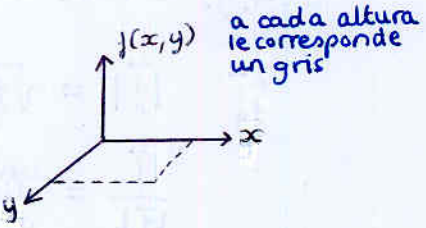
Tema 4. Transformada de Fourier bidimensional

4.1 Señales y sistemas continuos 2D

- **Señales continuas 2D**

Imagen → Señal 2D → 2 variables espaciales (x, y)

$f(x, y)$

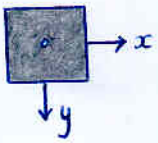


Ejemplos de señales 2D

- impulso unitario

$$\delta(x, y) = \delta(\underline{x}) = \begin{cases} 0 & x \neq 0 \\ \infty & x = 0 \end{cases}$$

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(x,y)\, dx\, dy = 1$$



- escalón

$$u(x, y) = u(\underline{x}) = u(x) \cdot u(y)$$

- recta

en eje x : $s(x, y) = \delta(y)$

en eje y : $s(x, y) = \delta(x)$

oblicua : $s(x, y) = \delta(ax + by + c)$

recta de ecuación $ax + by + c = 0$

- cuadrado

$$C(x, y) = \Pi(x, y) = \begin{cases} 1 \begin{cases} \text{si} & -0'5 < x \leq 0'5 \\ y & -0'5 < y \leq 0'5 \end{cases} \\ 0 \quad \text{resto} \end{cases}$$

ayuda mucho que sea separable

$\Pi(x, y) = \Pi(x) \cdot \Pi(y)$

- círculo

$$O(x, y) = O(\underline{x}) = \Pi\left(\sqrt{x^2+y^2}\right) = \begin{cases} 1 \text{ si } -0'5 \leq \underbrace{\sqrt{x^2+y^2}}_{\text{radio}} \leq 0'5 \\ 0 \text{ resto} \end{cases}$$

- FASORES

- exponencial compleja

$$s(x, y) = e^{j(\Omega_x x + \Omega_y y)} = e^{j(\underline{\Omega}^t \cdot \underline{x})} \quad \text{con} \begin{cases} \Omega_x = 2\pi f_x \\ \Omega_y = 2\pi f_y \end{cases}$$

$$\underline{\Omega} = \begin{pmatrix} \Omega_x \\ \Omega_y \end{pmatrix} \qquad \underline{x} = \begin{pmatrix} x \\ y \end{pmatrix}$$

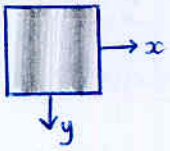
nota:

1D		2D	
tiempo	frec	espacio	frec
t	f	x, y	f_x, f_y
[s]	$[s^{-1}]$ ó [Hz]	[cm]	$[cm^{-1}]$

· sinusoides

$$C(x,y) = \cos(\Omega_x x + \Omega_y y)$$
$$S(x,y) = \operatorname{sen}(\Omega_x x + \Omega_y y)$$

ej: $C(x,y)$ con $\Omega_y = 0$



$$|\vec{f}|^2 = \sqrt{f_x^2 + f_y^2}$$

$$\frac{\vec{f}}{|\vec{f}|} = \text{unitario en la dirección de máxima variación}$$

ej: $S(x,y)$ con ambas frec.

· Sistemas 2D continuos

ya sea lineal, no lineal, con memoria, ...

$$g(x,y) = T\{s(x,y)\}$$

- Desplazamiento: $T\{s(x,y)\} = s(x-x_0, y-y_0)$
- Reflexión sobre eje x: $T\{s(x,y)\} = s(-x, y)$
- Reflexión sobre eje x e y: $T\{s(x,y)\} = s(-x,-y)$
- Promedio deslizante:

$$T\{s(x,y)\} = \frac{1}{\Delta x \Delta y} \int_{x-\frac{\Delta x}{2}}^{x+\frac{\Delta x}{2}} \int_{y-\frac{\Delta y}{2}}^{y+\frac{\Delta y}{2}} s(x', y')\, dx'\, dy'$$

cada punto hace la media de un cuadradito a su alrededor

- Giro

$T\{s(x,y)\} = s(x\cos\theta + y\operatorname{sen}\theta,\ y\cos\theta - x\operatorname{sen}\theta)$

· Tipos de sistemas 2D

· puntuales: cada punto de la salida viene dado por un punto de la entrada
  - transformaciones geométricas
  - transformaciones de intensidad
· sin memoria: cada punto de la salida depende sólo del mismo punto en la entrada
  - transformaciones de intensidad
· causales: cada punto de la salida sólo depende de ptos. anteriores (poco sentido en imágenes)
· lineales: $T\{a\cdot s + b\cdot g\} = a\cdot T\{s\} + b\cdot T\{g\}$
· invariantes: $T\{s(x,y)\} = g(x,y) \Rightarrow T\{s(x-x_0, y-y_0)\} = g(x-x_0, y-y_0)$

} sistemas LSI

· estables: BIBO (Bounded Input Bounded Output)
si la entrada es acotada, la salida será acotada

Sistemas LSI (linear and spatial invariant)

- Fáciles de caracterizar: $h(x,y) \xrightarrow{TF2D} H(f_{cx}, f_{cy})$
- Cálculo de la salida: $g(x,y) = s(x,y) * h(x,y)$
  $G(f_x, f_y) = S(f_x, f_y) \cdot H(f_{cx}, f_{cy})$
- Los fasores son autofunciones: $e^{j\underline{\omega}\underline{x}} \rightarrow [LSI] \rightarrow A \cdot e^{j\underline{\omega}\underline{x}}$ (amplitud y fase)

nota: $e^{j\underline{\omega}\underline{x}} = e^{j(2\pi f_{cx} x + 2\pi f_{cy} y)}$ mantiene la frec.

Transformada de Fourier 2D

TF2D directa:
Descompone la señal en forma de fasores

$$S(\Omega_x, \Omega_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} s(x,y) \cdot e^{-j(\Omega_x x + \Omega_y y)}\, dx\, dy$$

$$S(\underline{\Omega}) = TF2D(s(\underline{x}))$$

TF2D inversa:
Recompone la señal a partir de sus fasores

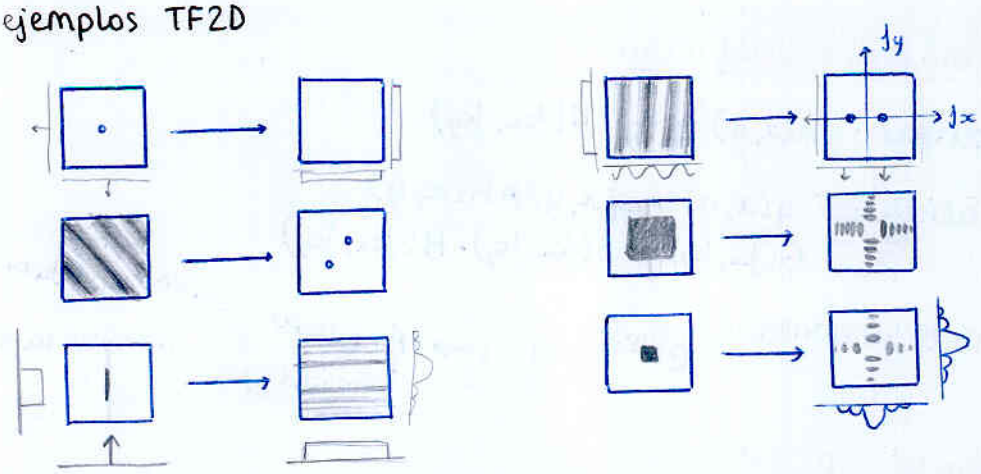
$$s(x,y) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} S(\Omega_x, \Omega_y)\, e^{j(\Omega_x x + \Omega_y y)}\, d\Omega_x\, d\Omega_y$$

	Espacio	Frecuencia
Variables	x, y	f_x, f_y $\quad \vec{f} = (f_x, f_y)$
Unidades	cm, cm	cm^{-1}, cm^{-1}
Interpretación:	variaciones rápidas - detalles	Altas frecuencias → En imagen hay pocas pero son importantes (formas)
	variación lenta - zona uniforme	Bajas frecuencias → Lo habitual en una imagen

Propiedades de la TF2D

- Linealidad: $a\, s_1(\underline{x}) + b\, s_2(\underline{x}) \xrightarrow{TF2D} a\, S_1(\underline{\Omega}) + b\, S_2(\underline{\Omega})$
- Desplazamientos espaciales: $s(\underline{x} - \underline{x_0}) \xrightarrow{TF2D} S(\underline{\Omega}) \cdot e^{j\underline{\Omega}^t \underline{x_0}}$
- Separabilidad: $s(x,y) = a(x) \cdot b(y) \xrightarrow{TF2D} S(\underline{\Omega}) = A(\Omega_x) \cdot B(\Omega_y)$ (i)
- Rotación: si rotamos ángulo θ una imagen su TF rota el mismo ángulo
- Convolución (filtrado): $s(\underline{x}) * h(\underline{x}) \xrightarrow{TF2D} S(\underline{\Omega}) \cdot H(\underline{\Omega})$
- Producto (enventanar): $s(\underline{x}) \cdot w(\underline{x}) \xrightarrow{TF2D} S(\underline{\Omega}) * W(\underline{\Omega})$
- Simetría: si $s(\underline{x})$ es real → $S(\underline{\Omega})$ es hermítica (conjugada e invertida)
  $S(\underline{\Omega}) = S^*(-\underline{\Omega})$

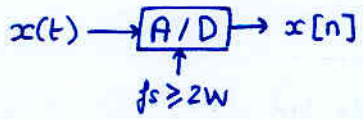
ejemplos TF2D

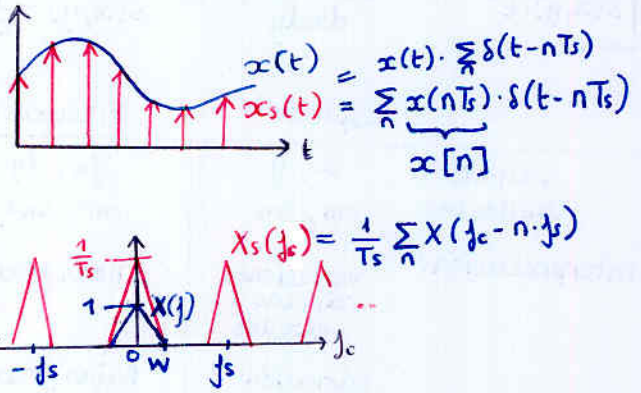


4.2 El muestreo de señales 2D

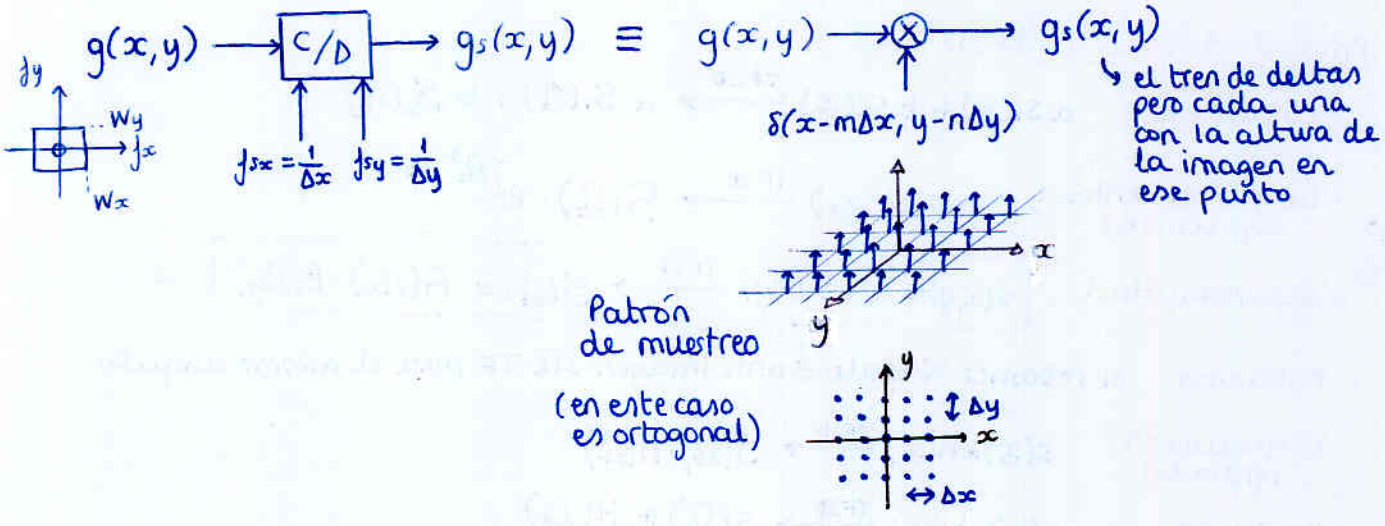
- **Muestreo Ortogonal**

Recuerda: muestreo 1D





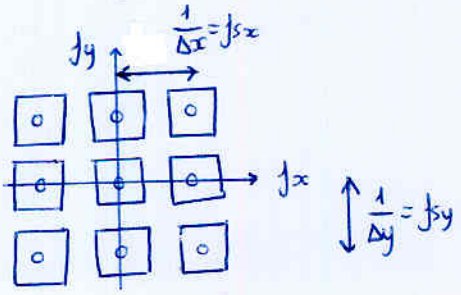
en 2D:



$$g_s(x,y) = g(x,y)\cdot \sum_m \sum_n \delta(x-m\Delta x, y-n\Delta y)$$
$$= \sum_m \sum_n \underbrace{g(m\Delta x, n\Delta y)}_{g[m,n]}\, \delta(x-m\Delta x, y-n\Delta y)$$

TF2D ⇓

$$G_s(f_x,f_y) = \frac{1}{\Delta x\,\Delta y}\cdot \sum_m \sum_n G\left(f_x - m\cdot\frac{1}{\Delta x}, f_y - n\cdot\frac{1}{\Delta y}\right)$$



Secuencia periódica

1D $\longrightarrow$ $x(t) = x(t+nT)$ $\quad T$: periodo

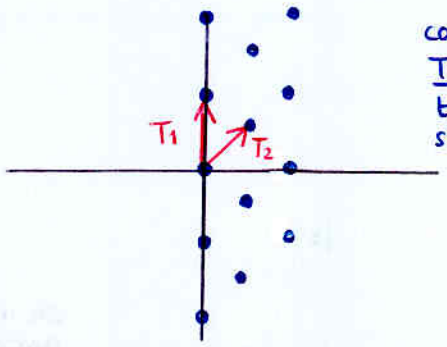
2D $\longrightarrow$ $s(x,y) = s(\underline{x}) = s(\underline{x} + m\cdot\underline{T_1} + n\cdot\underline{T_2}) = s(\underline{x} + \underline{\underline{P}}\cdot\underline{n})$

$\underline{x} = \begin{pmatrix} x \\ y \end{pmatrix}$ $\quad \underline{T_1} = \begin{pmatrix} T_{1x} \\ T_{1y} \end{pmatrix}$ $\quad \underline{T_2} = \begin{pmatrix} T_{2x} \\ T_{2y} \end{pmatrix}$

siendo $\underline{\underline{P}} = \begin{pmatrix} T_{1x} & T_{2x} \\ T_{1y} & T_{2y} \end{pmatrix}$ Matriz de periodicidad

$\underline{n} = \begin{pmatrix} m \\ n \end{pmatrix}$

ejemplo: $\underline{T_1} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$ $\quad \underline{T_2} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$



como vemos, $\underline{T_1}$ y $\underline{T_2}$ no tienen porque ser ortogonales

Muestreo periódico (no ortogonal necesariamente)

$$g(x,y) \longrightarrow (\times) \longrightarrow g_s(x,y) = g_s(\underline{x}) = g(x,y)\cdot\sum\sum \delta(\underline{x} - m\underline{T_1} - n\underline{T_2}) = g(\underline{x})\cdot\sum_{\underline{n}=-\infty}^{\infty} \delta(\underline{x} - \underline{\underline{P}}\,\underline{n})$$

$$\sum_m \sum_n \delta(\underline{x} - m\underline{T_1} - n\underline{T_2})$$

Caso particular: muestreo ortogonal

$$\underline{\underline{P}} = \begin{pmatrix} \Delta x & 0 \\ 0 & \Delta y \end{pmatrix}$$

$$g_s(\underline{x}) = g(\underline{x})\cdot\sum_{\underline{n}} \delta(\underline{x} - \underline{\underline{P}}\,\underline{n})$$

$\downarrow$ TF2D

$$G_s(f_x, f_y) = \frac{1}{\det(\underline{\underline{P}})} \sum_{\underline{n}=-\infty}^{\infty} G(\underline{f} - \underline{\underline{U}}\cdot\underline{n})$$

$\underline{\underline{U}} = (\underline{\underline{P}}^{-1})^t$

analogía 1D

$X_s(f_c) = \frac{1}{T_s}\sum_n X(f_c - \frac{n}{T_s})$

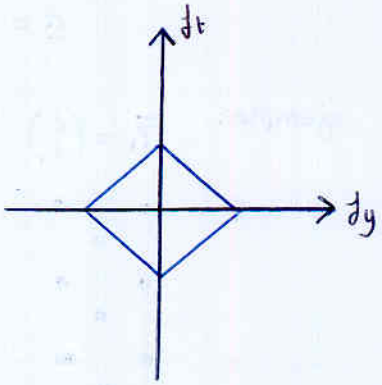
Densidad de muestreo $\propto \frac{1}{\det(\underline{\underline{P}})}$

El muestreo periódico es más eficiente (no dejar huecos espectrales entre las repeticiones) que el ortogonal en casos en que el espectro de la señal a muestrear tenga ciertas características.

Ejemplo de eficiencia del muestreo no ortogonal

Muestreo 2D espacio temporal (ej: muestrear señal de TV)

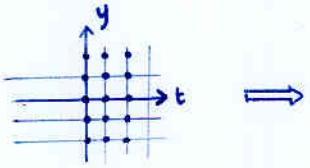
Espectro espaciotemporal tiene la respuesta de nuestro ojo (forma de rombo: i.e. para que mi ojo vea f_y alta se necesita una f_t pequeña y viceversa)



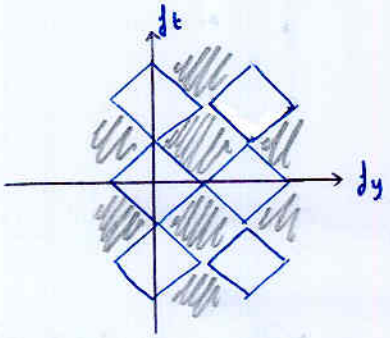
mucho movimiento $f_t \uparrow\uparrow \longrightarrow f_y \downarrow$ sólo se ve bien si tiene poco detalle

mucho detalle $f_y \uparrow\uparrow \longrightarrow f_t \downarrow\downarrow$ para ver mucho detalle necesitas movimiento lento

muestreo ortogonal: muestreo cada línea en cada instante de tiempo

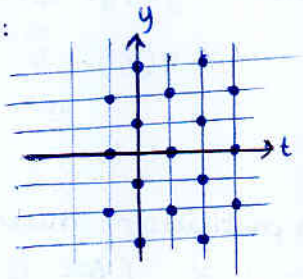


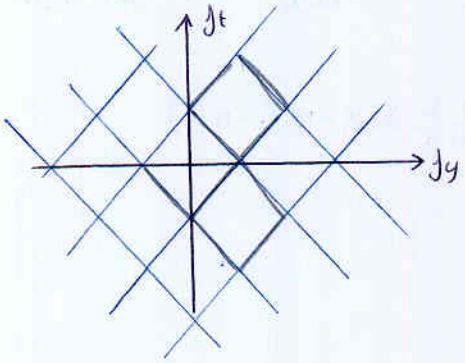
$\Longrightarrow$



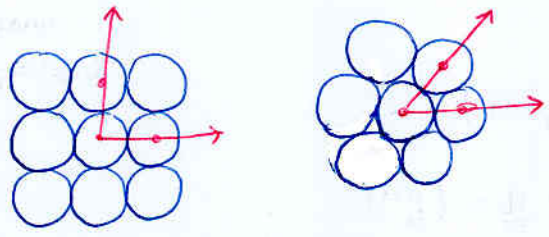
se malgasta ancho de banda

solución que se usó: entrelazado (líneas pares e impares en diferentes instantes)



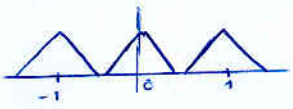


Ejemplo: espectro circular



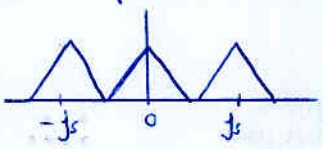
4.2.3 Recuperación de la señal a partir de sus muestras

$g[n] = g[\underline{P}\,\underline{n}]$



1. Poner las muestras en su posición real

$\underline{P}\,\underline{n}$ (en frec es $\times f_s$)



ej: imprimir puntitos de una imagen

2. Filtro paso bajo para quedarse con la repetición central

ej: el ojo humano viendo foto compuesta por puntitos

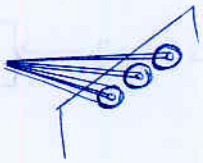
4.2.4 Aliasing en señales multidimensionales

$f_{sx} \geq 2W_x$
$f_{sy} \geq 2W_y$
$f_{st} \geq 2W_t$

Efectos de aliasing:
- espacial: efecto de escalera en bordes, efectos raros (ejemplo: persiana en la foto del cura)
- temporal: efecto estroboscópico

Solución: Filtro antialiasing paso bajo antes de muestrear

Espacial:
- ancho del haz de electrones para barrer en TV
- tamaño del CCD en las cámaras digitales



Temporal:
- tiempo de obturación de la cámara

4.3 Señales y sistemas 2D discretos

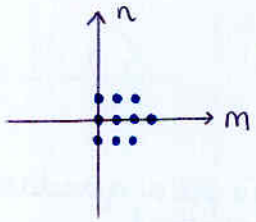
Señales discretas

$s(x,y) \rightarrow$ [muestreo] $\rightarrow s[m,n] = s(\underline{\underline{P}}\,\underline{n})$

↓

MATRIZ

siempre en rejilla ortogonal, aunque el muestreo no haya sido ortogonal

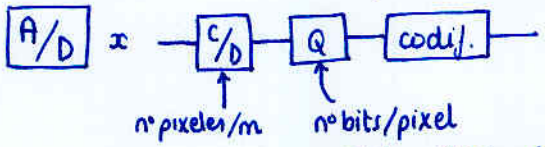


Ejemplos:

$$\delta[m,n] = \begin{cases} 1 & (m,n)=(0,0) \\ 0 & \text{resto} \end{cases}$$

idem continuas

Para obtener una señal digital aún falta cuantificar



n = 8 bits → 256 niveles (al ojo le bastan con 64)

Sistemas discretos

$s[m,n] \rightarrow [T\{\cdot\}] \rightarrow g[m,n]$

idem continuos

- sin memoria
- puntuales
- causales
- estables
- lineales
- invariantes

} → LSI

LSI

respuesta al impulso:

$\delta[m,n] \rightarrow [LSI] \rightarrow h[m,n]$

impulso

↓ TF2D

$H(e^{j\omega_x}, e^{j\omega_y})$

convol. en t

$$g[m,n] = s[m,n] * h[m,n] = \sum_{n} \sum_{m=-\infty}^{\infty} s[m',n'] \cdot h[m-m', n-n']$$

producto en f.

$$G(e^{j\omega_x}, e^{j\omega_y}) = S(e^{j\omega_x}, e^{j\omega_y}) \cdot H(e^{j\omega_x}, e^{j\omega_y})$$

Filtros fase lineal

fase = formas en imagen

Fase lineal: no alteran la fase

↳ desplazan la señal

Fase = 0: ni te la desplazan

Transformada Fourier 2D

de una señal discreta

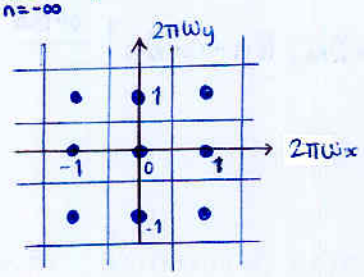
$x(t) \xrightarrow{TF} X(f_c)$

$x[n] \longrightarrow X(e^{j\omega})$

$s[m,n] \xrightarrow{TF2D} S(e^{j\omega_x}, e^{j\omega_y})$

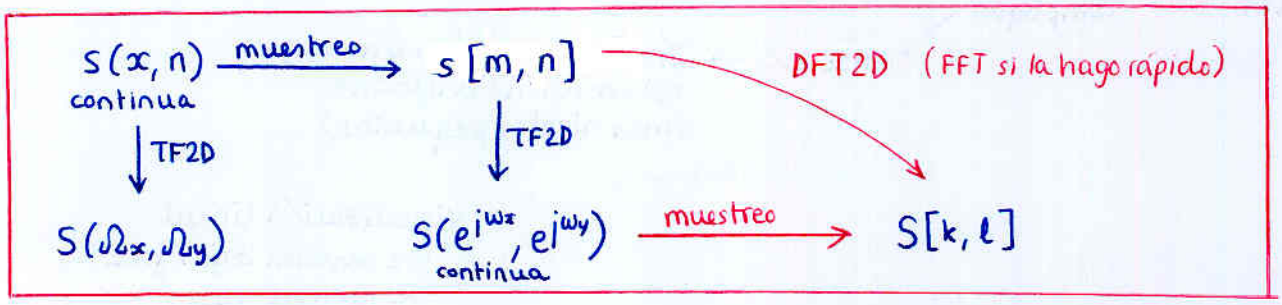
$$S(e^{j\omega_x}, e^{j\omega_y}) = \sum_{m}^{\infty}\sum_{n=-\infty}^{\infty} s[m,n]\, e^{-j(\omega_x m + \omega_y n)}$$

es { periódica, continua



Propiedades: las mismas que la continua

¡Pero el espectro sigue siendo continuo! ¿cómo lo visualizo en un ordenador?

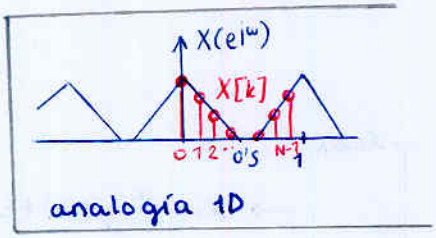


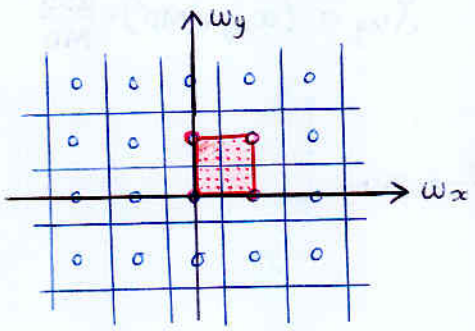
4.4 La DFT2D y la FFT2D

DFT2D

$$S[k,l] = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} s[m,n]\, e^{-j2\pi\left(\frac{km}{M} + \frac{ln}{N}\right)} \qquad \begin{matrix} 0 \le k \le M-1 \\ 0 \le l \le N-1 \end{matrix}$$

$$s[m,n] = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} S[k,l]\, e^{j2\pi\left(\frac{mk}{M} + \frac{nl}{N}\right)} \qquad \begin{matrix} 0 \le m \le M-1 \\ 0 \le n \le N-1 \end{matrix}$$





Diferencias DFT2D – TF2D

DFT2D { discreta / no periódica TF2D { continua / periódica

Propiedades: las mismas, pero en la DFT los desplazamientos e inversiones son cíclicas.

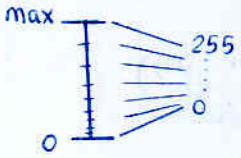
ej: $s[((m-m_0))_M, ((n-n_0))_N] \xrightarrow{DFT2D} S[k,l]e^{j2\pi\left(\frac{km_0}{M}+\frac{ln_0}{N}\right)}$

un desplazamiento en la imagen sólo altera la fase

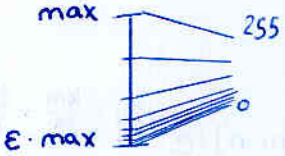
Gran margen dinámico: visualizaciones

$s[m,n]$ reales → $S[k,l]$ complejas

- FASE → información formas de la imagen
- MÓDULO → Gran margen dinámico y además no uniforme (más niveles pequeños)



Visualización lineal
- se ven bien bajas frec.
- se ven mal altas frecs (valores pequeños)



Visualización en dB's

$$\varepsilon = 10^{-MD/20}$$

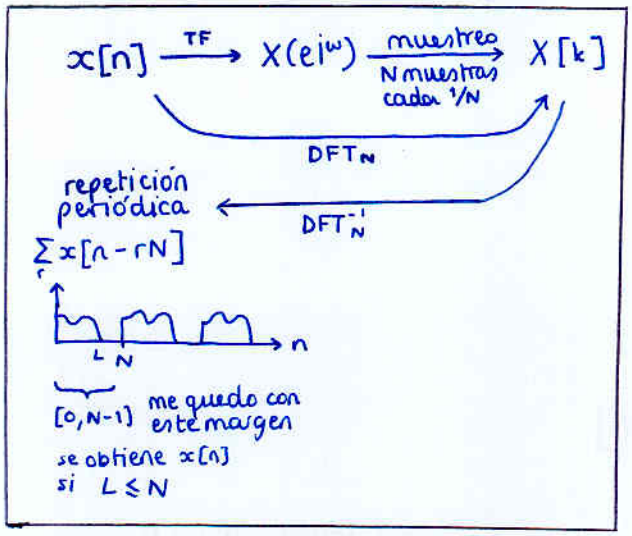
$$x_{log} = 20\log\left(\frac{x}{x_{max}}(1-\varepsilon)+\varepsilon\right)$$

$$X_{log} = (x_{log} + MD)\cdot\frac{255}{MD}$$

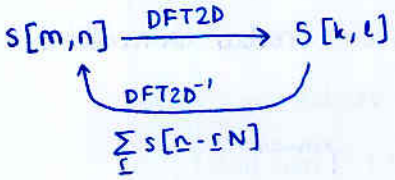
$$x \in [0, x_{max}] \longrightarrow x_1 = \left[\frac{x}{x_{max}}(1-\varepsilon)+\varepsilon\right] \in [\varepsilon, 1] \xrightarrow{20\log} x_{log} \in [-MD, 0] \longrightarrow X_{log} \in [0, 255]$$

($\varepsilon = 10^{-\frac{MD}{20}}$)

Recuerda: DFT^{-1} en 1D

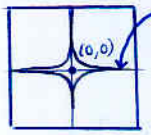


En 2D



DFT2D equivale a TF2D + muestreo de la imagen repetida periódicamente

Aparece un problema en la DFT:

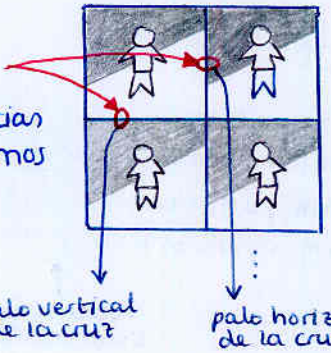


Aparece una cruz

razones:

Por usar DFT en lugar de TF + muestreo (ya que es como hacer la transformada sobre la imagen repetida periódicamente)

Aparecen frecuencias altas (horizontales y verticales) cuando en la imagen hay diferencias bruscas entre los extremos opuestos de la imagen



Pueden haber imágenes que no tengan uno o ambos palos de la cruz

ej

solución: · Enventanar la imagen (que en los extremos valga cero) se le quita la cruz

Tipos de enventanado

- Deformando la media

$$s[m,n] \cdot w[m,n] \xrightarrow{DFT2D} S[k,l] \textcircled{N} W[k,l]$$

- Sin deformar la media

$$(s[m,n] - \text{media}) \cdot w[m,n] + \text{media} \xrightarrow{DFT2D} \text{idem}$$

pero la frecuencia (0,0) es la media original

Procedimiento final de cálculo DFT2D

1. $s[m,n]$ − media (restar la media)
2. Multiplicar por ventana
3. Relleno con ceros (muestreo más fino)

3.5 opcional: sumar la media

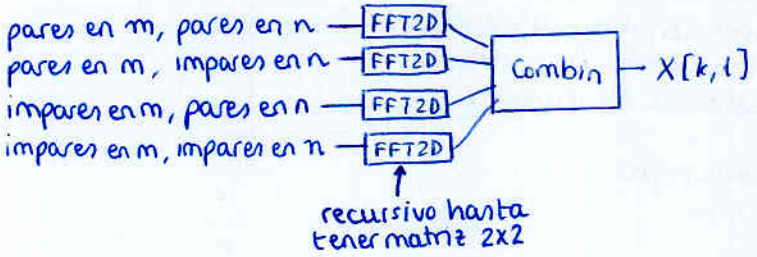
4. DFT2D

Cálculo eficiente de la DFT2D: La FFT2D

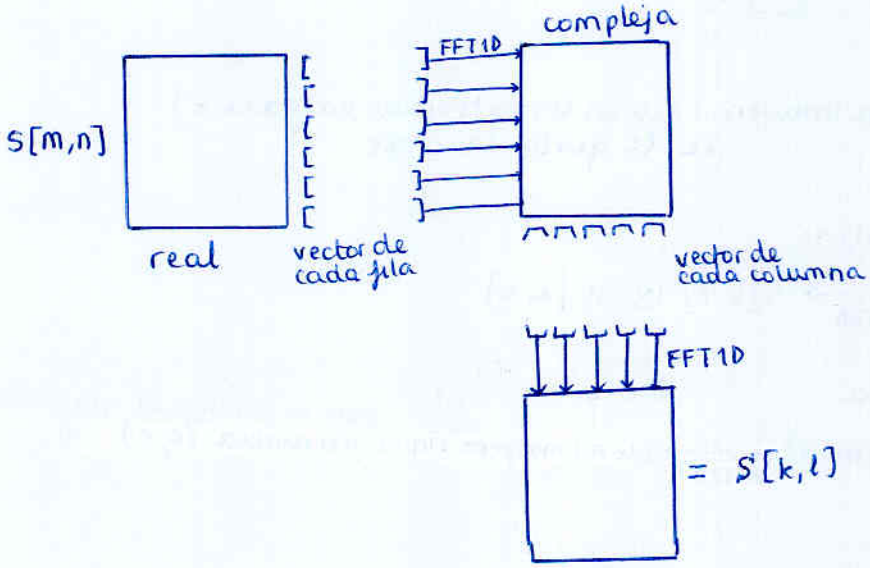
Método 1D

Dividir en mitades $x[n]$

Método 2D



Método 2D más usado



Aplicaciones FFT2D

- Filtrado en frecuencia
- Análisis de texturas
- Determinación de orientaciones preferentes
- Determinación de enfoque óptimo
- Eliminación de interferencias periódicas
- Teorema de las proyecciones
- Determinación de desplazamientos
- Determinación de degradaciones

Aplicaciones de la FFT2D

- Filtrado en el dominio de la frecuencia.
- Análisis de texturas.
- Determinación de orientaciones preferentes.
- Determinación del enfoque óptimo.
- Eliminación de ruido periódico.
- Teorema de las proyecciones.
- Estima de desplazamientos.
- Estima de degradaciones.

Filtrado en el dominio de la frecuencia

- Filtrado de una señal: Calcular la convolución con $h[\underline{n}]$

$$g[\underline{n}] = s[\underline{n}] * h[\underline{n}]$$

- La convolución equivale a multiplicar en la frecuencia.

$$G(e^{j\underline{\omega}}) = S(e^{j\underline{\omega}})\, H(e^{j\underline{\omega}})$$

- En la práctica se multiplican DFT2D y se toma la IDFT2D. Esto equivale a:
  - Convolución circular.
    ||
    Convolución lineal de la señal repetida periódicamente.
- Necesidad de enventanar.
- No usar $H[\underline{k}]$ con discontinuidades $\Rightarrow$ transitorios largos.

Análisis de texturas

- Textura: Patrón cuasi-periódico.
- Señal periódica: Espectro de picos.
- La posición, y amplitud de los picos caracterizan la textura.

Determinación de orientaciones preferentes

- Señal aleatoria con orientación preferente.
- Distinto de textura
- Hay que enventanar.
- El espectro es más ancho en la dirección ortogonal a la orientación.

$$f_x = \rho\cos\theta \qquad f_y = \rho\sin\theta$$

$$g(\theta) = \int_0^\infty G(\rho,\theta)\,d\rho \qquad 0 \le \theta \le \pi$$

- El pico de $g(\theta)$ da la dirección ortogonal a la de alineamiento preferente.

Determinación del enfoque óptimo

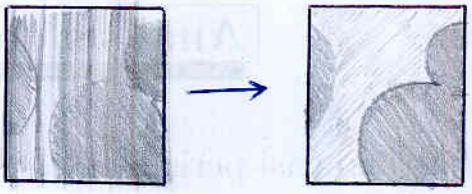
- Señal desenfocada: punto se convierte en círculo.

$$g(x,y) = s(x,y) * \mathcal{O}\left(\frac{\sqrt{x^2+y^2}}{2r}\right)$$

- Desenfoque: filtro paso-bajo.
- Puedo:
  1. Medir el contenido en altas frecuencias para una posición del enfoque.
  2. Mover el enfoque.
  3. Repetir la medida del contenido en altas frecuencias.
  4. Repetir hasta que se maximice el contenido en altas frecuencias.

Eliminación de interferencias periódicas

- Interferencia periódica ⇒ Espectro picos.



- Procedimiento:
  1. Enventanar
  2. Calcular DFT2D.
  3. Localizar (manualmente o automáticamente) los picos (puntos brillantes) de la interferencia mirando el módulo de la DFT2D.
  4. Anular las correspondientes componentes de la DFT2D.
  5. Calcular la IDFT2D.

Teorema de las proyecciones

- Procedimiento de reconstrucción de imágenes con proyecciones.
- Usado en Tomografía Axial Computerizada (TAC).
- Proyección sobre el eje x de una imagen $s(x, y)$:

$$g(x) = \int_{-\infty}^{\infty} s(x, y)\, dy$$

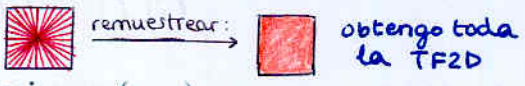
- Calculo la Transformada de Fourier de $g(x)$:

$$G(f_x) = \int_{-\infty}^{\infty} g(x) \exp(-2\pi j f_x x)\, dx$$

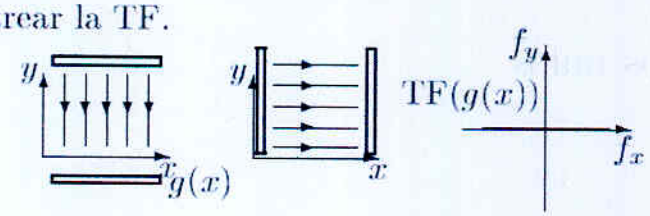
$$G(f_x) = S(f_x, 0)$$

- La TF1D de la proyección coincide con una línea de la TF2D ($f_y = 0$)
- Si ahora considero la proyección sobre el eje y, obtengo otra línea de la TF2D ($f_x = 0$). $G_2(f_y) = S(0, f_y)$

- Por la propiedad de la rotación de la TF puedo obtener más líneas de la TF2D, y rellenar toda la TF2D.



- Calculando la ITF2D puedo determinar $s(x, y)$
- En la práctica se usa FFT1D, y la IFFT2D, por ser señales de extensión finita
- Es necesario remuestrear la TF.



Determinación de desplazamientos

- Muchas veces: interesa estimar desplazamiento de bloque de imágenes.
- Ejemplos
  1. Codificación de TV.
  2. Medida de velocidades utilizando dos fotogramas.
- Los desplazamientos solo afectan a la fase transfromada de Fourier.
- Método

$$\text{IFFT2}\left\{\exp(j(\angle S_1[\underline{k}] - \angle S_2[\underline{k}]))\right\}$$

Determinación de degradaciones

- El paso previo a restaurar una imagen es saber como se degradó.
- Algunas degradaciones se modelan bien como LSI's.
  1. Imágenes movidas.
  2. Imágenes desenfocadas.
- Es necesario conocer el grado de desenfoque o movimiento.
- Las respuestas en frecuencia de ambas presentan nulos:
  1. En forma de anillos en desenfoque.
  2. En forma de líneas en movimiento.

$$G(\underline{f}) = S(\underline{f})\, H(\underline{f})$$

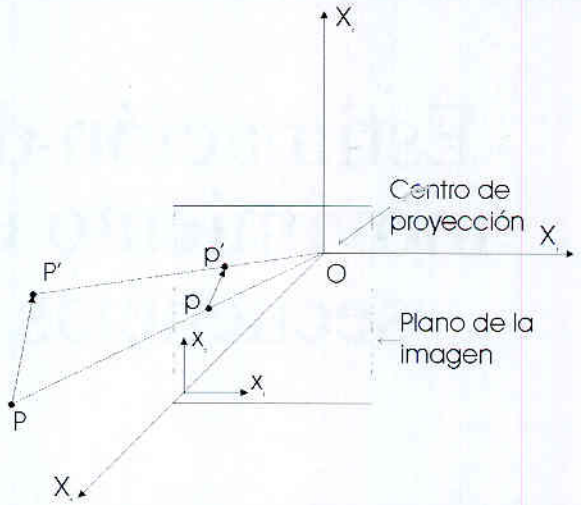
- Los nulos de $H(\underline{f})$ aparecen en la imagen degradada $G(\underline{f})$.
- Los nulos caracterizan totalmente la respuesta en frecuencia.
- Localizando los nulos

Estimación de movimiento en secuencias

Índice

- Introducción
- Métodos de bloques
- Métodos de fase
- Métodos de flujo óptico
- Aplicaciones

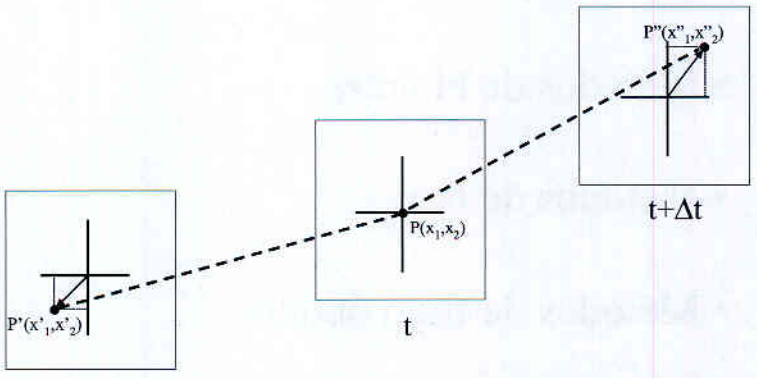
Vector desplazamiento 2D



3 Introducción

El único movimiento que yo podré estimar es únicamente la proyección sobre la imagen (movimiento aparente o vector desplazamiento)

Estimación de movimiento 2D



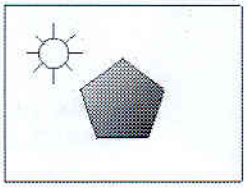
4 Introducción

Problemas:

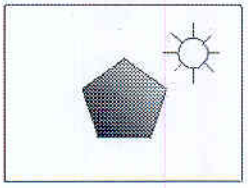
Desplaz. 2D ≠ Desplaz. aparente

Cambios de iluminación

Fotograma k



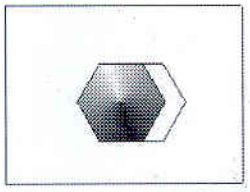
Fotograma k+1



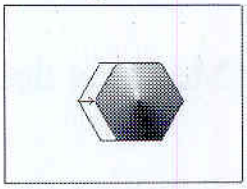
Problemas en la estimación

Oclusión

Fotograma k



Fotograma k+1



objetos tapados por otros

Problemas en la estimación

Falta de gradiente espacial

es necesario que hayan diferencias de intensidad en el objeto para que yo pueda detectar el movimiento

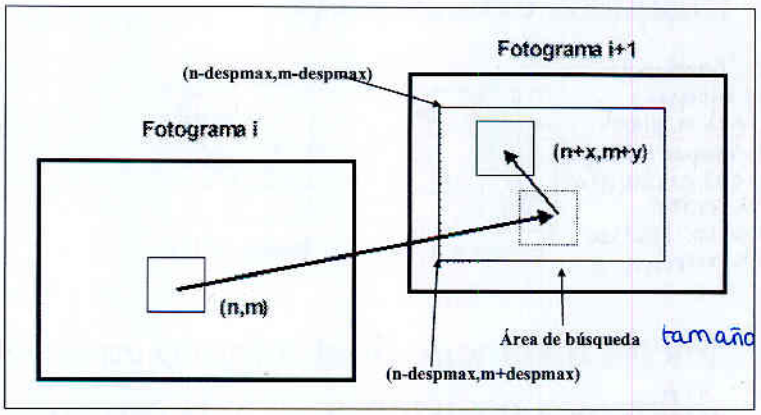
ej: disco girando

puedo no darme cuenta del giro

Índice

- Introducción
- Métodos de bloques
- Métodos de fase
- Métodos de flujo óptico
- Aplicaciones

Métodos de bloques



Comparar el bloque con bloques del mismo tamaño en la siguiente imagen

tamaño $(\text{desplaz}_{\max}+1)\times(\text{desplaz}_{\max}+1)$

¿Cómo mido lo parecidos que son los bloques?

M.de bloques: Medidas parecido

- Máxima correlación cruzada

$$R(d_1,d_2)=\frac{\sum_{m,n\in B} s(m,n,k)s(m+d_1,n+d_2,k+1)}{\sqrt{\sum_{m,n\in B} s^2(m+d_1,n+d_2,k+1)}}$$

- Mínimo error cuadrático medio (MSE)

$$MSE(d_1,d_2)=\frac{1}{MN}\sum_{m,n\in B}\left[s(m,n,k)-s(m+d_1,n+d_2,k+1)\right]^2$$

- Mínima diferencia absoluta media (MAD)

$$MAD(d_1,d_2)=\frac{1}{MN}\sum_{m,n\in B}\left|s(m,n,k)-s(m+d_1,n+d_2,k+1)\right|$$

Métodos de bloques

Búsqueda exhaustiva (I)

Ir "barriendo" los bloques pixel a pixel (no bloque a bloque) y calculando el error (hasta hallar el mínimo)

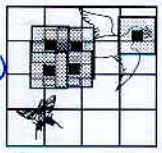


Imagen n: I_n

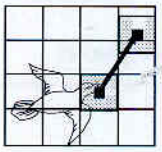


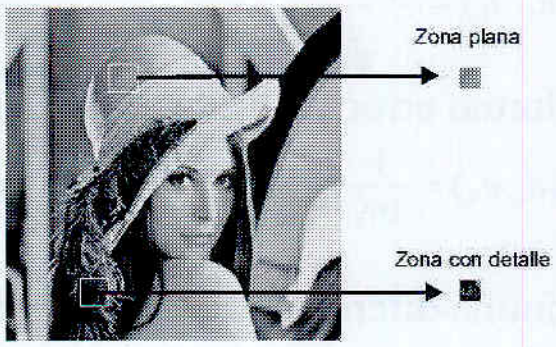
Imagen n+1: I_{n+1}

Factor distancia: Si el mínimo error está en diferentes posiciones, el más cercano ^ tomamos

Problema: método óptimo muy lento

Métodos de bloques

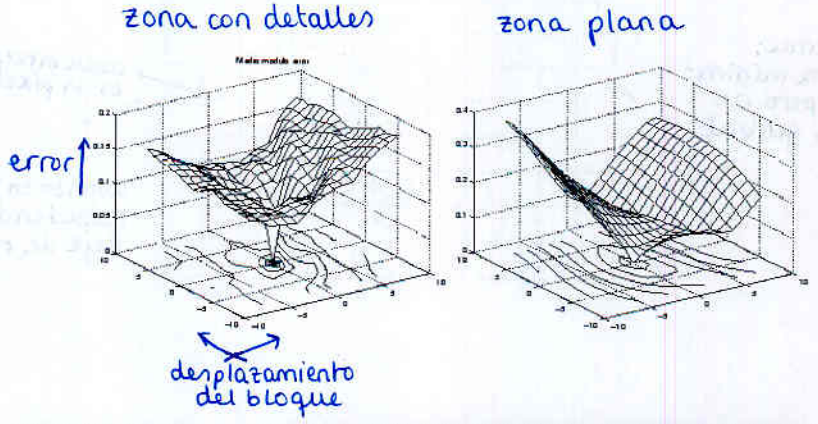
Búsqueda exhaustiva (II)



Métodos de bloques

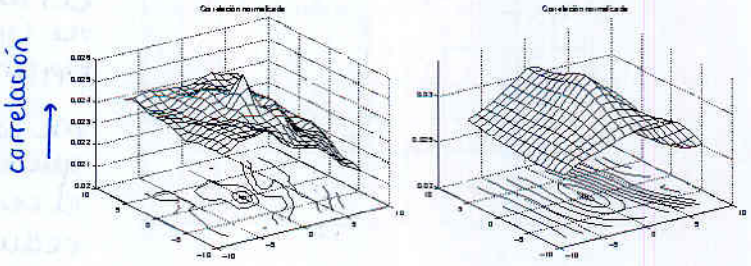
Búsqueda exhaustiva (II)

error cuadrático medio o valor absoluto de la diferencia



Métodos de bloques

Búsqueda exhaustiva (II)



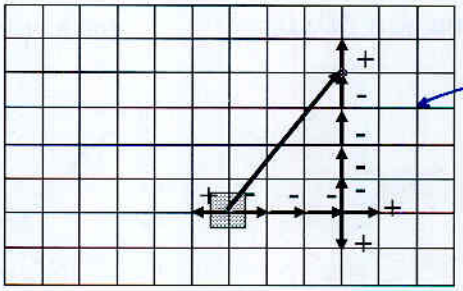
Problema de la búsqueda exhaustiva: lenta

solución: métodos subóptimos

Métodos de bloques

Búsqueda dirección conjugada

Problema: encuentra mínimo local, pero no mínimo global



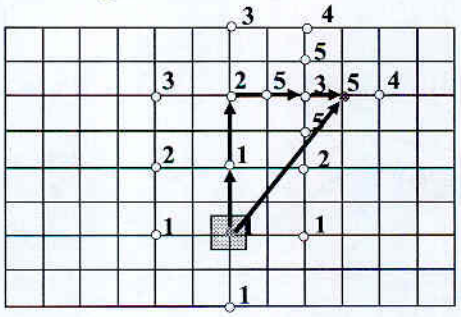
cada intersección es un pixel

avanzo en la dirección que va reduciendo el error hasta que deje de reducirse

Métodos de bloques

Búsqueda logarítmica

nota: los nºs en este diagrama indican la secuencia temporal de los acontecimientos

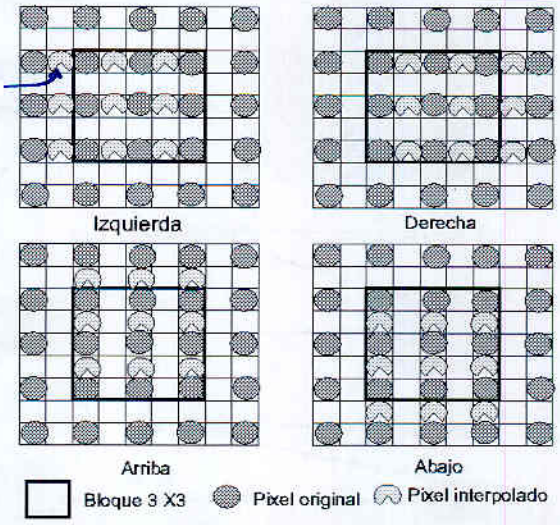


similar al anterior pero probando en todas las direcciones a la vez (y además d pixeles, típicamente 2, en lugar de 1)

me voy desplazando hasta que el menor me da en el centro, y entonces reduzco d a la mitad y repito el proceso

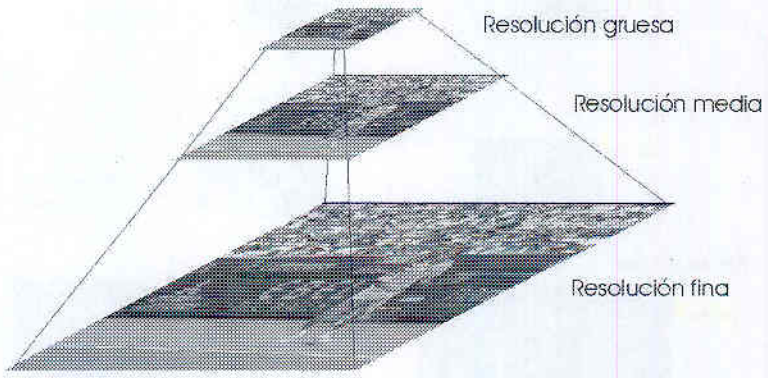
M.de bloques: Refinado 1/2 píxel

bloque (i,j) + bloque $(i-1,j)$ / 2

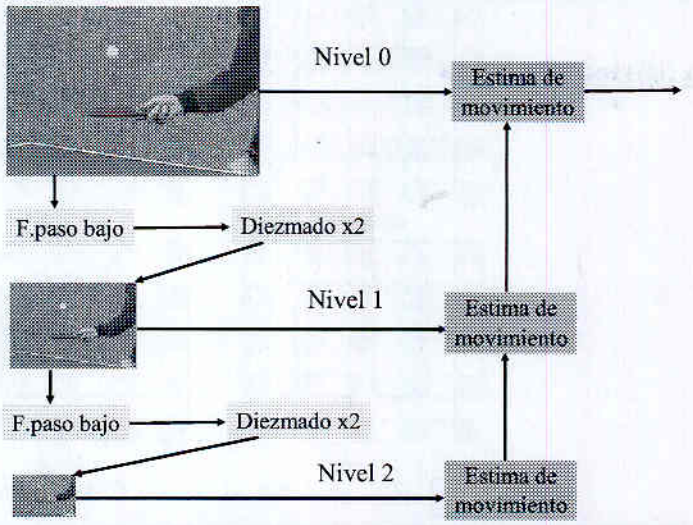


M. de bloques: Multirresolución

Búsqueda exhaustiva pero en 3 niveles



Métodos de bloques: Algoritmo

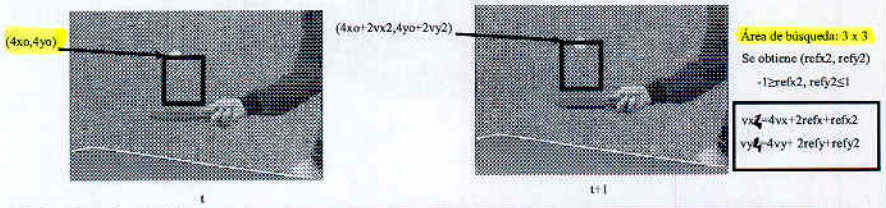


Métodos de bloques: Algoritmo

Resolución gruesa

Resolución media

Resolución fina



Búsqueda exhaustiva en la imagen pequeñita
$\vec{v} = (v_x, v_y)$
↓
$\vec{v} = (2v_x, 2v_y)$ siempre par refinamos para permitir impares

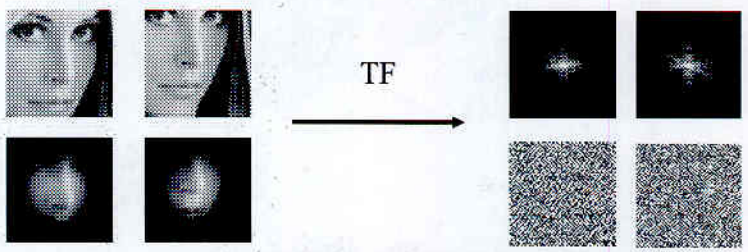
Índice

- Introducción
- Métodos de bloques
- Métodos de fase
- Métodos de flujo óptico
- Aplicaciones

Métodos basados en la fase

Propiedad :

$$TF(s(x-x_o, y-y_o)) = TF(s(x,y))\, e^{-j2\pi(f_x x_o + f_y y_o)}$$

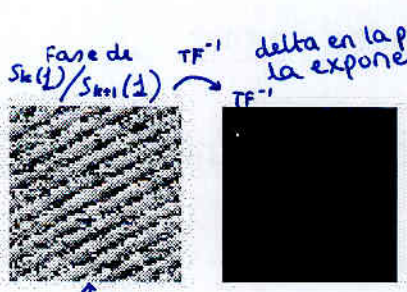


el módulo sería exactamente igual si el desplazamiento fuese cíclico

Métodos basados en la fase

$$TF^{-1}\left(\frac{S_k(f_x,f_y)}{S_{k+1}(f_x,f_y)}\right) = TF^{-1}\left(e^{i\,2\pi\,(f_x x_0 + f_y y_0)}\right)$$

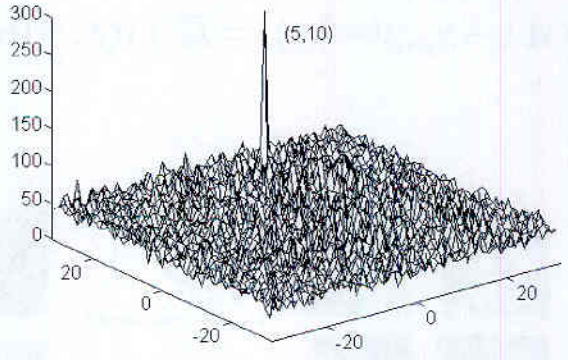
módulo 1
fase lineal
$2\pi(f_x x_0 + f_y y_0)$



23 M. fase

fase de la división
$2\pi(f_x x_0 + f_y y_0)$
es una rampa, pero queda sierra al ser la fase 2π-periódica

Métodos basados en la fase

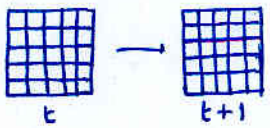


Desplazamiento máximo medible es $(N/2, N/2)$

Funciona muy bien cuando hay cambios de iluminación en la imagen, ya que no le afectan

24 M. fase

Este método funciona si se mueve TODA la imagen.
Para poder detectar movimiento de objetos, se divide en bloques y se calcula para cada bloque



Índice

- Introducción
- Métodos de bloques
- Métodos de fase
- Métodos de flujo óptico
- Aplicaciones

Métodos de flujo óptico

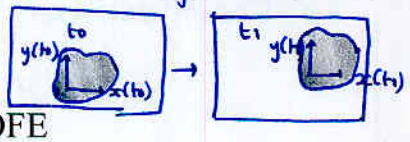
Suposición inicial

- Iluminación constante en espacio y tiempo

$$ds_c(x,y,t)/dt=0$$

$(x(t), y(t), t)$

para un objeto en movimiento



Suponemos que las variaciones de intensidad se deben sólo al movimiento (limitación)

- Ecuación de Flujo Óptico: OFE

usando la regla de la cadena

$$\frac{\partial s_c(x,y,t)}{\partial x}v_x+\frac{\partial s_c(x,y,t)}{\partial y}v_y+\frac{\partial s_c(x,y,t)}{\partial t}=0$$

$\frac{\partial x}{\partial t}$

la derivada en discreto son las diferencias (gradiente)

2 incógnitas

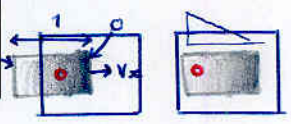
gradiente temporal (resta entre imágenes sucesivas)

ej: objeto parado

$$\frac{\partial s_c(x,y,t)}{\partial t}=0$$

ej: objeto moviéndose en dirección x

$$\frac{\partial s_c(x,y,t)}{\partial t}=-v_x\cdot\frac{\partial s_c(x,y,t)}{\partial x}$$



$$\frac{\partial s_c}{\partial t}=-v_x\cdot(-1)=v_x$$

1 ecuación y 2 incógnitas
necesito una segunda ecuación (distintos métodos)

Métodos de flujo óptico

- PROBLEMA:
  - 2 incógnitas y 1 ecuación

v_x, v_y

- SOLUCIÓN: Poner una condición
- Métodos
  - Diferenciales
  - De bloques
  - Bayesianos
  - Pel-recursivos

M. de flujo óptico (Diferenciales)

CONDICIÓN:

- Conservación en el tiempo del gradiente espacial

siendo el gradiente $\nabla S_c(x,y,t) = \begin{bmatrix} \frac{\partial S_c}{\partial x}(x,y,t) \\ \frac{\partial S_c}{\partial y}(x,y,t) \end{bmatrix}$

$$\frac{d\nabla s_c(x,y,t)}{dt} = 0$$

aplicando la regla de la cadena, se obtiene:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 s_c(x,y,t)}{\partial x^2} & \frac{\partial^2 s_c(x,y,t)}{\partial x \partial y} \\ \frac{\partial^2 s_c(x,y,t)}{\partial y \partial x} & \frac{\partial^2 s_c(x,y,t)}{\partial y^2} \end{bmatrix}^{-1} \begin{bmatrix} -\frac{\partial^2 s_c(x,y,t)}{\partial t \partial x} \\ -\frac{\partial^2 s_c(x,y,t)}{\partial t \partial y} \end{bmatrix}$$

Problema: la resta de píxeles es sólo un estimador de la derivada

M. de flujo óptico (Diferenciales)

PROBLEMA:

- No funciona con movimientos comunes
  - ZOOM (cambia el módulo del gradiente)
  - ROTACIÓN (cambia la dirección del gradiente)
- Difícil estimación de las derivadas de segundo orden

Métodos de flujo óptico (Bloques)

CONDICIÓN:

- Conservación del vector de movimiento en un conjunto de píxels

se aplica la ecuación del flujo óptico a todo un bloque

$$E = \sum_{(x,y)\in\beta} \left(\frac{\partial s_c(x,y,t)}{\partial x} v_x + \frac{\partial s_c(x,y,t)}{\partial y} v_y + \frac{\partial s_c(x,y,t)}{\partial t} \right)^2$$

⇩

MINIMIZACIÓN

no cada pixel se mueve sólo, sino en grupos

i.e. en lugar de

$\frac{\partial s_c}{\partial x} v_x + \frac{\partial s_c}{\partial y} v_y + \frac{\partial s_c}{\partial t} = 0$

lo hago para un sumatorio de píxeles

$\sum \left(\frac{\partial s_c}{\partial x} v_x + \frac{\partial s_c}{\partial y} v_y + \frac{\partial s_c}{\partial t} \right) = 0$

en lugar de exigir que sea cero, se minimiza la expresión

Métodos de flujo óptico (Bloques)

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \sum_{(x,y)\in\beta} \frac{\partial^2 s_c(x,y,t)}{\partial x^2} & \sum_{(x,y)\in\beta} \frac{\partial^2 s_c(x,y,t)}{\partial x \partial y} \\ \sum_{(x,y)\in\beta} \frac{\partial^2 s_c(x,y,t)}{\partial x \partial y} & \sum_{(x,y)\in\beta} \frac{\partial^2 s_c(x,y,t)}{\partial y^2} \end{bmatrix}^{-1} \begin{bmatrix} -\sum_{(x,y)\in\beta} \frac{\partial^2 s_c(x,y,t)}{\partial t \partial x} \\ -\sum_{(x,y)\in\beta} \frac{\partial^2 s_c(x,y,t)}{\partial t \partial y} \end{bmatrix}$$

PROBLEMA:

Cálculo de las derivadas parciales

Índice

- Introducción
- Métodos de bloques
- Métodos de fase
- Métodos de flujo óptico
- Aplicaciones

Aplicaciones

- Compresión
  - MPEG, H261, H263,..
- Eliminación de vibración en secuencias
- Detección de fotogramas perdidos
- Interpolación de fotogramas perdidos

A 1.

Aplicaciones: Eliminación de vibración

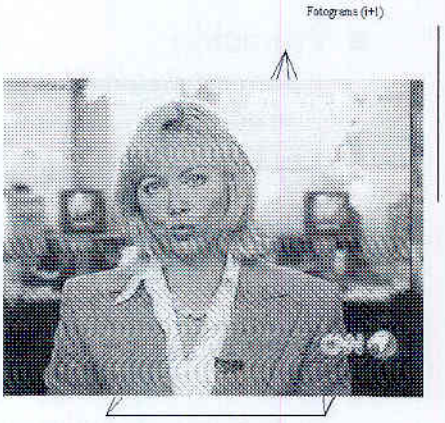
- Vibración:
  Movimiento global de la escena de naturaleza aleatoria.
  - BASE: Detección movimiento global.
  - Movimiento real
    Variación de la posición de los objetos y fondo de la escena.
  - Movimiento captado
    Variación de la intensidad de los píxeles de la imagen.

Aplicaciones: Eliminación de vibración

35 Aplicaciones

Aplicaciones: Eliminación de vibración

- Multi-resolución.
- Block-matching.
- Votación.



a pesar de tener un vector de movimiento para cada bloque, se hace la media (o la moda)

36 Aplicaciones

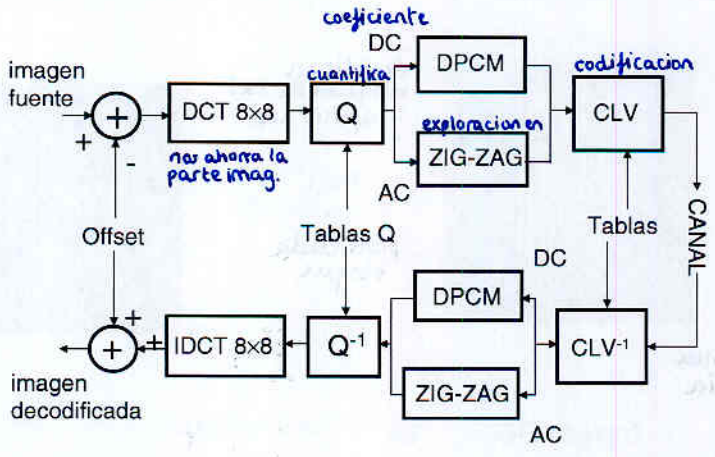
A2

Aplicaciones: Compresión vídeo

1 ■ JPEG

2 ■ MPEG

1 JPEG Secuencial



codigo Longitud Variable

Fórmula de la DCT

Secuencial

$$F[u,v]=\frac{1}{4}C(u)C(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\cos\left(\frac{\pi(2x+1)u}{2N}\right)\cos\left(\frac{\pi(2y+1)v}{2N}\right)$$

El resultado es real

$$C(u),C(v)=\frac{1}{\sqrt{2}} \quad para \quad u,v=0$$

$$C(u),C(v)=1 \quad para \quad resto \ de \ casos$$

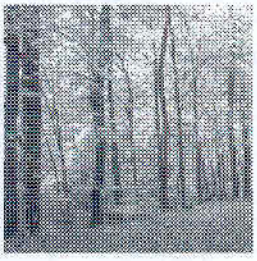
componente continua

x → / y ↓			
87	89	101	106
85	91	101	105
86	92	96	105
92	88	102	101

DCT

f_x → / f_y ↓			
381.75	-27.2969	1.75	4.006
0.4059	-4.5303	-0.6765	-1.6339
1.25	1.9695	1.25	5.4080
-0.9799	-0.1339	1.6332	-3.4697

Secuencial



dividir en bloques de 8x8 y aplicar la DCT para cada bloque

cada componente de continua de cada bloque es la media

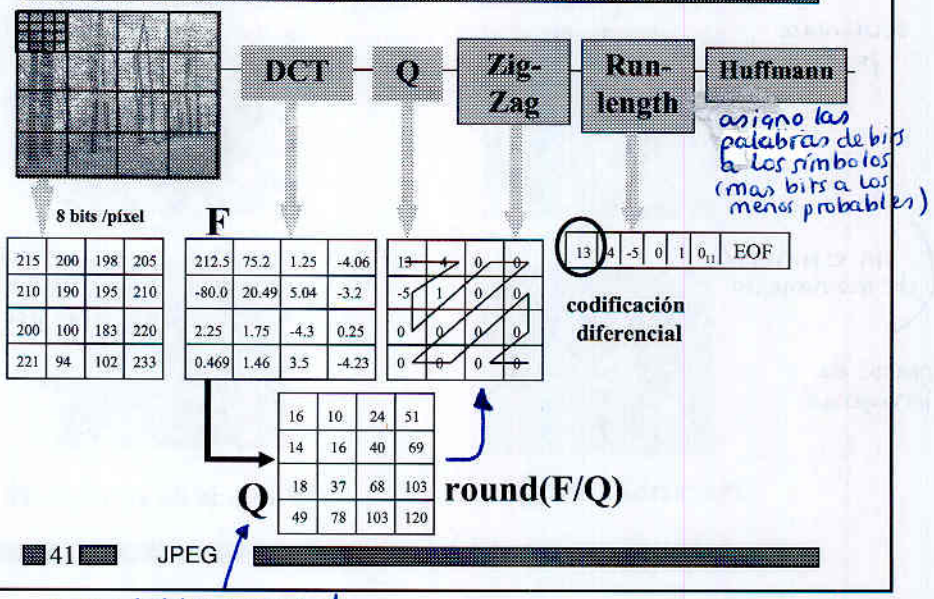
la imagen DC son las medias de cada bloque

Imagen DC

DPCM

Secuencial

bloques de 4×4 (en realidad de 8×8)



asigno las palabras de bits a los símbolos (más bits a los menos probables)

41 JPEG

estas tablas van en el fichero JPEG
si no está, se utiliza la tabla estándar

2

MPEG

- MPEG (Moving Pictures Expert Group) en 1982
- Sus bases:
  - JPEG (elimina redundancia espacial)
  - H.261 (estándar de videoconferencia para reducir redundancia temporal)
- Su objetivo: Compresión de vídeo eliminando redundancias:
  - Espaciales (JPEG: codificación por transformadas)
  - Temporales (H.261: diferencia compensada en movimiento)
- Introducción de nuevas mejoras
  - Acceso aleatorio.
  - Búsqueda rápido.
  - Retroceso.

42 MPEG

MPEG: Compresión de vídeo

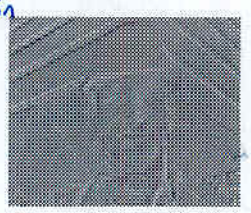
secuencia foreman

t

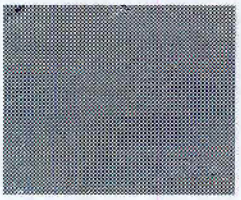
t+1

sin estimación de movimiento

resta de imágenes



Potencia de error=99.00



Potencia de error=28.10

con estimación de movimiento por bloques

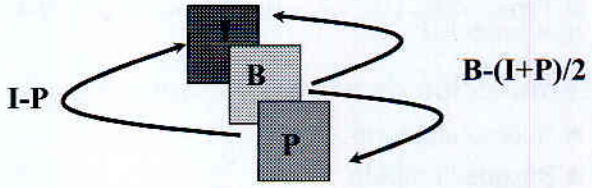
43

MPEG

ACCESO ALEATORIO

TIPOS DE IMÁGENES

- Imágenes Intraframe: I (se codifican como imagen fija) ← Permite acc. aleat. Poca compresión
- Imágenes Predictivas: P (se codifica con la diferencia entre ella y la anterior I o P)
- Imágenes Bidireccionales: B (se codifica como la diferencia entre ella y la media entre la anterior I o P y la siguiente I o P) ← Peor acc. aleat. Mejor compresión

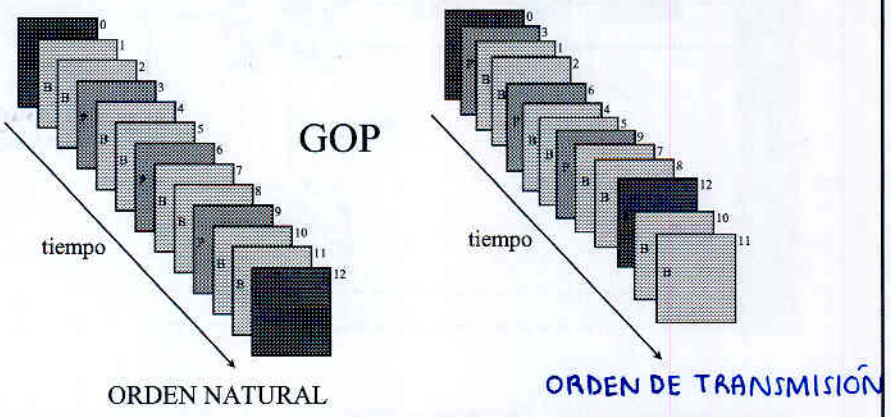


44 MPEG

MPEG

ACCESO ALEATORIO

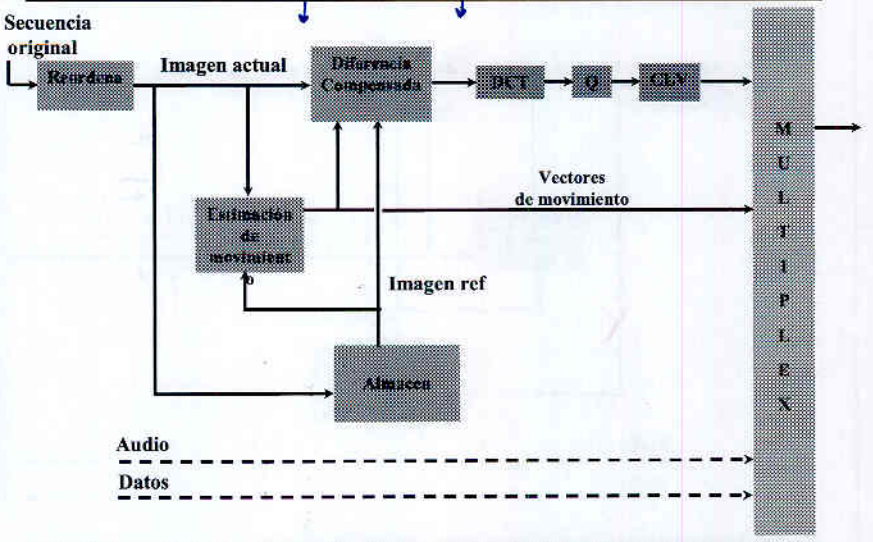
REORDENAMIENTO DE IMÁGENES

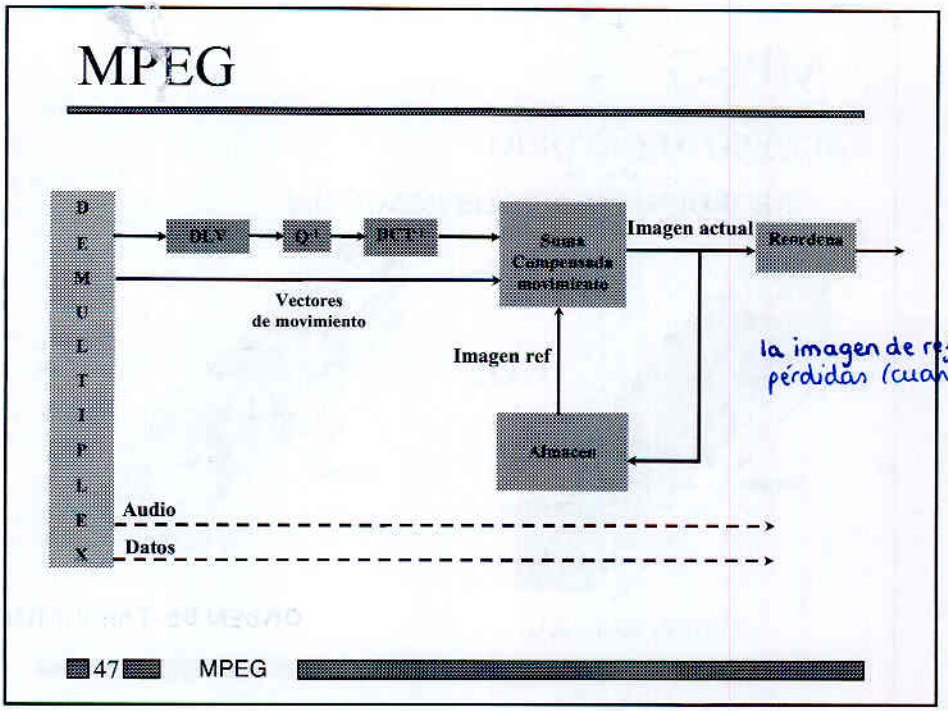


Antes de calcular la diferencia, se estima el movimiento y se compensa

Tras la diferencia la codifico como si fuera una imagen fija

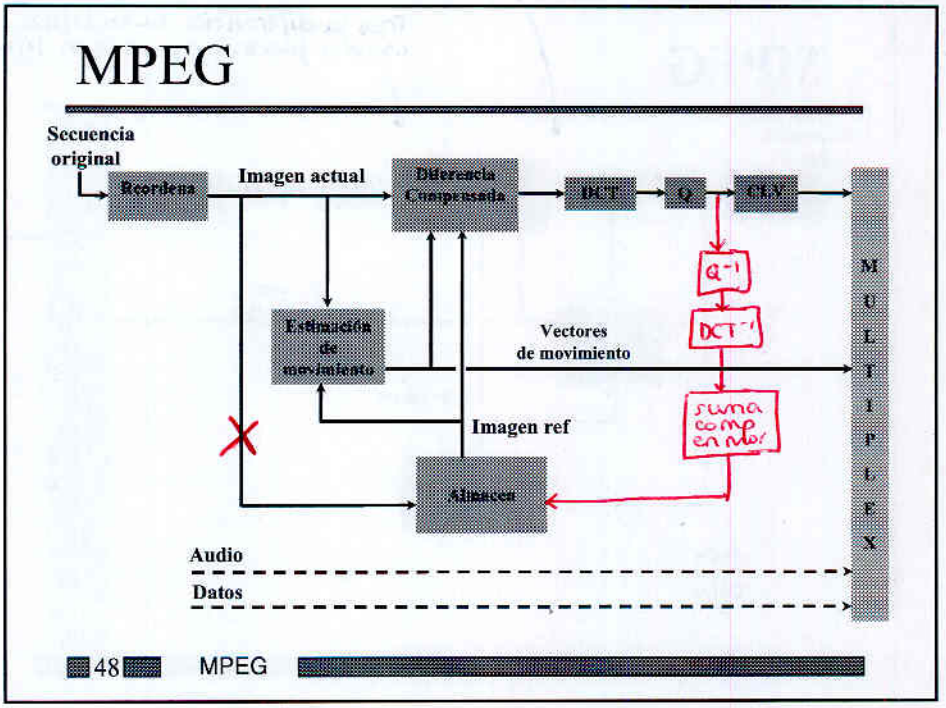
MPEG

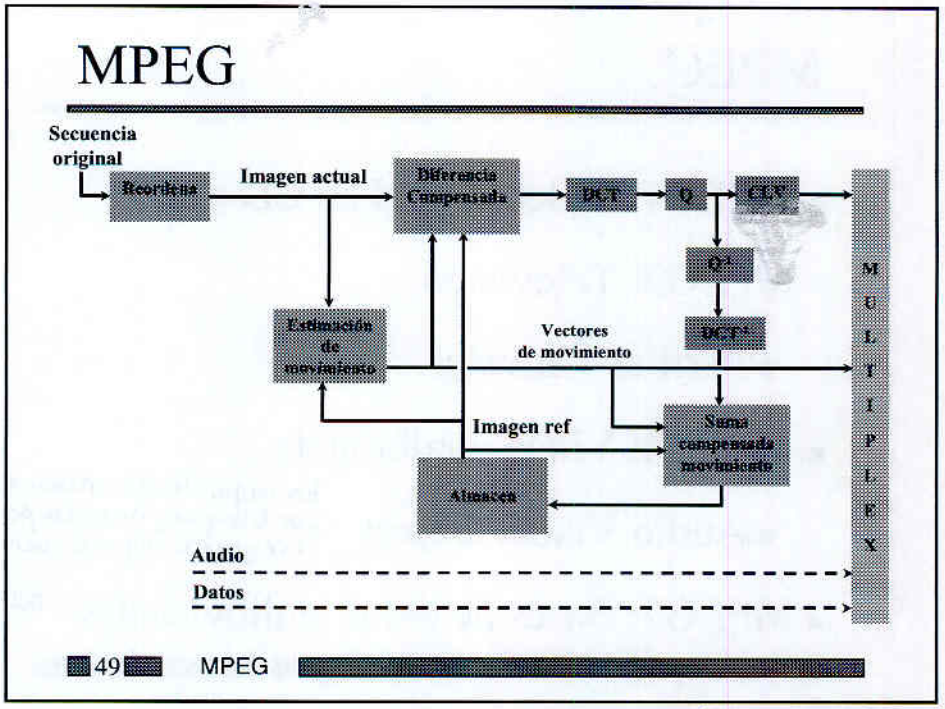




la imagen de ref tiene pérdidas (cuantificación)

merece la pena que el codificador utilice como referencia lo mismo que tendrá el receptor (con errores de cuantif.)

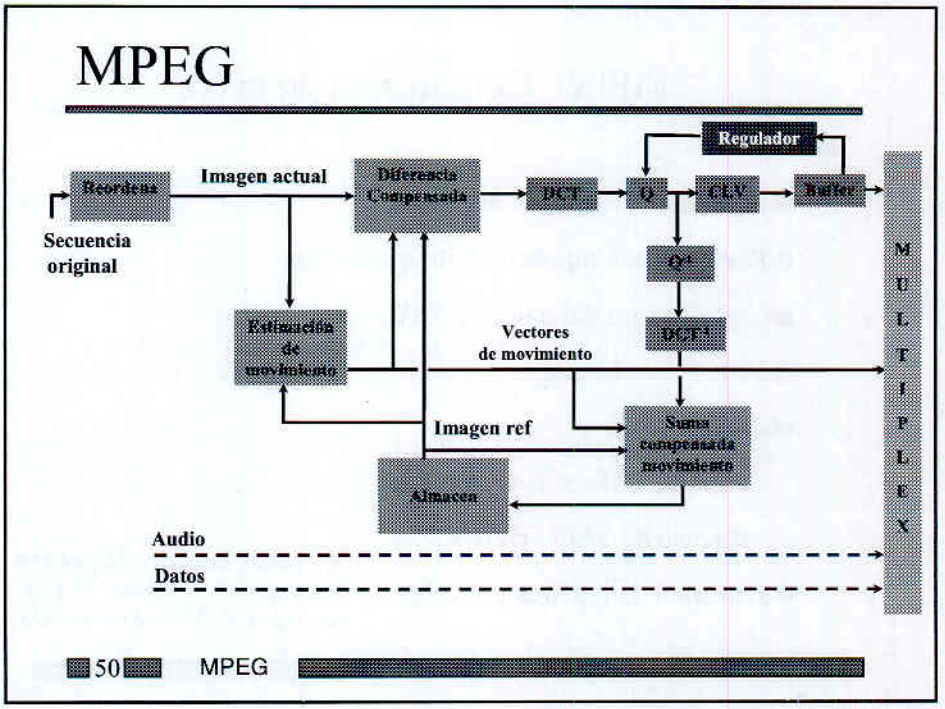




Lo que sale del multiplexor idealmente tendrá un régimen binario constante

↓

se introduce un regulador para cuantificar peor cuando haya mucho movimiento y al buffer le lleguen muchos bits.



mucho movimiento → peor calidad
poco movimiento → mejor calidad

} la calidad subjetiva se mantiene

MPEG

- MPEG1: Vídeo digital en CD
- MPEG2: Televisión
  - Perfiles y niveles
- MPEG4: Video conferencia
  - Audio Visual Object
- MPEG7: Bases de datos audiovisuales

En lugar de estimación de movimiento por bloques, hacerlo por objetos (requiere segmentación en objetos)

Detección de personajes, caras, ...

MPEG: Compresión de vídeo

- MPEG-2
  - Resolución espacial: 768 x 576
  - Resolución temporal: 30 imágenes/seg
  - Codificación del color: Y:U:V
  - Vídeo entrelazado
  - Aplicaciones
    - desde 5 Mb/s: PAL y NTSC
    - hasta 80 Mb/s: HDTV
  - Jerarquía de perfiles y niveles

El codificador depende del perfil/nivel escogido. Deben ir por 'capas' para que un decodificador malo pueda decodificar (aunque sea en la correspondiente calidad mala) una señal de perfil alto.

MPEG: Compresión de vídeo

- MPEG-2: Jerarquía de perfiles y niveles
  - Nivel:Rango de parámetros soportado por la implementación de cada servicio

Nivel	R. horizontal (muestras/línea)	R. vertical (líneas/imagen)	R. temporal (imágenes/s)
Bajo	352	288	30
Principal	720	576	30
Alto 1440	1440	1152	60
Alto	1920	1152	60

MPEG: Compresión de vídeo

- MPEG-2: Jerarquía de perfiles y niveles
  - Perfil:Conjunto de algoritmos añadidos a un perfil
    - Simple:
      - Vídeo entrelazado
      - Acceso aleatorio
      - Imágenes I y P
      - Color: 4:2:0 (crominancia diezmada por 2 en horizontal y vertical)
    - Principal
      - Introduce imágenes tipo B
    - Escalable en S/N
      - Introduce algoritmo para codificar en función de la S/N
    - Escalable espacialmente
      - Introduce algoritmo para codificar en función de la resolución
    - Alto
      - Color: 4:2:2 (crominancia diezmada por 2 sólo en horizontal)

De la intersección de un perfil con un nivel se obtiene una tasa binaria

MPEG: Compresión de vídeo

- MPEG-2: Jerarquía de perfiles y niveles

Perfil / Nivel	Alto 4:2:2 I P B	Escalable espacialmente 4:2:0 I P B	Escalable en SNR 4:2:0 I P B	Principal 4:2:0 I P B	Simple 4:2:0 I P
Alto 1920x1152	100 Mb/s			80 Mb/s	
Alto 1440 1440x1152	80 Mb/s	60 Mb/s		60 Mb/s	
Principal 720x576	20 Mb/s		15 Mb/s	15 Mb/s	15 Mb/s
Bajo 352x288			4 Mb/s	4 Mb/s	

TV Digital de alta definición

TV Digital Estándar

MPEG: Compresión de vídeo

- CALIDAD/VELOCIDAD
  - Dibujos animados: 2Mb/s
  - Telediario: 3Mb/s
  - Película: 4Mb/s
  - Partido de fútbol: 6 a 8 Mb/s

MPEG: Compresión de vídeo

- DIFERENCIAS MPEG-1 Y MPEG-2

Estándar (año)	**MPEG-1(1992)**	**MPEG-2(1994)**
Aplicación	Vídeo digital CD-ROM	TV digital (HDTV)
Resolución espacial	(1/4) TV: 288x360	TV 576x720 (4 TV 1152x1440)
Resolución temporal	25-30 imágenes/seg	50-60 imágenes/seg (100-120 imágenes/seg)
Tasa binaria	1.5 Mb/s	15Mb/s (60 Mb/s)
Calidad	VHS	NTSC/PAL
Compresión	20-30	30-40

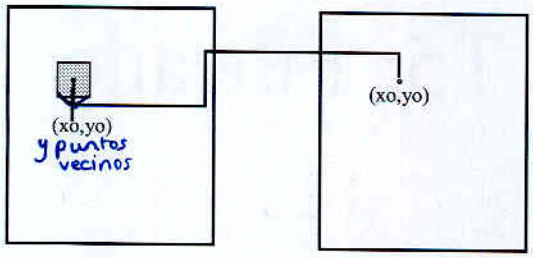
T5: Filtrado

Asignatura:Tratamiento digital de imagen

Índice

- Introducción
- Filtros lineales e invariantes
- Filtros no lineales
- Filtros espacio-temporales

Introducción



Operador local

Índice

- Introducción
- Filtros lineales e invariantes
- Filtros no lineales
- Filtros espacio-temporales

Filtros lineales e invariantes (LSI)

$h[m,n]$ o $H(e^{j\omega_x}, e^{j\omega_y})$

$$y[m,n] = x[m,n] * h[m,n] = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} x[k,l]h[m-k,n-l]$$

$$Y(e^{j\omega_x}, e^{j\omega_y}) = X(e^{j\omega_x}, e^{j\omega_y})\, H(e^{j\omega_x}, e^{j\omega_y})$$

Filtros lineales e invariantes (LSI)

FIR	IIR
Requiere un nº finito de operaciones/píxel: implementables	Implementables sólo si recuerrencia
Siempre estables	Difícil verificar estabilidad (polos de H(z) dentro del círculo unidad)
Fase lineal si simetría	Nunca fase lineal
	En muchos casos menos complejos (menos operaciones)

Filtros lineales e invariantes (LSI)

Filtros FIR

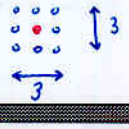
- h[m,n] simétrica: fase lineal
- h[m,n] simétrica respecto al origen: fase cero
- Si no fase cero: desplazamiento de ±1/2 píxel
- Para conseguier fase cero: tamaño impar

la pendiente de la fase (derivada) es el retardo de grupo, y es el punto de simetría de h[m,n]

retardo de grupo constante ⇓ la imagen no se deforma

retardo cero ⇓ la imagen no se traslada

(para poder elegir un punto como centro)

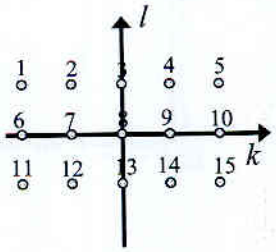


Filtros LSI: Implementación

- Implementación del filtrado
- Tipos de filtros LSI
  - Suavizadores
    - De media
    - De línea
  - Realzadores de bordes
  - Gradientes
  - Direccionales
- Filtrado de imágenes en color

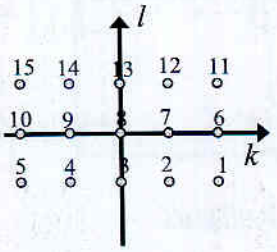
Filtros LSI: Implementación

$$y[m,n] = x[m,n] * h[m,n] = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} x[k,l]h[m-k,n-l]$$



$h[k,l]$

(indica a dónde se va una muestra origen)



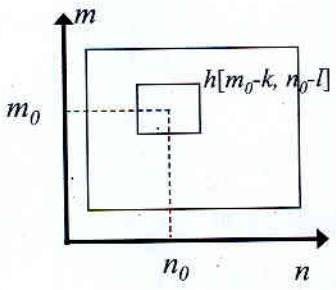
$h[-k,-l]$

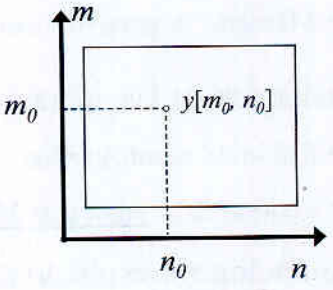
(indica de dónde viene una muestra destino)

nota de Pak:

Filtros LSI: Implementación

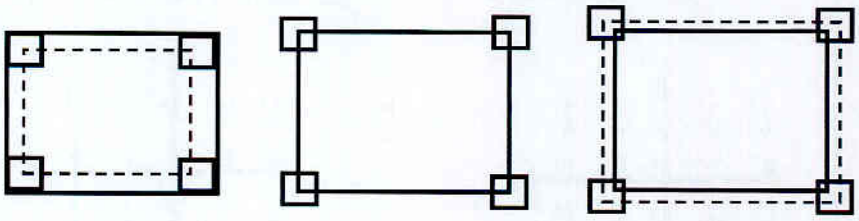
$$y[m,n] = x[m,n] * h[m,n] = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} x[k,l]h[m-k,n-l]$$





Filtros LSI: Implementación

TRATAMIENTO DE BORDES ⇒ tamaño de la imagen final



Relleno

- Espejo
- Ceros
- Gris medio

Filtros LSI: Implementación

3 problemas:

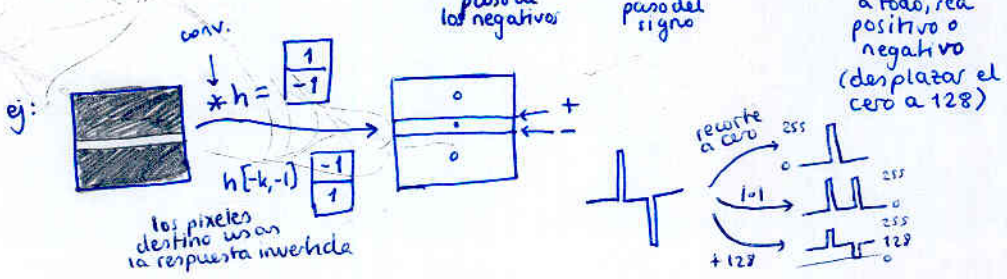
Efectos de precisión finita

- Cuantificación coeficientes
  - Máscara: Enteros+divisor $h[m,n]=\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$
- Redondeo en las operaciones
  - Ruido de cuantificación $\sigma_e^2=\frac{\Delta^2}{12}$
  - Mejorar S/N: aumentar MD antes de filtrar
- Saturaciones (resultado>255 o resultado<0)
  - result>255 result=255
  - result<0 result=0 o result=|result| o result=result+128

paso de los negativos

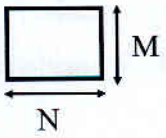
paso del signo

a todo, sea positivo o negativo (desplazar el cero a 128)



Filtros LSI: Implementación

IMPLEMENTACIONES EFICIENTES



Ops/pixel: MxN productos

M×N-1 sumas

- Simetrías (par o impar) en h
  - a x s_1+a x s_2= a x (s1+ s2)
  - a x s1-a x s2= a x (s1- s2)

coef filtro ↑ ↑ pixel imagen

ej:

$$h[m,n]=\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Filtros LSI: Implementación

IMPLEMENTACIONES EFICIENTES

Lo separo en respuestas en cada dimensión

- Separabilidad

$$y[m,n]=x[m,n]*h[m,n]=(x[m,n]*h_1[m])*h_2[n]$$

$$h[m,n]=h_1[m]\times h_2[n]$$

$$h[m,n]=\frac{1}{25}\begin{bmatrix}1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\end{bmatrix}=\frac{1}{5}\begin{bmatrix}1\\ 1\\ 1\\ 1\\ 1\end{bmatrix}\times\frac{1}{5}\begin{bmatrix}1 & 1 & 1 & 1 & 1\end{bmatrix}$$

convierto M×N prod en

Ops/pixel: M+N productos

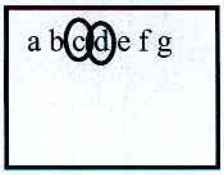
Filtros LSI: Implementación

IMPLEMENTACIONES EFICIENTES

- Recursividad (si todos los elementos iguales)

Desplazar la máscara= sumar la intensidad del pixel que entra y restar la del que sale

$$h[m,n]=\frac{1}{5}[1\ 1\ 1\ 1\ 1]$$



paso1: aux = (a+b+c+d+e)

salida=aux/5

paso2: aux = (b+c+d+e+f) = aux - a + f

salida=aux/5

ahorra un montón

Filtros LSI: Implementación

IMPLEMENTACIONES EFICIENTES

- Descomposición en cascada

$$y[m,n]=x[m,n]*h[m,n]=x[m,n]*(h_1[m,n]*h_2[m,n])$$

Duración h: $(M_1+M_2-1)\times(N_1+N_2-1)$

$$h[m,n]=h_1[m,n]*h_2[m,n]$$

Duración h_1: M_1xN_1 Duración h_2: M_2xN_2

Ops/pixel sin descomposición: $(M_1+M_2-1)\times(N_1+N_2-1)$

Ops/pixel con descomposición: $(M_1 x\ N_1) + (M_2 \times N_2)$

Filtros LSI: Implementación

IMPLEMENTACIONES EFICIENTES

- Descomposición en cascada

$$h[m,n]=\frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\ 4 & 16 & 24 & 16 & 4\\ 6 & 24 & 36 & 24 & 6\\ 4 & 16 & 24 & 16 & 4\\ 1 & 4 & 6 & 4 & 1\end{bmatrix}$$

$$h[m,n]=\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} * \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Filtros LSI: Implementación

- Implementación del filtrado
- Tipos de filtros LSI
  - Suavizadores o promediadores
    - De media
    - De línea
  - Realzadores de bordes
  - Gradientes
  - Direccionales
- Filtrado de imágenes en color

utilidad típica → reducir ruido (sólo para ruido uniforme)

Filtros LSI: Filtros suavizadores

- Filtros paso-bajo
- Suavizan los bordes (emborronan)
- Utilidad: reducción de ruido (sólo ruido uniforme, no impulsivo)

$$\sigma_i^2 \longrightarrow \boxed{h[m,n]} \longrightarrow \sigma_{out}^2$$

$$\sigma_{out}^2 = \sigma_i^2 \sum_m \sum_n |h[m,n]|^2$$

↳ factor de reducción de ruido

cuanto más pequeño es, más ruido reduce.

Condición: $\sum_m \sum_n h[m,n] = 1$ Conservar el brillo medio de la imagen

Promediadores de área

$$h_a[m,n] = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad h_b[m,n] = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$h_c[m,n] = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Promediadores de área

$$h_d[m,n] = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$h_e[m,n] = \frac{1}{120}\begin{bmatrix} 0 & 1 & 2 & 1 & 0 \\ 1 & 7 & 13 & 7 & 1 \\ 2 & 13 & 24 & 13 & 2 \\ 1 & 7 & 13 & 7 & 1 \\ 0 & 1 & 2 & 1 & 0 \end{bmatrix}$$

Promediadores de área

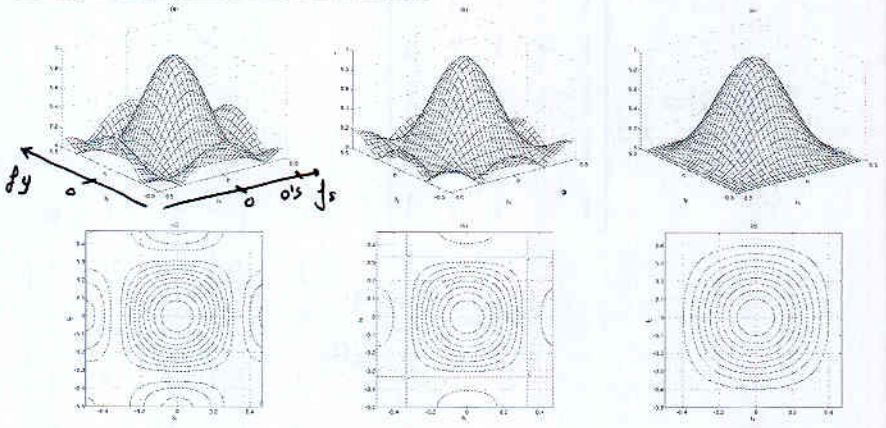
Filtro	Red.Ruido	Separable
a	**1/9**	**Si**
b	**3/25**	**No**
c	**9/64**	**Si**
d	**1/25**	**Si**
e	**23/225**	**No**

Se reduce más ruido a medida que crece la resp. al impulso.

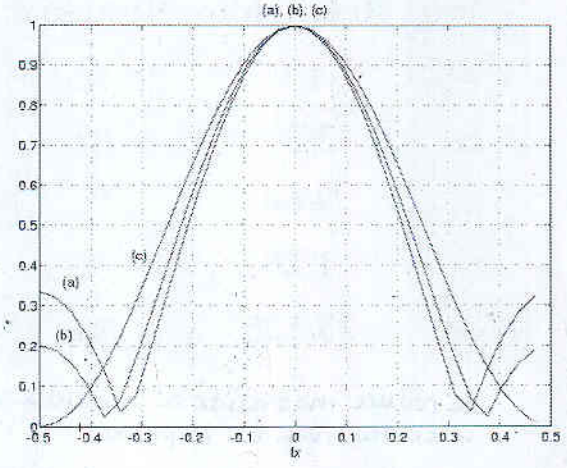
Para un mismo tamaño reduce más la respuesta plana (más estrechas en frecuencia pero más lóbulos secundarios)

Promediadores de área

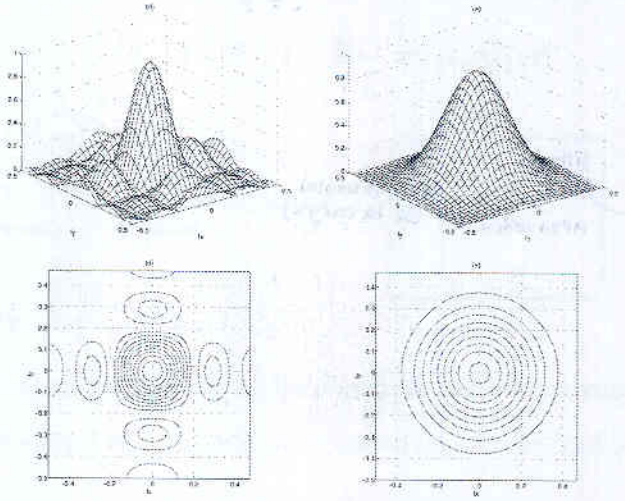
Respuesta en frecuencia



Promediadores de área



Promediadores de área



todo unos 5×5

no todo unos 5×5

puntitos blancos: ruido de sal

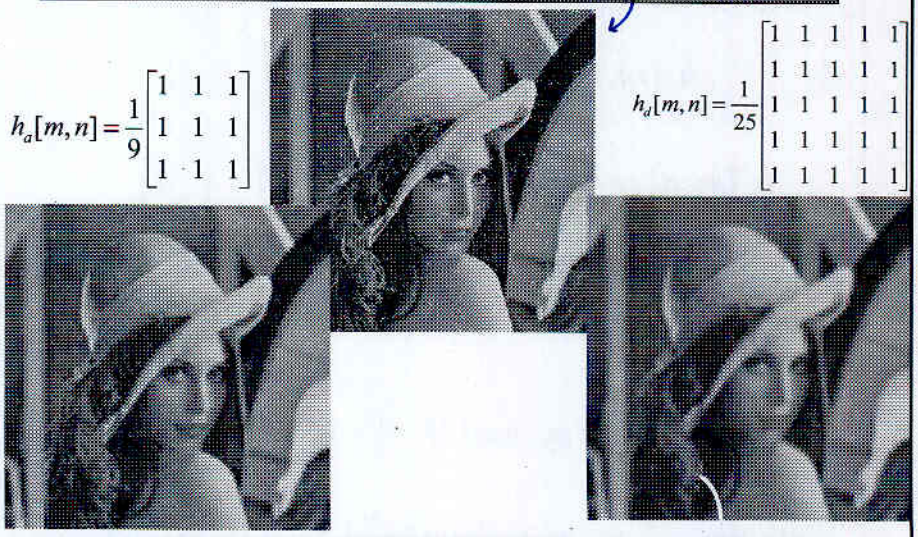
puntitos blancos y negros: ruido de sal y pimienta

Promediadores de área

ruido impulsivo

$$h_a[m,n] = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$h_d[m,n] = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

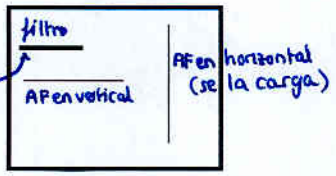


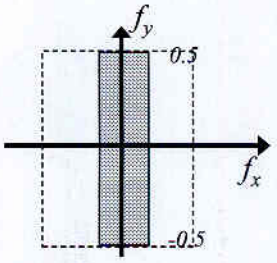
Promediadores de línea

$L = 5$

$$h_f[m,n] = \frac{1}{5}[1 \quad 1 \quad 1 \quad 1 \quad 1]$$

truco: alta frecuencia en vertical





- Suavizan bordes perpendiculares a la dirección de promediado

Reducen ancho de banda en la dirección de promediado

Reduce ruido sólo en la dirección de promediado

Promediadores de línea

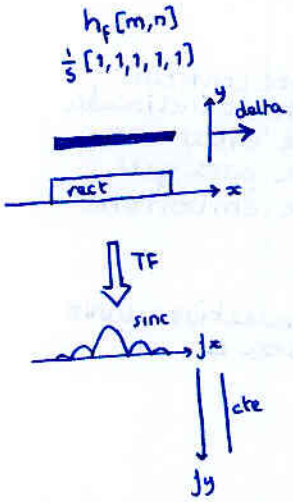
$$h_f[m,n] = \frac{1}{5}[1 \quad 1 \quad 1 \quad 1 \quad 1]$$

$$h_g[m,n] = \frac{1}{9}[1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1]$$

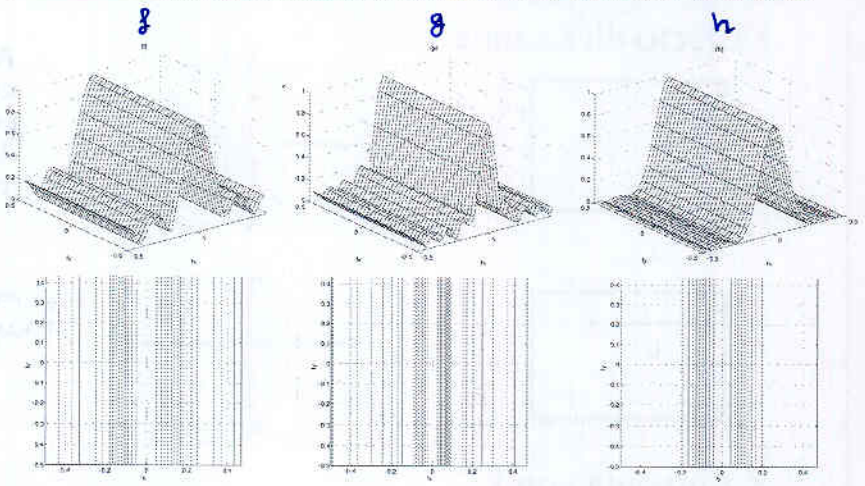
$$h_h[m,n] = \frac{1}{111}[2 \quad 5 \quad 14 \quad 22 \quad 25 \quad 22 \quad 14 \quad 5 \quad 2]$$

$$h_i[m,n] = \frac{1}{5}\, diagonal\{1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1\} =$$

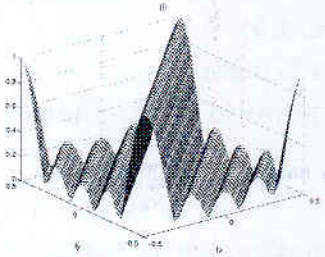
$$\frac{1}{5}\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$



Promediadores de línea

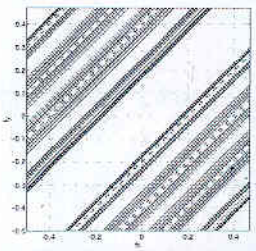


Promediadores de línea



$$h = \frac{1}{L}\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & \ddots \end{bmatrix}$$

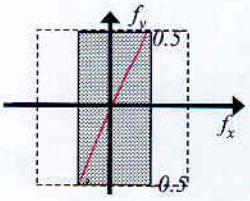
TF



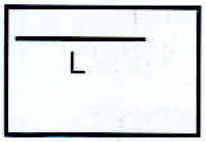
Los filtros de línea reducen el ancho de banda en la dirección de promediado

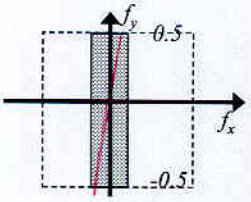
Promediadores de línea

- Efecto direccional



Permite controlar cuanto de inclinada puede estar una linea para que no se emborrone





es más selectivo cuanto mayor sea L

- Aumenta con L

Filtros LSI: Realzadores de contornos

- Filtros paso-alto
- Realzan los bordes
- Problema: amplifican también el ruido

imagen realzada = imagen + α · imagen de altas frec

$$Realzada = I+\alpha(AF) = I+\alpha(I-BF) = (1+\alpha)I-\alpha(BF)$$

(muestra central ↓ promediador ↓)

$$\sum_m \sum_n h[m,n] = 1$$ Conservar el brillo medio de la imagen

A mayor α ⟶ *mayor realce*

Filtros LSI: Realzadores de contornos

Ejemplo $\quad Realzada=(1+\alpha)I-\alpha(BF)$

$\alpha=9$ $\qquad h_l[m,n]=\begin{bmatrix}-1 & -1 & -1\\ -1 & 9 & -1\\ -1 & -1 & -1\end{bmatrix}$

típica máscara realzadora de bordes

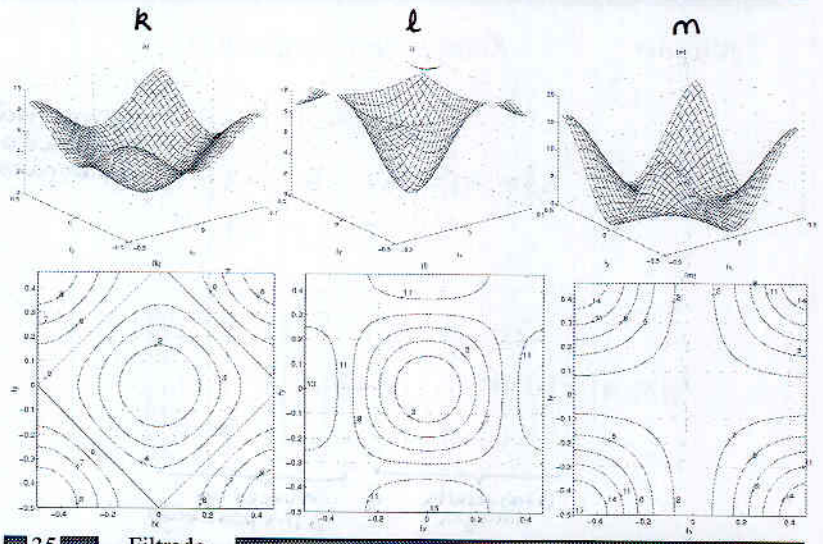
$$h_l[m,n]=10\underbrace{\begin{bmatrix}0 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}}_{(1+\alpha)\text{veces la imagen}}-9\underbrace{\left[\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}\right]}_{\alpha\text{ veces un filtro paso bajo}}$$

Realzadores de contornos

$$h_k[m,n]=\begin{bmatrix}0 & -1 & 0\\ -1 & 5 & -\\ 0 & -1 & 0\end{bmatrix}\qquad h_l[m,n]=\begin{bmatrix}-1 & -1 & -1\\ -1 & 9 & -1\\ -1 & -1 & -1\end{bmatrix}$$

$$h_m[m,n]=\begin{bmatrix}1 & -2 & 1\\ -2 & 5 & -2\\ 1 & -2 & 1\end{bmatrix}$$

Filtros LSI: Realzadores de contornos



35 Filtrado

Filtros LSI: Realzadores de contornos

ORIGINAL **REALZADA**

36 Filtrado

Filtros LSI: Gradientes

- Detectores de bordes

gradiente horizontal - filtro direccional

$$\nabla f(x,y) = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

Dirección: la de máximo crecimiento de f
Módulo: pendiente de f en esa dirección

$$|\nabla f(x,y)| = \begin{cases} G_1 = \sqrt{g_h^2 + g_v^2} \\ G_2 = |g_h| + |g_v| \end{cases}$$

no es exacto pero es más rapido

$$TF\{\nabla f(x,y)\} = [j\omega_x F(\omega_x,\omega_y), j\omega_y F(\omega_x,\omega_y)]$$

Filtros LSI: Gradientes

$$V_{ga}[m,n] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix} \quad H_{ga}[m,n] = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

pixel gradient

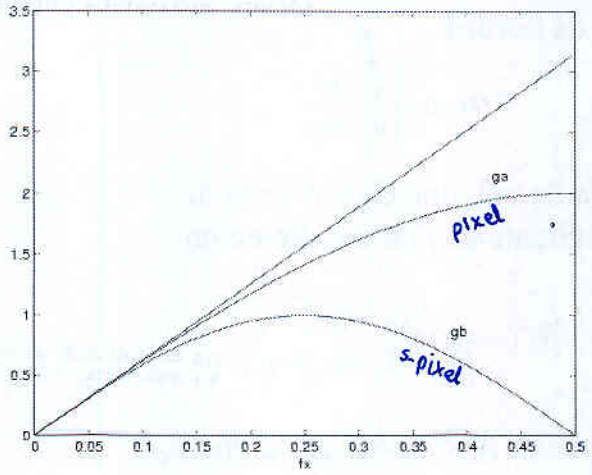
$$V_{gb}[m,n] = \frac{1}{2}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \quad H_{gb}[m,n] = \frac{1}{2}\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

separated
s - pixel gradient

detecta bordes horizontales

los divisores de los gradientes es la distancia de los pixeles que se restan

Filtros LSI: Gradientes



Ideal

$$|j2\pi f|$$

la respuesta del 'pixel-gradient' es más cercana a la ideal; como desventaja, desplaza la imagen medio pixel por no estar centrado

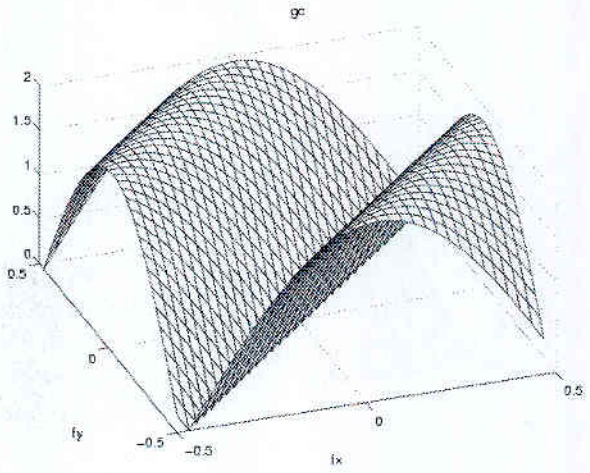
si la diferenciación es en el eje x, la 'V' aparece en f_x

Filtros LSI: Gradientes

- Gradientes diagonales: Roberts

$$D_{gc}[m,n] = \frac{1}{\sqrt{2}}\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad D'_{gc}[m,n] = \frac{1}{\sqrt{2}}\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Filtros LSI: Gradientes



Filtros LSI: Gradientes

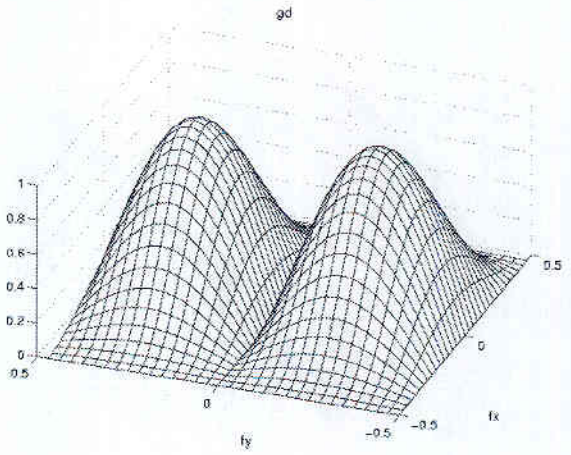
- Gradientes con promediado: Sobel, Prewitt,…

$$H_{gd}[m,n]=\frac{1}{8}\begin{bmatrix}1 & 0 & -1\\ 2 & 0 & -2\\ 1 & 0 & -1\end{bmatrix} \qquad V_{gd}[m,n]=\frac{1}{8}\begin{bmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix}$$

en una dirección promedia (elimina ruido)
y en la otra hace el gradiente (detecta borde)

El divisor es
1+2+1 = $\frac{1}{4}$ para mantener el brillo medio
$\frac{1}{2}$ por la distancia de la resta

Filtros LSI: Gradientes



Hay una V en fy → grad en y
Hay un F.P.Bj en fx → promedio en x

Filtros LSI: Gradientes

Filtros LSI: Filtros direccionales

- Detectar bordes en una dirección

$$H_{90^\circ}[m,n] = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad H_{45^\circ}[m,n] = \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$$

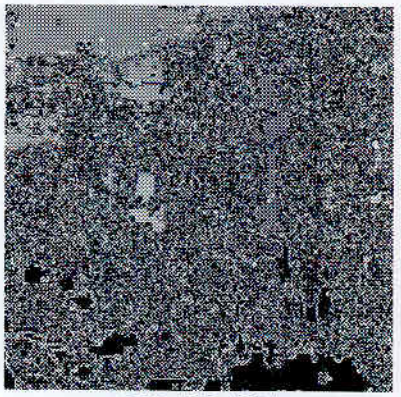
Filtrado de imágenes en color

Imagen verdadero color **FILTRADA**

Hace lo mismo para cada una de las componentes

Filtrado de imágenes en color

Imagen de paleta



FILTRADA

Índice

- Introducción
- Filtros lineales e invariantes
- Filtros no lineales
- Filtros espacio-temporales

Filtros no lineales

- Solucionar problemas filtros lineales
- Tipos de filtros no lineales
  - Reductores de ruido adaptativos → reducir ruido sin emborronar → detectar de bordes y quitar ruido en zonas uniformes sólo → la gente prefiere un borde ruidoso que uno borroso
  - Filtros de estadísticos ordenados ← los más usados para reducir ruido impulsivo
  - Reductores de ruido estacionario+impulsivo

Filtros no lineales : de mediana

11	12	12	13	14
10	11	12	13	15
10	10	14	16	15
12	9	15	8	9
11	9	8	9	9

ordenas los valores

8
9
10
11
(12) → 12
13
14
15
16

muy útil para reducir ruido impulsivo (siempre quita la raí, que queda arriba al ordenar)

Si hay un nº par de muestras, se toma la media de las dos centrales

Hay otros que cogen el máximo o el mínimo
*dilatación *erosión
Filtros de estadísticos ordenados

Algoritmos de morfología matemática

Filtros no lineales

Ruidosa

Media 3x3

Mediana 3x3

Filtros no lineales

- Ruido uniforme: filtro lineal
- Ruido impulsivo: filtro no lineal
- Ruido uniforme+impulsivo: algoritmo

11	12	12	13	14
10	11	12	13	15
10	10	14	16	15
12	9	15	8	9
11	9	8	9	9

8
9
10
11
12
13
14
15
16

← el ruido impulsivo cae fuera

→ media → 12

Índice

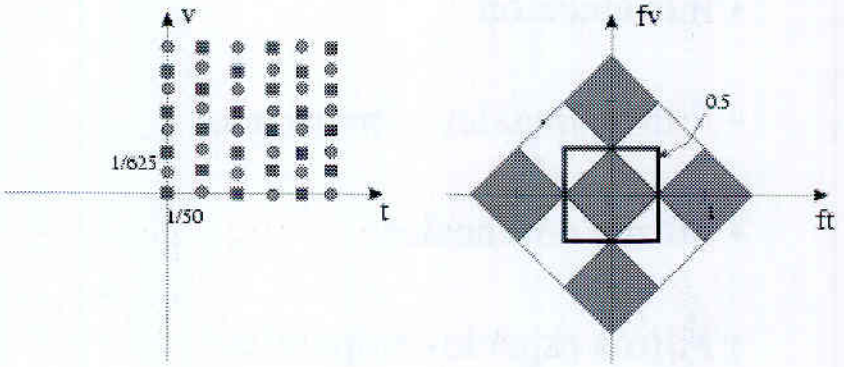
- Introducción
- Filtros lineales e invariantes
- Filtros no lineales
- Filtros espacio-temporales

Filtros espacio-temporales

$$f(x,t) = \cos(2\,\pi\,(f_{t0}t + f_{x0}x))$$

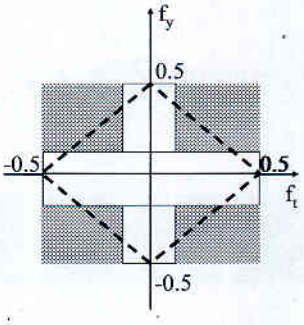
Filtros espacio-temporales

Cambio entrelazado-progresivo



55 Filtrado

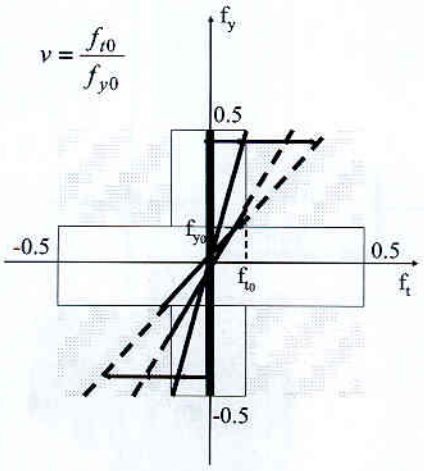
Eliminación ruido en secuencias



Gran detalle espacial **SI** baja frecuencia temporal

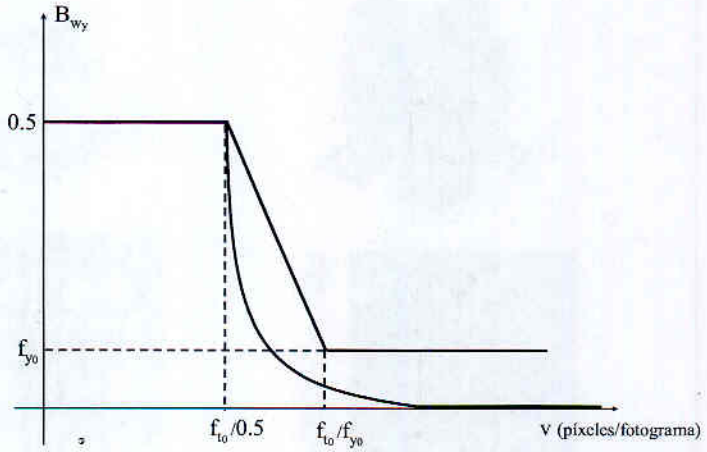
56

Funcionamiento del filtro esp-temp

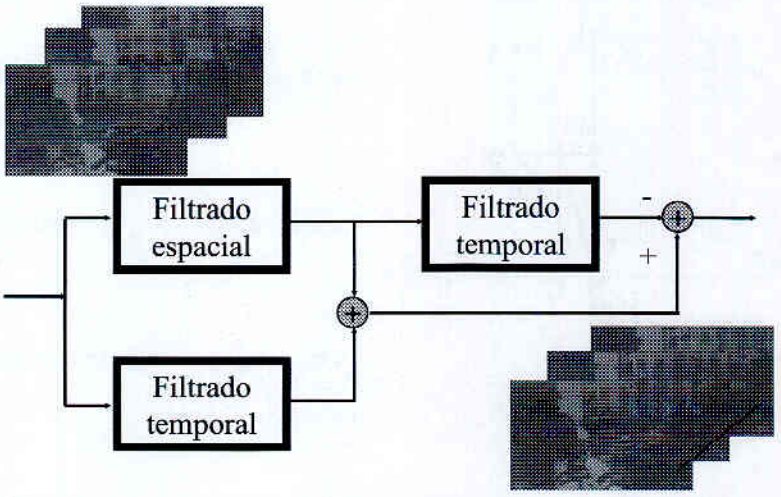


- Objetos con $v \leq f_{to}/0.5$ pasan intactos
- Si $f_{to}/0.5 \leq v < f_{to}/f_{yo}$ promedio espacial
- Si $v > f_{to}/f_{yo}$ ya no se emborrona más

Ancho de banda espacial

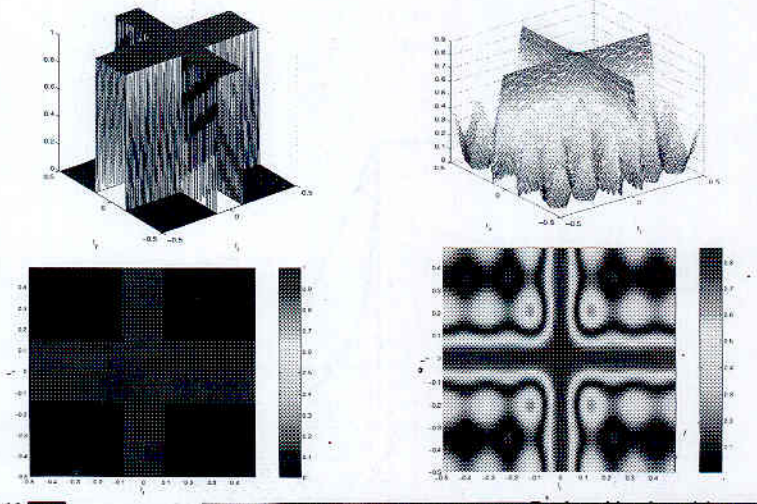


Implementación del filtro



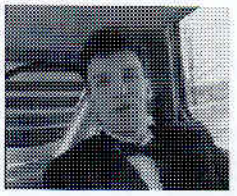
59

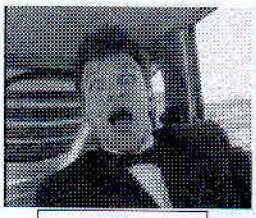
Filtro real vs. Filtro ideal



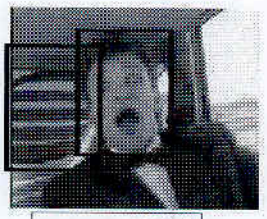
60

Resultados obtenidos

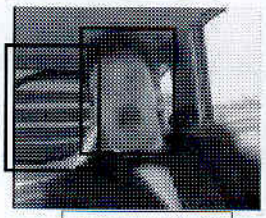




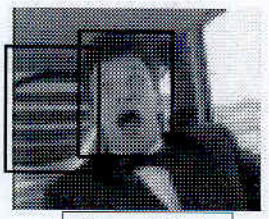
ruidosa



esp-temp:3x3x9



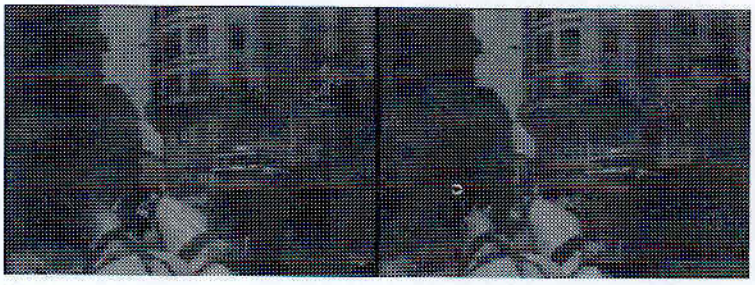
temporal: L=9



Espacial:3x3

61 Reducción de ruido

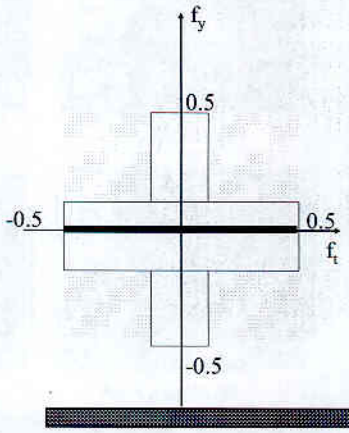
Eliminación ruido en secuencias



62 Filtrado

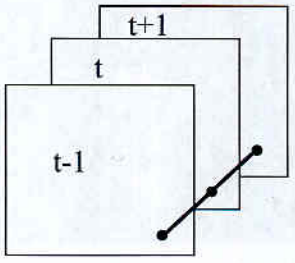
Resultados secuencias reales

Manchas mayores que la mitad del tamaño de la máscara espacial **NO** son eliminadas



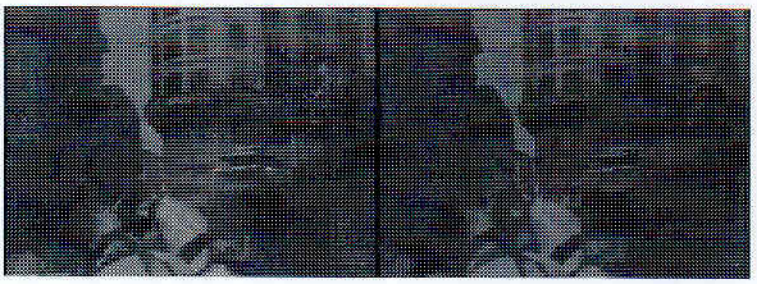
Eliminación ruido en secuencias

- Filtrado morfológico temporal



Apertura: Elimina objetos claros que no aparecen en los tres fotogramas

Eliminación ruido en secuencias



65 Filtrado