Universidad Autónoma de Baja California

Facultad de Ingeniería, Arquitectura y Diseño

Programa de Maestría y Doctorado en Ciencias e Ingeniería (MYDCI)

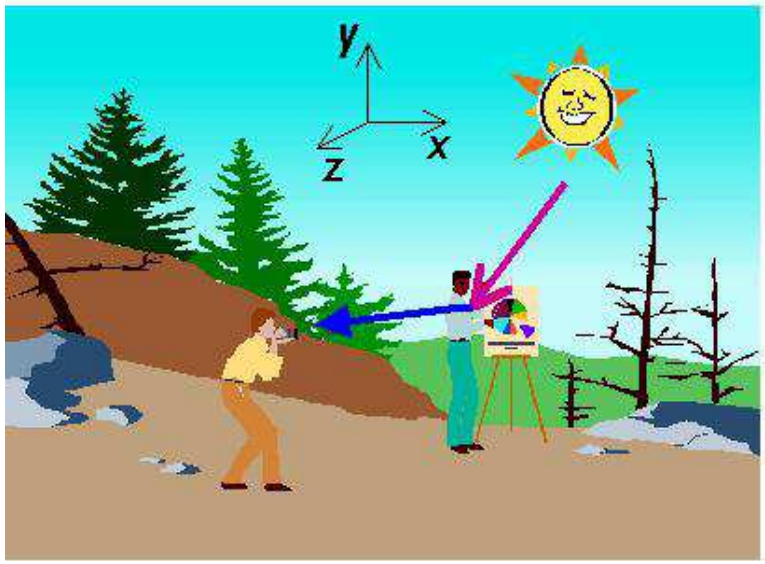
Notas del Curso de Procesamiento de Imágenes Digitales (PID)

Onur G. Guleryuz
Miguel Enrique Martínez Rosas
Manuel Moisés Miranda Velasco
Carlos Gómez Agis
José Antonio Michel Macarty

Ensenada, B.C. Agosto 2016.

Formación de Imagenes

- Una imágen natural requiere de una fuente de iluminación *(λ: longitud de onda)*
  - $E(x, y, z, \lambda)$: luz incidente en un punto *(x, y, z coordenadas del punto)*
- Cada punto en la escena tienen una función de reflectividad.
  - $r(x, y, z, \lambda)$: function de reflectividad
- La luz reflejada en un punto es capturada por un dispositivo de imagen.
  - $c(x, y, z, \lambda) = E(x, y, z, \lambda) \times r(x, y, z, \lambda)$: luz reflejada.



$$\rightarrow \quad E(x, y, z, \lambda)$$

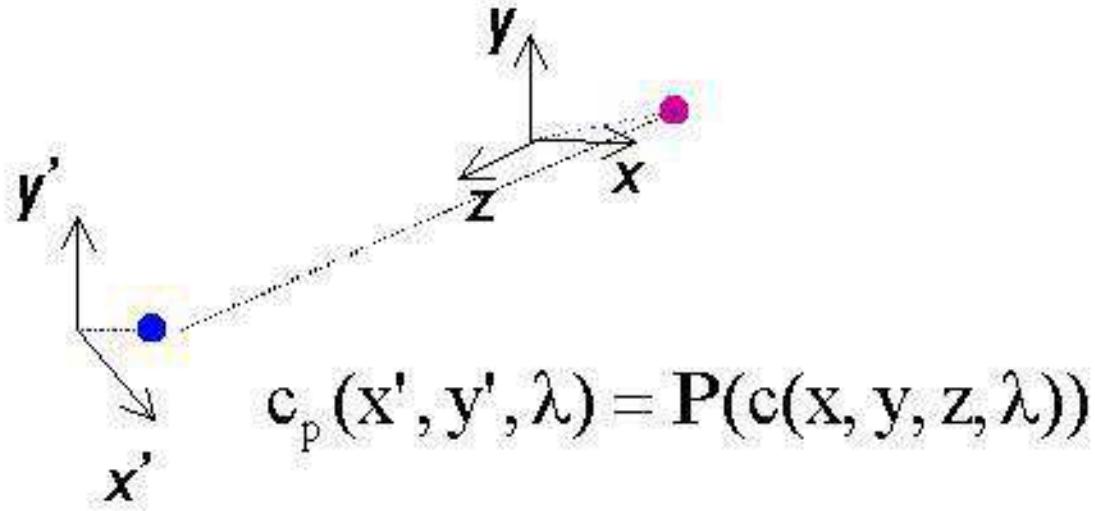
$$\rightarrow \quad c(x, y, z, \lambda) = E(x, y, z, \lambda) \cdot r(x, y, z, \lambda)$$

$$\text{Camera}(c(x, y, z, \lambda)) =$$

Dentro de la Cámara - Proyección

$\mathbf{Camera}(c(x, y, z, \lambda)) =$ 

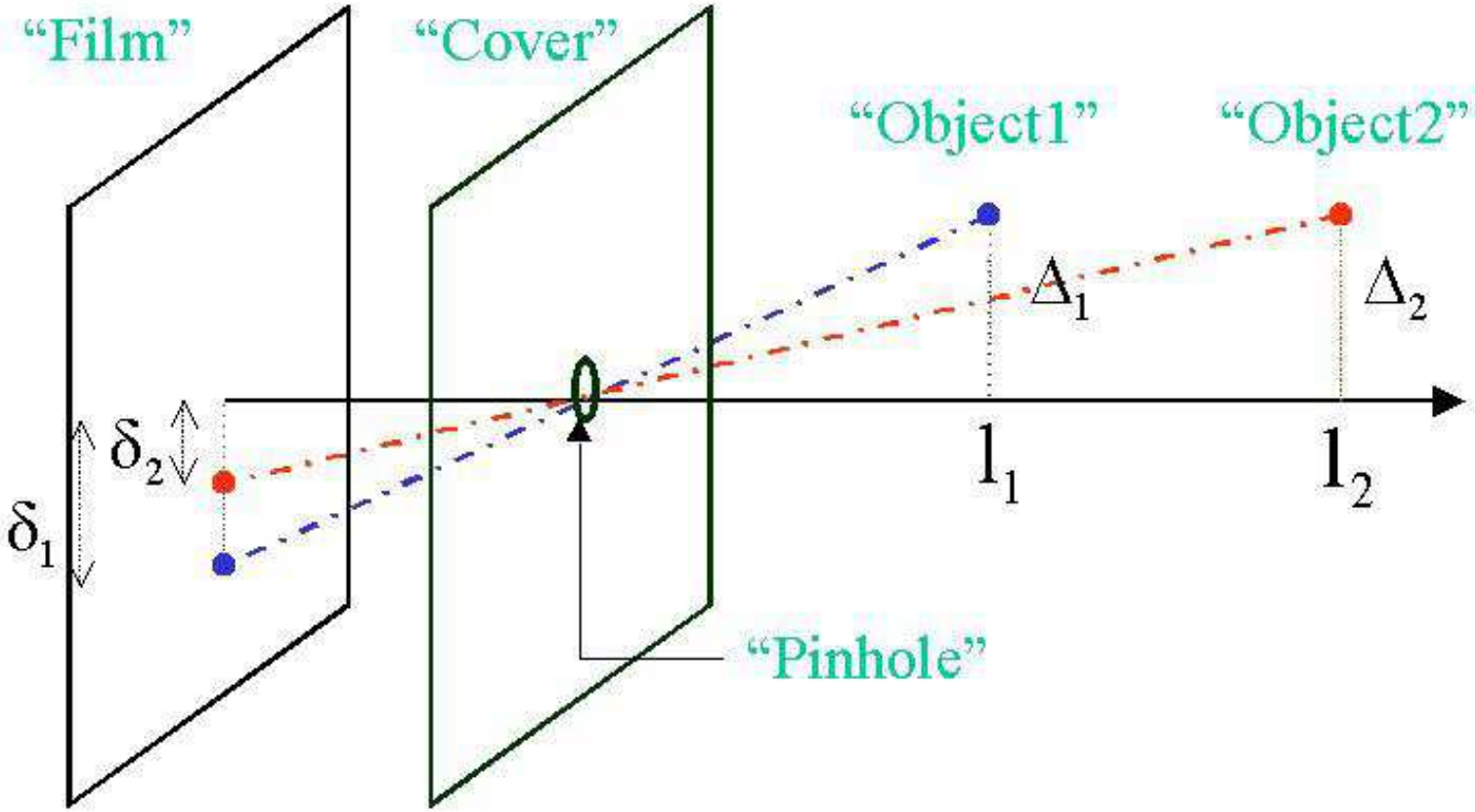
- Proyección $(\mathcal{P})$ de las coordenadas (x, y, z) a las coordenadas de la cámara o imagen (x', y') $\left[c_p(x', y', \lambda) = \mathcal{P}(c(x, y, z, \lambda))\right]$.



Proyecciones

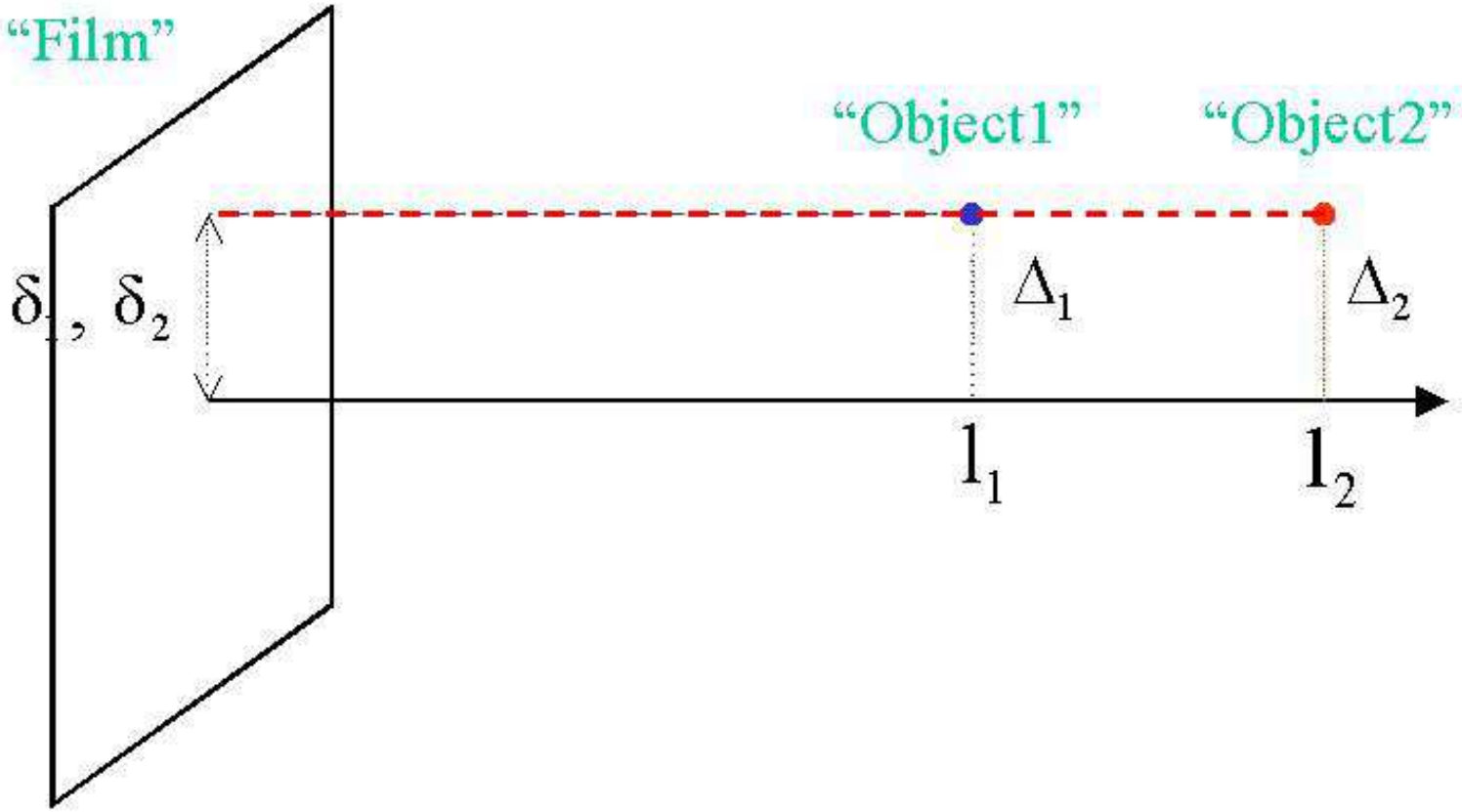
- Existen dos tipos de proyecciones ($\mathcal{P}$) que nos interesan:
  1. Proyección de Perspectiva
     - Los objetos cercanos al dispositivo de captura parecen más grandes. Se deben considerar múltiples situaciones de formación de la imagen que caen dentro de esta categoría, incluyendo las imágenes tomadas por una cámara o por el *ojo humano*.
  2. Proyección Ortográfica
     - Esta es "no-natural". Los objetos parecen del mismo tamaño independientemente de la distancia a la que se encuentren o del "dispositivo de captura".
- Ambos tipos de proyecciones pueden ser representados por medio de fórmulas matemáticas. La proyección Ortográfica es *más sencilla* y en ocasiones se utiliza por conveniencia matemática. Para obtener más detalles consultar [1].

Ejemplo - Perspectiva



- Proyección Perspectiva: $\Delta_1 = \Delta_2$, $l_1 < l_2 \rightarrow \delta_2 < \delta_1$.

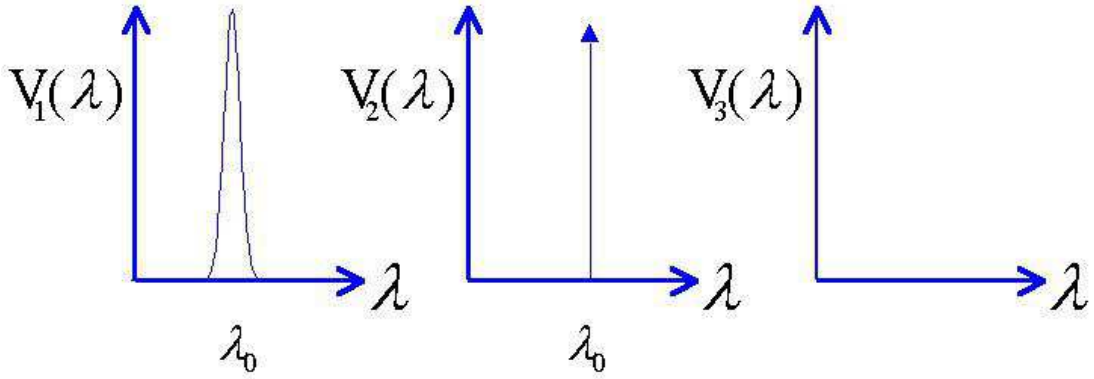
Ejemplo - Ortográfico



- Proyección Ortográfica: $\Delta_1 = \Delta_2$, $l_1 < l_2 \rightarrow \delta_2 = \delta_1$.

Dentro de la Cámara - Sensibilidad

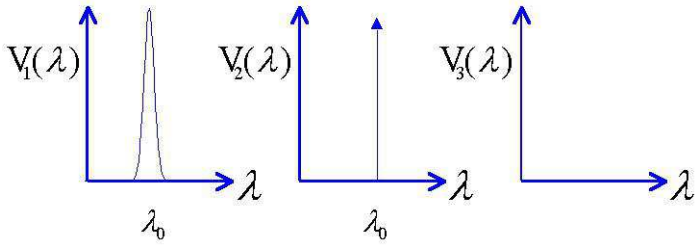
- Una vez que se tiene $c_p(x', y', \lambda)$ se toman en consideración las características del dispositivo de captura.
- $V(\lambda)$ es la *función de sensibilidad* del dispositivo de captura. Cada dispositivo poseé una función que determina que tan sensible es en el intervalo de *longitudes de onda* (λ) de captura presentes en $c_p(x', y', \lambda)$.



- El resultado es una "función de imagen" que determina la cantidad de luz reflejada que es capturada en las coordenadas de la cámara (x', y').

$$f(x', y') = \int c_p(x', y', \lambda) V(\lambda) d\lambda \tag{1}$$

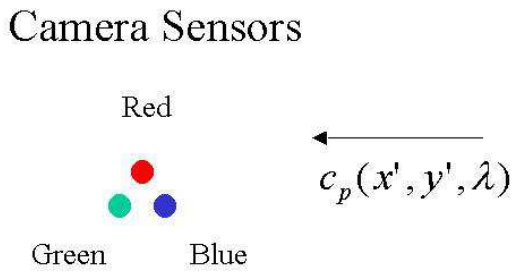
Ejemplo



Determinar las funciones de imagen de las funciones de sensibilidad mostradas arriba, suponiendo la misma escena:

1. Este es el más realista de los tres. La sensibilidad esta concentrada en una banda alrededor de λ_0.
$$f_1(x', y') = \int c_p(x', y', \lambda) V_1(\lambda) d\lambda$$
2. Este no es un dispositivo de captura realista, ya que presenta sensibilidad solamente en una longitud de onda λ_0 determinada por la función delta. Sin embargo, existen dispositivos que se acercan a dicho comportamiento "selectivo".
$$\begin{aligned} f_2(x', y') &= \int c_p(x', y', \lambda) V_2(\lambda) d\lambda = \int c_p(x', y', \lambda) \delta(\lambda - \lambda_0) d\lambda \\ &= c_p(x', y', \lambda_0) \end{aligned}$$
3. Esto es lo que sucede si se toma una foto sin quitar la cubierta de la lente de la cámara.
$$f_3(x', y') = \int c_p(x', y', \lambda) V_3(\lambda) d\lambda = \int c_p(x', y', \lambda)\ 0\ d\lambda = 0$$

Sensibilidad y Color



- Para una cámara que captura imágenes en color, imagine que tiene *tres sensores* en cada $(x',\ y')$ con funciones de sensibilidad sintonizadas a los colores o longitudes de onda rojo, verde y azul, las salidas son *tres* funciones de imagen:

$$f_{\text{R}}(x', y') = \int c_p(x', y', \lambda) V_{\text{R}}(\lambda) d\lambda$$

$$f_{\text{G}}(x', y') = \int c_p(x', y', \lambda) V_{\text{G}}(\lambda) d\lambda$$

$$f_{\text{B}}(x', y') = \int c_p(x', y', \lambda) V_{\text{B}}(\lambda) d\lambda$$

- Estas tres funciones de imagen pueden ser utilizadas por dispositivos de despliegue (*i.e.* monitor u ojo) para mostrar una imagen a "color".

Resumen

- La función de imagen $f_{\mathrm{C}}(x', y')$ $(C = R,\ G,\ B)$ se representa por:

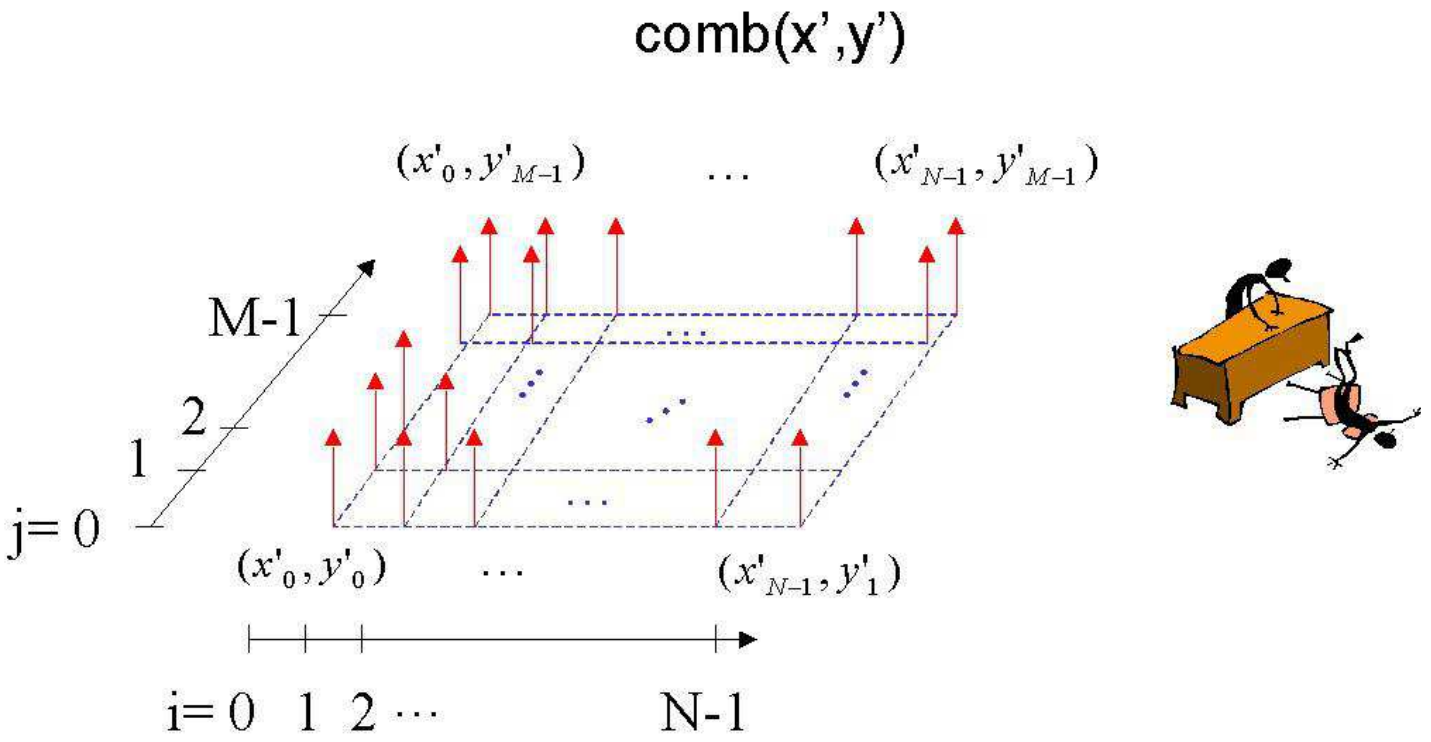
$$f_{\mathrm{C}}(x', y') = \int c_p(x', y', \lambda) V_{\mathrm{C}}(\lambda) d\lambda \tag{2}$$

- Esto es el resultado de:
  1. Luz incidente $E(x, y, z, \lambda)$ en el punto (x, y, z) de la escena,
  2. La función de reflectividad $r(x, y, z, \lambda)$ de este punto,
  3. La formación de la luz reflejada $c(x, y, z, \lambda) = E(x, y, z, \lambda) \times r(x, y, z, \lambda)$,
  4. La proyección de la luz reflejada $c(x, y, z, \lambda)$ del sistema dimensional de $tres$ coordenadas al sistema dimensional de dos coordenadas que forma $c_p(x', y', \lambda)$,
  5. Las funciones de sensibilidad de la cámara $V(\lambda)$.

Formación de Imágenes Digitales

- La función de imagen $f_{\mathrm{C}}(x', y')$ sigue siendo una función de $x' \in [x'_{min}, x'_{max}]$ y $y' \in [y'_{min}, y'_{max}]$ la cual varía en un continuo ("continuum") dado por los intervalos respectivos.
- Los valores que toma la función de imagen son números reales, los cuales, de nueva cuenta varian en un "continuum" o intervalo $f_{\mathrm{C}}(x', y') \in [f_{min}, f_{max}]$.
- Las computadoras digitales no pueden procesar parámetros/funciones que varían en un "continuum".
- Tenemos que *discretizar*:
  1. $x',\ y' \Rightarrow x'_i,\ y'_j \quad (i = 0, \ldots, N-1,\ j = 0, \ldots, M-1)$: Muestreo
  2. $f_{\mathrm{C}}(x'_i, y'_j) \Rightarrow \hat{f}_{\mathrm{C}}(x'_i, y'_j)$: Cuantización.

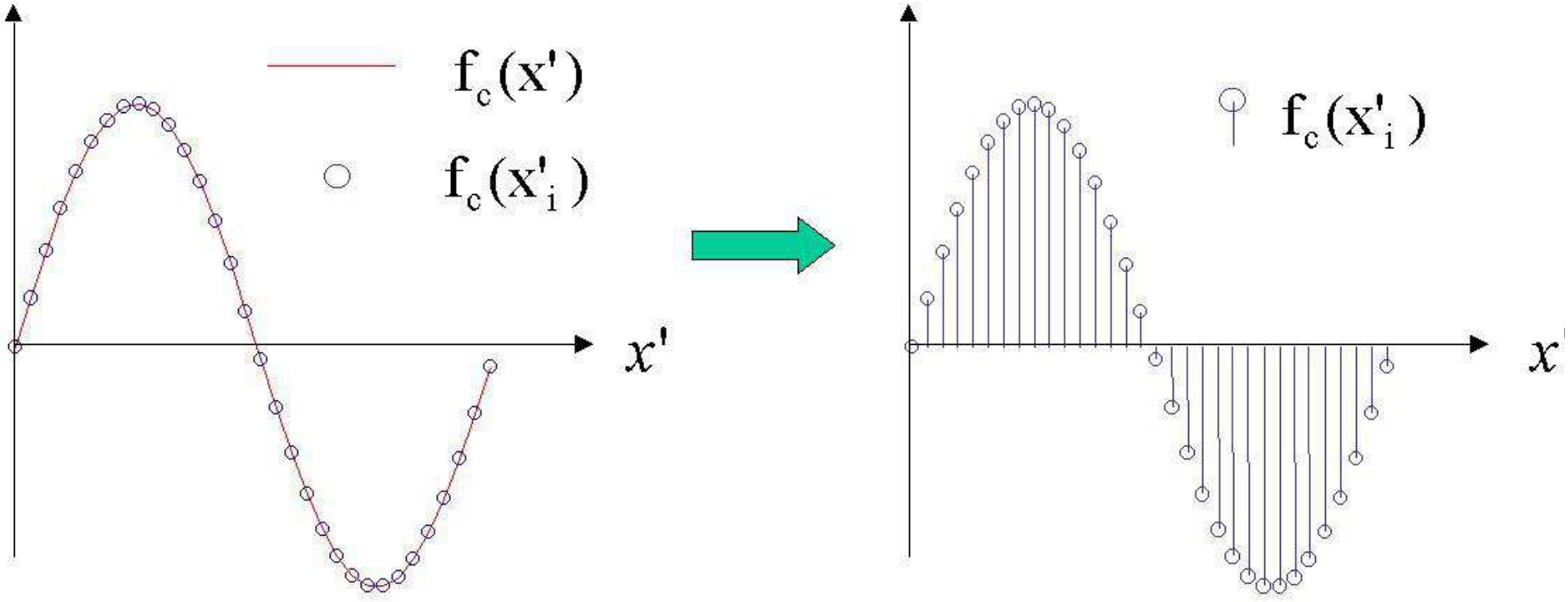
Muestreo



$$\texttt{comb}(x', y') = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \delta(x' - i\Delta_x, y' - j\Delta_y) \tag{3}$$

- El muestreo se obtiene al utilizar $f_{\mathrm{C}}(x', y') \times \texttt{comb}(x', y')$.

Ejemplo



Muestreo Ideal

- Posteriormente se verá que al utilizar $f_{\mathrm{C}}(x', y') \times \mathtt{comb}(x', y')$, es posible discretizar (x', y') y obtener una nueva "función de imagen" que esté definida en la rejilla discreta (x'_i, y'_j) $(i = 0, \ldots, N-1,\ j = 0, \ldots, M-1)$.

- Por ahora supongamos que de alguna manera obtenemos la función de imagen muestreada $f_{\mathrm{C}}(x'_i, y'_j)$.
- Para denotar esta discretización de ahora en adelante nos referiremos a $f_{\mathrm{C}}(x'_i, y'_j)$ como $f_{\mathrm{C}}(i, j)$.

Cuantización

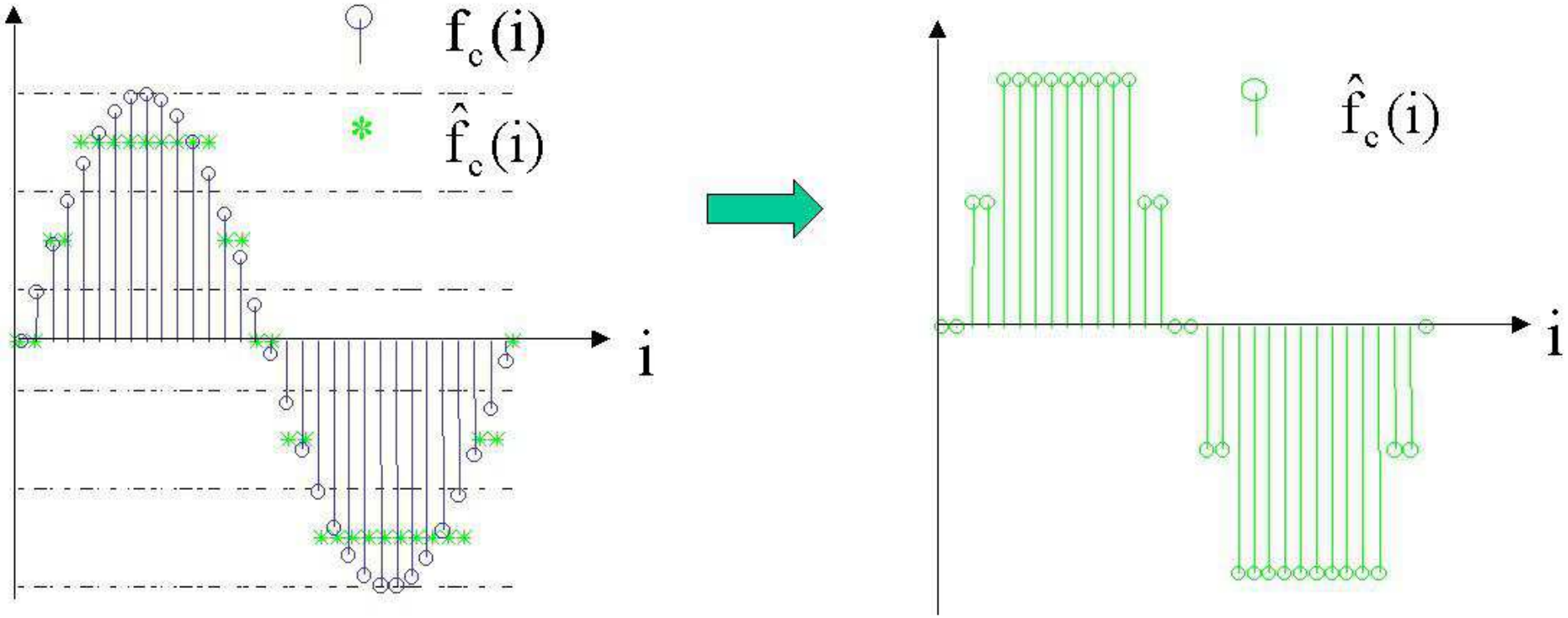
- $f_{\mathrm{C}}(i,j) \quad (i = 0,\ldots,N-1, \;\; j = 0,\ldots,M-1)$. Seguiremos ahora con el segundo paso de la discretización.
- $f_{\mathrm{C}}(i,j) \in [f_{min}, f_{max}], \;\; \forall (i,j)$.
- Se discretizan los valores $f_{\mathrm{C}}(i,j)$ a P *niveles* como sigue:
  Sea $\Delta_Q = \frac{f_{max}-f_{min}}{P}$.

$$\hat{f}_{\mathrm{C}}(i,j) = Q(f_{\mathrm{C}}(i,j)) \tag{4}$$

en donde

$$\begin{aligned} Q(f_{\mathrm{C}}(i,j)) \;\; &= \;\; (k+1/2)\Delta_Q + f_{min} \\ &\quad \textit{si y solo si } f_{\mathrm{C}}(i,j) \in [f_{min}+k\Delta_Q, f_{min}+(k+1)\Delta_Q) \\ &\quad \textit{si y solo si } f_{min}+k\Delta_Q \leq f_{\mathrm{C}}(i,j) < f_{min}+(k+1)\Delta_Q \\ &\quad \text{para } k = 0,\ldots,P-1 \end{aligned}$$

Ejemplo



Cuantización a P niveles

- Típicamente $P = 2^8 = 256$ y tenemos $\texttt{log}_2(P) = \texttt{log}_2(2^8) = 8$ **bits** de cuantización.
- Hemos alcanzado el segundo paso de discretización.
- De ahora en adelante se omitirán las referencias a f_{min}, f_{max} y a menos que se diga lo contrario se supondrá que las imágenes digitales originales están cuantizadas a 8 bits o 256 niveles.
- Para denotar esto nos referiremos a $\hat{f}_{\text{C}}(i,j)$ tomando los valores enteros de k en donde $0 \leq k \leq 255$, i.e., digamos que

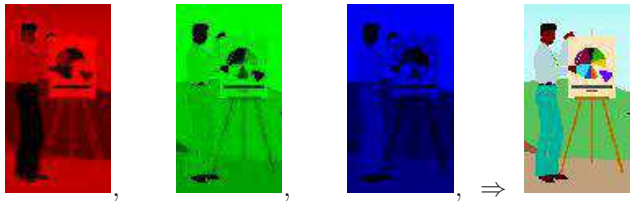
$$\hat{f}_{\text{C}}(i,j) \in \{0, \ldots, 255\} \tag{5}$$

Resumen

- "Hágase la luz" $\rightarrow$ luz incidente $\rightarrow$ reflectividad $\rightarrow$ luz reflejada $\rightarrow$ proyección $\rightarrow$ sensitividad $\rightarrow$ $f_{\mathrm{C}}(x', y')$.
- Muestreo: $f_{\mathrm{C}}(x', y') \rightarrow f_{\mathrm{C}}(i, j)$.
- Cuantización: $f_{\mathrm{C}}(i, j) \rightarrow \hat{f}_{\mathrm{C}}(i, j) \in \{0, \ldots, 255\}$.

Parametrización de Imágenes a Color (R,G,B)

$\hat{f}_{\mathrm{R}}(i,j)$, $\hat{f}_{\mathrm{G}}(i,j)$, $\hat{f}_{\mathrm{B}}(i,j)$ $\rightarrow$ imagen a color



- $\hat{f}_{\mathrm{R}}(i,j)$, $\hat{f}_{\mathrm{G}}(i,j)$ y $\hat{f}_{\mathrm{B}}(i,j)$ son llamados la parametrización (R,G,B) del "espacio de color" de la imagen de color.
- Hay otras parametrizaciones, cada una con sus propias ventajas y desventajas (ver Capítulo 3 del libro de texto [1]).

Imágenes en Escala de Grises

Imagen en "Escala de Grises" $\hat{f}_{\text{gray}}(i,j)$

- Una imagen en escala de grises o de luminancia puede ser considerada uno de los componentes de una parametrización diferente.
- Ventaja: captura la mayor parte de la "información de la imagen".
- El énfasis de esta clase se dará en el procesamiento en general. Por lo tanto, se trabajará principalmente con imágenes en escala de grises con el fin de evitar los diversos matices encontrados con diferentes parametrizaciones.

Imagenes como Matrices

- Recordando las operaciones de formación de imagen que se han discutido, note que la imágen $\hat{f}_{\text{gray}}(i,j)$ es una $matriz\ N \times M$ con elementos enteros en el intervalo $0,\ldots,255$.
- De ahora en adelante suprimiremos $\hat{(\,)}_{\text{gray}}$ y denotaremos una imagen como una matriz "$\mathbf{A}$" (o $\mathbf{B}$,..., etc.) con elementos $A(i,j) \in \{0,\ldots,255\}$ para $i = 0,\ldots,N-1$, $j = 0,\ldots,M-1$.
- Así que estaremos procesando matrices!
- Precaución: Algunos procesamientos que se harán, tomarán una imagen $\mathbf{A}$ con $A(i,j) \in \{0,\ldots,255\}$ pasándola a una nueva matriz $\mathbf{B}$ la cual podría no tener elementos enteros!

  En estos casos debemos $escalar$ y $redondear$ de manera adecuada los elementos de $\mathbf{B}$ a fin de desplegarla como una imagen.

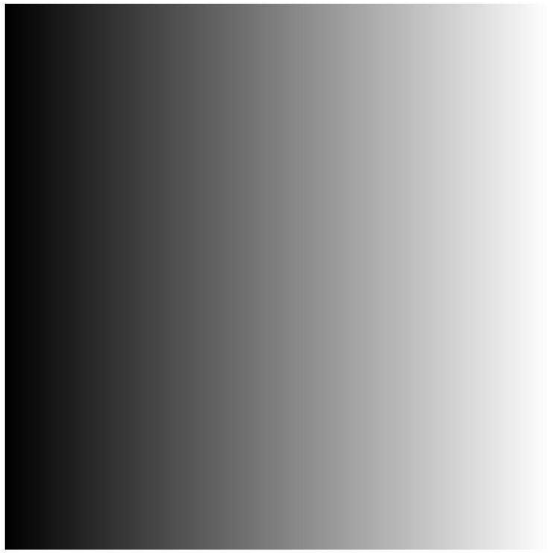
Matrices y Matlab

- El paquete de software Matlab es una herramienta que se especializa en realizar operaciones con matrices y su manejo es muy sencillo.
- A lo largo del curso utilizaremos Matlab para todo el procesamiento, ejemplos, tareas, etc. (lo cual no excluye la utilización de paquetes como: Python, OpenCV, Mathematica, etc.).

Ejemplo - I

- La imagen de una rampa (256×256):

$$\mathbf{A} = \left.\begin{bmatrix} 0 & 1 & 2 & \dots & 255 \\ 0 & 1 & 2 & \dots & 255 \\ \vdots & & & & \\ 0 & 1 & 2 & \dots & 255 \end{bmatrix}\right\} 256 \text{ renglones}$$

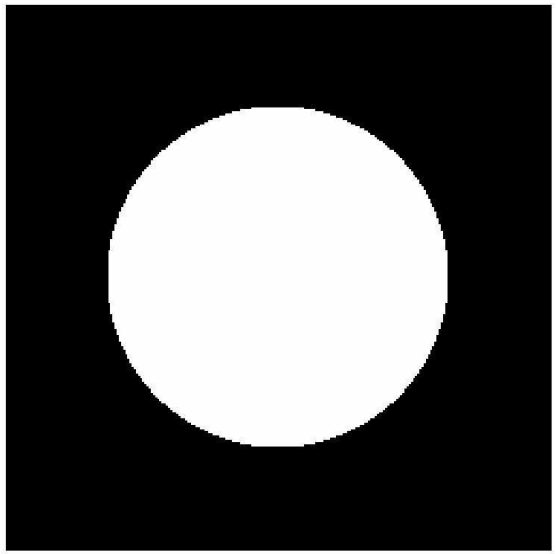


```
>> for i = 1 : 256
     for j = 1 : 256
       A(i, j) = j − 1;
       end
     end
>> image(A);
>> colormap(gray(256));
>> axis('image');
```

Ejemplo - II

- La imagen de un círculo (256×256) de radio 80 pixeles centrado en $(128, 128)$:

$$B(i,j) = \begin{cases} 255 & \text{si } \sqrt{(i-128)^2 + (j-128)^2} < 80 \\ 0 & \text{cualquier otro} \end{cases}$$

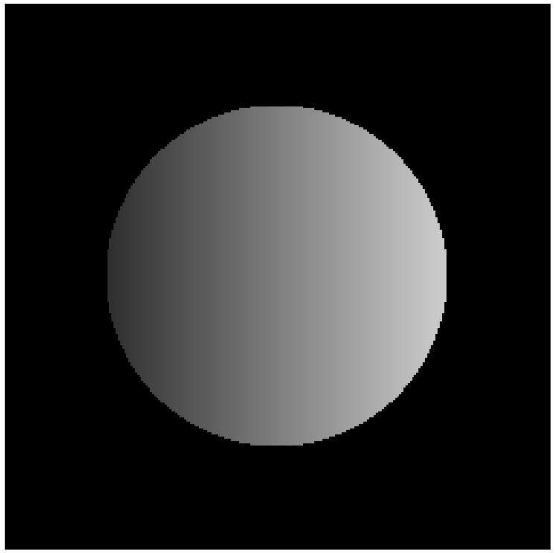


```
>> for i = 1 : 256
     for j = 1 : 256
       dist = ((i - 128)^2 + (j - 128)^2)^(.5);
       if (dist < 80)
         B(i, j) = 255;
       else
         B(i, j) = 0;
         end
       end
     end
>> image(B);
>> colormap(gray(256));
>> axis('image');
```

Ejemplo - III

- La imagen de un círculo "gradual" (256×256):

$$C(i,j) = A(i,j) \times B(i,j)/255$$



```
>> for i = 1 : 256
     for j = 1 : 256
       C(i,j) = A(i,j) * B(i,j)/255;
       end
     end
>> image(C);
>> colormap(gray(256));
>> axis('image');
```

Tarea I

1. Si es necesario estudie algunas notas introductorias sobre Matlab.
2. Tome una imagen de usted mismo *"selfie"*. Asegurese de que esté en escala de grises de 8 bits y *aprenda* como leer su imagen en Matlab como una matriz.
3. Despliegue su imagen y obtenga una impresión. Pruebe "`>> help print`" para obtener ayuda de instrucciones en Matlab.

[1] B. K. P. Horn, *Robot Vision.* Cambridge, MA: MIT Press, 1986.

[2] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

Imágenes como Matrices

- Matriz $(N \times M)$ de una imagen:

$$\mathbf{A} = \left[\begin{array}{ccccc} A(0,0) & A(0,1) & A(0,2) & \dots & A(0,M-1) \\ A(1,0) & A(1,1) & A(1,2) & \dots & A(1,M-1) \\ \vdots & & & & \\ A(N-1,0) & A(N-1,1) & A(N-1,2) & \dots & A(N-1,M-1) \end{array}\right] \Bigg\} N \text{ renglones}$$

- $A(i,j) \in \{0,1,\dots,255\}$.
- $A(i,j)$:
  - "caso Matriz:" El elemento de la matriz (i,j) con valor $A(i,j)$.
  - "caso Imagen:" El pixel (i,j) con valor $A(i,j)$.
  - Se utilizan ambas terminologías.

Procesamiento Simple - Transpuesta

- La imagen transpuesta $\mathbf{B}$ $(M \times N)$ de $\mathbf{A}$ $(N \times M)$ puede ser obtenida con $B(j,i) = A(i,j)$
  $(i = 0, \ldots, N-1,\ \ j = 0, \ldots, M-1)$.

A

B

```
>>  for i = 1 : 512
        for j = 1 : 512
            B(j,i) = A(i,j);
            end
        end
```

O

```
>>  B = A';
```

Procesamiento Simple - Giro Vertical

- La imagen girada verticalmente $\mathbf{B}$ $(N \times M)$ de $\mathbf{A}$ $(N \times M)$ puede ser obtenida con $B(i, M-1-j) = A(i,j)$ $(i = 0, \ldots, N-1,\ \ j = 0, \ldots, M-1)$.

A

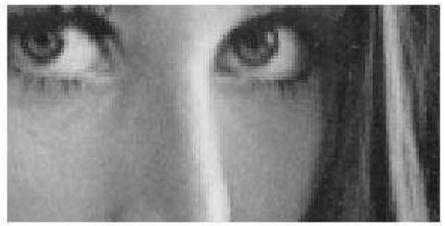
B

```
>> clear B;
>> for i = 1 : 512
     for j = 1 : 512
       B(i, 512 + 1 - j) = A(i, j);
       end
     end
```

Procesamiento Simple - Recortado

- La imagen recortada **B** $(N_1 \times N_2)$ de **A** $(N \times M)$, iniciando en (n_1, n_2), puede ser obtenida con $B(k,l) = A(n_1 + k, n_2 + l)$ $(k = 0, \ldots, N_1 - 1,\ l = 0, \ldots, N_2 - 1)$.

A



B

```
>> clear B;
>> for k = 0 : 64 - 1
     for l = 0 : 128 - 1
       B(k + 1, l + 1) = A(255 + k + 1, 255 + l + 1);   % n1=n2=255 N1=64,N2=128
       end
     end
```

Recortado como Función de Matlab

function $[B]$ =mycrop$(A, n1, n2, N1, N2)$
% mycrop.m
% $[B]$ =mycrop$(A, n1, n2, N1, N2)$
% crops the image A from location $n1, n2$
% with size $N1, N2$

for $k = 0 : N1 - 1$
    for $l = 0 : N2 - 1$
        $B(k + 1, l + 1) = A(n1 + k + 1, n2 + l + 1);$
        end
    end

---

function $[B]$ =mycrop$(A, n1, n2, N1, N2)$
% mycrop.m
% $[B]$ =mycrop$(A, n1, n2, N1, N2)$
% crops the image A from location $n1, n2$
% with size $N1, N2$

$B(1 : N1, 1 : N2) = A(n1 + 1 : n1 + N1, n2 + 1 : n2 + N2);$

>> help mycrop
>> B =mycrop$(A, 255, 255, 64, 128);$

Estadística Simple de Imagen - Media de Muestra y Varianza de Muestra

- La media de muestra (m_A) de una imagen $\mathbf{A}$ $(N \times M)$:

$$m_A = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} A(i,j)}{NM} \tag{6}$$

- La varianza de muestra (σ_A^2) de $\mathbf{A}$:

$$\sigma_A^2 = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (A(i,j) - m_A)^2}{NM} \tag{7}$$

- La desviación estandar de muestra, $\sigma_A = \sqrt{\sigma_A^2}$.

Estadística Simple de Imagen - Histograma

Considere el conjunto S y defina $\#S$ como la cardinalidad del conjunto, i.e., $\#S$ es el número de elementos de S.

- El histograma $h_A(l)$ $(l = 0, \ldots, 255)$ de la imagen $\mathbf{A}$ está definido como:

$$h_A(l) \;=\; \#\{(i,j) \mid A(i,j) = l,\;\; i = 0, \ldots, N-1,\;\; j = 0, \ldots, M-1\} \qquad (8)$$

- Note que:

$$\sum_{l=0}^{255} h_A(l) = \text{Número de pixeles en } \mathbf{A} \qquad (9)$$

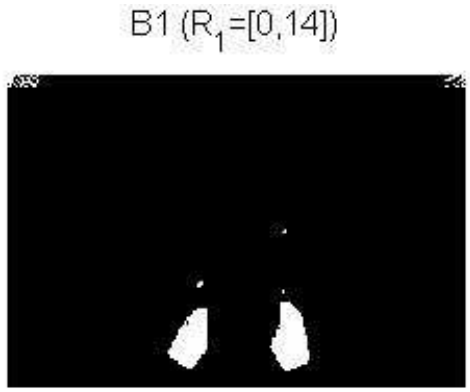
Cálculo del Histograma

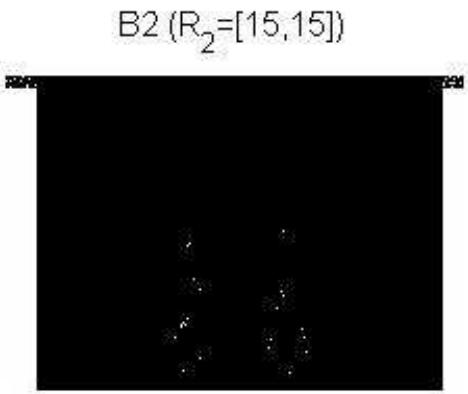
```
>> h=zeros(256,1);
>> for l = 0 : 255
     for i = 1 : N
       for j = 1 : M
         if (A(i, j) == l)
           h(l + 1) = h(l + 1) + 1;
           end
         end
       end
     end
>> bar(0:255,h);
```

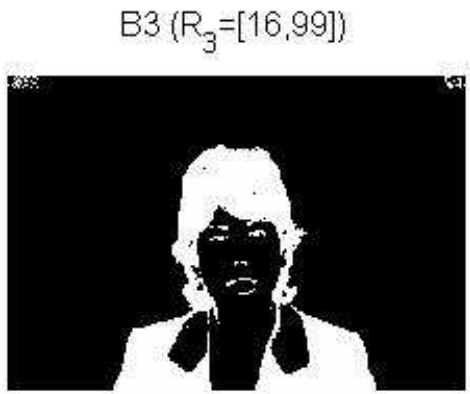
O

```
>> h=zeros(256,1);
>> for l = 0 : 255
       h(l + 1)=sum(sum(A == l));
       end
>> bar(0:255,h);
```

Ejemplo I

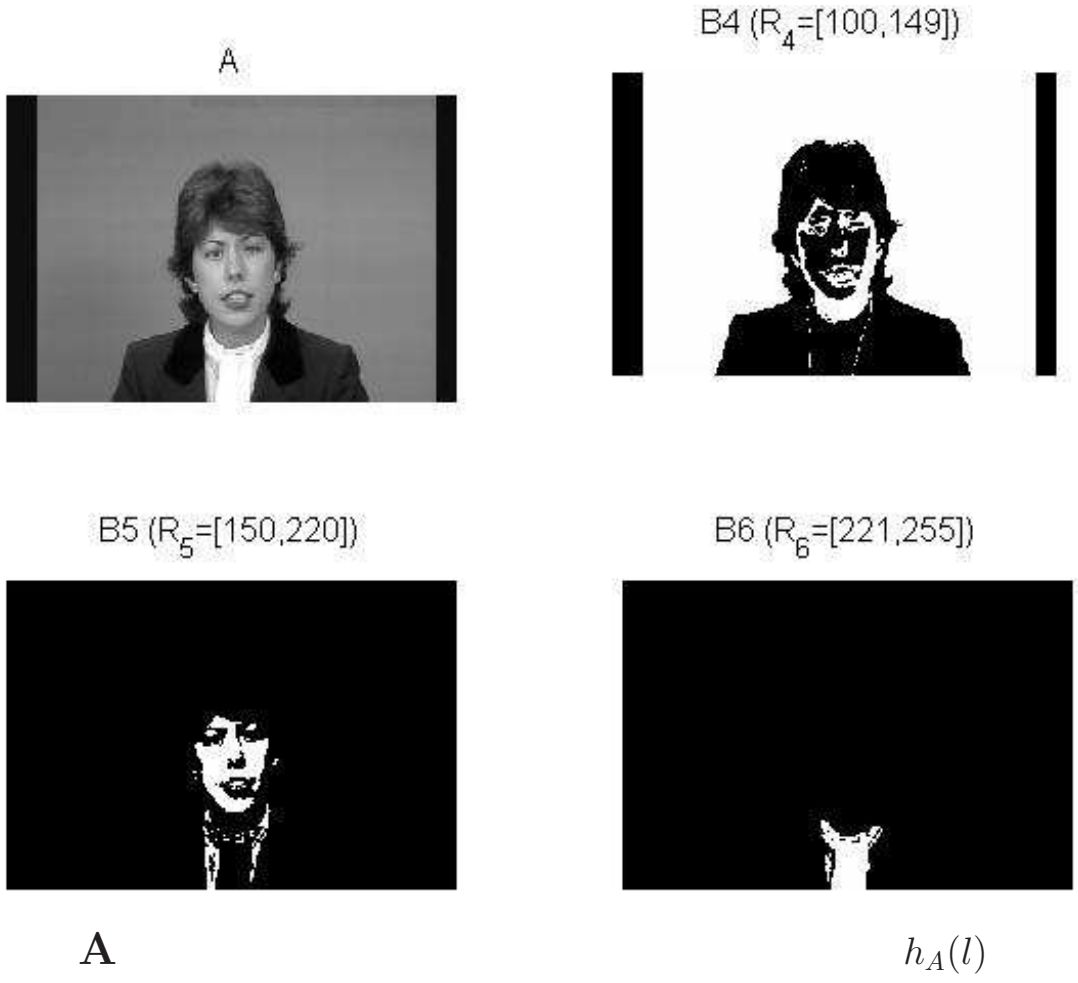






A $h_A(l)$

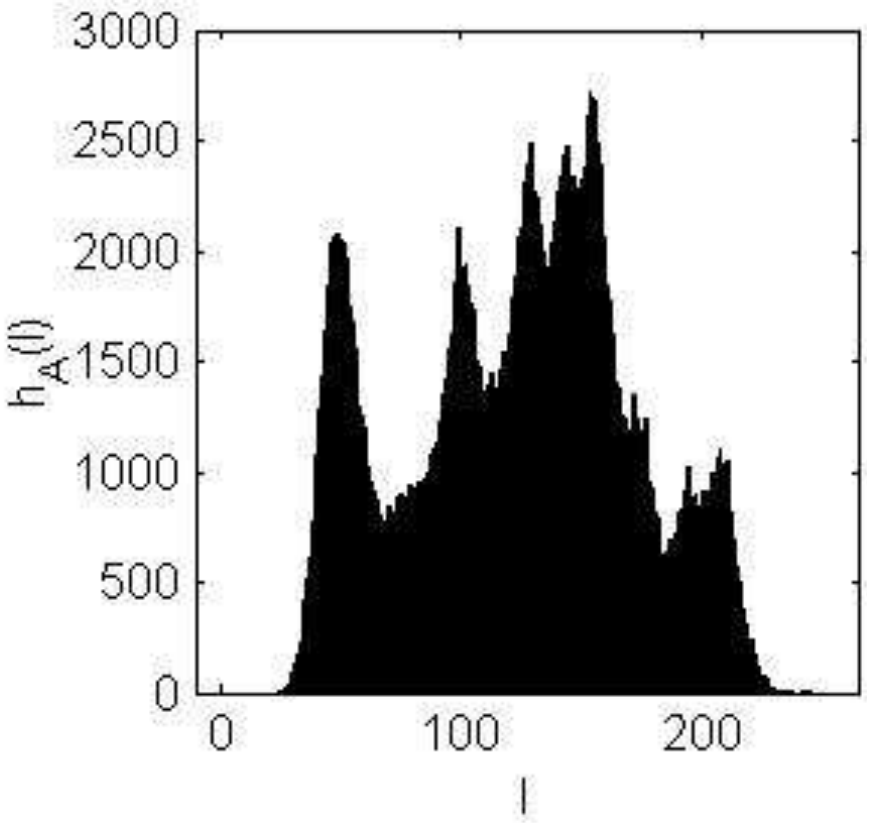
Ejemplo II



A $h_A(l)$

Ejemplo III

A



$h_A(l)$

Procesamiento de Puntos

- Ahora se utilizará una "función" $g(l)$ $(l = 0, \ldots, 255)$ para generar una nueva imagen $\mathbf{B}$ a partir de la imagen dada $\mathbf{A}$ vía:

$$B(i,j) = g(A(i,j)), \quad i = 0, \ldots, N-1, \; j = 0, \ldots, M-1 \tag{10}$$

- La función $g(l)$ opera en cada pixel de la imagen o en cada punto de la imagen de forma independiente.
- En general la imagen resultante $g(A(i,j))$ puede no ser una matriz de imagen, i.e., puede darse el caso de que $g(A(i,j)) \notin \{0, \ldots, 255\}$ para algunos (i,j).
- Por tanto, debemos asegurarnos de obtener una imagen $\mathbf{B}$ que sea una matriz de imagen.
- Los histogramas $h_A(l)$ y $h_B(l)$ jugarán papeles importantes.

Función de identidad de punto

- Haciendo $g(l) = l \ (l = 0, \ldots, 255)$.

$$\begin{aligned} B(i,j) &= g(A(i,j)), \quad i = 0, \ldots, N-1, \ j = 0, \ldots, M-1 \\ &= A(i,j) \end{aligned}$$

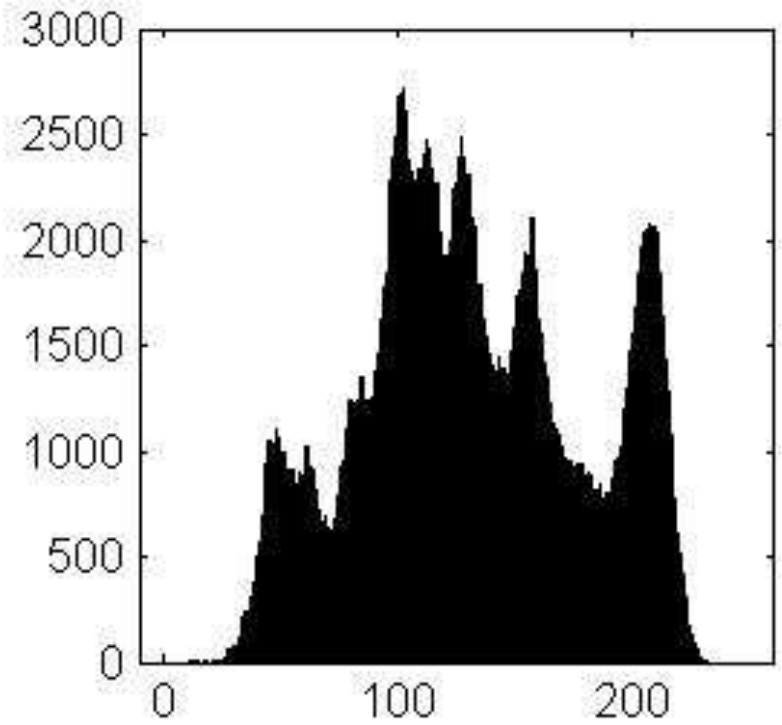
- En este caso $g(A(i,j)) \in \{0, \ldots, 255\}$ así que no se requiere más procesamiento para asegurar que $\mathbf{B}$ es una matriz de imagen.
- Note además que $\mathbf{B} = \mathbf{A}$ y por tanto $h_B(l) = h_A(l)$.

Negativo Digital

- Haciendo $g(l) = 255 - l \ \ (l = 0, \ldots, 255)$.

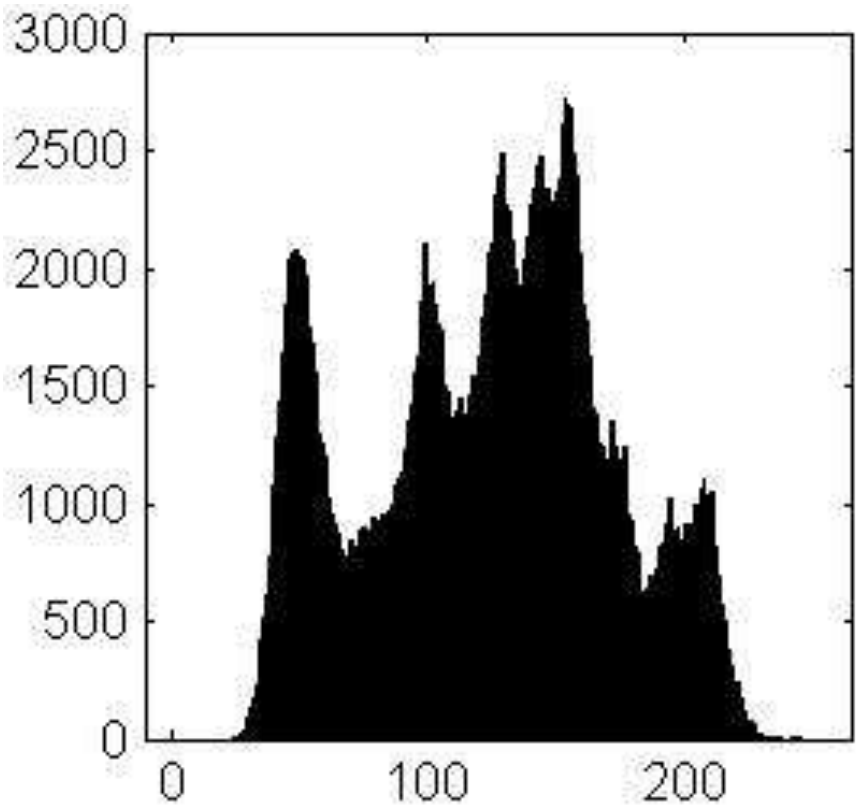
$$\begin{aligned} B(i,j) &= g(A(i,j)), \quad i = 0, \ldots, N-1, \ j = 0, \ldots, M-1 \\ &= 255 - A(i,j) \end{aligned}$$

B

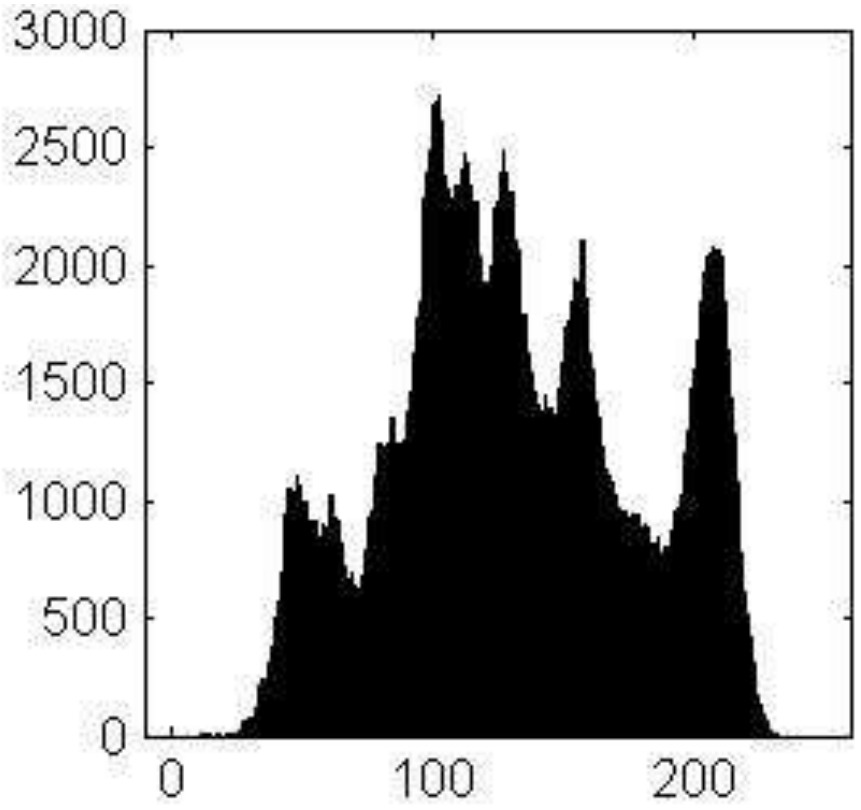


$h_B(l)$

Negativo Digital Cont.



$h_A(l)$



$h_B(l)$

- En este caso se puede observar fácilmente que $h_B(255 - l) = h_A(l)$ o $h_B(g(l)) = h_A(l)$.

Calculando el Histograma de Imagenes Procesadas por Punto

- Para una función de punto dada $g(l)$, "la función inversa de punto" $g^{-1}(k)$ puede no existir.
- Así que en general $h_B(g(l)) \neq h_A(l)$.
- Haciendo $S_{g^{-1}(k)} = \{l \mid g(l) = k,\ l = 0, \ldots, 255\}\ (k = 0, \ldots,\ 255)$.
- Entonces:

$$h_B(k) = \sum_{l \in S_{g^{-1}(k)}} h_A(l), \qquad k = 0, \ldots, 255 \tag{11}$$

- Se *debe* aprender como calcular y trazar $h_B(l)$ dado $h_A(l)$ y la función de punto $g(l)$.

Raíz cuadrada de la función de punto

- Haciendo $g(l) = \sqrt{l}$ $(l = 0, \ldots, 255)$.

$$\begin{aligned} B(i,j) &= g(A(i,j)), \quad i = 0, \ldots, N-1, \ j = 0, \ldots, M-1 \\ &= \sqrt{A(i,j)} \end{aligned}$$

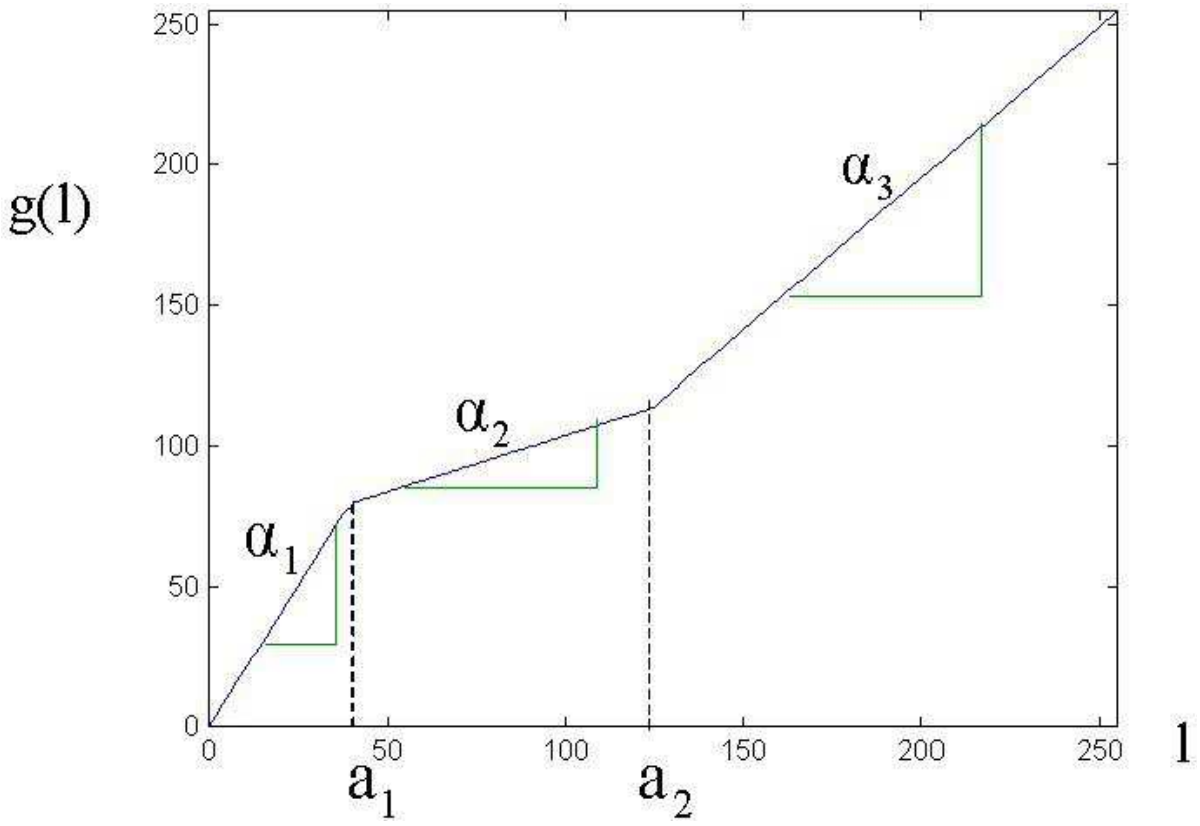
- En este caso $g(A(i,j)) \notin \{0, \ldots, 255\}$ y *debemos asegurar que* $\mathbf{B}$ sea una matriz de imagen para su posterior procesamiento.
- Podemos tratar de "redondear" $g(A(i,j))$ a enteros para definir una nueva función de punto $g_2(l) = \text{round}(g(l))$:

$$\begin{aligned} B(i,j) &= g_2(A(i,j)), \quad i = 0, \ldots, N-1, \ j = 0, \ldots, M-1 \\ &= \text{round}(\sqrt{A(i,j)}) \quad (>> \ B = \text{round}(A.\hat{}(.5));) \end{aligned}$$

  pero esto no dará exactamente lo que queremos, como se verá posteriormente.
- Se regresará a este ejemplo cuando se estudie la Expansión de Contraste.

Expansión de Contraste

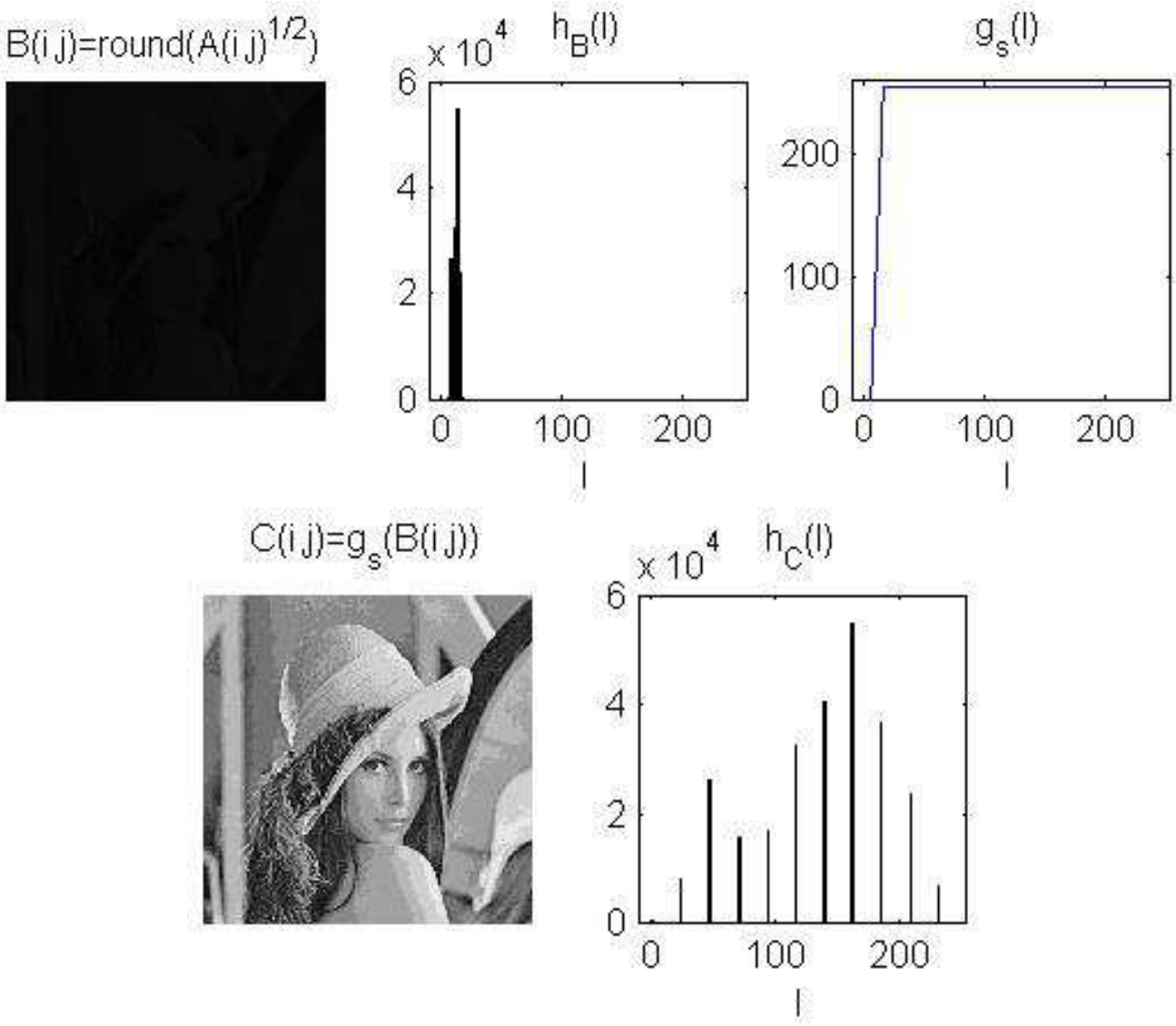
$$g(l) = \begin{cases} \alpha_1 l, & 0 \leq l < a_1 \\ \alpha_2(l - a_1) + \alpha_1 a_1, & a_1 \leq l < a_2 \\ \alpha_3(l - a_2) + (\alpha_2(a_2 - a_1) + \alpha_1 a_1), & a_2 \leq l \leq 255 \end{cases} \quad (12)$$



$\alpha_i > 1 \Rightarrow$ Intervalo de Expansión

$\alpha_i < 1 \Rightarrow$ Intervalo de Compresión

Ejemplo - Raíz cuadrada



Funciones de Punto "Continuas" por tramos Lineales

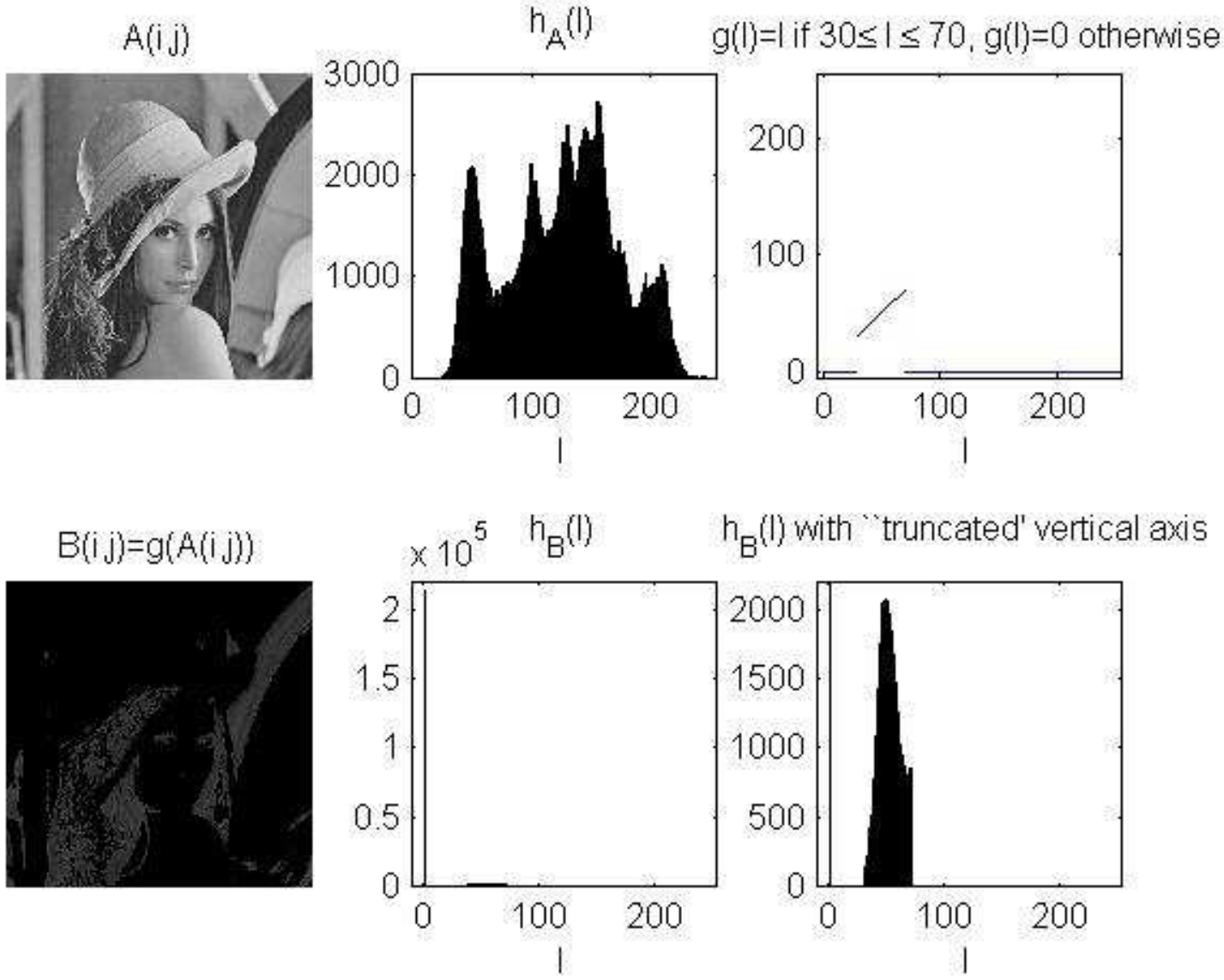
- Considerando que $\alpha_0 = a_0 = 0$.

$$g(l) = \begin{cases} \alpha_1 l, & 0 \leq l < a_1 \\ \alpha_2(l - a_1) + \alpha_1 a_1, & a_1 \leq l < a_2 \\ \vdots & \\ \alpha_i(l - a_{i-1}) + (\sum_{j=1}^{i-1} \alpha_j(a_j - a_{j-1})), & a_{i-1} \leq l < a_i \\ \vdots & \end{cases} \tag{13}$$

- $|\alpha_i| < 1 \Rightarrow$ Intervalo de Compresión.
- $|\alpha_i| > 1 \Rightarrow$ Intervalo de Expansión.
- Asumiendo que $0 \leq g(l) \leq 255$ $(l = 0, \ldots, 255)$, afecta la función de punto al incorporar la función $\text{round}(\ldots)$ cuando es necesario, i.e., $g_2(l) = \text{round}(g(l))$:

$$B(i,j) = g_2(A(i,j));$$

Una función de Punto "Discontinua"



Normalizando una Imagen

- Considere $mx_A = \text{máx}_{(i,j)}\, A(i,j)$ y $mn_A = \text{mín}_{(i,j)}\, A(i,j)$.

  Estos pueden generarse en Matlab vía:

  >> mx=max(max(A));

  >> mn=min(min(A));

- La *función especial de normalización de punto* para **A** se define como:

$$g_s^A(l) = \text{round}\left(\frac{l - mn_A}{mx_A - mn_A} \times 255\right) \tag{14}$$

- Convención:

  A $\Rightarrow$ procesamiento $\Rightarrow$ **B** $\Rightarrow$ "normaliza **B**" $\Rightarrow$ $C(i,j) = g_s^B(B(i,j))$.

Intervalo Expansión/Compresión

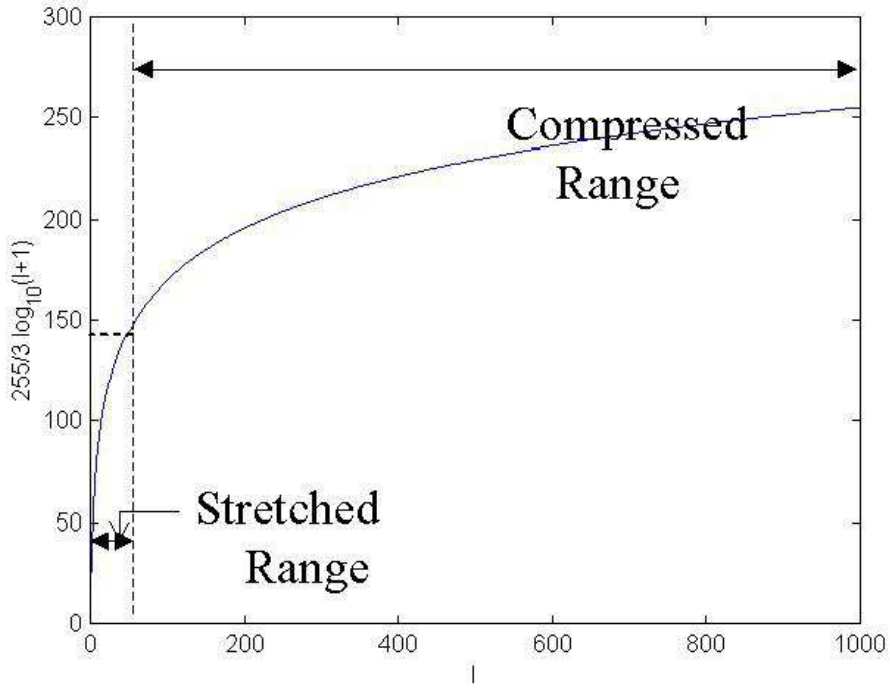
- En algunos casos se desea ver una matriz (matriz de imagen o alguna otra) que contiene un amplio intervalo de valores.
- Considerando

$$B(i,j) = \begin{cases} A(0,0) + 10^6 & i = j = 0 \\ A(i,j) & \text{cualquier otro} \end{cases}$$

  en donde $\mathbf{A}$ es una matriz de imagen $(A(i,j) \in \{0,\ldots,255\})$.
- Normalizando $\mathbf{B}$ empleando $C(i,j) = g_s^B(B(i,j))$ se mostrará una imagen $(\mathbf{C})$ completamente negra excepto para el pixel $(0,0)$, el cual tendrá un valor 255.
- Esto pierde toda la información en $\mathbf{C}$ referente a $\mathbf{A}$.
- Las siguientes dos diapositivas consideran un ejemplo que aborda dicho problema en el procesamiento de punto. La Función de Punto resultante algunas veces es llamada función de enfasis/de-enfasis.

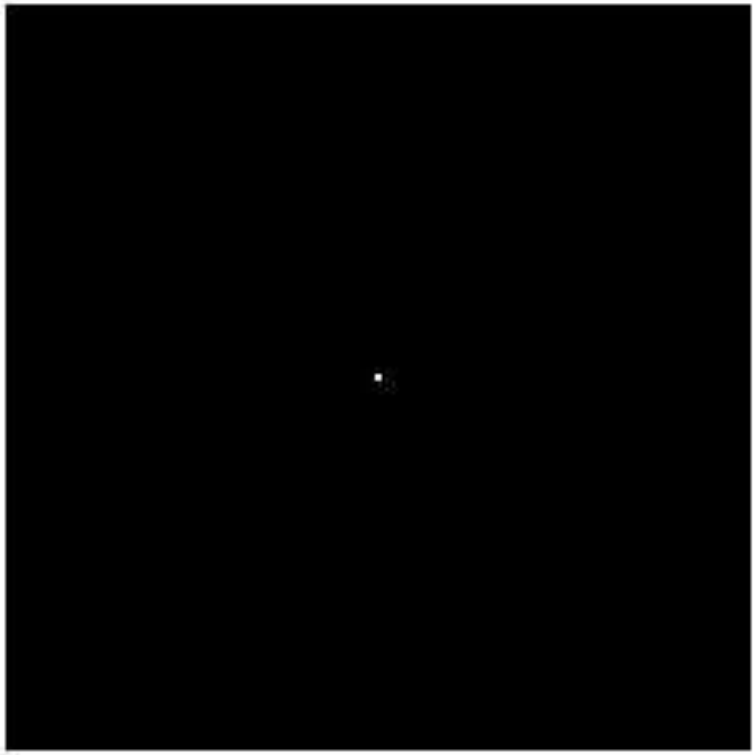
Ejemplo

- Un buen ejemplo del problema de "amplio intervalo" sucede cuando se desea observar la *Transformada Discreta de Fourier (DFT) en 2-D* de una imagen.
- $>> F = \text{round}(\text{fftshift}(\text{abs}(\text{fft2}(A))));$
- Considerando $B(i,j) = log_{10}(F(i,j)+1)$ y $C(i,j) = g_s^F(F(i,j))$, $D(i,j) = g_s^B(B(i,j))$.

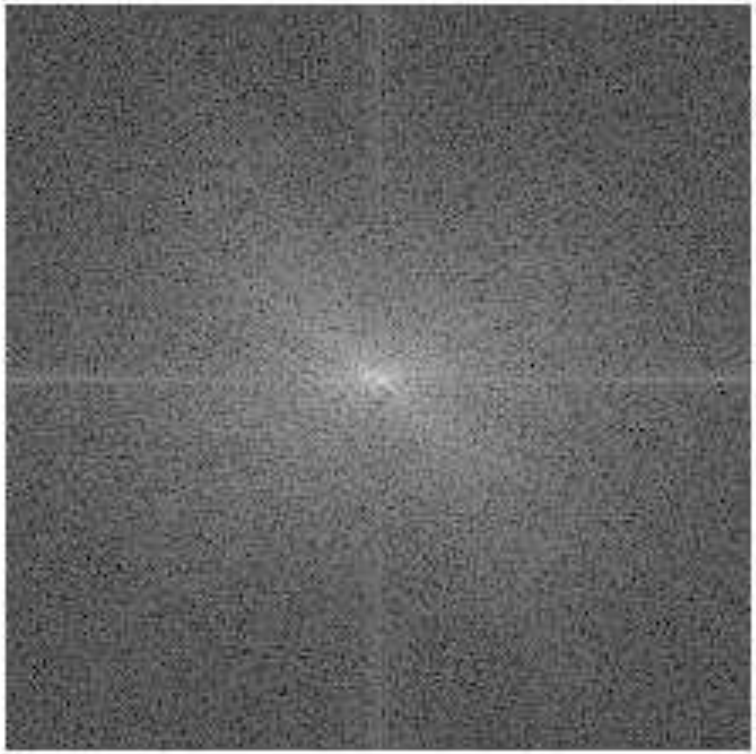


Ejemplo - cont.

C

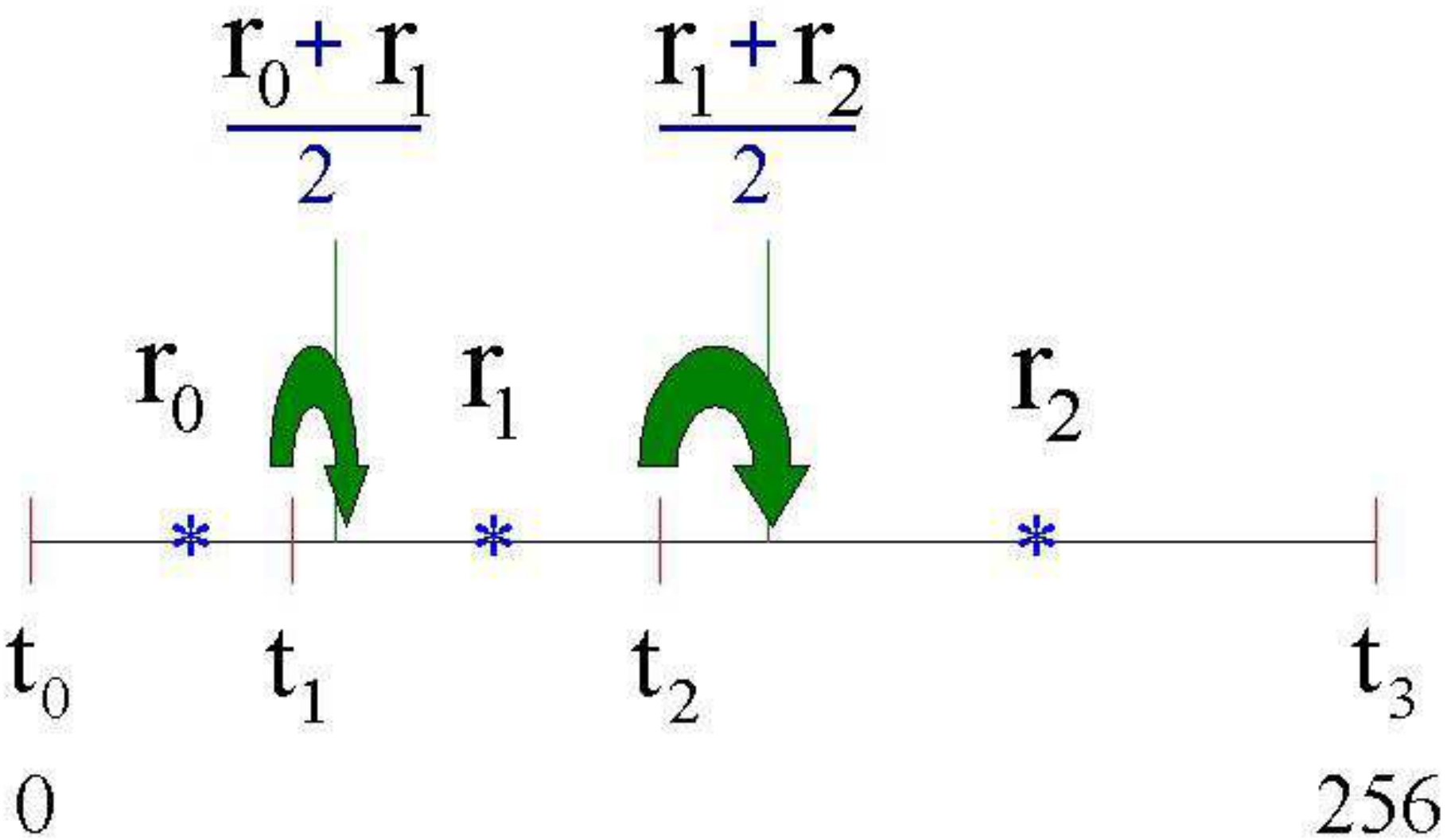


D



- Un intervalo específico de expansión/compresión puede generar aún mejores resultados. Esta es una herramienta efectiva para visualizar cosas rápidamente. Diferentes problemas se pueden abordar en el mismo sentido utilizando funciones de punto basadas en $log_{10}(\ldots)$, $log_{10}(log_{10}(\ldots))$, $e^{\cdots}$, etc.

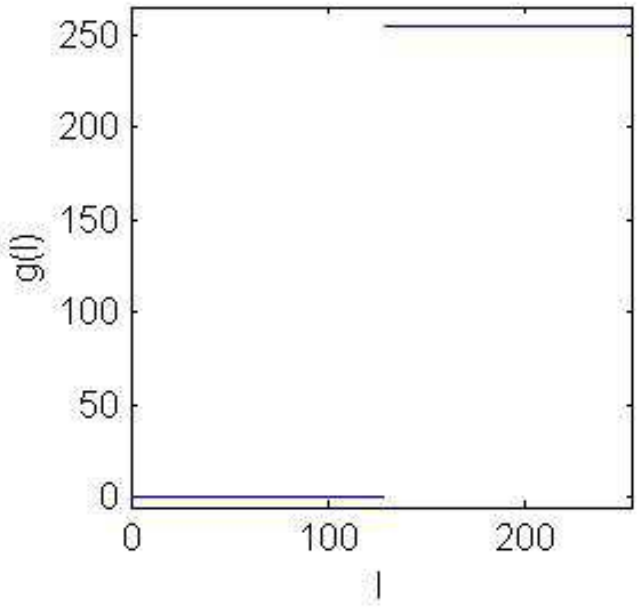
“Rebanado”



Umbralización

- Umbralización binaria $(T = 128)$

$$B(i,j) = \begin{cases} 0 & A(i,j) \leq T \\ 255 & A(i,j) > T \end{cases}$$



- Posteriormente veremos umbralizaciones más sofisticadas.

Resumen

- En esta parte se aprendieron técnicas de procesamiento sobre estructuras simples, tales como transposición, rotación y recorte.
- Se vió estadística simple, tal como media y varianza de muestra.
- Más importante aún fue el tema de los histogramas.
  - Los histogramas nos indican como están "distribuidos" los valores de pixeles individuales en una imagen.
  - Una imagen puede tener el mismo histograma que otra imagen.
- Se mostraron varias técnicas de procesamiento de punto. Estas serán de gran utilidad a medida que se tenga más experiencia en el manejo de imágenes. Es necesario leer el libro de texto [1] (Capítulo 7, pp. 233-241) para conocer más ejemplos.
- Se mostró como escribir funciones en Matlab, "scripts" para ejecución más rápida, etc.

Reglas para entrega de tareas

- Los textos de las tareas y reportes se escribirán en LaTeX (*.tex) y se utilizará como manejador de base de datos el paquete Mendeley. Los códigos de los ejercicios deberán estar en formatos: Matlab, Python, OpenCV, etc. (según sea solicitado). Las Figuras se editarán con el paquete Inkscape y se escribirán en formato SVG (Scalable Vector Graphics).
- Todos los los archivos fuente (individuales) correspondientes a tareas y reportes se enviarán a más tardar antes del inicio de la clase posterior a la asignación del trabajo (o fecha indicada por el Profesor) al siguiente correo electrónico: emartine@uabc.edu.mx.
- Todas las gráficas deben incluír etiquetas para cada eje (y barra) y agregar un título *descriptivo*.
- Sea claro y conciso, además incluya siempre la imagen original para comparar las imágenes procesadas.
- Todos los "scripts" desarrollados deber ser probados y deberán "correr" en otras máquinas (principalmente la del Profesor). Incluya comentarios descriptivos en los procedimientos de cada "script" (información de ayuda).
- Para mostrar las imágenes utilice los mandos *image*, *colormap(gray(256))*, y *axis('image')* . **No** utilice el mando *imagesc* en Matlab para realizar normalizaciones automáticas.
- Recuerde que para observar mejor las notas y trabajo en Matlab se debe ajustar la pantalla a 24 o 32 bits de profundidad ("depth") de color. De otra forma podría observar "características" artificiales.

Tarea II

1. Gire su imagen horizontal y diagonalmente. Imprima los resultados.
2. Recorte una porción de su imagen que muestre principalmente su cabeza. Imprima el resultado y los parámetros utilizados.
3. Calcule la media, varianza e histograma de su imagen. Muestre el histograma como una gráfica de barras al lado de su imagen. (utilice el mando *subplot*). Indique el tamaño de su imagen.
4. Calcule la raíz cuadrada de su imagen. "Redondeela" utilizando la función **round**. Muestre el resultado y su histograma. Ahora normalice la imagen "redondeada". Muestre el resultado y su histograma. Compare lo obtenido con la imagen sin procesar y describa brevemente.
5. Calcule la raíz cuadrada de su imagen. Normalice el resultado e imprimalo junto a su histograma. Compare lo obtenido con la salida de 4 y la imagen original. Describa brevemente. ¿Qué intervalos de valores de pixel se expandieron/comprimieron?
6. Calcule $B(i,j) = log_{10}(A(i,j)+1)$ en donde **A** es su imagen. **B** es una matríz general y no una matriz de imagen. Genere una matriz de imagen a partir de **B** al normalizarla. Muestre la imagen normalizada junto a la imagen original. Describa brevemente los cambios. ¿Hay algo que se pueda ver mejor? ¿Qué intervalos de valores de pixel se expandieron/comprimieron?
7. Para una imagen **A** con histograma $h_A(l)$, trace el histograma $h_B(l)$ para **(a)** $B(i,j) = g_1(A(i,j))$ y **(b)** $B(i,j) = g_2(A(i,j))$, en donde $h_A(l),\ g_1,\ g_2$ se muestran en la Figura 1.

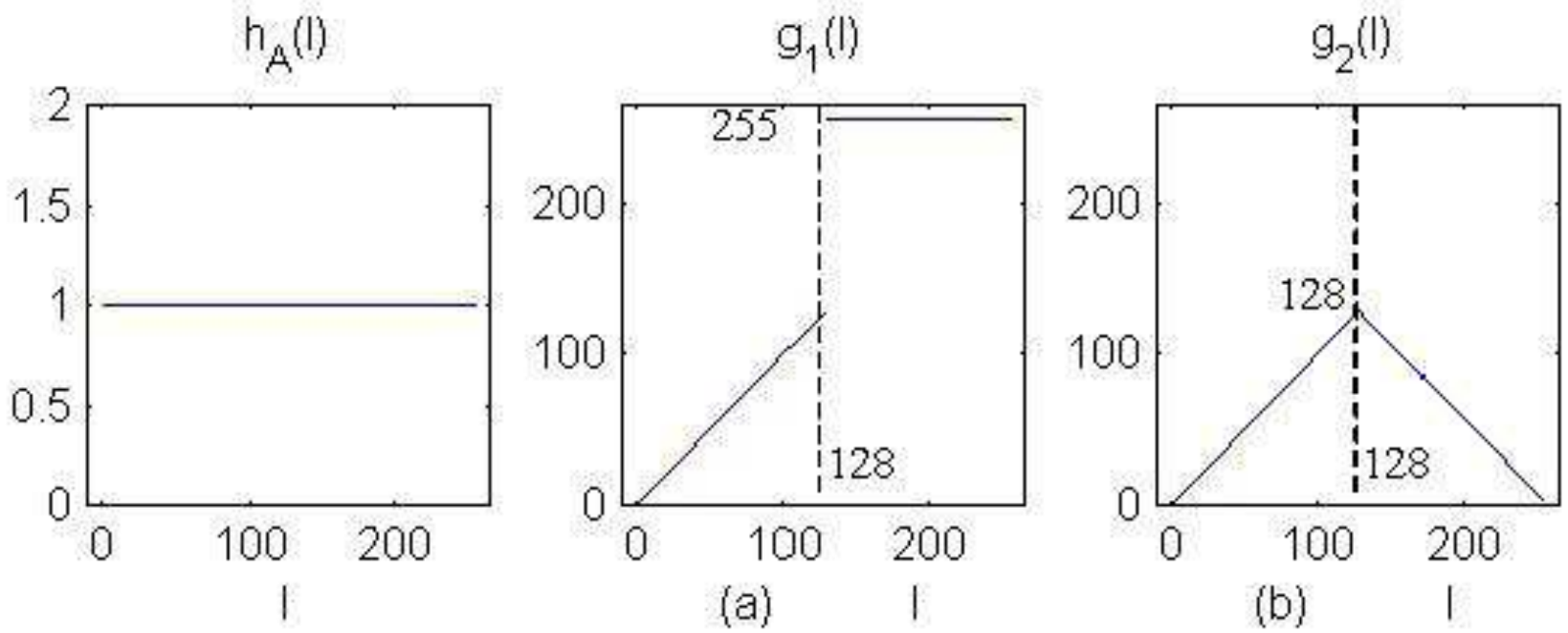
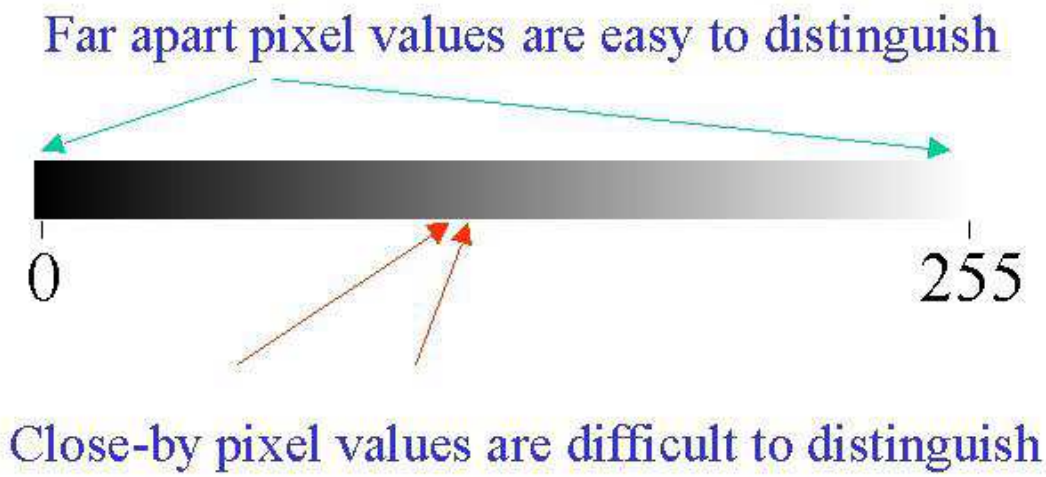


Figura 1: Pregunta 7 de la tarea.

Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

Intervalo Dinámico, Visibilidad y Mejoramiento de Contraste



- El Mejoramiento de Contraste de las Funciones de Punto vistas previamente expande el intervalo dinámico ocupado por ciertos valores de pixel "interesantes" en la imagen de entrada.
- Estos valores de pixel en la imagen de entrada pueden ser difíciles de distinguir y la meta del mejoramiento de contraste es hacerlos "más visibles" en la imagen de salida.
- No olvidar que se cuenta con un intervalo dinámico limitado $(0 - 255)$.

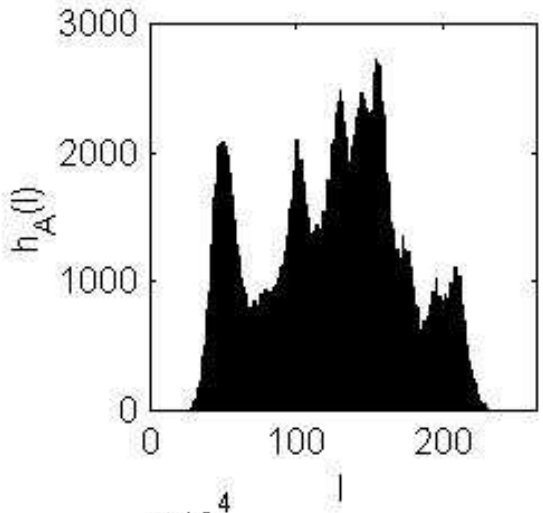
Funciones de Punto e Histogramas

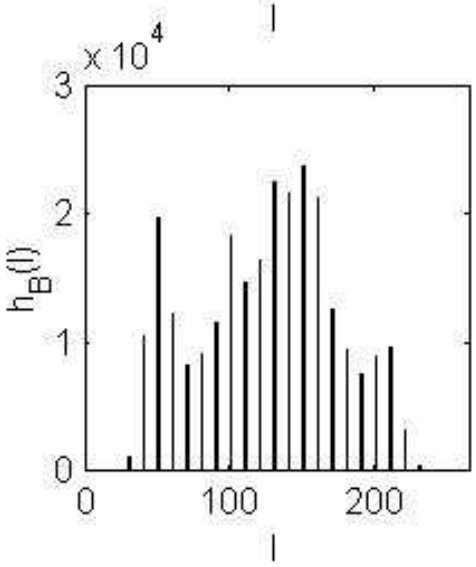
- En general una operación/función de punto $B(i,j) = g(A(i,j))$ da como resultado un nuevo histograma $h_B(l)$ para la imagen de salida que es diferente de $h_A(l)$.
- La relación entre $h_B(l)$ y $h_A(l)$ puede no ser tan evidente como ya se mencionó en la Clase 2.
- Se *debe* aprender como calcular $h_B(l)$ dada $h_A(l)$ y la función de punto $g(l)$:
  - Exactamente: Usualmente escribiendo un script en Matlab que calcule $h_B(l)$ a partir de $h_A(l)$ y $g(l)$.
  - Aproximadamente: Graficando $h_B(l)$ dadas los gráficas de $h_A(l)$ y $g(l)$.

Efecto "Inesperado" de algunas Funciones de Punto

A

B(i,j)=10 round(A(i,j)/10)





– **B** tiene ~ 10 veces
menos valores distintos de pixel.
– Note además la escala del
eje vertical en $h_B(l)$.

Intervalo de Valores de Pixel Expandidos/Comprimidos

- $B(i,j) = g(A(i,j))$
  Suponga que $g(l)$ representa una función de punto $general$ que incluye: expansión/compresión, énfasis/de-enfasis, redondeo, normalización, etc. de contraste.
- Dada una matríz de imagen $\mathbf{A}$, $B(i,j) = g(A(i,j))$ también es una matríz de imagen.
- $g(l)$ puede no estar conectada o "continua" y además puede no estar compuesta de segmentos de línea conectados.
- ¿Cómo se determina que intervalos de valores de pixel $g(l)$ expanden/comprimen?
  - En general se puede suponer que para intervalos pequeños $g(l)$ se puede linealizar por tramos, segmentos de linea conectados. Calcular la α_i implícita y evaluar $|\alpha_i| \gtrless 1$ ayudaría a determinar los intervalos de expansión/compresión.

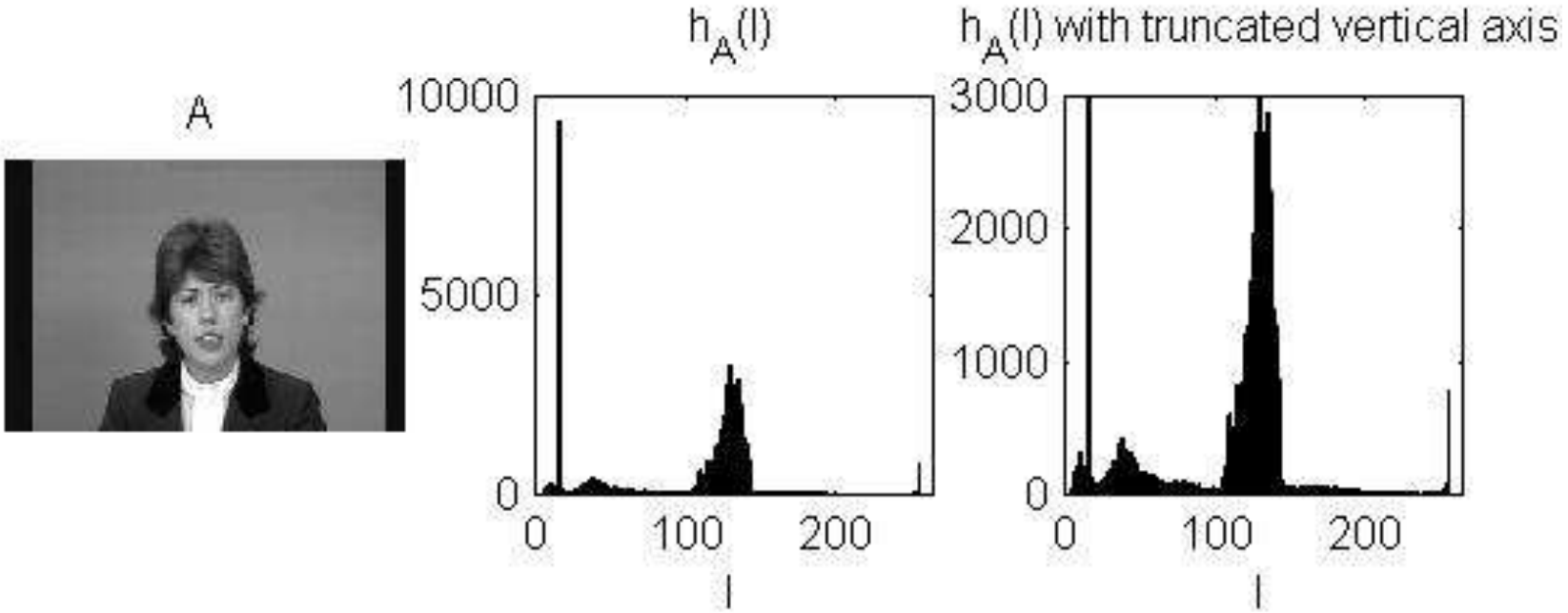
Breve Nota sobre Segmentación de Imagenes

- Si se ve una imagen como la formación de una escena compuesta de diferentes objetos, regiones, etc., entonces la segmentación es la descomposición de una imagen en esos objetos y regiones al asociar o "etiquetar" cada pixel con el objeto al que corresponde.
- La mayoría de los humanos podemos segmentar facilmente una imagen.
- La segmentación automática por computadora es un problema difícil, ya que requiere el uso de algoritmos sofisticados que trabajan en colaboración ("tandem").
- La segmentación de "Alto nivel", tal como segmentar humanos, automóviles, etc., de una imagen es un problema muy difícil que aún se considera sin resolver y se realiza mucha investigación al respecto.
- La segmentación de imágenes basada en histogramas (a partir de procesamiento de Punto) es un algoritmo muy simple que algunas ocasiones se utiliza como suposición inicial en la "verdadera" segmentación de una imagen.

Segmentación de Imágenes Basada en Histograma

- Para una imagen dada, se descompone el intervalo de valores de pixel $(0, \ldots, 255)$ en intervalos "discretos" $R_t = [a_t, b_t],\ t = 1, \ldots, T$, en donde T es el número total de segmentos.
- Cada R_t se obtiene típicamente como un intervalo de valores de pixel que corresponde a un pico de $h_A(l)$.
- Se "etiquetan" los pixeles con valores de pixel dentro de cada R_t a través de una función de punto.
- Suposición principal: Se supone que cada objeto se compone de pixeles con valores de pixel *similares*.

Ejemplo



- $R_1 = [0, 14], R_2 = [15, 15], R_3 = [16, 99], R_4 = [100, 149], R_5 = [150, 220], R_6 = [221, 255]$.
- Etiquetando en Matlab: `>> B1=255*((A>=0)&(A<=14));` , etc.

Ejemplo - cont.

A

B1 (R_1=[0,14])

B2 (R_2=[15,15])

B3 (R_3=[16,99])

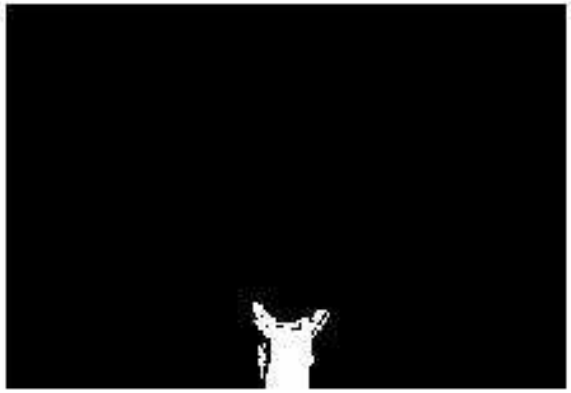
Ejemplo - cont.

A

B4 (R_4=[100,149])

B5 (R_5=[150,220])

B6 (R_6=[221,255])



Ejemplo - cont.

A

B (Histogram Segmented)

- Calcule la media de muestra de cada segmento (`>> m1=sum(sum(B1.*A))/sum(sum(B1))`, etc.).
- `C=m1*B1 + m2*B2 + m3*B3 + m4*B4 + m5*B5 + m6*B6.`
  $B(i,j) = g_s^C(C(i,j))$.

Limitaciones

- La segmentación basada en histograma opera en cada pixel de la imagen de forma independiente. Como se mencionó previamente, la suposición principal es que los objetos deben estar compuestos de pixeles con valores de pixel similares.
- Este procesamiento independiente ignora una segunda propiedad importante: Los pixeles dentro de un objeto deberían estar conectados *espacialmente*. Por ejemplo, B3, B4, B5 agrupan objetos/regiones desconectados espacialmente dentro del mismo segmento.
- En la práctica, se debería utilizar la segmentación basada en histograma en colaboración (tandem) con otros algoritmos que aseguraran que los objetos/regiones calculados estuvieran conectados espacialmente.

Igualación (Ecualización) de Histograma

- Para una imagen dada **A**, se diseñará una función de punto especial $g_A^e(l)$ llamada la función de punto igualadora de histograma para **A**.
- Si $B(i,j) = g_A^e(A(i,j))$, entonces nuestro interés es hacer $h_B(l)$ tan uniforme/plana como sea posible *independientemente* de $h_A(l)$!
- La igualación de histograma nos ayudará a:
  - Expandir/Comprimir una imagen de tal manera que:
    - Los valores de pixel que ocurren frecuentemente en **A** ocupan un intervalo dinámico mayor en **B**, i.e., se expanden y se vuelven más visibles.
    - Los valores de pixel que ocurren menos frecuentemente en **A** ocupan un intervalo dinámico menor en **B**, i.e., se comprimen y se vuelven menos visibles.
  - Comparar imágenes "mapeando" sus histogramas dentro de un histograma estándar y en ocasiones "deshaciendo" los efectos de algunos procesamientos desconocidos.
- Las técnicas que se van a utilizar para obtener $g_A^e(l)$ también son aplicables en la modificación/especificación de histogramas.

Variables Aleatorias Continuas de Amplitud

- Sea χ una variable aleatoria continua de amplitud $\chi \in (-\infty, +\infty)$.

  $f_\chi(x)$: función de densidad de probabilidad de χ,
  $F_\chi(x)$: función de distribución de probabilidad de of χ.

$$f_\chi(x)dx = \text{Probabilidad}(x \leq \chi < x + dx) \quad (15)$$

$$F_\chi(x) = \text{Probabilidad}(\chi \leq x) \quad (16)$$

- Propiedades:

$$F_\chi(x) = \int_{-\infty}^{x} f_\chi(t)dt \Rightarrow \frac{dF_\chi(x)}{dx} = f_\chi(x) \quad (17)$$

$$f_\chi(x) \geq 0 \Rightarrow F_\chi(x) \geq 0, \; F_\chi(x + dx) - F_\chi(x) \geq 0 \quad (18)$$

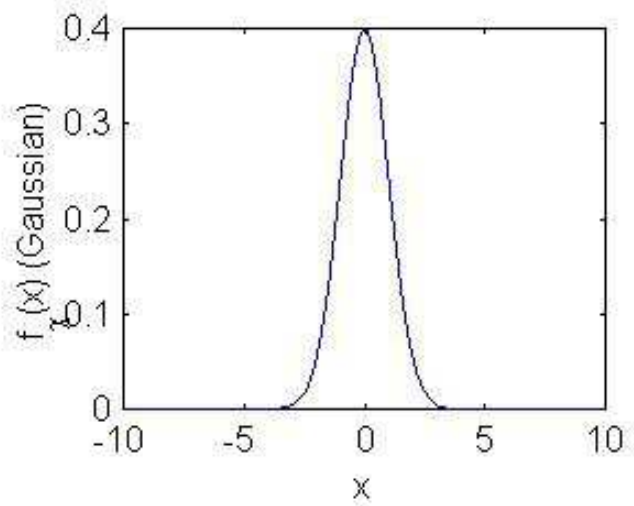
$F_\chi(x)$ es una función no-decreciente.

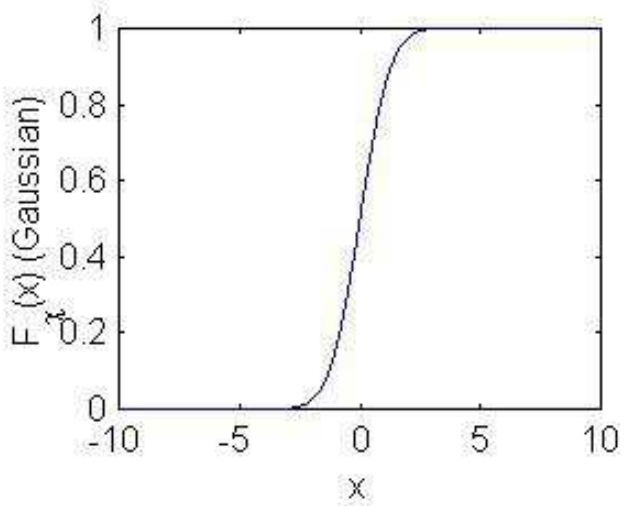
$$\int_{-\infty}^{+\infty} f_\chi(t)dt = 1 \Rightarrow f_\chi(x)|_{x=+/-\infty} = 0 \quad (19)$$

$$F_\chi(x)|_{x=+\infty} = 1 \quad (20)$$

$$F_\chi(x)|_{x=-\infty} = 0 \quad (21)$$

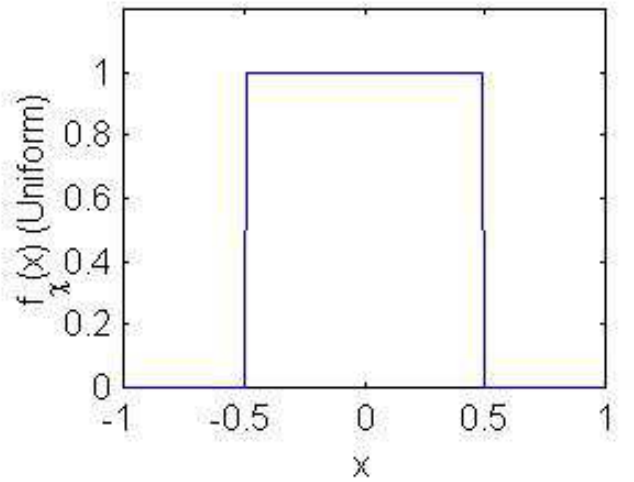
Ejemplo

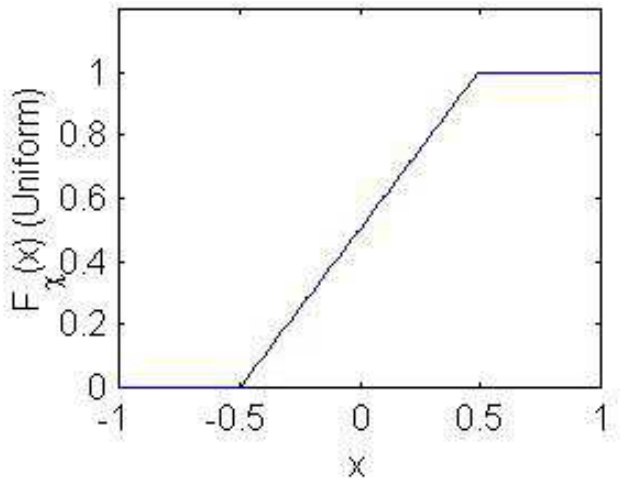




Gaussiana:

$$f_\chi(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$





Uniforme $(a < b)$:

$$f_\chi(x) = \begin{cases} \frac{1}{b-a} & a < x < b \\ 0 & \text{cualquier otro} \end{cases}$$

Cálculo de la Media y la Varianza

- Media (μ):

$$\mu = \int_{-\infty}^{+\infty} x f_\chi(x) dx \tag{22}$$

Analogía: Precio promedio de manzanas

  - "Si compro $f_\chi(x)dx$ manzanas a un precio de x, ..."
  - "El precio total que pagué: $P = \int_{-\infty}^{+\infty} x f_\chi(x) dx$.
  - "El número total de manzanas que compré: $N = \int_{-\infty}^{+\infty} f_\chi(x) dx = 1$.
  - "Mi precio promedio por la compra total: $\mu = P/N$.

- Varianza (σ^2):

$$\sigma^2 = \int_{-\infty}^{+\infty} (x-\mu)^2 f_\chi(x) dx \tag{23}$$

Derivación Principal

- Ahora se obtendrá una nueva variable aleatoria Y $(f_Y(y), F_Y(y))$ como *función* de la variable aleatoria χ, i.e., $Y = g(\chi)$.
- Se desea hacer de Y una variable aleatoria uniforme *(una variable aleatoria que tenga una función de densidad de probabilidad uniforme)* independientemente de la densidad de χ.
- La suposición principal será:
  - Suponga que $F_\chi(x)$ es una función continua y estrictamente creciente (comparable al caso general de no-decrecencia como en la Ecuación 18)
  - Note que $F_\chi(x)$ es uno a uno, lo que permite utilizar su inversa $F_\chi^{-1}(x)$.

Derivación Principal - cont.

- Sea $Y = F_\chi(\chi)$, i.e., $g(\chi) = F_\chi(\chi)$. Note que $Y \in [0,1]$ y $f_Y(y) = 0$ si $y \notin [0,1]$.
- Derivando $F_Y(y)$ para $y \in [0,1]$:

$$\begin{aligned} F_Y(y) &= \text{Probabilidad}(Y \leq y) \\ &= \text{Probabilidad}(F_\chi(\chi) \leq y) \\ &= \text{Probabilidad}(\chi \leq F_\chi^{-1}(y)) \\ &= F_\chi(F_\chi^{-1}(y)) \\ &= y \end{aligned}$$

  en donde se utilizó la Ecuación 16.
- Utilizando la Ecuación 17 y $f_Y(y) = 0$ si $y \notin [0,1]$, se tiene

$$f_Y(y) = \begin{cases} 0 & y < 0,\ y > 1 \\ 1 & y \in [0,1] \end{cases} \tag{24}$$

  i.e., Y es una variable aleatoria uniforme con $a = 0$ y $b = 1$.

Variables Aleatorias Discretas de Amplitud

- Sea Θ una variable aleatoria discreta de amplitud.
  $\Theta = x_i$ para algunos i, $\ldots, -1, 0, 1, \ldots$.
  x_i es una secuencia de valores posibles de Θ.

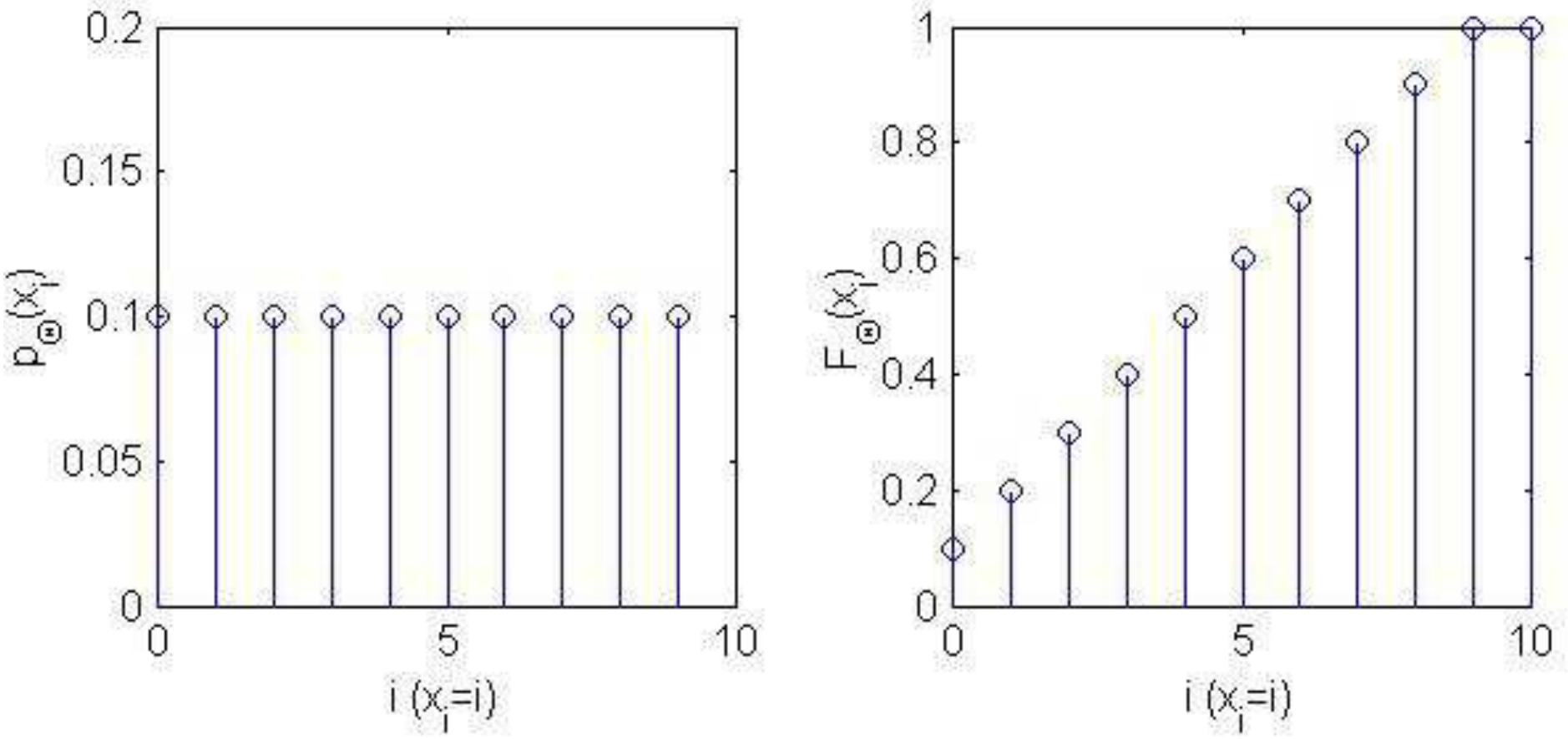
  $p_\Theta(x_i)$: función de probabilidad de masa de Θ,
  $F_\Theta(x_i)$: función de distribución de probabilidad of Θ.

$$p_\Theta(x_i) = \text{Probabilidad}(\Theta = x_i) \tag{25}$$
$$F_\Theta(x_i) = \text{Probabilidad}(\Theta \leq x_i) \tag{26}$$

- Propiedades:

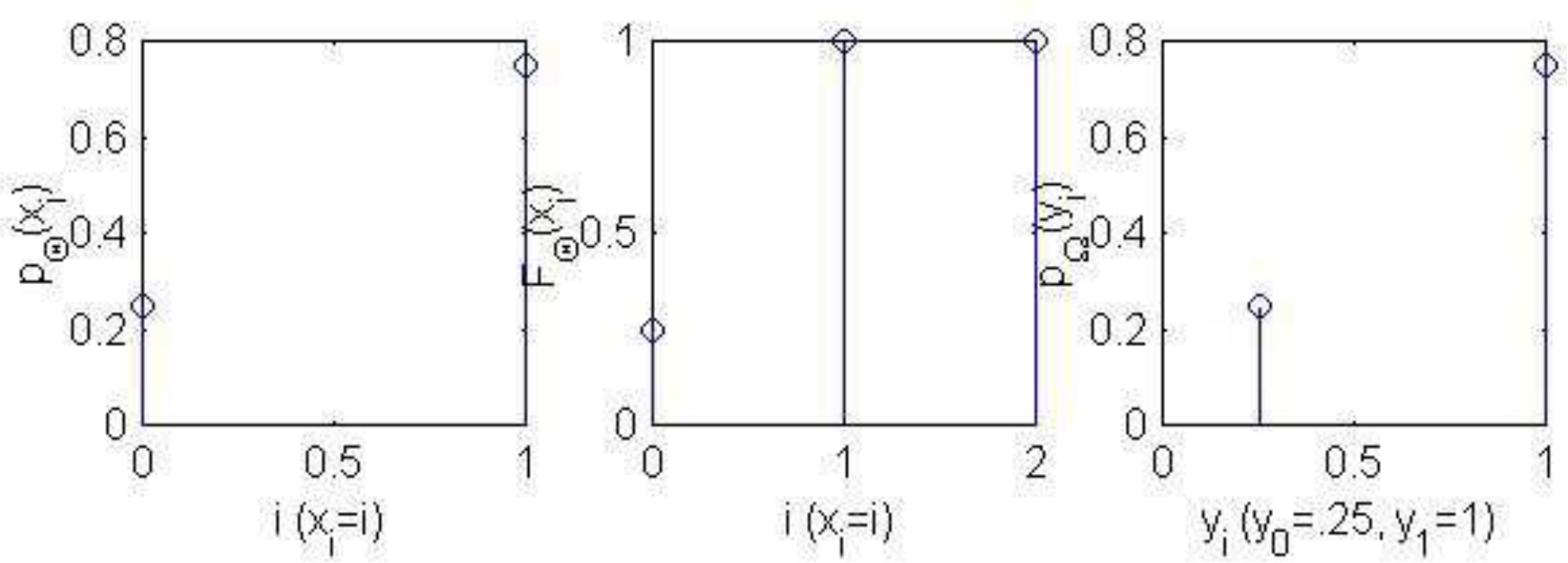
$$F_\Theta(x_i) = \sum_{j=-\infty}^{j=i} p_\Theta(x_j) \tag{27}$$
$$p_\Theta(x_i) = F_\Theta(x_i) - F_\Theta(x_{i-1}) \geq 0 \tag{28}$$
$$\sum_{j=-\infty}^{j=+\infty} p_\Theta(x_j) = 1 \tag{29}$$

Ejemplo



Funciones de probabilidad de masa y distribución de probabilidad para una variable aleatoria discreta de amplitud uniforme.

Derivación para V. A. Discretas de Amplitud



- Sea $\Omega = F_\Theta(\Theta)$. $\Omega = y_i = F_\Theta(x_i)$ para algunos i, $\ldots, -1, 0, 1, \ldots$.
- La derivación previa para variables aleatorias continuas de amplitud no "funciona" para variables aleatorias discretas de amplitud .
- En general Ω no es una variable aleatoria uniforme.

Histograma como una Función de Probabilidad de Masa

- Para una imagen dada $\mathbf{A}$, considere los pixeles de la imagen como las realizaciones de una variable aleatoria discreta de amplitud "A".
  - Por ejemplo, suponga que se lanza una moneda (Caras=255 y Cruces=0) $N \times M$ veces y se registran los resultados como una matriz de imagen de N por M.
- Defina la función de probabilidad de masa de muestra $p_A(l)$ como la probabilidad de elegir aleatoriamente un pixel que tiene el valor l.

$$p_A(l) = \frac{h_A(l)}{NM} \tag{30}$$

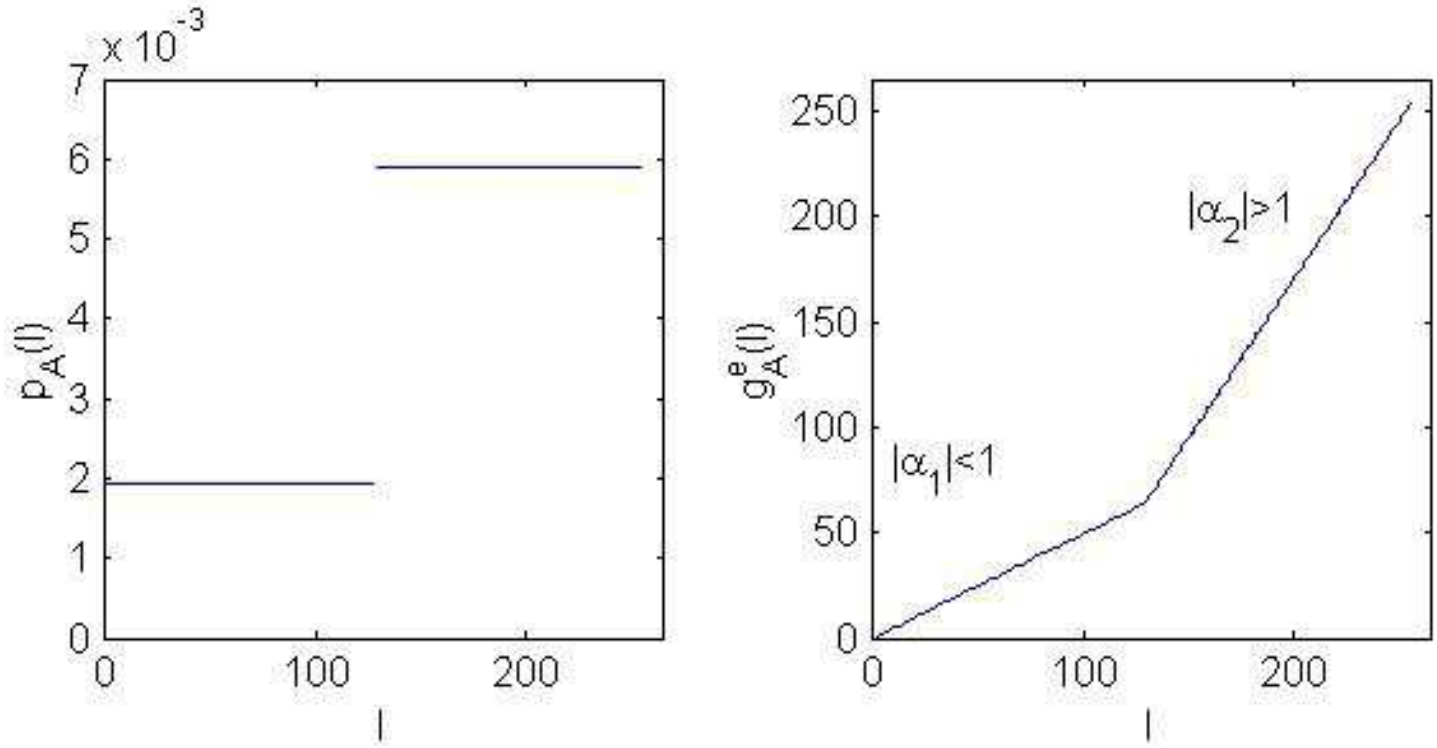
- Note que la media y varianza de muestra de la que se habló en la Clase 2 puede calcularse como:

$$\begin{aligned} m_A &= \sum_{l=0}^{255} l p_A(l) \\ \sigma_A^2 &= \sum_{l=0}^{255} (l - m_A)^2 p_A(l) \end{aligned}$$

Función de Punto de Igualación de Histograma

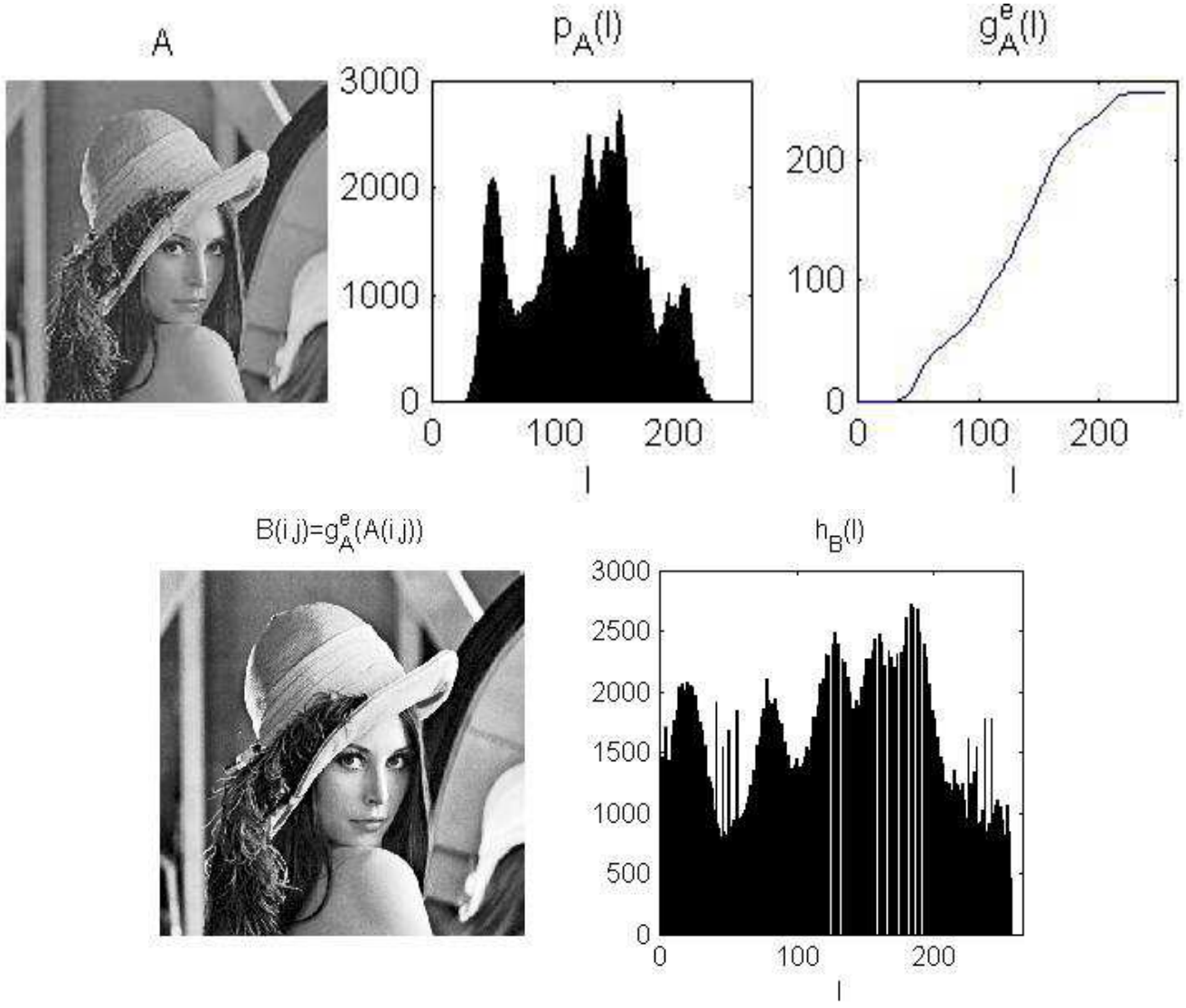
- Sea $g_1(l) = \sum_{k=0}^{l} p_A(k)$. Note que $g_1(l) \in [0, 1]$.
- $g_A^e(l) = \text{round}(255 g_1(l))$ es la función de punto de igualación de histograma para la imagen $\mathbf{A}$.
- Imagen $\mathbf{A}$ $\Rightarrow$ "imagen igualada (ecualizada)" $\Rightarrow B(i, j) = g_A^e(A(i, j))$.
- Como se ha visto, en general $p_B(l)$ no será una función de probabilidad de masa uniforme pero se espera que este cercana.
- En matlab $>>$ help filter para construir $g_A^e(A(i, j))$ rápidamente.
- Suponiendo que gAe es un arreglo que contiene la $g_A^e(l)$ calculada, se puede utilizar $>> B = gAe(A + 1);$ para obtener la imagen igualada.

Expansión y Compresión

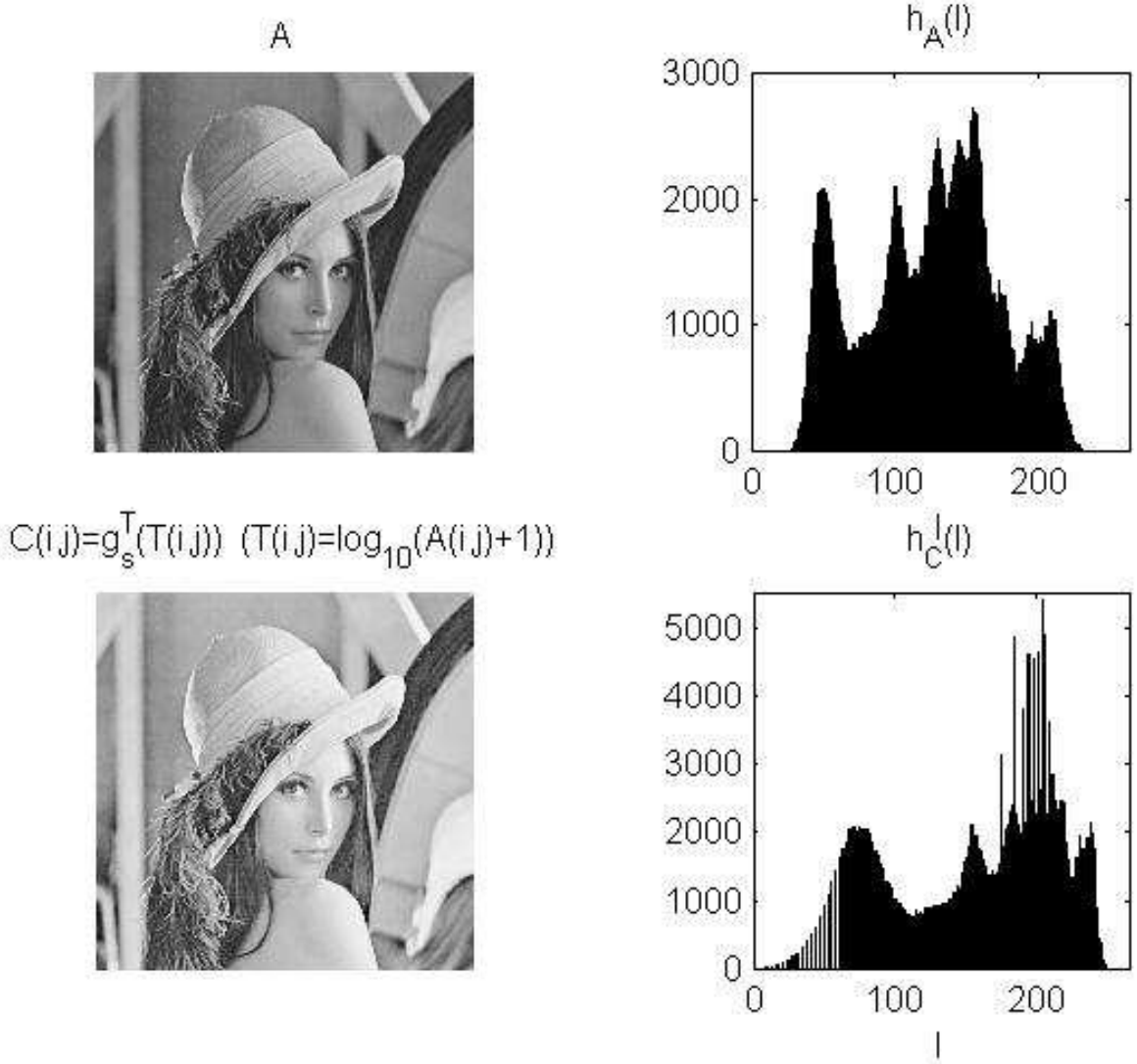


- $g_A^e(l)$ expande el intervalo de valores de pixel que ocurren frecuentemente en $\mathbf{A}$.
- $g_A^e(l)$ comprime el intervalo de valores de pixel que no ocurren frecuentemente en $\mathbf{A}$.

Ejemplo


A
$p_A(l)$
$g^e_A(l)$
3000
2000
1000
0
0
100
200
l
200
100
0
0
100
200
l
$B(i,j)=g^e_A(A(i,j))$
$h_B(l)$
3000
2500
2000
1500
1000
500
0
0
100
200
l

Comparación/ “Deshaciendo”

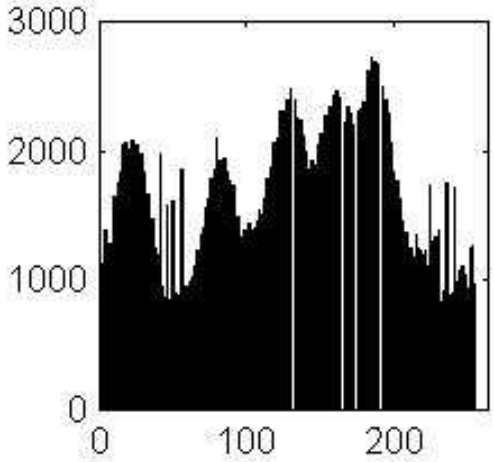


En lugar de comparar **A** y **C**, compare sus versiones igualadas.

Comparación/"Deshaciendo" - cont.

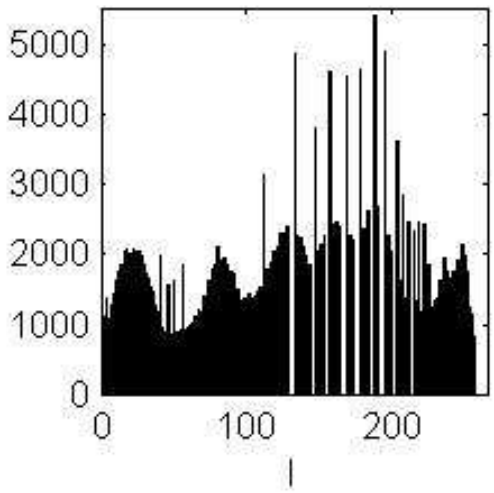
$B(i,j)=g_A^e(A(i,j))$

$h_B(l)$



$D(i,j)=g_C^e(C(i,j))$

$h_D^I(l)$



Resumen

- En esta clase se aprendió un método de segmentación de imágenes simple basado en histograma, así como sus limitaciones.
- Se hizo una revisión de variables aleatorias continuas y discretas de amplitud y se utilizaron sus propiedades para derivar la función de punto de igualación de histograma.
- También se definieron las relaciones entre histogramas de imagen y funciones de probabilidad de masa de muestra de imágenes.
- Finalmente se vieron algunos ejemplos de igualación y se aprendió que esperar de una función de punto de igualación de histograma cuando se aplica a una imagen.
- Leer Capítulo 7, pages 241-244 en el libro de texto.

Tarea III

1. El histograma de una imagen $\mathbf{A}$ es $h_A(l) = l,\ (l = 0, \ldots, 255)$. Una función de punto

$$g(l) = \begin{cases} l & 0 \leq l < 128 \\ 255 - l & 128 \leq l \leq 255 \end{cases}$$

   Sea $B(i, j) = g(A(i, j))$. Calcule $h_B(l)$ sin computadora. Muestre todo su trabajo.

2. Implemente la segmentación basada en histograma en su imagen identifique los picos de su histograma con los "objetos" a los que corresponden. Muestre su imagen, sus histogramas, los intervalos, etc. Muestre los objetos identificados. Finalmente construya la imagen de histograma segmentado.

3. Derive la media y varianza para densidades continuas de amplitud Gaussiana y uniforme.

4. Iguale su imagen. Muestre el "antes" y "después" de imágenes e histogramas. ¿Es uniforme el histograma de la imagen igualada? ¿Qué regiones se expandieron/comprimieron? (Sea lo más preciso posible)

5. Implemente un ejemplo de Comparación/"Deshacer" en su imagen.

Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

Igualación de Histograma

- $g_1(l) = \sum_{k=0}^{l} p_A(k) \Rightarrow g_1(l) - g_1(l-1) = p_A(l) = \frac{h_A(l)}{NM}$ $(l = 1, \ldots, 255)$.
- $g_A^e(l) = \text{round}(255 g_1(l))$ es la función de punto de igualación de histograma para la imagen $\mathbf{A}$.
- $B(i,j) = g_A^e(A(i,j))$ es la versión igualada de histograma de $\mathbf{A}$.
- En general, la igualación de histograma expande/comprime una imagen de manera que:
  - Los valores de pixel que ocurren frecuentemente en $\mathbf{A}$ ocupan un intervalo dinámico mayor en $\mathbf{B}$, i.e., se expanden y vuelven más visibles.
  - Los valores de pixel que no ocurren frecuentemente en $\mathbf{A}$ ocupan un intervalo dinámico menor en $\mathbf{B}$, i.e., se comprimen y vuelven menos visibles.
- La igualación de histograma no es ideal, i.e., en general $\mathbf{B}$ tendrá un histograma "más plano" que $\mathbf{A}$, pero no se garantiza que $p_B(l)$ sea uniforme (plano).

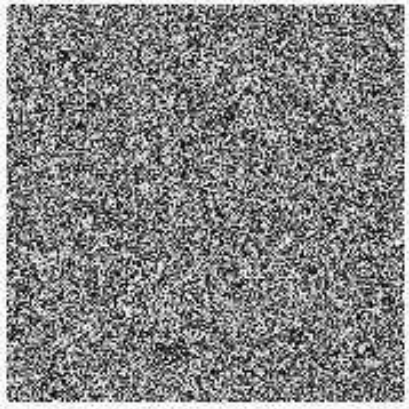
Imagenes Aleatorias - Imagenes de Ruido Blanco

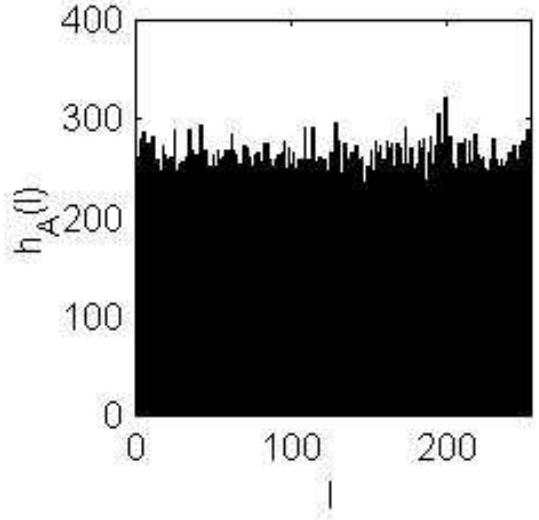
- Una salida simple de una variable aleatoria continua uniforme de amplitud $\chi \in [0, 1]$ en Matlab: $>>\ x = \text{rand}(1, 1);$
- Una matriz $N \times M$ de salidas de una variable aleatoria continua uniforme de amplitud $\chi \in [0, 1]$ en Matlab: $>>\ X = \text{rand}(N, M);$
- Una matriz de imagen $N \times M$ de salidas de una variable aleatoria discreta uniforme de amplitud $\Theta \in \{0, 1, \ldots 255\}$ en Matlab: $>>\ A = \text{round}(255 * X);$

- Una salida simple de una variable aleatoria continua gaussiana de amplitud $\chi\ (\mu = 0, \sigma^2 = 1)$ En matlab: $>>\ x = \text{randn}(1, 1);$
- Una matriz $N \times M$ de salidas de una variable aleatoria continua gaussiana de amplitud $\chi\ (\mu = 0, \sigma^2 = 1)$ en Matlab: $>>\ X = \text{randn}(N, M);$
- Una matriz de $N \times M$ imagen de salidas de una variable aleatoria discreta gaussiana de amplitud $\Theta \in \{0, 1, \ldots 255\}$: $A(i, j) = g^s_X(X(i, j))$.

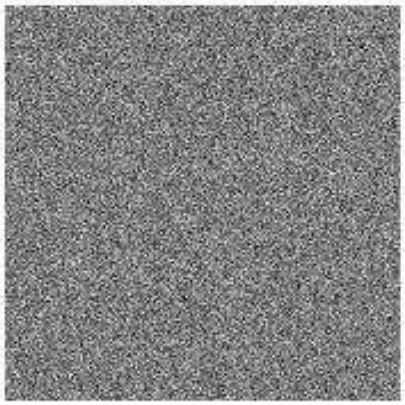
Ejemplo

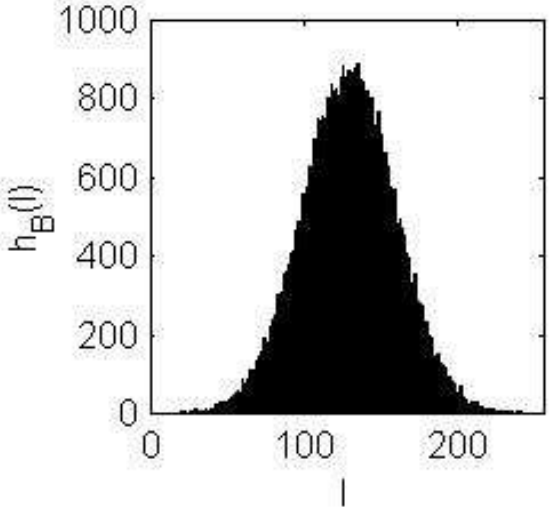
A=round(255*rand(256,256))



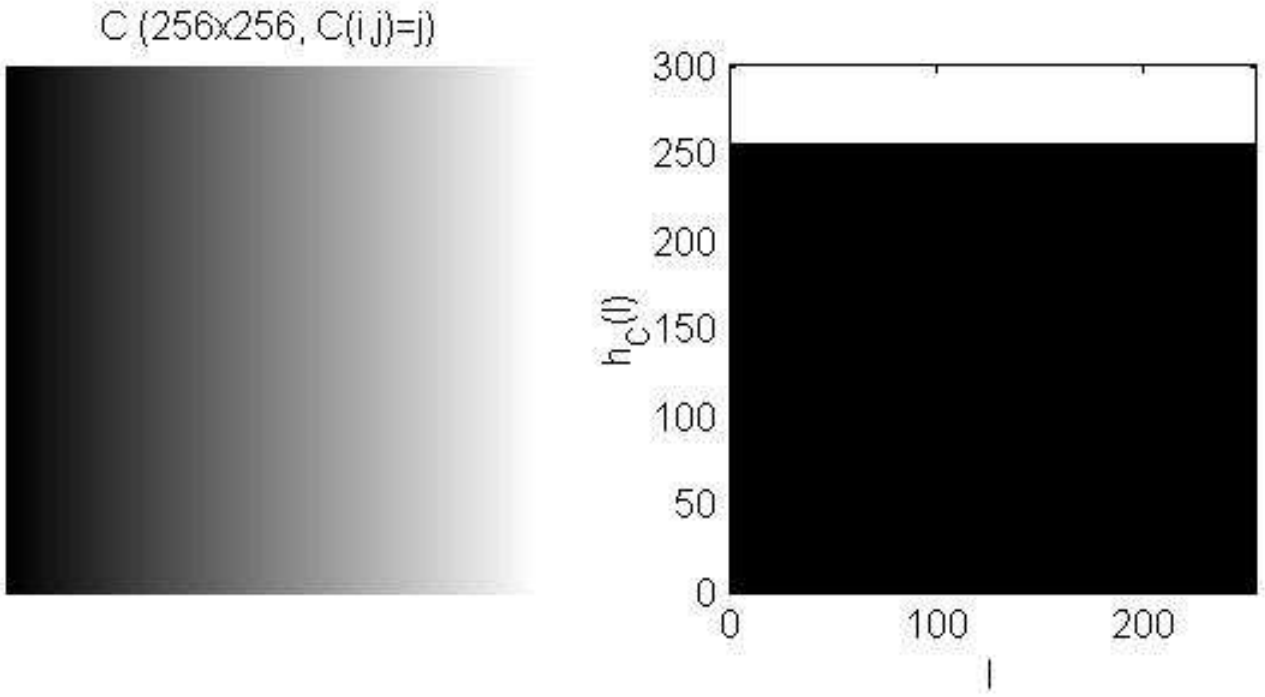


B(i,j)=g_X^S(X(i,j)) (X=randn(256,256))





Precaución



- Recuerde que dos imágenes completamente diferentes pueden tener histogramas muy similares.

Acoplamiento de Histograma - Especificación

- Dadas las *imágenes* $\mathbf{A}$ y $\mathbf{B}$, utilizando procesamiento de punto quisiéramos generar una imagen $\mathbf{C}$ de $\mathbf{A}$ tal que $h_C(l) \sim h_B(l),\ (l = 0, \ldots, 255)$.
- Más generalmente, dada una imagen $\mathbf{A}$ y un histograma $h_B(l)$ (o función de probabilidad de masa de muestra $p_B(l)$), quisiéramos generar una imagen $\mathbf{C}$ tal que $h_C(l) \sim h_B(l),\ (l = 0, \ldots, 255)$.
- El Acoplamiento de Histograma/Especificación nos permite "acoplar" la distribución de escala de grises de una imagen a la distribución de escala de grises de otra image.

Derivación para V.A. Continuas de Amplitud

- Se ha visto que para variables aleatorias (V.A.s) continuas de amplitud χ con incremento estricto y $F_\chi(x)$ continua, la V.A. $Y = F_\chi(\chi)$ tiene la función de densidad/distribución de probabilidad uniforme.
- *De forma equivalente*, para una V.A. continua de amplitud $Y \in [0, 1]$ que tiene la función de densidad de probabilidad uniforme, $\chi = F_\chi^{-1}(Y)$ tiene la función de densidad [distribución] de probabilidad $f_\chi(x)$ $[F_\chi(x)]$.

$$\chi \ [F_\chi(x)] \ \Rightarrow \ Y = F_\chi(\chi) \ [\text{uniforme}]$$
$$Y \ [\text{uniforme}] \ \Rightarrow \ \chi = F_\chi^{-1}(Y) \ [F_\chi(x)]$$

- Ahora, supongamos que tenemos una V.A. continua de amplitud Z con incremento estricto y $F_Z(z)$ continua. entonces:

$$Y \ (\text{uniform}) \ \Rightarrow \ W = F_Z^{-1}(Y) \ [F_Z(w)] \tag{31}$$

pero podemos generar la V.A. uniforme requerida Y de χ vía $Y = F_\chi(\chi)$ lo que significa que W puede ser generada a partir de χ vía:

$$W = F_Z^{-1}(Y) = F_Z^{-1}(F_\chi(\chi)) \tag{32}$$

Resultado principal

- Dada una V.A. continua de amplitud χ con incremento estricto y continua $F_\chi(x)$, sea $F_Z(z)$ la distribución *especificada* ($F_Z(z)$ con incremento estricto y continua).
- Entonces, $W = F_Z^{-1}(F_\chi(\chi))$ es una V.A. que es función de χ con $F_W(w) = F_Z(w)$.
- Para V.A. discretas de amplitud esta derivación en general no funciona. Sin embargo, de manera similar a la función de punto de igualación de histograma se generará una función de punto que opere sobre una imagen $\mathbf{A}$ para "acoplar" su histograma al de la imagen $\mathbf{B}$.

Algoritmo

- En general no podremos calcular la inversa de las funciones de distribución para V.A. discretas de amplitud.
- Sean $p_A(l),\ p_B(l)\ (l = 0, \ldots, 255)$ las funciones de probabilidad de masa de las imágenes $\mathbf{A}$ y $\mathbf{B}$ respectivamente.
- Sean $g_1(l) = \sum_{k=0}^{l} p_A(k)$ y $g_2(l) = \sum_{k=0}^{l} p_B(k)$.
- Genera el "histograma acoplado" $\mathbf{C}$ como $C(i,j) = g_3(A(i,j))$ en donde:

$$\begin{aligned} g_3(l) &= m \qquad (m \in \{0, 1, \ldots, 255\}) \\ m &= \text{mín}\{k | g_2(k) - g_1(l) \geq 0,\ \ k = 0, \ldots, 255\} \end{aligned} \tag{33}$$

- Suponiendo que g_2 y g_1 fueron precalculadas:

$$>> \quad \text{for} \quad i \ = 1 : 256 \tag{34}$$

$$g3(i) = 256 - \text{sum}(g2 >= g1(i)); \tag{35}$$

$$\text{end}; \tag{36}$$

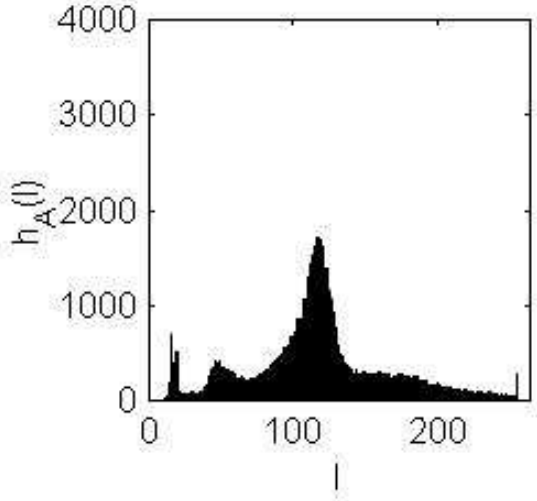
Ejemplo I

Ejemplo 7.1 en el libro de texto (P. 244) en "nuestra notación":

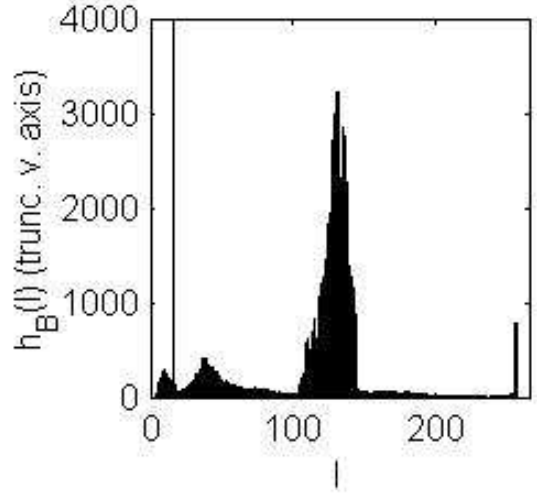
l	$p_A(l)$	$g_1(l)$	$g_3(l) = \text{mín}\{k \mid g_2(k) - g_1(l) \geq 0\}$	$g_2(k)$	$p_B(k)$	k
0	0.25	0.25	1	0	0	0
1	0.25	0.5	1	0.5	0.5	1
2	0.25	0.75	2	1.0	0.5	2
3	0.25	1.0	2	1.0	0	3

Ejemplo II - Diferentes Imágenes de Ajuste de Histograma

A

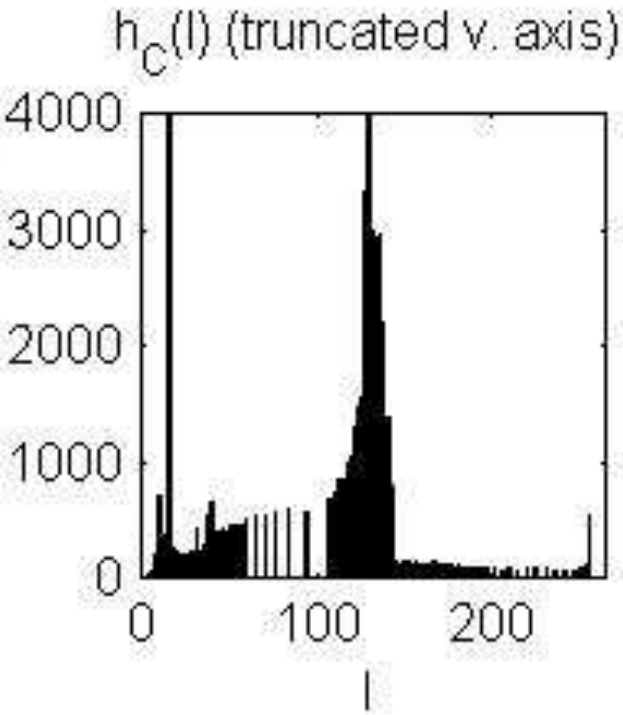


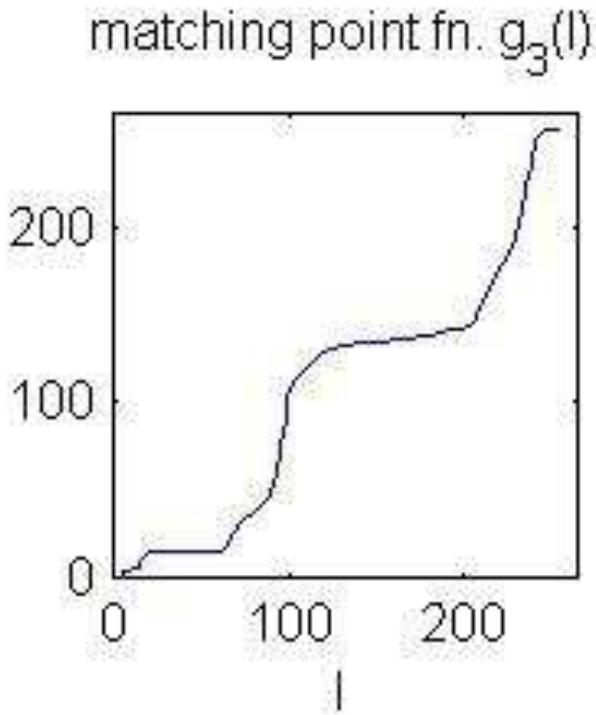
B



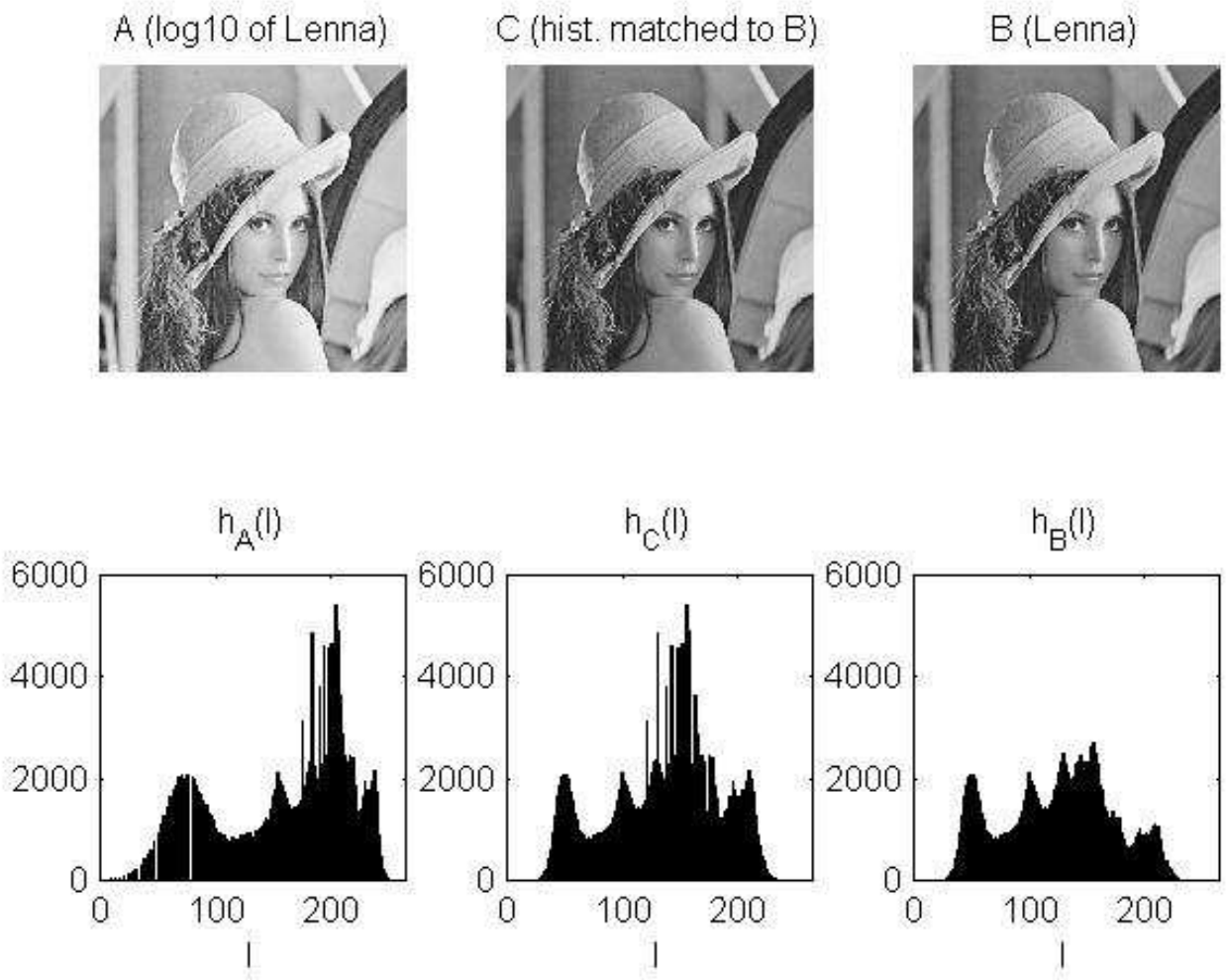
Ejemplo II - cont.

C (hist. matched to B)





Ejemplo III - “Deshacer” vía Especificación de Histograma



Cuantización

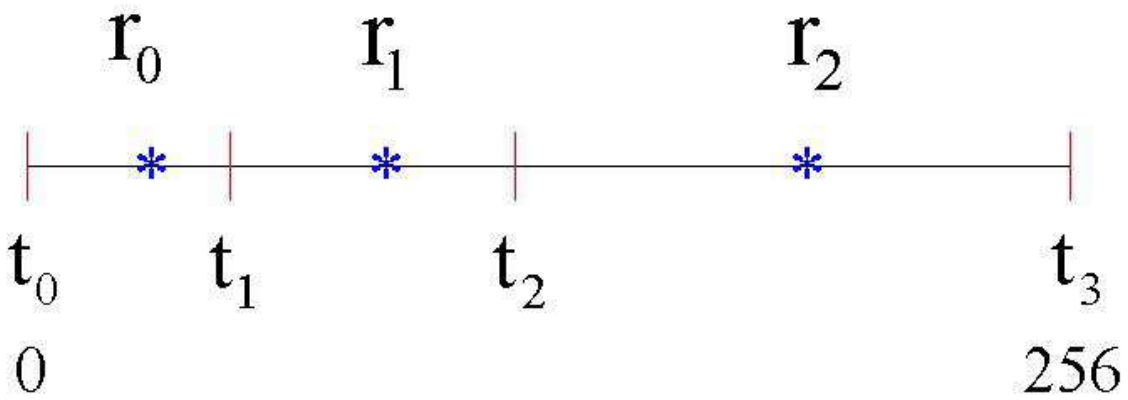
- Sea $t_n \in \{0, 1, \ldots, 255\}$ que denota una secuencia de umbrales $(n = 0, \ldots, P-1)$.
- Considere los "intervalos discretos medio-abiertos" P $R_n = [t_n, t_{n+1}\ )$ $(t_0 = 0,\ t_P = 256)$.
- Sea $r_n \in R_n$ el nivel de reproducción del intervalo R_n.
- Se define la función de cuantización de punto del cuantizador nivel-P $Q(l)$ en terminos de $R_n,\ r_n$ (o en términos de $t_n,\ r_n$) como sigue:

$$Q(l) = \{r_k | l \in R_k,\ k = 0, \ldots, P-1\} \tag{37}$$

  i.e., $l \in R_k \Leftrightarrow Q(l) = r_k$.
- Cuantizando una imagen A en Matlab:

$$\begin{aligned} >>\ & Q = \text{zeros}(256, 1);\ x = (0:255)'; \\ >>\ & \text{for}\ \ i = 1:P \\ & \quad Q = Q + r(i) * ((x >= t(i)) \& (x < t(i+1))); \\ & \quad \text{end};\ \ \ \% \ t(P+1) = 256 \\ >>\ & B = Q(A+1); \end{aligned}$$

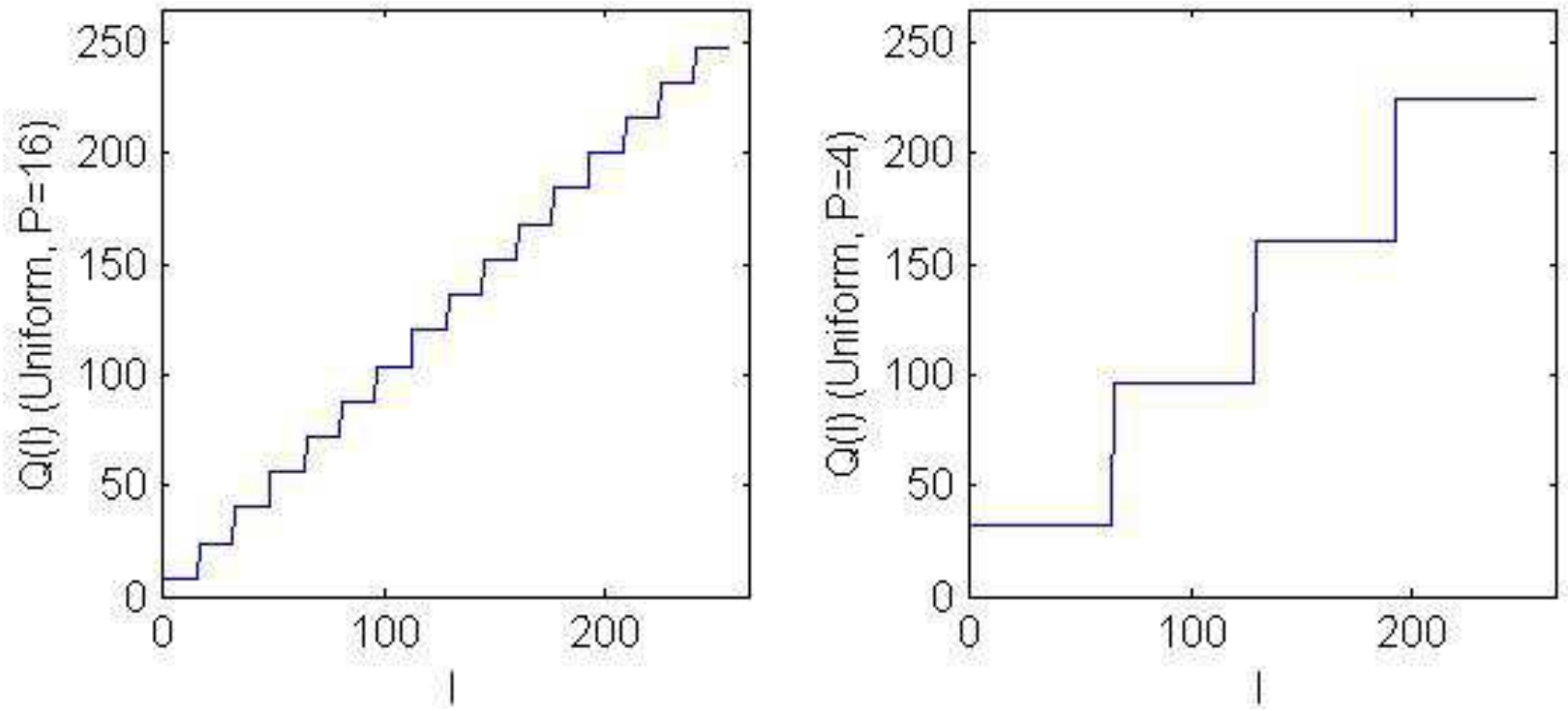
Vista del Intervalo de Partición de un Cuantizador



- $R_n = [t_n, t_{n+1})$.

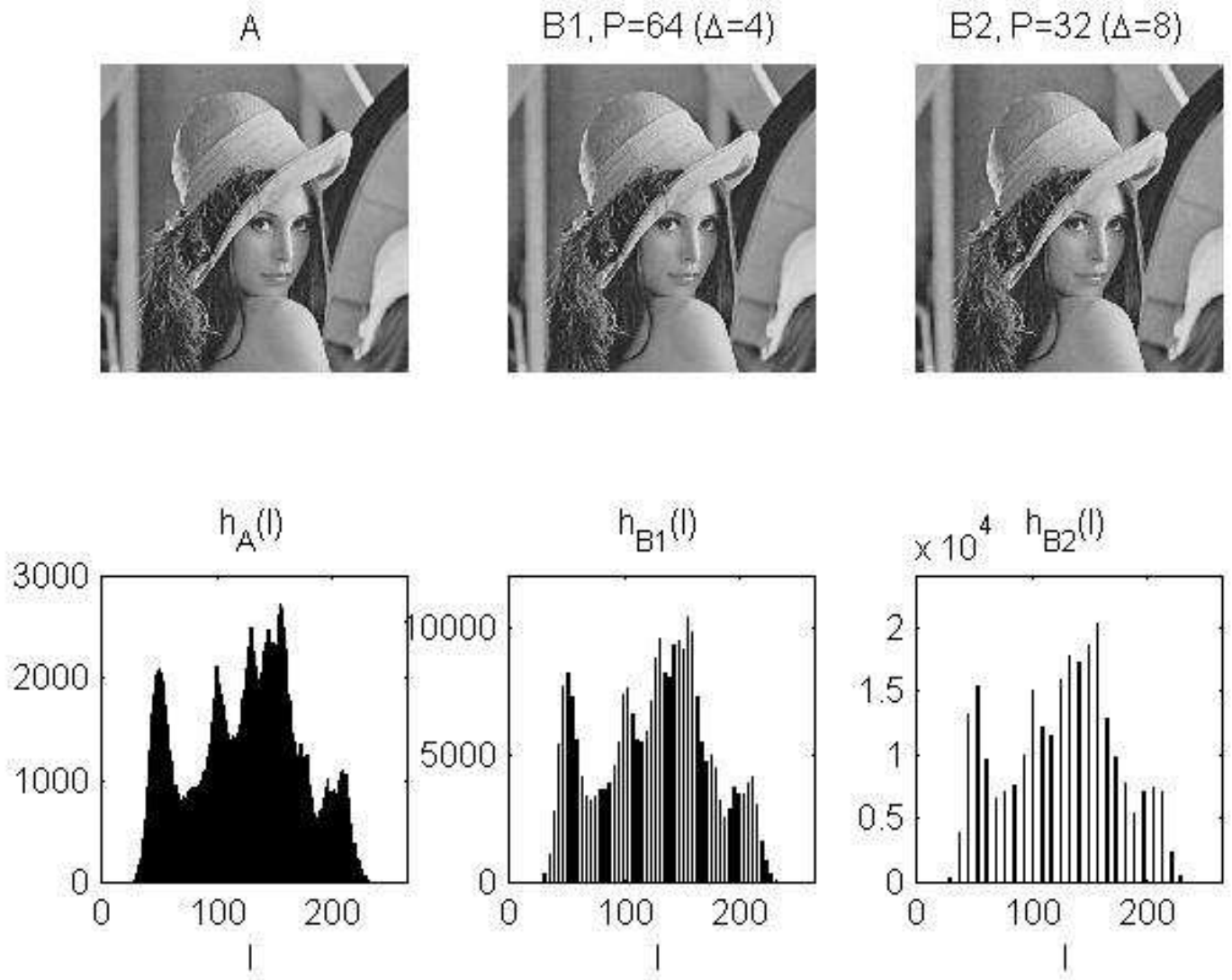
Cuantización Uniforme

- En cuantización uniforme $P = 256/\Delta$, $t_{n+1} - t_n = \Delta,\ \forall\, n$ y $r_n = \frac{t_n + t_{n+1}}{2}$.
- Δ es el tamaño de paso del cuantizador uniforme $(r_n = n\Delta + \Delta/2)$.

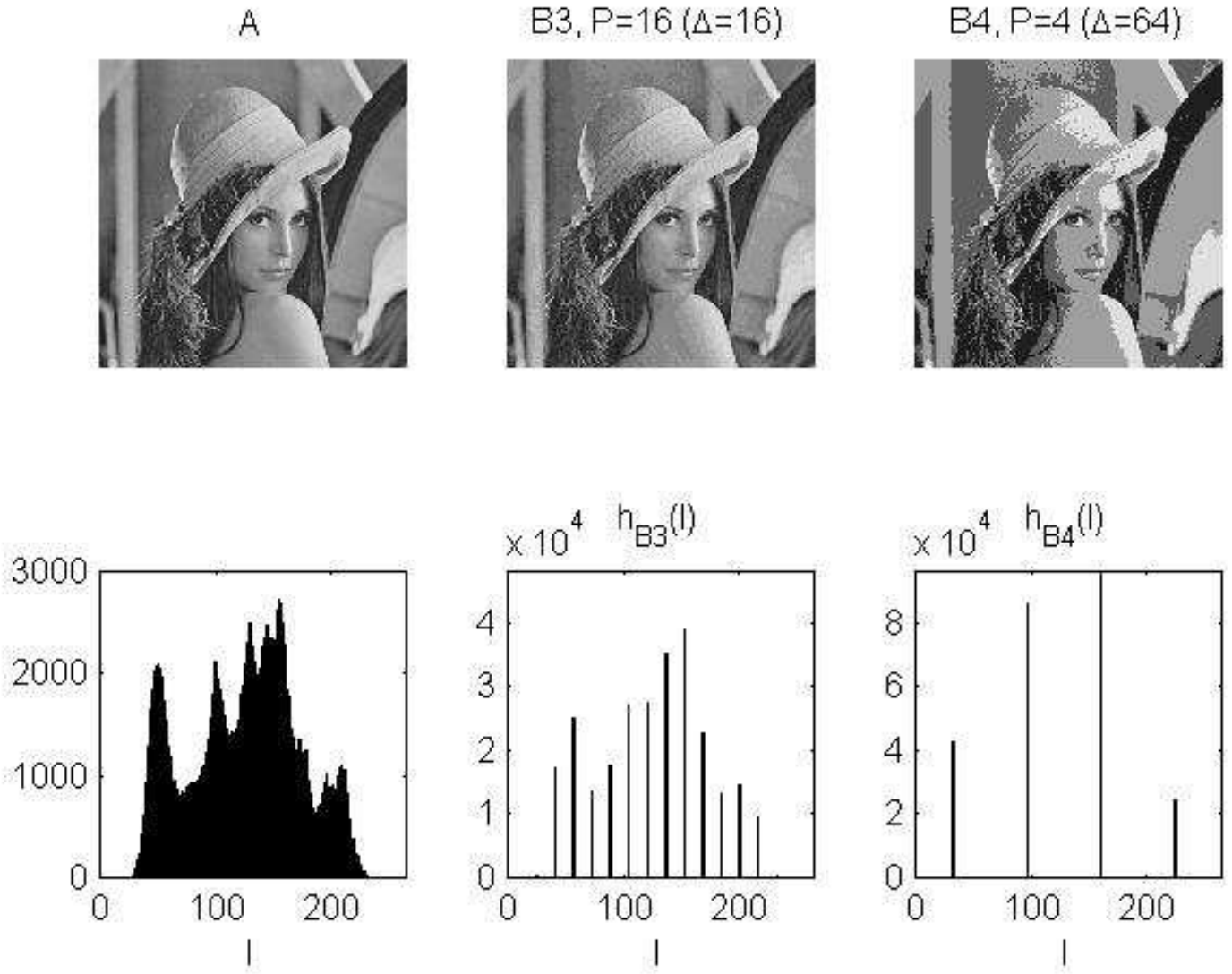


Cuantización uniforme sencilla: `>> B=delta* floor(A/delta)+delta/2;`

Ejemplo

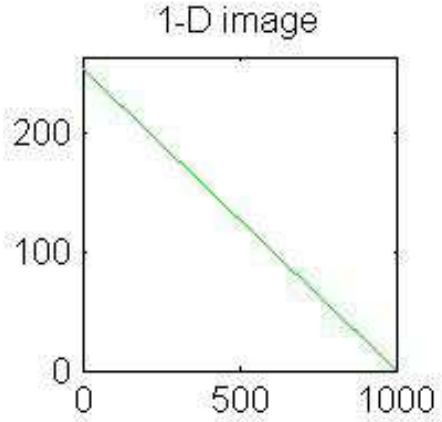

A
B1, P=64 (Δ=4)
B2, P=32 (Δ=8)
h_A(I)
h_B1(I)
h_B2(I)
x 10^4
3000
2000
1000
0
10000
5000
2
1.5
1
0.5
0
100
200
I

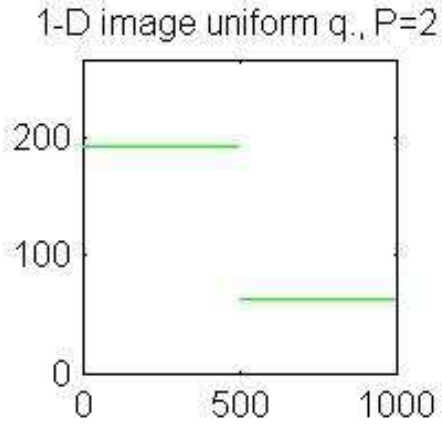
Ejemplo - cont.



Errores (Artifacts) de Cuantización - Contornos Falsos

Contornos Falsos o "Bordes Falsos" en una imagen 1-D y ejemplo previo:





Estadísticas de Cuantización

- La matriz de error de cuantización está definida como $\mathbf{E} = \mathbf{A} - Q(\mathbf{A})$.
- El error cuadrático medio de cuantización de muestra (MSQE) es:

$$\begin{aligned} \text{MSQE} &= \frac{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}(E(i,j))^2}{NM} \qquad (38) \\ &= \frac{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}(A(i,j)-Q(A(i,j)))^2}{NM} \\ &= \sum_{l=0}^{255}(l-Q(l))^2 p_A(l) \qquad (39) \end{aligned}$$

Ejemplo

Para el ejemplo previo:

Δ	Imagen Cuantizada	MSQE
4	B1	1.50
8	B2	5.49
16	B3	22.18
64	B4	334.77

Diseño de Buenos Cuantizadores

- Quisieramos diseñar el $R_n,\ r_n$ (o su equivalente $t_n,\ r_n$) tal que el MSQE sea lo más pequeño posible.
- Repitiendo las Ecuaciones 43 y 45:

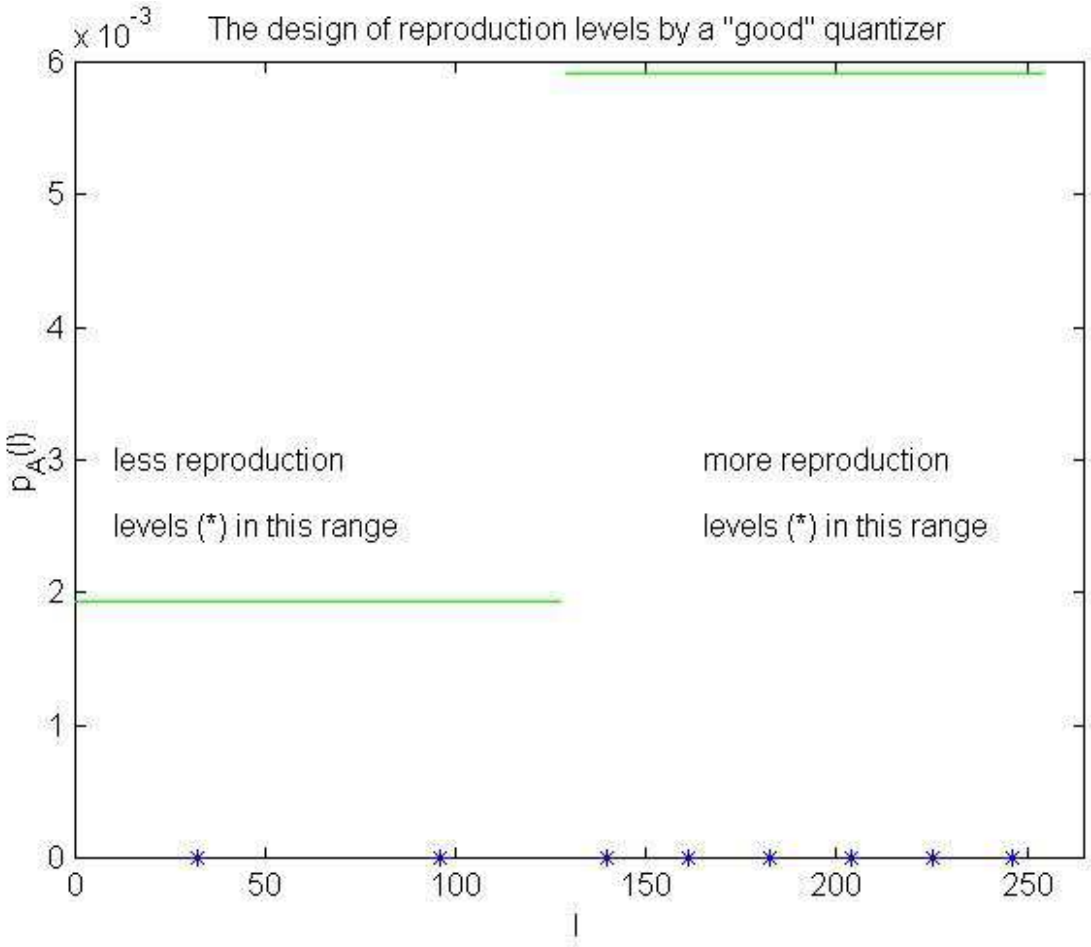
$$Q(l) = \{r_k | l \in R_k,\ k = 0, \ldots, P-1\}$$

$$\text{MSQE} = \sum_{l=0}^{255} (l - Q(l))^2 p_A(l)$$

podemos hacer las importantes observaciones siguientes, suponiendo que P es fijo:

- En los intervalos circundantes de l en donde $p_A(l)$ es grande, un buen cuantizador debería tener varios R_n pequeños, i.e., ya que podríamos tener máximo P intervalos discretos, muchos de estos intervalos deberían estar circundantes a l en donde $p_A(l)$ es grande.
- De manera equivalente, un buen cuantizador no debería "desperdiciar" muchos niveles de reproducción en intervalos alrededor de l en donde $p_A(l)$ es *pequeña*.

Ejemplo



Los niveles de reproducción $*$ son mostrados en la vista del intervalo de partición.

Diseño de Umbrales para Niveles de Reproducción Dados

- Suponga que se da P y "elegimos" los niveles de reproducción r_n en localidades donde $p_A(l)$ es grande.
- ¿Cómo elegimos *optimamente* los umbrales t_n ?
- Por definición $r_{n-1} < t_n \leq r_n$ excepto para $t_0 = 0,\ t_{P+1} = 255$.
- Suponiendo $r_{n-1} < l \leq r_n$. Sea $d_n = (r_{n-1} + r_n)/2$ el punto medio. Entonces, para minimizar MSQE, $Q(l)$ debe ser:

$$Q(l) = \begin{cases} r_{n-1} & l < d_n \\ r_n & l \geq d_n \end{cases} \tag{40}$$

- Para optimizar MSQE

$$t_n = \text{round}(\frac{r_{n-1} + r_n}{2}) \tag{41}$$

- Note que este resultado es *independiente* de $p_A(l)$ y solo depende de r_n.

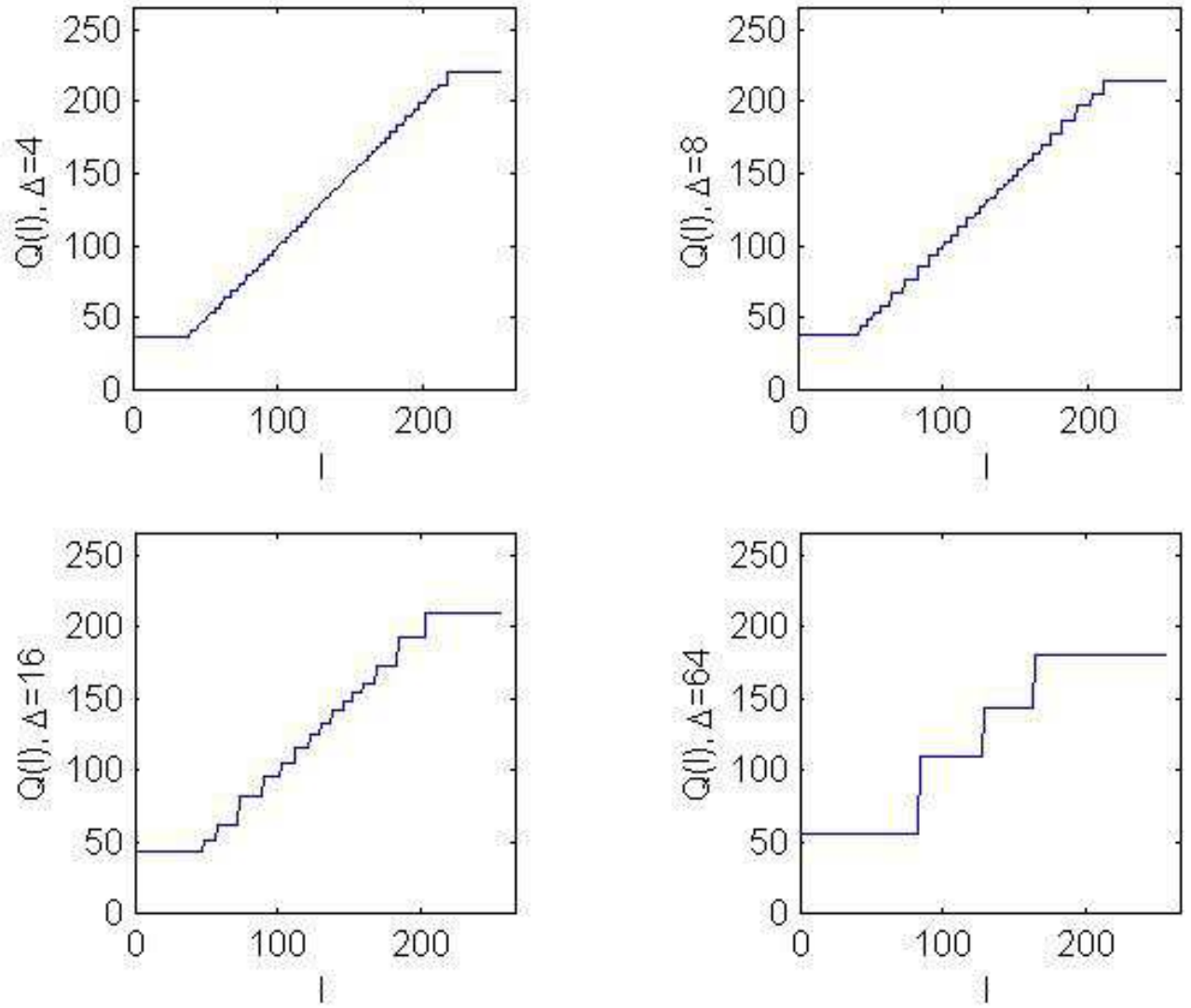
Compresión-Expansión (Companding)

- Dado el r_n sabemos como elegir el t_n.
- Además sabemos que deberíamos elegir r_n cercano a donde $p_A(l)$ es grande y separado de donde $p_A(l)$ es pequeña.
- Suponiendo que $p_A(l)$ es uniforme. Entonces, claramente podemos elegir r_n "uniformemente", i.e., utilizar el r_n que corresponde a un cuantizador uniforme.
- En general $p_A(l)$ no es uniforme, pero "se espera" que $p_B(l)$ para $B(i,j) = g_A^e(A(i,j))$ si lo sea.
- Sea $g_1(l) = \sum_{k=0}^{l} p_A(k)$. Sea $\Delta = 256/P$. eligiendo r_n $(n = 0, \ldots, P-1)$ vía:

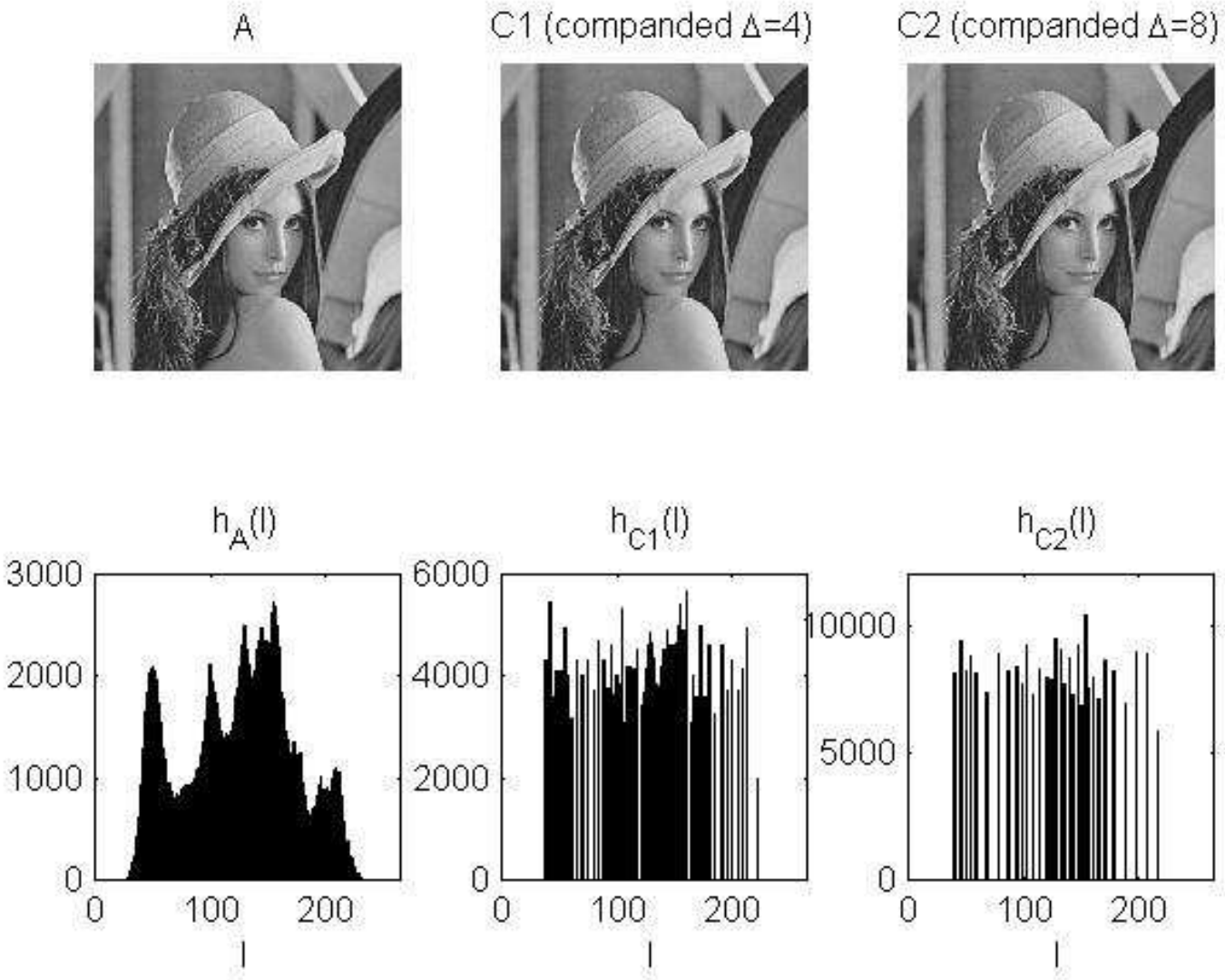
$$r_n = \text{mín}\{k | 255 g_1(k) - (n\Delta + \Delta/2) \geq 0, \;\; k = 0, \ldots, 255\} \tag{42}$$

(Note la similitud con la Ecuación 36 mientras estamos calculando de nuevo una "inversa discreta")

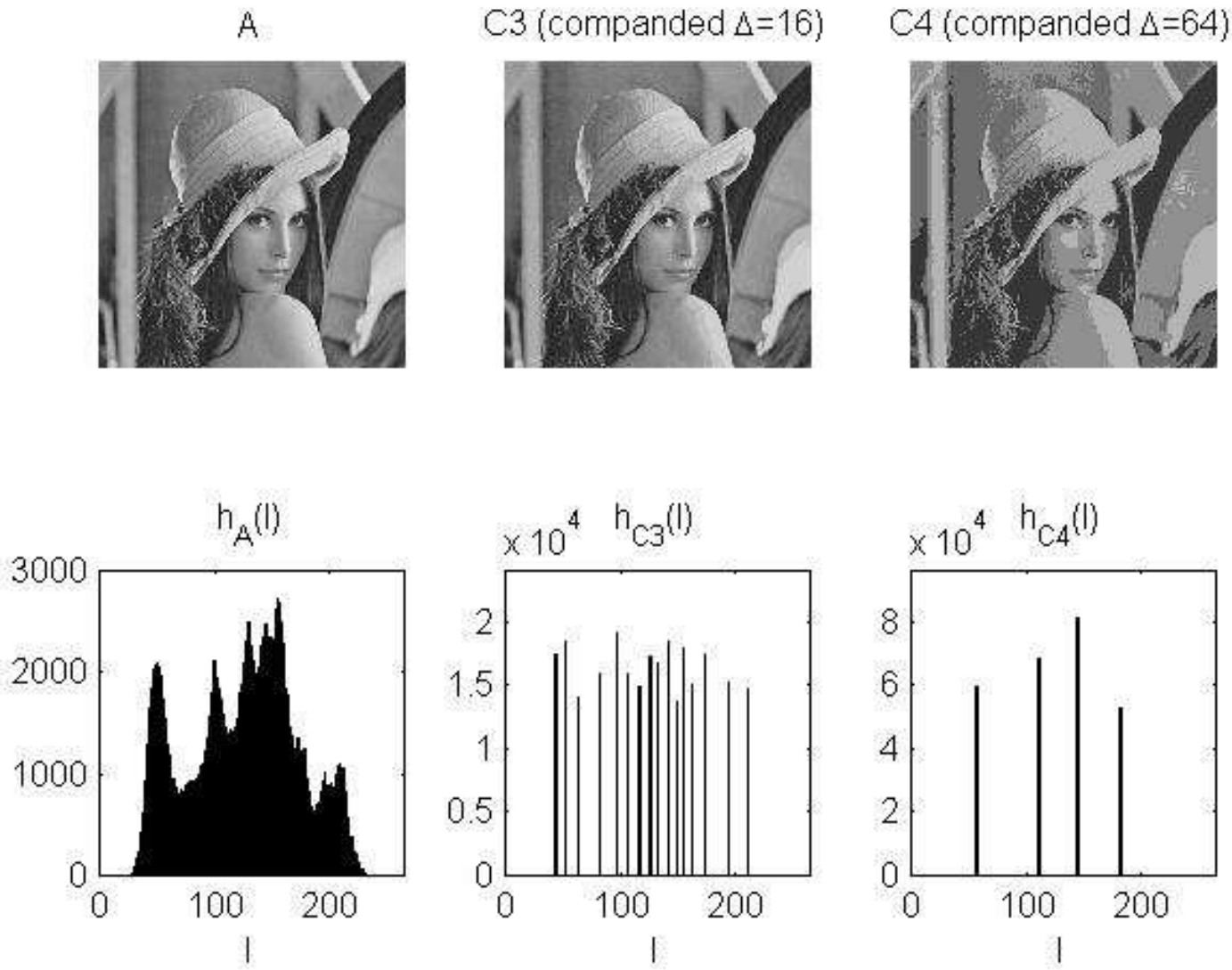
Funciones de Companding en Cuantizador de Punto



Ejemplo


A
C1 (companded Δ=4)
C2 (companded Δ=8)
h_A(I)
h_C1(I)
h_C2(I)
3000
2000
1000
0
6000
4000
2000
0
10000
5000
0
0
100
200
I

Ejemplo - cont.



Ejemplo - cont.

Uniform $\Delta = 16$

Companded $\Delta = 16$

Δ	Imagen Companded	MSQE
4	C1	1.56
8	C2	4.28
16	C3	13.84
64	C4	186.27

Comparado con los resultados de cuantización uniforme.

Resumen

- En esta clase se aprendió a generar imagenes aleatorias.
- Se aprendió lo referente a empatamiento de histograma lo que nos permite "empatar" el histograma de una imagen dada al histograma de otra imagen.
- Se aprendió a calcular las "inversas" de funciones discretas.
- Se aprendió sobre cuantización, cuantización uniforme simple y companded.
  - Cálculo de errores.
  - Estadísticas de cuantización simples.
  - Elección óptima de umbrales.
  - Leer las páginas 243-244, 99-118 del libro de texto.

Tarea IV

1. Generater una matriz $\mathbf{A}$ de 256×256 salidas de una variable aleatoria continua Gaussiana de amplitud con $\mu = 2$ y $\sigma^2 = 3$. Calcule su media y varianza de muestra. Note que $\mathbf{A}$ no es una matriz de imagen. Normalice $\mathbf{A}$ para obtener $\mathbf{B}$. Calcule la media, varianza, función de probabilidad de masa e histograma de muestra de $\mathbf{B}$. Muestre $\mathbf{B}$ y todos los valores calculados.

2. Utilizando modificación de histograma, modifique *su imagen* de manera que la imagen resultante tenga un histograma que se empata con $h_B(l)$ como en la pregunta 1. Muestre su imagen, la imagen modificada, sus histogramas y la función de acoplamiento de punto. Compare los histogramas de la imagen modificada y $h_B(l)$.

3. Haga el procesamiento del "ejemplo deshacer" en su imagen.

4. Cuantize uniformemente su imagen utilizando $\Delta = 4, 8, 16, 64$. Muestre la imagen cuantizada, su histograma y MSQE en cada caso.

5. Compande su imagen utilizando $\Delta = 4, 8, 16, 64$. Muestre la imagen cuantizada, su histograma y MSQE en cada caso. Compare los resultados con los obtenidos en el inciso 4.

6. $p_A(l) = [.1, 0, .3, .2, 0, 0, .3, .1]$ y $p_B(k) = [.2, 0, 0, .1, .4, .3]$ para dos *imágenes* $\mathbf{A}$ y $\mathbf{B}$. Calcule una función de punto $g(l)$ tal que $C(i, j) = g(A(i, j))$ tenga histograma $h_C(l)$ que se "empate" con $h_B(l)$. Suponga que todas las imágenes tienen un total de 10 pixeles. Calcule el histograma $h_C(l)$.

[1] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice Hall, 1989.

Cuantización

- $t_n \in \{0, 1, \ldots, 255\}$ una secuencia de umbrales $(n = 0, \ldots, P-1)$.
- P "intervalos discretos medio-abiertos" $R_n = [t_n, t_{n+1})$ $(t_0 = 0,\ t_P = 256)$.
- $r_n \in R_n$ el nivel de reproducción del intervalo R_n.
- Cuantizador:

$$Q(l) = \{r_k | l \in R_k,\ k = 0, \ldots, P-1\} \tag{43}$$

  i.e., $l \in R_k \Leftrightarrow Q(l) = r_k$.

Diseñando Buenos Cuantizadores

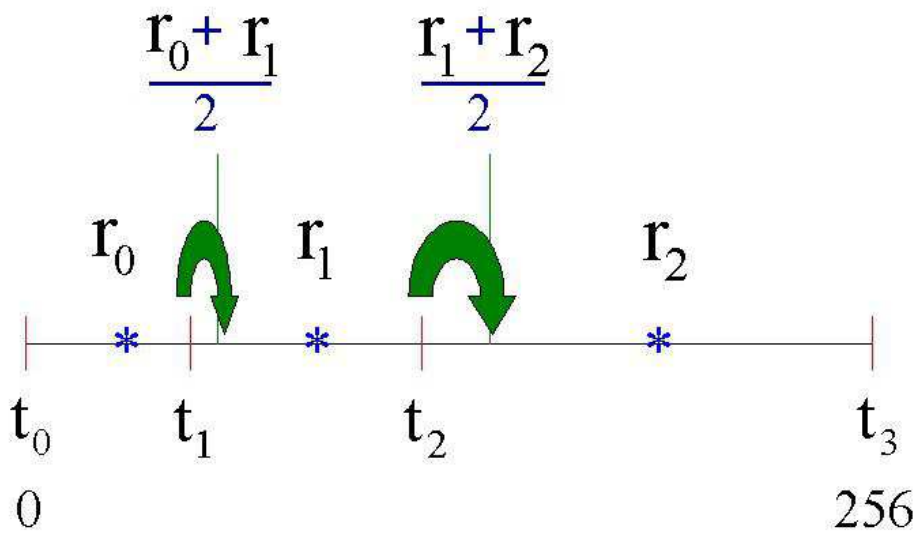
$$Q(l) = \{r_k | l \in R_k, \ k = 0, \ldots, P-1\} \tag{44}$$

$$\text{MSQE} = \sum_{l=0}^{255} (l - Q(l))^2 p_A(l) \tag{45}$$

Suponiendo que P es fija:

- Los intervalos alrededor de l donde $p_A(l)$ es grande, un buen cuantizador debería tener varios R_n pequeños, i.e., puesto que se pueden tener máximo P intervalos discretos, muchos de estos intervalos deberían estar alrededor de l en donde $p_A(l)$ es grande.
- En forma equivalente, un buen cuantizador no debería "desperdiciar" varios niveles de reproducción en intervalos alrededor de l donde $p_A(l)$ es *pequeña*.

Diseño de Umbrales para Niveles de Reproducción Dados



- Suponga que P y r_n son dados.
- Para optimizar el MSQE

$$t_n = \text{round}(\frac{r_{n-1} + r_n}{2}) \tag{46}$$

Diseño de Niveles de Reproducción para Umbrales Dados

- Suponga que se dan P y t_n.
- Considere $R_n = [t_n, t_{n+1})$.
- ¿Cómo podemos elegir $r_n \in R_n$ para minimizar el MSQE?
- La ecuación 45 puede escribirse como:

$$\begin{aligned}
\text{MSQE} &= \sum_{l=0}^{255}(l - Q(l))^2 p_A(l) = \sum_{m=0}^{P-1}\sum_{l=t_m}^{t_{m+1}-1}(l - Q(l))^2 p_A(l) \\
&= \sum_{m=0}^{P-1}\sum_{l=t_m}^{t_{m+1}-1}(l - r_m)^2 p_A(l) \\
&= \sum_{m=0, m\neq n}^{P-1}\sum_{l=t_m}^{t_{m+1}-1}(l - r_m)^2 p_A(l) + \sum_{l=t_n}^{t_{n+1}-1}(l - r_n)^2 p_A(l)
\end{aligned}$$

- Elegir r_n para minimizar:

$$\sum_{l=t_n}^{t_{n+1}-1}(l - r_n)^2 p_A(l) \tag{47}$$

Diseño de Niveles de Reproducción cont.

- Elegir r_n para minimizar:

$$\sum_{l=t_n}^{t_{n+1}-1} (l - r_n)^2 p_A(l)$$

- Obteniendo la derivada con respecto a r_n e igualando el resultado a 0 da:

$$-\sum_{l=t_n}^{t_{n+1}-1} 2(l - r_n) p_A(l) = 0$$

$$r_n = \text{round}\left(\frac{\sum_{l=t_n}^{t_{n+1}-1} l p_A(l)}{\sum_{k=t_n}^{t_{n+1}-1} p_A(k)}\right) \tag{48}$$

MSQE Optimo Cuantizador Lloyd-Max

Algoritmo: Encuentre el cuantizador óptimo para MSQE en *iteraciones* $v = 0, 1, \ldots$. Inicie con $v = 0$ y un conjunto de niveles de reproducción dado r_n^0. Sea $\epsilon > 0$ un número *pequeño*.

1. $v \to v + 1$.
2. Calcule MSQE óptimo t_n^v vía:

$$t_n^v = \text{round}\left(\frac{r_{n-1}^{v-1} + r_n^{v-1}}{2}\right) \tag{49}$$

3. Calcule MSQE óptimo r_n^v vía:

$$r_n^v = \text{round}\left(\frac{\sum_{l=t_n^v}^{t_{n+1}^v - 1} l p_A(l)}{\sum_{k=t_n^v}^{t_{n+1}^v - 1} p_A(k)}\right) \tag{50}$$

4. Calcule MSQE vía:

$$e^v = \sum_{m=0}^{P-1} \sum_{l=t_m^v}^{t_{m+1}^v - 1} (l - r_m^v)^2 p_A(l) \tag{51}$$

5. If $e^{v-1} - e^v < \epsilon$ terminate (cuantizador diseñado) else goto 1.

Propiedades

- $e^{v-1} - e^v \geq 0$, el MSQE *es no-creciente con* v. Se *garantiza* que el algoritmo converge.
- El cuantizador MSQE óptimo final satisface las condiciones de optimización para los umbrales y niveles de reproducción.
- El cuantizador MSQE óptimo final es *óptimo localmente*, i.e., el cuantizador final (y por tanto e^v) depende del valor inicial r_n^0 con el que se comenzó.
- El cuantizador final debe ser implementado de *forma general* como se describió en la Clase 4.
- Una buena manera de elegir el r_n^0 es tomar los niveles de reproducción de un cuantizador "companding".

Ejemplo

Companded, P=64, MSQE=1.14

Lloyd-Max, P=64, MSQE=1.03

Companded, P=32, MSQE=3.80

Lloyd-Max, P=32, MSQE=3.65

Ejemplo - cont.

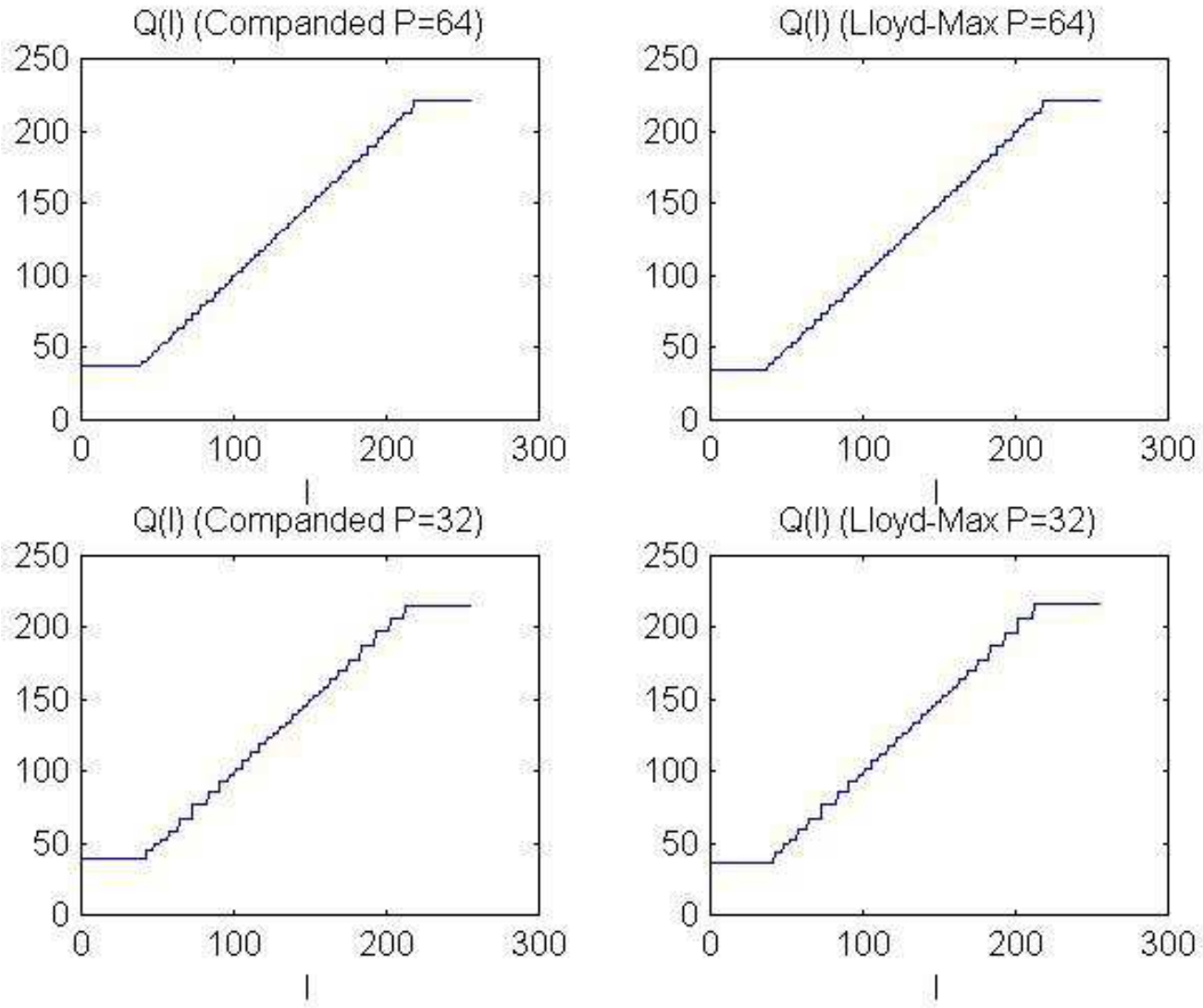
Companded, P=16, MSQE=13.15

Lloyd-Max, P=16, MSQE=12.36

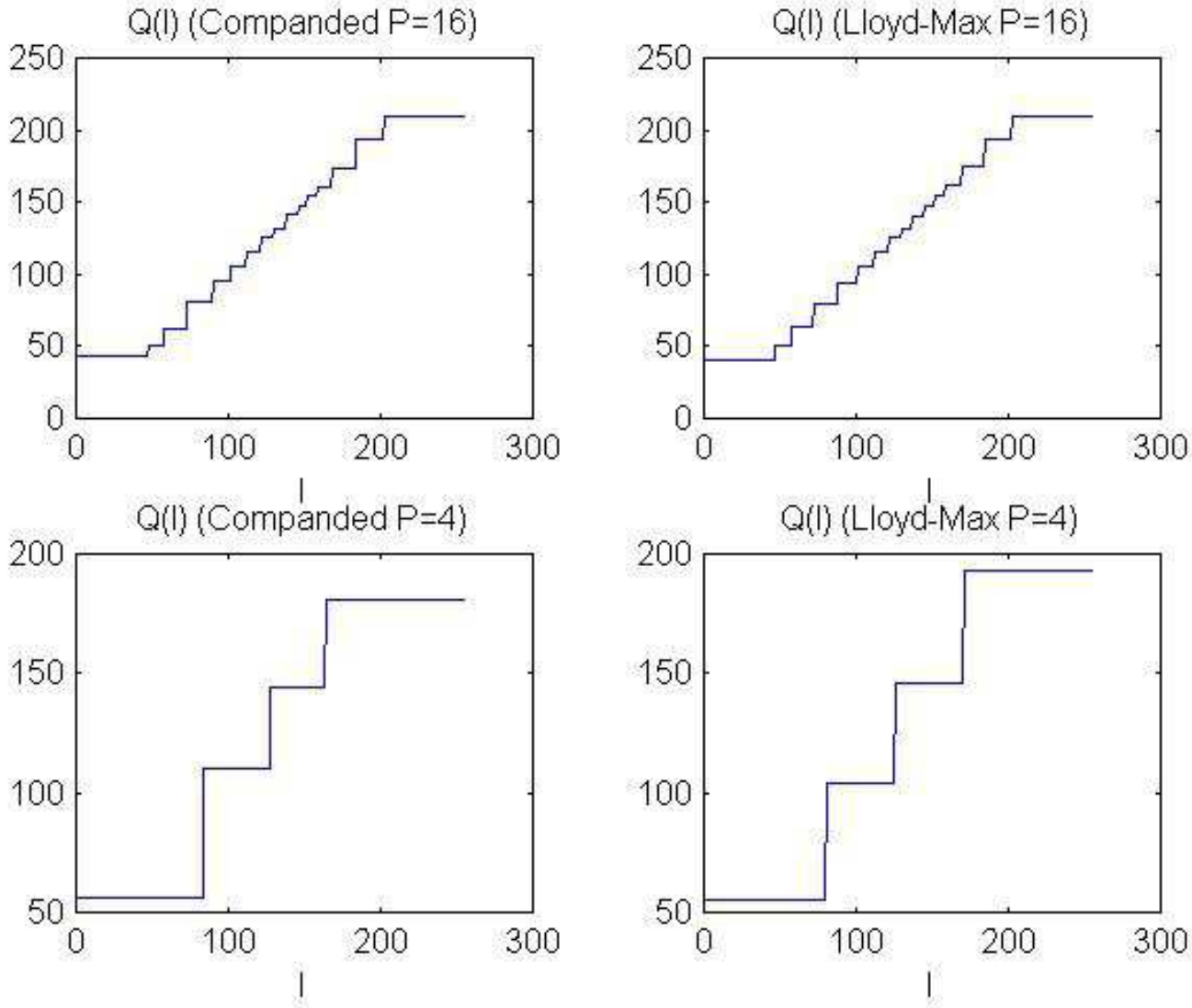
Companded, P=4, MSQE=186.04

Lloyd-Max, P=4, MSQE=161.91

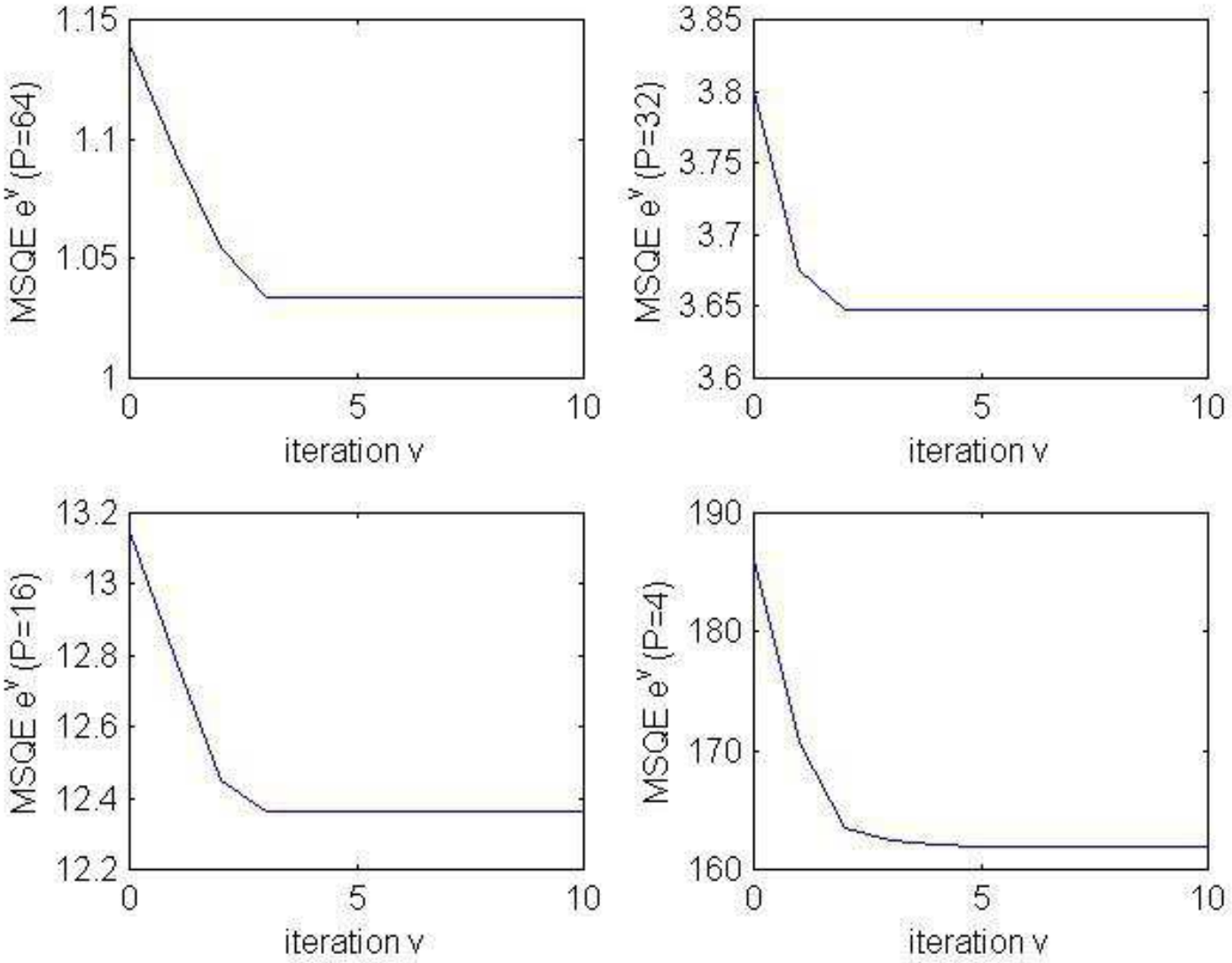
Ejemplo - cont.



Ejemplo - cont.



Ejemplo - cont.



Sistemas

- Las operaciones de procesamiento de imágenes pueden ser modeladas utilizando la teoría de sistemas.
- Por ahora supongamos que $\mathbf{A}$ y $\mathbf{B}$ son secuencias generales bidimensionales.
- Considere $\mathbf{A} \overset{\mathcal{S}}{\Rightarrow} \mathbf{B}$.
  - $\mathcal{S}$ es un sistema que "convierte" la entrada $\mathbf{A}$ en la salida $\mathbf{B}$.
  - La salida $\mathbf{B}$ *depende* de la entrada $\mathbf{A}$, i.e., en general diferentes entradas *pueden* dar origen a diferentes salidas.
  - El sistema $\mathcal{S}$ está caracterizado por sus relaciones de entrada-salida $(\mathcal{H})$:

$$\mathbf{B} = \mathcal{H}(\mathbf{A}) \tag{52}$$

Podemos también escribir:

$$B(m,n) = \mathcal{H}(\mathbf{A}(i,j)), \qquad m,n,i,j \in \{-\infty,\ldots,-1,0,1,\ldots,+\infty\}$$

Sistemas Lineales

- Sean a_1 y a_2 constantes.
- Un sistema lineal $\mathcal{S}$ es un sistema tal que

$$\mathcal{H}(a_1\mathbf{A_1} + a_2\mathbf{A_2}) \;=\; a_1\mathcal{H}(\mathbf{A_1}) + a_2\mathcal{H}(\mathbf{A_2}) \tag{53}$$

  para todo $a_1,\ a_2,\ \mathbf{A_1},\ \mathbf{A_2}$.
- Un gran número de operaciones de procesamiento/formación de imágenes puede ser *modelado* por sistemas lineales.

Sistemas Lineales Invariantes al Corrimiento (LSI)

- Un sistema invariante al corrimiento $\mathcal{S}$ es un sistema tal que si

$$B(m,n) = \mathcal{H}(A(i,j))$$

entonces

$$B(m-i_0, n-j_0) = \mathcal{H}(A(i-i_0, j-j_0)) \tag{54}$$

para todo $\mathbf{A}$ y $i_0, j_0 \in \{-\infty, \ldots, -1, 0, 1, \ldots, +\infty\}$.

- Un sistema lineal invariante al corrimiento (LSI) $\mathcal{S}$ es un sistema lineal que es *además* invariante al corrimiento.
- Muchas operaciones de procesamiento/formación de imágenes pueden ser *modeladas* por sistemas lineales invariantes al corrimiento.

Representación por Impulsos de Secuencias/Imágenes 2D

$$A(i,j) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)\delta(i-k, j-l) \tag{55}$$

donde

$$\delta(i,j) = \begin{cases} 1, & i = j = 0 \\ 0, & \text{cualquier otro} \end{cases} \tag{56}$$

es la función delta de Kronecker o función impulso de tiempo-discreto,

$$\delta(i,j) = \delta(i)\delta(j)$$

(Esta no debe confundirse con la función delta de Dirac o función impulso de tiempo-continuo $\delta(x,y)$).

Respuesta al Impulso de Sistemas LSI y Convolución

Suponiendo que $\mathcal{S}$ es un sistema lineal invariante al corrimiento con relación de entrada-salida $\mathcal{H}$. Usando la Ecuación 55:

$$\begin{aligned}\mathcal{H}(A(i,j)) &= \mathcal{H}(\sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)\delta(i-k,j-l)) \\ &= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)\mathcal{H}(\delta(i-k,j-l)) \qquad (57)\end{aligned}$$

Sea $h(i,j) = \mathcal{H}(\delta(i,j))$ que denota la respuesta al impulso del sistema $\mathcal{S}$. Entonces:

$$\mathcal{H}(\delta(i-k,j-l)) = h(i-k,j-l)$$

y se tiene:

$$\mathcal{H}(A(i,j)) = \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)h(i-k,j-l) \qquad (58)$$

la cual es llamada la convolución de suma.

- Esta se mantiene para toda $\mathbf{A}$, i.e., todo el sistema LSI $\mathcal{S}$ esta "dentro de" $h(i,j)$.

Convolución

$$B(i,j) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)h(i-k, j-l)$$

También se escribirá como: $\mathbf{B} = \mathbf{A} \otimes \mathbf{h}$.
Sea $k' = i - k,\ \ l' = j - l$.

$$\begin{aligned} \mathbf{A} \otimes \mathbf{C} &= \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)C(i-k, j-l) \\ &= \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} A(i-k', j-l')C(k', l') \\ &= \mathbf{C} \otimes \mathbf{A} \end{aligned}$$

En particular, $\delta \otimes \mathbf{h} = \mathbf{h} \otimes \delta = \mathbf{h}$ y para toda $\mathbf{A}$:

$$\begin{aligned} \mathbf{A} \otimes \delta &= \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)\delta(i-k, j-l) \\ &= \mathbf{A} \end{aligned} \tag{59}$$

Convolución - cont.

- en la práctica estamos interesados en la convolución de secuencias de dos dimensiones de longitud finita, por e.g.,

$$A(i,j) \quad \begin{cases} \neq 0, & 0 \leq i \leq N_1 - 1,\ 0 \leq j \leq M_1 - 1 \\ = 0 & \text{cualquier otro} \end{cases}$$

$$B(i,j) \quad \begin{cases} \neq 0, & 0 \leq i \leq N_2 - 1,\ 0 \leq j \leq M_2 - 1 \\ = 0 & \text{cualquier otro} \end{cases}$$

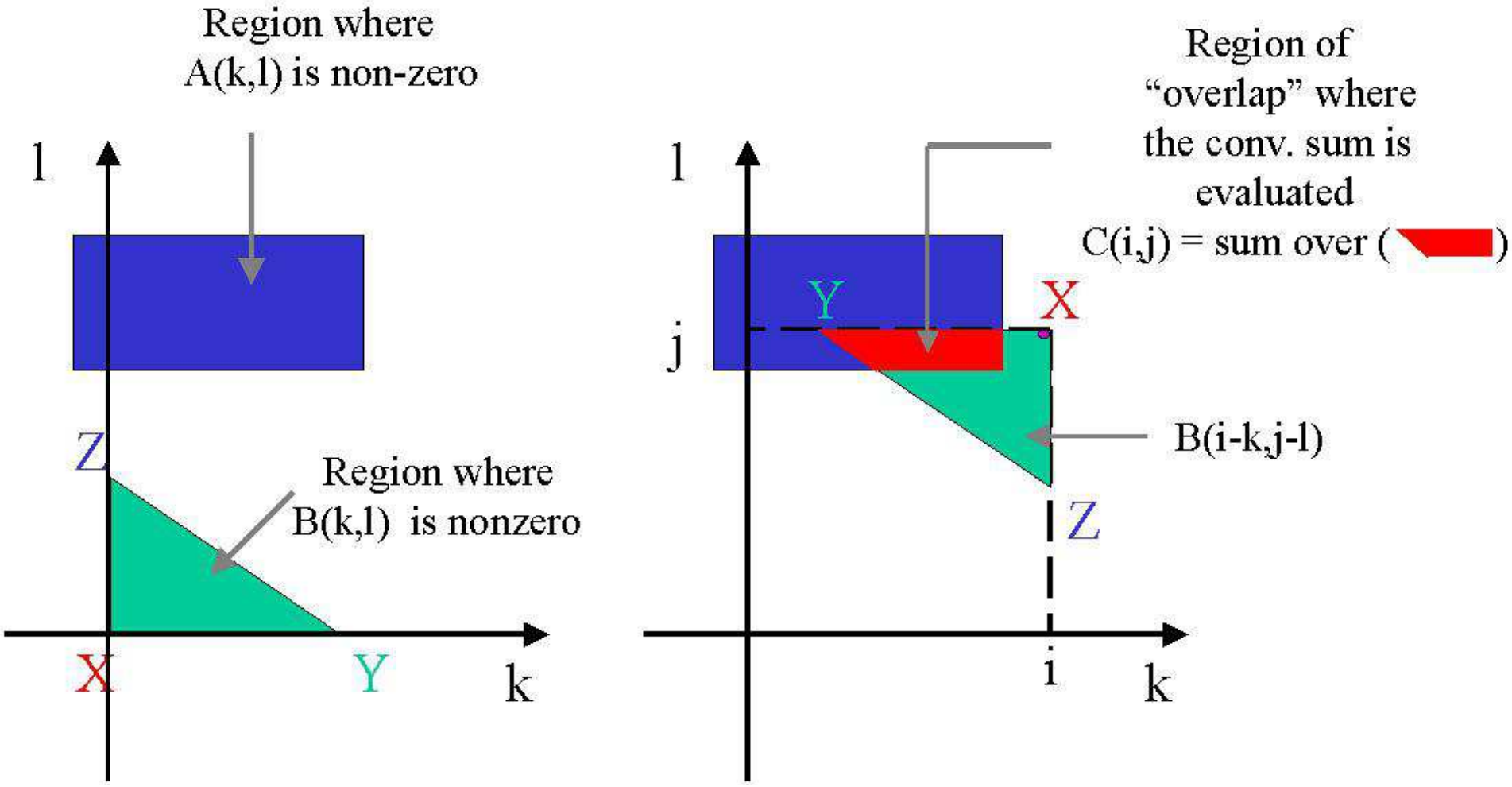
- Sea $\mathbf{A}$ y $\mathbf{B}$ como arriba.

$$\begin{aligned} \mathbf{A} \otimes \mathbf{B} &= \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l) B(i-k, j-l) \\ &= \sum_{k=0}^{N_1-1} \sum_{l=0}^{M_1-1} A(k,l) B(i-k, j-l) \end{aligned} \tag{60}$$

  Note que la convolución de suma previa incluye el término $A(N_1 - 1, M_1 - 1)B(i - (N_1 - 1), j - (M_1 - 1))$.

- Entonces si $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$, $\mathbf{A}$ es $N_1 \times M_1$ y $\mathbf{B}$ es $N_2 \times M_2$, entonces $\mathbf{C}$ es en general $(N_1 + N_2 - 1) \times (M_1 + M_2 - 1)$.

Ejemplo I


Region where
A(k,l) is non-zero
l
Z
Region where
B(k,l) is nonzero
X
Y
k
Region of
"overlap" where
the conv. sum is
evaluated
C(i,j) = sum over ( )
l
Y
X
j
B(i-k,j-l)
Z
i
k

Convolución - cont.

La ecuación 60 puede ser aún más simplificada en varias formas así que las sumas de 0 se evitarán para propósitos de implementación.

- $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$. $\mathbf{A}$ $(N_1 \times M_1)$ y $\mathbf{B}$ $(N_2 \times M_2)$.
  Suponga $N_2 << N_1,\ M_2 << N_2$ y sea

$$x_i = \begin{cases} i - (N_2 - 1), & i > N_2 - 1 \\ 0, & \text{cualquier otro} \end{cases}$$

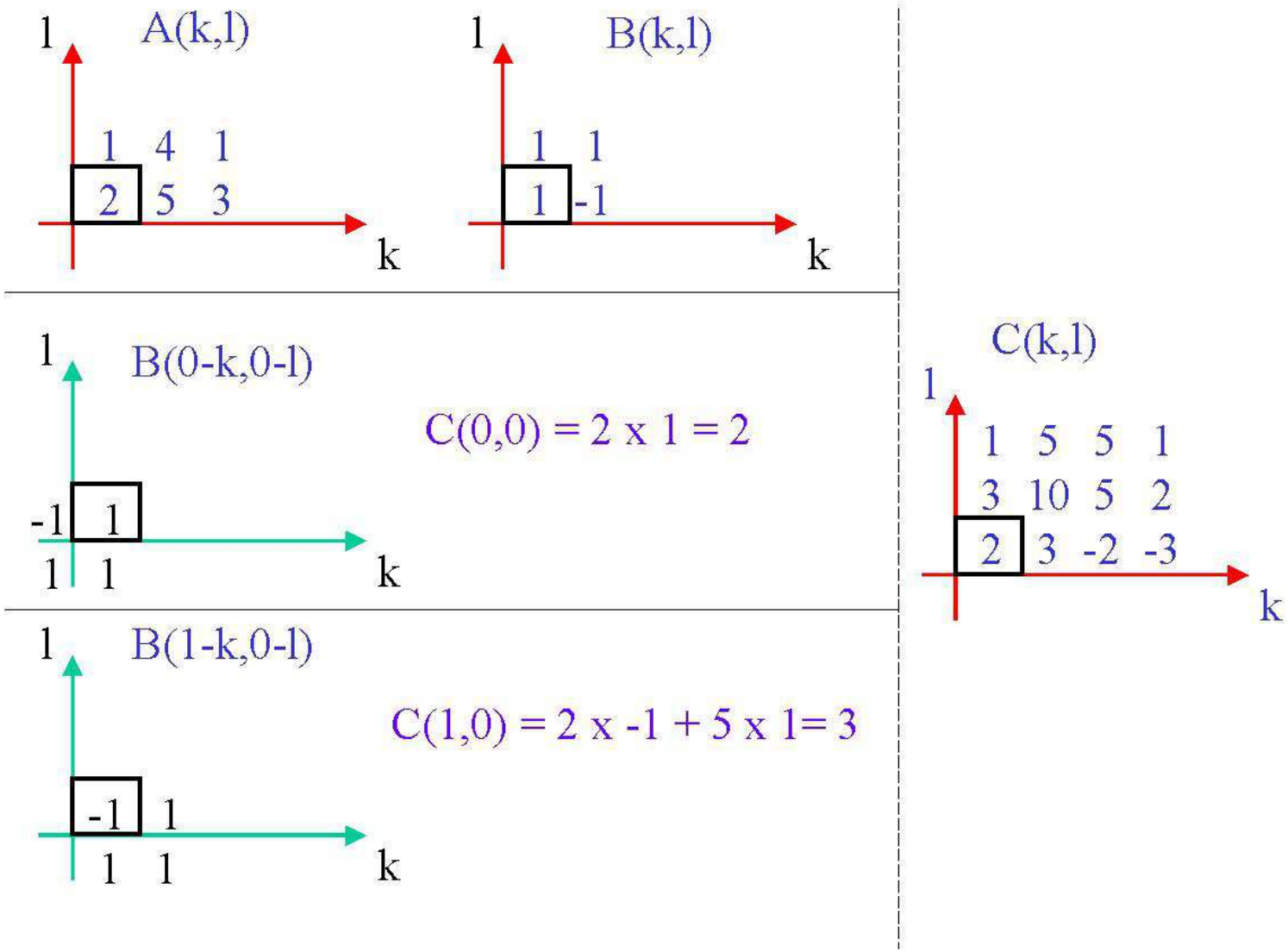
$$y_i = \begin{cases} N_1 - 1, & i > N_1 - 1 \\ i, & \text{cualquier otro} \end{cases}$$

  (Reemplace i con j, N con M para obtener $x_j,\ y_j$).

$$C(i,j) = \sum_{k=x_i}^{y_i} \sum_{l=x_j}^{y_j} A(k,l) B(i-k, j-l) \tag{61}$$

- Note que $i = 0, \ldots, N_1 + N_2 - 1 - 1$ y $j = 0, \ldots, M_1 + M_2 - 1 - 1$ para este *caso especial* de secuencias de extensión finita.

Ejemplo II


A(k,l)
1 4 1
2 5 3
B(k,l)
1 1
1 -1
B(0-k,0-l)
C(0,0) = 2 x 1 = 2
B(1-k,0-l)
C(1,0) = 2 x -1 + 5 x 1= 3
C(k,l)
1 5 5 1
3 10 5 2
2 3 -2 -3
k
l

Resumen

- En esta clase aprendimos como seleccionar los niveles de reproducción para umbrales dados.
- Aprendimos como diseñar cuantizadores MSQE óptimos.
- Repasamos los sistemas lineales, sistemas lineales invariantes al corrimiento y la convolución de suma.
- Leer páginas 11-19 del libro de texto.

Tarea V

1. Implemente el cuantizador Lloyd-Max para su imagen. Repita todo lo visto entre las páginas 9-13. Muestre todos los resultados así como su imagen original y su función de probabilidad de masa de muestra.

2. Obtenga la convolución de las secuencias 2-D con porciones diferentes de cero mostrada, i.e., obtenga $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$. Muestre sus cálculos gráficamente para al menos 5 valores diferentes de $\mathbf{C}$ similares al ejemplo previo.

$A(i,j)=$

	$j=0$	1	2
$i=0$	0	4	4
1	-2	1	3
2	5	1	1

,

$B(i,j)=$

	$j=0$	1	2
$i=0$	-1	3	4
1	1	2	2
2	-3	2	6

3. Implemente un script de convolución en Matlab tome en cuenta la simplificación computacional que se discutió. Obtenga la convolución de su imagen con *ella misma*. Normalice el resultado y muestrelo como imagen. Comente el resultado así como el tiempo de ejecución. ¿Cuál es la dimensión de la imagen resultante? Ahora haga lo mismo utilizando el mando `conv2` en Matlab. Asegurese que el resultado de su script y `conv2` sean el mismo. (Puede hacerlo calculando una imagen error y sumandole su valor absoluto. El resultado debería ser 0. Experimente con imágenes pequeñas hasta que obtenga el resultado correcto.)

Tarea V cont.

4. Sea

$$A(i,j) = \begin{array}{|c||c|c|c|}\hline & j=0 & 1 & 2 \\ \hline\hline i=0 & -1 & 0 & 1 \\ \hline 1 & -2 & 0 & 2 \\ \hline 2 & -1 & 0 & 1 \\ \hline \end{array}$$

Obtenga la convolución de su imagen con **A** (su script). Tome el valor absoluto del resultado, normalice y muestre. Comente el resultado y el tiempo de ejecución. ¿Cuál es el tamaño de la imagen resultante? Ahora haga lo mismo con `conv2` como arriba.

Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

Sistemas LSI y Convolución

$\mathcal{S}$ es un sistema lineal invariante al corrimiento con relación de entrada-salida $\mathcal{H}$.

$$\begin{aligned}\mathcal{H}(A(i,j)) &= \mathcal{H}(\sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)\delta(i-k,j-l)) \\ &= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)\mathcal{H}(\delta(i-k,j-l))\end{aligned}$$

$h(i,j) = \mathcal{H}(\delta(i,j))$ la respuesta al impulso del sistema $\mathcal{S}$.

$$\begin{aligned}\mathcal{H}(\delta(i-k,j-l)) &= h(i-k,j-l) \\ \mathcal{H}(A(i,j)) &= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)h(i-k,j-l)\end{aligned}$$

la cual es la convolución de suma.

- Todo respecto al sistema LSI $\mathcal{S}$ esta "dentro de" $h(i,j)$.

Convolución

$\mathbf{B} = \mathbf{A} \otimes \mathbf{h}$:

$$B(i,j) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)h(i-k, j-l) \tag{62}$$

Propiedades:

- $\mathbf{A} \otimes \mathbf{h} = \mathbf{h} \otimes \mathbf{A}$.
- $\mathbf{A} \otimes \delta = \mathbf{A}$.
- Secuencias $2-D$ de extensión finita $\mathbf{A}$ $(N_1 \times M_1)$, $\mathbf{h}$ $(N_2 \times M_2)$:
  (para e.g., $A(i,j) \neq 0,\ 0 \leq i \leq N_1 - 1,\ 0 \leq j \leq M_1 - 1$, etc.)
  - $\mathbf{C} = \mathbf{A} \otimes \mathbf{h}$ es $(N_1 + N_2 - 1) \times (M_1 + M_2 - 1)$.

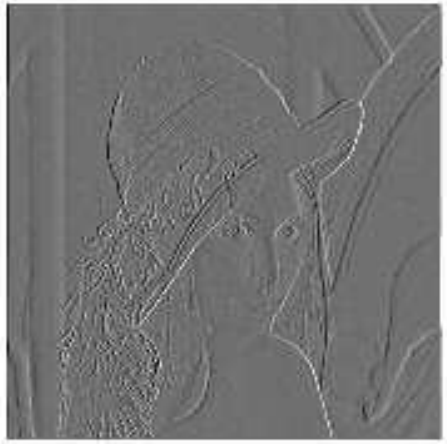
Convolución y Filtrado Lineal

- $\mathbf{B} = \mathbf{A} \otimes \mathbf{h}$
  - "$\mathbf{A}$ es convolucionada con $\mathbf{h}$ para producir $\mathbf{B}$".
  - "$\mathbf{A}$ es linealmente filtrada con $\mathbf{h}$ para producir $\mathbf{B}$".
- Claro que si se utiliza $\mathbf{A} \otimes \mathbf{h} = \mathbf{h} \otimes \mathbf{A}$ se puede decir:
  - "$\mathbf{h}$ es convolucionada con $\mathbf{A}$ para producir $\mathbf{B}$".
  - "$\mathbf{h}$ is linealmente filtrada con $\mathbf{A}$ para producir $\mathbf{B}$".
- Podemos resolver muchos problemas interesantes de procesamiento de imágenes eligiendo adecuadamente el "filtro" $\mathbf{h}$ y filtrando la imagen $\mathbf{A}$.

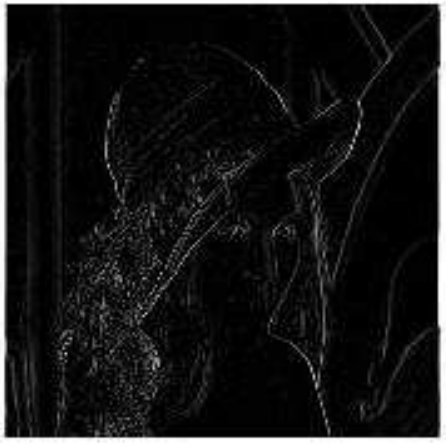
Ejemplo

A

B=A ⊗ h (normalized)



C=abs(A ⊗ h) (normalized)



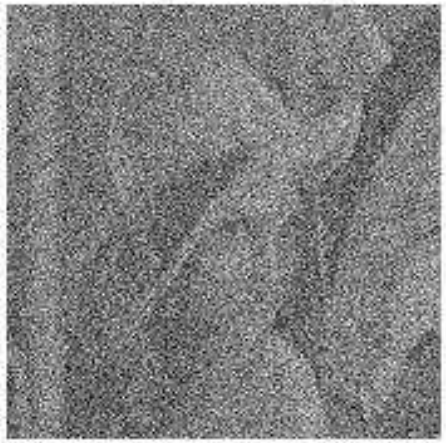
image((C>T)*255) (T=25)

Ejemplo - cont.

A

A ⊗ g (normalized)

B=A+100*randn(512,512) (normalized)



B ⊗ g (normalized)

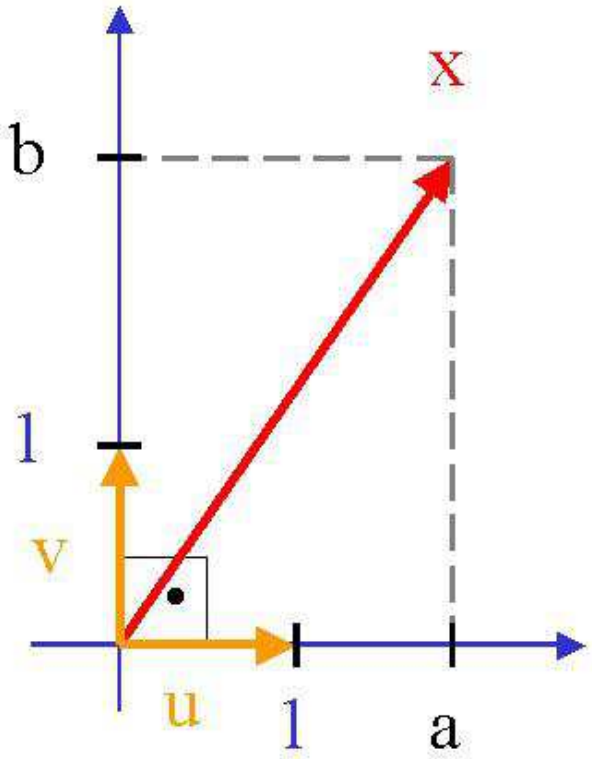
La Transformada de Fourier de Secuencias 2-D

Ahora repasaremos la Transformada de Fourier de Secuencias 2-D.
Motivación:

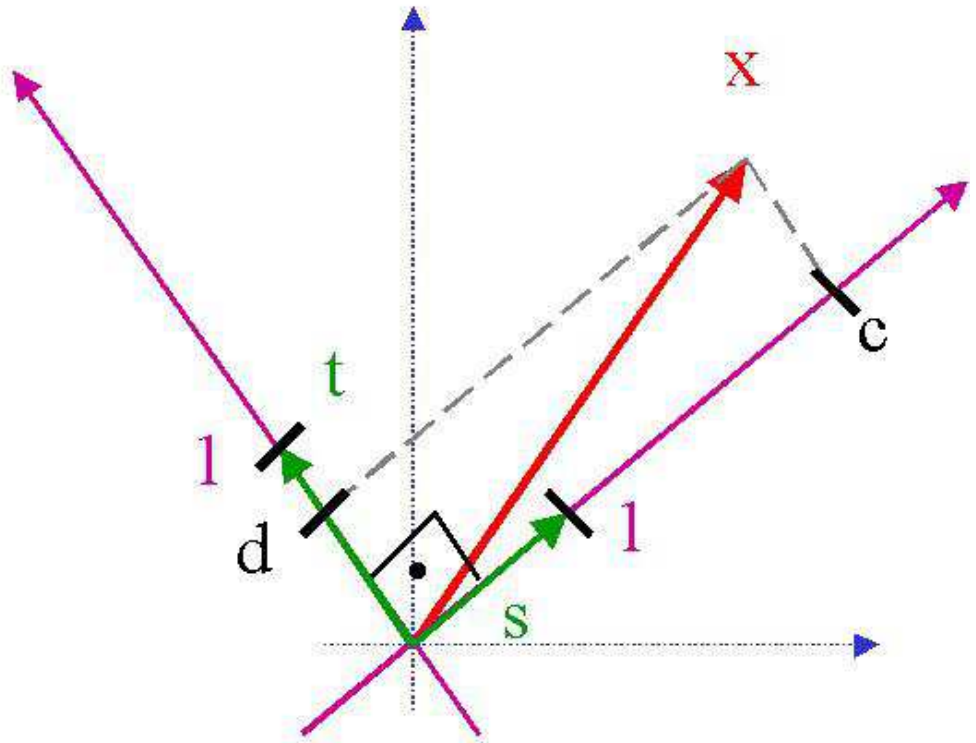
- La operación de convolución toma una forma muy especial en el "dominio" de la transformada de Fourier 2-D.
- La transformada de Fourier 2-D de imágenes revelará propiedades interesantes que son compartidas por muchas imágenes.
  - Esto nos permitirá distinguir entre imágenes naturales y "no-imágenes" (tales como ruido).
  - Podremos decir que el "tipo" de filtro lineal es bueno para cierta aplicación de procesamiento.
- Los efectos de operaciones de muestreo son comprendidos de forma más clara en el "dominio" de la transformada de Fourier 2-D.
- En esta clase hablaremos principalmente de las "herramientas" requeridas.

Intuición - Sistema Ortogonal de Coordenadas

Original Coordinate System



Rotated Coordinate System



u , v , s , t : unit vectors

x = au + bv = cs + dt

Definición

La Transformada de Fourier de una secuencia 2-D $\mathbf{A}$, $\mathcal{F}(\mathbf{A})$ se define como:

$$\begin{aligned}\mathcal{F}(\mathbf{A}) &= F_A(w_1, w_2) \\ &= \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} A(m,n) e^{-j(mw_1+nw_2)} \qquad -\pi \leq w_1, w_2 < \pi \end{aligned} \tag{63}$$

$\mathbf{A}$ puede ser *recuperada* de su transformada $F_A(w_1, w_2)$
vía la Transformada de Fourier 2-D inversa $\mathcal{F}^{-1}(\mathbf{A})$:

$$\begin{aligned} A(m,n) &= \mathcal{F}^{-1}(\mathbf{A}) \\ &= \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F_A(w_1, w_2) e^{+j(mw_1+nw_2)} dw_1 dw_2 \end{aligned} \tag{64}$$

- $w_1,\ w_2$ varía en un "continuo", i.e., el intervalo $[-\pi, \pi)$.
- $e^{j(mw_1+nw_2)} = \cos(mw_1 + nw_2) + j\ \sin(mw_1 + nw_2)$.
- $\mathbf{A} \overset{\mathcal{F}}{\leftrightarrow} F_A$

Intuición - cont.

Continuando con la intuición previa, considere la representación de secuencias 2-D con impulsos:

$$A(k,l) = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} A(m,n)\delta(m-k, n-l)$$

y sus transformadas de Fourier:

$$F_A(w_1, w_2) = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} A(m,n)e^{-j(mw_1+nw_2)}$$

Estas son en realidad las representaciones de la misma secuencia $\mathbf{A}$ en dos sistemas de coordenadas ortogonales:

- El primer sistema de coordenadas tiene "vectores" base dados por $\delta(m-k, n-l)$.
- El segundo sistema de coordenadas tiene "vectores" base dados por $e^{-j(mw_1+nw_2)}$.
- Las sumas son productos punto o "escalares".

Partes Real-Compleja y Simetría

- En general $F_A(w_1, w_2)$ es un valor complejo.
- Debido a que estaremos considerando principalmente secuencias 2-d reales podemos notar algunas propiedades de simetría utilizando la relación de la transformada inversa de Fourier.

$$A(m,n) = \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_A(w_1,w_2)e^{+j(mw_1+nw_2)}dw_1dw_2$$

Si $\mathbf{A}$ es real entonces:

$$F_A(w_1,w_2) = F_A^*(-w_1,-w_2) \quad (65)$$

$$|F_A(w_1,w_2)| = |F_A(-w_1,-w_2)| \quad (66)$$

$$\angle\ F_A(w_1,w_2) = -\angle\ F_A(-w_1,-w_2) \quad (67)$$

$$\Re(F_A(w_1,w_2)) = \Re(F_A(-w_1,-w_2)) \quad (68)$$

$$\Im(F_A(w_1,w_2)) = -\Im(F_A(-w_1,-w_2)) \quad (69)$$

Periodicidad

$$F_A(w_1, w_2) = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} A(m,n) e^{-j(mw_1+nw_2)} \qquad -\pi \leq w_1, w_2 < \pi$$

- $F_A(w_1, w_2)$ es periódica en w_1, w_2 con periodo 2π, i.e., para todos los enteros k, l:

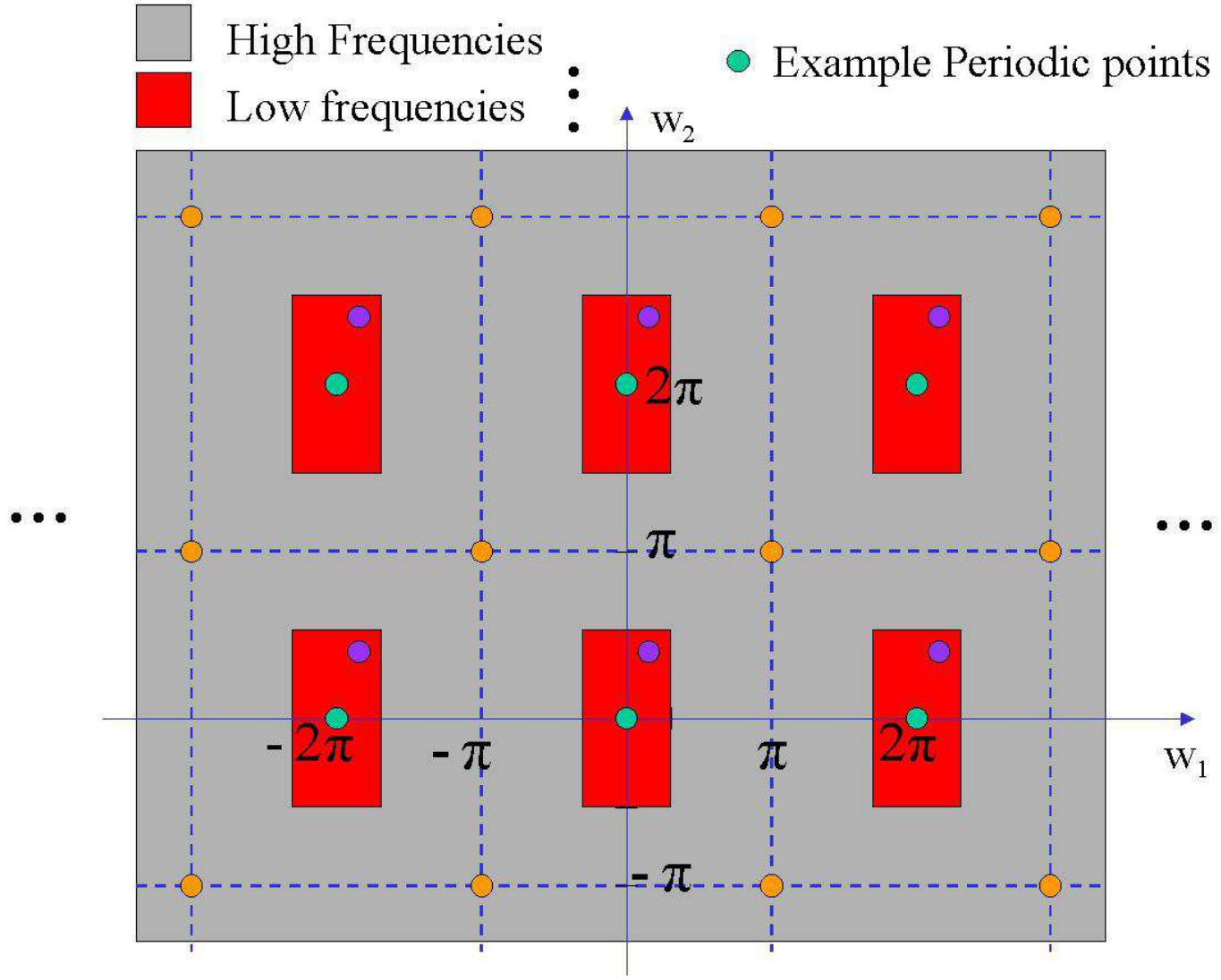
$$F_A(w_1 + k2\pi, w_2 + l2\pi) = F_A(w_1, w_2) \tag{70}$$

  Para ver esto considere:

$$\begin{aligned} e^{-j(m(w_1+k2\pi)+n(w_2+l2\pi))} &= e^{-j(mw_1+nw_2)} e^{-jk2\pi} e^{-jl2\pi} \\ &= e^{-j(mw_1+nw_2)} \quad \forall \text{ enteros } k, l \end{aligned}$$

- $e^{j(mw_1+nw_2)} = \cos(mw_1 + nw_2) + j \sin(mw_1 + nw_2)$.
  w_1, w_2 las frecuencias de las funciones trigonométricas periódicas.

Ejemplo


High Frequencies
Low frequencies
Example Periodic points
w2
w1
2π
π
- 2π
- π
π
2π
- π

Corrimiento y Modulación

- Corrimiento:

$$\begin{aligned}\mathcal{F}(A(m-m_0,n-n_0)) &= \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} A(m-m_0,n-n_0)e^{-j(mw_1+nw_2)} \\ &= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)e^{-j((k+m_0)w_1+(l+n_0)w_2)} \\ &= e^{-j(m_0w_1+n_0w_2)}\sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)e^{-j(kw_1+lw_2)} \qquad (71)\end{aligned}$$

$A(m-m_0,n-n_0) \overset{\mathcal{F}}{\leftrightarrow} e^{-j(m_0w_1+n_0w_2)}F_A(w_1,w_2)$.

- De forma similar, modulación:

$$e^{j(mw_{01}+mw_{02})}A(m,n) \overset{\mathcal{F}}{\leftrightarrow} F_A(w_1-w_{01},w_2-w_{02}) \qquad (72)$$

Producto Interno (Punto) y Conservación de Energía

- Conservación del producto interno:

$$\sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} A(m,n)B^*(m,n)$$

$$= \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} A(m,n)[\frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_B^*(w_1,w_2)e^{-j(mw_1+nw_2)}dw_1dw_2]$$

$$= \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}[\sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} A(m,n)e^{-j(mw_1+nw_2)}]F_B^*(w_1,w_2)dw_1dw_2$$

$$= \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_A(w_1,w_2)F_B^*(w_1,w_2)dw_1dw_2 \quad (73)$$

- Por tanto, conservación de energía:

$$\sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} |A(m,n)|^2 = \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} |F_A(w_1,w_2)|^2 dw_1dw_2 \quad (74)$$

Convolución

- Sea $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$.

$$
\begin{aligned}
C(m,n) &= \sum_{k=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A(k,l) B(m-k, n-l) \\
F_C(w_1, w_2) &= \sum_{k=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A(k,l) \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} B(m-k, n-l) e^{-j(mw_1+nw_2)} \\
&= \sum_{k=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A(k,l) F_B(w_1, w_2) e^{-j(kw_1+lw_2)} \\
&= F_B(w_1, w_2) \sum_{k=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A(k,l) e^{-j(kw_1+lw_2)} \\
&= F_A(w_1, w_2) F_B(w_1, w_2)
\end{aligned}
$$

en donde se utilizó la propiedad de corrimiento en el segundo paso del cálculo. Entonces tenemos el importante resultado:

$$\mathbf{A} \otimes \mathbf{B} \overset{\mathcal{F}}{\leftrightarrow} F_A(w_1, w_2) F_B(w_1, w_2) \tag{75}$$

Multiplicación

- Una propiedad dual de la propiedad de convolución se puede derivar para la multiplicación. Sea $C(m,n) = A(m,n)B(m,n)$.

$$\mathcal{F}(C(m,n)) = \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} A(m,n)B(m,n)e^{-j(mw_1+nw_2)}$$

$$= \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} A(m,n)[\frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_B(w_1',w_2')e^{j(mw_1'+nw_2')}dw_1'dw_2']e^{-j(mw_1+nw_2)}$$

$$= \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_B(w_1',w_2')[\sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} A(m,n)e^{-j(m(w_1-w_1')+n(w_2-w2'))}]dw_1'dw_2'$$

$$= \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_B(w_1',w_2')F_A(w_1-w_1',w_2-w_2')dw_1'dw_2'$$

Entonces:

$$A(m,n)B(m,n) \overset{\mathcal{F}}{\leftrightarrow} \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_B(w_1',w_2')F_A(w_1-w_1',w_2-w_2')dw_1'dw_2' \quad (76)$$

Funciones Delta

- La transformada de Fourier de una función delta de Kronecker es:

$$F_\delta(w_1, w_2) = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} \delta(m,n) e^{-j(mw_1+nw_2)} \quad (77)$$
$$= 1$$

- La transformada de Fourier de $A(m,n) = 1$ puede encontrarse vía la función delta de Dirac:

$$\frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} \delta(w_1, w_2) e^{j(mw_1+nw_2)} dw_1 dw_2 = \frac{1}{4\pi^2}$$

$$\Rightarrow A(m,n) = 1 \overset{\mathcal{F}}{\leftrightarrow} 4\pi^2 \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} \delta(w_1 - k2\pi, w_2 - l2\pi) \quad (78)$$

en donde $\delta(w_1, w_2)$ es la función delta de Dirac y utilizamos el hecho de que la transformada de Fourier tiene que ser periódica con 2π.

- Note que $\delta(w_1, w_2) = 0$ para $w_1, w_2 \neq 0$ y

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \delta(w_1, w_2) dw_1 dw_2 = 1 \quad (79)$$

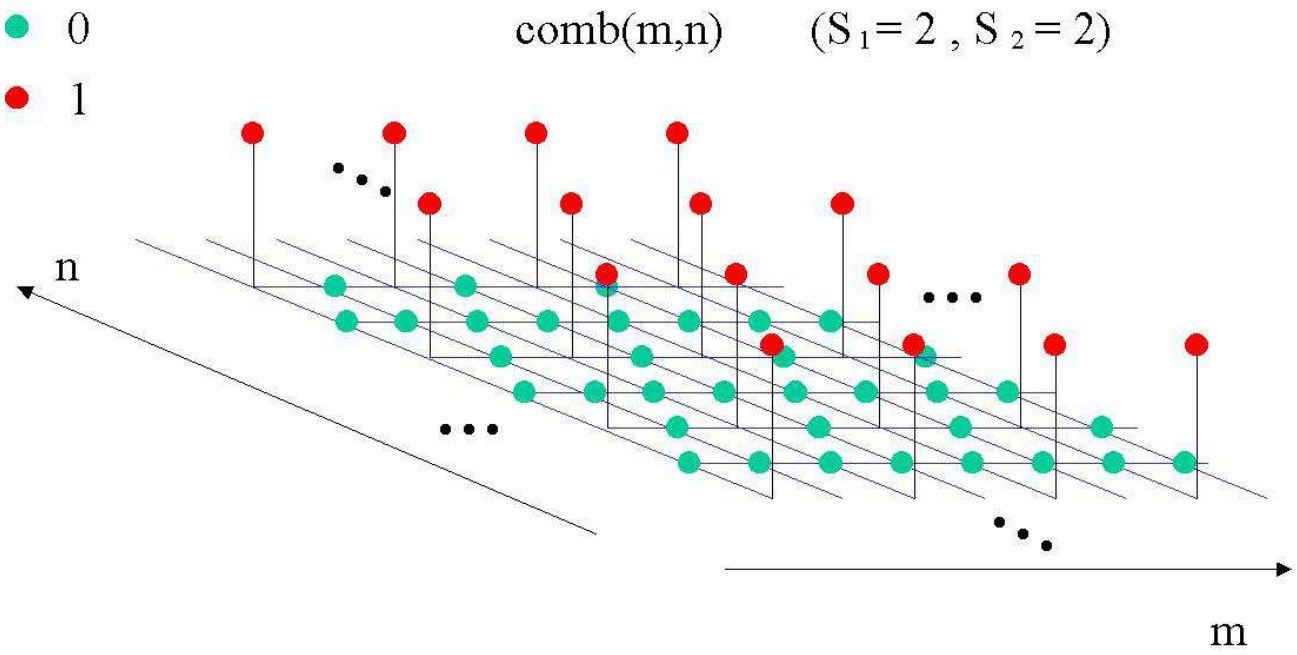
Función "peine" comb(m,n)

- Considere la función "peine" de Kronecker `comb(m,n)`:

$$\sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} \delta(m - kS_1, n - lS_2) \tag{80}$$

  en donde $S_1 > 0,\ S_2 > 0$ son enteros.

- `comb(m,n)` es muy util cuando se trabaja con muestreo.



comb(m,n) - cont.

La transformada de Fourier de una función "peine" puede calcularse como:

$$
\begin{aligned}
\mathcal{F}(comb(m,n)) &= \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} comb(m,n)e^{-j(mw_1+nw_2)} \\
&= \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty}\left[\sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty}\delta(m-kS_1,n-lS_2)\right]e^{-j(mw_1+nw_2)} \\
&= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty}\left[\sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty}\delta(m-kS_1,n-lS_2)e^{-j(mw_1+nw_2)}\right] \\
&= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} 1\ e^{-j(kS_1w_1+lS_2w_2)} \\
&= \sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty} 1\ e^{-j(mS_1w_1+nS_2w_2)} \qquad (81)
\end{aligned}
$$

comb(m,n) - cont.

Note que la Ecuación 81 es simplemente la transformada de Fourier de $A(m,n) = 1$ y por tanto:

$$\begin{aligned}
\mathcal{F}(comb(m,n)) &= \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} 1 \; e^{-j(mS_1w_1 + nS_2w_2)} \\
&= 4\pi^2 \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} \delta(S_1w_1 - k2\pi, S_2w_2 - l2\pi) \\
&= \frac{4\pi^2}{S_1S_2} \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} \delta\left(w_1 - \frac{k2\pi}{S_1}, w_2 - \frac{l2\pi}{S_2}\right) \qquad (82)
\end{aligned}$$

en donde la última línea se indica porque para cualquier función "regular" $G(w_1, w_2)$:

$$\begin{aligned}
\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \delta(S_1w_1 - k2\pi, S_2w_2 - l2\pi) G(w_1, w_2) dw_1 dw_2 &= \frac{1}{S_1S_2} G(k2\pi/S_1, l2\pi/S_2) \\
&= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \frac{1}{S_1S_2} \delta(w_1 - \frac{k2\pi}{S_1}, w_2 - \frac{l2\pi}{S_2}) G(w_1, w_2) dw_1 dw_2
\end{aligned}$$

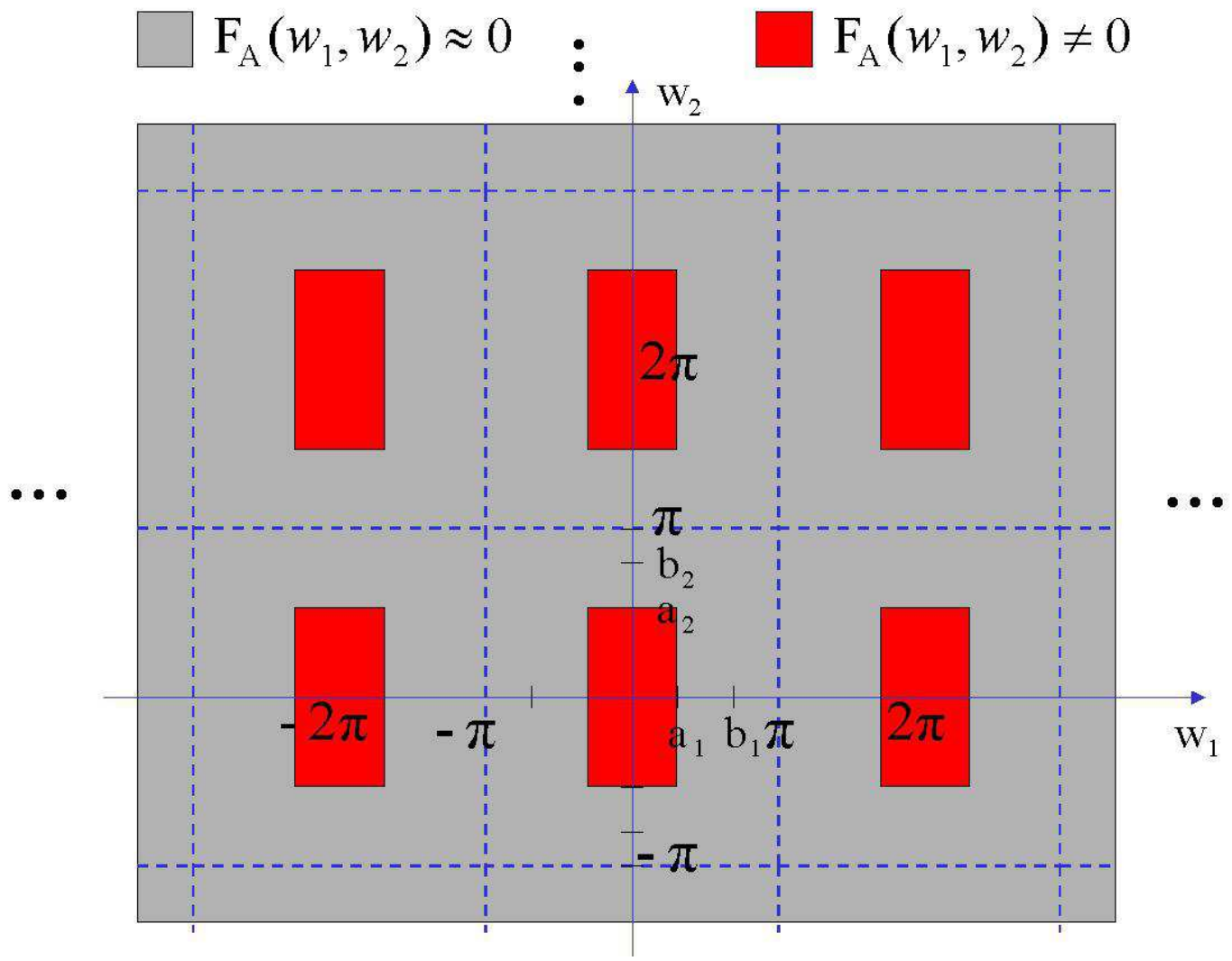
y las funciones delta de Dirac están definidas por integrales.

Tipos de Transformadas de Fourier

Sea $0 < a_1 < b_1 < \pi$ and $0 < a_2 < b_2 < \pi$.

- Se dirá que una transformada de Fourier $F_A(w_1, w_2)$ es pasa bajas si $|F_A(w_1, w_2)| \sim 0$ cuando $a_1 < |w_1| < \pi$ y $a_2 < |w_2| < \pi$.
- Se dirá que una transformada de Fourier $F_A(w_1, w_2)$ es pasa altas si $|F_A(w_1, w_2)| \sim 0$ cuando $0 < |w_1| < a_1$ y $0 < |w_2| < a_2$.
- Finalmente, se dirá que una transformada de Fourier $F_A(w_1, w_2)$ es pasa banda si $|F_A(w_1, w_2)| \sim 0$ cuando $0 < |w_1| < a_1,\ b_1 < |w_1| < \pi$ y $0 < |w_2| < a_2,\ b_2 < |w_2| < \pi$.

Ejemplo - Pasa Bajas



Muestreo y Aliasing

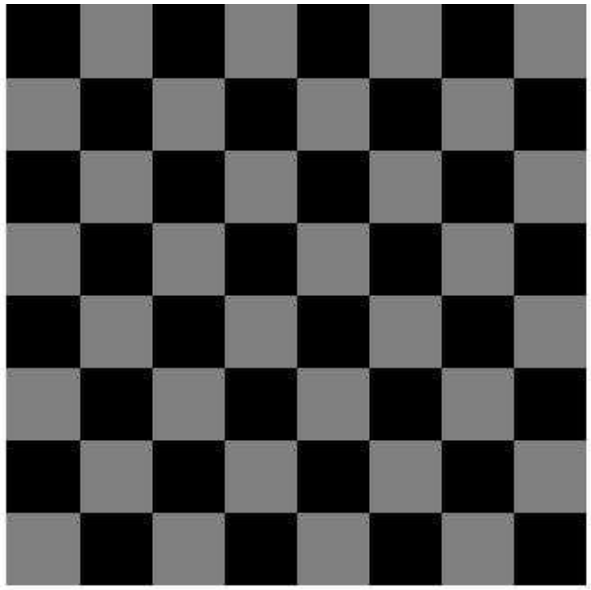
- Dada una secuencia 2-D $\mathbf{A}$ quisieramos obtener una secuencia $\mathbf{C}$ al sub-muestrear $\mathbf{A}$:

$$C(m,n) = A(S_1 m, S_2 n) \tag{83}$$

  en donde $S_1, S_2 > 0$ son enteros.

- Nos gustaría que $\mathbf{C}$ fuera muy parecida a $\mathbf{A}$.
- Por ejemplo, dada una imagen de 512×512 quisieramos obtener una imagen de 256×256 seleccionando un pixel si y otro no de la imagen original.
- Las cosas pueden salir muy mal en el muestreo con efectos inesperados.

Ejemplo



Original image (8x8)

Two possible 4x4 sub-sampled images

Transformada de Fourier de una Secuencia Muestreada

$$B(m,n) = A(m,n)comb(m,n)$$
$$C(m,n) = B(S_1 m, S_2 n)$$

- Primero se obtiene la transformada de Fourier de $\mathbf{B}$ utilizando la propiedad de multiplicación:

$$F_B(w_1,w_2) = \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_A(w_1',w_2')F_{comb}(w_1-w_1',w_2-w_2')dw_1'dw_2'$$
$$= \frac{1}{4\pi^2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_A(w_1',w_2')[\frac{4\pi^2}{S_1S_2}\sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty}\delta(w_1-w_1'-\frac{k2\pi}{S_1},w_2-w_2'-\frac{l2\pi}{S_2})]dw_1'dw_2'$$
$$= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty}\frac{1}{S_1S_2}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi} F_A(w_1',w_2')\delta(w_1-w_1'-\frac{k2\pi}{S_1},w_2-w_2'-\frac{l2\pi}{S_2})dw_1'dw_2'$$

Para $w_1, w_2 \in [-\pi,\pi)$, sea $K(w_1) = \{k | w_1 - \frac{k2\pi}{S_1} \in [-\pi,\pi)\}$ y $L(w_2) = \{l | w_2 - \frac{l2\pi}{S_2} \in [-\pi,\pi)\}$. Entonces:

$$F_B(w_1,w_2) = \frac{1}{S_1S_2}\sum_{k\in K(w_1)}\sum_{l\in L(w_2)} F_A(w_1-\frac{k2\pi}{S_1},w_2-\frac{l2\pi}{S_2}) \quad (84)$$

Ejemplo

Suponga que $S_1 = S_2 = 2$, i.e., estamos sub-muestreando por 2. Entonces para $w_1, w_2 \in (-\pi, \pi)$,

$$\begin{aligned} K(w_1) &= \{k | w_1 - k\pi \in (-\pi, \pi)\} = \{-1, 0, 1\} \\ L(w_2) &= \{l | w_2 - l\pi \in (-\pi, \pi)\} = \{-1, 0, 1\} \end{aligned}$$

y tenemos:

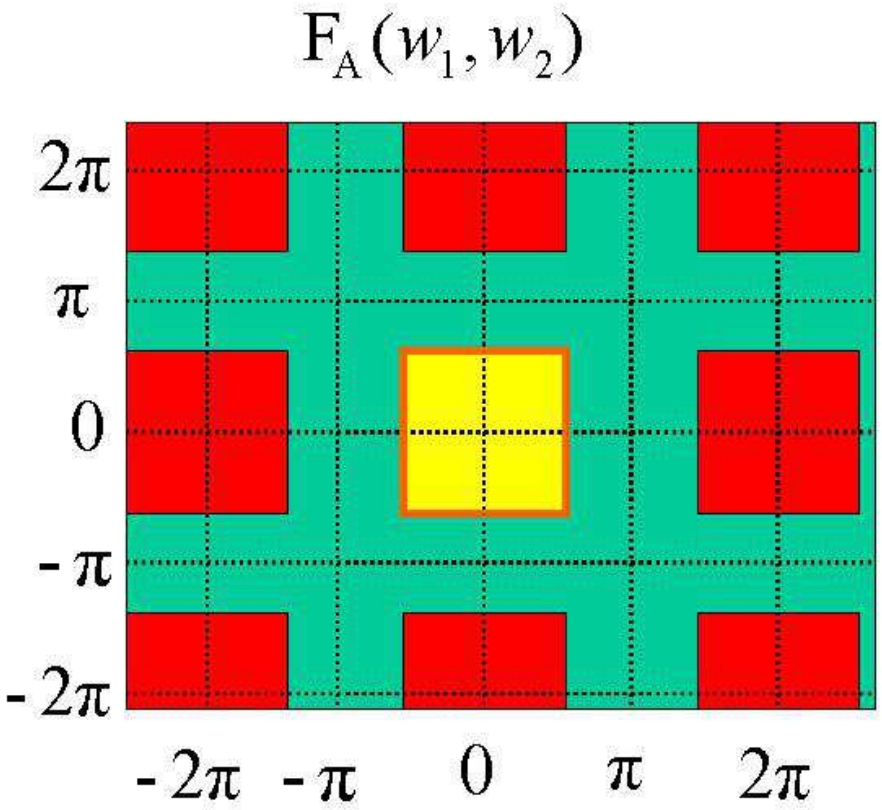
$$F_B(w_1, w_2) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} F_A(w_1 - k\pi, w_2 - l\pi) \tag{85}$$

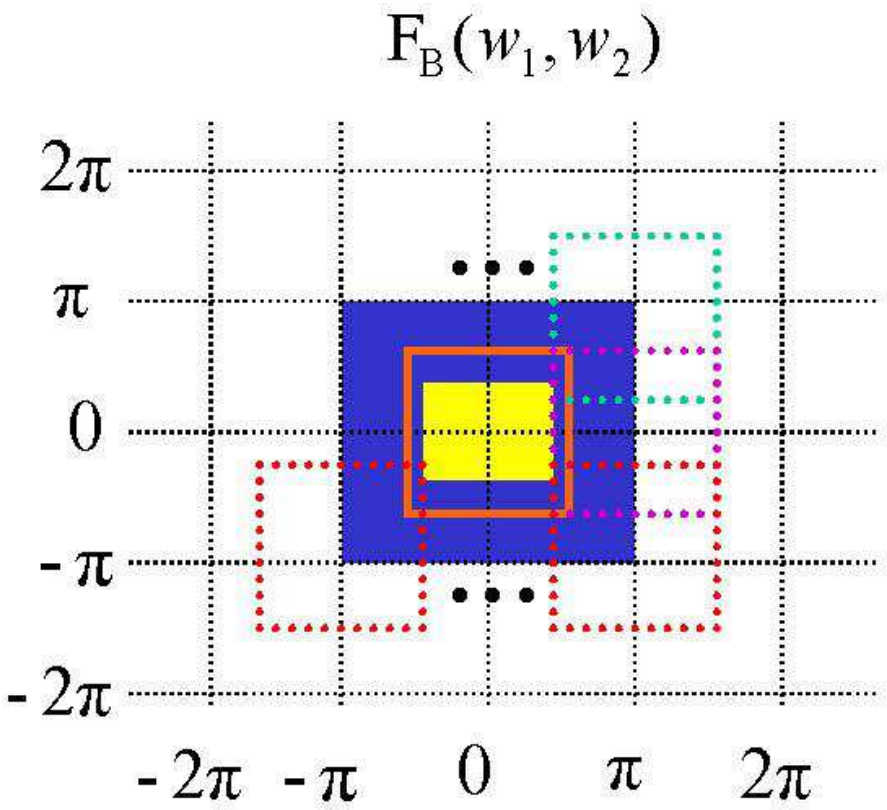
- Si durante este proceso hay traslape, i.e., digamos

$$(F_B(w_1, w_2) - F_A(w_1, w_2)) F_A(w_1, w_2) \neq 0$$

entonces diremos que hay aliasing en la operación de sub-muestreo.

Ejemplo - cont.





Aliased Frequencies

Aliased frequencies shown inside [- π, π) x [- π, π) only

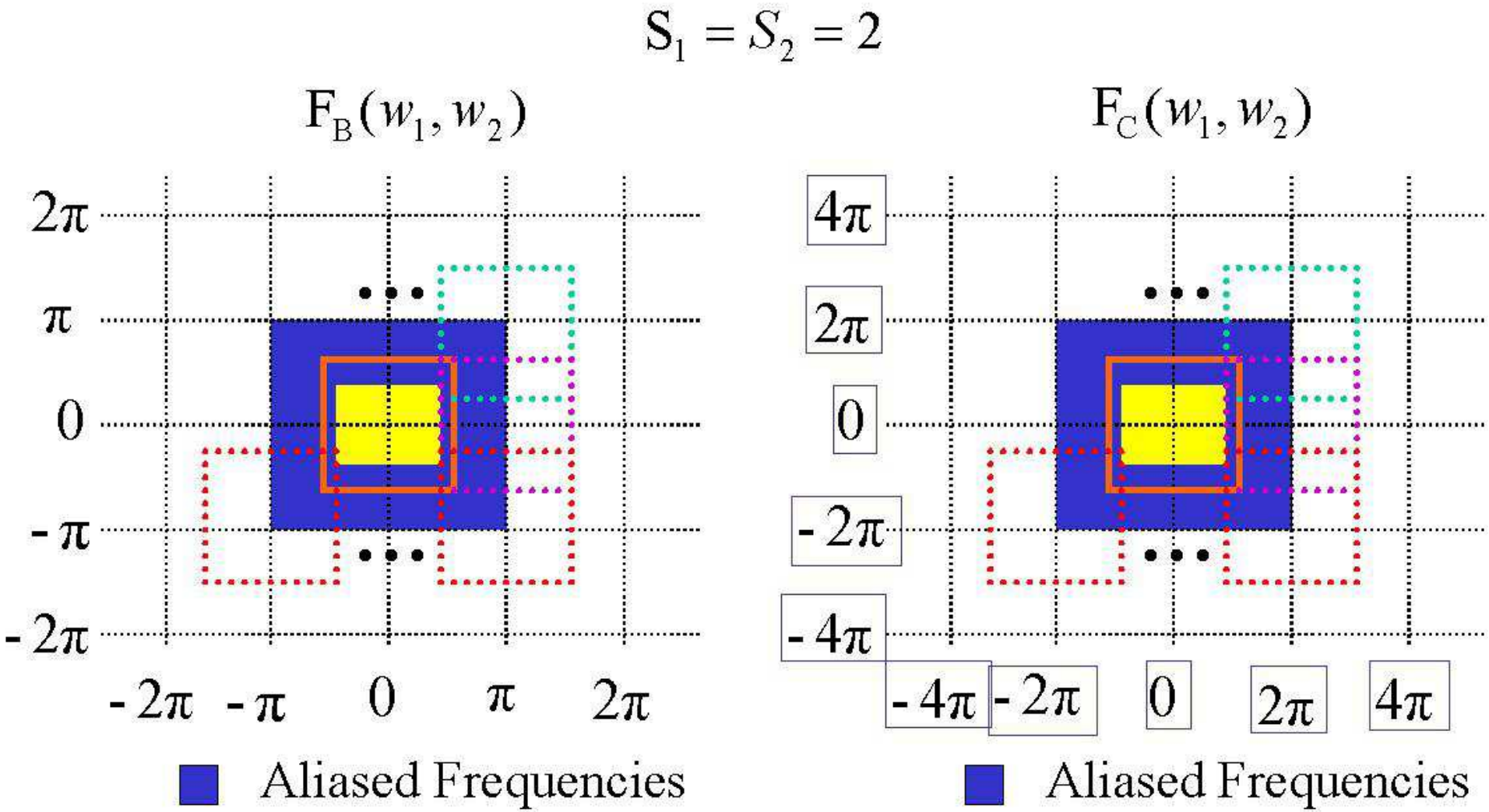
T-F de Secuencia Muestreada - cont.

Regresando a la transformada que estabamos calculando:

- Ahora podemos obtener $F_C(w_1, w_2)$

$$
\begin{aligned}
F_C(w_1, w_2) &= \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} B(S_1 m, S_2 n) e^{-j(mw_1 + nw_2)} \\
&= \sum_{m=\ldots,-S_1,0,S_1,\ldots} \sum_{n=\ldots,-S_2,0,S_2,\ldots} B(m, n) e^{-j(mw_1/S_1 + nw_2/S_2)} \\
&= F_B(w_1/S_1, w_2/S_2) \\
&= \frac{1}{S_1 S_2} \sum_{k \in K(w_1)} \sum_{l \in L(w_2)} F_A\left(\frac{w_1}{S_1} - \frac{k2\pi}{S_1}, \frac{w_2}{S_2} - \frac{l2\pi}{S_2}\right)
\end{aligned}
\tag{86}
$$

Ejemplo - cont.



Aliasing

- Esta claro que a menos de que seamos cuidadosos, la señal muestreada puede ser muy diferente de la original.
- Para que no ocurra el "aliasing" después del muestreo $F_A(w_1, w_2)$ es necesario que:

$$F_A(w_1, w_2) = 0, \ \frac{\pi}{S_1} < |w_1| < \pi, \ \frac{\pi}{S_2} < |w_2| < \pi \tag{87}$$

  de manera que no haya traslape.
- Pero ¿y si hay traslape?
  - Entonces tendremos que filtrar $\mathbf{A}$ de forma pasa-bajas para asegurar que las cosas vayan como se indico arriba.
  - Tal filtro pasa-bajas es llamado un filtro antialiasing.

Resumen

- En esta Clase aprendimos la equivalencia entre convolución y filtrado lineal.
- Repasamos las trasformadas de Fourier en dos dimensiones de secuencias en 2-D.
- Se comentaron varias propiedades de las transformadas Fourier y en particular se vio que la transformada de Fourier "convierte" la convolución a multiplicación.
- Utilizando las propiedades de la transformada de Fourier de las funciones delta de Kronecker y Dirac aprendimos acerca del muestreo y aliasing.

Tarea VI

1. Demuestre la propiedad de modulación de la transformada de Fourier.
2. Demuestre que si una secuencia bidimensional es separable entonces también lo es su transformada de Fourier, i.e., si $A(m,n) = A_1(m)A_2(n)$ entonces $F_A(w_1,w_2) = F_{A_1}(w_1)F_{A_2}(w_2)$ en donde $F_{A_1}(w_1), F_{A_2}(w_2)$ son trasformadas de Fourier *uni* dimensionales tales que $F_a(w) = \frac{1}{2\pi}\sum_{n=-\infty}^{+\infty} a(n)e^{-jwn}$.
3. Obtenga la transformada de Fourier de la secuencia 2-D $A(m,n)$ dada por:

	$n=0$	1	2
$m=0$	1	2	-1
1	2	4	-2
2	-1	-2	1

   Simplifique su respuesta tanto como pueda.
4. Calcule la transformada de Fourier de la secuencia limitada en extensión $A(m,n) = 1,\ 0 \le m < 8, 0 \le n < 8$ and $A(m,n) = 0$ otherwise.
5. Calculate the Fourier transform of the limited extent sequence $A(m,n) = (-1)^{m+n},\ 0 \le m < 8, 0 \le n < 8$ and $A(m,n) = 0$ cualquier otro.
6. Basado en los dos incisos anteriores, encuentre la transformada de Fourier de la imagen de tablero($D(m,n)$). (Tip: Suponga gris=2, negro=0). Para muestreo con $S_1 = S_2 = 2$ demuestre que hay aliasing. Calcule la transformada de Fourier de la secuencia sub-muestreada $C(m,n) = D(S_1m, S_2n)$. (Hagalo utilizando la $F_D(w_1,w_2)$. Tome la transformada inverse y verifique con sub-muestreo directo en el "dominio de la secuencia".

7. Encuentre la periodicidad de $F_B(w_1, w_2)$ y $F_C(w_1, w_2)$ como se obtuvo en las diapositivas de muestreo y aliasing. Dibuje una figura mostrando *ambos* el aliasing y periodicidad involucrados en $F_B(w_1, w_2)$, $F_C(w_1, w_2)$. Tome $F_A(w_1, w_2)$ de la secuencia original y S_1, S_2 a su gusto, siempre y cuando haya antialiasing.

Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

Definición de la Transformada de Fourier en 2-D

La Transformada de Fourier de una secuencia 2-D $\mathbf{A}$, $\mathcal{F}(\mathbf{A})$ está definida como:

$$\begin{aligned}\mathcal{F}(\mathbf{A}) &= F_A(w_1, w_2) \\ &= \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} A(m,n) e^{-j(mw_1+nw_2)} \qquad -\pi \leq w_1, w_2 < \pi \end{aligned} \tag{88}$$

La Transformada inversa de Fourier 2-D $\mathcal{F}^{-1}(\mathbf{A})$ es:

$$\begin{aligned} A(m,n) &= \mathcal{F}^{-1}(\mathbf{A}) \\ &= \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F_A(w_1, w_2) e^{+j(mw_1+nw_2)} dw_1 dw_2 \end{aligned} \tag{89}$$

$A(m,n) \overset{\mathcal{F}}{\leftrightarrow} F_A(w_1, w_2)$

- $A(m,n)$: secuencia discreta bidimensional, m, n varian sobre enteros
- $F_A(w_1, w_2)$: función periódica bidimensional en 2π, w_1, w_2 varian en un continuo.

Transformada de Fourier y Convolución

- $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$

$$
\begin{aligned}
C(m,n) &= \sum_{k=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A(k,l)B(m-k,n-l) \\
F_C(w_1,w_2) &= F_A(w_1,w_2)F_B(w_1,w_2) \qquad (90)
\end{aligned}
$$

Muestreo y Aliasing

- La secuencia muestreada:

$$C(m,n) = A(S_1 m, S_2 n) \tag{91}$$

en donde $S_1, S_2 > 0$ son enteros, tiene la transformada de Fourier:

$$F_C(w_1,w_2) = \frac{1}{S_1 S_2} \sum_{k \in K(w_1)} \sum_{l \in L(w_2)} F_A\left(\frac{w_1}{S_1} - \frac{k2\pi}{S_1}, \frac{w_2}{S_2} - \frac{l2\pi}{S_2}\right) \tag{92}$$

- Para que no ocurra aliasing después del muestreo $F_A(w_1,w_2)$ debe ser:

$$F_A(w_1,w_2) = 0, \;\; \frac{\pi}{S_1} < |w_1| < \pi, \;\; \frac{\pi}{S_2} < |w_2| < \pi \tag{93}$$

La necesidad de una Transformada de Fourier "Computable"

- La Transformada de Fourier en 2-D tiene muchas propiedades útiles para el análisis de secuencias 2-D y convolución.
- Desafortunadamente ésta Transformada de Fourier solo puede ser calculada explícitamente para algunas secuencias simples.
    - Las variables de la Transformada de Fourier w_1, w_2 varían en un continuo. Por lo tanto la transformada de Fourier $F_A(w_1, w_2)$ de una secuencia $A(m, n)$ no puede ser calculada directamente por una computadora digital.
- Ahora definiremos "otra" transformada de Fourier que sea calculable y proporcione propiedades similares.

La DFT en 2-D para Secuencias de Extensión Finita

Sea $A(m,n)$ una secuencia de extensión finita, i.e.,

$$A(m,n) \quad \begin{cases} \neq 0, & 0 \leq m \leq M_1 - 1,\ 0 \leq n \leq N_1 - 1 \\ = 0 & \text{cualquier otro} \end{cases}$$

La Transformada Discreta de Fourier (DFT) 2-D de $[M_1, N_1]$ puntos de $\mathbf{A}$ está definida como:

$$DF_A(k,l) = \sum_{m=0}^{M_1-1} \sum_{n=0}^{N_1-1} A(m,n) e^{-j(\frac{2\pi k}{M_1}m + \frac{2\pi l}{N_1}n)}, \quad k = 0, \ldots, M_1 - 1,\ l = 0, \ldots, N_1 - 1 \quad (94)$$

$A(m,n)$ puede "recuperarse" de $DF_A(k,l)$ vía:

$$A(m,n) = \frac{1}{M_1 N_1} \sum_{k=0}^{M_1-1} \sum_{l=0}^{N_1-1} DF_A(k,l) e^{j(\frac{2\pi m}{M_1}k + \frac{2\pi n}{N_1}l)} \quad (95)$$

La DFT 2-D y la FT 2-D

Recordando que $A(m,n)$ es una secuencia de extensión finita vamos a comparar las definiciones de las dos transformadas de Fourier:

$$DF_A(k,l) = \sum_{m=0}^{M_1-1}\sum_{n=0}^{N_1-1} A(m,n)e^{-j(\frac{2\pi k}{M_1}m+\frac{2\pi l}{N_1}n)}, \quad k=0,\ldots,M_1-1,\ l=0,\ldots,N_1-1$$

$$F_A(w_1,w_2) = \sum_{m=0}^{M_1}\sum_{n=0}^{N_1} A(m,n)e^{-j(mw_1+nw_2)} \qquad -\pi \le w_1, w_2 < \pi$$

- La DFT 2-D es una versión muestreada de $F_A(w_1,w_2)$, i.e.,

$$DF_A(k,l) = F_A\left(\frac{2\pi k}{M_1}, \frac{2\pi l}{N_1}\right) \tag{96}$$

Notando que $F_A(w_1,w_2)$ es periódica con 2π: $k=0 \rightarrow w_1 = 0$

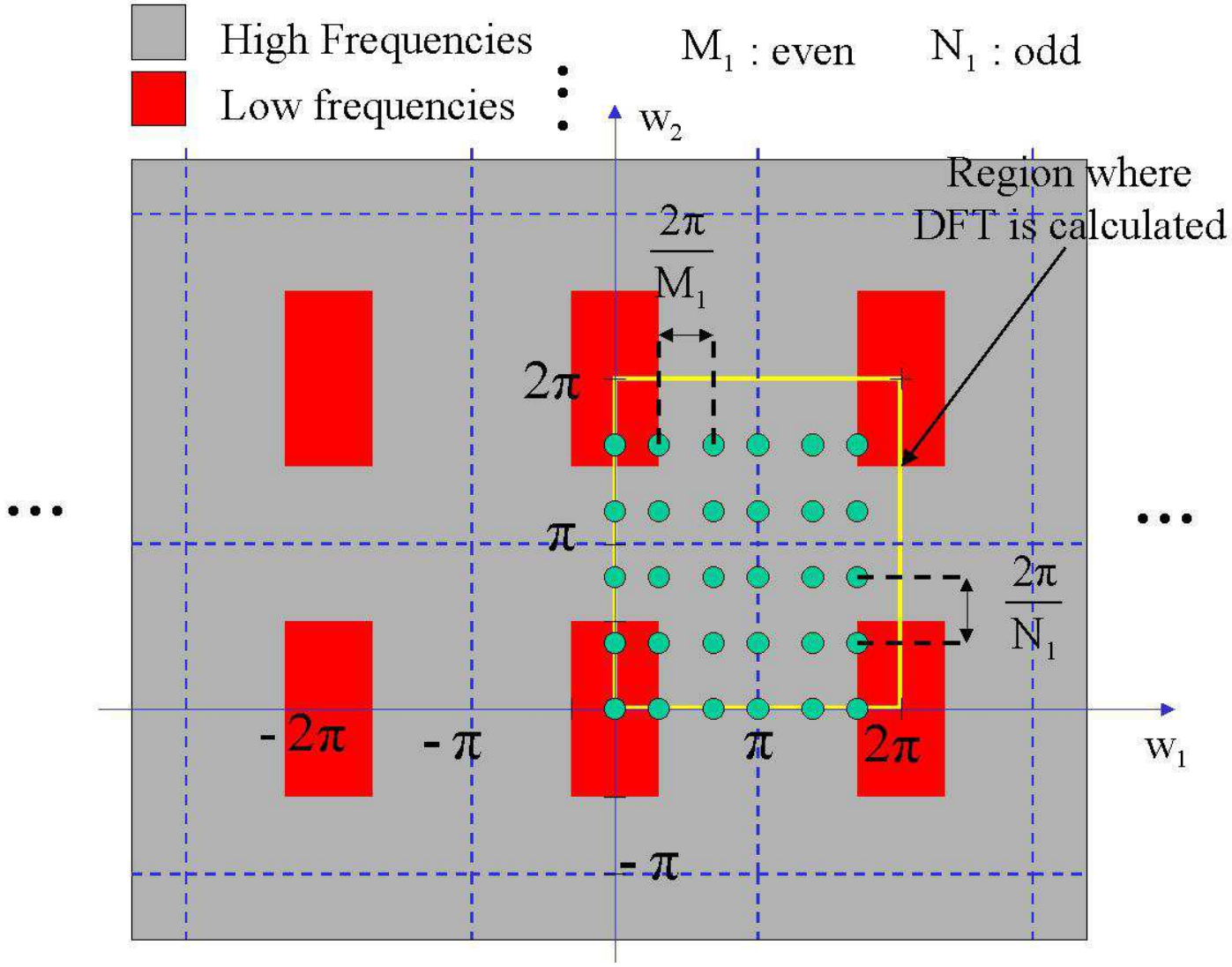
$$k = M_1 - 1 \ \rightarrow\ w_1 = \frac{2\pi(M_1-1)}{M_1} = 2\pi - \frac{2\pi}{M_1} = -\frac{2\pi}{M_1}$$

$$k = M_1/2 \ \rightarrow\ w_1 = \pi \quad \text{si } M_1 \text{ par}$$

$$k = (M_1-1)/2 \ \rightarrow\ w_1 = \pi - \frac{\pi}{M_1} \quad \text{si } M_1 \text{ impar}$$

y de forma similar para l y w_2.

Ejemplo


High Frequencies
Low frequencies
M1 : even
N1 : odd
w2
Region where
DFT is calculated
2π / M1
2π
π
2π / N1
- 2π
- π
π
2π
- π
w1

La DFT 2-D y la FT 2-D cont.

Demos ahora un vistazo a las transformadas inversas:

$$A(m,n) = \frac{1}{M_1 N_1} \sum_{k=0}^{M_1-1} \sum_{l=0}^{N_1-1} DF_A(k,l) e^{j(\frac{2\pi m}{M_1}k + \frac{2\pi n}{N_1}l)}$$

$$A(m,n) = \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} F_A(w_1,w_2) e^{+j(mw_1+nw_2)} dw_1 dw_2$$

Note que la FT y su inversa están definidas para *cualquier* secuencia mientras que la DFT esta definida *solo* para secuencias de extensión finita.

Usando la DFT inversa y notando que $e^{j\frac{2\pi M_1}{M_1}} = 1$ podemos ver que:

$$\begin{aligned} A(m,n) &= \frac{1}{M_1 N_1} \sum_{k=0}^{M_1-1} \sum_{l=0}^{N_1-1} DF_A(k,l) e^{j(\frac{2\pi m}{M_1}k + \frac{2\pi n}{N_1}l)} \times e^{j\frac{2\pi M_1}{M_1}} e^{j\frac{2\pi N_1}{N_1}} \\ &= \frac{1}{M_1 N_1} \sum_{k=0}^{M_1-1} \sum_{l=0}^{N_1-1} DF_A(k,l) e^{j(\frac{2\pi [m+M_1]}{M_1}k + \frac{2\pi [n+N_1]}{N_1}l)} \\ &= A(m+M_1, n+N_1) \ ! \end{aligned} \tag{97}$$

i.e., si "olvidamos" que $\mathbf{A}$ tiene extensión finita, entonces la DFT inversa reconstruirá una secuencia *periódica* llamada la extensión periódica de $\mathbf{A}$.

La DFT 2-D y Extensiones Periódicas

- La propiedad de extensión periódica normalmente no es un problema ya que siempre podemos obtener una DFT inversa para $0 \leq m < M_1,\ 0 \leq n < N_1$.
- Considere sin embargo la DFT y su inversa para $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$.
  - Ya sabemos que si $\mathbf{A}$ y $\mathbf{B}$ son de extensión finita ($\mathbf{A}$ $(M_1 \times N_1)$, $\mathbf{B}$ $(M_2 \times N_2)$) entonces $\mathbf{C}$ es de extensión finita $(M_1 + M_2 - 1 \times N_1 + N_2 - 1)$.
  - $DF_C(k,l)$ debe por lo tanto ser calculada vía una DFT de $[M_1 + M_2 - 1, N_1 + N_2 - 1]$ puntos.

La DFT 2-D y la Convolución

Sean $\mathbf{A}$ y $\mathbf{B}$ secuencias de extensión finita ($\mathbf{A}$ $(M_1 \times N_1)$, $\mathbf{B}$ $(M_2 \times N_2)$).

- La DFT de $[M_1 + M_2 - 1, N_1 + N_2 - 1]$ puntos de $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$ esta dada por:

$$DF_C(k,l) = DF_A(k,l)_{[M_1+M_2-1,N_1+N_2-1]} \times DF_B(k,l)_{[M_1+M_2-1,N_1+N_2-1]} \quad (98)$$

- (Si $\mathbf{A}$ y $\mathbf{B}$ son matrices, la DFT de $[M_1 + M_2 - 1, N_1 + N_2 - 1]$ puntos de $\mathbf{A}$ puede ser calculada al extender o "rellenar" $\mathbf{A}$ con ceros y de forma similar para $\mathbf{B}$).
- Se suprimirá la notación $DF_X(k,l)_{[M_1+M_2-1,N_1+N_2-1]}$ en el entendimiento de que las diferentes DFTs serán calculadas con los puntos requeridos.
- Artifacts ("defectos") causados por *no* calcular la DFT con los puntos requeridos serán debidos al aliasing ("falseo") temporal.

Calculando DFT 2-D y Convoluciones

- La DFT puede ser calculada por un algoritmo rápido conocido como la FFT (Fast Fourier Transform). En Matlab:

$$>> DFA = \text{fft2}(A, M1, N1);$$

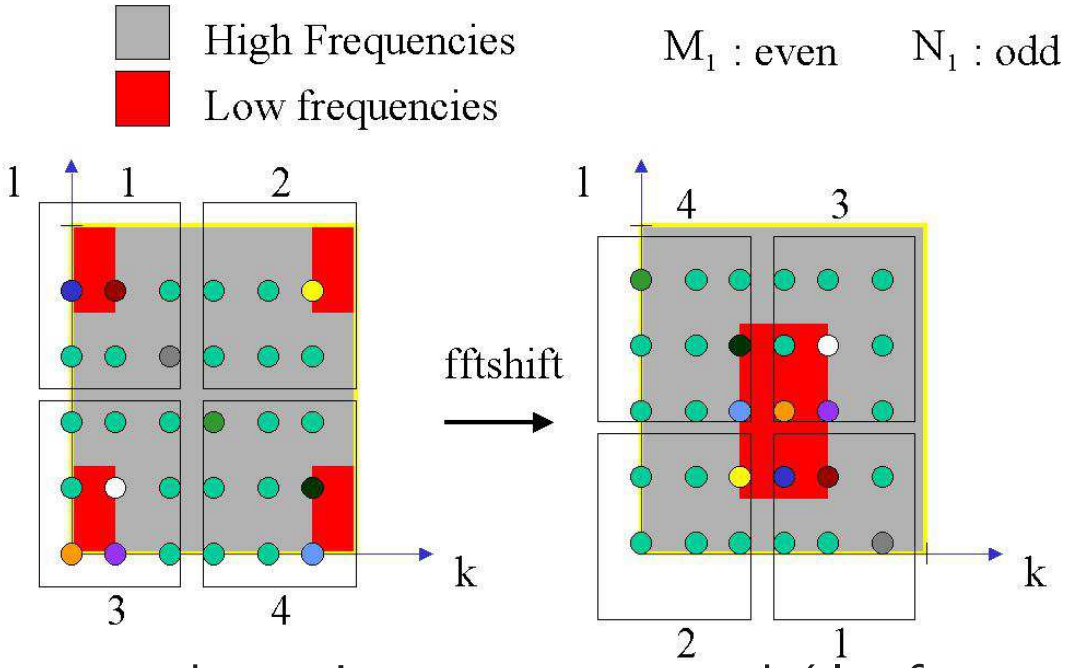
en donde $M1, N1$ denotan el punto al cual se calcula la DFT.

- La convolución $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$ de las secuencias de extensión finita $\mathbf{A}$ $(M_1 \times N_1)$ y $\mathbf{B}$ $(M_2 \times N_2)$ puede ser calculada *usando* el algoritmo fft vía:

$$\begin{aligned}
>> \quad & DFA = \text{fft2}(A, M1 + M2 - 1, N1 + N2 - 1); \\
>> \quad & DFB = \text{fft2}(B, M1 + M2 - 1, N1 + N2 - 1); \\
>> \quad & DFC = DFA. * DFB; \\
>> \quad & C = \text{ifft2}(DFC, M1 + M2 - 1, N1 + N2 - 1);
\end{aligned}$$

en donde ifft2 denota el algoritmo de la fft inversa y $M1 = M_1,\ N1 = N_1,$ etc.

DFT y fftshift



- La DFT $[M_1, N_1]$ de una imagen $M_1 \times N_1$ tendrá las frecuencias bajas alrededor de $k = 0, k = N_1 - 1$ y $l = 0, l = M_1 - 1$ (ver también el dibujo previo).
- Cuando graficamos la DFT de una imagen como imágenes es conveniente tener las frecuencias bajas en el centro de la gráfica.
- Este corrimiento por conveniencia de visualización puede hacerse vía el mando `fftshift` en Matlab.

DFT y fftshift cont.

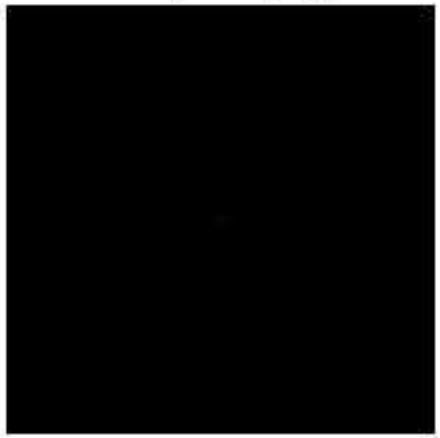
$$
\begin{aligned}
&>> \quad DFA = \text{fft2}(A, M1, N1);\\
&>> \quad DFA2 = \text{fftshift}(DFA);\\
&>> \quad \text{image}(\text{mynormalize}(\text{abs}(DFA2)));
\end{aligned}
$$

- `fftshift` en Matlab centrará las bajas frecuencias por conveniencia de visualización.
- Note que $DFA2 \neq DFA$ y entonces ifft2$(DFA2, M1, N1) \neq$ ifft2$(DFA, M1, N1)$.
- DFA puede ser obtenida de $DFA2$ haciendo *otra* `fftshift`, i.e., $DFA =$ fftshift$(DFA2)$.

Ejemplo

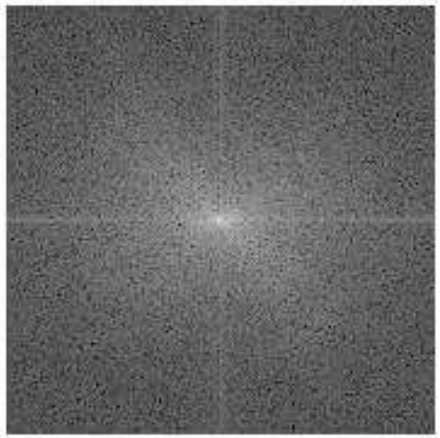
abs(F) (normalized, F=fft2(lenna))

abs(fftshift(F))



$\log_{10}$(abs(F)+1)

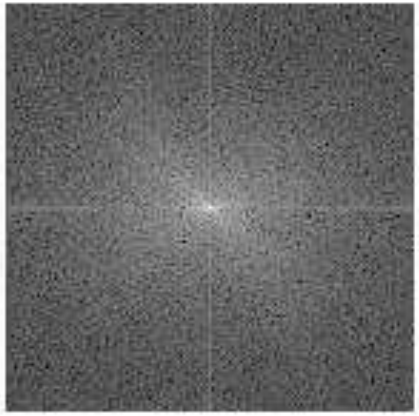
$\log_{10}$(abs(fftshift(F))+1)



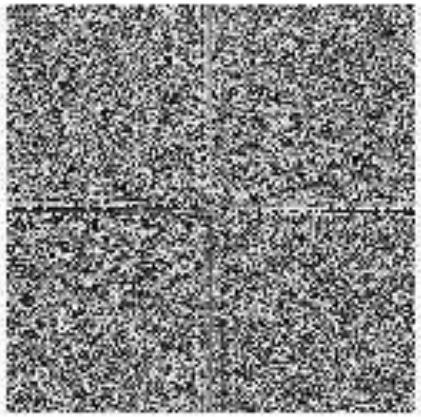
DFT de Imágenes Naturales

A (Lenna)

log10(abs(fft2(A))+1)



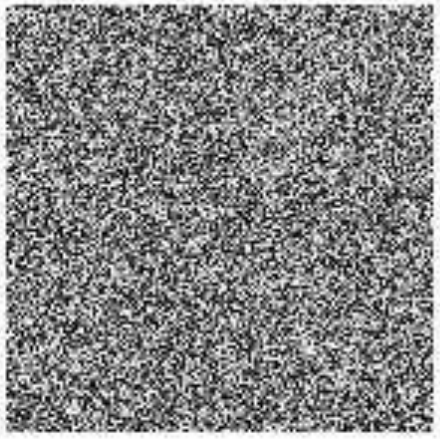
angle(fft2(A)) (normalized)



B (B=randn(512))

log10(abs(fft2(B))+1)

angle(fft2(B)) (normalized)



Importancia de las Bajas Frecuencias

La importancia de los coeficientes de bajas frecuencias de DFTs de imagen se puede demostrar como sigue:

- Sea $\mathbf{A}$ $M_1 \times N_1$. y que

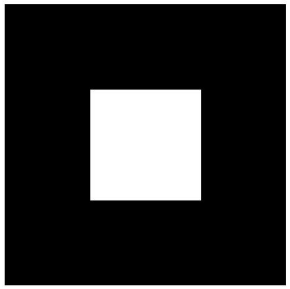
$$\mathcal{R}_1 = \{i | (0 \leq i \leq W_1) \text{ o } (M_1 - W_1 \leq i \leq M_1 - 1)\}$$
$$\mathcal{R}_2 = \{i | (0 \leq i \leq W_2) \text{ o } (N_1 - W_2 \leq i \leq N_1 - 1)\}.$$

  Defina una ventana $(2W_1 + 1) \times (2W_2 + 1)$ "alrededor" de las bajas frecuencias

$$w(k,l) = \begin{cases} 1 & k \in \mathcal{R}_1 \text{ y } l \in \mathcal{R}_2 \\ 0 & \text{cualquierotro} \end{cases} \tag{99}$$

- Considere $DF_C(k,l) = DF_A(k,l)w(k,l)$. Esto "mantiene" los $(2W_1 + 1) \times (2W_2 + 1)$ coeficientes DFT y ceros en el resto.
- Estamos interesados en el error cuadrático medio dado por: $1/(M_1 N_1) \sum_{m=0}^{M_1-1} \sum_{n=0}^{N_1-1} (A(m,n) - C(m,n))^2$.
- Note que como W_1, W_2 aumentan estamos agregando más y más coeficientes altos al conjunto de coeficientes que tenemos.

Ventana de Baja Frecuencia



w for $W_1 = W_2 = 100$ (normalizada y fftshifted)

La ventana puede ser implementada en Matlab vía

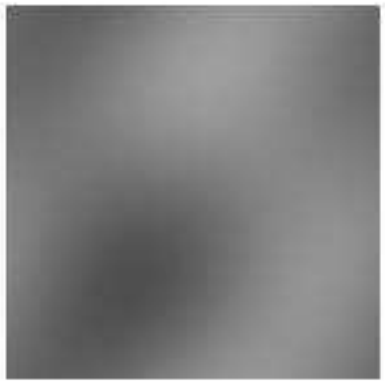
```
>> r1 = zeros(M1, 1);
>> r1(1 : W1 + 1, M1 − W1 + 1 : M1) = 1;
>> r2 = zeros(N2, 1);
>> r2(1 : W2 + 1, N1 − W2 + 1 : N1) = 1;
>> w = r1 ∗ r2′;
```

Dada $DF_C(k,l) = DF_A(k,l)w(k,l)$, Matlab puede introducir pequeños errores numéricos en la transformada inversa originando imágenes con valores complejos. $C(m,n)$ se reconstruye mejor por medio de:

```
>> C = real(ifft2(w. ∗ DFA));
```

Ejemplo

$W_1=W_2=1$



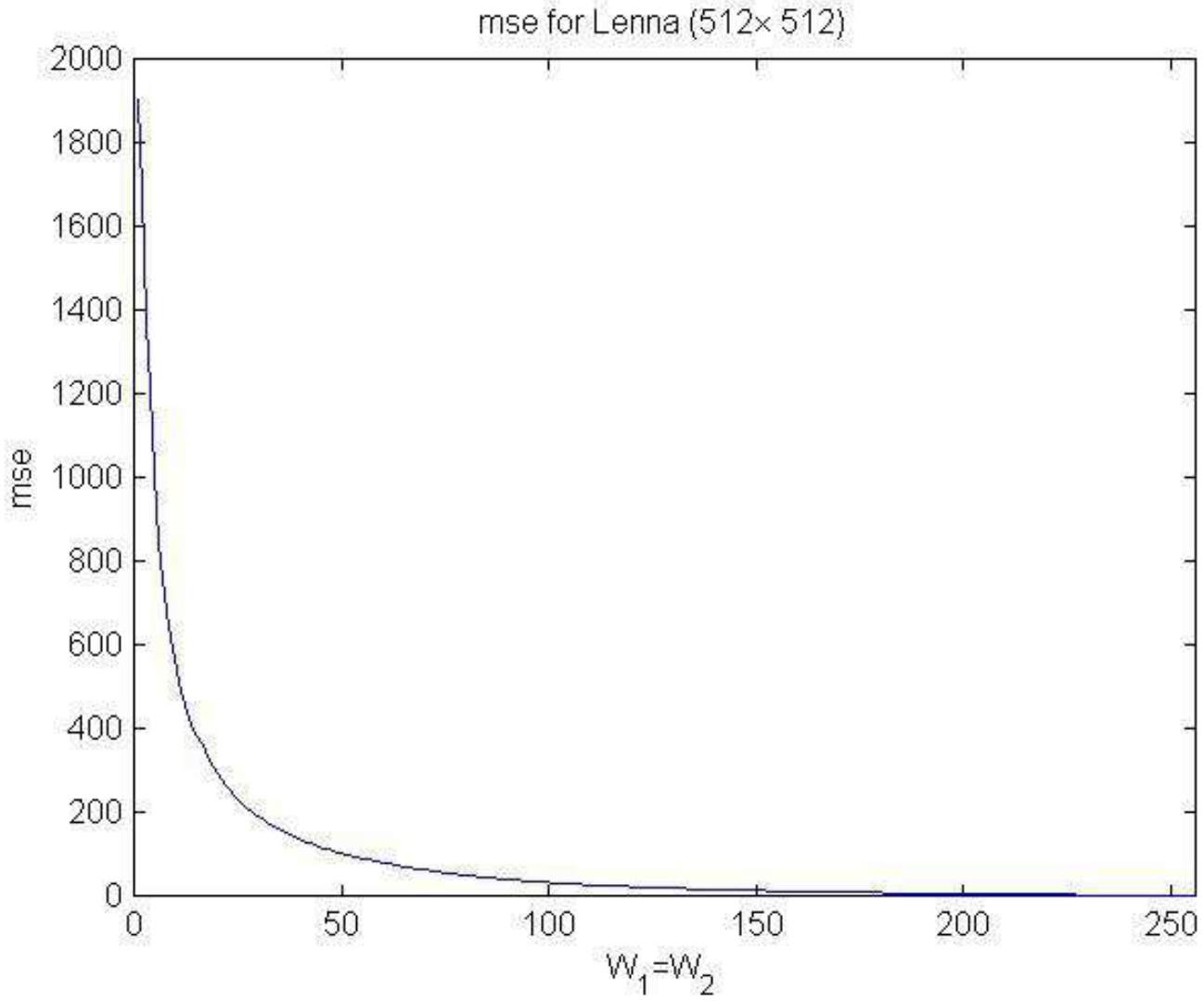
$W_1=W_2=10$

$W_1=W_2=20$

$W_1=W_2=30$

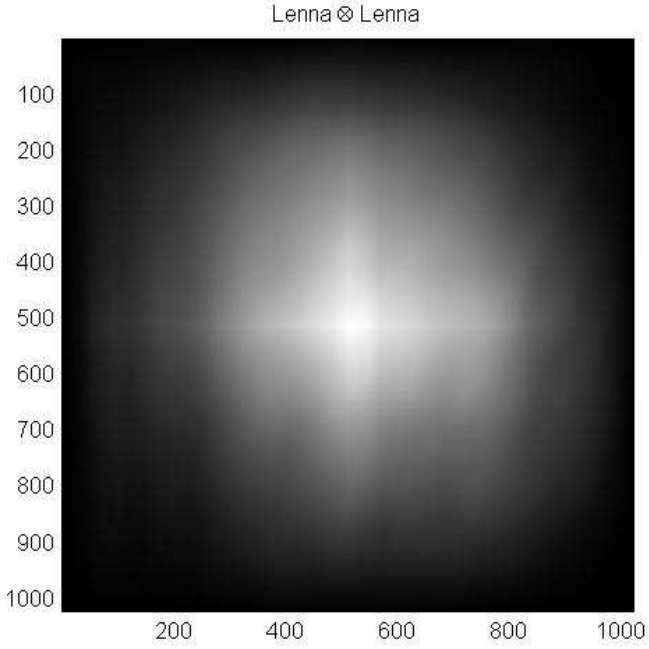
$W_1=W_2=40$

Ejemplo


mse for Lenna (512× 512)
mse
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
50
100
150
200
250
W1=W2

Convolución por DFTs

- Puesto que la DFT puede ser calculada con un algoritmo rápido esto puede ser benéfico para realizar la convolución de dos secuencias $\mathbf{A}$ $(M_1 \times N_1)$ y $\mathbf{B}$ $(M_2 \times N_2)$ vía la DFT de $[M_1 + M_2 + 1, N_1 + N_2 + 1]$ puntos.
- Sin embargo, las mejoras en velocidad solo son posibles si ambas secuencias tienen dimensiones grandes. En los otros casos las convoluciones se implementan mejor vía la convolución de suma.



Resumen

- En esta Clase aprendimos la DFT 2-D de secuencias bidimensionales de extensión finita.
- Aprendimos a calcular convoluciones usando DFTs.
- Aprendimos acerca de las propiedades básicas de la DFT de imágenes naturales.

Tarea VII

1. Calcular la DFT de su imagen. Mostrar la magnitud y fase antes y después de usar la `fftshift`. Use la función de punto `log10` en las gráficas de magnitud y normalice si es necesario.
2. Submuestree su imagen por 2 en cada dirección y calcule la DFT del resultado en dos formas:

   a) Puesto que la imagen submuestreada tiene la mitad de las dimensiones de la original, calcule la DFT hasta el *punto* de las dimensiones reducidas.

   b) Calcule la DFT hasta el punto de las dimensiones originales.

   Muestre las gráficas de magnitud y fase para ambos. Compare los resultados con el primer inciso. ¿Puede explicar las diferencias?
3. Calcule la convolución de su imagen con ella misma usando DFT.
4. Haga el procesamiento como el mostrado en las Páginas 19-20. Muestre las imágenes resultantes y la gráfica `mse`.

Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

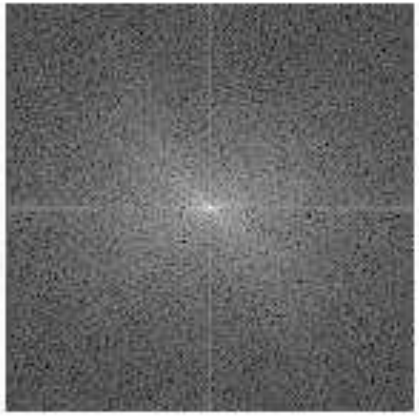
DFT 2-D y Convolución

- la DFT puede ser calculada con un algoritmo rápido y esto en ocasiones es benéfico para realizar la convolución de dos secuencias $\mathbf{A}$ $(M_1 \times N_1)$ y $\mathbf{B}$ $(M_2 \times N_2)$ vía la DTF de $[M_1 + M_2 + 1, N_1 + N_2 + 1]$ puntos.
- Las mejoras de velocidad solo son posibles si $ambas$ secuencias tienen dimensiones grandes. En los otros casos las convoluciones se implementan mejor vía la convolución de suma.

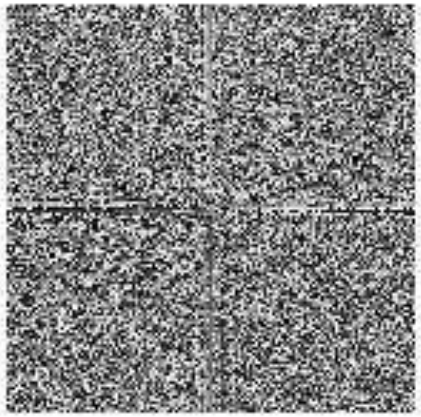
DFT de Imágenes Naturales

A (Lenna)

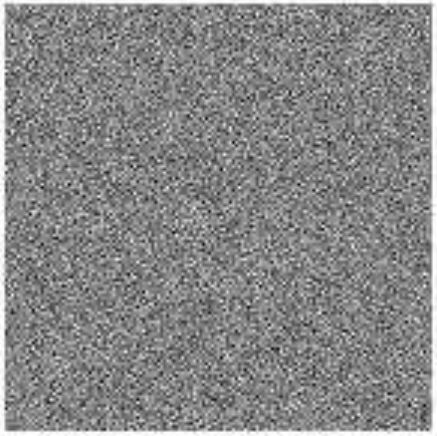
log10(abs(fft2(A))+1)



angle(fft2(A)) (normalized)

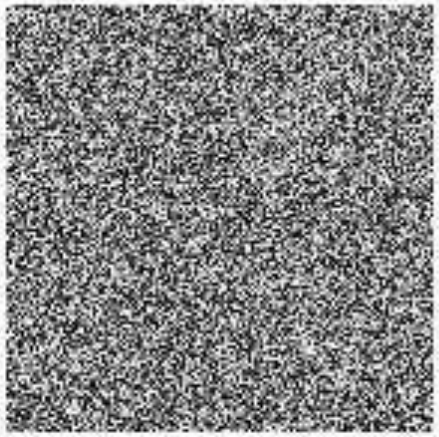


B (B=randn(512))



log10(abs(fft2(B))+1)

angle(fft2(B)) (normalized)

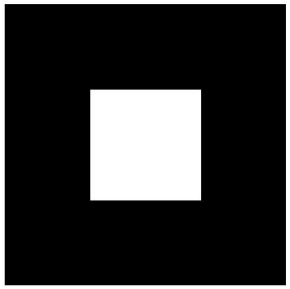


Filtrado Pasa-Bajas 2-D de Imágenes

Estamos interesados en dos maneras de implementar el filtrado pasa-bajas de imágenes:

- Definiendo "ventanas" en el dominio de DFT, seleccionando coeficientes de baja frecuencia de la DFT de imágenes y obteniendo la transformada inversa.
- Definiendo filtros pasa-bajas en el dominio espacial y obteniendo imágenes filtradas por convolución de suma.

Filtrado Pasa-Bajas por ventanas de DFT



w para $W_1 = W_2 = 100$ (normalizada y fftshifted)

Filtrado Pasa-Bajas en el Dominio Espacial

Las operaciones de filtrado pasa-bajas en el dominio espacial pueden pensarse como operaciones locales de promediación. Sea

$$L(m,n) = \begin{cases} \frac{1}{(2W+1)^2} & -W \leq m,n \leq W \\ 0 & \text{cualquier otro} \end{cases}$$

Considere $\mathbf{C} = \mathbf{L} \otimes \mathbf{A}$ en donde $\mathbf{A}$ es una imagen.

$$\begin{aligned} C(m,n) &= \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(m-k,n-l)L(k,l) \\ &= \frac{1}{(2W+1)^2} \sum_{k=-W}^{W} \sum_{l=-W}^{W} A(m-k,n-l) \end{aligned} \tag{100}$$

Esta es una promediación local si W es mucho menor que las dimensiones de $\mathbf{A}$. Para $W = 1$ La ecuación 100 se vuelve:

$$\begin{aligned} C(m,n) &= \frac{1}{9}(A(m-1,n-1) + A(m-1,n) + A(m-1,n+1) \\ &+ A(m,n-1) + A(m,n) + A(m,n+1) \\ &+ A(m+1,n-1) + A(m+1,n) + A(m+1,n+1)) \end{aligned}$$

Note que $\mathbf{L}$ se vuelve más y más pasa-bajas a medida que se incrementa W.

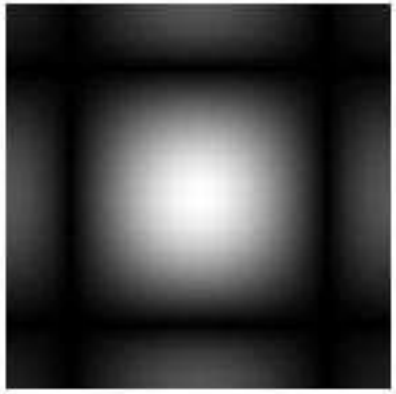
Ejemplo

C=A⊗ L (W=1)

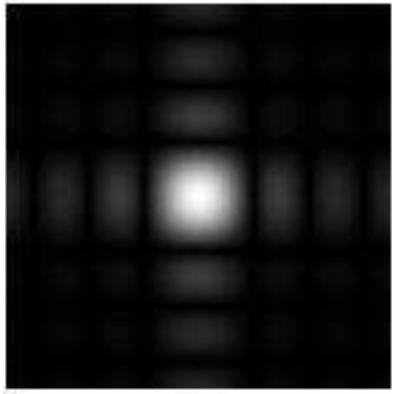
C=A⊗ L (W=3)

C=A⊗ L (W=8)

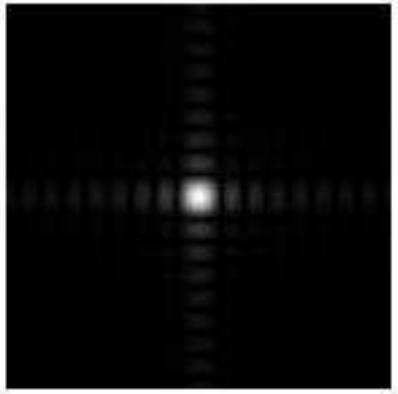
$|DF_L(k,l)|$ (normalized)



$|DF_L(k,l)|$ (normalized)



$|DF_L(k,l)|$ (normalized)



Filtrado Pasa-Bajas en el Dominio Espacial

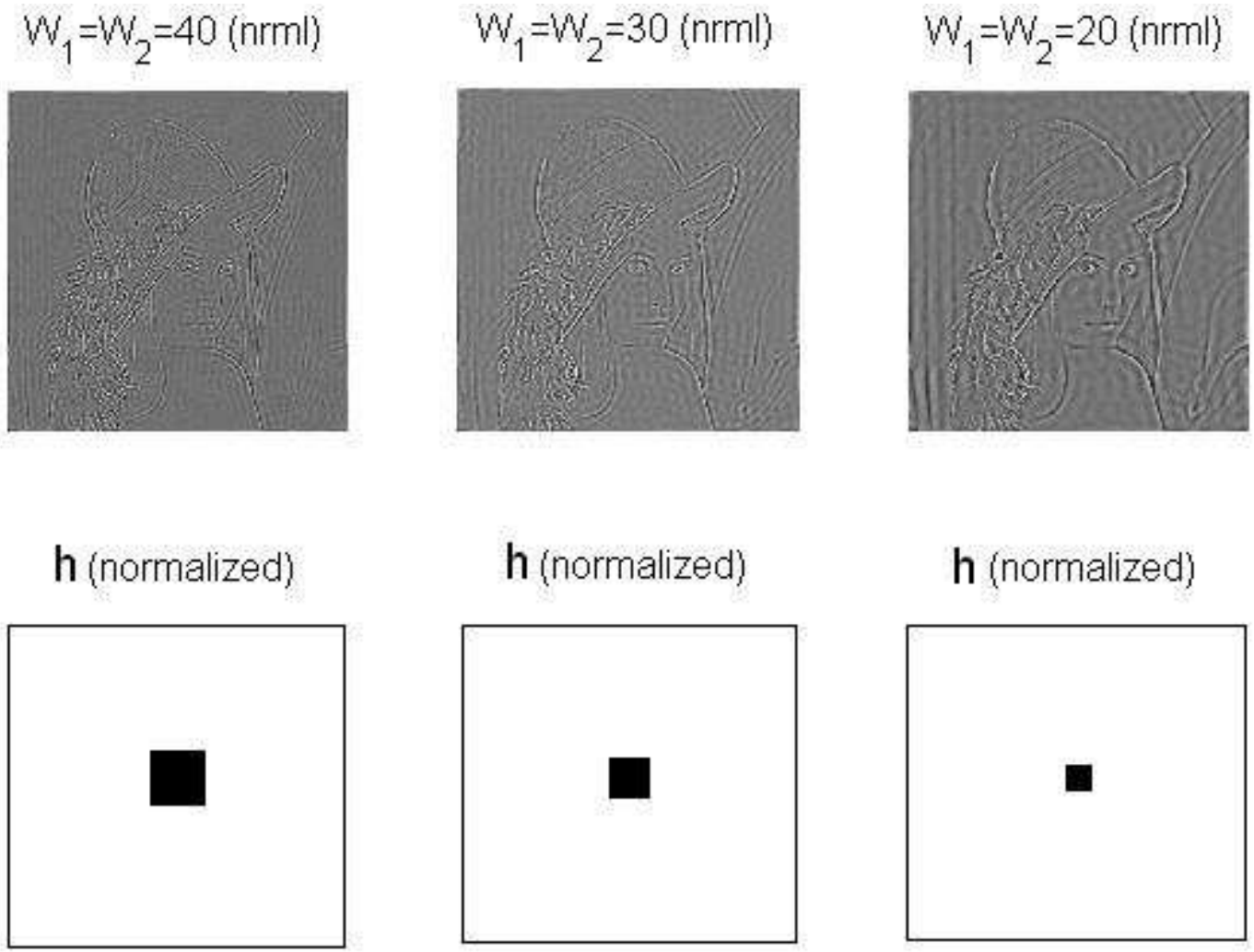
- Dado el tamaño $2W+1$ del filtro $\mathbf{L}$, podemos diseñar filtros pasa-bajas que implementen formas de promediación más complicadas utilizando conceptos de procesamiento de señales y procesamiento estadístico de señales.
- En esta Clase, nos concentraremos principalmente en filtros simples y no entraremos en los detalles de las técnicas de diseño de filtros.

Filtrado Pasa-Altas 2-D de Imágenes

El filtrado pasa-altas de una imagen puede pensarse como la imagen original menos la imagen filtrada pasa-bajas.

- Filtrado Pasa-Altas por ventanas de la DFT:
  - Si $w(k,l)$ $(W_1 \times W_2)$ es una ventana pasa-bajas de la DFT, una ventana pasa-altas $h(k,l)$ simplemente se define por $h(k,l) = 1 - w(k,l)$.
- Filtrado Pasa-Altas en el dominio espacial:
  - Si $\mathbf{L}$ es un filtro pasa-bajas de tamaño W, un filtro pasa-altas $\mathbf{H}$ simplemente se define vía $H(m,n) = \delta(m,n) - L(m,n)$.

Filtrado Pasa-Altas por ventanas de DFT



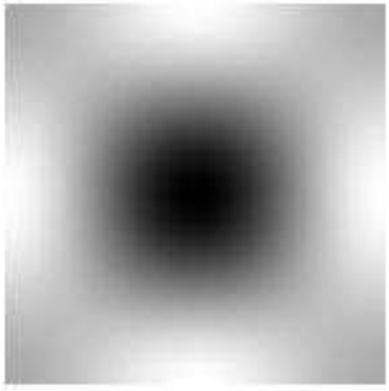
Filtrado Pasa-Altas Espacial

C=A⊗ H (W=1) (nrml)

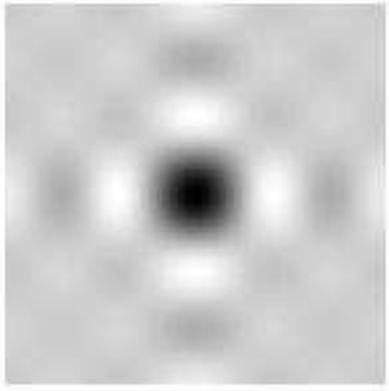
C=A⊗ H (W=3) (nrml)

C=A⊗ H (W=8) (nrml)

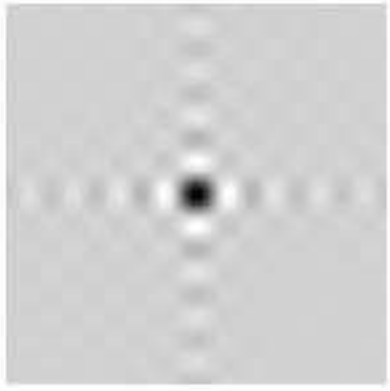
$|DF_H(k,l)|$ (normalized)



$|DF_H(k,l)|$ (normalized)



$|DF_H(k,l)|$ (normalized)

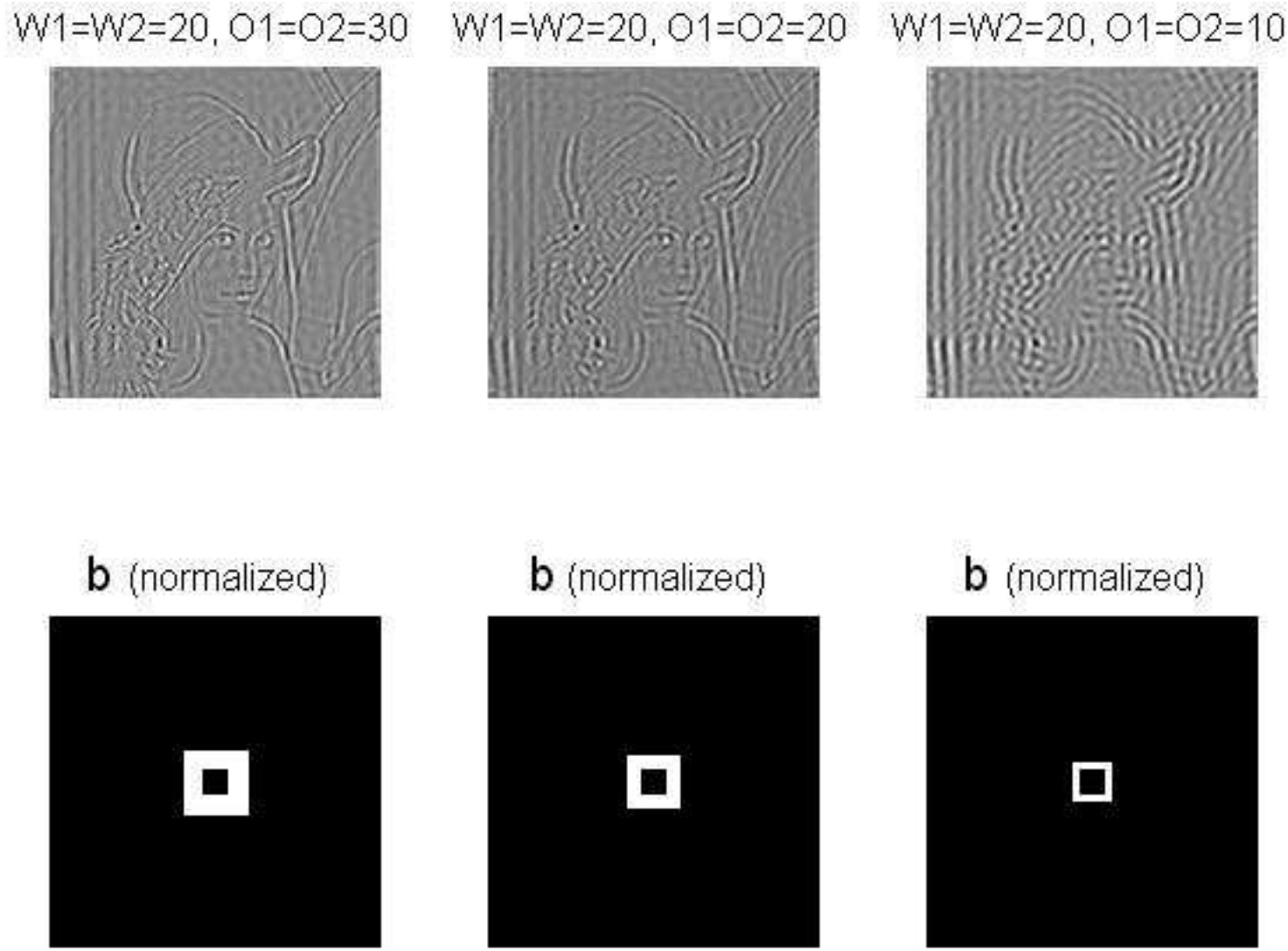


Filtrado Pasa-Banda 2-D de Imágenes

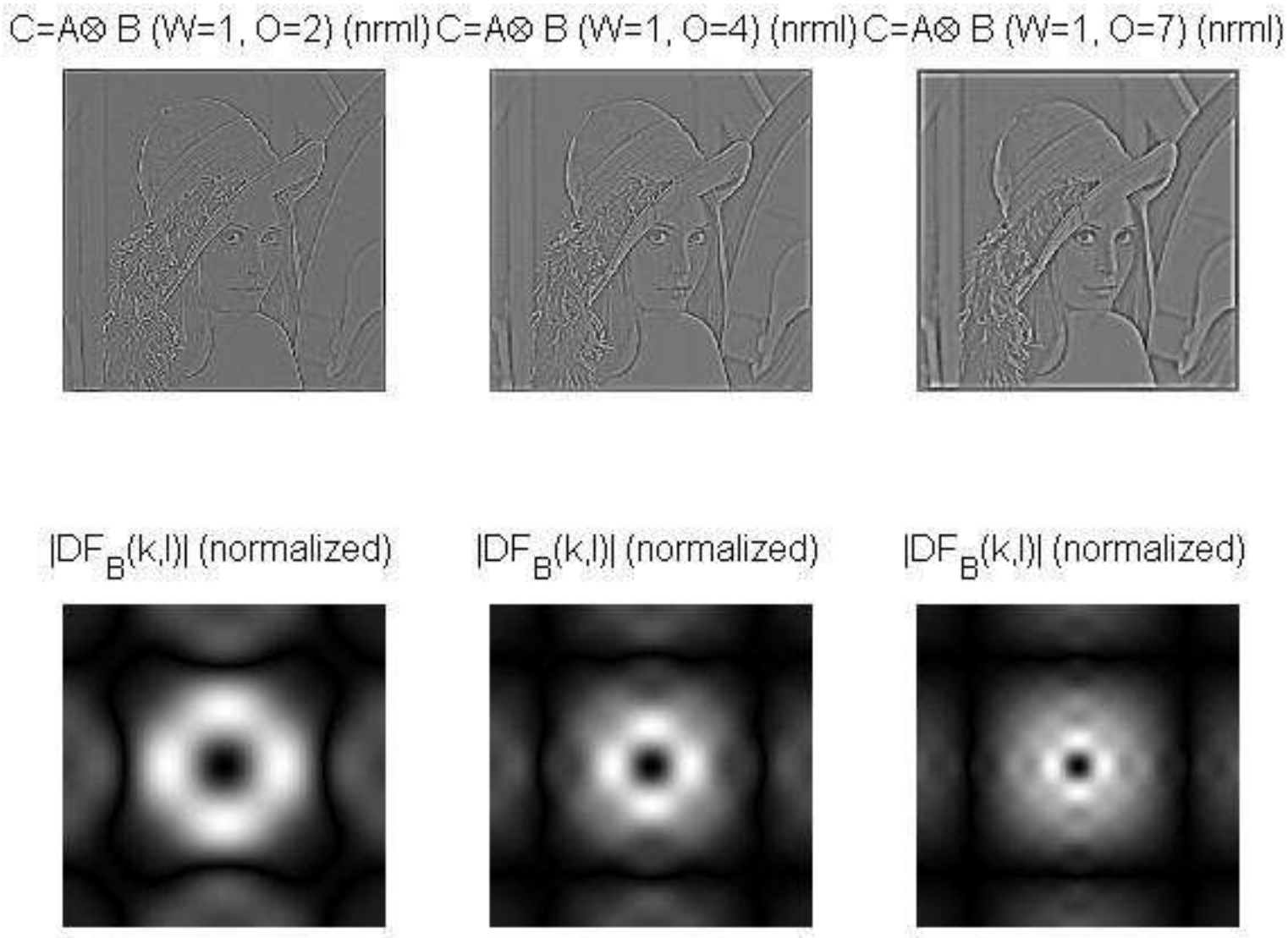
El filtrado pasa-banda de imagen puede pensarse como un filtrado pasa-bajas de imagen menos otro filtrado pasa-baja de imagen:

- Filtrado pasa-banda por ventana de DFT:
  - Si $w_1(k,l)$ $(W_1 \times W_2)$ y $w_2(k,l)$ $(W_1 + O1 \times W_2 + O2)$ son ventanas pasa-bajas de DFT, un filtro pasa-banda de ventana $b(k,l)$ simplemente se define por $b(k,l) = w_2(k,l) - w_1(k,l)$.
- Filtrado pasa-banda en el dominio espacial:
  - Si $\mathbf{L}_1$ (size W) y $\mathbf{L}_2$ (size $W+O$) son filtros pasa-bajas y $\mathbf{L}_2$ pasa "mas-bajas" frecuencias, un filtro pasa-banda $\mathbf{B}$ simplemente se define vía $B(m,n) = L_1(m,n) - L_2(m,n)$.

Filtrado Pasa-Banda por ventanas de DFT



Filtado Pasa-Banda Espacial



Convención de Filtrado

- Note que cuando $\mathbf{A}$ es $(M_1 \times N_1)$, $\mathbf{C} = \mathbf{L} \otimes \mathbf{A}$ es $(M_1 + W - 1 \times N_1 + W - 1)$.
- En general, quisieramos mantener $\mathbf{C}$ del mismo tamaño que $\mathbf{A}$.
- Entonces recortamos una porción adecuada de $\mathbf{C}$ y la consideramos como la imagen filtrada pasa-bajas.
- Para los filtros estudiados, una buena región de recorte es $m = 0, \ldots, M_1 - 1$, etc.

Filtros de Muestreo y "Antialiasing"

Barbara (512x512)

Lenna (512x512)

Muestreo sin Filtrado "Antialiasing"

Barbara S1=S2=4 (128x128)

Lenna S1=S2=4 (128x128)

Barbara S1=S2=8 (64x64)

Lenna S1=S2=8 (64x64)

Muestreo con Filtrado "Antialiasing"

Barbara S1=S2=4 (W1=W2=63)

Lenna S1=S2=4 (W1=W2=63)

Barbara S1=S2=8 (W1=W2=31)

Lenna S1=S2=8 (W1=W2=31)

Remoción de Ruido

Considere el escenario en donde una imagen $\mathbf{A}$ es *corrompida* con ruido aditivo para dar origen a la imagen $\mathbf{B}$:

$$\mathbf{B} = \mathbf{A} + \mathbf{N} \tag{101}$$

A+10*randn(512) (nrml)

A+20*randn(512) (nrml)

A+30*randn(512) (nrml)

Remoción de Ruido en el dominio de la DFT

Ya sabemos que las imágenes naturales tienen coeficientes de DFT dominantes de baja frecuencia. Intuitivamente, podemos hacer la siguientes observaciones.

- Suponiendo que el ruido no se "accesa" a bajas frecuencias esperamos:

$$\begin{aligned} DF_B(k,l) &= DF_A(k,l) + DF_N(k,l) \\ DF_B(k,l) &\tilde{=} DF_A(k,l) \end{aligned} \tag{102}$$

  ya que $|DF_A(k,l)|$ es grande a bajas frecuencias.

- A altas frecuencias esperamos:

$$\begin{aligned} DF_B(k,l) &= DF_A(k,l) + DF_N(k,l) \\ DF_B(k,l) &\tilde{=} DF_N(k,l) \end{aligned} \tag{103}$$

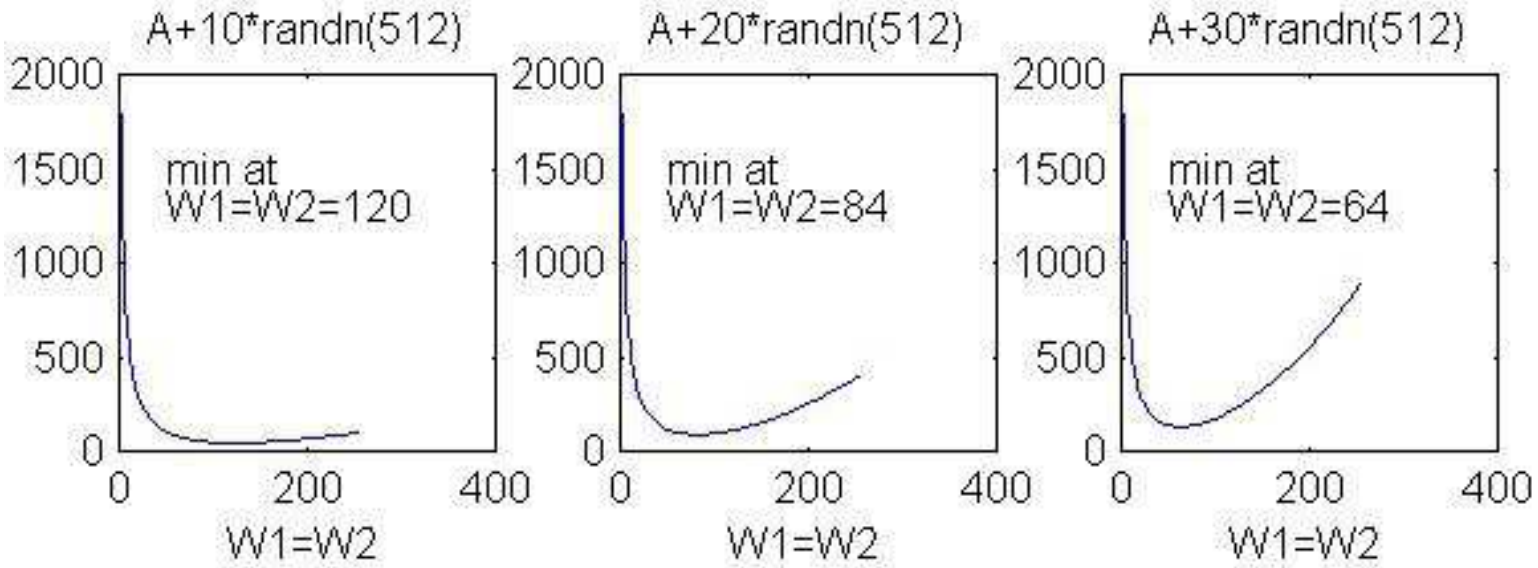
  ya que $|DF_A(k,l)|$ es pequeña a altas frecuencias.

- Podemos reducir la cantidad de ruido en $\mathbf{B}$ por un filtrado pasa-bajas.

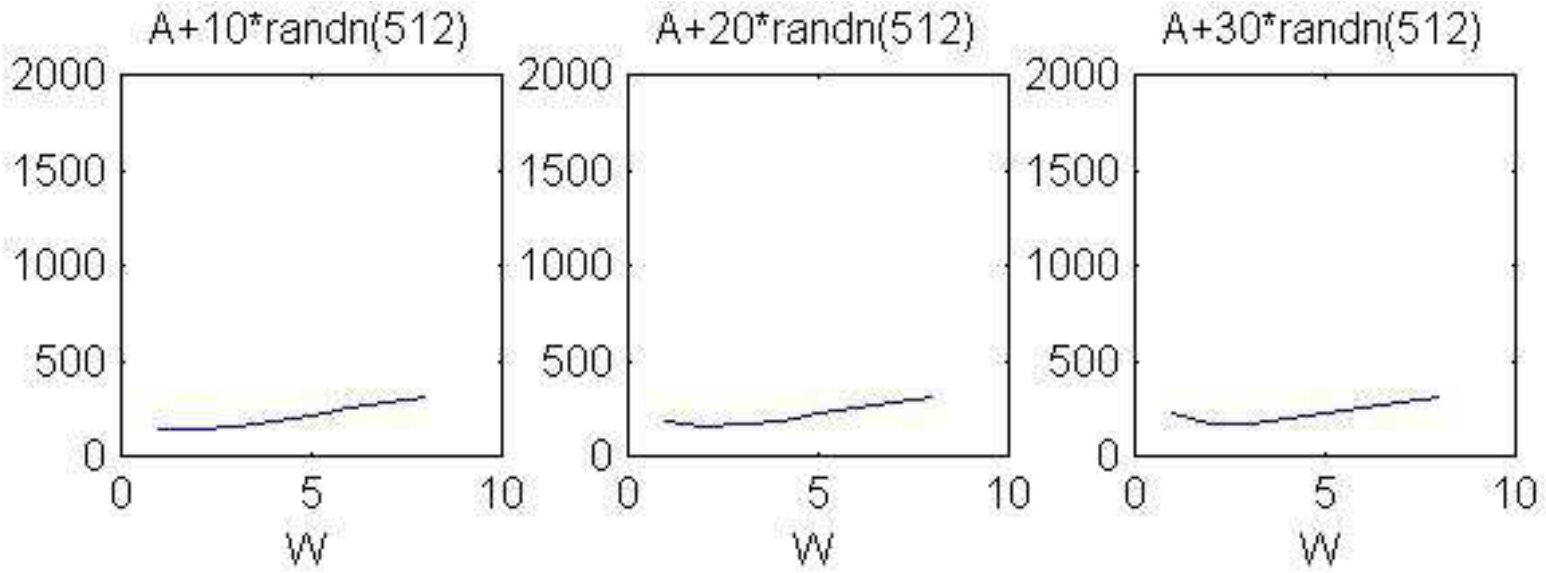
Remoción de Ruido por Filtrado Pasa-Bajas

- Dada una imagen ruidosa, le aplicamos un filtrado pasa-bajas para obtener $\mathbf{C} = \mathbf{B} \otimes \mathbf{L}$ o $DF_C(k,l) = DF_B(k,l)w(k,l)$, en donde $\mathbf{L}$ es un filtro pasa-bajas y $w(k,l)$ es una ventana pasa-bajas de DFT.
- En general, determinar los parámetros del filtro es dificil y se hace por prueba/error (digamos juzgando la calidad visual de $\mathbf{C}$) o basados en suposiciones/modelos.
- Para propósitos ilustrativos determinaremos los *mejores* parámetros para nuestros filtros basados en el error cuadrático medio entre $\mathbf{C}$ y $\mathbf{A}$. Note que esto no es posible en la práctica porque no se tiene acceso a la imagen original.

Remoción de Ruido por ventana de DFT



Remoción de Ruido por Filtrado Espacial Pasa-Bajas



Resumen

- En esta Clase aprendimos como aplicar filtros pasa-bajas, pasa-altas y pasa-banda a imágenes en dos formas diferentes.
- Consideramos dos aplicaciones de filtrado:
  - Submuestreo por filtros pasa-bajas "antialiasing".
  - Reducción/remoción de ruido por filtros pasa-bajas.

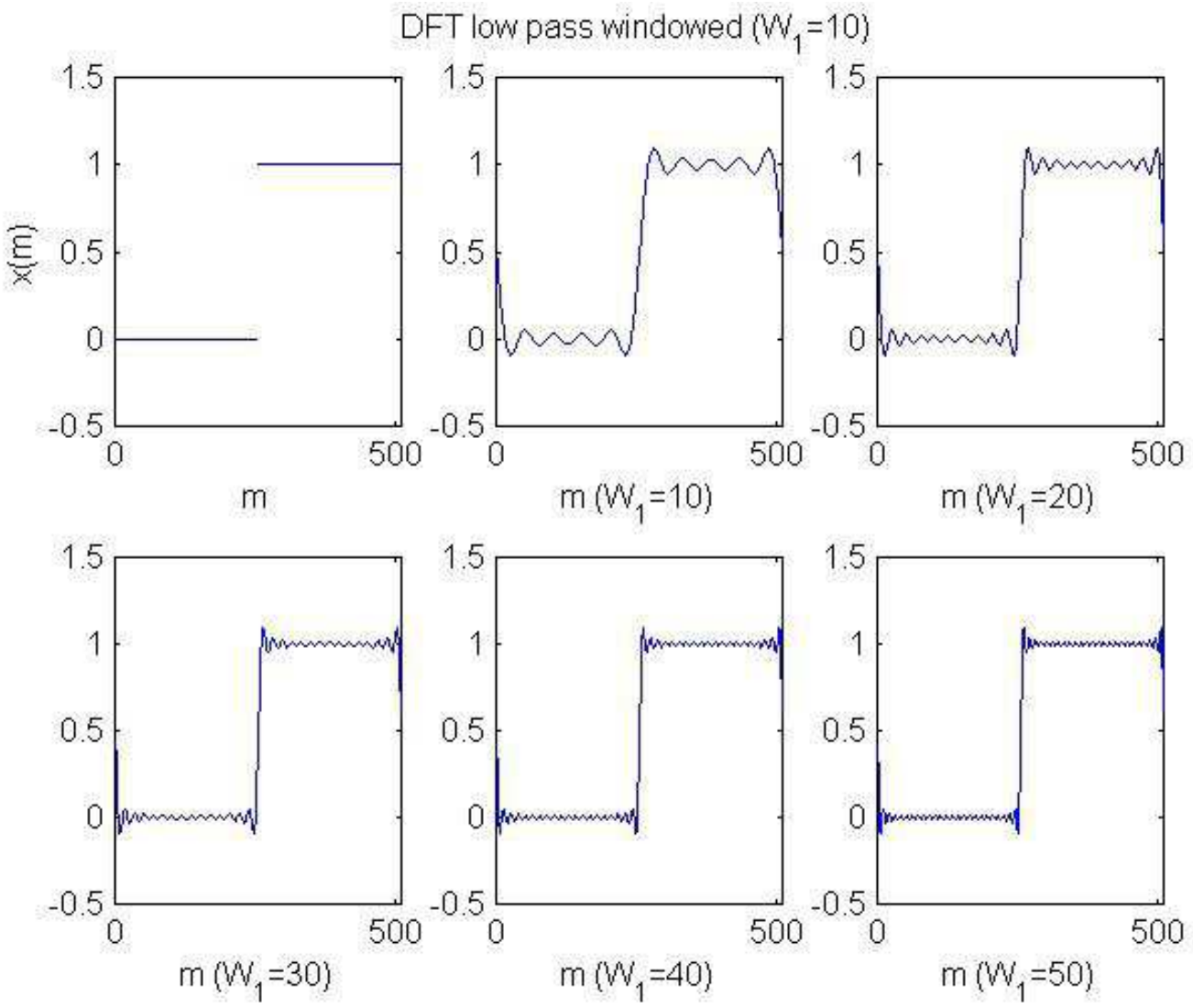
Tarea VIII

1. Aplique a su imagen filtrados pasa-bajas, pasa-banda y pasa-altas espacialmente y con ventana de DFT. Use por lo menos tres parámetros diferentes para filtrado pasa-bajas, pasa-banda y pasa-altas. Muestre sus resultados de la misma manera en que se han presentado en esta clase (ver por ejemplo las páginas 10 y 11).
2. Submuestree su imagen por 4 y 8 en cada dirección, con y sin "antialiasing" usando ventanas pasa-bajas de DFT. Asegurese de elegir los parámetros correctos para las ventanas. (Tip: sus imágenes no son cuadradas). Muestre los parámetros utilizados así como las imágenes resultantes. Comente sus resultados.
3. Haga el procesamiento de reducción de ruido como se hizo en las páginas 22 y 23. Inicie agregando ruido a su imagen etc. (Vea el Tip previo para parámetros de la ventana DFT pasa-bajas.) Muestre sus resultados como se presentaron en la Clase.

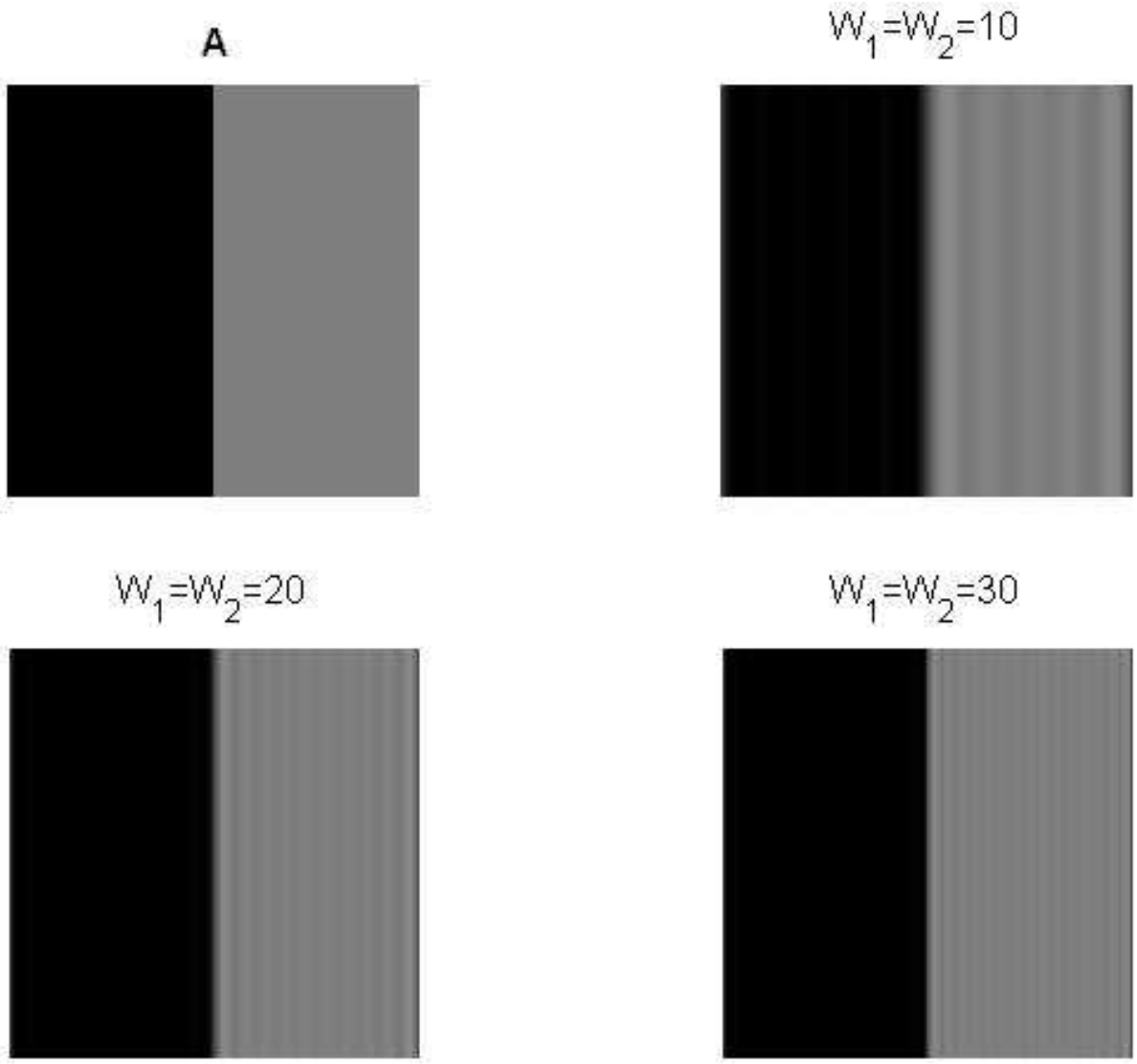
Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

Transformada de Fourier y Fenómeno de Gibbs

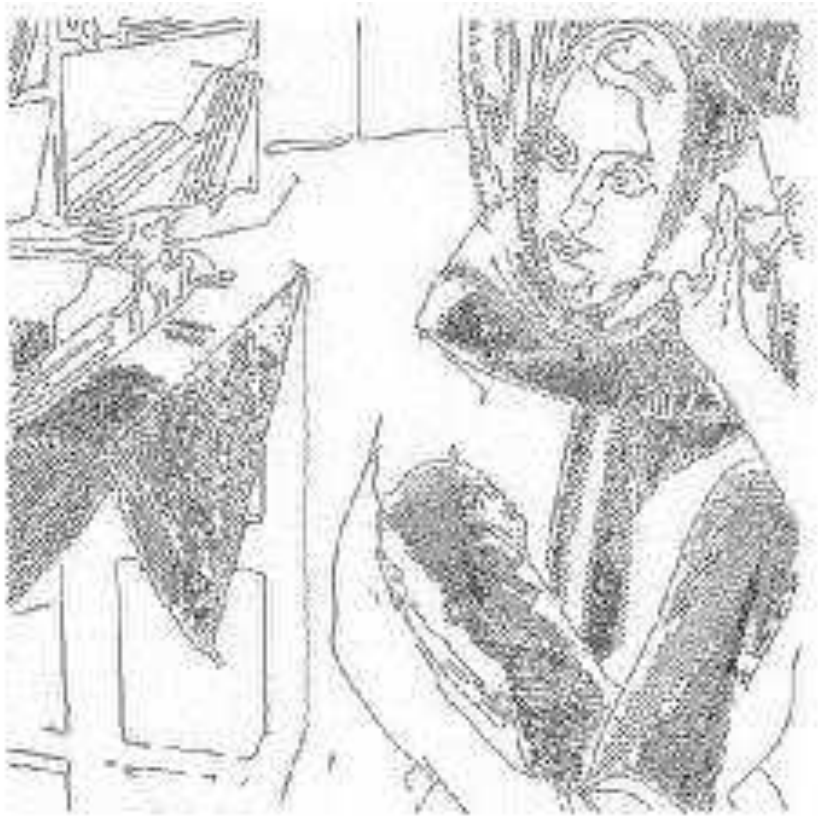


Transformada de Fourier y Fenómeno de Gibbs cont.

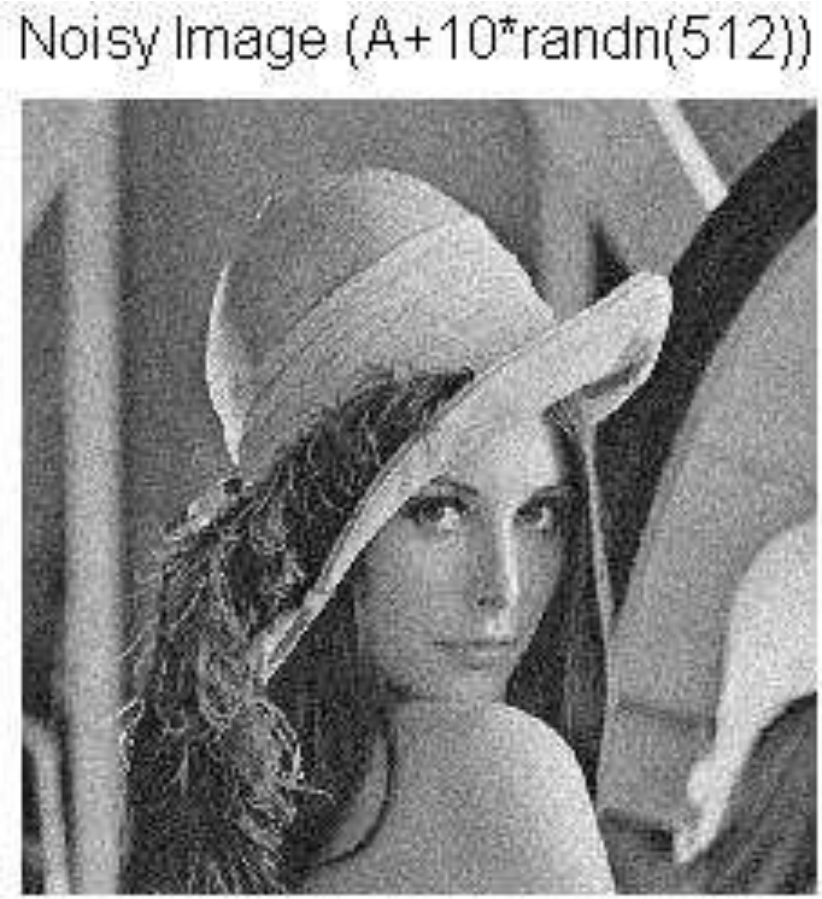


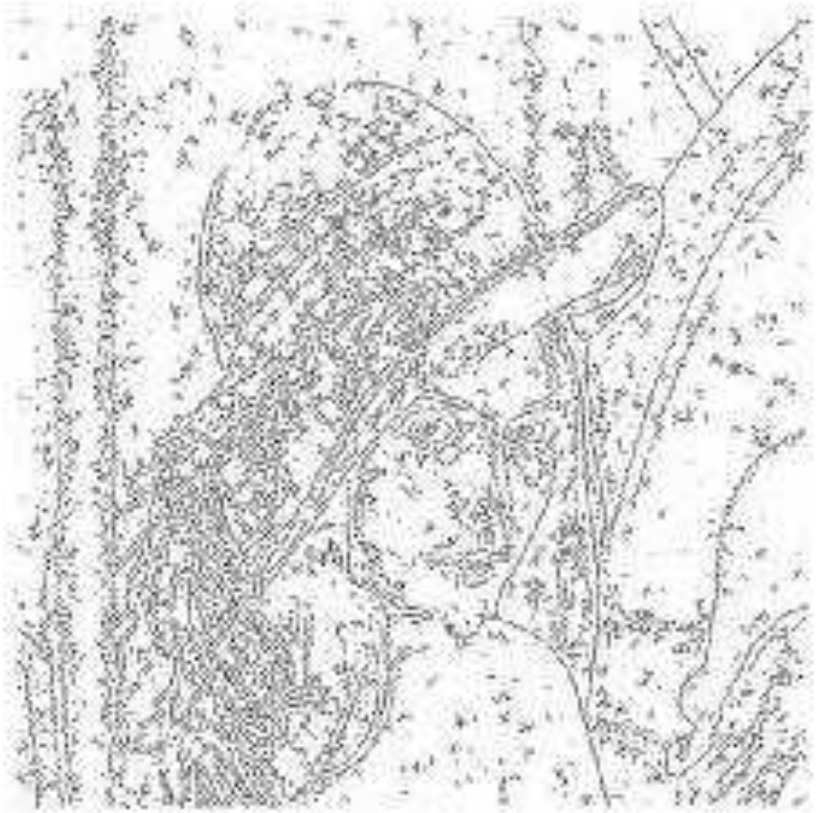
Imágenes y Bordes

Imágenes y Bordes cont.

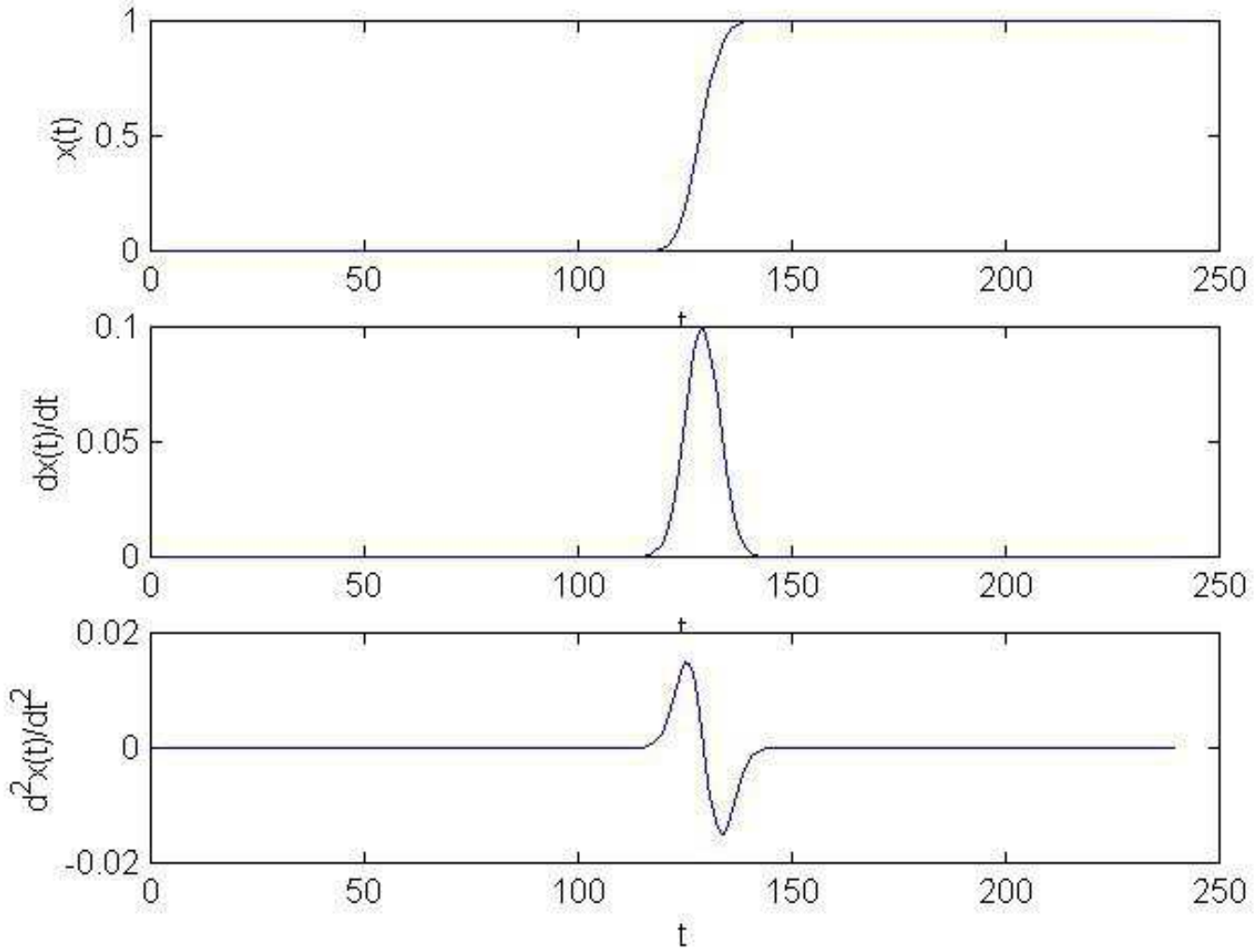


Bordes y Ruido





Detección de Bordes - Motivación



Algoritmos de Detección de Bordes - Tiempo Continuo

Suponga que $x(t)$ es una señal de tiempo continua que contiene "bordes". Los bordes pueden ser detectados en dos formas, vía:

1. Calculando $y(t) = \frac{dx(t)}{dt}$ y encontrando las localidades t_i en donde $|y(t)|$ tiene un máximo local, i.e., hay un borde en la localidad t_i si:

$$|y(t_i)| - |y(t_i + \epsilon)| \geq 0 \ , \qquad \forall \ 0 < |\epsilon| << 1$$

   De manera que se puedan incorporar señales "no-ideales" diremos que hay un borde en la localidad t_i si $|y(t_i)| > T_e$ en donde $T_e > 0$ es un umbral o "la fortaleza del borde'. Sin embargo, aún nos referiremos a este algoritmo como dección de máximos locales.

Algoritmos de Detección de Bordes - cont.

2. Calculando $y(t) = \frac{d^2x(t)}{dt^2}$ y encontrando las localidades t_i de los cruces por cero de $y(t)$, i.e., hay un borde en la localidad t_i si:

$$y(t_i + \epsilon) > 0 \Rightarrow y(t_i - \epsilon) < 0, \qquad \forall\ 0 < \epsilon << 1$$

o

$$y(t_i + \epsilon) < 0 \Rightarrow y(t_i - \epsilon) > 0, \qquad \forall\ 0 < \epsilon << 1$$

Sea

$$c(t) = \begin{cases} y(t) & |y(t)| > T_e \\ 0 & \text{cualquier otro} \end{cases} \tag{104}$$

en donde $T_e > 0$ es un umbral. Para incorporar señales "no-ideales" diremos que hay un borde en la localidad t_i si $c(t)$ tiene un cruce por cero en t_i. Nos seguiremos refiriendo a este algoritmo como detección de cruces por cero.

Algoritmos de Detección de Bordes - Tiempo Discreto

- Podemos simplemente aproximar las derivadas d/dt y d^2/dt^2 por aproximaciones discretas. Estas aproximaciones son llamadas operadores gradiente.
- Las imágenes son bidimensionales. Si solo incorporamos operadores gradiente de "renglón" solo podremos detectar bordes verticales:
  - Hacer dos pasos: Detectar bordes verticales y luego detectar bordes horizontales.
  - Podemos definir operadores gradiente "sin dirección" que detecten bordes horizontales y verticales en un paso.

Algoritmo General

Algoritmo general de detección de bordes:

1. Aplicar un operador gradiente a la imagen $\mathbf{A}$.
2. Detectar máximos locales o cruces por cero (con T_e).
3. Generar un "mapa de bordes" $\mathbf{E}_A$ en donde

$$E_A(m,n) = \begin{cases} 255 & \text{pixel } (m,n) \text{ es un pixel de borde} \\ 0 & \text{cualquier otro} \end{cases} \tag{105}$$

4. Si se utiliza una aproximación de dos pasos para obtener los mapas de borde horizontal $\mathbf{E}_A^h$ y vertical $\mathbf{E}_A^v$ se puede definir un mapa general por:

$$E_A(m,n) = \begin{cases} 255 & E_A^h(m,n) = 255 \text{ o } E_A^v(m,n) = 255 \\ 0 & \text{cualquier otro} \end{cases} \tag{106}$$

Operadores Gradiente

- Los operadores gradiente son definidos como filtros y la aplicación de los operadores gradiente es implementada por filtrado lineal de la imagen.
- Note sin embargo que la detección de bordes involucra un paso no lineal en donde detectamos los máximos locales por operadores "derivada" o cruce por cero para operadores de "segunda derivada" en la salida filtrada linealmente.

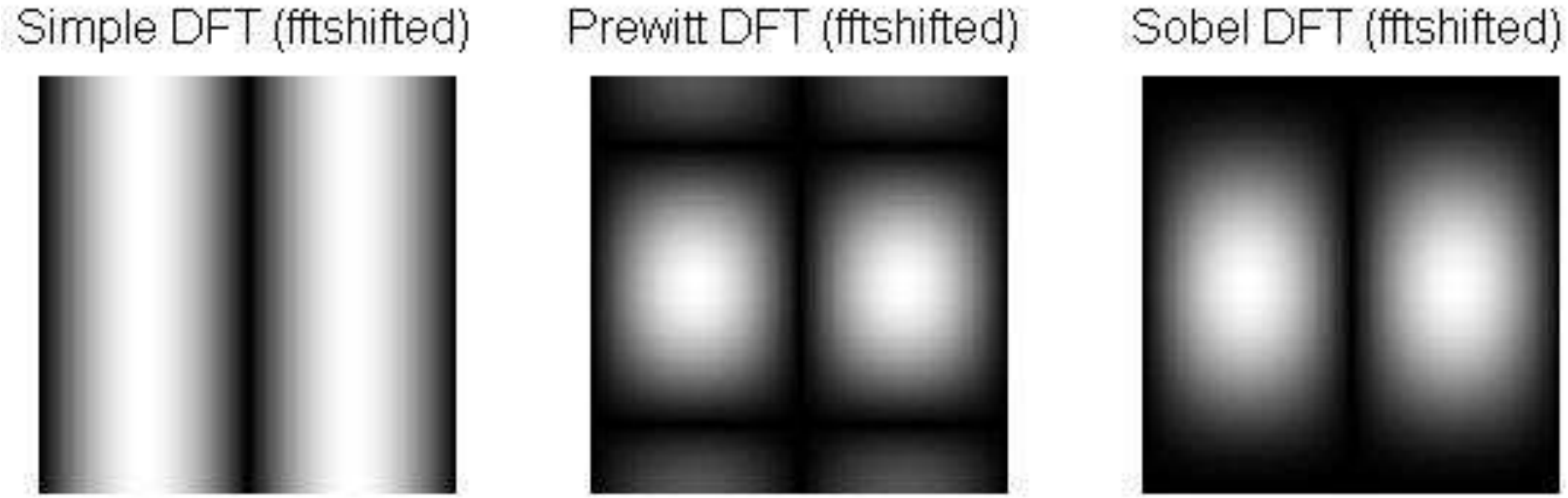
Operadores Gradiente cont.

- Máximos Locales: La determinación de máximos locales en dos dimensiones se establece al comparar los valores absolutos de la salida filtrada en (m, n) a T_e.
- Cruces por cero: Sea $\mathbf{B}$ que denota la imagen filtrada y umbralizada (con T_e).
  - Si $B(m, n) = 0$ y $\texttt{sign}(B(m, n-1)B(m, n+1)) = -1$ entonces hay un borde vertical en el pixel (m, n), etc.
  - Si $B(m, n) \neq 0$ y $\texttt{sign}(B(m, n)B(m, n+1)) = -1$ entonces hay un borde vertical entre los pixeles (m, n) y $(m, n+1)$.
- Cruces por Cero para Operaciones sin Dirección: En una rejilla bidimensional, se dice que ocurre un cruce por cero siempre que hay un cruce por cero en *al menos* una dirección.

Operadores de Primera Derivada

- Simple: $[-1 \quad 0 \quad 1]$, Prewitt: $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$, Sobel: $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

Los operadores para bordes horizontales pueden ser obtenidos al transponer los filtros.



Ejemplo - Bordes Verticales

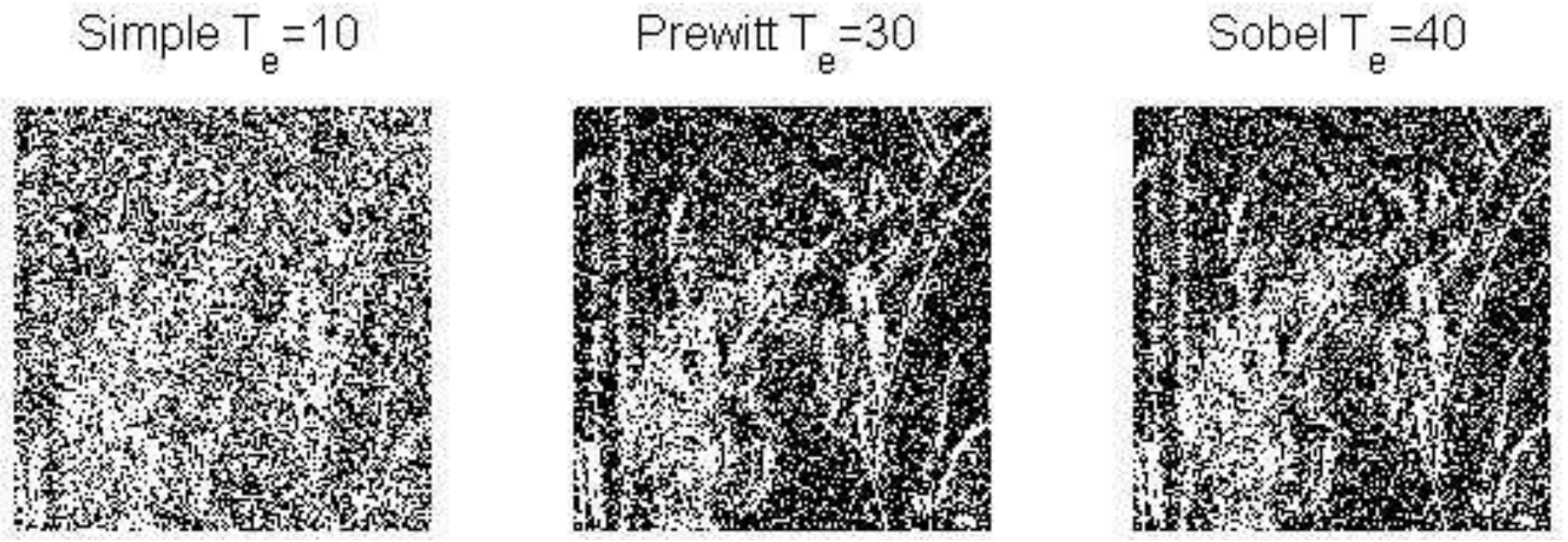
Simple T_e=10

Prewitt T_e=30

Sobel T_e=40

Detección de bordes verticales en Lenna.

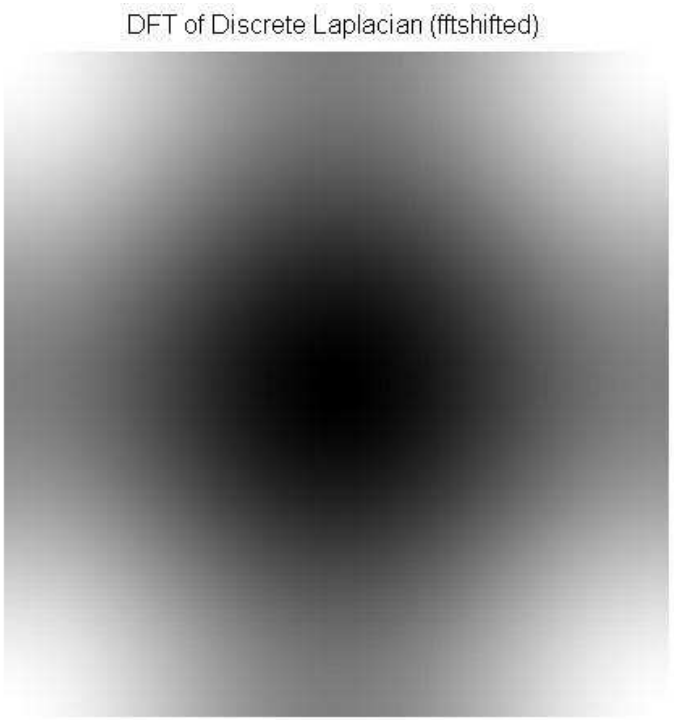
Ejemplo - Bordes Verticales cont.



Detección de bordes verticales en Lenna+`10*randn`(512).

Operadores de Segunda Derivada Sin Dirección

- Se puede demostrar que el Operador Laplaciano $\nabla^2 f = \partial^2 f/\partial^2 x + \partial^2 f/\partial^2 y$ es rotacionalmente simétrico.
- Por tanto, podemos esperar una aproximación discreta a este operador que sea aproximadamente rotacionalmente simétrico.
- Tal aproximación es $Z(m,n) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$



Detección de Bordes y Ruido

- Los operadores derivativos $D(m,n)$ usados en detección de bordes son susceptibles a ruido.
- Una estrategia sería aplicar en primer lugar un filtrado pasa-bajas ($L(m,n)$) a una imagen (para reducir el ruido) y luego emplear el operador derivativo.
- En la práctica se eligen filtros espaciales pasa-bajas para evitar el fenómeno de Gibbs.
- Sea $A(m,n)$ que denota la imagen original. Entonces tenemos $\mathbf{C} = \mathbf{D} \otimes \mathbf{L} \otimes \mathbf{A}$ como la imagen filtrada por completo en la cual se pueden hacer decisiones no lineales.
- Se puede entonces definir un operador gradiente combinado $\mathbf{H} = \mathbf{D} \otimes \mathbf{L}$.

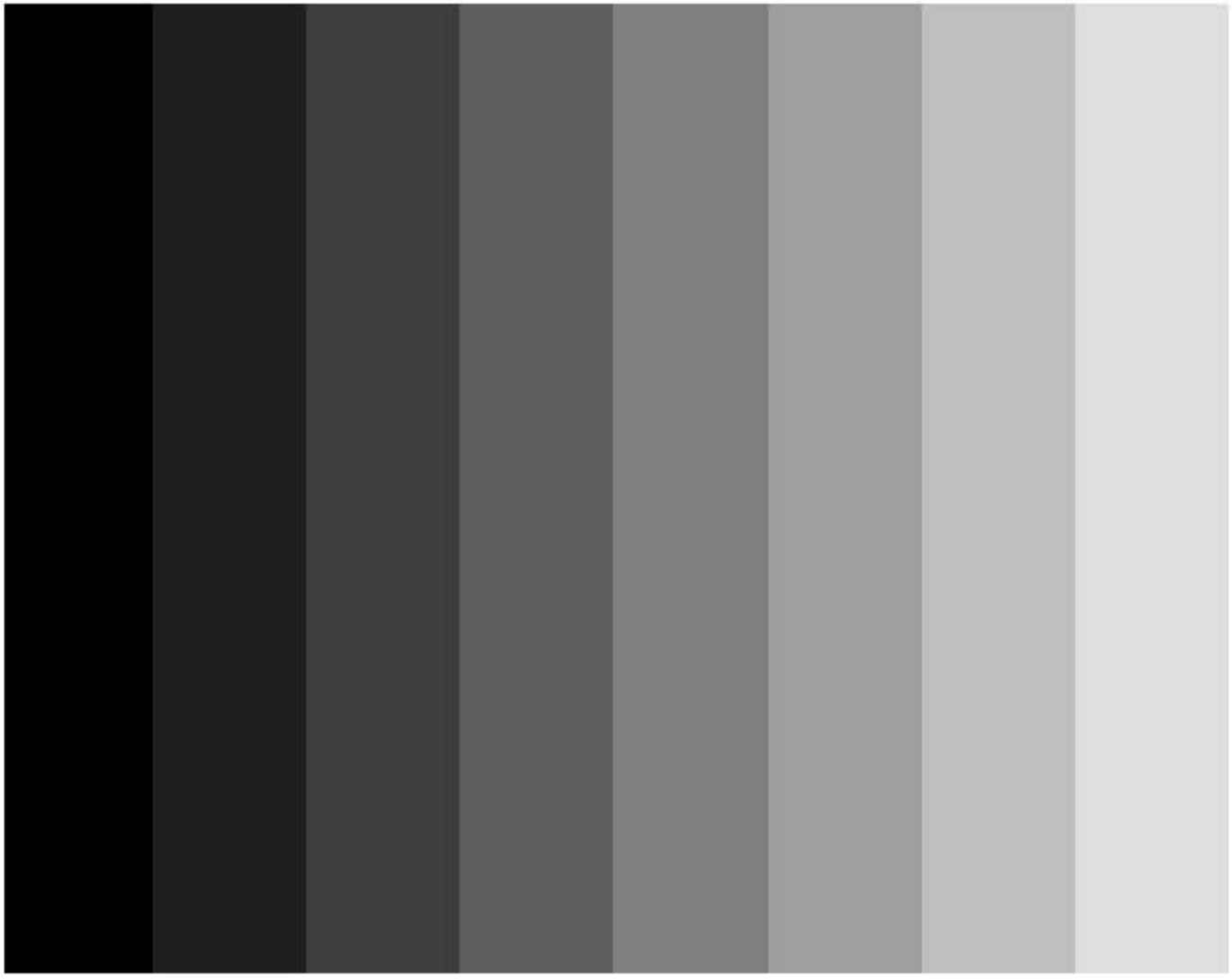
El Laplaciano de un Operador Gradiente Gaussiano

- La secuencia Gaussiana definida por $L(m,n) = e^{-(m^2+n^2)/2\sigma^2}$ es una secuencia pasa-bajas y puede ser utilizada como un filtro pasa-bajas (σ controla el ancho de banda de este filtro).
- El Laplaciano de una Gaussiana $\mathbf{H} = \mathbf{Z} \otimes \mathbf{L}$ puede entonces ser definido como un operador gradiente de segunda derivada con desempeño favorable bajo condiciones ruidosas.
- σ es elegida grande para condiciones muy ruidosas y pequeña para condiciones relativamente sin ruido.
- Típicamente $L(m,n)$ es empleada como una secuencia de extensión finita $P_1 \times P_2$ (m,n debe estar centrada alrededor de cero $-1 \leq m,n \leq 1$ etc.). Grandes valores de σ en general requerirán de extensiones más grandes.
- Los bordes son detectados utilizando los cruces por cero de la salida filtrada (con T_e).

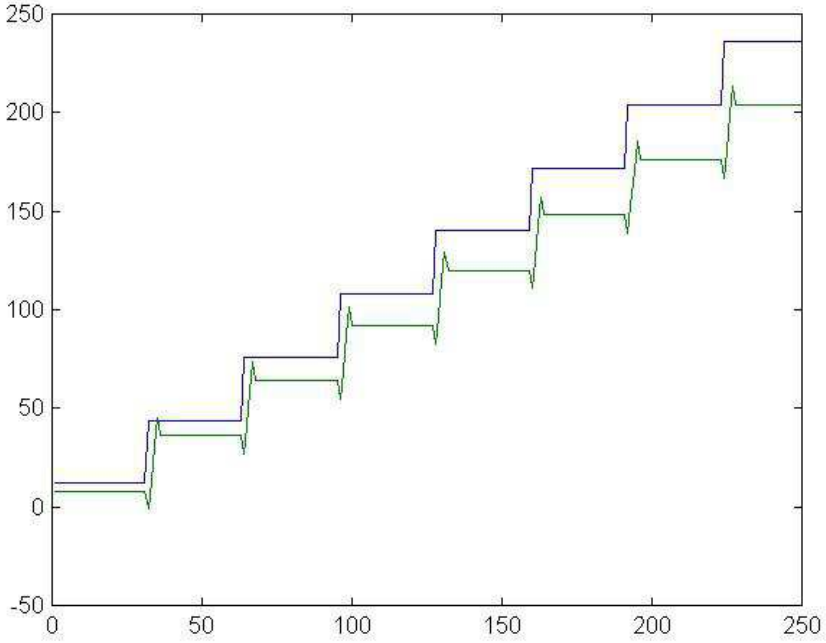
Ejemplo

$P_1 = P_2 = 3, T_e = 30, \sigma = 2$

Sistema Visual Human y Bandas Mach



Respuesta Pasa-Bajas del Sistema Visual Humano



- El sistema visual humano hace un filtrado pasa-bajas de las escenas bajo observación.
- Se explotará esta observación "escondiendo" errores de procesamiento donde los humanos no pueden verlos.

Resumen

- En esta Clase aprendimos acerca del fenómeno de Gibbs.
- Aprendimos la detección de bordes con:
  - Operadores gradiente basados en primeras derivadas y máximos locales asociados.
  - Operadores gradiente basados en segundas derivadas y cruces por cero asociados.

  (Leer páginas 347-353 del libro de texto).
- Aprendimos acerca de las bandas Mach y la naturaleza pas-bajas del sistema visual humano (leer páginas 51-56 del libro de texto).

Tarea IX

1. Detecte bordes horizontales y verticales en su imagen usando los operadores simple, Prewitt y Sobel. Muestre los mapas de bordes horizontales, bordes verticales y bordes combinados. Comente sobre las diferencias entre mapas de bordes horizontales y bordes verticales. Experimente con los umbrales relevantes y trate de escogerlos lo mejor que pueda.

2. Detecte bordes en su imagen usando el cruce por cero del operador Laplaciano de una Gaussiana. Ajuste P_1, P_2, T_e, σ para obtener los mejores resultados.

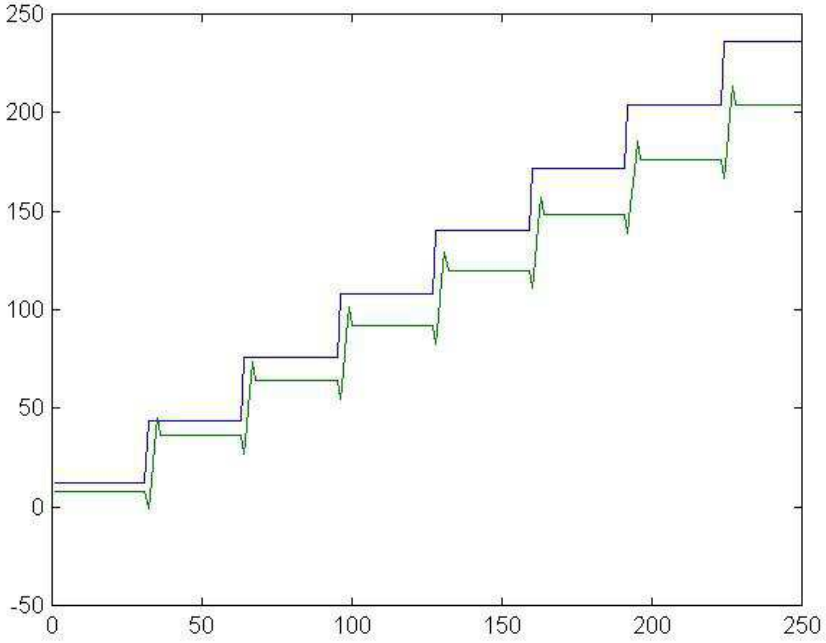
3. Repita el inciso 2 comenzando con una versión ruidosa de su imagen (digamos con imagen + `10randn()`). Trate de ajustar los parámetros para empatar el desempeño sin ruido. Comente las diferencias de los resultados con el inciso 2.

4. Genere una señal unidimensional y su versión percibida como se hizo en la Clase. Trate de encontrar un filtro, el cual cuando se convolucione con la señal original de como resultado la señal "percibida". Tip: Use la DFT de las dos señales y use la propiedad de convolución $\rightarrow$ multiplicación. Tenga cuidado de las divisiones por cero o valores pequeños. ¿Es pasa-bajas el filtro que descubrió? Muestre la DFT 1D y el filtro.

Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.

Respuesta pasa-bajas del Sistema Visual Humano



- El sistema visual humano filtra de forma pasa-bajas las escenas bajo observación.

Procesamiento de Imágenes "Perceptual"

- El sistema visual humano tiene una respuesta pasa-bajas, i.e., cuando los humanos miran una cierta imagen no ven completamente los "detalles" que se manifiestan a altas frecuencias.
- Note que el filtrado pasa-bajas $\sim$ promediado.
- Tomando ventaja de esto se "empujan" ciertos errores del procesamiento hacia altas frecuencias en donde los humanos no pueden mirarlos.
- El procesamiento de imágenes perceptual asume que los resultados del procesamiento serán vistos por humanos. Las técnicas de procesamiento son dirigidas hacia esta suposición particular.
- Los resultados del procesamiento de imágenes perceptual son buenos para evaluaciones subjetivas y usualmente no son buenas para evaluaciones objetivas (tales como mse).

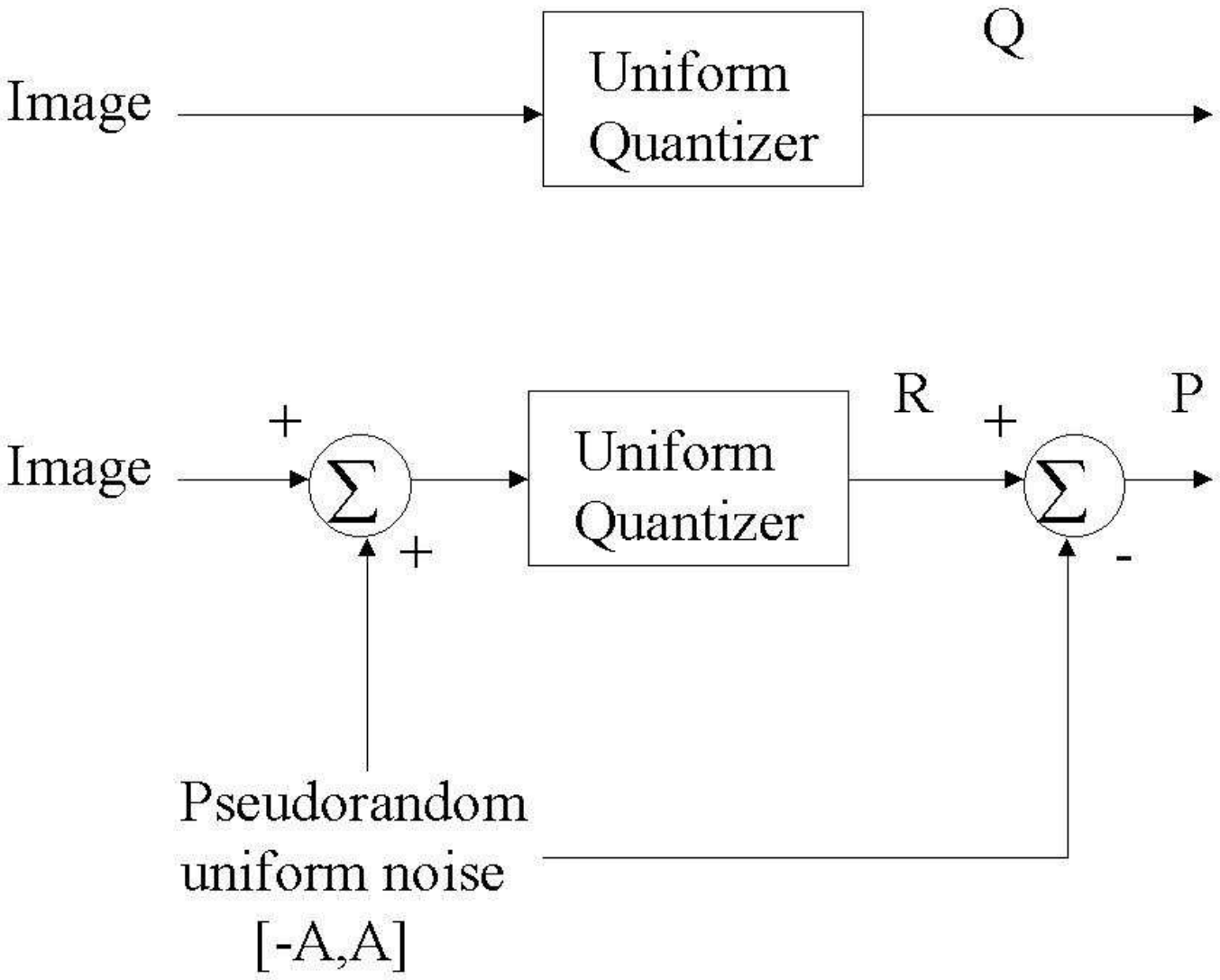
Cuantización y Falsos Contornos

Ya sabemos que los cuantizadores "burdos" producen falsos contornos en la imagen cuantizada, la cual no es agradable al observador humano.

Lenna

Uniform Q. Δ=64

Cuantización de Ruido Pseudo-aleatorio Interpolado



Cuantización de Ruido Pseudo-aleatorio Interpolado cont.

- En regiones de variación suave de los valores de pixeles en general se obtienen falsos contornos después de una cuantización "burda" (- no fina-).
- Con la cuantización interpolada con ruido, el ruido aditivo causa que los valores de los pixeles en esas regiones vayan por debajo o encima del umbral de cuantización, por lo que rompen los contornos.
- A se elige de tal forma que solo los bits menos significativos son afectados en la cuantización.

Ejemplo

Q (Δ= 32)

R (A=32)

P (A=32)

Ejemplo

Q (Δ= 64)

R (A=48)

P (A=48)

Ejemplo

Q (Δ= 128)

R (A=48)

P (A=48)

Medio tono de Imagen

- Algunos dispositivos de despliegue como impresoras solo pueden desplegar pixeles en blanco y negro, i.e., cuando una imagen se imprime esta incluye una cuantización muy burda.
- Estos dispositivos compensan dicha limitación teniendo la habilidad de desplegar muchos pixeles o "puntos" en áreas muy pequeñas.
- El tipo de algoritmos de medio tono en los que estamos interesados sobre muestrean una imagen, agregan ruido pseudo-aleatorio y cuantizan el resultado a dos niveles (i.e., obtenga $\mathbf{R}$ usando una imagen sobre muestreada.) Vea las páginas 120-122 del libro de texto.

Deformación de Imagen y Efectos Especiales

Sea $A(i,j)$ una imagen.

- En la deformación de imagen nuestro interés es generar una imagen $B(k,l)$ a partir de $A(i,j)$ en donde

$$B(k,l) = A(x(k), y(l))$$

- Llamaremos $x(\ldots)$ y $y(\ldots)$ las funciones de deformación de pixel.
- En las siguientes diapositivas se generarán imágenes $\mathbf{B}$ a partir de $\mathbf{A}$ utilizando diferentes funciones de deformación.
- Hay dos casos especiales:
  1. $x(k), y(l)$ están fuera de los "límites" de $\mathbf{A}$. En este caso simplemente se hace $B(k,l) = 0$.
  2. Uno o ambos $x(k), y(l)$ no son enteros. En este caso simplemente encontramos los dos o cuatro pixeles más cercanos en $\mathbf{A}$, promediando los valores de pixel correspondientes para obtener $B(k,l)$. (Además podemos redondear).

Traslación y Rotación

$$\begin{aligned}
\text{Traslación:} \quad & x(k,l) = k + 50; y(k,l) = l; \\
\text{Rotación:} \quad & x(k,l) = (k - x_0)\texttt{cos}(\theta) + (l - y_0)\texttt{sin}(\theta) + x_0; \\
& y(k,l) = -(k - x_0)\texttt{sin}(\theta) + (l - y_0)\texttt{cos}(\theta) + y_0;
\end{aligned}$$

$x_0 = y_0 = 256.5$ el centro de la imagen $\mathbf{A}$, $\theta = \pi/6$

“Ondas”

wave 1

wave 2

$$\text{wave1:} \quad x(k,l) = k + 20\texttt{sin}\left(\frac{2\pi}{128}l\right); y(k,l) = l;$$

$$\text{wave2:} \quad x(k,l) = k + 20\texttt{sin}\left(\frac{2\pi}{30}k\right); y(k,l) = l;$$

“Pandeo” y “Remolino”

$$\text{warp}: \quad x(k,l) = \texttt{sign}(k-x_0)/x_0 * (k-x_0)^2 + x_0; y(k,l) = l;$$

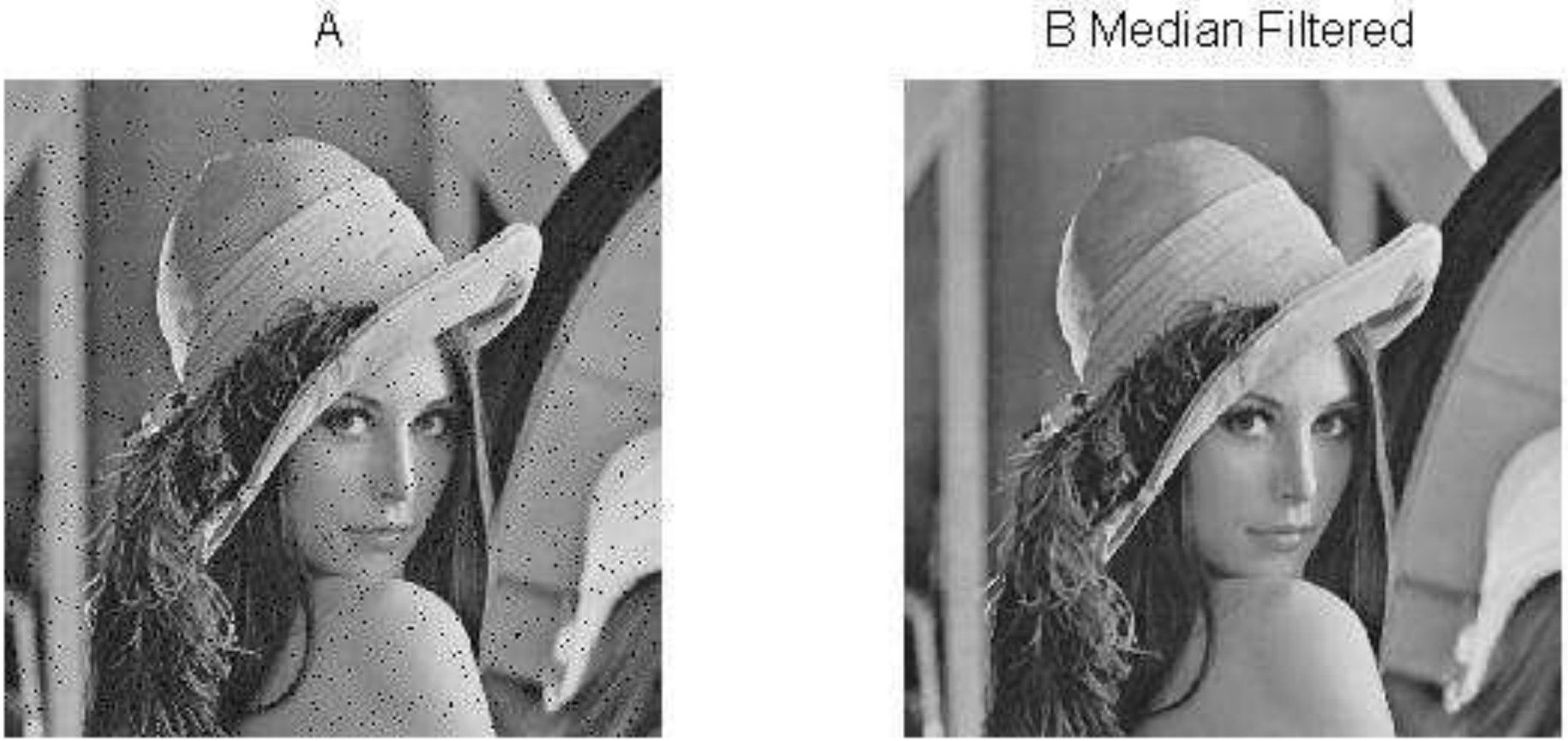
$$\begin{aligned}\text{swirl}: \quad & x(k,l) = (k-x_0)\texttt{cos}(\theta) + (l-y_0)\texttt{sin}(\theta) + x_0; \\ & y(k,l) = -(k-x_0)\texttt{sin}(\theta) + (l-y_0)\texttt{cos}(\theta) + y_0; \\ & r = ((k-x_0)^2 + (l-y_0)^2)^{1/2}, \ \theta = \pi/512 \ r\end{aligned}$$

$x_0 = y_0 = 256.5$ el centro de la imagen **A**

“Vidrio”

$$x(k,l) = k + (\texttt{rand}(1,1) - .5) * 10;$$
$$y(k,l) = l + (\texttt{rand}(1,1) - .5) * 10;$$

Filtrado de Mediana



Se define el conjunto $Z(i,j) = \{A(m,n) | i-W \leq m \leq i+W,\ j-W \leq n \leq j+W\}$. Entonces $B(i,j) = \texttt{median}(Z(i,j))$.

Pintura al óleo

Se define el conjunto $Z(i,j) = \{A(m,n) | i - W \leq m \leq i + W,\ j - W \leq n \leq j + W\}$. Entonces $B(i,j) =$ el valor más frecuente en $Z(i,j)$.

Tarea X

1. Implemente el cuantizador de interpolación de ruido pseudo-aleatorio con $\Delta = 32,\ 64,\ 128$. En cada caso trate de elegir el "mejor" parámetro posible de ruido A.
2. Implemente las funciones de deformación para traslación, rotación, onda, pandeo, remolino y vidrio. Trate de usar diferentes parámetros. Los parámetros utilizados en las diapositivas x_0, y_0, etc., estan ajustados a una imagen de 512×512.
3. Implemente el filtro de mediana para su imagen. En primer lugar elija 4000 pixeles al azar y ajuste sus valores a cero para obtener una imagen corrompida con ruido tipo pimienta. Luego aplique el filtro de mediana con $W = 3$. Repita con 40000 pixeles corrompidos. ¿Qué puede hacer ahora para mejorar los resultados? (Tip: Trate de incrementar W y/o corra varias iteraciones del filtrado de mediana.)

Referencias

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.