



Universidad  
Rey Juan Carlos

Escuela de Ingeniería  
de Fuenlabrada



# Visión Artificial

## 4. Transformaciones y correcciones

JOSÉ MIGUEL GUERRERO HERNÁNDEZ

E MAIL: [JOSEMIGUEL.GUERRERO@URJC.ES](mailto:JOSEMIGUEL.GUERRERO@URJC.ES)

# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

# Índice de contenidos

---

## 1. Transformaciones geométricas:

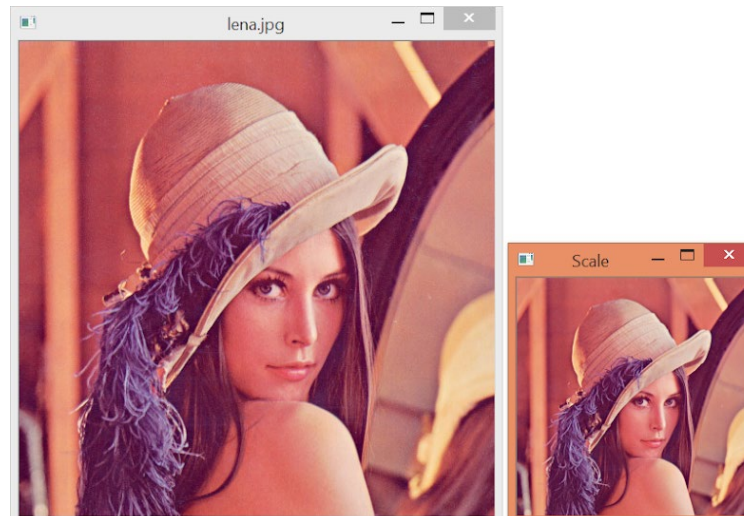
- Transformadas elementales
- Sistema de coordenadas
- Interpolación
- Registro
- Ejemplos

## 2. Transformaciones radiométricas:

- Suavizado
- Histograma
- Filtrado homomórfico
- Correspondencia de histogramas
- Comparación de histogramas
- Ejemplos

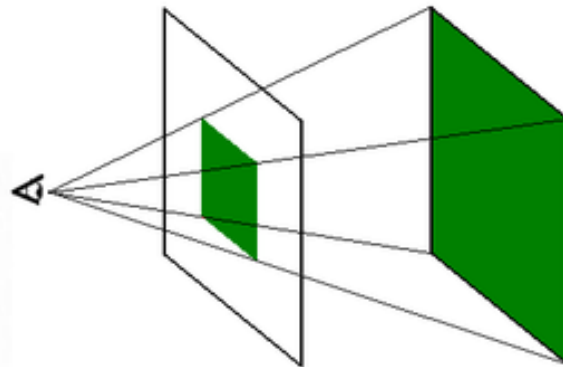
# 1. Transformación geométrica

- Las transformaciones geométricas modifican las coordenadas
- Estas transformaciones cambian la posición, rotación, escala, o inclinación de una imagen
- Este tipo de transformación no cambia el contenido de la imagen, la deformación se produce por un cambio en la posición de los píxeles que la componen



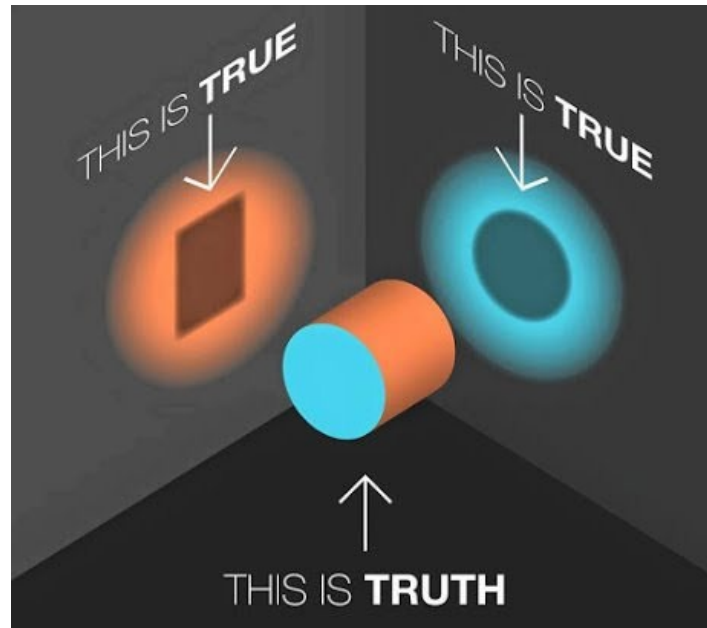
# 1. Transformación geométrica

- La geometría proyectiva se define en el contexto de la geometría euclidiana y tiene sus orígenes en la pintura del Renacimiento, al investigar la visión que nuestro ojo tiene de una figura cuando la vemos en distintos planos colocados entre ella y nosotros (**perspectiva**)



# 1. Transformación geométrica

- El **espacio proyectivo** se relaciona con la forma en la que un ojo o una cámara proyecta una escena 3D sobre una imagen 2D



# 1. Transformación geométrica

---

- Las **coordenadas homogéneas** son un instrumento usado en geometría para describir un punto en el espacio proyectivo
- Fueron introducidas por el matemático alemán August Ferdinand Möbius en el año 1837
- La representación mediante coordenadas homogéneas de objetos en un espacio dimensional se realiza a través de coordenadas de un espacio **(n+1)-dimensional**
- Las coordenadas homogéneas nos van a permitir realizar transformaciones de manera sencilla al trabajar con matrices

# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos



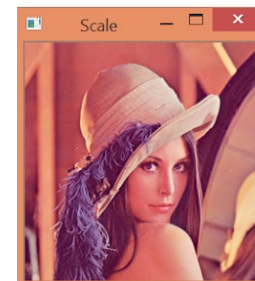
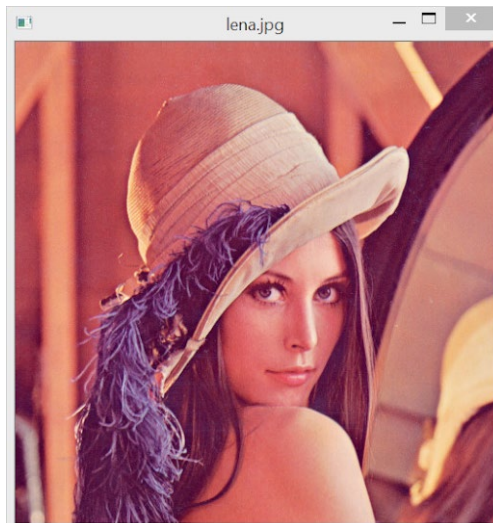
# 1.1. Transformaciones elementales

- Escalado:

$$x = S_x i$$

$$y = S_y j$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ 1 \end{bmatrix}$$



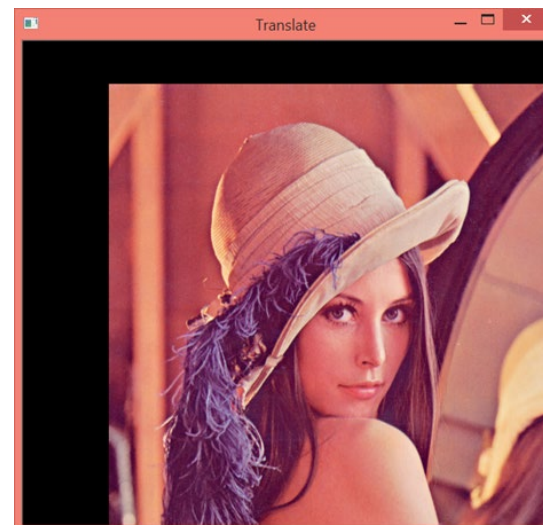
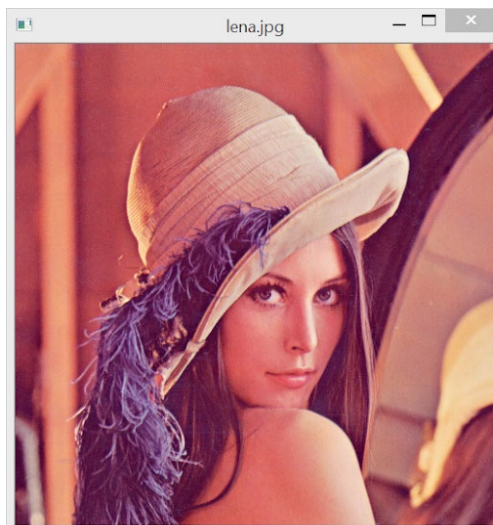
# 1.1. Transformaciones elementales

- Traslación:

$$x = i + i_d$$

$$y = j + j_d$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & i_d \\ 0 & 1 & j_d \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ 1 \end{bmatrix}$$



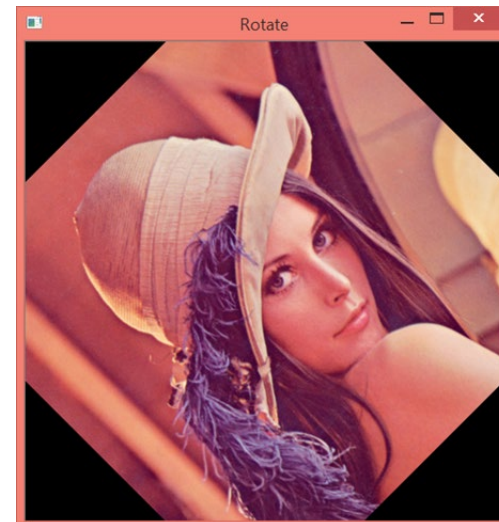
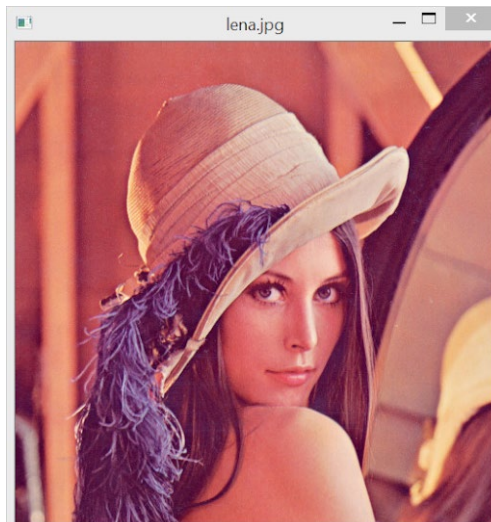
# 1.1. Transformaciones elementales

- Rotación:

$$x = \cos \theta i - \sin \theta j$$

$$y = \sin \theta i + \cos \theta j$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ 1 \end{bmatrix}$$



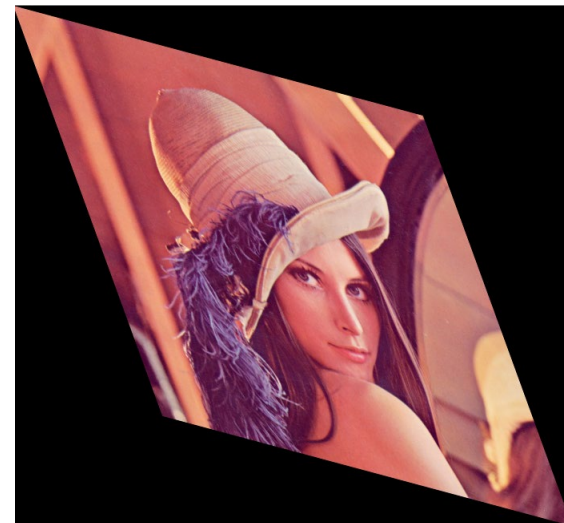
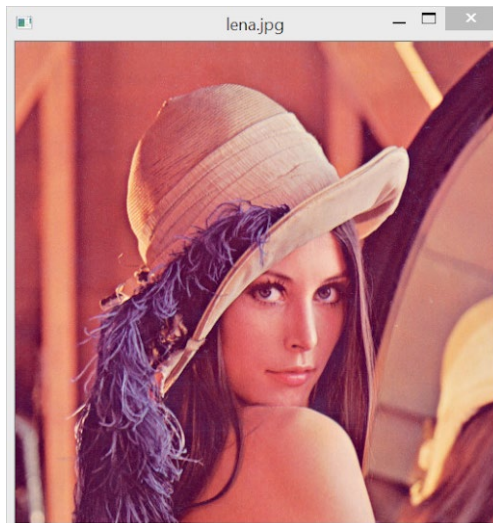
# 1.1. Transformaciones elementales

- Inclinación:

$$x = i + \theta * j$$

$$y = i * \theta + j$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & \theta & 0 \\ \theta & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ 1 \end{bmatrix}$$



# Índice de contenidos

---

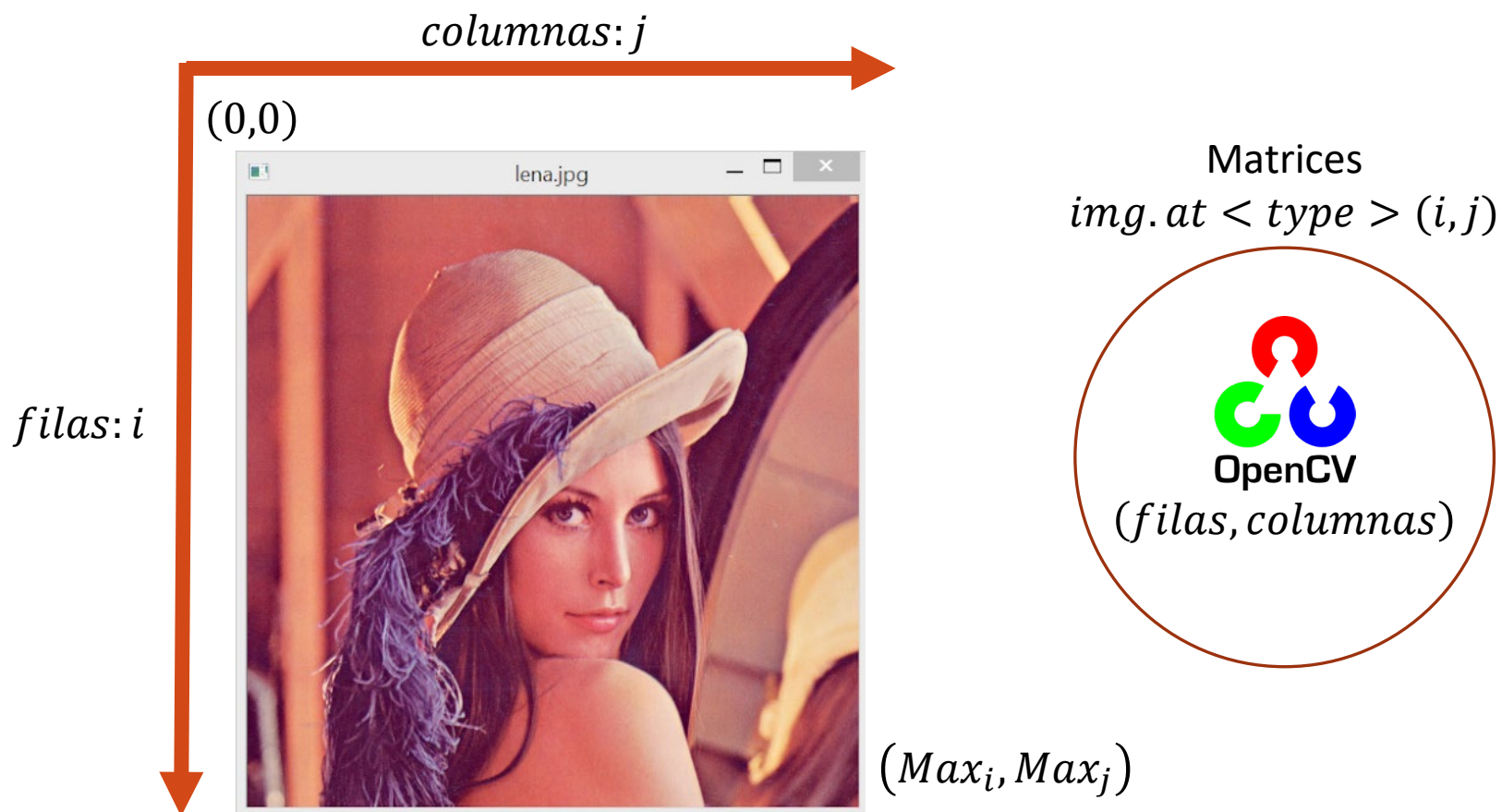
## 1. Transformaciones geométricas:

- Transformadas elementales
- Sistema de coordenadas
- Interpolación
- Registro
- Ejemplos

## 2. Transformaciones radiométricas:

- Suavizado
- Histograma
- Filtrado homomórfico
- Correspondencia de histogramas
- Comparación de histogramas
- Ejemplos

## 1.2. Sistema de coordenadas

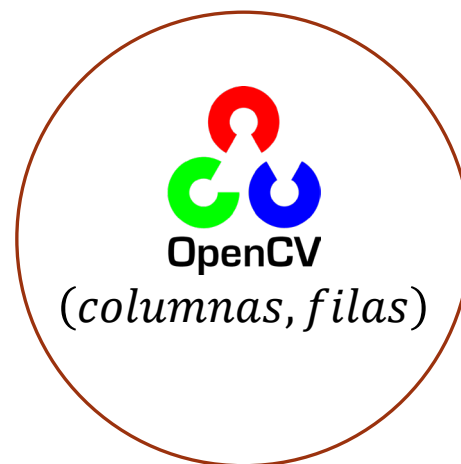




## 1.2. Sistema de coordenadas



Imágenes  
 $img.at < type > (cv :: Point(i, j))$



# Índice de contenidos

---

## 1. Transformaciones geométricas:

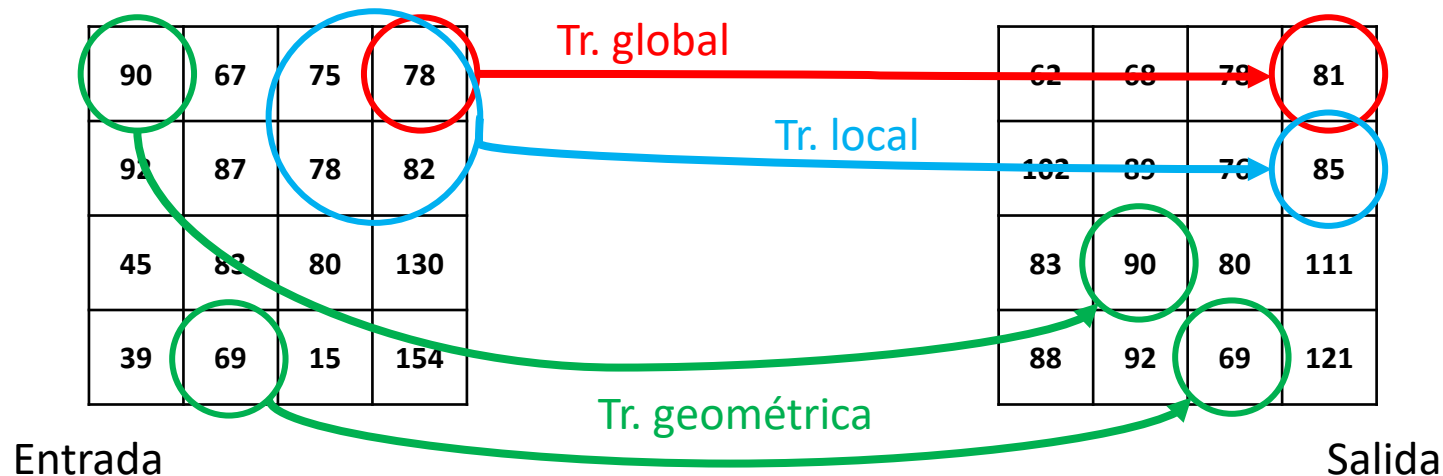
- Transformadas elementales
- Sistema de coordenadas
- Interpolación
- Registro
- Ejemplos

## 2. Transformaciones radiométricas:

- Suavizado
- Histograma
- Filtrado homomórfico
- Correspondencia de histogramas
- Comparación de histogramas
- Ejemplos






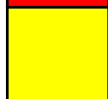


# 1.3. Interpolación



$$Out(i, j) = In(f_1(i, j), f_2(i, j))$$

- El valor de un píxel depende de otro píxel (o varios) cuya posición es calculada a través de un par de funciones  $f_1$  y  $f_2$
- El tamaño de la imagen de salida puede ser distinto del tamaño de la imagen de entrada

# 1.3. Interpolación

	0	1	2	
0				$In$
1				

- Reflejos y rotaciones exactas (sentido horario):
- Siendo  $In$  la imagen de entrada, y su tamaño  $(0 \dots M_i, 0 \dots M_j)$



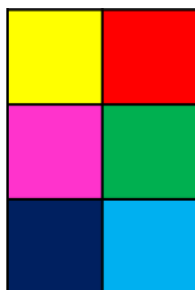
Espejo horizontal

$$Out(i, j) = In(M_i - i, j)$$



Espejo vertical

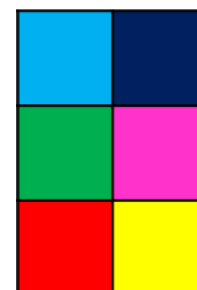
$$Out(i, j) = In(i, M_j - j)$$



Rotación 90°



Rotación 180°



Rotación 270°

$$Out(i, j) = In(j, M_j - i)$$

$$Out(i, j) = In(M_i - i, M_j - j)$$

$$Out(i, j) = In(M_i - j, i)$$

# 1.3. Interpolación

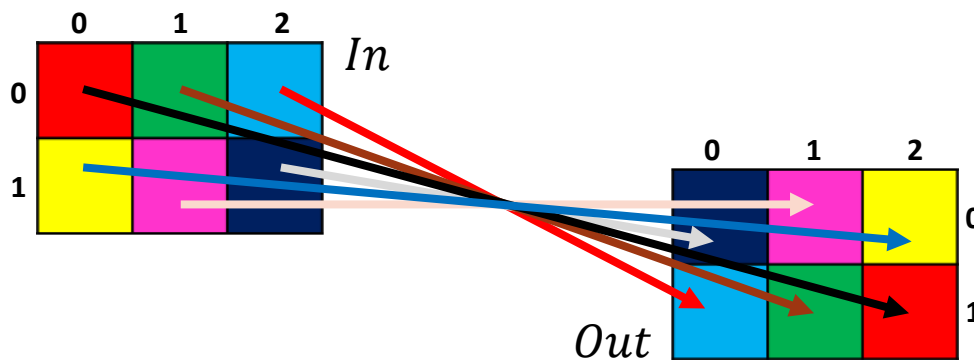
- En general la transformación tendrá la forma:

$$Out(i, j) = In(f_1(i, j), f_2(i, j))$$

- Siendo  $f_1$  y  $f_2$  dos funciones cualesquiera del tipo:

$$f_1, f_2: N \times N \rightarrow R$$

- $f_1$  : posición en  $i$  del resultado para el píxel original  $(i, j)$
- $f_2$  : posición en  $j$  del resultado para el píxel original  $(i, j)$



Rotación 180°

$$f_1(i, j) = M_i - i$$

$$f_2(i, j) = M_j - j$$

# 1.3. Interpolación: aumento

- ¿Qué ocurre cuando el resultado no es un número entero?
- Por ejemplo, la siguiente transformación genera un aumento x2:

$$Out(i, j) = In(i/2, j/2)$$

	0	1	2	
0				<i>In</i>
1				

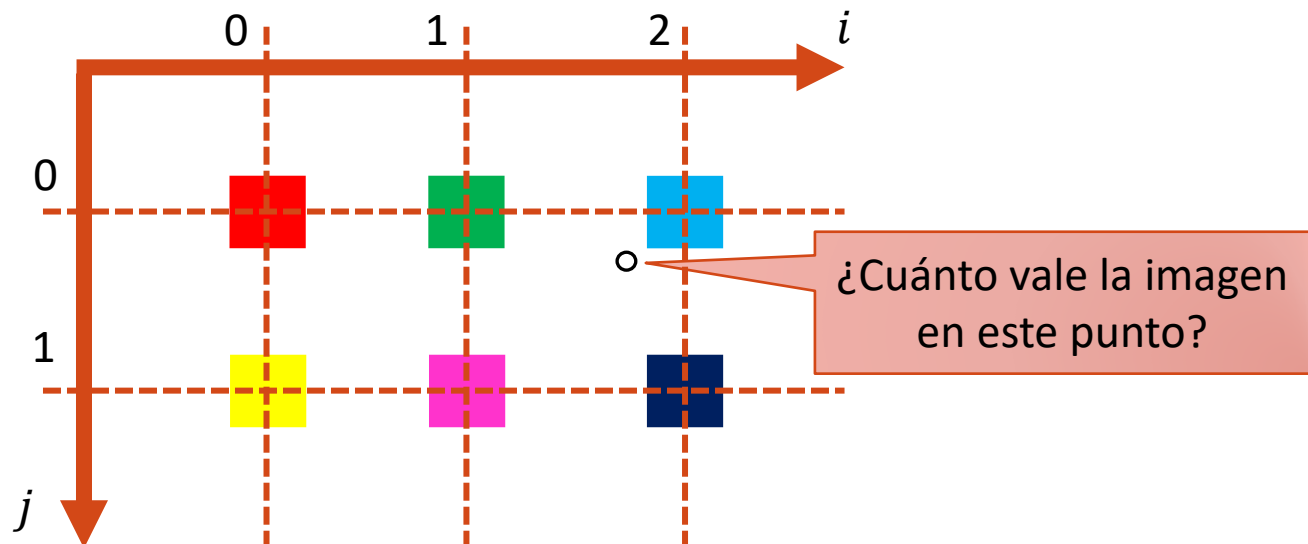
	0	1	2	3	4	5	
0							<i>Out</i>
1							
2							
3							

- $Out(0,0) = In(0,0)$
- $Out(1,0) = In(0.5,0)$
- $Out(1,1) = In(0.5,0.5)$

¡Los índices decimales no están definidos en la matriz!

# 1.3. Interpolación: aumento

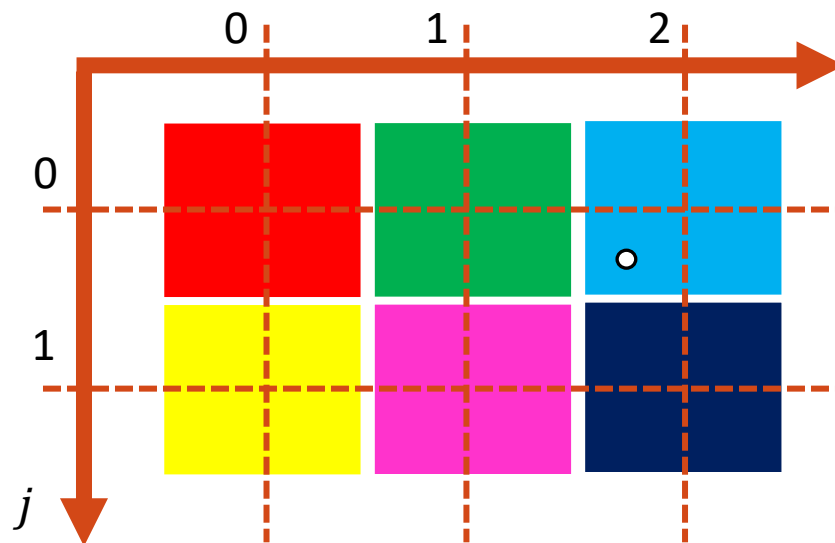
- Problema: las imágenes son señales discretas, pero la transformación geométrica las trata como si fueran continuas (definidas en todo el plano)



- Solución: aplicar una interpolación
- Tipos de interpolación: vecino más próximo, bilineal, bicúbica, supermuestreo

# 1.3. Interpolación: vecinos

- Vecino más próximo:
  - Cualquier punto del espacio toma el valor del píxel más cercano



- Implementación sencilla, redondeo:

$$\left. \begin{array}{l} f_1(i,j) \rightarrow \text{round}(f_1(i,j)) \\ f_2(i,j) \rightarrow \text{round}(f_2(i,j)) \end{array} \right\} \text{Out}(i,j) = \text{In}(\text{round}(f_1(i,j)), \text{round}(f_2(i,j)))$$

# 1.3. Interpolación: vecinos

$$Out(i,j) = In(round(i/2), round(j/2))$$

	0	1	2	
0				<i>In</i>
1				

	0	1	2	3	4	5	
0							<i>Out</i>
1							
2							
3							

- $Out(0,0) = In(0,0)$
- $Out(1,0) = In(0,0)$
- $Out(3,1) = In(1,0)$
- $Out(5,1) = In(2,0)$

# 1.3. Interpolación: vecinos

- Esta interpolación puede hacerse de múltiples formas, por ejemplo, calculando la distancia euclídea con respecto a cada vecino, lo que da como resultado valores intermedios



Imagen original  
25 x 26 px

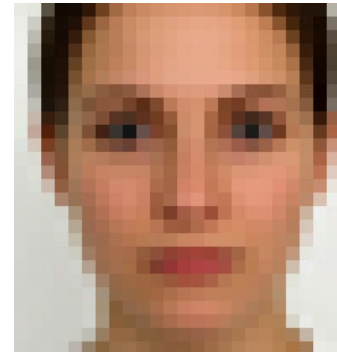


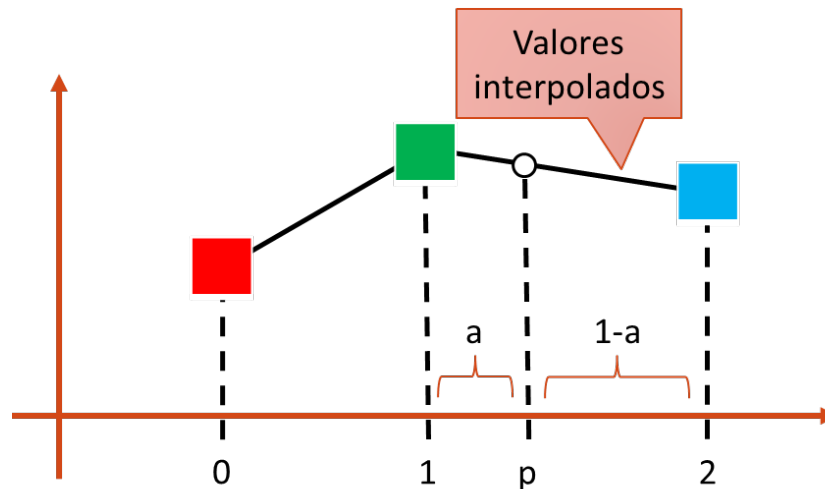
Imagen interpolada (x10)  
250 x 260 px

- Ventajas:
  - Es muy sencilla y rápida de calcular
- Inconvenientes:
  - El efecto de cuadriculado es evidente, y da lugar imágenes de poca calidad



# 1.3. Interpolación: bilineal

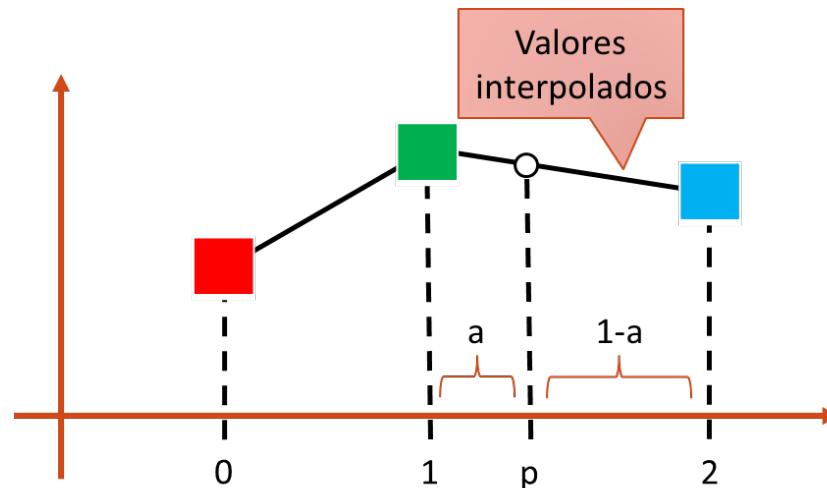
- Cálculo de la interpolación lineal



- En una dimensión, una interpolación lineal significa trazar una línea recta entre cada par de puntos consecutivos

# 1.3. Interpolación: bilineal

- Cálculo de la interpolación lineal

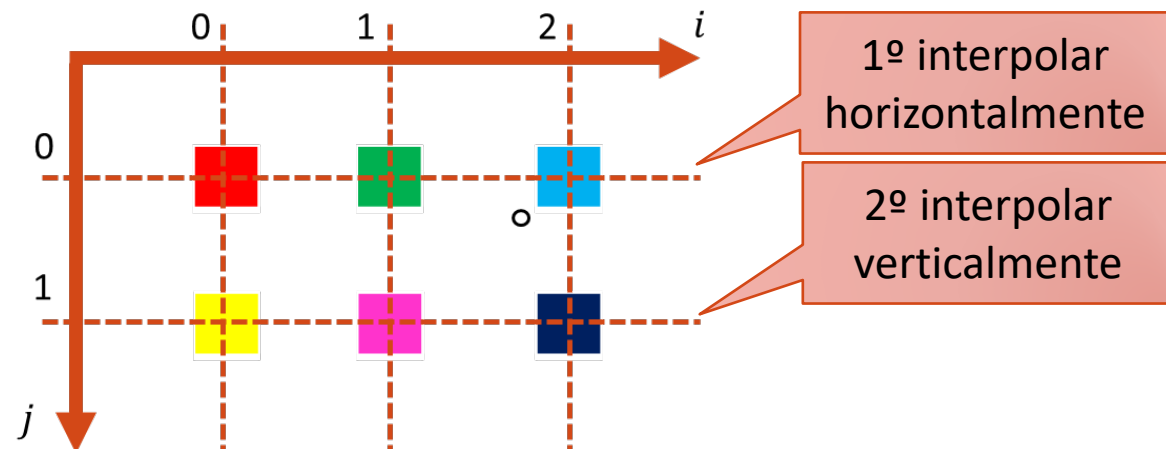


- Sea  $p$  el punto que queremos interpolar:
  - Supongamos que  $p$  se encuentra entre  $i$  y  $d$ , es decir:  $i = \lfloor p \rfloor, d = i + 1$
  - El valor interpolado en  $p$  será:  $O(p) = (1 - a) * I(i) + a * I(d)$  siendo  $a = p - i$

Media ponderada, de los valores de  $i$  y  $d$  según la distancia  $a$

# 1.3. Interpolación: bilineal

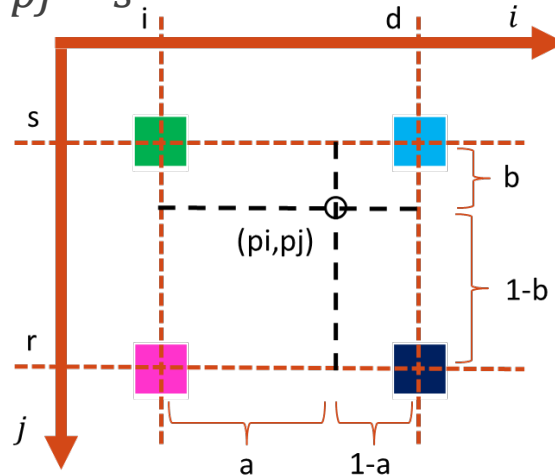
- En dos dimensiones, la interpolación bilineal consiste en aplicar dos interpolaciones lineales:



- Interpolar la función horizontalmente, en las filas existentes
- Interpolar la función verticalmente en todo el espacio

# 1.3. Interpolación: bilineal

- Sea  $p = (p_i, p_j)$ , con  $i = p_i$ ,  $d = i + 1$ ,  $s = p_j$ ,  $r = s + 1$   
con  $a = p_i - i$ ,  $b = p_j - s$



- Cálculo de la interpolación bilineal:
  - $O(p_i, s) = (1 - a) * I(i, s) + a * I(d, s)$
  - $O(p_i, r) = (1 - a) * I(i, r) + a * I(d, r)$
  - $O(p_i, p_j) = (1 - b) * O(p_i, s) + b * O(p_i, r)$
  - $O(p_i, p_j) = (1 - a) * (1 - b) * I(i, s) + a * (1 - b) * I(d, s) + (1 - a) * b * I(i, r) + a * b * I(d, r)$

Media ponderada de los 4  
píxeles circundantes

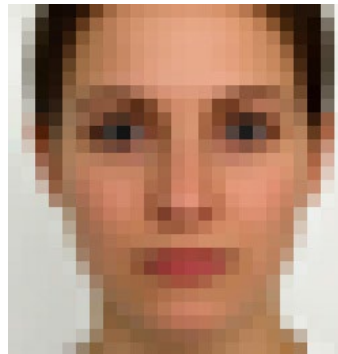
# 1.3. Interpolación: bilineal

- Ejemplo: Zoom de 10x con interpolación bilineal

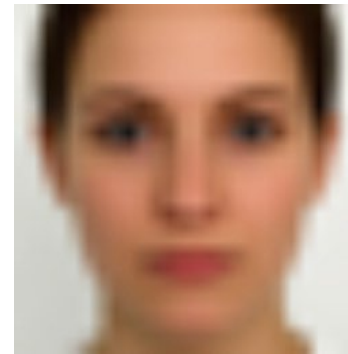
$$O(i, j) = I(i/10, j/10)$$



Imagen original  
25 x 26 px



Vecino más próximo (x10)  
250 x 260 px

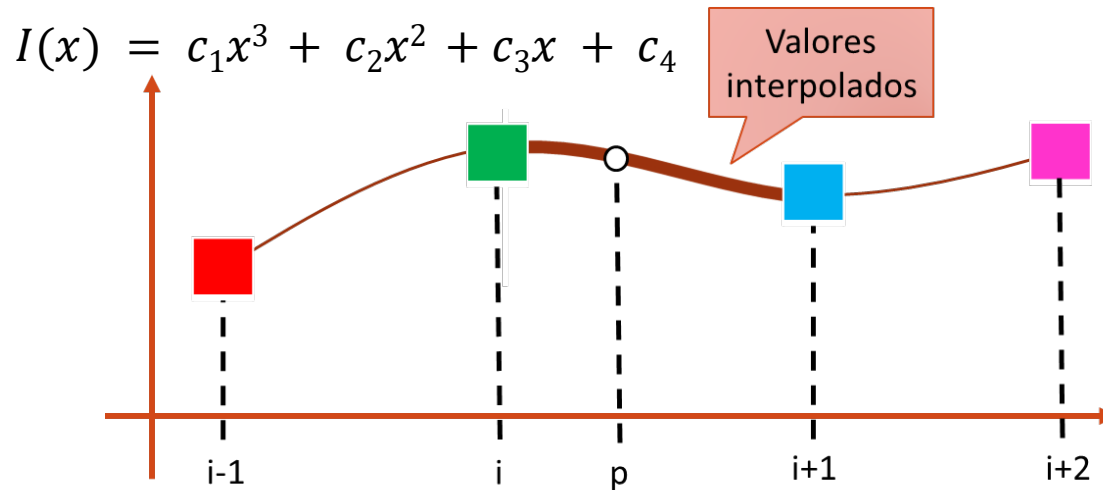


Interpolación bilineal (x10)  
250 x 260 px

- Nota: un zoom entero de  $K$  con interpolación bilineal es parecido (= a veces) a un zoom de  $K$  con vecino más próximo, seguido de un filtro de media de  $K \times K$

# 1.3. Interpolación: bicúbica

- La interpolación bilineal mejora la de vecino más próximo, pero produce un efecto de “zonas rectangulares”
- Interpolación bicúbica: basada en dos interpolaciones cúbicas



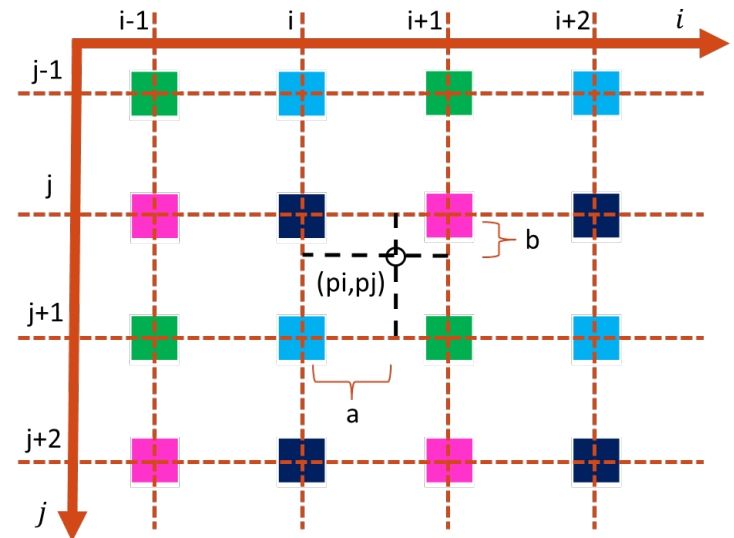
- En una dimensión, la interpolación cúbica consiste en trazar una cúbica entre los 4 puntos más próximos (2 a la izquierda y 2 a la derecha)

# 1.3. Interpolación: bicúbica

- Cálculo de la interpolación cúbica:
  - Sea  $p$  el punto que queremos interpolar:  $i = \lfloor p \rfloor$
  - Obtener las 4 ecuaciones:  
 $O(i - 1) = I(i - 1); O(i) = I(i); O(i + 1) = I(i + 1); O(i + 2) = I(i + 2)$
  - 4 ecuaciones, 4 incógnitas  $\rightarrow$  despejar y obtener  $c1, c2, c3, c4$
  - Aplicar las constantes, obteniendo  $O(p)$
- Interpolación bicúbica. Igual que la bilineal, se basa en dos interpolaciones cúbicas:
  1. Interpolación cúbica horizontal, en las filas existentes (usando 4 puntos)
  2. Interpolación cúbica vertical en todo el espacio usando 4 puntos (usando la anterior interpolación)
- En la interpolación bicúbica de un punto  $(p_i, p_j)$  intervienen los 16 puntos circundantes

# 1.3. Interpolación: bicúbica

- Cálculo de la interpolación cúbica:
  - Igual que con la bilineal, el valor del punto se puede calcular como una media ponderada de los 4x4 píxeles circundantes



$$Out(pi, pj) = \sum_{n=-1..2} \sum_{m=-1..2} In(i + n, j + m) * P(n - a) * P(b - m)$$

- Siendo:

$$P(k) = 1/6 (C(k + 2)^3 - 4 * C(k + 1)^3 + 6 * C(k)^3 - 4 * C(k - 1)^3)$$

$$C(k) = \max(0, k)$$

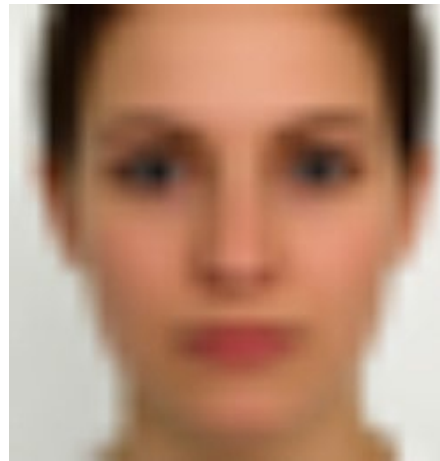


# 1.3. Interpolación: bicúbica

- Ejemplo: Zoom de 10x con interpolación bicúbica



Imagen original  
25 x 26 px



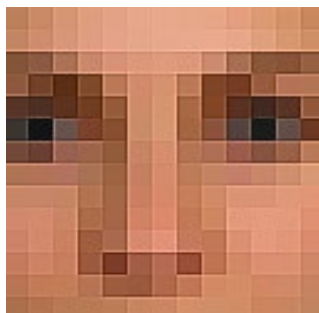
Interpolación bilineal (x10)  
250 x 260 px



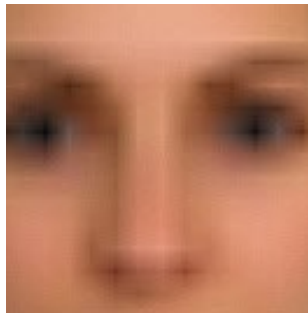
Interpolación bicúbica (x10)  
250 x 260 px

# 1.3. Interpolación: comparativa

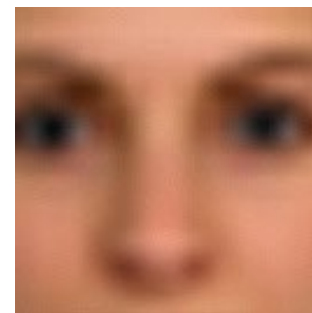
- Detalle del zoom de 10x, con vecino más próximo, interpolación bilineal y bicúbica. Se ha aplicado un perfilado en las 3, para destacar el efecto del zoom



Vecino más próximo



Interpolación bilineal



Interpolación bicúbica

- En todos los casos se nota la falta de detalle (obviamente), pero en la bilineal son más evidentes los artificios horizontales y verticales que en la bicúbica

# 1.3. Interpolación: reducción

- En las operaciones de reducción también se aplican interpolaciones, pero...
- Reducción de  $k$ :  $O(i, j) = In(k * i, k * j)$ , con Out:  $\max(i) / k * \max(j) / k$
- Ejemplo: Reducción de 5x



Imagen original  
500x386



Reducción 5x con  
vecino más próximo

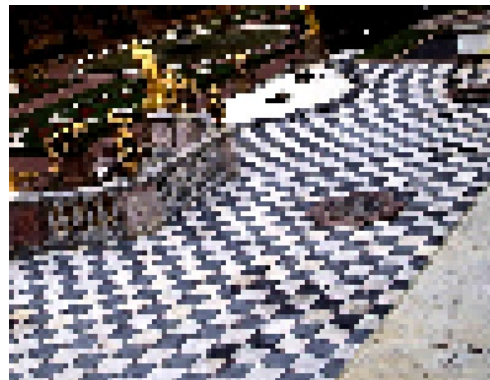
Estructuras extrañas  
Aliasing

# 1.3. Interpolación: reducción

- El problema no se soluciona con interpolación bilineal o bicúbica



Imagen original  
500x386



Reducción 5x con  
interpolación bilineal

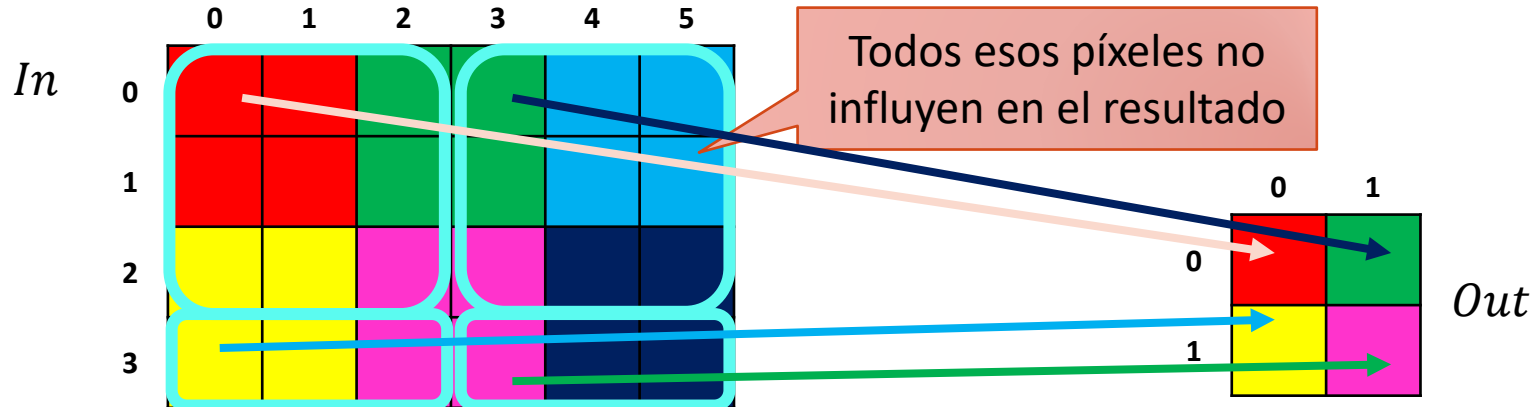


Reducción 5x con  
interpolación bicúbica

- El problema se debe a que los detalles son más pequeños que la resolución de salida. Pero, además, los métodos de interpolación no mejoran la situación: cada píxel de salida es un muestreo ordenado de uno de entrada

# 1.3. Interpolación: supermuestreo

- Ejemplo: Reducción de 3x.  $Out(i, j) = In(3i, 3j)$



- **Solución:** cada píxel de salida debería ser la media de los 3x3 píxeles de entrada correspondientes
- Interpolación por **supermuestreo** (super sampling):
  - Considerar el píxel como un “volumen” con cierto área
  - Aplicar varias veces la transformación y tomar la media

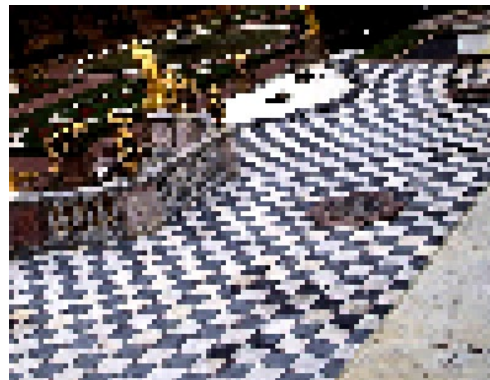


# 1.3. Interpolación: supermuestreo

- Ejemplo: Reducción de 5x, con supermuestreo



Imagen original  
500x386



Reducción 5x con  
interpolación bilineal



Reducción 5x con  
supermuestreo

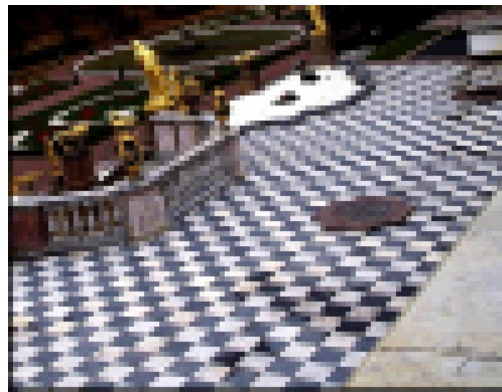
- Resultado: el supermuestreo logra un resultado de mucha más calidad. Evita el problema del aliasing
- Sin embargo, el supermuestreo es mucho más costoso, requiere más cálculos
- Cuanto mayor reducción, mayor es el efecto del aliasing

# 1.3. Interpolación: supermuestreo

- Una alternativa al supermuestreo es aplicar primero un filtro de suavizado (por ejemplo, de media) y después un simple vecino más próximo



Imagen original  
500x386



Reducción 5x con  
vecino más próximo,  
de la suavizada



Reducción 5x con  
supermuestreo, de  
la original

- Pero esto sólo es aplicable en las transformaciones que impliquen una reducción de resolución

## 1.3. Interpolación: conclusiones

---

- **Transformación geométrica:** cada píxel de salida depende de uno de entrada cuya posición es calculada de acuerdo a un par de funciones
- Como las posiciones pueden ser no enteras, es necesario aplicar interpolaciones: vecino más próximo, bilineal, bicúbica
- Qué utilizar:
  - En **zoom**: interpolación bicúbica
  - En **reducción**: supermuestreo
- Pero las técnicas que **mejores resultados** obtienen son **más costosas** que las demás aproximaciones
- Las interpolaciones bilineal y bicúbica (y otras más avanzadas) dan la sensación de mejorar la resolución de la imagen, pero ¡cuidado! cualquier detalle aparente de resolución inferior a un píxel es una mera alucinación



# Índice de contenidos

---

## 1. Transformaciones geométricas:

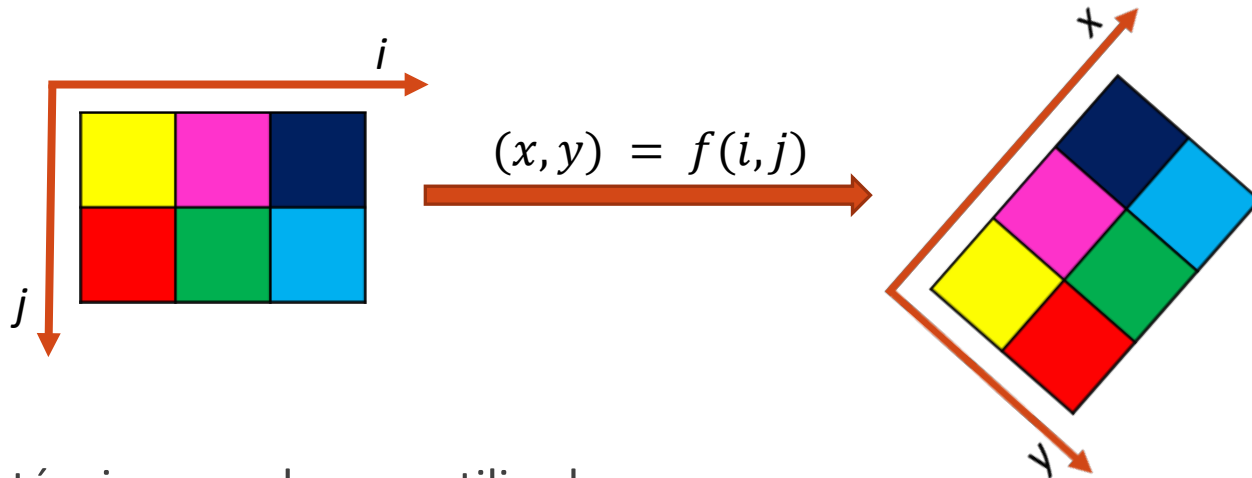
- Transformadas elementales
- Sistema de coordenadas
- Interpolación
- Registro
- Ejemplos

## 2. Transformaciones radiométricas:

- Suavizado
- Histograma
- Filtrado homomórfico
- Correspondencia de histogramas
- Comparación de histogramas
- Ejemplos

# 1.4. Registro

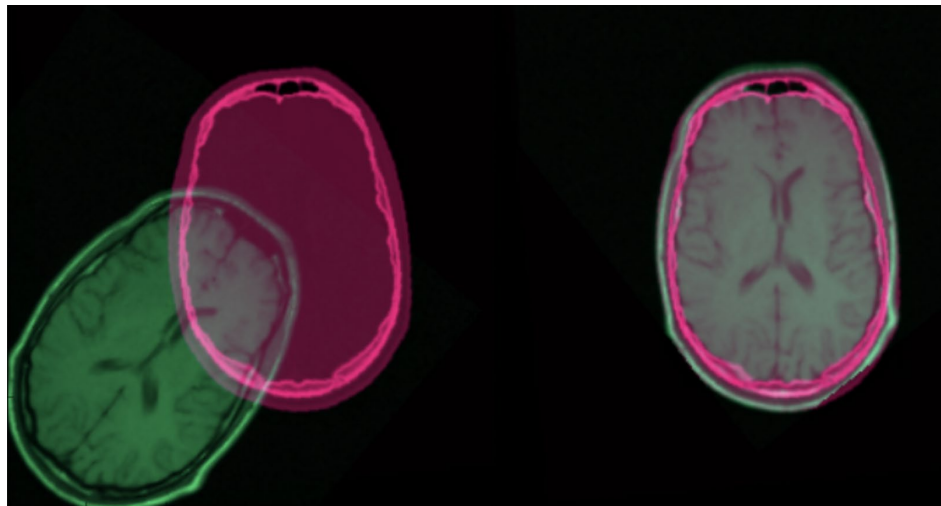
- Consiste en encontrar los parámetros de la transformada (afín) para “descubrir” qué transformaciones ha sufrido una imagen con diferentes fines



- Estas técnicas pueden ser utilizadas:
  - Comparar dos imágenes
  - Ver si una imagen está contenida en otra
  - Integrar datos obtenidos de diferentes sensores (imágenes médicas)
  - ...

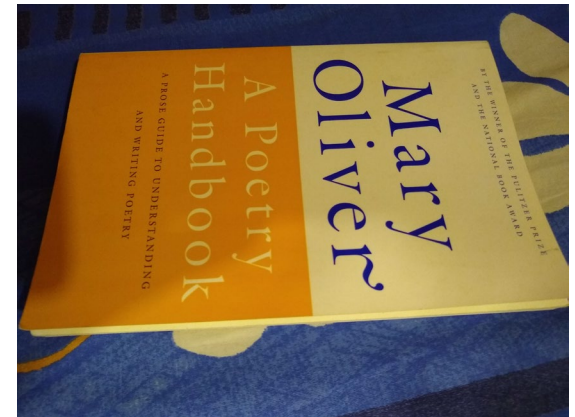
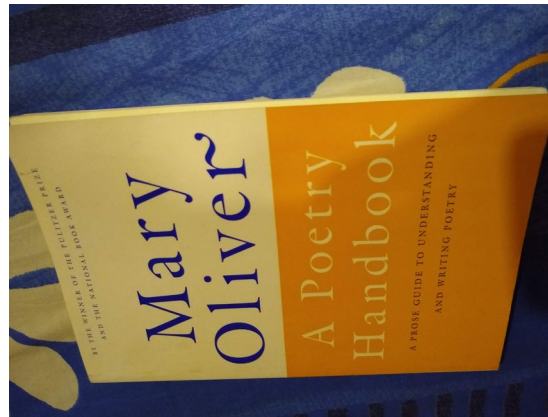
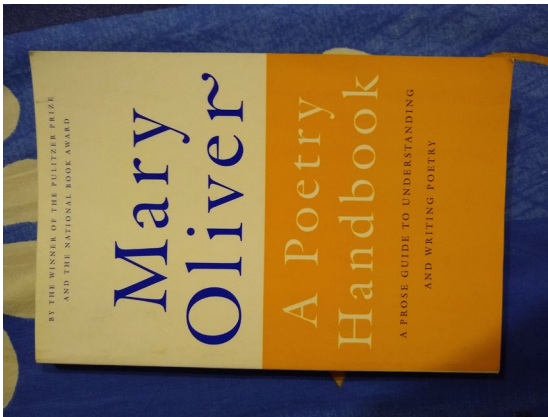
# 1.4. Registro

- El registro puede llevarse a cabo de múltiples maneras, las principales son:
  - Basadas en transformaciones lineales
  - Basadas en características
  - Basadas en cambio de dominio o frecuencia



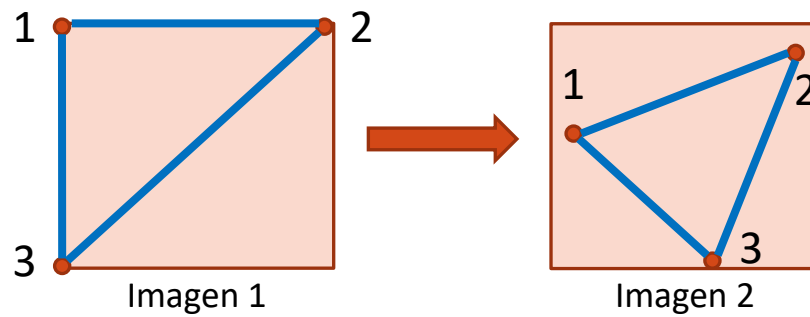
# 1.4. Registro

- Si se quiere alinear una imagen con el mismo ángulo que otra imagen de referencia, es necesario hacer una transformación de coordenadas
- Supongamos que tenemos la primera imagen como imagen de referencia, el algoritmo de registro nos ayuda a alinear la segunda y tercera imagen en el mismo plano que la primera



# 1.4. Registro

- Transformación afín:
  - Las transformaciones afines mapean puntos en nuevos puntos aplicando una combinación lineal de operaciones de traslación, rotación y escalado



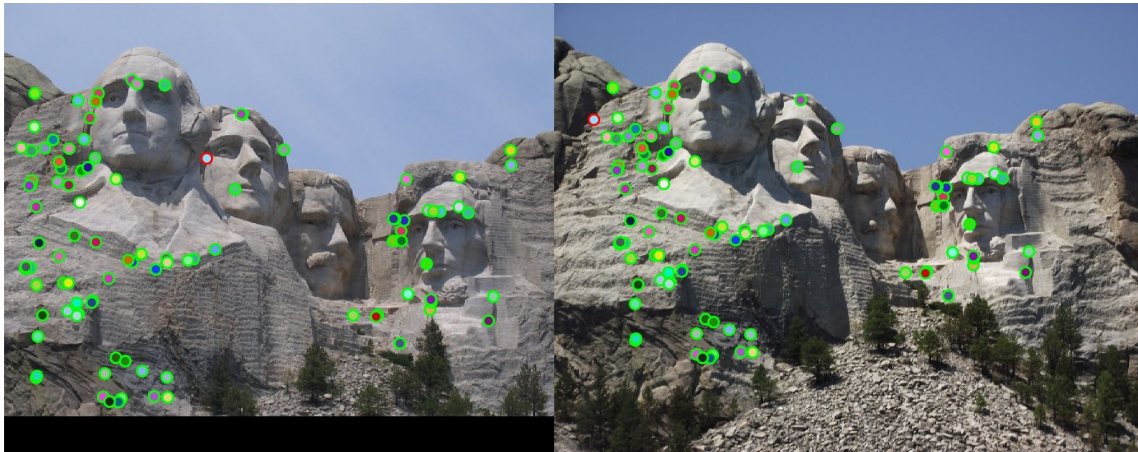
- Los puntos 1, 2 y 3 (formando un triángulo en la imagen 1) son mapeados en la imagen 2 formando un triángulo, aunque notablemente modificado
- Si se encuentra la transformada afín con esos tres puntos (se pueden elegir más si se quiere), se puede aplicar dicha relación a todos los píxeles de la imagen

# 1.4. Registro

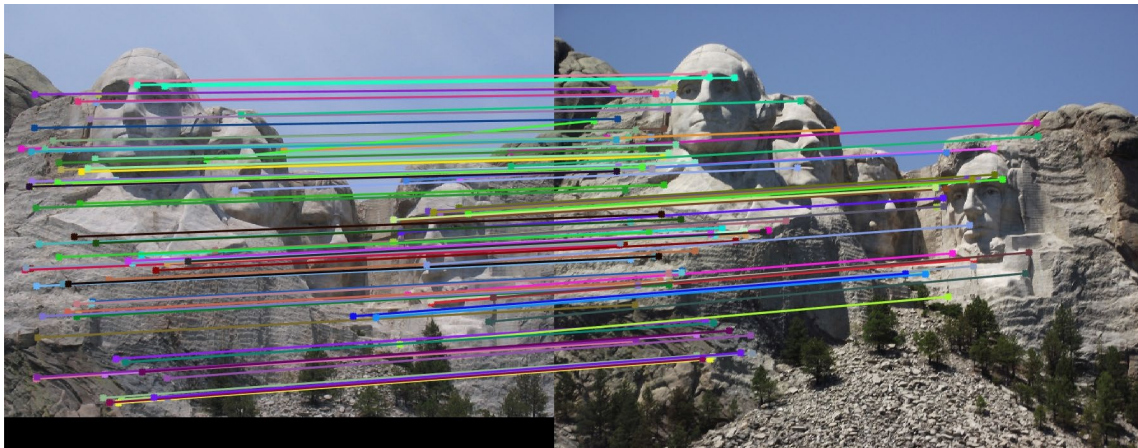
---

- El algoritmo sería el siguiente:
  - Convertir ambas imágenes a escala de grises
  - Hacer coincidir las características de la imagen a alinear con la imagen de referencia y almacenar las coordenadas de los puntos clave correspondientes
  - Los puntos clave son simplemente los pocos puntos seleccionados que se utilizan para calcular la transformación (puntos destacados), y los descriptores son histogramas de los gradientes de la imagen para caracterizar la apariencia de un punto clave. La extracción de estos puntos lo veremos más adelante
  - Hacer coincidir los puntos clave entre las dos imágenes
  - Elegir las mejores coincidencias y eliminar las ruidosas
  - Encontrar la transformada homomórfica
  - Aplicar esta transformación a la imagen no alineada original para obtener la imagen de salida

# 1.4. Registro



Identificación de los  
puntos más relevantes



Correspondencia  
entre los puntos



# Índice de contenidos

---

## 1. Transformaciones geométricas:

- Transformadas elementales
- Sistema de coordenadas
- Interpolación
- Registro
- Ejemplos

## 2. Transformaciones radiométricas:

- Suavizado
- Histograma
- Filtrado homomórfico
- Correspondencia de histogramas
- Comparación de histogramas
- Ejemplos



# 1.5. Ejemplos: escalado

```
int main( int argc, char** argv ) {
    // Load an image
    // ...
    // Create windows
    namedWindow( "Original image", WINDOW_AUTOSIZE );
    namedWindow( "Resize x10", WINDOW_AUTOSIZE );
    namedWindow( "Resize /5", WINDOW_AUTOSIZE );

    imshow( "Original image", src );

    // Resize
    Mat resize_dst;
    int size_col = 10, size_row = 10;
    resize(src, resize_dst, cv::Size(), size_col, size_row, INTER_NEAREST);
    imshow( "Resize x10", resize_dst );

    // Resampling
    Mat resize_dst2;
    int res = 5;
    resize(resize_dst, resize_dst2, Size(resize_dst.cols/res, resize_dst.rows/res), INTER_CUBIC);
    imshow( "Resize /5", resize_dst2 );

    waitKey();
    return 0;
}
```

**INTER\_NEAREST.** Es una interpolación del vecino cercano  
**INTER\_LINEAR.** Interpolación bilineal  
**INTER\_CUBIC.** Interpolación bicúbica (área de 4×4 píxeles)

Aumento de una imagen utilizando la interpolación por vecinos cercanos

Reducción de una imagen utilizando la interpolación bicúbica

# 1.5. Ejemplos: rotación

```
int main( int argc, char** argv ) {
    // Load an image
    // ...

    // Create windows
    namedWindow( "Original image", WINDOW_AUTOSIZE );
    namedWindow( "Rotation", WINDOW_AUTOSIZE );

    imshow( "Original image", src );

    // Rotation
    Mat rotation_dst;
    Point center = Point( src.cols/2, src.rows/2 );
    double angle = -50.0;
    double scale = 0.6;
    Mat rot_mat = getRotationMatrix2D( center, angle, scale );
    warpAffine(src, rotation_dst, rot_mat, src.size());
    imshow( "Rotation", rotation_dst );

    waitKey();
    return 0;
}
```

Para rotar se necesitan:

1. Punto en el que se va a rotar la imagen
2. Ángulo de rotación: un valor positivo rota en dirección contraria a las agujas del reloj

Creación de la matriz de rotación

A través de la función **warpAffine** se aplica la matriz de rotación a toda la imagen

# 1.5. Ejemplos: traslación

```
int main( int argc, char** argv ) {
    // Load an image
    // ...

    // Create windows
    namedWindow( "Original image", WINDOW_AUTOSIZE );
    namedWindow( "Translation", WINDOW_AUTOSIZE );

    imshow( "Original image", src );

    // Translation
    Mat translation_dst;
    float displacement_x = 100, displacement_y = 100;
    float data[6] = {1, 0, displacement_x, 0, 1, displacement_y};
    Mat trans_mat(2, 3, CV_32F, data);
    warpAffine(src, translation_dst, trans_mat, src.size());
    imshow( "Translation", translation_dst );

    waitKey();
    return 0;
}
```

Creación de la matriz de traslación

A través de la función **warpAffine** se aplica la matriz de traslación a toda la imagen

# 1.5. Ejemplos: inclinación + rotación

```
int main( int argc, char** argv ) {
    // Load an image
    // ...
    // Points to the affine transform calculation
    Point2f srcTri[3];
    srcTri[0] = Point2f( 0.f, 0.f );
    srcTri[1] = Point2f( src.cols - 1.f, 0.f );
    srcTri[2] = Point2f( 0.f, src.rows - 1.f );

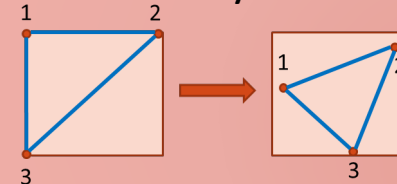
    Point2f dstTri[3];
    dstTri[0] = Point2f( 0.f, src.rows*0.33f );
    dstTri[1] = Point2f( src.cols*0.85f, src.rows*0.25f );
    dstTri[2] = Point2f( src.cols*0.15f, src.rows*0.7f );

    // Calculate the affine transform
    Mat warp_mat = getAffineTransform( srcTri, dstTri );
    Mat warp_dst = Mat::zeros( src.rows, src.cols, src.type() );

    // Apply the affine transform
    warpAffine( src, warp_dst, warp_mat, warp_dst.size() );

    // ...
}
```

Se necesitan tres puntos en la imagen original y destino para calcular la relación entre ellos y su transformada



A través de la función **getAffineTransform** se calcula la transformada afín

A través de la función **warpAffine** se aplica la matriz con la transformada a toda la imagen

## 1.5. Ejemplos: inclinación + rotación

```
int main( int argc, char** argv ) {
    // ...
    // Rotation point, angle and scale (optional)
    Point center = Point( warp_dst.cols/2, warp_dst.rows/2 );
    double angle = -50.0;
    double scale = 0.6;

    // Calculate and apply rotation matrix
    Mat rot_mat = getRotationMatrix2D( center, angle, scale );

    Mat warp_rotate_dst;
    warpAffine( warp_dst, warp_rotate_dst, rot_mat, warp_dst.size() );

    // Show images
    imshow( "Source image", src );
    imshow( "Warp", warp_dst );
    imshow( "Warp + Rotate", warp_rotate_dst );

    waitKey();
    return 0;
}
```

Punto a partir del cual se va a rotar

Ángulo de rotación (positivo es sentido antihorario) y factor de escala (opcional)

Se genera la matriz de rotación

A través de la función **warpAffine** se aplica la matriz de rotación a toda la imagen

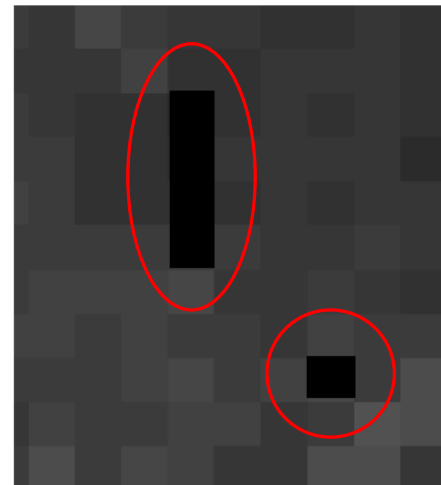
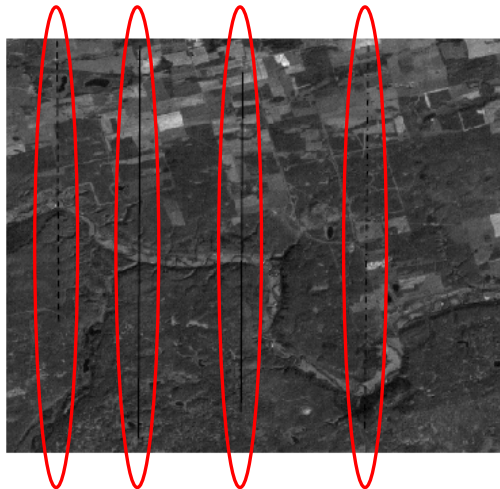
# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

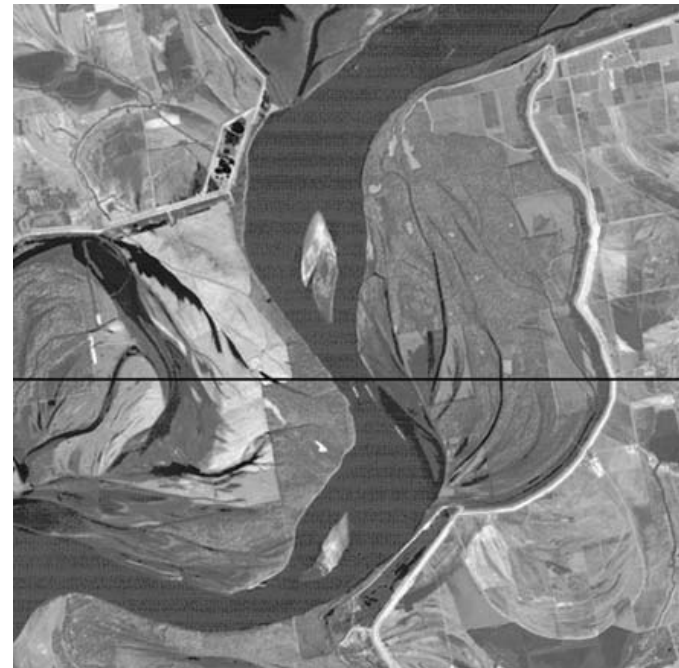
## 2. Transformación radiométrica

- Las correcciones radiométricas son operaciones que tienden a eliminar cualquier anomalía en el valor radiométrico ND de la imagen, ocurridas durante la adquisición de ésta
- Estas anomalías son debidas al mal funcionamiento del sensor: líneas perdidas, píxeles aislados, bandeo por efecto de la atmósfera (dispersión)



## 2. Transformación radiométrica

- Un ejemplo sencillo, es la restauración de líneas o píxeles perdidos:
  - Procedimientos estadísticos simples
  - Sustitución de píxel o línea
  - Promedio de valores
  - Con banda auxiliar





# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

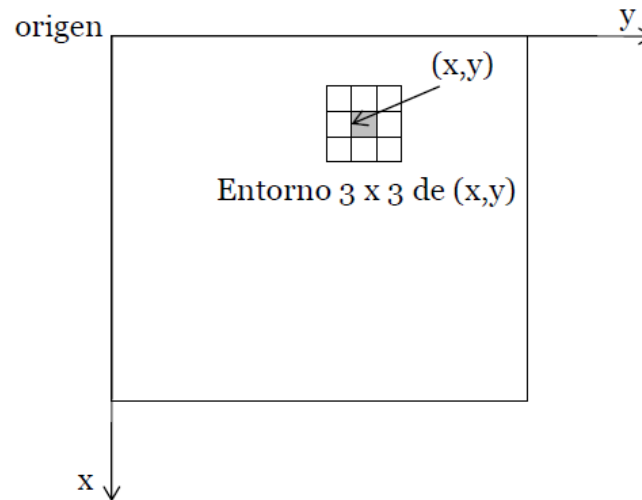
## 2.1. Suavizado

- El suavizado de una imagen se utiliza con diferentes fines:
  - Reducir las variaciones de intensidad entre píxeles vecinos
  - Eliminar pequeños detalles antes de la segmentación de un objeto de interés
  - Rellenar pequeños espacios
  - Eliminar ruido: modificar aquellos píxeles cuyo nivel de intensidad es muy diferente al de sus vecinos



## 2.1. Suavizado: vecinos

- Para cada píxel que se desea procesar, es necesario coger una ventana de vecindad, o núcleo



- Sobre esta ventana de vecindad, se aplicará una operación, cuyo resultado asignará un valor al píxel de estudio  $(x, y)$

## 2.1. Suavizado: vecinos general

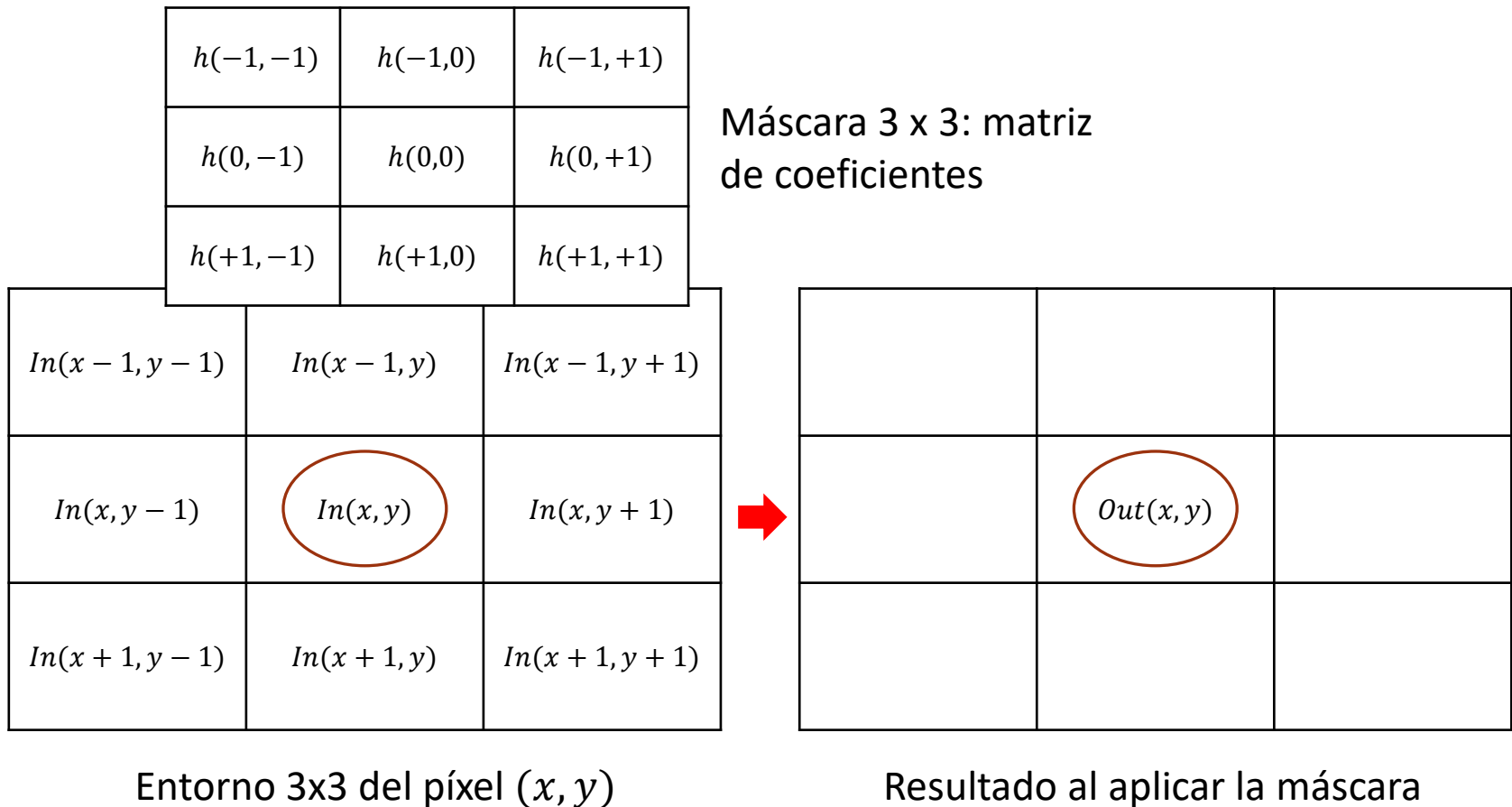
---

- Las funciones de procesamiento de la imagen en el dominio espacial pueden expresarse como:

$$Out(x, y) = h * In(x, y)$$

- Siendo:
  - *Out* la imagen de salida, la procesada
  - *In* la imagen de entrada, la original
  - *h* un operador que actúa sobre *In* y se define en algún entorno de  $(x, y)$ , también conocido como núcleo

## 2.1. Suavizado: vecinos general



## 2.1. Suavizado: vecinos lineal

Máscara 3 x 3

$h(-1, -1)$	$h(-1, 0)$	$h(-1, +1)$
$h(0, -1)$	$h(0, 0)$	$h(0, +1)$
$h(+1, -1)$	$h(+1, 0)$	$h(+1, +1)$

$$Out(x, y) = \sum_{s=-1}^1 \sum_{t=-1}^1 h(s, t) * In(x + s, y + t)$$

$In(x - 1, y - 1)$	$In(x - 1, y)$	$In(x - 1, y + 1)$
$In(x, y - 1)$	$In(x, y)$	$In(x, y + 1)$
$In(x + 1, y - 1)$	$In(x + 1, y)$	$In(x + 1, y + 1)$



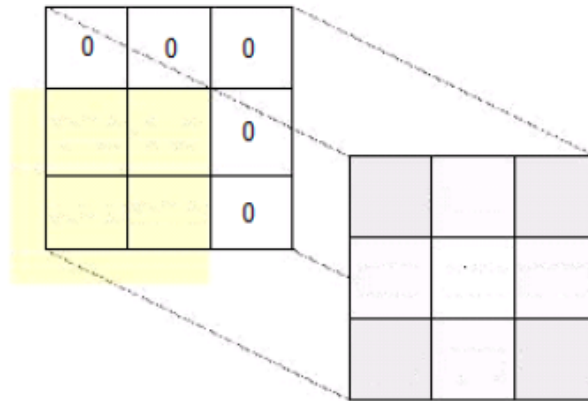
	$Out(x, y)$	

Entorno 3x3 del píxel  $(x, y)$

Resultado al aplicar la máscara

## 2.1. Suavizado: vecinos

- **Límites de la imagen:** Podemos aplicar la máscara añadiendo un marco de ceros de la anchura adecuada. Esto puede tener efectos no deseados (p. ej., de difuminación en los límites de la imagen) pero en general, poco significativos si la máscara es pequeña en relación con el tamaño de la imagen



- Otra forma: duplicando el borde de la imagen

## 2.1. Suavizado: filtro de media

- Se reemplaza el valor de cada píxel por la media de los valores de los píxeles vecinos. Se puede operar mediante convolución con una máscara determinada
- Supongamos el siguiente núcleo:  $h = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$
- El resultado, sin procesar el borde, sería :

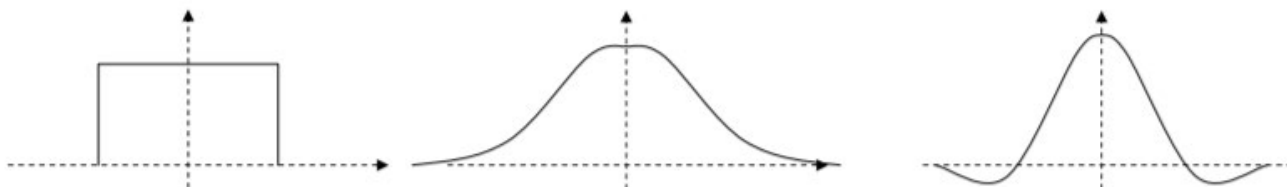
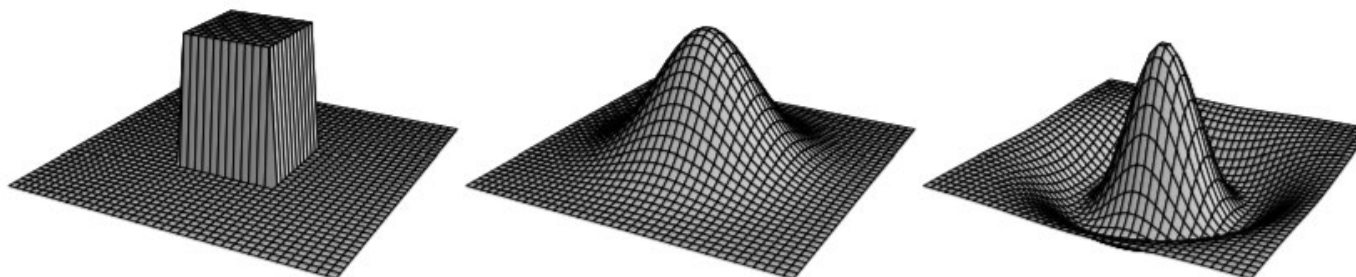
5	1	1	3	2	3
6	4	0	2	3	2
7	7	0	3	3	1
9	7	0	1	1	1
8	8	0	1	1	0
7	7	0	1	0	0



5	1	1	3	2	3
6	3	2	2	2	2
7	4	3	1	2	1
9	5	3	1	1	1
8	5	3	0	0	0
7	7	0	1	0	0



## 2.1. Suavizado: filtro gaussiano



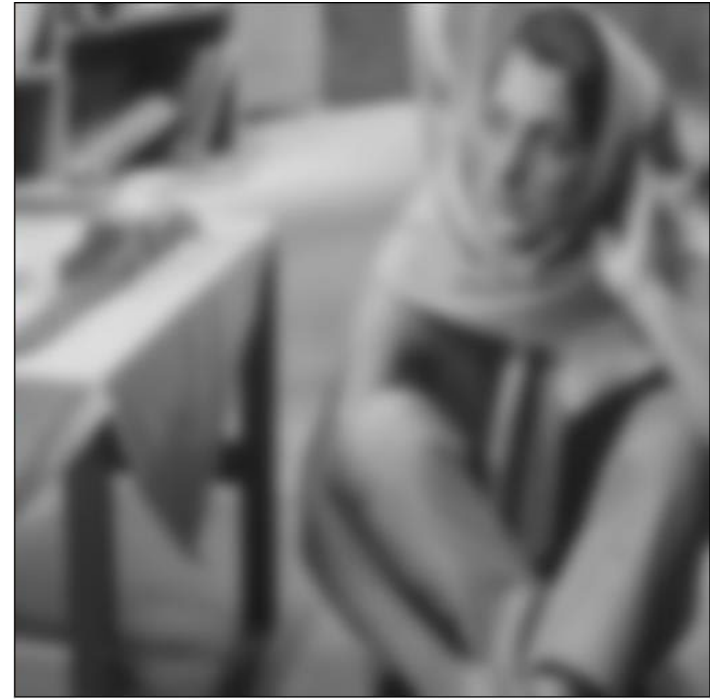
0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

0	1	2	1	0
1	3	5	3	1
2	5	9	5	2
1	3	5	3	1
0	1	2	1	0

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

## 2.1. Suavizado

---



# Índice de contenidos

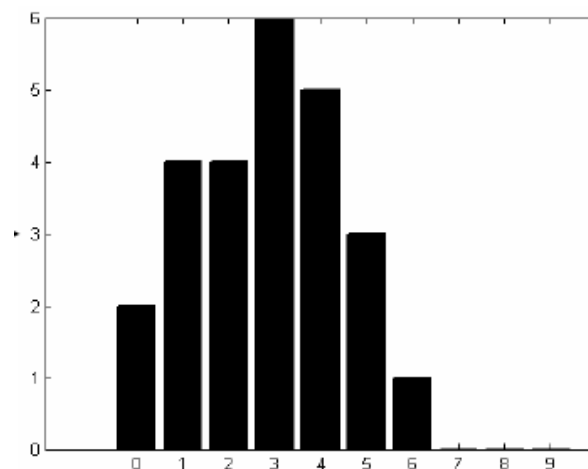
---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

## 2.2. Histograma

- El histograma es la representación gráfica de las frecuencias relativas con las que aparecen los distintos colores en una determinada imagen

$$B = \begin{bmatrix} 0 & 0 & 1 & 2 & 6 \\ 1 & 3 & 3 & 1 & 3 \\ 2 & 2 & 4 & 3 & 3 \\ 2 & 4 & 5 & 4 & 3 \\ 1 & 5 & 5 & 4 & 4 \end{bmatrix}$$



g	0	1	2	3	4	5	6	7	8	9
N(g)	2	4	4	6	5	3	1	0	0	0

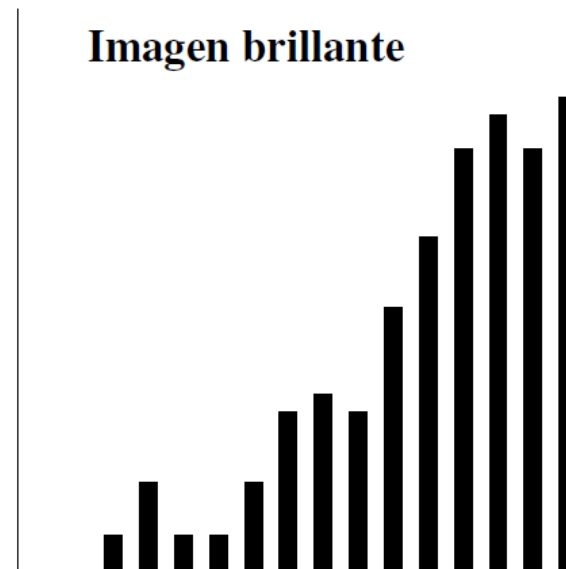
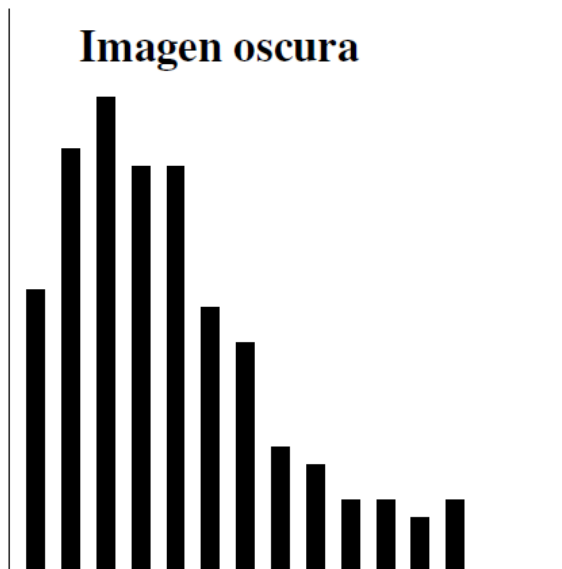
## 2.2. Histograma

---

- Permite analizar la imagen utilizando la distribución de los niveles de gris
- Da información sobre el **contraste** de una imagen
- Al tomar una fotografía, el histograma ofrece información sobre la **exposición** de la imagen
- El histograma de la imagen se puede manipular para mejorarla
- **Número de objetos:** máximo locales (acumulaciones)

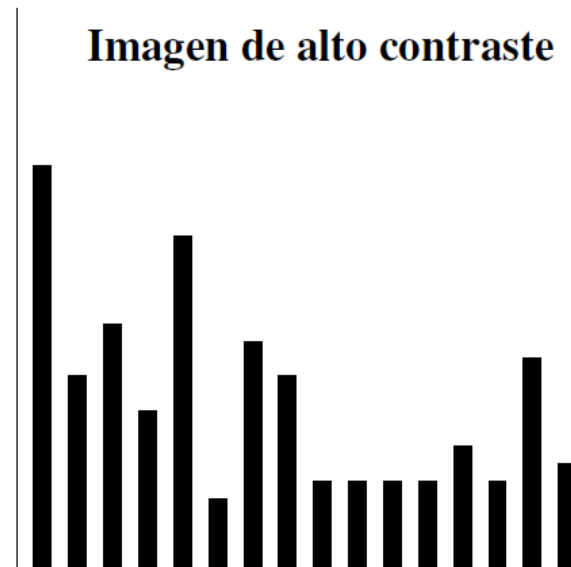
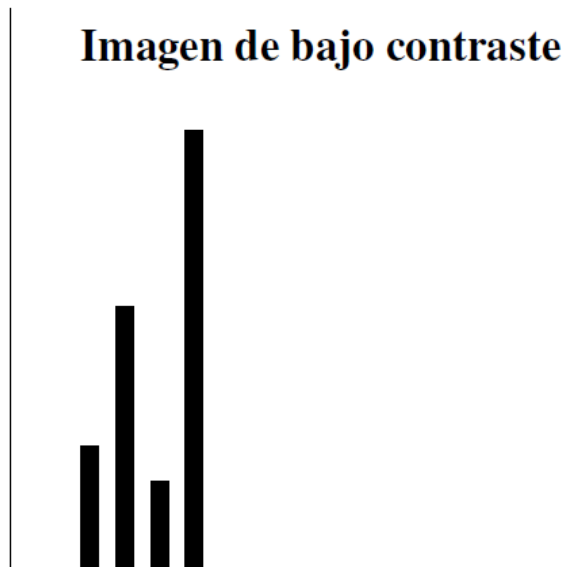
## 2.2. Histograma

- Histogramas:



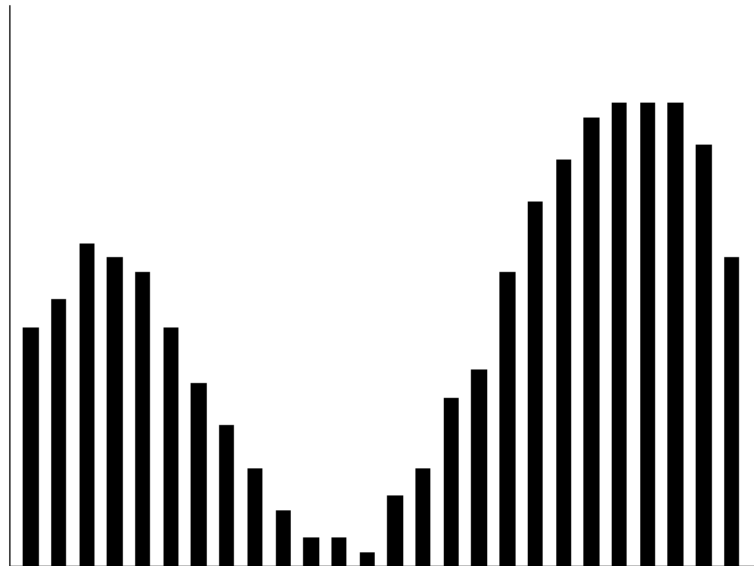
## 2.2. Histograma

- Histogramas:



## 2.2. Histograma

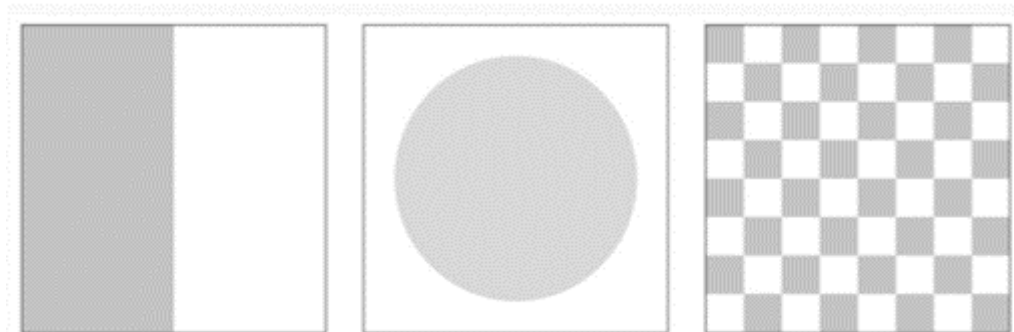
- Histogramas:
  - Imagen con dos objetos bien diferenciados





## 2.2. Histograma: interpretación

- El histograma no ofrece información espacial
- No codifica en donde está cada píxel
- Dado un histograma no es posible reconstruir una imagen
- Varias imágenes pueden tener el mismo histograma



## 2.2. Histograma: interpretación

---

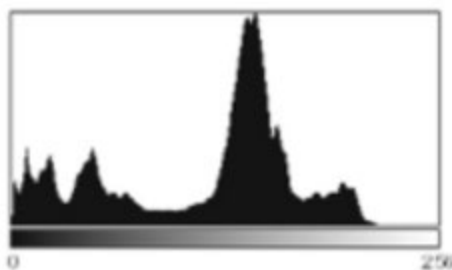
- El histograma muestra información sobre el **contraste** y el **rango dinámico** de la imagen
- También puede mostrar **artefactos** resultantes de pasos de procesamiento de imágenes aplicados a la imagen
- **Artefactos** son áreas pequeñas de una imagen digital que tienen **distorsión** o un **color diferente**
- Examinando el tamaño y uniformidad de la distribución del histograma se puede determinar si la imagen está haciendo un uso efectivo de su rango de intensidades

## 2.2. Histograma: exposición

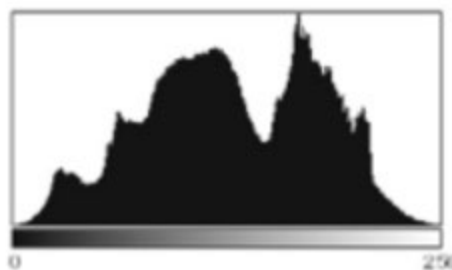
---

- **Exposición** es la **cantidad de luz** que llega el sensor de la cámara
- Es lo que hace que una imagen salga clara u oscura
- Los histogramas hacen evidentes estos problemas
- Por ejemplo, un histograma en el que una gran parte del rango de intensidad en un extremo casi no se utiliza mientras que el otro extremo está lleno de picos de alto valor es representativo de una imagen expuesta incorrectamente

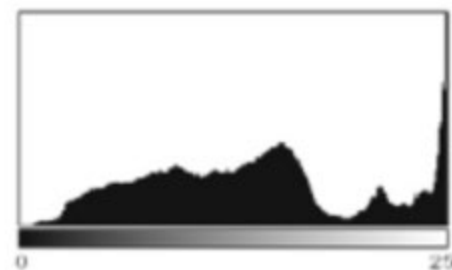
## 2.2. Histograma: exposición



subexposición



normal



sobreexpuesta

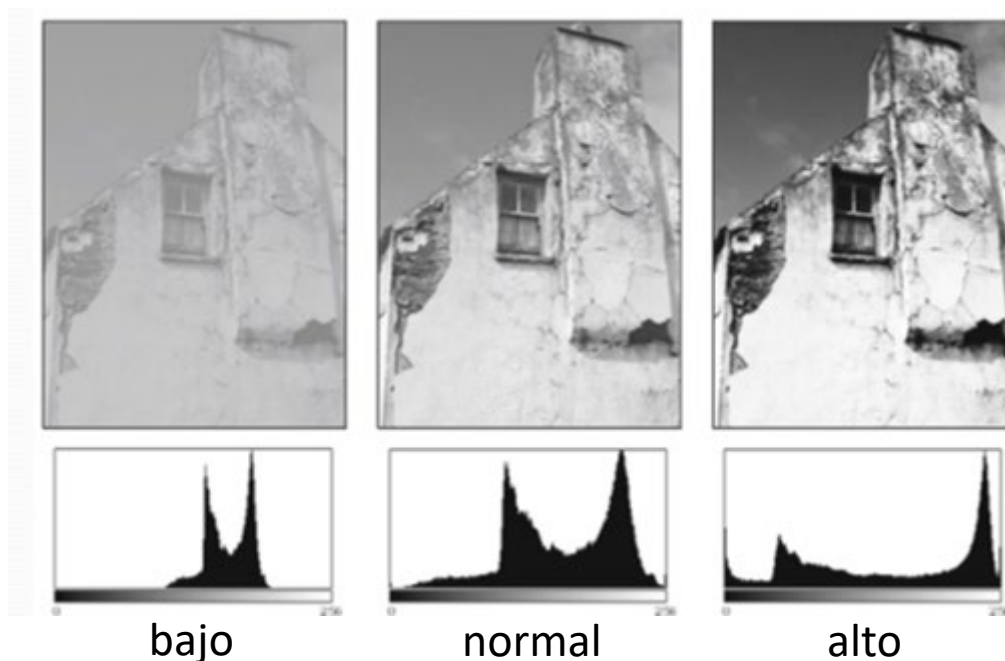
## 2.2. Histograma: contraste

---

- El **contraste** es el **rango de valores** de intensidad utilizados por una imagen
- Es la diferencia entre los valores máximo y mínimo de los píxeles de la imagen ( $a = a_{max} - a_{min}$ )
- Una imagen de contraste completo (*full-contrast*) es la que hace un uso efectivo de todo el rango de valores de intensidad disponibles, negro ( $a_{min} = 0$ ) y blanco ( $a_{max} = K - 1$ )
- Con esta definición, el contraste de la imagen se puede leer fácilmente directamente desde el histograma

## 2.2. Histograma: contraste

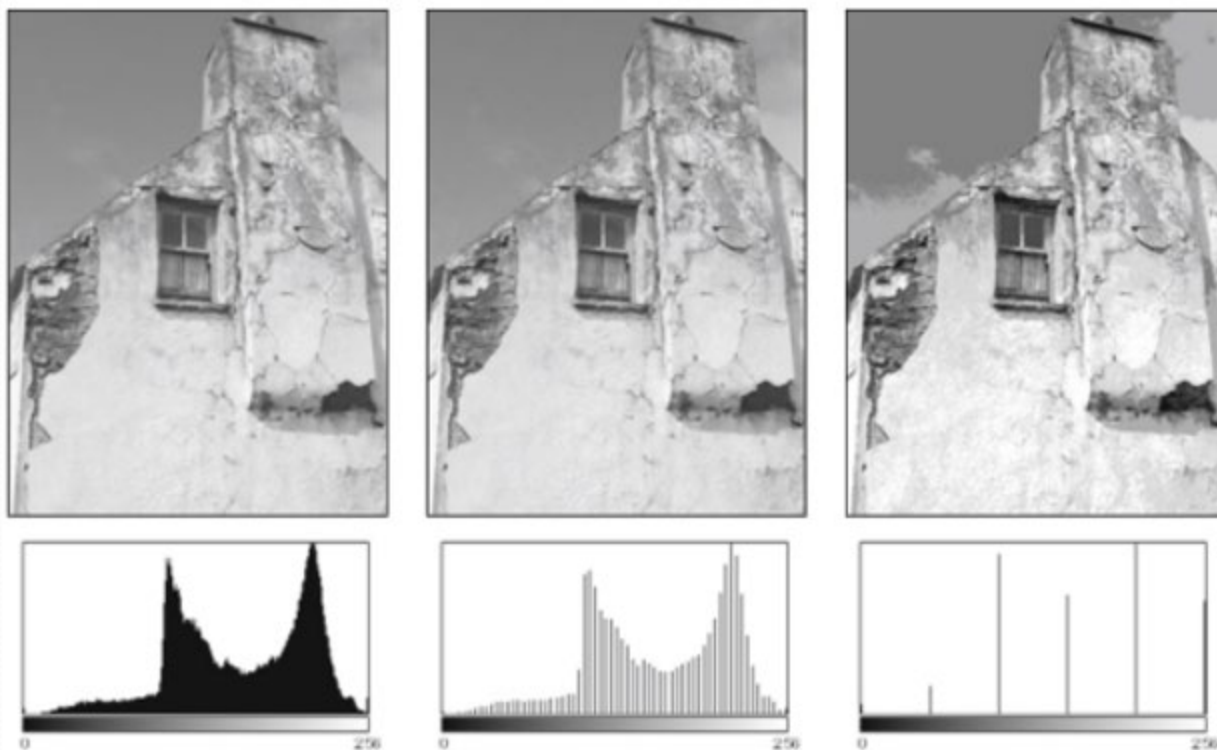
- Alto: hay mucha diferencia entre el tono más negro y más blanco. Hay áreas extremadamente claras y oscuras
- Bajo: no presenta blancos o negros puros, hay una gama de tonos medios



## 2.2. Histograma: rango dinámico

- El **rango dinámico** de una imagen es el **número de valores** de píxeles **distintos** en una imagen
- En el caso ideal, el rango dinámico abarca todos los  $K$  valores de píxeles utilizables, en cuyo caso el rango de valores se utiliza por completo
- Cuando una imagen tiene un rango de contraste disponible  $a = a_{bajo} \dots a_{alto}$  con  $a_{min} < a_{bajo}$  y  $a_{alto} < a_{max}$ , el máximo rango dinámico posible se logra cuando se utilizan todos los valores de intensidad que se encuentran en este rango

## 2.2. Histograma: rango dinámico



alto

bajo  
(64 valores)

muy bajo  
(6 valores)



## 2.2. Histograma: propiedades

- Algunas propiedades estadísticas que se pueden obtener del histograma son:

- Media: 
$$\bar{g} = \sum_{g=0}^{L-1} gP(g) = \sum_i \sum_j \frac{I(i,j)}{M}$$

$$P(g) = \frac{N(g)}{M}$$

- Varianza: 
$$\sigma^2 = \sum_{g=0}^{L-1} (g - \bar{g})^2 P(g)$$

$N(g)$ : Nivel de gris

$M$ : Máximo nivel de gris

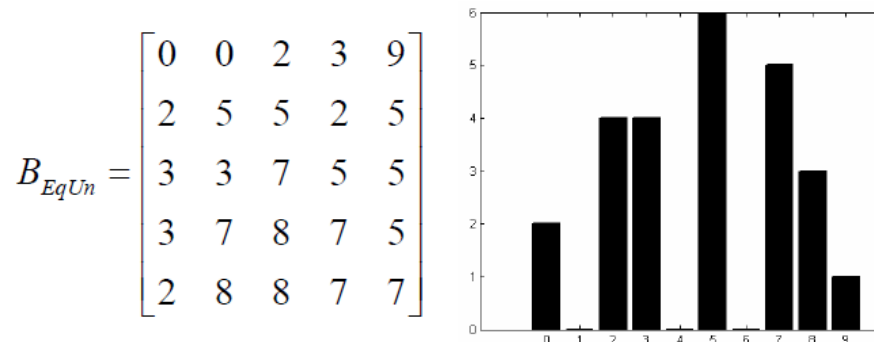
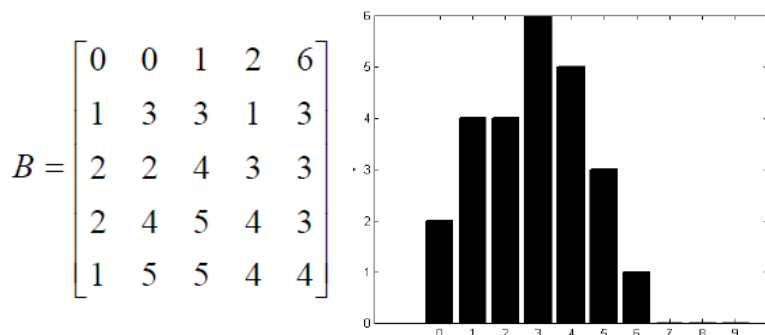
- Asimetría: 
$$a = \sum_{g=0}^{L-1} (g - \bar{g})^3 P(g)$$

- Entropía: 
$$e = - \sum_{g=0}^{L-1} P(g) \log_2 [P(g)]$$

## 2.2. Histograma: ecualización

- Mejora el contraste de la imagen. Reparte de forma más o menos uniforme los valores del histograma
- Ecualización uniforme:  $F(g) = g_{max} \sum_{g=0}^g p(g)$

g	0	1	2	3	4	5	6	7	8	9
N(g)	2	4	4	6	5	3	1	0	0	0
p(g)	$2/25$	$4/25$	$4/25$	$6/25$	$5/25$	$3/25$	$1/25$	$0/25$	$0/25$	$0/25$
P(g)	$2/25$	$6/25$	$10/25$	$16/25$	$21/25$	$24/25$	$25/25$	$25/25$	$25/25$	$25/25$
F(g)	0	2	3	5	7	8	9	9	9	9

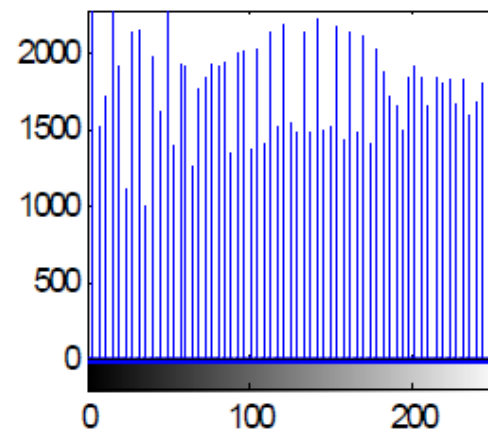
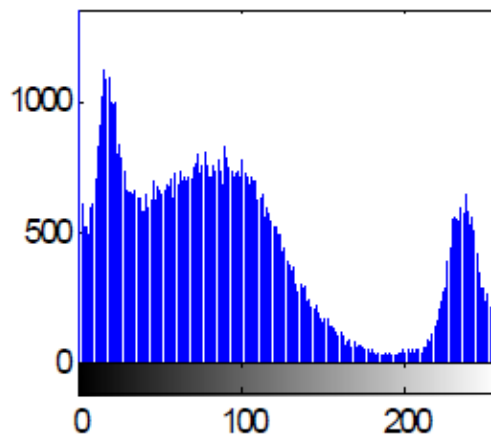
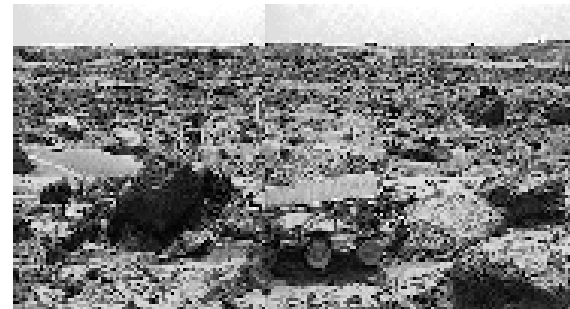
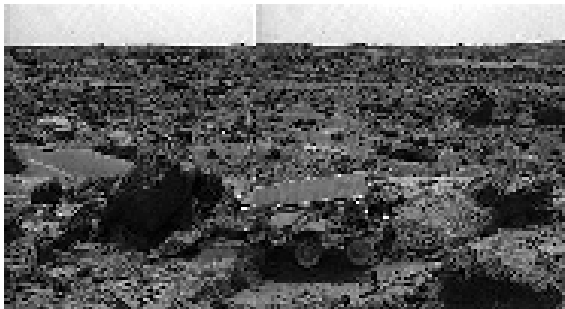


## 2.2. Histograma: ecualización

- Otras funciones de ecualización:

Uniforme	$F(g) = [g_{max} - g_{min}]P_g(g) + g_{min}$
Exponencial	$F(g) = g_{min} - \frac{1}{\alpha} \ln[1 - P_g(g)]$
Rayleigh	$F(g) = g_{min} + \left[ 2\alpha^2 \ln \left\{ \frac{1}{1 - P_g(g)} \right\} \right]^{\frac{1}{2}}$
Hipercúbica	$F(g) = ([\sqrt[3]{g_{max}} - \sqrt[3]{g_{min}}]P_g(g) + \sqrt[3]{g_{min}})^3$
Logaritmo hiperbólico	$F(g) = g_{min} \left[ \frac{g_{max}}{g_{min}} \right] P_g(g)$

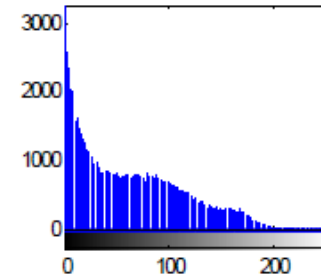
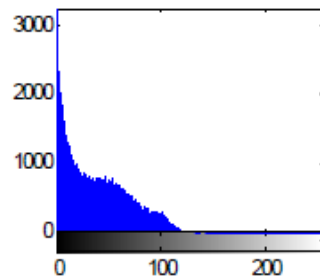
## 2.2. Histograma: ecualización



## 2.2. Histograma: expansión

- Consiste en distribuir las frecuencias en todo el ancho del histograma
- Se modifica el histograma de manera que se distribuyen las intensidades en la escala de valores disponibles para abarcar el máximo posible

$$Out(i, j) = \left[ \frac{In(i, j) - In(i, j)_{MIN}}{In(i, j)_{MAX} - In(i, j)_{MIN}} \right] [MAX - MIN] + MIN$$



## 2.2. Histograma: contracción

---

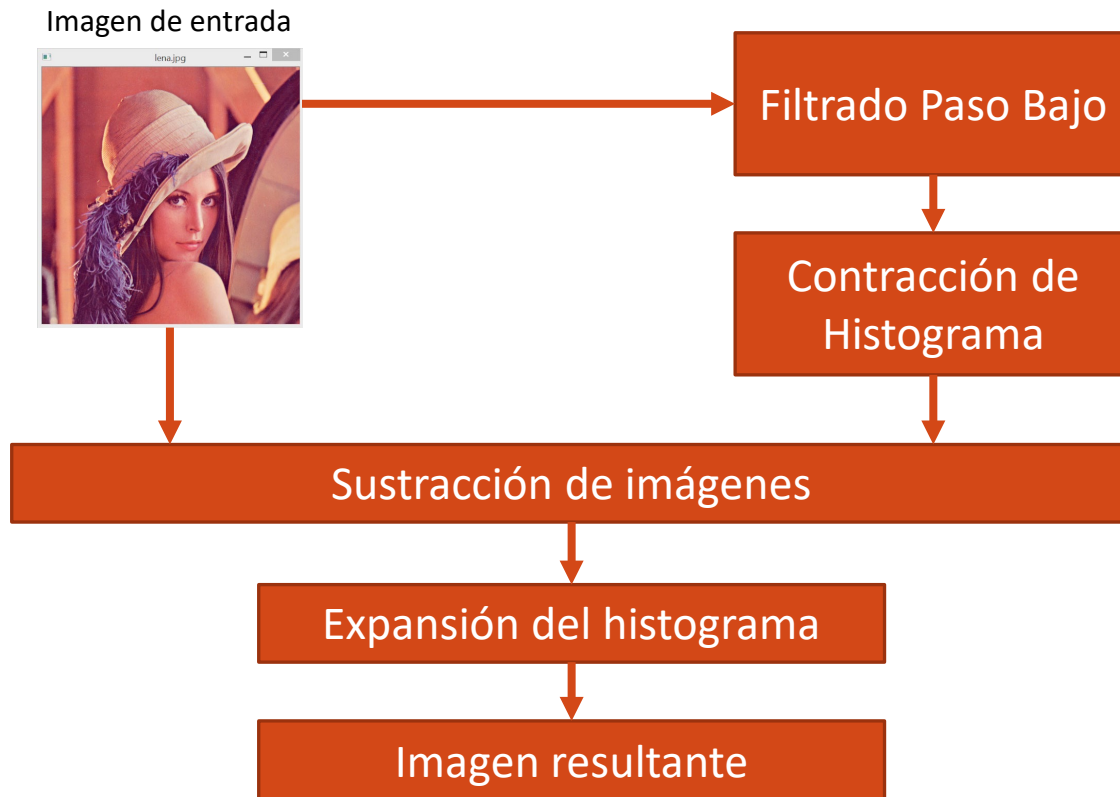
- La operación contraria a la expansión es la contracción
- Al contraer el histograma se disminuye el rango dinámico de la distribución de niveles de gris de la imagen

$$S_k = \frac{C_{max} - C_{min}}{r_{max} - r_{min}} [r_k - r_{min}] + C_{min}$$

- Donde :
  - $C_{max}$  y  $C_{min}$  son los valores deseados de la comprensión
  - $r_{max}$  y  $r_{min}$  son el máximo y mínimo nivel de gris de la imagen de respectivamente
  - $r_k$  es el nivel del pixel evaluado

## 2.2. Histograma: realzado

- Las técnicas de realzado se pueden combinar



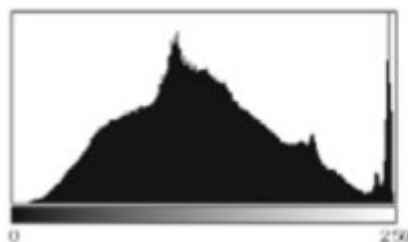
## 2.2. Histograma: defectos

---

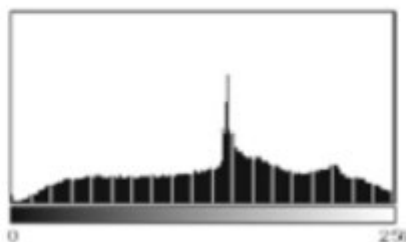
- Los histogramas se pueden utilizar para detectar defectos de imagen originados durante la adquisición o como resultado de un procesamiento posterior de imágenes
- En general, las imágenes correctamente adquiridas y sin procesar, tienen una distribución suave de intensidades
- Los picos y huecos en el histograma aparecen cuando la imagen ha sido procesada:
  - Cuando el rango de intensidades de la luz de la escena es mayor que el rango del sensor de la cámara, aparecen picos en los extremos del histograma
  - Al **aumentar el contraste** de una imagen se crean **huecos** y al **reducir el contraste** se crean **picos** en el histograma



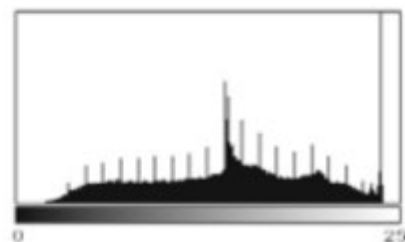
## 2.2. Histograma: defectos



saturación de  
altas intensidades



huecos: incremento  
del contraste



picos: reducción  
del contraste

# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

## 2.3. Histograma: filtro homomórfico

- Simultáneamente **normaliza el brillo** (comprime el rango de intensidad, iluminación) y **aumenta el contraste** de una imagen:

$$Im(x, y) = i(x, y) * r(x, y)$$

- Donde  $Im$  es la imagen,  $i$  es la iluminación (bajas frecuencias) y  $r$  la reflectancia (altas frecuencias)
- Para aplicar un filtro de paso alto, es necesario transformar la ecuación en el dominio de la frecuencia
- Como es muy difícil hacer un cálculo después de aplicar la transformación de Fourier a esta ecuación porque ya no es una ecuación de producto, usamos la escala logarítmica 'log' para ayudar a resolver este problema

$$\ln(Im(x, y)) = \ln(i(x, y)) + \ln(r(x, y))$$

## 2.3. Histograma: filtro homomórfico

- Luego se aplica la transformada de Fourier

$$F(\ln(I_m(x, y))) = F(\ln(i(x, y))) + F(\ln(r(x, y)))$$



$$F(u, v) = I(u, v) + R(u, v)$$

- A continuación, se aplica un filtro de paso alto  $H$  a la imagen. Esto hace que la iluminación sea más uniforme, se aumentan los componentes de alta frecuencia y se reducen los componentes de baja frecuencia

$$N(u, v) = H(u, v) \cdot F(u, v)$$

- Donde  $H$  es cualquier filtro de paso alto y  $N$  la imagen filtrada en el dominio de la frecuencia

## 2.3. Histograma: filtro homomórfico

- Posteriormente, se devuelve el dominio de la frecuencia al dominio espacial utilizando la transformada de Fourier inversa

$$n(x, y) = invF(N(u, v))$$

- Finalmente, usando la función exponencial para eliminar el registro que usamos al principio se obtiene la imagen mejorada

$$newImage(x, y) = \exp(n(x, y))$$

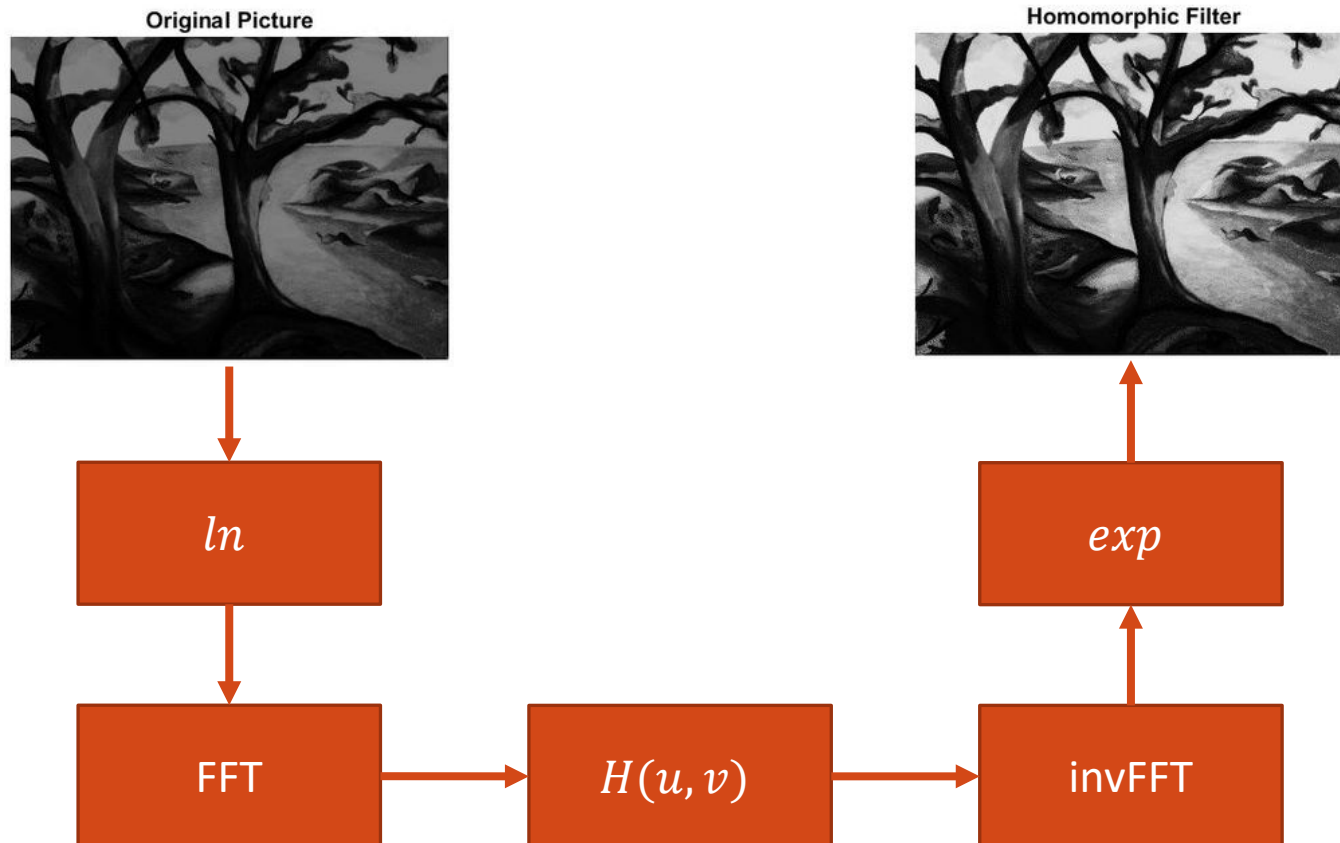
Original Picture



Homomorphic Filter



## 2.3. Histograma: filtro homomórfico



# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

## 2.4. Correspondencia de histogramas

- Dadas dos imágenes A y B y considerando como referencia la primera, se trata de modificar la imagen B tomando como referencia la A
- Este proceso se lleva a cabo a partir de los histogramas  $h_A$  y  $h_B$  respectivamente a partir de los cuales se obtienen los valores de probabilidad acumulados para cada nivel de gris  $g_a$  y  $g_b$  en las respectivas imágenes A y B como sigue:

$$P(g_a) = \sum_{i=0}^{g_a} P(g_i)$$

$$P(g_b) = \sum_{i=0}^{g_b} P(g_i)$$

- El procedimiento de correspondencia consiste en buscar para cada valor  $P(g_b)$  asociado con el nivel de gris  $g_b$  cuál es el valor más próximo  $P(g_a)$  a aquél que nos permita intercambiar el valor  $g_b$  por  $g_a$  en la imagen
- Tras este intercambio de niveles de intensidad se obtiene una nueva imagen  $B_t$  transformada



## 2.4. Correspondencia de histogramas

$$A = \begin{bmatrix} 1 & 1 & 0 & 1 & 4 \\ 2 & 2 & 2 & 3 & 5 \\ 2 & 2 & 3 & 2 & 5 \\ 4 & 4 & 1 & 2 & 4 \\ 3 & 4 & 1 & 2 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 4 & 3 & 2 & 5 & 7 \\ 7 & 8 & 4 & 4 & 7 \\ 6 & 5 & 4 & 9 & 3 \\ 9 & 6 & 5 & 2 & 8 \\ 8 & 8 & 7 & 7 & 6 \end{bmatrix}$$

Imagen A										
g	0	1	2	3	4	5	6	7	8	9
N(g)	1	6	8	3	5	2	0	0	0	0
P(g)	$1/25$	$7/25$	$15/25$	$18/25$	$23/25$	$25/25$	$25/25$	$25/25$	$25/25$	$25/25$

Imagen B										
g	0	1	2	3	4	5	6	7	8	9
N(g)	0	0	2	2	4	3	3	5	4	2
P(g)	$0/25$	$0/25$	$2/25$	$4/25$	$8/25$	$11/25$	$14/25$	$19/25$	$23/25$	$25/25$

## 2.4. Correspondencia de histogramas

$$A = \begin{bmatrix} 1 & 1 & 0 & 1 & 4 \\ 2 & 2 & 2 & 3 & 5 \\ 2 & 2 & 3 & 2 & 5 \\ 4 & 4 & 1 & 2 & 4 \\ 3 & 4 & 1 & 2 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 4 & 3 & 2 & 5 & 7 \\ 7 & 8 & 4 & 4 & 7 \\ 6 & 5 & 4 & 9 & 3 \\ 9 & 6 & 5 & 2 & 8 \\ 8 & 8 & 7 & 7 & 6 \end{bmatrix}$$

$8/_{25}$  de B  $\longrightarrow$   $7/_{25}$  de A

Imagen A										
g	0	1	2	3	4	5	6	7	8	9
P(g)	$1/_{25}$	$7/_{25}$	$15/_{25}$	$18/_{25}$	$23/_{25}$	$25/_{25}$	$25/_{25}$	$25/_{25}$	$25/_{25}$	$25/_{25}$

Imagen B										
g	0	1	2	3	4	5	6	7	8	9
P(g)	$0/_{25}$	$0/_{25}$	$2/_{25}$	$4/_{25}$	$8/_{25}$	$11/_{25}$	$14/_{25}$	$19/_{25}$	$23/_{25}$	$25/_{25}$

## 2.4. Correspondencia de histogramas

4 de B  1 de A

Imagen $B$									
0	1	2	3	4	5	6	7	8	9
Imagen $B_t$									
0	0	0	1	1	2	2	3	4	5

$$A = \begin{bmatrix} 1 & 1 & 0 & 1 & 4 \\ 2 & 2 & 2 & 3 & 5 \\ 2 & 2 & 3 & 2 & 5 \\ 4 & 4 & 1 & 2 & 4 \\ 3 & 4 & 1 & 2 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 4 & 3 & 2 & 5 & 7 \\ 7 & 8 & 4 & 4 & 7 \\ 6 & 5 & 4 & 9 & 3 \\ 9 & 6 & 5 & 2 & 8 \\ 8 & 8 & 7 & 7 & 6 \end{bmatrix}$$

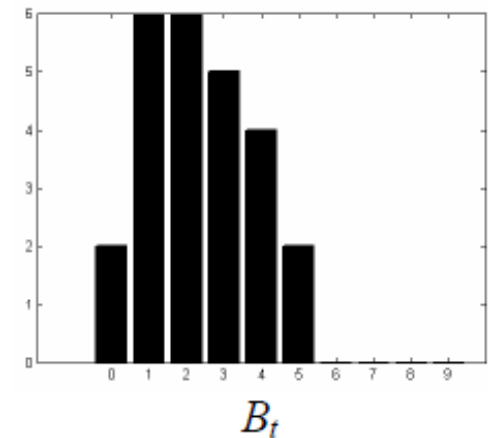
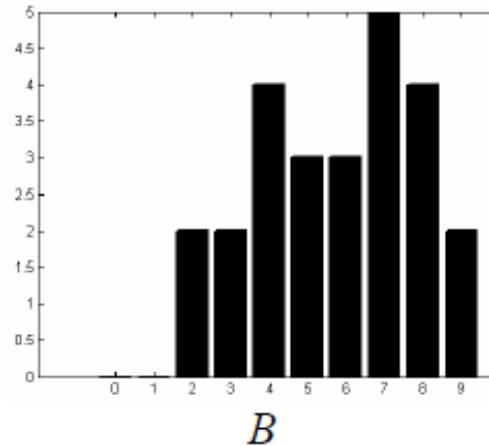
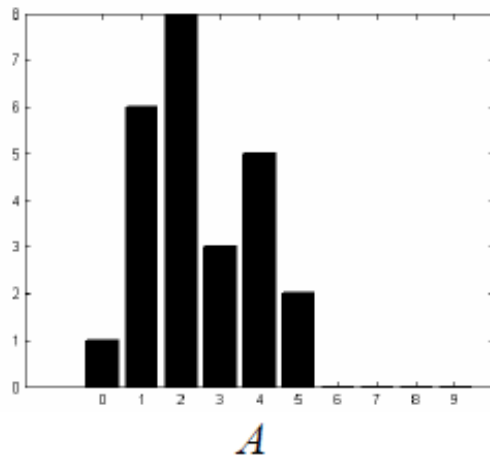
$$B_t = \begin{bmatrix} 1 & 1 & 0 & 2 & 3 \\ 3 & 4 & 1 & 1 & 3 \\ 2 & 2 & 1 & 5 & 1 \\ 5 & 2 & 2 & 0 & 4 \\ 4 & 4 & 3 & 3 & 2 \end{bmatrix}$$

## 2.4. Correspondencia de histogramas

$$A = \begin{bmatrix} 1 & 1 & 0 & 1 & 4 \\ 2 & 2 & 2 & 3 & 5 \\ 2 & 2 & 3 & 2 & 5 \\ 4 & 4 & 1 & 2 & 4 \\ 3 & 4 & 1 & 2 & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 4 & 3 & 2 & 5 & 7 \\ 7 & 8 & 4 & 4 & 7 \\ 6 & 5 & 4 & 9 & 3 \\ 9 & 6 & 5 & 2 & 8 \\ 8 & 8 & 7 & 7 & 6 \end{bmatrix}$$

$$B_t = \begin{bmatrix} 1 & 1 & 0 & 2 & 3 \\ 3 & 4 & 1 & 1 & 3 \\ 2 & 2 & 1 & 5 & 1 \\ 5 & 2 & 2 & 0 & 4 \\ 4 & 4 & 3 & 3 & 2 \end{bmatrix}$$



# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

## 2.5. Comparar histogramas

- Para comparar dos histogramas  $H_1$  y  $H_2$ , es necesario aplicar alguna métrica  $d(H_1, H_2)$  que nos indique cómo de bien coinciden dos histogramas:

- Correlación: 
$$d(H_1, H_2) = \frac{\sum_{i=0}^N (H_1(i) - \bar{H}_1) (H_2(i) - \bar{H}_2)}{\sqrt{\sum_{i=0}^N (H_1(i) - \bar{H}_1)^2 \sum_{i=0}^N (H_2(i) - \bar{H}_2)^2}}$$

donde  $\bar{H}_k = \frac{1}{N} \sum_{j=0}^N H_k(j)$  y  $N$  es el número de bins del histograma

- Chi-square: 
$$d(H_1, H_2) = \sum_{i=0}^N \frac{(H_1(i) - H_2(i))^2}{H_1(i)}$$

## 2.5. Comparar histogramas

- Intersección:

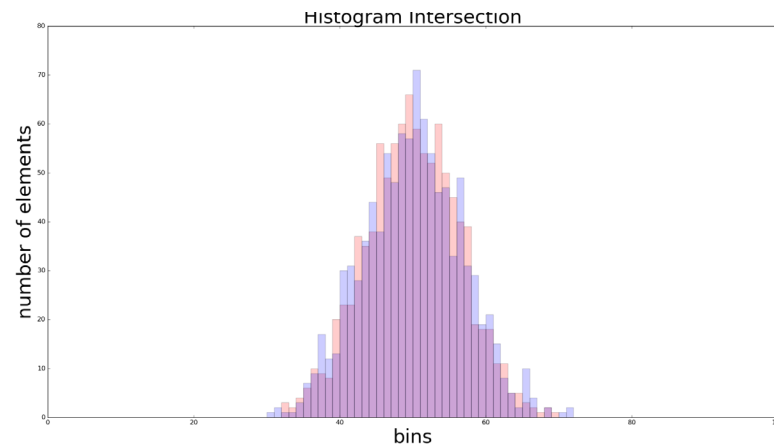
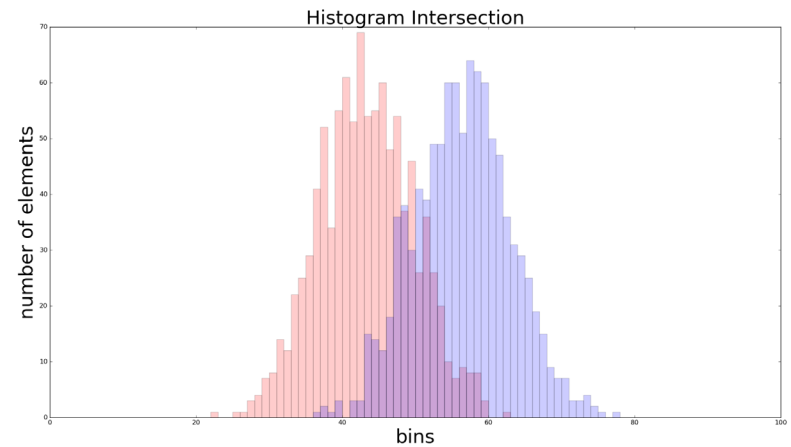
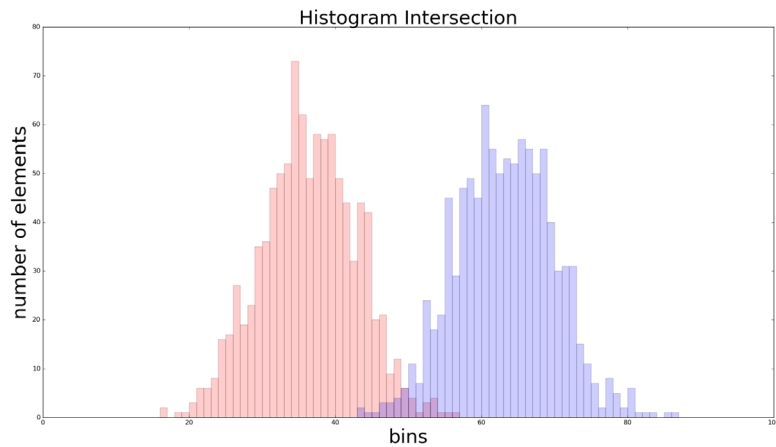
$$d(H_1, H_2) = \sum_{i=0}^N \min(H_1(i), H_2(i))$$

- Distancia Bachattacharyya:

$$d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_1 * \bar{H}_2 * N^2} \sum_{i=0}^N \sqrt{H_1(i) * H_2(i)}}$$

- Para los métodos de **Correlación e Intersección**, cuanto **mayor** es el **valor** obtenido, **mayor** es la **coincidencia**
- Para el método de **Chi-square y Bachattacharyya**, cuanto **menor** es el **valor** obtenido, **mayor** es la **coincidencia**

## 2.5. Comparar histogramas





# Índice de contenidos

---

1. Transformaciones geométricas:
  - Transformadas elementales
  - Sistema de coordenadas
  - Interpolación
  - Registro
  - Ejemplos
2. Transformaciones radiométricas:
  - Suavizado
  - Histograma
  - Filtrado homomórfico
  - Correspondencia de histogramas
  - Comparación de histogramas
  - Ejemplos

## 2.6. Ejemplos: histograma ecualización

```
int main( int argc, char** argv ) {
    // Read image
    // ...

    // Split BGR planes
    vector<Mat> bgr_planes;
    split( src, bgr_planes );

    // Establish the number of
    int histSize = 256;
    // Set the ranges ( for B,G,R )
    float range[] = { 0, 256 }; //the upper bounda
    const float* histRange = { range };
    bool uniform = true, accumulate = false;

    // Compute the histograms for each channel
    Mat b_hist, g_hist, r_hist;
    calcHist( &bgr_planes[0], 1, 0, Mat(), b_hist, 1, &histSize, &histRange, uniform, accumulate );
    calcHist( &bgr_planes[1], 1, 0, Mat(), g_hist, 1, &histSize, &histRange, uniform, accumulate );
    calcHist( &bgr_planes[2], 1, 0, Mat(), r_hist, 1, &histSize, &histRange, uniform, accumulate );

    // ...
}
```

Se separan los tres planos de la imagen a color

Se establece el número de divisiones (bins),  
el cuál estará en el intervalo [0,255]

Indicamos si los bins tienen el  
mismo tamaño (uniform)

Se calculan los histogramas  
con la función **calcHist**

## 2.6. Ejemplos: histograma ecualización

```
int main( int argc, char** argv ) {
    // ...
    // Draw the histograms for B, G and R
    int hist_w = 512, hist_h = 400;
    int bin_w = cvRound( (double) hist_w/histSize );

    Mat histImage( hist_h, hist_w, CV_8UC3, Scalar( 0,0,0) );

    // normalize the histograms between 0 and histImage.rows
    normalize(b_hist, b_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat() );
    normalize(g_hist, g_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat() );
    normalize(r_hist, r_hist, 0, histImage.rows, NORM_MINMAX, -1, Mat() );

    // Draw the intensity line for histograms
    for( int i = 1; i < histSize; i++ ) {
        line( histImage, Point( bin_w*(i-1), hist_h - cvRound(b_hist.at<float>(i-1)) ),
              Point( bin_w*(i), hist_h - cvRound(b_hist.at<float>(i)) ),
              Scalar( 255, 0, 0), 2, 8, 0 );
        line( histImage, Point( bin_w*(i-1), hist_h - cvRound(g_hist.at<float>(i-1)) ),
              Point( bin_w*(i), hist_h - cvRound(g_hist.at<float>(i)) ),
              Scalar( 0, 255, 0), 2, 8, 0 );
        line( histImage, Point( bin_w*(i-1), hist_h - cvRound(r_hist.at<float>(i-1)) ),
              Point( bin_w*(i), hist_h - cvRound(r_hist.at<float>(i)) ),
              Scalar( 0, 0, 255), 2, 8, 0 );
    }
    // ...
}
```

Se normalizan los histogramas entre 0 y el número de filas del histograma calculado

Pintamos una línea por cada histograma

## 2.6. Ejemplos: histograma ecualización

```
int main( int argc, char** argv ) {
    // ...
    // Equalization
    Mat b_eqhist,g_eqhist,r_eqhist;
    equalizeHist( bgr_planes[0], b_eqhist );
    equalizeHist( bgr_planes[1], g_eqhist );
    equalizeHist( bgr_planes[2], r_eqhist );

    // Compute the histograms for each channel
    // normalize the histograms between 0 and histImage.rows
    // Draw the intensity line for equalized histograms

    // Create Equalized image
    vector<Mat> equalized;
    equalized.push_back(b_eqhist);
    equalized.push_back(g_eqhist);
    equalized.push_back(r_eqhist);
    // Merge channels
    Mat equalized_image;
    merge(equalized, equalized_image);

    // Show images
    imshow("Equalized image", equalized_image );
    imshow("calcHist Equalized", histImageEq );
}
```

Para ecualizar un canal, se utiliza la función **equalizeHist** la cuál recibe el canal en cuestión, no el histograma

Si se quiere visualizar el histograma ecualizado, se realiza como en el caso anterior

Para ver la imagen ecualizada, se compone la misma a partir de los tres canales resultantes de la ecualización

Se muestra la imagen ecualizada y su histograma

## 2.6. Ejemplos: comparación histograma

```
int main( int argc, char** argv ) {
    // Load the base image (src_base) and the other two test images:
    Mat src_base = imread( "../images/Histogram_Comparison_Source_0.jpg", IMREAD_COLOR );
    Mat src_test1 = imread( "../images/Histogram_Comparison_Source_1.jpg", IMREAD_COLOR );
    Mat src_test2 = imread( "../images/Histogram_Comparison_Source_2.jpg", IMREAD_COLOR );
    if( src_base.empty() || src_test1.empty() || src_test2.empty() ) {
        cout << "Could not open or find the images!\n" << endl;
        return -1;
    }

    // Convert them to HSV format
    Mat hsv_base, hsv_test1, hsv_test2;
    cvtColor( src_base, hsv_base, COLOR_BGR2HSV );
    cvtColor( src_test1, hsv_test1, COLOR_BGR2HSV );
    cvtColor( src_test2, hsv_test2, COLOR_BGR2HSV );

    imshow("im1", hsv_base);
    imshow("im2", hsv_test1);
    imshow("im3", hsv_test2);

    // Also, create an image of half the base image (in HSV format):
    Mat hsv_half_down = hsv_base( Range( hsv_base.rows/2, hsv_base.rows ), Range( 0, hsv_base.cols ) );
    imshow("half", hsv_half_down);

    // ...
}
```

Se cargan la imagen de referencia (src\_base) y las otras dos imágenes que serán de test, para comparar

Se convierten a formato HSV

Se crea una nueva imagen que es la mitad de la imagen de referencia

## 2.6. Ejemplos: comparación histograma

```
int main( int argc, char** argv ) {
    // ...

    // Initialize the arguments to calculate the histograms
    int h_bins = 50, s_bins = 60;
    int histSize[] = { h_bins, s_bins };
    // hue varies from 0 to 179, saturation from 0 to 255
    float h_ranges[] = { 0, 180 };
    float s_ranges[] = { 0, 256 };
    const float* ranges[] = { h_ranges, s_ranges };
    // Use the 0-th and 1-st channels
    int channels[] = { 0, 1 };

    // Calculate the Histograms for the base image, the 2 test images and the half base image:
    Mat hist_base, hist_half_down, hist_test1, hist_test2;
    calcHist( &hsv_base, 1, channels, Mat(), hist_base, 2, histSize, ranges, true, false );
    normalize( hist_base, hist_base, 0, 1, NORM_MINMAX, -1, Mat() );
    calcHist( &hsv_half_down, 1, channels, Mat(), hist_half_down, 2, histSize, ranges, true, false );
    normalize( hist_half_down, hist_half_down, 0, 1, NORM_MINMAX, -1, Mat() );
    calcHist( &hsv_test1, 1, channels, Mat(), hist_test1, 2, histSize, ranges, true, false );
    normalize( hist_test1, hist_test1, 0, 1, NORM_MINMAX, -1, Mat() );
    calcHist( &hsv_test2, 1, channels, Mat(), hist_test2, 2, histSize, ranges, true, false );
    normalize( hist_test2, hist_test2, 0, 1, NORM_MINMAX, -1, Mat() );

    // ...
}
```

Se inicializan los argumentos para  
calcular los histogramas  
(divisiones, rangos y canales H y S)

Se calculan los histogramas de la  
imagen base, las dos imágenes de  
test y la imagen recortada

## 2.6. Ejemplos: comparación histograma

```
int main( int argc, char** argv ) {
    // ...

    // Apply sequentially the 4 comparison methods between the
    // histogram of the base image (hist_base) and the other histograms:
    for( int compare_method = 0; compare_method < 4; compare_method++ ) {
        double base_base  = compareHist( hist_base, hist_base,      compare_method );
        double base_half  = compareHist( hist_base, hist_half_down, compare_method );
        double base_test1 = compareHist( hist_base, hist_test1,    compare_method );
        double base_test2 = compareHist( hist_base, hist_test2,    compare_method );
        cout << "Method " << compare_method << " Perfect, Base-Half, Base-Test(1), Base-Test(2) : "
              << base_base << " / " << base_half << " / " << base_test1 << " / " << base_test2 << endl;
    }

    // For the Correlation and Intersection methods, the more accurate the match.
    // For the other two metrics, the less the match.

    cout << "Done \n";
    waitKey(0);
    return 0;
}
```

Se aplican secuencialmente los 4 métodos de comparación entre los histogramas de la imagen base (base\_image) y los demás histogramas