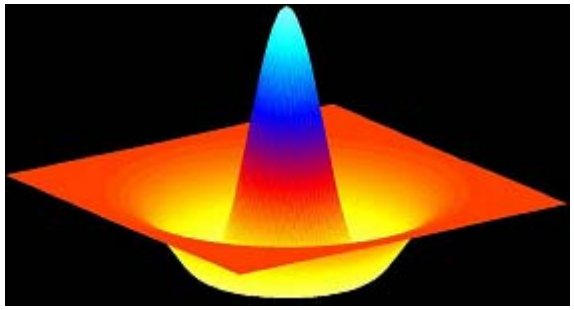
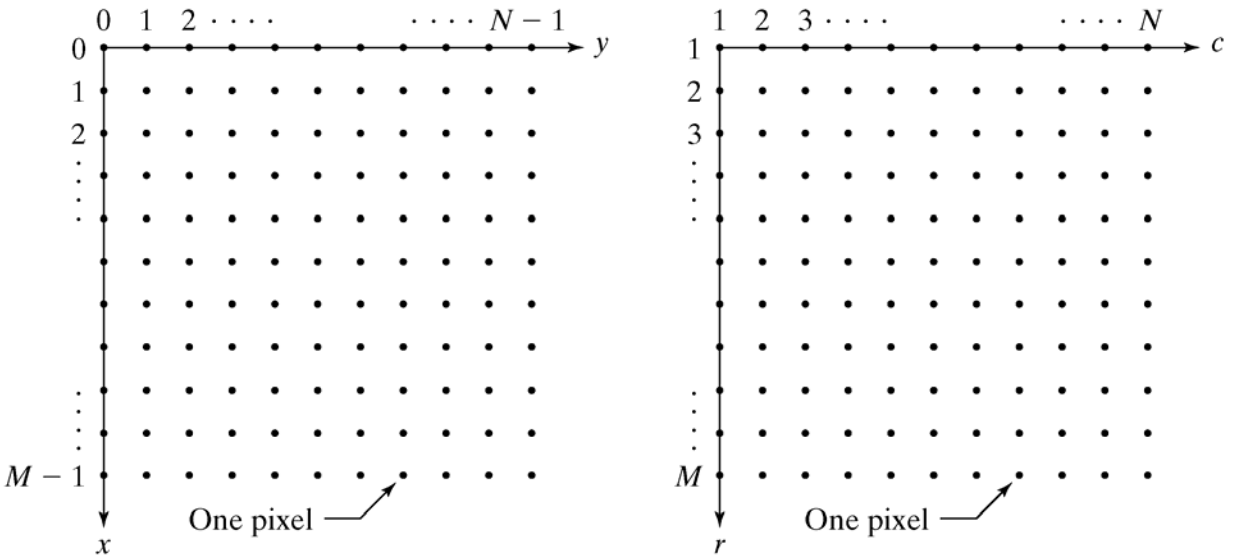


Procesamiento Digital de Imágenes

Fundamentos

Sistema de Coordenadas



a b

FIGURE 2.1 Coordinate conventions used (a) in many image processing books, and (b) in the Image Processing Toolbox.

IPT Matlab

Representación Matricial

$$f = \begin{bmatrix} f(1,1) & f(1,2) & \cdots & f(1,N) \\ f(2,1) & f(2,2) & \cdots & f(2,N) \\ \vdots & \vdots & & \vdots \\ f(M,1) & f(M,2) & \cdots & f(M,N) \end{bmatrix}$$

Imagen digital

Imagen Digital → Matriz M x N $f(m,n)$ → Valor de intensidad del pixel

M: nro. de filas

N: nro. de columnas

Formatos de Imagen

Format Name	Description	Recognized Extensions
TIFF	Tagged Image File Format	`.tif`, `.tiff`
JPEG	Joint Photographic Experts Group	`.jpg`, `.jpeg`
GIF	Graphics Interchange Format†	`.gif`
BMP	Windows Bitmap	`.bmp`
PNG	Portable Network Graphics	`.png`
XWD	X Window Dump	`.xwd`

† `GIF` is supported by `imread`, but not by `imwrite`.

TABLE 2.1 Some of the image/graphics formats supported by `imread` and `imwrite`, starting with MATLAB 6.5. Earlier versions support a subset of these formats. See online help for a complete list of supported formats.

Lectura de Imágenes

```
>>f = imread('cameraman.tif');
>>[M,N] = size(f);
>>whos f
```

Visualización de Imágenes

`>>imshow(f,G)`

`G:` niveles de intensidad usado para mostrar la imagen (si se omite, default `G=256`)

Visualización de Imágenes

```
>>imshow(f,[low,high])
```

Muestra en negro los valores de intensidad menores o iguales que `low`, y en blanco los valores mayores o iguales que `high`.

```
>>imshow(f,[ ])
```

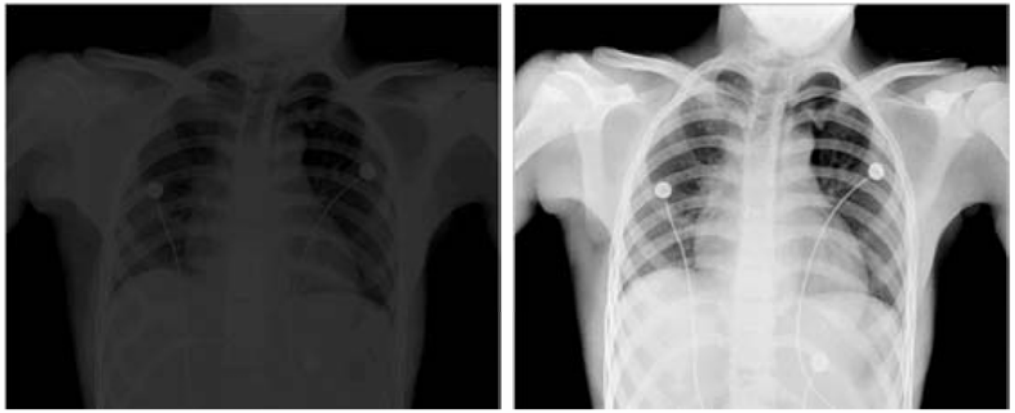
Setea como `low` al menor valor en `f`, y como `high` al máximo valor. (mejora el rango dinámico)

`>>pixval`

`>>impixelinfo`

Permite obtener la intensidad de cada pixel.

Visualización de Imágenes



a b

FIGURE 2.3 (a) An image, h, with low dynamic range. (b) Result of scaling by using `imshow(h,[])`. (Original image courtesy of Dr. David R. Pickens, Dept. of Radiology & Radiological Sciences, Vanderbilt University Medical Center.)

La Figura 2.3(a) tiene bajo rango dinámico

```
>>h = imread('xray-chest.png');
```

```
>>imshow(h)
```

Se mejora el rango dinámico (Fig, 2.3(b)) con los comandos

```
>>figure, imshow(h,[])
```

Escritura de Imágenes

```
>>imwrite(h,'filename','tif')
>>imwrite(h,'filename.jpg',q)
```

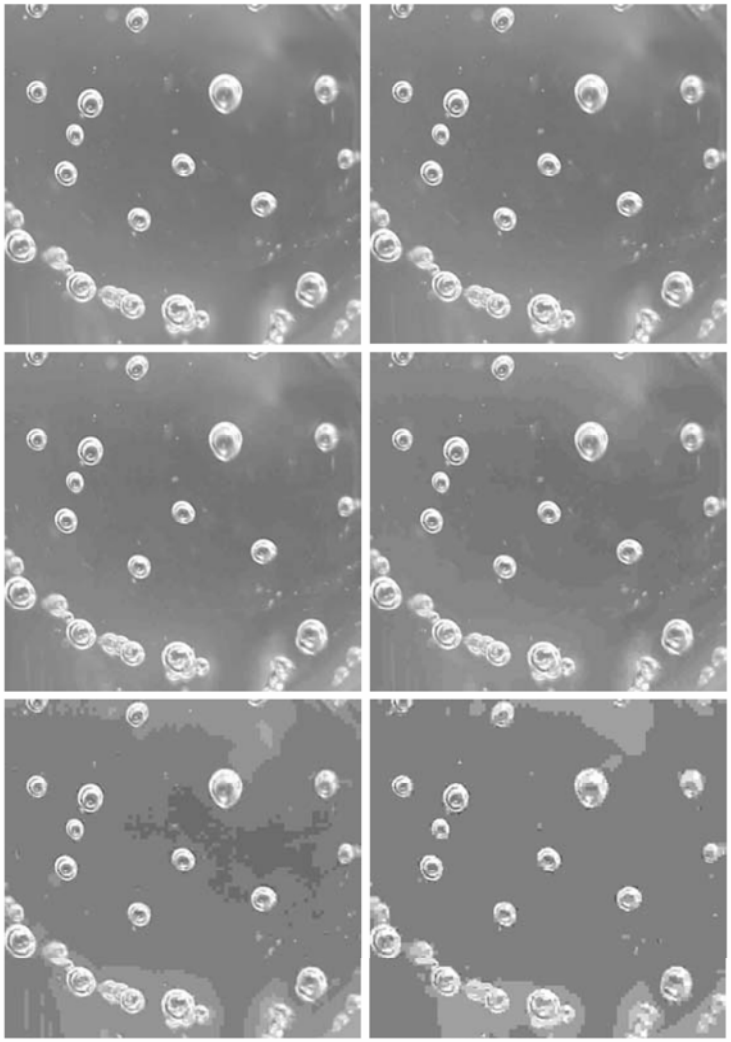
Donde q determina el grado de compresión jpeg (0 < q < 100).

Los detalles de una imagen pueden verse con el comando:

```
>>imfinfo bubbles.png
```

Escritura de Imágenes

Un factor de compresión alto (q bajo) introduce artefactos en la imagen.



a b
c d
e f

FIGURE 2.4
(a) Original image. (b) through (f) Results of using jpg quality values q = 50, 25, 15, 5, and 0, respectively. False contouring begins to be barely noticeable for q = 15 [image (d)] but is quite visible for q = 5 and q = 0.

```
X = imread('bubbles.png');
imwrite(X,'bubbles50.jpeg','q',50);
imwrite(X,'bubbles25.jpeg','q',25);
imwrite(X,'bubbles10.jpeg','q',10);
imwrite(X,'bubbles05.jpeg','q',5);
X05=imread('bubbles05.jpeg');
K=imfinfo('bubbles05.jpeg');
X05_bytes=K.Width*K.Height*K.BitDepth/8;
X05_Compressed_bytes=K.FileSize;
X05_Compression_ratio=X05_bytes/X05_Compressed_bytes;
X05_Compression_ratio =

  163.8396
```

Métricas de Distorsión de Imágenes

$A_{M\times N}$ imagen original

$\tilde{A}_{M\times N}$ imagen comprimida

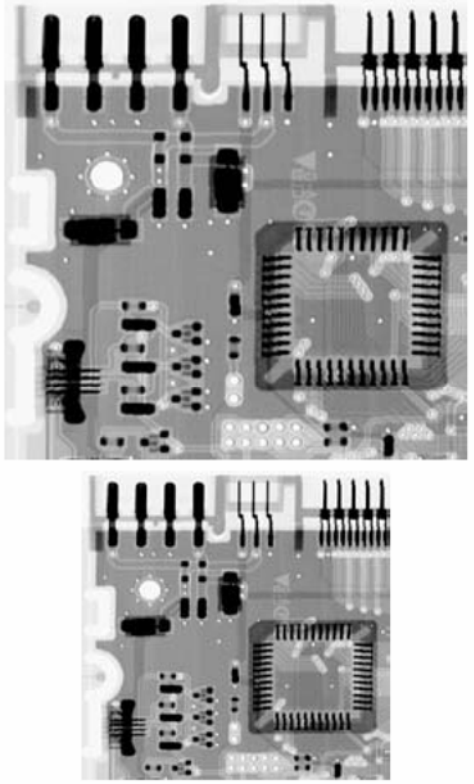
Root Mean Square Error (Métrica no perceptual)

$$RMSE = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left[\tilde{A}(i,j) - A(i,j)\right]^2}$$

Cambio de resolución dpi

dpi: dots per inches

X es una imagen en formato jpg de 450 x 450 pixeles, con resolución 200 dpi, resulta en una imagen con dimensiones de 2.25 x 2.25 inches (pulgadas). Manteniendo el número de pixeles pero ahora con resolución de 300 dpi, resulta en una imagen con dimensiones 1.5 x 1.5 inches.



a
b

FIGURE 2.5 Effects of changing the dpi resolution while keeping the number of pixels constant. (a) A 450 × 450 image at 200 dpi (size = 2.25 × 2.25 inches). (b) The same 450 × 450 image, but at 300 dpi (size = 1.5 × 1.5 inches). (Original image courtesy of Lixi, Inc.)

```
>> imwrite(X,'Xsc.tif','compression', 'none','resolution', [300,300])
```

Aplicable solo para imágenes tif

Clases de datos

Name	Description
`double`	Double-precision, floating-point numbers in the approximate range -10^{308} to 10^{308} (8 bytes per element).
`uint8`	Unsigned 8-bit integers in the range [0, 255] (1 byte per element).
`uint16`	Unsigned 16-bit integers in the range [0, 65535] (2 bytes per element).
`uint32`	Unsigned 32-bit integers in the range [0, 4294967295] (4 bytes per element).
`int8`	Signed 8-bit integers in the range [−128, 127] (1 byte per element).
`int16`	Signed 16-bit integers in the range [−32768, 32767] (2 bytes per element).
`int32`	Signed 32-bit integers in the range [−2147483648, 2147483647] (4 bytes per element).
`single`	Single-precision floating-point numbers with values in the approximate range -10^{38} to 10^{38} (4 bytes per element).
`char`	Characters (2 bytes per element).
`logical`	Values are 0 or 1 (1 byte per element).

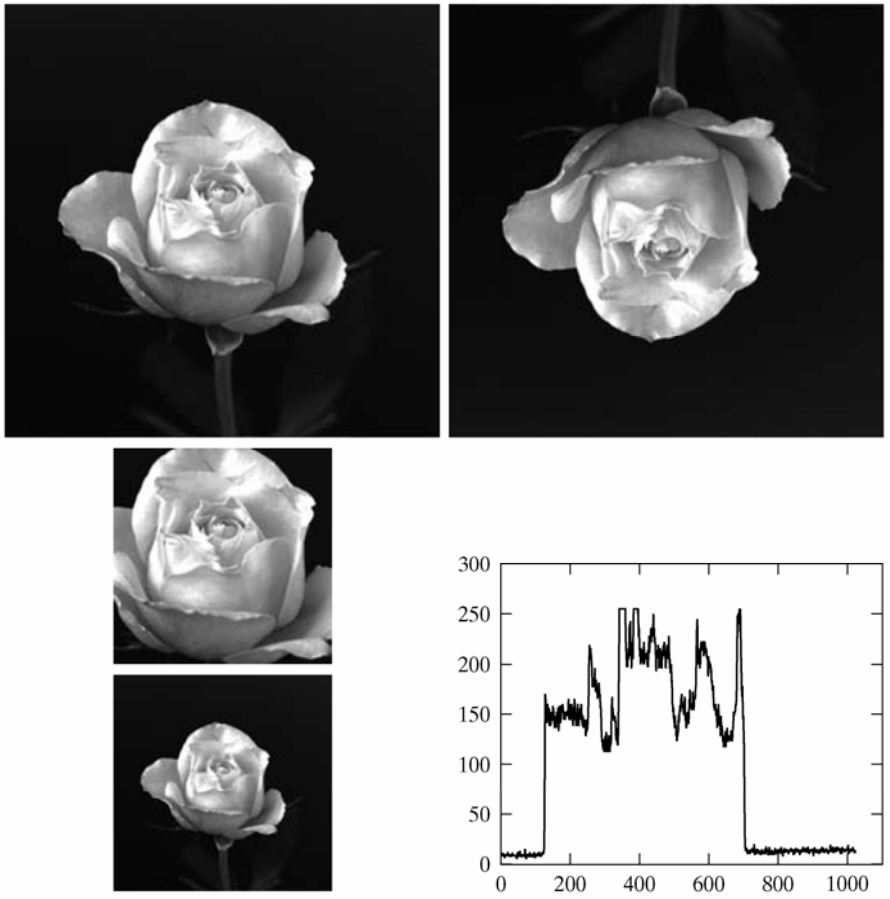
TABLE 2.2
Data classes. The first eight entries are referred to as *numeric* classes; the ninth entry is the *character* class, and the last entry is of class *logical*.

Conversión entre clases y tipos de imágenes

Name	Converts Input to:	Valid Input Image Data Classes
`im2uint8`	`uint8`	`logical`, `uint8`, `uint16`, and `double`
`im2uint16`	`uint16`	`logical`, `uint8`, `uint16`, and `double`
`mat2gray`	`double` (in range [0, 1])	`double`
`im2double`	`double`	`logical`, `uint8`, `uint16`, and `double`
`im2bw`	`logical`	`uint8`, `uint16`, and `double`

TABLE 2.3 Functions in IPT for converting between image classes and types. See Table 6.3 for conversions that apply specifically to color images.

Transformación de Imágenes mediante Indexado de arreglos



a b
c
d e

FIGURE 2.6
Results obtained using array indexing.
(a) Original image. (b) Image flipped vertically.
(c) Cropped image.
(d) Subsampled image. (e) A horizontal scan line through the middle of the image in (a).

Operator	Name	MATLAB Function	Comments and Examples
+	Array and matrix addition	plus(A, B)	a + b, A + B, or a + A.
–	Array and matrix subtraction	minus(A, B)	a – b, A – B, A – a, or a – A.
.*	Array multiplication	times(A, B)	C = A.*B, C(I, J) = A(I, J)*B(I, J).
*	Matrix multiplication	mtimes(A, B)	A*B, standard matrix multiplication, or a*A, multiplication of a scalar times all elements of A.
./	Array right division	rdivide(A, B)	C = A./B, C(I, J) = A(I, J)/B(I, J).
.\	Array left division	ldivide(A, B)	C = A.\B, C(I, J) = B(I, J)/A(I, J).
/	Matrix right division	mrdivide(A, B)	A/B is roughly the same as A*inv(B), depending on computational accuracy.
\	Matrix left division	mldivide(A, B)	A\B is roughly the same as inv(A)*B, depending on computational accuracy.
.^	Array power	power(A, B)	If C = A.^B, then C(I, J) = A(I, J)^B(I, J).
^	Matrix power	mpower(A, B)	See online help for a discussion of this operator.
.'	Vector and matrix transpose	transpose(A)	A.'. Standard vector and matrix transpose.
'	Vector and matrix complex conjugate transpose	ctranspose(A)	A'. Standard vector and matrix conjugate transpose. When A is real A.' = A'.
+	Unary plus	uplus (A)	+A is the same as 0 + A.
–	Unary minus	uminus (A)	–A is thc samc as 0 – A or –1*A.
:	Colon		Discussed in Section 2.8.

TABLE 2.4
Array and matrix arithmetic operators. Computations involving these operators can be implemented using the operators themselves, as in A + B, or using the MATLAB functions shown, as in plus (A, B). The examples shown for arrays use matrices to simplify the notation, but they are easily extendable to higher dimensions.

Operaciones aritméticas con imágenes

Function	Description
`imadd`	Adds two images; or adds a constant to an image.
`imsubtract`	Subtracts two images; or subtracts a constant from an image.
`immultiply`	Multiplies two images, where the multiplication is carried out between pairs of corresponding image elements; or multiplies a constant times an image.
`imdivide`	Divides two images, where the division is carried out between pairs of corresponding image elements; or divides an image by a constant.
`imabsdiff`	Computes the absolute difference between two images.
`imcomplement`	Complements an image. See Section 3.2.1.
`imlincomb`	Computes a linear combination of two or more images. See Section 5.3.1 for an example.

TABLE 2.5
The image arithmetic functions supported by IPT.

Operadores relacionales y lógicos

Operator	Name
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
==	Equal to
~=	Not equal to

TABLE 2.6
Relational operators.

Operator	Name
&	AND
|	OR
~	NOT

TABLE 2.7
Logical operators.

Funciones lógicas

Function	Comments
`xor` (exclusive OR)	The `xor` function returns a 1 only if both operands are logically different; otherwise `xor` returns a 0.
`all`	The `all` function returns a 1 if all the elements in a vector are nonzero; otherwise `all` returns a 0. This function operates columnwise on matrices.
`any`	The `any` function returns a 1 if any of the elements in a vector is nonzero; otherwise `any` returns a 0. This function operates columnwise on matrices.

TABLE 2.8
Logical functions.

Funciones lógicas

Function	Description
`iscell(C)`	True if `C` is a cell array.
`iscellstr(s)`	True if `s` is a cell array of strings.
`ischar(s)`	True if `s` is a character string.
`isempty(A)`	True if `A` is the empty array, [ ].
`isequal(A, B)`	True if `A` and `B` have identical elements and dimensions.
`isfield(S, 'name')`	True if `'name'` is a field of structure `S`.
`isfinite(A)`	True in the locations of array `A` that are finite.
`isinf(A)`	True in the locations of array `A` that are infinite.
`isletter(A)`	True in the locations of `A` that are letters of the alphabet.
`islogical(A)`	True if `A` is a logical array.
`ismember(A, B)`	True in locations where elements of `A` are also in `B`.
`isnan(A)`	True in the locations of `A` that are `NaN`s (see Table 2.10 for a definition of `NaN`).
`isnumeric(A)`	True if `A` is a numeric array.
`isprime(A)`	True in locations of `A` that are prime numbers.
`isreal(A)`	True if the elements of `A` have no imaginary parts.
`isspace(A)`	True at locations where the elements of `A` are whitespace characters.
`issparse(A)`	True if `A` is a sparse matrix.
`isstruct(S)`	True if `S` is a structure.

TABLE 2.9
Some functions that return a logical 1 or a logical 0 depending on whether the value or condition in their arguments are `true` or `false`. See online help for a complete list.

Algunas variables y constantes

Function	Value Returned
`ans`	Most recent answer (variable). If no output variable is assigned to an expression, MATLAB automatically stores the result in `ans`.
`eps`	Floating-point relative accuracy. This is the distance between 1.0 and the next largest number representable using double-precision floating point.
`i` (or `j`)	Imaginary unit, as in `1 + 2i`.
`NaN` or `nan`	Stands for Not-a-Number (e.g., `0/0`).
`pi`	3.14159265358979
`realmax`	The largest floating-point number that your computer can represent.
`realmin`	The smallest floating-point number that your computer can represent.
`computer`	Your computer type.
`version`	MATLAB version string.

TABLE 2.10 Some important variables and constants.

Control de flujo

Statement	Description
`if`	`if`, together with `else` and `elseif`, executes a group of statements based on a specified logical condition.
`for`	Executes a group of statements a fixed (specified) number of times.
`while`	Executes a group of statements an indefinite number of times, based on a specified logical condition.
`break`	Terminates execution of a `for` or `while` loop.
`continue`	Passes control to the next iteration of a `for` or `while` loop, skipping any remaining statements in the body of the loop.
`switch`	`switch`, together with `case` and `otherwise`, executes different groups of statements, depending on a specified value or string.
`return`	Causes execution to return to the invoking function.
`try...catch`	Changes flow control if an error is detected during execution.

TABLE 2.11 Flow control statements.

Visualización de funciones de 2 variables

Función sinusoidal en 2D

$$f(x,y) = A\sin\left(u_0 x + v_0 y\right)$$

$$x = 0,1,2,\cdots,M-1$$

$$y = 0,1,2,\cdots,N-1$$

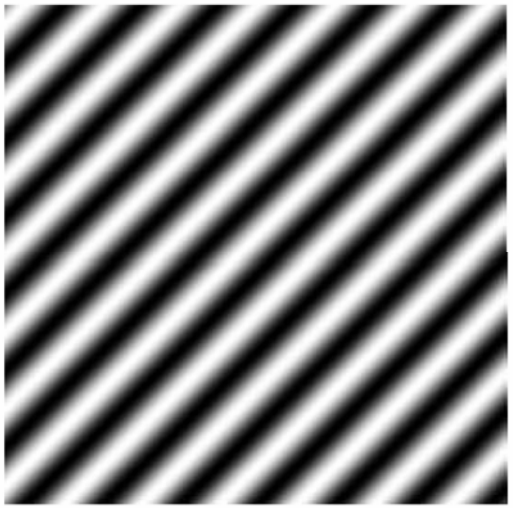


FIGURE 2.7 Sinusoidal image generated in Example 2.13.

Ejemplo 2.13: Implementar usando lazos `for` anidados y el comando `meshgrid.` Comparar tiempos de cómputo usando comandos `tic y toc.`