Procesamiento digital de señales

FACULTAD DE INGENIERIA

Escuela de ingeniería en Telecomunicaciones

Cuarto Semestre

Unidad IV: INTRODUCCIÓN PROCESAMIENTO DE IMÁGENES

PhD. Daniel Antonio Santillán Haro

Resumen

Resumen

Imagen Digital

Una imagen es una función bidimensional f(x,y), donde x e y representan las coordenadas espaciales y el valor de f en cualquier par de coordenadas (x,y) representa la intensidad de la imagen en dicho punto.

- Una imagen digital f[x,y] descrita en un espacio 2D discreto se deriva de una imagen análoga f(x,y) en un espacio 2D continuo a través de un proceso llamado digitalización.
- Al digitalizarla, la imagen continua en 2D es dividida en M filas y N columnas
- La intersección de una fila y una columna se llama pixel.
- Es común que una imagen contenga sub-imagenesllamadas regiones de interés (ROI, regionsof interest).

Fuente: BURGER, Wilhelm, et al. Principles of digital image processing. London: Springer, 2009.

- Imagen digital.

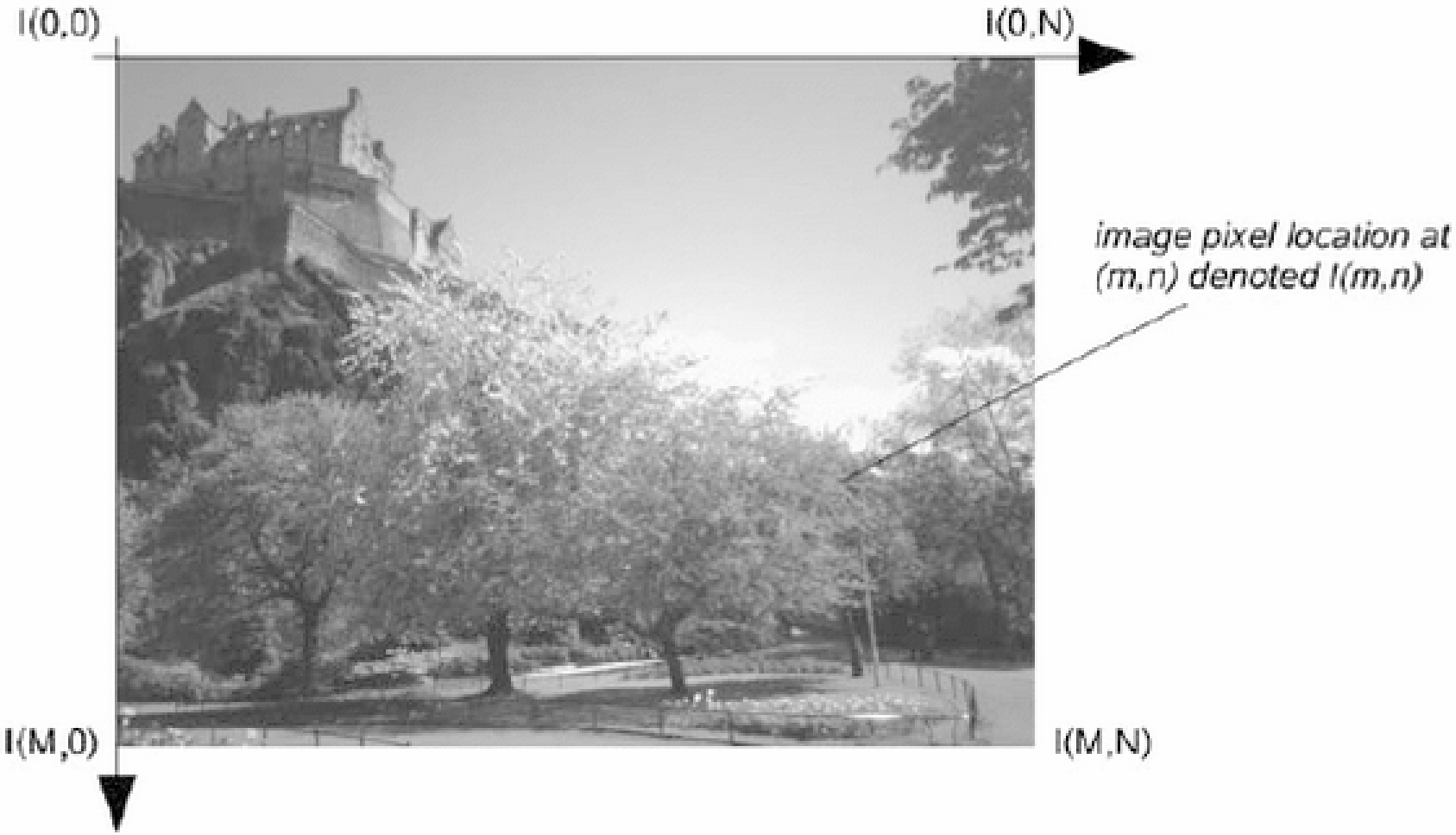


Fig The 2-D Cartesian coordinate space of an M x N digital image

Tipos de datos en una imagen

El tipo de dato que contendrá una imagen puede ser de varios tipos (según el tipo de dato de cada pixel):

double: Doble precisión, números en punto flotante que varían en un rango aproximado de -10308 a 10308 (8 bytes por elemento)

- **uint8**: Enteros de 8 bits en el rango de [0,255] (1 byte por elemento)
- **uint16**: Enteros de 16 bits en el rango de [0, 65535] (2 bytes por elemento)
- **uint32**: Enteros de 32 bits en el rango de [0, 4294967295] (4 bytes por elemento)
- **int8**: Enteros de 8 bits en el rango de [-128, 127] (1 byte por elemento)
- **int16**: Enteros de 16 bits en el rango de [-32768, 32767] (2 bytes por elemento)

int32: Enteros de 32 bits en el rango de [-2147483648,2147483647] (4 bytes por elemento)

- **logical**: Los valores son 0 ó 1 (1 bit por elemento)

Fuente: SOLOMON, Chris; BRECKON, Toby. Fundamentals of Digital Image Processing: A practical approach with examples in Matlab. John Wiley & Sons, 2011.

Tipos de imagen

Imagen de intensidad es una matriz de datos cuyos valores han sido escalados para que representen intensidades de una escala de grises. Cuando los elementos de una imagen de intensidad son de clase uint8 (enteros almacenados en 8 bits) o de clase uint16 (enteros almacenados en 16 bits), pueden almacenar, respectivamente, 2^8=256 valores en el rango [0:255] o 2^{16}=65536 valores en el rango[0:65535]. Si la imagen es de clase double, los valores son números en punto flotante (que se almacenan en 32 bits). En este último caso, los valores se toman en el rango de [0:1] por convención.

La imagen binaria es una imagen en blanco y negro. Cada pixel tiene asignado un valor lógico de 0 ó 1 donde 0 representa el negro y 1 el blanco

La imagen en color es como la imagen de intensidad pero tiene **tres canales**, es decir, a cada pixel le corresponden tres valores de intensidad (RGB) en lugar de uno.

Fuente: SOLOMON, Chris; BRECKON, Toby. Fundamentals of Digital Image Processing: A practical approach with examples in Matlab. John Wiley & Sons, 2011.

Adquisición de imágenes

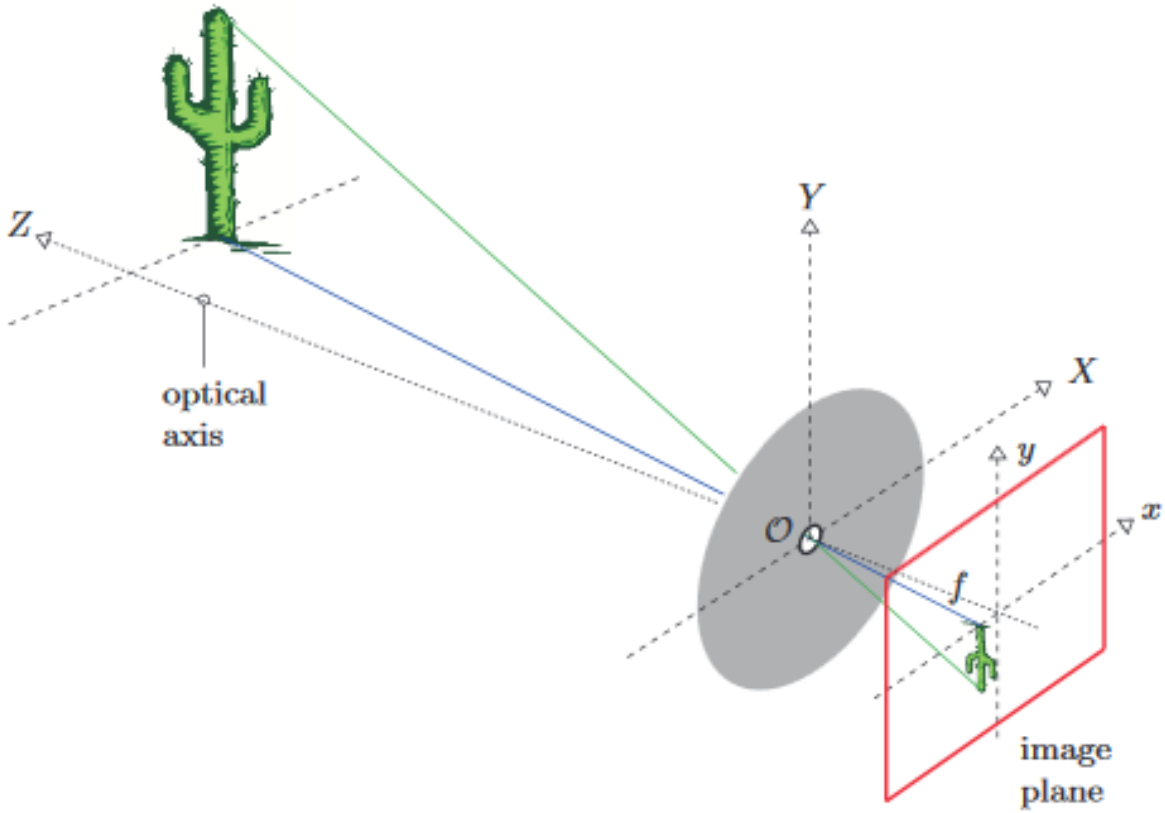


Figure Geometry of the pinhole camera. The pinhole opening serves as the origin ($\mathcal{O}$) of the three-dimensional coordinate system (X, Y, Z) for the objects in the scene. The optical axis, which runs through the opening, is the Z axis of this coordinate system. A separate two-dimensional coordinate system (x, y) describes the projection points on the image plane. The distance f ("focal length") between the opening and the image plane determines the scale of the projection.

BURGER, Wilhelm, et al. Principles of digital image processing. London: Springer, 2009.

Modelo lente delgada

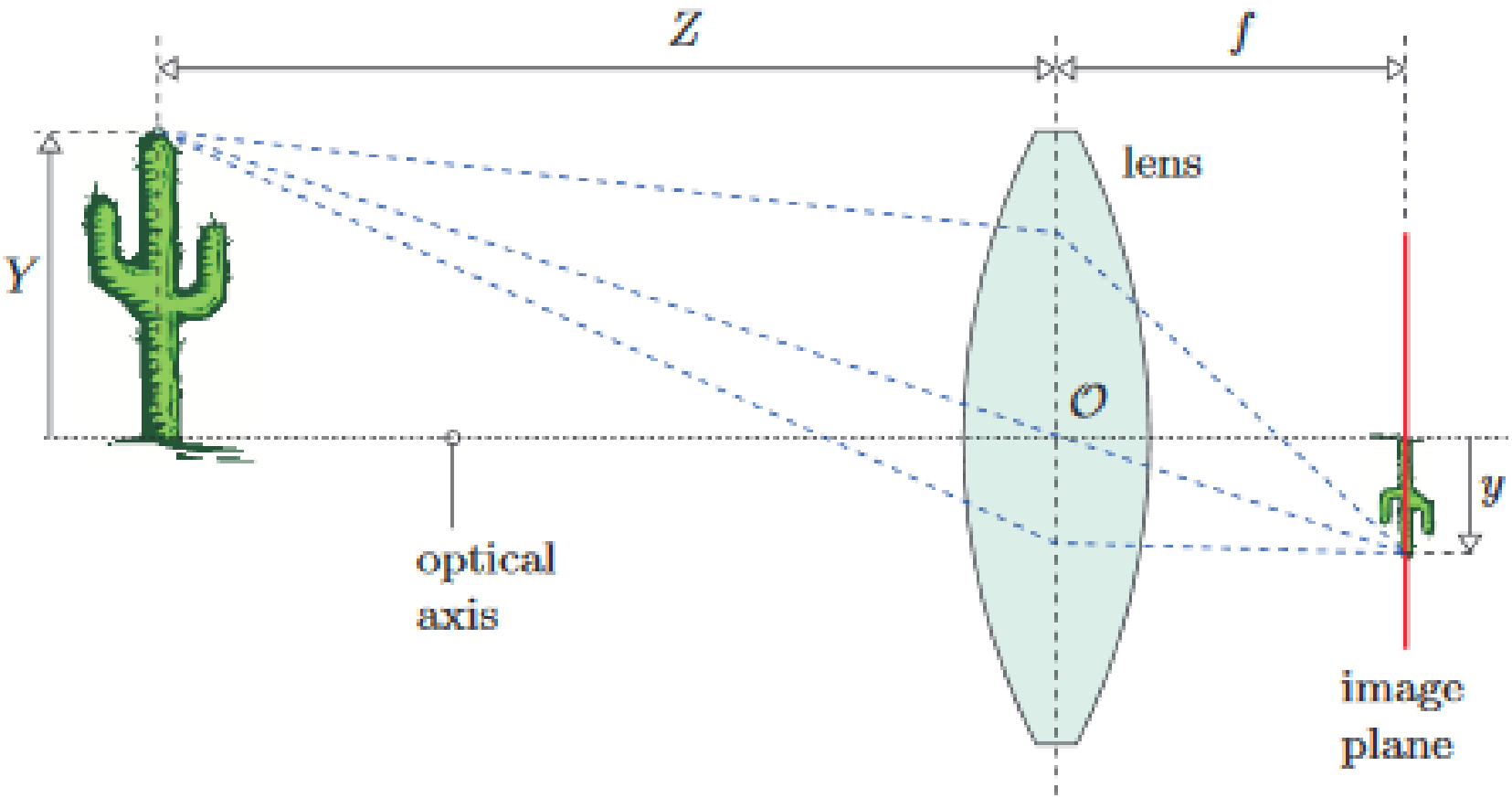


Figure The thin lens model.

BURGER, Wilhelm, et al. Principles of digital image processing. London: Springer, 2009.

Conversión de un tipo a otro

Cuando realizamos transformaciones matemáticas de imágenes, a menudo necesitamos que la imagen sea de tipo double. Pero cuando la leemos y almacenamos ahorramos espacio usando codificación entera. Podemos usar las siguientes **funciones**:

- **im2uint8**: de cualquier tipo a uint8,
- **im2double**: de cualquier tipo a double,
- **im2bw**: de cualquier tipo a logical,
- **rgb2gray**: RGB color a gris, o sea de tres capas a una sola.

Ejemplo: I es de tipo entero y quiero convertirla a double

```
>>D= im2double(I)
```

Nota: Se pueden utilizar también las funciones de conversión de tipo, pero el resultado es diferente

BURGER, Wilhelm, et al. Principles of digital image processing. London: Springer, 2009.

Resumen

Lectura, visualización y escritura de imágenes en Matlab

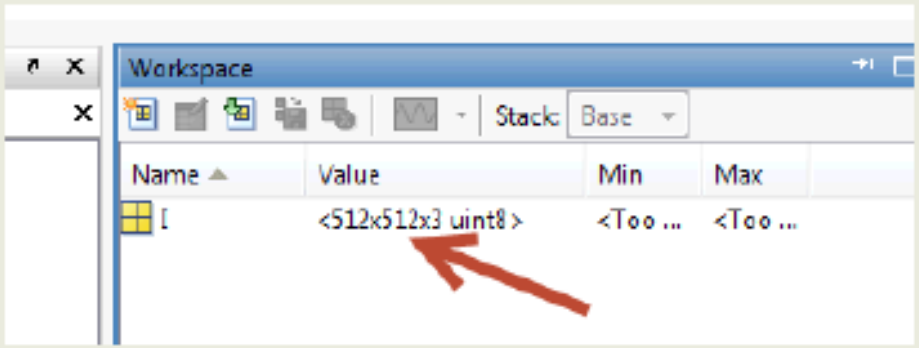
- Para leer una imagen se usa el comando **imread:**

I= imread('nombre_archivo')

Ejemplo:

I=imread('lena_std.tif');

Si observamos el workspace, vemos la variable I definida como una matriz de tres dimensiones:



Lectura, visualización y escritura de imágenes en Matlab

Para visualizar una imagen se usa el comando **imshow:**

```
imshow(I)
```

Donde I es la matriz leída con el comando anterior.

Ojo!!! Ya no utilizamos el nombre del archivo. A partir de ahora nos manejamos con la matriz donde leímos la imagen.

Lectura, visualización y escritura de imágenes en Matlab

- Como vimos, la variable I tiene 3 dimensiones. Esto es porque la imagen tomada con el comando **imread** posee 3 canales. Podemos manipular cada canal por separado:

  J= I(:, : , 1); Esto significa **"Dame todas las filas y todas las columnas del canal 1 y guardalo en J"**

  Cuando vamos a visualizar J en el espacio de trabajo, vemos que tiene 2 dimensiones. Esto es porque extrajimos una capa o canal.

```
>> I=imread('lena_std.tif');
>> subplot(2,2,1)
>> imshow(I)
>> subplot(2,2,2)
>> imshow( I(:,:,1) )
>> subplot(2,2,3)
>> imshow( I(:,:,2) )
>> subplot(2,2,4)
>> imshow( I(:,:,3) )
```

- ¿Por que cuando hacemos

  >> J= I(:, : , 1)

  >>imshow(**I(:,:,1)**)

  la imagen se ve en blanco y negro?
- Simplemente porque tomamos un solo canal de los 3 que tiene la imagen
- Si vemos cada canal por separado lo que se visualiza son las intensidades en escala de gris de cada canal (rojo, verde y azul)

- ¿Como haríamos para ver cada canal en su color?
- Lo que debemos hacer es “anular” los otros canales asignandoles el valor 0.
- Veamos a Lena solo en el canal rojo:

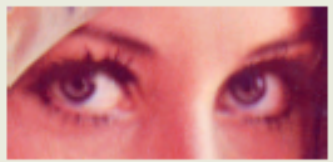
```
>> T = I; %hago copia de la imagen
>> T(:,:,2) = 0; %Capa verde en cero
>> T(:,:,3) = 0; %Capa azul en cero
>> imshow(T)
```

- Una vez procesada la imagen, podemos guardarla en otro archivo. Supongamos que queremos guardar un trozo de la figura de Lena. Tenemos que usar el comando:

imwrite(I , 'nombre_archivo')

Ejemplo:

```
>> G = I(238:293 , 241:357 , :);  % Extraemos una subimagen
>> imshow(G)
>> imwrite(G , 'trozoLena.png')
```



Como vimos antes hay muchas opciones una vez que tenemos la imagen en el workspace:

- Hemos visto que podemos extraer una capa
- Hemos visto que podemos extraer una subimagen
- Hemos visto que podemos anular (poner en cero) uno o dos canales

Podemos generar una imagen con Matlab. Para eso precisamos una matriz:

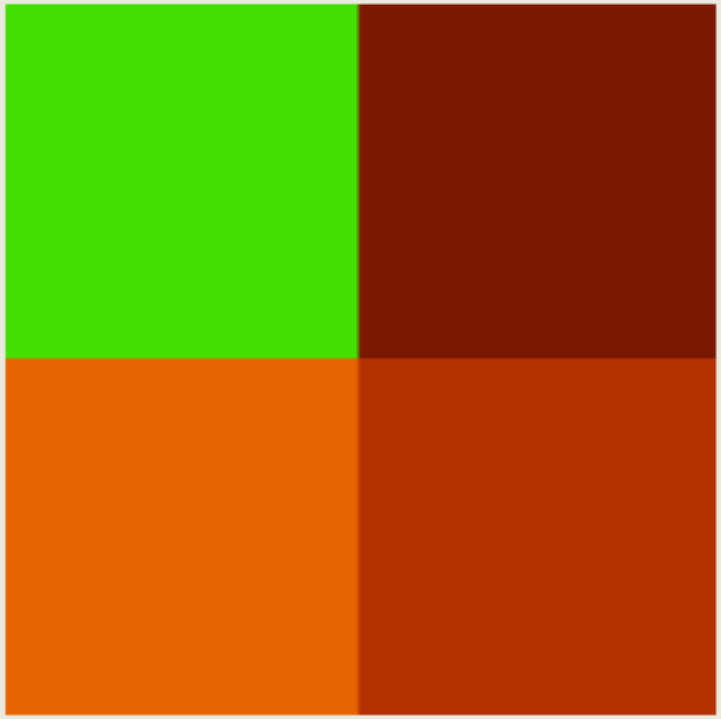
```
>> h= zeros( [500,500,3], 'uint8');
```

Ahora tenemos que darle color. Para eso vamos a manipular pequeñas subimagenes y coloreamos:

```
>> h(1:250, 1:250, 1)=70;
>> h(1:250, 251:end, 1)=120;
>> h(251:end, 1:250, 1)=230;
>> h(251:end, 251:end, 1)=180;
>> imshow(h)
```

Vamos a modificar el resto de los canales con otros colores:

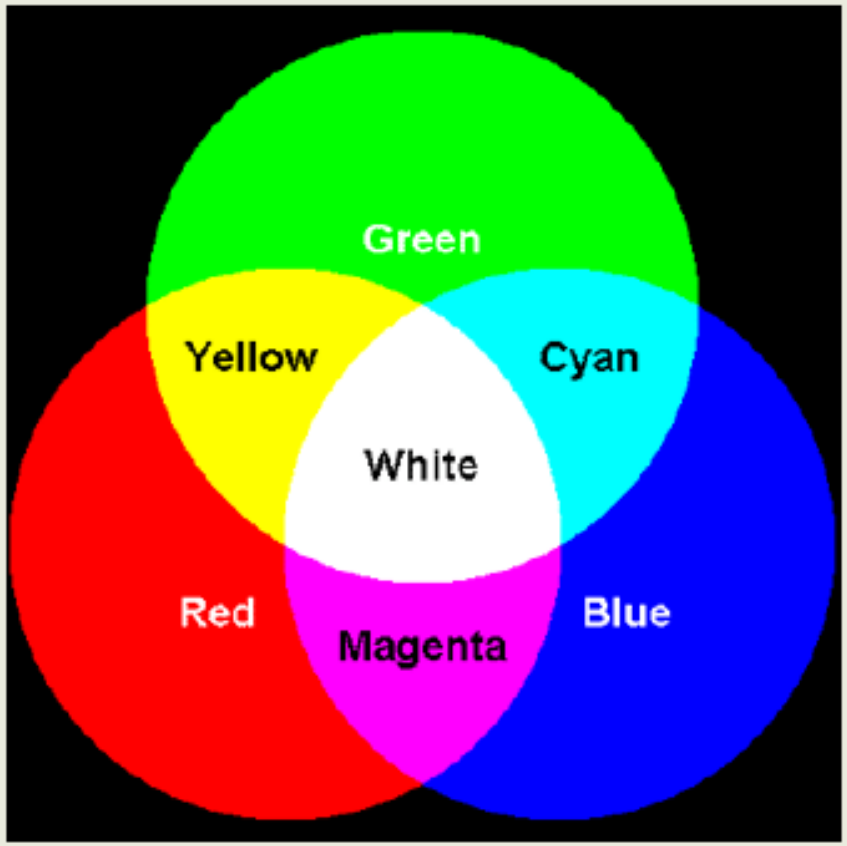
```
>> h(1:250, 1:250, 2)=220;
>> h(1:250, 251:end, 2)=25;
>> h(251:end, 1:250, 2)=100;
>> h(251:end, 251:end, 2)=50;
>> imshow(h)
```



Vamos a modificar el último canal:

```
>> h(1:250,1:250,3)=150;
>> h(1:250,251:end,3)=235;
>> h(251:end,1:250,3)=22;
>> h(251:end,251:end,3)=197;
>> imshow(h)
```

Combinación de colores



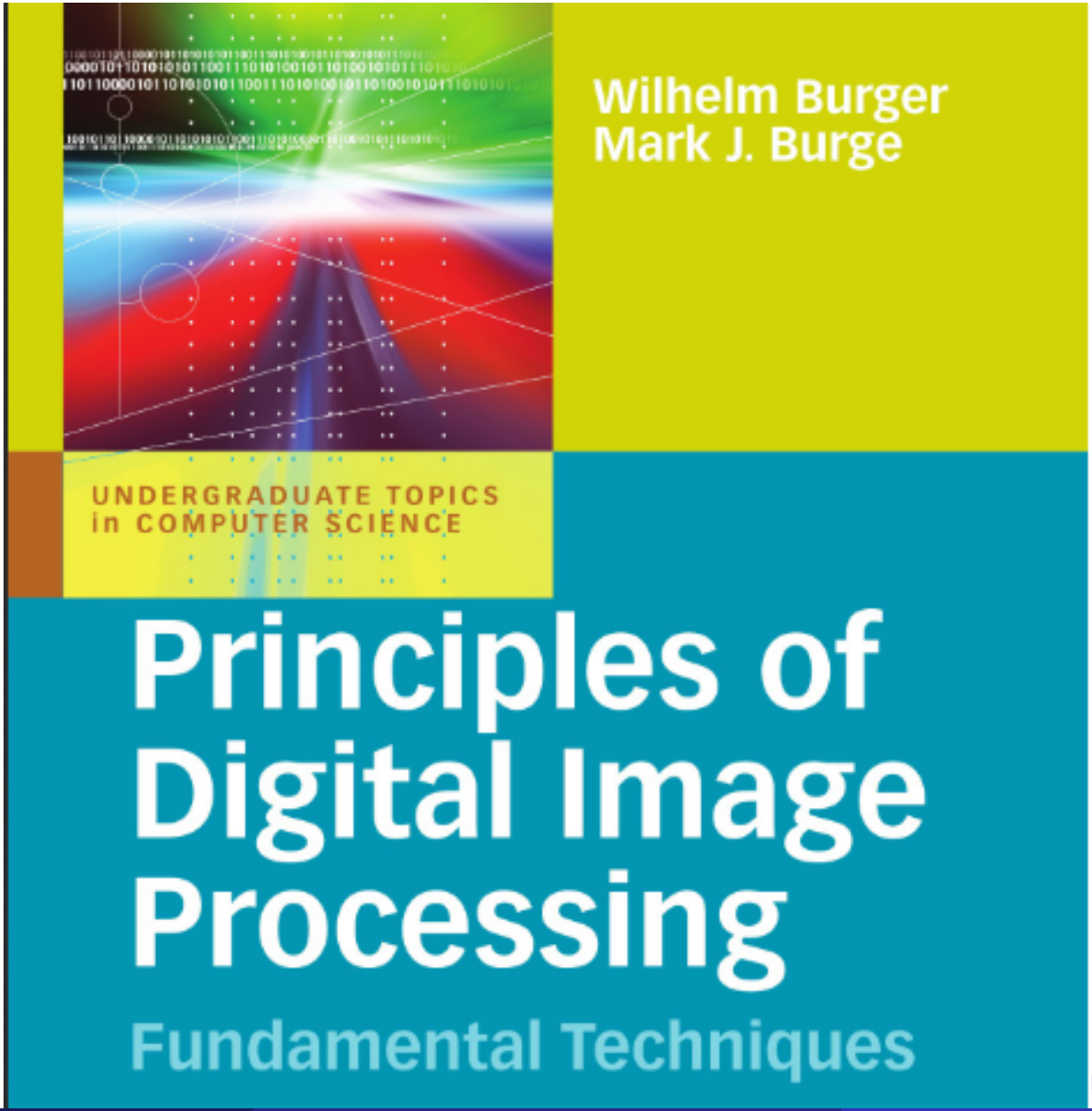
1. The continuous light distribution must be *spatially sampled*.
2. This resulting "discrete" function must then be *sampled in the time domain* to create a single (still) image.
3. Finally, the resulting values must be *quantized* to a finite set of numeric values so that they are representable within the computer.

Step 1: Spatial sampling

The spatial sampling of an image (that is, the conversion of the continuous signal to its discrete representation) depends on the geometry of the sensor elements of the acquisition device (e. g., a digital or video camera). The individual sensor elements are usually arranged as a rectangular array on the sensor plane (Fig. 1.4). Other types of image sensors, which include hexagonal elements and circular sensor structures, can be found in specialized camera products.

Step 2: Temporal sampling

Temporal sampling is carried out by measuring at regular intervals the amount of light incident on each individual sensor element. The CCD[2] or CMOS[3] sensor in a digital camera does this by triggering an electrical charging process,


Wilhelm Burger
Mark J. Burge
UNDERGRADUATE TOPICS
in COMPUTER SCIENCE
Principles of
Digital Image
Processing
Fundamental Techniques

Procesamiento digital de señales

FACULTAD DE INGENIERIA

Escuela de ingeniería en Telecomunicaciones

Cuarto Semestre

Unidad IV: INTRODUCCIÓN PROCESAMIENTO DE IMÁGENES

PhD. Daniel Antonio Santillán Haro