Procesamiento de imágenes

ALMA NAYELI RODRÍGUEZ VÁZQUEZ

PLANOS DE COLOR RGB

UNIVERSIDAD PANAMERICANA

Objetivo

- Imagen digital
- Formato de color RGB
- Imagen en escala de grises
- Imagen binaria
- La clase Mat
- Funciones de entrada y salida
- Extracción de planos de color
- Conversión de RGB a grises

Imagen digital

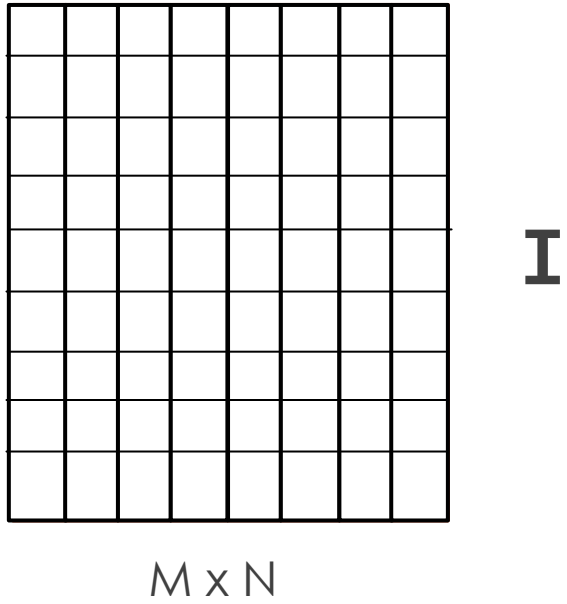
UNIVERSIDAD PANAMERICANA

Una imagen digital es definida como una función bidimensional discreta $I(x,y)$ que denota a una matriz de tamaño M por N cuyos valores discretos representan su intensidad y el par de coordenadas espaciales (x,y) representa la posición discreta de un pixel presente en la imagen.

Imagen digital

UNIVERSIDAD PANAMERICANA

$$I(x,y) = \begin{bmatrix} I(1,1) & I(2,1) & \dots & I(N,1) \\ I(1,2) & I(2,2) & \dots & I(N,2) \\ \vdots & \vdots & \ddots & \vdots \\ I(1,M) & I(2,M) & \dots & I(N,M) \end{bmatrix}$$

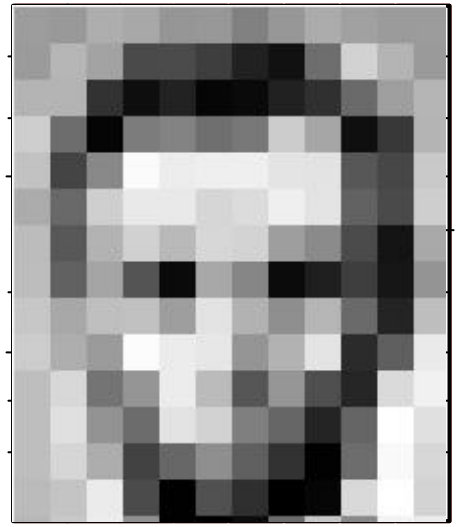


I

M x N

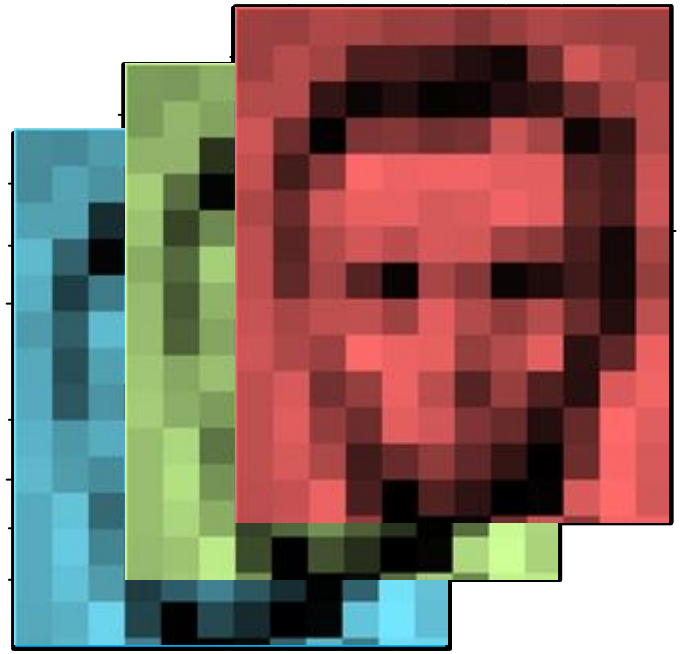
Formato de color RGB

UNIVERSIDAD PANAMERICANA



I

M x N



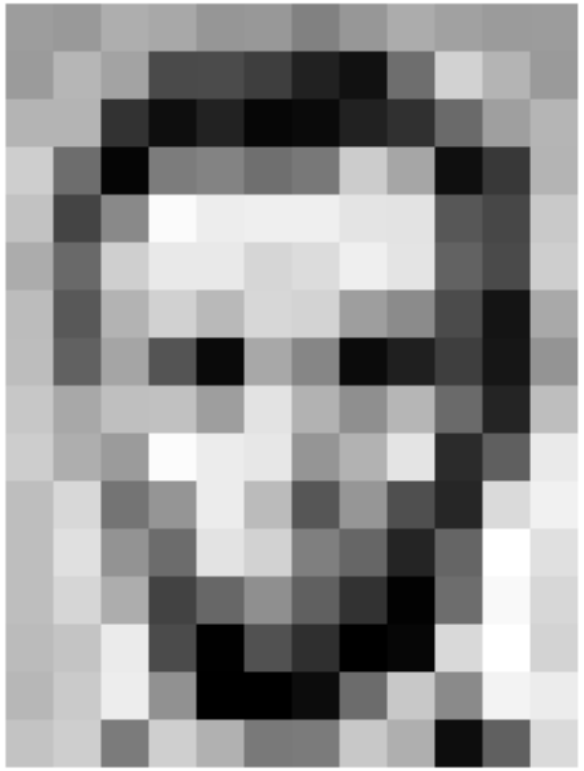
I

M x N

RGB → BGR

Imagen en escala de grises

UNIVERSIDAD PANAMERICANA



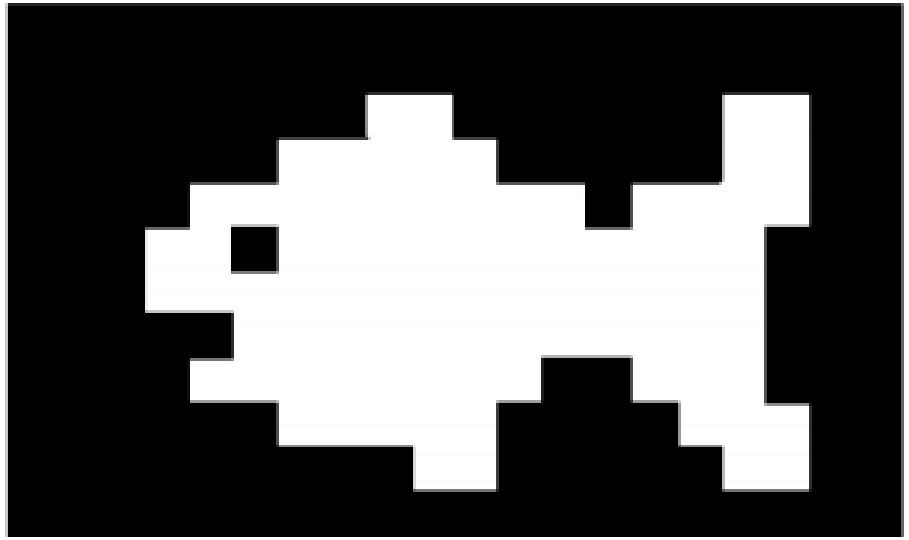
157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

- Formato
  - 8UC (2^8 bits)
  - 256 niveles
  - Imagen de 8 bits

Imagen binaria

UNIVERSIDAD PANAMERICANA



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

La clase cv::Mat

UNIVERSIDAD PANAMERICANA

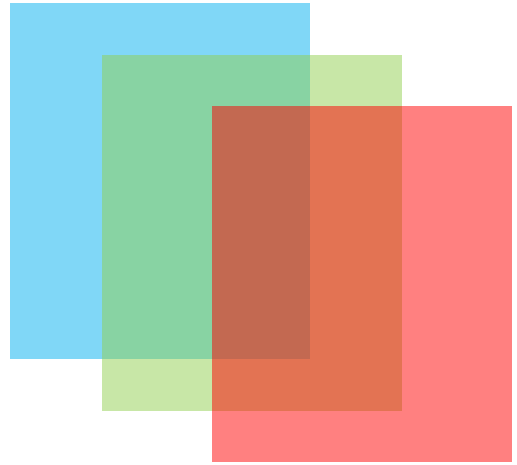
Constructor

- Sintaxis:

  Mat objeto (int filas, int columnas, int tipo);

- Ejemplo:

```
Mat imagen_gris(200,100,CV_8UC1);
Mat imagen_RGB(200,100,CV_8UC3);
Mat imagen_gris(otra_imagen.rows,otra_imagen.cols,CV_8UC1);
Mat imagen_RGB(otra_imagen.rows,otra_imagen.cols,CV_8UC3);
```



Funciones de entrada y salida

UNIVERSIDAD PANAMERICANA

- Lectura de una imagen desde archivo

```
Mat imread(const String &filename, int flags =1)

enum { IMREAD_GRAYSCALE = 0, // 8 bit gray
       IMREAD_COLOR = 1 // valor por defecto}
```

- Ejemplo:

```
Mat imagen = imread("miImagen.jpg");
Mat imagen = imread("miImagen.jpg",IMREAD_GRAYSCALE);
```

Funciones de entrada y salida

Despliegue de una imagen en pantalla

- Sintaxis

```
void imshow(const String &winname, InputArray Mat)
```

- Ejemplo:

```
Mat imagen = imread(“miImagen.jpg”);
imshow(“imagen a color”, imagen);
```

El método ptr<>

Este método permite acceder a las filas de un objeto de la clase Mat

- Sintaxis:

```
tipo* objetoMat.ptr<tipo>(int nfila);
```

- Ejemplo:

```
imagen.ptr<Vec3b>(0);
imagen.ptr<uchar>(0);
```

Los tipos Vec3b, uchar

UNIVERSIDAD PANAMERICANA

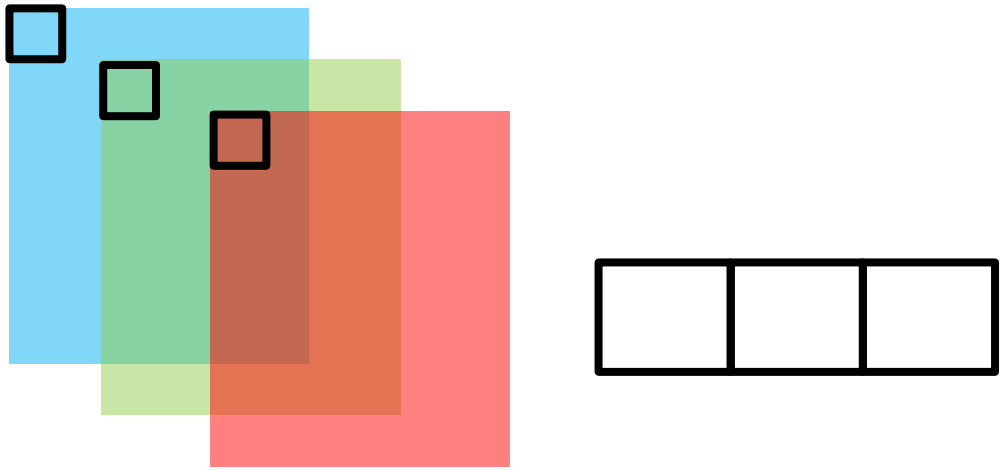
Vec3b

- Sintaxis:

```
Vec3b* fila;
```

- Ejemplo:

```
Vec3b* fila;
fila = imagen.ptr<Vec3b>(0);
```



Uchar

- Sintaxis:

```
uchar* fila;
```

- Ejemplo:

```
uchar* fila;
fila = imagen.ptr<uchar>(0);
```

La función waitkey()

Esta función espera a que una tecla sea presionada. El tiempo de retardo se especifica entre paréntesis en milisegundos. Por lo que espera a que cualquiera de los dos eventos suceda, es decir, que se presione una tecla, o que se termine el tiempo de espera. Si el valor del parámetro es cero, la función espera indefinidamente.

- Sintaxis:

```
waitkey(int delay = 0);
```

- Ejemplo:

```
waitkey(0); //espera indefinidamente
waitkey(3000); // espera 3 segundos
```

Práctica 1:
Extracción de planos de color

Métodos de conversión de RGB a escala de grises

UNIVERSIDAD PANAMERICANA

- Método del promedio

```
Gray = (Red + Green + Blue) / 3
```

- Luma
  - Photoshop/openCV `Gray = (Red * 0.3 + Green * 0.59 + Blue * 0.11)`
  - BT.709 `Gray = (Red * 0.2126 + Green * 0.7152 + Blue * 0.0722)`
  - BT.601/Matlab `Gray = (Red * 0.299 + Green * 0.587 + Blue * 0.114)`

Métodos de conversión de RGB a escala de grises

UNIVERSIDAD PANAMERICANA

- Desaturación (suaves grises, menos contraste)

```
Gray = ( Max(Red, Green, Blue) + Min(Red, Green, Blue) ) / 2
```

- Descomposición
  - Mínima descomposición (menos brillo)

```
Gray = Min(Red, Green, Blue)
```

  - Máxima descomposición (más brillo)

```
Gray = Max(Red, Green, Blue)
```

Práctica 2:
Métodos de conversión de RGB a gris