Image Formation

Computer Vision (CS0029)

Type of Images

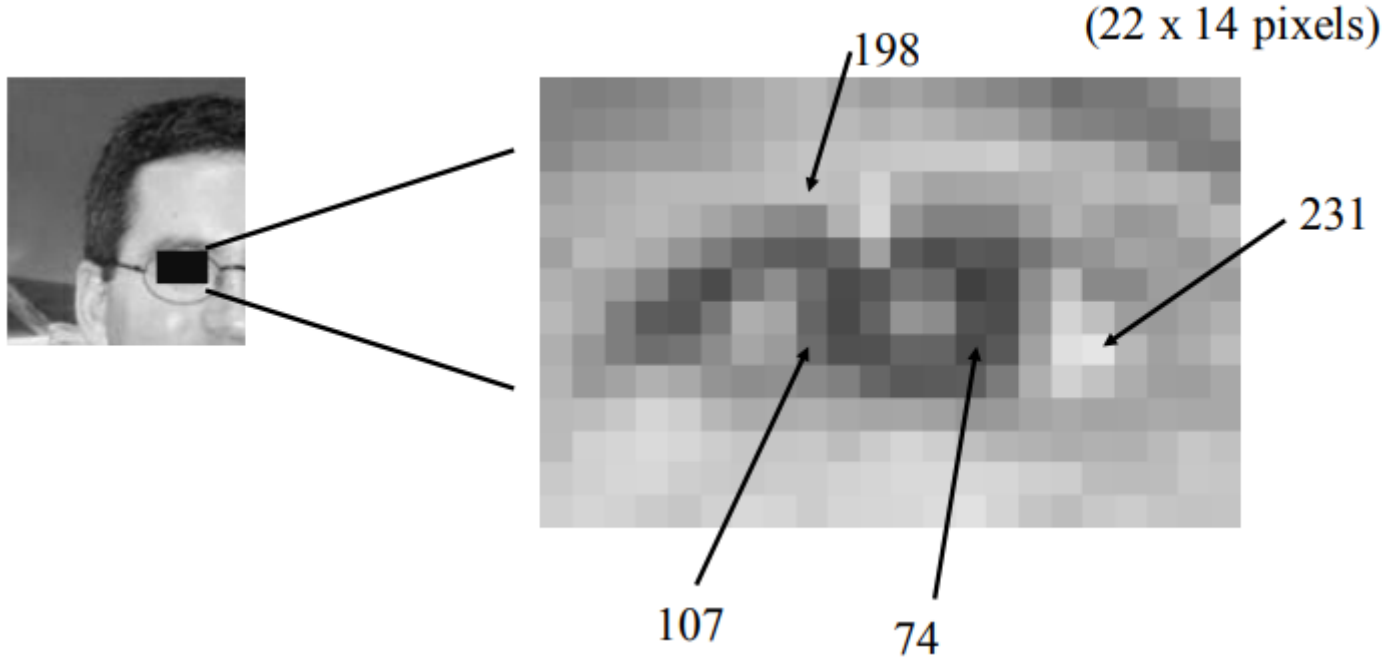
- Binary image
  - Simplest image type
  - Digital image with all pixel values 0 or 1
    - Usually, 0 is black and 1 is white in display
  - Useful to studying object shape

	0	1	2	3
0	1	1	0	0
1	1	1	0	0
2	1	1	1	1

Type of Images

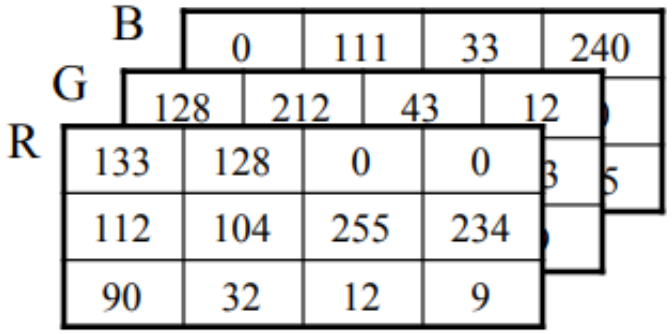
- Grayscale image
  - Monochrome digital image with one intensity value per pixel
  - Grayscale value for 8-bit are 0-255
    - 0: darkest, 255-brightest

	0	1	2	3
0	133	128	0	0
1	112	104	255	234
2	90	32	12	9



Type of Images

- RGB image
  - Color digital image with three intensity values/bytes per piexel (24bit color)
    - Red, Green, Blue
  - Some image formats use 8-bit colormap



Digital Image formats

- Too many image format in use
  - PPM, PNG, JPG, BMP, …….
- Two components in a digital image file
- Image file header
  - Info on image dimension, type, date of creation etc.
- Pixel data
  - Stream of data in raster order (row-by-row)
  - Bytes, ASCII(decimal)
- May have compressed image

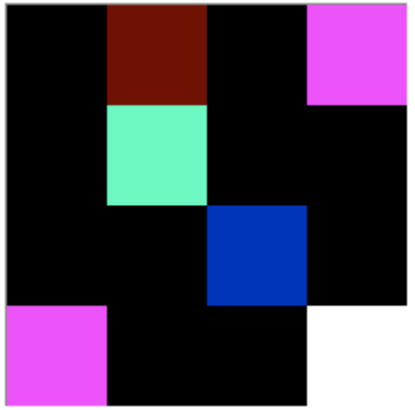
Plain PPM Format

```
P3
4 4
255
```

Image header

```
0   0   0     100 0   0       0   0   0     255   0 255
0   0   0       0 255 175     0   0   0       0     0   0
0   0   0       0   0   0     0 15 175       0     0   0
255 0 255     0   0   0       0   0   0     255   255 255
```

Image body



- P3: magic number to confirm the image format
- 4 4: 4x4 image
- 255: the maximum value of a channel is 255
- The rest of the data: pixel values

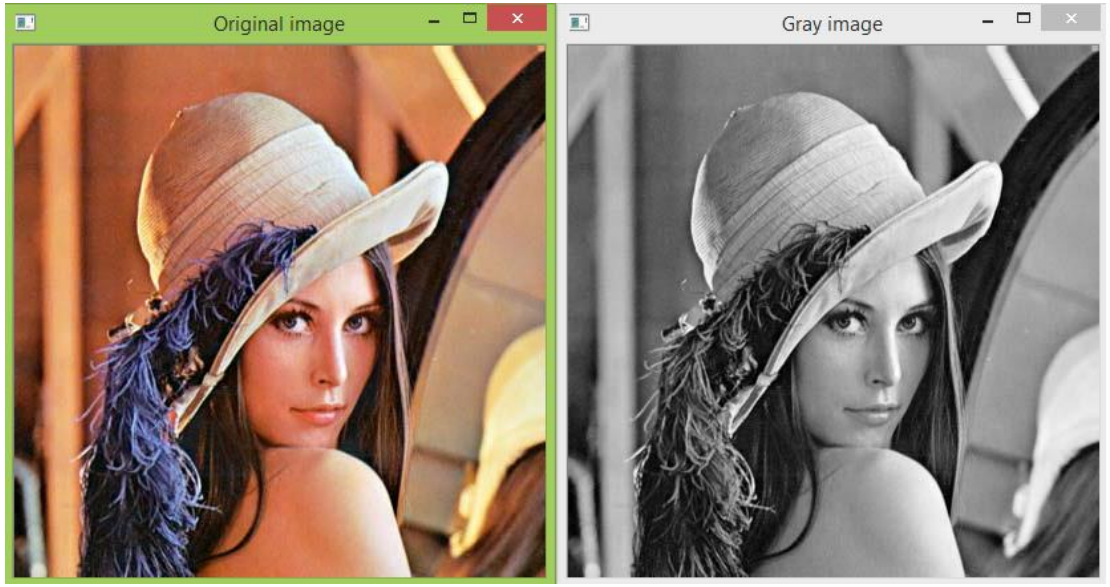
Digital Image Format

- Data Compression
  - Many images have local pixel correlations
  - Compression may reduce image size considerably
  - The compression is lossless if it can recover image exactly
    - RLE 000001111011111000000001111

      5(0) 4(1) 1(0) 5(1) 9(0) 4(1)
  - The compression is lossy if it cannot reconstruct exactly
    - JPEG

RGB to Grayscale

- A monochrome luminance signal (Y) can be created by combining RGB signals
- NTSC broadcast TV quantization formula:
  - Y= 0.299R + 0.587G + 0.114B



Python with OpenCV

```python
import cv2

img = cv2.imread('ntnu.png')

#get the pixel RGB at 50,100
pixel = img[50,100]
print(pixel)

#set pixels in a region to red
img[10:20, 50:90]= [0, 0, 255]

cv2.imshow("Original Image", img)
if cv2.waitKey(0) == 27:
    cv2.destroyAllWindows()
```

img[50,100]: the first number is for vertical axis
and the second number is for horizontal axis

Note: the order of RGB in Python-Opencv is [B, G, R]