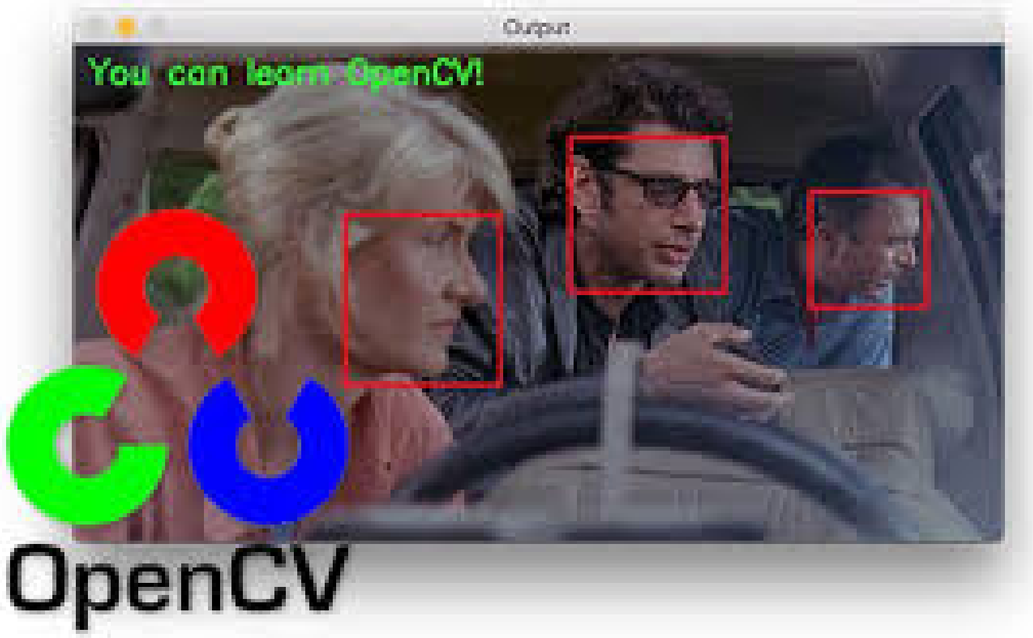
OpenCV Lec-1

OpenCV

- OpenCV (Open Source Computer Vision Library) is an open-source toolkit for real-time image and video processing.
- Originally developed by Intel, now maintained by OpenCV.org and supported by a large community.
- Written in C/C++ with bindings available for Python, Java, and other languages.
- Provides tools for image processing, computer vision, machine learning, and deep learning.
- Widely used in applications such as robotics, drones, self-driving cars, surveillance, and augmented reality.



What is an image ?

Image

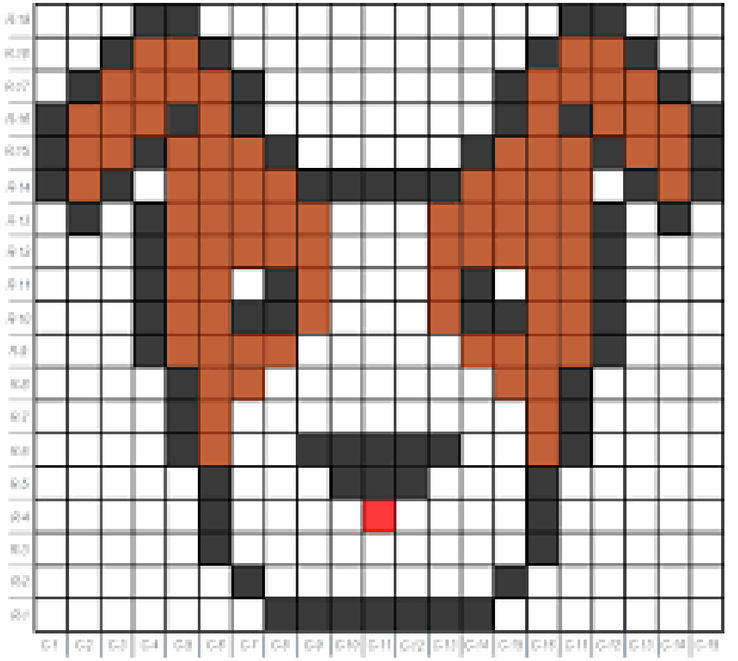
Image stores a particular type of data as in any other format

An image is a grid of pixels, where each pixel is a small square with color (or brightness) information

Each pixel has a colour to it

Color Spaces

- **RGB**: Standard for images
- **BGR**: OpenCV default
- **HSV**: For color-based filtering



[white white white
white white black
white black brown]

Image Representation

GreyScale

```
gray_img = np.array([
    [0, 128, 255],
    [60, 180, 90],
    [255, 255, 0]
], dtype=np.uint8)
```

Colour

```
color_img = np.array([
    [[255,   0,   0], [0, 255,   0], [0,   0, 255]],  # Red, Green, Blue
    [[255, 255,   0], [0, 255, 255], [255, 0, 255]],  # Yellow, Cyan, Magenta
    [[128, 128, 128], [64, 64, 64], [0, 0, 0]]        # Gray, Dark Gray, Black
], dtype=np.uint8)
```

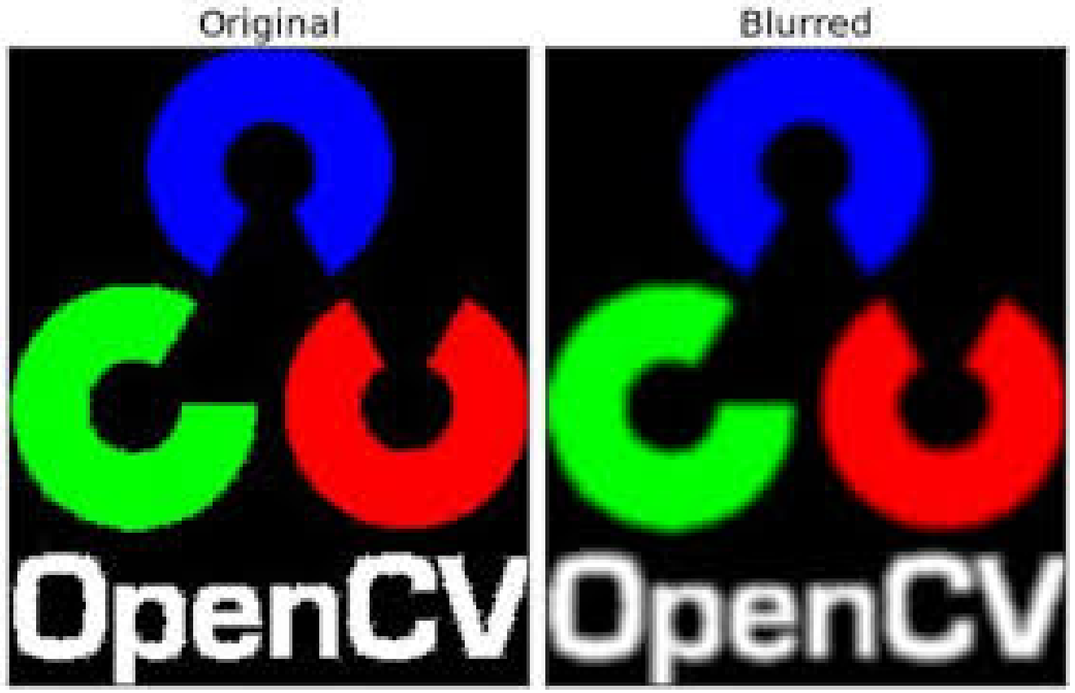
Image Properties

Property	What It Means
Resolution	Total number of pixels = Width × Height e.g., 1920 × 1080 = Full HD
Shape	Grayscale: (H, W) Color: (H, W, 3)
Channels	Number of color components: - 1 (grayscale) - 3 (RGB/BGR) - 4 (RGBA)
Bit Depth	- 8-bit: pixel values from 0–255 - 16-bit: 0–65535 (used in medical/scientific imaging)
Data Type	Usually uint8 (unsigned int, 8 bits), can also be float32 for scientific tasks
Color Space	RGB, BGR, HSV, LAB, etc. — changes how color is represented internally
Pixel Value	Each pixel is just a **number** (grayscale) or **vector of numbers** (RGB/BGR)
Intensity/Brightness	Intensity = how strong a color channel is Brightness = how light the final color appears to human eyes

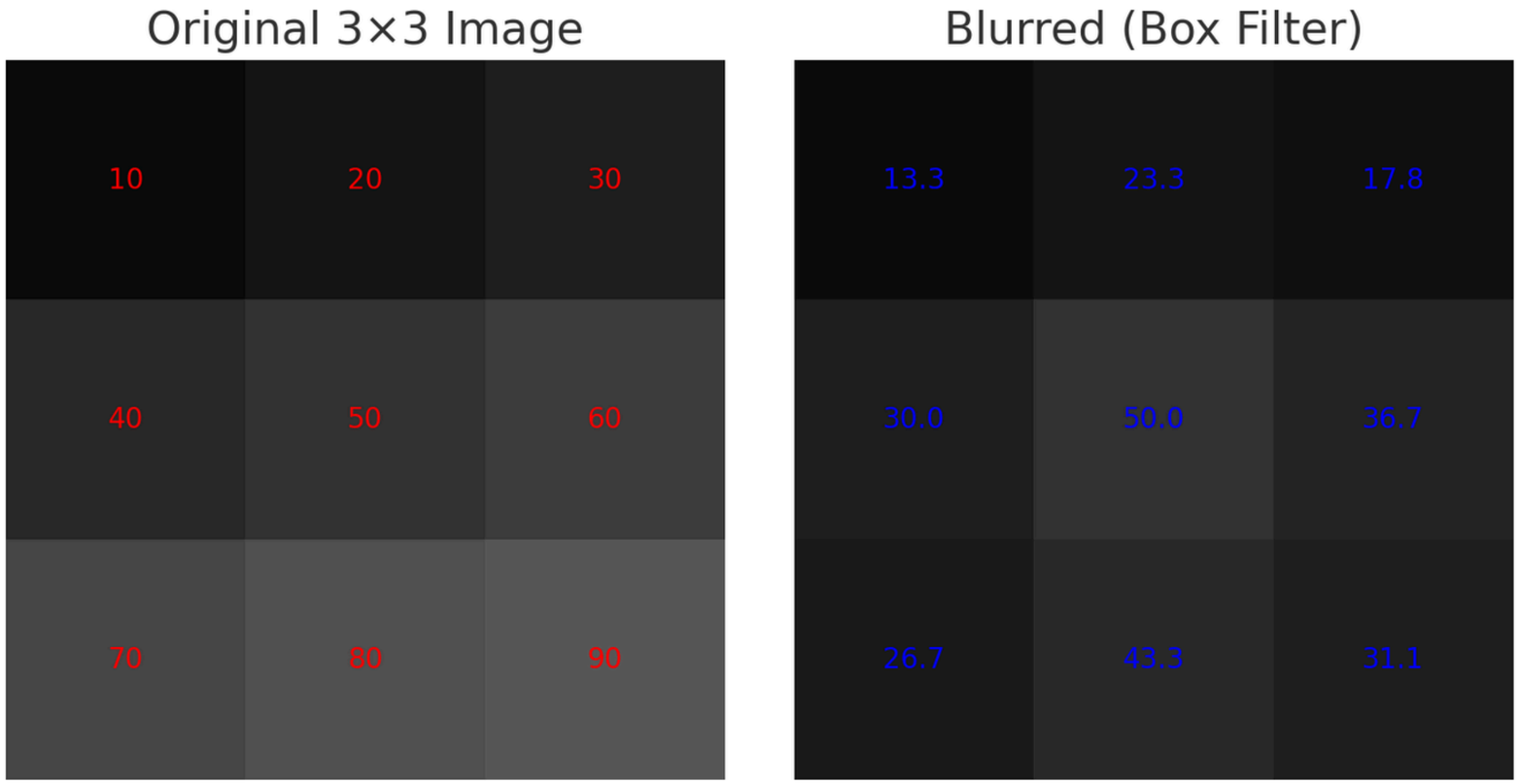
Image Operations

Blur

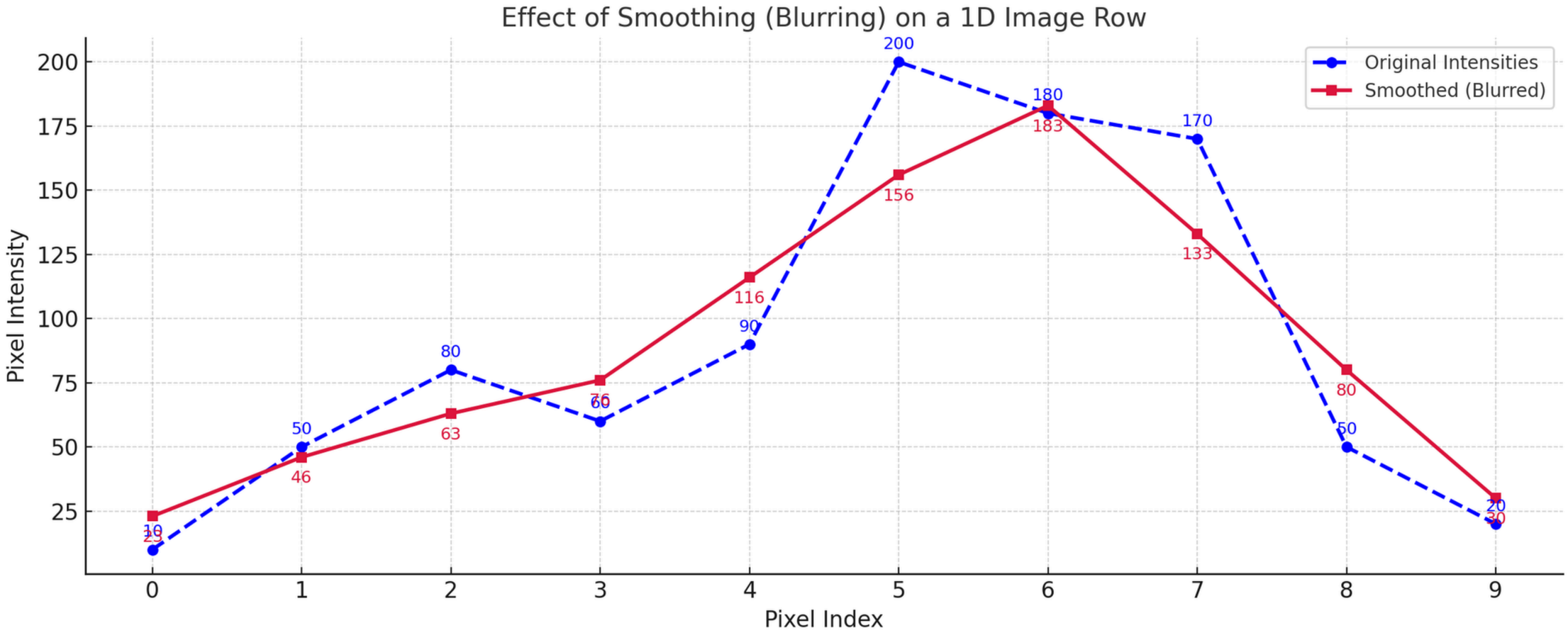
Blurring is used to:
Reduce noise
Soften edges
Pre-process before edge detection or object recognition



Blur


Original 3×3 Image
10
20
30
40
50
60
70
80
90
Blurred (Box Filter)
13.3
23.3
17.8
30.0
50.0
36.7
26.7
43.3
31.1

Blur


Effect of Smoothing (Blurring) on a 1D Image Row
Original Intensities
Smoothed (Blurred)
Pixel Intensity
Pixel Index
200
175
150
125
100
75
50
25
0
1
2
3
4
5
6
7
8
9
25
50
80
60
90
200
180
170
50
30
10
46
63
76
116
156
183
133
80
20

Blur

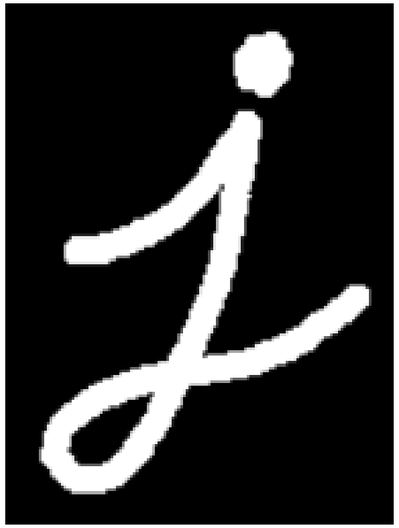
Blur Type	OpenCV Function	What It Does
Averaging Blur	cv2.blur()	Mean of surrounding pixels
Gaussian Blur	cv2.GaussianBlur()	Weighted mean using Gaussian function
Median Blur	cv2.medianBlur()	Median of pixel neighborhood
Bilateral Filter	cv2.bilateralFilter()	Blur but keeps edges

Other Operations

Operation	Function Name	Syntax / Parameters	Use-Case / Purpose
Resize	cv2.resize()	cv2.resize(img, (width, height))	Resize image (scale up/down)
Rotate	cv2.rotate()	cv2.rotate(img, cv2.ROTATE_90_CLOCKWISE)	Rotate by 90/180 degrees
Flip	cv2.flip()	cv2.flip(img, 0) (vertical), 1 (horizontal)	Flip/mirror image
Color Convert	cv2.cvtColor()	cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)	Convert BGR ↔ Grayscale / HSV / RGB
Crop	NumPy slicing	img[y1:y2, x1:x2]	Select a region of interest
Concatenate	cv2.hconcat() / vconcat()	cv2.hconcat([img1, img2])	Combine multiple images side-by-side

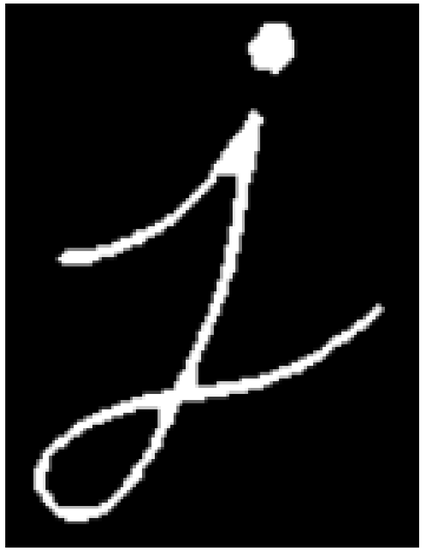
Morphological Operations

A set of operations that process images based on shapes. Morphological operations apply a structuring element to an input image and generate an output image.



Normal

Dilated



Erosion

Thresholding

Thresholding is a technique to convert a grayscale image into a binary image (black & white) based on a certain intensity cutoff

```
if pixel_value > threshold:
    output = 255   # white
else:
    output = 0     # black
```

Original Image

Thresholded and segmented Image

Types of Thresholding

Type	Code	Description
Binary	cv2.THRESH_BINARY	If > threshold → white; else → black
Binary Inverted	cv2.THRESH_BINARY_INV	If > threshold → black; else → white
Truncate	cv2.THRESH_TRUNC	If > threshold → set to threshold
To Zero	cv2.THRESH_TOZERO	If > threshold → keep value; else → 0
To Zero Inverted	cv2.THRESH_TOZERO_INV	If > threshold → 0; else → keep value