# MIS4311
# Machine Learning Applications

Fall 2025

Lecture #13

# Computer Vision and Image Processing

The ability to see and perceive the world comes naturally to us humans.

When it comes to machines, this learning process becomes **complicated**. The process of parsing through an image and detecting objects involves **multiple and complex steps**, including feature extraction (edges detection, shapes, etc), feature classification, etc.

**Computer Vision** is a field of deep learning that enables machines to see, identify and process images like humans.

## Computer Vision and Image Processing

**OpenCV,** or Open Source Computer Vision library, started out as a research project at Intel.

It is currently **the largest computer vision library** in terms of the number of functions it contains.

OpenCV contains implementations of more than 2500 algorithms.

It is freely available for commercial as well as academic purposes.

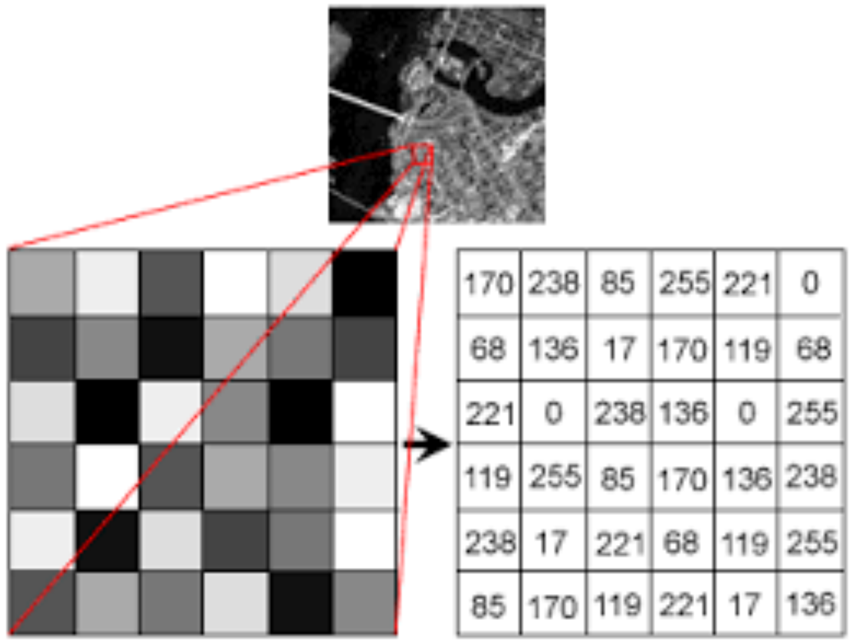
The library has **interfaces for multiple languages**, including Python, Java, and C++.

# Reading, Writing and Displaying Images

Machines see and **process everything using numbers**, including images and text. How do you convert images to numbers?

**Pixel values:**

Every number represents the pixel intensity at that particular location. In the image, pixel values shown for a grayscale image where **every pixel contains only one value i.e. the intensity of the black color at that location.**



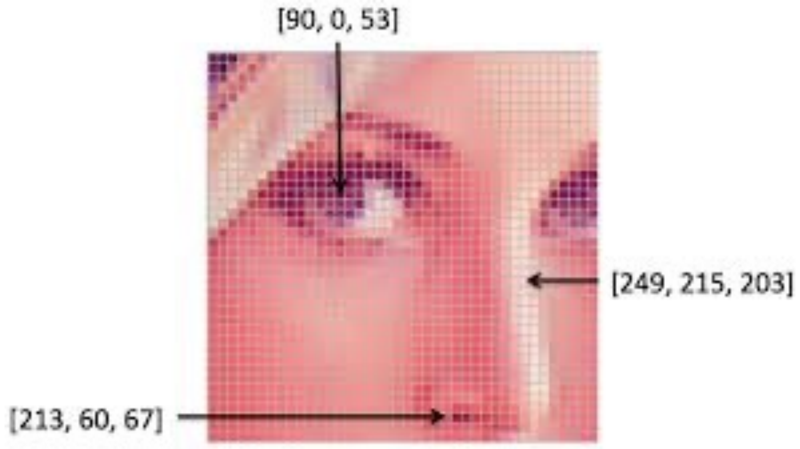
| | | | | | |
|---|---|---|---|---|---|
| 170 | 238 | 85 | 255 | 221 | 0 |
| 68 | 136 | 17 | 170 | 119 | 68 |
| 221 | 0 | 238 | 136 | 0 | 255 |
| 119 | 255 | 85 | 170 | 136 | 238 |
| 238 | 17 | 221 | 68 | 119 | 255 |
| 85 | 170 | 119 | 221 | 17 | 136 |

Note that **color images will have multiple values for a single pixel**. These values represent the **intensity of respective channels** – Red, Green and Blue channels for RGB images, for instance.

# Reading, Writing and Displaying Images

Reading and writing images is essential to any computer vision project. And the OpenCV library makes this function a whole lot easier.

By default,
the ***imread*** function reads images in the BGR (Blue-Green-Red) format. We can read images in different formats using extra flags in the imread function:



- **cv2.IMREAD_COLOR:** Default flag for loading a color image
- **cv2.IMREAD_GRAYSCALE:** Loads images in grayscale format
- **cv2.IMREAD_UNCHANGED:** Loads images in their given format

## Changing Color Spaces

**A color space** is a protocol for representing colors in a way that makes them easily reproducible. We know that grayscale images have single pixel values and color images contain 3 values for each pixel – the intensities of the Red, Green and Blue channels.
Most computer vision use cases process images in RGB format. However, **applications like video compression** and device independent storage – these are **heavily dependent on other color spaces**, like the Hue-Saturation-Value or HSV color space.
As you understand a RGB image consists of the color intensity of different color channels, i.e. the intensity and color information are mixed in RGB color space but in HSV color space the color and intensity information are separated from each other. **This makes HSV color space more robust to lighting changes.**
OpenCV reads a given image in the BGR format by default. So, you'll need to change the color space of your image from BGR to RGB when reading images using OpenCV by COLOR_BGR2RGB function

# Resizing Images

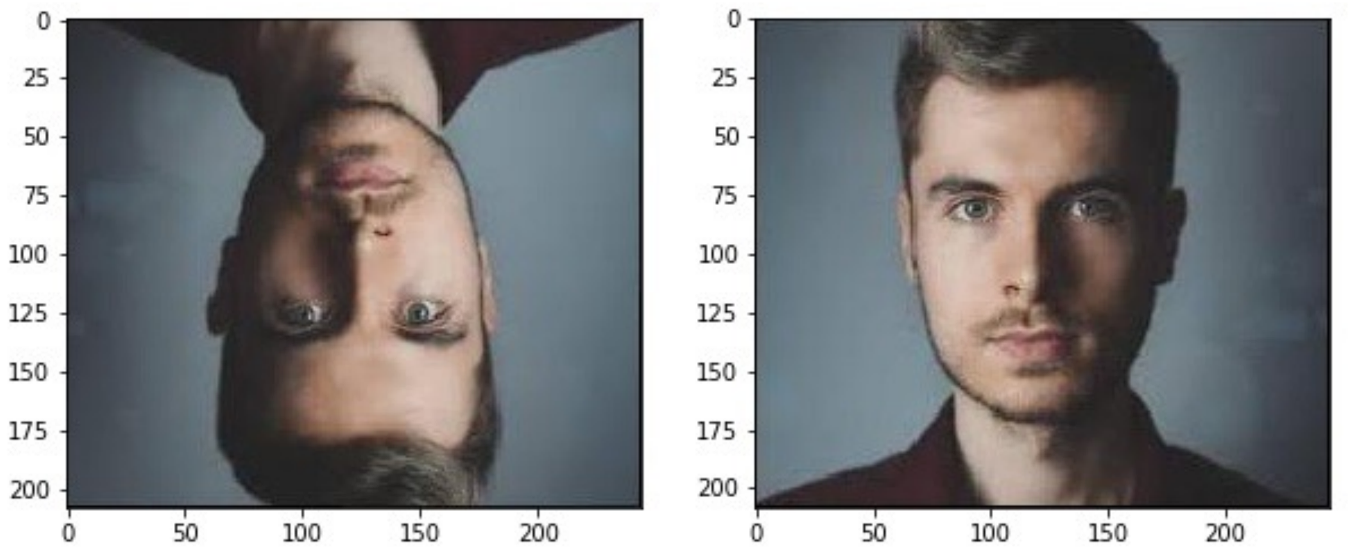
Images can be easily **scaled up and down using OpenCV**. This operation is **useful for training deep learning models** when we need **to convert images to the model's input shape**.

Different interpolation and **downsampling methods** are supported by OpenCV, which can be used by the **following parameters:**

1.INTER_NEAREST: Nearest neighbor interpolation
2.INTER_LINEAR: Bilinear interpolation
3.INTER_AREA: Resampling using pixel area relation
4.INTER_CUBIC: Bicubic interpolation over 4×4 pixel neighborhood
5.INTER_LANCZOS4: <u>Lanczos interpolation</u> over 8×8 neighborhood

# Image Rotation

Rotation is one of the most used and easy to implement data augmentation techniques. As the name suggests, **it involves rotating the image at an arbitrary angle** and providing it the same label as the original image.



## Image Translation

Image translation is **a geometric transformation that maps the position of every object in the image to a new location** in the final output image. After the translation operation, an object present at location (x,y) in the input image is shifted to a new position (X,Y):

$$X = x + dx$$
$$Y = y + dy$$

Here, dx and dy are the respective translations along different dimensions.

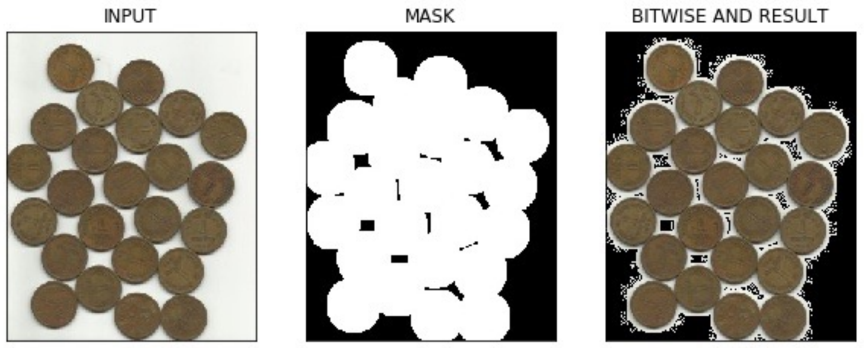
Image translation **can be used to add shift invariance to the model**, as by translation we can change the position of the object in the image give more variety to the model that leads to better generalizability which works in difficult conditions i.e. when the object is not perfectly aligned to the center of the image.

# Image Processing

**Thresholding** is an image **segmentation** method. It compares pixel values with a threshold value and updates it accordingly.

**Image segmentation** is the task of classifying every pixel in the image to some class. For example, classifying every pixel as foreground or background. Image segmentation is important for extracting the relevant parts from an image.

**Bitwise operations** include AND, OR, NOT and XOR. In computer vision, these operations are very useful when we have a mask image and want to apply that mask over another image to extract the region of interest



# Image Processing

**Edge Detection:**

Edges are the points in an image where the image brightness changes sharply or has discontinuities. Such discontinuities generally correspond to:

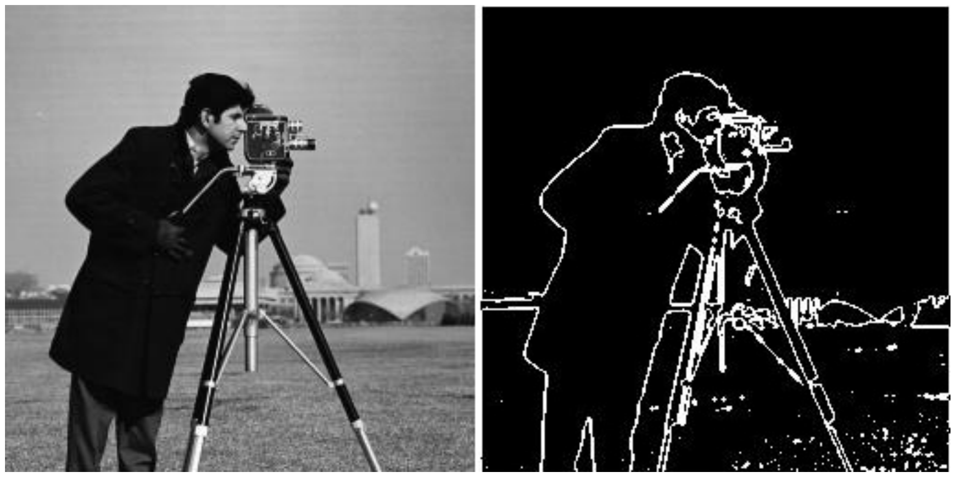
•Discontinuities in depth

•Discontinuities in surface orientation

•Changes in material properties

•Variations in scene illumination

Edges are very useful features of an image that can be used for different applications like classification of objects in the image and localization



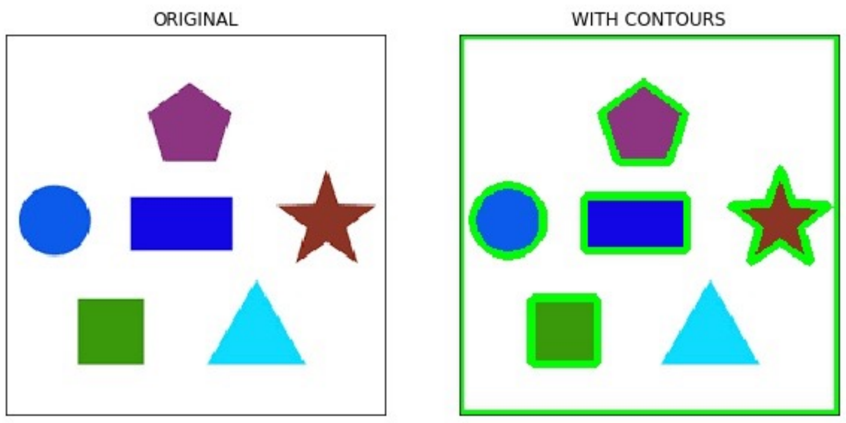
# Image Processing

**Image Contours:**

A contour is **a closed curve of points or line segments** that represents the boundaries of an object in the image. Contours are essentially the shapes of objects in an image.

Unlike edges, **contours are not part of an image**. Instead, they are an abstract collection of points and line segments corresponding to the shapes of the object(s) in the image.

We can use **contours to count the number of objects in an image, categorize objects on the basis of their shapes, or select objects of particular shapes from the image.**



## Next Week

*General Application

Thank you for your participation ☺