The Superior University, Lahore

Assignment-I (Fall 2023)

Course Title:	Programming for AI				Course Code:	CAI601410	Credit Hours: 4
Instructor:	Prof. Rasikh Ali				Programme Name:	BSDS	
Semester:	4th	Batch:	F23	Section:	BSDSM-4A	Date:	1st February, 2025
Time Allowed:					Maximum Marks:		
Student's Name:	Asaad Iqbal				Reg. No.	SU92-BSDSM-F23-020	

Lab-Task 5
1: Computer Vision using Open-CV

Task 1

Computer Vision Using Open-CV

1. Introduction

This notebook explores fundamental image processing techniques using OpenCV, including reading, displaying, saving images, color transformations, drawing shapes, and applying various image operations.

2. Image Handling and Displaying

2.1 Reading an Image

- **Functionality:** Loads an image from a specified path using cv2.imread().
- **Error Handling:** Checks if the image is successfully loaded; otherwise, it prints an error message.

2.2 Displaying an Image

- **Functionality:** Uses matplotlib.pyplot to display an image after converting it from BGR to RGB.
- **Purpose:** Ensures the correct color representation when displaying images using OpenCV.

2.3 Saving an Image

- **Functionality:** Saves the loaded image to the current working directory using cv2.imwrite().

- **Error Handling:** Ensures the directory exists and the image is successfully written.

3. Image Processing Techniques

3.1 Converting to Grayscale

- **Functionality:** Converts a colored image to grayscale using cv2.cvtColor(..., cv2.COLOR_BGR2GRAY).
- **Purpose:** Reduces computational complexity and is essential for many computer vision applications.

3.2 Bitwise Operations

- **Functionality:** Performs bitwise AND, OR, XOR, and NOT operations on images.
- **Purpose:** Useful for masking and extracting specific regions of interest.

3.3 Image Resizing

- **Functionality:** Resizes an image to specified dimensions using cv2.resize().
- **Use Case:** Standardizes image dimensions for machine learning models.

3.4 Image Rotation

- **Functionality:** Rotates an image by a given angle using cv2.getRotationMatrix2D() and cv2.warpAffine().
- **Use Case:** Augmenting datasets and correcting orientation.

4. Image Drawing Operations

4.1 Drawing a Line

- **Functionality:** Draws a straight line on an image using cv2.line().
- **Parameters:**
    - Start and end points.
    - Line color (BGR format).
    - Thickness of the line.

- **Use Case:** Highlighting boundaries and marking specific regions.

4.2 Drawing a Rectangle

- **Functionality:** Draws a rectangle on an image using cv2.rectangle().
- **Parameters:**
    - Top-left and bottom-right corner points.
    - Rectangle color and thickness.
    - Optional filled rectangle with thickness=-1.
- **Use Case:** Bounding boxes for object detection.

4.3 Drawing a Circle

- **Functionality:** Draws a circle on an image using cv2.circle().
- **Parameters:**
    - Center point and radius.
    - Circle color and thickness.
    - Optional filled circle with thickness=-1.
- **Use Case:** Marking points of interest or centroids.

5. Image Filtering and Transformation

5.1 Blurring

- **Functionality:** Applies Gaussian blur using cv2.GaussianBlur() to reduce noise.
- **Use Case:** Useful for pre-processing before edge detection.

5.2 Edge Detection (Canny)

- **Functionality:** Detects edges in an image using cv2.Canny().
- **Use Case:** Extracts structural information, commonly used in feature extraction.

5.3 Image Thresholding

- **Functionality:** Converts images to binary using global and adaptive thresholding.
- **Use Case:** Segmentation and object detection.

7. Conclusion

This notebook provides an introduction to OpenCV with basic image processing techniques. It serves as a foundation for advanced computer vision applications, such as object detection and deep learning-based image analysis.