

The Superior University

|  |  |  |
| --- | --- | --- |
| Name: Muhammad Bilal Sajid | Roll No: 018 | Course: PAI Lab |
| Semester: 4th | Section: BSAI 4A | Department: |
| Submitted To: | Total Marks: | Date: |

**Lab 5**

**Task: Implement all the topics/functions of OpenCV, in a notebook file**

**Image Processing using OpenCV in Python**

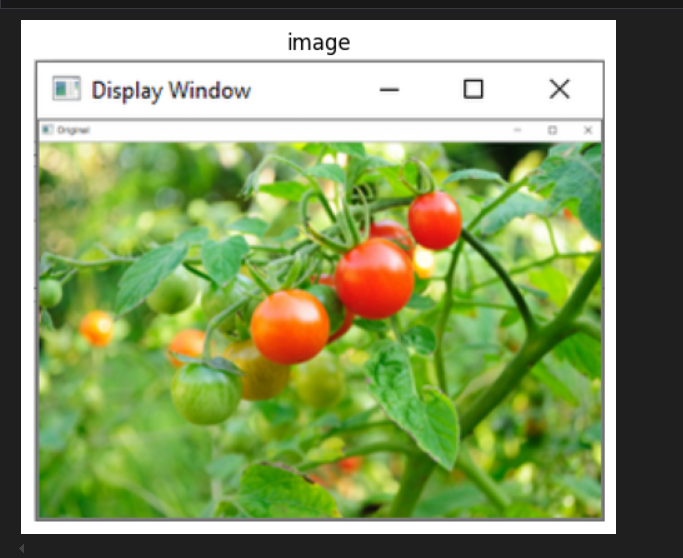
**1. Displaying Images**

* The script loads an image from a specified path using cv2.imread().
* Converts it from BGR to RGB using cv2.cvtColor().
* Uses matplotlib.pyplot to display the image without axes.

**Why?**

* OpenCV loads images in BGR format, while matplotlib expects RGB format.
* Conversion ensures proper color representation.

**Screenshot:**

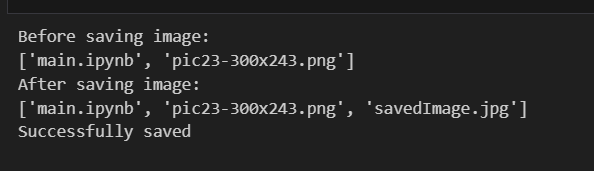


**2. Saving Images**

* Reads an image and changes the directory.
* Saves the image using cv2.imwrite().
* Prints the directory contents before and after saving to verify the process.

**Why?**

* Essential for storing processed images for further use.

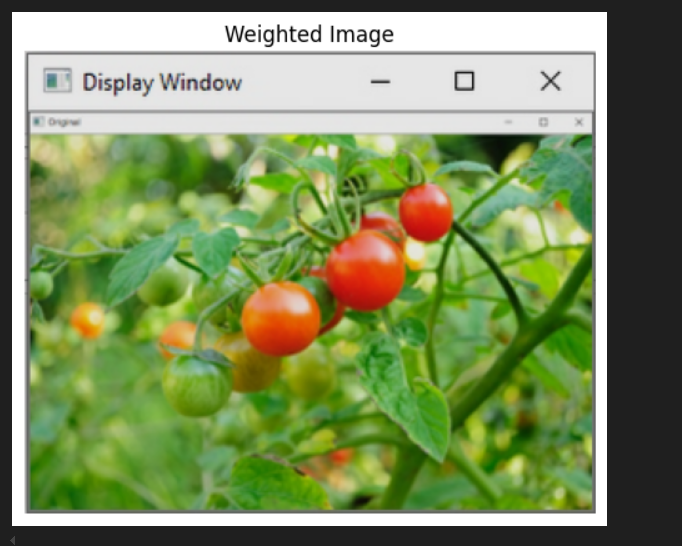
**Screenshot:** **

**3. Image Blending**

* Loads two images and merges them using cv2.addWeighted().
* Uses different weight factors to control blending.
* Displays the final blended image.

**Why?**

* Useful in overlaying images for effects and enhancements.

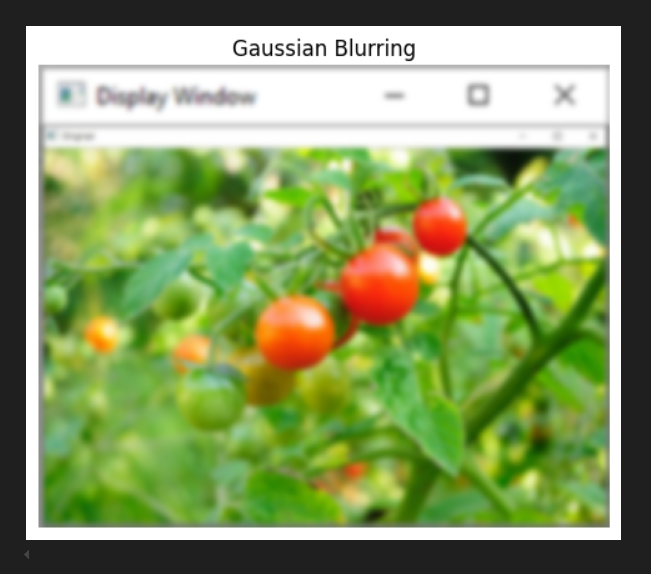
**Screenshot:** 

**4. Blurring Techniques**

* Applies different blurring methods:
  + **Gaussian Blur**: Removes high-frequency noise.
  + **Median Blur**: Effective in reducing salt-and-pepper noise.
  + **Bilateral Filter**: Preserves edges while reducing noise.
* Displays results using matplotlib.pyplot.

**Why?**

* Blurring smoothens images and is useful in preprocessing before object detection.

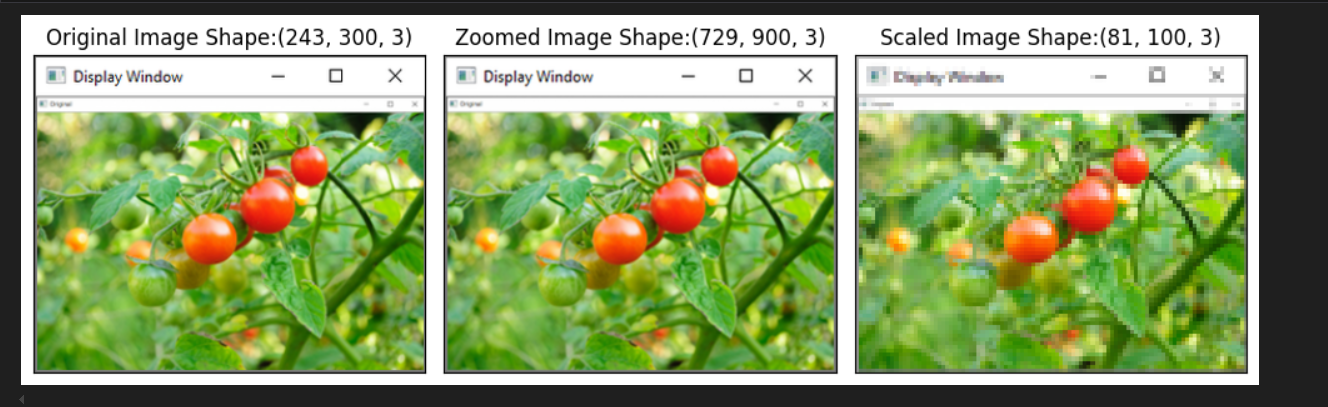
**Screenshot:** 

**5. Image Scaling**

* Resizes the image using cv2.resize():
  + Zooms in using cubic interpolation.
  + Zooms out using area interpolation.
* Displays original, zoomed, and scaled images in a subplot.

**Why?**

* Scaling is essential in computer vision tasks to normalize input sizes.

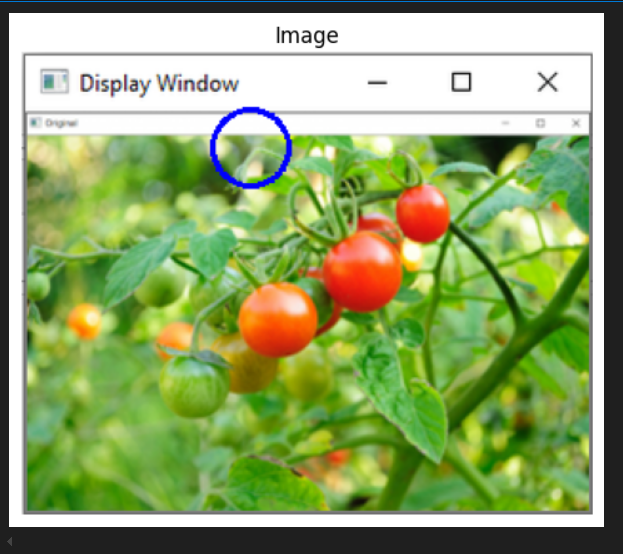
**Screenshot:** **

**6. Drawing a Circle on an Image**

* Reads an image using cv2.imread().
* Defines center coordinates and radius for the circle.
* Uses cv2.circle() to draw a blue circle with a thickness of 2 pixels.
* Converts the image from BGR to RGB for correct color display in matplotlib.pyplot.
* Displays the modified image without axes.

**Why?**

* Useful for marking specific points in an image, such as object locations or key features in computer vision tasks.

**Screenshot:** 

### 7. Shi-Tomasi Corner Detection

* Loads an image and converts it to grayscale using cv2.cvtColor().
* Applies Shi-Tomasi Corner Detection using cv2.goodFeaturesToTrack().
* Detects 100 best corners with a minimum quality level of 0.01 and minimum distance of 10 pixels between corners.
* Converts corner coordinates to integers and marks them with red circles.
* Converts the processed image to RGB for display and visualizes it using matplotlib.pyplot.

**Why?**

* Shi-Tomasi Corner Detection is useful for detecting feature points in images for tracking, matching, and object recognition.

**Screenshot:** 