**COMPUTER VISION**

Surya Suresh

NSTI W TRIVANDRUM

**1. Image Resizing, Cropping, and Rotation**

**# Load the necessary library**

import cv2

import matplotlib.pyplot as plt

**# Load an image**

image = cv2.imread('3.jpg')

**# Convert the image from BGR (OpenCV format) to RGB (Matplotlib format)**

image\_rgb = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

**# Resize image to 256x256 pixels**

resized\_image = cv2.resize(image\_rgb, (125, 128))

**# Display the original and resized images**

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

plt.title('Original Image')

plt.imshow(image\_rgb)

plt.axis('off')

plt.subplot(1, 2, 2)

plt.title('Resized Image (125x128)')

plt.imshow(resized\_image)

plt.axis('off')

plt.show()

**# Save or display the resized image**

# cv2.imwrite('resized\_image.jpg', resized\_image)

**# Crop image to a region (x, y, width, height)**

cropped\_image = image\_rgb[50:130, 50:200]

**# Display the original and resized images**

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

plt.title('Original Image')

plt.imshow(image\_rgb)

plt.axis('off')

plt.subplot(1, 2, 2)

plt.title('cropped\_image')

plt.imshow(cropped\_image)

plt.axis('off')

plt.show()

**# Rotate image by 45 degrees**

(h, w) = image\_rgb.shape[:2]

center = (w // 2, h // 2)

M = cv2.getRotationMatrix2D(center, 45, 1.0)

rotated\_image = cv2.warpAffine(image\_rgb, M, (w, h))

**# Display the original and resized images**

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

plt.title('Original Image')

plt.imshow(image\_rgb)

plt.axis('off')

plt.subplot(1, 2, 2)

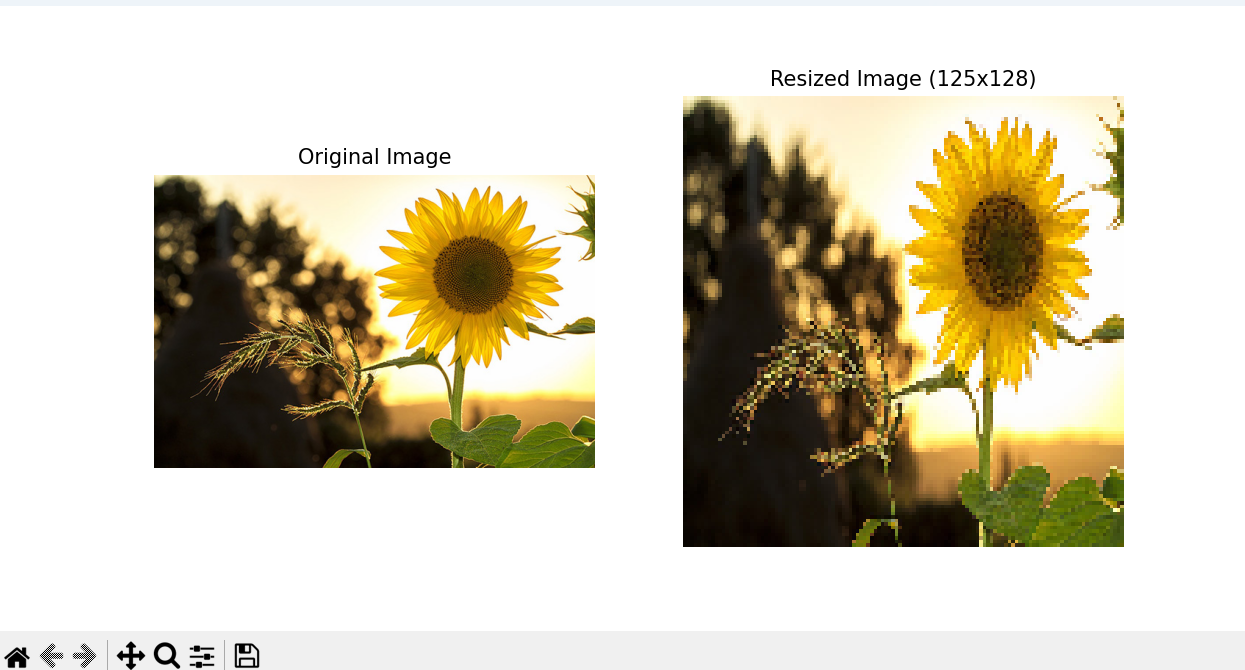
plt.title('rotated\_image')

plt.imshow(rotated\_image)

plt.axis('off')

plt.show()

**OUTPUT:-**

****

****

****

**2. Loading\_Image\_Formats\_Tutorial**

import cv2

import matplotlib.pyplot as plt

**# Load an image using OpenCV**

image\_path = "3.jpg"

image\_cv2 = cv2.imread(image\_path)

**# Convert the image from BGR to RGB**

image\_cv2\_rgb = cv2.cvtColor(image\_cv2, cv2.COLOR\_BGR2RGB)

**# Display the image**

plt.imshow(image\_cv2)

plt.title('Image loaded with OpenCV')

plt.show()

from PIL import Image

**# Load an image using PIL**

image\_pil = Image.open(image\_path)

**# Display the image**

plt.imshow(image\_pil)

plt.title('Image loaded with PIL')

plt.show()

import imageio

**# Load an image using imageio**

image\_imageio = imageio.imread(image\_path)

**# Display the image**

plt.imshow(image\_imageio)

plt.title('Image loaded with imageio')

plt.show()

**# PNG image path**

image\_path\_png = "4.png"

image\_path\_jpg = "2.jpg"

**# OpenCV**

image\_cv2\_png = cv2.imread(image\_path\_png)

image\_cv2\_png\_rgb = cv2.cvtColor(image\_cv2\_png, cv2.COLOR\_BGR2RGB)

plt.imshow(image\_cv2\_png\_rgb)

plt.title('PNG loaded with OpenCV')

plt.show()

**# PIL**

image\_pil\_png = Image.open(image\_path\_png)

plt.imshow(image\_cv2\_png\_rgb)

plt.title('PNG loaded with OpenCV')

plt.show()

**# imageio**

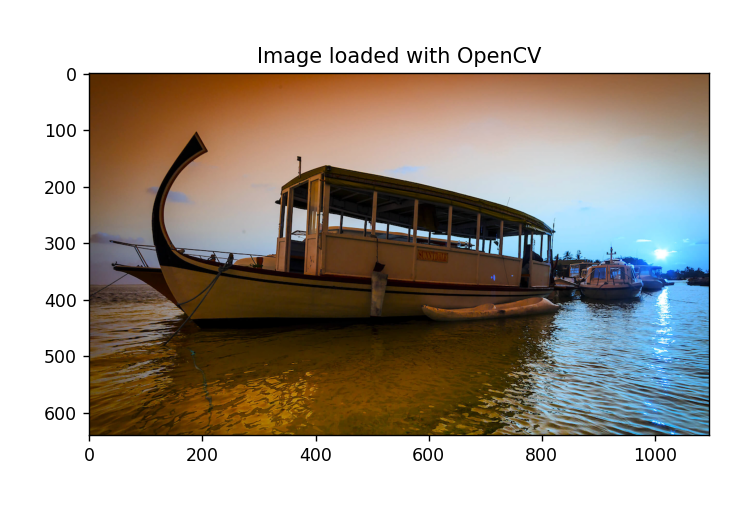
image\_imageio\_png = imageio.imread(image\_path\_png)

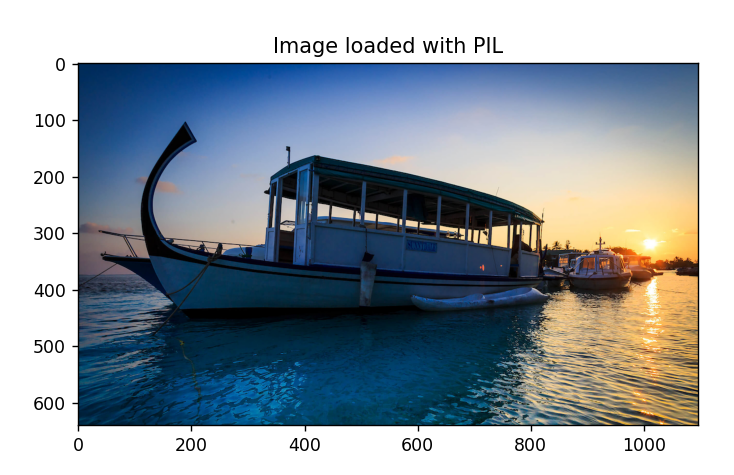
plt.imshow(image\_cv2\_png\_rgb)

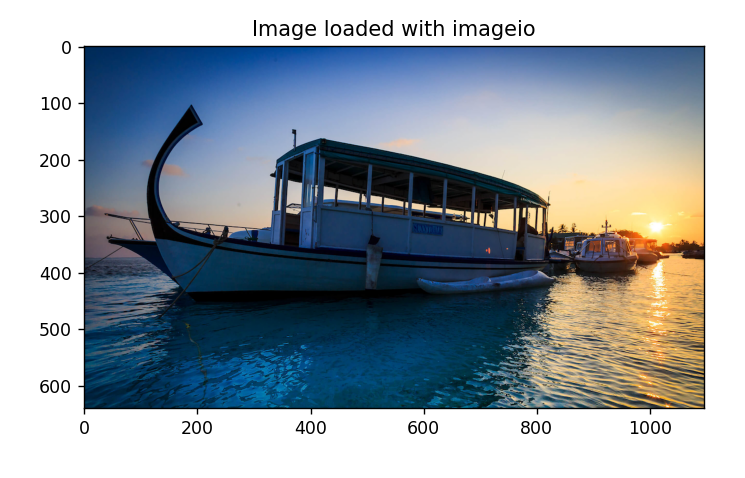
plt.title('PNG loaded with OpenCV')

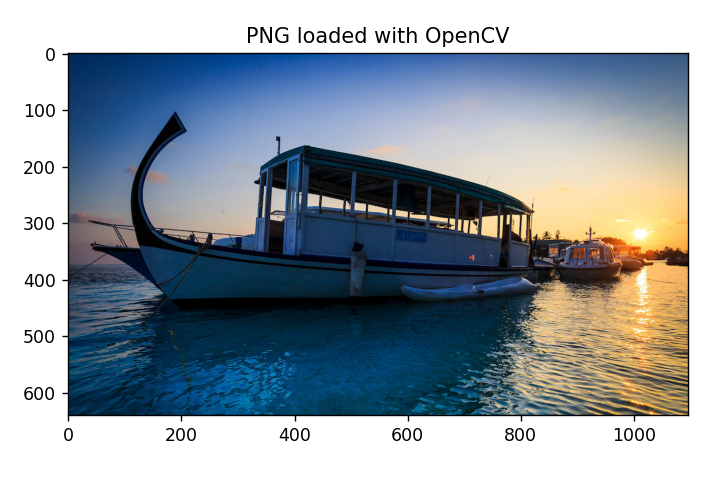
plt.show()

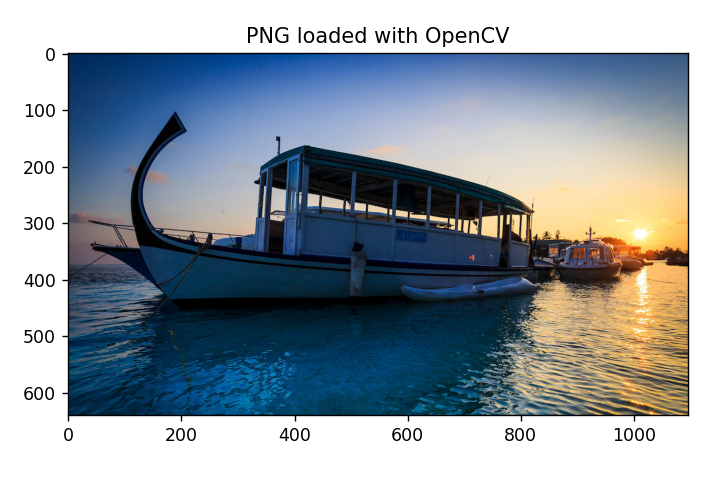
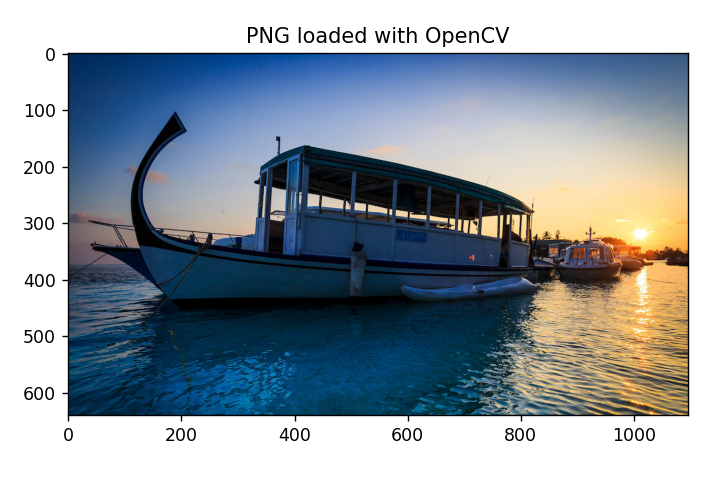
**OUTPUT:-**

****

****

****

****

****

**3. Image Denoising**

**# import necessary libraries**

import cv2

import matplotlib.pyplot as plt

**# Load an image**

image = cv2.imread('1.jpg')

**# Convert the image from BGR (OpenCV format) to RGB (Matplotlib format)**

image\_rgb = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

**# Apply Gaussian blur to denoise**

denoised\_image = cv2.GaussianBlur(image\_rgb, (11, 11), 0)

**# Display the original and resized images**

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

plt.title('Original Image')

plt.imshow(image\_rgb)

plt.axis('off')

plt.subplot(1, 2, 2)

plt.title('denoised\_image')

plt.imshow(denoised\_image)

plt.axis('off')

plt.show()

**# Convert to grayscale**

gray\_image = cv2.cvtColor(image\_rgb, cv2.COLOR\_BGR2GRAY)

**# Apply histogram equalization**

equalized\_image = cv2.equalizeHist(gray\_image)

**# Display the original and resized images**

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

plt.title('Gray Image')

plt.imshow(gray\_image, cmap="gray")

plt.axis('off')

plt.subplot(1, 2, 2)

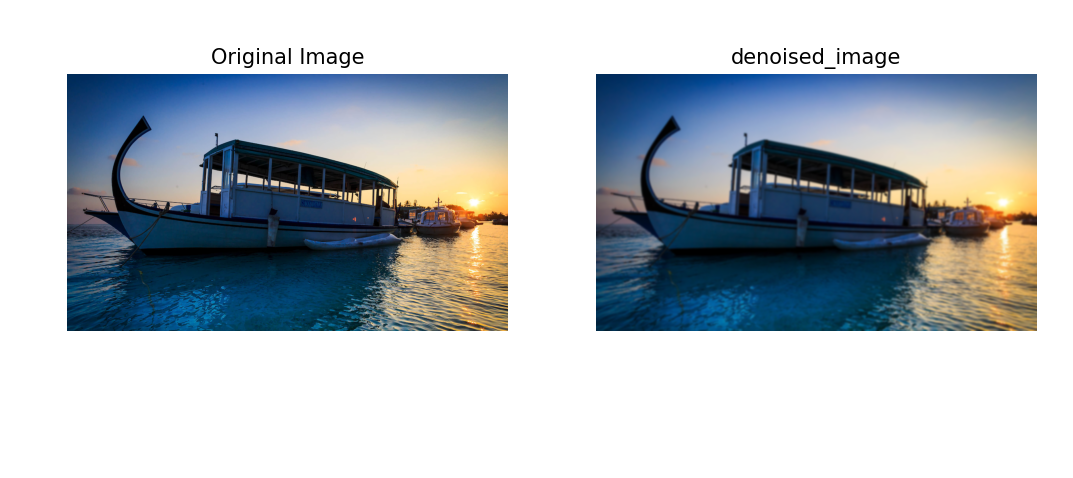
plt.title('equalized\_image')

plt.imshow(equalized\_image, cmap="gray")

plt.axis('off')

plt.show()

**OUTPUT:-**

****

****