COMPUTER VISION

NANDANA S S

NSTI W TRIVANDRUM

1. Image Resizing, Cropping, and Rotation

```
import cv2
import matplotlib.pyplot as plt
from PIL import Image
import imageio

# Load an image using OpenCV
image_path = "apple.jpeg"
image_cv2 = cv2.imread(image_path)

# Convert the image from BGR to RGB for proper display
image_cv2_rgb = cv2.cvtColor(image_cv2, cv2.COLOR_BGR2RGB)

# Display the image loaded with OpenCV
plt.imshow(image_cv2_rgb)
plt.title('Image loaded with OpenCV')
plt.axis('off')  # Hide the axis
plt.show()

# Load an image using PIL
image_pil = Image.open(image_path)

# Display the image loaded with PIL
plt.imshow(image_pil)
plt.title('Image loaded with PIL')
plt.axis('off')  # Hide the axis
plt.show()

# Load an image using imageio
image_imageio = imageio.imread(image_path)
```

```
# Display the image loaded with imageio
plt.imshow(image_imageio)
plt.title('Image loaded with imageio')
plt.axis('off')  # Hide the axis
plt.show()

# Define path for JPG images
image_path_jpg = "logo.jpg"

# OpenCV
image_cv2_jpg = cv2.imread(image_path_jpg)
image_cv2_jpg_rgb = cv2.cvtColor(image_cv2_jpg, cv2.COLOR_BGR2RGB)
plt.imshow(image_cv2_jpg_rgb)
plt.title('JPG loaded with OpenCV')
plt.axis('off')  # Hide the axis
plt.show()

# PIL
image_pil_jpg = Image.open(image_path_jpg)
plt.imshow(image_pil_jpg)
plt.title('JPG loaded with PIL')
plt.axis('off')  # Hide the axis
plt.show()

# imageio
image_imageio_jpg = imageio.imread(image_path_jpg)
plt.imshow(image_imageio_jpg)
plt.title('JPG loaded with imageio')
plt.axis('off')  # Hide the axis
plt.show()
```

OUTPUT:

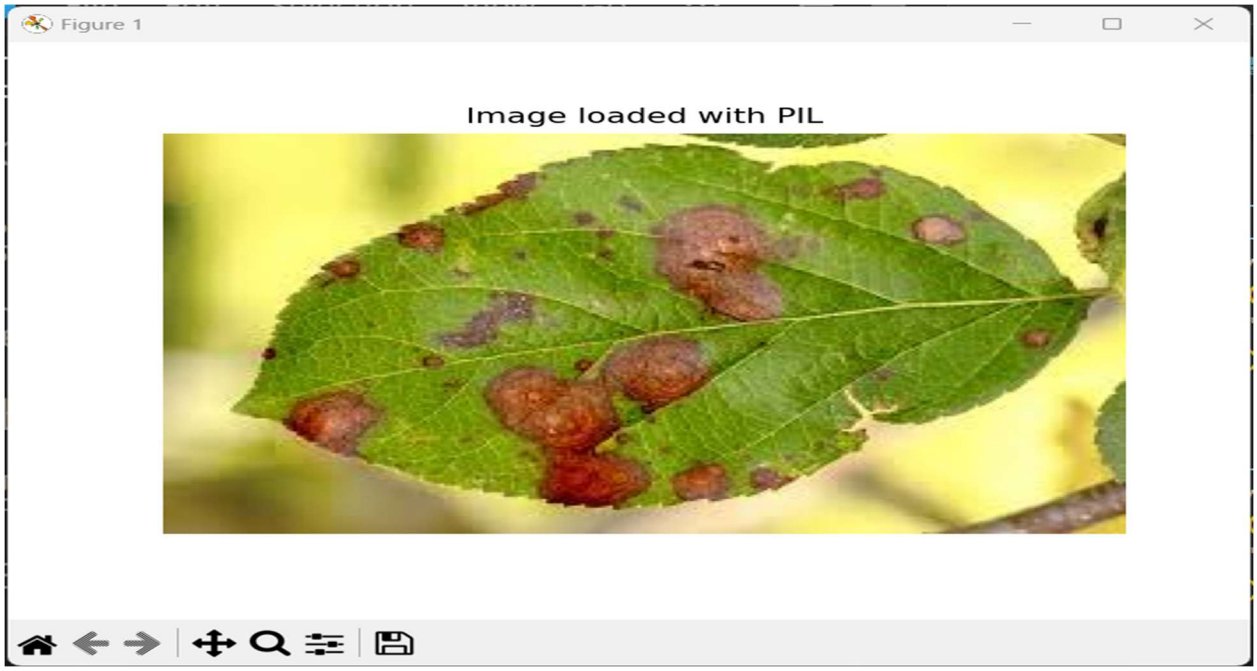


Figure 1
Image loaded with PIL


Figure 1
Image loaded with OpenCV

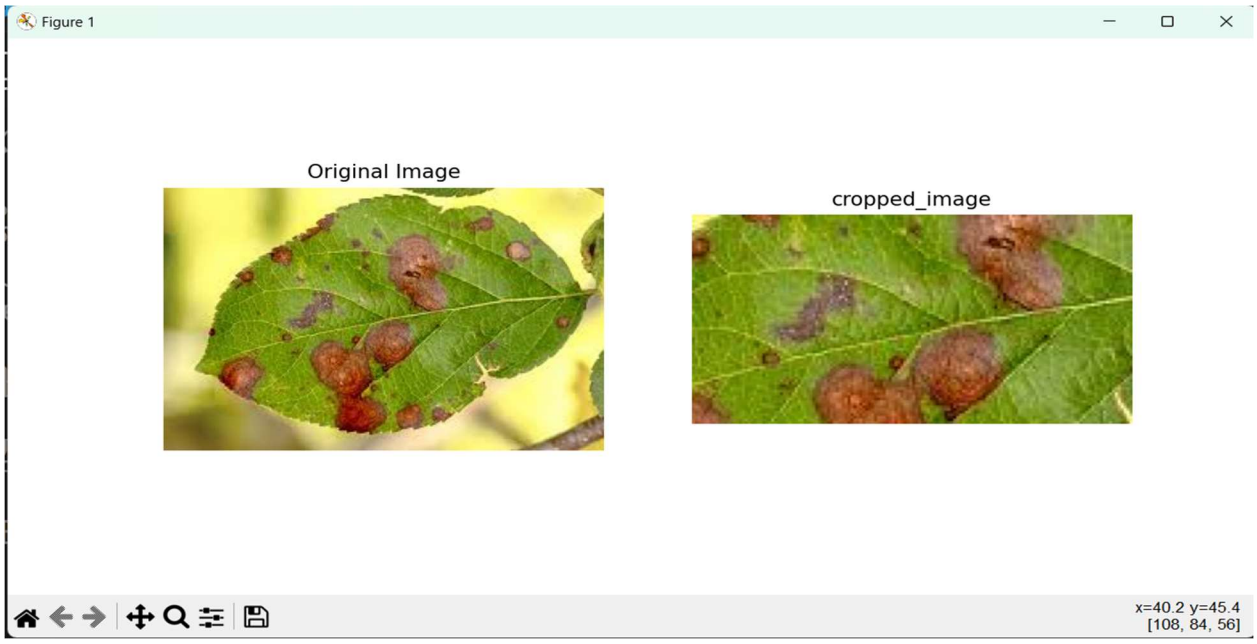

Figure 1
Image loaded with imageio
x=213.8 y=109.2
[119, 149, 35]

2. Loading_Image_Formats_Tutorial

```
import cv2
import matplotlib.pyplot as plt
# Load an image
image = cv2.imread('apple.jpeg')

# 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()

cropped_image = image_rgb[50:130, 50:200]
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 deg
(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))

# show 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:-



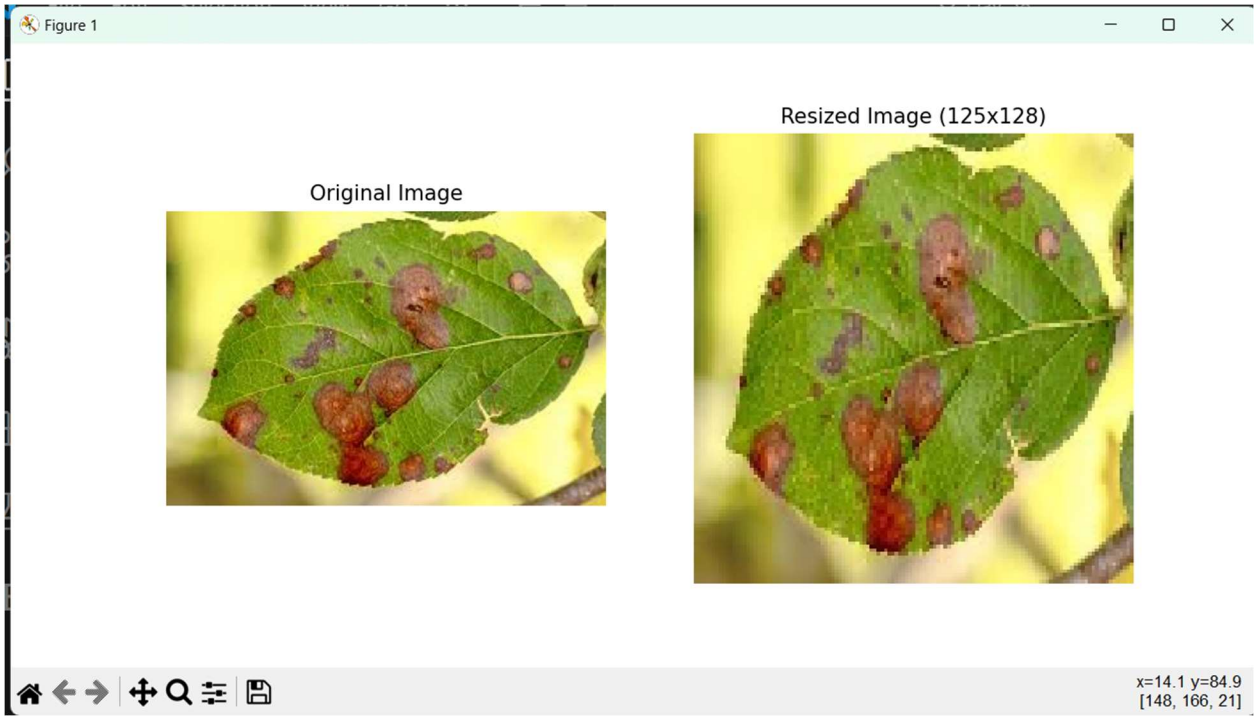


Figure 1
Original Image
Resized Image (125x128)
x=14.1 y=84.9
[148, 166, 21]

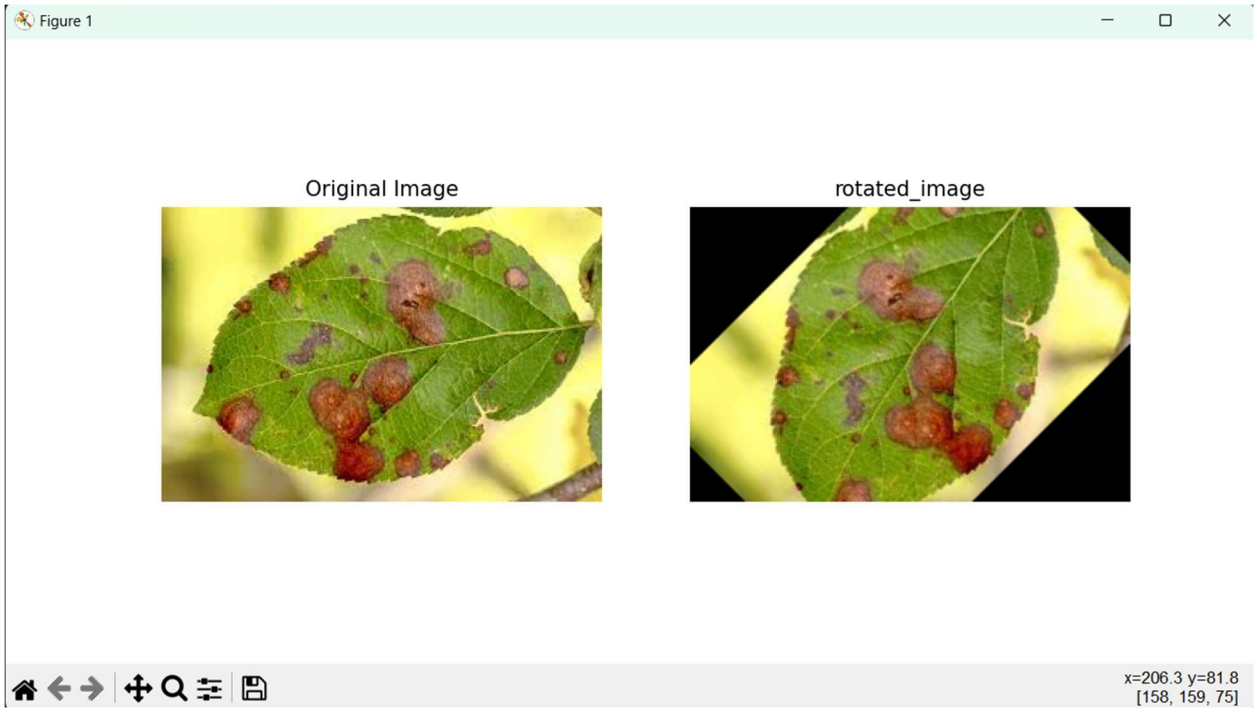


Figure 1
Original Image
rotated_image
x=206.3 y=81.8
[158, 159, 75]

3. Image Denoising

```
# import necessary libraries
import cv2
import matplotlib.pyplot as plt

# Load an image
image = cv2.imread('apple.jpeg')

# Convert the image from BGR to RGB
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:-

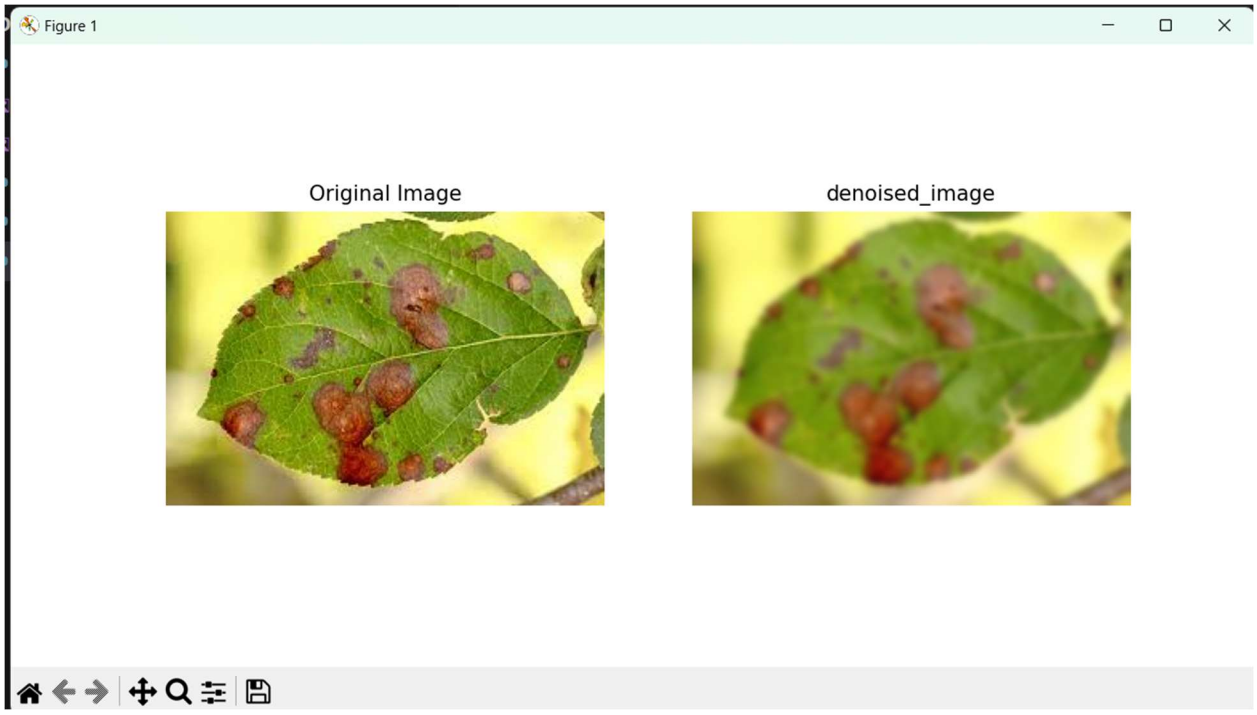



Figure 1
Gray Image
equalized_image