6CS012 Worksheet - 1

Image Processing with Python

2357606

Sanam Tamang

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
```

Exercise 1 - Working with Color Images

```
def exercise1(image_path):
    img = Image.open(image_path)
    plt.figure(figsize=(15, 10))
    plt.subplot(2, 3, 1)
    plt.imshow(img)
    plt.title('Original Image')

    img_array = np.array(img)
    top_left = img_array[:100, :100]
    plt.subplot(2, 3, 2)
    plt.imshow(top_left)
    plt.title('Top Left 100x100')

    r, g, b = img_array[:,:,0], img_array[:,:,1], img_array[:,:,2]

    plt.subplot(2, 3, 3)
    plt.imshow(r, cmap='Reds')
    plt.title('Red Channel')

    plt.subplot(2, 3, 4)
    plt.imshow(g, cmap='Greens')
    plt.title('Green Channel')

    plt.subplot(2, 3, 5)
    plt.imshow(b, cmap='Blues')
    plt.title('Blue Channel')

    modified_img = img_array.copy()
    modified_img[:100, :100] = 210
    plt.subplot(2, 3, 6)
    plt.imshow(modified_img)
    plt.title('Modified Top Left')

    plt.tight_layout()
    plt.show()
```

Exercise 2 - Working with Grayscale Images

```
def exercise2(image_path):
    # 1. Load and display grayscale image
    img = Image.open(image_path).convert('L')
    img_array = np.array(img)

    plt.figure(figsize=(15, 10))
    plt.subplot(2, 3, 1)
    plt.imshow(img_array, cmap='gray')
    plt.title('Grayscale Image')

    # 2. Extract middle section (150 pixels)
    h, w = img_array.shape
```

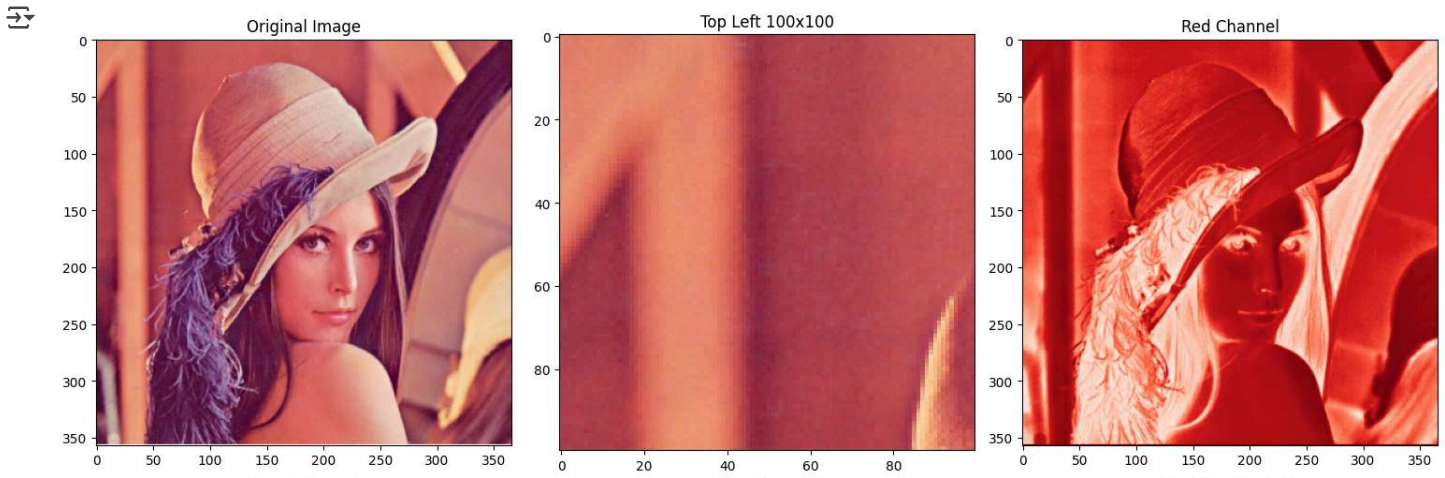
```
    center_h, center_w = h//2, w//2
    middle_section = img_array[center_h-75:center_h+75, center_w-75:center_w+75]
    plt.subplot(2, 3, 2)
    plt.imshow(middle_section, cmap='gray')
    plt.title('Middle Section (150px)')

    # 3. Apply threshold
    binary_img = img_array.copy()
    binary_img[binary_img < 100] = 0
    binary_img[binary_img >= 100] = 255
    plt.subplot(2, 3, 3)
    plt.imshow(binary_img, cmap='gray')
    plt.title('Thresholded Image')

    # 4. Rotate 90 degrees
    rotated_img = np.rot90(img_array, k=-1)  # k=-1 for clockwise
    plt.subplot(2, 3, 4)
    plt.imshow(rotated_img, cmap='gray')
    plt.title('Rotated Image')

    # 5. Convert to RGB
    rgb_img = np.stack([img_array]*3, axis=-1)
    plt.subplot(2, 3, 5)
    plt.imshow(rgb_img)
    plt.title('RGB Conversion')

    plt.tight_layout()
    plt.show()
```

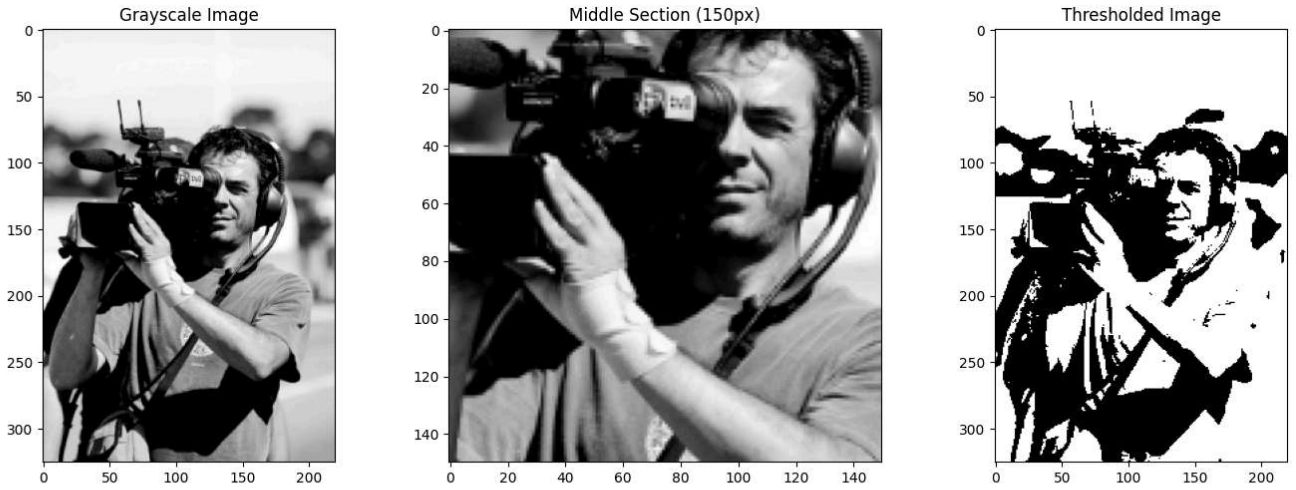
Exercise 3 - PCA Image Compression

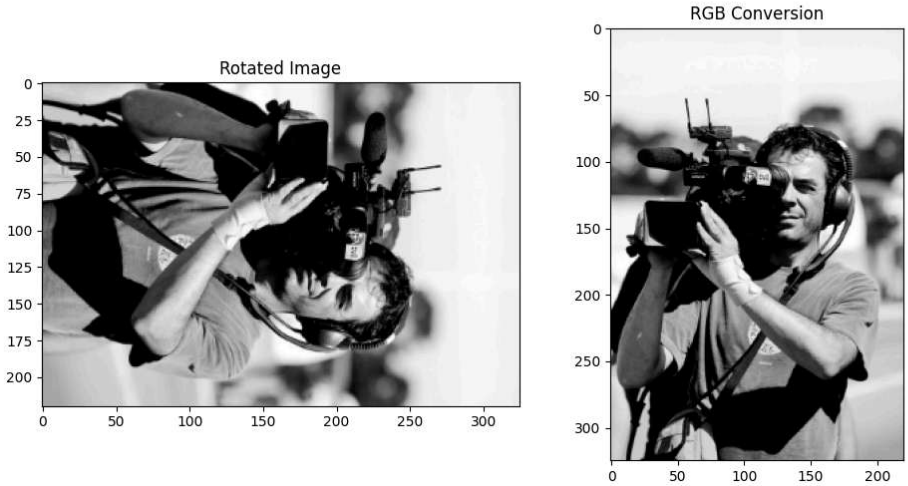
```
def exercise3(image_path):
    img = Image.open(image_path).convert('L')
    img_array = np.array(img, dtype=float)

    img_standardized = (img_array - np.mean(img_array)) / np.std(img_array)

    covariance_matrix = np.cov(img_standardized.T)

    eigenvalues, eigenvectors = np.linalg.eigh(covariance_matrix)

    idx = eigenvalues.argsort()[::-1]
    eigenvalues = eigenvalues[idx]
    eigenvectors = eigenvectors[:, idx]

    explained_variance_ratio = eigenvalues / np.sum(eigenvalues)
    cumulative_variance_ratio = np.cumsum(explained_variance_ratio)

    plt.figure(figsize=(10, 5))
    plt.plot(cumulative_variance_ratio)
    plt.title('Cumulative Explained Variance Ratio')
    plt.xlabel('Number of Components')
    plt.ylabel('Cumulative Explained Variance')
    plt.show()

    n_components_list = [5, 20, 50, 100]
    plt.figure(figsize=(15, 10))

    for i, n in enumerate(n_components_list, 1):
        pca_vectors = eigenvectors[:, :n]
        projected = img_standardized @ pca_vectors
        reconstructed = projected @ pca_vectors.T
        reconstructed = (reconstructed * np.std(img_array)) + np.mean(img_array)

        plt.subplot(2, 2, i)
        plt.imshow(reconstructed, cmap='gray')
        plt.title(f'Components: {n}\nVar Explained: {cumulative_variance_ratio[n-1]:.2%}')

    plt.tight_layout()
    plt.show()
```

Results:

```
if __name__ == "__main__":
    # Exercise 1 with color image
    exercise1('/content/drive/MyDrive/AI ML/lenna_image.png')

    # Exercise 2 with grayscale image
    exercise2('/content/drive/MyDrive/AI ML/camera_man.jpg')

    # Exercise 3 with grayscale image
    exercise3('/content/drive/MyDrive/AI ML/camera_man.jpg')
```







Cumulative Explained Variance Ratio