5/10/25, 10:18 AM LAB_1.py

LAB_1.py

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from skimage import io, color, util, img_as_float
4 from scipy.ndimage import convolve
   from scipy.ndimage import gaussian_filter
5
   from skimage.filters import gaussian
6
7
   from skimage.restoration import denoise bilateral
   from skimage.filters.rank import median
8
   from skimage.morphology import disk
9
   from skimage.util import random noise
10
11
   from skimage.filters import rank
12
13
   img = img_as_float(color.rgb2gray(io.imread(r'Low Contrast.jpg')))
14
15
    gauss_noise = random_noise(img, mode='gaussian', var=0.01)
    sp_noise = random_noise(img, mode='s&p', amount=0.05)
16
17
18
    avg_kernel = np.ones((3, 3)) / 9
19
   gauss_sigma = 0.5
20
21
   results = [
22
        (img, 'Original'),
23
        (gauss_noise, 'Gaussian Noise'),
24
        (sp noise, 'Salt & Pepper'),
        (convolve(gauss_noise, avg_kernel), 'Mean (Gauss)'),
25
26
        (convolve(sp_noise, avg_kernel), 'Mean (S&P)'),
27
        (median((gauss_noise * 255).astype(np.uint8), disk(1)) / 255.0, 'Median (Gauss)'),
        (median((sp_noise * 255).astype(np.uint8), disk(1)) / 255.0, 'Median (S&P)'),
28
        (gaussian(gauss_noise, sigma=gauss_sigma), 'Gauss (Gauss)'),
29
30
        (gaussian(sp_noise, sigma=gauss_sigma), 'Gauss (S&P)')
31
   1
32
33
   fig, axes = plt.subplots(3, 3, figsize=(12, 12))
34
   for ax, (image, title) in zip(axes.ravel(), results):
35
        ax.imshow(image, cmap='gray')
36
        ax.set_title(title)
37
        ax.axis('off')
   plt.tight_layout()
38
39
    plt.show()
40
```

5/10/25, 11:17 AM LAB_2.py

LAB_2.py

```
1 import numpy as np
   import matplotlib.pyplot as plt
2
3
   from skimage import io, color, img_as_float, exposure
4
5
   image = io.imread(r'Low Contrast.jpg')
6
   gray_image = color.rgb2gray(image)
7
   gray_image = img_as_float(gray_image)
8
9
    global_he = exposure.equalize_hist(gray_image)
   clahe = exposure.equalize_adapthist(gray_image, clip_limit=0.02, nbins=256)
10
11
   plt.figure(figsize=(12, 8))
12
   plt.subplot(3, 3, 1)
13
14
   plt.imshow(gray_image, cmap='gray')
   plt.title('Original')
   plt.axis('off')
16
17
   plt.subplot(3, 3, 2)
18
19
   plt.imshow(global_he, cmap='gray')
20
   plt.title('Global HE')
   plt.axis('off')
21
22
23
   plt.subplot(3, 3, 3)
24
   plt.imshow(clahe, cmap='gray')
25
    plt.title('CLAHE')
26
   plt.axis('off')
27
28
   plt.subplot(3, 3, 4)
   plt.hist(gray image.ravel(), bins=256, range=(0, 1), color='black')
29
   plt.title('Original Histogram')
30
31
32
   plt.subplot(3, 3, 5)
33
    plt.hist(global_he.ravel(), bins=256, range=(0, 1), color='black')
34
   plt.title('Global HE Histogram')
35
36
   plt.subplot(3, 3, 6)
37
    plt.hist(clahe.ravel(), bins=256, range=(0, 1), color='black')
   plt.title('CLAHE Histogram')
38
39
40
   # Step 4: Print contrast (standard deviation)
41
    print("Contrast (Standard Deviation):")
   print(f"Original: {np.std(gray_image):.4f}")
    print(f"Global HE: {np.std(global he):.4f}")
43
44
   print(f"CLAHE:
                        {np.std(clahe):.4f}")
45
46 plt.tight_layout()
47 plt.show()
```

Content Script

5/10/25, 11:21 AM LAB_3.py

LAB_3.py

```
1 import matplotlib.pyplot as plt
2
   from skimage import io, color, filters
3
   from skimage.filters import threshold otsu
4
   from skimage.filters import threshold_local
5
   image = io.imread(r'Low Contrast.jpg')
6
7
   if image.ndim == 3:
8
        gray_image = color.rgb2gray(image)
9
10
        gray_image = image
11
12
   otsu_thresh = threshold_otsu(gray_image)
13
   otsu_binary = gray_image > otsu_thresh
14
15
    adaptive_thresh = threshold_local(gray_image, block_size=35, offset=0.0)
16
    adaptive_binary = gray_image > adaptive_thresh
17
18
   plt.figure(figsize=(12, 4))
19
20
   plt.subplot(1, 3, 1)
   plt.imshow(gray_image, cmap='gray')
21
22
   plt.title('Original')
23
   plt.axis('off')
24
25
   plt.subplot(1, 3, 2)
26
   plt.imshow(otsu_binary, cmap='gray')
    plt.title("Otsu's Threshold")
27
28
   plt.axis('off')
29
   plt.subplot(1, 3, 3)
30
31
   plt.imshow(adaptive_binary, cmap='gray')
32
   plt.title('Adaptive Threshold')
33
   plt.axis('off')
34
35
   plt.tight_layout()
   plt.show()
36
37
```

5/10/25, 11:25 AM LAB_4.py

LAB_4.py

```
1 import matplotlib.pyplot as plt
   import numpy as np
2
3
   from skimage import io, color, filters, feature, img_as_float
4
5
   image = io.imread(r'Low Contrast.jpg')
   gray_image = color.rgb2gray(image)
6
7
   gray_image = img_as_float(gray_image)
8
9
    sobel_edges = filters.sobel(gray_image)
    prewitt_edges = filters.prewitt(gray_image)
10
11
    canny_edges = feature.canny(gray_image)
12
13
   plt.figure(figsize=(10, 8))
14
15
   plt.subplot(2, 2, 1)
   plt.imshow(gray_image, cmap='gray')
16
17
    plt.title('Original')
   plt.axis('off')
18
19
20
   plt.subplot(2, 2, 2)
   plt.imshow(sobel edges, cmap='gray')
21
22
   plt.title('Sobel')
23
   plt.axis('off')
24
25
   plt.subplot(2, 2, 3)
26
   plt.imshow(prewitt_edges, cmap='gray')
27
   plt.title('Prewitt')
28
   plt.axis('off')
29
30
   plt.subplot(2, 2, 4)
31
   plt.imshow(canny_edges, cmap='gray')
   plt.title('Canny')
32
33
   plt.axis('off')
34
35
   plt.tight_layout()
36
   plt.show()
37
38
   fig, ax = plt.subplots(1, 3, figsize=(12, 4))
   titles = ['Sobel', 'Prewitt', 'Canny']
39
    edges = [sobel_edges, prewitt_edges, canny_edges]
40
41
42
   for i in range(3):
        ax[i].imshow(edges[i], cmap='gray')
43
44
        ax[i].set_title(titles[i])
45
        ax[i].axis('off')
    plt.suptitle('Sobel | Prewitt | Canny', fontsize=14)
46
47
    plt.tight_layout()
   plt.show()
```

Content Script

5/10/25, 11:25 AM LAB_4.py

```
49
50
   sobel_density = 100 * np.count_nonzero(sobel_edges > 0) / sobel_edges.size
   prewitt_density = 100 * np.count_nonzero(prewitt_edges > 0) / prewitt_edges.size
51
52
    canny_density = 100 * np.count_nonzero(canny_edges) / canny_edges.size
53
54
   print("Edge Density:")
55
   print(f"Sobel: {sobel_density:.2f}%")
   print(f"Prewitt: {prewitt_density:.2f}%")
56
   print(f"Canny: {canny_density:.2f}%")
57
58
```

5/10/25, 11:26 AM LAB_5.py

LAB_5.py

```
1 import matplotlib.pyplot as plt
   import numpy as np
2
3 from skimage import io, color, filters, morphology, measure
   from skimage.filters import threshold_local
   from skimage.util import invert
5
6
7
   image = io.imread(r'Low Contrast.jpg')
8
   gray = color.rgb2gray(image)
9
   adaptive_thresh = threshold_local(gray, block_size=35, offset=0.1)
10
11
    binary = gray < adaptive_thresh</pre>
12
13
   cleaned = morphology.remove small objects(binary, min size=50)
14
15
   label_image = measure.label(cleaned)
    regions = measure.regionprops(label_image)
16
17
    regions = sorted(regions, key=lambda r: r.bbox[1])
18
19
20
   plt.figure(figsize=(10, 8))
   plt.subplot(2, 2, 1)
21
22
   plt.imshow(image)
23
   plt.title('Original Image')
   plt.axis('off')
24
25
26
   plt.subplot(2, 2, 2)
27
   plt.imshow(~binary, cmap='gray')
28
    plt.title('Binarized Image')
29
   plt.axis('off')
30
31
   plt.subplot(2, 2, 3)
   plt.imshow(cleaned, cmap='gray')
32
33
   plt.title('Cleaned Image')
34
   plt.axis('off')
35
36
   plt.subplot(2, 2, 4)
37
   plt.imshow(image)
   plt.title('Detected Characters')
38
39
   plt.axis('off')
   for region in regions:
40
41
        minr, minc, maxr, maxc = region.bbox
42
        rect = plt.Rectangle((minc, minr), maxc - minc, maxr - minr,
                             edgecolor='red', facecolor='none', linewidth=1)
43
44
        plt.gca().add_patch(rect)
   plt.tight_layout()
45
   plt.show()
46
47
48
   num_chars = len(regions)
```

```
49
   cols = 5
50
    rows = int(np.ceil(num_chars / cols))
51
52
    plt.figure(figsize=(15, rows * 2))
53
   for i, region in enumerate(regions):
        plt.subplot(rows, cols, i + 1)
54
55
        plt.imshow(region.image, cmap='gray')
56
        plt.title(f'{i+1}')
57
        plt.axis('off')
    plt.suptitle('Segmented Characters')
58
59
    plt.tight_layout()
60
    plt.show()
61
62
   print(f"Total characters detected: {num_chars}")
63
```

5/10/25, 11:58 AM LAB_6.py

LAB_6.py

```
1 import matplotlib.pyplot as plt
2
   import numpy as np
   from skimage import io, color, filters, measure, morphology
3
4
   from skimage.filters import threshold_local
5
6
   img = io.imread(r'rice (1).tiff')
7
   gray = color.rgb2gray(img)
8
9
   block size = 35
   adaptive_thresh = threshold_local(gray, block_size, offset=10)
10
    binary = gray < adaptive_thresh</pre>
12
    cleaned = morphology.remove_small_objects(binary, min_size=50)
13
14
   label_img = measure.label(cleaned)
15
   regions = measure.regionprops(label_img)
16
17
    plt.figure(figsize=(8, 8))
   plt.imshow(gray, cmap='gray')
18
19
   plt.title('Rice Grains')
20
   for i, region in enumerate(regions):
        y, x = region.centroid
21
22
        plt.text(x, y, str(i + 1), color='red', fontsize=8)
23
   plt.axis('off')
   plt.show()
24
25
26
   areas = np.array([r.area for r in regions])
27
    majors = np.array([r.major_axis_length for r in regions])
28
    perims = np.array([r.perimeter for r in regions])
29
   print(f"Total grains: {len(regions)}\n")
30
    print(f"{'No.':<5} {'Area':<7} {'MajorAxisLen':<15} {'Perimeter':<10}")</pre>
31
32
   for i in range(len(regions)):
33
        print(f"{i+1:<5} {areas[i]:<7.1f} {majors[i]:<15.1f} {perims[i]:<10.1f}")</pre>
34
35
   print(f"\nMin: {areas.min():.1f}, Max: {areas.max():.1f}, Mean: {areas.mean():.1f}")
36
    in_range = np.sum((areas >= 200) & (areas <= 400))</pre>
    print(f"Grains in area [200-400]: {in range}")
37
38
```

5/10/25, 12:02 PM LAB_7.py

LAB_7.py

```
1 import matplotlib.pyplot as plt
2
   import numpy as np
3
   from skimage import io, color, img_as_float
   from scipy.signal import convolve2d
5
6
   img = io.imread(r'Low Contrast.jpg')
7
   gray = color.rgb2gray(img)
8
   gray = img_as_float(gray)
9
10
   mask = np.ones((3, 3)) / 9
11
12
    print("Convolution Mask:")
13
   print(mask)
14
15
   conv_img = convolve2d(gray, mask, mode='same', boundary='symm')
16
17
   plt.figure(figsize=(10, 4))
18
19
   plt.subplot(1, 2, 1)
20
   plt.imshow(gray, cmap='gray')
21
   plt.title('Original')
22
   plt.axis('off')
23
24
   plt.subplot(1, 2, 2)
   plt.imshow(conv_img, cmap='gray')
25
26
   plt.title('Convolved')
   plt.axis('off')
27
28
29
   plt.tight_layout()
   plt.show()
30
31
```