An-Najah National University

Department of Computer Engineering

Digital Image Processing - 10636318

First Semester 2024/2025 – P2

OpenCV Project

Dr.Anas Toma

Student Name: Nassar Derrar Brahemeh

Registration Number: 12218619

Date: 10/01/2025

Project Objective

To execute a series of operations on an input image, including converting the image to grayscale, histogram equalization, brightness adjustment, applying filters, and performing tasks such as edge detection and region-growing algorithm. These operations are implemented using the OpenCV library and Python programming language.

- # Steps Executed

1. Read and show the input image.

```
path_to_image = r"C:\\Users\\s1221\\OneDrive\\Desktop\\nassar\\New folder\\nassar.jpg"
input_image = cv2.imread(path_to_image)
if input_image is None:
    print(" تعذر العثور على الصورة.")
    exit()


resized_image = reduce_image_size(input_image, 50)
cv2.imshow("Original Image", resized_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

functions used :

cv2.imread: Reads the image from the specified file path.

reduce_image_size: Resizes the image to a smaller scale to optimize performance.

cv2.imshow: Displays the resized original image in a window

التنقل بين الصور من خلال اغلاق الفريم الحالي

2. Convert the image to grayscale.

```
41
42  gray_image = cv2.cvtColor(resized_image, cv2.COLOR_BGR2GRAY)
43  cv2.imshow("Grayscale Image", gray_image)
44  cv2.waitKey(0)
45  cv2.destroyAllWindows()
```

functions used :

cv2.cvtColor: Converts the input image from BGR (color) to grayscale format.

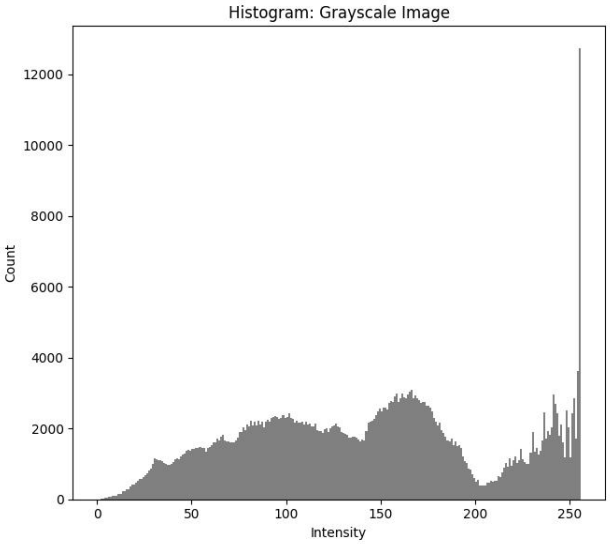
cv2.imshow: Displays the grayscale image in a window.


Grayscale Image
7-5827-98

3. Show the histogram of the grayscale image.

```
49    show_histogram_with_image(gray_image, "Grayscale Image")
```

functions used :



4. Perform histogram equalization and display the resulting histogram

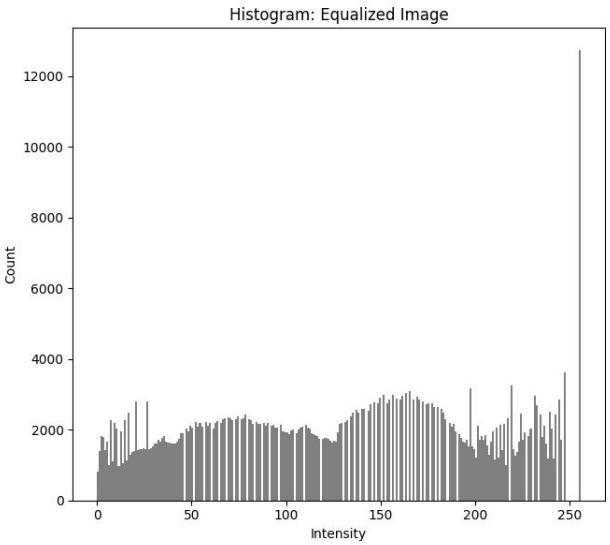
```
51    equalized_image = cv2.equalizeHist(gray_image)
52    show_histogram_with_image(equalized_image, "Equalized Image")
```

functions used :

cv2.equalizeHist: Applies histogram equalization to improve image contrast.

plt.hist: Displays the histogram of the equalized image.

plt.imshow: Shows the equalized image alongside its histogram



5. Modify the brightness of the grayscale image by applying the following equation

s = c*r

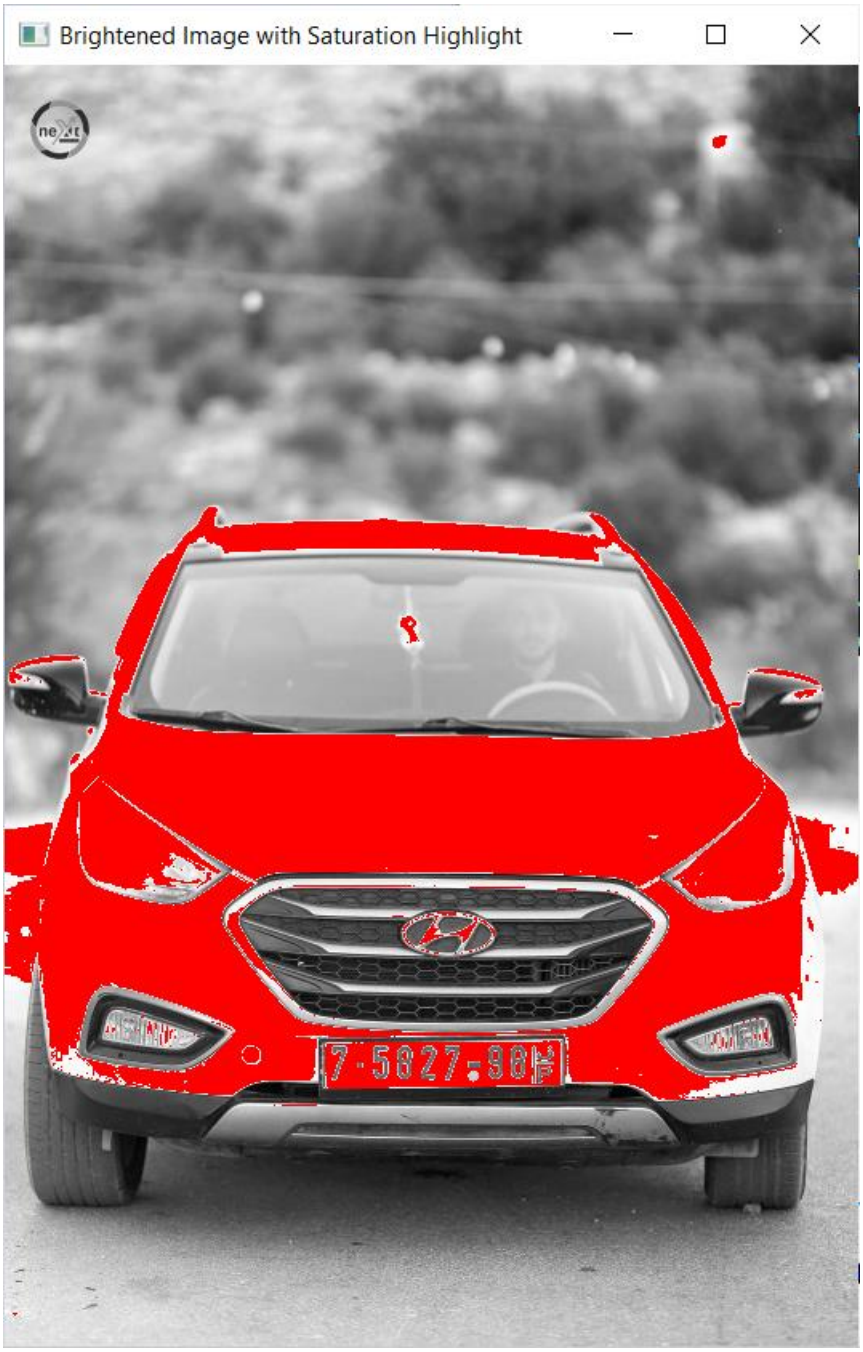
where:

• s is the output gray level.

• r is the input gray level.

• c is a random value where 0<c<2.

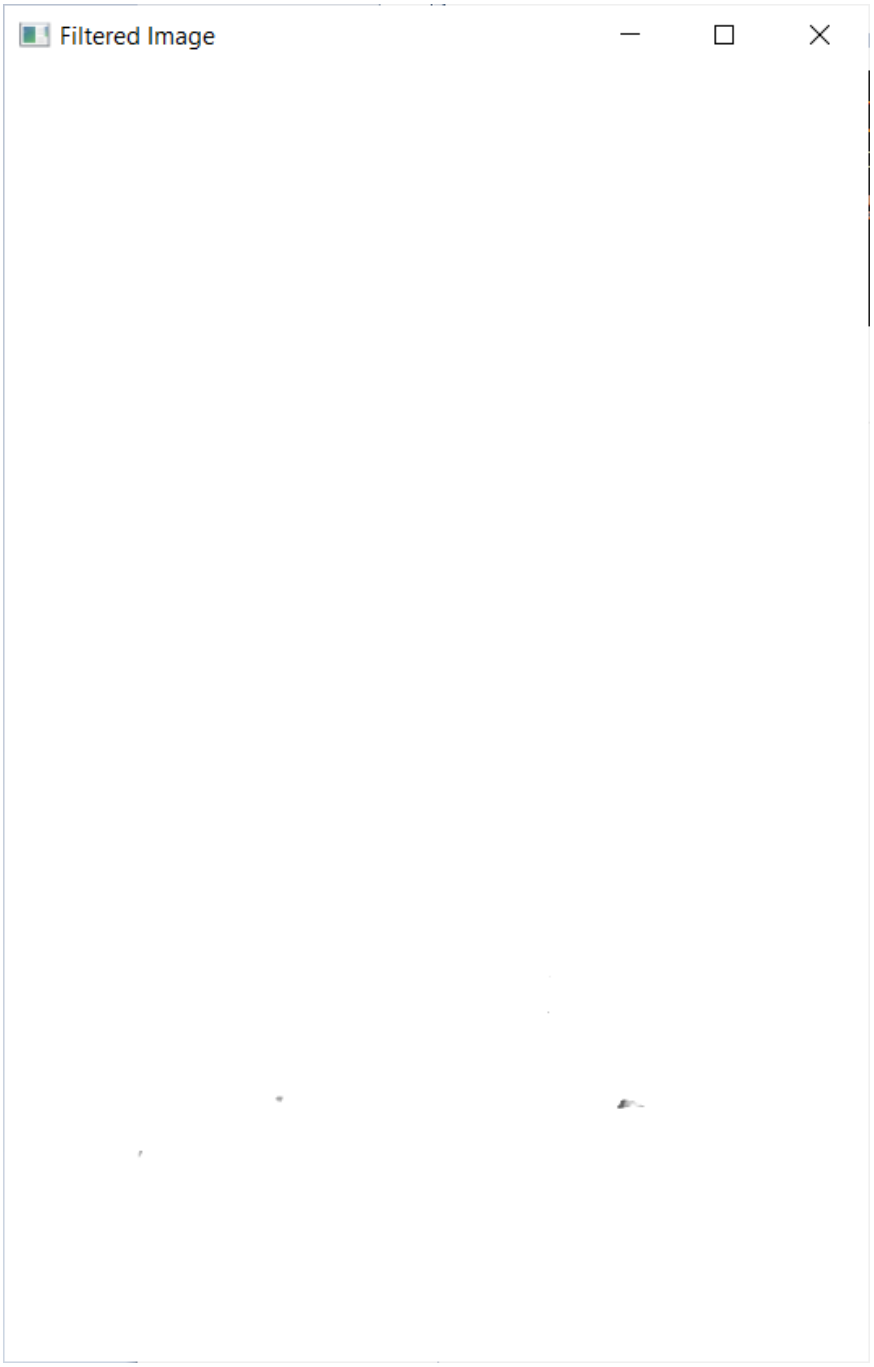
```
c = np.random.uniform(0, 2)
print(f"Value of c: {c}")

scaled = c * gray_image
modified_image = np.clip(scaled, 0, 255).astype(np.uint8)

saturated = (scaled > 255)

s = cv2.cvtColor(modified_image, cv2.COLOR_GRAY2BGR)
s[saturated] = [0, 0, 255]

cv2.imshow("Brightened Image with Saturation Highlight", s)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



6. Apply a 3x3 kernel to the grayscale image, using the 8 digits of your

university ID as the kernel coefficients (ordered left to right, top to bottom). Duplicate the last digit to obtain 9 coefficients. Display the output image after scaling to the [0-255] range.

```
64
65  student_id = "12218619"
66  kernel_vals = [int(num) for num in student_id] + [int(student_id[-1])]
67  kernel_matrix = np.array(kernel_vals[:9]).reshape((3, 3))
68  filtered_image = cv2.filter2D(gray_image, -1, kernel_matrix)
69  cv2.imshow("Filtered Image", filtered_image)
70  cv2.waitKey(0)
71  cv2.destroyAllWindows()
```

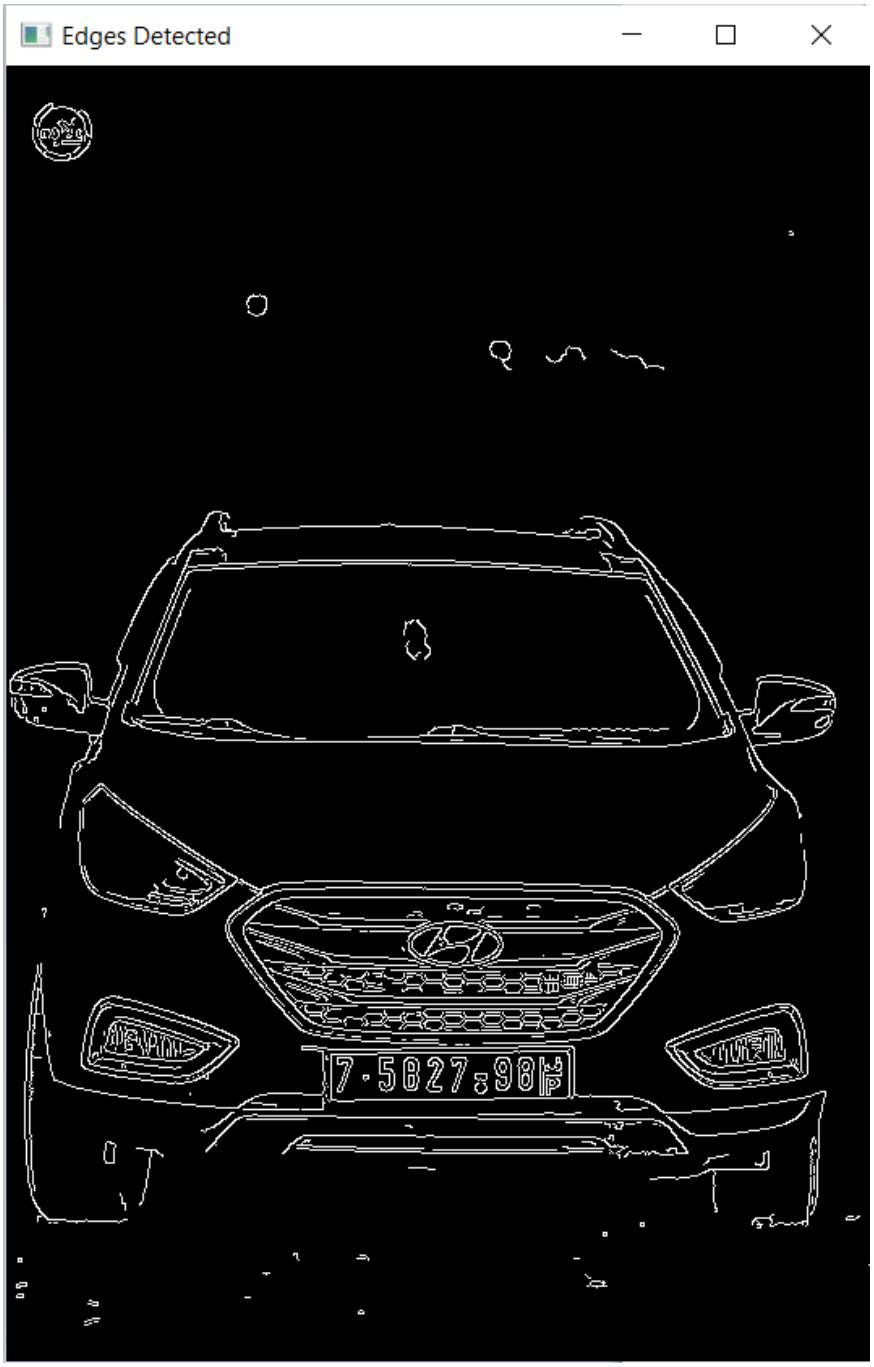


7. Perform a masking operation on the grayscale image using a mask of size 500x500 pixels, placed at coordinates (x, y), where x and y are random values.

```
73
74  height, width = gray_image.shape
75  x_start = random.randint(0, width - 500)
76  y_start = random.randint(0, height - 500)
77  mask = np.zeros_like(gray_image, dtype=np.uint8)
78  mask[y_start:y_start + 500, x_start:x_start + 500] = 255
79  masked_image = cv2.bitwise_and(gray_image, mask)
80  cv2.imshow("Masked Image", masked_image)
81  cv2.waitKey(0)
82  cv2.destroyAllWindows()
```

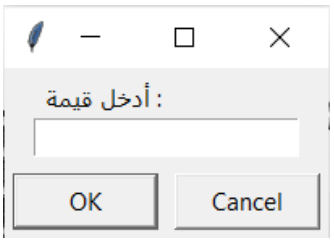
8. Display the grayscale image after applying edge detection

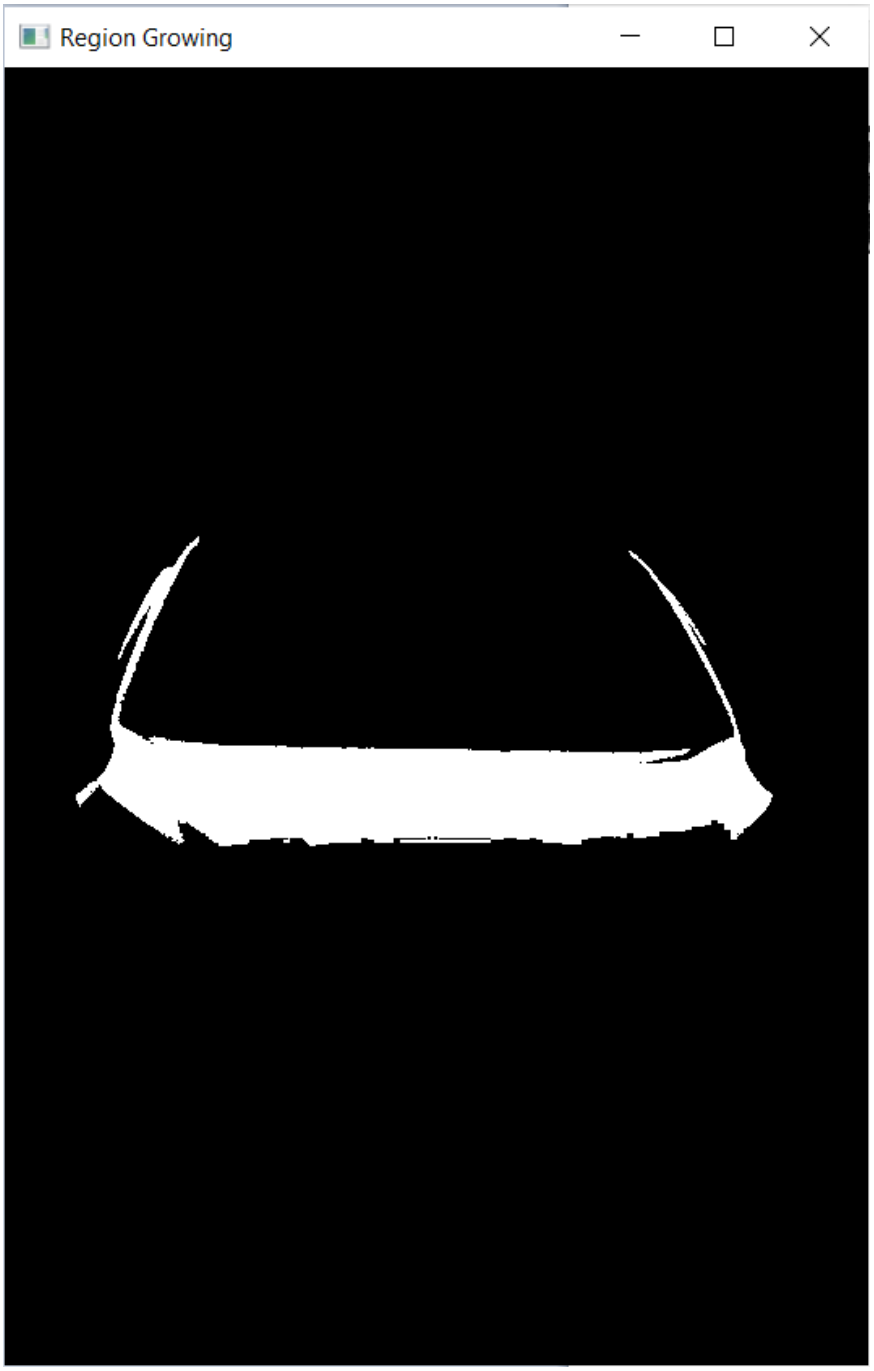
```
83
84  edges_detected = cv2.Canny(gray_image, 100, 200)
85  cv2.imshow("Edges Detected", edges_detected)
86  cv2.waitKey(0)
87  cv2.destroyAllWindows()
88
```



9. Implement a region growing algorithm that starts from a point selected with a mouse click. The range of gray levels should be determined by the user.

```python
def region_grow(image, seed, threshold):
    visited = np.zeros_like(image, dtype=bool)
    result = np.zeros_like(image, dtype=np.uint8)
    stack = [seed]
    seed_value = image[seed[1], seed[0]]

    while stack:
        x, y = stack.pop()
        if visited[y, x]:
            continue
        visited[y, x] = True

        if abs(int(image[y, x]) - int(seed_value)) <= threshold:
            result[y, x] = 255
            neighbors = [(x + dx, y + dy) for dx, dy in [(1, 0), (-1, 0), (0, 1), (0, -1)]
                         if 0 <= x + dx < image.shape[1] and 0 <= y + dy < image.shape[0]]
            stack.extend(neighbors)
    return result
def mouse_event(event, x, y, flags, param):
    if event == cv2.EVENT_LBUTTONDOWN:
        print(f"نقطة البداية: ({x}, {y})")
        try:
            root = Tk()
            root.withdraw()
            threshold = simpledialog.askinteger("Threshold Input", "أدخل قيمة :", minvalue=0, maxvalue=255)
            root.destroy()
            if threshold is not None:
                grown_area = region_grow(gray_image, (x, y), threshold)
                cv2.imshow("Region Growing", grown_area)
        except Exception as e:
            print(f"erore:{e}")

cv2.imshow("Select Seed Point", gray_image)
cv2.setMouseCallback("Select Seed Point", mouse_event)
cv2.waitKey(0)
cv2.destroyAllWindows()
```





functions used:

region_grow: Implements the region-growing algorithm:

Input: A seed point and threshold value.

Logic: Grows a region by checking pixel similarity within the threshold.

Output: A binary image showing the grown region.

cv2.setMouseCallback: Captures the user’s mouse click to select the seed point.

simpledialog.askinteger: Prompts the user to input the threshold value.

cv2.imshow: Displays the grown region.

Resources:

- OpenCV
- Stack overflow

Thank you