Assignment 7

Hemant Goyal (2k21/co/196)

Task 1 Perform Image Segmentation using K-Means Clustering with k=3.

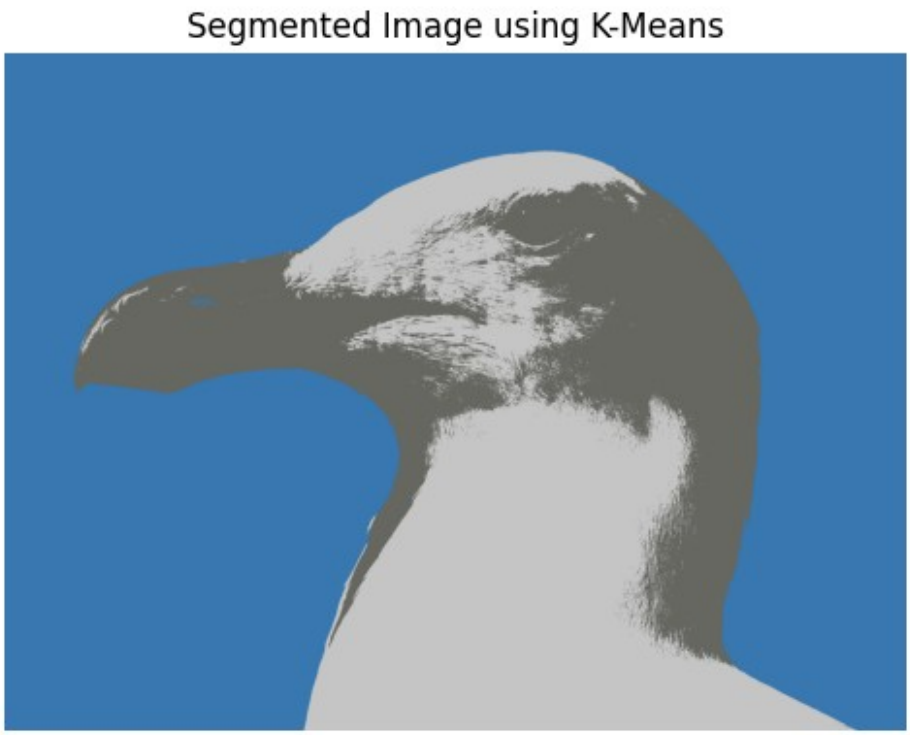
```
!pip install opencv-python-headless scikit-learn
```

```
Requirement already satisfied: opencv-python-headless in
/usr/local/lib/python3.10/dist-packages (4.10.0.84)
Requirement already satisfied: scikit-learn in
/usr/local/lib/python3.10/dist-packages (1.5.2)
Requirement already satisfied: numpy>=1.21.2 in
/usr/local/lib/python3.10/dist-packages (from opencv-python-headless)
(1.26.4)
Requirement already satisfied: scipy>=1.6.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
```

```
import cv2
import numpy as np
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

!wget -O sample_image.jpg
https://upload.wikimedia.org/wikipedia/commons/9/9a/Gull_portrait_ca_u
sa.jpg

# Step 4: Load the image
img = cv2.imread('sample_image.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

def show_img(image, title=''):
    plt.figure(figsize=(6,6))
    plt.imshow(image)
    plt.title(title)
    plt.axis('off')
    plt.show()

# 1. Image Segmentation using K-Means Clustering with k=3
pixel_values = img.reshape((-1, 3))
pixel_values = np.float32(pixel_values)
k = 3
kmeans = KMeans(n_clusters=k)
```

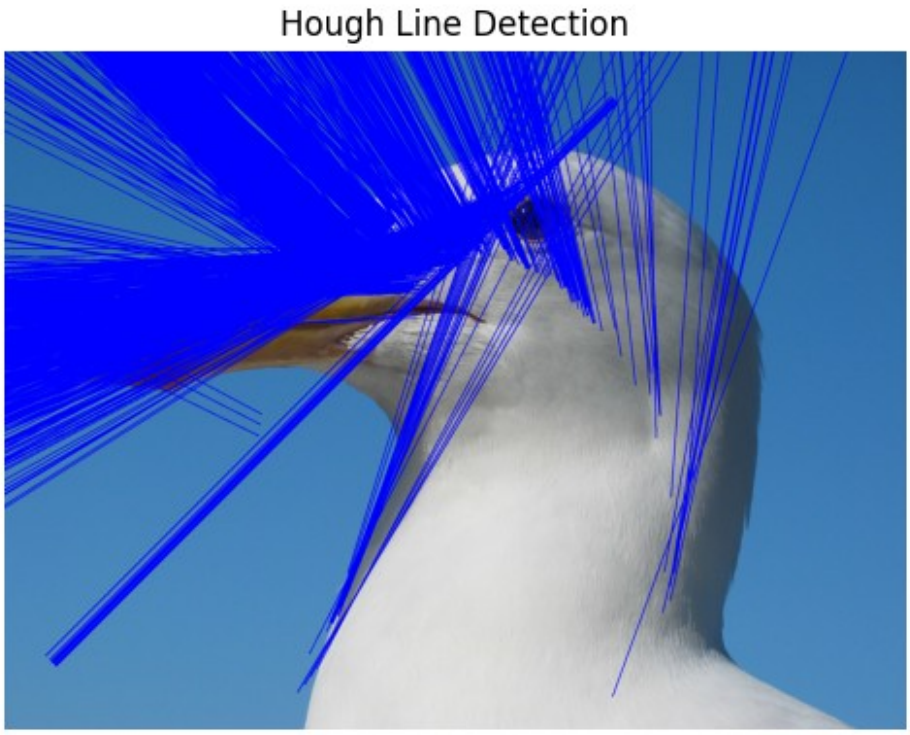
```
kmeans.fit(pixel_values)
labels = kmeans.labels_
segmented_img = kmeans.cluster_centers_[labels.flatten()]
segmented_img = segmented_img.reshape(img.shape)
segmented_img = np.uint8(segmented_img)

show_img(segmented_img, "Segmented Image using K-Means")
```

```
--2024-10-19 20:41:27--
https://upload.wikimedia.org/wikipedia/commons/9/9a/Gull_portrait_ca_usa.jpg
Resolving upload.wikimedia.org (upload.wikimedia.org)...
208.80.153.240, 2620:0:860:ed1a::2:b
Connecting to upload.wikimedia.org (upload.wikimedia.org)|
208.80.153.240|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 718902 (702K) [image/jpeg]
Saving to: ‘sample_image.jpg’

sample_image.jpg    100%[===================>] 702.05K  3.49MB/s    in
0.2s

2024-10-19 20:41:27 (3.49 MB/s) - ‘sample_image.jpg’ saved
[718902/718902]
```



Task 2 Hough Transform for line detection using OpenCV

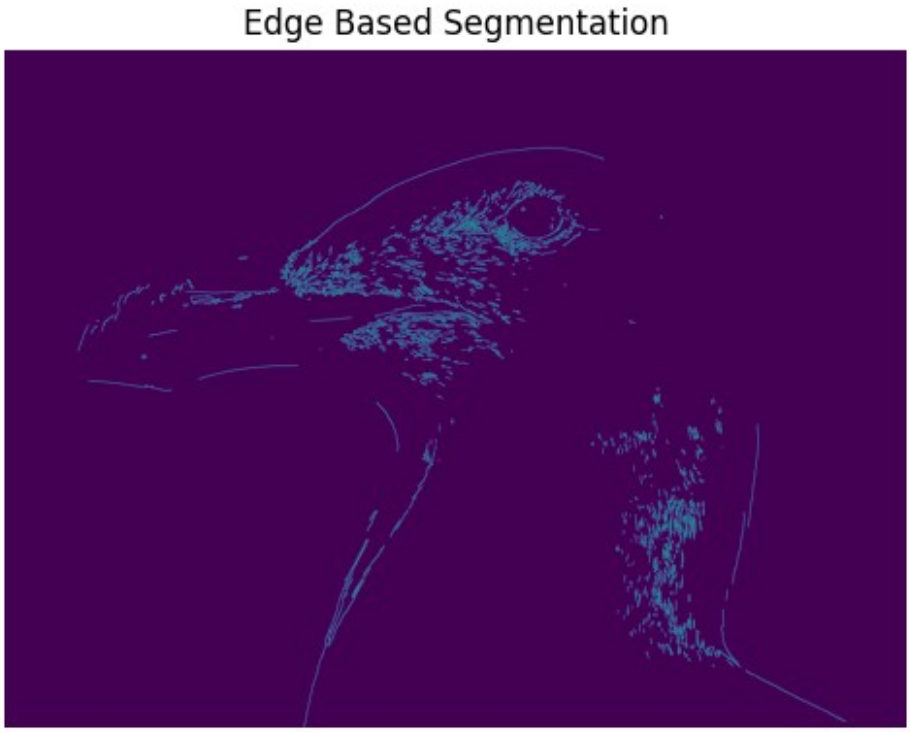
```
gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
edges = cv2.Canny(gray, 50, 150, apertureSize=3)
lines = cv2.HoughLines(edges, 1, np.pi / 180, 200)
line_img = img.copy()
if lines is not None:
    for rho, theta in lines[:, 0]:
        a = np.cos(theta)
        b = np.sin(theta)
        x0 = a * rho
        y0 = b * rho
        x1 = int(x0 + 1000 * (-b))
        y1 = int(y0 + 1000 * (a))
        x2 = int(x0 - 1000 * (-b))
        y2 = int(y0 - 1000 * (a))
        cv2.line(line_img, (x1, y1), (x2, y2), (0, 0, 255), 2)

show_img(line_img, "Hough Line Detection")
```



Task 3 Edge based Segmentation

```
edges = cv2.Canny(gray, 100, 200)
```

```
show_img(edges, "Edge Based Segmentation")
```



Task 4. Region based Segmentation

```
_, thresh = cv2.threshold(gray, 128, 255, cv2.THRESH_BINARY_INV)
contours, _ = cv2.findContours(thresh, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)
region_img = img.copy()
cv2.drawContours(region_img, contours, -1, (0, 255, 0), 2)
show_img(region_img, "Region Based Segmentation")
```

Region Based Segmentation