Ai and Ethics Lab -07

Name – Hitesh Choudhary

Prn - 20200802146

Title: Implementation of Image Segmentation Using Computer Vision

1. Introduction:

- Computer vision plays a pivotal role in interpreting and processing visual data. This report focuses on the implementation of an image segmentation task, which involves partitioning an image into distinct regions.

- The primary objective is to introduce the concept of image segmentation, explain the program's working, and provide insights into real-world applications.

2. Aim: The aim of this project is to implement image segmentation using Python in a variety of development environments, including Jupyter Notebook, IDLE, and other Python editors.

- This implementation will serve as a practical exercise in the field of computer vision and image processing.

3. Software Required:

- The software required for this project includes Jupyter Notebook for interactive development, IDLE for Python programming, and any Python editor of your choice.

- These tools are essential for coding, testing, and visualizing the results of the image segmentation program.

4. Prerequisite:

- Prior knowledge in problem-solving is essential for devising effective image segmentation solutions.

- Understanding transition operators and heuristic functions is crucial for implementing advanced image segmentation techniques.

5. Theory/Concept:

- Image segmentation is the process of dividing an image into multiple regions or objects based on certain characteristics.
- The concept involves using various algorithms and techniques to identify boundaries and group pixels with similar attributes together.

6. Program Explanation:

- The image segmentation program will be implemented in Python.
- It will use established computer vision libraries like OpenCV to perform the task.
- The program will follow a step-by-step process, including preprocessing, segmentation, and visualization.
- Problem-solving skills will be applied to handle challenges in the segmentation process.

7. Algorithm/Pseudo Code:

- The image segmentation program will employ algorithms such as k-means clustering or watershed segmentation.
- The algorithm chosen will depend on the specific requirements of the task.
- Pseudo code will be provided to outline the logic and structure of the program.

8. Relative Applications:

- Image segmentation is widely used in medical imaging to identify and isolate specific anatomical structures.
- In autonomous vehicles, it aids in recognizing objects and obstacles on the road.
- In agriculture, it is applied to assess crop health.
- Security and surveillance systems use image segmentation to detect and track objects of interest.

9. Output –

1. Read an Image using cv2 library.

+ Code + Markdown

```
import cv2
import numpy as np

#importing the opencv module
import cv2
from google.colab.patches import cv2_imshow
# using imread('path') and 1 denotes read as  color image
image = cv2.imread('veg.png',1)
print(image.shape)
cv2_imshow(image)
```

2. Chage the color space

```
## change color space

# Convert the image to Grayscale
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Save the Grayscale image
cv2.imwrite('gray_image.jpg', gray_image)
cv2_imshow(gray_image)
```

```
## Rotating the image
import cv2

# Load an image
image = cv2.imread('veg.png')


# Rotate the image by 45 degrees
angle = 60
# Change this to the desired rotation angle

# Get the height and width of the image
height, width = image.shape[:2]

# Calculate the rotation matrix
rotation_matrix = cv2.getRotationMatrix2D((width / 2, height / 2), angle, 1)

# Apply the rotation to the image
rotated_image = cv2.warpAffine(image, rotation_matrix, (width, height))

# Save the rotated image
cv2.imwrite('rotated_image.jpg', rotated_image)
cv2_imshow(rotated_image)
```

```python
#5. Image Segmentation
import cv2
import numpy as np

# Load the image
image = cv2.imread('image_seg.jpeg')
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Apply thresholding to create a binary image
_, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)

# Morphological operations to clean the image
kernel = np.ones((3, 3), np.uint8)
opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=2)
sure_bg = cv2.dilate(opening, kernel, iterations=3)

# Finding sure foreground area
dist_transform = cv2.distanceTransform(opening, cv2.DIST_L2, 5)
_, sure_fg = cv2.threshold(dist_transform, 0.7 * dist_transform.max(), 255, 0)

# Finding unknown region
sure_fg = np.uint8(sure_fg)
unknown = cv2.subtract(sure_bg, sure_fg)

# Marker labeling
_, markers = cv2.connectedComponents(sure_fg)

# Add one to all labels so that sure background is not 0, but 1
markers = markers + 1
```

10. Conclusion:

- This lab report provides an overview of image segmentation, its implementation in Python, and its real-world applications.
- The project aims to enhance our understanding of computer vision and its practical use.