Experiment- 3

Student Name: Kush Bhasin
UID: 20BCS7677
Branch: BE(CSE)
Section/Group: 20BCS-1/B
Semester: 7th
Date of Performance: 20/09/23
Subject Name: Computer Vision Lab
Subject Code: 20CSP- 422

1. **Aim:** Write a program to analyze the impact of refining feature detection for image segmentation.

2. **System Requirements:**
   - Python 3.9
   - Jupyter Notebook
   - Visual Studio Code

3. **Description:**

Refining feature detection for image segmentation is crucial for enhancing accuracy and quality. Various techniques are commonly employed:

i. **Multi-scale Analysis:** Detecting features at multiple scales using methods like SIFT or SURF enables capturing details of different sizes, improving overall segmentation accuracy.
ii. **Non-Maximum Suppression:** Reduces redundancy by selecting the most prominent feature responses and eliminating weaker ones, reducing false positives.
iii. **Adaptive Thresholding:** Dynamically adjusts detection thresholds based on local image characteristics, accommodating varying lighting conditions and contrasts.
iv. **Edge Refinement:** Enhances edges detected by techniques like Canny or gradient-based methods, improving their accuracy, continuity, and connectivity.
v. **Feature Filtering and Selection:** Eliminates irrelevant or redundant features, improving the quality and relevance of detected features.
vi. **Feature Fusion:** Integrates information from multiple feature types or descriptors, offering a more comprehensive representation of image content.
vii. **Contextual Information:** Leverages relationships between neighboring pixels or features, incorporating spatial constraints and semantic cues for refined detection.
viii. **Deep Learning-based Approaches:** Utilizes deep learning models like CNNs, capable of learning complex patterns and contextual information directly from data, thereby improving feature detection.

4. Steps:

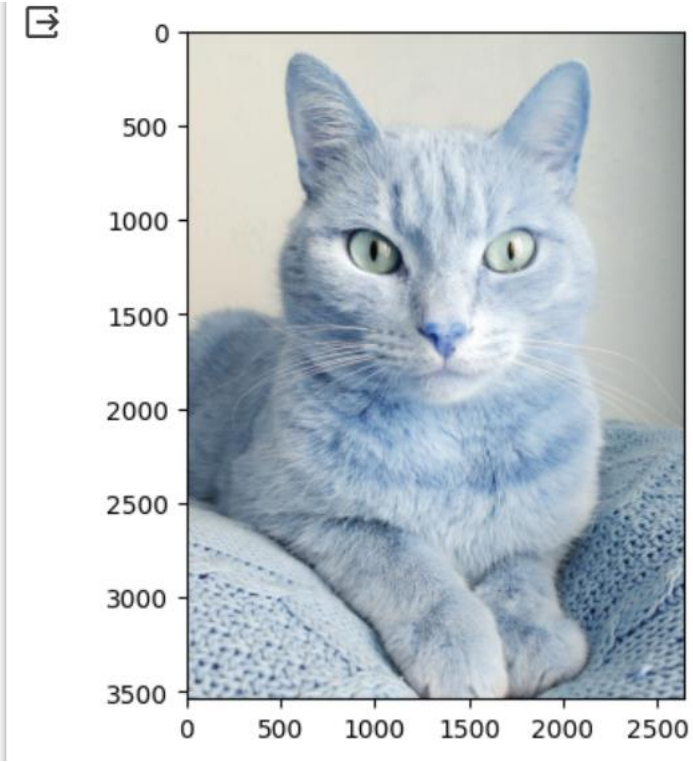
i. Import necessary Libraries
ii. Load the image
iii. Convert the image to grayscale
iv. Refine feature detection
v. Perform Segmentation
vi. Display the result

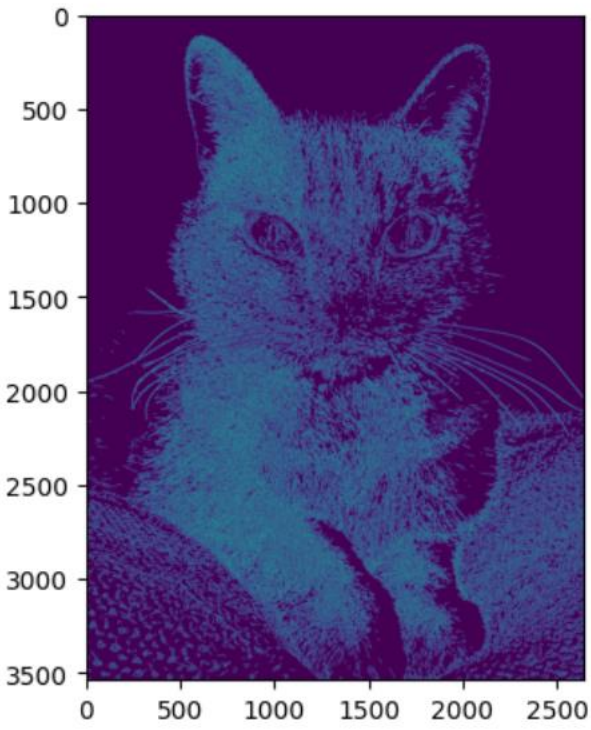
5. Code:

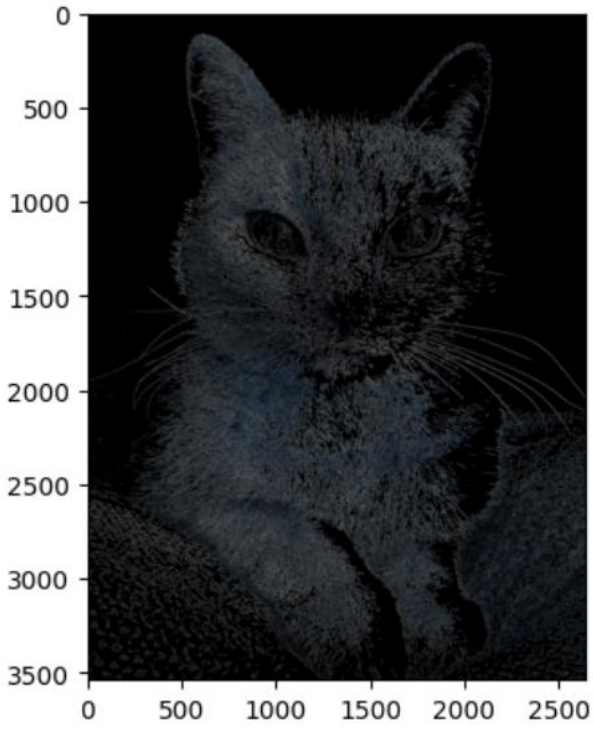
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

def refine_feature_detection(cat):
# Convert the image to grayscale
   gray_cat = cv2.cvtColor(cat, cv2.COLOR_BGR2GRAY)
   blurred = cv2.GaussianBlur(gray_cat, (5, 5), 0)
   # Apply Canny edge detection
   edges = cv2.Canny(blurred, threshold1=30, threshold2=70)
   return edges

def segment_image(panda, edges):
   # Perform image segmentation using the edges
   segmented_image = cv2.bitwise_and(panda, panda, mask=edges)
   return segmented_image

# Load the image
original_image = cv2.imread("cat.jpg")

# Refine feature detection
refined_edges = refine_feature_detection(original_image)

# Perform segmentation
segmented_result = segment_image(original_image, refined_edges)

# Display original image and segmented result
plt.imshow(original_image)
plt.show()
plt.imshow(refined_edges)
```

```
plt.show()
plt.imshow(segmented_result)
plt.show()
```

6. Output:




0
500
1000
1500
2000
2500
3000
3500
0
500
1000
1500
2000
2500


0
500
1000
1500
2000
2500
3000
3500
0
500
1000
1500
2000
2500