VR Assignment 1: Coin Detection and Image Stitching

Aryan Singhal
IMT2022036

February 23, 2025

1 Introduction

- **Part 1**: Detecting, segmenting, and counting coins from an image.
- **Part 2**: Creating a stitched panorama from multiple overlapping images.

The methods employed leverage OpenCV and NumPy for image processing and Matplotlib for visualization. This document details the steps taken and the outcomes achieved.

—

2 Part 1: Coin Detection, Segmentation, and Counting

Figure 1: Original Image taken as input for Part 1.

2.1 Edge Detection using Canny Edge Detector

Purpose: To detect the edges of objects in the image. **Key Idea:** Uses gradient-based detection to highlight abrupt intensity changes. **Steps:**

1. Convert the image to grayscale.
2. Apply **CLAHE (Contrast Limited Adaptive Histogram Equalization)** to enhance contrast.
3. Use **Gaussian Blur** to remove noise.
4. Apply the **Canny Edge Detector** to highlight object boundaries.

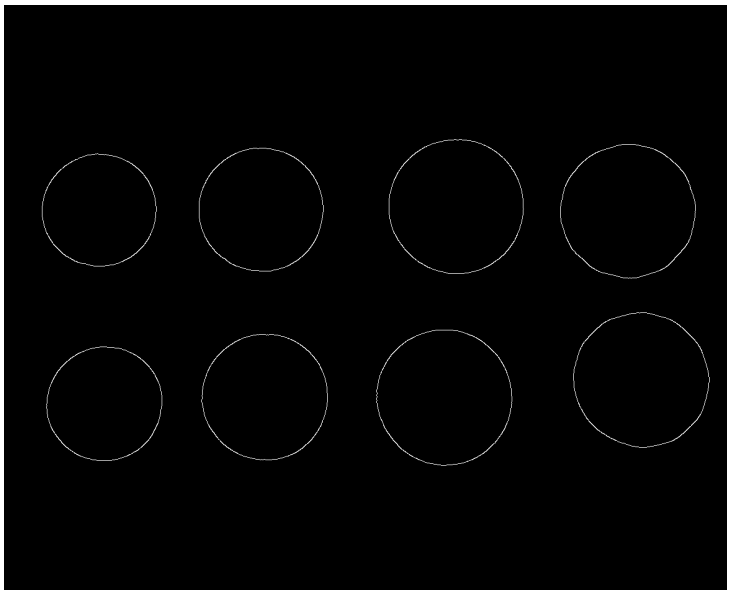

Figure 2: Edge detection using Canny algorithm

—

2.2 Segmentation using Watershed Algorithm

Purpose: To separate overlapping objects in an image. **Key Idea:** Uses morphological transformations and distance transformation to separate objects. **Steps:**

1. Convert image to grayscale and apply CLAHE for contrast enhancement.
2. Apply Gaussian blur to reduce noise.
3. Perform morphological transformations for noise removal.
4. Compute **distance transformation** to identify object centers.
5. Apply the **Watershed Algorithm** for segmentation.Figure 3 is obtained as the result.
6. Using the markers obtained from above extract each coin separately.Individual photos of each coin is part of the zip file.A collage of them can be seen in Figure 4

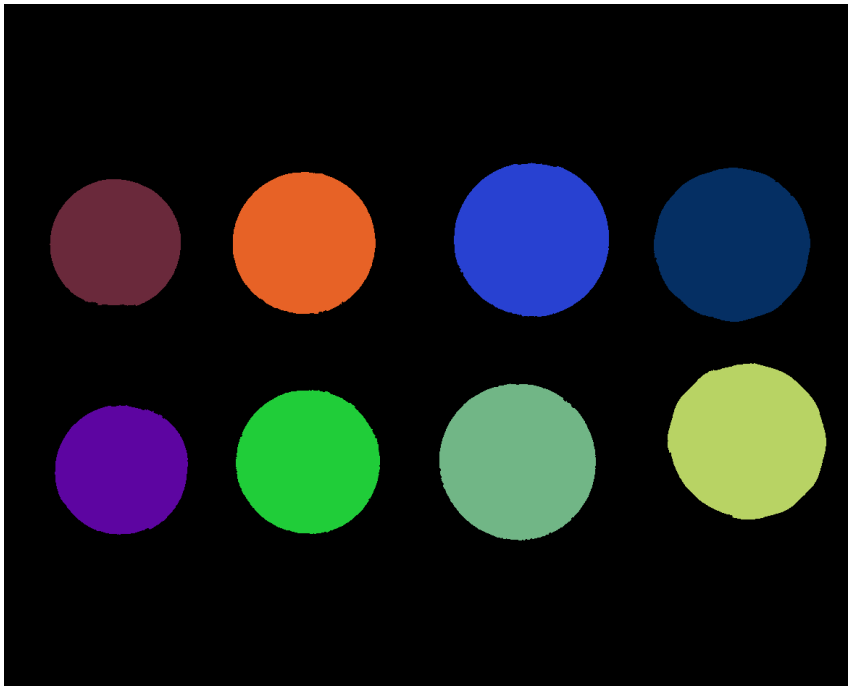

Figure 3: Segmented coins using the Watershed Algorithm

Figure 4: Collage of points extracted after segmentation.

—

2.3 Counting Coins using Hough Circle Transform

Purpose: To detect circular objects within an image. **Key Idea:** Uses parametric space mapping to detect circles. **Steps:**

1. Convert image to grayscale and apply median blur.
2. Use **Hough Circle Transform** to detect circular shapes.
3. Draw circles around detected coins.

Figure 5: Detected coins using the Hough Transform. 8 coins have been detected

—

3 Part 2: Image Stitching

3.1 Original Images

Input Image 1

Input Image 2

Input Image 3

Figure 6: Original input images for panorama stitching

3.2 Steps for Image Stitching

1. Resize the input images to reduce resolution.
2. Detect keypoints in images using **SIFT**.
3. Match features between adjacent images.
4. Compute homography matrix using **RANSAC**.
5. Apply perspective transformation to align images.
6. Merge images smoothly using **Mask Blending**.
7. Trim black regions to enhance output quality.

Figure 7: Keypoints detected using SIFT

Figure 8: Final Stitched Panorama

4 Directory Structure

The following directory structure is used:

```
IMT2022036_Aryan/
 input/
    Part_1/
       coins.jpeg
    Part_2/
       1.jpg
       2.jpg
       3.jpg
 output/
    Part_1/
       coin_1.png,coin_2.png,...
       canny_edges.png
       segmented_image.png
       count_coins.png
       extracted_coins.png
```

```
   Part_2/
      keypoints_detected.jpg
      panorama_output.jpg
.gitattributes
Part_1.py
Part_2.py
IMT2022036_Aryan.pdf
```

5 Instructions to Run

1. Install dependencies:

   ```
   pip install opencv-python numpy matplotlib
   ```

2. Open the current folder in terminal

3. Run Part 1:

   ```
   python Part_1.py
   ```

4. Run Part 2:

   ```
   python Part_2.py
   ```