AIM 825 Visual Recognition: Assignment 1

Praveen Peter Jay
IMT2022064

March 1, 2025

1 Introduction

This report documents the implementation and results of two computer vision tasks: coin detection and image stitching. The goal was to apply image processing techniques to detect coins in images and stitch multiple images into a panorama.

2 Coin Detection and Counting

2.1 Approach

The coin detection algorithm processes images as follows:

1. Convert the input image to grayscale to simplify processing.
2. Apply Gaussian blur to reduce noise and smooth the image.
3. Use edge detection techniques to identify boundaries of coins.
4. Find and draw contours around detected objects.
5. Segment individual coins and save them as separate images.
6. Count the number of detected coins.
7. Verify that the detected objects are coins using shape analysis and size filtering.

2.2 Edge Detection Techniques

Edge detection is crucial in identifying coin boundaries. Three different methods were applied:

- **Sobel Operator:** Computes the gradient magnitude in the x and y directions, emphasizing regions with high-intensity changes.
- **Laplacian Operator:** A second-order derivative method that detects rapid intensity variations.
- **Canny Edge Detector:** A multi-stage algorithm involving Gaussian filtering, gradient intensity computation, non-maximum suppression, and hysteresis thresholding for optimal edge detection.

2.3 Object Identification: Distinguishing Coins

To differentiate coins from other objects, the algorithm applies:

- **Circular Hough Transform (CHT):** The Circular Hough Transform is used to detect circular objects within the image. This method works by transforming image edges into a parameter space where circular shapes can be detected based on their radius and center position. The algorithm finds peaks in this space that correspond to potential circular objects.
- **Contour Area and Circularity Check:** Not all circular shapes detected are necessarily coins. Some may be reflections, overlapping objects, or other circular items. To filter out incorrect detections, we use contour analysis:
  - The area of the detected contour is compared to the expected area of a coin.

- The circularity of the object is determined using the formula:

$$C = \frac{4\pi A}{P^2} \tag{1}$$

where A is the area and P is the perimeter of the contour. A value close to 1 indicates a perfect circle, which helps filter out non-circular objects.

- **Size Filtering:** Coins typically fall within a known range of sizes. Objects that are significantly smaller or larger than expected are likely not coins and are removed. The algorithm uses a predefined size threshold based on the average radius of real-world coins.

- **Hierarchy Filtering:** To ensure individual coin detection in cases where coins are overlapping, we use contour hierarchy analysis. This prevents counting the same coin multiple times or misidentifying merged objects.

2.4 Results

The result of operations on `image-3` has been included in this report. Other results can be viewed in `Part-1/Output`.

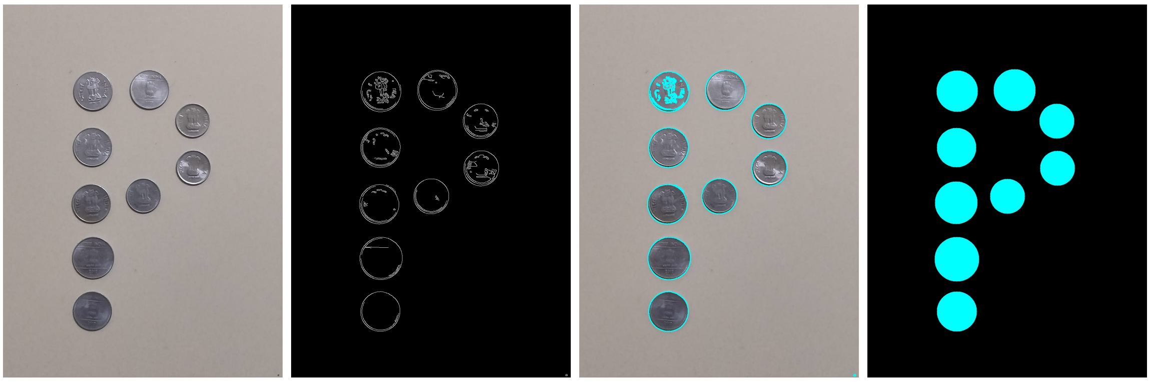

Figure 1: [Left-to-right] Original image, Image after applying canny edge detector, Detected coins, Segmented Coins

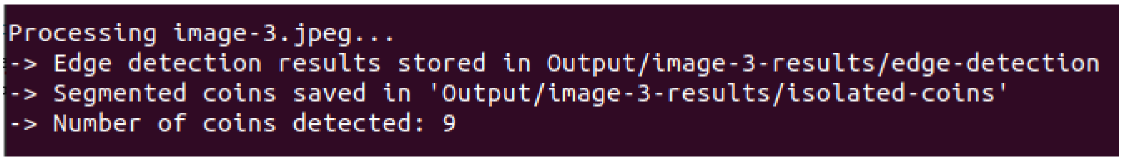

Figure 2: Output on terminal after execution

3 Image Stitching

3.1 Approach

The image stitching algorithm works as follows:

1. Detect keypoints in images using the Scale-Invariant Feature Transform (SIFT) algorithm.
2. Match keypoints between overlapping images using a FLANN-based matcher.
3. Use OpenCV's built-in `Stitcher` to automatically align and merge images.
4. If automatic stitching fails, manually stitch images using homography estimation.
5. Crop black borders to refine the final stitched panorama.

3.2 Feature Detection and Matching

Feature detection is performed using **SIFT**, which identifies keypoints and descriptors that are invariant to scaling, rotation, and illumination changes. Matching these features across images is done using the **FLANN-based matcher**, which efficiently finds correspondences between keypoints.

3.3 Homography and Image Warping

If automatic stitching fails, homography estimation is used to align images. Homography is a transformation matrix computed from matched keypoints. The transformation is applied using:

$$p' = Hp \tag{2}$$

where H is the homography matrix, p is a point in one image, and p' is its corresponding point in another image. Images are then warped using perspective transformation to achieve alignment.

3.4 Results

For keypoints of all images, refer `Part-2/Output/Set-1`.

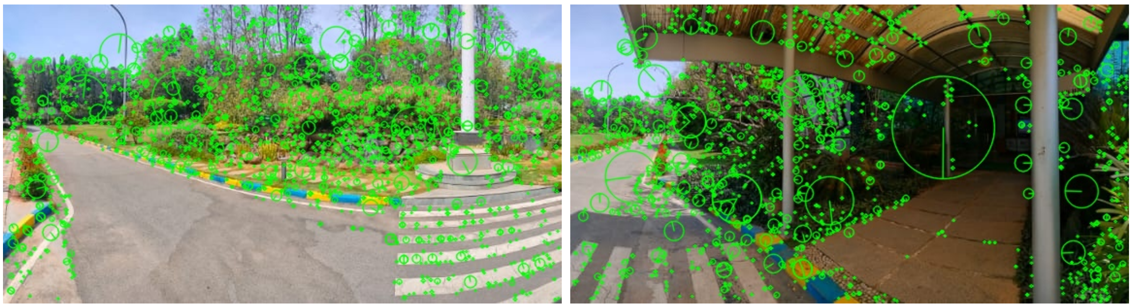

Figure 3: [Left-to-right] Keypoints detected in `image-1` and `image-5` of `Set-1`

Figure 4: Stitched panoramic image

4 Repository

https://github.com/PraveenPeterJay/VR_Assignment1_PraveenPeterJay_IMT2022064.git