AIM 825-Visual Recognition-Sec-B Assignment-1

Uttam Hamsaraj - IMT2022524

GitHub Repo: https://github.com/uttam24uttam/VR_Assignment1_UttamHamsaraj_IMT2022524

February 21, 2025

TASKS

This assignment consists of two parts:

- **Part 1: Coin Detection and Segmentation** – Detecting, segmenting, and counting coins from an image containing scattered Indian coins.
- **Part 2: Panorama Creation** – Creating a stitched panorama from multiple overlapping images.

Prerequisites

Install Python 3.10 . The following Python libraries are required: `cv2`, `numpy`, `matplotlib`, and `imutils`.

Install all dependencies using: `pip install opencv-python numpy matplotlib imutils`.

Repository Structure

```
VR_Assignment1_UttamHamsaraj_IMT2022524/
 PART-1/
    coins.py                # The code
    coin_image/             # The input image
    output_images/          # Created when the script is run, contains all output images

 PART-2/
    images/                 # Folder containing all input images
    image_stitching.py      # The code
    output_images/          # Created when the script is run, contains all output images
```

Steps to Run

1. Clone this repository:

   ```
   git clone https://github.com/uttam24uttam/VR_Assignment1_UttamHamsaraj_IMT2022524.git
   cd VR_Assignment1_UttamHamsaraj_IMT2022524
   ```

2. To run **Part 1** (Coin Detection):

   ```
   cd PART_1
   python3 coins.py
   ```

3. To run **Part 2** (Panorama Creation):

   ```
   cd PART_2
   python3 image_stitching.py
   ```

Part 1: Coin Detection and Segmentation

Input Image

Figure 1: Input coin image

Methods Used

- **Preprocessing:** The image is converted to grayscale and Gaussian Blur is applied to reduce noise and smooth the image.
- **Edge Detection:** Canny Edge Detection is used to detect the edges. Otsu's thresholding is used to create a binary image for better edge contrast. Morphological closing is used to refine object boundaries by filling small gaps.
- **Segmentation:** : Region-based segmentation using contour detection extracts individual coin regions.
- **Contours Detection:** `findContours()` is applied to the edge-detected image to extract shape outlines. This helps distinguish coins from background noise.
- **Counting:** Contours are filtered using an area threshold to exclude small artifacts. The valid contours (coins) are counted to get the total number of coins.

Output Images

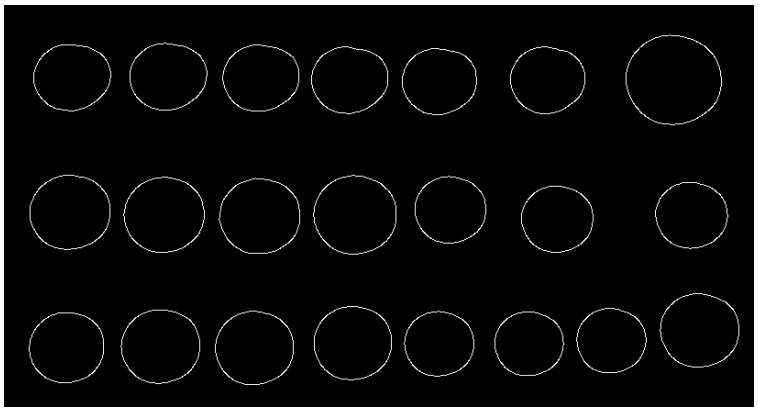

Figure 2: Edge Detection Output

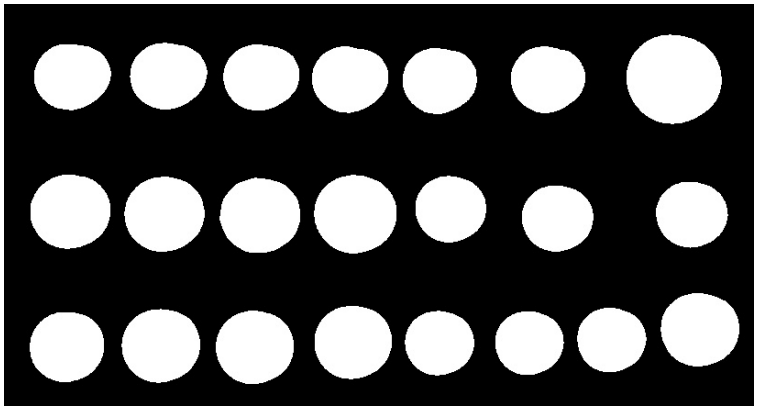

Figure 3: Segmentation Output

Figure 4: Contour Output

Figure 5: Total Number of Coins

Part 2: Panorama Creation

Input Images

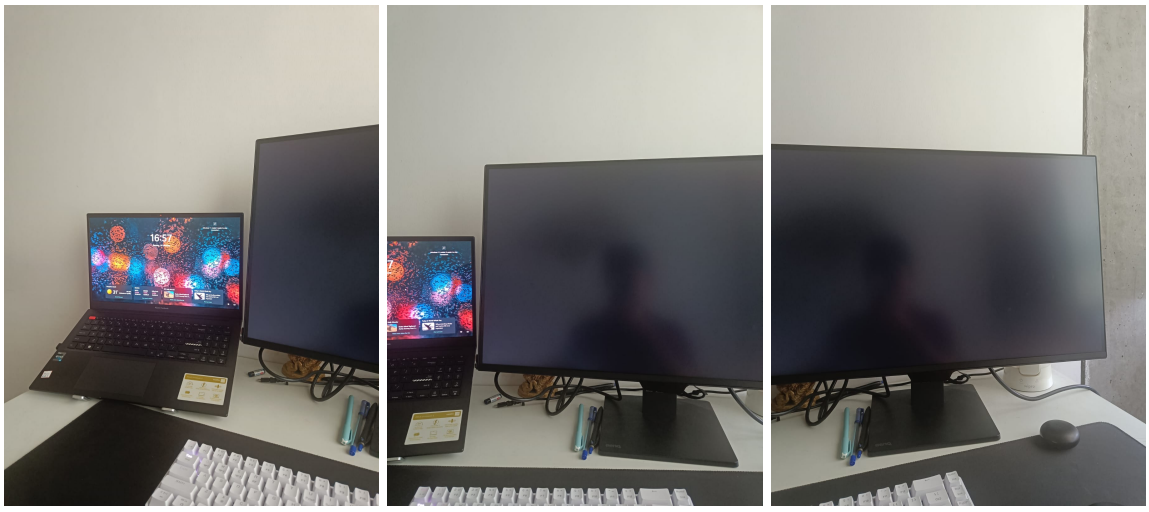

Figure 6: Input images (left, center, right) for panorama stitching

Implementation

- **Preprocessing:** Image is converted to grayscale.
- **Keypoint Detection:** Keypoints are detected using SIFT.
- **Keypoint Matching:** Keypoints are matched using FLANN-based matcher.
- **Stitching:** Stitcher function is used to create the panorama.
- **Thresholding:** Image is segmented using binary thresholding.

Output Images

Figure 7: KeyPoints matching

.

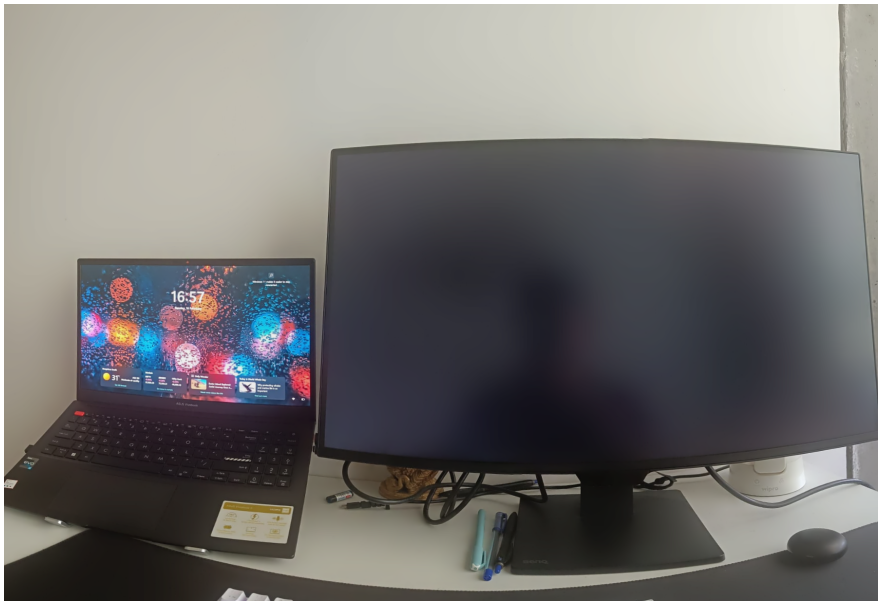

Figure 8: Final stitched panorama

Observations

Part 1

- Some internal parts were mistakenly detected as edges, so morphological closing was applied to refine the segmentation.
- Edge detection depends on the image taken and the lighting of the image as well.
- The segmentation method depends on the arrangement of coins, region-based segmentation was found to be the most suitable approach.

Part 2

- Proper image overlap improves stitching accuracy.
- Lighting and angle differences affect keypoint matching.
- Good keypoint matching ensures better alignment.
- Cropping and contour detection refine the final output.

.

GitHub Repo : https://github.com/uttam24uttam/VR_Assignment1_UttamHamsaraj_IMT2022524