

# Visual Recognition Assignment 1

Uttam Hamsaraj  
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/

<code>coins.py</code>	# The code
<code>coin_image/</code>	# The input image
<code>output_images/</code>	# Created when the script is run, contains all output images

PART-2/

<code>images/</code>	# Folder containing all input images
<code>image_stitching.py</code>	# The code
<code>output_images/</code>	# 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 smoothed using Gaussian Blur to remove noise and converted to grayscale.
- **Edge Detection:** Canny Edge Detection is applied to detect object boundaries. Morphological closing was applied to refine the segmentation.
- **Segmentation:** Region-based segmentation using contour detection extracts individual coin regions.
- **Contours:** `findContours()` is used to find contours from improved edges.
- **Counting:** Coins are counted by filtering contours based on area thresholding.

### Output Images

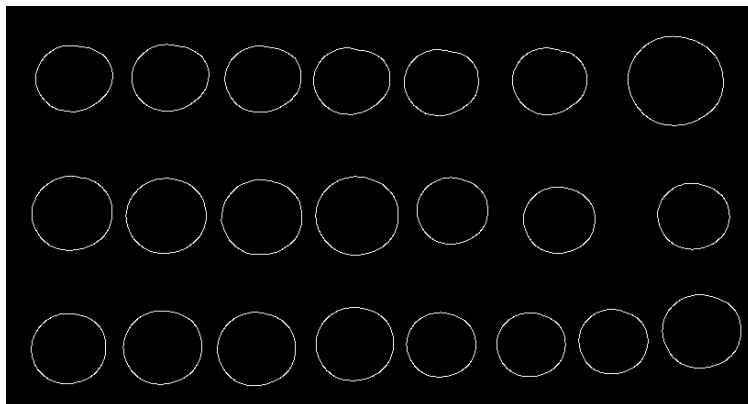


Figure 2: Edge Detection Output

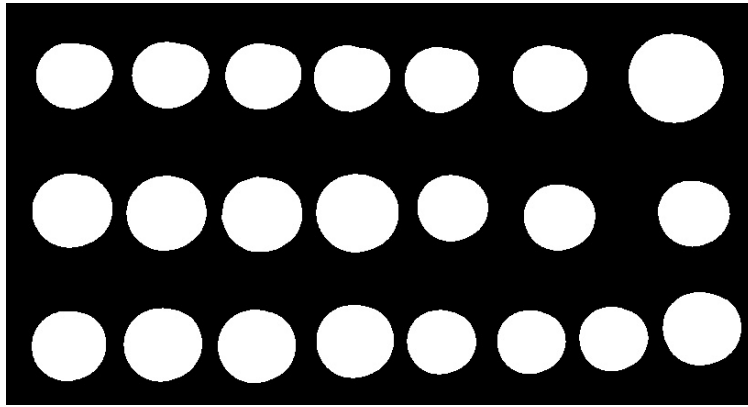


Figure 3: Segmentation Output



Figure 4: Contour Output



Figure 5: Total Number of Coins

## Part 2: Panorama Creation

### Input Images

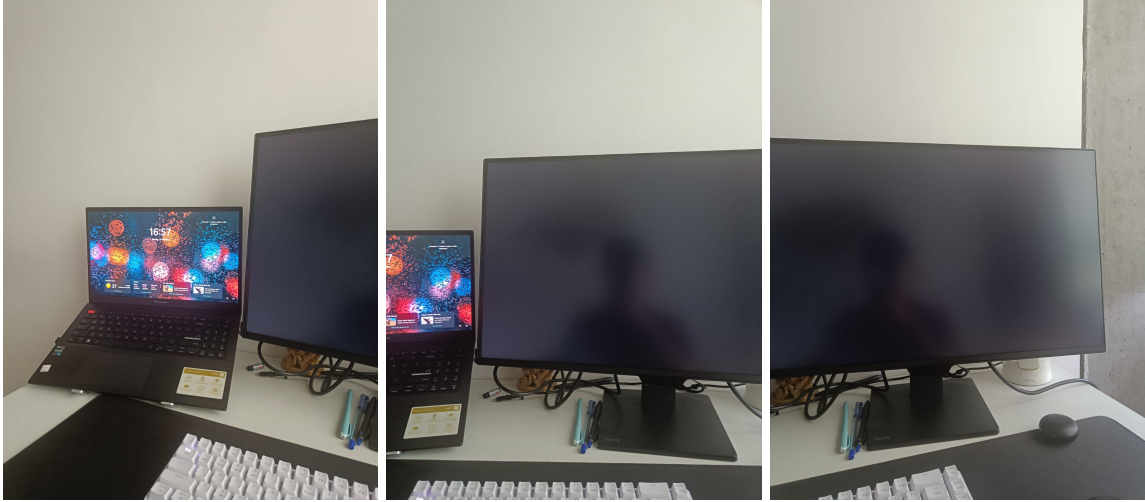


Figure 6: Input images (left, center, right) for panorama stitching

### 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.