Coin Detection and Panorama Stitching

Bijeet Basak(IMT2022510)
Github Link:
https://github.com/bijeet1221/VR_Assignment1_Bijeet_Basak_IMT2022510

1. Introduction

This assignment aims to achieve two key computer vision tasks:

> **Coin Detection and Segmentation:** Detect, segment, and count the number of coins in an image.
> **Panorama Stitching:** Stitch multiple overlapping images into a single seamless panorama.

I have used OpenCV for feature detection, image processing, and stitching, along with NumPy for numerical operations.

---

2. Coin Detection and Segmentation

Approaches Tried:

Initially, the approach involved simple **thresholding** and **contour detection**, but this led to excessive noise and false positives. To improve accuracy, I tried:

> **Gaussian Blurring** to smoothen the image and reduce noise.
> **Canny Edge Detection** to detect clear coin boundaries.
> **Contour Detection** to extract coin shapes.
> **Region-Based Segmentation** to isolate individual coins from the background.

What Worked:

> Using a **blurred grayscale image** before applying Canny edge detection improved contour detection.
> **Region-based segmentation** effectively extracted individual coins with minimal noise.
> **Bounding box extraction** helped isolate and count individual coins.

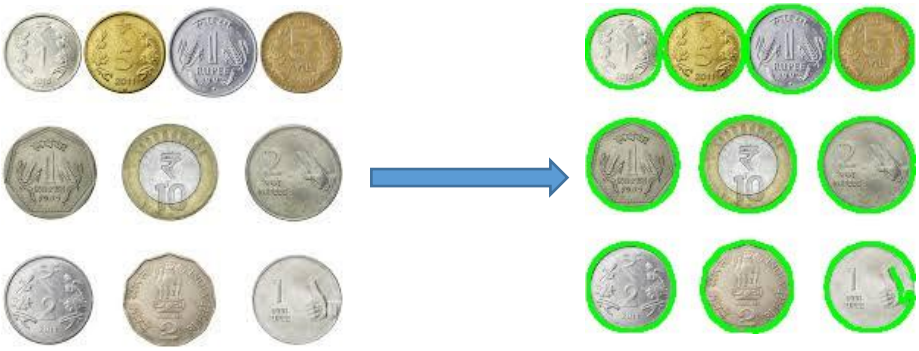
What Didn’t Work:

> Simple thresholding caused false detections when coins had reflections or overlapping regions, or images with poor quality led to detetction of engraved edges of the coin which resulted in wrong count of coins.
> Direct contour detection without blurring led to fragmented edges.
> Poor lighting conditions made it difficult to detect all coins accurately.

Final Approach:

Convert image to **grayscale**.
Apply **Gaussian blur** to smoothen the image.
Use **Canny edge detection** to identify coin edges.
Extract contours and draw bounding boxes around detected coins.
Save **Detected_coins.jpeg** in the output/ folder.

Results:



Input Image Detected coins

Total number of coins detected: 10

Observations:

The method successfully detects and counts the coins in images with good lighting.
Some false positives may still occur if the background has strong edges.
Overlapping coins can sometimes merge into a single detection.

---

3. Panorama Stitching

Approaches Tried:

Initially, I used **ORB (Oriented FAST and Rotated BRIEF)** for feature detection, but it was ineffective in detecting sufficient keypoints for matching. I then switched to:

SIFT (Scale-Invariant Feature Transform) for feature detection and keypoint extraction.
OpenCV's **Stitcher module** for aligning and blending images.
Keypoint Visualization to analyze detected features before stitching.

What Worked:

SIFT provided excellent keypoint detection, making image alignment more accurate.
Using OpenCV's **Stitcher_create()** simplified the stitching process significantly.

Storing both stitched and keypoint-annotated images helped visualize the process.

What Didn’t Work:

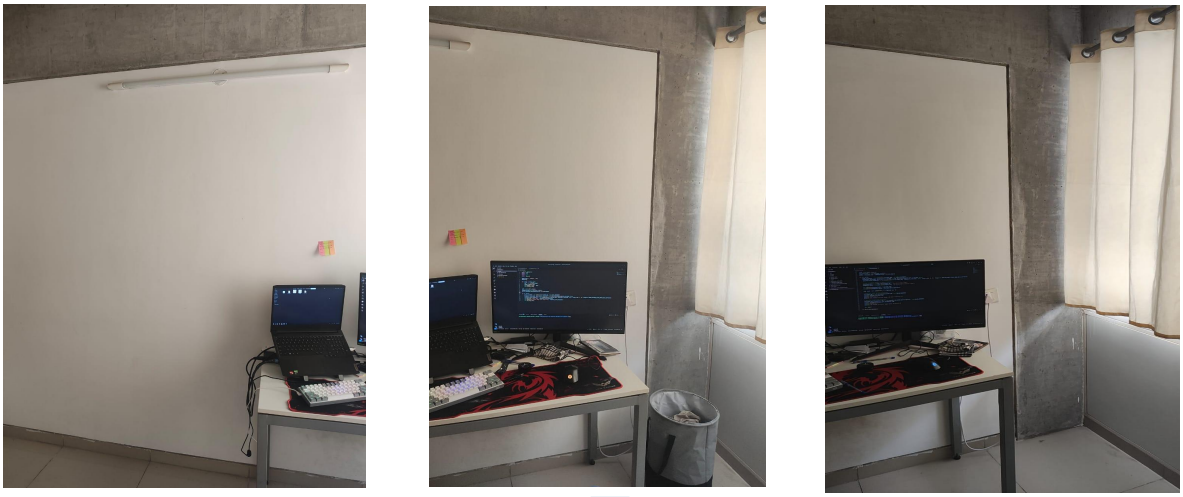
ORB-based stitching failed due to insufficient feature matches.
Poorly aligned images with insufficient overlap resulted in stitching errors.
Images with low texture struggled to generate keypoints.

Final Approach:

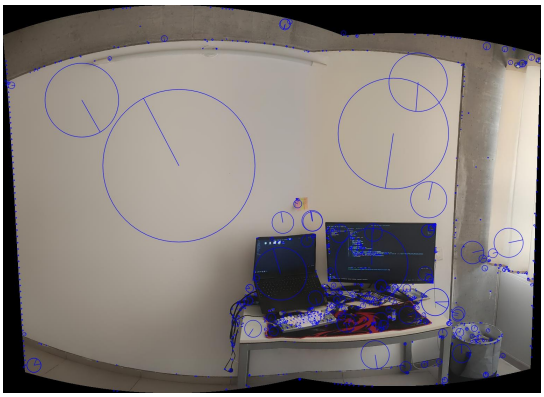
Read images from the images/ folder.
Use **SIFT** to extract keypoints and descriptors.
Apply OpenCV’s **Stitcher_create()** to blend images.
Save **stitched panorama** and **stitched image with keypoints** in output/.

Results:

Input Images



Stiched without keypoints



Stiched with keypoints

Observations:

Successful stitching requires images with sufficient overlap.
Feature-rich images with clear textures yield better stitching results.
Images with minimal overlapping features fail to stitch correctly.

---

4. Summary

Task	Initial Approach	Issues Faced	Final Approach	Outcome
Coin Detection	Thresholding, Contour Detection	False positives, noise, poor segmentation	Gaussian Blur + Canny Edge Detection + Contour Detection	Accurate coin detection & counting
Panorama Stitching	ORB for Feature Matching	Insufficient keypoints, poor alignment	SIFT for Keypoints + OpenCV Stitcher	Successful image stitching with keypoint visualization

While initial approaches faced challenges, refining methods with feature extraction and preprocessing techniques significantly improved results. The final scripts run seamlessly and produce accurate outputs for coin detection and panorama stitching.