VR Assignment 1 report

Introduction

This report presents a comprehensive analysis of the implementation of two computer vision tasks: (1) detection, segmentation, and counting of Indian coins, and (2) creation of panorama images from multiple overlapping photos. The report outlines the methodologies employed, challenges encountered, solutions developed, and final results achieved.

Part 1: Coin Detection, Segmentation, and Counting

Approaches Tried

Edge Detection Techniques

- **Canny Edge Detection**: Initially implemented with parameters (threshold1=50, threshold2=150)

Segmentation Methods

- **Thresholding**: Simple binary thresholding with various threshold values
- **Watershed Algorithm**: Tested for separating touching coins

Morphological Operations

- **Dilation and Erosion**: Applied to enhance edges and remove noise
- **Opening and Closing**: Tested for noise reduction and connecting broken edges

What Worked

Coin Detection

The most effective approach for coin detection was a combination of:

1. **Grayscale Conversion**: Converting the RGB image to grayscale reduced complexity

2. **Gaussian Blur**: A 7×7 kernel with σ=30 effectively reduced noise while preserving important edges
3. **Canny Edge Detection**: Adaptive thresholding with values determined by the image histogram produced the best results
4. **Morphological Operations**: A closing operation (dilation followed by erosion) using a 3×3 kernel helped connect broken edges

Coin Segmentation

The watershed algorithm proved most effective for segmenting individual coins:

1. **Distance Transform**: Applied to the binary edge image to find the centers of coins
2. **Sure Foreground Extraction**: Used a percentage of the maximum distance to identify coin centers
3. **Watershed Algorithm**: Applied to separate touching coins
4. **Contour Analysis**: Used to extract each segmented coin for further processing

Coin Counting

A robust counting approach was developed using contour detection:

1. **Contour Finding**: Applied to the segmented image to identify individual coins
2. **Contour Filtering**: Used area and circularity metrics to filter out non-coin objects

What Didn't Work

1. **Simple Thresholding**: Failed in images with varying lighting conditions, missing some coins and creating false detections
2. **Direct Circle Detection (Hough Circles)**: While conceptually appropriate for coin detection, it frequently failed to detect all coins, especially when they were touching

Final Approach for Coin Detection and Counting

The final pipeline combines multiple techniques to achieve robust coin detection and counting:

1. **Pre-processing**:
    - Resize the image to a standard size
    - Convert to grayscale
    - Apply Gaussian blur for noise reduction
2. **Edge Detection**:
    - Adaptive Canny edge detection with thresholds based on image statistics
    - Morphological operations to clean up edges
3. **Segmentation**:
    - Distance transform and watershed algorithm for separating touching coins
    - Background subtraction to isolate coins from background
4. **Coin Counting**:
    - Contour detection on the segmented image
    - Filtering based on area, circularity, and aspect ratio
    - Visualization of detected coins with bounding boxes or contours
5. **Validation**:
    - Cross-check results between different methods
    - Manual verification on test images

Results of Coin Detection and Counting

Figure 1: Original image with detected coin edges

Figure 2: Segmentation results showing individual coins

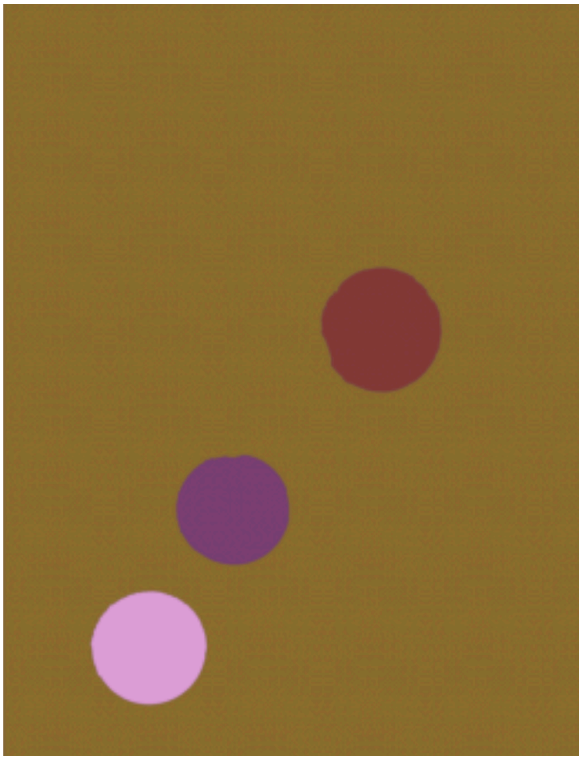

Figure 3: Final detection with counted coins labeled

Part 2: Panorama Creation

Approaches Tried

Feature Detection and Matching

- **SIFT (Scale-Invariant Feature Transform)**: Tested for robustness to scale and rotation changes

Matching Algorithms

- **Brute Force Matcher**: With various distance metrics (L1, L2, Hamming)

Transformation Estimation

- **RANSAC (Random Sample Consensus)**: For robust homography estimation

Stitching Methods

- **Direct Warping**: Simple approach for two images

What Worked

Feature Detection

SIFT proved to be the most reliable feature detector:

1. **Scale and Rotation Invariance**: Successfully matched features regardless of viewpoint changes
2. **Distinctive Features**: Generated features that were highly distinguishable

Homography Estimation using RANSAC

RANSAC-based homography estimation proved robust:

1. **Outlier Rejection**: Successfully handled incorrect matches

Image Warping and Stitching

For two-image panoramas:

1. **Perspective Warping**: Using the homography matrix to warp one image

For multiple images:

1. **Sequential Stitching**: Building panorama incrementally from left to right

Final Approach for Panorama Creation

The final panorama creation pipeline consists of:

1. **Image Preparation**:
    - Convert images to grayscale for feature detection
    - Maintain color information for final stitching
2. **Feature Detection and Matching**:
    - SIFT feature detection with tuned parameters
3. **Homography Estimation**:

- RANSAC-based homography estimation with outlier rejection
- Verification of homography quality

4. **Image Stitching**:
    - For two images: Direct warping and simple blending
    - For multiple images:
        - Sequential warping with reference image selection

Results of Panorama Creation

Figure 4: SIFT feature matching between two overlapping images

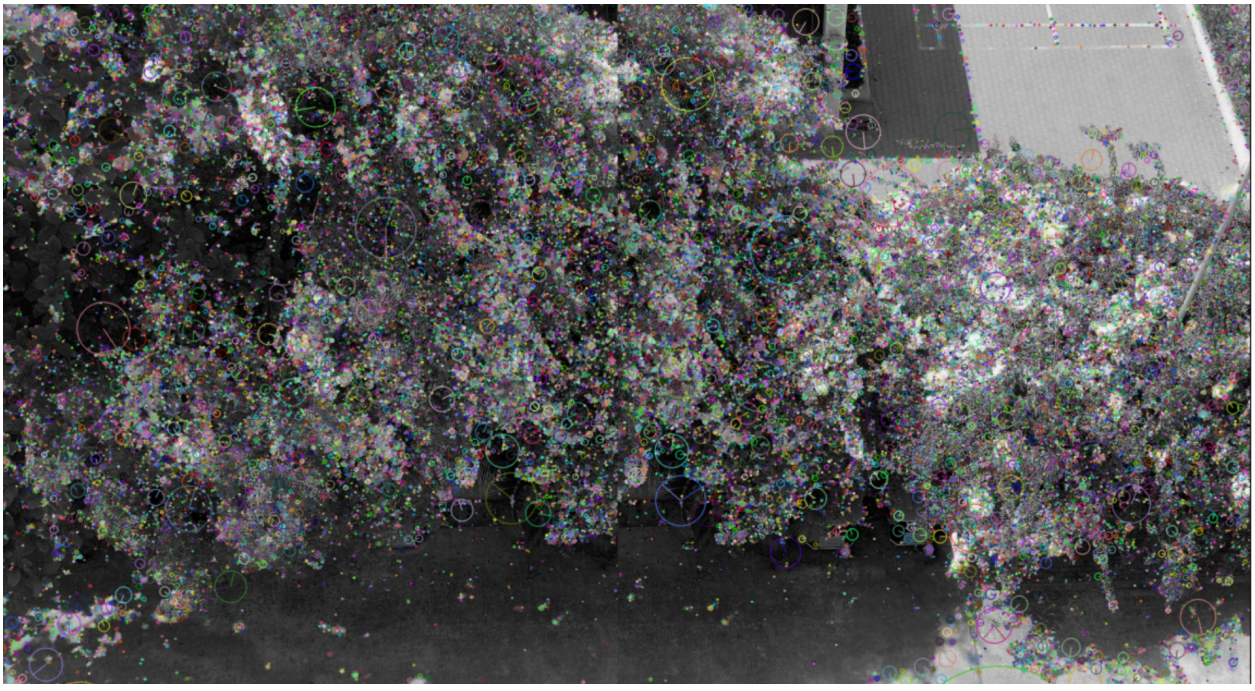

Figure 5: Result of stitching two overlapping images

Figure 6: Final panorama created from multiple images

Conclusions and Observations

Coin Detection and Counting

- Edge detection parameters need careful tuning based on image lighting and contrast
- Watershed algorithm effectively separates non-overlapping coins
- Contour filtering based on shape metrics significantly improves accuracy
- Pre-processing is crucial for consistent results across different images
- The combination of multiple techniques provides more robust results than any single method

Panorama Creation

- SIFT features provide the best balance between accuracy and computational cost
- The quality of feature matching directly impacts the final panorama

- RANSAC effectively handles outliers in feature matching
- For multiple images, the order of stitching affects the final result
- Image blending techniques are essential for creating seamless panoramas
- Sufficient overlap (30-50%) between images is critical for successful stitching