| Exercises No. 3 | | | |
|---|---|---|---|
| Topic: | Module 2.0: Feature Extraction and Object Detection | Week No. | 6-7 |
| Course Code: | CSST106 | Term: | 1st Semester |
| Course Title: | Perception and Computer Vision | Academic Year: | 2024-2025 |
| Student Name | | Section | |
| Due date | | Points | |

## Advanced Feature Extraction and Image Processing

### Exercise 1: Harris Corner Detection

**Task**: Harris Corner Detection is a classic corner detection algorithm. Use the Harris Corner Detection algorithm to detect corners in an image.

- Load an image of your choice.
- Convert it to grayscale.
- Apply the Harris Corner Detection method to detect corners.
- Visualize the corners on the image and display the result.

**Key Points**:

- Harris Corner Detection is used to find corners, which are points of interest.
- It's particularly useful for corner detection in images where object edges intersect.

### Exercise 2: HOG (Histogram of Oriented Gradients) Feature Extraction

**Task**: The HOG descriptor is widely used for object detection, especially in human detection.

- Load an image of a person or any object.
- Convert the image to grayscale.
- Apply the HOG descriptor to extract features.
- Visualize the gradient orientations on the image.

**Key Points**:

- HOG focuses on the structure of objects through gradients.
- Useful for human detection and general object recognition.

## Exercise 3: FAST (Features from Accelerated Segment Test) Keypoint Detection

**Task**: FAST is another keypoint detector known for its speed.
- Load an image.
- Convert the image to grayscale.
- Apply the FAST algorithm to detect keypoints.
- Visualize the keypoints on the image and display the result.

**Key Points**:
- FAST is designed to be computationally efficient and quick in detecting keypoints.
- It is often used in real-time applications like robotics and mobile vision.

## Exercise 4: Feature Matching using ORB and FLANN

**Task**: Use ORB descriptors to find and match features between two images using FLANN-based matching.
- Load two images of your choice.
- Extract keypoints and descriptors using ORB.
- Match features between the two images using the FLANN matcher.
- Display the matched features.

**Key Points**:
- ORB is fast and efficient, making it suitable for resource-constrained environments.
- FLANN (Fast Library for Approximate Nearest Neighbors) speeds up the matching process, making it ideal for large datasets.

## Exercise 5: Image Segmentation using Watershed Algorithm

**Task**: The Watershed algorithm segments an image into distinct regions.
- Load an image.
- Apply a threshold to convert the image to binary.
- Apply the Watershed algorithm to segment the image into regions.
- Visualize and display the segmented regions.

**Key Points**:
- Image segmentation is crucial for object detection and recognition.
- The Watershed algorithm is especially useful for separating overlapping objects.

These exercises extend the concepts covered in the document, introducing a mix of foundational and real-time applications. Let me know if you'd like more details on any specific task!

**Rubric for Exercise 3: Advanced Feature Extraction and Image Processing**

| Criteria | Excellent (90-100%) | Good (75-89%) | Satisfactory (60-74%) | Needs Improvement (0-59%) |
|---|---|---|---|---|
| **Exercise 1: Harris Corner Detection** | Harris corners detected accurately, with clear and well-labeled visualization. Code is efficient and well-commented. | Minor issues in visualization or explanation. Code works but needs optimization. | Basic corner detection but unclear visualization or minimal explanation. | Incorrect corner detection, poor visualization, or missing explanation. |
| **Exercise 2: HOG Feature Extraction** | Correct application of the HOG descriptor with clear visualization of gradient orientations. Explanation is detailed and insightful. | Minor issues in visualization or explanation of HOG features. | Basic implementation of HOG, lacks clarity in visualization or explanation. | Incorrect HOG extraction, missing or poor explanation. |
| **Exercise 3: FAST Keypoint Detection** | Accurate and efficient FAST keypoint detection, with clear visualization and a thorough explanation. | Minor issues with keypoint detection or visualization. Code is mostly correct. | Basic keypoint detection with limited explanation or unclear visualization. | Incorrect or no keypoint detection. Missing explanation or poor visualization. |
| **Exercise 4: Feature Matching using ORB and FLANN** | ORB and FLANN applied correctly with accurate feature matching between images. Visualization is clear, and explanation is well-organized. | Minor visualization or explanation issues. Feature matching is mostly correct but not fully explained. | Basic feature matching with ORB and FLANN, but lacks depth in explanation or clarity in visualization. | Incorrect or incomplete feature matching. Poor or missing explanation. |
| **Exercise 5: Image Segmentation using Watershed** | Clear and correct segmentation using the Watershed algorithm. Boundaries are well-marked, and the regions are clearly defined. The explanation is thorough. | Minor issues with segmentation or unclear boundary marking. Code works but has minor flaws. | Basic segmentation but lacks clarity in visualization or explanation. | Incorrect or no segmentation. Poor or missing explanation or visualization. |
| **Code Quality** | Code is efficient, follows best practices, and is well-structured with clear comments throughout. | Code works but could be optimized. Comments are present but minimal. | Code is functional but lacks proper structure or detailed comments. | Code is incorrect, poorly structured, or lacks comments. |
| **Visualization of Results** | Visuals are clear, well-labeled, and easy to interpret. Proper titles and axis labels are used for each output. | Minor issues with visualization, such as unclear labels or poor formatting. | Basic visualization is present but lacks clarity or proper labels. Results are somewhat difficult to interpret. | Poor or missing visualization. No labels, titles, or clear output. |
| **Explanation and Documentation** | Comprehensive explanation of methods, clear reasoning, and well-written documentation provided. | Explanation is mostly correct but lacks depth or misses some details. Documentation is adequate but could be more detailed. | Basic explanation with minimal or unclear reasoning. Documentation is incomplete. | Poor or missing explanation. Documentation is missing or incorrect. |