# Scale-Invariant Feature Transform (SIFT) & Harris Corner Detection for Feature Mapping

Team YOLO - s19 (205, 355, 408, 420)

## 1. Introduction

The Scale-Invariant Feature Transform (SIFT) is a widely used algorithm for detecting and describing local features in images. Developed by David Lowe, SIFT is robust to changes in scale, rotation, and illumination, making it a powerful tool for tasks such as object recognition, image retrieval, and robotic localization.And also, Harris corner detection algorithm is a popular method used in computer vision to detect interesting points, or "corners," in an image. This report explores the SIFT method using OpenCV, compares it with the Harris corner detector, and evaluates the impact of different parameters on feature detection and matching.

## 2. Key Definitions

Interest Points

Distinctive locations in an image, such as corners, edges, or blobs, that can be reliably detected under varying conditions.

Feature Detection

The process of identifying interest points in an image.

Feature Description

The process of representing the neighborhood around an interest point in a way that is invariant to transformations.

Feature Matching

The process of comparing descriptors from different images to find corresponding points.

SIFT

A method for detecting and describing interest points that is invariant to scale, rotation, and illumination.

Harris Corner Detector

A method for detecting corners in an image based on the local autocorrelation of intensity gradients.

## 3. Project Steps

I. Problem Identification & Project Planning
II. Data Collection
III. Algorithm Development and optimizing
IV. Testing with a couple of images
V. Report writing

## 4. Methodology

This is the process of detecting and describing interest points using SIFT and Harris Corner Detector.

**1. SIFT**

- First, extract interest points from a pair of images using OpenCV's SIFT detector. Then, test with sample images to observe performance.
- Implement a basic descriptor using pixel intensity values in a 5x5 window around each keypoint.
- Generate scale-invariant descriptors for each keypoint using OpenCV's SIFT descriptor.
- Next, calculate the Sum of Squared Differences (SSD) to find the distance between feature vectors and match keypoints based on the smallest distance.
- Then, use SIFT matching for more accurate results.
- Test the SIFT matching ratio to control the number of matches.
- Finally, observe how changes in parameters affect matching performance.

### 2. Harris Corner Detector

This is the method we have followed.

- First find the 2 images and load the images using OpenCV.
- Convert the images to grayscale to simplify processing.
- Apply the Harris Corner Detector to detect keypoints in both images.
- Extract key points by filtering strong corner responses based on a threshold.
- Convert detected keypoints into OpenCV KeyPoint objects for further processing.
- Use the ORB (Oriented FAST and Rotated BRIEF) descriptor to compute feature descriptors for the detected Harris keypoints.
- Perform feature matching using OpenCV's BFMatcher with the Hamming distance metric.
- Sort the matches based on distance to retain the best ones.
- Draw the matched keypoints between the two images using OpenCV's drawMatches().
- Display the final matched image using Matplotlib.

## 5. Results and Discussion

### 1. SIFT

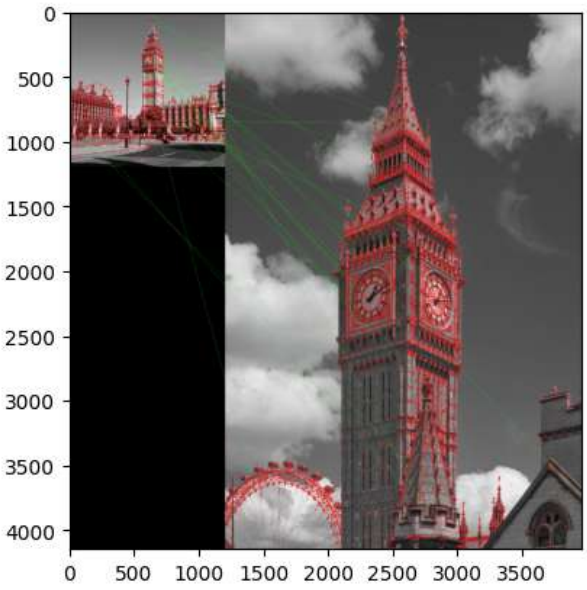


Figure 1 : SIFT matching of different scale image

SIFT has detected a large number of keypoints across the image, including edges, corners. According to the above result,The SIFT descriptor will perform well under scale changes. Reason for that is its gradient-based orientation assignment and normalization. SIFT matching will produce more accurate and reliable matches, especially when using a distance ratio threshold to filter out poor matches.

**More results on SIFT**

1. Car Image with its Cropped & Rotated Image

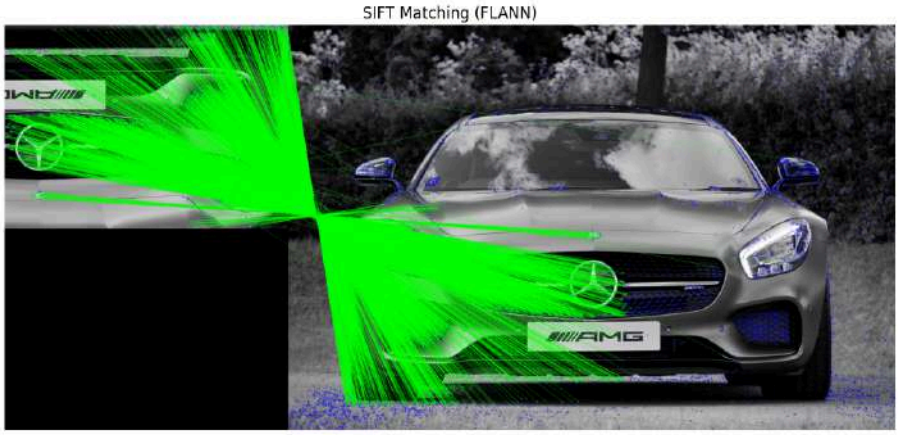


Figure 2 : SIFT matching of rotated image

2. Mouse Image with its Cropped, Rotated & Ear Cropped Images

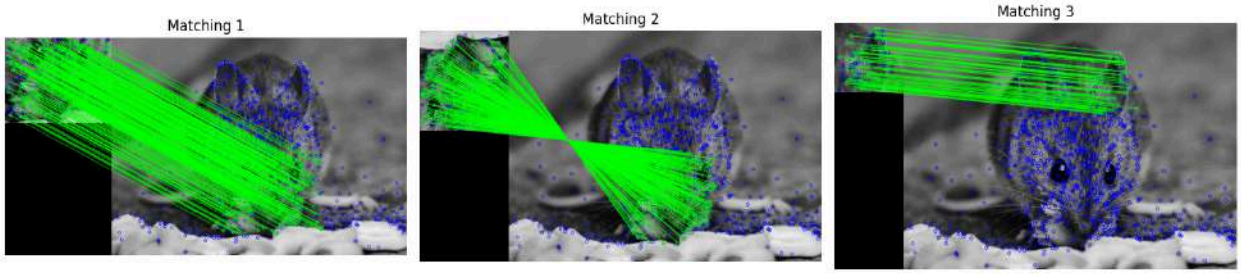


Figure 3 : SIFT matching on different image transformations

**Limitations and Possible Improvements**

SIFT can detect spurious keypoints in noisy images, leading to incorrect matches. And also SIFT may struggle to detect keypoints in regions with little texture or repetitive patterns. SIFT

descriptors are high-dimensional (128 elements per keypoint), leading to high memory usage for large datasets.

As Improvements we can apply preprocessing techniques like Gaussian smoothing or median filtering to reduce noise before keypoint detection. And we can combine SIFT with other detectors (e.g., Harris) to improve keypoint detection in low-texture regions.and also, we can add dimensionality reduction techniques to compress SIFT descriptors.

**2. Harris Corner Detector**


Figure 4 : Harris matching of different scale image

Harris will detect corners efficiently but may miss key points in regions with low texture or repetitive patterns.It will not be invariant to scale or rotation. According to the above result, Intensity-based descriptors will work well for simple translations but fail under rotation, scale, or illumination changes.

The number of correct matches will be lower compared to SIFT, especially for images with significant scale or rotation differences. We can say Increasing the window size for the Harris detector will detect larger corners but may miss smaller ones.

**More results on Harris**

1. Mouse Image with its Cropped Image

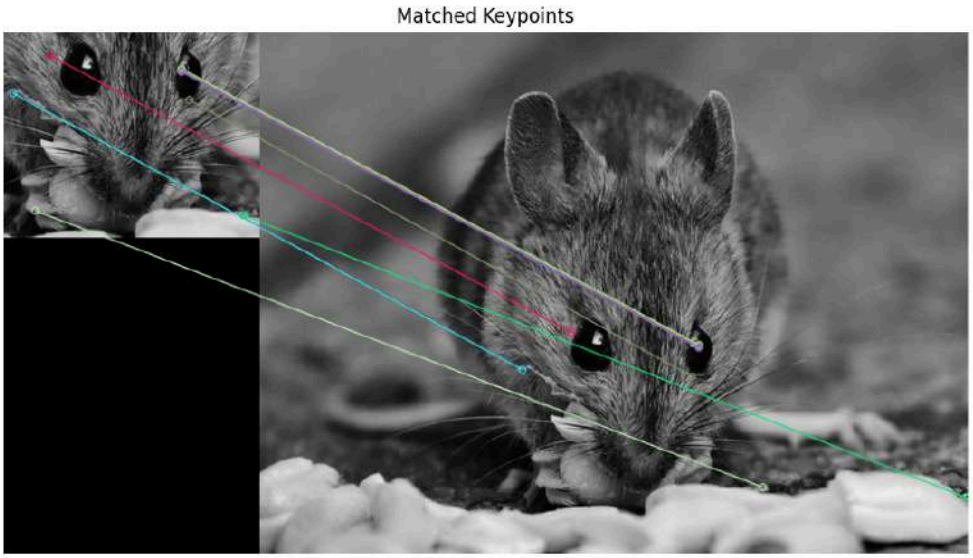


Figure 5 : Harris matching of cropped image

2. Mouse Image with its Cropped & Rotated Image

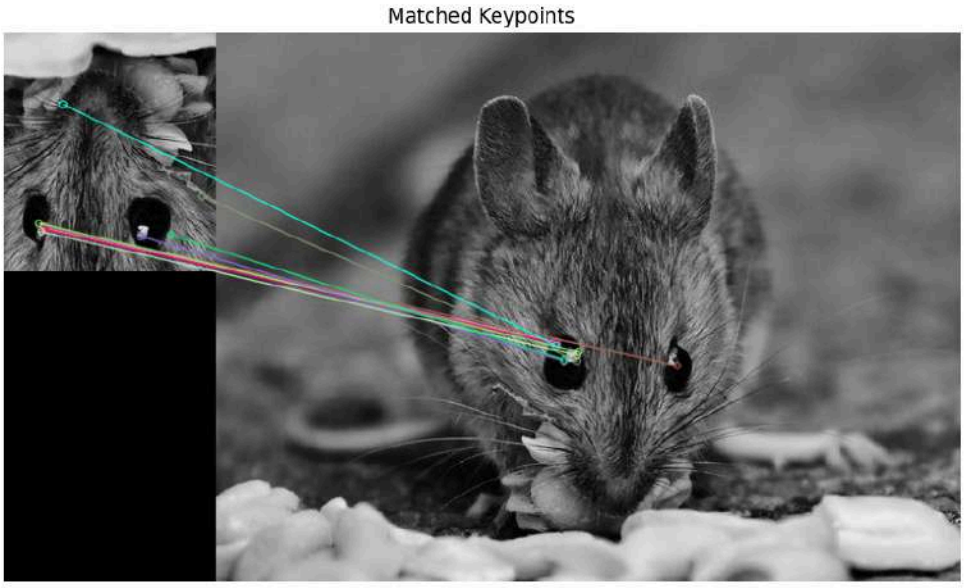


Figure 6 : Harris matching on different image transformations

**Limitations and Possible Improvements**

Harris is not scale-invariant and not rotation-invariant. not scale-invariant meaning is it may miss corners at different scales or detect false corners in scaled images. Harris struggles to detect corners in regions with little texture.

As Improvements, we can combine Harris with a rotation-invariant descriptor to improve resilience under rotation.and we can merge Harris with other detectors (e.g., FAST or SIFT) to improve performance in low-texture regions.

**Comparison of SIFT and Harris**

| Feature | SIFT | Harris Corner |
|---|---|---|
| Scale Invariance | Yes | No |
| Rotation Invariance | Yes | Yes |
| Affine Transformation Invariance | Partial | No |
| Illumination Invariance | Yes | Limited |
| Descriptor Available | Yes | No |
| Speed | Slow | Fast |
| Robustness | High | Low |
| Best for | Object recognition, matching | Simple corner detection |

Table 1 : Comparison of SIFT and Harris

## 6. Task Breakdown

The project was divided into two main tasks,

| Task | Team Members |
|---|---|
| Image Processing, Code Implementation, Experimentation & Results Analysis | S19205, S19355, S19420 |
| Project Reporting & Documentation | S19408 |
| Presentation Creation & Presenting | S19205, S19355 |

## 7. Conclusion

In this study, we explored two fundamental methods for interest point detection and description: SIFT and Harris Corner Detector. SIFT demonstrated exceptional resilience to scale, rotation, and illumination changes. Its gradient-based descriptors provided highly distinctive features, enabling accurate and reliable matching even under significant transformations. However, its high computational cost and sensitivity to noise pose challenges for real-time applications.

On the other hand, Harris was efficient for corner detection and performed well in tasks like motion tracking. Matching features with Harris often struggled under transformations. Both methods highlighted the trade-offs.

From this analysis, we learned that matching accuracy heavily depends on the quality of descriptors and the method's invariance properties. SIFT's descriptors consistently outperformed Harris in matching tasks, especially under scale, rotation, and illumination changes.

However, Harris's simplicity and speed make it suitable for applications where computational efficiency is critical. By addressing these challenges, we can enhance the performance of these methods and expand their applicability to more demanding real-world tasks in computer vision.

## 8. References


1. OpenCV: Introduction to SIFT (Scale-Invariant Feature Transform)
2. OpenCV: Harris corner detector
3. https://docs.opencv.org/master/dc/dc3/tutorial_py_matcher.html