|  |
| --- |
| Team 18 |
| Feature Detection and Matching |
| |  |  | | --- | --- | | Name | BN | | Talal Mahmoud Emara | 34 | | Maya Mohamed | 52 | | Meram Mahmoud | 68 | | Nouran Hani | 75 | |

|  |
| --- |
|  |

# Corner Detection

## Harris Detection

### Gradient Computation

The Harris corner detector begins by computing the image gradients using a custom Sobel operator to produce two gradient images, Iₓ (horizontal) and Iᵧ (vertical). These gradient maps capture edge information.

### Second-Moment Matrix Assembly

At each pixel, three structure components are formed: Iₓₓ = Iₓ², Iᵧᵧ = Iᵧ², and Iₓᵧ = Iₓ · Iᵧ. These components are summed over a cantered sliding window of size window size by window size to create the second-moment matrix H:

Where Sxx, Syy, Sxy represents the summation of the pixel intensities within a window of size cantered around the pixel of interest. This summation is applied to each of the components of the second moment matrix (Iₓₓ, Iᵧᵧ, Iₓᵧ).

Corner Response Calculation

For each pixel, the corner response R (I, j) is defined as:

R = det(H) - k · (trace(H)) ²

= (Sₓₓ·Sᵧᵧ - Sₓᵧ²) - k· (Sₓₓ + Sᵧᵧ) ².

The sensitivity factor k (typically between 0.02 and 0.08) controls how sharply the detector distinguishes corners from edges: lower values produce more candidates (including some edges), while higher values yield fewer, more reliable corner points.

### Normalization and Thresholding

The response map R is linearly normalized to the range [0,255] for visualization, then thresholder—commonly at the 95th percentile of positive responses—to extract strong corner candidates. Adjusting this threshold affects detection density: lowering it retains more (weaker) corners, while raising it focuses on the strongest corners.

### 5. Non-Maxima Suppression

Two non-maxima suppression steps are applied to refine the detected corners:

* **Local NMS**: With a window size parameter, local non-maxima suppression ensures that only the highest response is retained in each small neighbourhood. This step helps eliminate weaker corners but may still leave some redundant corner points in proximity.
* **Distance-based NMS**: To further refine the corner set, distance-based non-maxima suppression is applied. The parameter **distance** specifies the minimum allowed distance between two corners. Corners within this distance are considered redundant, and only the stronger corner is kept. This helps ensure that the final set of detected corners is not cluttered with nearby points, improving the accuracy of corner localization.

### Results

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

Harris corner detection offers fast performance, with the first image (358 × 496) processed in **1687.5 ms** and the second image (225 × 225) in **472.2 ms**. The overall average time per pixel is **0.0111 ms/pixel**.

## Lambda (λ) Corner Detector

### Introduction

Corner detection is a crucial step in many computer vision tasks such as object recognition, tracking, and image matching. It helps identify key points in an image where the intensity changes significantly in multiple directions—these points are often rich in visual information. This report explains a custom implementation of the **Minimum Eigenvalue corner detection method**, commonly known as the **Shi-Tomasi** or **λ (lambda) detector**, and compares it with OpenCV’s built-in version.

### How the λ Detector Works

#### Image Preprocessing with Gaussian Smoothing

To reduce noise before analyzing the image, Gaussian smoothing is applied. This step ensures that small variations in pixel intensity due to noise don't falsely appear as corners. The amount of smoothing is controlled by two parameters: the kernel size and the standard deviation (sigma) of the Gaussian function.

#### Gradient Calculation

After smoothing, the image gradients in the horizontal and vertical directions are calculated. These gradients represent how quickly pixel values change across the image and are essential for identifying areas with strong edge or corner features.

#### Building the Structure Tensor

Each pixel is analyzed using a small 2×2 matrix called the **structure tensor** or **second moment matrix**. This matrix captures the intensity variation in the neighborhood of the pixel based on the previously computed gradients. Specifically, it measures how "corner-like" a region is.

#### Computing the Minimum Eigenvalue

Instead of computing both eigenvalues of the structure tensor (which is more computationally intensive), the implementation calculates only the smaller of the two. This is sufficient to assess the presence of a corner. A higher minimum eigenvalue suggests a stronger corner at that pixel.

#### Thresholding and Non-Maximum Suppression

To select significant corners, a threshold is applied to the minimum eigenvalue map. Only values above a certain percentage of the maximum value are kept. Then, a technique called **non-maximum suppression** is applied: in each small region, only the strongest corner is retained. This prevents clusters of nearby detections and ensures well-distributed, high-quality keypoints.

#### Comparison with OpenCV

OpenCV has a built-in function that performs the same λ corner detection method. The custom implementation is compared to OpenCV’s output using:

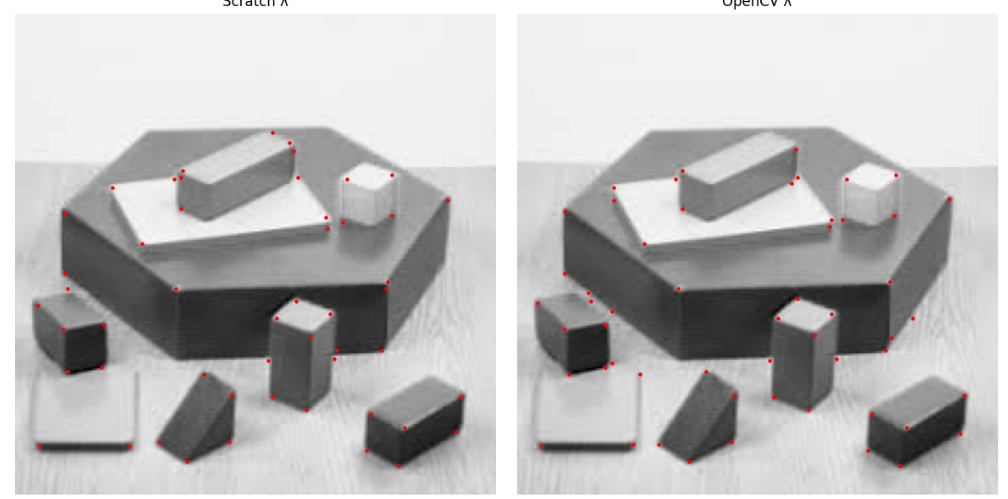
* **Number of detected corners**
* **Accuracy of corner positions**
* **Processing time**

In most cases, the custom implementation produces results very close to OpenCV’s but takes more time due to the manual computation of gradients, smoothing, and eigenvalues.

#### Results and Comparison

**Keypoint Detection Comparison**

* **Custom Implementation**:
  + Number of Keypoints: \_\_51\_\_\_
  + Time consumed: \_\_\_ 0.1495s\_\_
* **OpenCV Implementation**:
  + Number of Keypoints: \_\_\_54\_\_



**Conclusion**

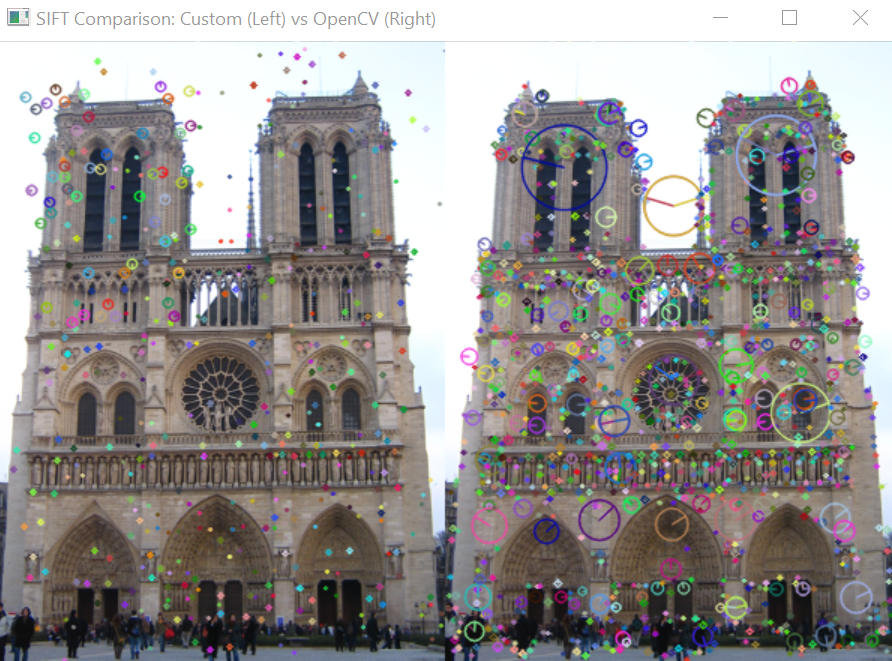
The custom λ detector successfully replicates the functionality of OpenCV’s corner detection algorithm, with similar results in keypoint localization. Although it is slower due to lack of optimization, it serves as a valuable tool for learning and debugging.

# Feature Matching

## SIFT

Overview of Pipeline

The main function, sift, contains the SIFT pipeline, performing the following major steps:

1. Scale-Space Construction

* Builds a multi-scale representation of the image using a Gaussian pyramid and computes a Difference of Gaussians (DoG) pyramid:
* The input image is first converted to grayscale and upsampled.
* A set of progressively blurred images is created per octave using Gaussian filters.
* DoG images are created by subtracting consecutive Gaussian-blurred images.

1. Keypoint Detection  
   Identifies local extrema in the DoG pyramid:

* It searches for pixels that are maxima or minima in a 3×3×3 neighborhood across scales.
* Low contrast and edge responses are suppressed using a Hessian-based filter.
* *Keypoint Refinement*  
  Refines detected extrema positions using subpixel interpolation via a Taylor series expansion:
* Unstable or poorly localized keypoints are discarded.
* Offset corrections are iteratively applied to converge to the keypoint location.

1. Orientation Assignment  
   Computes the dominant gradient orientation around each keypoint:

* A histogram of orientations is built using a Gaussian-weighted window.
* Keypoints may be duplicated with different orientations if multiple strong peaks exist.

1. Descriptor Computation  
   Generates a 128-dimensional descriptor per keypoint:

* The image patch around the keypoint is divided into 4×4 subregions.
* An 8-bin orientation histogram is computed for each subregion, considering the keypoint's dominant orientation.
* The final descriptor is normalized and thresholded to enhance robustness.

## Feature Matching

After **extracting feature descriptors from both images using SIFT**, we match these descriptors with one of two approaches: Sum of Squared Differences (SSD) or Normalized Cross-Correlation (NCC). The goal is to establish correspondences **between key points extracted from the two images** and to compare the efficiency and accuracy of the two matching techniques.

#### *Sum of Squared Differences (SSD)*

SSD is a simple and efficient method for measuring the distance between two descriptor vectors. The process is as follows:

Compute the **squared Euclidean distance** between each descriptor in the first image and every descriptor in the second image.

A black text on a white background

AI-generated content may be incorrect.

For each descriptor in the first image, find the descriptor in the second image with **the minimum distance**.

**Apply Lowe’s Ratio Test** to ensure the match is reliable by comparing the distance of the best match to the second-best match. If the ratio is below a specified threshold, the match is accepted.

**Advantages**:

**Fast computation** due to direct vectorized distance calculations.

Well-suited for real-time applications when speed is critical.

**Disadvantages**:

Less robust to noise and illumination changes compared to NCC.

#### Normalized Cross-Correlation (NCC)

NCC measures the similarity between two normalized descriptors by computing the dot product of their normalized forms:

Normalize each descriptor by dividing it by its norm.

A diagram of mathematical equations

AI-generated content may be incorrect.

Compute the dot product or correlation between each pair of descriptors.

Select the descriptor with the highest correlation value as the match.

**Apply Lowe’s Ratio Test** to maintain match reliability by ensuring the best match is significantly better than the second-best match, based on their correlation scores. If the ratio passes the threshold, the match is accepted.

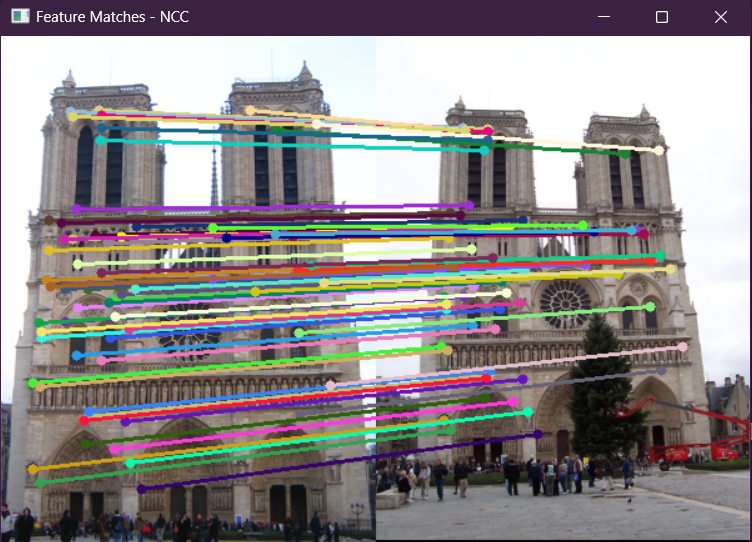
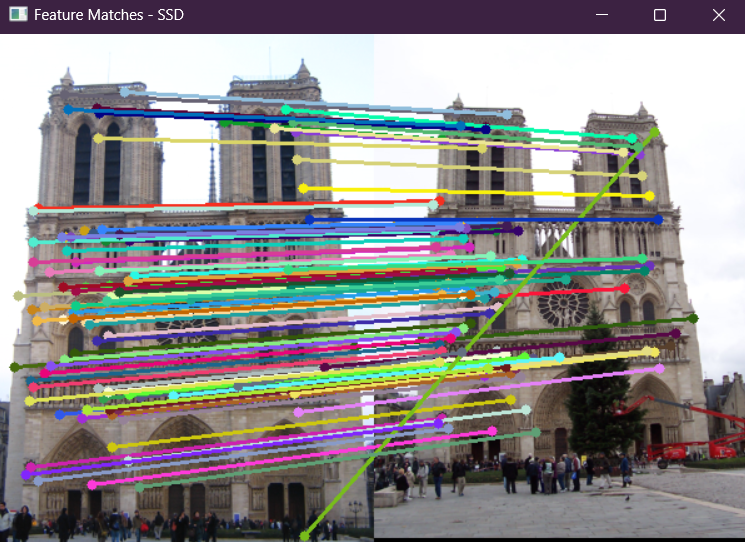
**Advantages**:

More accurate and reliable in cases with illumination variations.

Better handles differences in contrast and brightness.

**Disadvantages**:

**Slower computation** due to normalization and exhaustive pairwise comparisons.

 As shown in Figure 1 and Figure 2, **SSD produces some matching errors** due to sensitivity to noise and illumination changes, while **NCC delivers more stable and accurate matches** under varying conditions.

***Figure 1 - SSD Figure 2 - NCC***

## Lowe’s Ratio Test

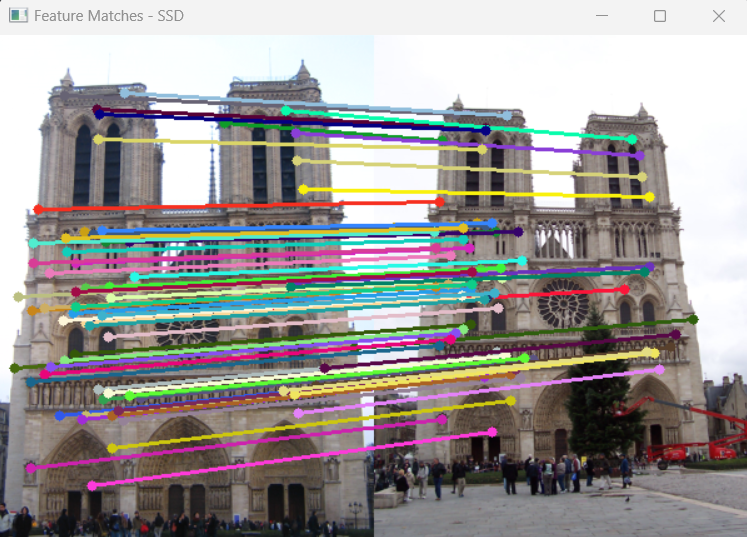
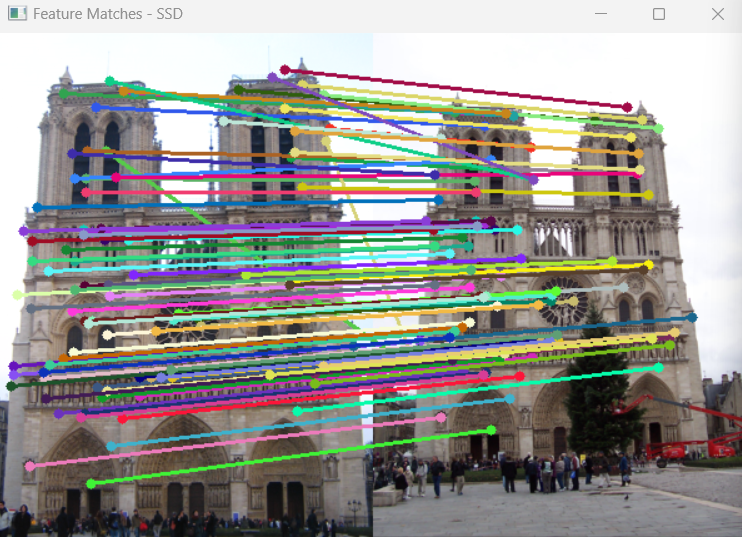
In feature matching, it’s common for a descriptor from one image to have multiple similar matches in another image. Many of these potential matches could be unreliable, ambiguous, or false positives. To improve match reliability and reduce false positives, **Lowe’s Ratio Test is applied**.

The test ensures that a match is distinctive and unambiguous by checking how much better the best match is compared to the second-best. If both are similarly good or similarly bad, the match is likely unreliable and should be discarded.

For each descriptor in Image A:

Find its two nearest neighbors in Image B based on the chosen distance metric (SSD or NCC).

Compute the ratio: Ratio = best match / second best match

If this ratio passes a predefined threshold, the match is accepted. Otherwise, it is discarded.

***Figure 3 - SSD ratio=0.8 Figure 4 – SSD – ratio =0.4***

**Figure 3 and Figure 4: Feature matching results using SSD with different ratio thresholds. Figure 3 shows matches with a ratio of 0.8, resulting in more matches but higher false positives, while Figure 4 uses a stricter ratio of 0.4, reducing false matches and improving reliability.**

## Feature matching Parameters

Several parameters control the behavior and flexibility of the matching process:

**method**: Selects the a method ("ssd" or "ncc").

**top\_n**: Determines how many of the best matches to show based on their similarity scores. If the number of valid matches exceeds this number, the top N matches are selected according to their matching score (lowest for SSD, highest for NCC).

**ratio\_threshold**: A critical parameter for **Lowe’s Ratio Test**. It controls the strictness of accepting a match:

For SSD, the ratio between the best and second-best distance must be less than this value.

For NCC, the best score must be above this threshold and sufficiently better than the second-best score

These parameters allow for flexible tuning of the matching process to balance between precision and performance as shown in figures above.