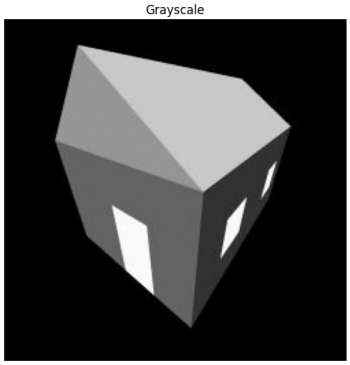
Computer vision

Lab 3

Task 1: Harris Corner Detector

Implement `harris_corners` to detect corners.

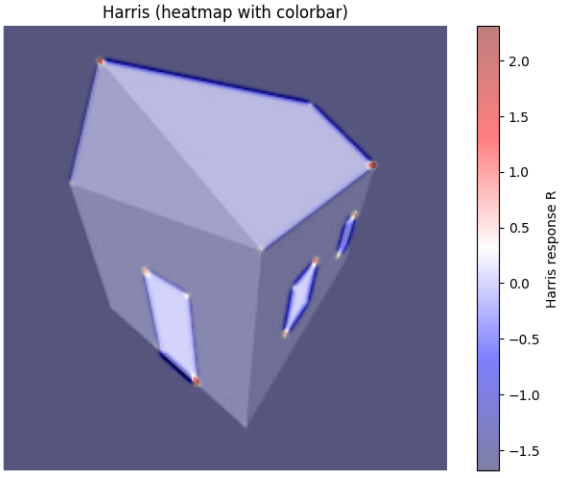


The goal is to identify interest points in an image, that is, pixels whose neighborhood shows strong variation in two different directions. Such points typically occur at junctions or corners, in contrast to edges, where the variation is strong in only one direction, or flat regions, where there is little to no variation at all.

The method works by first computing image gradients in the horizontal and vertical directions, which capture how intensity changes across the image. Within a small window, these gradient values (I) are aggregated into a structure tensor, the second-moment matrix. From this matrix, a corner response score is computed using the Harris formulas:

$$M = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

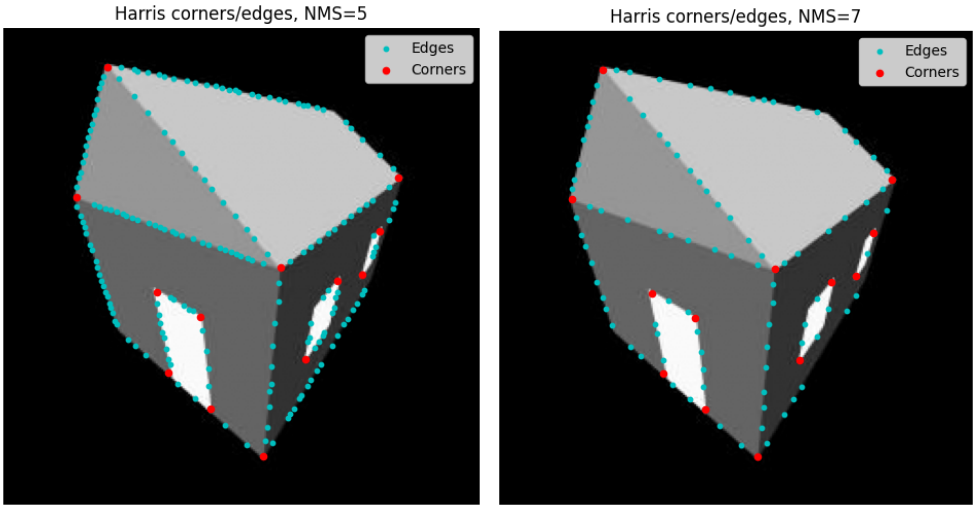
$$R = \det(M) - k \cdot (\mathrm{tr}(M))^2.$$



When the Harris response R is large and positive, the intensity changes strongly in both directions, and the location is a corner. When R is negative with large magnitude, the local structure varies mainly in one direction (the trace of the matrix is large but the determinant is small), meaning the location lies along an edge. When both the determinant and the trace are small, the region is flat and contains no meaningful feature. On the heatmap of raw R (using a diverging colormap), positive values (for example, red) indicate corners, negative values (for example, blue) indicate edges, and values close to zero appear neutral (white or light color).

In this function (Harris from scratch), several additional steps were included to improve the quality of the Harris detector results:

- Gaussian blur smooths and stabilizes responses.
- Dilate + thresholds (NMS) isolates strong corners/edges.
- Centroids clean up clusters into single points.



As we see above, the **Non-Maximum Suppression (NMS)** parameter (values 5 and 7 in this example) works as a thresholding step that keeps only the strongest local responses while suppressing weaker neighboring points. With a larger MNS the plot shows less noise for edges points. However, not all corner points (red markers) are detected, even when the parameters are adjusted - this happens because some corners have weaker responses or are blurred by smoothing, causing them to fall below the detection threshold.

Rotate image 45° and reapply detector. Analyze rotation invariance.

The Harris corner detector is in theory rotation-invariant. When we rotate an image, the gradient directions rotate too, but the values that matter for the corner response - the eigenvalues of the local structure tensor - don't change. Rotating a symmetric matrix by a rotation transform doesn't change its eigenvalues, only its eigenvectors (directions):

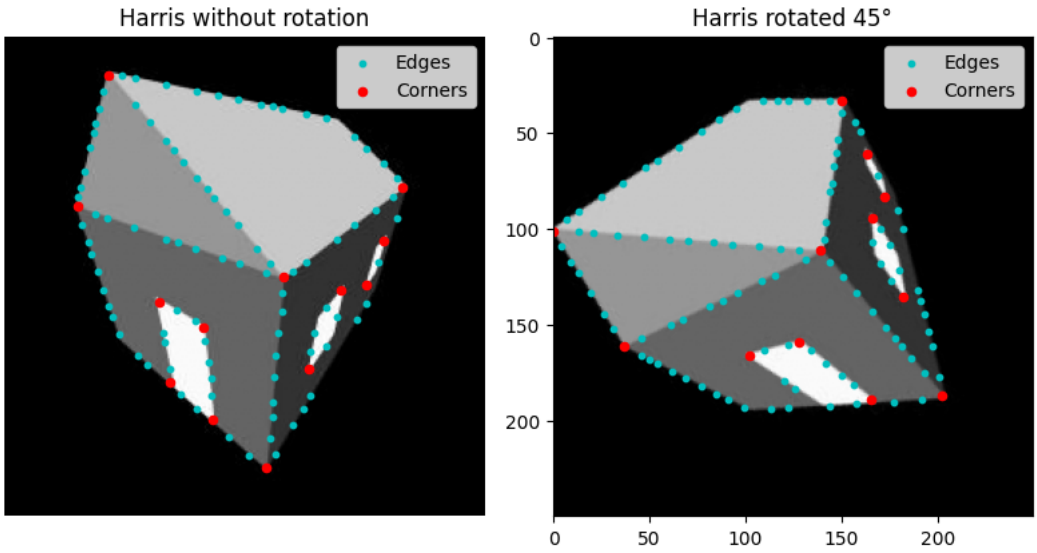
$$R = \det(M) - k \cdot (\mathrm{tr}(M))^2.$$

$\det(M) = \lambda_1 \lambda_2$
$\mathrm{trace}(M) = \lambda_1 + \lambda_2$
λ_1 and λ_2 are the eigenvalues of M

$$R = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

$$\lambda_1' = \lambda_1, \quad \lambda_2' = \lambda_2$$

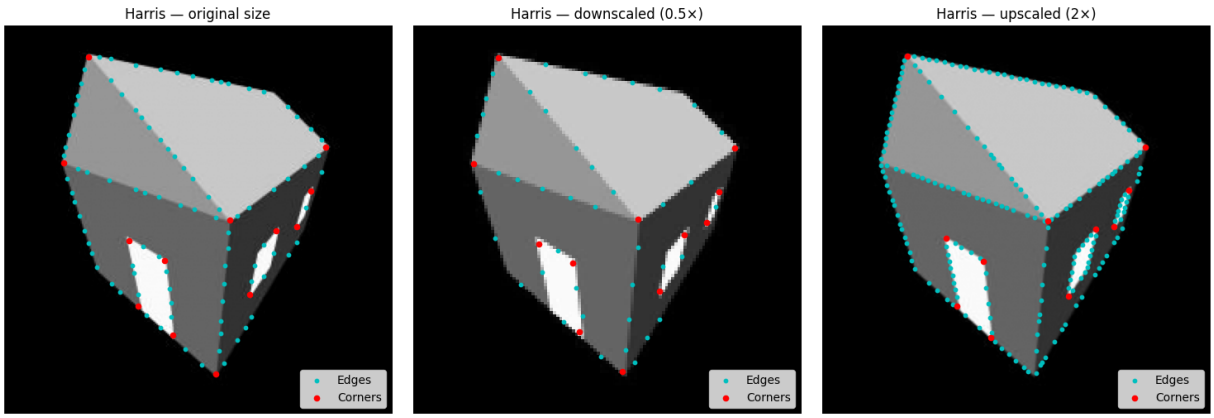
Because the eigenvalues stay the same, the Harris response R does not change after rotation.



This was confirmed when we rotated the image by 45°. The same main features, such as roof tips, window corners and line junctions, were detected both before and after rotation.

Scale image and observe corner detection behavior.

Scaling the image gave different results. The Harris detector is not scale-invariant, since the analysis window it uses is fixed in pixel size.

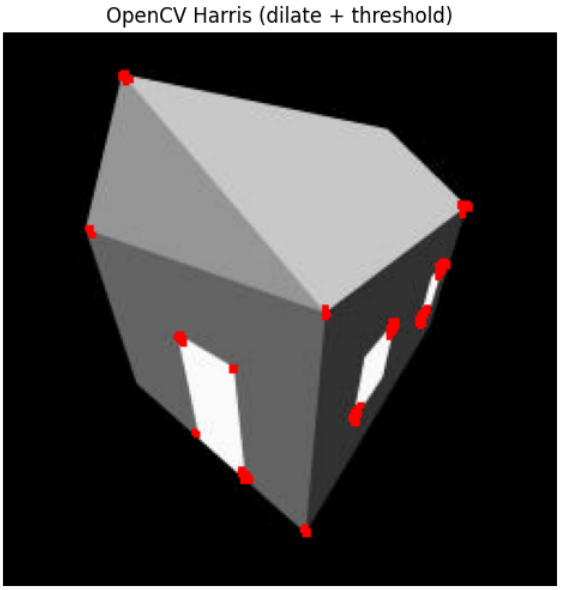


We expected that scaling up (2×) should give more visible points to detect, not fewer corners. But we see more edges and fewer corners after upscaling, which has to do with how the Harris detector interacts with image interpolation and the σ parameter. Rescaling doesn't invent new sharp features, it smooths between existing pixels:

- Intensity transitions (gradients) become smoother,
- Sharp corners get blurred out (R values weaker),
- But edges (which are continuous) stay visible (more smooth pixels along their direction).

In the downscaled image, the corners are not detected exactly at the door edges or window corners. This happens because downscaling reduces image resolution — many small details and sharp intensity changes are averaged together.

Compare with OpenCV's built-in Harris detector.



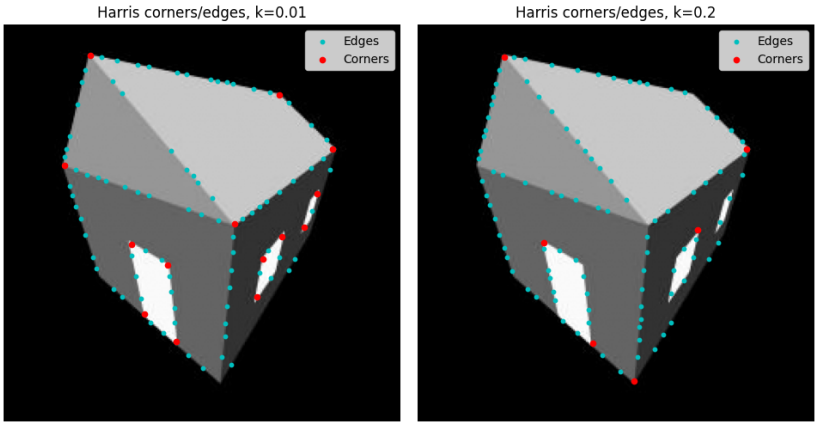
In the built-in OpenCV Harris detector, we used only the corner detection part without any edge detection to keep the function simple. However, we can see that it produces the same corner points as our function implemented from scratch. This shows that the gradient distribution (or structure tensor) is calculated in the same way in both our implementation and OpenCV's built-in function.

Experiment with parameter tuning.

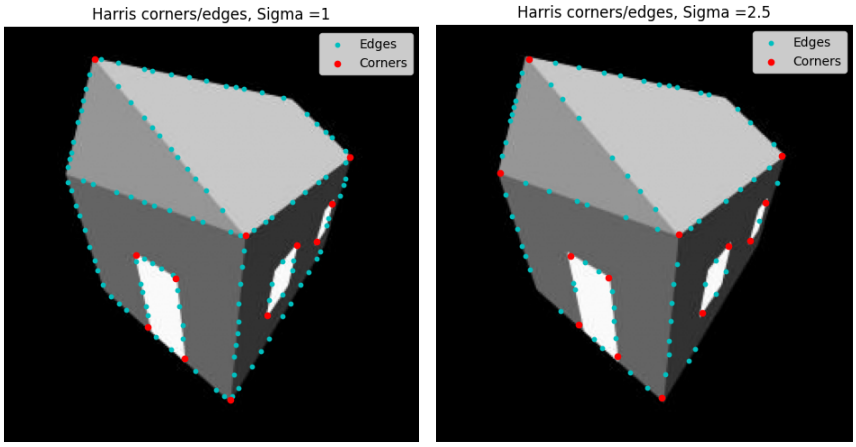
Adjusting Harris parameters changes sensitivity and stability. A lower k detects more corners, a higher k keeps only strong ones. Increasing σ smooths noise but reduces detections, and a higher R threshold filters out weak corners for more reliable results.

Non-max suppression window changing we did already in a previous task. Now we will fine-tune other parameters:

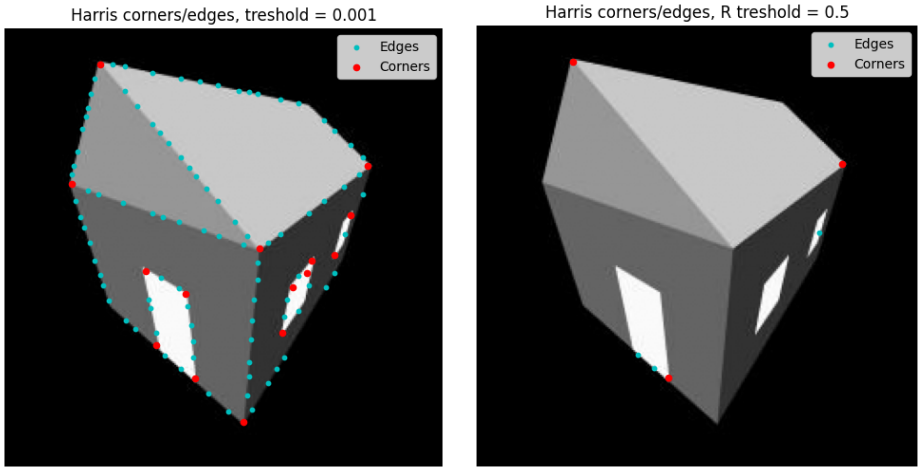
k = Sensitivity to edges vs corners: Higher → stricter, fewer corners [0.01, 0.2]



sigma = Gaussian blur strength: Higher → smoother, more stable corners [1, 2.5]



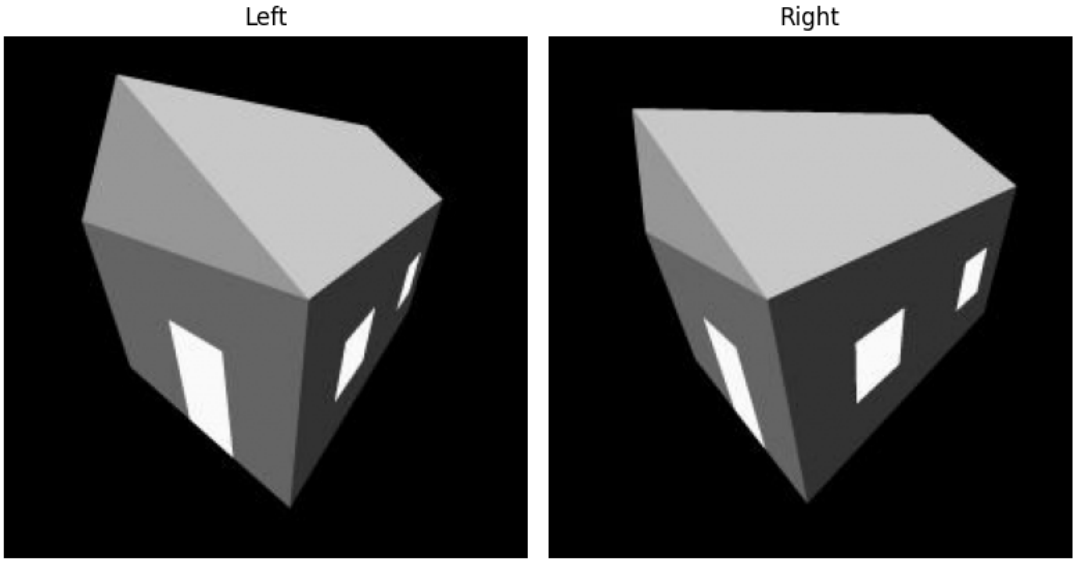
Threshold for R (both edges, corners): Higher → fewer points [0.001, 0.5]

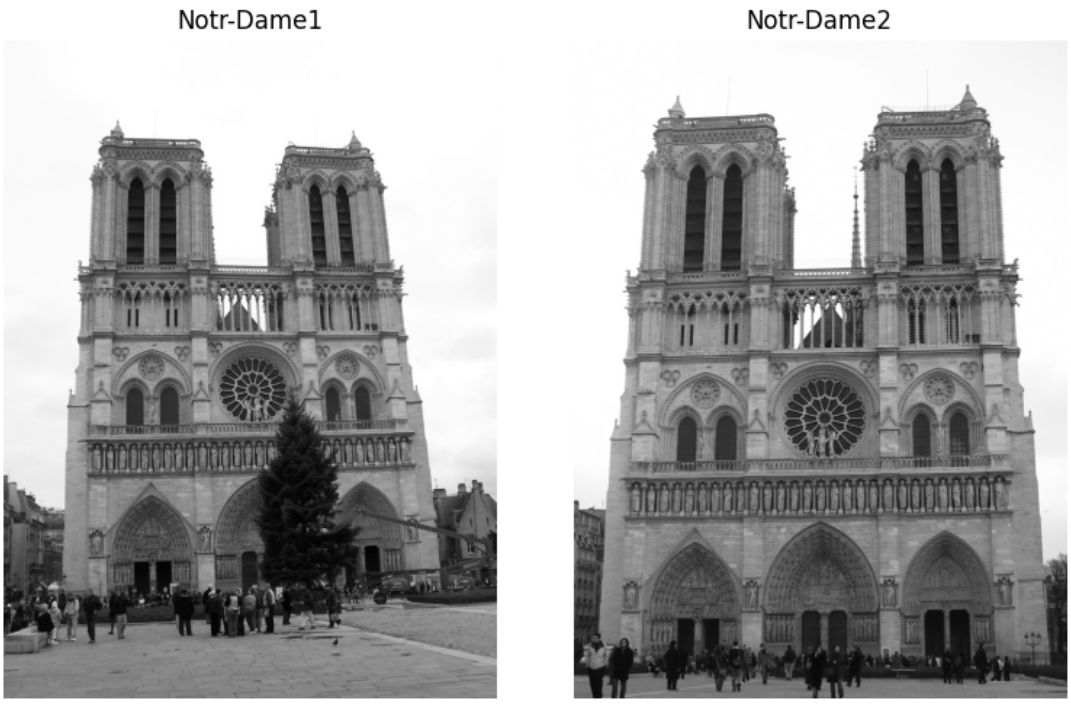


Task 2: Feature Detection and Matching

Objective: Compare SIFT and Harris features across image pairs.

Convert the paired images to grayscale

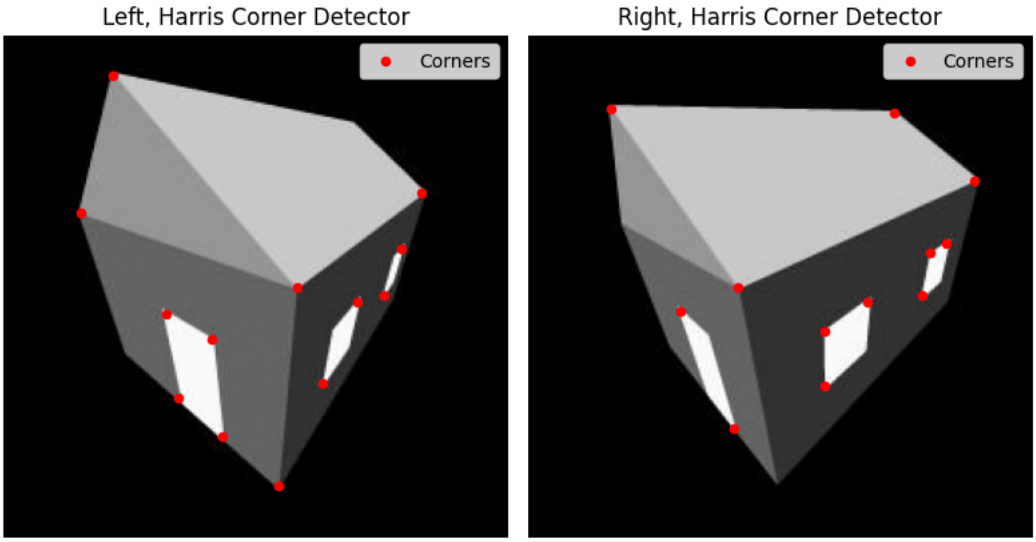


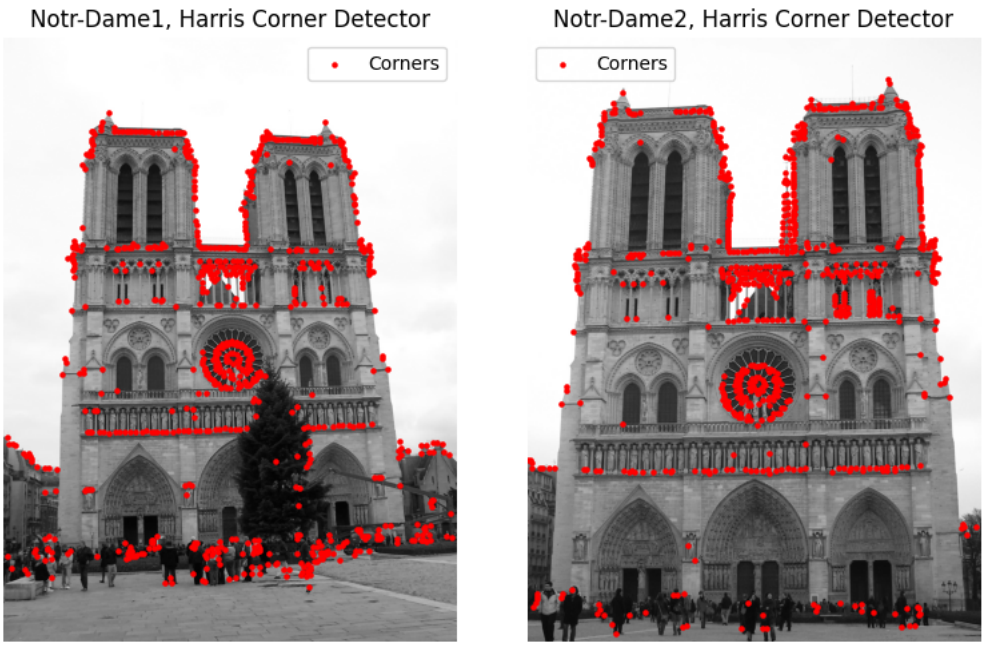


We will manipulate with these paired images for testing Harris-SIFT method of detector-descriptor and SIFT-SIFT.

Detect corners by Harris Corner detector

We will use the previous customyzed Harris function, but this time we will detect only corners on the image, edges are not of the interest at this task. For the house we get the same picture as before.

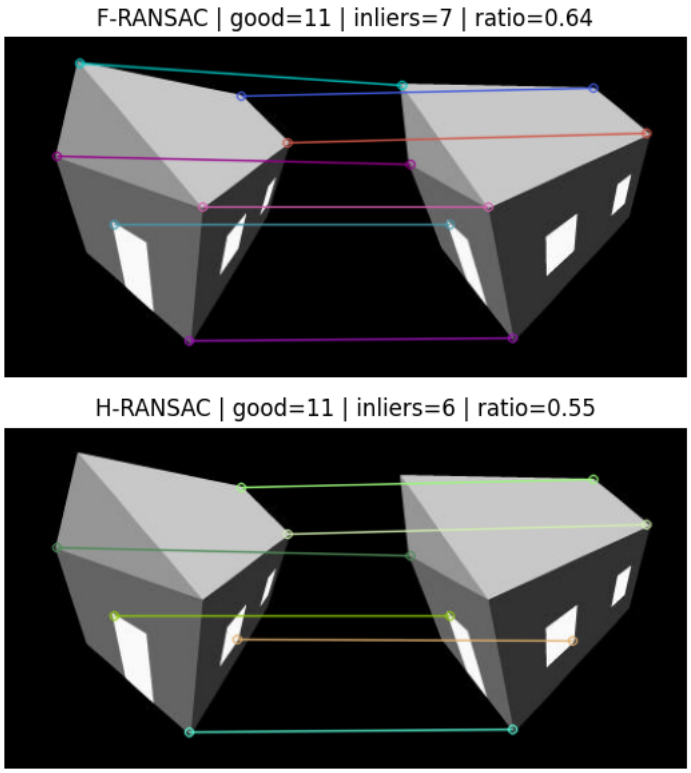




For the Notre-Dame images, many corners are detected along the building's edges, windows, and detailed structures, as well as in the crowd. However, the lower part of the building has fewer detected corners due to lower texture, shadows, and uniform areas. This shows that the Harris detector performs best on regions with strong intensity variations and clear geometric features.

We will match Harris detector with SIFT descriptor and use metrics for results:

- **good** — number of feature pairs that pass the descriptor similarity test (SIFT + L2 + Lowe ratio).
- **inliers** — subset of those matches that are consistent with geometric constraints found by RANSAC.
- **inlier ratio = inliers / good** — precision: how many tentative matches are geometrically correct.
- **absolute inliers** — recall: total number of reliable correspondences available for further use.

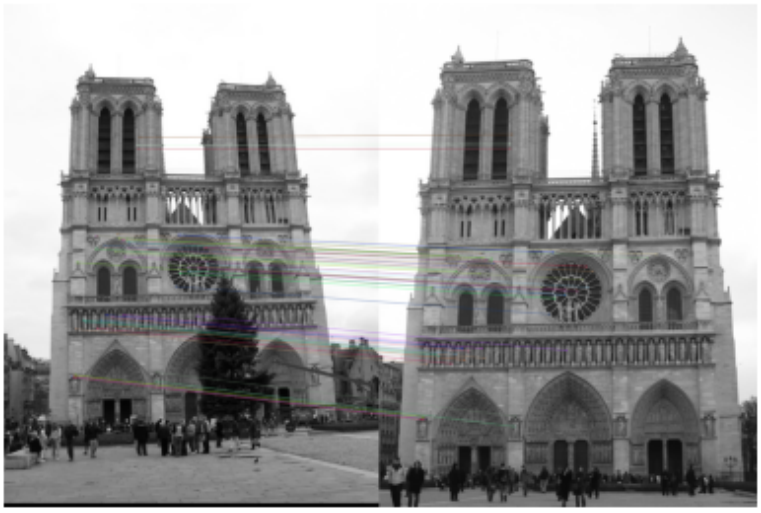


F-RANSAC performs better than H-RANSAC (ratio ≈ 0.64 vs 0.55) because the house scene is not planar**.**

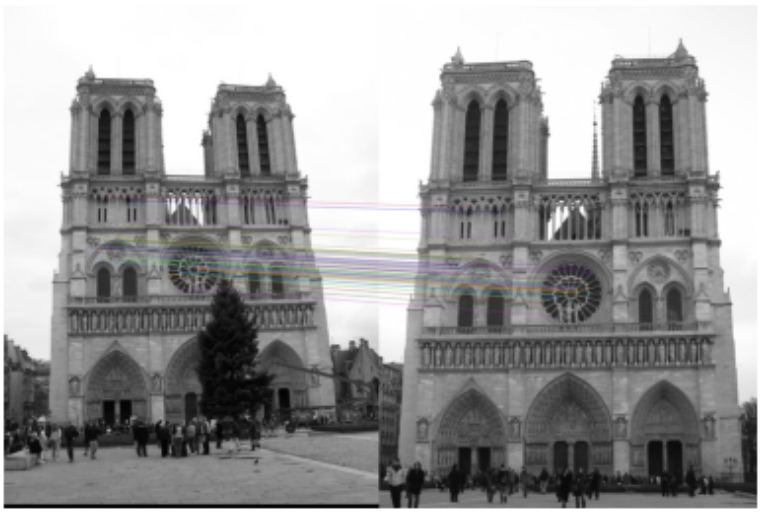
The **fundamental matrix (F)** correctly models the 3D epipolar geometry, preserving more valid matches as inliers. In contrast, **homography (H)** assumes a flat surface or pure camera rotation, which underfits the 3D structure and leads to more mismatches.

We check Harris-SIFT for Notre-Dame images:

Notre-Dame: F-RANSAC | good=540 | inliers=296 | ratio=0.55



Notre-Dame: H-RANSAC | good=540 | inliers=230 | ratio=0.43

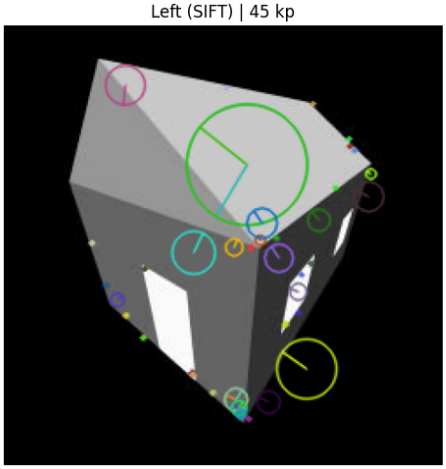


Although the Notre-Dame facade is nearly planar, the scene includes non-planar elements such as the tree, people, and background, along with a slight viewpoint change. These factors break the single-plane assumption of homography, so **F-RANSAC**, which models full 3D epipolar geometry, achieves a higher inlier ratio and more reliable matches than **H-RANSAC**.

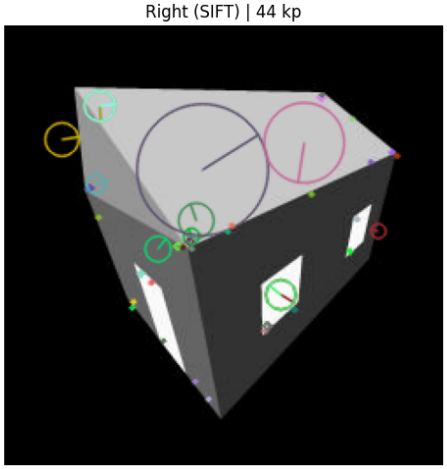
SIFT as detector:

Shows gradients pretty similar to the areas and amount of the gradient change, just in a different way than Harris does.

Left (SIFT) | 45 kp



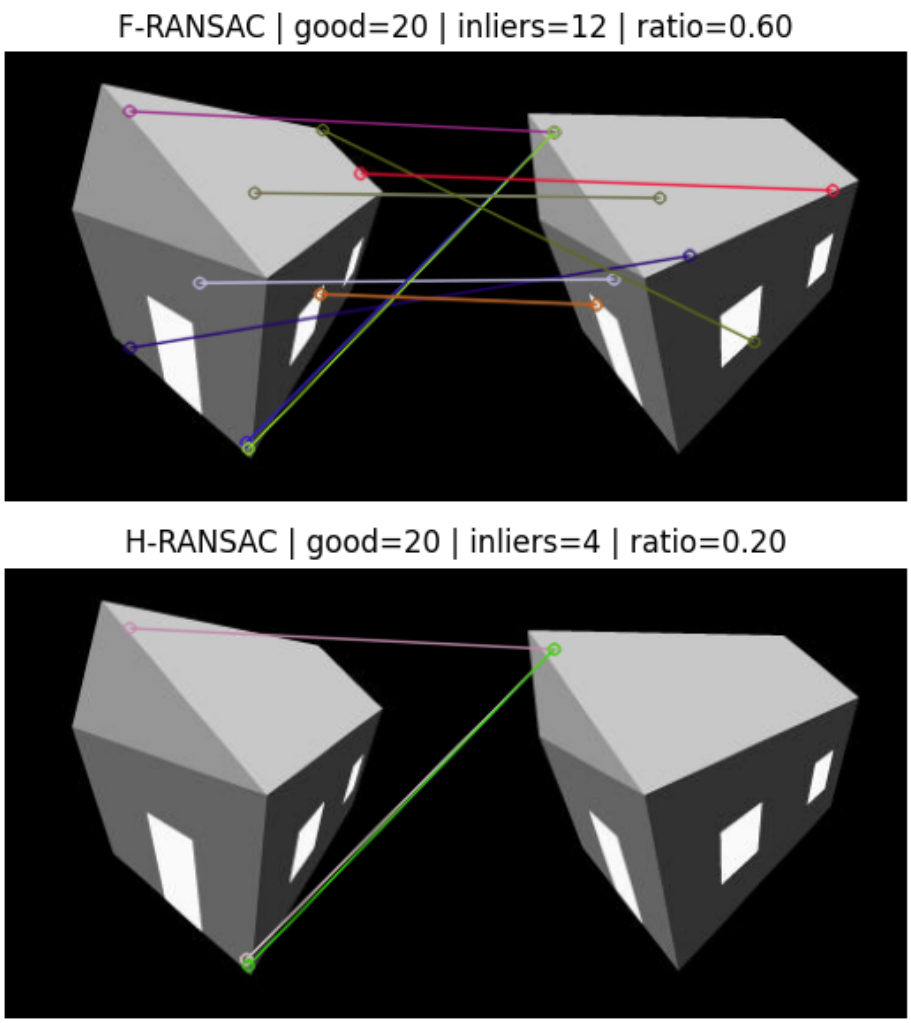
Right (SIFT) | 44 kp



ND1 (SIFT) | 20369 kp

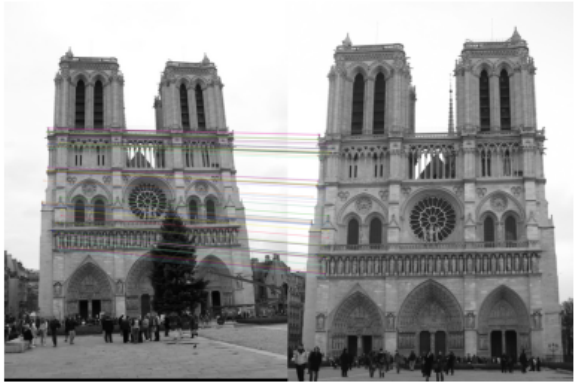
ND2 (SIFT) | 24549 kp

We will match SIFT detector and SIFT descriptor:

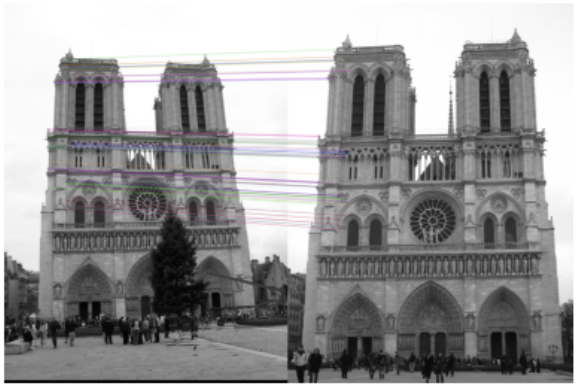


In this example, SIFT–SIFT matching yields more total correspondences but also introduces more noise, resulting in a lower inlier ratio unless filtered by F-RANSAC. Harris+SIFT**,** on the other hand, detects fewer but stronger corner points, giving cleaner matches and a higher precision ratio, though with fewer overall inliers.

Notre-Dame: F-RANSAC | good=1665 | inliers=789 | ratio=0.47



Notre-Dame: H-RANSAC | good=1665 | inliers=400 | ratio=0.24



In the Notre-Dame example, both methods detect a large number of tentative matches, but F-RANSAC keeps nearly twice as many inliers (789 vs. 400) and achieves a higher inlier ratio (0.47 vs. 0.24). This shows that the fundamental matrix (F) correctly models the 3D geometry and parallax between the two views, while the homography (H) oversimplifies the scene by assuming a single plane, leading to more mismatches and rejected points.

Detector–Descriptor:

- **Harris+SIFT** detects fewer but stronger corners → higher precision (inlier ratio), fewer total matches.
- **SIFT–SIFT** detects many features across scales → more absolute inliers but lower ratio on repetitive or low-texture scenes.

Model:

- **F-RANSAC** fits general 3D geometry with parallax → better for real scenes like the house or Notre-Dame.
- **H-RANSAC** assumes a single plane or pure rotation → works only for flat, planar views.

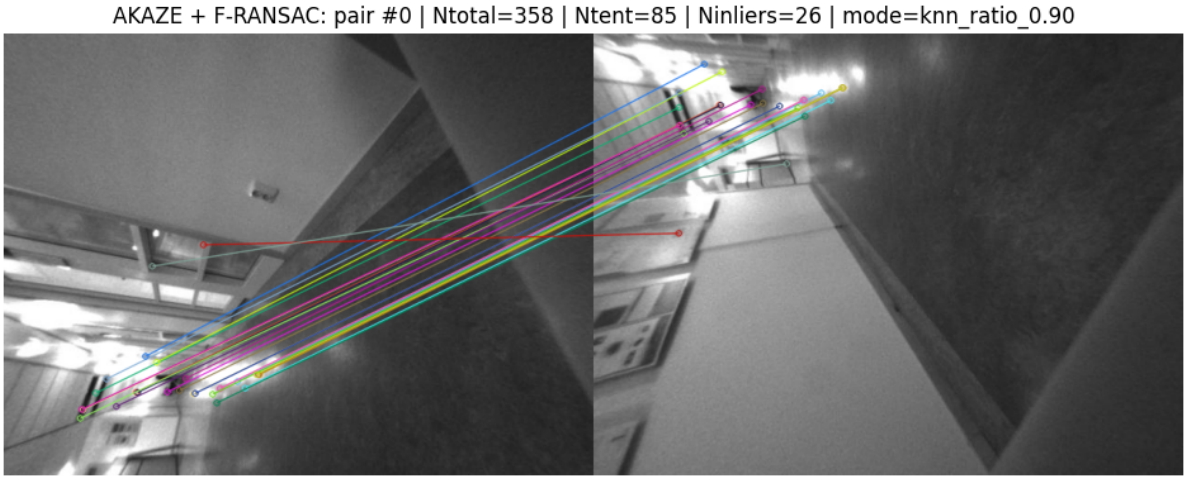
Notre-Dame and the house scenes are not perfectly planar and contain depth variations. Thus, F-RANSAC keeps more valid correspondences, while H-RANSAC rejects many due to violated planar assumptions.

Task 3: Robotic Scenario – Stereo Matching

Objective: Apply feature matching to stereo video feeds.

Method: AKAZE detector + binary MLDB descriptor + F-RANSAC for geometric filtering.

First data set has rotation between camears views and images.




```
[AKAZE + F-RANSAC] FPS=21.8 | ⟨Ntotal⟩=338.6 | ⟨Ntent⟩=82.8 | ⟨Ninliers⟩=19.1 |
inliers/tentatives=0.23 | inliers/total=0.06 | zero-inlier frames=0/240
```

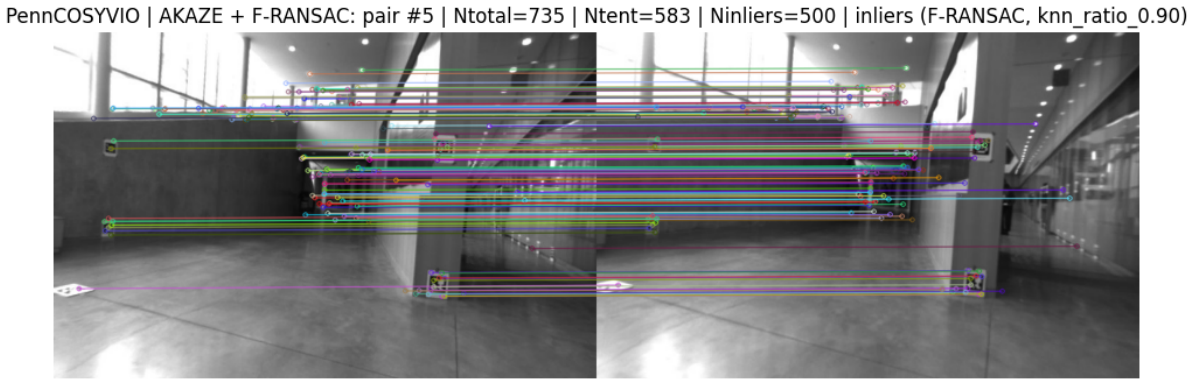
- **Performance:** ~22 FPS (real-time capable), ~340 detected features, ~80 tentative matches, ~19 geometrically valid inliers.
- **Precision:** inliers/tentatives ≈ 0.23 — typical for low-texture, unrectified scenes.
- **Stability:** zero-inlier frames = 0 — tracking remains consistent across the full sequence.

The modest inlier ratio is expected: corridor scenes have repetitive geometry and weak texture, causing ambiguous matches that F-RANSAC must reject.

Because the cameras are uncalibrated and unrectified, epipolar constraints cannot simplify the search, so only strict geometric verification keeps true stereo correspondences.

Overall, these results are normal for indoor robotic stereo with AKAZE + F-RANSAC on unrectified data — stable, but sparse and low-precision matching.

The second data set consists of the images where cameras have the same position.




```
[PennCOSYVIO | AKAZE + F-RANSAC] FPS=20.3 | ⟨Ntotal⟩=1160.2 | ⟨Ntent⟩=849.5 |
⟨Ninliers⟩=688.5 | ⟨Nfailed⟩=471.6 | inliers/tentatives=0.81 | inliers/total=0.59 |
zero-inlier frames=0/131
```

- **Performance:** ~20 FPS; ≈ 1160 detected features; ≈ 850 tentative matches; ≈ 690 geometrically valid inliers.
- **Precision:** inliers/tentatives ≈ 0.81 — very high consistency; zero-inlier frames = 0 → fully stable pipeline.

This dataset is ideal for stereo matching - the cameras are calibrated, rectified, and synchronized**,** with minimal rotation and rich indoor texture.

As a result, F-RANSAC retains most correspondences, giving a strong inlier ratio and reliable geometry. Residual outliers come mainly from occlusions, reflections, and border regions.

Accurate camera calibration ensures horizontal epipolar alignment, enabling efficient and precise stereo matching — exactly why the results here are significantly better than in the uncalibrated corridor dataset.