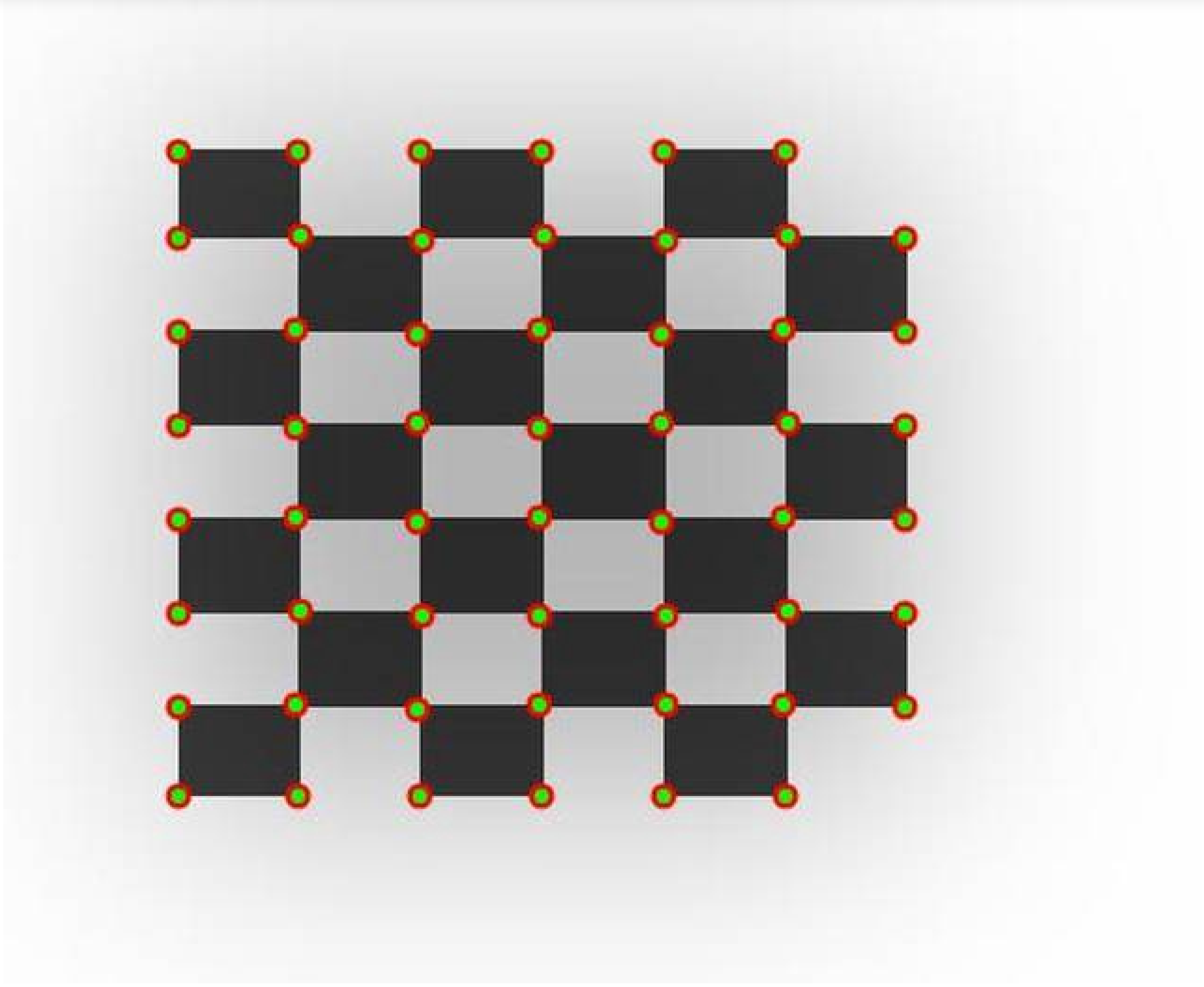
Published in Data Breach

Deepanshu Tyagi Follow

Mar 16, 2019 · 5 min read · Listen

Introduction to Harris Corner Detector

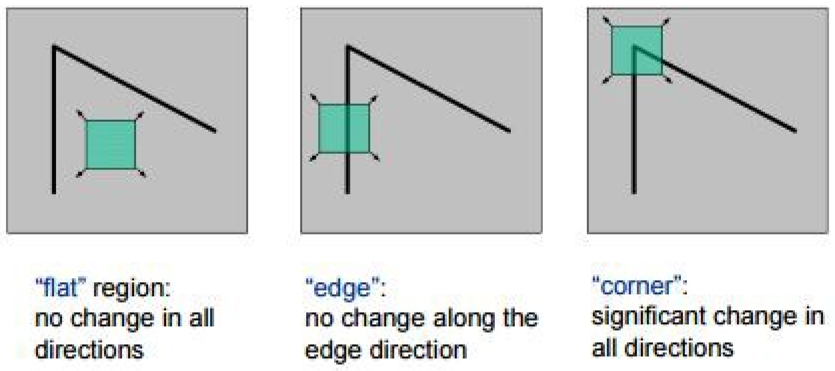
Harris Corner Detector is a corner detection operator that is commonly used in computer vision algorithms to extract corners and infer features of an image. It was first introduced by Chris Harris and Mike Stephens in 1988 upon the improvement of Moravec's corner detector. Compared to the previous one, Harris' corner detector takes the differential of the corner score into account with reference to direction directly, instead of using shifting patches for every 45-degree angles, and has been proved to be more accurate in distinguishing between edges and corners. Since then, it has been improved and adopted in many algorithms to preprocess images for subsequent applications.



This is part of a 7-series Feature Detection and Matching. Other articles included

- Introduction To Feature Detection And Matching
- Introduction to SIFT (Scale-Invariant Feature Transform)
- Introduction to SURF (Speeded-Up Robust Features)
- Introduction to FAST (Features from Accelerated Segment Test)
- Introduction to BRIEF (Binary Robust Independent Elementary Features)
- Introduction to ORB (Oriented FAST and Rotated BRIEF)

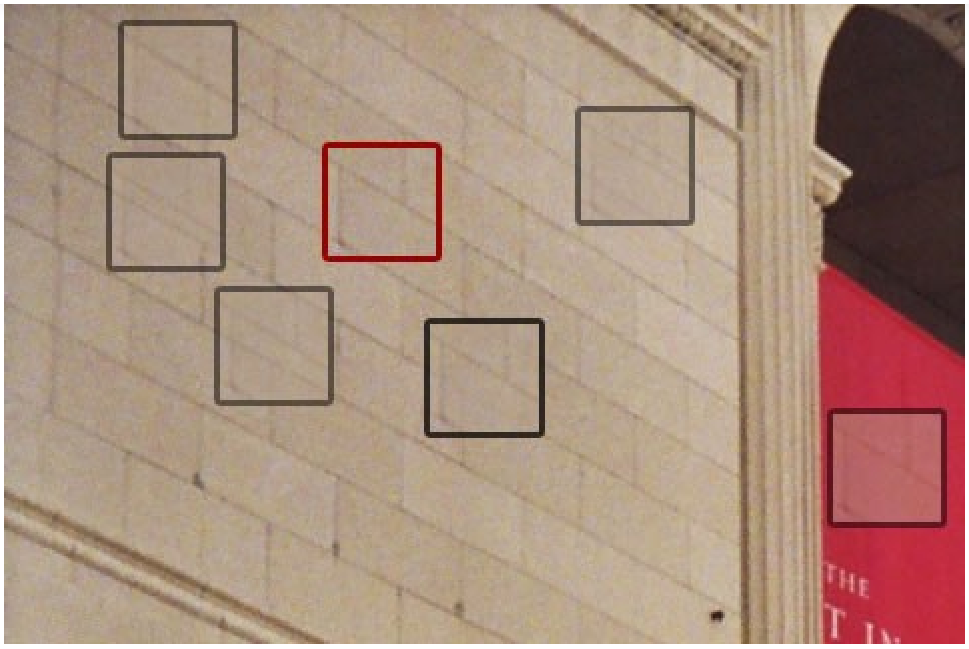
an edge is a sudden change in image brightness. Corners are the important features in the image, and they are generally termed as interest points which are invariant to translation, rotation, and illumination.


"flat" region:
no change in all directions
"edge":
no change along the edge direction
"corner":
significant change in all directions

So let's understand why corners are considered better features or good for patch mapping. In the above figure, if we take the flat region then no gradient change is observed in any direction. Similarly, in the edge region, no gradient change is observed along the edge direction. So both flat region and edge region are bad for patch matching since they not very distinctive (there are many similar patches in along edge in edge region). While in corner region we observe a significant gradient change in all direction. Due this corners are considered good for patch matching(shifting the window in any direction yield a large change in appearance) and generally more stable over the change of viewpoint.

Corner Detection

The idea is to consider a small window around each pixel p in an image. We want to identify all such pixel windows that are unique. Uniqueness can be measured by shifting each window by a small amount in a given direction and measuring the amount of change that occurs in the pixel values.



More formally, we take the sum squared difference (SSD) of the pixel values before and after the shift and identifying pixel windows where the SSD is large for shifts in all 8 directions. Let us define the change function E(u,v) as the **sum** of all the sum squared differences (SSD), where u,v are the x,y coordinates of every pixel in our 3 x 3 window and I is the intensity value of the pixel. The features in the image are all pixels that have large values of E(u,v), as defined by some threshold.

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u,y+v) - I(x,y)]^2$$

The equation

We have to maximize this function *E*(*u*,*v*) for corner detection. That means, we have to maximize the second term. Applying Taylor Expansion to the above equation and using some mathematical steps, we get the final equation as:

$$E(u,v) \approx [u \quad v] \left(\sum \begin{bmatrix} I_x{}^2 & I_x I_y \\ I_x I_y & I_y{}^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum w(x,y) \begin{bmatrix} I_x{}^2 & I_x I_y \\ I_x I_y & I_y{}^2 \end{bmatrix}$$

So the equation now becomes:

$$E(u,v) \approx [u \quad v]\, M \begin{bmatrix} u \\ v \end{bmatrix}$$

Remember that we want the SSD to be large in shifts for all eight directions, or conversely, for the SSD to be small for none of the directions. By solving for the eigenvectors of M, we can obtain the directions for both the largest and smallest increases in SSD. The corresponding eigenvalues give us the actual value amount of these increases. A score, R, is calculated for each window:

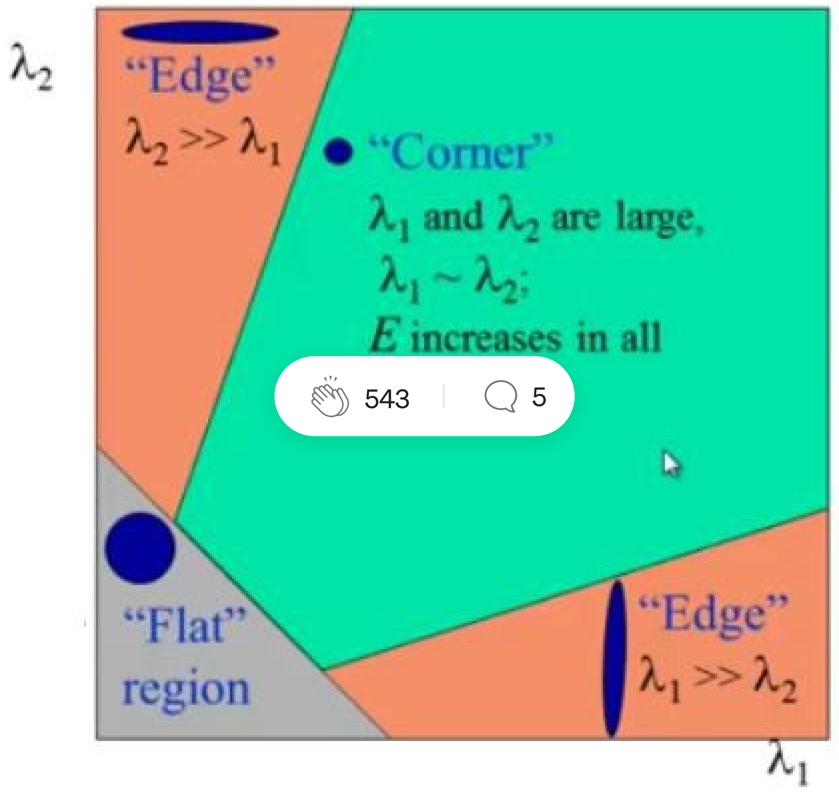
$$R = \det M - k(trace\, M)^2$$

$$\det M = \lambda_1 \lambda_2$$

$$trace\, M = \lambda_1 + \lambda_2$$

$\lambda 1$ and $\lambda 2$ are the eigenvalues of M. So the values of these eigenvalues decide whether a region is a corner, edge or flat.

- When $|R|$ is small, which happens when $\lambda 1$ and $\lambda 2$ are small, the region is flat.
- When $R<0$, which happens when $\lambda 1>>\lambda 2$ or vice versa, the region is an edge.
- When R is large, which happens when $\lambda 1$ and $\lambda 2$ are large and $\lambda 1 \sim \lambda 2$, the region is a corner.



High-level pseudocode

1. Take the grayscale of the original image

2. Apply a Gaussian filter to smooth out any noise

3. Apply Sobel operator to find the x and y gradient values for every pixel in the grayscale image

4. For each pixel p in the grayscale image, consider a 3×3 window around it and compute the corner strength function. Call this its Harris value.

5. Find all pixels that exceed a certain threshold and are the local maxima within a certain window (to prevent redundant dupes of features)

6. For each pixel that meets the criteria in 5, compute a feature descriptor.

Harris Corner Detection

Import resources and display image

In [1]:

```
import matplotlib.pyplot as plt
import numpy as np
import cv2

%matplotlib inline

# Read in the image
image = cv2.imread('images/waffle.jpg')

# Make a copy of the image
image_copy = np.copy(image)

# Change color to RGB (from BGR)
image_copy = cv2.cvtColor(image_copy, cv2.COLOR_BGR2RGB)

plt.imshow(image_copy)
```

Out[1]:

```
<matplotlib.image.AxesImage at 0x7ffb1f306eb8>
```

harris.ipynb hosted with ❤ by GitHub view raw

Github link for the code: https://github.com/deepanshut041/feature-detection/tree/master/harris

References

- https://docs.opencv.org/3.4/dc/d0d/tutorial_py_features_harris.html
- https://in.udacity.com/course/computer-vision-nanodegree--nd891
- http://aishack.in/tutorials/harris-corner-detector/

Thanks for reading! If you enjoyed it, hit that clap button below and follow Data Breach for more updates