VIT-AP UNIVERSITY

Computer Vision

(Course Code: 4047)

Module-2:Lecture-2: Corner Detection

Gundimeda Venugopal, Professor of Practice, SCOPE

Interest Points

- ❖ A point in an image which has a well-defined position and can be robustly detected and is relevant for higher level processing (a general term in computer vision).
- ❖ Interest points are commonly used by image stabilization and structure from motion applications to track how the image changes from frame to frame
- ❖ Typically associated with a significant change of one or more image properties simultaneously (e.g., intensity, color, texture).

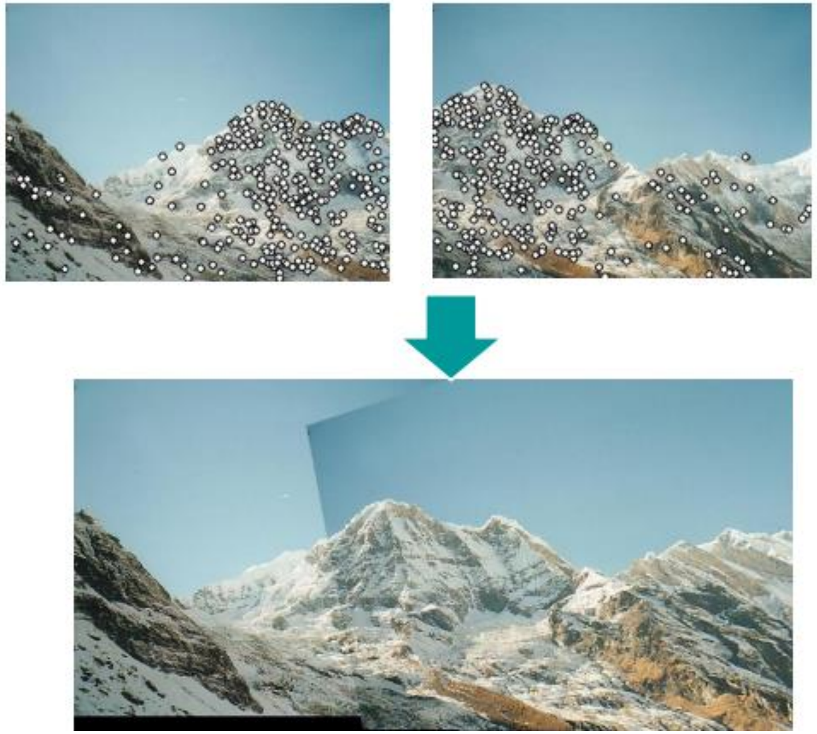
Interest Points: How they are useful?

- Could be used to find corresponding points between images which is very useful for numerous applications!

stereo matching

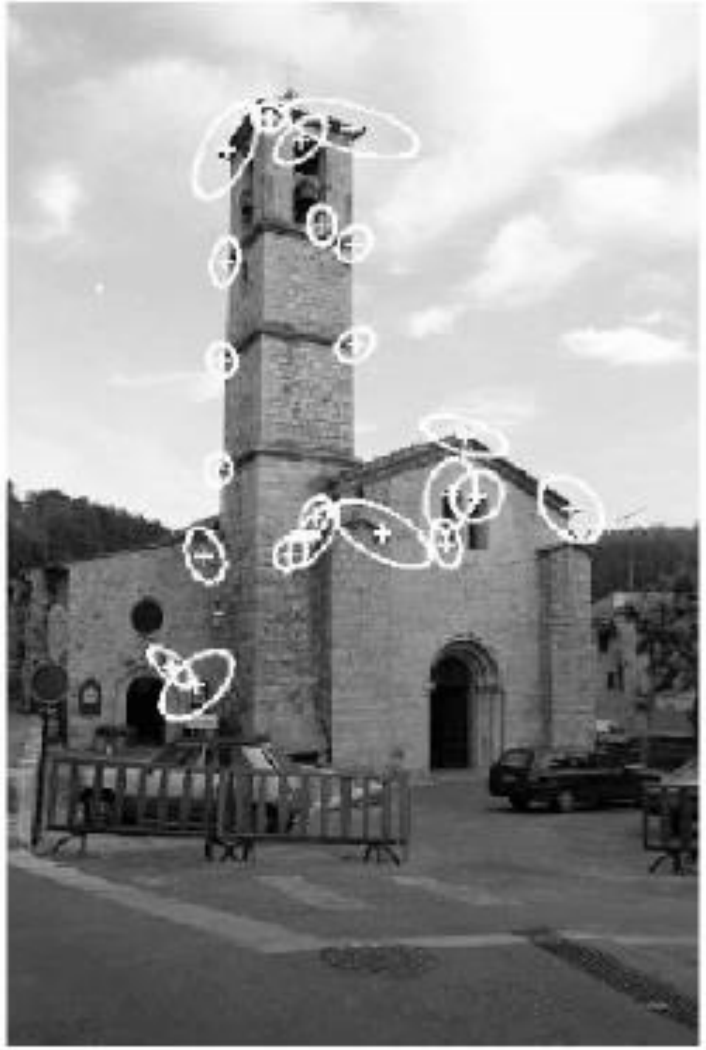
left camera right camera

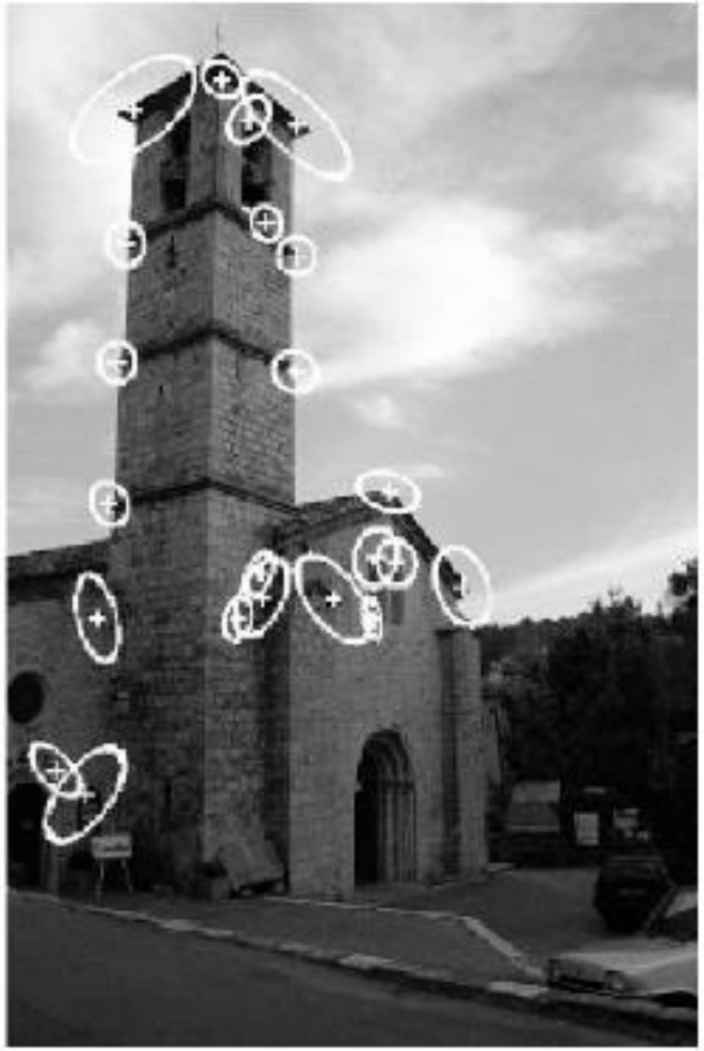
panorama stitching



Motivation: Matching Problem

- Vision tasks such as stereo, panorama stitching and motion estimation require finding corresponding features across two or more views.





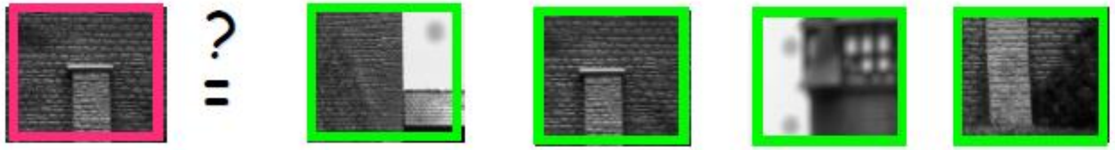
Look at cross marks in those two images , the position of these points(cross marks) haven’t changed even the position of camera changed.

Take a small window around each of these points in both the images.

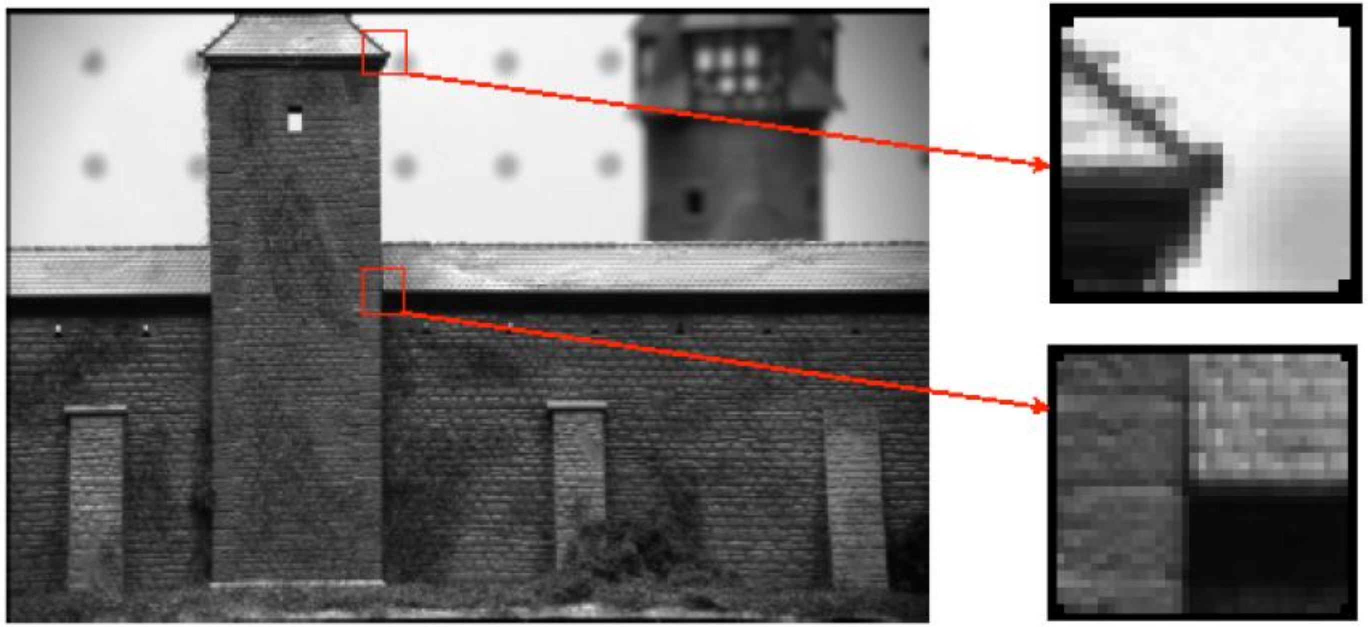
Consider you have a mechanism to map the window in one image to another window in the second image according to their features(feature extraction), then we completed our task of recognizing the objects.

Not all patches are Created Equal

Inituition: this would be a good patch for matching, since it is very distinctive (there is only one patch in the second frame that looks similar).



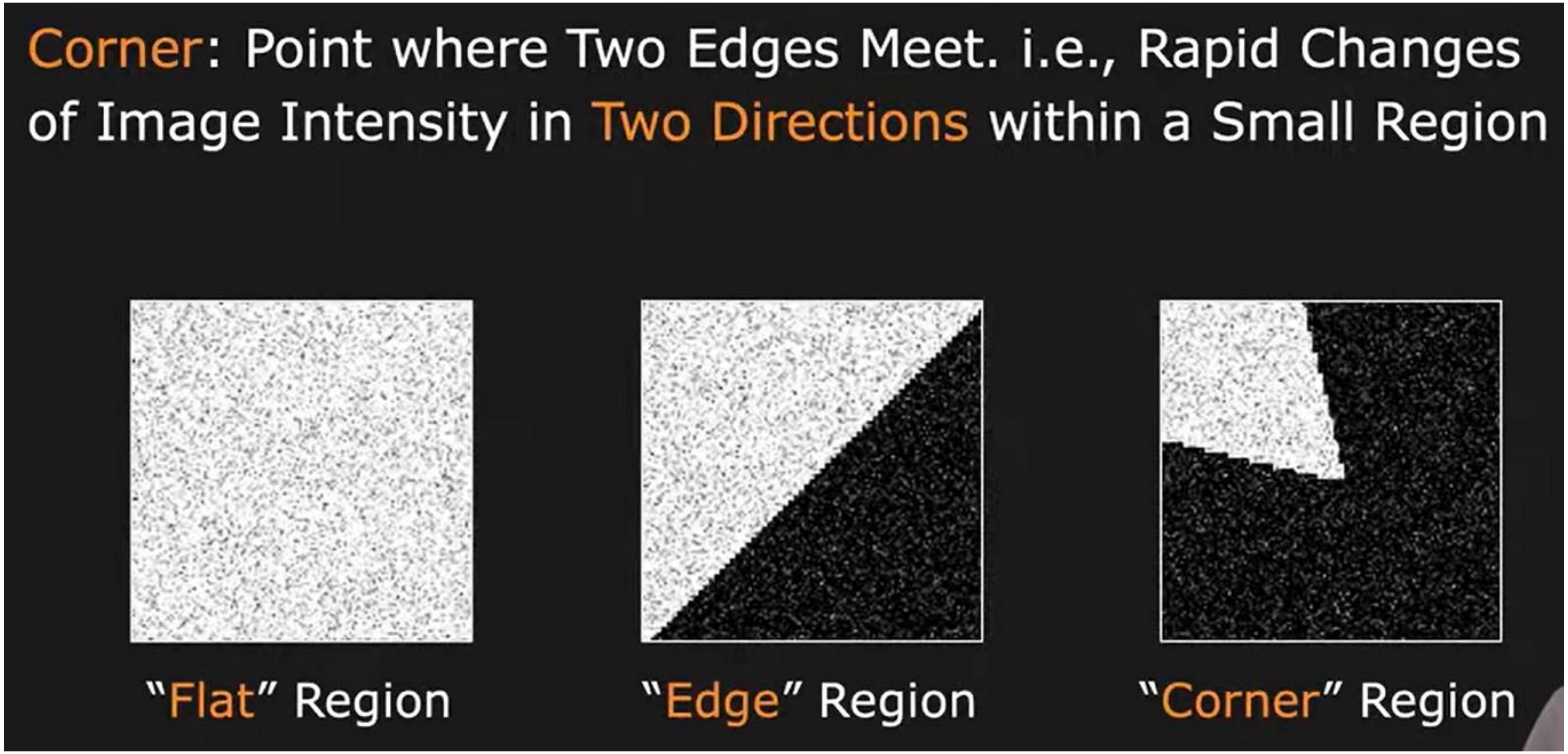
What are Corners?



- Intuitively, junctions of contours.
- Generally more stable features over changes of viewpoint
- Intuitively, large variations in the neighborhood of the point in all directions
- They are good features to match!

Note: Corners is a special case of interest points. However, interest points could be more generic than corners.

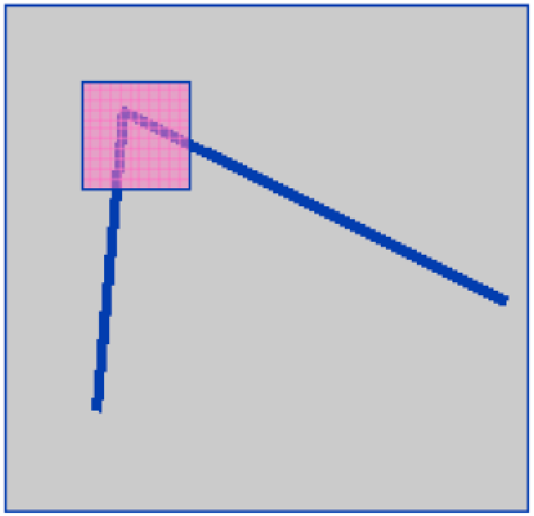
Corners



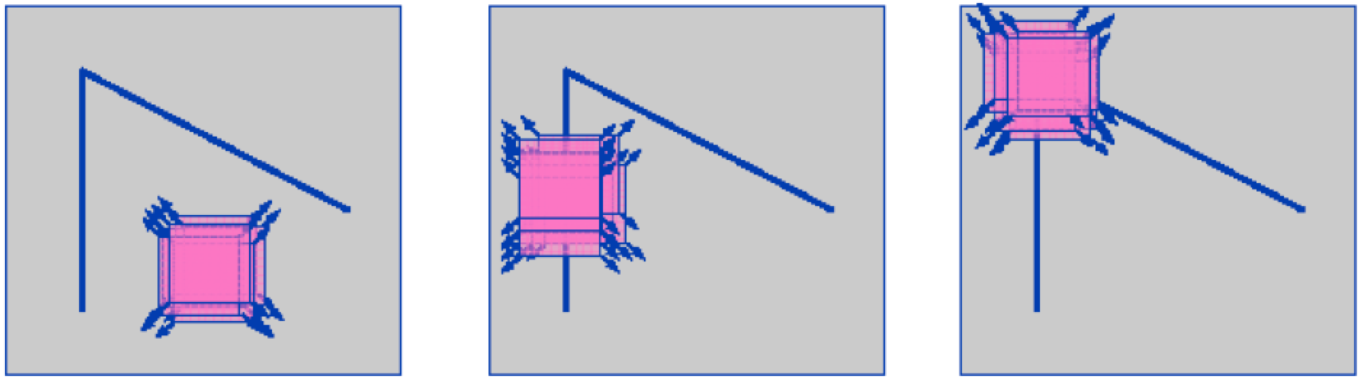
Corner detection is very useful for shape description and matching.

Corner Points: Basic Idea

- We should easily recognize the point by looking at intensity values within a small window
- Shifting the window in *any direction* should yield a *large change* in appearance.



Harris Corner Detector: Basic idea



"flat" region:
no change in
all directions

"edge":
no change along
the edge direction

"corner":
significant change
in all directions

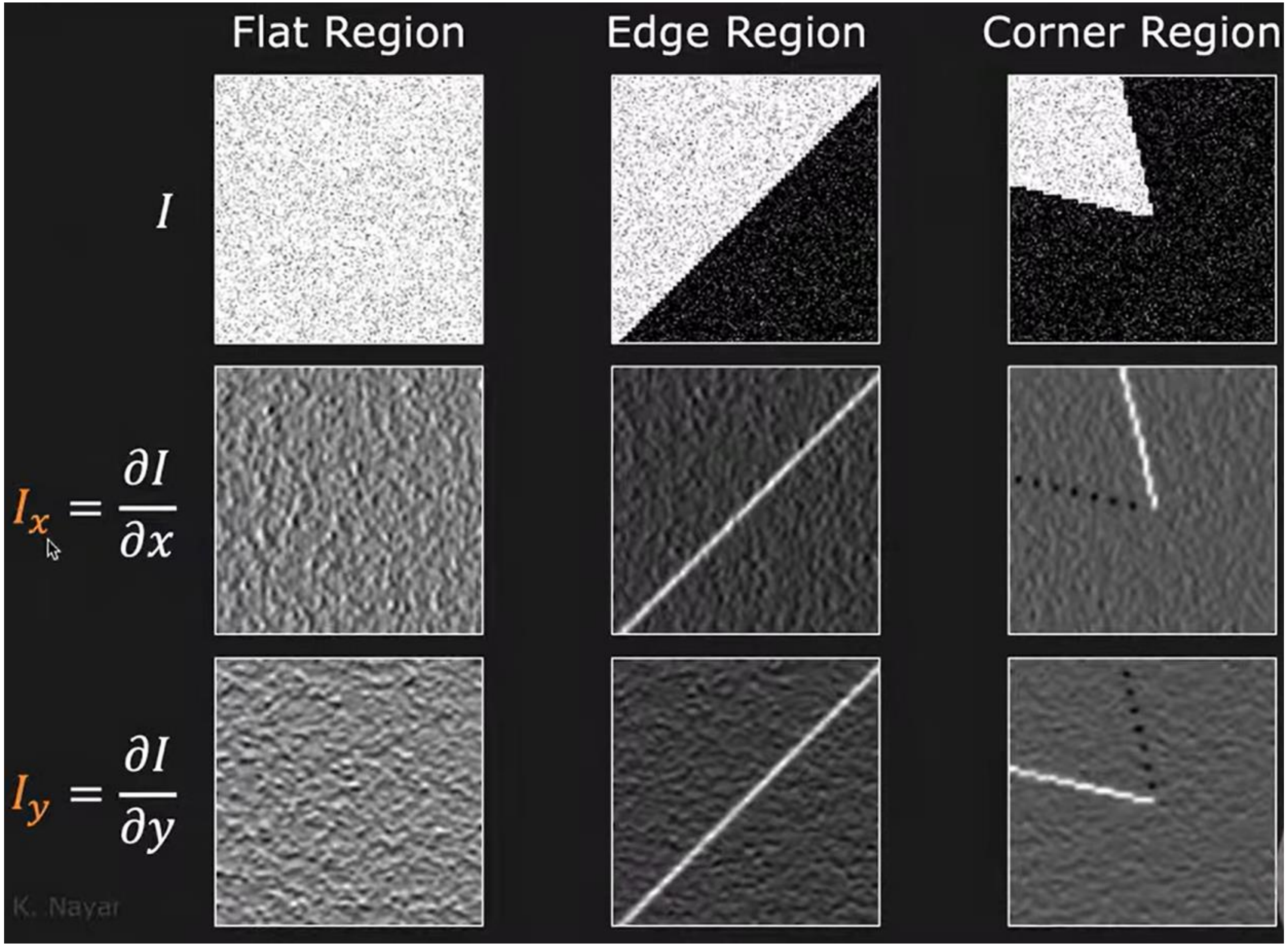
Harris corner detector gives a mathematical approach for determining which case holds.

- Corner detection works on the principle that if you place a small window over an image, if that window is placed on a corner then if it is moved in any direction there will be a large change in intensity.
- If the window is over a flat area of the image then there will be obviously be no intensity change when the window moves.
- If the window is over an edge there will only be an intensity change if the window moves in one direction.

Intuitive way to understand Harris:

- Treat gradient vectors as a set of (dx, dy) points with a center of mass defined as being at (0,0).
- Fit an ellipse to that set of points via scatter matrix.
- Analyze ellipse parameters for varying cases…

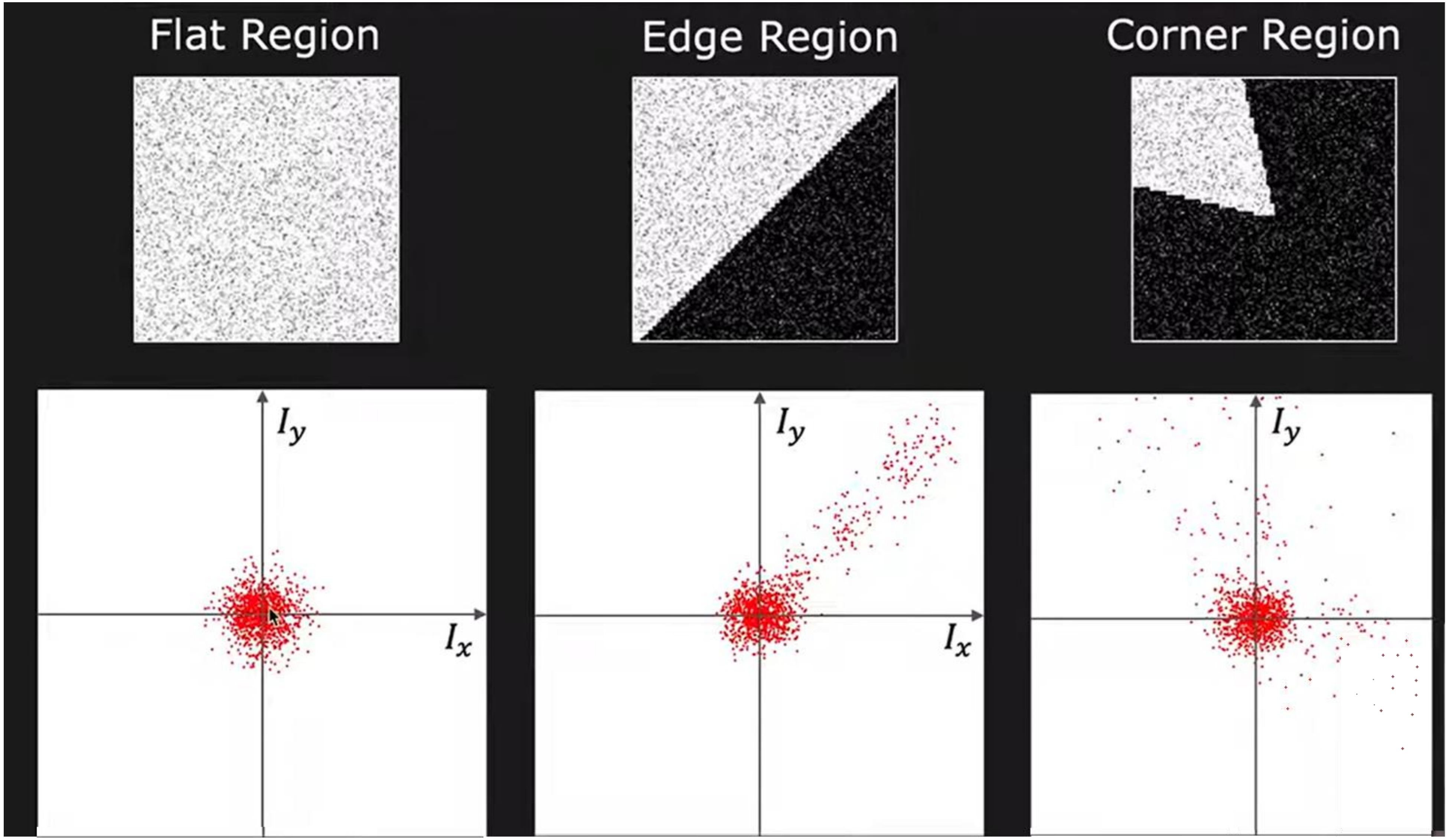
Image Gradients



For the small Window of pixels, Compute Gradient in x-direction (I_x) and Gradient in x-direction (I_y).

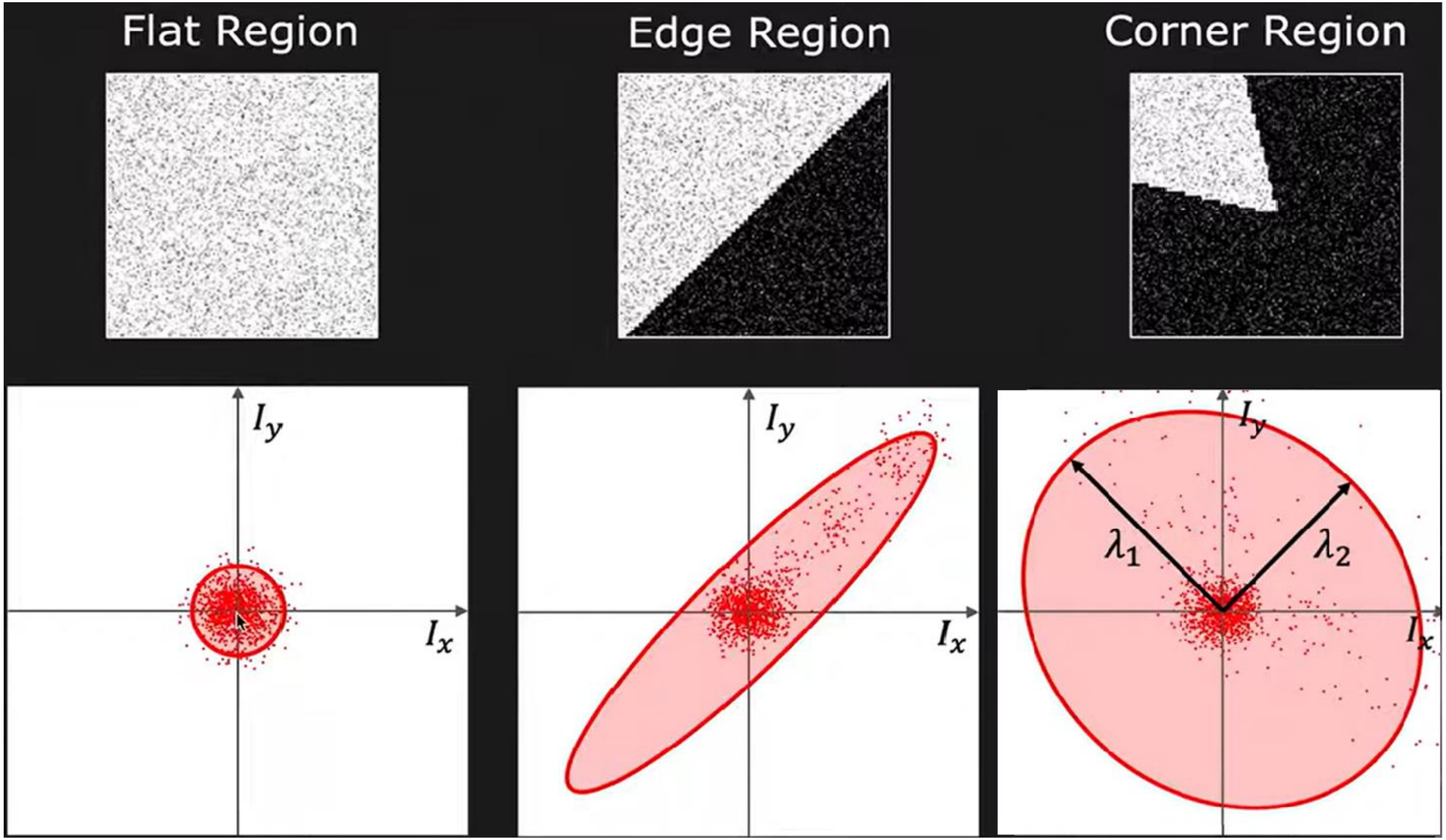
Normalize the output so that white corresponds to a strong +ve value, Black corresponds to a strong Negative value and Grey corresponds to close to zero value

Distribution of Image Gradients



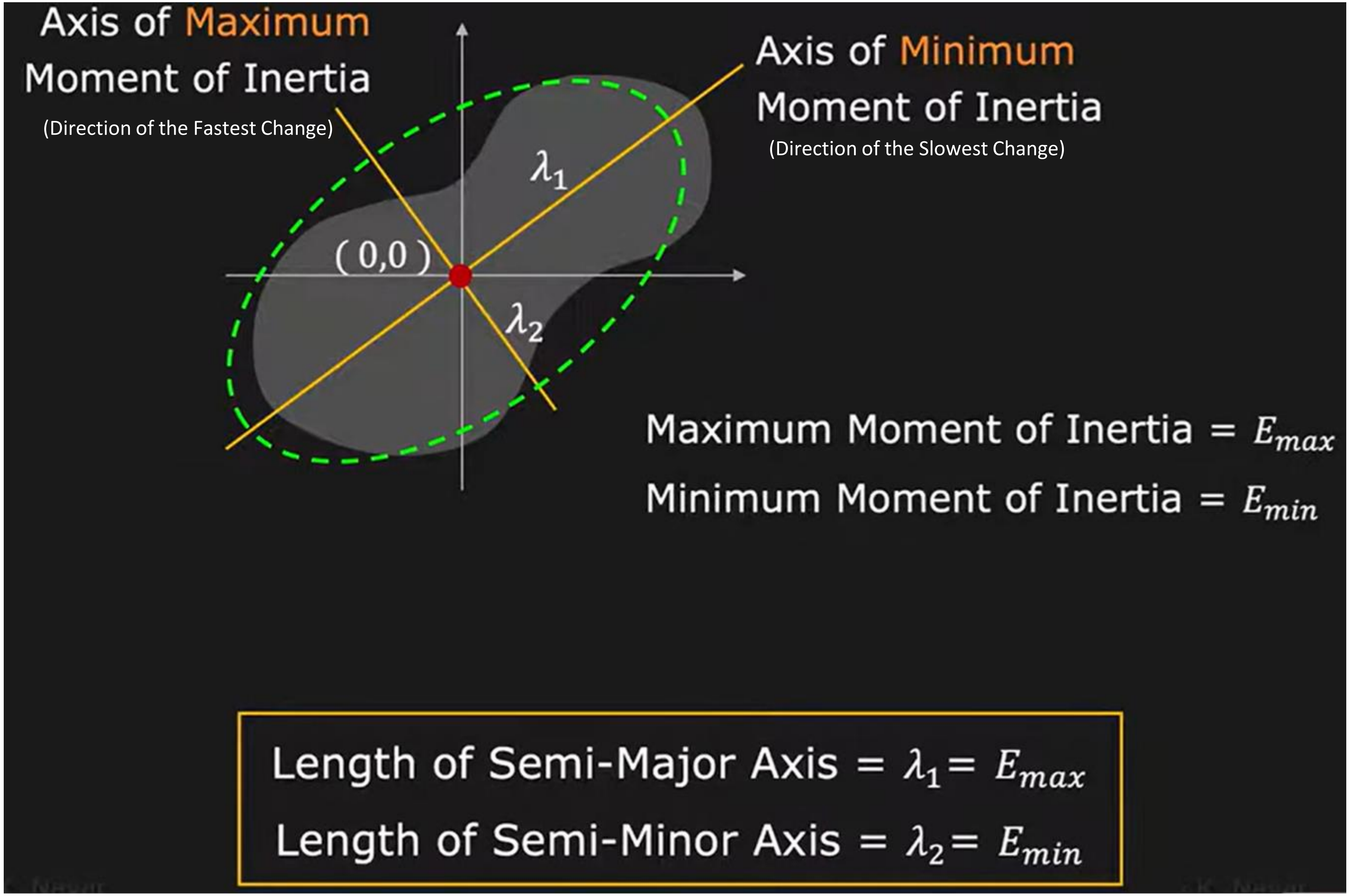
Distribution of I_x and I_y is different for all three regions

Fitting Elliptical Disk to Distribution



Distribution of I_x and I_y is different for all three regions

Fitting an Elliptical disk



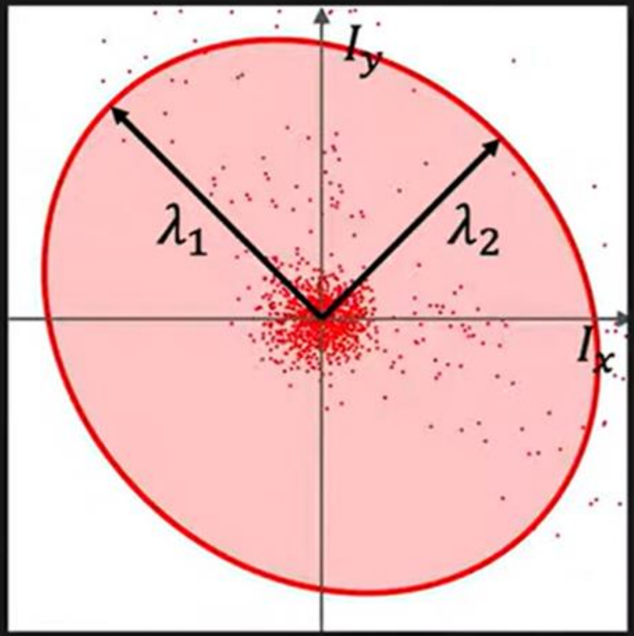
Fitting an Elliptical disk

Second Moments for a Region:

$$a = \sum_{i \in W} \left({I_{x_i}}^2 \right) \qquad b = 2 \sum_{i \in W} \left(I_{x_i} I_{y_i} \right)$$

$$c = \sum_{i \in W} \left({I_{y_i}}^2 \right)$$

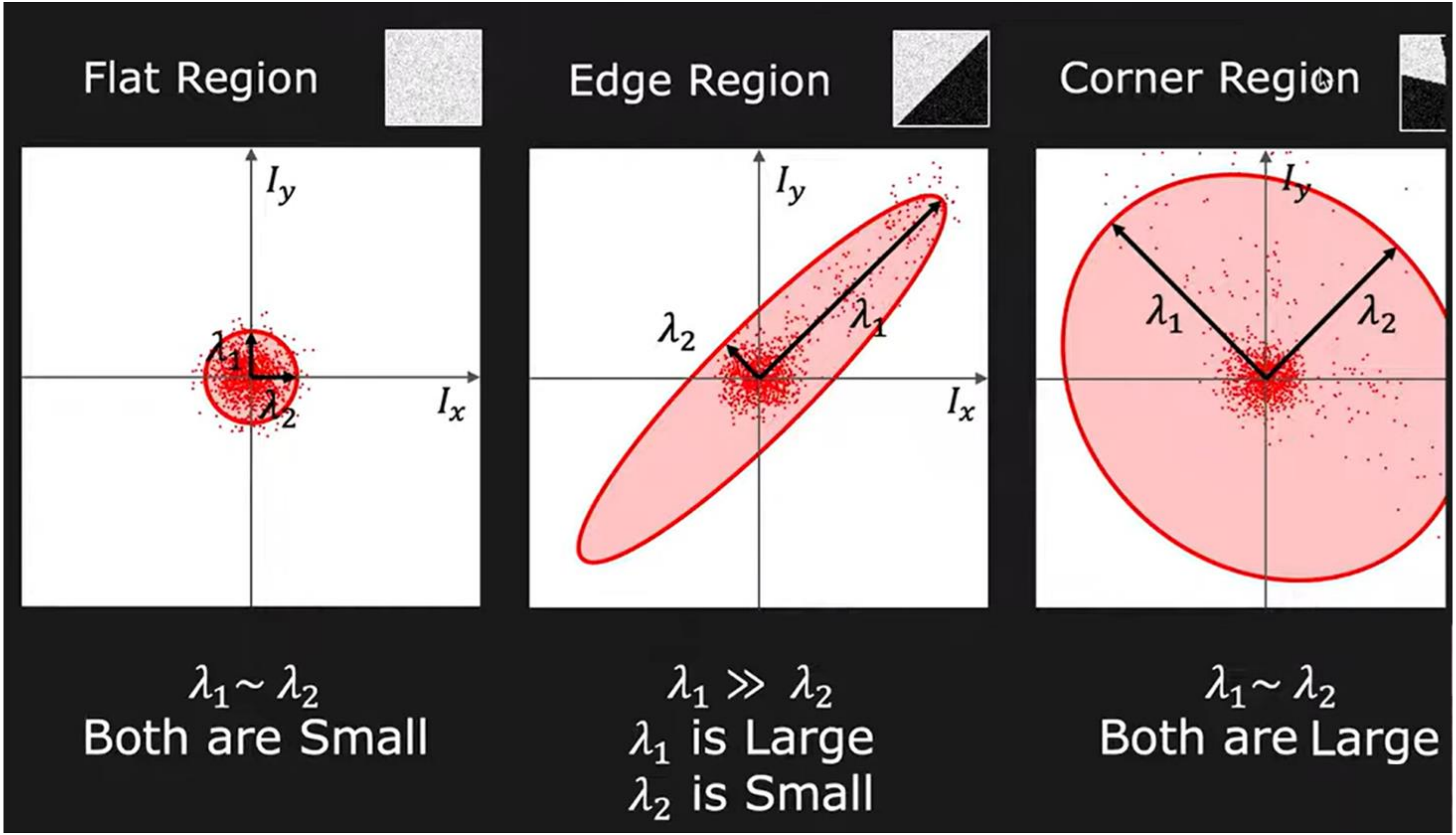
W: Window centered at pixel



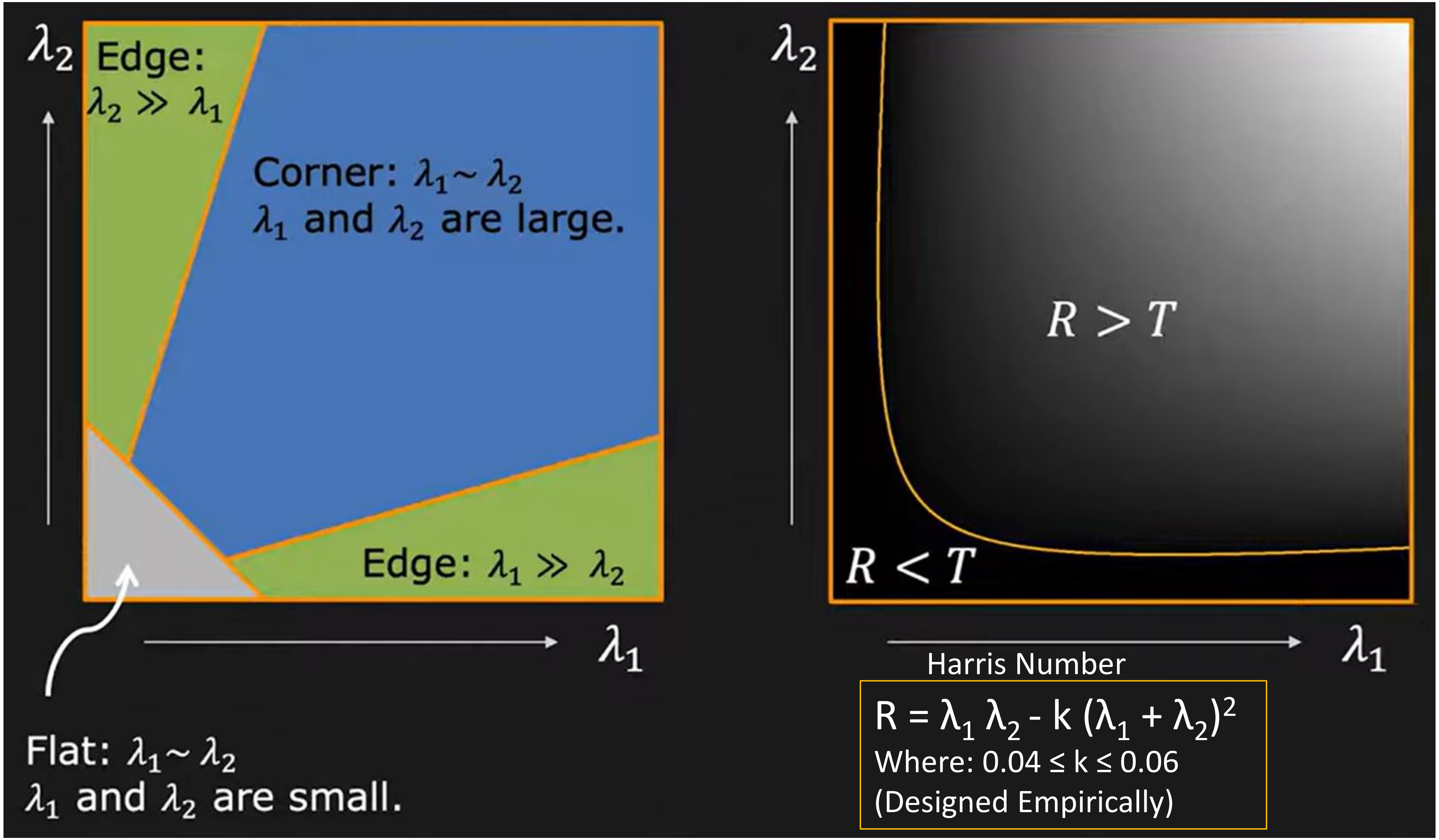
Ellipse Axes Lengths:

$$\boxed{\begin{aligned} \lambda_1 &= E_{max} = \frac{1}{2}\left[a + c + \sqrt{b^2 + (a-c)^2}\right] \\ \lambda_2 &= E_{min} = \frac{1}{2}\left[a + c - \sqrt{b^2 + (a-c)^2}\right] \end{aligned}}$$

Interpretation of λ_1 and λ_2



Harris Corner Response Function



Harris Detector: Second Moment Matrix

❖ The Harris detector starts with this characteristic of corners. It finds pixel-intensity displacement of *(u, v)* such that the function *E* gets maximized for pixels in the window, as is expressed below.

$$E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{window function}} \; [\underbrace{I(x+u, y+v)}_{\text{shifted intensity}} - \underbrace{I(x,y)}_{\text{intensity}}]^2$$

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \; (\textit{approximated using}\ \text{Taylor Expansion})$$

$$\textit{, where} \quad M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

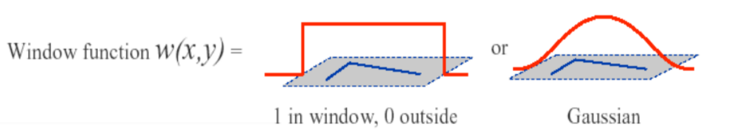
Derivation of the second moment matrix M

Compute the Sum of Squared Differences (SSD) to calculate the intensity variation due to window W shift by a small amount (u, v).

By applying the Taylor Expansion, we can represent *E(u, v)* in a form of matrix multiplication like the second line of the above equations.

What is important here is the matrix ***M*** which is called the second-moment matrix (Covariance Matrix: Symmetrical), where I_x and I_y are x and y-derivative, respectively.

Windowing function: Computing a weighted sum (Simple case, w=1)



Note: These are just products of components of the gradient I_x and I_y

Harris Detector: Second Moment Matrix

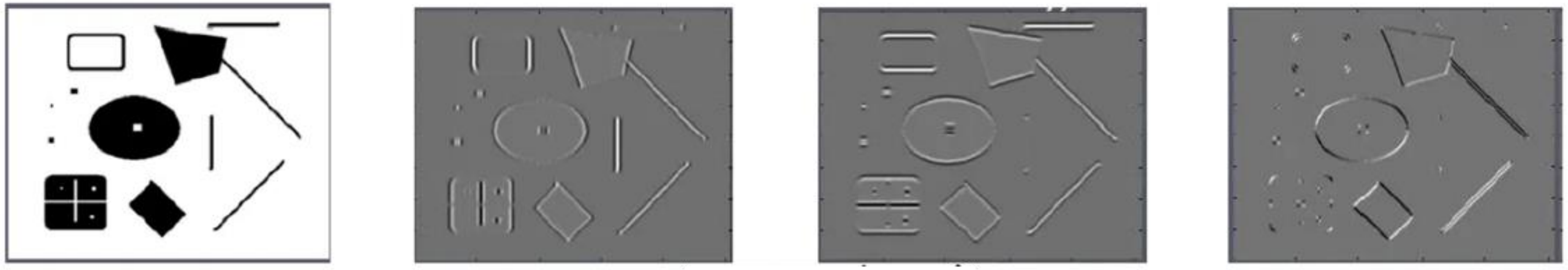


Image I $\quad I_x \quad I_y \quad I_xI_y$

↳ $I_x \cdot I_y$ (product)

①

second-moment matrix $M = \sum\limits_{x,y} \underbrace{w(x,y)}_{\text{window function}} \underbrace{\begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix}}_{\text{compute for pixels inside window}}$

Gradient with respect to x, times gradient with respect to y

② Sum over image region – the area we are checking for corner

$$M = \begin{bmatrix} \sum I_xI_x & \sum I_xI_y \\ \sum I_xI_y & \sum I_yI_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y]$$

chosen window function w(x, y) is the uniform window

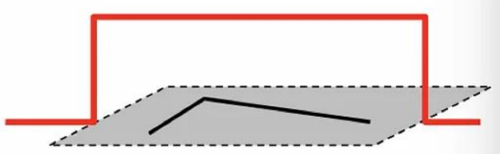
Choice of Window function w(x,y)

Gaussian smoothing vs Uniform window as a window function

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

Option 1: uniform window

$$M = \sum_{x,y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

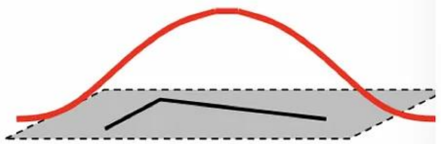


1 in window, 0 outside

Problem: not rotation invariant

Option 2: Smooth with Gaussian

$$M = g(\sigma) * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

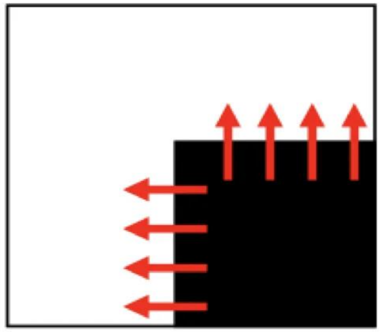


Gaussian

Gaussian already performs weighted sum

Result is rotation invariant

Why the second-moment matrix is important?



$$M = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

Second-moment matrix of a window with axis-aligned corner

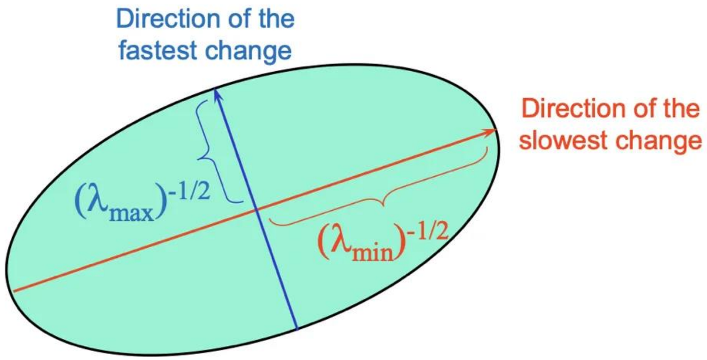
- Second Moment gives us a clue about whether a corner lies inside of the window or not, by looking at the matrix
- The figure contains an axis-aligned corner.
- Then M is guaranteed to be a diagonal matrix, where non-diagonal entries ($I_x \times I_y$) are zero.
  - In vertical edge, $I_y = 0$ as there is no chance in pixel intensity for pixels on the edge.
  - Similarly, in the horizontal edge, $I_x = 0$.
  - In the flat regions, both I_x and I_y are zero as pixel intensities do not change in both x and y directions

Second Moment matrix of a window with non-axis-aligned corner

orthogonal diagonalization

$$M = \underbrace{\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}}_{\text{Symmetric Matrix}} = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

M is symmetric and therefore, it can be orthogonally diagonalizable (recall the characteristic of symmetric matrix regarding diagonalization).

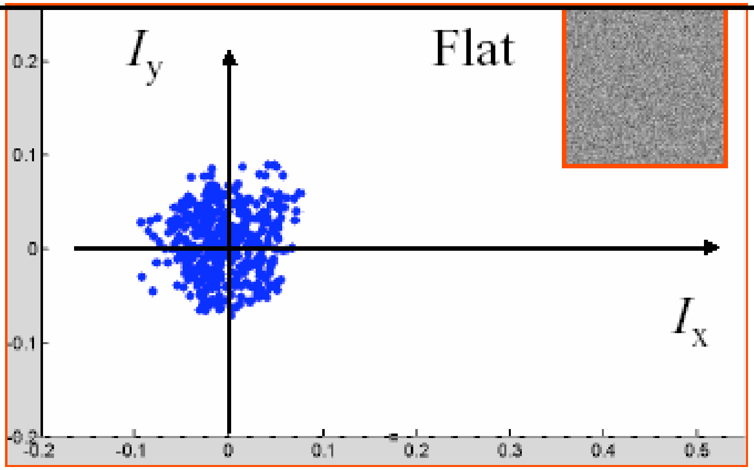


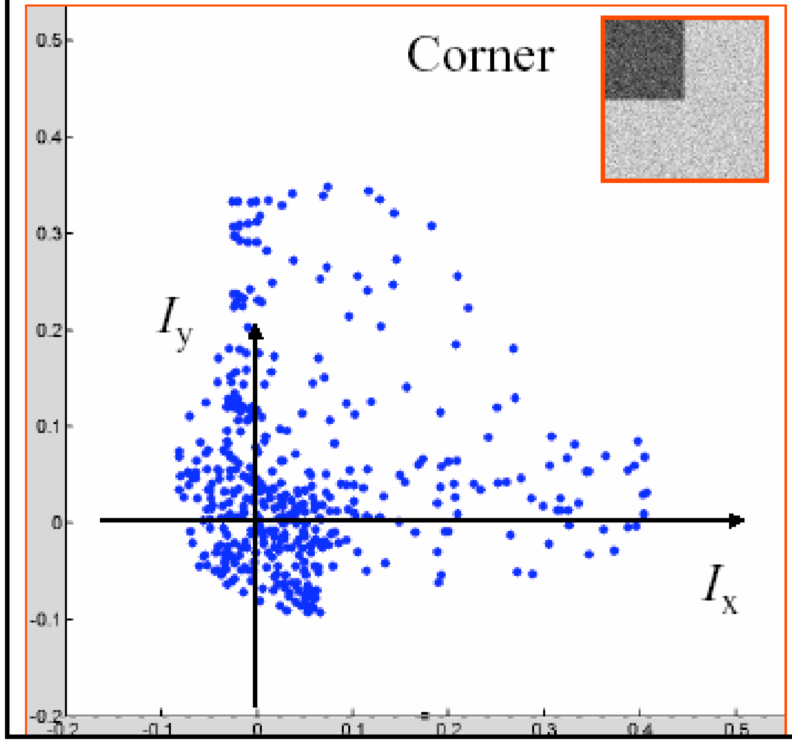
Visualized ***M***

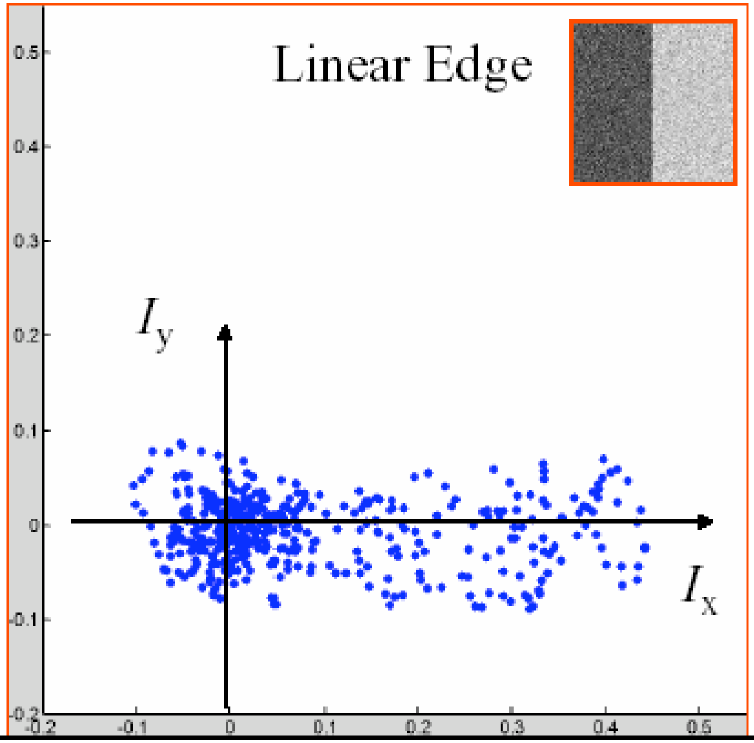
M becomes a product of two rotation(orthogonal) matrices and a diagonal matrix whose entries are the eigenvalues.

Plotting Derivatives as 2D Points

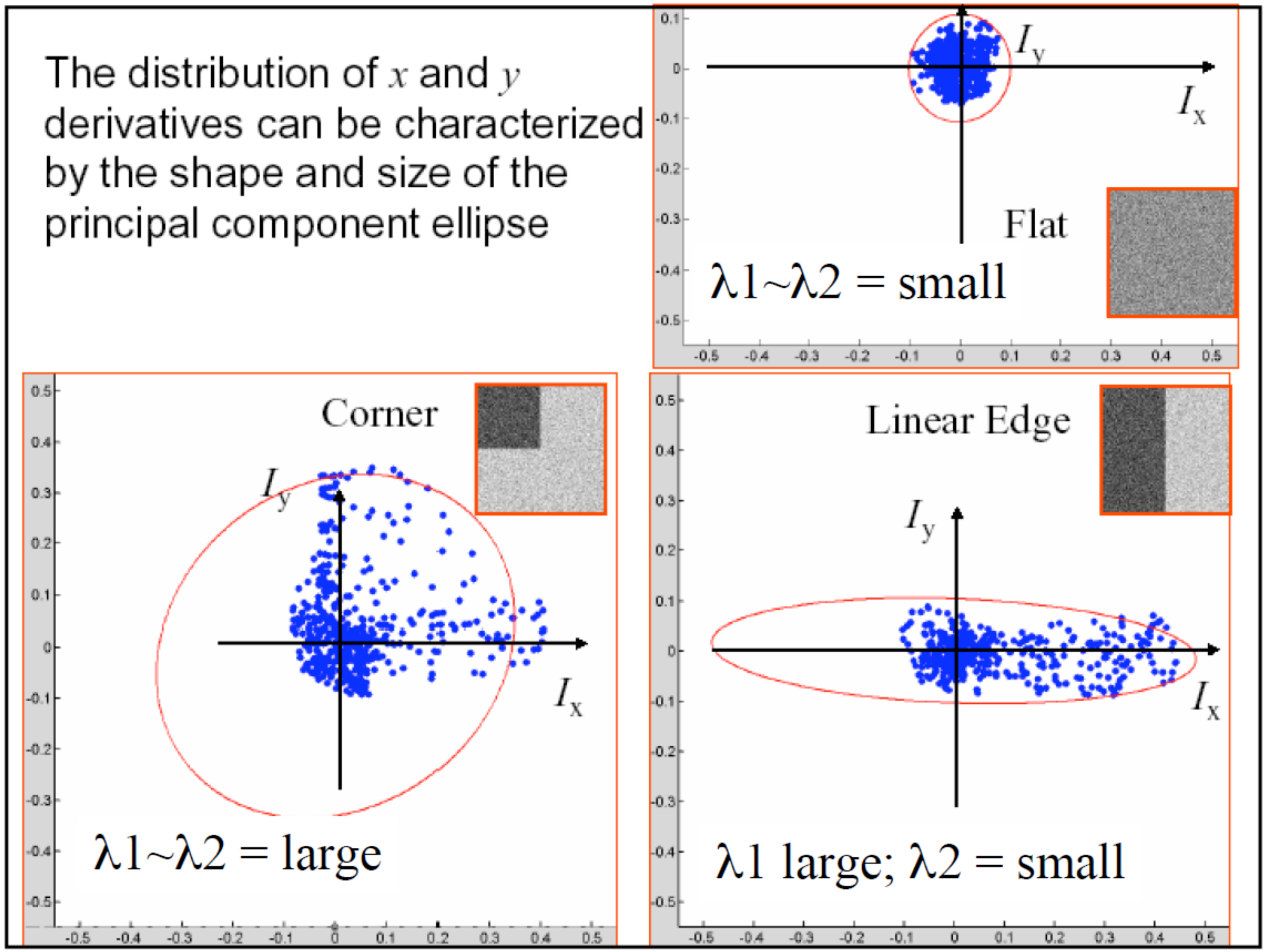
The distribution of the x and y derivatives is very different for all three types of patches





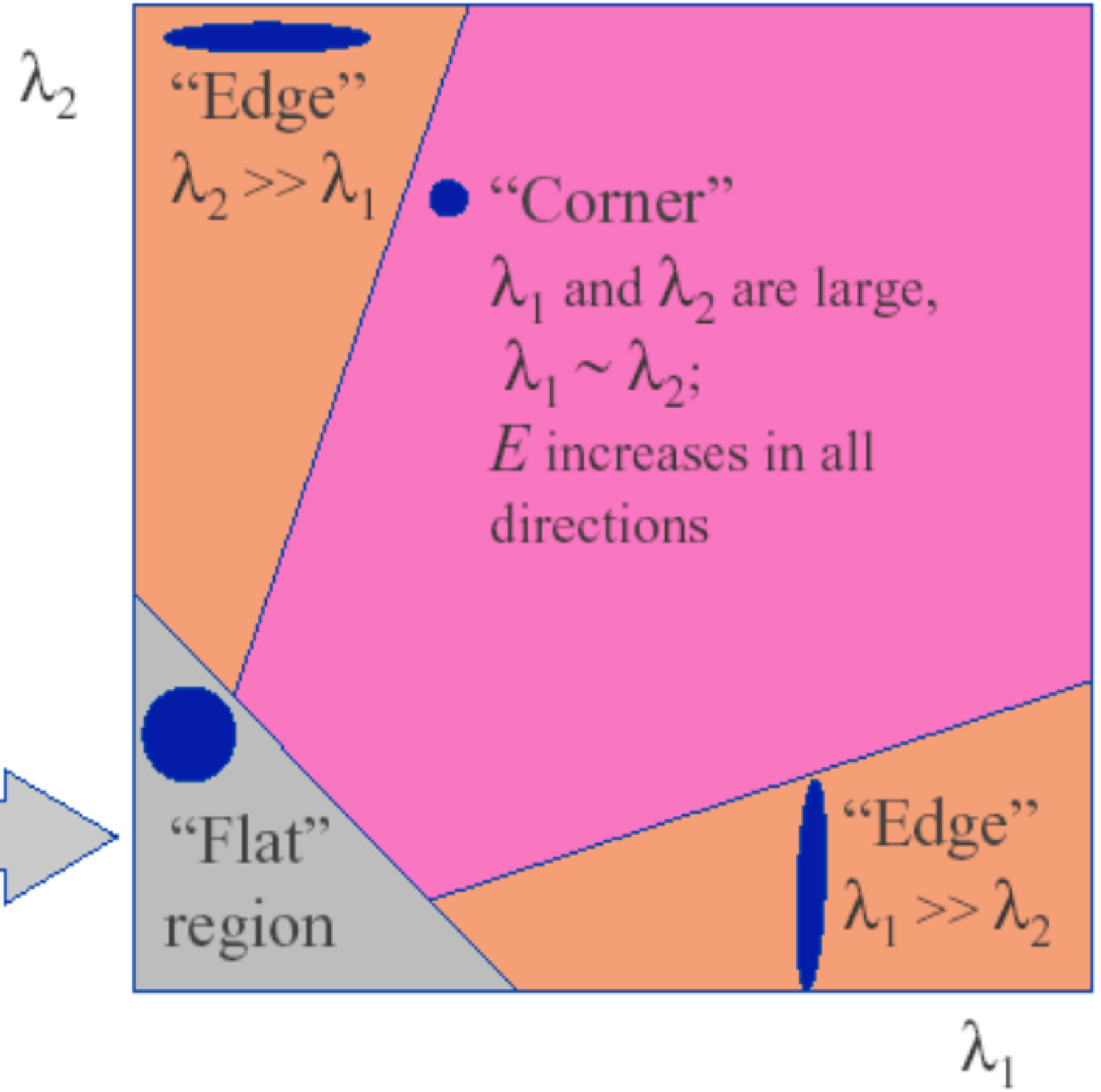


Fitting Ellipse to each Set of Points



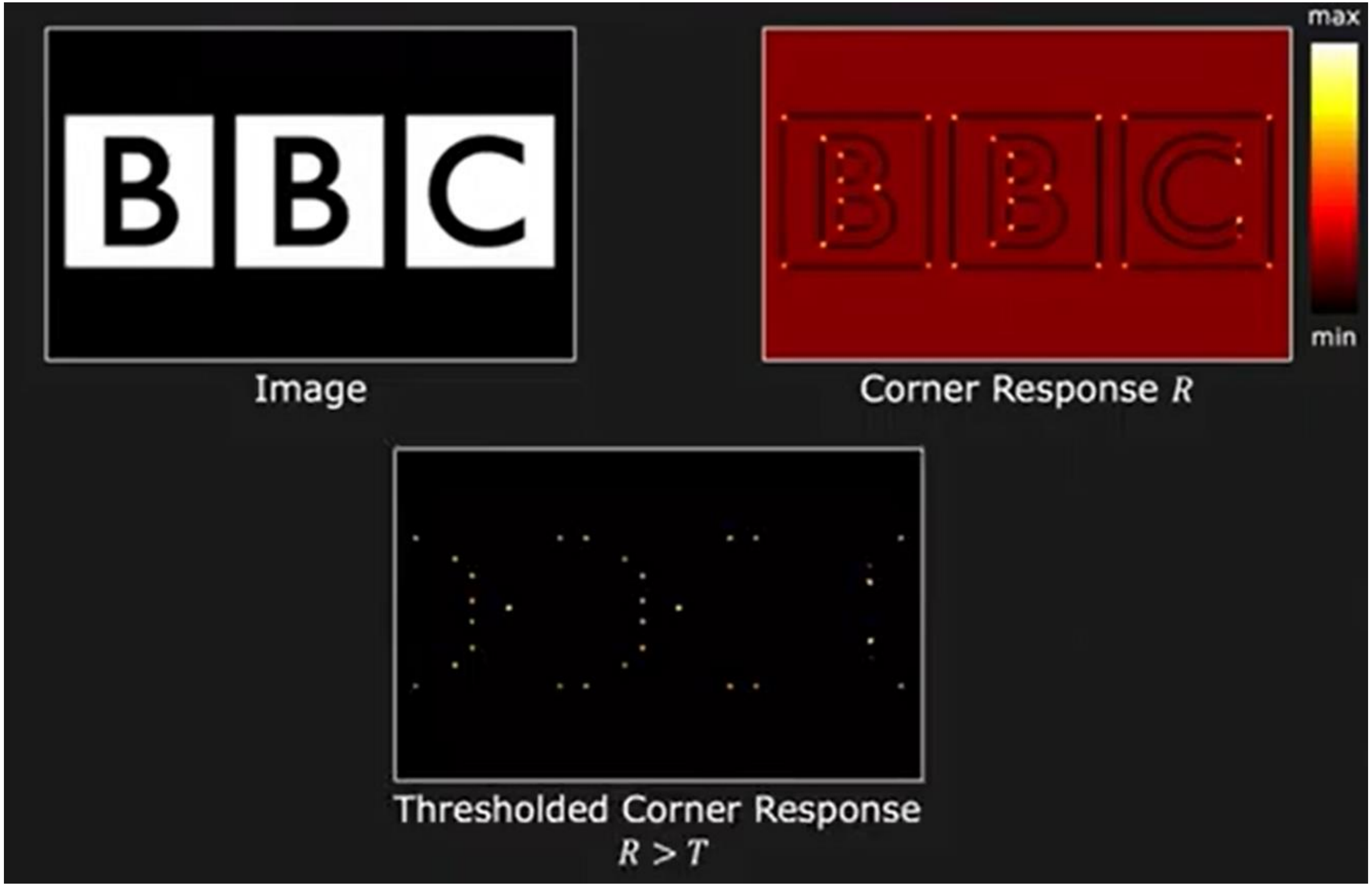
Classification via Eigen Values

Classification of image points using eigenvalues of M:

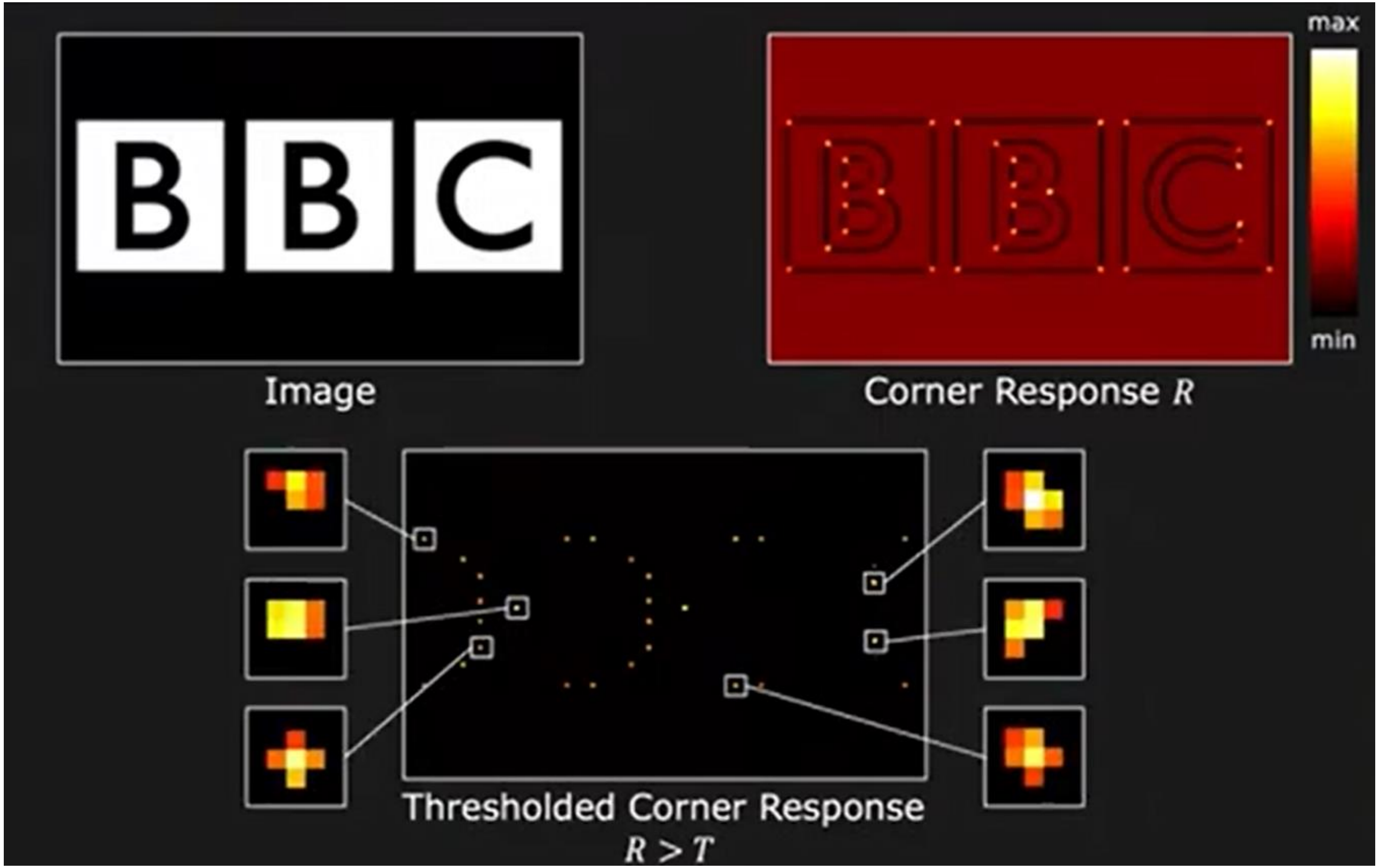


λ_1 and λ_2 are small;
E is almost constant in all directions

Harris Corner Detection Example



Harris Corner Detection Example



Non-Maximal Suppression

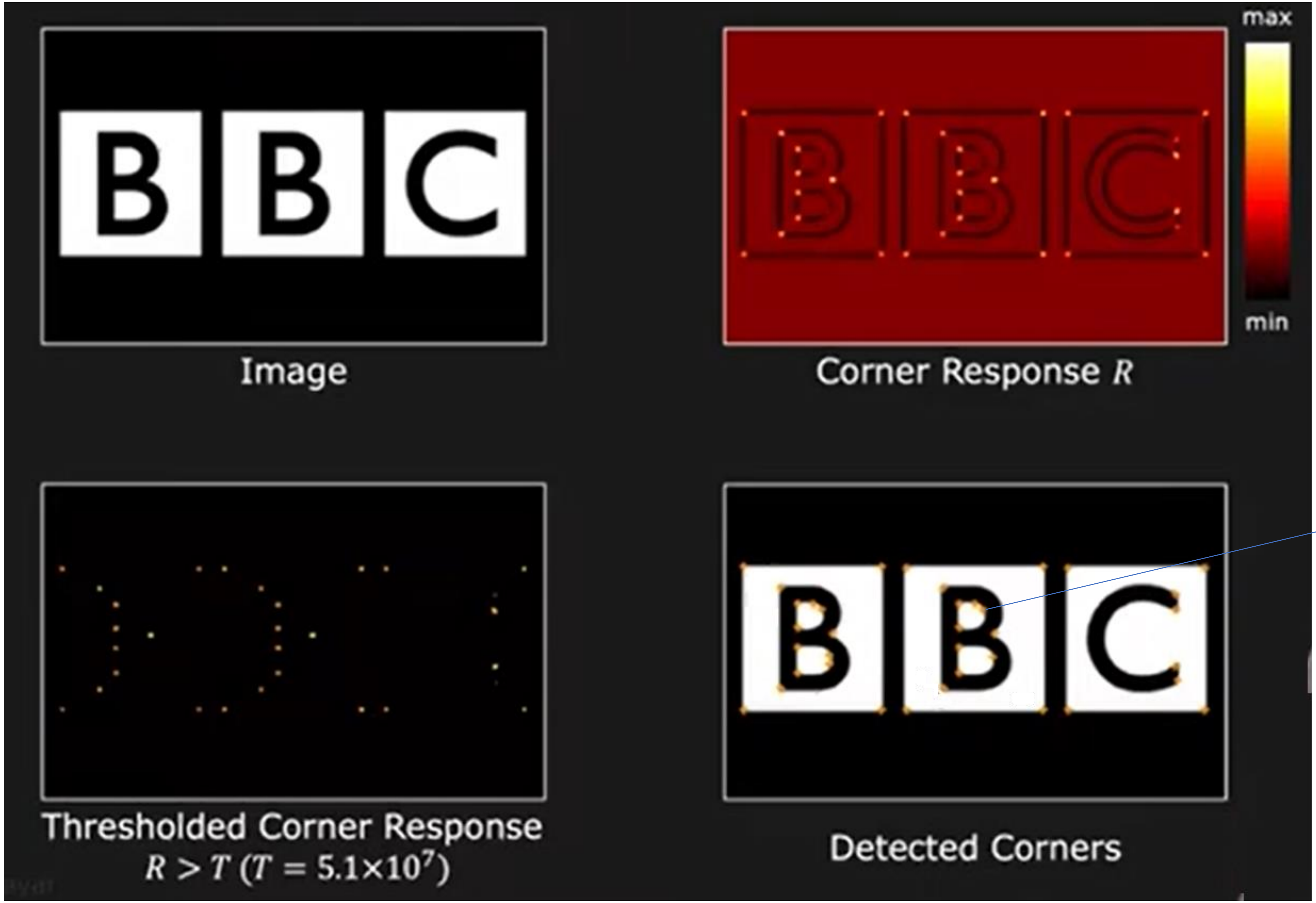
1. Slide a window of size k over the image.

2. At each position, if the pixel at the center is the maximum value within the window, label it as positive (retain it). Else label it as negative (suppress it).

Suppress | Suppress | Retain

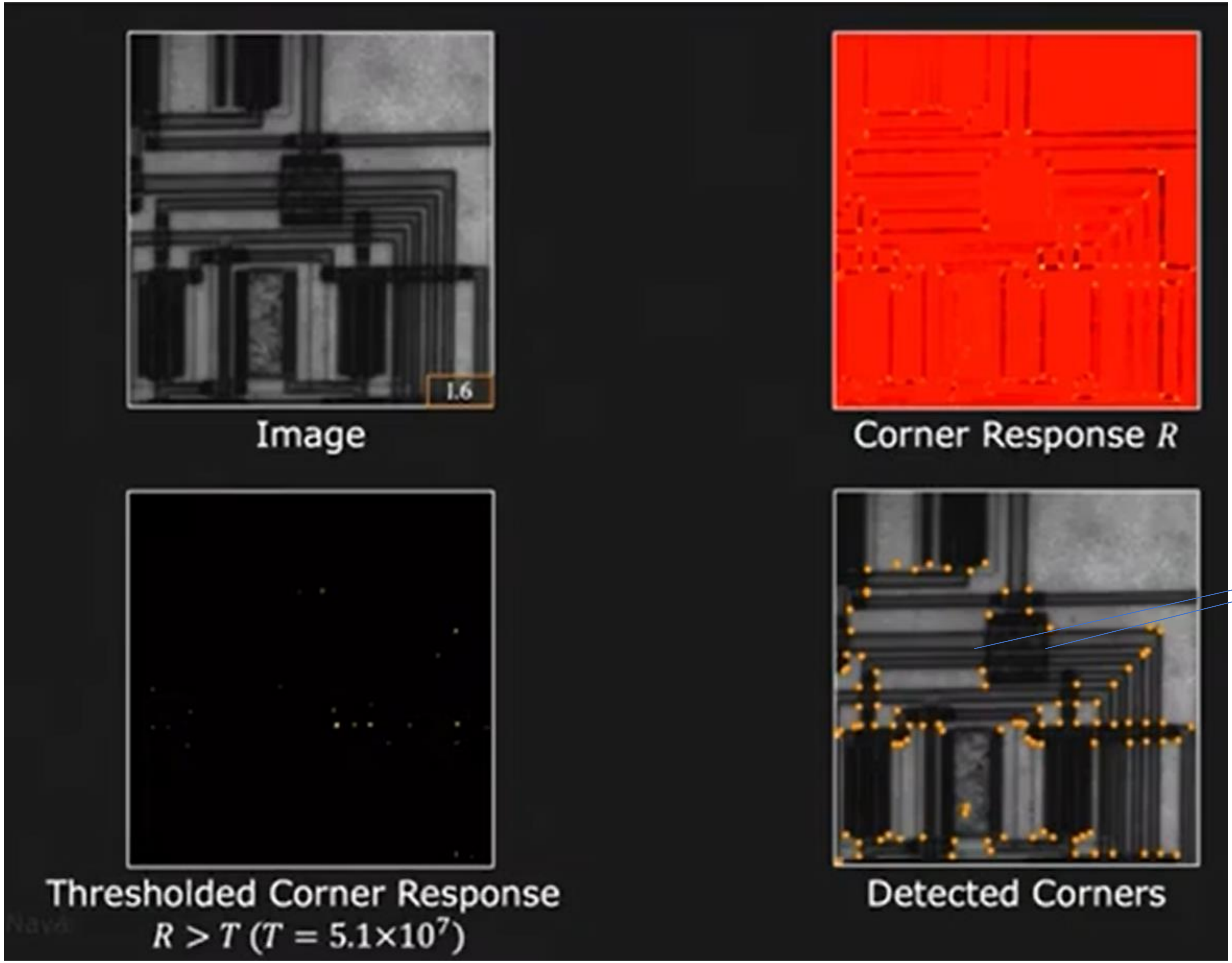
Used for finding Local Extrema (Maxima/Minima)

Harris Corner Detection Example



Curve Point
Detected as a corner

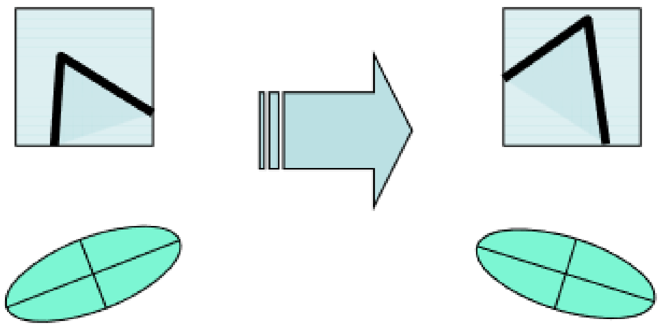
Harris Corner Detection Example



Missed corners

Harris Detector: Properties

- Rotation invariance?



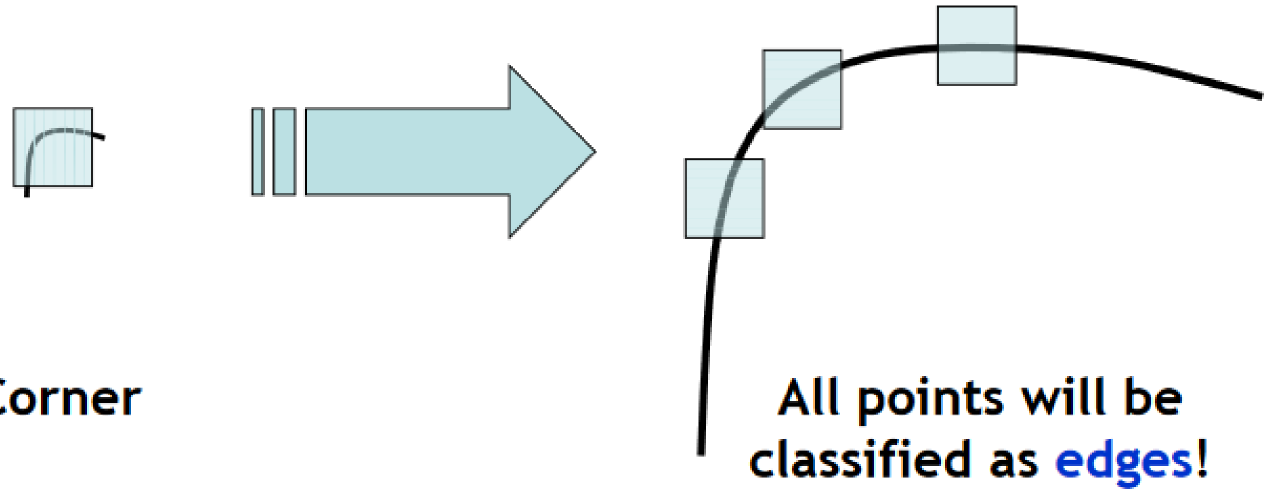
Ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner response R **is invariant to image rotation**

Slide credit: Kristen Grauman

Harris Detector: Properties

- Rotation invariance
- Scale invariance?



***Not** invariant to image scale!*

Slide credit: Kristen Grauman

Harris Corner Detector

Harris corner detector consists of the following steps.

1. Convert the original image into a grayscale image I. The pixel values of I are computed as a weighted sum of the corresponding R, G, B values:

$$I = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$

2. Compute the derivatives I_x and I_y by convolving the image I with the Sobel operator:

$$I_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * I \quad \text{and} \quad I_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I$$

3. Compute the products of the derivatives $I_x I_x$, $I_x I_y$, $I_y I_y$.
4. Convolve the images $I_x I_x$, $I_x I_y$, $I_y I_y$ with a Gaussian filter or a mean filter. Define the structure tensor for each pixel as expressed in eq. (5).
5. Compute the response function for each pixel:

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

k is a constant to chose in the range $[0.04, 0.06]$. Since M is a symmetric matrix, $\det(M) = \lambda_1 \lambda_2$ and $\mathrm{tr}(M) = \lambda_1 + \lambda_2$ where λ_1 and λ_2 are the eigenvalues of M. Hence, we can express the corner response as a function of the eigenvalues of the structure tensor:

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

So the eigenvalues determine whether a region is an edge, a corner or flat:

- if λ_1 and λ_2 are small, then $|R|$ is small and the region is flat;
- if $\lambda_1 >> \lambda_2$ or viceversa, then $R < 0$ and the region is an edge;
- if $\lambda_1 \approx \lambda_2$ and both eigenvalues are large, then R is large and the region is a corner.

6. Set a threshold T on the value of R and find pixels with responses above this threshold. Finally, compute the non-max suppression in order to pick up the optimal corners.

Hessian Detector

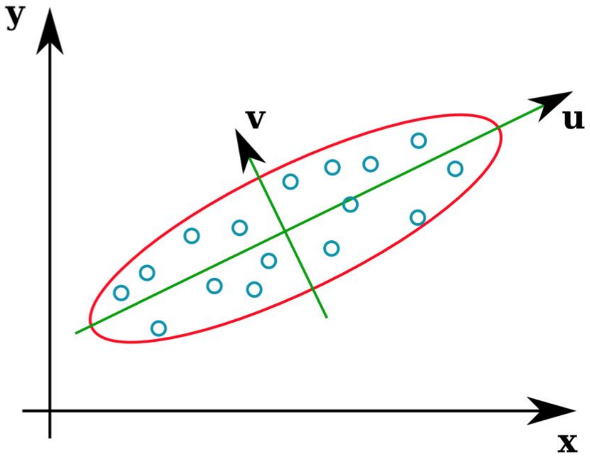
❖ The basic ideas of detecting corners remain the same as the Harris detector, the Hessian detector makes use of the Hessian matrix and determinant.

$$Hessian(I) = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

Note: these are 2nd derivatives!

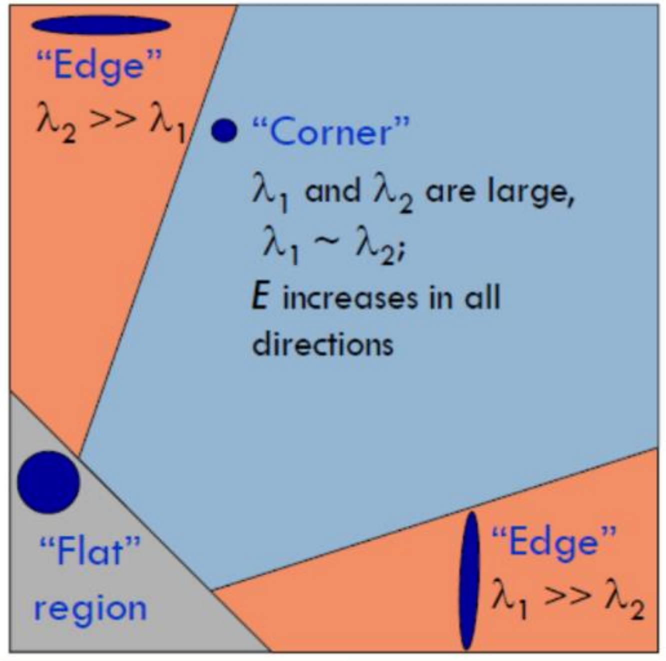
$$\det(Hessian(I)) = I_{xx} I_{yy} - I_{xy}^2$$

Calculate eigen values using following equation: $|\lambda I - H| = 0$



Two eigen vectors *u* and *v* *with* a large variation in *u* direction and lower variation in *v* direction.

λ2



λ1

Intuition: Search for strong derivatives in two orthogonal directions

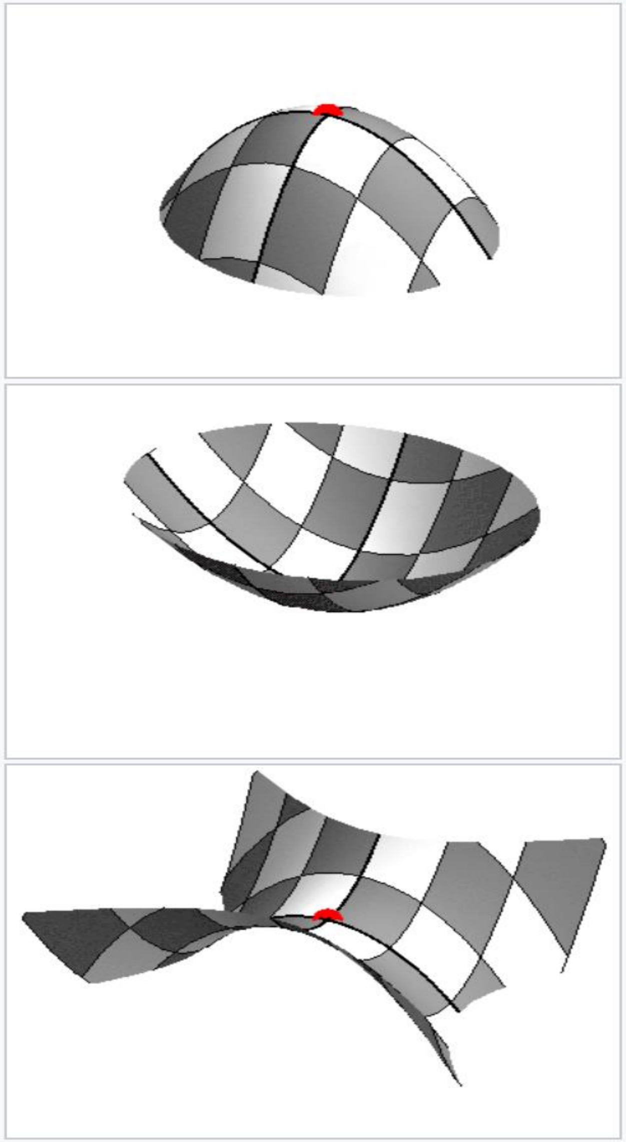
[Beaudet78]

Determinant of Hessian

$$\det(Hessian(I)) = I_{xx} I_{yy} - I_{xy}^2$$

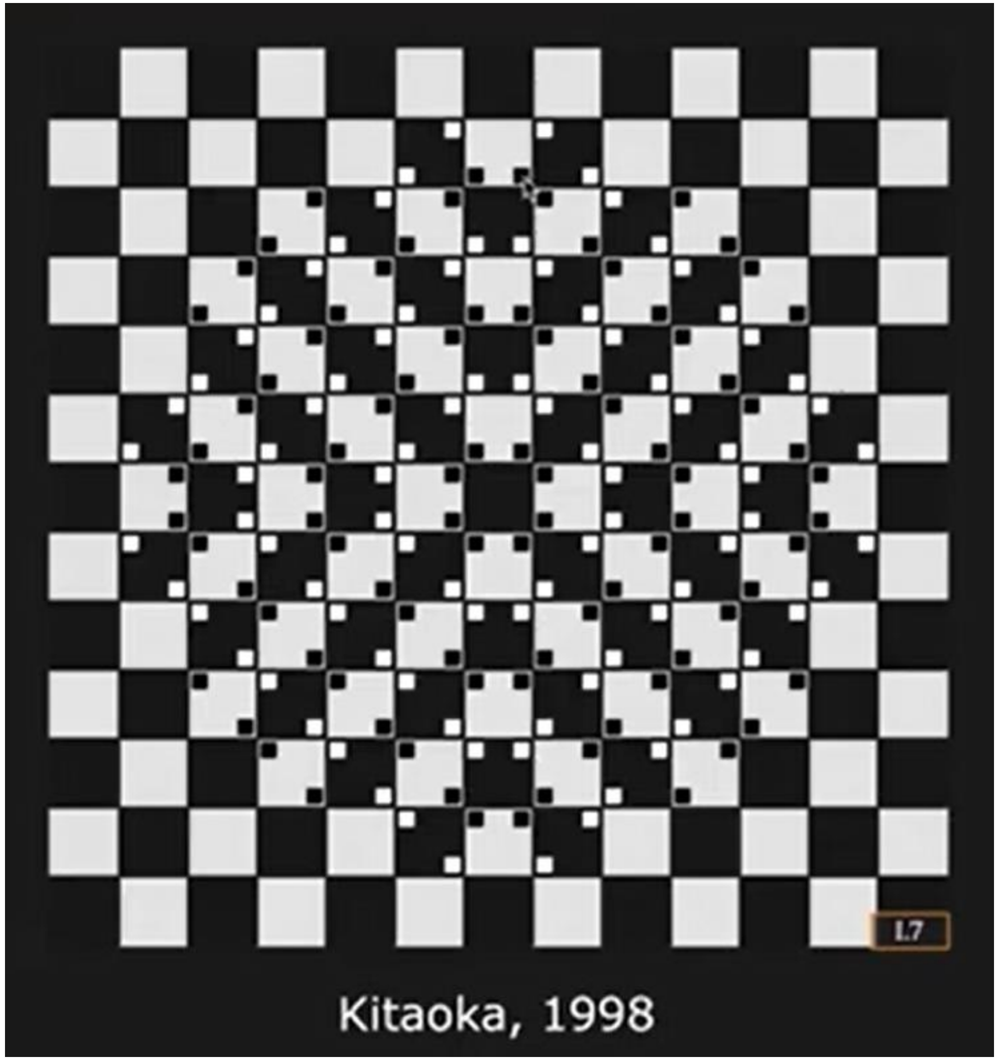
- “Determinant of the Hessian" summarizes the curvature at a point.
  - *If D*>0, the point looks like either a bowl upward or downward,
  - If *D*<0, the point looks like a saddle point.

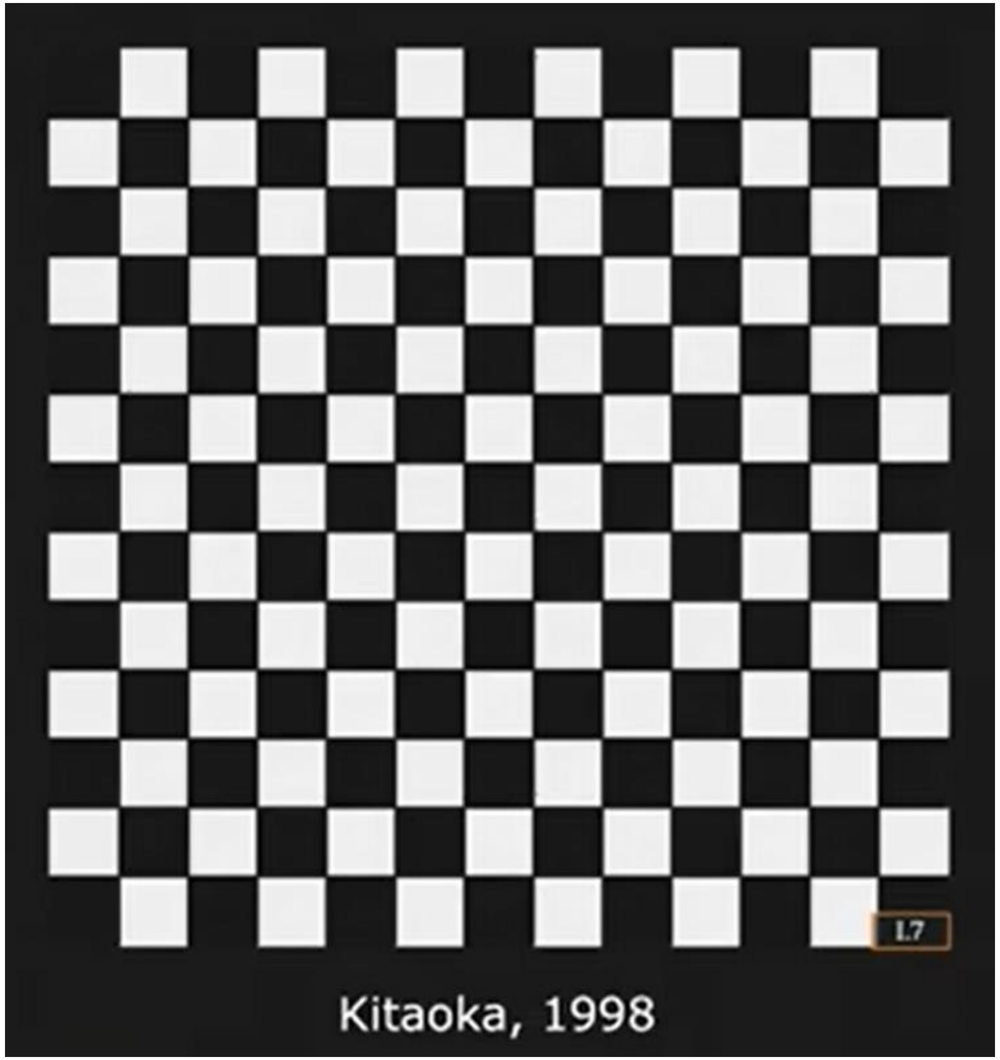
- As the edge looks more like a saddle than a bowl, this is one way that you can separate good, stable interest points from edges using the Hessian.



The Hessian approximates the function at a critical point with a second degree polynomial.

Corner Illusion: The Bulge





Biases in the Human Visual around corners

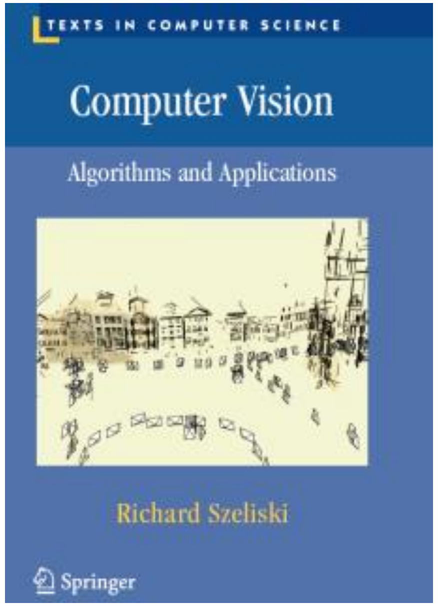
Text Books and References

❖ Text Books

- Richard Szeliski. "Computer vision: Algorithms and Applications. Springer Nature, Second Edition", 2022
- E. R. Davies, Computer Vision Principles, Algorithms, Applications, Learning, Elsevier,5th Edition, 2017

❖ References

- Rafael C. Gonzales, Richard E. Woods, "Digital Image Processing", Fourth Edition, Pearson Education, 2018.
- Richard Szeliski ,"Computer Vision: Algorithms and Applications" , Springer 2015
- Richard Szeliski. "Computer Vision: Algorithms and Applications 2nd Edition – final draft", 2021
- Intro to Digital Image Processing
- Digital Image Processing 2nd edition



PDF online

❖ Credits: Slides and Video content borrowed from:

- First Principles of Computer Vision
- https://robotacademy.net.au/
- MIT 6.S094: Computer Vision
- MIT Introduction to Deep Learning | 6.S191
- Digital Image Processing 2nd edition
- CSE/EE486 Computer Vision I (Penn State University)
- http://vision.stanford.edu/teaching/cs231a_autumn1112/lecture/lecture11_detectors_descriptors_cs231a.pdf
- https://www.cse.psu.edu/~rtc12/CSE486/lecture06.pdf
- https://medium.com/jun94-devpblog/cv-10-local-feature-descriptors-harris-and-hessian-corner-detector-7d524888abf