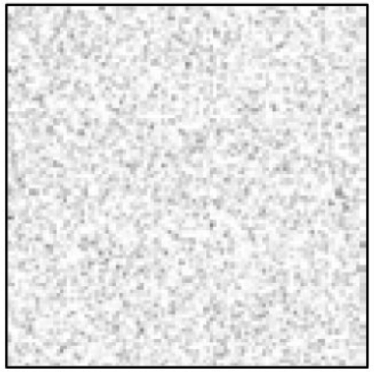
Corner Detection

Introduction to Computer Vision
Spring 2026, Lecture 4

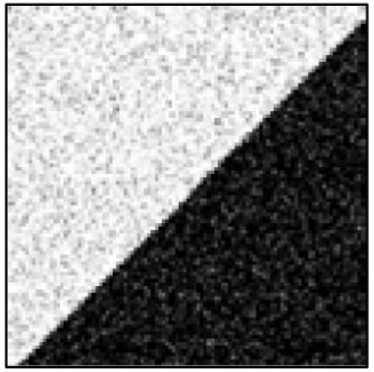
https://liaoux.github.io/uir-cv-2026/

Corners

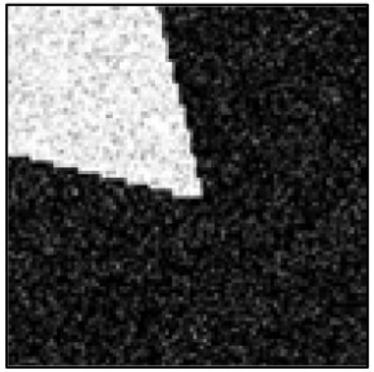
Corner: Point where Two Edges Meet. i.e., Rapid Changes of Image Intensity in Two Directions within a Small Region.



"Flat" Region



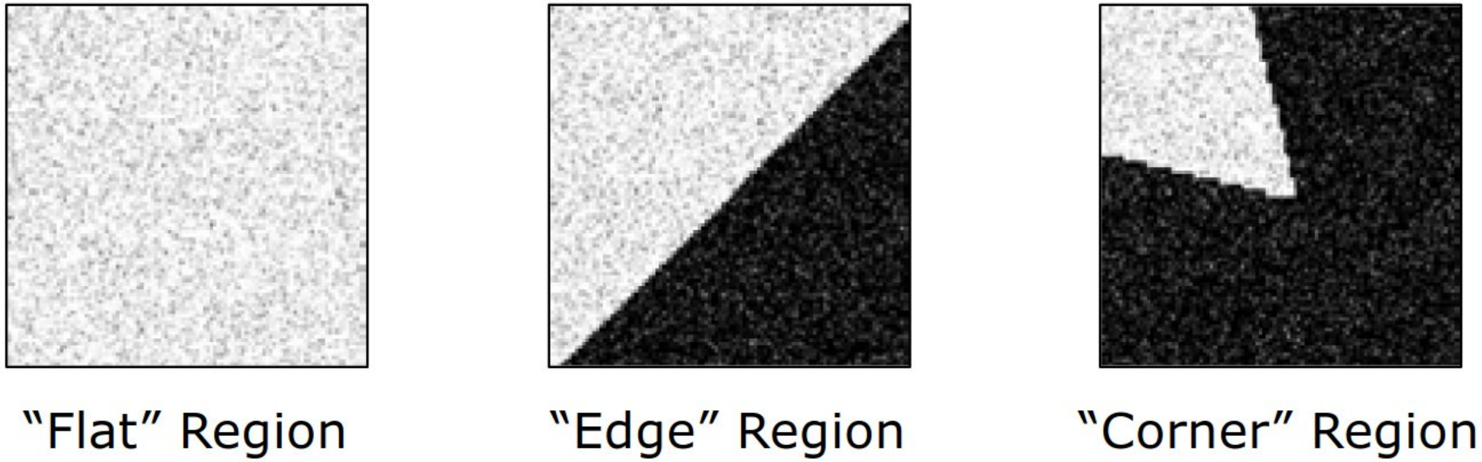
"Edge" Region



"Corner" Region

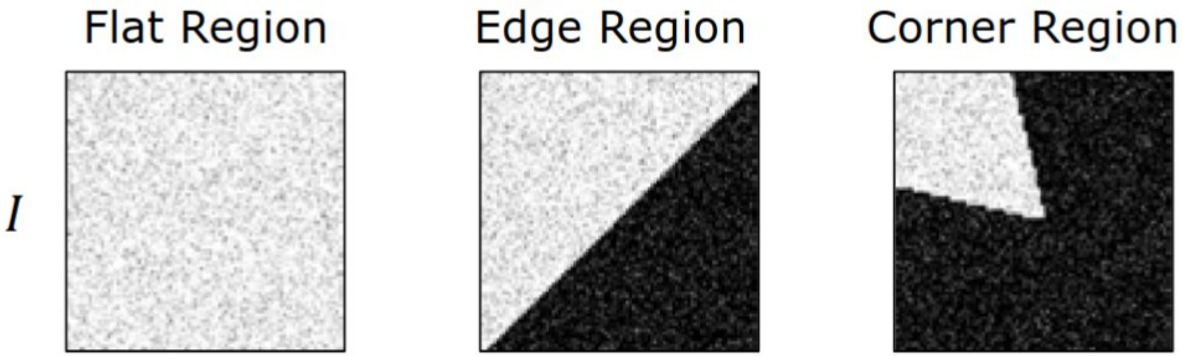
Corners

Corner: Point where Two Edges Meet. i.e., Rapid Changes of Image Intensity in Two Directions within a Small Region.



What operator we used previously to detect the edges?

Image Gradients



Transition (left → right)	**The derivative sign**	**Displayed as**
Black → White	Large positive	Bright white
White → Black	Large negative	Dark black
No change	≈ 0	Middle gray

Image Gradients

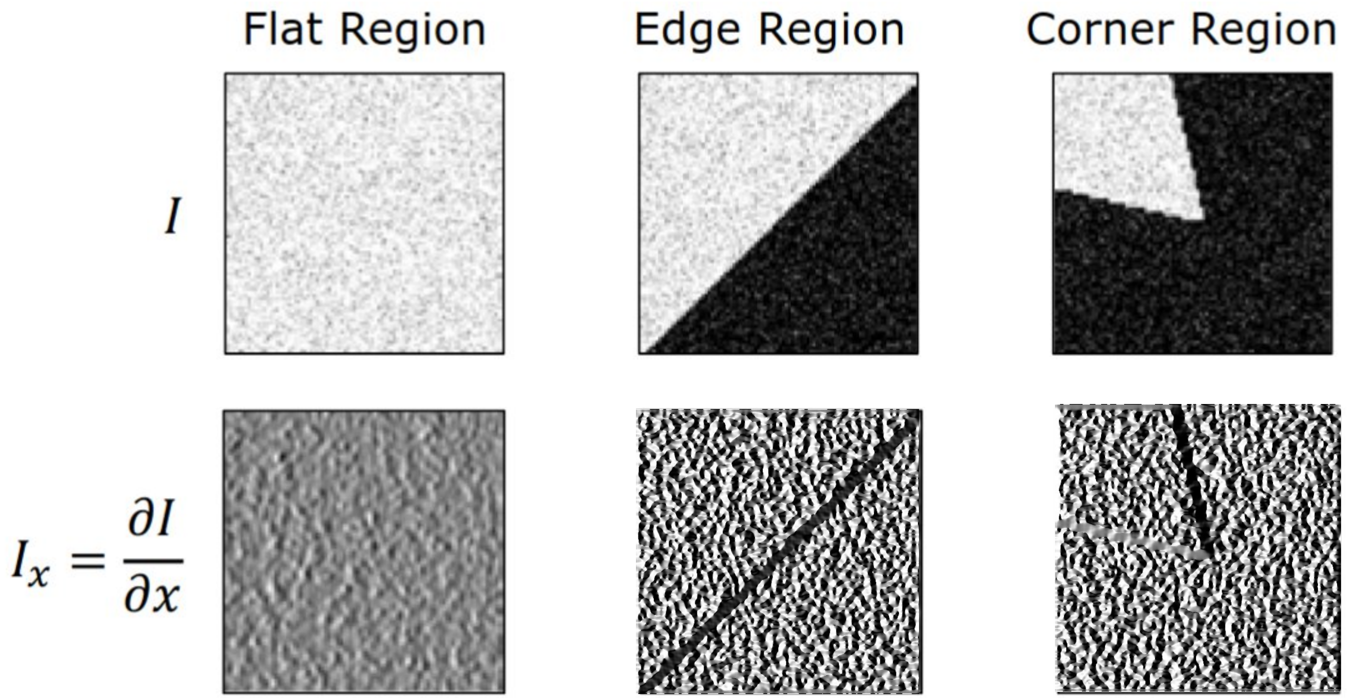
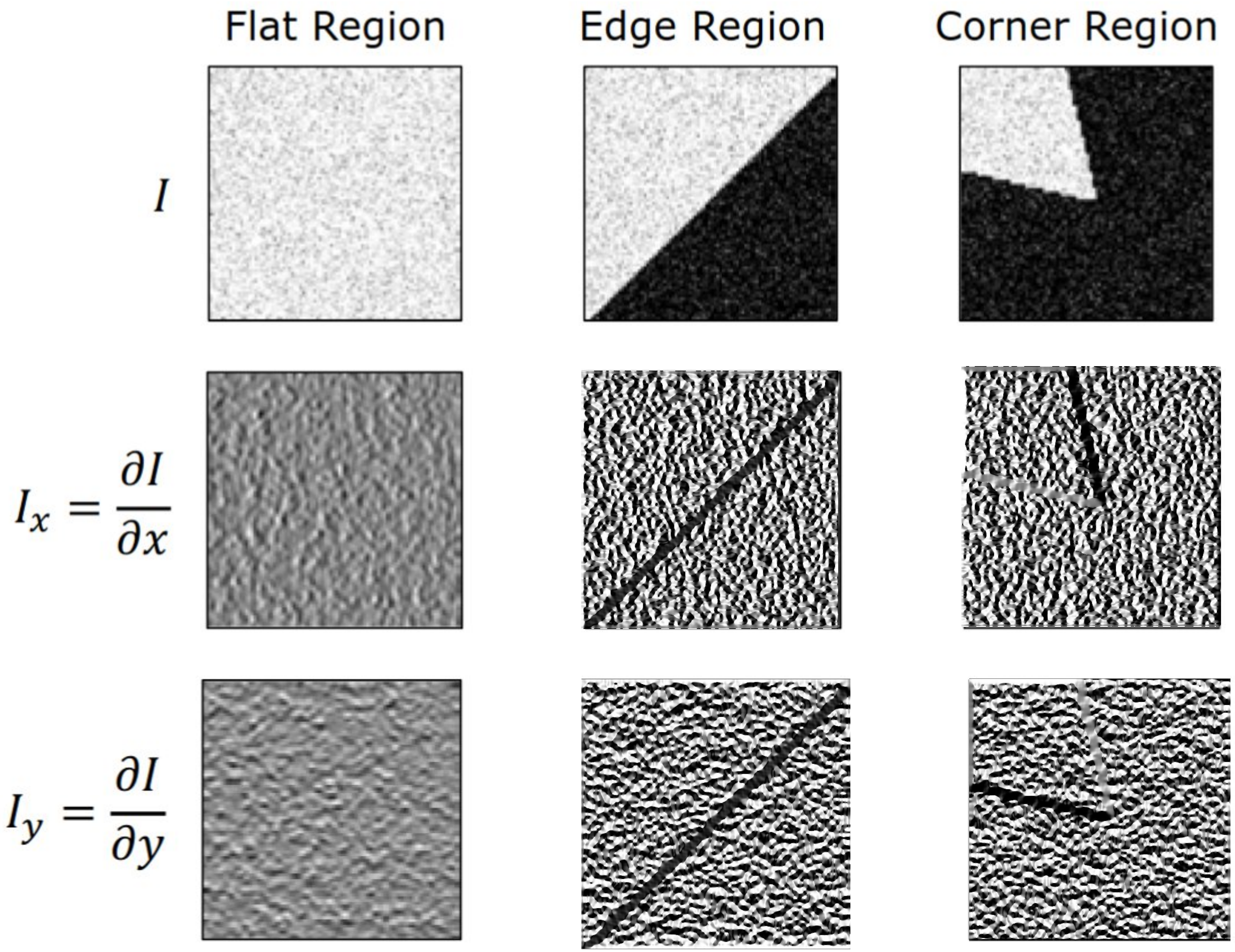
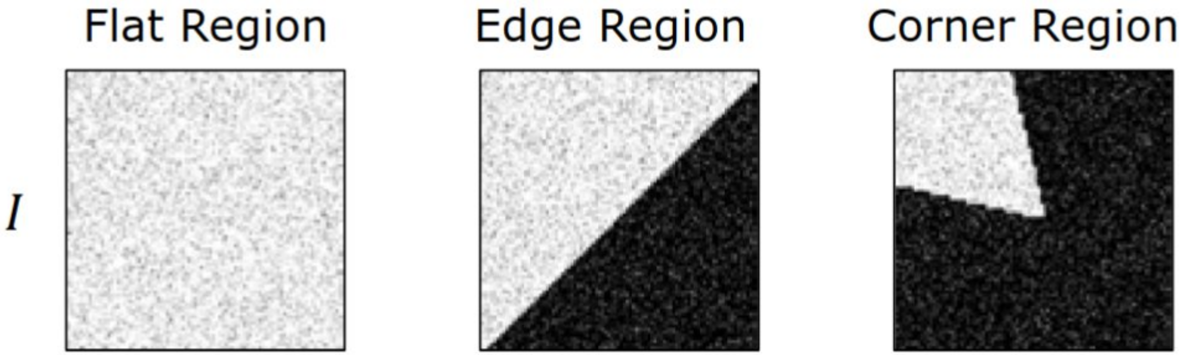


Image Gradients

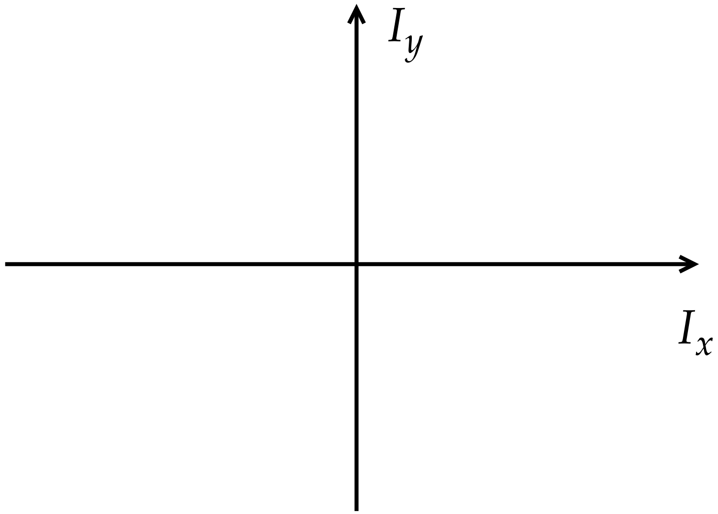


Distribution of Image Gradients

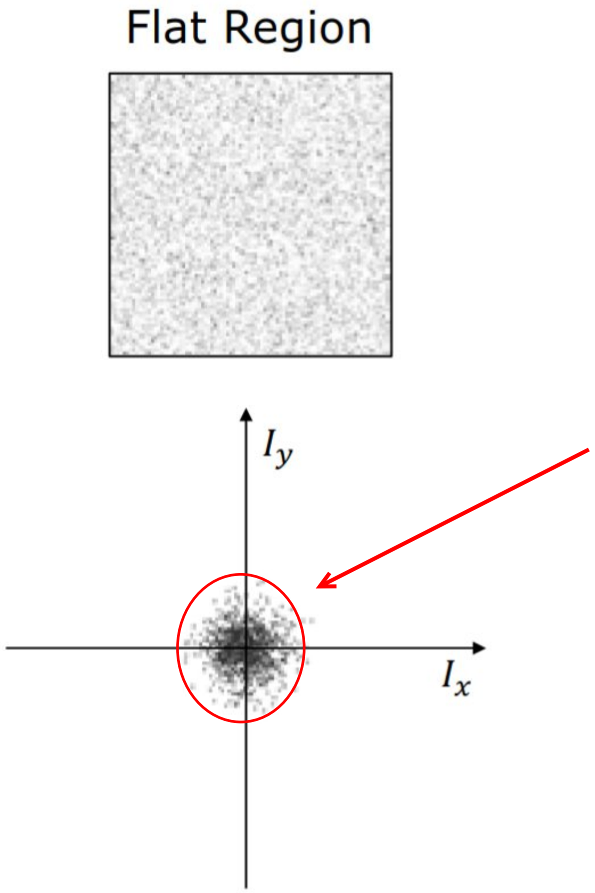


- Let us now construct a 2D space whose dimensions correspond to the x and y image gradients, I_x and I_y.

- For each of the three image regions, we will plot the gradient values at each pixel in this space

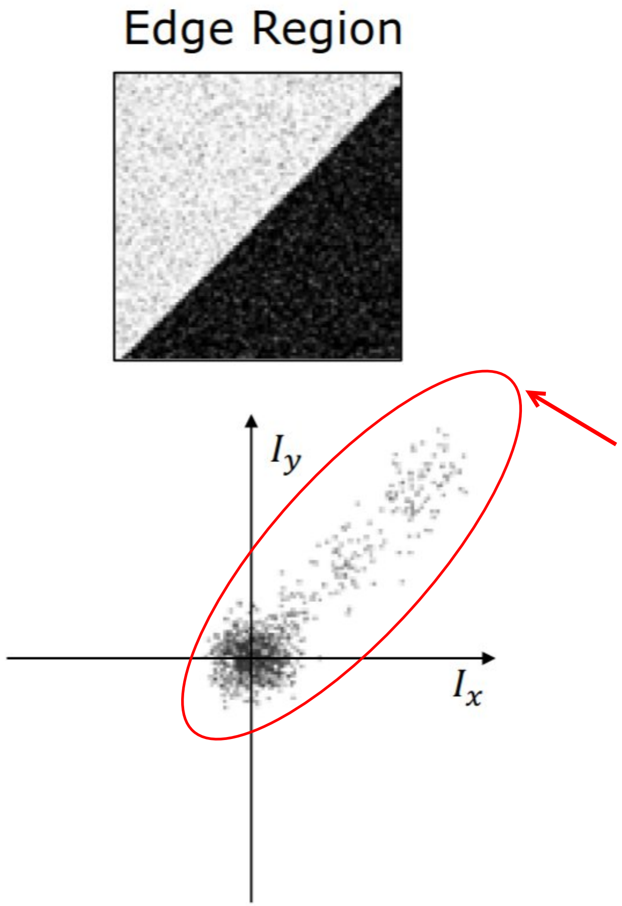


Distribution of Image Gradients



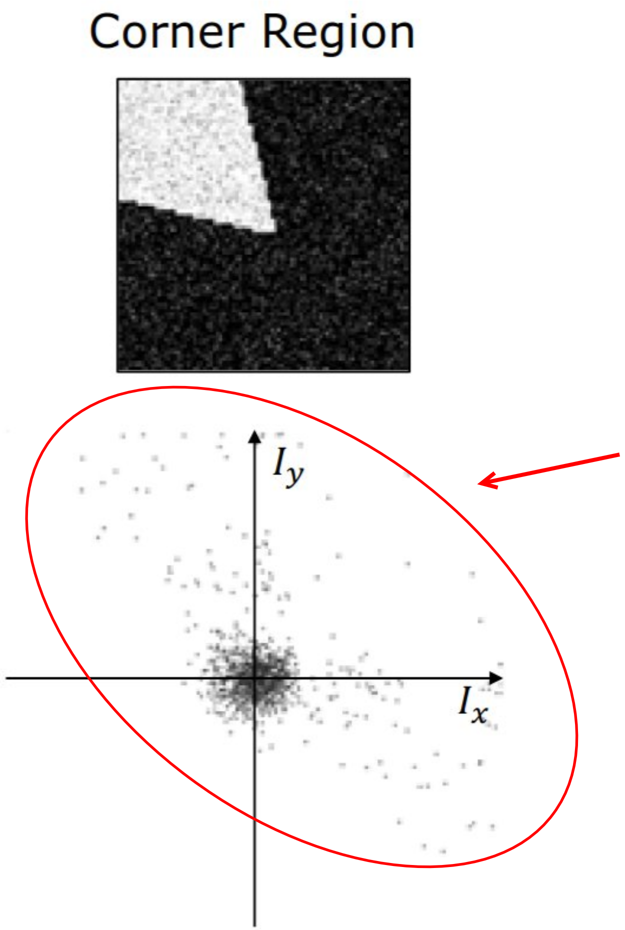
For the **flat region**, we get a very **compact cluster** close to the **origin**. In fact, if the region did not have any **noise**, the cluster would shrink to a **single point**, namely, the **origin**.

Distribution of Image Gradients



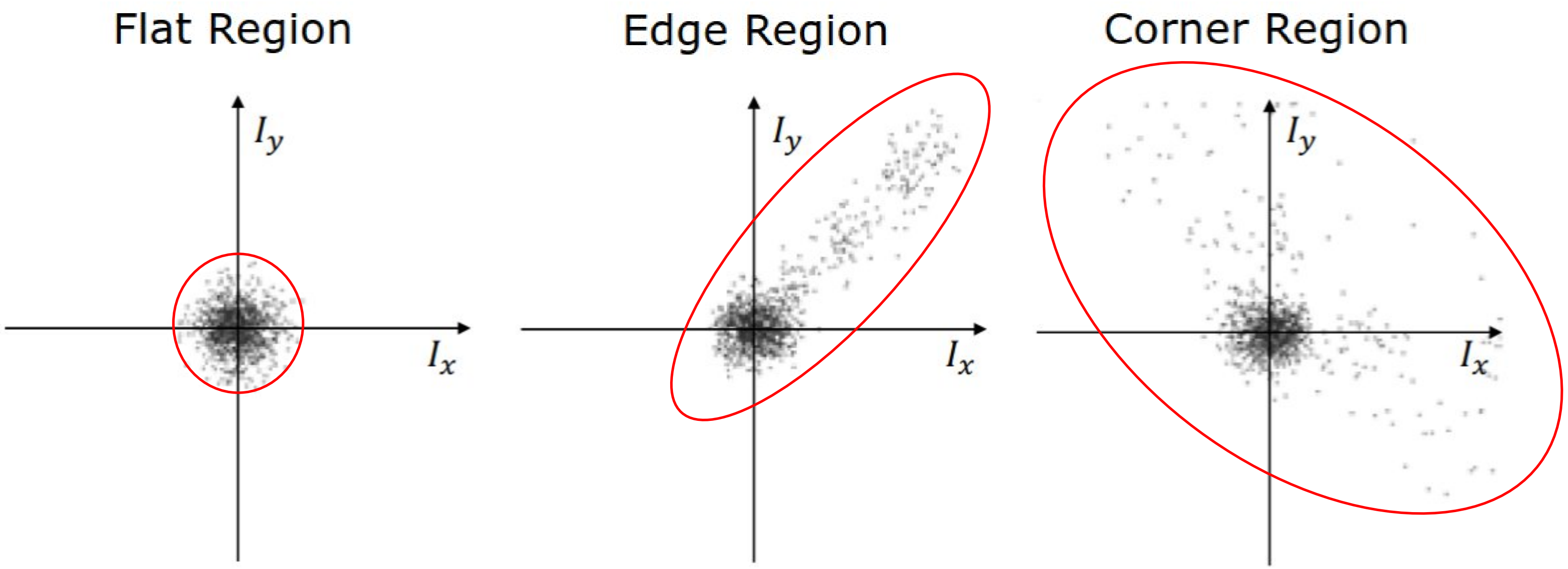
For the **edge region**, we once again get a **compact cluster** because there are significant **flat regions** on both sides of the edge, but we also get a **thin and long cluster**. This cluster corresponds to the **large gradient values** obtained at or near the **edge**.

Distribution of Image Gradients



In the case of the **corner**, we again get a **compact cluster** because of the **flat regions**, but we get **two additional clusters**, one for each of the **two edges** that make up the corner.

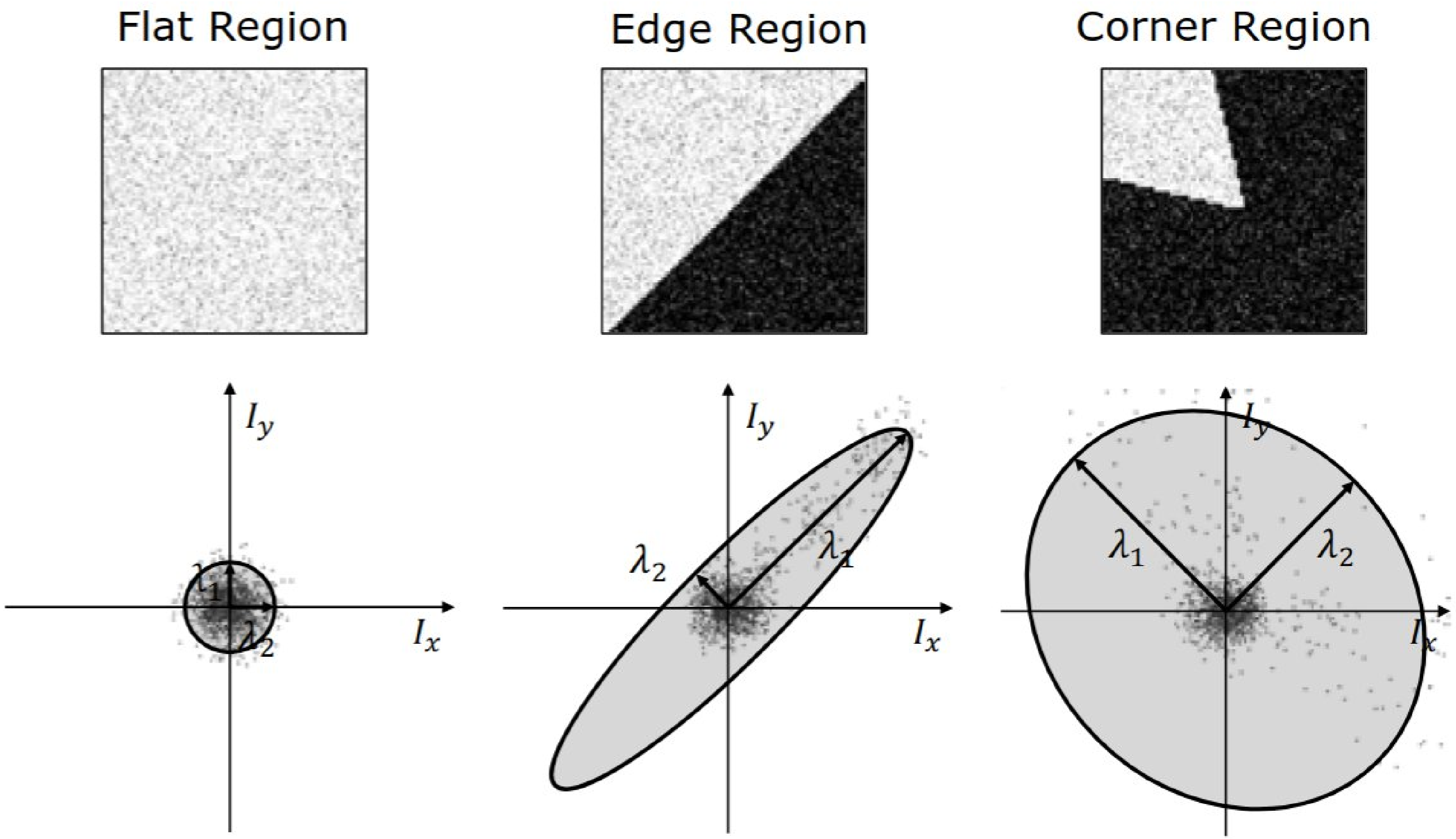
Distribution of Image Gradients



The Goal:

Quantify the **structure** of the **distribution of points** in this **gradient space** with a **simple model** that can be described using a **small number of parameters.** Then, use these **parameters** to **classify** each **local image region** as being **flat**, an **edge**, or a **corner**.

Fitting Elliptical Disk to Distribution



$\boldsymbol{\lambda_1}$**:** Length of Semi-Major Axis

$\boldsymbol{\lambda_2}$**:** Length of Semi-Minor Axis

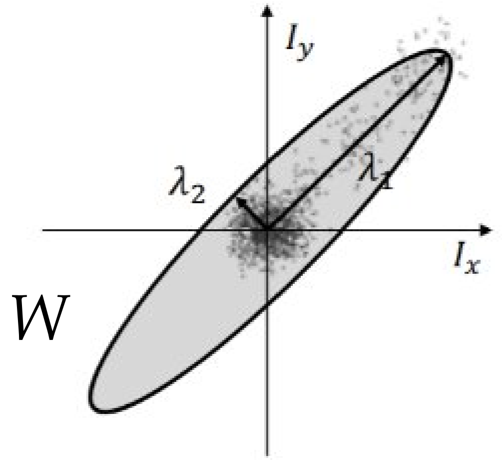
Fitting Elliptical Disk to Distribution

To fit this ellipse mathematically, we compute three quantities over the window W:

$$a = \sum_{i \in W} \left({I_x}_i^{\,2} \right) \qquad b = 2\sum_{i \in W} \left({I_x}_i {I_y}_i \right)$$

$$c = \sum_{i \in W} \left({I_y}_i^{\,2} \right)$$

W: Window centered at pixel



Think of them as:

a. how much the distribution **spreads horizontally** (in I_x)
b. whether the distribution is **tilted diagonally** (correlation between I_x and I_y)
c. how much the distribution **spreads vertically** (in I_y)

Together, they form the **second moment matrix** (structure tensor) ⇨ $M = \begin{bmatrix} a & b/2 \\ b/2 & c \end{bmatrix}$

Fitting Elliptical Disk to Distribution

The Ellipse Axes = Eigenvalues of M

For any 2×2 symmetric matrix, the eigenvalues have a **closed-form solution** ($\det(M - \lambda I) = 0$):

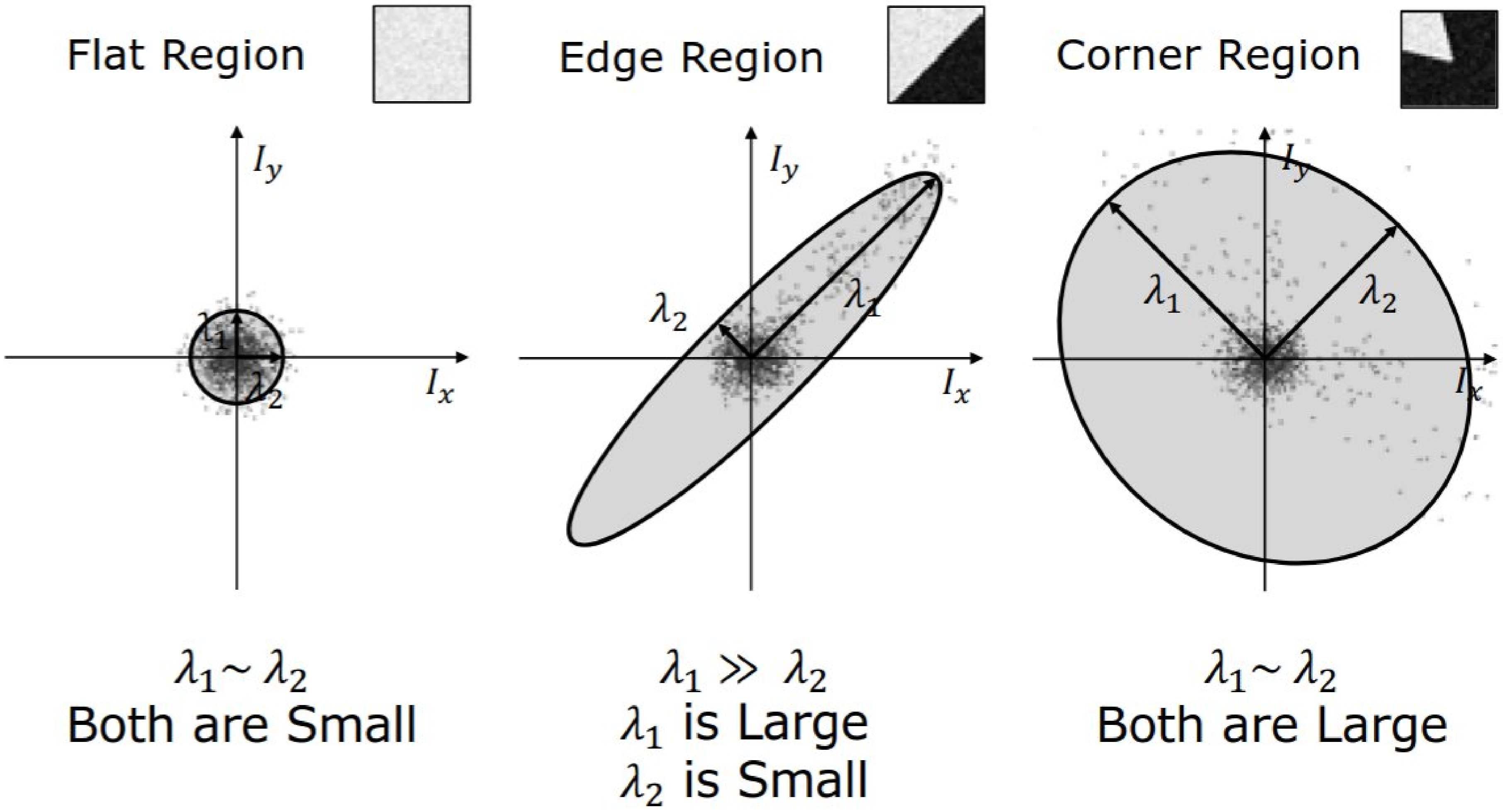
$$\lambda = \frac{(a + c) \pm \sqrt{(a-c)^2 + 4d^2}}{2}$$

Here $d = {}^{b}/_{2}$, so $4d^2 = b^2$. Substituting directly gives the Ellipse Axes Lengths::

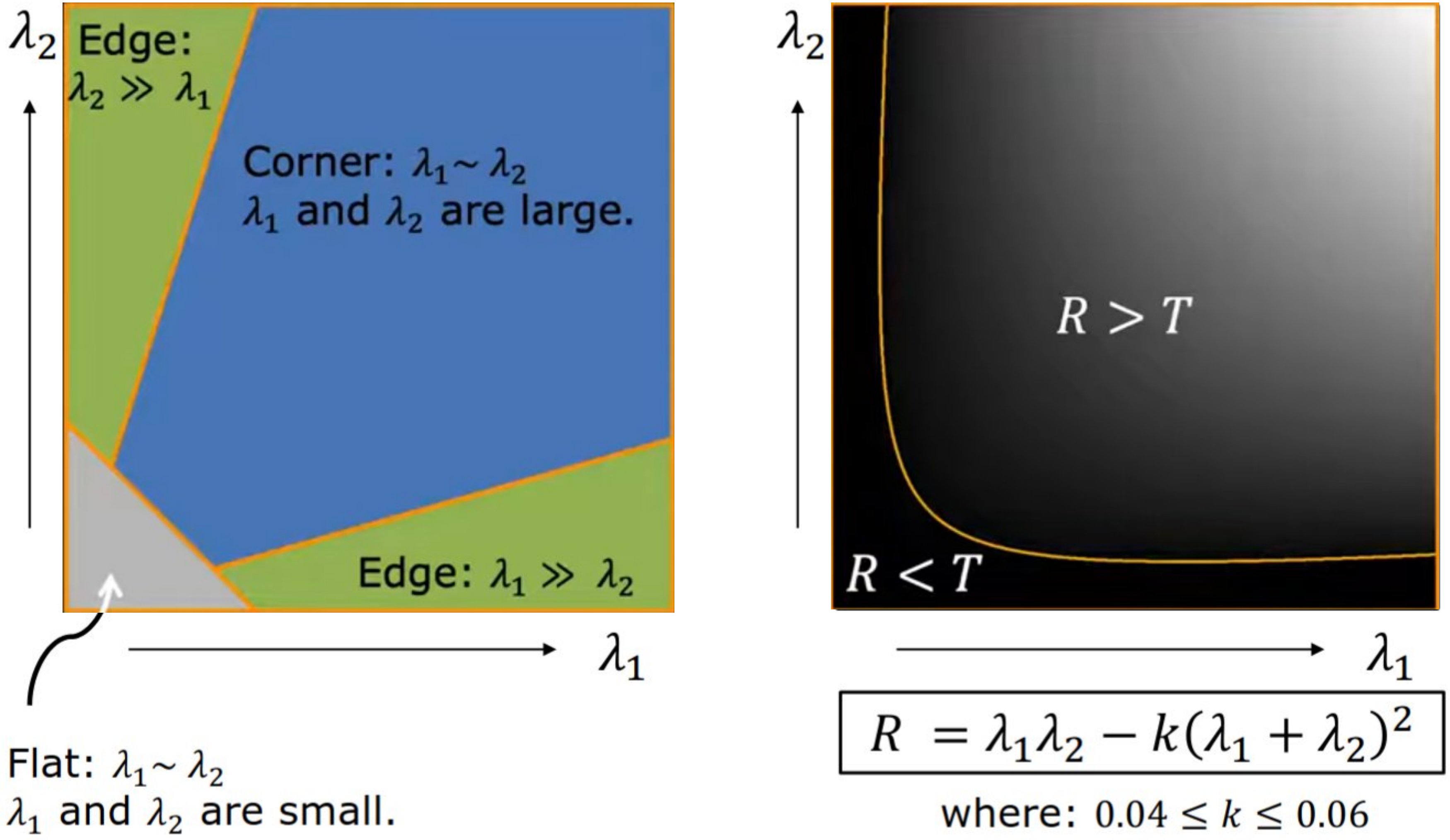
$$\lambda_1 = E_{max} = \frac{1}{2}\left[a + c + \sqrt{b^2 + (a - c)^2}\right]$$

$$\lambda_2 = E_{min} = \frac{1}{2}\left[a + c - \sqrt{b^2 + (a - c)^2}\right]$$

Interpretation of λ_1 and λ_2



Harris Corner Response Function



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

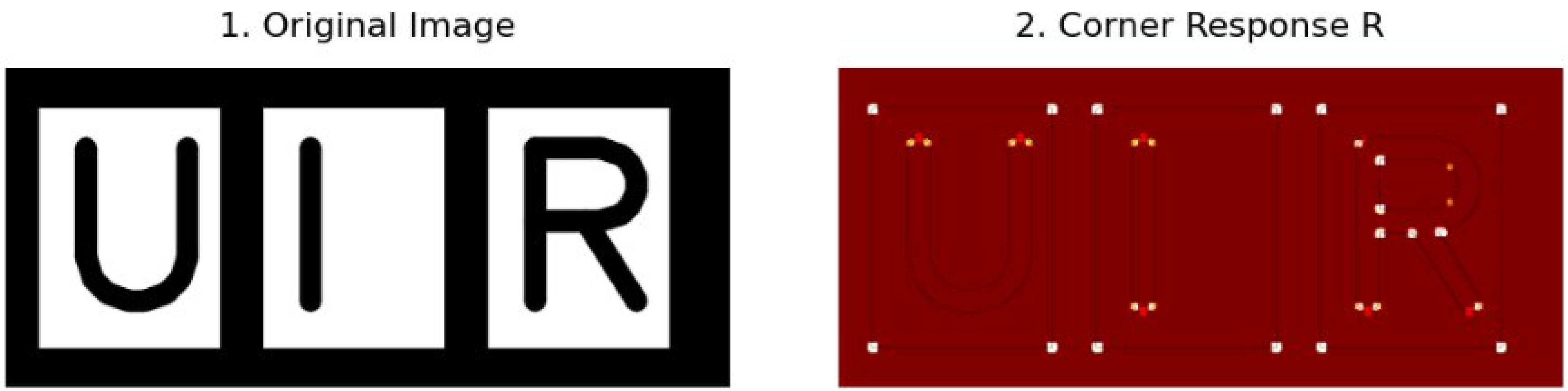
where: $0.04 \leq k \leq 0.06$
(Designed Empirically)

Harris 1988

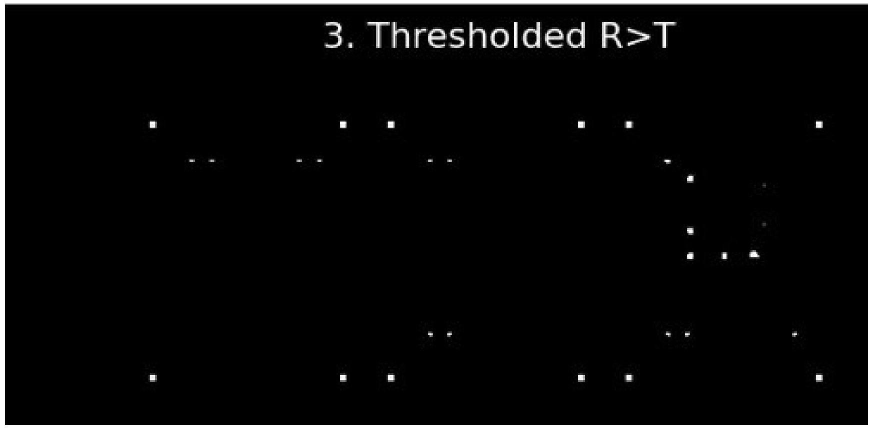
Harris Corner Detection Algorithm

1. **Compute Image Gradients**
   Horizontal (I_x) and vertical (I_y) changes in intensity
2. **Second Moments (for every pixel window)**
   $a = \Sigma(I_x^2) \rightarrow$ x-direction spread
   $b = 2\Sigma(I_x \cdot I_y) \rightarrow$ diagonal tilt
   $c = \Sigma(I_y^2) \rightarrow$ y-direction spread
3. **Eigenvalues**
   λ_1, λ_2 = eigenvalues of matrix M = [[a, b/2], [b/2, c]]
4. **Corner Response**
   $R = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$, where k = sensitivity parameter
5. **Classification**
   $R \gg 0 \rightarrow$ CORNER, $R < 0 \rightarrow$ EDGE, $R \approx 0 \rightarrow$ FLAT
6. **Threshold**
   $T = \alpha \times \max(R)$ across entire image
   Keep pixels where $R > T$

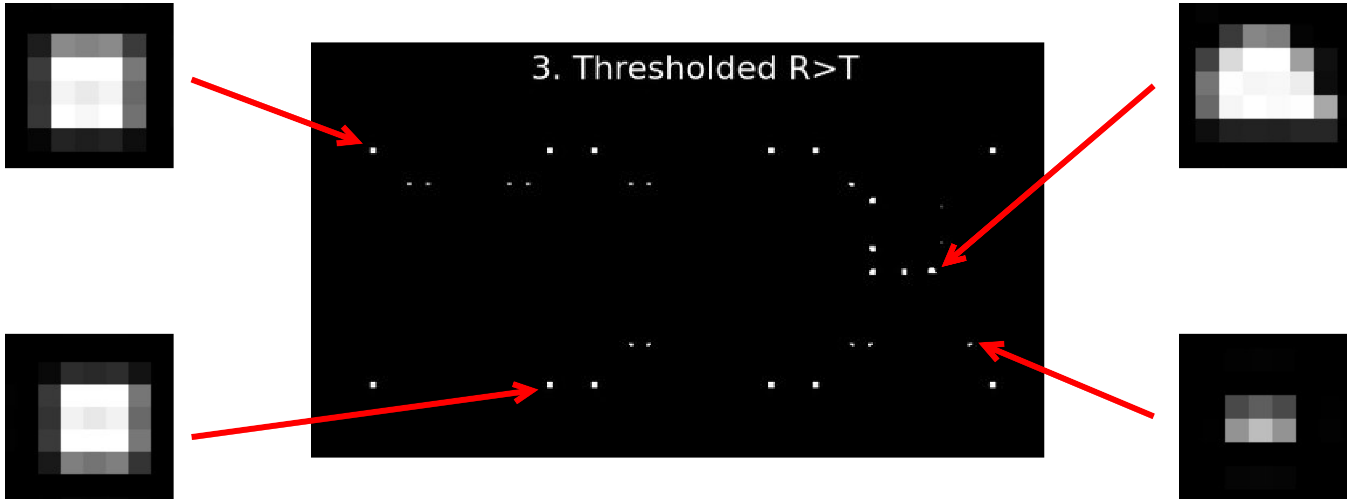
Harris Corner Detection Example



If we keep pixels where corner strength (R) ≥ 0.5% (T) of the strongest corner found

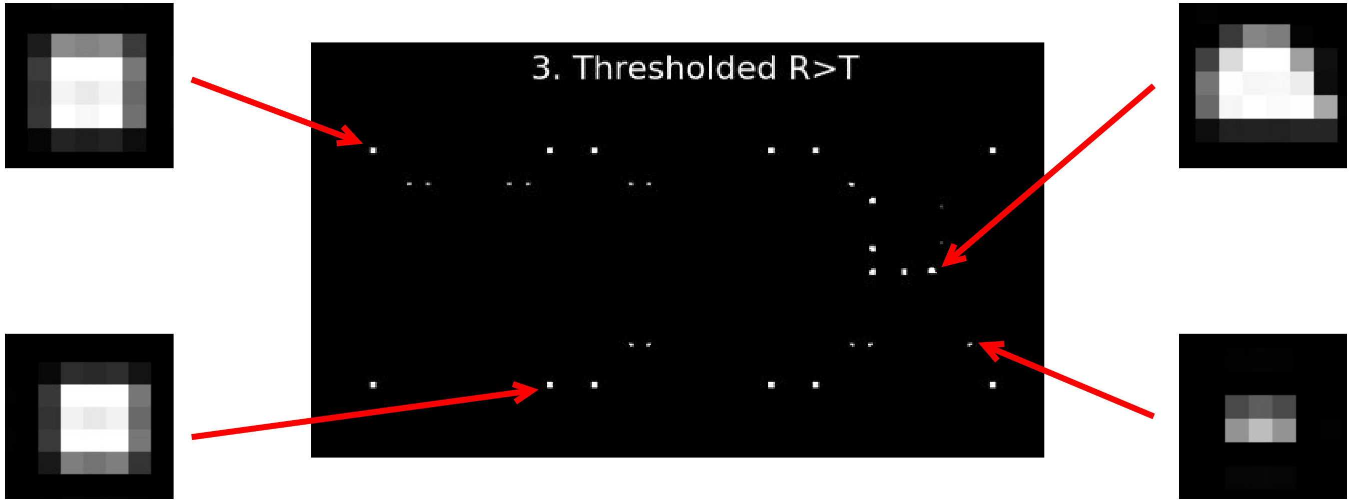


Harris Corner Detection Problem



The magnified image regions show large R values at and around each of the corners in the image. To find the exact locations of the corners, we need to detect the peak of each of these clusters in the response image.

Harris Corner Detection Problem



We need to use a peak finding (Thinning) algorithm.

Non-Maximal Suppression

Non-Maximal Suppression

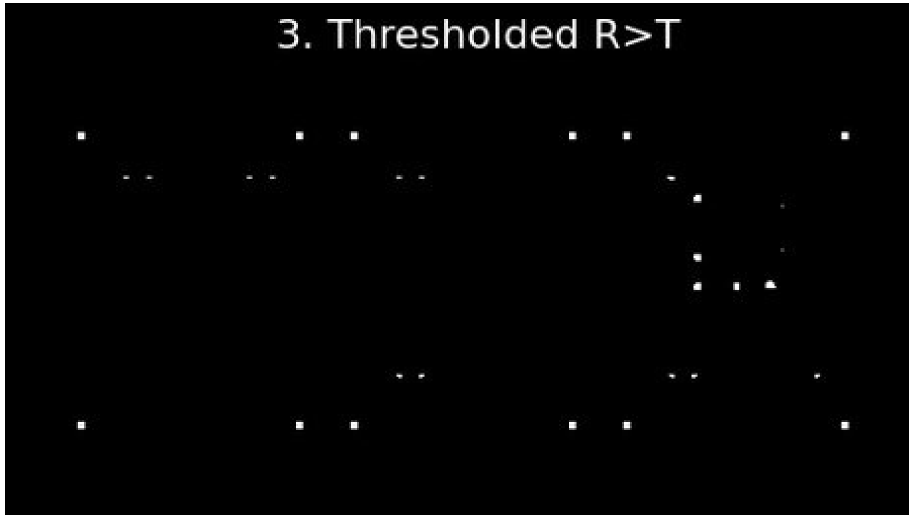
1. Slide a window of size k over the image.

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).

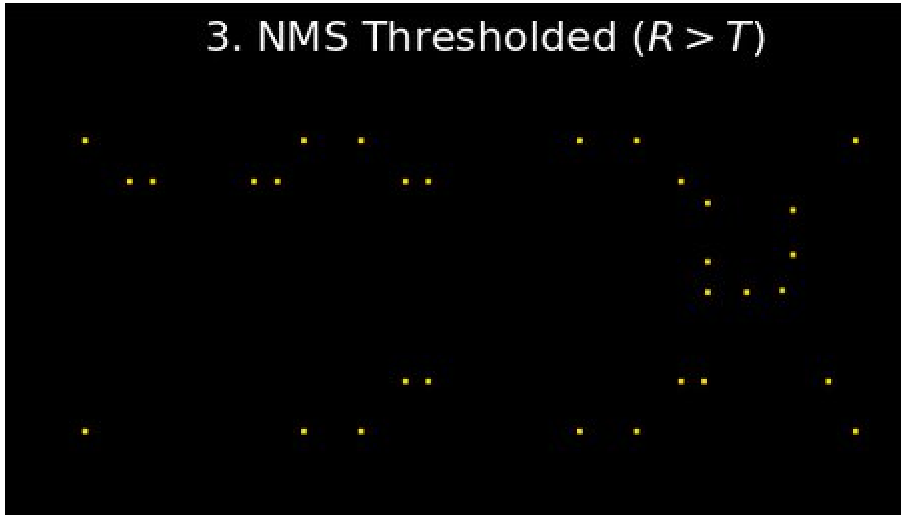
Suppress | Suppress | Retain

Used for finding Local Extrema (Maxima/Minima)

Harris Corner Detection with NMS

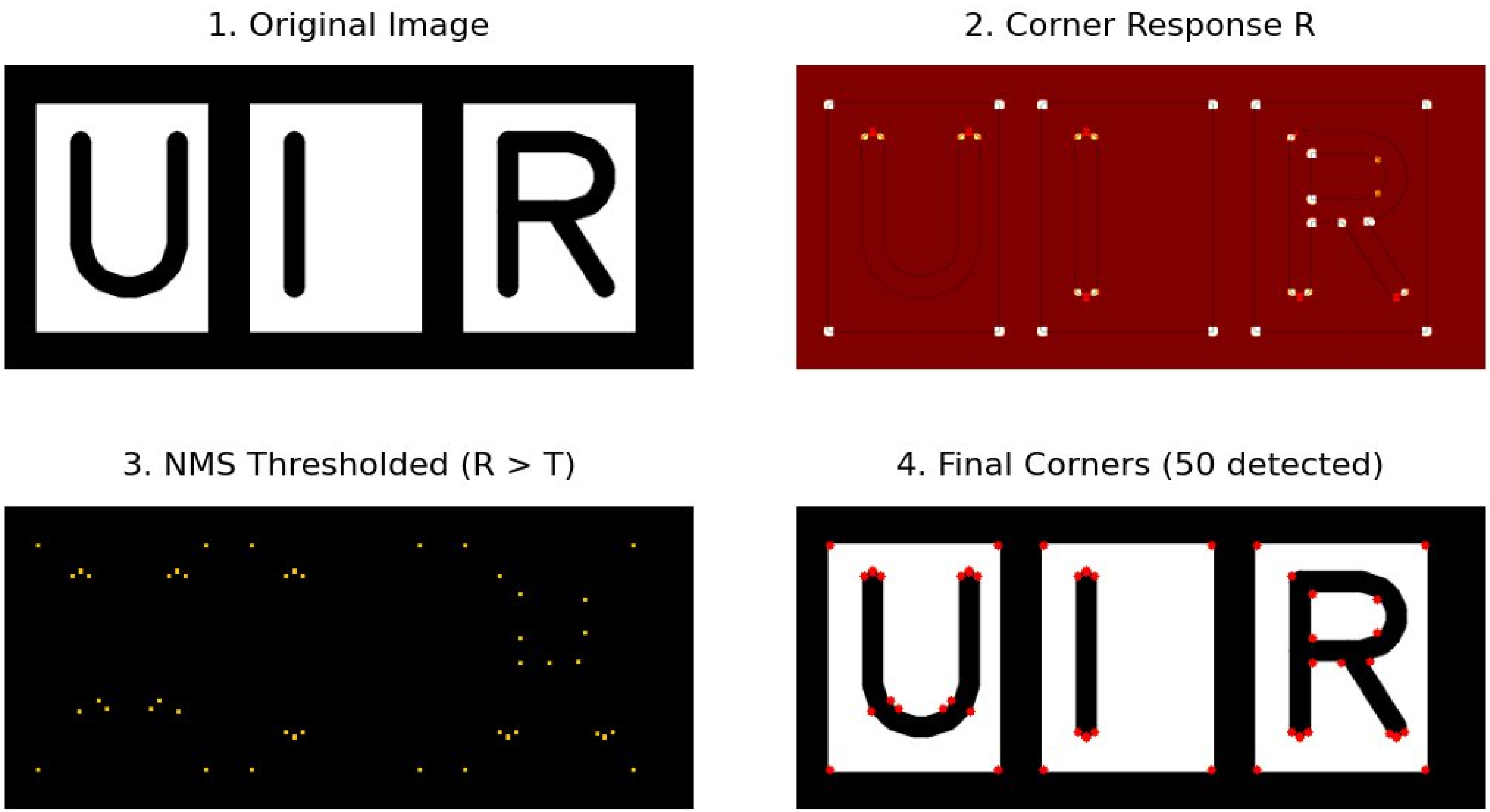


Thick blobs around every corner

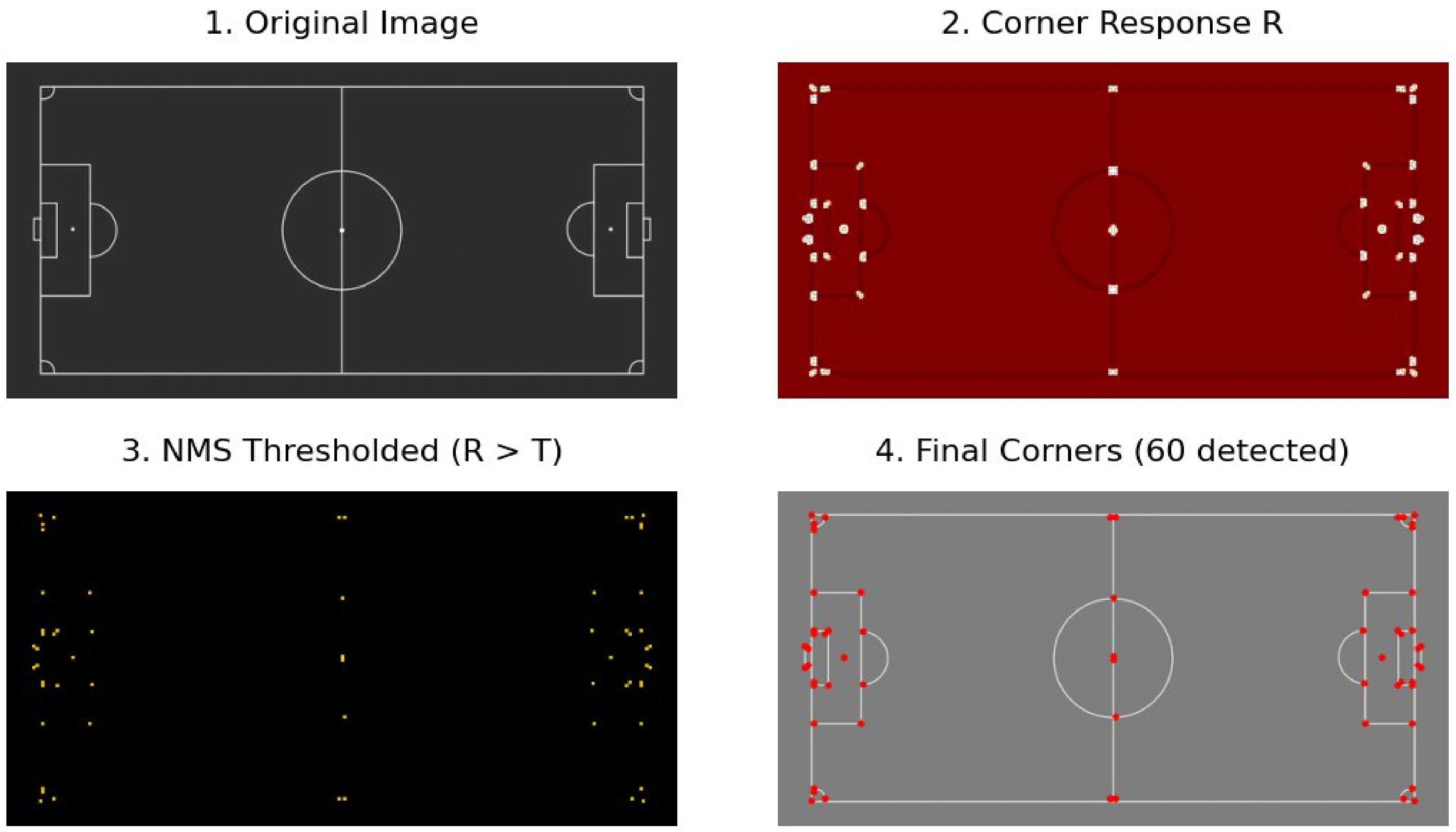


Clean single dots (NMS applied)

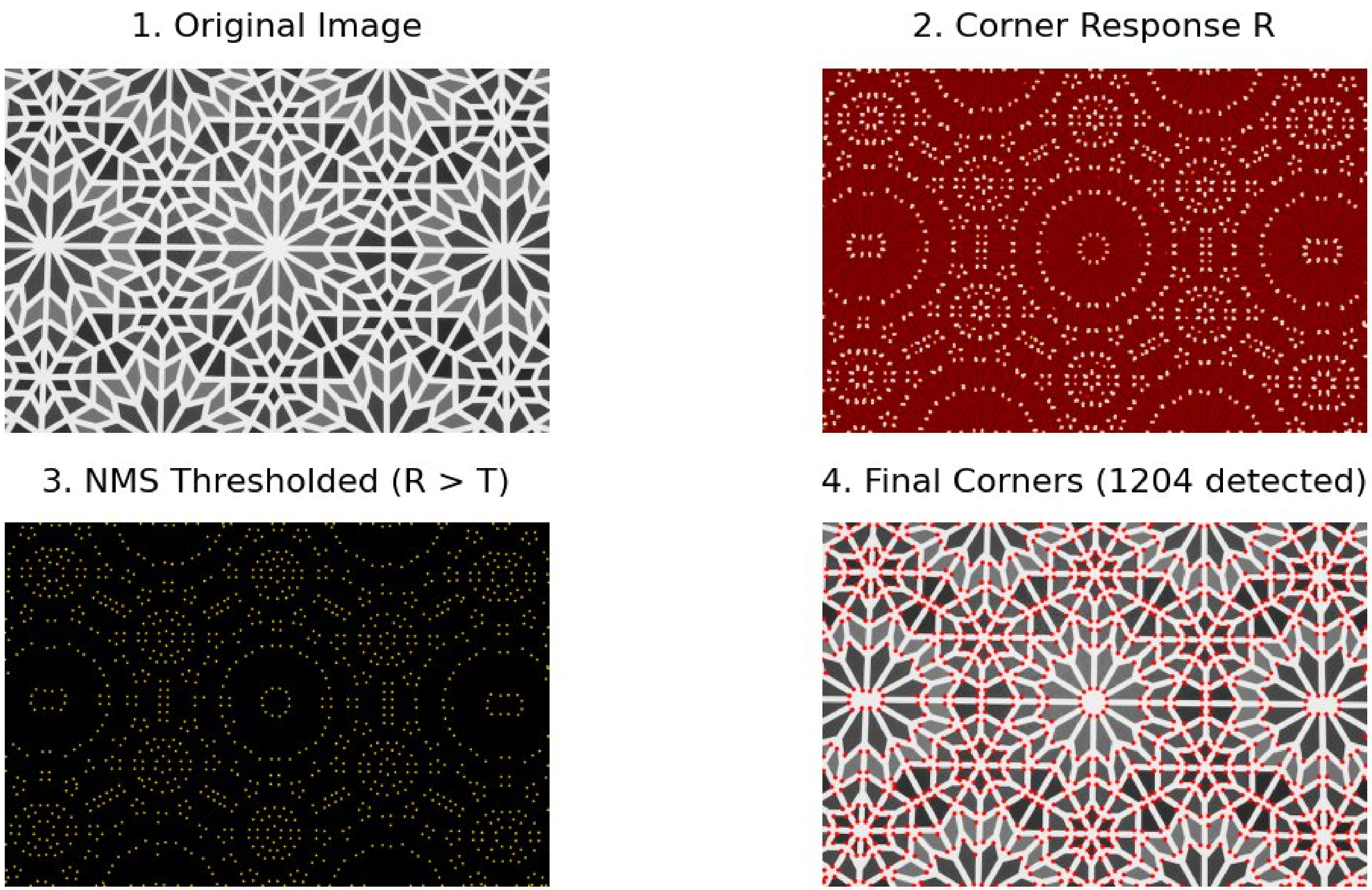
Harris Corner Detection Example



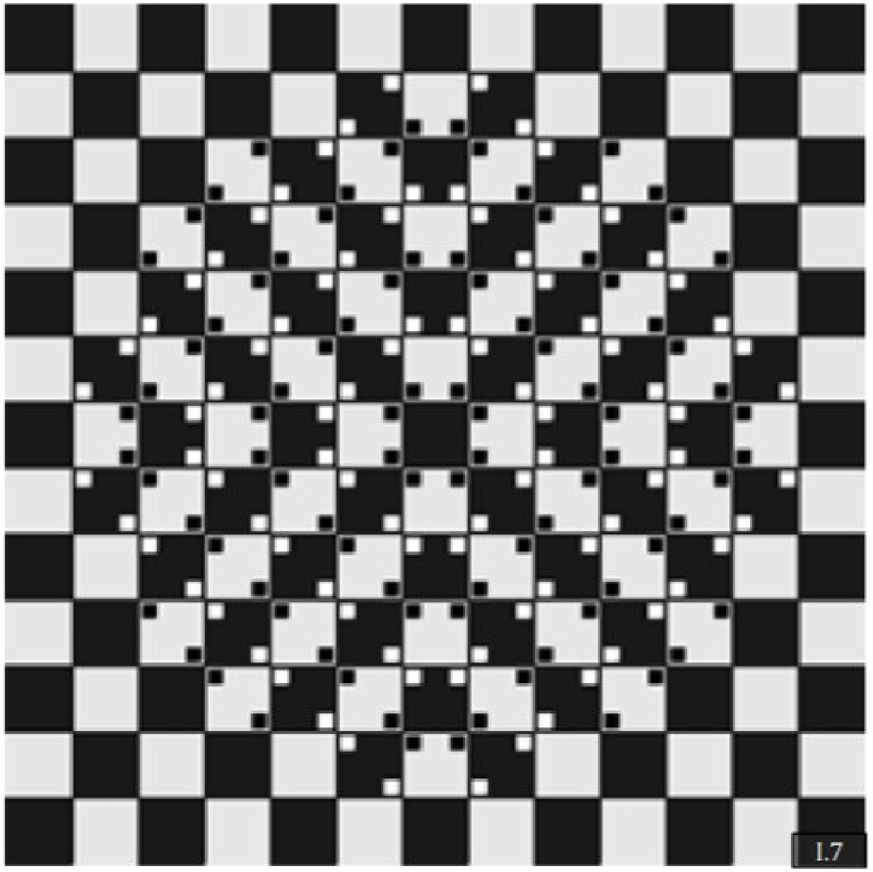
Harris Corner Detection Example



Harris Corner Detection Example

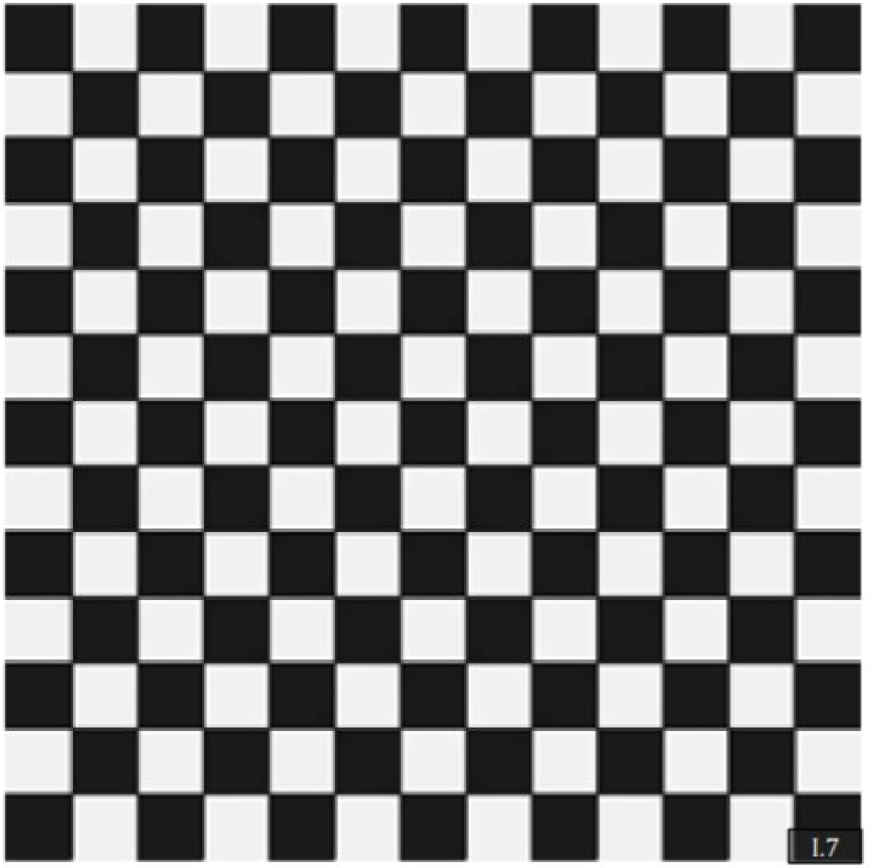


Corner Illusion: The Bulge



Kitaoka, 1998

Corner Illusion: The Bulge



Kitaoka, 1998