Computer Vision

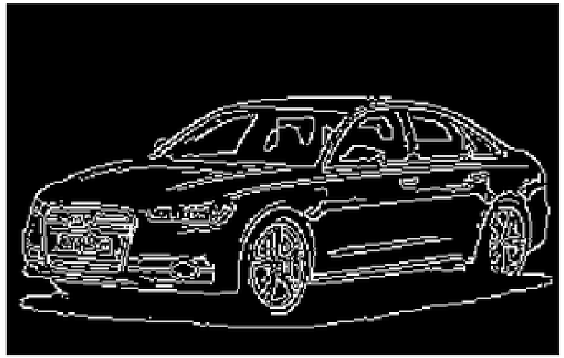
Feature Extraction

(Corner and interest point detection, Local variant feature detectors and descriptors)

Contents

- Corner and interest point detection
- Local variant feature detectors and descriptors

Why Feature Descriptor?

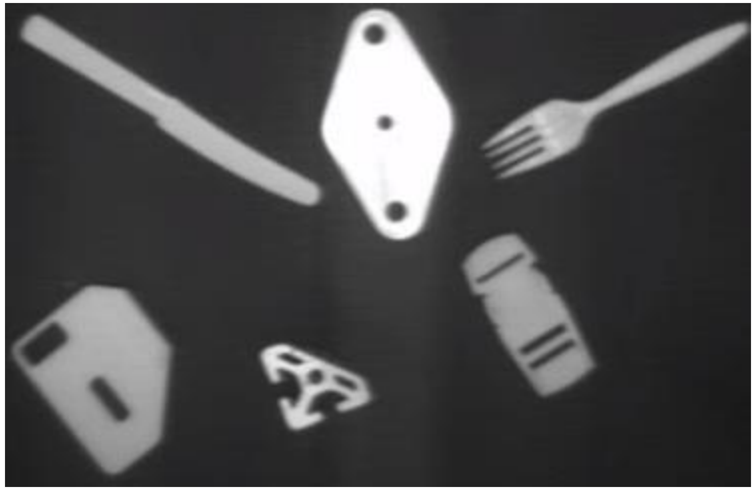


- Edges can be used to differentiate cat and dog
- Easy to classify cat and dog

- Shape and edges are known
- Can differentiate the two images
- Can not be used to match two objects
- Feature descriptor is a representation of the image
- It contains important information about the image

Object Detection

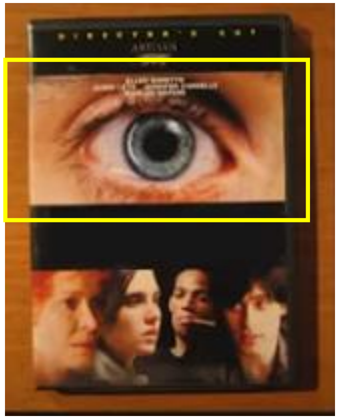
- Detects and match features which are descriptive and unique



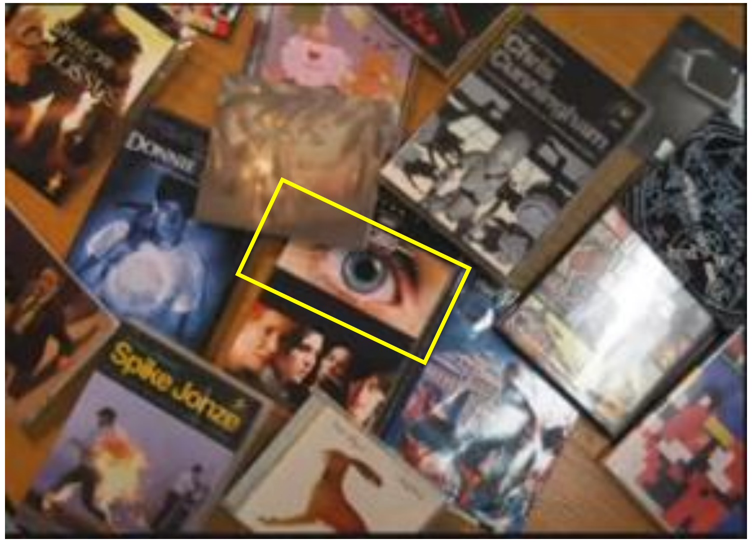
- Apply thresholding to convert it to binary
- Compare geometric properties and recognize the objects
- Also recognize position and orientation

- More complex
- Still, can apply thresholding
- And recognize letter and identify license plates

Object Detection



Template



Image

Find key point (interest points) and match them to recognize objects

- Find template in image
- Template may not exactly be same as the object to be detected in image
- Objects in image may be
  - Rotated
  - scaled
  - occluded
- To detect such object create several templates with different rotations and scales
- Technique would be computationally expensive
- Use key points to reduce complexity

Feature descriptors

- Interest points can be particular elements like unique points, edges, or corners
- Define local patches surrounding interest points
- Extract feature descriptors for each key point
- Match descriptors in images to find correspondence for image matching and other applications

Local and Global Descriptors

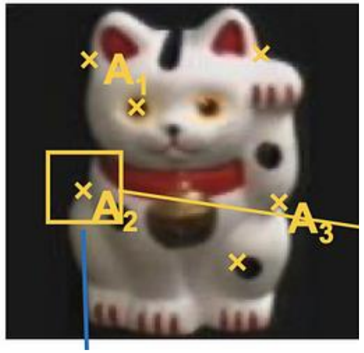
Global feature representation

Local feature representation

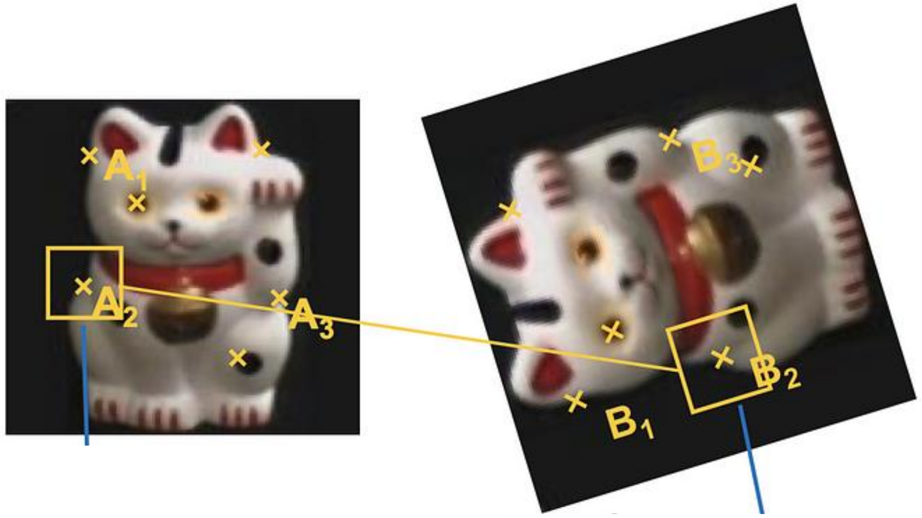
Local and Global Descriptors

- Global Descriptors
  - Describe the image as a whole
  - Contour representations, Shape descriptors, Texture features
  - DoG, HOG, histograms of optical flow (HOF) etc are few examples
  - Limitations: difficult to detect objects with occlusions, profile variations
- Local Descriptors
  - Describes a patch within an image
  - Use multiple local descriptors to match an object
  - More accurate than global
  - Provides more robustness against occlusions and profile variations
  - SIFT, SURF, LBP, BRISK, MSER, and FREAK are examples

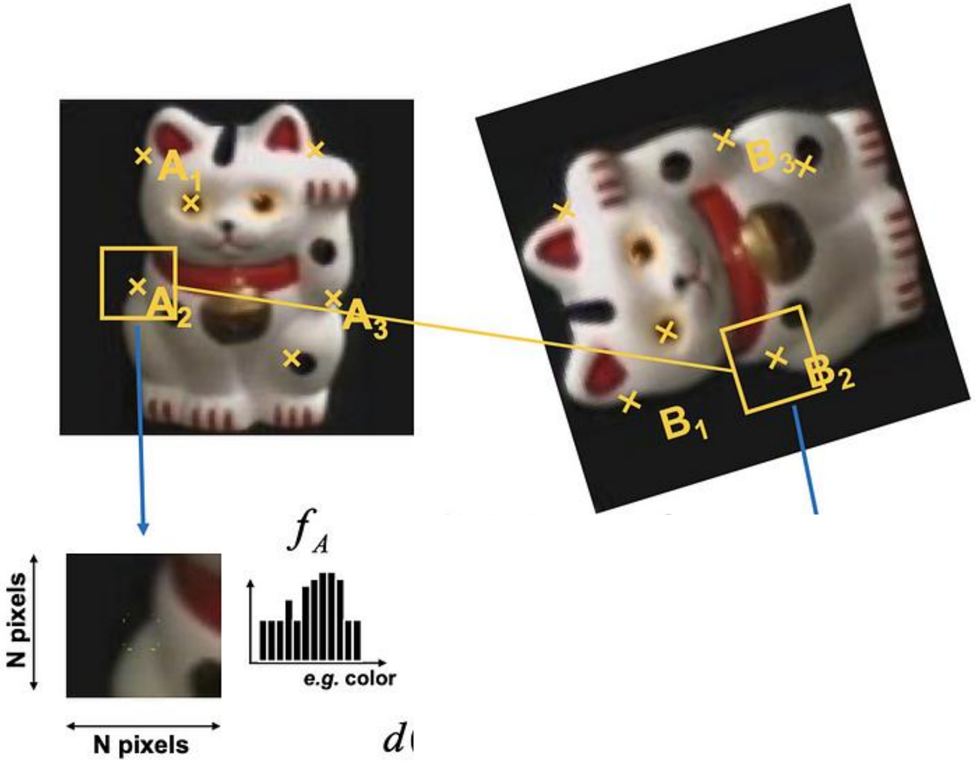
Overview of Local Features Matching



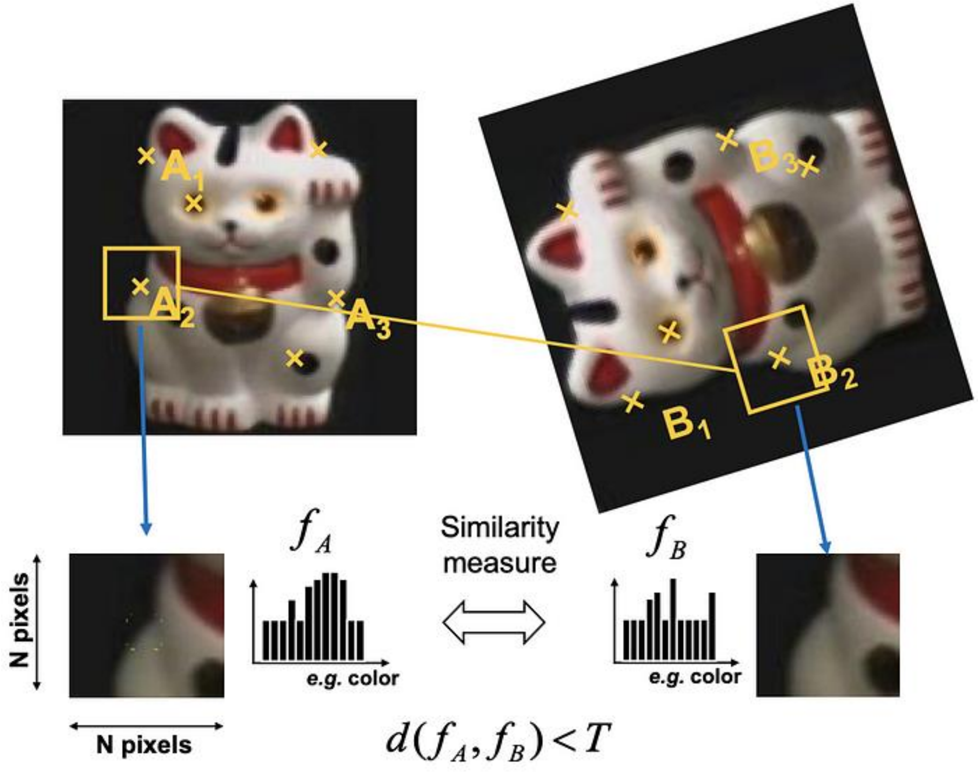
Overview of Local Features Matching



Overview of Local Features Matching



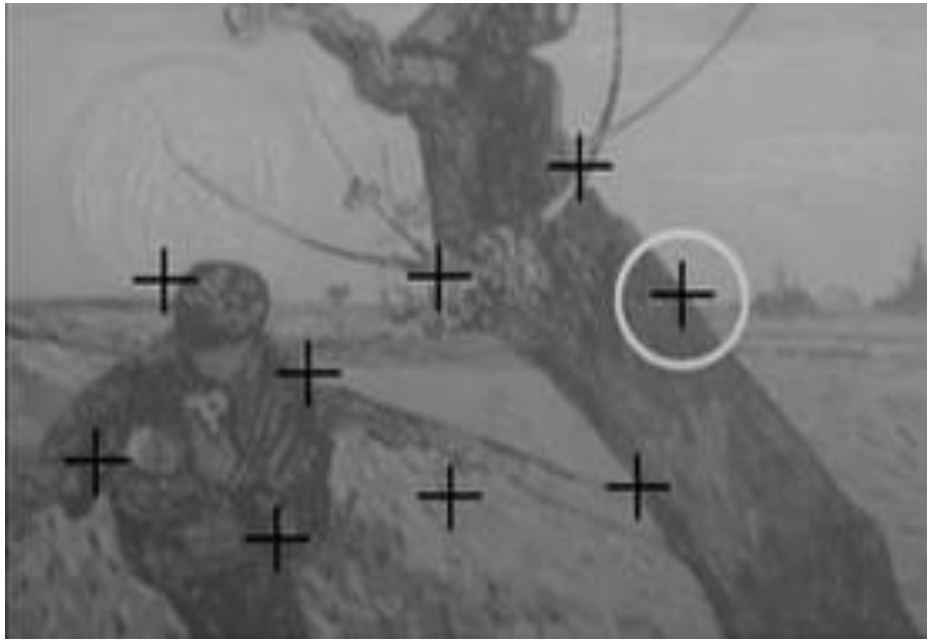
Steps for Local Features Matching



- Determine distance between local descriptors of two interest points
- Match the local descriptors between images
- If matched, determine correspondence between two patches
- Apply same correspondence to other interest points

How to define interest point?

- There may be a change of one or more image properties (color, texture, intensity etc)
- Interest points should be robust to these changes
- Corner point is an example of interest point

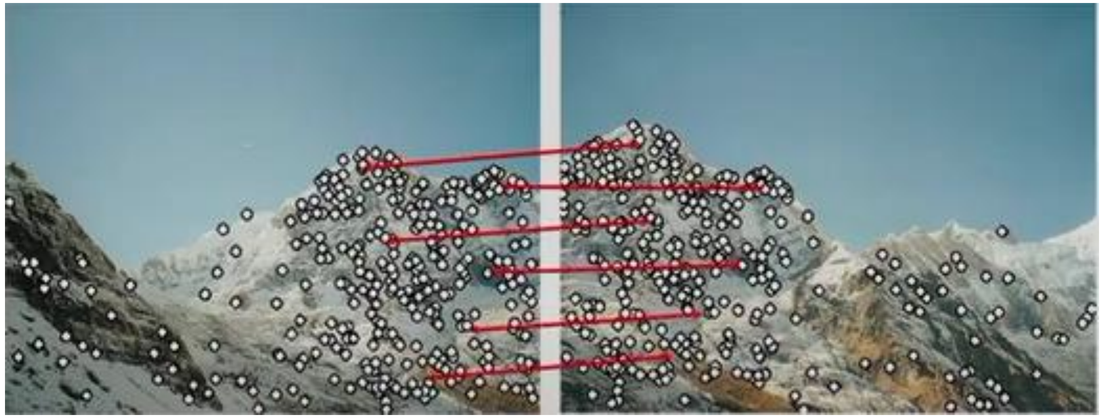


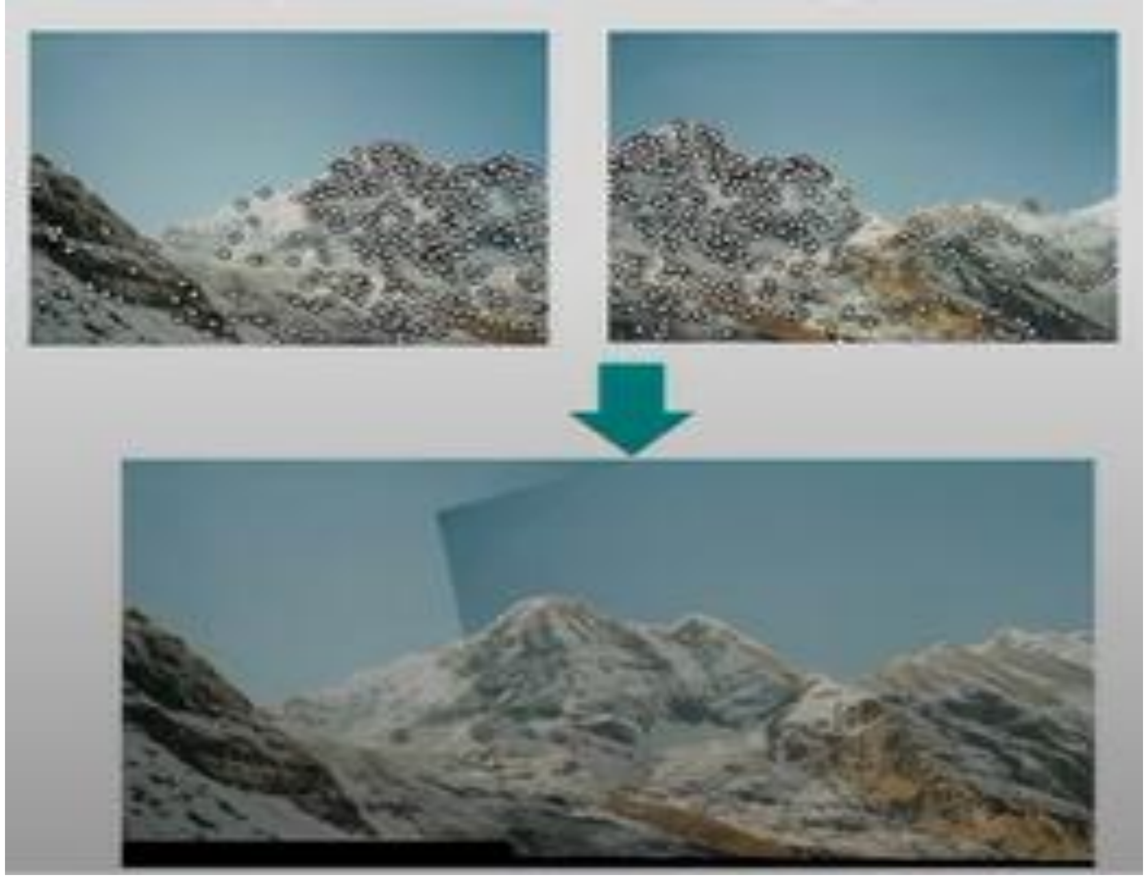
Interest point Detection and Applications

- Stereo image matching
  - Choose corner interest points on both images
  - Apply image matching between left and right image for stereo correspondence

Interest point Detection and Applications

- Join two images to get a bigger image
  - Choose interest point (corner point)
  - Use correspondence between interest points for image alignment

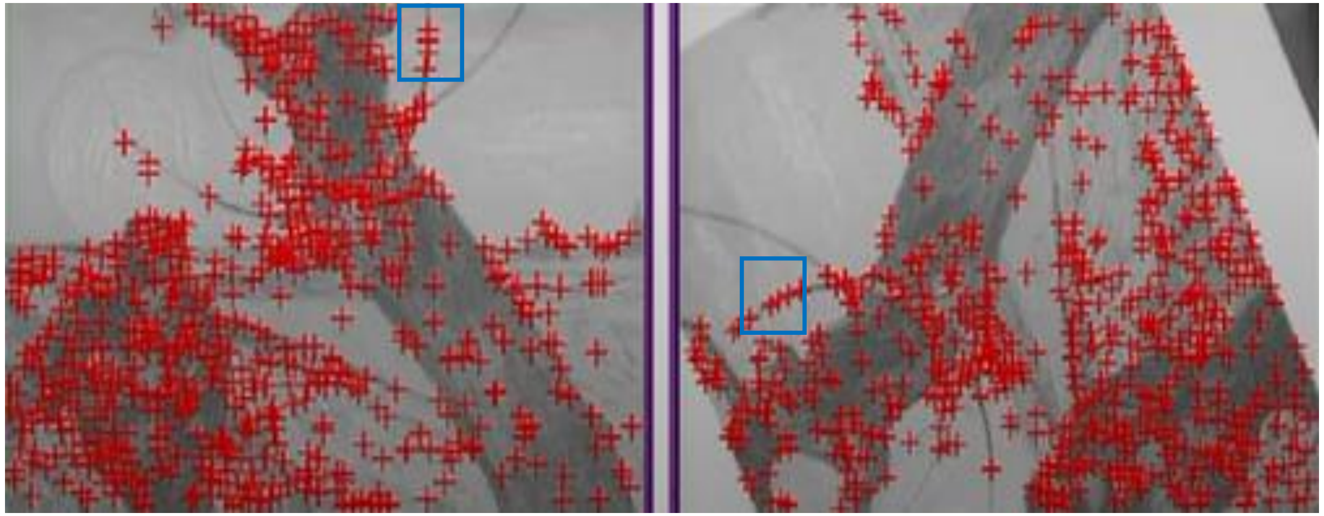




Panorama Stitching

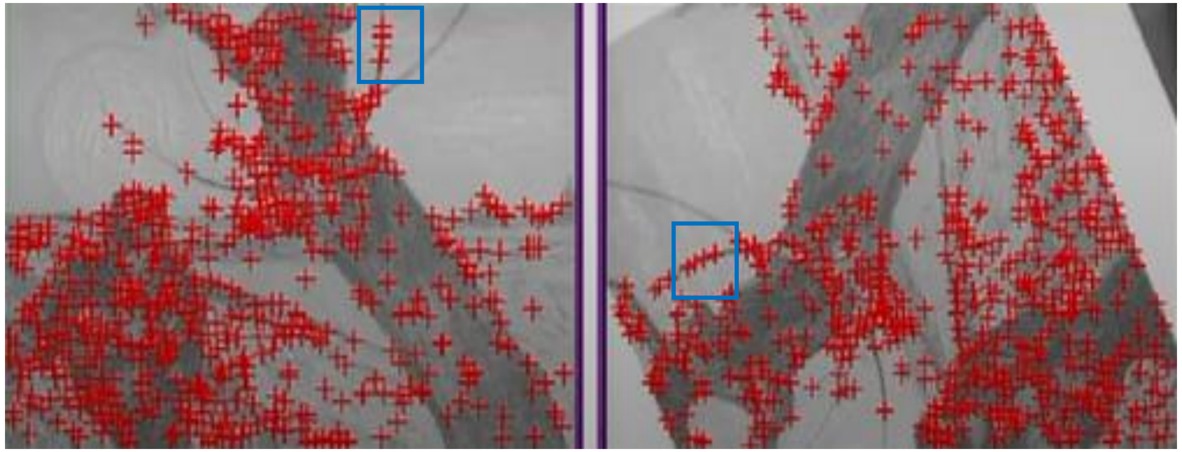
Characteristics of good features of interest points

- Repeatability
  - Find same feature in multiple images irrespective of geometric and photometric transformations
- Saliency
  - Each feature is distinct/unique
- Compactness
  - fewer features than number of image pixels which represent entire image



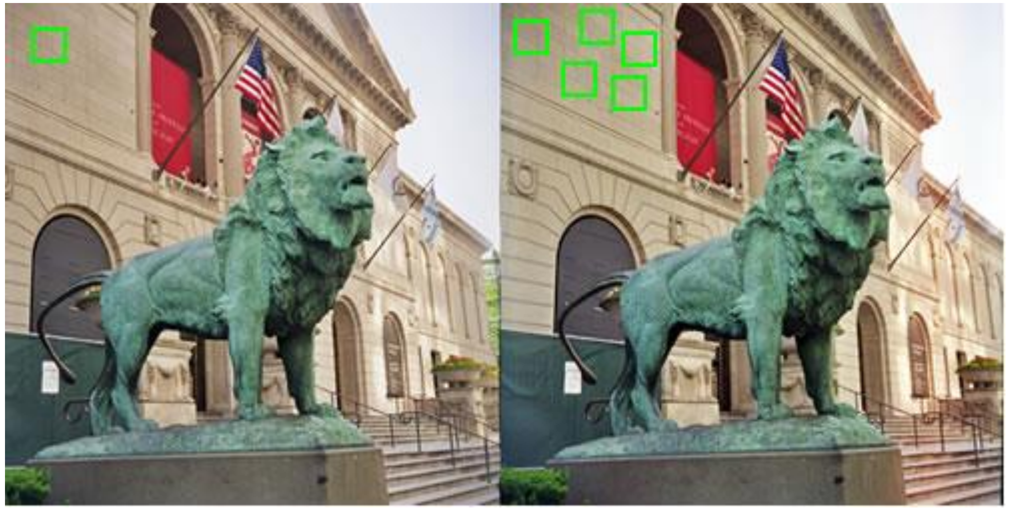
Characteristics of good features

- Efficiency
  - Computationally faster for real time applications
- Locality
  - Occupy small area of image
  - Robust to clutter and occlusion
- Covariant
  - Should be detected in corresponding locations despite geometric and photometric variations



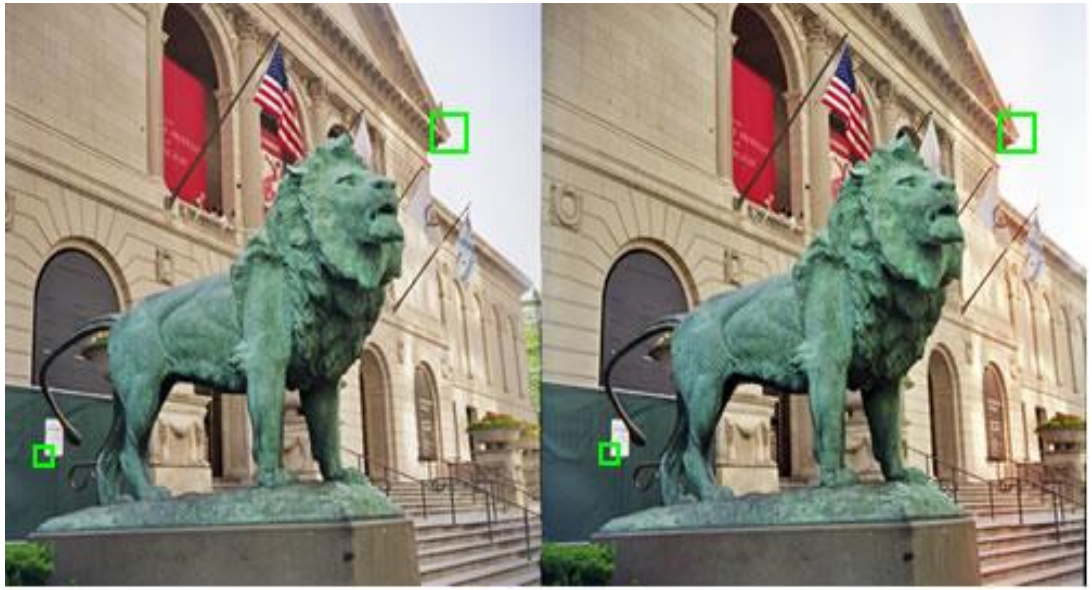
Patches of images for matching

- Select a patch in one image
- Match it with a patch in the other image



- Patch should have unique shape in the image

Patch Matching



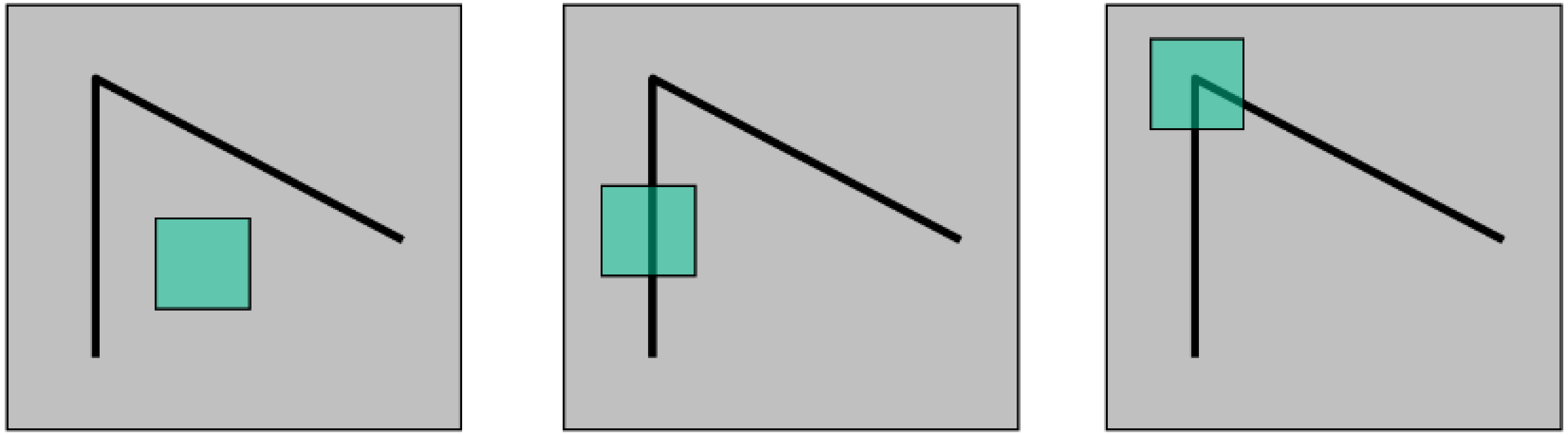
- Corners are unique points in images
- Locate patch around corners and map it to image feature

Corner Point

- Corner points are special case of interest points
- Corner –
  - Is the intersection of two or more edge segments
  - Well localized or have a well-defined location in the image space
  - Maintain their stability against scaling and rotation
  - Can compute the interest points accurately
  - Offers effective detection

Corner Point

- Window with a fixed size is chosen to detect the corner
- If window is shifted,



Flat portion: No change in pixel intensities in all directions

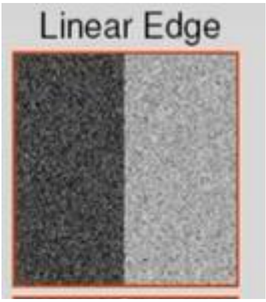
Edge: No change in pixel intensities along edge direction

Corner: Significant change in pixel intensities in all directions

Difference in pixel intensities in window can be used to identify corners

Examples of Derivatives

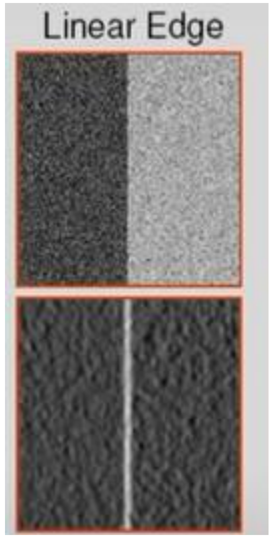
image



Examples of Derivatives

image

X derivative

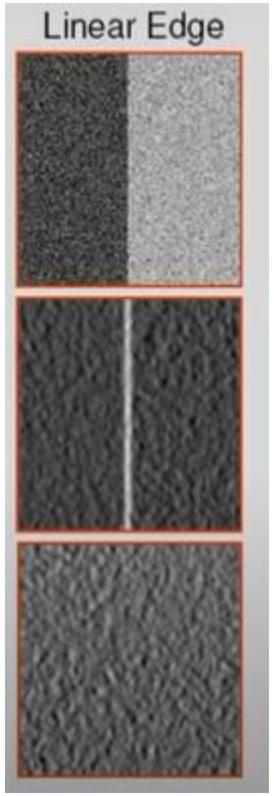


Examples of Derivatives

image

X derivative

Y derivative

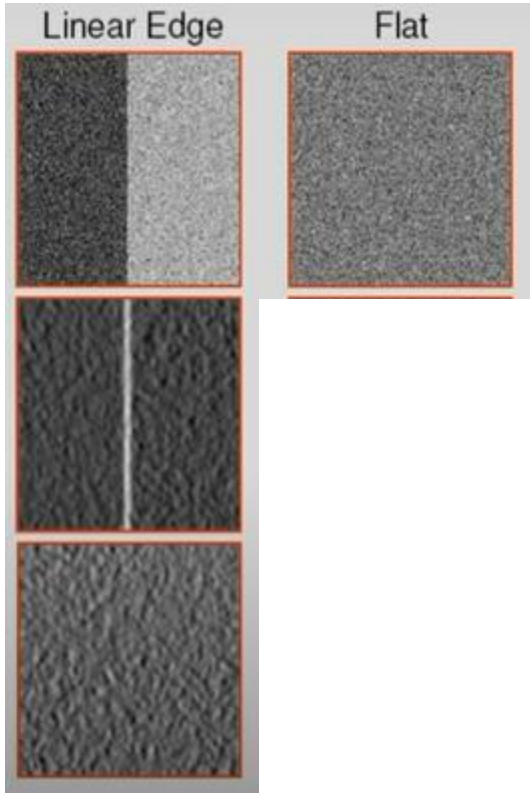


Examples of Derivatives

image

X derivative

Y derivative

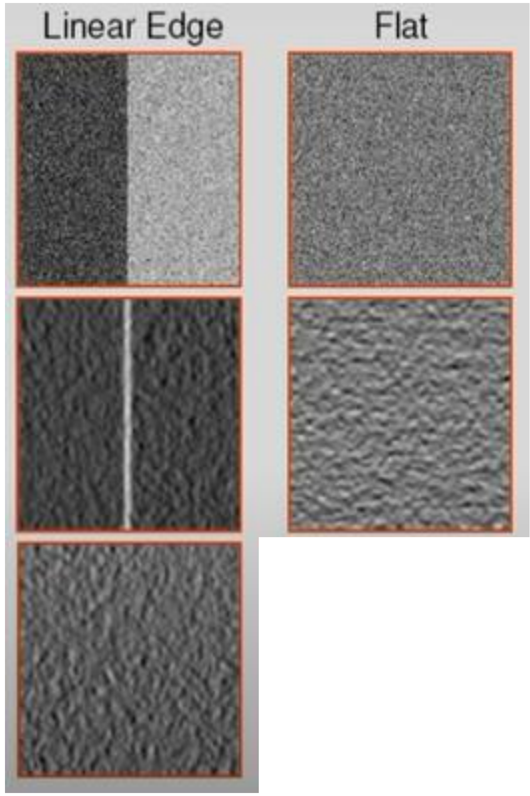


Examples of Derivatives

image

X derivative

Y derivative

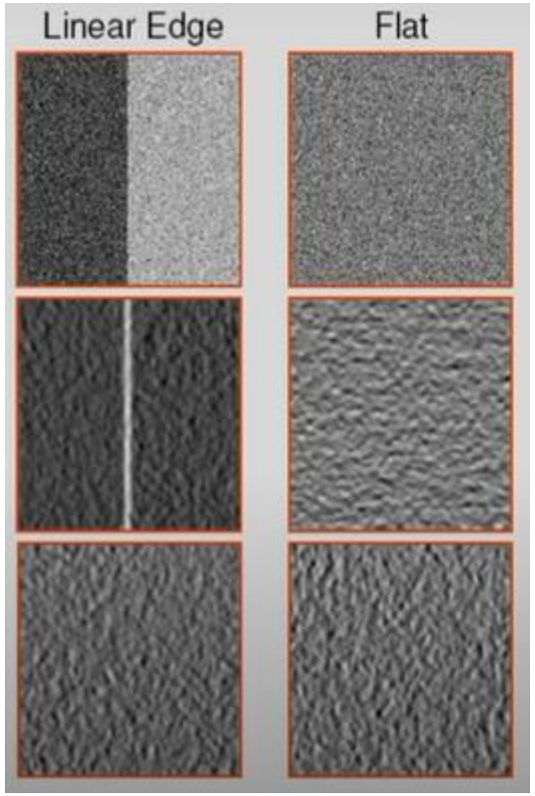


Examples of Derivatives

image

X derivative

Y derivative

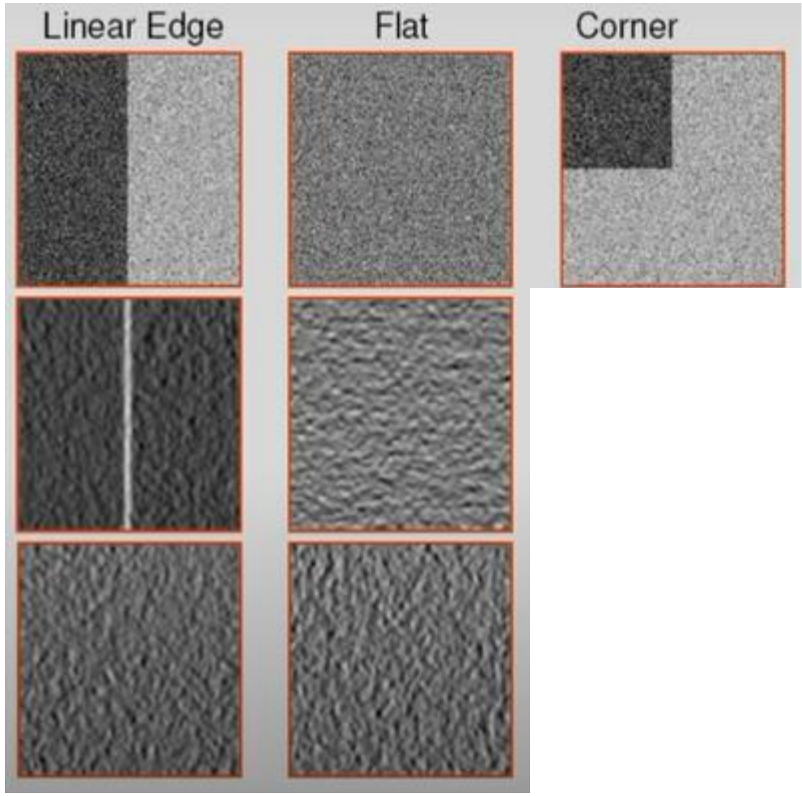


Examples of Derivatives

image

X derivative

Y derivative

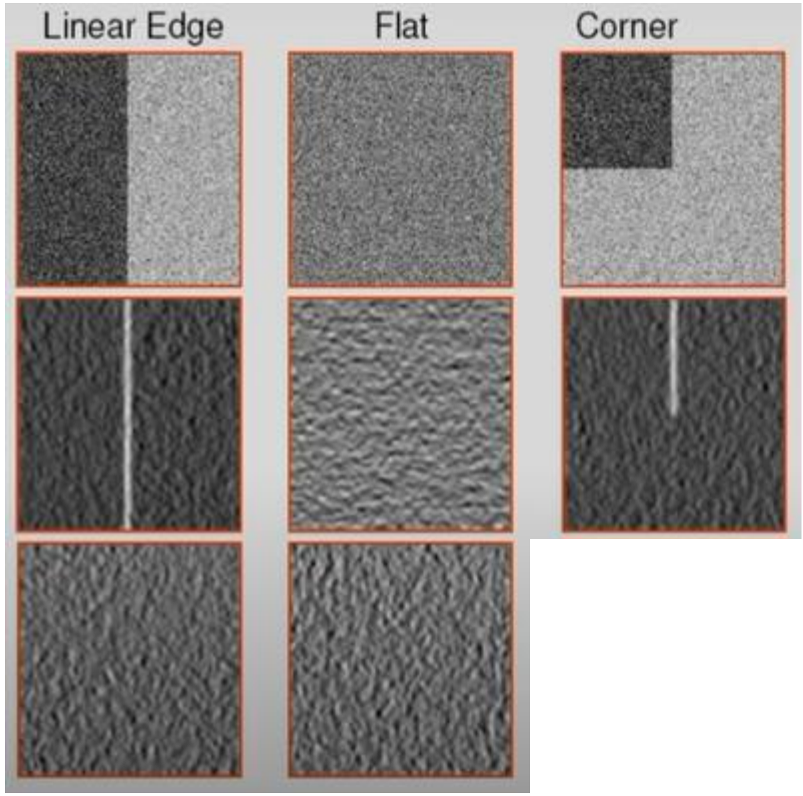


Examples of Derivatives

image

X derivative

Y derivative

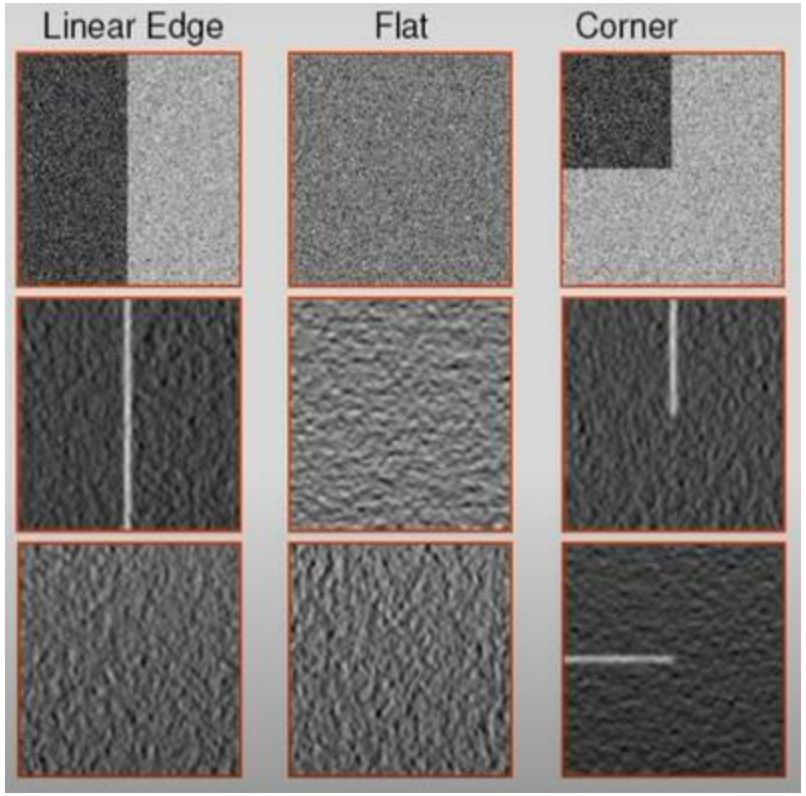


Examples of Derivatives

image

X derivative

Y derivative



Corner detector

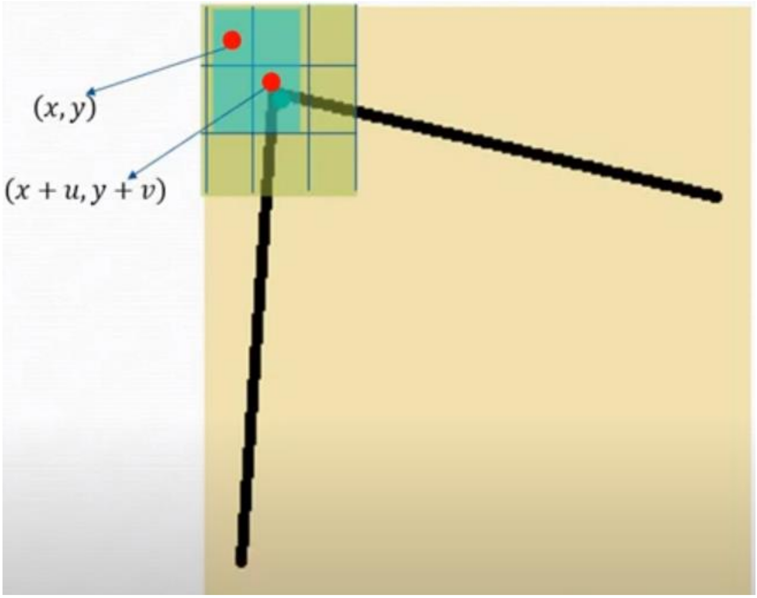
- Locate interest points where the surrounding neighborhood shows edges in more than one direction
- These points are image corners
- Change in intensity is tested for the shift (u,v) of a window

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

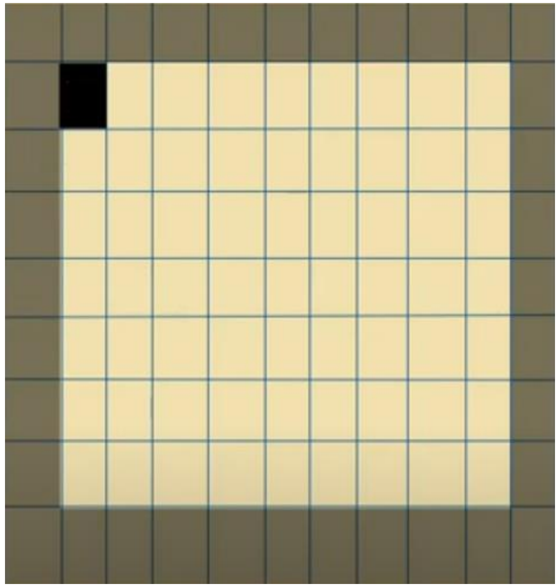
- For nearly constant patches, change of intensity, E(u,v) is almost 0
- For different patches, E(u,v) is large
  - Then it is corner point

Corner detector

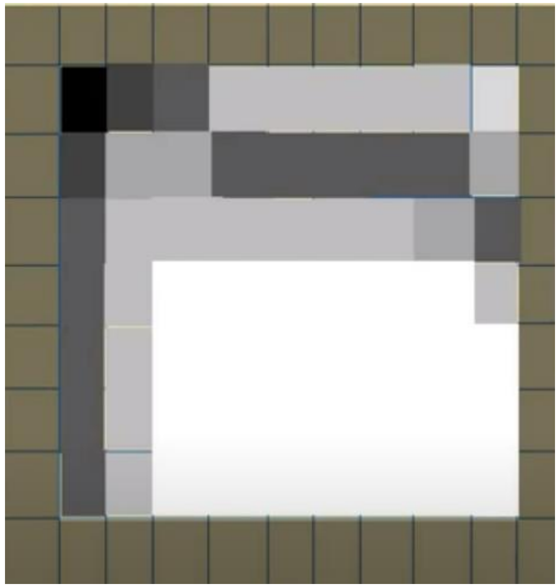
$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2$$



E(u,v) for window at the corner point

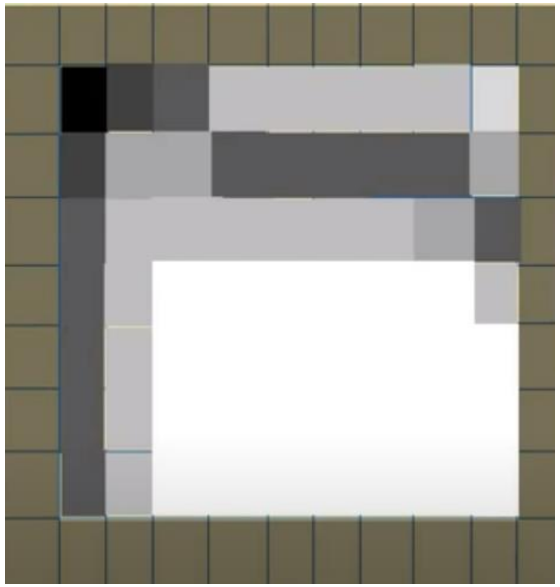


E(u,v) for window at each point of image



Maximum value: black, Minimum value: white

Corner detector



Maximum value: black,
Minimum value: white

- E(u,v) for
  - Corner has maximum value
  - Edges has relatively low value
  - Patches have minimum value
- Entire process of calculation is computationally expensive, especially for large images
- Harris corner detector is computationally less expensive

Harris Corner Detector

- For small shifts, (u,v), E(u,v) can be approximated as

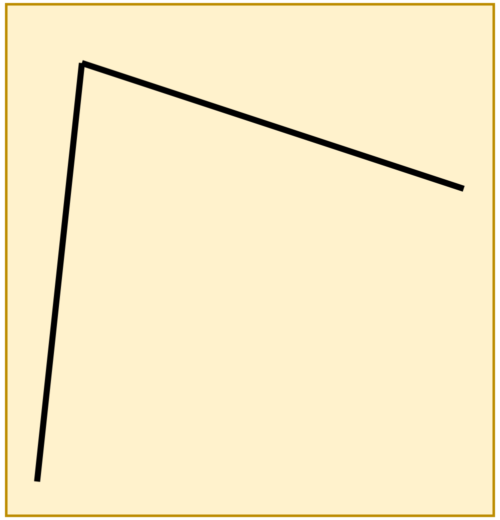
$$E(u,v) = [u,v]M\begin{bmatrix} u \\ v \end{bmatrix}$$

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

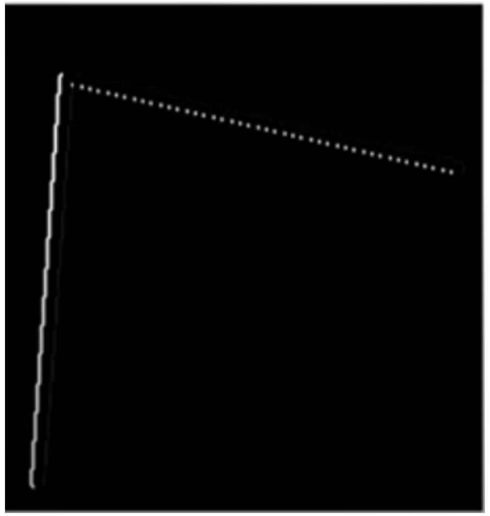
- I_x and I_y are derivatives in x and y directions in window (patch)

Harris Corner Detector

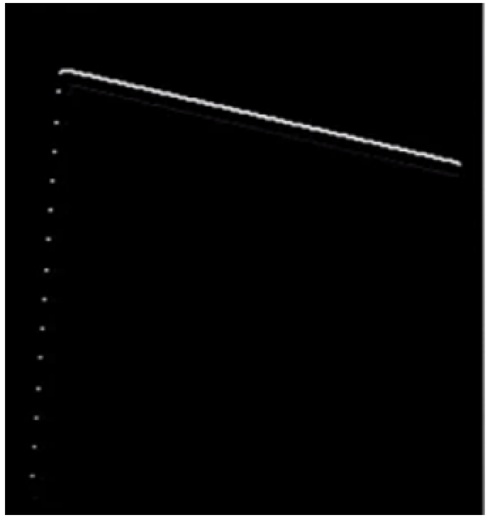
- Apply prewitt or sobel operator to determine gradients, I_x and I_y
- For perfect vertical edge, $I_y = 0$ and for perfect horizontal edge, $I_x = 0$
- For flat regions, I_x and I_y are zero



Image



I_x



I_y

Prewitt operator

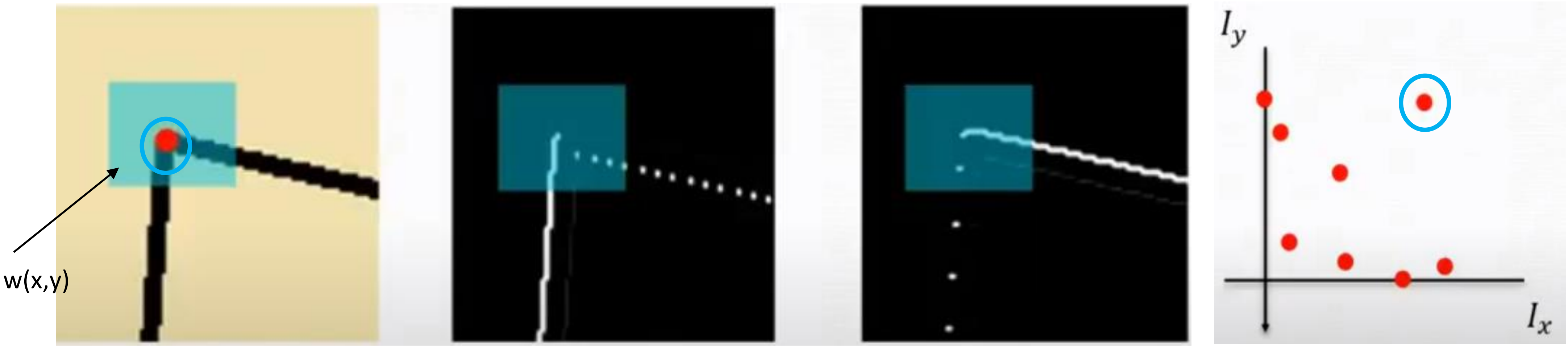
1	0	-1
1	0	-1
1	0	-1

To generate I_x

1	1	1
0	0	0
-1	-1	-1

To generate I_y

Harris Corner Detector

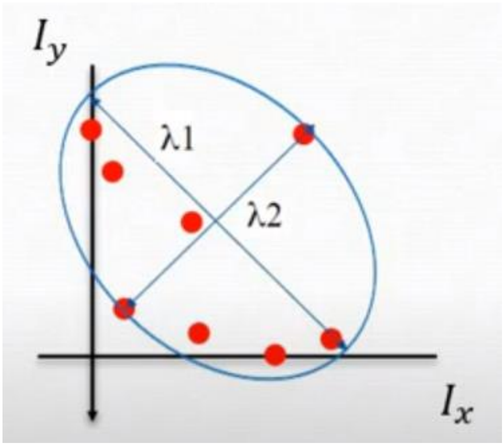


Window, w(x,y)

I_x

I_y

Gradients of points in window



λ_1 and λ_2 are large

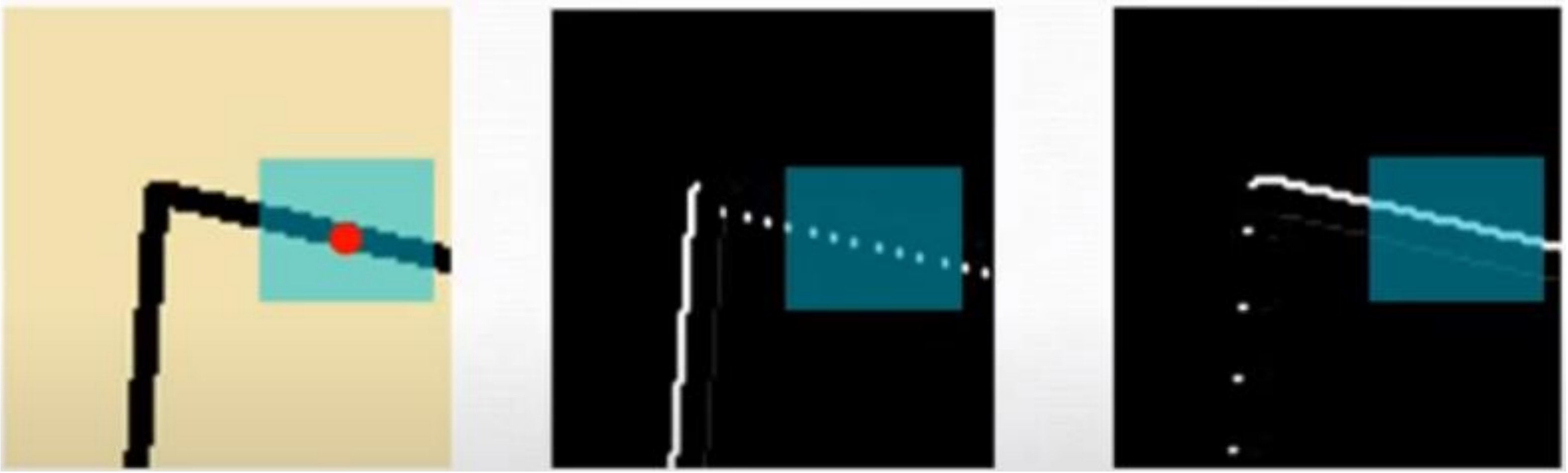
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

- λ_1 and λ_2 are Eigen values of M
- λ_1 corresponds to major axis and λ_2 corresponds to minor axis

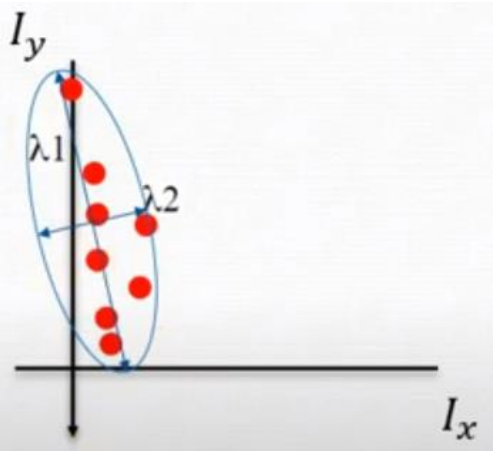
If λ_1 and λ_2 are large, point at the center of window is a corner

Harris Corner Detector

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



I_x $\qquad$ I_y

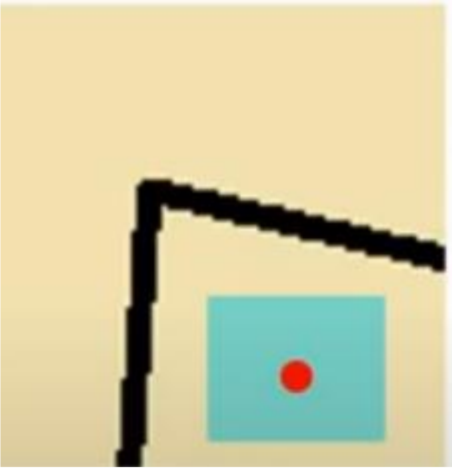


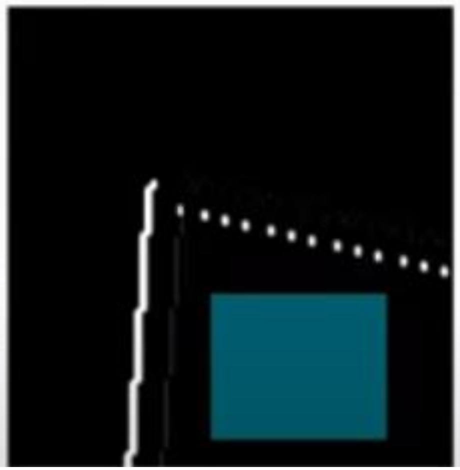
Gradients of points in window

Point at the center of window is on the edge

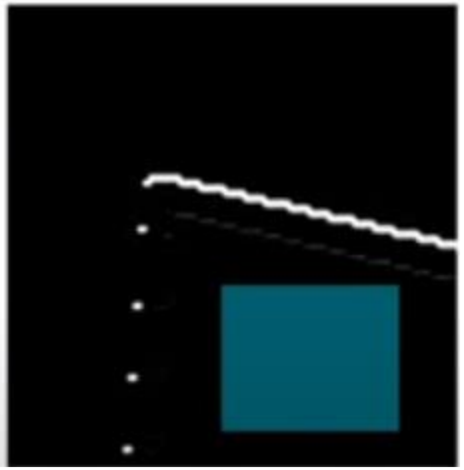
Harris Corner Detector

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

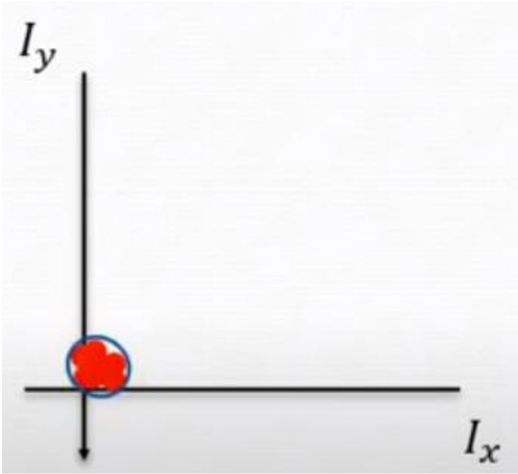




I_x



I_y

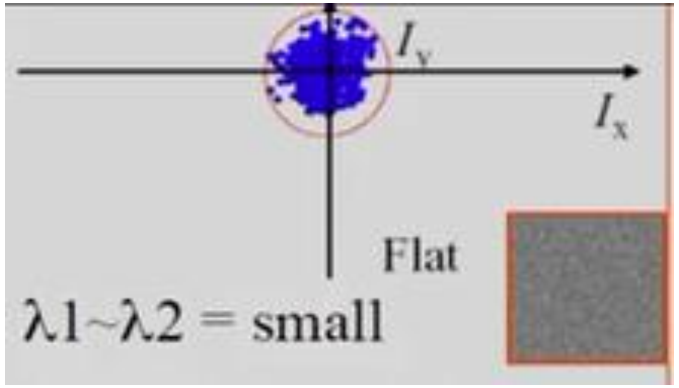


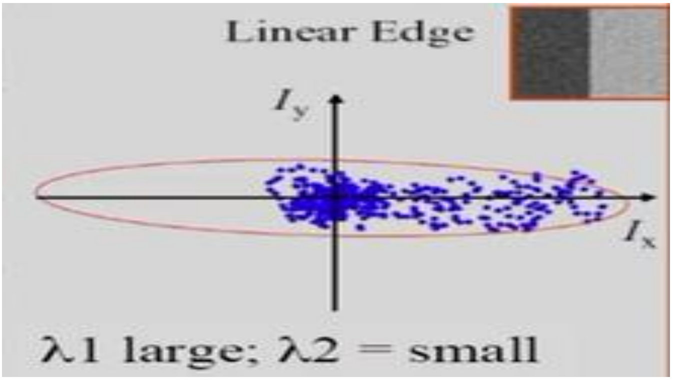
Gradients of points in window

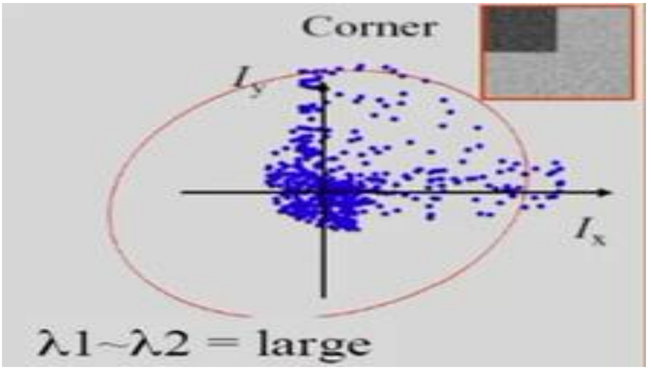
Point at the center of window is in flat portion of image

Eigen values of M can identify corner, edge and flat patch

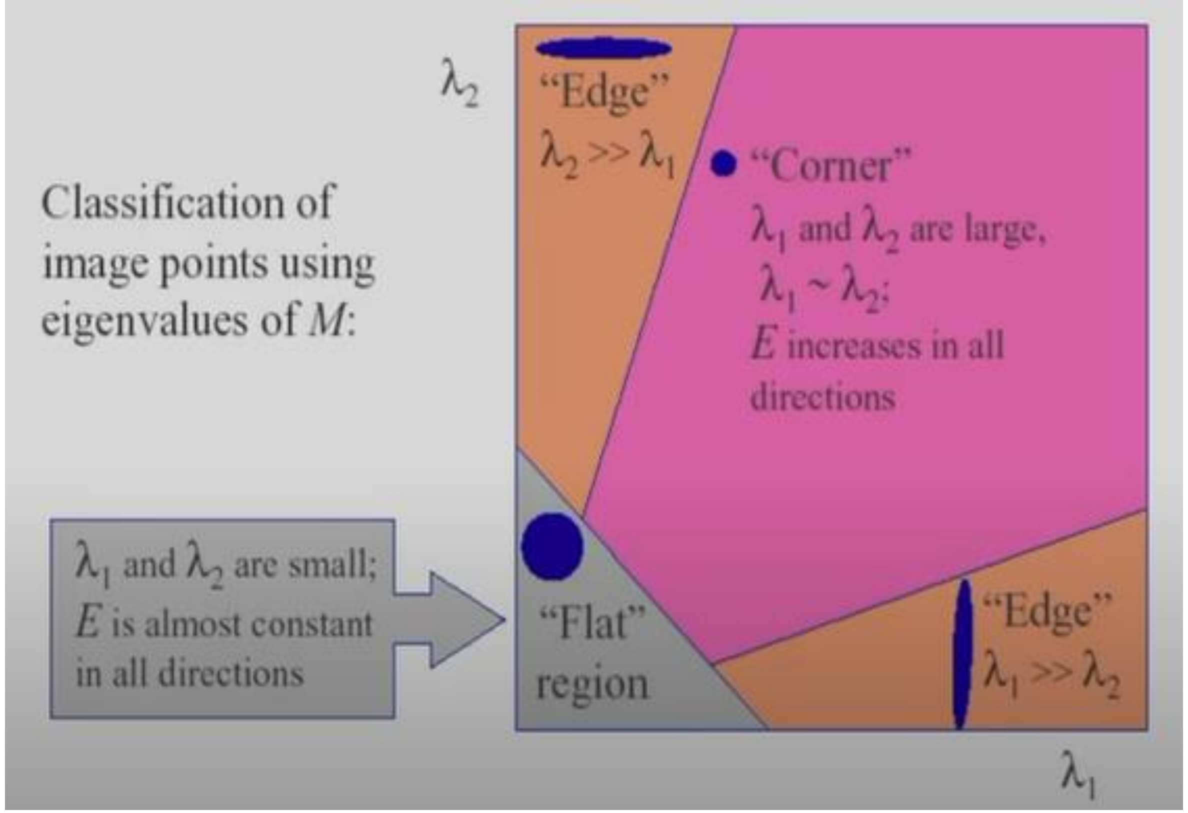
Eigen Values for Corner Point Detection







Eigen Values for Corner Point Detection



- One eigenvalue is significantly high
  - derivative with respect to one direction is much stronger than the other
  - pixel lies on an edge
- Both eigenvalues are small
  - pixel intensities do not change in any direction
  - pixel lies on a flat region
- Both eigenvalues are large
  - pixel intensities largely change in both *x* and *y* direction
  - pixel lies at the corner

Harris Corner Detector

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

- Computation of eigen value is costly
- Instead, compute corner response,
  - R = det{M} – k (trace {M})^2

  Where Det{M} = λ_1x λ_2 and Trace{M} = $\lambda_1 + \lambda_2$
    - Or,

      Det{M} = $I_x^2 \ I_y^2 - I_{xy}^2$

      Trace{M}=$I_x^2 + I_y^2$
- Empirically determined value of k is 0.04 – 0.06
- R is +ve and large for corner points
- R is –ve and large magnitude for edge
- |R| is small for flat region

Ex: Harris Corner Detector

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

window

Image

1	0	-1
2	0	-2
1	0	-1

Sobel mask for I_x

1	2	1
0	0	0
-1	-2	-1

Sobel mask for I_y

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

Ex: Harris Corner Detector

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

1	0	-1
2	0	-2
1	0	-1

Sobel mask for I_x

1	2	1
0	0	0
-1	-2	-1

Sobel mask for I_y

0	0	0	0	0
0	-20	10	0	0
0	-10	30	0	0
0	0	40	0	0
0	0	0	0	0

I_x

0	0	0	0	0
0	-20	-10	0	0
0	10	30	40	0
0	0	0	0	0
0	0	0	0	0

I_y

Ex: Harris Corner Detector

0	0	0	0	0
0	-20	10	0	0
0	-10	30	0	0
0	0	40	0	0
0	0	0	0	0

I_x

0	0	0	0	0
0	-20	-10	0	0
0	10	30	40	0
0	0	0	0	0
0	0	0	0	0

I_y

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

0	0	0	0	0
0	400	100	0	0
0	100	900	0	0
0	0	1600	0	0
0	0	0	0	0

I_x^2 (square of each element of I_x)

0	0	0	0	0
0	400	100	0	0
0	100	900	1600	0
0	0	0	0	0
0	0	0	0	0

I_y^2(square of each element of I_y)

0	0	0	0	0
0	400	-100	0	0
0	-100	900	0	0
0	0	0	0	0
0	0	0	0	0

I_x I_y (multiplication of each element of I_x and I_y

Ex: Harris Corner Detector

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0	0	0
0	400	100	0	0
0	100	900	0	0
0	0	1600	0	0
0	0	0	0	0

I_x^2

0	0	0	0	0
0	400	100	0	0
0	100	900	1600	0
0	0	0	0	0
0	0	0	0	0

I_y^2

0	0	0	0	0
0	400	-100	0	0
0	-100	900	0	0
0	0	0	0	0
0	0	0	0	0

$I_x I_y$

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

$$M = \begin{bmatrix} 3100 & 1100 \\ 1100 & 3100 \end{bmatrix}$$

$\sum I_x^2 = 3100$

$$\sum I_y^2 = 3100$$

$\sum I_x I_y = 1100$

Determinant of M

= 3100x3100 – 1100x1100

= 9610000 – 12100

=8400000

Trace of M

= 3100 +3100

= 6200

Ex: Harris Corner Detector

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0	0	0
0				0
0		R = 6862400		0
0				0
0	0	0	0	0

$$M = \begin{bmatrix} 3100 & 1100 \\ 1100 & 3100 \end{bmatrix}$$

Determinant of M =8400000

Trace of M = 6200

R = det{M} – k (trace {M})^2, k = 0.04

R = 8400000 – 0.04 (6200)^2
= 6862400

- For threshold = 3000000
- R > threshold
- Therefore, point at the center of window is a corner point

Ex: Harris Corner Detector

0	0	0	0	0
0	0	0	0	0
10	10	10	10	10
0	0	0	0	0
0	0	0	0	0

Image

1	0	-1
2	0	-2
1	0	-1

Sobel mask for I_x

1	2	1
0	0	0
-1	-2	-1

Sobel mask for I_y

Ex: Harris Corner Detector

0	0	0	0	0
0	0	0	0	0
10	10	10	10	10
0	0	0	0	0
0	0	0	0	0

Image

1	0	-1
2	0	-2
1	0	-1

Sobel mask for I_x

1	2	1
0	0	0
-1	-2	-1

Sobel mask for I_y

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

I_x

0	0	0	0	0
0	-40	-40	-40	0
0	0	0	0	0
0	40	40	40	0
0	0	0	0	0

I_y

Ex: Harris Corner Detector

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

I_x

0	0	0	0	0
0	-40	-40	-40	0
0	0	0	0	0
0	40	40	40	0
0	0	0	0	0

I_y

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

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

I_x^2

0	0	0	0	0
0	1600	1600	1600	0
0	0	0	0	0
0	1600	1600	1600	0
0	0	0	0	0

I_y^2

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

$I_x I_y$

Ex: Harris Corner Detector

Image

0	0	0	0	0
0	0	0	0	0
10	10	10	10	10
0	0	0	0	0
0	0	0	0	0

I_x^2

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

I_y^2

0	0	0	0	0
0	1600	1600	1600	0
0	0	0	0	0
0	1600	1600	1600	0
0	0	0	0	0

I_xI_y

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

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

$$M = \begin{bmatrix} 0 & 0 \\ 0 & 9600 \end{bmatrix}$$

$\sum I_x^2$ =0

$$\sum I_y^2 = 9600$$

$\sum I_xI_y$ =0

Determinant of M
= 0x(9600)-0
=0

Trace of M
= 9600 +0
= 9600

Ex: Harris Corner Detector

0	0	0	0	0
0	0	0	0	0
10	10	10	10	10
0	0	0	0	0
0	0	0	0	0

Image

0	0	0	0	0
0				0
0		-3686400		0
0				0
0	0	0	0	0

Corner Response at the center of window, R

$$M = \begin{bmatrix} 0 & 0 \\ 0 & 9600 \end{bmatrix}$$

Determinant of M = 0

Trace of M = 9600

R = det{M} – k (trace {M})^2

Where k is 0.04 – 0.06

R = 0 – 0.04 (9600)^2
= -3686400

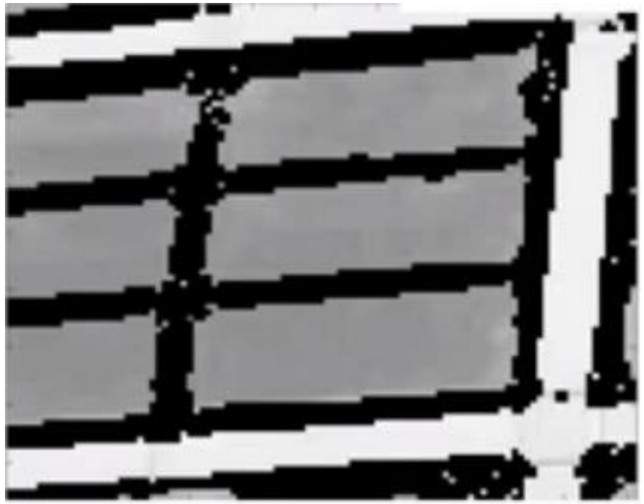
- For threshold = -3000000
- R is negative and < negative threshold
- Therefore, point at the center of window is edge point

Harris Corner Detector

Image

R < -10000 for edges

R > 10000 for corners



For -10,000 < R< 10000
Neither edges nor corners

Choice of Window Function

- By default, uniform window is used
- If uniform window is chosen as the window function, Harris detector is not rotation invariant

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

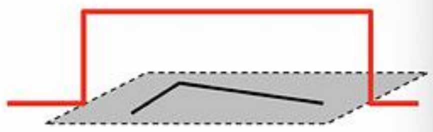
Choice of Window Function

- By default, uniform window is used
- If uniform window is chosen as the window function,
  Harris detector is not rotation invariant

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

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

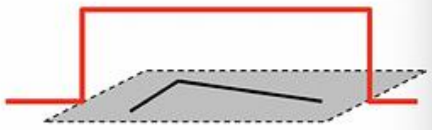
Choice of Window Function

- By default, uniform window is used
- If uniform window is chosen as the window function,
  Harris detector is not rotation invariant

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

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

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

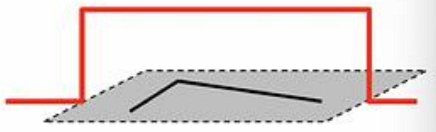
Choice of Window Function

- By default, uniform window is used
- If uniform window is chosen as the window function, Harris detector is not rotation invariant

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

Option 1: uniform window

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

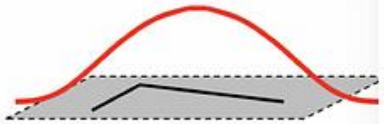


1 in window, 0 outside

Problem: not rotation invariant

Option 2: Smooth with Gaussian

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



Gaussian

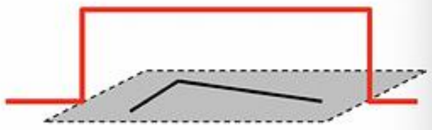
Choice of Window Function

- By default, uniform window is used
- If uniform window is chosen as the window function,
  Harris detector is not rotation invariant

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

Option 1: uniform window

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

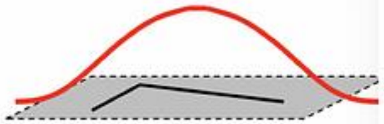


1 in window, 0 outside

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 provides weighted sum
Result is rotation invariant

Multiply I_x^2, I_y^2 and I_xI_y by Gaussian window

Ex: Harris Corner Detector (with Gaussian Window)

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0	0	0
0	400	100	0	0
0	100	900	0	0
0	0	1600	0	0
0	0	0	0	0

I_x^2

0	0	0	0	0
0	400	100	0	0
0	100	900	1600	0
0	0	0	0	0
0	0	0	0	0

I_y^2

0	0	0	0	0
0	400	-100	0	0
0	-100	900	0	0
0	0	0	0	0
0	0	0	0	0

$I_x I_y$

0	0	0	0	0
0	1	2	1	0
0	2	4	2	0
0	1	2	1	0
0	0	0	0	0

Gaussian Window, G

0	0	0	0	0
0	400	200	0	0
0	200	3600	0	0
0	0	3200	0	0
0	0	0	0	0

GI_x^2

0	0	0	0	0
0	400	200	0	0
0	200	3600	3200	0
0	0	0	0	0
0	0	0	0	0

GI_y^2

0	0	0	0	0
0	400	-200	0	0
0	-200	3600	0	0
0	0	0	0	0
0	0	0	0	0

GI_xI_y

Ex: Harris Corner Detector (with Gaussian Window)

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0	0	0
0	400	200	0	0
0	200	3600	0	0
0	0	3200	0	0
0	0	0	0	0

GI_x^2

0	0	0	0	0
0	400	200	0	0
0	200	3600	3200	0
0	0	0	0	0
0	0	0	0	0

GI_y^2

0	0	0	0	0
0	400	-200	0	0
0	-200	3600	0	0
0	0	0	0	0
0	0	0	0	0

GI_xI_y

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

$$\sum GI_x^2 = 400 + 200 + 200 + 3600 + 3200$$

$$\sum GI_y^2 = 400 + 200 + 200 + 3600 + 3200$$

$$\sum GI_x\ I_y = 400 - 200 - 200 + 3600$$

Ex: Harris Corner Detector (with Gaussian Window)

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0	0	0
0	400	200	0	0
0	200	3600	0	0
0	0	3200	0	0
0	0	0	0	0

GI_x^2

0	0	0	0	0
0	400	200	0	0
0	200	3600	3200	0
0	0	0	0	0
0	0	0	0	0

GI_y^2

0	0	0	0	0
0	400	-200	0	0
0	-200	3600	0	0
0	0	0	0	0
0	0	0	0	0

GI_xI_y

$\sum GI_x^2 = 400 + 200 + 200 + 3600 + 3200$

$$\sum GI_y^2 = 400 + 200 + 200 + 3600 + 3200$$

$$\sum GI_x\ I_y = 400 - 200 - 200 + 3600$$

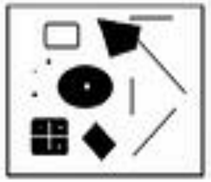
$$M = \begin{bmatrix} GI_x^2 & GI_xI_y \\ GI_xI_y & GI_y^2 \end{bmatrix}$$

R = det{M} – k (trace {M})^2

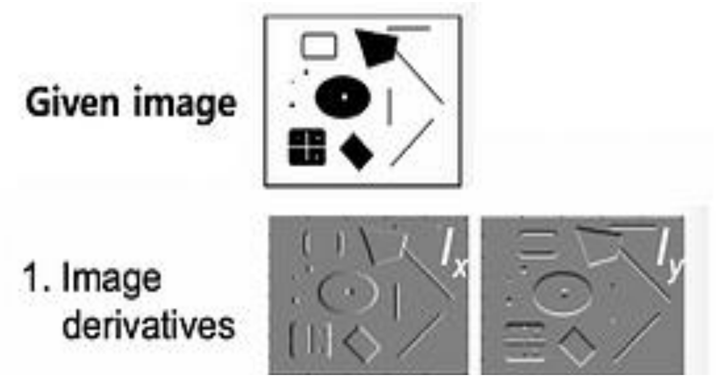
R =

Steps of Harris Detector

Given image



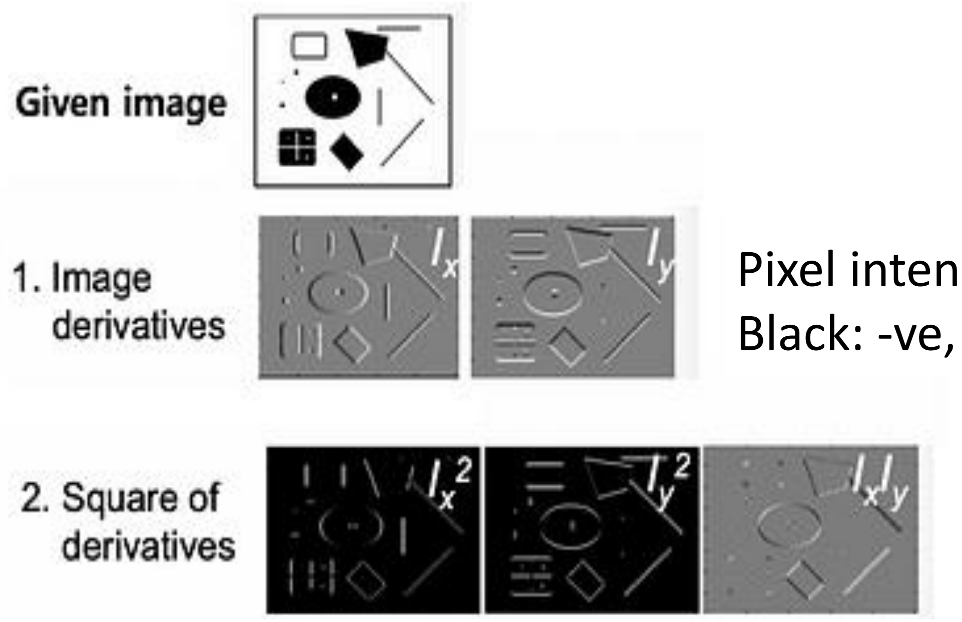
Steps of Harris Detector



Pixel intensity

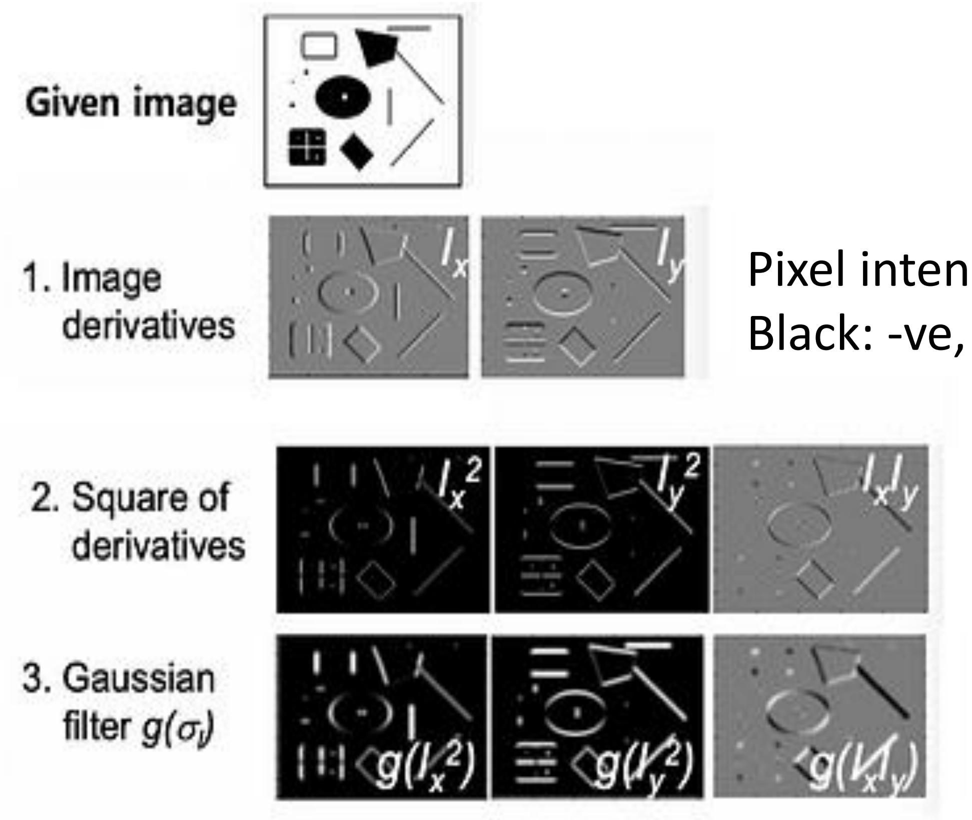
Black: -ve, Grey: 0, White: +ve

Steps of Harris Detector



Pixel intensity
Black: -ve, Grey: 0, White: +ve

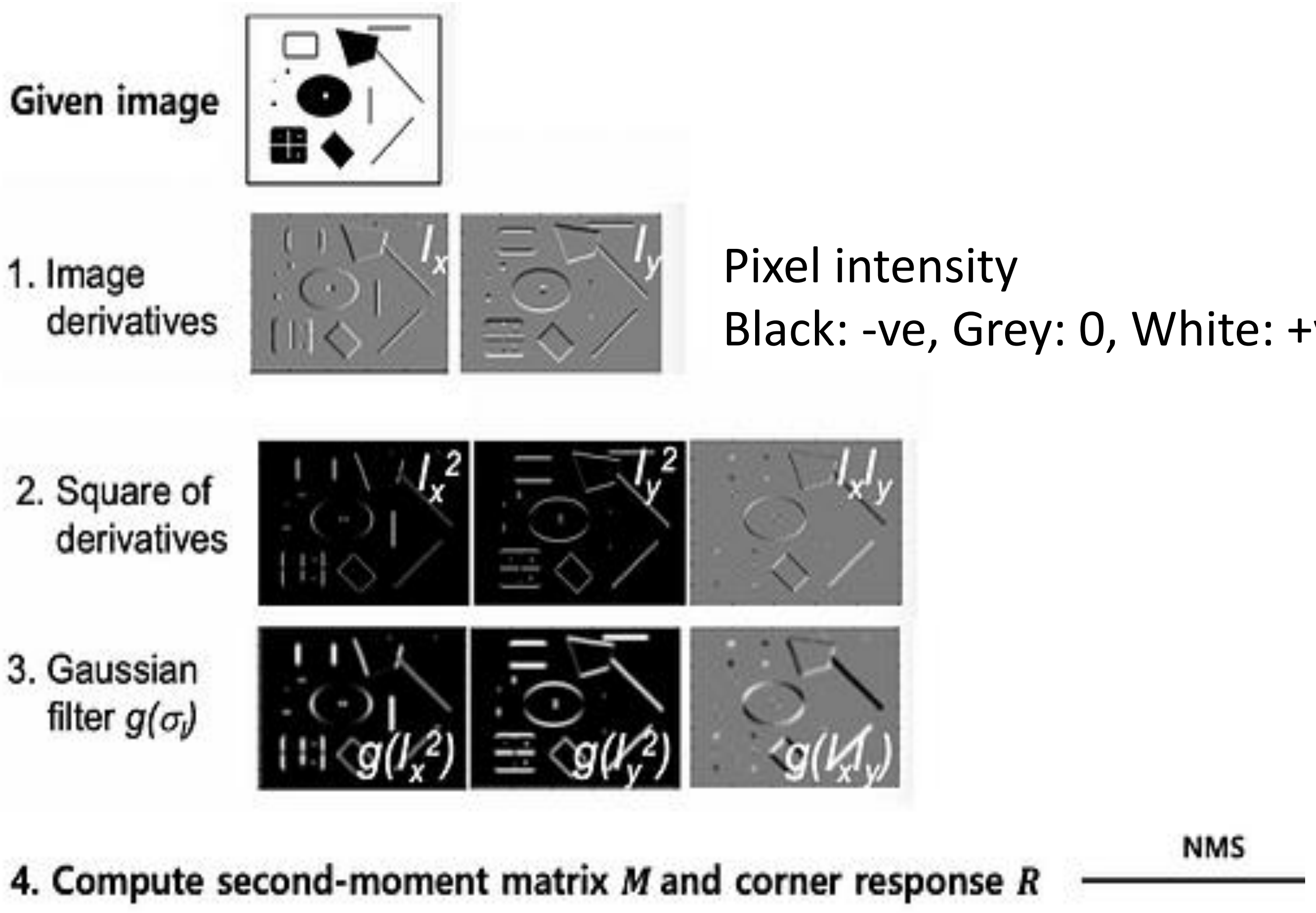
Steps of Harris Detector



Pixel intensity

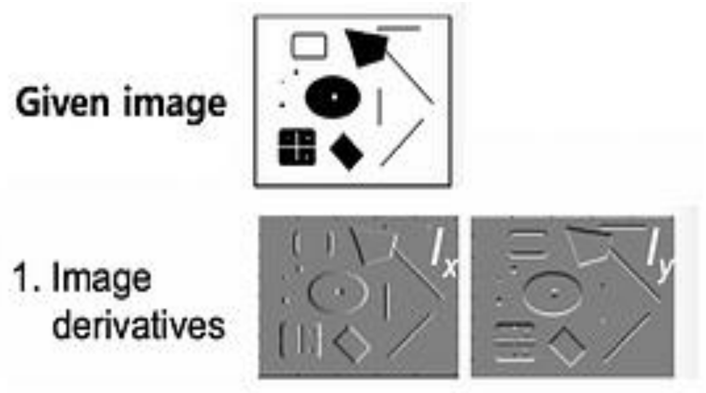
Black: -ve, Grey: 0, White: +ve

Steps of Harris Detector

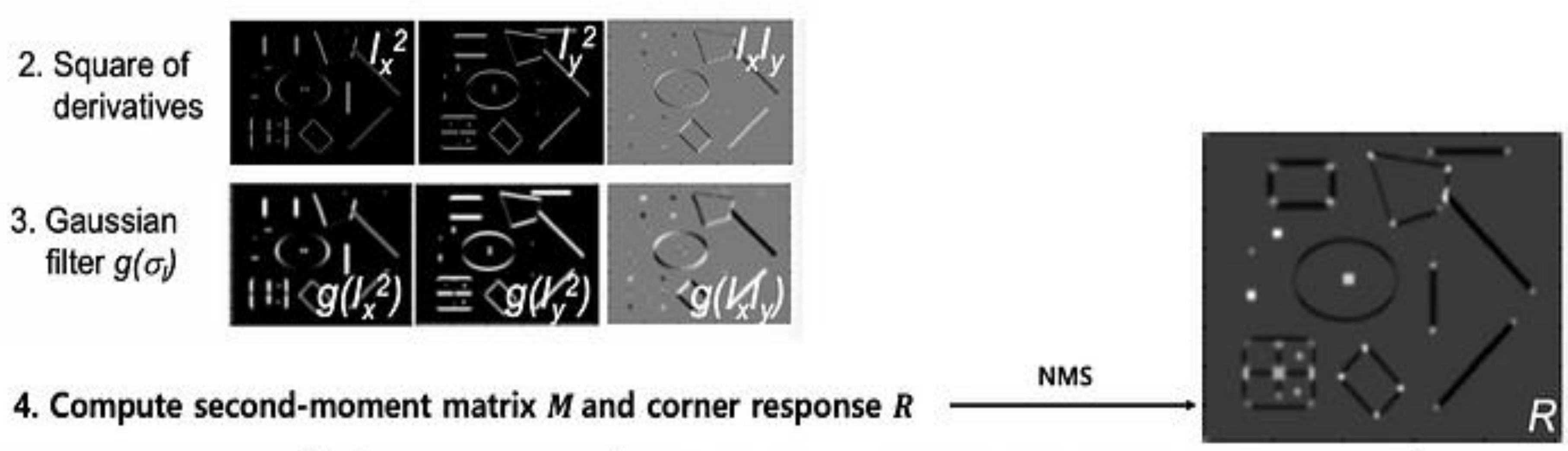


Pixel intensity
Black: -ve, Grey: 0, White: +ve

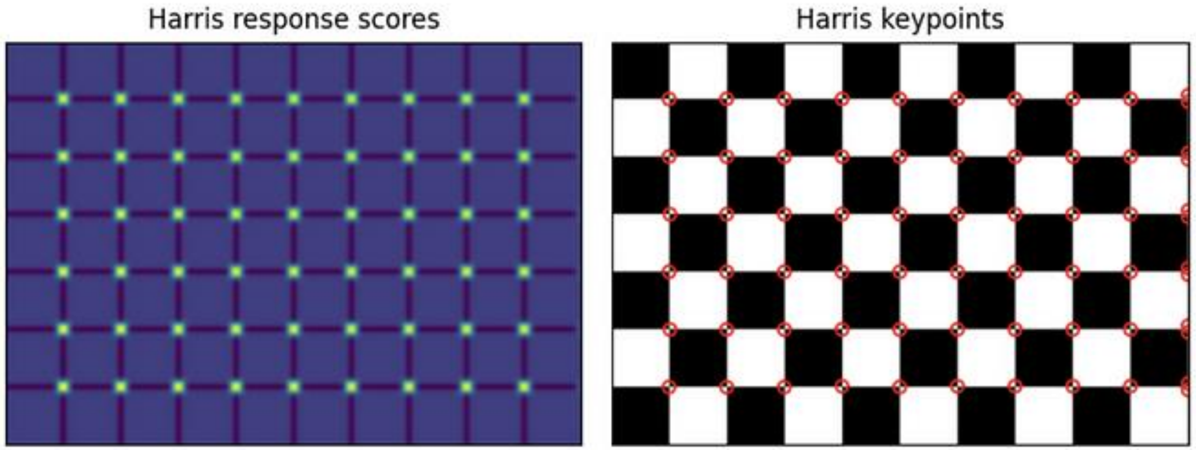
Steps of Harris Detector



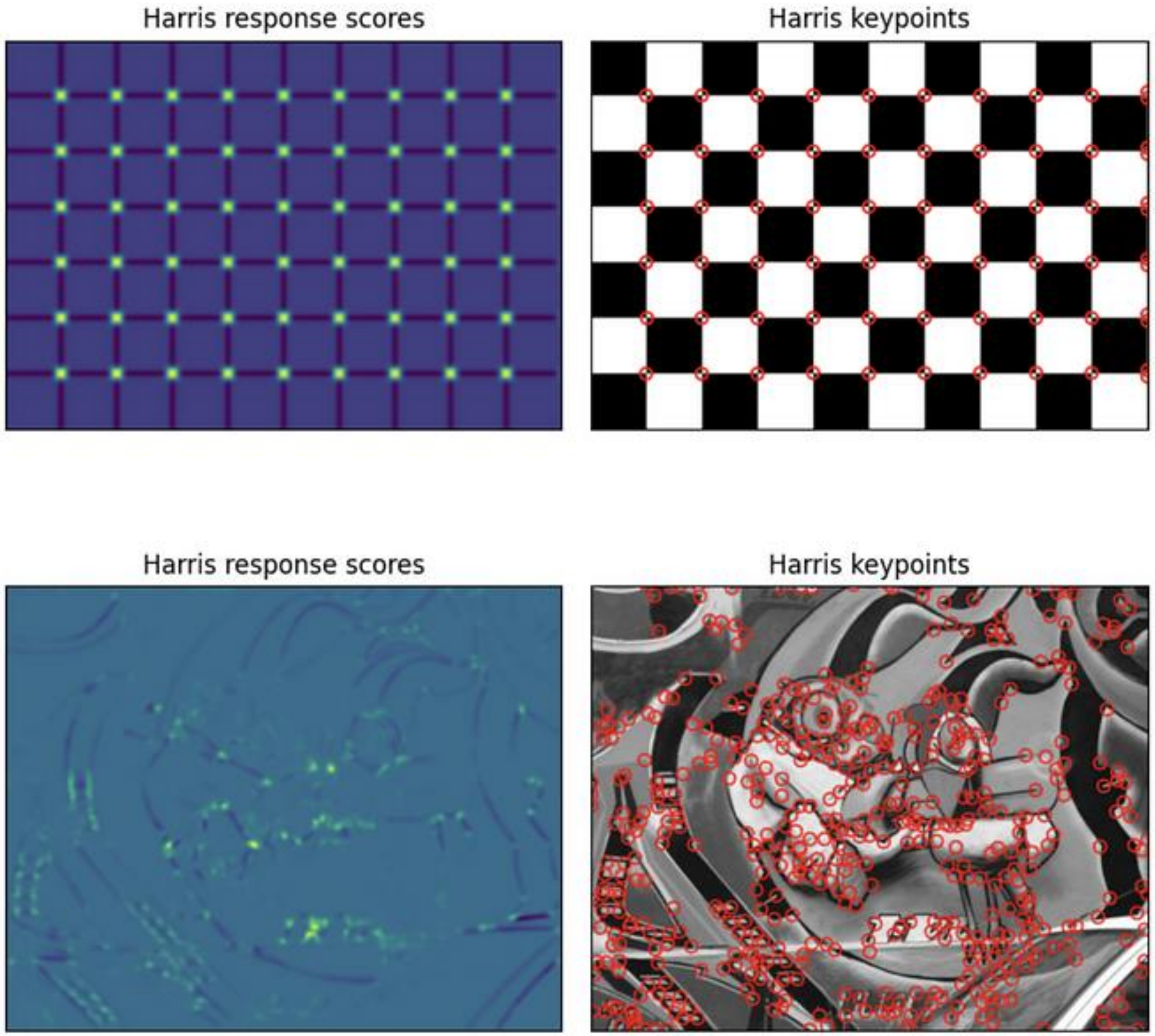
Pixel intensity
Black: -ve, Grey: 0, White: +ve



Harris Corner Detector



Harris Corner Detector



Harris Corner Detector

Image 1

Harris Corner Detector

Image after rotation and variation in illumination

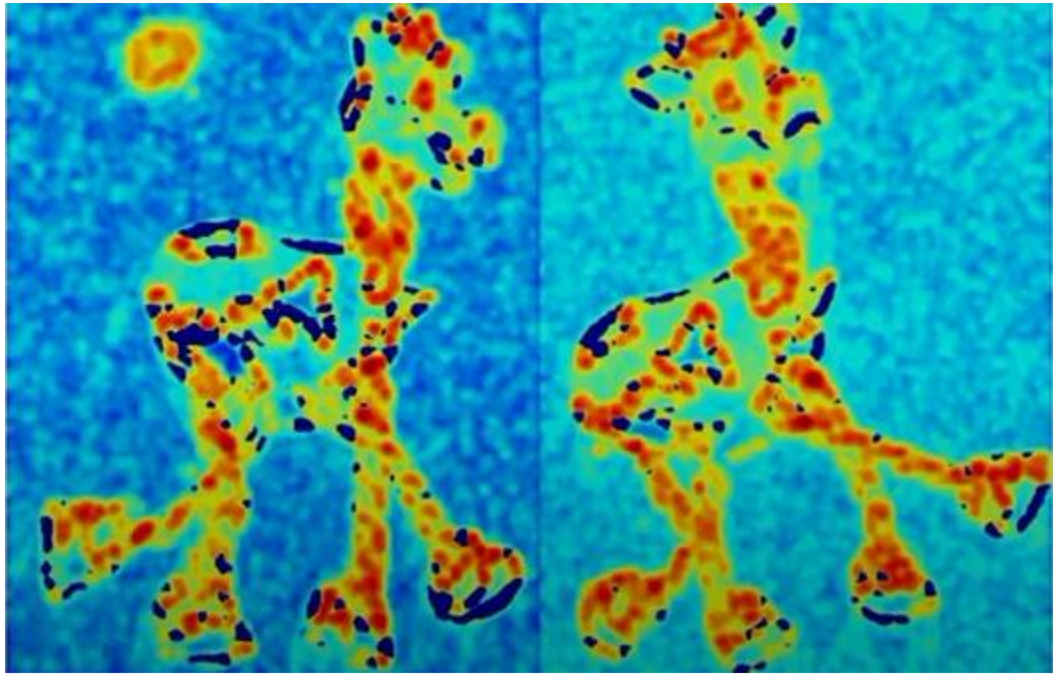
Image 1

Image 2

Harris Corner Detector

Image

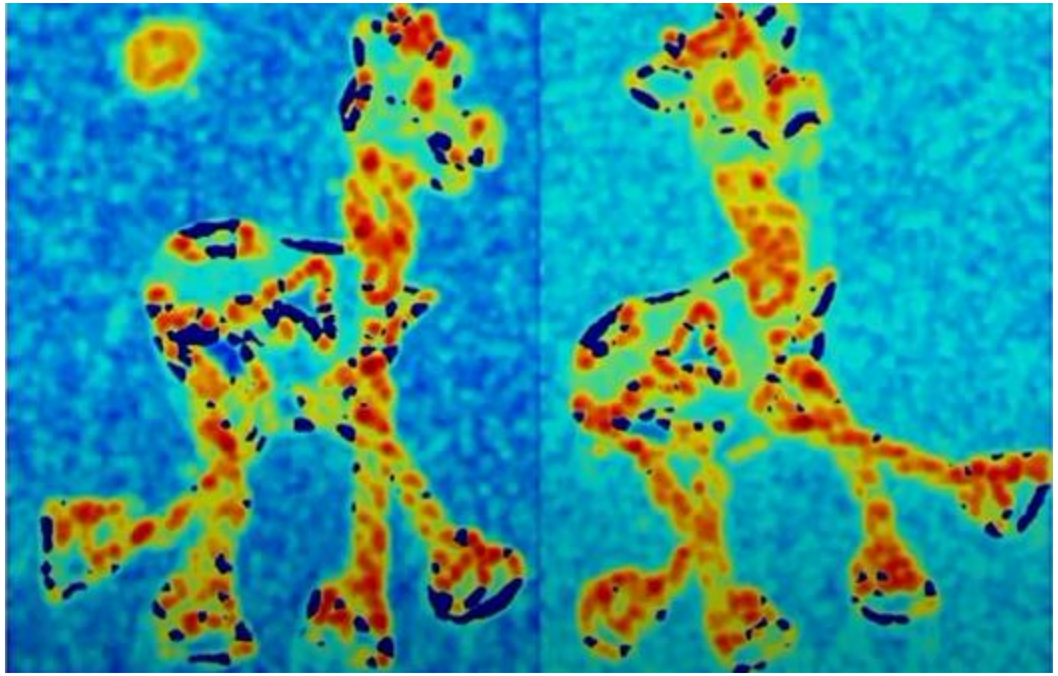
Image after rotation and variation in illumination



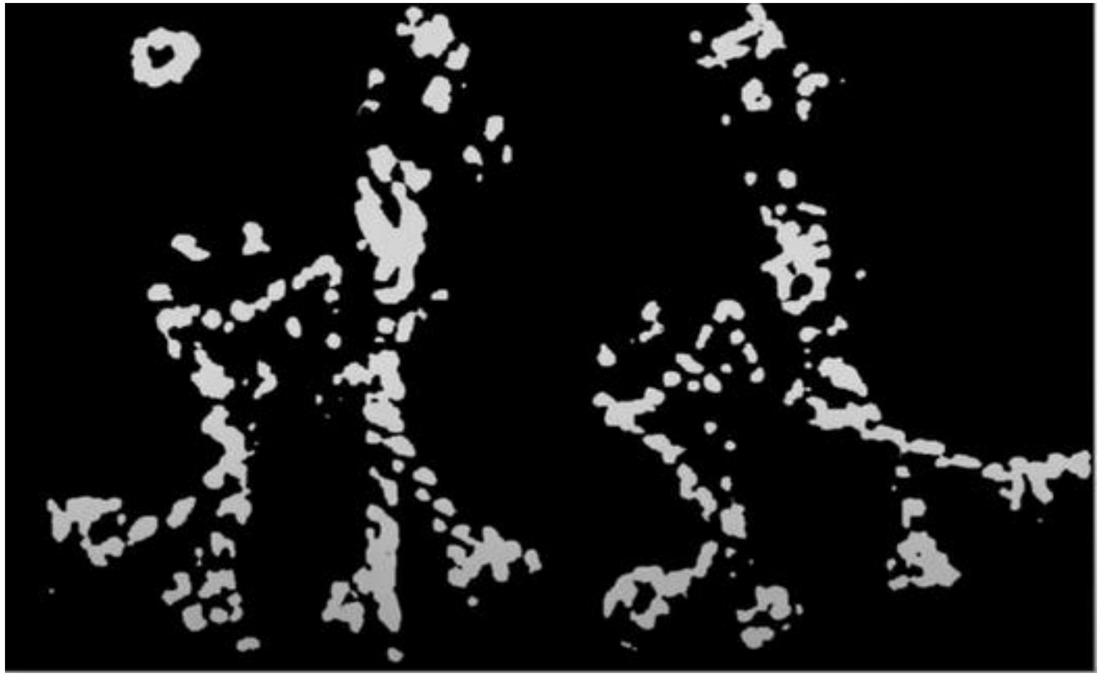
Corner response, R

Blue for low or negative R
Red for positive R

Harris Corner Detector

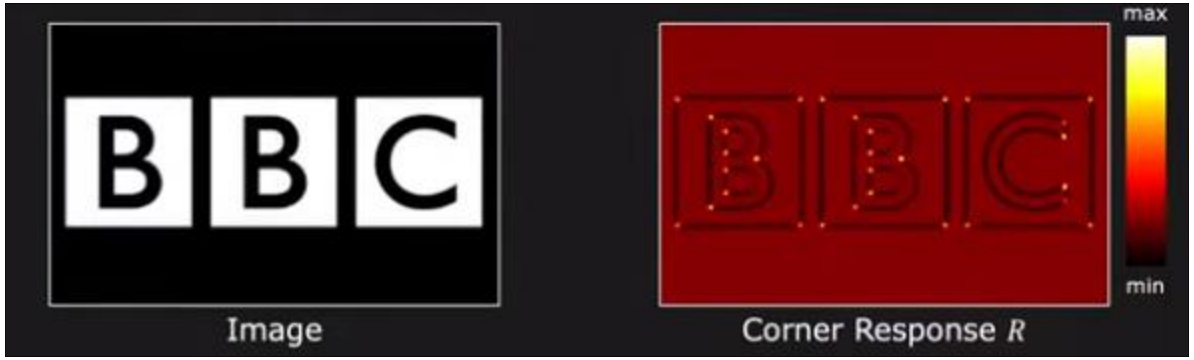


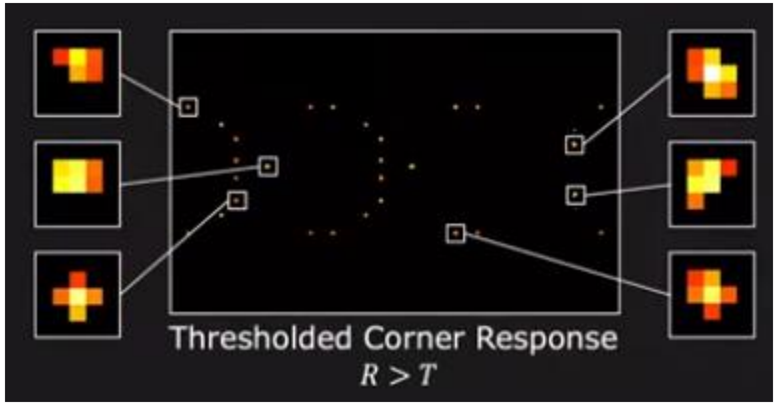
Corner response, R



Points with large value of R (R> threshold)

Harris Corner Detector





- After thresholding on R, we get clusters of corner points not a single point

Non Maximal Suppression

- Slide a window of size k over the cluster
- At each position, if pixel at the center of window is maximum, label it as a corner and retain it
- Else label it as not corner and suppress it

White pixel represents maximum value of R

Non Maximal Suppression

- Slide a window of size k over the cluster
- At each position, if pixel at the center of window is maximum, label it as a corner and retain it
- Else label it as not corner and suppress it

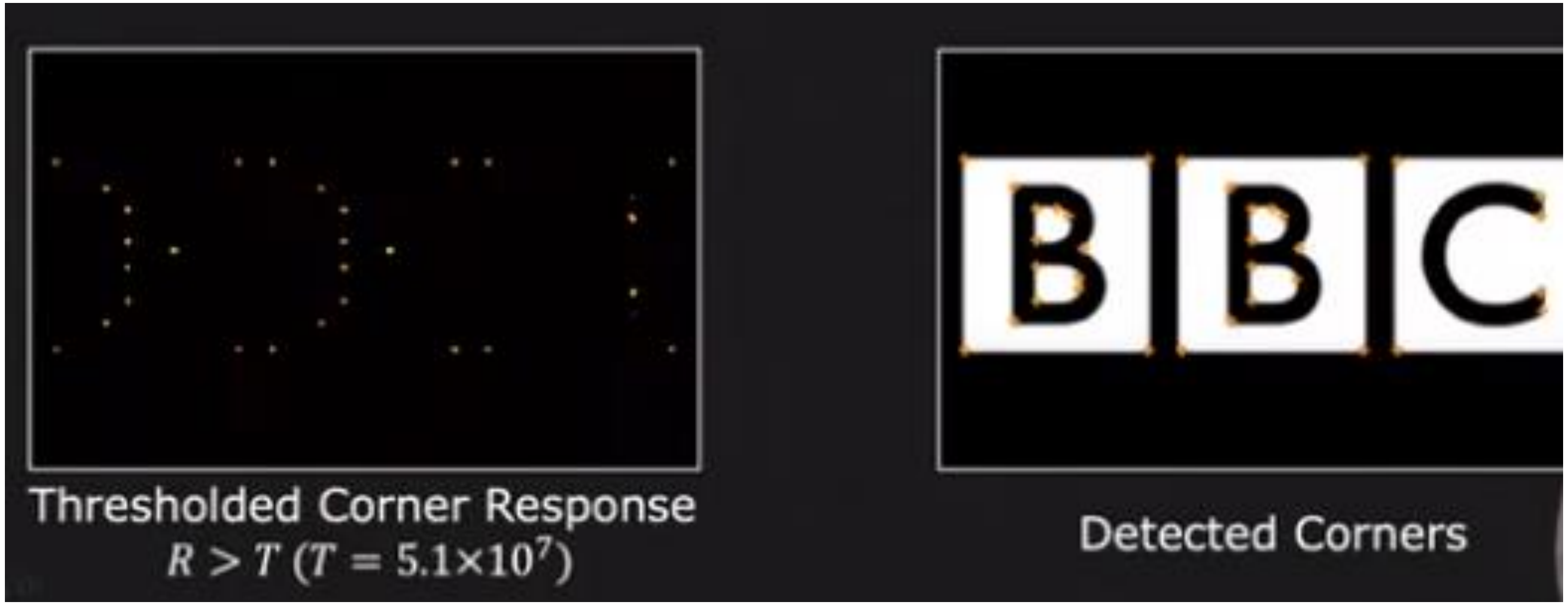
White pixel represents maximum value of R

Non Maximal Suppression

- Slide a window of size k over the cluster
- At each position, if pixel at the center of window is maximum, label it as a corner and retain it
- Else label it as not corner and suppress it

White pixel represents maximum value of R

Harris Corner Detector



Hessian Corner Detector

- Harris and Hessian detectors are similar
- Herris detector uses first moment and corner response function *R* for corner detection
- Hessian detector uses Second moment and determinant to detect corners
- Hessian detector provides good performance in terms of computation time and accuracy
- Given a pixel, the Hessian of pixel is

$$H = \sum W(x, y) \begin{bmatrix} \mathrm{I_{xx}} & \mathrm{I_{xy}} \\ \mathrm{I_{xy}} & \mathrm{I_{yy}} \end{bmatrix}$$

- I_{xx} and I_{yy} are second order derivative in x and y direction respectively
- I_{xy} is first order derivative in x direction (I_x) and then derivative of I_x in y direction

Det(H) = $I_{xx}I_{yy} - (I_{xy})^2$
If det(H) is high then it is a corner point

Ex: Hessian Corner Detector

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0
1	-2	1
0	0	0

Mask for Second order derivative, I_{xx} in x direction

0	1	0
0	-2	0
0	1	0

Mask for Second order derivative, I_{yy} in y direction

1	0	-1
2	0	-2
1	0	-1

Mask for first order derivative, I_x in x direction

1	2	1
0	0	0
-1	-2	-1

Mask for first order derivative of I_x in y direction that is I_{xy}

Ex: Hessian Corner Detector

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0
1	-2	1
0	0	0

Mask for Second order derivative, I_{xx} in x direction

0	1	0
0	-2	0
0	1	0

Mask for Second order derivative, I_{yy} in y direction

1	0	-1
2	0	-2
1	0	-1

Mask for first order derivative, I_x in x direction

1	2	1
0	0	0
-1	-2	-1

Mask for first order derivative of I_x in y direction that is I_{xy}

0	0	0	0	0
0	-10	0	0	10
0	-20	10	0	0
0	-20	10	0	0
0	10	0	0	0

I_{xx}

0	0	0	0	0
0	-10	-20	-20	10
0	0	10	10	0
0	0	0	0	0
0	10	0	0	0

I_{yy}

0	0	0	0	0
0	-20	10	0	0
0	-10	30	0	0
0	0	40	0	0
0	0	0	0	0

I_x

0	0	0	0	0
0	-50	-50	-30	10
0	-70	-80	-30	0
0	10	50	30	0
0	10	0	0	0

I_{xy}

Ex: Hessian Corner Detector

0	0	0	0	0
0	10	10	10	10
0	10	0	0	0
0	10	0	0	0
0	10	0	0	0

Image

0	0	0	0	0
0	-10	0	0	0
0	-20	10	0	0
0	-20	10	0	0
0	0	0	0	0

I_{xx}

0	0	0	0	0
0	-10	-20	-20	0
0	0	10	10	0
0	0	0	0	0
0	0	0	0	0

I_{yy}

0	0	0	0	0
0	-50	-50	-30	10
0	-70	-80	-30	0
0	10	50	30	0
0	10	0	0	0

I_{xy}

$$\text{Hessian, H} = \sum \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}$$

$$\text{Hessian, H} = \begin{bmatrix} -30 & -220 \\ -220 & -30 \end{bmatrix}$$

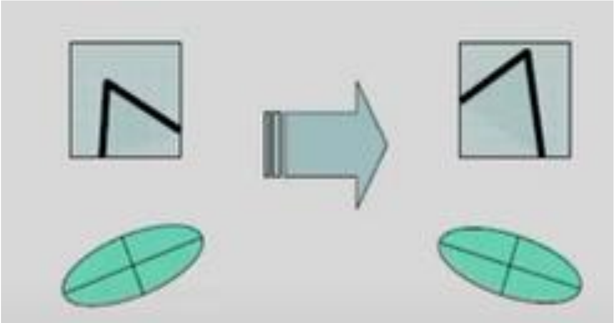
|Det(H)| = |900-48400|

= 47500

- For threshold =15000
- It is a corner point

Harris and Hessian Corner Detectors

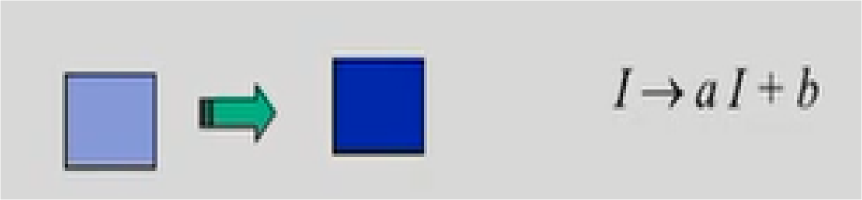
- Corner points should be detected at corresponding locations in other image even if image is rotated
- Harris and Hessian detectors are rotation invariant

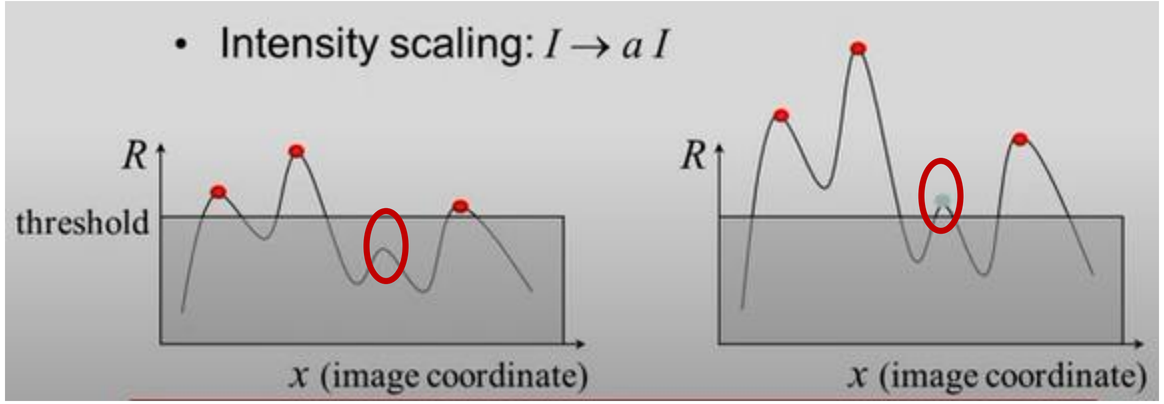


- Ellipse rotates but shape (Eigen values) remains the same
- Also, corner points should be invariant to photo metric transformation and geometric transformation

Harris and Hessian Corner Detectors

- Photometric/ intensity transformation

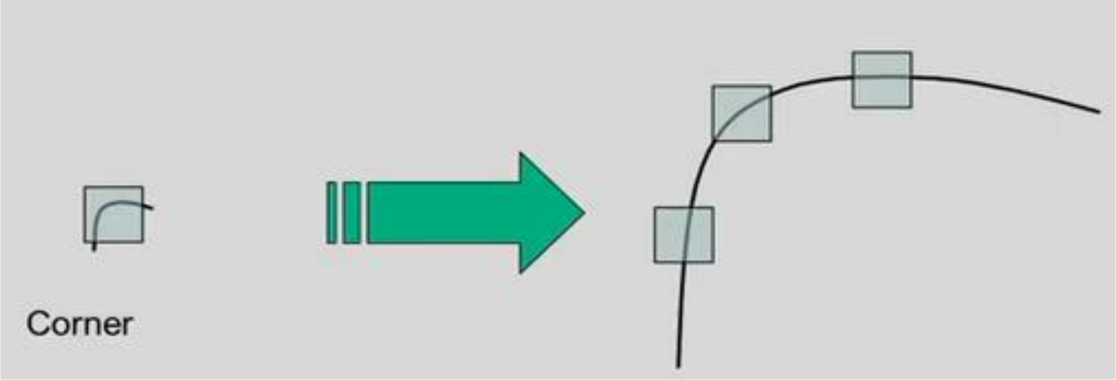




- For Hessian Detector,
  - Intensity shift (I→ I + b) does not change derivative
  - Therefore R is invariant to shift in intensity
  - If I is scaled (I → a×I)
  - Then false point appears after scaling
- Harris detector
  - partially invariant to affine intensity change if intensity change is minor

Harris and Hessian Corner Detectors

- Image scaling in size



- Corner gets magnified and becomes bigger than the size of the window by zooming
- After scaling, corner points are classified as edges
- Harris and Hessian can not detect the corner if image is up scaled
- They are not invariant to scaling

Applications of Interest points

- Image alignment
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval

References

- https://nptel.ac.in/courses/108103174
- https://sbme-tutorials.github.io/2018/cv/notes/6_week6.html
- https://medium.com/data-breach/introduction-to-harris-corner-detector-32a88850b3f6
- https://www.baeldung.com/cs/harris-corner-detection
- https://www.codingninjas.com/codestudio/library/harris-corner-detection
- https://www.google.com/url?sa=t&source=web&rct=j&url=https://www.cs.umd.edu/class/fall2019/cmsc426-0201/files/12_HarrisCornerDetection.pdf&ved=2ahUKEwj_q4fY5br-AhWu-jgGHeTBCJAQFnoECD8QAQ&usg=AOvVaw0WjY5eRFeu-vCUFu-g6o90
- https://fiveko.com/feature-points-using-harris-corner-detector/