中山大學
SUN YAT-SEN UNIVERSITY

Lecture 4:
Features and Fitting

Pattern Recognition (in Computer Vision)

Jinhua Ma,
School of Computer Science and Engineering, Sun Yat-Sen University

What we will learn today?

- A model fitting method for line detection
  - RANSAC
- Local invariant features
  - Motivation
  - Requirements, invariances
- Keypoint localization
  - Harris corner detector

What we will learn today?

- A model fitting method for line detection
  - RANSAC
- Local invariant features
  - Motivation
  - Requirements, invariances
- Keypoint localization
  - Harris corner detector

Fitting as Search in Parametric Space

- **Let's say we have chosen a parametric model for a set of features:**

  – For example, we have a line equation that we want to fit to a set of edge points

- **We can 'search' in parameter space by trying many potential parameter values and see which set of parameters 'agree'/fit with our set of features**

- **Three main questions:**

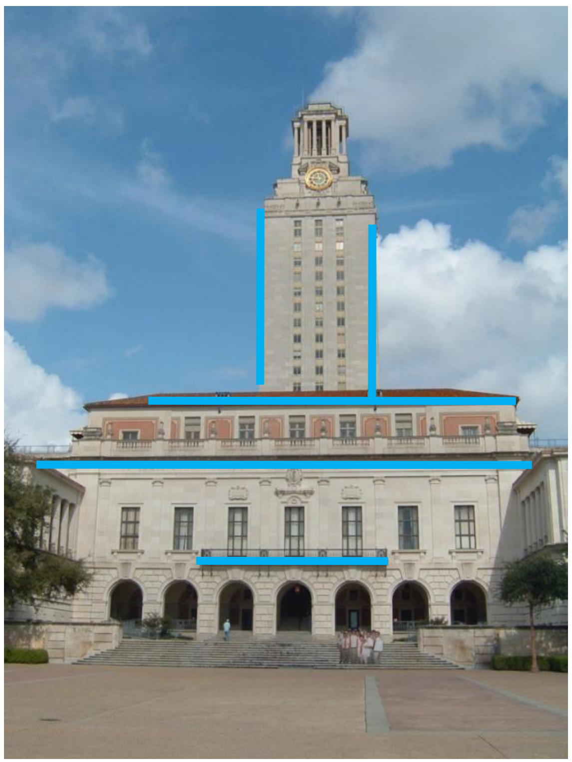
1. What model represents this set of features best?
2. Which of several model instances gets which feature?
3. How many model instances are there?

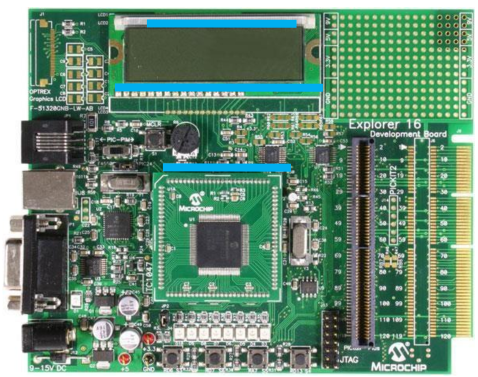
- **Computational complexity is important**

  – It is infeasible to examine every possible set of parameters and every possible combination of features

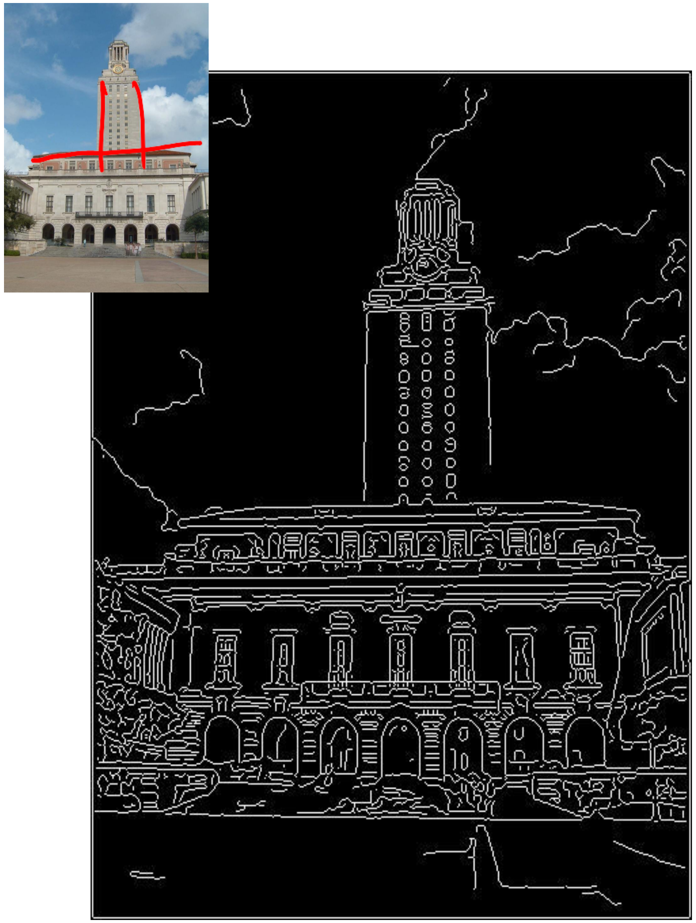
Example: Line Fitting

- Why fit lines? Many objects characterized by presence of straight lines:



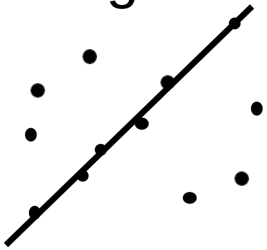


Difficulty of Line Fitting



- **Extra edge points (clutter), multiple models:**

  – Which points go with which line, if any?



- **Only some parts of each line detected, and some parts are missing:**

  – How to find a line that bridges missing evidence?

- **Noise in measured edge points, orientations:**

  – How to detect true underlying parameters?

Voting as a fitting technique

- **It's not feasible to check all combinations of features by fitting a model to each possible subset. For example, the naïve line fitting between every pair of two points is $O(N^2)$.**
- **Voting is a general technique where we let the features vote for all models that are compatible with it.**
  – Cycle through features, cast votes for model parameters.
  – Look for model parameters that receive a lot of votes.
- **Noise & clutter features will cast votes too, but typically their votes should be inconsistent with the majority of "good" features.**
- **Ok if some features not observed, as model can span multiple fragments.**

RANSAC (RANdom SAmple Consensus)

- **RANSAC [Fischler & Bolles 1981]**

  – **Approach: we want to avoid the impact of outliers, so let's look for "inliers", and use only those.**

- **Intuition: if an outlier is chosen to compute the current fit, then the resulting line won't have much support from rest of the points.**

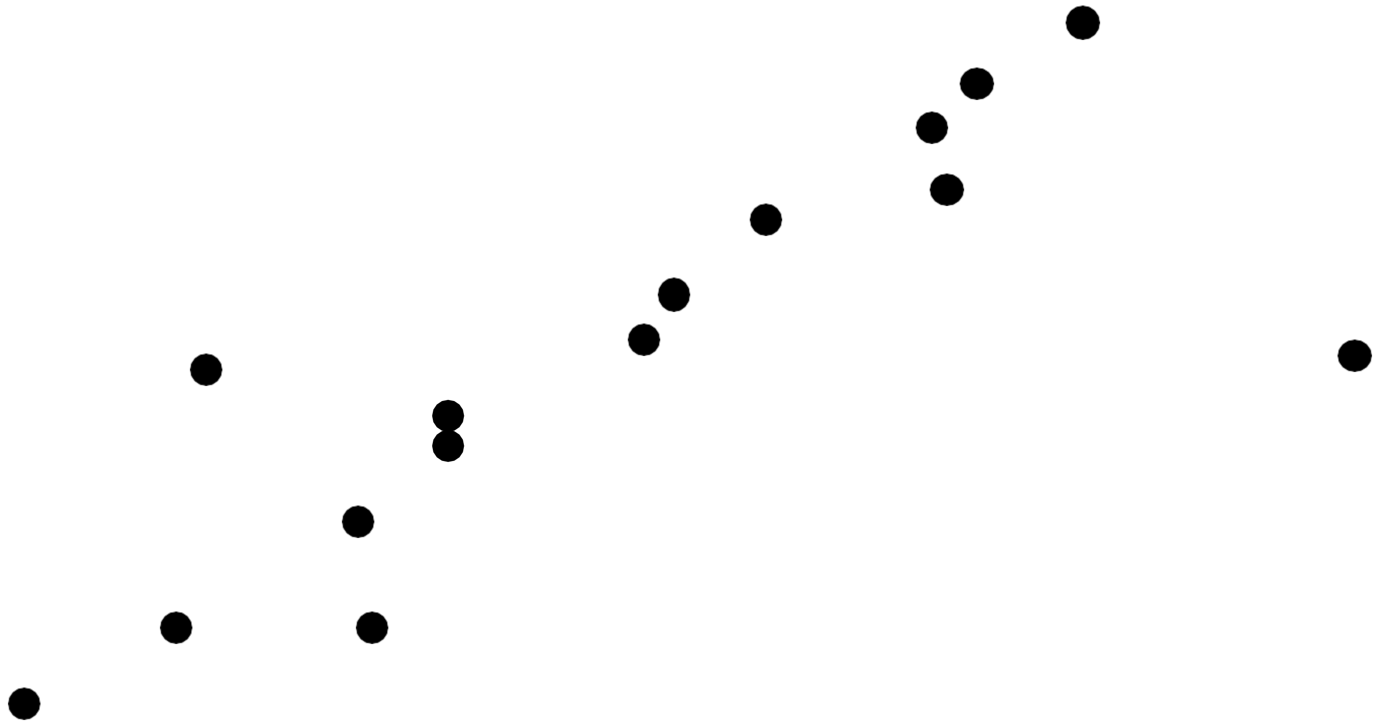
- **RANSAC loop:**

1. Randomly select a seed group of points on which to perform a model estimate (e.g., a group of good points)
2. Compute model parameters from seed group
3. Find inliers to this model
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of model on all of the inliers

   – Keep the model with the largest number of inliers

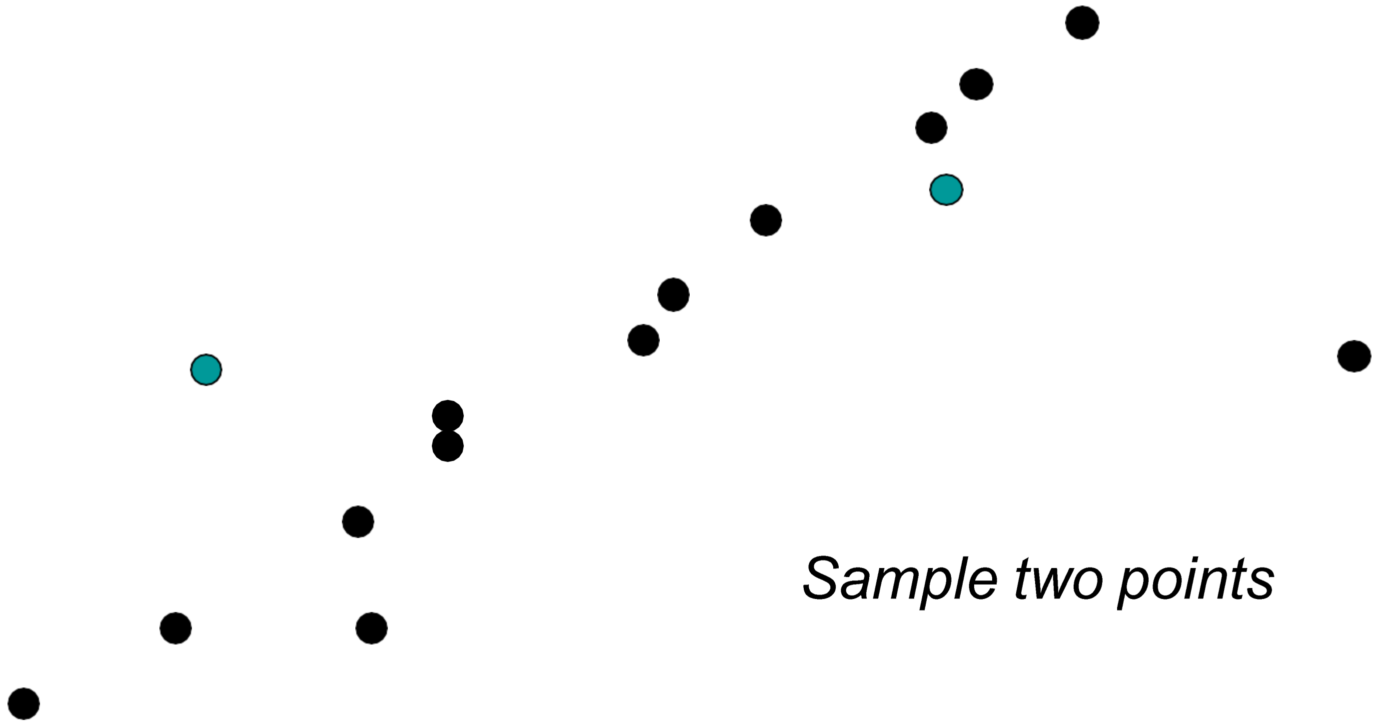
RANSAC Line Fitting Example

- **Task: Estimate the best line**
  - How many points do we need to estimate the line?



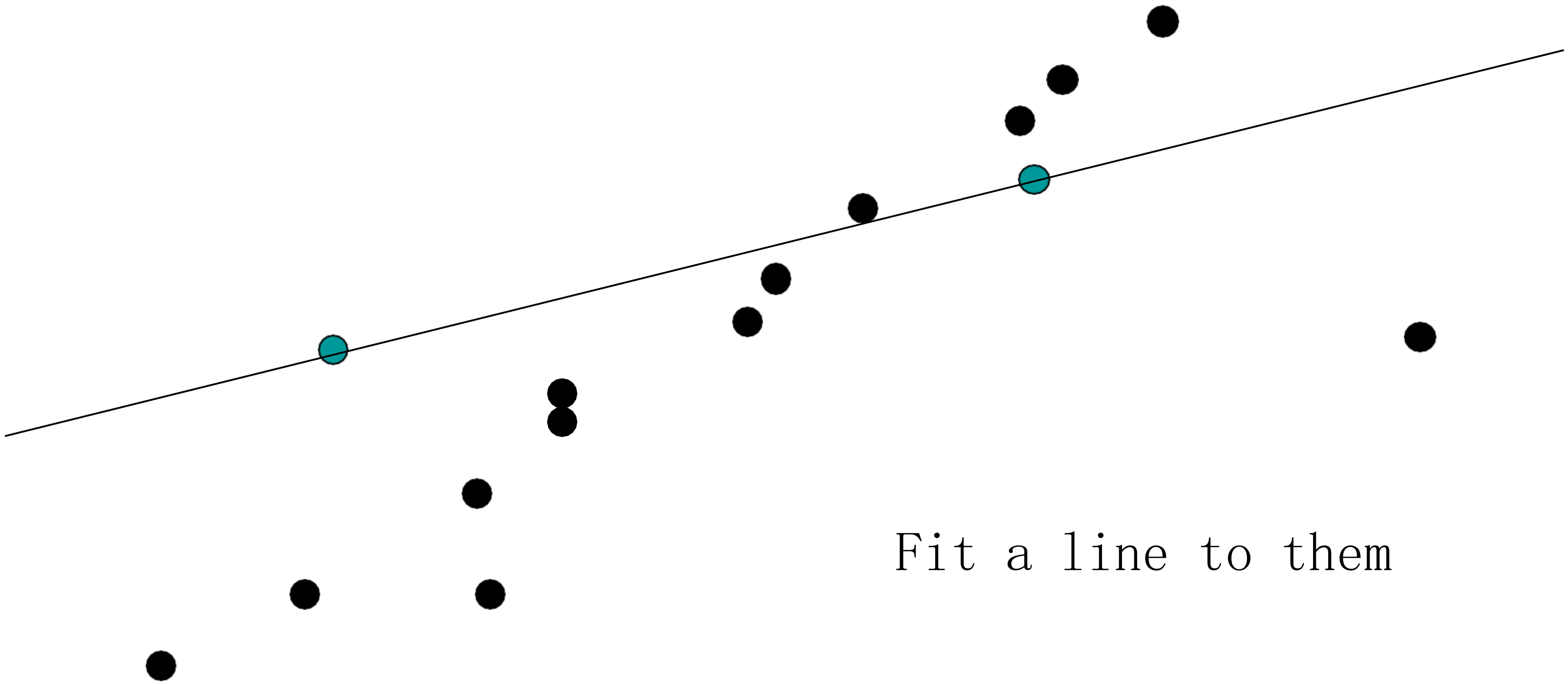
RANSAC Line Fitting Example

- **Task: Estimate the best line**



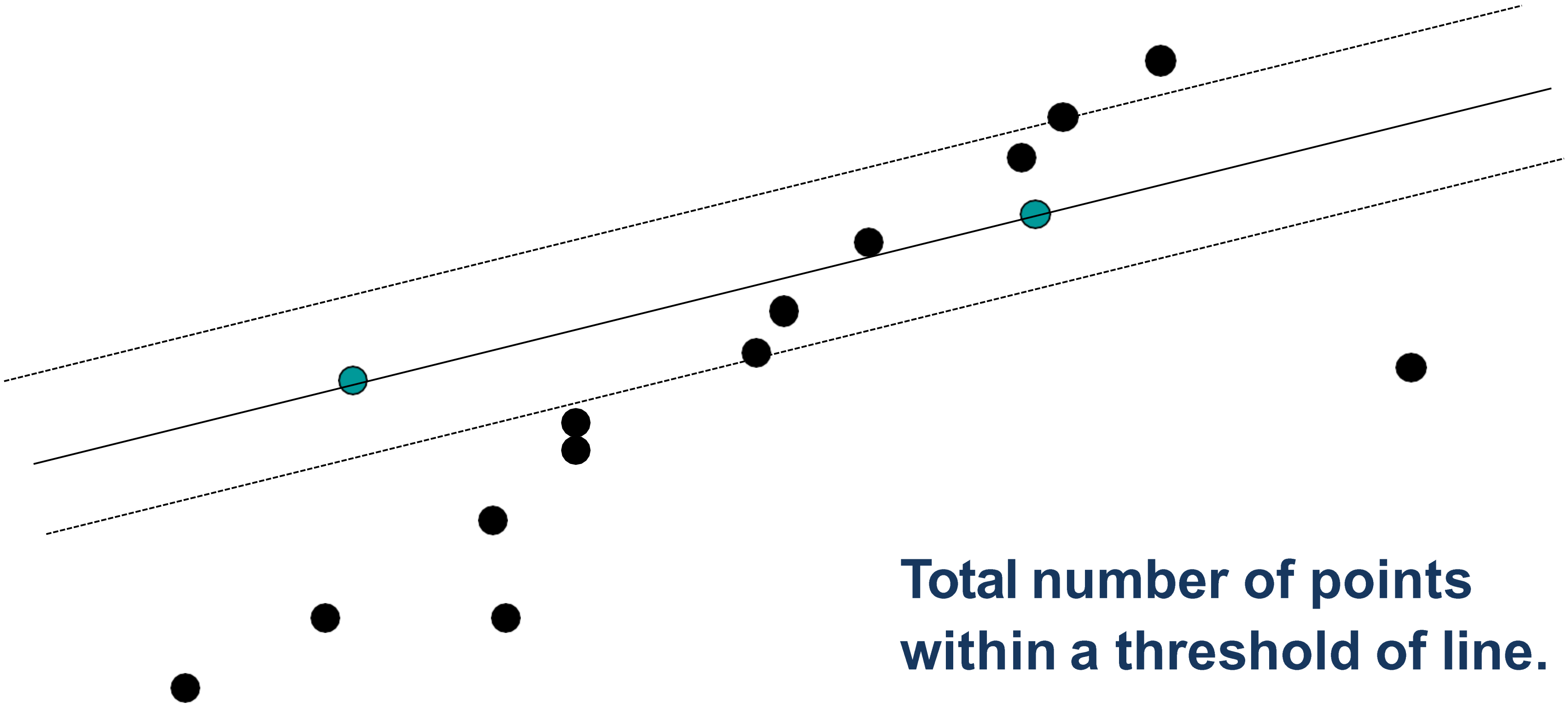
RANSAC Line Fitting Example

- **Task: Estimate the best line**



RANSAC Line Fitting Example

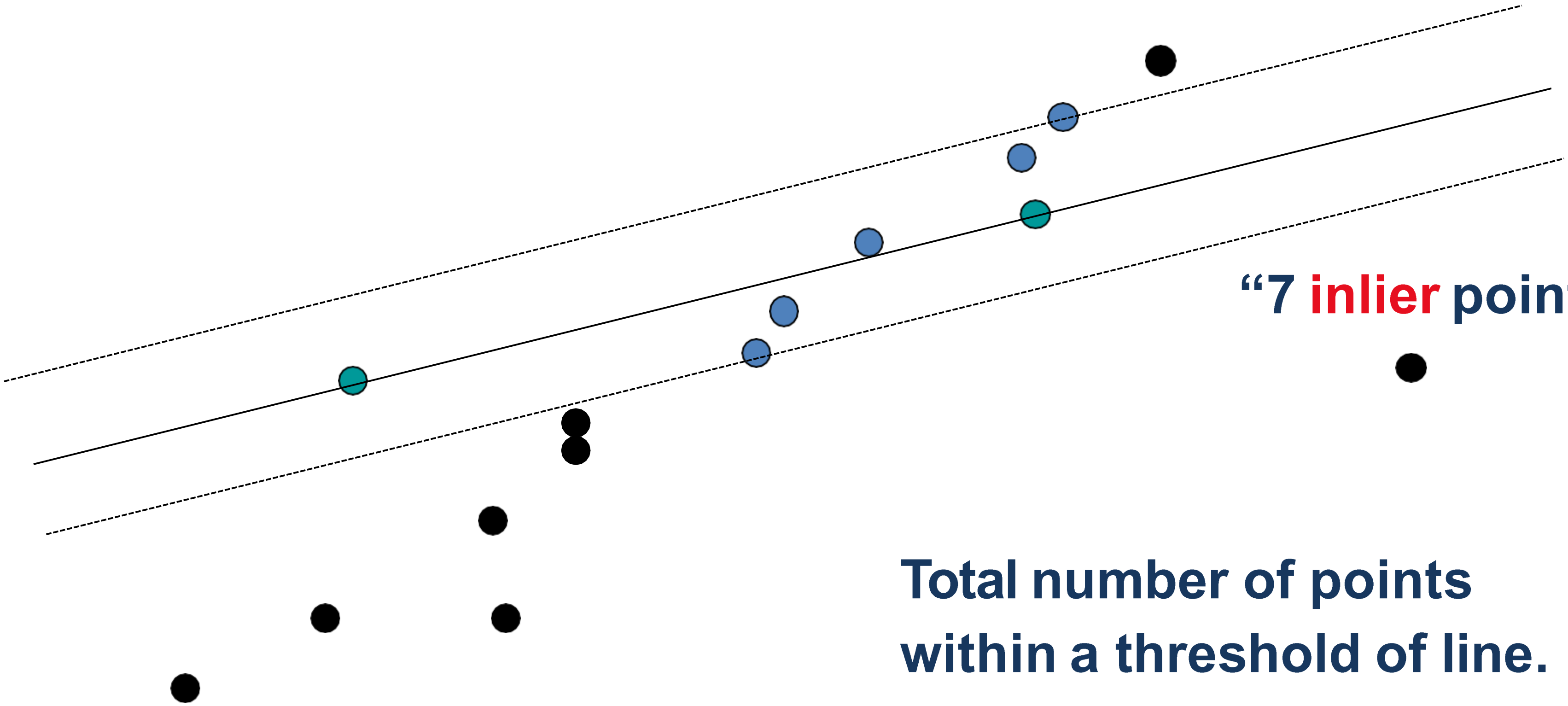
- **Task: Estimate the best line**



Total number of points within a threshold of line.

RANSAC Line Fitting Example

- **Task: Estimate the best line**

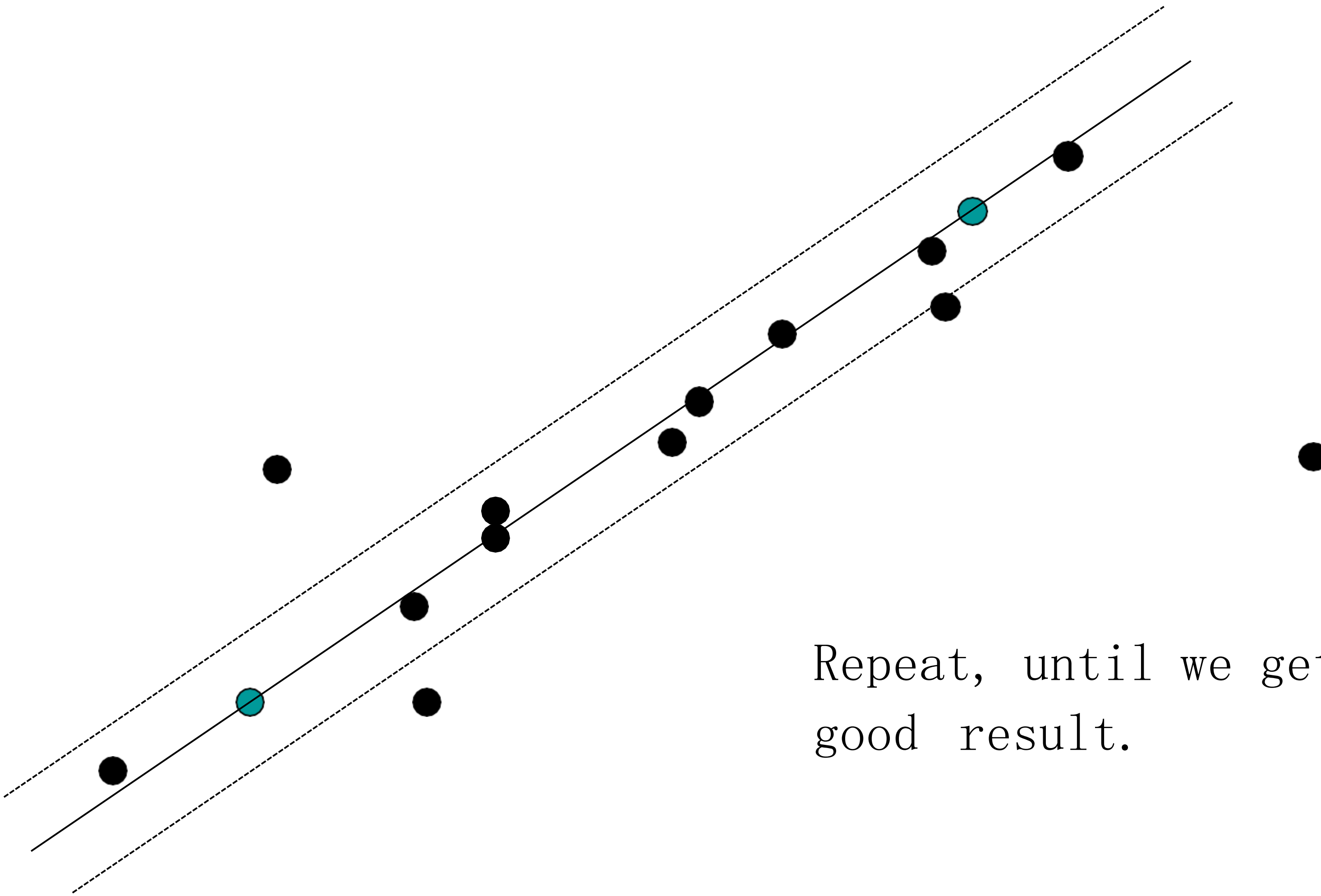


"7 inlier points"

Total number of points within a threshold of line.

RANSAC Line Fitting Example

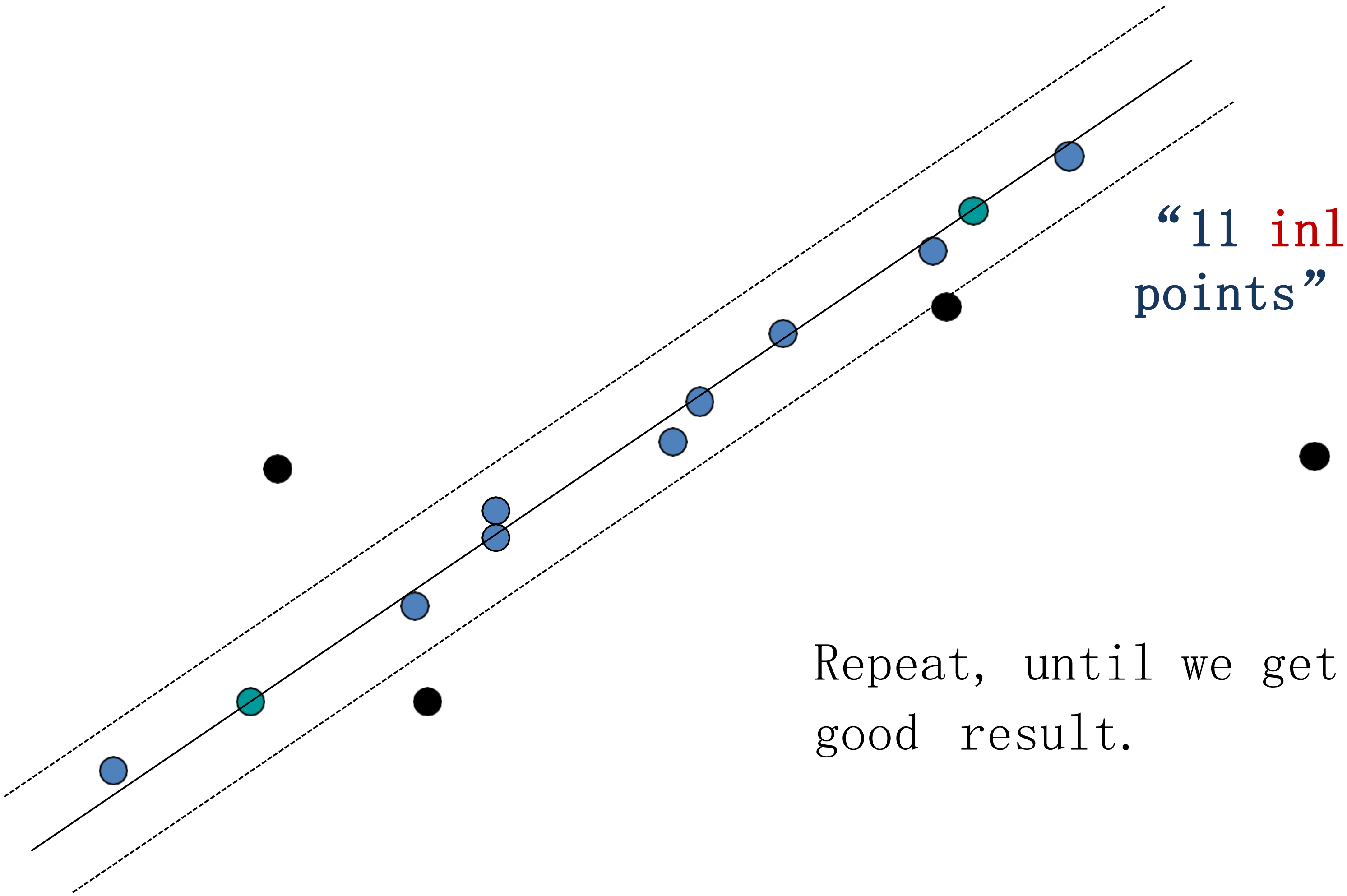
- **Task: Estimate the best line**



Repeat, until we get a good result.

RANSAC Line Fitting Example

- **Task: Estimate the best line**



"11 inlier points"

Repeat, until we get a good result.

Algorithm of RANSAC

Algorithm 15.4: RANSAC: fitting lines using random sample consensus

Determine:
  n — the smallest number of points required
  k — the number of iterations required
  t — the threshold used to identify a point that fits well
  d — the number of nearby points required
    to assert a model fits well
Until k iterations have occurred
  Draw a sample of n points from the data
    uniformly and at random
  Fit to that set of n points
  For each data point outside the sample
    Test the distance from the point to the line
      against t; if the distance from the point to the line
      is less than t, the point is close
  end
  If there are d or more points close to the line
    then there is a good fit. Refit the line using all
    these points.
end
Use the best fit from this collection, using the
  fitting error as a criterion

RANSAC: How many iterations "k"?

- **How many samples (iterations) are needed?**
  - Suppose w is fraction of inliers (points from line).
  - n points needed to define hypothesis (2 for lines)
  - k samples chosen.
- **Prob. that a single sample of n points is correct:** $\mathrm{w}^{\boldsymbol{n}}$
- **Prob. that a single sample of n points fails:** $\mathbf{1} - \mathrm{w}^{\boldsymbol{n}}$
- **Prob. that all k samples fail is:** $(\mathbf{1} - \mathrm{w}^{\boldsymbol{n}})^{\boldsymbol{k}}$
- **Prob. that at least one of the k samples is correct:**

$\mathbf{1} - (\mathbf{1} - \mathrm{w}^{\boldsymbol{n}})^{\boldsymbol{k}}$

- $\Rightarrow$ Choose k high enough to keep this below desired failure rate.

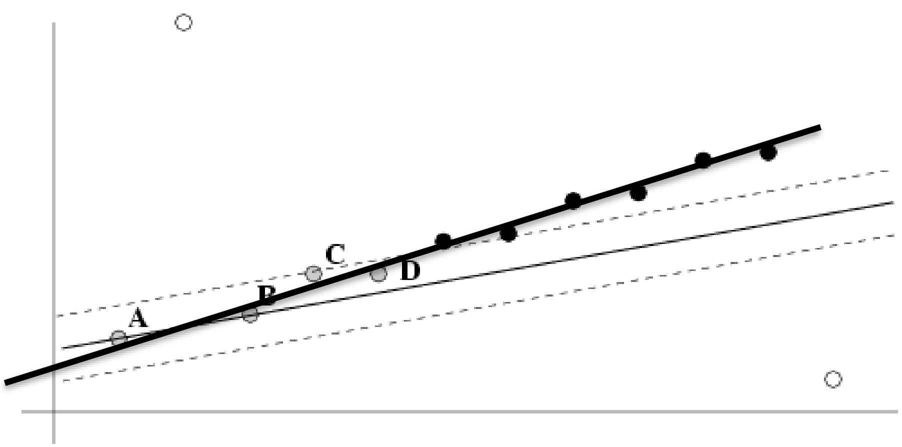
RANSAC: Computed k (p=0.99)

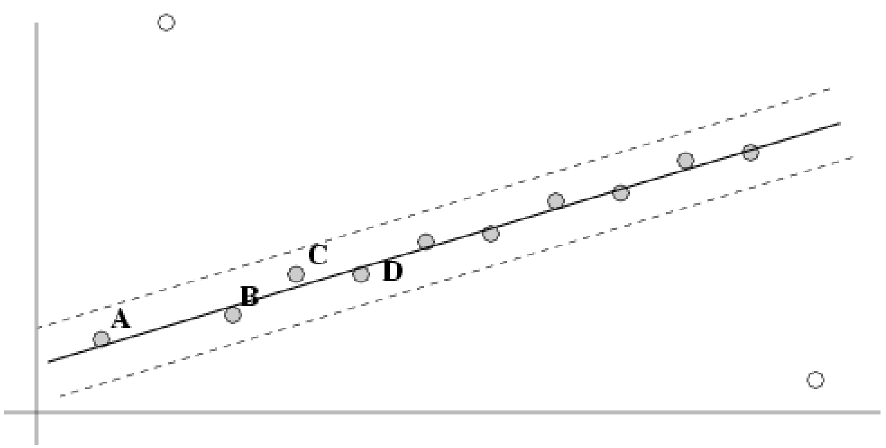
$$k = \frac{\log(z)}{\log(1 - w^n)} \qquad z = 1 - p$$

Sample size n	Proportion of outliers						
	5%	10%	20%	25%	30%	40%	50%
2	**2**	**3**	**5**	**6**	**7**	**11**	**17**
3	**3**	**4**	**7**	**9**	**11**	**19**	**35**
4	**3**	**5**	**9**	**13**	**17**	**34**	**72**
5	**4**	**6**	**12**	**17**	**26**	**57**	**146**
6	**4**	**7**	**16**	**24**	**37**	**97**	**293**
7	**4**	**8**	**20**	**33**	**54**	**163**	**588**
8	**5**	**9**	**26**	**44**	**78**	**272**	**1177**

Refining RANSAC estimate

- RANSAC computes its best estimate from a minimal sample of n points, and divides all data points into inliers and outliers using this estimate.
- We can improve this initial estimate by estimation over all inliers (e.g. with standard least-squares minimization).
- But this may change inliers, so alternate fitting with re-classification as inlier/outlier.





RANSAC: Pros and Cons

- **Pros:**
  – General method suited for a wide range of model fitting problems
  – Easy to implement and easy to calculate its failure rate (1-p)
- **Cons:**
  – Only handles a moderate percentage of outliers without cost blowing up
  – Many real problems have high rate of outliers (but sometimes selective choice of random subsets can help)
- **A voting strategy, The Hough transform [Hough, 1959], can handle high percentage of outliers**
  – Each point votes separately
  – But complexity of search time increases exponentially with the number of model parameters (e.g. 3 for circle)

Summary

- **RANSAC**
  - Algorithm
  - Analysis
    - Number of samples
    - Pros and cons

What we will learn today?

- A model fitting method for line detection
  - RANSAC
- Local invariant features
  - Motivation
  - Requirements, invariances
- Keypoint localization
  - Harris corner detector

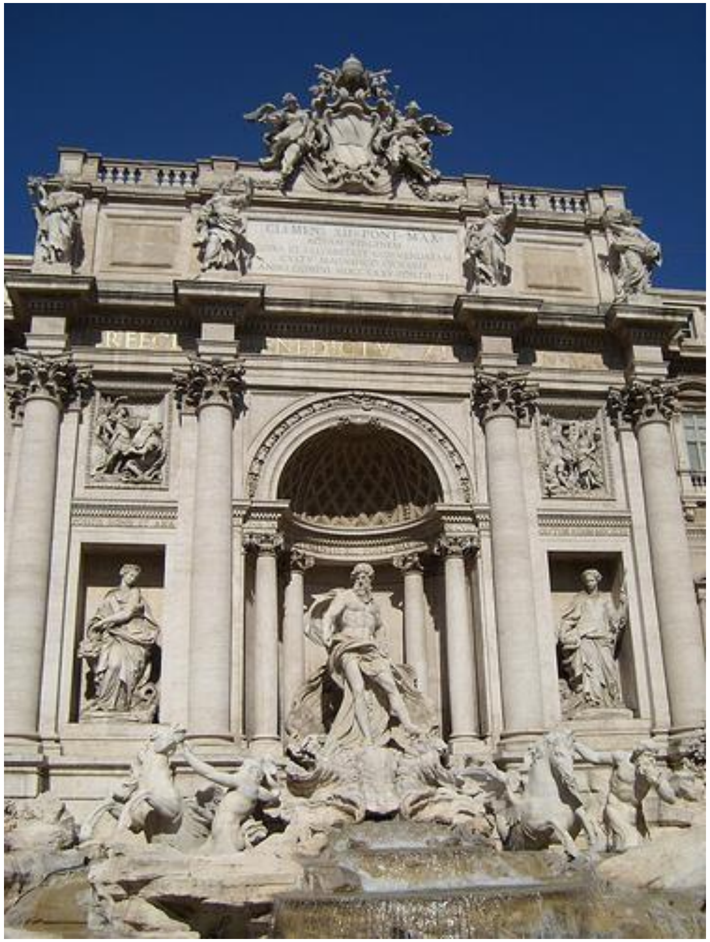
Some background reading:
Rick Szeliski, Chapter 4.1.1; David Lowe, IJCV 2004

Image matching: a challenging problem

Template

Query

Image matching: a challenging problem



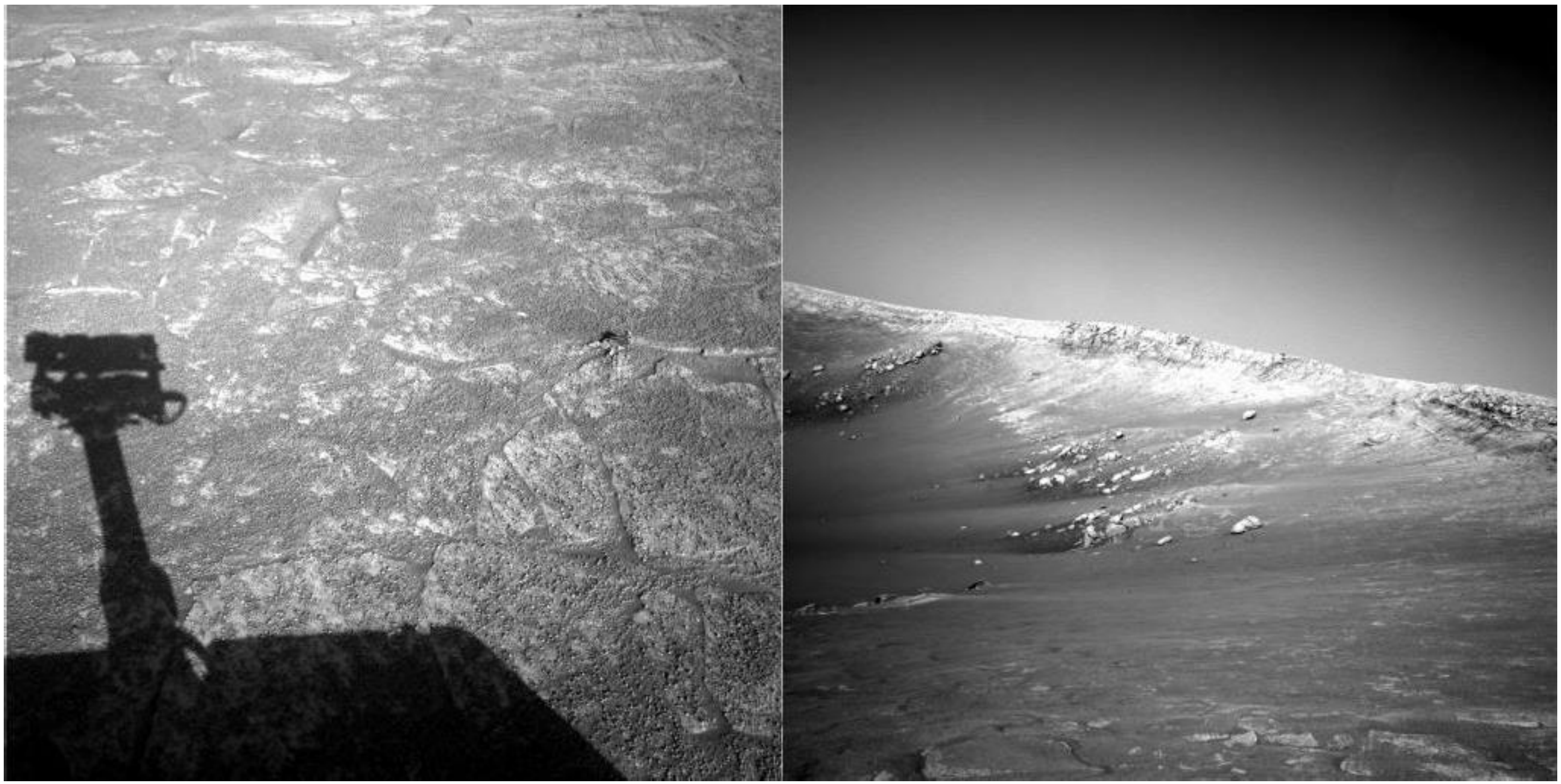
Different photos of the same location, Roma Trevi Fountain

Image matching: a challenging problem

- **Harder case:**

Image matching: a challenging problem

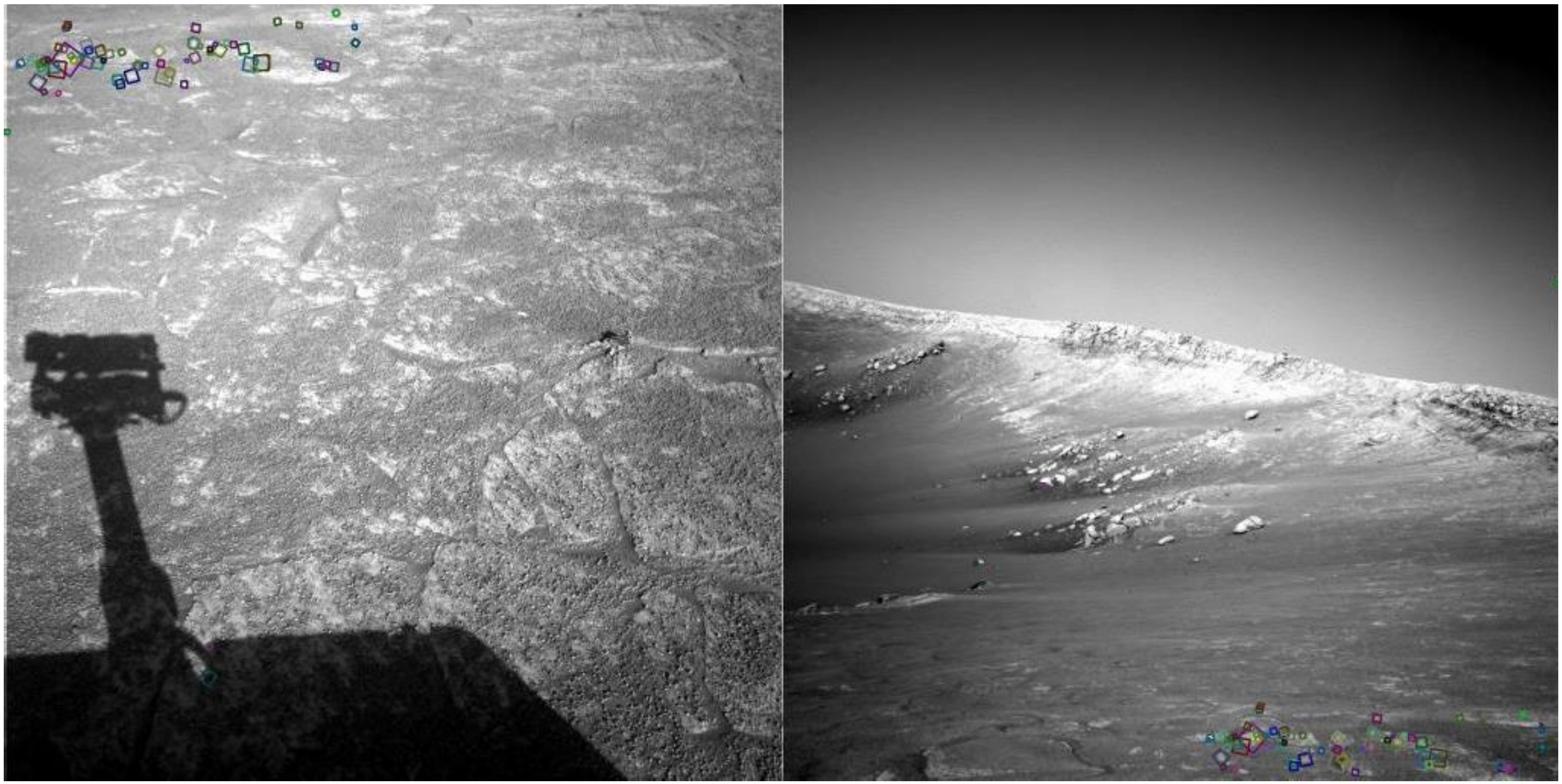
- **Harder Still?**



NASA Mars Rover images

Image matching: a challenging problem

- **Answer Below (Look for tiny colored squares)**

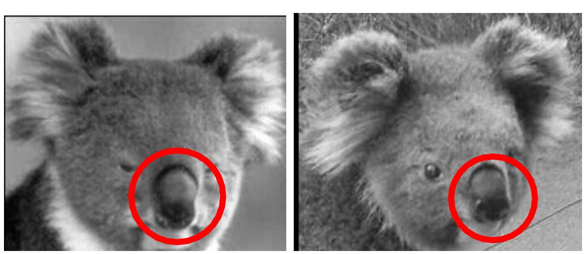


NASA Mars Rover images with SIFT feature matches

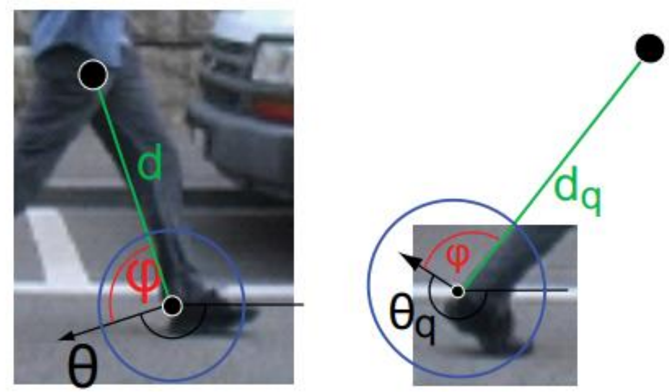
Motivation for using local features

- **Global representations have major limitations.**
- **Instead, describe and match only local regions**
- **Increased robustness to**
  - *Occlusions*

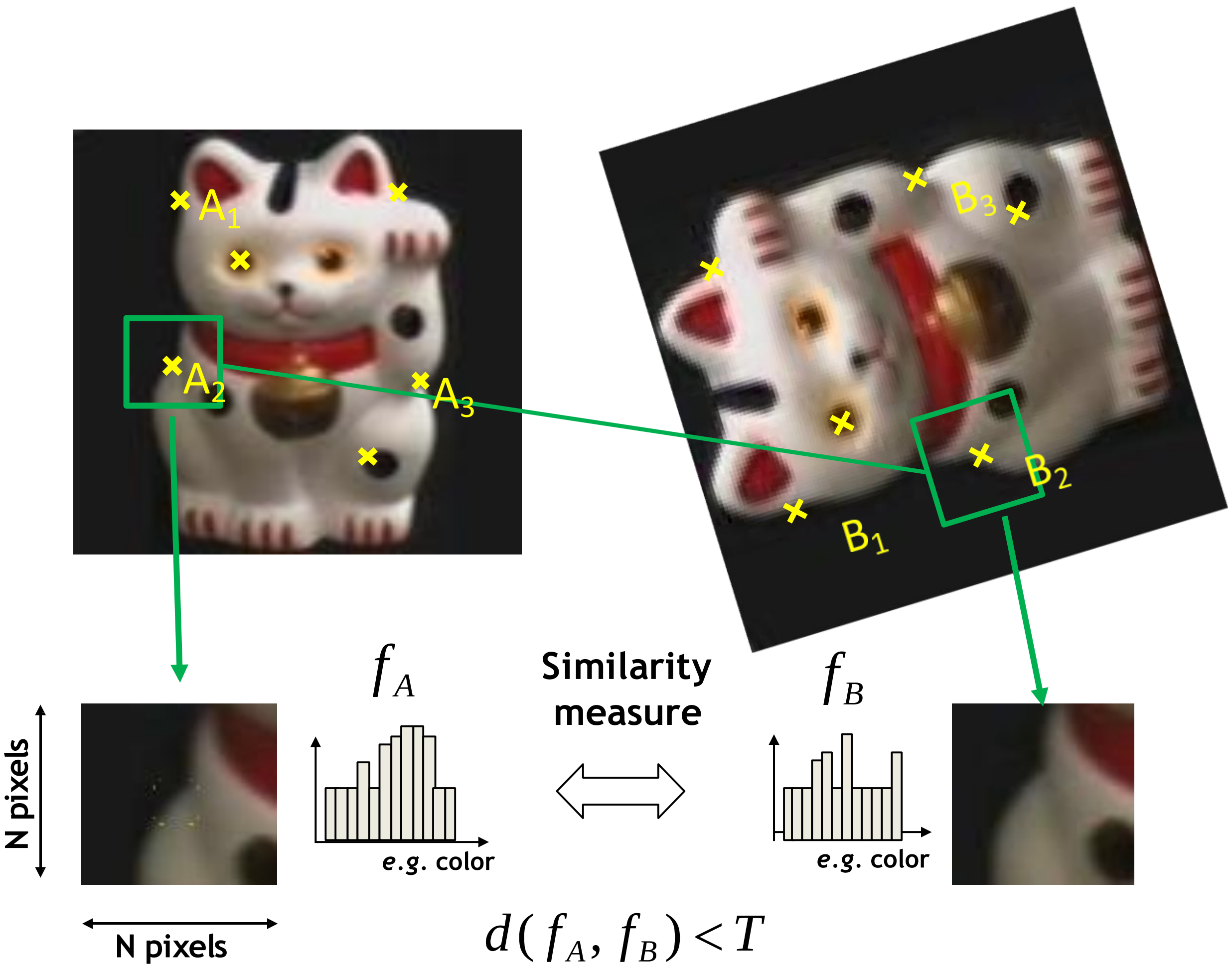
  - *Intra-category variations*



  - *Articulation*



General Approach



1. **Find a set of distinctive key-points**
2. **Define a region around each keypoint**
3. **Extract and normalize the region content**
4. **Compute a local descriptor from the normalized region**
5. **Match local descriptors**

Common Requirements

- Problem 1:
  - Detect the same point independently in both images

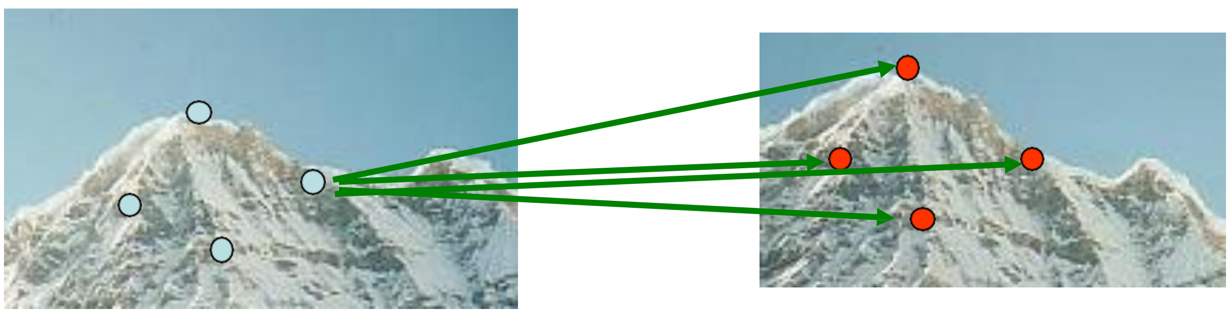
 

No chance to match!

We need a repeatable detector!

Common Requirements

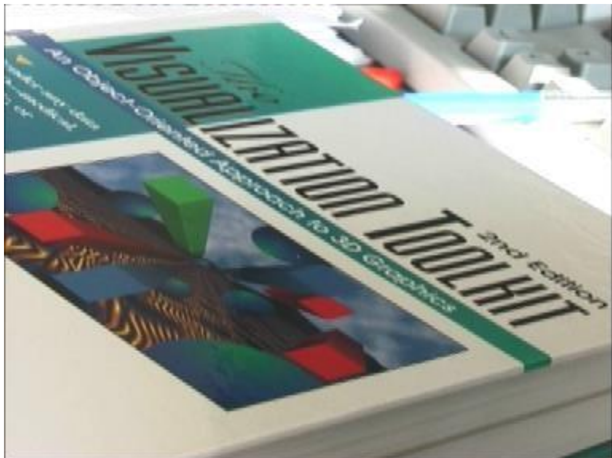
- Problem 1:
  – Detect the same point independently in both images
- Problem 2:
  – For each point correctly recognize the corresponding one

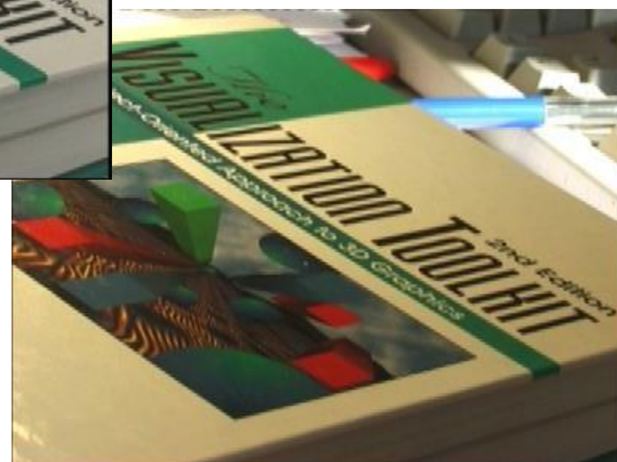


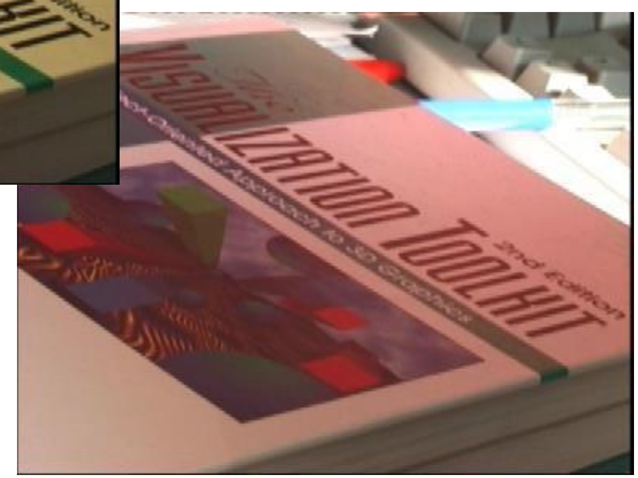
We need a repeatable detector!

Feature Invariances: Geometric Transformations

Feature Invariances: Photometric Transformations







- Often modeled as a linear transformation:
  - Scaling + Offset

Requirements for Local Features

- **Region extraction needs to be repeatable and accurate**
  – Invariant to translation, rotation, scale changes
  – Robust or covariant to out-of-plane (≈affine) transformations
  – Robust to lighting variations, noise, blur, quantization
- **Locality: Features are local, therefore robust to occlusion and clutter.**
- **Quantity: We need a sufficient number of regions to cover the object.**
- **Distinctiveness: The regions should contain “interesting” structure.**
- **Efficiency: Close to real-time performance.**

Many Existing Detectors Available

- Hessian & Harris [Beaudet ‘78], [Harris ‘88]
- Laplacian, DoG [Lindeberg ‘98], [Lowe ‘99]
- Harris-/Hessian-Laplace [Mikolajczyk & Schmid ‘01]
- Harris-/Hessian-Affine [Mikolajczyk & Schmid ‘04]
- EBR and IBR [Tuytelaars & Van Gool ‘04]
- MSER [Matas ‘02]
- Salient Regions [Kadir & Brady ‘01]
- Others…

- Those detectors have become a basic building block for many applications in Computer Vision

Summary

- **Region extraction needs to be repeatable and accurate**
  - Invariant to translation, rotation, scale changes
  - Robust or covariant to out-of-plane (≈affine) transformations
  - Robust to lighting variations, noise, blur, quantization
- **Local invariant features**
  - Motivation
  - General approach and requirements

What we will learn today?

- A model fitting method for line detection
  – RANSAC
- Local invariant features
  – Motivation
  – Requirements, invariances
- Keypoint localization
  – Harris corner detector

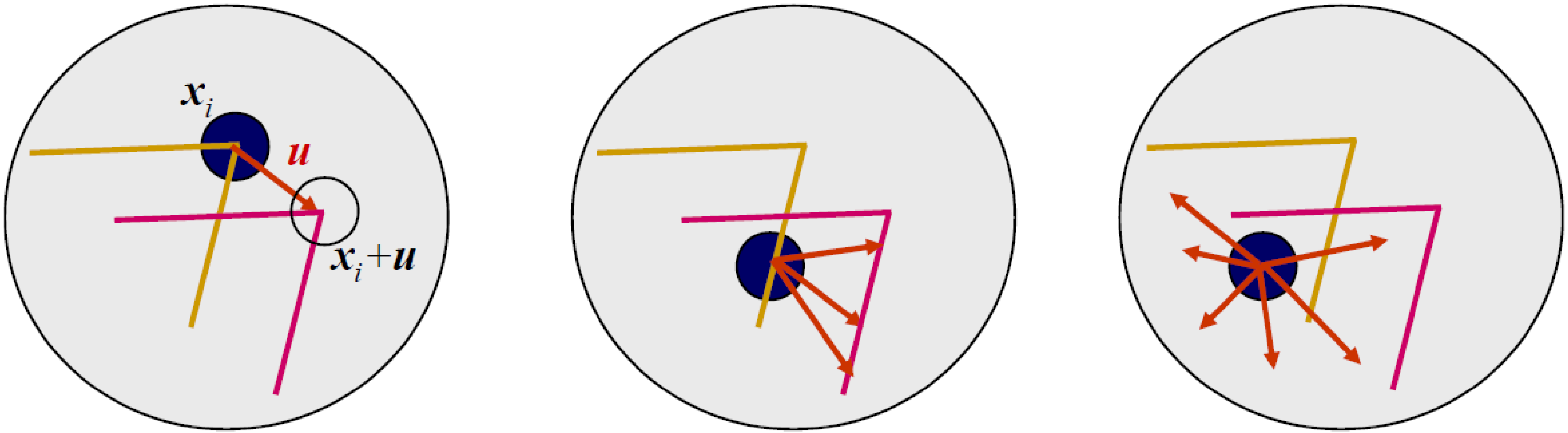
Keypoint Localization

- **Goals:**

1. Repeatable detection
2. Precise localization
3. Interesting content

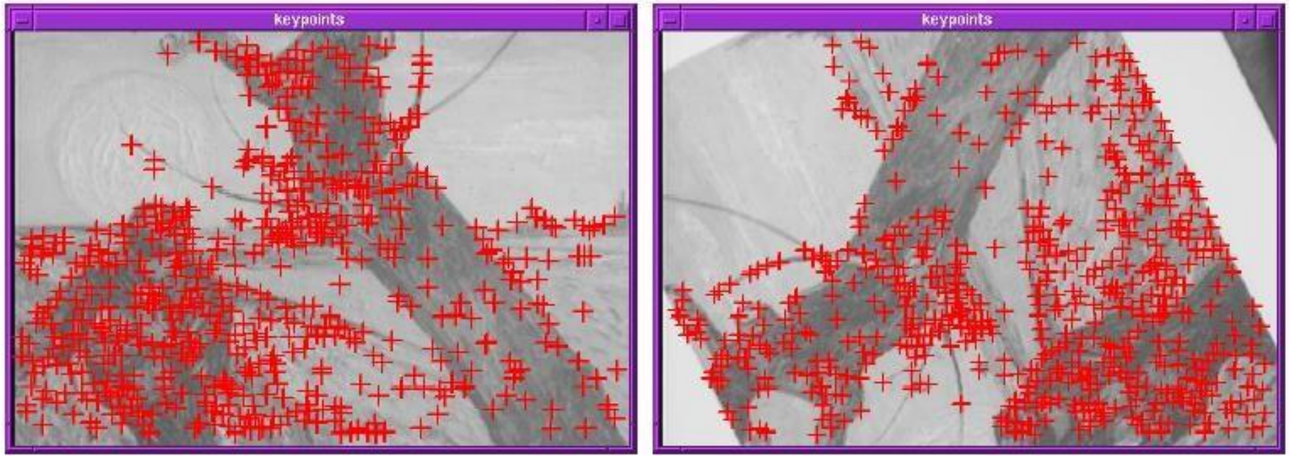
⇒ **Look for two-dimensional signal changes**

What are good keypoints?



(a) Corner (b) Edge (c) Textureless

Finding Corners



- **Key property:**

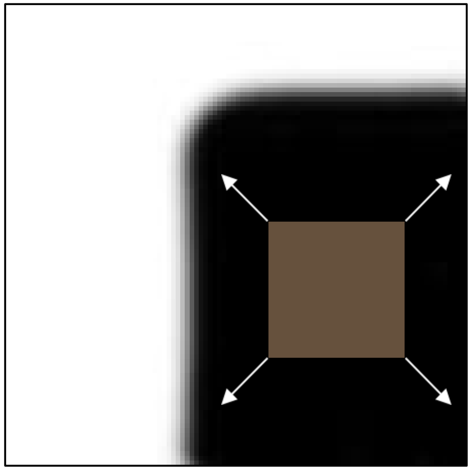
In the region around a corner, the image gradient has two or more dominant directions.

- **Corners are repeatable and distinctive**

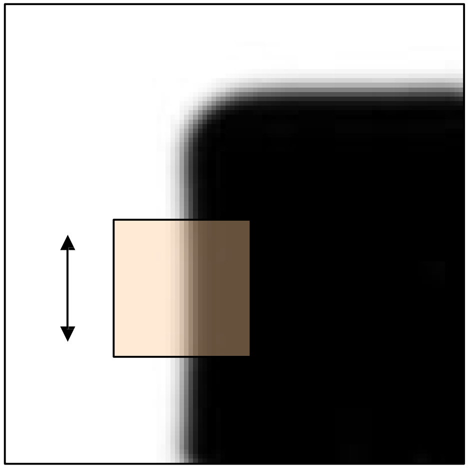
Corners as Distinctive Interest Points

- **Design criteria**

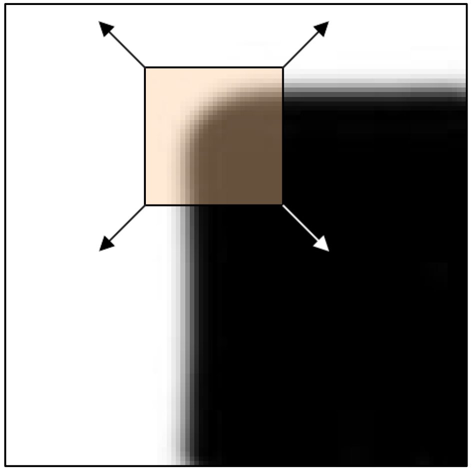
1. We should easily recognize the corner point by looking through a small window (**locality**).
2. Shifting the window in any direction should give a large change in intensity (**good localization**).



**"flat" region:
no change in
all directions**

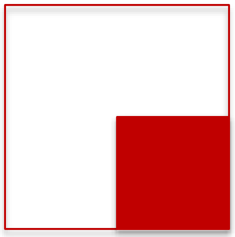


**"edge":
no change along
the edge direction**



**"corner":
significant change
in all directions**

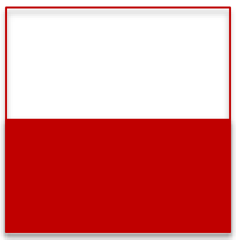
Corners versus edges



$\sum I_x^2 \longrightarrow$ Large

$\sum I_y^2 \longrightarrow$ Large

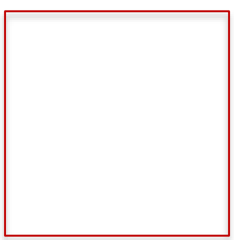
Corner



$\sum I_x^2 \longrightarrow$ Small

$\sum I_y^2 \longrightarrow$ Large

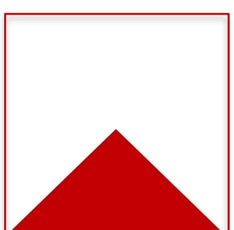
Edge



$\sum I_x^2 \longrightarrow$ Small

$\sum I_y^2 \longrightarrow$ Small

Flat



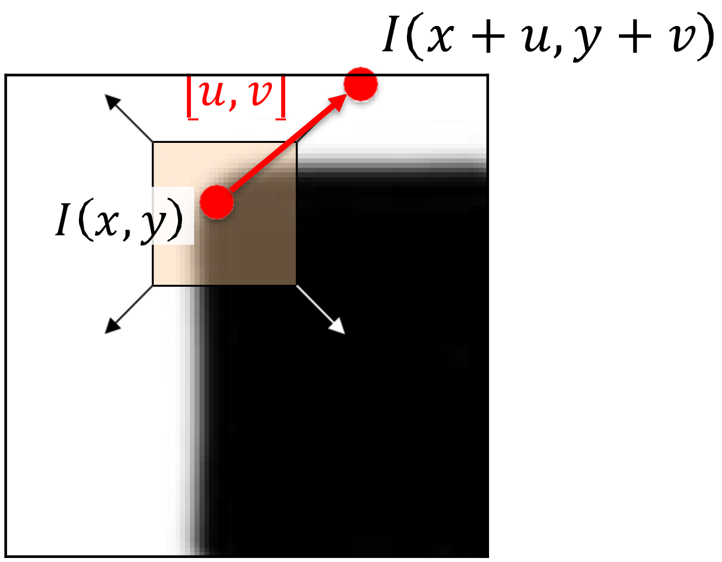
$\sum I_x^2 \longrightarrow$??

$\sum I_y^2 \longrightarrow$??

Corner

Harris Detector Formulation

- Localize patches that result in large change of intensity when shifted in any direction.
- When we shift by $[u, v]$, the intensity change at the center pixel is:



- *Measure change as intensity difference:*

$$(I(x+u, y+v) - I(x, y))$$

- *That's for a single point, but we have to accumulate over the patch or "small window" around that point…*

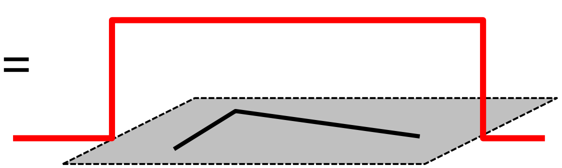
"corner": significant change in all directions

Harris Detector Formulation

- When we shift by [u, v], the change in intensity for the "small window" is:

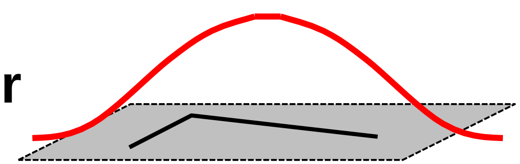
$$E(u,v) = \underbrace{\sum_{x,y}}_{\text{Sum over window}} \underbrace{w(x,y)}_{\text{Window function}} \underbrace{[\underbrace{I(x+u,y+v)}_{\text{Shifted intensity}} - \underbrace{I(x,y)}_{\text{Intensity}}]^2}_{\text{Intensity change}}$$

Window function $w(x,y) =$



1 in window, 0 outside

or



Gaussian

Harris Detector Formulation

- This measure of change can be approximated by (Taylor expansion):

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

where M is a 2×2 matrix computed from image derivatives:

Auto-correlation matrix of gradients

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

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

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

Harris Detector Formulation

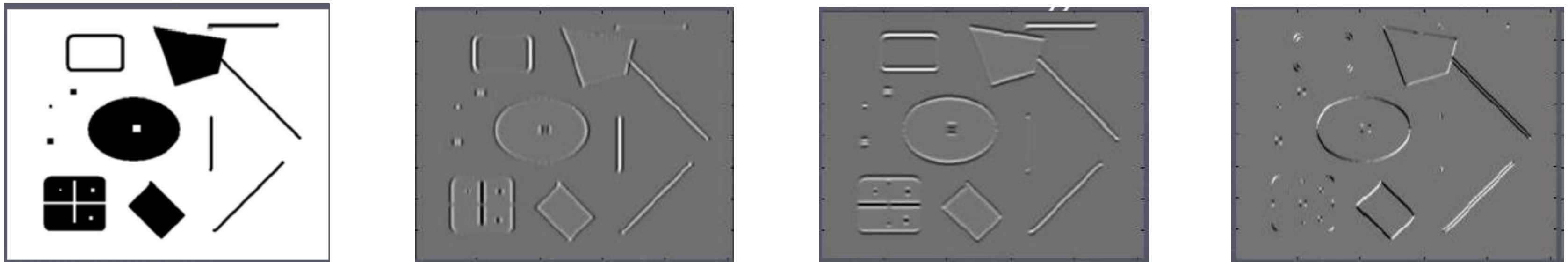


Image I $\quad I_x \quad I_y \quad I_x I_y$

- *where M is a 2×2 matrix computed from image derivatives:*

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

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

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

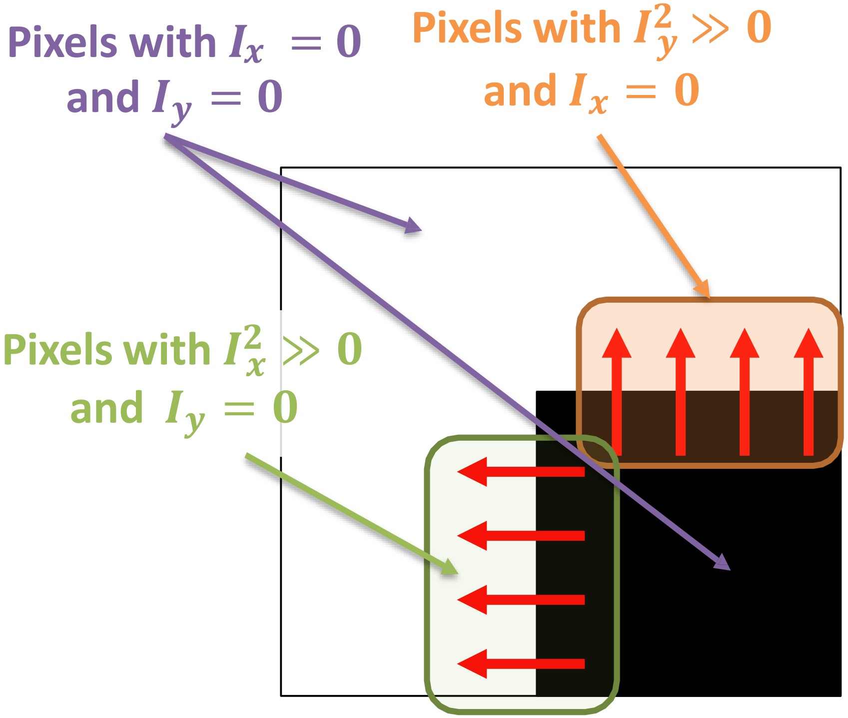
$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y]$$

What Does This Matrix Reveal?

- *First, let's consider an axis-aligned corner.*
- *In that case, the dominant gradient directions align with the x or the y axis。*

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

- *This means: if either λ is close to 0, then this is not a corner, so look for image windows where both λ are large.*
- *What if we have a corner that is not aligned with the image axes?*

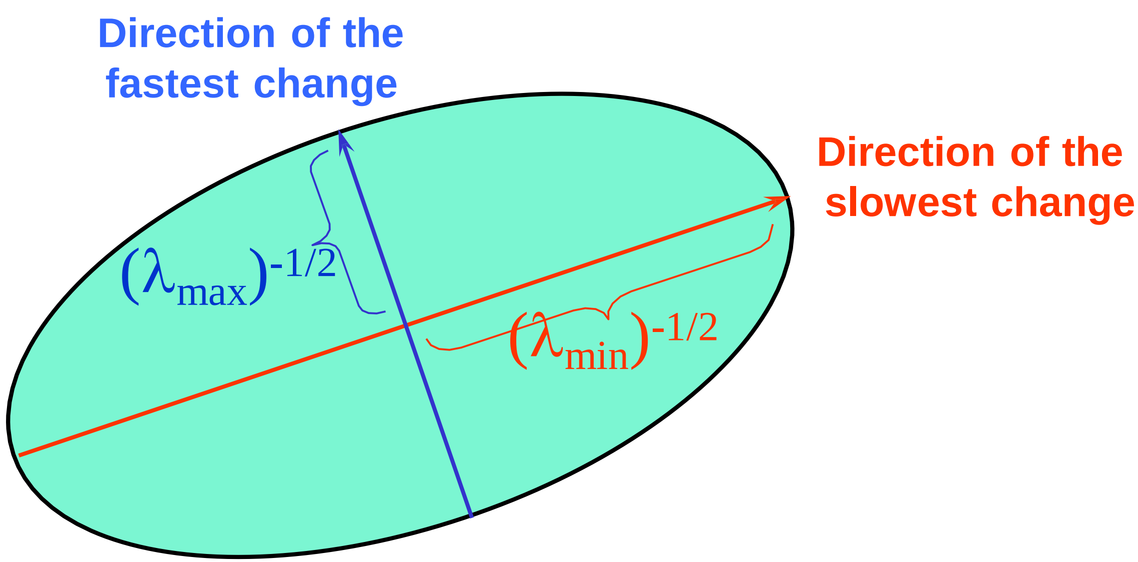


General Case

- *Since* $M = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$ *is symmetric, we can re-rewrite*

  $M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$. **(Eigenvalue decomposition)**

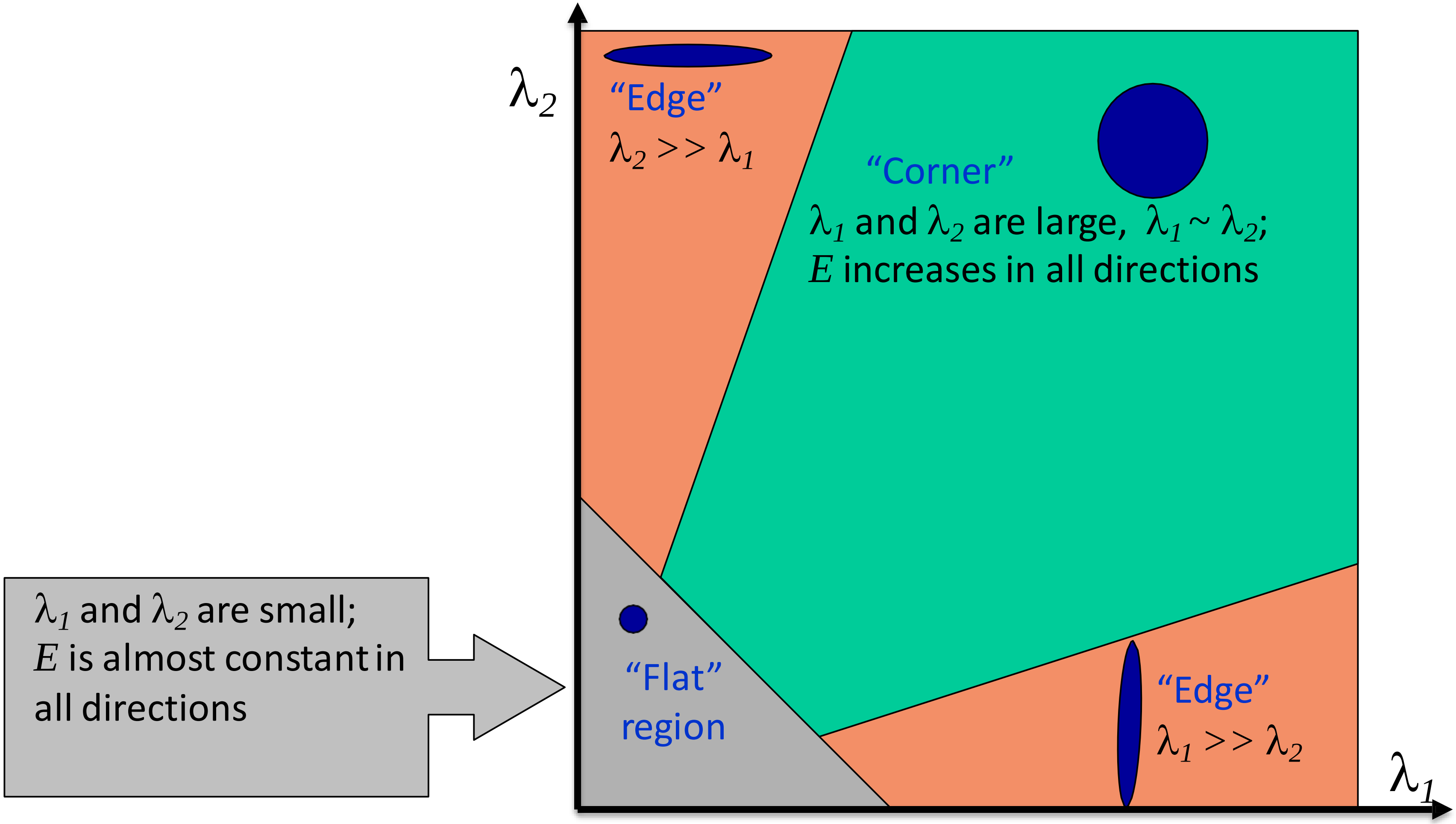
- We can think of M as an ellipse with its axis lengths determined by the eigenvalues $\lambda_1$and λ_2; and its orientation determined by R



- *A rotated corner would produce the same eigenvalues as its non-rotated version.*

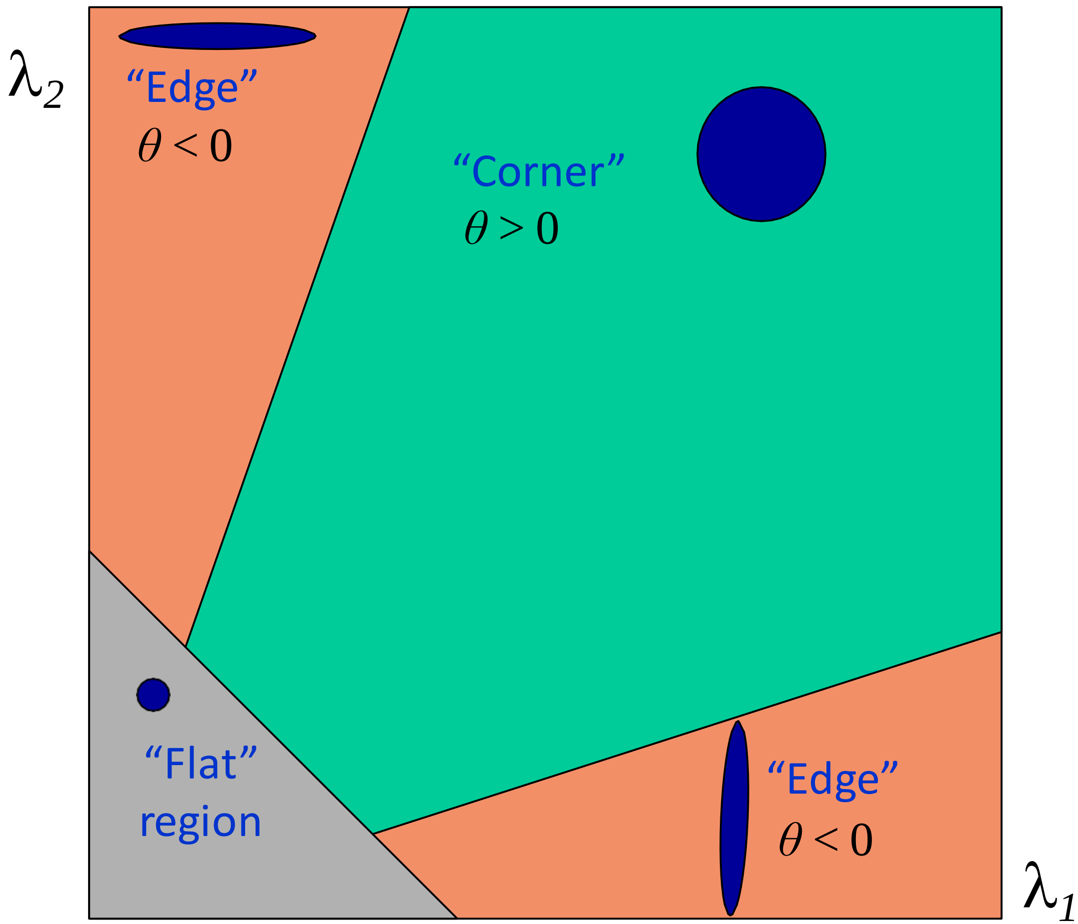
Interpreting the Eigenvalues

- *Classification of image points using eigenvalues of M:*



Corner Response Function

$$\theta = \det(M) - \alpha \,\mathrm{trace}(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$



Fast approximation [Harris and Stephens, 1988]
– Avoid computing the eigenvalues
– α: constant
(0.04 to 0.06)

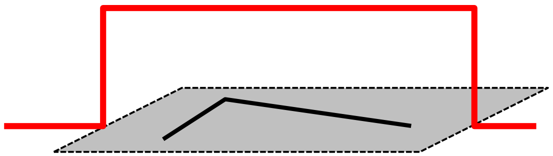
Window Function w(x,y)

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

- Option 1: uniform window
  - Sum over square window

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

  - Problem: not rotation invariant

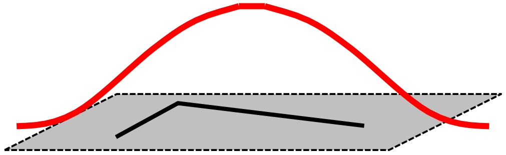


1 in window, 0 outside

- Option 2: Smooth with Gaussian
  - Gaussian already performs weighted sum

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

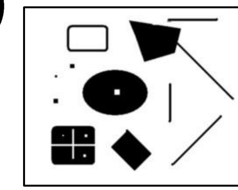
  - Result is rotation invariant



Gaussian

Summary: Harris Detector

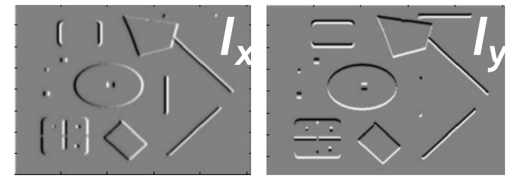
- *Compute second moment matrix (autocorrelation matrix)*



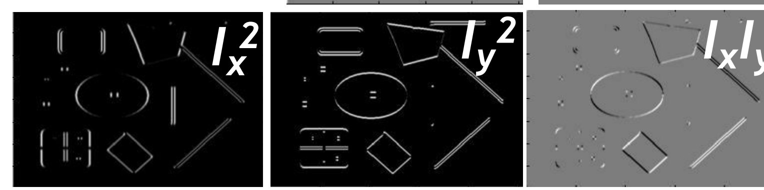
$$M(\sigma_I,\sigma_D) = g(\sigma_I) * \begin{bmatrix} I_x^2(\sigma_D) & I_x I_y(\sigma_D) \\ I_x I_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

σ_D: for Gaussian in the derivative calculation
σ_I : for Gaussian in the windowing function

1. Image derivatives



2. Square of derivatives



3. Gaussian filter $g(\sigma_I)$

4. Cornerness function – two strong eigenvalues

$$\theta = \det[M(\sigma_I,\sigma_D)] - \alpha[\mathrm{trace}(M(\sigma_I,\sigma_D))]^2$$
$$= g(I_x^2)g(I_y^2) - [g(I_xI_y)]^2 - \alpha[g(I_x^2)+g(I_y^2)]^2$$

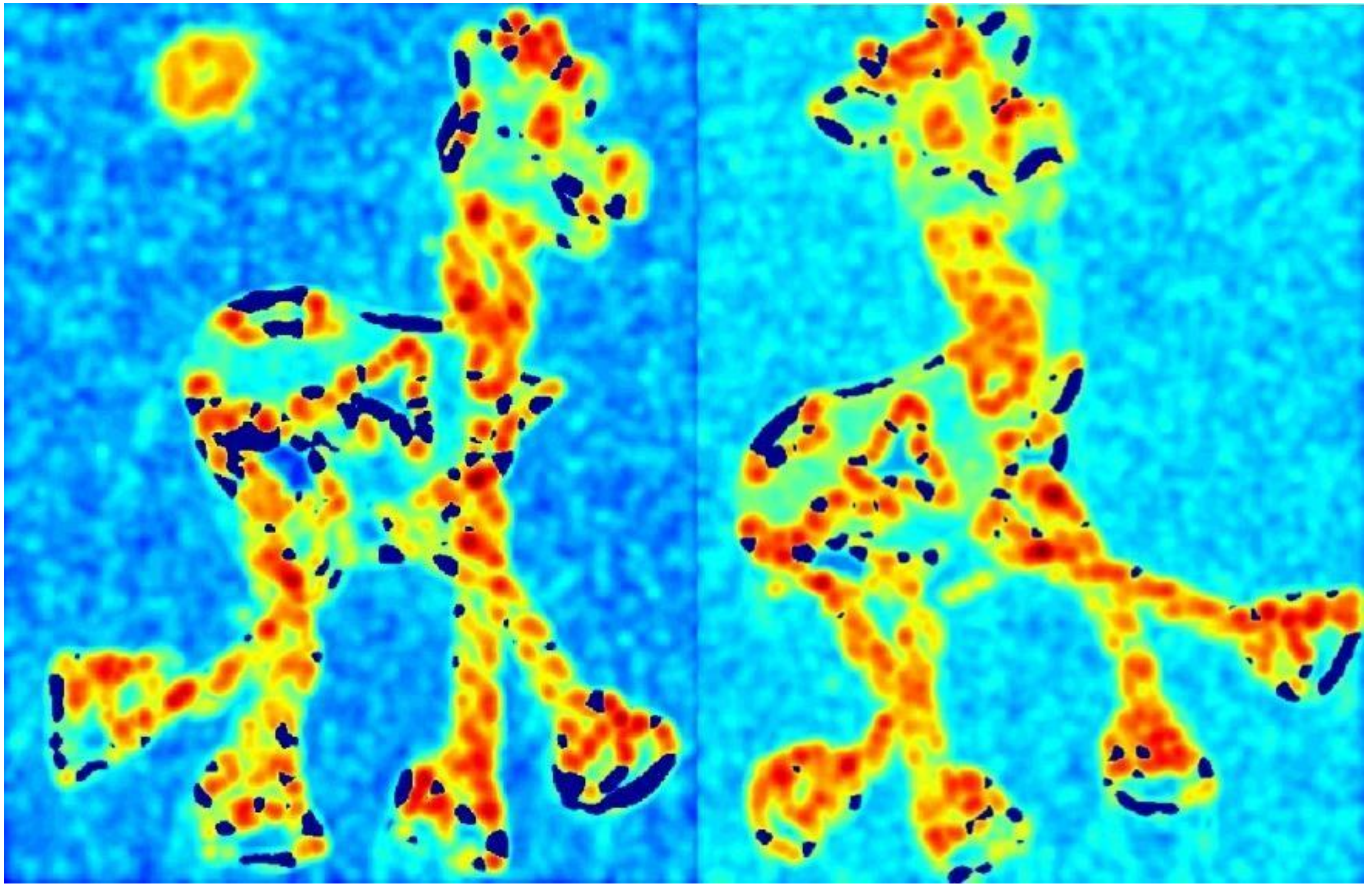
5. Perform non-maximum suppression

Harris Detector: Workflow

- Input image

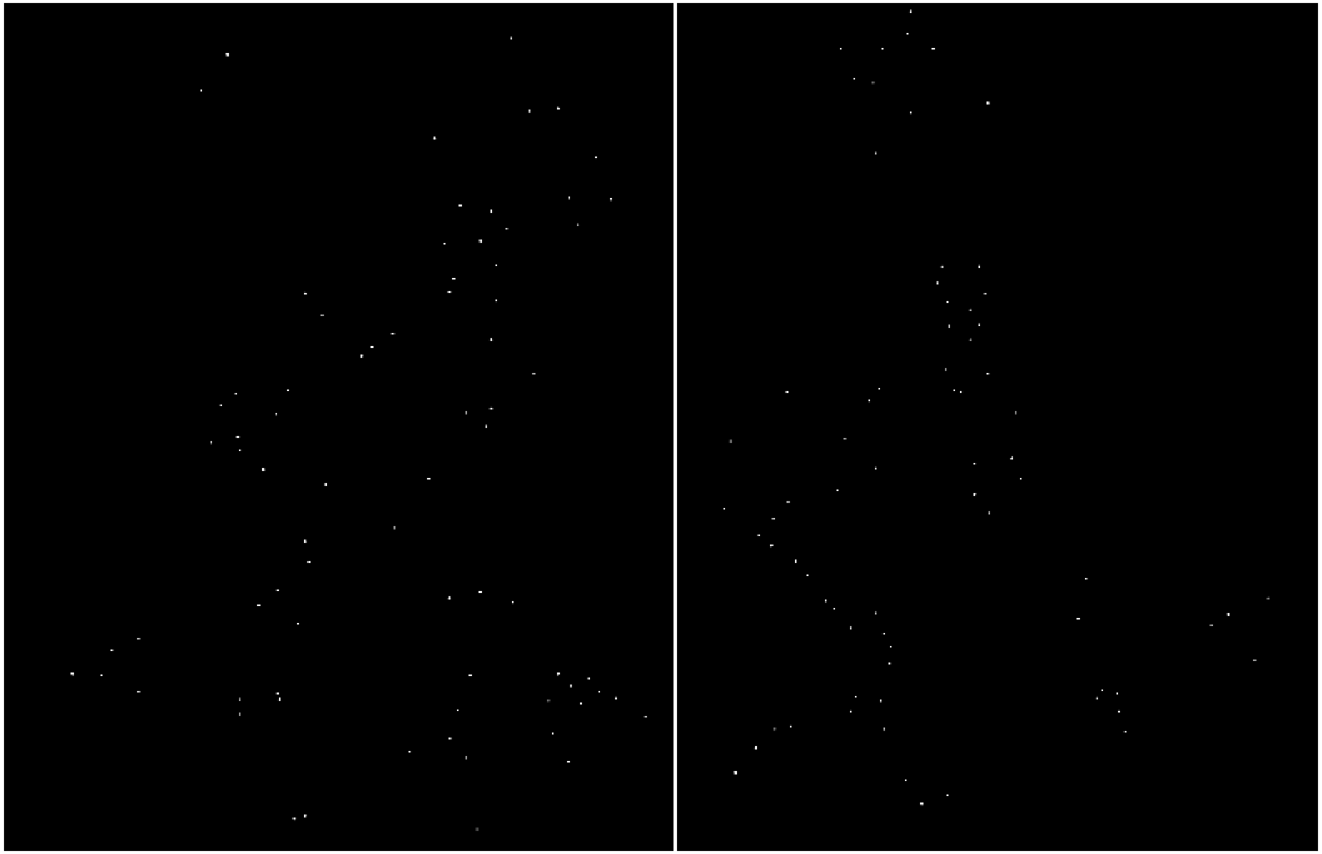
Harris Detector: Workflow

- Input Image
- Compute corner response function θ



Harris Detector: Workflow

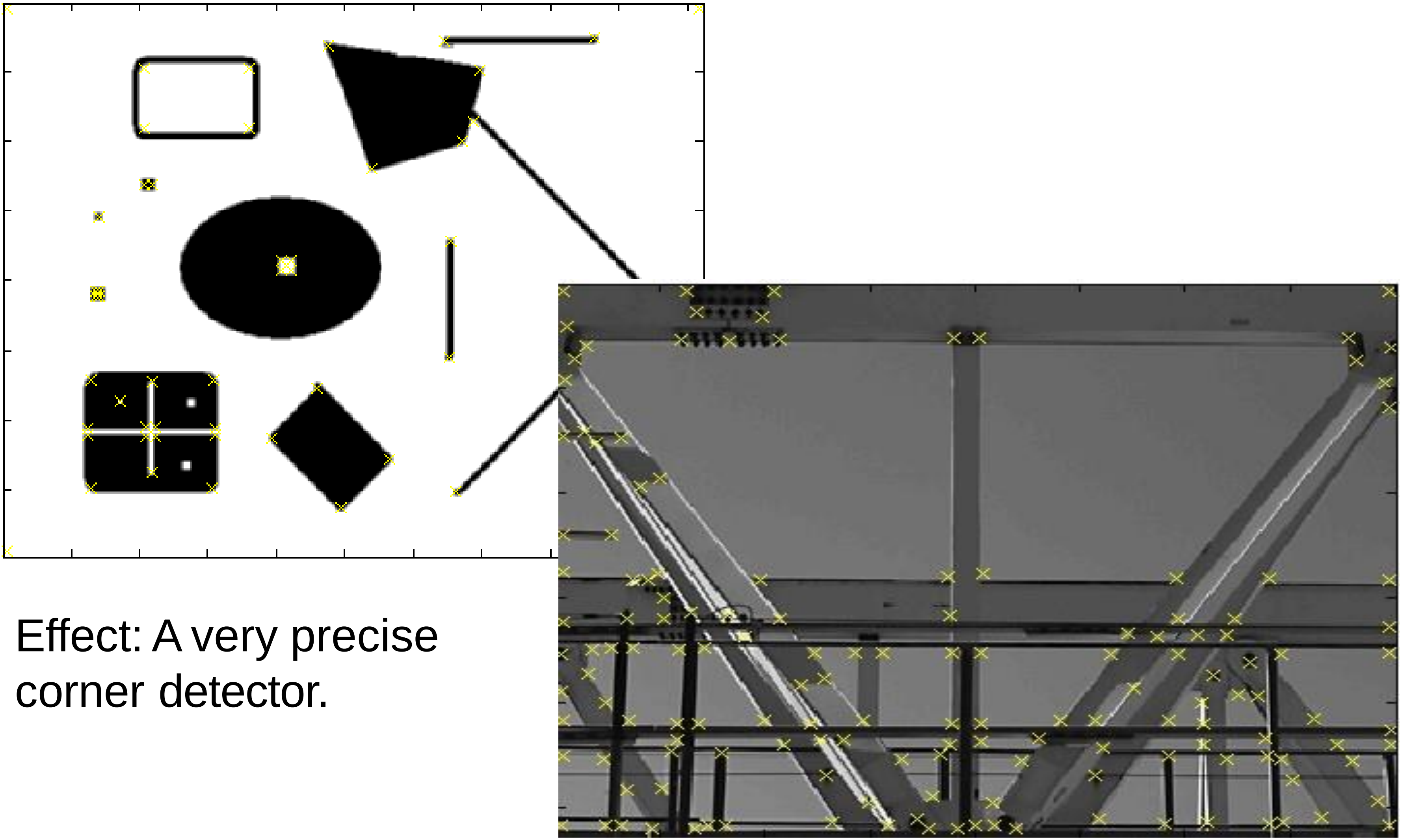
- Input Image
- Compute corner response function θ
- Take only the local maxima of θ, where θ > threshold



Harris Detector: Workflow

- Input Image
- Compute corner response function θ
- Take only the local maxima of θ, where θ > threshold

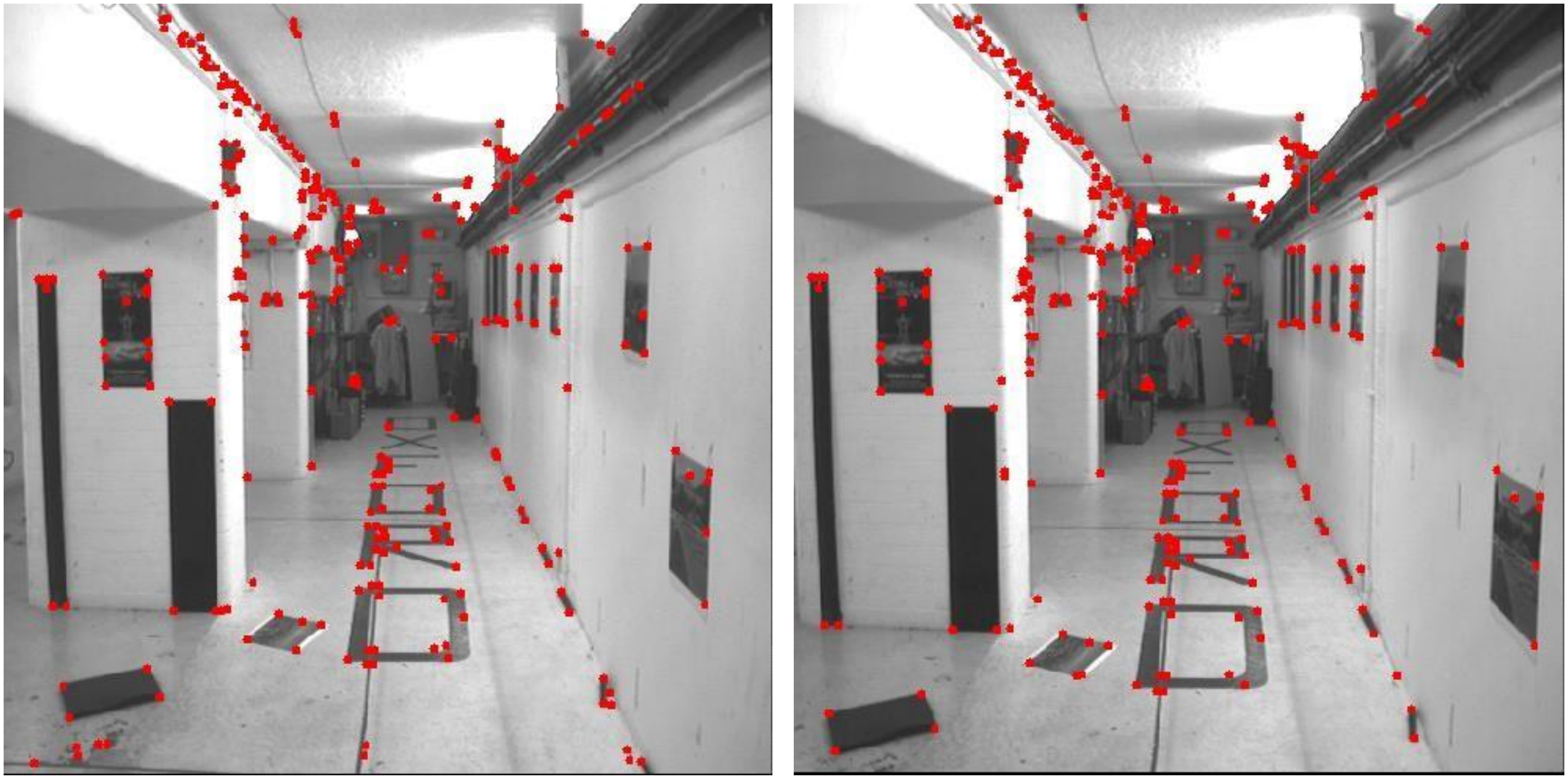
Harris Detector – Responses



Effect: A very precise corner detector.

Harris Detector – Responses

Harris Detector – Responses



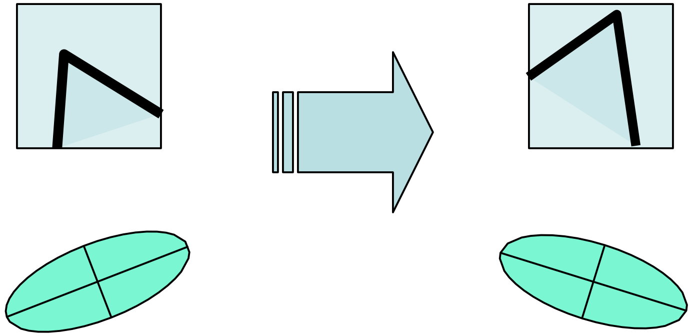
- Results are well suited for finding stereo correspondences

Harris Detector: Properties

- Translation invariance?

Harris Detector: Properties

- Translation invariance
- Rotation invariance?

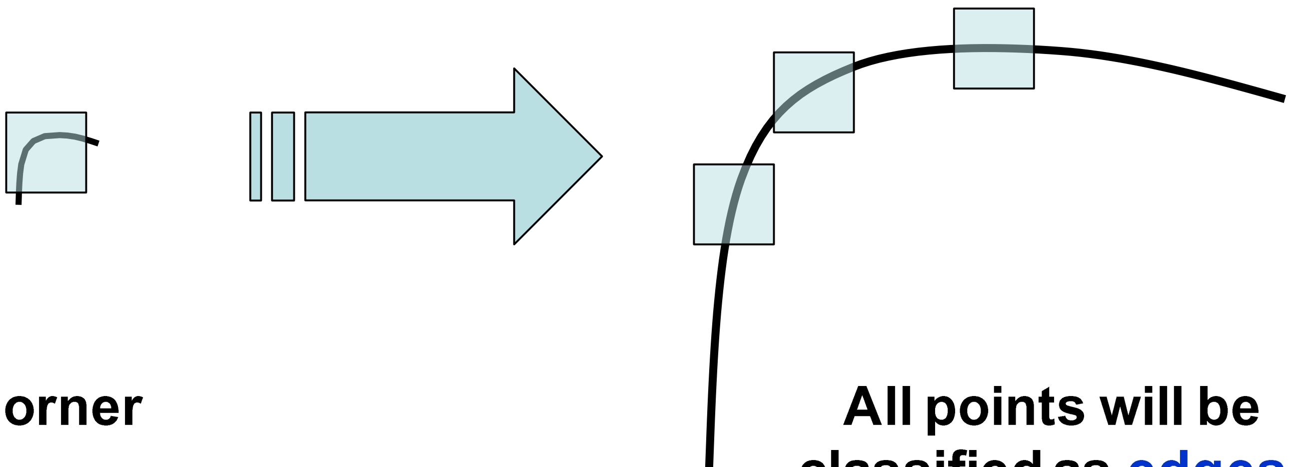


Ellipse rotates but its shape (i.e. eigenvalues) remains the same

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

Harris Detector: Properties

- Translation invariance
- Rotation invariance
- Scale invariance?



Not invariant to image scale!

Summary

- Harris corner detector
  - Formulation
  - Examples

中山大學
SUN YAT-SEN UNIVERSITY

Next time:

Feature descriptors

Pattern Recognition (in Computer Vision)

Jinhua Ma,
School of Computer Science and Engineering, Sun Yat-Sen University

Most of our slides have been borrowed from the courses of Stanford CS131 and CS231N, Thanks!