SAIR

Spatial AI & Robotics Lab

CSE 473/573-A

L8: FEATURE DETECTION & MATCHING

Chen Wang

Spatial AI & Robotics Lab

Department of Computer Science and Engineering

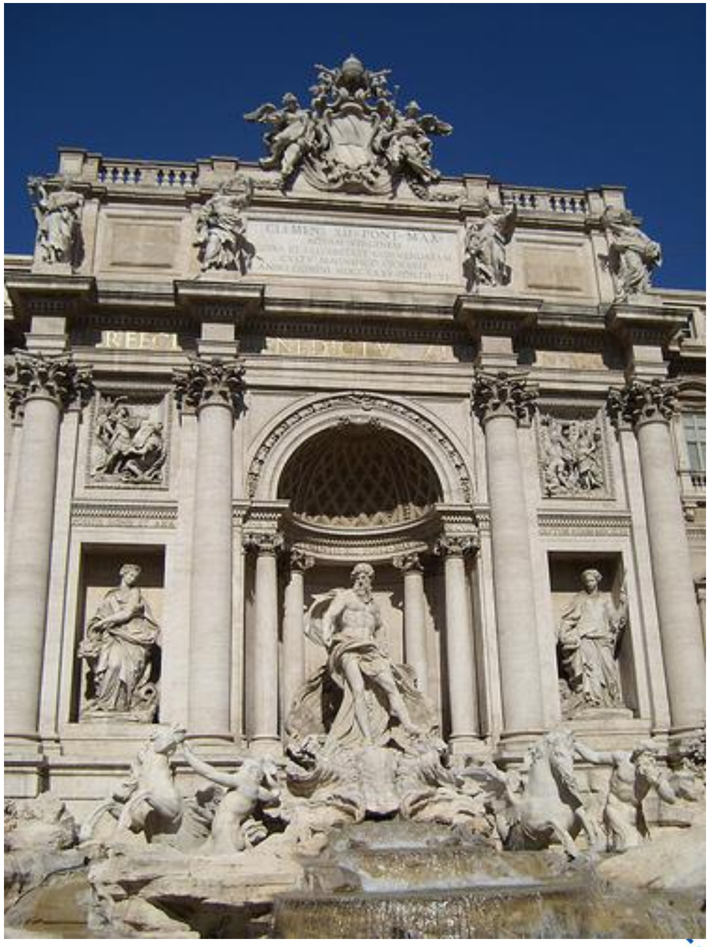
University at Buffalo The State University of New York

Content

- Feature Extraction
  - Local features, Pyramids for invariant feature detection
  - Invariant descriptors and matching
    - Harris Detection, SIFT.
  - Matching
    - Precision, Recall, F1, ROC
  - SURF, Integral Images

Image matching

by Diva Sian



by swashford

Harder case

by Diva Sian

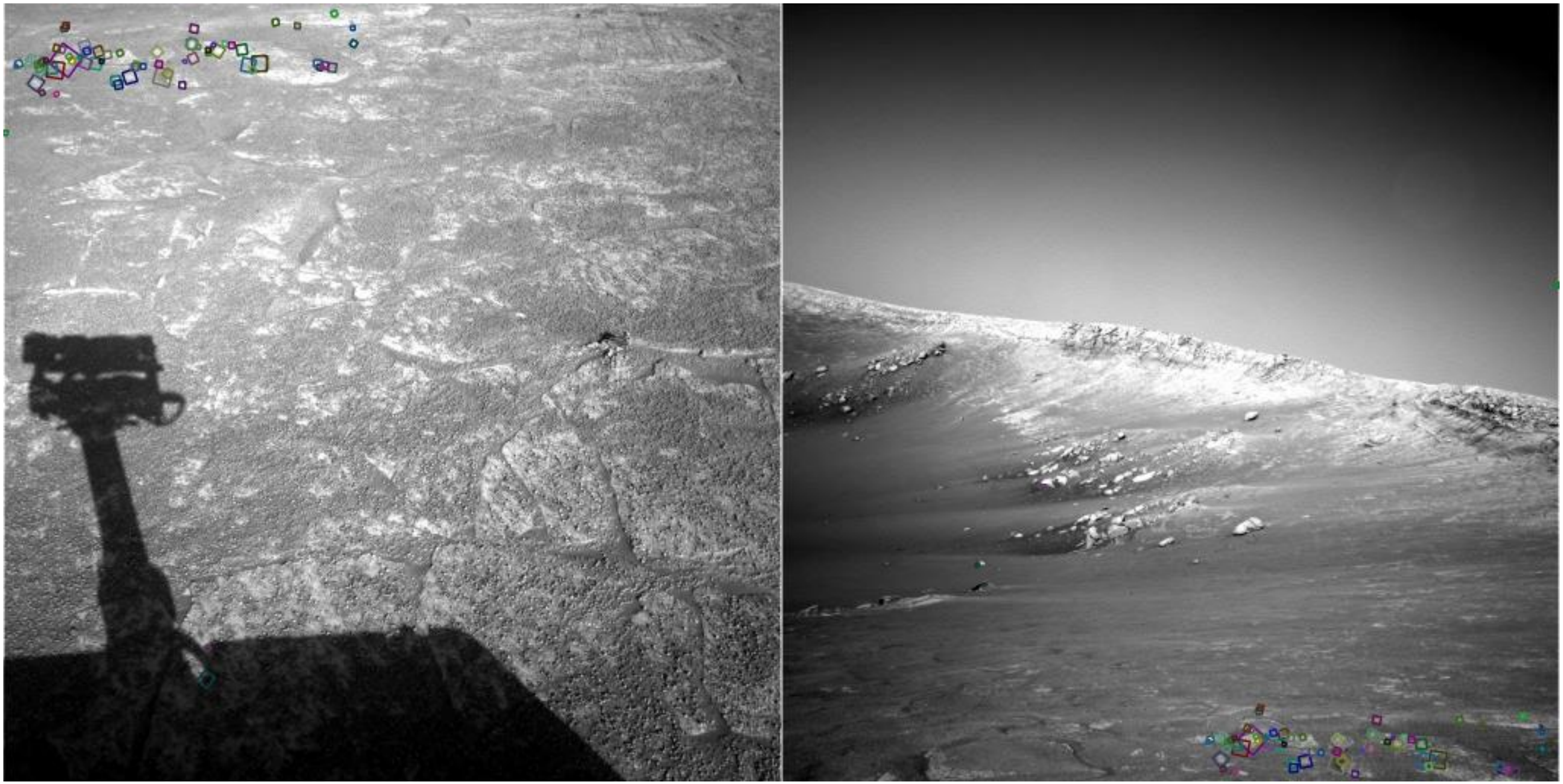
by scgbt

Harder still?

NASA Mars Rover images

Answer below (look for tiny colored squares…)



NASA Mars Rover images with SIFT feature matches
Figure by Noah Snavely

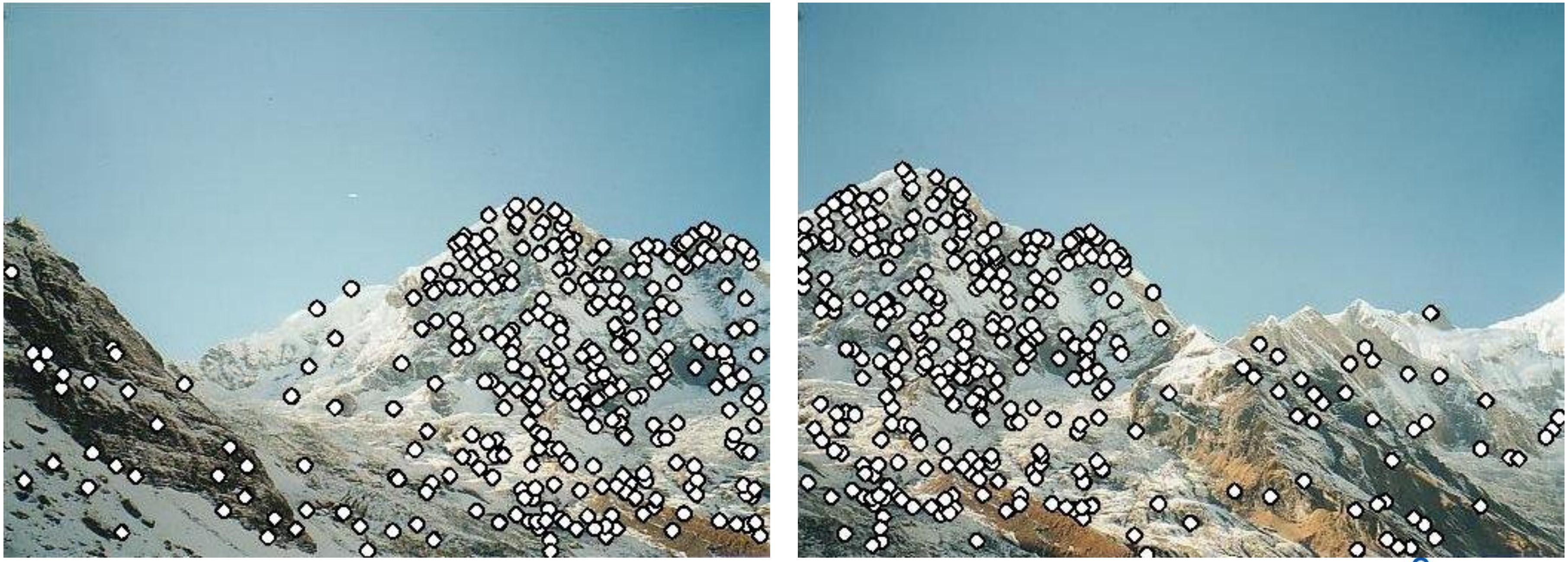
Local features and alignment

- We need to match (align) images
- Global methods sensitive to occlusion, lighting, parallax.
- Look for local features that match well.
- How would you do it by eye?

[Darya Frolova and Denis Simakov]

Local features and alignment

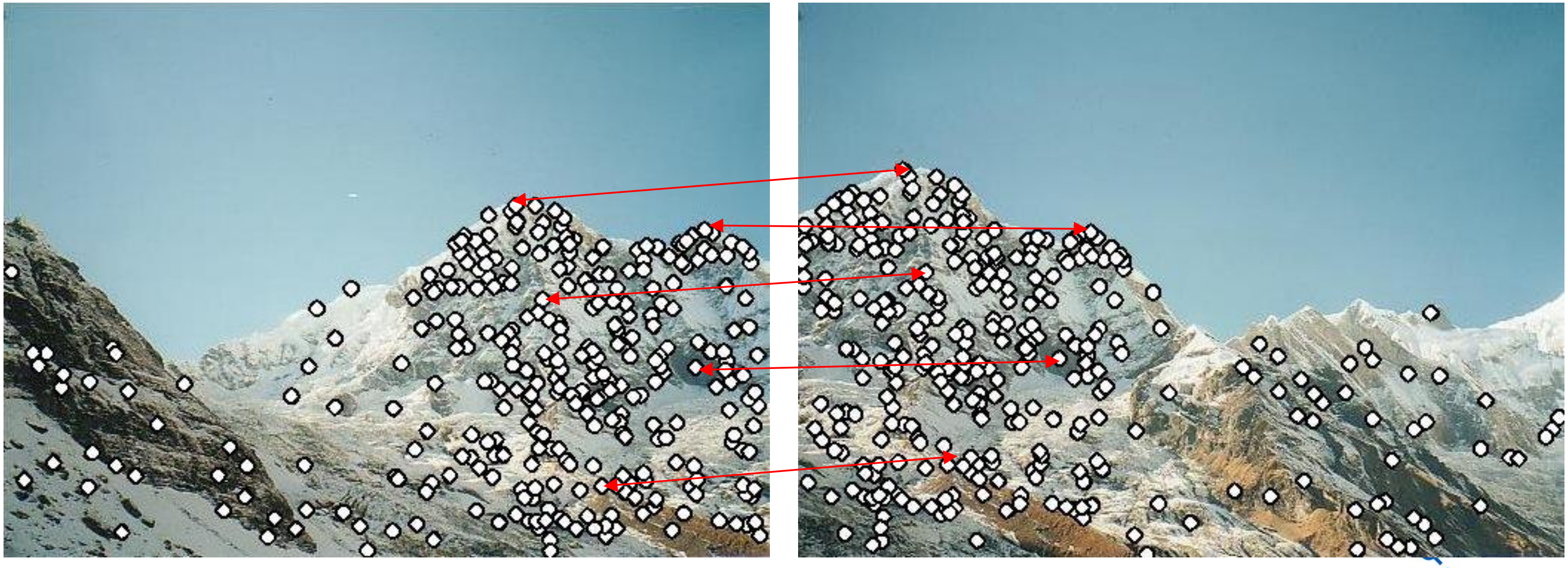
- 1. Detect feature points in both images.



[Darya Frolova and Denis Simakov]

Local features and alignment

- 1. Detect feature points in both images.
- 2. Find corresponding pairs.



[Darya Frolova and Denis Simakov]

Local features and alignment

- 1. Detect feature points in both images.
- 2. Find corresponding pairs.
- 3. Use these pairs to align images

[Darya Frolova and Denis Simakov]

Local features and alignment

- Problem 1:
  - Detect the *same* points *independently* in both images


no chance to match!

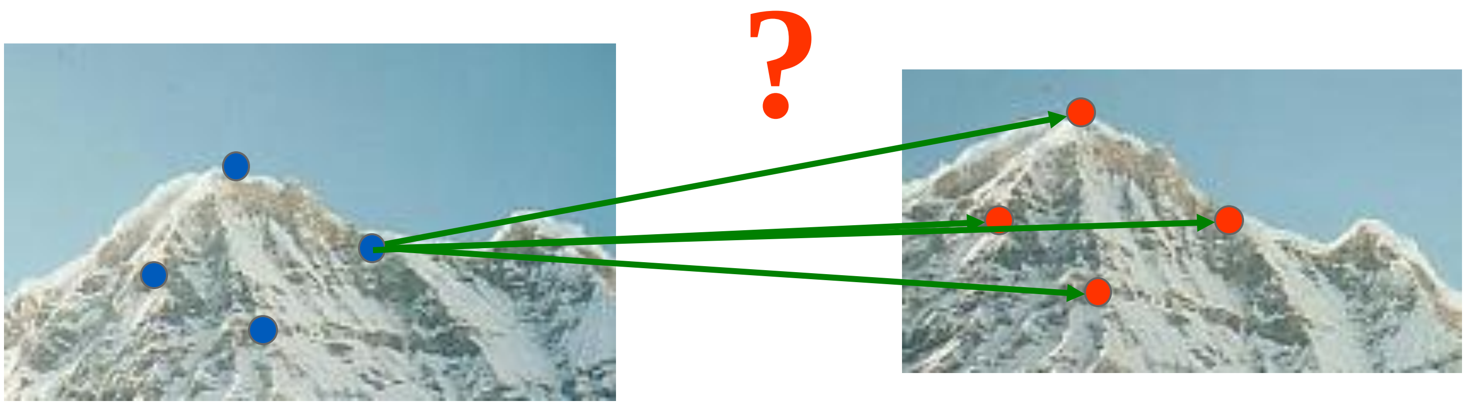
We need a repeatable detector

[Darya Frolova and Denis Simakov]

SAIR Spatial AI & Robotics Lab

Local features and alignment

- Problem 2:
  - For each point correctly recognize the corresponding one

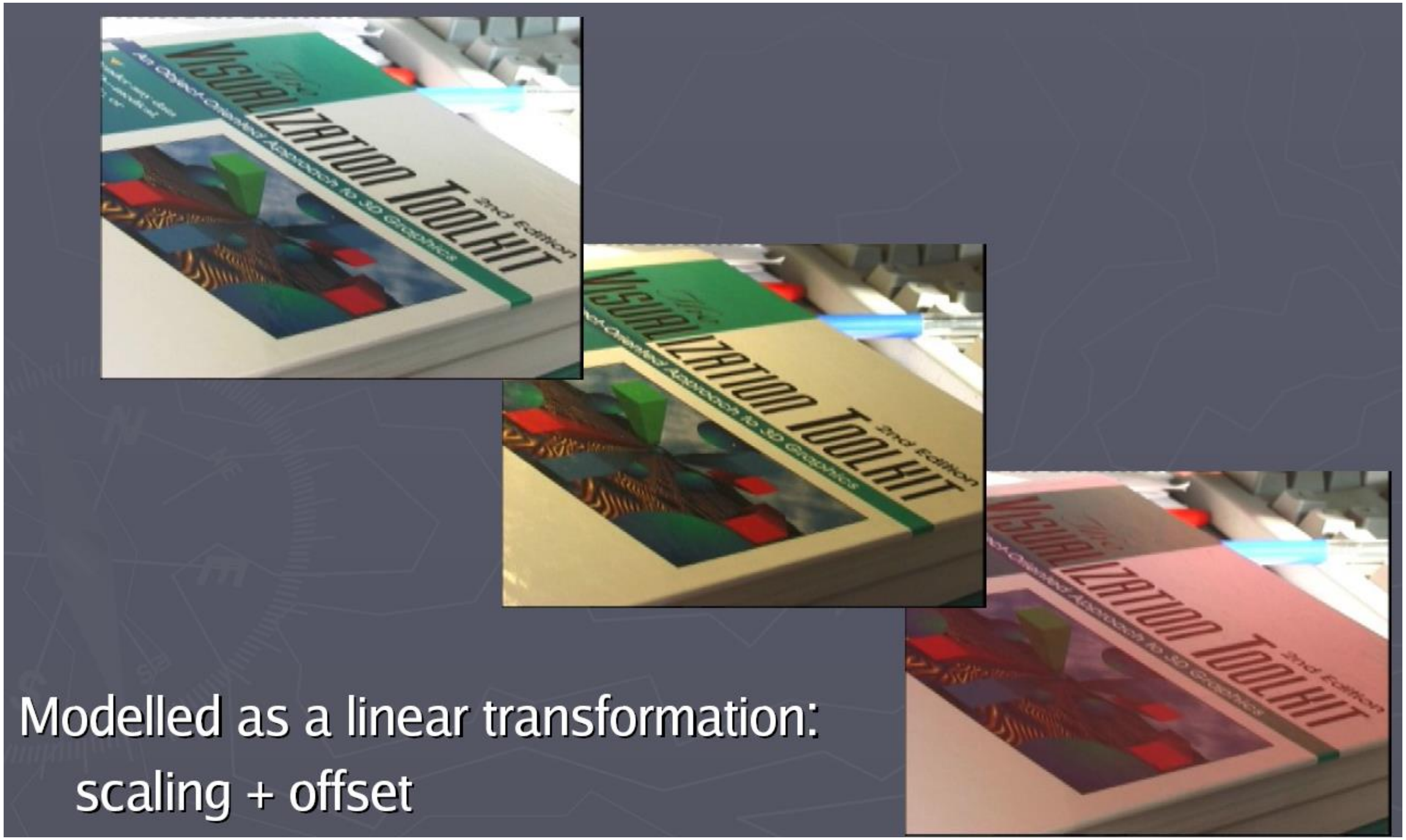


We need a reliable and distinctive descriptor

[Darya Frolova and Denis Simakov]

Geometric transformations

Photometric transformations



Other challenges: Noise, Blur, Compression, Artifacts, etc.

Figure from T. Tuytelaars ECCV 2006 tutorial

Invariant local features

Designed to be invariant to common geometric and photometric transformations.

Basic steps:

1) Detect distinctive **interest points**
2) Extract invariant **descriptors**

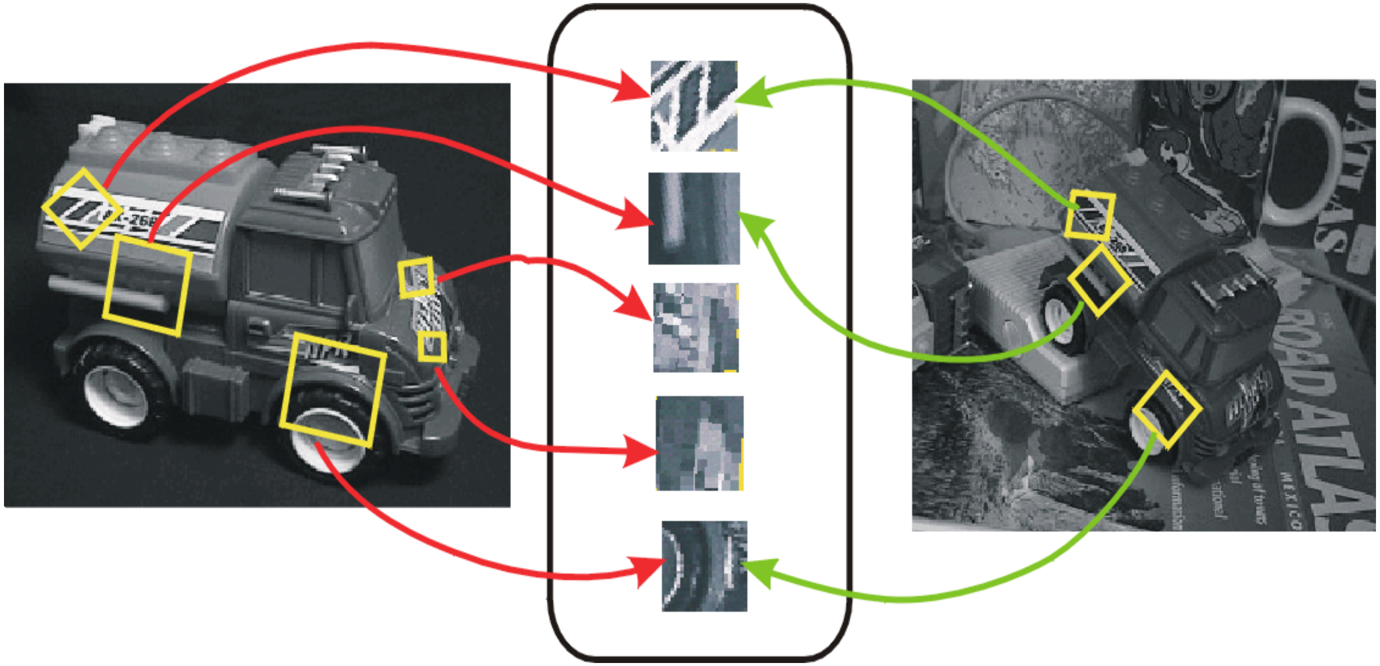


Figure: David Lowe

Main questions

- Where will the interest points come from?
- What are salient features that we'll *detect* in multiple views?
- How to *describe* a local region?
- How to establish *correspondences*, i.e., compute matches?

Feature Vectors – Image Templates

- How do we do image template matching (correlation)?
  - Given an image and a template, pass the template over the image
  - Find the max response over the image

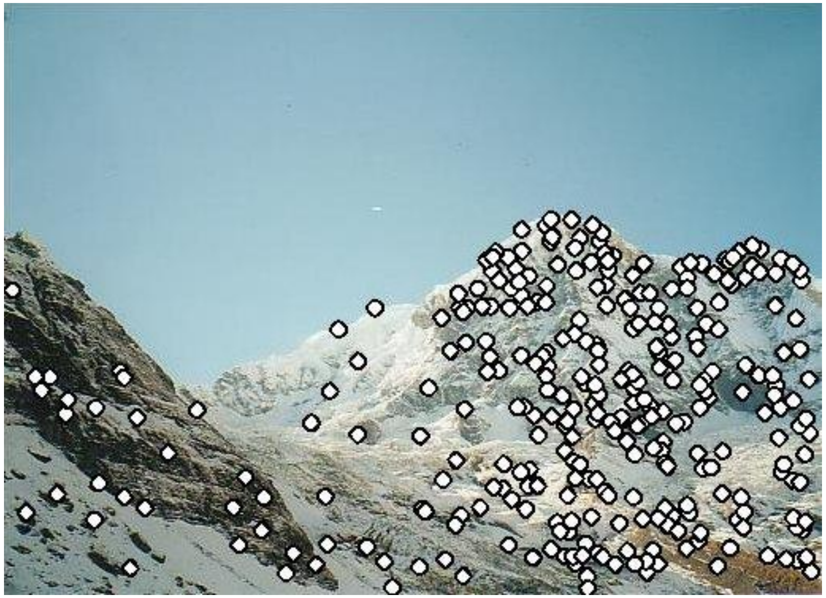
- What are the challenges using image templates?
  - Must match at all locations in image.
  - Must match each template separately
  - Does not generalize
    - Scale dependent, rotation sensitive, etc..

Image Templates – Improvements*

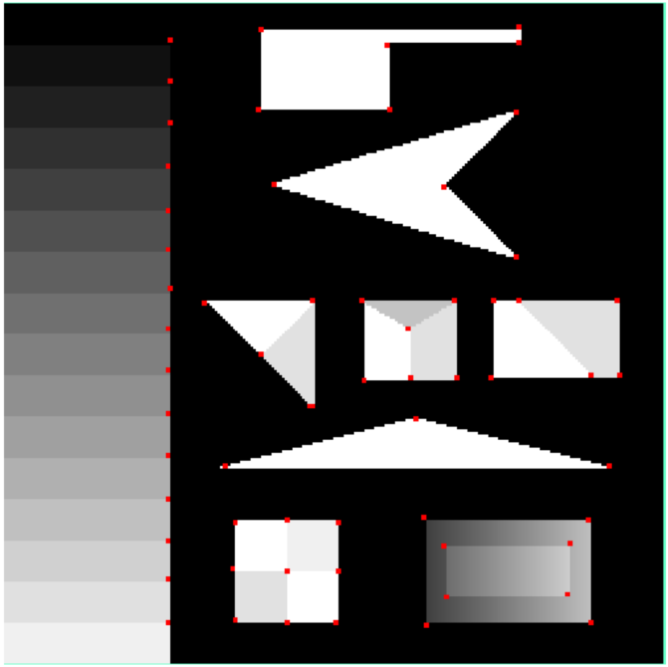
- Matching all locations?
  - Change the Stride of matching
- Match templates separately?
- Other Efficiency Issues
  - Extract and match other features
    - smaller feature vector
  - Efficient Image Computation
    - Integral Images
- Generalization?
  - Rescale and rotate the images and templates

Local features: Main components

1. Detection: Identify the **interest points.**
2. Description: Extract vector **feature descriptor** surrounding each interest point.
3. Matching: Determine **correspondence** between descriptors in two views.



Detection: Corners



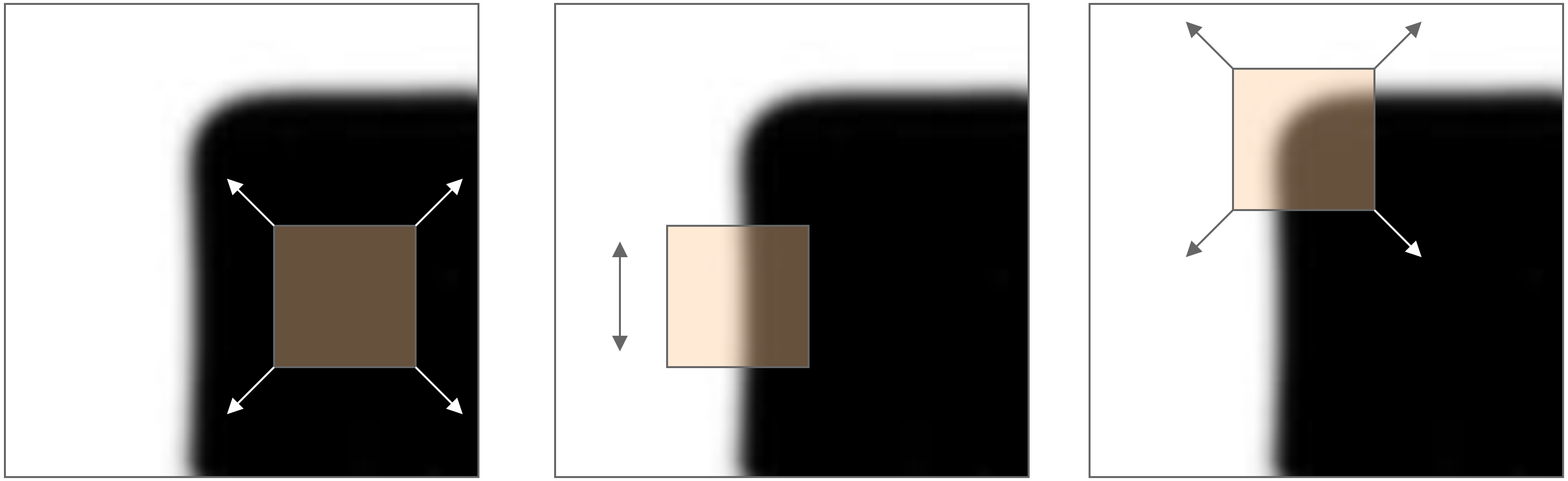
- Key property: in the region around a corner, image **gradient** has **two or more dominant directions**.
- Corners are **repeatable** and **distinctive.**

C.Harris and M.Stephens. "A Combined Corner and Edge Detector." *Proceedings of the 4th Alvey Vision Conference*: pages 147--151.

Lana Lazebnik

Corner Detection: Basic Idea

- Recognize the point by looking through a window.
- Shifting a window in ***any direction*** should give ***a large change* in intensity.**



"flat" region:
no change in all directions

"edge":
no change along the edge direction

"corner":
significant change in all directions

Source: A. Efros

Harris Detector

- Consider an image patch and shift it for $[\Delta x, \Delta y]$.
- *Sum of squared differences* (SSD) of two patches:

$$f(\Delta x, \Delta y) = \sum_{(x_k, y_k) \in W} (I(x_k, y_k) - I(x_k + \Delta x, y_k + \Delta y))^2$$

- $\mathrm{I}(\mathrm{x} + \Delta x, y + \Delta \mathrm{y})$ can be approximated by Taylor expansion. Let I_x, I_y be the partial derivative of I.

$$I(x + \Delta x, y + \Delta y) \approx I(x, y) + I_x(x, y)\Delta x + I_y(x, y)\Delta y$$

- This produces $f(\Delta x, \Delta y) \approx \sum_{(x,y) \in W} (I_x(x, y)\Delta x + I_y(x, y)\Delta y)^2$

$$f(\Delta x, \Delta y) \approx \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

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

Harris Detector

- The SSD becomes:

$$f(\Delta x, \Delta y) \approx \begin{pmatrix} \Delta x & \Delta y \end{pmatrix} M \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

where *M* is a 2×2 matrix with image derivatives:

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

Structure Tensor

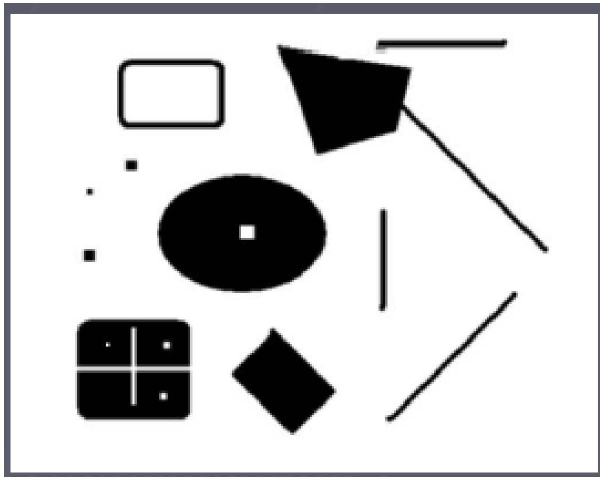
Sum over the patch

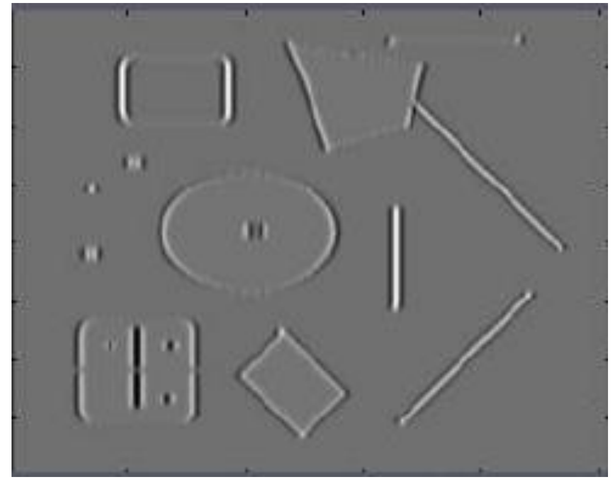
Gradient with respect to x,
times gradient with respect to y

- We sometimes add window weights to the SSD:

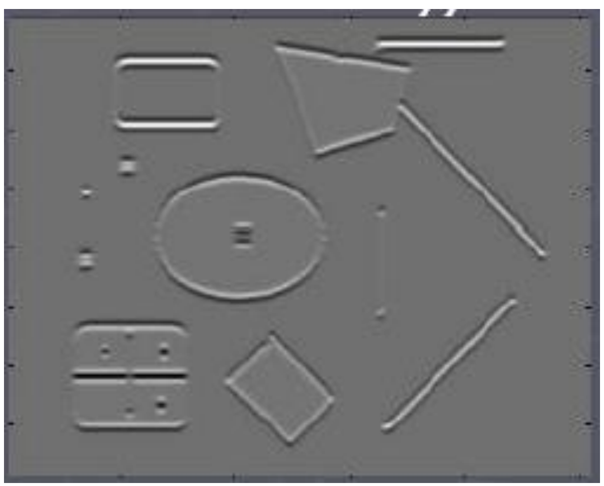
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Harris Detector

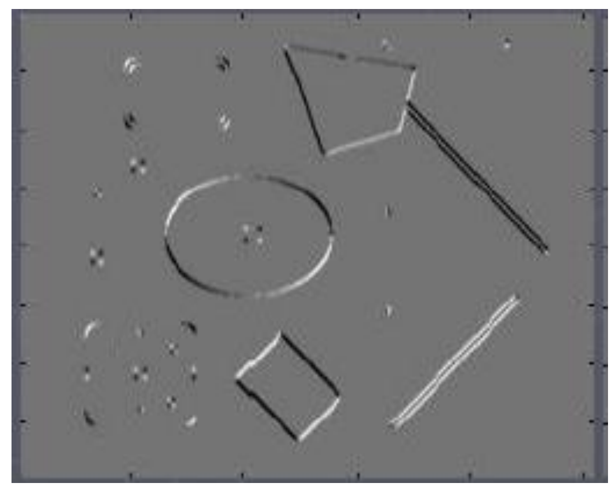




I_x



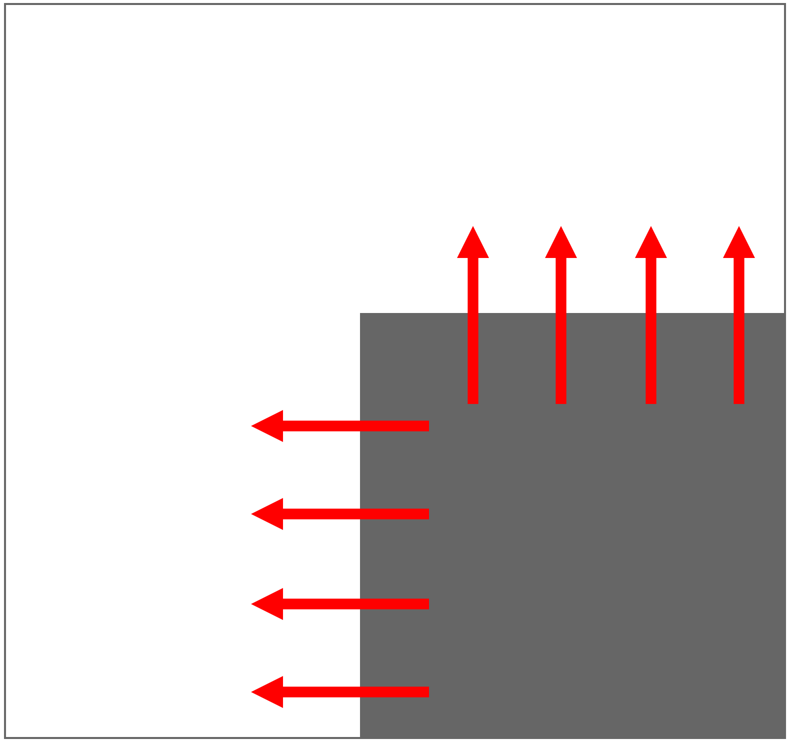
I_y



Harris

What does this matrix reveal?

- First, consider an axis-aligned corner:



What does this matrix reveal?

- First, consider an axis-aligned corner:

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

- Dominant gradient directions align with x or y axis
- If either λ is close to 0, then this is not a corner, so look for locations where both are large.
- What if we have a corner that is not aligned with the image axes?

General Case

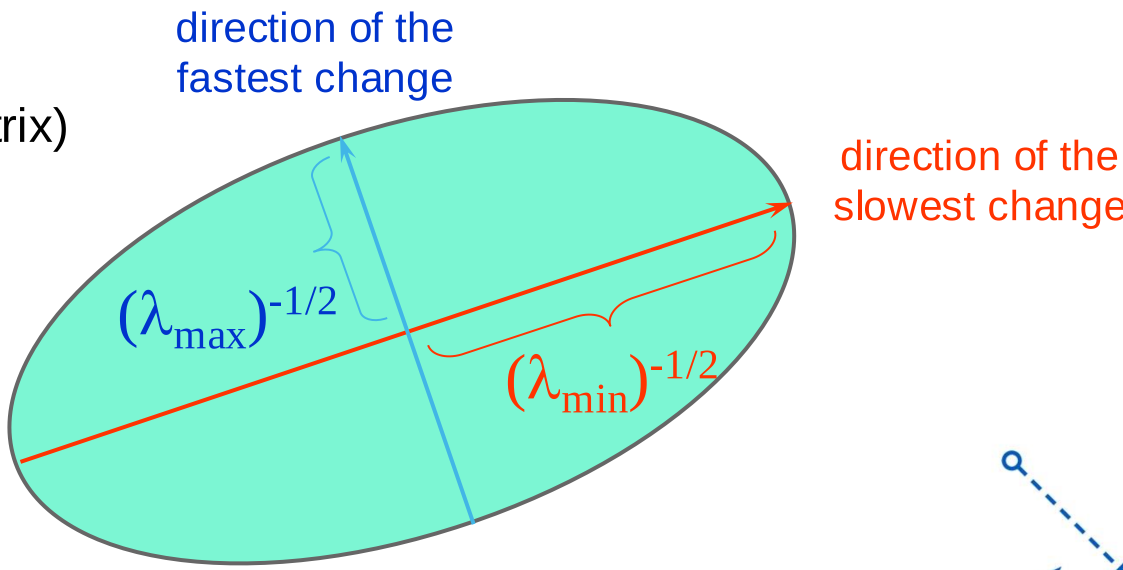
Since M is symmetric, we can use eigen-value decomposition

$$M = \mathbf{Q}^{-1}\begin{bmatrix}\lambda_1 & 0 \\ 0 & \lambda_2\end{bmatrix}\mathbf{Q}$$

We can visualize *M* as an ellipse with axis lengths determined by the eigenvalues and orientation determined by $\boldsymbol{Q}$ matrix.

$\boldsymbol{Q}$ is orthonormal (rotation matrix)

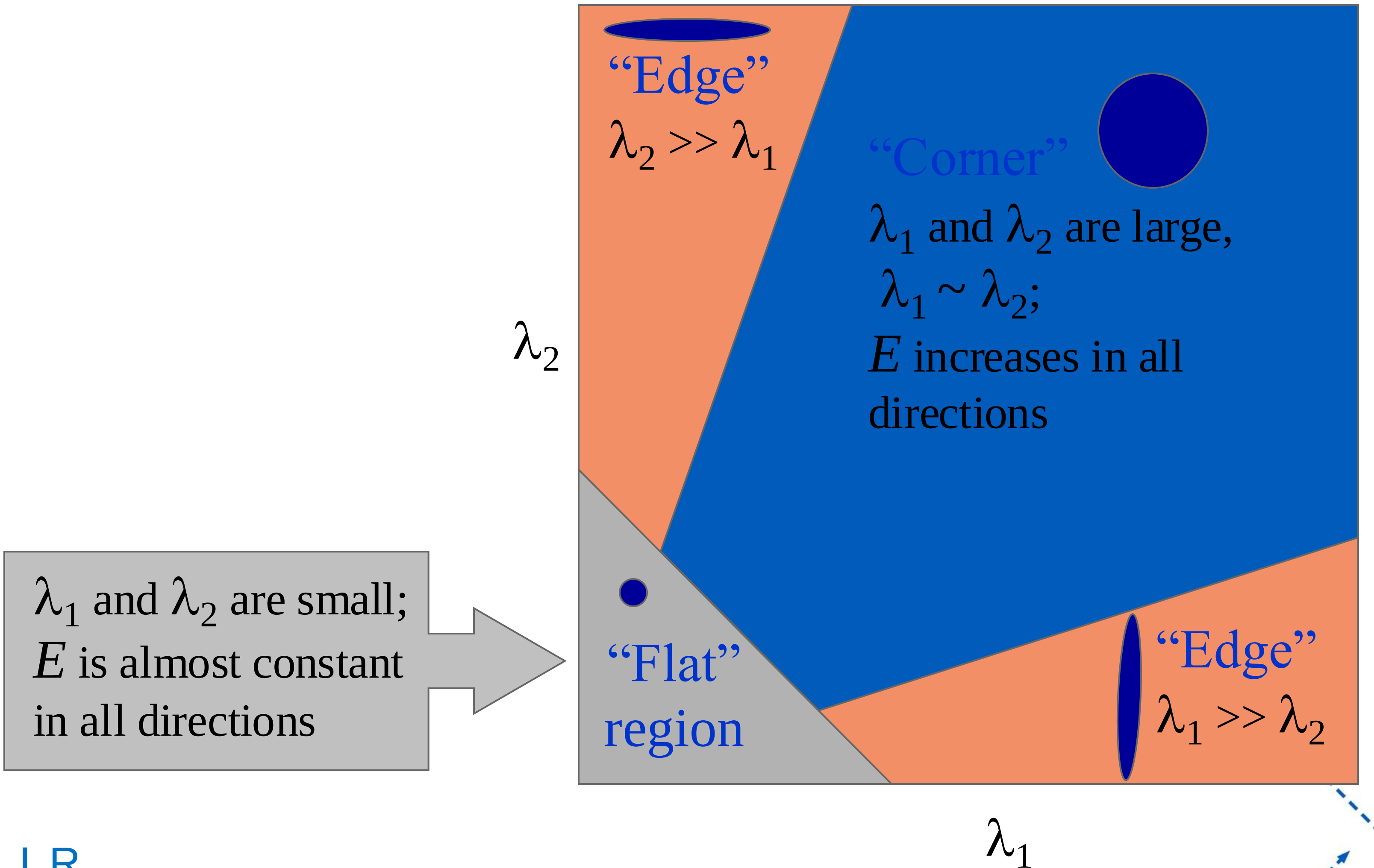
$\boldsymbol{Q}^{-1} = \boldsymbol{Q}^{\mathrm{T}}$.



Slide adapted form Darya Frolova, Denis Simakov.

Interpreting the eigenvalues

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



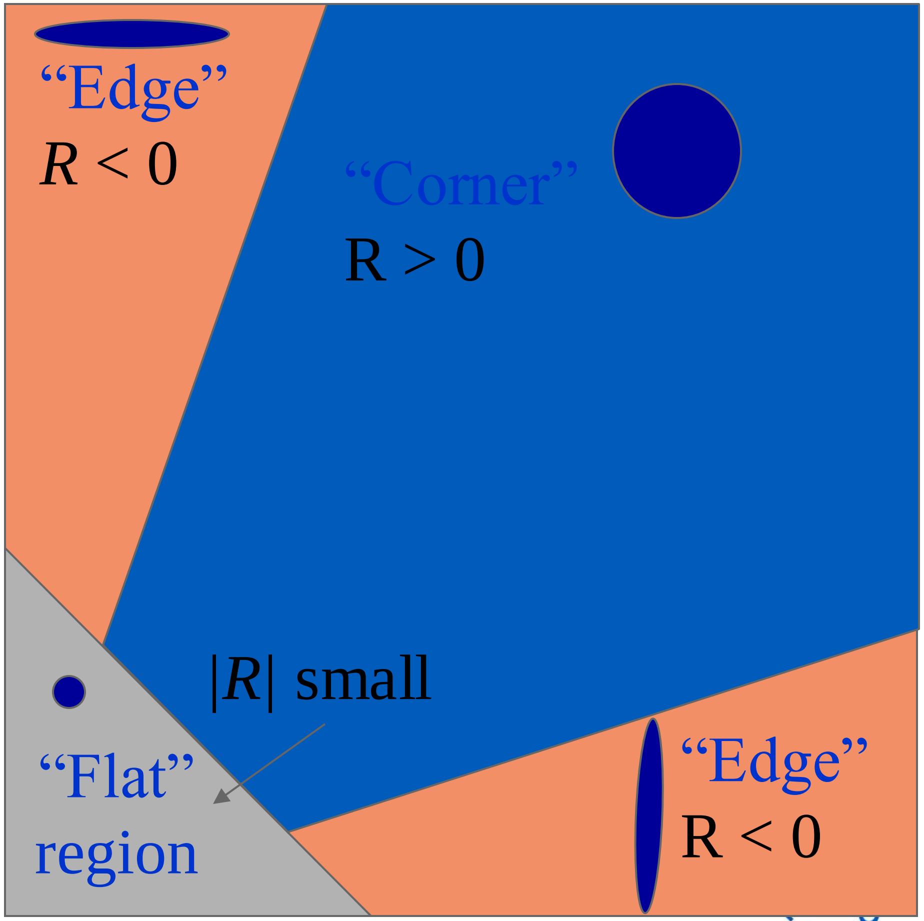
Harris Response Function

- The smallest eigenvalue of M:

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

- where α is an empirically determined constant.

$$\alpha \in [0.04\ to\ 0.06]$$



Harris corner detector: Algorithm

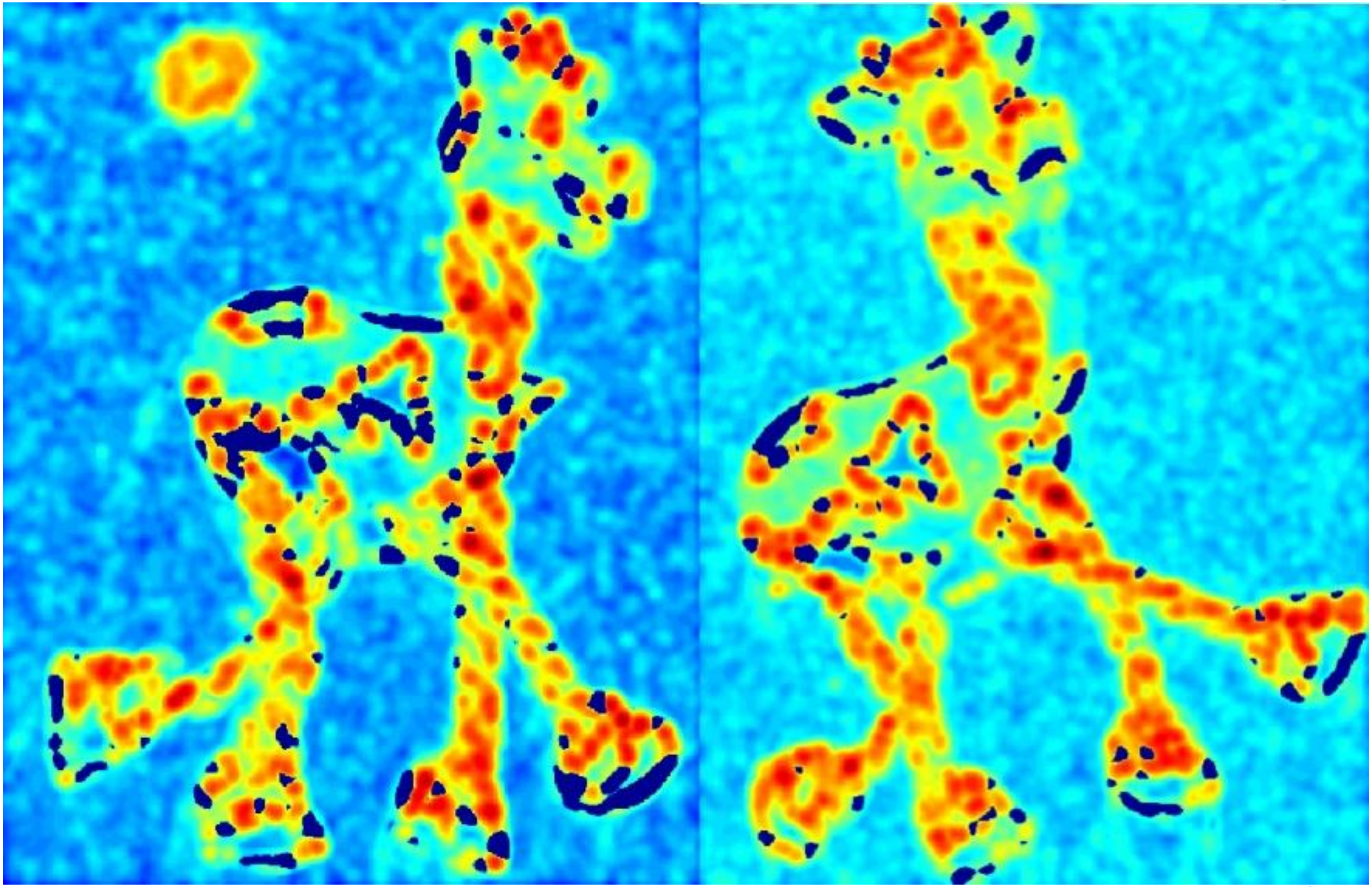
1. Color to grayscale
2. Calculate spatial derivative
3. Compute structure tensor for all windows
4. Harris response calculation
   - Find points with large corner response: (R > threshold)
5. Non-maximum suppression
   - Take the points of local maxima of R

C.Harris and M.Stephens. "A Combined Corner and Edge Detector."
Proceedings of the 4th Alvey Vision Conference: pages 147—151, 1988.

Harris Corner Detector

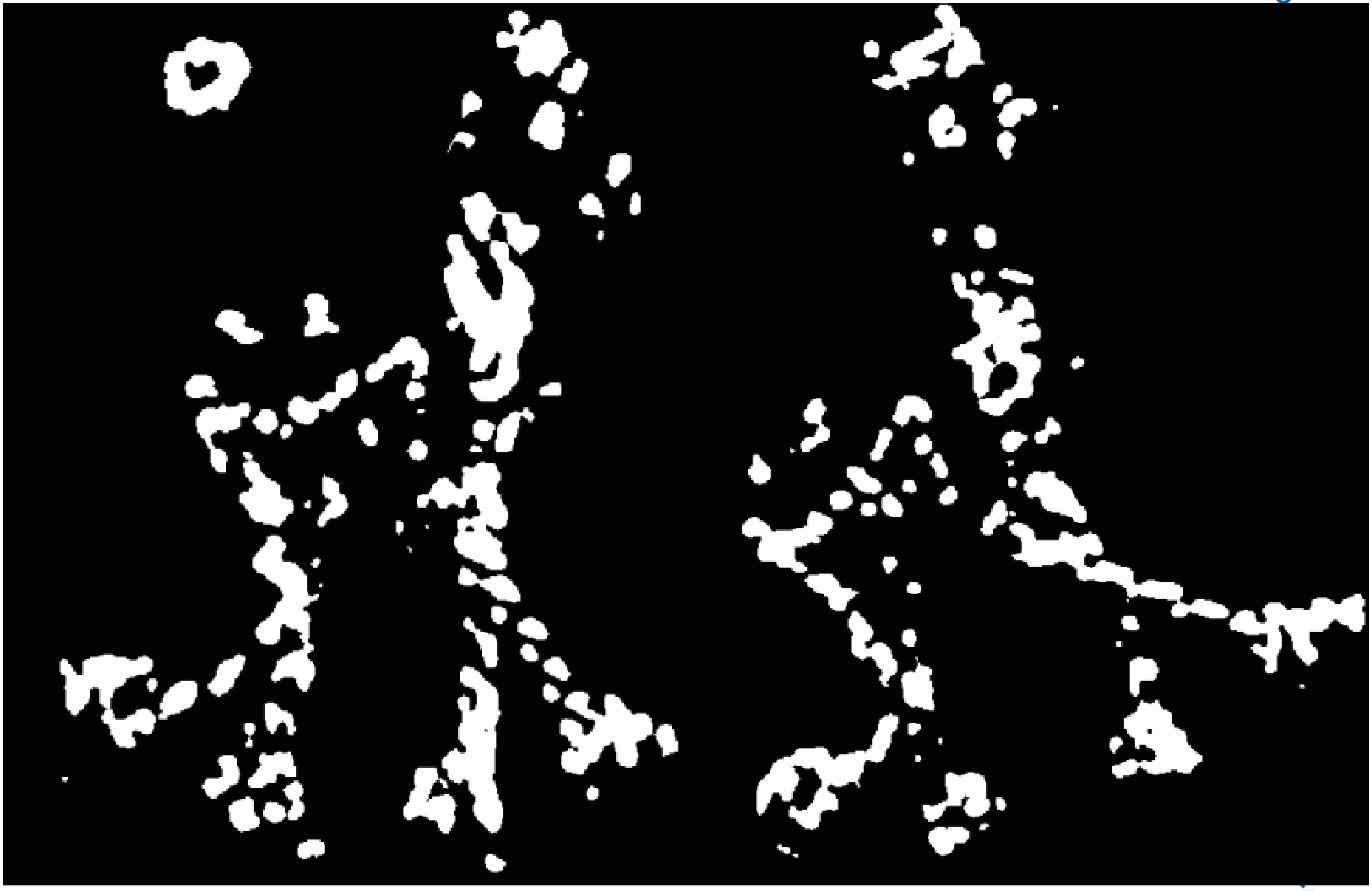
Slide adapted form Darya Frolova, Denis Simakov, Weizmann Institute.

Harris Corner detector



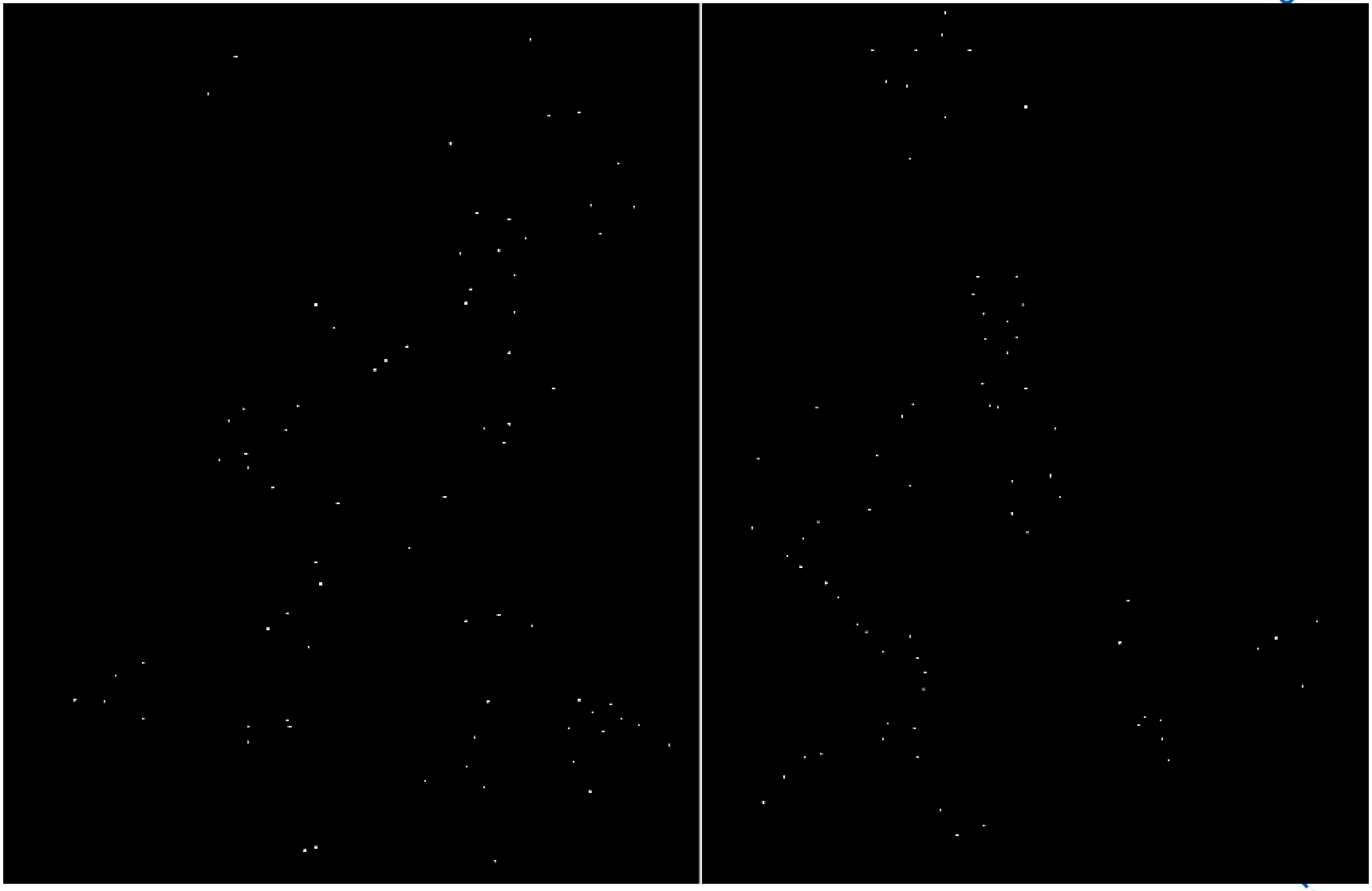
Corner response R

Harris Corner detector



Find points with large corner response: R>threshold

Harris Detector: Workflow



Take only the points of local maxima of R

Harris Corner

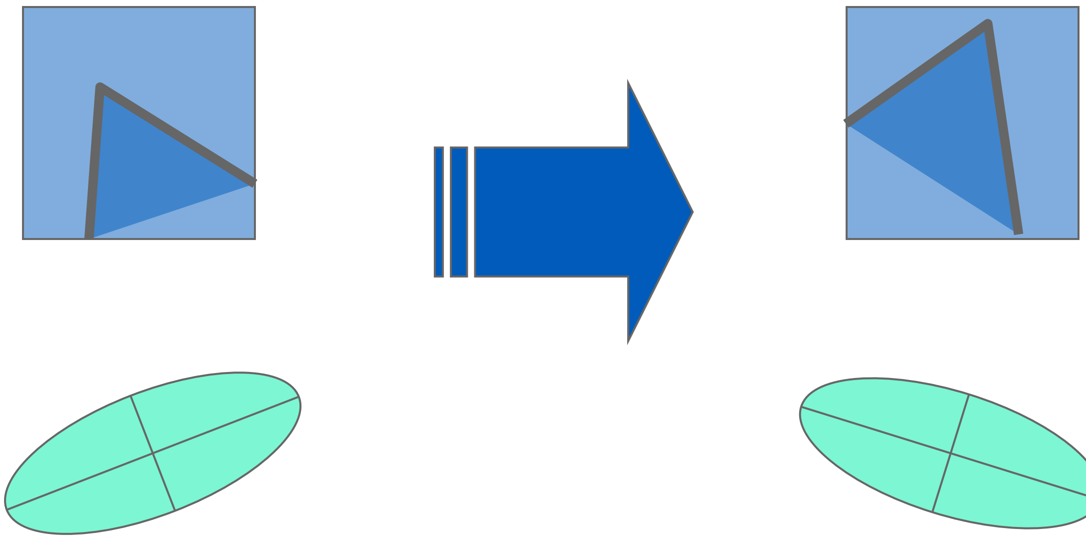
Image with Harris Corner Overlay.

Harris Corner

Original Image

Harris Detector: Properties

- Rotation invariance

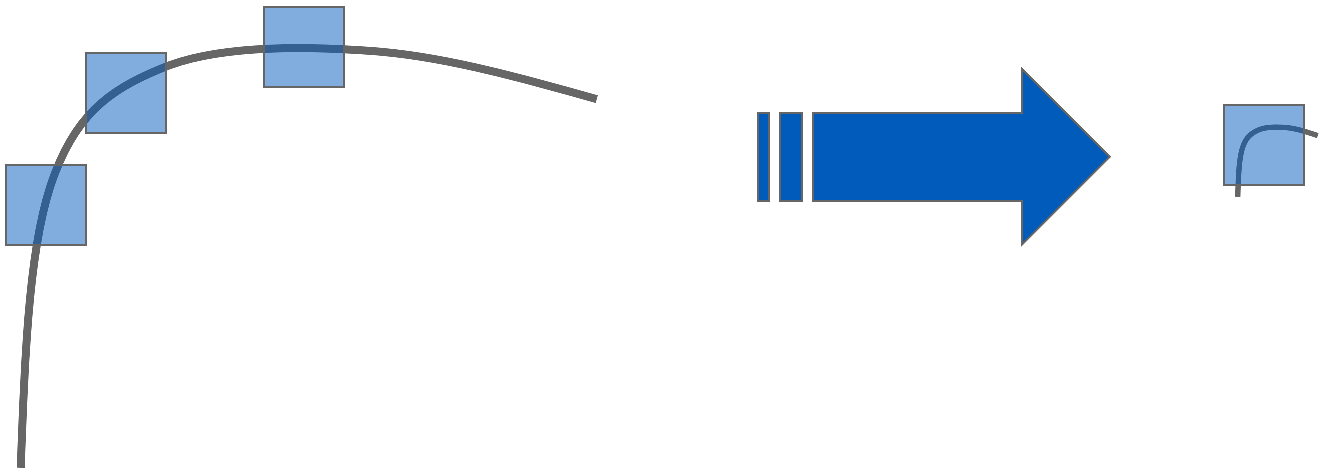


Ellipse rotates but its shape (eigenvalues) remains the same.

Corner response R is invariant to image rotation

Harris Detector: Properties

- Not invariant to image scale



All points will be classified as edges

Corner !

- How can we detect **scale invariant** interest points?
  - **Image Pyramids!**

Invariance and Equivariance

- We want corner locations to be invariant to photometric transformations and equivariant to geometric transformations
  - **Invariance:** same corners are detected with photometric transformations, such as histogram equalization.
  - **Equivariance** (Covariance): Corners are detected **in corresponding locations** in geometrically transformed image.

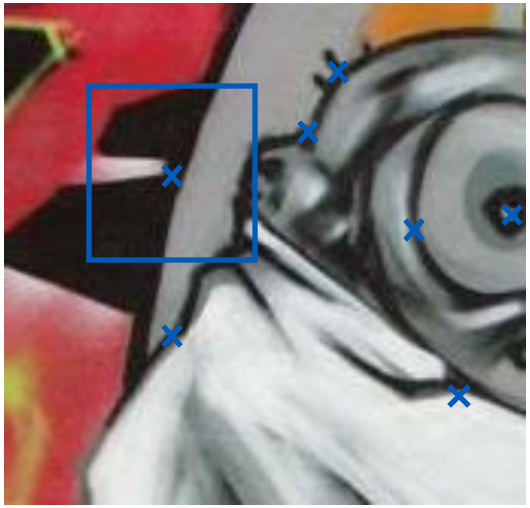
Scale Invariance

- Multi-scale approach (Image Pyramids)

Slide from T. Tuytelaars ECCV 2006 tutorial

Scale Invariance

- Multi-scale approach (Image Pyramids)

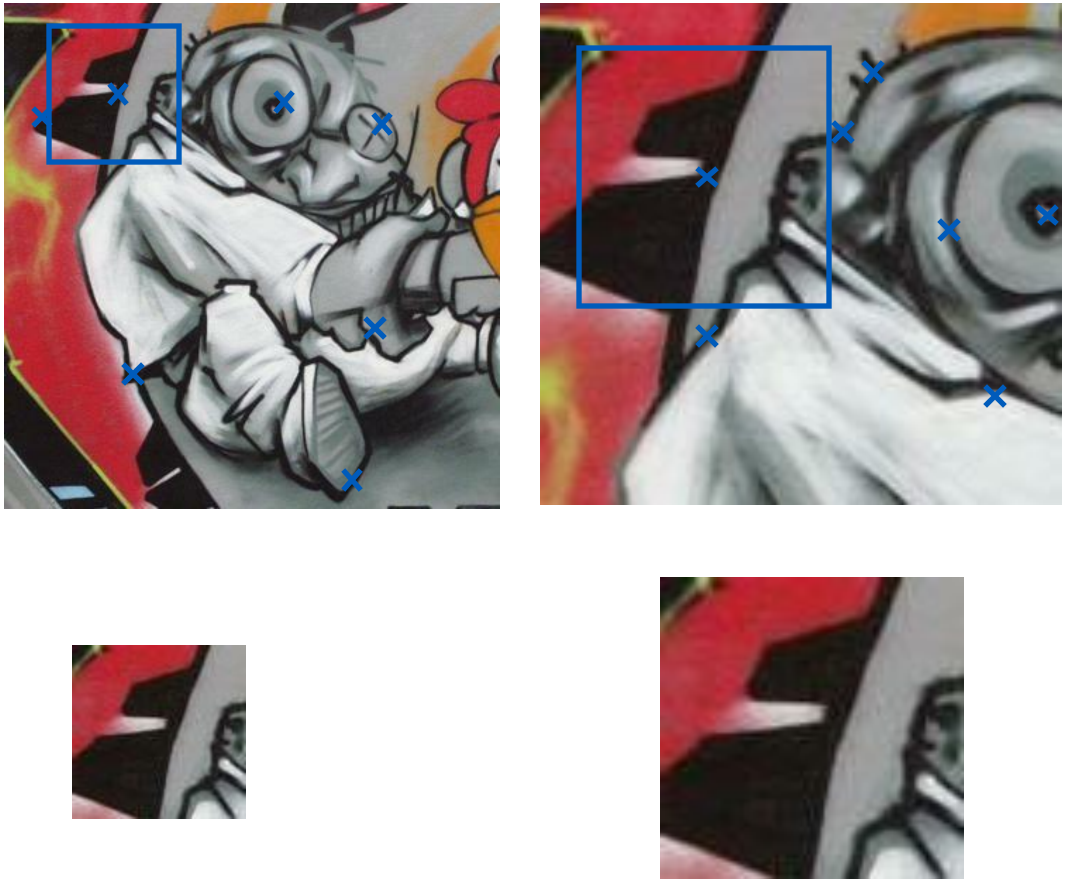


Scale Invariance

- Multi-scale approach (Image Pyramids)

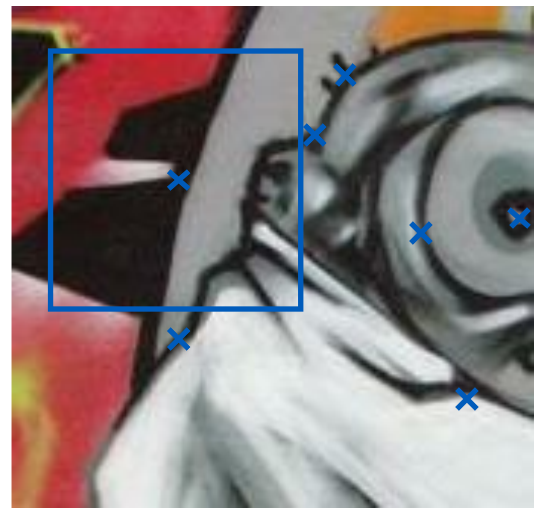
Scale Invariance

- Multi-scale approach (Image Pyramids)



Scale Invariance

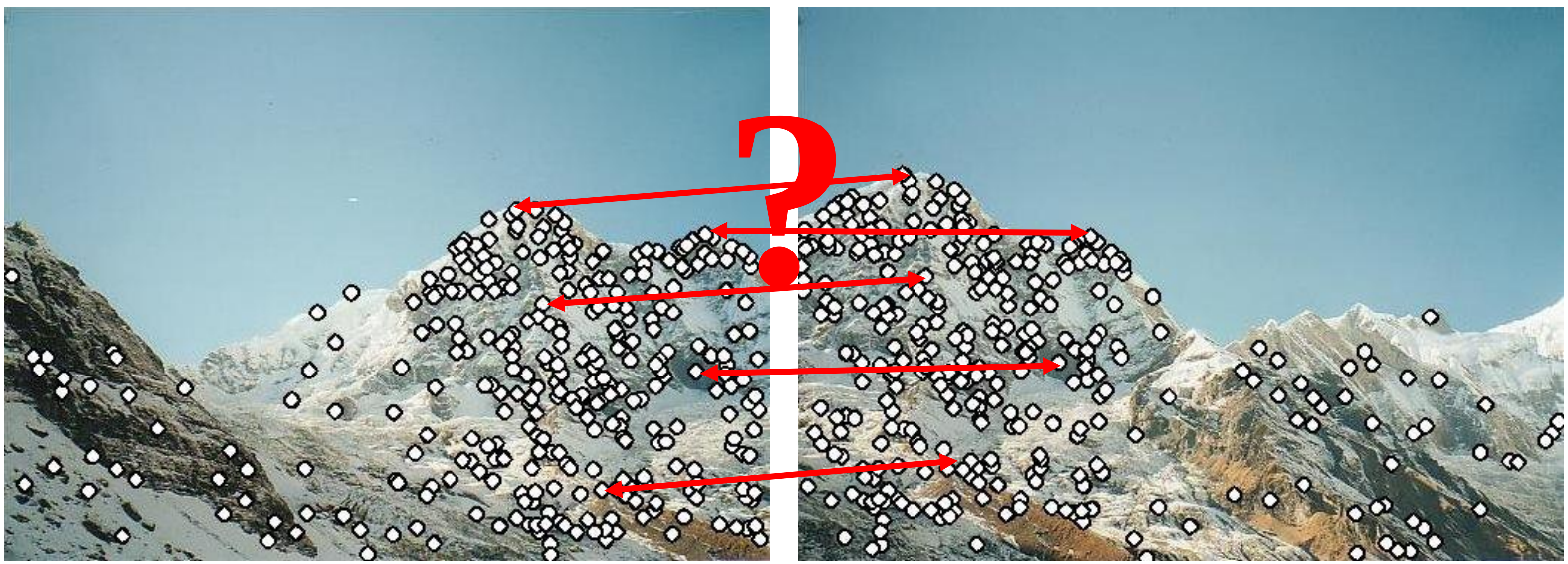
- Extract patch from each image individually

Local descriptors

- We know how to detect points
- Next question:

How to *describe* them for matching?



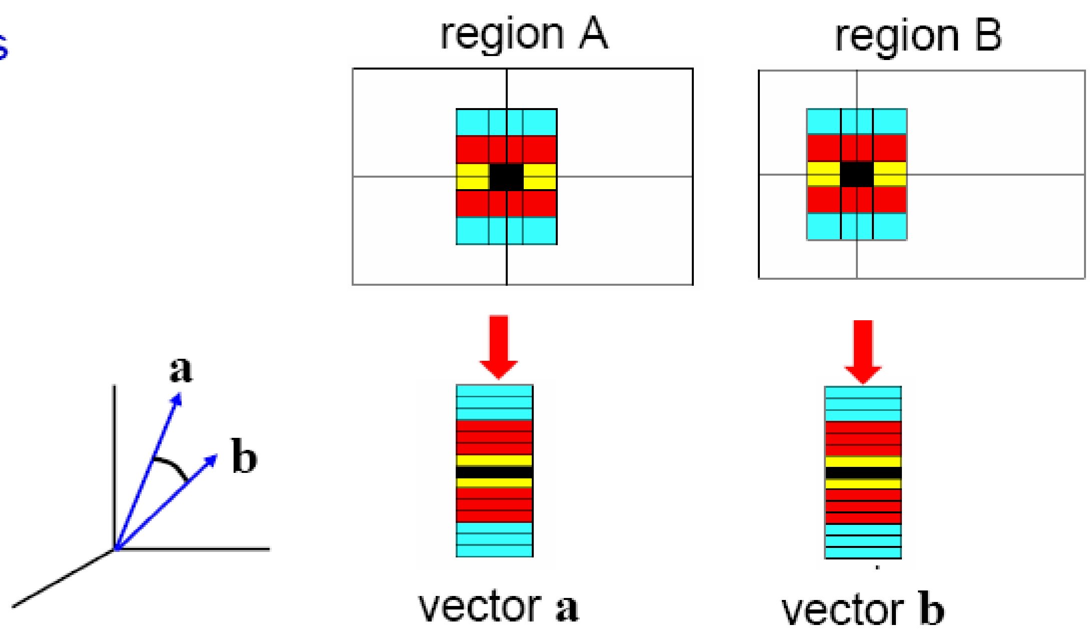
Point descriptor should be:
1. Invariant
2. Distinctive

Local descriptors

- Simplest: list of intensities within a patch.
- What is this going to be invariant to?
  - Translation

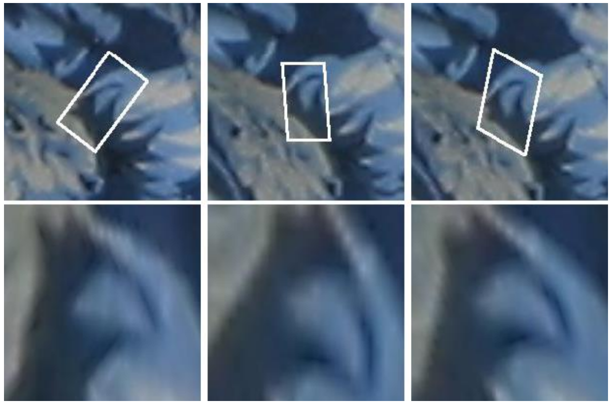
Write regions as vectors

$$A \rightarrow \mathbf{a}, \;\; B \rightarrow \mathbf{b}$$

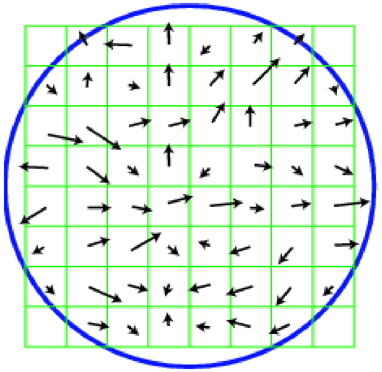


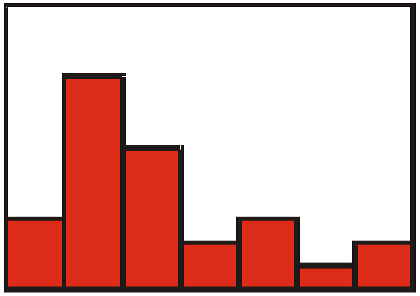
Feature descriptors

- Disadvantage of patches as descriptors:
  - Small shifts / brightness affect matching a lot.
  - What is invariant to pixel shuffle?

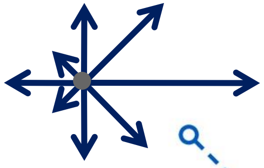


- Solution: histogram of oriented gradients (HOG).





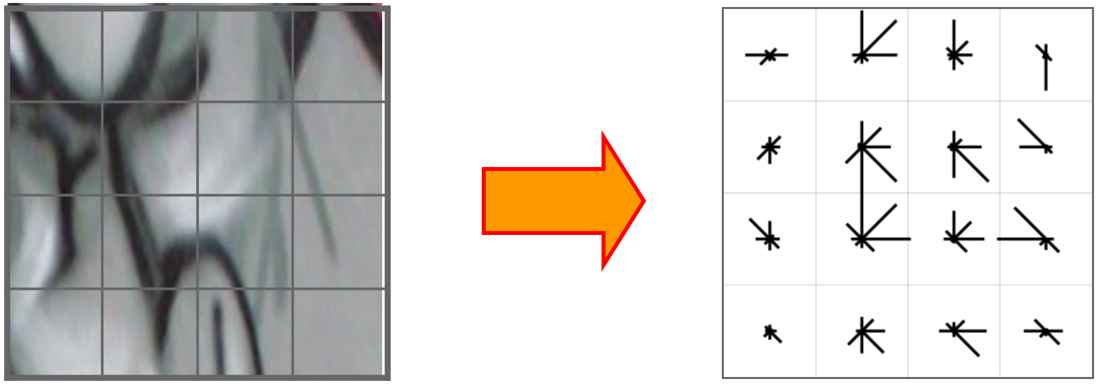
0 2 π



Lana Lazebnik

Feature descriptors: SIFT

- **S**cale **I**nvariant **F**eature **T**ransform
  - Divide patch into 4x4 sub-patches: 16 cells
  - Compute histogram of gradient orientations (8 reference angles) for all pixels in each sub-patch
  - Resulting descriptor: 4x4x8 = 128 dimensions



David G. Lowe. "Distinctive image features from scale-invariant keypoints." *IJCV* 60 (2), pp. 91-110, 2004.

Lana Lazebnik

Rotation Invariant Descriptors

Dominant direction of gradient for the image patch

- Find local orientation

- Rotate patch according to this angle
    - This puts the patches into a canonical orientation.

Rotation Invariant Descriptors

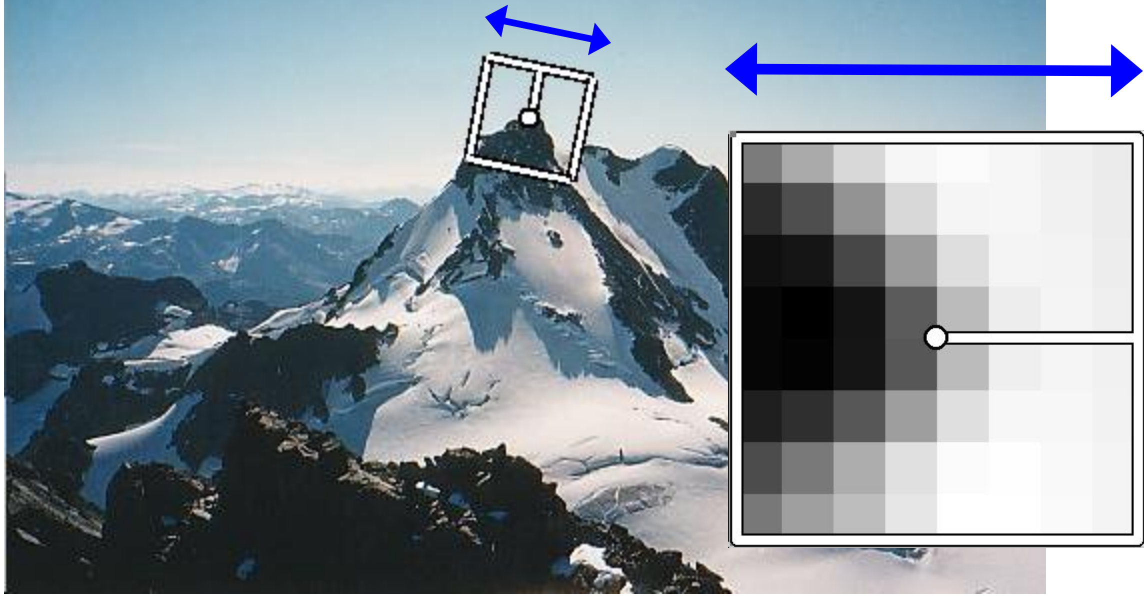


Image from Matthew Brown

Feature descriptors: SIFT

- Extraordinarily robust matching technique
  - Can handle changes in viewpoint
    - Up to about 60 degree out of plane rotation
  - Can handle significant changes in illumination
    - Sometimes even day vs. night
  - Fast and efficient—can run in real time.

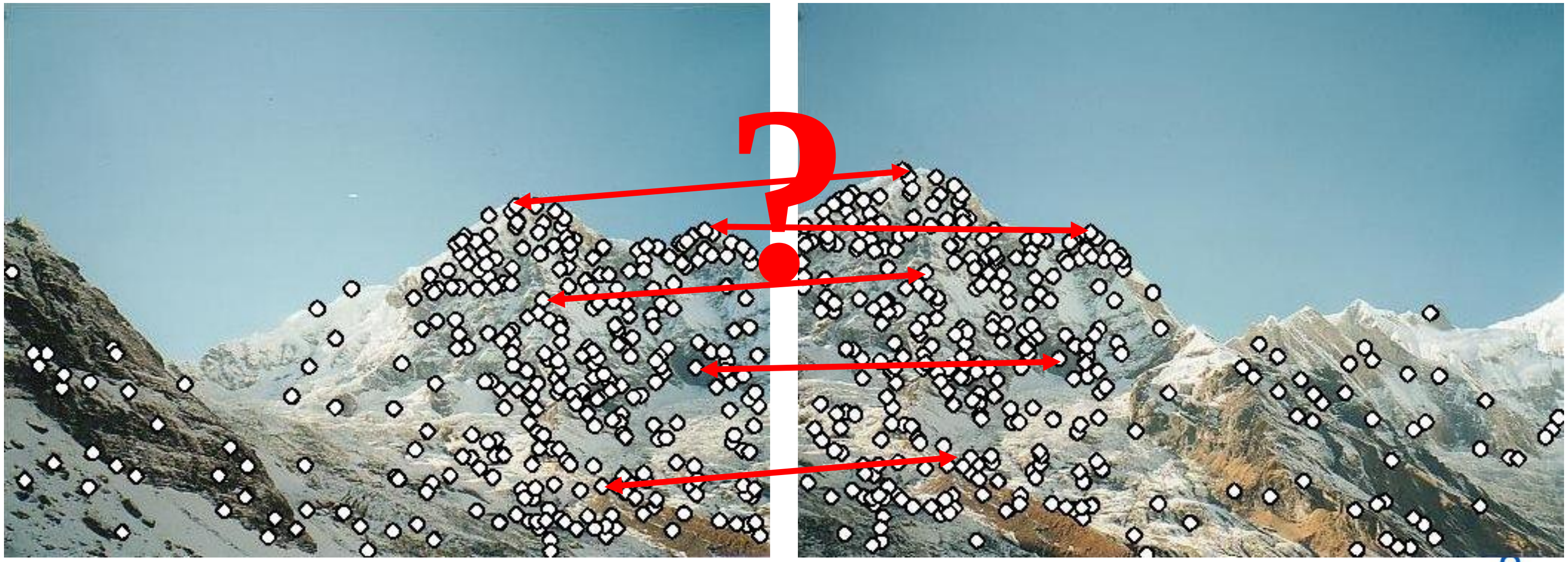
Working with SIFT descriptors

- One image yields:
  - n 128-dimensional descriptors:
    - each one is a histogram of the gradient orientations within a patch
    - [n x 128 matrix]
  - n scale parameters specifying the size of each patch
    - [n x 1 vector]
  - n orientation parameters specifying the angle of the patch
    - [n x 1 vector]
  - n 2d points giving positions of the patches
    - [n x 2 matrix]

Feature matching

We know how to detect **and describe** good points

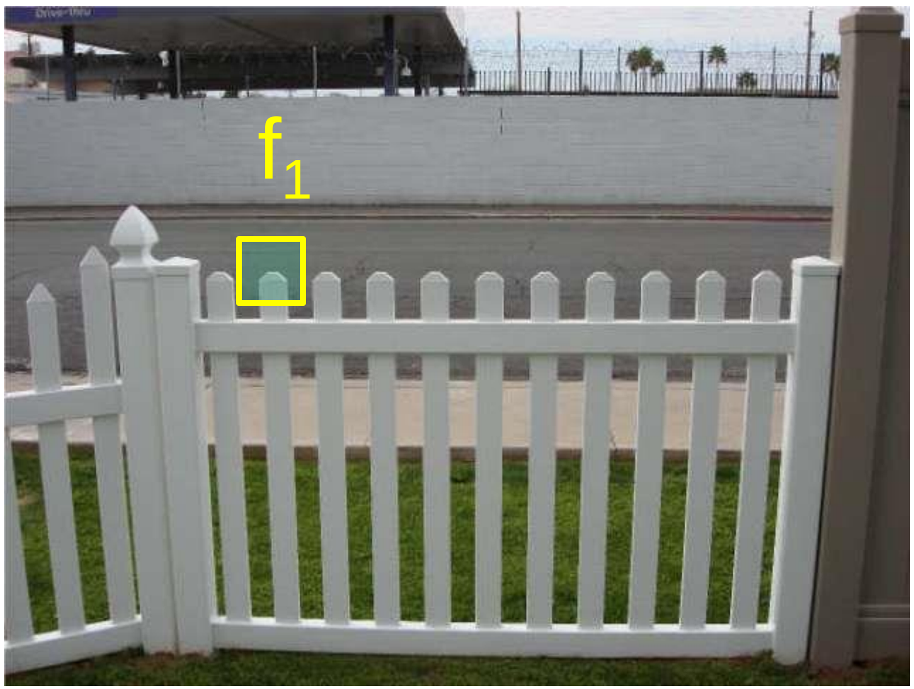
Next question: **How to match them?**

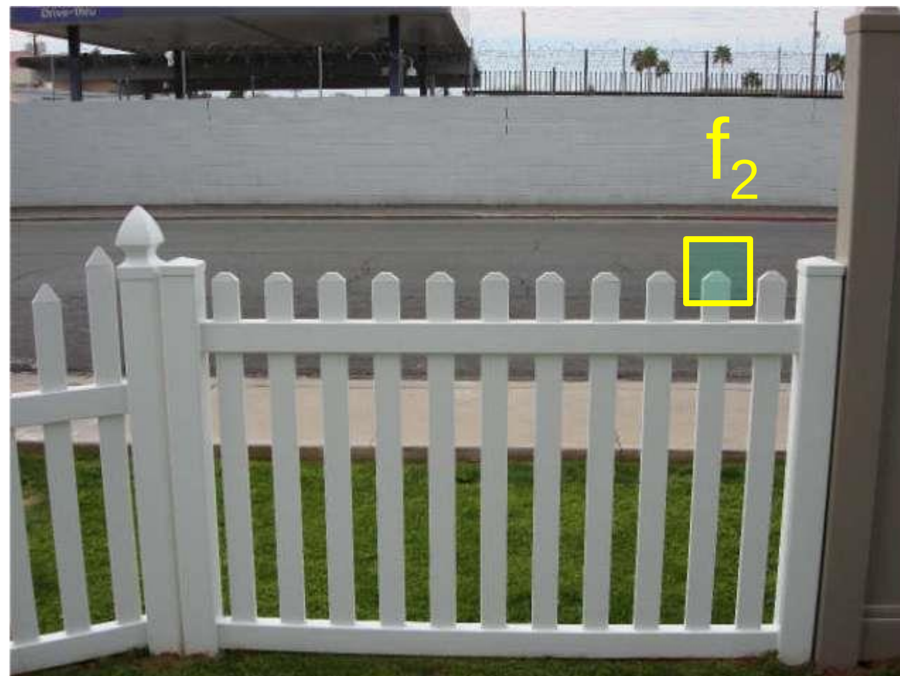


Feature distance

How to define difference of two features f_1, f_2?

- Simple approach is SSD(f_1, f_2)
  - Sum of Square Differences between two descriptors
  - can give good scores to ambiguous (bad) matches

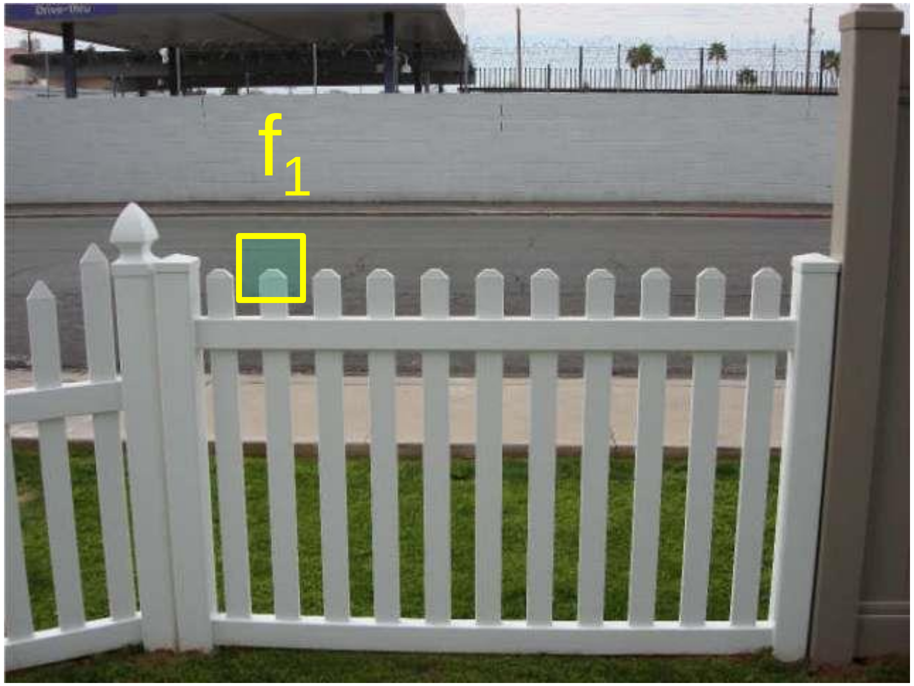


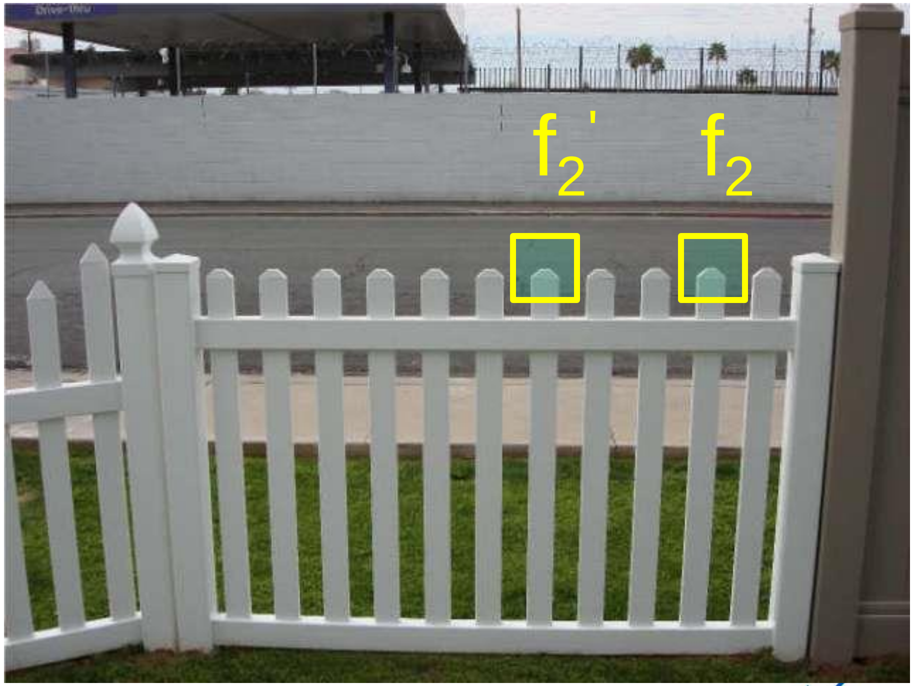


Feature distance

How to define difference between two features f_1, f_2?

- Better approach:
  - ratio distance = $SSD(f_1, f_2) / SSD(f_1, f_2')$.
  - f_2 is best SSD match to f_1 in I_2 , while f_2' is 2^{nd} best.
  - gives small values for ambiguous matches.





Feature Matching Summary

Given a feature in I_1, how to find the best match in I_2?

1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with min difference

- Simple approach is SSD(f_1, f_2)
  - sum of square differences between two descriptors
  - can give good scores to very ambiguous (bad) matches
- Better approach: ratio distance = SSD(f_1, f_2) / SSD(f_1, f_2')
  - f_2 is best SSD match to f_1 in I_2
  - f_2' is 2^{nd} best SSD match to f_1 in I_2
  - gives small values for ambiguous matches

Other Distance Measures

- Sum of Squared differences (SSD)

$$distance(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

- Correlation

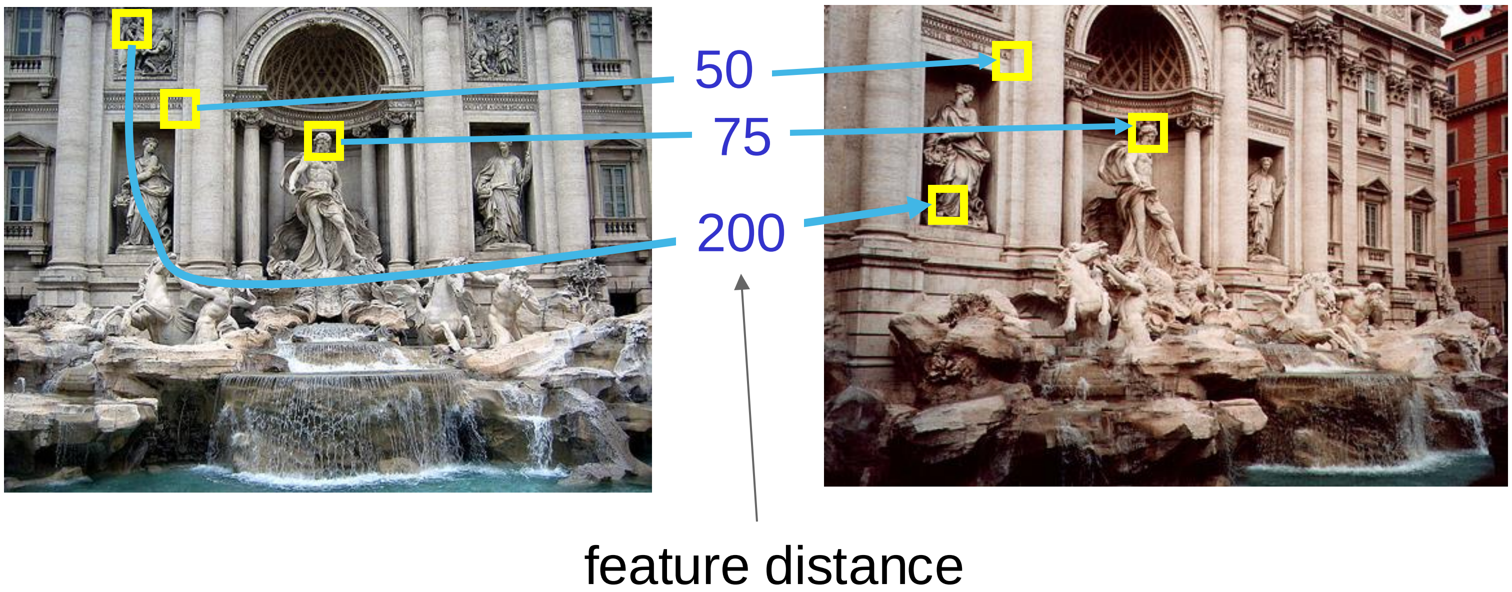
$$\rho_{X,Y} = \mathrm{corr}(X,Y) = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{\mathrm{E}[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

- Cosine Similarity

$$\text{cosine similarity} = S_C(A,B) := \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

Evaluation of matches

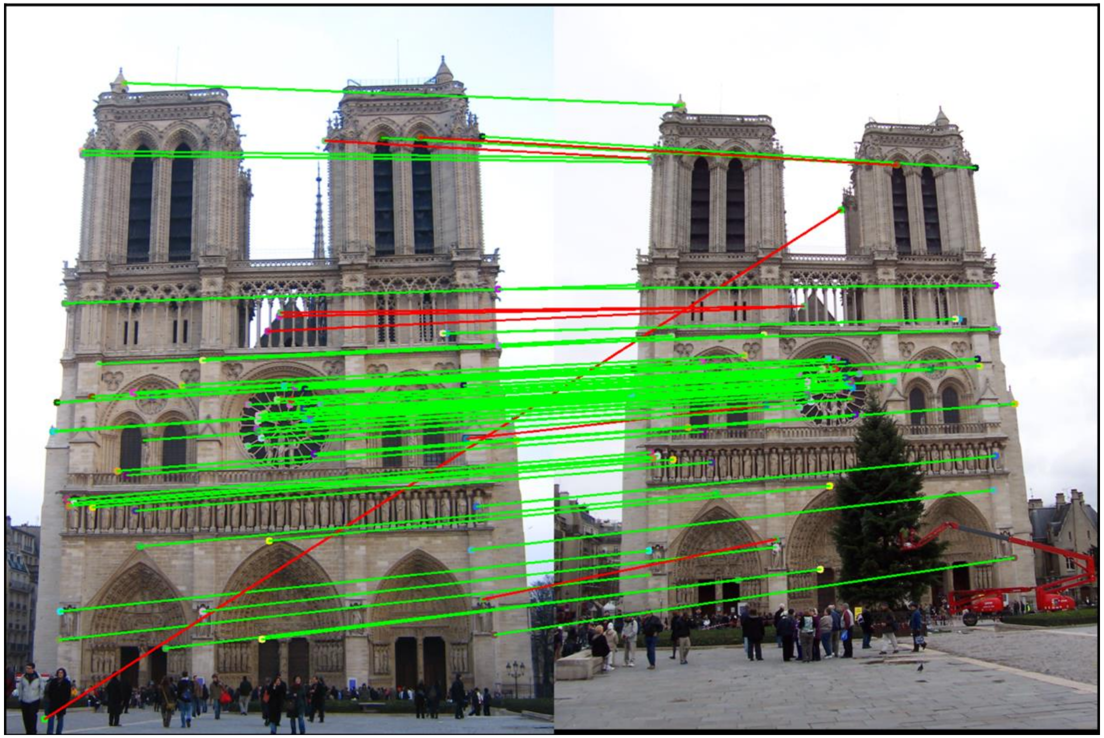
How can we measure the performance of a feature matcher?



True/false positives

The distance threshold affects performance

- True positives = # of detected correct matches.
- False positives = # of detected incorrect matches.

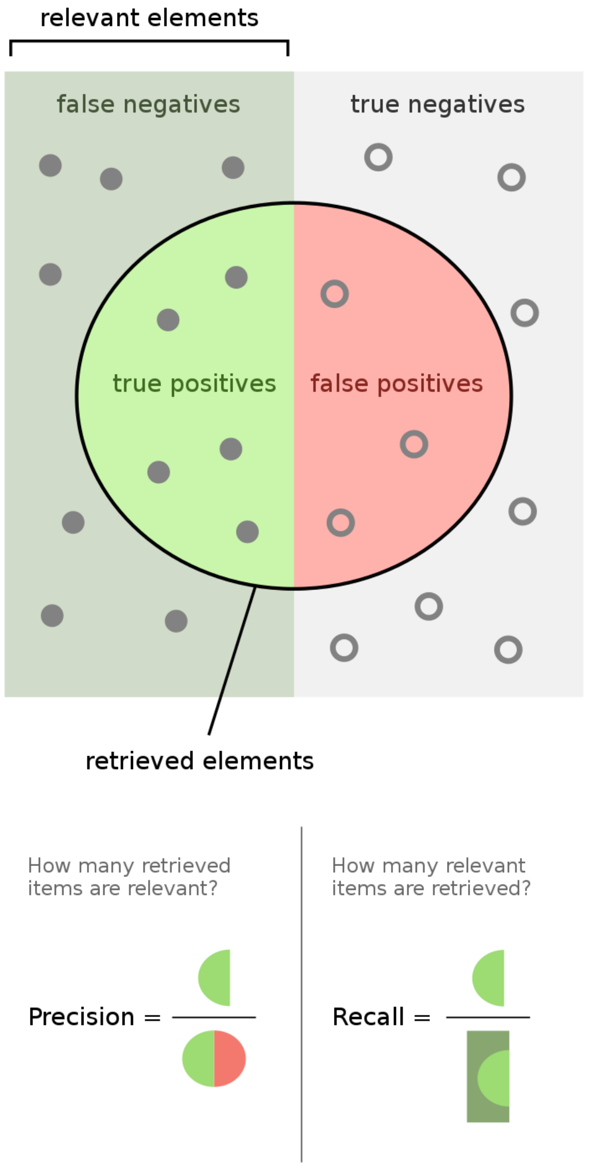


Precision, Recall, F1

$$\text{Precision} = \frac{\text{True Positive}}{\text{Predicted Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{Actual Positive}}$$

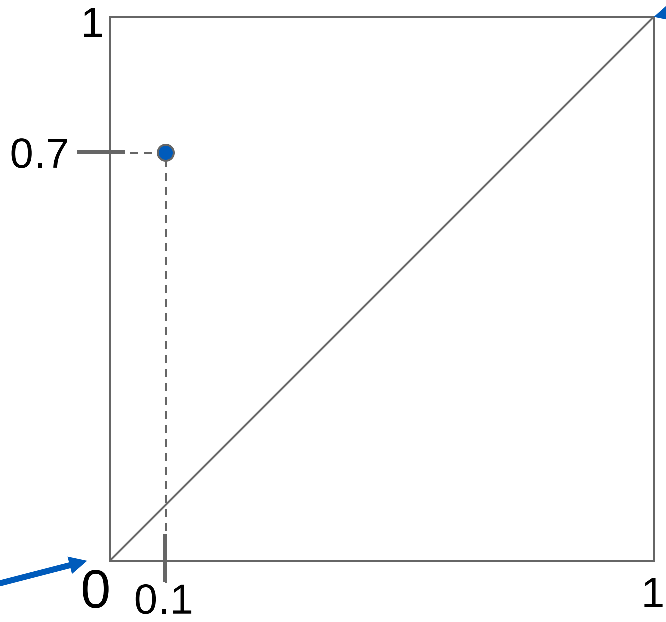
$$F_1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

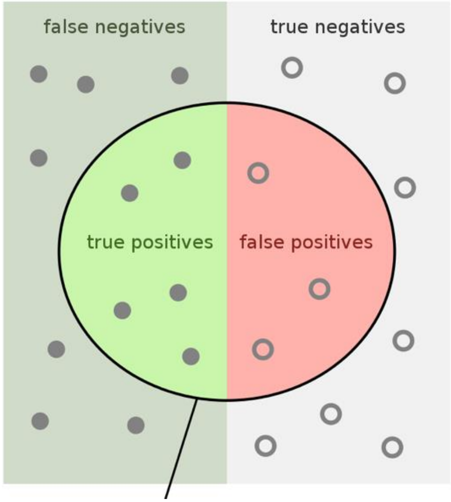


Evaluation of matches

- How can we measure the performance of a feature matcher?

Lowest threshold
all pairs as match





True Positive Rate $\frac{\# TP}{\# TP + FN}$

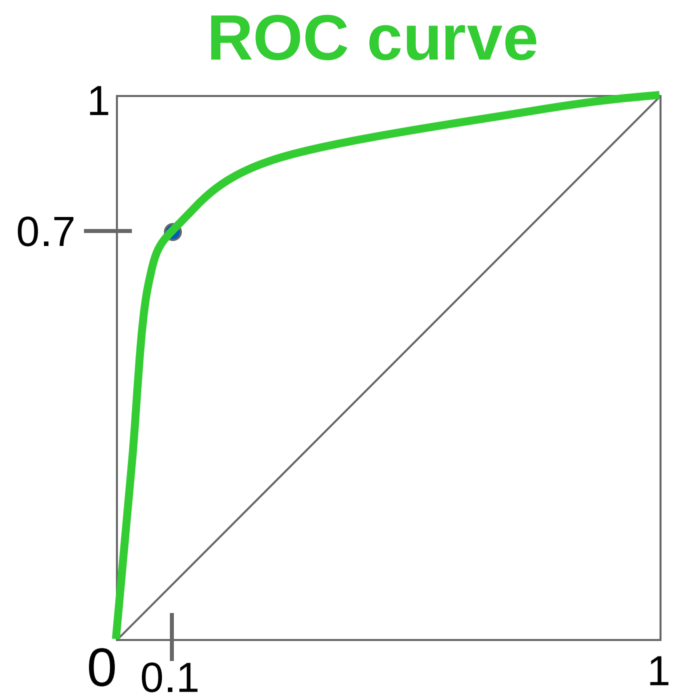
Highest threshold
no match

False Positive Rate $\frac{\# FP}{\# FP + TN}$

Evaluation of matches

ROC Curves (Receiver Operator Characteristic)

- Generated by counting # correct/incorrect matches, for different thresholds.
- Want to maximize area under the curve (AUC)
- Useful for comparing different feature matching methods.

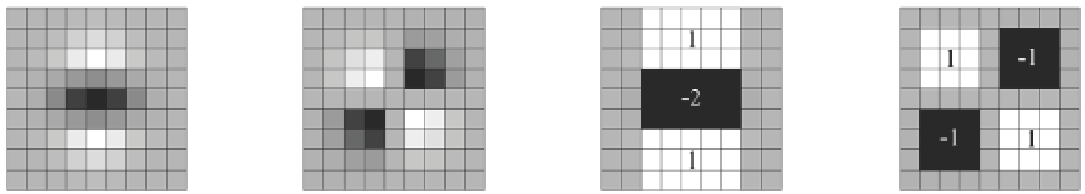


true positive rate $\dfrac{\#\,TP}{\#\,TP + FN}$

false positive rate $\dfrac{\#\,FP}{\#\,FP + TN}$

Other Local Descriptors: SURF

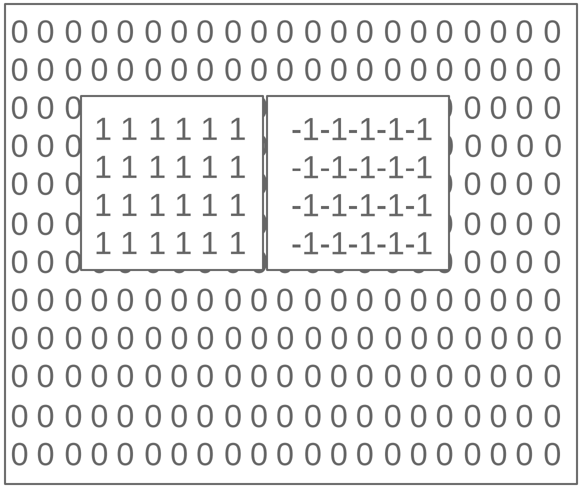
- Speeded up robust features (SURF)
  - Fast approximation of SIFT, 6 times faster.
- Accelerated by 2D filters (Harris) & **integral images.**



Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." *ECCV* (2006): 404-417.

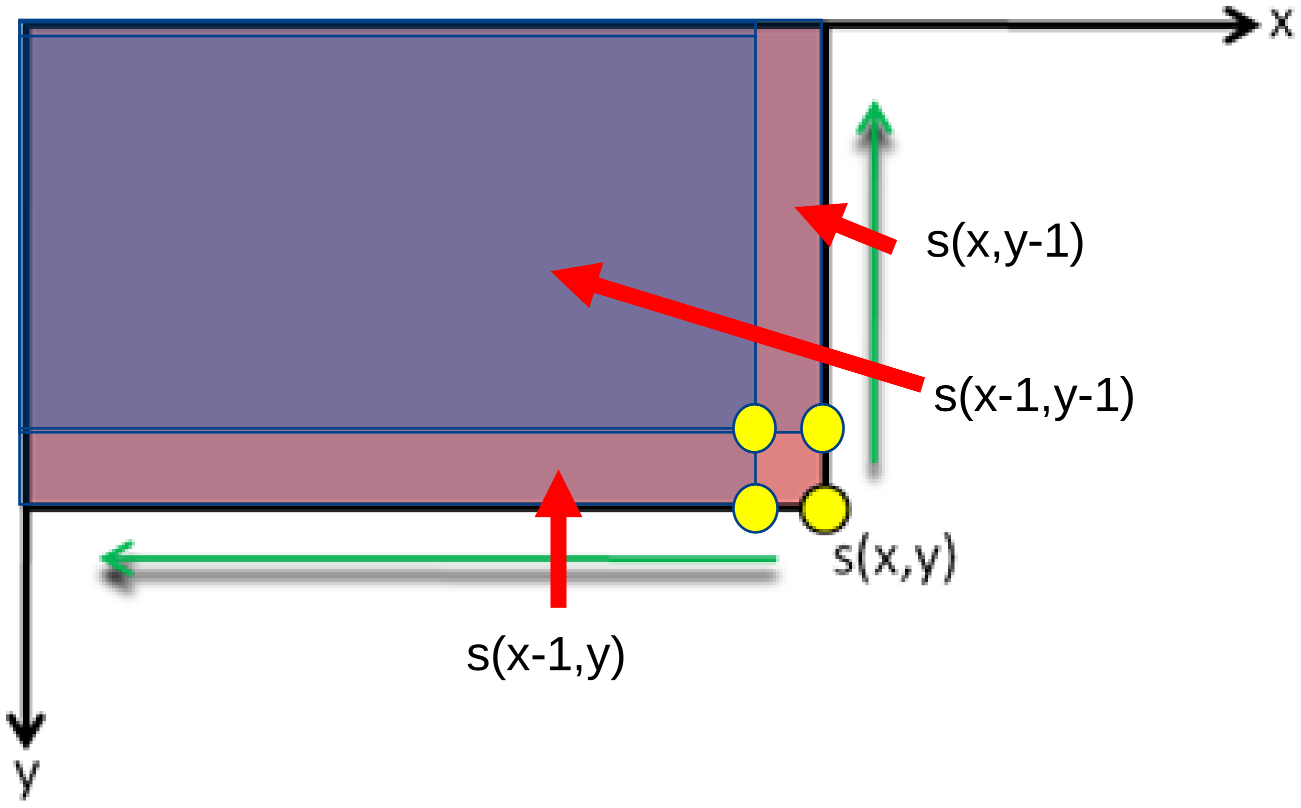
Integral Image: Motivation

- How can we make things faster?
- Many of the filters we have defined are made of rectangles or combinations of rectangles!



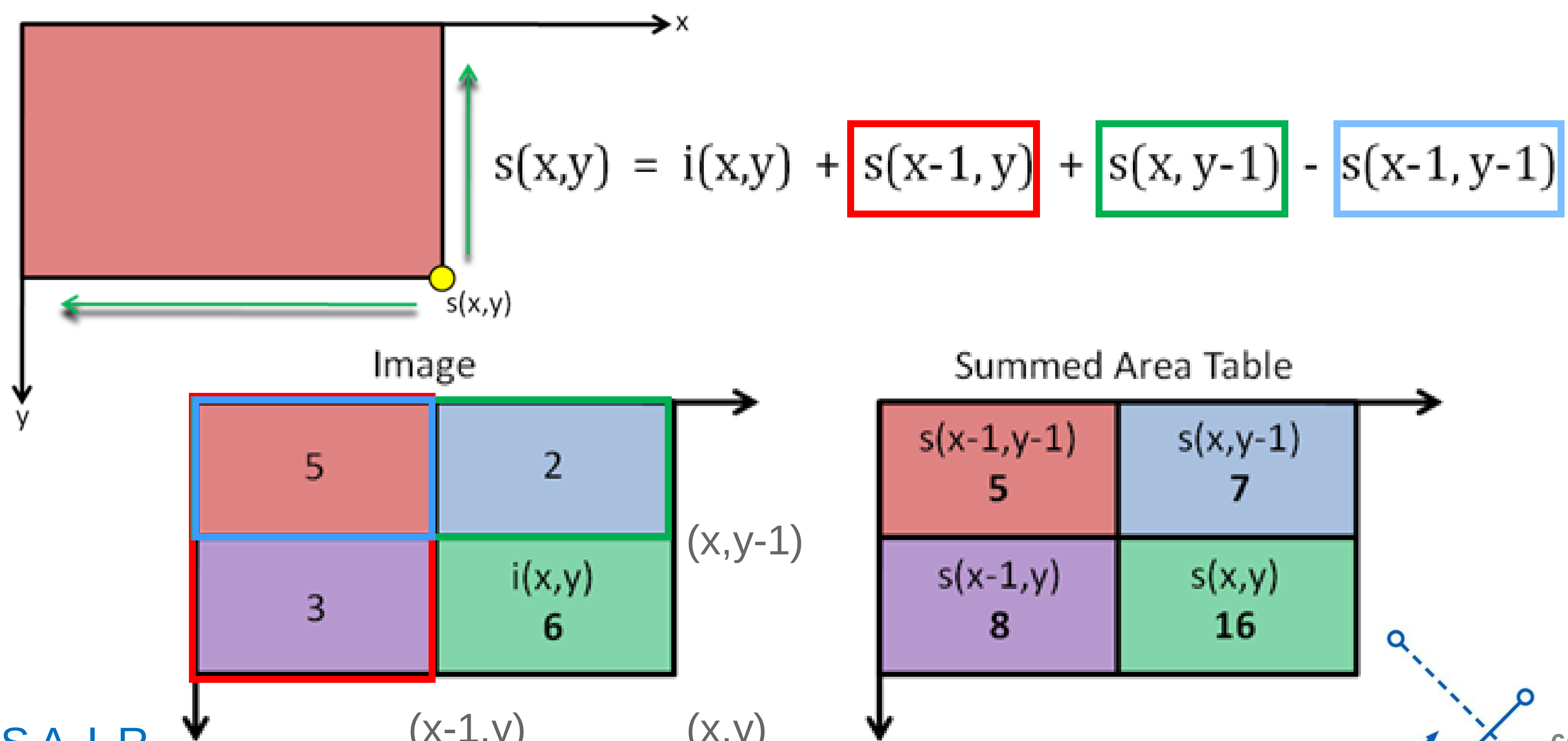
Integral Image

- A transformed image where every pixel is the sum of all pixels **above** and to the **left** of original image.



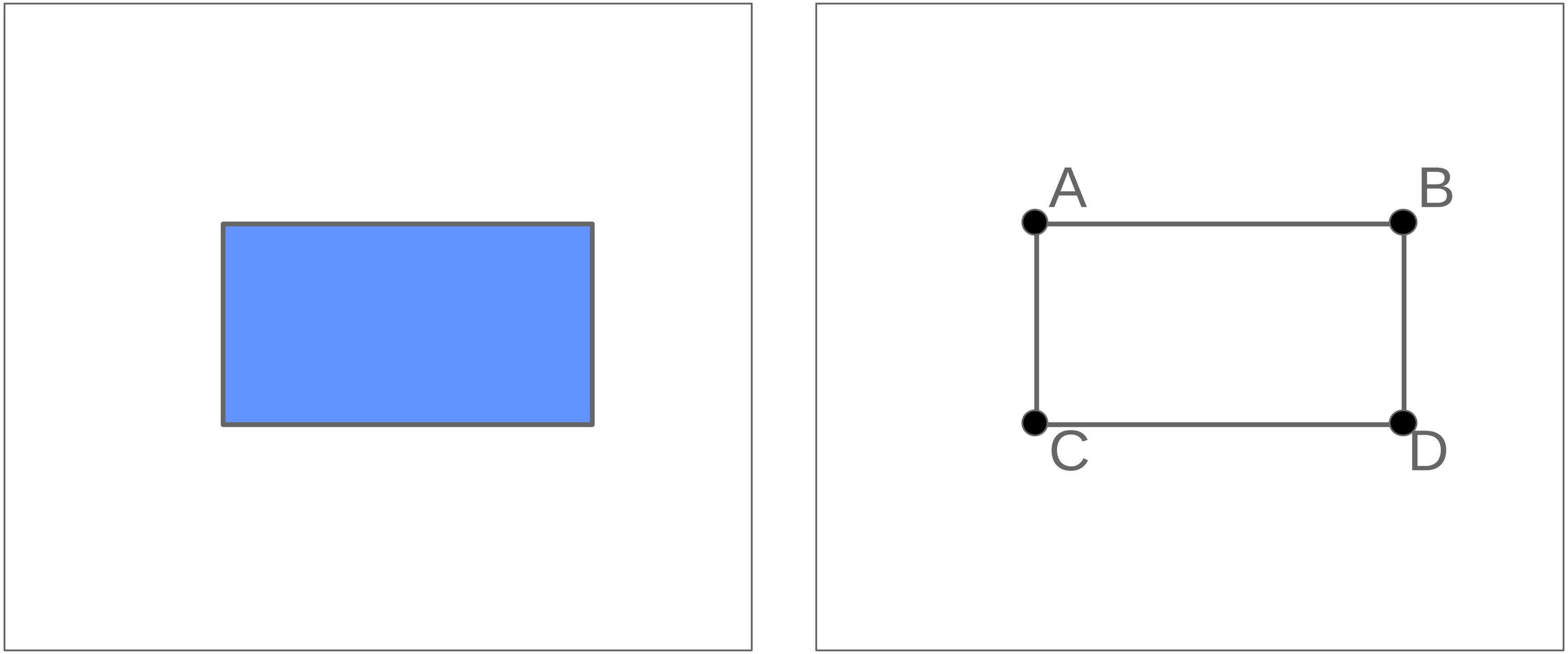
Integral image

- a quick and effective way of calculating the sum of values (pixel values)



SAIR
Spatial AI & Robotics Lab

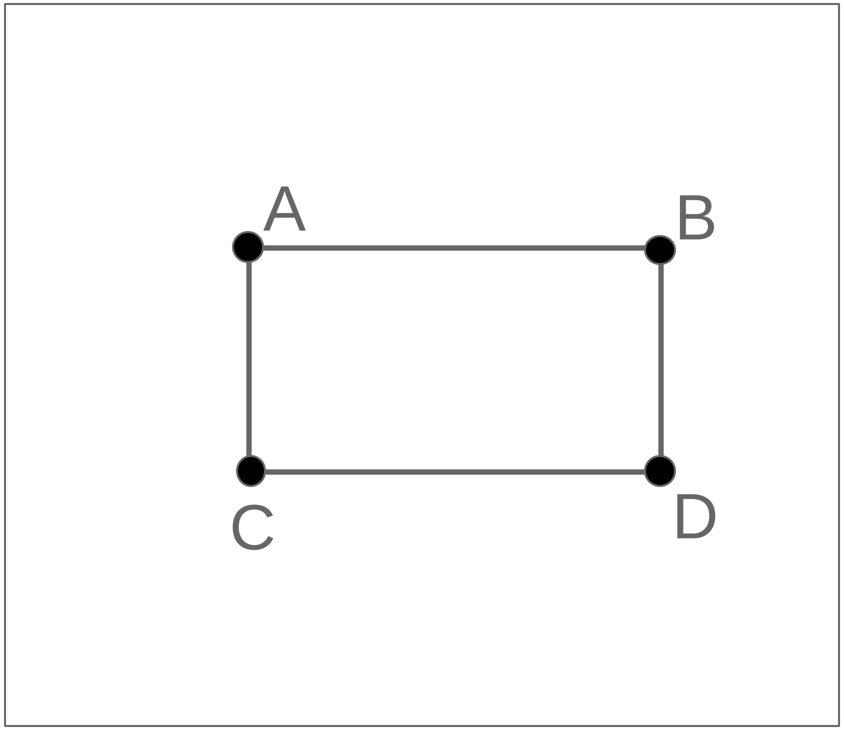
Integral images



input image   integral image

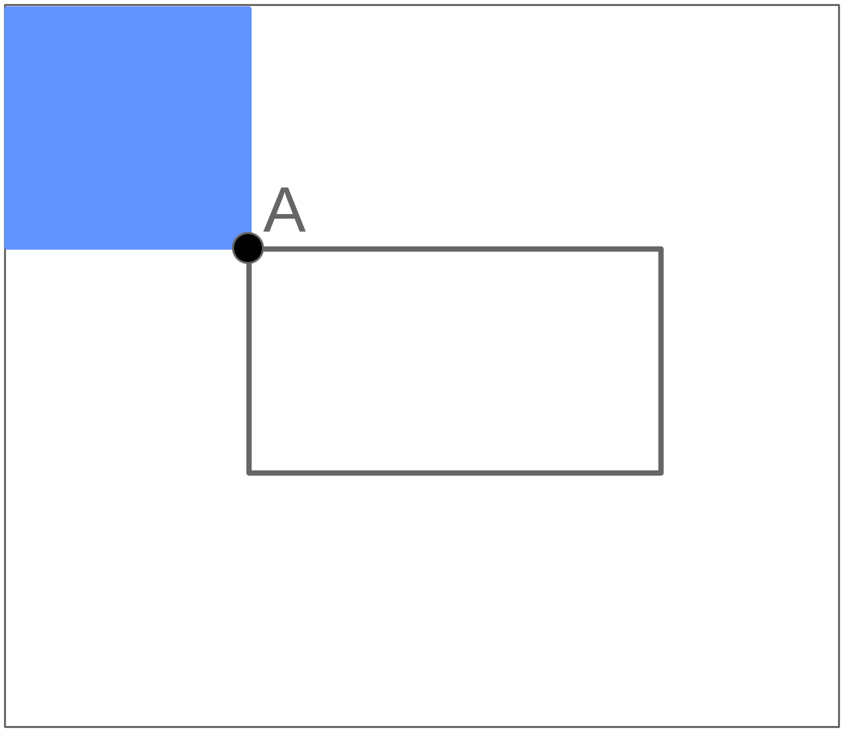
- What's the sum of pixels in the blue rectangle?

Integral images



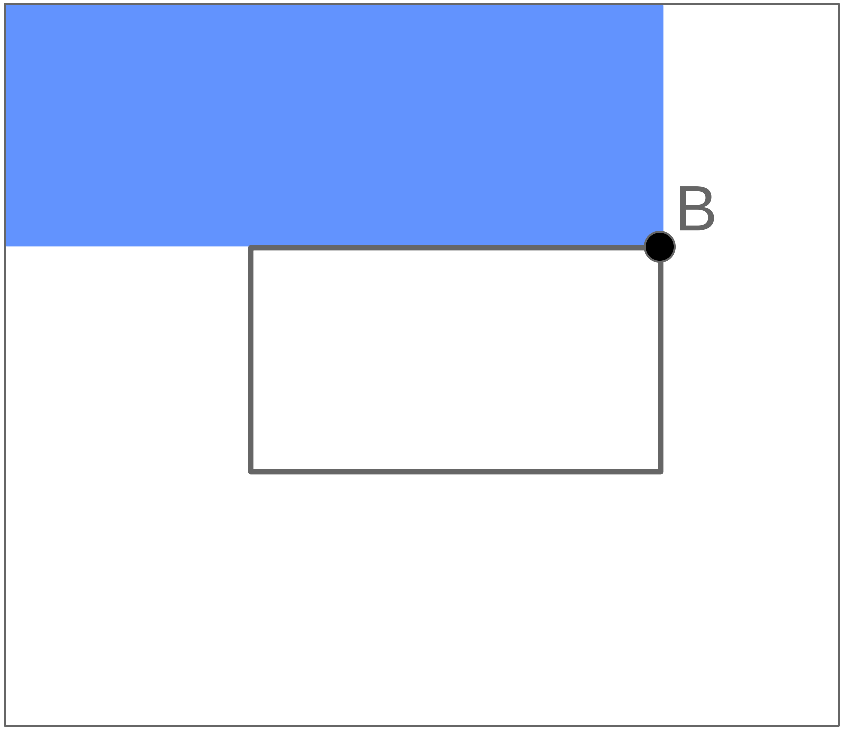
integral image

Integral images



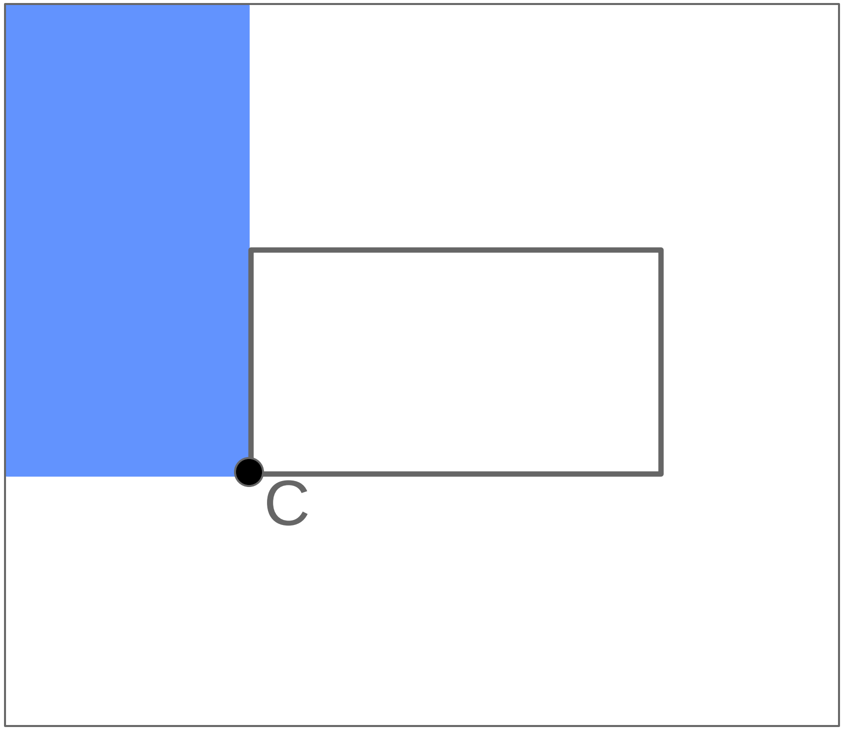
integral image

Integral images



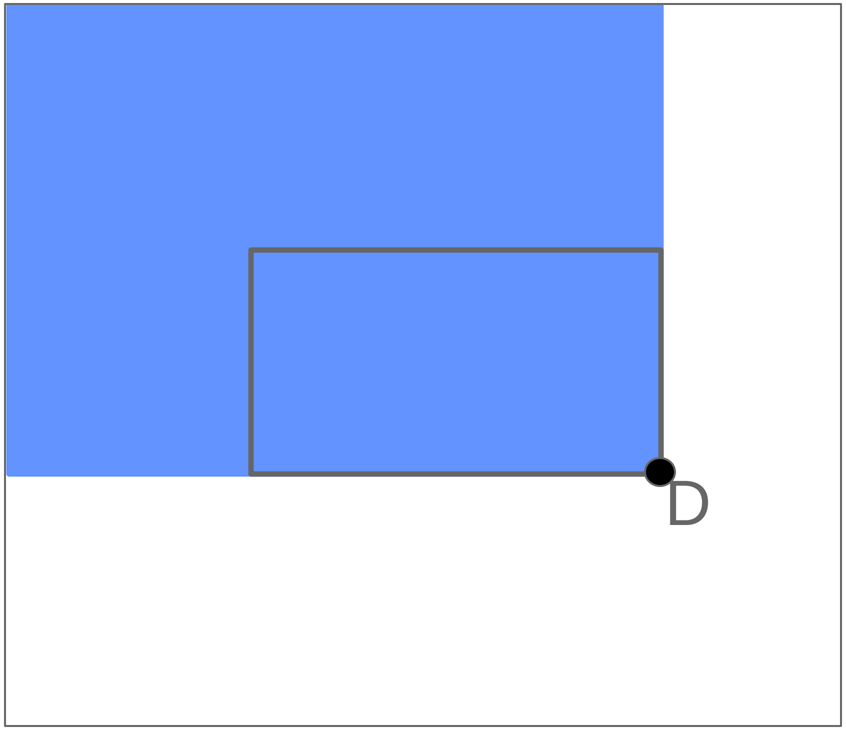
integral image

Integral images



integral image

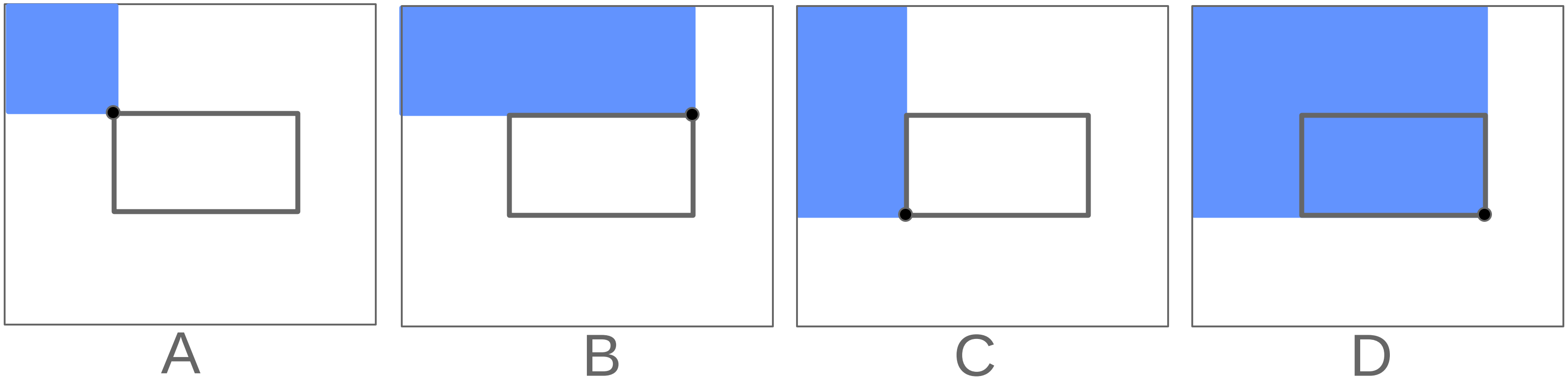
Integral images



integral image

Integral images

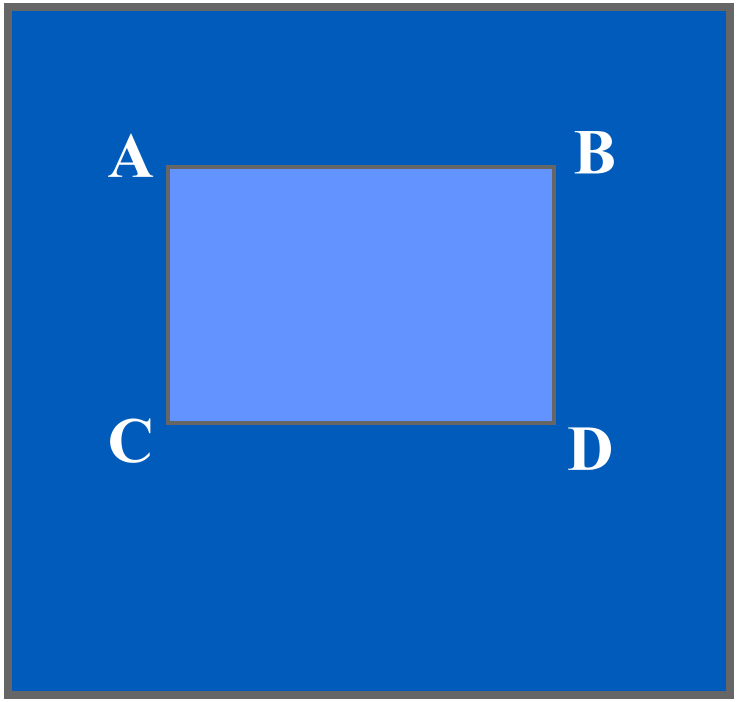
- What’s the sum of pixels in the rectangle?



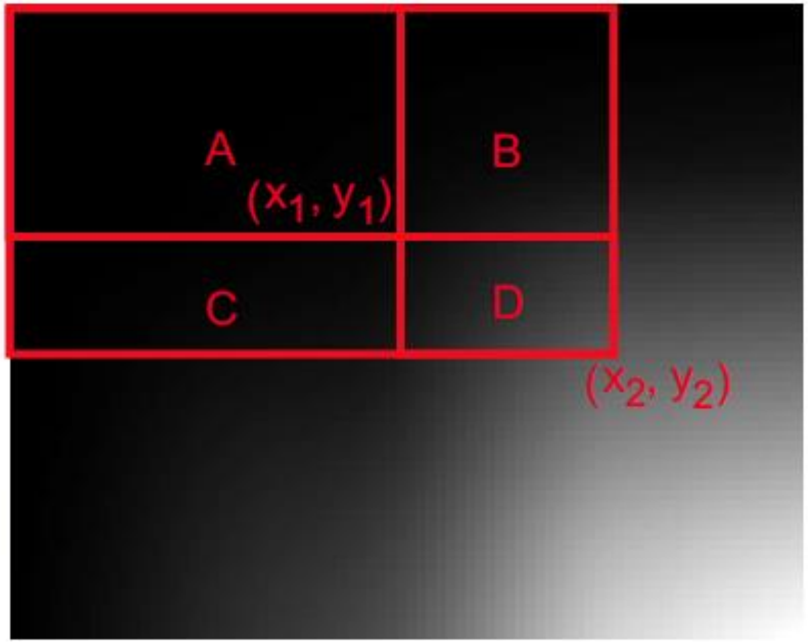
Computing sum within a rectangle

- Let A,B,C,D be the values of the integral image at the corners of a rectangle
- Then the sum of original image values within the rectangle can be computed as:

  sum = D – B – C + A
- **Only 3 additions are required for any size of rectangle!**

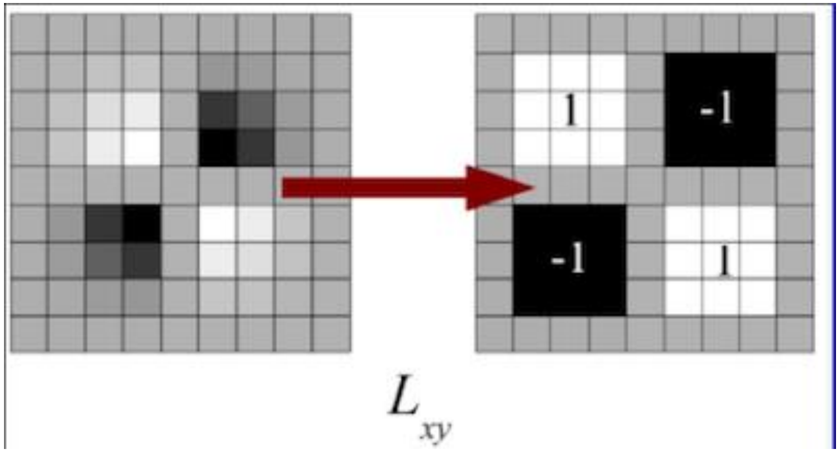


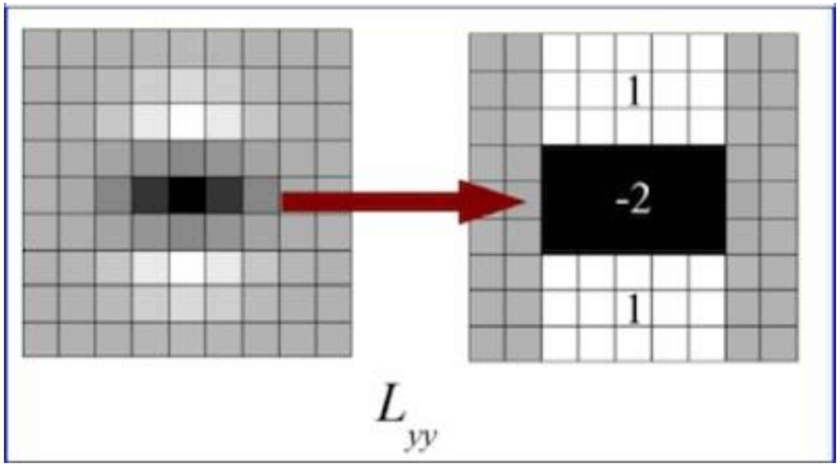
Integral Image Example



Why SURF is Fast?

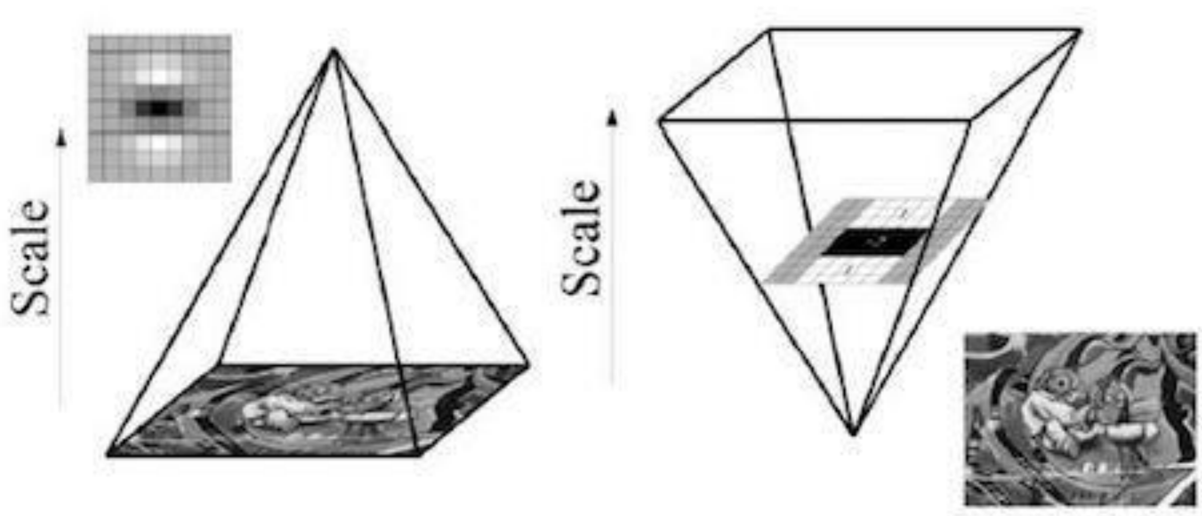
- Second-order Gaussian derivatives and can be approximated at a very low computational cost using integral images



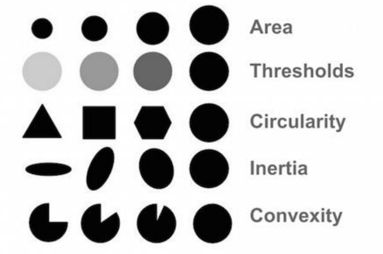


Why SURF is Fast?

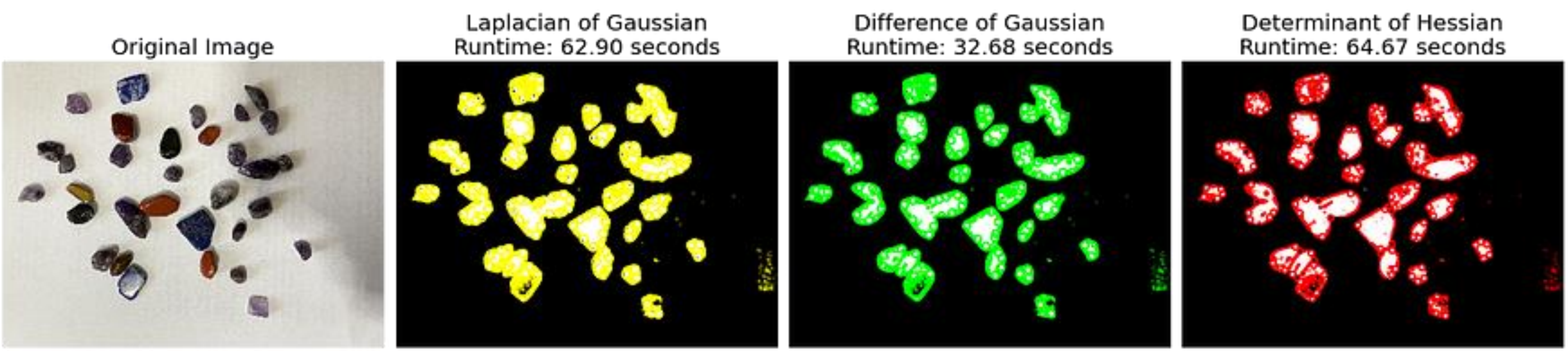
- Instead of iteratively reducing the image size (left), the use of integral images allows the up-scaling of the filter at constant cost (right).
- The scale space is analyzed by up-scaling the filter size,   rather than iteratively reducing the image size.
- So for each new octave, the filter size increase is doubled simultaneously the sampling intervals for the extraction of the interest points(σ) can be doubled, as well which allow the up-scaling of the filter at constant cost.



Other Types of Detector: Blob



- Blob detection
  - A Blob is a group of connected pixels sharing some common property (E.g, grayscale value).
  - Blob detection aims to identify and mark these regions.



Important Concepts

- Key-point detection
  - Corners
  - Repeatable and Distinctive
  - Harris
- Descriptors
  - Robust and selective
  - Histograms of gradient orientation
  - SIFT, SURF
- Evaluation
  - Precision, Recall, F1, ROC