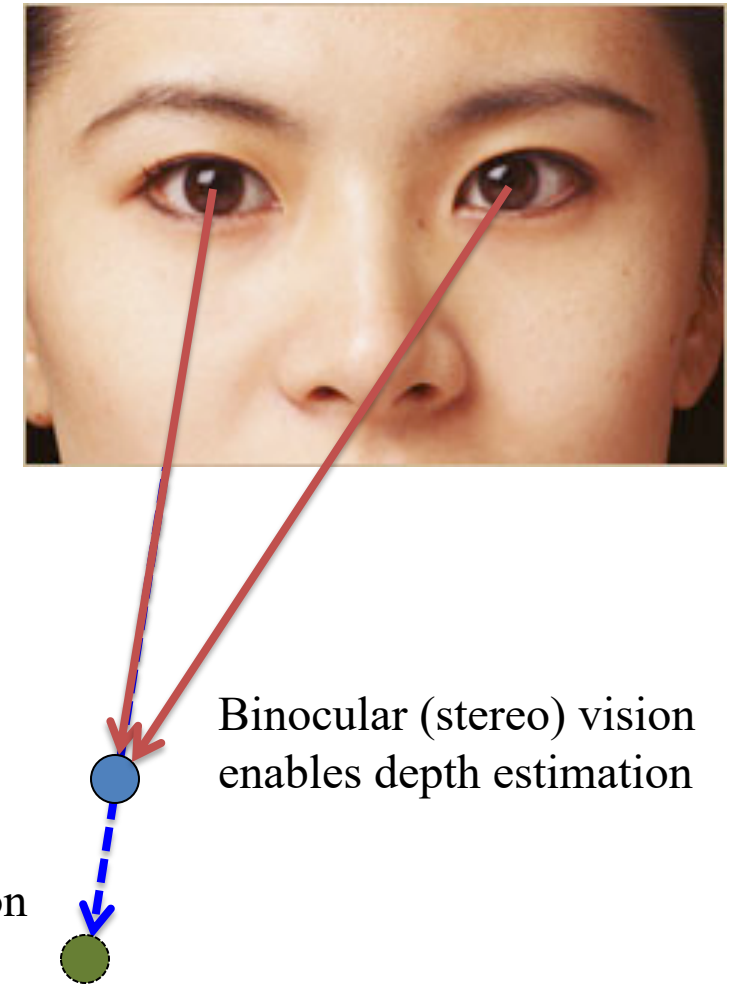
CSE 152: Computer Vision

Hao Su

Lecture 1: Correspondence and Features

Credit: Manmohan Chandraker

Why Do We Have Two Eyes?



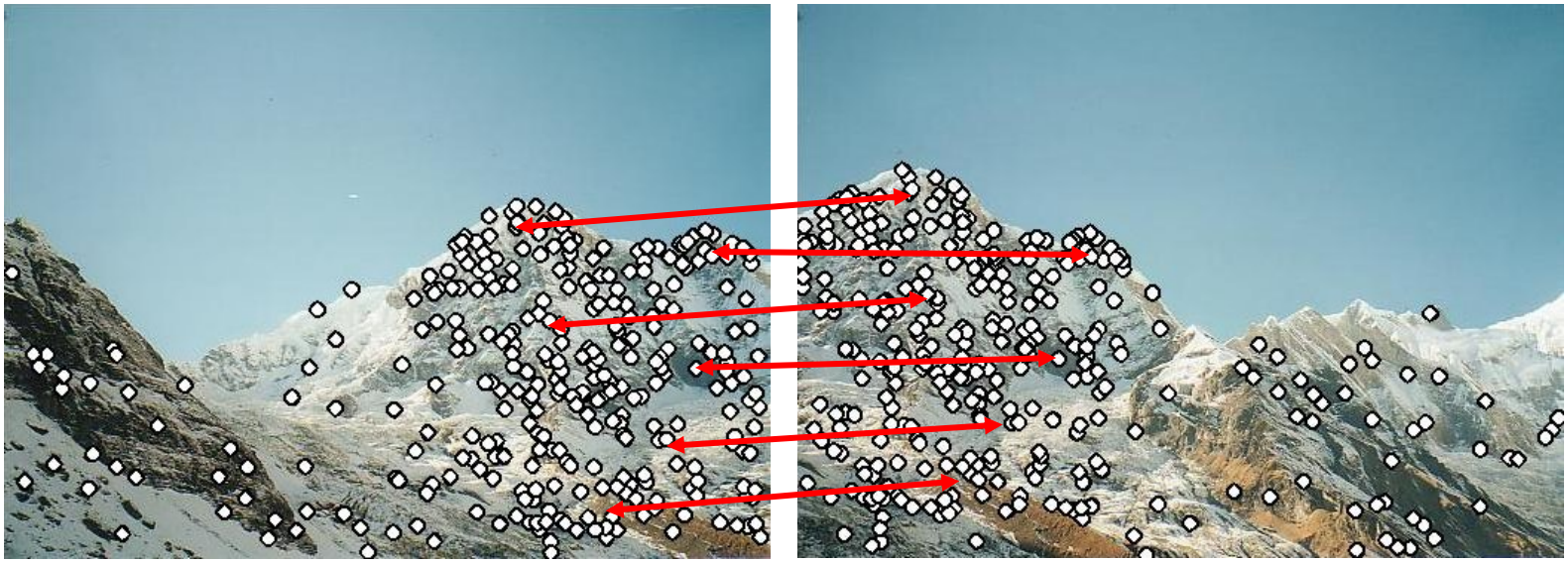
Correspondence estimation

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Slide courtesy: Rick Szeliski

Correspondence estimation

- Motivation: panorama stitching
  - We have two images – how do we combine them?



Step 1: extract features
Step 2: match features

Slide courtesy: Rick Szeliski

Correspondence estimation

- Motivation: panorama stitching
  - We have two images – how do we combine them?

Step 1: extract features
Step 2: match features
Step 3: align images

Slide courtesy: Rick Szeliski

Image matching

by Diva Sian

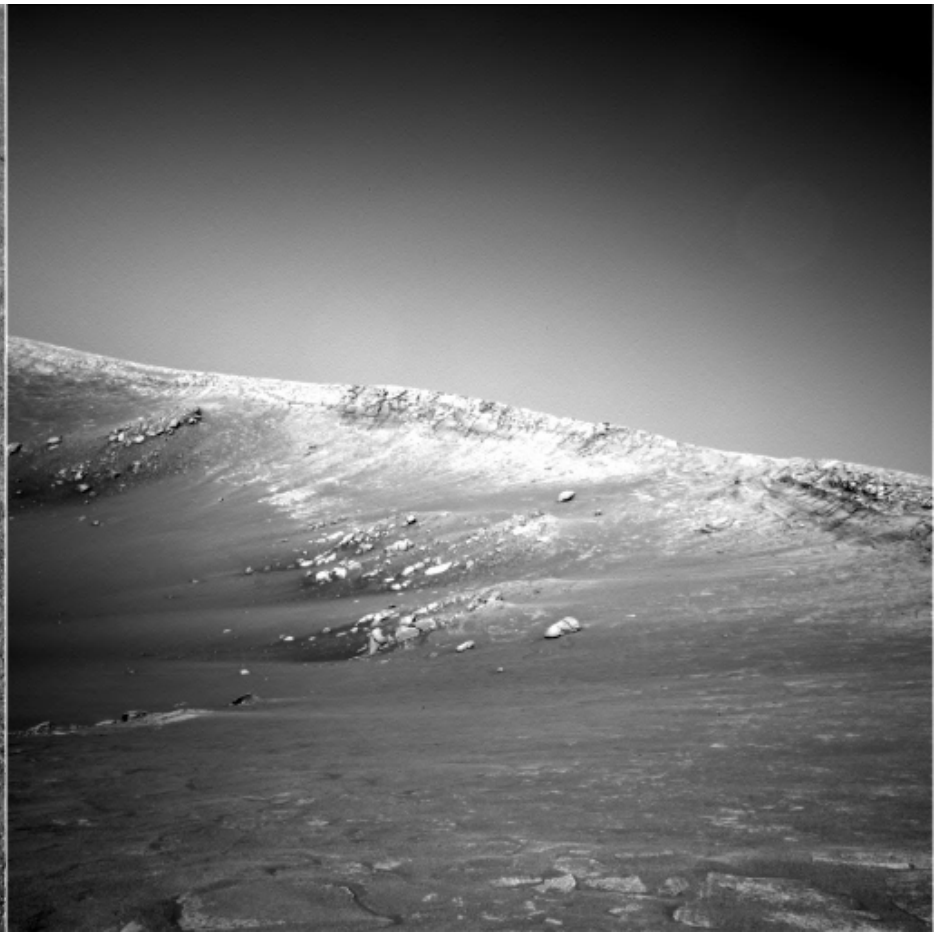
by swashford

Harder case

by Diva Sian

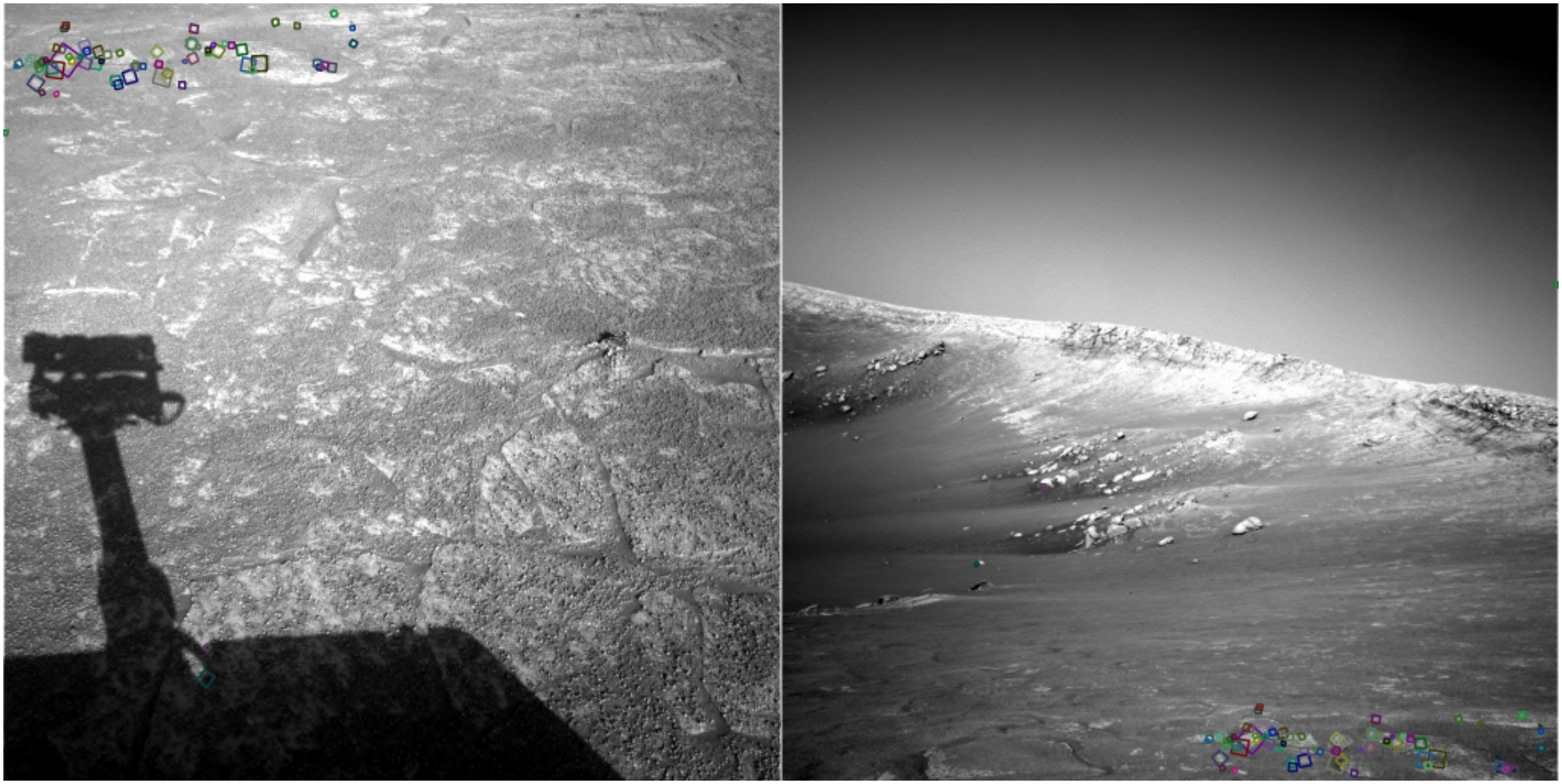
by scgbt

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

Feature detection

Look for image regions that are unusual

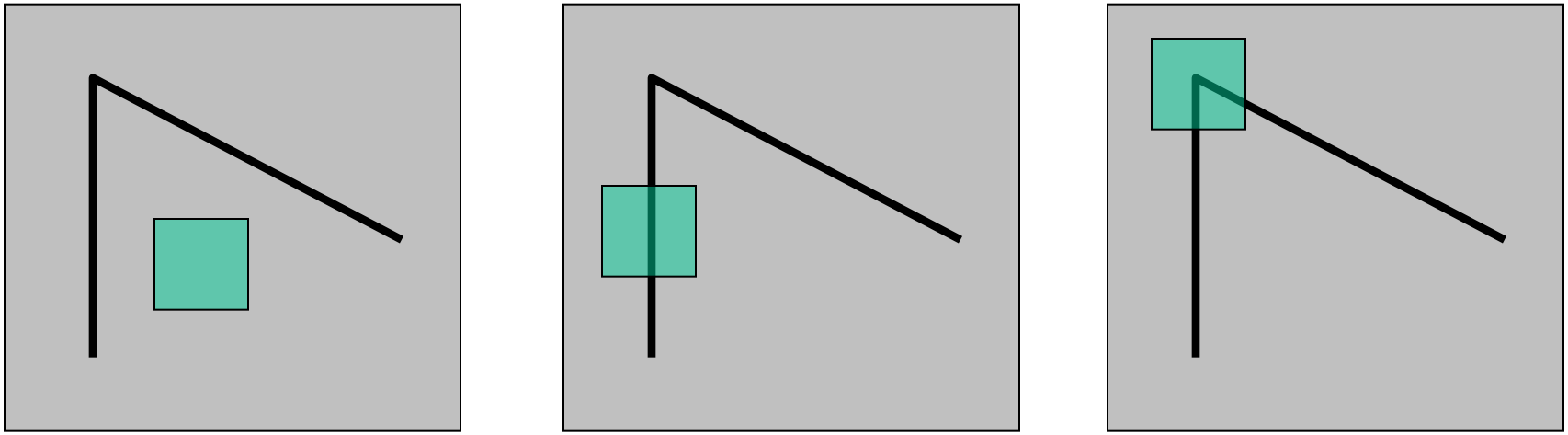
- Lead to unambiguous matches in other images

How to define “unusual”?

Local measures of uniqueness

Suppose we consider a small window of pixels

- What defines whether a feature is a good or bad candidate?

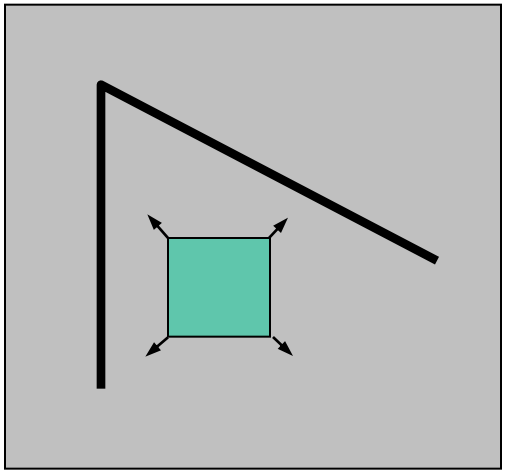


Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

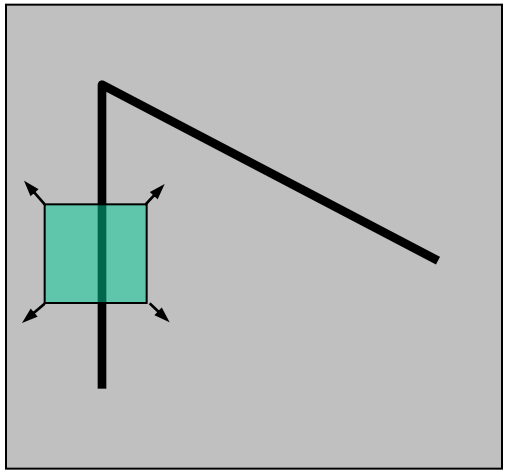
Feature detection

Local measure of feature uniqueness

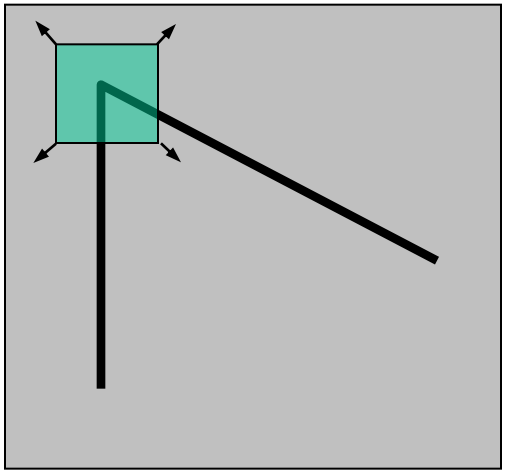
- How does the window change when you shift it?
- Shifting the window in *any direction* causes a *big change*



"flat" region:
no change in all directions



"edge":
no change along the edge direction



"corner":
significant change in all directions

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

The Harris corner detector

Compute second-moment matrix:

Sum over a small region around the hypothetical corner

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

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Matrix is symmetric

Slide credit: David Jacobs

Simple Case

First, consider case where:

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

This means 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.

Slide credit: David Jacobs

General Case

It can be shown that since C is symmetric:

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

So every case is like a rotated version of the one on previous slide.

Slide credit: David Jacobs

Harris Corner Detector

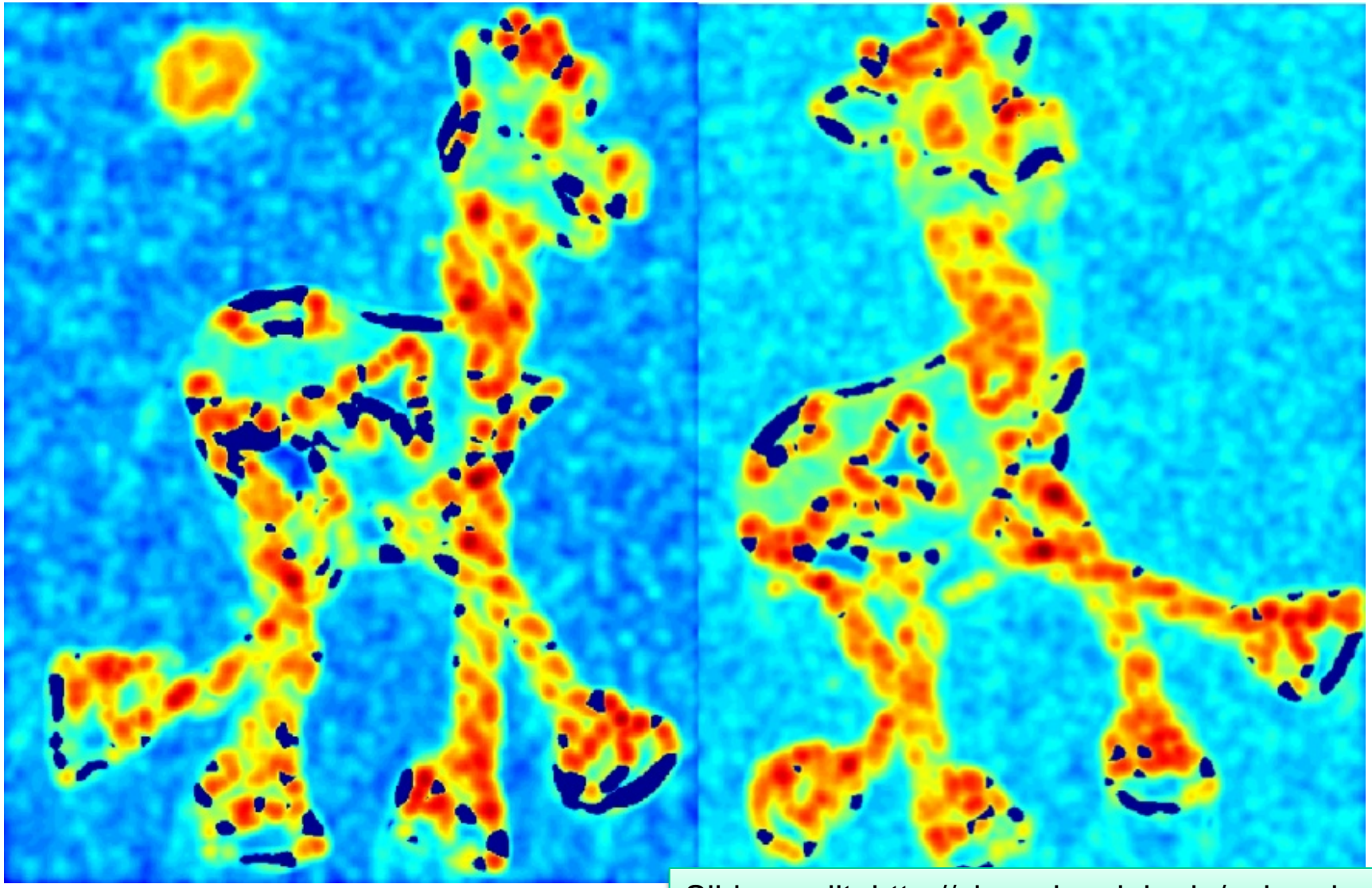
- Filter image with Gaussian to reduce noise
- Compute magnitude of the x and y gradients at each pixel
- Construct C in a window around each pixel
- If λs are both big, we have a corner.

Harris Detector: Workflow

Slide credit: http://vims.cis.udel.edu/~chandra/

Harris Detector: Workflow

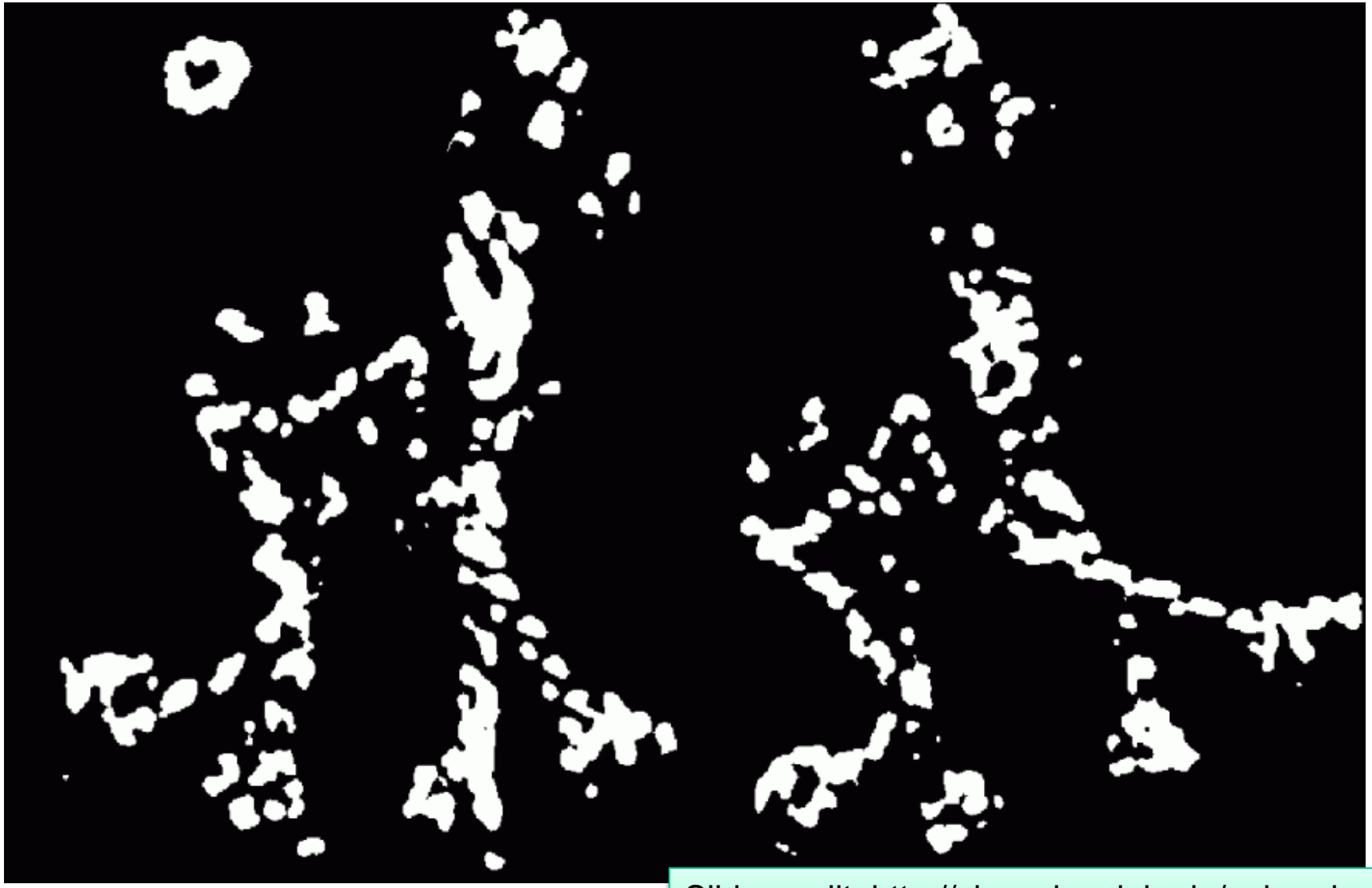
Compute corner response R



Slide credit: http://vims.cis.udel.edu/~chandra/

Harris Detector: Workflow

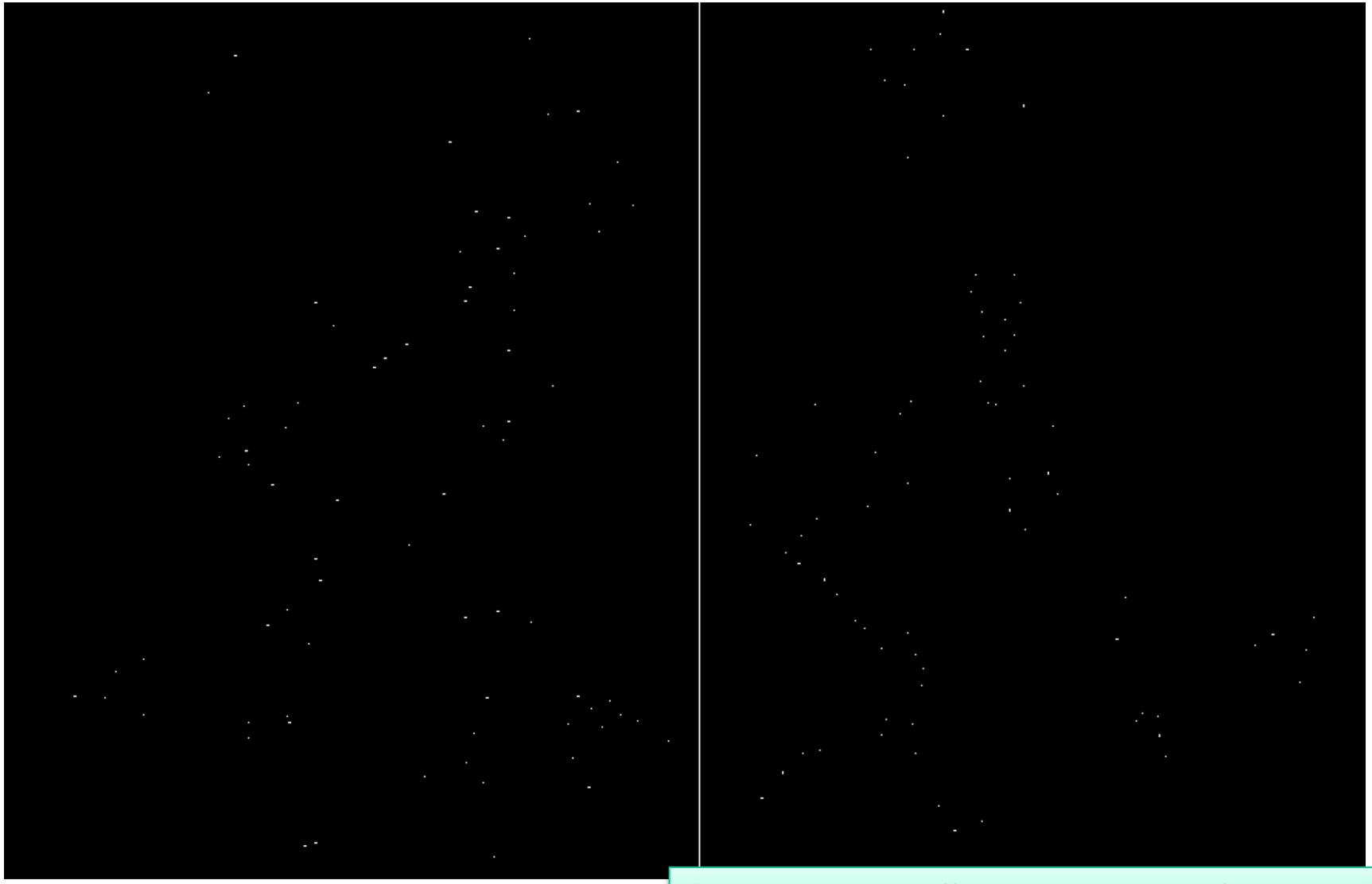
Find points with large corner response: R > threshold



Slide credit: http://vims.cis.udel.edu/~chandra/

Harris Detector: Workflow

Take only the points of local maxima of R



Slide credit: http://vims.cis.udel.edu/~chandra/

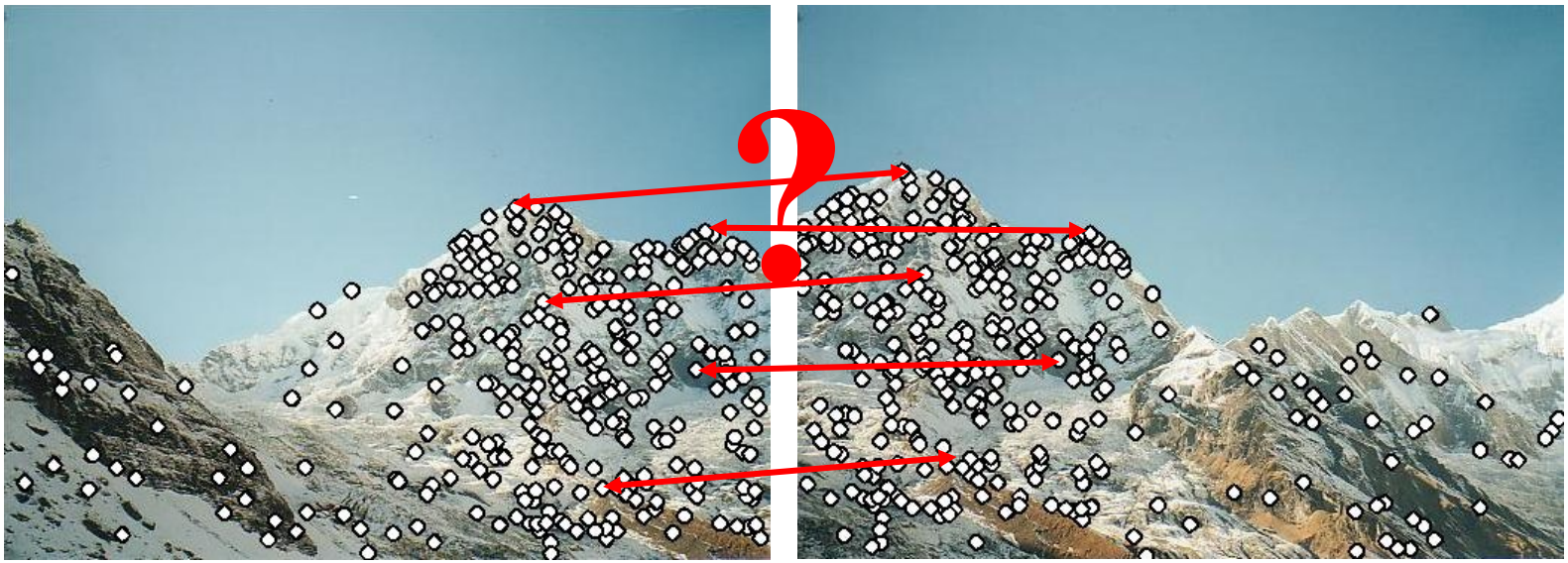
Harris Detector: Workflow

Slide credit: http://vims.cis.udel.edu/~chandra/

Feature descriptors

We know how to detect good points

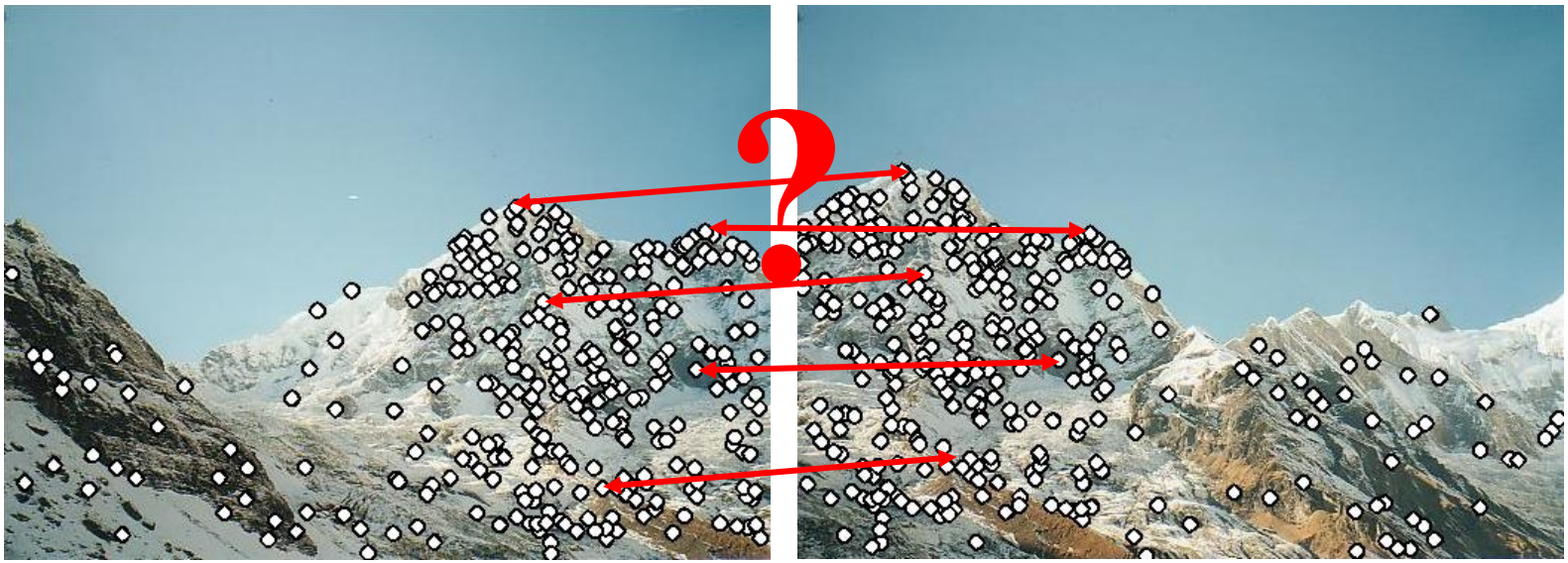
Next question: **How to match them?**



Slide courtesy: Rick Szeliski

Feature descriptors

We know how to detect good points
Next question: **How to match them?**

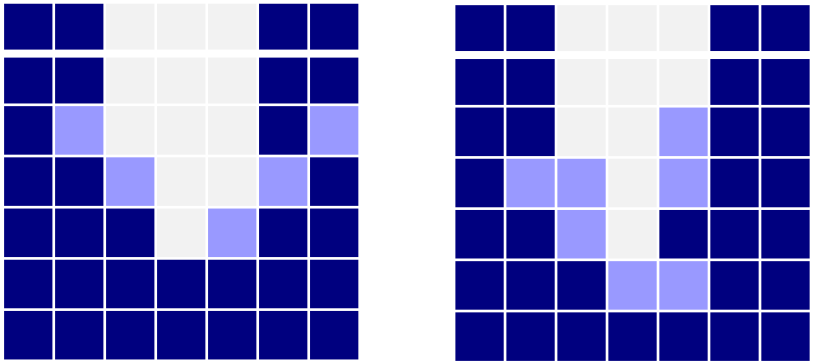


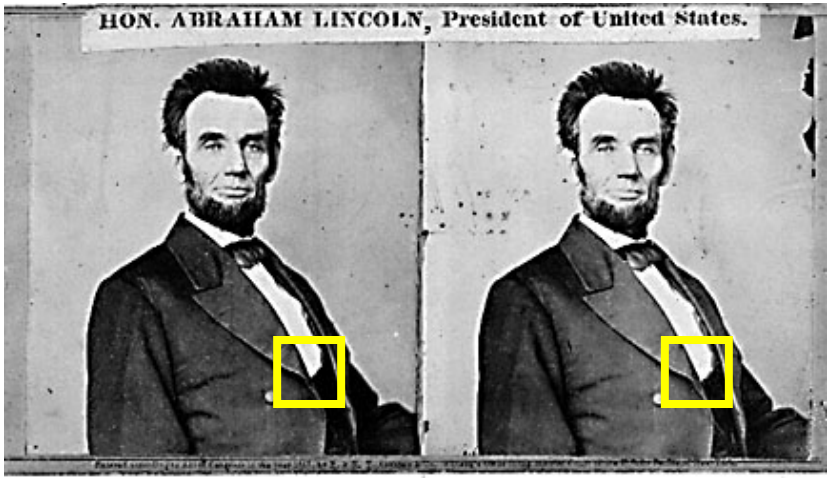
Lots of possibilities (this is a popular research area)

- Simple option: match square windows around the point
- State of the art approach: SIFT
  - David Lowe, UBC http://www.cs.ubc.ca/~lowe/keypoints/

Slide courtesy: Rick Szeliski

Simple matching methods





- SSD (Sum of Squared Differences)

$$\sum_{x,y} |W_1(x,y) - W_2(x,y)|^2$$

- NCC (Normalized Cross Correlation)

$$\frac{\sum_{x,y}(W_1(x,y) - \overline{W_1})(W_2(x,y) - \overline{W_2})}{\sigma_{W_1}\sigma_{W_2}}$$

where $\overline{W_i} = \frac{1}{n}\sum_{x,y} W_i$, $\sigma_{W_i} = \sqrt{\frac{1}{n}\sum_{x,y}(W_i - \overline{W_i})^2}$

- What advantages might NCC have over SSD?

Invariance

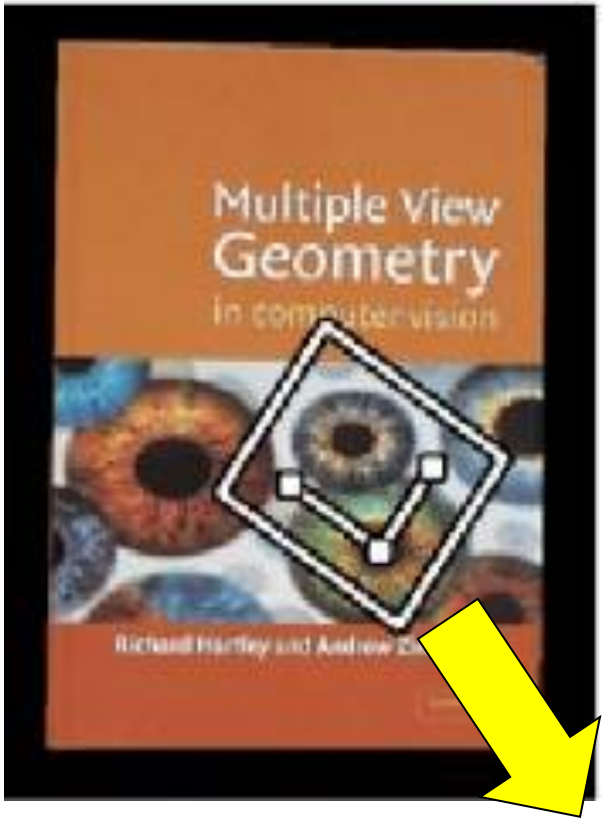
Suppose we are comparing two images I_1 and I_2

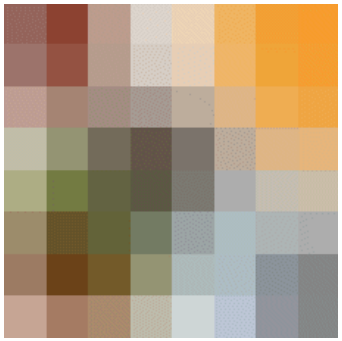
- I_2 may be a transformed version of I_1
- What kinds of transformations are we likely to encounter in practice?

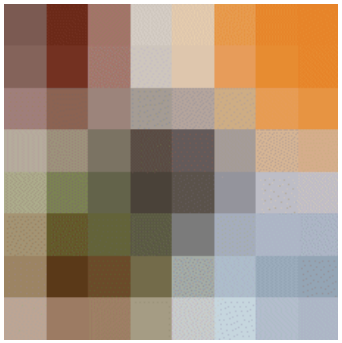
We'd like to find the same features regardless of the transformation

- This is called transformational ***invariance***
- Most feature methods are designed to be invariant to
  - Translation, 2D rotation, scale
- They can usually also handle
  - Limited 3D rotations (SIFT works up to about 30 degrees)
  - Limited illumination or contrast changes

Feature Matching



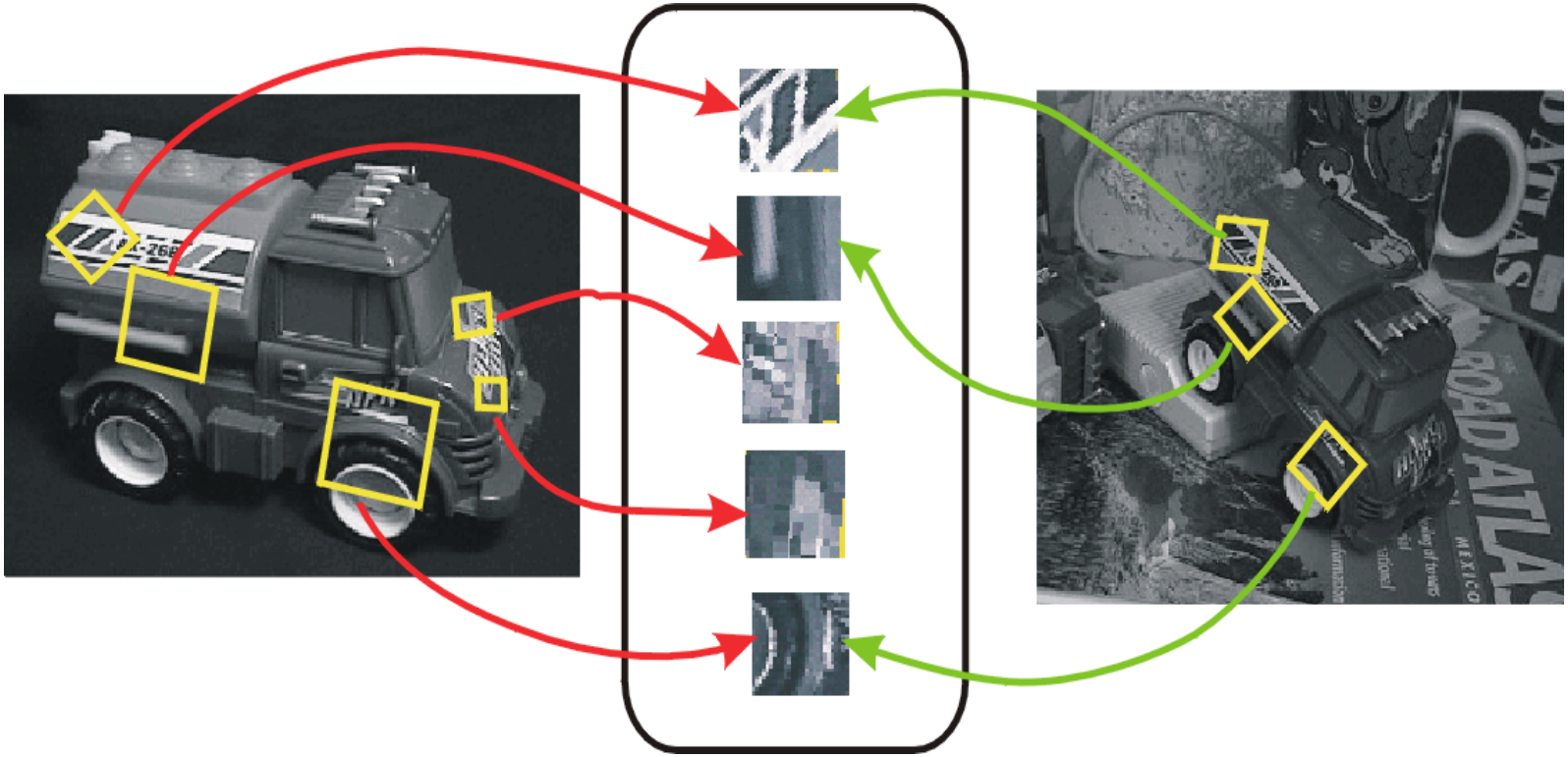




Desirable property: invariance

Find features that are invariant to transformations

- geometric invariance: translation, rotation, scale
- photometric invariance: brightness, exposure, …



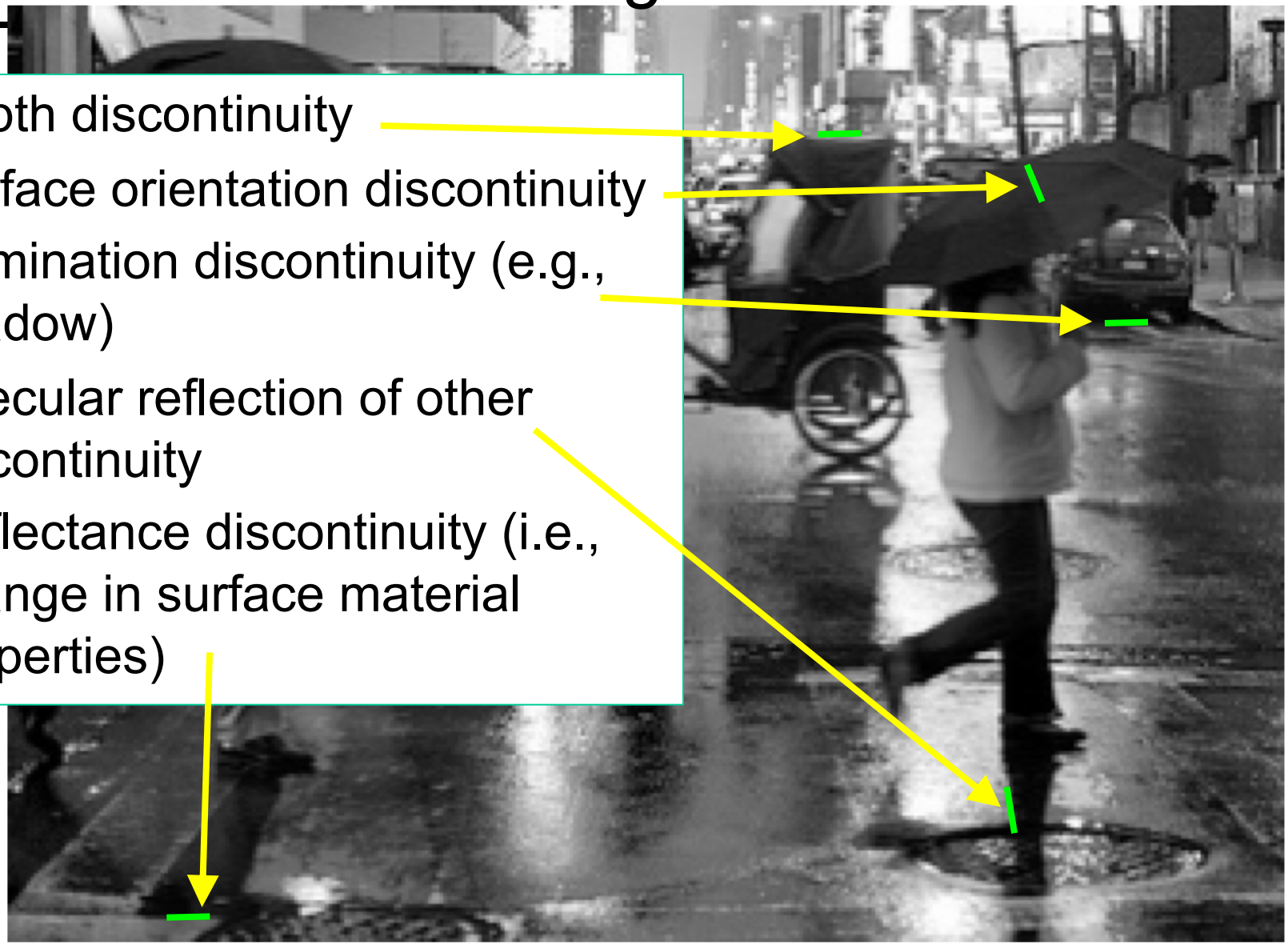
Feature Descriptors

Edges in Natural Images

Source: Photografr.com

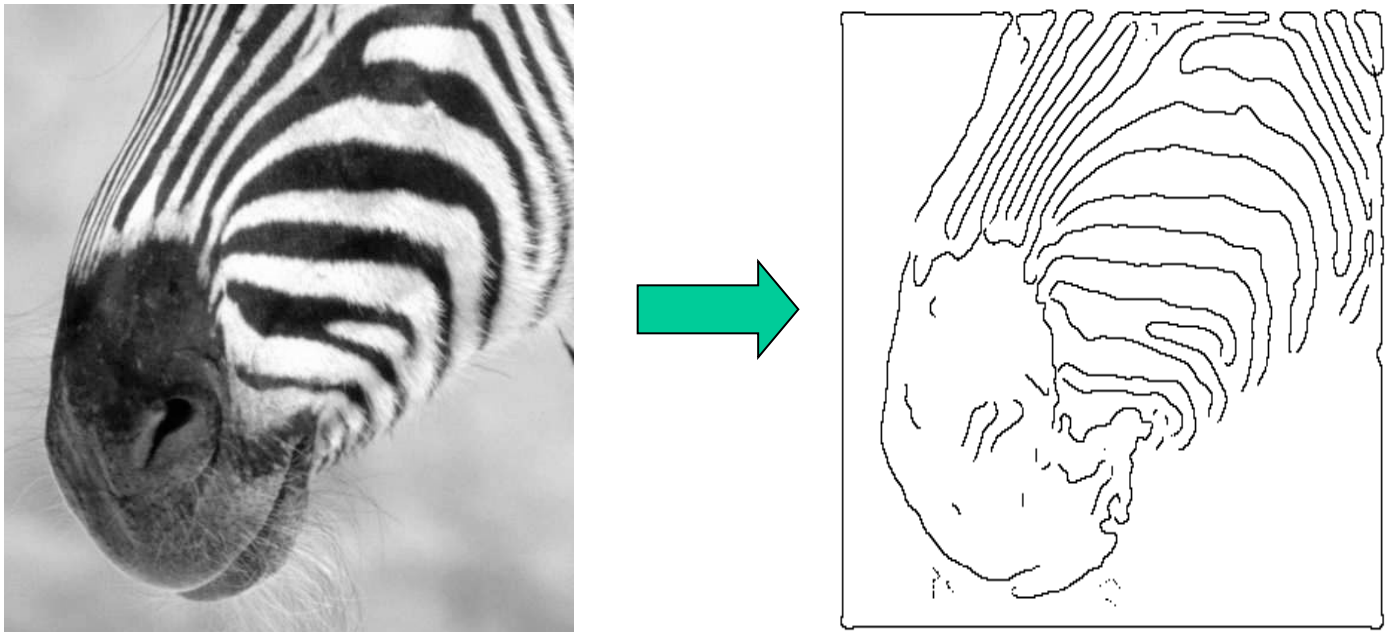
What Causes an Edge?

- Depth discontinuity
- Surface orientation discontinuity
- Illumination discontinuity (e.g., shadow)
- Specular reflection of other discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)



Source: Photografr.com

How Can We Find Edges?

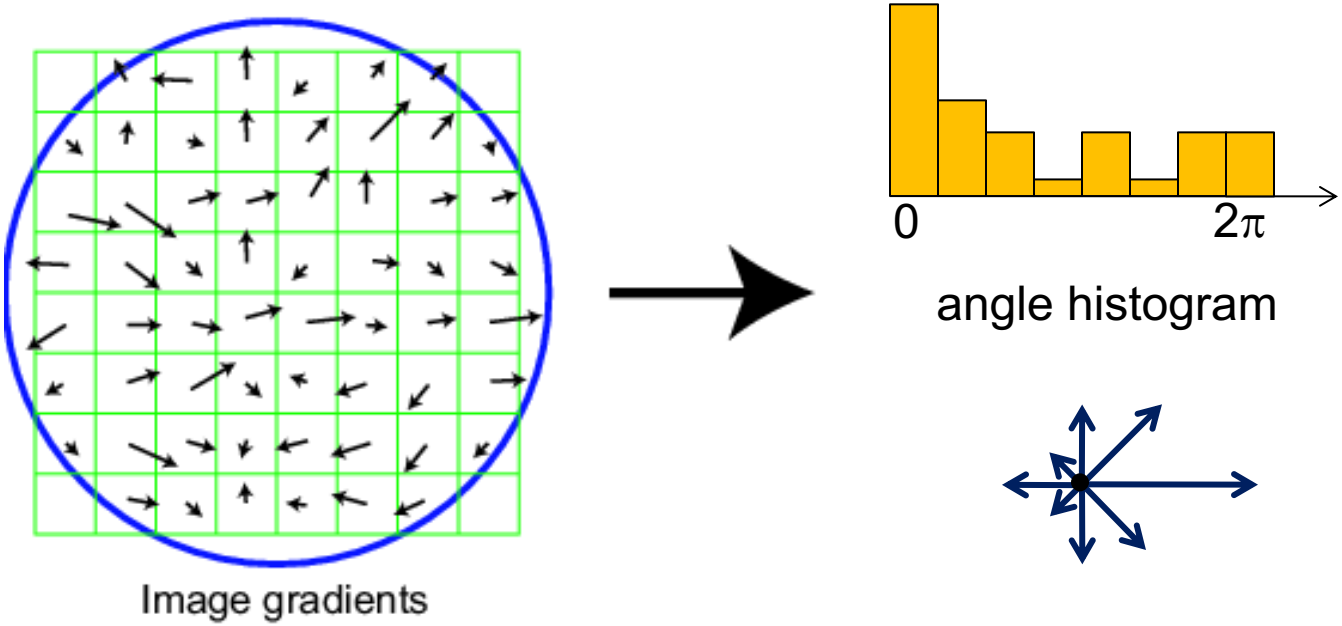


Find regions where magnitude of gradient is large.

Scale Invariant Feature Transform

Basic idea:

- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations



Adapted from slide by David Lowe

Properties of SIFT

Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 30 degrees out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)
- Fast and efficient—can run in real time
- Lots of code available
  - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT

Feature matching

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

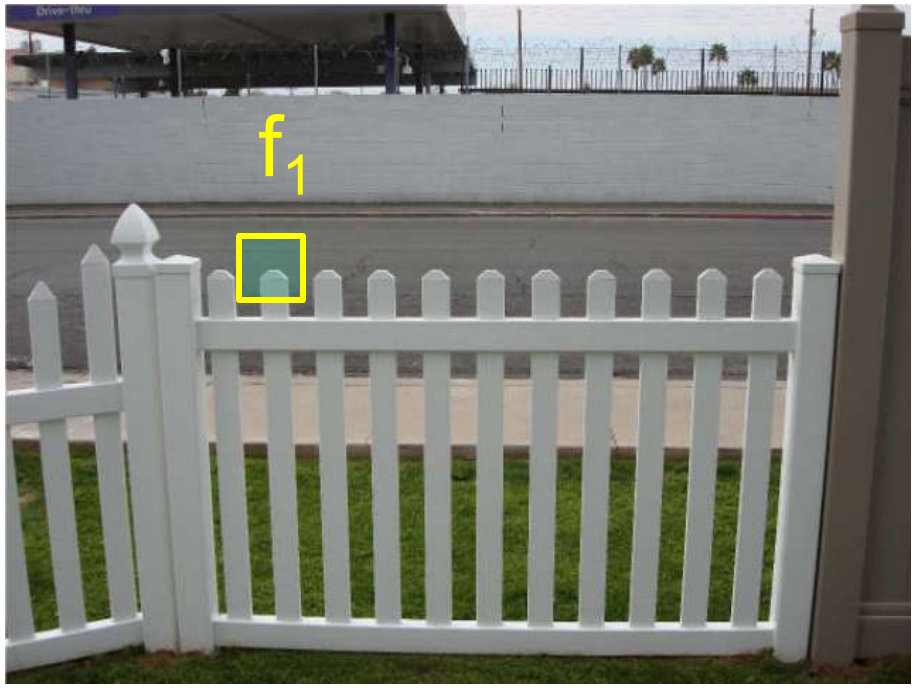
1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with min distance

Slides courtesy: Rick Szeliski

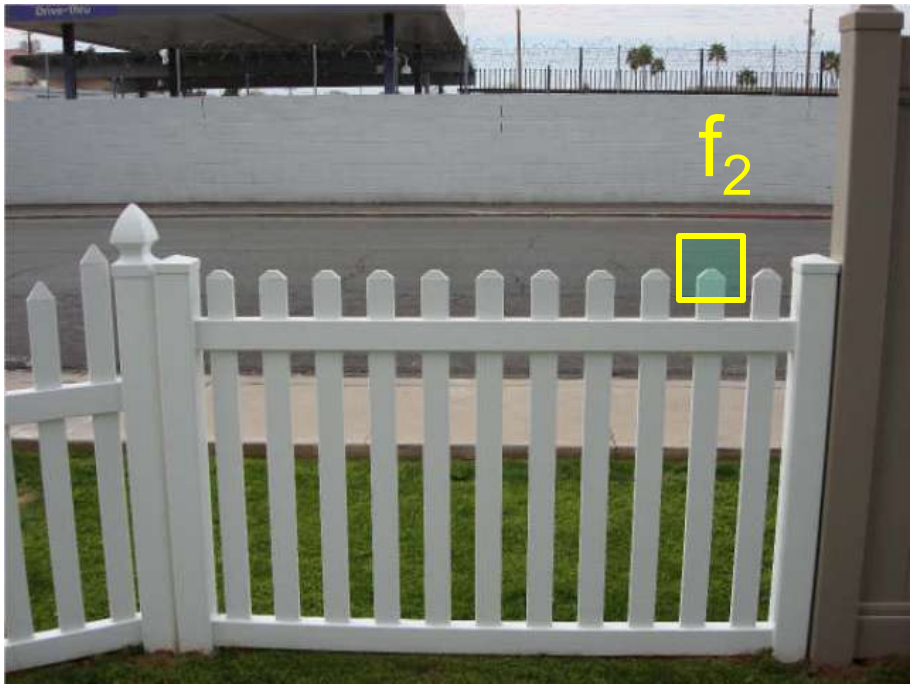
Feature distance

How to define the difference between two features f_1, f_2?

- Simple approach is SSD(f_1, f_2)
  - sum of square differences between entries of the two descriptors
  - can give good scores to very ambiguous (bad) matches



I_1

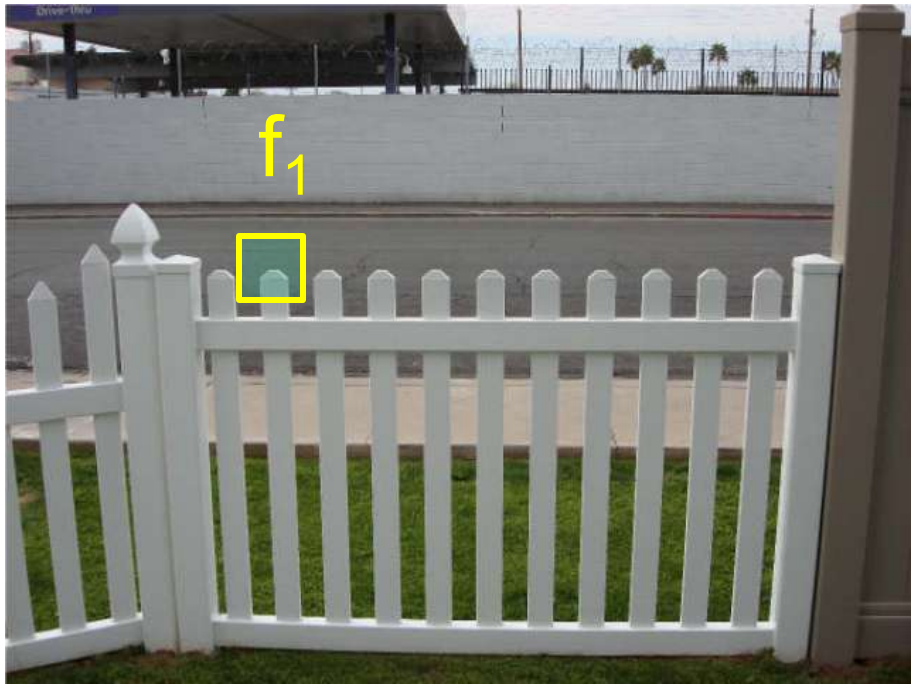


I_2

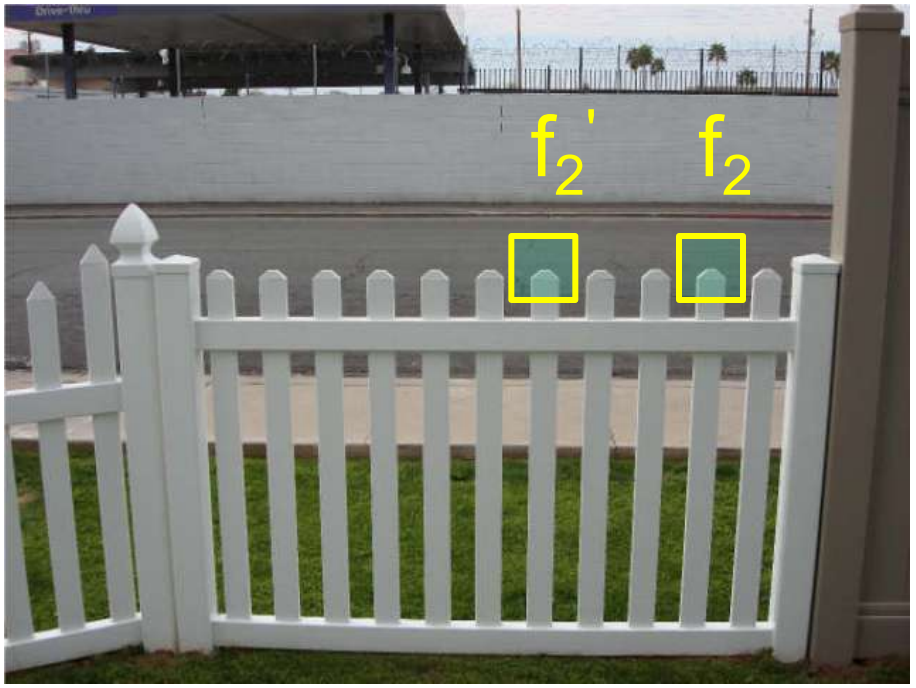
Feature distance

How to define the 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
  - f_2' is 2nd best SSD match to f_1 in I_2
  - gives small values for ambiguous matches



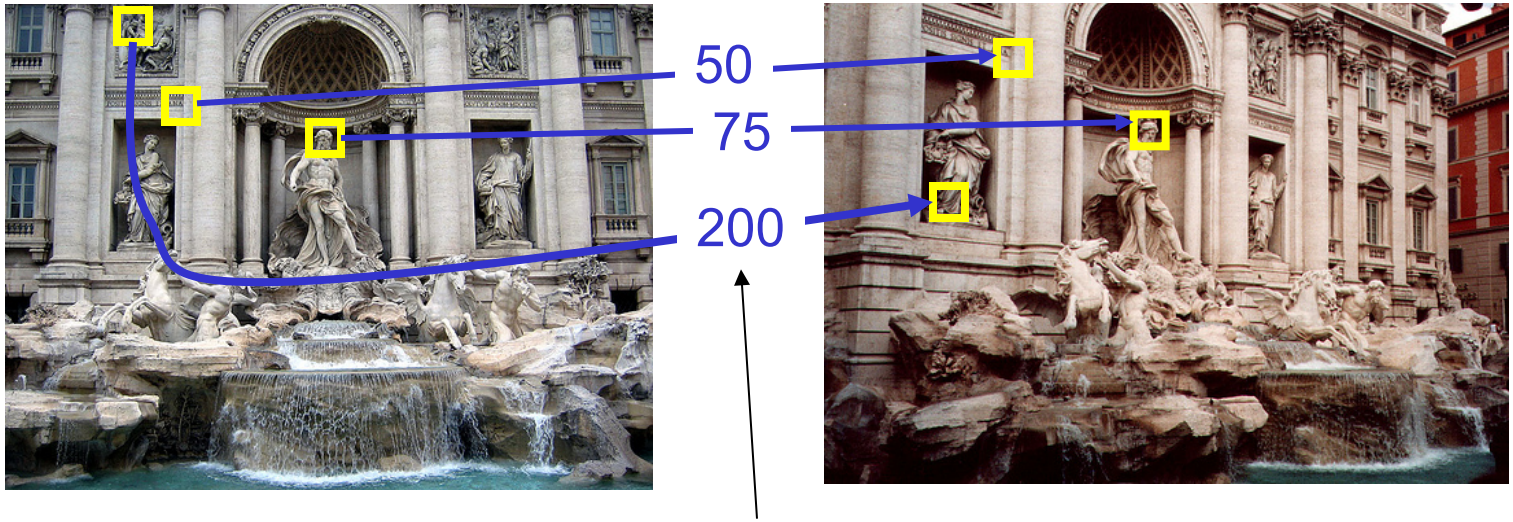
I_1



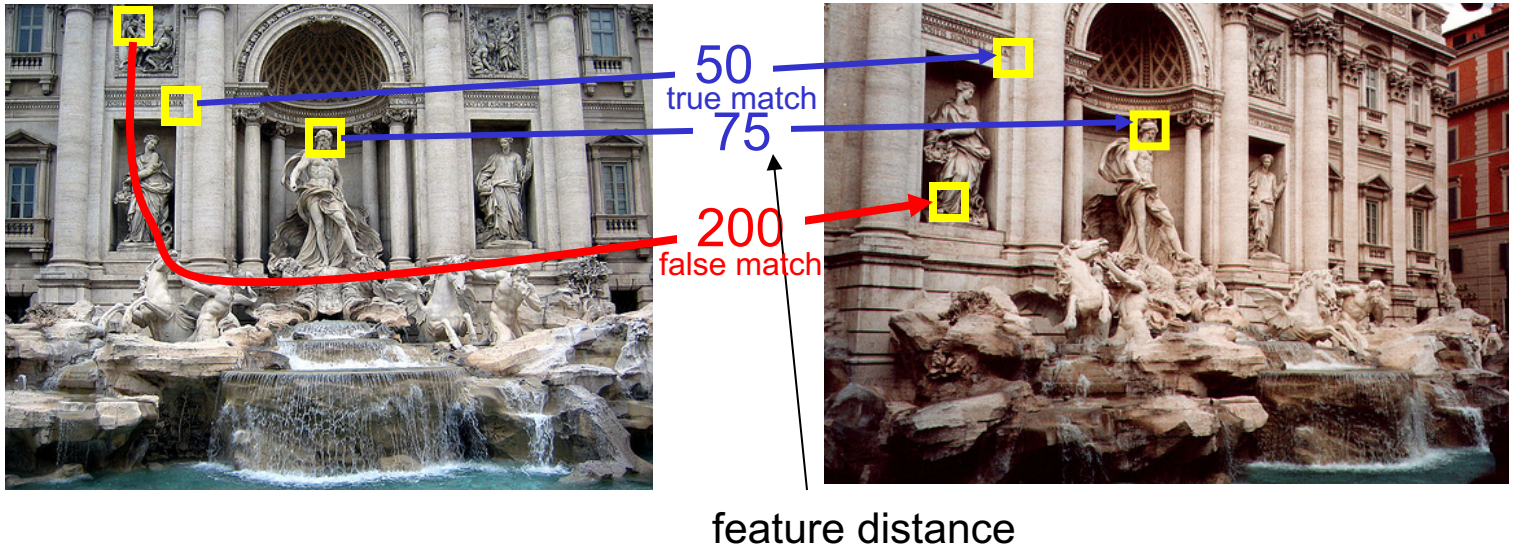
I_2

Evaluating the results

How can we measure the performance of a feature matcher?



True or false positives



The distance threshold affects performance

- True positives = number of detected matches that are correct
  - Suppose we want to maximize these—how to choose threshold?

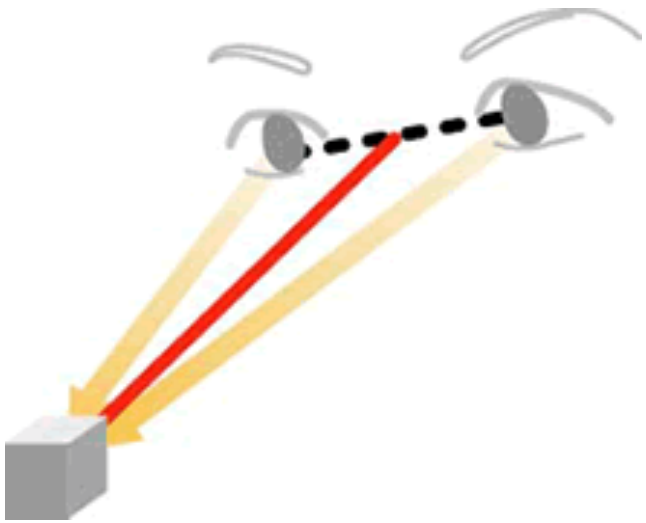
- False positives = number of detected matches that are incorrect
  - Suppose we want to minimize these—how to choose threshold?

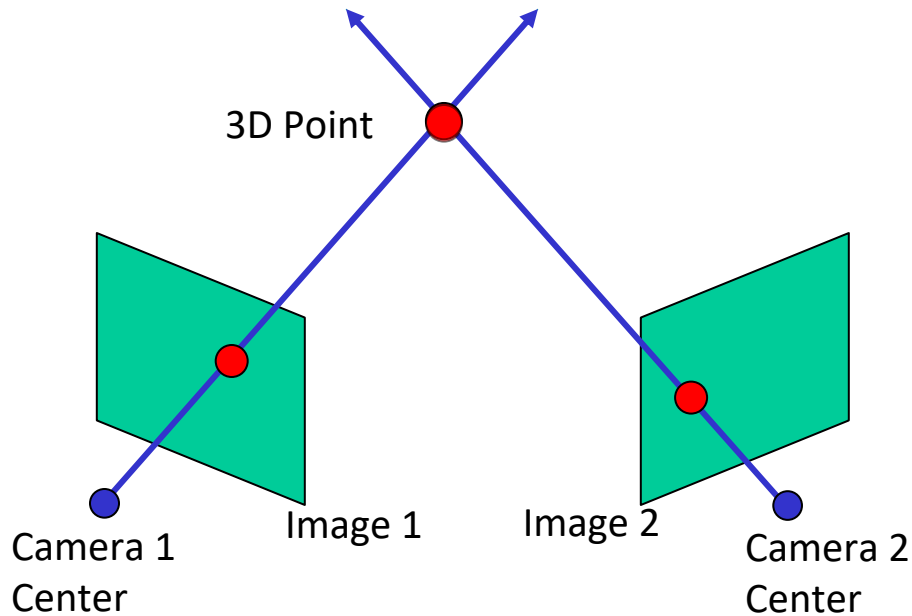
Sony Aibo

SIFT usage:

- Recognize charging station
- Communicate with visual cards
- Teach object recognition

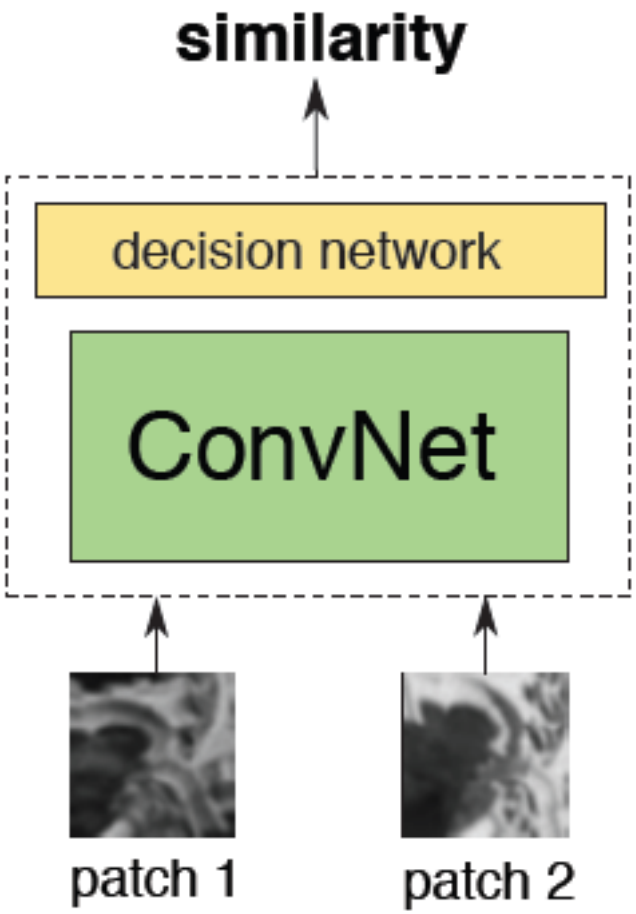
Correspondence is a vital 3D cue





We haven't yet described how such projections are performed

Learning correspondence



Idea:

- Siamese network to decide patch similarity
- Use intermediate activations as features.

Advantages:

- Easy to train

Issues:

- Inefficient: extract features for overlapping regions within patches
- $O(n^2)$ feed-forward passes to compare n patches in each image.

[Zagoruyko and Komodakis, CVPR 2015]