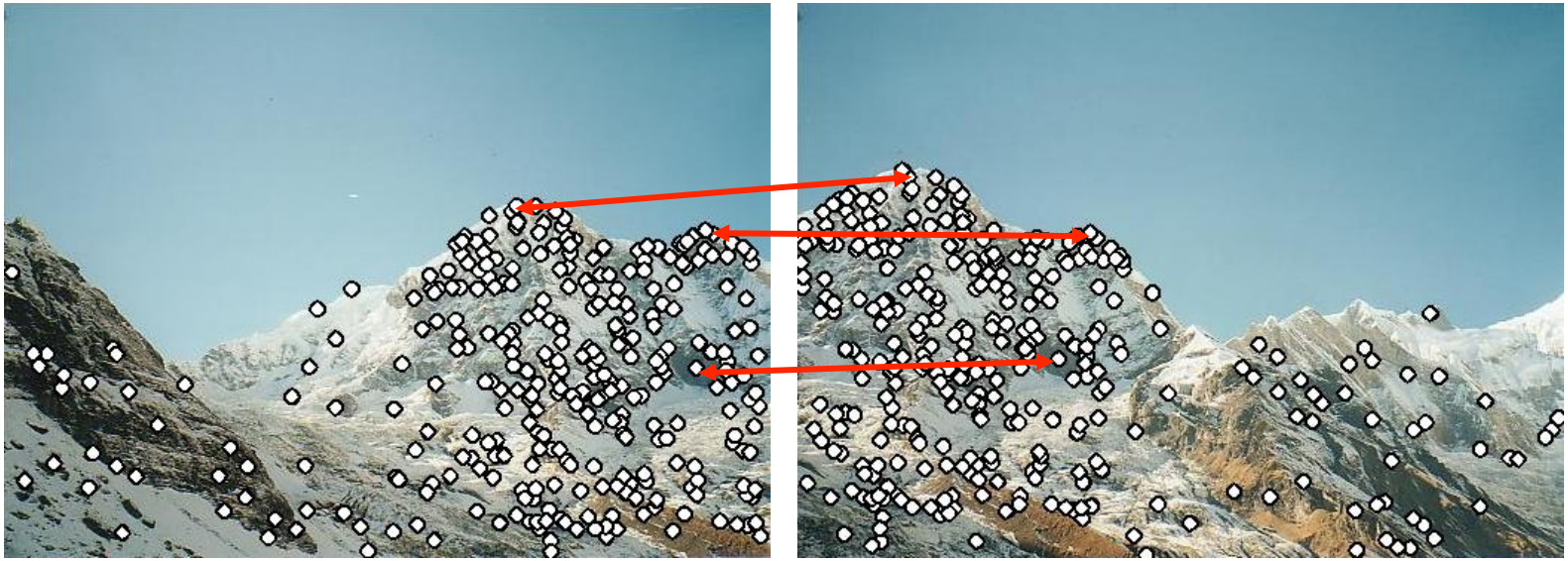
FEATURE EXTRACTION (SIFT)

Class 6: Artificial Vision



Szeliski: Computer Vision: Algorithms and applications, Chapter 4, Feature extraction and matching

How much information do we need in order to recognize objects?

Informative vs. non-informative regions

http://petitinvention.wordpress.com/2008/02/10/future-of-internet-search-mobile-version/

How do we use the informative regions?

Diva Sian

swashford

How do we use the informative regions?

Diva Sian

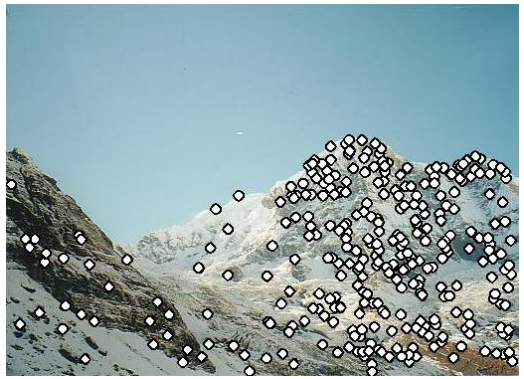
scgbt

Today

- Local invariant features
  - Detection of interest points
    - Harris corner detection
    - Scale invariance
  - Description of local patches
    - SIFT : Histograms of oriented gradients
  - Matching

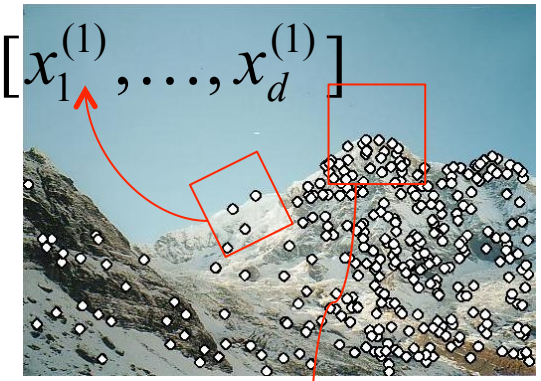
Local features: main components

1) **Detection**: Identify the interest points



2) **Description**: Extract vector feature descriptor surrounding each interest point.

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



3) **Matching**: Determine correspondence between descriptors in two views

$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

Kristen Grauman

Local features: desired properties

- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description

- Compactness and efficiency
  - A few features vs. all image pixels
- Locality
  - A feature occupies a relatively small area of the image, simple but robust and stable.

Goal: interest operator repeatability

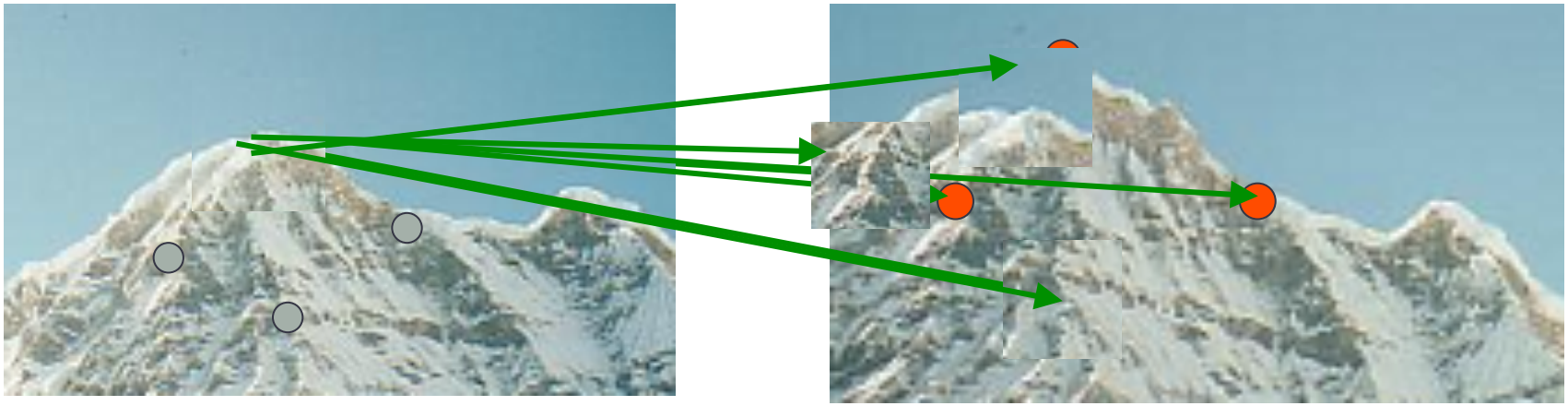
- We want to detect (at least some of) the same points in both images.

No chance to find true matches!

- Yet we have to be able to run the detection procedure *independently* per image.

Goal: descriptor distinctiveness

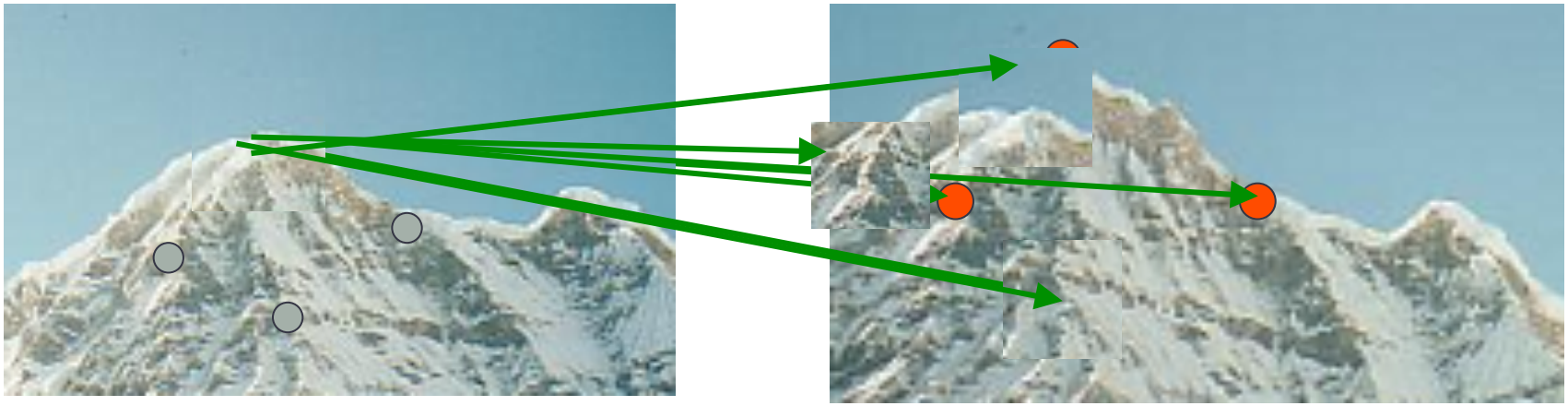
- We want to be able to reliably determine which point goes with which.



- Must provide some invariance to geometric and photometric differences between the two views.

Goal: compactness, efficiency and locality

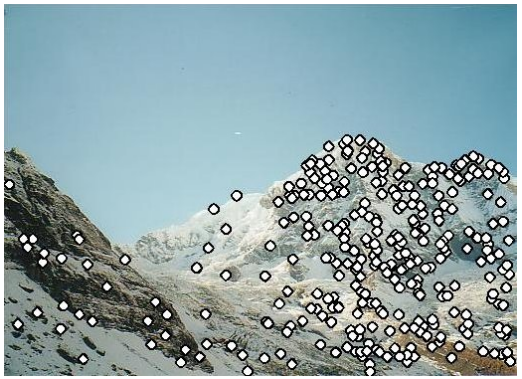
- We want to extract the minimum information that allows us to put into correspondence.



- Which should it be?

Local features: main components

1) **Detection:** Identify the interest points

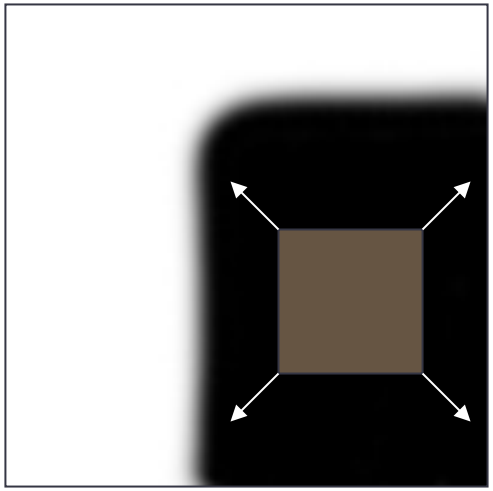


2) **Description:** Extract vector feature descriptor surrounding each interest point.

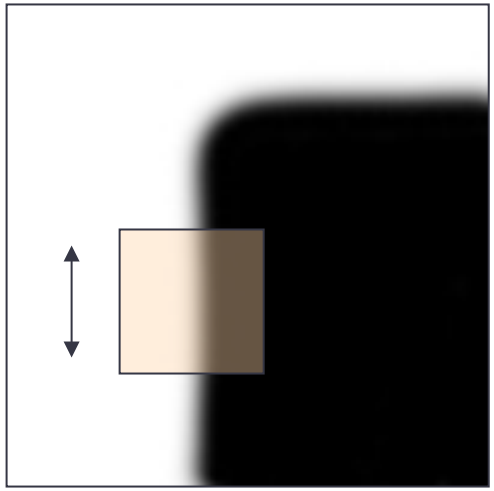
3) **Matching:** Determine correspondence between descriptors in two views

Corners as distinctive interest points

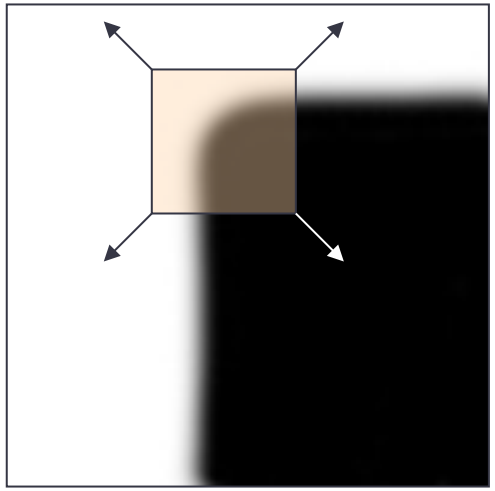
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in all directions

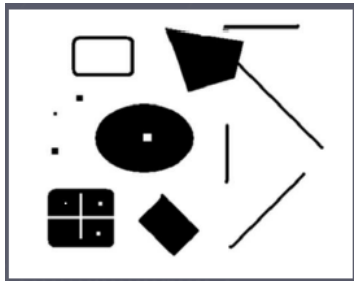


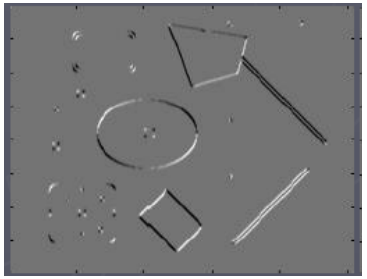
"edge":
no change along the edge direction



"corner":
significant change in all directions

Slide credit: Alyosha Efros, Darya Frolova, Denis Simakov





Notation:

$$I_x \Leftrightarrow \frac{\partial I}{\partial x} \qquad I_y \Leftrightarrow \frac{\partial I}{\partial y} \qquad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$$

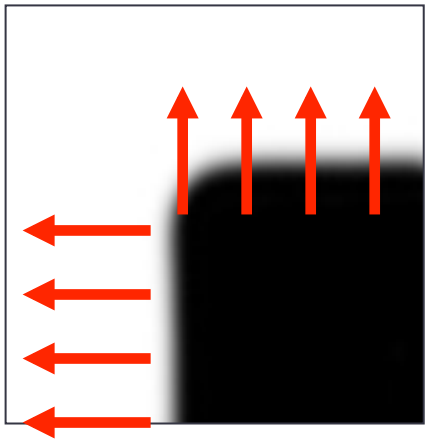
Definition: Corners are distinctive image interest points

$$M = \sum_{\text{Neighborhood of (x,y)}} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

2x2 matrix of image derivatives (averaged in pixel neighborhood).

What does this matrix reveal?

First, consider an axis-aligned corner:



What does this matrix reveal?

First, consider an axis-aligned corner:

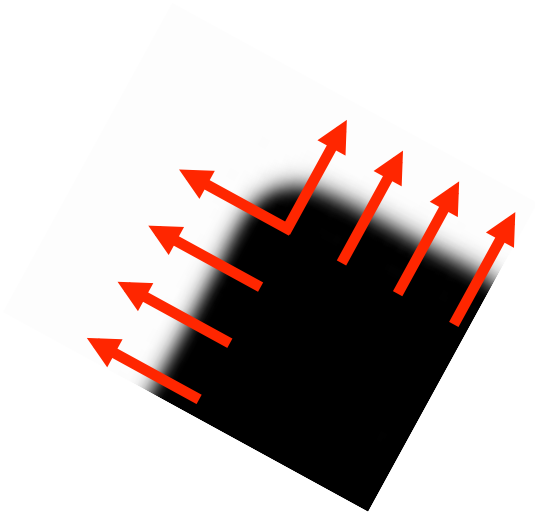
$$M = \sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

- What would be the λ_1 and λ_2, if in each pìxel of the neighbourhood I_y=0?
- This means that dominant gradient directions align with x or y axis
- If either λ is close to 0, then this is **not** corner-like.
- Look for locations where **both** λ's are large.

What if we have a corner that is not aligned with the image axes?

Corners - distinctive interest points

Since *M* is symmetric, we have $M = E\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}E^T$

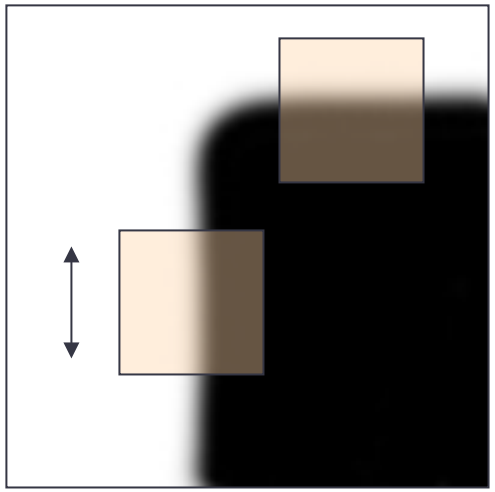


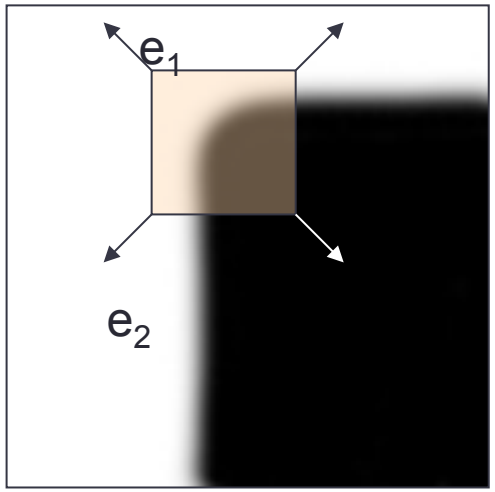
$$Me_i = \lambda_i e_i, E = [e_1, e_2]$$

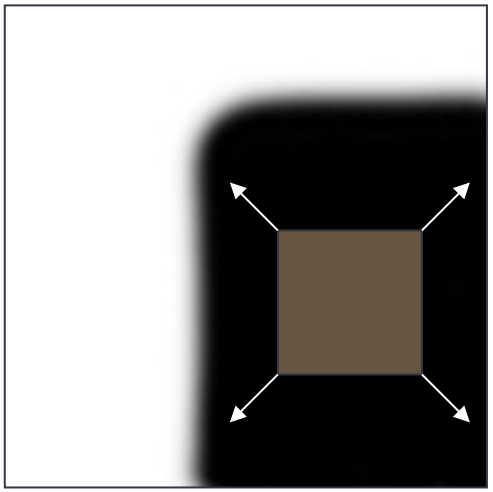
The *eigenvalues* of *M* (λ_1 and λ_2) reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.

The *eigenvectors* of M (e_1 and e_2) give the direction of maximum and minimum change in the image.

Corners as distinctive interest points







"edge":

$|\lambda_1| >> |\lambda_2|$
or

$|\lambda_2| >> |\lambda_1|$

"corner":

$|\lambda_1|$ and $|\lambda_2|$ are large, $\lambda_1 \sim \lambda_2$;

"flat" region

$|\lambda_1|$ and $|\lambda_2|$ are small;

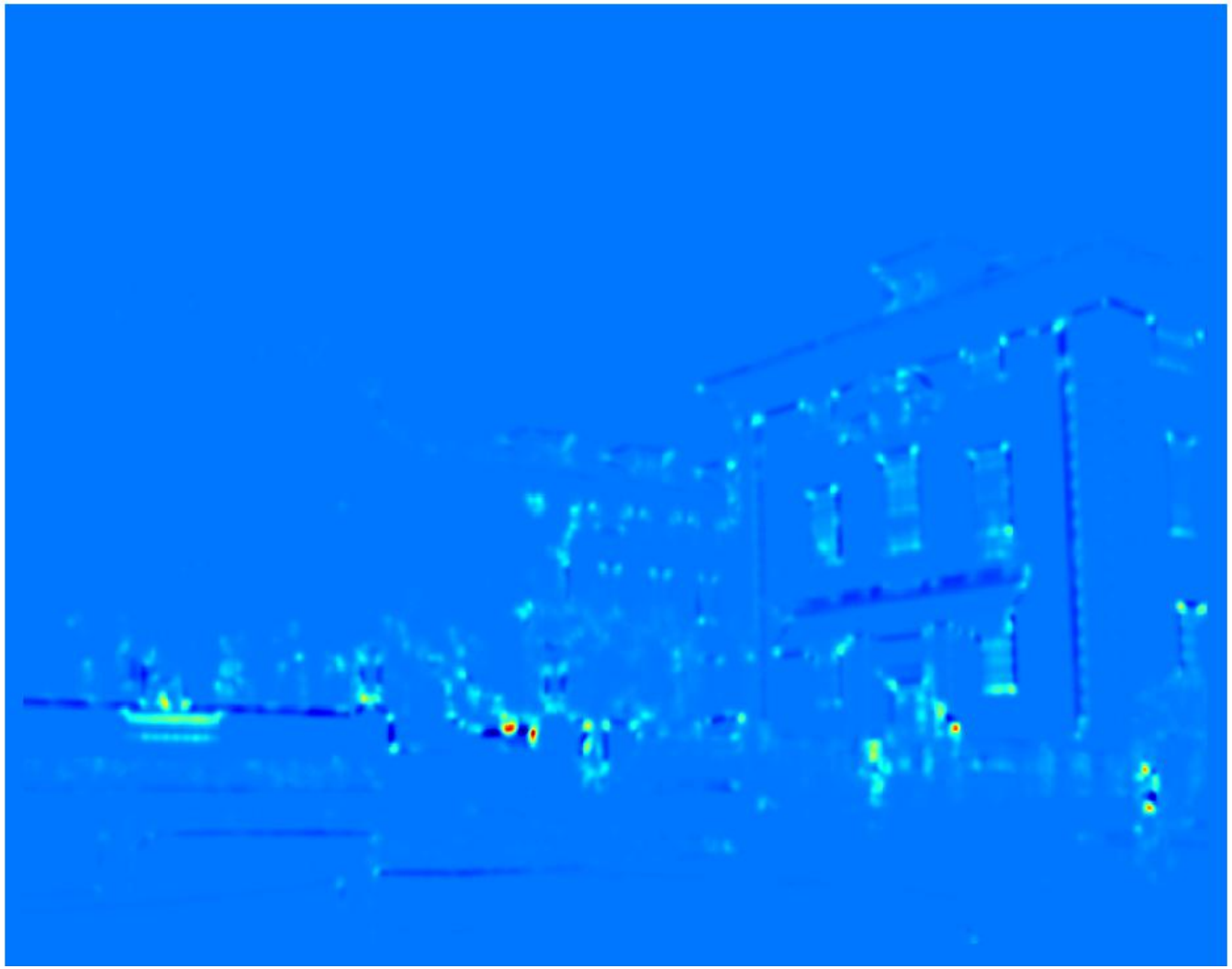
Harris corners: 'cornerness' score:

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

- What points would you choose?

Example of Harris application

- Compute corner response at every pixel.



Example of Harris application

Kristen Grauman

Harris corner detector

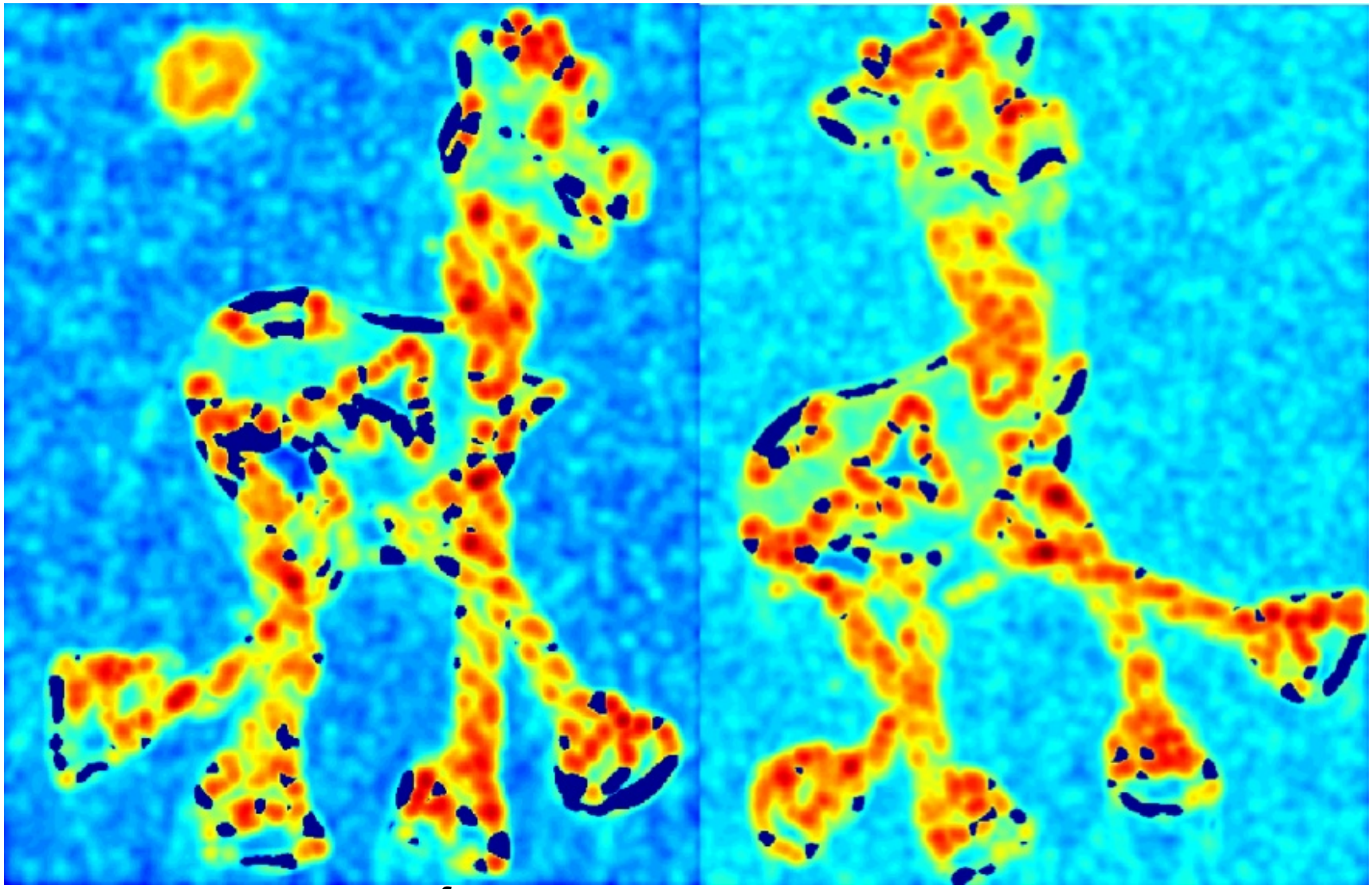
1) For each pixel of the image, compute matrix *M* for image window surrounding it to get its *cornerness* score.

2) Find points with large corner response (f > threshold).

3) Take the points of local maxima for each window, i.e. perform non-maximum suppression.

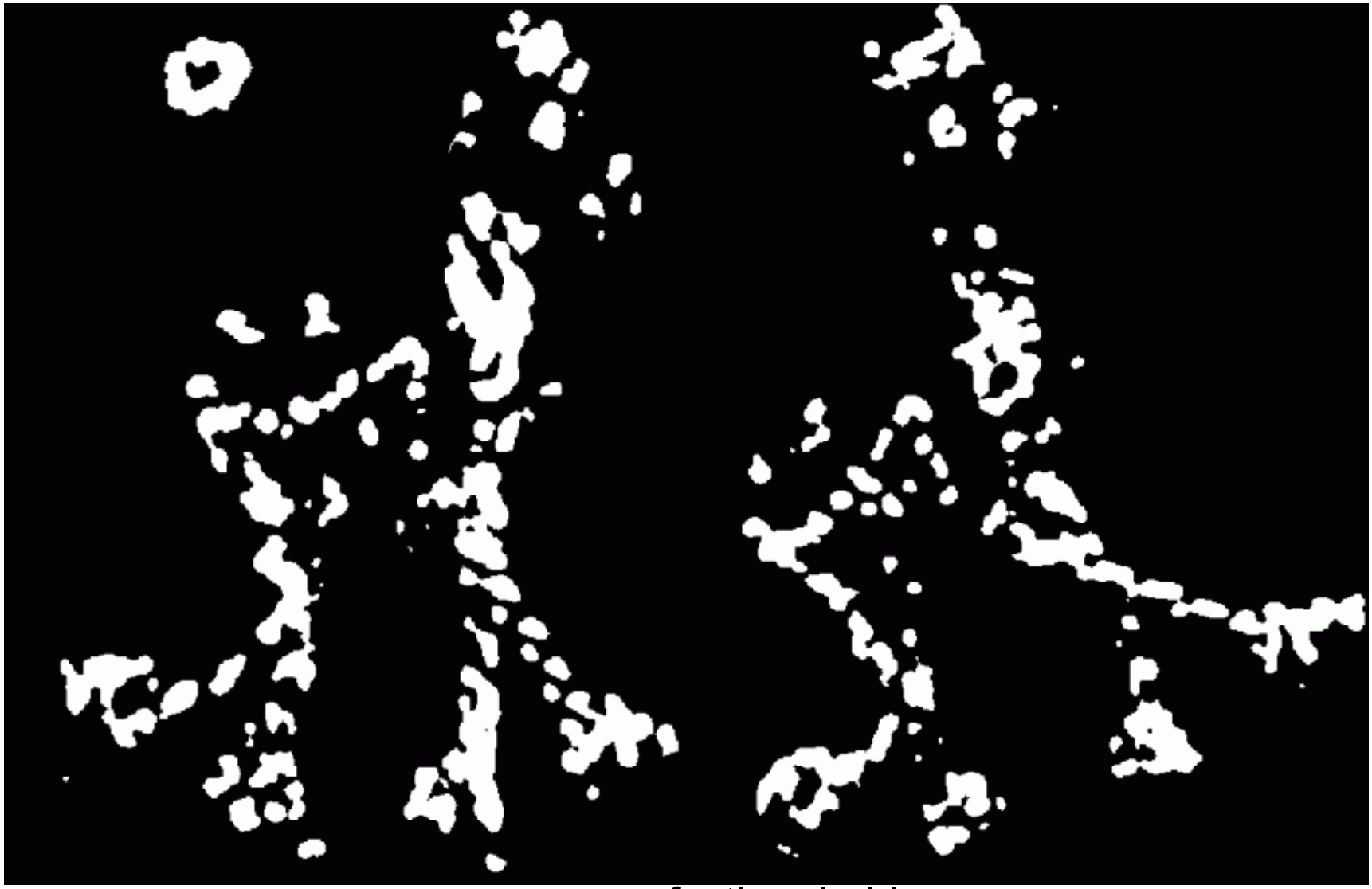
Harris Detector: Steps

Harris Detector: Cornesness evaluation



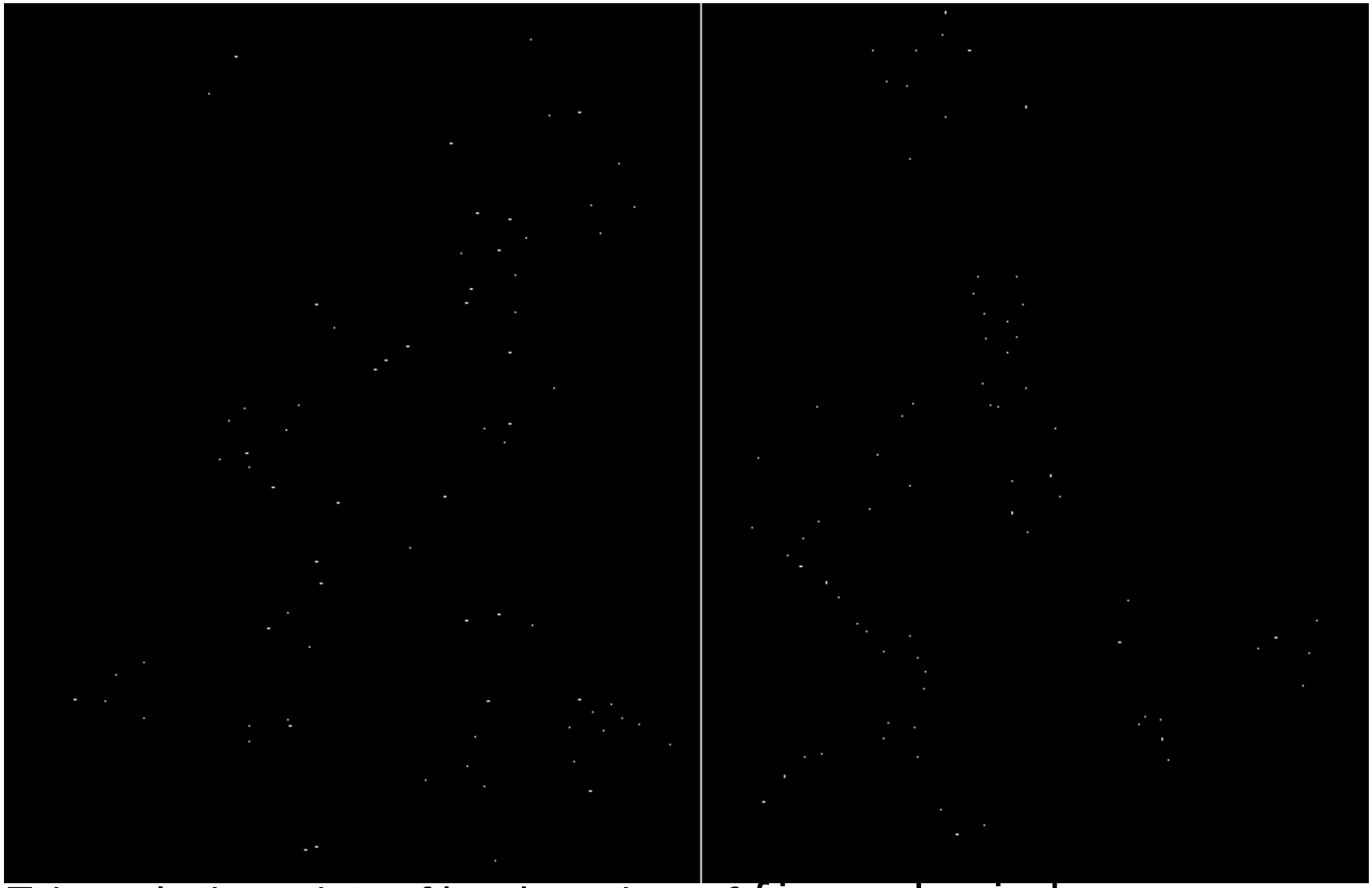
Compute corner response *f*

Harris Detector: Steps



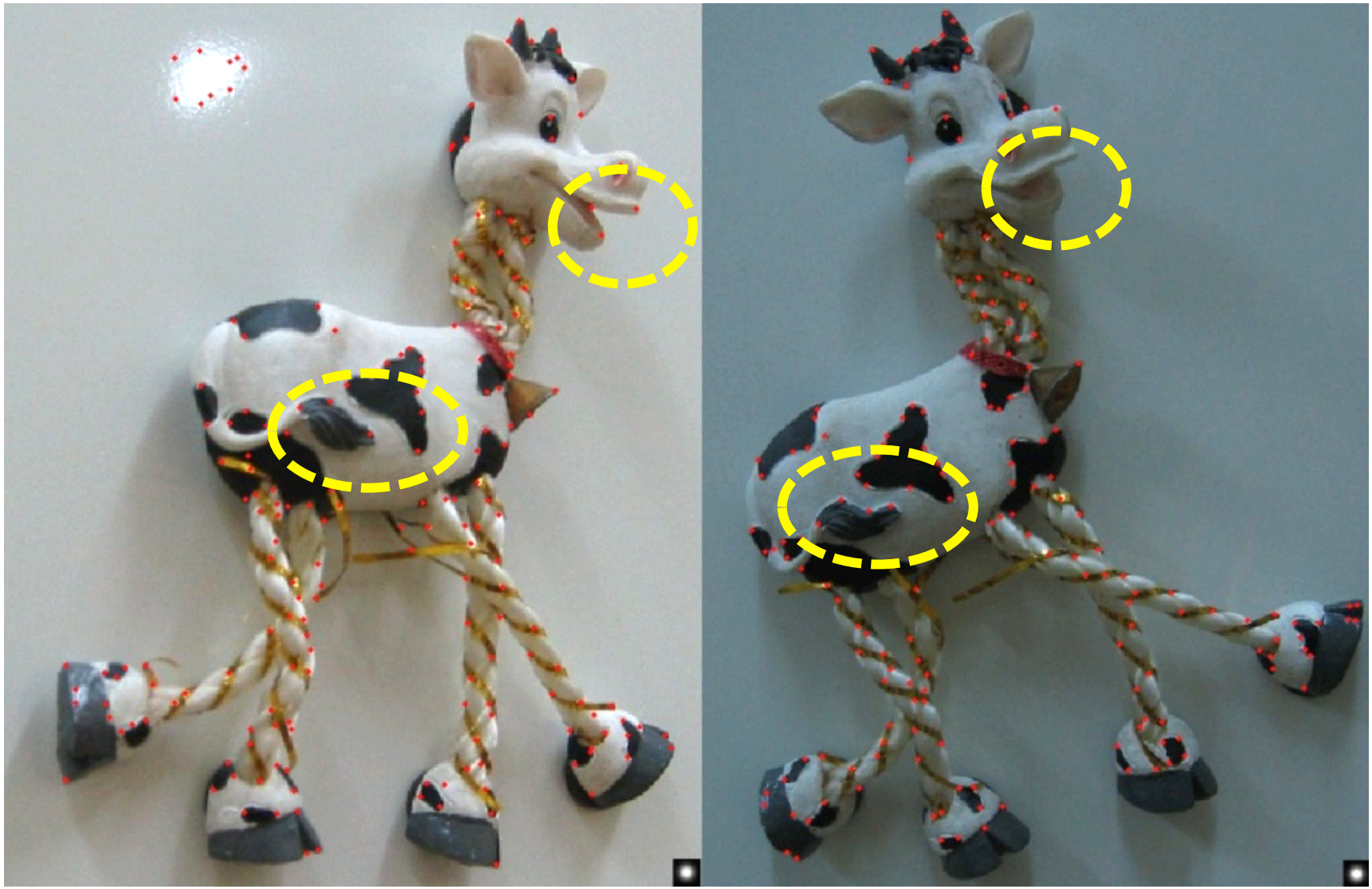
Find points with large corner response: f > threshold

Harris Detector: Steps



Take only the points of local maxima of *f* in each window

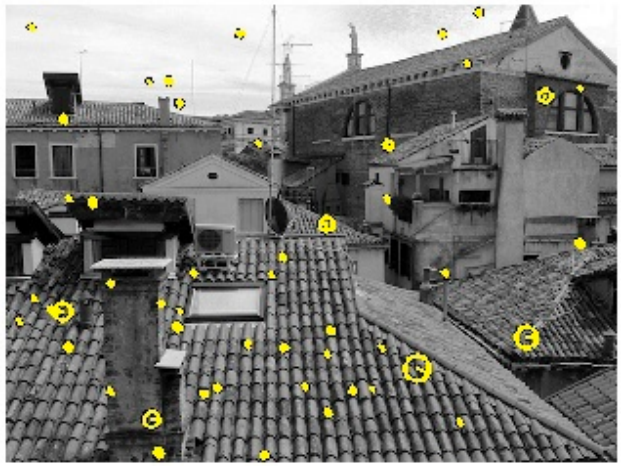
Harris Detector: Steps

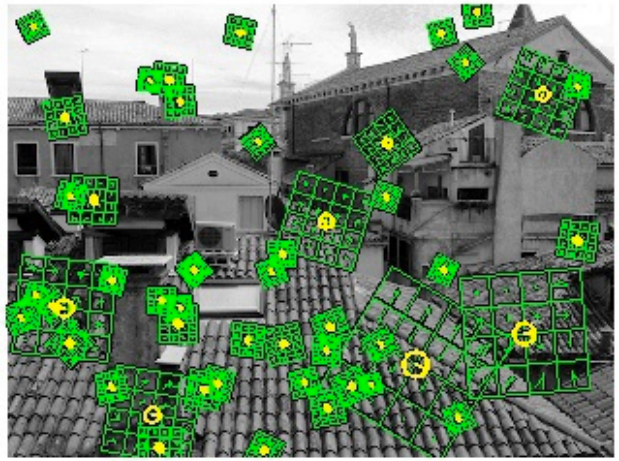


Sift in Matlab

We will use the library: vlfeat-0.9.20/toolbox (www.vlfeat.org)

```
[f,d] = vl_sift(I) ;
```

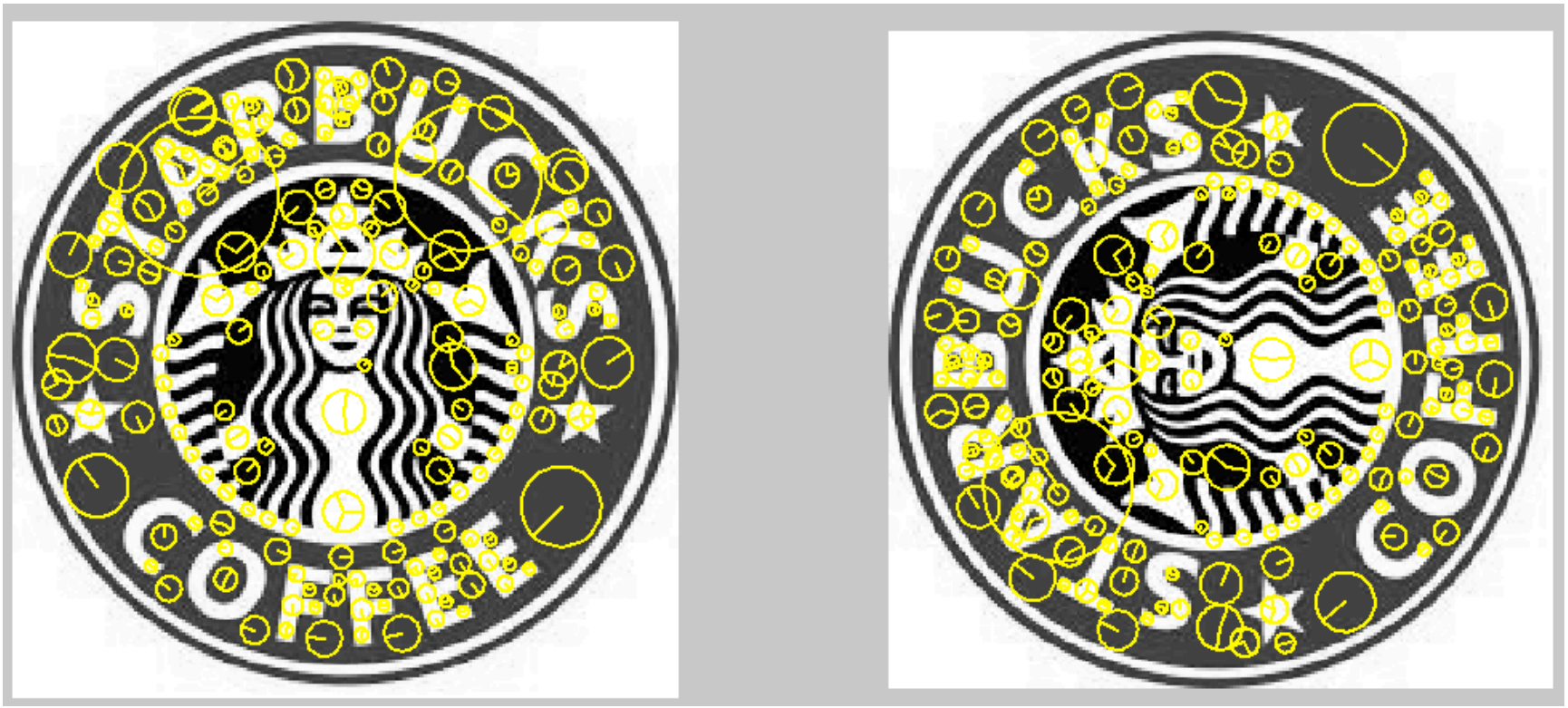




```
perm = randperm(size(f,2)) ;
sel = perm(1:50) ;
h1 = vl_plotframe(f(:,sel)) ;
h2 = vl_plotframe(f(:,sel)) ;
set(h1,'color','k','linewidth',3) ;
set(h2,'color','y','linewidth',2) ;
```

```
h3 = vl_plotsiftdescriptor(d(:,sel),f(:,sel)) ;
set(h3,'color','g') ;
```

Exercise: Explain the Sift invariance



Properties of the Harris corner detector

- Rotation invariant?

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

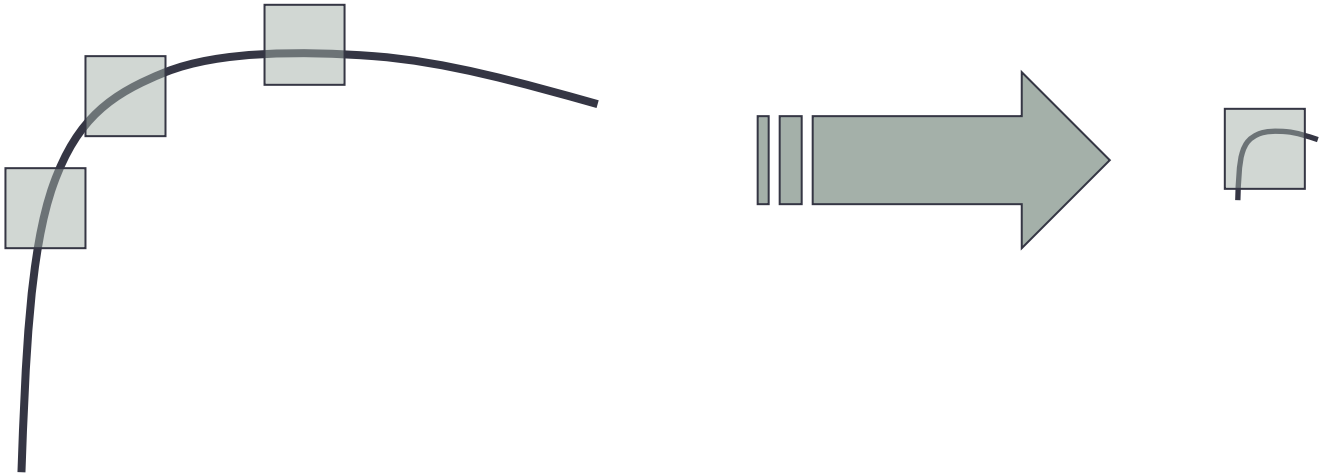
Yes

- Scale invariant?

Properties of the Harris corner detector

- Rotation invariant? Yes

- Scale invariant? No



All points will be classified as edges

Corner !

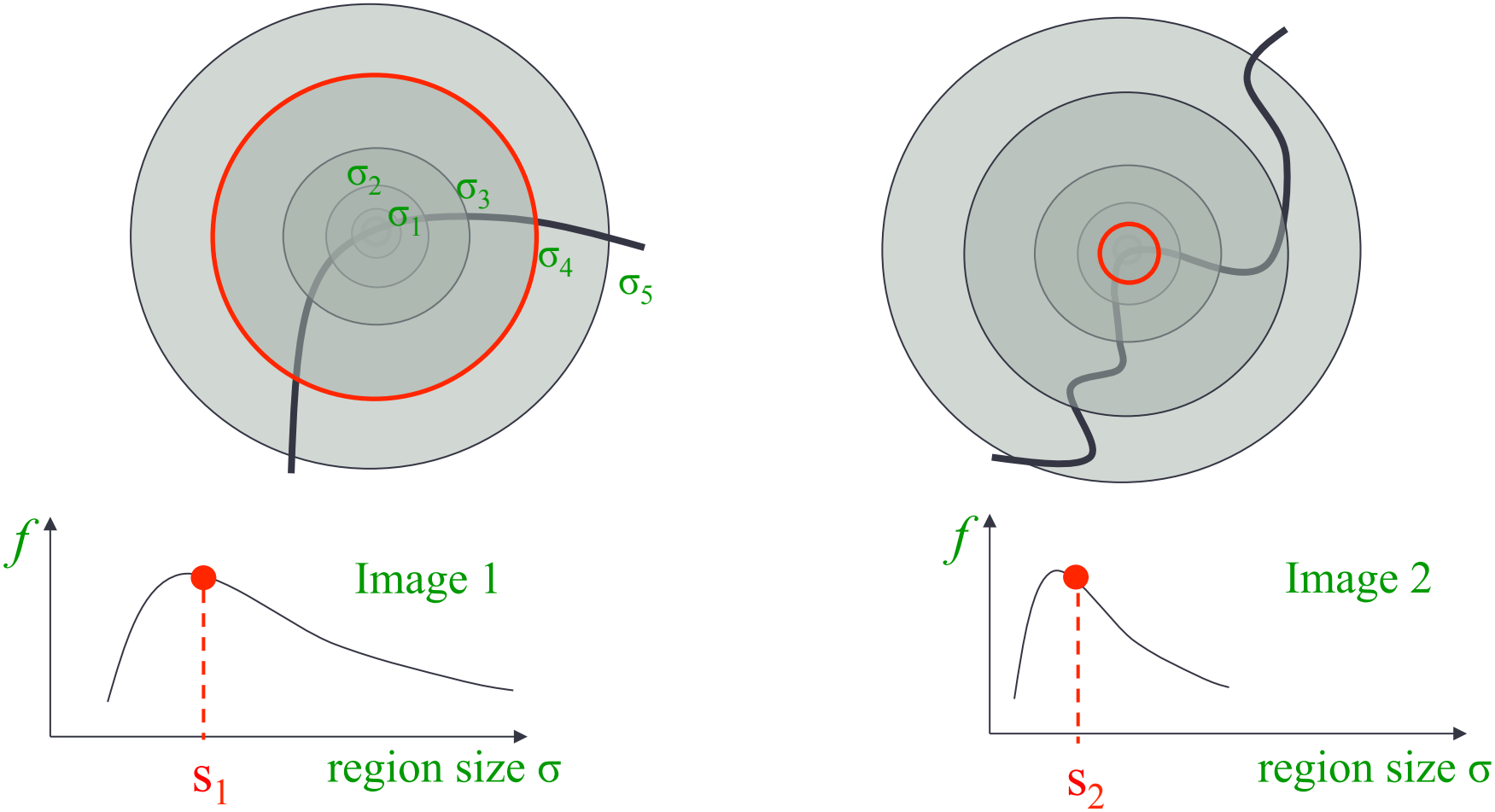
Scale invariant interest points

How can we independently select interest points in each image, such that the detections are repeatable across different scales?

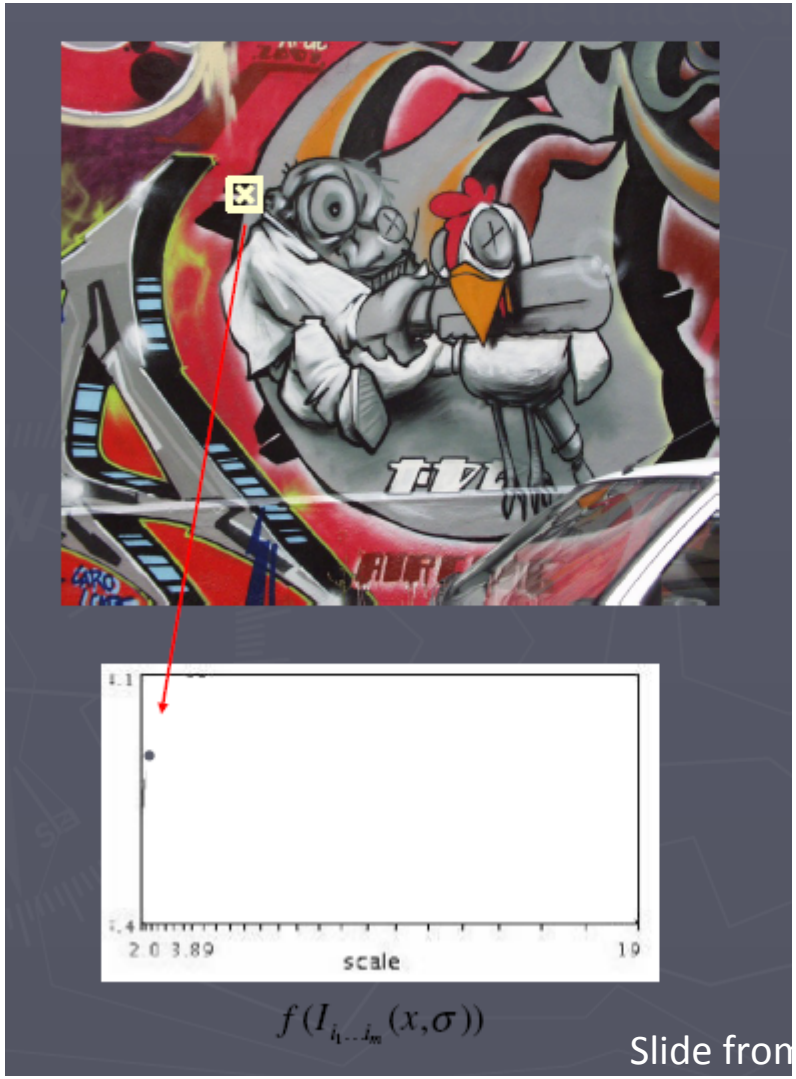
Automatic scale selection

Intuition:

- Compute the derivatives (I_x,I_y) with different sigma, and find the scale that gives local maxima of the *cornerness* function *f* in both position and scale.

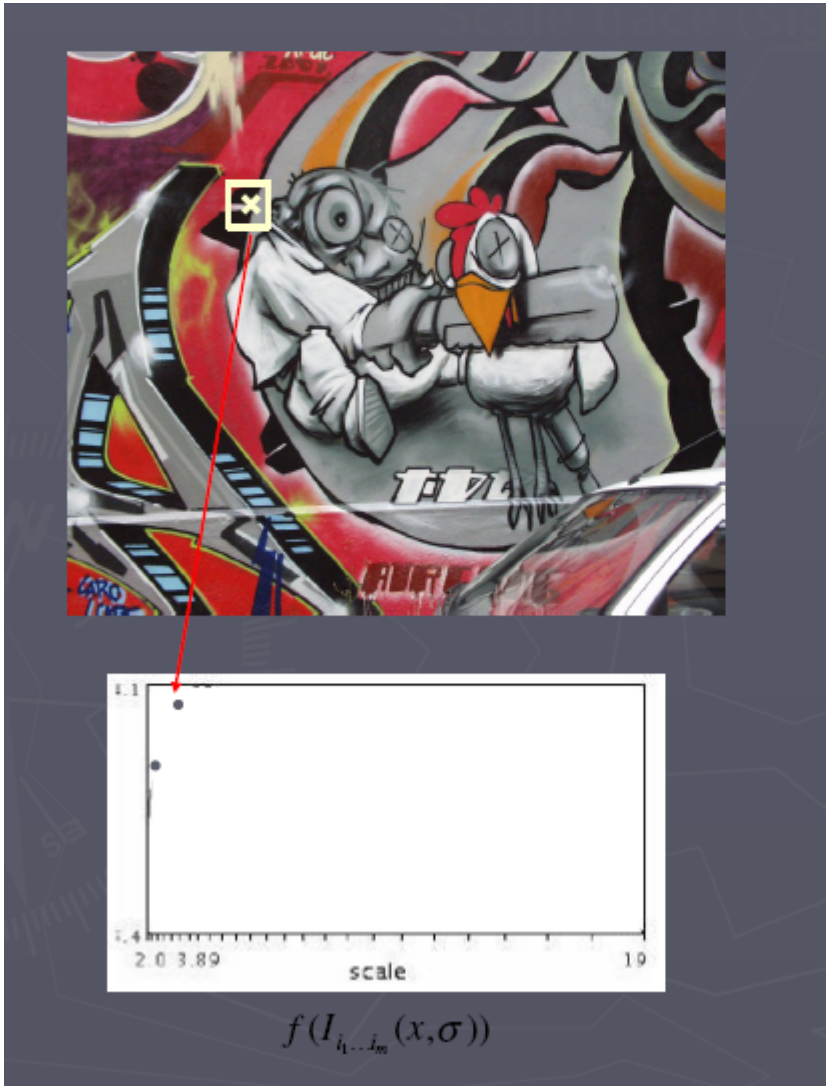


Automatic scale detection



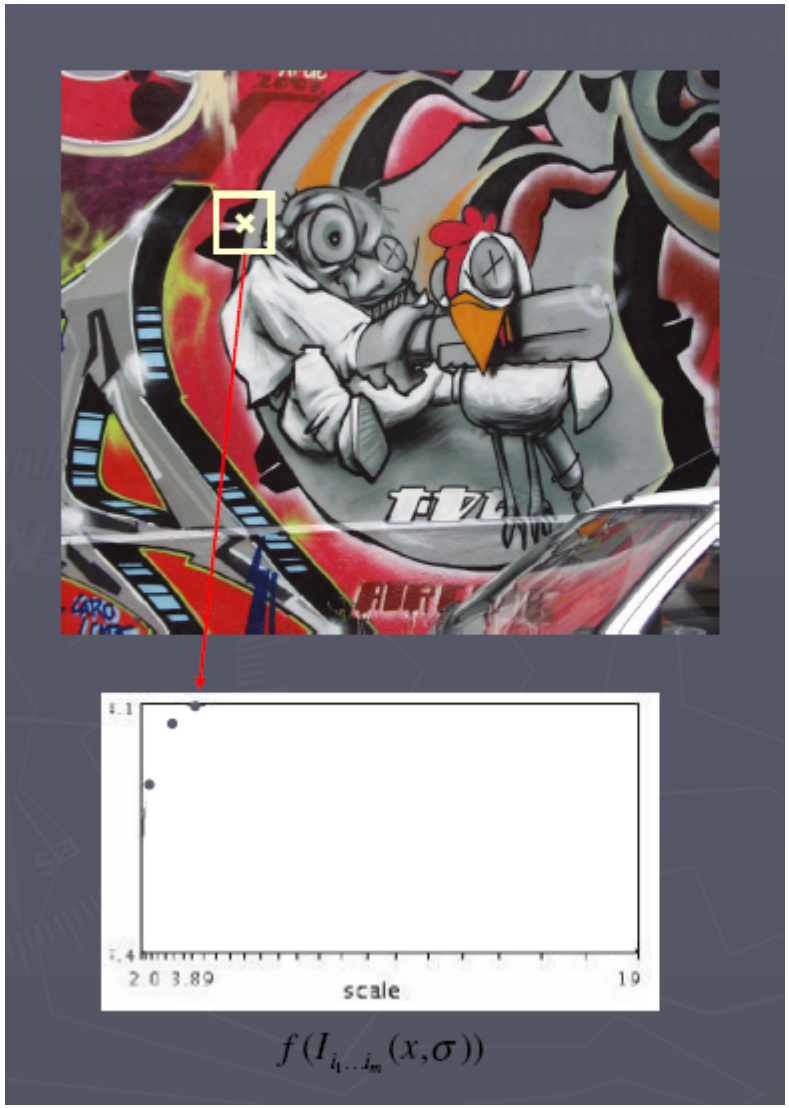
Slide from

Automatic scale detection



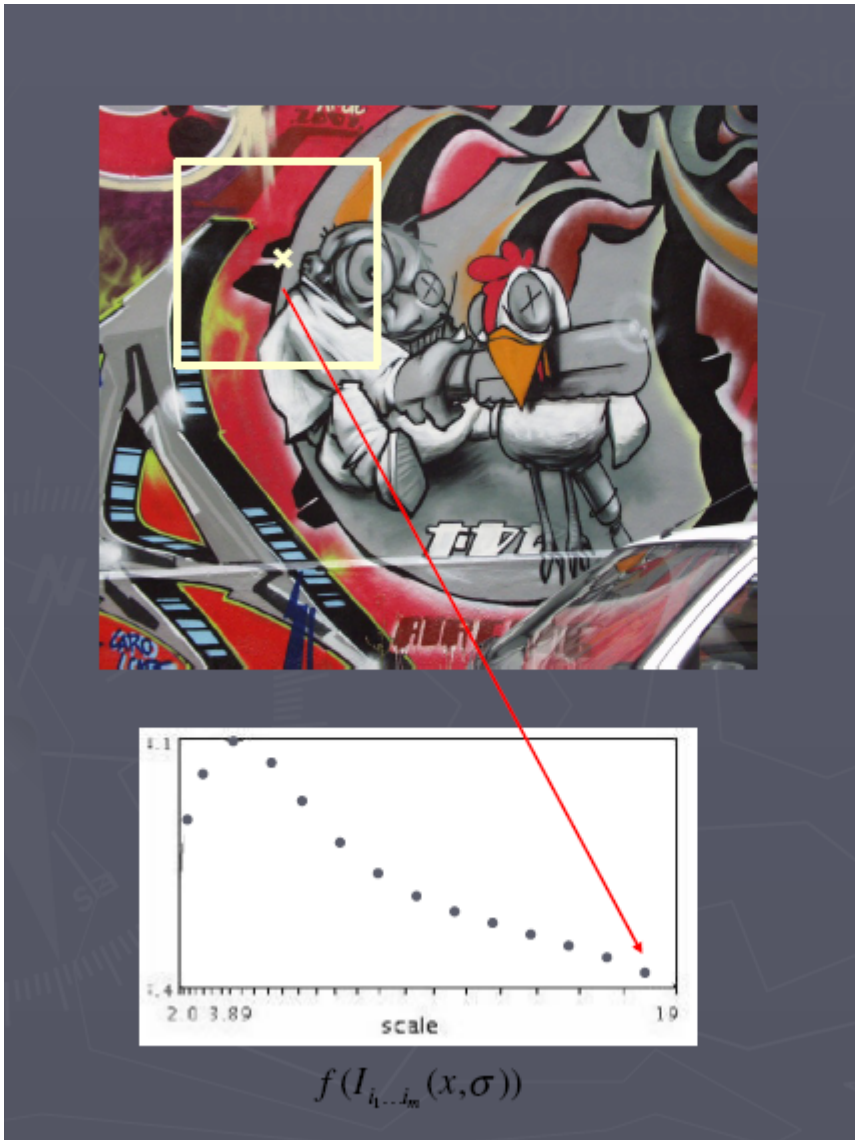
$f(I_{i_1 \ldots i_m}(x,\sigma))$

Automatic scale detection

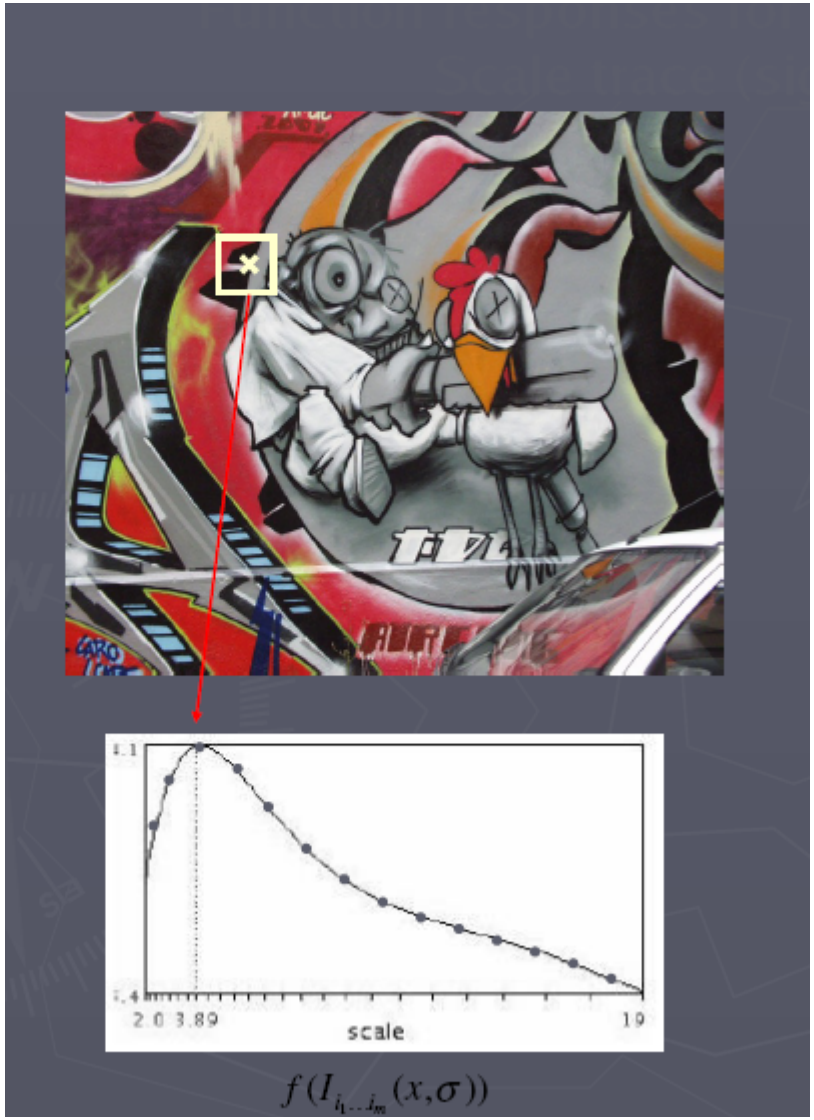


Automatic scale detection

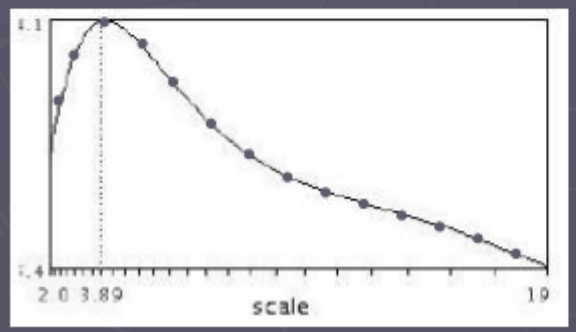
Automatic scale detection



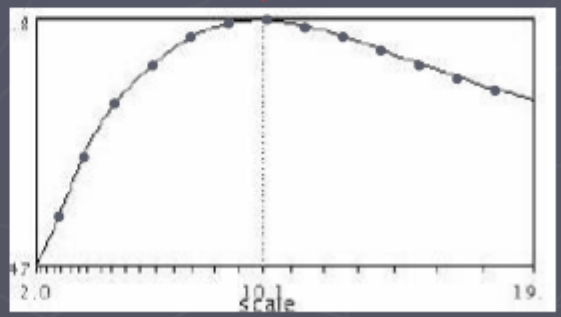
Automatic scale detection



Automatic scale detection

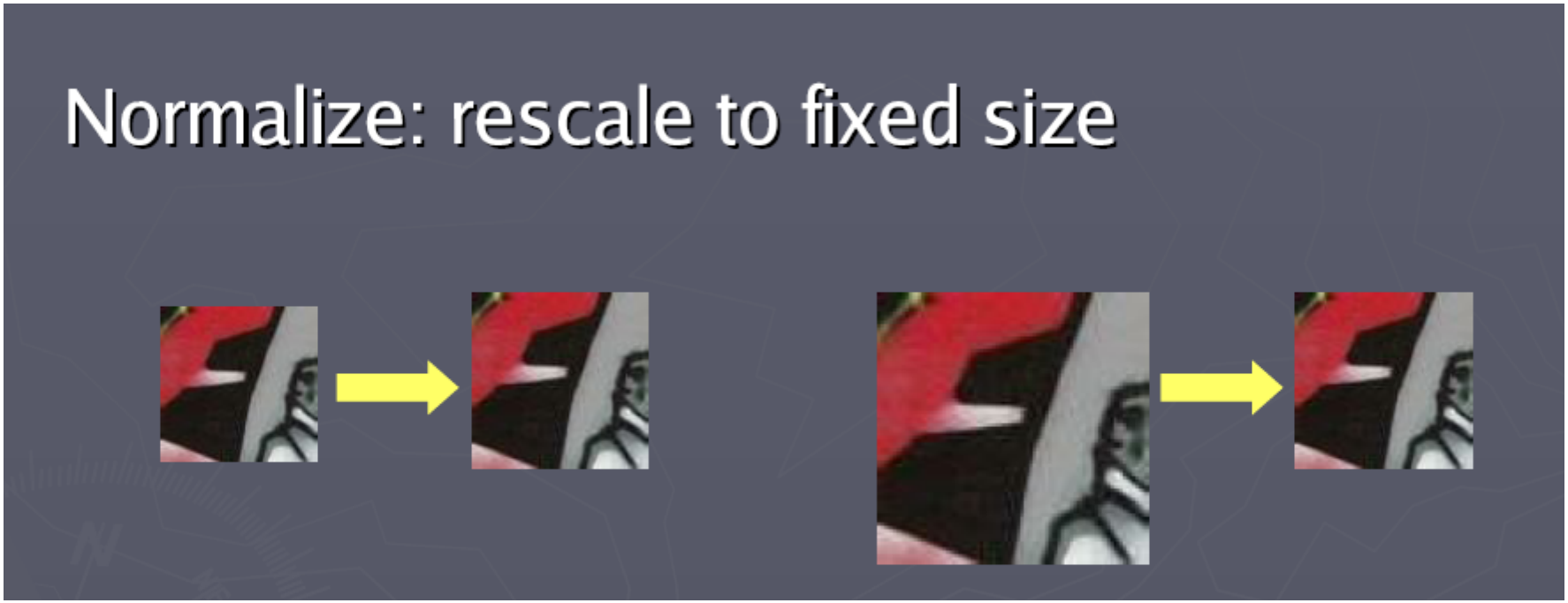


$f(I_{i_1 \ldots i_m}(x,\sigma))$



$f(I_{i_1 \ldots i_m}(x',\sigma'))$

Automatic scale detection



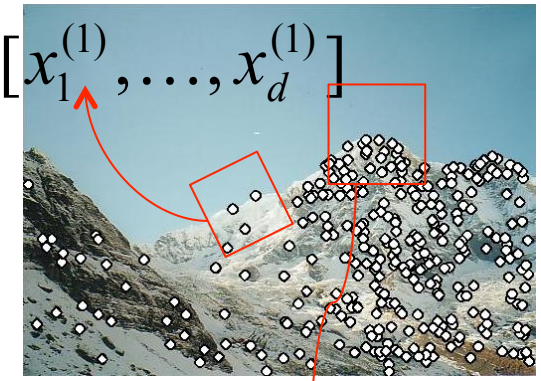
Exercise: Explain the Sift appearance

Local features: main components

1) Detection: Identify the interest points

2) **Description**: Extract feature descriptor vector surrounding each interest point.

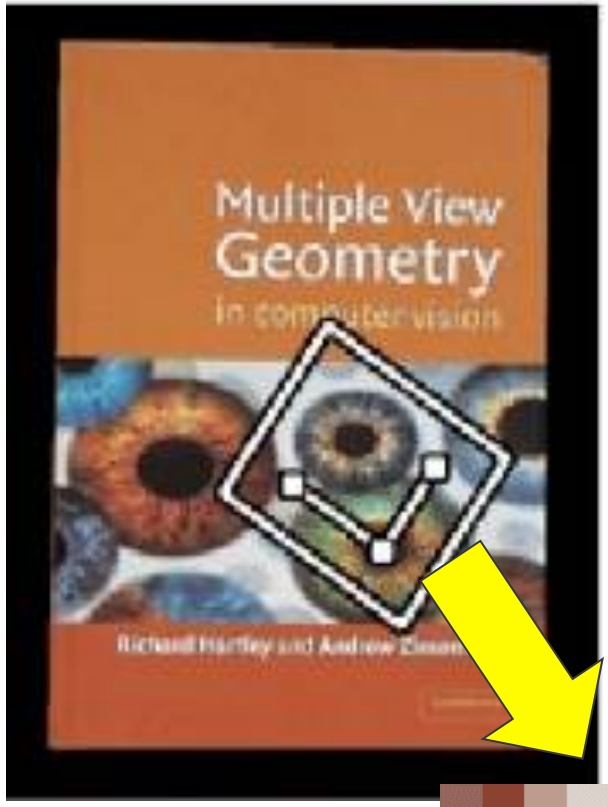
$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

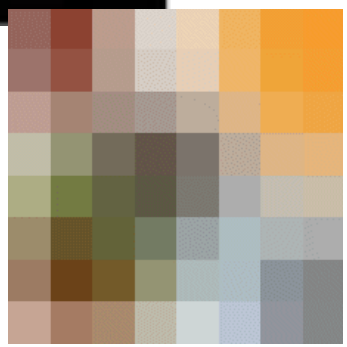


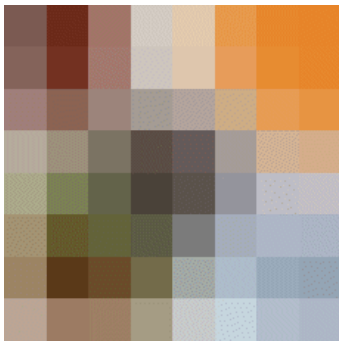
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) Matching: Determine correspondence between descriptors in two views

Geometric transformations

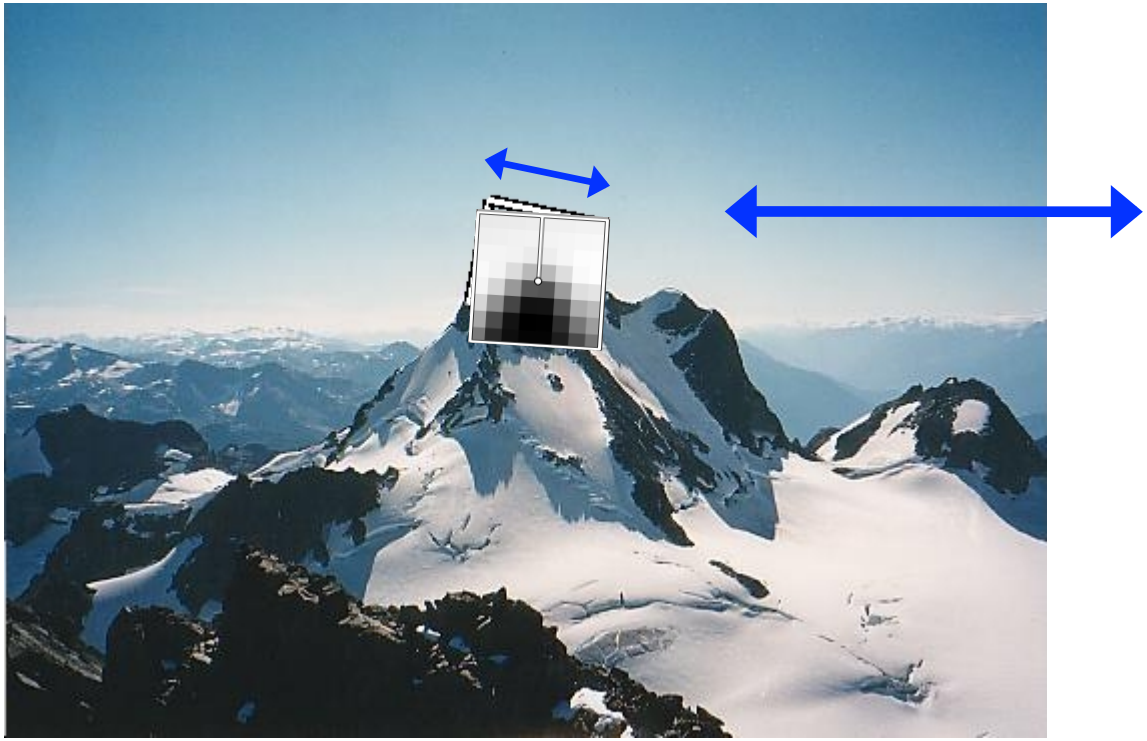






e.g. scale, translation, rotation

Making the descriptor rotationally invariant



- Rotate patch according to its dominant gradient orientation
  - It can be shown that the dominant direction is given by the first eigenvector (e_1, if $|\lambda_1|>|\lambda_2|$, and e_2, vice versa) of M.

- This puts the patches into a canonical orientation.

Image from Matthew Brown

Photometric transformations

Figure from T. Tuytelaars ECCV 2006 tutorial

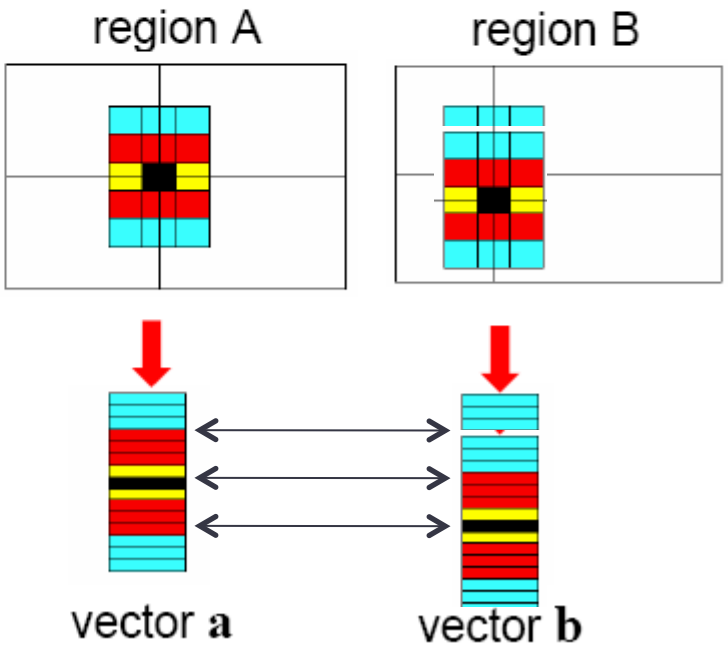
Illumination invariance

What does happen when image illumination change?

- Image gradient magnitude?
- Image gradient direction?

=> Define Sift descriptors based on Gradient direction!

Raw patches as local descriptors

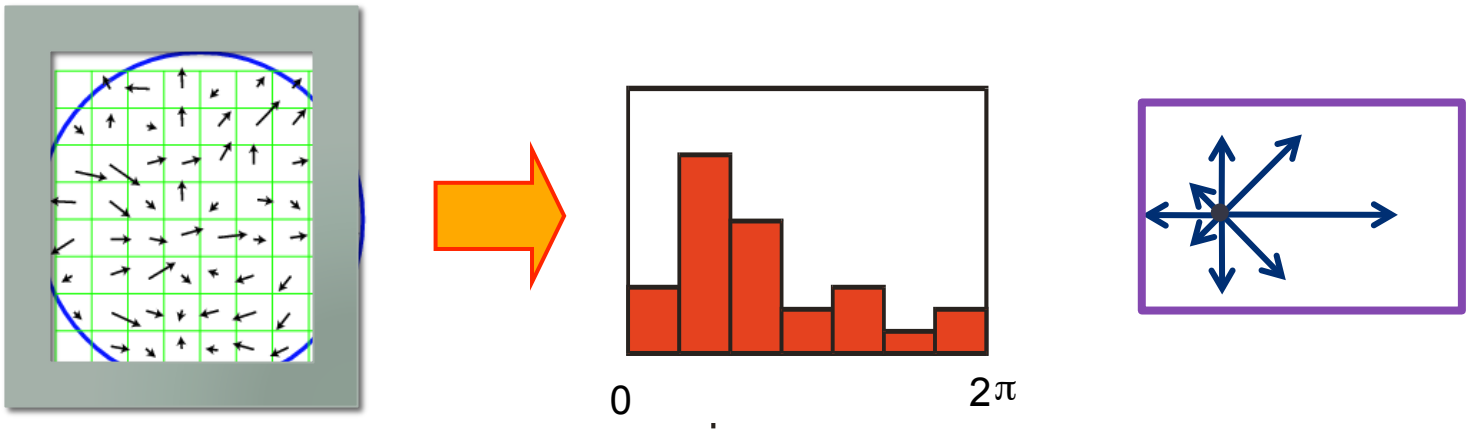


The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, and illumination changes.

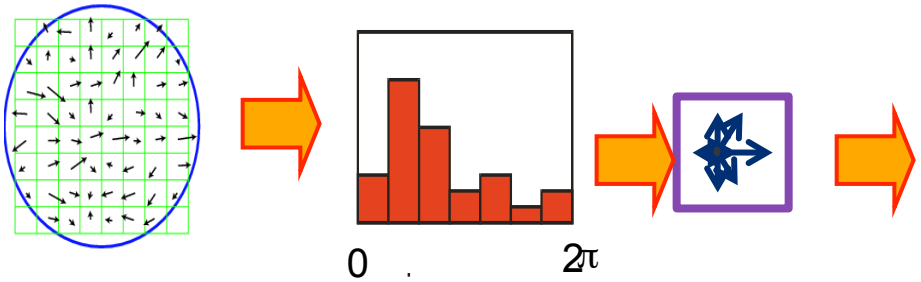
SIFT descriptor [Lowe 2004]

- Use histograms to bin pixels within sub-patches instead of just pixels according to their orientation.

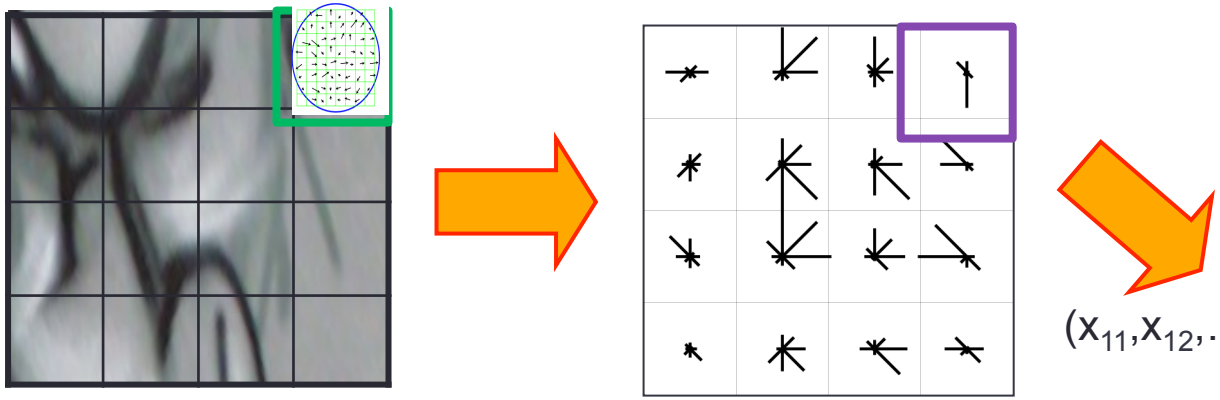


SIFT descriptor [Lowe 2004]

- Use histograms to bin pixels within sub-patches according to their orientation.



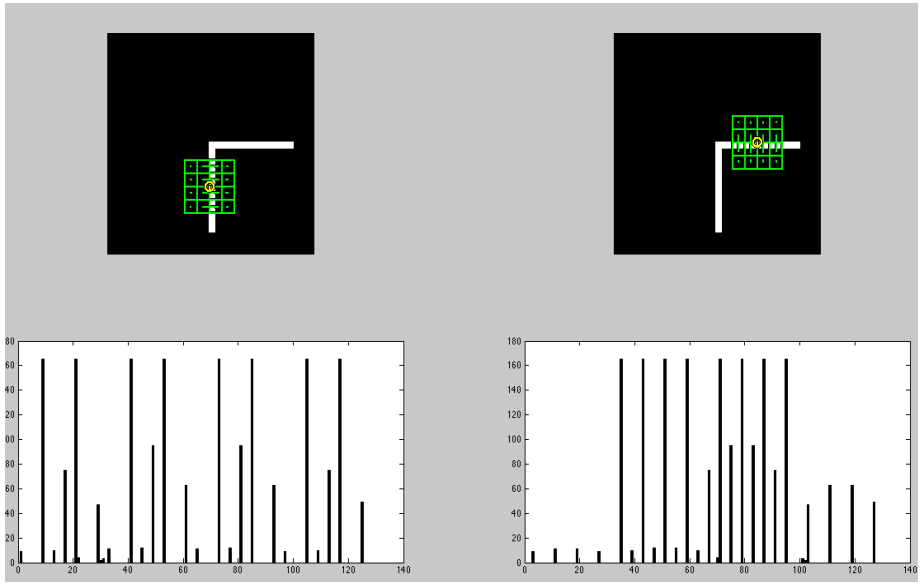
$(x_1, x_2, \ldots x_n)$



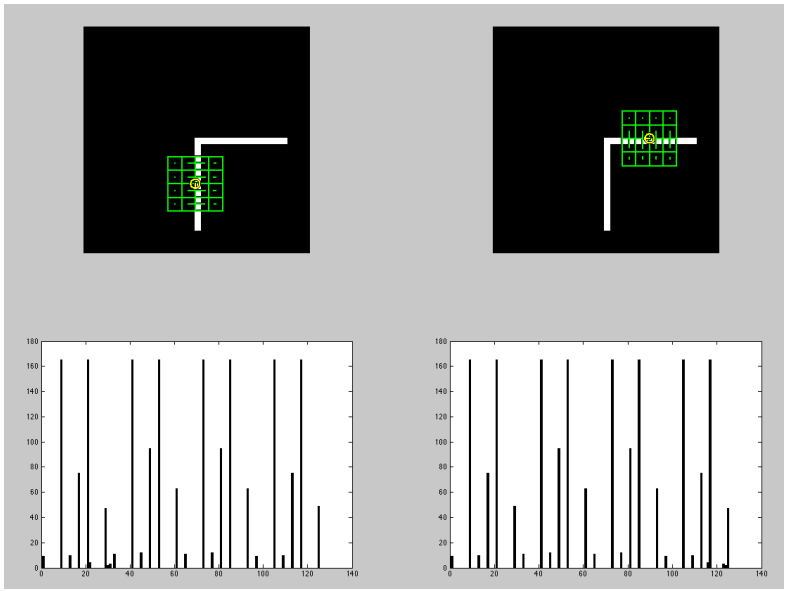
Why subpatches?

$(x_{11}, x_{12}, \ldots x_{1n}\ x_{21}, x_{22}, \ldots x_{2n},\ x_{m1}, x_{m2}, \ldots x_{mn})$

Exercise: Explain the Sift descriptors



```
f_edge1=[50;70;2;0];
f_edge2=[70;50;2;0];
[f1,d1] = vl_sift(testim,'frames',f_edge1);
[f2,d2] = vl_sift(testim,'frames',f_edge2);
```



```
[f1,d1] = vl_sift(testim,'frames',f_edge1,'orientations');
[f2,d2] = vl_sift(testim,'frames',f_edge2,'orientations');
```

What is exactly the difference between HOG and SIFT?

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

Matching local features

Kristen Grauman

Matching local features

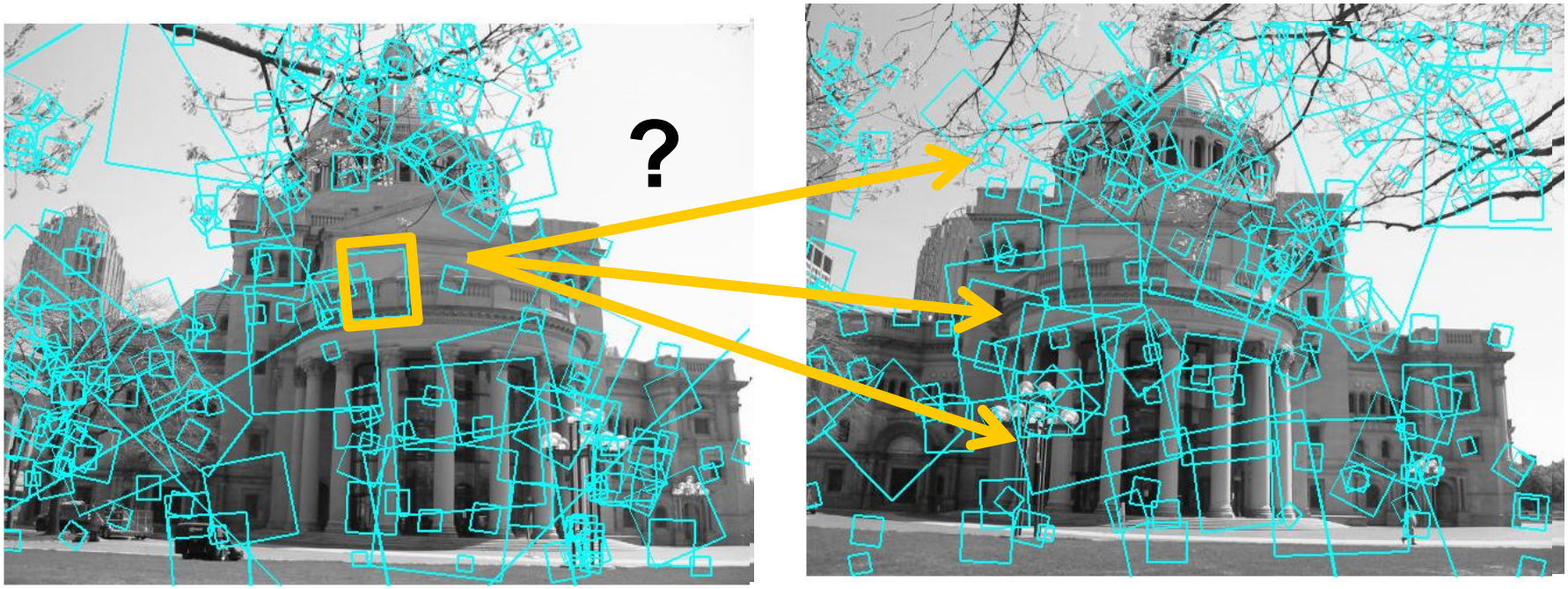


Image 1 Image 2

To generate **candidate matches**, find patches that have the most similar appearance (e.g., lowest Sum of Squared Distances (SSD)).

Simplest approach: compare them all, take the closest (or closest k, or within a thresholded distance).

Ambiguous matches

Image 1

Image 2

At what SSD value do we have a good match?

To add robustness to matching, we can consider the **ratio** : distance to best match / distance to second best match

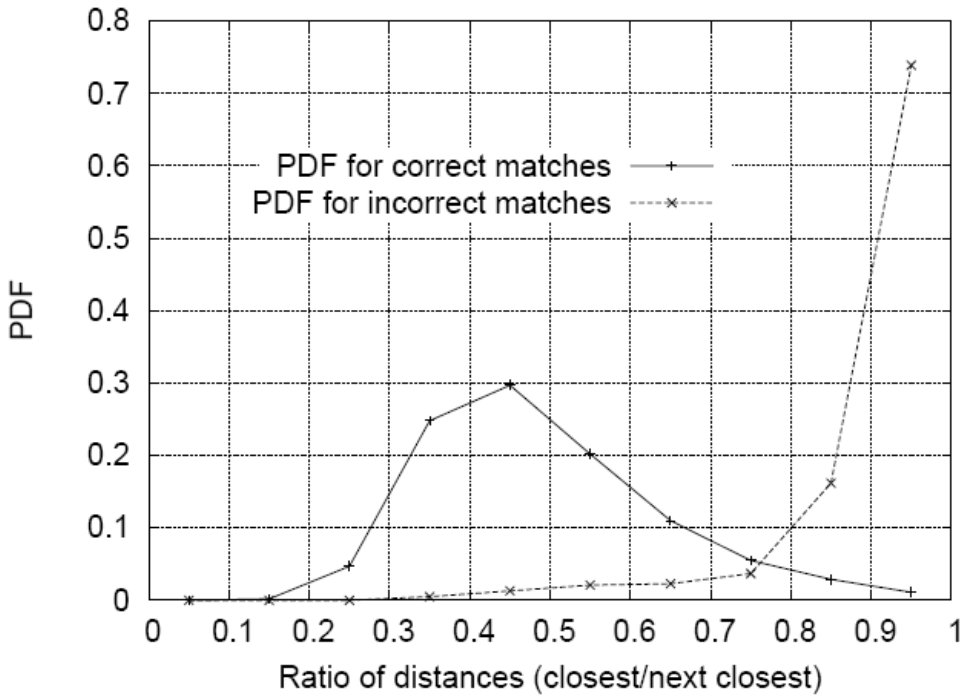
If low, first match looks good.

If high, could be ambiguous match.

Kristen Grauman

Matching SIFT Descriptors

- Nearest neighbor (Euclidean distance)
- Threshold ratio of nearest to 2nd nearest descriptor

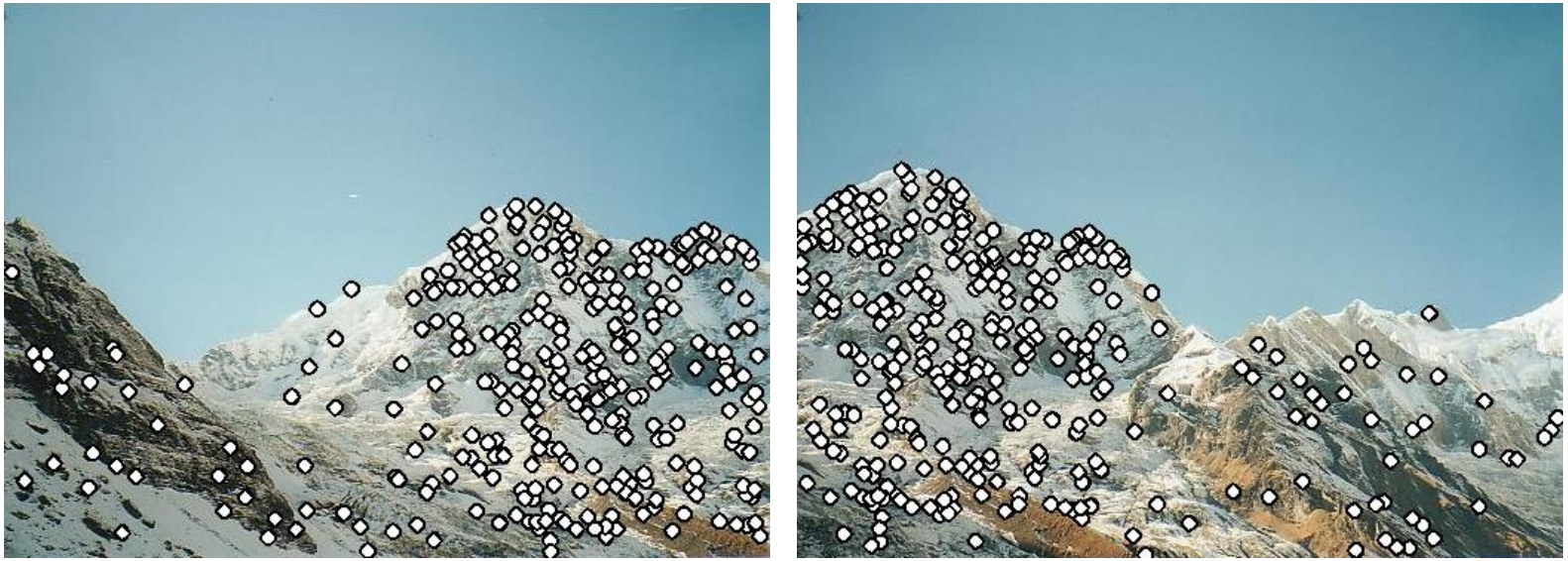


Lowe IJCV 2004

Robust feature-based alignment

Source: L. Lazebnik

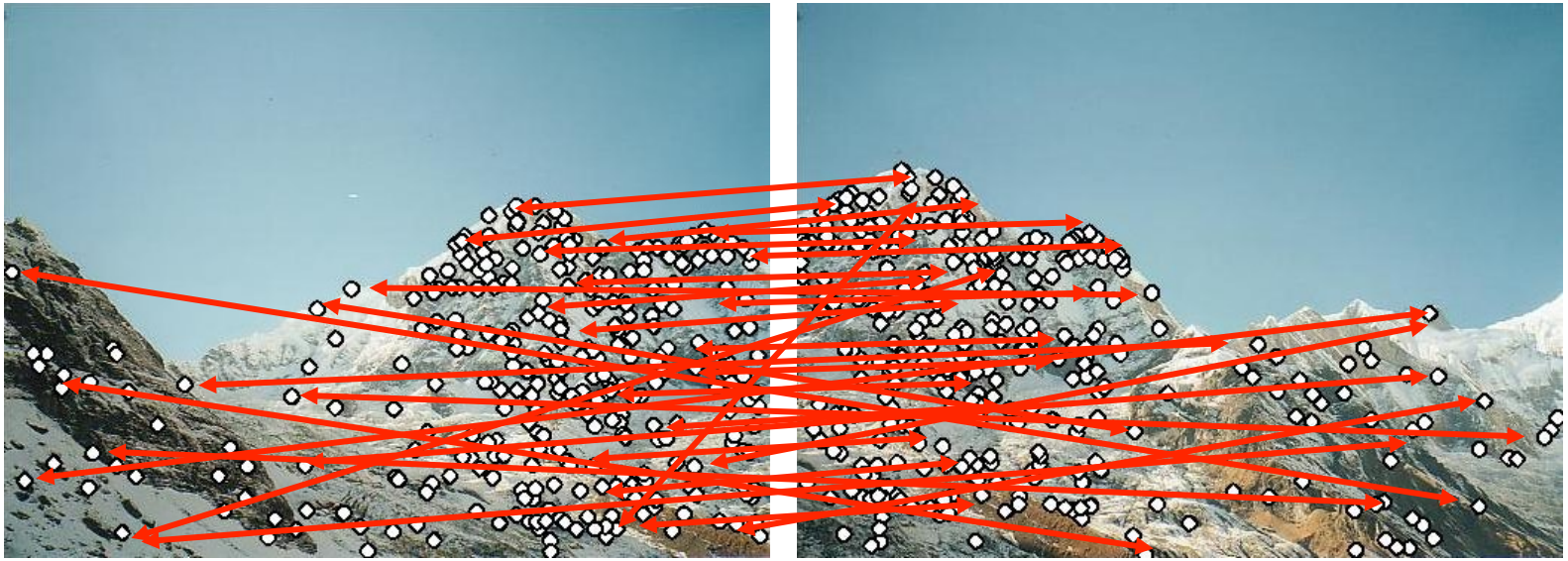
Robust feature-based alignment



- Extract features

Source: L. Lazebnik

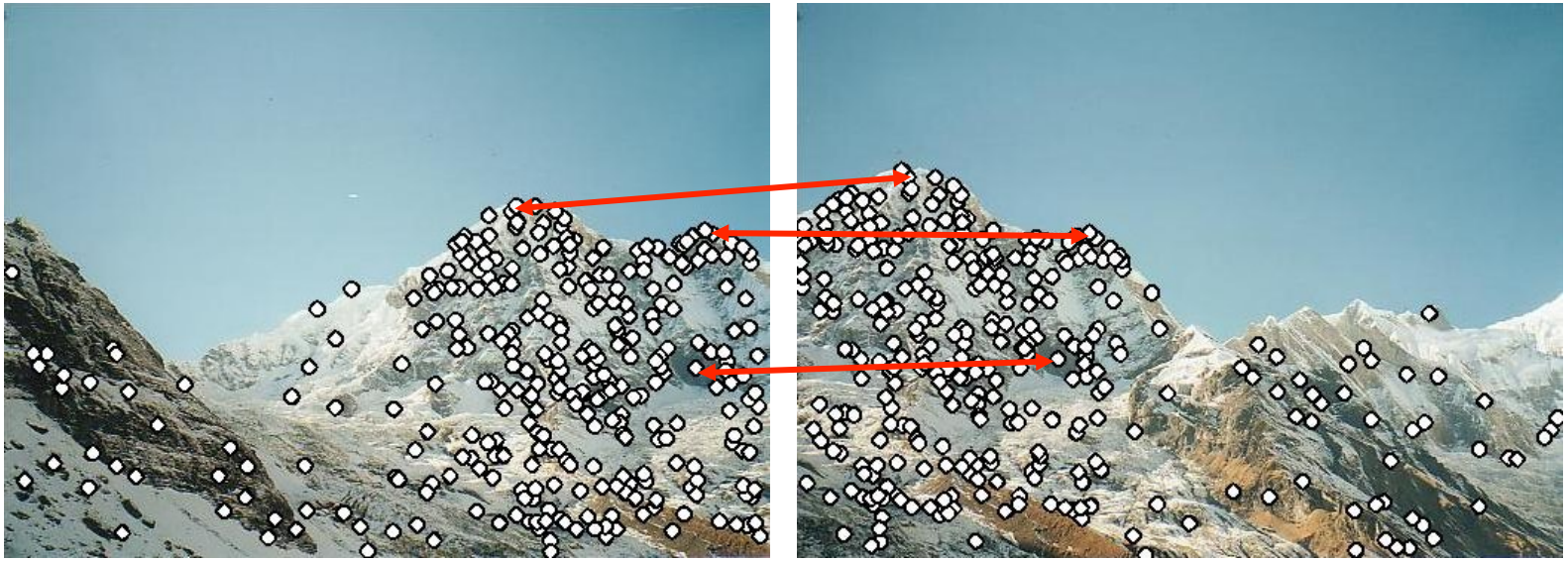
Robust feature-based alignment



- Extract features
- Compute *putative matches*

Source: L. Lazebnik

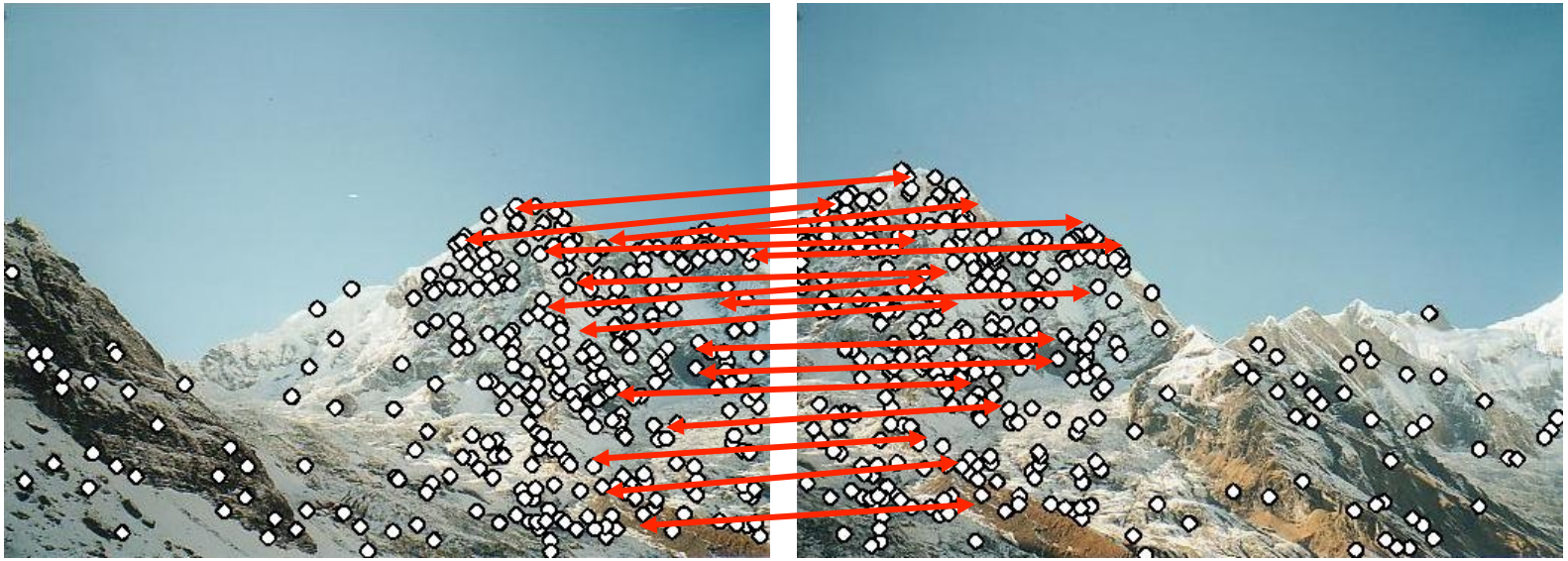
Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)

Source: L. Lazebnik

Robust feature-based alignment



- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)
  - *Verify* transformation (search for other matches consistent with *T*)

Source: L. Lazebnik

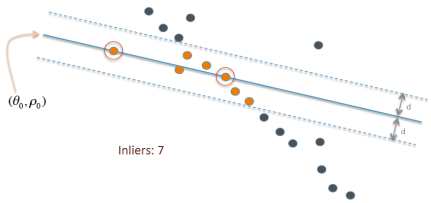
Robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)
  - *Verify* transformation (search for other matches consistent with *T*)

Source: L. Lazebnik

Mosaic construction by the Ransac method

Select a random subset of the original data (hypothetical inliers).



A model is fitted to the set of hypothetical inliers.

All other data are then tested against the fitted model.

Those points that fit the estimated model well, according to some model-specific loss function, are considered as part of the consensus set.

The estimated model is reasonably good if sufficiently many points have been classified as part of the consensus set.

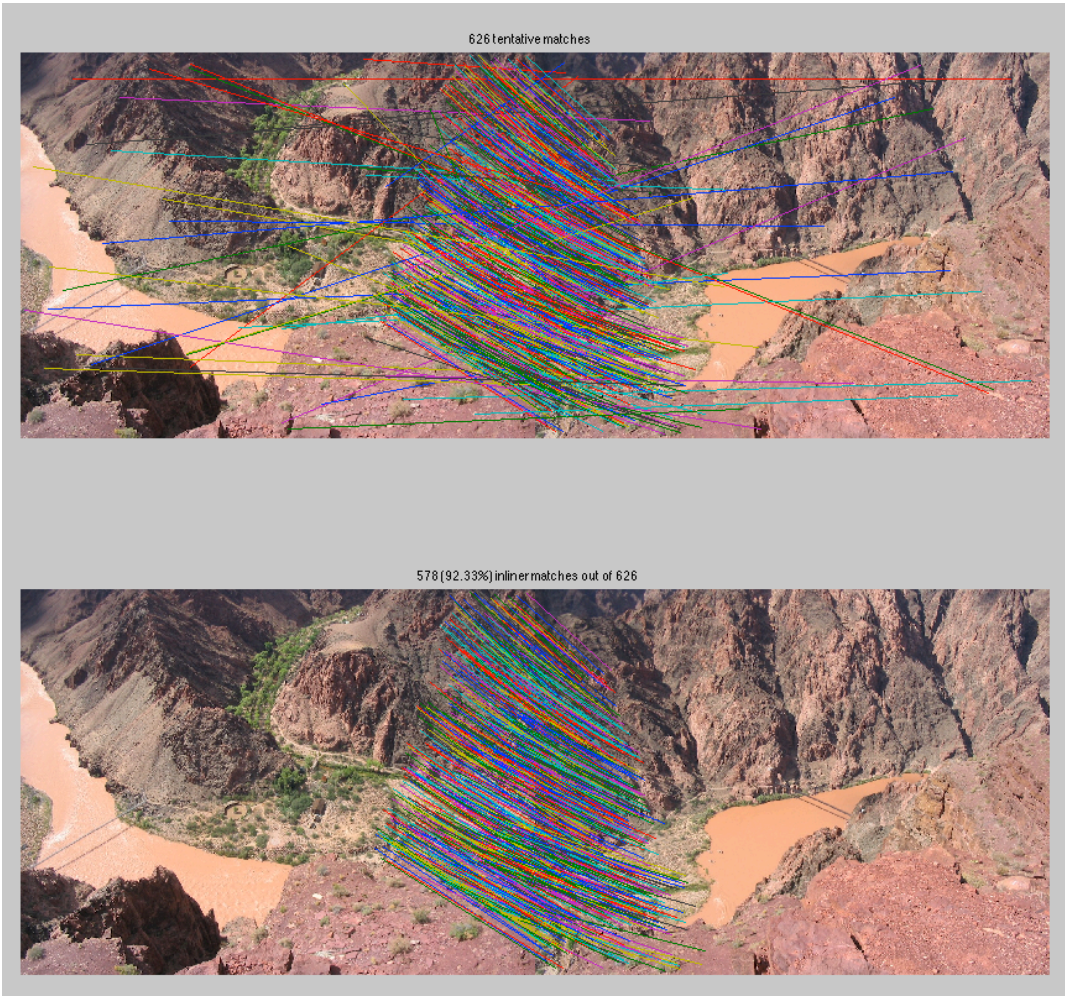
This procedure is repeated a fixed number of times,

Each time producing either a model which is rejected because too few points are part of the consensus set, or a refined model together with a corresponding consensus set size.

In the latter case, we keep the refined model if its consensus set is larger than the previously saved model.

Construct the mosaic through SIFT and Ransac

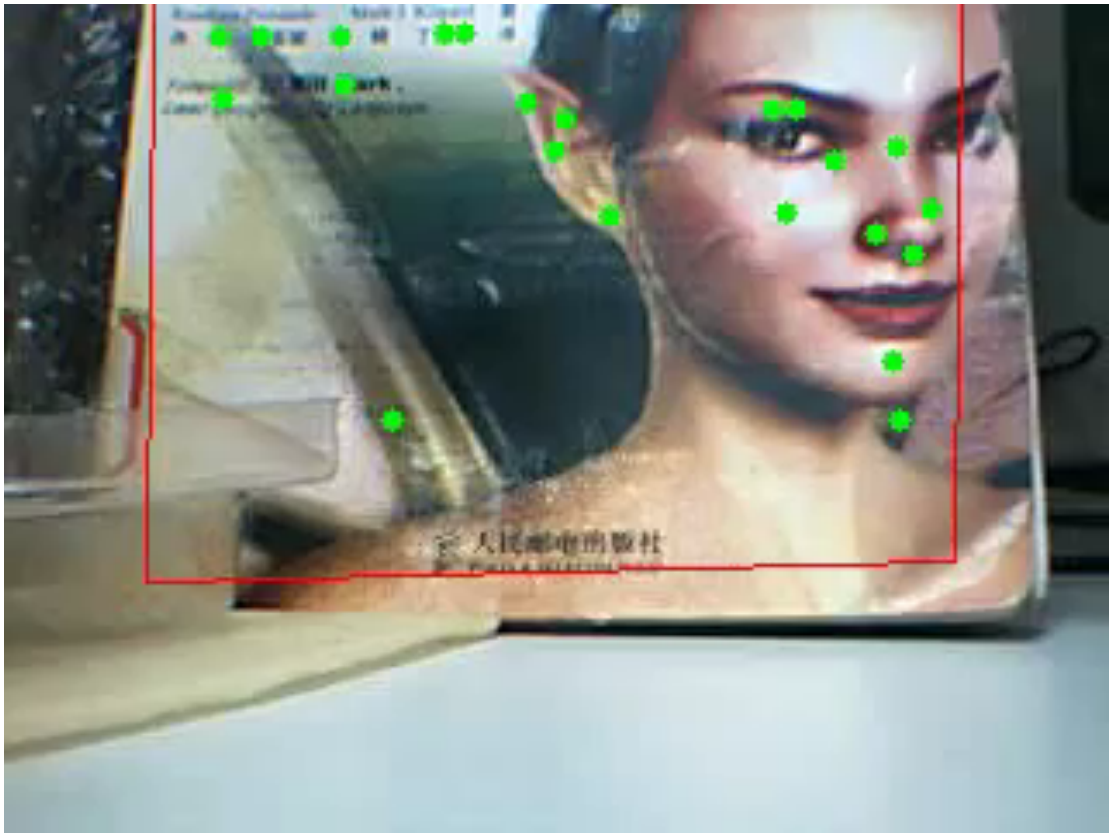
Construct the mosaic through SIFT and Ransac



In VLFeat: [matches, scores]=showMatches(im1,im2);

Construct the mosaic through SIFT and Ransac

Stability

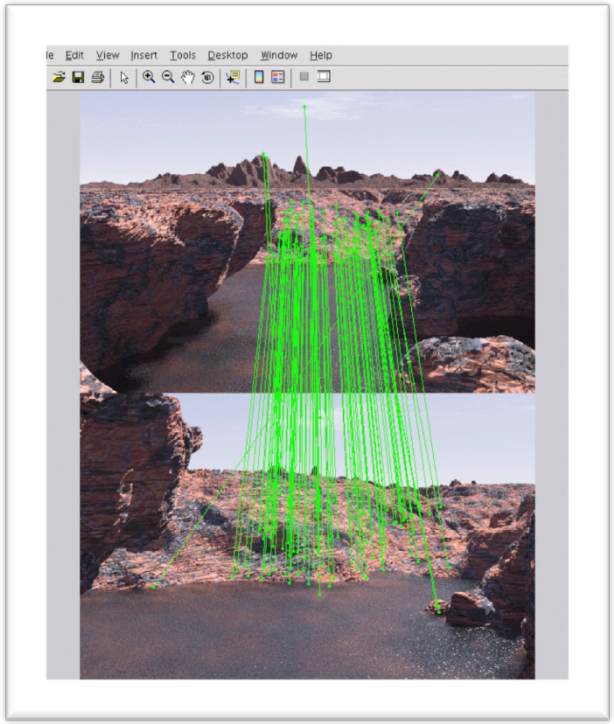


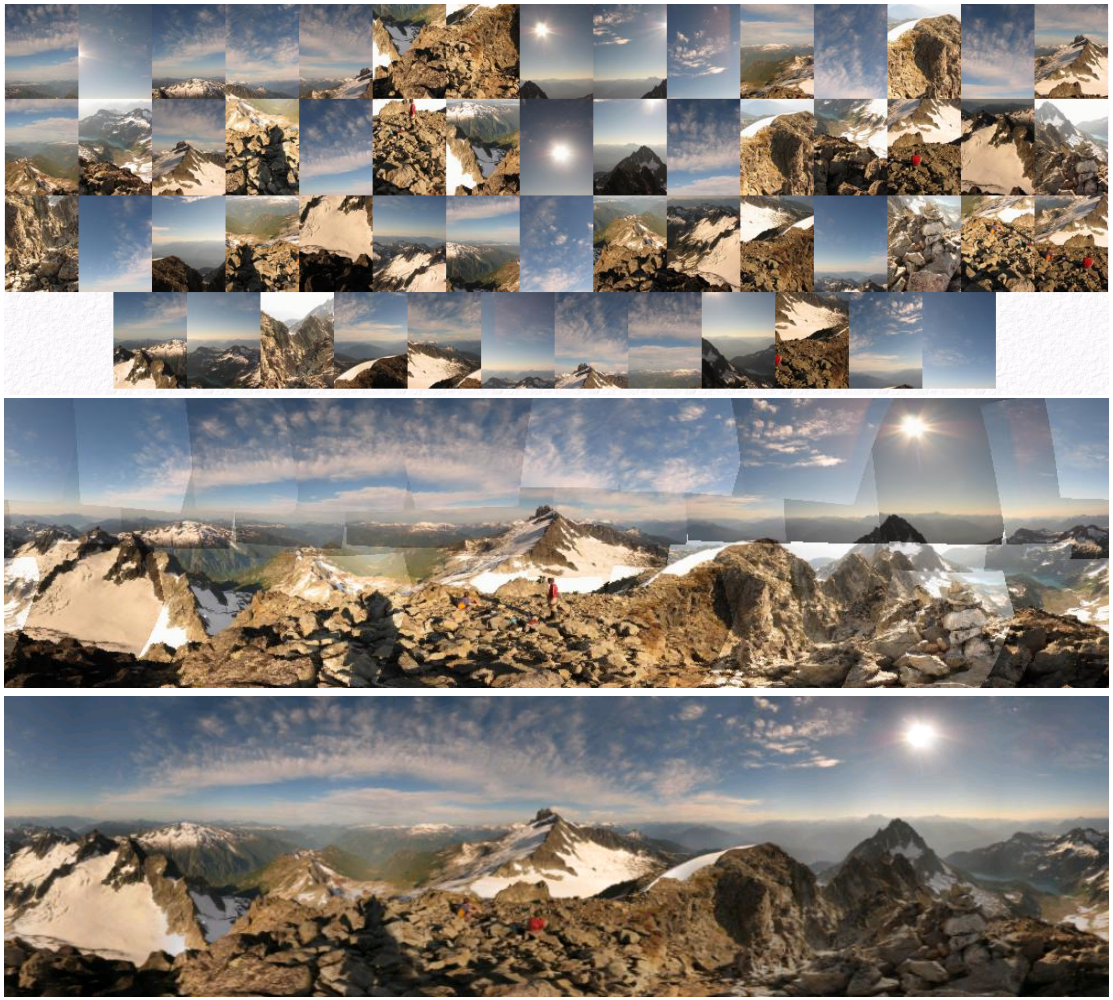
http://www.youtube.com/watch?v=aYd2kAN0Y20
http://www.youtube.com/watch?v=f4qwuPV-j00
http://www.youtube.com/watch?v=qLuwTqbe6wo

Applications of local invariant features SIFT

- Automatic mosaicing
- Stereo images
- Recognition
- Motion tracking
- Mobile robot navigation
- 3D reconstruction
- …

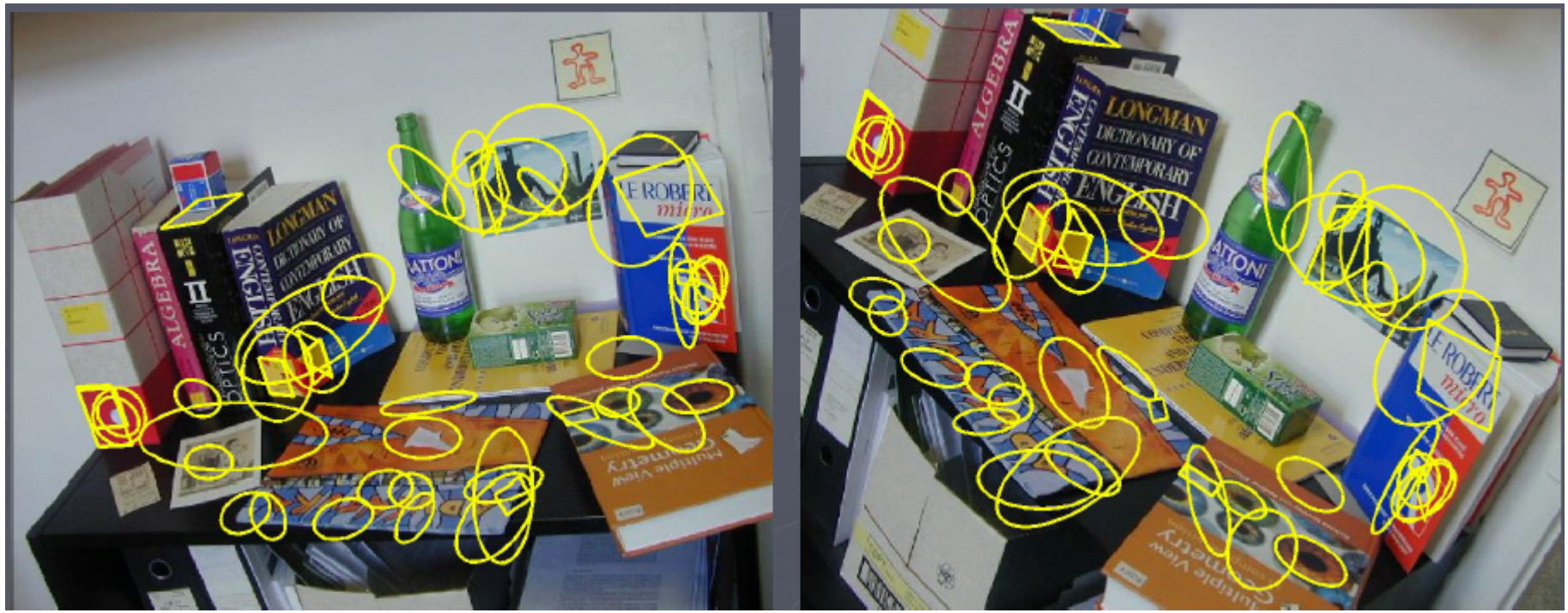
Automatic mosaicing





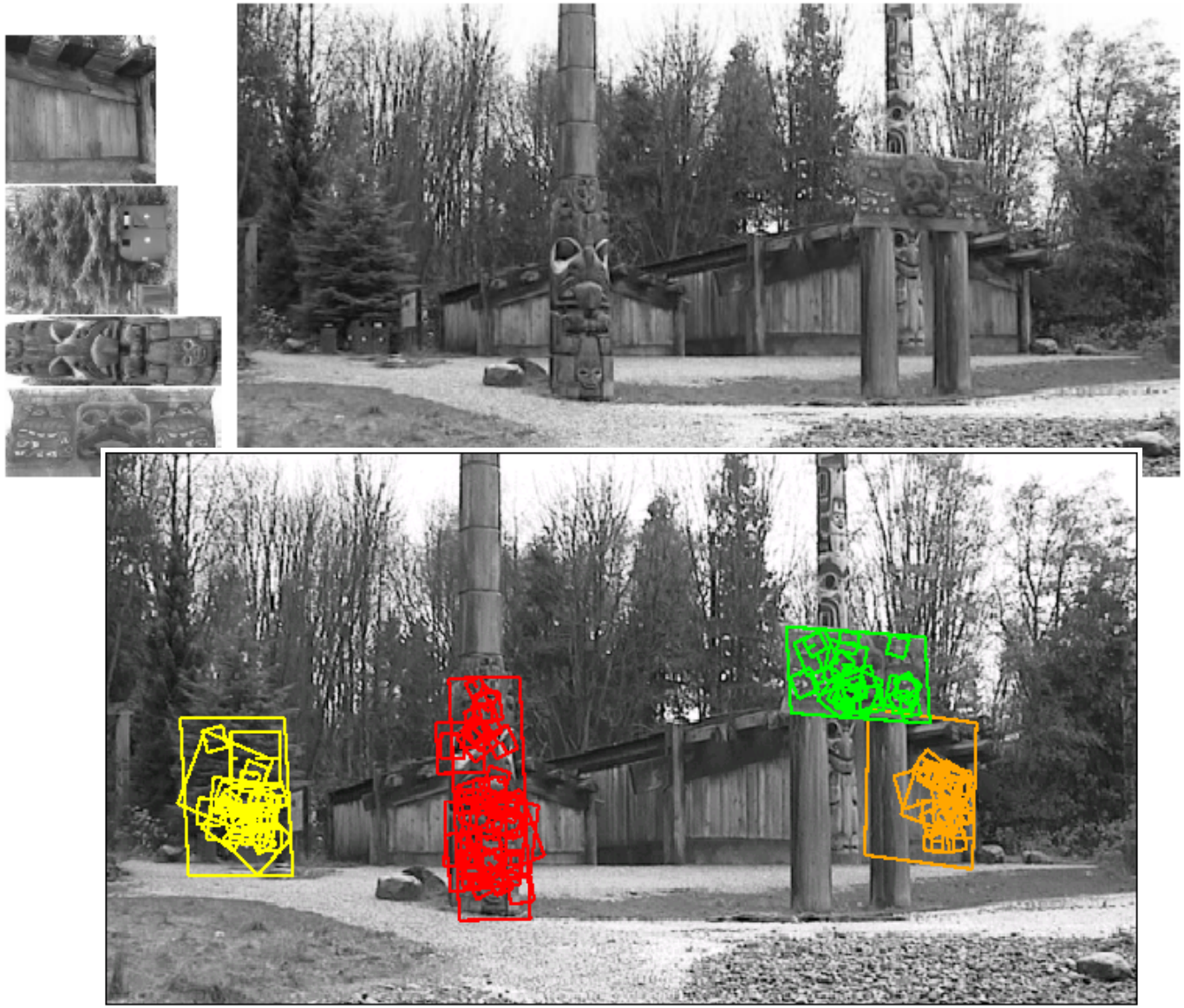
http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

Stereo images



[Image from T. Tuytelaars ECCV 2006 tutorial]

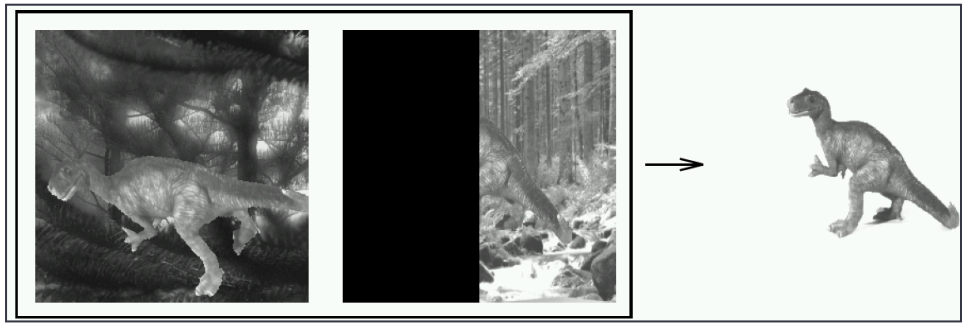
Object deteccion



Recognition of 3D objects

- Small amount of features could give good recognition, more features more feasible recognition.
- Not trivial to recover the transformation between shapes of the objects.

Recognition of specific objects and scenes



Schmid and Mohr 1997

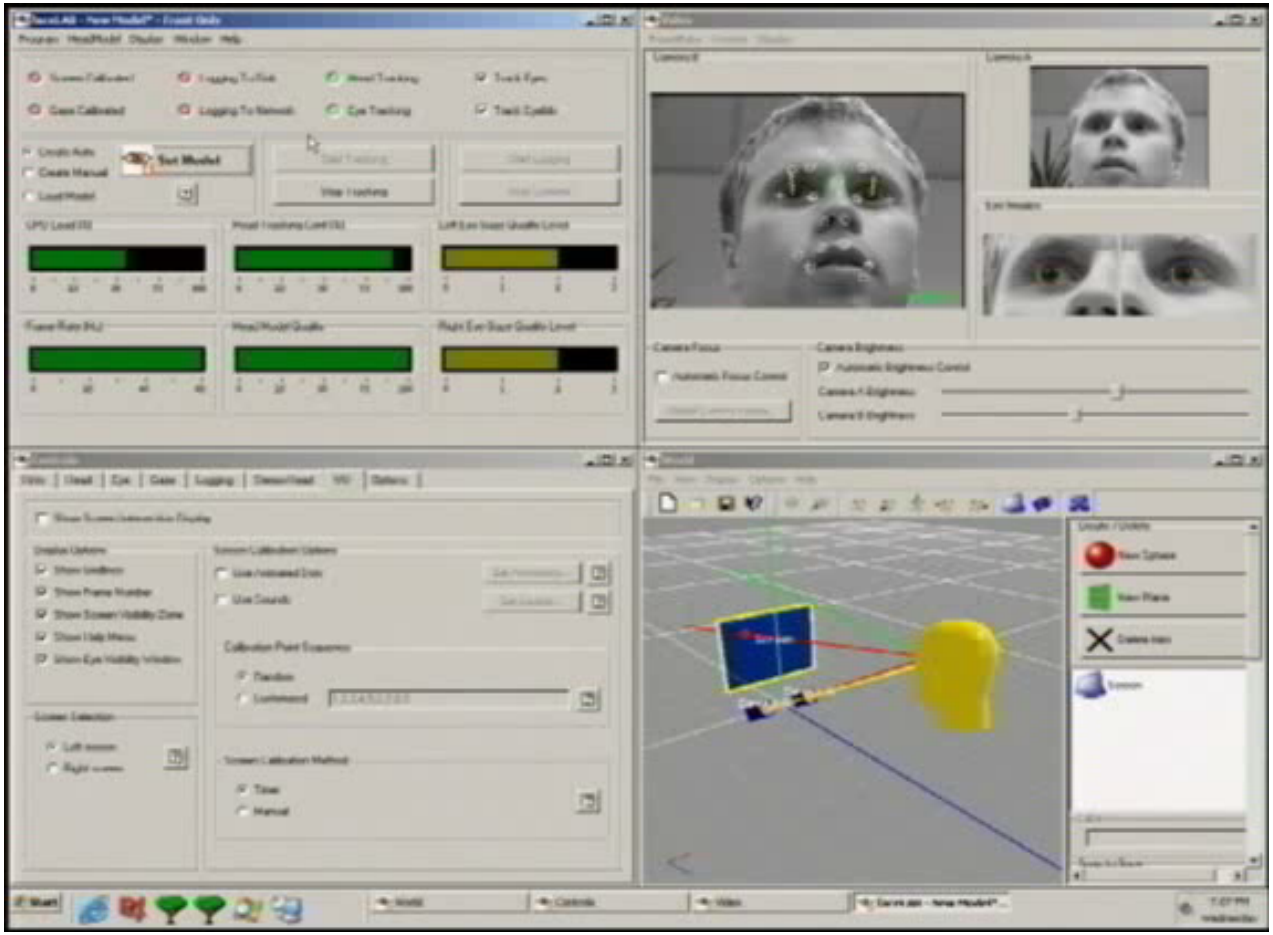
Sivic and Zisserman, 2003

Rothganger et al. 2003

Lowe 2002

Kristen Grauman

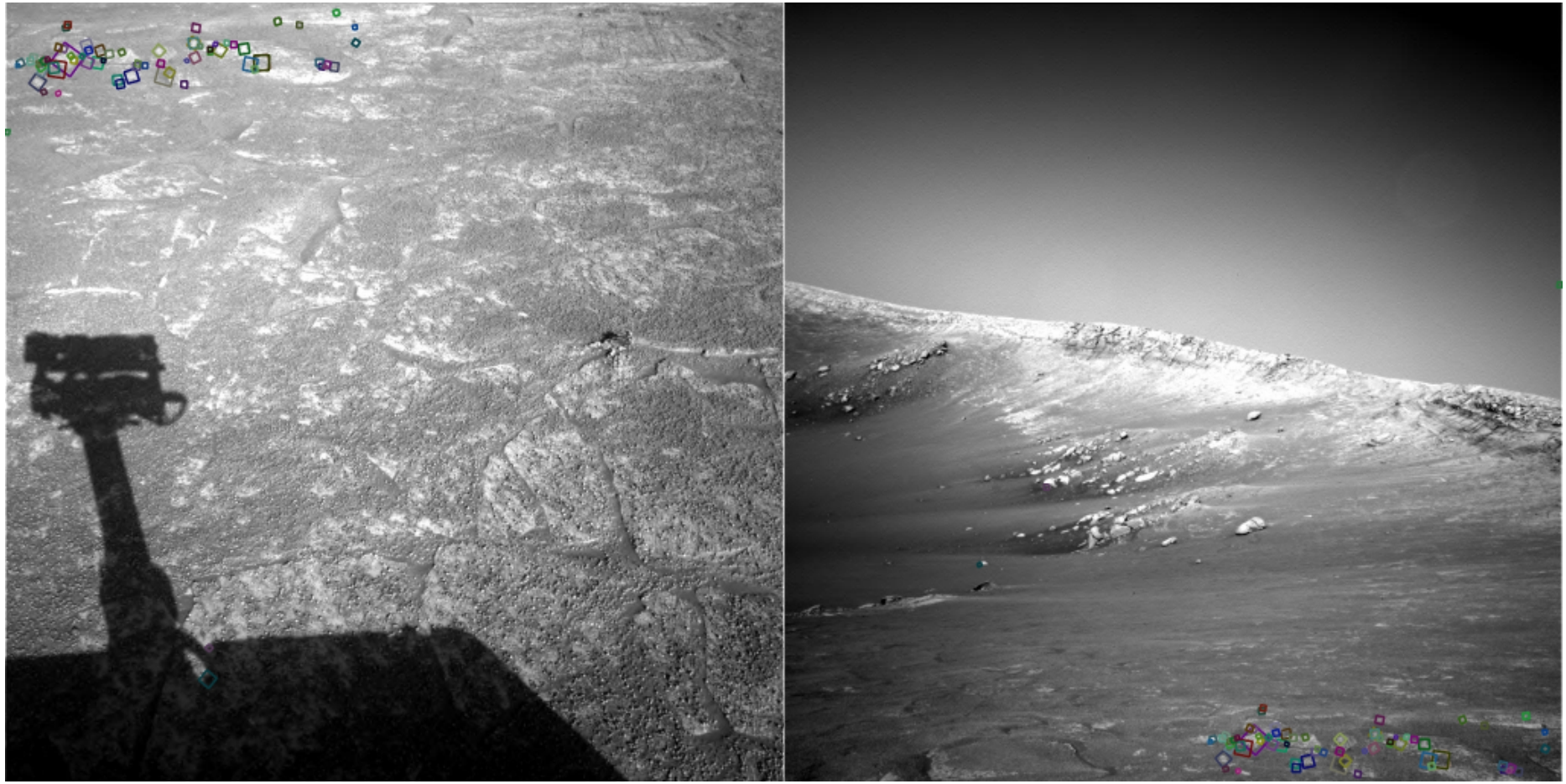
Application FaceLab



http://www.seeingmachines.com/product/faceapi/

Imitations search

Mobile robot navigation

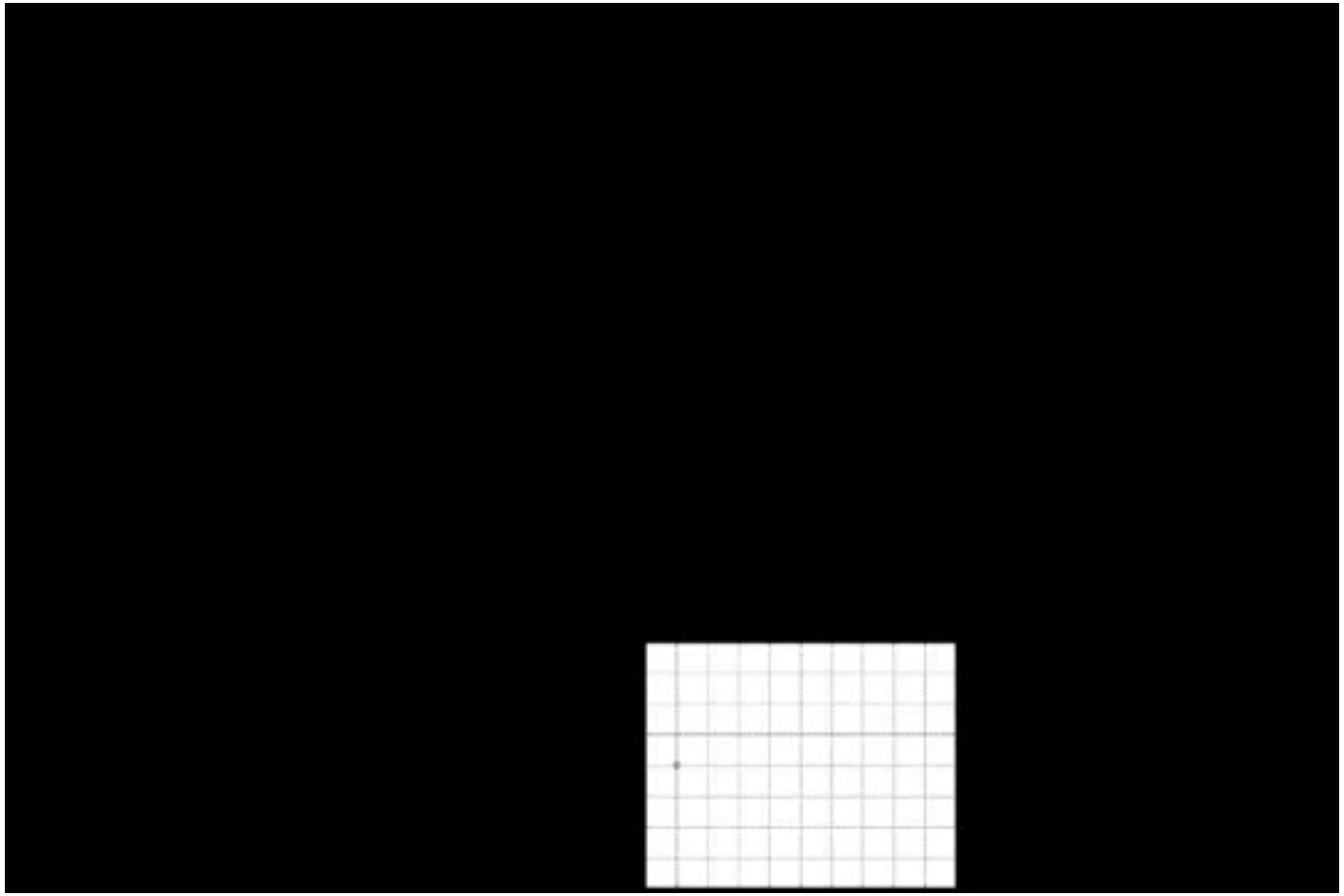


NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

Sony Aibo

- Recognition of charge station
- Communication through visual cards
- Learn to recognize objects

Sift application



http://www.youtube.com/watch?v=aYd2kAN0Y20
http://www.youtube.com/watch?v=f4qwuPV-j00
http://www.youtube.com/watch?v=qLuwTqbe6wo

Sift application

http://www.youtube.com/watch?v=aYd2kAN0Y20
http://www.youtube.com/watch?v=f4qwuPV-j00
http://www.youtube.com/watch?v=qLuwTqbe6wo

SIFT descriptor [Lowe 2004]

- Robust matching technique
  - Fast and efficient—can run in real time
  - Can handle changes in rotation and viewpoint (Up to about 60 degree out of plane rotation)
  - Can handle significant changes in illumination (sometimes even day vs. night)
  - Based on histograms of Gradients in patches for robustness to small shifts and translations
- Partially invariant to occlusion, clutter
  - Lots of code available
    - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT

Steve Seitz