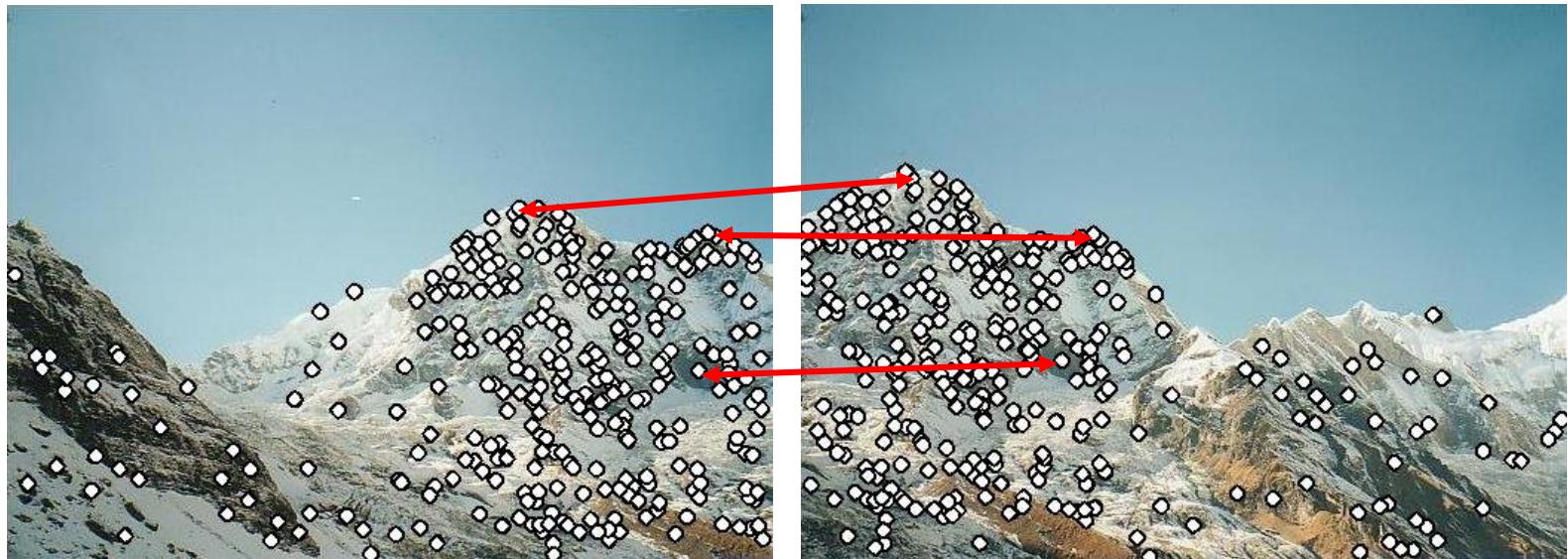


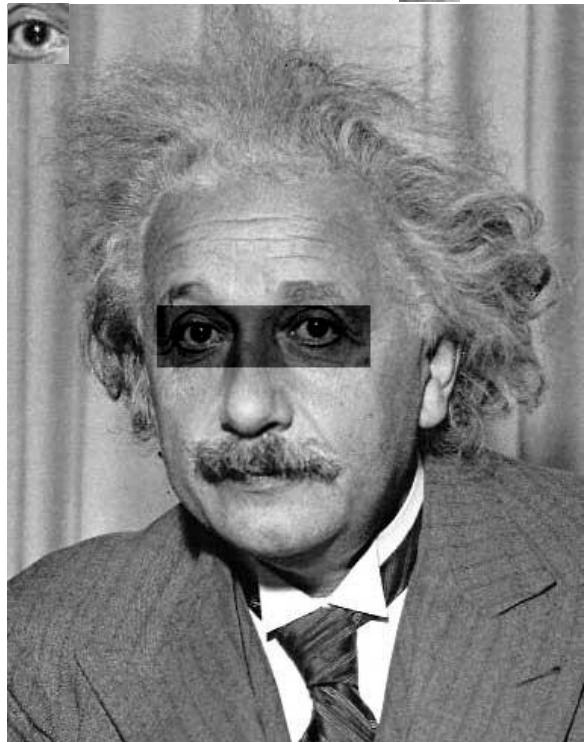
FEATURE EXTRACTION (SIFT)

Class 6: Artificial Vision



Recall: Template detection

- Goal: find  in an image



Input

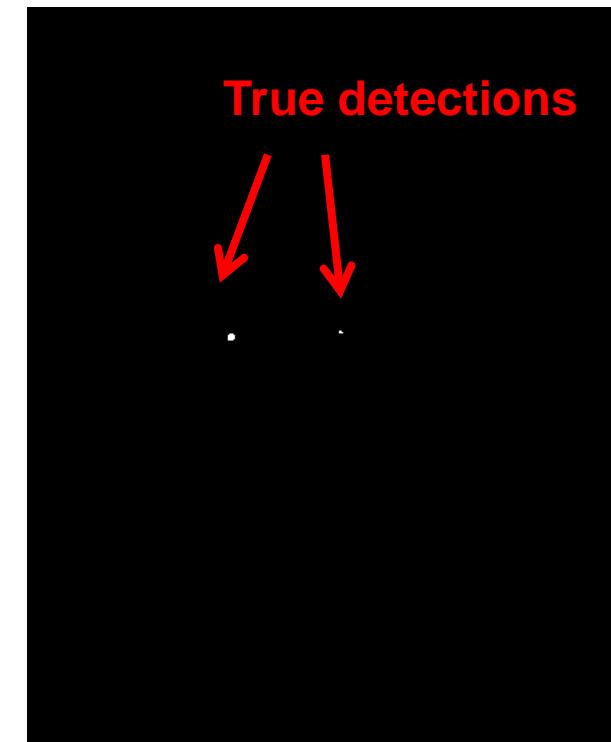


Normalized X-Correlation Detection: Thresholded Image

Distance/similarity: $d(\text{template}, \text{image}) = \text{Normalized X-correlation}$

Detection = $\max(d(\text{template}, \text{image}))$

Disadvantages? Alternatives?



True detections

Problem	Technic/Tool/Method	Application

Template matching

Eye tracking

Normalized cross-correlation

HOG

Image retrieval

Pedestrian detection

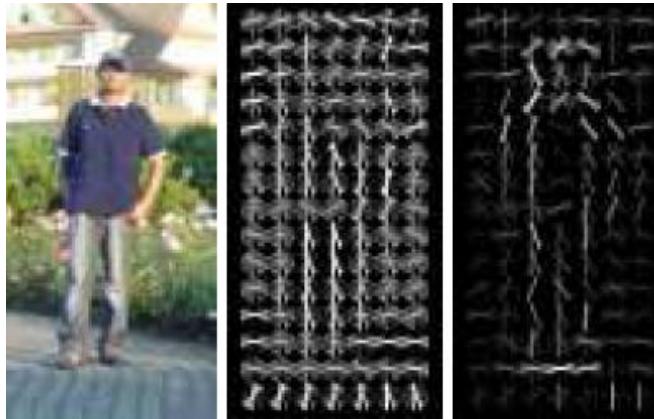
Object detection

Natural scene classification

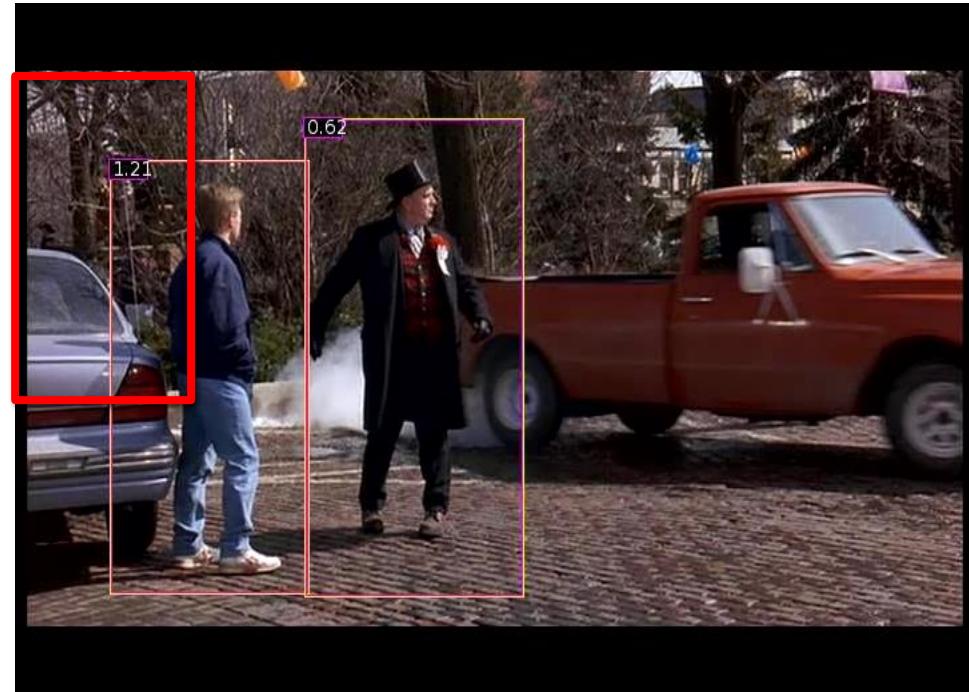
Feature extraction

K-Nearest neighbour

Recall: Detection based on HOG descriptors



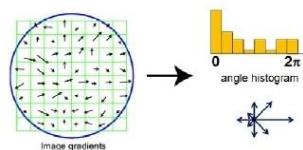
HOG – Histogram of Gradients



Distance/similarity: $d(\text{template}, \text{image}) = d(\text{HOG}(\text{template}), \text{HOG}(\text{region}))$

Detection = $\max(d(\text{template}, \text{image}))$

Recall: Retrieval based on HOG descriptors



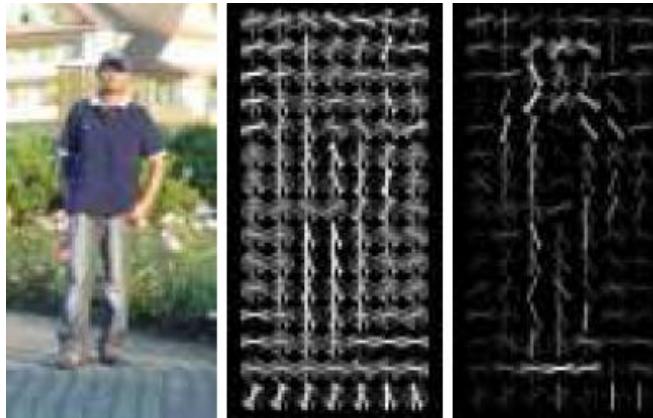
HOG



Distance/similarity: $d(\text{query}, \text{image}) = d(\text{mean}(\text{HOG}(\text{query})), \text{mean}(\text{HOG}(\text{image})))$

Retrieval = $d(\text{query}, \text{image})$

Recall: Detection based on HOG descriptors



HOG – Histogram of Gradients



Distance/similarity: $d(\text{template}, \text{image}) = d(\text{HOG}(\text{template}), \text{HOG}(\text{region}))$

Detection = $\max(d(\text{template}, \text{image}))$

Disadvantages?

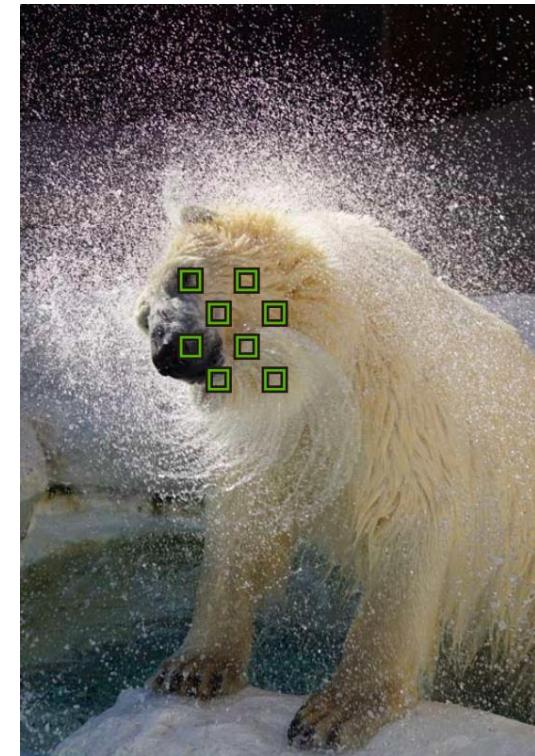
- Are all pixels equally important?
- Is the detection invariant to scale?
- Is the detection invariant to rotation?

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



Did you think...?

- What are the green points in the images taken by your camera?



Informative vs. non-informative regions for object recognition



How do we use the informative regions?



[Diva Sian](#)



[swashford](#)

How do we use the informative regions?



Diva Sian



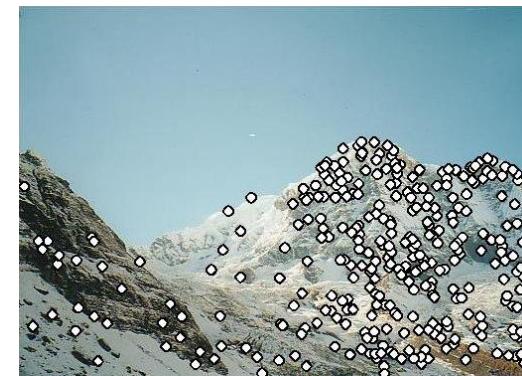
scgbt

Today

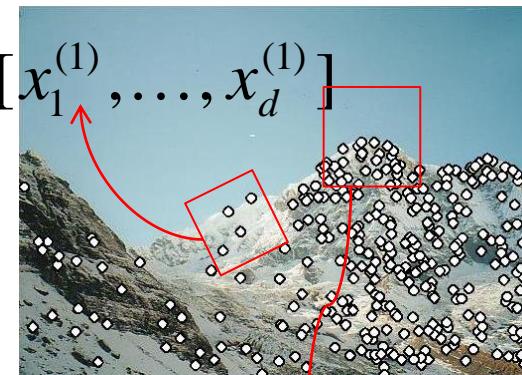
- Local invariant features
 - Detection of interest points
 - Harris corner detection
 - Scale invariance
 - Censure
 - Description of local patches
 - SIFT : Histograms of oriented gradients
 - ORB feature descriptor
 - Matching

Local features: main components

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

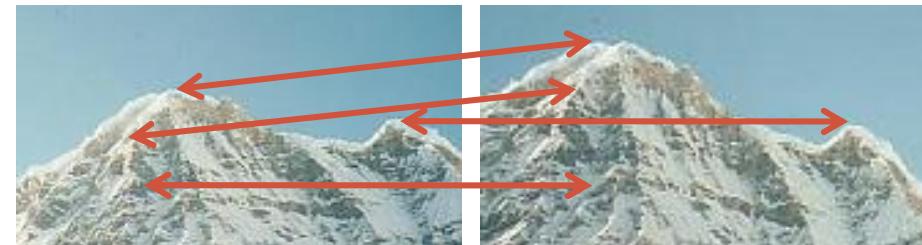


- 2) **Description:** Extract vector feature descriptor surrounding $\mathbf{x}_1 = [x_1^{(1)}, \dots, x_d^{(1)}]$ each interest point.



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

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



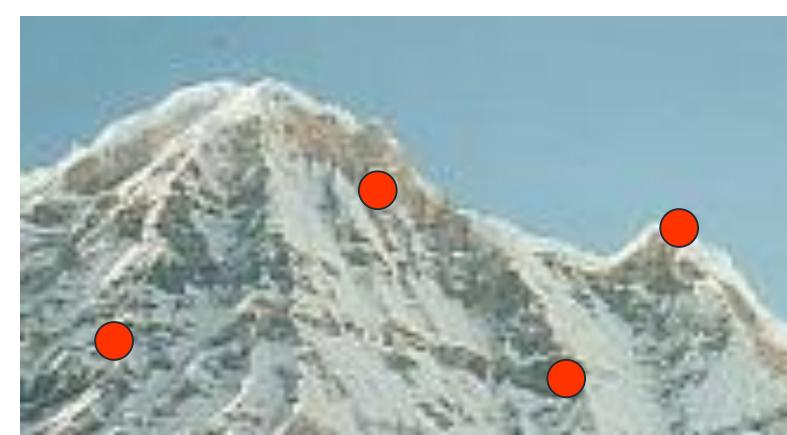
Local features: desired properties

- **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.

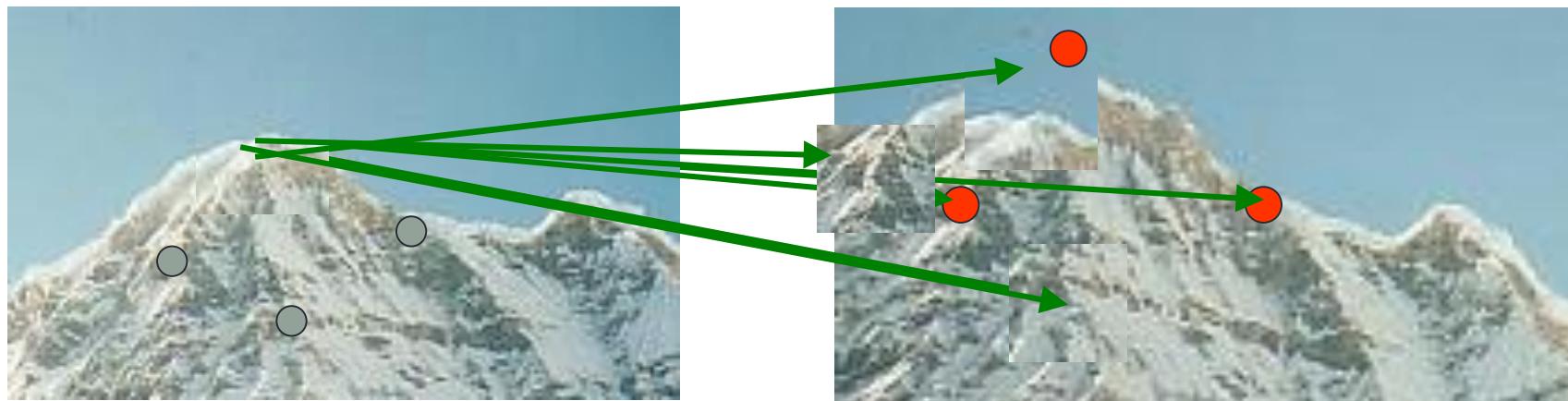


No chance to find true matches!

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

Goal: descriptor distinctiveness

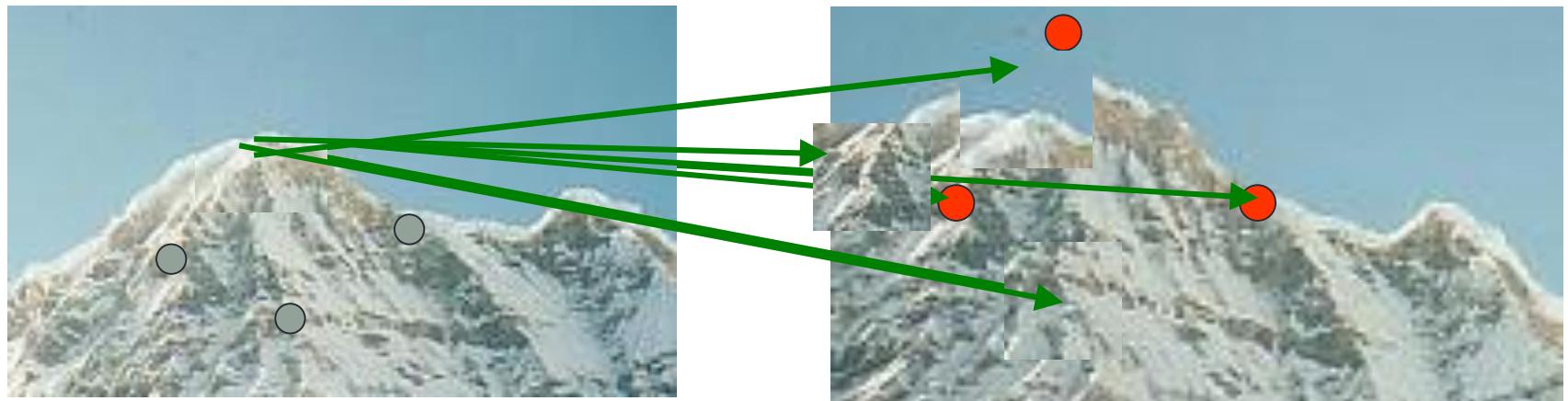
- We want to be able to reliably determine which point goes with which.



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

Goal: compactness, efficiency and locality

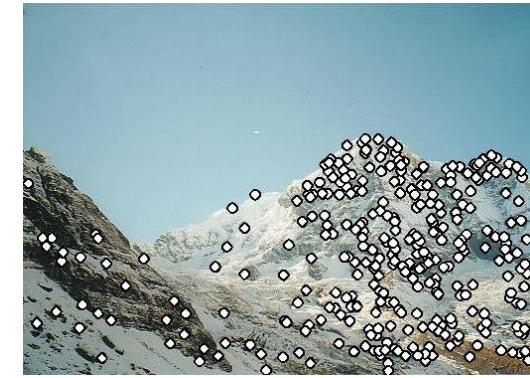
- We want to extract the minimum information that allows us to put into correspondence.



- Which should it be?

Local features: main components

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

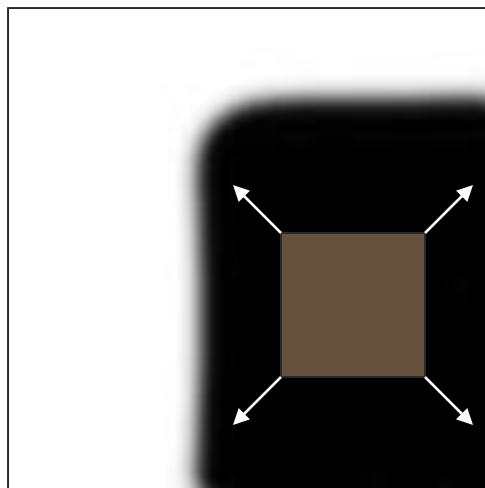


- 2) **Description:** Extract vector feature descriptor surrounding each interest point.

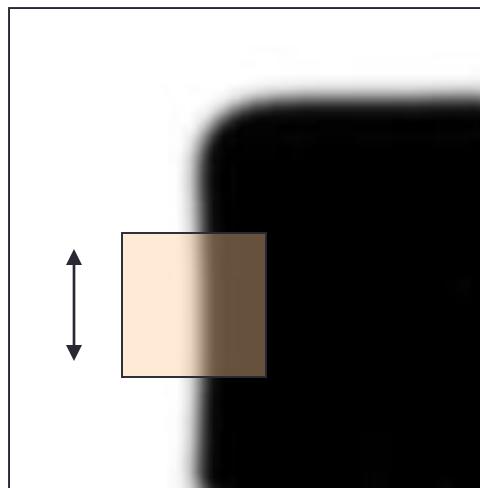
- 3) **Matching:** Determine correspondence between descriptors in two views

Corners as distinctive interest points

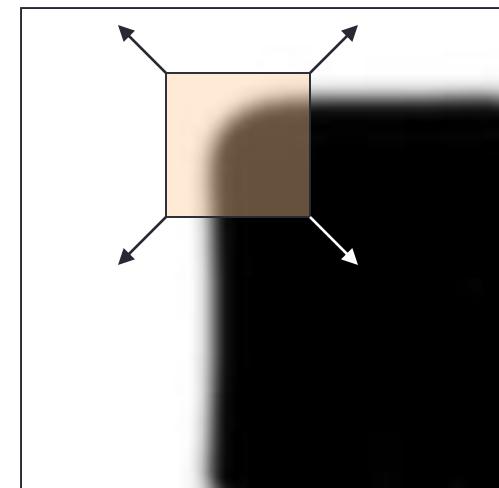
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



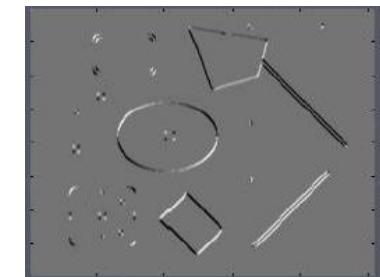
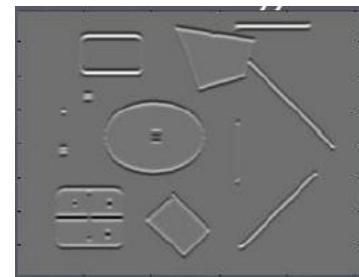
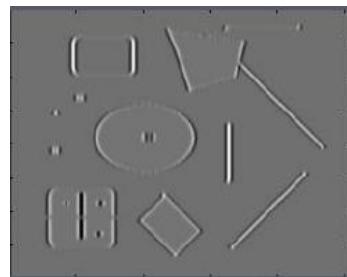
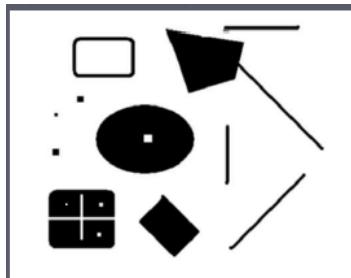
“flat” region:
no change in all
directions



“edge”:
no change along the edge
direction



“corner”:
significant change in all
directions



Notation:

$$I_x \Leftrightarrow \frac{\partial I}{\partial x}$$

$$I_y \Leftrightarrow \frac{\partial I}{\partial y}$$

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

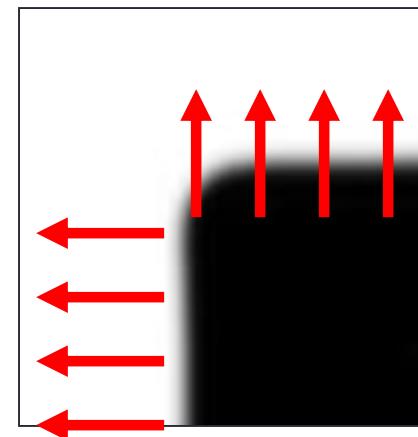
Definition: Corners are distinctive image interest points

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

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 pixel 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?

A little bit of Math:

Given a symmetric matrix M , it can be decomposed of:

$$M = E \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_2 \end{pmatrix} E^T \quad Me_i = \lambda_i e_i, E = [e_1, e_2]$$

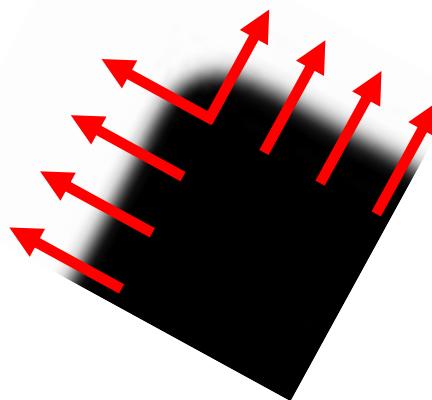
λ_1 and λ_2 are called **eigenvalues**.

Vectors e_1 and e_2 are called **eigenvectors**.

Since E is orthogonal, $E^{-1}=E^T$.

Corners - distinctive interest points

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



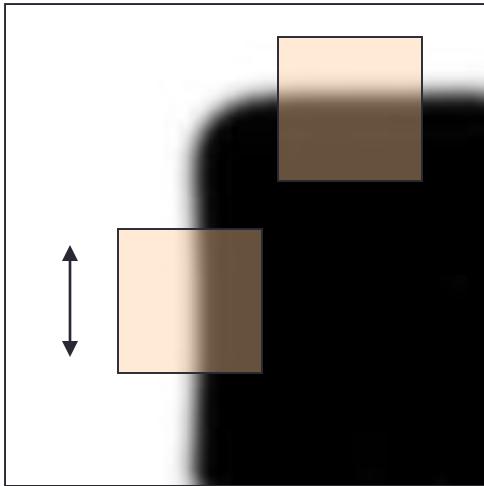
$$Me_i = \lambda_i e_i, E = [e_1, e_2]$$

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.

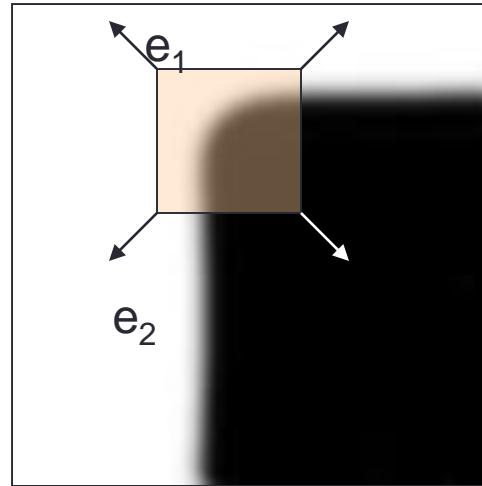
$E \rightarrow$ rotational matrix

Corners as distinctive interest points



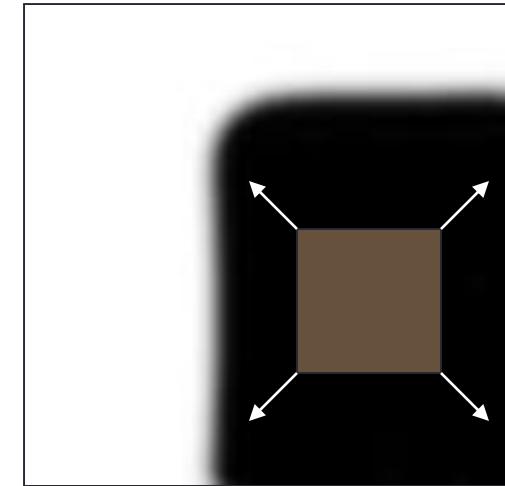
“edge”:

$$\begin{aligned} |\lambda_1| &>> |\lambda_2|, \text{ or} \\ |\lambda_2| &>> |\lambda_1| \end{aligned}$$



“corner”:

$$\begin{aligned} |\lambda_1| \text{ and } |\lambda_2| &\text{ are large,} \\ \lambda_1 &\sim \lambda_2; \end{aligned}$$



“flat” region

$$|\lambda_1| \text{ and } |\lambda_2| \text{ are small;}$$

Harris corners: ‘cornerness’ score:

Why shall we normalize?

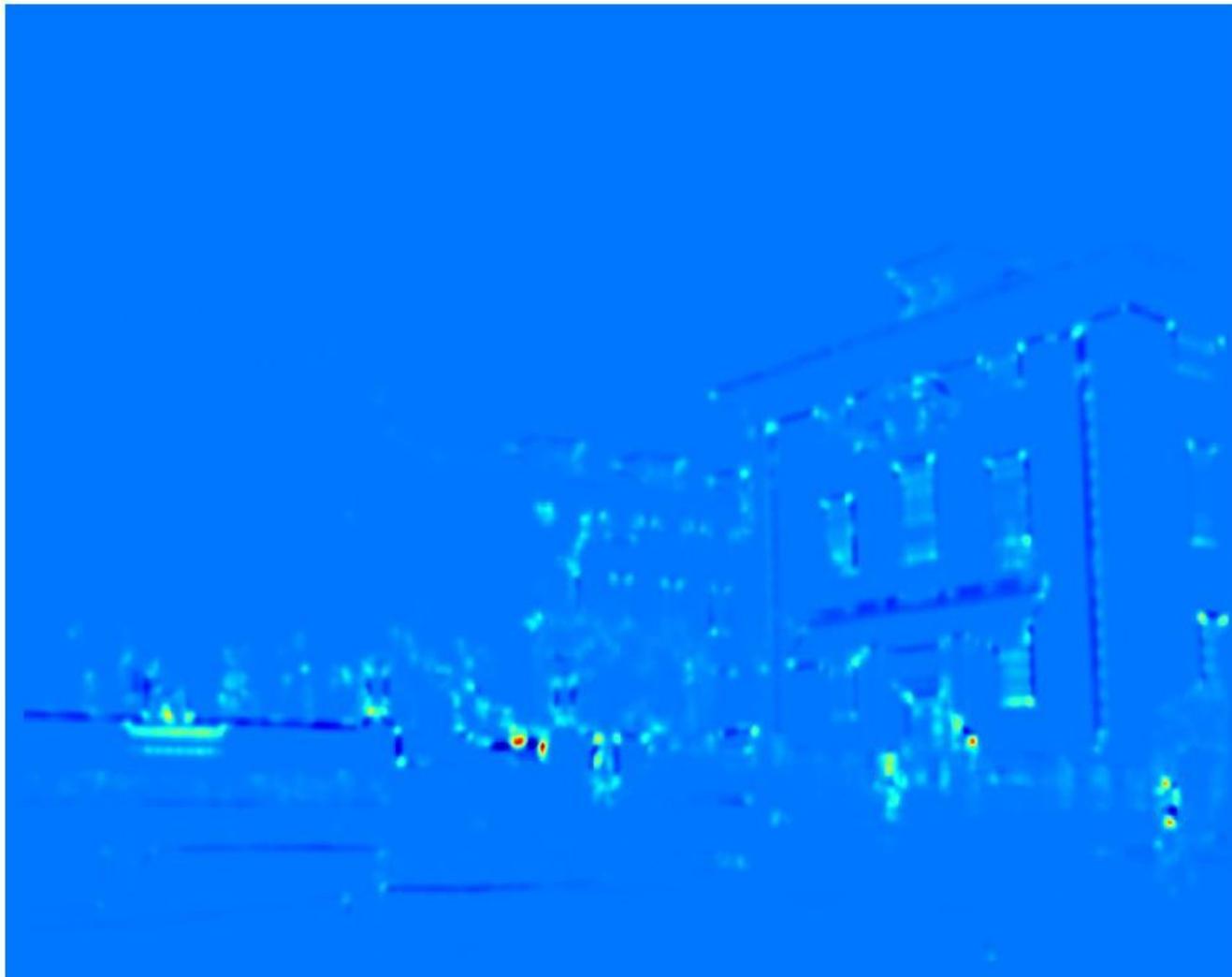
$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$



- What points would you choose?

Example of Harris application

- Compute corner response at every pixel.



Example of Harris application



Harris corner detector

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

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

2. Compute **eigenvectors and eigenvalues**

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

3. Find **points with large corner response ($f >$ threshold)**.

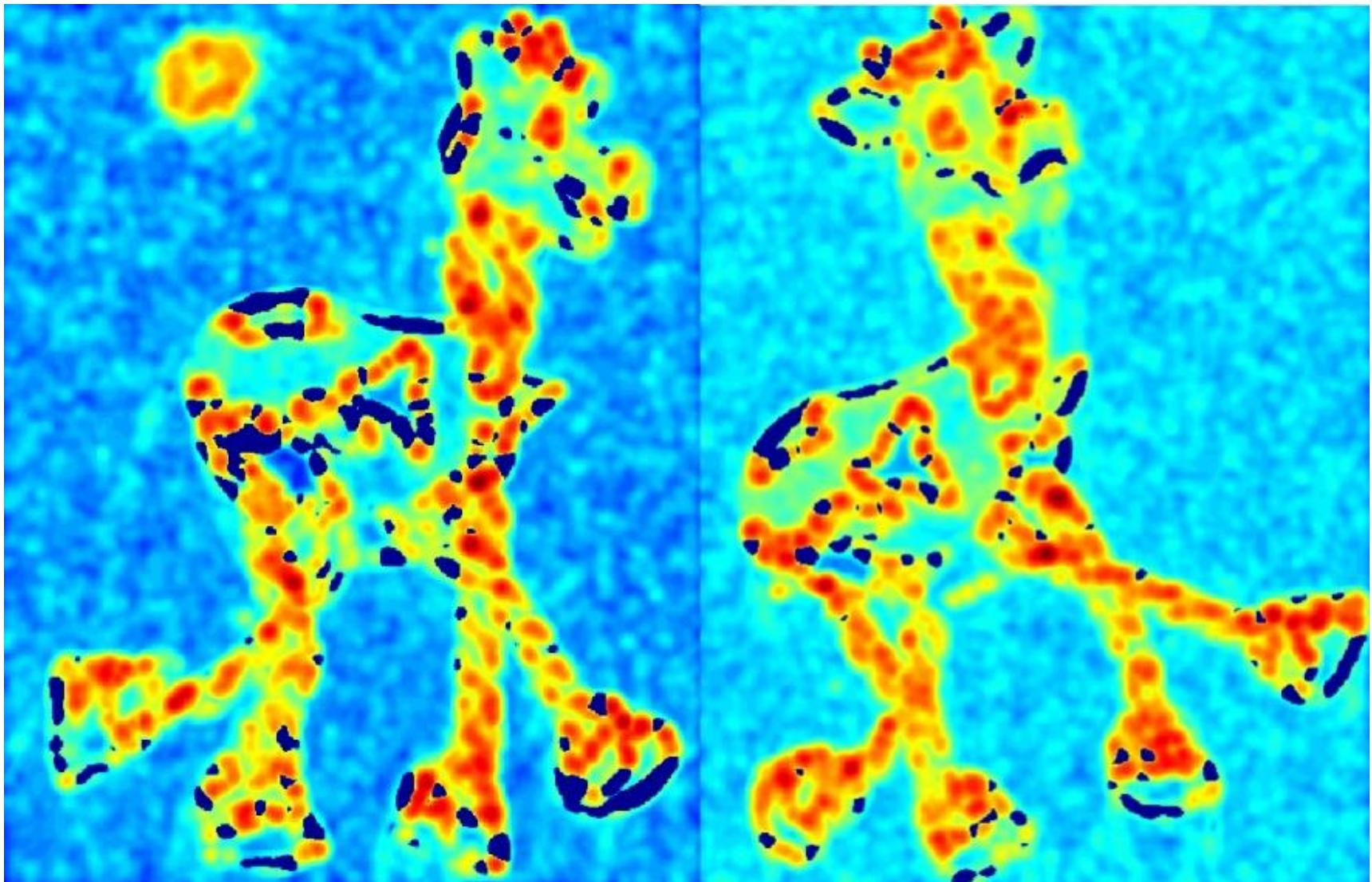
$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

4. Take the points of **local maxima for each window**, i.e. perform non-maximum suppression in the neighborhood.

Harris Detector: Steps

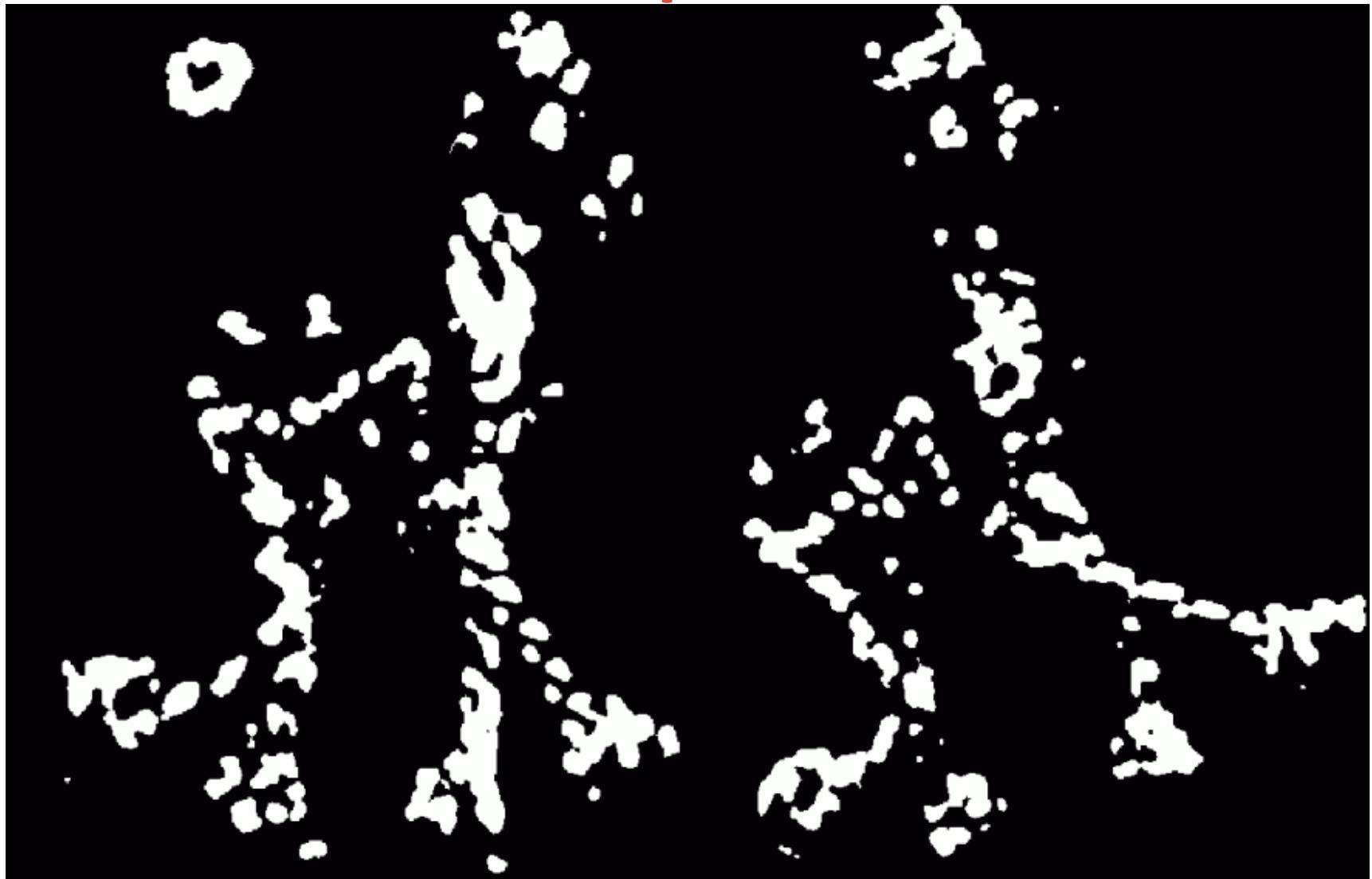


Harris Detector: Cornesness evaluation



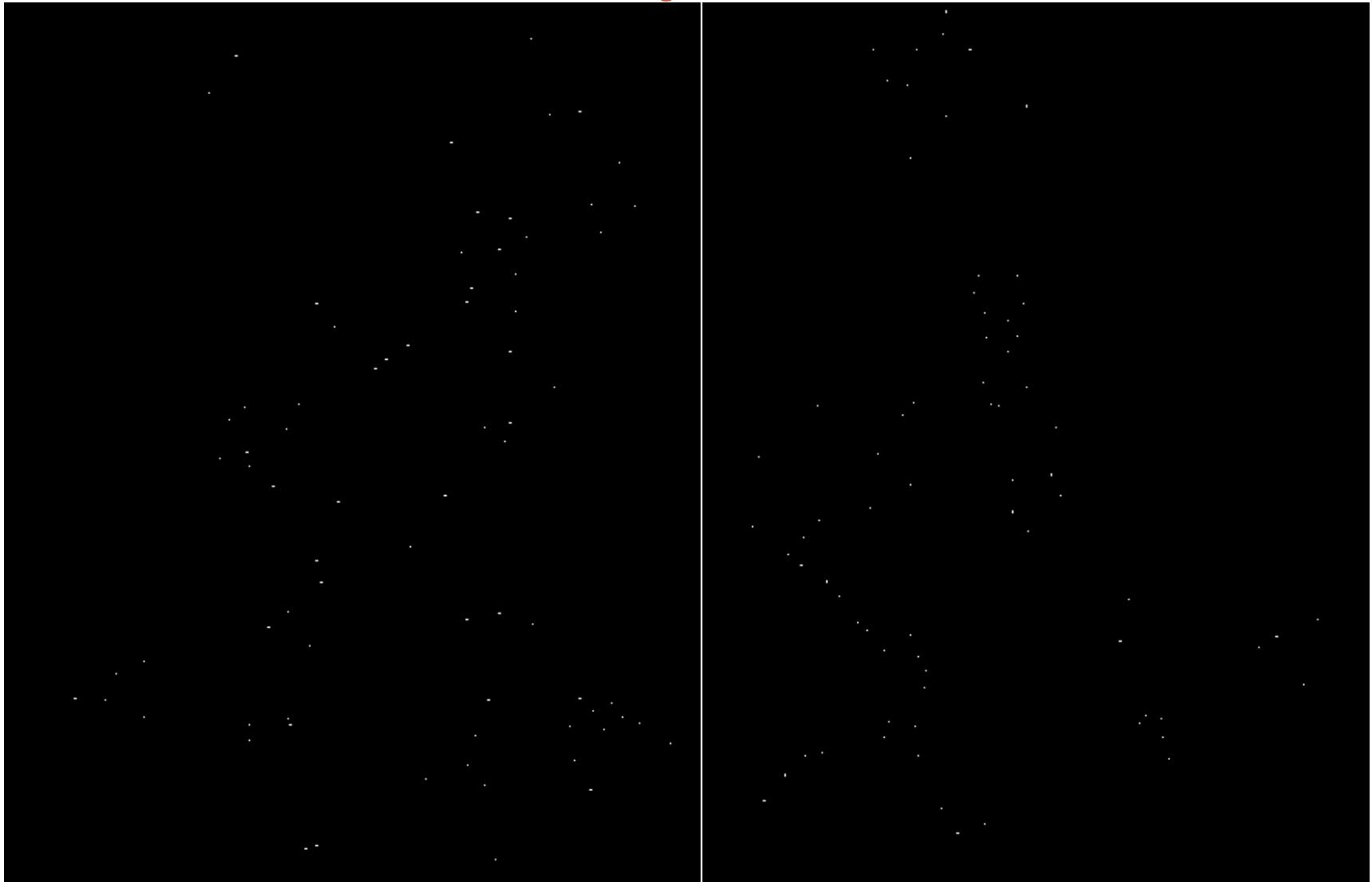
Compute corner response f

Harris Detector: Steps



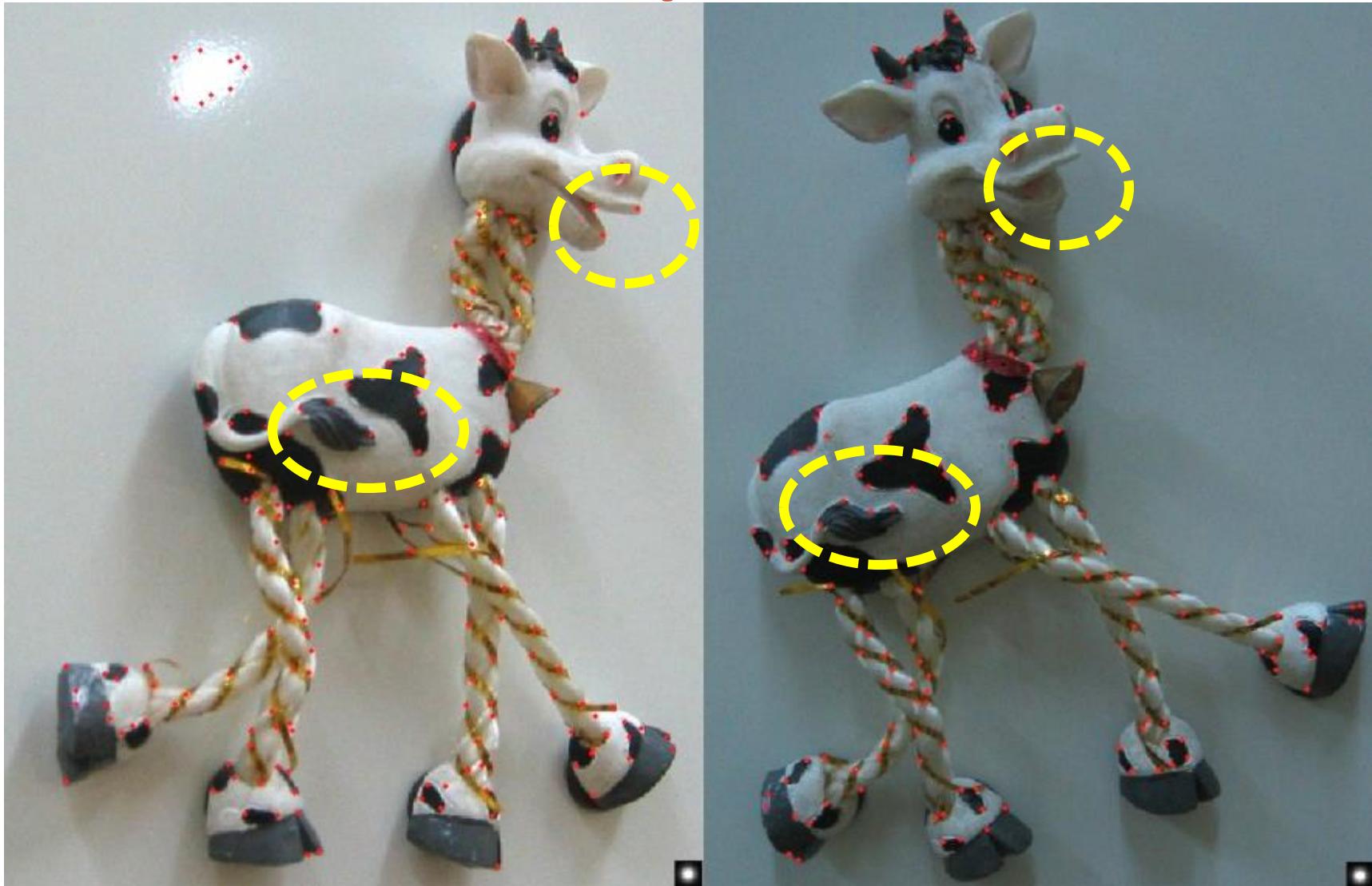
Find points with large corner response: $f > \text{threshold}$

Harris Detector: Steps



Non-maxima suppression: Take only the points of local maxima of f in each window

Harris Detector: Steps



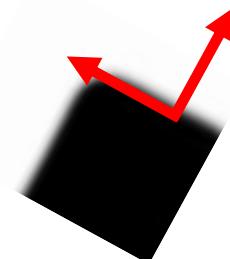
Exercise: Explain the Harris invariance



Properties of the Harris corner detector

- Rotation invariant?

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



Yes, X rotates the matrix so that the maximal change is in the direction of e_1 (if $|\lambda_1| > |\lambda_2|$) or e_2 , viceversa.

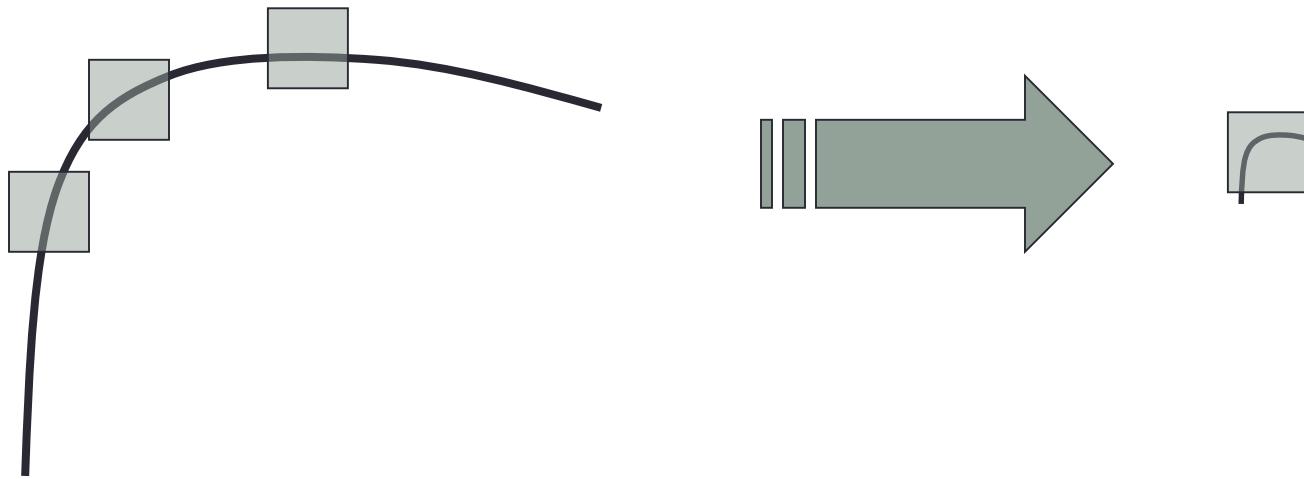
- Scale invariant?



Properties of the Harris corner detector

- Rotation invariant? Yes

- Scale invariant? No



All points will be
classified as **edges**

Corner !

Scale invariant interest points

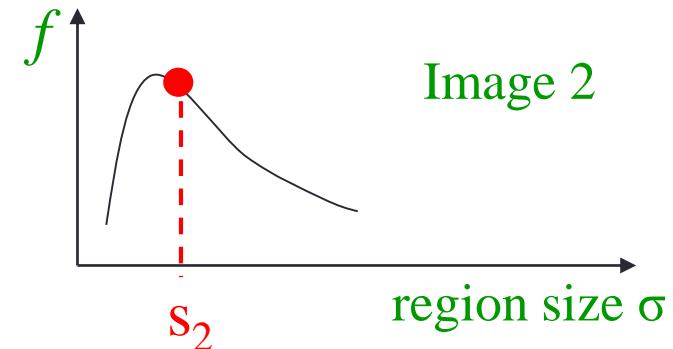
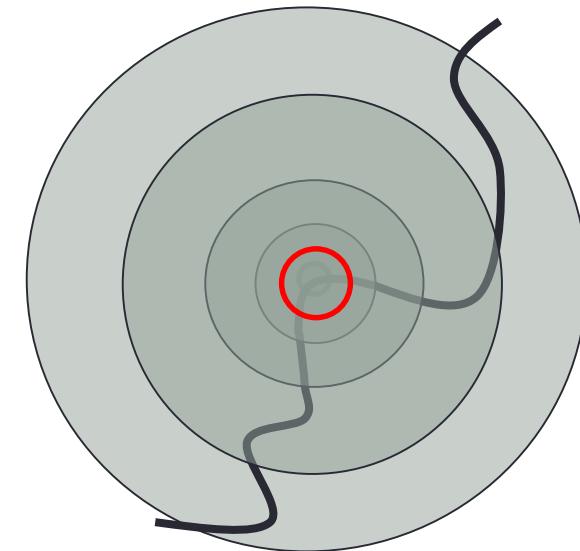
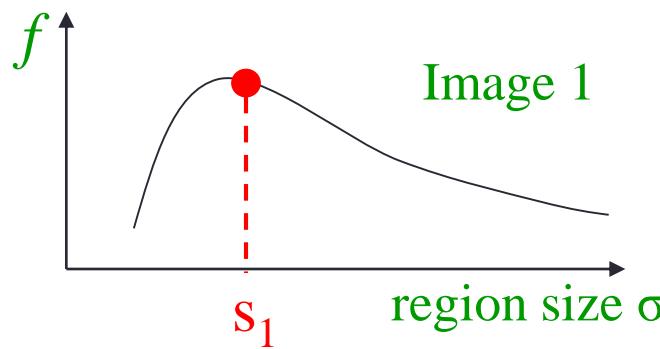
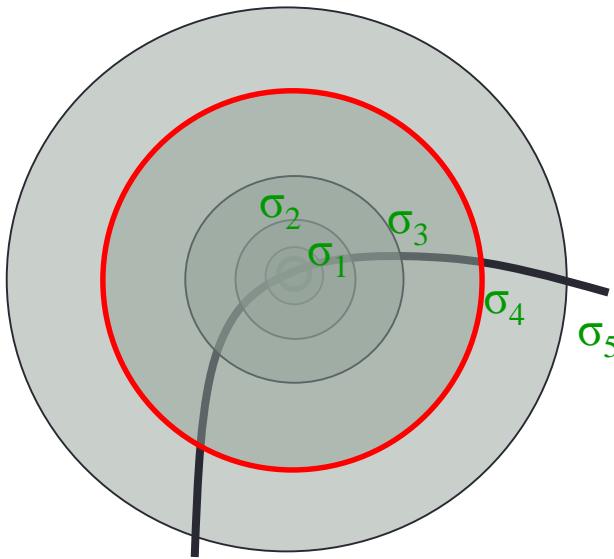
How can we independently select interest points in each image, such that the detections are repeatable across different scales?



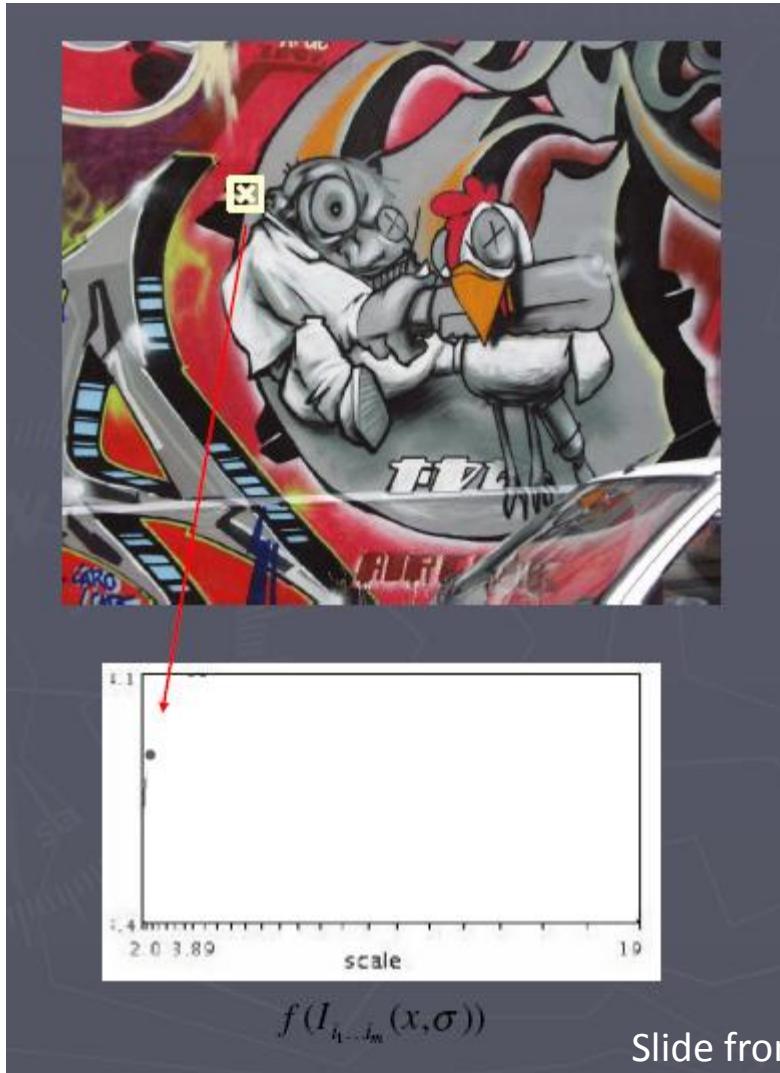
Automatic scale selection

Intuition:

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



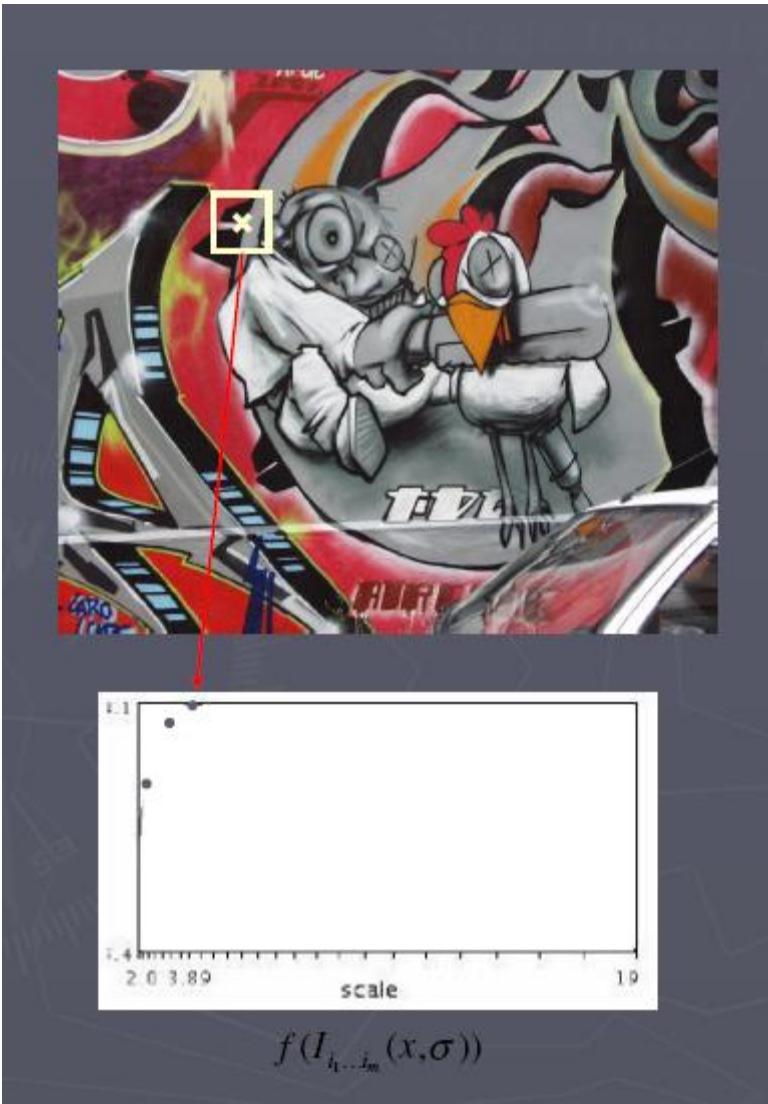
Automatic scale detection



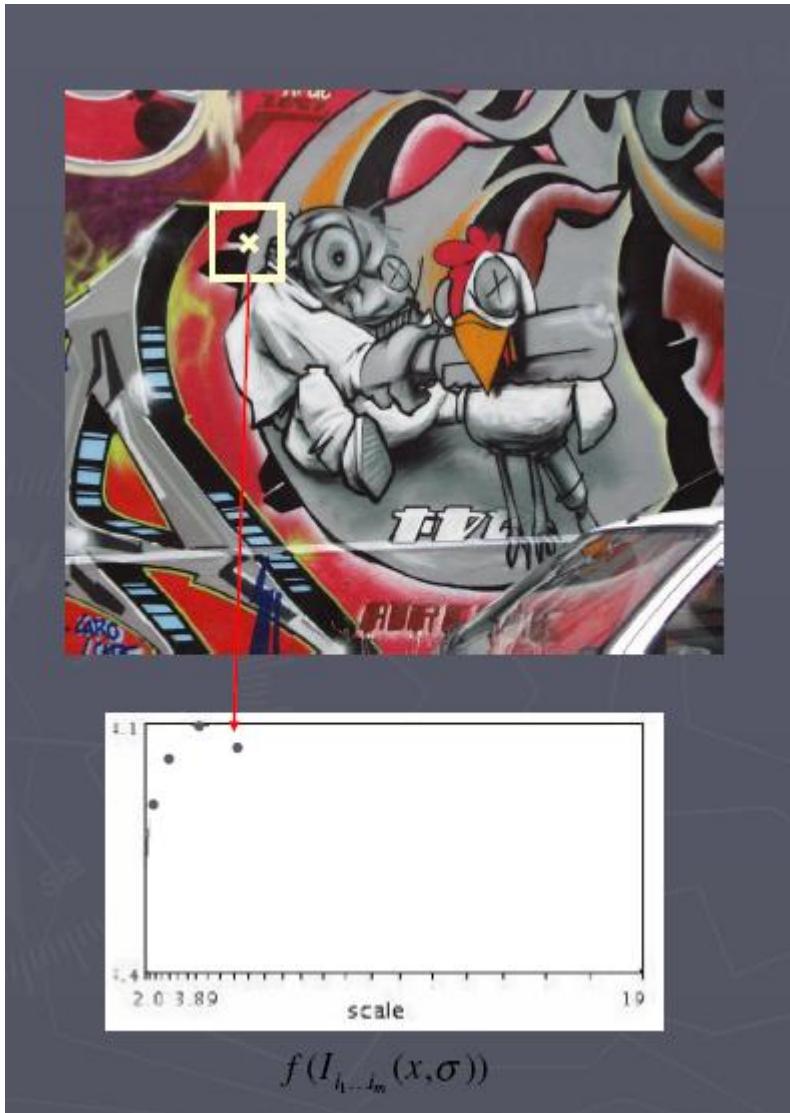
Automatic scale detection



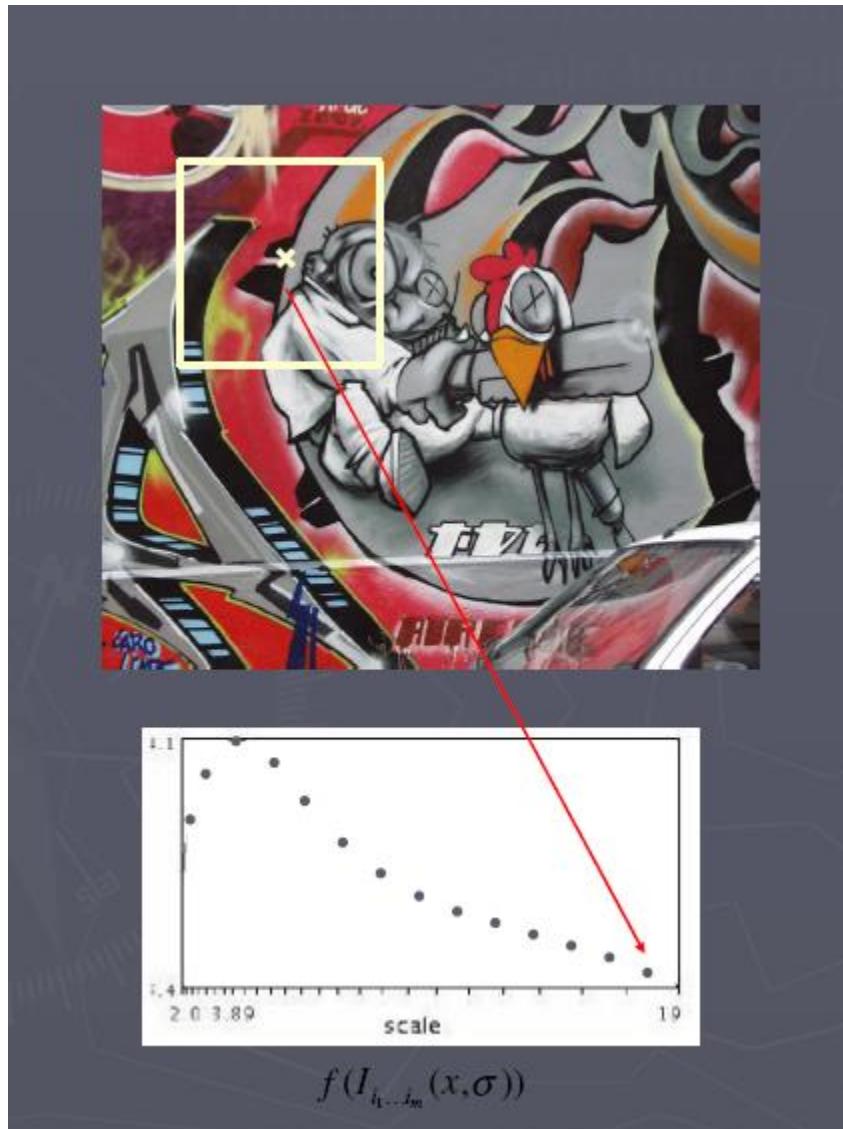
Automatic scale detection



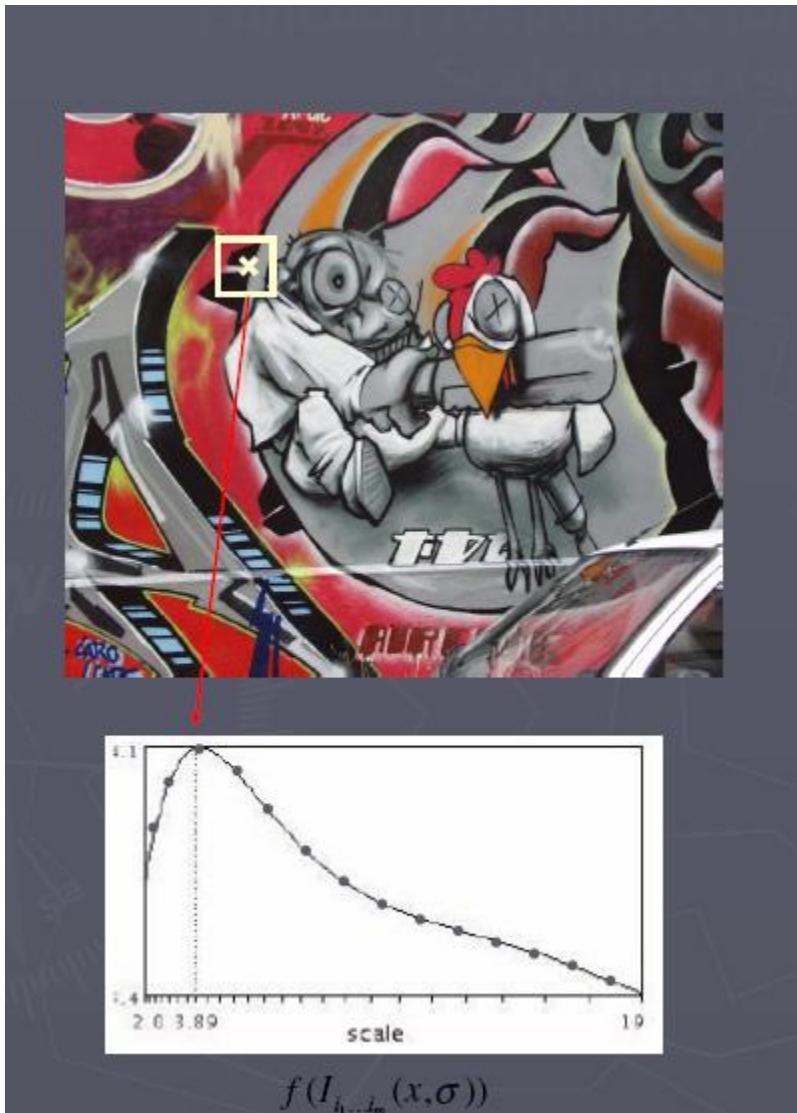
Automatic scale detection



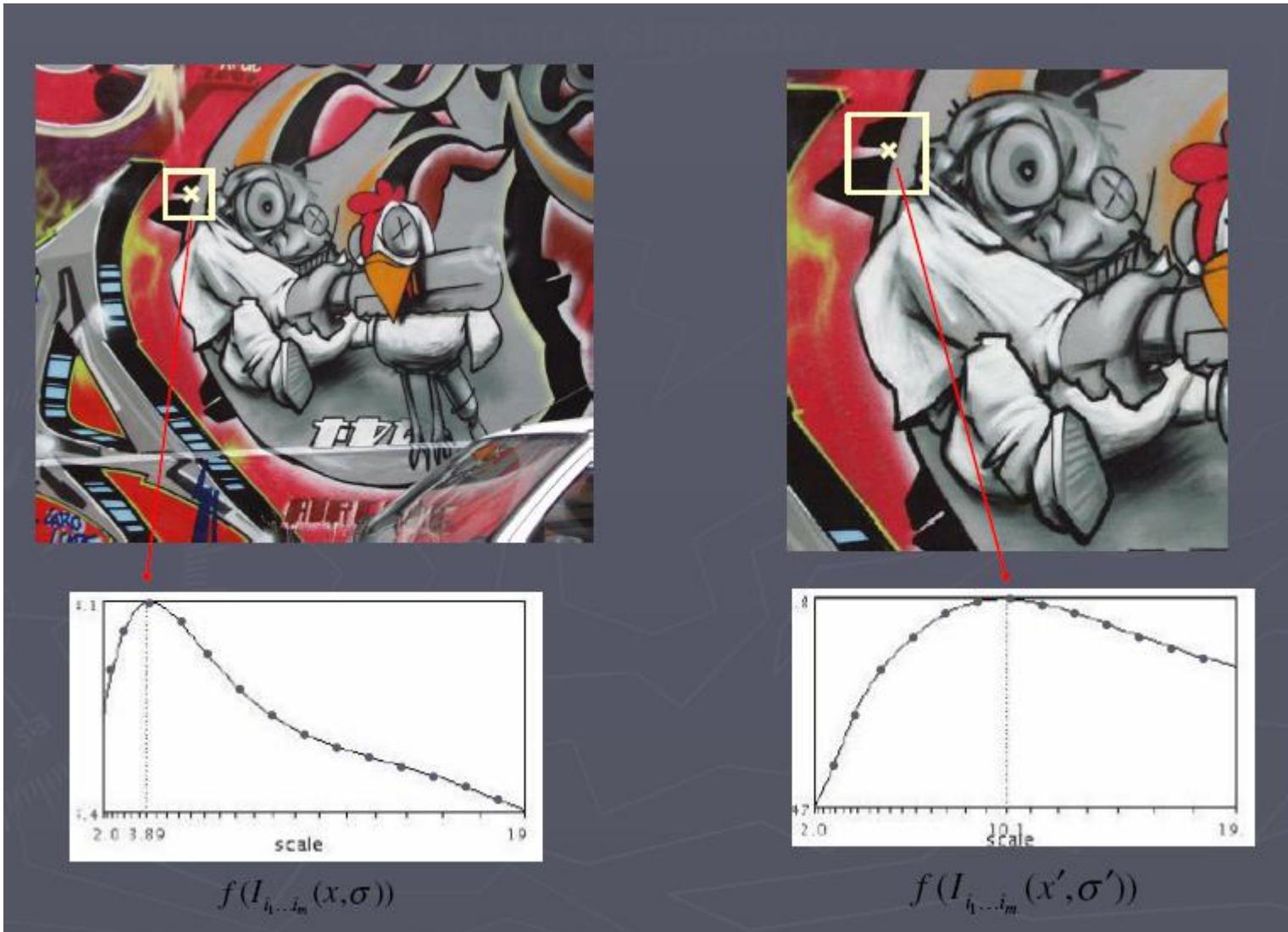
Automatic scale detection



Automatic scale detection

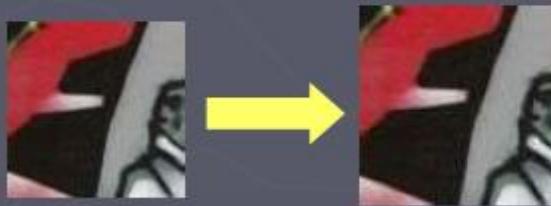


Automatic scale detection



Automatic scale detection

Normalize: rescale to fixed size



The Censure feature detector (Alternative of Harris corner detector)

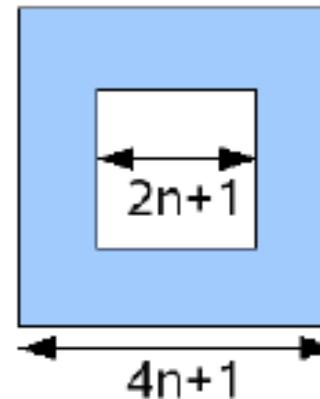
Use difference of boxes to compute image derivatives.

Censure approximates corners and determine the optimal scale but not the orientation!

- Why?

Skimage:

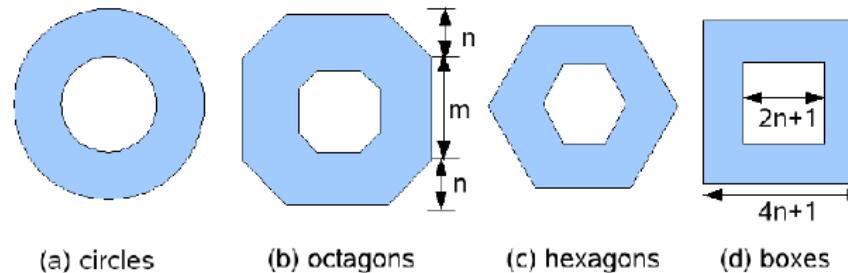
```
>> detector = CENSURE()  
>> detector.detect(img_orig)  
>> plt.imshow(img_orig, cmap=plt.cm.gray)  
>> plt.scatter(detector.keypoints[:, 1], detector.keypoints[:, 0], 2 ** detector.scales,  
    facecolors='none', edgecolors='r')
```



The Censure feature detector



The Censure feature detector (Alternative of Harris corner detector)



Consider different scales: n is the scale of the filer ($n=1,2,3,4,5,6,7$)

Interior weights I_n and outer weights O_n are given by:

$$O_n(4n + 1)^2 = I_n(2n + 1)^2$$

Why?

The Censure feature detector

- Illumination invariance?



- Rotation invariance?



(approximate)

- Scale invariance?



(approximate)

Stability: the persistence of features across viewpoint change.

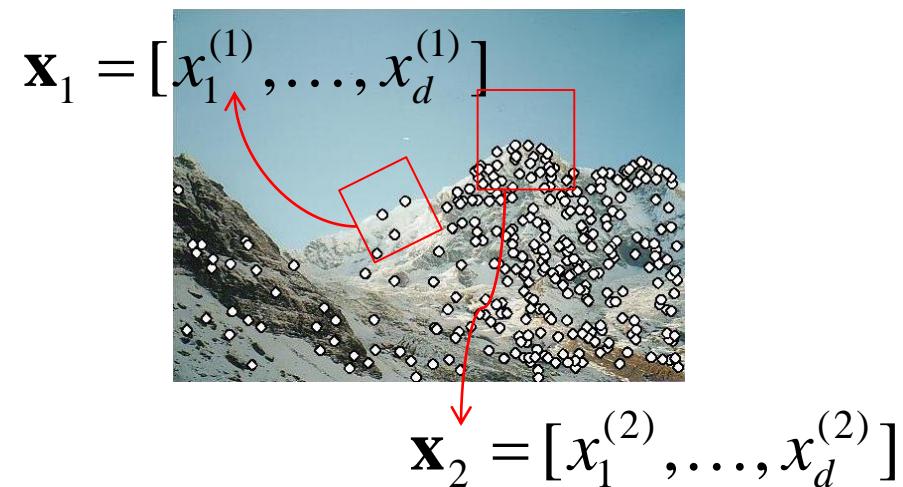


Accuracy: the consistent localization of a feature across viewpoint change.

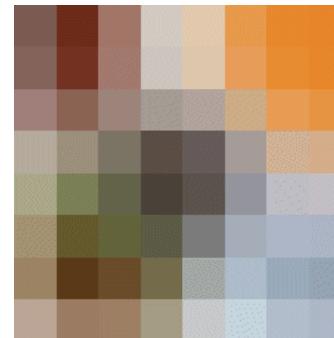
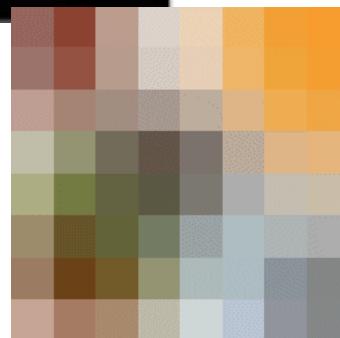
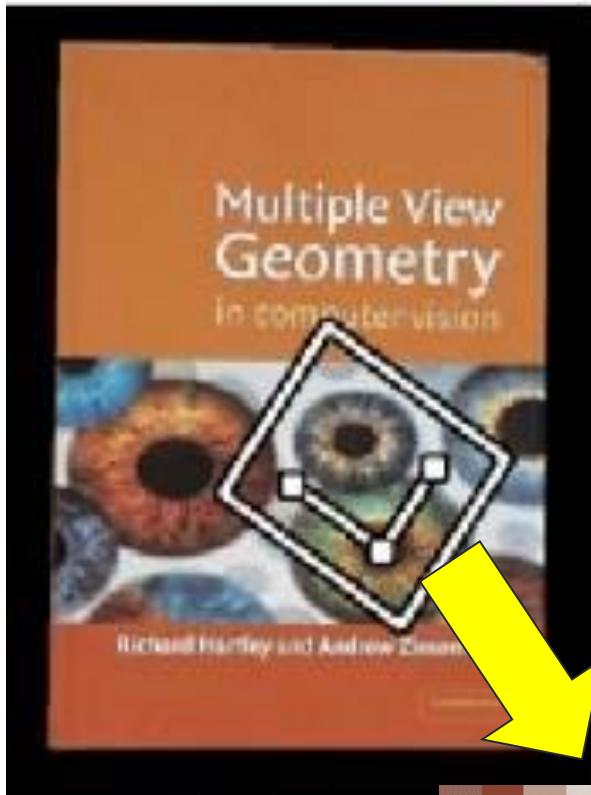


Local features: main components

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

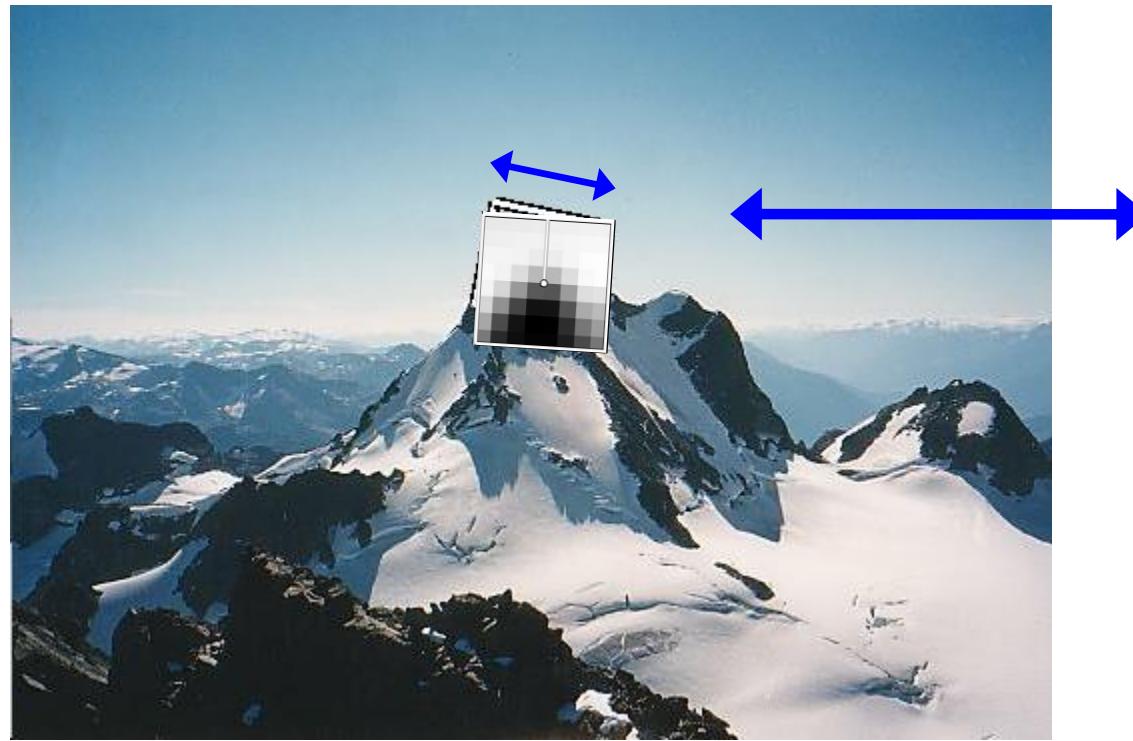


Geometric transformations



e.g. scale,
translation,
rotation

Making the descriptor rotationally invariant



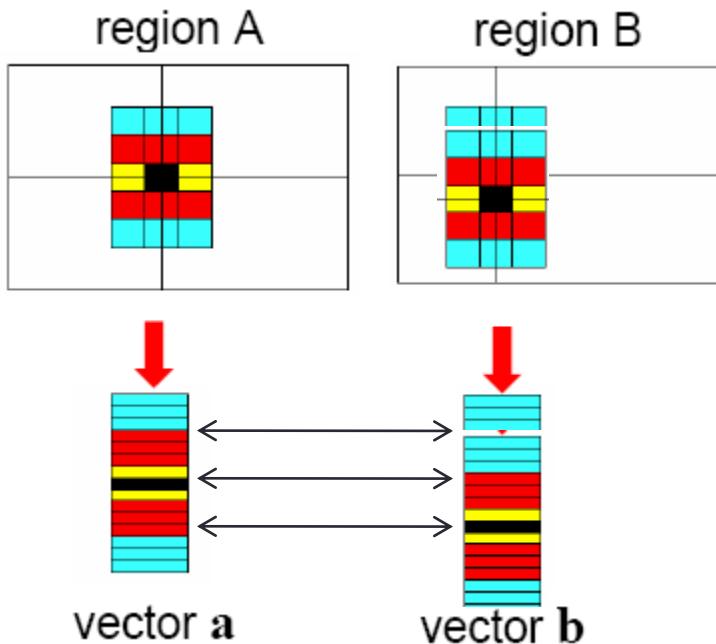
- Rotate patch according to its dominant gradient orientation
 - It can be shown that the dominant direction is given by the first eigenvector (e_1 , if $|\lambda_1| > |\lambda_2|$, and e_2 , vice versa) of M .
- This puts the patches into a canonical orientation.

Photometric transformations



Figure from T. Tuytelaars ECCV 2006 tutorial

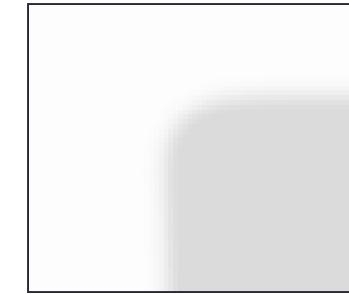
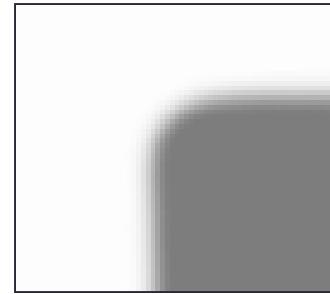
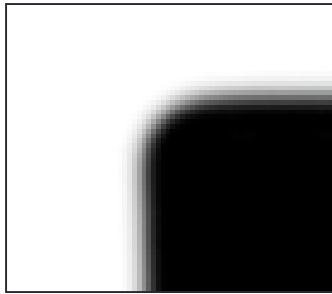
Raw patches as local descriptors



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.

Illumination invariance



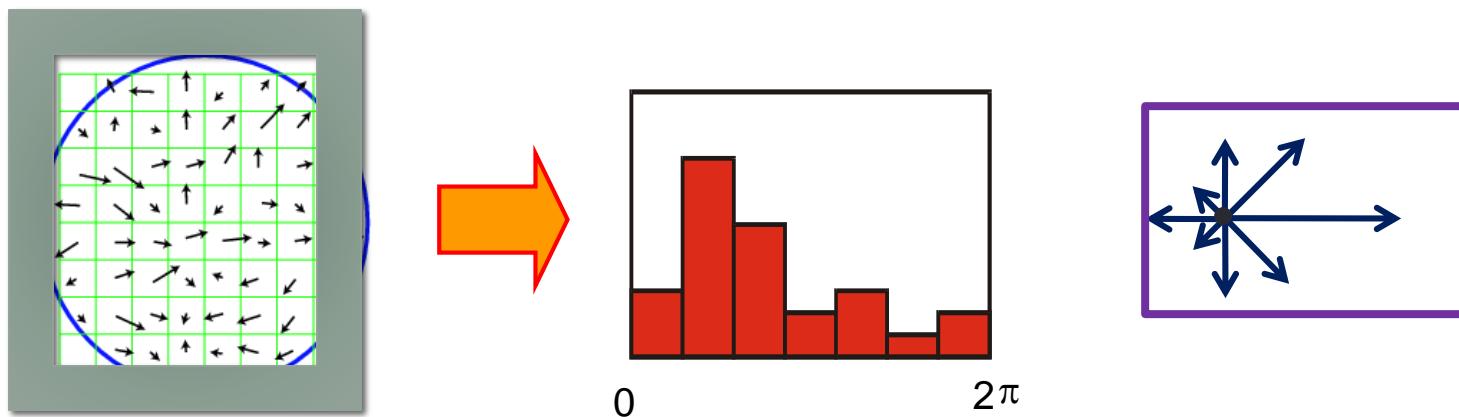
What does happen when image illumination change?

- Image gradient magnitude?
- Image gradient direction?

=> Define Sift descriptors based on Gradient direction!

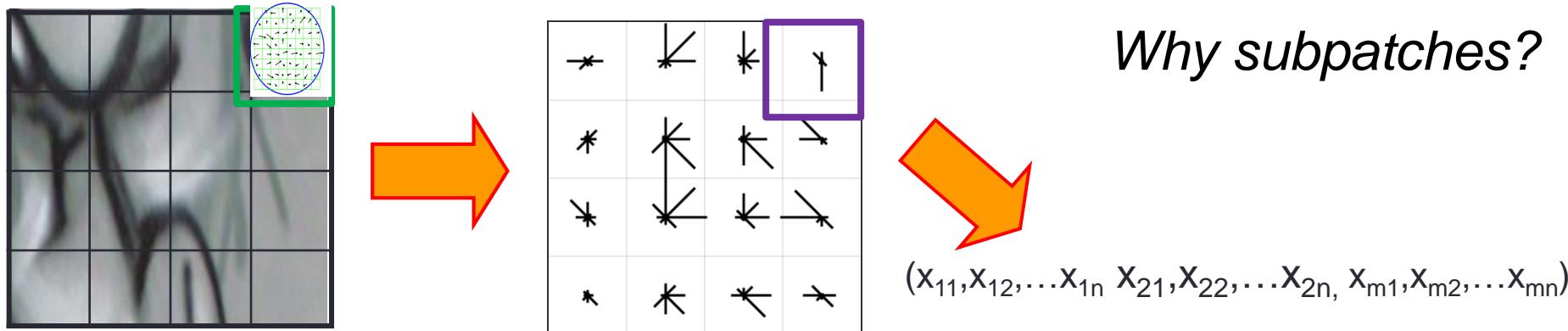
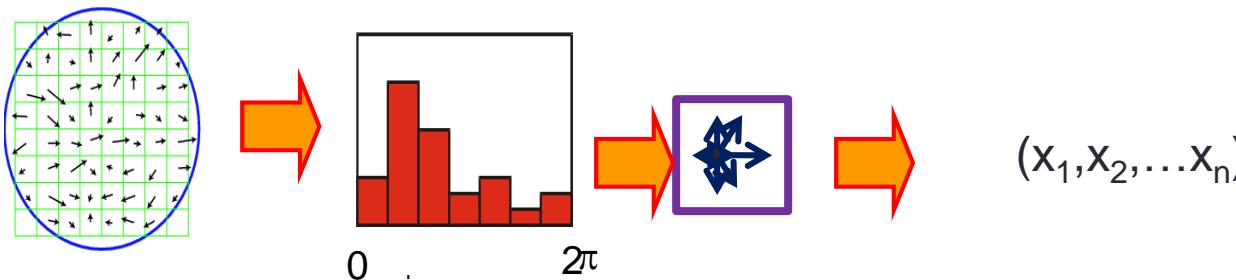
SIFT descriptor [Lowe 2004]

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



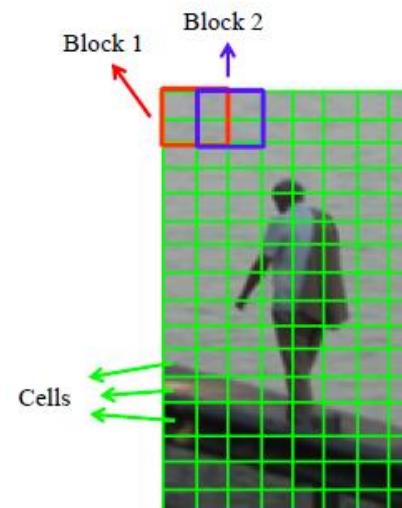
SIFT descriptor [Lowe 2004]

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



Recall: Histogram of gradient (HOG)

- For a 64x128 image
- Divide the image in cells of 8x8 pixels (8x16 cells)
- Group cells into blocks of 2x2 cells (16x16 pixels) of 50% overlap
- Total number of blocks: $7 \times 15 = 105$
- Quantize the gradient orientation into 9 bins
- Concatenate histograms: $105 \times 4 \times 9 = 3780$ feature vector



ORB feature descriptor (Approximation of SIFT)

- Since Sift is patented, a free feature detector – ORB.
- Advantages
 - fast, computational efficient
 - real time and low power device
- ORB is based on a Detector that
 - measures intensity between center pixel and those in a circular ring around the center

ORB feature descriptor

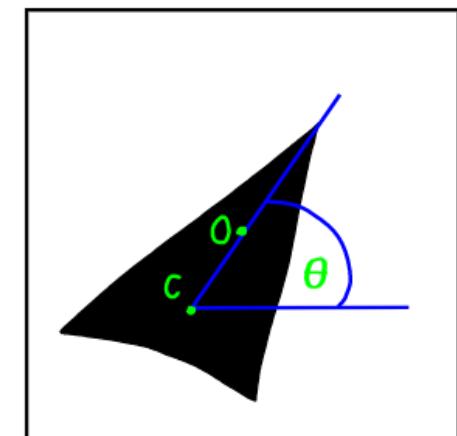
ORB orientation:

Assumption based on the Intensity Weighted Center of Mass: corner's intensity is offset from center:

$$\forall \underbrace{p, q \in \{0, 1\}}_{\text{binary selector for } x \text{ and } y \text{ direction}} : m_{pq} = \sum_{\substack{x, y \\ \text{circular window}}} \underbrace{x^p y^q}_{\text{weighted by coordinate}} \underbrace{I(x, y)}_{\text{image function}}$$

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right)$$

dominant direction \overrightarrow{OC} : $\theta = \text{atan2}(m_{01}, m_{10})$

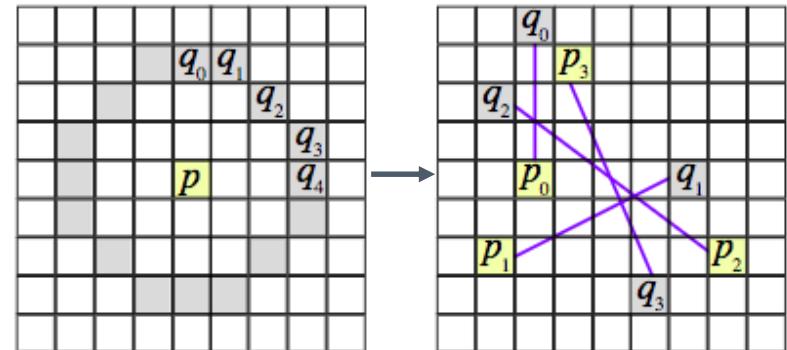


ORB descriptor

$$\tau(p; x, y) = \begin{cases} 1 & \text{if } p(x) \geq p(y) \\ 0 & \text{otherwise} \end{cases}$$

feature vector $f_n(p) := \sum_{1 \leq i \leq n} 2^{i-1} \tau(p; x_i, y_i)$

vector length $n = 256$ bit



Pixel location p and 256 pairs of pixel locations around p ;
 $f_n(p) = \tau(p_0, q_0) \cdot 2^0 + \tau(p_1, q_1) \cdot 2^1 + \dots + \tau(p_{255}, q_{255}) \cdot 2^{255}$

Which pairs provide best information for a given patch?

- maximize variance and thus minimize correlation.

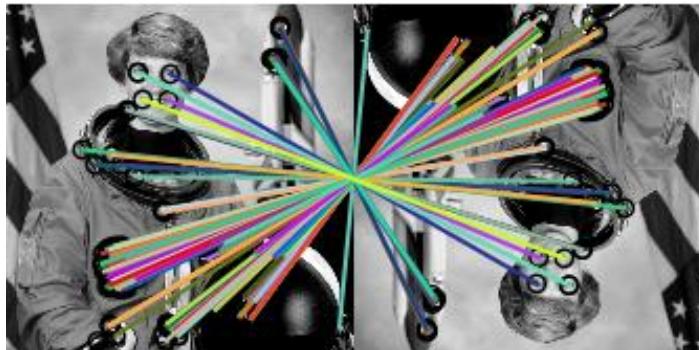
ORB learns which pairs to consider

- input: reference image series / output: static list of pairs

ORB descriptor

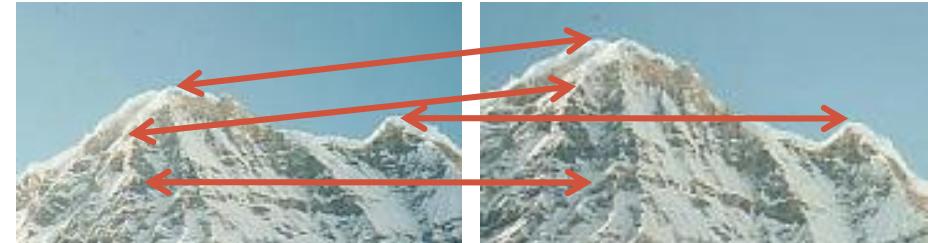
- Skimage

```
>>descriptor_extractor = ORB(n_keypoints=200)
>>descriptor_extractor.detect_and_extract(img1)
>>keypoints1 = descriptor_extractor.keypoints
>>descriptors1 = descriptor_extractor.descriptors
...
>>matches12 = match_descriptors(descriptors1, descriptors2)
>>plot_matches(fig, img1, img2, keypoints1, keypoints2, matches12)
```



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



ORB descriptor

- Illumination invariance?
- Rotation invariance?
- Scale invariance?



Shortcomings of the ORB detection approach:

1. No quality measure (“cornerness”)
solution: use Harris cornerness measure
2. Not scale invariant
solution: use SIFT scale invariance

Matching local features



Matching local features



Image 1

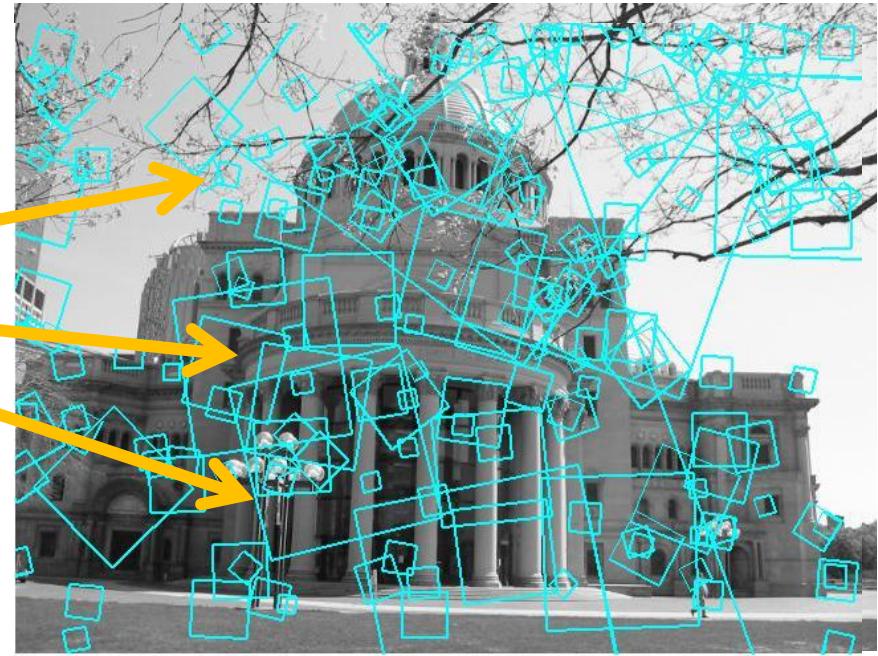


Image 2

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?

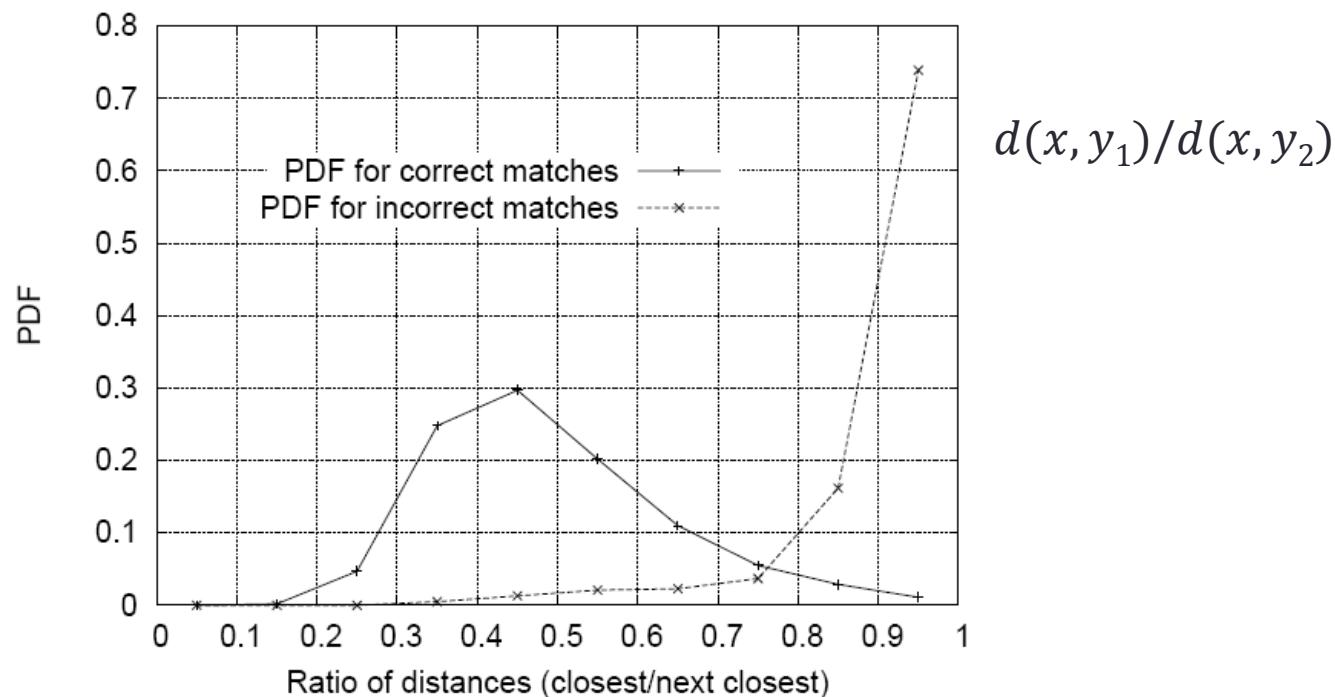
To add robustness to matching, we can consider the **ratio**: distance (difference) to best match / distance to second best match

If low, first match looks good.

If high, it could be ambiguous match.

Matching SIFT Descriptors

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



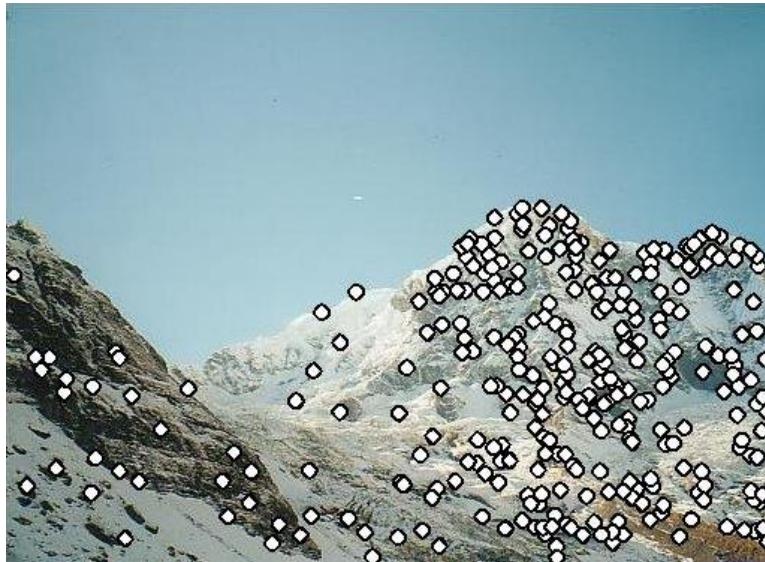
Do you like to do panoramic images?



Robust feature-based alignment

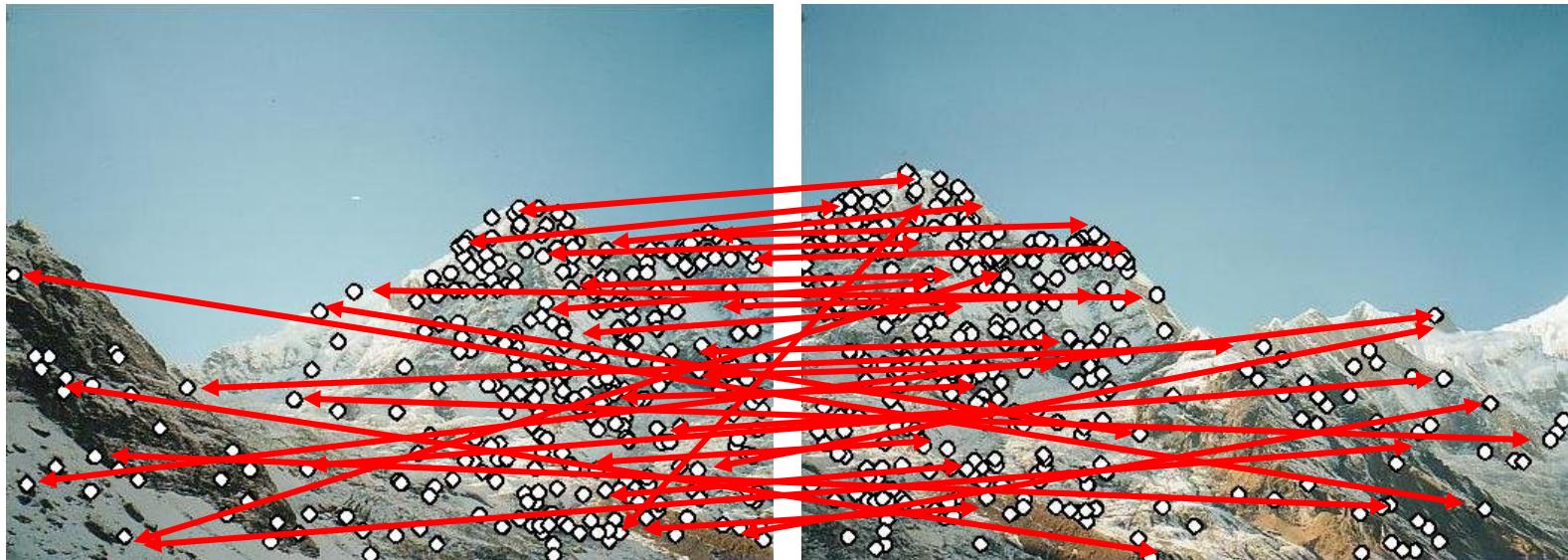


Robust feature-based alignment



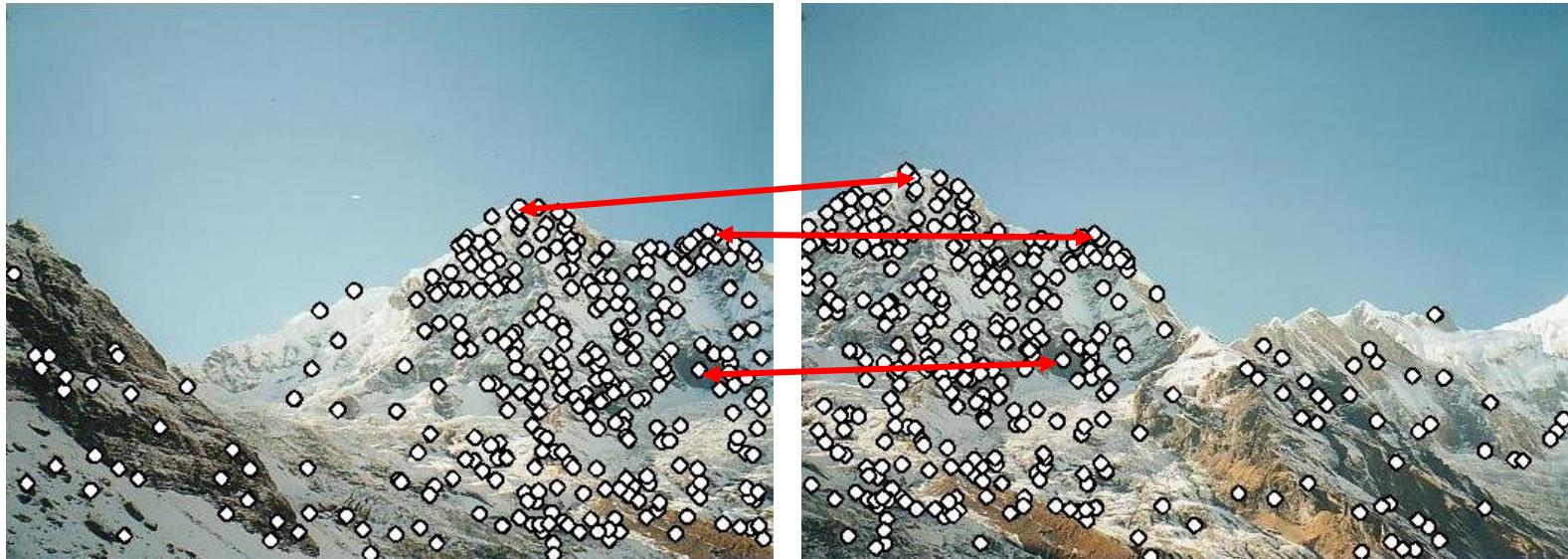
- Extract features

Robust feature-based alignment



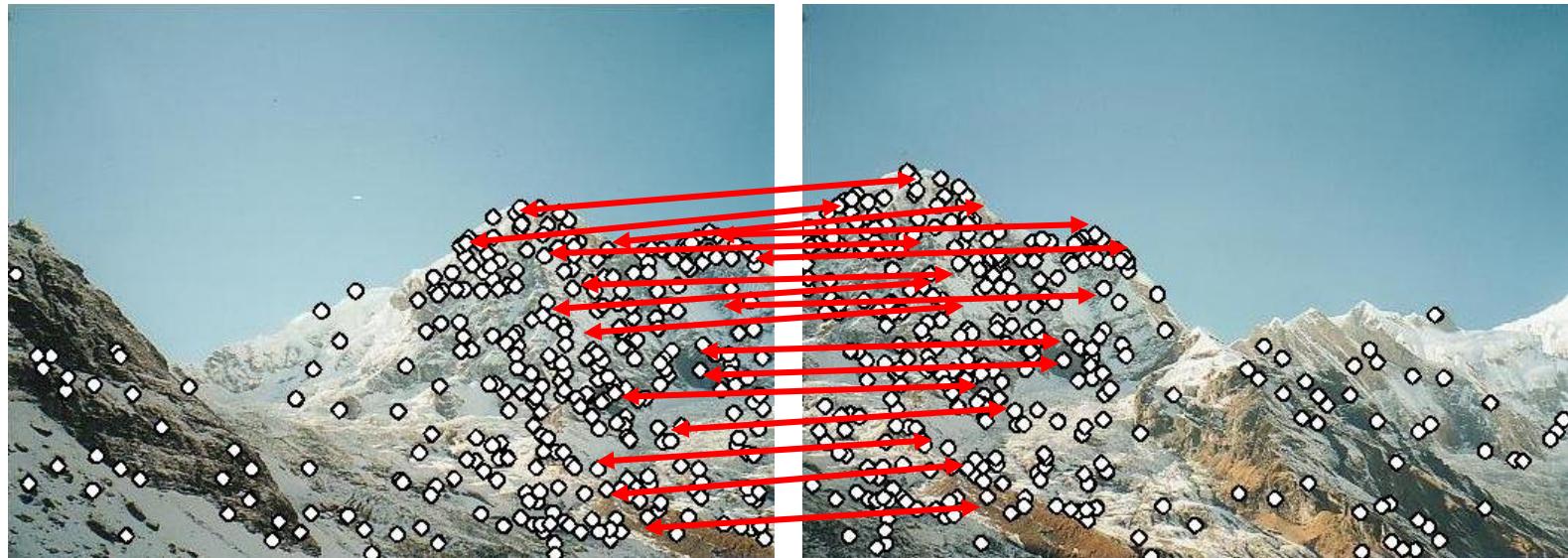
- Extract features
- Compute *putative matches*

Robust feature-based alignment



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

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)

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)

Mosaic construction by the Ransac method

The algorithm:

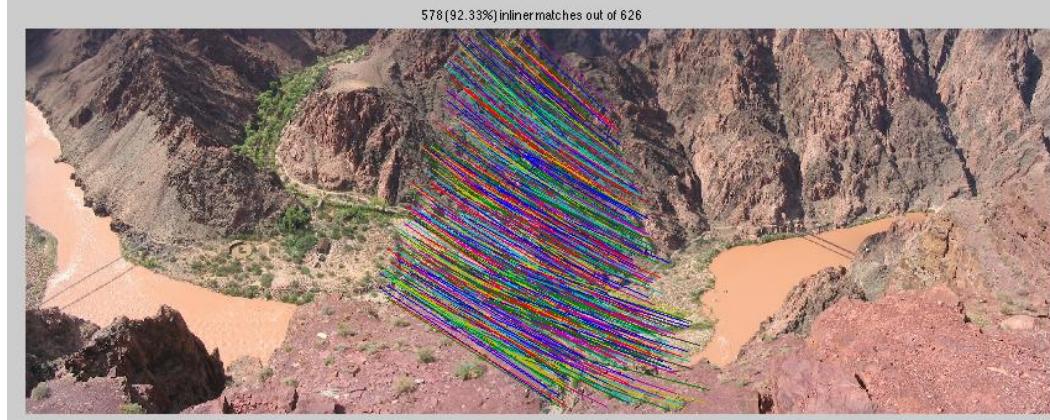
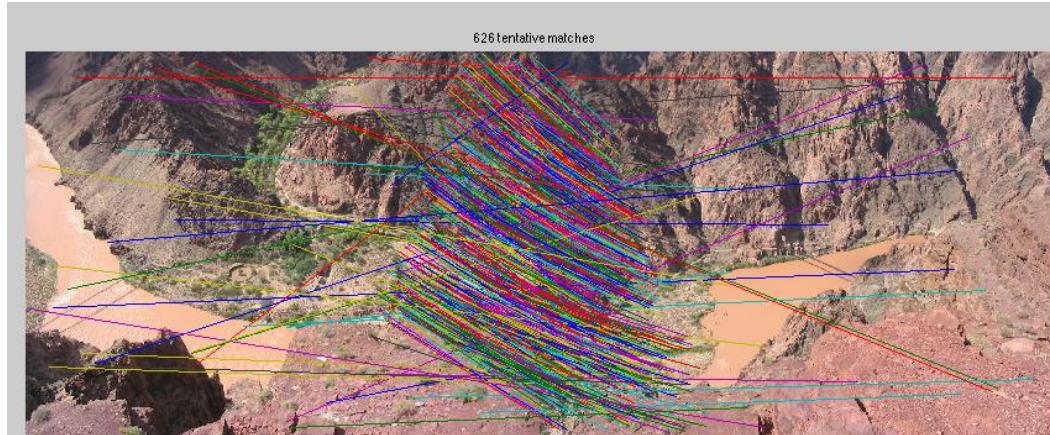
1. Select a random subset of the original data (hypothetical inliers).
2. A model is fitted to the set of hypothetical inliers.
3. All other data are then tested against the fitted model.
 - Those points that fit the estimated model well, according to some model-specific loss function, are considered as part of the consensus set.
4. The estimated model is reasonably good if sufficiently many points have been classified as part of the consensus set.
5. 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.
6. 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



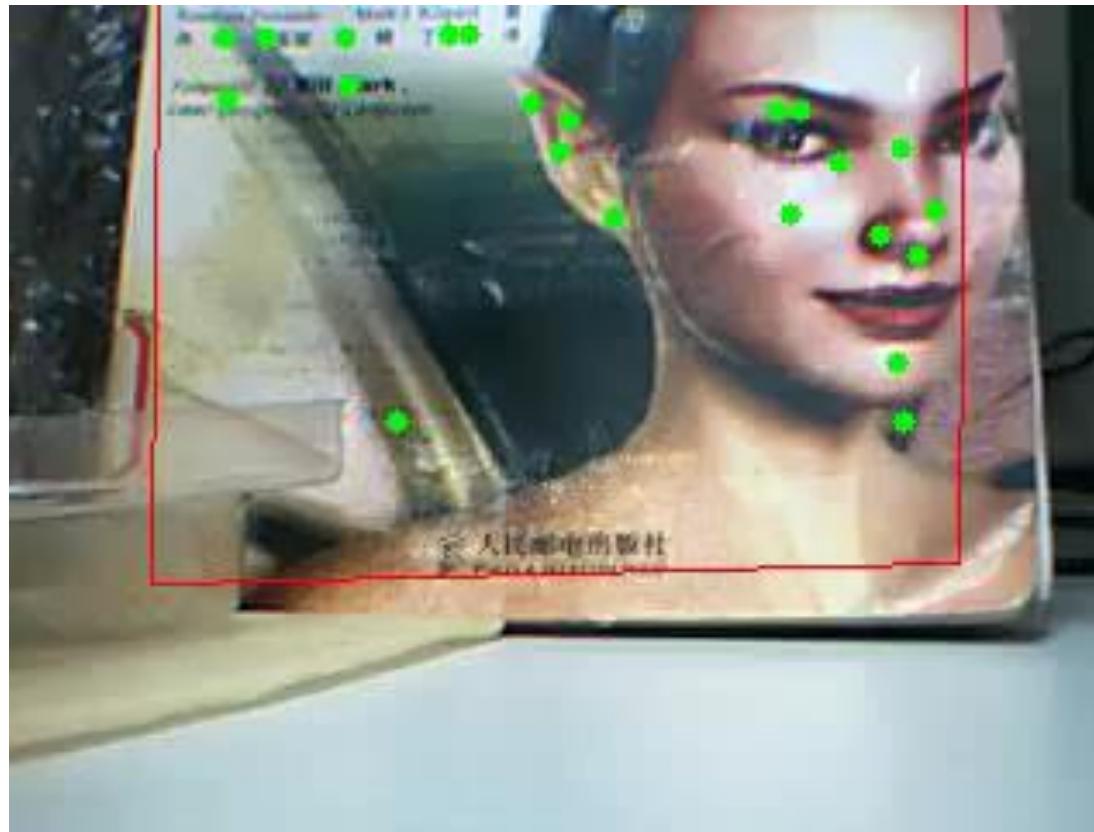
Construct the mosaic through SIFT and Ransac



Construct the mosaic through SIFT and Ransac



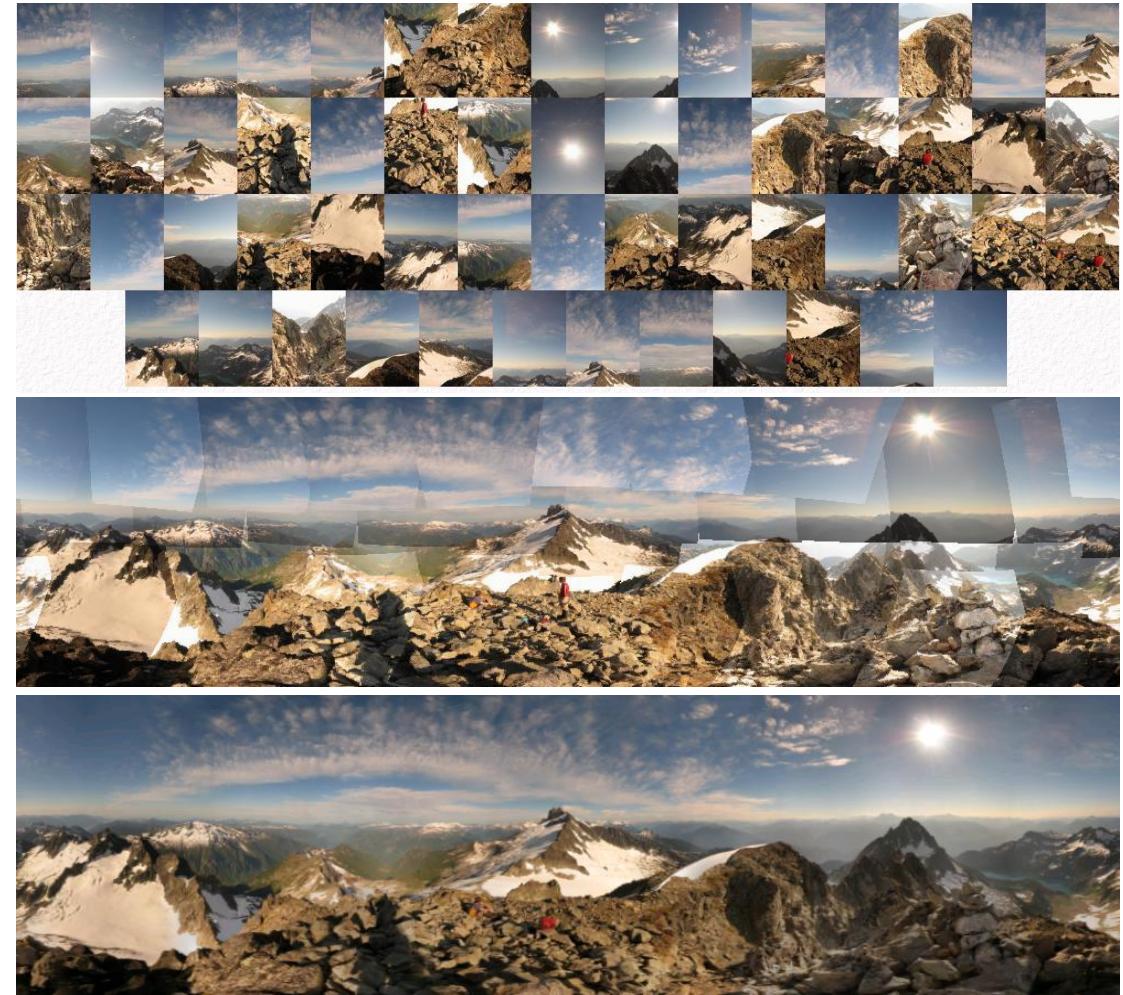
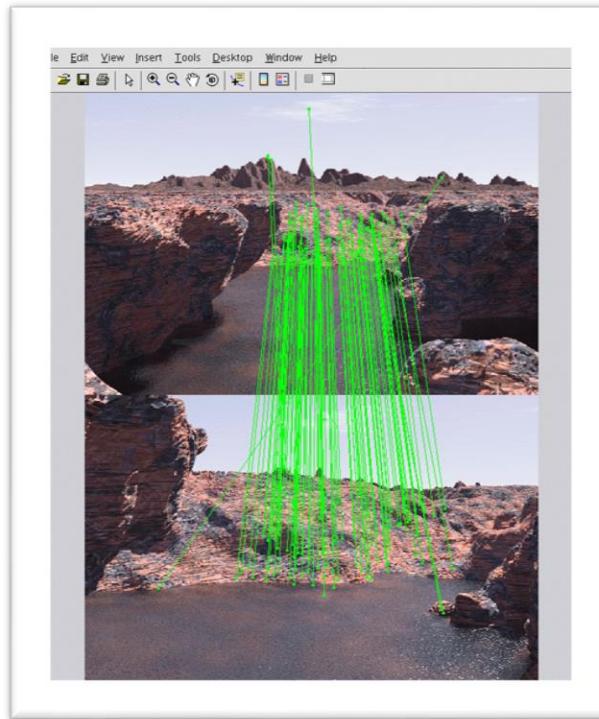
Stability



Applications of local invariant features SIFT

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

Automatic mosaicing



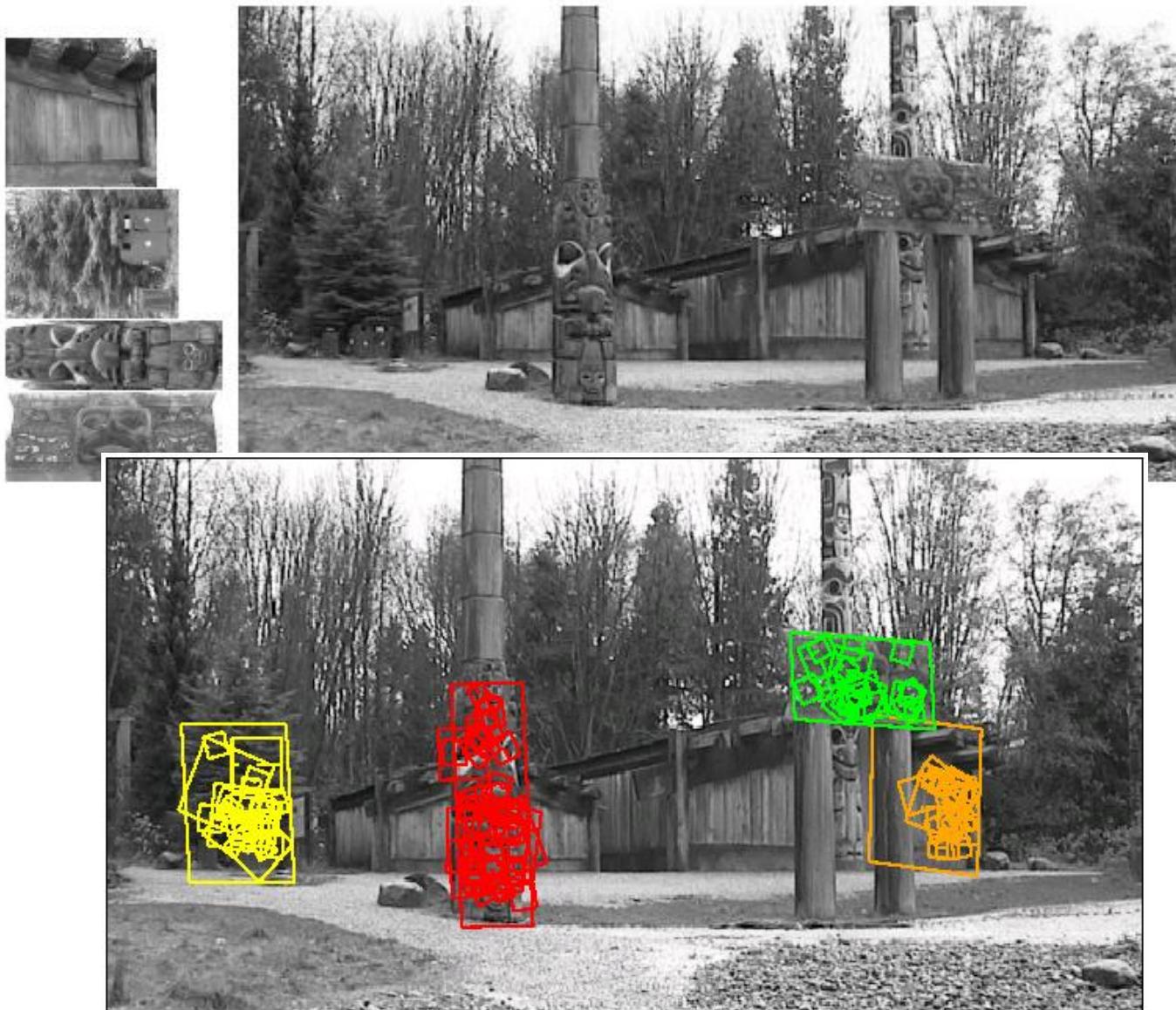
<http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html>

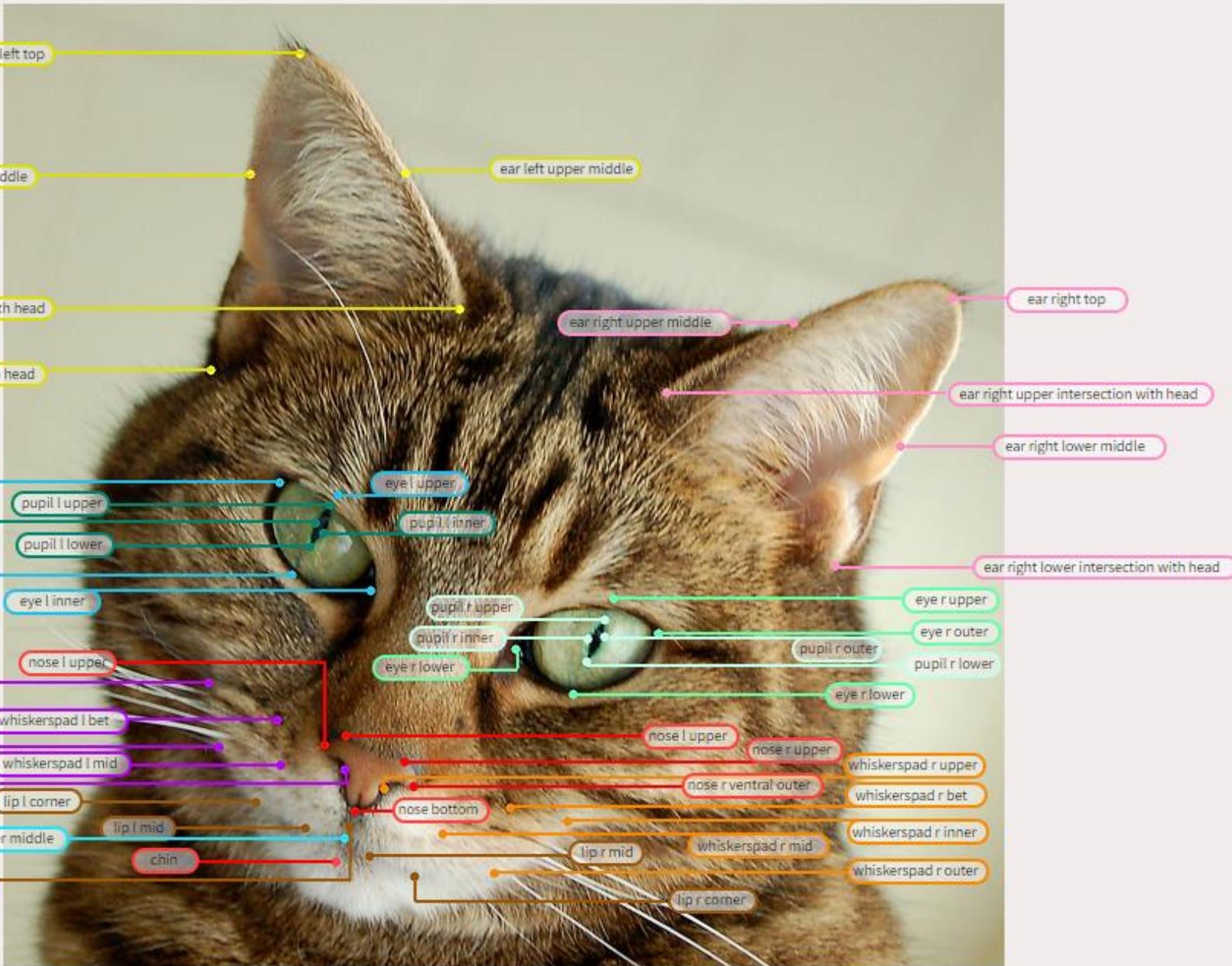
Stereo images



[Image from T. Tuytelaars ECCV 2006 tutorial]

Object detection





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



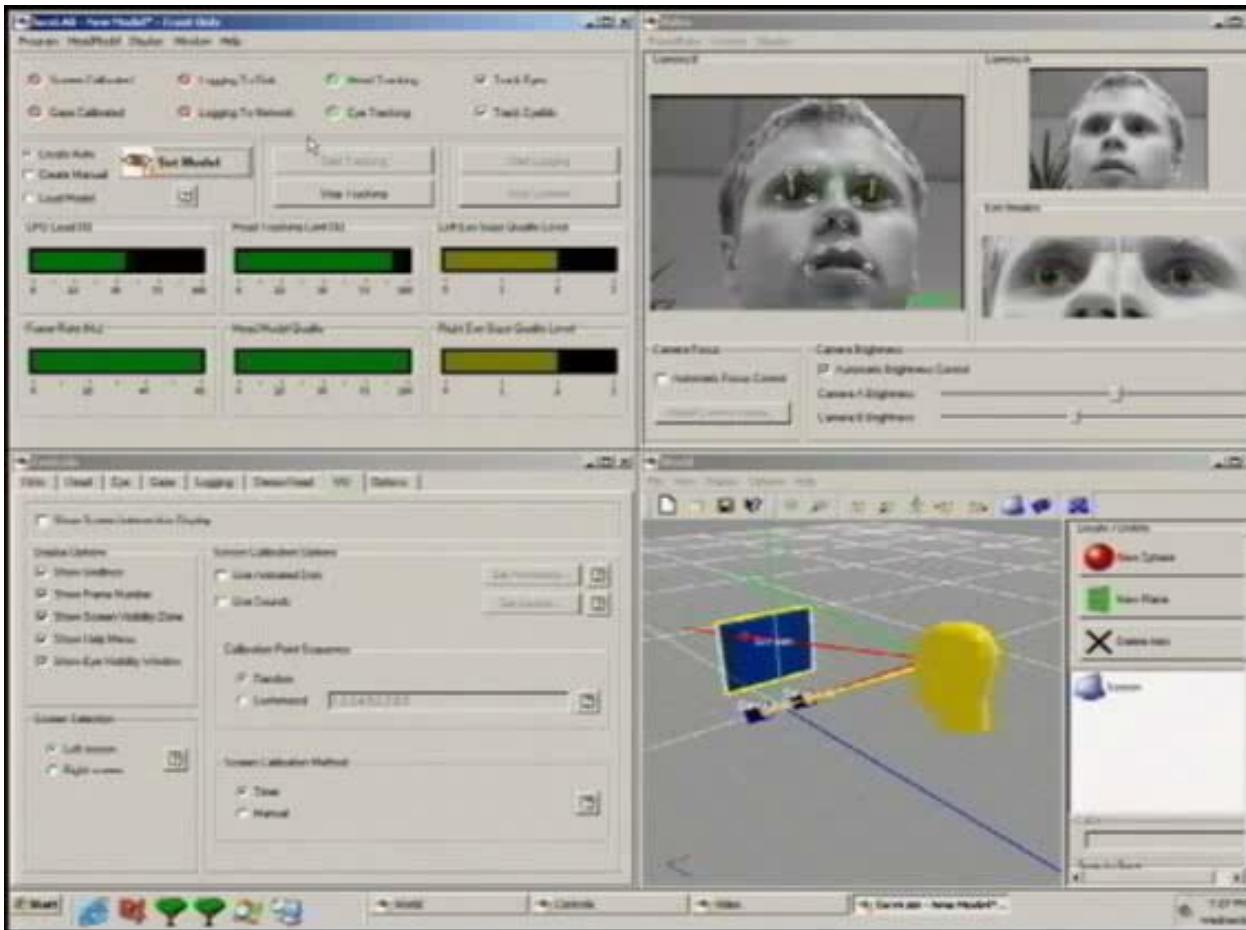
Sivic and Zisserman, 2003



Rothganger et al. 2003

[Sift demo](#)

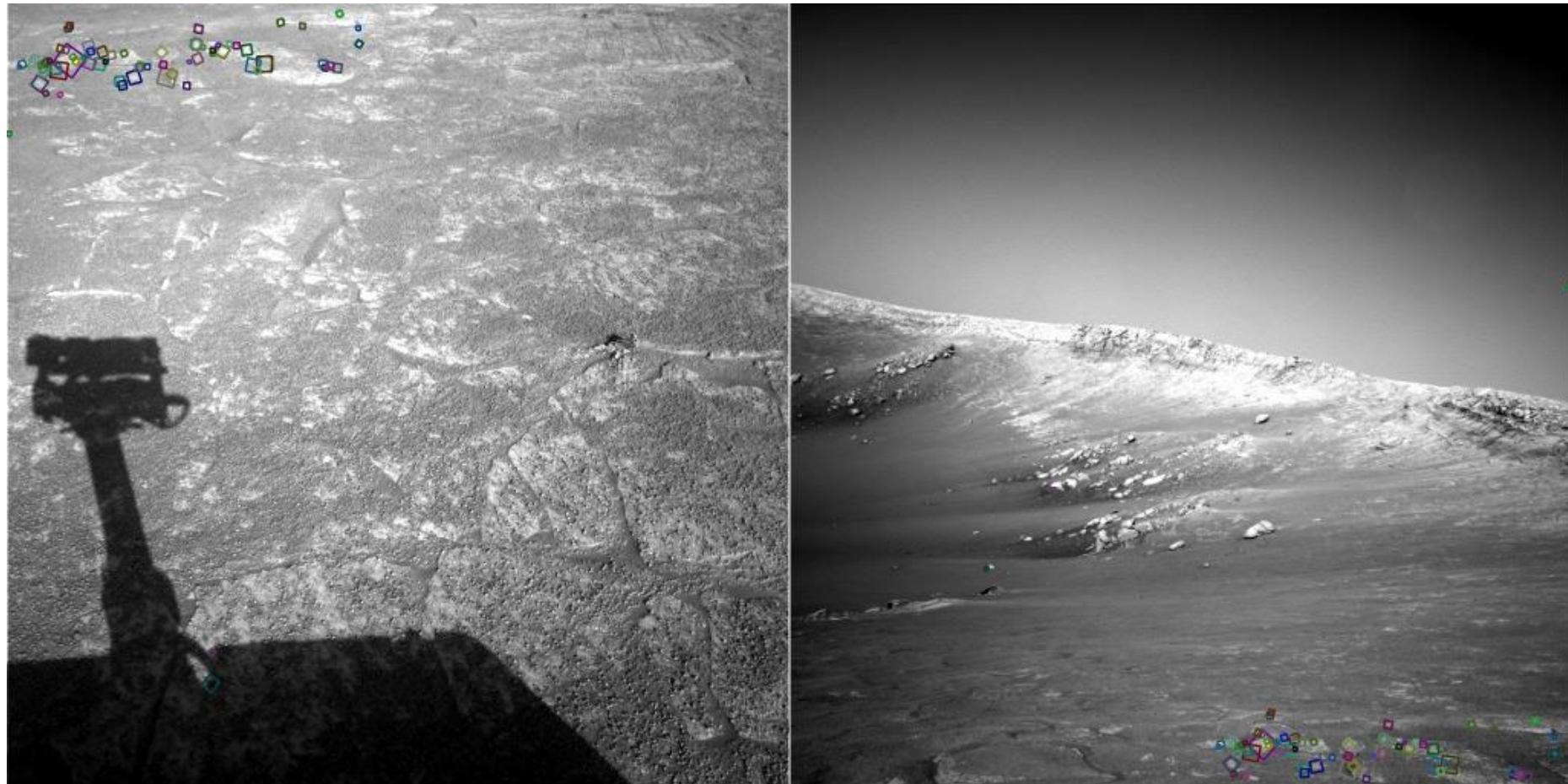
Application FaceLab



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

AIBO® Entertainment Robot
Official U.S. Resources and Online Destinations

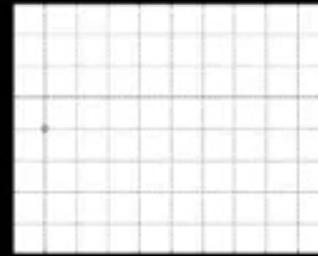
The advertisement features a central image of the Sony AIBO ERS-7 Entertainment Robot, which is white and dog-shaped. It is standing next to a pink ball. To the left of the robot is a small image of a visual card showing a black and white checkered pattern. To the right is a vertical column of text listing accessories: Wireless LAN, AIBO MIND software, Energy Station, AIBOne, Pink Ball, AIBO Cards (15), WLAN Manager CD, and Battery & AC Adapter. At the bottom, it says "3rd Generation Pre-order Now!"

ERS-7
Entertainment Robot AIBO

ERS-7 with:
Wireless LAN
AIBO MIND software
Energy Station
AIBOne
Pink Ball
AIBO Cards (15)
WLAN Manager CD
Battery & AC Adapter

3rd Generation
Pre-order Now!

Sift application



Omnidirectional visual odometry and mosaicing

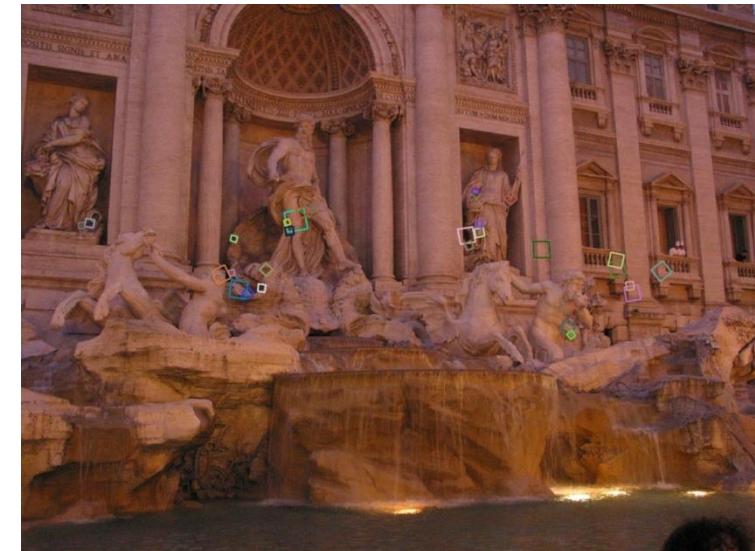
Sift application



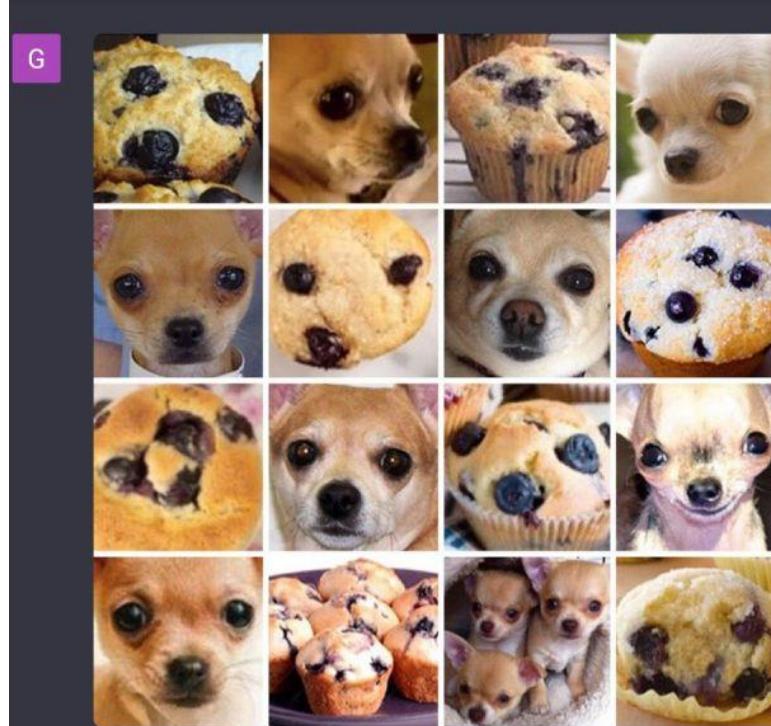
[Markerless Camera Tracking and "Handy" AR User Interface](#)

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



However, sometimes ignoring the context/background could be critical!



Please tell me what is it in each cell in a tabular format aligned with this picture grid.



Certainly! Here's a breakdown of the content of each cell in the format of a table:

Row	Column 1	Column 2	Column 3	Column 4
1	Muffin with blueberries	Chihuahua (side profile)	Blueberry Muffin	Chihuahua (profile looking up)
2	Chihuahua (frontal face)	Cookie with chocolate chips	Chihuahua (face close-up)	Blueberry Muffin with sugar on top
3	Blueberry Muffin with a smiley face	Chihuahua (frontal face close-up)	Blueberry Muffin with blueberries on top	Chihuahua (slightly showing teeth)
4	Chihuahua (puppy face close-up)	Group of Cookies with chocolate chips	Group of Chihuahua puppies	Muffin with blueberries