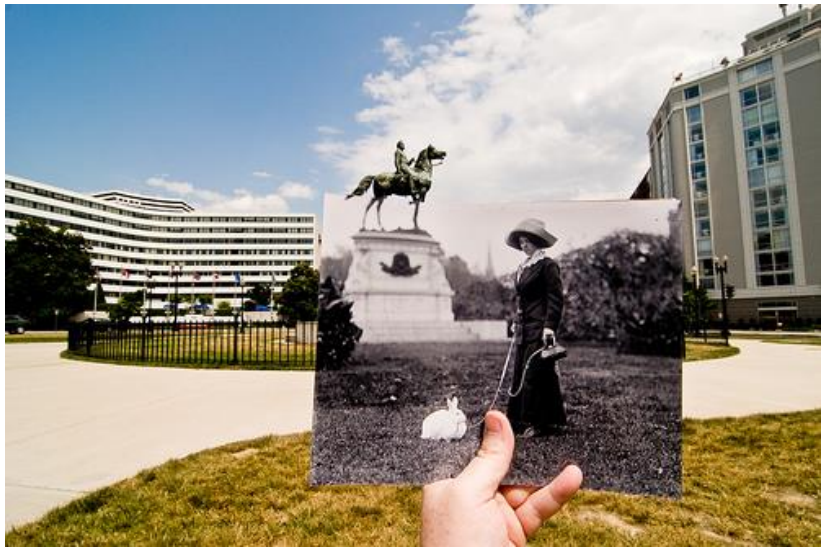
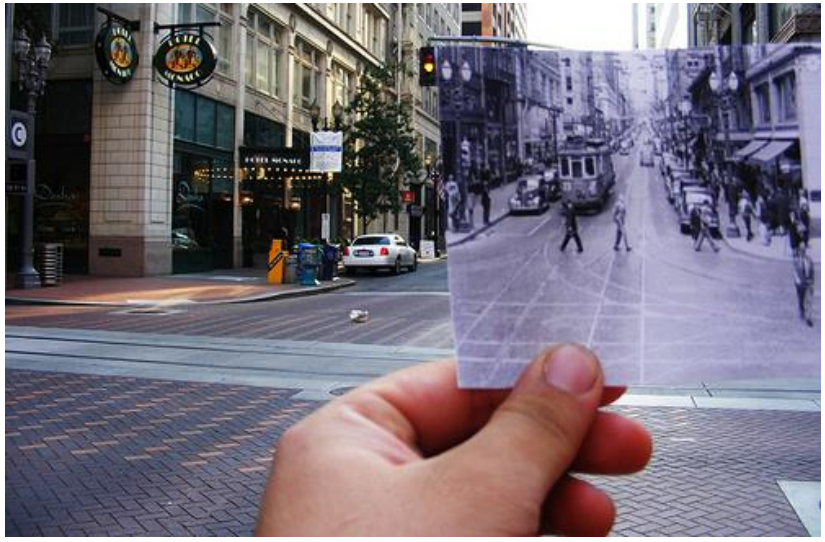
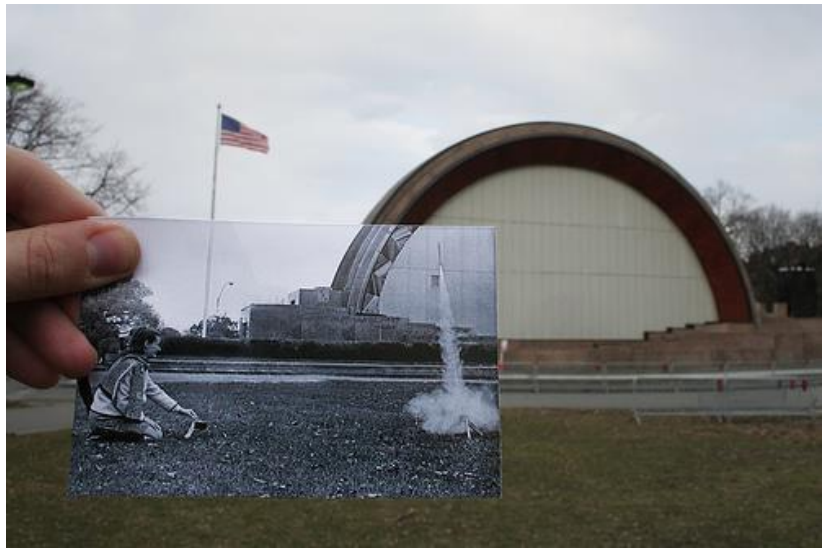
Correspondence and Interest Points



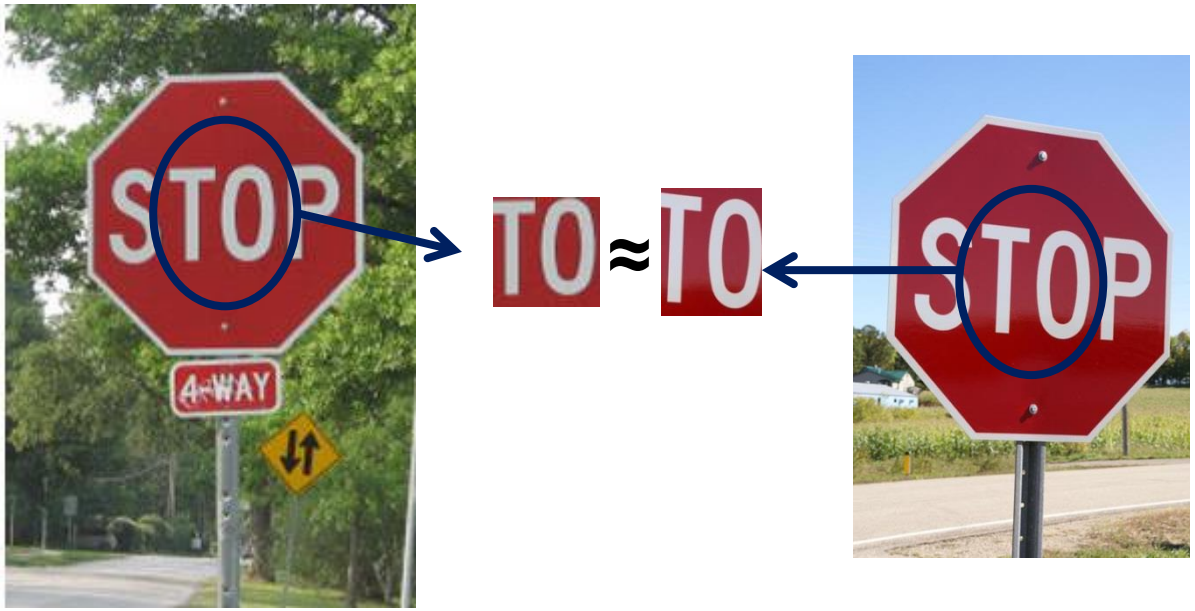




http://blog.flickr.net/en/2010/01/27/a-look-into-the-past/

Correspondence across views

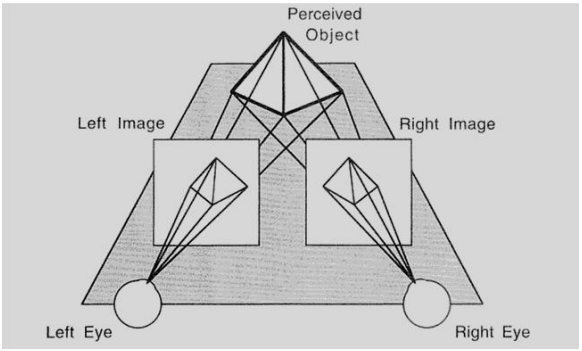
Correspondence: matching points, patches, edges, or regions across images



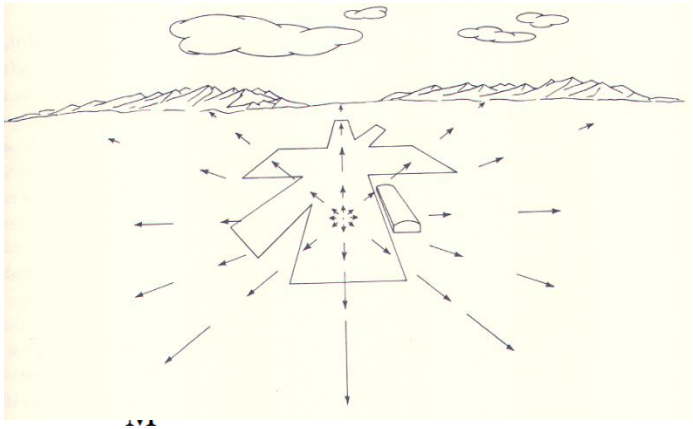
Brightness Constancy may not apply!

Everything depends on correspondence

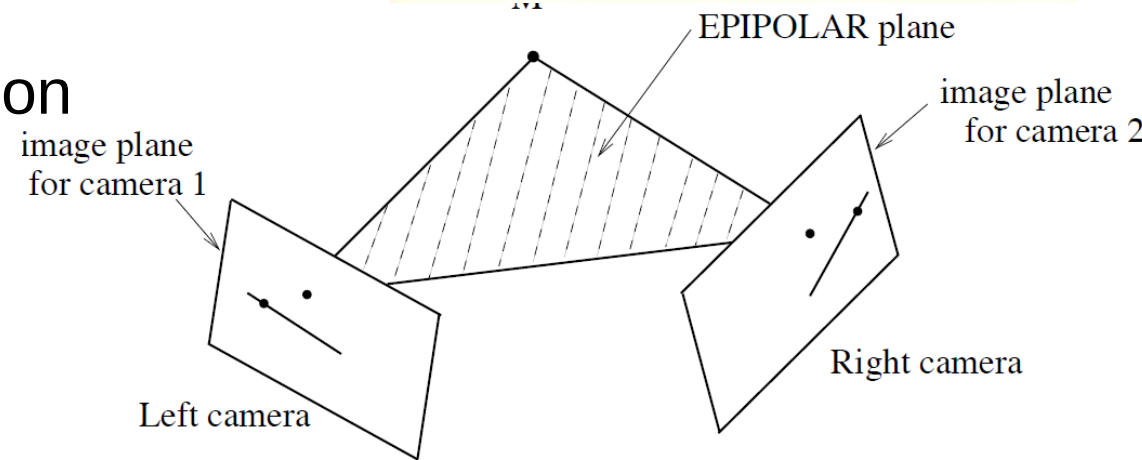
- Stereopsis



- Optical Flow

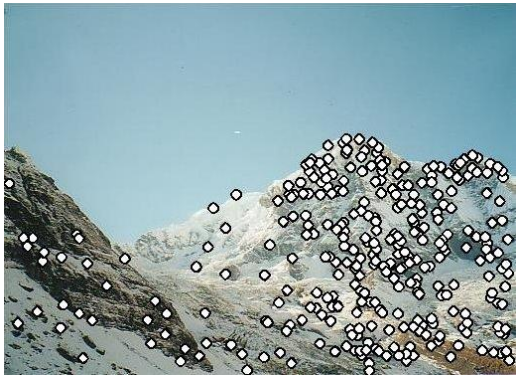


- Structure from Motion



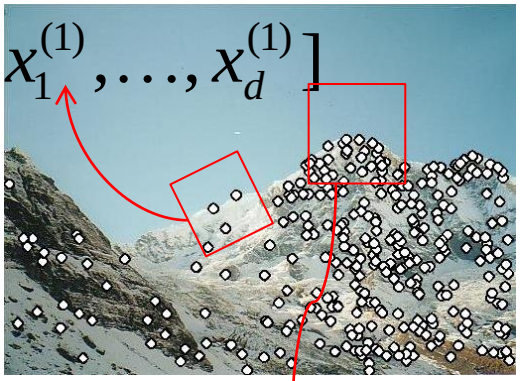
Keypoint Matching: main components

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



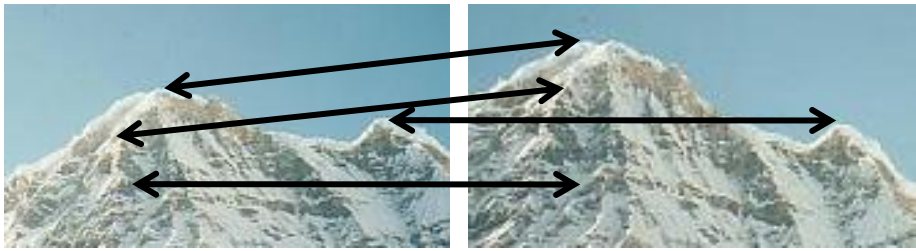
2) **Description**: Extract vector feature descriptor surrounding each interest point.

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



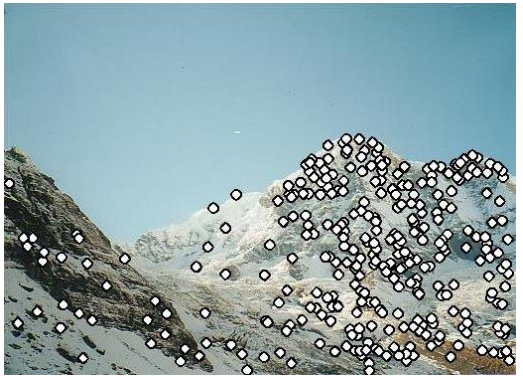
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

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



Local features: main components

1) Detection: Identify the interest points



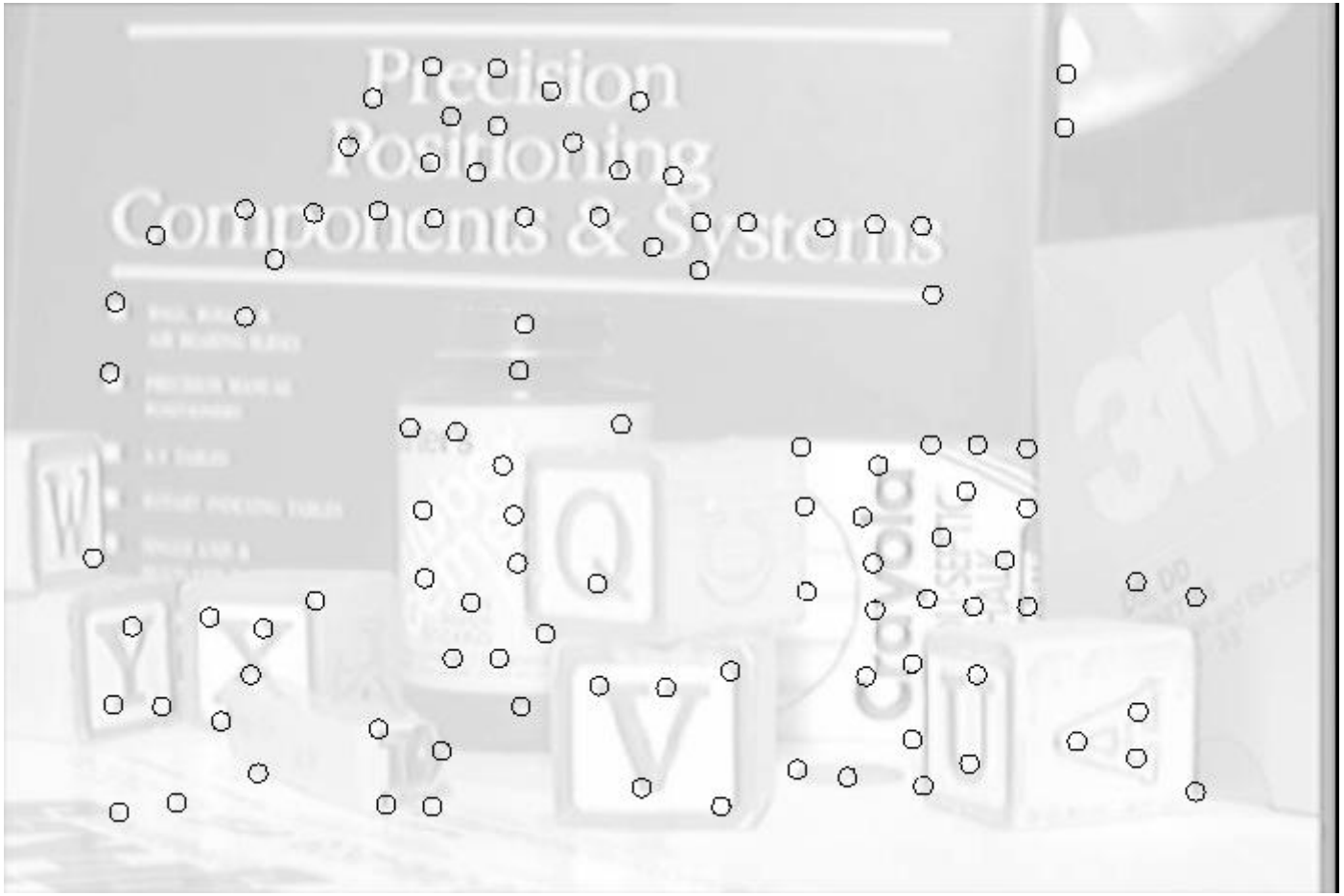
2) Description:Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

Many Existing Detectors Available

Hessian & Harris	[Beaudet ‘78], [Harris ‘88]
Laplacian, DoG	[Lindeberg ‘98], [Lowe 1999]
Harris-/Hessian-Laplace	[Mikolajczyk & Schmid ‘01]
Harris-/Hessian-Affine	[Mikolajczyk & Schmid ‘04]
EBR and IBR	[Tuytelaars & Van Gool ‘04]
MSER	[Matas ‘02]
Salient Regions	[Kadir & Brady ‘01]
Others…	

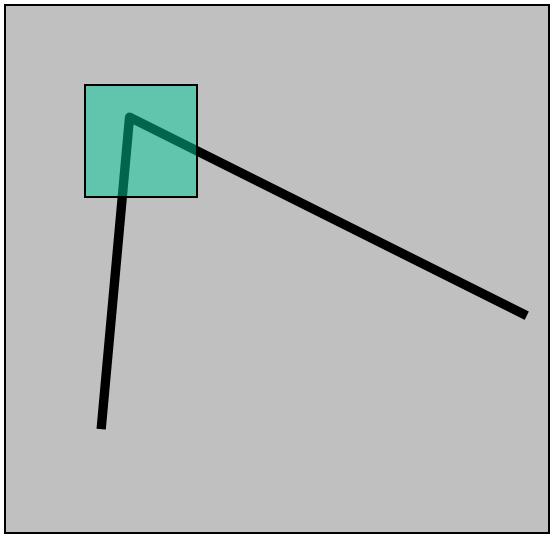
Corner Detection



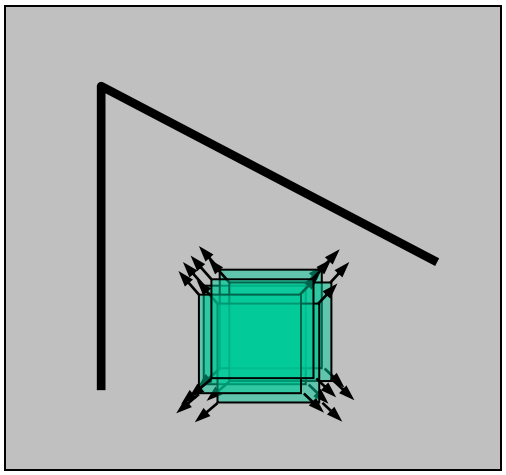
The Basic Idea

We should easily recognize the point by looking through a small window

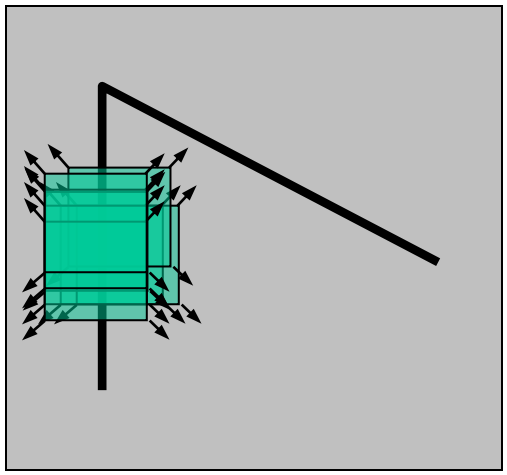
Shifting a window in *any direction* should give *a large change* in intensity



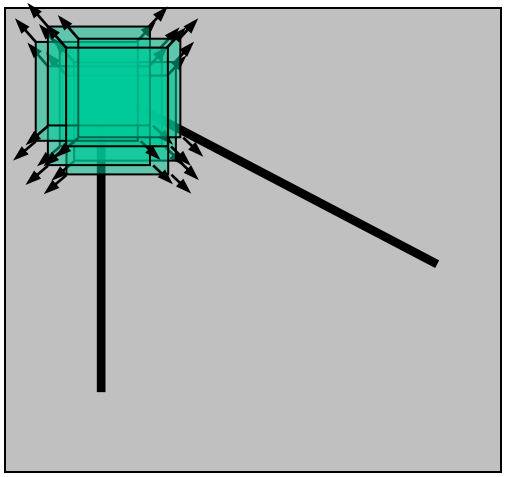
Corner Detector: Basic Idea



"flat" region:
no change in
all directions



"edge":
no change along
the edge direction

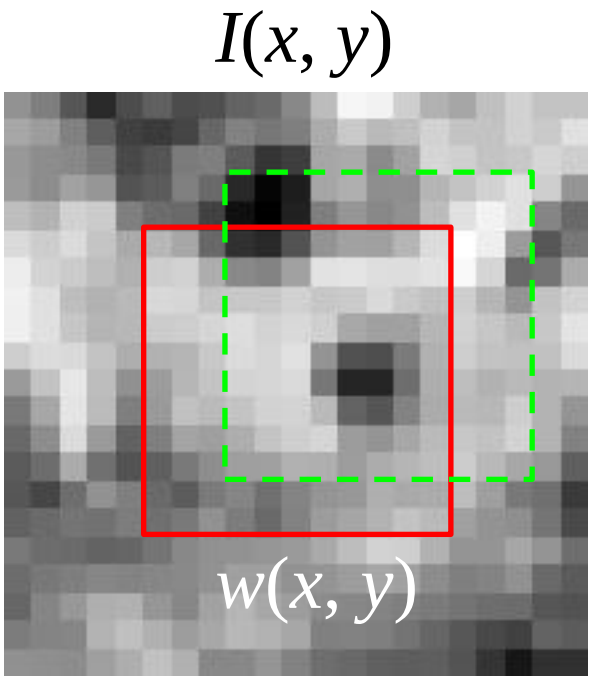


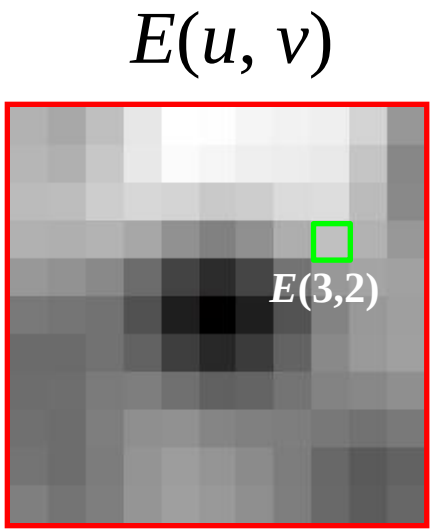
"corner":
significant change
in all directions

Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

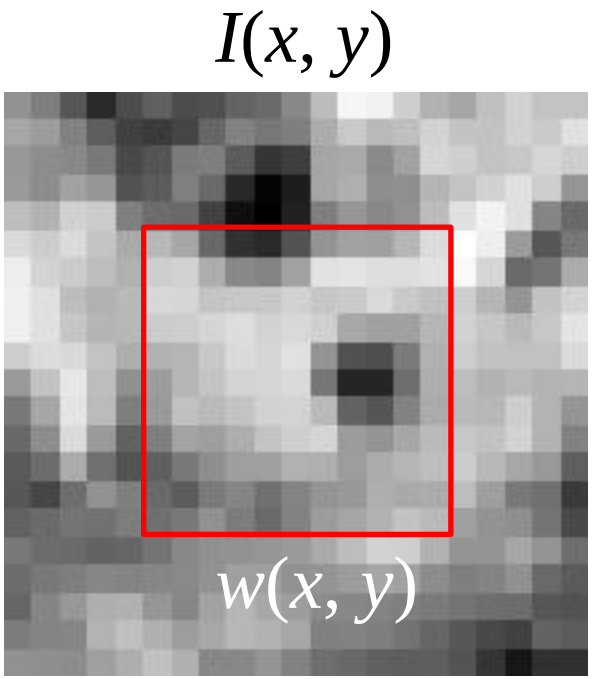


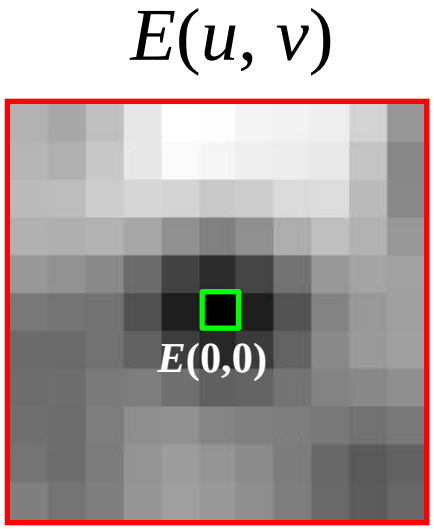


Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$





Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$:

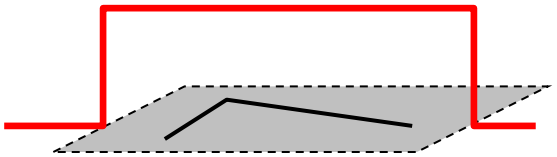
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

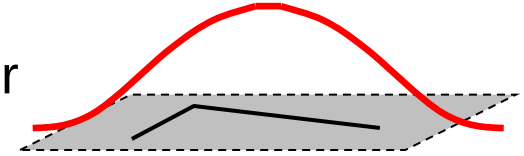
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Source: R. Szeliski

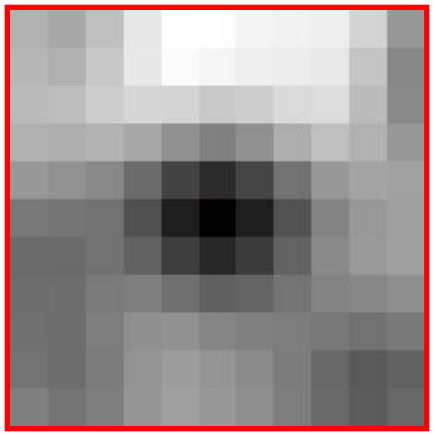
Corner Detection: Mathematics

Change in appearance of window $w(x,y)$ for the shift $[u,v]$:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

We want to find out how this function behaves for small shifts

$E(u, v)$



Corner Detection: Mathematics

The quadratic approximation simplifies to

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

where *M* is a *second moment matrix* computed from image derivatives:

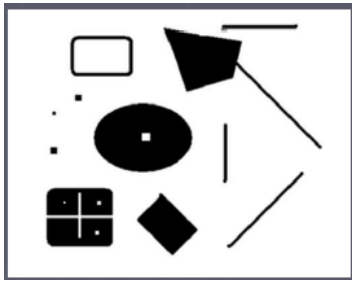
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

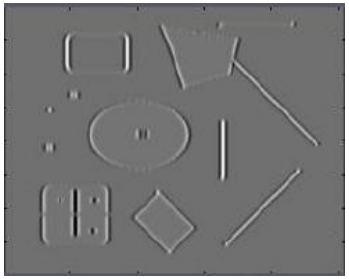
$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y] = \sum \nabla I (\nabla I)^T$$

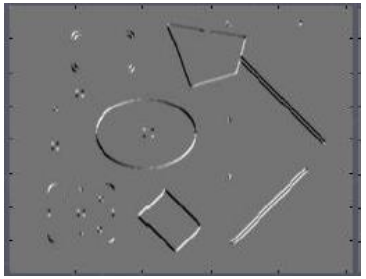
Corners as distinctive interest points

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

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).







Notation:

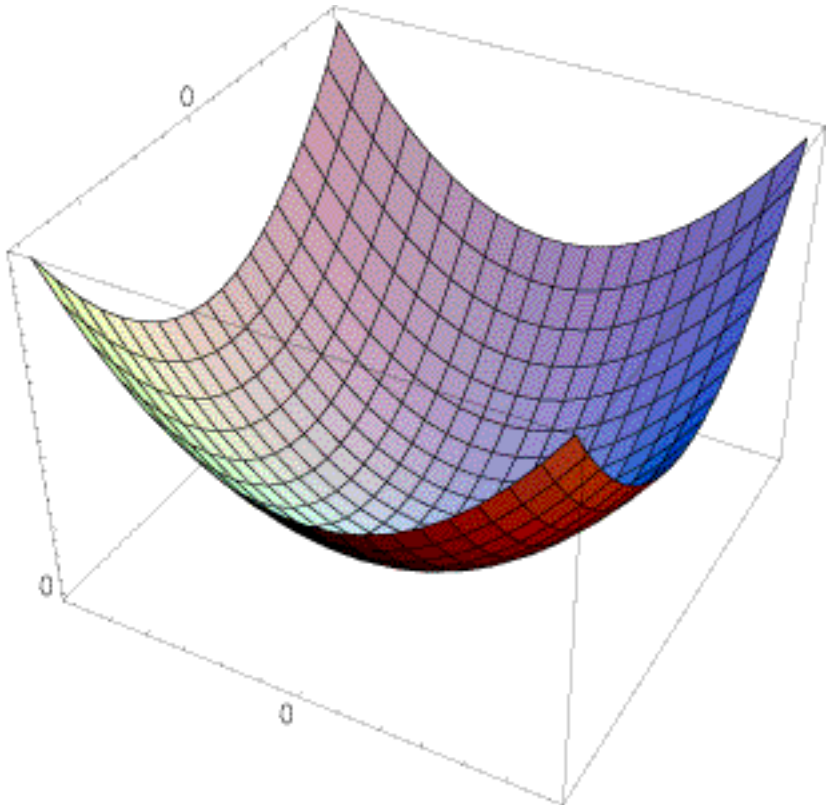
$$I_x \Leftrightarrow \frac{\partial I}{\partial x} \qquad I_y \Leftrightarrow \frac{\partial I}{\partial y} \qquad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}$$

Interpreting the second moment matrix

The surface $E(u,v)$ is locally approximated by a quadratic form. Let's try to understand its shape.

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

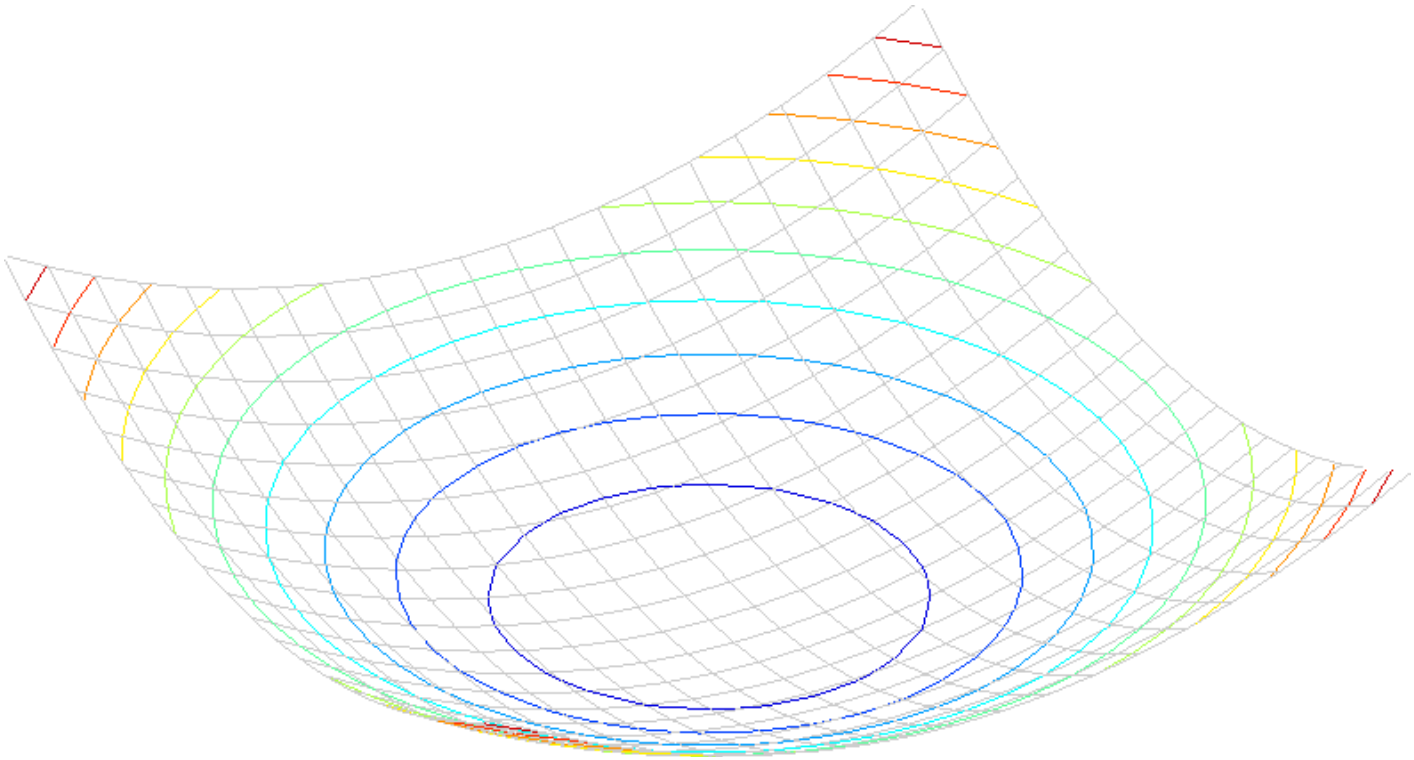
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$



Interpreting the second moment matrix

Consider a horizontal "slice" of $E(u, v)$: $\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.



Interpreting the second moment matrix

First, consider the axis-aligned case
(gradients are either horizontal or vertical)

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

If either λ is close to 0, then this is **not** a corner, so look for locations where both are large.

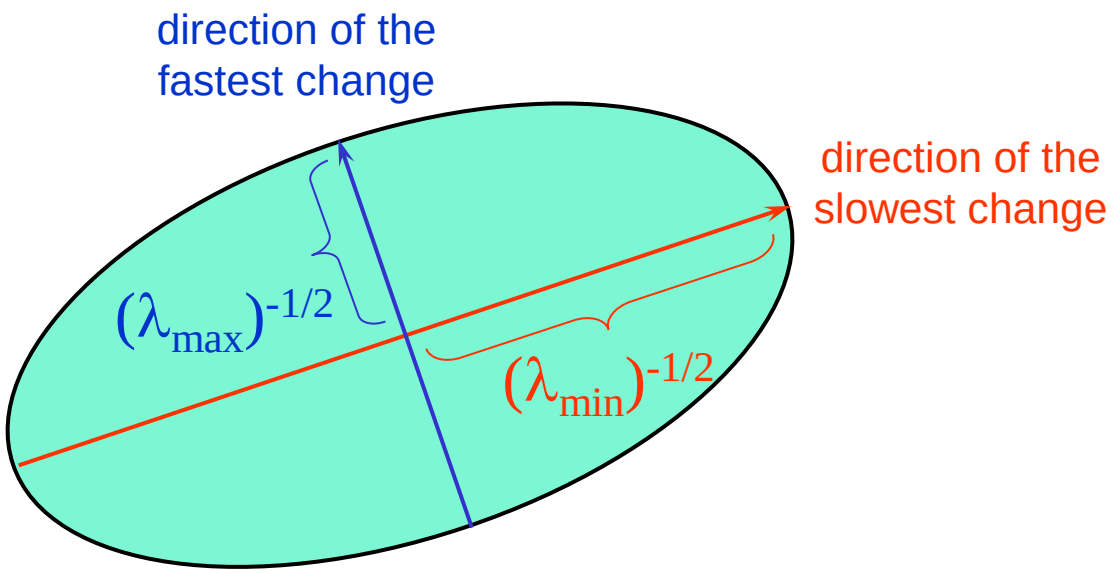
Interpreting the second moment matrix

Consider a horizontal "slice" of $E(u, v)$: $\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.

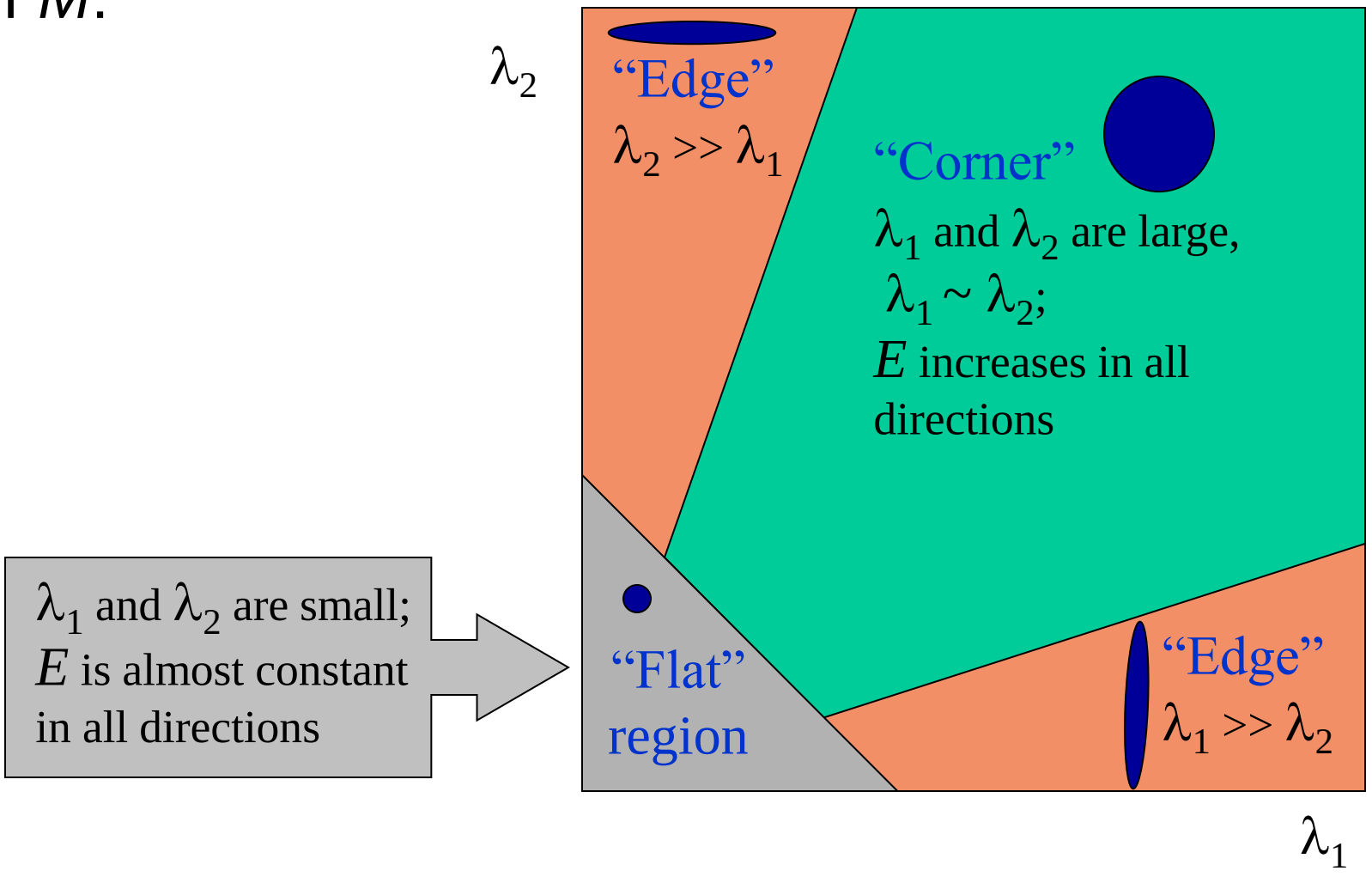
Diagonalization of M: $M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by R



Interpreting the eigenvalues

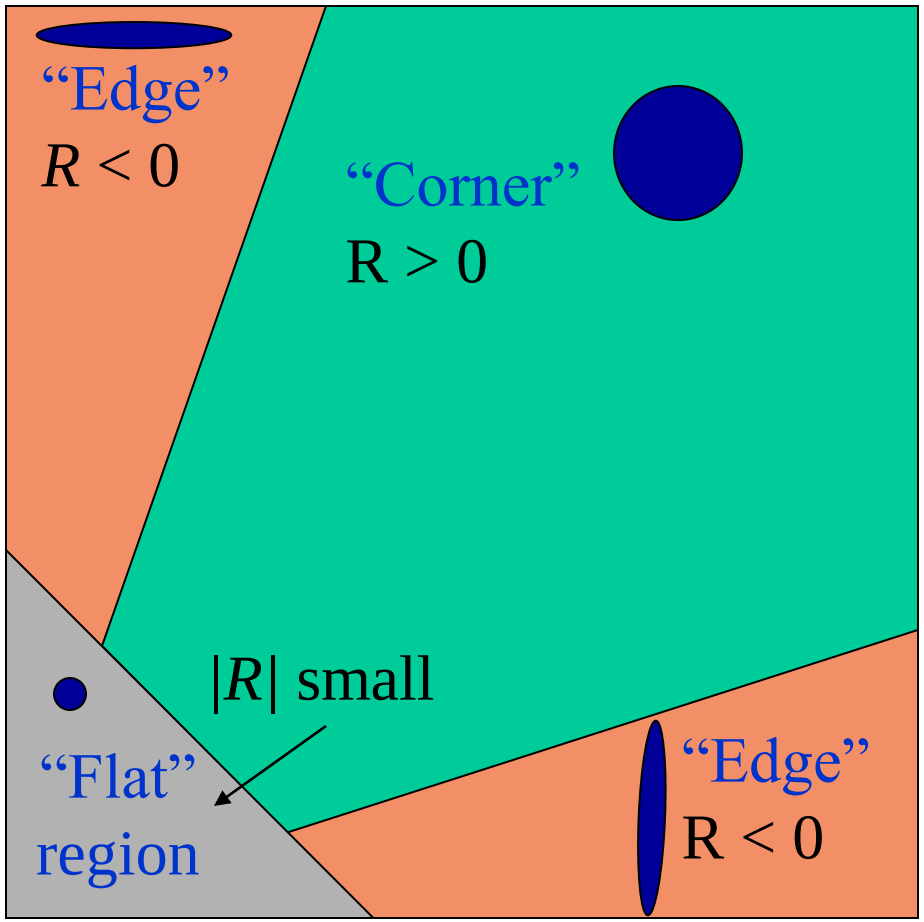
Classification of image points using eigenvalues of *M*:



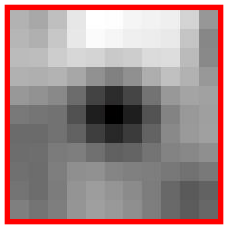
Corner response function

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

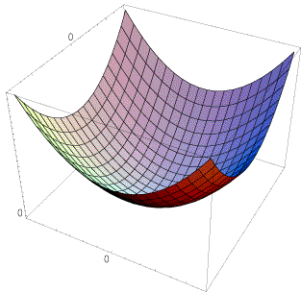
α: constant (0.04 to 0.06)

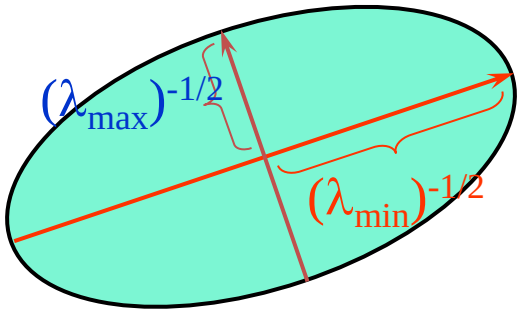


Review: Harris corner detector



- Approximate distinctiveness by local auto-correlation.
- Approximate local auto-correlation by second moment matrix
- Quantify distinctiveness (or cornerness) as function of the eigenvalues of the second moment matrix.
- But we don't actually need to compute the eigenvalues by using the determinant and trace of the second moment matrix.

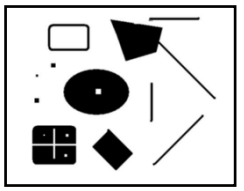




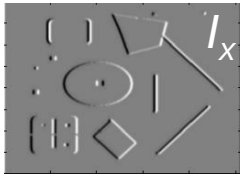
Harris Detector [Harris88]

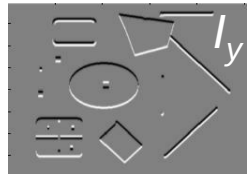
- Second moment matrix

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

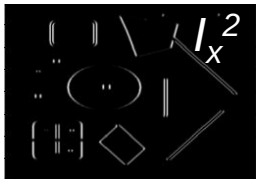


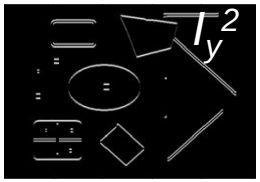
1. Image derivatives (optionally, blur first)

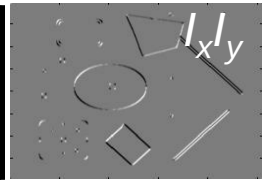




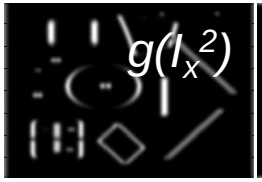
2. Square of derivatives

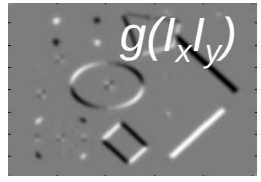






3. Gaussian filter $g(\sigma_I)$





$$\det M = \lambda_1 \lambda_2$$
$$\text{trace}\, M = \lambda_1 + \lambda_2$$

4. Cornerness function – both eigenvalues are strong

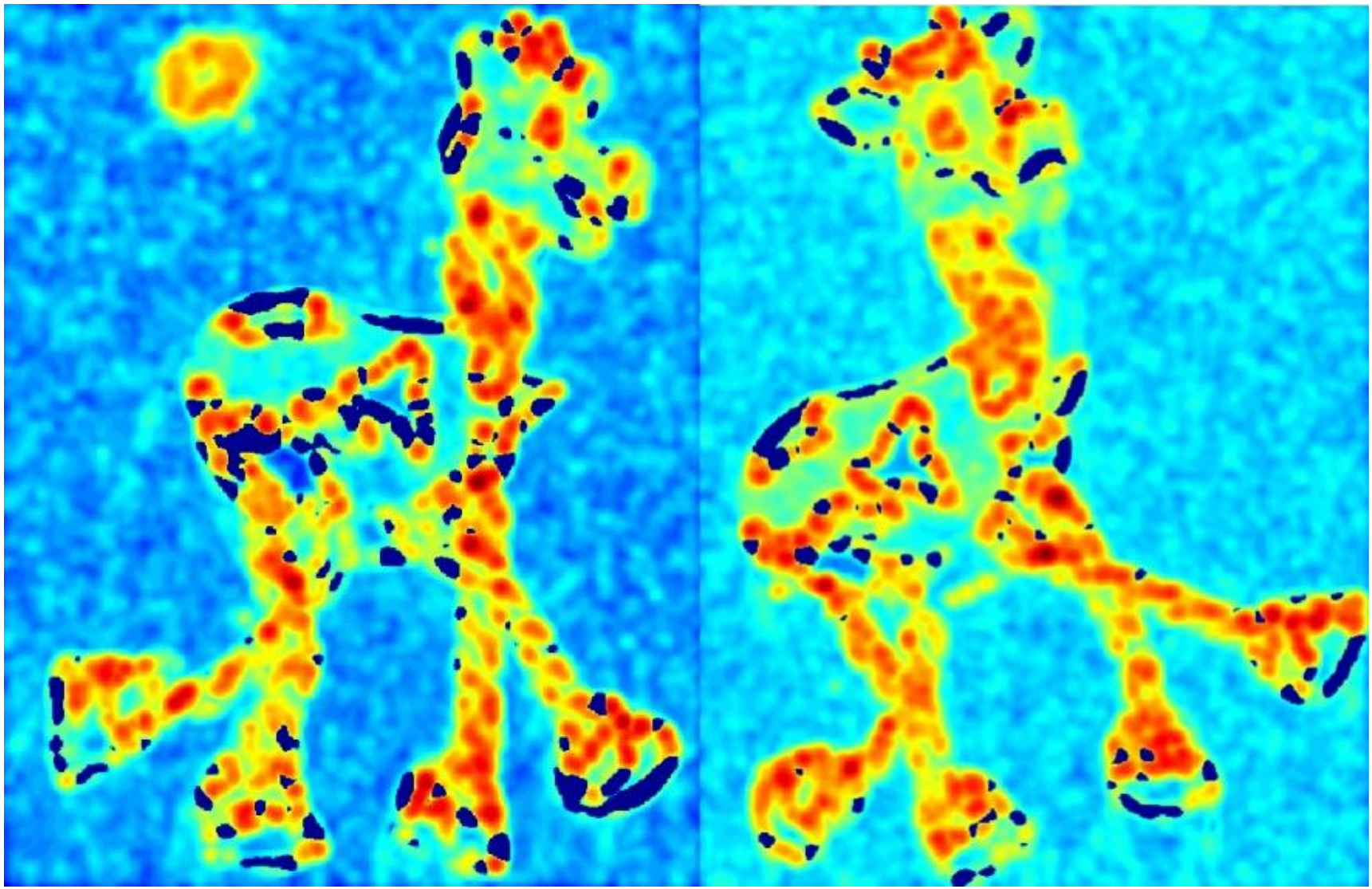
$$har = \det[\mu(\sigma_I, \sigma_D)] - \alpha[\text{trace}(\mu(\sigma_I, \sigma_D))^2] =$$
$$g(I_x^2)g(I_y^2) - [g(I_x I_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2$$

5. Non-maxima suppression

Harris Detector: Steps

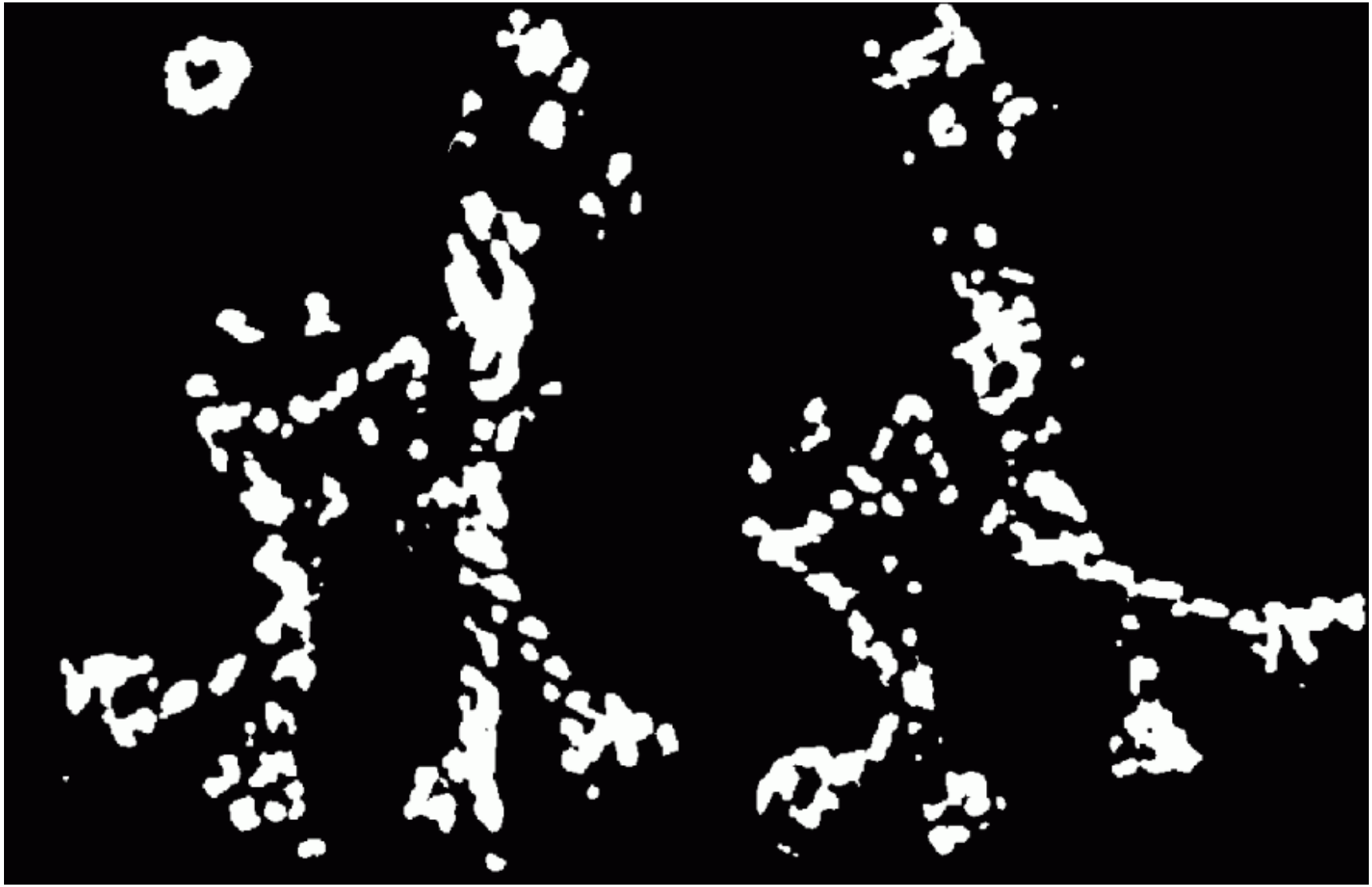
Harris Detector: Steps

Compute corner response R



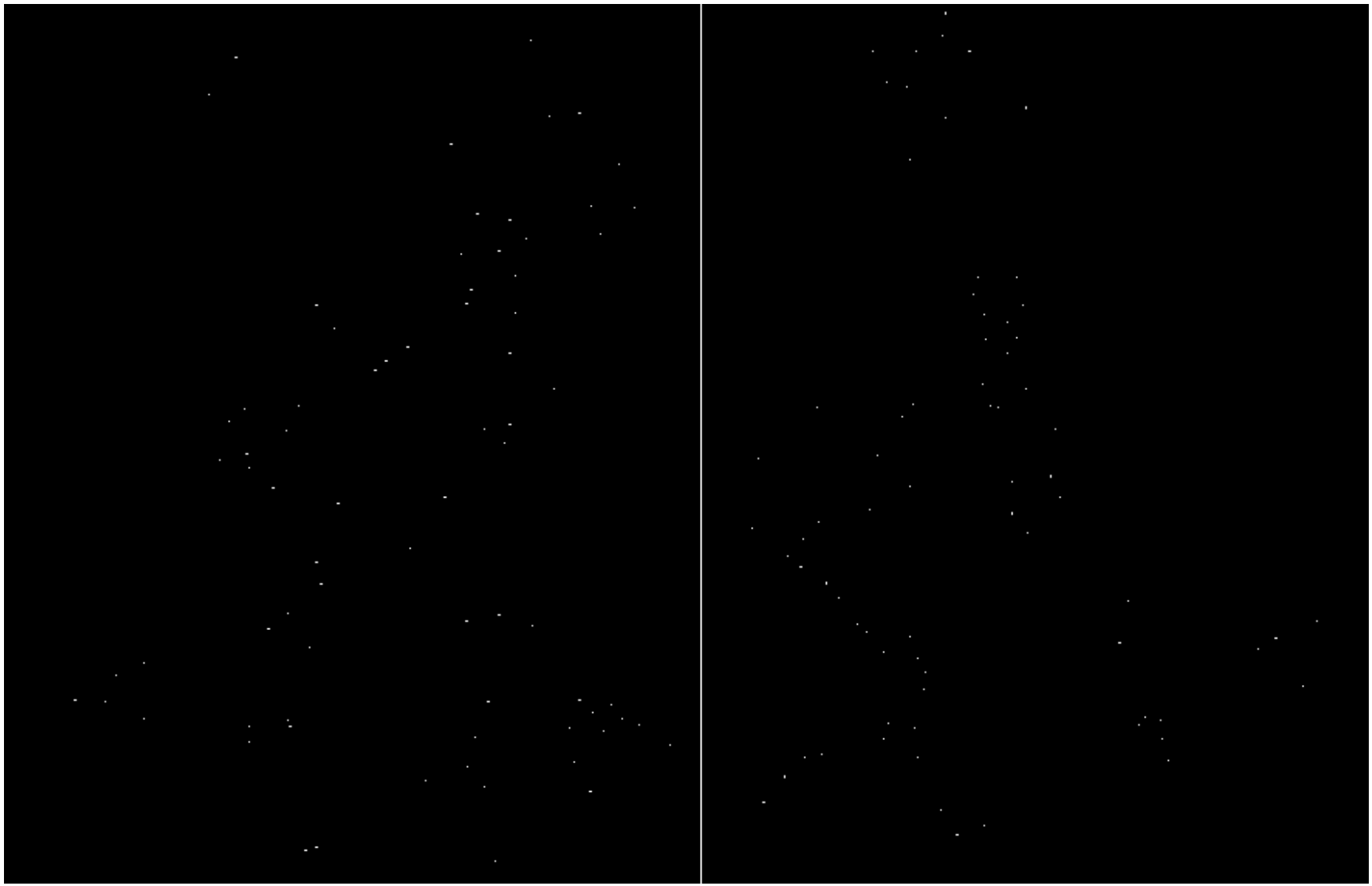
Harris Detector: Steps

Find points with large corner response: R>threshold



Harris Detector: Steps

Take only the points of local maxima of R

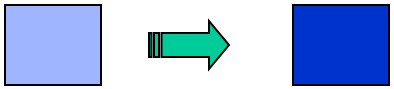


Harris Detector: Steps

Invariance and covariance

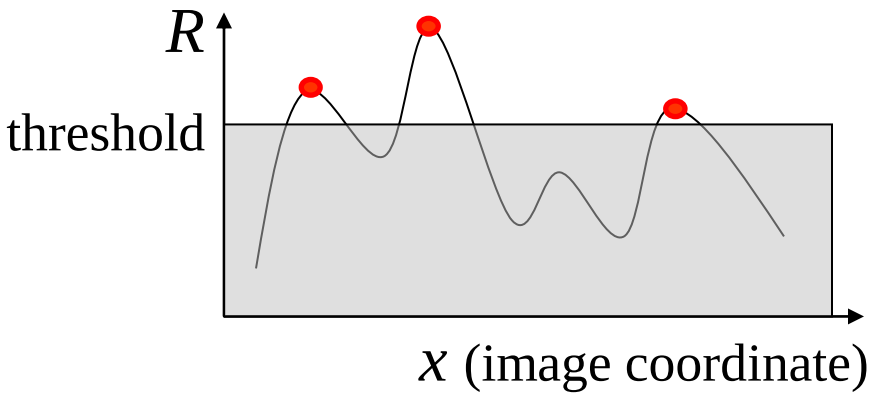
- We want corner locations to be *invariant* to photometric transformations and *covariant* to geometric transformations
  - **Invariance:** image is transformed and corner locations do not change
  - **Covariance:** if we have two transformed versions of the same image, features should be detected in corresponding locations

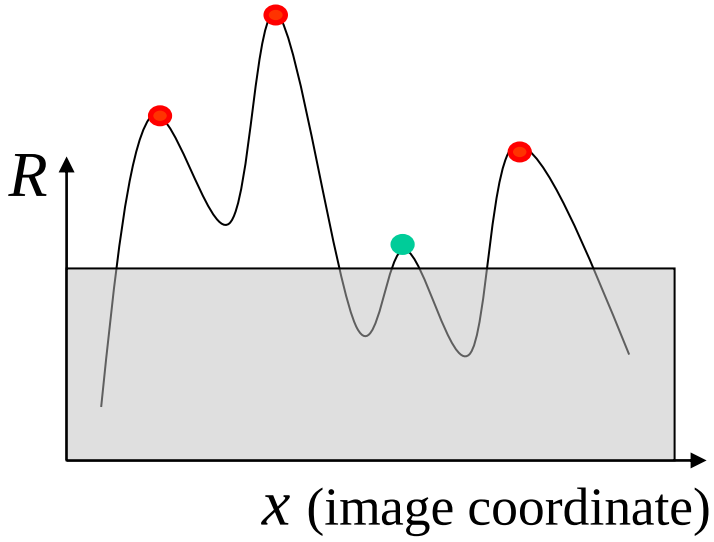
Affine intensity change



$$I \rightarrow a\,I + b$$

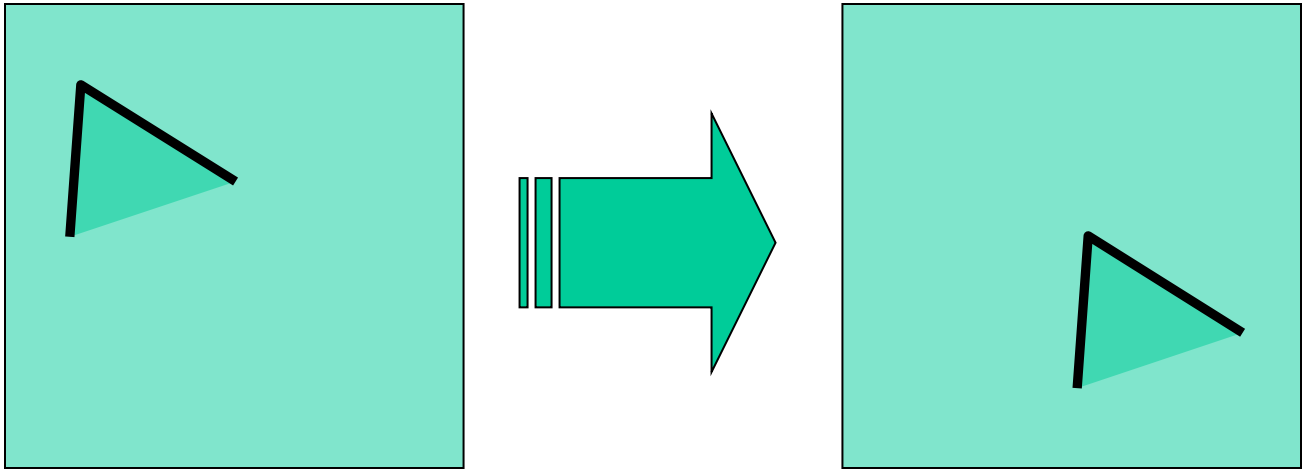
- Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$
- Intensity scaling: $I \rightarrow a\,I$





Partially invariant to affine intensity change

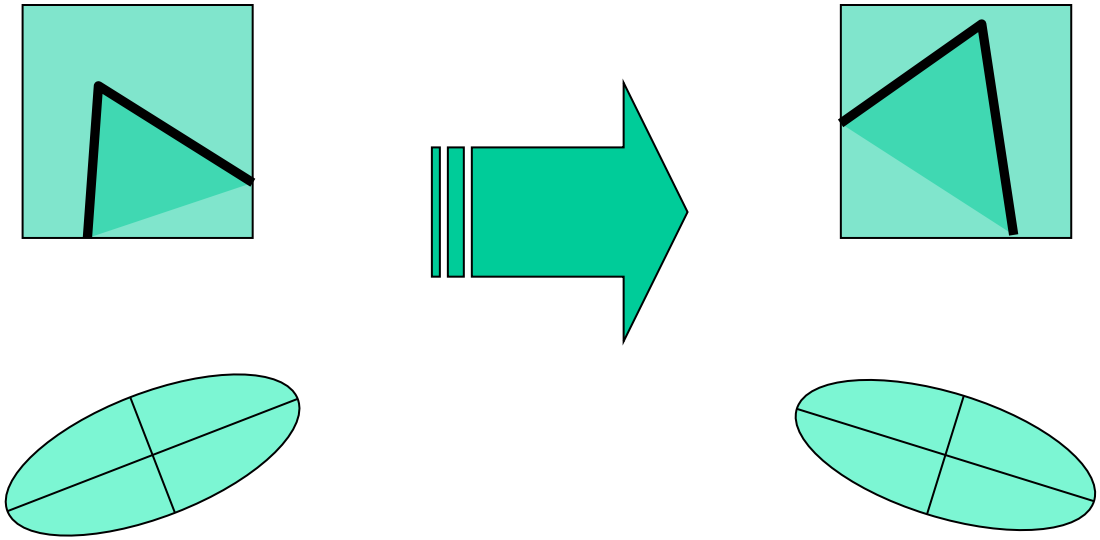
Image translation



- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation

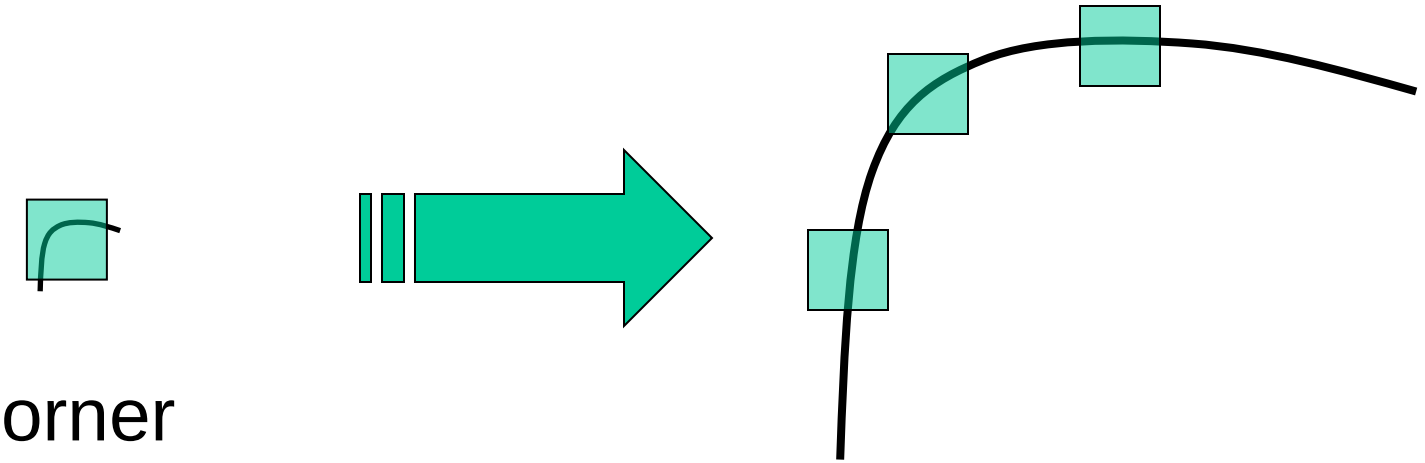
Image rotation



Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation

Scaling



Corner

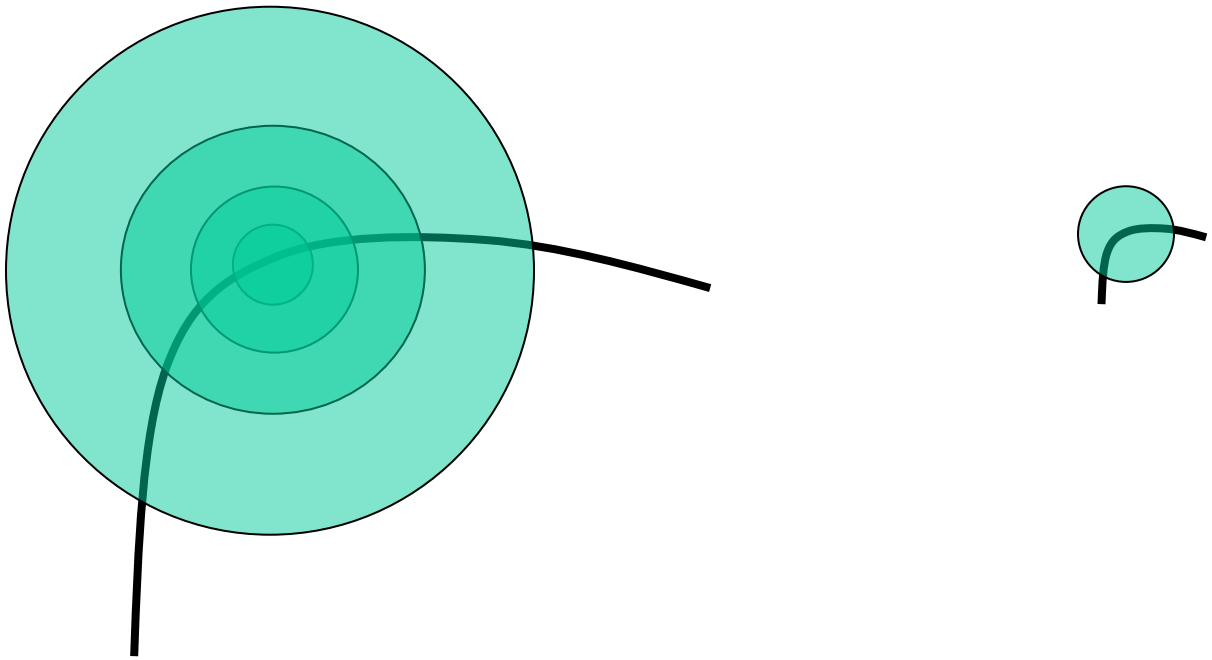
All points will be classified as edges

Corner location is not covariant to scaling!

Scale Invariant Detection

Consider regions (e.g. circles) of different sizes around a point

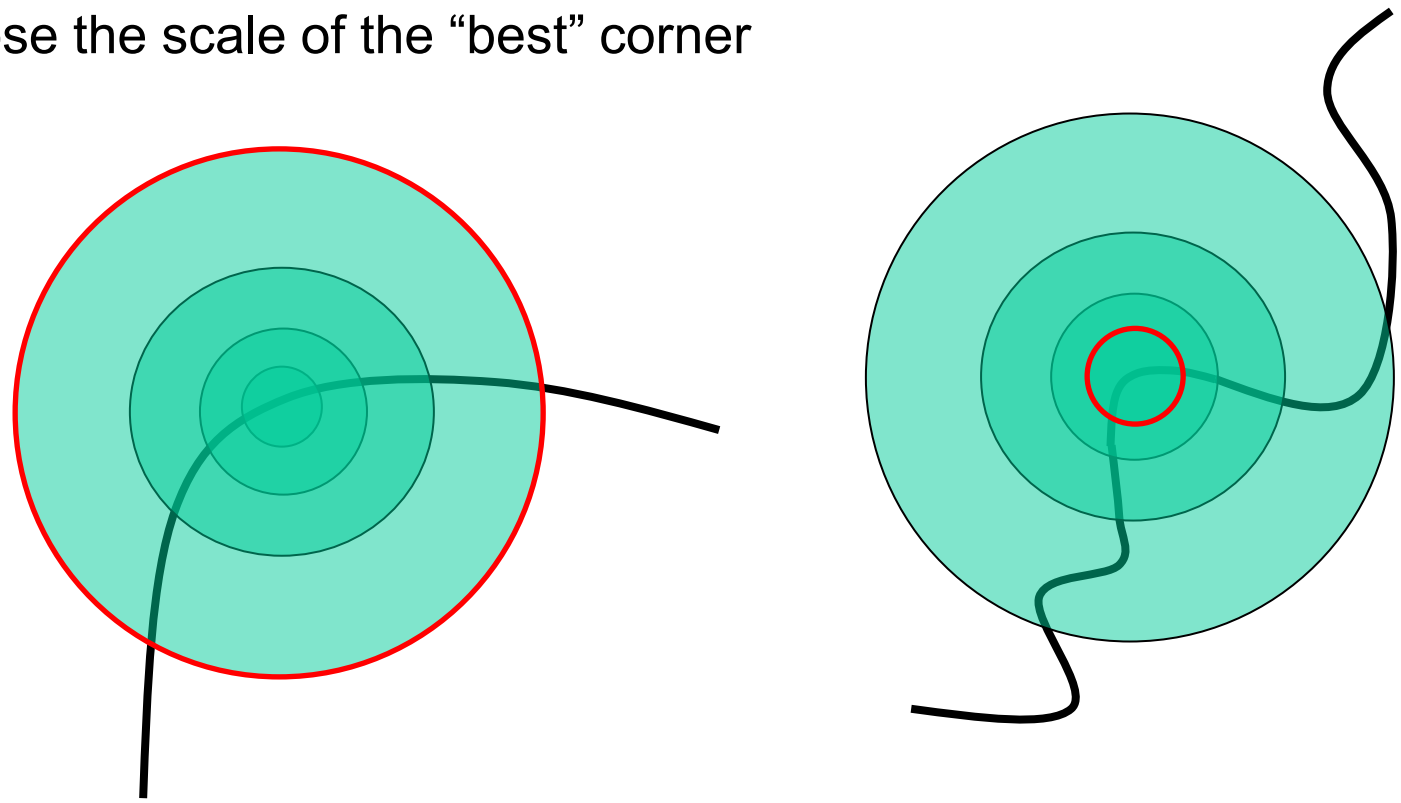
Regions of corresponding sizes will look the same in both images



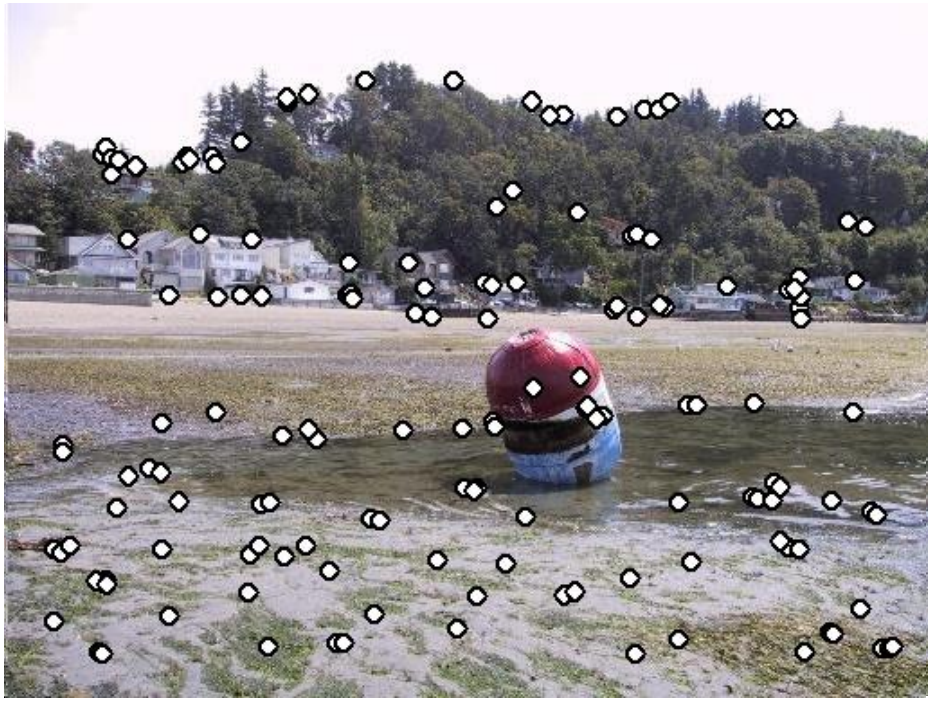
Scale Invariant Detection

The problem: how do we choose corresponding circles
independently in each image?

Choose the scale of the “best” corner



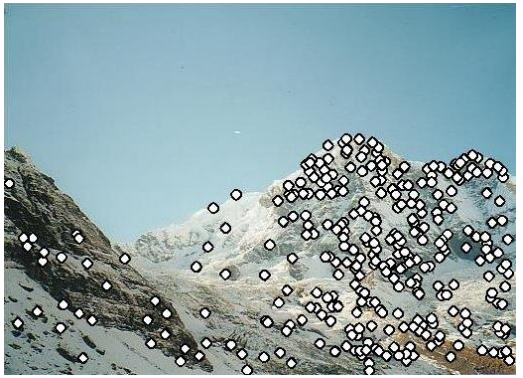
Pick how many corners you want

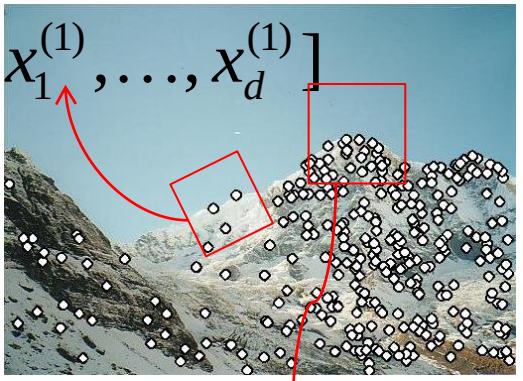


Local features: main components

1) Detection: Identify the interest points

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



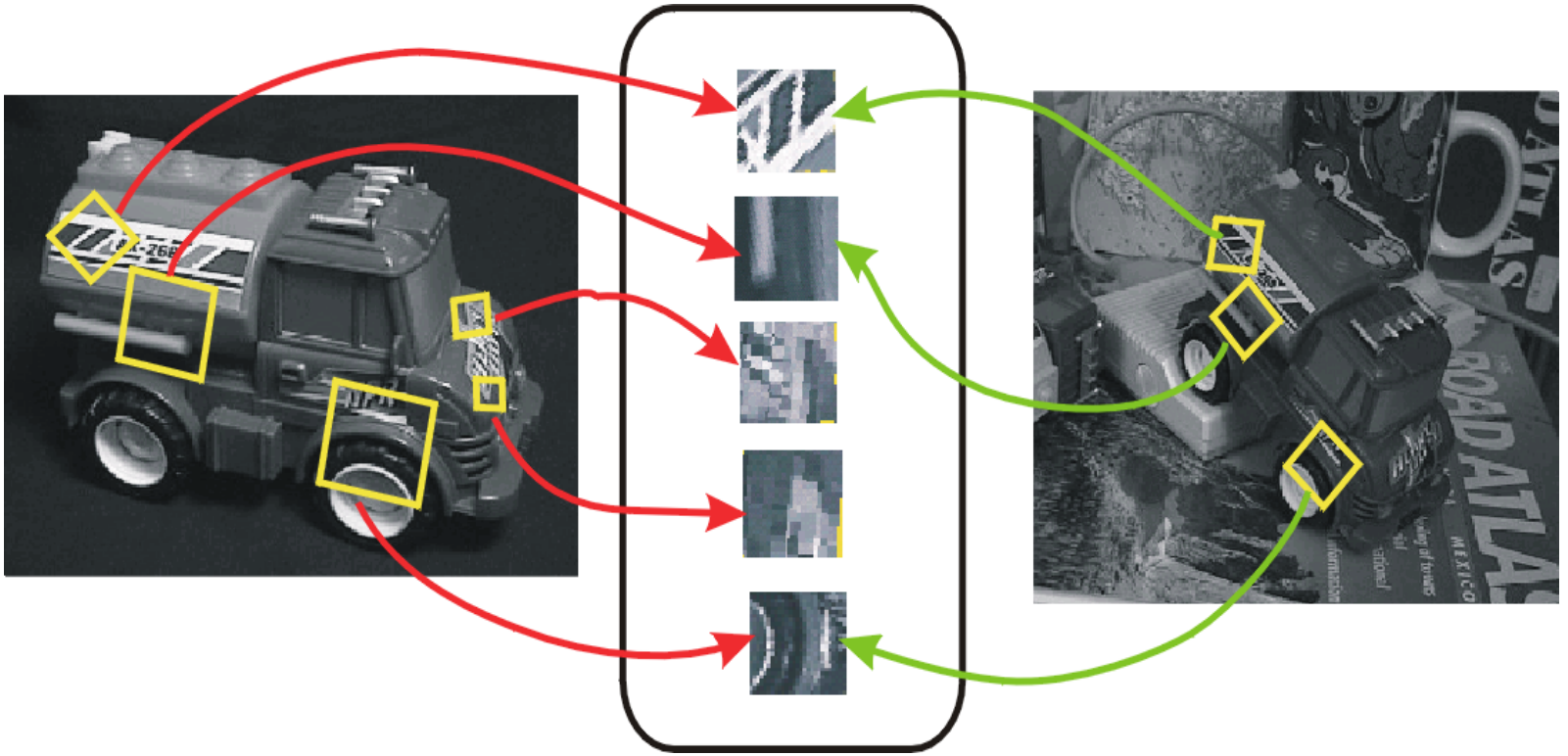


$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$

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

Invariant Local Features

•Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters

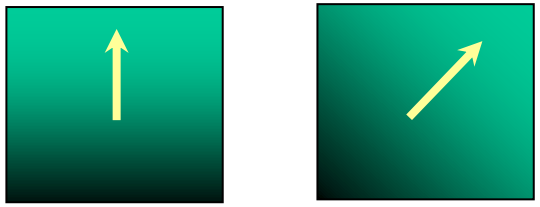


Features Descriptors

Descriptors Invariant to Rotation

Find local orientation

Dominant direction of gradient



- Extract image patches relative to this orientation

Descriptor Vector

Rotation Invariant Frame

- Scale-space position (x, y, s) + orientation (θ)

Detections at multiple scales

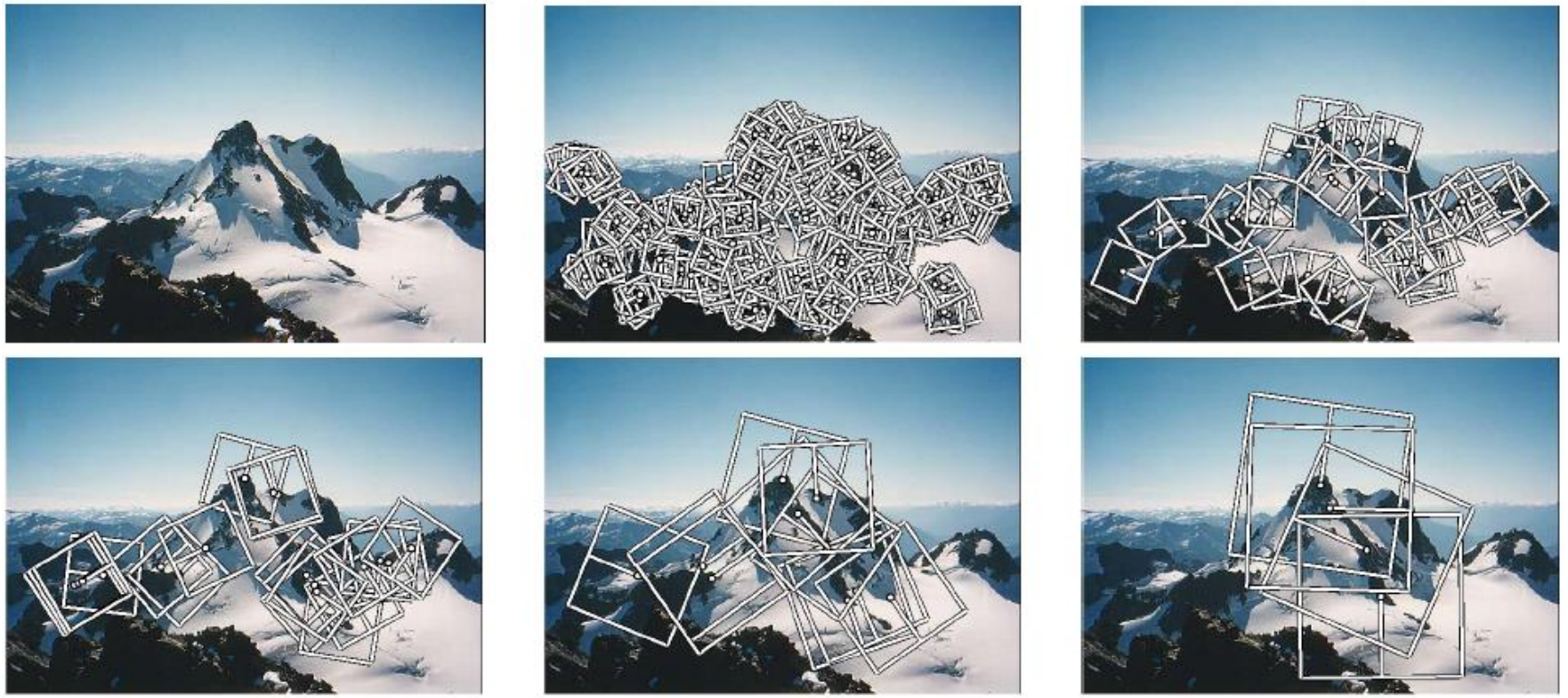


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

Disadvantage of intensity vectors as descriptors

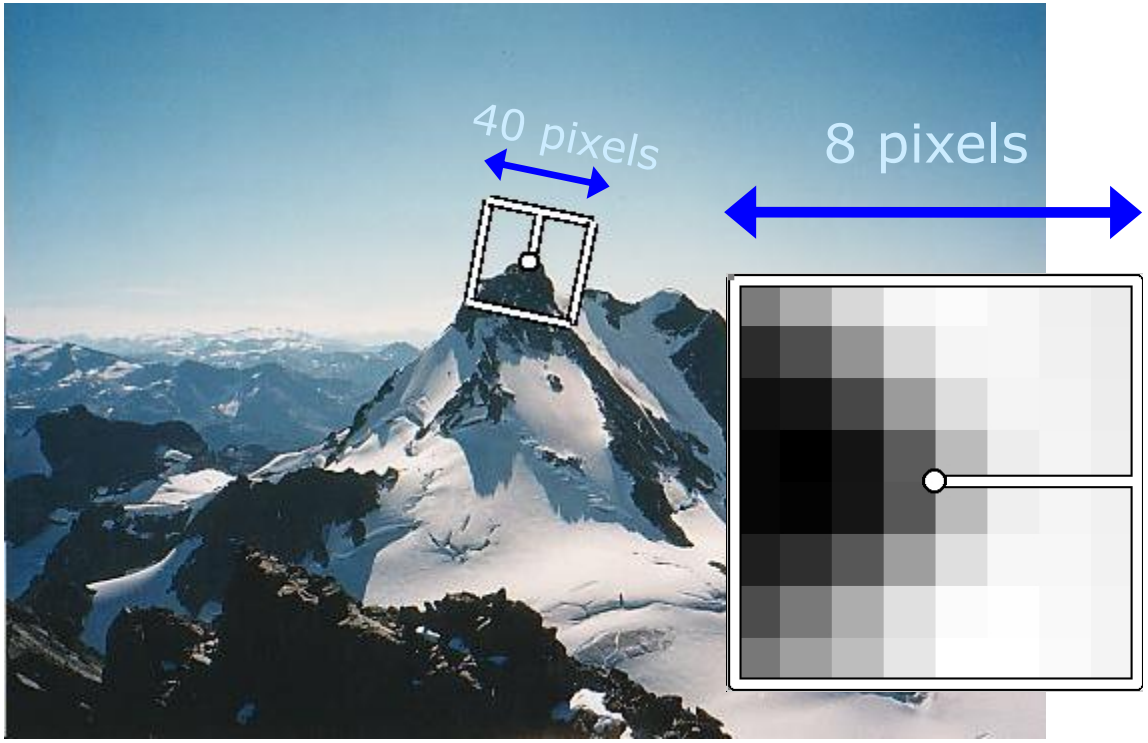
- Small deformations can affect the matching score a lot

MOPS descriptor vector

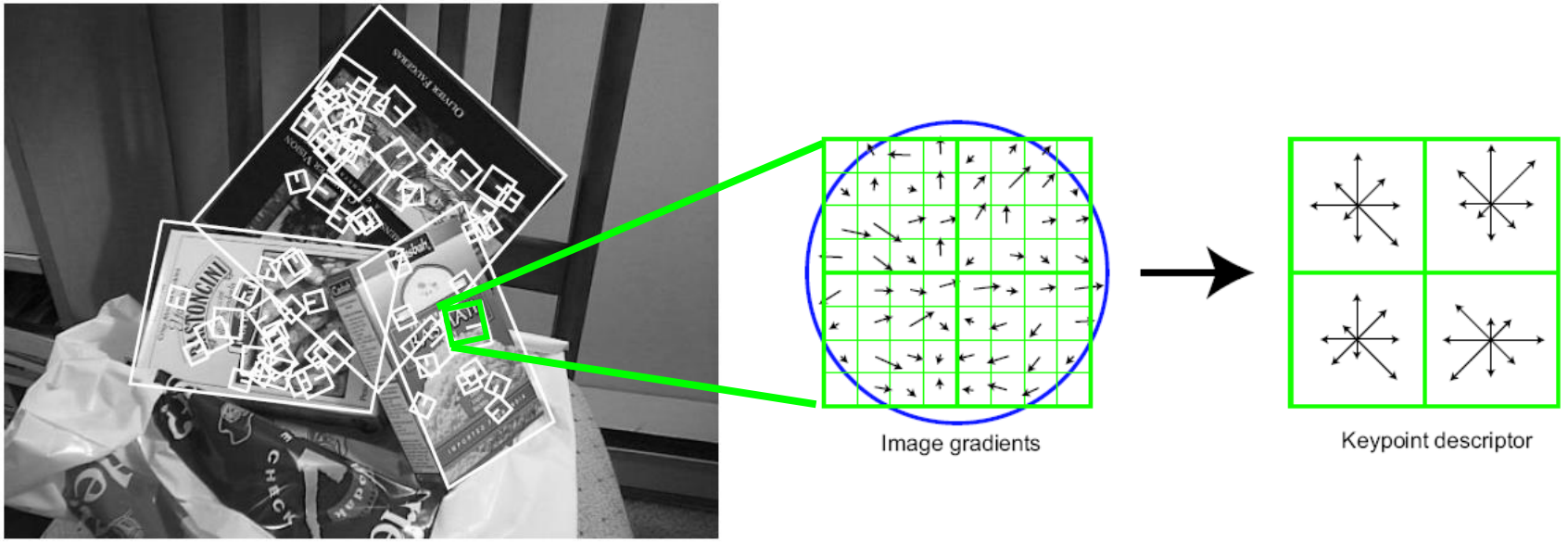
8x8 oriented patch

- Sampled at 5 x scale

Bias/gain normalisation: $I' = (I - \mu)/\sigma$



Local Descriptors: SIFT Descriptor



Histogram of oriented gradients

- **Captures important texture information**
- **Robust to small translations / affine deformations**

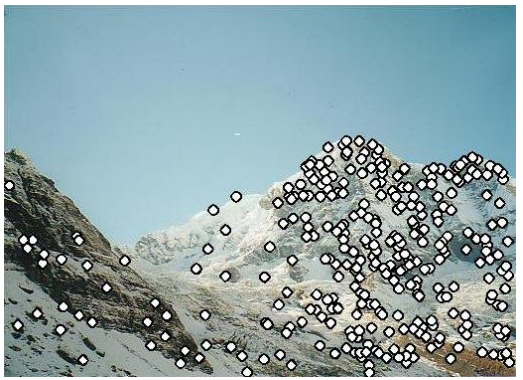
[Lowe, ICCV 1999]

Invariant Feature Descriptors

Schmid & Mohr 1997, Lowe 1999, Baumberg 2000, Tuytelaars & Van Gool 2000, Mikolajczyk & Schmid 2001, Brown & Lowe 2002, Matas et. al. 2002, Schaffalitzky & Zisserman 2002

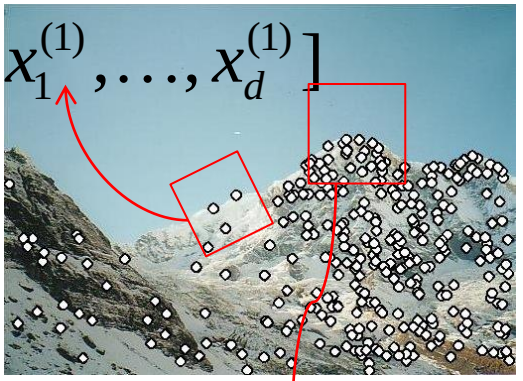
Local features: main components

1) Detection: Identify the interest points



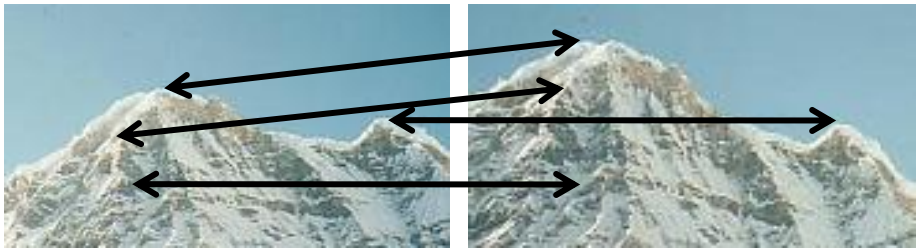
2) Description: Extract vector feature descriptor surrounding each interest point.

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

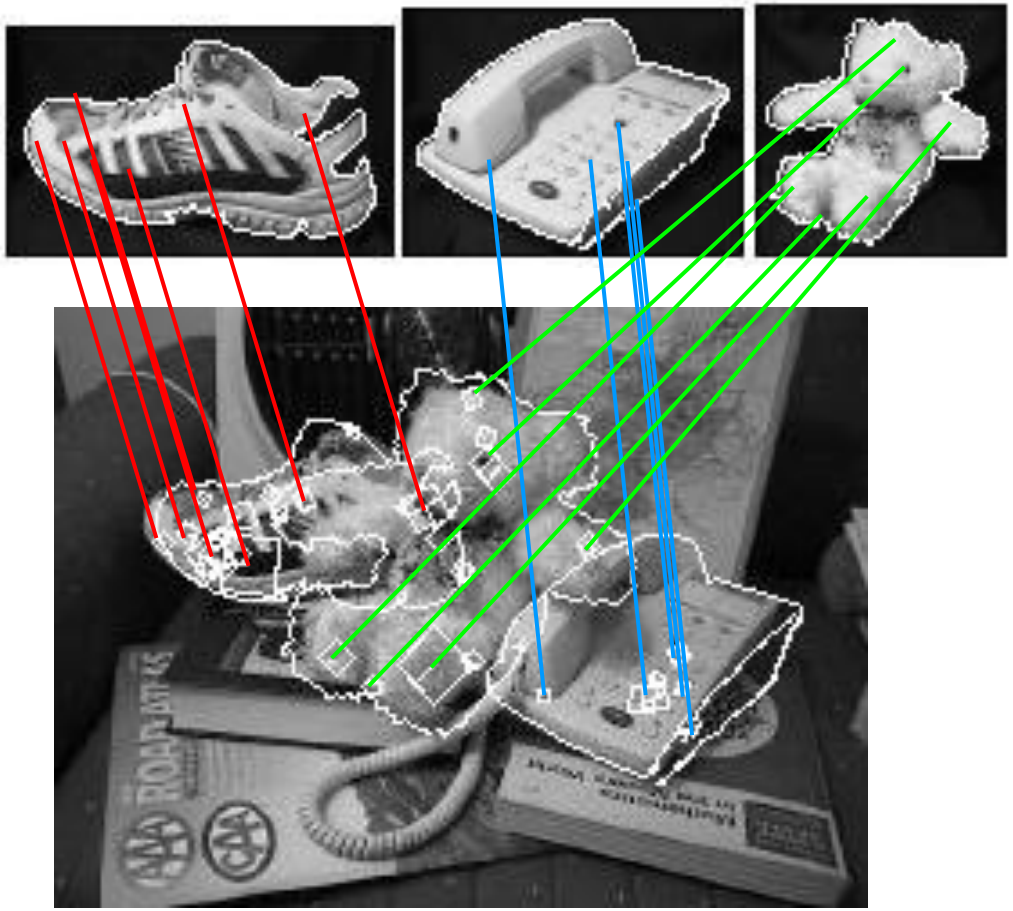


$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) Matching: Determine correspondence between descriptors in two views

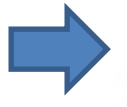


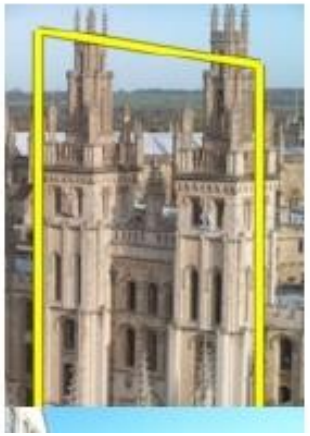
Large-scale Instance Retrieval



Slide Credits: Kristen Grauman, Josef Sivic, James Hays

How to quickly find images in a large database that match a given image region?

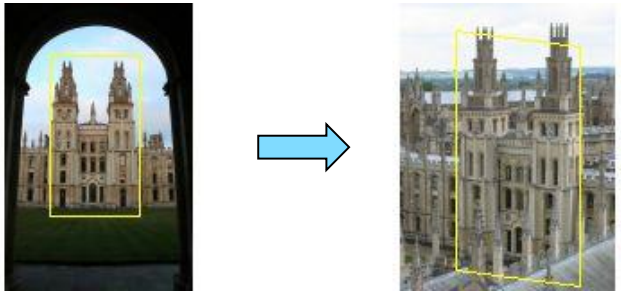




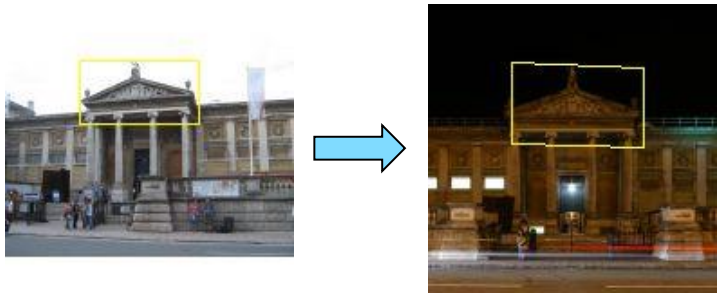
Why is it difficult?

Want to establish correspondence despite possibly large changes in scale, viewpoint, lighting and partial occlusion

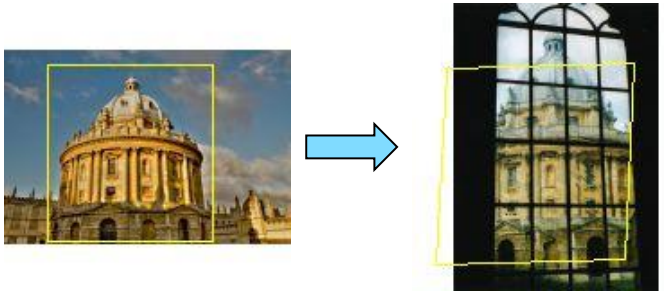
Scale



Viewpoint



Lighting



Occlusion

… and the image collection can be very large (e.g. 1M images)

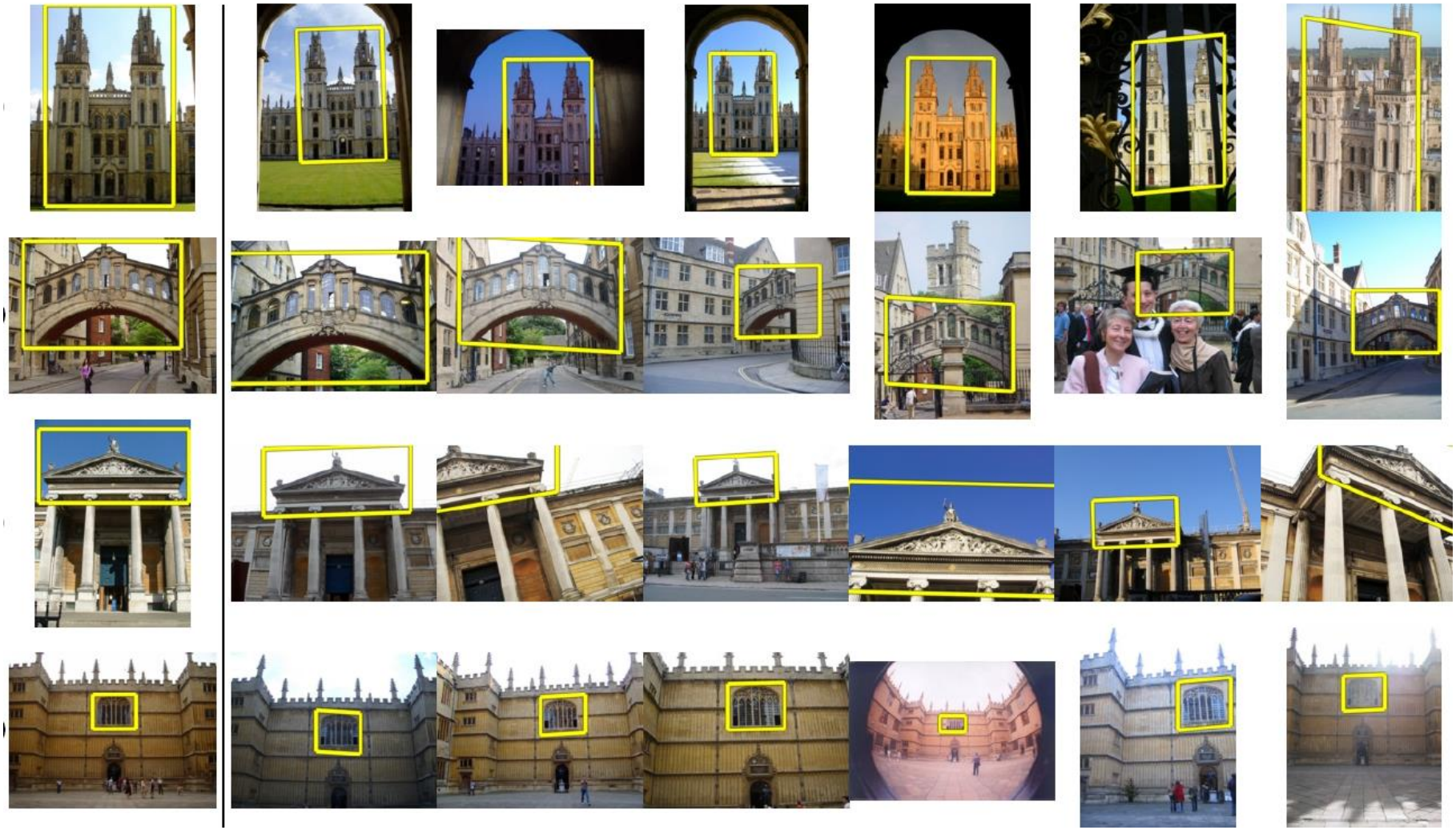
Example Applications

Mobile tourist guide

- Self-localization
- Object/building recognition
- Photo/video augmentation

[Quack, Leibe, Van Gool, CIVR'08]

Application: Large-Scale Retrieval

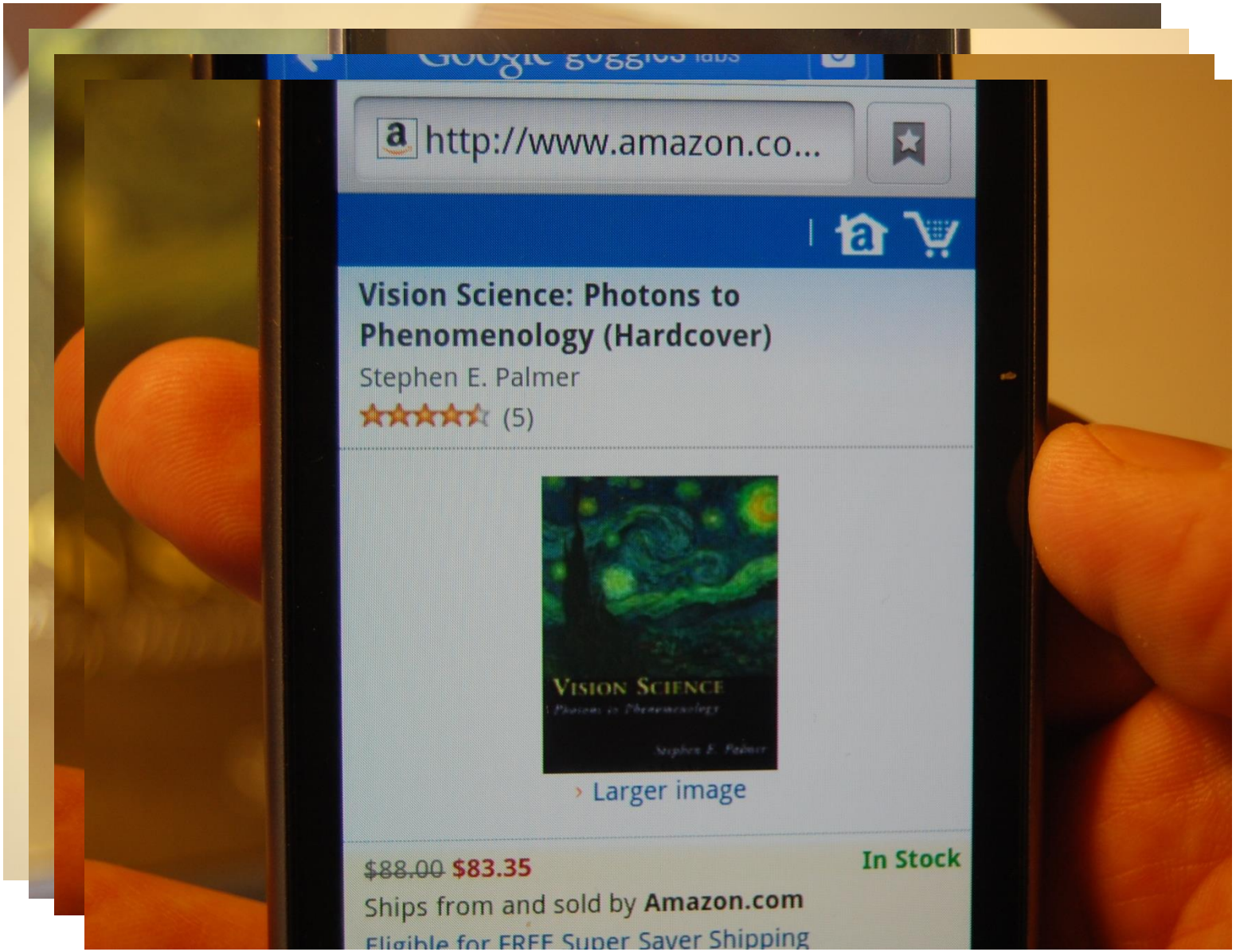


Query

Results from 5k Flickr images (demo available for 100k set)

[Philbin CVPR'07]

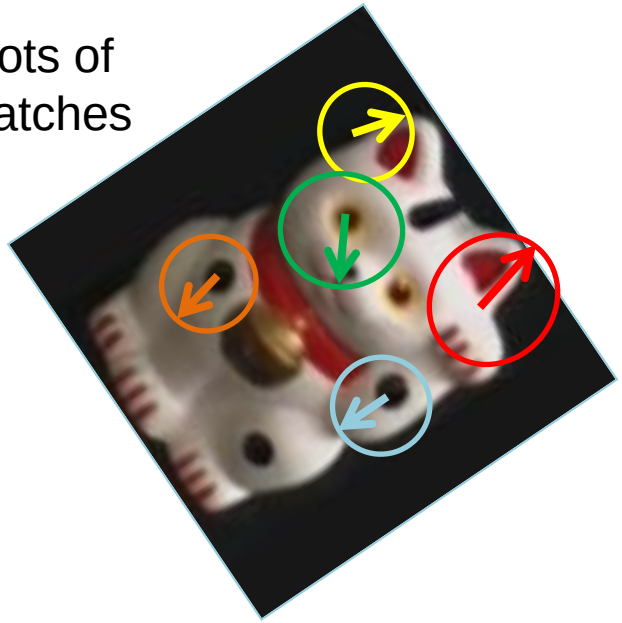
Example


http://www.amazon.co...
Vision Science: Photons to Phenomenology (Hardcover)
Stephen E. Palmer
(5)
VISION SCIENCE
› Larger image
$88.00 $83.35
In Stock
Ships from and sold by Amazon.com
Eligible for FREE Super Saver Shipping

Simple idea

See how many keypoints are close to keypoints in each other image

Lots of Matches

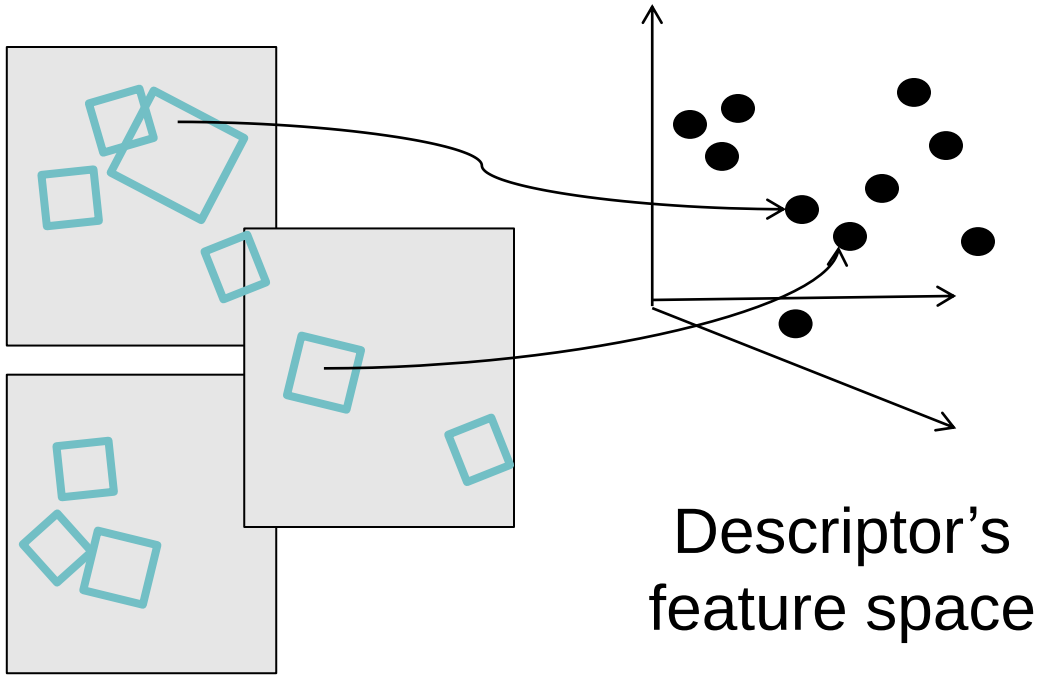


Few or No Matches

But this will be really, really slow!

Indexing local features

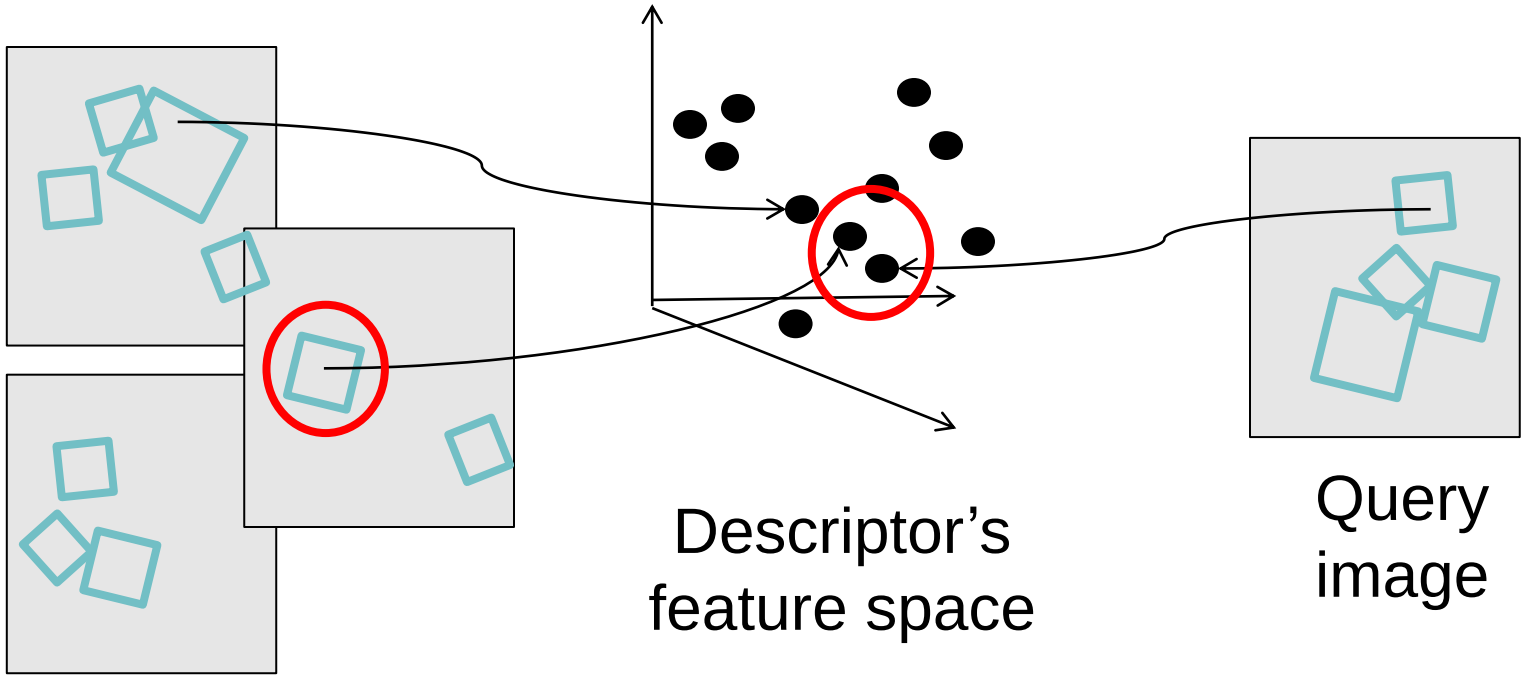
- Each patch / region has a descriptor, which is a point in some high-dimensional feature space (e.g., SIFT)



Kristen Grauman

Indexing local features

- When we see close points in feature space, we have similar descriptors, which indicates similar local content.



Easily can have millions of features to search!

Inverted file index

Image #1

Image #2

Image #3

Word #	Image #
1	3
2 …	
7	1, 2
8	3
9	
10 …	
91	2

…

- Database images are loaded into the index mapping words to image numbers

Kristen Grauman

Inverted file index

New query image

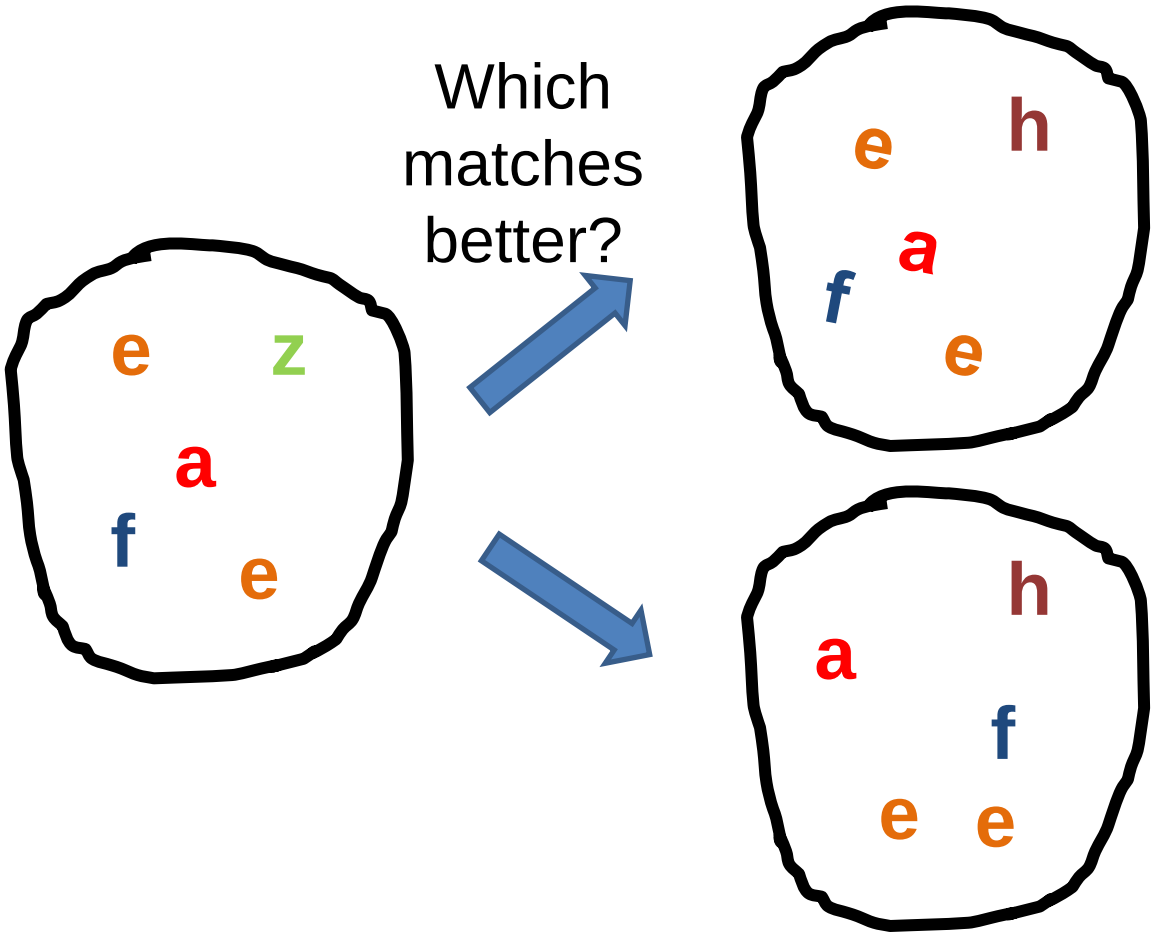
Word #	Image #
1	3
2	
…	
7	1, 2
8	3
9	
10	
…	
91	2

- New query image is mapped to indices of database images that share a word.

Kristen Grauman

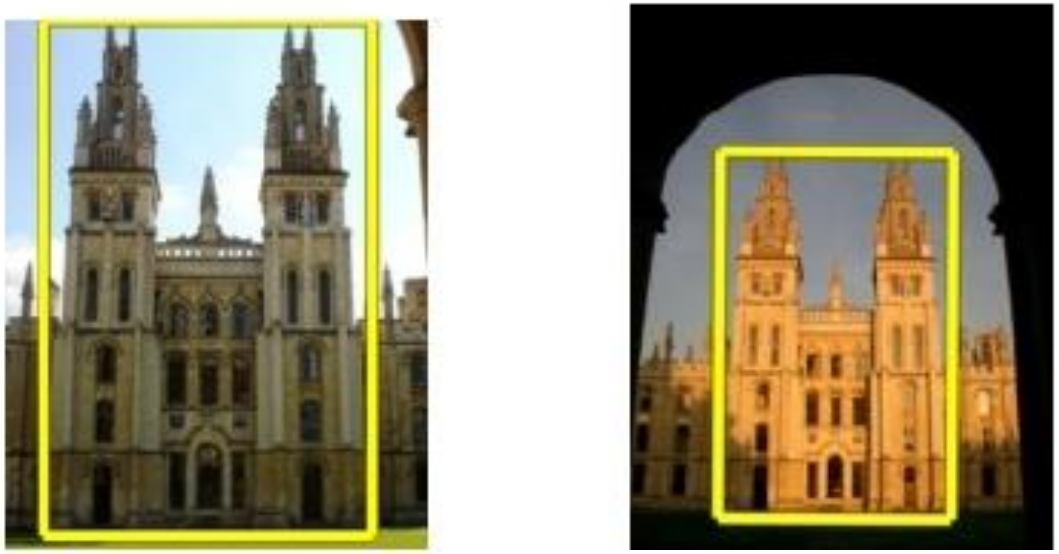
Can we be more accurate?

So far, we treat each image as containing a "bag of words", with no spatial information



Can we be more accurate?

So far, we treat each image as containing a "bag of words", with no spatial information



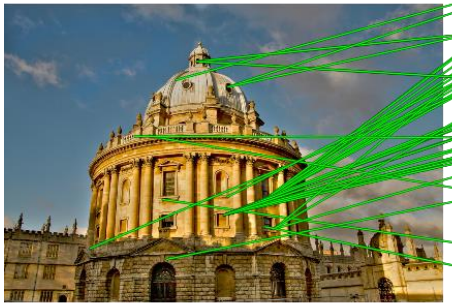
Real objects have consistent geometry

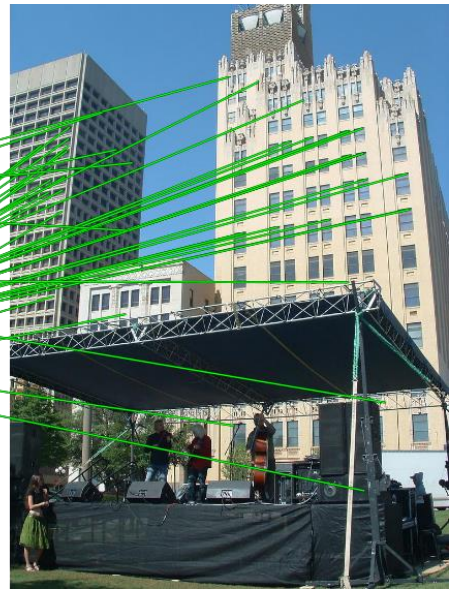
Spatial Verification

Query

DB image with high BoW similarity

Query





DB image with high BoW similarity

Both image pairs have many visual words in common.

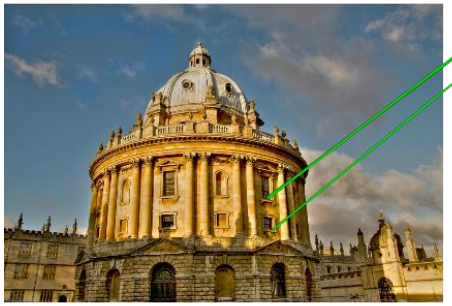
Slide credit: Ondrej Chum

Spatial Verification

Query

DB image with high BoW similarity

Query



DB image with high BoW similarity

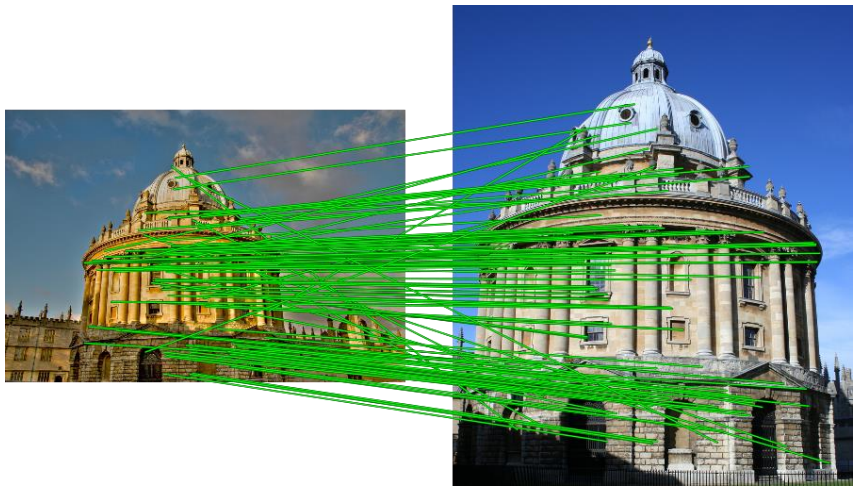
Only some of the matches are mutually consistent

Slide credit: Ondrej Chum

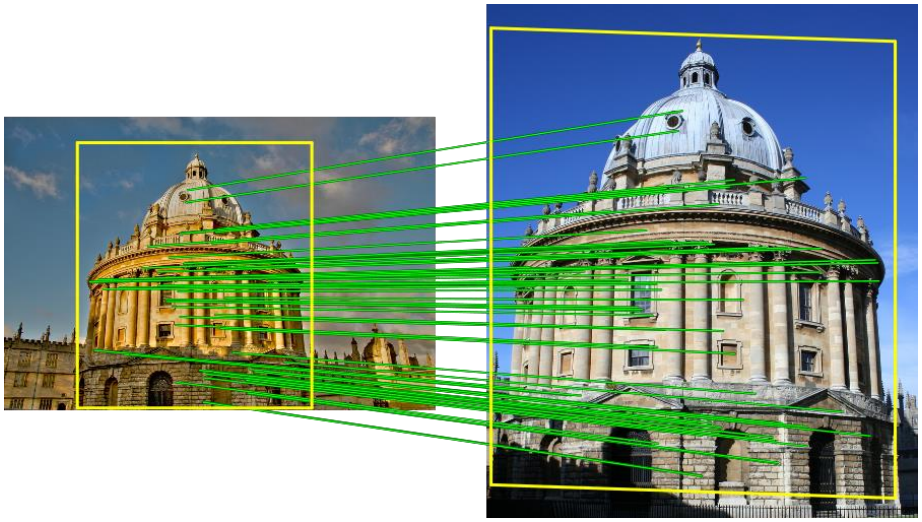
Geometric verification with global constraints

- All matches must be consistent with a global geometric relation / transformation.

- Need to simultaneously (i) estimate the geometric relation / transformation and (ii) the set of consistent matches



Tentative matches

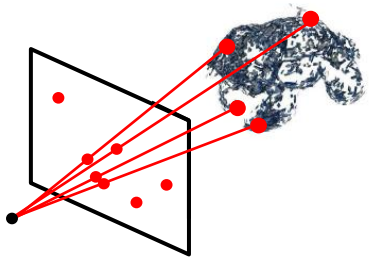


Matches consistent with an affine transformation

Examples of global constraints

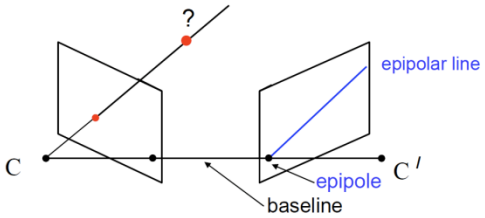
1 view and known 3D model.

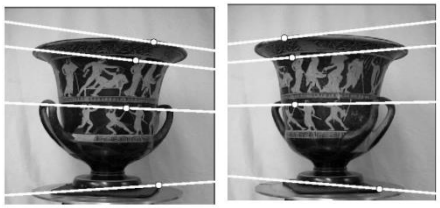
- Consistency with a (known) 3D model.

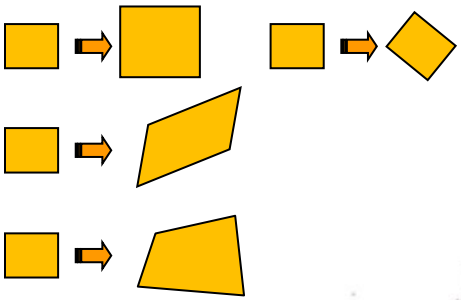


2 views

- Epipolar constraint
- **2D transformations**
  - Similarity transformation
  - Affine transformation
  - Projective transformation







N-views

Are images consistent with a 3D model?