Object recognition and Computer Vision 2017

Instance-level recognition I: Local invariant features, correspondence, image matching

Josef Sivic

http://www.di.ens.fr/~josef

INRIA, WILLOW, ENS/INRIA/CNRS UMR 8548

Laboratoire d'Informatique, Ecole Normale Supérieure, Paris

With slides from: O. Chum, K. Grauman, J. Hays, D. Hoiem**, S. Lazebnik**, B. Leibe, D. Lowe, J. Philbin, J. Ponce, D. Nister, C. Schmid, N. Snavely, A. Zisserman

Class logistics

- Please register for the class via form on the class web-site
    1. Name, 2. School/department and 3. Email

- Assignment 1 – Instance-level recognition

http://www.di.ens.fr/willow/teaching/recvis17/assignment1/

Due in 2 weeks: Oct 24 2017

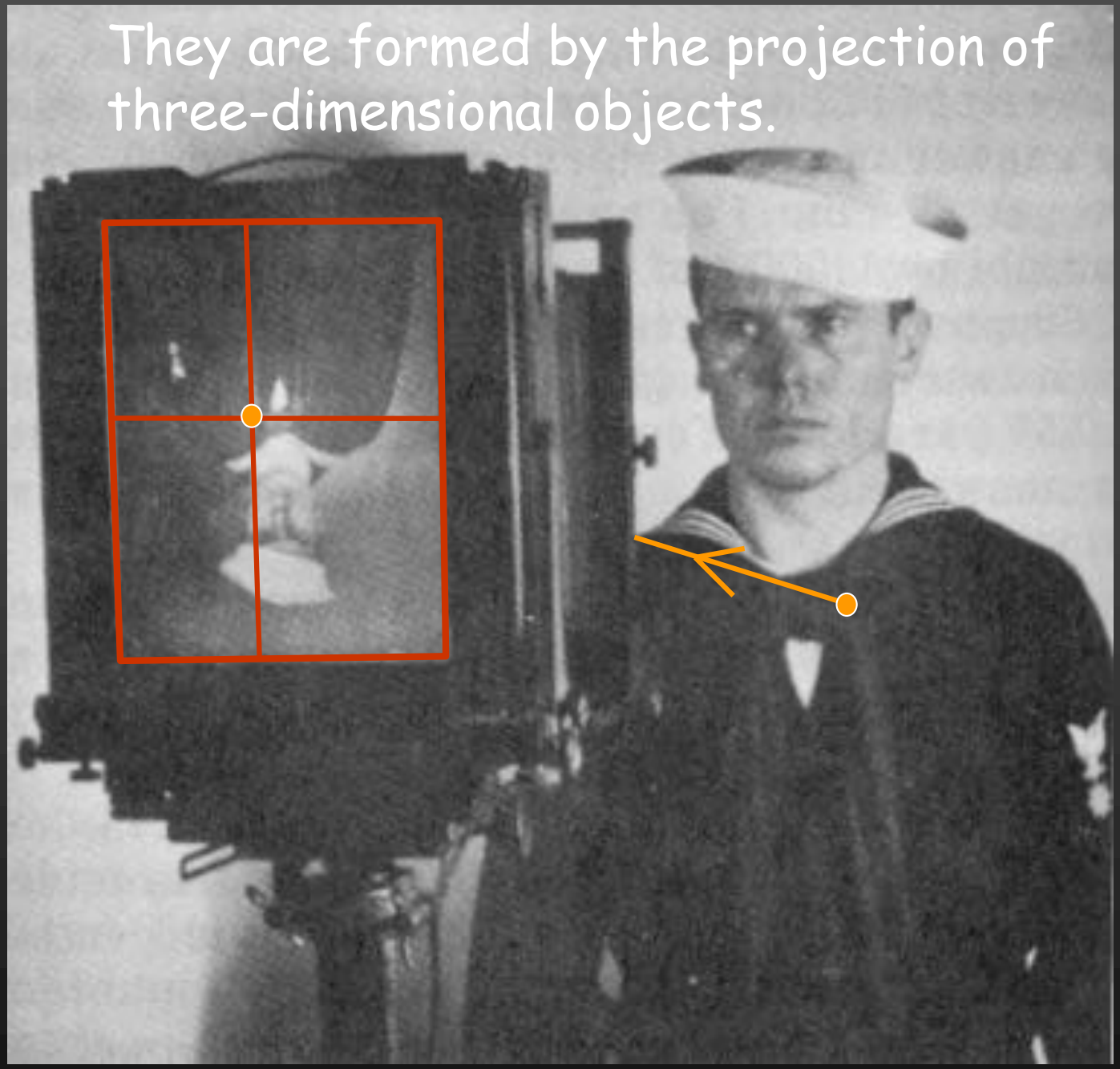
- Matlab tutorial (by TAs Gul Varol and Ignacio Rocco)

Fill-in Doodle on class webpage by Tue 10/10 6pm (today).

Dates will be announced by email on Wed 11/10.

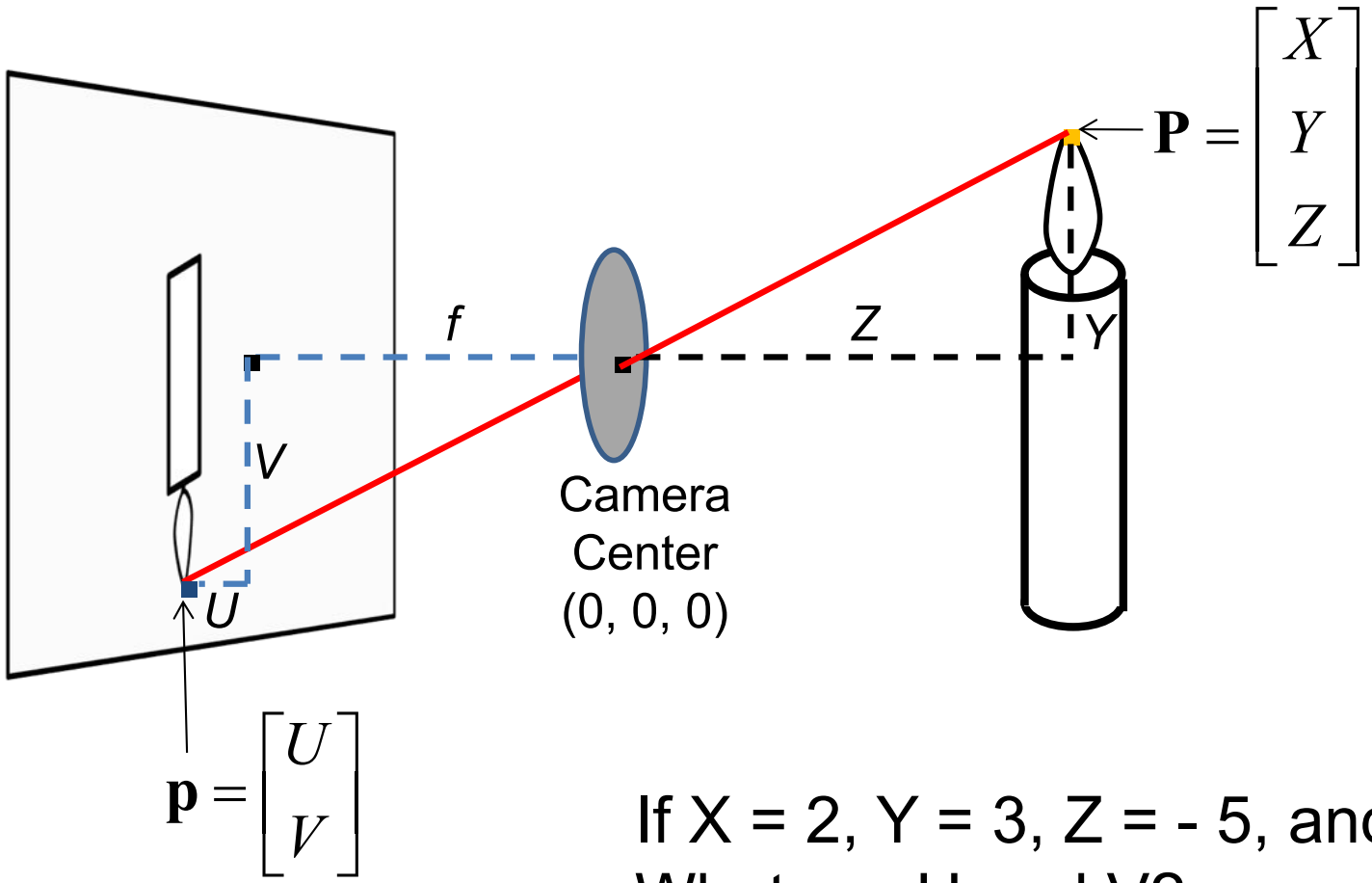
Recap of camera geometry (from last lecture)

They are formed by the projection of three-dimensional objects.



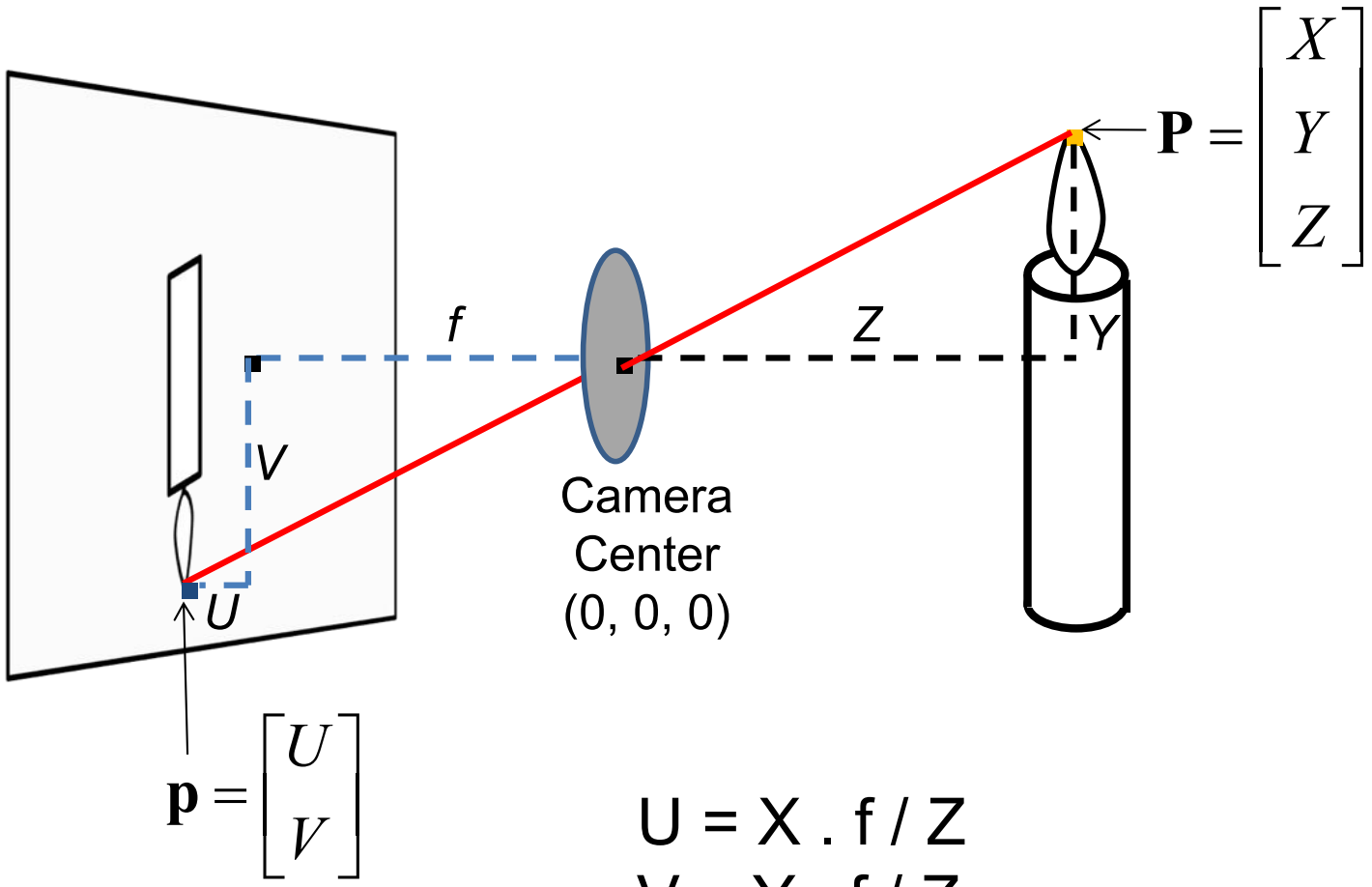
Images are brightness/color patterns drawn in a plane.

Projection: world coordinates→image coordinates



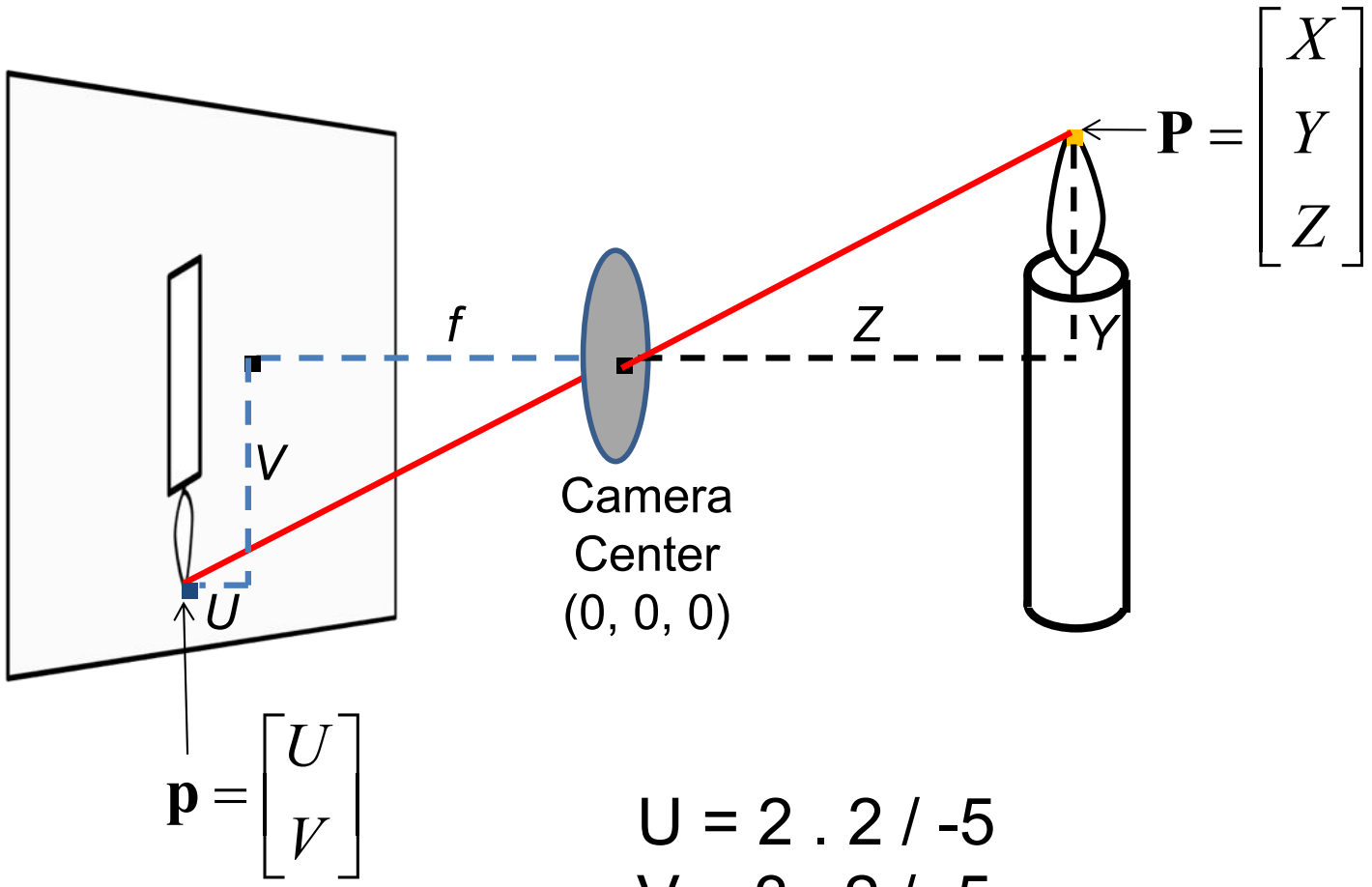
If X = 2, Y = 3, Z = - 5, and f = 2
What are U and V?

Projection: world coordinates→image coordinates



U = X . f / Z
V = Y . f / Z

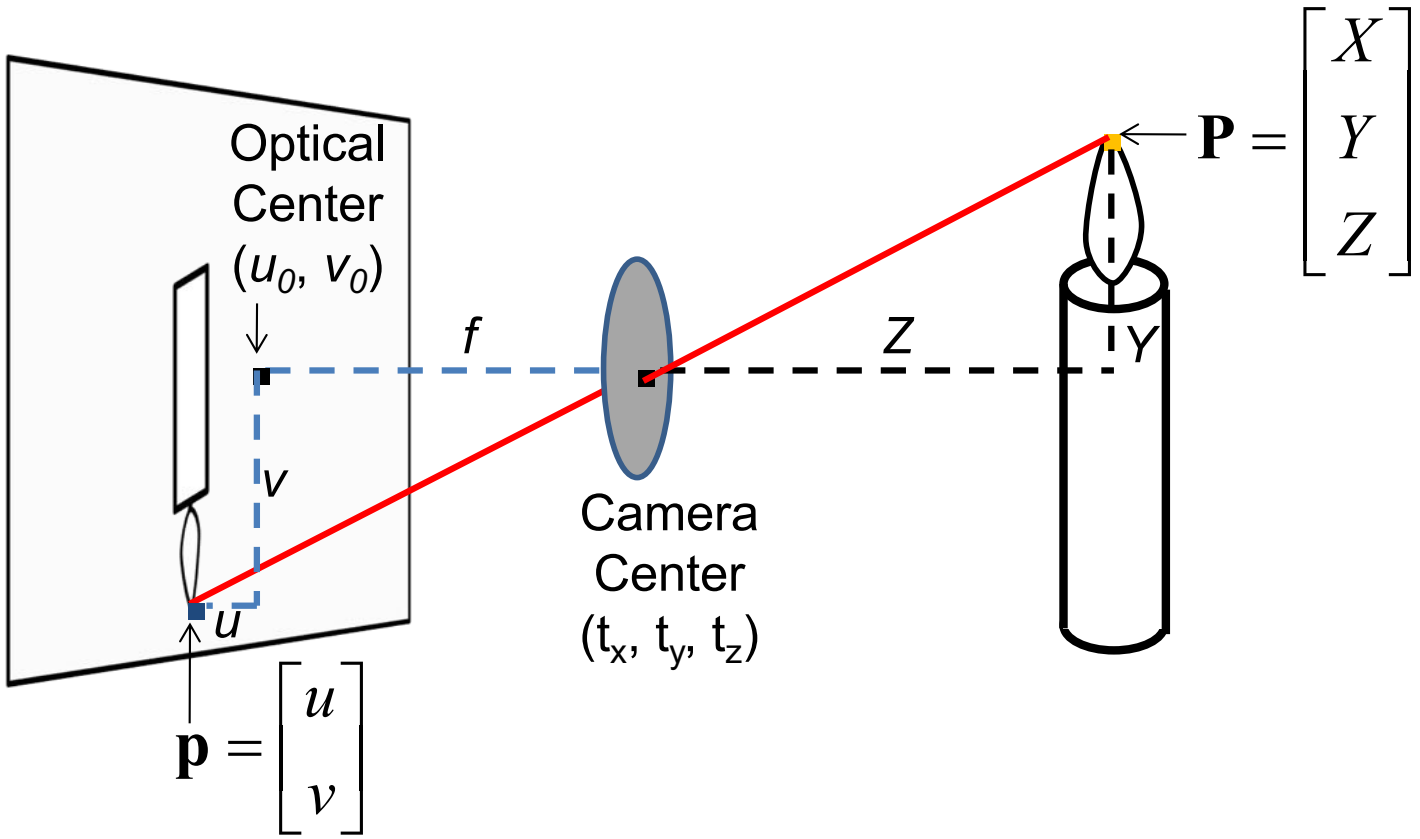
Projection: world coordinates→image coordinates



U = 2 . 2 / -5

V = 3 . 2 / -5

Projection: world coordinates→image coordinates



Homogeneous coordinates

Conversion

Converting to *homogeneous* coordinates

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene coordinates

Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Homogeneous coordinates

Invariant to scaling

$$k\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} kx \\ ky \\ kw \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{kx}{kw} \\ \frac{ky}{kw} \end{bmatrix} = \begin{bmatrix} \frac{x}{w} \\ \frac{y}{w} \end{bmatrix}$$

Homogeneous Coordinates

Cartesian Coordinates

Point in Cartesian is ray in Homogeneous

Transformation of co-ordinates

Non-homogeneous

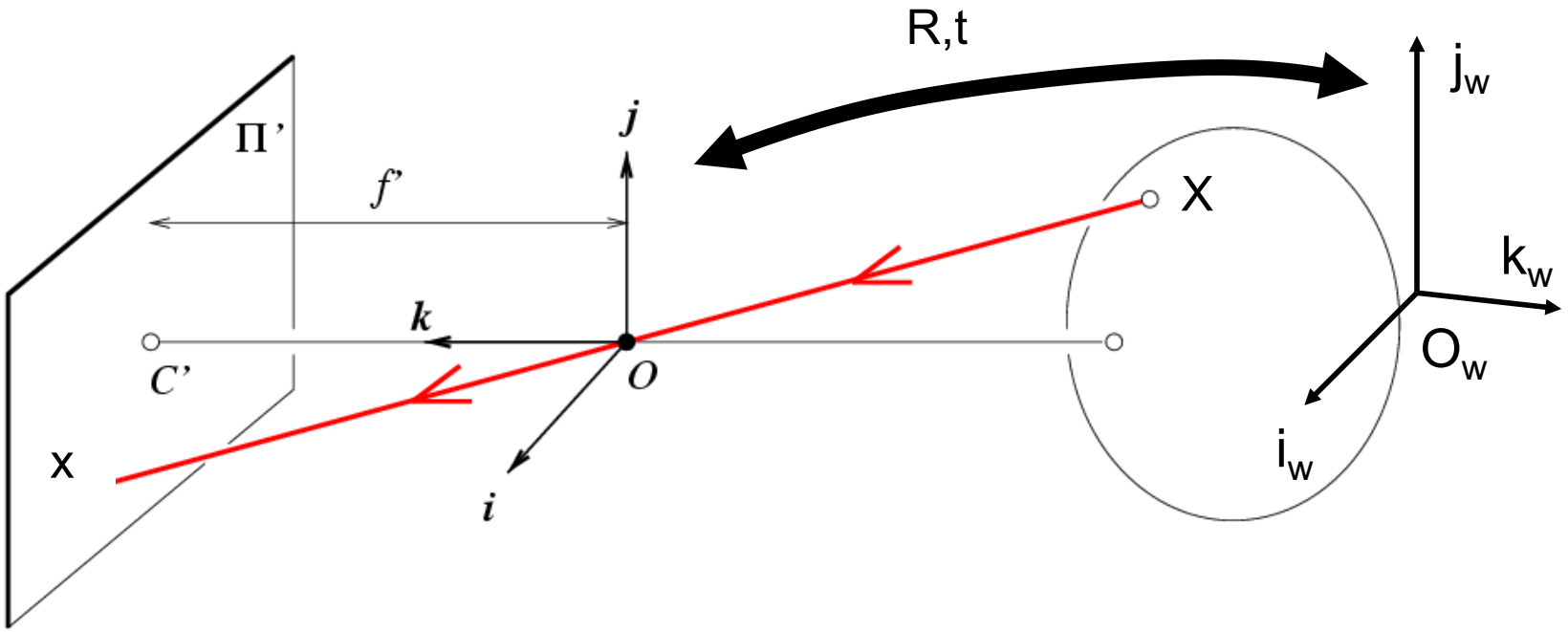
$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Homogeneous Coordinates

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

Slide Credit: Savarese

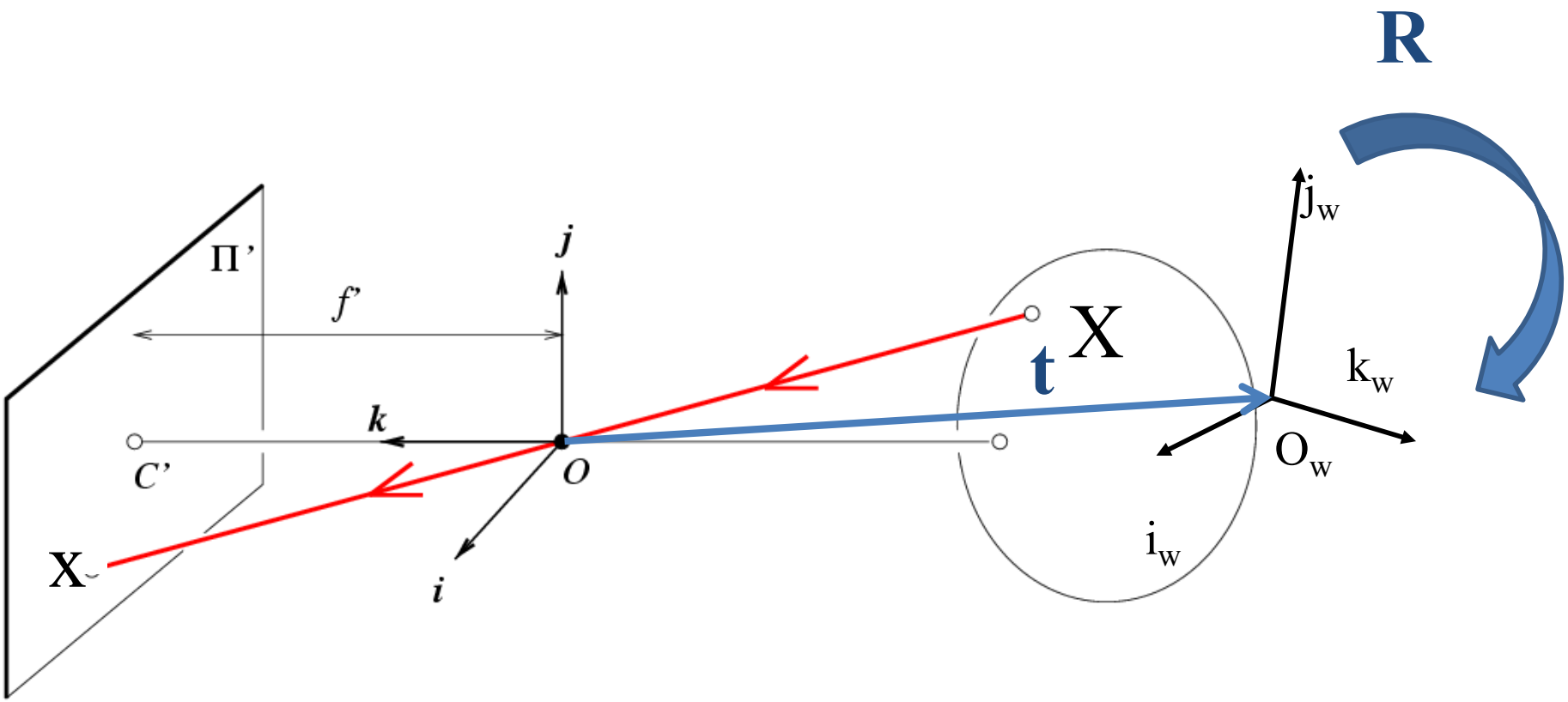
Projection matrix



$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

x: Image Coordinates: (u,v,1)
K: Intrinsic Matrix (3x3)
R: Rotation (3x3)
t: Translation (3x1)
X: World Coordinates: (X,Y,Z,1)

Oriented and Translated Camera



Camera projection

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

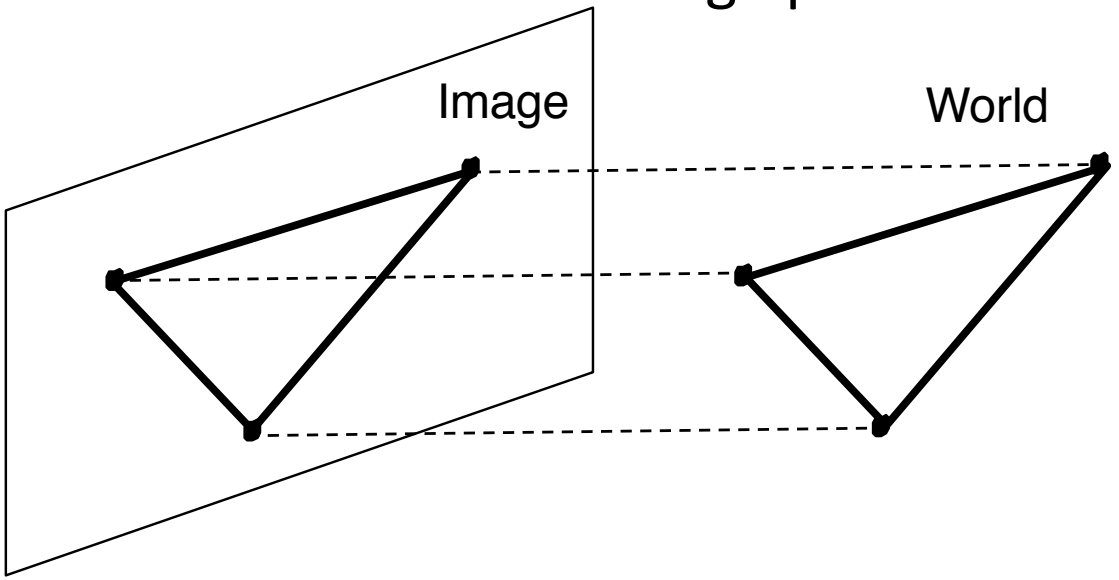
Degrees of freedom

$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix}}_{5}\underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}}_{6}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Orthographic Projection

- Special case of perspective projection
  - Distance from the COP to the image plane is infinite



  - Also called “parallel projection”
  - What’s the projection matrix?

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Slide by Steve Seitz

Object recognition and Computer Vision 2017

Instance-level recognition I: Local invariant features, correspondence, image matching

Image matching and recognition with local features

The goal: establish correspondence between two or more images

$\mathbf{x} = \mathrm{P}\mathbf{X} \quad \mathbf{x}' = \mathrm{P}'\mathbf{X}$

P : 3×4 matrix

$\mathbf{X}$: 4-vector

$\mathbf{x}$: 3-vector

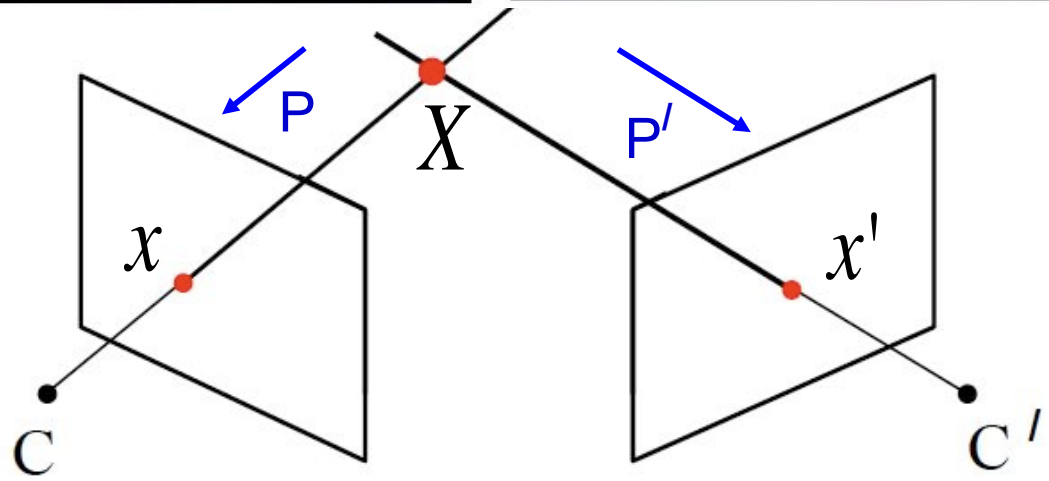


Image points x and x' are in correspondence if they are projections of the same 3D scene point X.

Images courtesy A. Zisserman

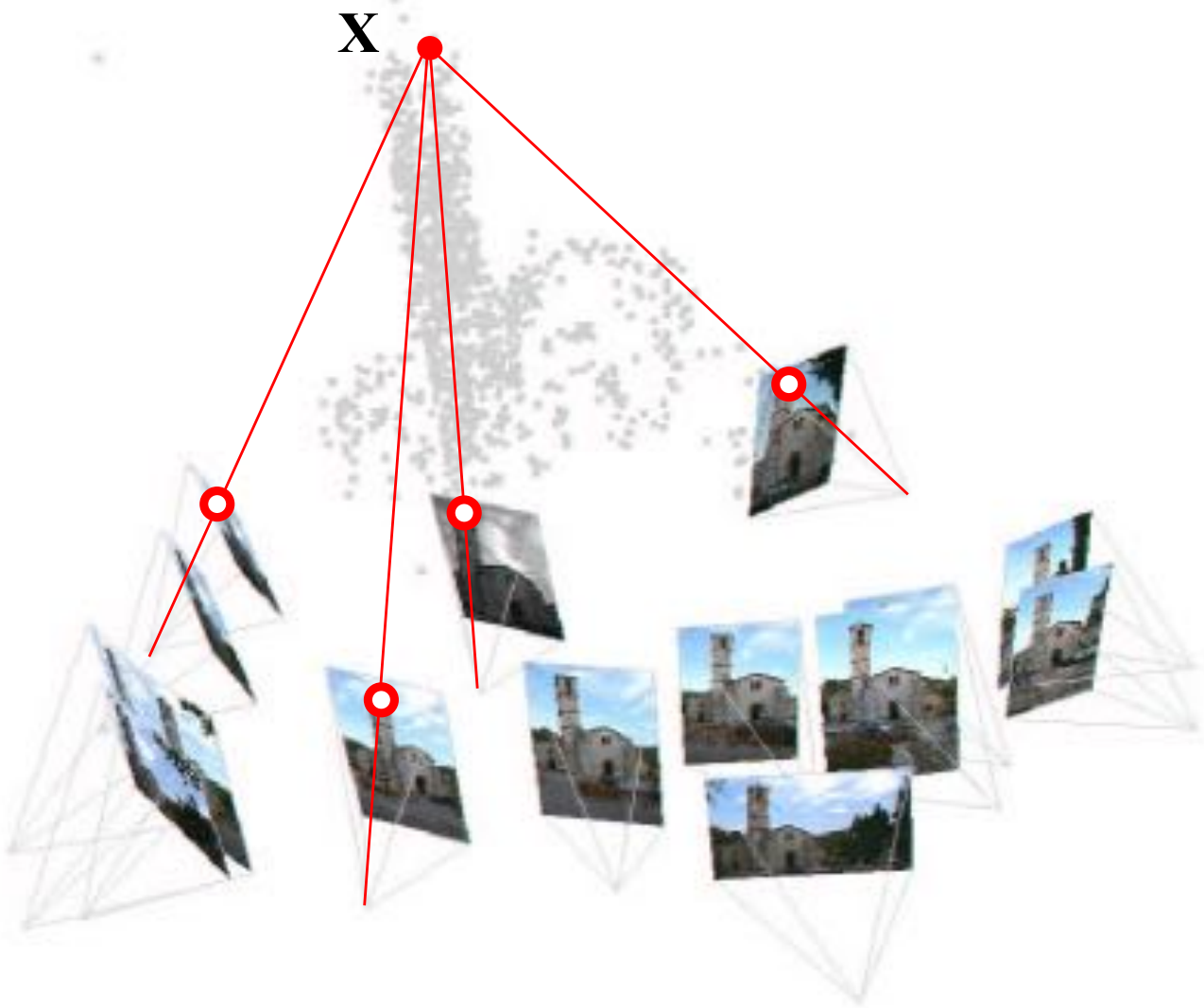
Example I: Wide baseline matching and 3D reconstruction

Establish correspondence between two (or more) images.

[Schaffalitzky and Zisserman ECCV 2002]

Example I: Wide baseline matching and 3D reconstruction

Establish correspondence between two (or more) images.



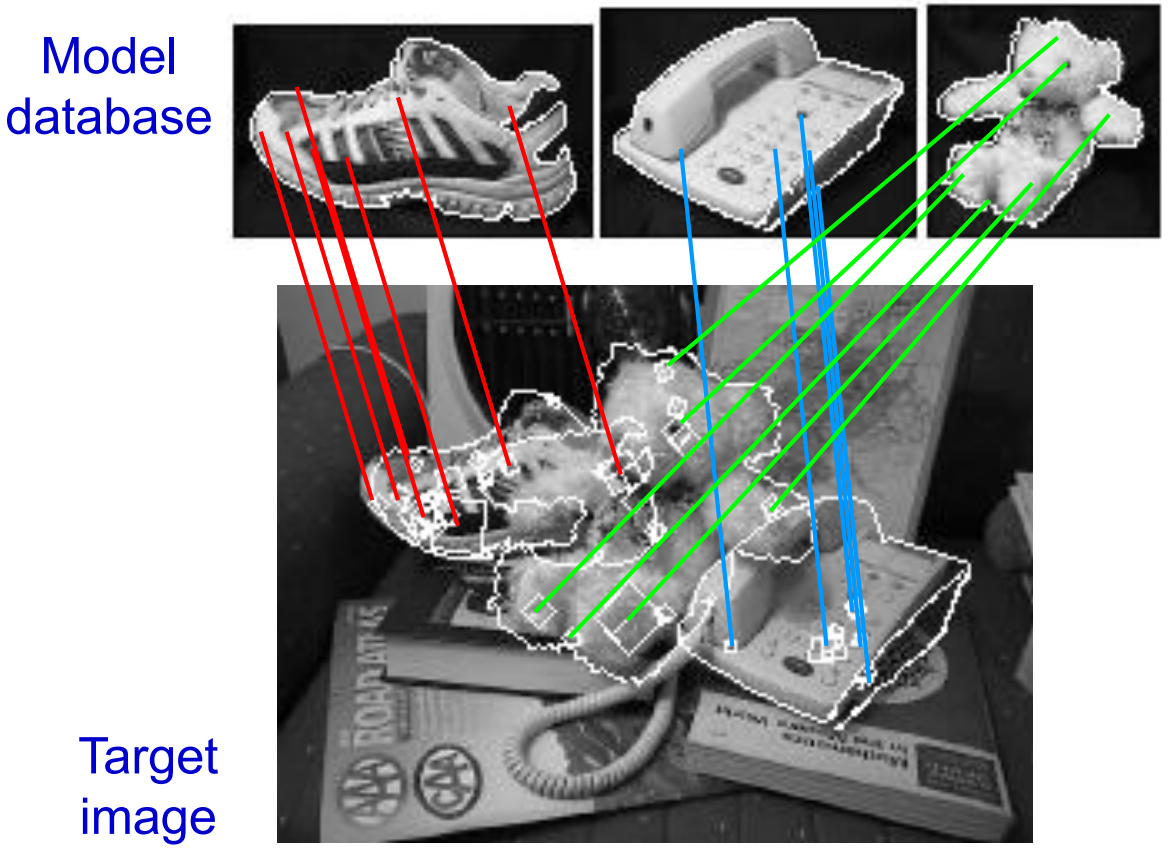
[Schaffalitzky and Zisserman ECCV 2002]

[Agarwal, Snavely, Simon, Seitz, Szeliski, ICCV'09] – Building Rome in a Day

57,845 downloaded images, 11,868 registered images. This video: 4,619 images.

Example II: Object recognition

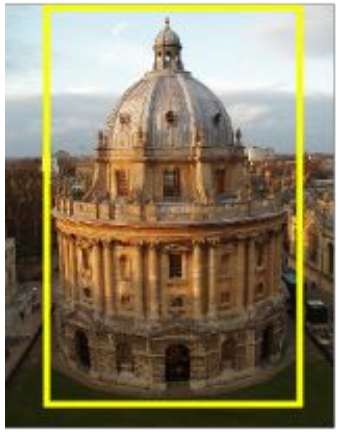
Establish correspondence between the target image and (multiple) images in the model database.



[D. Lowe, 1999]

Example III: Visual search

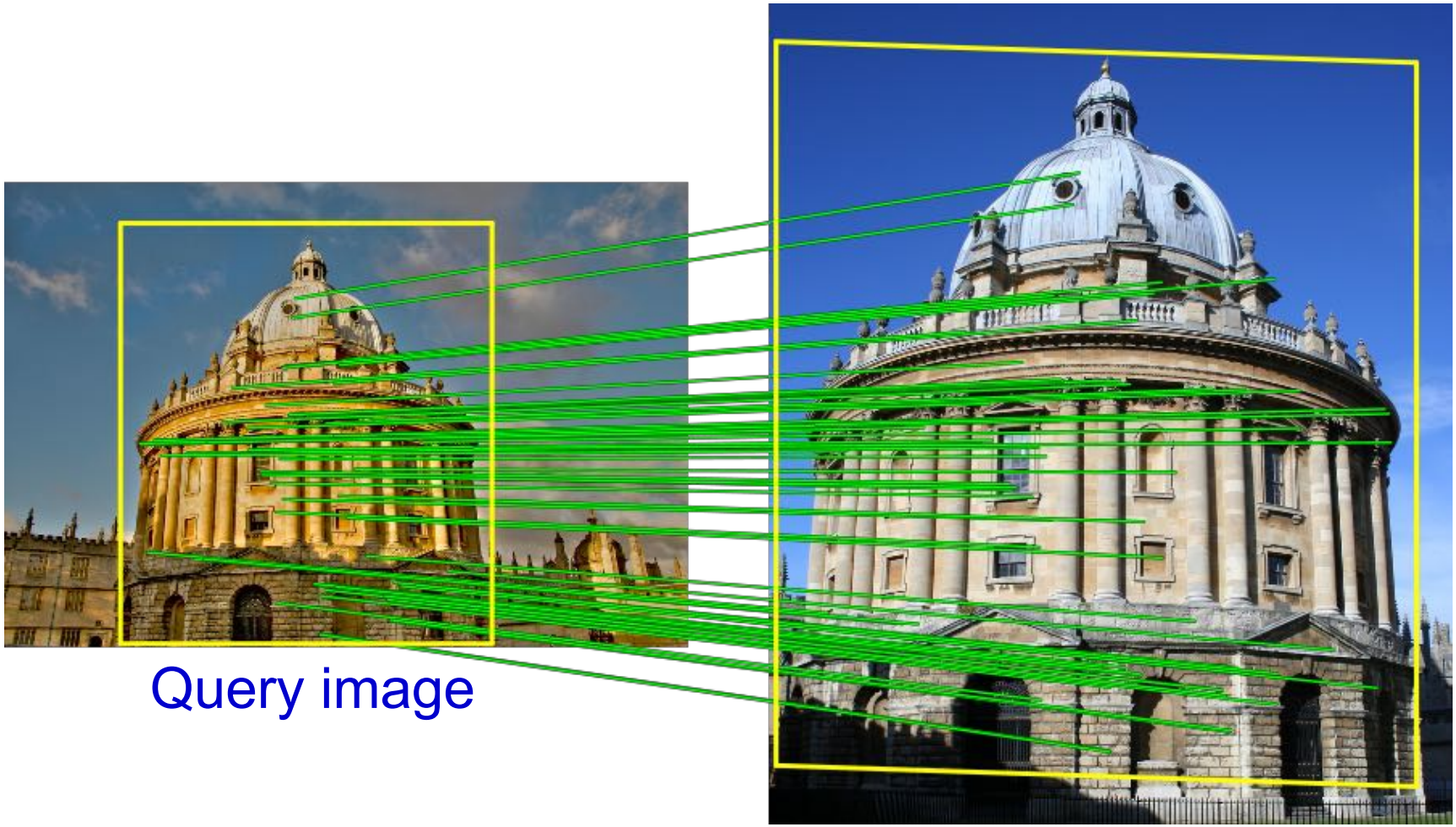
Given a query image, find images depicting the same place / object in a large unordered image collection.



Find these landmarks

...in these images and 1M more

Establish correspondence between the query image and all images from the database depicting the same object / scene.

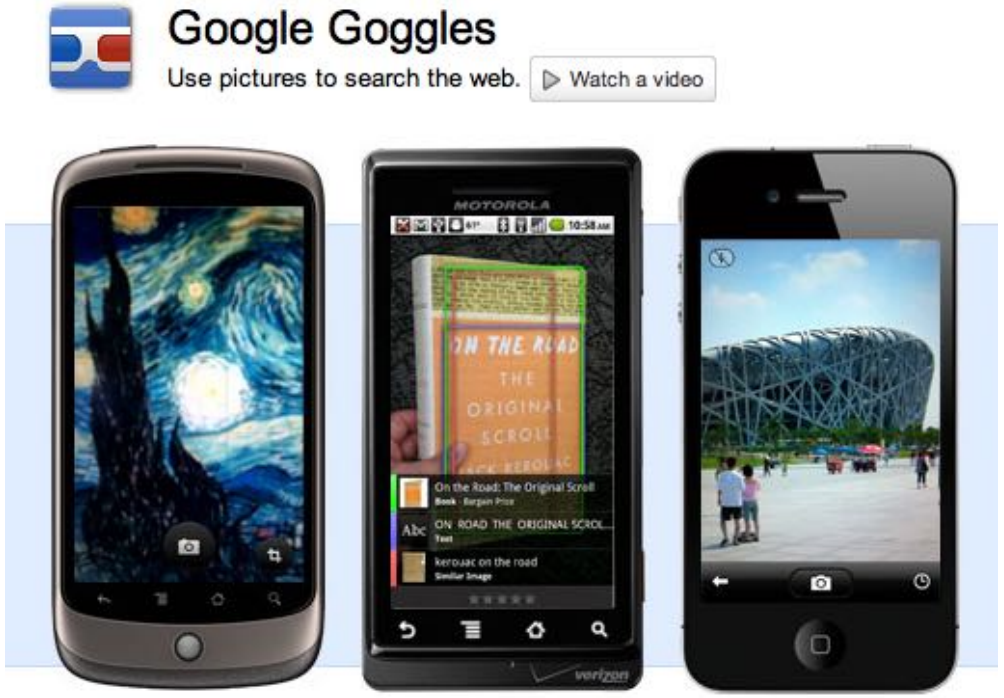


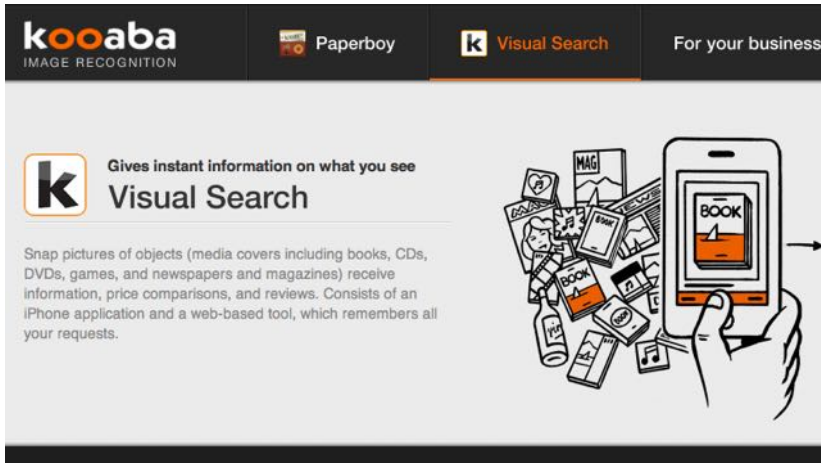
Query image

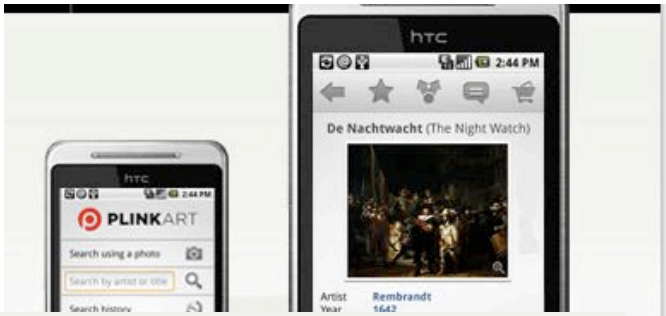
Database image(s)

Mobile visual search

Bing visual scan





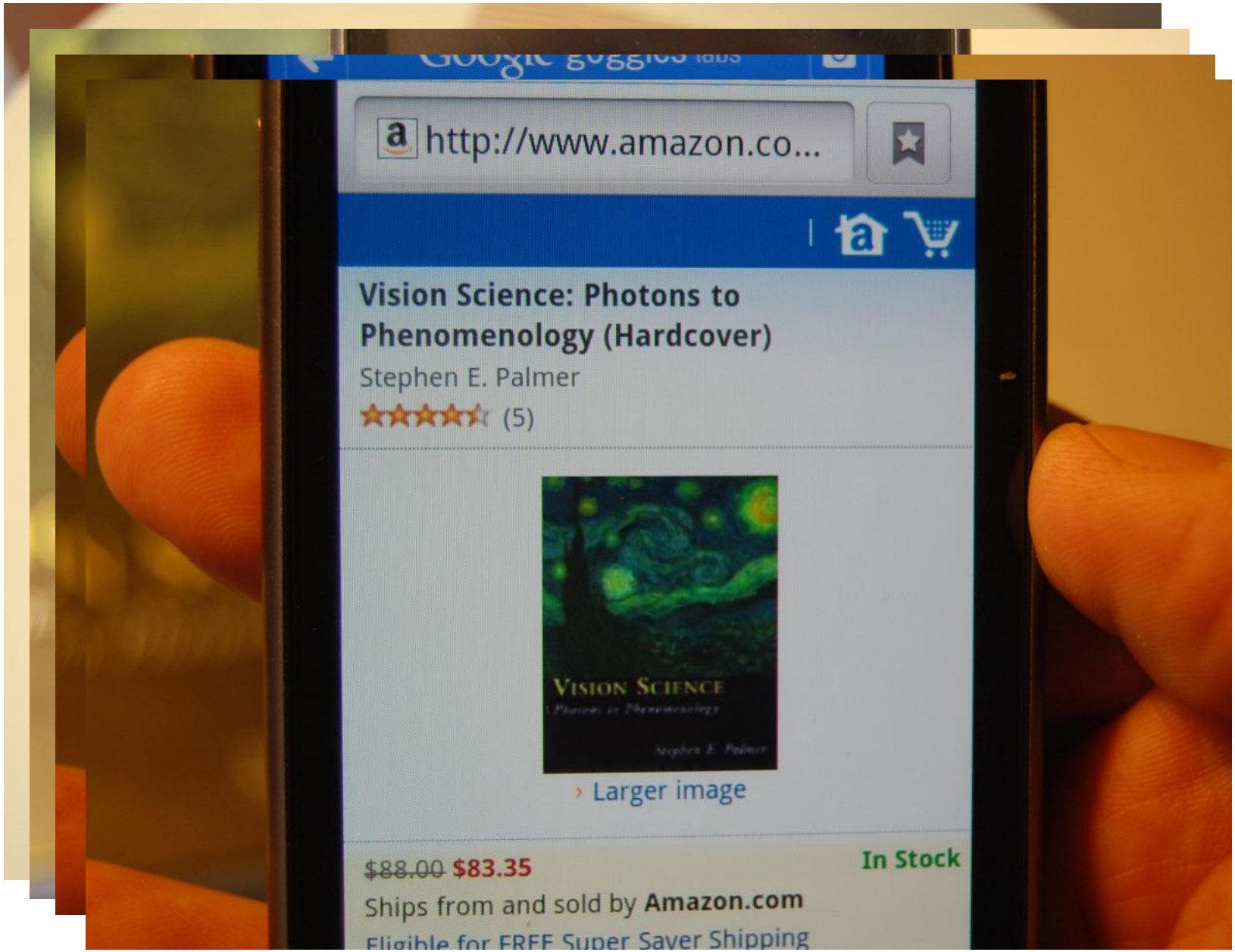


PLINKART

Plink Art is an app for your mobile phone that lets you identify almost any work of art just by taking a photo of it.

and others… Snaptell.com, Millpix.com

Example

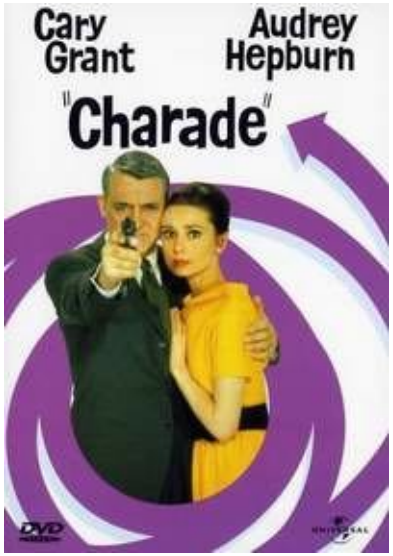


Slide credit: I. Laptev

Example: Visual search in an entire feature length movie

Visually defined query

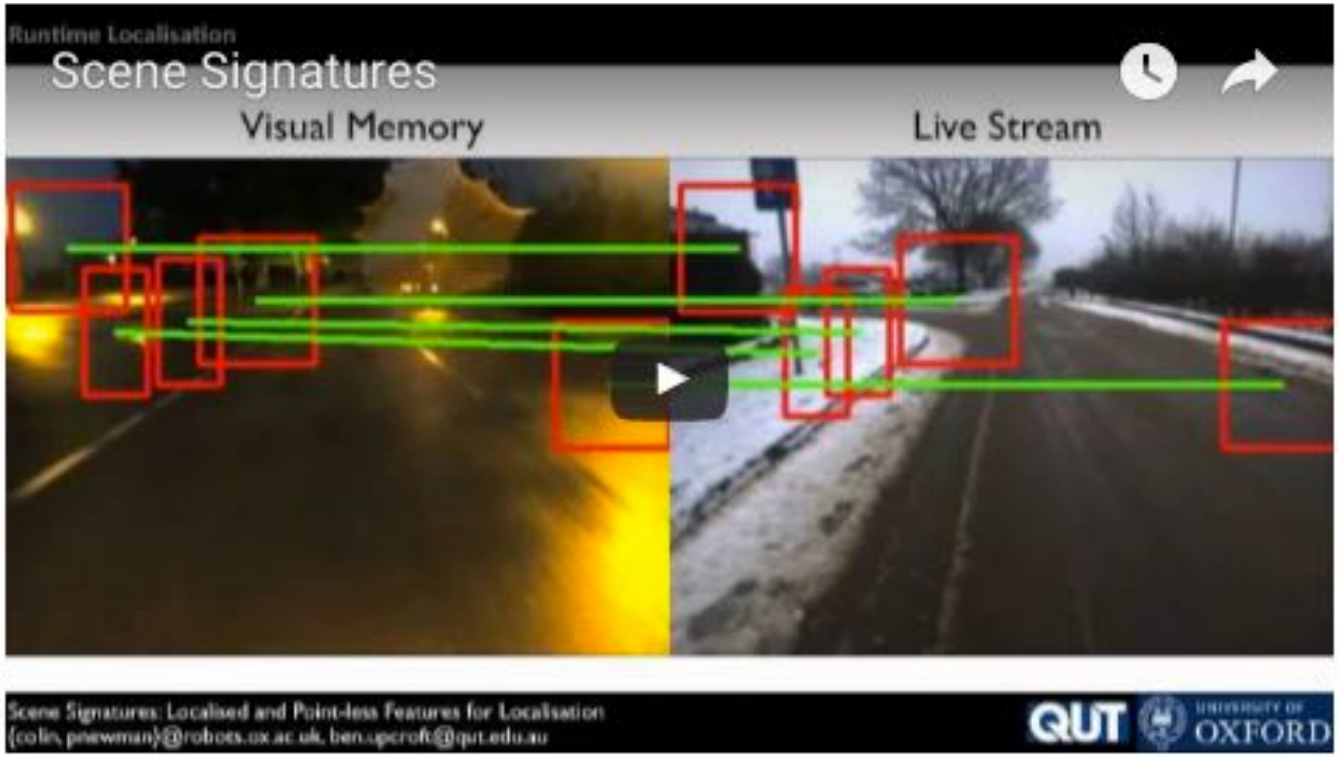
“Find this bag”



“Charade” [Donen, 1963]

Demo:
http://www.robots.ox.ac.uk/~vgg/research/vgoogle/index.html

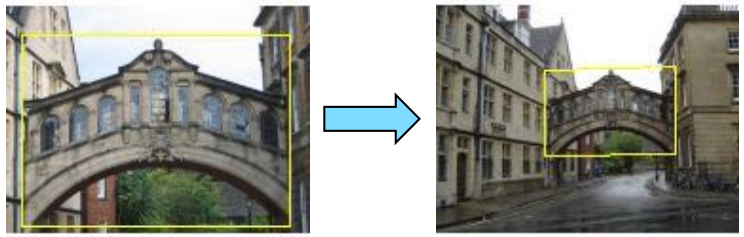
Visual navigation for autonomous robotics



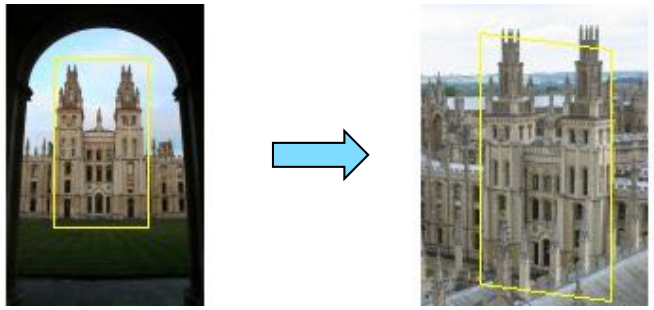
http://mrg.robots.ox.ac.uk/theme/localisation/

Why is it difficult?

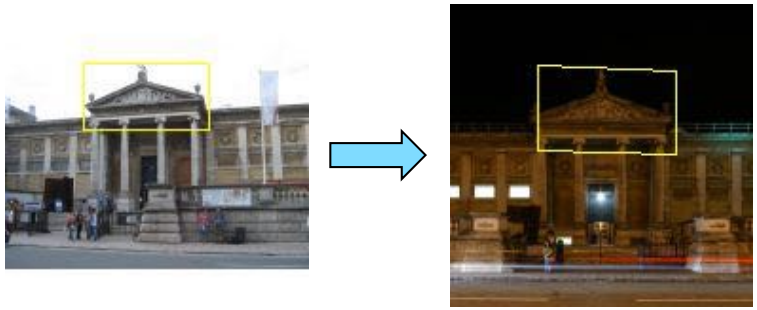
Want to establish correspondence despite possibly large changes in scale, viewpoint, lighting and partial occlusion



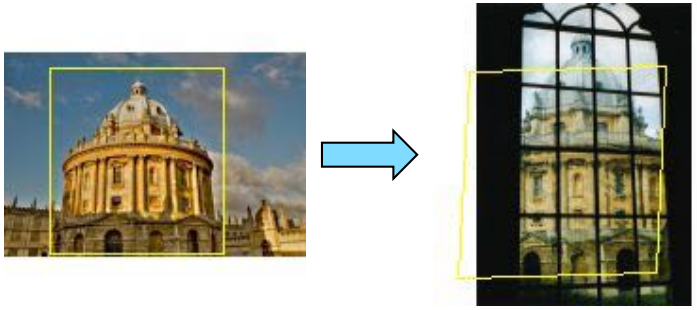
Scale



Viewpoint



Lighting



Occlusion

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

Approach

1. **Pre-processing:**
   - Detect local features.
   - Extract descriptor for each feature.

2. **Matching:** Establish tentative (putative) correspondences based on local appearance of individual features (their descriptors).

3. **Verification:** Verify matches based on semi-local / global geometric relations.

4. **Current research challenges**

Outline: feature detection

Edges

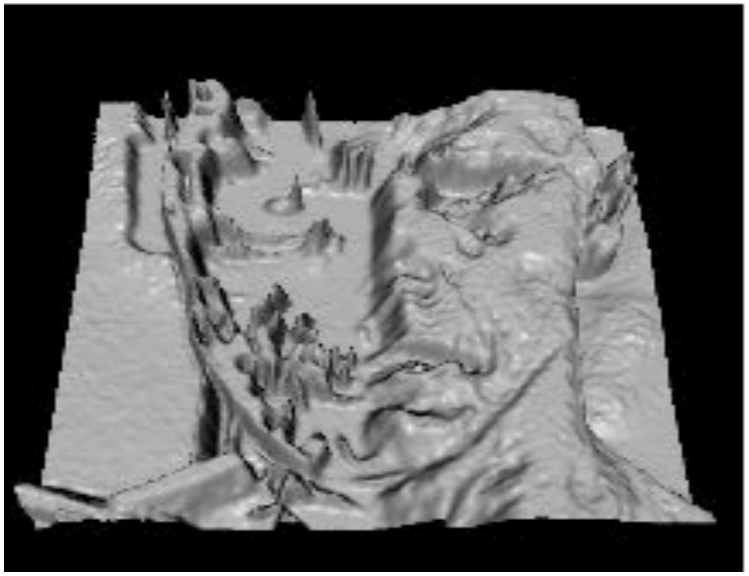
Corners

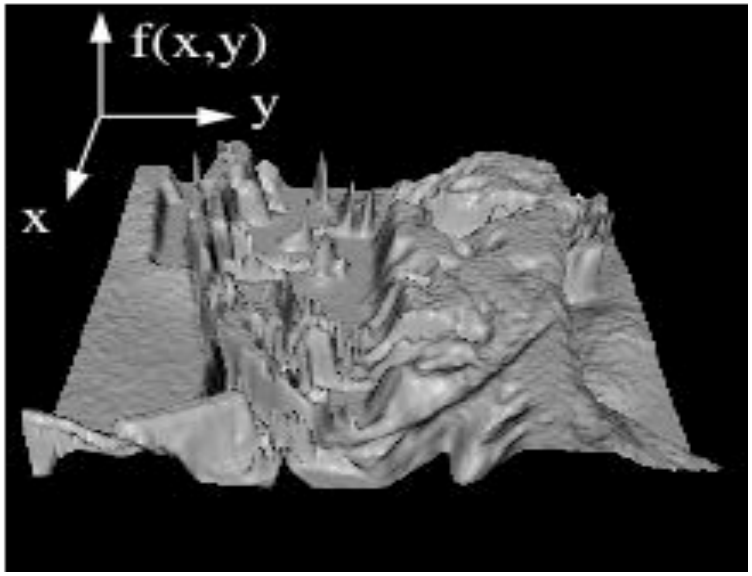
Blobs

[Contours]

[Regions]

Images as functions





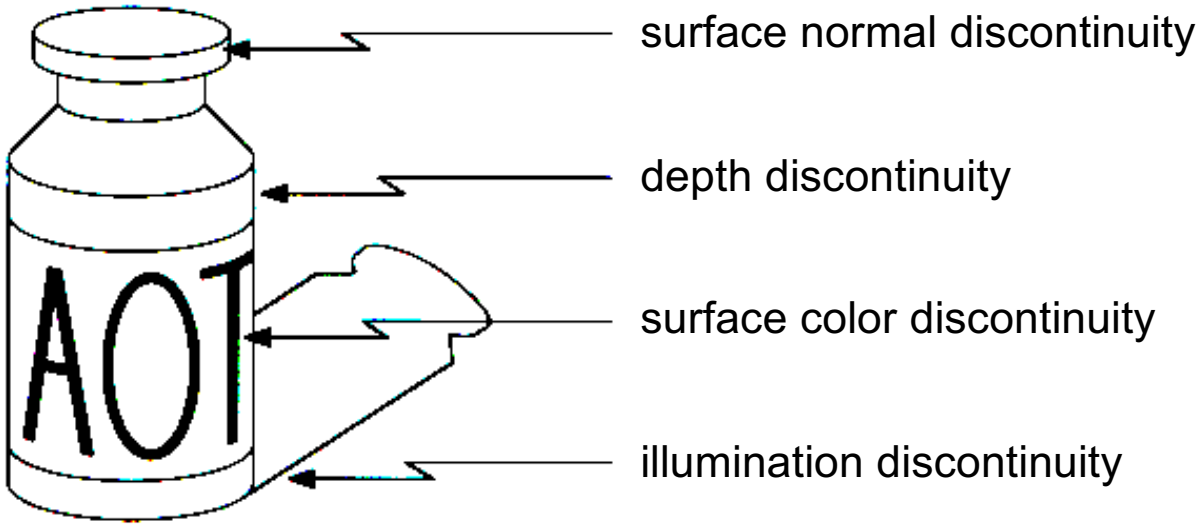
Source: S. Seitz

Edge detection

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels
- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

Edge detection: Origin of Edges

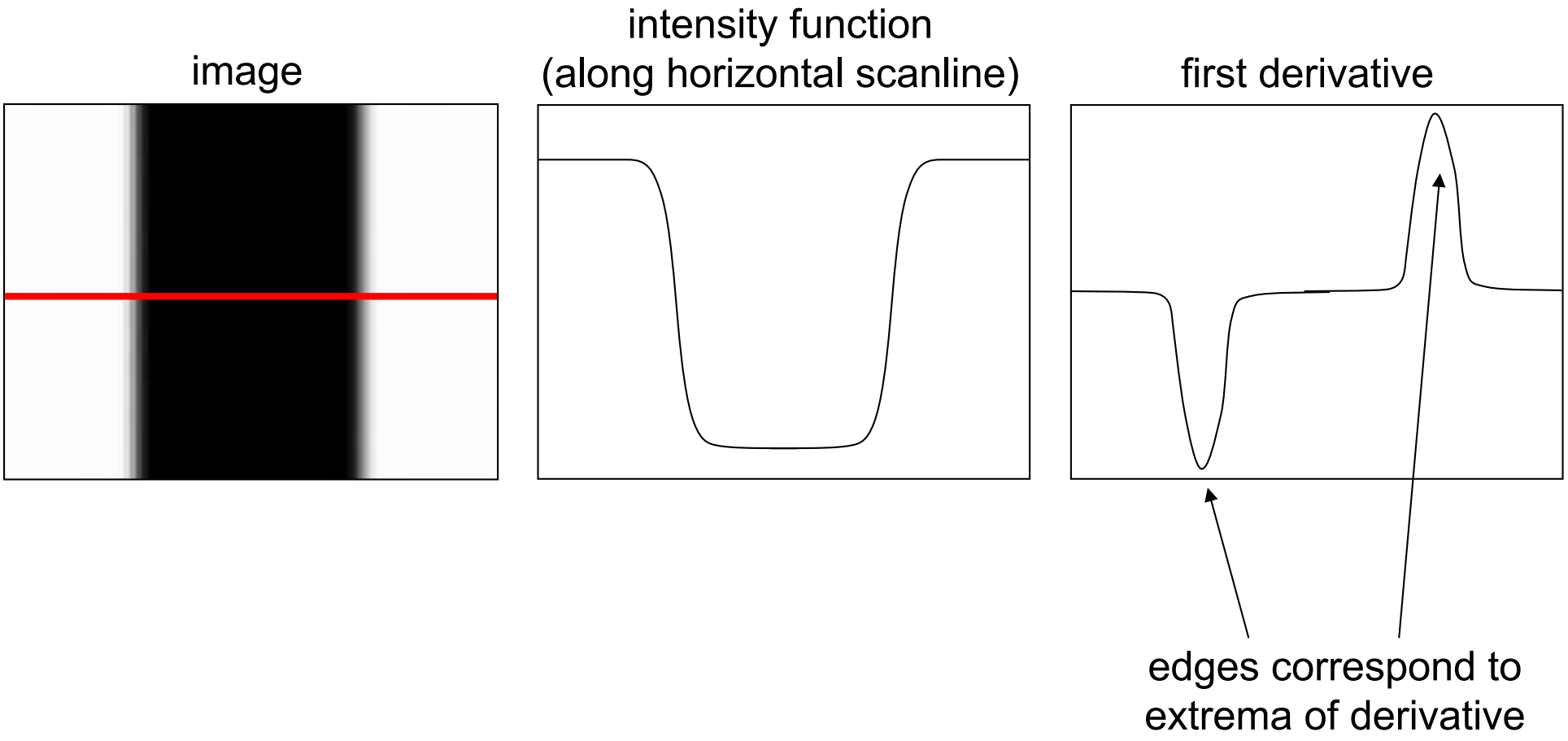


Edges are caused by a variety of factors

Source: Steve Seitz

Characterizing edges

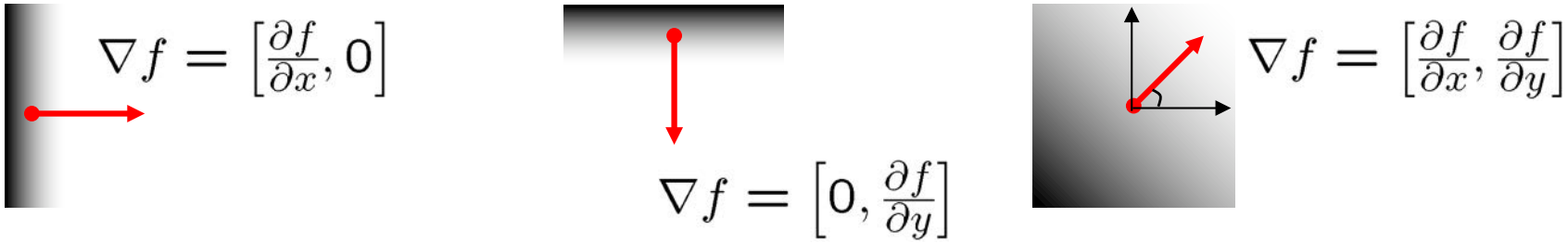
- An edge is a place of rapid change in the image intensity function



Slide: S. Lazebnik

Image gradient

The gradient of an image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$



The gradient points in the direction of most rapid increase in intensity

The gradient direction is given by $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

- how does this relate to the direction of the edge?

The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Source: Steve Seitz

Differentiation and convolution

Recall, for 2D function, f(x,y):

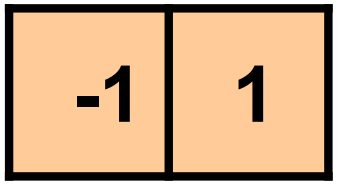
$$\frac{\partial f}{\partial x} = \lim_{\varepsilon \to 0} \left(\frac{f(x+\varepsilon, y)}{\varepsilon} - \frac{f(x,y)}{\varepsilon} \right)$$

We could approximate this as

$$\frac{\partial f}{\partial x} \approx \frac{f(x_{n+1}, y) - f(x_n, y)}{\Delta x}$$

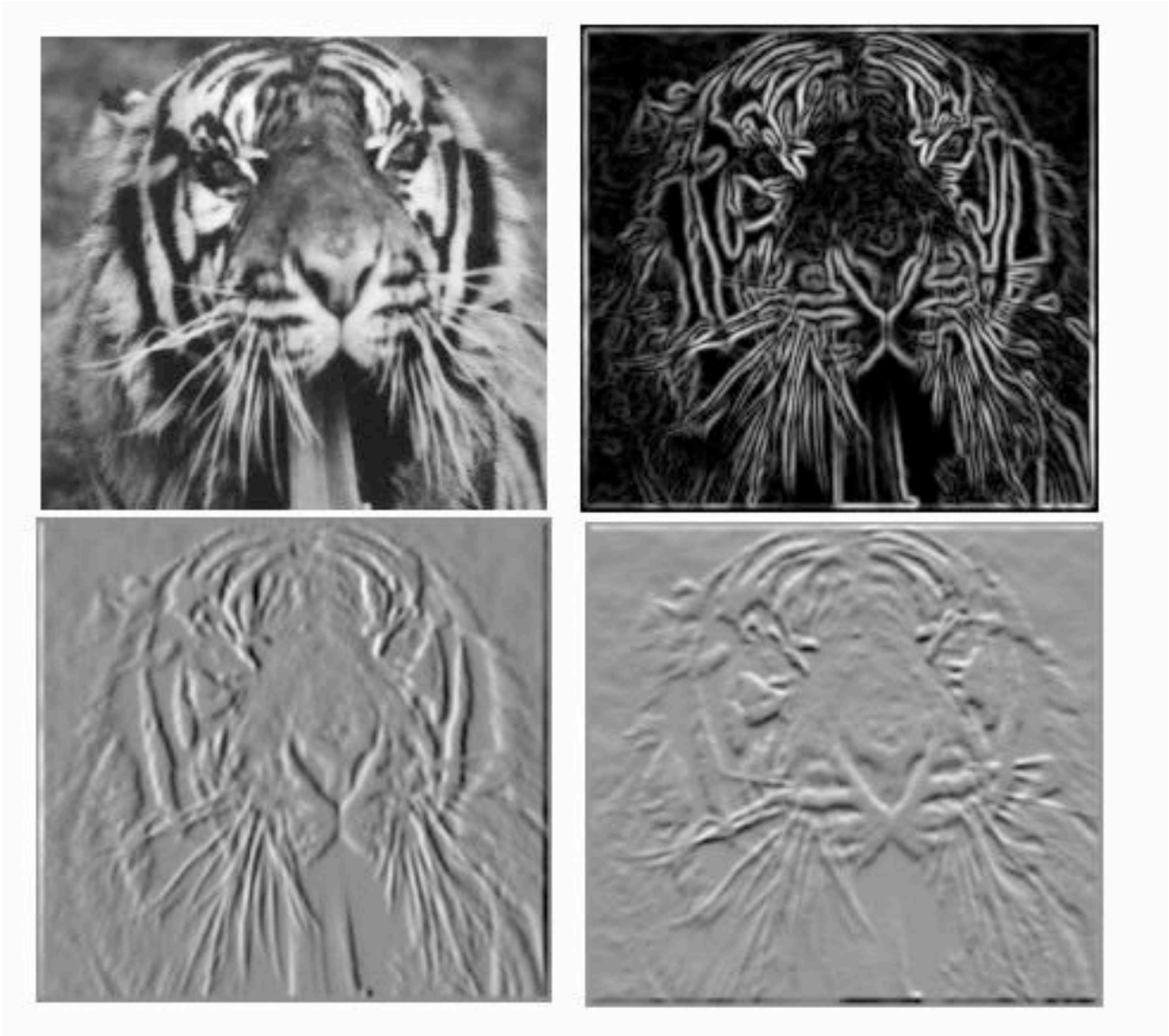
This is linear and shift invariant, so must be the result of a convolution.

(which is obviously a convolution)



Source: D. Forsyth, D. Lowe

Finite differences: example



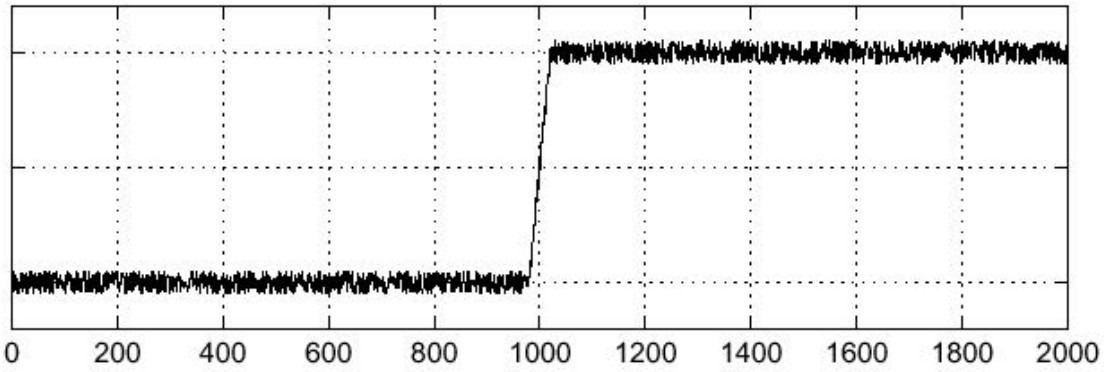
Which one is the gradient in the x-direction (resp. y-direction)?

Slide: S. Lazebnik

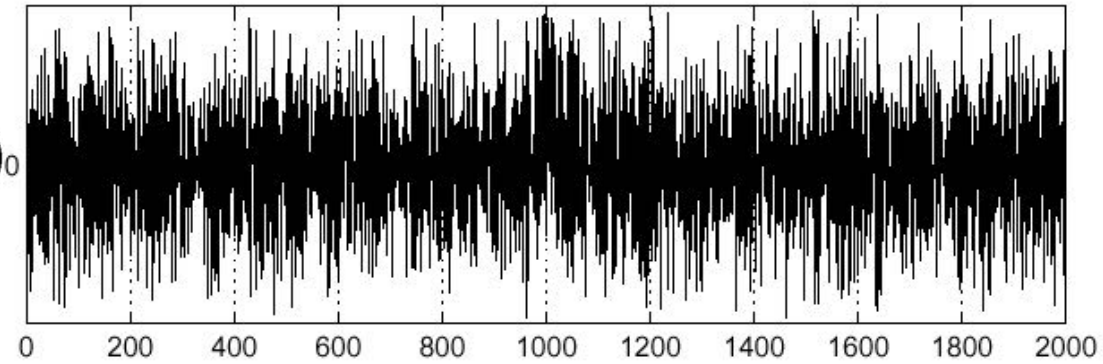
Effects of noise

Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal



$\frac{d}{dx} f(x)$



Where is the edge?

Source: S. Seitz

Effects of noise

- Finite difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- What is to be done?

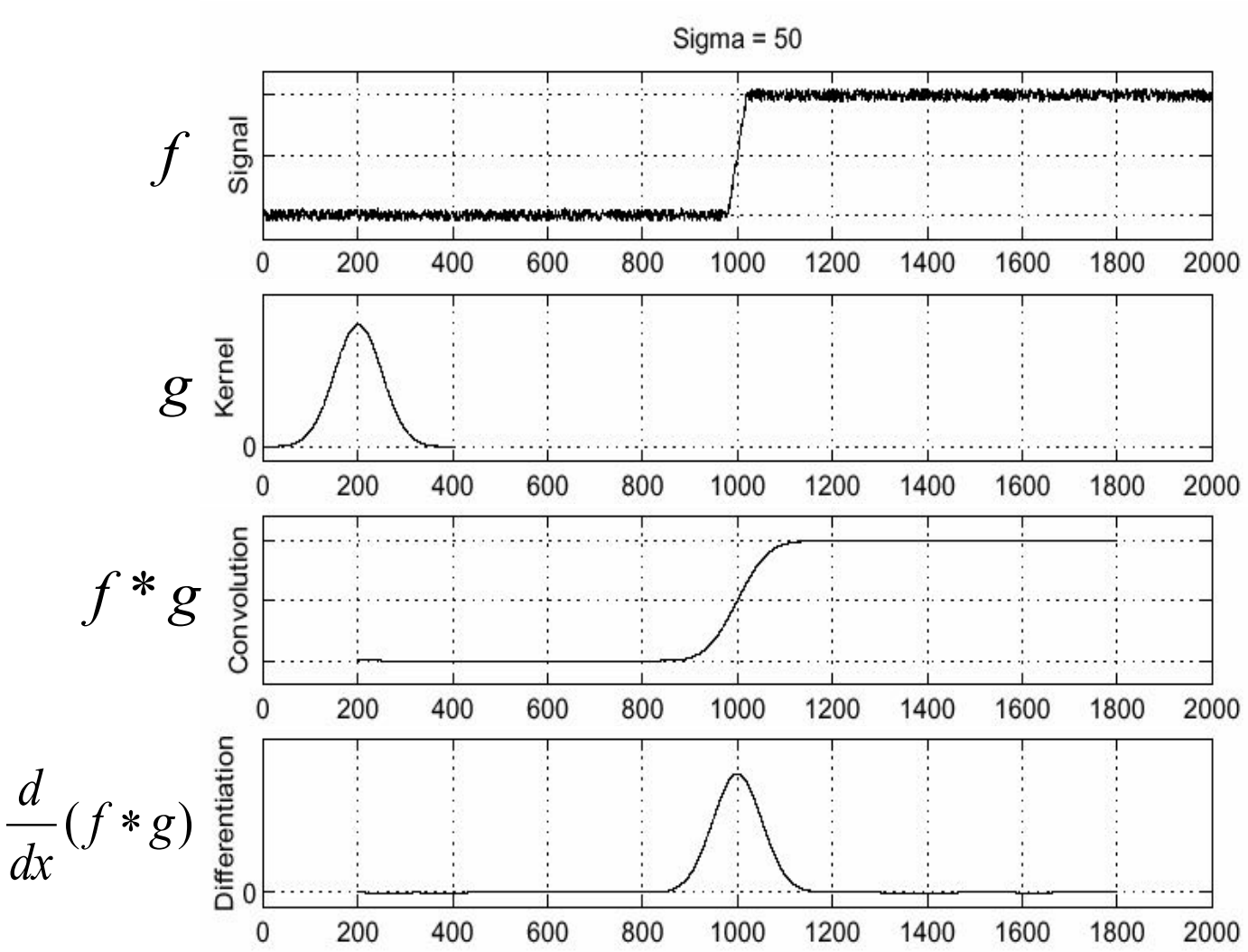
Source: D. Forsyth

Effects of noise

- Finite difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- What is to be done?
  - Smoothing the image should help, by forcing pixels different to their neighbors (=noise pixels?) to look more like neighbors

Source: D. Forsyth

Solution: smooth first



f

g

$f * g$

$\frac{d}{dx}(f * g)$

- To find edges, look for peaks in $\frac{d}{dx}(f * g)$

Source: S. Seitz

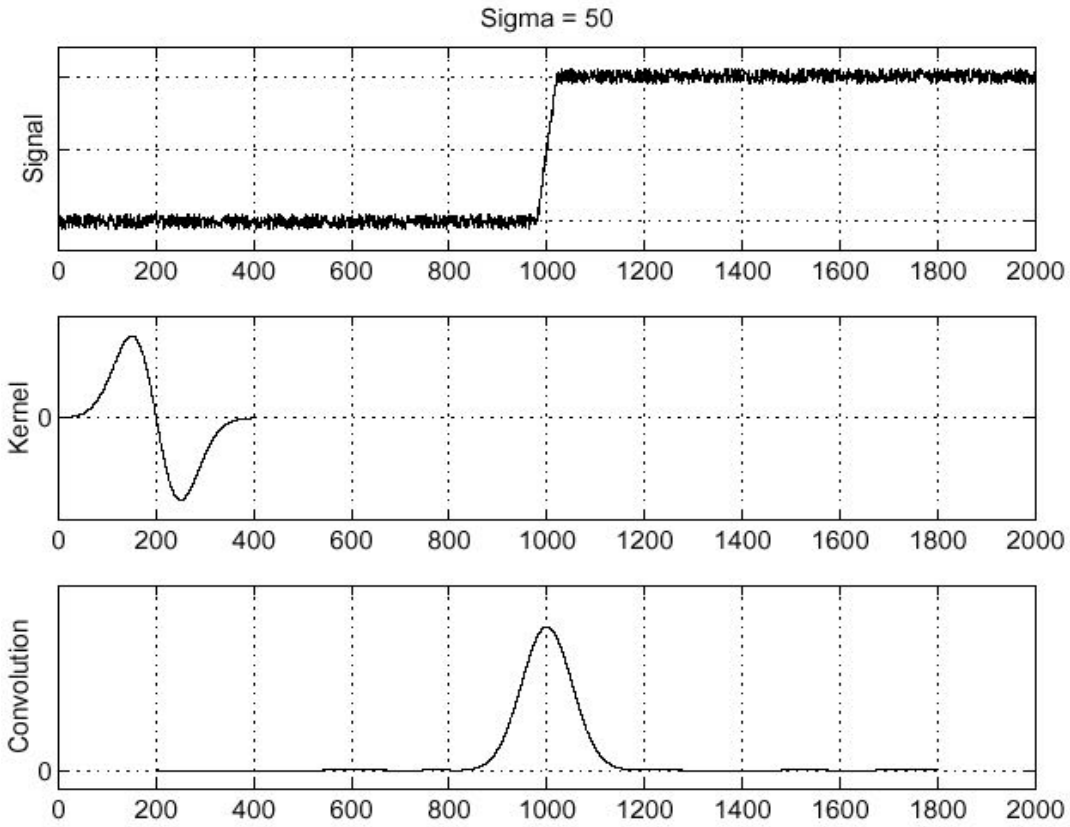
Derivative theorem of convolution

- Differentiation is convolution, and convolution is associative: $\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$
- This saves us one operation:

f

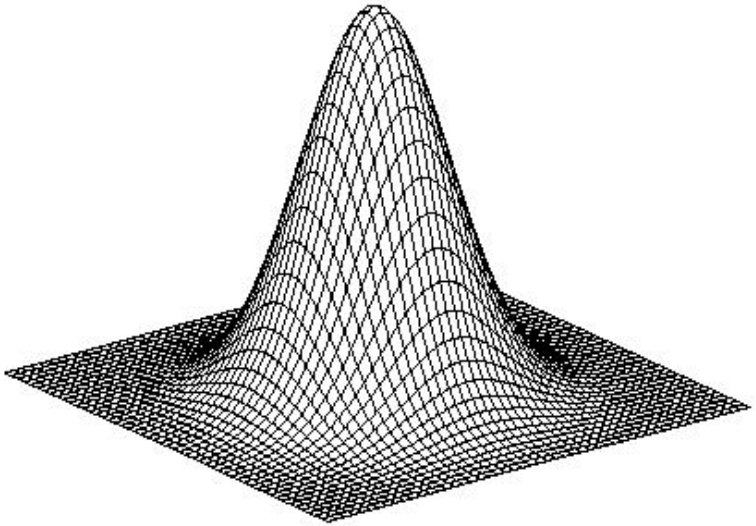
$\frac{d}{dx} g$

$f * \frac{d}{dx} g$

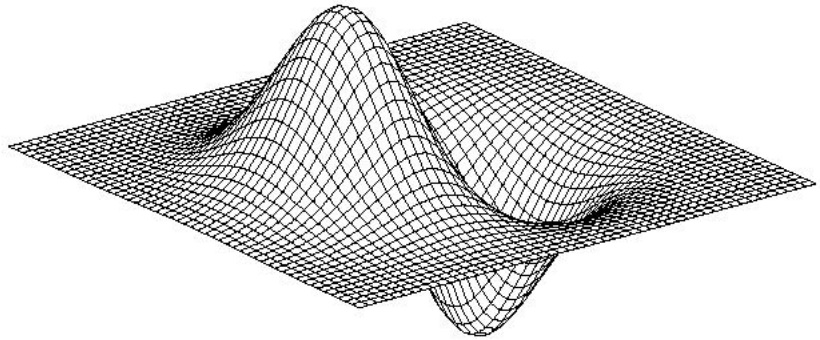


Source: S. Seitz

Derivative of Gaussian filter



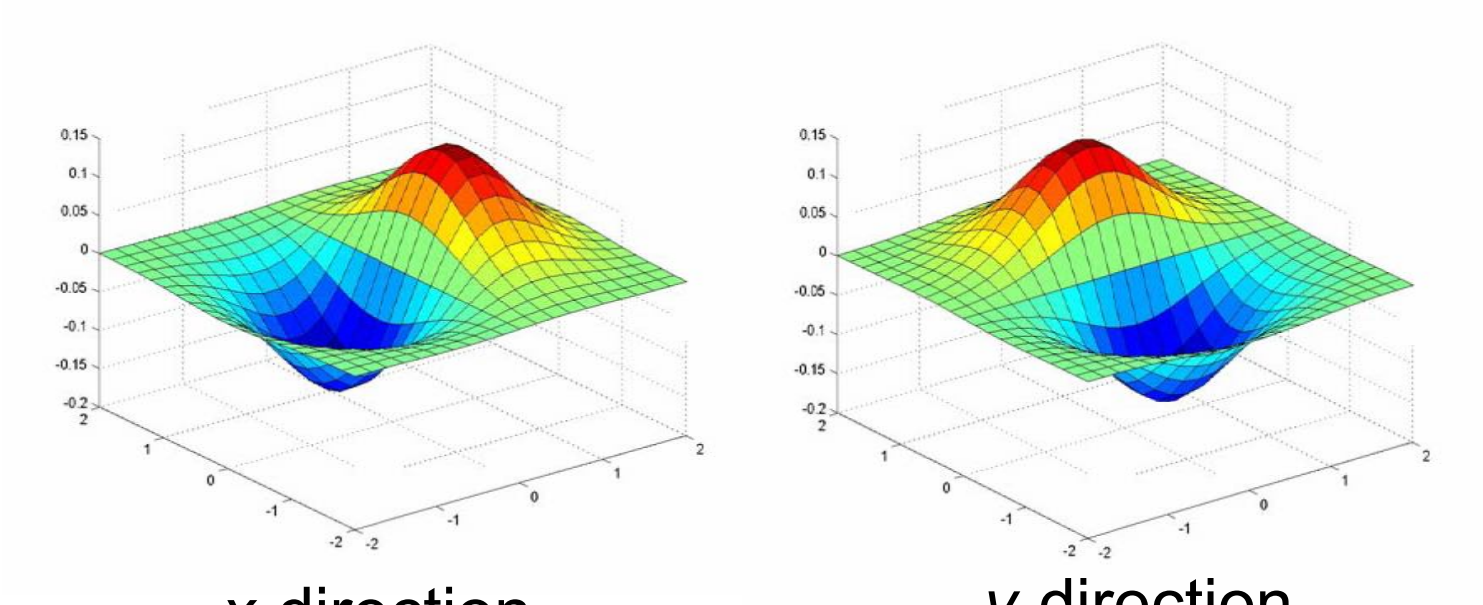
* [1 -1] =



Is this filter separable?

Slide: S. Lazebnik

Derivative of Gaussian filter



x-direction

y-direction

Which one finds horizontal/vertical edges?

Slide: S. Lazebnik

Outline: feature detection

Edges

Corners

Blobs

[Contours]

[Regions]

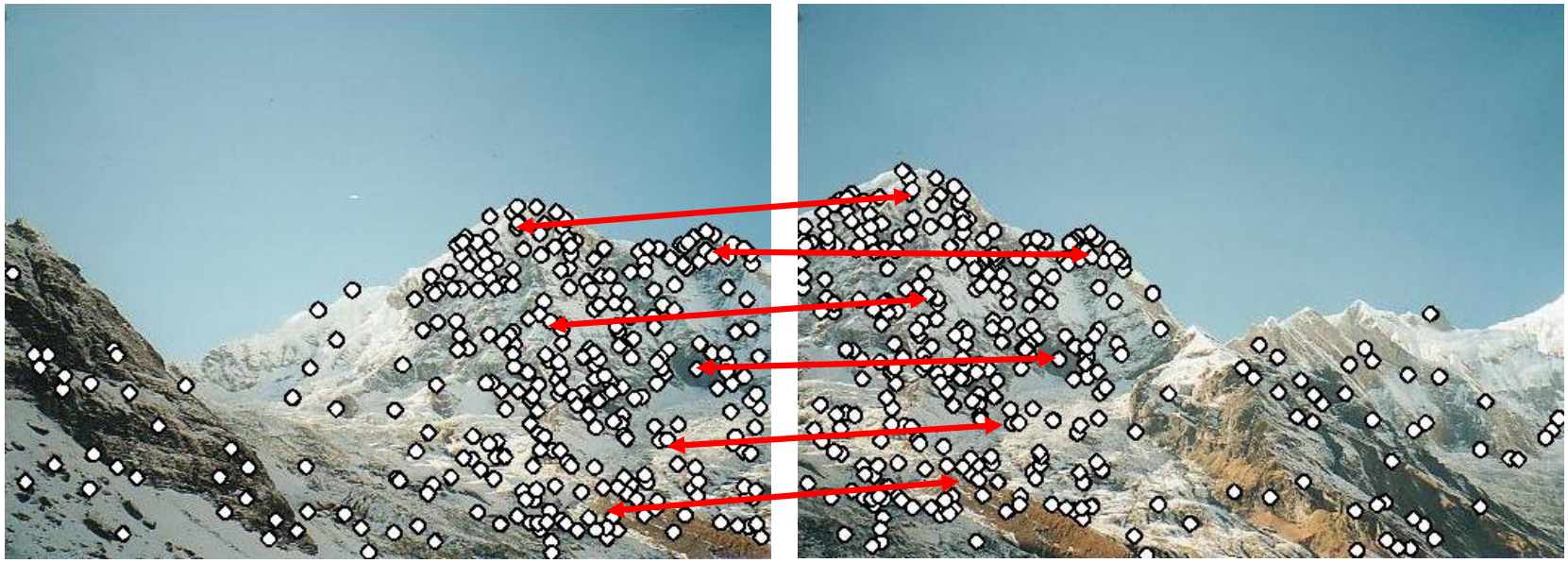
Why extract features?

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

Slide: S. Lazebnik

Why extract features?

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



Step 1: extract features

Step 2: match features

Slide: S. Lazebnik

Why extract features?

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

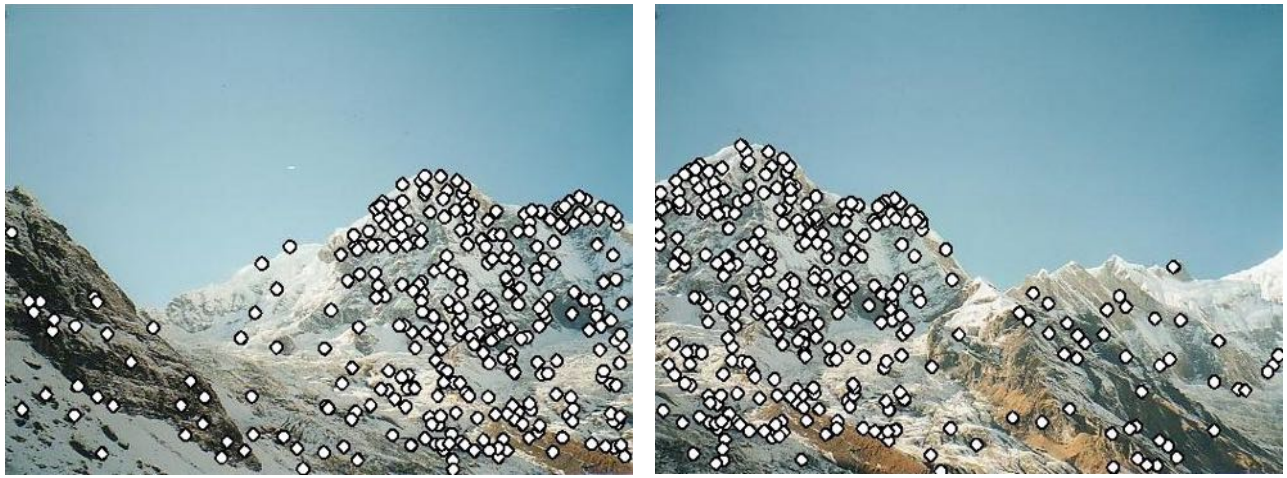
Step 1: extract features

Step 2: match features

Step 3: align images

Slide: S. Lazebnik

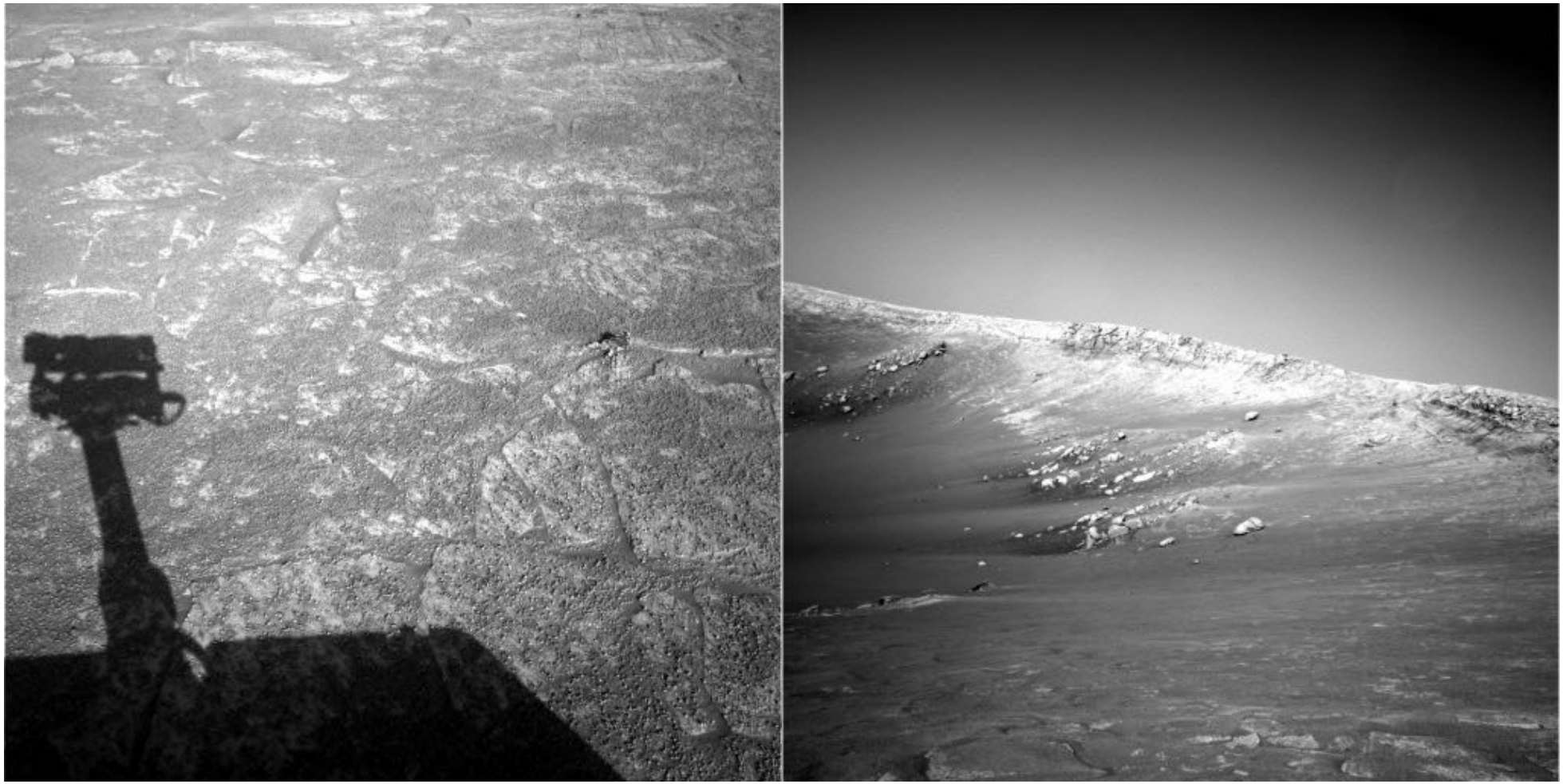
Characteristics of good features



- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature is distinctive
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

Slide: S. Lazebnik

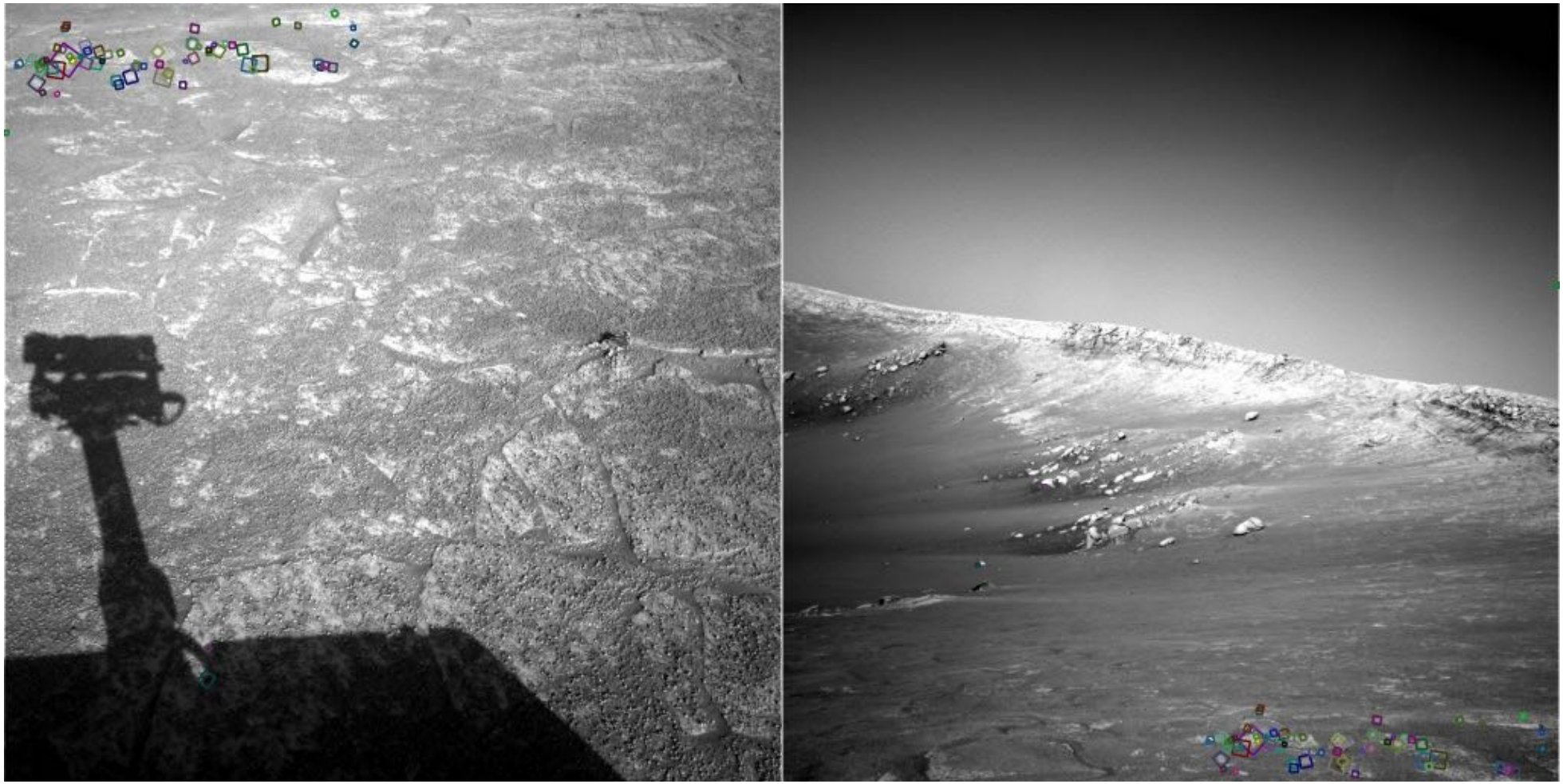
A hard feature matching problem



NASA Mars Rover images

Slide: S. Lazebnik

Answer below (look for tiny colored squares…)

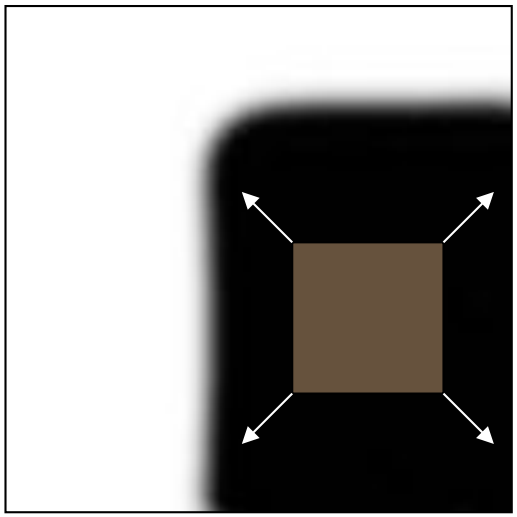


NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

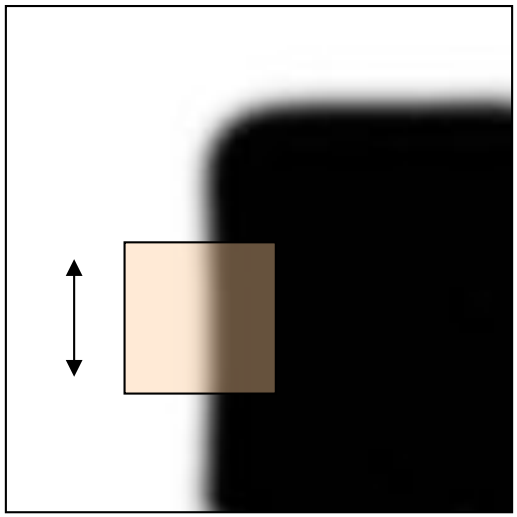
Slide: S. Lazebnik

Corner Detection: Basic Idea

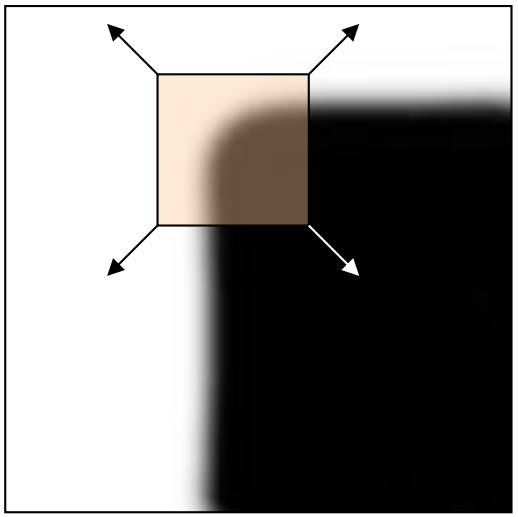
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region: no change in all directions



"edge": no change along the edge direction



"corner": significant change in all directions

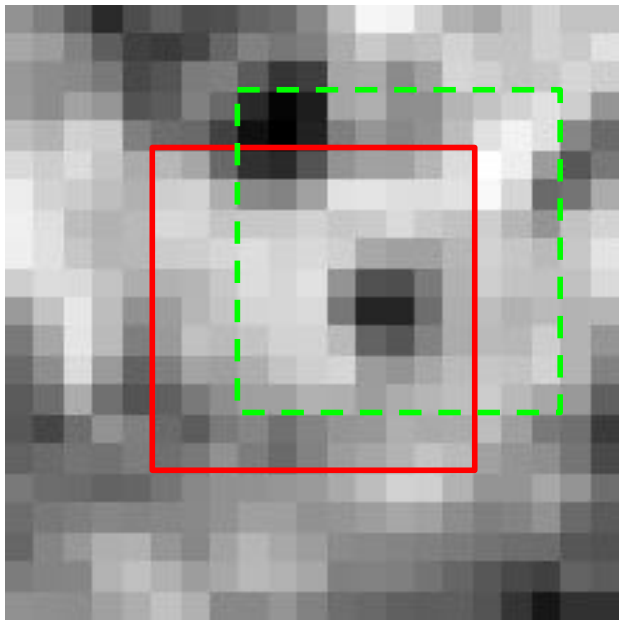
Source: A. Efros

Corner Detection: Mathematics

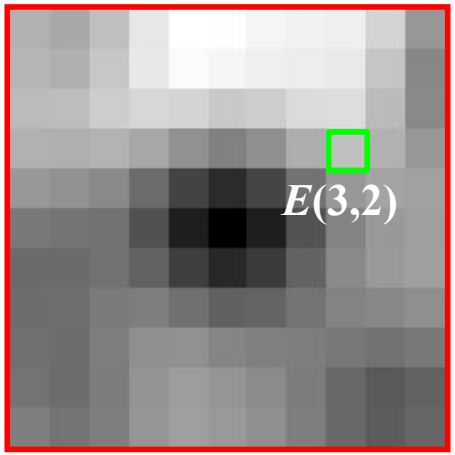
Change in appearance of window *W* for the shift [*u,v*]:

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

$I(x, y)$



$E(u, v)$



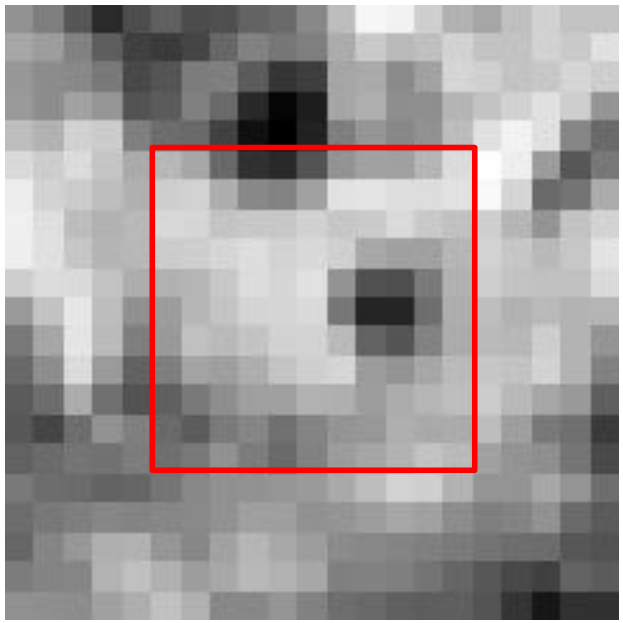
Slide: S. Lazebnik

Corner Detection: Mathematics

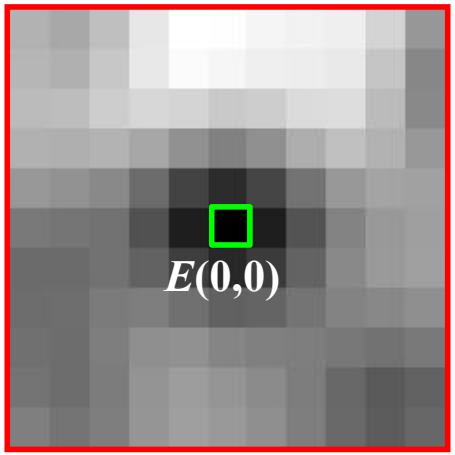
Change in appearance of window *W* for the shift [*u,v*]:

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

$I(x, y)$



$E(u, v)$



Slide: S. Lazebnik

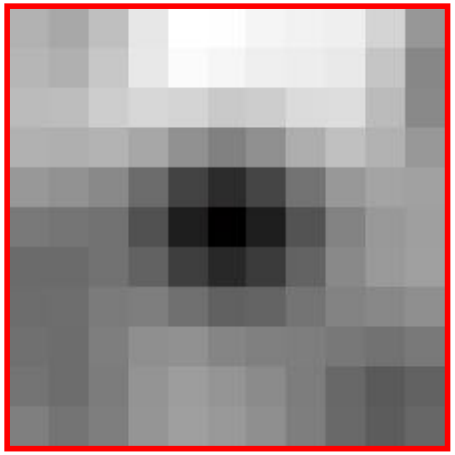
Corner Detection: Mathematics

Change in appearance of window *W* for the shift [*u,v*]:

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

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

$E(u, v)$



Slide: S. Lazebnik

Corner Detection: Mathematics

- First-order Taylor approximation for small motions [*u*, *v*]:

$$I(x+u, y+v) = I(x,y) + I_x u + I_y v + \text{higher order terms}$$

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

$$= I(x,y) + \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

- Let's plug this into

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

Slide: S. Lazebnik

Corner Detection: Mathematics

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

$$\approx \sum_{(x,y)\in W} [I(x,y) + \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} - I(x,y)]^2$$

$$= \sum_{(x,y)\in W} \left(\begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \right)^2$$

$$= \sum_{(x,y)\in W} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

Slide: S. Lazebnik

Corner Detection: Mathematics

The quadratic approximation simplifies to

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

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

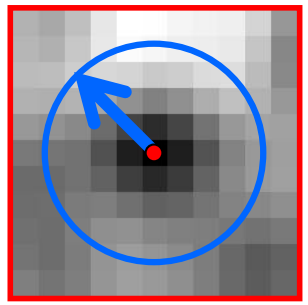
$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Slide: S. Lazebnik

Interpreting the second moment matrix

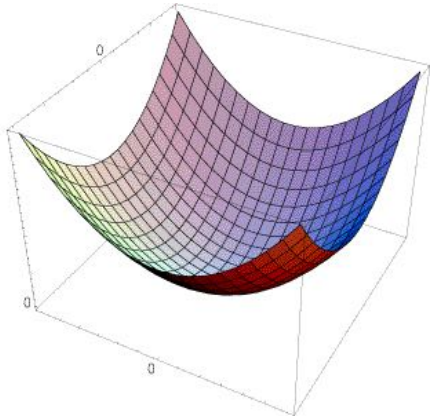
- The surface $E(u,v)$ is locally approximated by a quadratic form. Let's try to understand its shape.
  - Specifically, in which directions does it have the smallest/greatest change?

$E(u, v)$



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

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



Slide: S. Lazebnik

Interpreting the second moment matrix

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

$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

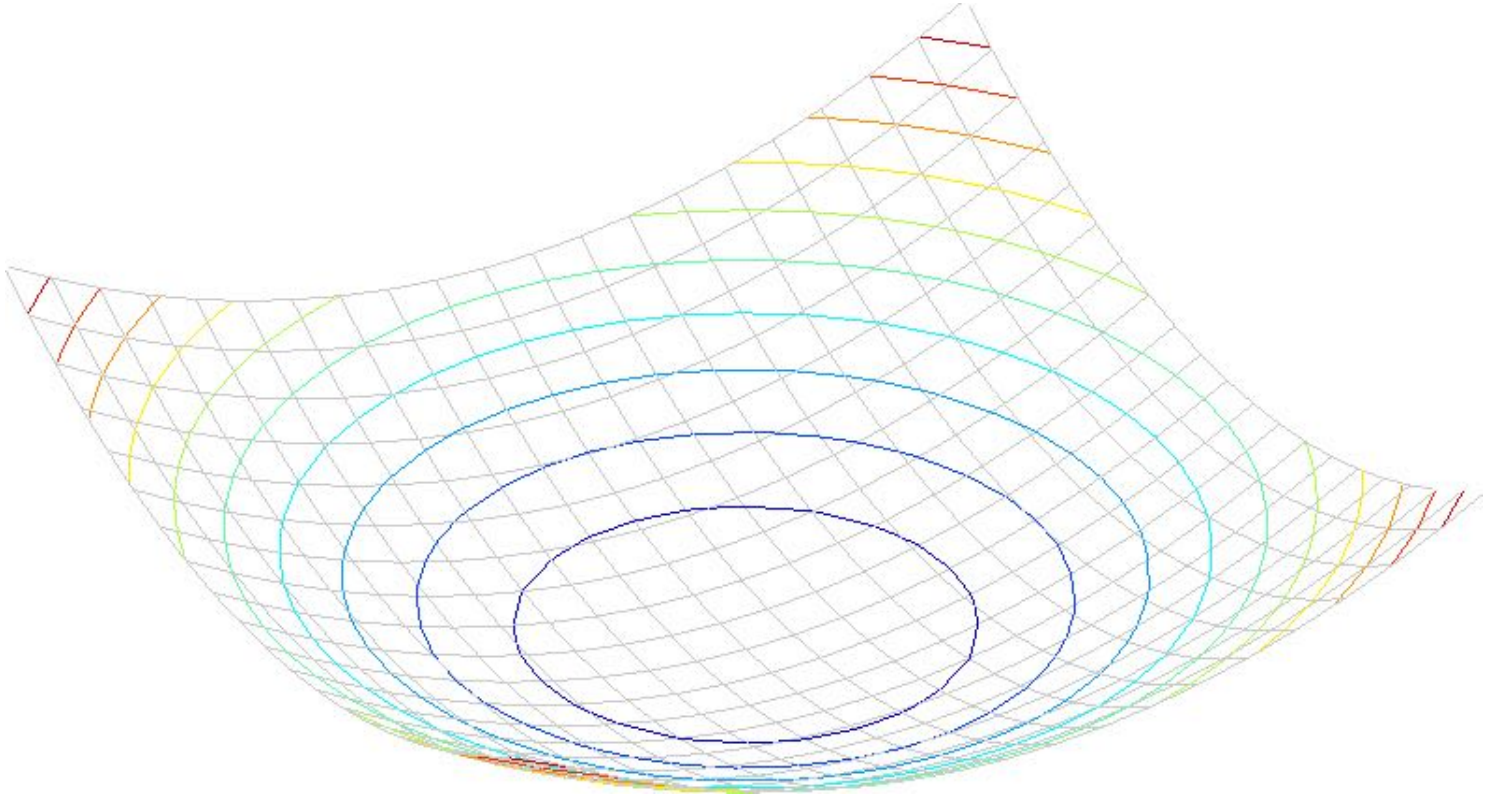
If either *a* or *b* is close to 0, then this is **not** a corner,
so look for locations where both are large.

Slide: S. Lazebnik

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.

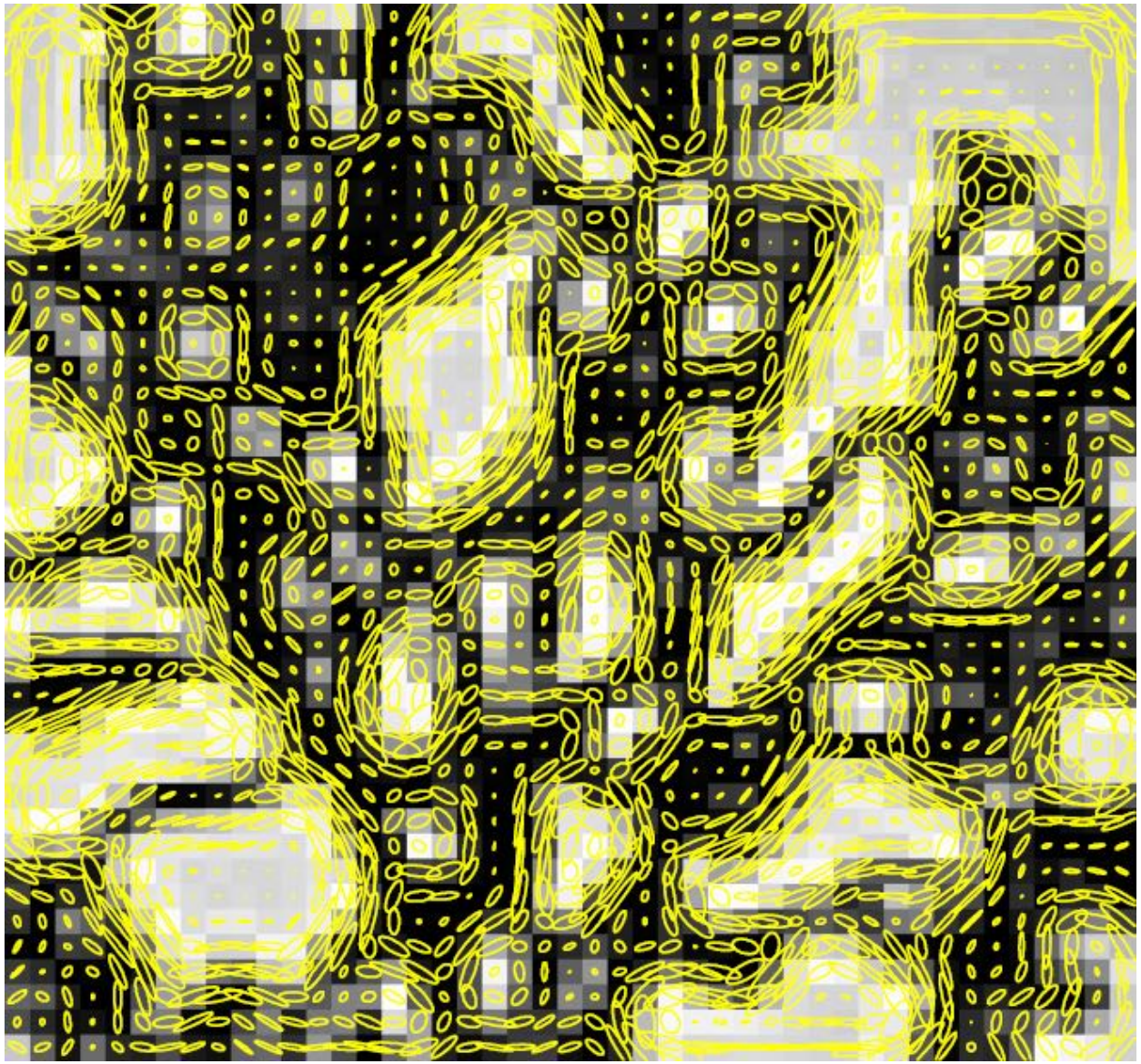


Slide: S. Lazebnik

Visualization of second moment matrices

Slide: S. Lazebnik

Visualization of second moment matrices



Slide: S. Lazebnik

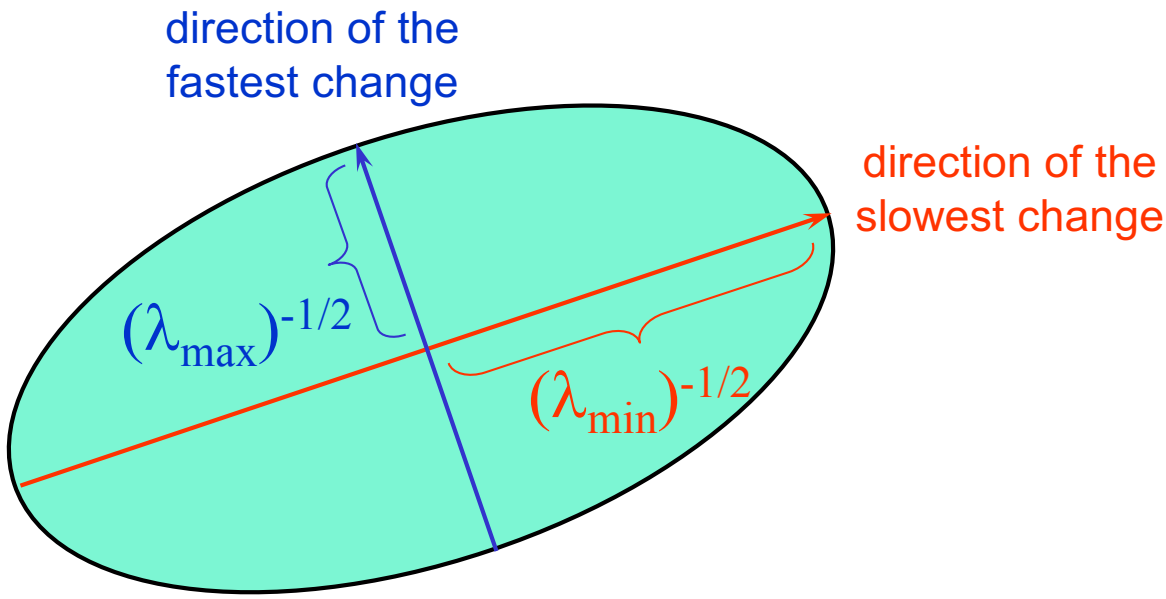
Interpreting the second moment matrix

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

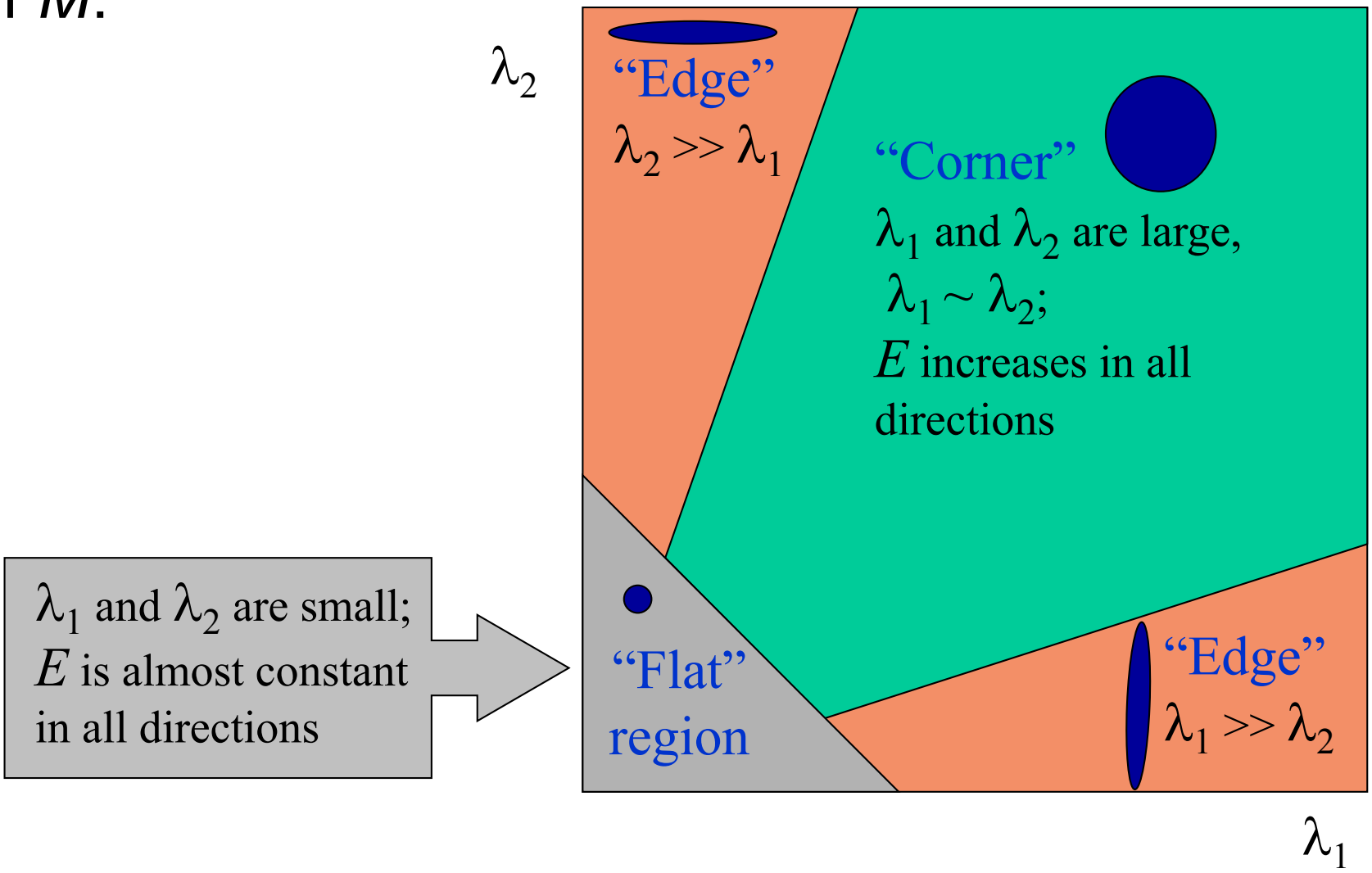
The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by R



Slide: S. Lazebnik

Interpreting the eigenvalues

Classification of image points using eigenvalues of *M*:

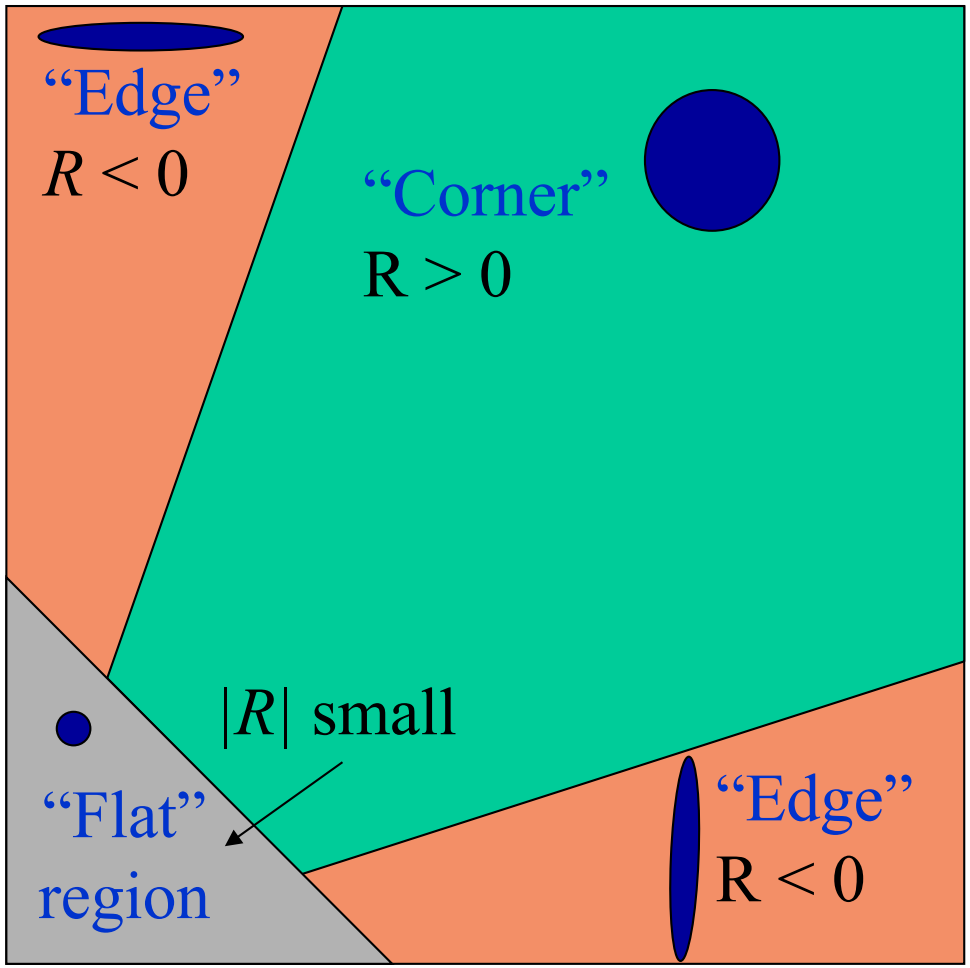


Slide: S. Lazebnik

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)



Slide: S. Lazebnik

Harris detector: Steps

1. Compute Gaussian derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel
3. Compute corner response function *R*
4. Threshold *R*
5. Find local maxima of response function (nonmaximum suppression)

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

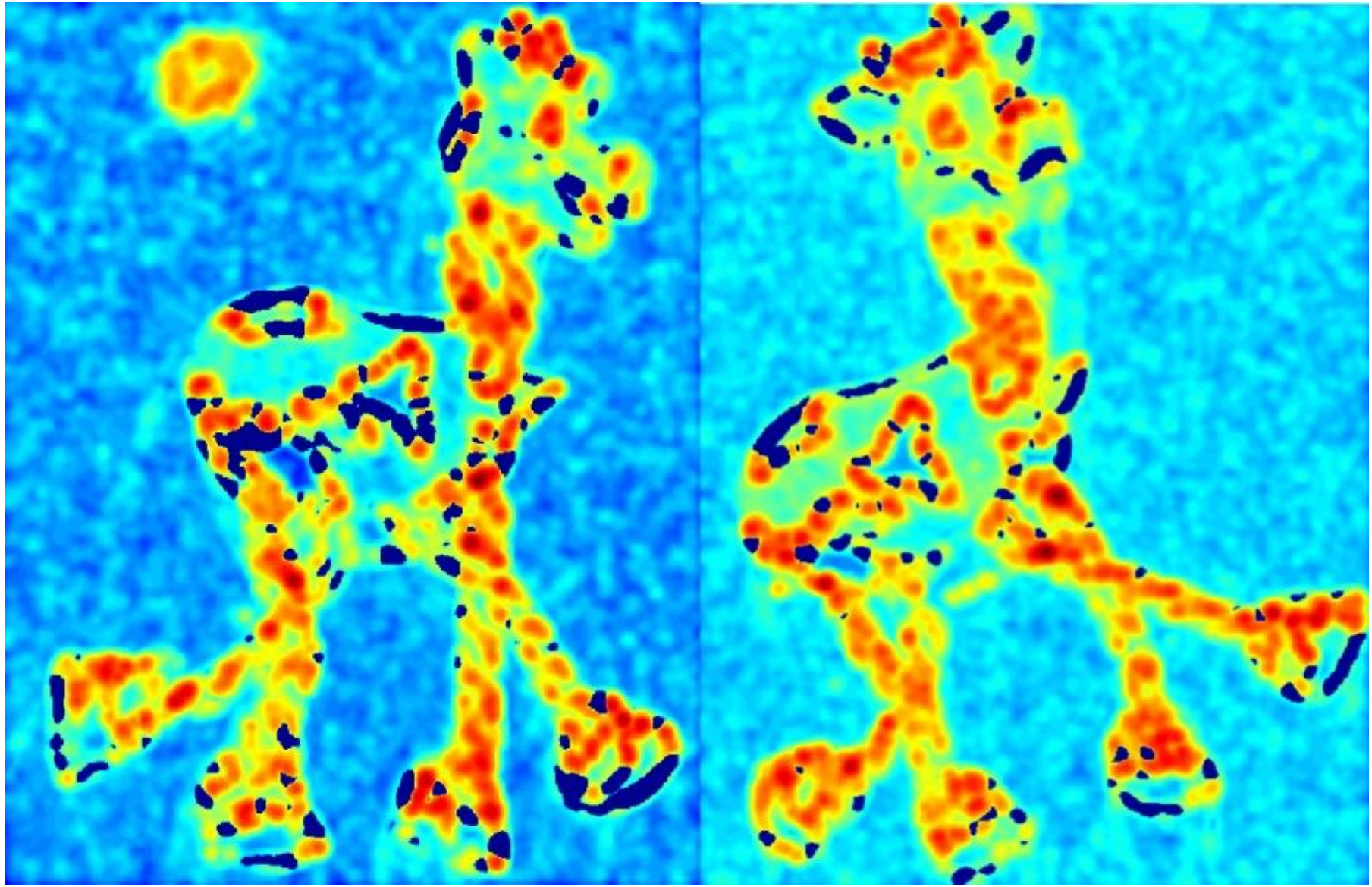
Slide: S. Lazebnik

Harris Detector: Steps

Slide: S. Lazebnik

Harris Detector: Steps

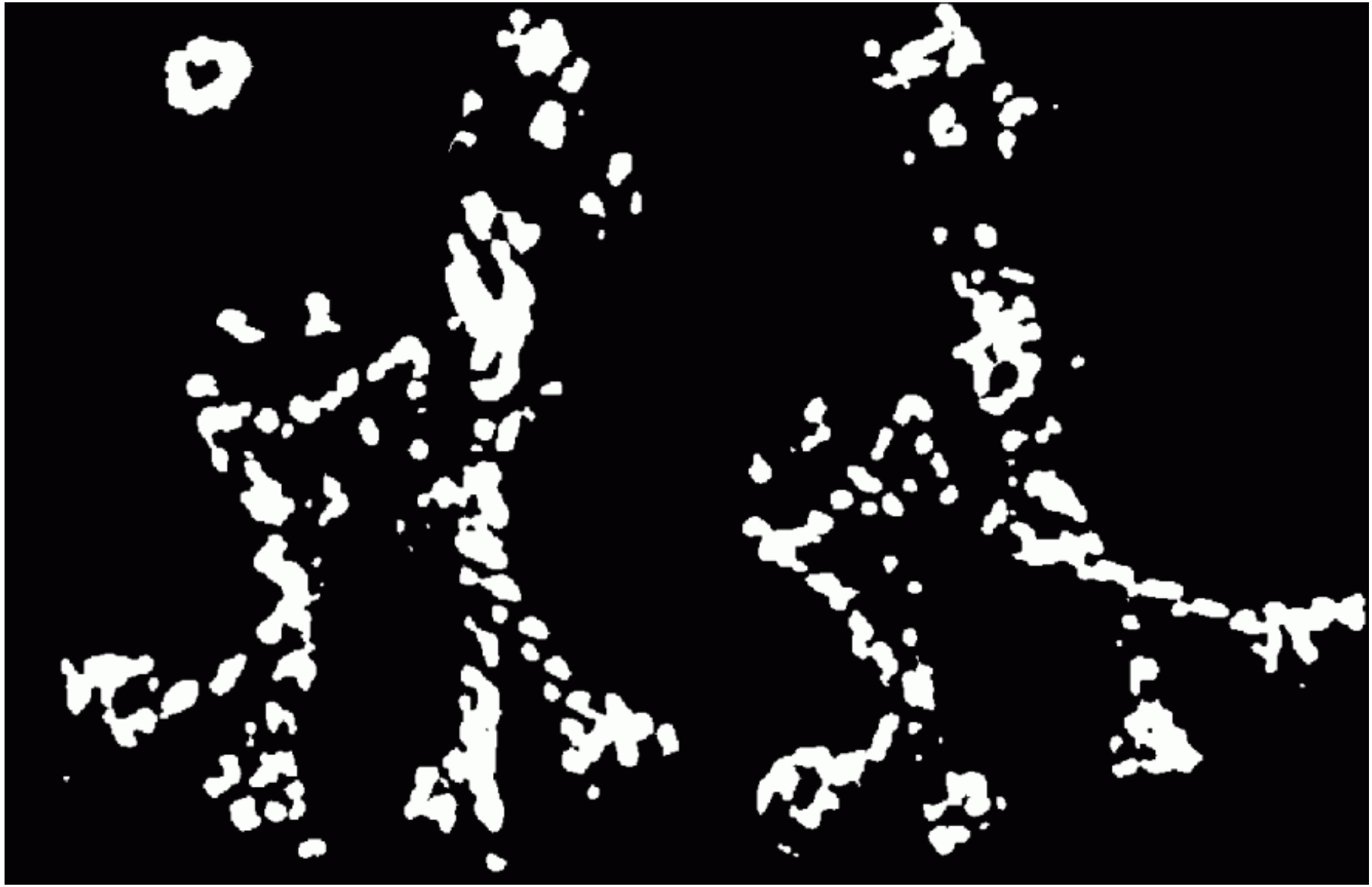
Compute corner response R



Slide: S. Lazebnik

Harris Detector: Steps

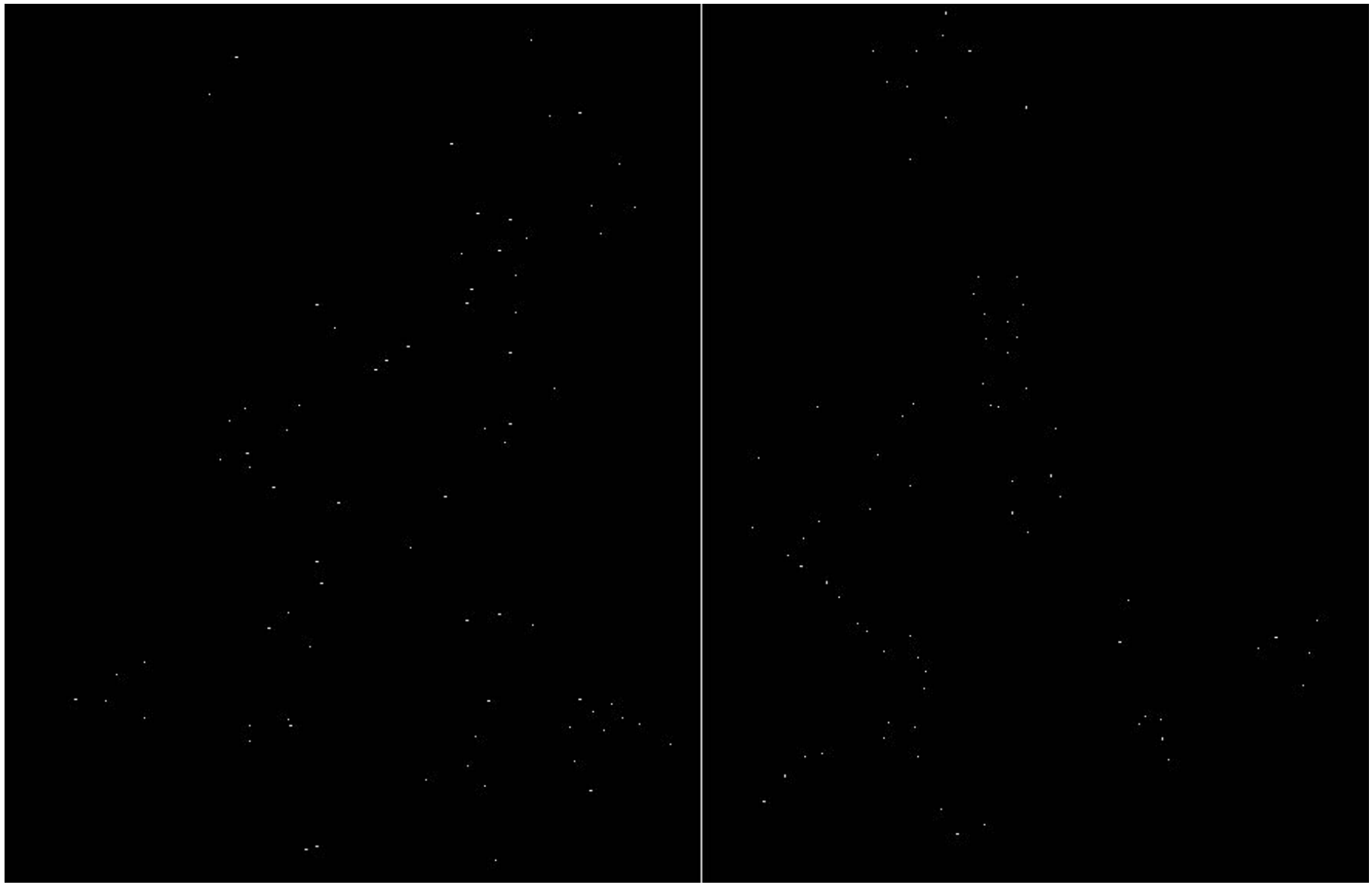
Find points with large corner response: R>threshold



Slide: S. Lazebnik

Harris Detector: Steps

Take only the points of local maxima of R



Slide: S. Lazebnik

Harris Detector: Steps

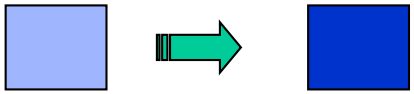
Slide: S. Lazebnik

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

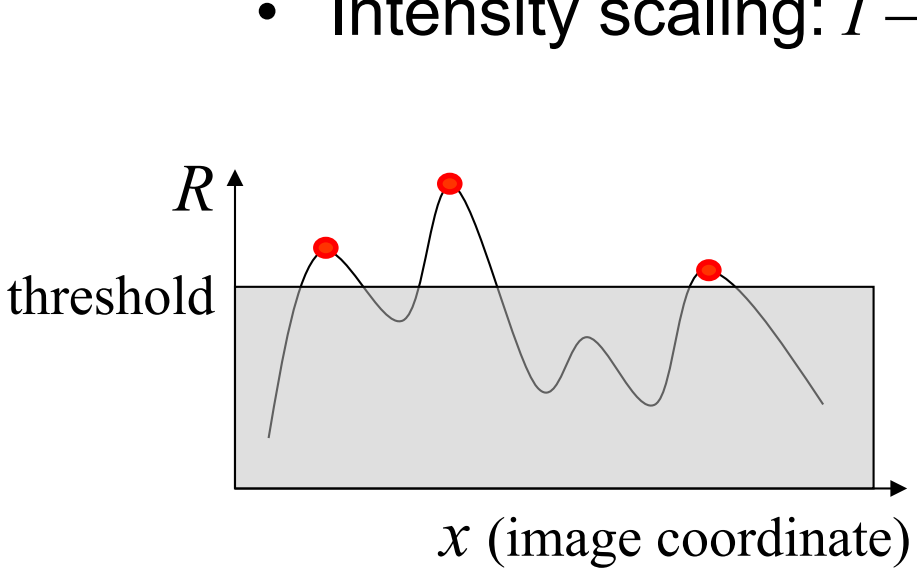
Slide: S. Lazebnik

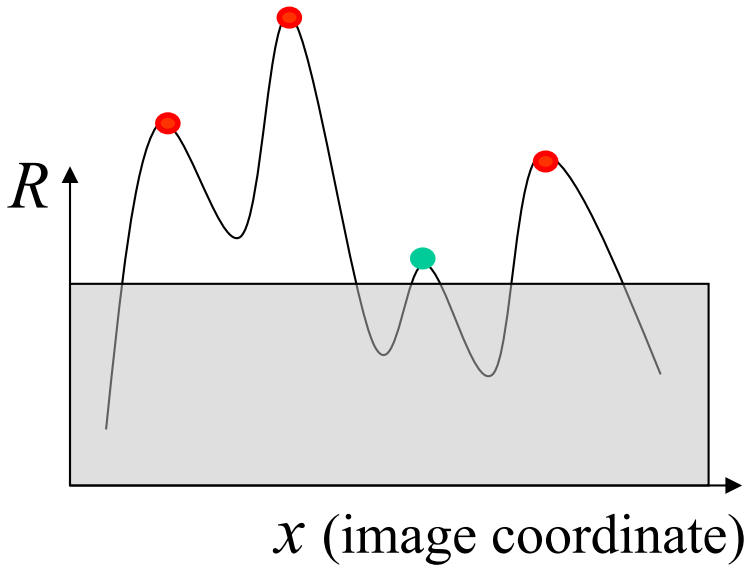
Affine intensity change



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

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

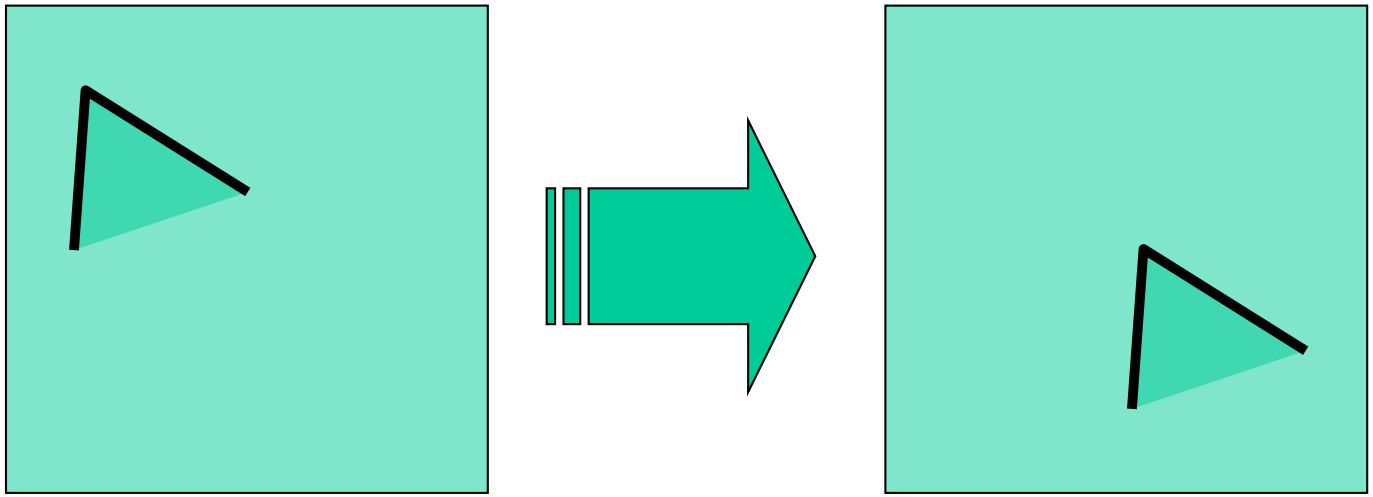




Partially invariant to affine intensity change

Slide: S. Lazebnik

Image translation

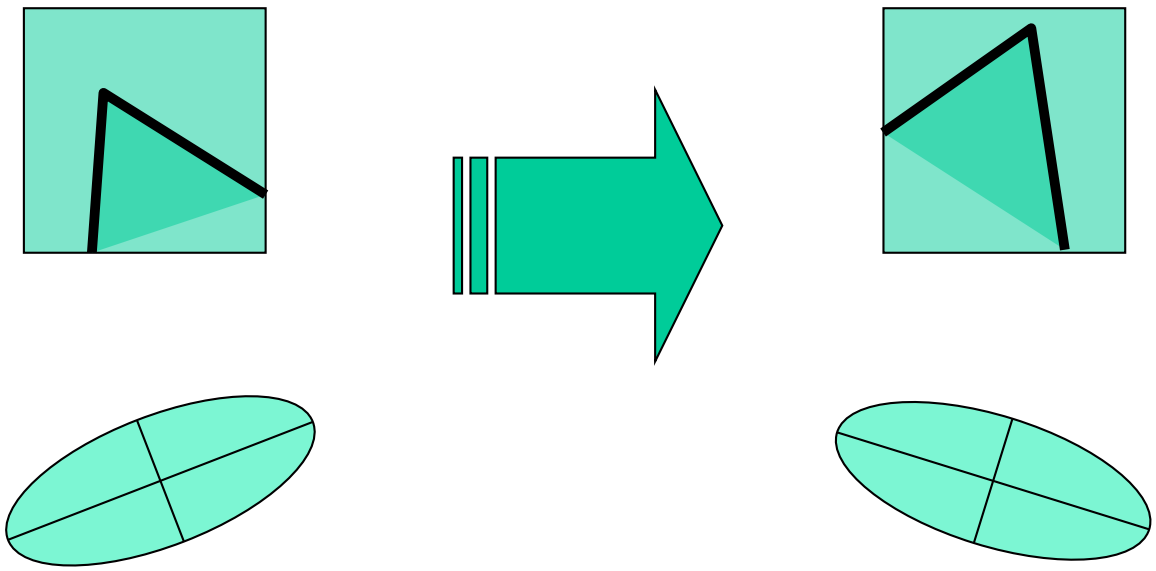


- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation

Slide: S. Lazebnik

Image rotation

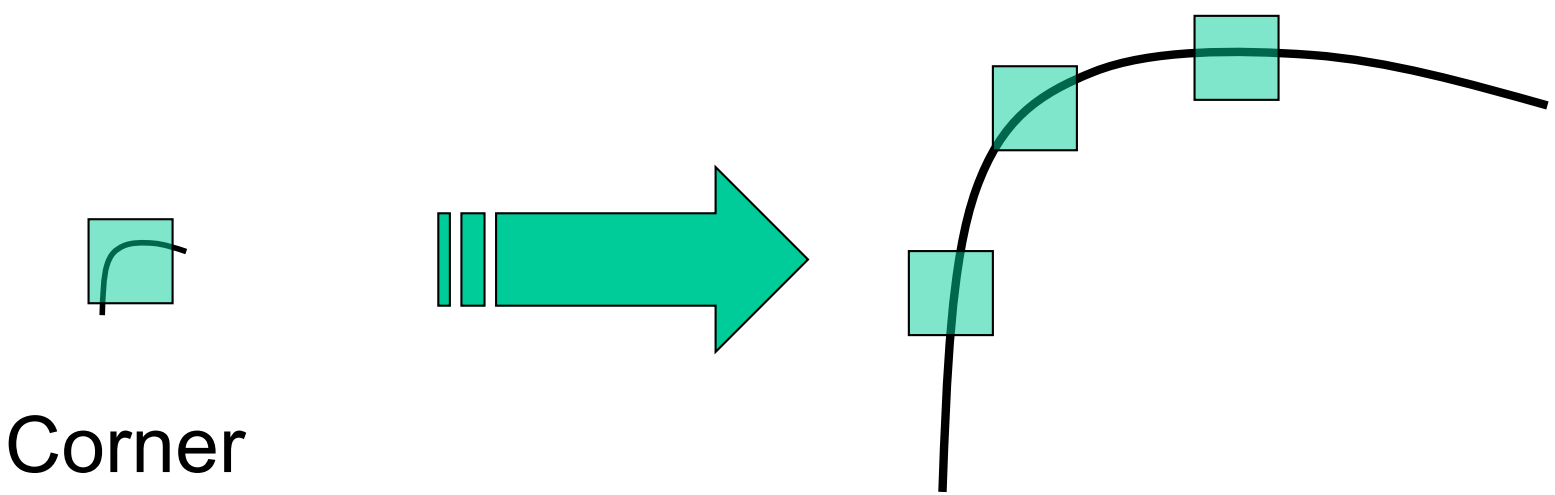


Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation

Slide: S. Lazebnik

Scaling



Corner

All points will be classified as edges

Corner location is not covariant to scaling!

Slide: S. Lazebnik

Outline: feature detection

Edges

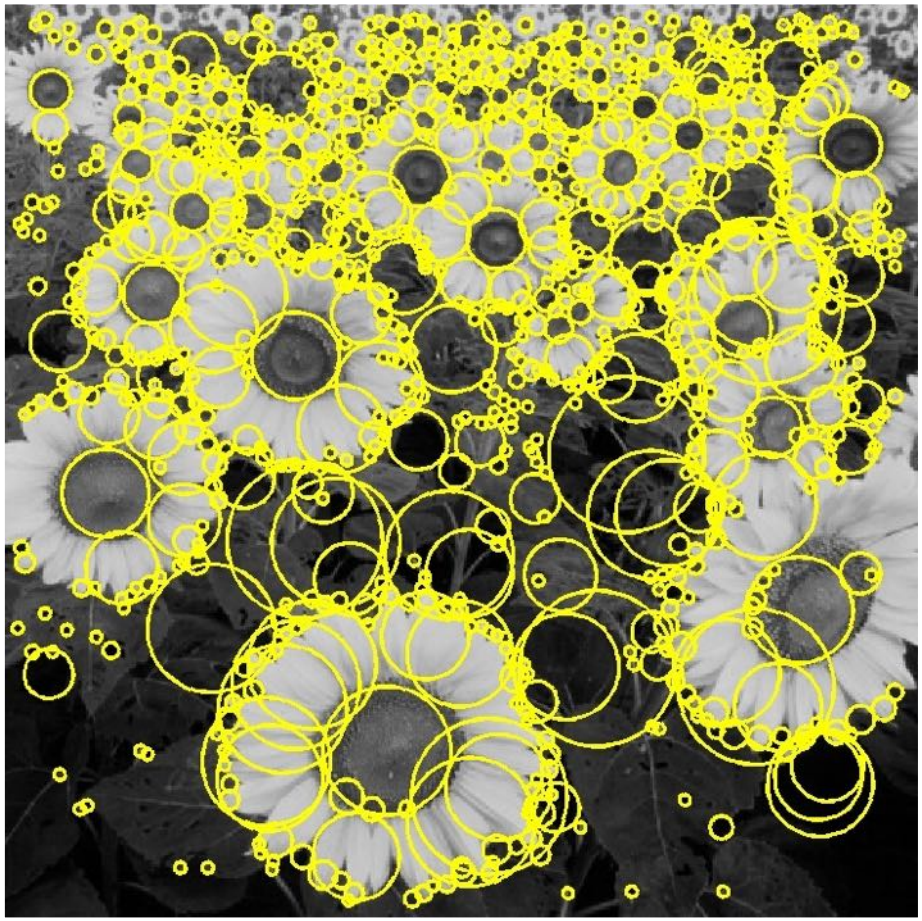
Corners

Blobs

[Contours]

[Regions]

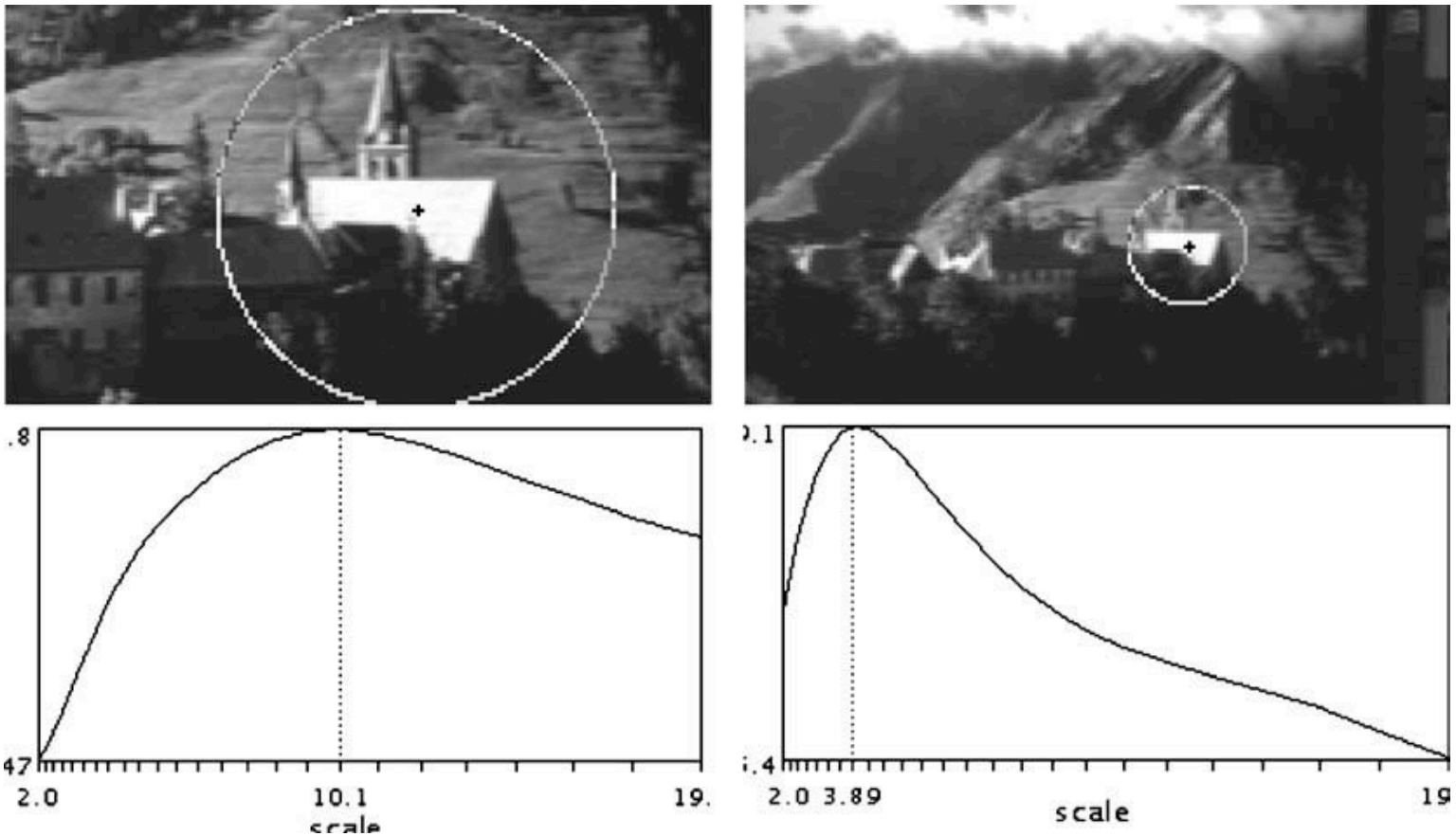
Blob detection



Slide: S. Lazebnik

Feature detection with scale selection

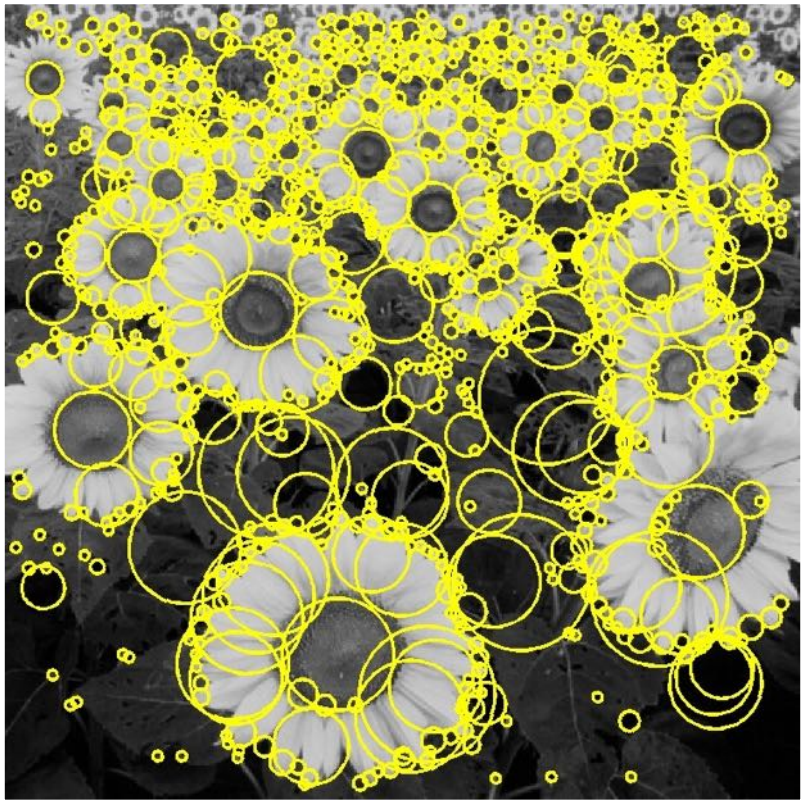
- We want to extract features with characteristic scale that is *covariant* with the image transformation

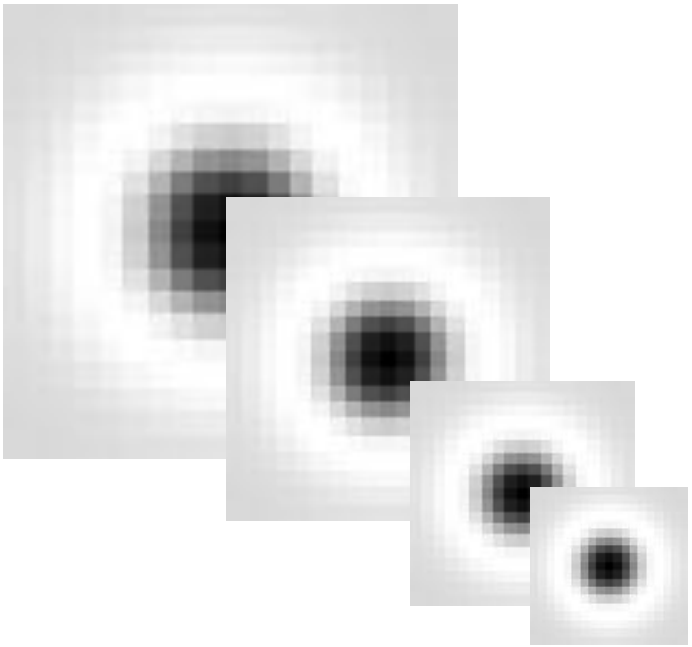


Slide: S. Lazebnik

Blob detection: basic idea

- To detect blobs, convolve the image with a “blob filter” at multiple scales and look for maxima of filter response in the resulting *scale space*

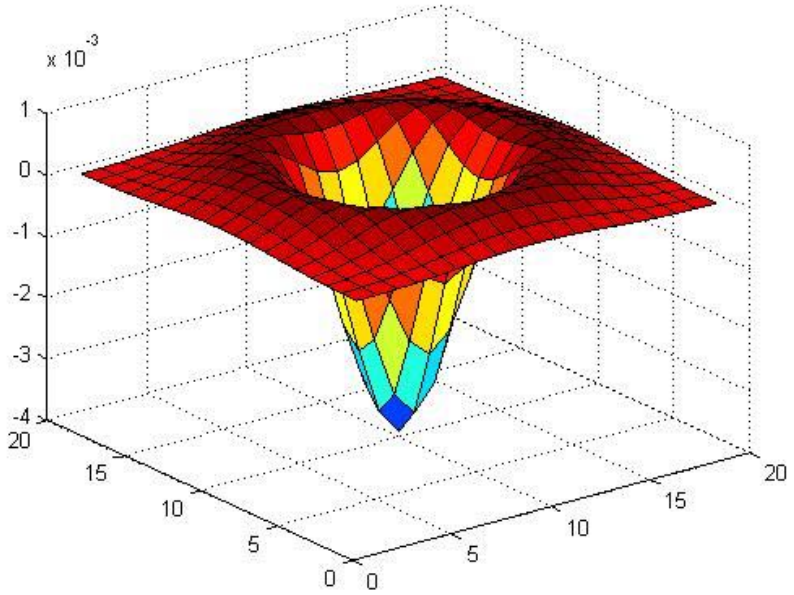


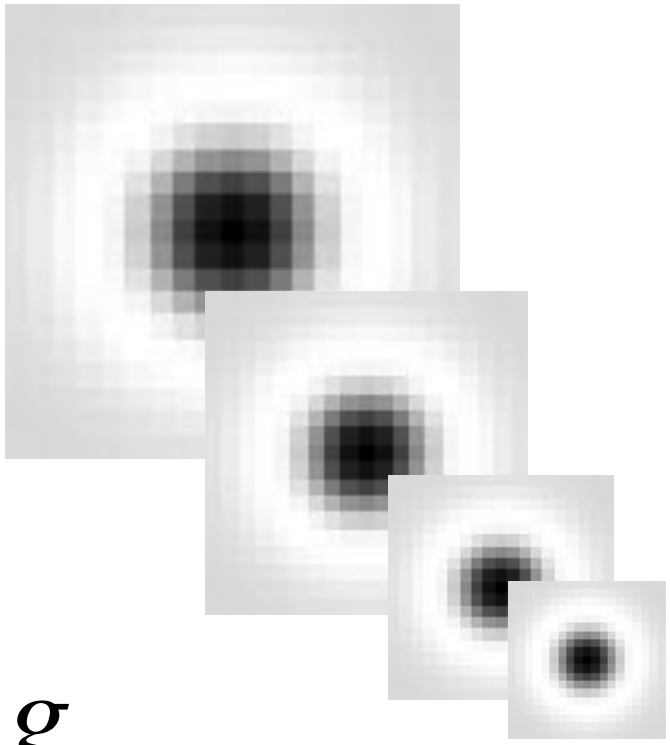


Slide: S. Lazebnik

Blob filter

Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D



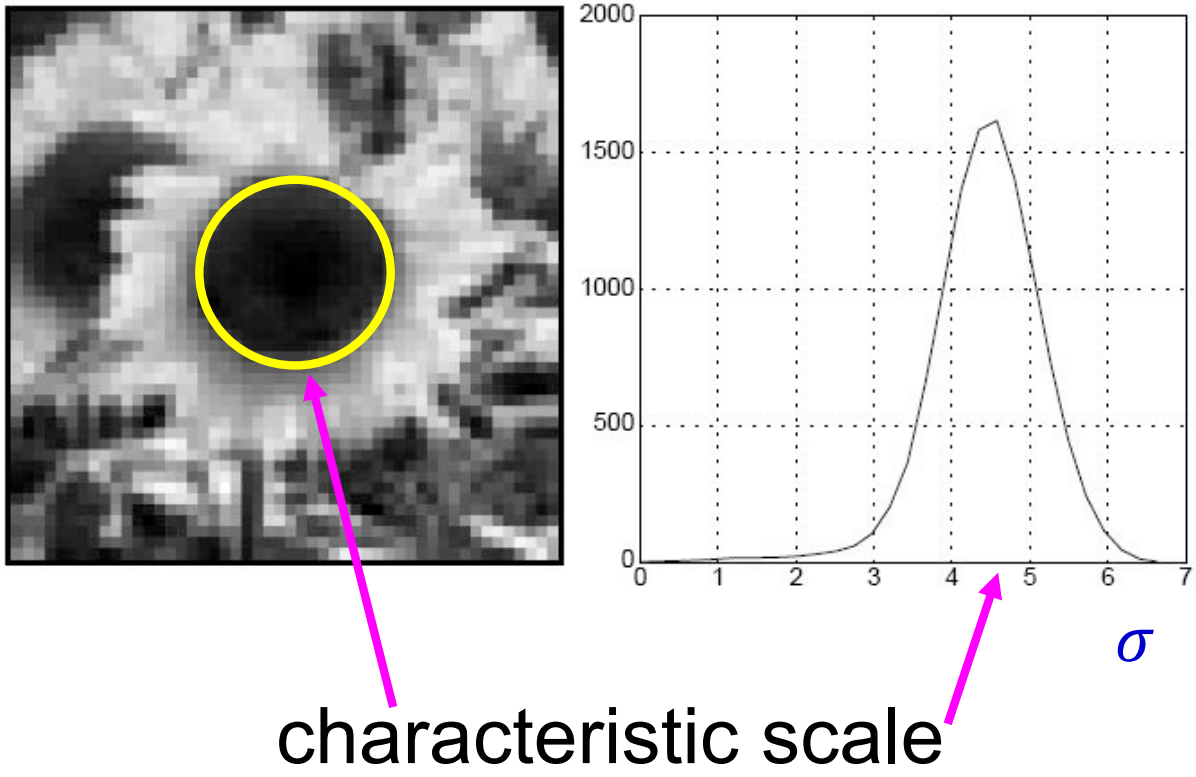


$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

Slide: S. Lazebnik

Characteristic scale

- We define the characteristic scale of a blob as the scale that produces peak of Laplacian response in the blob center



T. Lindeberg (1998). "Feature detection with automatic scale selection." *International Journal of Computer Vision* **30** (2): pp 77--116.

Slide: S. Lazebnik

Scale-space blob detector: Example

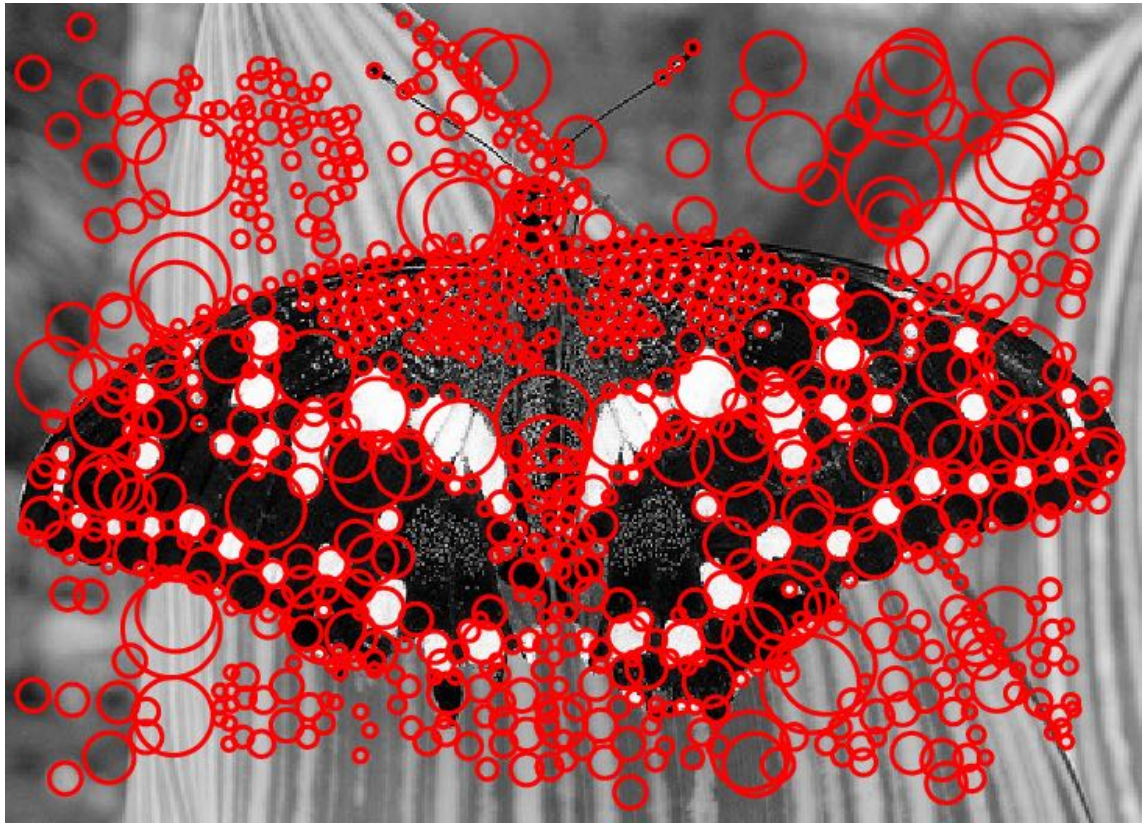
Slide: S. Lazebnik

Scale-space blob detector: Example

sigma = 11.9912

Slide: S. Lazebnik

Scale-space blob detector: Example



Slide: S. Lazebnik

Outline: feature detection

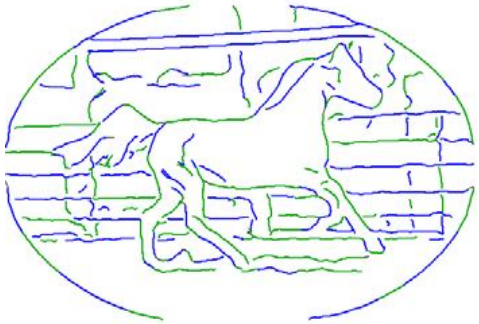
Edges

Corners

Blobs

[Contours]

[Regions]



Contours/lines

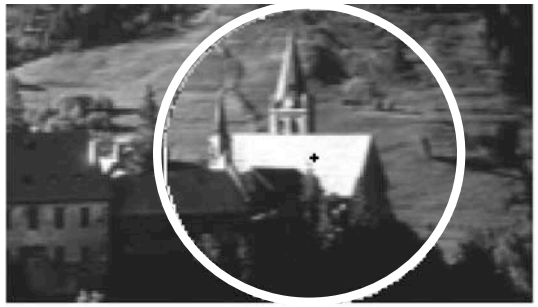
Mi-points, angles

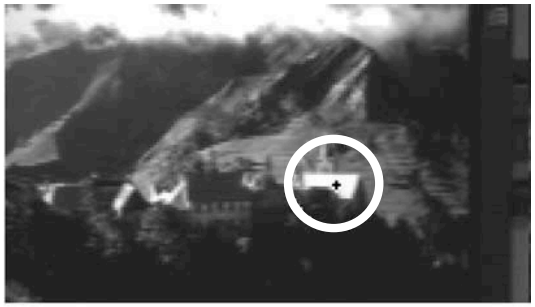
Image regions

[Felzenszwalb et al., 2014]

From feature detection to feature description

- Scaled and rotated versions of the same neighborhood will give rise to blobs that are related by the same transformation
- What to do if we want to compare the appearance of these image regions?
  - *Normalization*: transform these regions into same-size circles
  - Problem: rotational ambiguity

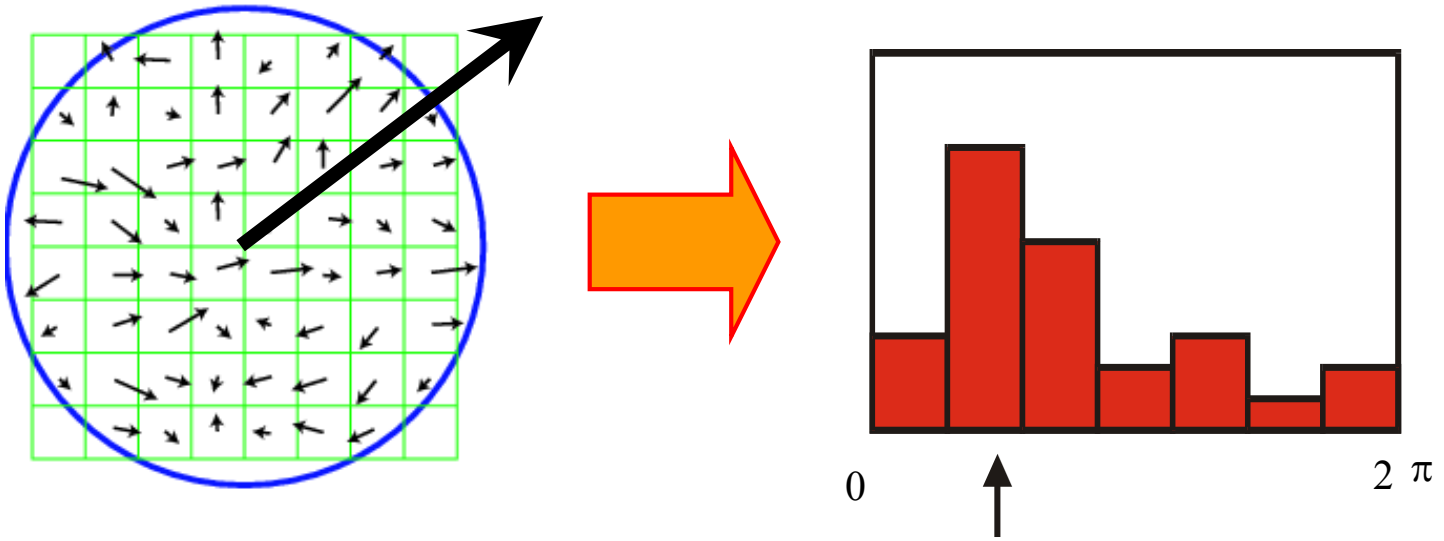




Slide: S. Lazebnik

Eliminating rotation ambiguity

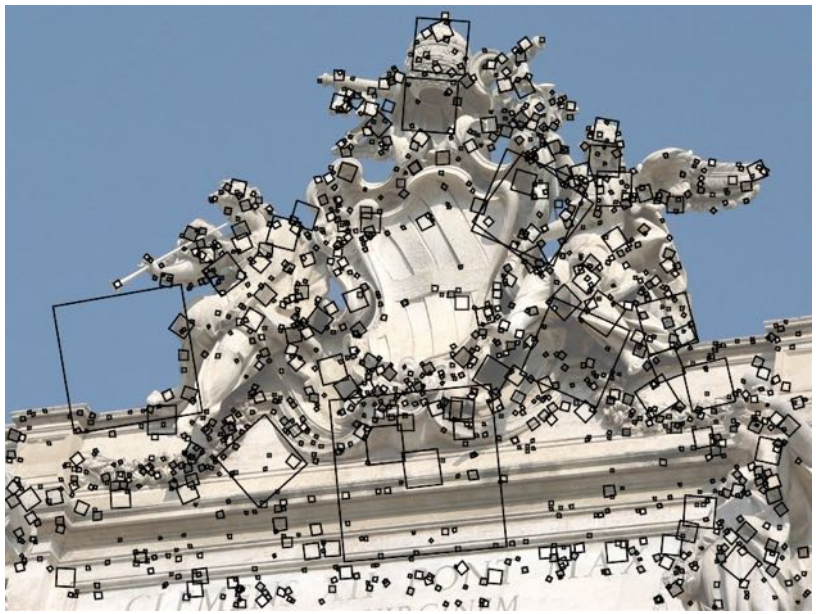
- To assign a unique orientation to circular image windows:
  - Create histogram of local gradient directions in the patch
  - Assign canonical orientation at peak of smoothed histogram



Slide: S. Lazebnik

SIFT features

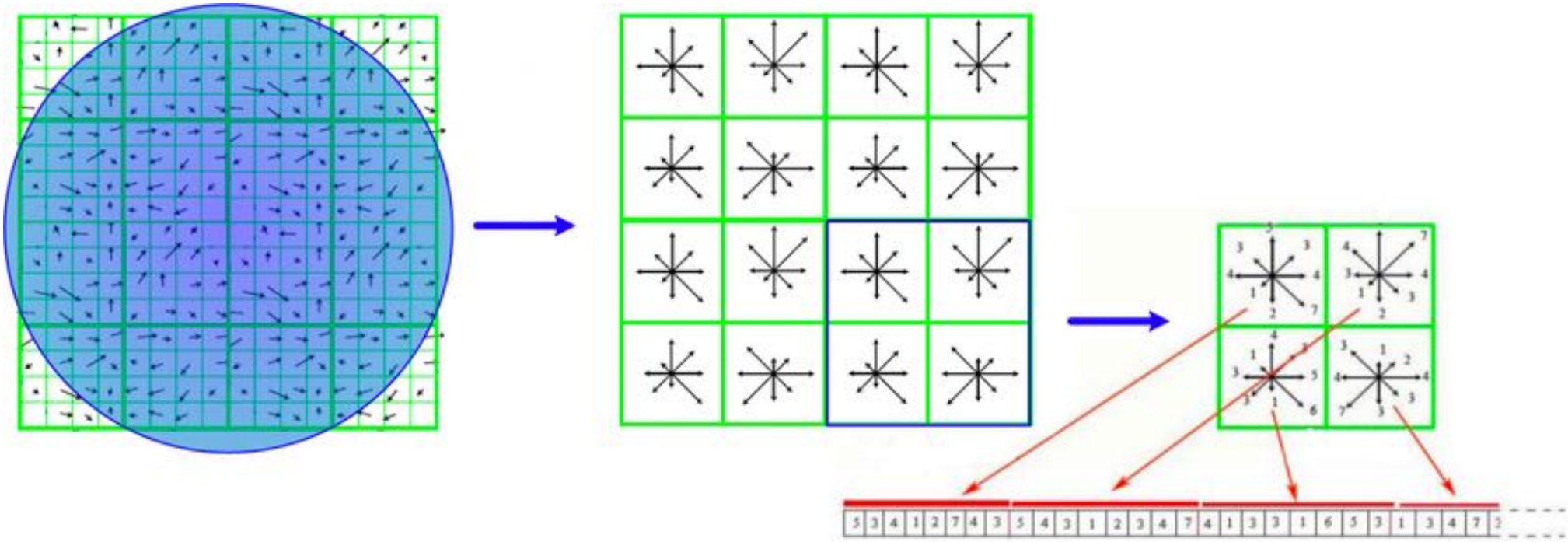
- Detected features with characteristic scales and orientations:



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

Slide: S. Lazebnik

SIFT descriptors



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

Slide: S. Lazebnik

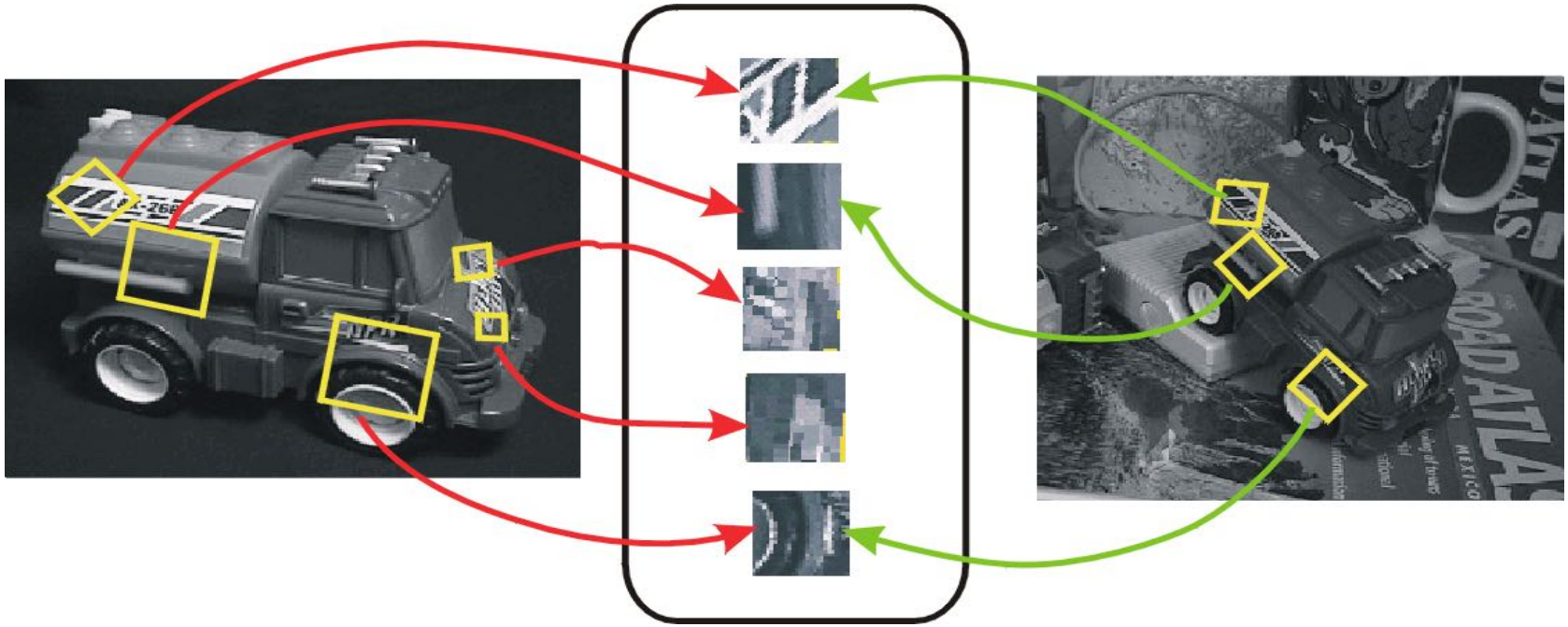
Invariance vs. covariance

Invariance:

- features(transform(image)) = features(image)

Covariance:

- features(transform(image)) = transform(features(image))

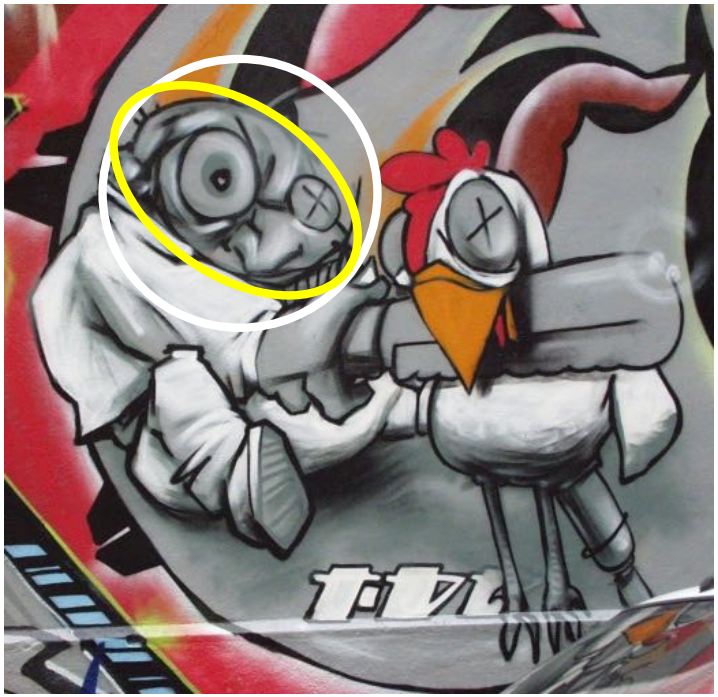


Covariant detection => invariant description

Slide: S. Lazebnik

Affine adaptation

- Affine transformation approximates viewpoint changes for roughly planar objects and roughly orthographic cameras



Slide: S. Lazebnik

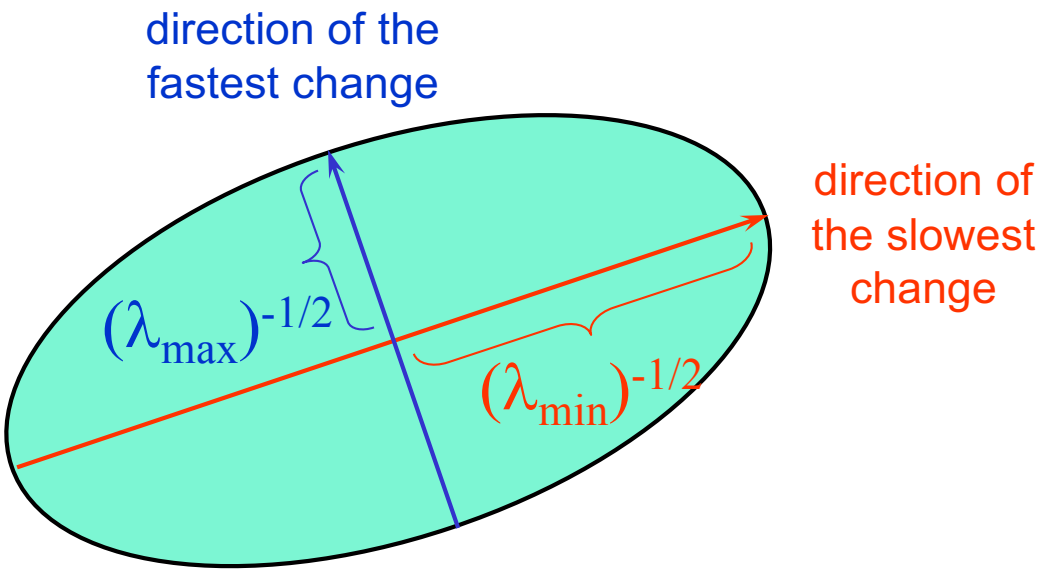
Affine adaptation

Consider the second moment matrix of the window containing the blob:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

Recall:

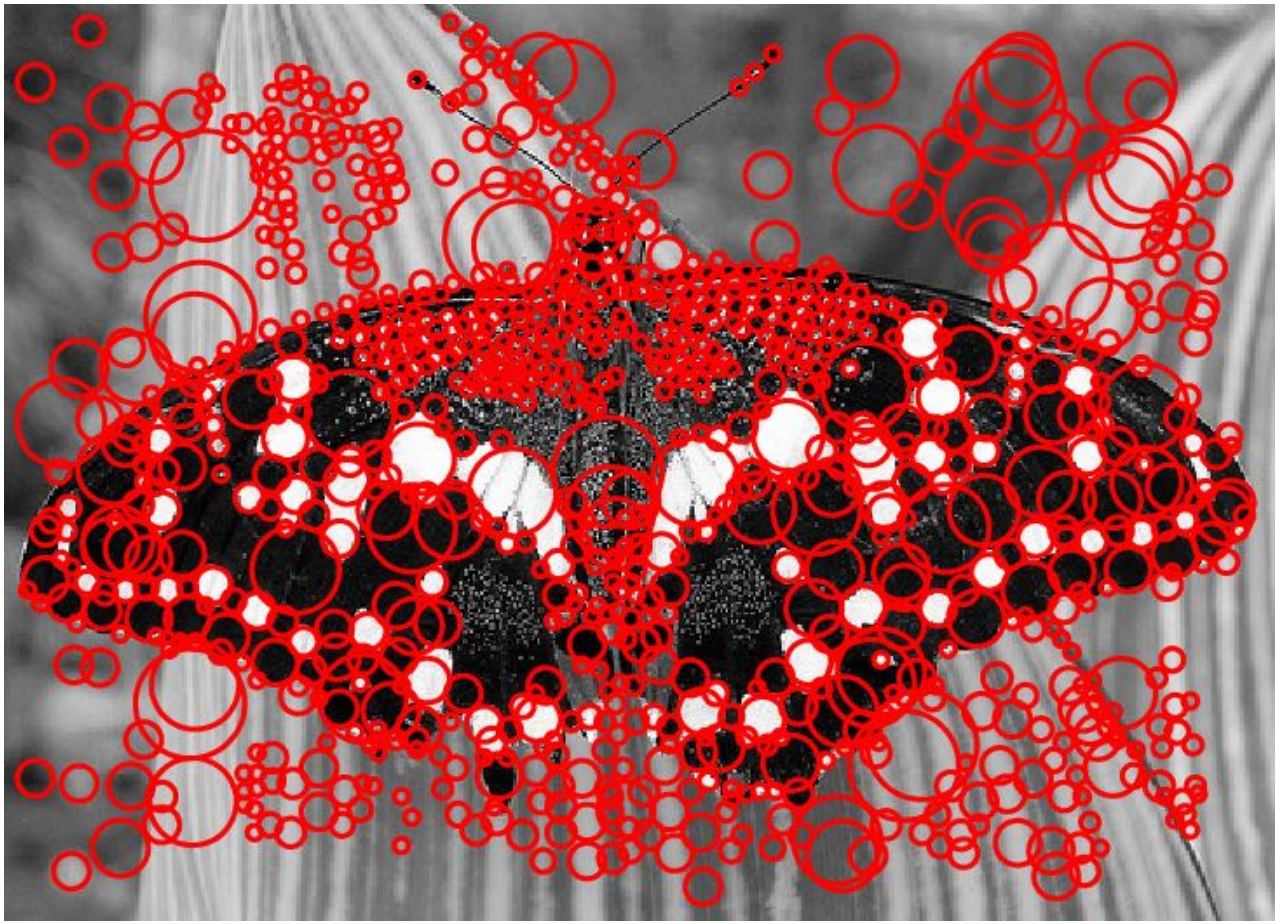
$$[u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$



This ellipse visualizes the "characteristic shape" of the window

Slide: S. Lazebnik

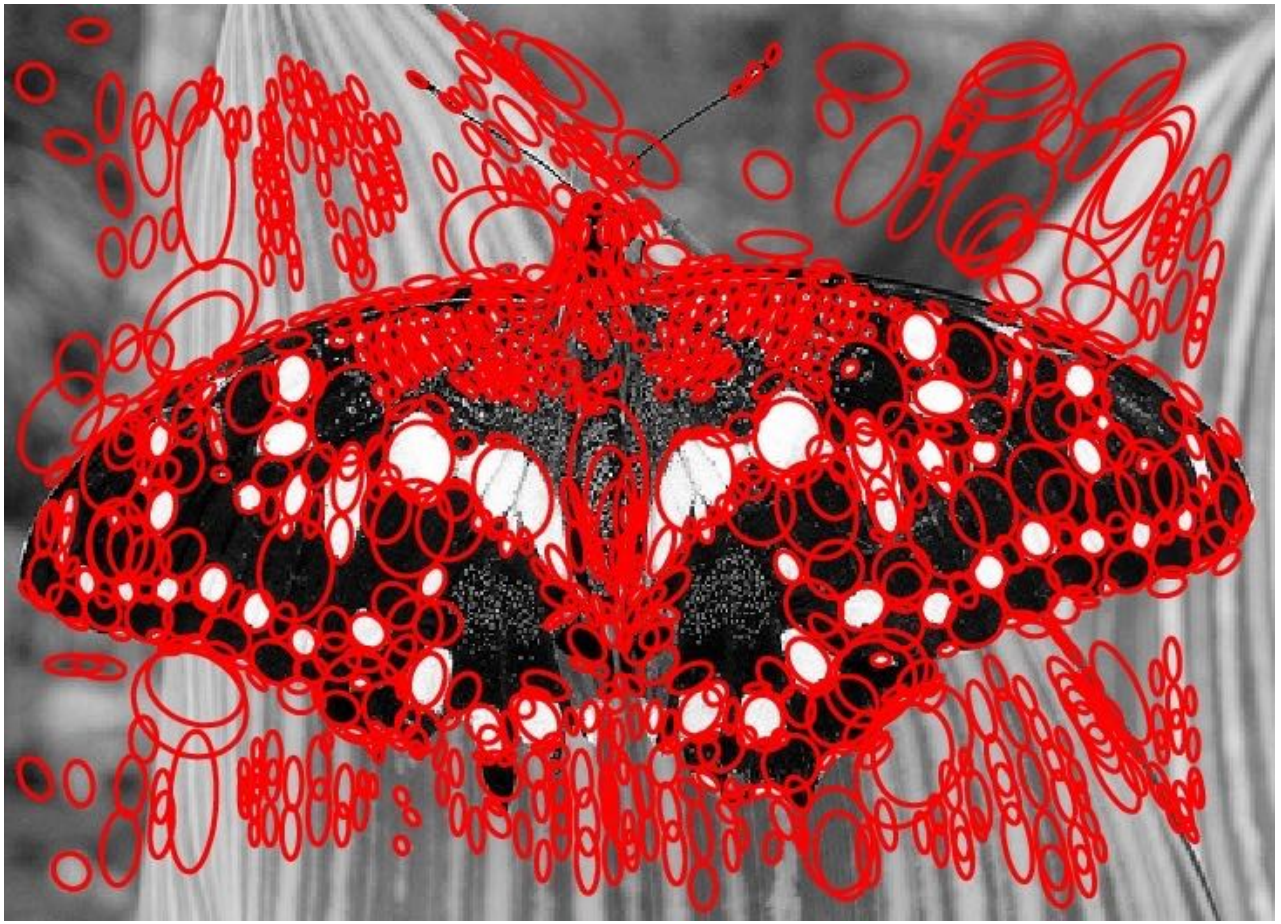
Affine adaptation example



Scale-invariant regions (blobs)

Slide: S. Lazebnik

Affine adaptation example



Affine-adapted blobs

Slide: S. Lazebnik

Summary : feature detection

Edges – convolution with Gaussian derivatives

Corners – eigenvalues of second moment matrix

Blobs – Laplacian of Gaussian at different scales

Approach

1. **Pre-processing:**
   - Detect local features.
   - Extract descriptor for each feature.

2. **Matching:** Establish tentative (putative) correspondences based on local appearance of individual features (their descriptors).

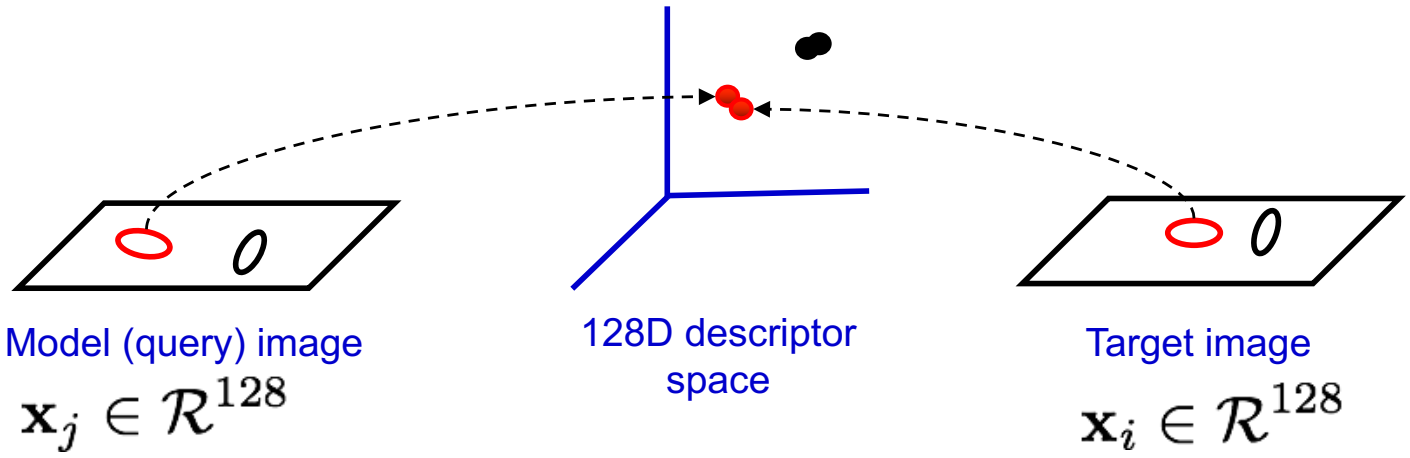
3. **Verification:** Verify matches based on semi-local / global geometric relations.

4. **Current research challenges**

Example I: Two images -“Where is the Graffiti?”

Step 1. Establish tentative correspondence

Establish tentative correspondences between object model image and target image by nearest neighbour matching on SIFT vectors



Need to solve some variant of the "nearest neighbor problem" for all feature vectors, $\mathbf{x}_j \in \mathcal{R}^{128}$, in the query image:

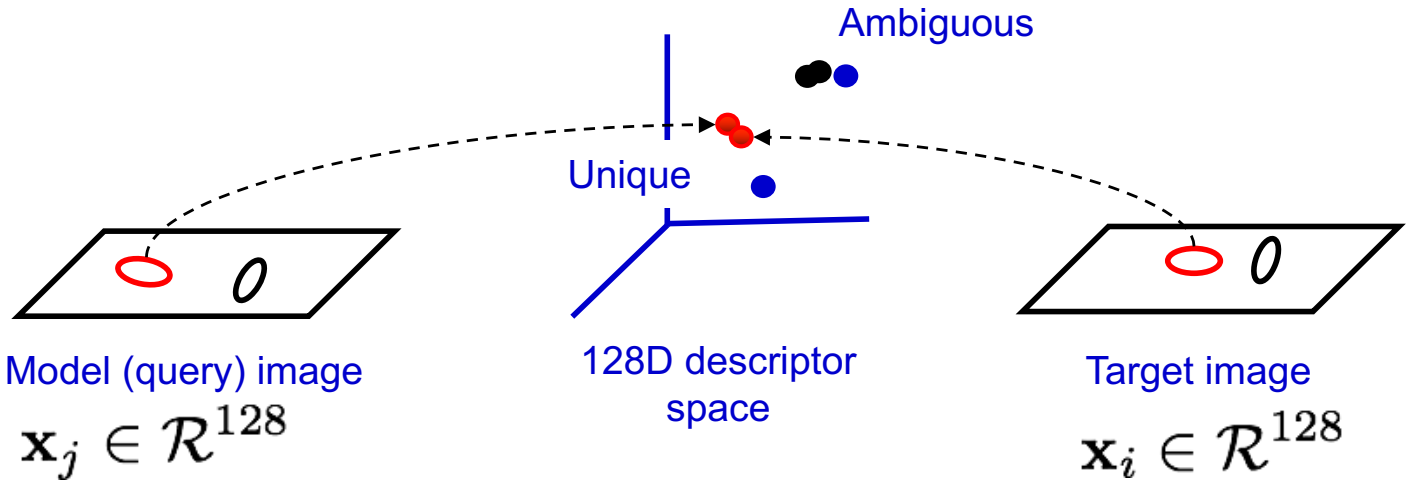
$$\forall j \; NN(j) = \arg\min_i ||\mathbf{x}_i - \mathbf{x}_j||,$$

where, $\mathbf{x}_i \in \mathcal{R}^{128}$, are features in the target image.

Can take a long time if many target images are considered.

Step 1. Establish tentative correspondence

Examine the distance to the 2nd nearest neighbour [Lowe, IJCV 2004]



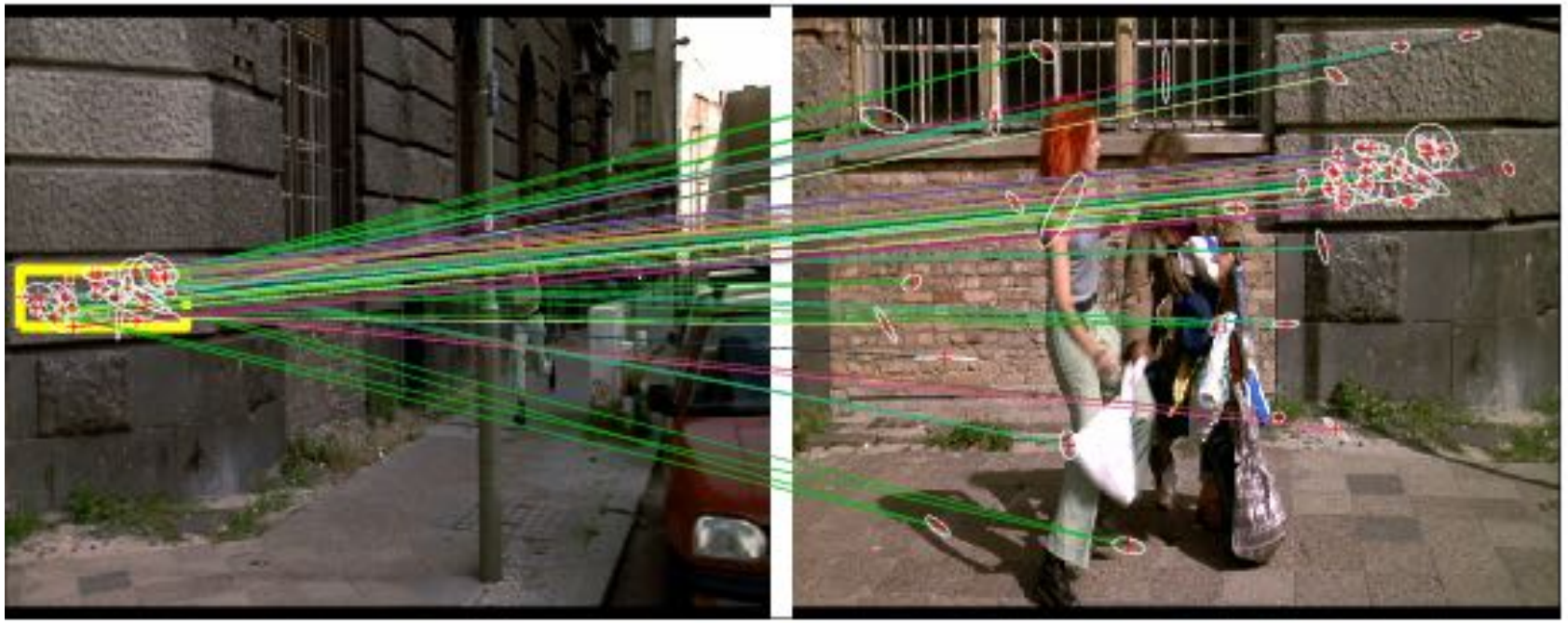
If the 2nd nearest neighbour is much further than the 1st nearest neighbour
Match is more “unique” or discriminative.

Measure this by the ratio: $r = d_{1NN} / d_{2NN}$

r is between 0 and 1
r is small the match is more unique.

Works very well in practice.

Problem with matching on local descriptors alone



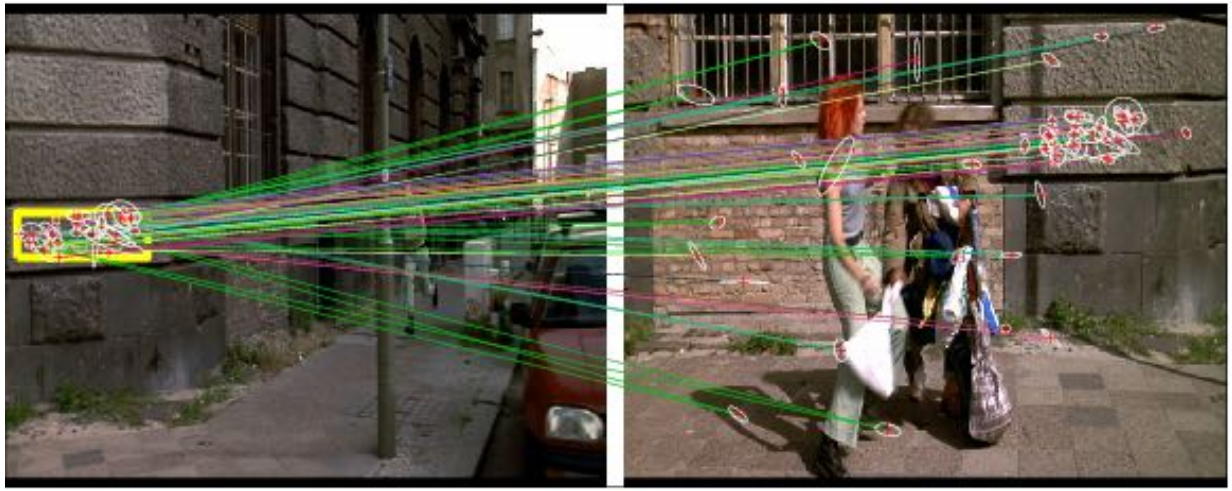
- too much individual invariance
- each region can affine deform independently (by different amounts)
- locally appearance can be ambiguous

Solution: use semi-local and global spatial relations to verify matches.

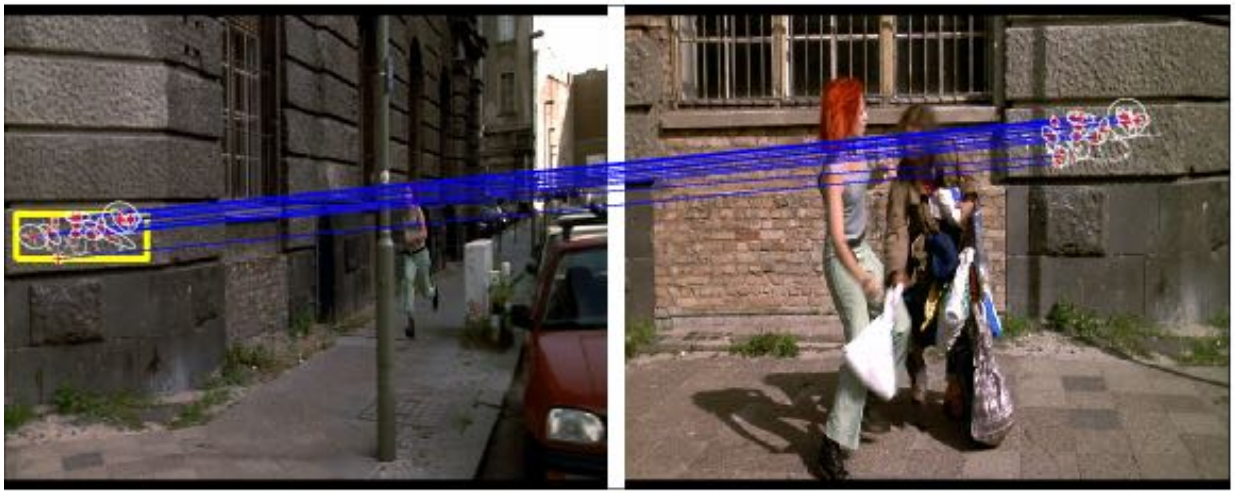
Example I: Two images -“Where is the Graffiti?”

Initial matches

Nearest-neighbor search based on appearance descriptors alone.



After spatial verification



Approach

0. **Pre-processing:**
   - Detect local features.
   - Extract descriptor for each feature.

1. **Matching:** Establish tentative (putative) correspondences based on local appearance of individual features (their descriptors).

2. **Verification:** Verify matches based on semi-local / global geometric relations.

Step 2: Spatial verification (now)

a. Semi-local constraints

Constraints on spatially close-by matches

b. Global geometric relations

Require a consistent global relationship between all matches

Semi-local constraints: Example I. – neighbourhood consensus

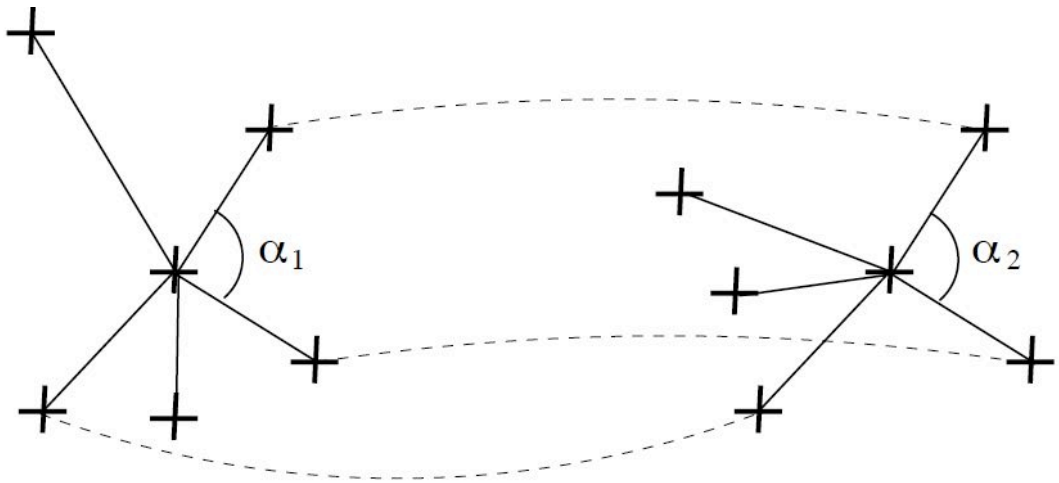


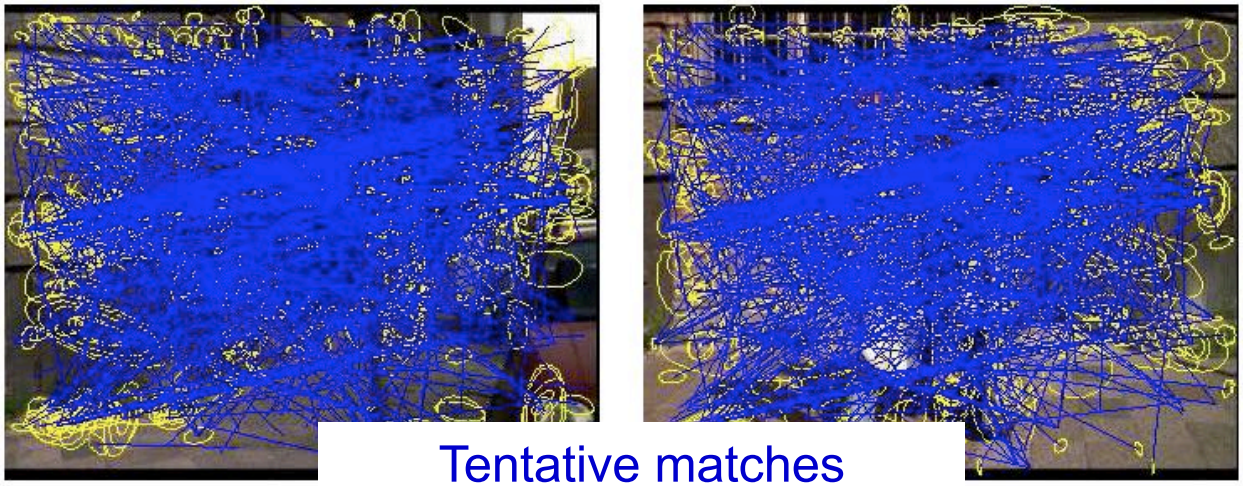
Fig. 4. Semi-local constraints : neighbours of the point have to match and angles have to correspond. Note that not all neighbours have to be matched correctly.

[Schmid&Mohr, PAMI 1997]

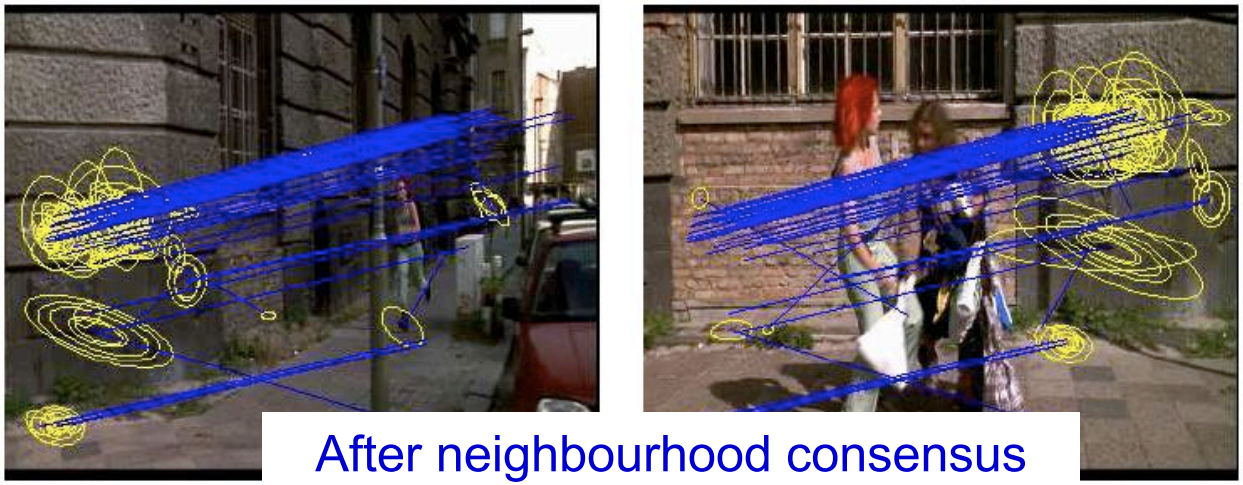
Semi-local constraints:

Example I. – neighbourhood consensus

Original images



Tentative matches



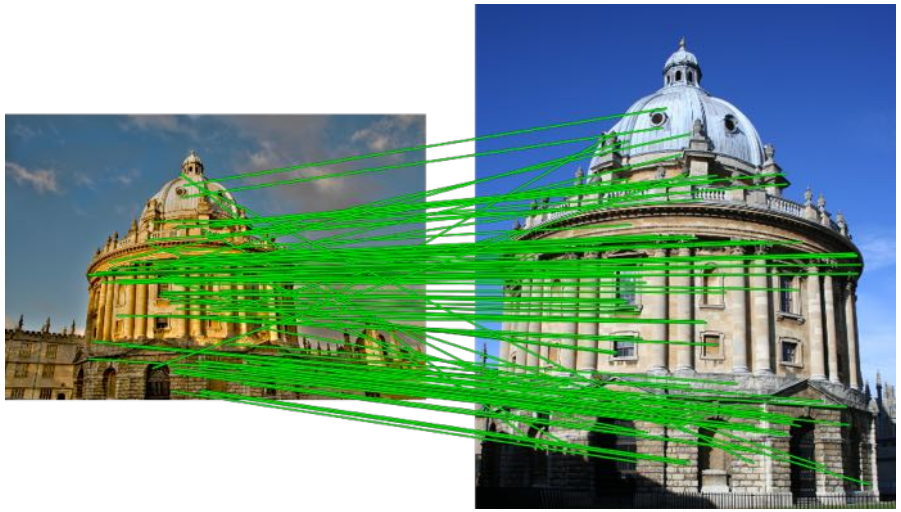
After neighbourhood consensus

[Schaffalitzky & Zisserman, CIVR 2004]

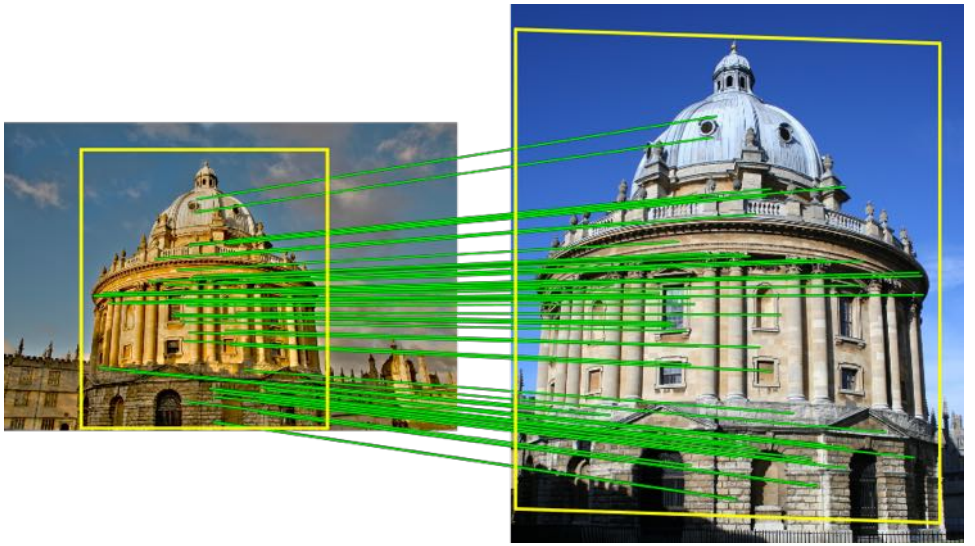
Geometric verification with global constraints

- 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

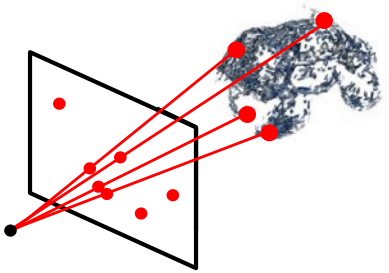


Tentative matches



Matches consistent with an affine transformation

Examples of global constraints

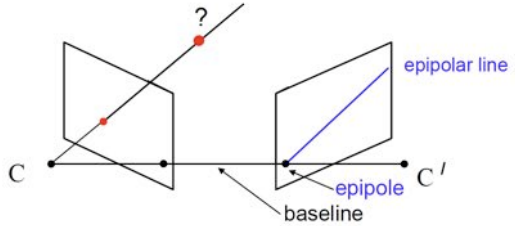


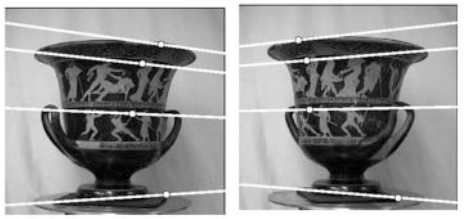
1 view and known 3D model.

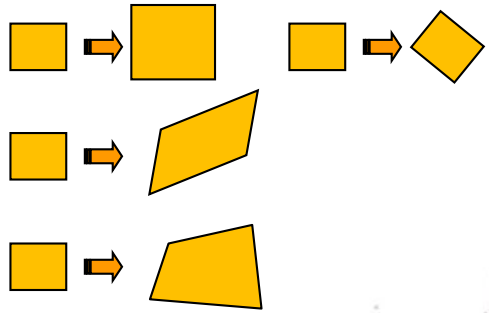
- Consistency with a (known) 3D model.

2 views

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







N-views

Are images consistent with a 3D model?

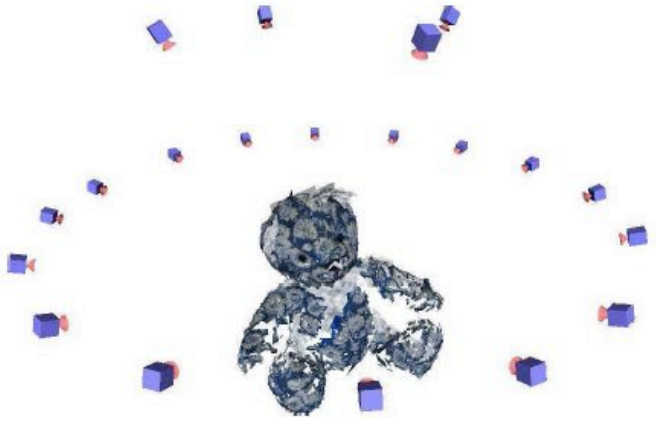
3D constraint: example

- Matches must be consistent with a 3D model

Offline: Build a 3D model

3 (out of 20) images used to build the 3D model

(a)



Recovered 3D model

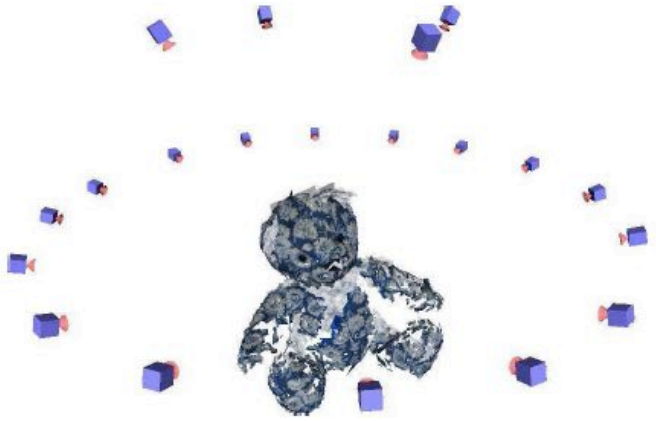
[Lazebnik, Rothganger, Schmid, Ponce, CVPR'03]

3D constraint: example

- Matches must be consistent with a 3D model

Offline: Build a 3D model

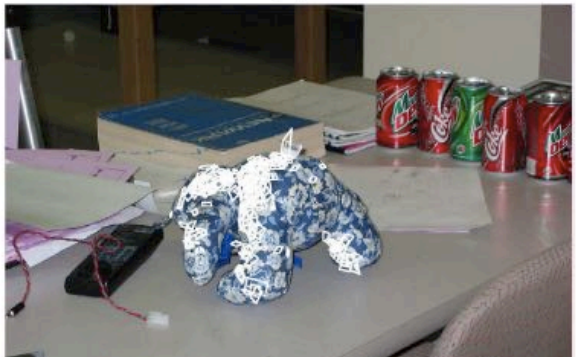
3 (out of 20) images used to build the 3D model



Recovered 3D model

At test time:

(a)



Object recognized in a previously unseen pose

Recovered pose

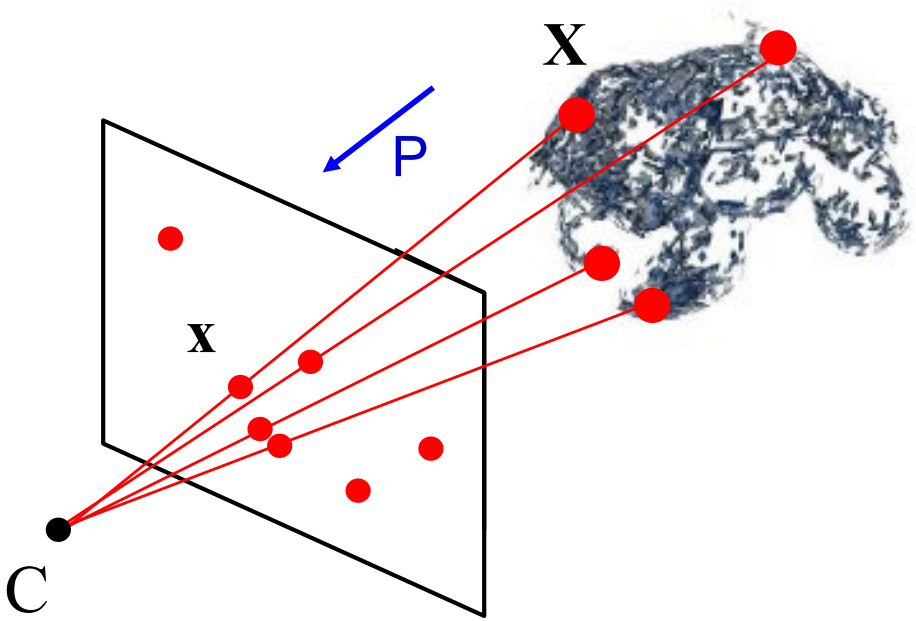
(d)

[Lazebnik, Rothganger, Schmid, Ponce, CVPR'03]

3D constraint: example

Given 3D model (set of known 3D points X's) and a set of measured 2D image points x,

find camera matrix P and a set of geometrically consistent correspondences $x \leftrightarrow X$.



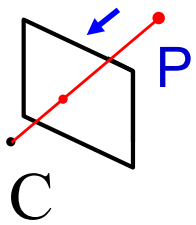
$$\mathbf{x} = \mathrm{P}\mathbf{X}$$

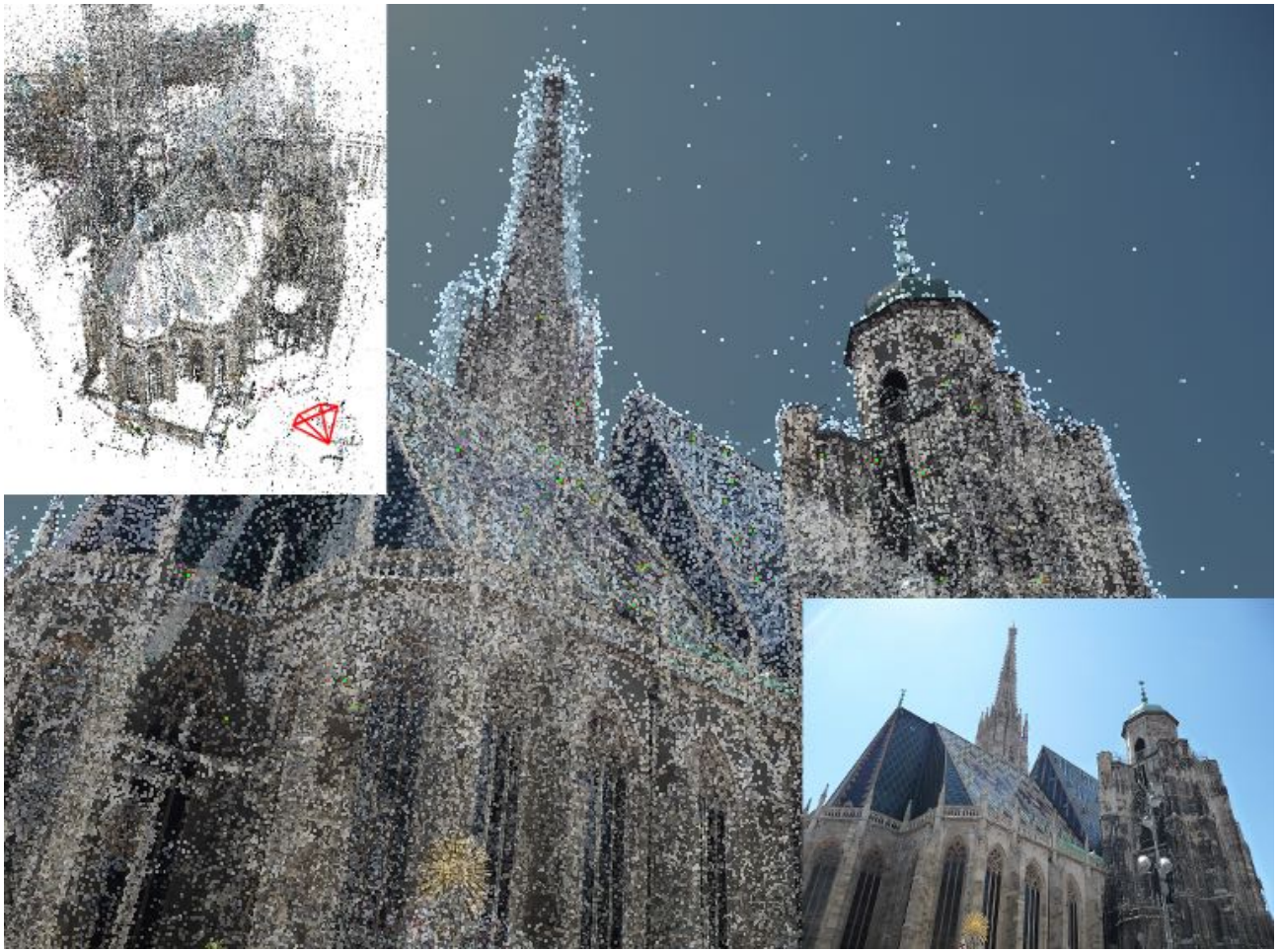
P : 3 × 4 matrix

X : 4-vector

x : 3-vector

Camera localization with respect to know 3D scene

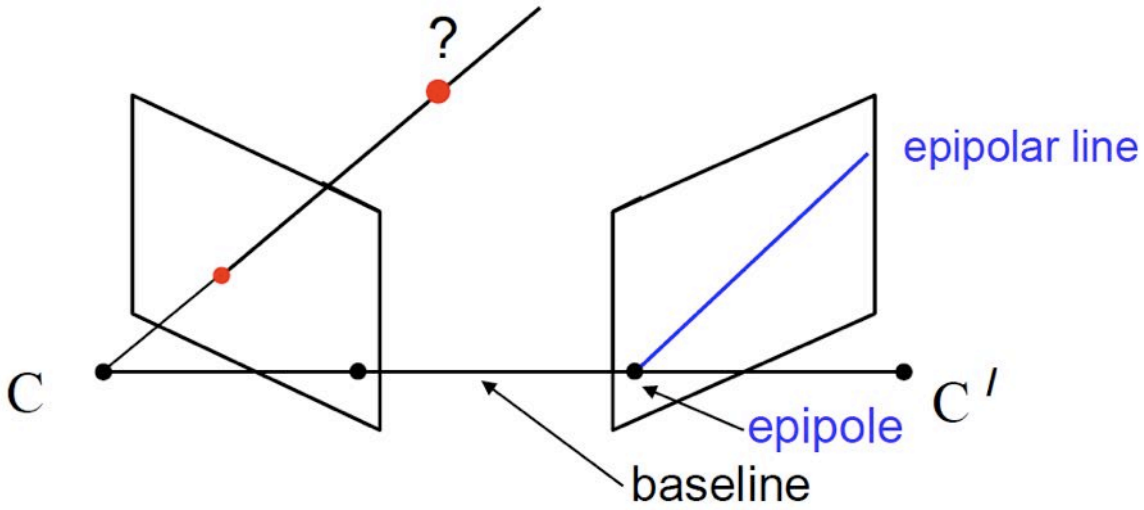


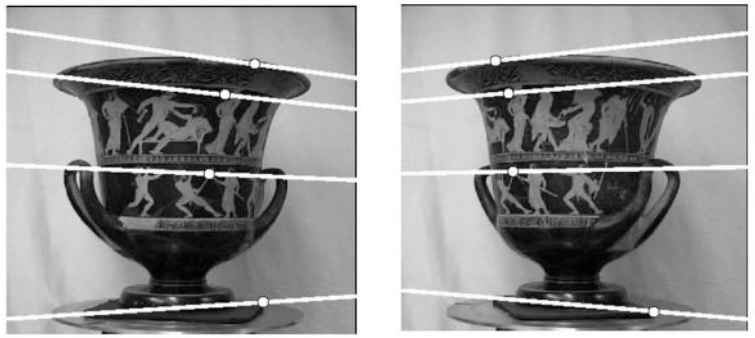


[Aubry et at'14, Irschara et al.'09, Li et al.'12, Sattler et al.'11, Sheehan et al.'13, …]

Epipolar geometry (not considered here)

In general, two views of a 3D scene are related by the epipolar constraint.



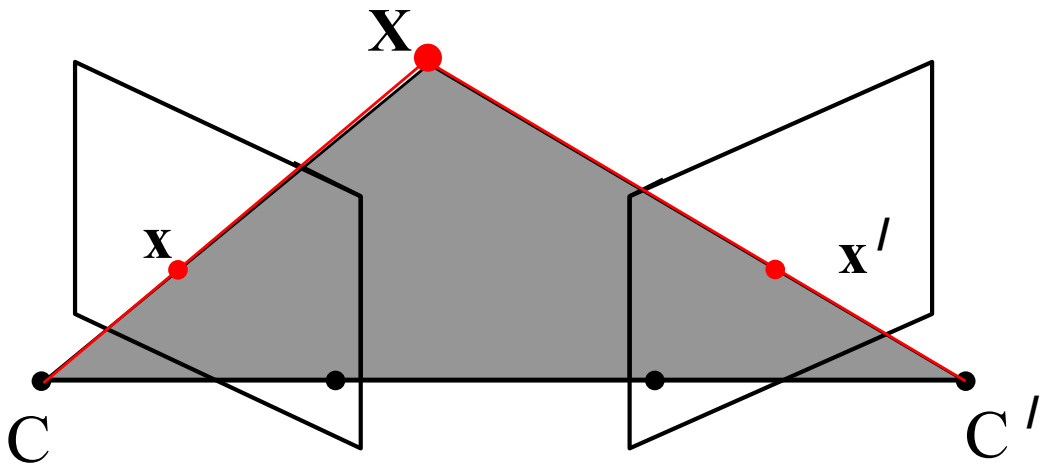


- A point in one view “generates” an epipolar line in the other view
- The corresponding point lies on this line.

Slide credit: A. Zisserman

Epipolar geometry (not considered here)

Epipolar geometry is a consequence of the coplanarity of the camera centres and scene point



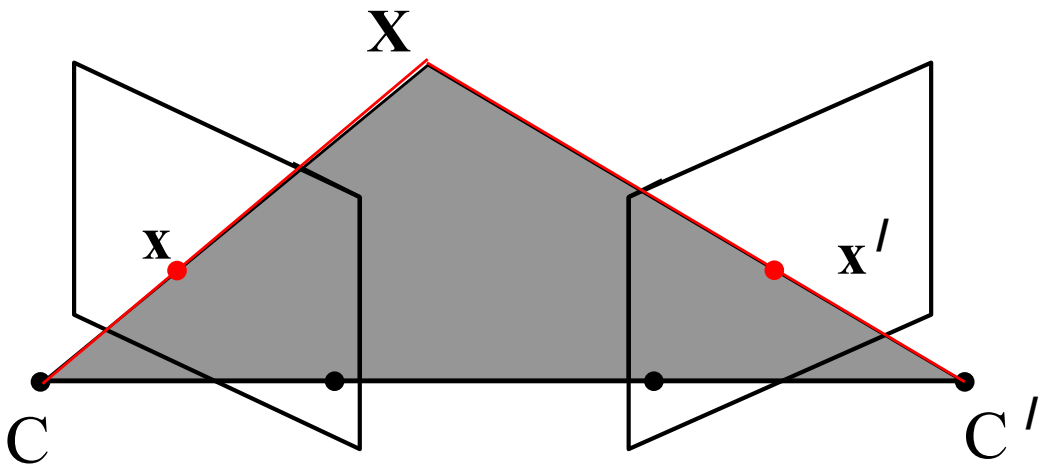
The camera centres, corresponding points and scene point lie in a single plane, known as the epipolar plane

Slide credit: A. Zisserman

Epipolar geometry (not considered here)

Algebraically, the epipolar constraint can be expressed as

$$\mathbf{x}'^{\top}\mathbf{F}\mathbf{x} = 0$$



where

- x, x' are homogeneous coordinates (3-vectors) of **corresponding** image points.

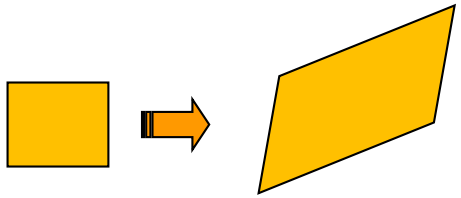
- F is a 3x3, rank 2 homogeneous matrix with 7 degrees of freedom, called the **fundamental matrix**.

Slide credit: A. Zisserman

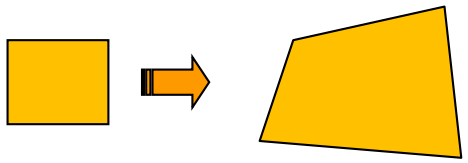
2D transformation models

Similarity
(translation,
scale, rotation)

Affine

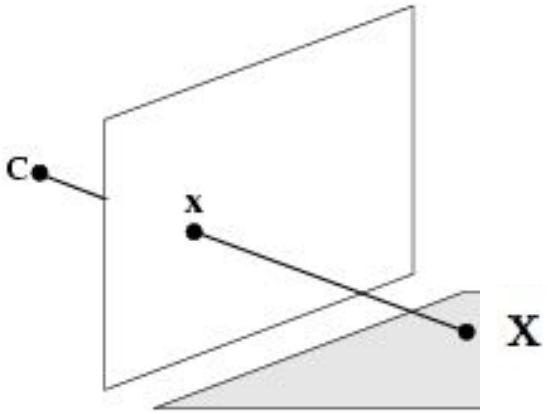


Projective
(homography)



Why are 2D planar transformations important?

Recall perspective projection



$$\mathbf{x} = \mathrm{P}\mathbf{X}$$

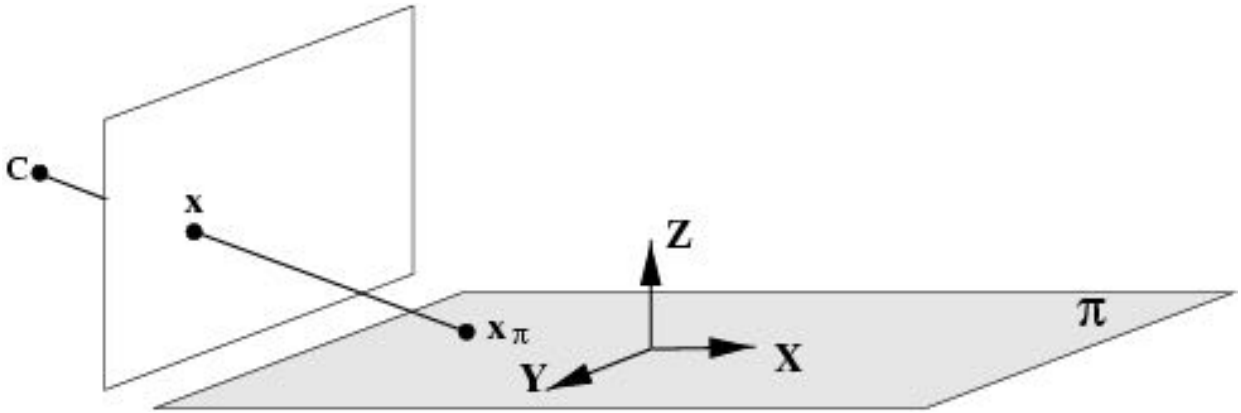
P : 3 × 4 matrix

$\mathbf{X}$: 4-vector

$\mathbf{x}$: 3-vector

Slide credit: A. Zisserman

Plane projective transformations



Choose the world coordinate system such that the plane of the points has zero z coordinate. Then the 3×4 matrix P reduces to

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{pmatrix} x \\ y \\ 0 \\ 1 \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

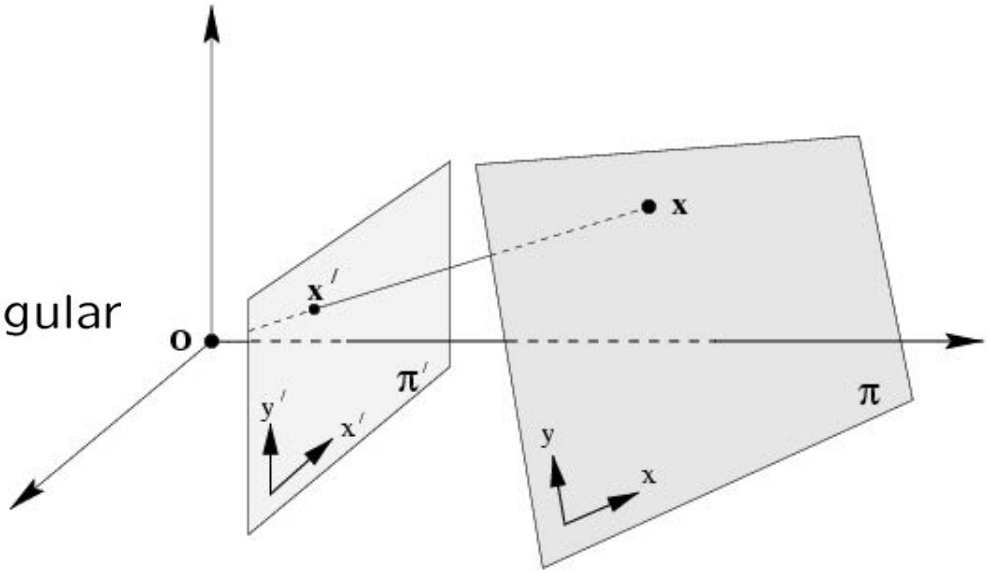
which is a 3×3 matrix representing a general plane to plane projective transformation.

Slide credit: A. Zisserman

Projective transformations continued

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

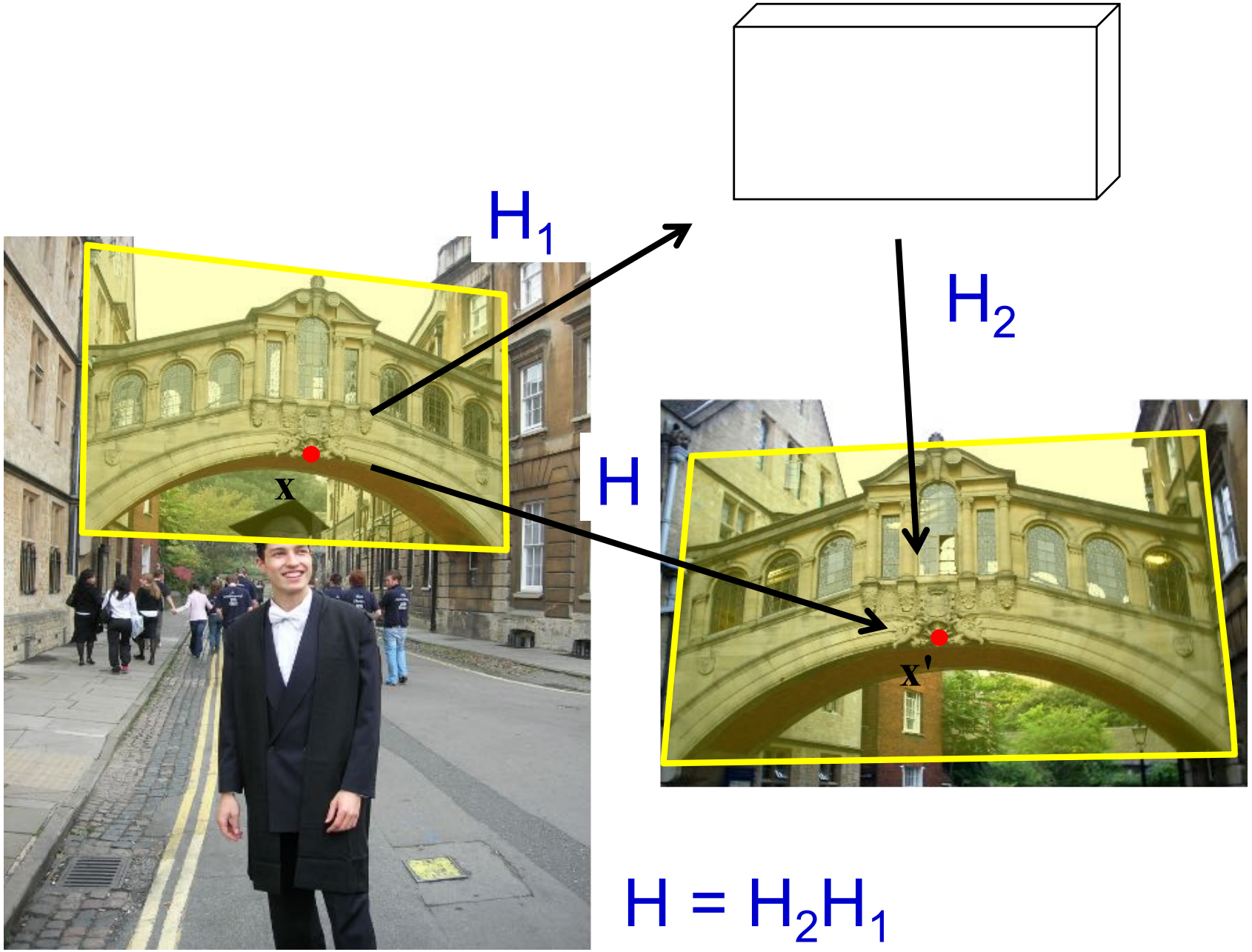
or $\mathbf{x}' = \mathtt{H}\mathbf{x}$, where $\mathtt{H}$ is a 3×3 non-singular homogeneous matrix.



- This is the most general transformation between the world and image plane under imaging by a perspective camera.
- It is often only the 3 x 3 form of the matrix that is important in establishing properties of this transformation.
- A projective transformation is also called a ``homography'' and a ``collineation''.
- H has 8 degrees of freedom. How many points are needed to compute H?

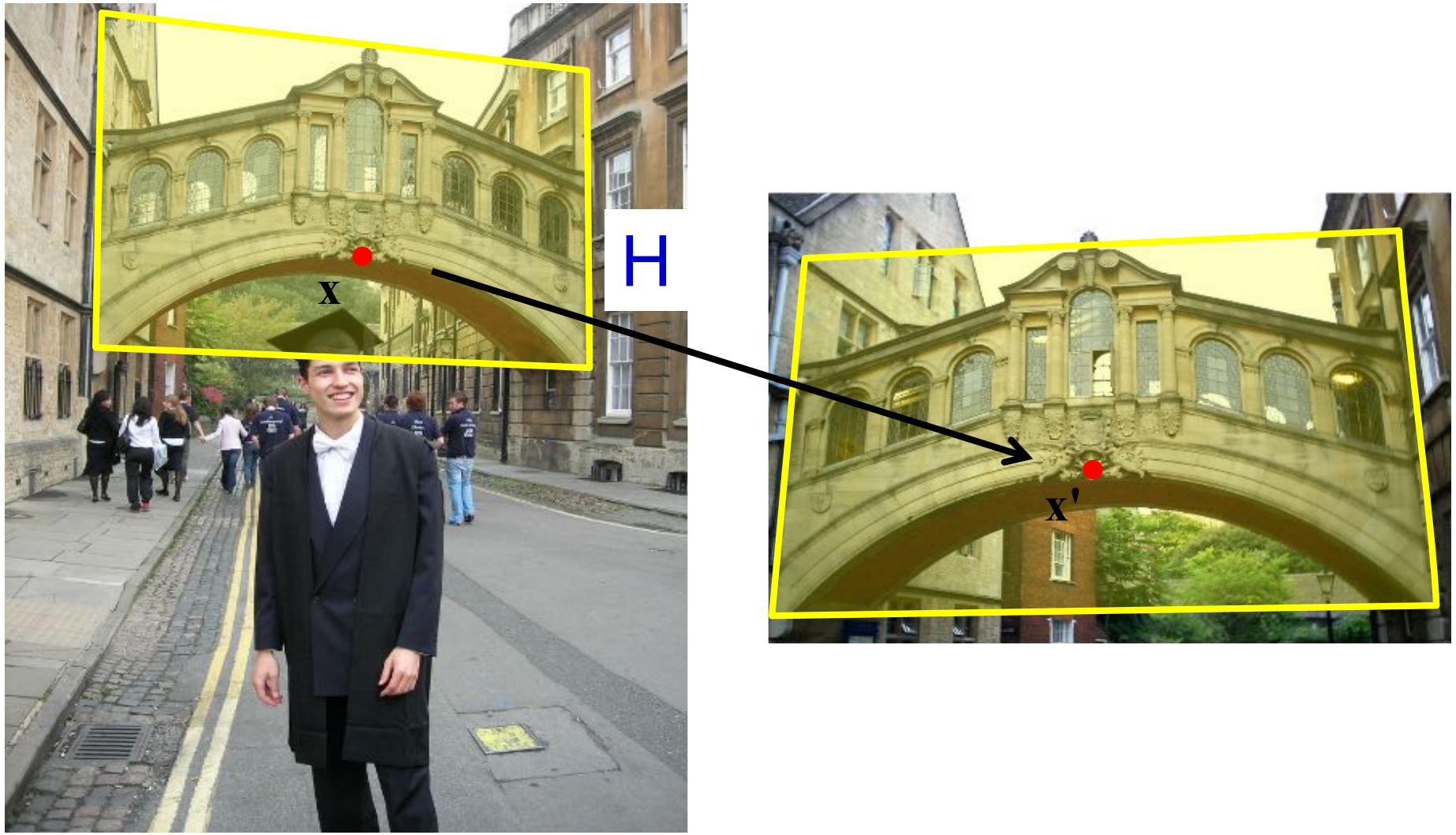
Slide credit: A. Zisserman

Planes in the scene induce *homographies*

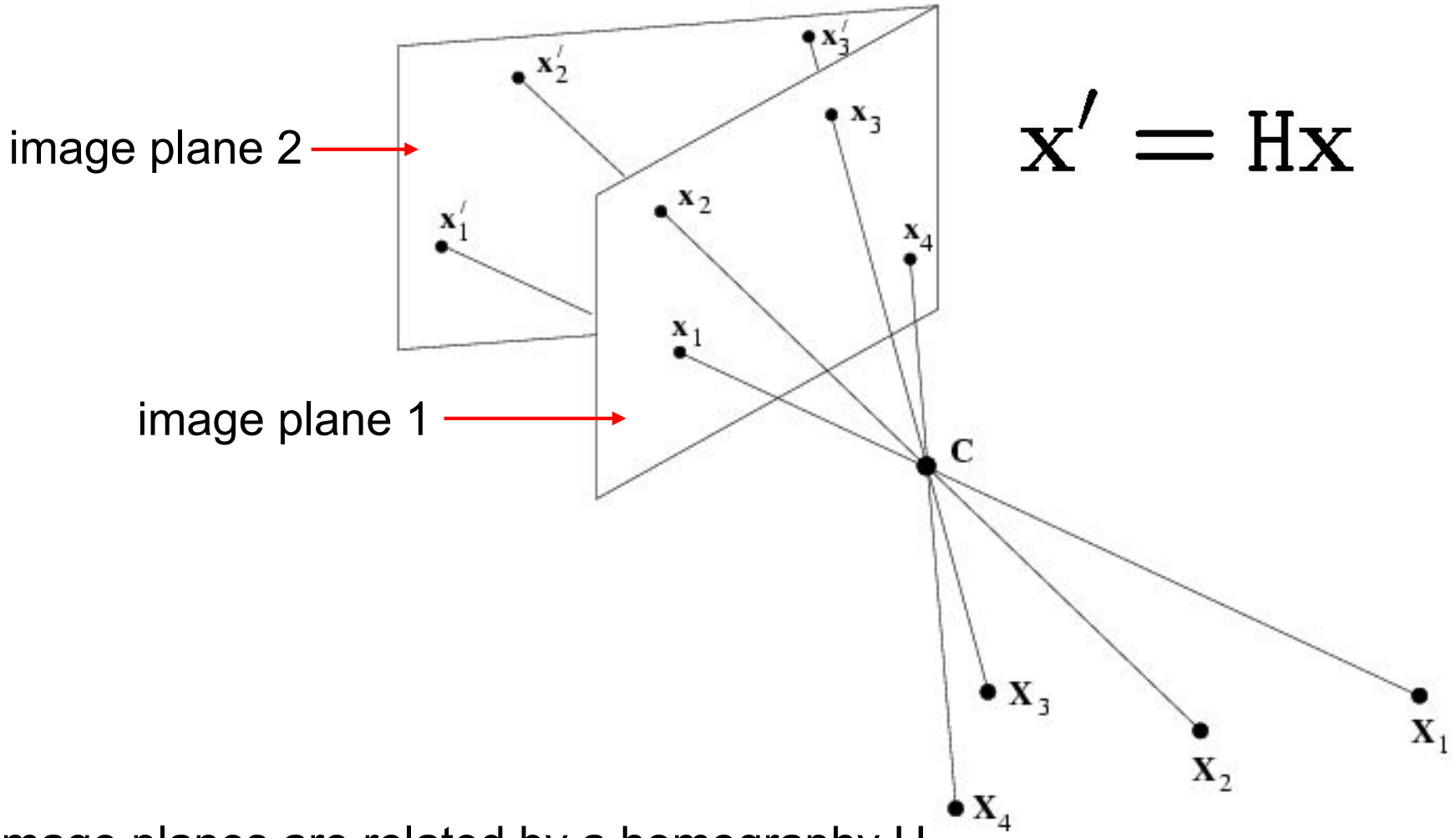


Planes in the scene induce *homographies*

Points on the plane transform as $x' = H x$, where x and x' are image points (in homogeneous coordinates), and H is a 3x3 matrix.

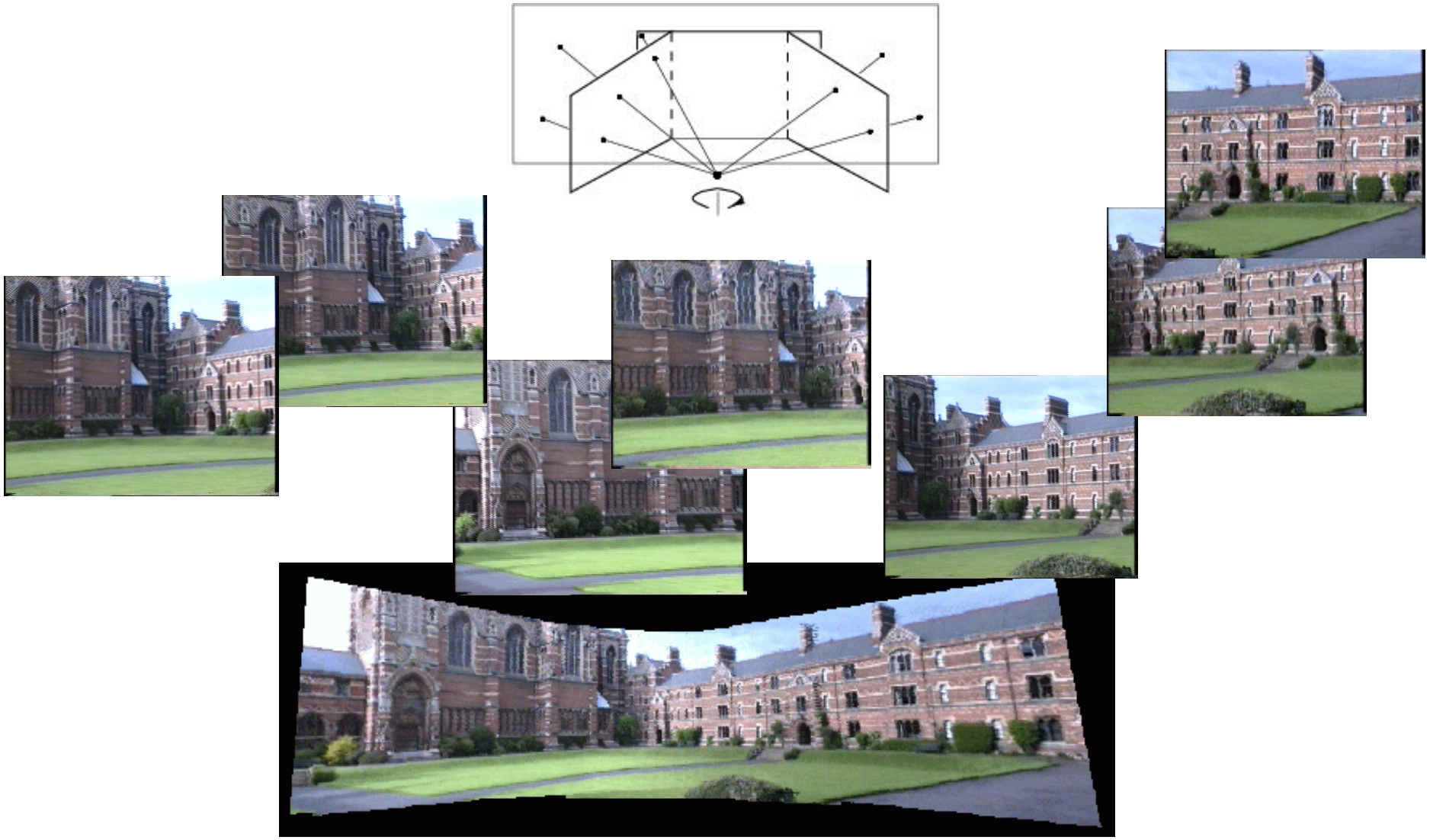


Case II: Cameras rotating about their centre



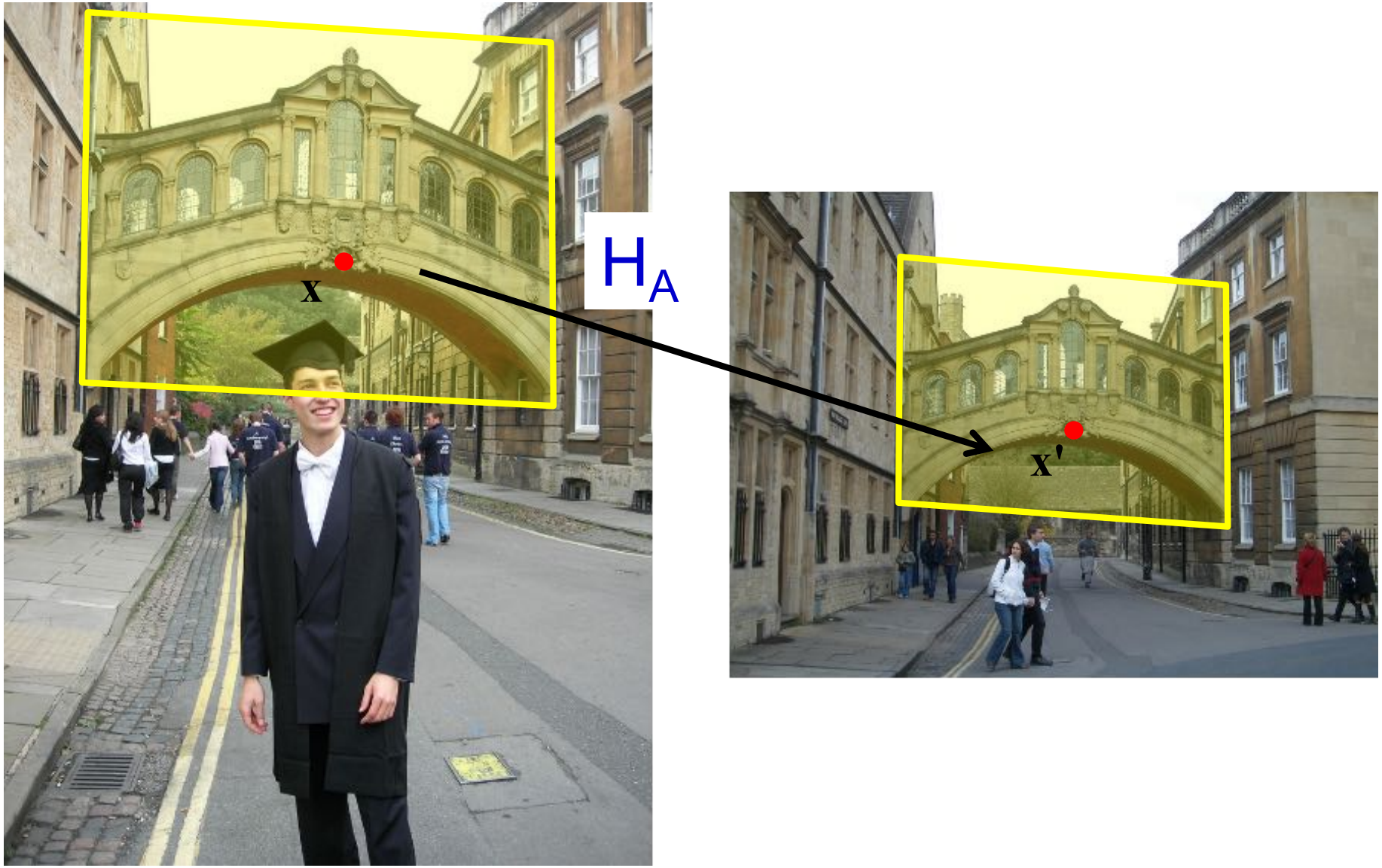
- The two image planes are related by a homography H
- H depends only on the relation between the image planes and camera centre, C, not on the 3D structure

Case II: Example of a rotating camera



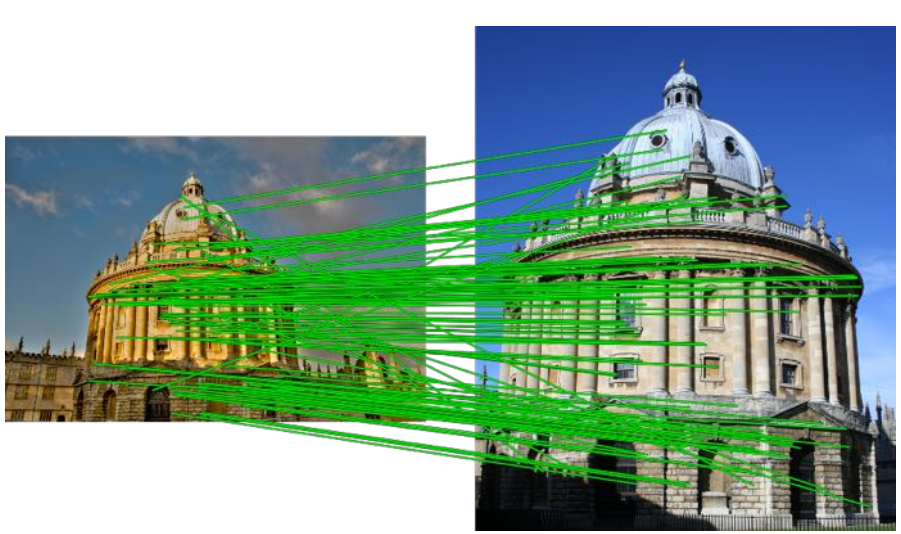
Images courtesy of A. Zisserman.

Homography is often approximated well by 2D affine geometric transformation

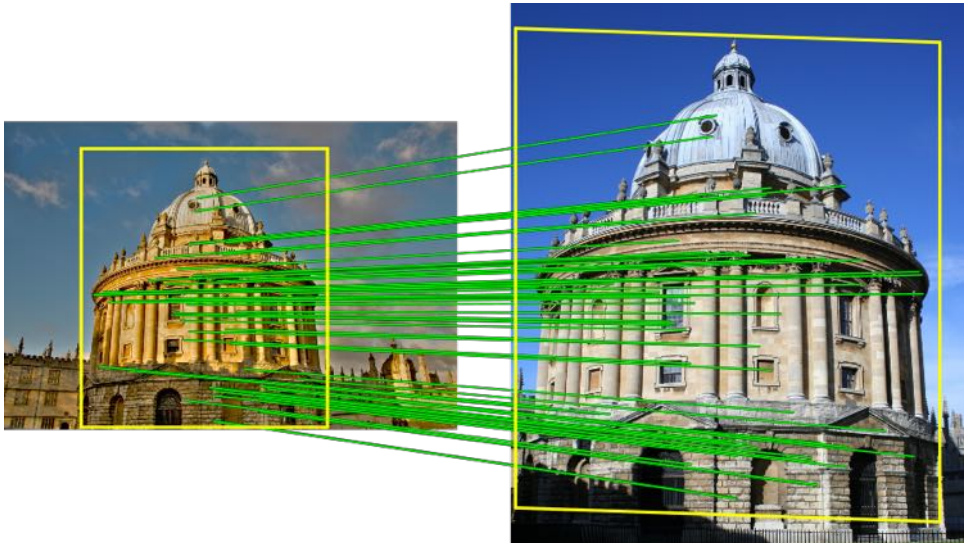


Homography is often approximated well by 2D affine geometric transformation – Example II.

Two images with similar camera viewpoint



Tentative matches



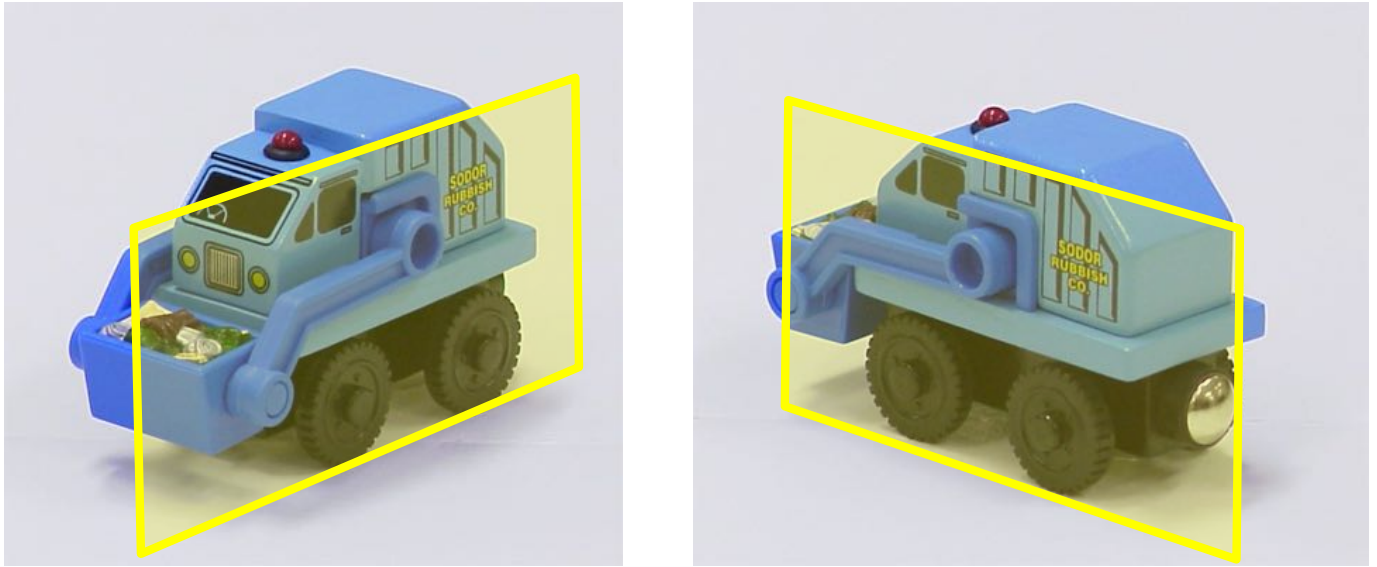
Matches consistent with an affine transformation

Example: estimating 2D affine transformation

- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- Can be used to initialize fitting for more complex models

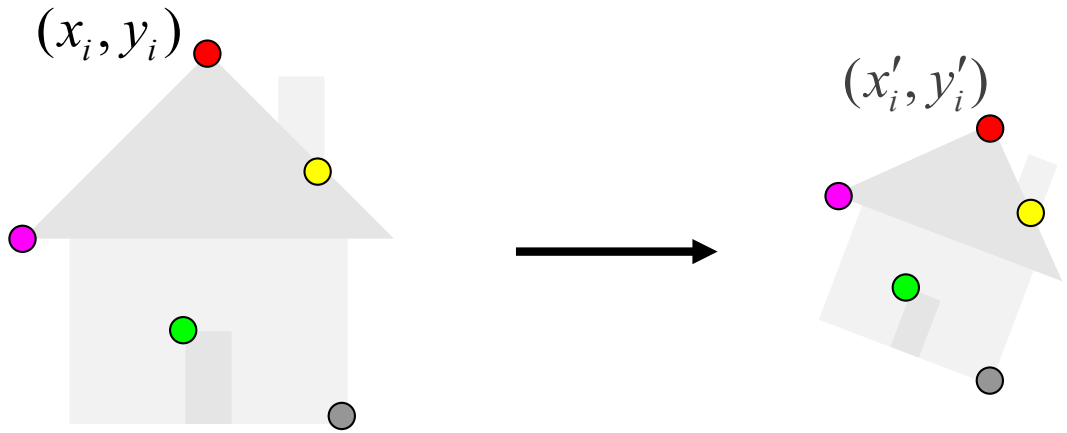
Example: estimating 2D affine transformation

- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for **roughly planar objects** and **roughly orthographic cameras**
- Can be used to initialize fitting for more complex models



Fitting an affine transformation

Assume we know the correspondences, how do we get the transformation?



$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

$$\begin{bmatrix} & \cdots & & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & \cdots & & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

Fitting an affine transformation

$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

Linear system with six unknowns

Each match gives us two linearly independent equations: need at least three to solve for the transformation parameters

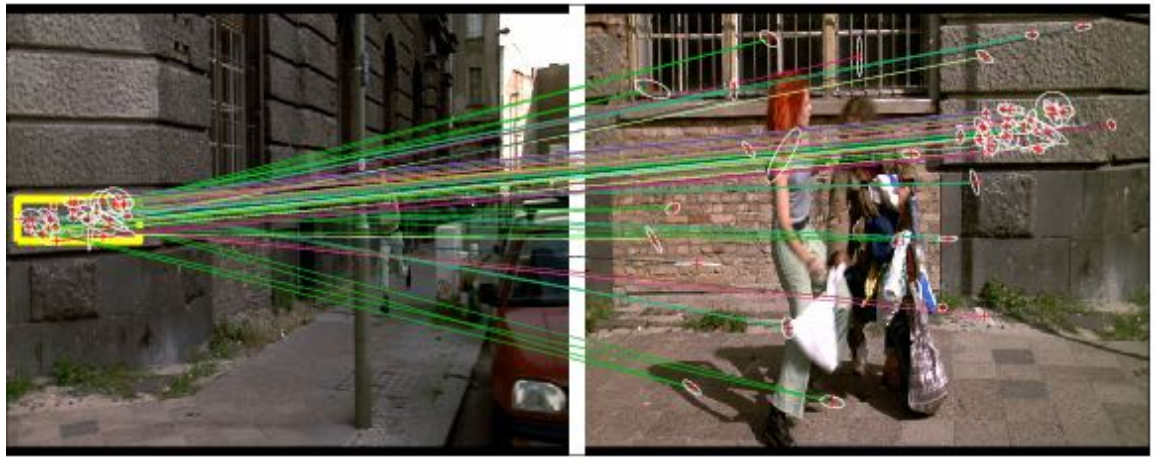
Dealing with outliers

The set of putative matches may contain a high percentage (e.g. 90%) of outliers

How do we fit a geometric transformation to a small subset of all possible matches?

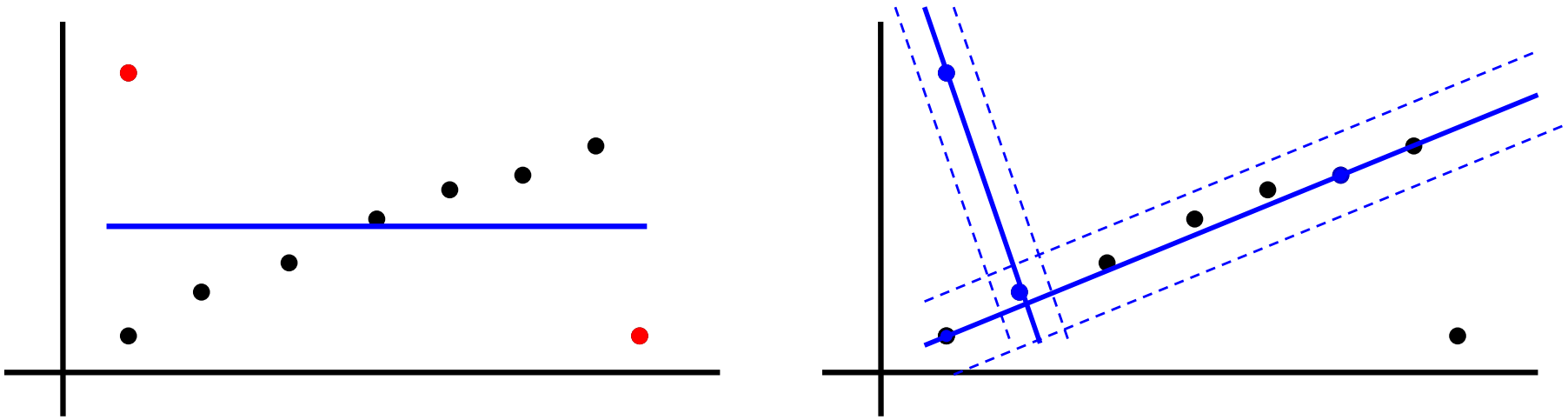
Possible strategies:

- RANSAC
- Hough transform



Example: Robust line estimation - RANSAC

Fit a line to 2D data containing outliers



There are two problems

1. a line fit which minimizes perpendicular distance
2. a classification into inliers (valid points) and outliers

Solution: use robust statistical estimation algorithm RANSAC

(RANdom Sample Consensus) [Fishler & Bolles, 1981]

Slide credit: A. Zisserman

RANSAC robust line estimation

Repeat

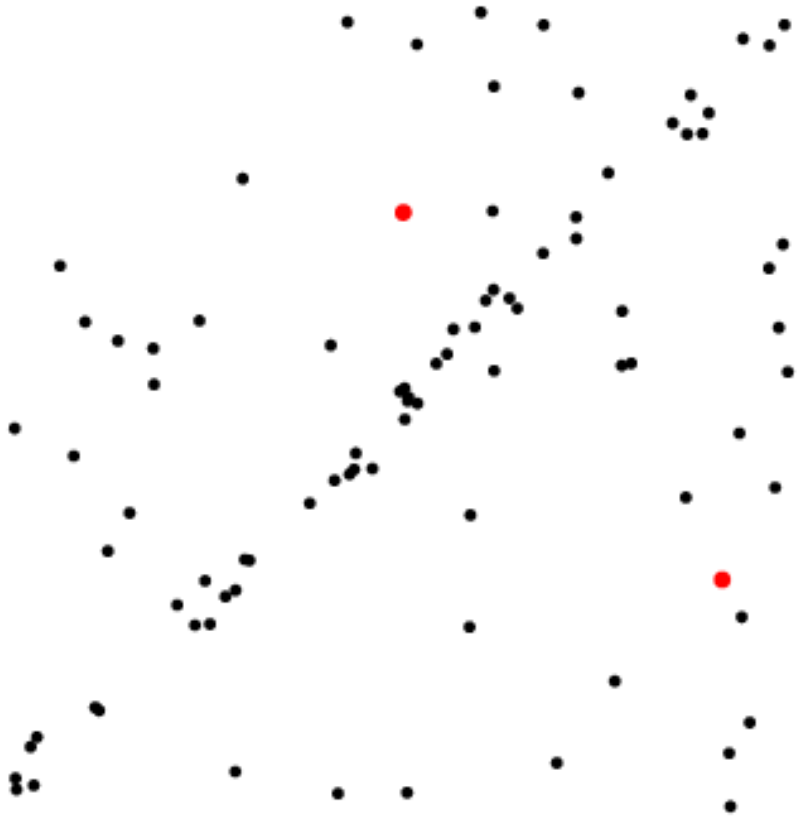
1. Select random sample of 2 points
2. Compute the line through these points
3. Measure support (number of points within threshold distance of the line)

Choose the line with the largest number of inliers

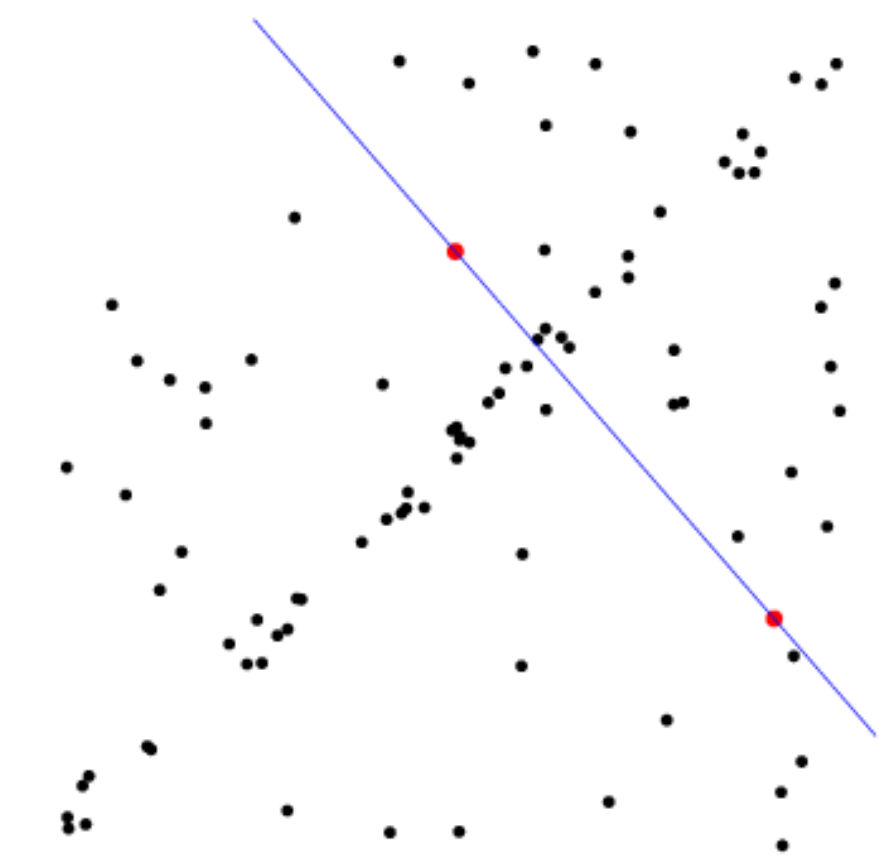
- Compute least squares fit of line to inliers (regression)

Slide credit: A. Zisserman

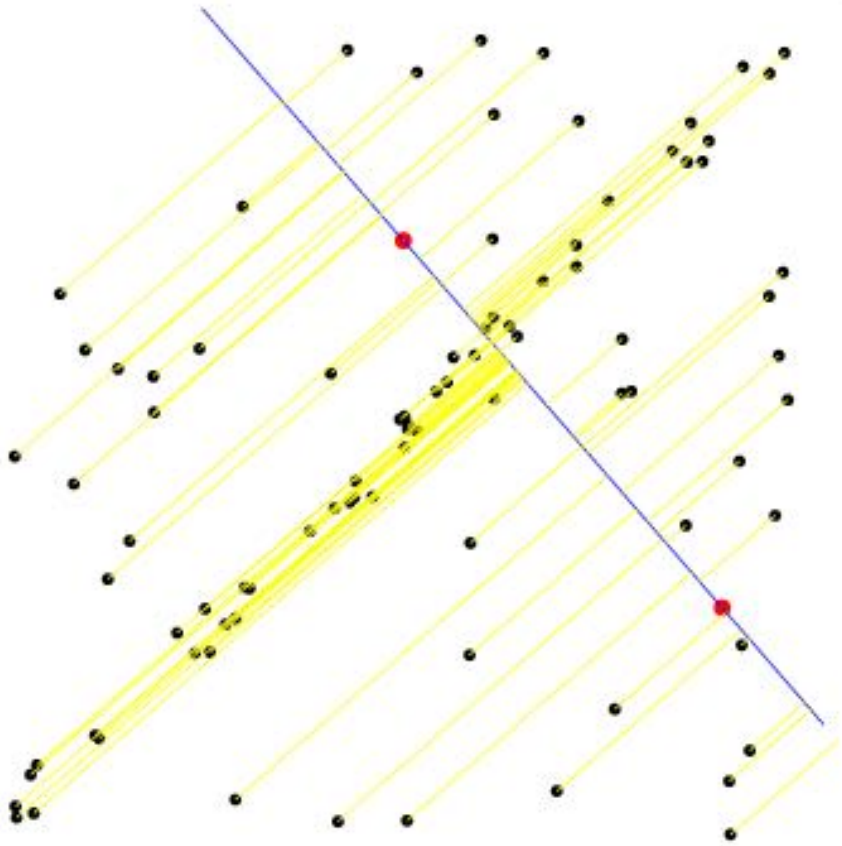
Slide credit: O. Chum



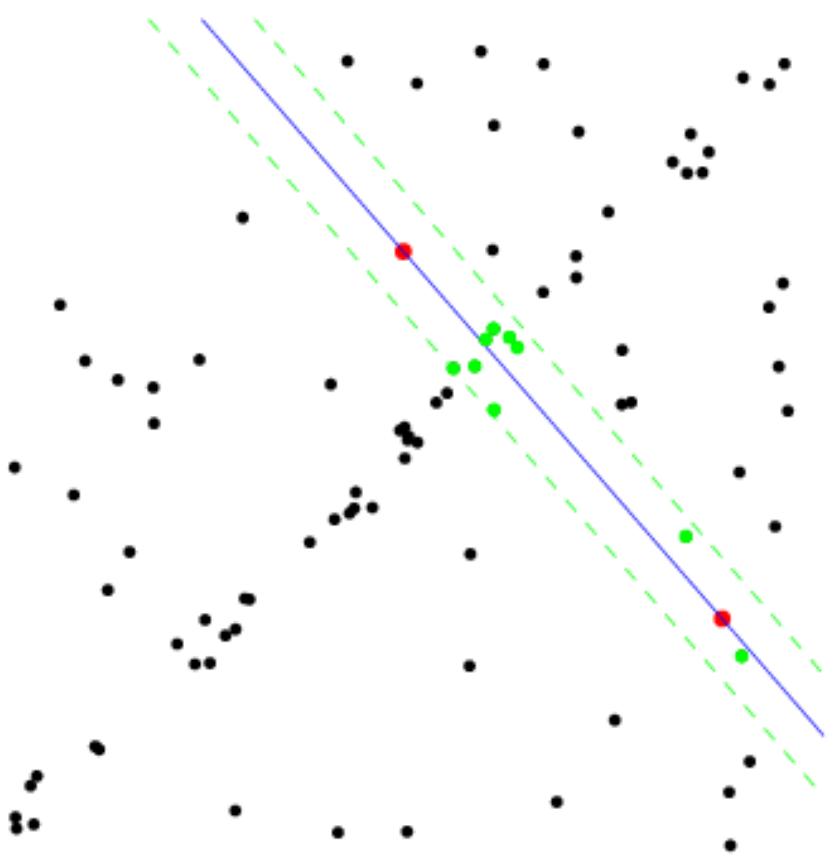
Slide credit: O. Chum



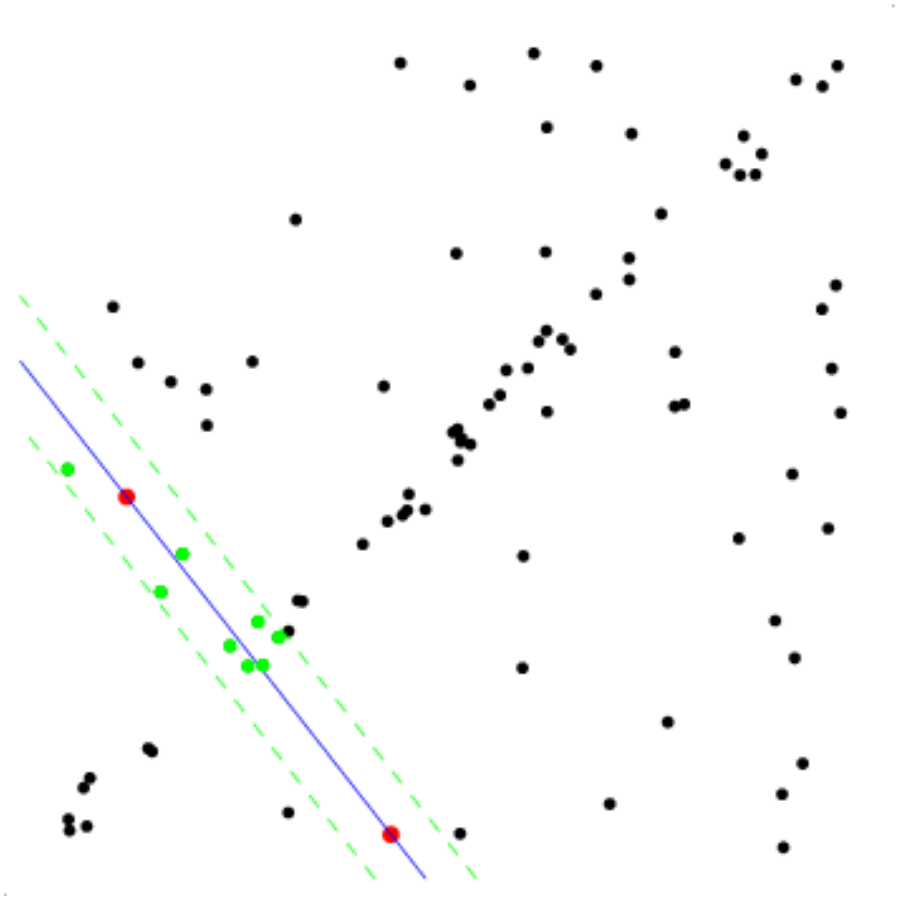
Slide credit: O. Chum



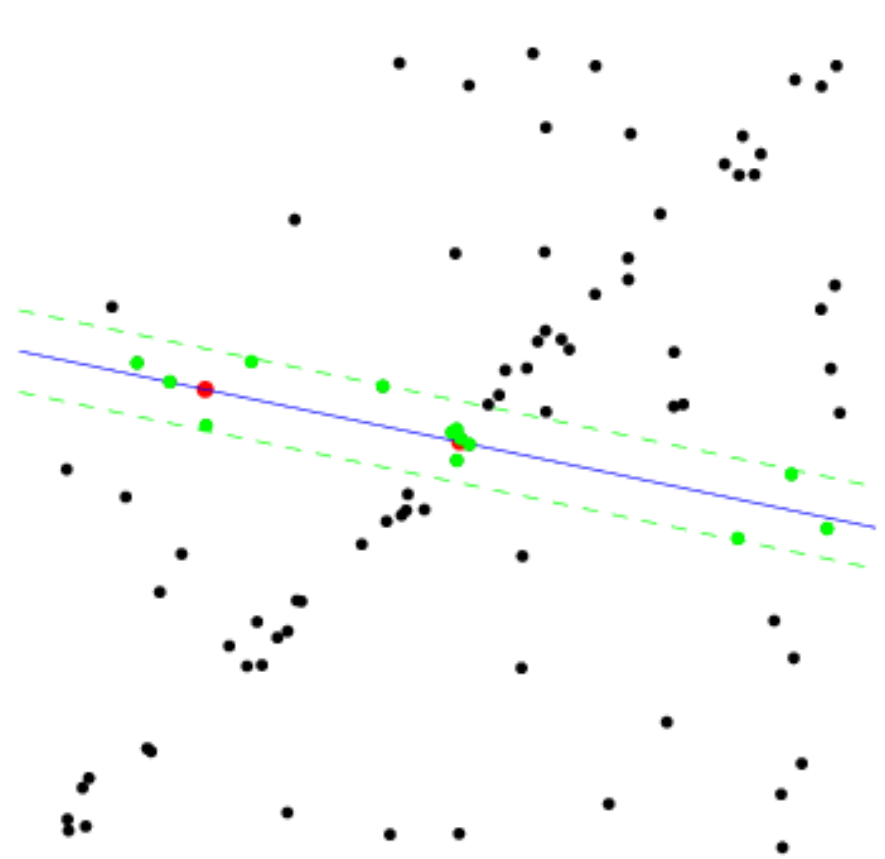
Slide credit: O. Chum



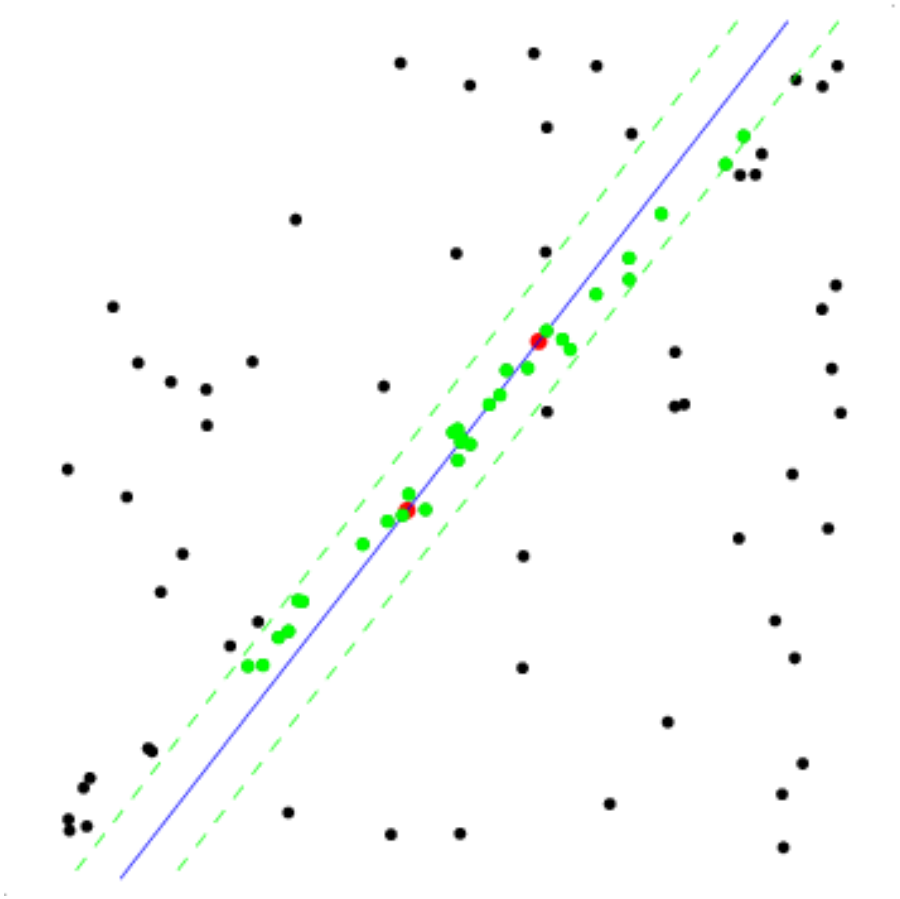
Slide credit: O. Chum



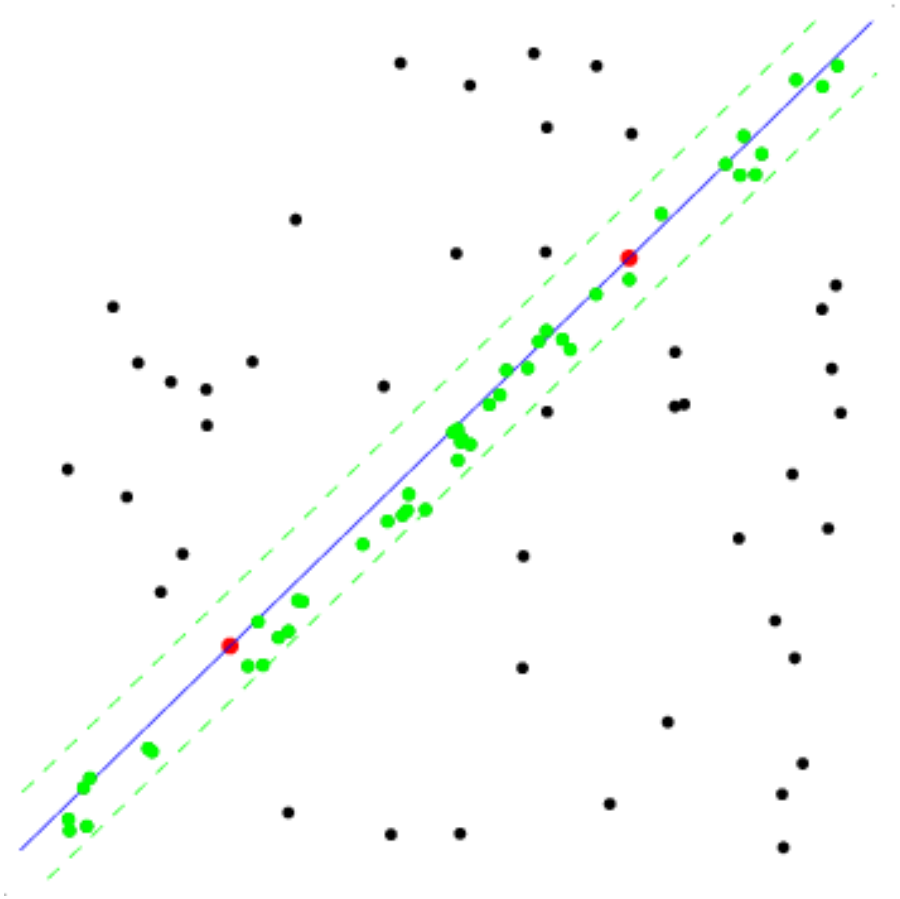
Slide credit: O. Chum



Slide credit: O. Chum



Slide credit: O. Chum



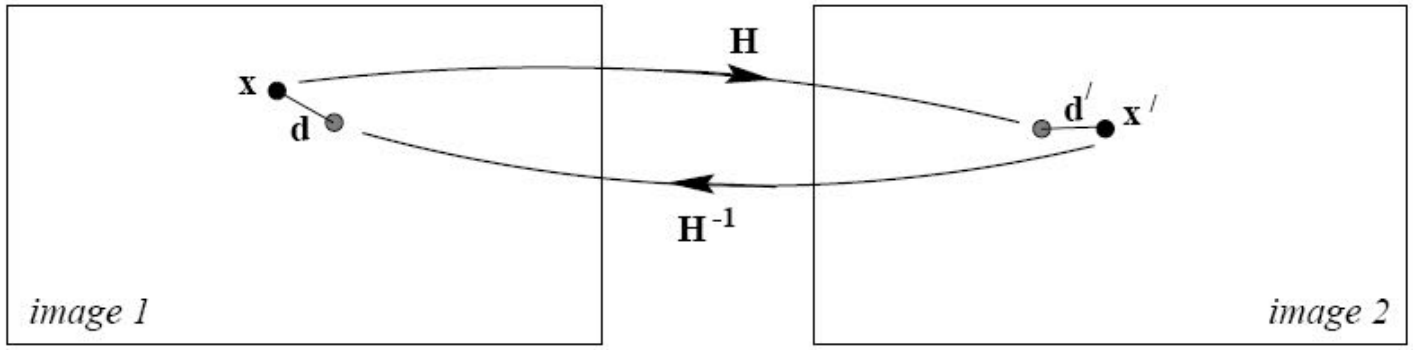
Slide credit: O. Chum

Algorithm summary – RANSAC robust estimation of 2D affine transformation

Repeat

1. Select 3 point to point correspondences
2. Compute H (2x2 matrix) + t (2x1) vector for translation
3. Measure support (number of inliers within threshold distance, i.e. $d^2_{\text{transfer}} < t$)

$$d^2_{\text{transfer}} = d(\mathbf{x}, \mathrm{H}^{-1}\mathbf{x}')^2 + d(\mathbf{x}', \mathrm{H}\mathbf{x})^2$$



Choose the (H,t) with the largest number of inliers

(Re-estimate (H,t) from all inliers)

How many samples?

Number of samples *N*

- Choose *N* so that, with probability *p*, at least one random sample is free from outliers
- e.g.:
  - > *p*=0.99
  - > outlier ratio: *e*

Probability a randomly picked point is an inlier

$$\left(1-\underbrace{\left(1-e\right)^{s}}\right)^{N}=1-p$$

Probability of all points in a sample (of size s) are inliers

Source: M. Pollefeys

How many samples?

Number of samples *N*

- Choose *N* so that, with probability *p*, at least one random sample is free from outliers
- e.g.:
  - > *p*=0.99
  - > outlier ratio: *e*

Probability that all N samples (of size s) are corrupted (contain an outlier)

$$\left(1-\underbrace{\left(1-e\right)^{s}}_{\text{Probability of at least one point in a sample (of size s) is an outlier}}\right)^{N}=1-p$$

$$N=\log\left(1-p\right)/\log\left(1-\left(1-e\right)^{s}\right)$$

	proportion of outliers *e*						
s	5%	10%	20%	30%	40%	50%	90%
1	2	2	3	4	5	6	43
2	2	3	5	7	11	17	458
3	3	4	7	11	19	35	4603
4	3	5	9	17	34	72	4.6e4
5	4	6	12	26	57	146	4.6e5
6	4	7	16	37	97	293	4.6e6
7	4	8	20	54	163	588	4.6e7
8	5	9	26	78	272	1177	4.6e8

Source: M. Pollefeys

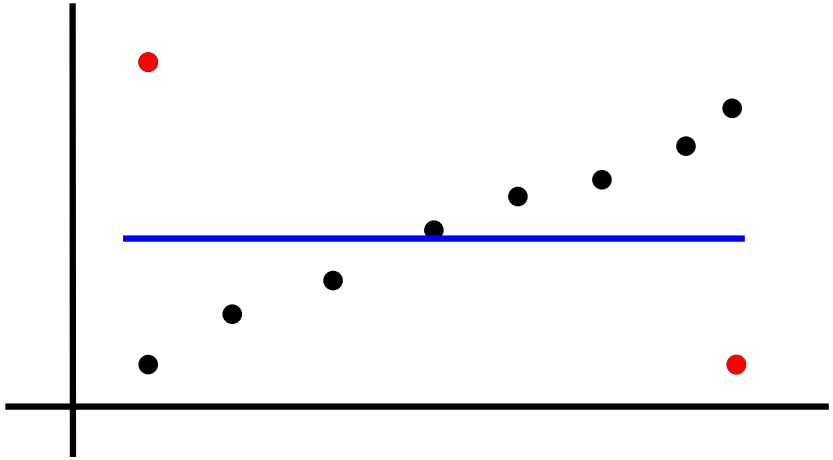
Example: line fitting

p = 0.99

s = ?

e = ?

N = ?



Source: M. Pollefeys

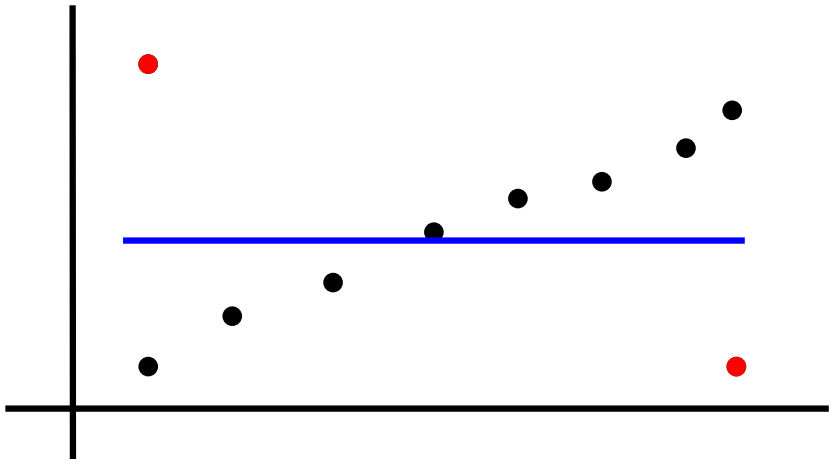
Example: line fitting

p = 0.99

s = 2

e = 2/10 = 0.2

N = 5



Compare with exhaustively trying all point pairs:

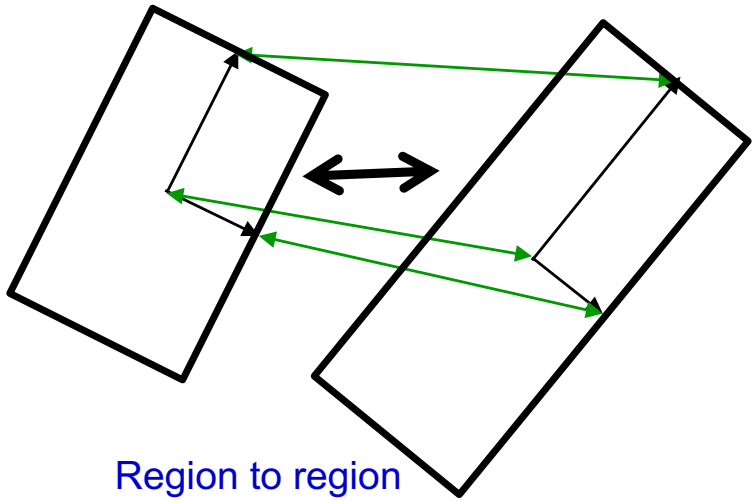
$$\binom{10}{2} = 10*9 / 2 = 45$$

	proportion of outliers e						
s	5%	10%	20%	30%	40%	50%	90%
1	2	2	3	4	5	6	43
2	2	3	5	7	11	17	458
3	3	4	7	11	19	35	4603
4	3	5	9	17	34	72	4.6e4
5	4	6	12	26	57	146	4.6e5
6	4	7	16	37	97	293	4.6e6
7	4	8	20	54	163	588	4.6e7
8	5	9	26	78	272	1177	4.6e8

Source: M. Pollefeys

How to reduce the number of samples needed?

1. Reduce the proportion of outliers.
2. Reduce the sample size
   - use simpler model (e.g. similarity instead of affine tnf.)
   - use local information (e.g. a region to region correspondence is equivalent to (up to) 3 point to point correspondences).



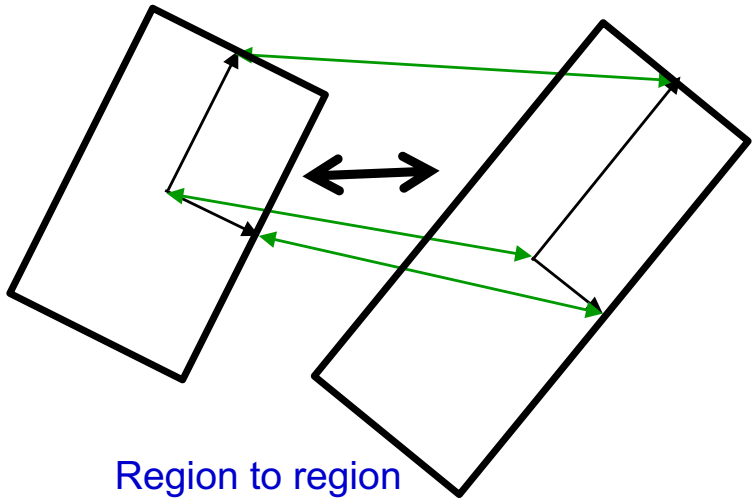
Region to region correspondence

Number of samples *N*

	proportion of outliers *e*						
s	5%	10%	20%	30%	40%	50%	90%
1	2	2	3	4	5	6	43
2	2	3	5	7	11	17	458
3	3	4	7	11	19	35	4603
4	3	5	9	17	34	72	4.6e4
5	4	6	12	26	57	146	4.6e5
6	4	7	16	37	97	293	4.6e6
7	4	8	20	54	163	588	4.6e7
8	5	9	26	78	272	1177	4.6e8

How many samples are needed?

1. Depends on the proportion of outliers.
2. Depends on the sample size “s”
   - use simpler model (e.g. similarity instead of affine tnf.)
   - use local information (e.g. a region to region correspondence is equivalent to (up to) 3 point to point correspondences).



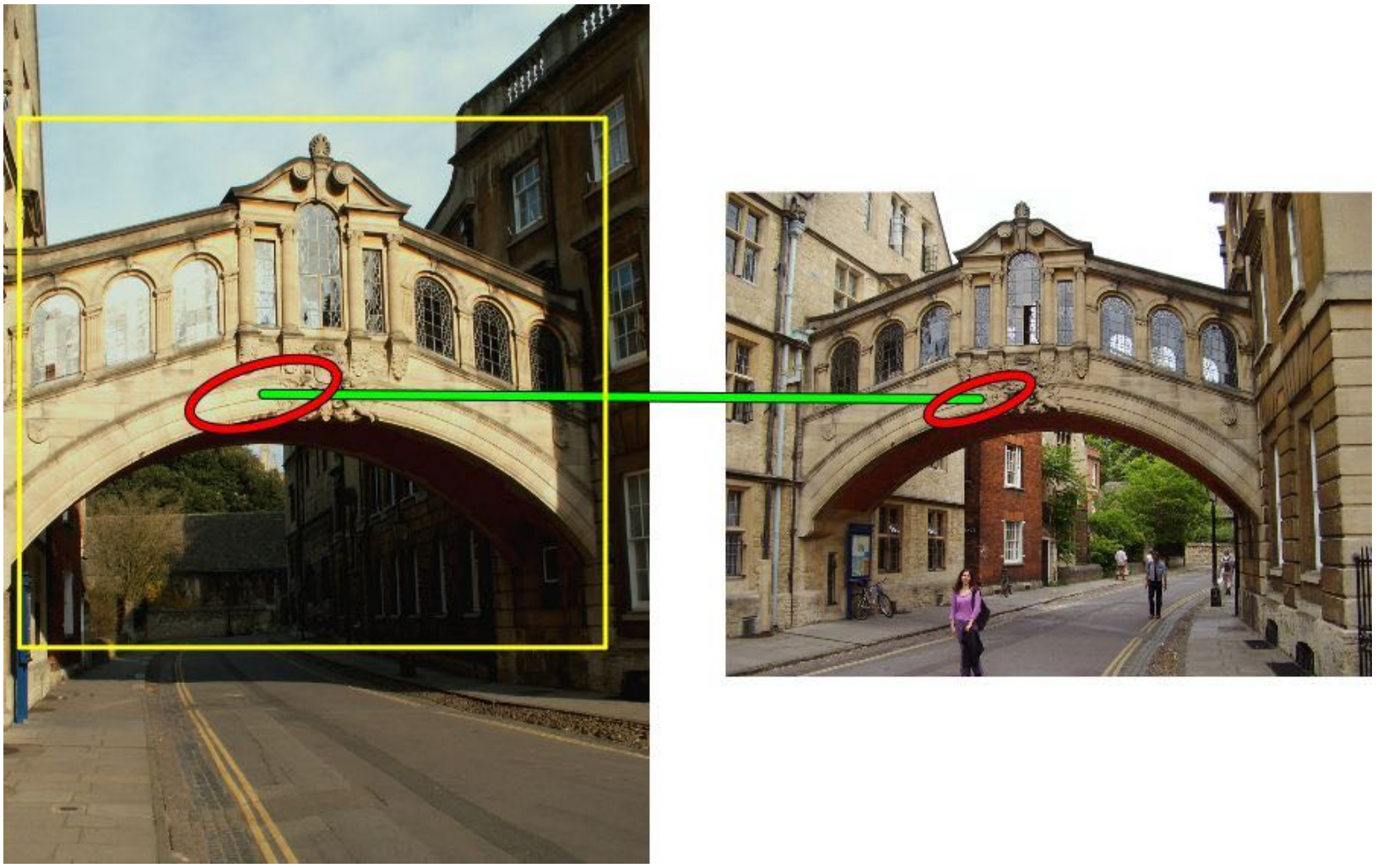
Region to region correspondence

Number of samples *N*

	proportion of outliers *e*						
s	5%	10%	20%	30%	40%	50%	90%
1	2	2	3	4	5	6	43
2	2	3	5	7	11	17	458
3	3	4	7	11	19	35	4603
4	3	5	9	17	34	72	4.6e4
5	4	6	12	26	57	146	4.6e5
6	4	7	16	37	97	293	4.6e6
7	4	8	20	54	163	588	4.6e7
8	5	9	26	78	272	1177	4.6e8

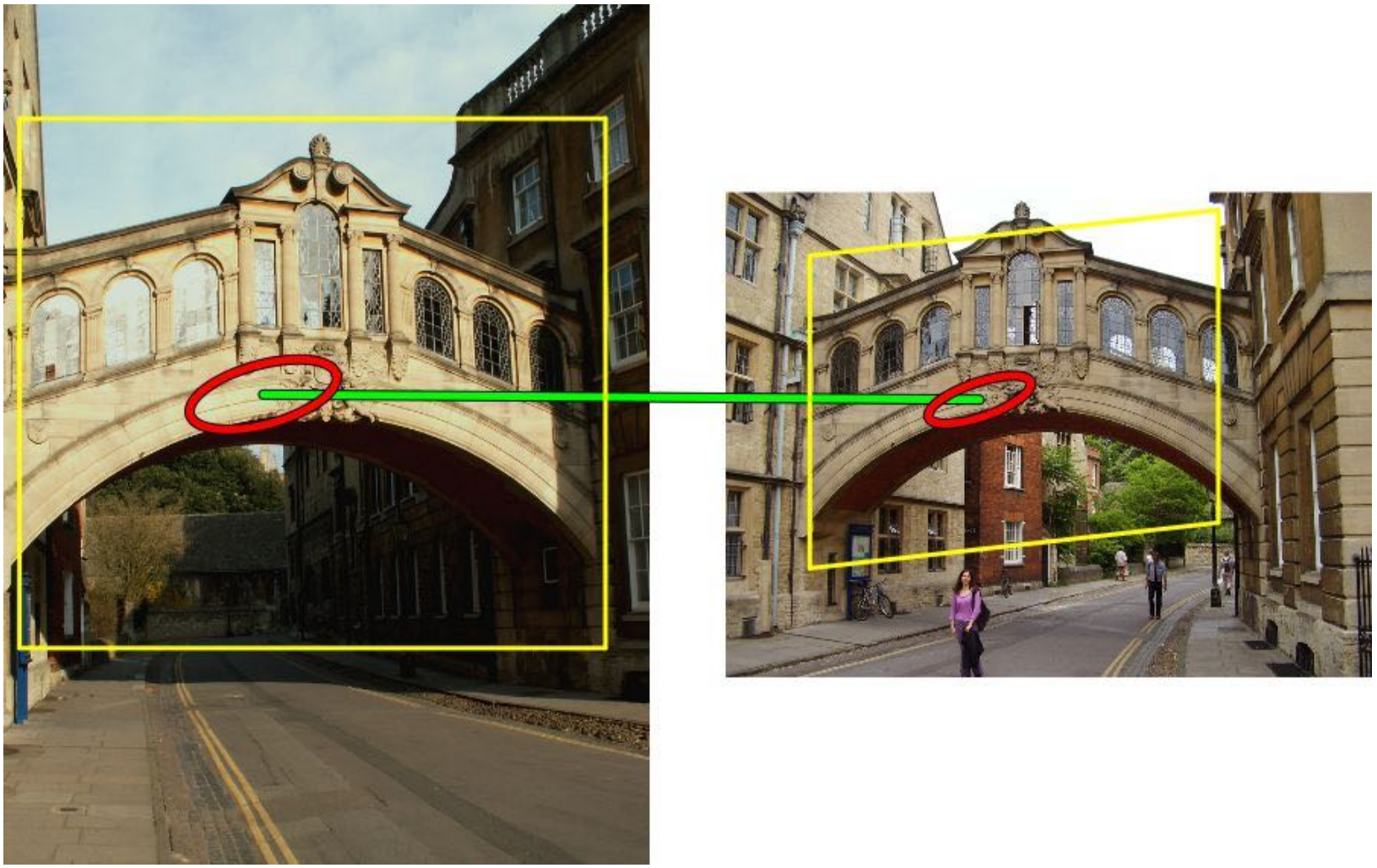
Example: restricted affine transform

1. Test each correspondence



Example: restricted affine transform

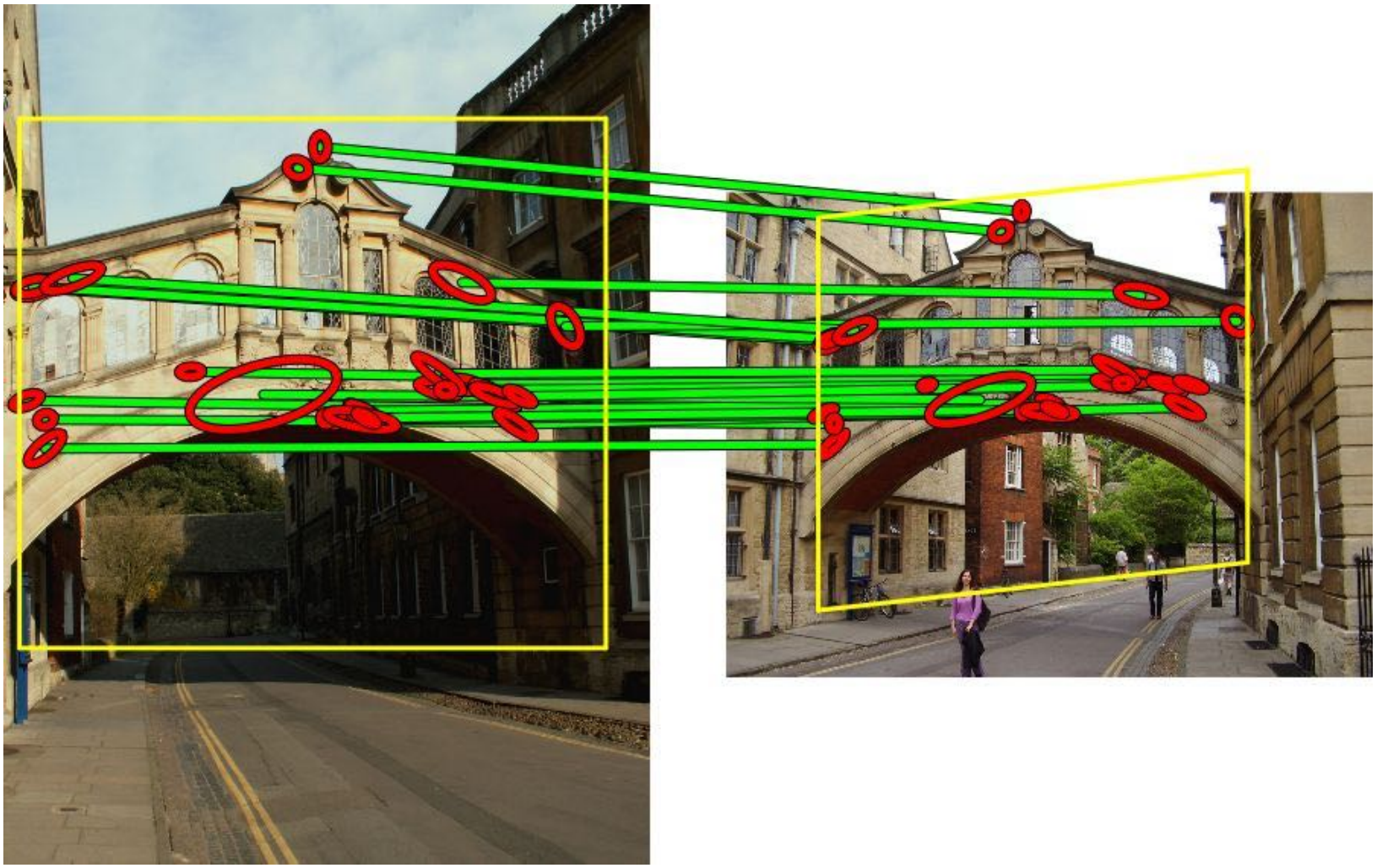
2. Compute a (restricted) planar affine transformation (5 dof)



Need just one correspondence

Example: restricted affine transform

3. Score by number of consistent matches

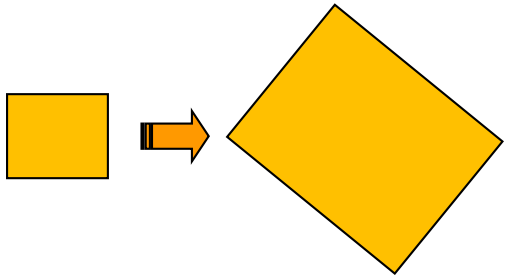


Re-estimate full affine transformation (6 dof)

Example II: Similarity transformation

Similarity transformation is specified by four parameters: scale factor s, rotation θ, and translations t_x and t_y.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = sR(\theta) \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$



Recall, each SIFT detection has: position (x_i, y_i), scale s_i, and orientation θ_i.

How many correspondences are needed to compute similarity transformation?

RANSAC (references)

M. Fischler and R. Bolles, “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,” Comm. ACM, 1981

R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd ed., 2004.

Extensions:

B. Tordoff and D. Murray, “Guided Sampling and Consensus for Motion Estimation, ECCV’03

D. Nister, “Preemptive RANSAC for Live Structure and Motion Estimation, ICCV’03

Chum, O.; Matas, J. and Obdrzalek, S.: Enhancing RANSAC by Generalized Model Optimization, ACCV’04

Chum, O.; and Matas, J.: Matching with PROSAC - Progressive Sample Consensus , CVPR 2005

Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching, CVPR’07

Chum, O. and Matas. J.: Optimal Randomized RANSAC, PAMI’08

Lebeda, Matas, Chum: Fixing the locally optimized RANSAC, BMVC’12 (code available).

Geometric verification for visual search (references)

Schmid and Mohr, Local gray-value invariants for image retrieval, PAMI 1997

Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. CVPR (2007)

Perdoch, M., Chum, O., Matas, J.: Efficient representation of local geometry for large scale object retrieval. CVPR (2009)

Wu, Z., Ke, Q., Isard, M., Sun, J.: Bundling features for large scale partial-duplicate web image search. In: CVPR (2009)

Jegou, H., Douze, M., Schmid, C.: Improving bag-of-features for large scale image search. IJCV 87(3), 316–336 (2010)

Lin, Z., Brandt, J.: A local bag-of-features model for large-scale object retrieval. ECCV 2010)

Zhang, Y., Jia, Z., Chen, T.: Image retrieval with geometry preserving visual phrases. In: CVPR (2011)

Tolias, G., Avrithis, Y.: Speeded-up, relaxed spatial matching. In: ICCV (2011)

Shen, X., Lin, Z., Brandt, J., Avidan, S., Wu, Y.: Object retrieval and localization with spatially-constrained similarity measure and k-nn re-ranking. In: CVPR. IEEE (2012)

H. Stewénius, S. Gunderson, J. Pilet. Size matters: exhaustive geometric verification for image retrieval, ECCV 2012.

Summary

Finding correspondences in images is useful for

- Image matching, panorama stitching, object recognition
- Large scale image search: lecture 4 (C. Schmid)

Represent images using local invariant features

- Harris corners, scale selection using Laplacian of Gauss.

Beyond local point matching

- Semi-local relations
- Global geometric relations:
  - Epipolar constraint
  - 3D constraint (when 3D model is available)
  - 2D tnfs: Similarity / Affine / Homography
- Algorithms: RANSAC, [Hough transform]

$$\mathbf{x}'^{\top}\mathbf{F}\mathbf{x} = 0$$

$$\mathbf{x} = \mathbf{P}\mathbf{X}$$

$$\mathbf{x}' = \mathbf{H}\mathbf{x}$$

Approach

1. **Pre-processing:**
   - Detect local features.
   - Extract descriptor for each feature.

2. **Matching:** Establish tentative (putative) correspondences based on local appearance of individual features (their descriptors).

3. **Verification:** Verify matches based on semi-local / global geometric relations.

4. **Current research challenges**

What next?

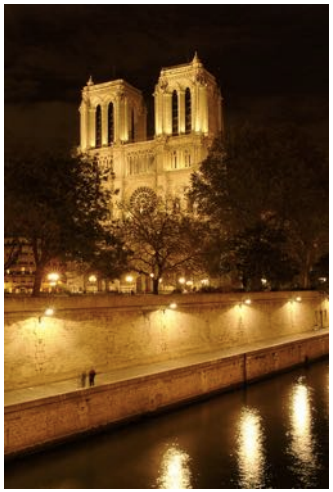
Visual search for texture-less, wiry, deformable and 3D objects..

See e.g.
- Where to buy it: Matching street clothing photos in online shops, M Hadi Kiapour, X Han, S Lazebnik, AC Berg, TL Berg, ICCV 2015.
- Learning visual similarity for product design with convolutional neural networks, S Bell, K BalaACM Transactions on Graphics (TOG) 34 (4), 98

What next?

Match objects across large changes of appearance

Examples: non-photographic depictions, degradation over time, change of season, change of illumination, …

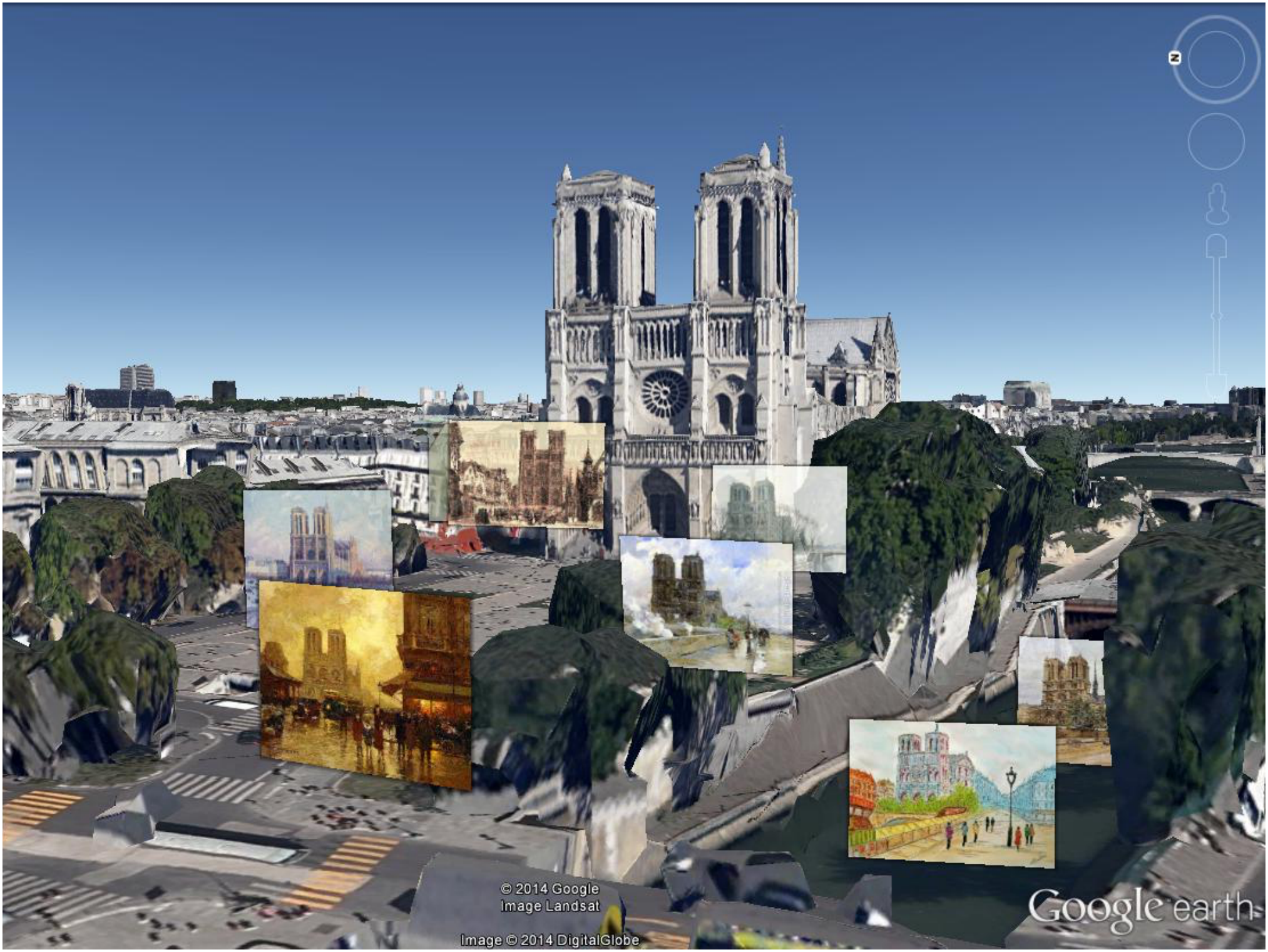

Example I.: Localize non-photographic depictions

Inputs: paintings, drawings, historical photographs, reference 3D model

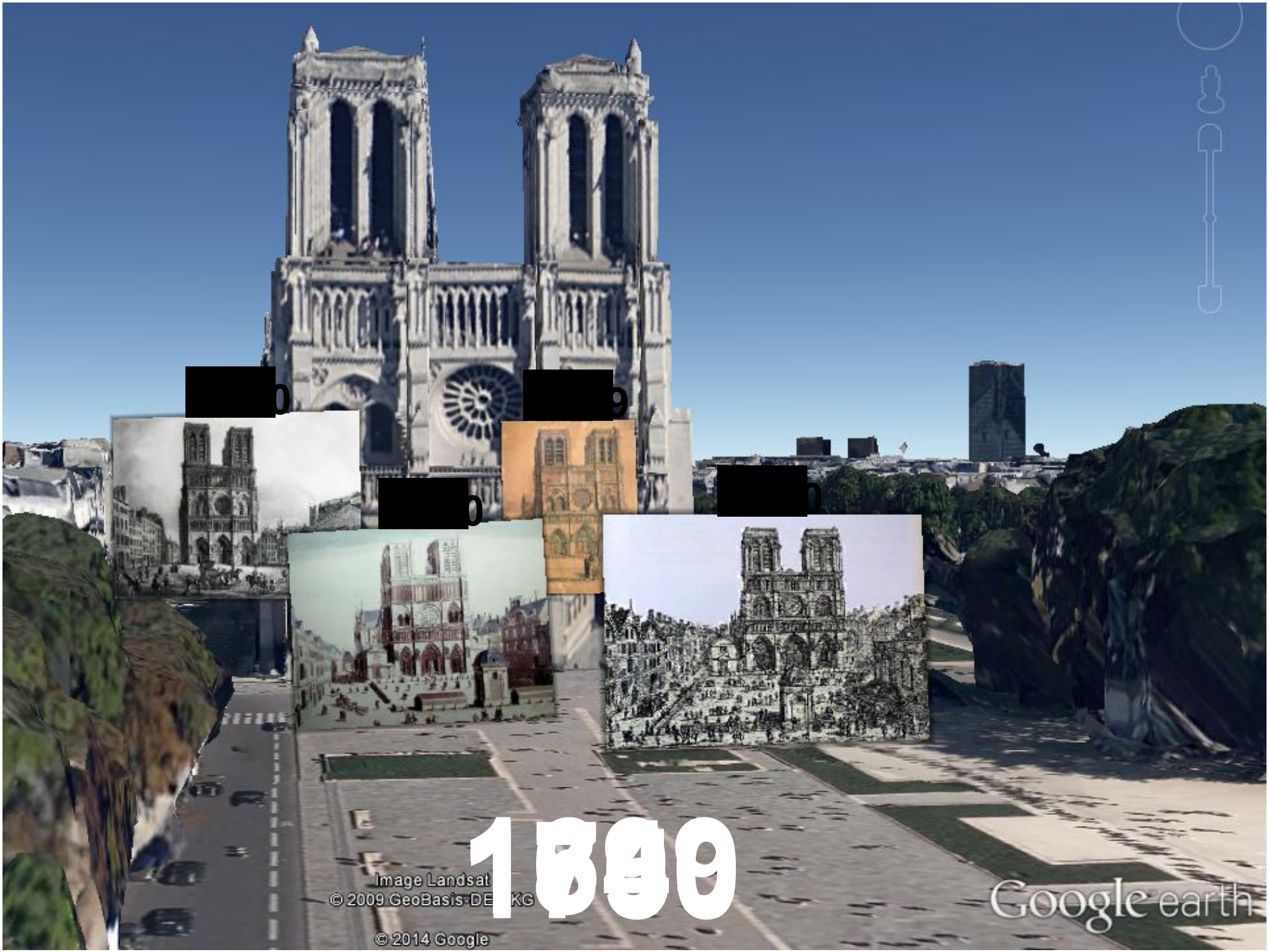
Output: recovered artist/camera viewpoints

[Aubry, Russell, Sivic, 2013]
http://www.di.ens.fr/willow/research/painting_to_3d/

Geo-localization of historical and non-photographic depictions


© 2014 Google
Image Landsat
Image © 2014 DigitalGlobe
Google earth

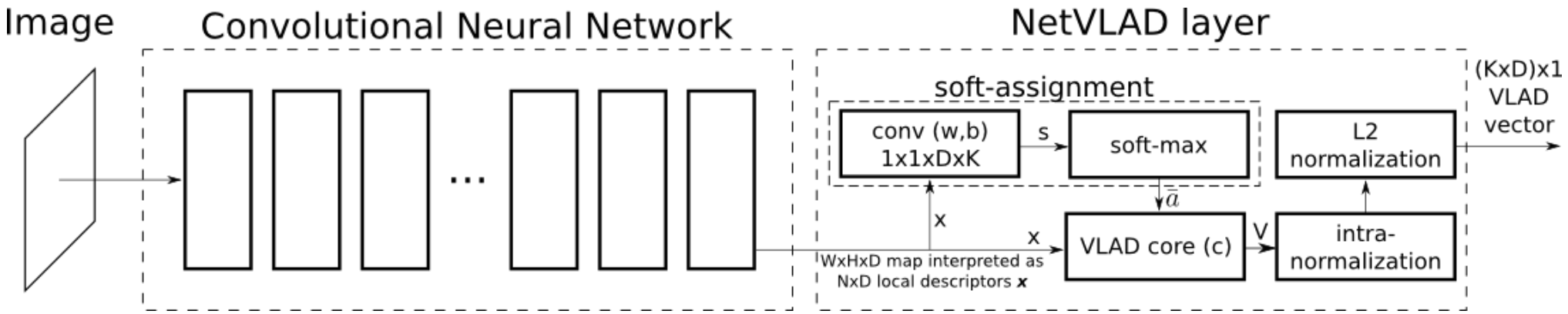
“History-lapse”


1900
Image Landsat
© 2009 GeoBasis-DE/BKG
© 2014 Google
Google earth

Example: Visual localization in changing conditions

- [Sattler et al., arXiv:1707.09092]

NetVLAD: CNN architecture for place recognition



(a) Mobile phone query

(b) Retrieved image of same place

[Arandjelovic, Gronat, Torii, Pajdla, Sivic, 2016]
http://www.di.ens.fr/willow/research/netvlad/

Convolutional neural networks for correspondence and instance-level recognition

Still an active area of research with some successes.

Instance level matching and retrieval:

Babenko et al., ECCV 2014
Razavian et al., ArXiv 2014
Azizpour et al., ArXiv 2014
Babenko and Lempitsky, ICCV 2015
Gong et al., ECCV 2014
Altwaijry et al., CVPR 2015
Arandjelovic et al., CVPR 2016.
Radenovic and Chum, ECCV 2016.
A Gordo, J Almazan, J Revaud, D Larlus, ECCV 2016.

Patch descriptors and correspondence:

Verdie, Kwank, Fua and Lepetit, CVPR 2015
Fischer, A Dosovitskiy and T Brox, Arxiv, 2015
Simo-Serra, Trulls, Ferraz, Kokkinos, Fua, and Moreno-Noguer, CVPR 2015
Han, Leung, Jia, Sukthankar, and C Berg, CVPR 2015
Zagoruyko and Komodakis, CVPR 2015
Gwak, Savarese and Chandraker, ECCV 2016
KM Yi, E Trulls, V Lepetit, P Fua, ECCV 2016
Balntas, Johns, Tang, and Mikolajczyk, CVPR 2016
A Mishchuk, D Mishkin, F Radenovic, J Matas, NIPS 2017

Dense correspondence for motion estimation

Fischer, Dosovitskiy, Ilg, Häusser, Hazırbaş, Golkov, van der Smagt, Cremers and Brox, ICCV 2015
T Zhou, M Brown, N Snavely, DG Lowe, CVPR 2017