9th Russian Summer School in Information Retrieval (RuSSIR)
August 24-28, 2015
St Petersburg, Russia

Visual object recognition and localization

Part 2:
Instance-level recognition

Ivan Laptev

ivan.laptev@inria.fr

INRIA, WILLOW, ENS/INRIA/CNRS UMR 8548
Laboratoire d'Informatique, Ecole Normale Supérieure, Paris, France

Includes slides from: Mark Everingham, Svetlana Lazebnik, Jean Ponce, Cordelia Schmid, Steven M. Seitz, Josef Sivic, A. Torralba and Andrew Zisserman

Image matching and recognition with local features

The goal: establish correspondence between two or more images

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

P : 3×4 matrix

$\mathbf{X}$: 4-vector

$\mathbf{x}$: 3-vector

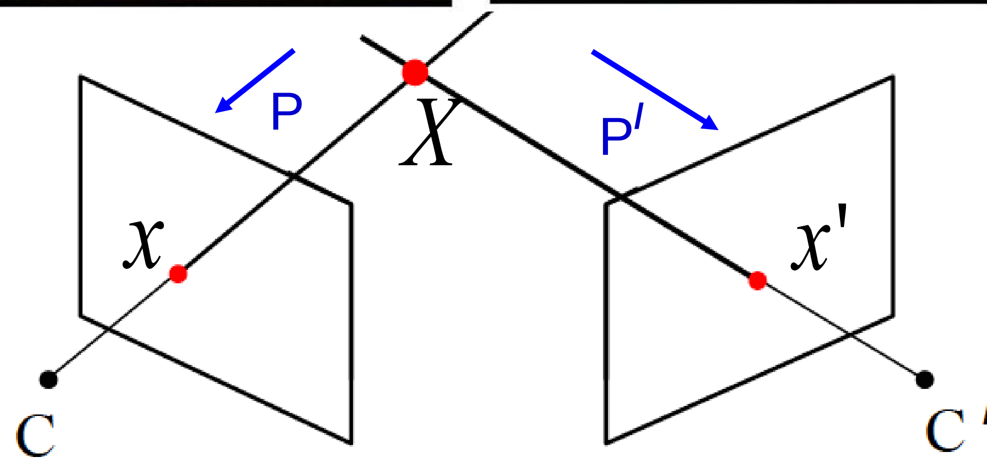


Image points x and x' are in correspondence if they are projections of the same 3D scene point X.

Images courtesy A. Zisserman

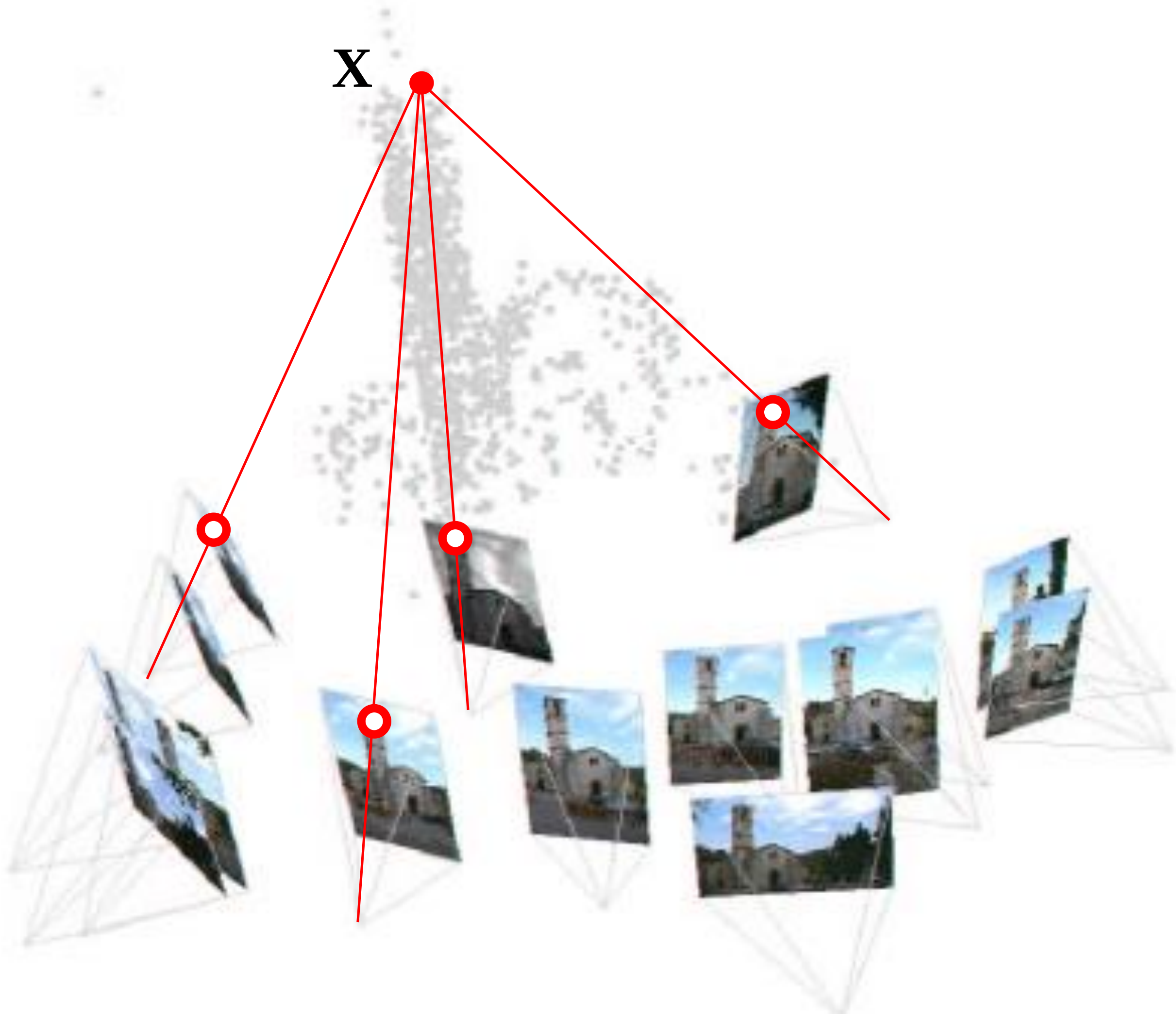
Example I: Wide baseline matching and 3D reconstruction

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]

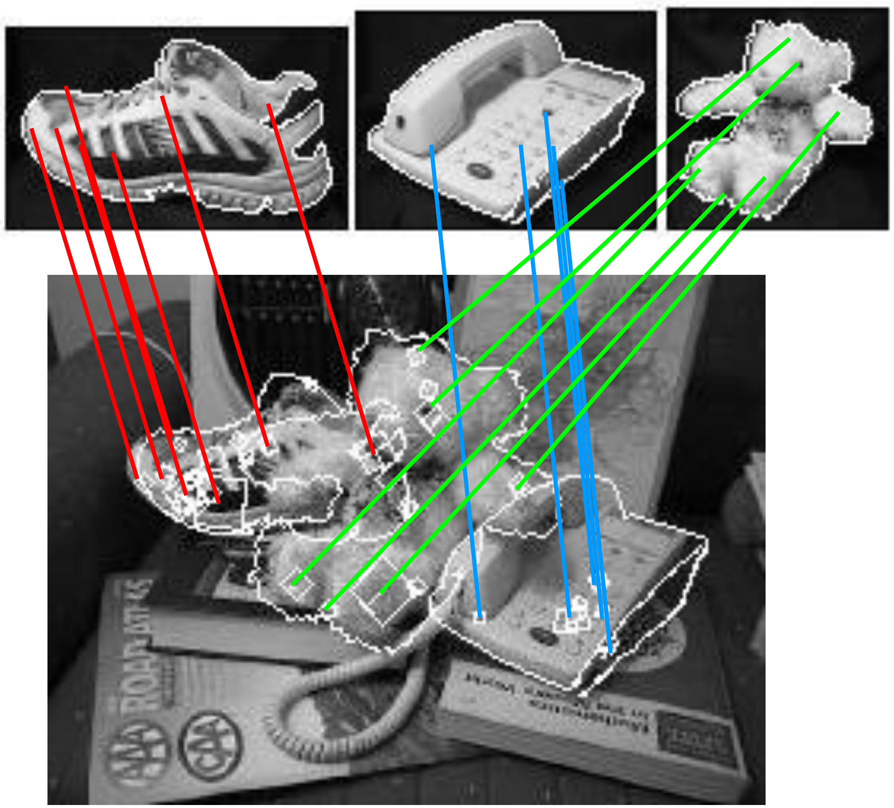
[Agarwal, Snavely, Simon, Seitz, Szeliski, ICCV'09] – Building Rome in a Day

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

Example II: Object recognition

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

Model database

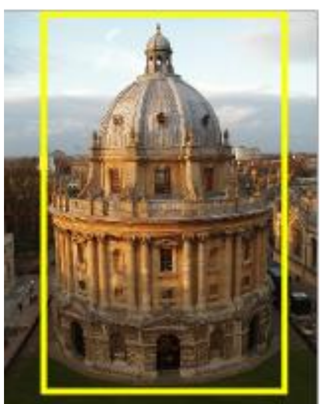


Target image

[D. Lowe, 1999]

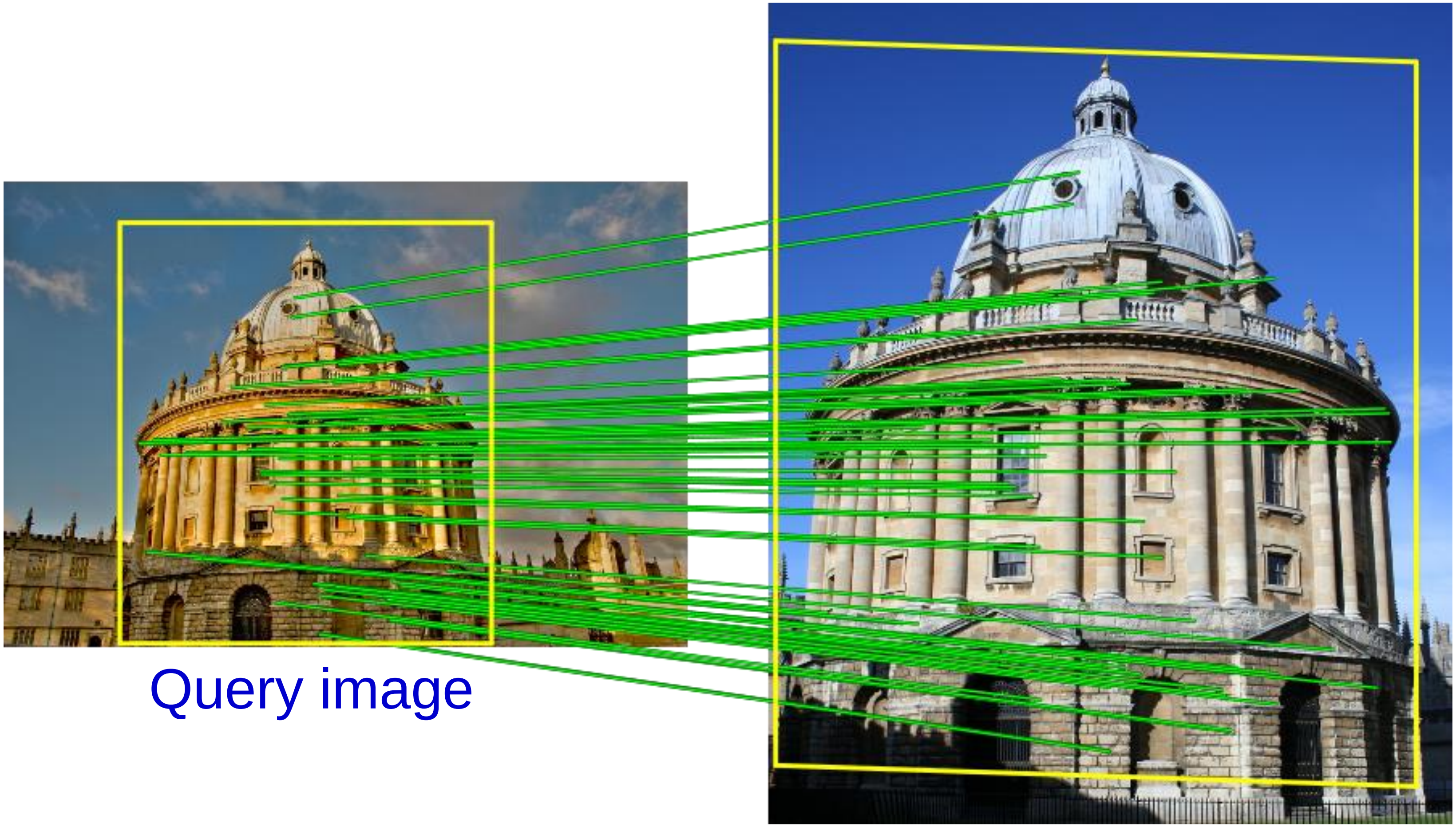
Example III: Visual search

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.



Applications

Take a picture of a product or advertisement

→ find relevant information on the web

[Pixee – Milpix]

Applications

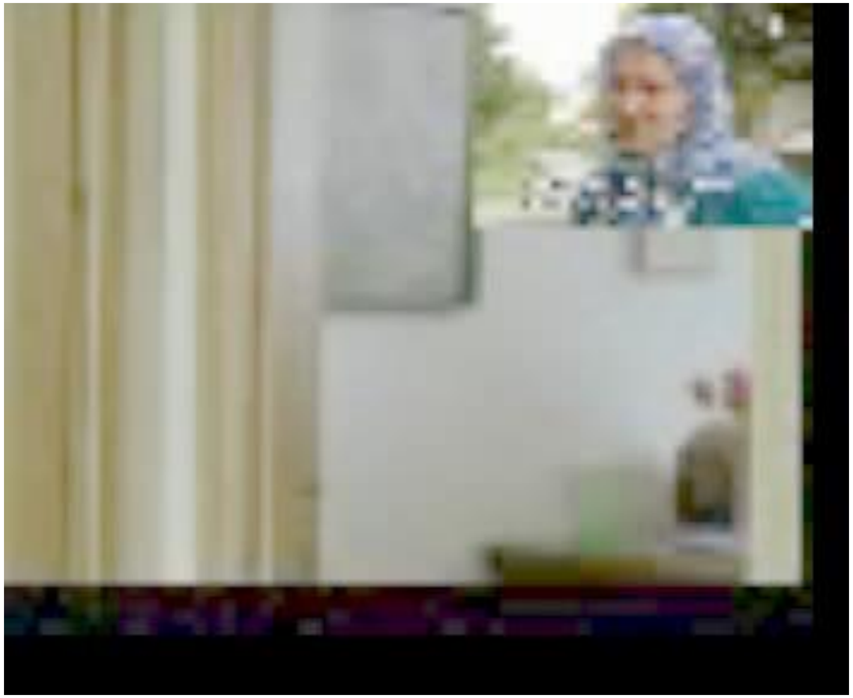
Finding stolen/missing objects in a large collection

•
•
•

Applications

Copy detection for images and videos

Query video

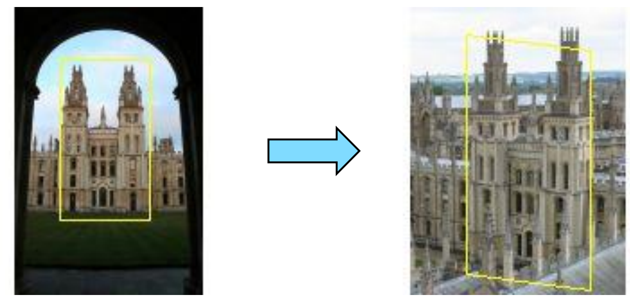


Search in 200h of video

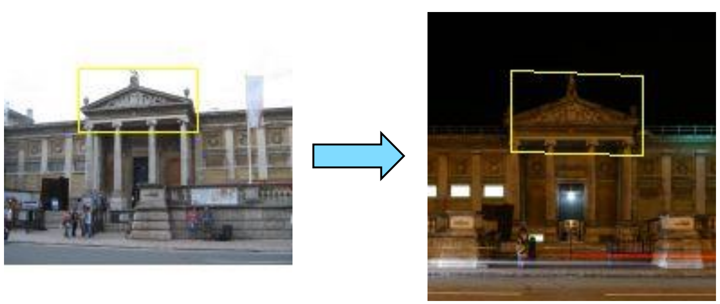
Why is it difficult?

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

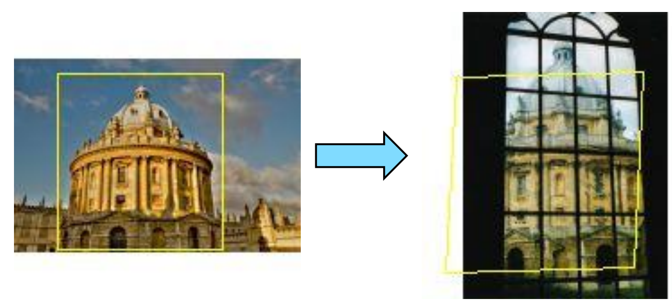
Scale



Viewpoint



Lighting



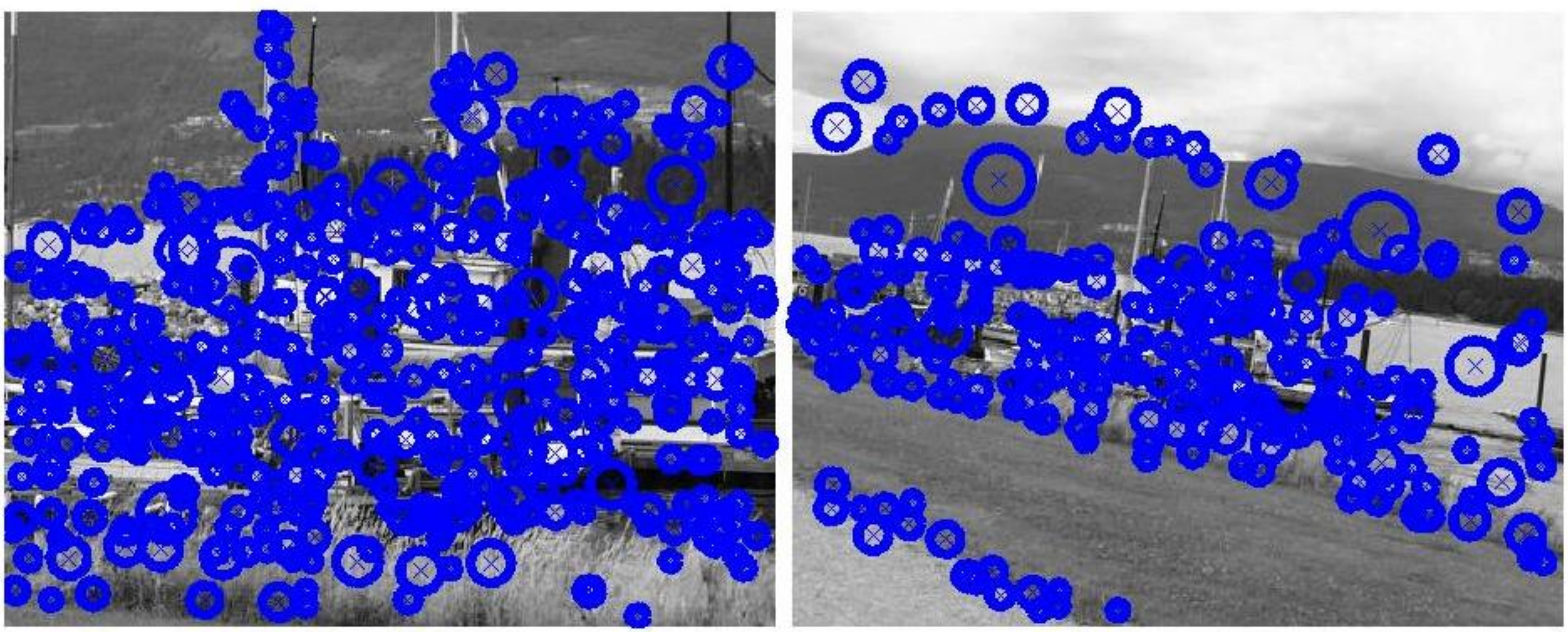
Occlusion

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

How does it work?

Approach:

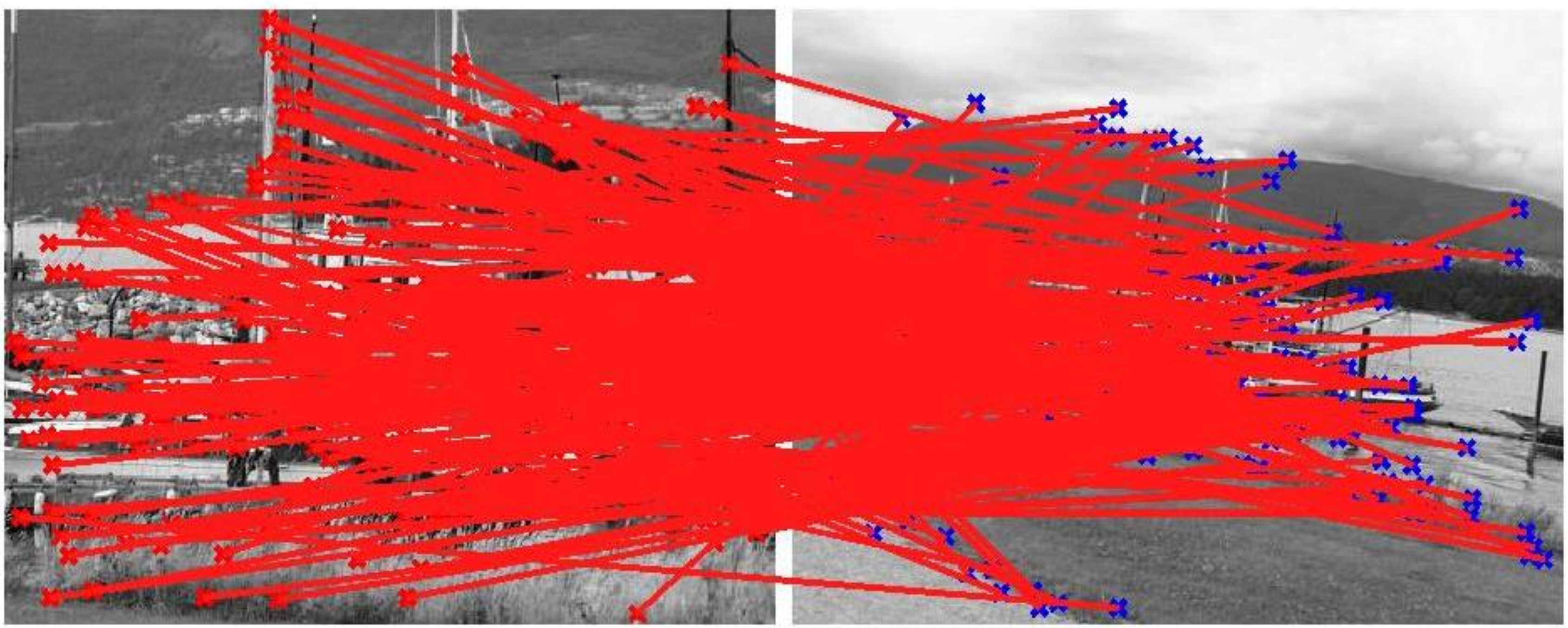
- Compute scale / affine co-variant *local features*
- Estimate pairwise best matches between *local features*
- Enforce geometric constraints between *local features*



How does it work?

Approach:

- Compute scale / affine co-variant *local features*
- Estimate pairwise best matches between *local features*
- Enforce geometric constraints between *local features*



How does it work?

Approach:

- Compute scale / affine co-variant *local features*
- Estimate pairwise best matches between *local features*
- Enforce geometric constraints between *local features*

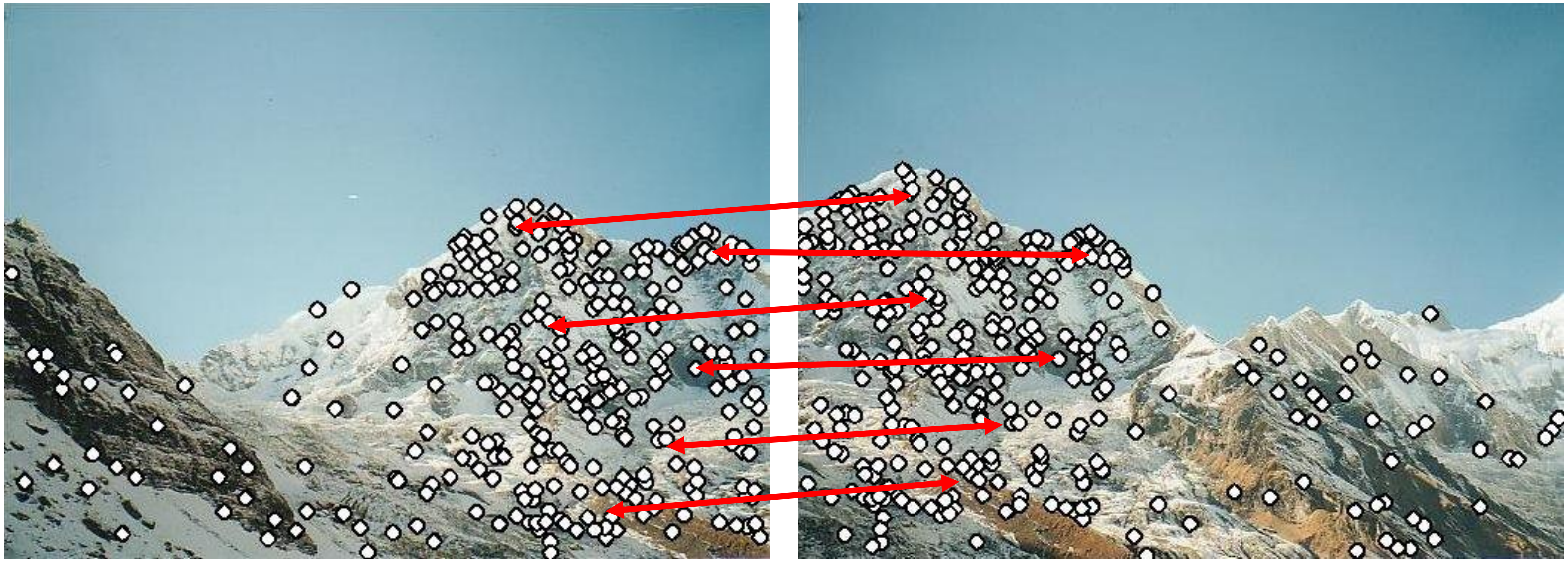
Why extract features?

- 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?

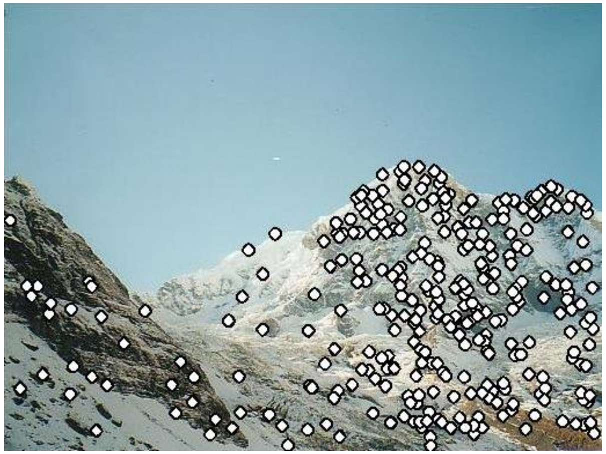
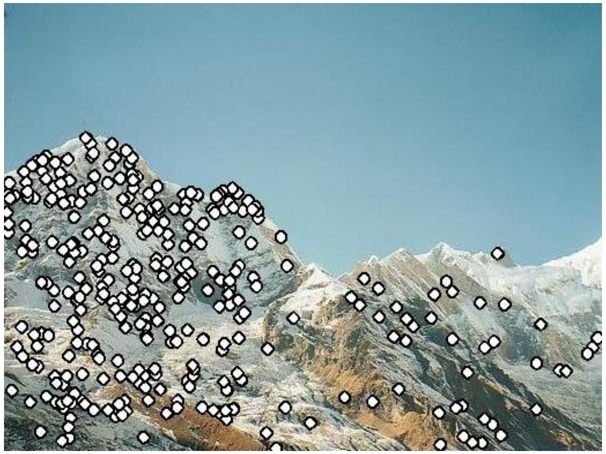
Step 1: extract features

Step 2: match features

Step 3: align images

Slide: S. Lazebnik

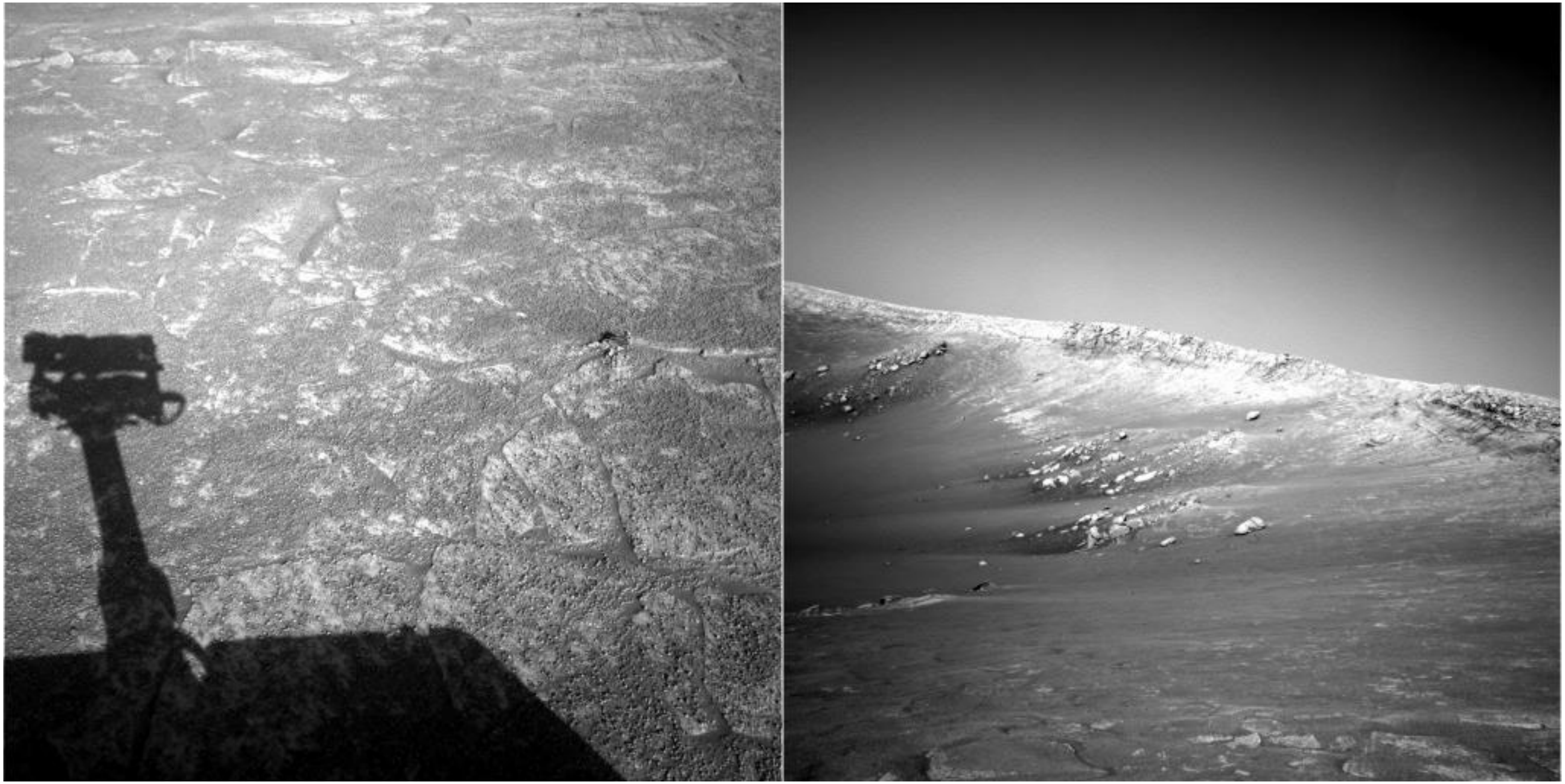
Characteristics of good features

- 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

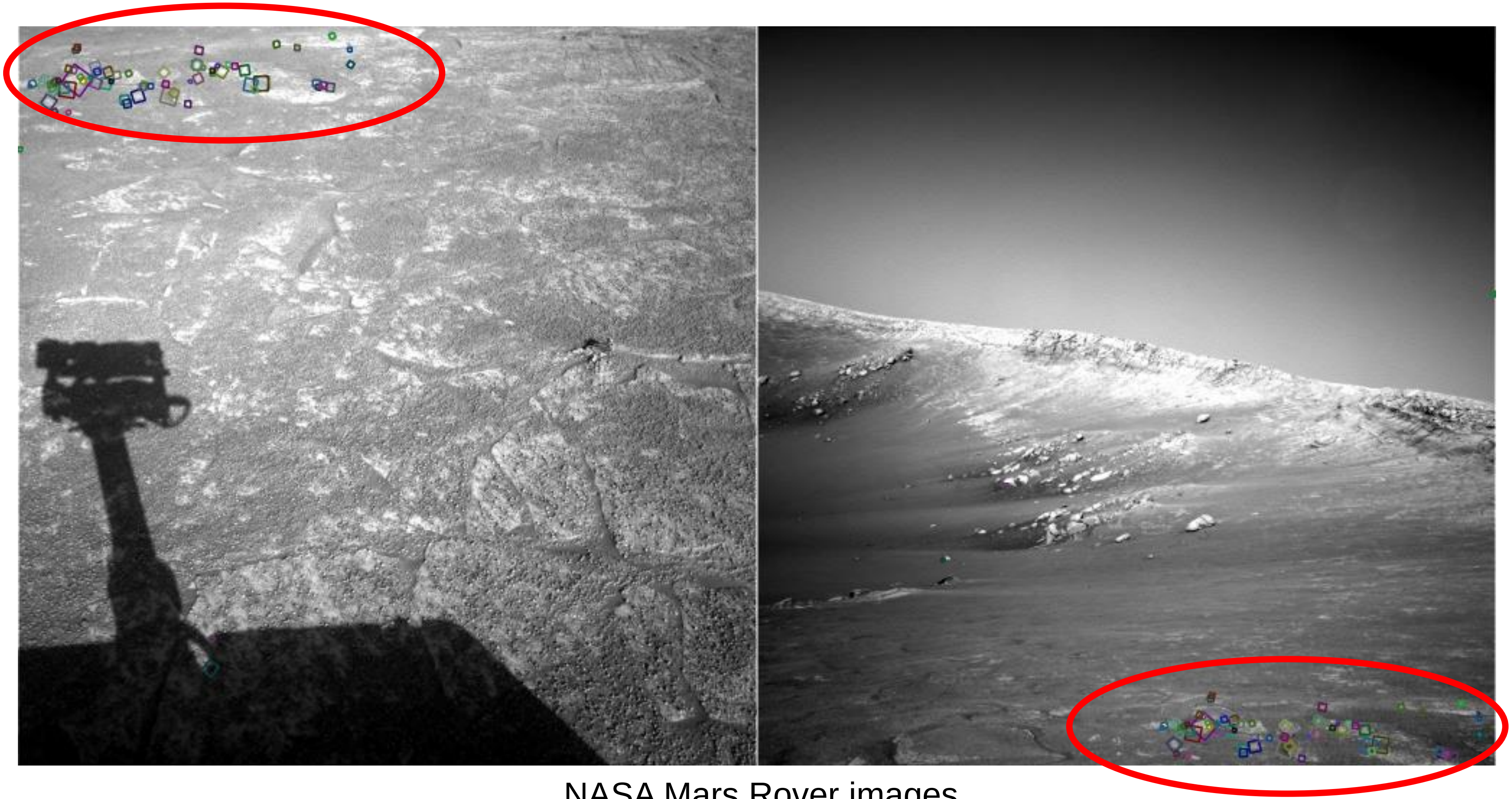
A hard feature matching problem



NASA Mars Rover images

Slide: S. Lazebnik

Answer below (look for tiny colored squares…)

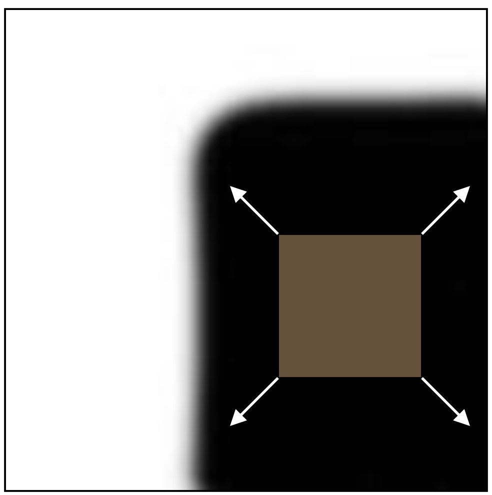


NASA Mars Rover images
with SIFT feature matches
Figure by Noah Snavely

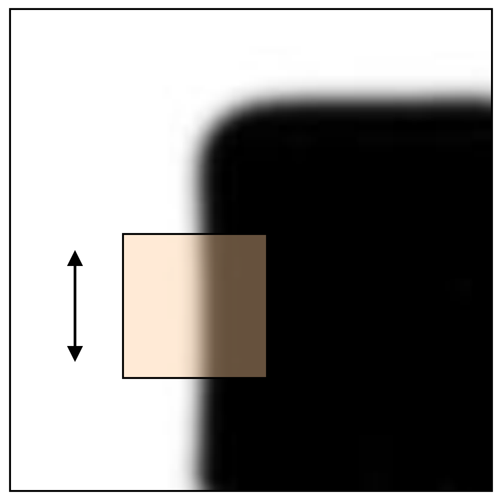
Slide: S. Lazebnik

Corner Detection: Basic Idea

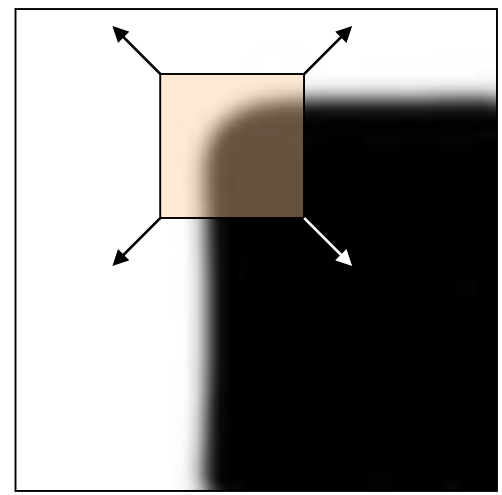
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region:
no change in all directions



"edge":
no change along the edge direction



"corner":
significant change in all directions

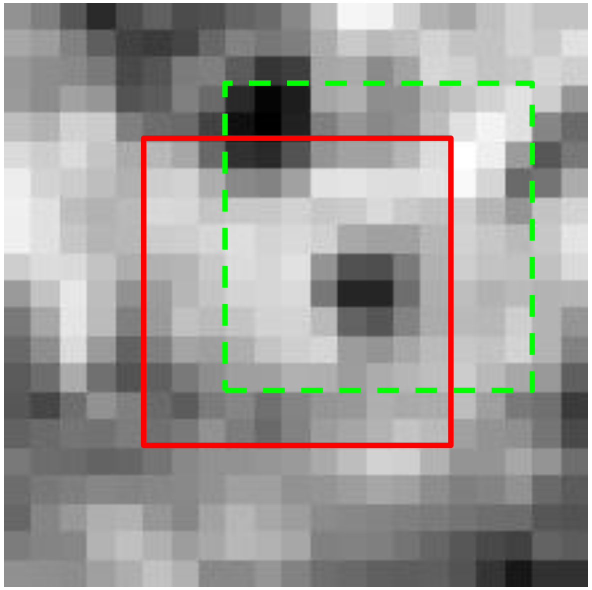
Source: A. Efros

Corner Detection: Mathematics

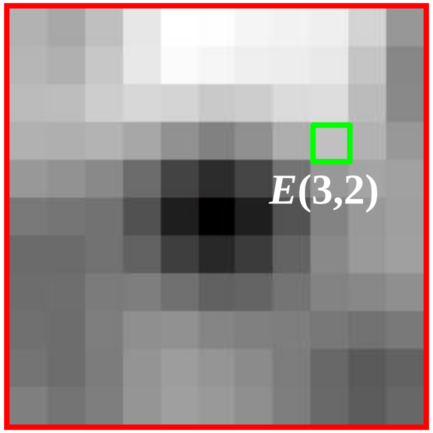
Change in appearance of window *W* for the shift [*u,v*]:

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

$I(x, y)$



$E(u, v)$



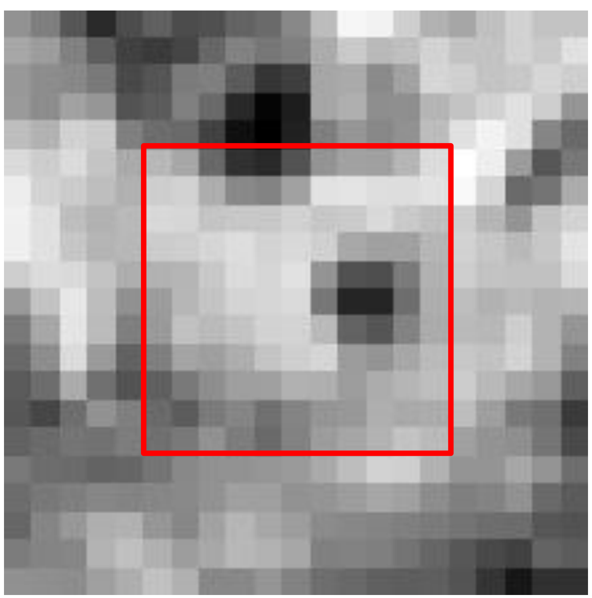
Slide: S. Lazebnik

Corner Detection: Mathematics

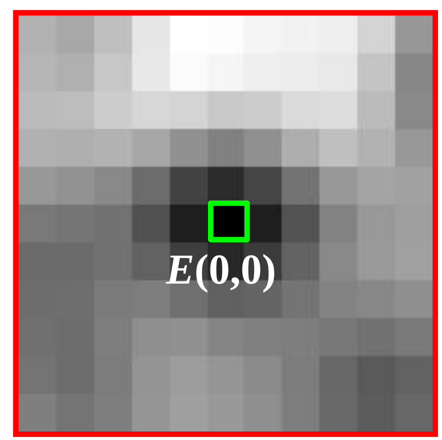
Change in appearance of window *W* for the shift [*u,v*]:

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

$I(x, y)$



$E(u, v)$



Slide: S. Lazebnik

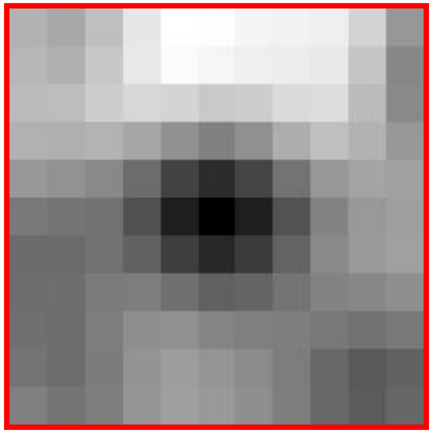
Corner Detection: Mathematics

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

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

Corner Detection: Mathematics

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

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

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

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

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:

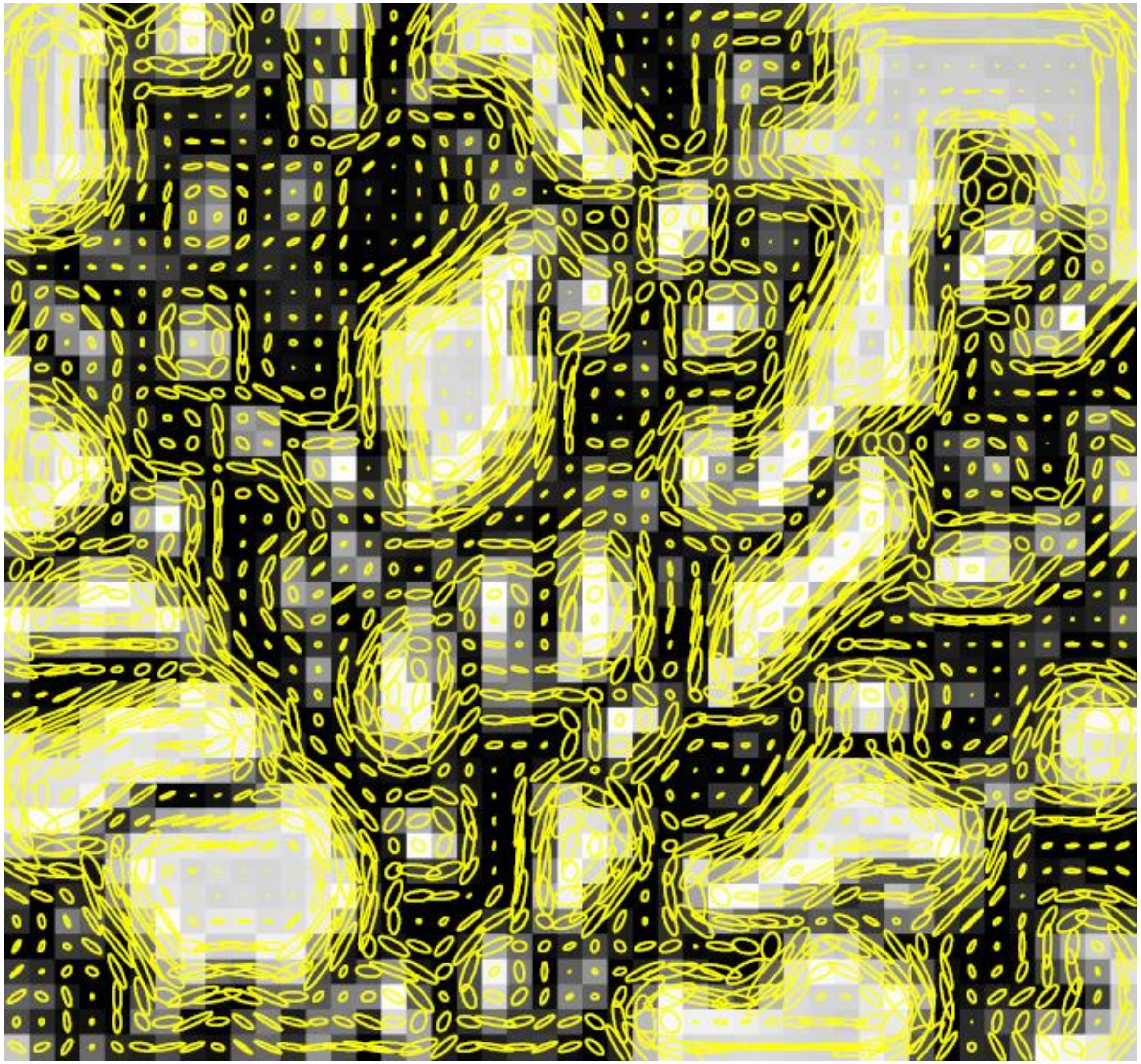
$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Slide: S. Lazebnik

Visualization of second moment matrices

Slide: S. Lazebnik

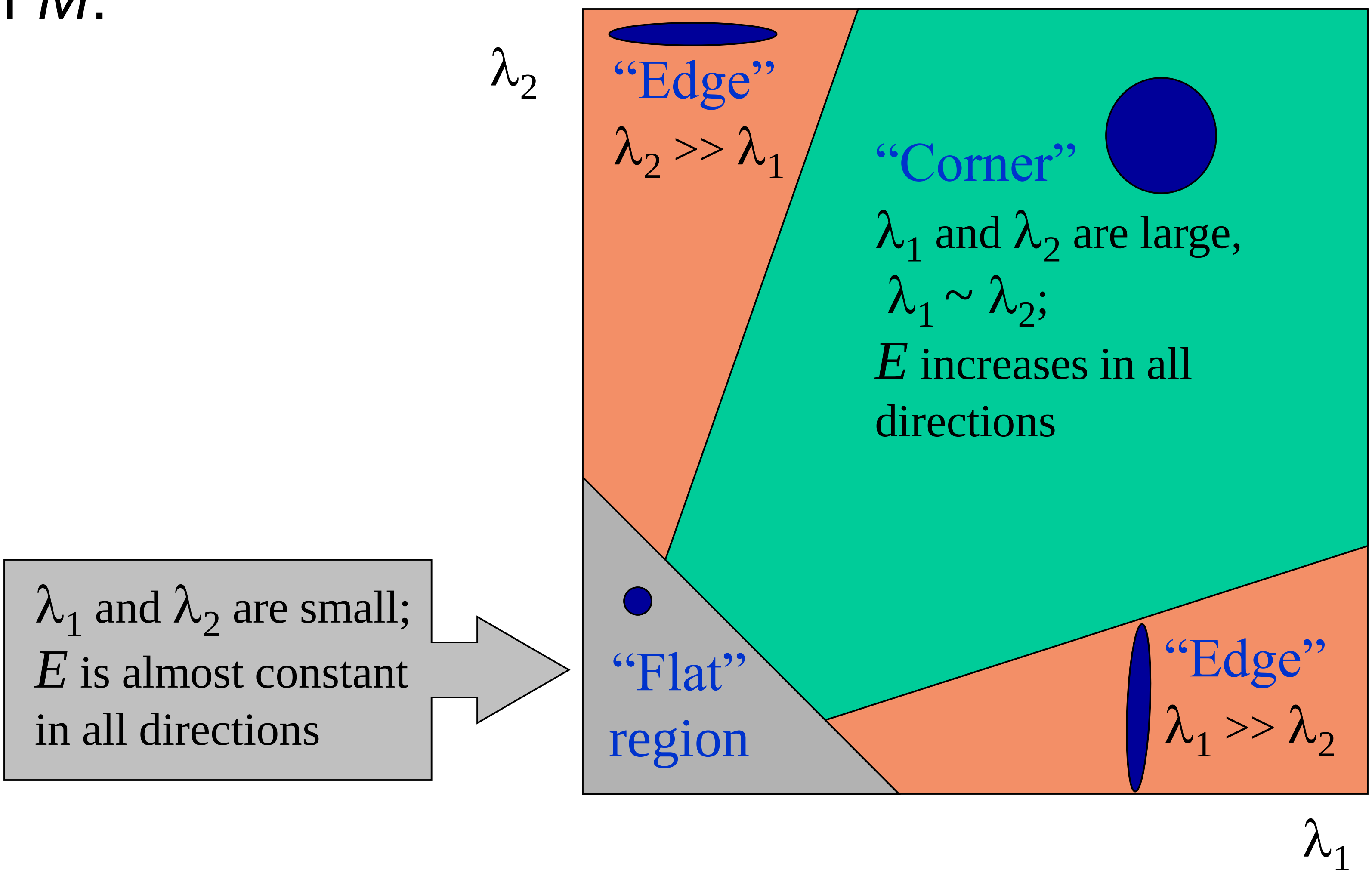
Visualization of second moment matrices



Slide: S. Lazebnik

Interpreting the eigenvalues

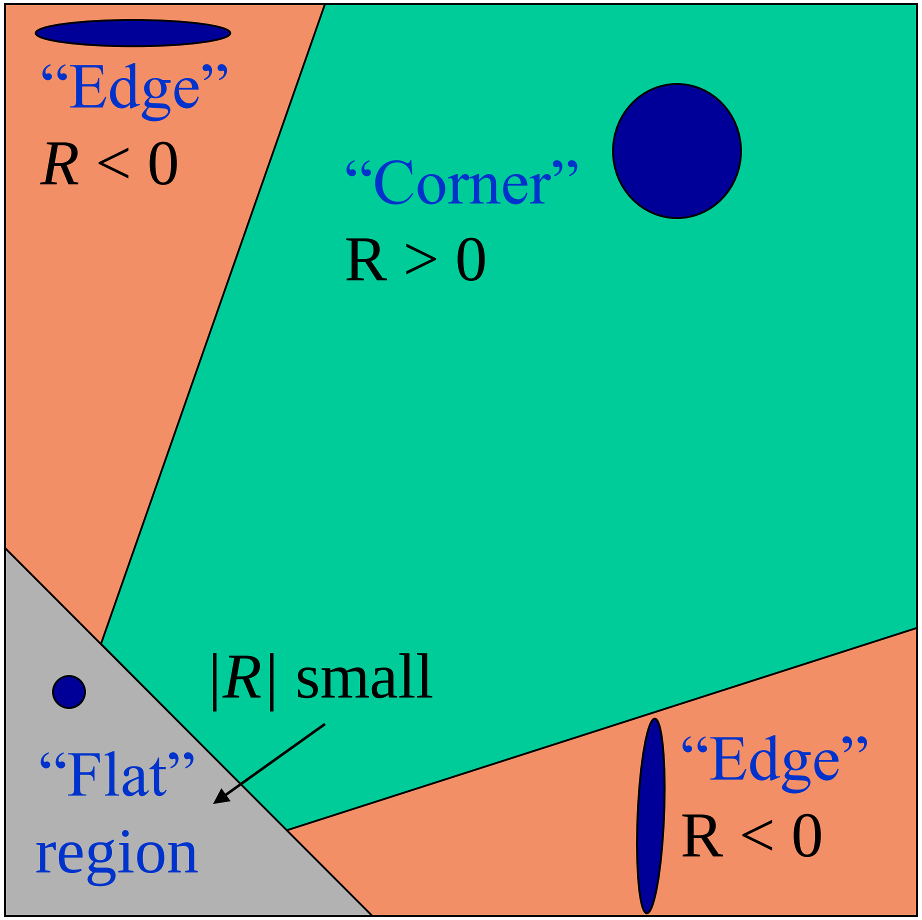
Classification of image points using eigenvalues of *M*:



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)



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.

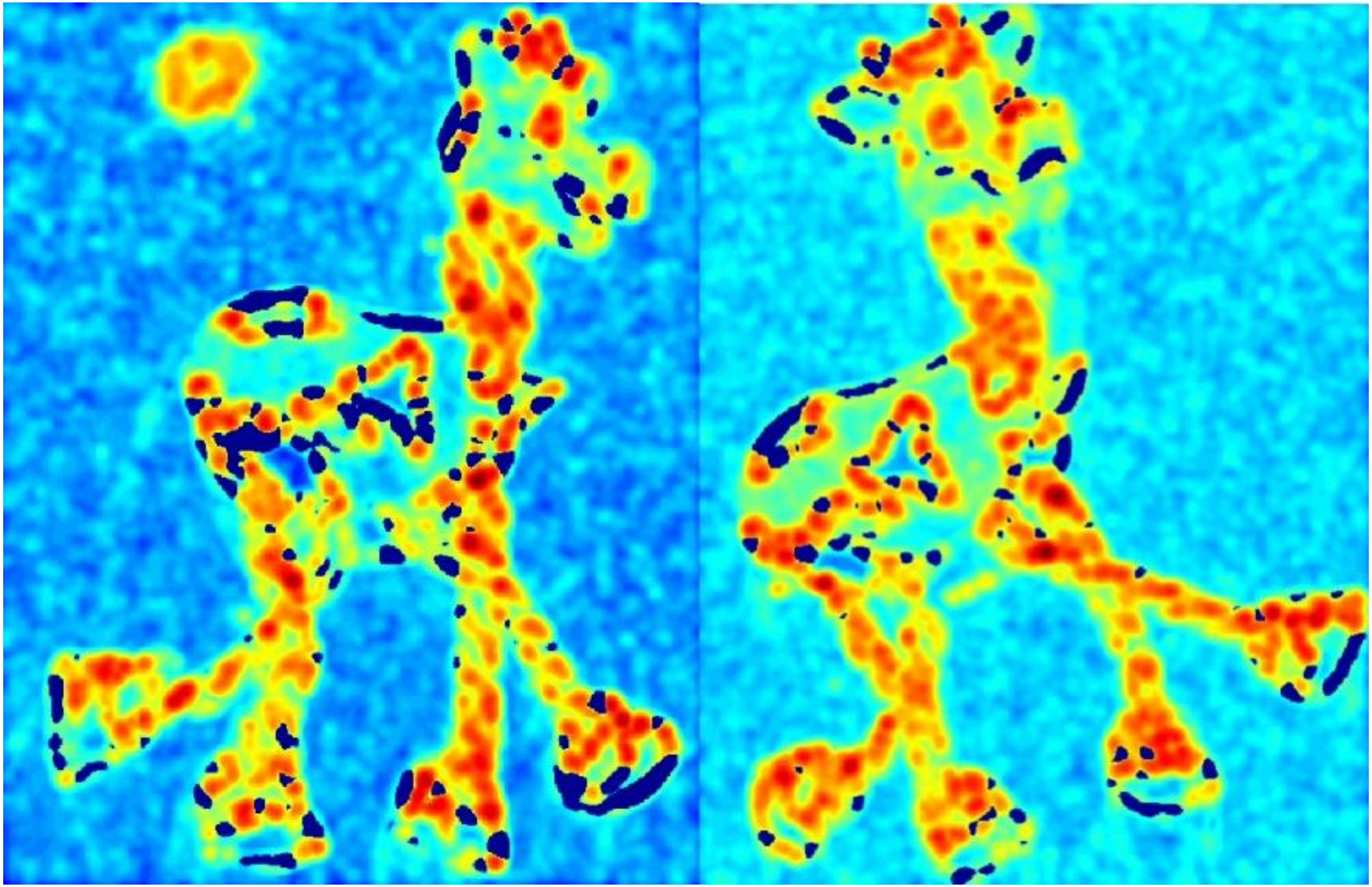
Slide: S. Lazebnik

Harris Detector: Steps

Slide: S. Lazebnik

Harris Detector: Steps

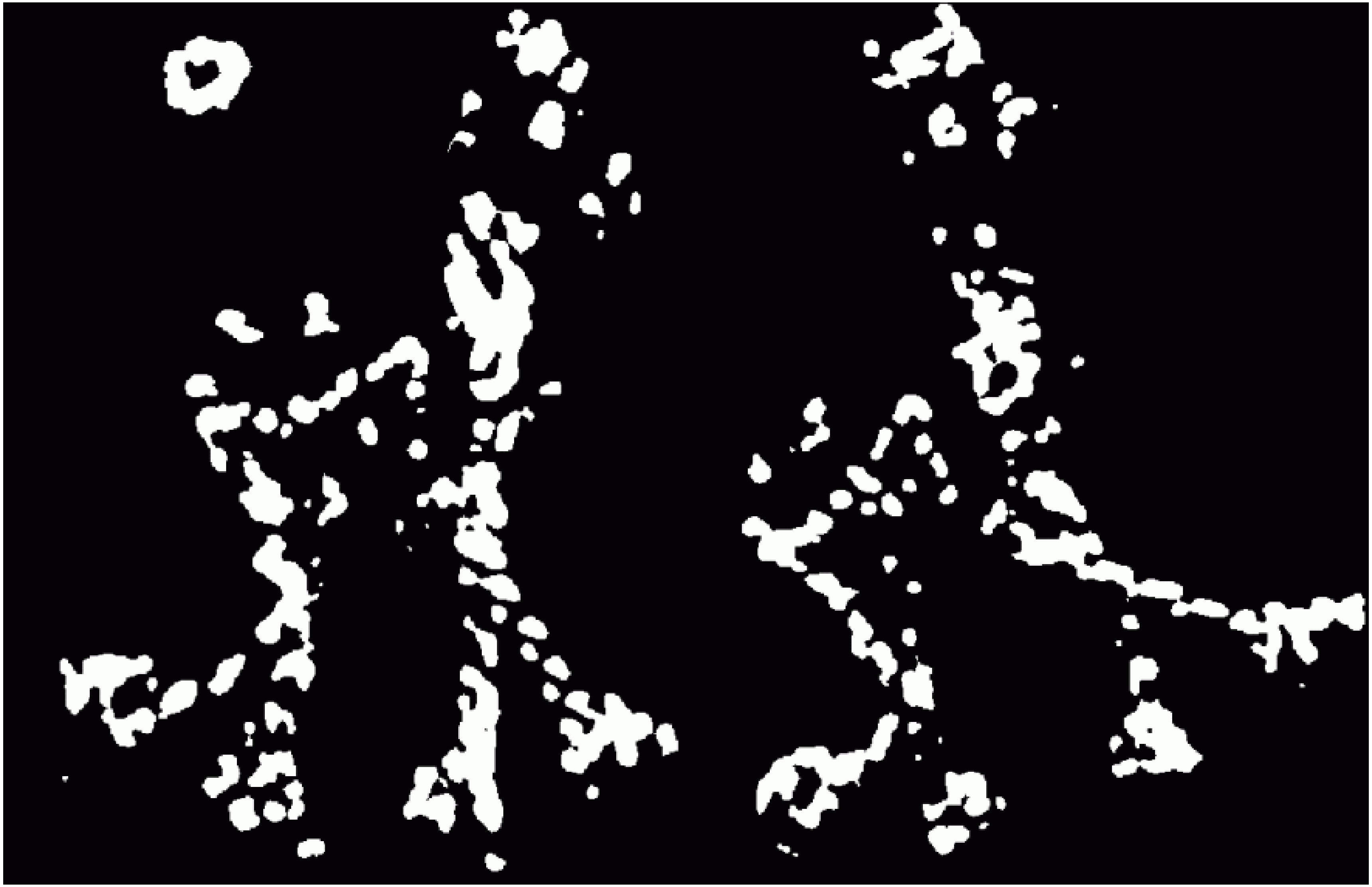
Compute corner response R



Slide: S. Lazebnik

Harris Detector: Steps

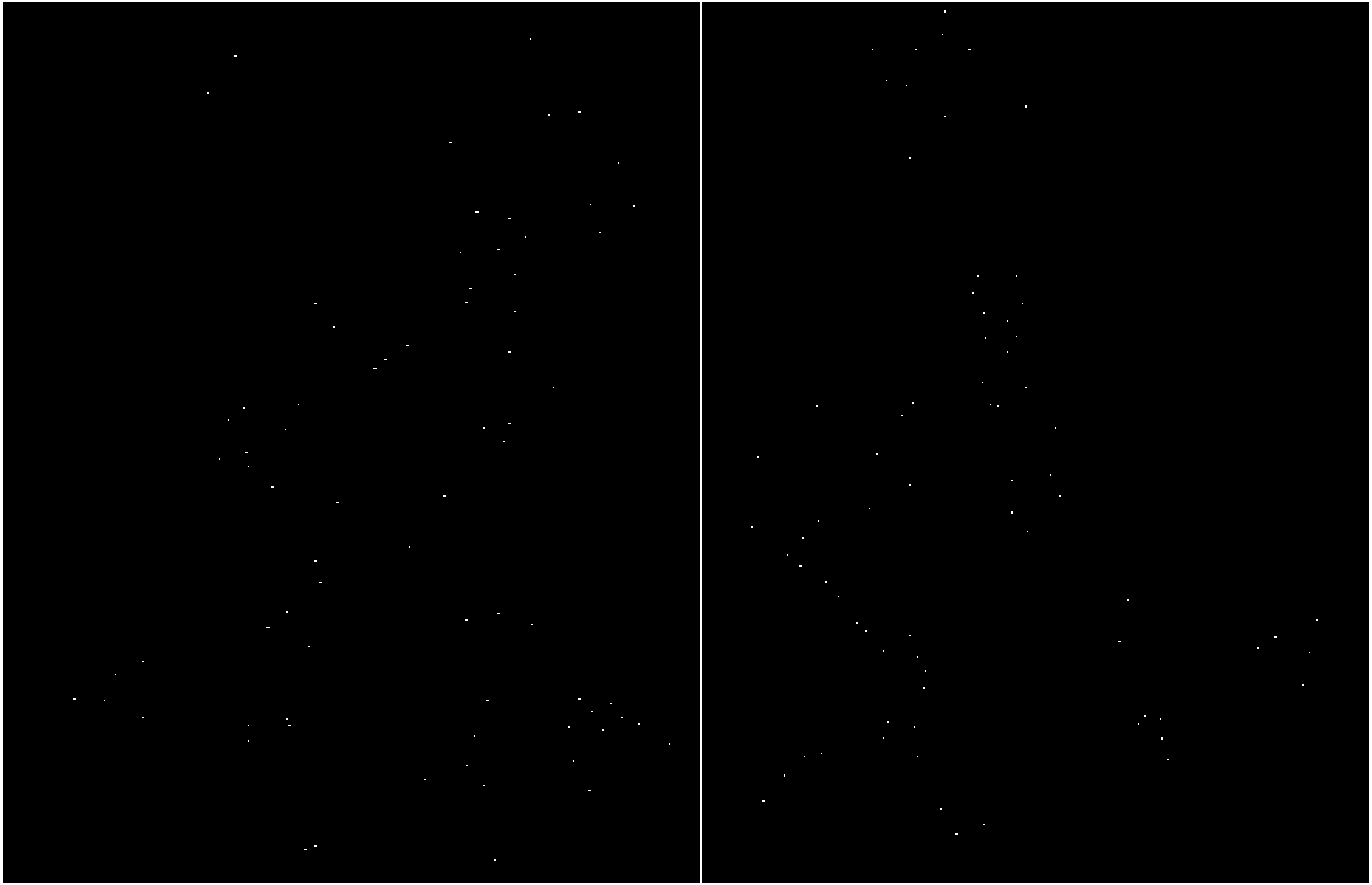
Find points with large corner response: R>threshold



Slide: S. Lazebnik

Harris Detector: Steps

Take only the points of local maxima of R



Slide: S. Lazebnik

Harris Detector: Steps

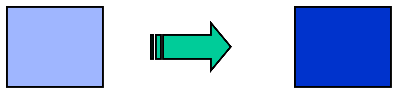
Slide: S. Lazebnik

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

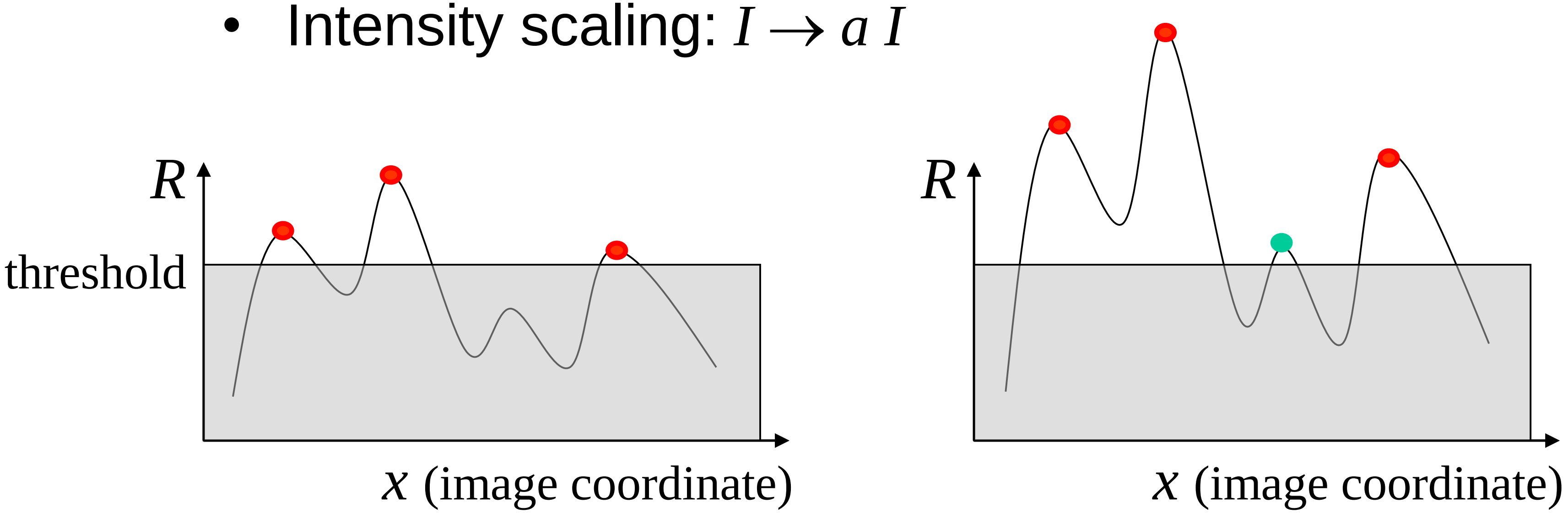
Affine intensity change



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

- Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

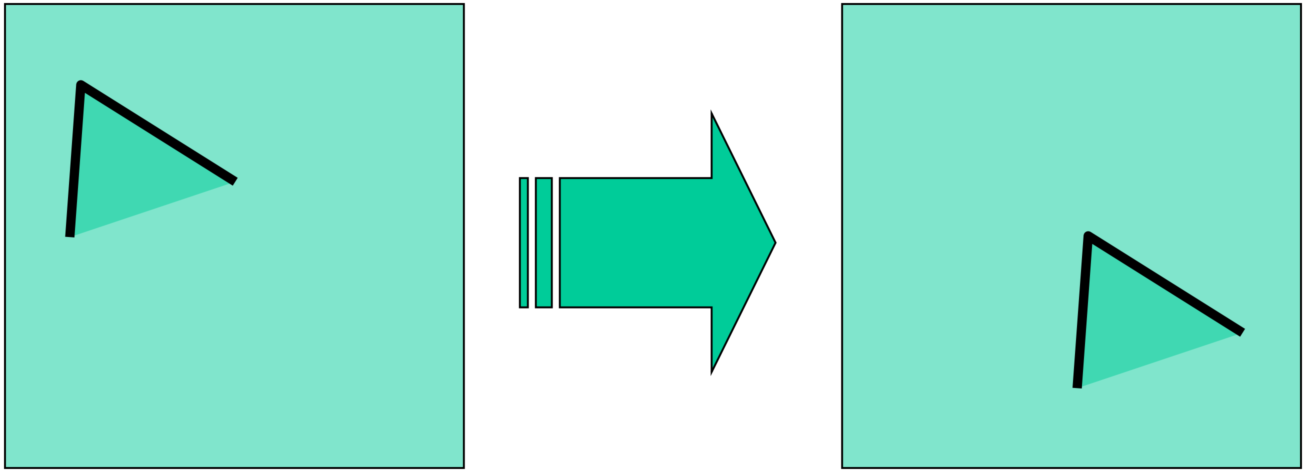
- Intensity scaling: $I \rightarrow a\,I$



Partially invariant to affine intensity change

Slide: S. Lazebnik

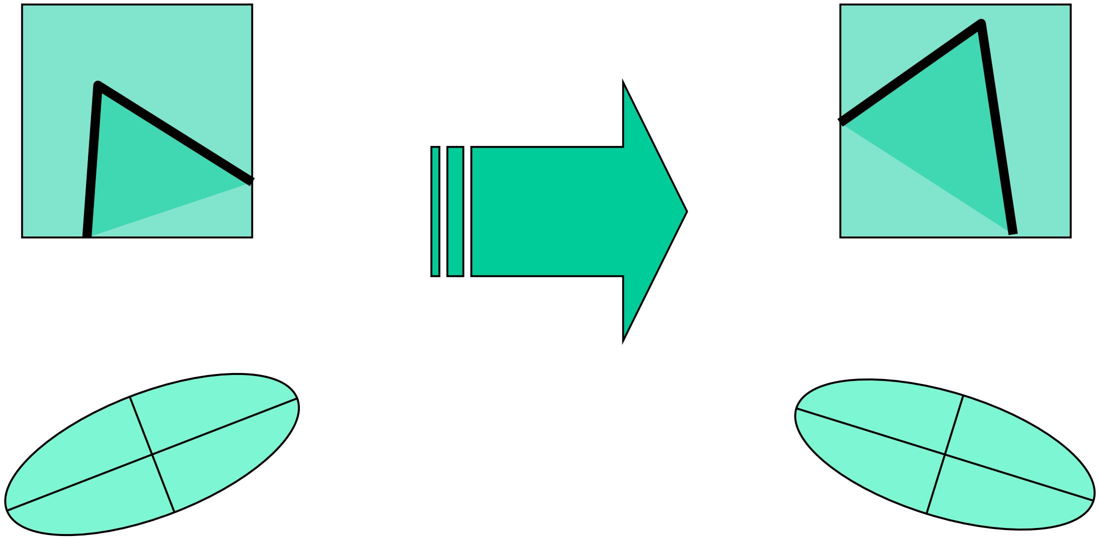
Image translation



- Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation

Image rotation

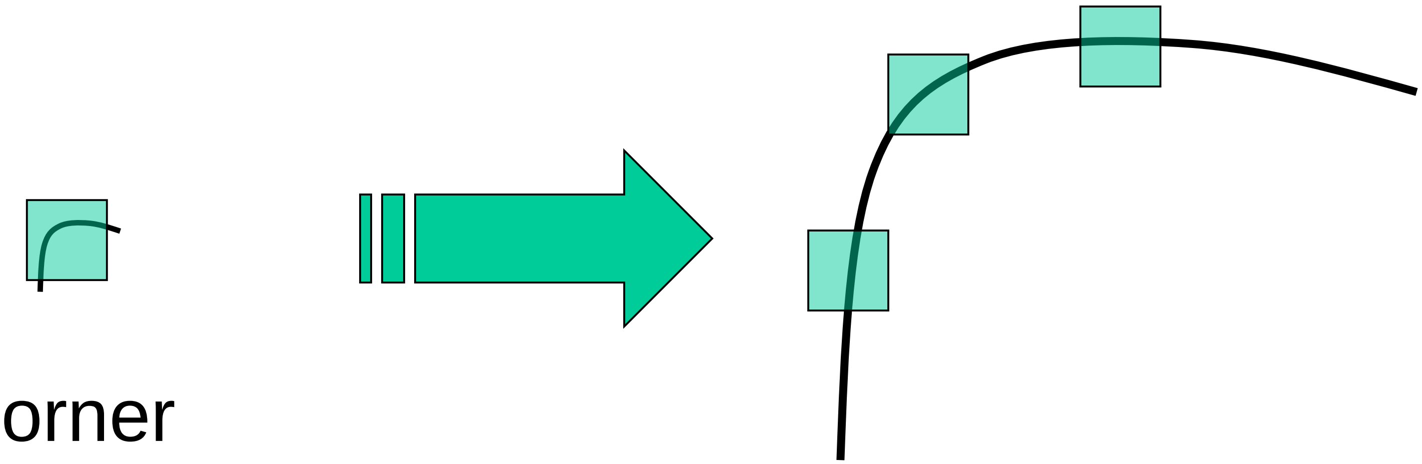


Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation

Slide: S. Lazebnik

Scaling



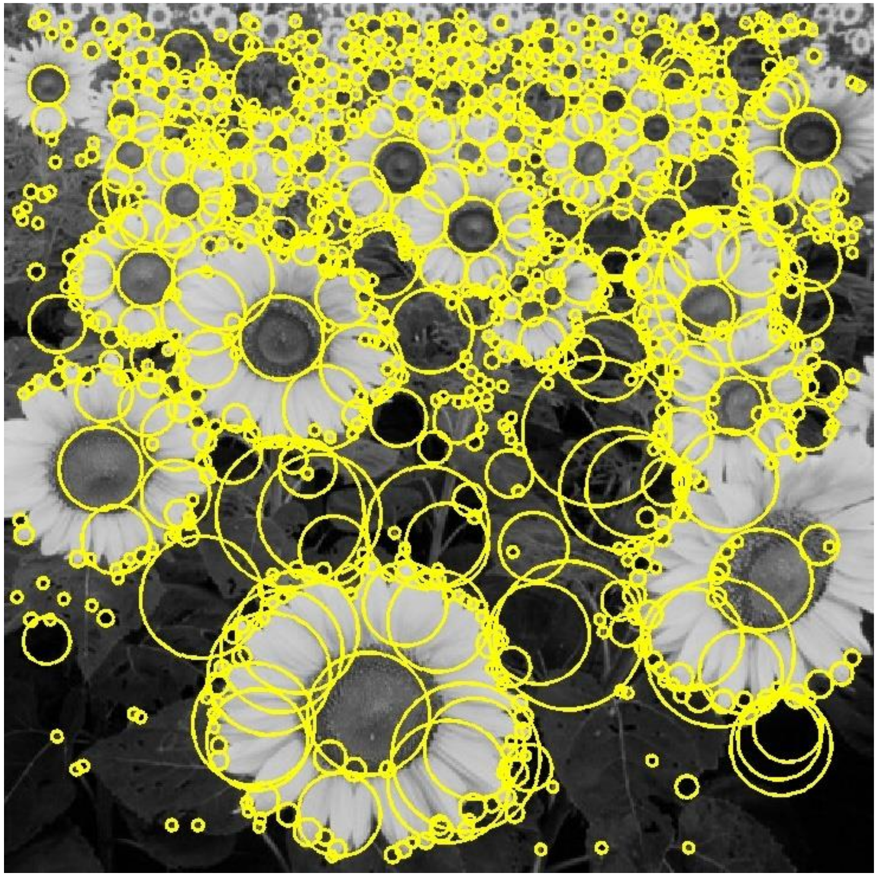
Corner

All points will be classified as edges

Corner location is not covariant to scaling!

Slide: S. Lazebnik

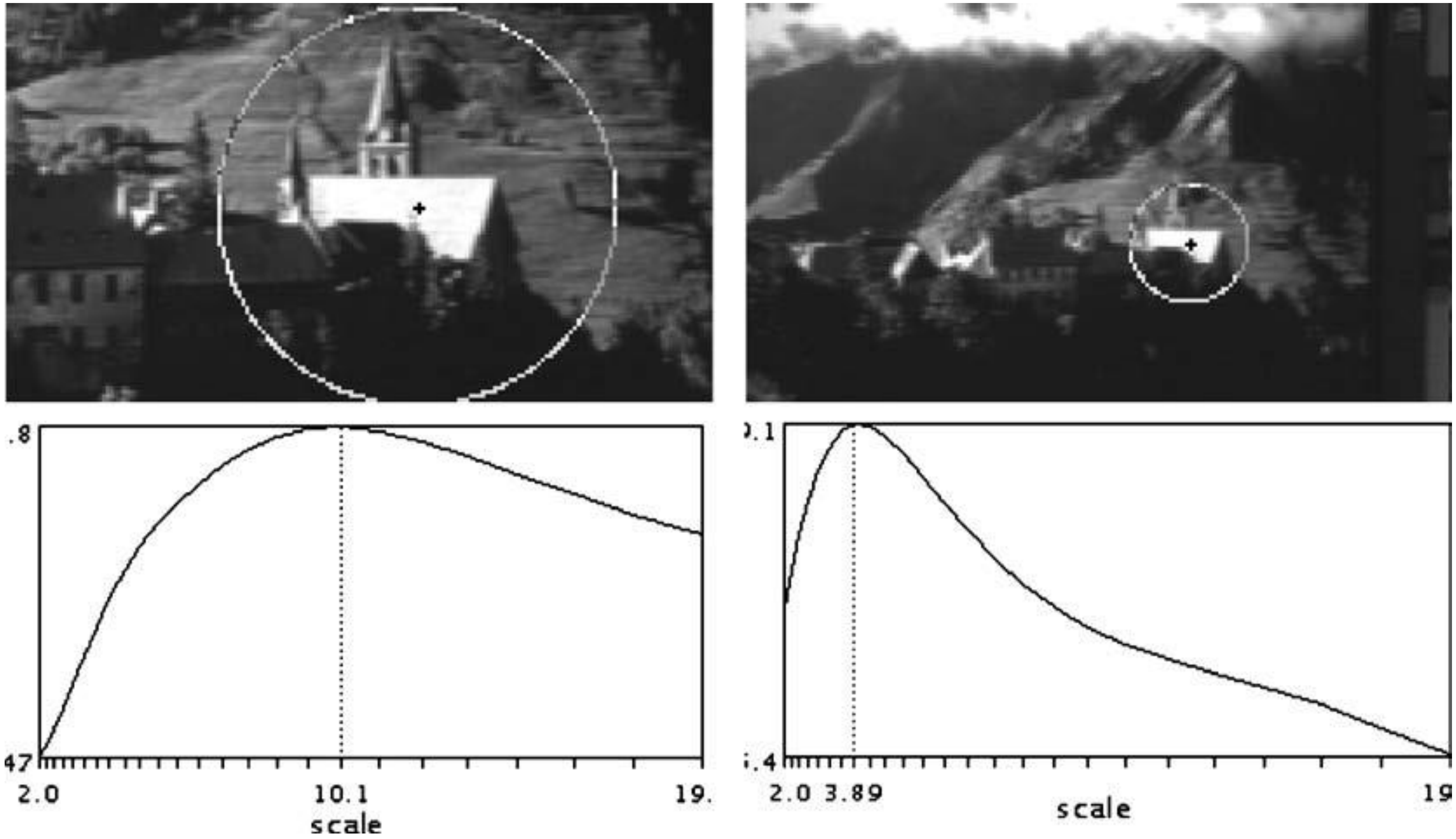
Blob detection



Slide: S. Lazebnik

Feature detection with scale selection

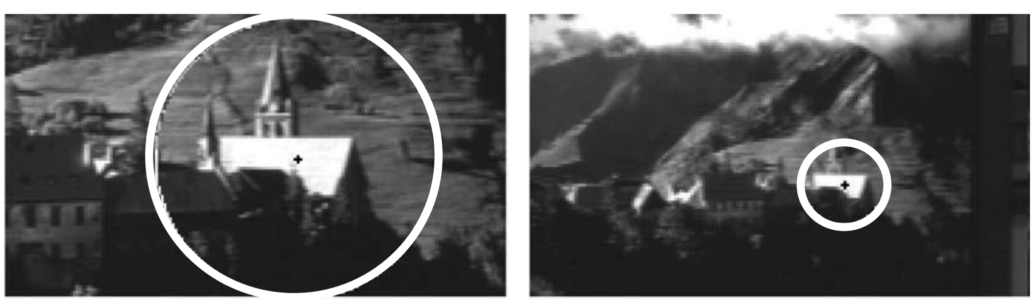
- We want to extract features with characteristic scale that is *covariant* with the image transformation

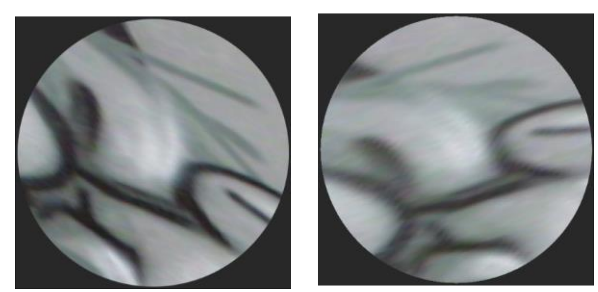


Slide: S. Lazebnik

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

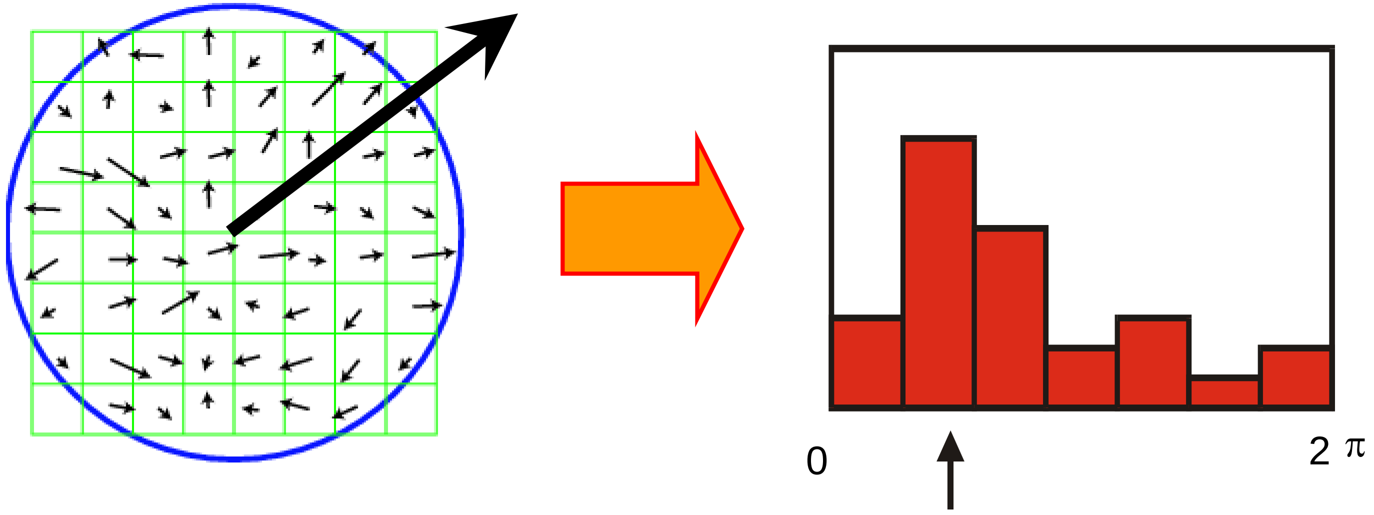




Slide: S. Lazebnik

Eliminating rotation ambiguity

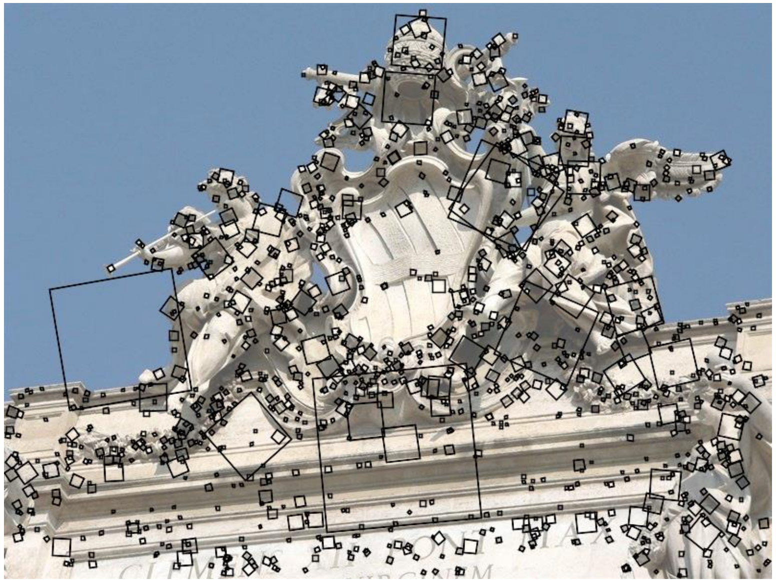
- To assign a unique orientation to circular image windows:
  - Create histogram of local gradient directions in the patch
  - Assign canonical orientation at peak of smoothed histogram



Slide: S. Lazebnik

SIFT features

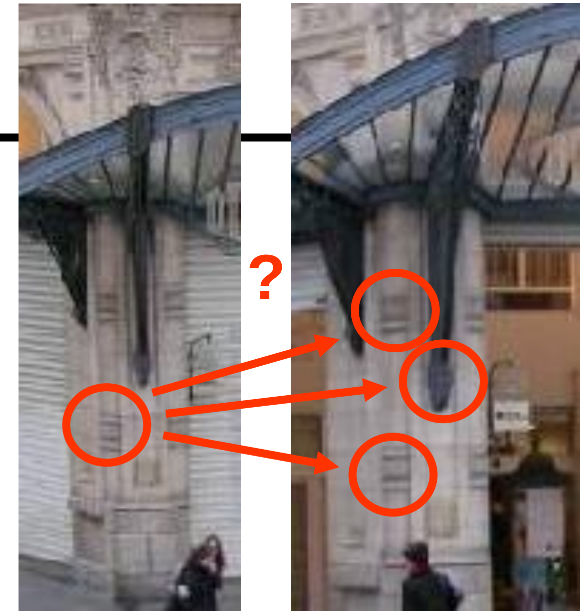
- Detected features with characteristic scales and orientations:

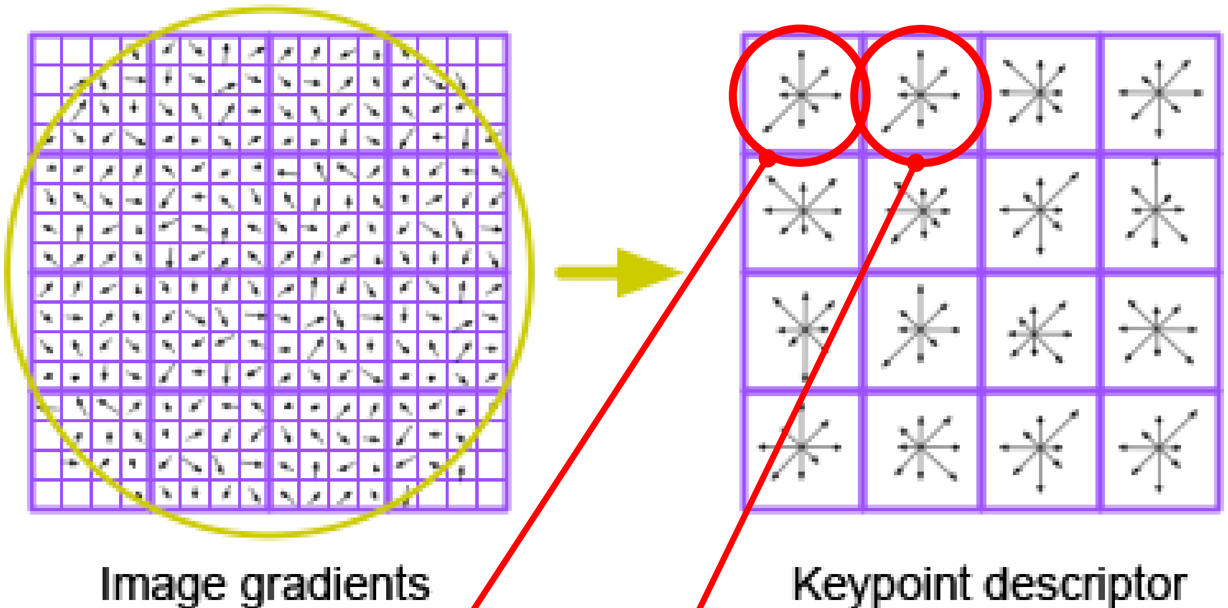


David G. Lowe. **["Distinctive image features from scale-invariant keypoints."]** *IJCV* 60 (2), pp. 91-110, 2004.

Slide: S. Lazebnik

SIFT descriptors





$$D(\xi, \Sigma) = (\underbrace{h1, \ldots, h_8}, \underbrace{h_9, \ldots, h_{16}}, \ldots, h_{128})$$

SIFT descriptor is a 128-bin histogram vector accumulated from 8 quantized gradient orientations in 16 *position-dependent* cells of a region

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

Slide: S. Lazebnik

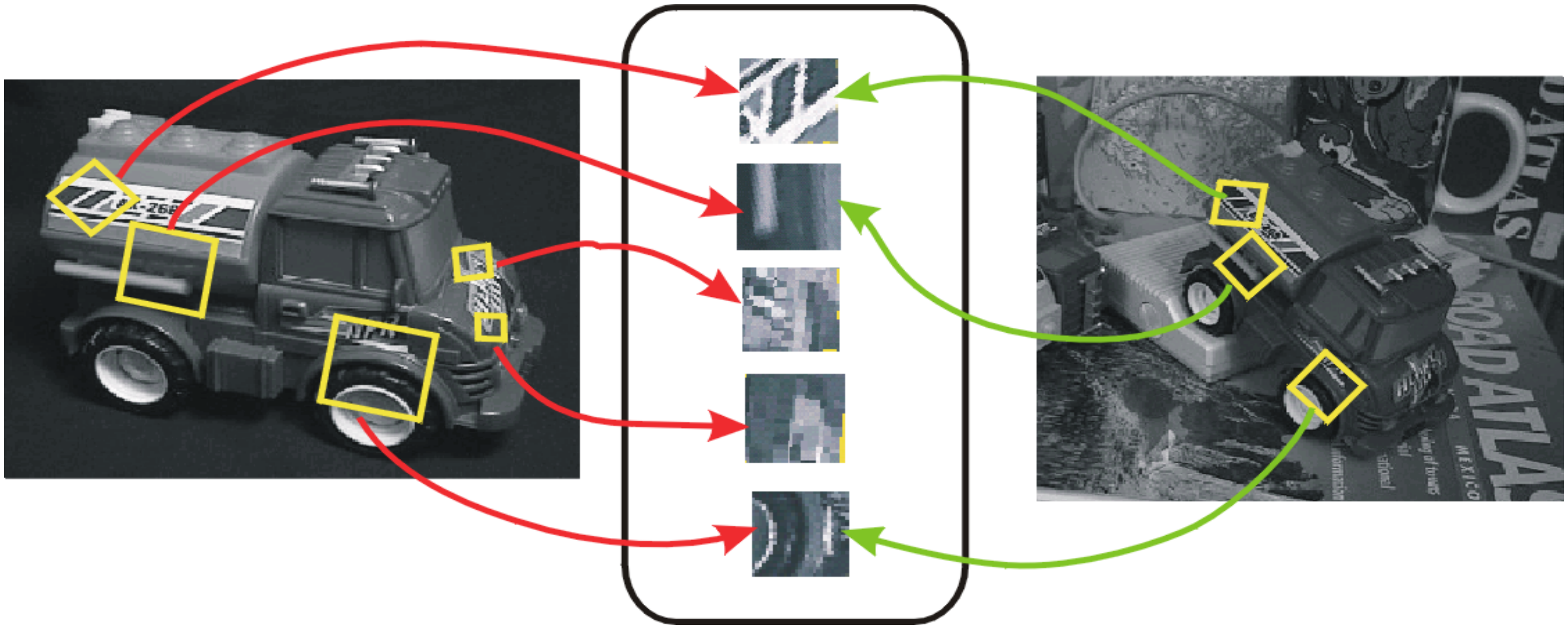
Invariance vs. covariance

Invariance:

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

Covariance:

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



Covariant detection => invariant description

Software

VLFeat: Vision Library Features http://www.vlfeat.org/
(will be used in this course)

- Local image features (Harris,SIFT, MSER, …)
- Local image descriptors (SIFT, LBP, …)
- Feature encodig (VLAD, Fisher)
- Machine learning tools (k-means, GMM, SVM)
- Matlab and C interfaces

References

- Schmid, C. and Mohr, R. (1997). Local grayvalue invariants for image retrieval, IEEE Transactions on Pattern Analysis and Machine Intelligence 19(5): 530–535.
- Lindeberg, T. (1998). Feature detection with automatic scale selection, International Journal of Computer Vision 30(2): 77–116.
- Lowe, D. (2004). Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision.
- Mikolajczyk, K. and Schmid, C. (2002). An affine invariant interest point detector, Proc. Seventh European Conference on Computer Vision, Vol. 2350 of Lecture Notes in Computer Science, Springer Verlag, Berlin, Copenhagen, Denmark, pp. I:128–142.
- Mikolajczyk, K. and Schmid, C. (2003). A performance evaluation of local descriptors, Proc. Computer Vision and Pattern Recognition, pp. II: 257–263.
- Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos, Proc. Ninth International Conference on Computer Vision, Nice, France, pp. 1470–1477.
- VLFeat (Vision Library Features) http://www.vlfeat.org/

Approach

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

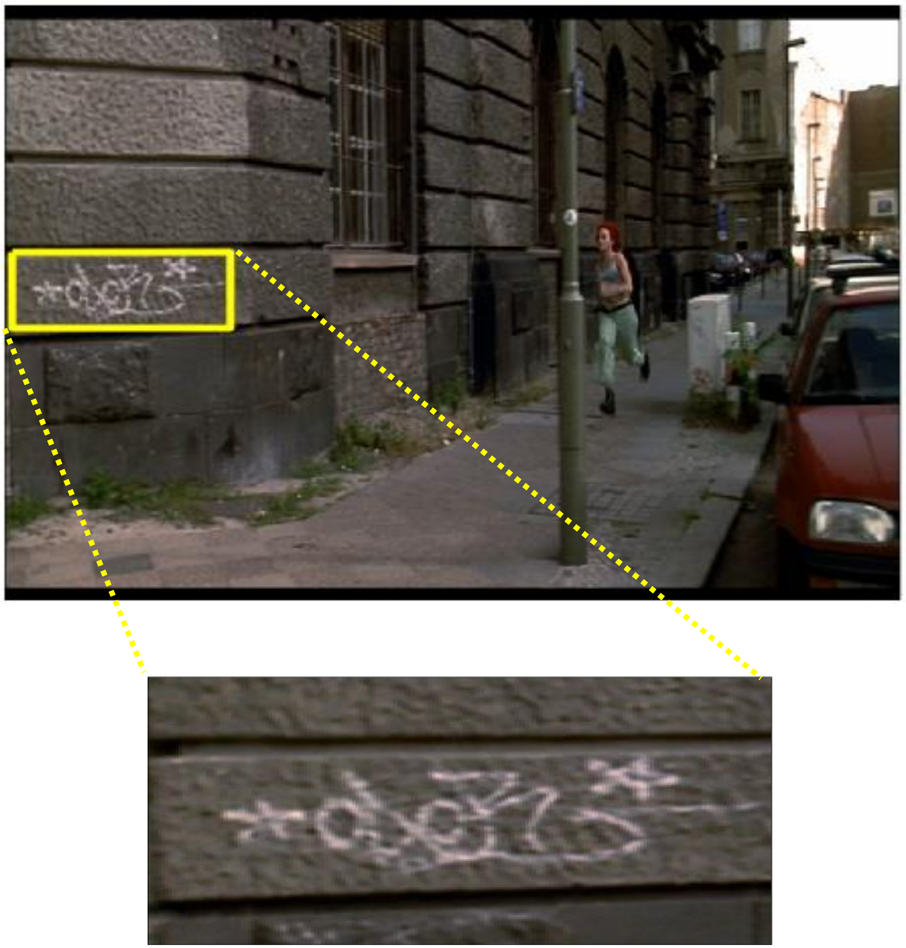
Done ✓

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.

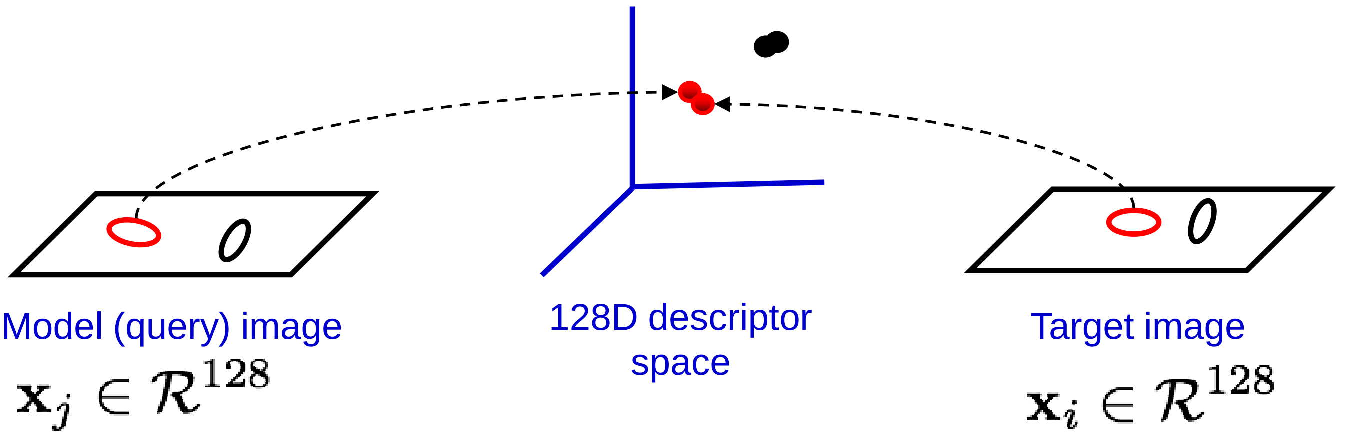
Next

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:

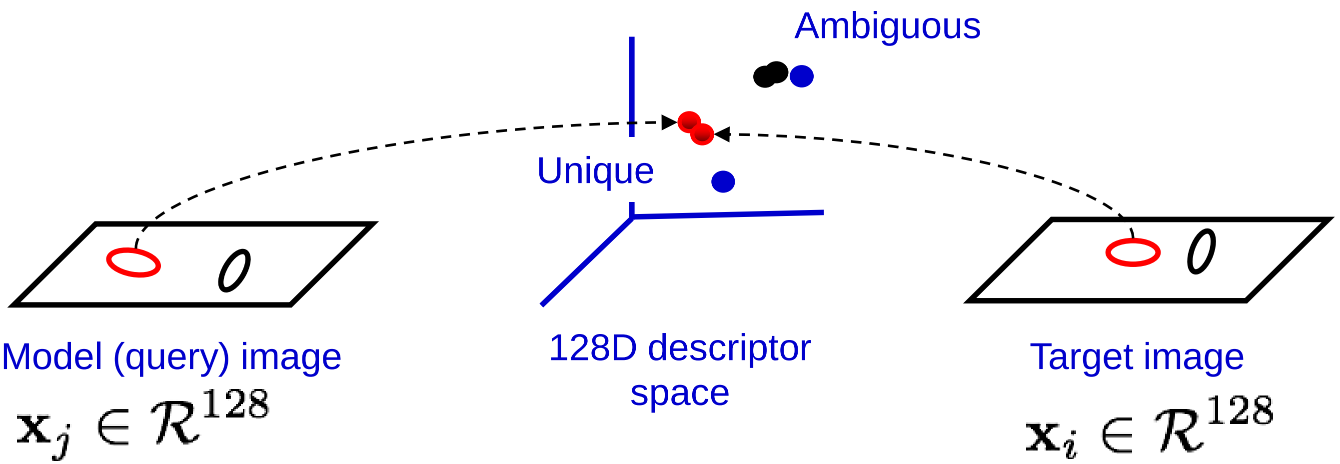
$$\forall j \; NN(j) = \arg\min_i ||\mathbf{x}_i - \mathbf{x}_j||,$$

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, the match is more "unique" or discriminative.

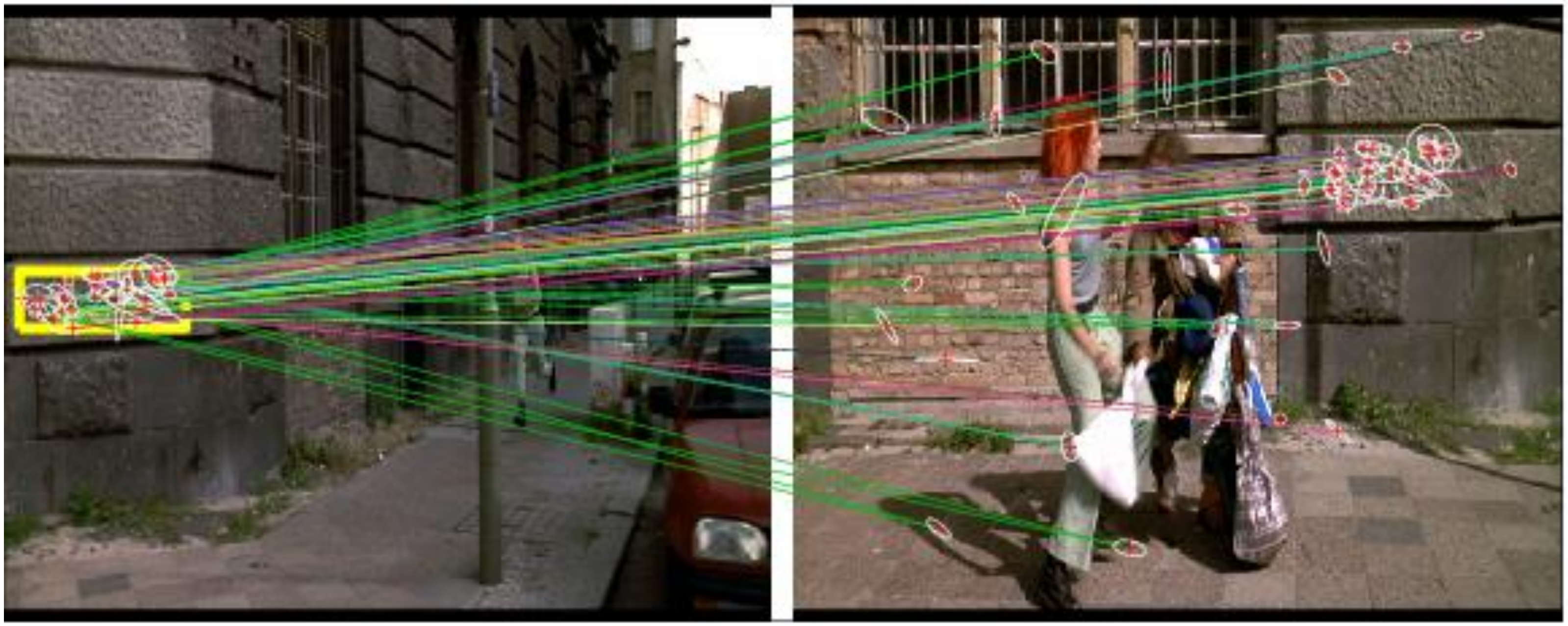
Measure this by the ratio: $r = d_{1NN} / d_{2NN}$

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



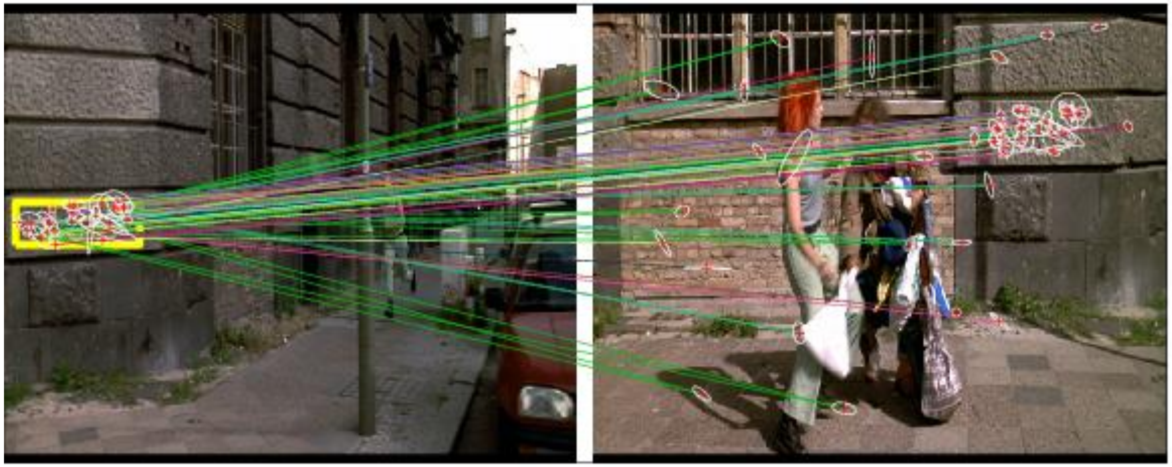
- too much individual invariance
- each region can affine deform independently (by different amounts)
- locally appearance can be ambiguous

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

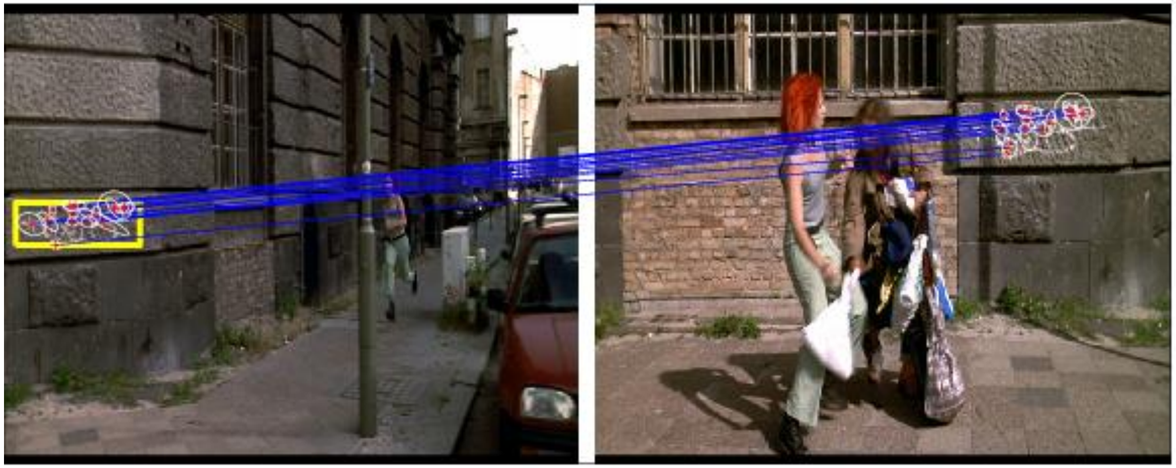
Example I: Two images -“Where is the Graffiti?”

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.

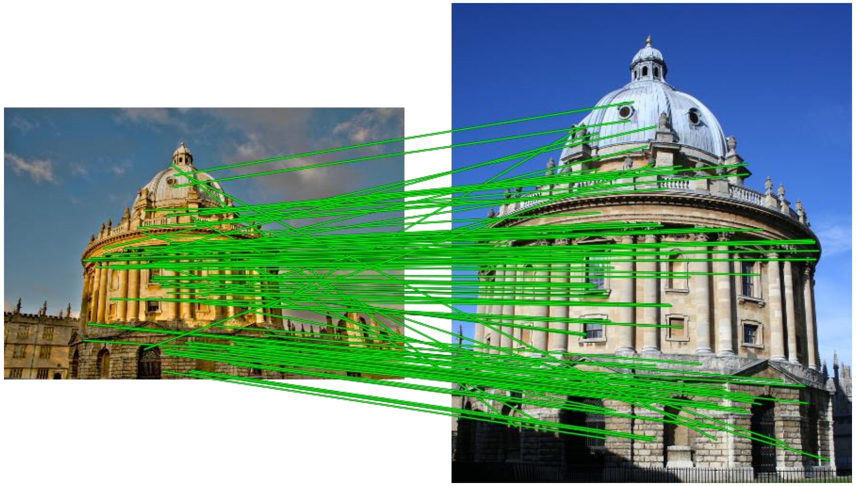
1. **Matching:** Establish tentative (putative) correspondences based on local appearance of individual features (their descriptors).

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

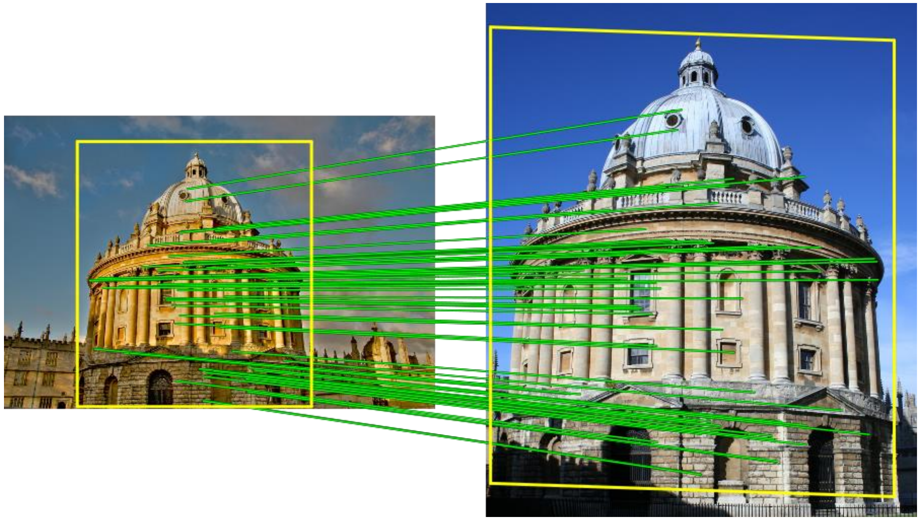
Geometric verification with global constraints

- 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

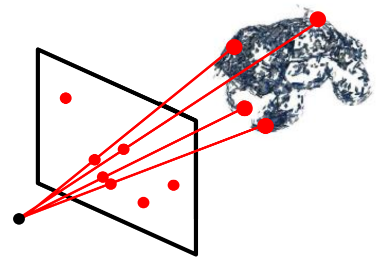


Tentative matches



Matches consistent with an affine transformation

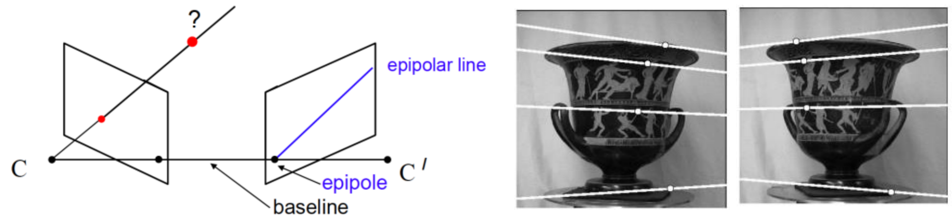
Examples of global constraints



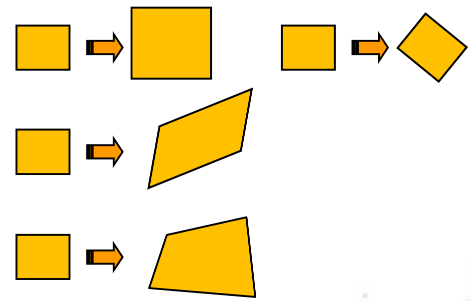
1 view and known 3D model.

- Consistency with a (known) 3D model.

2 views



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

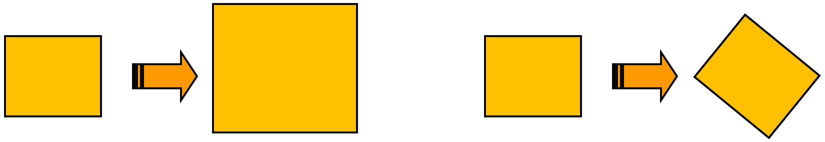


N-views

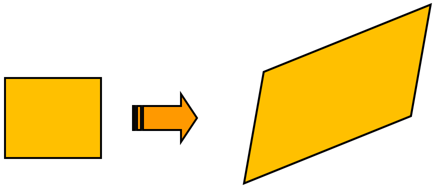
Are images consistent with a 3D model?

2D transformation models

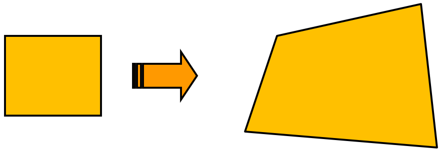
Similarity
(translation,
scale, rotation)



Affine

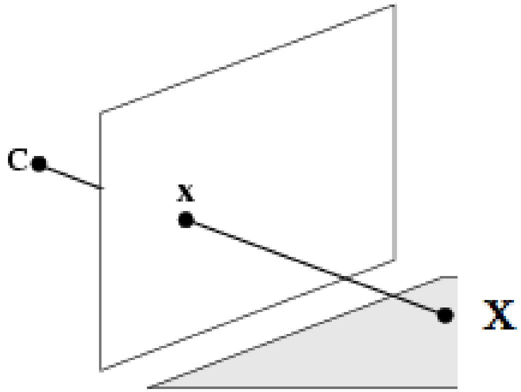


Projective
(homography)



Why are 2D planar transformation important?

Recall perspective projection



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

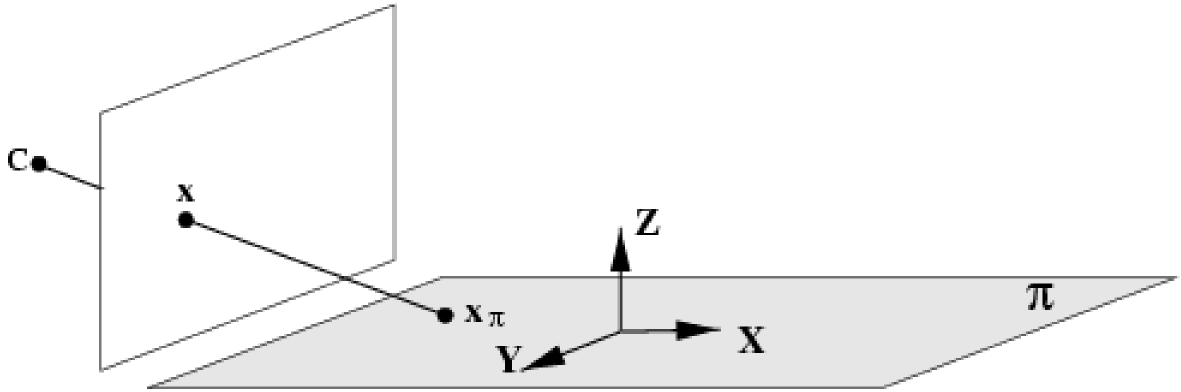
P : 3 × 4 matrix

$\mathbf{X}$: 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 $\mathtt{P}$ reduces to

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

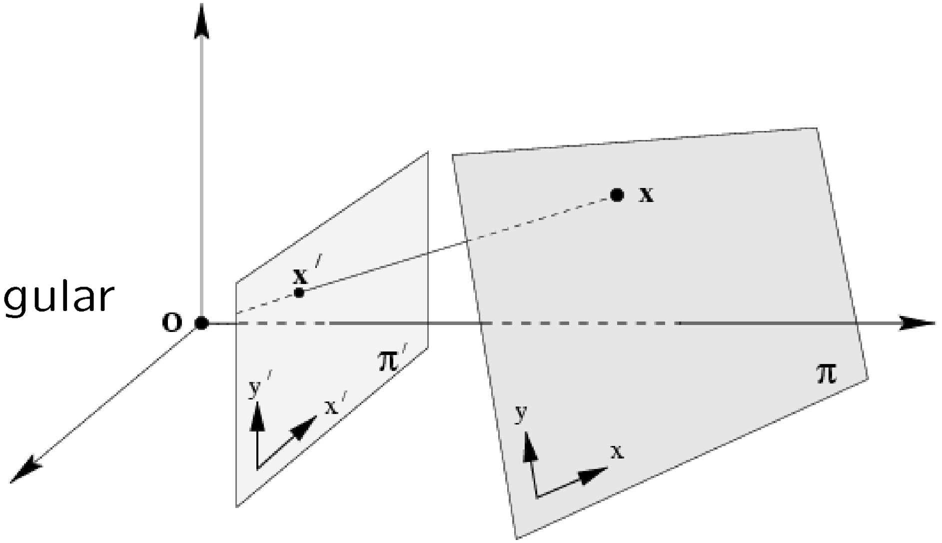
which is a 3×3 matrix representing a general plane to plane projective transformation.

Slide credit: A. Zisserman

Projective transformations continued

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

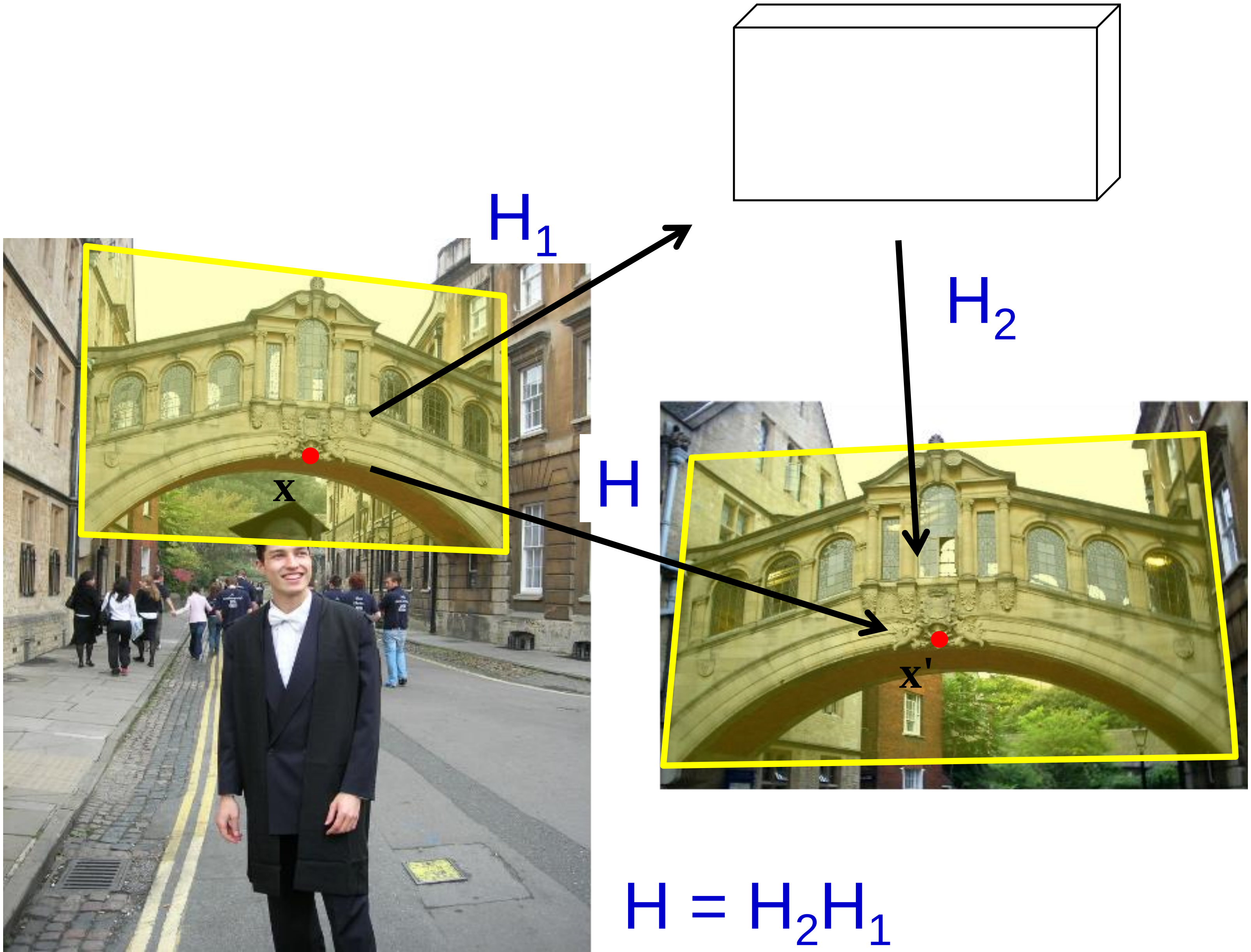
or $\mathbf{x}' = \mathtt{H}\mathbf{x}$, where $\mathtt{H}$ is a 3×3 non-singular homogeneous matrix.



- 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

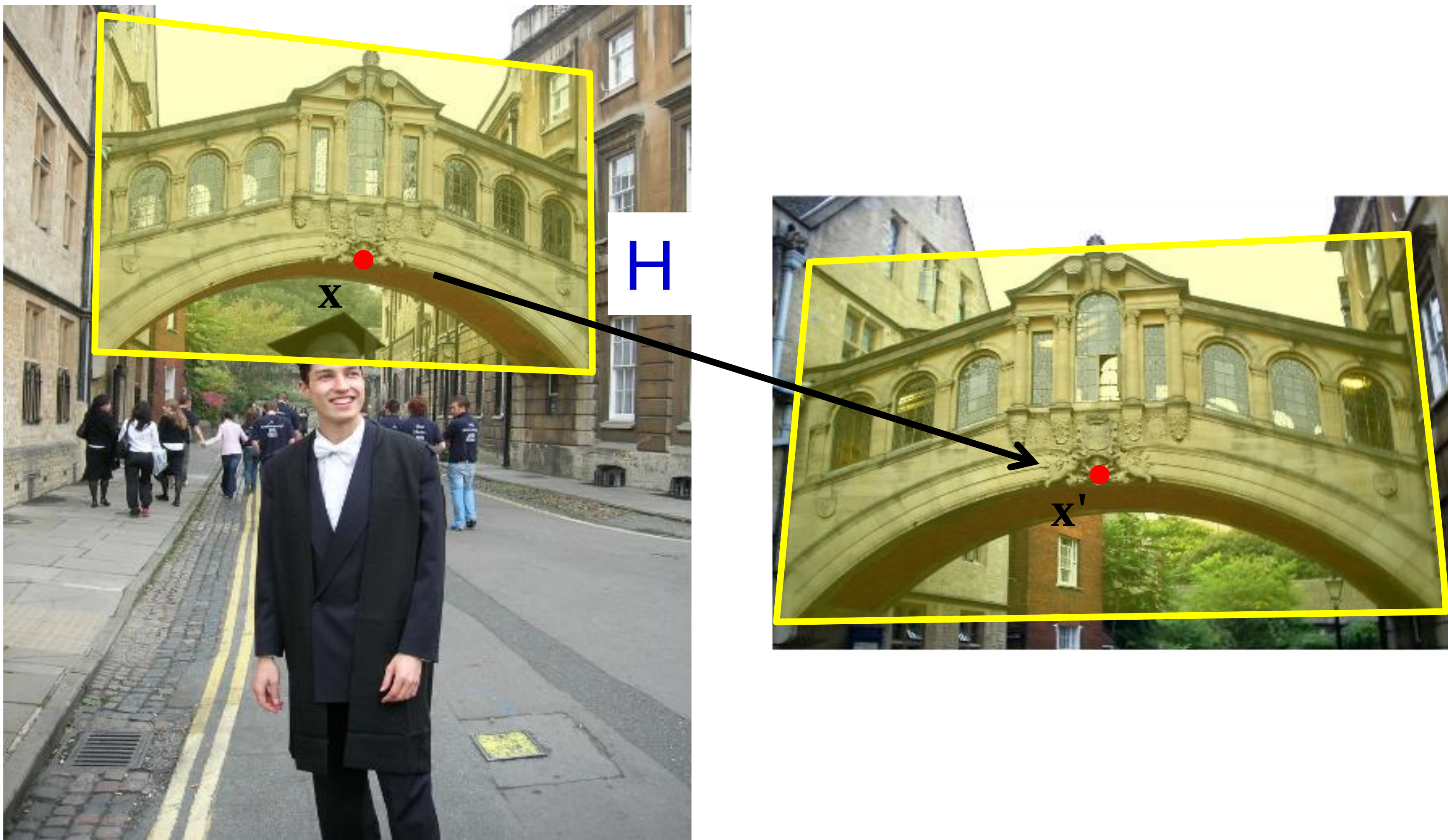
Planes in the scene induce *homographies*



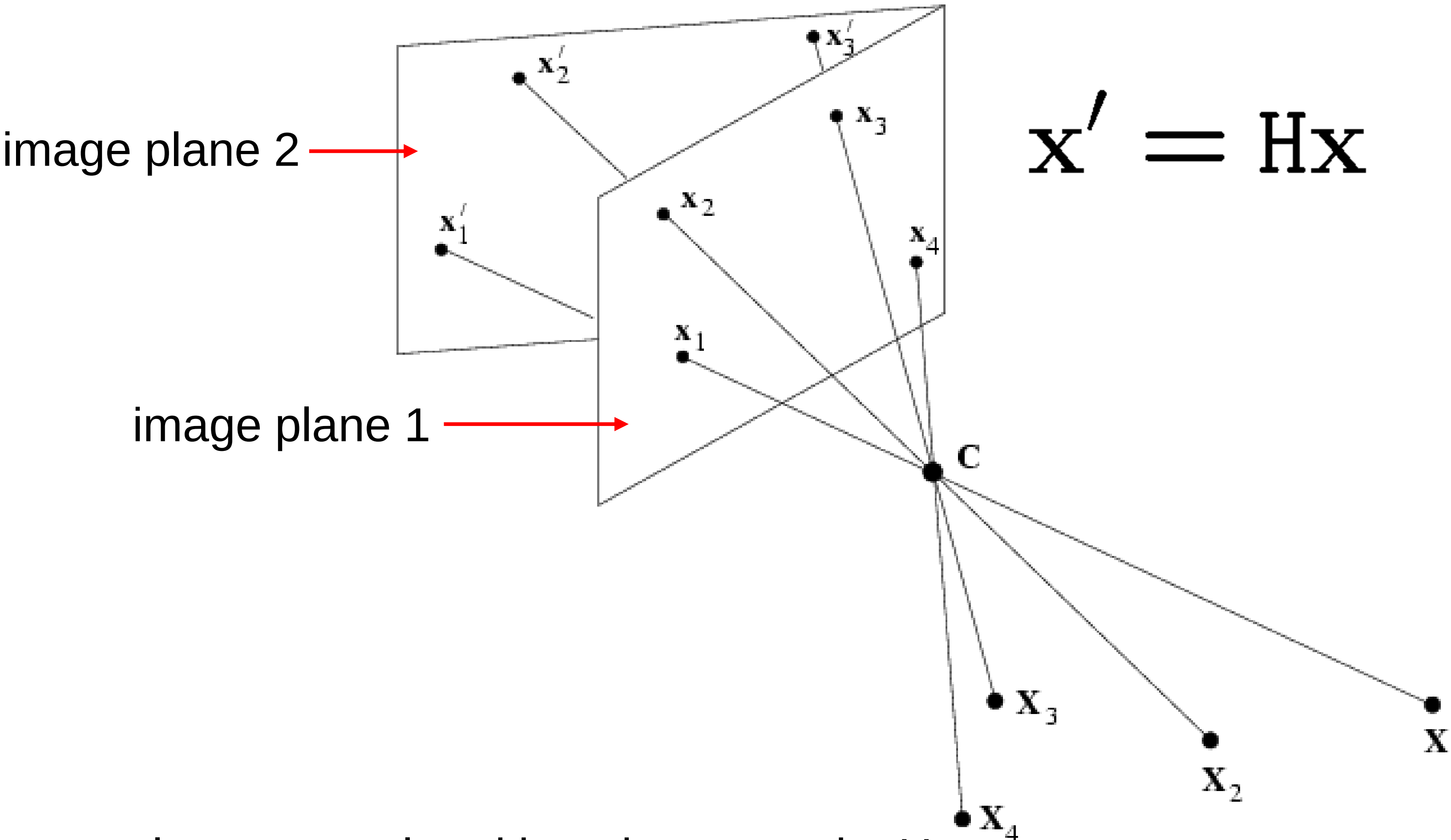
$$H = H_2 H_1$$

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.



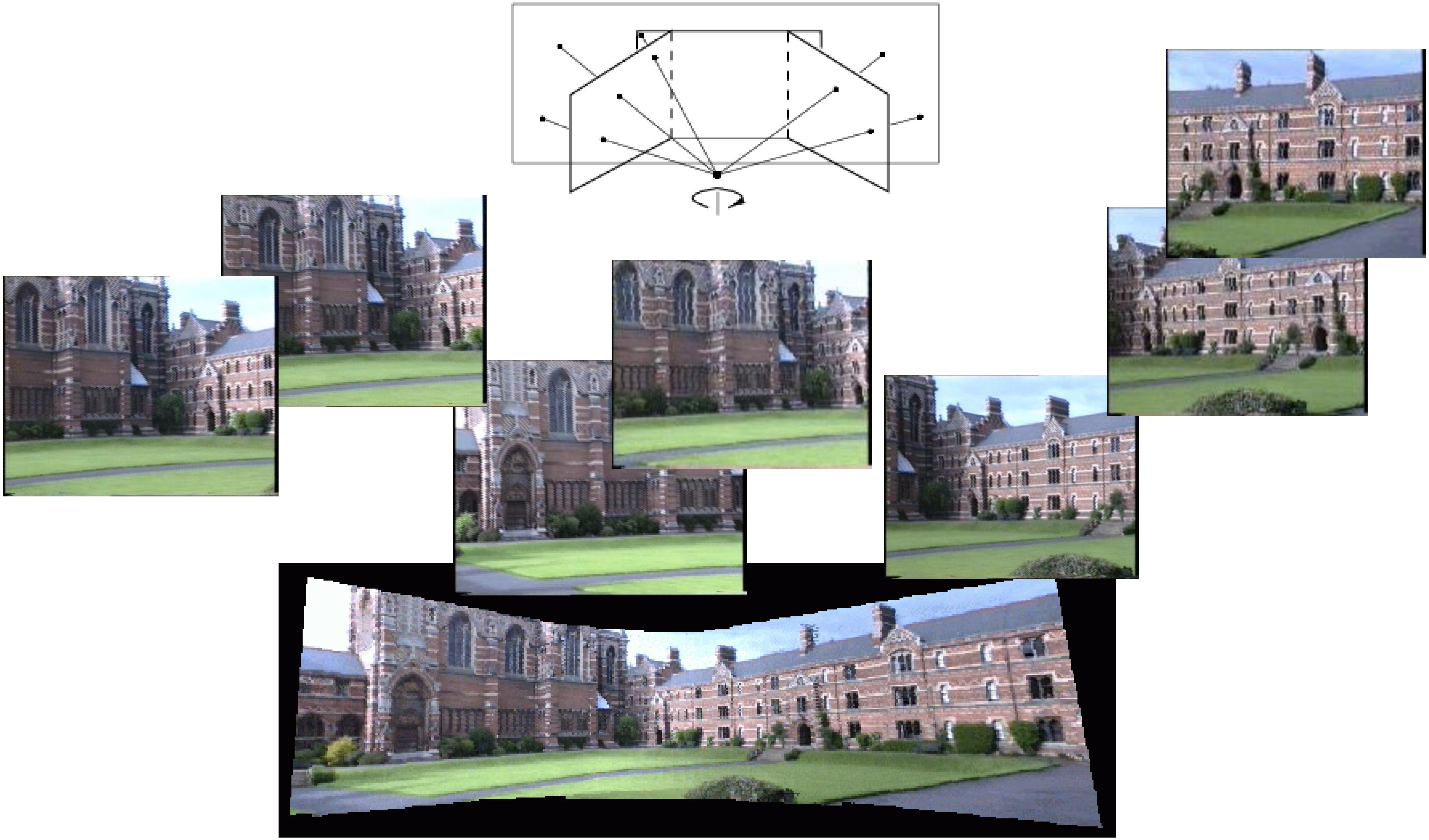
Case II: Cameras rotating about their centre



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

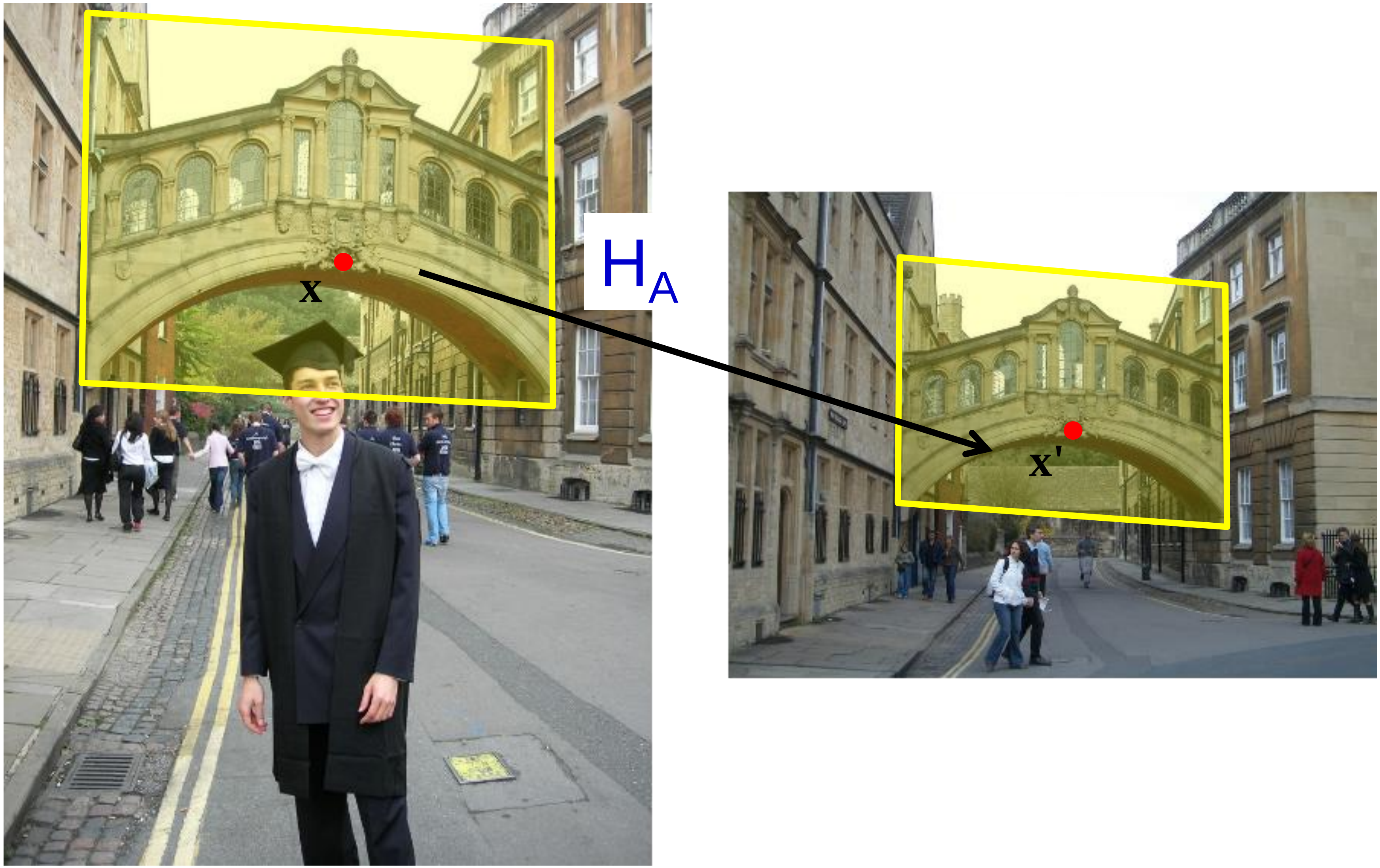
- The two image planes are related by a homography H
- H depends only on the relation between the image planes and camera centre, C, not on the 3D structure

Case II: Example of a rotating camera



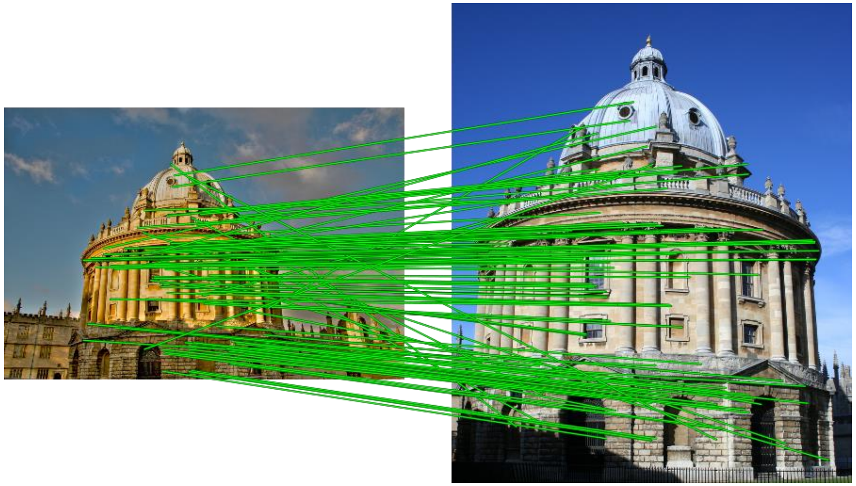
Images courtesy of A. Zisserman.

Homography is often approximated well by 2D affine geometric transformation

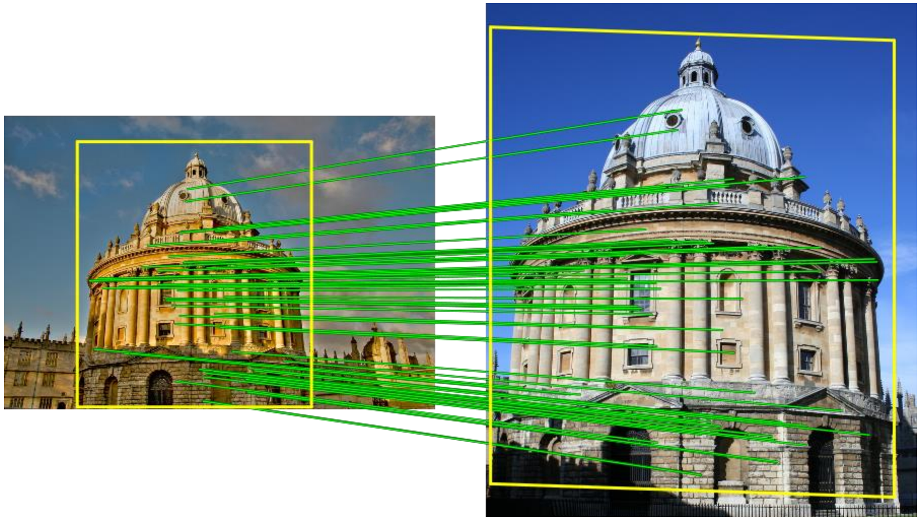


Homography is often approximated well by 2D affine geometric transformation – Example II.

Two images with similar camera viewpoint



Tentative matches



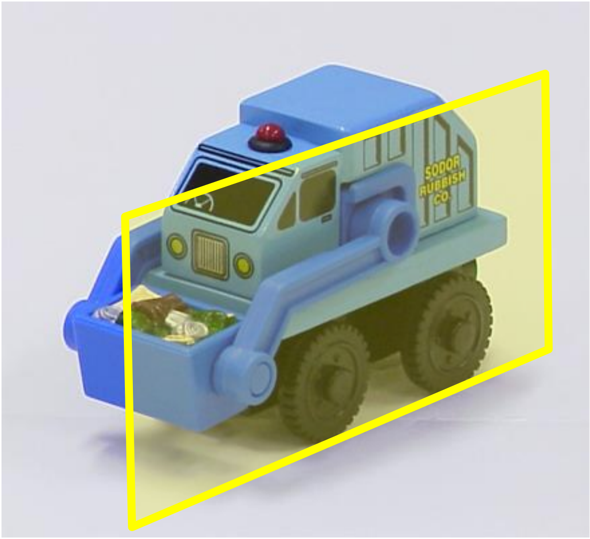
Matches consistent with an affine transformation

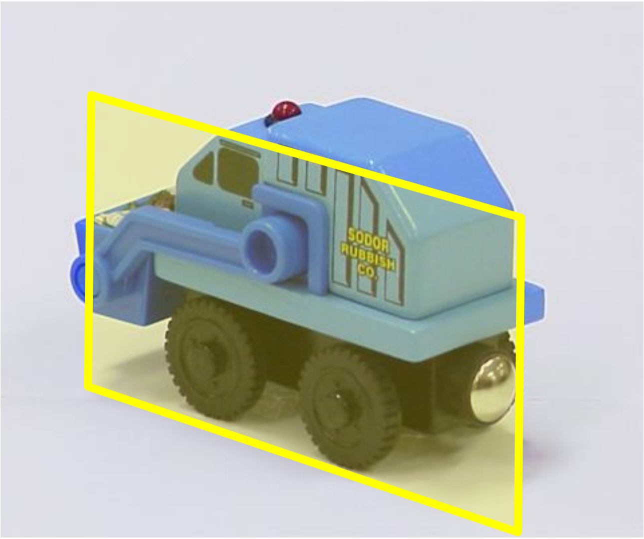
Example: estimating 2D affine transformation

- 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

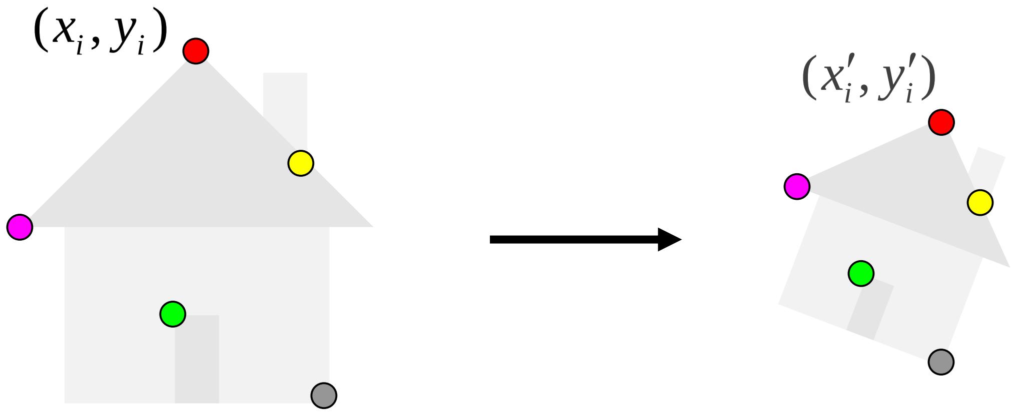
- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for **roughly planar objects** and **roughly orthographic cameras**
- Can be used to initialize fitting for more complex models





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

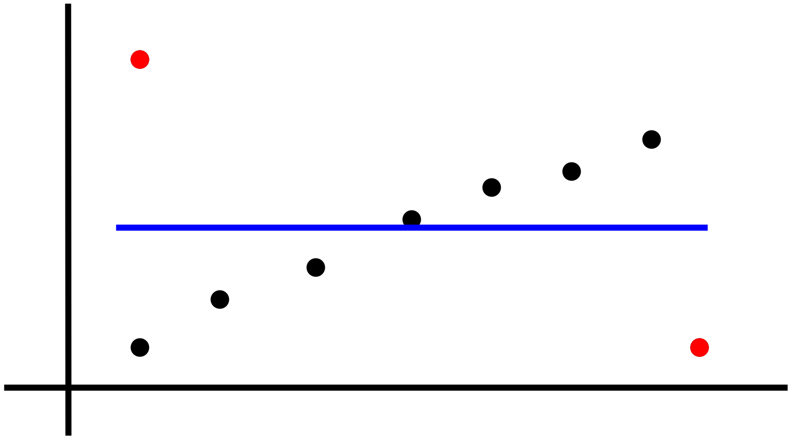
How do we fit a geometric transformation to a small subset of all possible matches?

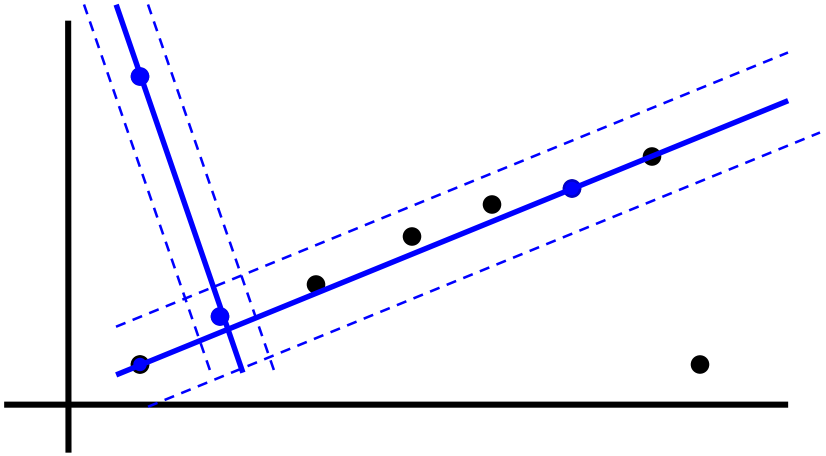
Possible strategies:

- 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

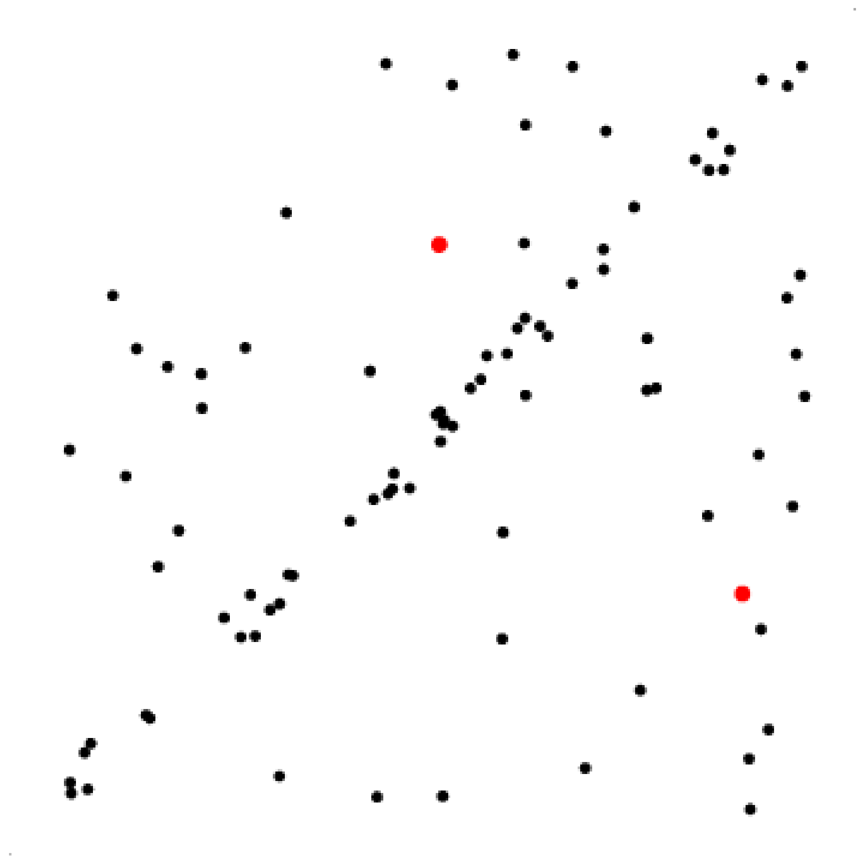
1. Select random sample of 2 points
2. Compute the line through these points
3. Measure support (number of points within threshold distance of the line)

Choose the line with the largest number of inliers

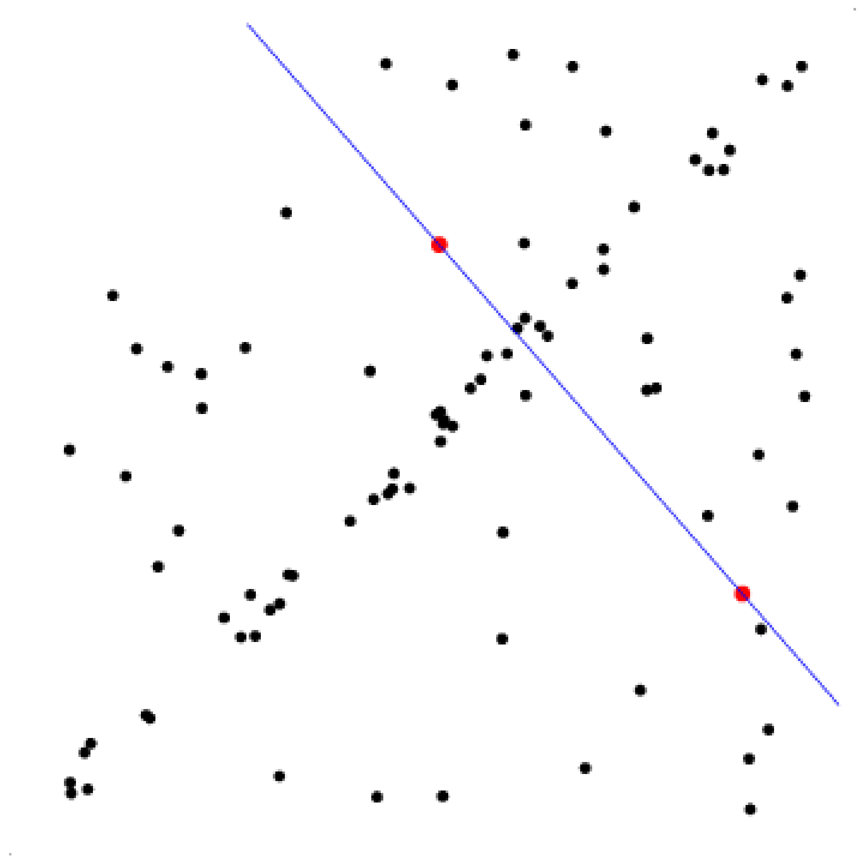
- Compute least squares fit of line to inliers (regression)

Slide credit: A. Zisserman

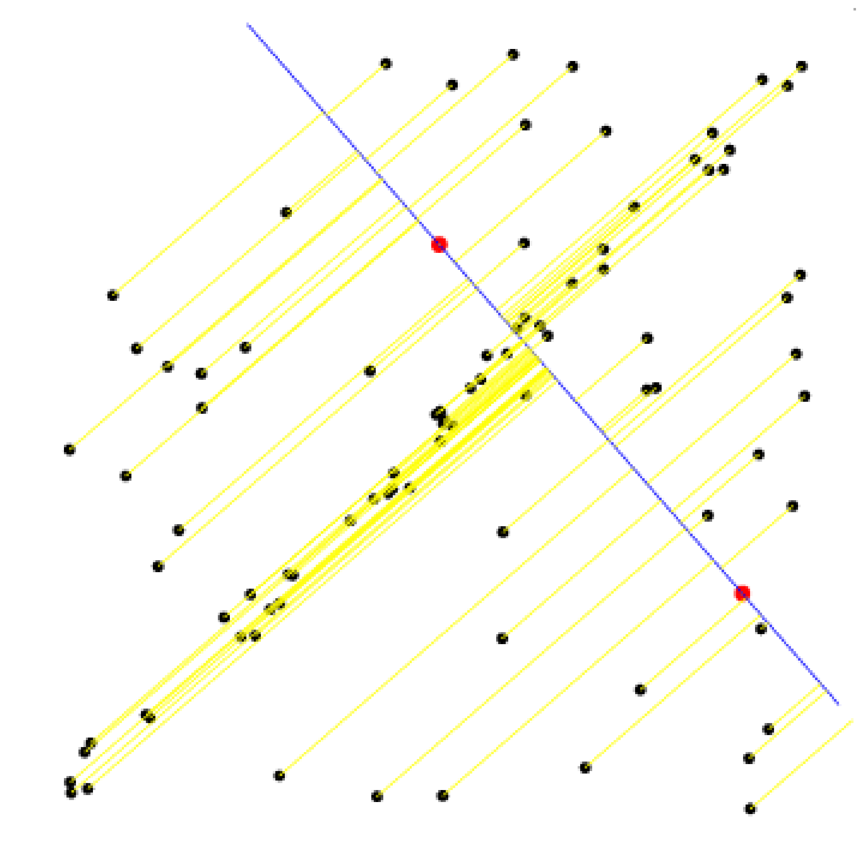
Slide credit: O. Chum



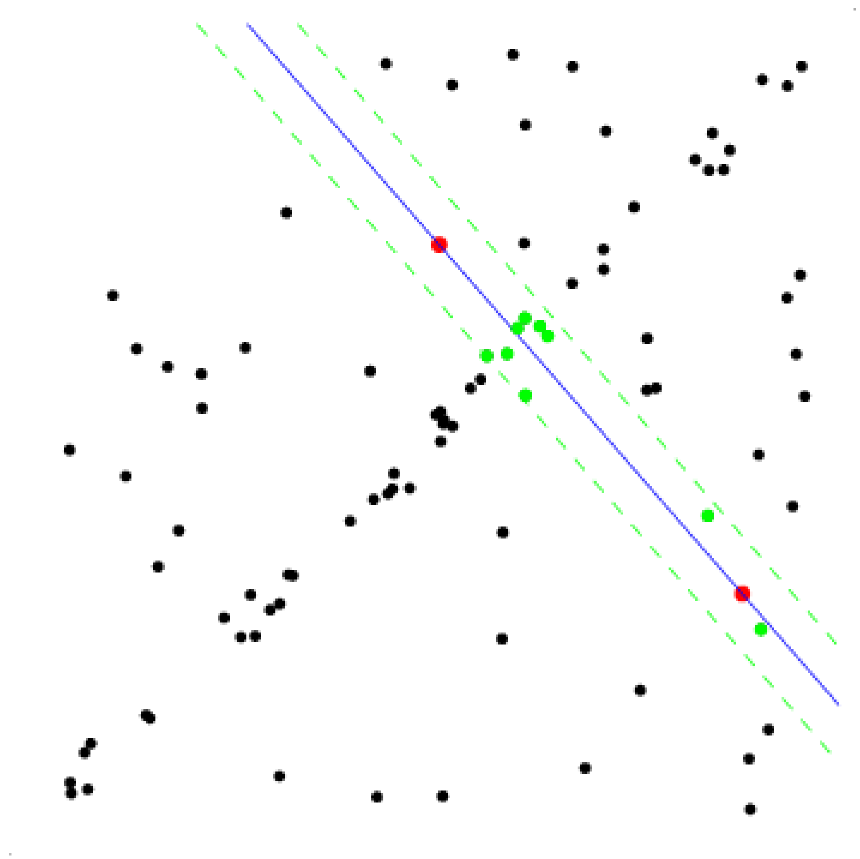
Slide credit: O. Chum



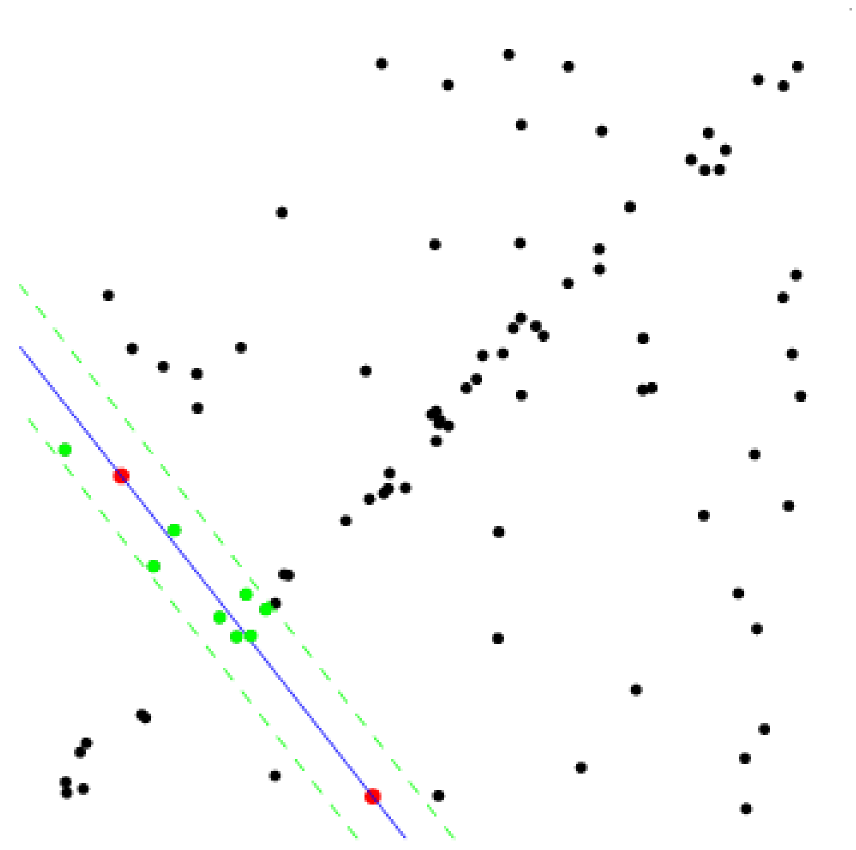
Slide credit: O. Chum



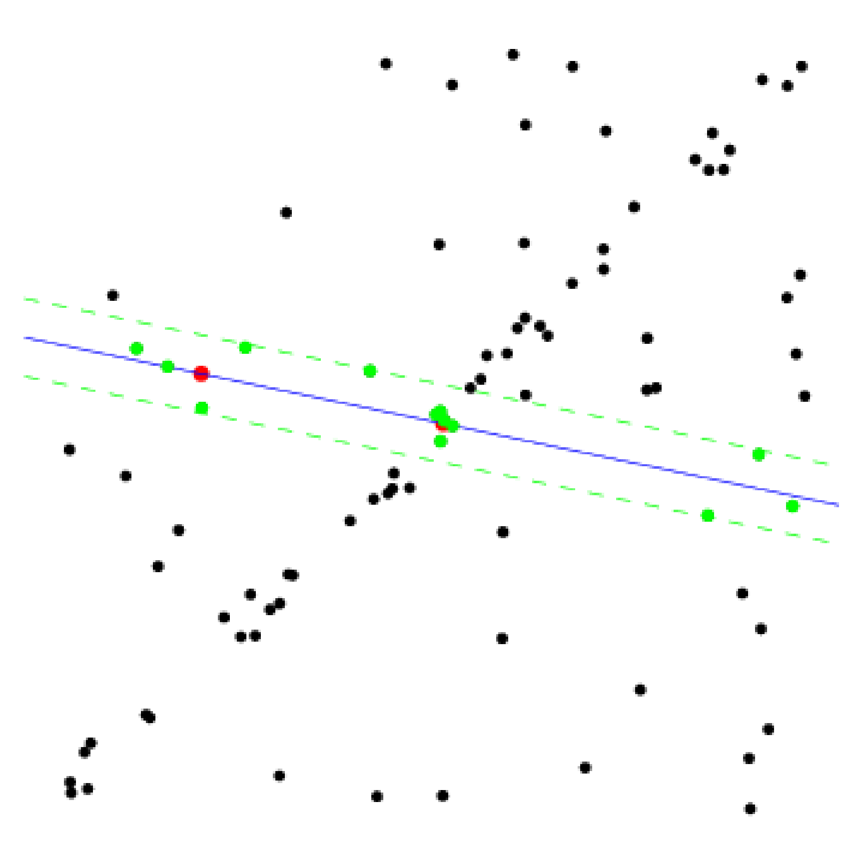
Slide credit: O. Chum



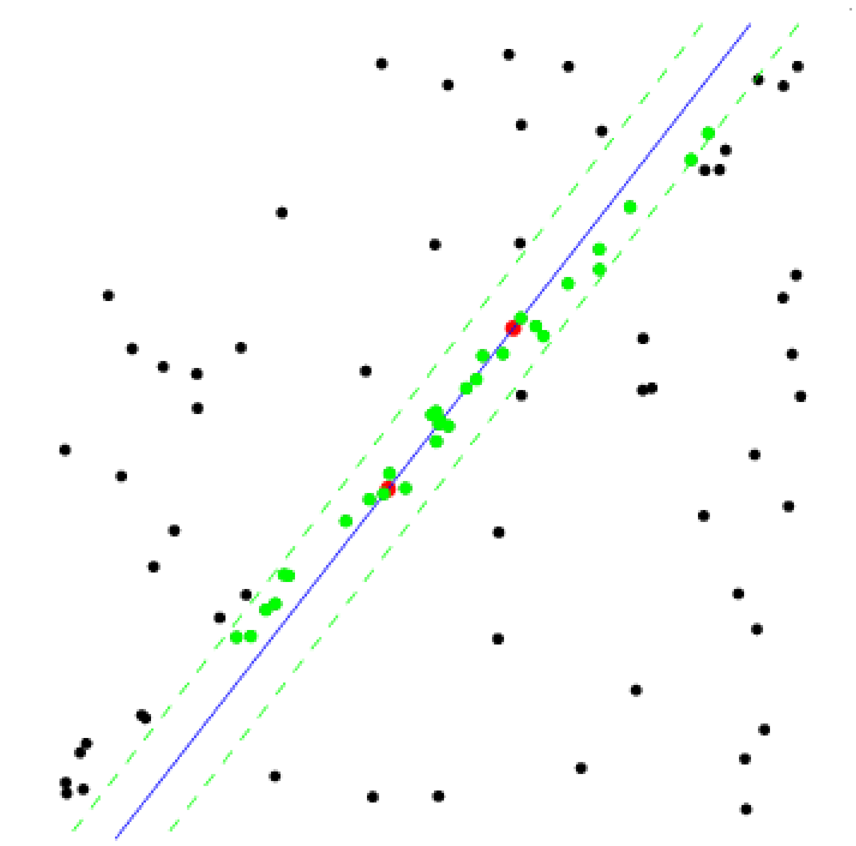
Slide credit: O. Chum



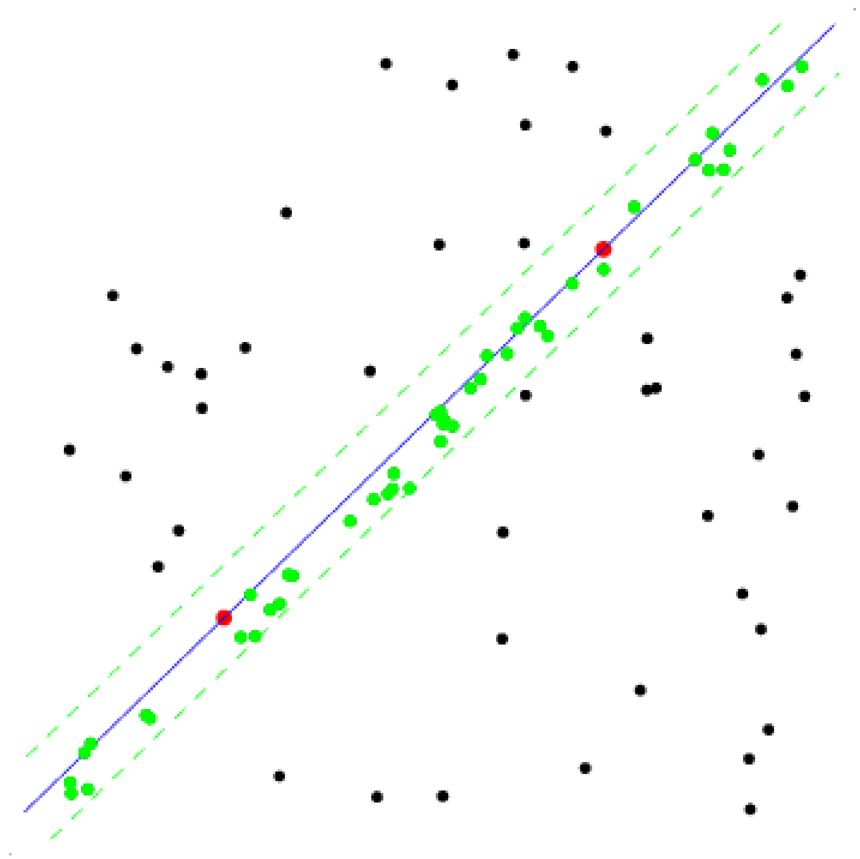
Slide credit: O. Chum



Slide credit: O. Chum



Slide credit: O. Chum



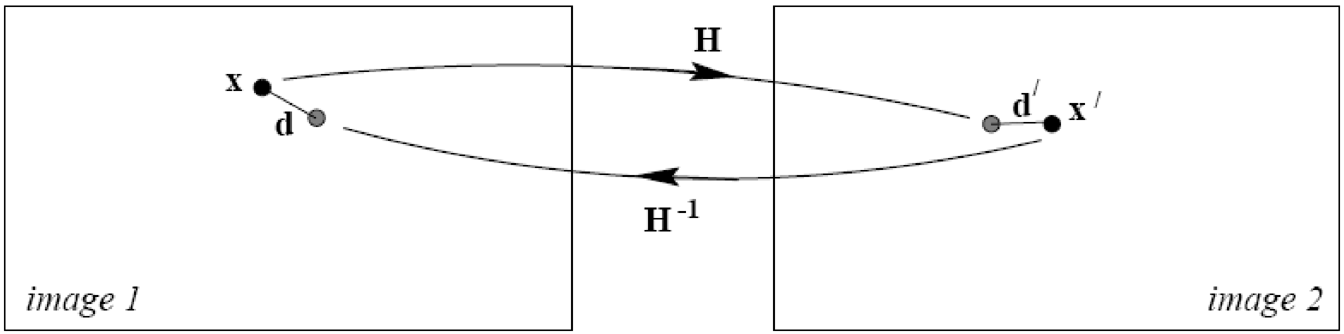
Slide credit: O. Chum

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. $d^2_{transfer} < t$)

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

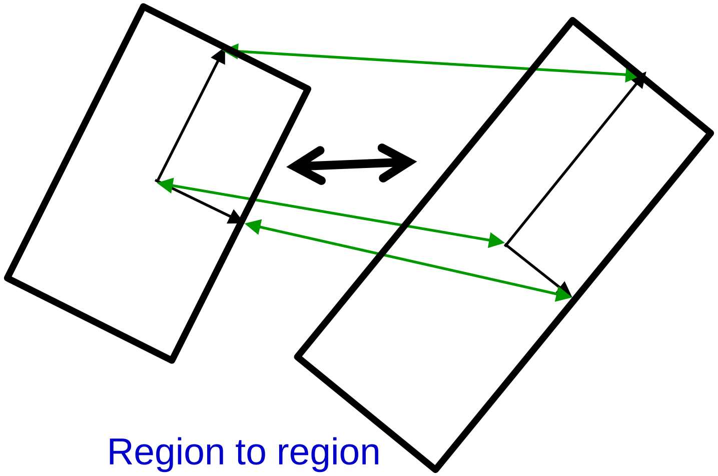


Choose the (H,t) with the largest number of inliers

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

How many samples 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).



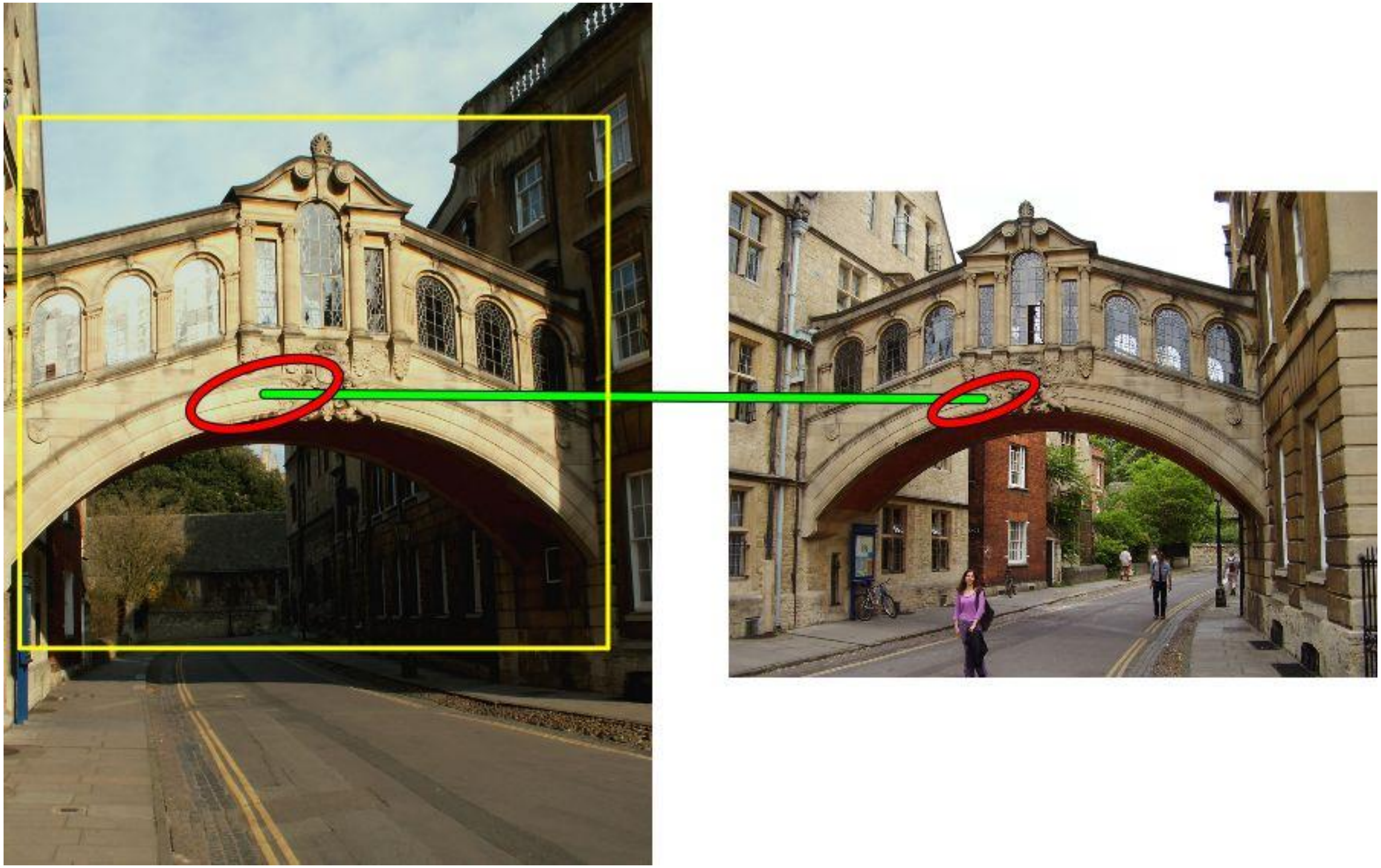
Region to region correspondence

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



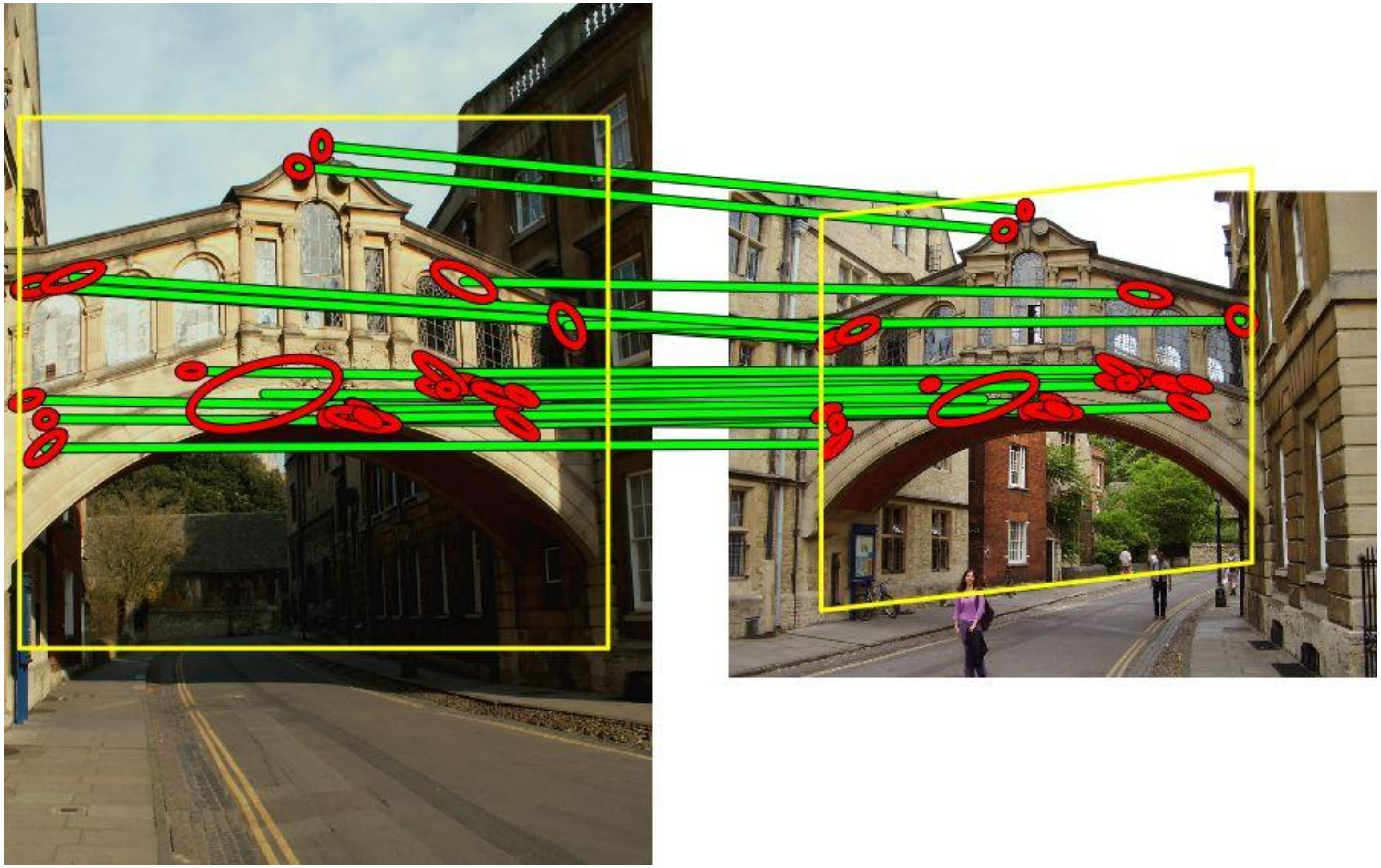
Example: restricted affine transform

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)

Summary

Finding correspondences in images is useful for

- Image matching, panorama stitching
- Object recognition
- Large scale image search: next part of the lecture

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]

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

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

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

References: RANSAC

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).

References: Geometric verification for visual search

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.

Approach

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

Done ✓

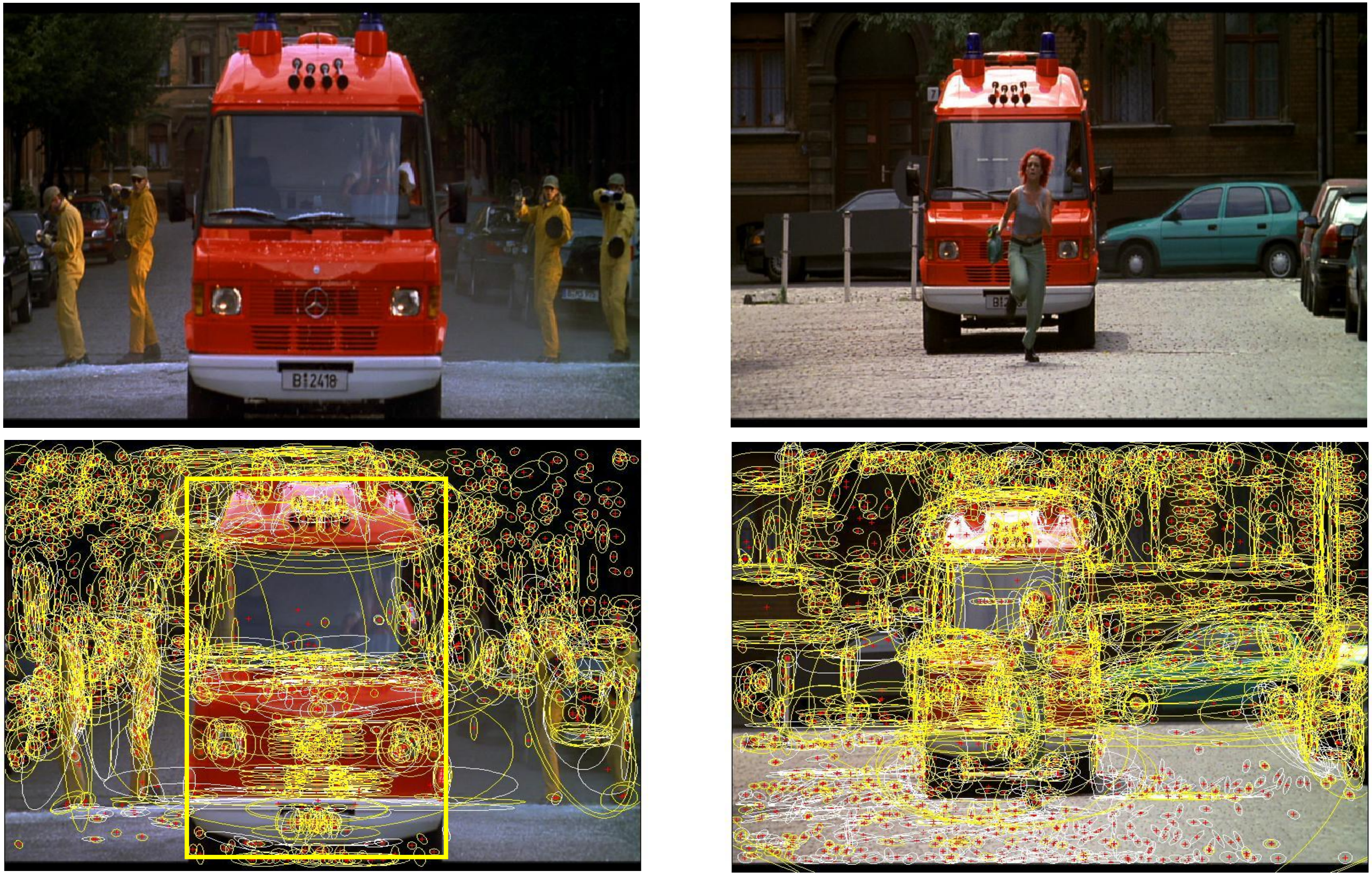
1. **Matching:** Establish tentative (putative) correspondences based on local appearance of individual features (their descriptors).

Next

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

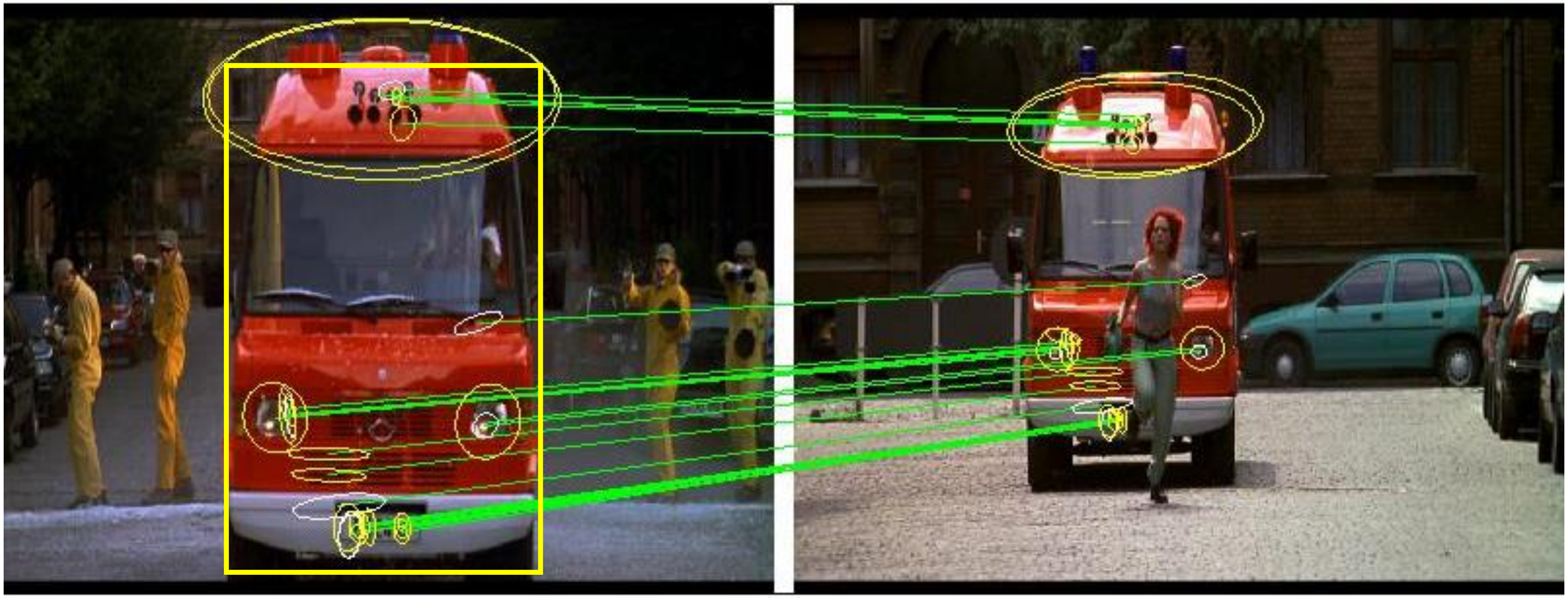
Done ✓

Example II: Two images again



1000+ descriptors per image

Match regions between frames using SIFT descriptors and spatial consistency



Multiple regions overcome problem of partial occlusion

Approach - review

1. Establish tentative (or putative) correspondence based on local appearance of individual features (now)

2. Verify matches based on semi-local / global geometric relations (You have just seen this).

What about multiple images?

- So far, we have seen successful matching of a query image to a single target image using local features.

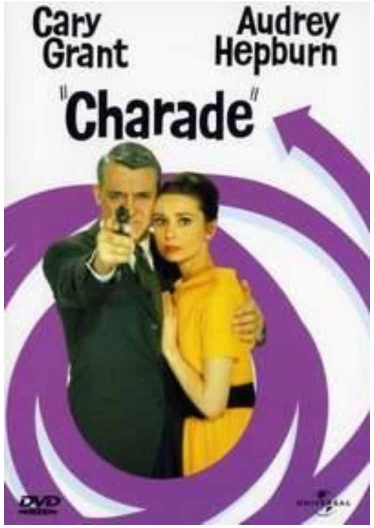
- How to generalize this strategy to multiple target images with reasonable complexity?

  - $10, 10^2, 10^3, \ldots, 10^7, \ldots 10^{10}, \ldots$ images?

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

History of “large scale” visual search with local regions

Schmid and Mohr ’97	– 1k images
Sivic and Zisserman’03	– 5k images
Nister and Stewenius’06	– 50k images (1M)
Philbin et al.’07	– 100k images
Chum et al.’07 + Jegou et al.’07	– 1M images
Chum et al.’08	– 5M images
Jegou et al. ’09	– 10M images
Jegou et al. ’10	– ~100M images

…

All on a single machine in ~ 1 second!

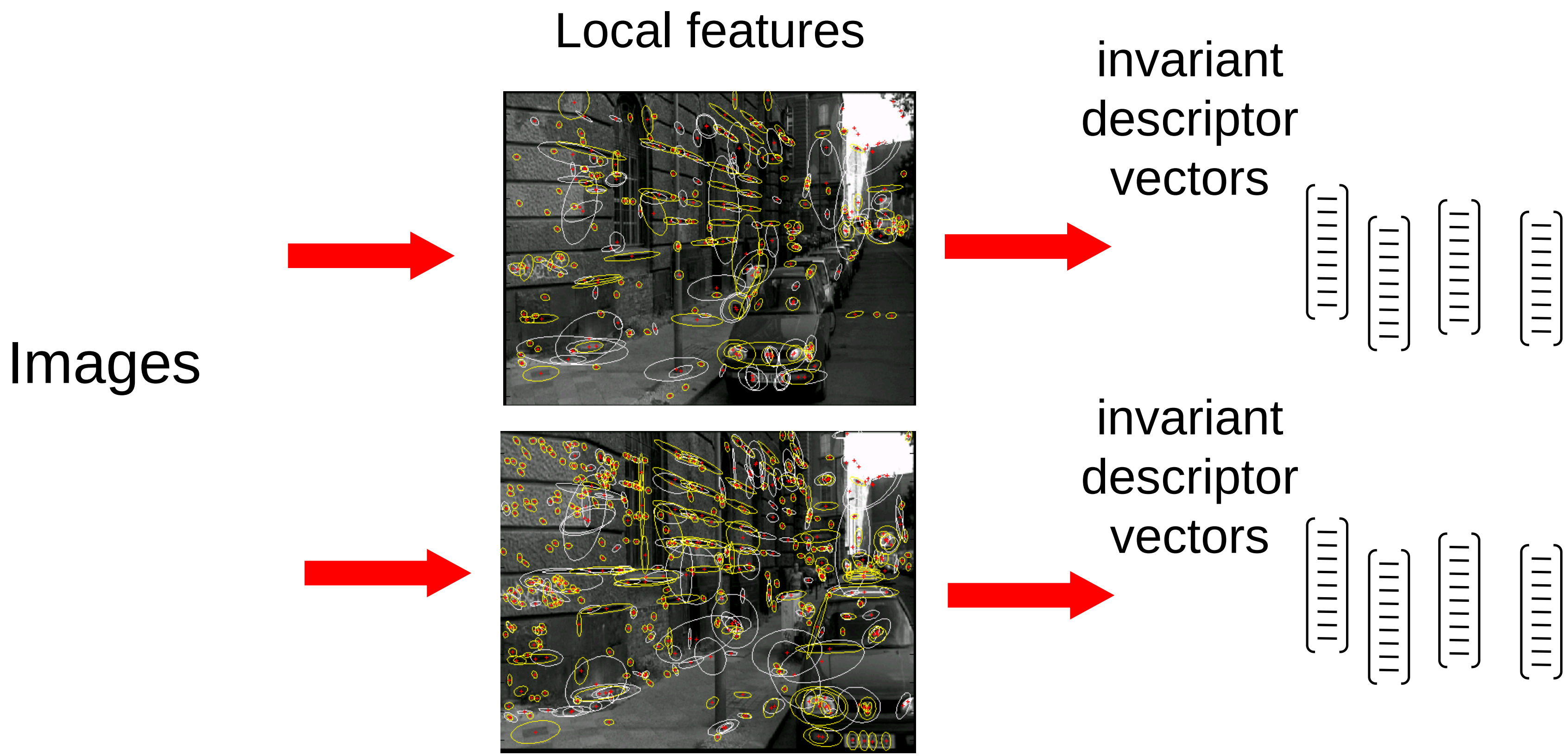
Two strategies

1. Efficient approximate nearest neighbour search on local feature descriptors.

2. Quantize descriptors into a “visual vocabulary” and use efficient techniques from text retrieval.

   (Bag-of-words representation)

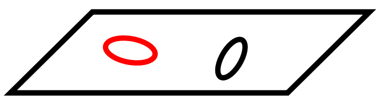
Strategy I: Efficient approximate NN search



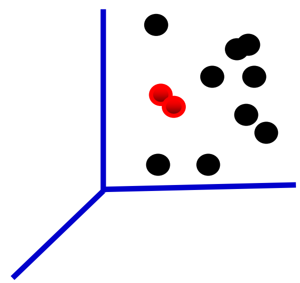
1. Compute local features in each image independently
2. “Label” each feature by a descriptor vector based on its intensity
3. Finding corresponding features is transformed to finding nearest neighbour vectors
4. Rank matched images by number of (tentatively) corresponding regions
5. Verify top ranked images based on spatial consistency

Finding nearest neighbour vectors

Establish correspondences between object model image and images in the database by **nearest neighbour matching** on SIFT vectors



Model image



128D descriptor space

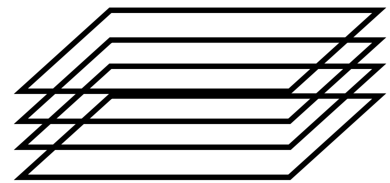


Image database

Solve following 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 from all the database images.

Quick look at the complexity of the NN-search

N … images

M … regions per image (~1000)

D … dimension of the descriptor (~128)

Exhaustive linear search: O(M NMD)

Example:

- Matching two images (N=1), each having 1000 SIFT descriptors
  Nearest neighbors search: 0.4 s (2 GHz CPU, implemenation in C)
- Memory footprint: 1000 * 128 = 128kB / image

# of images	CPU time	Memory req.
N = 1,000 …	~7min	(~100MB)
N = 10,000 …	~1h7min	(~ 1GB)
…		
N = 10^7	~115 days	(~ 1TB)
…		
All images on Facebook:		
N = 10^{10} …	~300 years	(~ 1PB)

Nearest-neighbor matching

Solve following problem for all feature vectors, $\mathbf{x_j}$, in the query image:

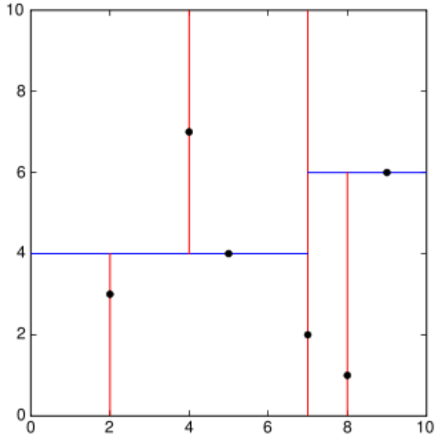
$$\forall j \; NN(j) = \arg\min_i ||\mathbf{x}_i - \mathbf{x}_j||$$

where $\mathbf{x_i}$ are features in database images.

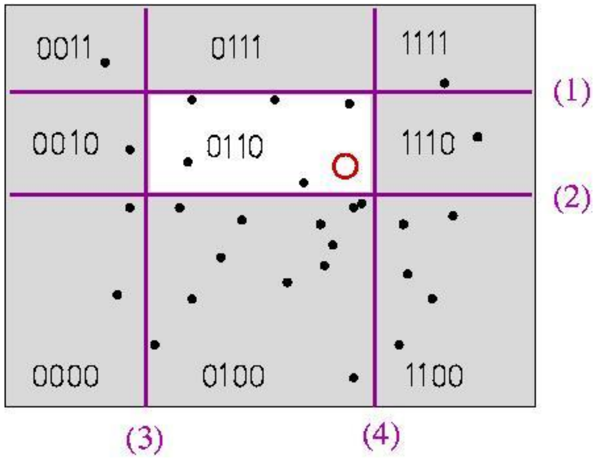
Nearest-neighbour matching is the major computational bottleneck

- Linear search performs *dn* operations for *n* features in the database and *d* dimensions
- No exact methods are faster than linear search for d>10
- Approximate methods can be much faster, but at the cost of missing some correct matches. Failure rate gets worse for large datasets.

Indexing local features: approximate nearest neighbor search



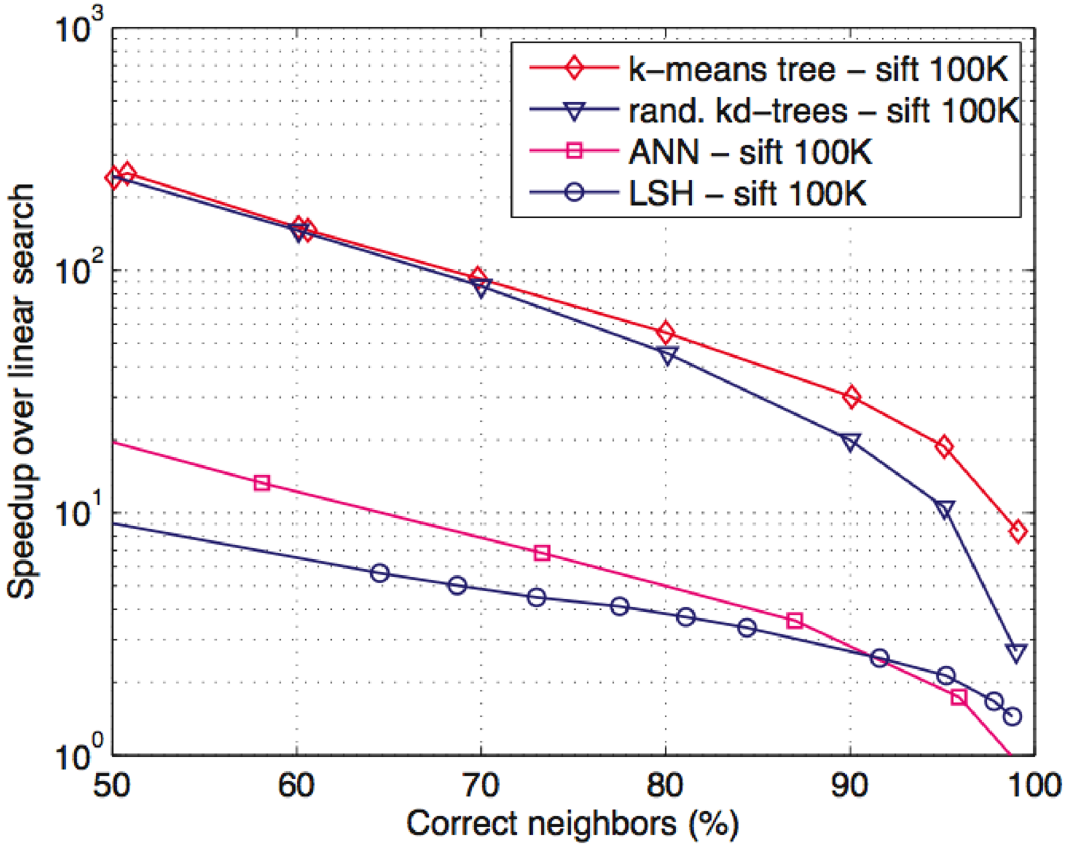
Best-Bin First (BBF), a variant of k-d trees that uses priority queue to examine most promising branches first [Beis & Lowe, CVPR 1997]



Locality-Sensitive Hashing (LSH), a randomized hashing technique using hash functions that map similar points to the same bin, with high probability [Indyk & Motwani, 1998]

Comparison of approximate NN-search methods

Dataset: 100K SIFT descriptors



Code for all methods available online, see Muja&Lowe'09

Figure: Muja&Lowe'09

Approximate nearest neighbor search (references)

J. L. Bentley. Multidimensional binary search trees used for associative searching. Comm. ACM, 18(9), 1975.

Freidman, J. H., Bentley, J. L., and Finkel, R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw., 3:209–226, 1977.*

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., and Wu, A. Y. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM, 45:891–923, 1998.*

C. Silpa-Anan and R. Hartley. Optimised KD-trees for fast image descriptor matching. In CVPR, 2008.

M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In VISAPP, 2009.

P. Indyk and R. Motwani, “Approximate nearest neighbors: towards removing the curse of dimensionality,” in *Proc. of 30th ACM Symposium on Theory of Computing, 1998*

G. Shakhnarovich, P. Viola, and T. Darrell, “Fast pose estimation with parameter-sensitive hashing,” in *Proc. of the IEEE International Conference on Computer Vision, 2003.*

R. Salakhutdinov and G. Hinton, “Semantic Hashing,” ACM SIGIR, 2007.

Y. Weiss, A. Torralba, and R. Fergus, “Spectral hashing,” in *NIPS, 2008.*

ANN - search (references continued)

O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. BMVC., 2008.

M. Raginsky and S. Lazebnik, “Locality-Sensitive Binary Codes from Shift-Invariant Kernels,” in *Proc. of Advances in neural information processing systems, 2009.*

B. Kulis and K. Grauman, “Kernelized locality-sensitive hashing for scalable image search,” Proc. of the IEEE International Conference on Computer Vision, 2009.

J. Wang, S. Kumar, and S.-F. Chang, “Semi-supervised hashing for scalable image retrieval,” in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

J. Wang, S. Kumar, and S.-F. Chang, “Sequential projection learning for hashing with compact codes,” in Proceedings of the 27th International Conference on Machine Learning, 2010.

So far …

- Linear exhaustive search can be prohibitively expensive for large image collections

- Answer (so far): approximate NN search methods
  - Randomized KD-trees
  - Locality sensitive hashing

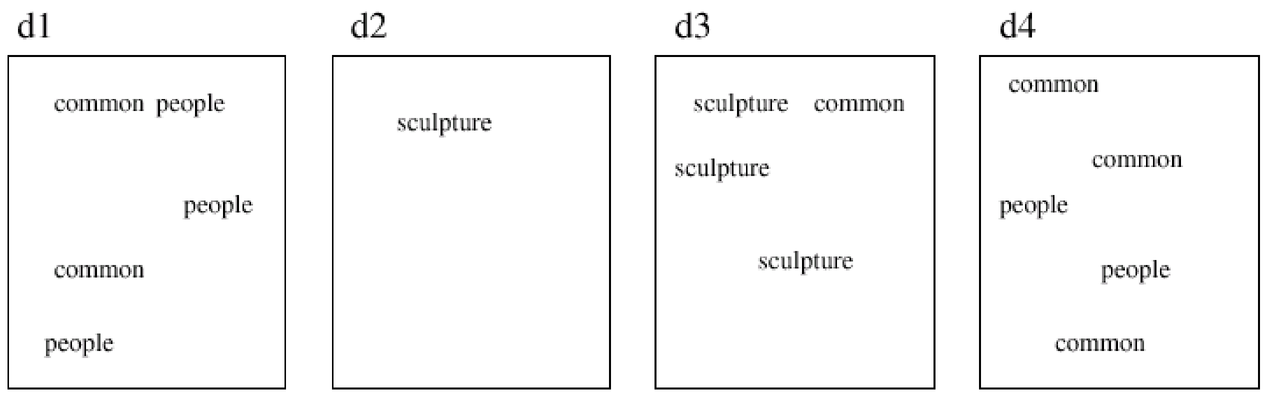
- However, memory footprint can be still high.

  Example: N = 10^7 images, 10^{10} SIFT features with 128B per feature $\Longrightarrow$ 1TB of memory

Look how text-based search engines (Google) index documents – **inverted files**.

Indexing text with inverted files

Document collection:

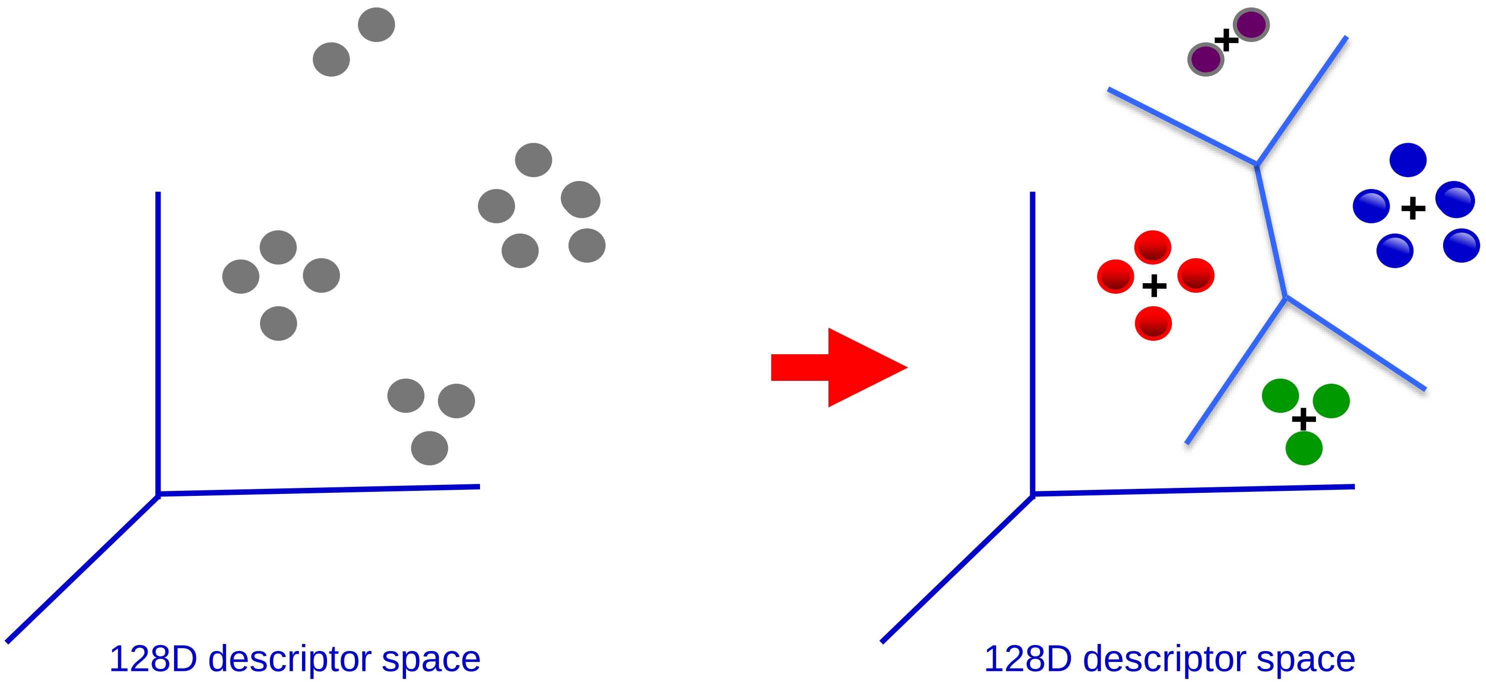


Inverted file:

Term	**List of hits** (occurrences in documents)
People	[d1:hit hit hit], [d4:hit hit] …
Common	[d1:hit hit], [d3: hit], [d4: hit hit hit] …
Sculpture	[d2:hit], [d3: hit hit hit] …

Need to map feature descriptors to “visual words”.

Build a visual vocabulary



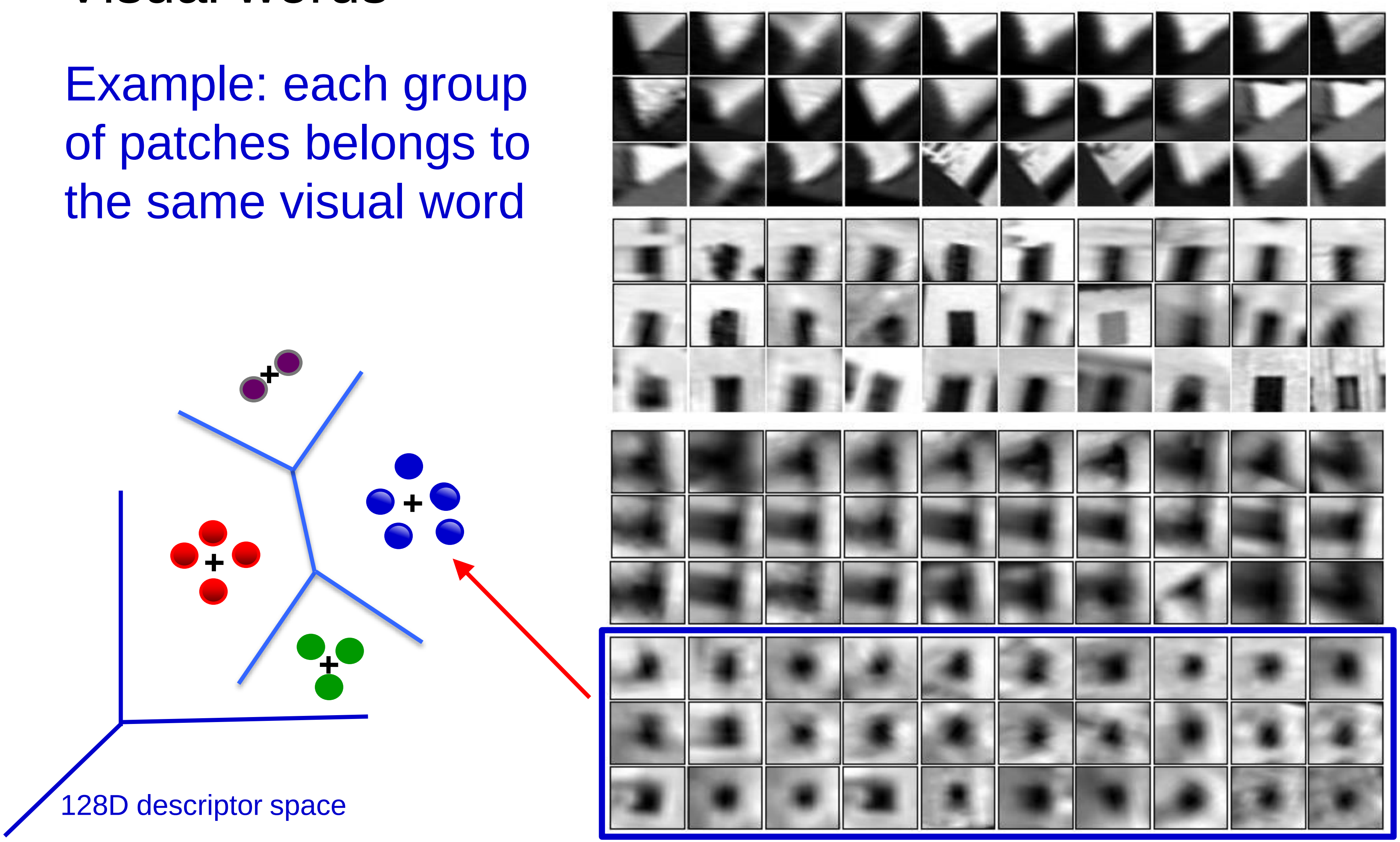
Vector quantize descriptors

- Compute SIFT features from a subset of images
- K-means clustering (need to choose K)

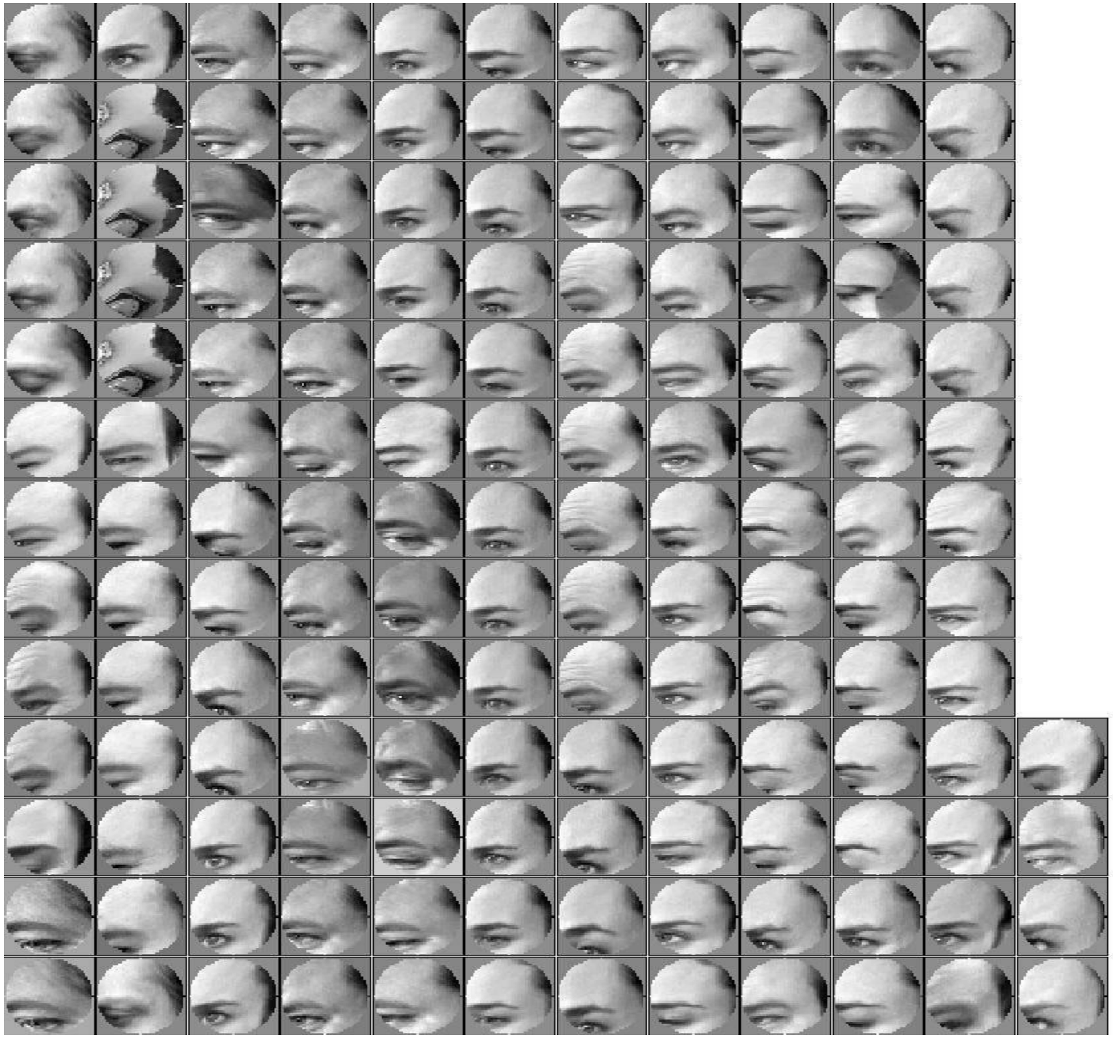
[Sivic and Zisserman, ICCV 2003]

Visual words

Example: each group of patches belongs to the same visual word

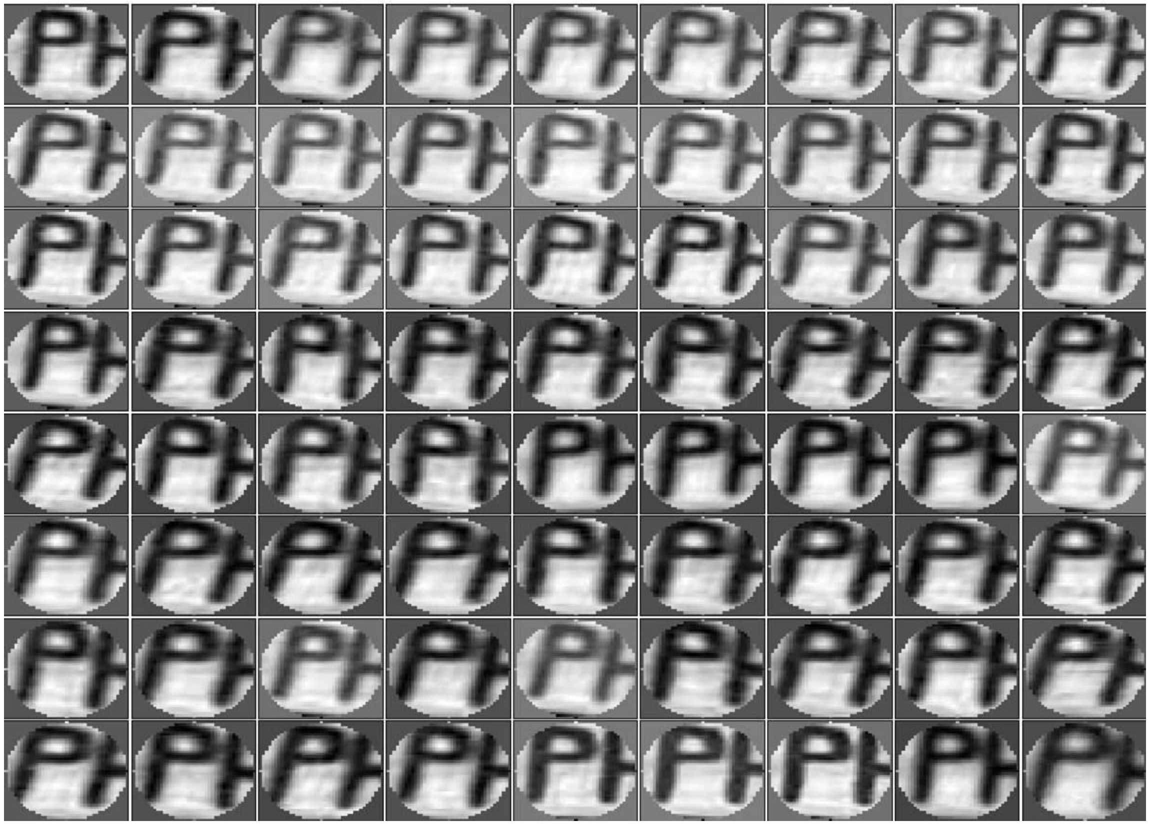


Samples of visual words (clusters on SIFT descriptors):



More specific example

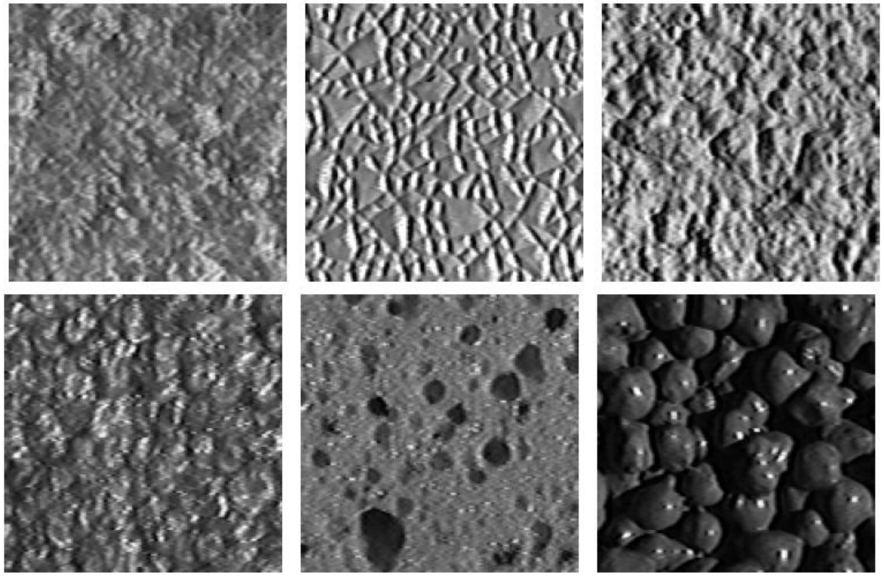
Samples of visual words (clusters on SIFT descriptors):



More specific example

Visual words

- First explored for texture and material representations
- *Texton* = cluster center of filter responses over collection of images
- Describe textures and materials based on distribution of prototypical texture elements.



Leung & Malik 1999; Varma & Zisserman, 2002; Lazebnik, Schmid & Ponce, 2003;

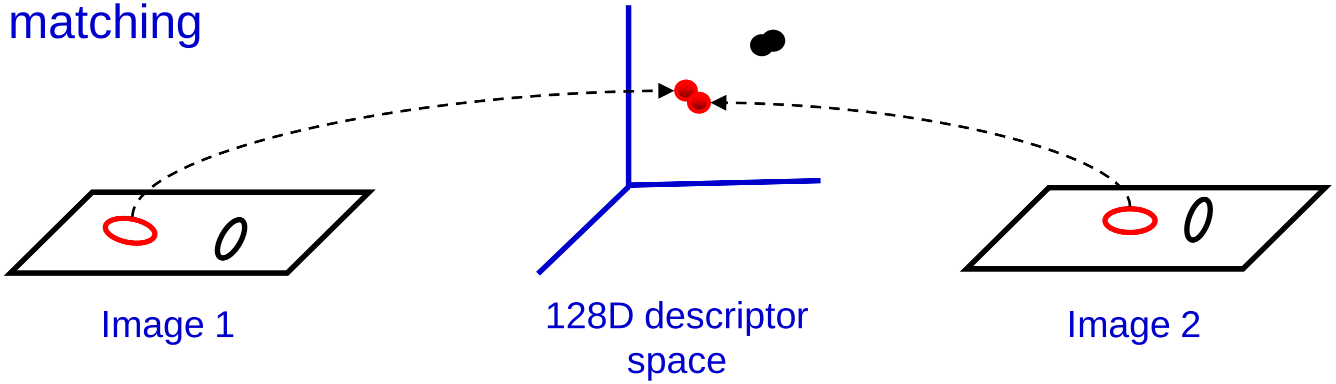
Slide: Grauman&Leibe

Visual words: quantize descriptor space

Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames

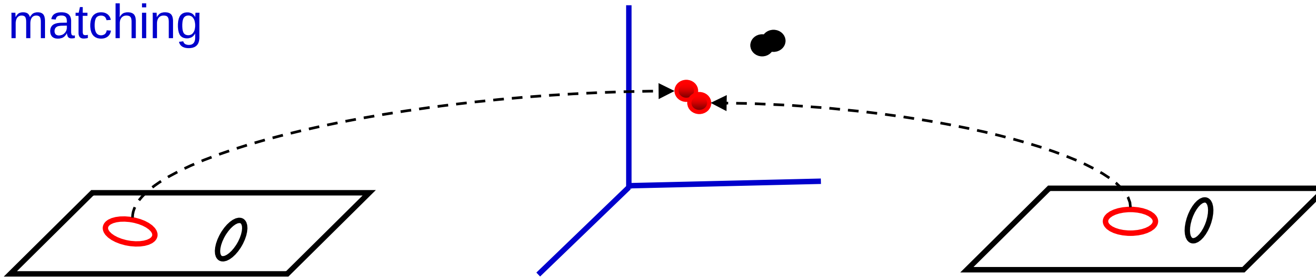


Visual words: quantize descriptor space

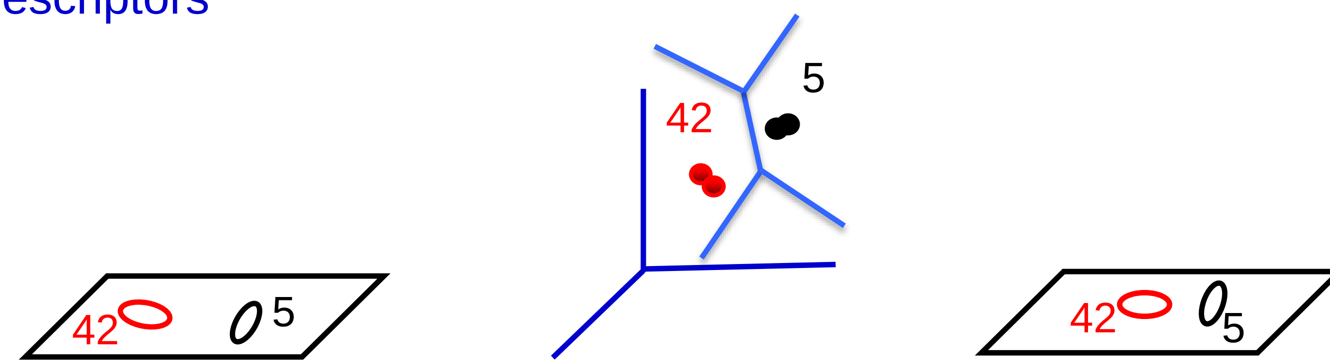
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames



Vector quantize descriptors

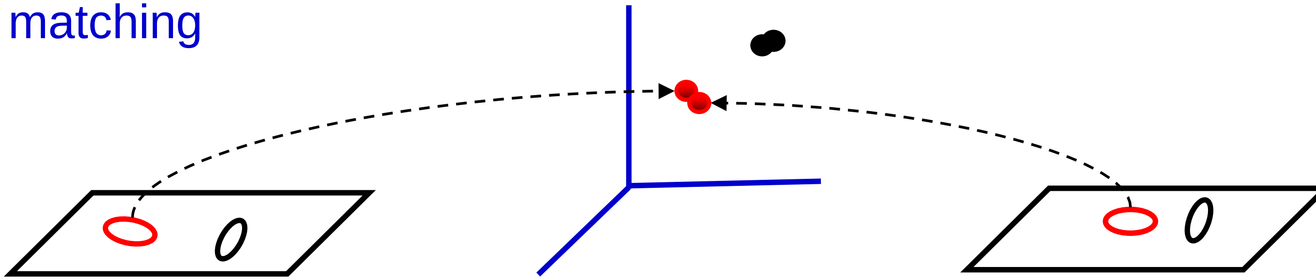


Visual words: quantize descriptor space

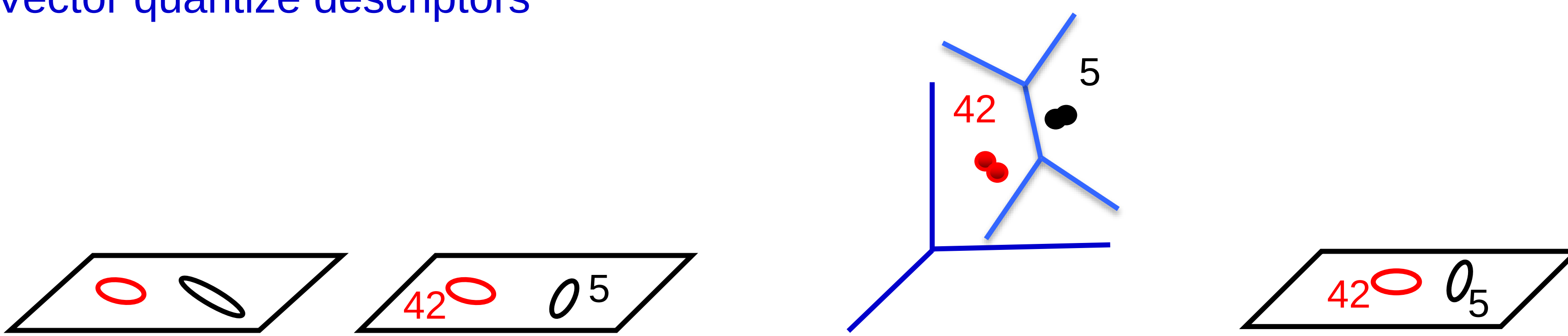
Sivic and Zisserman, ICCV 2003

Nearest neighbour matching

- expensive to do for all frames



Vector quantize descriptors



Visual words: quantize descriptor space

Sivic and Zisserman, ICCV 2003

Nearest neighbour matching
- expensive to do for all frames

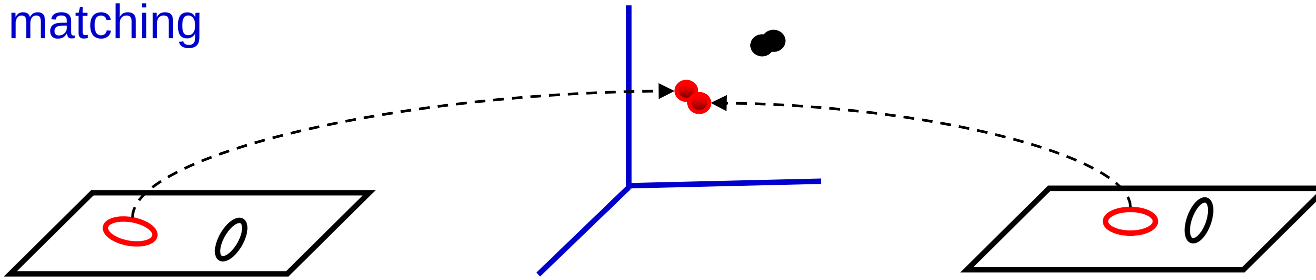
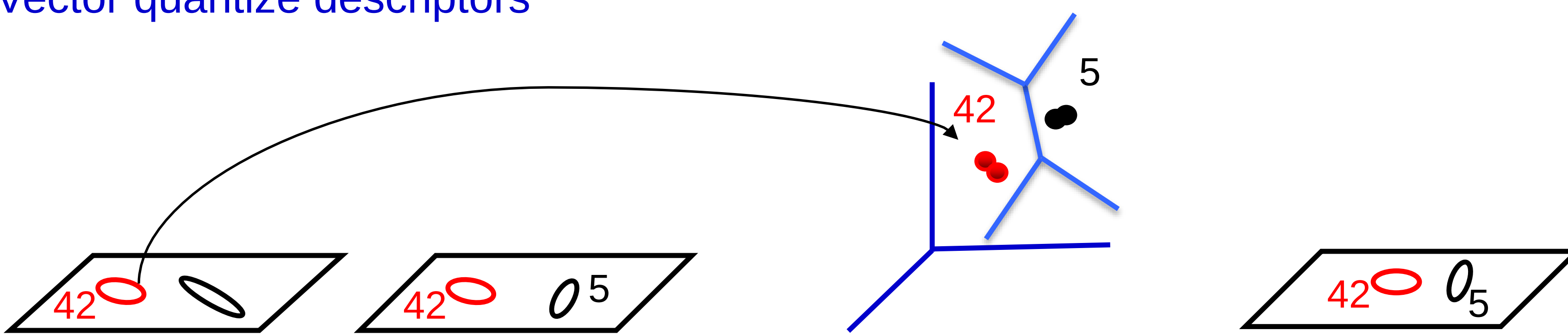


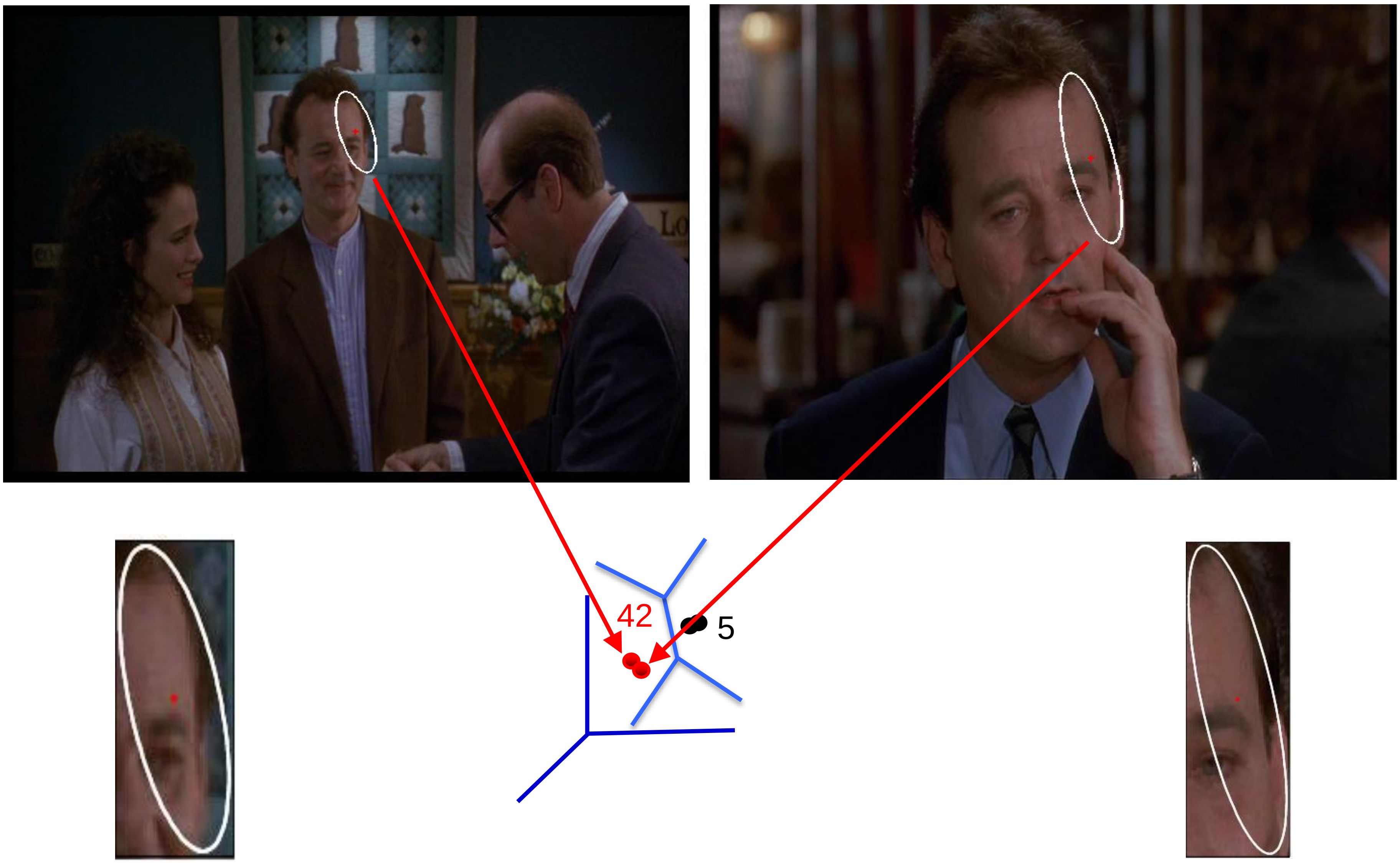
Image 1 | 128D descriptor space | Image 2

Vector quantize descriptors



New image | Image 1 | 128D descriptor space | Image 2

Vector quantize the descriptor space (SIFT)

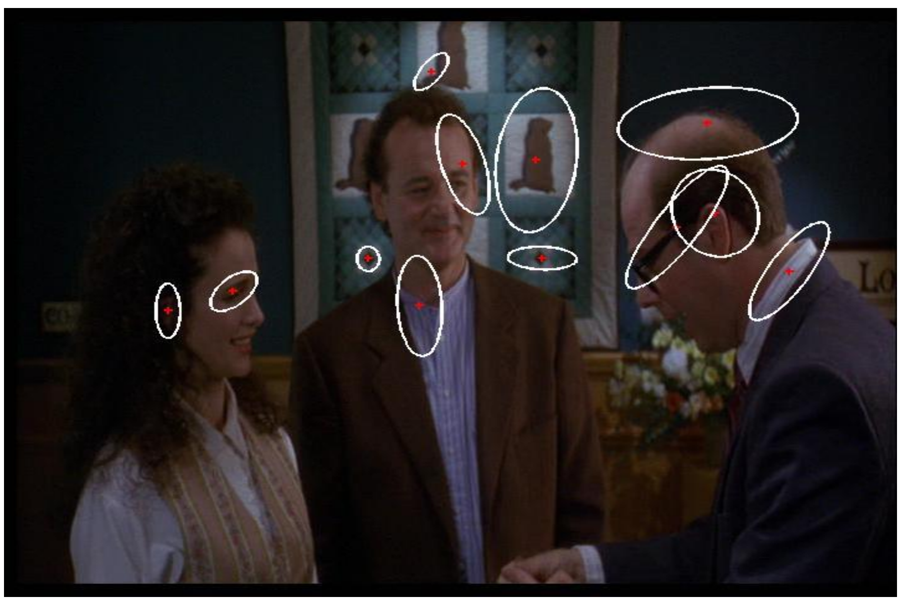


The same visual word

Representation: bag of (visual) words

Visual words are ‘iconic’ image patches or fragments

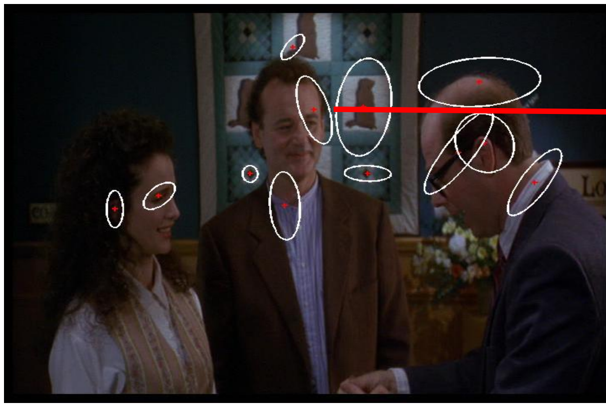
- represent their frequency of occurrence
- but not their position



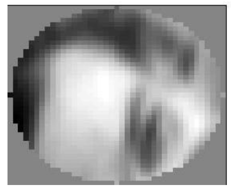
Image

Colelction of visual words

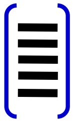
Offline: Assign visual words and compute histograms for each image



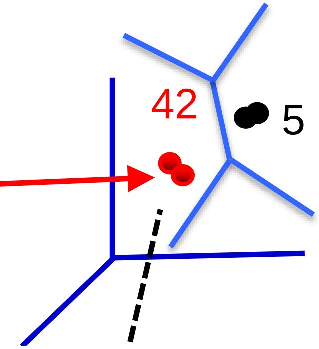
Detect patches



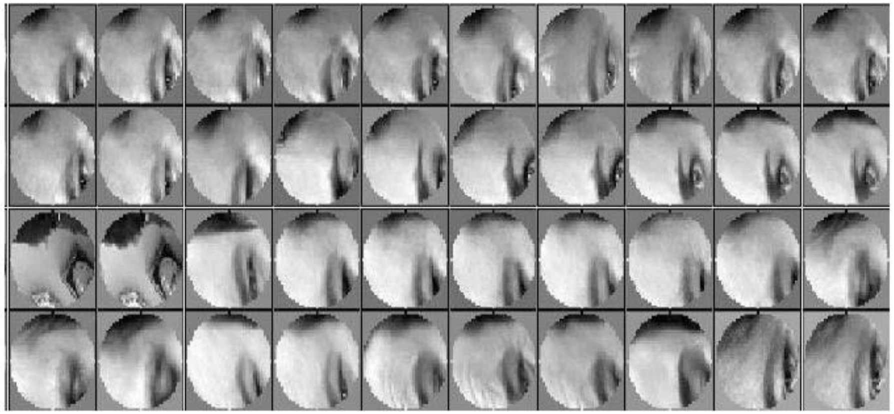
Normalize patch



Compute SIFT descriptor



Find nearest cluster center



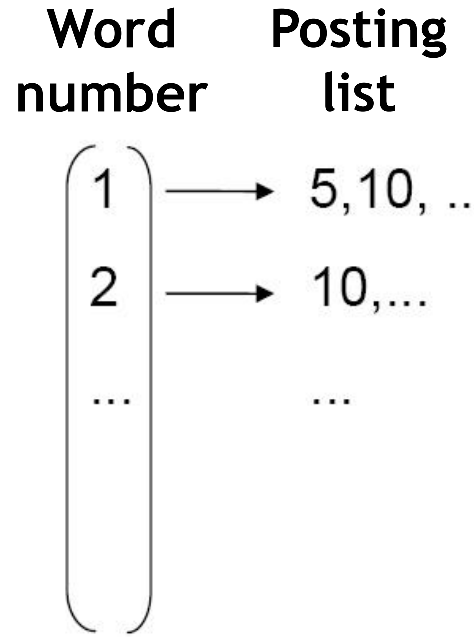
$$\begin{bmatrix} 2 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ \ldots \end{bmatrix}$$

Represent image as a sparse histogram of visual word occurrences

Offline: create an index

frame #5

frame #10



- For fast search, store a “posting list” for the dataset
- This maps visual word occurrences to the images they occur in

(i.e. like the “book index”)

At run time

- User specifies a query region
- Generate a short-list of images using visual words in the region
    1. Accumulate all visual words within the query region
    2. Use “book index” to find other frames with these words
    3. Compute similarity for images which share at least one word

At run time

frame #5

frame #10

Word number	Posting list
1	5,10, ...
2	10,...
...	...

- Score each image by the (weighted) number of common visual words (tentative correspondences)
- Worst case complexity is linear in the number of images N
- In practice, it is linear in the length of the lists (<< N)

Another interpretation: the bag-of-visual-words model

For a vocabulary of size K, each image is represented by a K-vector

$$\mathbf{v}_d = (t_1, \ldots, t_i, \ldots, t_K)^\top$$

where t_i is the number of occurrences of visual word i.

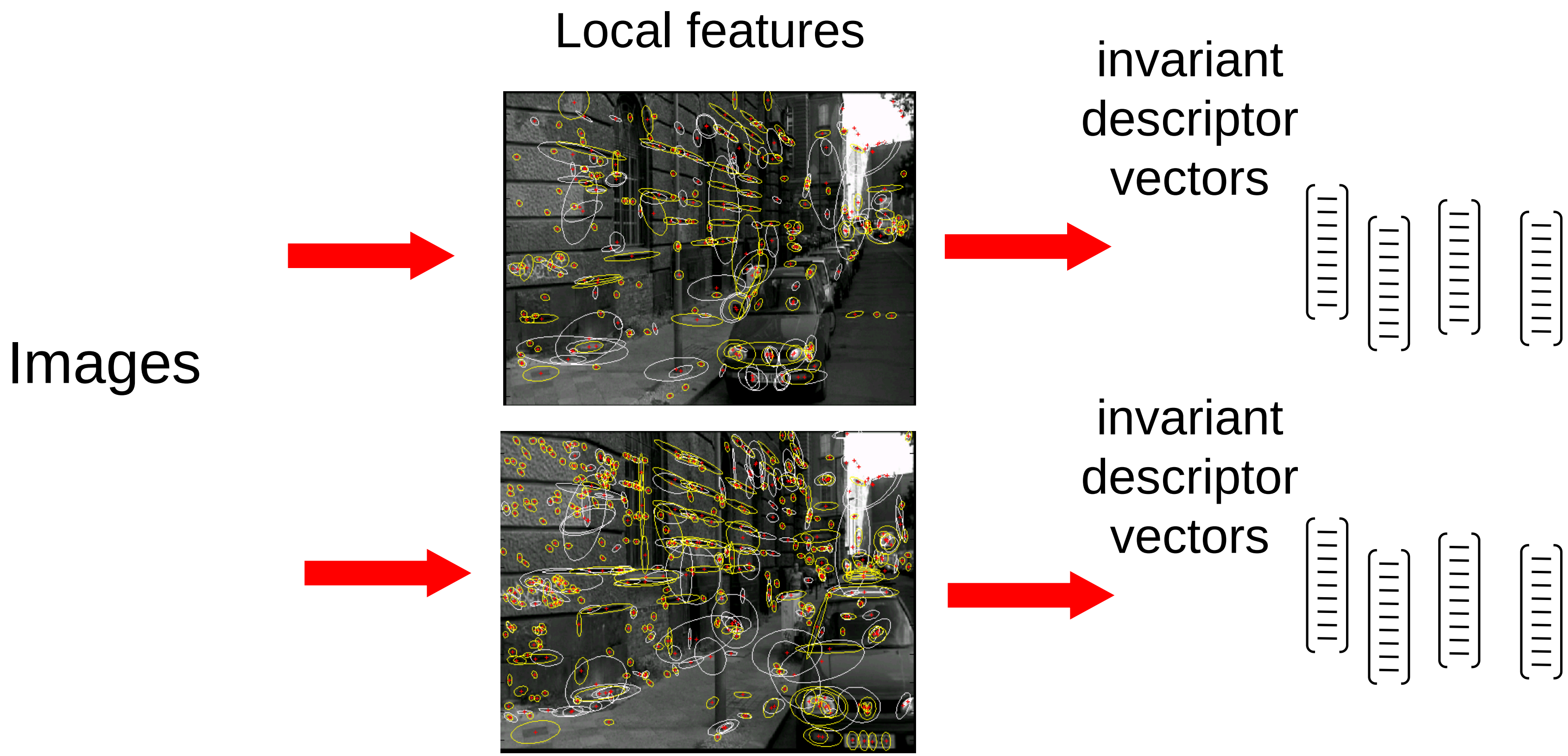
Images are ranked by the normalized scalar product between the query vector v_q and all vectors in the database v_d:

$$f_d = \frac{\mathbf{v}_q{}^\top \mathbf{v}_d}{\|\mathbf{v}_q\|_2 \ \|\mathbf{v}_d\|_2}$$

Scalar product can be computed efficiently using inverted file.

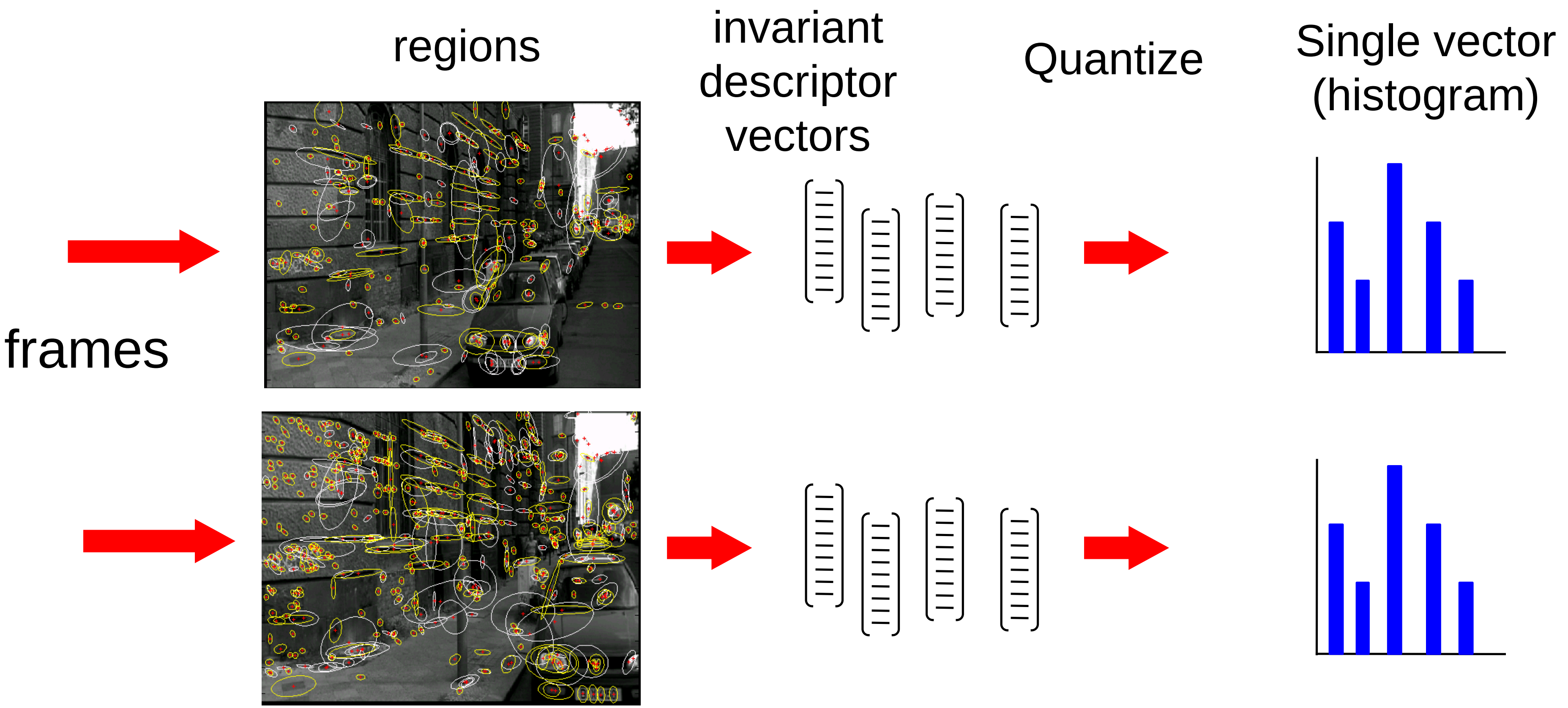
What if vectors are binary? What is the meaning of $\mathbf{v}_q{}^\top \mathbf{v}_d$?

Strategy I: Efficient approximate NN search



1. Compute local features in each image independently (offline)
2. “Label” each feature by a descriptor vector based on its intensity (offline)
3. Finding corresponding features is transformed to finding nearest neighbour vectors
4. Rank matched images by number of (tentatively) corresponding regions
5. Verify top ranked images based on spatial consistency (The first part of this lecture)

Strategy II: Match histograms of visual words



1. Compute affine covariant regions in each frame independently (offline)
2. “Label” each region by a vector of descriptors based on its intensity (offline)
3. **Build histograms of visual words by descriptor quantization (offline)**
4. **Rank retrieved frames by matching vis. word histograms using inverted files.**
5. Verify retrieved frame based on spatial consistency (The first part of the lecture)

Visual words: discussion I.

Efficiency – cost of quantization

- Need to still assign each local descriptor to one of the cluster centers. Could be prohibitive for large vocabularies (K=1M)

- Approximate NN-search still needed
    - e.g. randomized k-d trees

- True also for building the vocabulary
    - approximate k-means [Philbin et al. 2007]

Visual words: discussion II.

Generalization

- Is vocabulary/quantization learned on one dataset good for searching another dataset?

- Experimentally observe a loss in performance.

But, see also a recent work by Jegou et al.:

Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search, ECCV'2008

http://lear.inrialpes.fr/pubs/2008/JDS08a/

Visual words: discussion III.

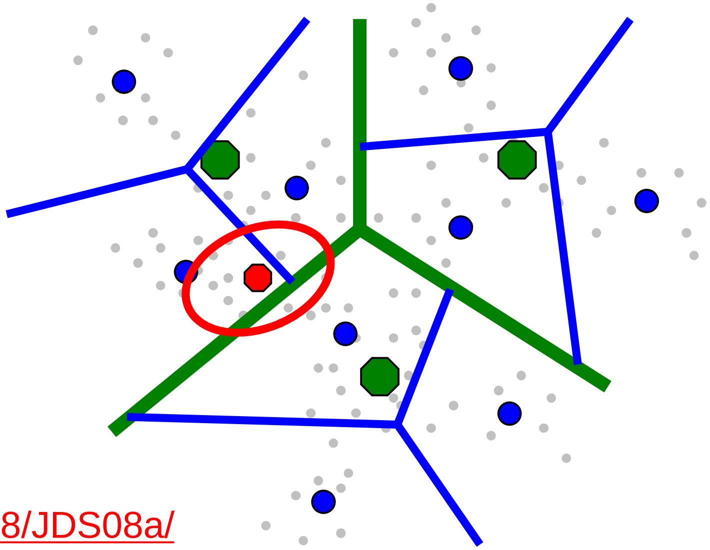
What about quantization effects?

- Visual word assignment can change due to e.g.

  noise in region detection,

  descriptor computation or

  non-modeled image variation (3D effects, lighting)



See also:

Jegou et al., ECCV'2008, http://lear.inrialpes.fr/pubs/2008/JDS08a/

Philbin et al. CVPR'08, http://www.robots.ox.ac.uk/~vgg/publications/html/philbin08-bibtex.html

Mikulik et al., ECCV'10, http://cmp.felk.cvut.cz/~chum/papers/mikulik_eccv10.pdf

Philbin et al., ECCV'10, http://www.di.ens.fr/~josef/publications/philbin10b.pdf

Visual words: discussion IV.

• Need to determine the size of the vocabulary, K.

• Other algorithms for building vocabularies, e.g.
agglomerative clustering / mean-shift, but typically more expensive.

• Supervised quantization?
Also give examples of images / descriptors which should and should not match.

E.g.:
Philbin et al. ECCV'10, http://www.robots.ox.ac.uk/~vgg/publications/html/philbin10b-bibtex.html

Visual search using local regions (references)

C. Schmid, R. Mohr, Local Greyvalue Invariants for Image Retrieval, PAMI, 1997

J. Sivic, A. Zisserman, Text retrieval approach to object matching in videos, ICCV, 2003

D. Nister, H. Stewenius, Scalable Recognition with a Vocabulary Tree, CVPR, 2006.

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

O. Chum, J. Philbin, M. Isard, J. Sivic, A. Zisserman, Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval, ICCV, 2007

H. Jegou, M. Douze, C. Schmid, Hamming embedding and weak geometric consistency for large scale image search, ECCV'2008

O. Chum, M. Perdoch, J. Matas: Geometric min-Hashing: Finding a (Thick) Needle in a Haystack, CVPR 2009

H. Jégou, M. Douze and C. Schmid, On the burstiness of visual elements, CVPR, 2009

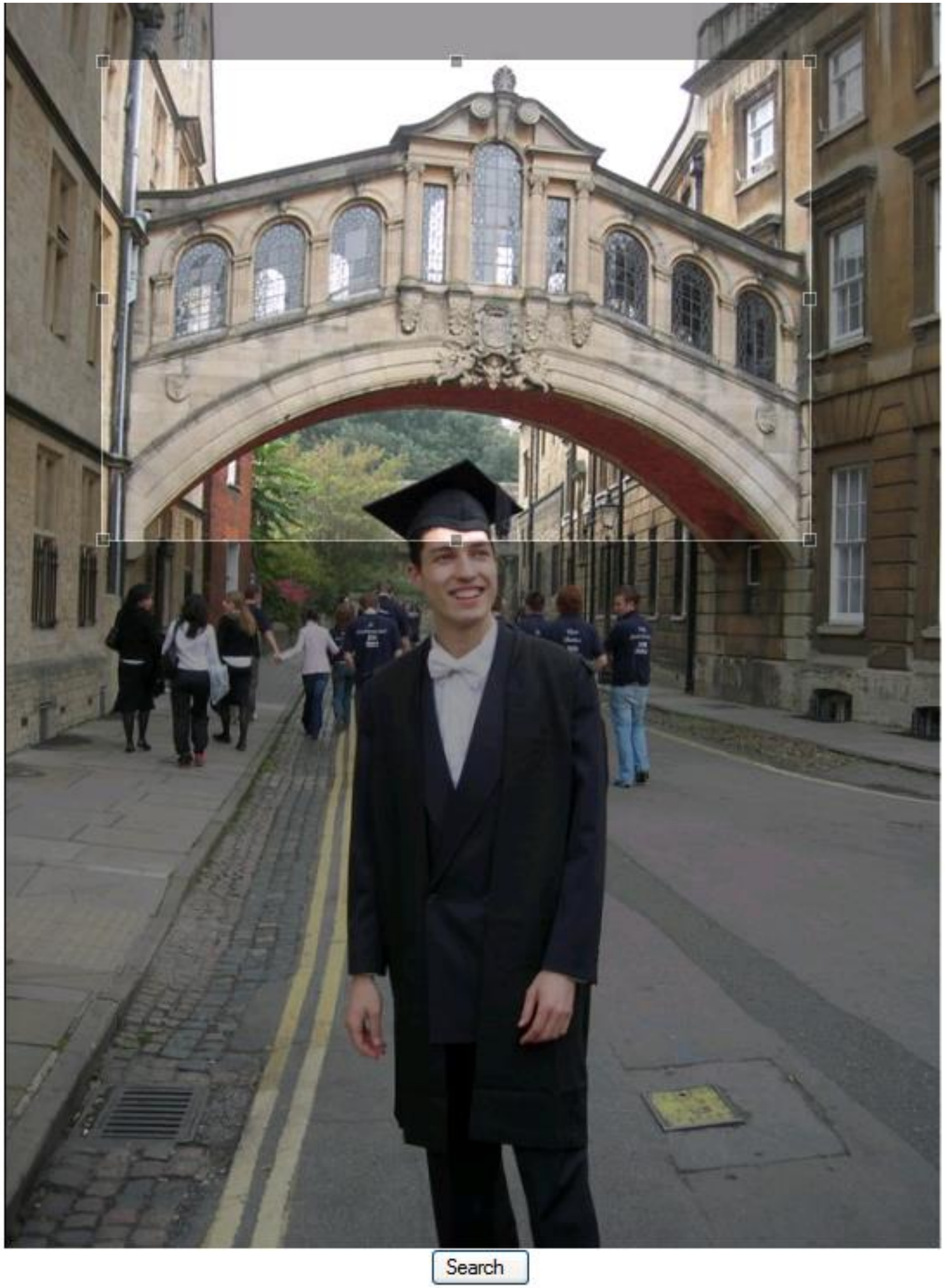
H. Jégou, M. Douze, C. Schmid and P. Pérez, Aggregating local descriptors into a compact image representation, CVPR'2010

Efficient visual search for objects and places

Oxford Buildings Search - demo

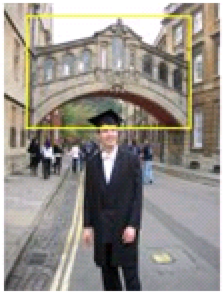
http://www.robots.ox.ac.uk/~vgg/research/oxbuildings/index.html

Example



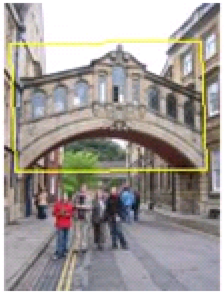
Search results 1 to 20 of 104844

1



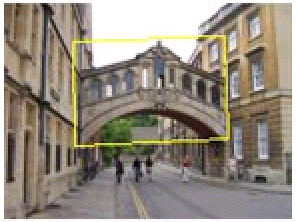
ID: oxc1_hertford_000011
Score: 1816.000000
Putative: 2325
Inliers: 1816
Hypothesis: 1.000000 0.000000 0.000015 0.000000 1.000000 0.000031
Detail

2



ID: oxc1_all_souls_000075
Score: 352.000000
Putative: 645
Inliers: 352
Hypothesis: 1.162245 0.041211 -70.414459 -0.012913 1.146417 91.276093
Detail

3



ID: oxc1_hertford_000064
Score: 278.000000
Putative: 527
Inliers: 278
Hypothesis: 0.928686 0.026134 169.954620 -0.041703 0.937558 97.962112
Detail

4

ID: oxc1_oxford_001612
Score: 252.000000
Putative: 451
Inliers: 252
Hypothesis: 1.046026 0.069416 51.576881 -0.044949 1.046938 76.264442
Detail

5

ID: oxc1_hertford_000123
Score: 225.000000
Putative: 446
Inliers: 225
Hypothesis: 1.361741 0.090413 -34.673317 -0.084659 1.301689 -32.281090
Detail

6

ID: oxc1_oxford_001085
Score: 224.000000
Putative: 389
Inliers: 224
Hypothesis: 0.848997 0.000000 195.707611 -0.031077 0.895546 114.583961
Detail

7

ID: oxc1_hertford_000077
Score: 195.000000
Putative: 386
Inliers: 195
Hypothesis: 1.465144 0.069286 -108.473091 -0.097598 1.461877 -30.205191
Detail

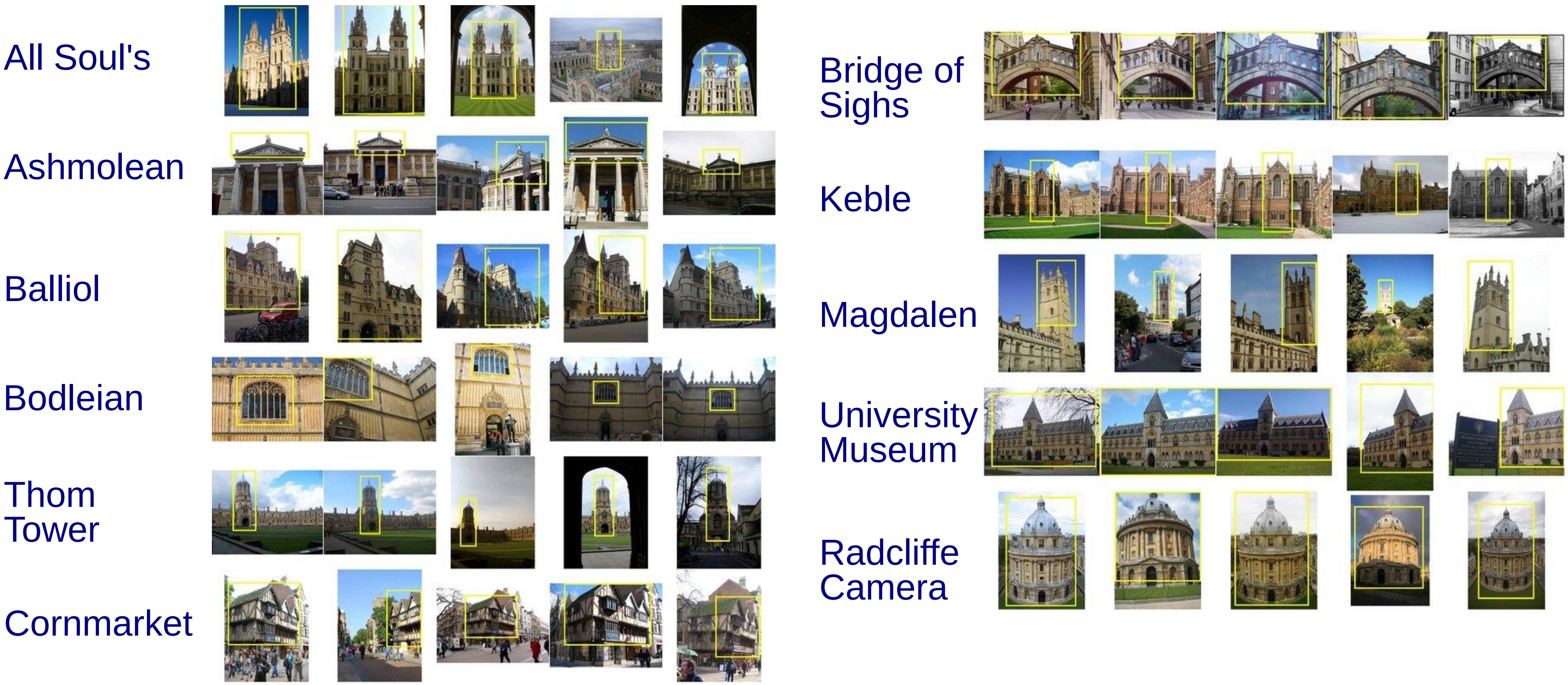
Oxford buildings dataset

- Automatically crawled from flickr
- Consists of:

Dataset	Resolution	# images	# features	Descriptor size
i	1024 × 768	5,062	16,334,970	1.9 GB
ii	1024 × 768	99,782	277,770,833	33.1 GB
iii	500 × 333	1,040,801	1,186,469,709	141.4 GB
Total		1,145,645	1,480,575,512	176.4 GB

Oxford buildings dataset

- Landmarks plus queries used for evaluation

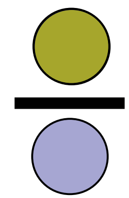


- Ground truth obtained for 11 landmarks

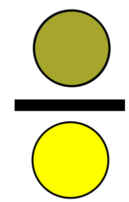
- Evaluate performance by mean Average Precision

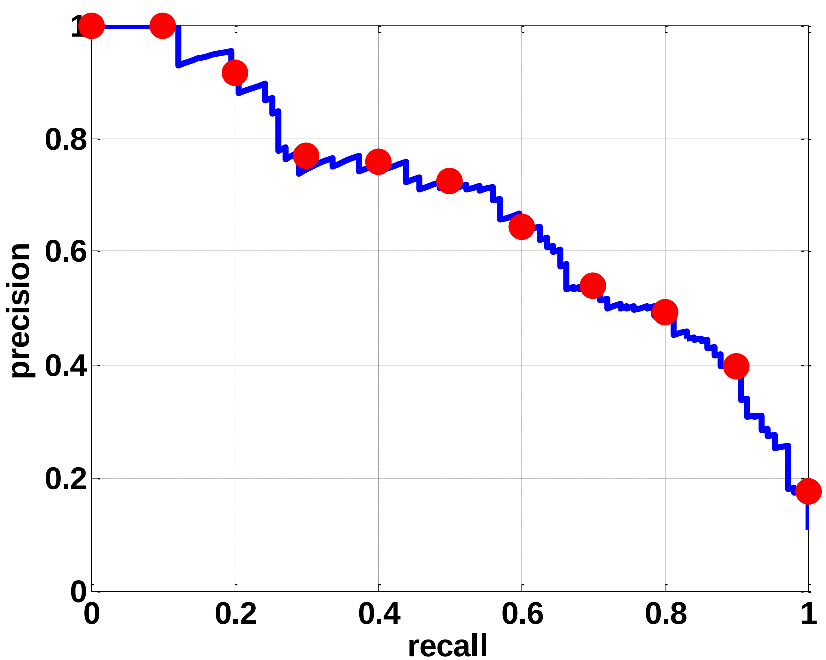
Measuring retrieval performance: Precision - Recall

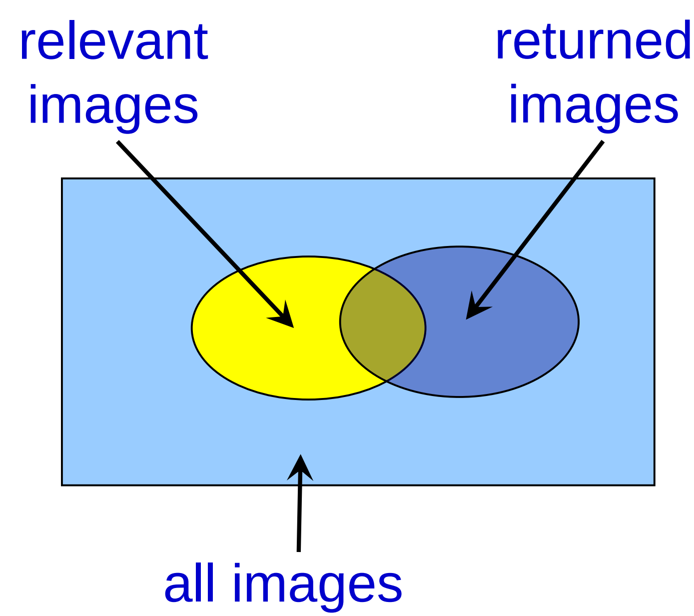
- Precision: % of returned images that are relevant



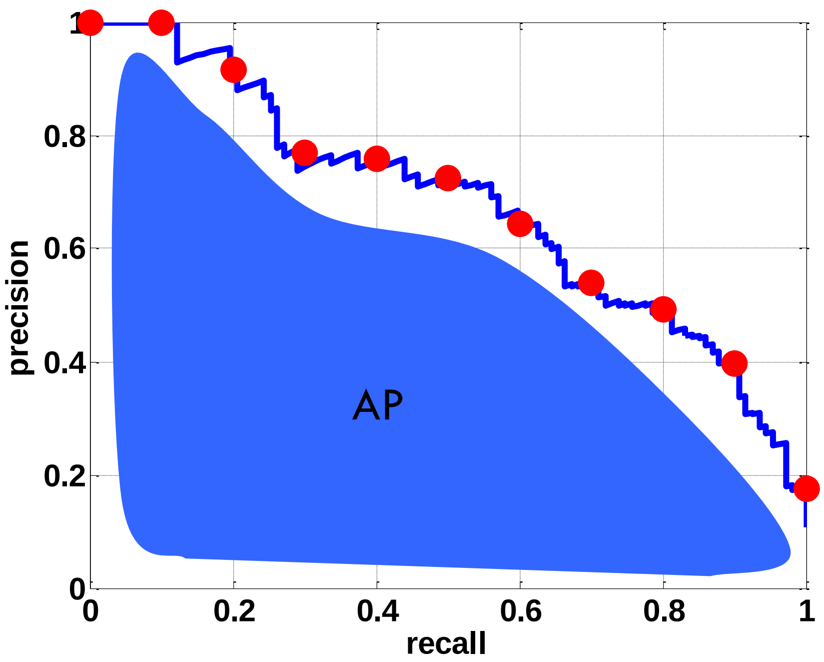
- Recall: % of relevant images that are returned





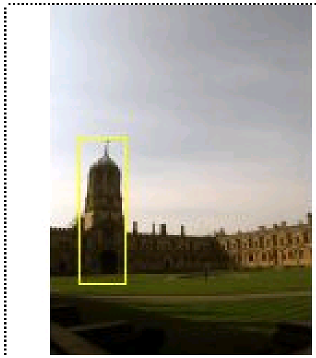


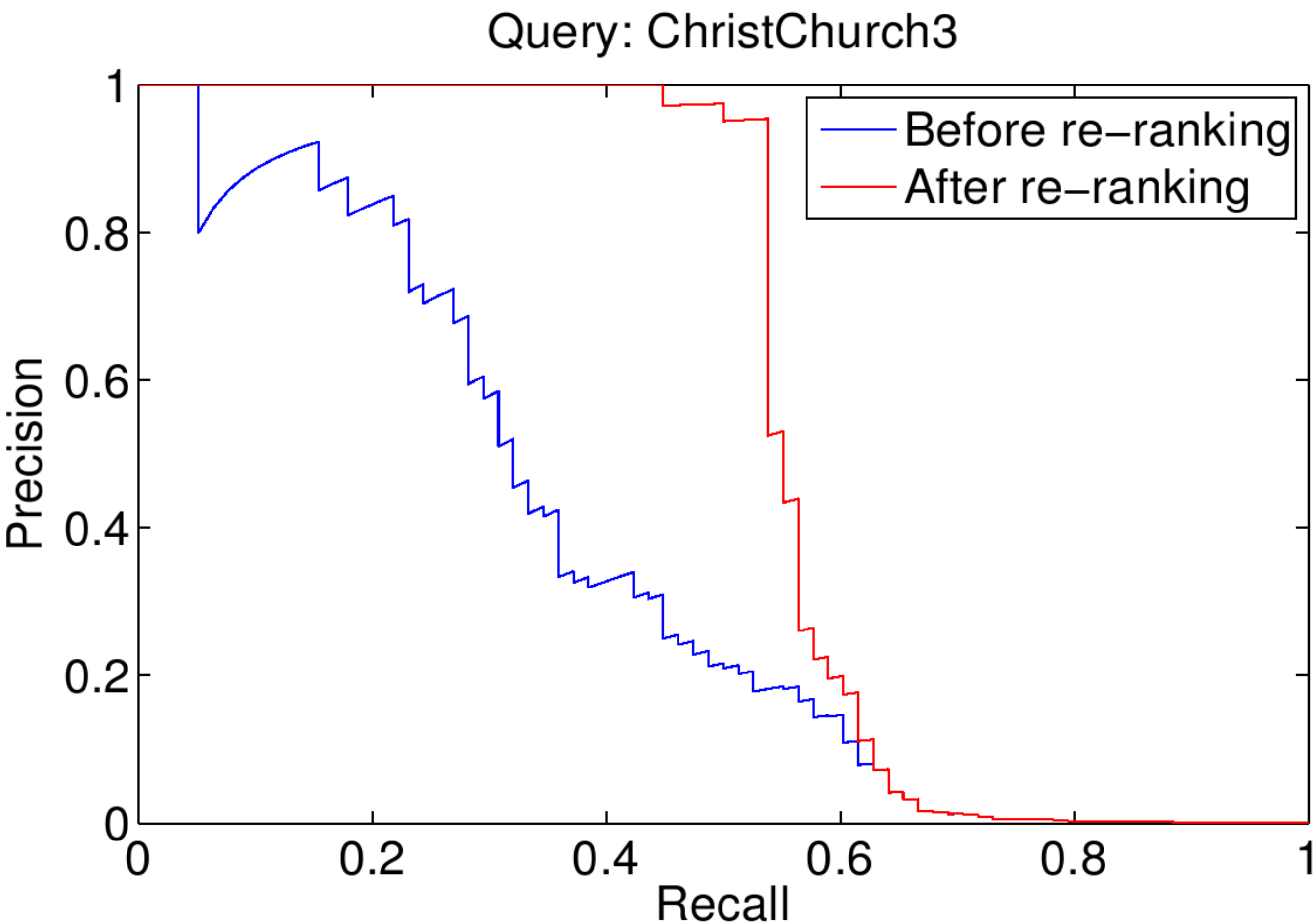
Average Precision



- A good AP score requires both high recall and high precision
- Application-independent

Performance measured by mean Average Precision (mAP) over 55 queries on 100K or 1.1M image datasets

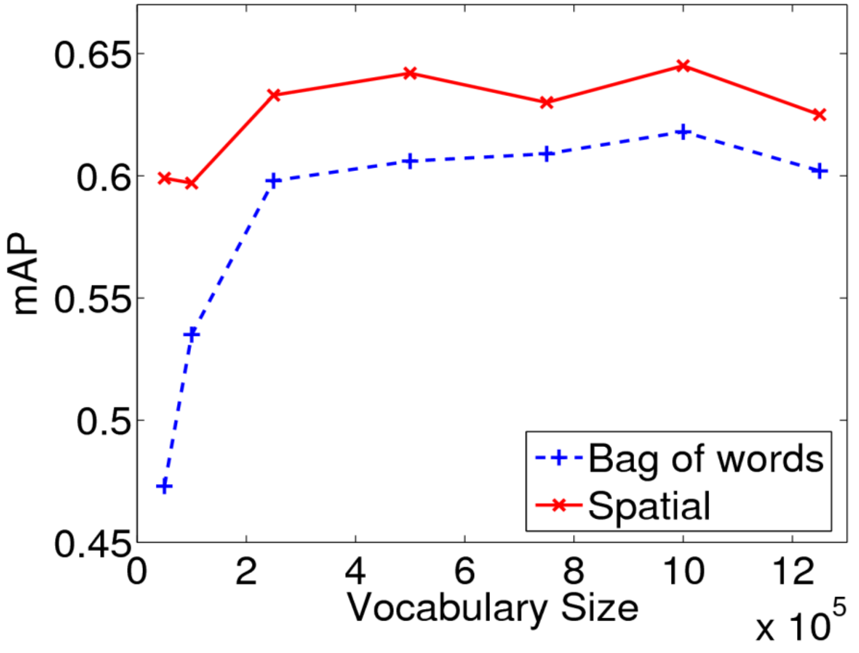




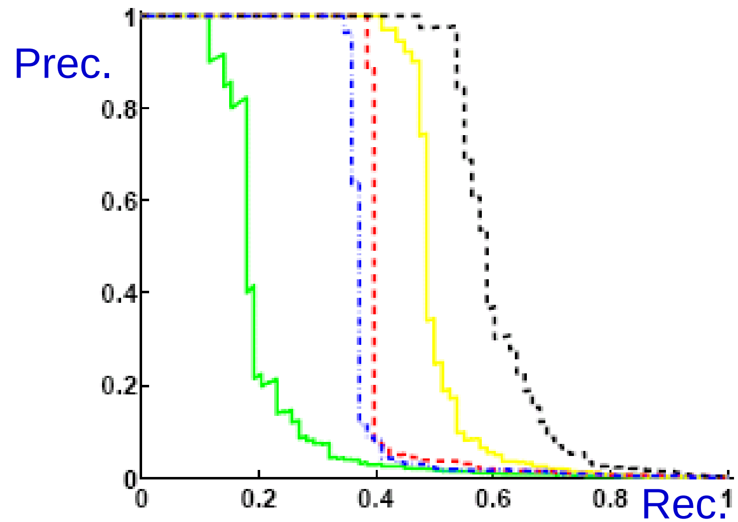
Query: ChristChurch3
Before re–ranking
After re–ranking
Precision
Recall
0
0.2
0.4
0.6
0.8
1

Mean Average Precision variation with vocabulary size

vocab size	bag of words	spatial
50K	0.473	0.599
100K	0.535	0.597
250K	0.598	0.633
500K	0.606	0.642
750K	0.609	0.630
1M	0.618	0.645
1.25M	0.602	0.625

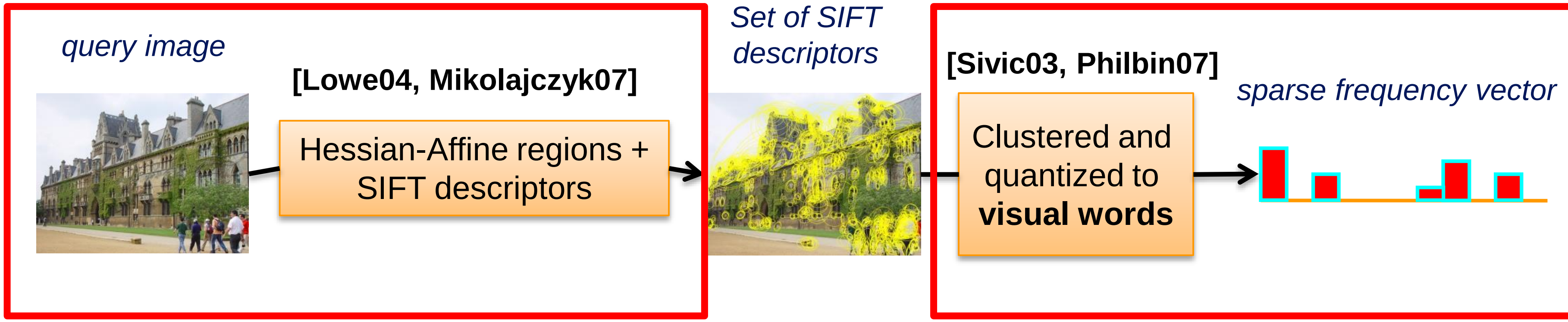


Query images



- high precision at low recall (like google)
- variation in performance over query
- none retrieve all instances

Why aren't all objects retrieved?



Obtaining visual words is like a sensor measuring the image

"noise" in the measurement process means that some visual words are missing or incorrect, e.g. due to

- Missed detections
- Changes beyond built in invariance
- Quantization effects

1. Query expansion
2. Better quantization

Consequence: Visual word in query is missing in target image

Query Expansion in text

In text :

- Reissue top n responses as queries
- Pseudo/blind relevance feedback
- Danger of topic drift

In vision:

- Reissue spatially verified image regions as queries

Query Expansion: Text

Original query: Hubble Telescope Achievements

Query expansion: Select top 20 terms from top 20 documents according to tf-idf

Added terms: Telescope, hubble, space, nasa, ultraviolet, shuttle, mirror, telescopes, earth, discovery, orbit, flaw, scientists, launch, stars, universe, mirrors, light, optical, species

Example from: Jimmy Lin, University of Maryland

Automatic query expansion

Visual word representations of two images of the same object may differ (due to e.g. detection/quantization noise) resulting in missed returns

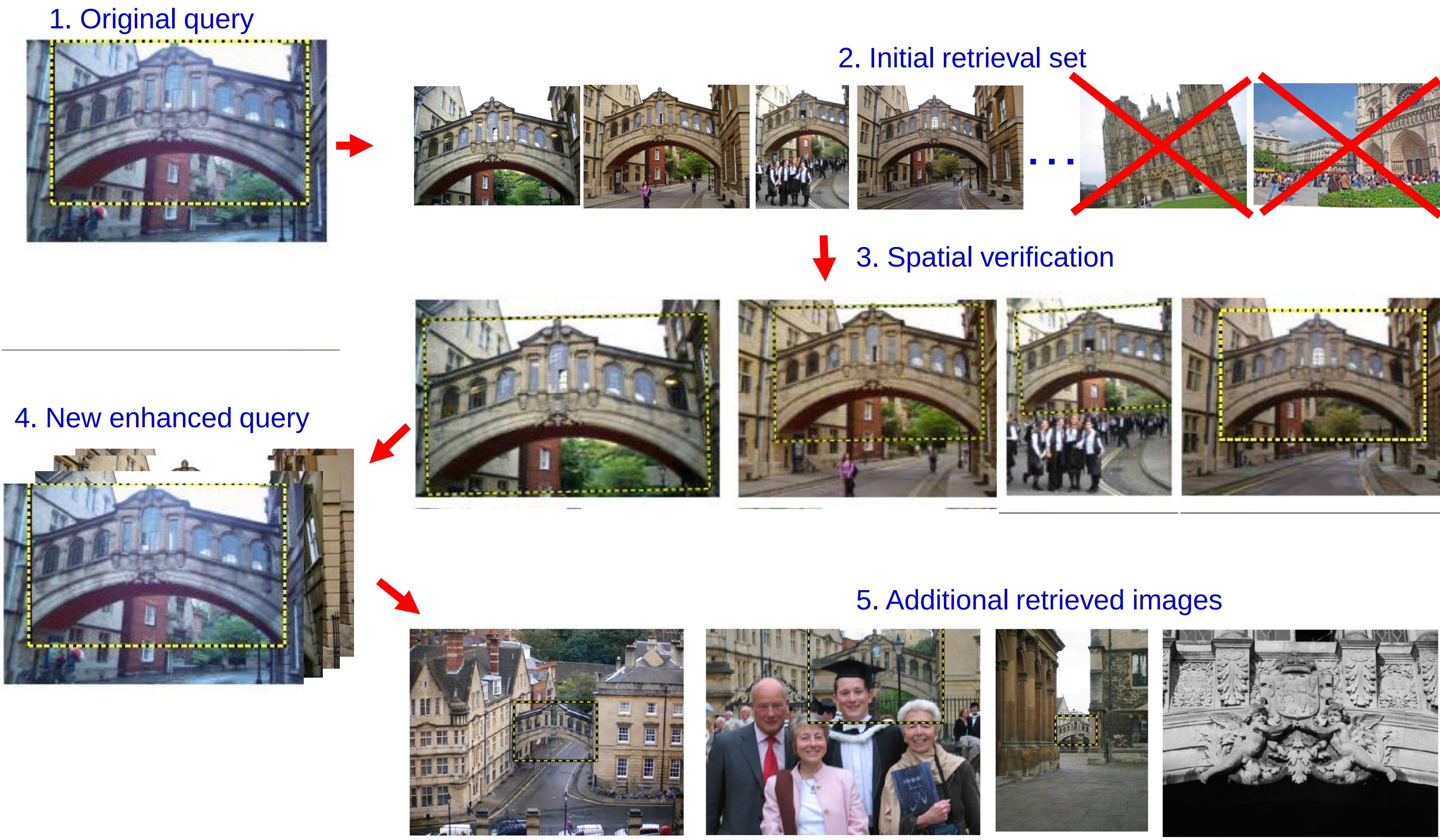
Initial returns may be used to add new relevant visual words to the query

Strong spatial model prevents ‘drift’ by discarding false positives

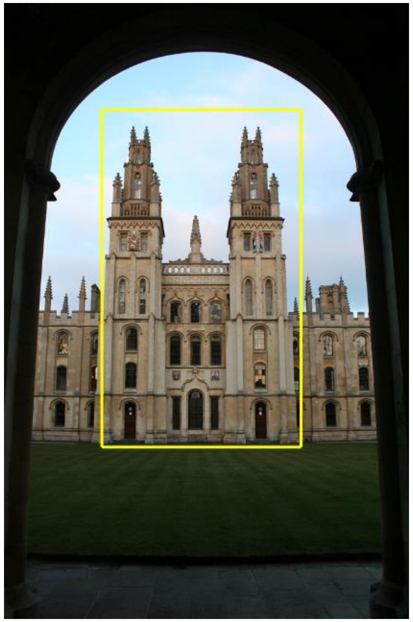
[Chum, Philbin, Sivic, Isard, Zisserman, ICCV’07;

Chum, Mikulik, Perdoch, Matas, CVPR’11]

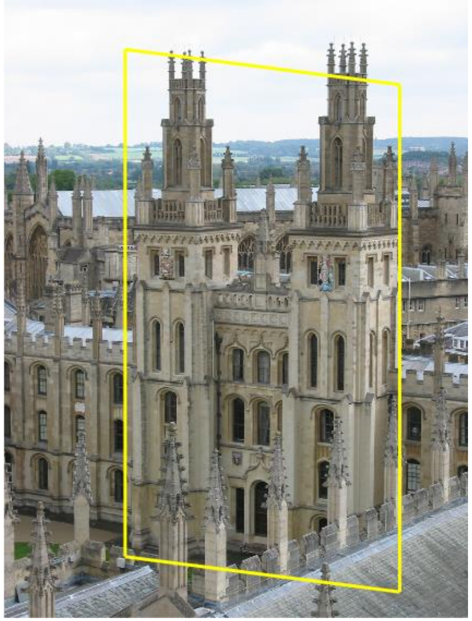
Visual query expansion - overview



Query Expansion



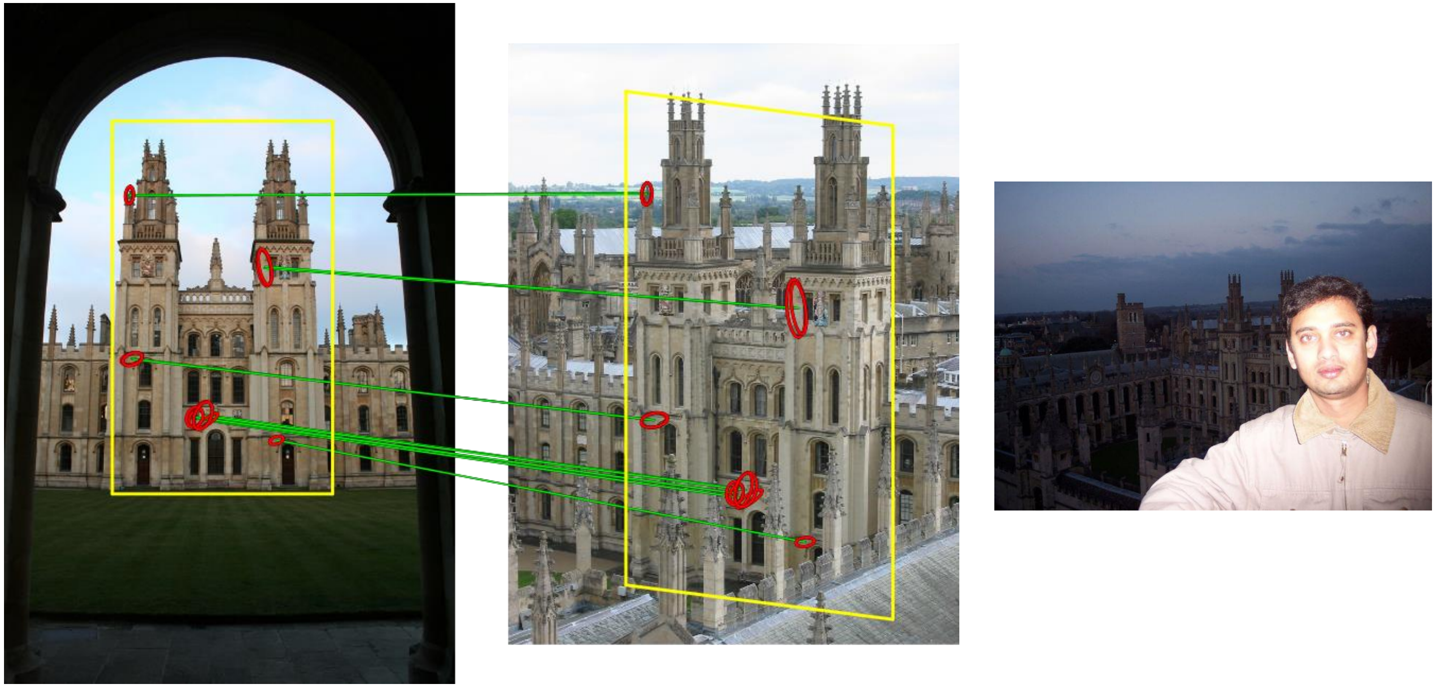
Query Image



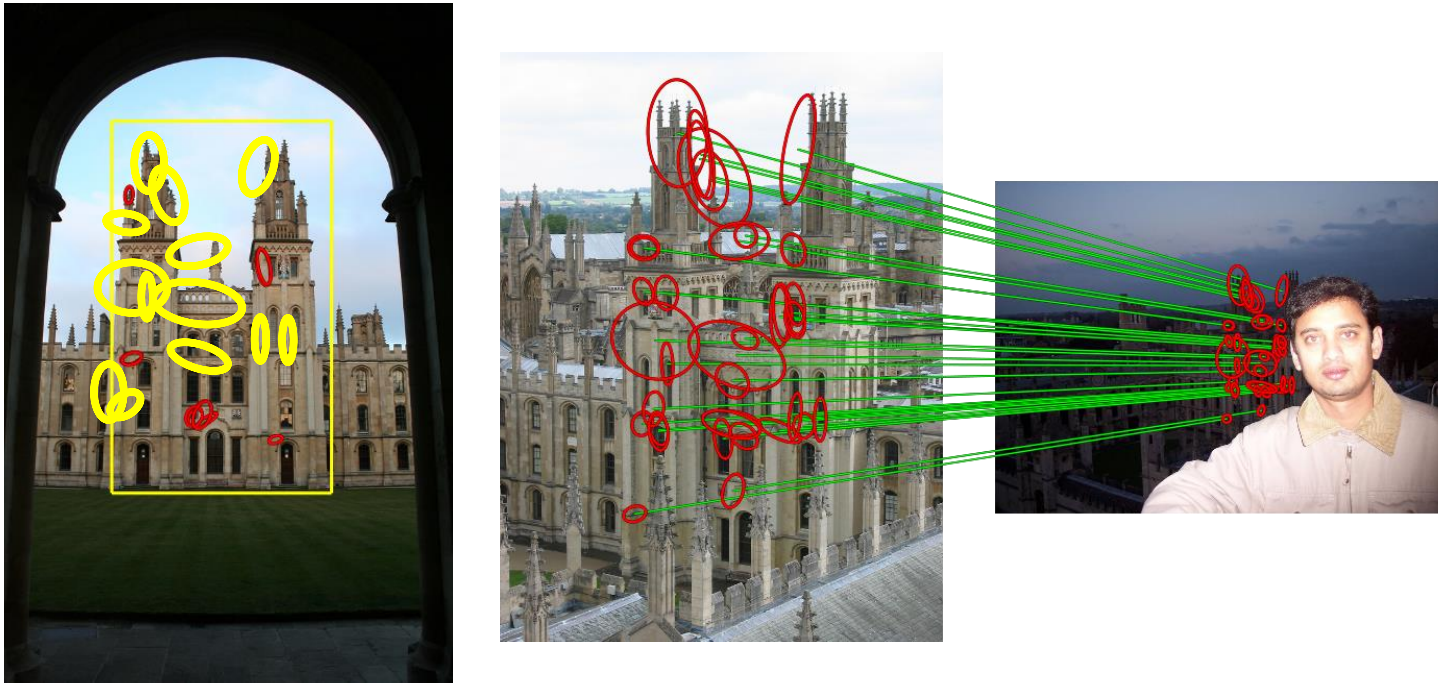
Originally retrieved image

Originally not retrieved

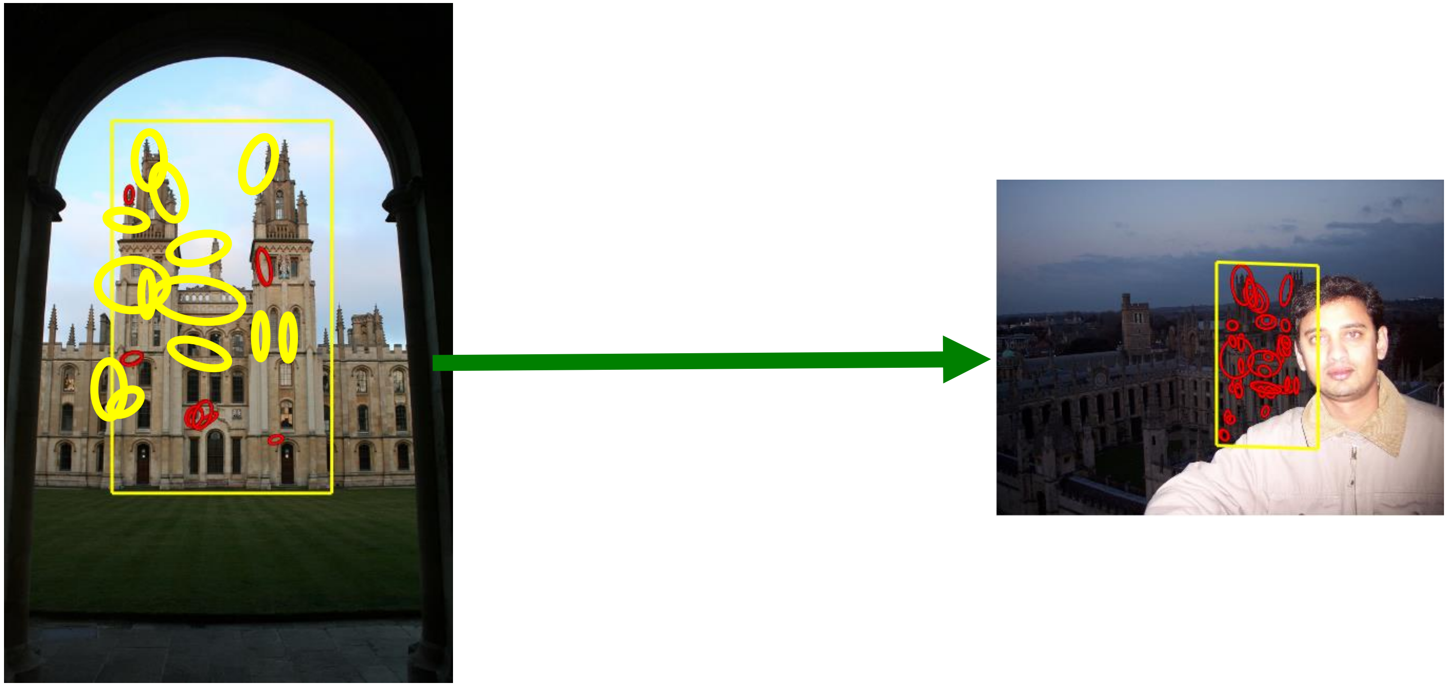
Query Expansion



Query Expansion

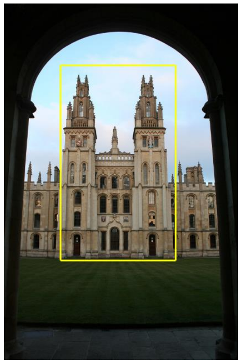


Query Expansion

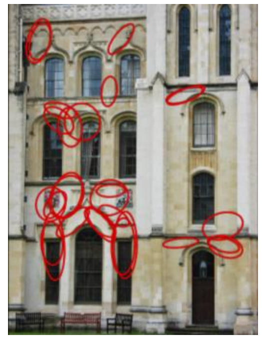


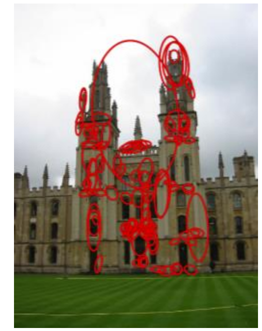
Query Expansion

Query Image



Spatially verified retrievals with matching regions overlaid






...

New expanded query

New expanded query is formed as

- the average of visual word vectors of spatially verified returns
- only inliers are considered
- regions are back-projected to the original query image

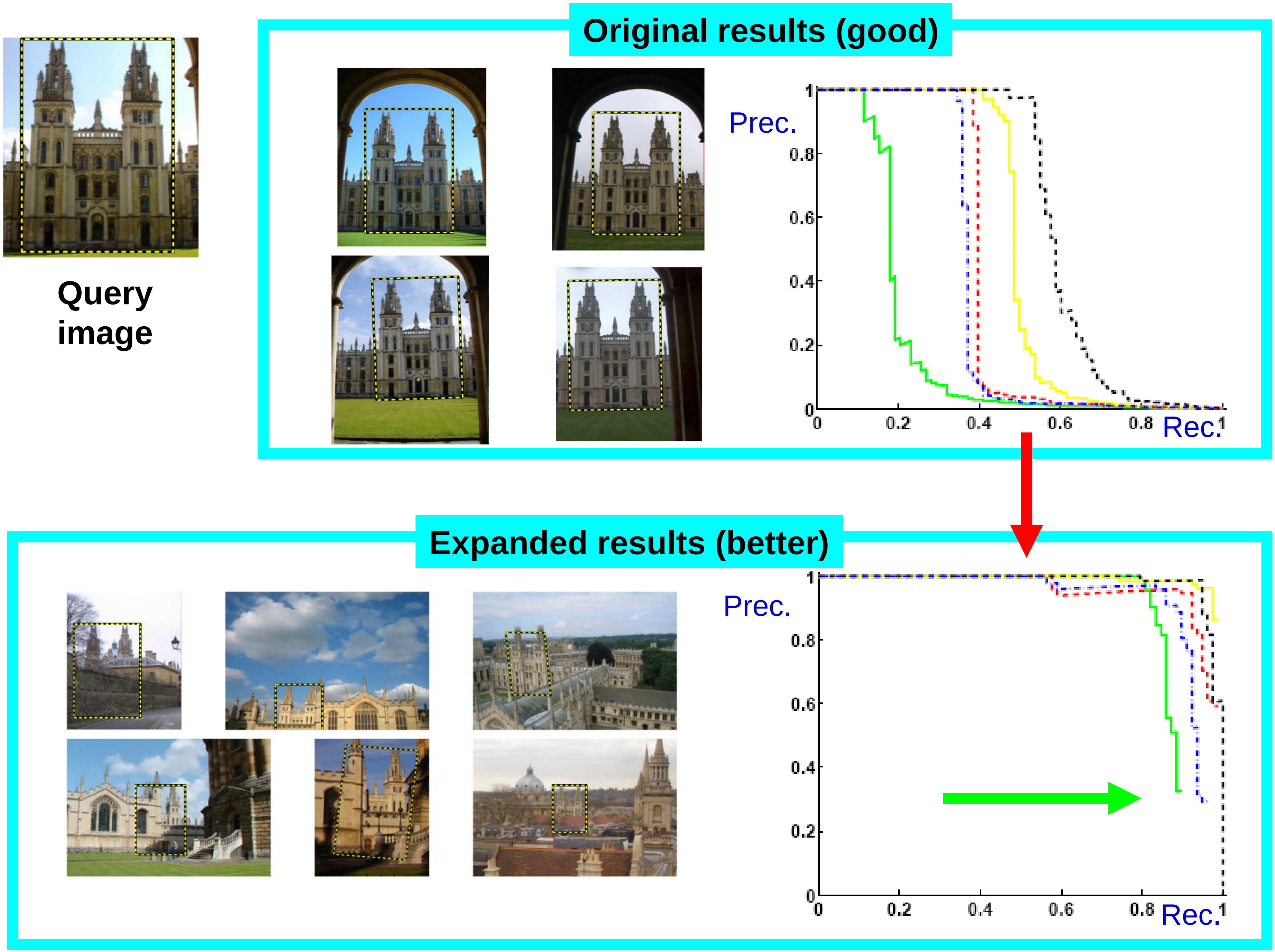
Demo

Query Expansion

Query image

Originally retrieved

Retrieved only after expansion



Query image
Original results (good)
Prec.
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
Rec.
1
Expanded results (better)
Prec.
1
0.8
0.6
0.4
0.2
0
0
0.2
0.4
0.6
0.8
Rec.
1

What objects/scenes local regions do not work on?

What objects/scenes local regions do not work on?

(a) (b)

(c) (d) (e) (f) (g) (h)

E.g. texture-less objects, objects defined by shape, deformable objects, wiry objects.

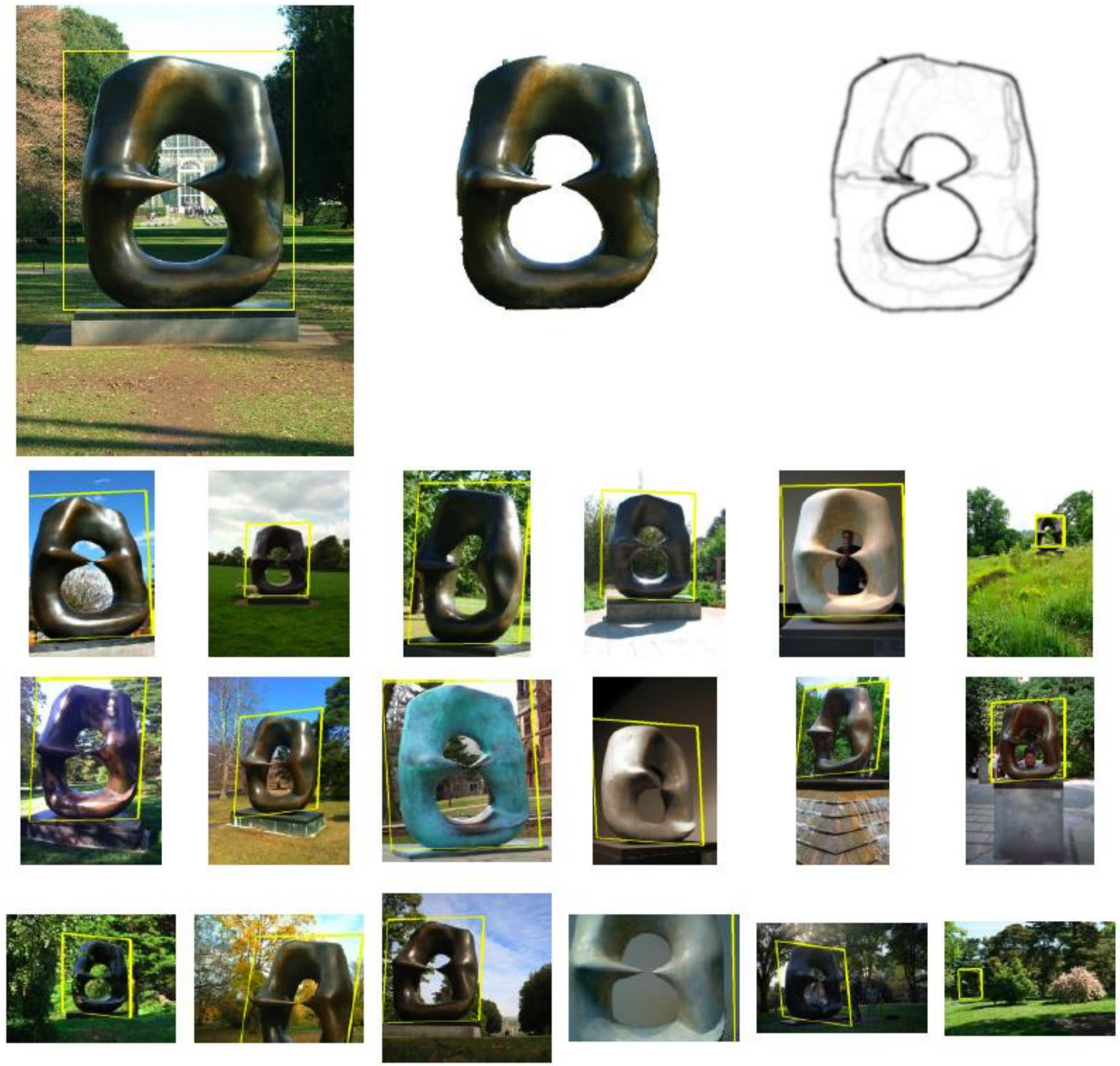
What next?

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

Example:
Smooth object retrieval using a bag of boundaries by Arandjelovic and Zisserman, ICCV 2011

Query

Retrieved matches



Category-level visual search [See next lecture]

Query

same category

See also e.g. [Torresani et al. ECCV 2010]

What next?

Match objects across large changes of appearance

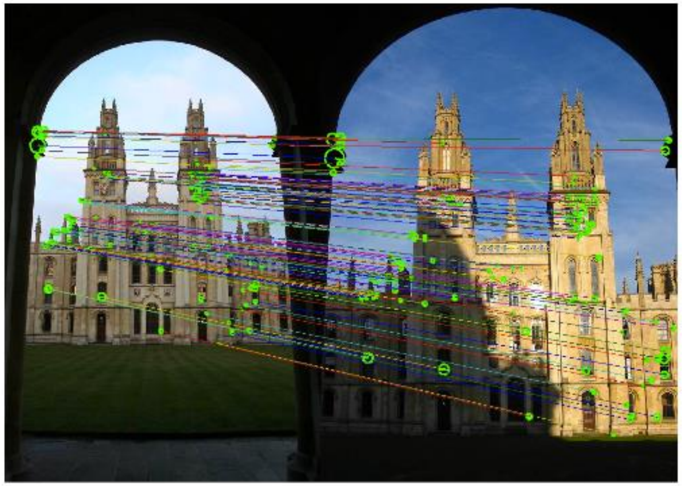
Examples: non-photographic depictions, degradation over time, change of season, …

Useful practical exercise (Matlab)

http://www.di.ens.fr/willow/teaching/recvis14/assignment1/

Assignment 1: Instance-level recognition

(Adapted from A. Vedaldi and A. Zisserman)



Goal

The goal of instance-level recognition is to match (recognize) a specific object or scene. Examples include recognizing a specific building, such as Notre Dame, or a specific painting, such as `Starry Night' by Van Gogh. The object is recognized despite changes in scale, camera viewpoint, illumination conditions and partial occlusion. An important application is image retrieval - starting from an image of an object of interest (the query), search through an image dataset to obtain (or retrieve) those images that contain the target object.

The goal of this assignment is to experiment and get basic practical experience with the methods that enable specific object recognition. It includes: (i) using SIFT features to obtain sparse matches between two images; (ii) using affine co-variant detectors to cover changes in viewpoint; (iii) vector quantizing the SIFT descriptors into visual words to enable large scale retrieval; and (iv) constructing and using an image retrieval system to identify objects.