Matching Indexing Search

Computer Vision – Lecture 05

Further reading

- Slides from A Vedaldi
- Slides from S Lazebnik
- Slides from J Johnson and D Fouhey

So far

- Lectures 02 – 04:
  - Various operations a single image

- Usually, we are dealing with more than one image.

- We want to relate one image to another in various ways.
  - Points
  - Transformations
  - Images

What are the three most important problems in computer vision?

1. Correspondence
2. Correspondence
3. Correspondence

Takeo Kanade (金出 武雄)

Image source

Correspondences

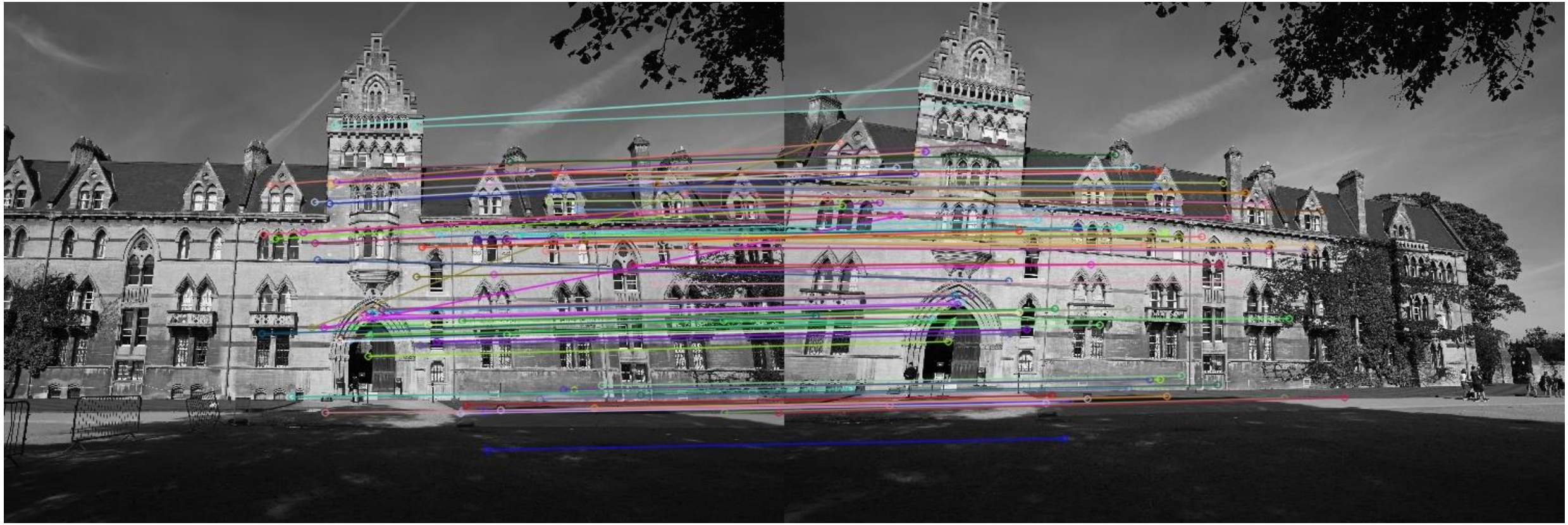
- Given two images of the same scene
- Find corresponding points

Correspondences

- Simply subtracting the images does not work

Correspondences

Point correspondences estimated by a classic algorithm: SIFT



Scale Invariant Feature Transform (SIFT)

TITLE	CITED BY	YEAR
Distinctive image features from scale-invariant keypoints DG Lowe International journal of computer vision 60 (2), 91-110	73590	2004

Algorithm overview:

1. Keypoint detection
2. Keypoint description
3. Keypoint matching

Keypoints

- What makes a good keypoint?
- Easy to find in other images!

Intuitions:

- A: untextured regions are difficult to reidentify
- B: good keypoint
- C: stable under viewpoint changes, but multiple occurrences
- D: hard to find exact point along the edge

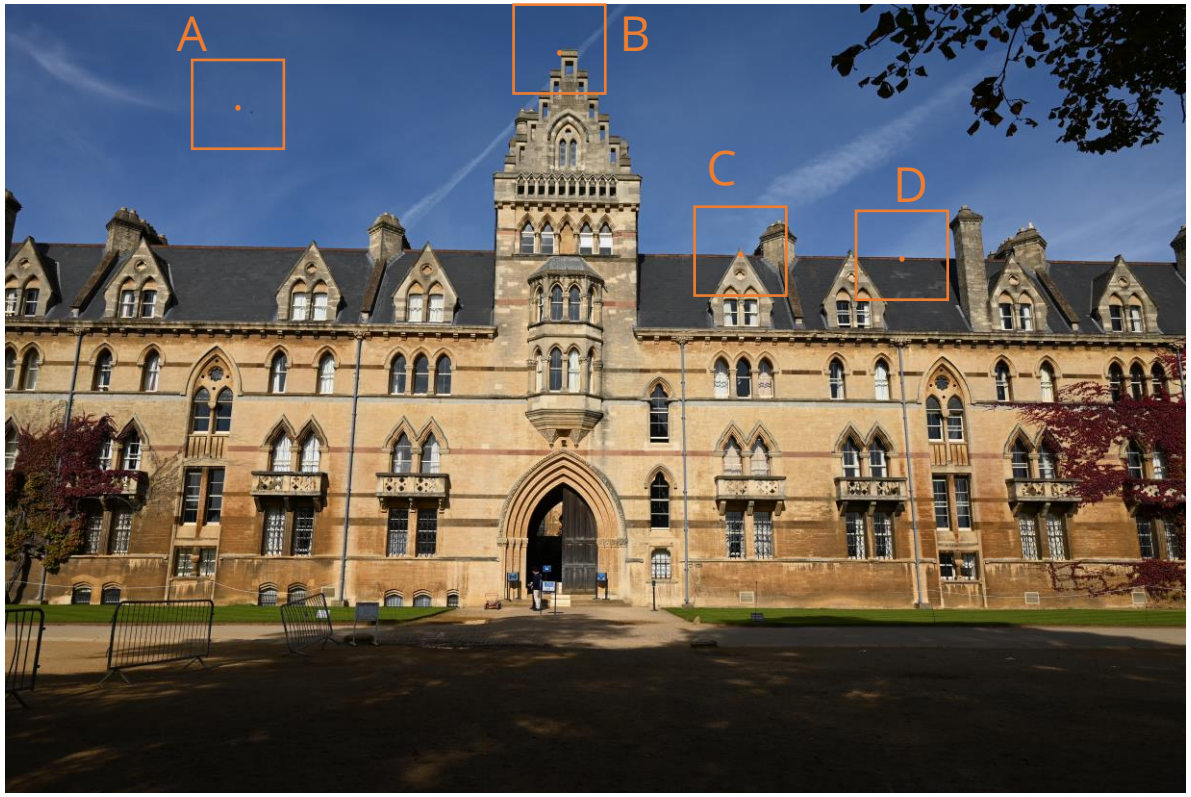


Image Transformations - again

- We know how to model rotation and translation:

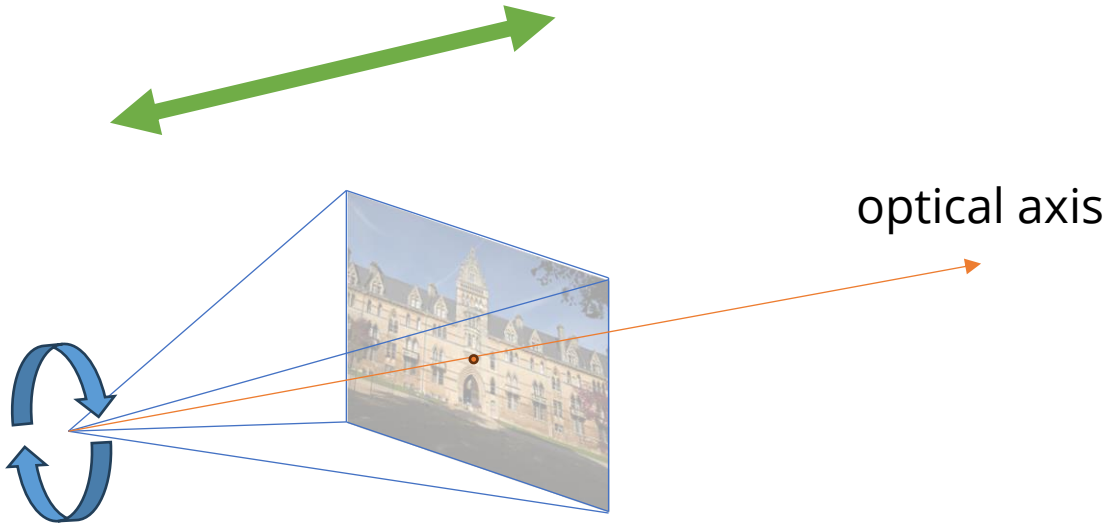
$$T(x,y) = A\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix}\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

This corresponds to

- Rotation around the *optical axis*
- Translation along the *optical axis*

Optical Axis

- Connects the centre of the camera with the centre of the image plane.
- Camera rotation around the axis rotates the image around its centre.
- Camera translation along the axis scales the image around its centre.



Camera Motion vs. Image Motion

- Relating how 3D camera motions change the image is important to model viewpoint changes.
- We know: rotation and translation around/along the optical axis maps to affine 2D transformations.
- What about other viewpoint changes?

Homography

- We will assume that everything we see lies on a single 3D plane.
- A circle on the 3D plane maps to an ellipse in the image.
- We will see the mathematical derivation in lecture 16.

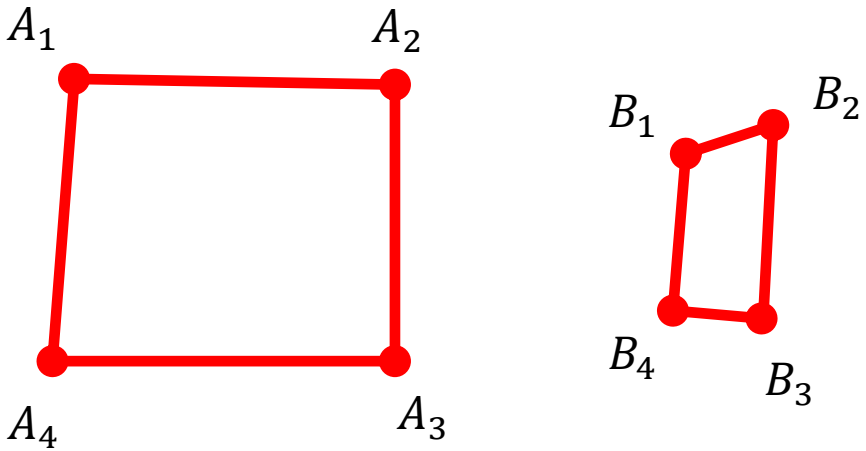
Image source

Homography

Homography

- We are looking for a transformation that maps 4 points (from a 3D plane) in one image to another image.
- $T(A_i) = B_i,\ \forall i \in \{1,2,3,4\}$
- All 8 points can be in arbitrary 2D positions.
- Degrees of freedom: 8

(2 equations for every 2D point pair)



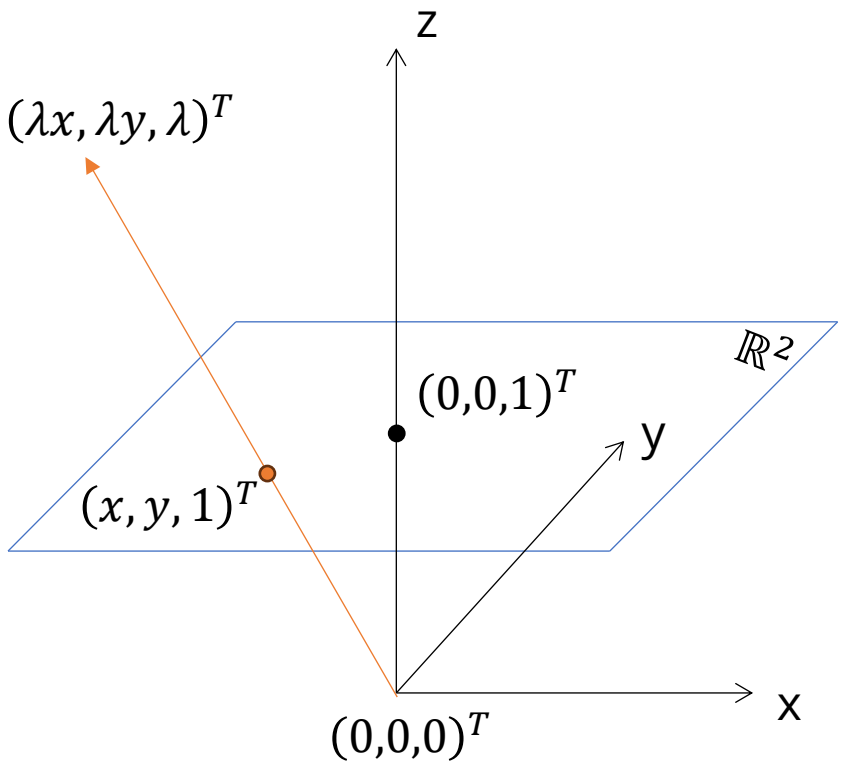
Homogeneous Coordinates II

Geometric interpretation:

- We define an equivalence class:

$$\left[\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}\right] = \left\{\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \lambda \in \mathbb{R} \ \wedge \lambda \neq 0\right\}$$

- This embeds $\mathbb{R}^2$ in $\mathbb{R}^3$.
- We identify the 2D point $(x, y)^{\mathrm{T}}$ with the class $[(x, y, 1)^{\mathrm{T}}]$.



Homogeneous Coordinates II

- Given a point in homogeneous coordinates $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ we can compute its 2D counterpart as $\begin{pmatrix} y/z \\ x/z \end{pmatrix}$ if $z \neq 0$.
- Points with $z = 0$ have a special meaning: they *lie at infinity* (later).

- Now we can describe homographies.

Homography

A homography H is a transformation of 2D homogeneous coordinates p of the form

$$T(p) = H\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

- Since homogeneous coordinates are equivalence classes of points, this matrix has only 8 degrees of freedom:

$$[Hp] = [\lambda Hp]$$

- We often rescale the matrix by $\lambda = \frac{1}{h_{33}}$ such that the bottom right element is 1.

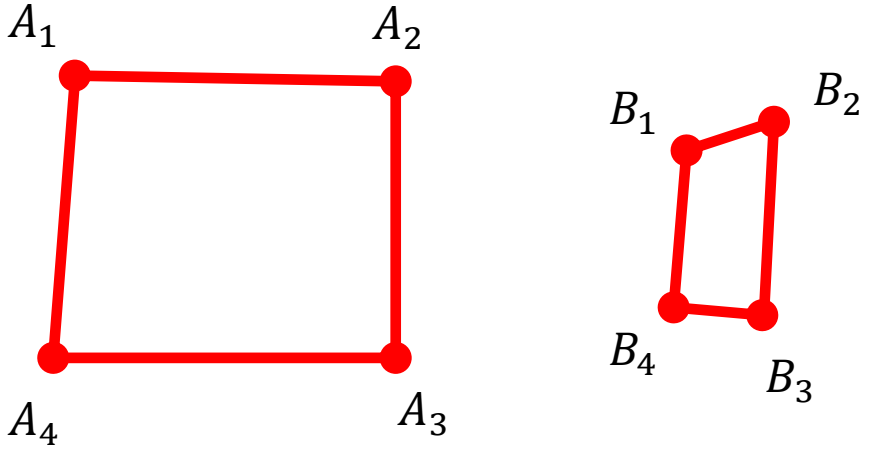
Homography

- When applying a homography, make sure renormalise points to treat them as 2D coordinates $\begin{pmatrix} y/z \\ x/z \end{pmatrix}$!

- A homography is uniquely defined by 4 point-correspondences if no 3 points lie on the same line.

- Naturally, also images can be transformed with a homography. (Lecture 02)

Estimating Homographies



$H[A_i] = [B_i], \qquad \forall i \in \{1,2,3,4\}$

- Two equations per correspondence

$$B_{x,i} = \frac{h_{11}A_{x,i} + h_{12}A_{y,i} + h_{13}}{h_{31}A_{x,i} + h_{32}A_{y,i} + h_{33}}, \qquad B_{y,i} = \frac{h_{21}A_{x,i} + h_{22}A_{y,i} + h_{23}}{h_{31}A_{x,i} + h_{32}A_{y,i} + h_{33}}$$

Multiplying by the denominator:

$$B_{x,i}\left(h_{31}A_{x,i} + h_{32}A_{y,i} + h_{33}\right) = h_{11}A_{x,i} + h_{12}A_{y,i} + h_{13}$$
$$B_{y,i}\left(h_{31}A_{x,i} + h_{32}A_{y,i} + h_{33}\right) = h_{21}A_{x,i} + h_{22}A_{y,i} + h_{23}$$

Estimating Homographies

Homogeneous linear system

$$\begin{pmatrix} -A_{x,1} & -A_{y,1} & -1 & 0 & 0 & 0 & B_{x,1}A_{x,1} & B_{x,1}A_{y,1} & B_{x,1} \\ 0 & 0 & 0 & -A_{x,1} & -A_{y,1} & -1 & B_{y,1}A_{x,1} & B_{y,1}A_{y,1} & B_{y,1} \\ -A_{x,2} & -A_{y,2} & -1 & 0 & 0 & 0 & B_{x,2}A_{x,2} & B_{x,2}A_{y,2} & B_{x,2} \\ & & & & \vdots & & & & \\ 0 & 0 & 0 & -A_{x,4} & -A_{y,4} & -1 & B_{y,4}A_{x,4} & B_{y,4}A_{y,4} & B_{y,4} \end{pmatrix} \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{pmatrix} = \mathbf{0}$$

- H is in the null-space of this matrix.
- Finding a non-trivial solution: solve for H using SVD.

Homography

Homography

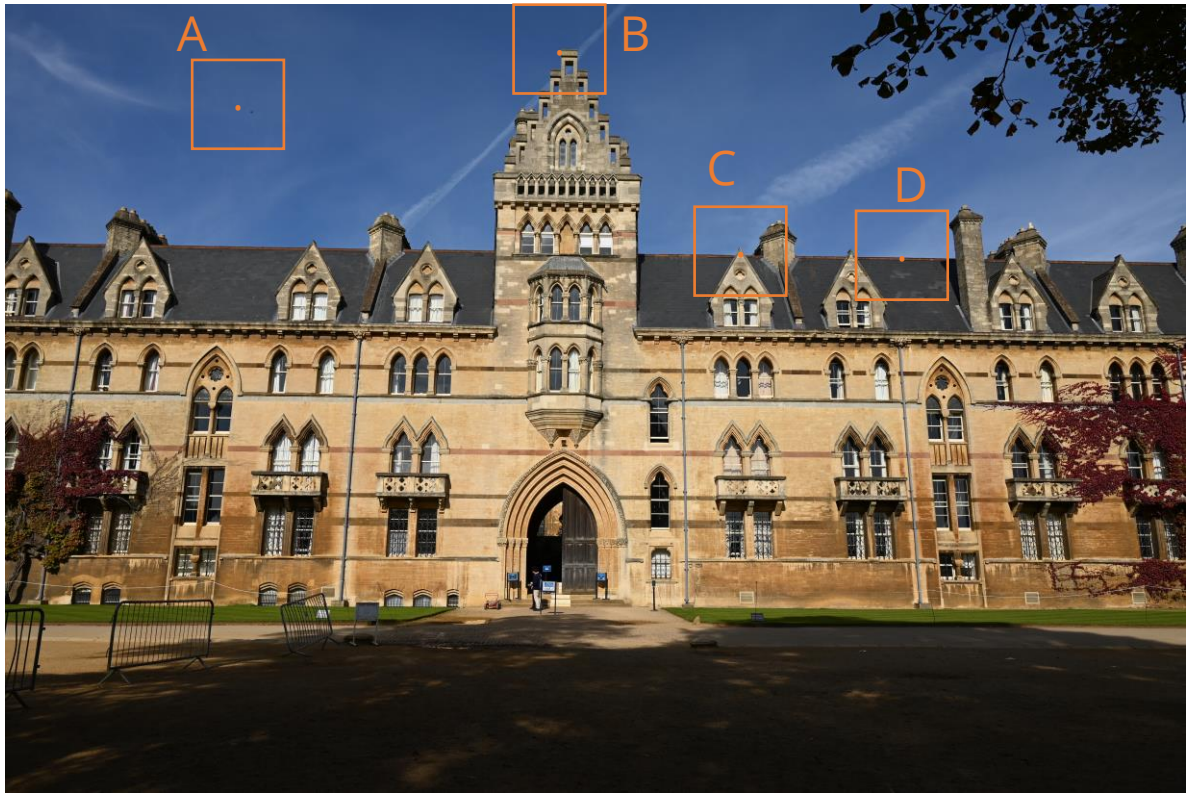
Homography

Keypoints

What makes a good keypoint?

Keypoints should stable under:

- Perspective changes (homographies).
- Contrast changes.
- Lighting changes.
- Other changes, as much as possible.



Keypoints

- Keypoints on blobs, corners, and high-contrast regions are the most stable.

- We will describe each keypoint through its local neighbourhood (patch).

- If we make the patch small, we can assume mostly simple geometric transformations of the neighbourhood (e.g. homographies)

Scale

Scale

- To arrive at a good local neighbourhood descriptor, we need to define the size of the neighbourhood.

- This defines the *scale* of a keypoint.

Detecting Scale

Scale Space

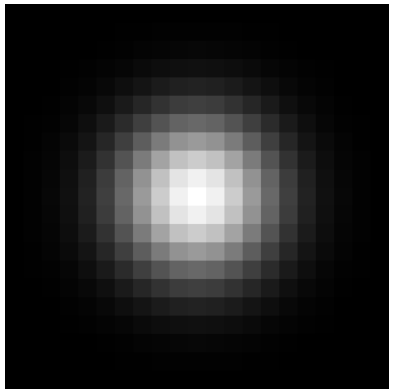
- We need to define a unique scale for a keypoint: the *characteristic scale.*

- This turns the problem of finding keypoints into a search across three parameters: location and scale (x, y, σ).

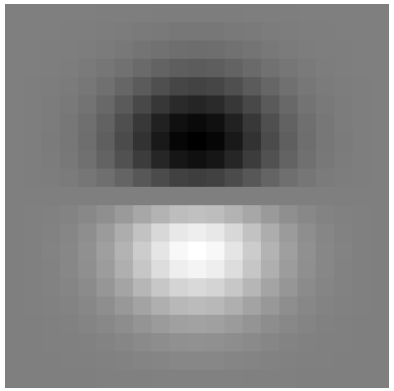
- We will find keypoints with scale by finding local minima in an energy function $\mathrm{E}(x, y, \sigma)$.

Laplacian of Gaussian

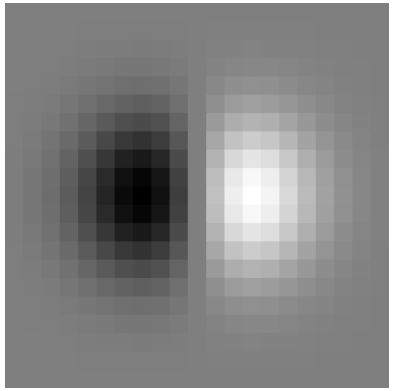
Gaussian g



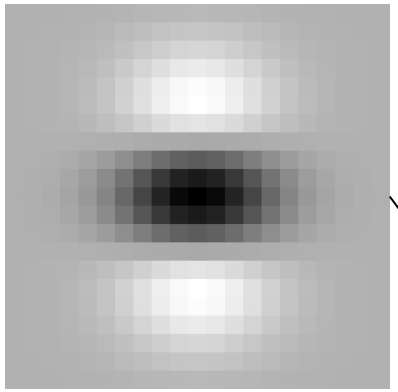
$\frac{\partial}{\partial y} g$



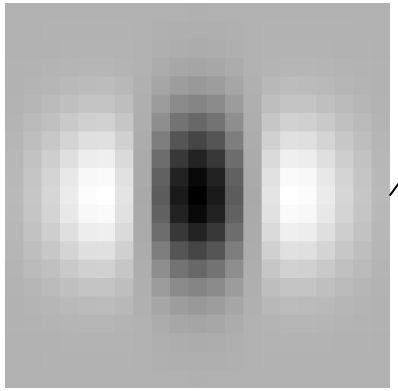
$\frac{\partial}{\partial x} g$

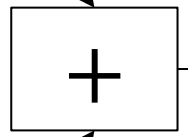


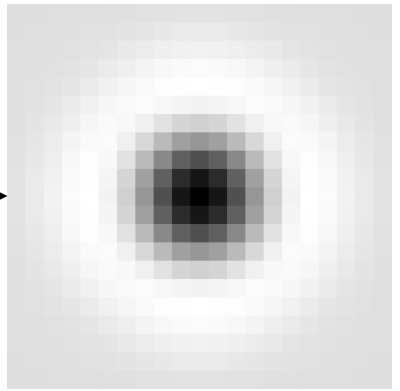
$\frac{\partial^2}{\partial^2 y} g$



$\frac{\partial^2}{\partial^2 x} g$





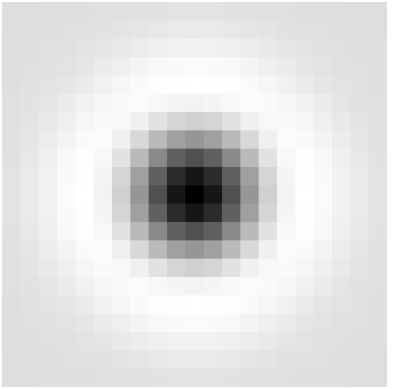


$$\frac{\partial^2}{\partial^2 x} g + \frac{\partial^2}{\partial^2 y} g$$

Source: J. Johnson and D. Fouhey

Laplacian of Gaussian

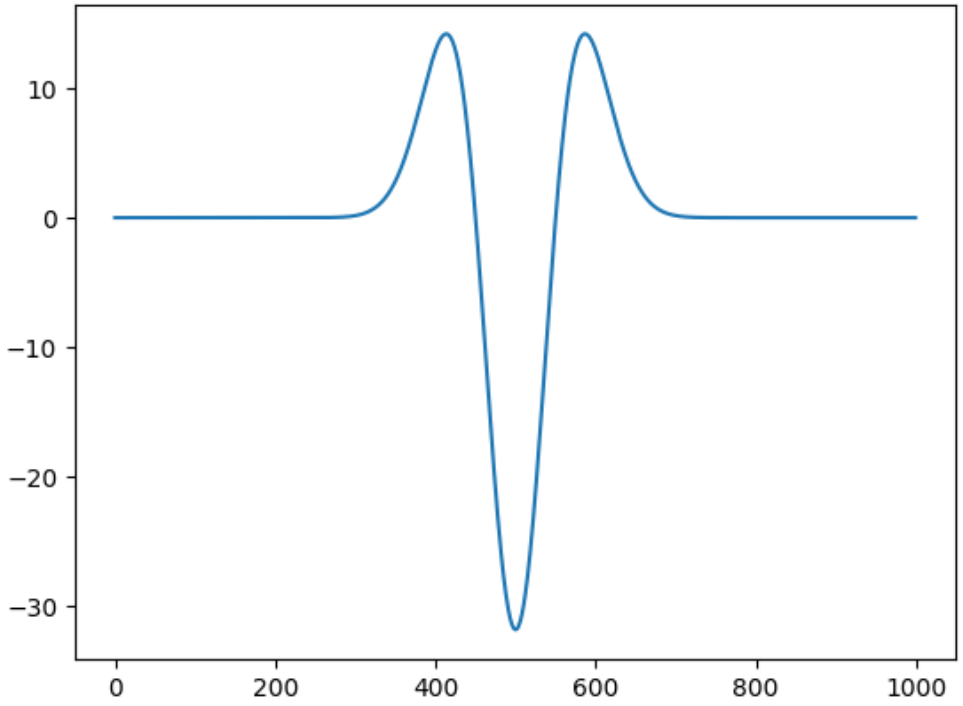
- This is a *blob detector*.
- It will have minima in space and scale when the filter “matches” a blob
- We will analyse why in 1D.



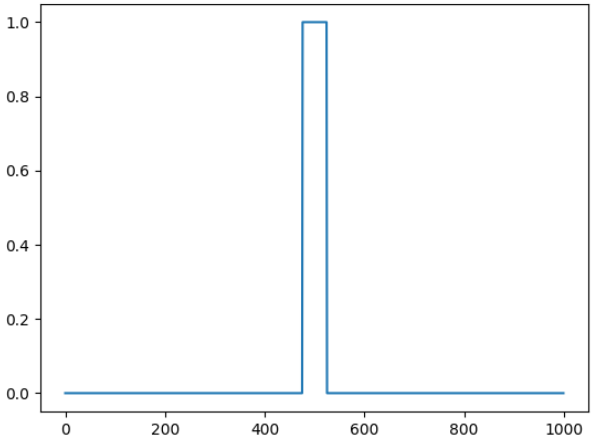
2D LoG filter

Laplacian of Gaussian 1D

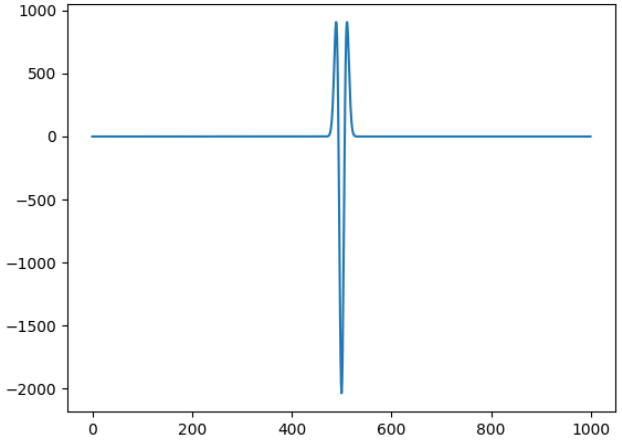
- Laplacian of Gaussian is the spatial 2nd order derivative of a Gaussian.
- $\mathrm{LoG}(x) = -\frac{1}{\pi\sigma^2}\left(1 - \frac{x^2}{\sigma^2}\right)e^{-\frac{x^2}{2\sigma^2}}$
- We will vary σ and convolve a simple signal with the filter.
- To make the output comparable, the response needs to be scaled by σ.



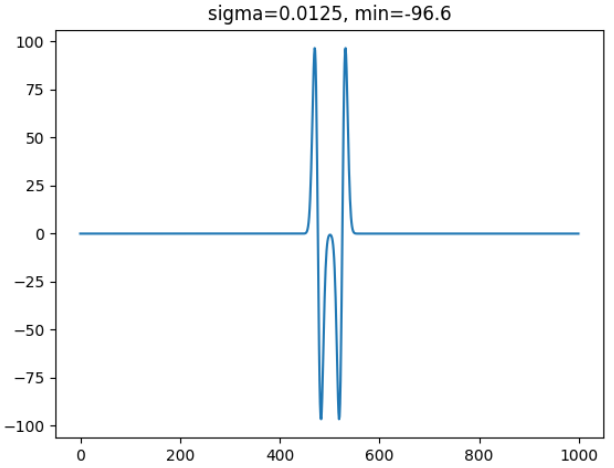
Laplacian of Gaussian 1D



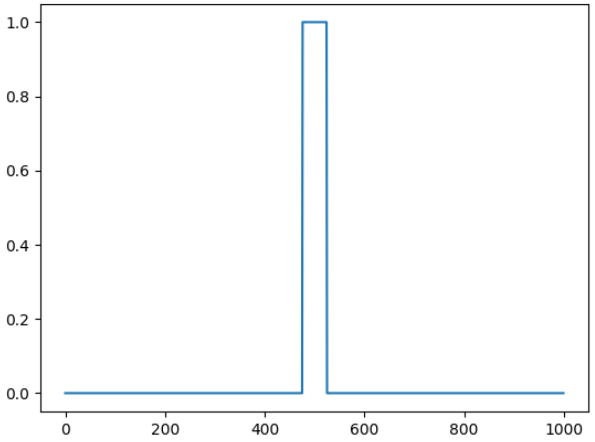
*



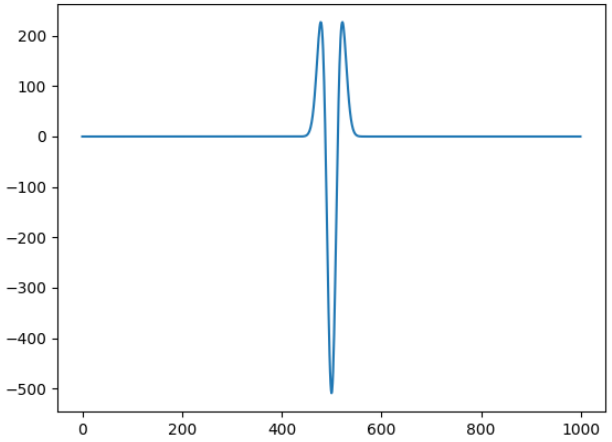
=



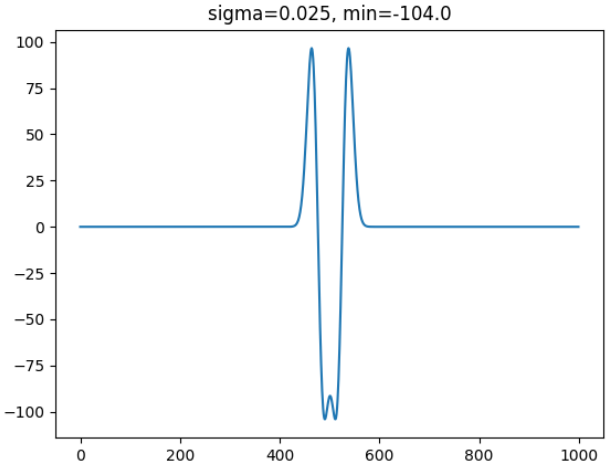
Laplacian of Gaussian 1D



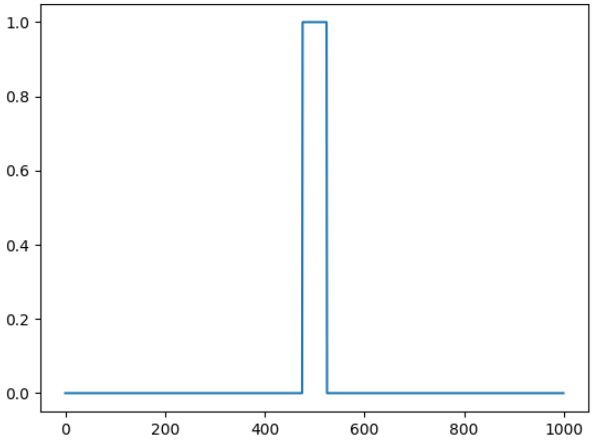
*



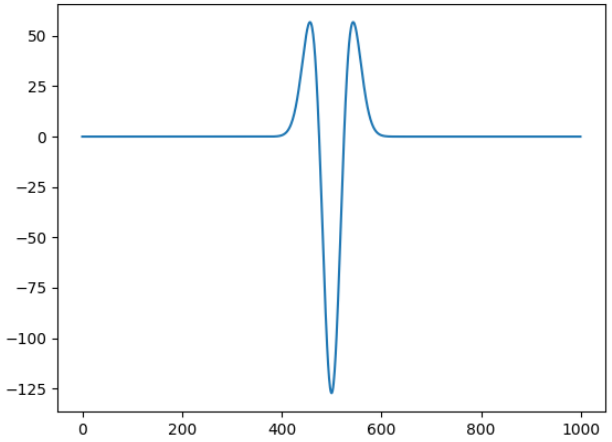
=



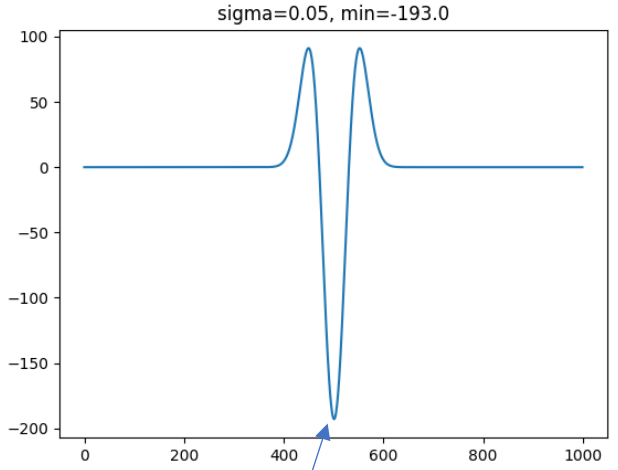
Laplacian of Gaussian 1D



*

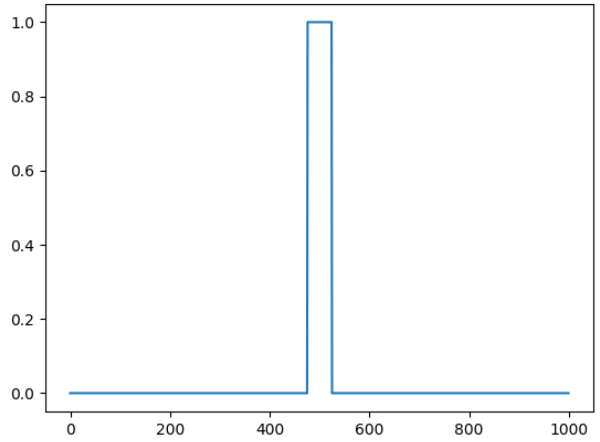


=

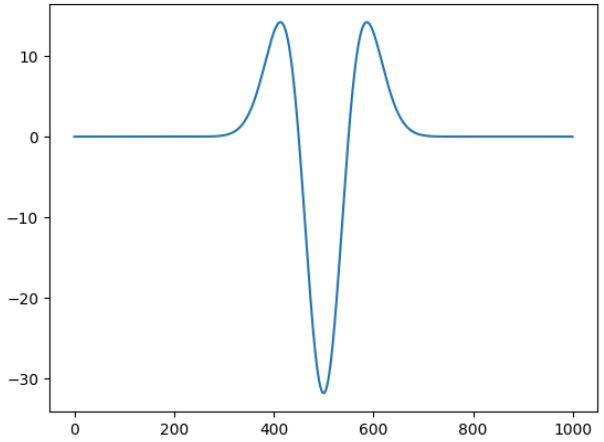


minimum over all scales

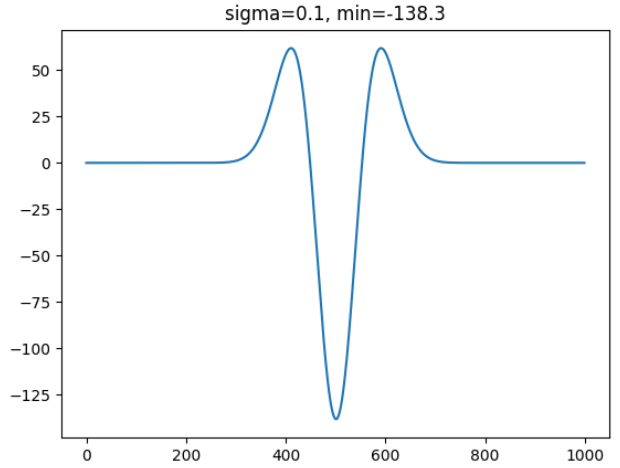
Laplacian of Gaussian 1D



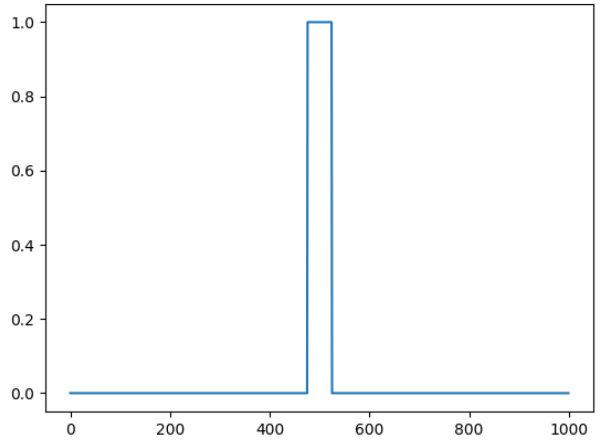
*



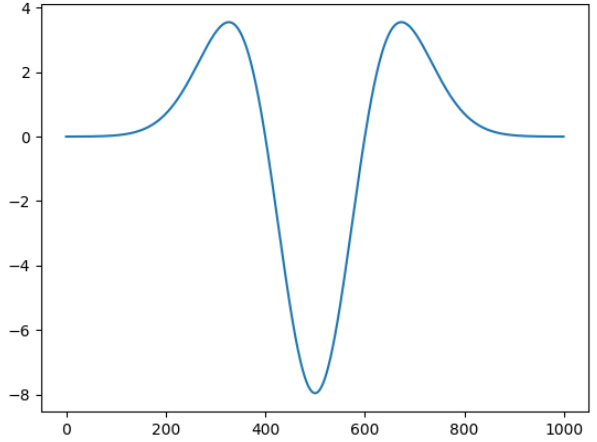
=



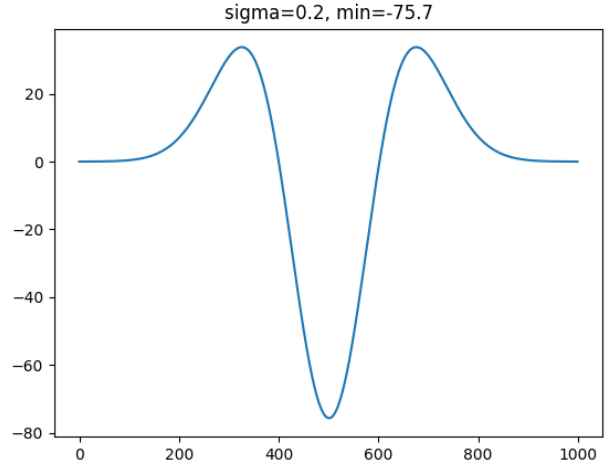
Laplacian of Gaussian 1D



*



=



2D Multiscale Blob Detection

- Convolve image with Laplacian of Gaussian filter at several values of σ.

- Find maxima of *squared* Laplacian response in space *and* across scales.

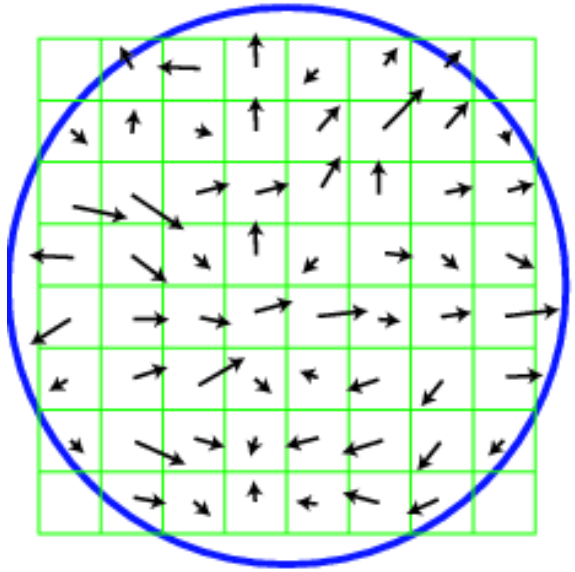
- In practice, this means looking at a 3x3x3 neighborhood in the (x, y, σ) space. If the center is larger than its 26 neighbours: you found a blob!

Beyond Blob Detection

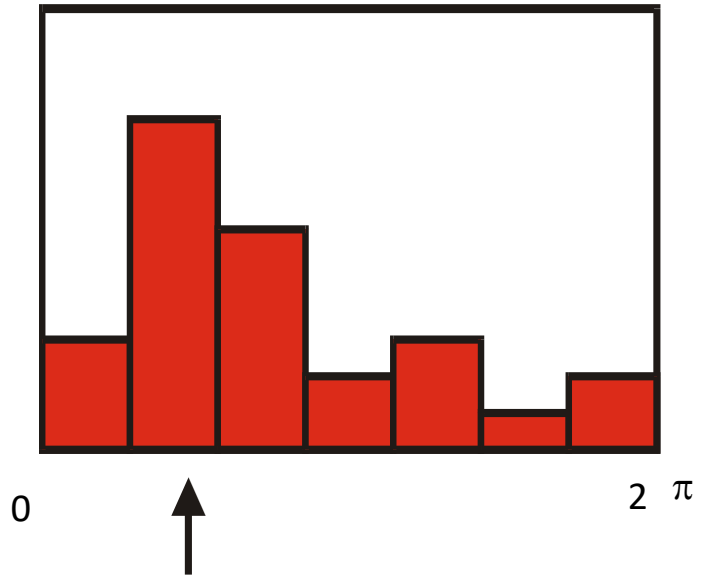
- Now we can detect blobs and their scale.
- Next, we want to find the orientation of a keypoint.
- This allows us to describe a keypoint invariant to rotation.

Keypoint Orientation

- Compute the angles of the local edges around the keypoint. (Sobel)
- Discretise into 45 deg. increments.
- Pick the most common direction as the overall orientation.



Edge directions in the range of the keypoint



Discretise angles and chose most common

Adapted from S. Lazebnik

SIFT Descriptor

- Compute a descriptor for a keypoint.
- A descriptor is a vector that characterises the local information around the keypoint so that it can be found in other images.
- Idea: use edge directions again (invariant to brightness changes, equivariant to rotation).
- Compute and store local edge histograms.

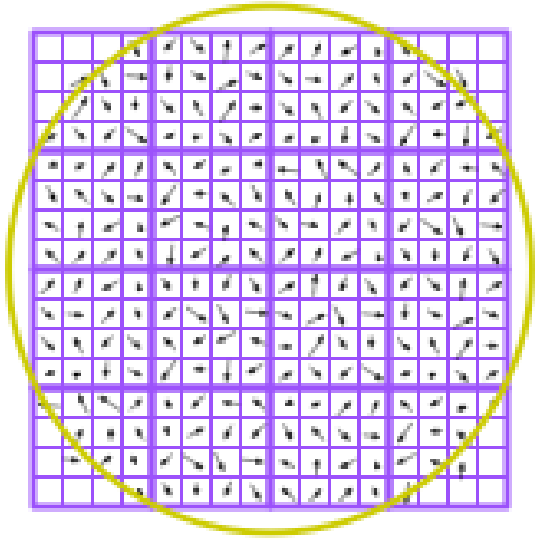
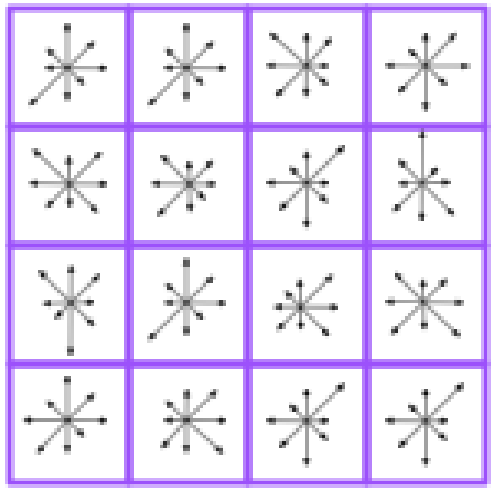


Image gradients



Keypoint descriptor

SIFT Descriptor

- Compute edge orientations and global orientation.
- Rotate all edges so that the global orientation is “up”.
- Split the local area around the keypoint into 4x4=16 regions.
- Compute edge histograms (8 directions) for each region.
- Concatenate histograms: descriptor 128 dimensional vector.

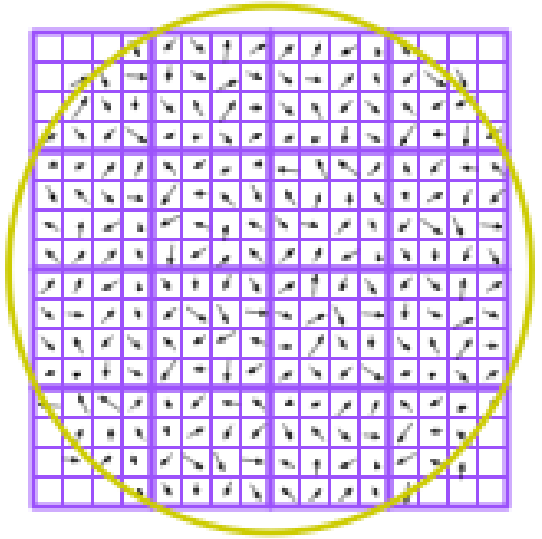
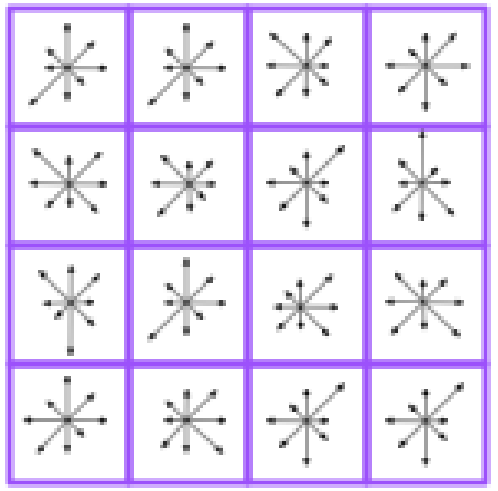


Image gradients



Keypoint descriptor

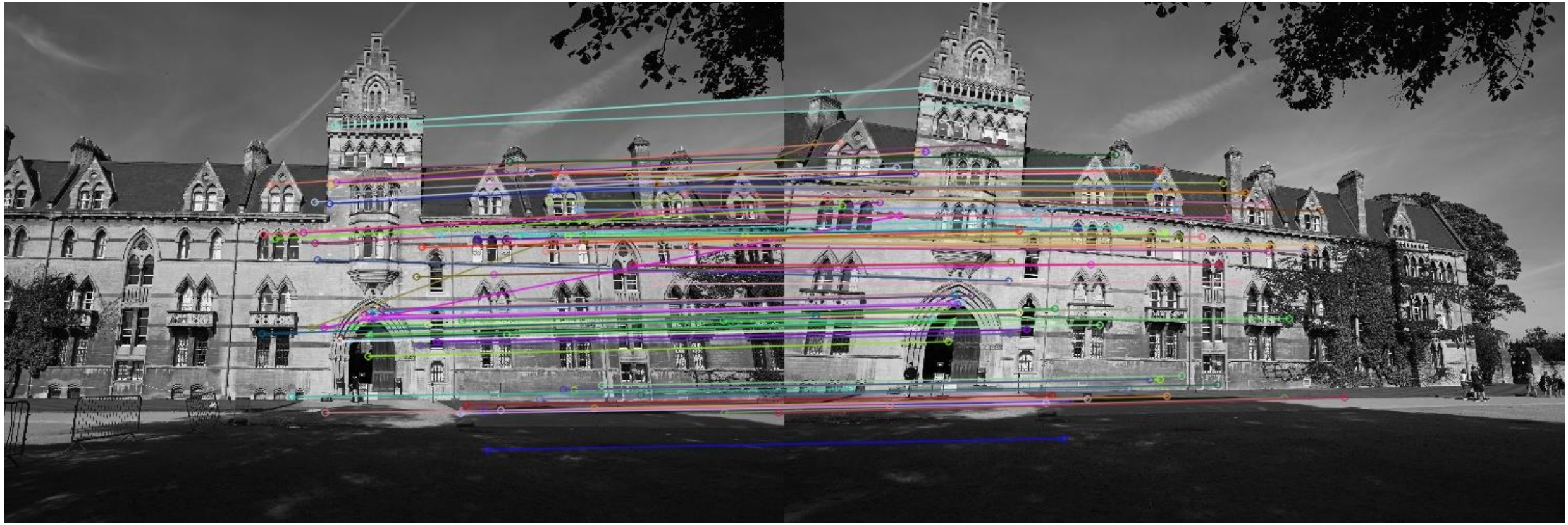
Feature Matching

- Many different strategies.
- Brute force matching:
  - For each descriptor in the first image, return the one with the lowest distance (e.g. Euclidean distance).
  - Sort all matches by their distance and take the top N.

- Later: better strategies such as RANSAC.
  - Idea: check if matches are consistent with a transformation, e.g. homography. This is geometric verification.

Correspondences

Point correspondences estimated by a classic algorithm: SIFT



Local to Global Matching

- Given an image, we now want to find similar other images.
- We can use feature matching, but this means comparing all image features to all database features: very slow!

10^{10} images * 10^3 kpts * 10^2 dims * 10^3 query kpts = 10^{18}ops!

Fastest super-computer: 1000 Peta-FLOPS (=1 sec for every image search)

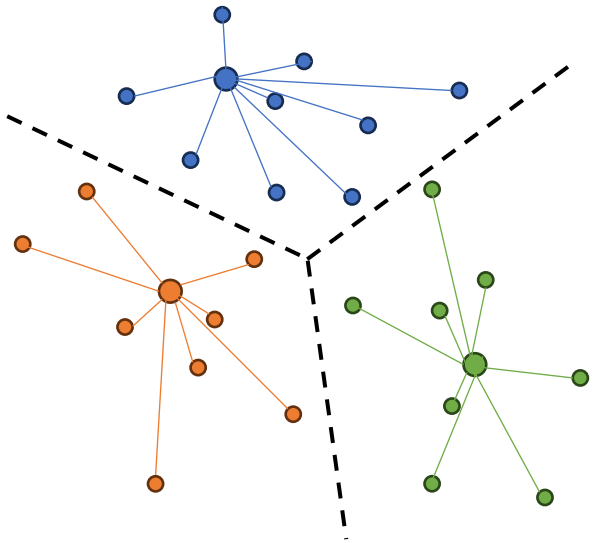
- Idea: compute a single global descriptor for an image.
- The global descriptor should capture the local information form the keypoints.

K-means Clustering

- Divide the space into K clusters $\mathcal{S}$.
- Minimise the sum of squared distances of points in a cluster.

$$\arg\min_{\mathcal{S}} \sum_{i=1}^{K} \frac{1}{|S_i|} \sum_{x,y \in S_i} \|x - y\|^2$$

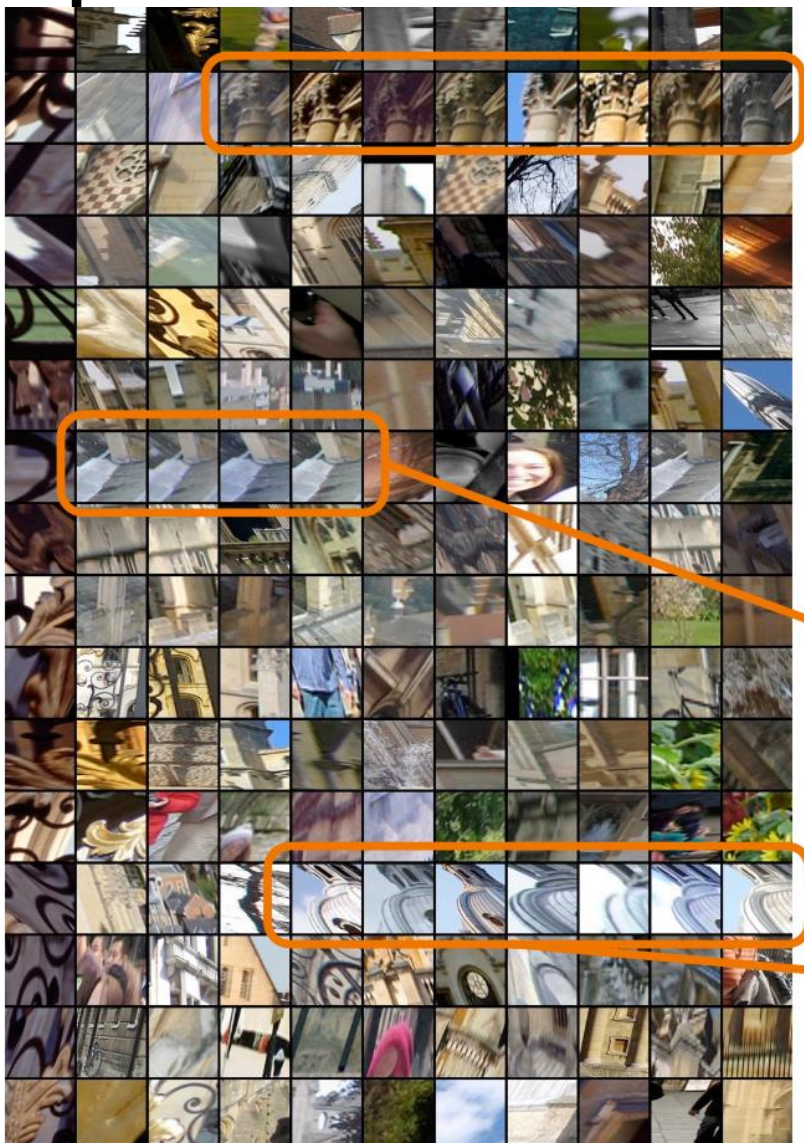
- We can treat the cluster assignment as a category for the point.



Visual Words

- Compute SIFT features on a large image dataset.
- Compute K-means clustering and assign each feature to one of K “classes”.
- An image can now be described with a histogram of feature classes it contains.
- This is a reduction in descriptor size per image from #features*#dimensions to K.

Example



Visual word examples. Each row is an equivalence class of patches mapped to the same cluster by K-means.

Slide from A Vedaldi

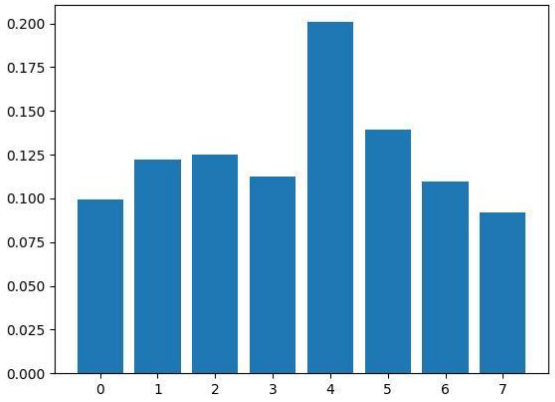
Bag of Visual Words

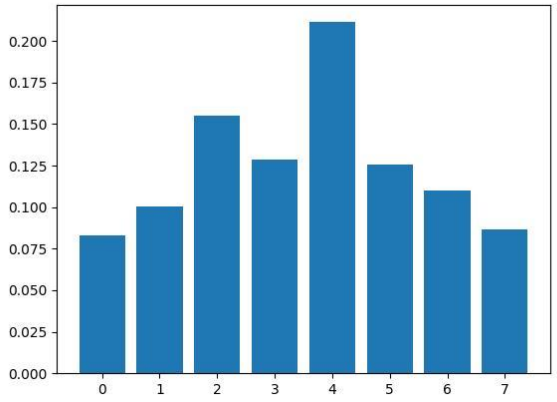
- When computing the histogram, it does not matter where each feature comes from in the image.
- This is similar to classic text-based retrieval systems: count how often each word appears in a document. Documents on similar topics have similar statistics.
- With large K (and a large dataset) the image descriptor is usually *sparse* (i.e. most entries are 0)
- Sparse vectors can be stored and compared efficiently: only store/compare non-zero elements.

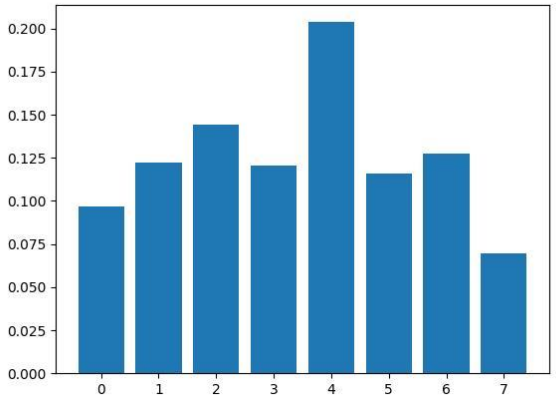
Bag of Visual Words

Bag of Visual Words







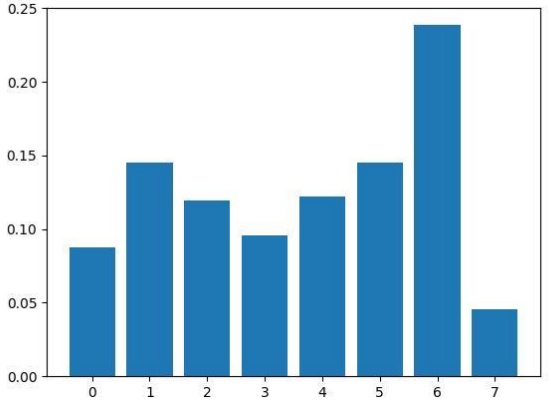


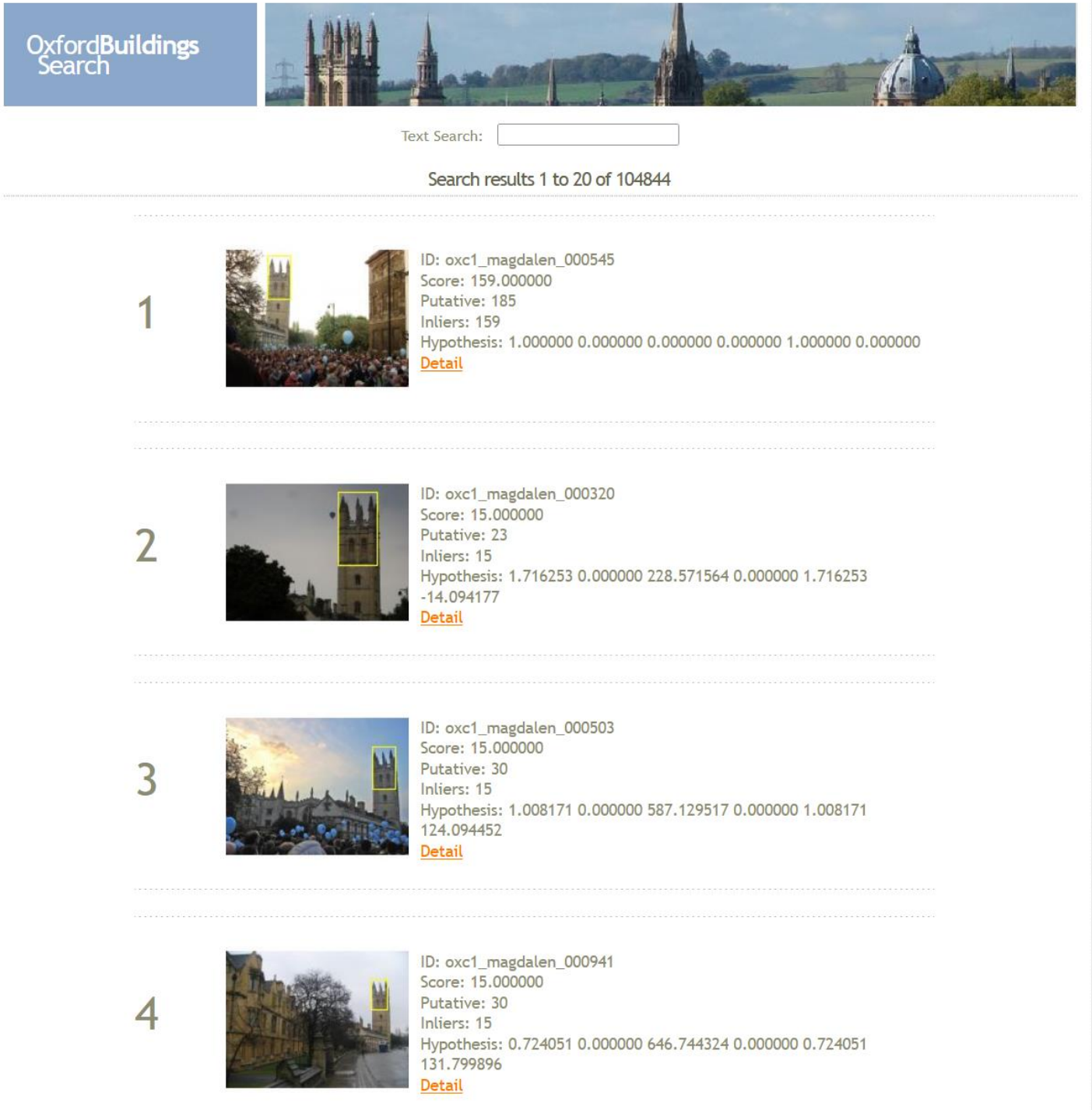
Image Retrieval

1. Given a query image, compute keypoints and descriptors.
2. Find keypoint labels from pre-computed (k-)means.
3. Compute bag of visual words for the query.
4. Compare BoW query to all database BoWs and retrieve top M.
5. Optional: perform additional (slower) verification for the top M results.

Image retrieval

- Can also be used to search for image regions
- Idea: compute BoW within a region.

- Demo



Philbin, J. , Chum, O. , Isard, M. , Sivic, J. and Zisserman, A.

Object retrieval with large vocabularies and fast spatial matching

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2007)

Chum, O. , Philbin, J. , Sivic, J. , Isard, M. and Zisserman, A.

Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval

Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, Brazil (2007)