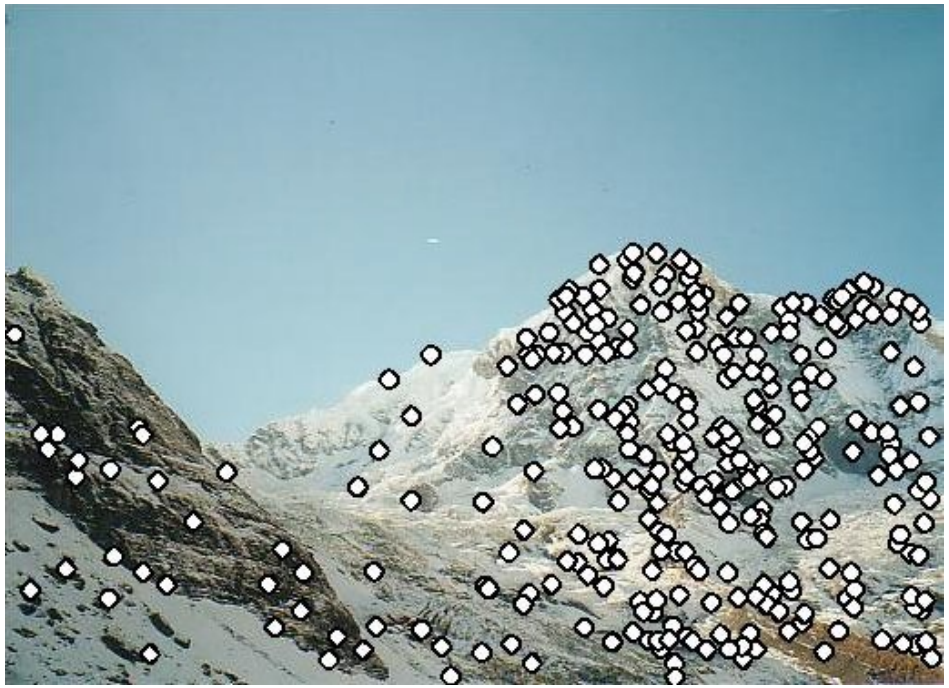
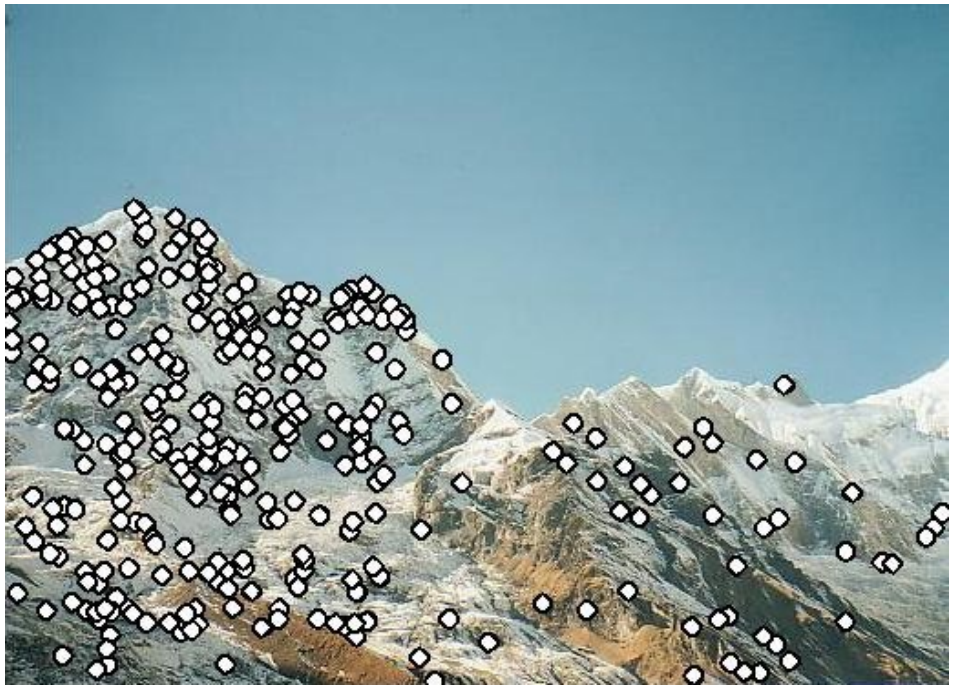
Feature Description I

Dr Wenjia Bai

Department of Computing & Brain Sciences





- Now we know how to detect interest points on these two images.
- How do we know which point corresponds to which?
- We need to describe the feature for each point so that they can be compared and matched.

In this lecture

- Given a point on an image, how do we describe the local feature or content around this point?
- After feature description, how do we match interest points between two images?

Feature description

- Simple descriptors
  - Pixel intensity
  - Patch intensities
  - Gradient orientation
  - Histogram
- SIFT descriptor that is robust to rotation and scaling

Pixel intensity

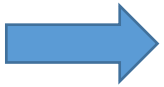
- Simply using the intensity of a single pixel.
- Problems
  - Sensitive to absolute intensity value.
    - The intensity of the same point will change under different illuminations.
  - Not very discriminative.
    - The grayscale intensity ranges from 0 to 255. There are thousands of pixels with exactly the same intensity.
    - A single pixel does not represent any local content (edge, blob, peak of the mountain etc.).

199	192	158	111	110	123	130	130
189	149	108	111	113	120	126	125
130	100	98	108	113	113	114	120
85	100	96	**104**	108	107	101	94
85	95	98	96	100	103	100	96
79	94	87	77	69	70	87	84
77	80	72	71	60	52	59	64
68	67	63	58	53	51	54	52

Patch intensities

- Better than a single pixel.
  - Represent the local pattern.
  - Perform well if the images are of similar intensities and roughly aligned [1,2].
- Problems
  - Sensitive to absolute intensity value
  - Not rotation-invariant

1	2	3
4	5	6
7	8	9



[1, 2, 3, 4, 5, 6, 7, 8, 9]

Patch representation

2	4	6
8	10	12
14	16	18

[2, 4, 6, 8, 10, 12, 14, 16, 18]

Problem: if intensity changes

7	4	1
8	5	2
9	6	3

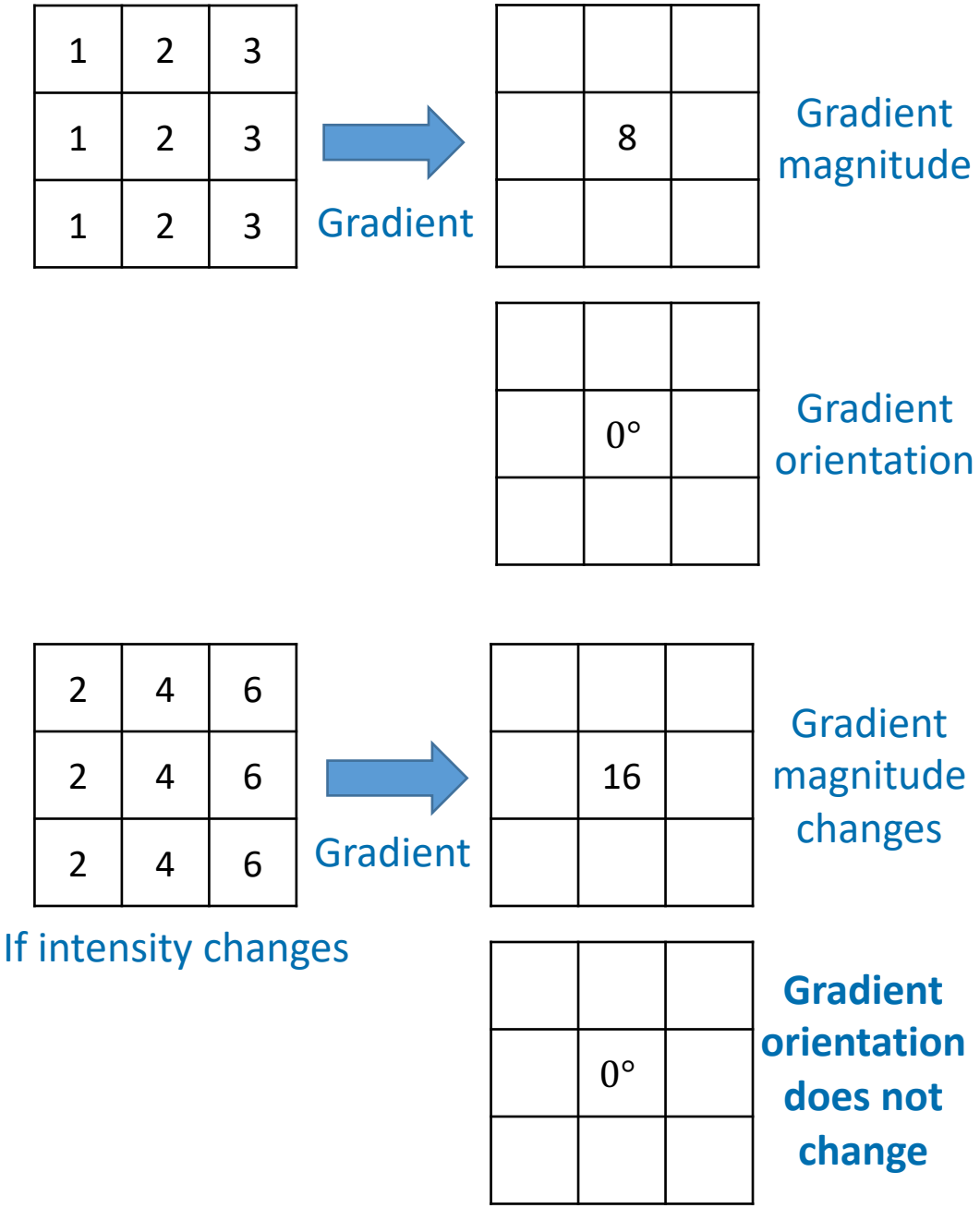
[7, 4, 1, 8, 5, 2, 9, 6, 3]

Problem: if being rotated

[1] C. Barnes et al. PatchMatch: A randomized correspondence algorithm for structural image editing, SIGGRAPH 2009.
[2] P. Coupe et al. Patch-based segmentation using expert priors. NeuroImage, 2011.

Gradient orientation

- Although gradient magnitude is sensitive to intensity changes, its orientation is robust.
- Problem
  - Not rotation-invariant.



Histogram

- The intensity histogram of a patch
  - Robust to rotation
  - Robust to scaling
- Problem
  - Sensitive to intensity changes

1	2	3
4	5	6
7	8	9

Histogram representation

7	4	1
8	5	2
9	6	3

If being rotated

1	1	2	2	3	3
1	1	2	2	3	3
4	4	5	5	6	6
4	4	5	5	6	6
7	7	8	8	9	9
7	7	8	8	9	9

If being scaled

Histogram

- A rotated or scaled image patch will result in the same or similar intensity histogram.

Thoughts

- Gradient orientation
  - Robust to change of absolute intensity values
- Histogram
  - Robust to rotation and scaling
- Can we combine the advantages of the two?
  - Yes.
  - It is used in the scale-invariant feature transform (SIFT) algorithm.

Scale-invariant feature transform (SIFT)

- SIFT is an algorithm for detecting and describing local features in images.
- SIFT transforms an image into a large set of interest points, each of which is described by a feature vector that is invariant to translation, scaling and rotation.

Scale-invariant feature transform (SIFT)

- SIFT consists of the following steps:
  1. Detection of scale-space extrema
  2. Keypoint localization
  3. Orientation assignment
  4. Keypoint descriptor

Detection

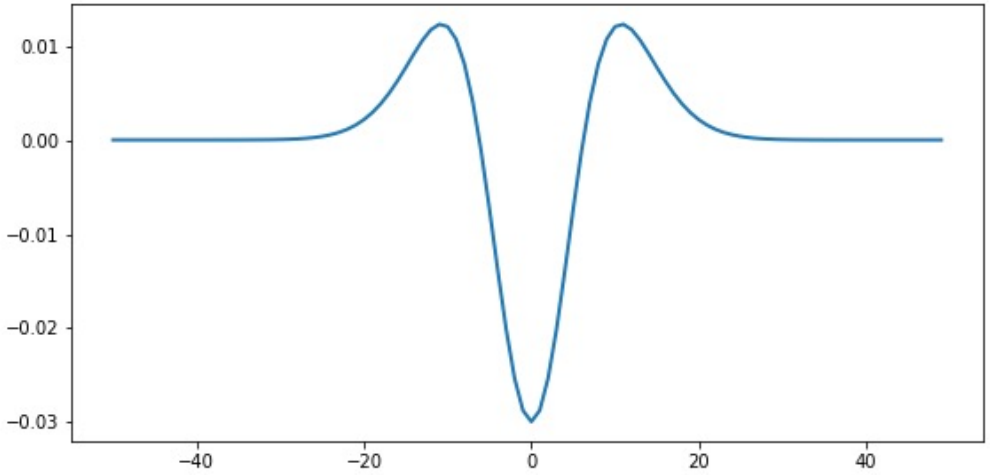
Description

Detection of scale-space extrema

- The first step is to search across scales and pixel locations, looking for interest points.
- The difference of Gaussian filter is used to identity potential interest points.

$$DoG(x, y, \sigma) = I * G(k\sigma) - I * G(\sigma)$$

- It is similar to Laplacian of Gaussian or a flipped Mexican hat.



Difference of Gaussians (DoG)

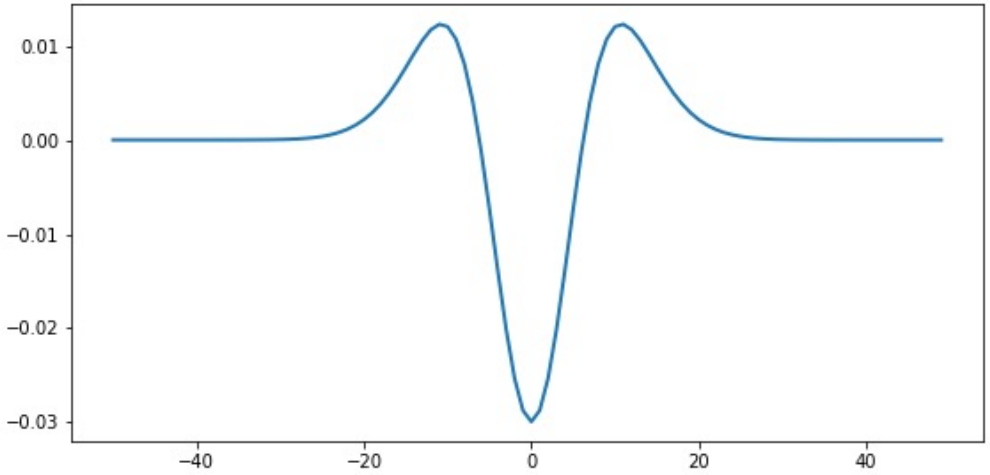
Mexican hat

Detection of scale-space extrema

- The first step is to search across scales and pixel locations, looking for interest points.
- The difference of Gaussian filter is used to identity potential interest points.

$$DoG(x, y, \sigma) = I * G(k\sigma) - I * G(\sigma)$$

- It is similar to Laplacian of Gaussian or a flipped Mexican hat.

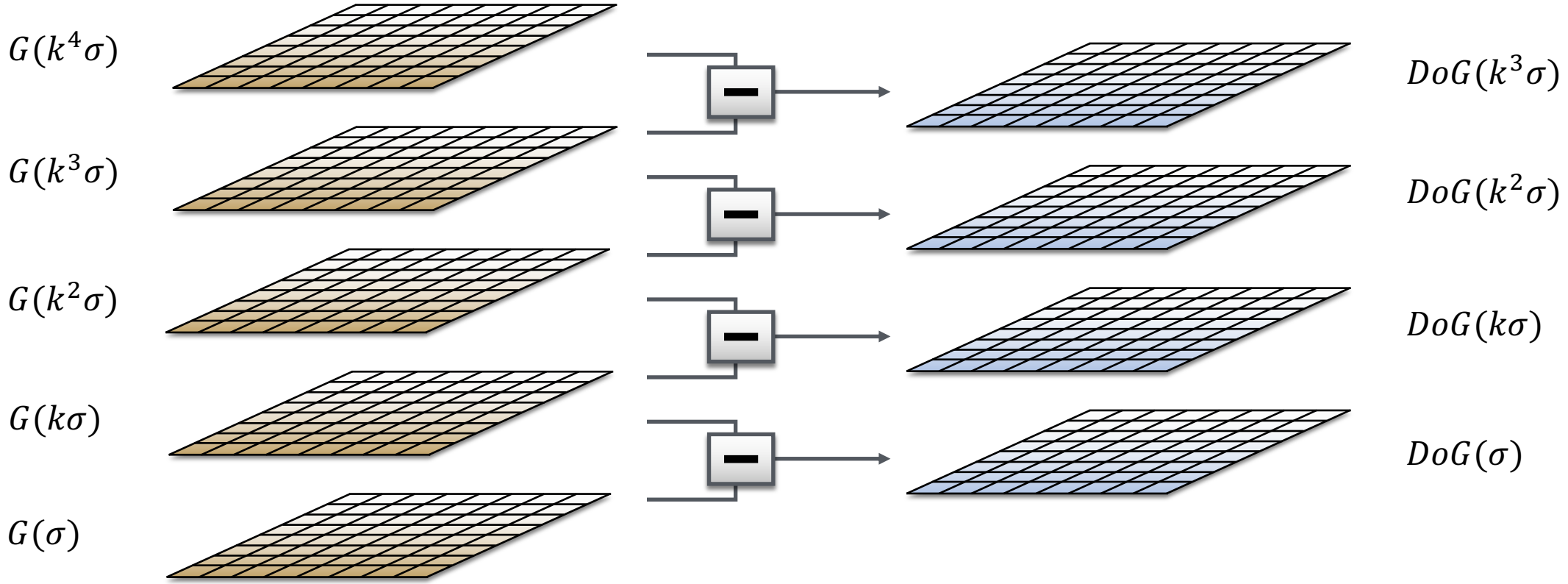


Difference of Gaussians (DoG)

Mexican hat

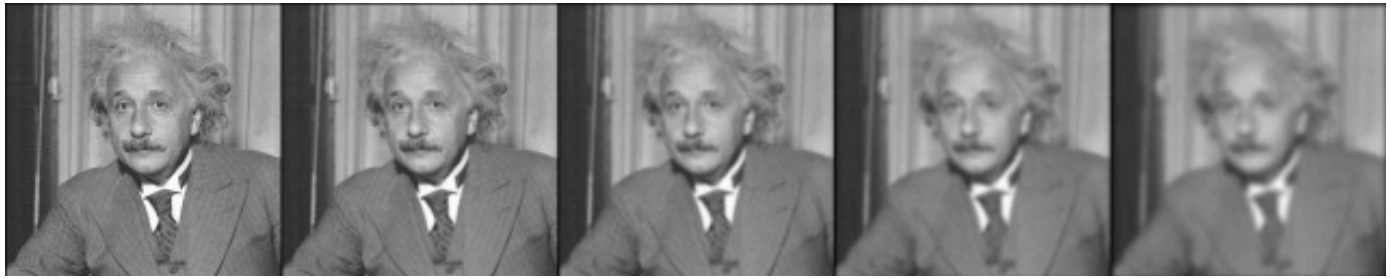
Detection of scale-space extrema

- We calculate the difference of Gaussian for different scales σ, from fine to coarse.
  - For example, $\sigma, \sqrt{2}\sigma, 2\sigma\ , 2\sqrt{2}\sigma\ , 4\sigma, \cdots$
- It is implemented as first calculating a stack of Gaussian smoothed images, then calculating their differences.



Gaussian

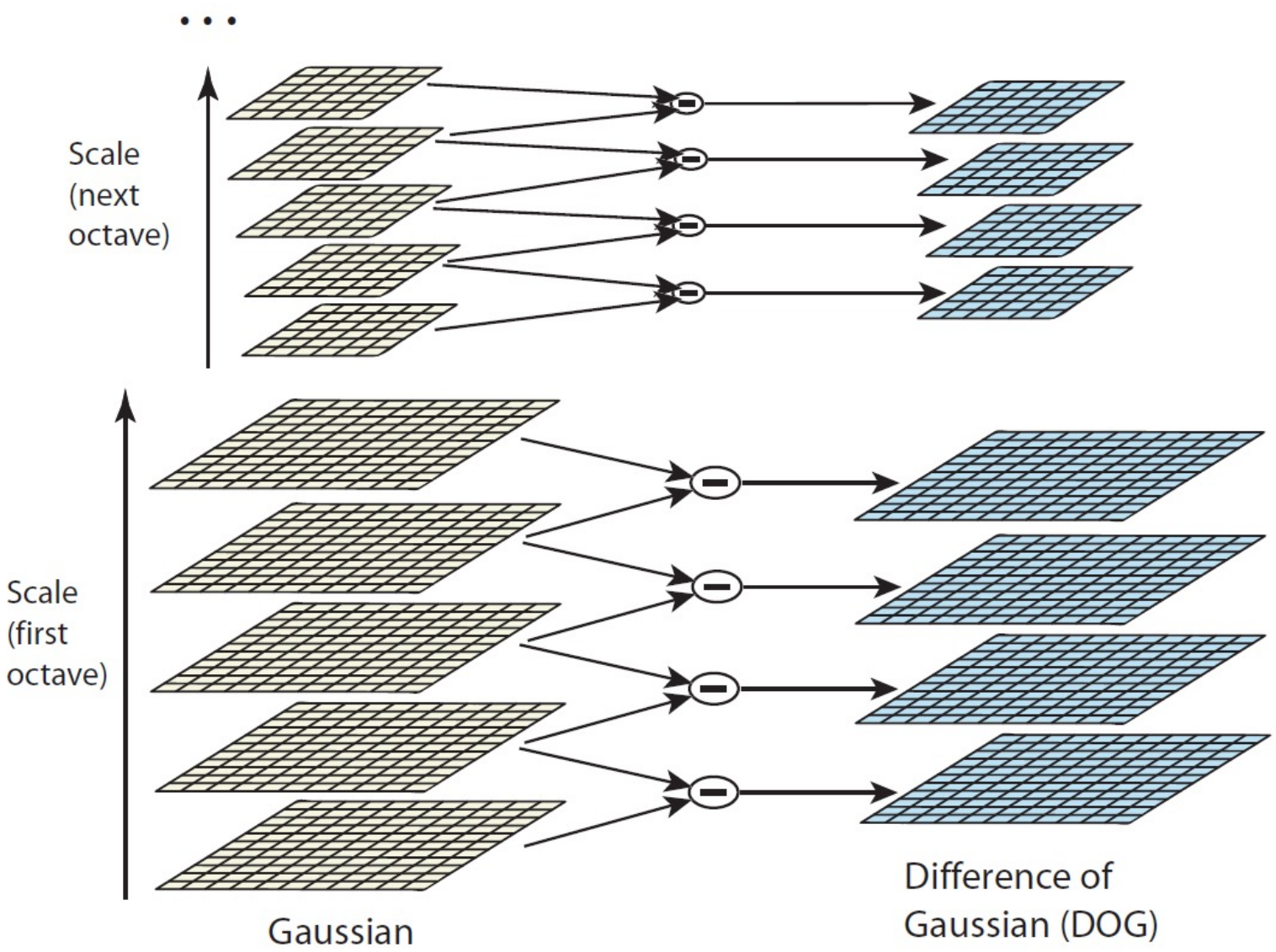
Difference of Gaussians (DoG)



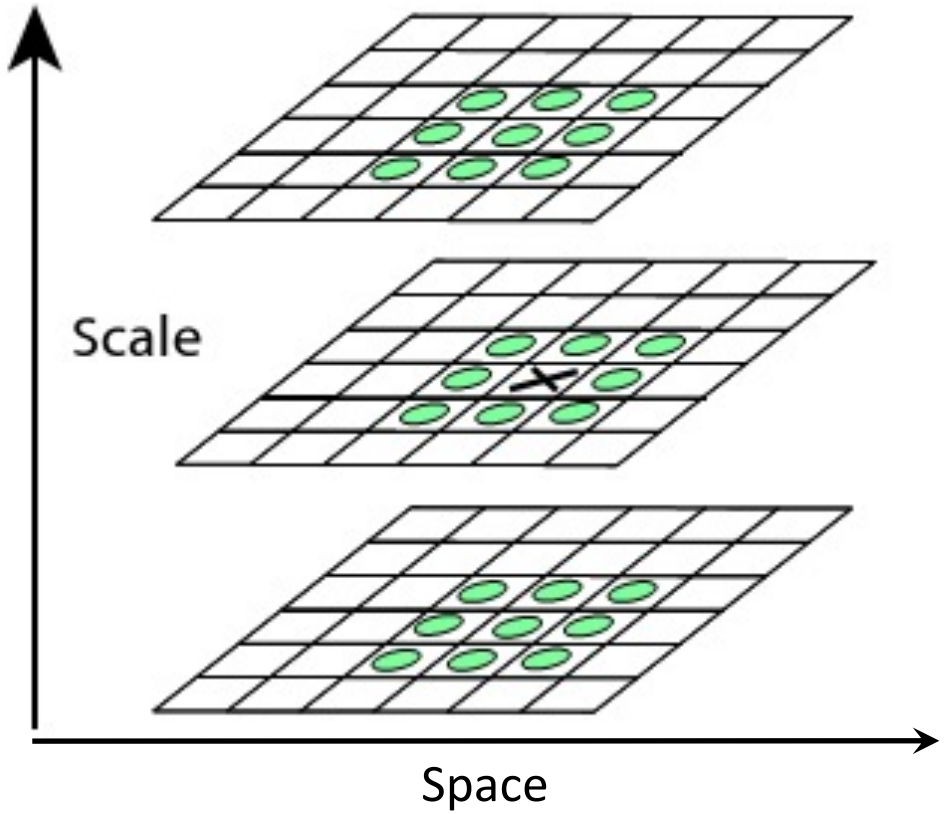
$G(\sigma)$ $G(k\sigma)$ $G(k^2\sigma)$ $G(k^3\sigma)$ $G(k^4\sigma)$

$DoG(\sigma)$ $DoG(k\sigma)$ $DoG(k^2\sigma)$ $DoG(k^3\sigma)$

Detection of scale-space extrema



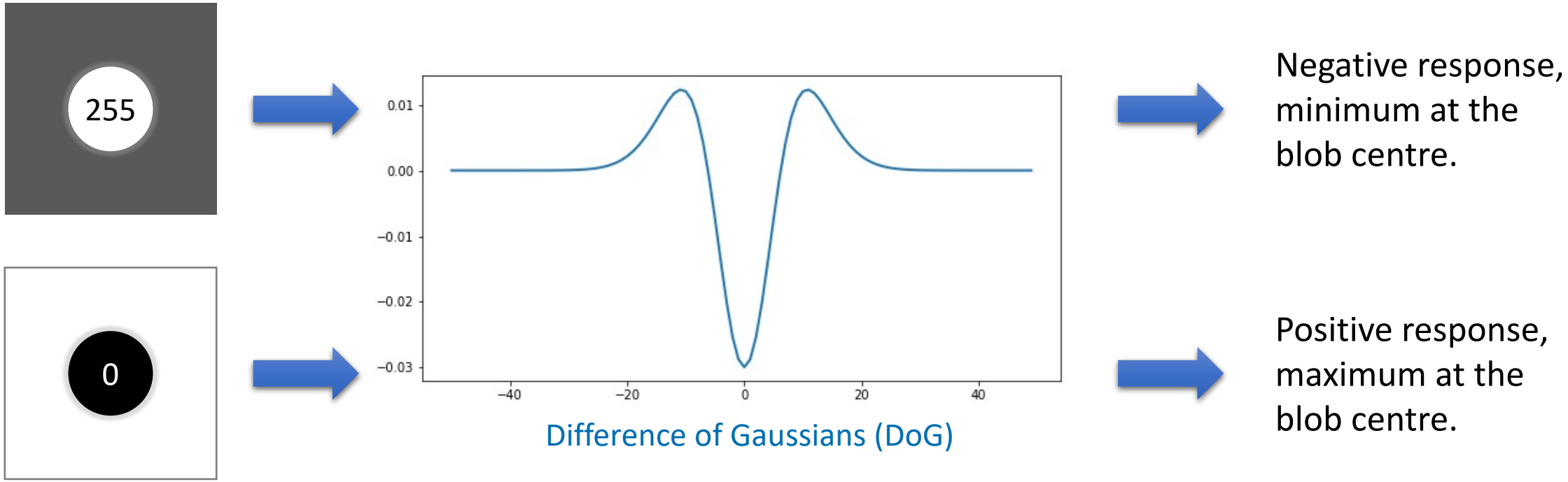
Detection of scale-space extrema



X is detected as an interest point if it is a local extremum both along scale dimension (most appropriate scale) and across space. A threshold may also be applied.

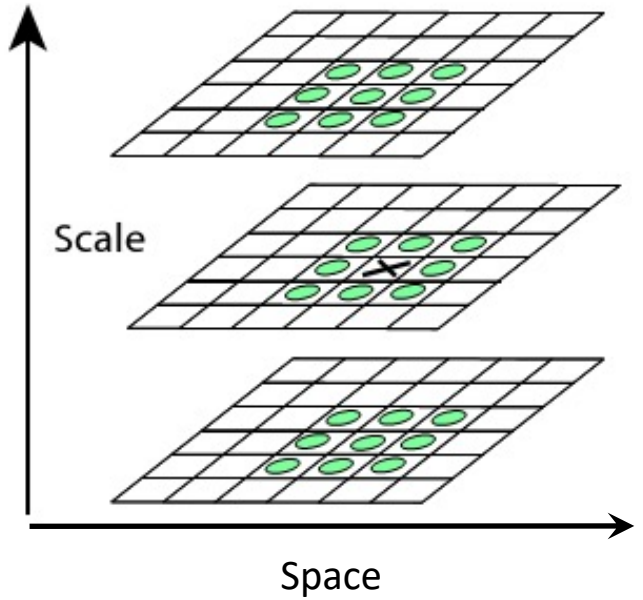
Detection of scale-space extrema

- We use the term extrema here, which means both minima and maxima.



Keypoint localisation

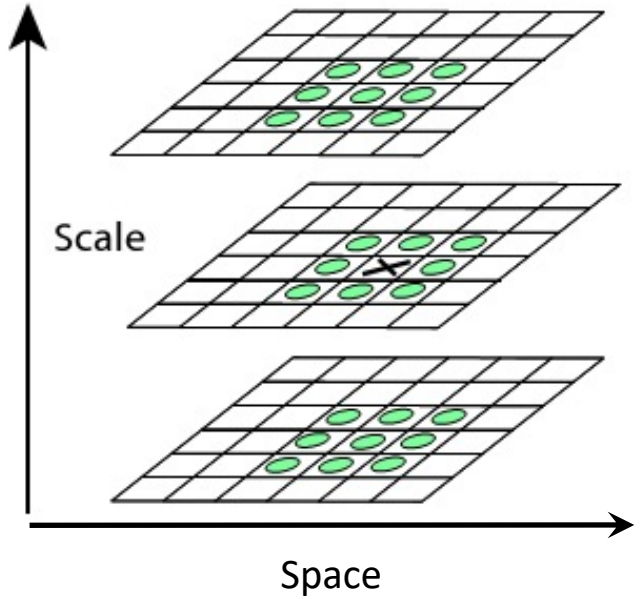
- The initial version of SIFT [1] simply locates the keypoints at the scale-space extrema.
- An improved version [2] refines the localisation and scale by fitting a model onto to nearby data, which can improve the accuracy for image matching.

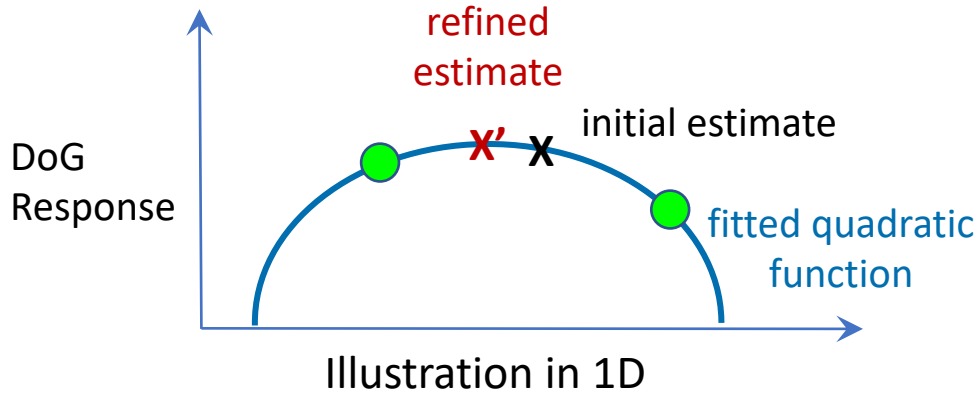


[1] D. Lowe. Object recognition from local scale-invariant features, ICCV 1999.
[2] D. Lowe. Distinctive image features from scale-invariant keypoints, IJCV 2004.

Keypoint localisation

- A quadratic function is fitted to the DoG response of neighbouring pixels.
- Then the location and scale of extremum for this quadratic function can be estimated.





Keypoint localisation

- Let us denote the DoG response by $D(x, y, \sigma)$ or $D(\boldsymbol{x})$, where $\boldsymbol{x} = (x, y, \sigma)^T$ is a 3D vector, containing both location and scale.
- By Taylor expansion, we have

$$D(\boldsymbol{x} + \Delta\boldsymbol{x}) = D(\boldsymbol{x}) + \frac{\partial D^T}{\partial \boldsymbol{x}} \Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^T \frac{\partial^2 D}{\partial \boldsymbol{x}^2}\Delta\boldsymbol{x}$$

We omit the higher order terms

The first derivatives can be estimated using finite difference, e.g. $\frac{\partial D}{\partial x} = \frac{D(x+1,y,\sigma) - D(x-1,y,\sigma)}{2}$.

The second derivatives or Hessian can be estimated using finite difference as well.

This is a quadratic function for variable $\Delta\boldsymbol{x}$. Here $\boldsymbol{x}$ is the initial estimate, $\Delta\boldsymbol{x}$ is the shift to the refined estimate.

Keypoint localisation

- To get a refined extrema for this function, we can let

$$\frac{\partial D(\boldsymbol{x}+\Delta\boldsymbol{x})}{\partial \Delta\boldsymbol{x}} = \frac{\partial D}{\partial \boldsymbol{x}} + \frac{\partial^2 D}{\partial \boldsymbol{x}^2}\Delta\boldsymbol{x} = 0$$

- Therefore,

$$\Delta\boldsymbol{x} = -\frac{\partial^2 D}{\partial \boldsymbol{x}^2}^{-1}\frac{\partial D}{\partial \boldsymbol{x}}$$

- We move from $\boldsymbol{x}$ by $\Delta\boldsymbol{x}$ and arrive at refined estimate.

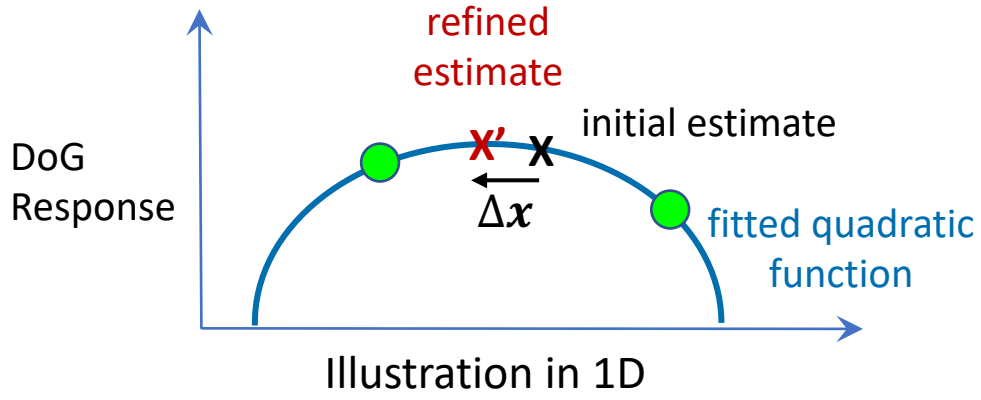


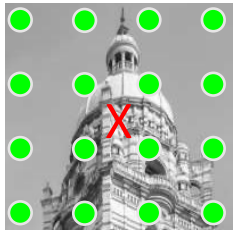
Illustration in 1D

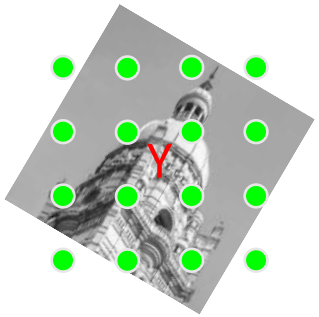
Keypoint localisation

- Note that $\boldsymbol{x} = (x, y, \sigma)^T$ is a 3D vector, containing both location and scale.
- We get refined estimates for both location (x, y) and scale σ.
  - Location: sub-pixel accuracy
  - Scale: some value between these scales σ, $\sqrt{2}\sigma$, 2σ , $2\sqrt{2}\sigma$, 4σ , $\cdots$

Orientation assignment

- This step assigns a consistent orientation to each keypoint, based on local image properties. As a result, the feature descriptor can be represented relative to this orientation and achieve invariance to image rotation.



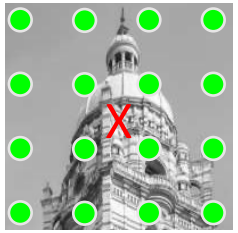


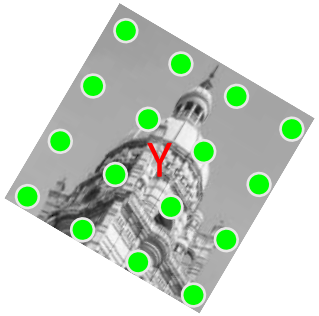
Idea:
If we know which orientation this building is, when we calculate features, we can **sampling** in a rotated coordinate system.

Feature descriptor at keypoint X should be similar to that at keypoint Y.

Orientation assignment

- This step assigns a consistent orientation to each keypoint, based on local image properties. As a result, the feature descriptor can be represented relative to this orientation and achieve invariance to image rotation.



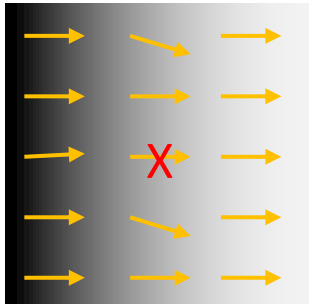


Idea:
If we know which orientation this building is, when we calculate features, we can **sampling** in a rotated coordinate system.

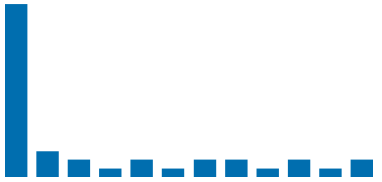
Feature descriptor at keypoint X should be similar to that at keypoint Y.

Orientation assignment

- How do we determine the dominant orientation θ for a neighbourhood of a keypoint?
- We can calculate the gradient orientation for pixels in this neighbourhood and vote for the dominant orientation.
  - An orientation histogram with 36 bins covering 360 degrees is created.
  - Each pixel votes for an orientation bin, weighted by the gradient magnitude.
  - The keypoint will be assigned an orientation, which is the peak of the histogram.



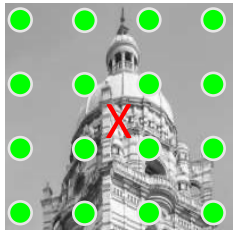
Gradient orientations in the neighbourhood of keypoint X.

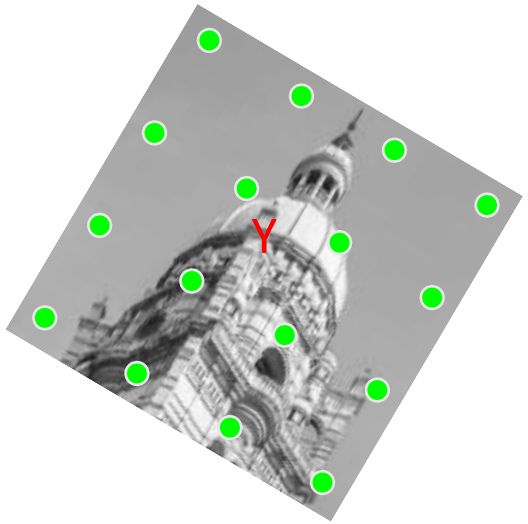


Orientation histogram. For example, the pixels in the above patch vote for the orientation of 0°.

Orientation assignment

- According to the keypoint location (x, y), scale σ and dominant orientation θ, we draw samples around this keypoint.
- For example, we can draw 4 x 4 samples using a window size proportional to σ, centred at (x, y), rotated by θ.



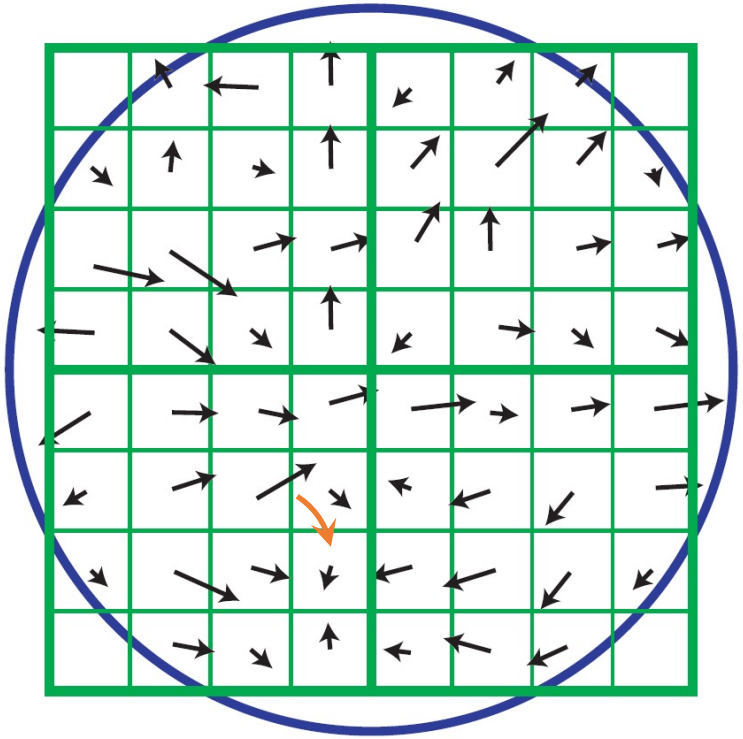


Local image descriptor

- Now we have these sample points, how do we describe the local image content?
  - Pixel intensities?
  - Gradient orientations?
  - Histograms?
- SIFT uses histograms of gradient orientations.

Local image descriptor

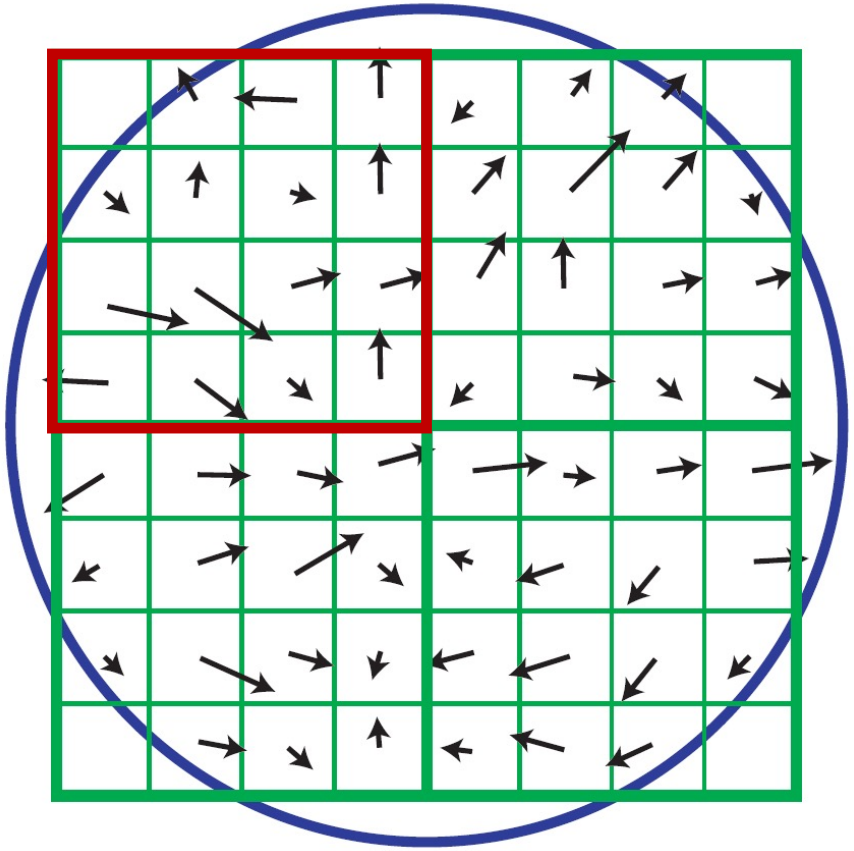
- Calculate the gradient magnitudes and orientations for these sampling points.
  - Note these orientations are calculated relative to the dominant orientation θ.
- Create a histogram of gradient orientations.
- In practice, subregions are used and each subregion has 4 x 4 samples.
  - Each subregion will have one orientation histogram.
  - We have a number of histograms, which together describe what it looks like around the keypoint.

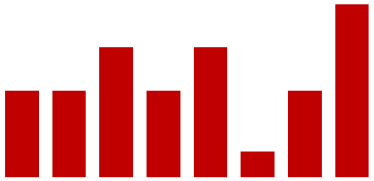


gradient orientation is rotated by θ for each sample

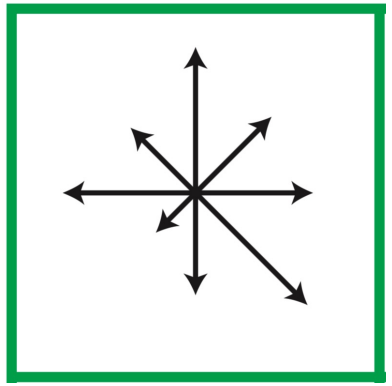
dominant orientation θ

Local image descriptor



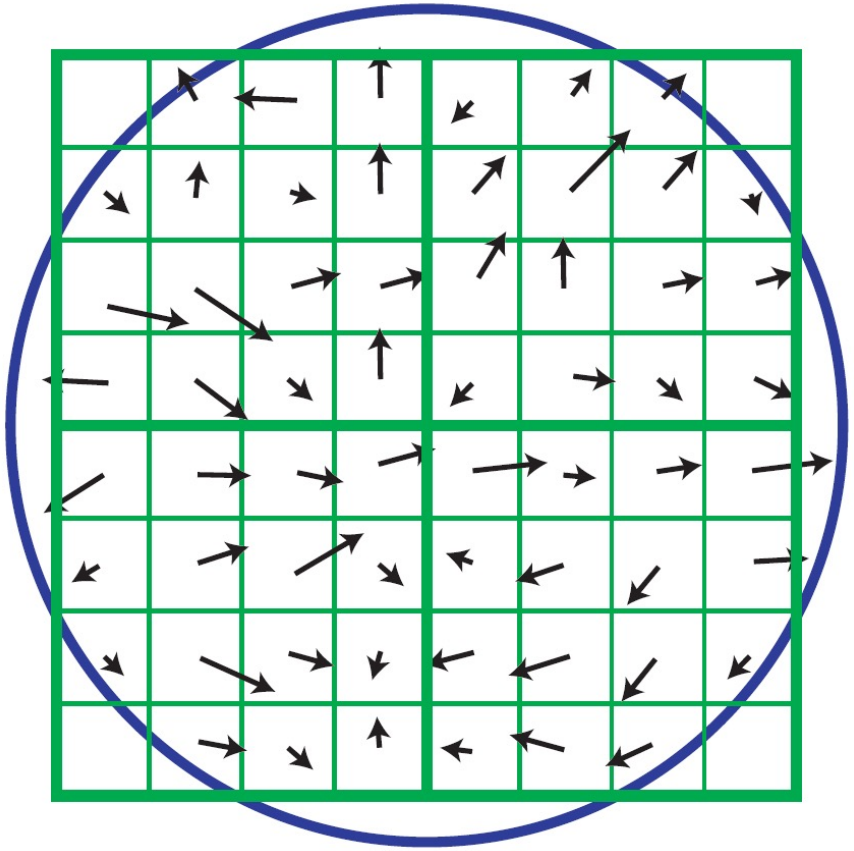


Orientation histogram
(8 bins)

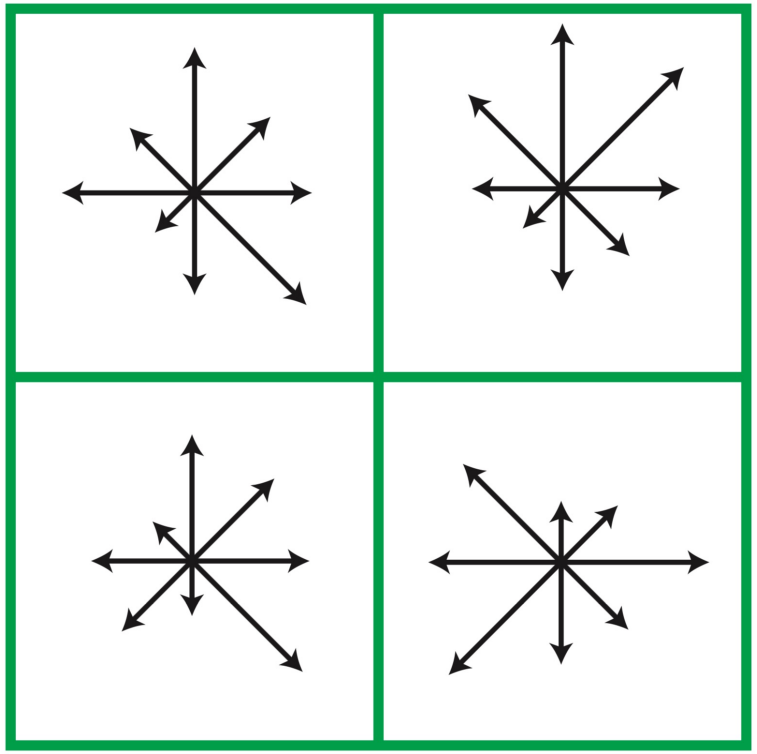


Length of arrow denotes the sum of gradient magnitude along this direction.

Local image descriptor

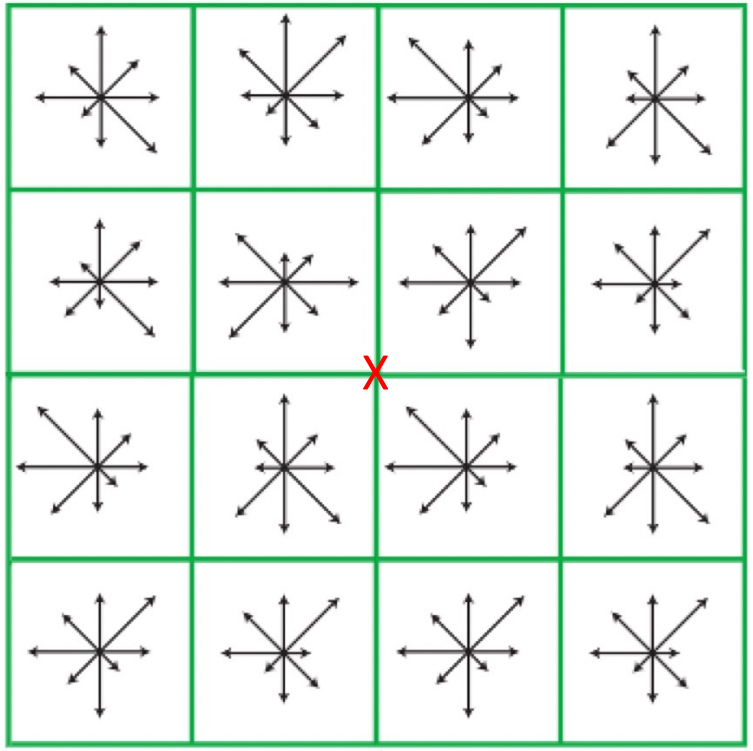


Orientation histograms
for 4 subregions



Local image descriptor

- In Lowe's implementation, 16 subregions are used, so the final SIFT descriptor looks like this.
- What is the dimension of this descriptor?
  - 16 subregions x 8 bins = 128
- We use a feature vector of 128 elements to describe each keypoint.

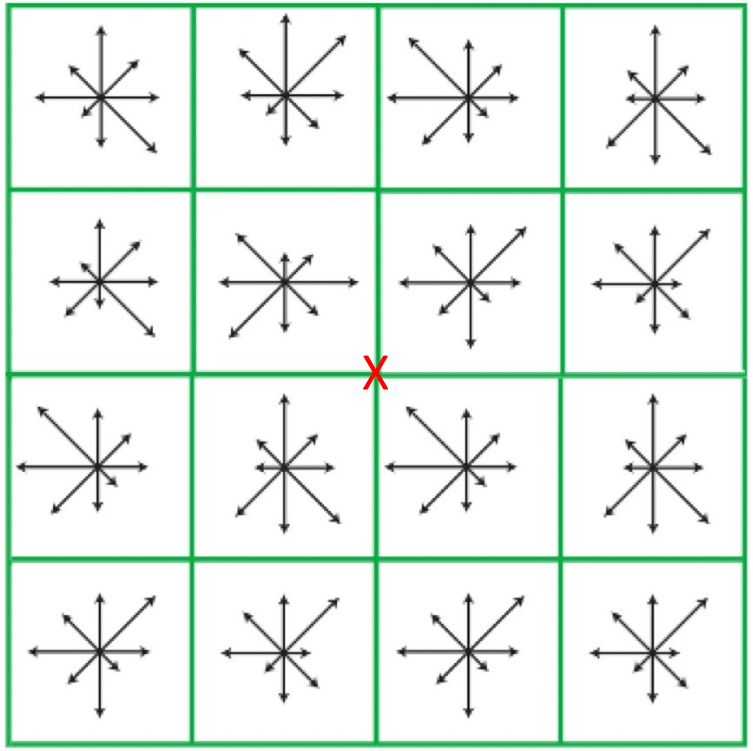


SIFT descriptor

[1] D. Lowe. Object recognition from local scale-invariant features, ICCV 1999.
[2] D. Lowe. Distinctive image features from scale-invariant keypoints, IJCV 2004.

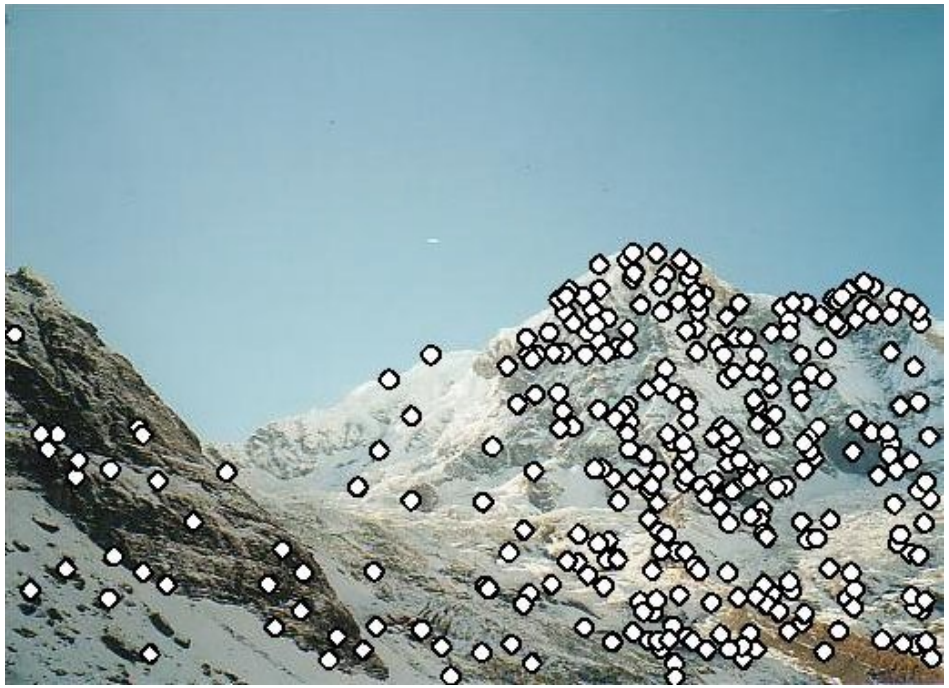
Local image descriptor

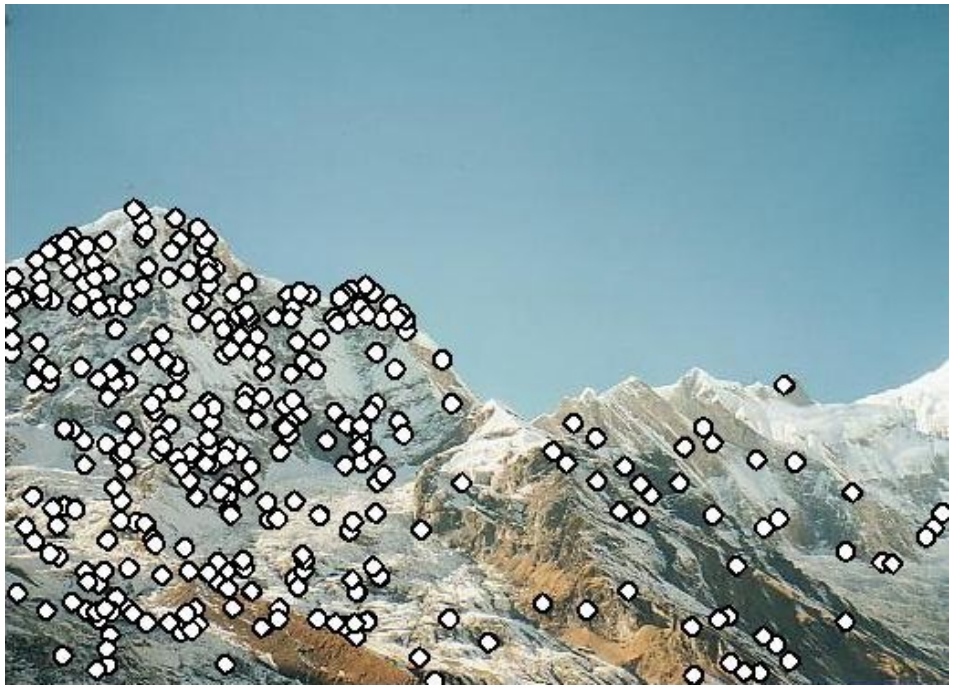
- This descriptor is robust to rotation, scaling and change of illuminations, due to
  - consideration of dominant orientation
  - a sampling window proportional to scale
  - use of gradient orientations for feature description
  - use of histograms



SIFT descriptor at keypoint X

[1] D. Lowe. Object recognition from local scale-invariant features, ICCV 1999.
[2] D. Lowe. Distinctive image features from scale-invariant keypoints, IJCV 2004.





- Now we have a good feature descriptor for each interest point in the two images.
- How do we know which point corresponds to which?

Keypoint matching

- Keypoints between two images are matched by identifying the nearest neighbours.
  - For each keypoint in image A, identify its nearest neighbours in the database of keypoints for image B.
  - The distance is defined by the Euclidean distance of SIFT descriptors.
- For efficiency, we may not need to find the exact nearest neighbour.
  - There are fast algorithms for finding approximate nearest neighbours.

Fast Library for Approximate Nearest Neighbour (FLANN). https://www.cs.ubc.ca/research/flann/

Keypoint matching

- After nearest neighbour search, we find that a keypoint (x, y) in image A corresponds to another keypoint (u, v) in image B.
- Assume that they are related by an affine transformation,

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

rotation, scaling … translation

Keypoint matching

- For many pairs of corresponding keypoints, the equation can be written as,

keypoint 1
keypoint 2
keypoint …

$$\begin{bmatrix} x_1 & y_1 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_1 & y_1 & 0 & 1 \\ x_2 & y_2 & 0 & 0 & 1 & 0 \\ 0 & 0 & x_2 & y_2 & 0 & 1 \\ & & \vdots & & & \end{bmatrix} \cdot \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \end{bmatrix}$$

- It can be written as a linear system,

$$Am = b$$

unknown affine parameters

Keypoint matching

- The least-square solution to the linear system is given by,

$$m = (A^T A)^{-1} A^T b$$

Moore-Penrose inverse

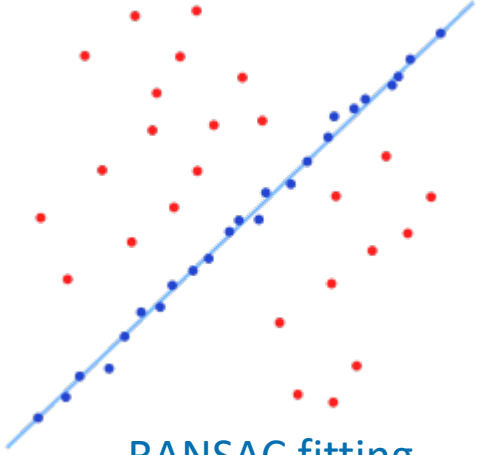
  which minimises the squared difference $\|Am - b\|^2$.

- Once the affine parameters are solved, we know the spatial transformation between the two images.
- Matching is done!

Keypoint matching

- The squared difference $\|Am - b\|^2$ is sensitive to outliers.
- If we want to improve the robustness of matching, we can employ methods which consider outliers in model fitting, such as Random Sample Consensus (RANSAC).

Data points



RANSAC fitting
(red: outliers; blue: inliers)

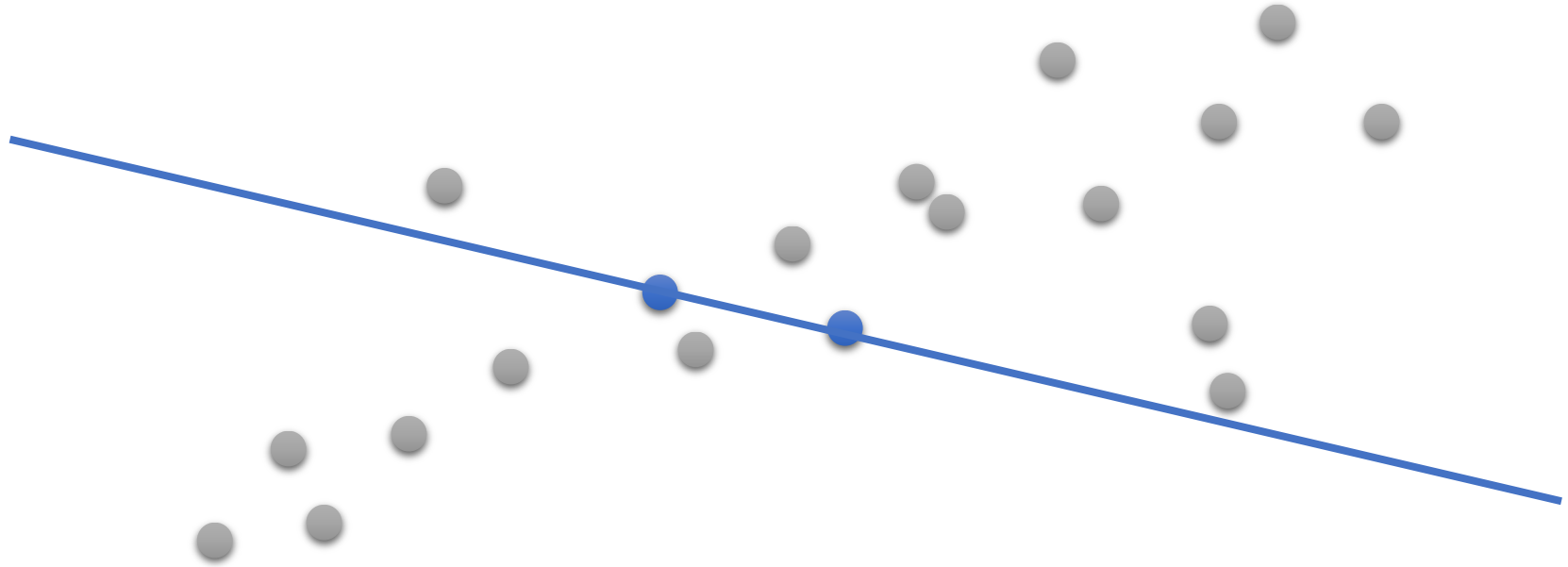
RANSAC

Task: find the best fitting line to the points.

RANSAC

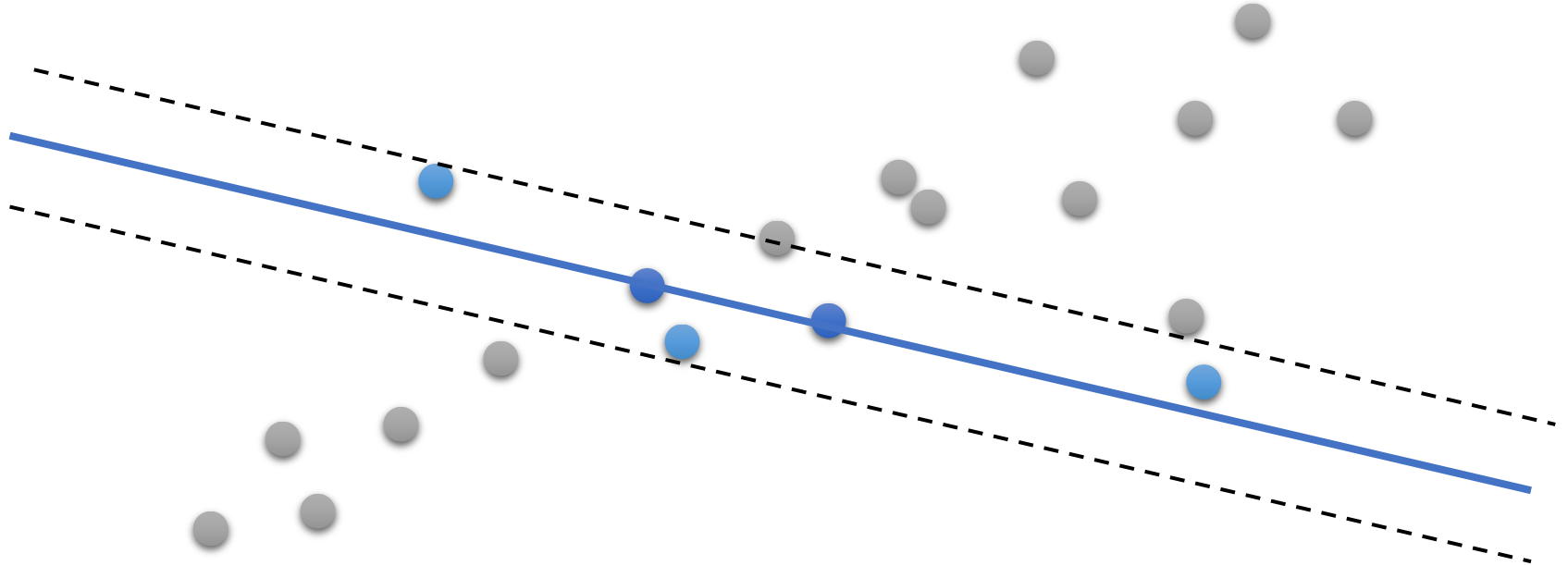
Randomly sample some points.

RANSAC



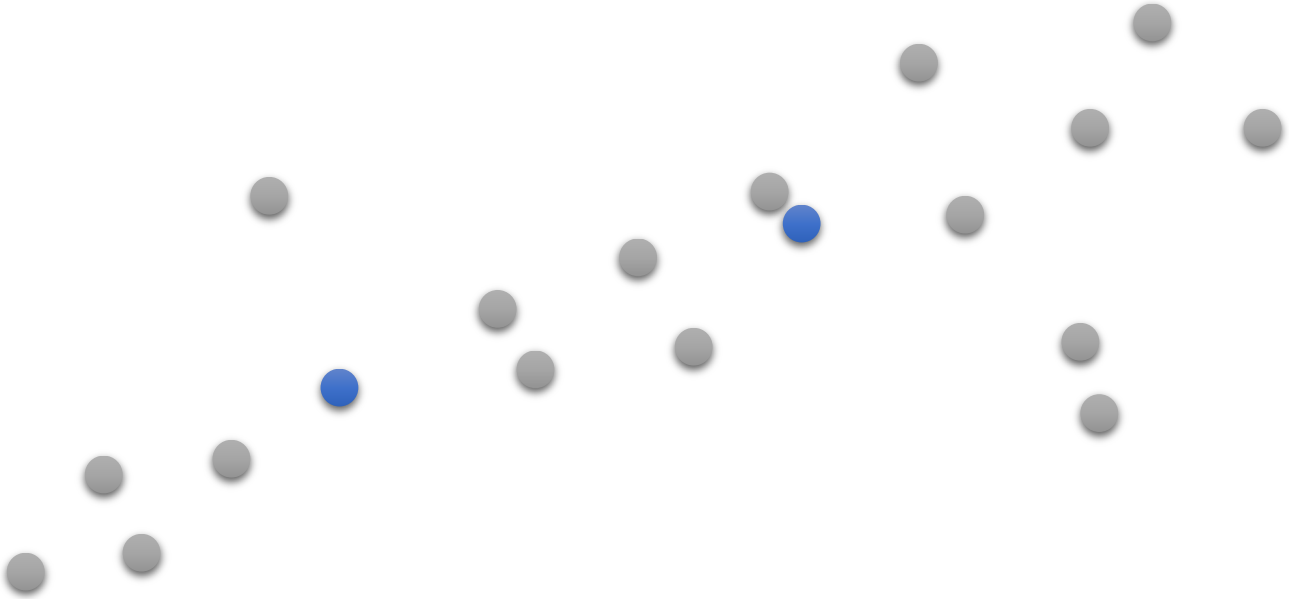
Fit a line.

RANSAC



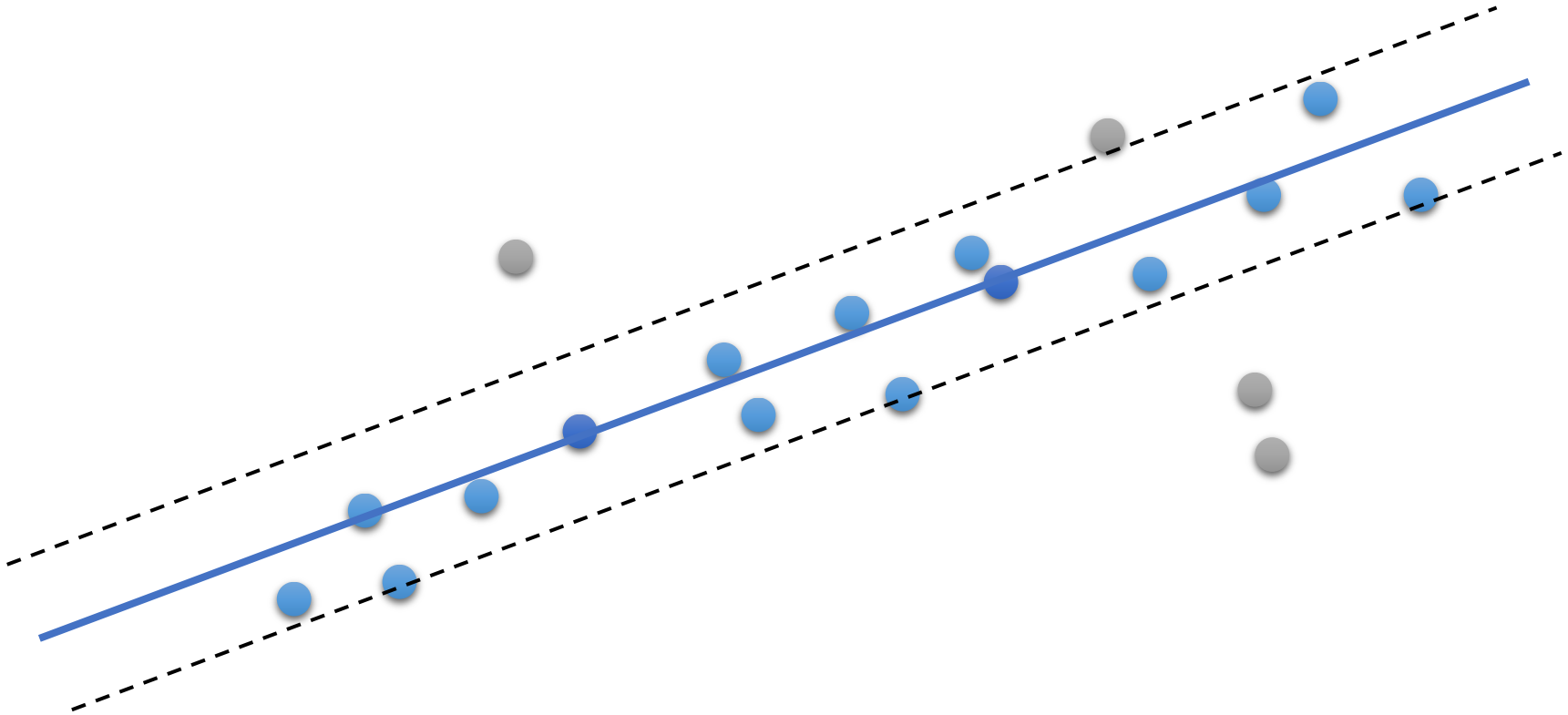
5 inliers found within a threshold to the line.

RANSAC



Repeat the random sampling.

RANSAC



15 inliers found.
Repeat until we get a good fitting line.

RANSAC

- For each iteration
  - Select random sample points.
  - Fit a model.
  - Based on the model, check how many points are inliers.
  - Terminate after a certain number of iterations or enough inliers have been found.

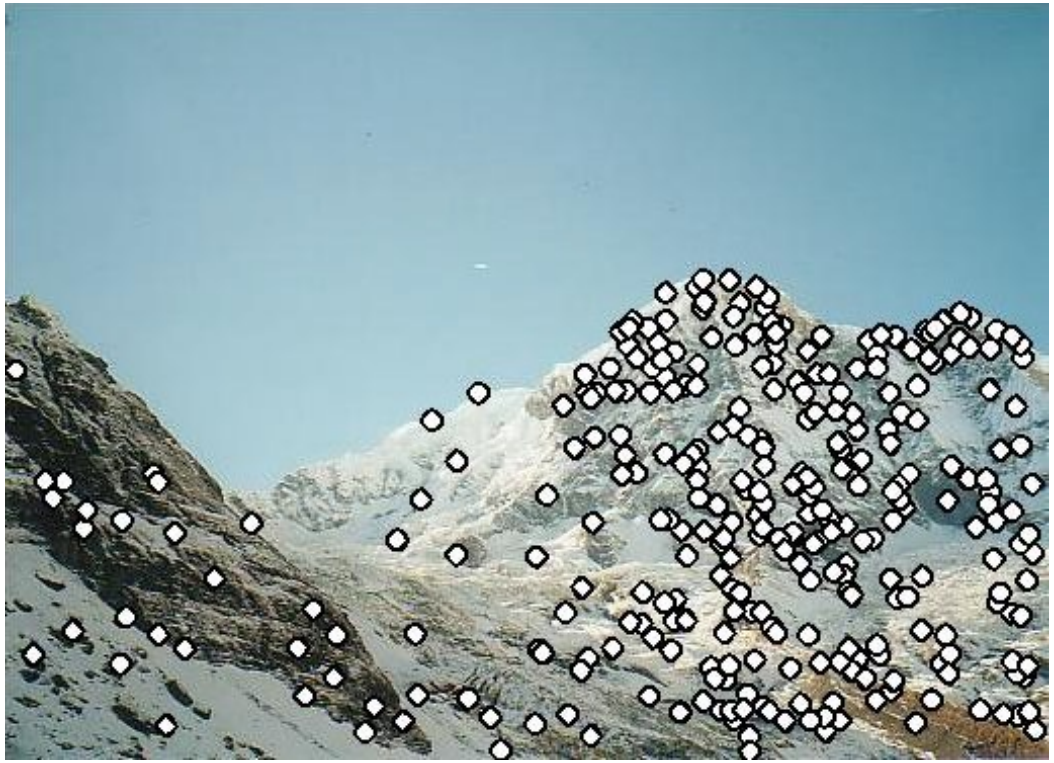
Keypoint matching

- Using RANSAC, we can find a good solution of this linear system, even when there are noise or outliers.

$$Am = b$$

Applications of keypoint matching

- By matching the keypoints, we can estimate the transformation between two images and stitch them to create a panorama.



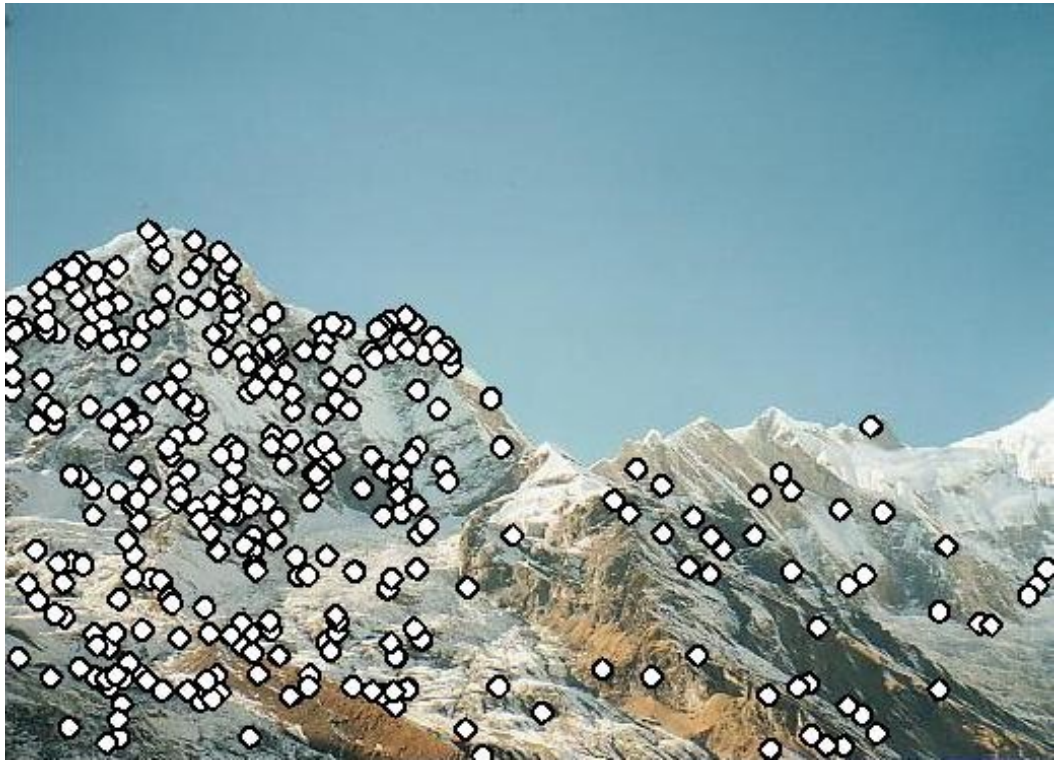


Image stitching, panorama on cameras

Multiple images

Image stitching

Image blending (weighted average of overlapping regions)

M. Brown and D. Lowe. Automatic panoramic image stitching using invariant features. IJCV, 2007.

Applications of keypoint matching

template image

find the template
in a new image

Object recognition by image matching

Summary

- Feature descriptors extract local features near an interest point.
- They are useful for finding the correspondence between two images.
- Robustness to rotation, scaling and intensity changes are the three main aspects to consider in designing feature descriptors.
- We have described in detail how SIFT works.

References

- Sec. 4.1.2 Feature descriptors, Sec 6.1 2D and 3D feature-based alignment. Richard Szeliski, Computer Vision: Algorithms and Applications (http://szeliski.org/Book).