CS 773
Scale-invariant feature transform

Chia-Yen Chen

Dept. of Computer Science
The University of Auckland
yen@cs.auckland.ac.nz

Feature detection

- For more general applicability, develop a detector that is invariant to scale and rotation, as well as being robust to the variations corresponding to typical viewing conditions
- Instead of just detecting particular types of features, use descriptor to distinctively represent the feature

SIFT

Scale-invariant feature transform

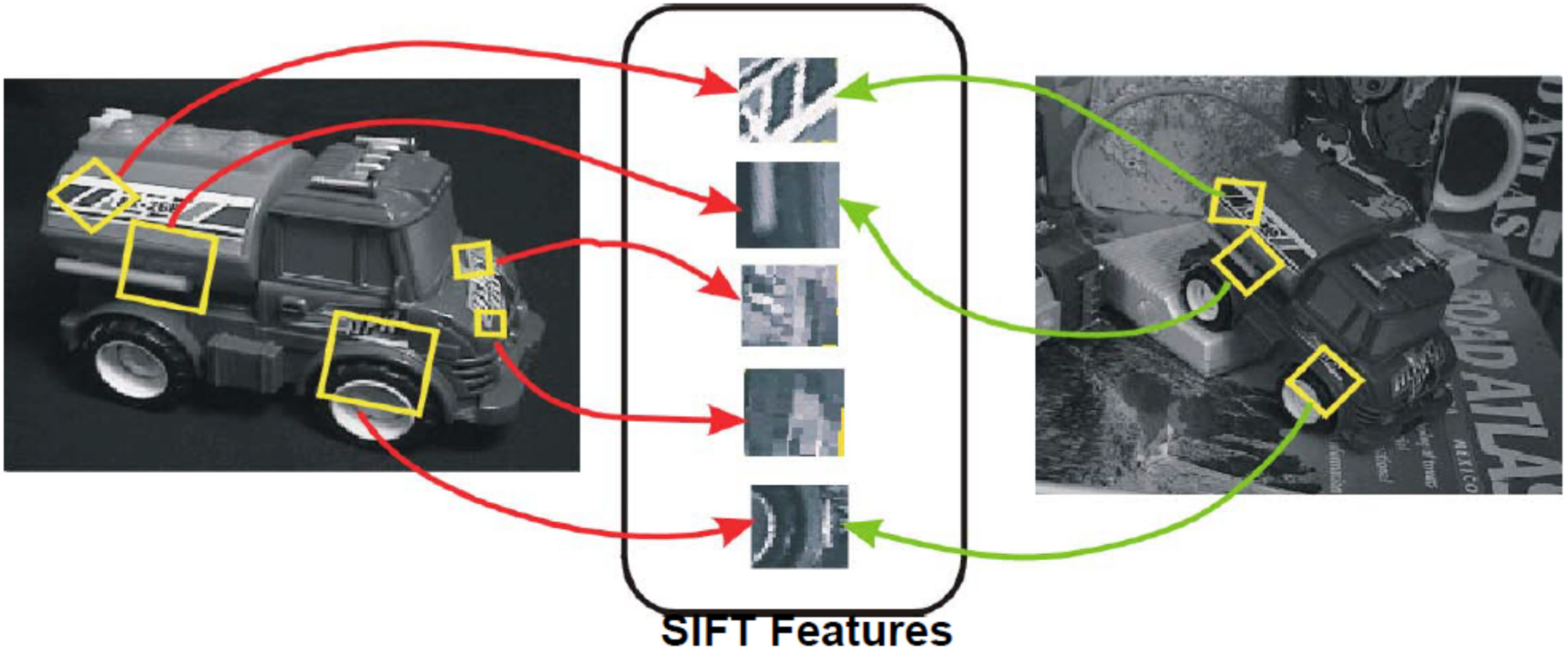
- Proposed by David Lowe in 1999 [1] and improved in 2004 [2]
- Extract feature points and represent them with descriptors
- SIFT feature descriptor is invariant to
  - uniform scaling
  - orientation
  - illumination changes
  - affine distortion (partially)

1. Lowe, David G. (1999). "Object recognition from local scale-invariant features". Proceedings of the International Conference on Computer Vision. 2. pp. 1150–1157. doi:10.1109/ICCV.1999.790410.
2. Lowe, David G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints". International Journal of Computer Vision. 60 (2): 91–110. doi:10.1023/B:VISI.0000029664.99615.94.

Recommended reading

Concepts of SIFT

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



Adapted from https://courses.cs.washington.edu/courses/cse576/11sp/notes/SIFT_white2011.pdf

Advantages of SIFT

- **Locality:** features are local, robust to occlusion and clutter (no prior segmentation)
- **Distinctiveness:** individual features can be matched to a large database of objects
- **Quantity:** many features can be generated for even small objects
- **Efficiency:** close to real-time performance
- **Extensibility:** can easily be extended to wide range of differing feature types, with each adding robustness

Steps in SIFT

- Detection of Scale-Space Extrema
  - Search for possible points of interest over multiple scales and locations
- Keypoint Localization
  - Determine feature points based on stability
- Orientation Assignment
  - Use local image gradient direction to determine best orientation for each keypoint
- Local Image Descriptor
  - Compute a distinctive descriptor vectors (128 elements) from the gradients at selected scale and rotation to represent the keypoints

Detection of Scale-Space Extrema

- Goal: Identify locations and scales that can be repeatably assigned under different views of the same scene or object.
- Method: search for stable features across multiple scales using a continuous function of scale.

Gaussian

- The scale space of an image is a function $L(x,y,\sigma)$ that is produced from the convolution of a Gaussian kernel (at different scales) with the input image.

Detection of Scale-Space Extrema

Scale space :

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$*$: convolution of variable scale Gaussian, $G(x, y, \sigma)$, with input image $I(x, y)$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

Detection of Scale-Space Extrema

- Perform scale-space filtering using Difference of Gaussian (DoG)

$$D(\mathrm{x}, \mathrm{y}, \sigma) = (G(\mathrm{x}, \mathrm{y}, k\sigma) - G(\mathrm{x}, \mathrm{y}, \sigma)) * I(\mathrm{x}, \mathrm{y})$$

$$= L(\mathrm{x}, \mathrm{y}, k\sigma) - L(\mathrm{x}, \mathrm{y}, \sigma)$$

Detection of Scale-Space Extrema

- Scale space is separated into octaves:
  - Octave 1 uses scale σ
  - Octave 2 uses scale 2σ

- In each octave, the initial image is repeatedly convolved with Gaussians to produce a set of scale space images.
- Adjacent Gaussians are subtracted to produce the DoG
- After each octave, the Gaussian image is down-sampled by a factor of 2 to produce an image ¼ the size to start the next level.

Detection of Scale-Space Extrema

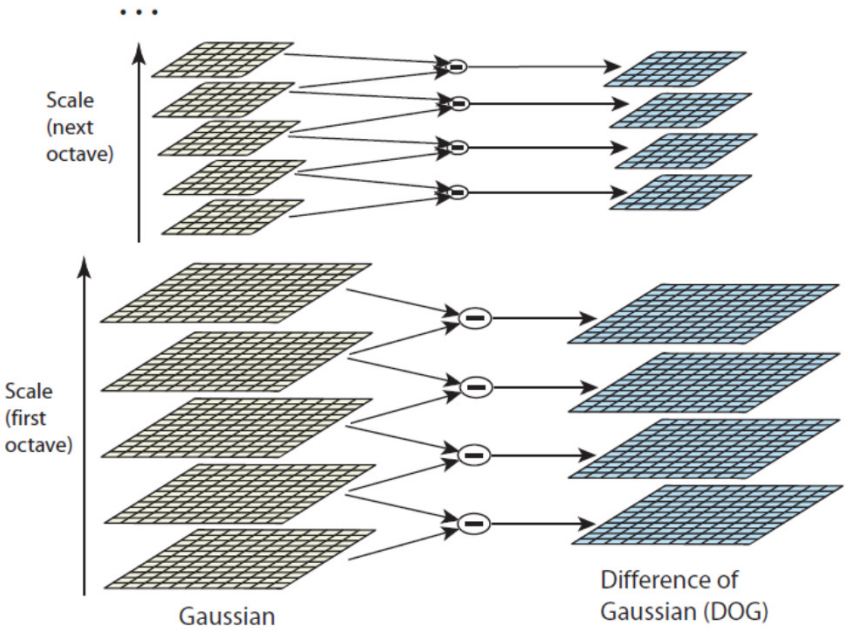


Figure 1: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

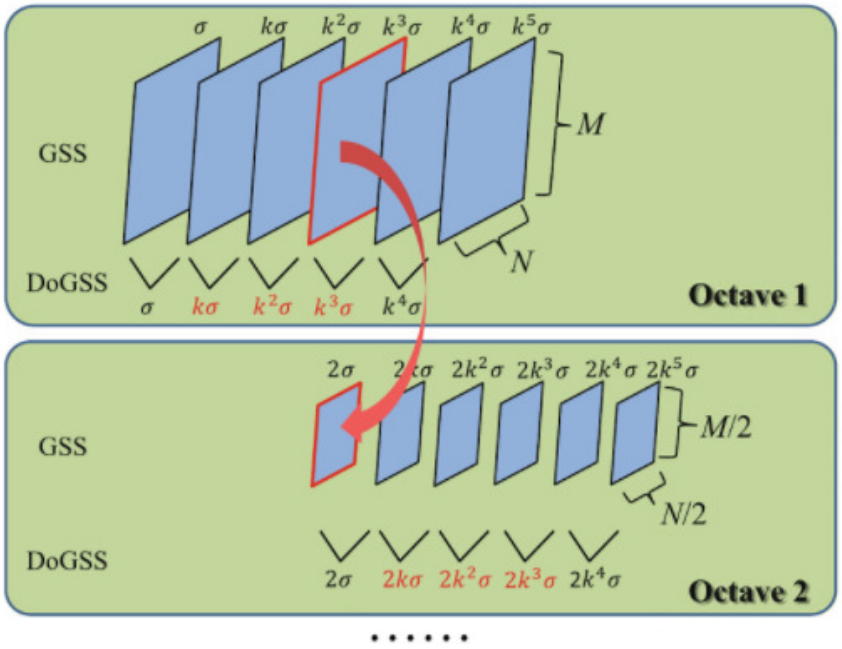
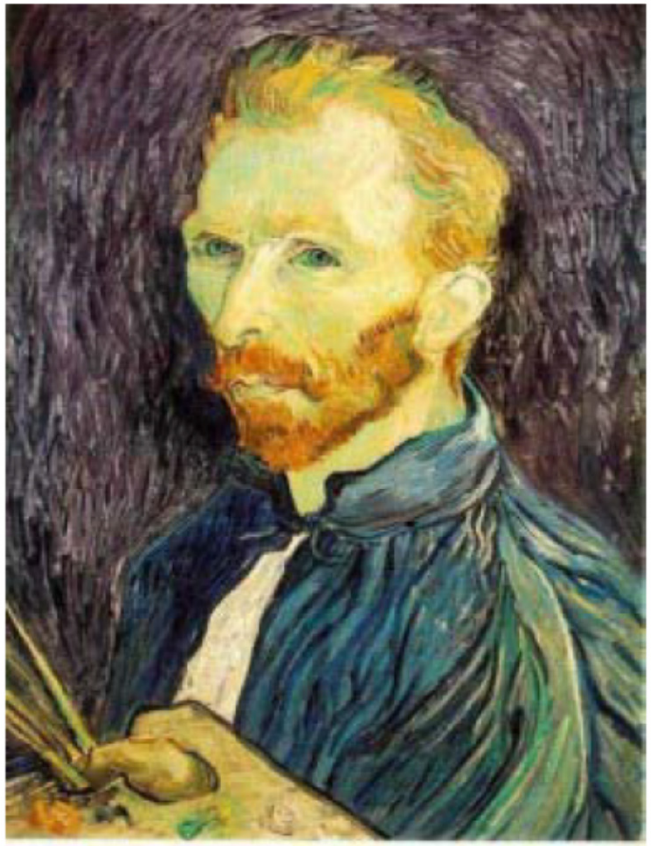


Fig. 2.1 The scale space representation implemented in SIFT when $s = 3$. DoG scale space (*DoGSS*) is generated by subtracting adjacent images in the Gaussian scale space (*GSS*). The *red* values indicate the scales that will be used for keypoint detection by 3D extrema search. To make these scales consistently differ by a factor of k, it has to generate $s + 2$ images in each octave of DoGSS and $s + 3$ images in GSS. See the text for details

Bin Fan, Zhenhua Wang, Fuchao Wu, Local Image Descriptor: Modern Approaches

Example of subsampling



Gaussian 1/2

G 1/4

G 1/8

Adapted from https://courses.cs.washington.edu/courses/cse576/11sp/notes/SIFT_white2011.pdf

Keypoint Localization

- Detect maxima and minima of difference-of-Gaussian in scale space
- Each point is compared to its 8 neighbours in the current image and 9 neighbours each in the scales above and below

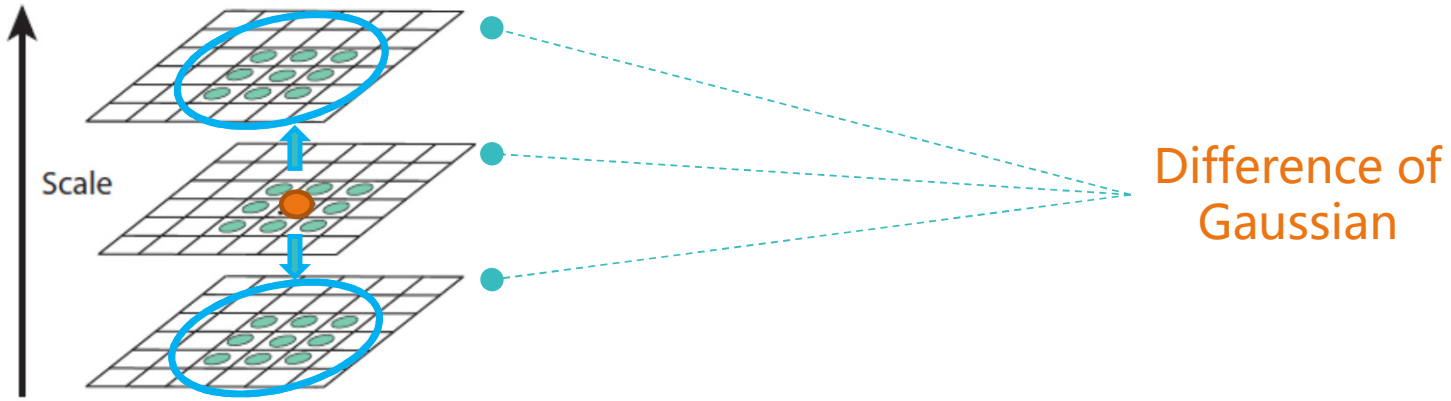


Figure 2: Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in 3x3 regions at the current and adjacent scales (marked with circles).

Keypoint Localization

- Detect maxima and minima of difference-of-Gaussian in scale space
- Eliminate unstable feature points
  - Low contrast
  - Poorly localized edge points

Keypoint Localization

- Once a keypoint candidate is found, perform a detailed fit to nearby data to determine
  - location, scale, and ratio of principal curvatures
- Initially, keypoints were found at location and scale of a central sample point (Lowe, 1999).
- In newer work, fit a 3D quadratic function to improve interpolation accuracy (Brown and Lowe, 2002).
- Use shifted Taylor expansion of $D(x, y, \sigma)$

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

Offset from sample point: $\mathbf{x} = (x, y, \sigma)^T$

Keypoint Localization

- Taking derivative of $D(\mathrm{x})$ to obtain extremum

$$\hat{\mathrm{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathrm{x}^2}\frac{\partial D}{\partial \mathrm{x}}$$

- Substitute $\hat{\mathrm{x}}$ back into $D(\mathrm{x})$

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

- If $|D(\hat{\mathrm{x}})|$ is lower than a threshold (in paper, 0.03), then discarded as unstable

Keypoint Localization

- Use Hessian matrix to eliminate edge responses.

$$\mathrm{H}=\begin{vmatrix} \frac{\partial^2 D}{\partial x^2} & \frac{\partial^2 D}{\partial y \partial x} \\ \frac{\partial^2 D}{\partial y \partial x} & \frac{\partial^2 D}{\partial y^2} \end{vmatrix}$$

- Eigenvalues α and β, with $\alpha > \beta$

$$Tr(\mathrm{H}) = \alpha + \beta = \frac{\partial^2 D}{\partial x^2} + \frac{\partial^2 D}{\partial y^2}$$

$$Det(\mathrm{H}) = \alpha\beta = \frac{\partial^2 D}{\partial x^2}\frac{\partial^2 D}{\partial y^2} - \frac{\partial^2 D}{\partial xy}\frac{\partial^2 D}{\partial yx}$$

- Let $\alpha = r\beta$, and use ratio of principal curvature $\frac{Tr(\mathrm{H})}{Det(\mathrm{H})} < \frac{(r+1)^2}{r}$ to eliminate point on edges (in paper, *r*=10)

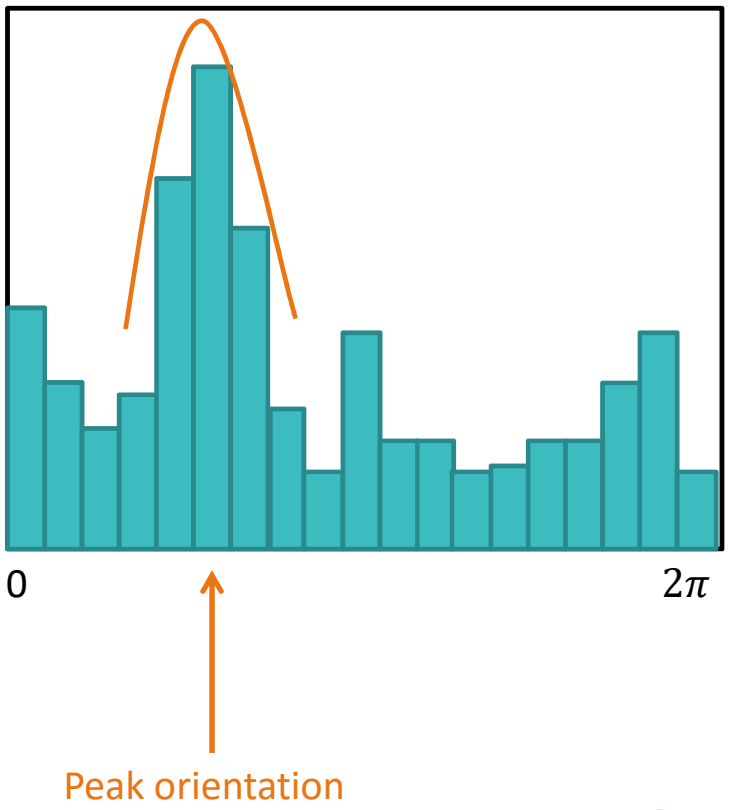
Orientation Assignment

- Create histogram of local gradient directions at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)

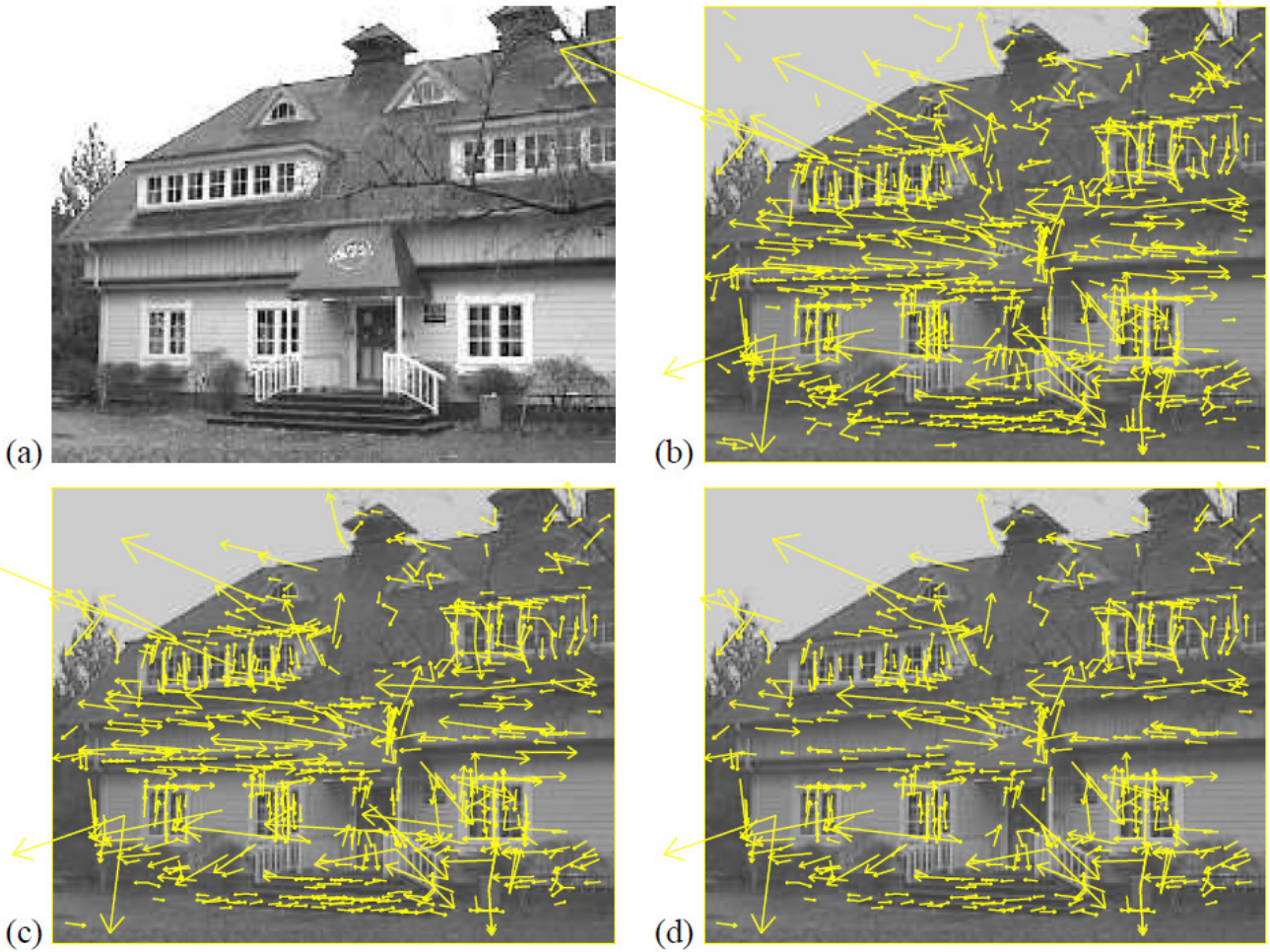
- Gradient magnitude $$m(x,y) = \sqrt{\left(L(x+1,y) - L(x-1,y)\right)^2 + \left(L(x,y+1) - L(x,y-1)\right)^2}$$
- Gradient orientation $$\theta(x,y) = \tan^{-1}(L(x+1,y) - L(x-1,y) / \left(L(x,y+1) - L(x,y-1)\right)$$

Orientation Assignment

- Construct orientation histogram from gradient orientations with 36 bins covering 360 degrees
- Each sample is weighted by magnitude and a Gaussian-weighted circular window
- Peak in the orientation histogram correspond to dominant direction
- Parabola fitted to values around peak to improve accuracy



Orientation Assignment



This figure shows the stages of keypoint selection.
(a) The 233x189 pixel original image.
(b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location.
(c) After applying a threshold on minimum contrast, 729 keypoints remain.
(d) The final 536 keypoints that remain following an additional threshold on ratio of principal curvatures.

Local Image Descriptor

- Each keypoint now has
  - Location
  - Scale
  - orientation

- Next, compute a descriptor for the local image region about each keypoint that is
  - highly distinctive
  - as invariant as possible to variations such as changes in viewpoint and illumination

Local Image Descriptor

- A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left.

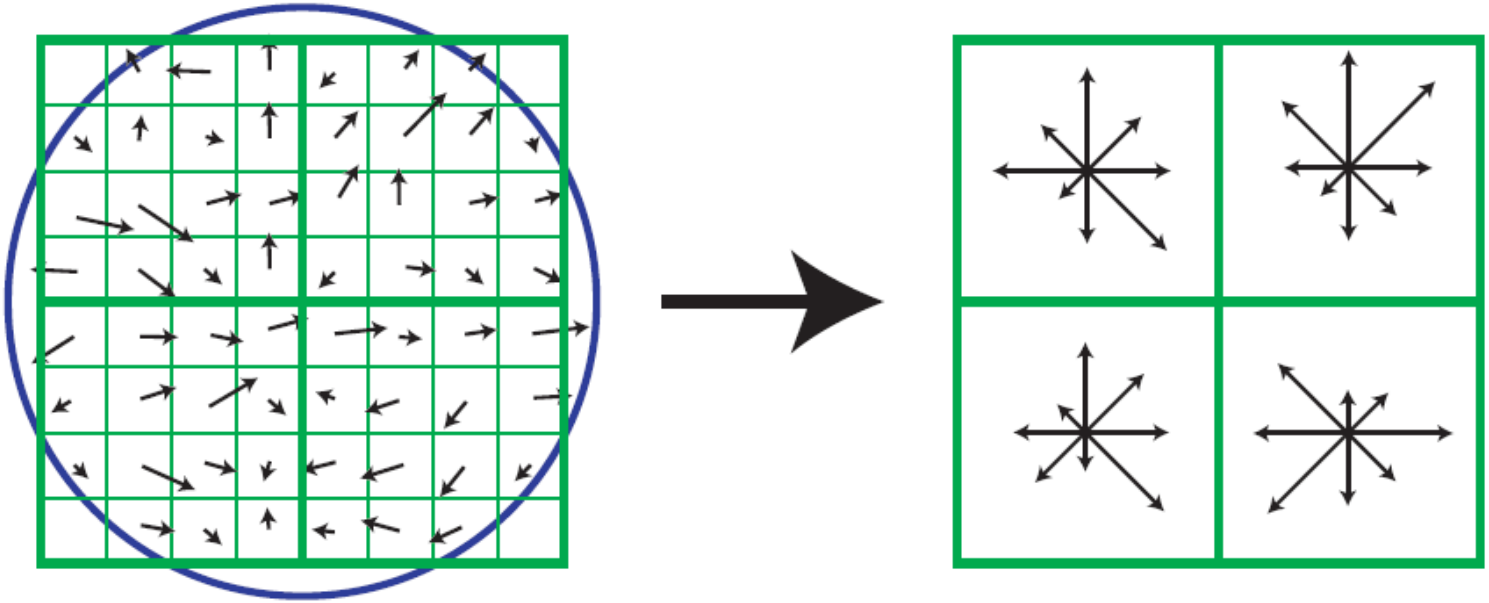


Image gradients

Keypoint descriptor

Local Image Descriptor

- These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region.

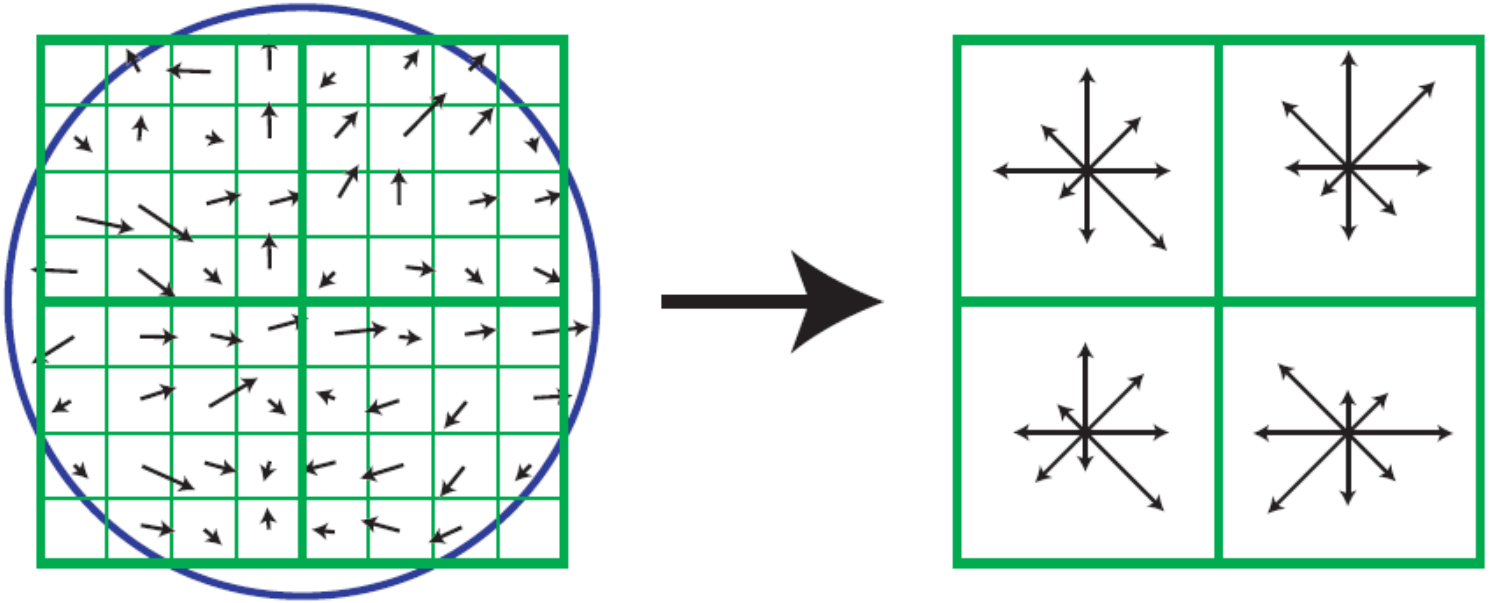


Image gradients

Keypoint descriptor

Local Image Descriptor

- This figure shows a 2x2 descriptor array computed from an 8x8 set of samples. Descriptor is formed from a vector containing values of all the orientation histogram entries corresponding to the lengths of the arrows on the right side

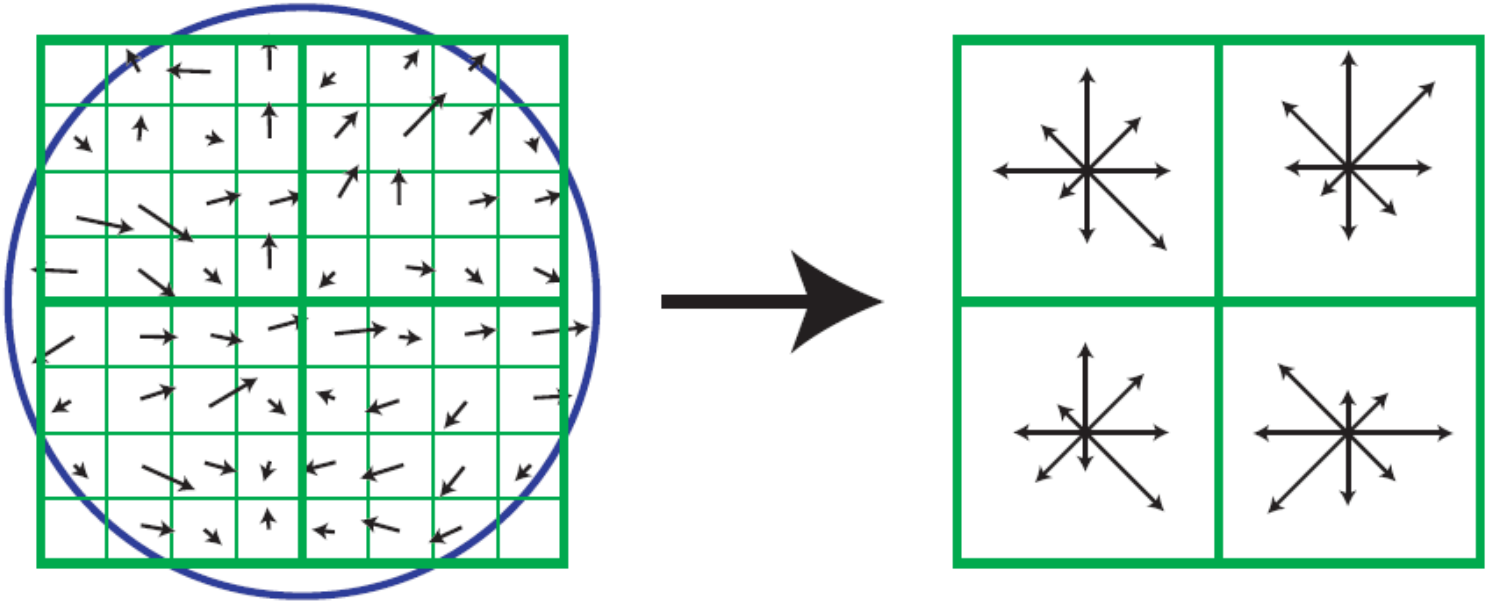


Image gradients

Keypoint descriptor

Local Image Descriptor

- The feature vector is modified to reduce the effects of illumination change.
  - the vector is normalized to unit length.
- Reduce the influence of large gradient magnitudes
  - thresholding the values in the unit feature vector to be no larger than 0.2, and then renormalizing to unit length.

Applications of SIFT

SIFT has been quite popular for feature extraction

- Image alignment (homography, fundamental matrix)
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation
- … many others