Blobs and SIFT features

Morten R. Hannemose, mohan@dtu.dk

March 25, 2022

02504 Computer vision course lectures,
DTU Compute, Kgs. Lyngby 2800, Denmark

Learning objectives

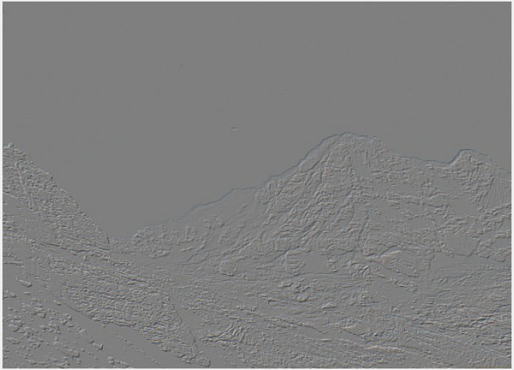
After this lecture you should be able to:

- implement and use blob detection using Difference-of-Gaussians
- analyse and use SIFT features and feature matching

Similarity

Pixel-wise comparison

- Shift of single pixel vs. two views



Similarity

Basic idea

- Locally appearance between views is the same
- Variation can be handled via invariances

Local image features

SIFT – key elements

- Features localized at interest points
- Adapted to scale and invariant to appearance changes

SIFT – scale invariant feature transform (Lowe, 2004)

- Scale-space blob detection – difference of Gaussians
- Interest point localization
- Orientation assignment
- Interest point descriptor
- Note – SIFT is one example of interest point feature

Harris corners and BLOBs

Harris corners are features that have a large change of intensity in two orthogonal directions. They are:

- local,
- can be found at different scales by changing the Gaussian filters, and
- work in rotated frames.

Harris corners are found by first order derivatives whereas blobs are response to second order image derivatives.

BLOBs – Binary Large OBjects

Correspond to:

- a dark area surrounded by brighter intensities or,
- a bright area surrounded by darker intensities

Hessian

The Hessian matrix contains the second order derivatives

$$\mathbf{H}(x, y) = \begin{bmatrix} I_{xx}(x, y) & I_{xy}(x, y) \\ I_{xy}(x, y) & I_{yy}(x, y) \end{bmatrix},$$

where

$I_{xx}(x, y) = \frac{\partial^2 I(x,y)}{\partial x^2}$, $I_{yy}(x, y) = \frac{\partial^2 I(x,y)}{\partial y^2}$, and $I_{xy}(x, y) = \frac{\partial^2 I(x,y)}{\partial x \partial y}$.

Curvature

Second order derivatives measure curvature.

Eigenvalues (λ_1, λ_2) measure the principal curvature, i.e. degree of change in intensity

Eigenvectors measure direction of change where the eigenvector corresponding to the largest eigenvalue (λ_1) is the direction of most change and the second is orthogonal to that.

Blob detection with Hessian

Similar to the Harris corner detector, we can use either of the measures

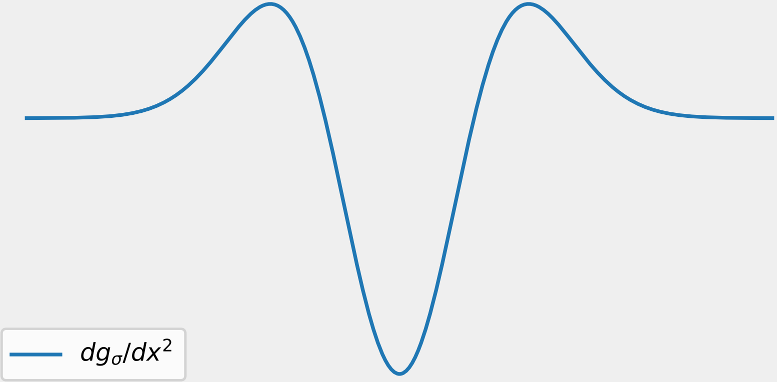
$$\det(\mathbf{H}) = \lambda_1 \lambda_2,$$
$$\operatorname{trace}(\mathbf{H}) = \lambda_1 + \lambda_2,$$

where λ_i are the eigenvalues of the Hessian.

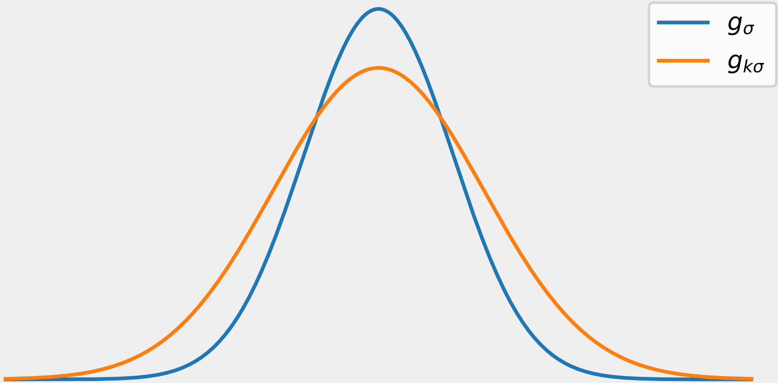
$\det(\mathbf{H})$ is the Gaussian curvature.

$\operatorname{trace}(\mathbf{H}) = \nabla^2 I$ is the Laplacian, which we use for blob detection.

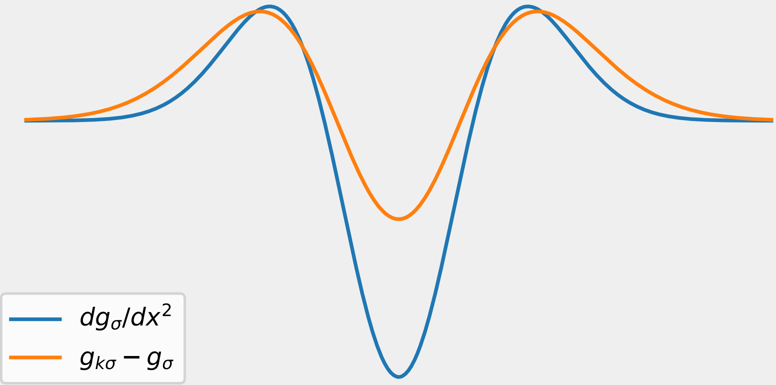
The Laplacian



Two Gaussians with different variance



Difference of Gaussians vs Laplacian



Blob detection with DoG

An approximation of $\mathrm{trace}(\mathbf{H})$ is the Difference-of-Gaussian (DoG).

Consider the following convolution with two different Gaussian kernels:

$$\nabla^2 I \approx D_\sigma = (G_{k\sigma} - G_\sigma) * I = L_{k\sigma} - L_\sigma,$$

where G_σ is a Gaussian convolution with a width of σ and $k > 1$ is a scale factor.

Blob detection with DoG

Not that kind of dog..

SIFT – Scale invariance

SIFT – Scale invariance

- Using difference of Gaussians for blob detection

$$
\begin{aligned}
D(x, y, \sigma) &= \big((G_{k\sigma} - G_\sigma) * I\big)(x, y) \\
&= L(x, y, k\sigma) - L(x, y, \sigma)
\end{aligned}
$$

SIFT

We need to ensure that the DoG is scale normalized (the DoG kernel)

$$\sigma^2 \nabla^2 G$$

If we take an offset in the heat equation, we have

$$\sigma \nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G_{k\sigma} - G_\sigma}{k\sigma - \sigma}$$

SIFT

Multiplying both sides with σ we get

$$\begin{aligned}\sigma^2\nabla^2 G &\approx \frac{\sigma}{k\sigma - \sigma}(G_{k\sigma} - G_\sigma)\\ &= \frac{1}{k-1}(G_{k\sigma} - G_\sigma)\end{aligned}$$

From this we get

$$(k-1)\sigma^2\nabla^2 G \approx G_{k\sigma} - G_\sigma$$

Since k is constant over scales it does not affect the relative response of the DoG kernel.

Gaussian scale space – Efficient

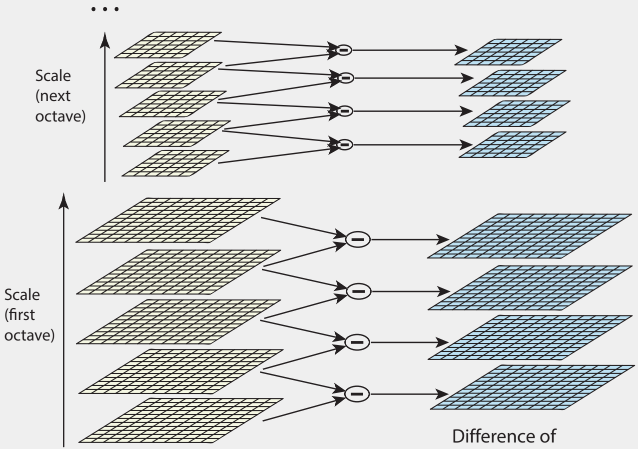
- Convolution of two Gaussians yield a new Gaussian
- Generate scale space by iteratively blurring already blurred images more
  - Requires smaller Gaussian kernels
- $k = 2^{\frac{1}{3}}$

Gaussian scale space – Efficient

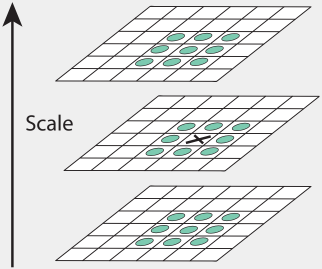
- Convolution of two Gaussians yield a new Gaussian
- Generate scale space by iteratively blurring already blurred images more
  - Requires smaller Gaussian kernels
- $k = 2^{\frac{1}{3}}$
- σ doubles after three images, the image is downsampled.
  - This is an octave.

SIFT – Estimation of DoG

Difference of Gaussians



Extrema localization



SIFT – Interest point localization

- Taylor approximation to second degree of 2D surface

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

- Setting the derivative of $D(x)$ to zero

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

SIFT – Interest point localization

- We get

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

- $|D(\hat{\mathbf{x}})| > 0.03$ otherwise the point is discarded.

SIFT – Interest point along edges discarded

- The eigenvalues of the Hessian are proportional to the principal curvatures

$$\mathbf{H} = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix} = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix}$$

Interest point along edges discarded

- λ_1 and λ_2 are the eigenvalues of the Hessian

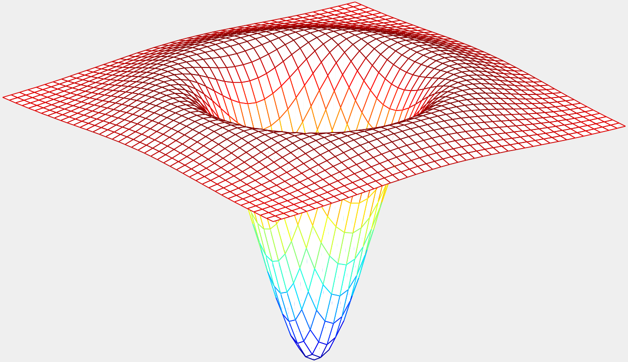
$$\mathrm{trace}(\mathbf{H}) = I_{xx} + I_{yy} = \alpha + \beta$$
$$\det(\mathbf{H}) = I_{xx}I_{yy} - I_{xy}^2 = \alpha\beta$$

Points are kept if

$$\frac{\mathrm{trace}(\mathbf{H})^2}{\det(\mathbf{H})} < \frac{(r+1)^2}{r},$$

where $r = 10$ (found to be a good heuristic)

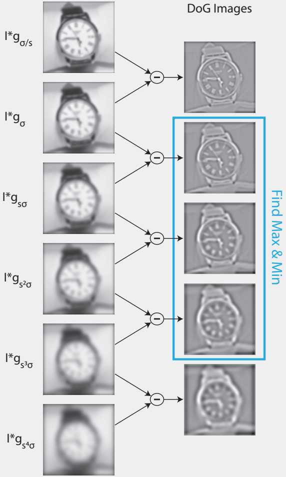
DoG measure



Features have $|\mathrm{DoG}(I)| > \tau$, where τ is a threshold.

DoG in scale pyramids

Scale pyramids are smaller scaled versions of the same image.
Scaled DoG is subtraction between all layers.



Scale space blobs and DoG

DoG at different scales makes for a scale invariant feature detector.
Small and large details are recoverable in different DoGs.

SIFT – Orientation assignment

Compute the orientation of gradients in a small region around the BLOB.

$$m(x, y) = \sqrt{L_x^2 + L_y^2}$$
$$\theta(x, y) = \arctan 2(L_y, L_x)$$

Where

$$L_x = L(x+1, y) - L(x-1, y)$$
$$L_y = L(x, y+1) - L(x, y-1)$$

SIFT – Orientation assignment

- Compute circular histogram of gradient orientations
  - Weighted by magnitude, smoothed, and has 36 bins

SIFT – Orientation assignment

- Compute circular histogram of gradient orientations
  - Weighted by magnitude, smoothed, and has 36 bins
- Use peak in histogram to assign orientation of point
- This introduces rotation invariance.
- Can we have multiple peaks in histogram?

SIFT – Orientation assignment

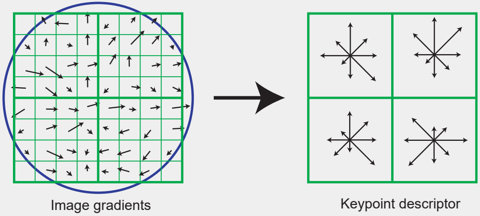
- Compute circular histogram of gradient orientations
  - Weighted by magnitude, smoothed, and has 36 bins
- Use peak in histogram to assign orientation of point
- This introduces rotation invariance.
- Can we have multiple peaks in histogram?
  - Yes, this can happen at e.g. corners.
  - Create a new point at the same location if peak is over 80% of max.

SIFT – Invariances

- Position
- Scale
- Rotation
- Linear intensity change
- Perspective changes?

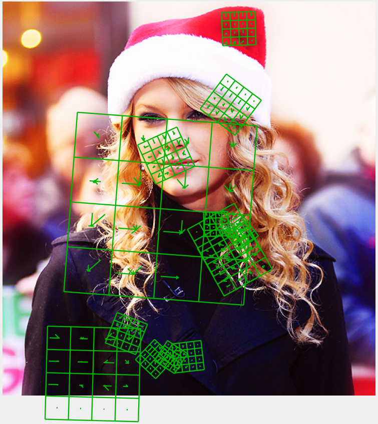
SIFT – Descriptor

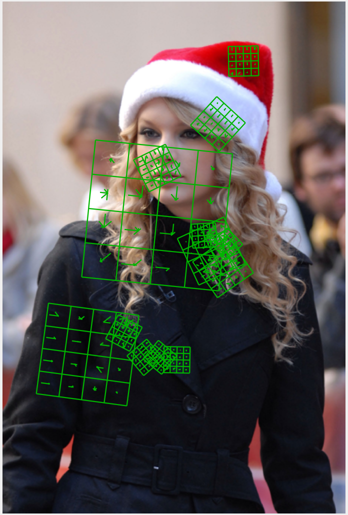
- Create local patch at scale and orientation of point
- Build a histogram of local gradient orientations



- Normalized using L_2 norm: $\mathbf{d}_n = \frac{1}{\sqrt{\sum_{i=1}^{128} \mathbf{d}(i)^2}} \mathbf{d}$

Taylor SIFT





Taylor SIFT

SIFT – Matching of descriptors

- Use Euclidean distance between normalized vectors

$$\delta(\mathbf{d}_i, \mathbf{d}_j) = \sqrt{\sum_{n=1}^{128} (d_{i,n} - d_{j,n})^2}$$

- Note – for comparison the square root is not needed

RootSIFT

Simple trick to improve SIFT matching

- SIFT is a histogram
- Euclidean distance is dominated by large values
- RootSIFT is a transformation that measures distance using the Hellinger kernel.
  - L1 normalize
  - Take the square root of each element
  - L2 normalize the resulting vector
- Compare using Euclidean distance

SIFT – Matching of descriptors

- For each feature in image 1 ($\boldsymbol{d_{1,i}}$) find the closest feature in image 2 ($\boldsymbol{d_{2,j}}$)
  - This will give a lot of incorrect matches
- Cross checking
  - Only keep matches where $\boldsymbol{d_{2,j}}$ is also the closest to $\boldsymbol{d_{1,i}}$ of all features in image 1

SIFT – Matching of descriptors

- For each feature in image 1 ($\boldsymbol{d_{1,i}}$) find the closest feature in image 2 ($\boldsymbol{d_{2,j}}$)
  - This will give a lot of incorrect matches
- Cross checking
  - Only keep matches where $\boldsymbol{d_{2,j}}$ is also the closest to $\boldsymbol{d_{1,i}}$ of all features in image 1
- Ratio test
  - Compute the ratio between the closest and second closest match, and keep where this is below a threshold, e.g. 0.7.

SIFT – Summary

- SIFT is both a feature detector and descriptor
- Find local extrema of DoGs in scale space
- Place patch oriented along local gradients
- Compute histograms of gradients.
- Allows matching of images invariant to: scale, rotation, illumination and viewpoint
- Partly visible objects can be matched

Other descriptors

SIFT is widely used.

- ORB
- SURF
- BRISK
- FAST

Learned descriptors

- Deep Learning has created improved feature descriptors.
- Mostly in improvement in invariance to changing lighting.
- Some examples:
  - R2D2: Repeatable and Reliable Detector and Descriptor
    https://github.com/naver/r2d2
  - Superpoint
    https://github.com/magicleap/SuperPointPretrainedNetwork

Exercise

Build a blob detector and match points with SIFT detector.

Python: Recommended to use OpenCV (4.2.0 or newer)

Matlab: Recommended to use VLFeat

https://www.vlfeat.org/overview/sift.html

Midterm Evaluation

Exercise time!