# Beeldverwerken Lab4

Hannah Min, Harm Manders

May 2018

## 1 Scale Invariant Feature Transform: SIFT

Sift is an algorithm that can match different images of the same objects based on key point features in each image. It is scale and orientation invariant so no matter from what angle the object is seen, you would be able to match them as long as both images have enough comparable keypoints. To generate keypoints Sift uses a gaussian filter to look for local structures in an image. It uses multiple gaussian filters with different $\sigma$'s and stores them in a 3D matrix. To localise the keypoints at each scale it uses an approximation of the Laplacian of Gaussians (LoG) namely the Difference of Gaussian (DoG). The basic idea behind this is that the LoG is a good filter to detect blobs at a certain scale but it requires a processing power to convolve the image multiple times on multiple scales. So Lowe has created the DoG which simply subtracts two Gaussian-blurred images with scales that are close to each other (See Appendix). After keypoints are localised they are described by a 1x128 vector. In this vector all kinds of information is stored about the keypoint so it will be scale, orientation and illumination invariant.

After an image is "described" by its keypoint features and descriptors two images can be compared by comparing their descriptors. The descriptors that are closest to each other in each image can be matched together. To prevent too many false positives SIFT looks at the second nearest neighbour to discard them.

When two images are matched a multitude of algorithms can be applied to them. One such application is image stitching/mosaicing in which two images of the same scene are stitched together to render one total scene, Another could be image recognition.

## 2 Random Sample Consensus: RANSAC

Ransac is a method to estimate which points are outliers and which aren't. The idea is that when you randomly sample a couple of times and every time calculate how many outliers you have. After a couple of tries you will find a sample that has the most outliers, as long as you sample often enough. To calculate the correct amount of samples you can use the following formula:

$$1 - (1 - (1 - e)^s)^N = p$$

where:
e = probability that a point is an outlier
s = number of points to sample
N = number of samples
p = desired probability that we get a good sample
When we rewrite it:

$$N = \frac{log(1 - p)}{log(1 - (1 - e)^s)}$$

# Appendix: Theory questions

**3.3** Scale space is way to look at images structures from different scales. It is used to create a sequence of smoothed images. Here, the smoothing parameter t determines the size of the (Gaussian) kernel that is used to smooth the image by convolution.
To find the scale space extrema for each pixel we look at it's 8 neighbours in the same layer and his 9 neighbours in the layer above and below it. If this pixel is either the lowest or highest value of the 27 pixels, it is called a scale space extreme.

**3.4** Relation D and . Key points have the property that they are invariant to scale change of an image. Therefore Lowe uses a Difference-of-Gaussian (DoG) function to detect those key points. This calculation means subtracting the smoothed image of scale $\sigma$ from the smoothed image of nearby scale $k\sigma$.

Lowe says that the DoG is an approximation of the Lapplacian-of-Gaussians (LoG). The LoG is the $\frac{\partial^2}{\partial\sigma}$ and an approximation of $\frac{\partial^2}{\partial\sigma}$ with $\delta\sigma = k$ is:

$$\sigma^2 \nabla G * f(x, y) = \frac{\partial^2}{\partial\sigma} * f(x, y) \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{\sigma(k - 1)} * f(x, y)$$

therefore

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k - 1)\sigma^2 \nabla G$$

The DoG is the subtraction of two Gaussian smoothed images.

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$$

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

So

$$D(x, y, \sigma) = [G(x, y, k\sigma) - G(x, y, \sigma)] * f(x, y)$$
$$D(x, y, \sigma) = (k - 1)\sigma^2 \nabla G * f(x, y)$$

This shows that DoG is an approximation of LoG.

**3.5** We want to schow that the trace of the Hessian is equal to the sum of its eigenvalues.
The Hessian is as follows: $\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$
To calculate the eigenvalues $\lambda$ of $\mathbf{H}$ we solve:
$|\mathbf{H} - \lambda\mathbf{I}|$=det$\begin{bmatrix} D_{xx} - \lambda & D_{xy} \\ D_{xy} & D_{yy} - \lambda \end{bmatrix} = 0.$
$(D_{xx} - \lambda)(D_{yy} - \lambda) - (D_{xy})(D_{yx}) = 0$
$-D_{xy}^2 + D_{xx}D_{yy} - \lambda D_{xx} - \lambda D_{yy} + \lambda^2 = 0$
$\lambda^2 - \lambda(D_{xx} + D_{yy}) + (D_{xx}D_{yy} - D_{xy}^2) = 0$
Using the abc-formula to solve for $\lambda$ we get:
$\lambda = \frac{(D_{xx}+D_{yy})\pm\sqrt{(D_{xx}+D_{yy})^2-4(D_{xx}+D_{yy}-D_{xy}^2)}}{2}$
$\lambda_1+\lambda_2 = \frac{(D_{xx}+D_{yy})+\sqrt{(D_{xx}+D_{yy})^2-4(D_{xx}+D_{yy}-D_{xy}^2)}}{2}+\frac{(D_{xx}+D_{yy})-\sqrt{(D_{xx}+D_{yy})^2-4(D_{xx}+D_{yy}-D_{xy}^2)}}{2}$
$\lambda_1 + \lambda_2 = \frac{2(D_{xx}+D_{yy})}{2} = D_{xx} + D_{yy}$
This is equal to the sum of the diagonal of $\mathbf{H}$, trace$(\mathbf{H}) = D_{xx} + D_{yy}$
An Hessian is always diagonalizable, because it is a symmetrical matrix.

**3.6** Magic Numbers (1pt) D(x̂) = 0.03

**3.8** $D(\hat{x}) = D + \frac{\partial D^T}{\partial \hat{x}}\hat{x} + \frac{1}{2}\hat{x}^T\frac{\partial^2 D}{\partial \hat{x}^2}\hat{x}$, where $\hat{x} = -\frac{\partial^2 D}{\partial \hat{x}^2}^{-1}\frac{\partial D}{\partial \hat{x}}$

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

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

$D(\hat{x}) = D + \frac{\partial D^T}{\partial \hat{x}} \cdot -\frac{\partial^2 D}{\partial \hat{x}^2}^{-1}\frac{\partial D}{\partial \hat{x}} + \frac{1}{2} \cdot (\frac{\partial^2 D}{\partial \hat{x}^2}^{-1}\frac{\partial D}{\partial \hat{x}})^T\frac{\partial D}{\partial \hat{x}}$
$D(\hat{x}) = D + -\frac{\partial D^T}{\partial \hat{x}}\frac{\partial^2 D}{\partial \hat{x}^2}^{-1}\frac{\partial D}{\partial \hat{x}} + \frac{1}{2} \cdot \frac{\partial D}{\partial \hat{x}}^T(\frac{\partial^2 D}{\partial \hat{x}^2}^T)^{-1}\frac{\partial D}{\partial \hat{x}}$
Since $\frac{\partial^2 D}{\partial \hat{x}^2}$ is the Hessian which is a symmetrical matrix, it's transpose is the same, so $\frac{\partial^2 D}{\partial \hat{x}^2}^T = \frac{\partial^2 D}{\partial \hat{x}^2}$
$D(\hat{x}) = D + -\frac{\partial D^T}{\partial \hat{x}}\frac{\partial^2 D}{\partial \hat{x}^2}^{-1}\frac{\partial D}{\partial \hat{x}} + \frac{1}{2} \cdot \frac{\partial D}{\partial \hat{x}}^T\frac{\partial^2 D}{\partial \hat{x}^2}^{-1}\frac{\partial D}{\partial \hat{x}}$
$D(\hat{x}) = D - \frac{1}{2}\frac{\partial D^T}{\partial \hat{x}}\frac{\partial^2 D}{\partial \hat{x}^2}^{-1}\frac{\partial D}{\partial \hat{x}}$
The factor $\frac{1}{2}$ isn't a typo.

**3.9**

**3.10**