Bauhaus-Universität Weimar

Image Analysis and Object Recognition

Exercise 2

image filtering and
interest points

SS 2014

(Course notes for internal use only!)

- Exercise dates
  - 15.04.2014 → Introduction + Exercise 1
  - **29.04.2014 → Exercise 2**
  - 13.05.2014 → Exercise 3
  - 27.05.2014 → Exercise 4
  - 10.06.2014 → Exercise 5
  - 24.06.2014 → Exercise 6
  - 08.07.2014 → Final Meeting / Summary / Discussion

- Every 2nd Week, **17:00 – 18:30**

Image Analysis and Object Recognition

The course gives an introduction to the basic concepts of pattern recognition and image analysis. It covers topics as image enhancement, local and morphological operators, edge detection, image representation in frequency domain, Fourier transform, Hough transform, segmentation, thinning and object categorization.

Please notice: The materials for our lectures and exercises are only available through the network of the Bauhaus-University Weimar.

Lecture

Exercise

Please note: Due to the license terms MATLAB can only be installed on computers of the university! If you plan to use MATLAB for the exercises please use the LiNT-Pool (Bauhausstraße 11, first floor). If you prefer to use your own computer please use the software Octave. Information how to install Octave are provided in the slides of the first exercise session.

MATLAB primer (Mathworks, University of Florida)

Exercise 1 (15.04.2014, 17:00, SR 015)

Basic information extraction: image enhancement, global thresholding (binarization) and morphological filtering (slides, exercise sheet, input image, example source code).

Exercise 2 (29.04.2014, 17:00, SR 015)

Teaching

- Photogrammetric Computer Vision
- Image Analysis and Object Recognition
- Parallel and Distributed Systems
- Spatial Information Systems (GIS)
- Geodesy
- Image-based 3D Reconstruction
- Hot Topics in Computer Vision

Quicklinks

- Faculty Media
- Faculty Civil Engineering
- Computer Science and Media
- Geodesy

Submissions / Registrations

- 12 groups
- 30 students
- Any submissions missing?

  → Don’t hesitate to send them!

- **All groups completed the first assignment!**

Overview

- Exercise 1

- Exercise 2
  - Image-filtering: Gradient of Gaussian (GoG)
  - Interest points: Förstner operator

Exercise 1

Tasks for me:

- Give a better introduction to MATLAB
- Give better explanation of single tasks
- Formulate single tasks more precisely

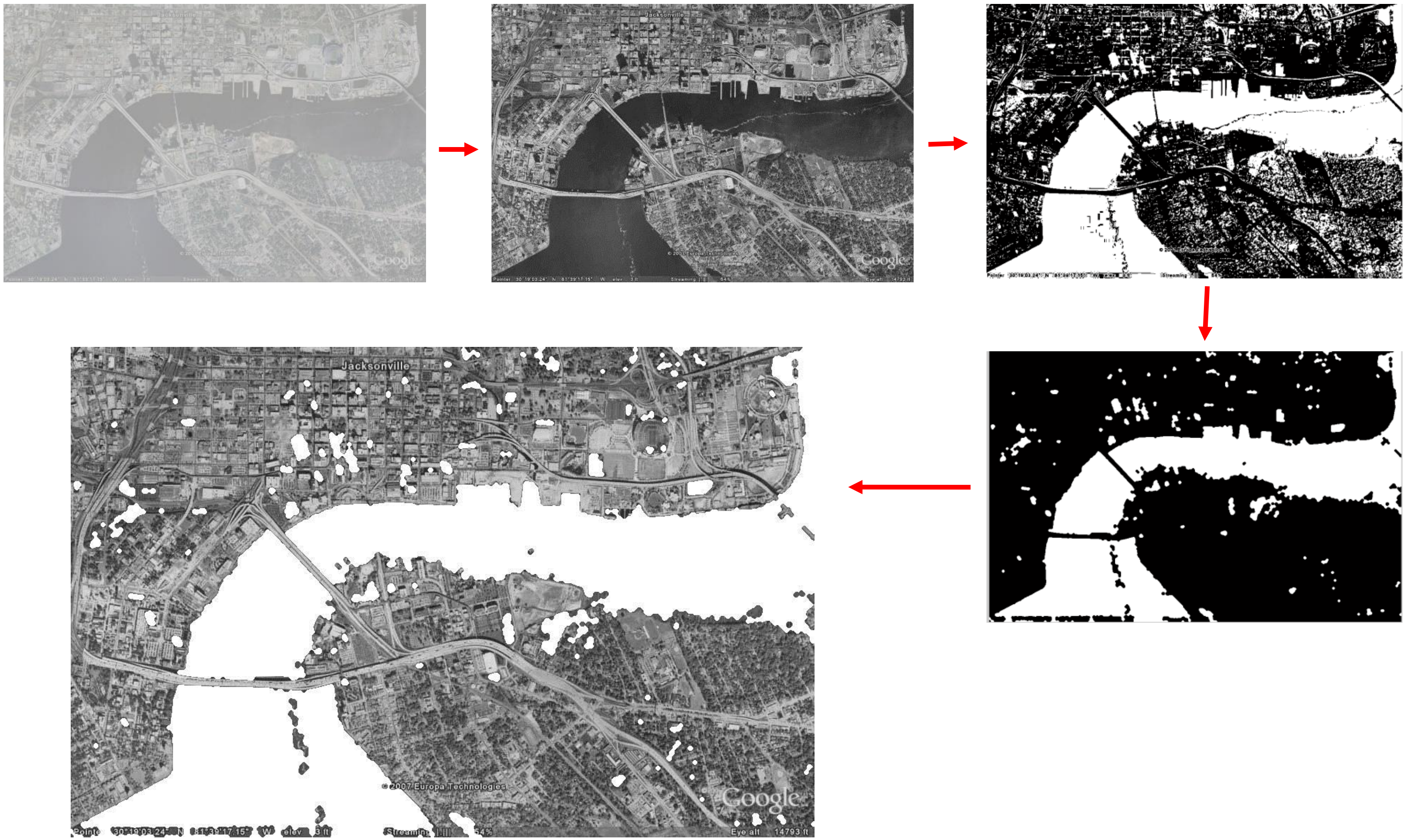
Exercise 1

Topic: basic information extraction

- Extract “regions of interest” from an image
  - Getting familiar with MATLAB
  - Image enhancement → histogram stretching
  - Global thresholding → derive a binary image
  - Morphological operators → dilation and erosion, opening and closing

Exercise 1 (A-C)

Workflow:



Exercise 1

A: Image enhancement

- Computing a grayscale image *GI* from rgb *image*:
  - *GI = mean(image, 3); % equal weights* → ***preferred***
  - *GI = rgb2gray(image); % unequal weights*

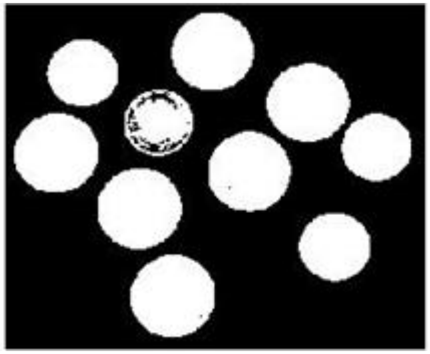
- Get the maximum value of an 2d-array:
  - *Maxi = max(max(GI));*
  - *Maxi = max(GI(:));*

- Histogram stretching:
  - *SI = (GI – Mini)/(Maxi-Mini);*

  → For-loops not necessary

Exercise 1

B: Global Thresholding



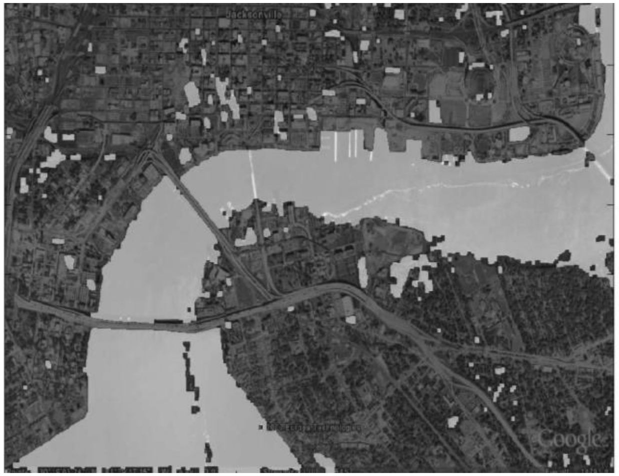
- Finding a threshold: trial and error *or* use function “*graythresh*”
- Apply threshold: using operators “<, >, <=,…” or function “*im2bw*”
- *mask = image < threshold;*

→ For-loops not necessary

Exercise 1: some results

C: Morphological filtering

- Application of opening and closing consecutively
- Watermask was desired



Matlab function *imfuse*

Summary: Binary image processing

- Pro’s:
  - Easy techniques and fast to compute
  - Binary images are easy to store
  - Can be useful in constrained scenarios with well known conditions
- Con’s:
  - Hard to extract the “clean” object silhouettes
  - Influence of noise
  - Too simple technique to solve “hard” problems

Exercise 2

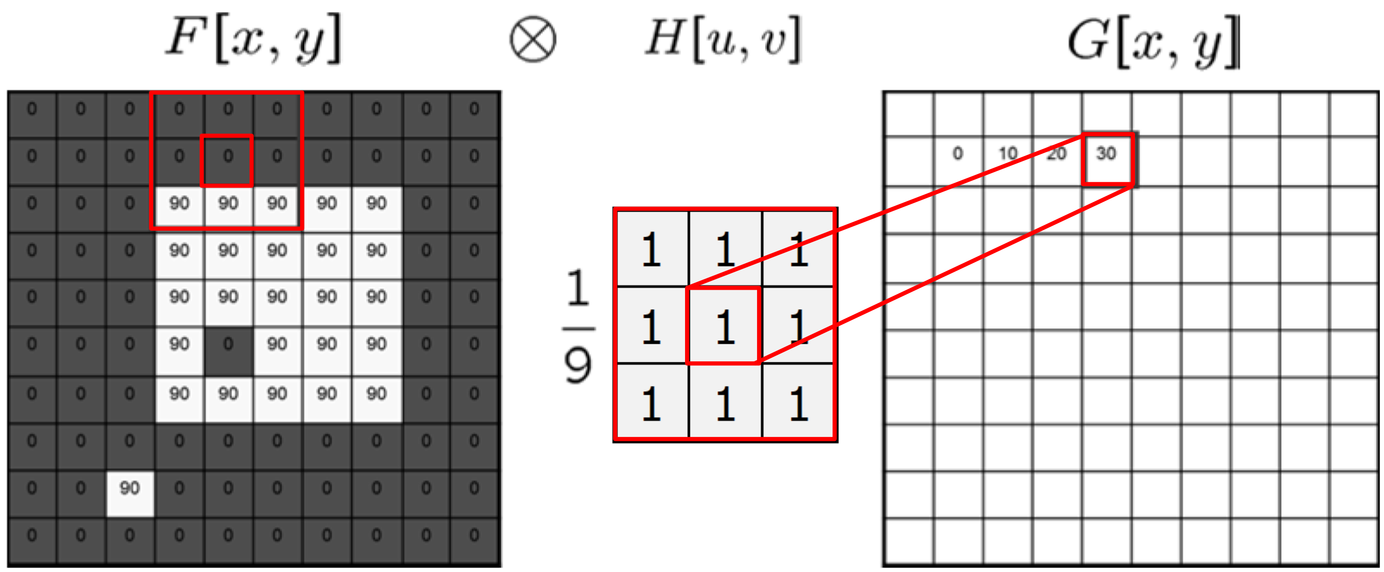
- **Task A:** Image-filtering (GoG)
- **Task B:** Interest points (Förstner)
- Aims
  - Derive edge information from an image
  - Reducing noise and derive edge information **simultaneously** using GoG-filtering
  - Use edge information to identify “points of interest” in image
- Relevant for
  - Understanding filtering
  - Edge detection and image smoothing
  - Finding corresponding points in different images

Exercise 2

Task A: Gradient of Gaussian Image-filtering

Image Filtering

- Replace each pixel with a linear combination of its neighbors
  - Filter Mask H : contains weights for the linear combination
  - Example: Moving average (image smoothing)

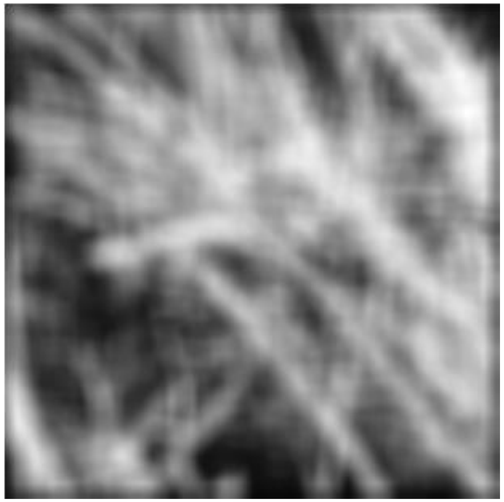


$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]\, F[i+u, j+v]$ → $k = 1$

$G = H \otimes F$ → **Cross-correlation**

Image Filtering

- Replace each pixel with a linear combination of its neighbors
  - Filter Mask H : contains weights for the linear combination
  - Example: Moving average (image smoothing)



$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]\, F[i+u, j+v]$ → $k = 1$

$G = H \otimes F$ → **Cross-correlation**

Image Filtering

- Replace each pixel with a linear combination of its neighbors
- Filter kernel H: coefficients or weights

- **Cross-correlation:**

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]\boxed{F[i+u, j+v]}$$

$$G = H \otimes F$$

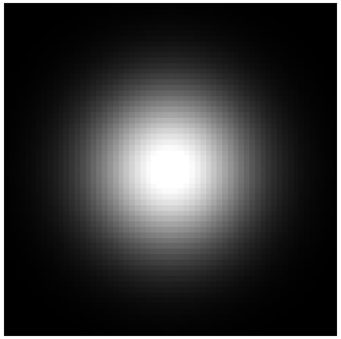
  - Check similarity of two signals
- **Convolution:**

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]\boxed{F[i-u, j-v]}$$

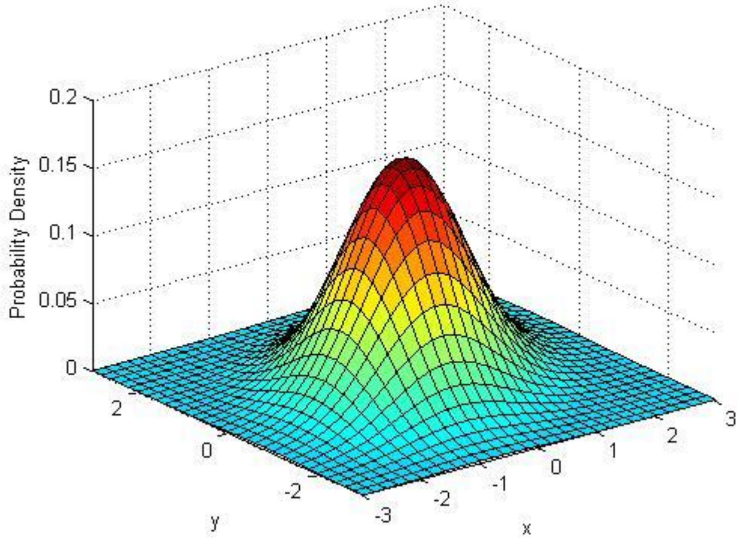
$$G = H \star F$$

  - Apply filter H on image F for: information extract or processing tasks
  - Can easily be done in frequency-domain
  - **Symmetric filter kernel → Correlation = Convolution**

2D Gaussian Filter

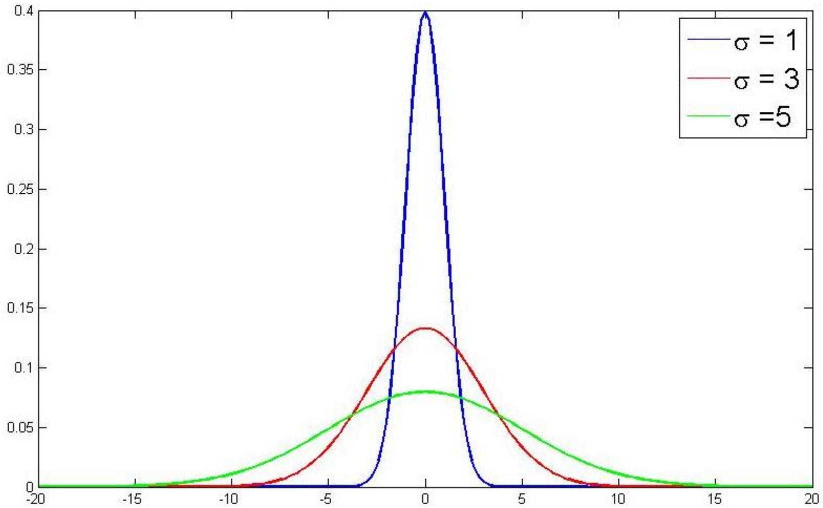


Continuous,
rotationally symmetric
weighted average



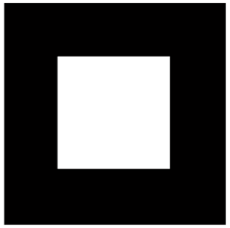
$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

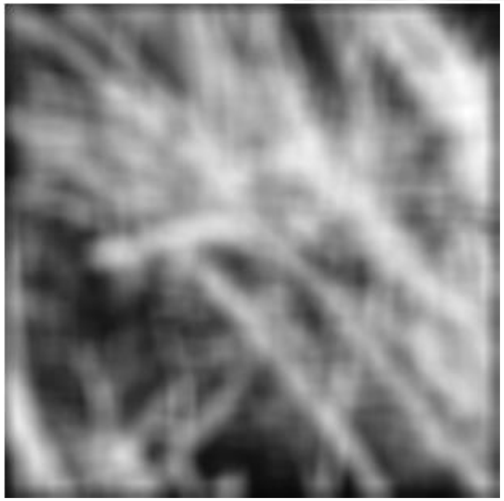
Effect of standard deviation σ

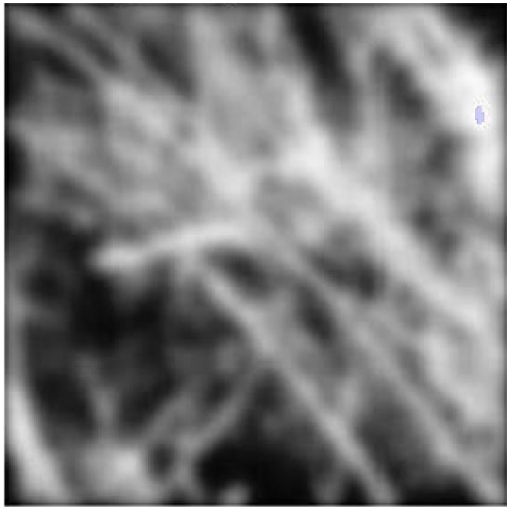


Mask radius $k = 2\sqrt{2}\sigma \approx |3\sigma|$

2D Gaussian Filter







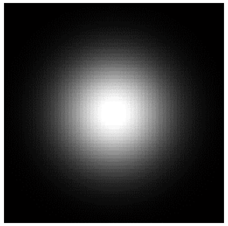
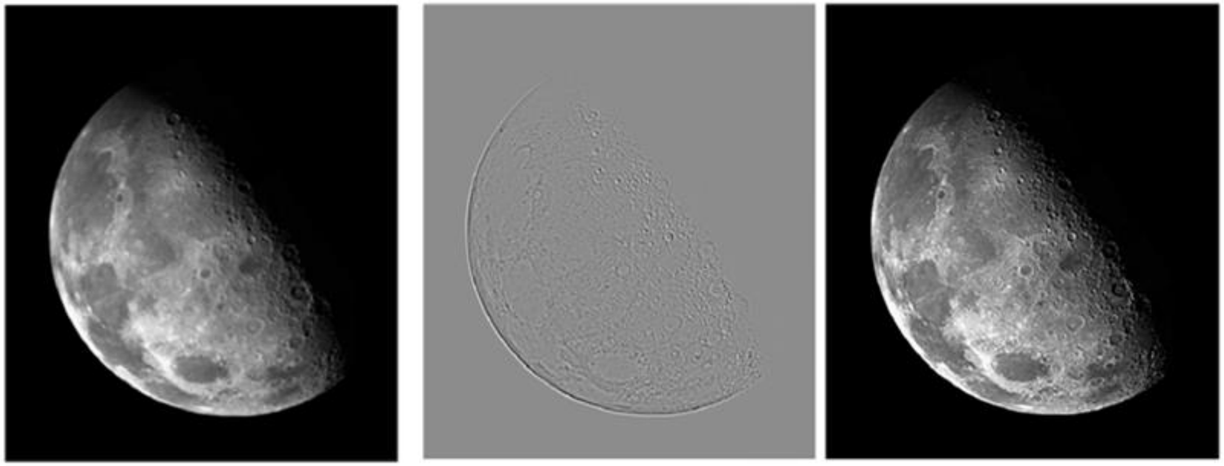


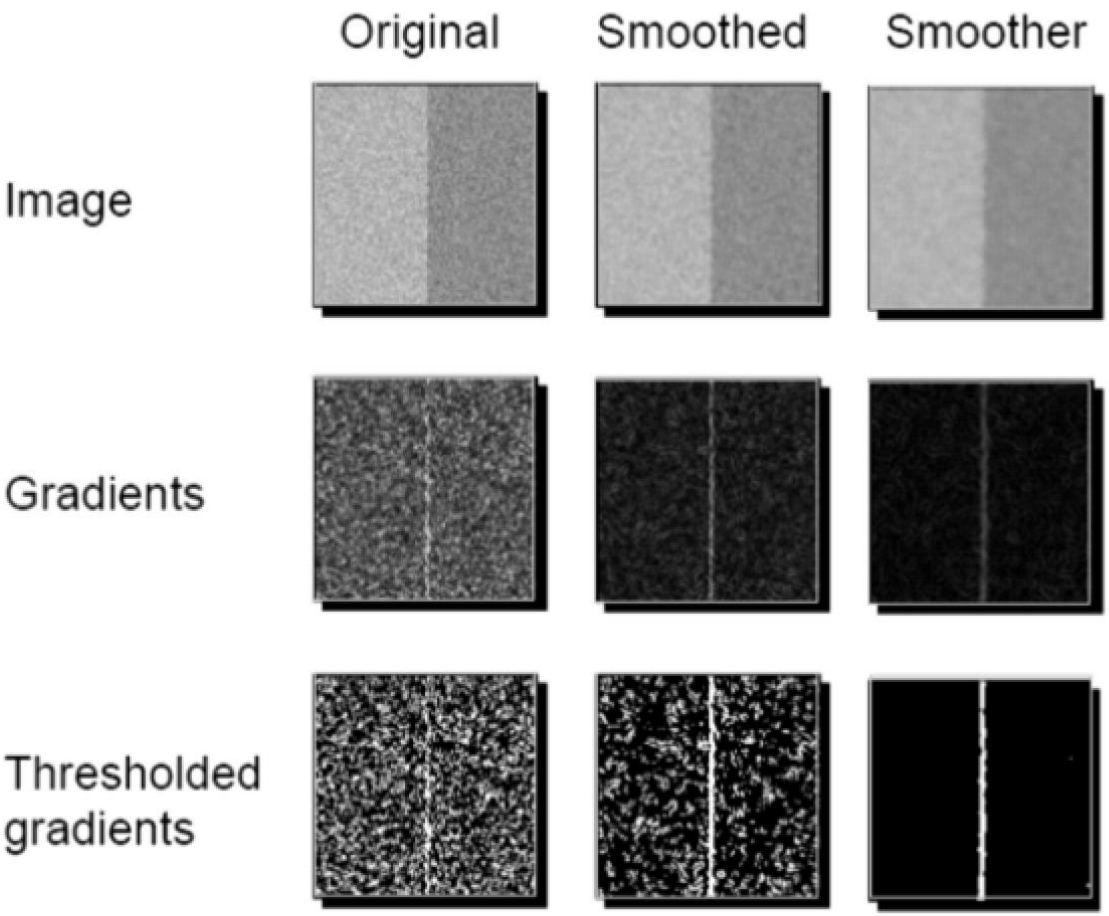
Image Sharpening

- Mean and Gaussian filter
  - Remove high-frequency components from images
  - Low-pass filter
- Smoothing → integration
- Sharpening → differentiation
  - Edge detection
  - Image enhancement



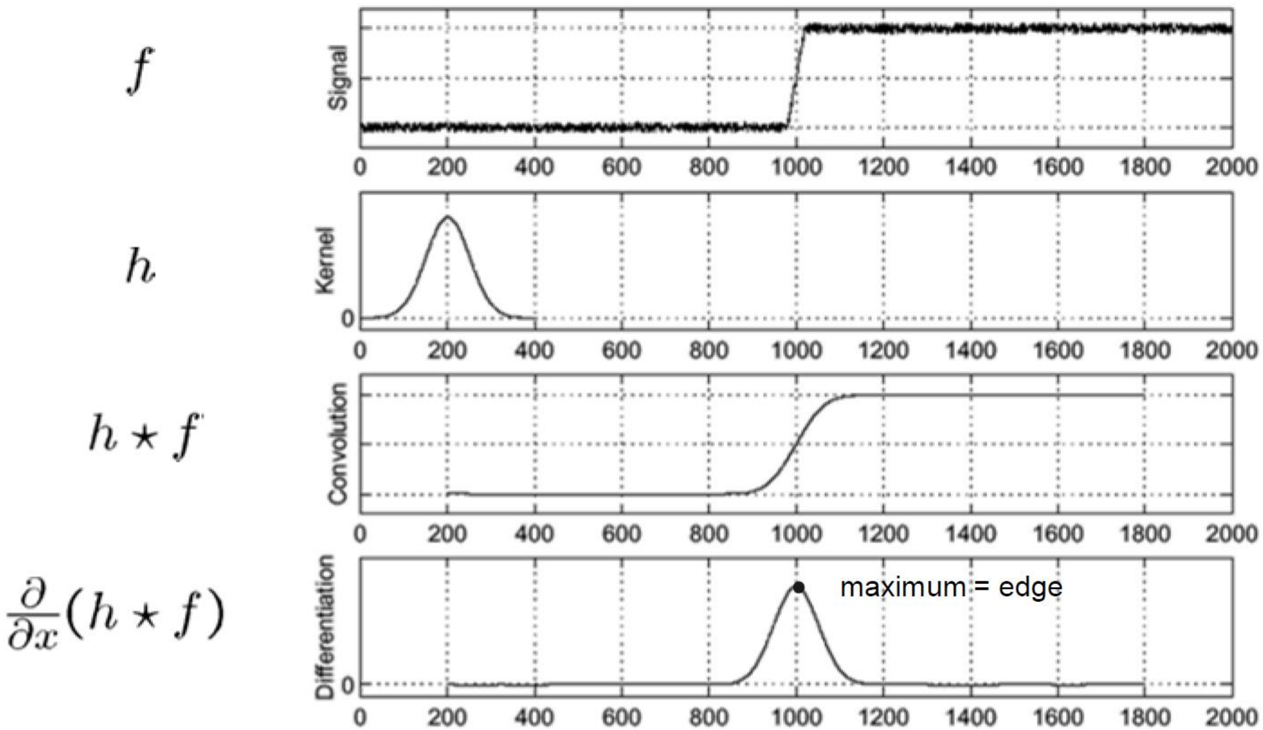
Sharpening and Smoothing

Benefits of smoothing in edge detection



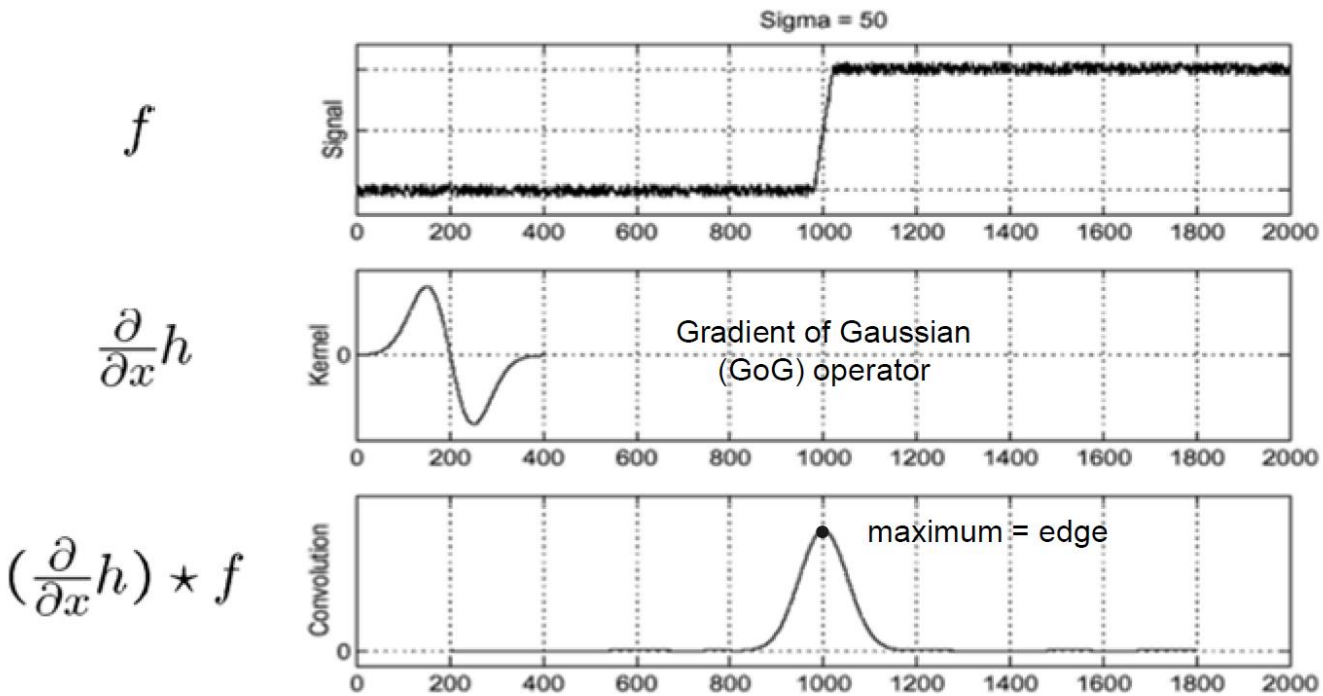
Sharpening and Smoothing

- Smoothing before computing the differentiation
  - → Two independent filter operations (convolutions)



Sharpening and Smoothing

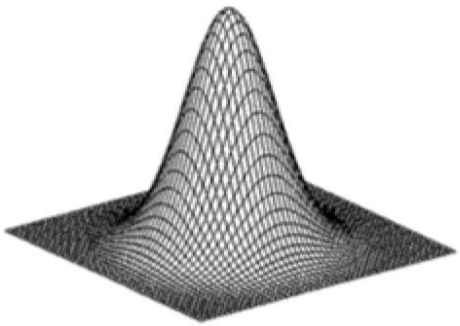
- Differentiation property of convolution: $\frac{\partial}{\partial x}(h \star f) = \left(\frac{\partial h}{\partial x}\right) \star f$

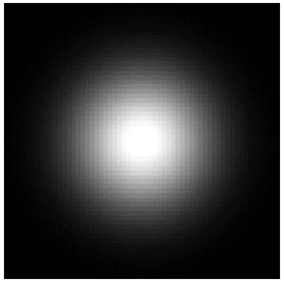


2D GoG filtering

- Gaussian filter

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

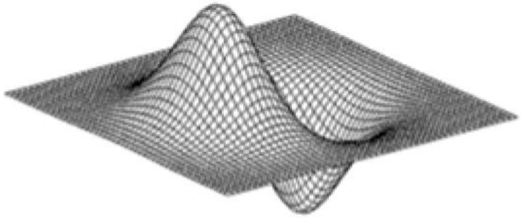


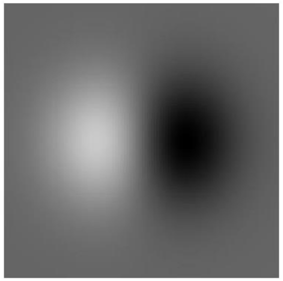


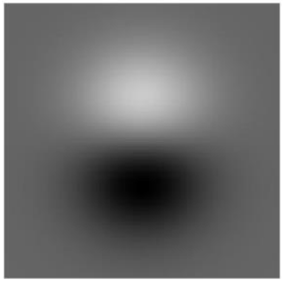
- Gradient of Gaussian

$$\frac{\partial G(x,y,\sigma)}{\partial x} = -\frac{x}{2\pi\sigma^4} exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

$$\frac{\partial G(x,y,\sigma)}{\partial y} = -\frac{y}{2\pi\sigma^4} exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$







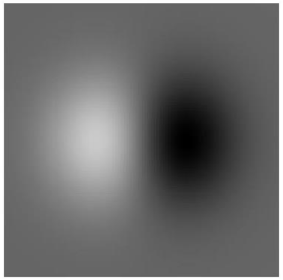
2D GoG filter computation

$$\frac{\partial G(x,y,\sigma)}{\partial x} = -\frac{x}{2\pi\sigma^4} exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$

1) Define standard deviation, e.g. $\sigma = 0.5$

2) "Size" of filter kernel from center pixel: $r = |3 \cdot \sigma| = 2.0$

3)
$$c_x = \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \\ -2 & -1 & 0 & 1 & 2 \end{bmatrix}; \quad c_y = c_x^T$$



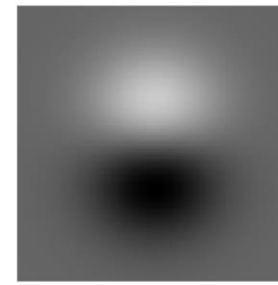
4) Compute filter using c_x and c_y for x and y

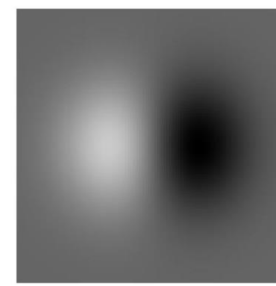
$$G_x = \frac{\partial G(x,y,\sigma)}{\partial x} = \begin{bmatrix} 0.0000 & 0.0001 & 0.0000 & -0.0001 & -0.0000 \\ 0.0002 & 0.0466 & 0.0000 & -0.0466 & -0.0002 \\ 0.0017 & 0.3446 & 0.0000 & -0.3446 & -0.0017 \\ 0.0002 & 0.0466 & 0.0000 & -0.0466 & -0.0002 \\ 0.0000 & 0.0001 & 0.0000 & -0.0001 & -0.0000 \end{bmatrix}; \quad G_y = \frac{\partial G(x,y,\sigma)}{\partial y} = \frac{\partial G(x,y,\sigma)}{\partial x}^T$$

Task A: GoG filtering

Input image:

Compute grayscale image, if necessary.





a. Compute GoG-filter masks for filtering in x- and y- direction
b. Apply the two filters G_x and G_y on the input image using *for-loops* → Convolution, result: I_x and I_y
c. Compute the gradient magnitude image using equation

$$G = \sqrt{(I_x)^2 + (I_y)^2}$$

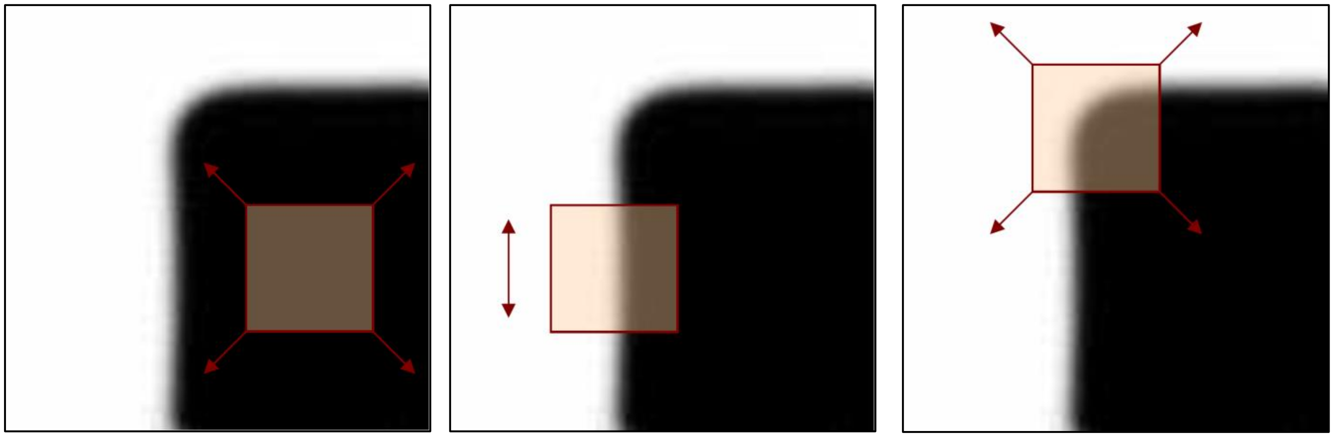
Plot and export the resulting image G (by-product!).

Exercise 2

Task B: Förstner Operator

Corners as distinctive interest points

- We should easily recognize the point by looking through a small window
- Shifting a window in any direction should give a large change in intensity



Flat region:
no changes in all
directions

Edge:
no change along
edge direction

Corner:
significant change
in all directions

Auto-correlation Matrix

- Identification of corners
- Input: First derivatives in x- and y-direction I_x and I_y (result of A.b.)

→

Grayscale image

I_x (GoG)

I_y (GoG)

Auto-correlation Matrix M

- Input Arrays: I_x^2, I_y^2 and $I_x I_y$
- Computation of M for each pixel:

$$M = \sum_{x,y \in N} w(x,y) \cdot \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = w \star \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

- w: Defines weights of local Neighborhood N

Definition: $w = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & \color{red}{1} & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \rightarrow M = \sum_{x,y \in N} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$

Auto-correlation Matrix M

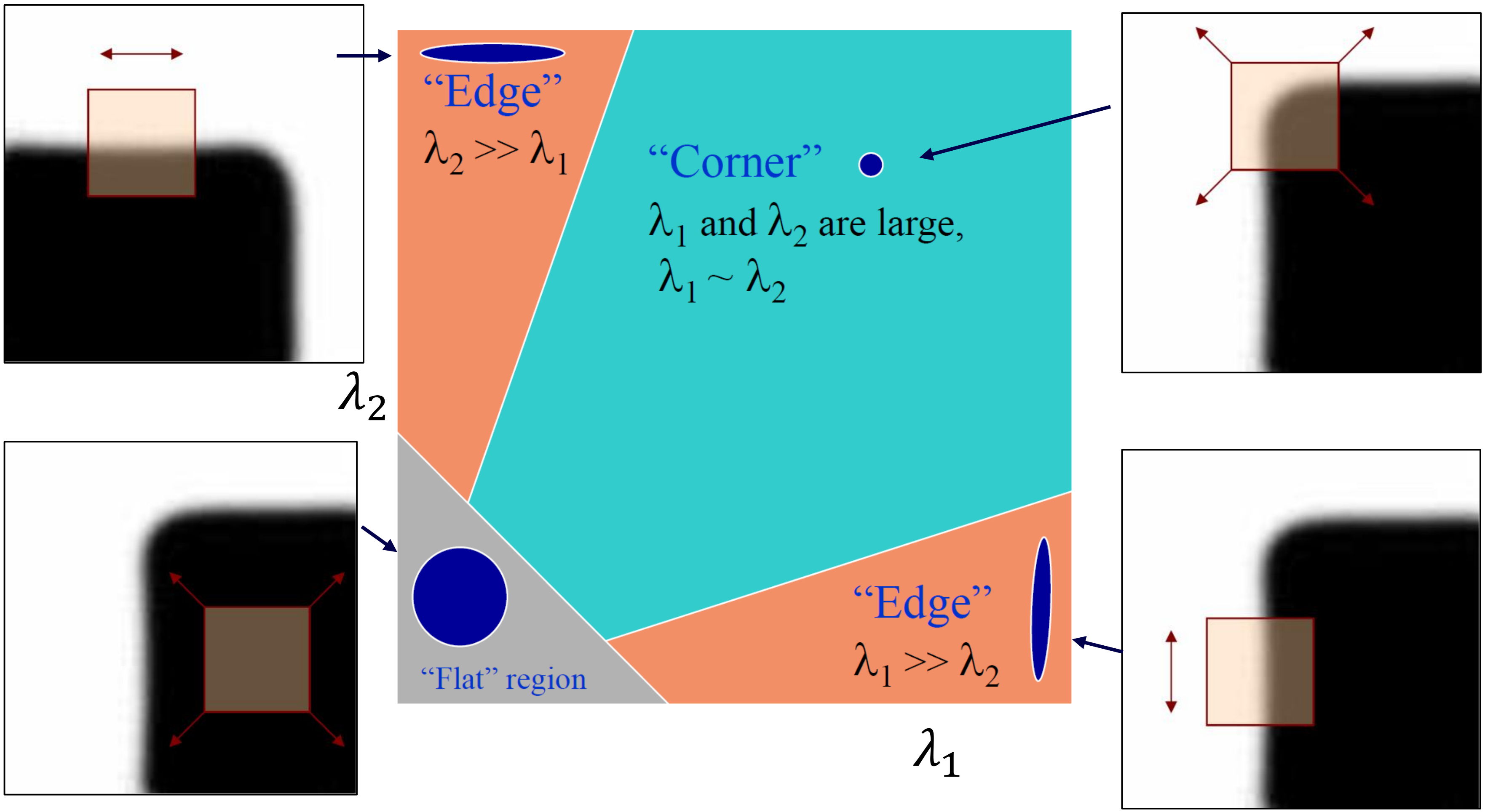
Do for each pixel in the image (except edges):

- Extract local image chip (covered by w) from I_x^2, I_y^2 and $I_x I_y$
- Compute M for each pixel:
  → summarize all local values
  → $\bar{I}_x^2 = \sum_N I_x^2$, also for $\bar{I}_y^2$ and $\bar{I}_x \bar{I}_y$
- Build M

$$M = \begin{bmatrix} \bar{I}_x^2 & \bar{I}_x \bar{I}_y \\ \bar{I}_x \bar{I}_y & \bar{I}_y^2 \end{bmatrix}$$

Auto-correlation Matrix M

Use Eigenvalues of M to detect corners



Förstner Interest Operator

- Corneness:

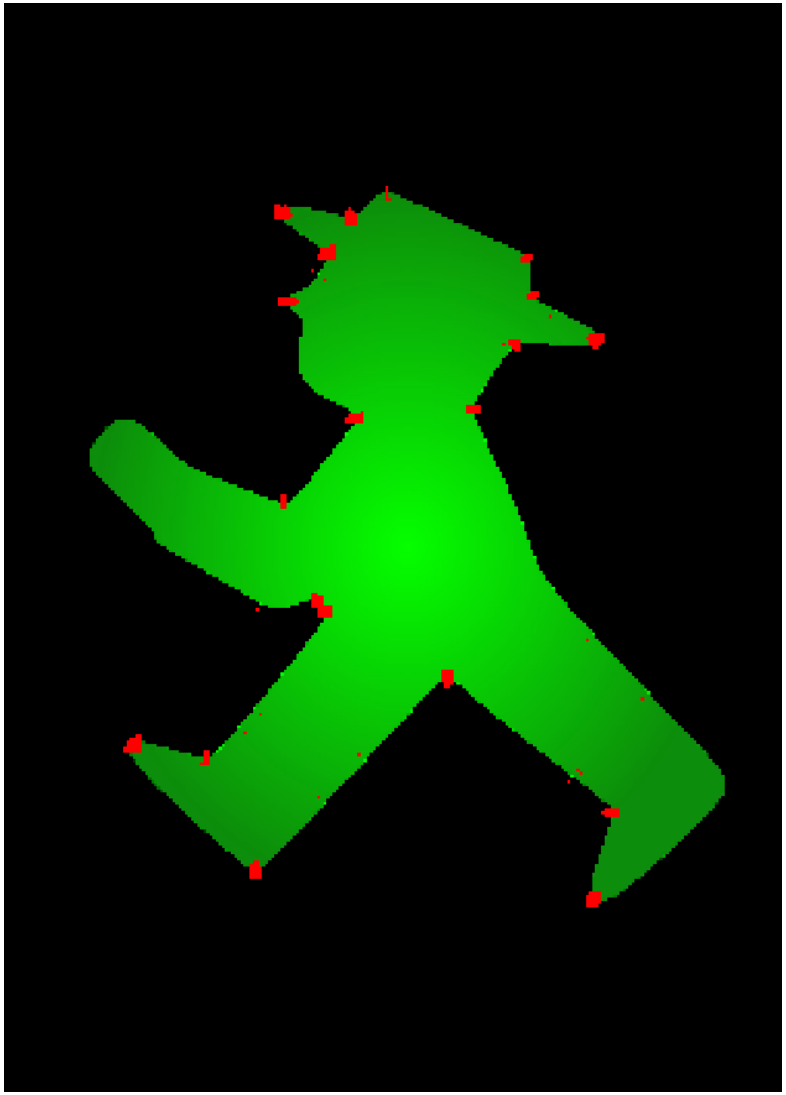
$$w = \frac{trace(M)}{2} - \sqrt{\left(\frac{trace(M)}{2}\right)^2 - det(M)}, \qquad w > 0$$

- Roundness

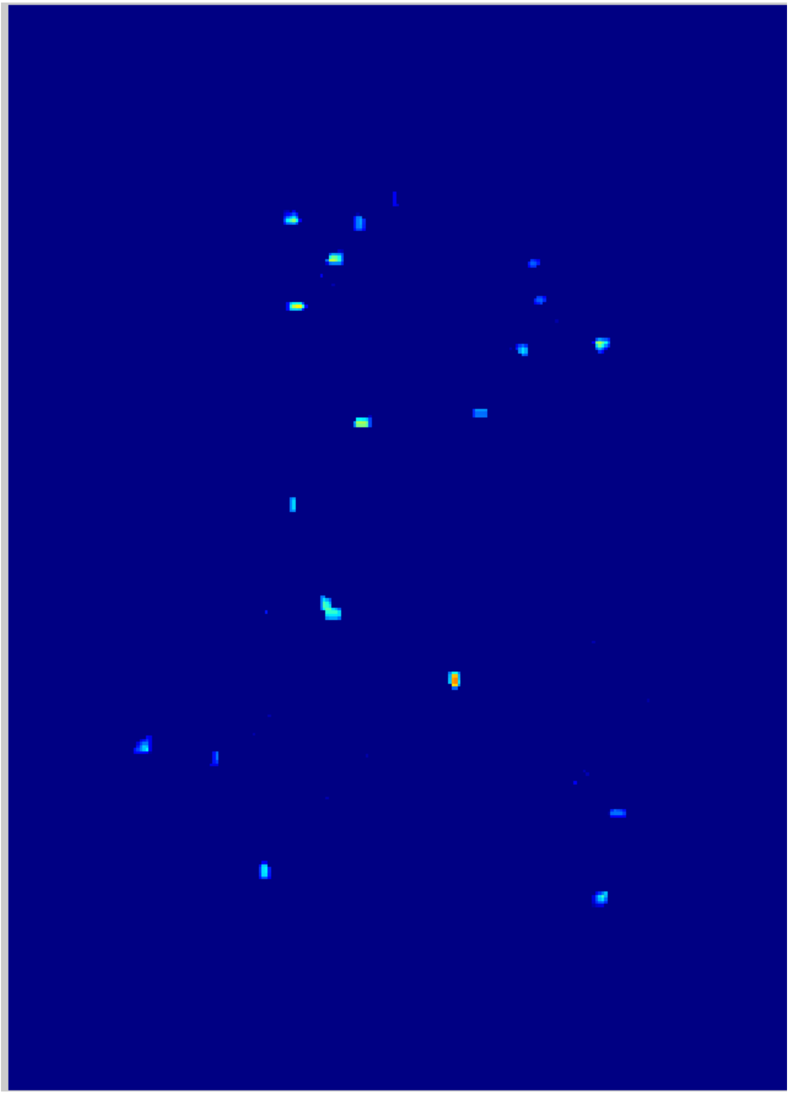
$$q = \frac{4 \cdot det(M)}{trace(M)^2}, \qquad 0 \leq q \leq 1$$

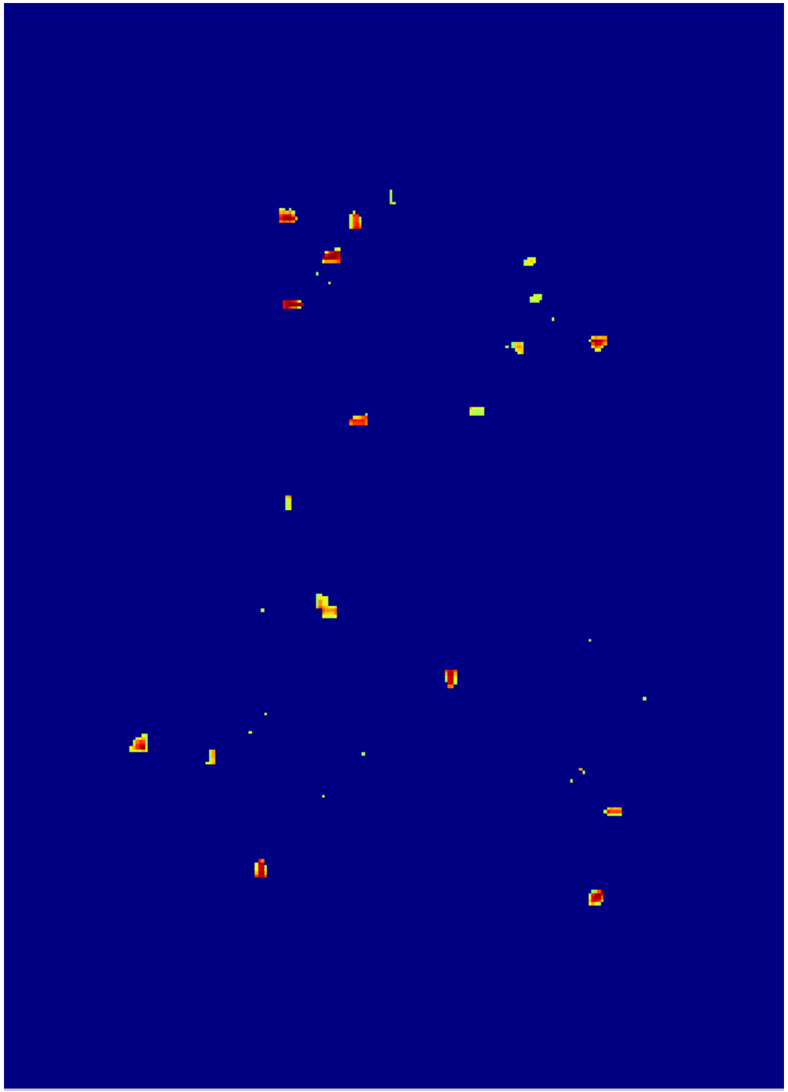
- Find corner point candidates M_C

$$M_C = w > t_w \,\&\, q > t_q$$
$$t_w = [0.5, \dots, 1.5], t_q = [0.5, \dots, 0.75]$$

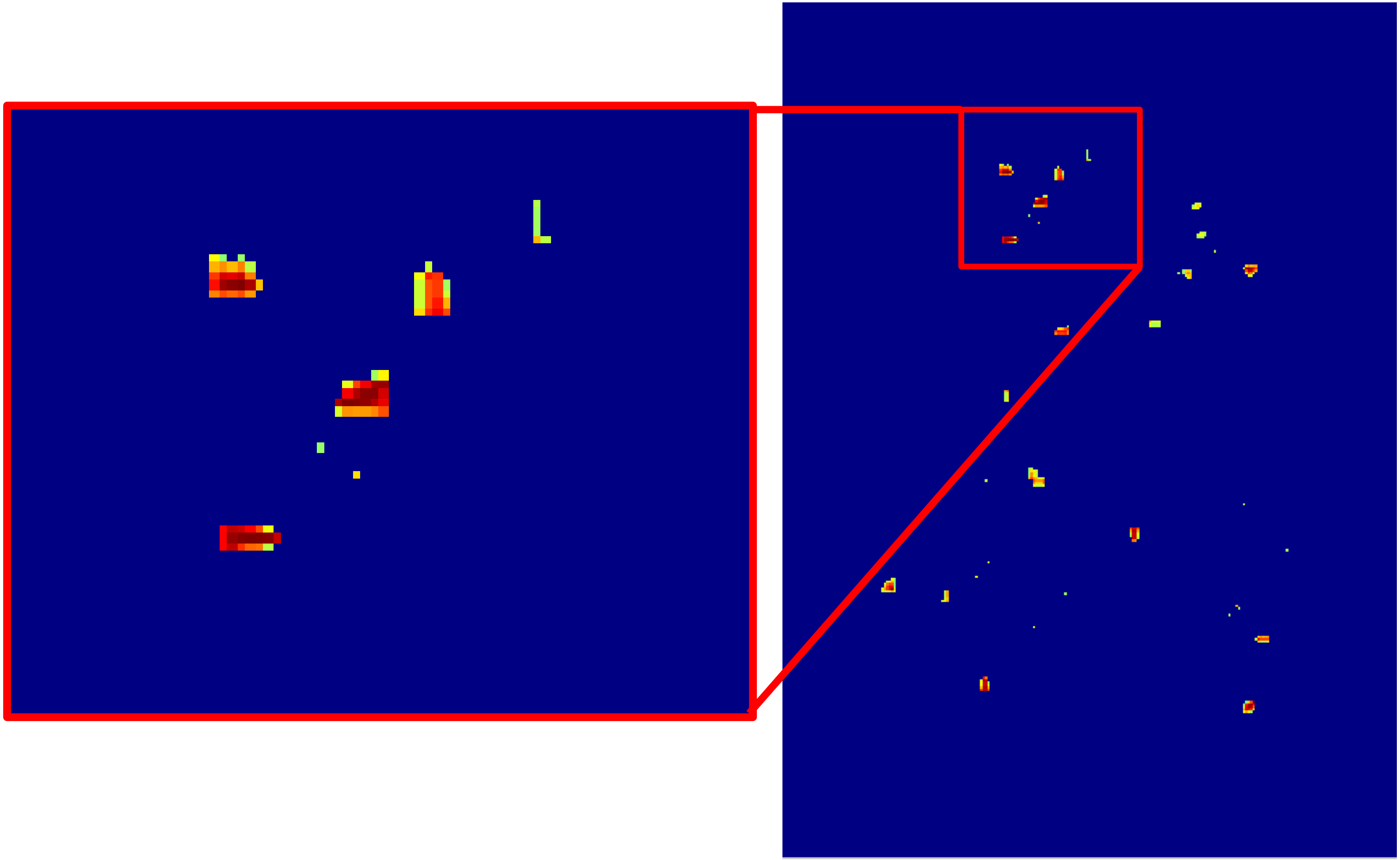


Thresholded regions of w and q



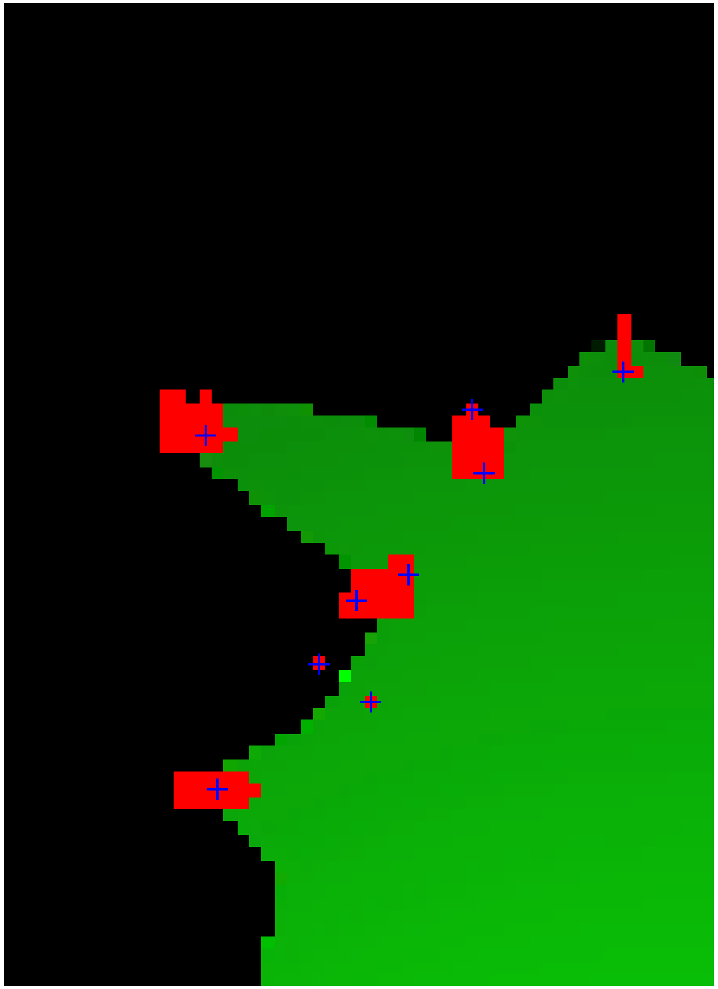


Thresholded regions of w and q



Extract interest points

- Use einter w or q:
  - Apply MATLAB function *houghpeaks* to detect local maxima
  - Output: position of peaks in image coordinates



Task B: Förstner Operator

Idea: Use GoG-images to identify Förstner points

a. Compute the autocorrelation matrix M for each pixel using a 5x5 moving window
b. Instead of storing M for each pixel, compute w and q from that matrix and store these values. Make a plot of these two arrays.
c. Derive a mask of potential interest points by simultaneously applying thresholds $t_w = 1.0$ and $t_q = 0.5$ on w and q.
d. Multiply w *or* q with the resulting mask of step c and apply the function ***houghpeaks*** to derive the coordinates of interest points.
e. Plot an overlay of the initial input image and the detected points.

Thank you!

Questions?