Bauhaus-Universität Weimar

Image Analysis and Object Recognition

Exercise 2

Summer Semester 2024

(Course materials for internal use only!)

Computer Vision in Engineering – Prof. Dr. Rodehorst
M.Sc. Mariya Kaisheva
mariya.kaisheva@uni-weimar.de

Agenda

Topics:

Assignment 1.	Image enhancement, Binarization, Morphological operators
Assignment 2.	**Gradient of Gaussian filtering, Förstner interest operator**
Assignment 3.	Shape detection based on Hough-voting
Assignment 4.	Filtering in the frequency domain, Fourier descriptors for shape recognition
Assignment 5.	Image segmentation using clustering
Assignment 6.	Convolutional neural networks for image classification
Final Project.	**-** *Will be announced during the last exercise class* **-**

Agenda

Start date and submission deadlines:

Assignment 1.	~~18.04.24 – 01.05.24~~
Assignment 2.	**02.05.24 – 15.05.24**
Assignment 3.	16.05.24 – 29.05.24
Assignment 4.	30.05.24 – 12.06.24
Assignment 5.	13.06.24 – 26.06.24
Assignment 6.	27.06.24 – 10.07.24
Final Project.	11.07.24 – 22.09.24

Wednesday by 23:00
(Central European Time)

Bauhaus-
Universität
Weimar

Assignment 1:
Sample Solution

Assignment 1: Overview

Topics:

- Image enhancement → histogram stretching
- Global Thresholding → binary mask
- Morphological operators → opening, closing

Goal:

- Getting familiar with MATLAB
- Extracting image pixels representing foreground objects
  → e.g. extraction of the water regions

Assignment 1: Workflow

Low-contrast input image

Enhanced grayscale image

Binary image

Refined binary image

Final overlay image

Bauhaus-Universität Weimar

MATLAB

```
function Assignment1a                                                    % task 4
%        ============
input = rgb2gray(imread('input_sat_image.jpg'));
improved = Enhancing(input);
output = imoverlay(improved, Filtering(Thresholding(improved)), 'white');
figure, imshow(output), title('Output image');

function output = Enhancing(input)                                       % task 1
%                 =========
figure, imshow(input),  title('Input image');
figure, imhist(input),  title('Grayvalue histogram');
output = imadjust(input);
figure, imhist(output), title('Stretched histogram');
figure, imshow(output), title('Enhanced image');

function output = Thresholding(input)                                    % task 2
%                 ============
thresh = graythresh(input);
output = imcomplement(imbinarize(input, thresh));
figure, imshow(output), title('Binary mask');

function output = Filtering(input)                                       % task 3
%                 =========
se = strel('disk', 7);
output = imclose(imopen(input, se), se);
figure, imshow(output), title('Morphological Filtering');
```

Bauhaus-
Universität
Weimar

MATLAB

```
function Assignment1a                                          % task 4
%        ============
input = rgb2gray(imread('input_sat_image.jpg'));
improved = Enhancing(input);
output = imoverlay(improved, Filtering(Thresholding(improved)), 'white');
figure, imshow(output), title('Output image');

function output = Enhancing(input)                             % task 1
%                 =========
figure, imshow(input),  title('Input image');
figure, imhist(input),  title('Grayvalue histogram');
output = imadjust(input);
figure, imhist(output), title('Stretched histogram');
figure, imshow(output), title('Enhanced image');

function output = Thresholding(input)                          % task 2
%                 ============
thresh = graythresh(input);
output = imcomplement(imbinarize(input, thresh));
figure, imshow(output), title('Binary mask');

function output = Filtering(input)                             % task 3
%                 =========
se = strel('disk', 7);
output = imclose(imopen(input, se), se);
figure, imshow(output), title('Morphological Filtering');
```

convenience function also available in Octave

(However, for this assignment we asked you to implement contrast stretching on your own!)

Bauhaus-Universität Weimar

MATLAB

```
function Assignment1a                                          % task 2
%        ============
input = rgb2gray(imread('input_sat_image.jpg'));
improved = Enhancing(input);
output = imoverlay(improved, Filtering(Thresholding(improved)), 'white');
figure, imshow(output), title('Output image');

function output = Enhancing(input)                             % task 1
%                 =========
figure, imshow(input),  title('Input image');
figure, imhist(input),  title('Grayvalue histogram');
output = imadjust(input);
figure, imhist(output), title('Stretched histogram');
figure, imshow(output), title('Enhanced image');

function output = Thresholding(input)                          % task 2
%                 ============
thresh = graythresh(input);
output = imcomplement(imbinarize(input, thresh));
figure, imshow(output), title('Binary mask');

function output = Filtering(input)                             % task 3
%                 =========
se = strel('disk', 7);
output = imclose(imopen(input, se), se);
figure, imshow(output), title('Morphological Filtering');
```

The equivalent function in Octave is **im2bw()**

Octave

```
function Assignment1b                                          % task 4
%        ============
input = double(imread('input_sat_image.jpg')) / 255;
improved = Enhancing(mean(input, 3));
mask = Filtering(Thresholding(improved)); improved(mask) = 1;
figure, imshow(improved), title('Output image');

function output = Enhancing(input)                             % task 1
%                 =========
figure, imshow(input),  title('Input image');
figure, imhist(input),  title('Grayvalue histogram');
mn = min(min(input)); mx = max(input(:));
output = (input - mn) / (mx - mn);
figure, imhist(output), title('Stretched histogram');
figure, imshow(output), title('Enhanced image');

function output = Thresholding(input)                          % task 2
%                 ============
thresh = graythresh(input);
output = ~(input > thresh);
figure, imshow(output), title('Binary mask');

function output = Filtering(input)                             % task 3
%                 =========
se = ones(15, 15);
output = imclose(imopen(input, se), se);
figure, imshow(output), title('Morphological Filtering');
```

Bauhaus-
Universität
Weimar

Octave

```
function Assignment1b                                          % task 4
%        ============
input = double(imread('input_sat_image.jpg')) / 255;
improved = Enhancing(mean(input, 3));
mask = Filtering(Thresholding(improved)); improved(mask) = 1;
figure, imshow(improved), title('Output image');

function output = Enhancing(input)                             % task 1
%                 =========
figure, imshow(input),  title('Input image');
figure, imhist(input),  title('Grayvalue histogram');
mn = min(min(input)); mx = max(input(:));
output = (input - mn) / (mx - mn);
figure, imhist(output), title('Stretched histogram');
figure, imshow(output), title('Enhanced image');

function output = Thresholding(input)                          % task 2
%                 ============
thresh = graythresh(input);
output = ~(input > thresh);
figure, imshow(output), title('Binary mask');

function output = Filtering(input)                             % task 3
%                 =========
se = ones(15, 15);
output = imclose(imopen(input, se), se);
figure, imshow(output), title('Morphological Filtering');
```

work with **double values** to avoid rounding errors

Bauhaus-Universität Weimar

Octave

```
function Assignment1b                                          % task 4
%        ============
input = double(imread('input_sat_image.jpg')) / 255;
improved = Enhancing(mean(input, 3));
mask = Filtering(Thresholding(improved)); improved(mask) = 1;
figure, imshow(improved), title('Output image');

function output = Enhancing(input)                             % task 1
%                 =========
figure, imshow(input),  title('Input image');
figure, imhist(input),  title('Grayvalue histogram');
mn = min(min(input)); mx = max(input(:));
output = (input - mn) / (mx - mn);
figure, imhist(output), title('Stretched histogram');
figure, imshow(output), title('Enhanced image');

function output = Thresholding(input)                          % task 2
%                 ============
thresh = graythresh(input);
output = ~(input > thresh);
figure, imshow(output), title('Binary mask');

function output = Filtering(input)                             % task 3
%                 =========
se = ones(15, 15);
output = imclose(imopen(input, se), se);
figure, imshow(output), title('Morphological Filtering');
```

same result with different syntax

Bauhaus-Universität Weimar

Octave

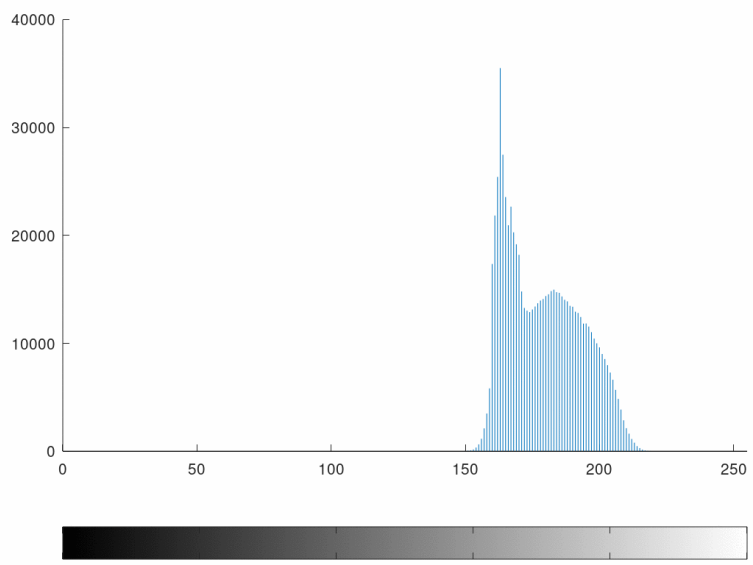
```
function Assignment1b                                          % task 4
%        ============
input = double(imread('input_sat_image.jpg')) / 255;
improved = Enhancing(mean(input, 3));
mask = Filtering(Thresholding(improved)); improved(mask) = 1;
figure, imshow(improved), title('Output image');

function output = Enhancing(input)                             % task 1
%                 =========
figure, imshow(input),  title('Input image');
figure, imhist(input),  title('Grayvalue histogram');
mn = min(min(input)); mx = max(input(:));
output = (input - mn) / (mx - mn);
figure, imhist(output), title('Stretched histogram');
figure, imshow(output), title('Enhanced image');

function output = Thresholding(input)                          % task 2
%                 ============
thresh = graythresh(input);
output = ~(input > thresh);
figure, imshow(output), title('Binary mask');

function output = Filtering(input)                             % task 3
%                 =========
se = ones(15, 15);
output = imclose(imopen(input, se), se);
figure, imshow(output), title('Morphological Filtering');
```

logical values →

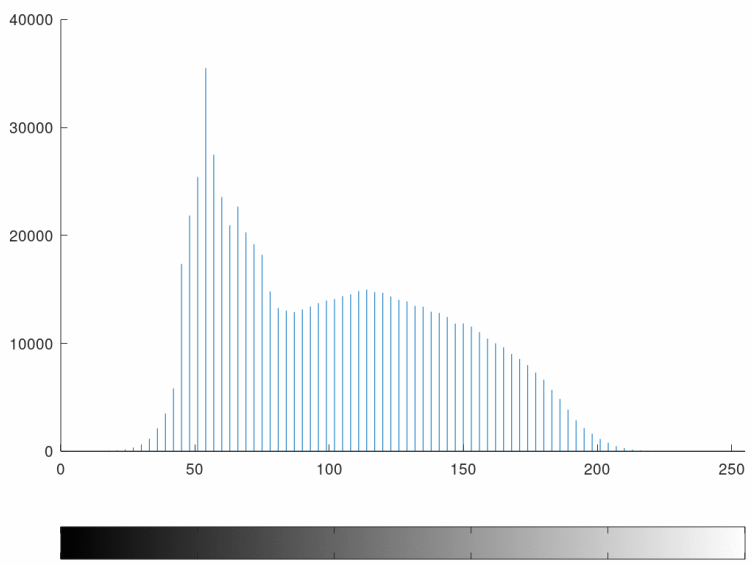
Bauhaus-Universität Weimar

Assignment 1 – sample results

Initial Histogram



Histogram after Enhancement



Assignment 1 – sample results

Enhanced grayscale image

Influence of the size of the structuring element
on the refined binary mask

SE: disk, 5

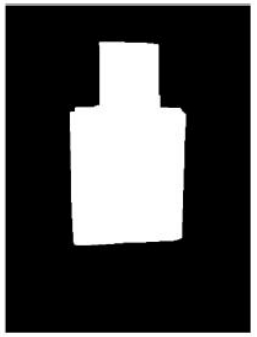
SE: disk, 7

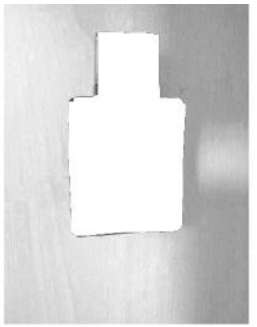
SE: disk, 10

Bauhaus-
Universität
Weimar

Assignment 1 – selected submission results

```
2   image=enhance(0,15);

63  SE = strel ("square", 3);
64  newImage=imopen (image, SE);
65  %structural element shape and size for Closing.
66  SE = strel ("square", 35);
67  newImage=imclose(newImage,SE);
```

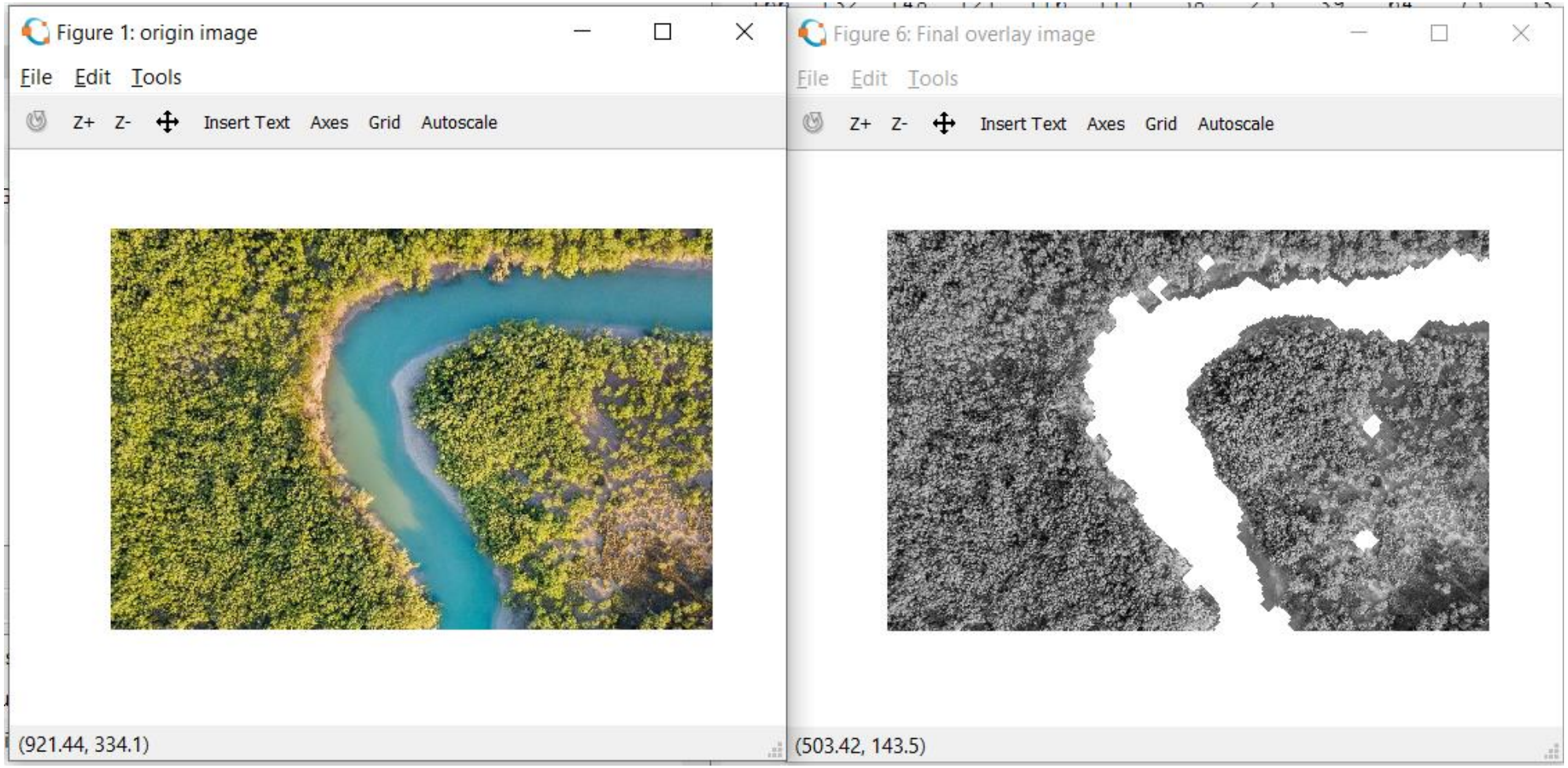




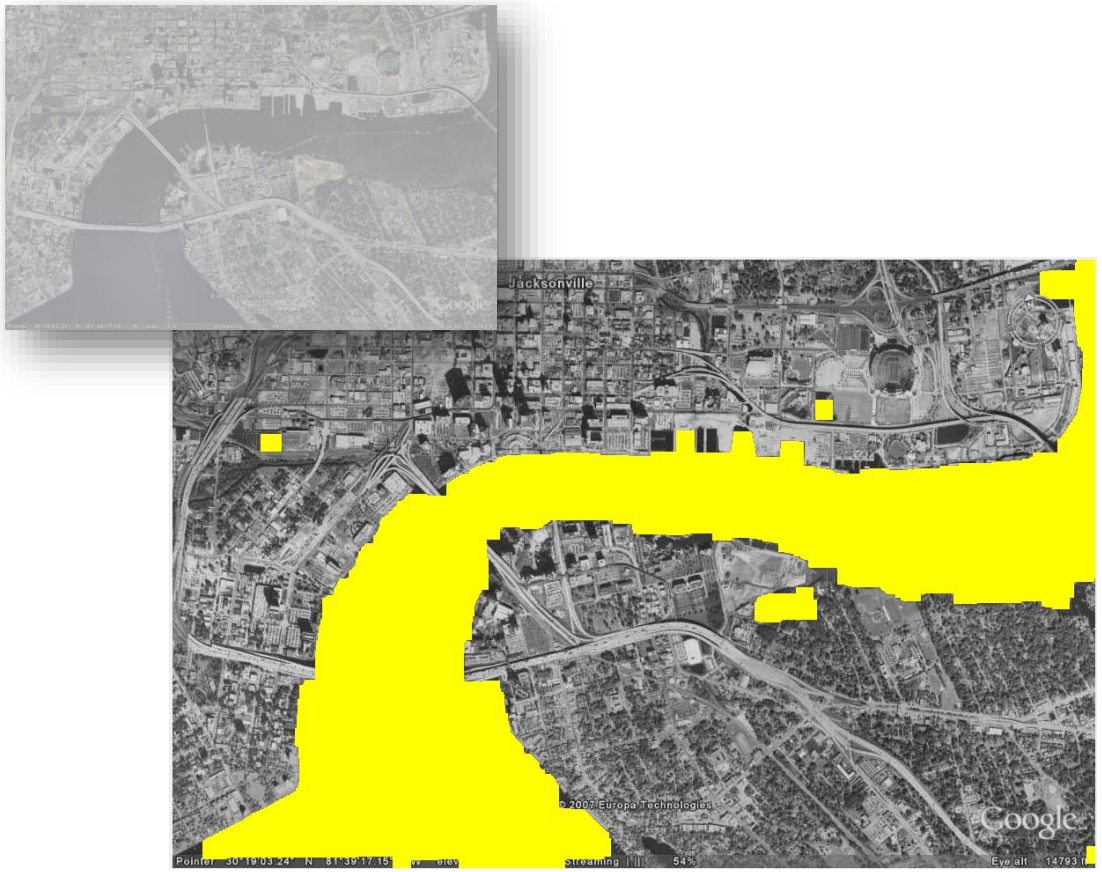
```
2   image=enhance(0,0);

63  SE = strel ("square", 2);
64  newImage=imopen (image, SE);
65  %structural element shape and size for Closing.
66  SE = strel ("square", 2);
67  newImage=imclose(newImage,SE);
```

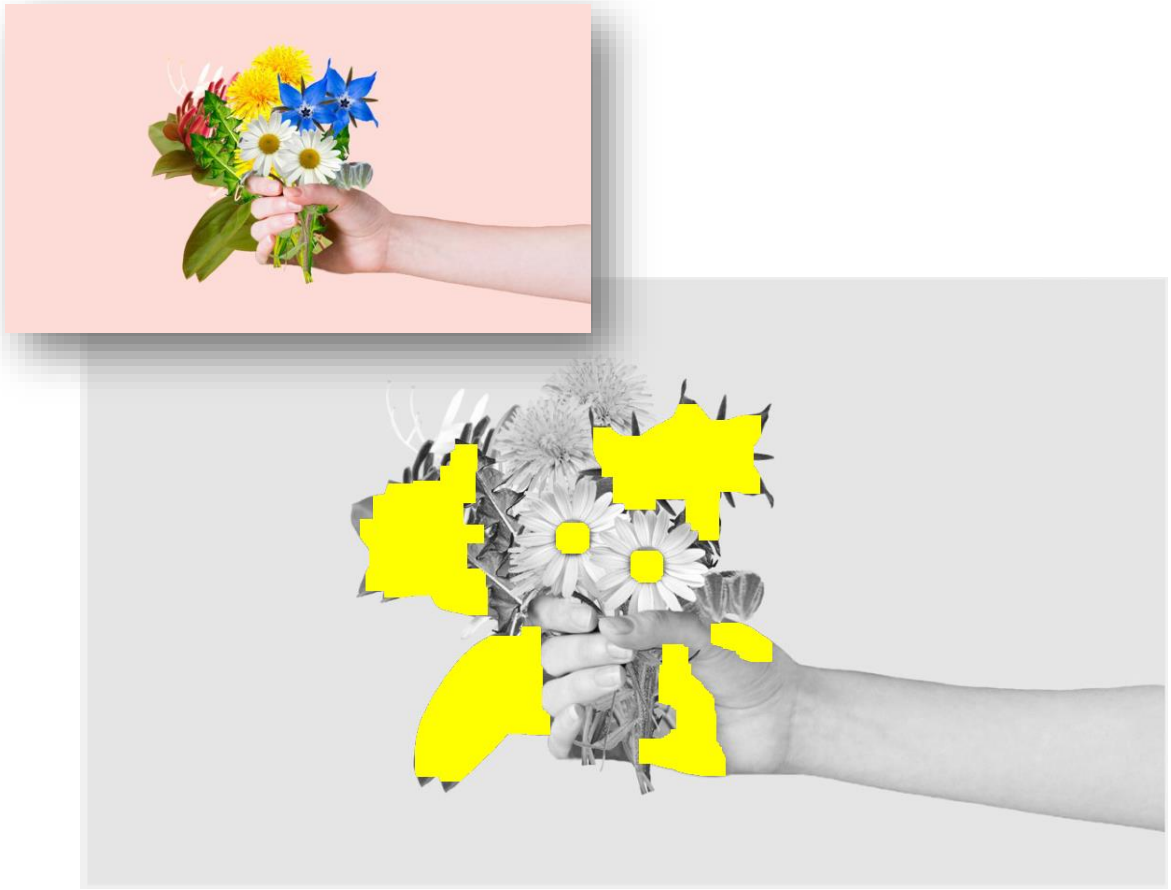
Bauhaus-Universität Weimar

Assignment 1 – selected submission results



Assignment 1 – selected submission results

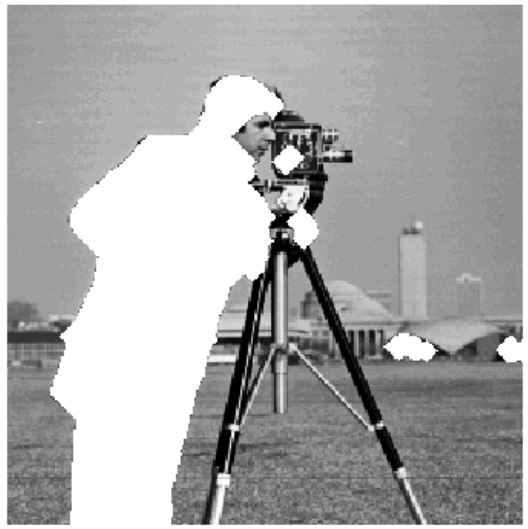




Assignment 1 – selected submission results







Bauhaus-
Universität
Weimar

Assignment 2

Assignment 2: Overview

Topics:

- Image filtering with Gradient of Gaussian (GoG)
- Interest points

Goal:

- Learn how to perform image filtering
- Practice reducing noise and **simultaneously** deriving image gradients (intensity changes)
- Practice identifying points of interest with the help of image gradients

Assignment 2: Overview

Input:

- Provided image → *ampelmaennchen.png*
- Or a different image of your own choice

Tasks:

- **A:** Gradient of Gaussian (GoG) filtering
- **B:** Förstner operator

The topics of GoG filtering and Förstner operator will be covered in detail during **lecture number 5**.

Provided input image

Assignment 2: Task A

GoG filtering:

a. Compute **GoG-filter kernels** for filtering in *x*- and *y*- direction

b. Apply the two filters G_x and G_y on the input image *I* using convolution to derive **two gradient images I_x and I_y**

c. Compute and visualize the **gradient magnitude**

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

Input image I

Grayscale converted I

Note:
Compute grayscale image and scale it to double [0.0 , 1.0].

Assignment 2: Task A

GoG filtering:

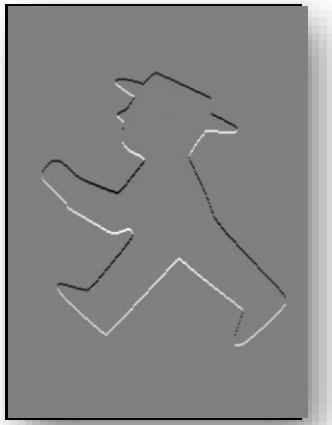
a. Compute **GoG-filter kernels** for filtering in *x*- and *y*- direction

b. Apply the two filters G_x and G_y on the input image *I* using convolution to derive **two gradient images I_x and I_y**

c. Compute and visualize the **gradient magnitude**

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

Gradient image I_x



Gradient image I_y

Assignment 2: Task A

GoG filtering:

a. Compute **GoG-filter kernels** for filtering in *x*- and *y*- direction

b. Apply the two filters G_x and G_y on the input image *I* using convolution to derive **two gradient images I_x and I_y**

c. Compute and visualize the **gradient magnitude**

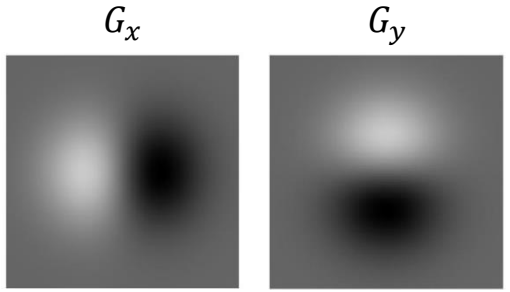
$$G = \sqrt{{I_x}^2 + {I_y}^2}$$

Gradient magnitude

2D GoG filter computation

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

general formula for computation of G_x

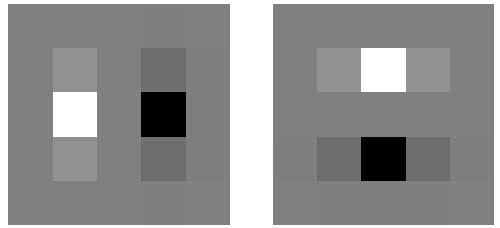


general GoG filters in x and y directions

$$G_y = {G_x}^T$$

$$G_x = \begin{bmatrix} 0.0000 & 0.0001 & 0.0000 & -0.0001 & -0.0000 \\ 0.0002 & 0.0466 & 0.0000 & -0.0466 & -0.0002 \\ 0.0017 & 0.3446 & 0.0000 & -0.3446 & -0.0017 \\ 0.0002 & 0.0466 & 0.0000 & -0.0466 & -0.0002 \\ 0.0000 & 0.0001 & 0.0000 & -0.0001 & -0.0000 \end{bmatrix}$$

numerical example with $\sigma = 0.5$



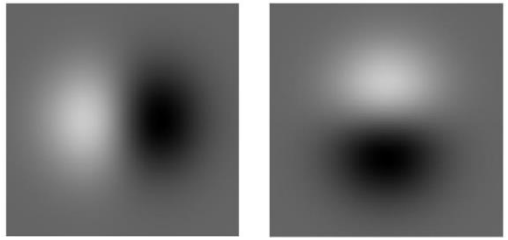
5x5 pixel GoG filters in x and y directions

2D GoG filter computation

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

general formula for computation of G_x

G_x G_y



$$G_y = {G_x}^T$$

1) Define standard deviation, e.g. $\sigma = 0.5$
2) Filter kernel radius: $r = |3 \cdot \sigma| = 2.0$
3) Define two arrays c_x and c_y with $(r \cdot 2 + 1)$ columns and rows for centered local coordinates

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

4) Compute filter using c_x and c_y for x and y $\rightarrow \frac{\partial G(x,y,\sigma)}{\partial x} = -\frac{c_x}{2\pi\sigma^4} exp\left(-\frac{({c_x}^2 + {c_y}^2)}{2\sigma^2}\right)$

Assignment 2: Task B

Förstner interest operator:

a. Compute the **autocorrelation matrix $\boldsymbol{M}$** for each pixel using a 5×5 moving window

b. Instead of storing M for each pixel, compute the **cornerness $\boldsymbol{w}$** and **roundness $\boldsymbol{q}$** from M and store these values in matrices W and Q. Plot these arrays.

c. Derive a **binary mask $\boldsymbol{M_c}$** of potential interest points by simultaneously applying thresholds, e.g. $t_w = 0.004$ and $t_q = 0.5$, on W and Q

d. Plot an overlay of the initial input image and the **detected points**

Auto-correlation Matrix M

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

→

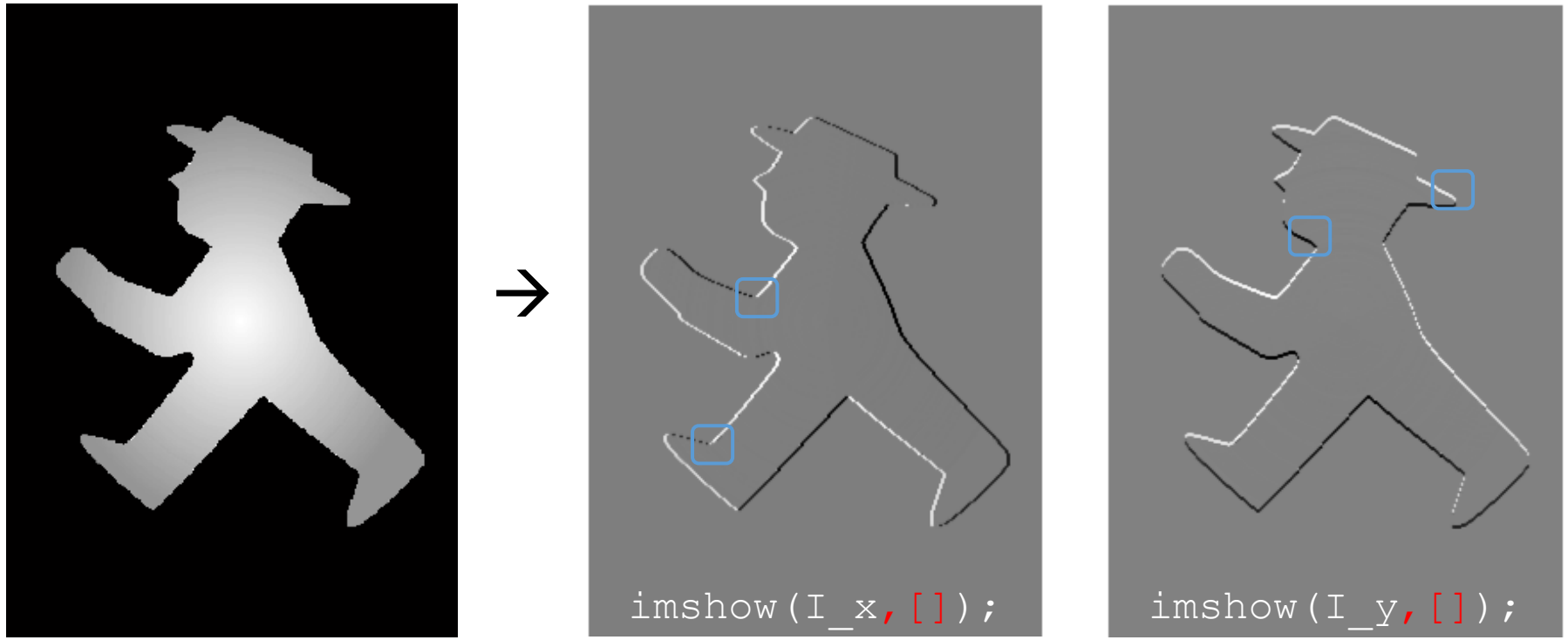
Grayscale image

I_x (GoG)

I_y (GoG)

Auto-correlation Matrix M

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



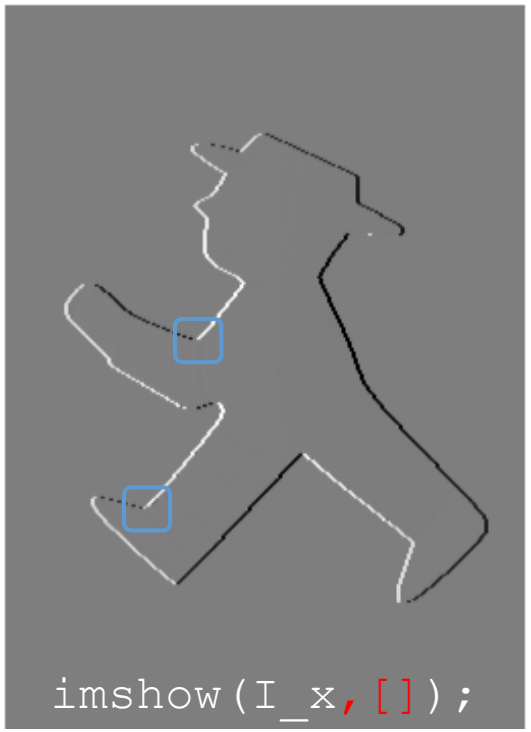
Grayscale image I_x (GoG) I_y (GoG)

Auto-correlation Matrix M

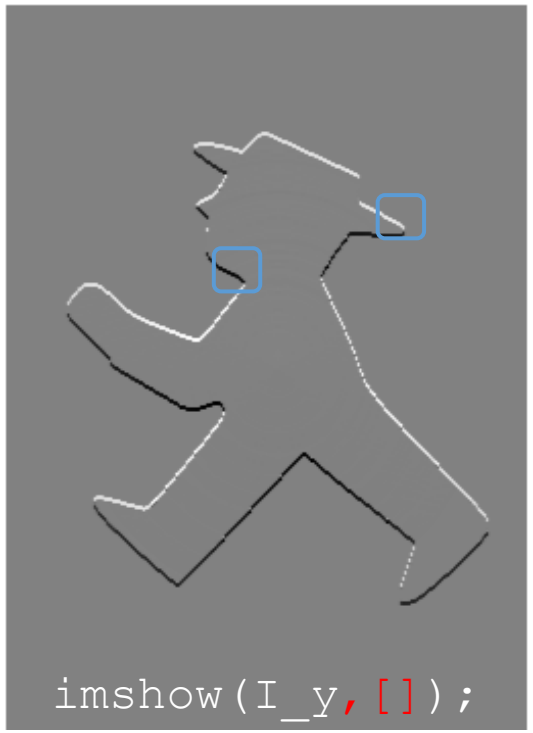
- Identification of corners
- Input: First order derivatives in x- and y-direction I_x and I_y (i.e. the result of Task A.b.)

Grayscale image

→



I_x (GoG)



I_y (GoG)

→

I_x^2, I_y^2
and
$I_x I_y$
3 arrays
necessary for the next steps

Auto-correlation Matrix M

Computation of M for each pixel:

Definition: $w_N = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$ → weights for the local neighborhood N

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

→ $M = \sum_{x,y \in N} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$ M contains the sum of all values of I_x^2, I_y^2 and $I_x I_y$ in the local neighborhood N of 5x5 pixels

Auto-correlation Matrix M

For each pixel in the image (except boundaries)**:**

1) Extract local image window w_N for I_x^2, I_y^2 and $I_x I_y$

2) Compute M :

→ sum up extracted values I_x^2, I_y^2 and $I_x I_y$

→ $\bar{I}_x^2 = \sum_N I_x^2$, also for $\bar{I}_y^2$ and $\bar{I}_x \bar{I}_y$

3) Build M (2×2 matrix) for each pixel

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

Or: convolve I_x^2, I_y^2 and $I_x I_y$ with w_N and then compute M for each pixel

Auto-correlation Matrix M

Cornerness:

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

Roundness:

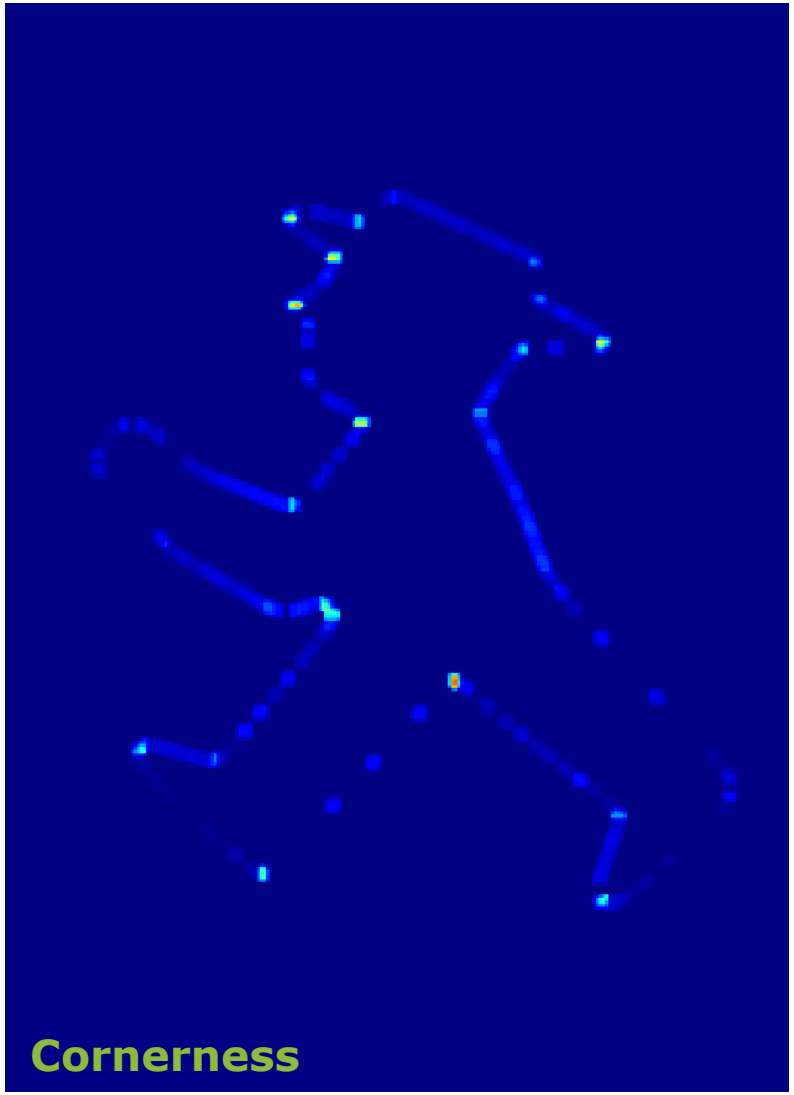
$$q = \frac{4 \cdot det(M)}{trace(M)^2}, \qquad 0 \leq q \leq 1$$

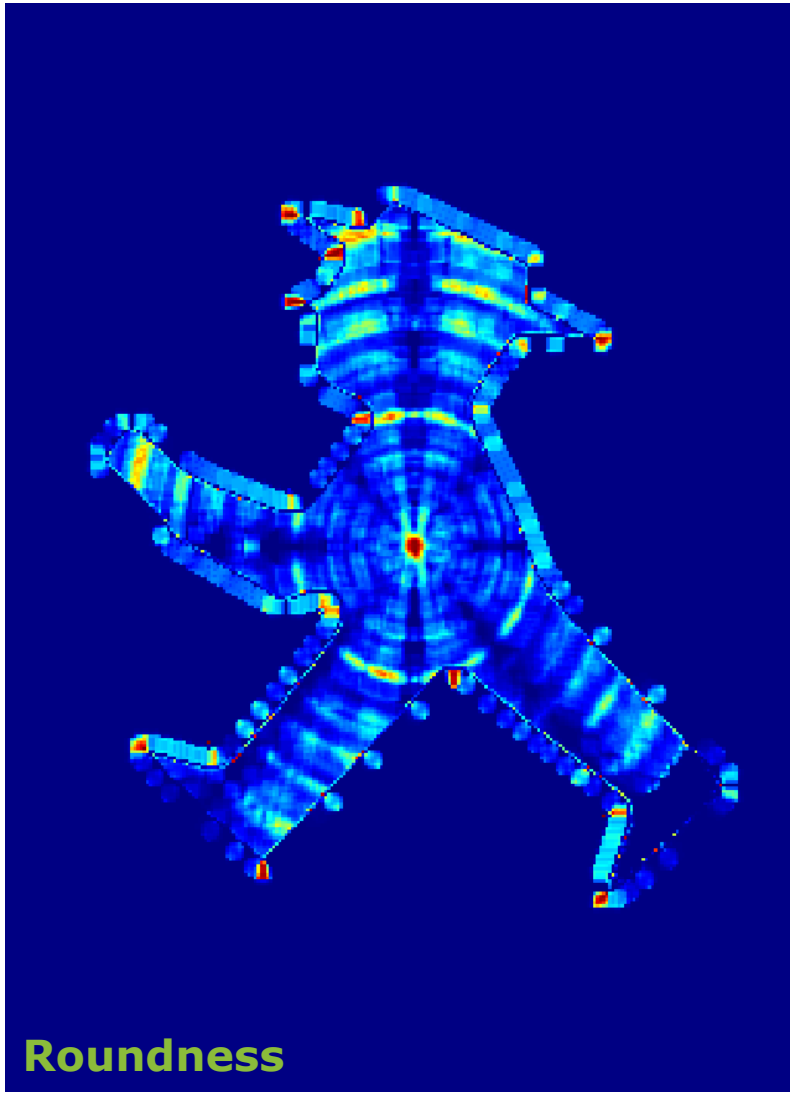
Find **corner point candidates $\boldsymbol{M_c}$,** if

$$w > t_w \quad \text{and} \quad q > t_q$$

$$t_w = [0.001 \ldots 0.01], t_q = [0.5 \ldots 0.75]$$

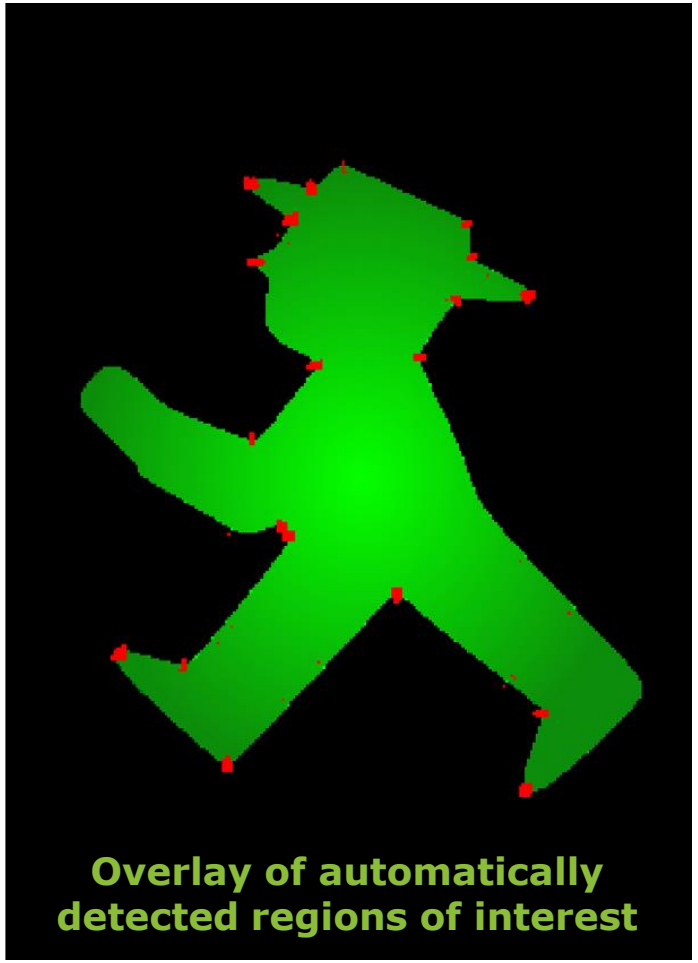
Thresholded regions of w and q



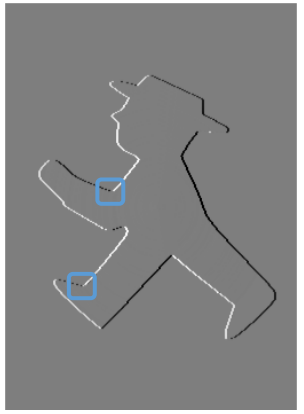


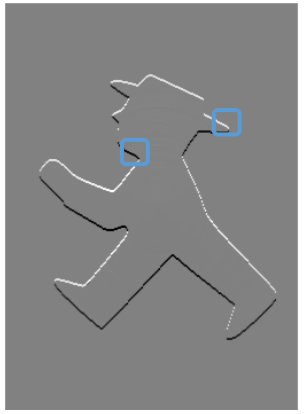
Bauhaus-
Universität
Weimar

Overlay of I with M_c



Overlay of automatically detected regions of interest





Gradient images I_x and I_y

- areas expected to be detected as interest regions

Assignment 2: Task B

Förstner interest operator:

a. Compute the **autocorrelation matrix $\boldsymbol{M}$** for each pixel using a 5×5 moving window

b. Instead of storing M for each pixel, compute the **cornerness $\boldsymbol{w}$** and **roundness $\boldsymbol{q}$** from M and store these values in matrices W and Q. Plot these arrays

c. Derive a **binary mask $\boldsymbol{M_c}$** of potential interest points by simultaneously applying thresholds, e.g. $t_w = 0.004$ and $t_q = 0.5$, on W and Q

d. Plot an overlay of the initial input image and the **detected points**

Original input image I overplayed with detected interest points