Digital Image Processing

Berlin University of Technology (TUB),
Computer Vision and Remote Sensing Group
Berlin, Germany

Digital Image Processing

(One) **Goal of Computer Vision:**

Automatic Understanding of digital Images!

Image is distorted?

→ Image restoration (e.g. Wiener filter)

Image has still bad quality?

→ Image enhancement (e.g. unsharp masking)

How to describe an image?

→ Just pixel intensity?!

Image Features

Digital Image Processing

Automatic Image Analysis

Image → Image

Image → Features

Features → “Understanding”

[DIP Summer term]

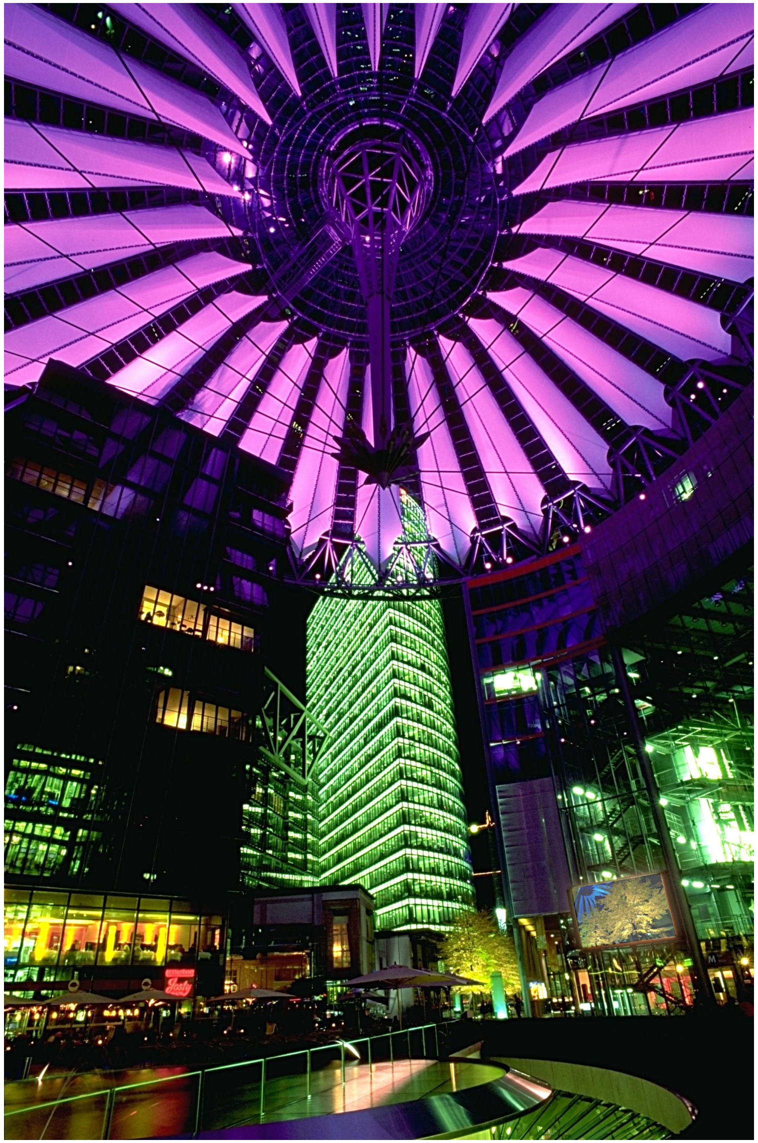
[AIA Winter term]
[PCV Winter term]

How to get a meaningful image description?

How to use this image description to infer information about image content?

Image Features

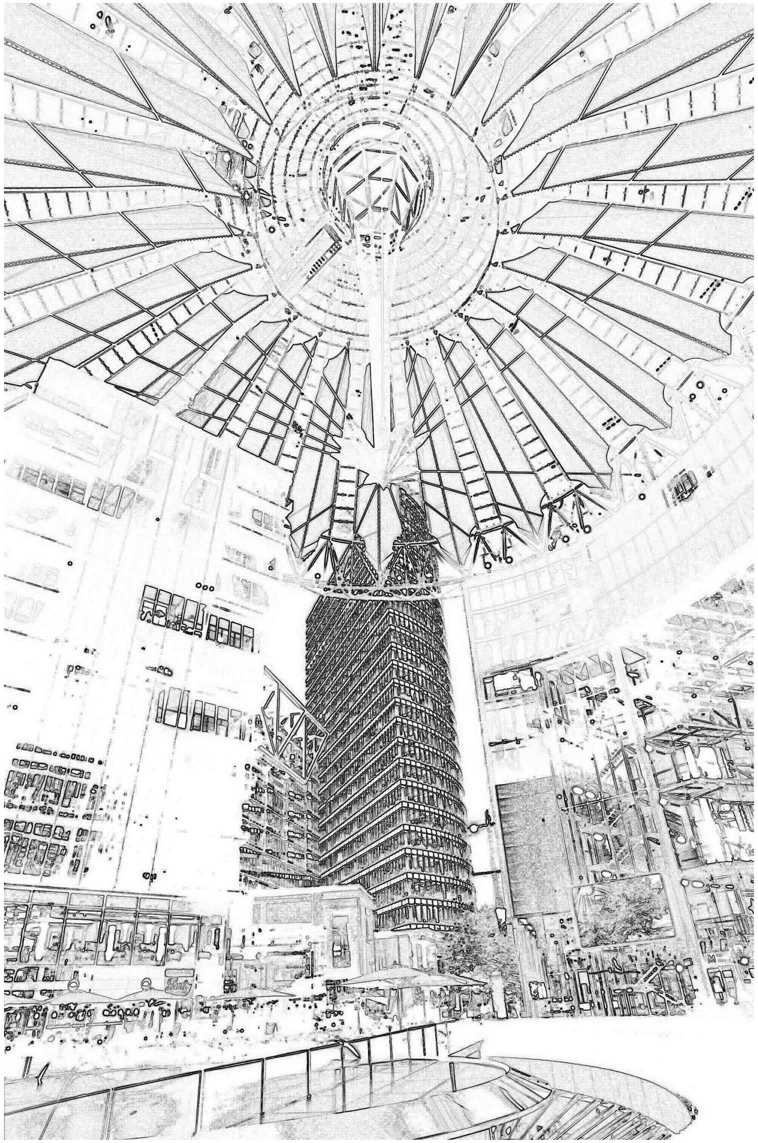
Color



Intensity

Image Features

Edges



Segments

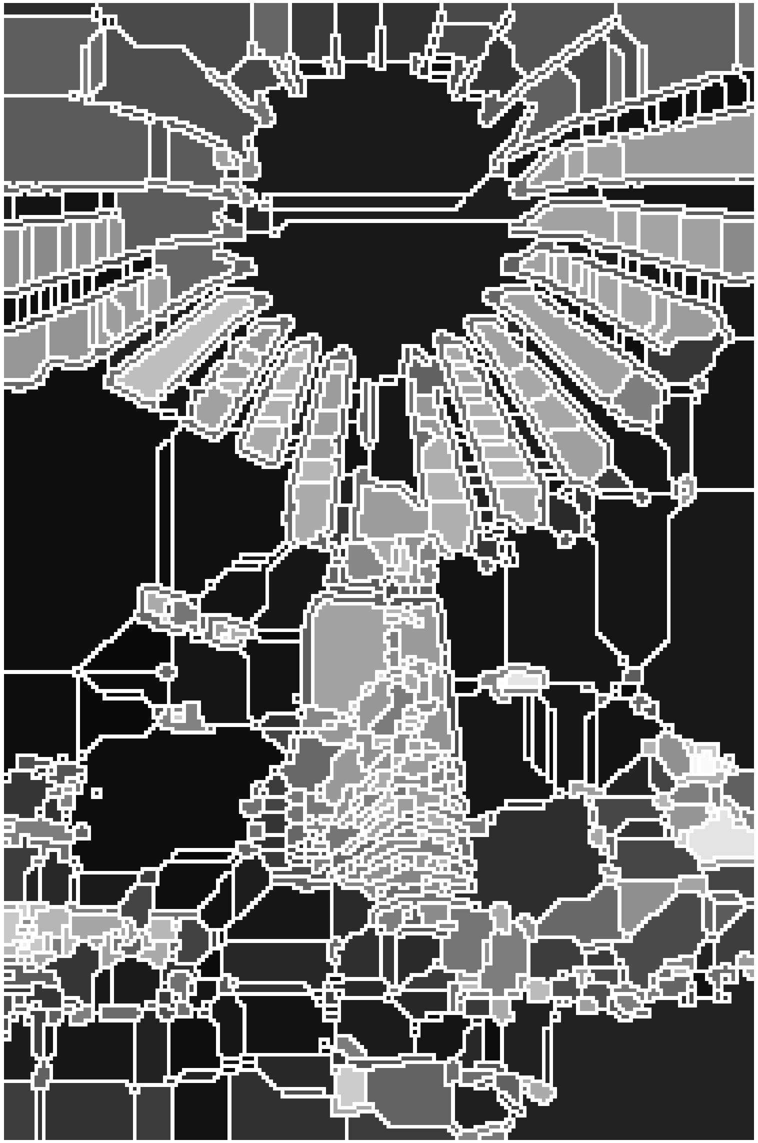


Image Features

Texture

Interest Points

Basics

- Directional Gradients
- Covariance Matrices

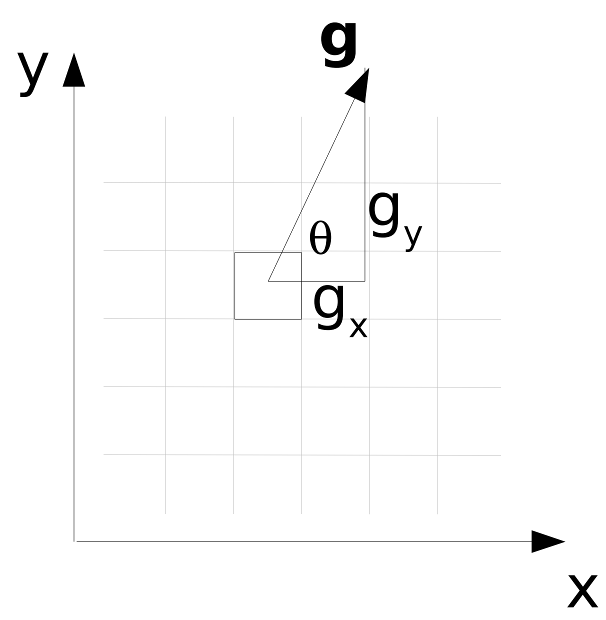
Directional Gradients

- Computing the gradient magnitude:
  - Use e.g. a radially symmetric filter
  - No information concerning the direction of gradients

- Directional gradients: Convolution with suitable filters, e.g. G_x and G_y
  - **Image ⊗ $\mathbf{G_x}$** -> Gradient in x direction
  - **Image ⊗ $\mathbf{G_y}$** -> Gradient in y direction

- Each pixel is associated with a gradient vector $\mathbf{g} = (g_x, g_y)^T$

Directional Gradients



- Gradient magnitude: $|\mathbf{g}| = \sqrt{g_x^2 + g_y^2}$

- Gradient direction: $\theta = \tan^{-1}\left(\frac{g_y}{g_x}\right)$

  - Direction in which intensity increases quickest

Directional Gradients

- Commonly Used:
  Composition of differential operator and low-pass
- E.g. derivatives of the normal distribution:

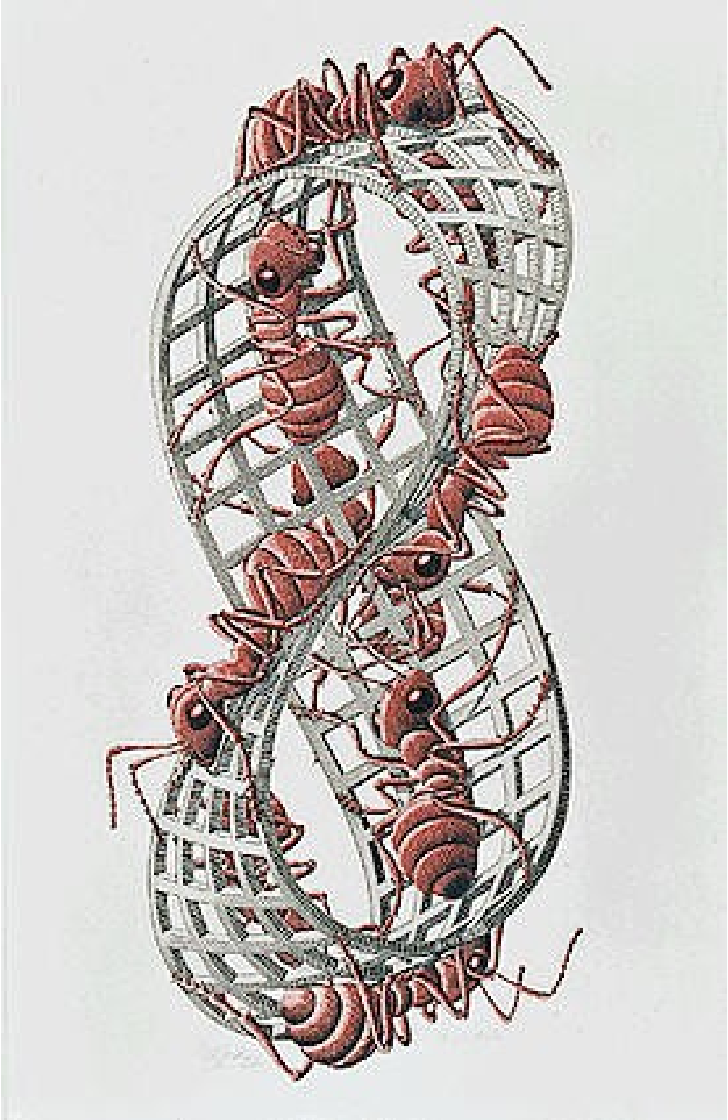
$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

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

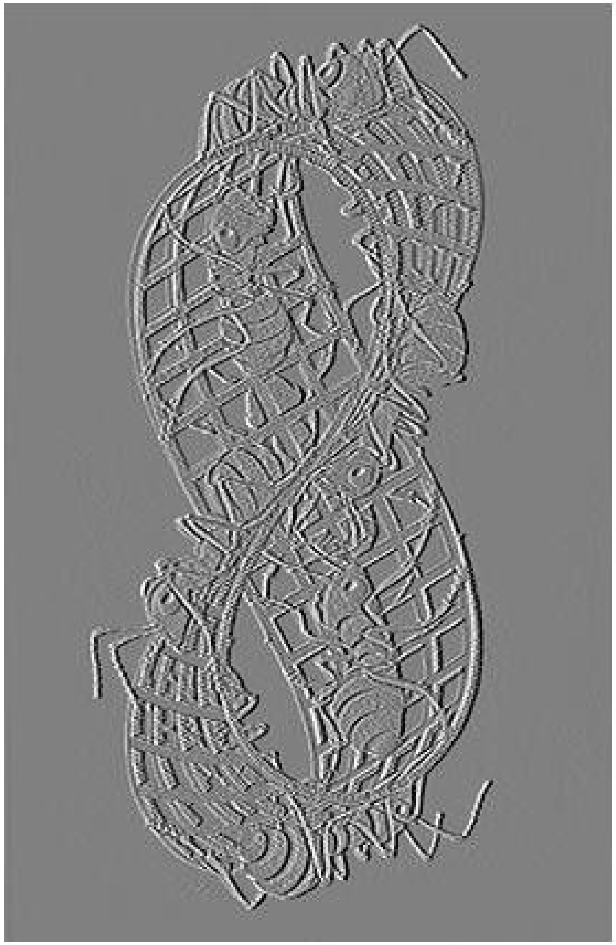
$$G_y(x, y) = \frac{\partial}{\partial y} G(x, y; \sigma) = \frac{-y}{2\pi\sigma^4} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

- σ: Scale and noise sensitivity
  - σ small: Small structures discernable, noise/texture preserved
  - σ large: Large structures emphasized, noise suppressed

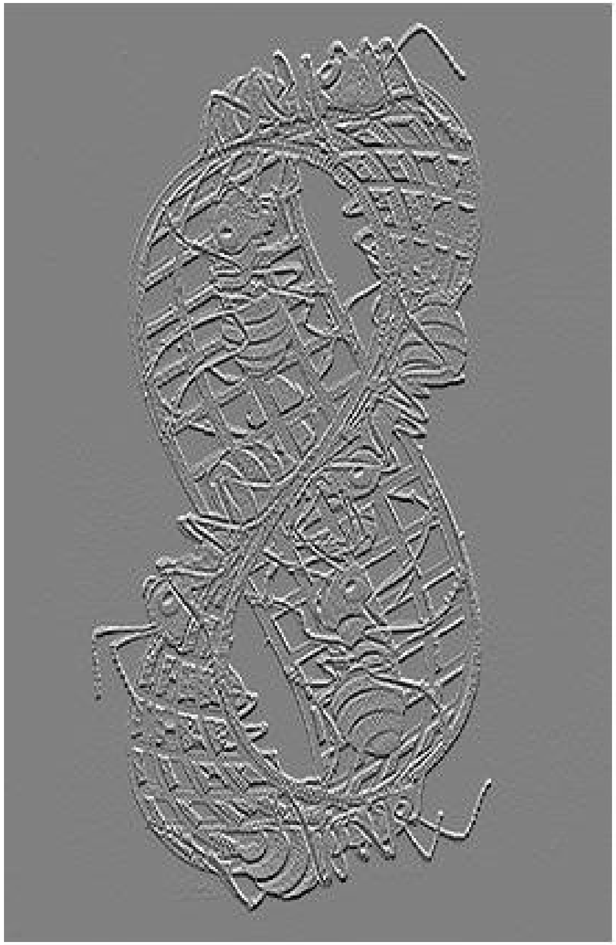
Directional Gradients



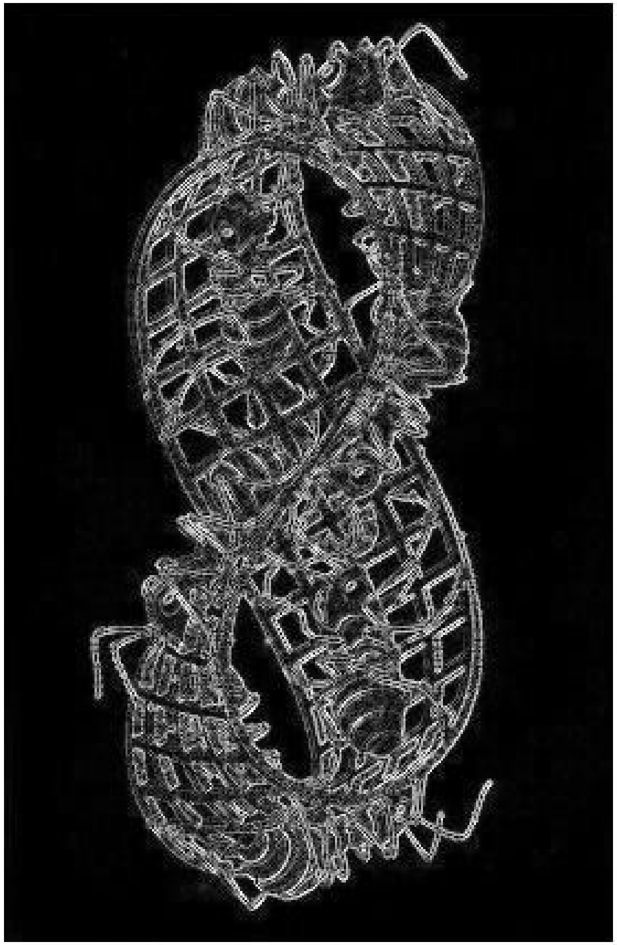
Directional Gradients



g_x



g_y



$|\mathbf{g}|$

Directional Gradients

g_x

g_y

$|\mathbf{g}|$

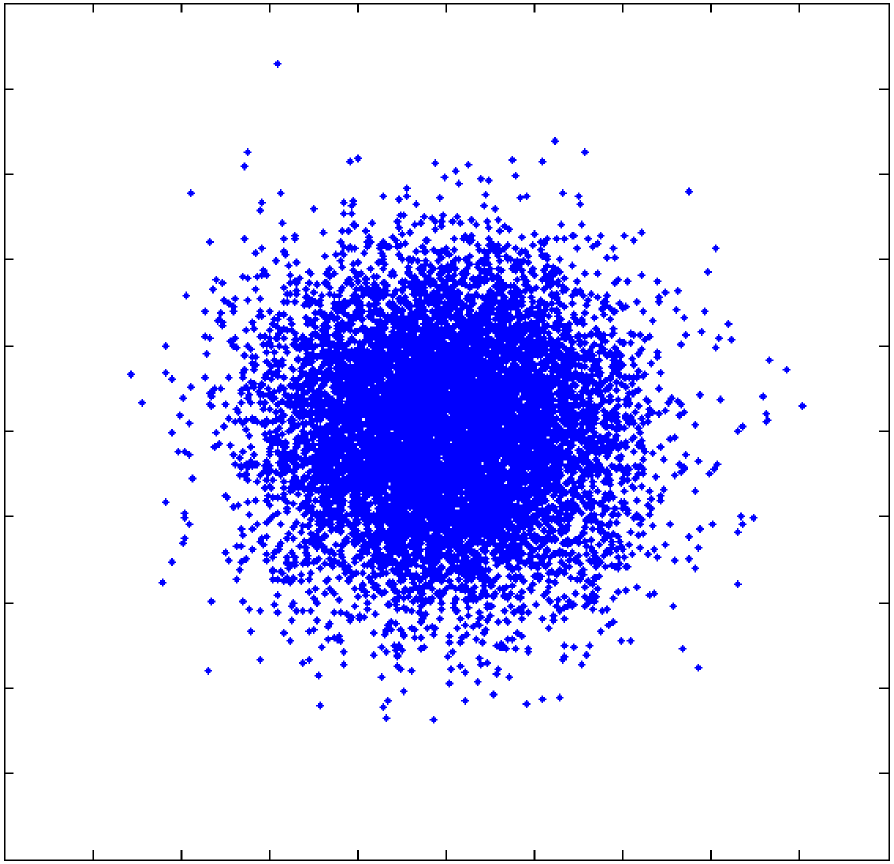
Covariance Matrices

- Variance of scalars $\{x_1, x_2, x_3, \ldots, x_N\}$: Measures dispersion around mean μ
- For vector sets $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_N\}$ ($\mathbf{x}_i$ M-dimensional):
  - Mean $\mu = \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}_j$
  - Covariance $\mathbf{\Sigma} = \frac{1}{N}\sum_{j=1}^{N} (\mathbf{x}_j - \mu)(\mathbf{x}_j - \mu)^T$
- For $\mathbf{x}_j = (x_{j,1}, x_{j,2}, x_{j,3}, \ldots, x_{j,M})^T$:

$$\mathbf{\Sigma} = \begin{pmatrix} \sum_j (x_{j,1}-\mu_1)^2 & \sum_j (x_{j,1}-\mu_1)(x_{j,2}-\mu_2) & \ldots & \sum_j (x_{j,1}-\mu_1)(x_{j,M}-\mu_M) \\ \sum_j (x_{j,2}-\mu_2)(x_{j,1}-\mu_1) & \sum_j (x_{j,2}-\mu_2)^2 & \ldots & \sum_j (x_{j,2}-\mu_2)(x_{j,M}-\mu_M) \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$$

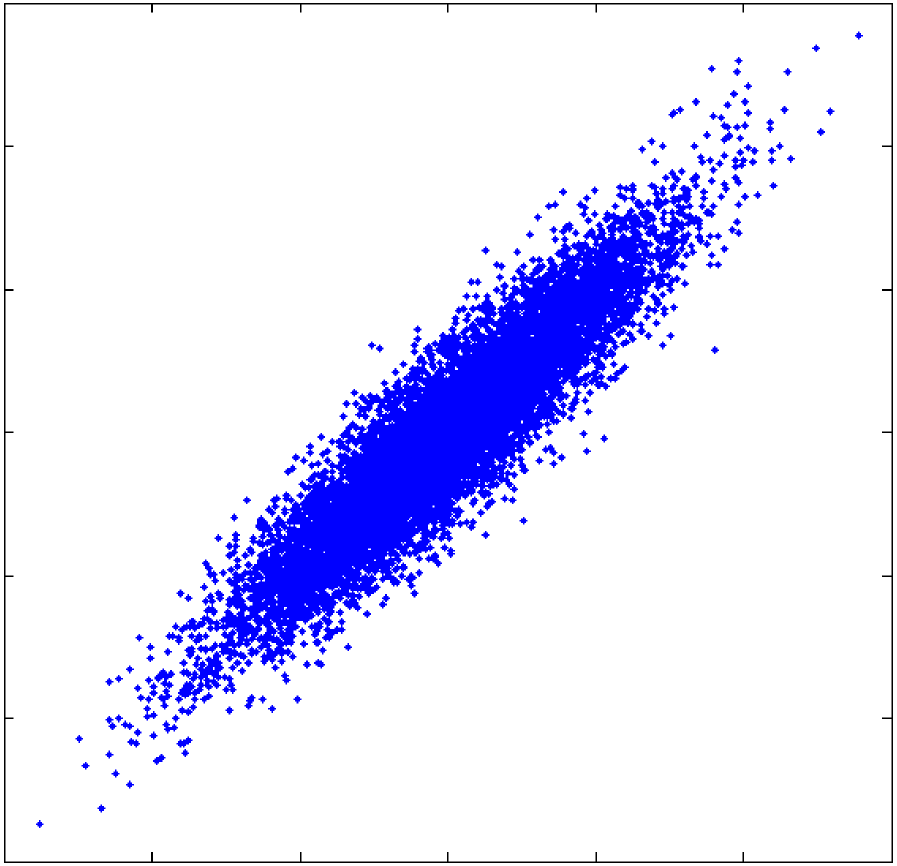
  - Diagonal: Variance along individual dimensions
  - Otherwise: Correlation between dimensions

Covariance Matrices



$$\Sigma = \begin{pmatrix} 0.9976 & -0.0187 \\ -0.0187 & 0.9700 \end{pmatrix}$$

x & y almost uncorrelated

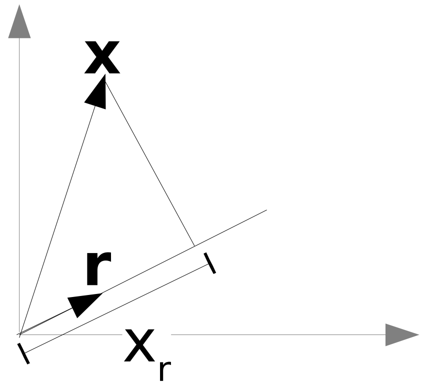


$$\Sigma = \begin{pmatrix} 0.4998 & 0.4625 \\ 0.4625 & 0.5054 \end{pmatrix}$$

x & y strongly correlated

Covariance Matrices

- Component of a vector **x** in direction **r**:



  - **r** = (cos θ, sin θ)^T
  - x_r: Scalar, component of **x** along **r**
  - x_r **=** $\mathbf{r}^T \mathbf{x}$

- Mean and variance of vectors $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_N\}$ along direction **r**:

$$\mu_r \;=\; \frac{1}{N}\sum_j \mathbf{r}^T\,\mathbf{x}_j \;=\; \mathbf{r}^T\,\frac{1}{N}\sum_j \mathbf{x}_j \;=\; \mathbf{r}^T\,\mu$$

$$\begin{aligned} Var_r(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \;&=\; Var(\mathbf{r}^T\mathbf{x}_1, \mathbf{r}^T\mathbf{x}_2, \ldots, \mathbf{r}^T\mathbf{x}_N) \\ &=\; \tfrac{1}{N}\textstyle\sum_j (\mathbf{r}^T\mathbf{x}_j - \mu_r)^2 \;=\; \tfrac{1}{N}\textstyle\sum_j (\mathbf{r}^T\mathbf{x}_j - \mathbf{r}^T\mu)^2 \\ &=\; \tfrac{1}{N}\textstyle\sum_j (\mathbf{r}^T\mathbf{x}_j - \mathbf{r}^T\mu)(\mathbf{r}^T\mathbf{x}_j - \mathbf{r}^T\mu)^T \;=\; \tfrac{1}{N}\textstyle\sum_j \mathbf{r}^T(\mathbf{x}_j - \mu)(\mathbf{x}_j - \mu)^T\mathbf{r} \\ &=\; \mathbf{r}^T\Sigma\mathbf{r} \end{aligned}$$

- The covariance matrix determines the variance in all directions!

Covariance Matrices

Task: Find directions with maximal variance, i.e.: $r^T \Sigma r = max$

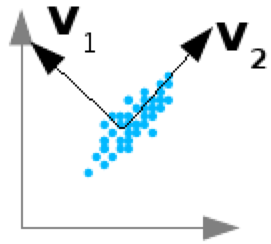
Solution:

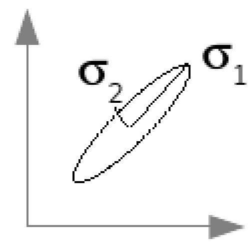
$$\Sigma = \mathbf{V}\,\mathbf{D}\,\mathbf{V}^T$$

$$\begin{pmatrix} \uparrow & \uparrow & \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots \\ \downarrow & \downarrow & \end{pmatrix} \qquad \begin{pmatrix} l_1 & 0 & \cdots \\ 0 & l_2 & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

Eigenvectors Eigenvalues (diagonal)

- In the eigenbasis, dimensions of vectors **x** are not correlated





- Variance along in the directions $\mathbf{v}_1$, $\mathbf{v}_2$ ... : l_1, l_2, ...
- Standard deviations in other directions form an ellipse with major/minor axes along eigenvectors with deviations given by eigenvalues

Structure Tensor

- For each pixel, the structure tensor **A** is defined as:

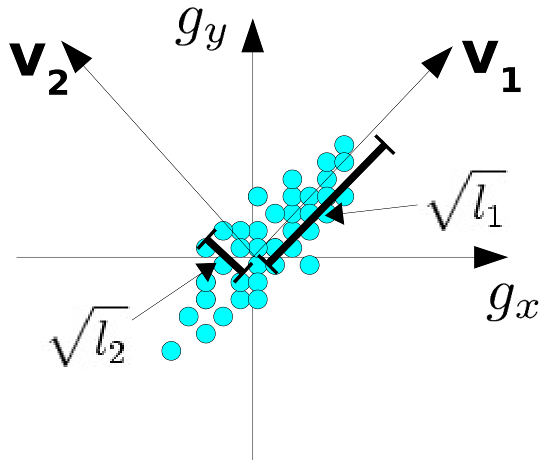
$$\mathbf{A} = \sum_W \mathbf{g}\mathbf{g}^T = \begin{pmatrix} \sum_W g_x^2 & \sum_W g_x g_y \\ \sum_W g_y g_x & \sum_W g_y^2 \end{pmatrix}$$

- *W* denotes the neighbourhood of the pixel considered
  - → In this exercise: Gaussian window with std-dev N_W around the pixel
- **A** is a covariance matrix computed assuming $\mu = 0$
  - → **A** describes the distribution of gradients around **g** = $(0,0)^T$

- For covariance **Σ**: $\mathbf{r}^T \mathbf{\Sigma} \mathbf{r} =$ Variance along **r**

- For the structure tensor **A**: $\mathbf{r}^T \mathbf{A} \mathbf{r} =$ (Squared) gradient magnitude along **r**

Structure Tensor



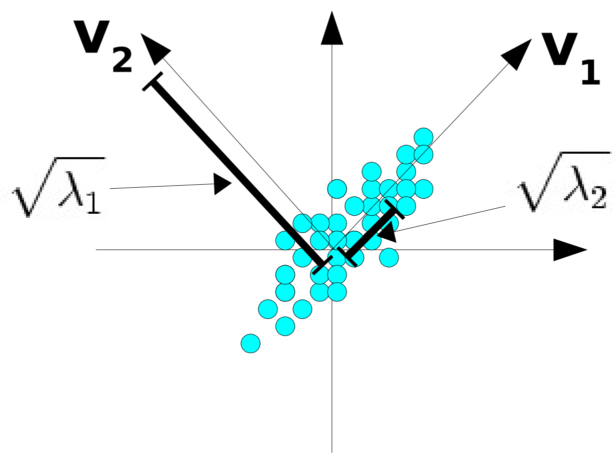
$$\mathbf{A} = \mathbf{V}\,\mathbf{D}\,\mathbf{V}^T$$

$$\mathbf{V} = \begin{pmatrix} \uparrow & \uparrow \\ \mathbf{v}_1 & \mathbf{v}_2 \\ \downarrow & \downarrow \end{pmatrix} \qquad \mathbf{D} = \begin{pmatrix} l_1 & 0 \\ 0 & l_2 \end{pmatrix}$$

$$l_1 \geq l_2$$

- $\mathbf{v}_1$: Direction with the greatest gradient magnitude (max. eigenvalue l_1)
  - Gradient direction $\mathbf{v}_1$ dominates neighbourhood W
- l_1: Total (squared) gradient magnitude along direction $\mathbf{v}_1$
- $\mathbf{v}_2$: Direction with the smallest gradient magnitude
  - Gradient direction $\mathbf{v}_2$ is rare in neighbourhood W
- Gradient magnitude as a function of direction describes an ellipse with major/minor axes along $\mathbf{v}_1$ und $\mathbf{v}_2$

Structure Tensor



$$\mathbf{A}^{-1} = (\mathbf{V}\ \mathbf{D}\ \mathbf{V}^T)^{-1} = \mathbf{V}\ \mathbf{D}^{-1}\ \mathbf{V}^T$$

$$\mathbf{D}^{-1} = \begin{pmatrix} \frac{1}{l_2} & 0 \\ 0 & \frac{1}{l_1} \end{pmatrix} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

- Eigenvectors remain unchanged
- Eigenvalues are inverted
- Small eigenvalues λ_1 und λ_2 indicate strong gradients in the neighborhood
- If λ_1 and λ_2 are large, the image is homogeneous

Förstner Operator

- The structure tensor can be used to derive salient information:
- Weight **w**: Strength of gradients in the neighbourhood

$$w = \frac{1}{\operatorname{tr}\left(\mathbf{A}^{-1}\right)} = \frac{1}{\lambda_1 + \lambda_2} = \frac{\det(\mathbf{A})}{\operatorname{tr}(\mathbf{A})} \qquad w > 0$$

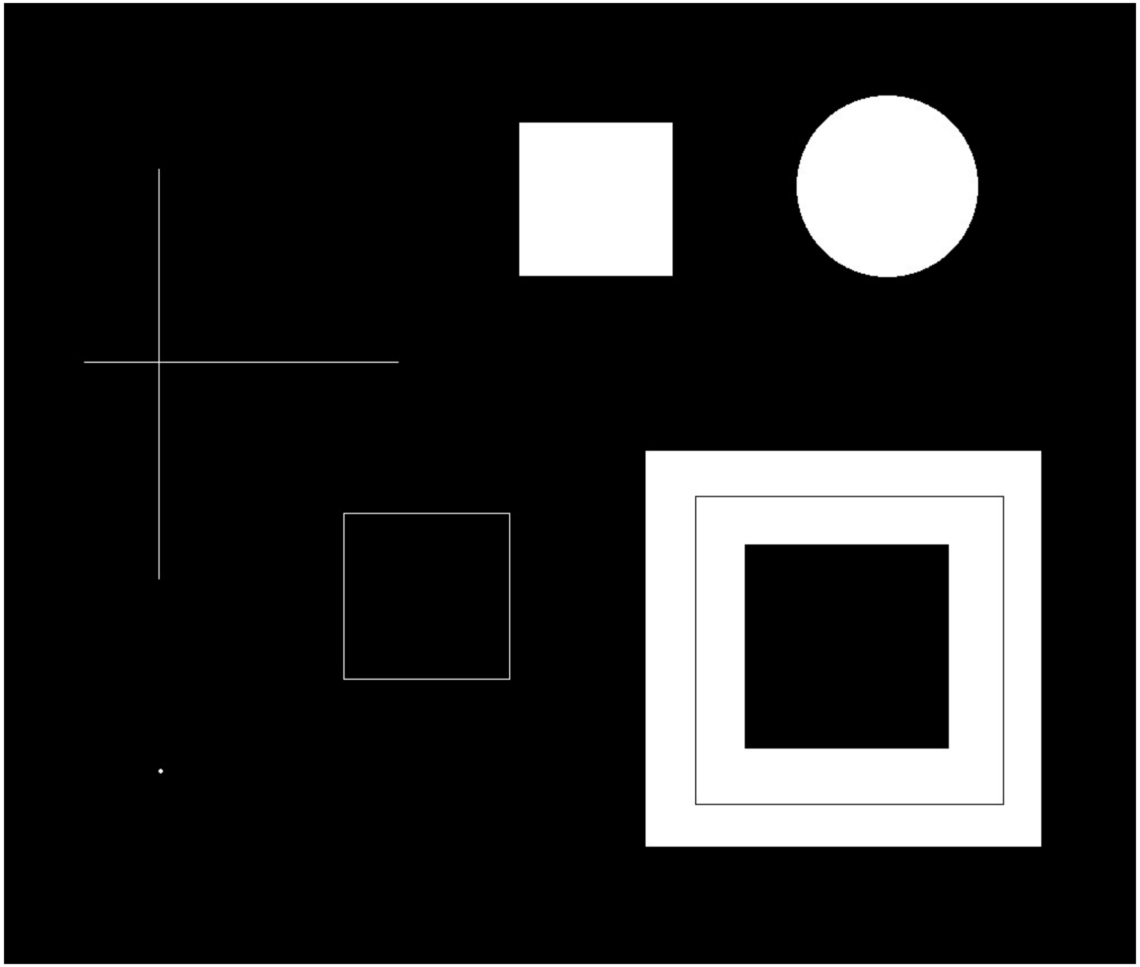
  - w large: λ_1 und λ_2 small, i.e. strong gradients in the neighbourhood
  - $w_{min} = 0.5, \ldots, 1.5 \cdot \bar{w}$, $\bar{w}$ is the mean of w over whole image

- Isotropy **q**: Measures the uniformity of gradient directions in the neighbourhood

$$q = 1 - \left(\frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2}\right)^2 = \frac{4\det\left(\mathbf{A}\right)}{\operatorname{tr}\left(\mathbf{A}\right)^2} \qquad 0 \leq q \leq 1$$

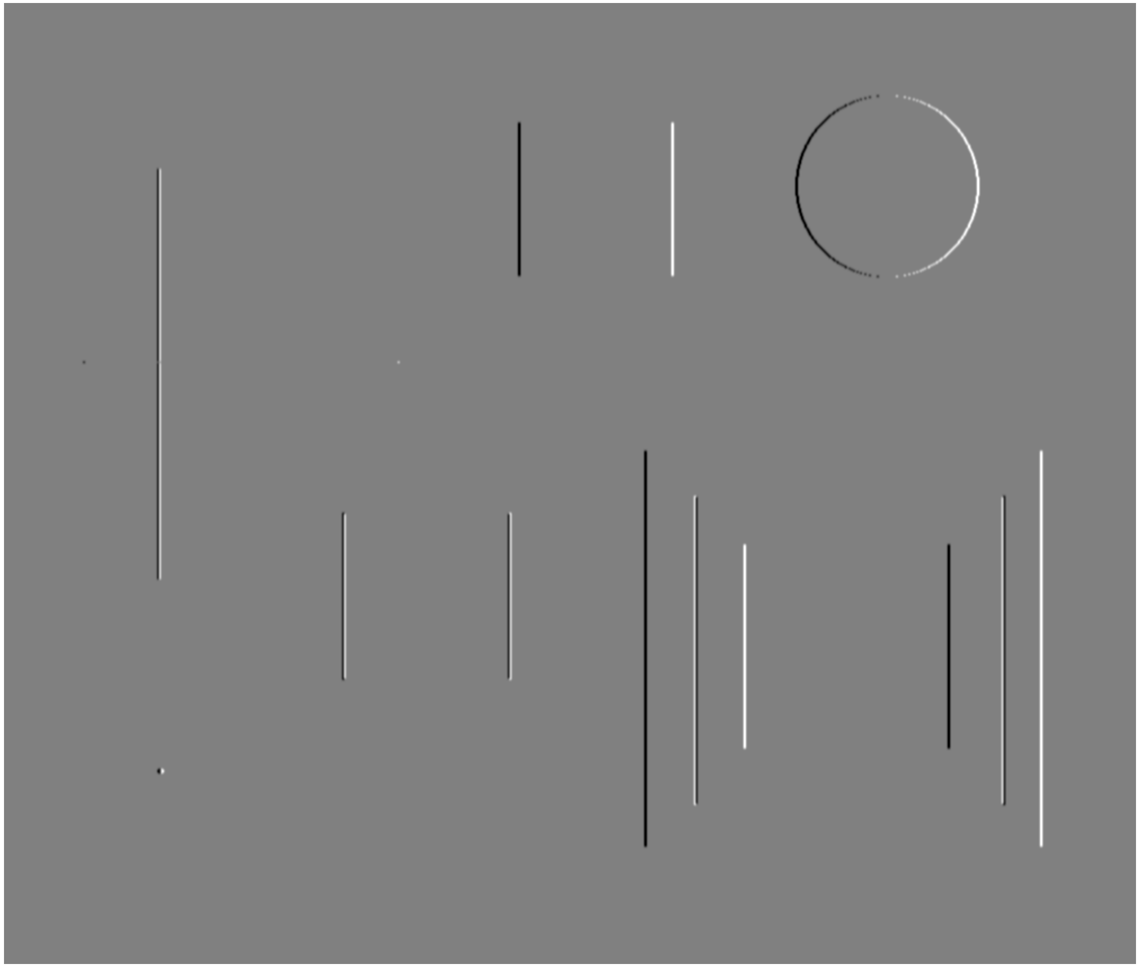
  - **q** small: Gradients occur primarily in one direction
  - $q_{min} = 0.5, \ldots, 0.75$

Förstner Operator



Original image

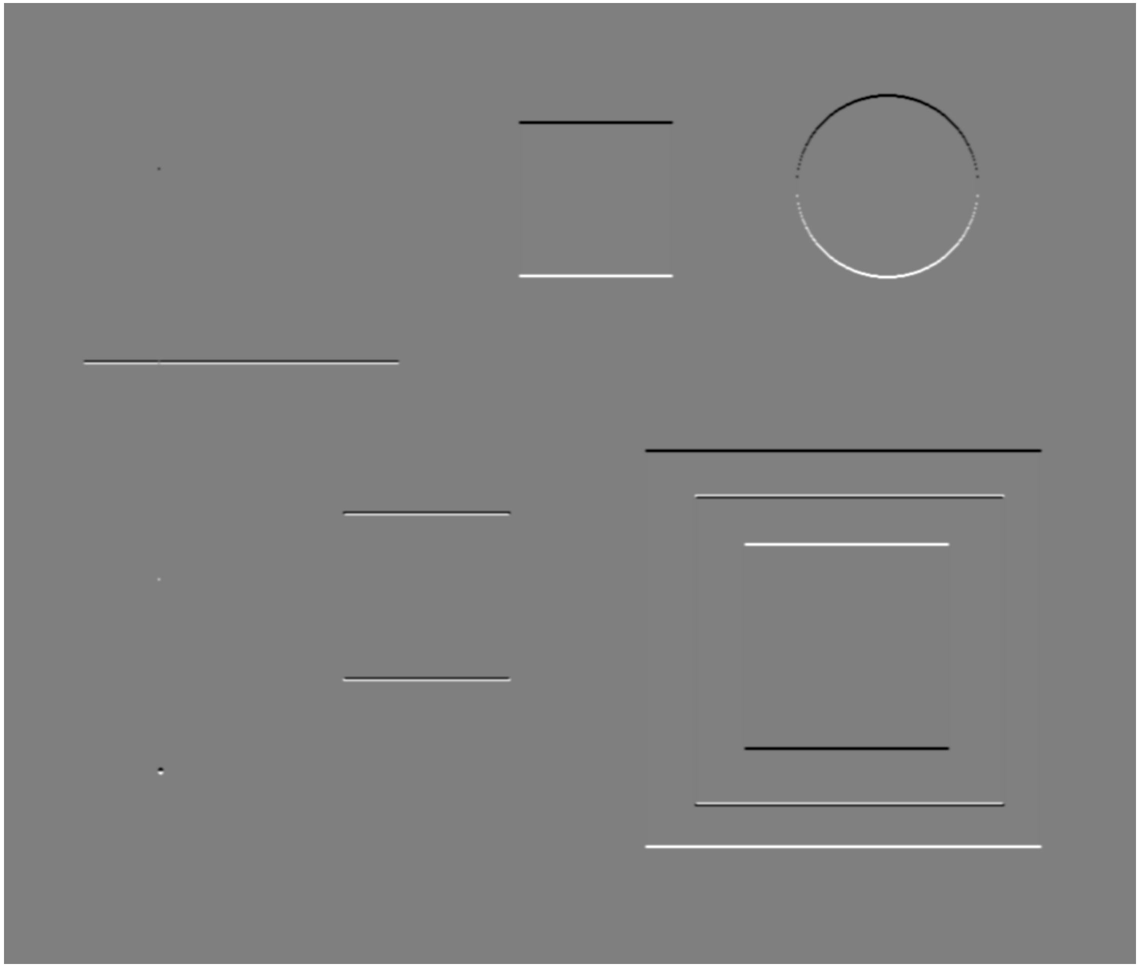
Förstner Operator



Gradient in x-direction

1. Gradient in x-direction

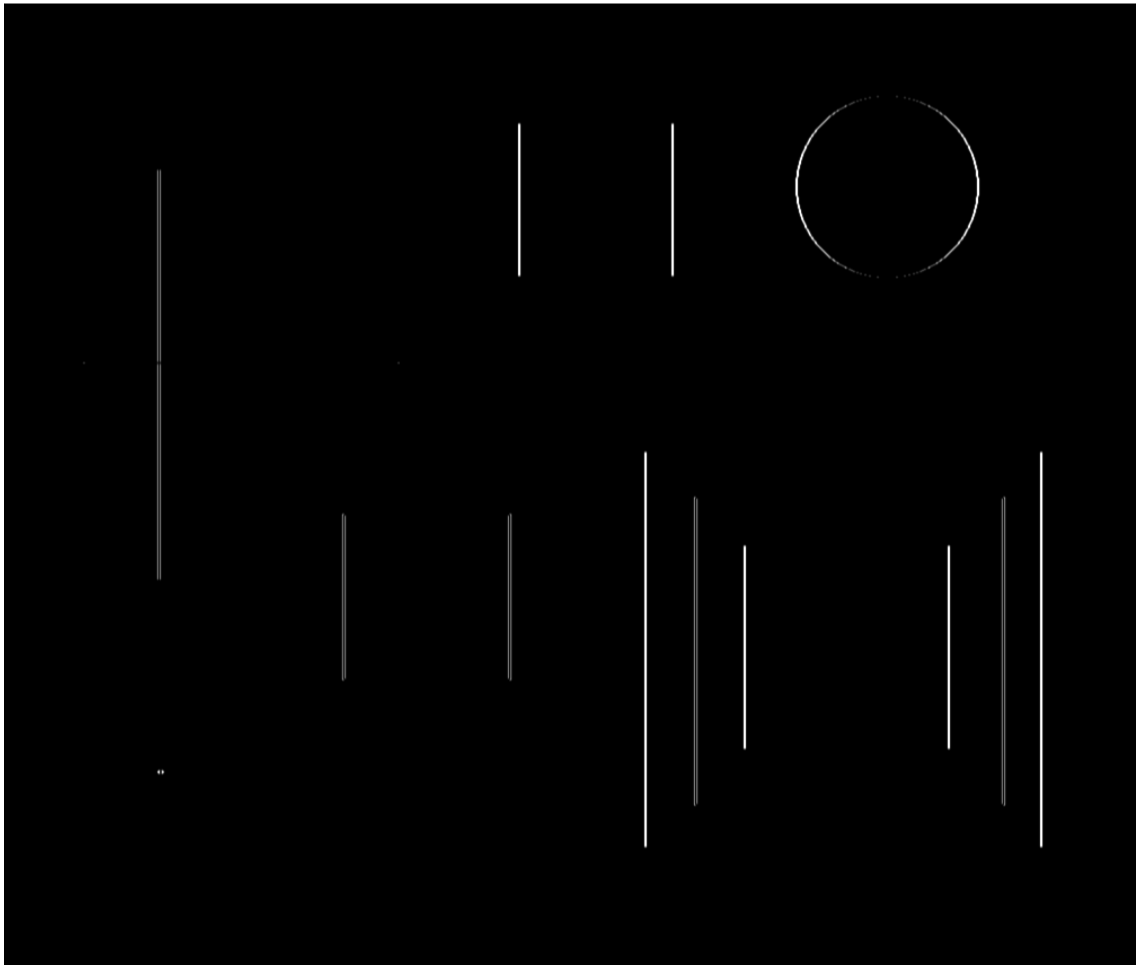
Förstner Operator



Gradient in y-direction

1. Gradient in x- and y-direction

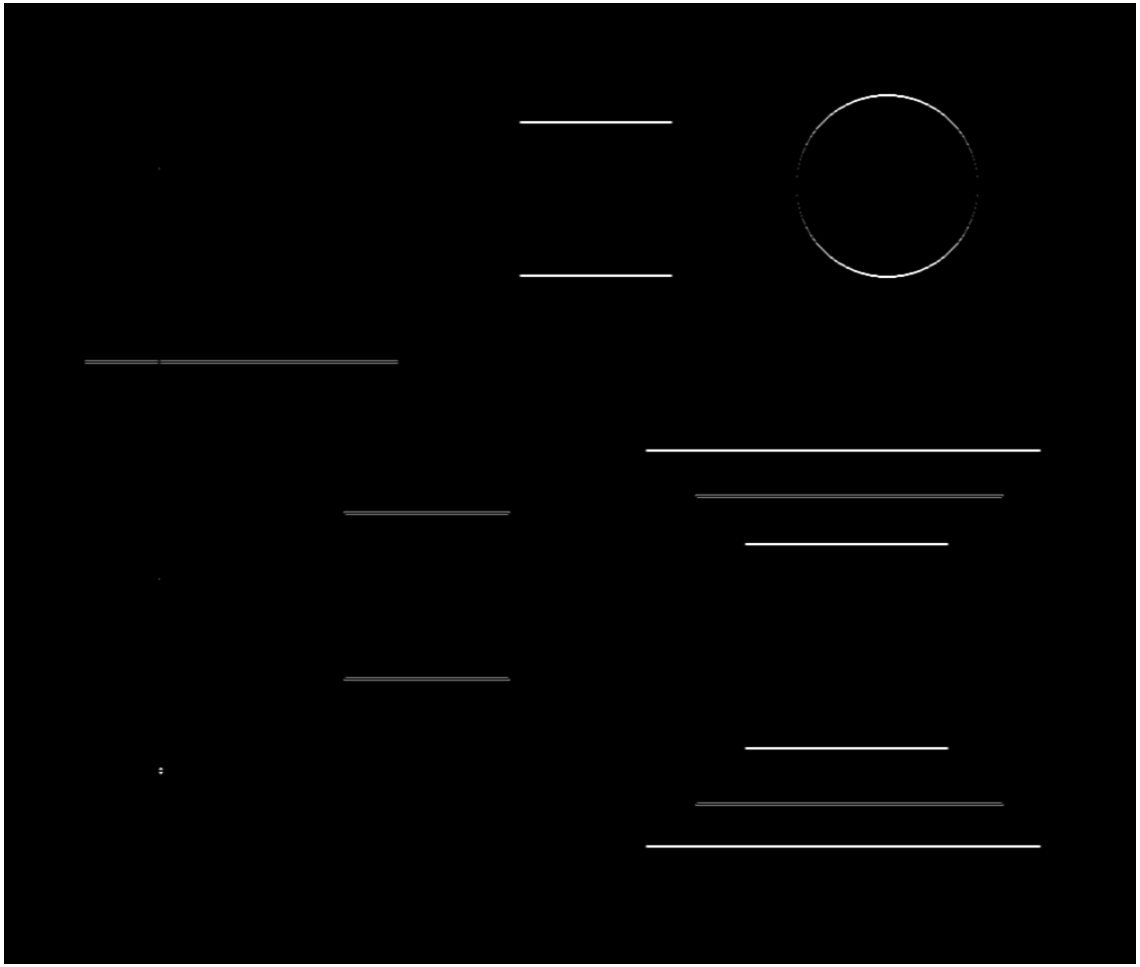
Förstner Operator



$g_x \cdot g_x$

1. Gradient in x- and y-direction
2. $g_x \cdot g_x$

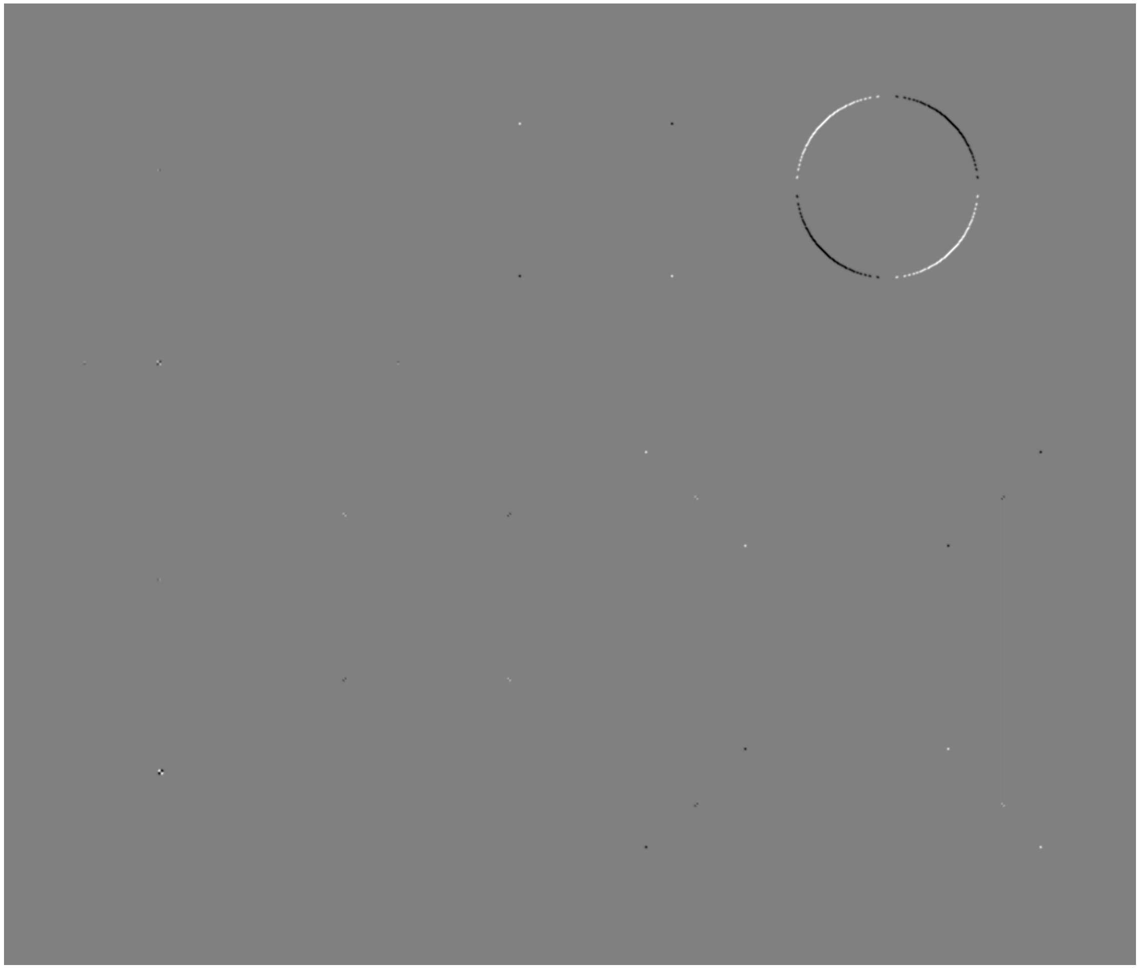
Förstner Operator



$g_y \cdot g_y$

1. Gradient in x- and y-direction
2. $g_x \cdot g_x, g_y \cdot g_y$

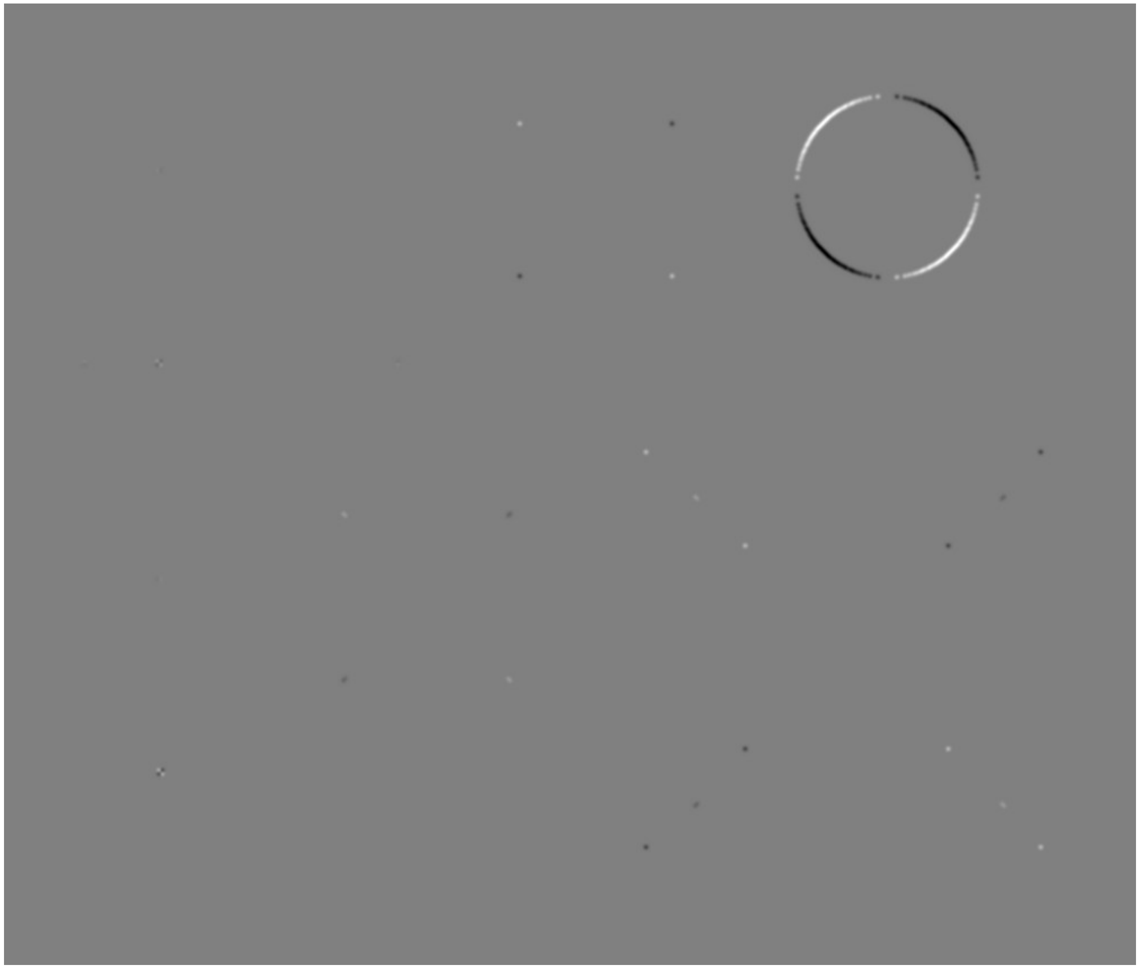
Förstner Operator



$g_x \cdot g_y$

1. Gradient in x- and y-direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$

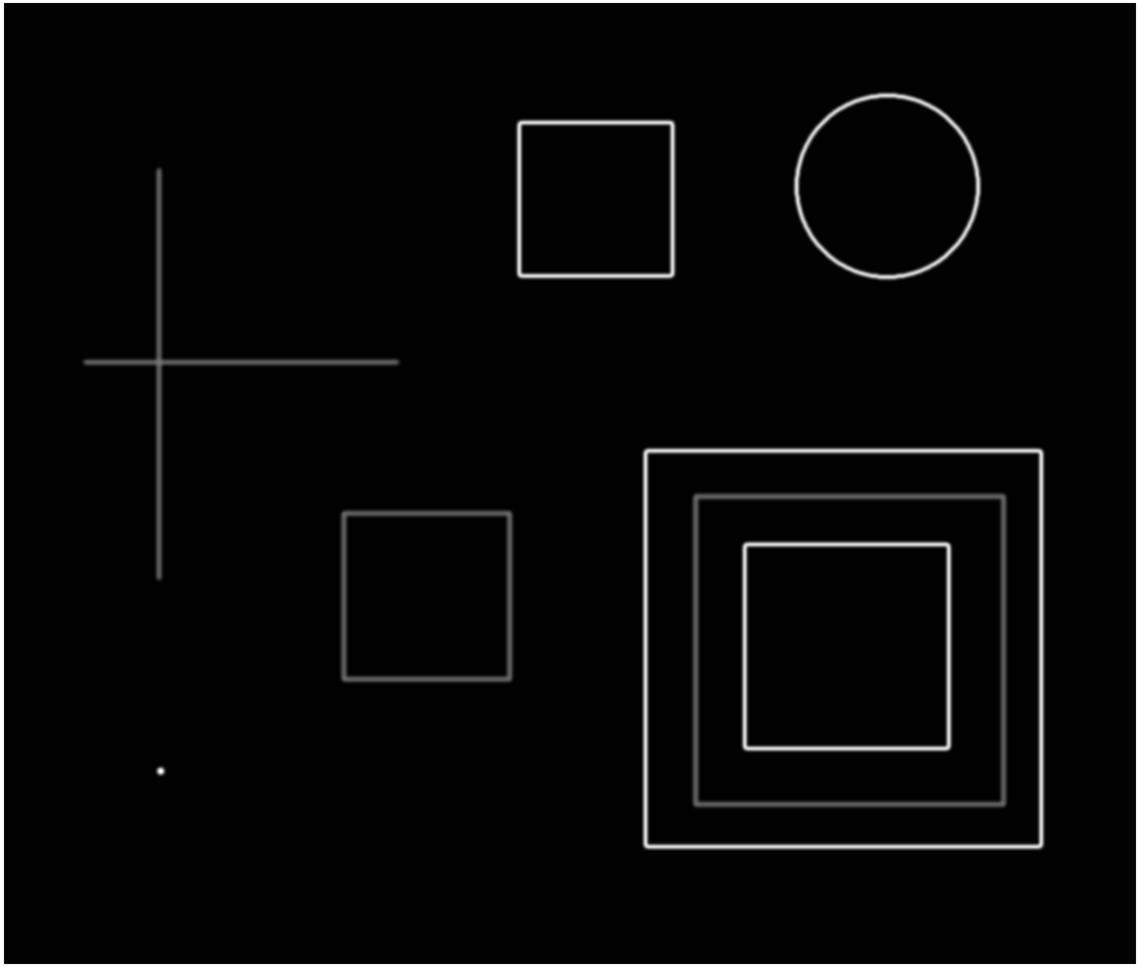
Förstner Operator



Averaged (smoothed) $g_x \cdot g_y$

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)

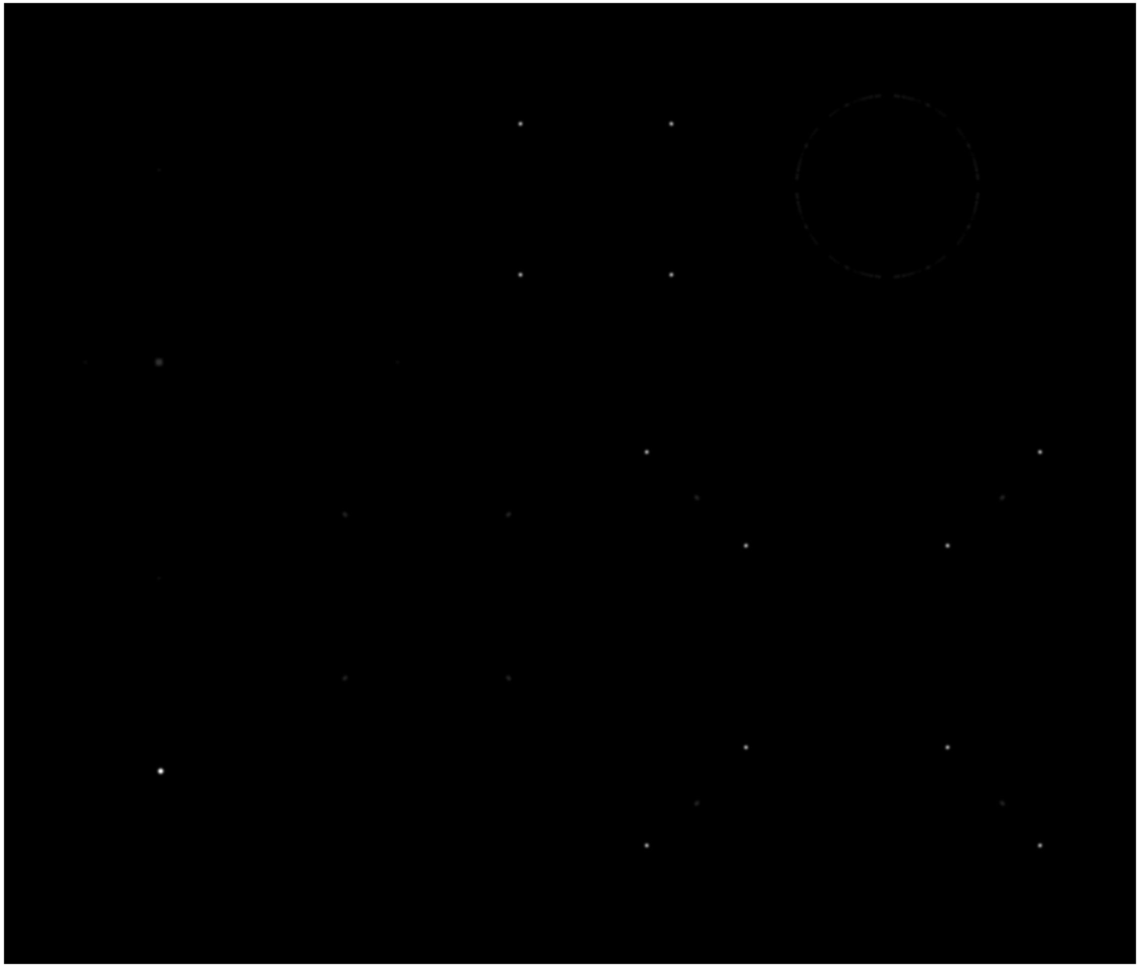
Förstner Operator



tr(A)

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor

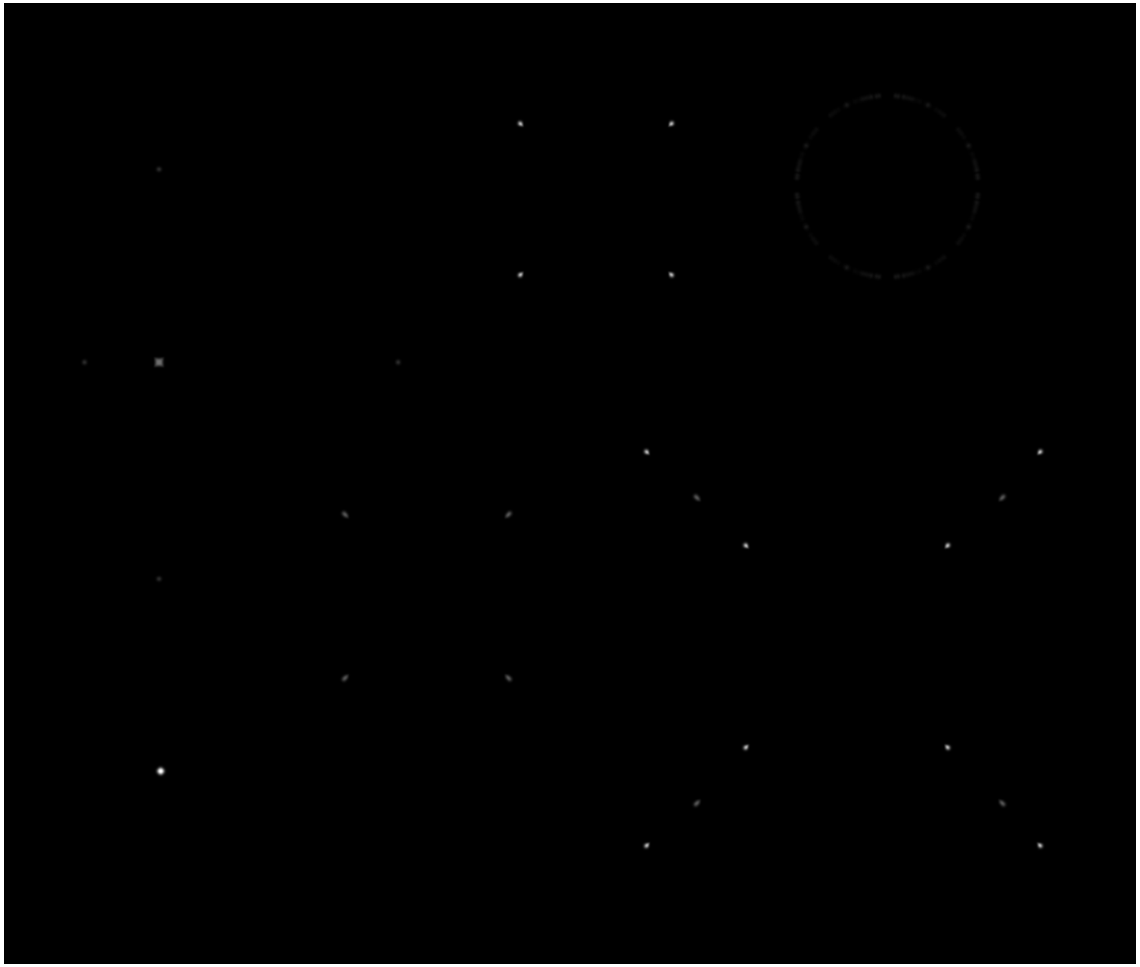
Förstner Operator



|A|

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor

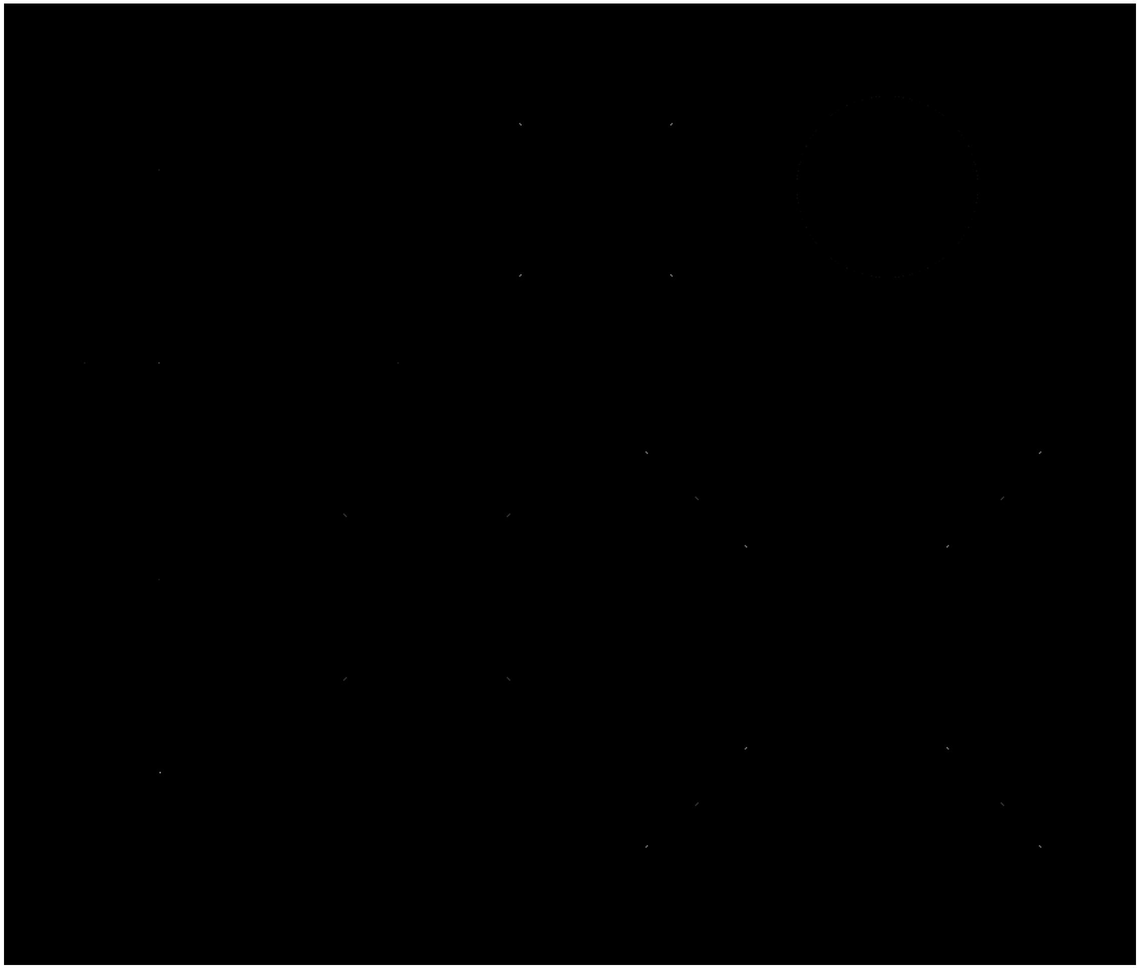
Förstner Operator



Weight w

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor
6. weight calculation

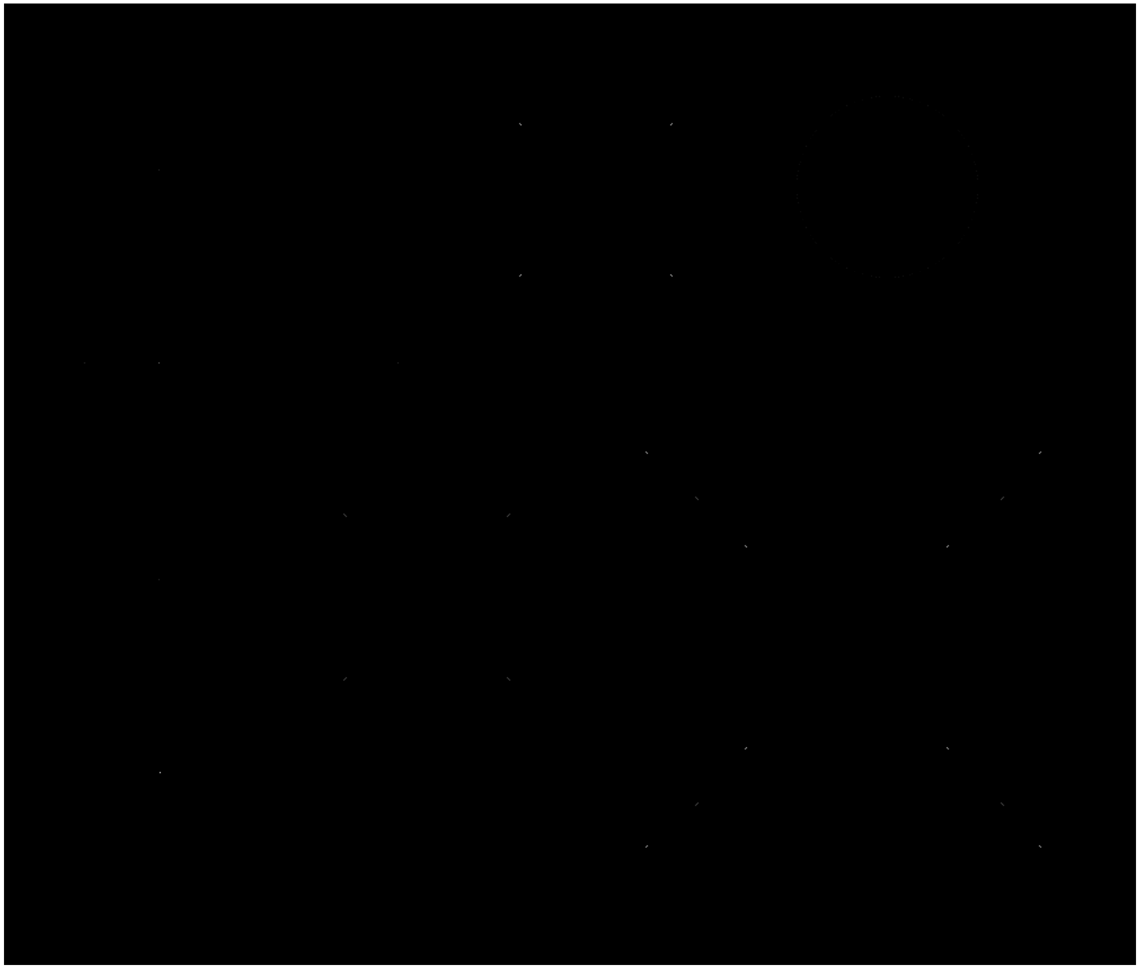
Förstner Operator



Weight w

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor
6. weight calculation
7. weight non-max suppression

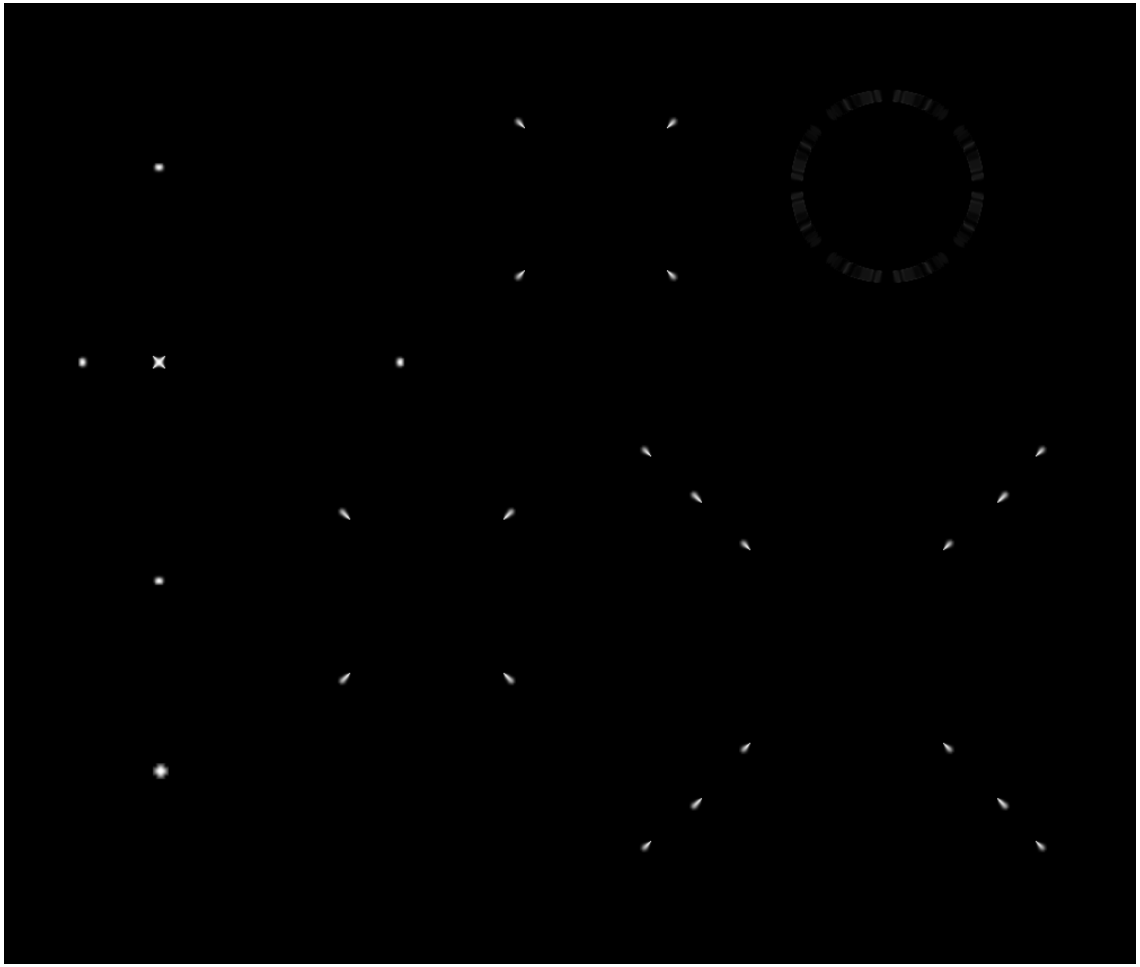
Förstner Operator



Weight w

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor
6. weight calculation
7. weight non-max suppression
8. weight thresholding

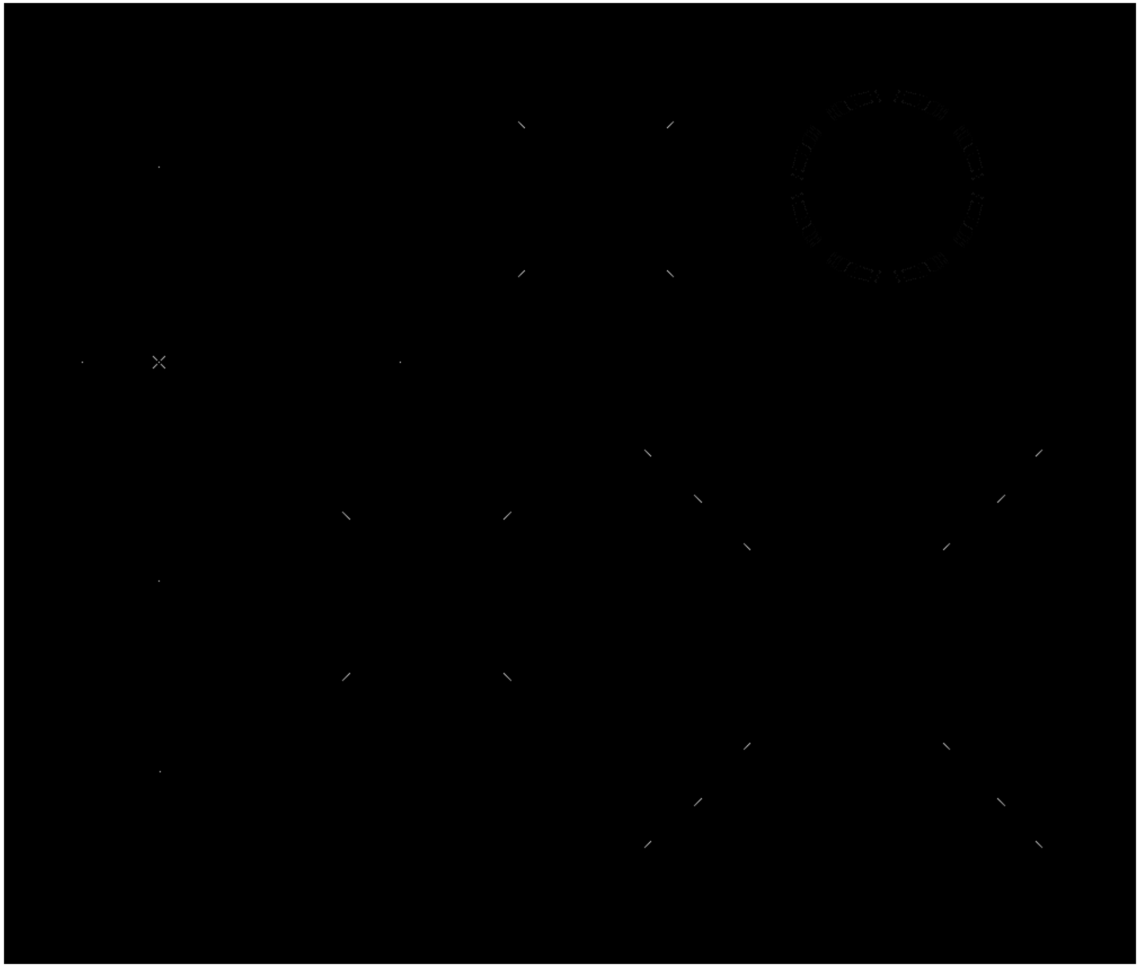
Förstner Operator



Isotropy q

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor
6. weight calculation
7. weight non-max suppression
8. weight thresholding
9. isotropy calculation

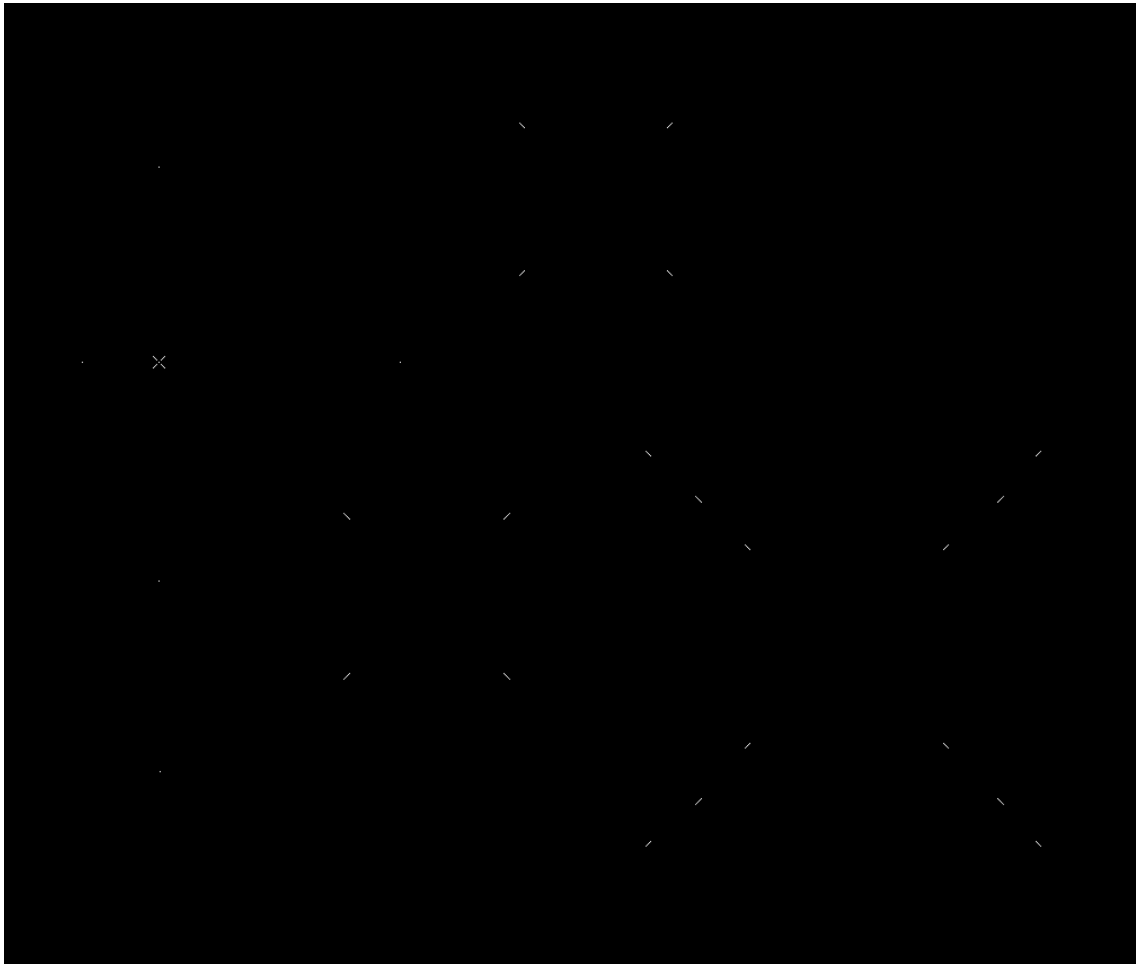
Förstner Operator



Isotropy q

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor
6. weight calculation
7. weight non-max suppression
8. weight thresholding
9. isotropy calculation
10. isotropy non-max suppression

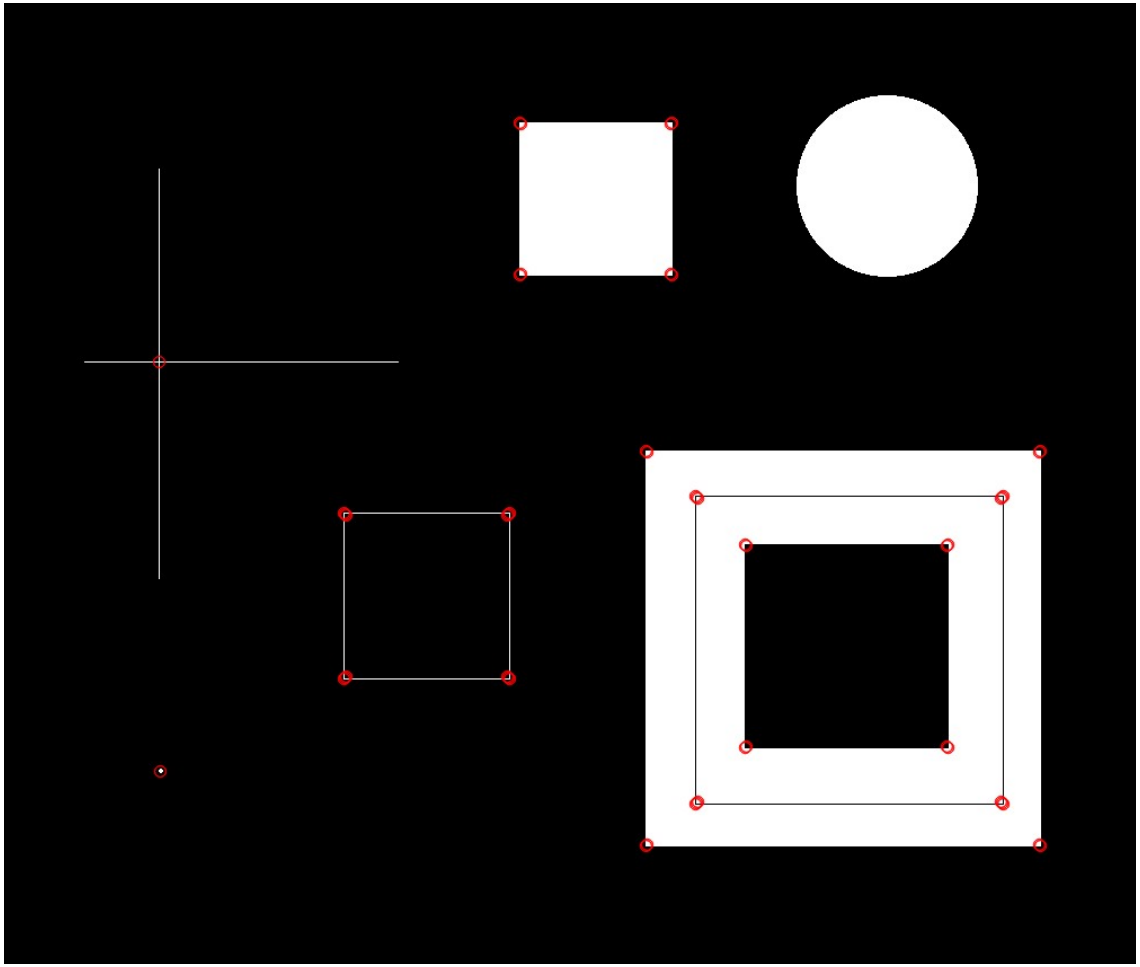
Förstner Operator



Isotropy q

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor
6. weight calculation
7. weight non-max suppression
8. weight thresholding
9. isotropy calculation
10. isotropy non-max suppression
11. isotropy thresholding

Förstner Operator



Keypoints

1. Gradient in x- and y direction
2. $g_x \cdot g_x, g_y \cdot g_y, g_x \cdot g_y$
3. Average (Gaussian Window)
4. Trace of structure tensor
5. Determinant of structure tensor
6. weight calculation
7. weight non-max suppression
8. weight thresholding
9. isotropy calculation
10. isotropy non-max suppression
11. isotropy thresholding
12. Keypoints found

6. Exercise – Given

`int main(int argc, char** argv)`

- Loads image, extract and shows keypoints
- argv[1] == path to image
- argv[2] == scale of kernel (std-dev)

`void showImage(Mat& img, const char* win, int wait, bool show, bool save)`

`img`	image
`win`	window/file name
`wait`	time to wait (0 == wait for key pressed)
`show`	shall the image be shown...
`save`	and/or saved?

- shows/saves image (normalized! [min, max] → [0,255])

`void nonMaxSuppression(Mat& img, Mat& out)`

`img`	input image
`out`	output image

- deletes all non-maxima (maxima = largest value in 4-neighbourhood)

6. Exercise – ToDo

```
void getInterestPoints(Mat& img, double sigma, vector<KeyPoint>& points)
```

`img`	input image
`sigma`	std-dev of filter kernel (first derivative of gaussian)
`Points`	found keypoints

- Computes keypoints using structure tensor
- Needs image gradients: → use functions from previous exercises to calculate directional gradients (convolution with first dev. of gaussian)

On due date: Exam preparation

EXAM

10.02.2012

10:00 - 12:00 o'clock

Room H 2032

EXAM

10.02.2012

10:00 - 12:00 o'clock

Room H 2032

Registration

Case 1: Written exam registration

The proof of written exam registration at the exam office has to be handed in three working days before the exam date.

Written exam registrations have to be handed in also until this date.

EXAM

10.02.2012

10:00 - 12:00 o'clock

Room H 2032

Registration

Case 2: Online exam registration

If you register via QISPOS please draw a print-out (e.g. screenshot) for the online registered exam and hand it in three working days before the exam date.

Please pay attention to the registration deadline on QISPOS.

EXAM

10.02.2012

10:00 - 12:00 o'clock

Room H 2032

Registration

Case 3: Informal / formless registration

If you do not have to register send an email to the office that you would like to attend the exam (marion.dennert@tu-berlin.de).

The email has to arrive 3 working days before the exam at the latest.