

Symmetrierkennung in Theorie und Praxis



***FH Wedel Entwicklung trifft FU
Berlin Forschung***

“Symmetry is a complexity-reducing concept [...]; seek it everywhere”

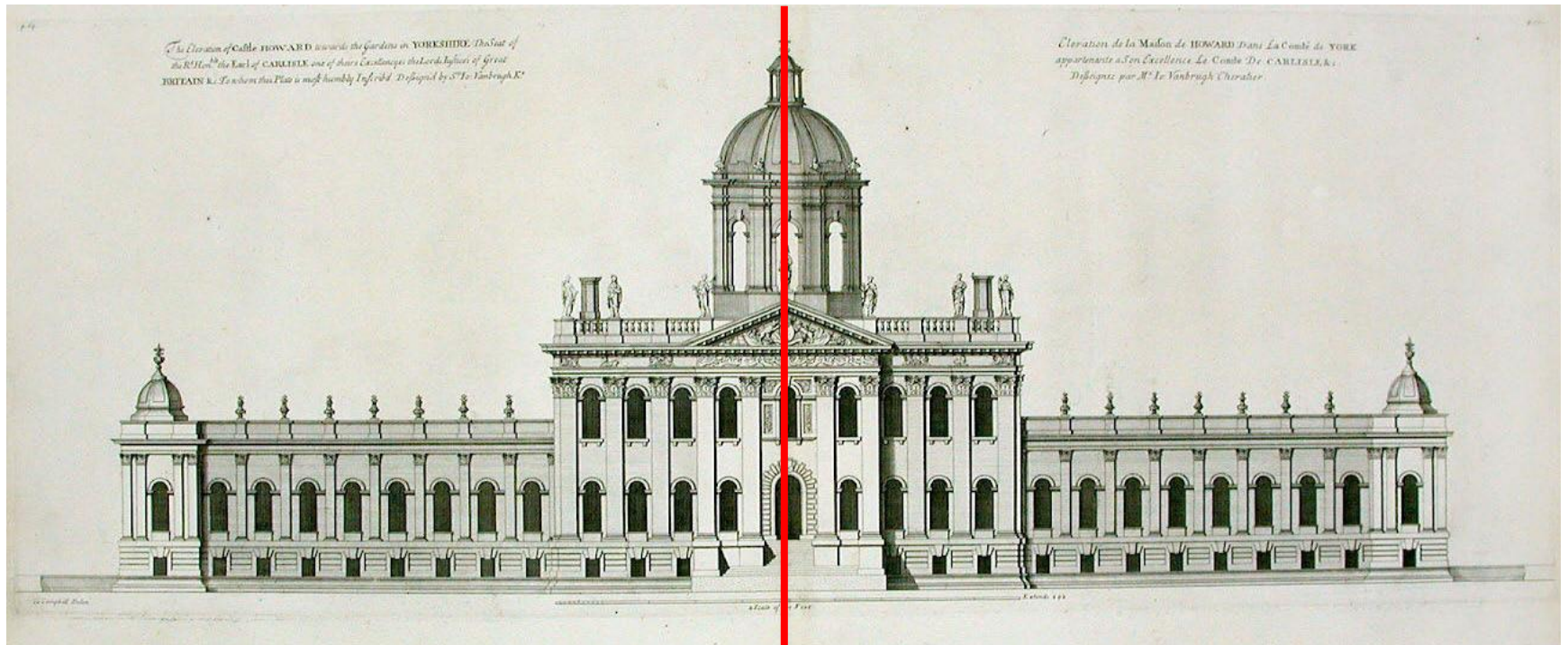
- Alan J. Perlis

221B

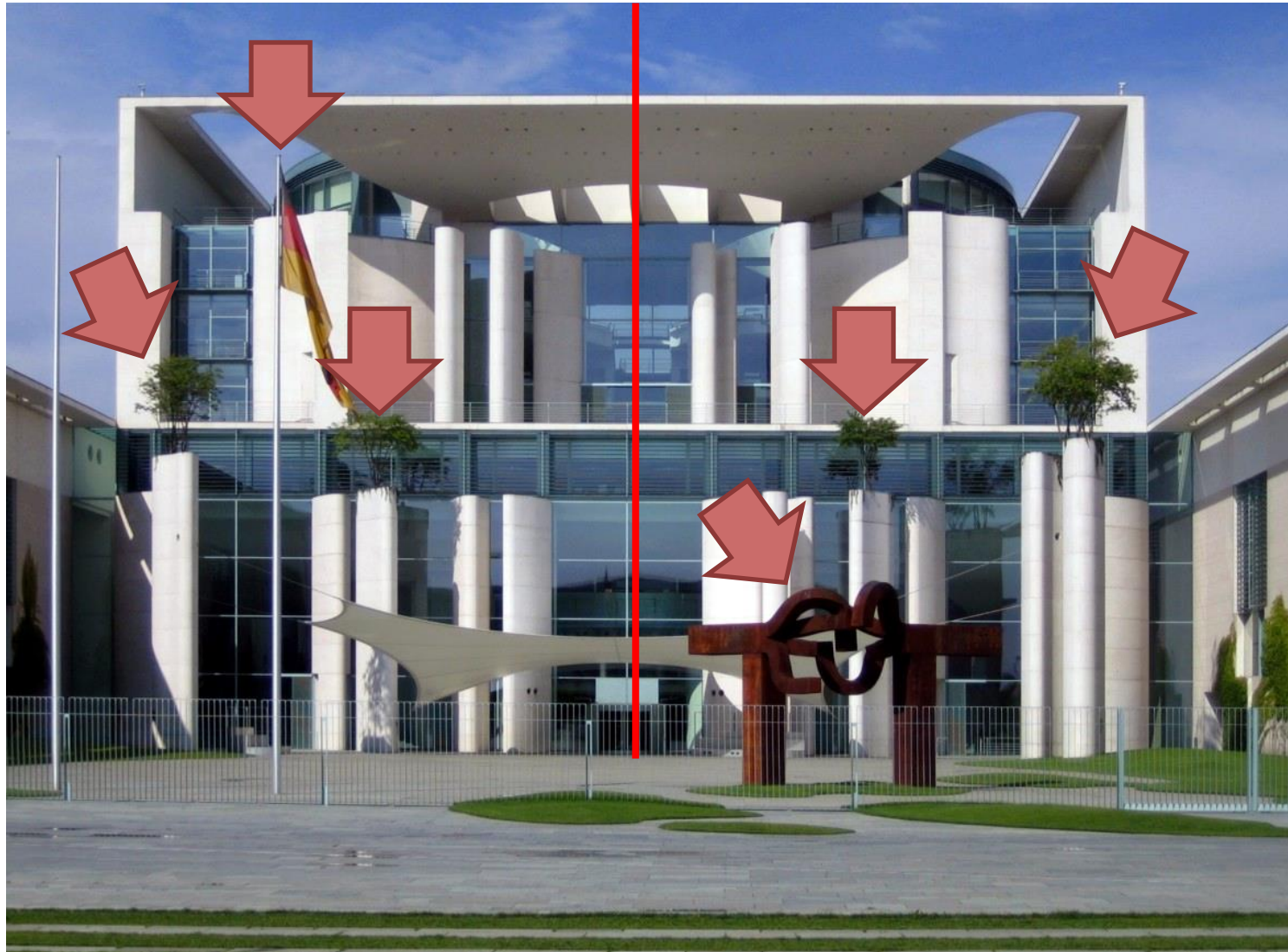


EINFÜHRUNG

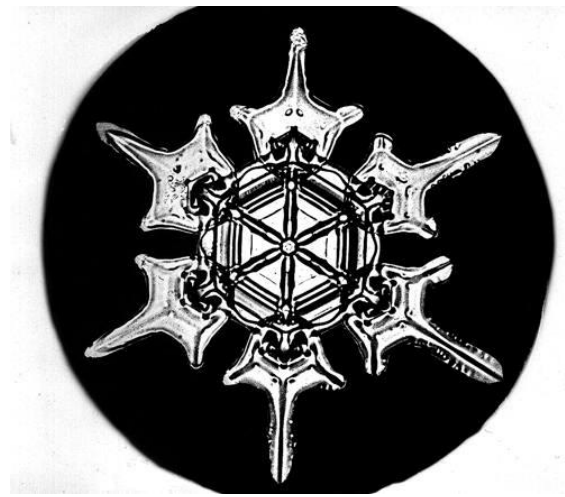
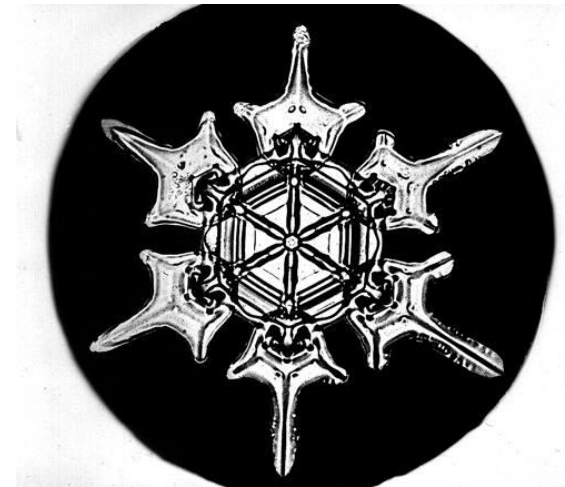
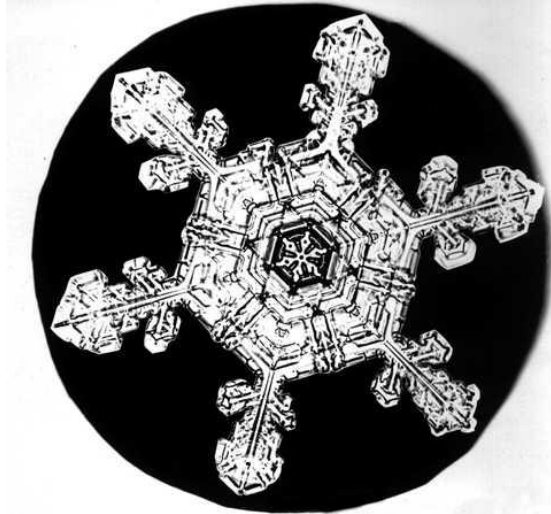
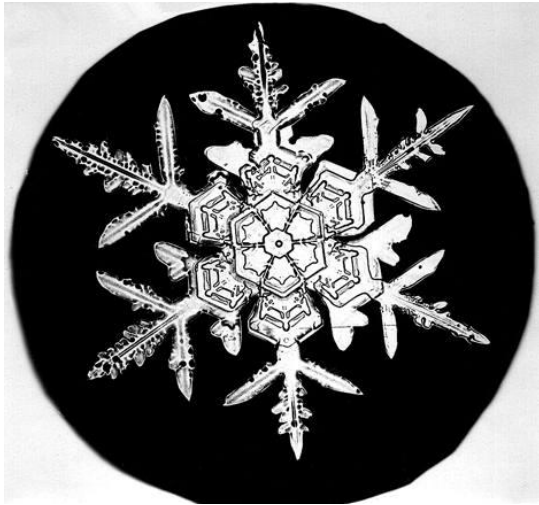
Symmetrien im Alltag



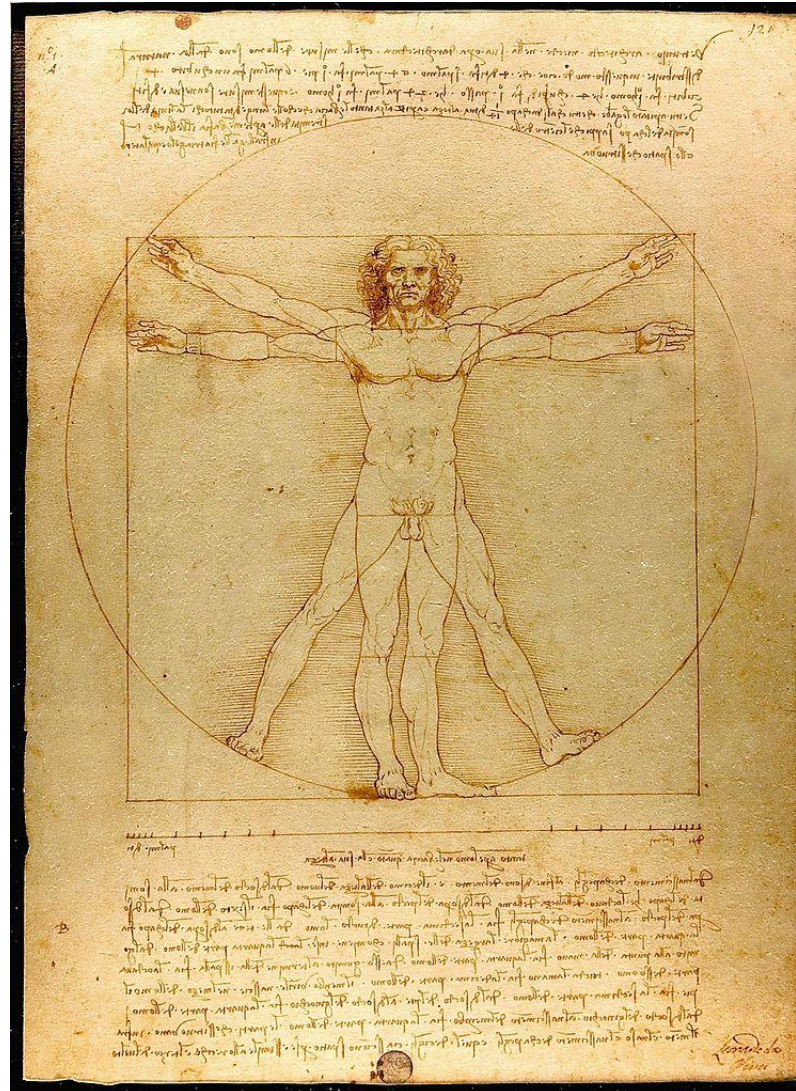
Symmetrien im Alltag



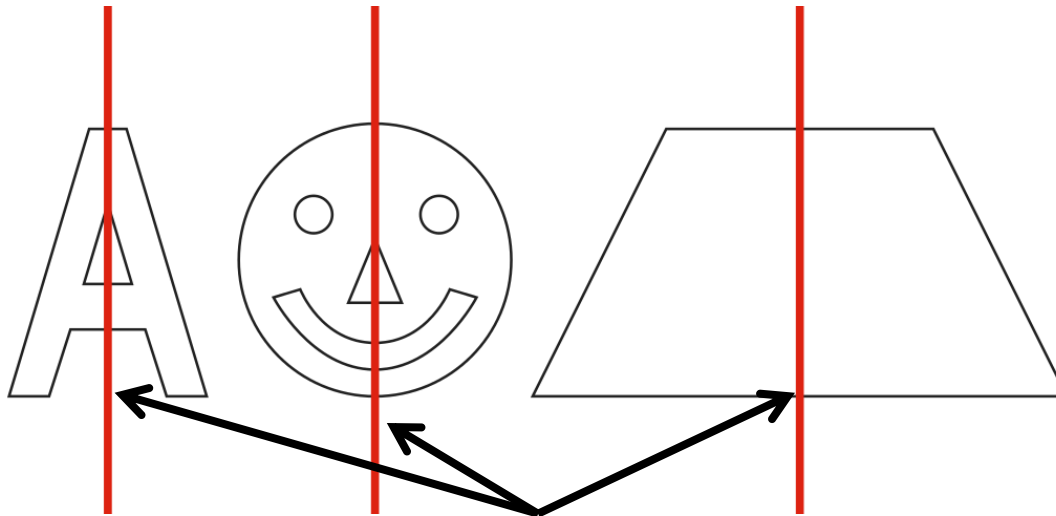
Symmetrien im Alltag



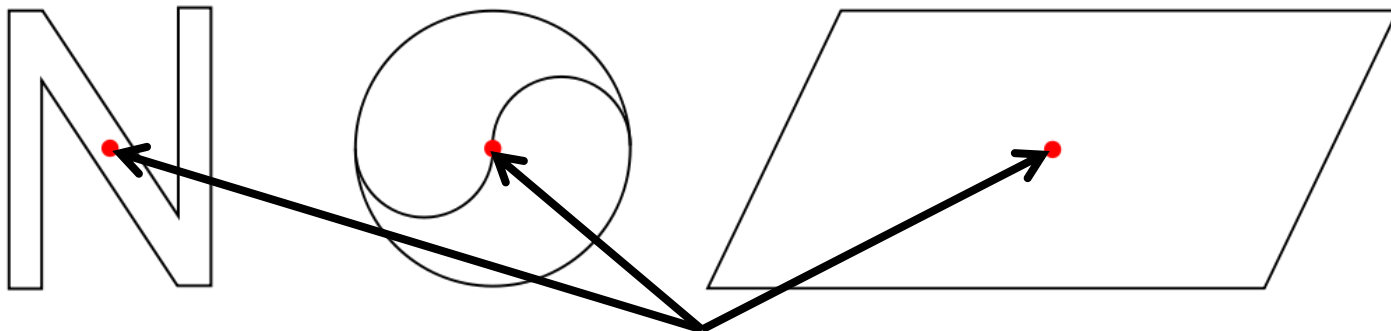
Symmetrien im Alltag



Wiederholung Analysis



Achse



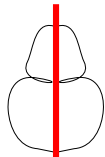
Punkt

Achsen- und Rotationssymmetrie

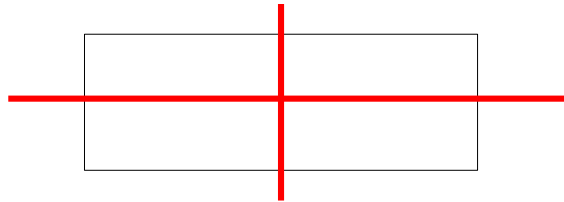
Diedergruppe D_n

n Spiegelachsen durch den Mittelpunkt

D_1 = Einzelne Achsenspiegelung



D_2 = Nicht quadratisches Rechteck



D_n = Regelmäßiges n-Eck

Zyklische Gruppe C_n

Alle Drehungen um einen Punkt um Vielfache
von $\frac{360^\circ}{n}$

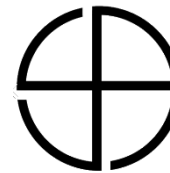
C_1 = Komplettsymmetrisches Objekt

C_2 = Punktspiegelung

C_3 = Triskele



C_4 = Swastika

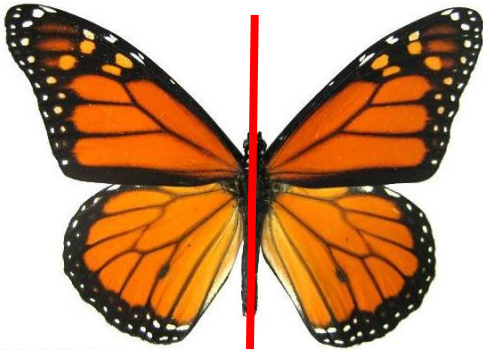


$$D_n \rightarrow C_n$$

Symmetrie

Symmetrie ist die Gleichgültigkeit bezüglich bestimmten Kongruenztransformationen

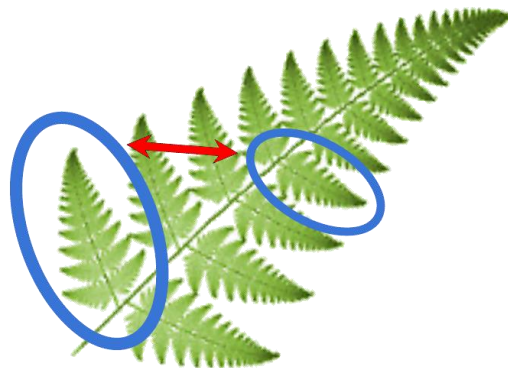
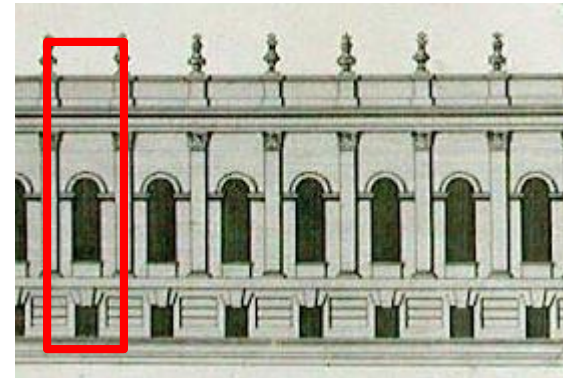
Spiegelung



Rotation



Translation



Spiegelung
Rotation
Translation
Skalierung

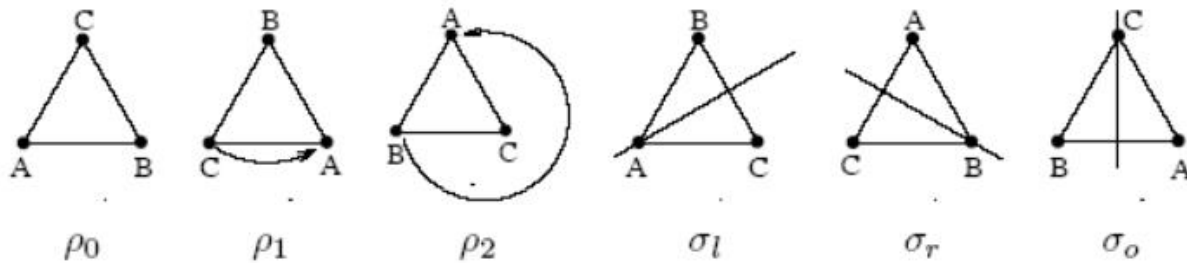
Symmetriegruppe

In der mathematischen Gruppentheorie ist die Symmetriegruppe eines geometrischen Objektes die Gruppe, die aus der Menge aller Kongruenzabbildungen besteht, die das Objekt auf sich selbst abbilden, zusammen mit der Verkettung von Abbildungen als Gruppenoperation.

Symmetriegruppe

In der mathematischen **Gruppentheorie** ist die Symmetriegruppe eines geometrischen Objektes die Gruppe, die aus der **Menge** aller **Kongruenzabbildungen** besteht, die das Objekt auf sich selbst abbilden, zusammen mit der **Verkettung** von Abbildungen als **Gruppenoperation**.

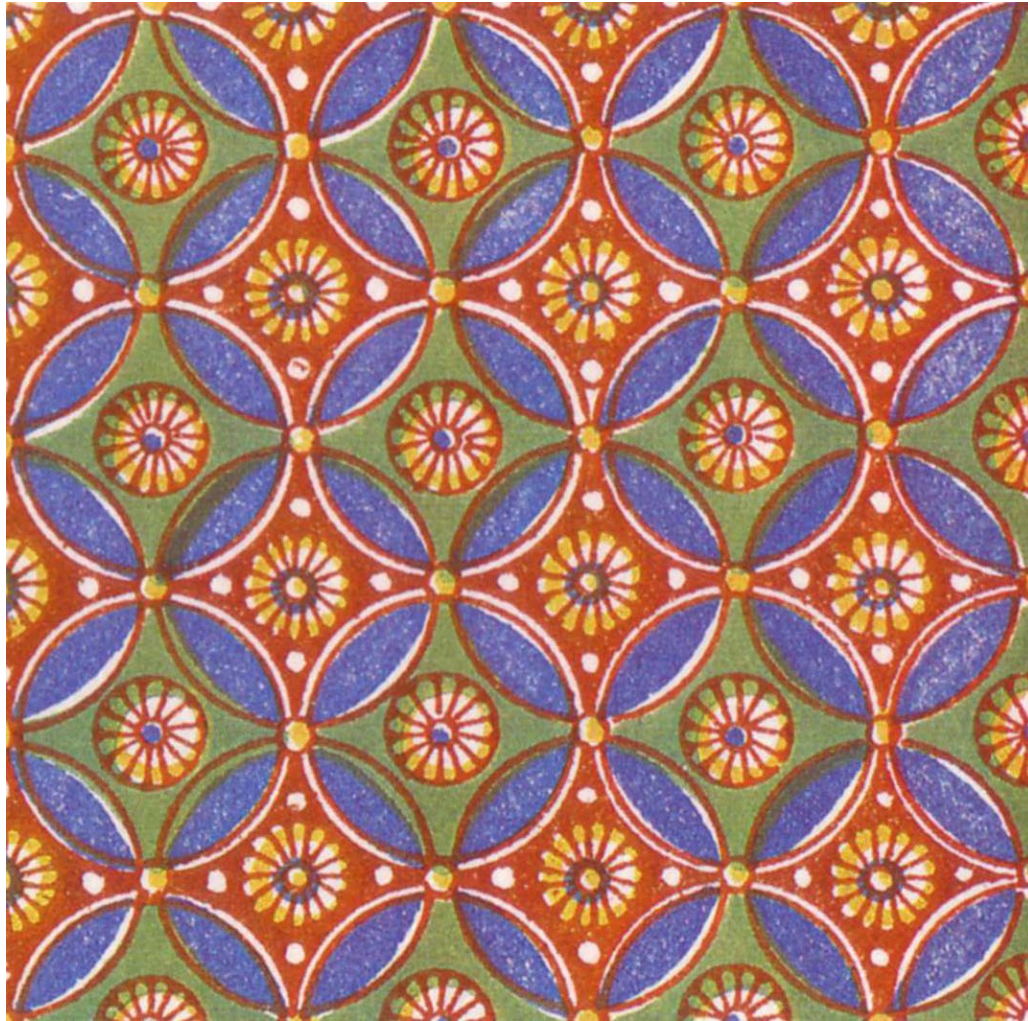
Endliche Symmetriegruppen



(S_3, \circ) :

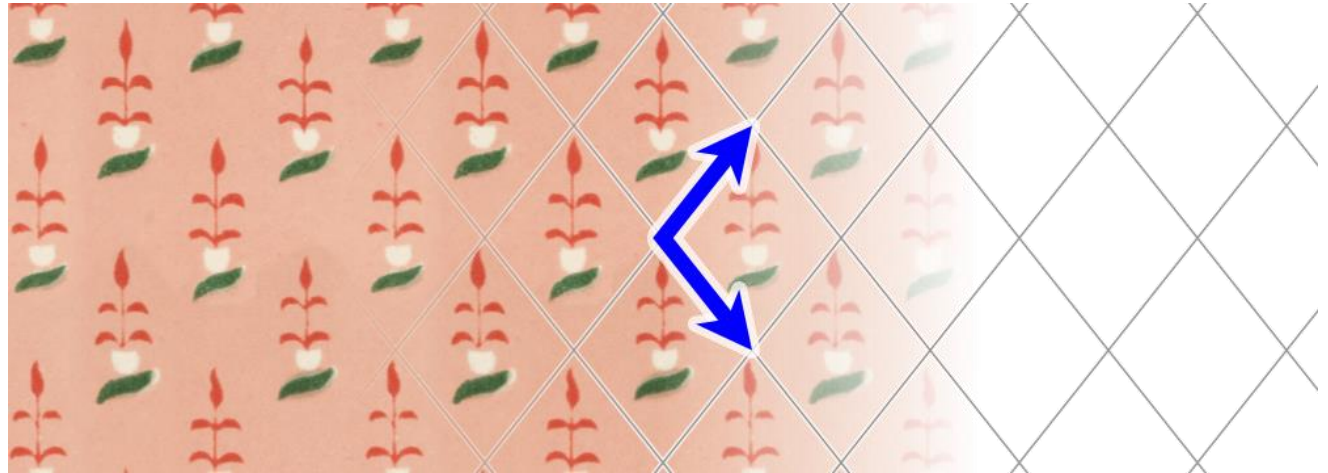
\circ	ρ_0	ρ_1	ρ_2	σ_l	σ_r	σ_o
ρ_0	ρ_0	ρ_1	ρ_2	σ_l	σ_r	σ_o
ρ_1	ρ_1	ρ_2	ρ_0	σ_o	σ_l	σ_r
ρ_2	ρ_2	ρ_0	ρ_1	σ_r	σ_o	σ_l
σ_l	σ_l	σ_r	σ_o	ρ_0	ρ_1	ρ_2
σ_r	σ_r	σ_o	σ_l	ρ_2	ρ_0	ρ_1
σ_o	σ_o	σ_l	σ_r	ρ_1	ρ_2	ρ_0

Unendliche Symmetriegruppen

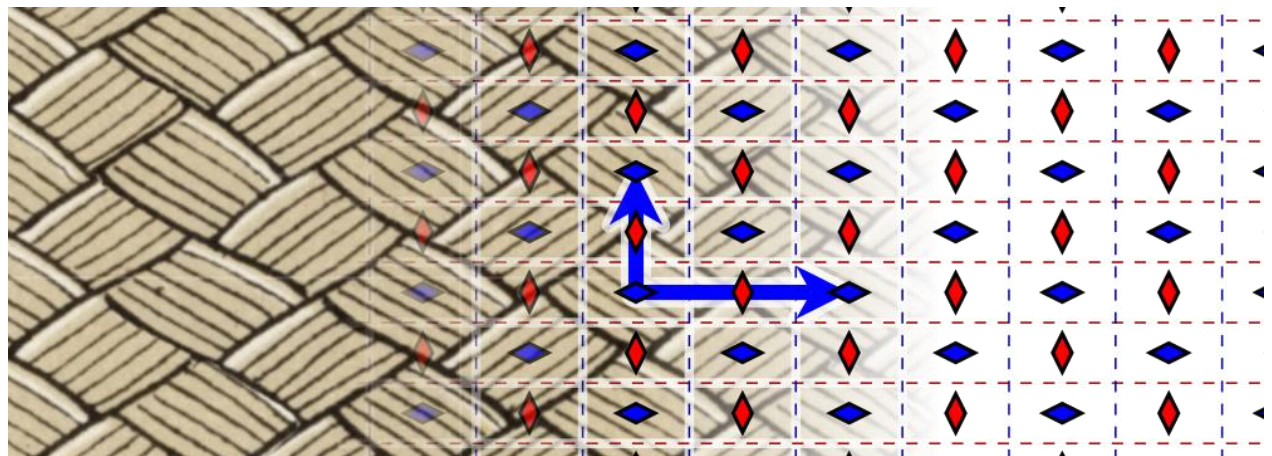


Unendliche Symmetriegruppen

p1
(Translation)



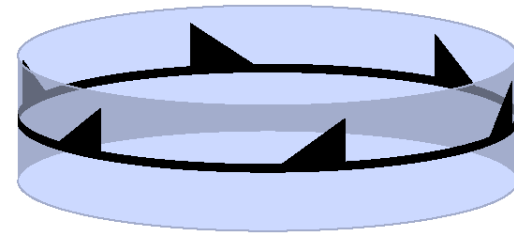
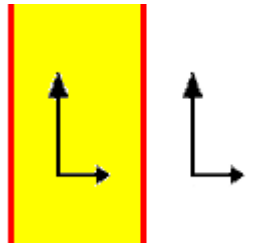
p2gg
(2 Rotationszentren +
Verschiebung)



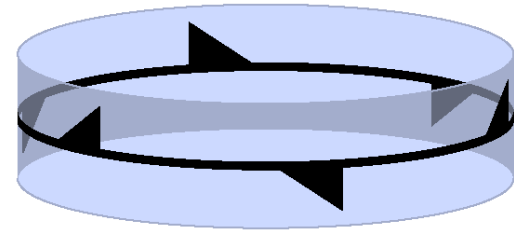
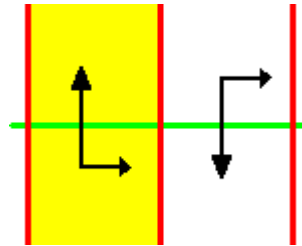
Frieze Gruppe

Unendlich, aber nur in eine Dimension

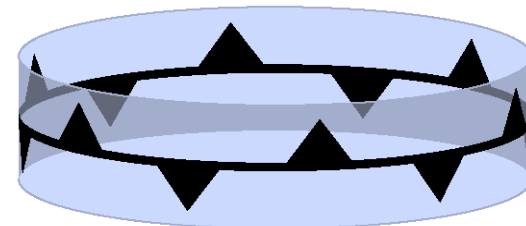
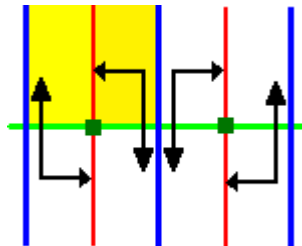
p1
(Translation)



p11g
(Translation,
Spiegelung)



p2mg
(Translation,
Spiegelung,
Rotation)





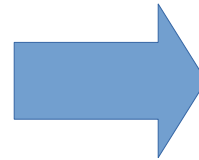
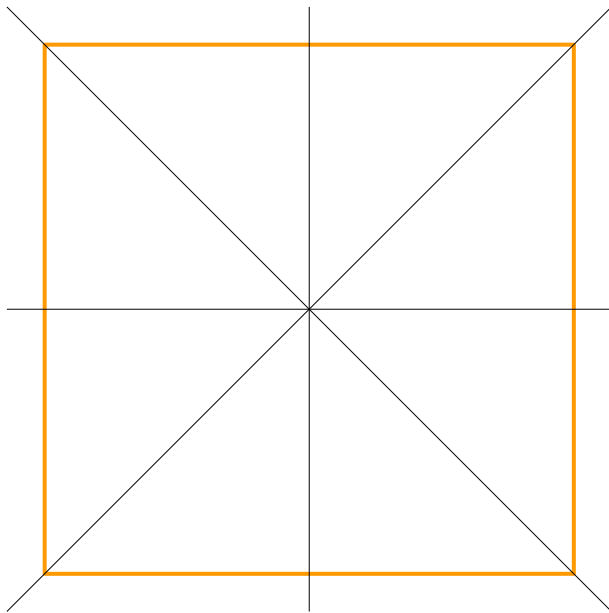
ANWENDUNG

Fragestellungen

1. Erkennung einer Symmetrie
2. Beschreibung der Symmetrie (Approximation)

Erkennung

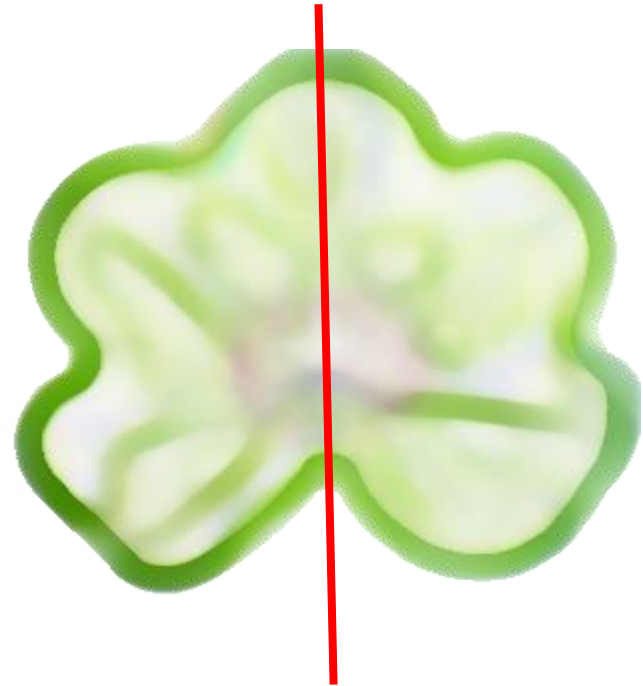
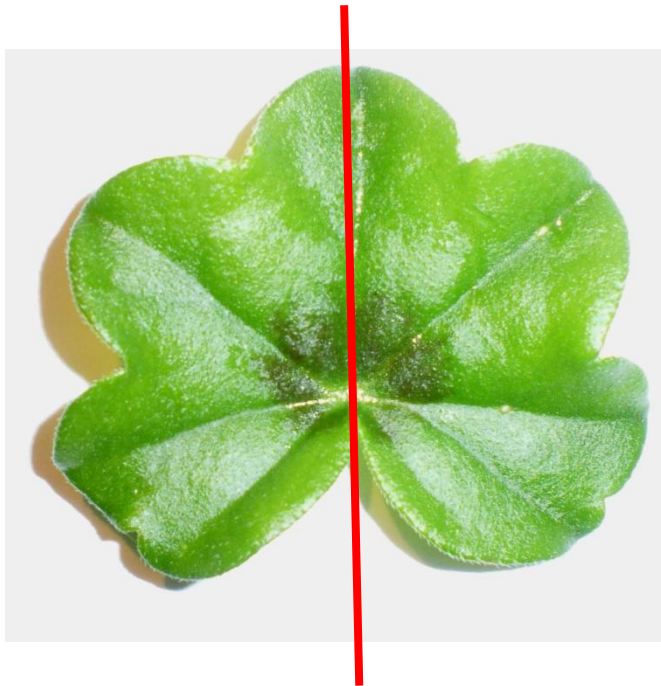
- Gegeben: Bild
- Gesucht: Symmetriegruppe



D_4

Probleme bei der Erkennung

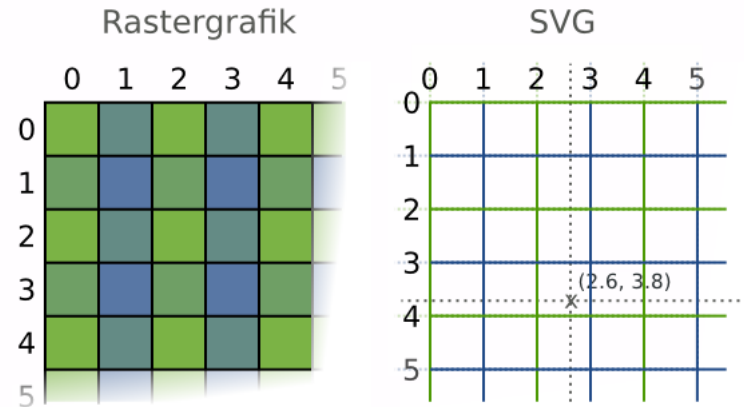
- Bild nicht „mathematisch“ symmetrisch
→ nur intuitiv erkennbar
- Weiche Erkennung erforderlich



Eingabe für die Erkennung

- Vektorgrafik
 - Geometrische Beschreibung von Pfaden
 - Stetige Werte

- Pixelgrafik
 - (Farb)Wert pro Pixel
(Zelle in einem 2D-Raster)



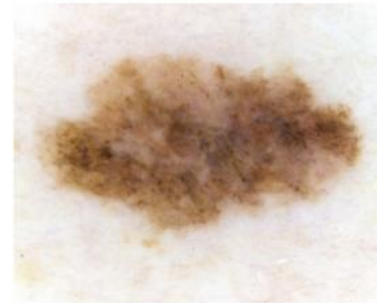
- Punktwolke (reduzierte Vektorgrafik)
 - „Liste von Koordinaten“

Beispiele für Einsatzbereiche

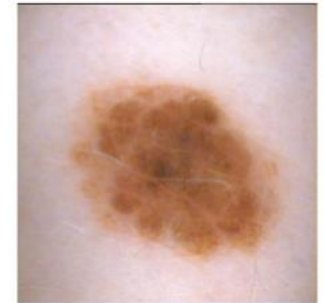
- Objekterkennung in Bildern
- Astronomische Objekte klassifizieren
- Medizin: Muttermale von Hautkrebs unterscheiden
- Bildkompression



Distinguishing **Melanomas** from **Moles**



Melanoma

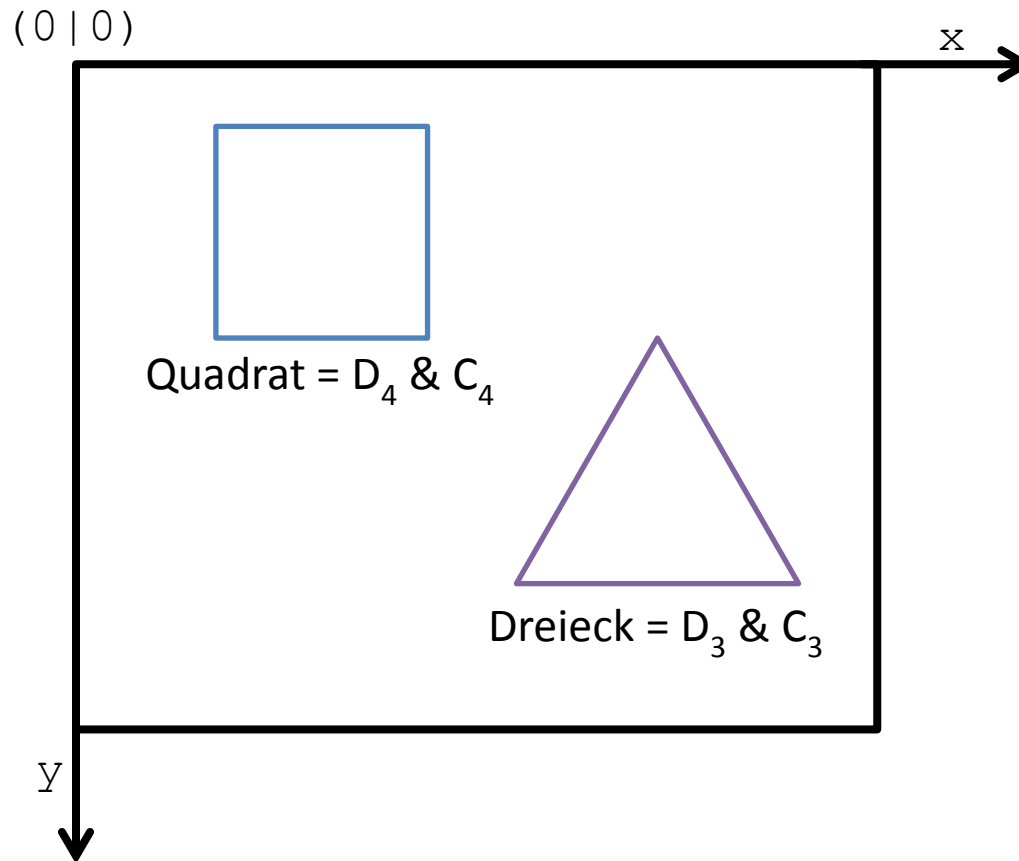


Mole

⇒ A. Rodriguez, J. Stangl, C. Shakiban

Objekterkennung

Keinerlei Symmetrie im Gesamtbild



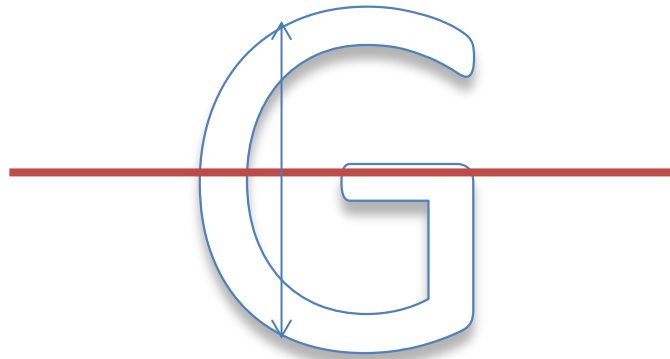
Approximation

- Gegeben:
 - Bild
 - Symmetrie-Eigenschaft (z. B. Symmetriegruppe)
- Gesucht:
 - Symmetrie-Beschreibung (z. B. konkrete Achsen)
 - Soll vorhandene Symmetrie möglichst genau annähern

Approximation

- NP-schwer für Punkte, die „zu“ nah beisammen liegen
- Ursprünglich von S. Iwanowski in seiner Doktorarbeit bewiesen
„Approximate congruence and symmetry detection in the plane“ (1990)
- Verfeinert von C. Dieckmann
„Symmetry Detection and Approximation“ (2012)

Beispiel

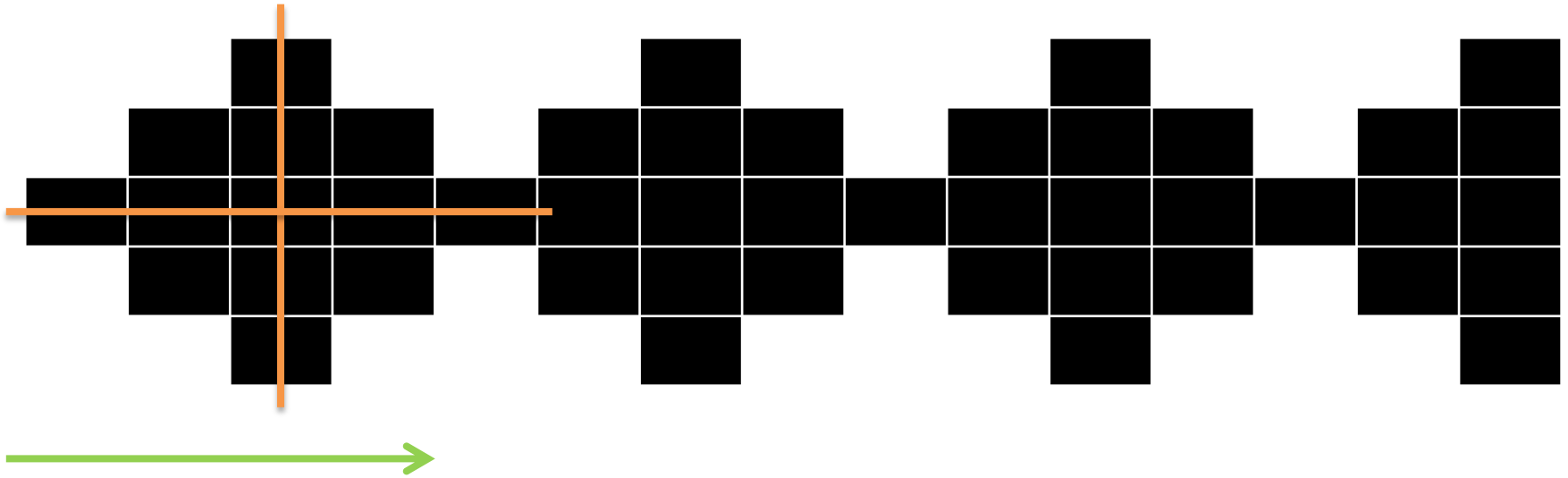


Erkannte Symmetriegruppe: D_1



ERKENNUNG (1)

Frieze Patterns



Rotation?

Nein

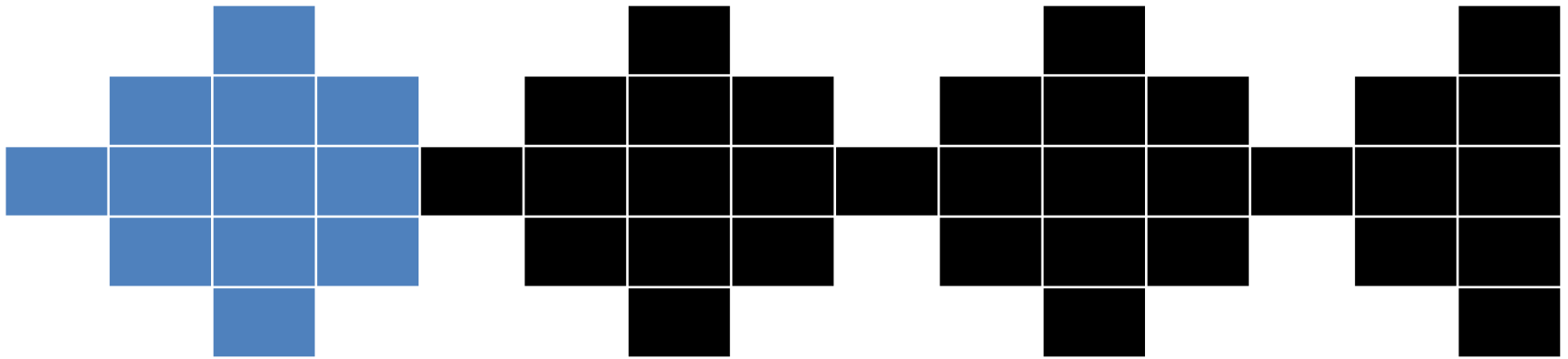
Spiegelung?

Nein

Translation?

Ja

Frieze Patterns

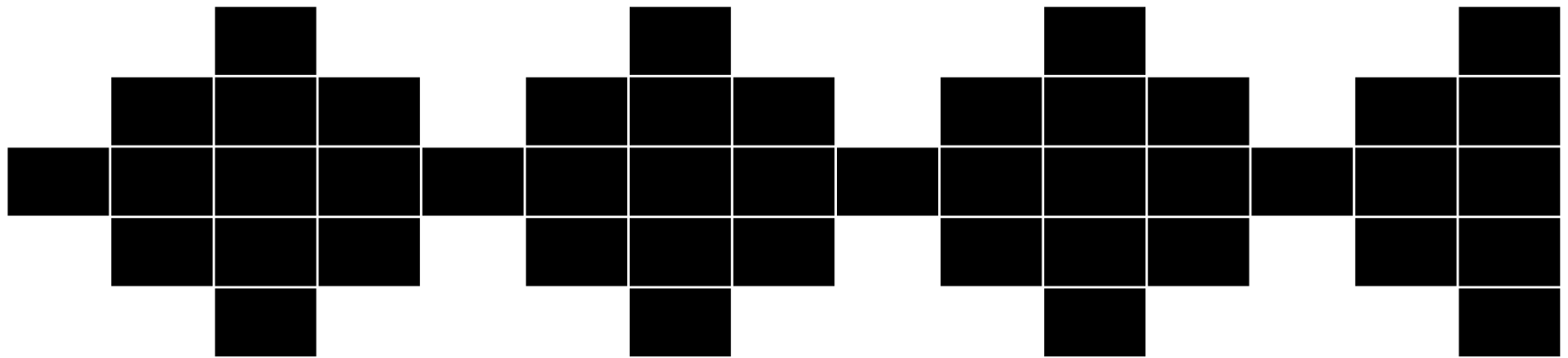


Muster P



Gesamtes Rasterbild T

Frieze Patterns und String Matching

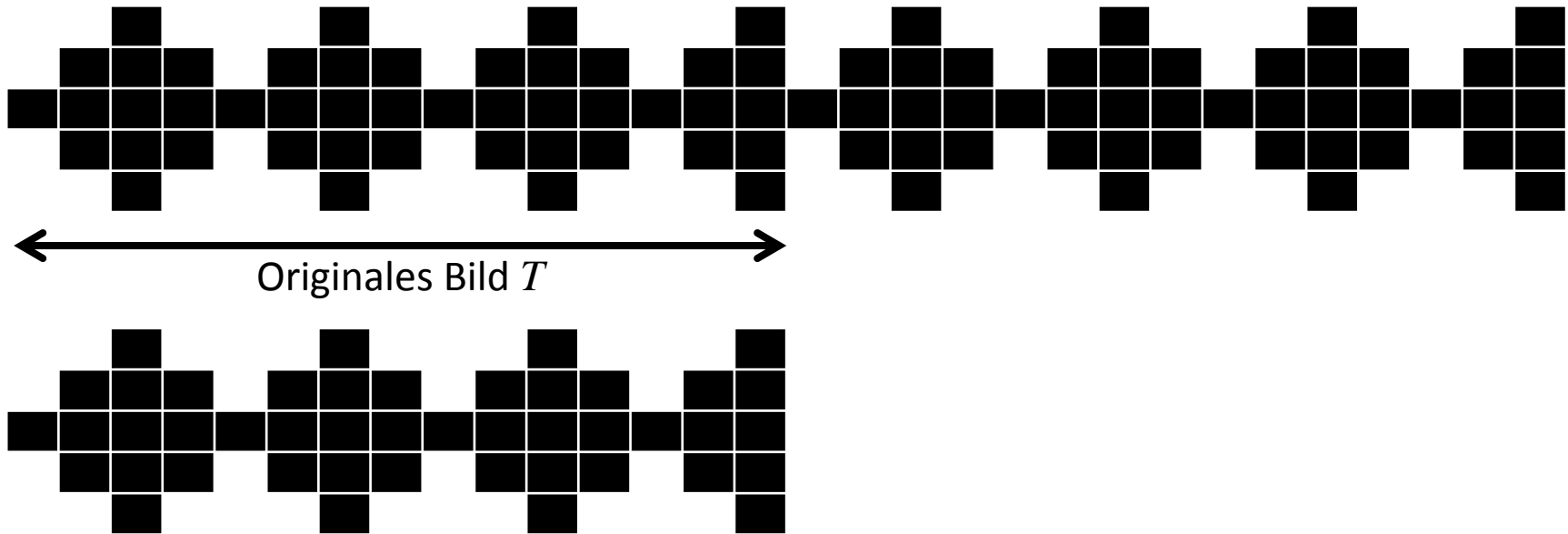


	m															→
n	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	
	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	
	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	

Alphabet $\Sigma = \{0, 1\}$

Zeile R_i wobei $1 \leq i \leq m$ und $R_i \subset T_i \in \Sigma^*$

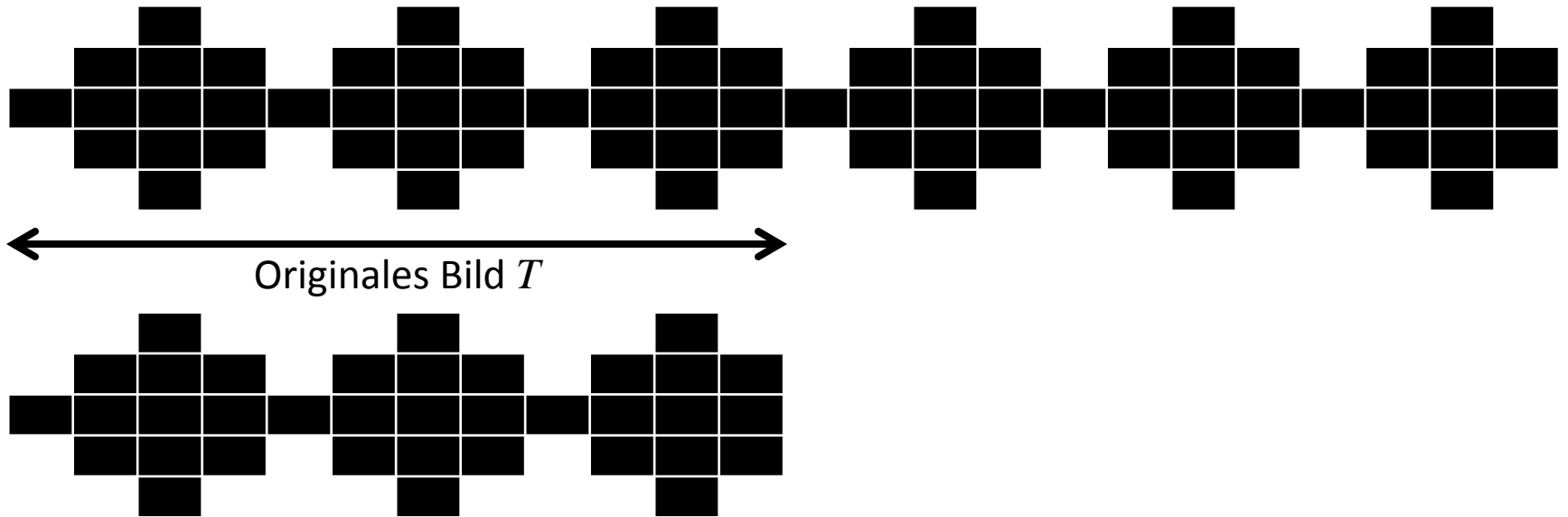
Breite des Musters ermitteln



- 1) **For** R_i, R_i' **in** $T, T + T$
- 2) $W_i = \mathbf{Calc}$ Second match of R_i in R_i'
- 3) $P_{width} = \mathbf{lcm}(W_i, P_{width})$

Nur Erkennung von ganzzahligen Wiederholungen

Breite des Musters ermitteln



- 1) **For** R_i, R_i' **in** $T, T + T$
- 2) $W_i = \mathbf{Calc}$ Second match of R_i in R_i'
- 3) $P_{width} = \mathbf{lcm}(W_i, P_{width})$

Laufzeit des Verfahrens

```
1) For  $R_i, R_i'$  in  $T, T + T$   
2)  $W_i = \text{Calc}$  Second match of  $R_i$  in  $R_i'$   
3)  $P_{\text{width}} = \text{lcm}(W_i, P_{\text{width}})$ 
```

1) $T + T \in O(1)$
#Durchläufe = N (Anzahl der Zeilen)

2) String Matching $\in O(|T| + |P|)$
 $|T| = 2 * M$ (Anzahl der Spalten)
 $|P| = M$

Insgesamt also $O(m)$ Aufwand

3) $\text{gcd} \in O(\log x)$ wobei $x = \text{Anzahl Bits für Parameter}$

$$O(n * (m + \log x)) \subseteq O(mn)$$

Umgang mit Graustufen



Def $eq(l, r)$
 $l=r$



Def $eq(l, r, \delta)$
 $|l - r| < \delta$

Unscharfes Matching

Bisher: Lokale Ähnlichkeit, Jetzt: Globale Ähnlichkeit

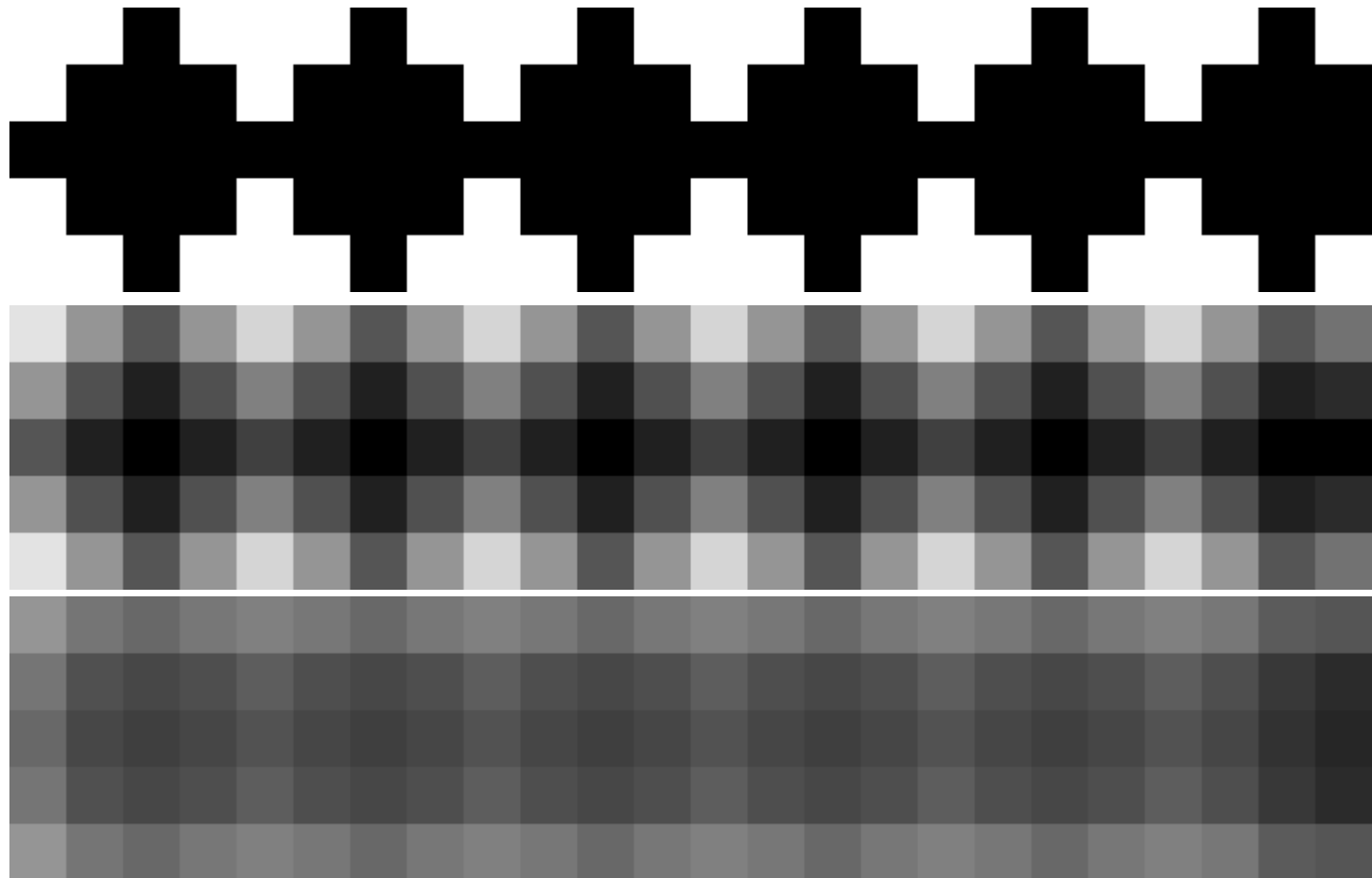
$$\text{filter}(j) = \frac{1}{m} \sum_{i=1}^m |R[i] - R'[i + j]|$$

- 1) **For** R_i, R_i' **in** $T, T + T$
- 2) **For** pos **in** R_i
- 3) $R_i[\text{pos}] = \text{filter}(\text{pos})$
- 4) W_i = Second match of R_i in R_i' with δ
- 5) $P_{\text{width}} = \text{lcm}(W_i, P_{\text{width}})$

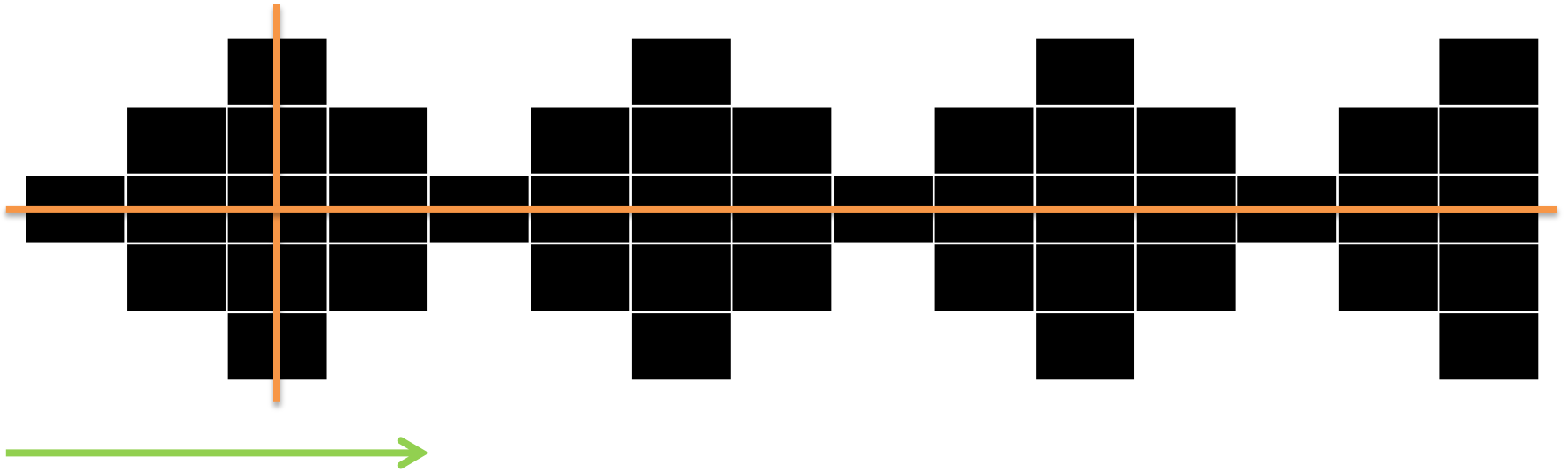
$$O(n * (m^2 + m + \log x)) \subseteq O(nm^2)$$

Unscharfes Matching

Bisher: Lokale Ähnlichkeit, Jetzt: Globale Ähnlichkeit



Zusammenfassung: Frieze Pattern Matching



Rotation?

Nein

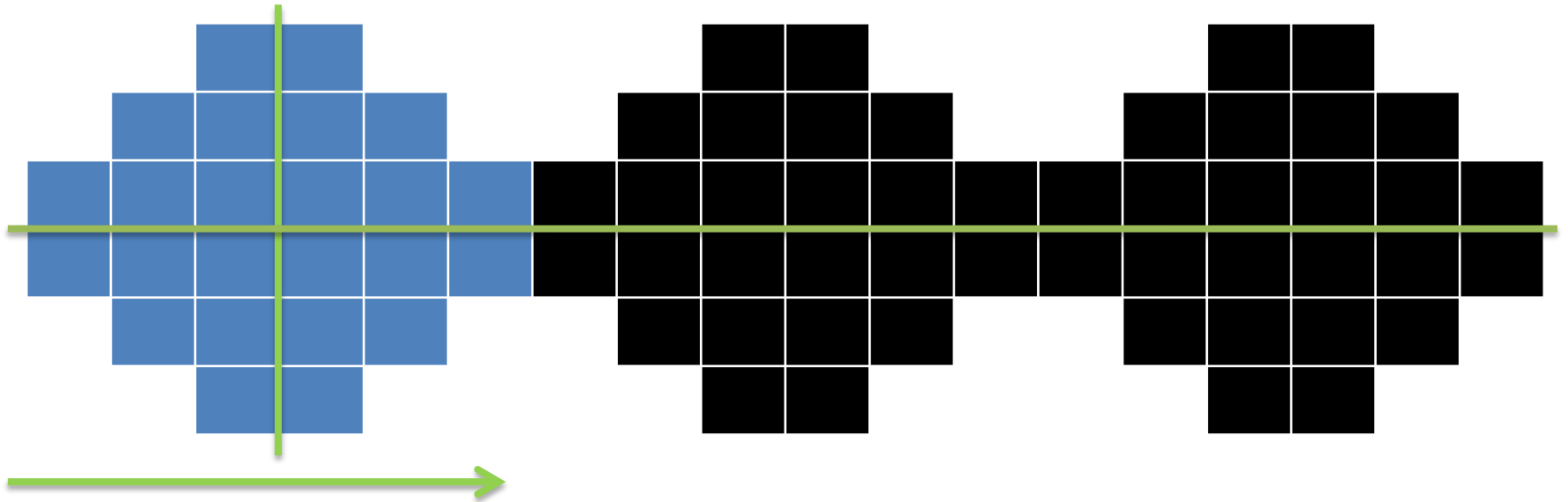
Spiegelung?

Nein

Translation?

Ja

Zusammenfassung: Frieze Pattern Matching



Rotation?

Ja

Spiegelung?

Ja

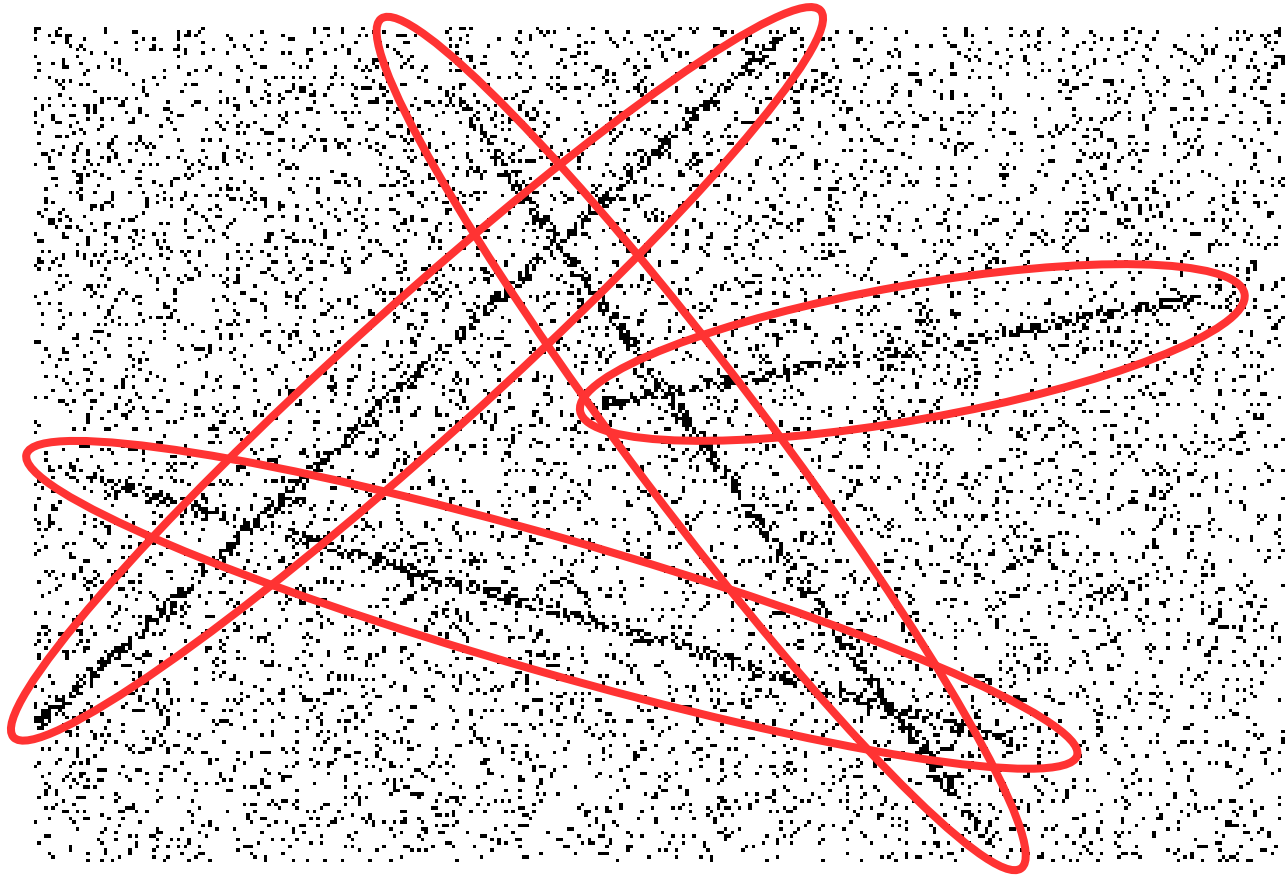
Translation?

Ja

A black and white photograph of a man in profile, wearing a textured cap and a patterned shirt. He is holding a magnifying glass to his eye, looking intently at a wall on the left. The wall has a rough, textured surface with some faint markings. The lighting is dramatic, highlighting the man's face and the magnifying glass.

ERKENNUNG (2)

Hough Transformation



Hough Transformation

- Methode von Paul V. C. Hough
 - US-Patent 3,069,654 (1962)
„Method and means for recognizing complex patterns“
- allgemeiner Ansatz beliebige, parametrisierbare Formen zu finden

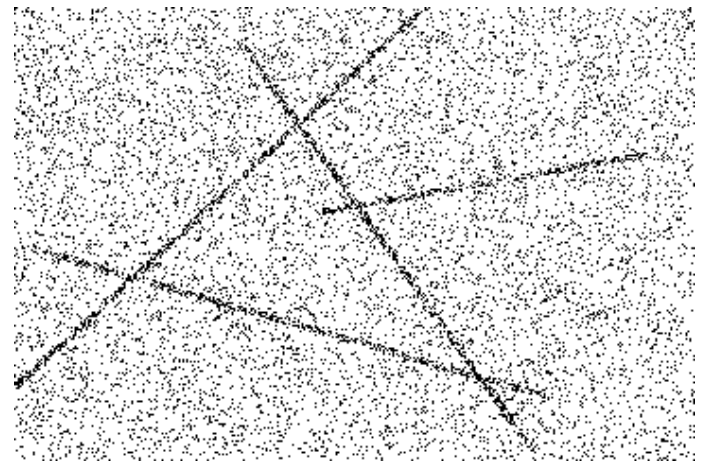
Linienerkennung

Hough-Transformation häufig hierfür eingesetzt

Binärbild

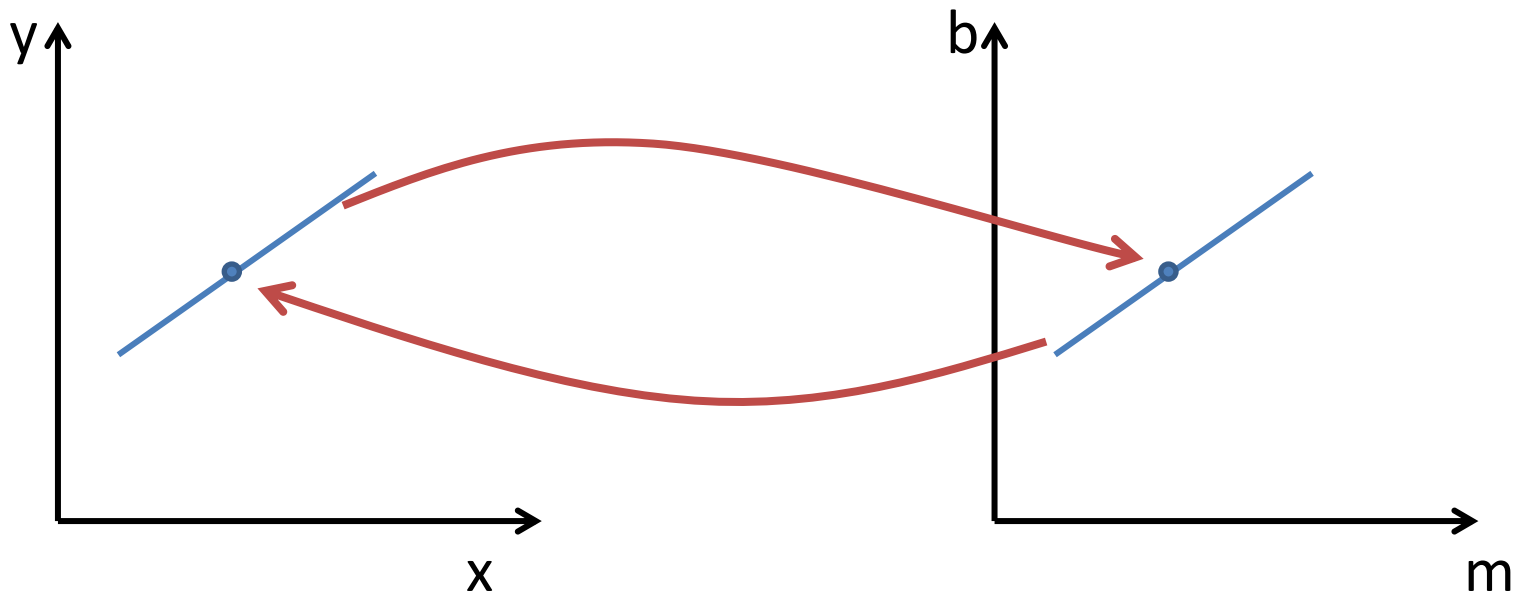


Linie, die möglichst viele Punkte überdeckt



Bildraum \leftrightarrow Parameterraum

Bildraum (x , y)		Parameterraum (m , b)	
Punkt	(x_i , y_i)	$b = -m \cdot x_i + y_i$	Gerade
Gerade	$y = m_j \cdot x + b_j$	(n_j , b_j)	Punkt

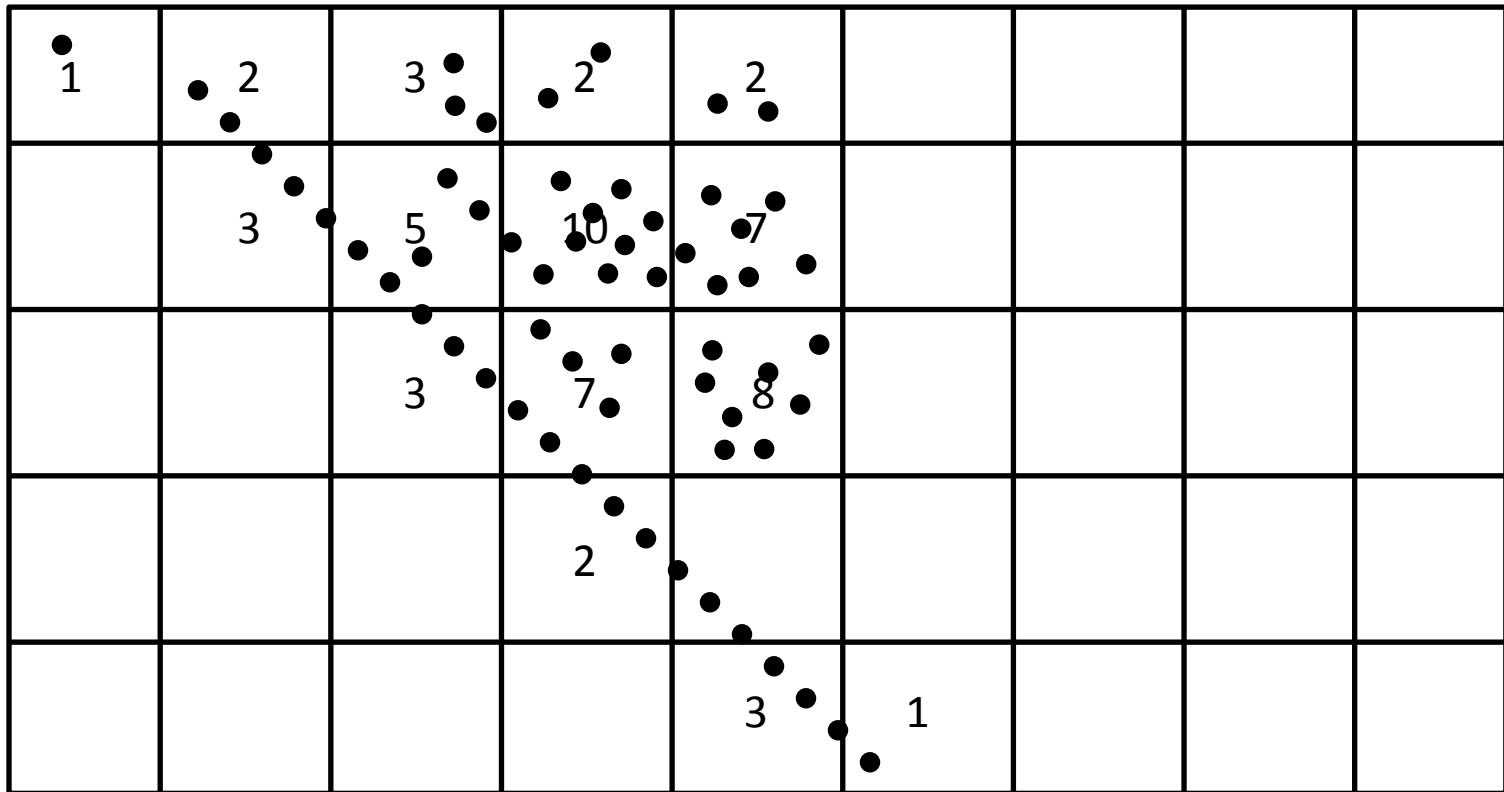


Parameterraum

- Problem: Beispiele für Parameter einer Linie nicht genau gleich
- daher: diskretisieren
- Datenstruktur: 2D-Array

		1					
1		1	3	1			
		4	6	2			1
		1	1	1			
					1		
2							

Diskretisierung

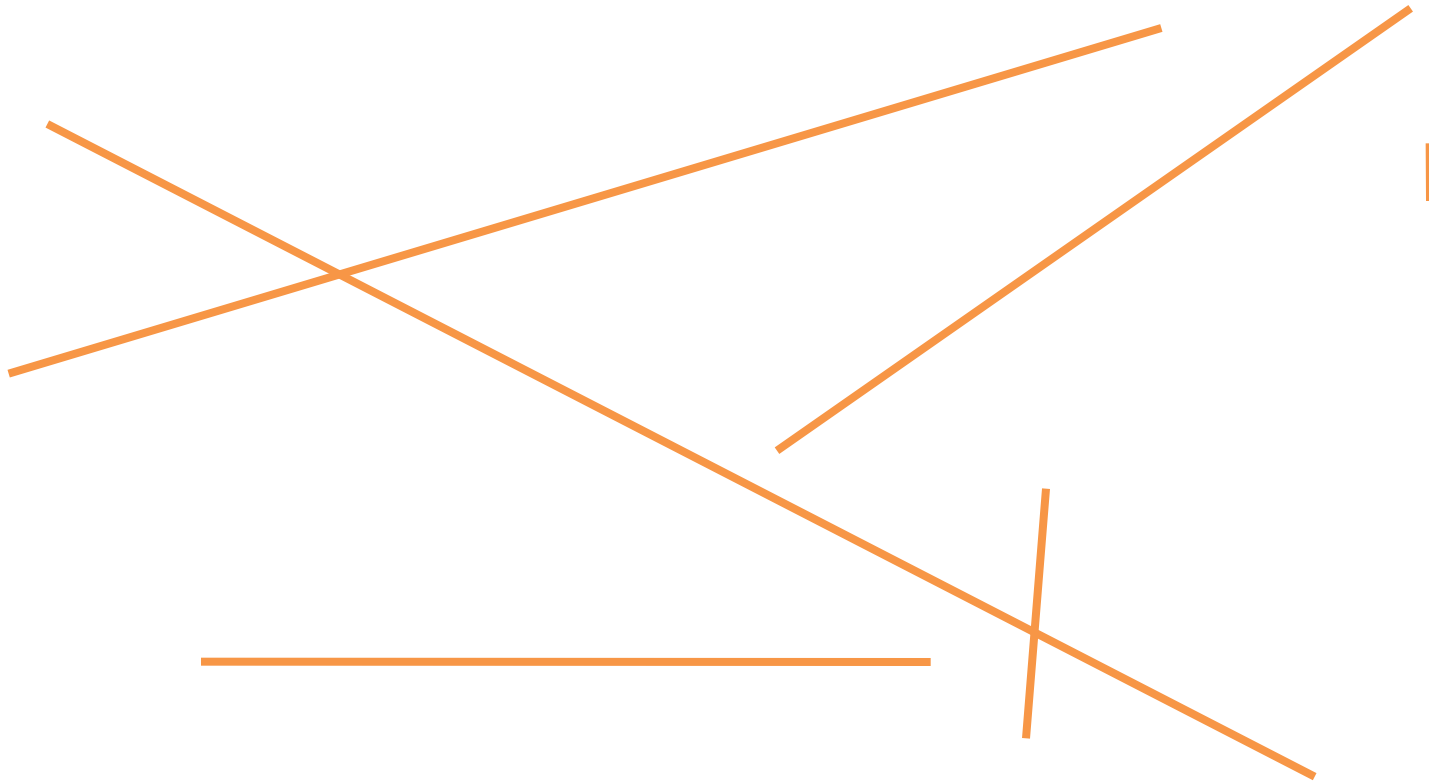


Geradenrepräsentation

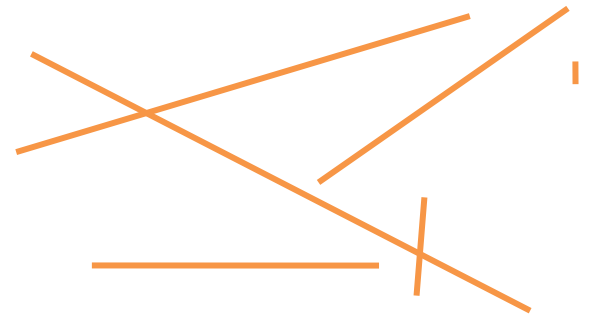
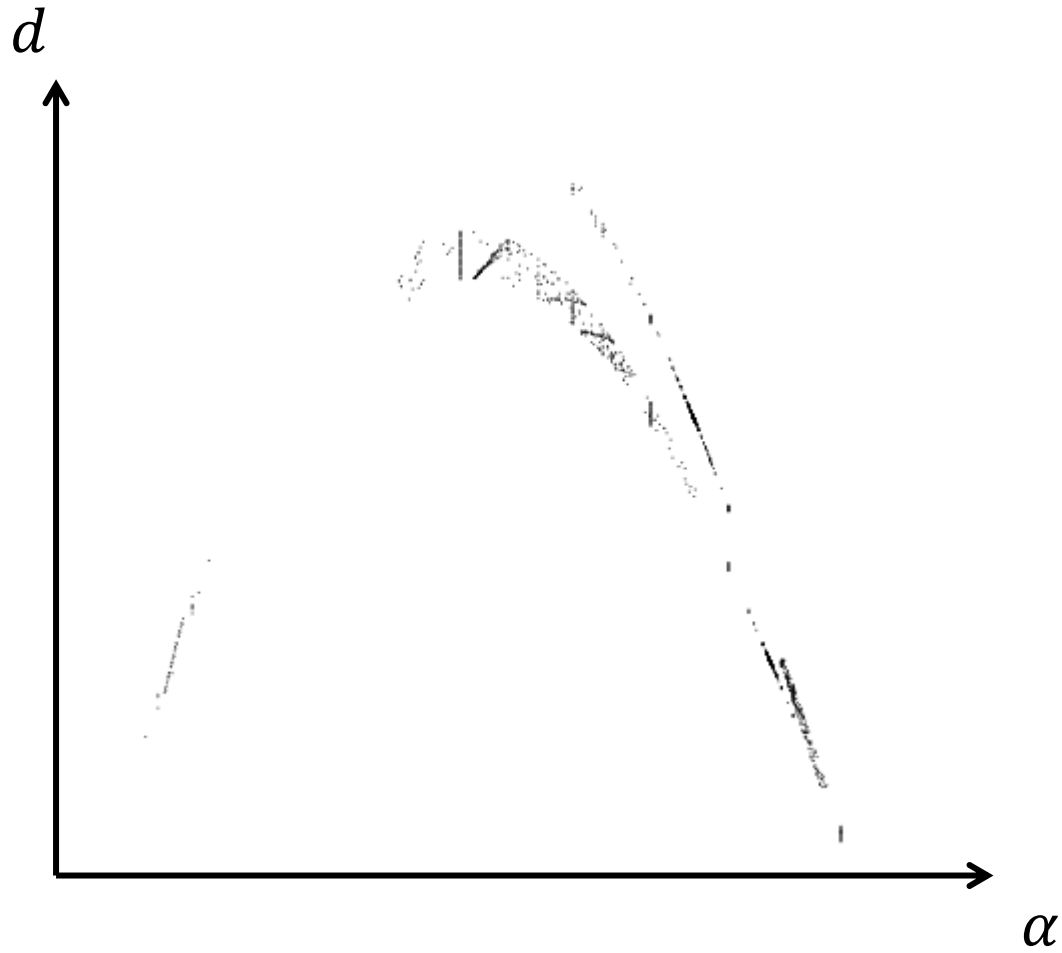
- bisher: $y = m \cdot x + b$
 - vertikale Geraden?

- besser: Hesse'sche Normalform
 - Winkel, Distanz vom Ursprung (Polarkoordinaten)

Beispiel

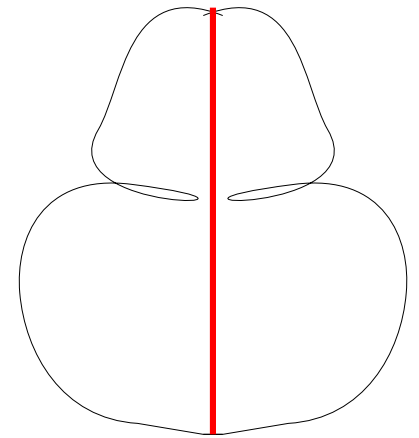


Resultat

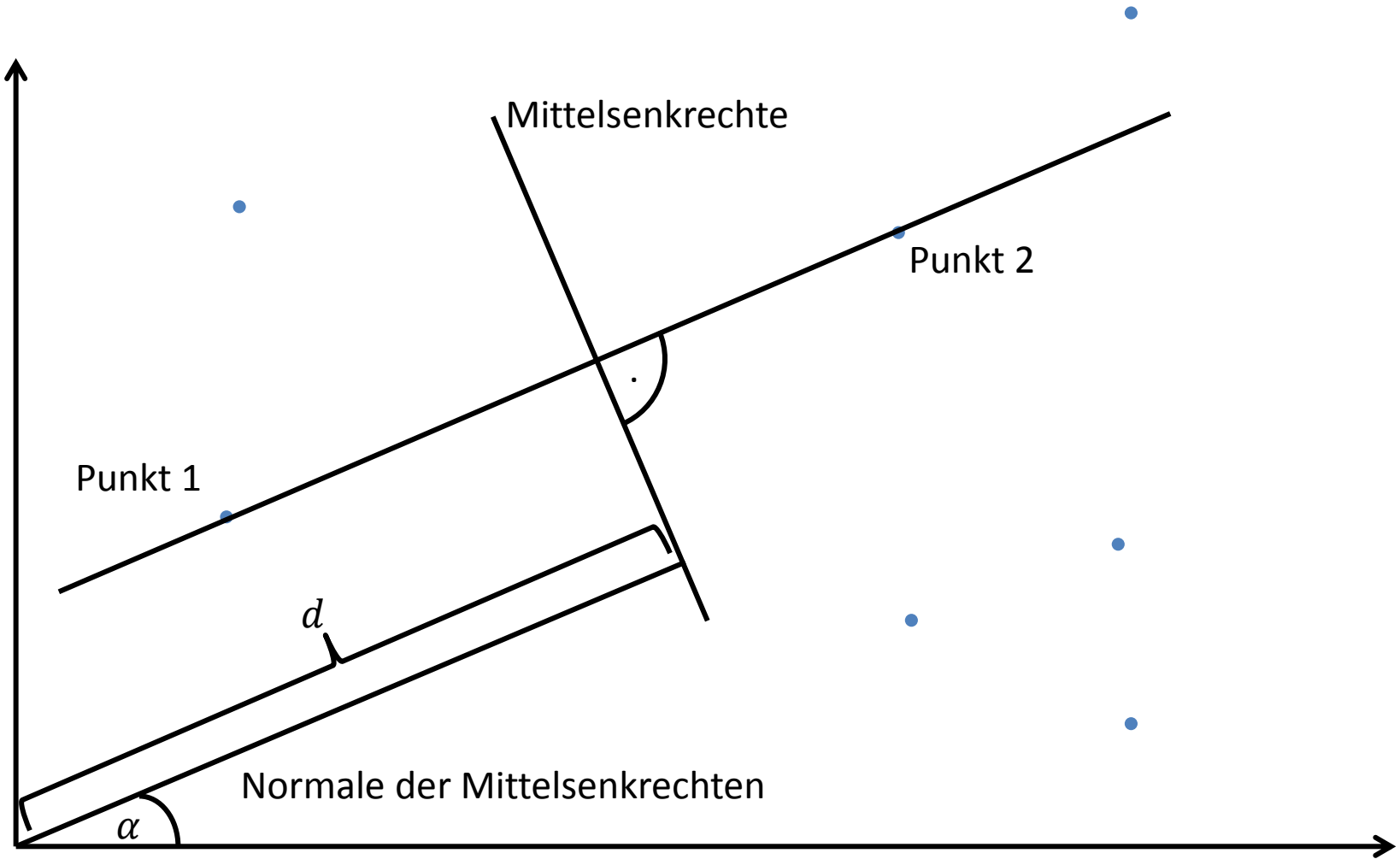


Linien \rightarrow Symmetrie

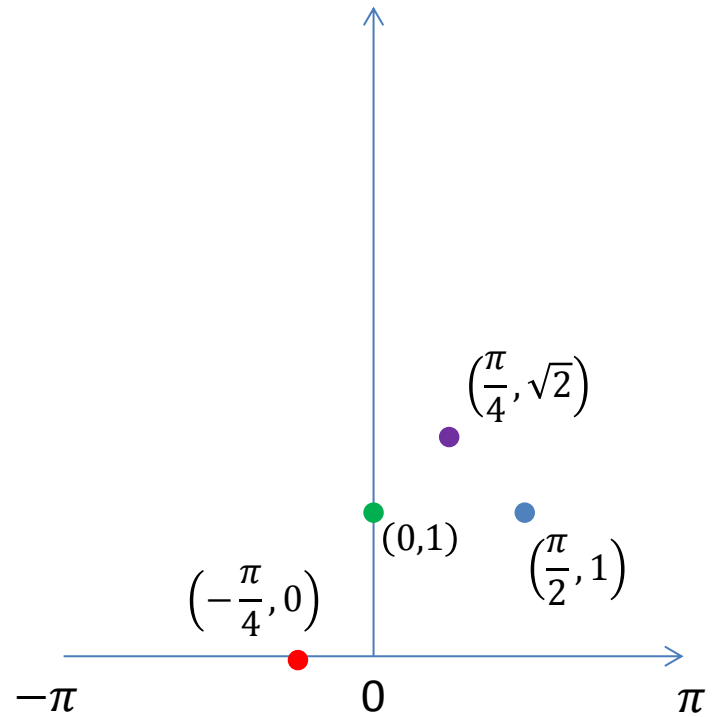
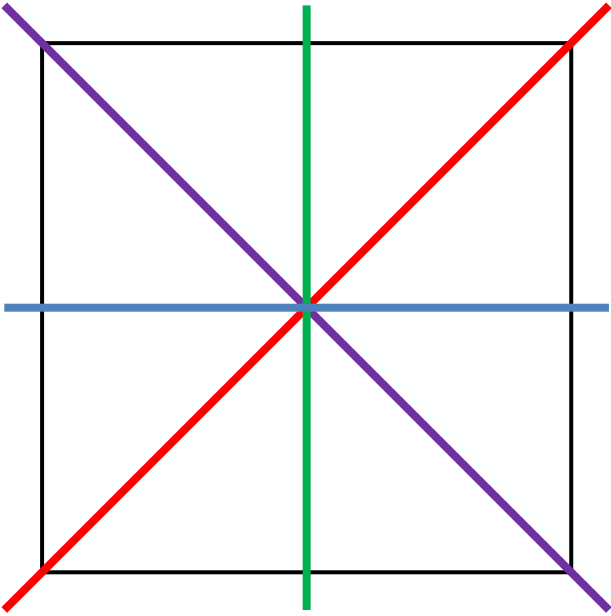
- \rightarrow Diedergruppe beschreibt Symmetrieachsen
- \rightarrow Symmetrieachse ist eine Gerade
- \rightarrow Suche nach einer „anderen“ Geraden



Berechnung



Beispiel



Algorithmus

$n \triangleq \#Punkte$

- 1) **For** (p_1, p_2) **in** Punkte X Punkte
- 2) **Calc** Parameter für Parameterraum
 (hier: Polarkoordinaten der Mittelsenkrechten)
- 3) **Inc** Akkumulationsraum[diskretisierte Parameter]
- 4) **Calc** Häufungen im Akkumulationsraum suchen

$O(n^2)$

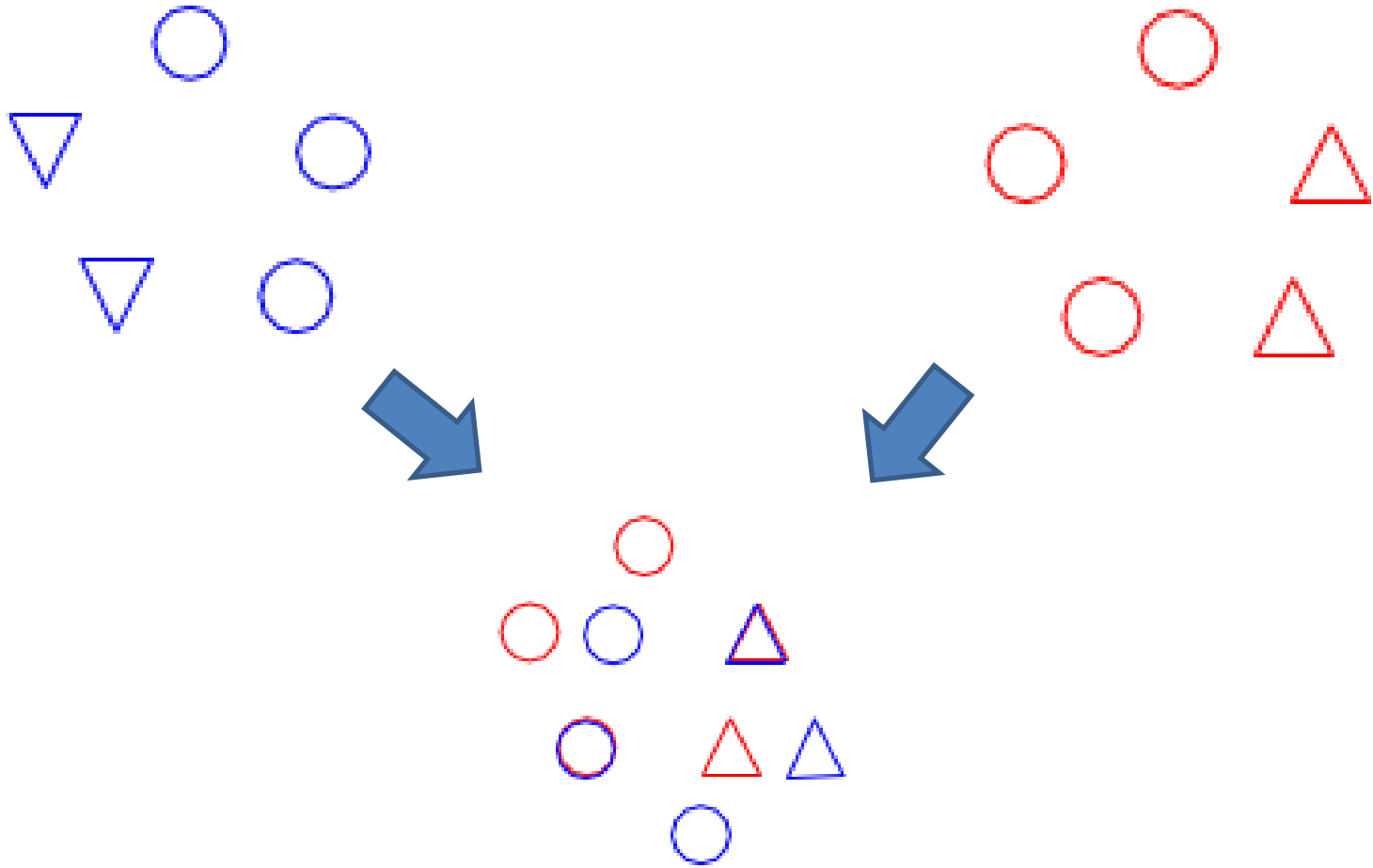
$O(1)$

$O(1)$

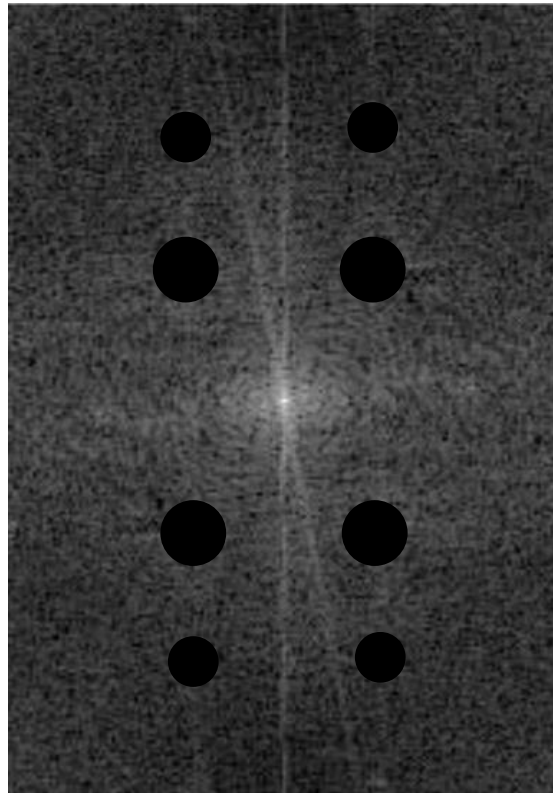
$O(n^2)$

→ Laufzeit: $O(n^2)$

Kongruenz



Symmetrierkennung und die Fourier Transformation

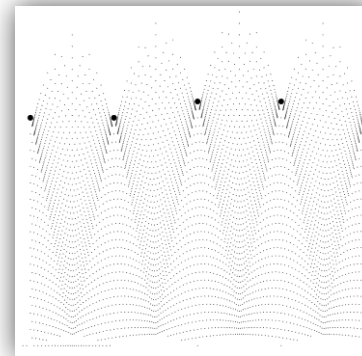
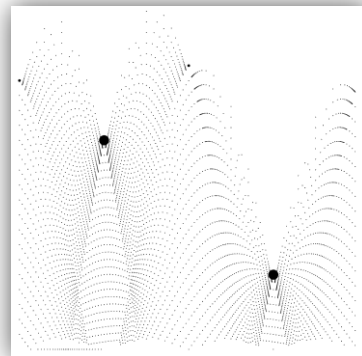
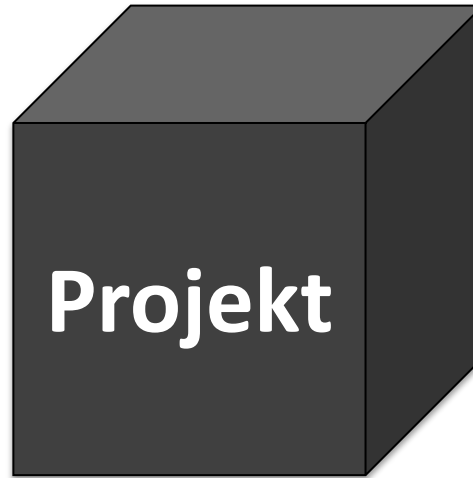
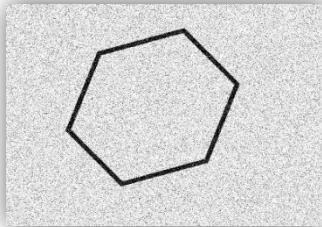
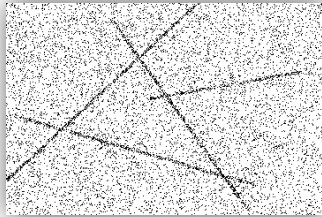




MASTER-PROJEKT

Master Projekt

Graustufenbild



Symmetriegruppe

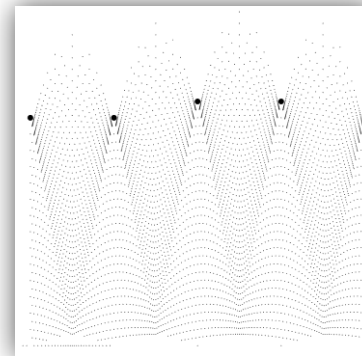
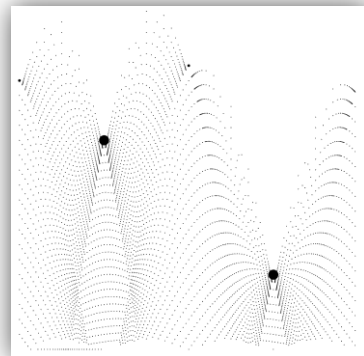
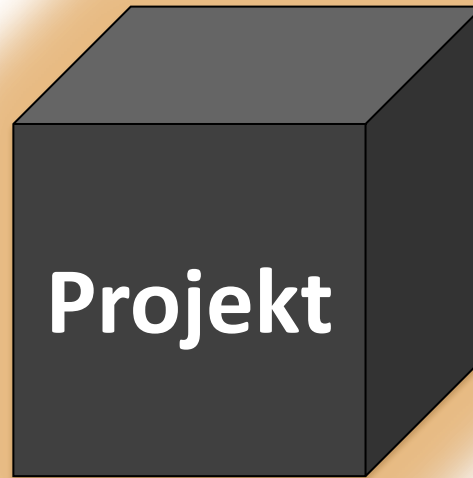
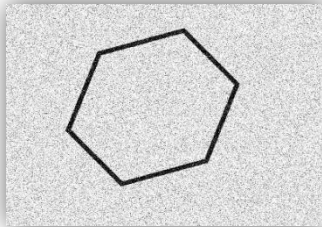
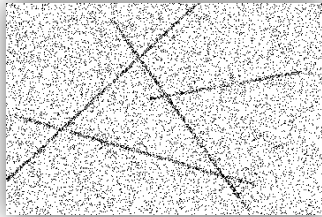
Keine Symmetrie

Keine Symmetrie

D_6 & C_6

Master Projekt

Graustufenbild



Symmetriegruppe

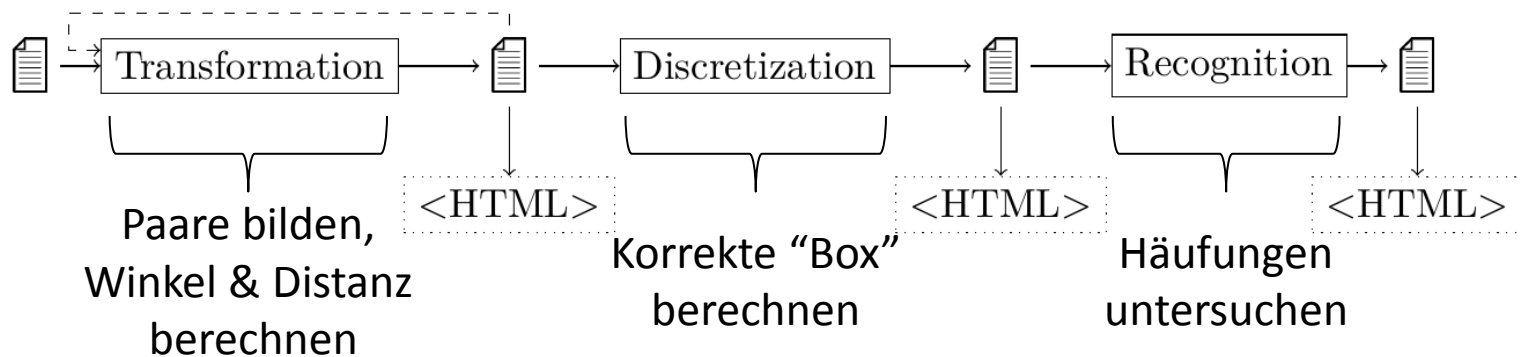


Keine Symmetrie

Keine Symmetrie

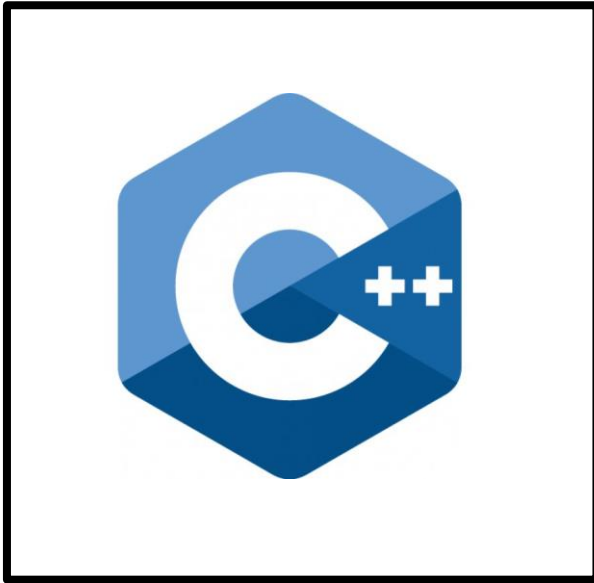
D_6 & C_6

Die Pipeline



- 1) **For** p_1, p_2 **in** Punkte X Punkte
- 2) $g_{\text{param}} = mx + b - y$ aus (p_1, p_2) konstruieren
- 3) $g_{\text{polar}} = \text{polarcoord}(g_{\text{param}})$
- 4) Parameterraum += g_{polar}
- 5) **For** p_1', p_2' **in** Parameterraum
- 6) DiskreterRaum += $\text{interpolated}(p_1', p_2')$
- 7) Suchen von Häufungen(DiskreterRaum)

Software Architektur



Backend



Protocol Buffer



Frontend

Warum C++ und HTML?

Harte Vorgaben von der FU Berlin

- Kein Haskell ☹️
- C++ oder Java

Weichere Vorgaben von uns

- Viele Experimente mit der Oberfläche
 - Strikte Trennung zwischen Berechnung und Visualisierung
 - Qt & GTK (C++) bzw. Swing, AWT & JavaFX (Java) sind eher “klassische” UI Bibliotheken
- Plattformübergreifend (sogar auf mobilen Geräten)

Warum Protocol Buffer?

	JSON	Protobuf
Typisierung <ul style="list-style-type: none">• Zahlen bestimmter Größen• Strukturen, Listen• Automatische Validierung	- (keine Validierung)	++
Einfach (Javascript) <ul style="list-style-type: none">• Externe Abhängigkeiten?	++ (nativ unterstützt)	- (Bibliotheken, aber keine nativen Integer)
Einfach (C++) <ul style="list-style-type: none">• Externe Abhängigkeiten?	- (Bibliotheken, eigene Datenstrukturen)	+ (Offizielle Code Generatoren)
Größe der Nachrichten	--	++

Warum Protocol Buffer?

```
message ExactSample {  
  required double angle = 1;  
  required double distance = 2;  
  required Intensity i1 = 3;  
  required Intensity i2 = 4;  
}
```



```
message Intensity {  
  repeated int32 channels = 1;  
}
```

Datentyp	Protocol Buffer	JSON
double	8 Byte	16 ASCII Zeichen
int32 (0 – 255)	1 – 2 Byte	1 – 3 ASCII Zeichen

Protocol Buffer $\frac{1}{5}$ bis $\frac{1}{2}$ der Größe

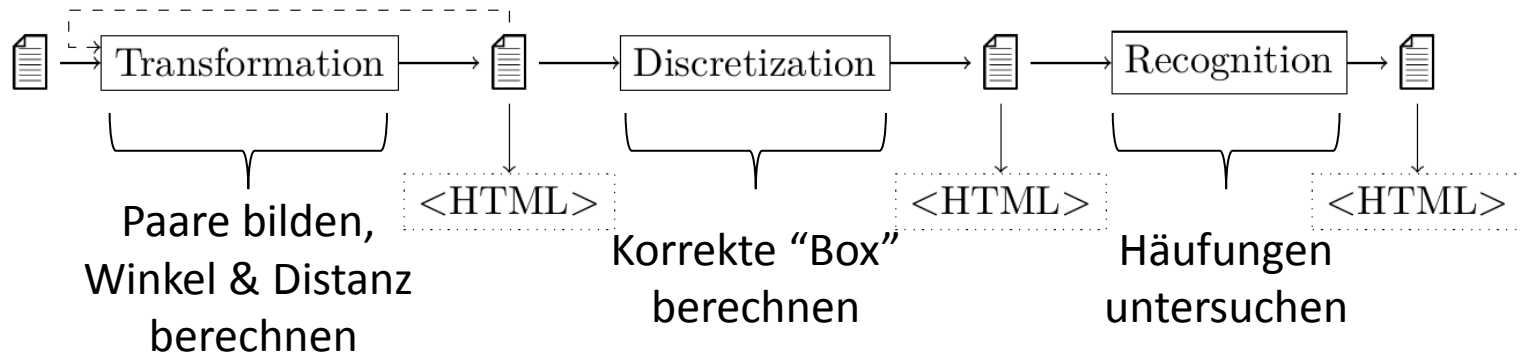
+ 33% für Base64 Encoding



$$x * \frac{1}{5} * \frac{13}{10} \sim 0.26x$$

$$x * \frac{1}{2} * \frac{13}{10} \sim 0.65x$$

Parallelisierung mit der Pipeline



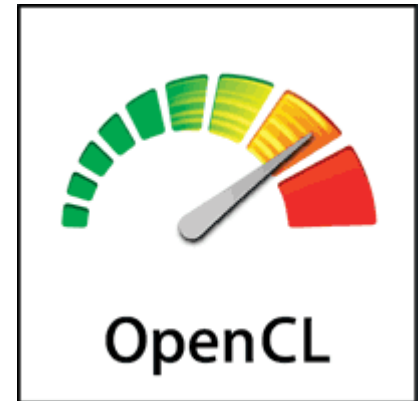
Gewinne durch die Pipeline eher marginal 😞

- 3 Stufen, also auch maximal 3 Threads
- **Stufen nicht gleichermaßen aufwändig**
- Overhead durch Synchronisierung bei naiver Implementierung recht groß
- Trotzdem softwaretechnisch eine gute Struktur

Parallelisierung mit Partitionierung

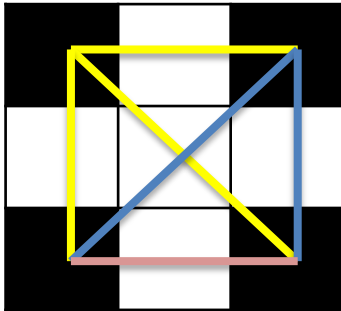
Annahmen

- Erste Pipelinestufe dauert am Längsten
- Statische Partitionierung ist unkritisch



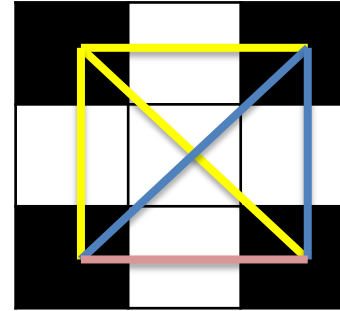
```
1) partitions = divide(Punkte X Punkte, #Cores)
2) For thread in Threads
3)   thread.compute
4)     For p1, p2 in partitions[thread.num]
5)       gparam = mx + b - y aus (p1, p2) konstruieren
6)       gpolar = polarcoord(gparam)
7)       thread.Parameterraum += gpolar
8)     Parameterraum += thread.Parameterraum
9) Threads.join()
```


Auswahl von Paaren



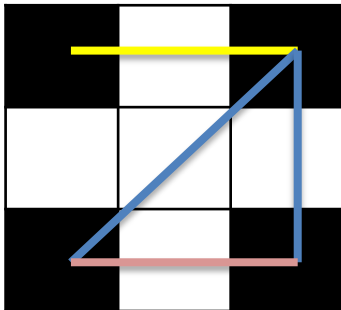
Alle Paare

- 3 Gelb
- 2 Blau
- 1 Rot



Jedes 2. Paar

- 2 Gelb
- 1 Blau
- 0 Rot

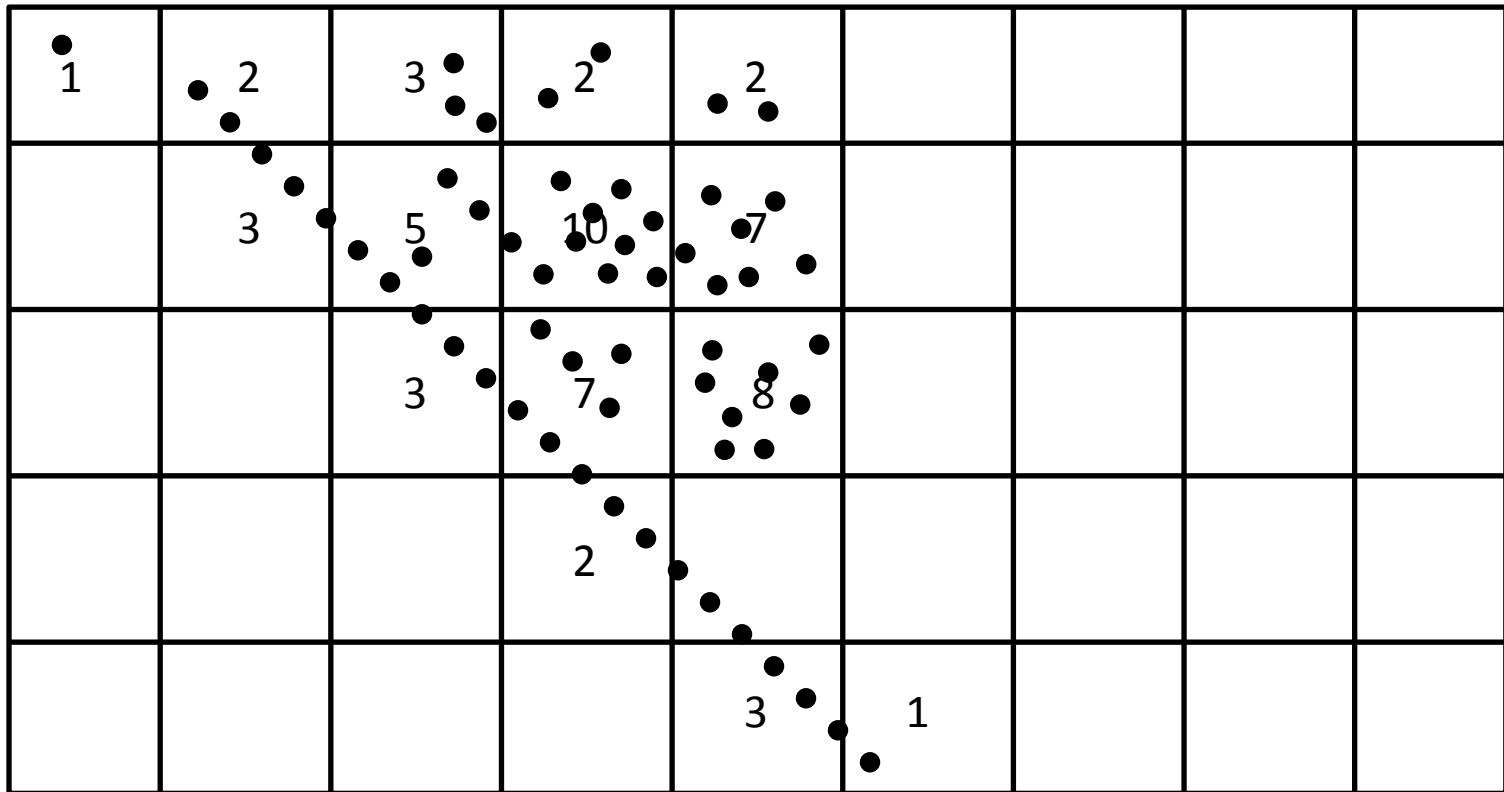


Zufällige Paare

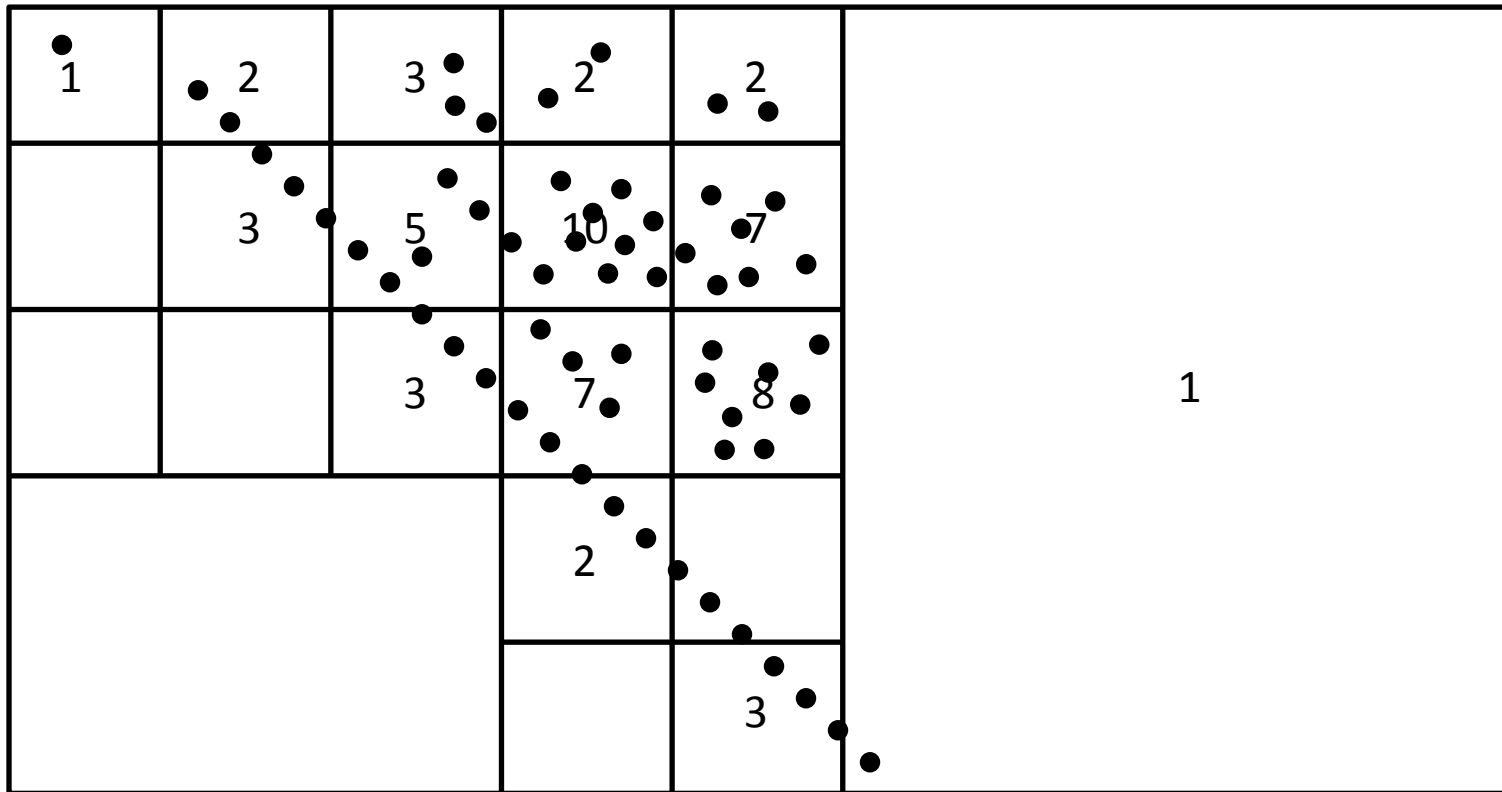
- 1 Gelb
- 2 Blau
- 1 Rot

Leicht zu implementieren, schwierig zu untersuchen

Diskretisieren der Werte



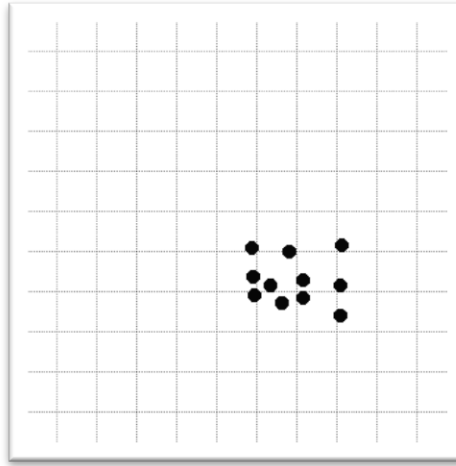
Diskretisieren der Werte



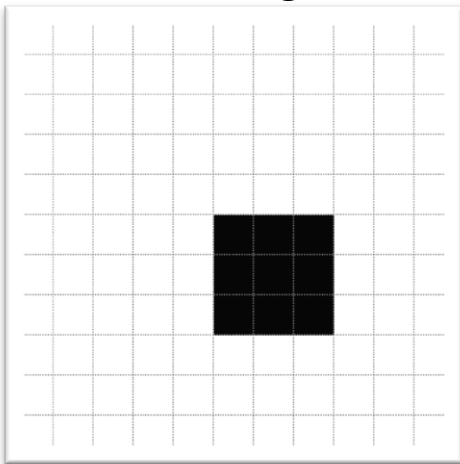
Diskretisieren der Werte

1	2	3	2	2	1
	3	5	1 3	7	
		3	3 3	8	
			2		
				3	

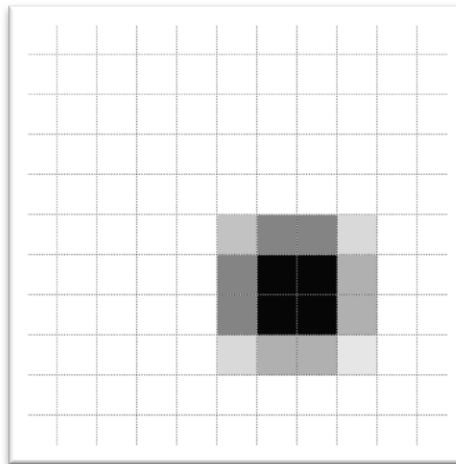
Diskretisieren der Werte



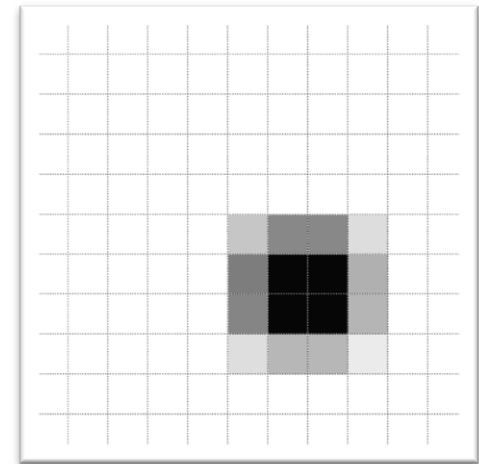
Nearest Neighbour



Bilinear



Bikubisch



Programme der FU

Shapes Configuration Window

Transformati...
● Translation
○ Rigid Motion
○ Similarity

Autoconfig?

Vote Generation I
 Get votes from polygon interior
Number of votes per step: 100,000
Maximum number of votes: 3,000,000

Vote Generation II
Rigid Motion: 3+1 Points
Similarity: 4 Points
 Allow only reasonable votes

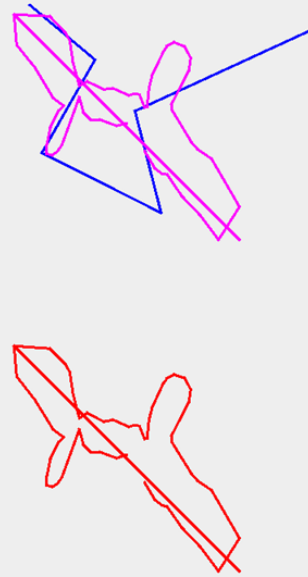
Clustering Algorithm
Clustering Algorithm: Discretized
Data Format for Votes: lambda, alpha, vx, vy
 Use Scaling Factor as Weight

Clustering Resolution
Translation (pixels): 20
Rotation (degree²): 5
Scaling Factor: 0.1

Control

Start Match

Source
Target



Source Shape
Border Length: 1494.871
Area: - not a multipolygon -
Bounding Box: (22.5; 548.6) -- (294.3; 820.4)
Self Similarity (using the specified parameters): [calc...](#)

Target Shape
Border Length: 751.801
Area: - not a multipolygon -
Bounding Box: (41; 136.5) -- (378; 387.5)

Transformed Shape
Symmetric Difference to Target: - not a multipolygon -
Area of Union with Target: - not a multipolygon -
Area of Intersection with Target: - not a multipolygon -
Best found transformation: rotation = 0.0° | scale = 1.0 | translation = (0.0; -400.0)
Number of votes in the best cluster: 20010
Votes in this cluster / total votes: 0.00667
Ratio: Percentage / Self Similarity: - Self Similarity is unknown -

Display
 Source
 Target
 Best Transformation
 2nd best transformation
 Apply inverted transform

2,999,999 maps
Resolution: translation=20.0

User-Interface

Responsive Design

SymmRecog Transformation ▾ Discretization ▾ Recognition ▾

No image selected! Please upload an image from the gallery.

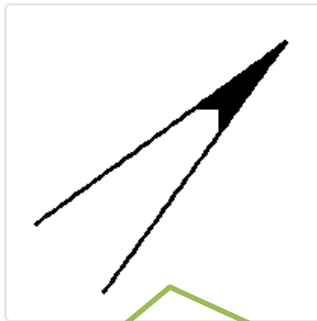
Discretization step size
Interpolation

Pair selection method

All pairs ▾

Dynamisch einblendbare Optionen

Gallery



Eine Aktion für Berechnung

Original graphic

Please select an image from the gallery!

Transformation space

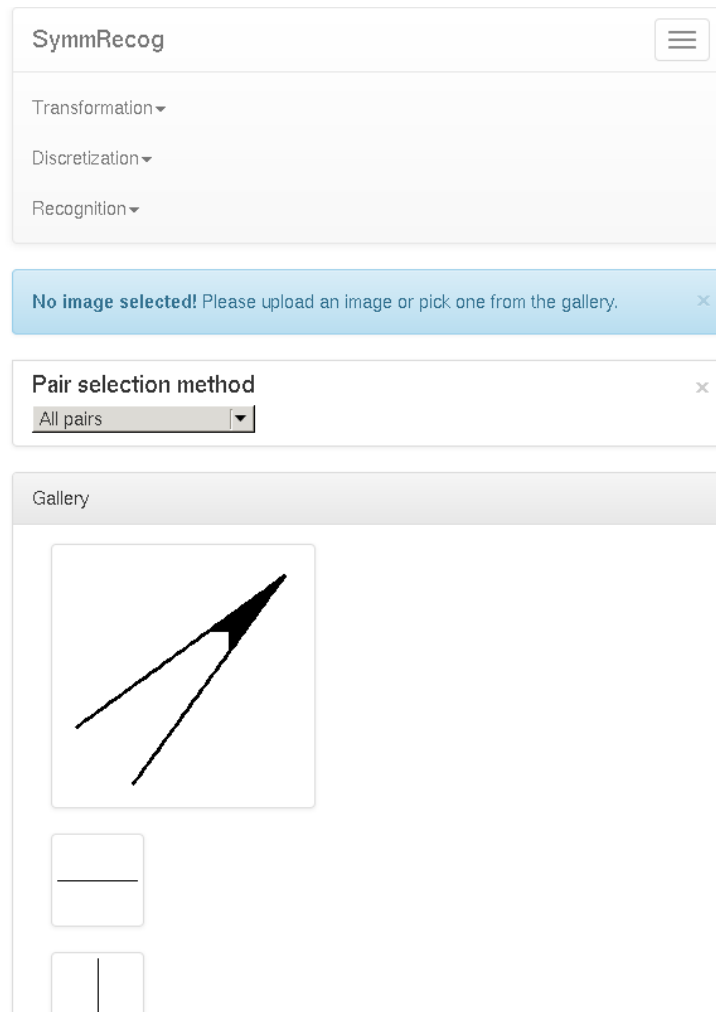
Please select an image from the gallery!

Platz für Inhalte

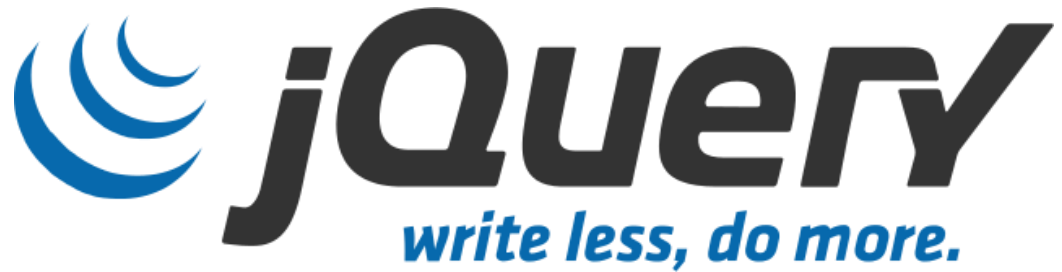
Log output

Image at images/big-mess.png (800x800) with 24 bits per pixel, maximum intensity is 1

User - Interface (mobil)



User - Interface



Javascript



Visualisierung

SymmRecog

Transformation ▾

Discretization ▾

Recognition ▾

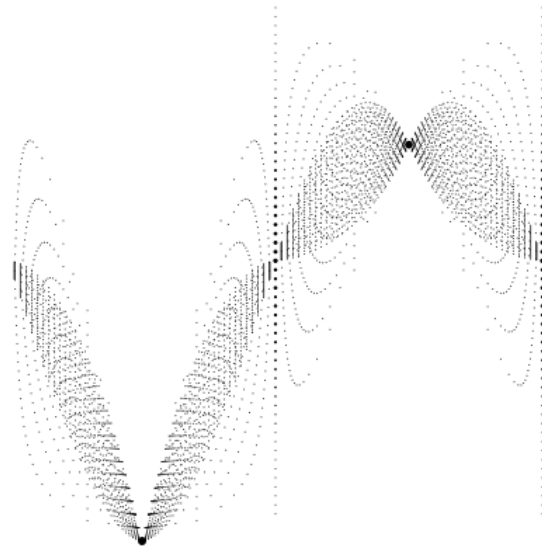
Gallery



Original graphic




Transformation space



Potenziell
große
Datenmenge

Log output

Canvas ↔ SVG

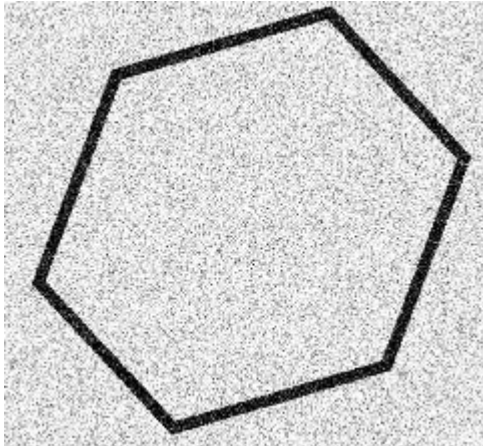
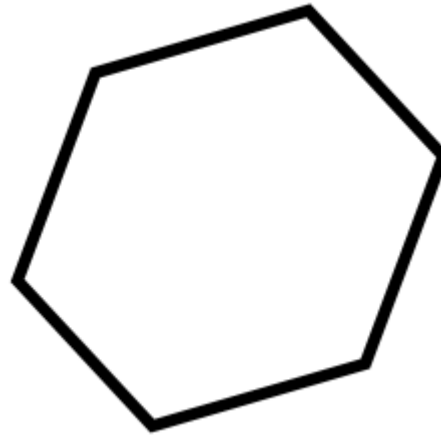
	HTML5 Canvas	Scalable Vector Graphic
 Zeichnen-Beschreibung	Canvas Primitive (Funktionen)	XML Beschreibung
Einbettung	HTML Element	SVG Namespace
Bildformat	Rastergrafik	Vektorgrafik
Interaktion	Browserevents des Canvas	Element-individuelle Events
Manipulation	Neuzeichnen	DOM-Manipulation
Performance	Gut	Langsamer als Canvas

A promotional photograph of the actors Benedict Cumberbatch and Martin Freeman as Sherlock Holmes and Dr. Watson. They are standing in front of a dark green door with the address '221B' in gold lettering. Sherlock is on the right, wearing a dark blue tweed coat and a matching scarf. Dr. Watson is on the left, wearing a dark blue raincoat over a checkered shirt. The scene is set in a classic London street environment.

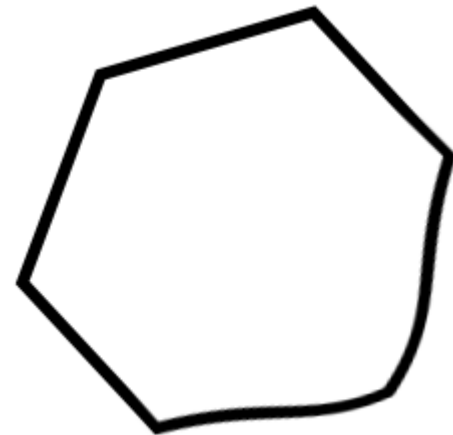
221B

AUSBLICK UND FAZIT

Ausblick: Störungen



Rauschen



Verformen

Ausblick: Zu untersuchende Größen

- Störungen
- Generierung der Punktepaare
 - Einfluss der verschiedenen Methoden der Paarbildung?
 - Anzahl nötiger Paare für gute Erkennung?
- Diskretisierung
 - Einfluss der variabel großen Zellen?
 - Einfluss der verschiedenen Interpolationsmethoden?

Ende əbnɛ
ɛnqɛ əpnɛ