

# **5. Objekterkennung in Bildern und Videos**

## **Videoanalyse**

Stephan Kopf

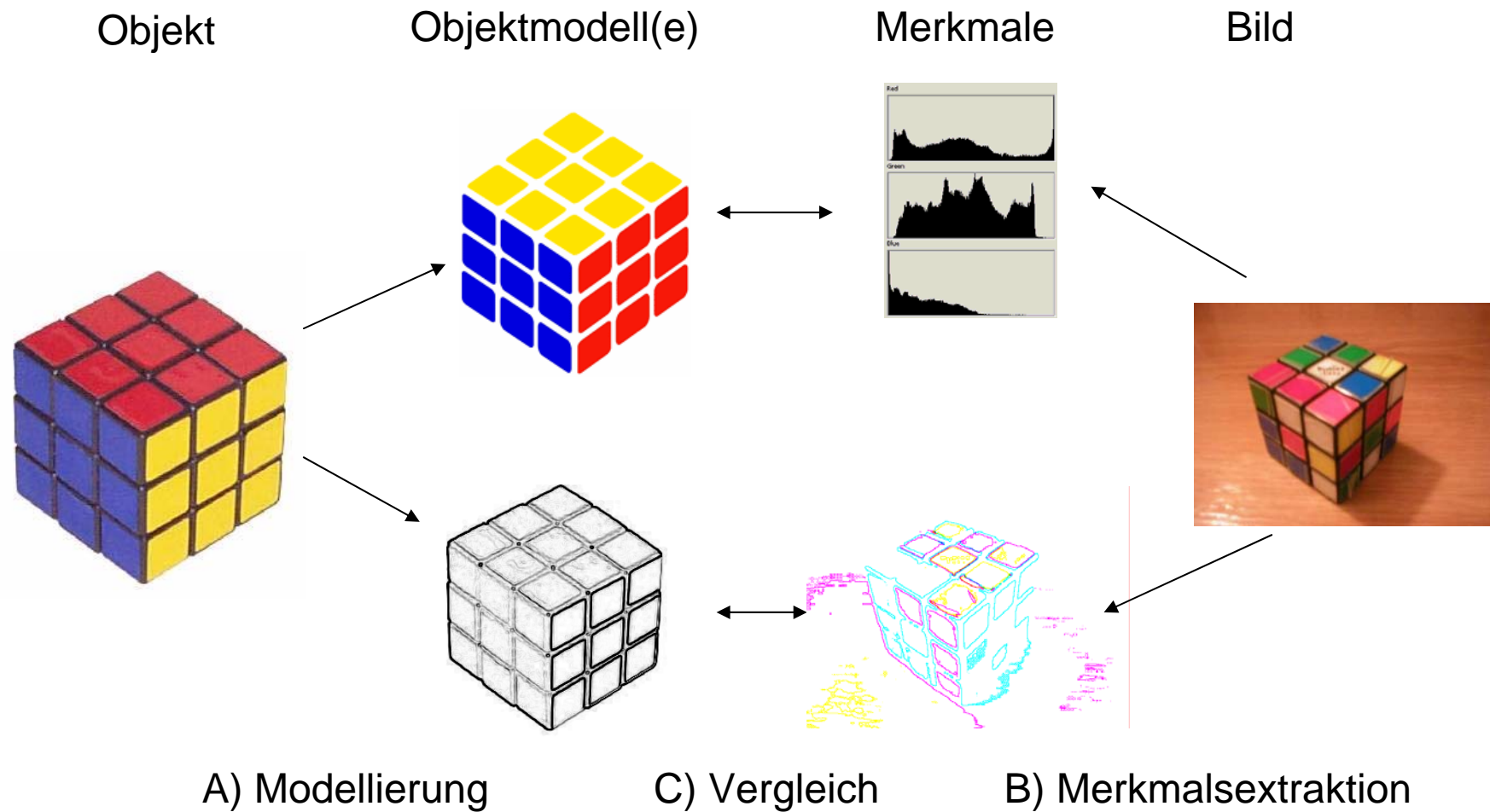
# Übersicht

- Motivation
- Anforderungen an Objekterkennungsalgorithmen
- Objekterkennung mittels
  - Konturen
  - Helligkeiten
  - Farben
  - Formen
- Objekterkennung in Videos
  - Video-OCR
  - Analyse der Bewegungen von Personen

# Warum Objekterkennung ?

- **Schrifterkennung**
  - OCR-Software zur Texterkennung
  - Automatische Postleitzahlenerkennung
- **Video-Überwachung:**
  - Identifikation verdächtiger Personen (Diebstahl, Bedrohung)
  - Erkennung verdächtiger Gegenstände (unbeaufsichtigte Koffer auf Flughäfen oder Bahnhöfen)
- **Inhaltsbasierte Bildsuche**
  - Internetsuche (z.B. [Google Bilder](#))
  - Bild- und Videoarchive (z.B. [gettyimages](#))

# Modell-basierte Objekterkennung



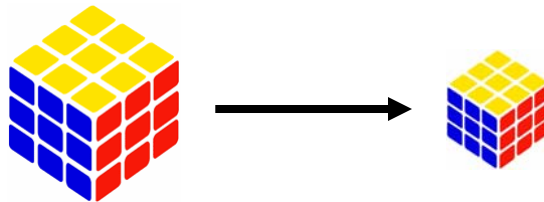
# Was ist ein gutes Objektmodell ?

- ***sensitivity***
  - Das Modell muss alle relevanten Merkmale darstellen und zwischen diesen unterscheiden können.
- ***uniqueness***
  - Das Modell soll in der Lage sein, Objekte möglichst eindeutig zu beschreiben; gleichartige Objekte sollen die gleiche Beschreibung haben.
- ***stability***
  - Kleine Veränderungen am Objekt sollen kleine Auswirkungen auf das Modell haben.
- ***efficiency***
  - Es muss möglich sein, Merkmale effizient aus Daten zu berechnen und Merkmale effizient zu vergleichen.

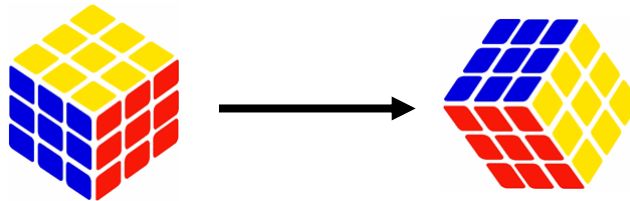
# Anforderungen (I)

**Das Modell sollte invariant sein gegenüber:**

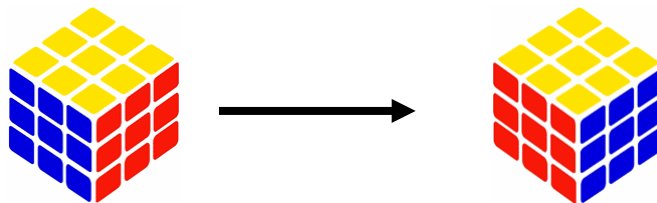
- Skalierungen



- Rotationen



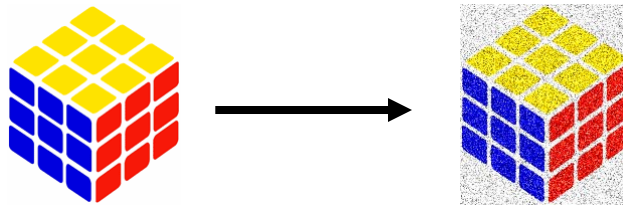
- Spiegelungen



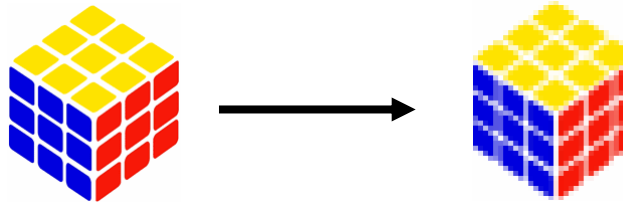
# Anforderungen (II)

**Das Modell sollte invariant sein gegenüber:**

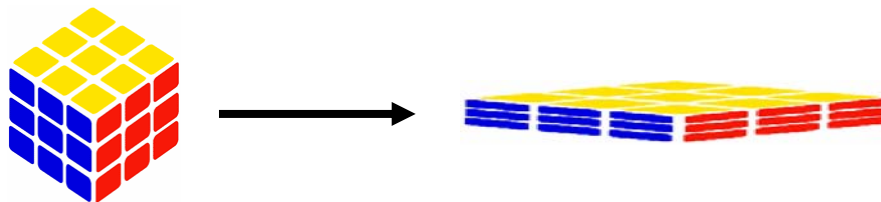
- Rauschen



- Unschärfe



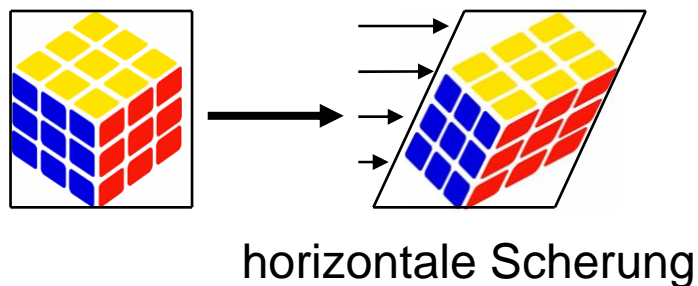
- Stauchungen



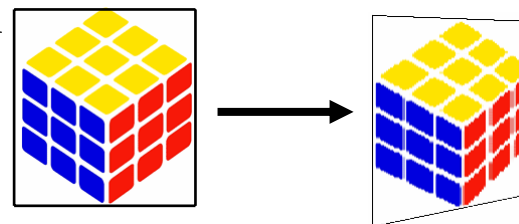
# Anforderungen (III)

**Das Modell sollte invariant sein gegenüber:**

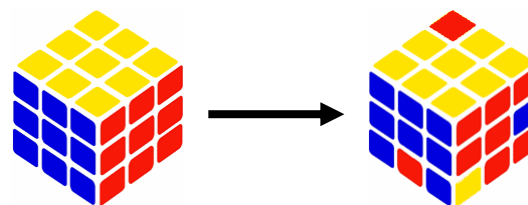
- Scherungen



- Perspektivischen Verzerrungen

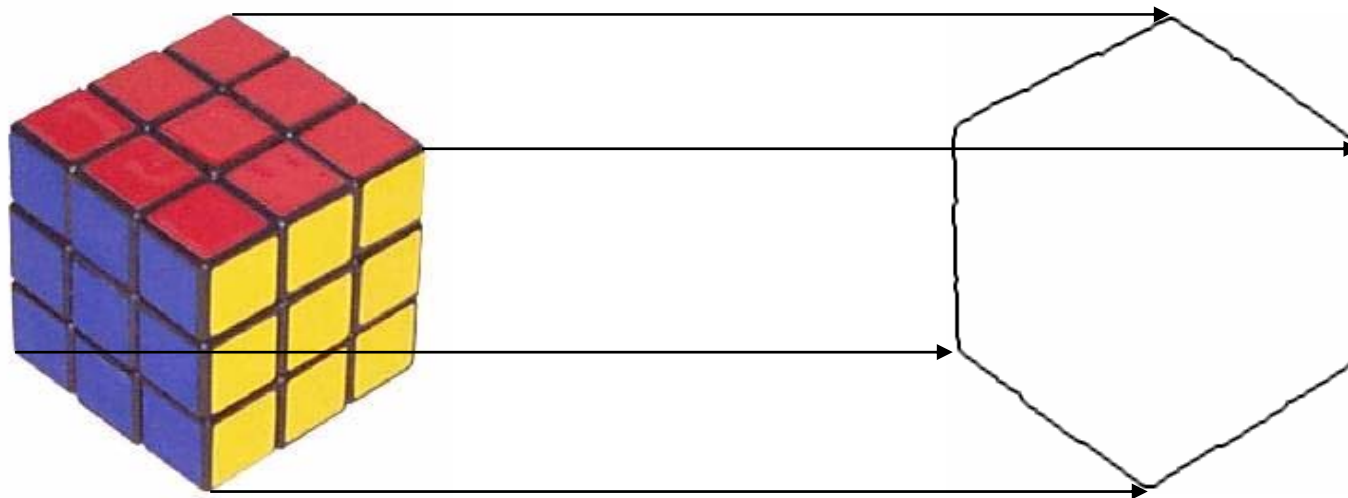


- Objektverformungen

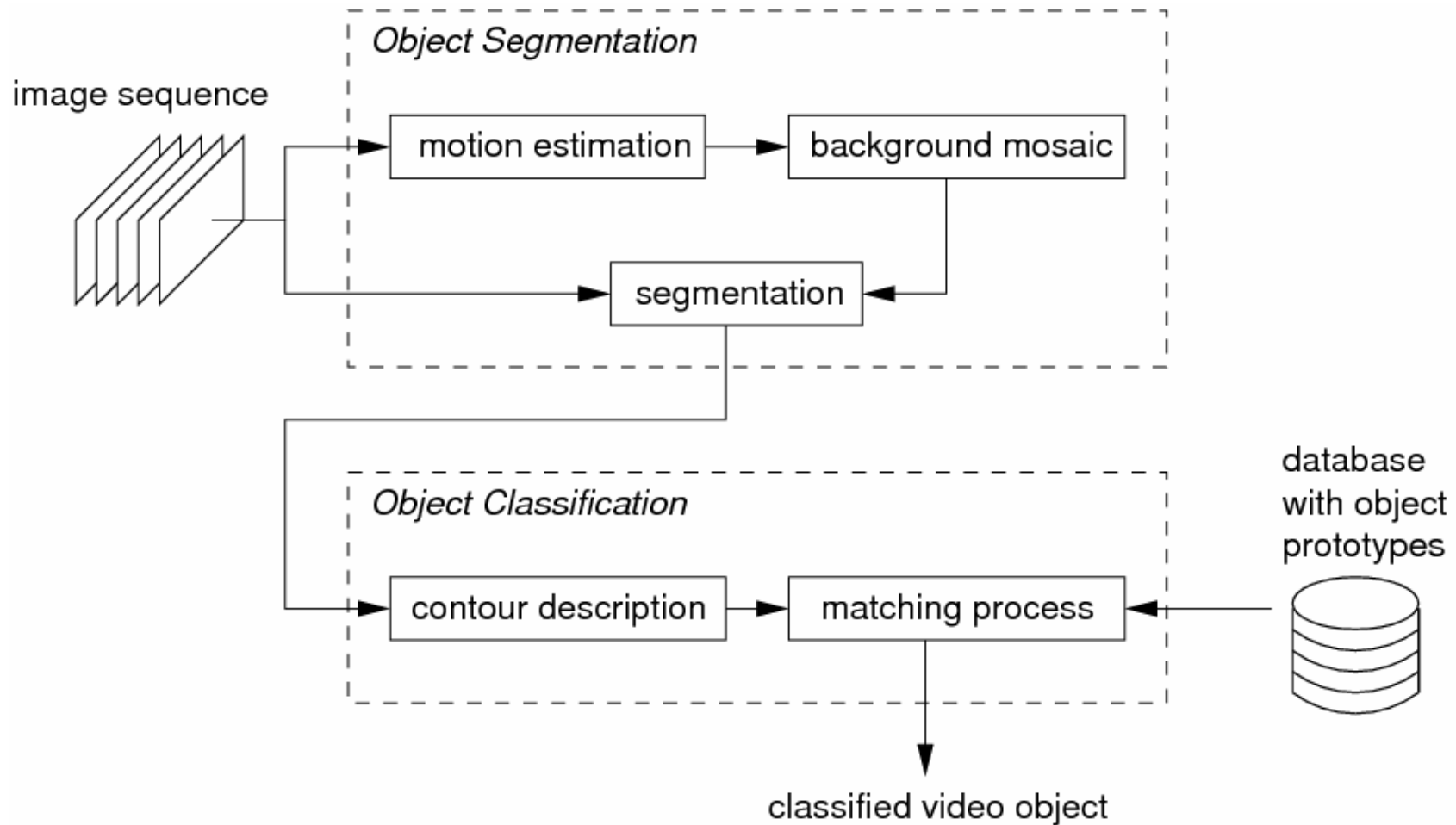




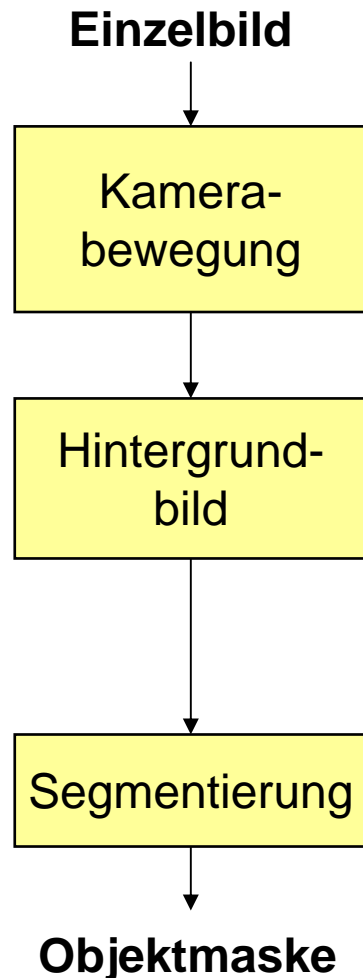
# Objekterkennung durch Vergleich von Konturen (I)



# Objekterkennung durch Vergleich von Konturen (II)



# Objektsegmentierung



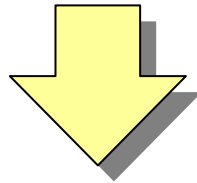
- Annahme: Die Hälfte aller Pixel eines Bildes sind Hintergrundpixel.
- Berechne Kamerabewegung zwischen zwei benachbarten Bildern des Videos.
- Richte alle Bilder einer Kameraeinstellung entsprechend des Kameramodells aus, so dass der Bildhintergrund deckungsgleich ist.
- Verwende einen Medianfilter, um aus den transformierten Bildern ein Hintergrundbild zu erzeugen.
- Vergleiche das Hintergrundbild mit den transformierten Einzelbildern.

# Ablauf der Objekterkennung

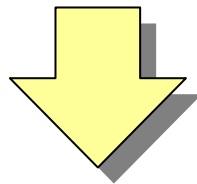
1. Abtastung der Kontur mit einer vordefinierten Anzahl an Konturpunkten.
2. Identifiziere Merkmalspunkte zur Beschreibung der Kontur (verwende das *Curvature-Scale-Space*-Verfahren).
3. Vergleiche die Merkmale mit Merkmalen von bekannten Objekten (*Shape Matching*)

# Parametrisierung einer Kontur

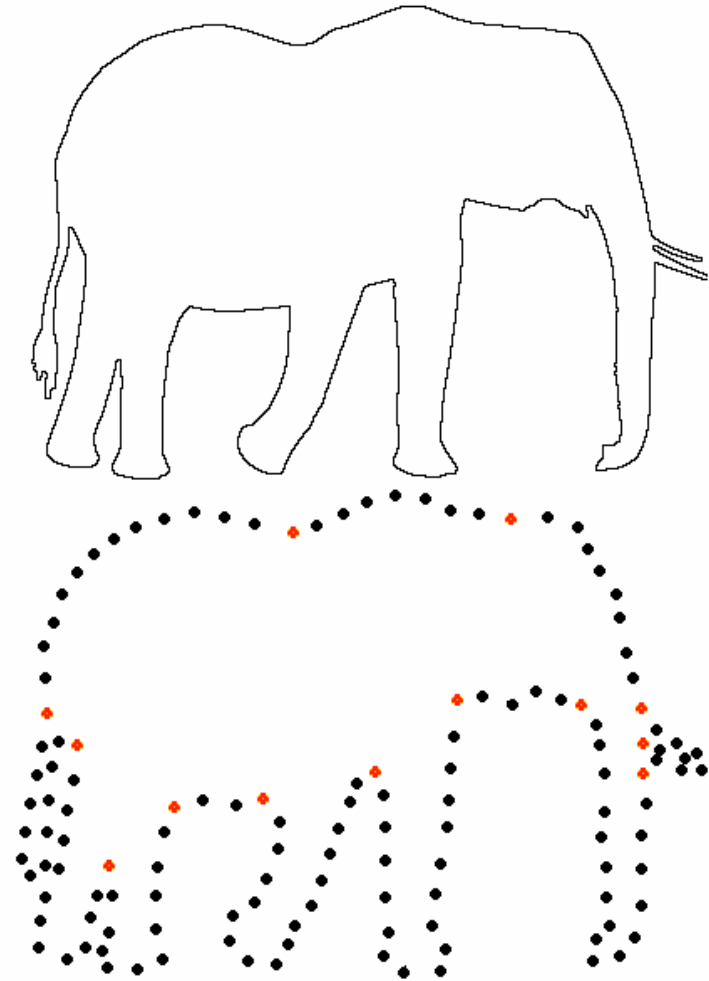
Segmentierung



Parametrisierung  
der Kontur



Berechnung der  
Skalenraumabbildung

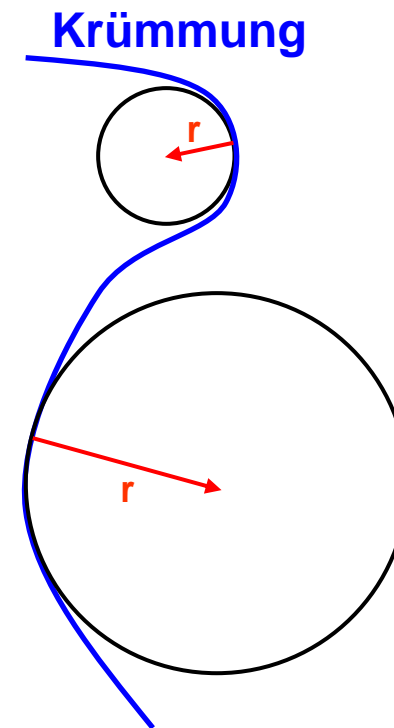


# Definition der Krümmung (I)

- Die **Stärke der Krümmung** in einem Punkt entspricht dem Kehrwert des Radius eines sich anschließenden Kreises (der Kreis berührt die Kurve):

$$K = \frac{1}{r}$$

- Die Krümmung ist ein Vektor, der in die Richtung der Kreismittelpunktes zeigt.
- Ein kleiner Kreis repräsentiert eine hohe Krümmung; eine Gerade hat eine Krümmung von Null.



# Definition der Krümmung (II)

- Gegeben ist eine planare Kurve  $u(t)$  innerhalb eines 2D Raums.  $u(t)$  wird durch die Bogenlänge  $t$  parametrisiert.
- Die Kurve  $u(t)$  wird durch die beiden Funktionen  $x(t)$  und  $y(t)$  definiert:

$$u(t) = (x(t), y(t)).$$

- Die Krümmung  $K$  für die Kurve  $u(t)$  ist definiert als:

$$K = \frac{\dot{x} \cdot \ddot{y} - \dot{y} \cdot \ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{3/2}}$$

$\dot{x}$  und  $\dot{y}$  definieren die erste Ableitung (Gradient),

$\ddot{x}$  und  $\ddot{y}$  definieren die zweite Ableitung (Änderung des Gradienten).

# Definition der Krümmung (III)

- Herleitung einer nicht-allgemeinen Definition der Krümmung: verwende explizite planare Kurven:  $y = f(x)$ .
- Krümmung im Punkt  $(x, f(x))$ :

$$K = \frac{f''(x)}{\left(1 + (f'(x))^2\right)^{3/2}}$$

- Typische Berechnung der Krümmung in vielen Bereichen:
  - berechne die Bewegung von Gasen oder Flüssigkeiten entlang von Oberflächen,
  - schätze das Verhalten, falls Bauteile gebogen werden (Statik von Brücken).



# Definition der Krümmung (IV)

## Beispiel

- Beispielfunktion  $u(t) = (x(t), y(t)) = (t, t^2)$ .

Explizite Definition der Funktion:  $y = f(x) = x^2$ .

- Krümmung basierend auf der **parametrisierten Kurve**:

Erste und zweite Ableitung:  $\dot{x} = 1$ ,  $\ddot{x} = 0$ ,  $\dot{y} = 2t$ ,  $\ddot{y} = 2$ .

$$K(t) = \frac{\dot{x} \cdot \ddot{y} - \dot{y} \cdot \ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{3/2}} = \frac{1 \cdot 2 - 2t \cdot 0}{(1^2 + (2t)^2)^{3/2}} = \frac{2}{(1 + 4t^2)^{3/2}}$$

- Krümmung basierend auf der **expliziten Definition**:

$$f'(x) = 2x, \quad f''(x) = 2 \quad K(x) = \frac{f''(x)}{(1 + (f'(x))^2)^{3/2}} = \frac{2}{(1 + 4x^2)^{3/2}}$$

# Definition der Krümmung (V)

- Approximation der Ableitung für diskrete Werte (parametrisierte Kurve):

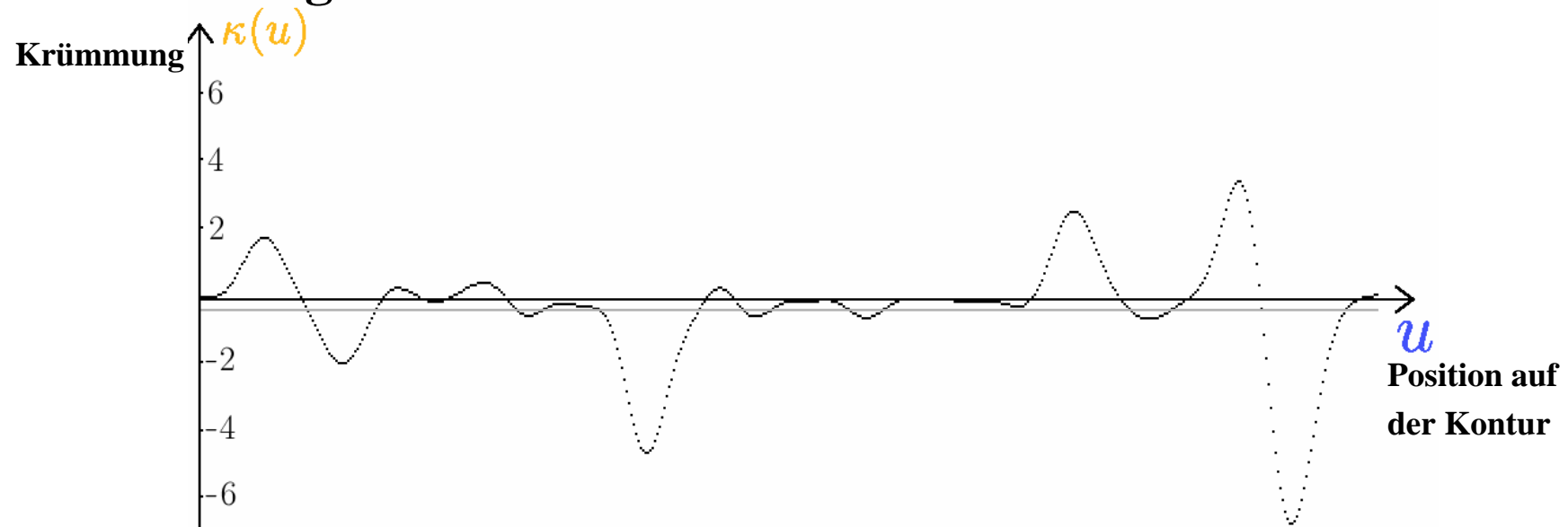
$$\dot{x}(t) = \frac{x(t+1) - x(t-1)}{2 \cdot h_x}$$

$$\dot{y}(t) = \frac{y(t+1) - y(t-1)}{2 \cdot h_y}$$

- Der Parameter  $t$  ist für ganze Zahlen definiert ( $t \in \mathbb{N}$ ).
- $h_x$  und  $h_y$  normalisieren die Ableitungen abhängig von der Entfernung der Abtastpunkte.

# Konturvergleich mit Hilfe von Krümmungen

## Krümmungsfunktion der Kontur

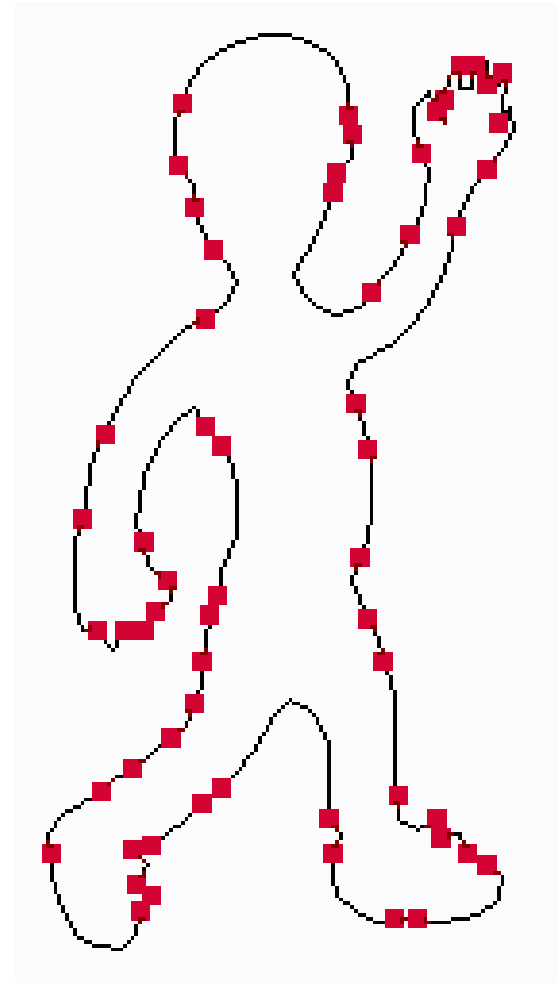


**Problem:** Zwei Krümmungsfunktionen können nur sehr schwer miteinander verglichen werden.

→ Identifiziere signifikante Punkte der Krümmungsfunktion (das Curvature-Scale-Space-Verfahren liefert eine gute Beschreibung)

# Curvature-Scale-Space-Verfahren (I)

- Analysiere äußere Kontur eines Objektes.
- Glätte Kontur mit einem Gaußfilter.
- Die **Wendepunkte der Kontur** werden als Merkmalspunkte gespeichert.



# Curvature-Scale-Space-Verfahren (II)

## Ablauf

1. Iterative Glättung der Kontur
2. Berechnung der Krümmung
3. Betrachte Nullstellen der Krümmungsfunktion  
(entspricht den Wendepunkten der Kontur) :

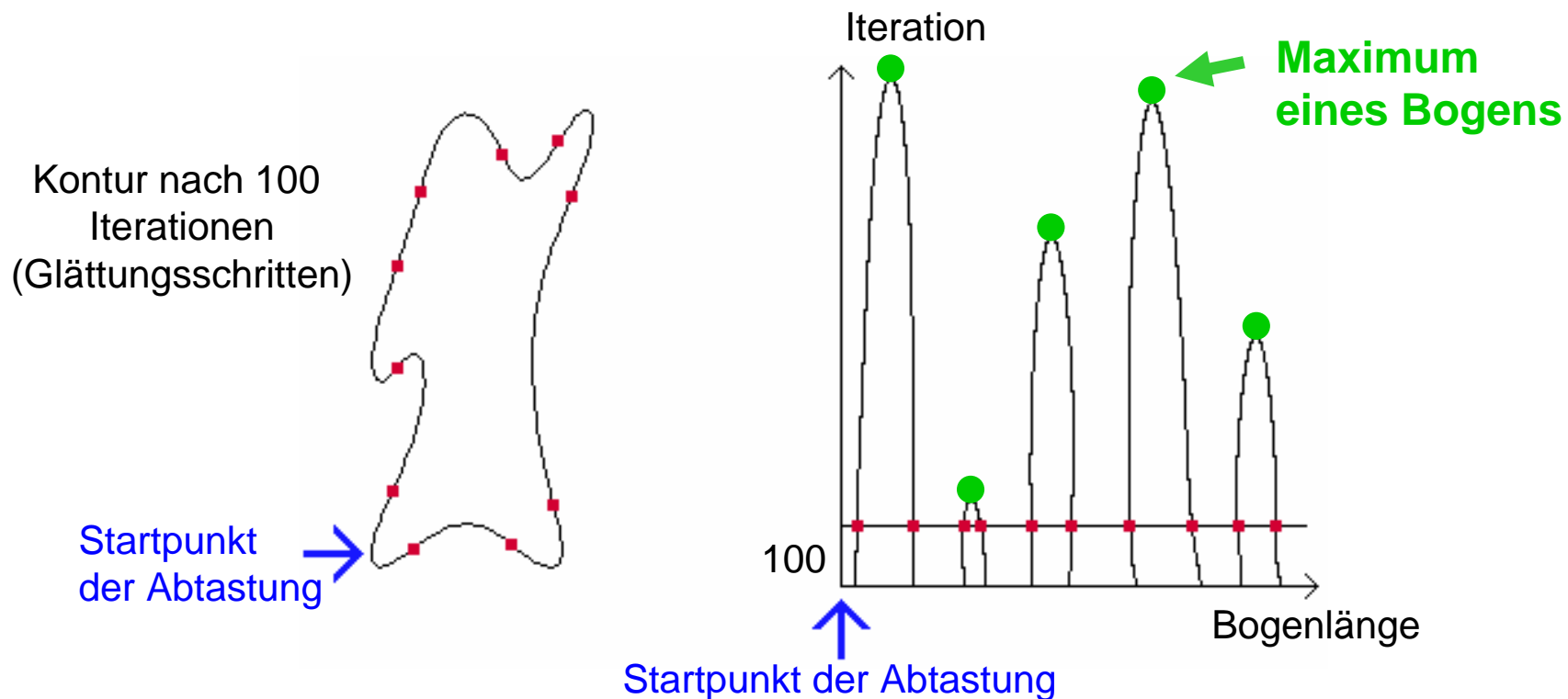
$$\kappa(u, n) = 0$$

**Definition: Skalenraumabbildung (Curvature-Scale-Space-Bild)**

$$I(u, n) = \{ (u, n) \mid \kappa(u, n) = 0 \}$$

# Curvature-Scale-Space-Verfahren (III)

- Ein **Curvature-Scale-Space-Bild** (CSS-Bild) stellt die Wendepunkte, die bei der Glättung entstehen, visuell dar.



Die **Bögen** werden als Merkmale zur Beschreibung des Objektes verwendet.

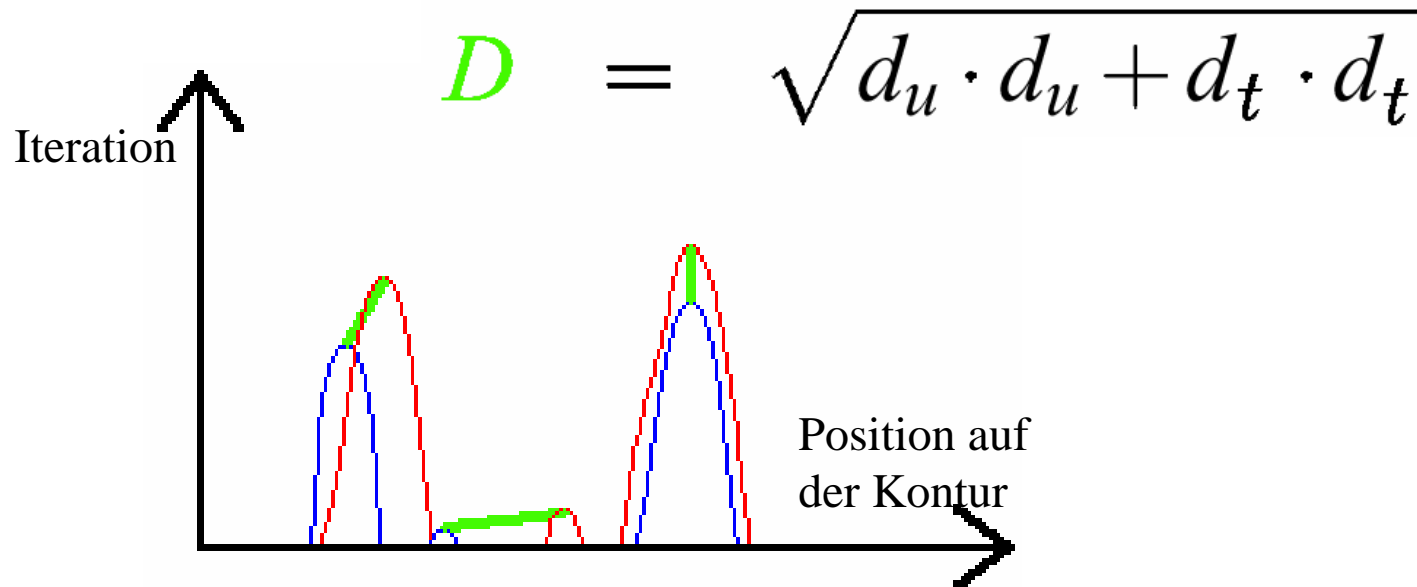
# Curvature-Scale-Space-Verfahren (IV)

## Merkmale von CSS-Bildern

- Die Bögen im CSS-Bild beschreiben **konkave Bereiche** einer Kontur.
- Die Bögen liefern die Merkmale zur Beschreibung einer Kontur.
- Jeder Bogen wird beschrieben durch
  - eine **Position**:  
relative Position im Vergleich zu den anderen Bögen,
  - eine **Höhe** (Anzahl der erforderlichen Glättungen):  
Stärke eines konkaven Segmentes.

# Vergleich zweier Konturen

1. Verschiebe ein CSS-Bild horizontal bis die höchsten Bögen übereinstimmen (Verfahren wird invariant gegenüber Rotationen)
2. Berechne die Euklidische Distanz zwischen zwei Bögen

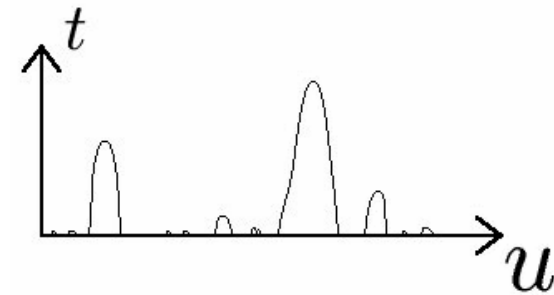
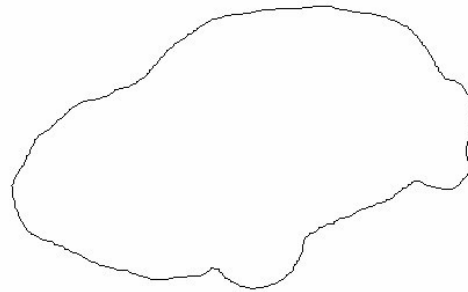


3. Summiere die Distanzen

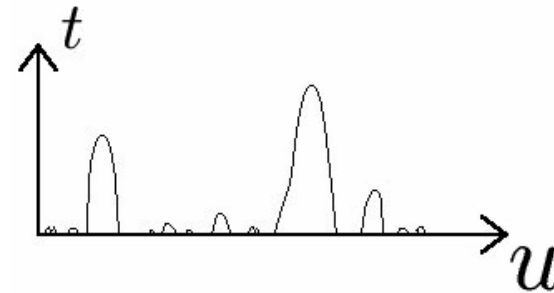
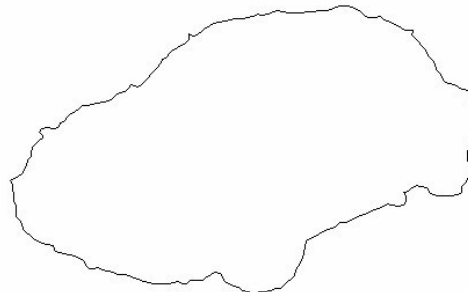


# Merkmale von CSS-Bildern (I)

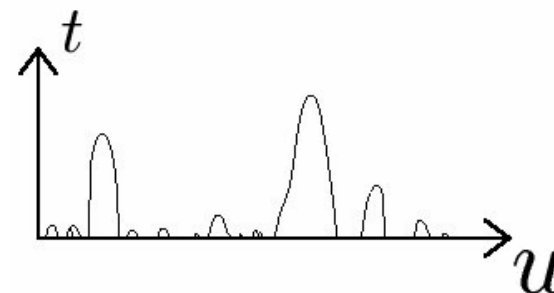
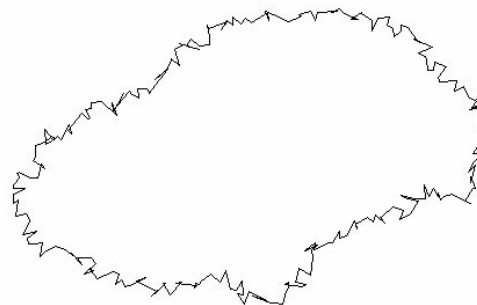
- Originalkontur



- Verrauschte Kontur



- Stark verrauschte Kontur



→ Die CSS-Bilder sind sehr ähnlich.

# Merkmale von CSS-Bildern (II)

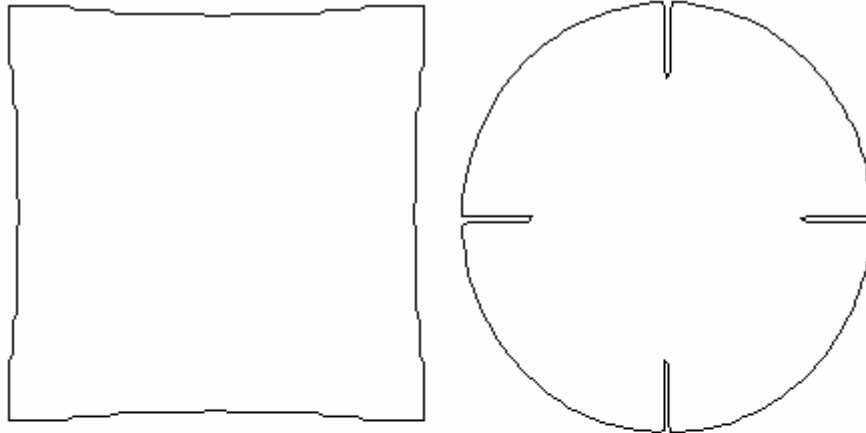
## Vorteile

- Gute Klassifikationsergebnisse
- Nur wenige Werte werden zur Beschreibung komplexer Objekte benötigt.
- Der Ansatz ist invariant gegenüber Rotationen oder Skalierungen.
- Robust bei Rauschen und perspektivischen Verzerrungen
- Schnelle Berechnung der Merkmale und beim Vergleich zweier Konturen.

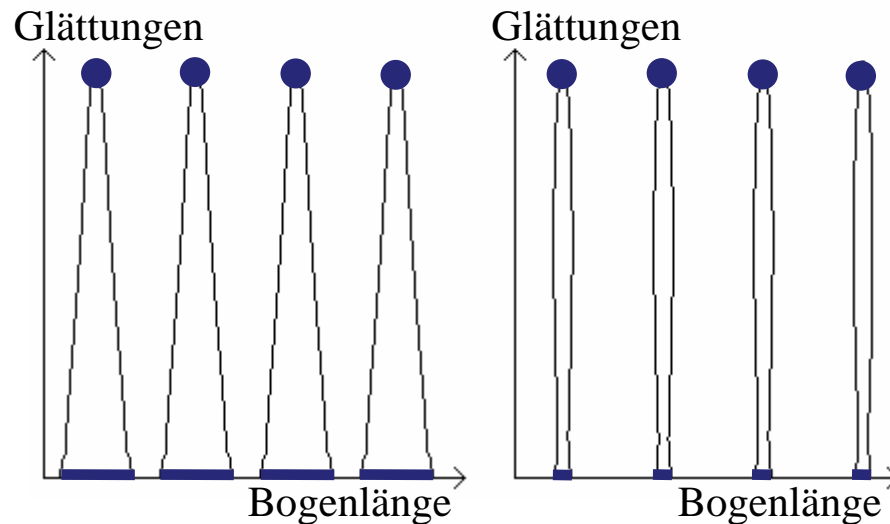
## Nachteile

- Schlechte Klassifikationsergebnisse bei einzelnen Konturen:
  - Mehrdeutigkeiten
  - Konvexe Objektregionen werden nicht ausreichend berücksichtigt

# Mehrdeutigkeiten von CSS-Bildern (I)



- Die **Höhe eines Bogens** beschreibt die Länge der gekrümmten Region **und** die Stärke der Krümmung.



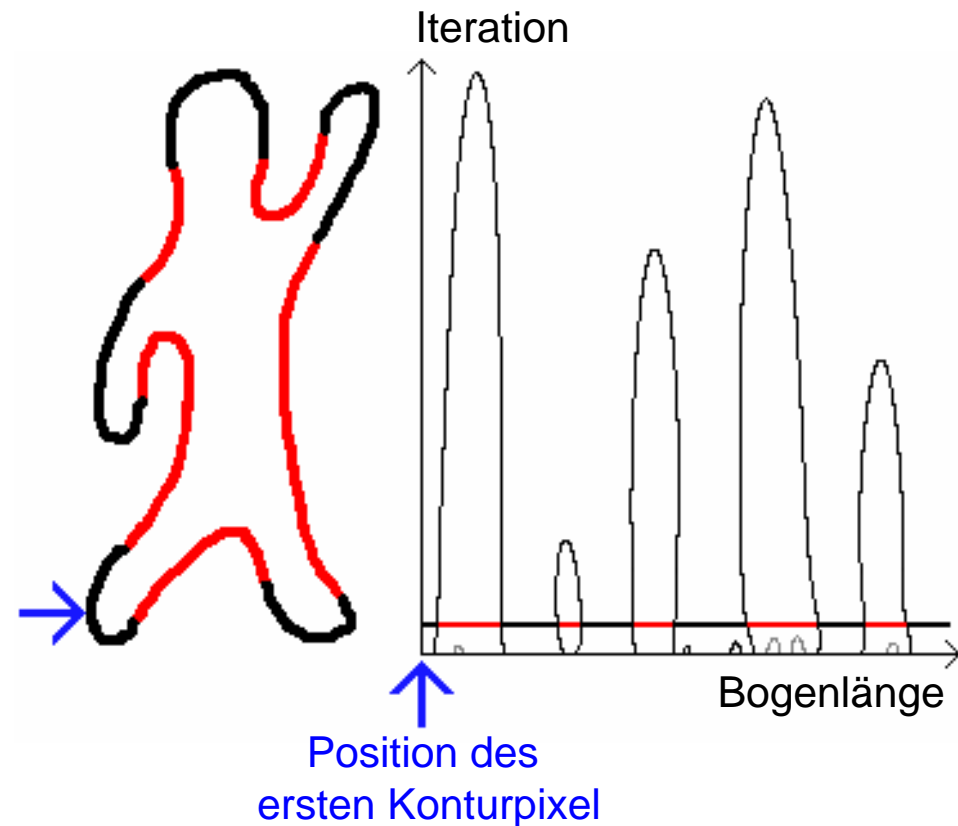
- Für jeden Bogen wird zusätzlich dessen **Breite** gespeichert.

# Mehrdeutigkeiten von CSS-Bildern (II)

## Konvexe Regionen

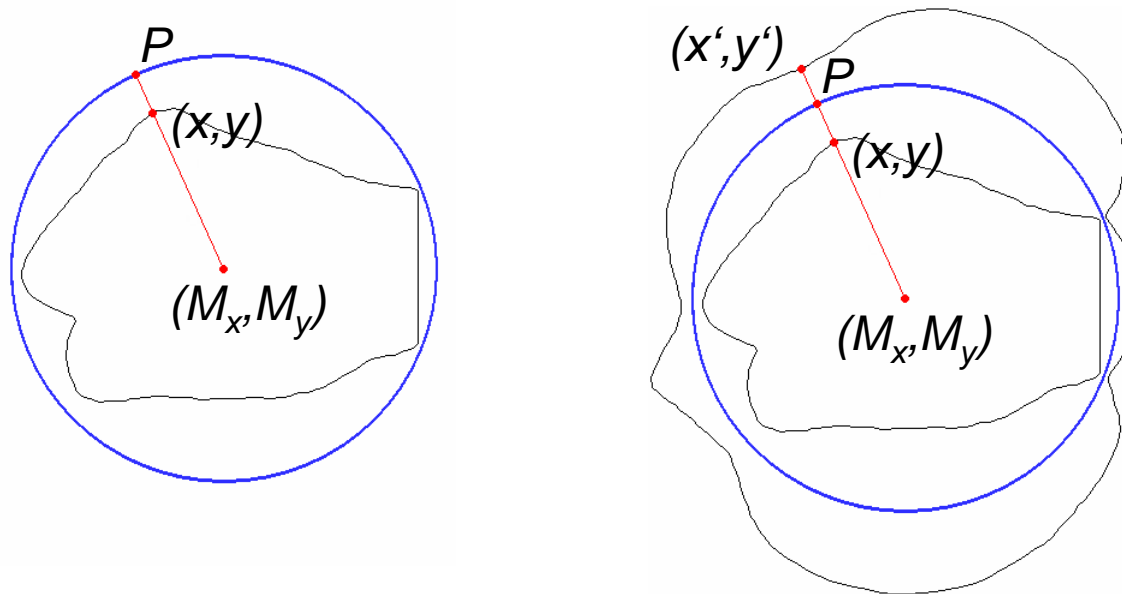
- Schlechte Repräsentation der konvexen Region einer Kontur.
- Konvexe Objekte können gar nicht unterschieden werden.

→ Lösung:  
Spiegelung der Kontur

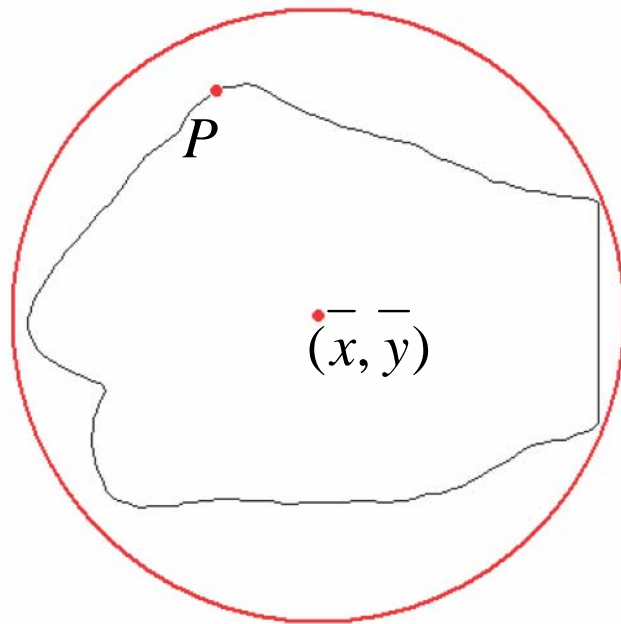


# Gespiegelte Konturen (I)

- **Idee:** Spiegele jedes Konturpixel an einem die Kontur umgebenden Kreis und erzeuge eine neue Kontur.
- Starke konvexe Regionen der ursprünglichen Kontur werden konkave Regionen der gespiegelten Kontur.



# Gespiegelte Konturen (II)

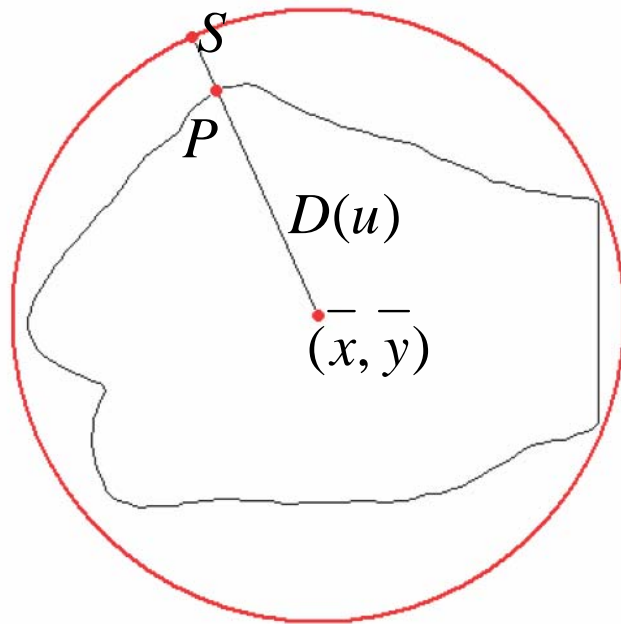


$(\bar{x}, \bar{y})$  Schwerpunkt der Konturpixel

$P$  Konturpixel

$$R = \max_u \{ \sqrt{(\bar{x} - x(u))^2 + (\bar{y} - y(u))^2} \}$$

# Gespiegelte Konturen (III)



$(\bar{x}, \bar{y})$  Schwerpunkt der Konturpixel

$P$  Konturpixel

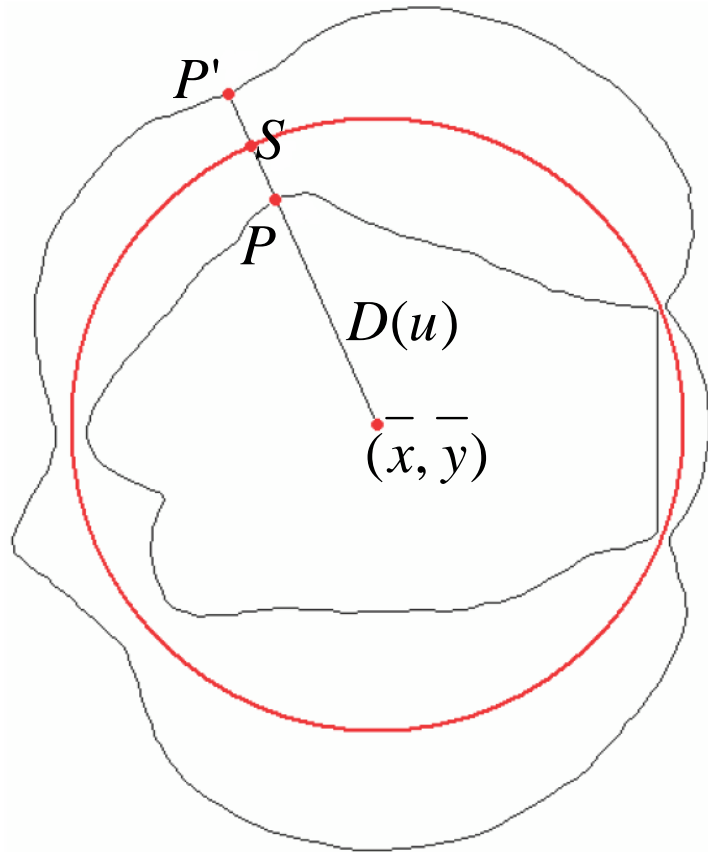
$D(u)$  Entfernung zum Mittelpunkt

$S$  Punkt auf Kreislinie

$$R = \max_u \{ \sqrt{(\bar{x} - x(u))^2 + (\bar{y} - y(u))^2} \}$$

$$D_u = \sqrt{(\bar{x} - x(u))^2 + (\bar{y} - y(u))^2}$$

# Gespiegelte Konturen (IV)



$(\bar{x}, \bar{y})$  Schwerpunkt der Konturpixel

$P$  Konturpixel

$D(u)$  Entfernung zum Mittelpunkt

$S$  Punkt auf Kreislinie

$P'$  Gespiegeltes Konturpixel

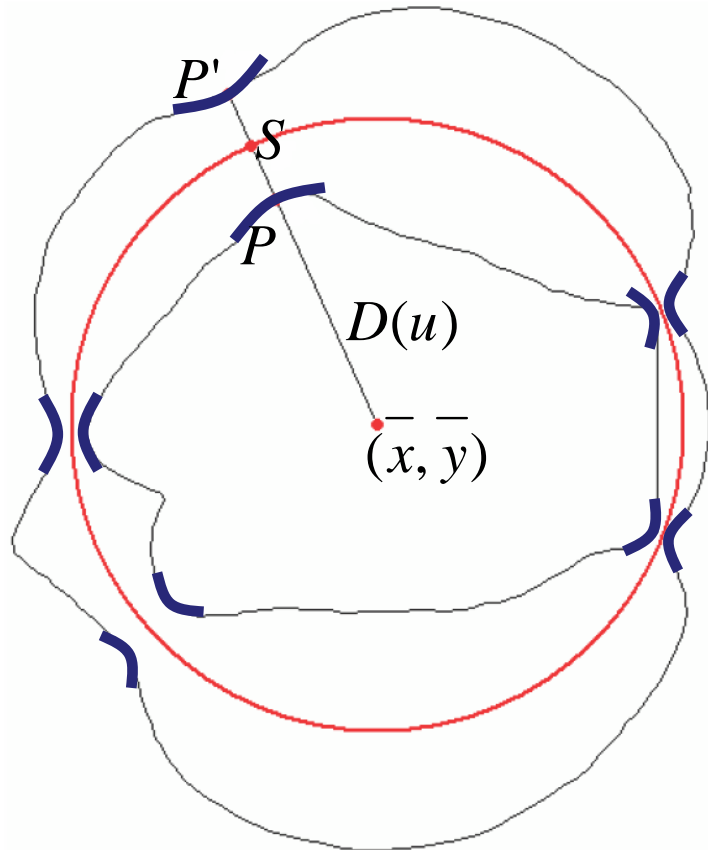
$$R = \max_u \{ \sqrt{(\bar{x} - x(u))^2 + (\bar{y} - y(u))^2} \}$$

$$D_u = \sqrt{(\bar{x} - x(u))^2 + (\bar{y} - y(u))^2}$$

$$x'(u) = \frac{2R - D(u)}{D(u)} \cdot (x(u) - \bar{x}) + \bar{x}$$



# Gespiegelte Konturen (V)



$(\bar{x}, \bar{y})$  Schwerpunkt der Konturpixel

$P$  Konturpixel

$D(u)$  Entfernung zum Mittelpunkt

$S$  Punkt auf Kreislinie

$P'$  Gespiegeltes Konturpixel

$$R = \max_u \{ \sqrt{(\bar{x} - x(u))^2 + (\bar{y} - y(u))^2} \}$$

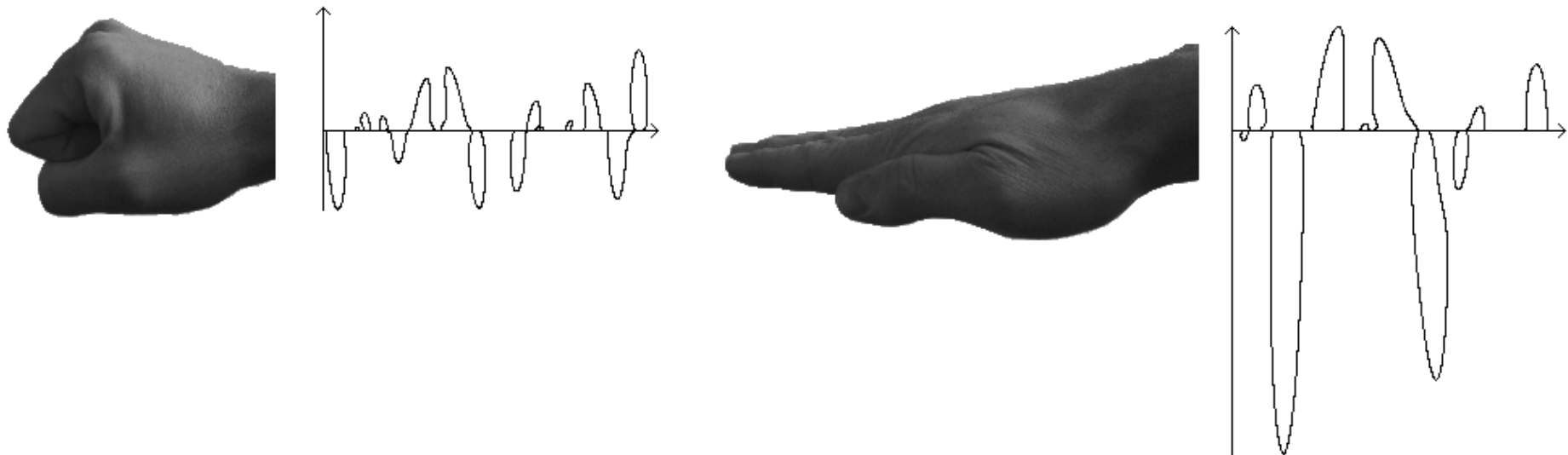
$$D_u = \sqrt{(\bar{x} - x(u))^2 + (\bar{y} - y(u))^2}$$

$$x'(u) = \frac{2R - D(u)}{D(u)} \cdot (x(u) - \bar{x}) + \bar{x}$$

# Gespiegelte Konturen (VI)

## Erweiterter Konturvergleich

- Berechne ursprüngliche Merkmale der CSS-Bilder.
- Berechne Merkmale für die gespiegelte Kontur.



# Objekterkennung in Videos (I)

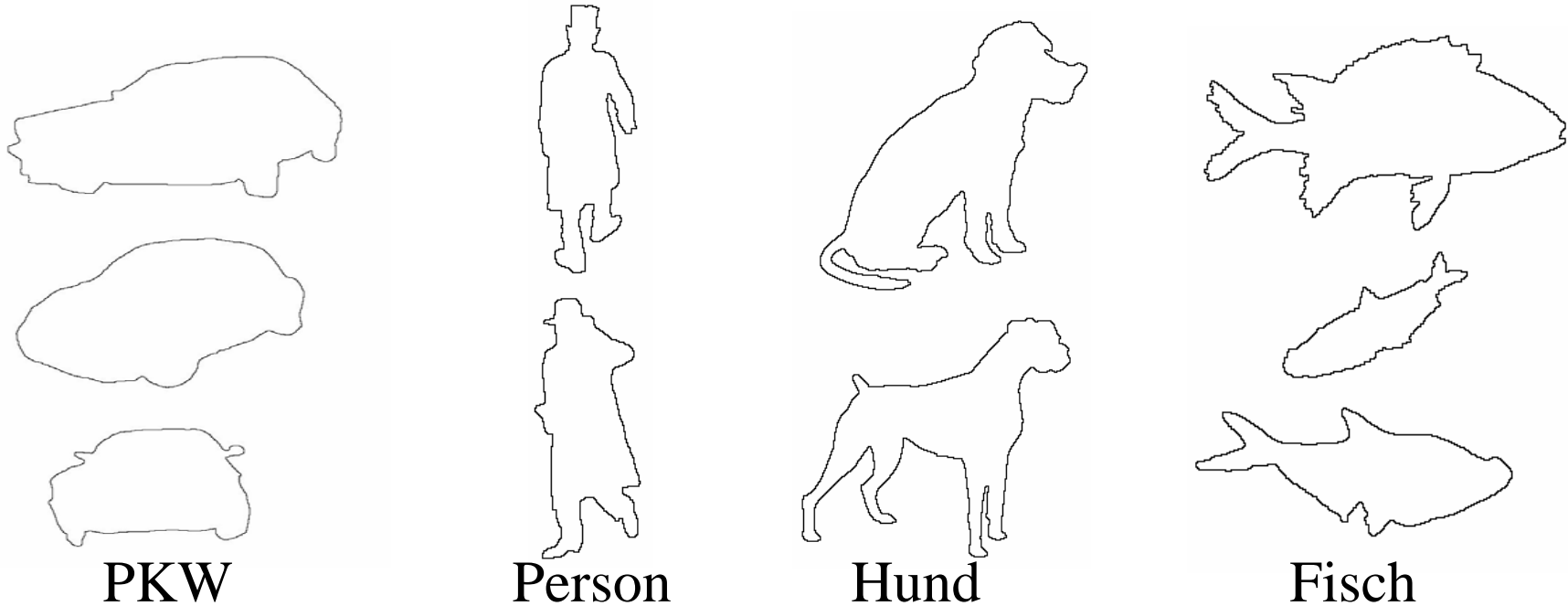
## Ansatz

- Vergleiche jedes Objekt in den einzelnen Bildern mit bekannten Objekten einer Datenbank.
- Berechne die durchschnittliche Differenz zwischen jedem Objekt und eine Objektklasse. Dabei werden ähnliche Objekte in einer Objektklasse zusammengefasst.
- Gebe die ähnlichsten Objekte aus.

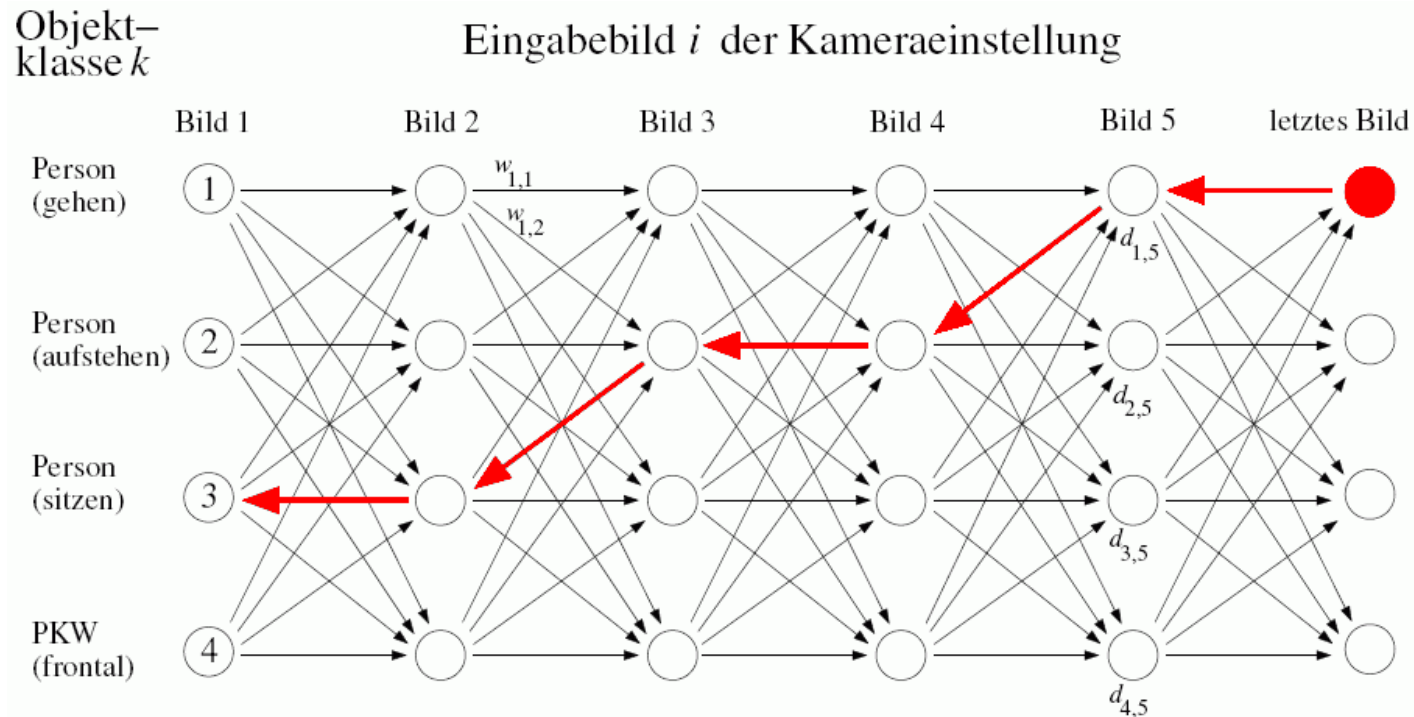
# Objekterkennung in Videos (II)

## Objekte der Datenbank

- Datenbank enthält mehr als 300 Objekte
- 13 Objektklassen gruppieren ähnliche Objekte



# Objekterkennung in Videos (III)



- Übergänge zwischen Objektklassen:  $w_{c_{i-1},i}$
- Knoten  $d_{c_i}$  speichert Differenz des Objektes  $i$  zur Objektklasse  $c_i$
- Pfad mit minimalen Gesamtkosten
- Letzter Knoten mit minimalen Kosten

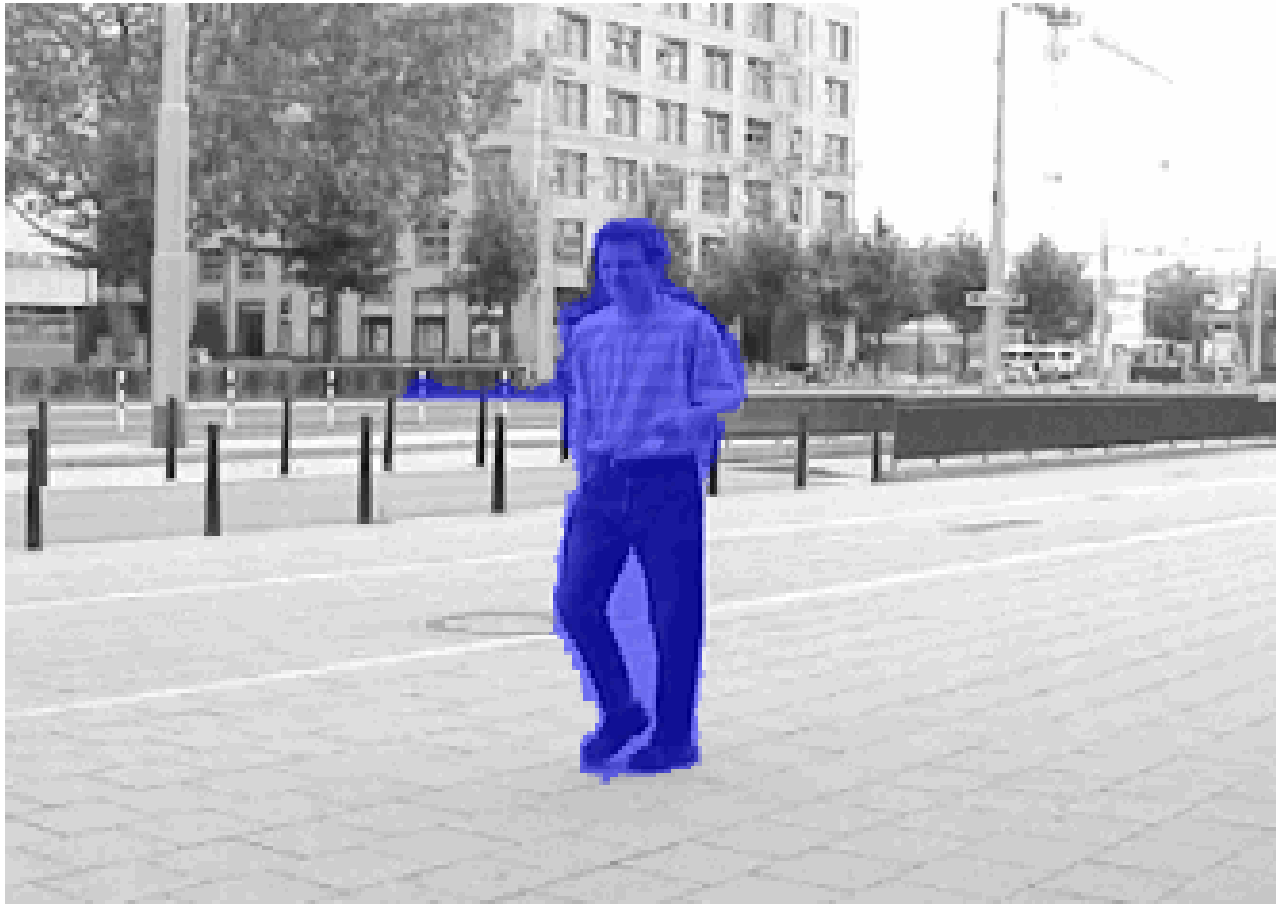
$$K = \min_c \sum_{i=1}^N d_{c_i,i} + w_{c_{i-1},c_i}$$

# Objekterkennung in Videos (IV)

## Erkennungsraten

- Die Erkennungsraten liegen zwischen 25-95 % (abhängig von der Komplexität des Objektes und der Anzahl an verfügbaren Objekten in der Datenbank).
- Starre Objekte (z.B Autos) werden deutlich zuverlässiger erkannt.
- Das CSS-Verfahren ist invariant gegenüber Skalierungen und Rotationen; es ist sehr robust bei Rauschen.
- Ein Vergleich von Konturen ist sehr effizient möglich (einmaliges Glätten, häufiger Vergleich mittels Euklidischer Distanz)

# Objekterkennung in Videos (V)



**stehen**

**gehen**

umdrehen

**hinsetzen**

**sitzen**