



4.4.3 Lokale Bildmerkmale

WS 15/16
Multimedia Grundlagen 1

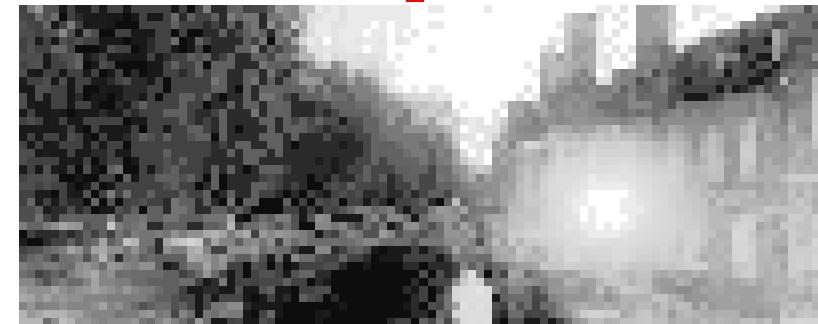
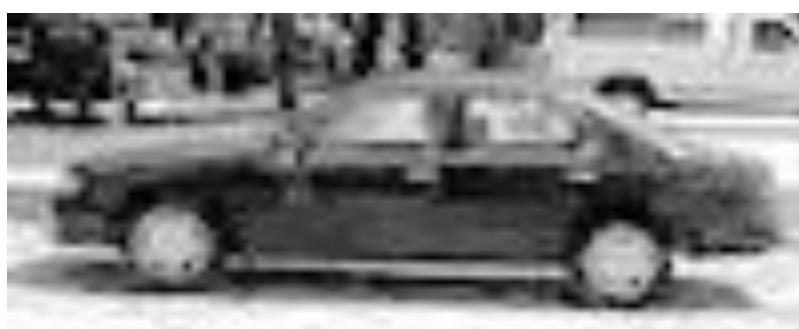
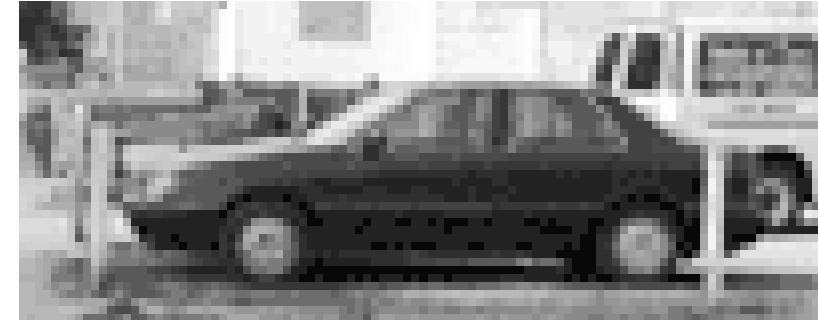
Multimedia Computing & Computer Vision Lab

Rainer.Lienhart@informatik.uni-augsburg.de

www.multimedia-computing.{de,org}



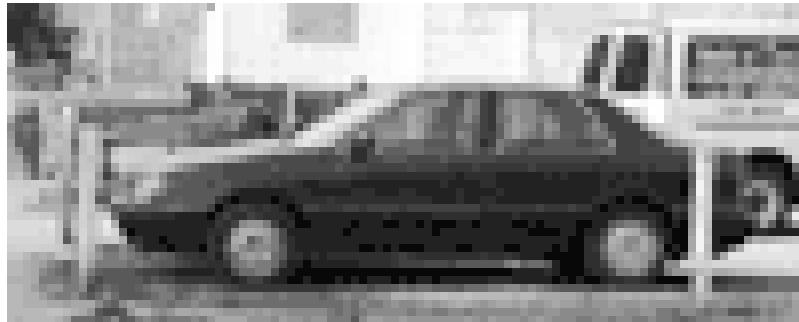
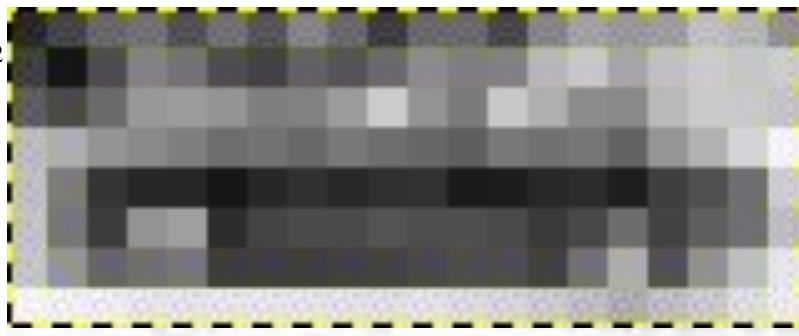
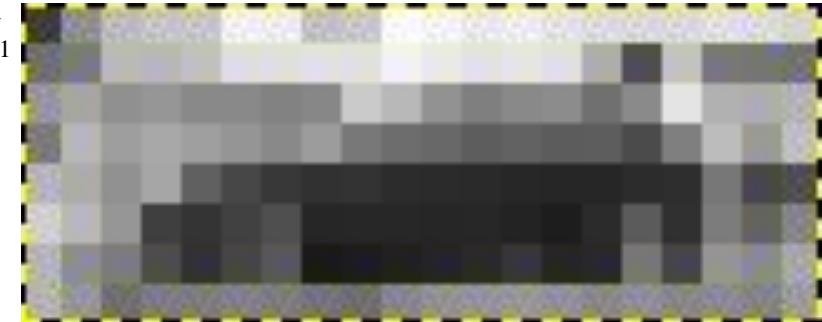
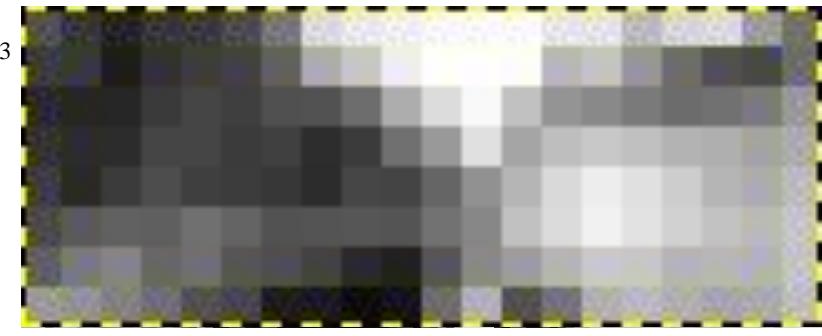
- Abstands- und Ähnlichkeitsmaß
- Lokale Bildmerkmale
 - Selbstähnlichkeit
 - SIFT
 - SURF
 - HOG



= ?

= ?

Annahme: Beide Muster haben die gleiche Größe

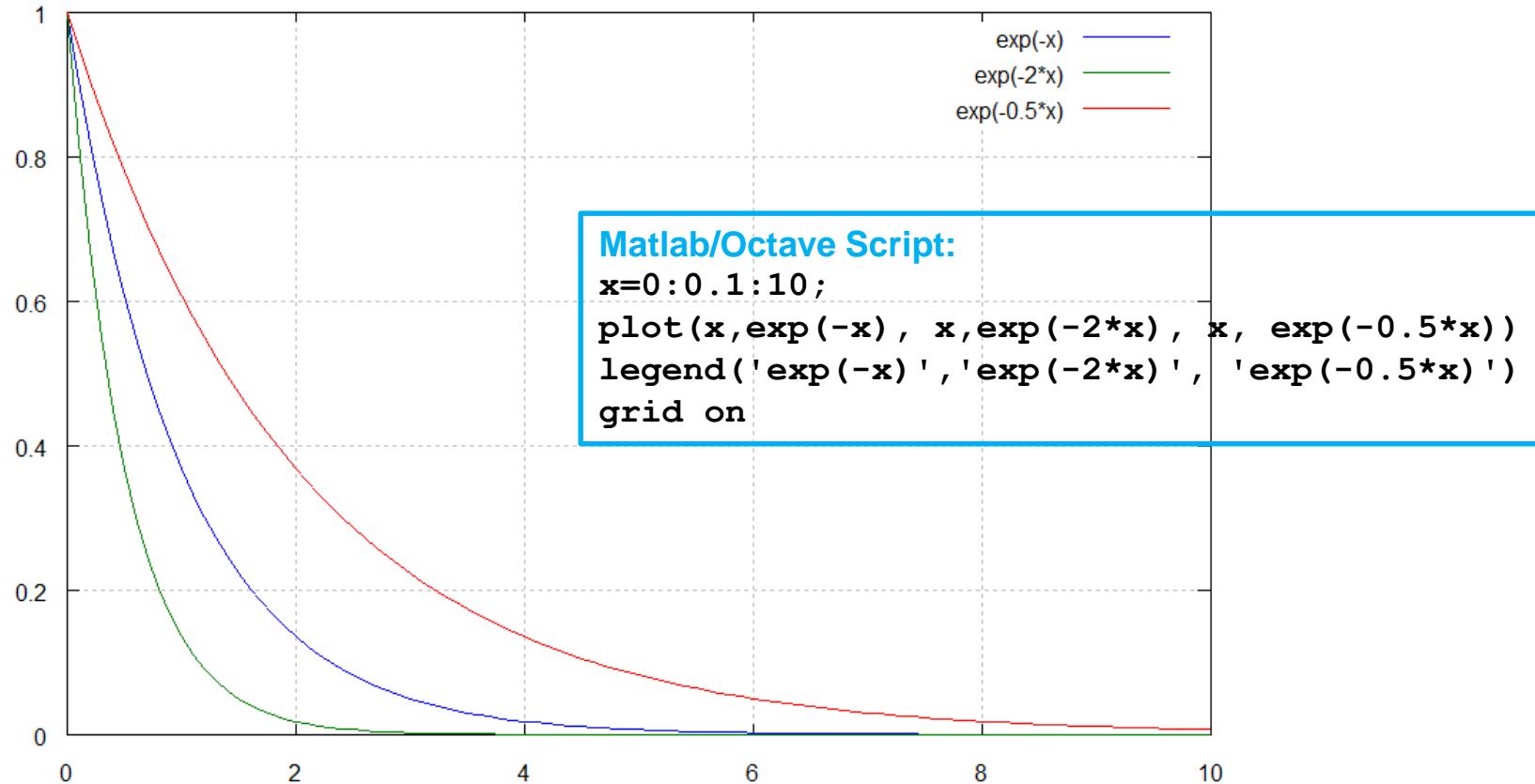
I_1  I_2  I_1  I_3 

$$D(I_1, I_2) = \|I_1 - I_2\|_{L2}^2$$

$$= \sum_{x,y} (I_1(x, y) - I_2(x, y))^2$$

$$D(I_1, I_3) = \|I_1 - I_3\|_{L2}^2$$

$$= \sum_{x,y} (I_1(x, y) - I_3(x, y))^2$$





Selbstähnlichkeit



Ein Bild wird durch die Gesamtheit seiner
lokalen Merkmale beschrieben

Ähnliche „Objekte“ zeigen ähnliche lokale
Merkmale bezüglich:

- Farbe und Helligkeiten
- Muster und Texturen
- Kanten und Gradienten
- Formen
- **Lokale Selbstähnlichkeitsstrukturen**

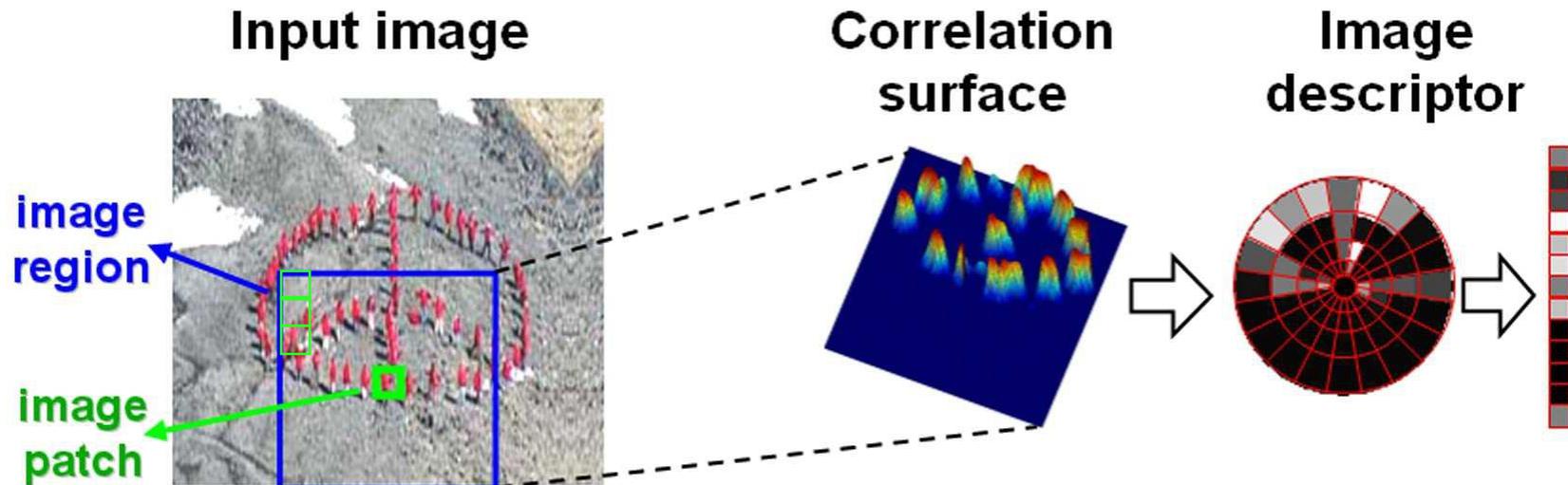
Folgende Bilder unterscheiden sich in so ziemlich allem (u.a. Farbe, Textur, Kanten). Nur das Konzept hinter dem zentralen Objekt ist gleich: ein Herz



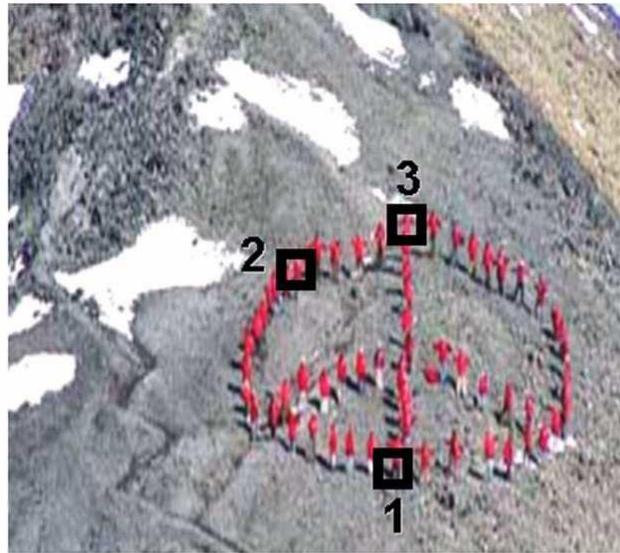
Grundidee:

- Bilder sind ähnlich bezüglich der räumlichen Anordnung der lokalen Selbstähnlichkeiten

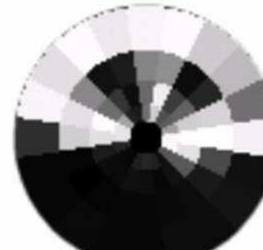
- Lokale Intensitätsmuster werden in der Nachbarschaft in charakteristischer geometrischer Anordnung wiederholt



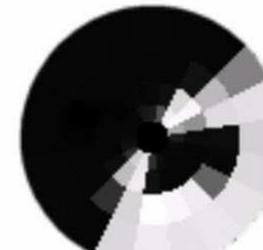
- Ref: E. Shechtman, M. Irani. **Matching Local Self-Similarities across Images and Videos.** In CVPR2007, 2007



$20 * 4 = 80$ Komponenten



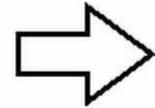
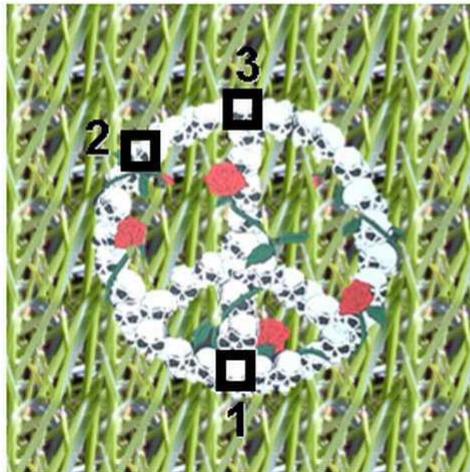
1



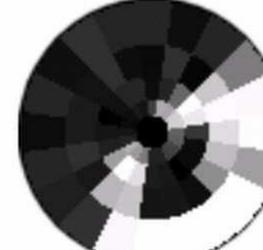
2



3



1



2



3

Suchbeispiel

(a) Vorgabemuster (a)



(b) Beispiele, wo
Vorgabemuster
gefunden wurde

(b)





SIFT (Shift Invariant Feature Transform)

Most parts are taken from
David Lowe's CVPR'03 Tutorial



Bitte lesen:

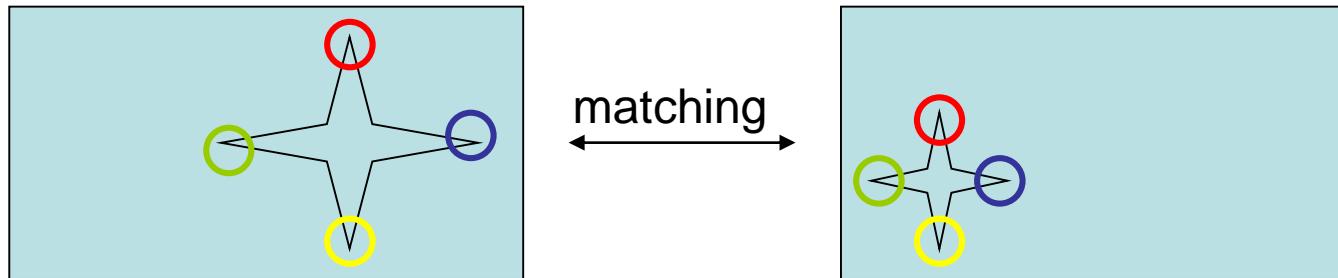
David G. Lowe

"Distinctive image features from scale-invariant keypoints"

International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.

Erkenne charakteristische Punkte in einer Szene zuverlässig wieder,
die in Bildern der gleichen Szene aus unterschiedlicher
Perspektive und mit unterschiedlichen Kameras aufgenommen
wurden

→ Suche nach “Singularitäten” in der 2D bzw. 3D Bildfunktion



- Bildpunktverfolgung vs. Bildpunktindizierung
- **Anwendung:** Stereo Vision, Objekt-/Bildpunktverfolgung, Erkennung von Objektkategorien, Kamerakalibrierung, etc.

Eingabe:



Ausgabe:

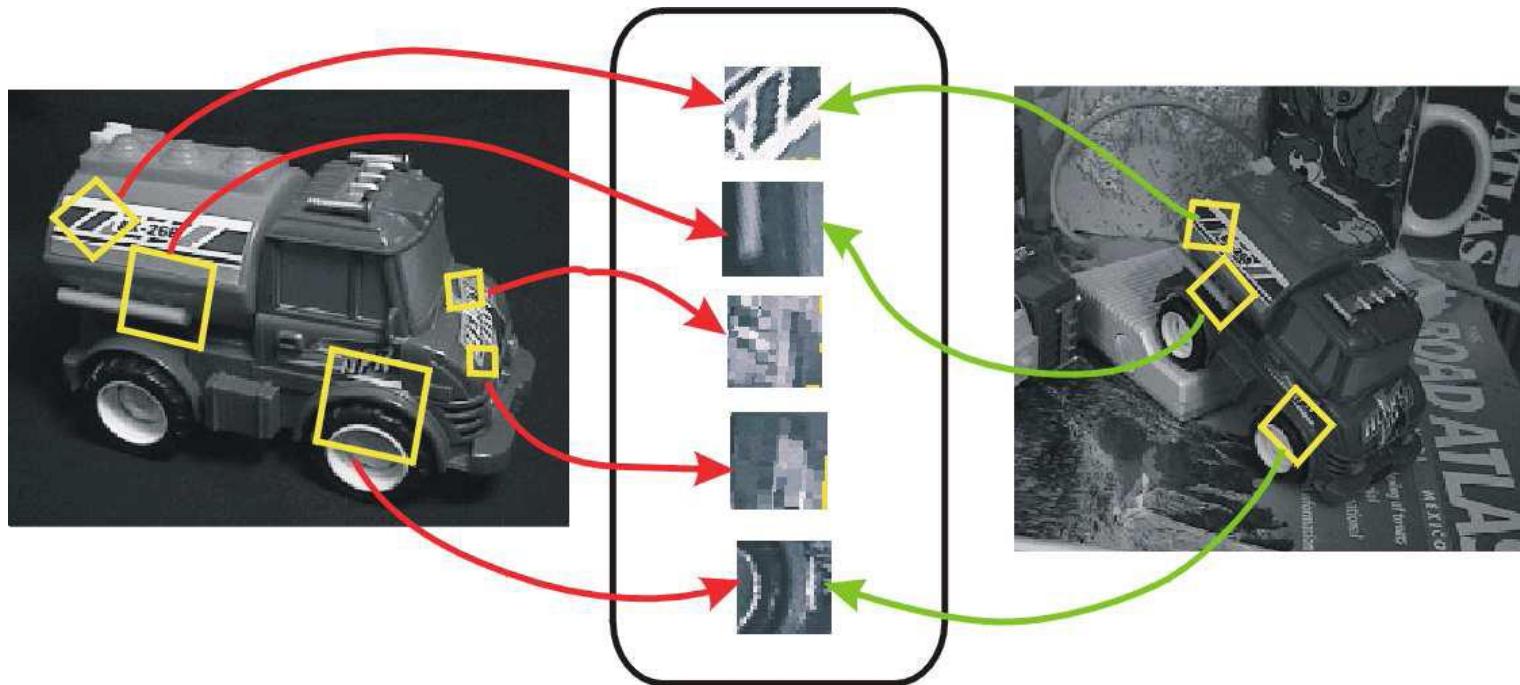




Gewünschte Eigenschaften

- **Allgemein:** Identifizierung hochmarkanter Punkte in einer Szene, welche sich nach Möglichkeit trotz großer Änderungen in den Aufnahmeegebenheiten immer wieder erkennen lassen.
- **Invariant** bezüglich
 - Bildskalierung und Bildrotation
 - Additiven und multiplikativen Beleuchtungsänderungen
- **Teilweise invariant** bezüglich
 - Änderungen in der Beleuchtung (allgemein)
 - Unterschiedlichen Kameraaufnahmeperspektiven
 - Additivem Rauschen
- **Robust** bezüglich beliebigem Rauschen, stark texturiertem und kompliziertem Hintergrund und/oder teilweiser Verdeckung
- **Achtung:** Markante Punkte können objekt-/szenenspezifisch als auch objektklassenspezifisch sein.

Der Bildinhalt um das markante Merkmal wird so in ein lokales Merkmalskoordinatensystem transformiert, dass die lokale Umgebung invariant bezüglich der konkreten Verschiebung, Rotation, Skalierung und anderer Abbildungsparametern erfasst wird



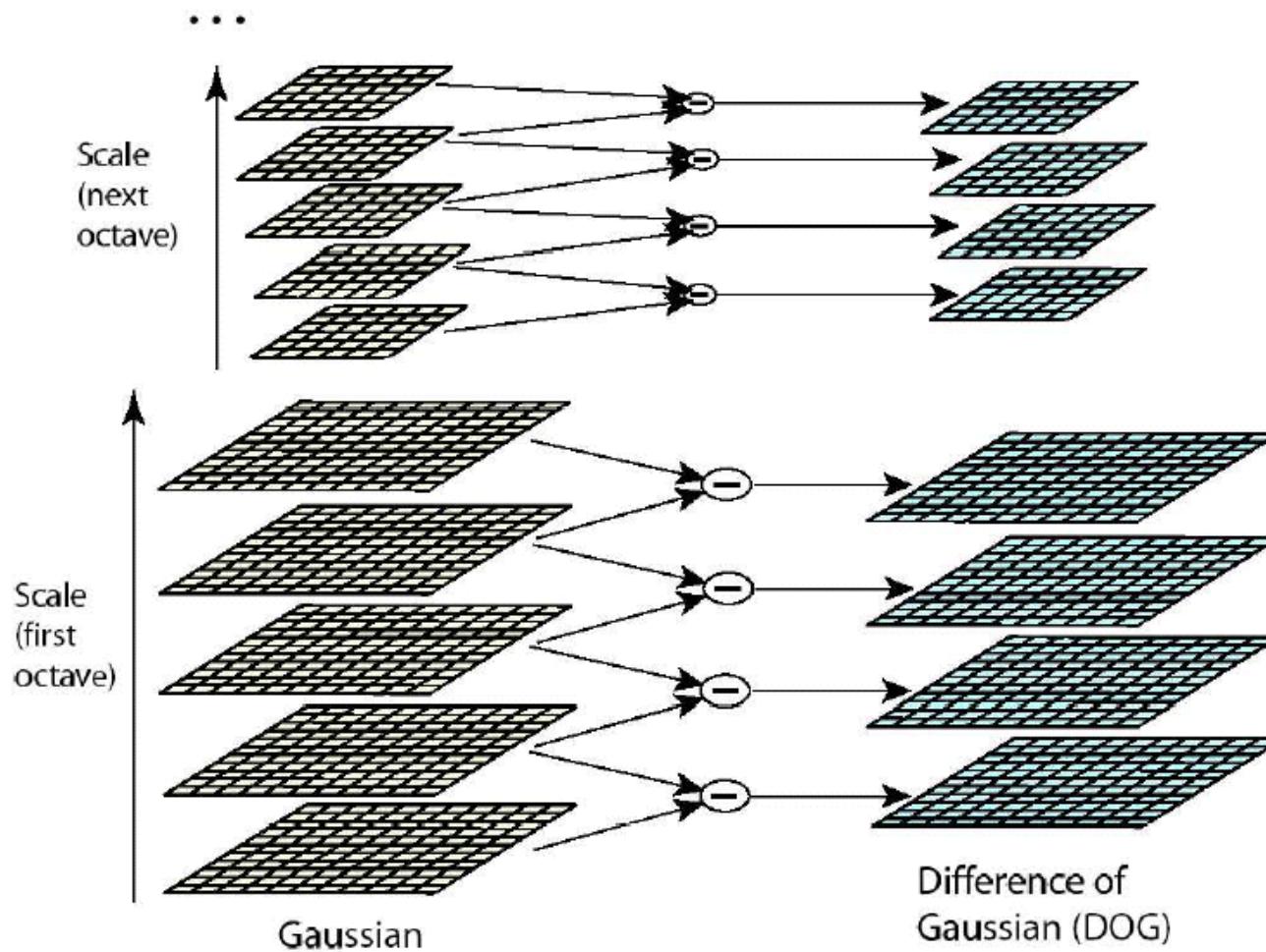


- **Lokalität:** Merkmale sind lokal → robust bezüglich Verdeckungen & Umgebung (Segmentierung ist unnötig)
- **Unterscheidungskraft:** Die Übereinstimmung von individuell markanten Merkmale kann selbst in großen Datenbanken gefunden werden
- **Quantität:** Selbst für kleine Objekte können zahlreiche markante Merkmale gefunden werden
- **Effizienz:** Echtzeitberechnung möglich; skalierbar bezüglich der Zahl von Objekten, die erkannt werden sollen

Verlangt eine Methode, die immer wieder charakteristische Punktumgebungen über die Position und Skala auswählen kann:

- Berechne Bildpyramide $L(x, y, \sigma)$:
$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$
- Berechne DoG Pyramide, $k = \text{const.}, k > 1$:
$$\text{DoG}(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$$
- Aufspüren von Maxima in der DoG Pyramide

Markante Punktkandidaten = Extremstellen in $\text{DoG}(x, y, \sigma)$



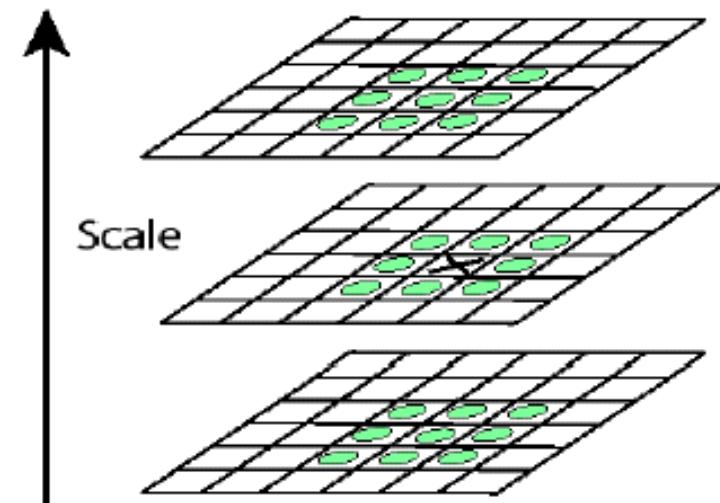
- Abtastrate im Skalenraum
 - $k :=$ Skalenfaktor (e.g., $\text{SQRT}(2)$)
 - $s :=$ # der Schritte pro Oktave \rightarrow ($#+1$) DoG-Bilder
 - $k^s = 2 \leftrightarrow k = 2^{1/s}$ ($\leftrightarrow \text{Id}(k) = 1/s \leftrightarrow s = 1 / \text{Id}(k)$)
 - $s+3$ L-Bilder nötig
 - Empirisch guter Wert: $s = 3$
- Abtastrate im Ortsraum
 - Anfangsglättung mit σ bevor Skalenraumbildung
 - Empirisch guter Wert:
 - Verdopplung der Bildgröße (Hochskalierung)
 - Dann Glättung mit $\sigma=1.6$

- Detect maxima and minima of difference-of-Gaussian in scale space
- Fit a quadratic to surrounding values for sub-pixel and sub-scale interpolation (Brown & Lowe, 2002)
- Taylor expansion around point:

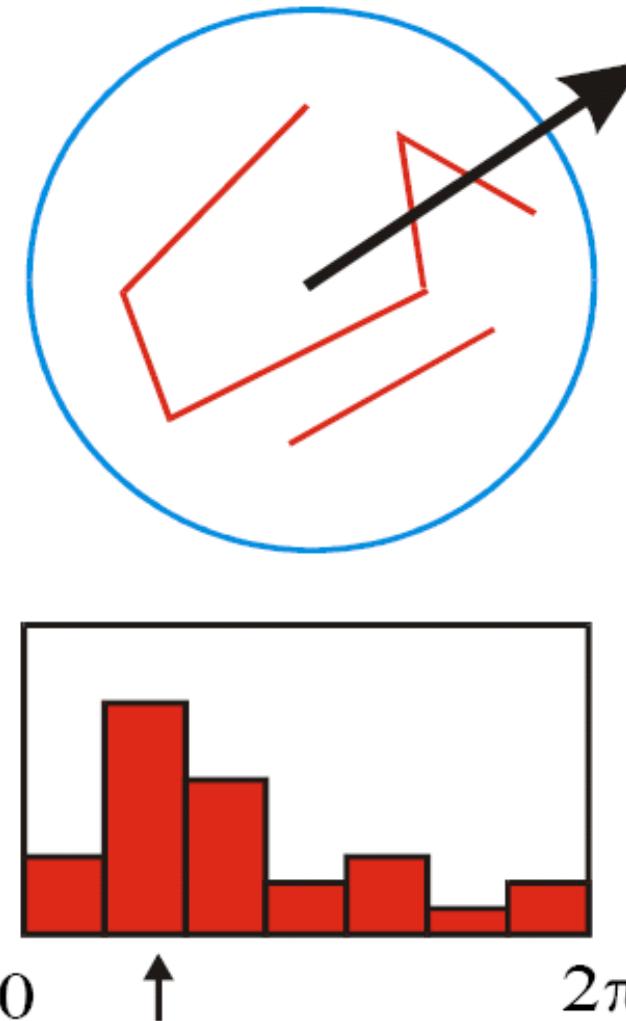
$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2} \mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2} \mathbf{x}$$

- Offset of extremum (use finite differences for derivatives):

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}}$$

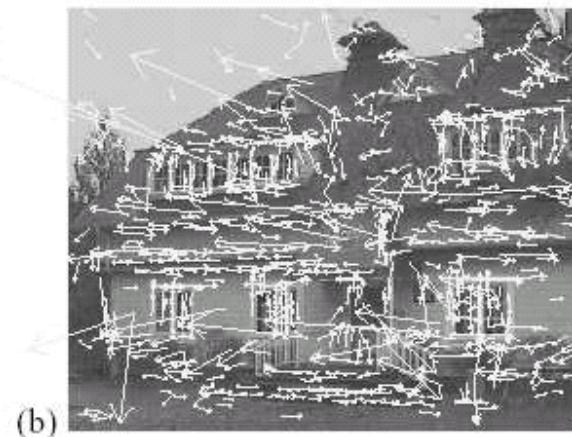


- Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)



Beispiel

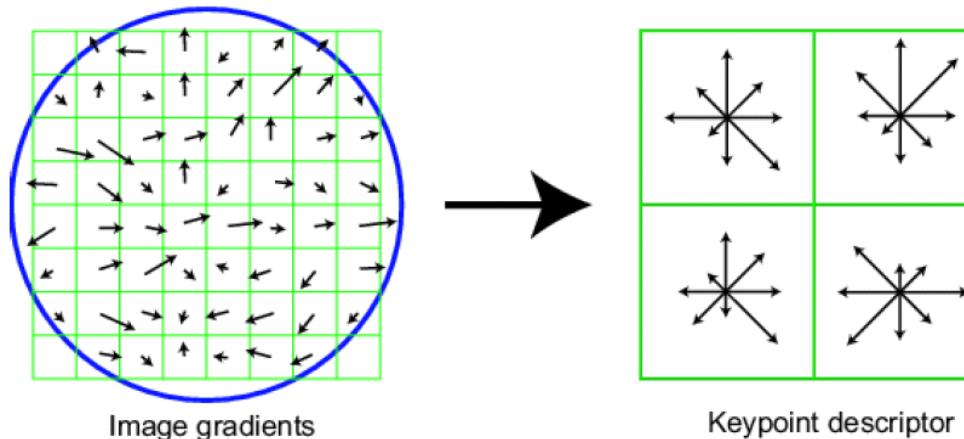
Threshold on value at DOG peak and on ratio of principle curvatures (Harris approach)



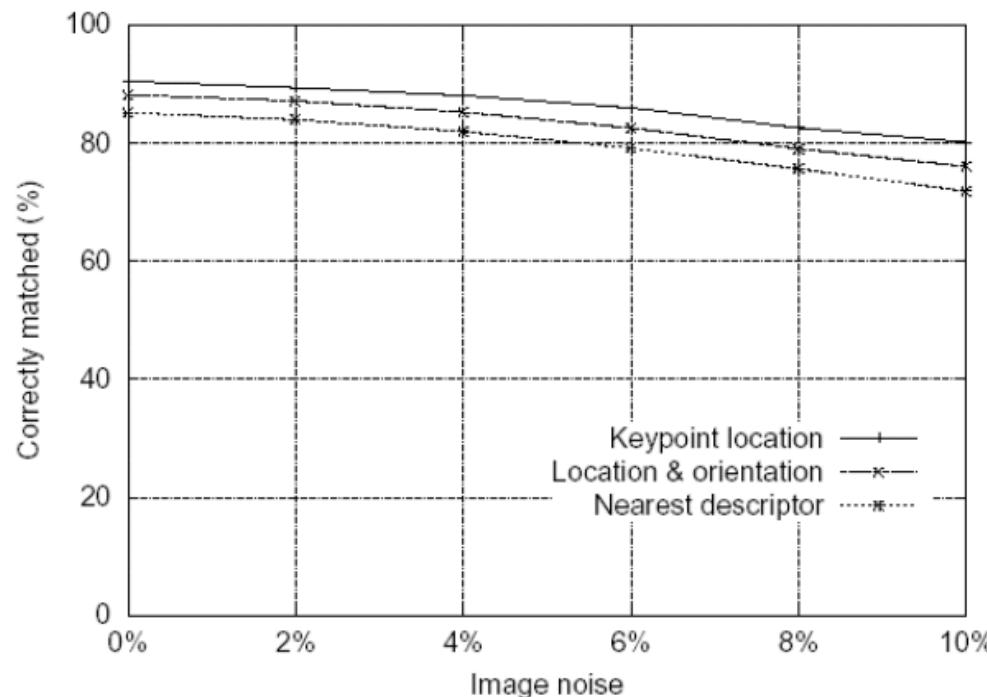
- (a) 233x189 image
- (b) 832 DOG extrema
- (c) 729 left after peak value threshold
- (d) 536 left after testing ratio of principle curvatures



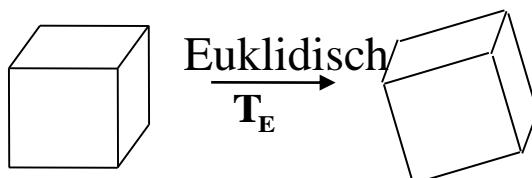
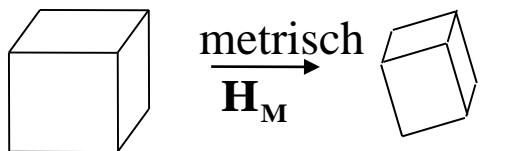
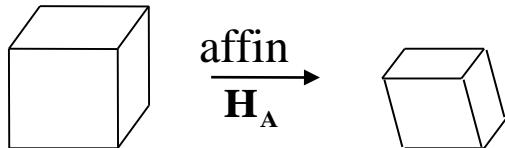
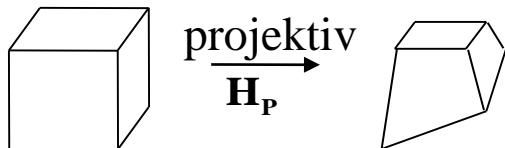
- Merkmalsvektor wird aus einem 16x16 großen Feld im Gradientenbild errechnet
- Erstelle das 2D-Feld von Orientierungshistogrammen
- 8 Orientierungen x 4x4 Histogrammen = 128 Dimensionen



- Finde den ähnlichsten SIFT-Vektor in einer Datenbank von 30,000 SIFT-Vektoren, nachdem das Eingabebild, aus dem der SIFT-Vektor berechnet wurde, zufällig in Größe und Orientierung mit unterschiedlichem Rauschanteil verändert wurde. Bestimme, wie oft der „gleiche“ Punkt wiedergefunden wurde.



Gleiche Würfelformen



Unverändert bleibt:

Kreuzverhältnis,
Berührung der Oberflächen

Parallelität,
Volumenverhältnisse,
relative Distanzen entlang
der Richtungen

Relative Distanzen,
Winkel

Absolute Distanzen,
Volumen

Matrixform

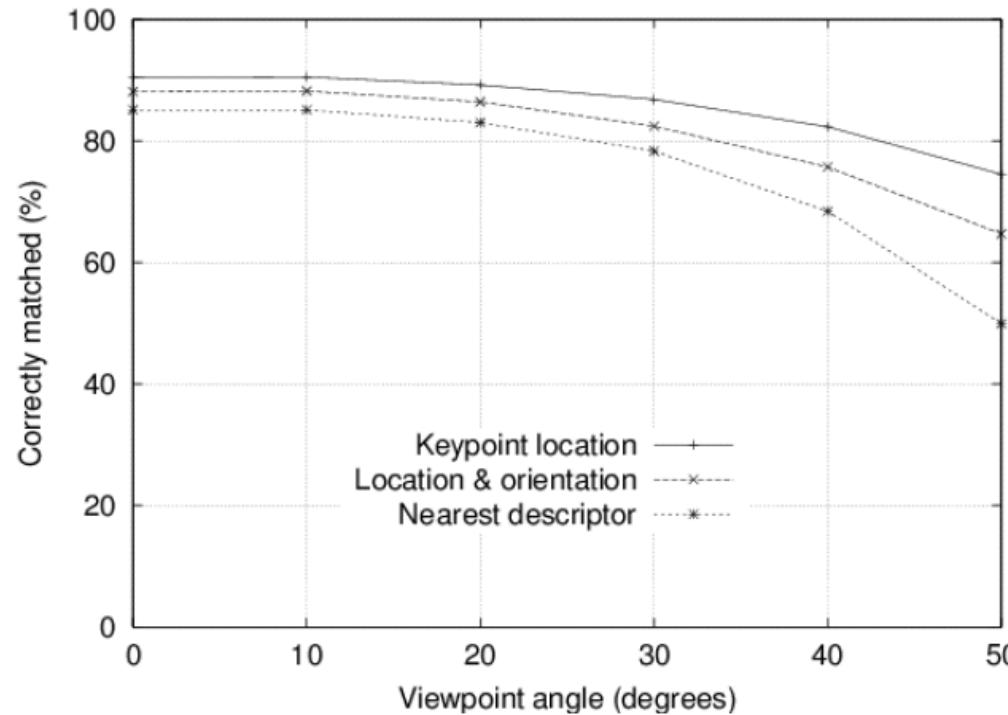
$$\mathbf{H}_P = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{pmatrix}$$

$$\mathbf{H}_A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

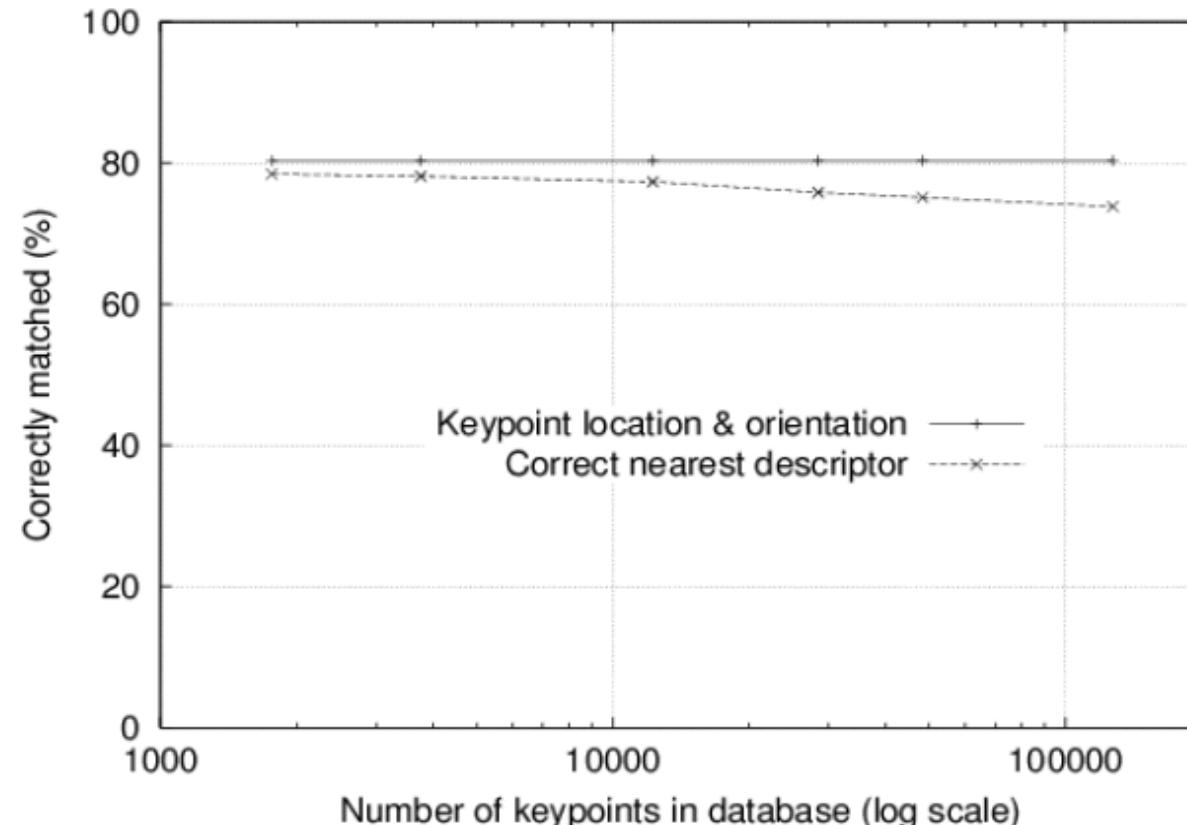
$$\mathbf{H}_M = \begin{pmatrix} \sigma r_{11} & \sigma \cdot r_{12} & \sigma \cdot r_{13} & t_x \\ \sigma r_{21} & \sigma \cdot r_{22} & \sigma \cdot r_{23} & t_y \\ \sigma r_{31} & \sigma \cdot r_{32} & \sigma \cdot r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{H}_E = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

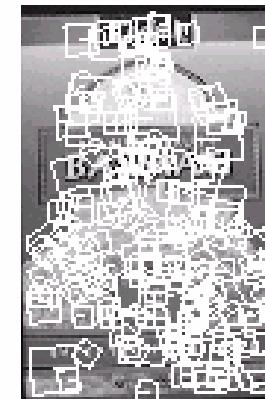
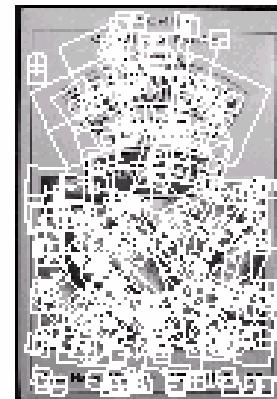
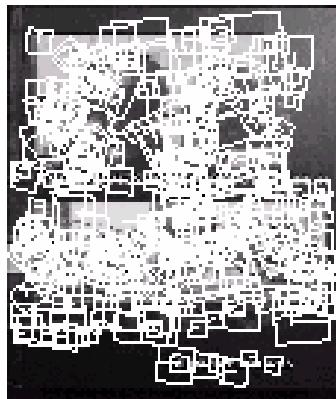
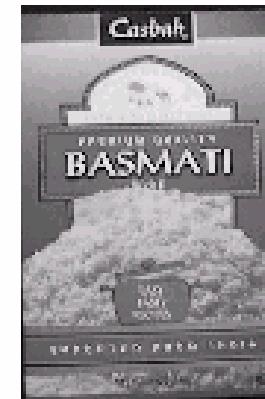
- Finde den ähnlichsten SIFT-Vektor in einer Datenbank von 30,000 SIFT-Vektoren, nachdem das Eingabebild, aus dem der SIFT-Vektor berechnet wurde, zufällig in Größe und Orientierung, mit 2% Rauschen und affinen Verzerrungen verändert wurde. Bestimme, wie oft der „gleiche“ Punkt wiedergefunden wurde.



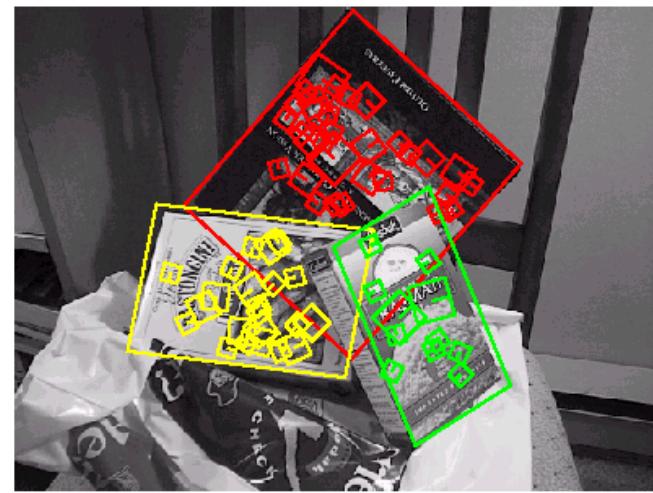
- Bei unterschiedlicher Datenbankgröße, bis zu 30° affinen Verzerrungen und 2% Bildrauschen wurde gemessen, wie oft (in %) das nächste Merkmal in der Referenzdatenbank das Korrekte war.



Mittels SIFT Merkmalen:



- Flache Oberflächen können zuverlässig bis zu 60% in die Tiefe gedreht erkannt werden
- Affine Verzerrungen approximieren die perspektivischen Verzerrungen gut
- Im Prinzip sind nur 3 erkannte Punkte für die Lokalisierung eines Objekts notwendig





Schritt 1:

Bestimmung der
Objekthülle

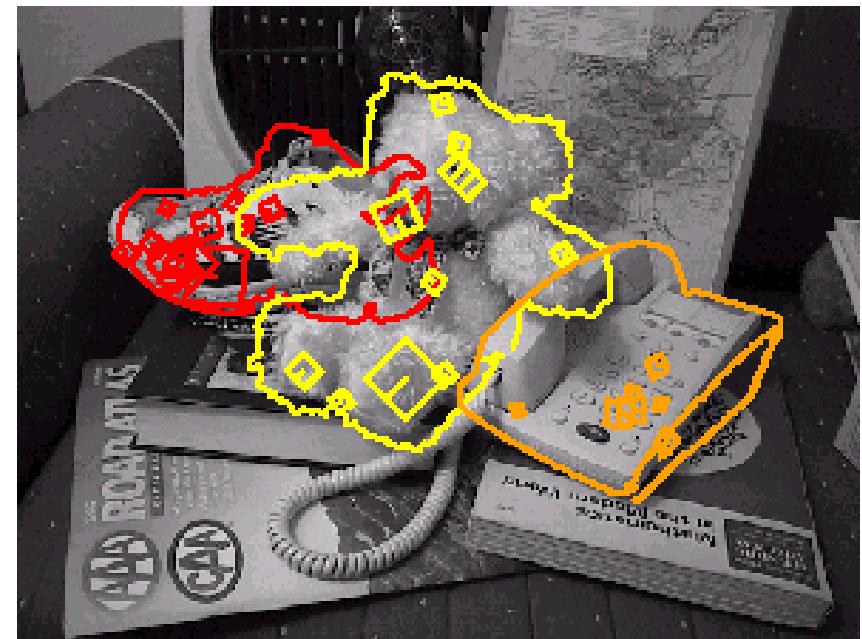
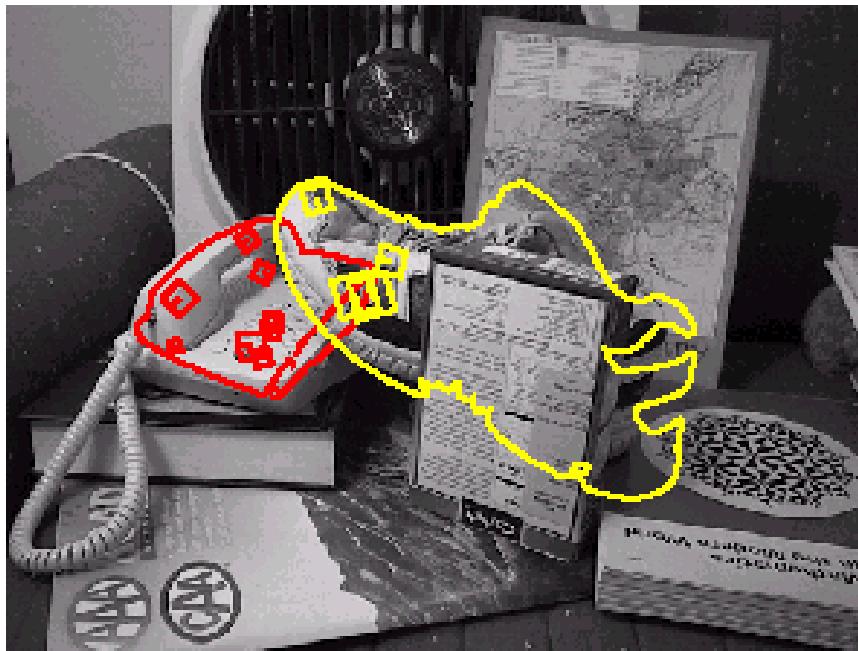
Schritt 2:

Berechnung der SIFT-
Merkmale auf
Objektpixeln

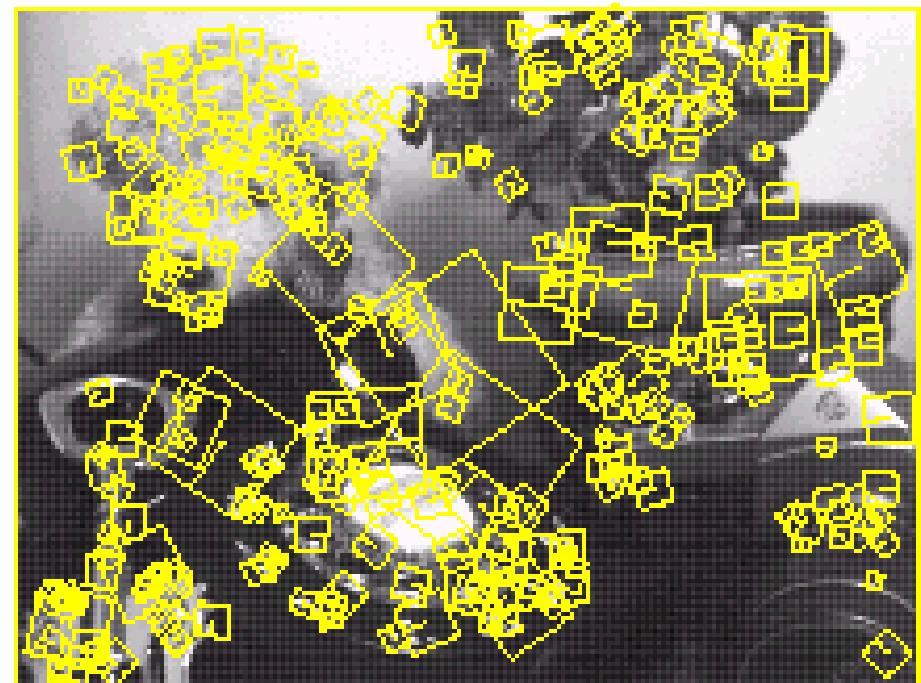


- Ab 3 erkannten SIFT-Merkmalen kann das Objekt lokalisiert werden
- Mehr erkannte Merkmale machen die Erkennung gegenüber Fehlern robuster

Erkennung trotz Verdeckung

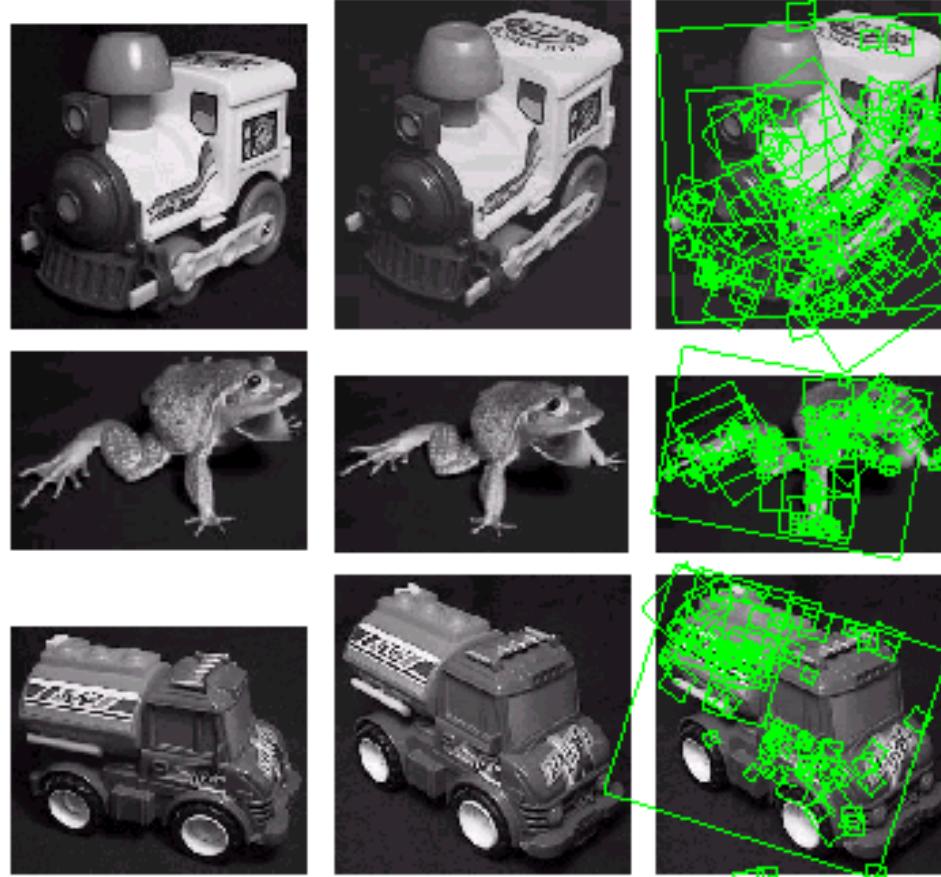


Dasselbe Bilder mit unterschiedlicher Beleuchtung

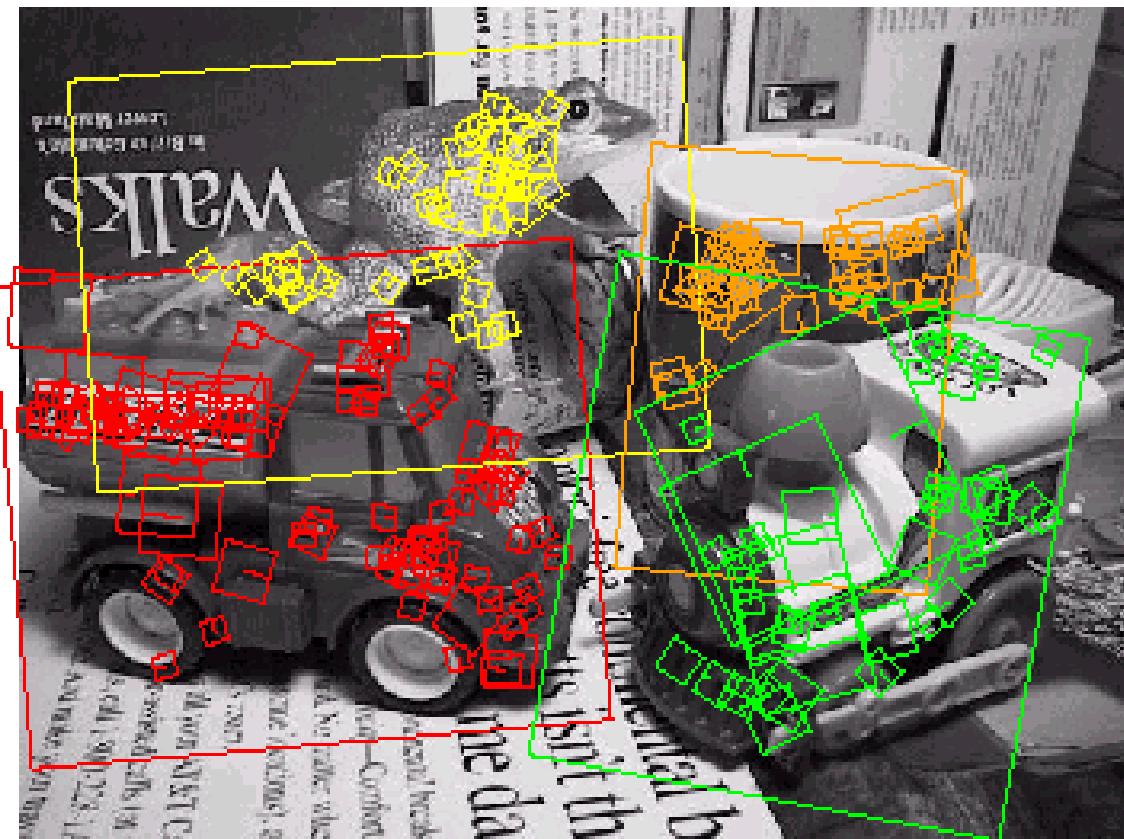


273 übereinstimmende markante
Merkmale gefunden

Test: Unterschiedliche Aufnahmeorte



... in einer kompletten Szene



Weitere Beispiel



- Aufgrund der markanten Merkmale (SIFT) kann die Überlappung selbstständig aus der ungeordneten Menge von Bildern erkannt werden,
- um dann durch Überblenden auf allen Skalen die einzelnen Bilder nahtlos aneinanderzufügen



Das Panorama des Labors wurde automatisch aus 143 Bildern erstellt.

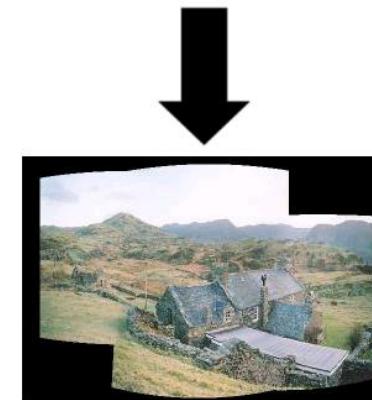
Erzeugung von Panoramafotos

Gegeben:
Menge an Bildern



Input images

Ergebnis:
Menge an Panoramafotos



Output panorama 1

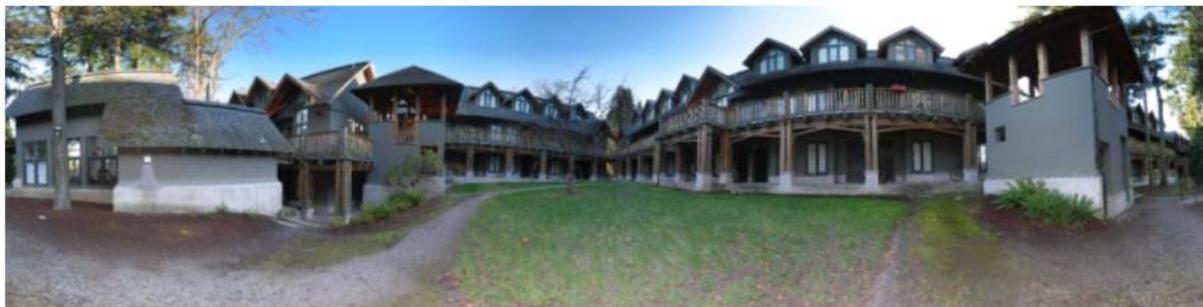




(a) 40 of 80 images registered



(b) All 80 images registered





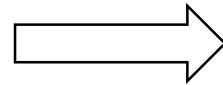
HOG

Histograms of Oriented Gradients

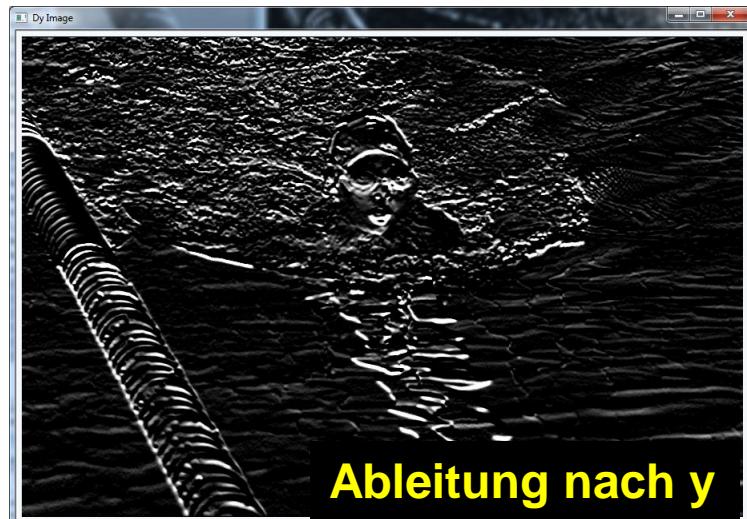
Schritt 1: Ableitung



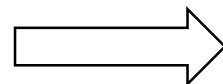
Eingabebild



Ableitung nach x



Ableitung nach y



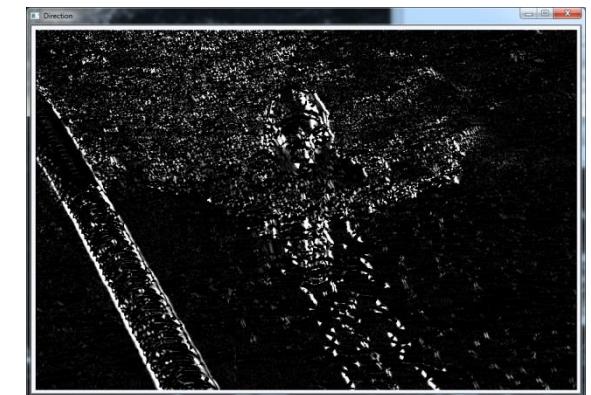
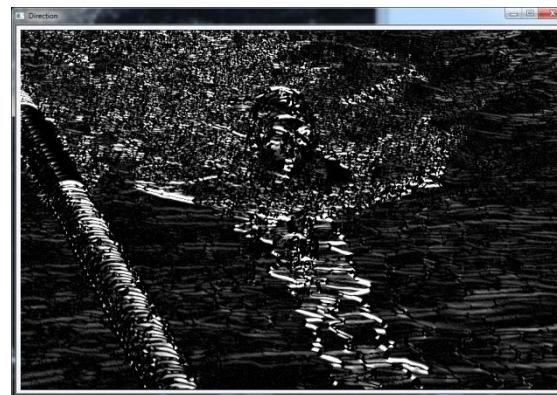
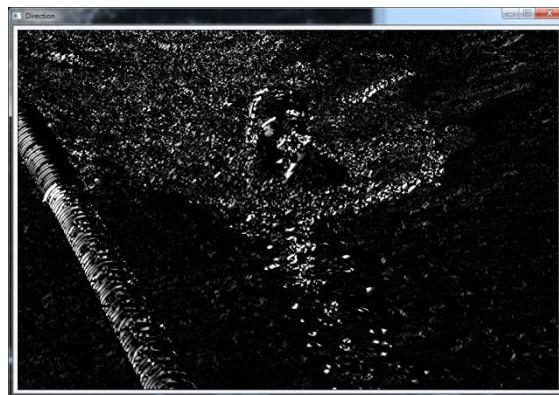
Betrag der Ableitung

2. Richtungsquantisierung

Nur die Orientierung ($0-18^\circ$), nicht die Richtung ($0-360^\circ$) wird berücksichtigt

Quantisiert in 9 Orientierungsintervalle

Beispiel: (Nur 3 Orientierungsintervalle hier)

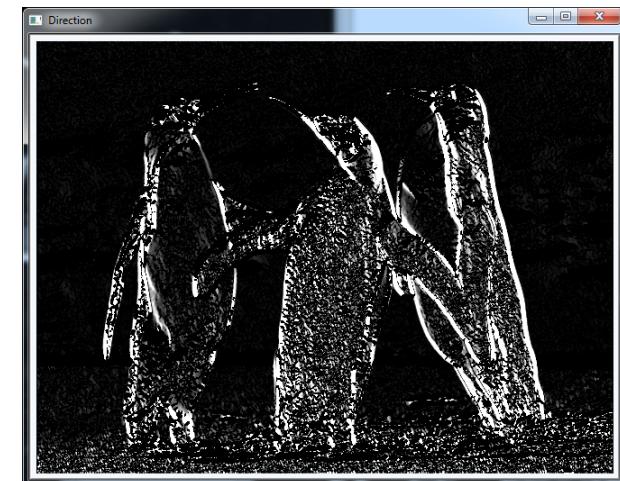


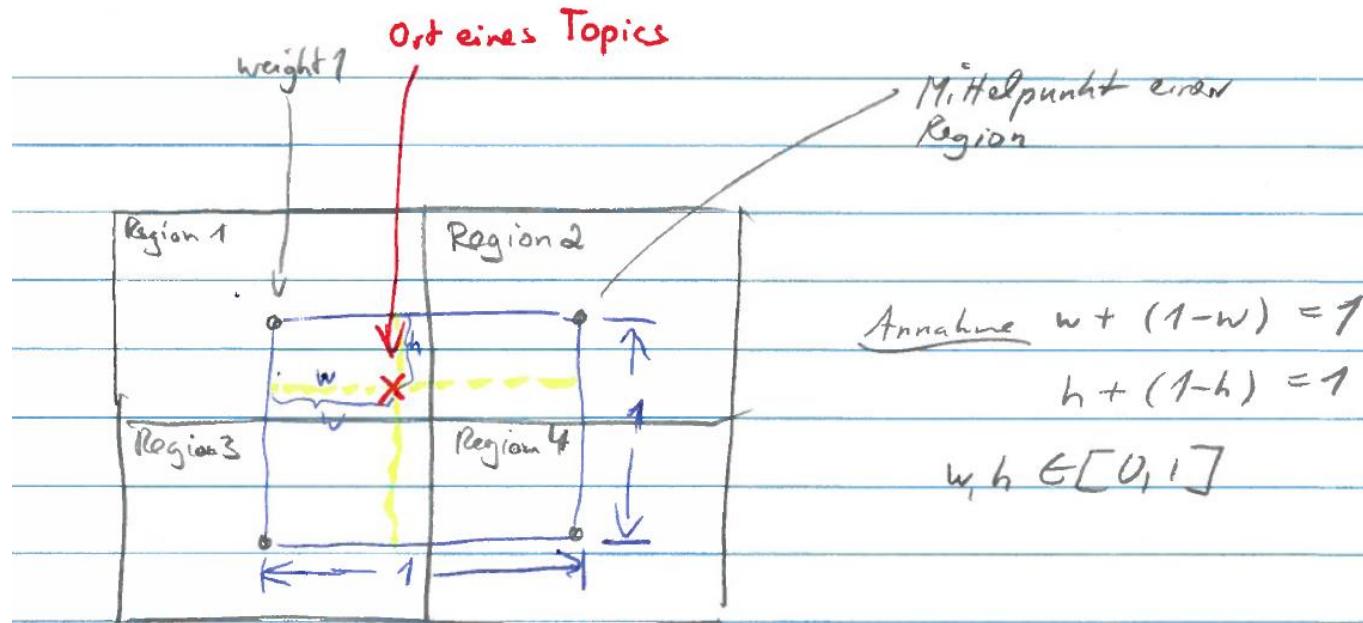
2. Richtungsquantisierung

Nur die Orientierung ($0-180^\circ$), nicht die Richtung ($0-360^\circ$) wird berücksichtigt

Quantisiert in 9 Orientierungsintervalle

Beispiel: (Nur 3 Orientierungsintervalle hier)





$$\text{bei weight 1} = (1-w) \cdot (1-h)$$

$$\text{“ 2} = w \cdot (1-h)$$

$$\text{“ 3} = (1-w) \cdot h$$

$$\text{“ 4} = w \cdot h$$

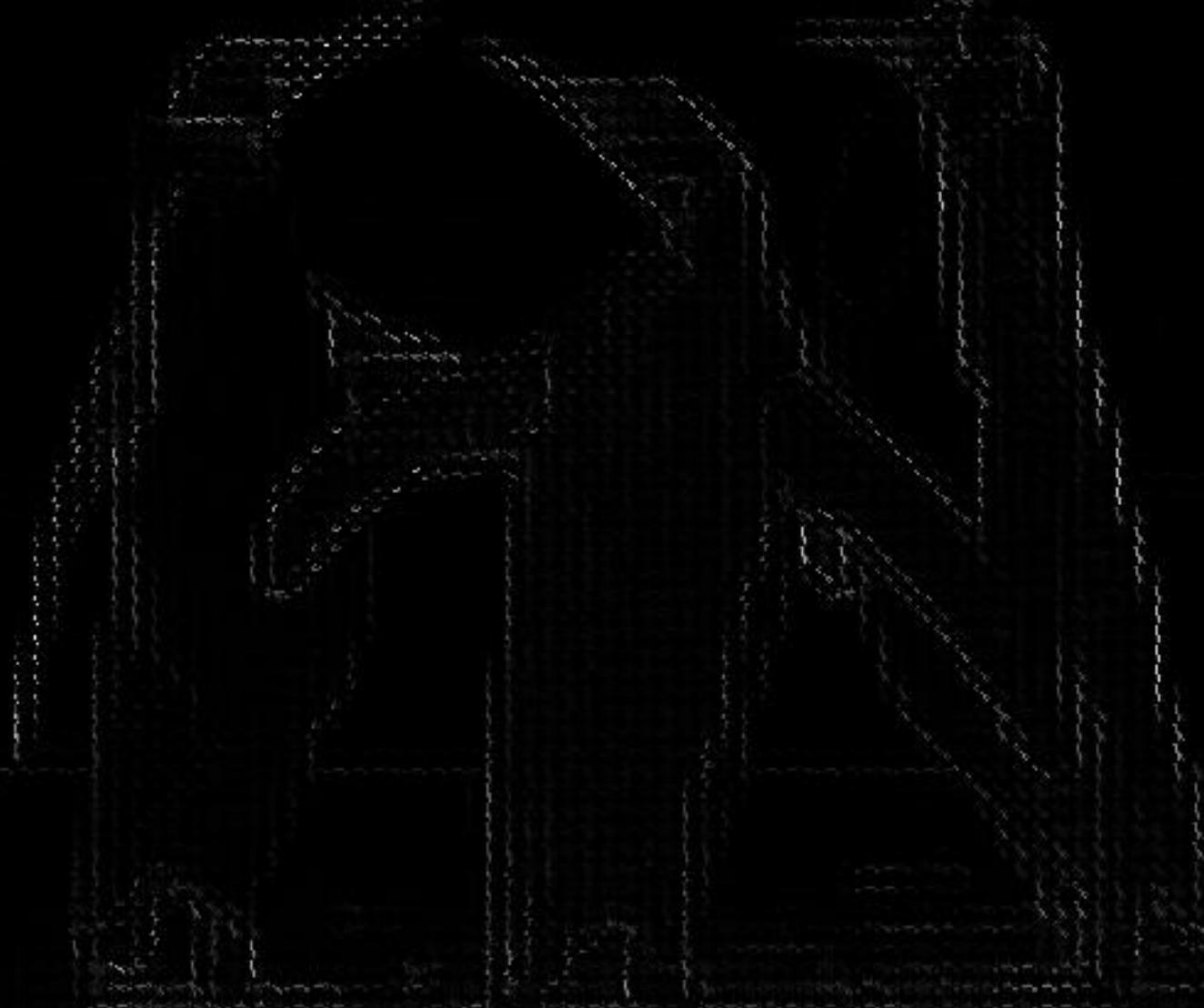
dto. für Orientierung

4. Zusammenföhrung

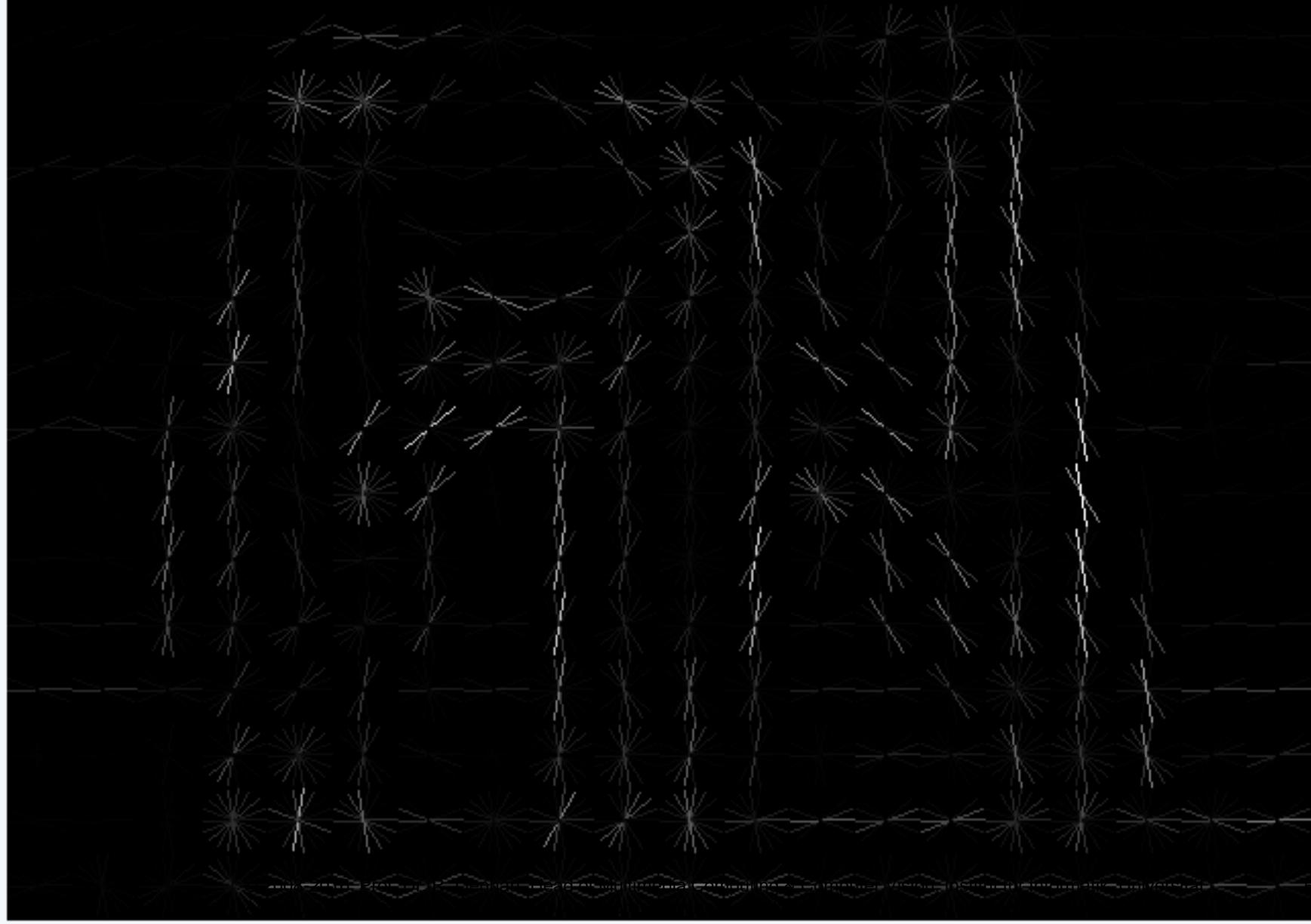
4. Zusammenführung

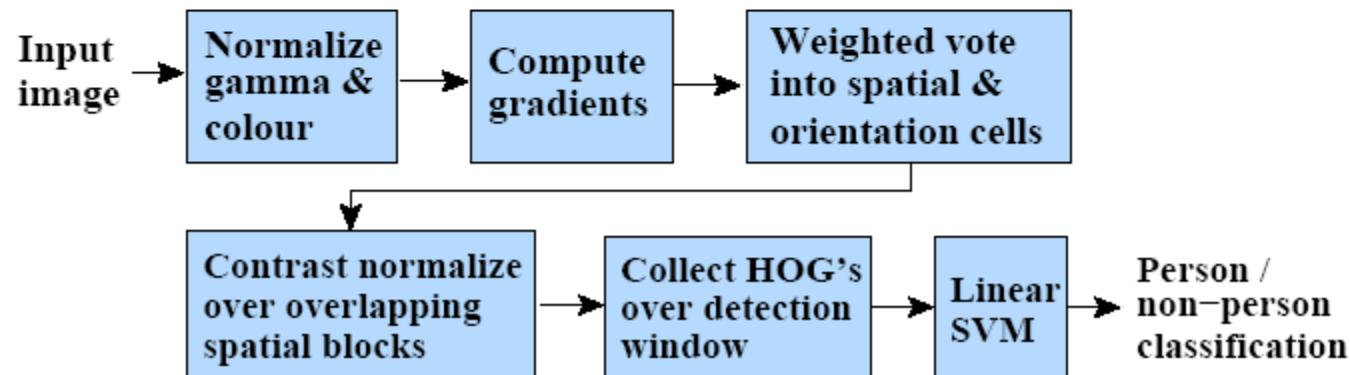
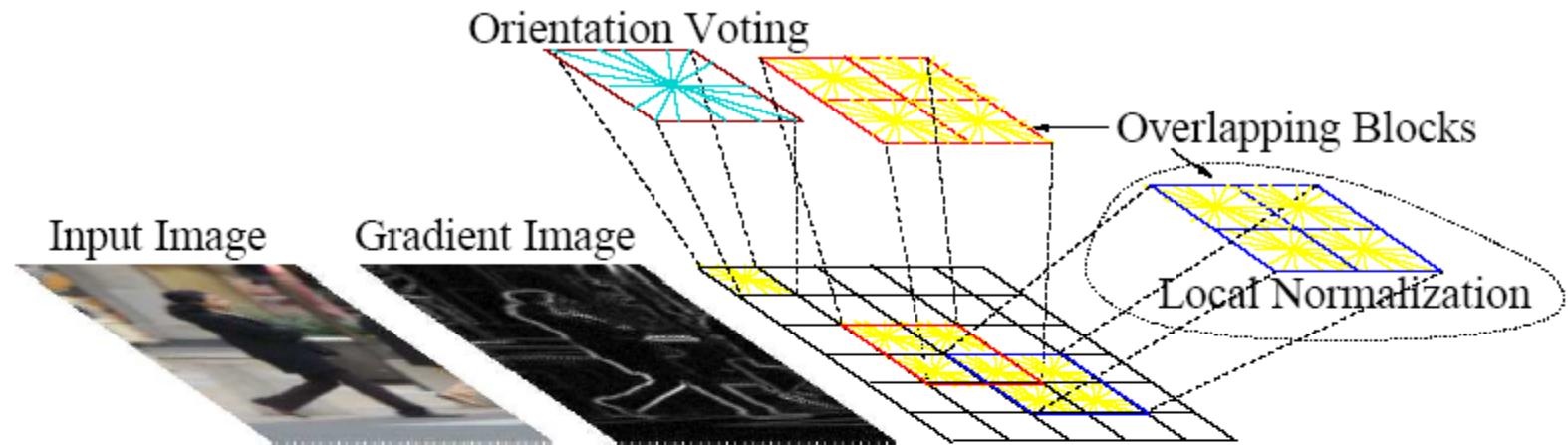


4. Zusammenfuehrung



4. Zusammenführung







Ein Anwendungsbeispiel

- Template images featuring key poses

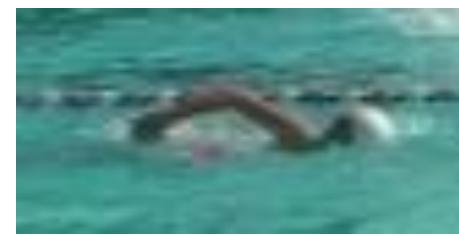
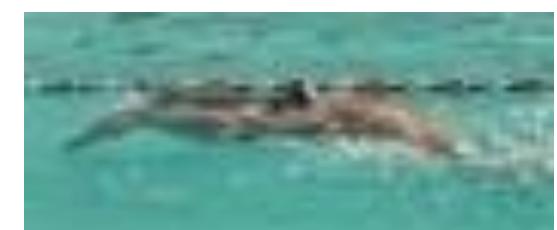
Backstroke



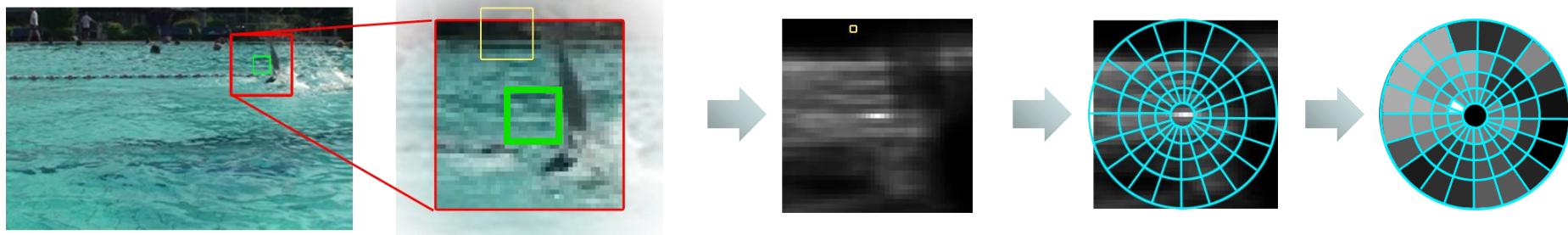
Freestyle



Butterfly

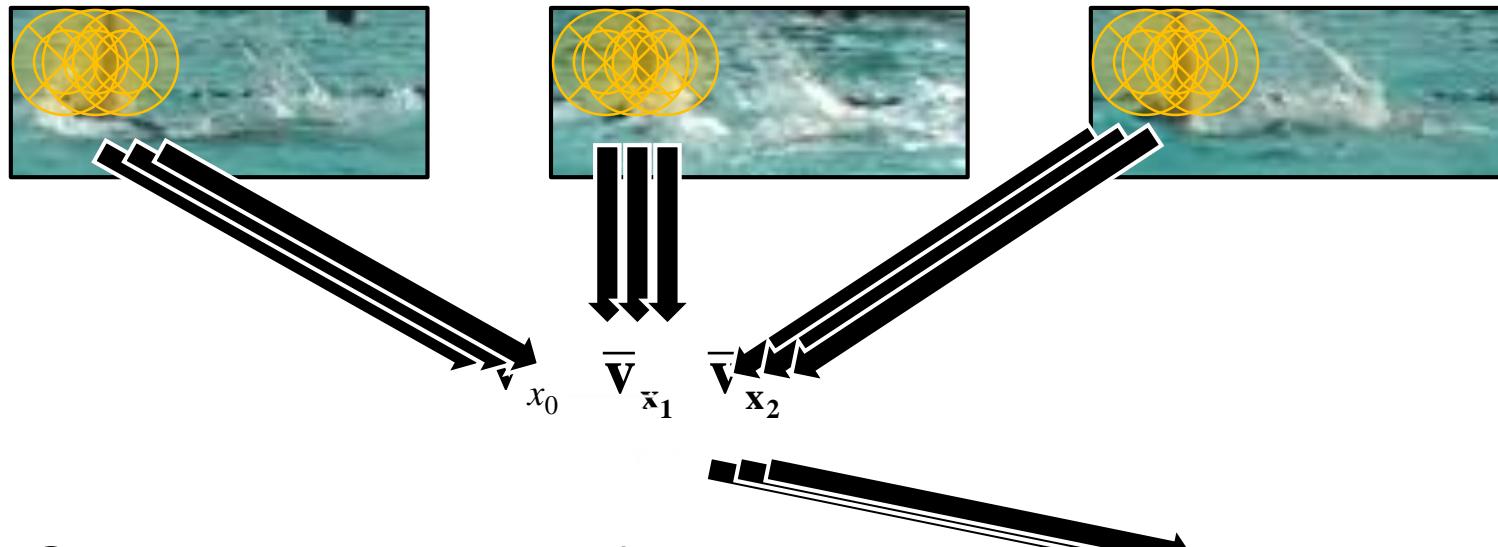


- Feature: self-similarity
- Compute similarity between center patch and each patch within surrounding region (correlation surface)
- Transform to log-polar representation
- Log-polar bins constitute feature vector



- Compute feature descriptors from template images
- Gaussian descriptor model

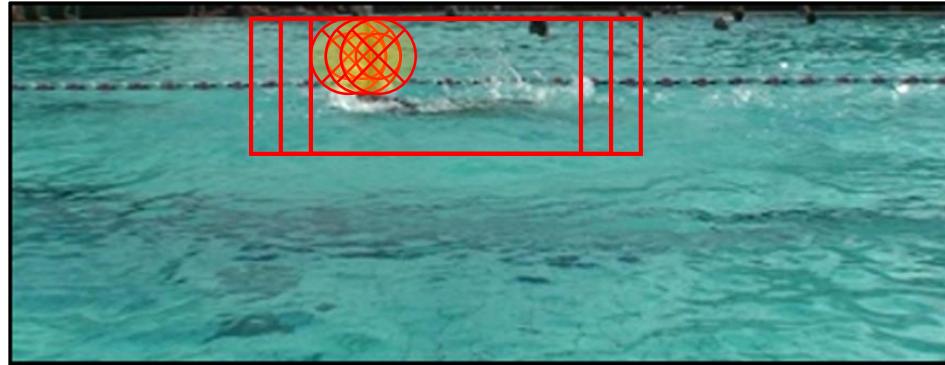
$$\bar{\mathbf{d}}_{\mathbf{x}} = N(\bar{\mathbf{v}}_{\mathbf{x}}, \Sigma_{\mathbf{x}})$$



- Ordered sequence of descriptors:

$$D = [\bar{\mathbf{d}}_{x_0}, \bar{\mathbf{d}}_{x_1}, \bar{\mathbf{d}}_{x_2}, \dots]$$

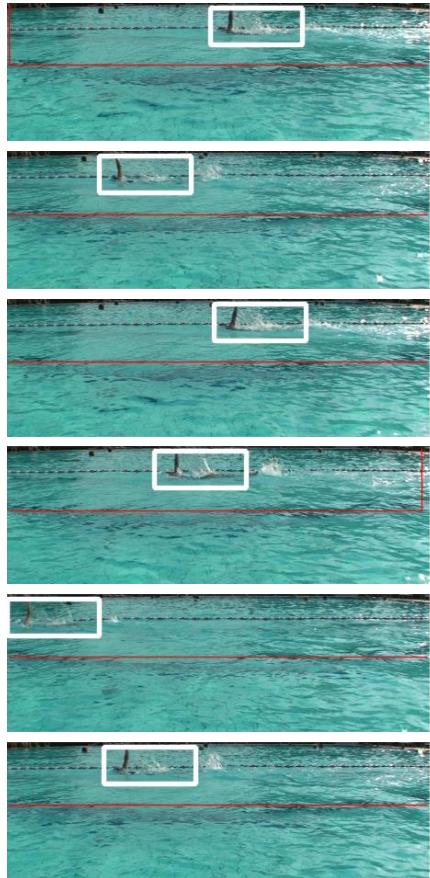
- Target video frames are scanned (over time)
 - at multiple scales
 - Using a sliding window of same size as templates



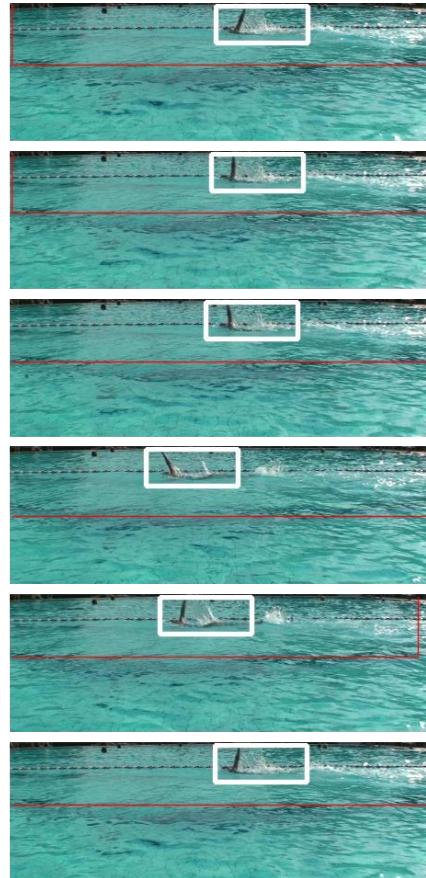
- At each window position (candidate region) a feature vector sequence is computed
- Sequences are matched to template descriptor sequence

Detection results

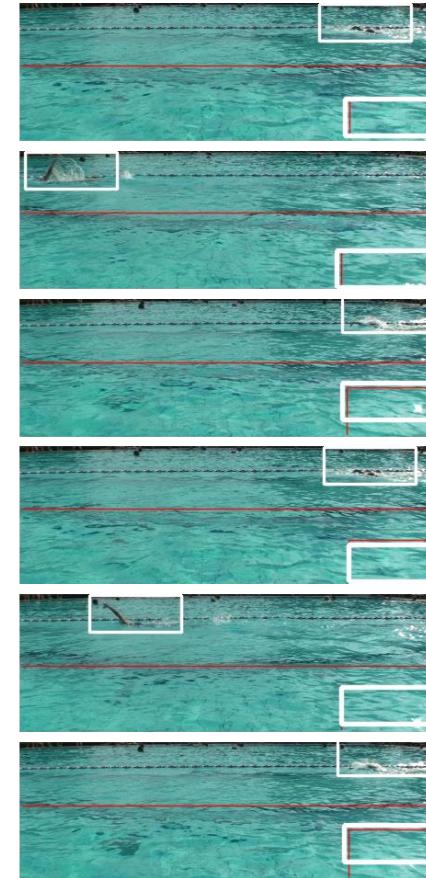
Self-Similarity



SIFT



Geometric Blur



HoG

