6. Texterkennung in Videos

Videoanalyse

Dr. Stephan Kopf

Übersicht

- Motivation
- Texterkennung in Videos
  1. Erkennung von Textregionen/Textzeilen
  2. Segmentierung einzelner Buchstaben
  3. Auswahl der Buchstabenpixel
  4. Erkennung einzelner Buchstaben (OCR)
     - Pattern matching
     - Zoning
     - Shape contexts
     - Konturprofile
     - Skelette
     - Skalenraumabbildungen
- Zusammenfassung

Motivation

- Wichtige semantische Informationen werden in Videos durch Texte übermittelt:
  - Namen der Schauspieler in Spielfilmen
  - In Nachrichtensendungen werden die neben dem Sprecher gezeigten Bilder durch einen Text beschrieben.
  - Ort oder Zeit in einem Spielfilm
  - Fragen bei Quizshows
  - Namen/Beruf der Teilnehmer einer Diskussionsrunde
  - Titel eines Films

Erkennung von Textregionen (I)

Ablauf

1. Suche Blöcke mit starken Kanten
2. Fasse benachbarte Blöcke zu Textregionen zusammen
3. Verwende horizontale Projektionsprofile zur Erkennung einzelner Textzeilen

Suche Blöcke mit starken Kanten (Summe Kantenstärke pro Block > T)

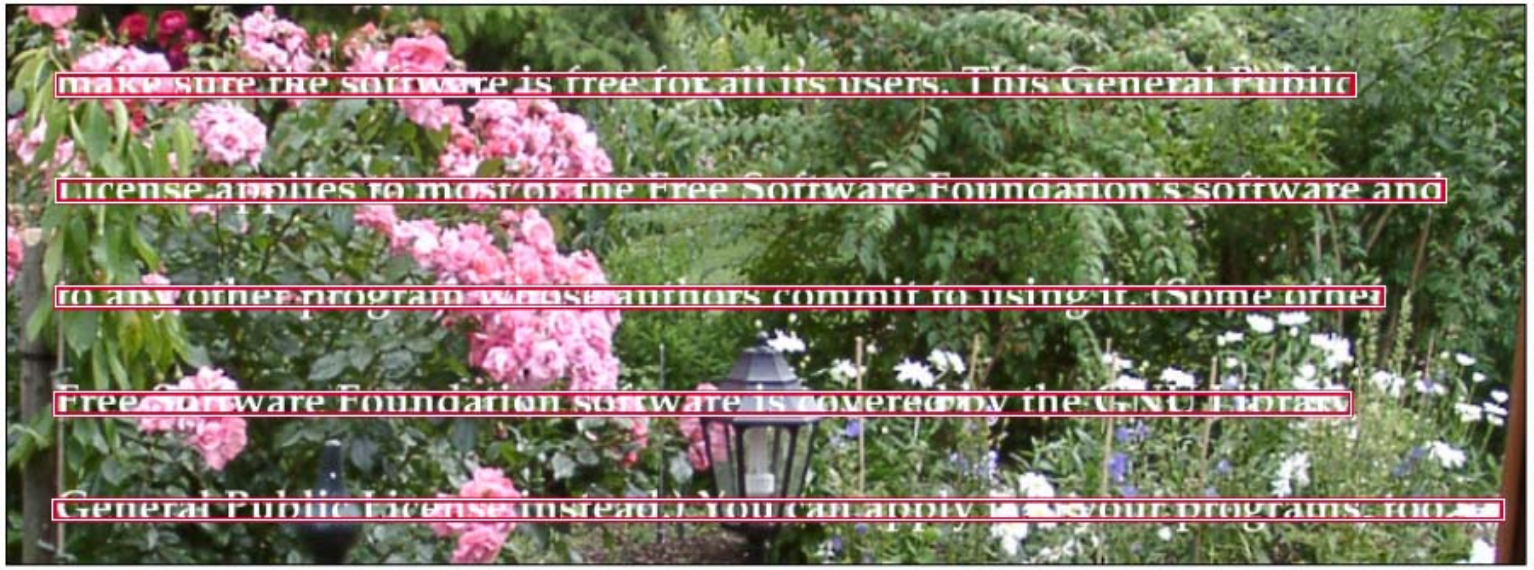
Erkennung von Textregionen (II)

Zusammenfassung von Blöcken

Erkennung von Textregionen (III)

Horizontale Projektionsprofile

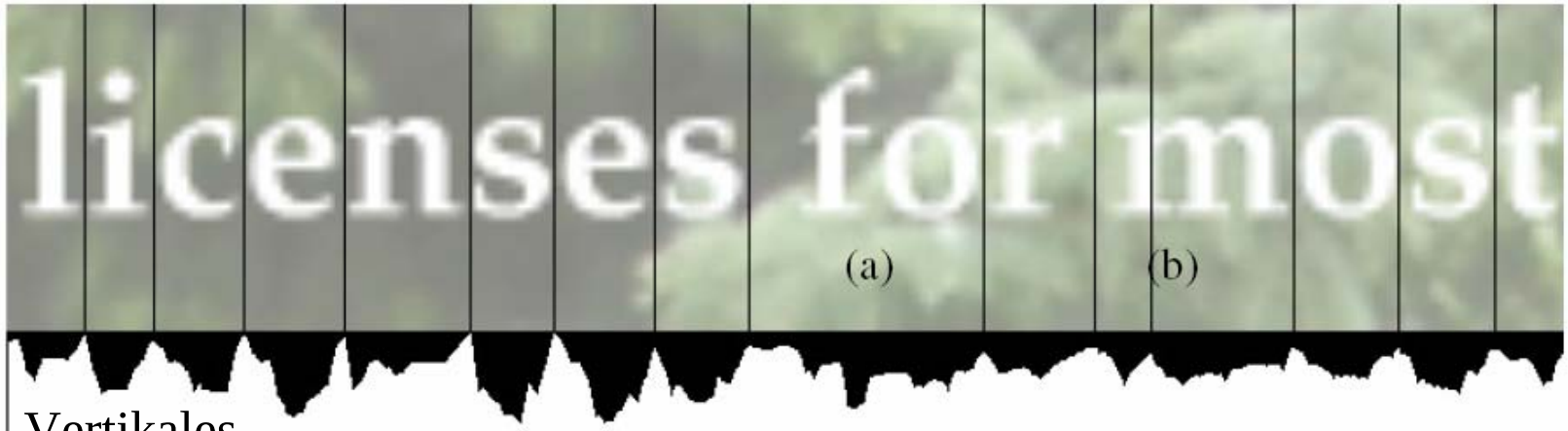
Erkannte Textzeilen



Segmentierung von Buchstaben (I)

Erster Ansatz

Analysiere vertikale Projektionsprofile um *Trenner* zwischen Buchstaben zu erkennen.



Problem: geteilte oder verbundene Buchstaben

Segmentierung von Buchstaben (II)

Neuer Ansatz

- Analysiere *kürzeste Pfade* und nutze diese als Trenner zwischen den Buchstaben.

- Suche für jede Spalte von der obersten Zeile einen kürzesten Pfad bis zur untersten Zeile.
- Die Kosten für den Pfad sollen minimal sein.
- Die Kosten sind als Summe der absoluten Differenzen zwischen benachbarten Pfadpixeln definiert.

Segmentierung von Buchstaben (III)

Kürzester-Pfade-Algorithmus

- Der *Kürzeste-Pfade-Algorithmus* von *Dijkstra* wird verwendet, um den Pfad mit den geringsten Kosten zu finden.
- Ziel: Berechnung eines kürzesten Pfades zwischen einem Startknoten (Pixel in oberster Zeile) und einem beliebigen Knoten in einem kantengewichteten Graphen (beliebiges Pixel in unterster Textzeile).
- Initialisiere: Setze Entfernung für alle Knoten auf unendlich.
- Betrachte Knoten *u* mit geringstem Abstand zum Startknoten
  - Falls Knoten u in unterster Zeile liegt: kürzester Pfad gefunden.
  - Sonst:
    - Betrachte alle erreichbaren Pixel, d.h. die drei benachbarten (in der Zeile darunter liegenden) Pixel.
    - Prüfe, ob der Weg zu jedem Pixel über das aktuelle Pixel günstiger ist als der bisher bekannte Weg zu diesem Pixel.
    - Setze neuen Pfad, falls dieser günstiger ist.

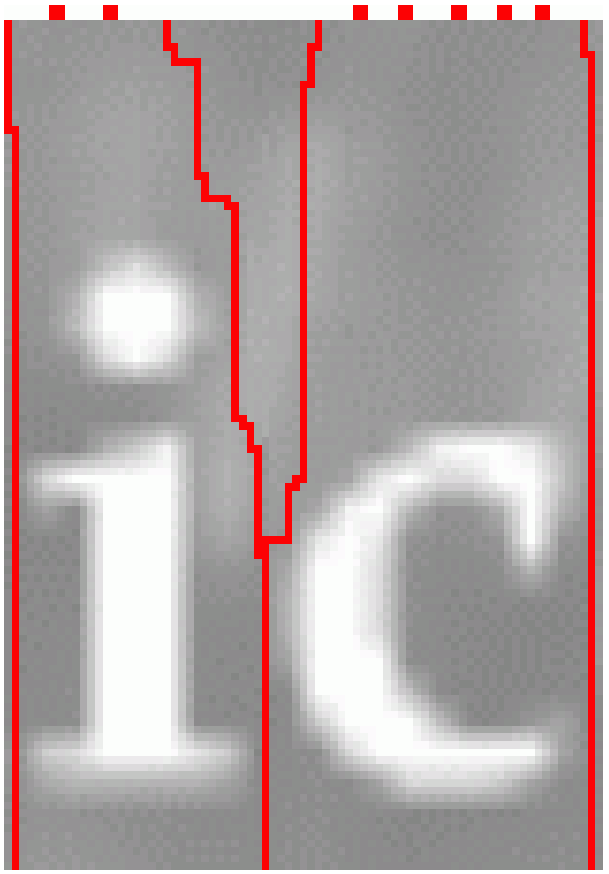
Optimierung des Kürzesten-Pfade-Algorithmus

Problem

- Der Aufwand zur Berechnung des kürzesten Pfads für jedes Pixel ist sehr aufwändig.

Optimierung

1. Initialisiere mögliche Startpixel.
2. Markiere linkes und rechtes Pixel.
3. Berechne Pfad für markierte Pixel.
4. Entferne Startpixel zwischen zwei Pfaden, falls diese zusammen laufen.
5. Wähle nächstes Startpixel.
6. Wiederhole mit 3.



Auswahl der Buchstabenpixel (I)

Identifiziere Buchstabenpixel mit Hilfe eines **modifizierten**

Region-Merging-Algorithmus

- Berechne das Histogramm einer Textregion und identifiziere ein bis zwei dominante Farben.
  Annahme: Eine dieser Farben ist die Textfarbe.
- Identifiziere Regionen mit einem Region-Growing-Algorithmus.
- Jede Region kann einen der drei Zustände annehmen: *Text*, *Hintergrund*, *undefiniert*. Alle Regionen sind zunächst *undefiniert*.
- Setze alle Regionen mit Textfarbe zu *Textregionen*.
- Undefinierte Regionen am oberen oder unteren Rand der Textzeile werden als *Hintergrund* definiert.

Auswahl der Buchstabenpixel (II)

6. Berechne Distanz $\boldsymbol{D_{i,j}}$ zwischen einer undefinierten Region ***i*** und einer bekannten Region ***j*** (Text oder Hintergrund) anhand der Farben $\boldsymbol{C_i}$ und dem Schwerpunkt einer Region $\boldsymbol{G_i}$:

$$D_{i,j} = |C_i - C_j| + |G_i - G_j|.$$

7. Wähle minimale Distanz $\boldsymbol{D_{i,j}}$ und definiere Region als *Text* oder *Hintergrund*.
8. Wiederhole mit Schritt 6 bis alle undefinierten Regionen bekannt sind.

Erkennung einzelner Buchstaben (I)

Erkennung einzelner Buchstaben (OCR)

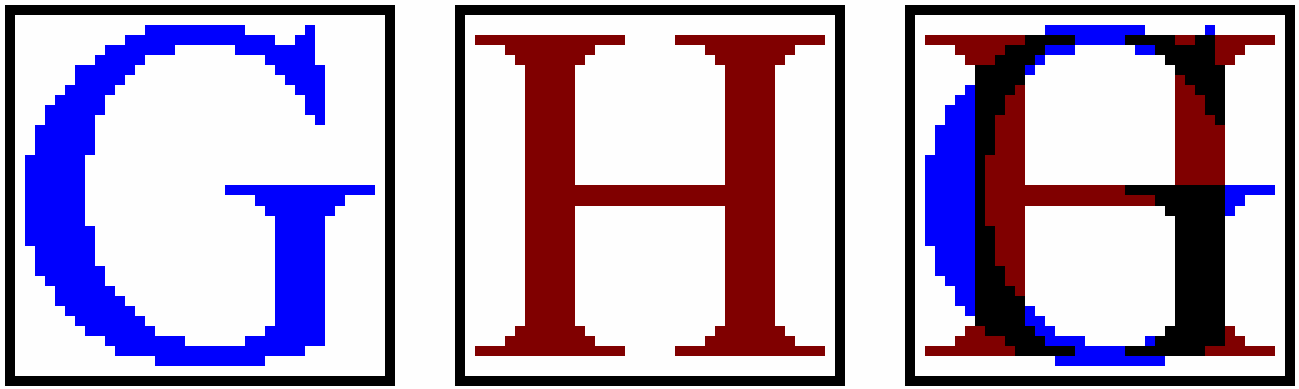
- Pattern matching
- Zoning
- Shape contexts
- Konturprofile
- Skelette
- Skalenraumabbildungen

Erkennung einzelner Buchstaben (II)

Pattern matching

Berechne die Differenz zwischen zwei Binärbildern:

$$D_{Q,J} = \frac{1}{n_x \cdot n_y} \cdot \sum_{x=1}^{n_x} \sum_{y=1}^{n_y} \begin{cases} 0 & \text{falls} \quad Q_{x,y} = J_{x,y}, \\ 1 & \text{sonst.} \end{cases}$$

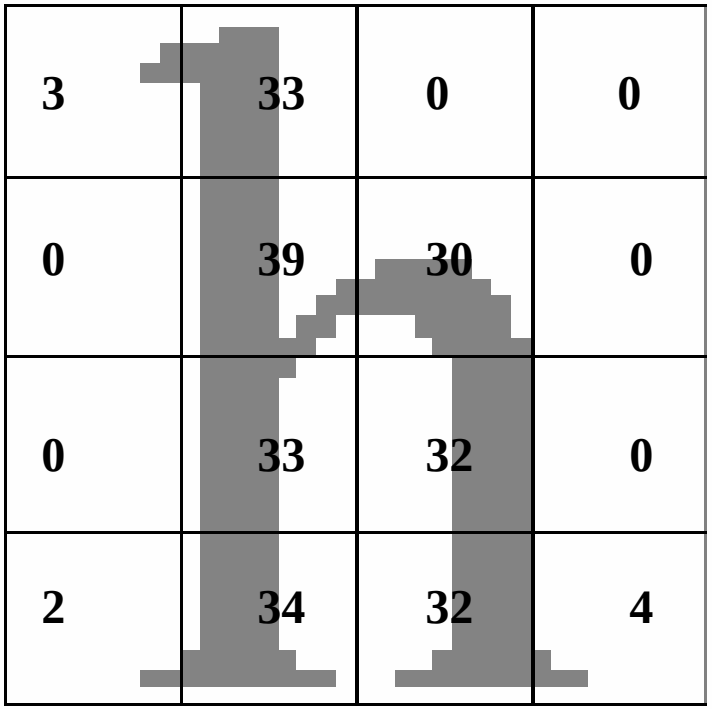


18 % der Pixel unterscheiden sich

Erkennung einzelner Buchstaben (III)

Zoning

- Definiere ein Gitter mit N x M Zellen.
- Zähle die Anzahl der Textpixel in jedes Zelle.
- Vergleiche zwei Vektoren, die durch die Anzahl der Textpixel jeder Zelle definiert sind.

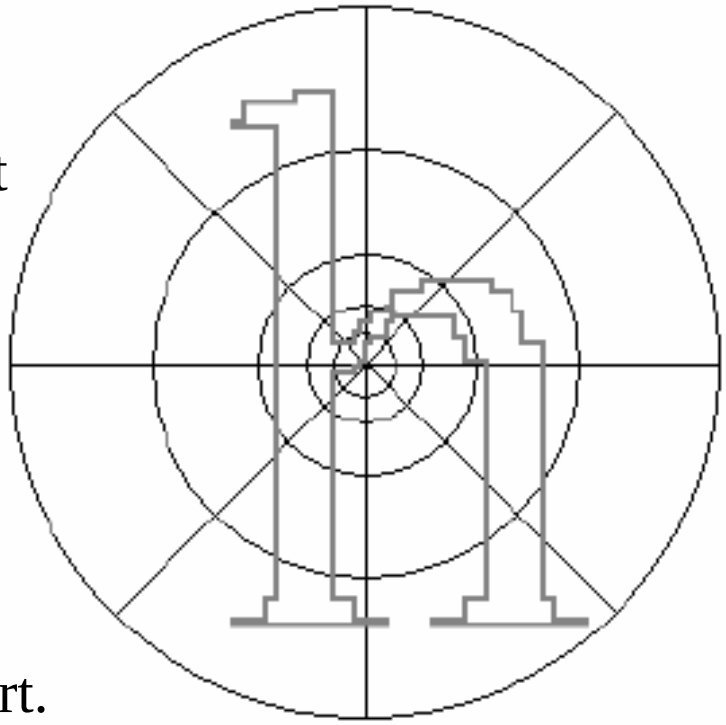


Vektor: (3,33,0,0,0,39,30,0,0,33,32,0,2,34,32,4)

Erkennung einzelner Buchstaben (IV)

Shape contexts

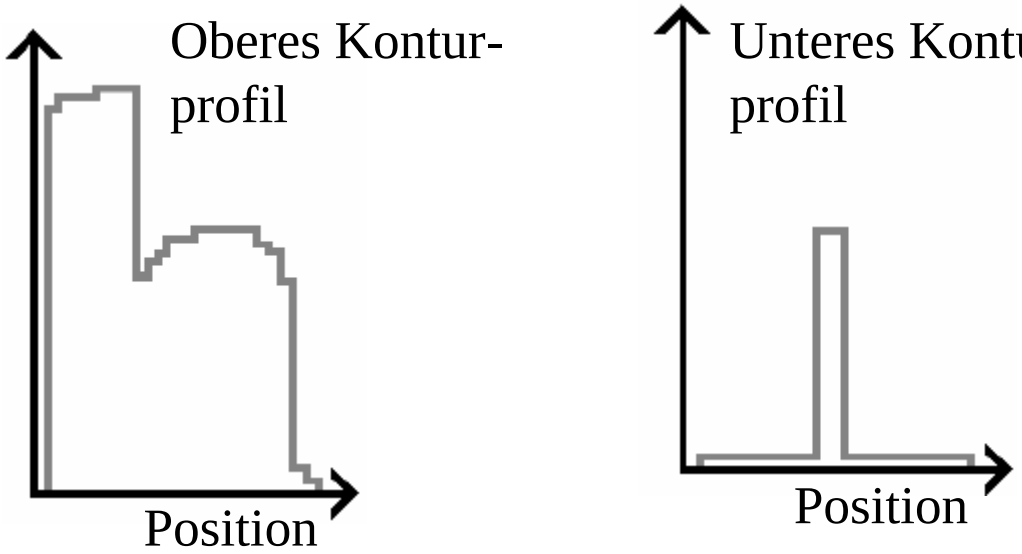
- Der Shape-Context-Algorithmus ist ein spezieller Zoning-Algorithmus.
- Ein rundes Raster wird zur Definition der Zellen verwendet.
- Ein Konturpixel definiert den Mittelpunkt des Rasters.
- Die Anzahl der ***Konturpixel*** (nicht der Textpixel) in jeder Zelle definieren den Merkmalsvektor.
- Als Referenzbuchstaben werden Merkmalsvektoren für ***jedes*** einzelne Konturpixel eines Buchstabens gespeichert.



Erkennung einzelner Buchstaben (V)

Konturprofile (contour profiles)

- Horizontales Profil: Analysiere obere und untere Konturpixel.
- Vertikales Profil : Analysiere linke und rechte Konturpixel.
- Aggregiere diese Profile in einen Vektor.



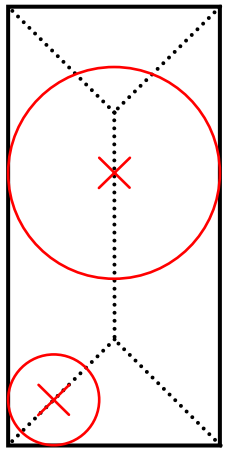
Erkennung einzelner Buchstaben (VI)

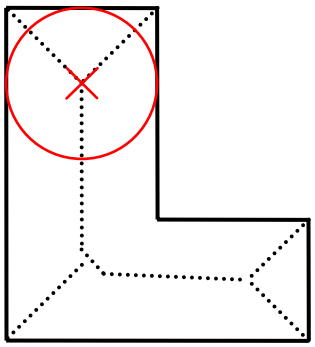
Skelette

- Idee: Ermittle die Struktur einer Region
- Vorgehen: Trage die Region iterativ ab
- Im Jahr 1967 wurde das Verfahren der *Medial-Axis*-Transformation vorgestellt:

  R: Region, B: Rand

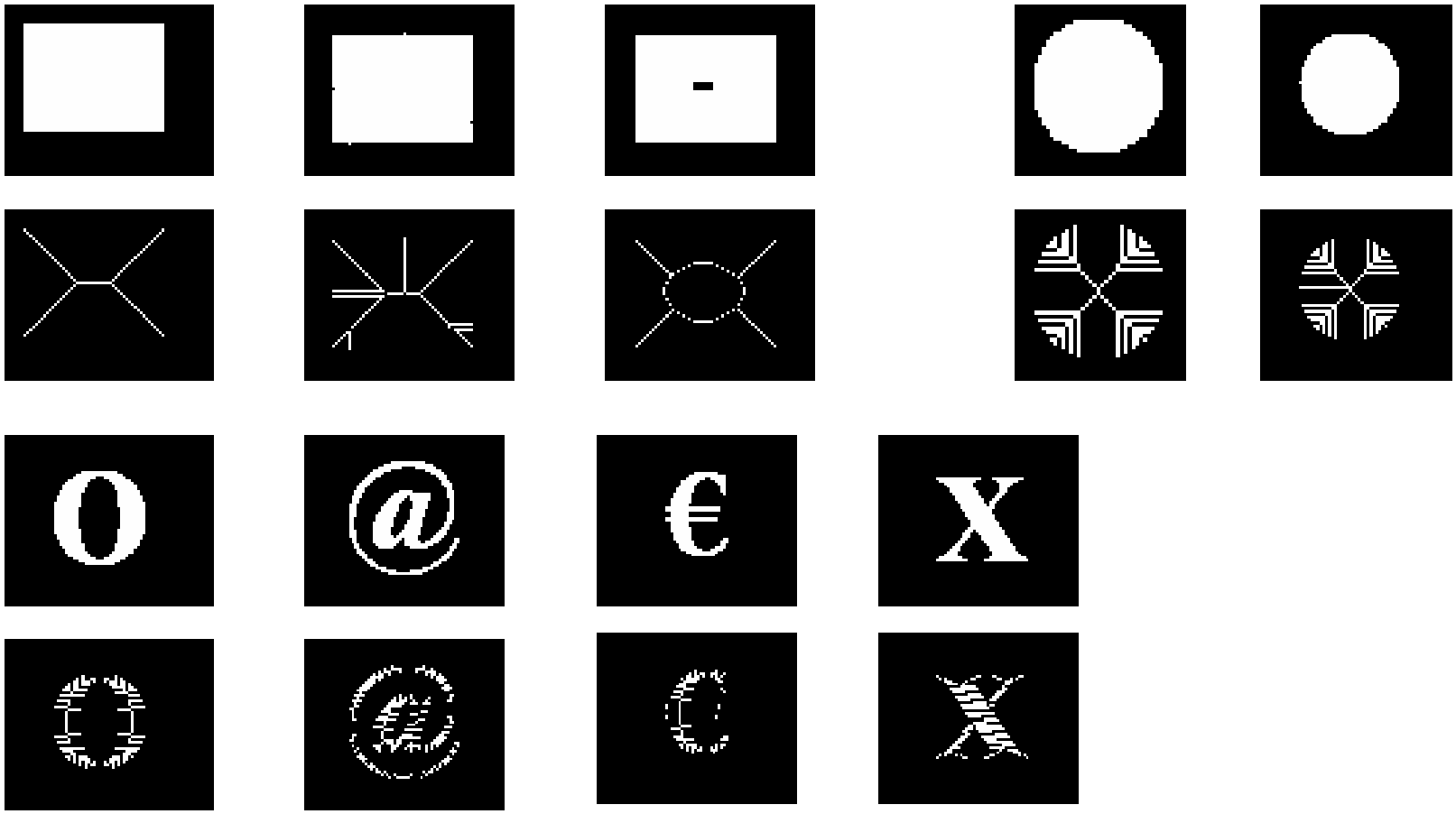
  - Für alle p $\in$ R: Suche nächsten Nachbarn (z.B. mittels City-Block-Distanz) in B.
  - Falls (Anzahl Nachbarn > 1): Pixel ist mediale Achse (Skelett) von R





Erkennung einzelner Buchstaben (VII)

Beispiel für Skelette bei Verwendung der City-Block-Distanz



Erkennung einzelner Buchstaben (VIII)

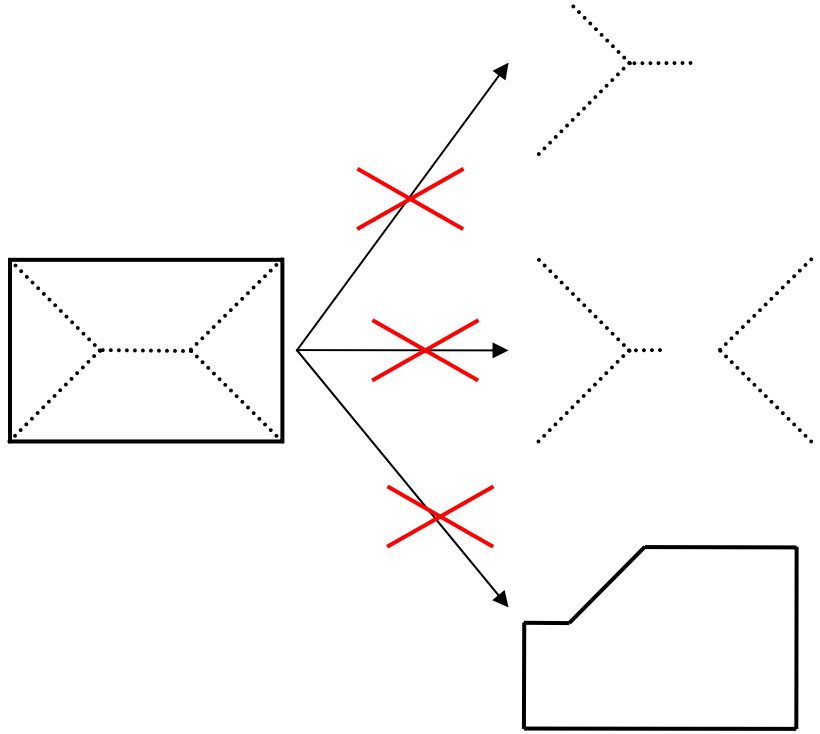
Probleme bei der Erzeugung von Skeletten

- Minimale Änderungen der Kontur ergeben sehr unterschiedliche Skelette.
- Der Rechenaufwand ist sehr hoch, da für jedes Pixel die Distanz zu jedem anderen Pixel berechnet werden muss.
- **Optimierung**:
  Trage Regionen ab (entferne Randpixel), so dass:
  - Endpunkte des Skeletts möglichst wenig abgetragen werden,
  - eine Region nicht in zwei Regionen unterteilt wird,
  - alle Regionen des Objektes gleichmäßig stark abgetragen werden (eine Region soll nicht übertrieben stark abgetragen werden).

Erkennung einzelner Buchstaben (IX)

Bedingungen

- Endpunkte des Skeletts sollen möglichst wenig abgetragen werden,
- eine Region soll nicht in zwei Regionen unterteilt werden,
- alle Regionen des Objektes sollen gleichmäßig stark abgetragen werden (eine Region soll nicht übertrieben stark abgetragen werden).



Erkennung einzelner Buchstaben (X)

***Thinning*-Algorithmus zur Erzeugung von Skeletten**

- Gegeben: Binärbild (Hintergrund=0, Objekt=1)
- 8-Pixel-Nachbarschaft
  aktuelles Pixel: p_1

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

1. Betrachte jedes Randpixel und markiere Pixel falls alle Bedingungen erfüllt sind:
   - $3 \leq N(p_1) \leq 6$
   - $S(p_1) = 1$
   - $p_2 * p_4 * p_6 = 0$
   - $p_4 * p_6 * p_8 = 0$
2. Lösche markierte Pixel

$N(p_1)$: Anzahl der Objektpixel $p_2 \ldots p_9$ in der Umgebung von p_1

$S(p_1)$: Anzahl der Übergänge von Hintergrundpixeln nach Objektpixeln beim Ablaufen von $p_2, p_3 \ldots p_9, p_2$

Erkennung einzelner Buchstaben (XI)

***Thinning*-Algorithmus zur Erzeugung von Skeletten**

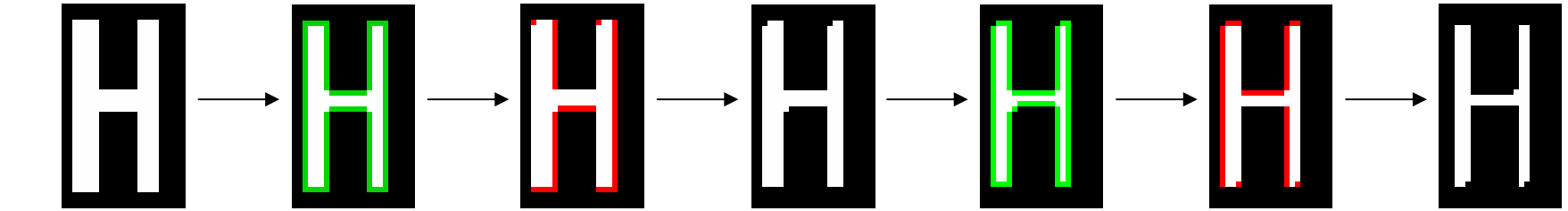
3. Betrachte jedes Randpixel und markiere Pixel falls alle Bedingungen erfüllt sind:
   - $3 \leq N(p_1) \leq 6$
   - $S(p_1) = 1$
   - $p_2 * p_4 * p_8 = 0$
   - $p_2 * p_6 * p_8 = 0$
4. Lösche markierte Pixel
5. Gehe zu 1. falls mindestens ein Pixel gelöscht wurde.

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

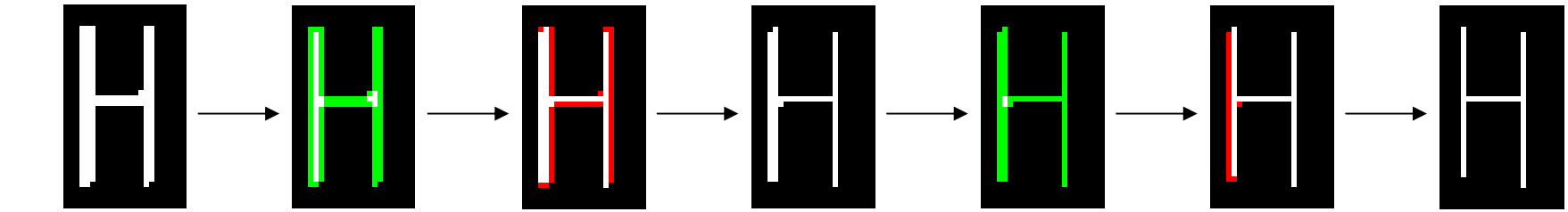
Erkennung einzelner Buchstaben (XII)

***Thinning*-Algorithmus zur Erzeugung von Skeletten**

Iteration 1

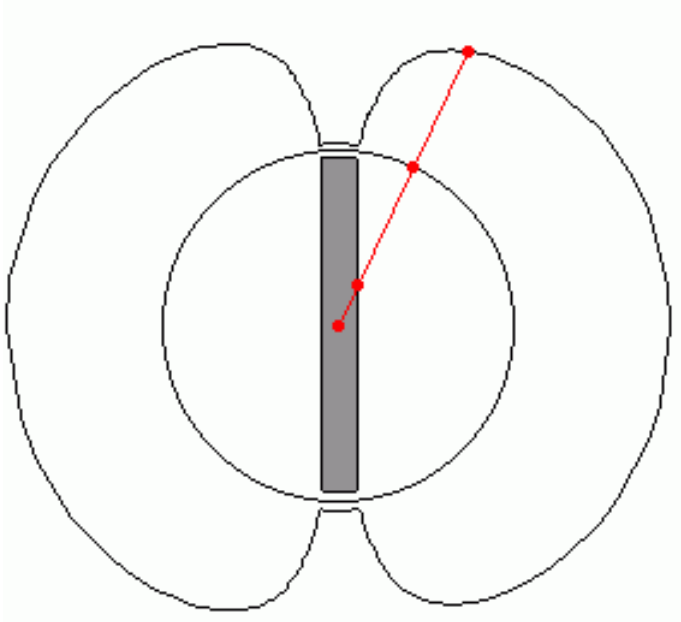


Iteration 2



Erkennung einzelner Buchstaben (XIII)

Texterkennung mit Skalenraumabbildungen



Ablauf:

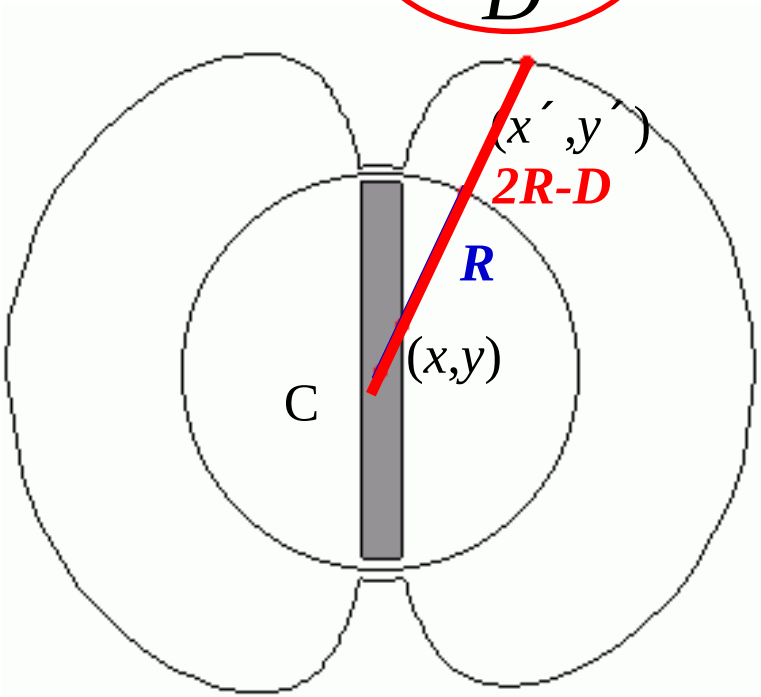
- Identifiziere Schwerpunkt
- Lege Kreis um Buchstaben
- Spiegele Konturpixel des Buchstabens an der Kreislinie

→ Stark konvex gekrümmte Regionen werden zu konkaven Regionen.

Erkennung einzelner Buchstaben (XIV)

Texterkennung mit Skalenraumabbildungen

$$x' = \frac{2R - D}{D}(x - C_x) + C_x$$

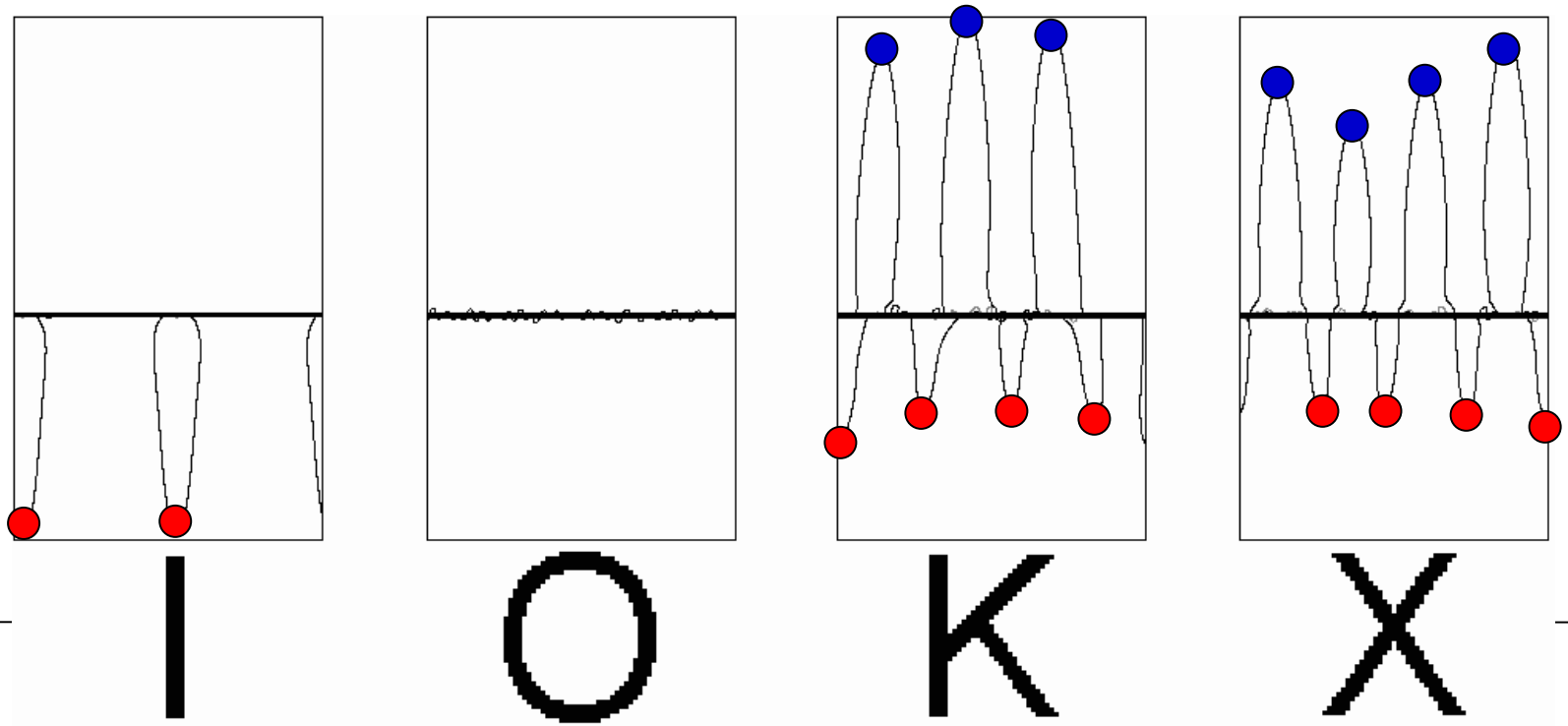


(x,y)	Konturpixel
$(x´,y´)$	Gespiegeltes Konturpixel
$C=(C_x,C_y)$	Schwerpunkt
R	Radius des Kreises
D	Entfernung zwischen Konturpixel und C
$2R - D$	Entfernung zwischen $(x´,y´)$ und C

Erkennung einzelner Buchstaben (XV)

Texterkennung mit Skalenraumabbildungen

- Berechne normale Skalenraumabbildung
- Berechne Skalenraumabbildung für gespiegelte Kontur



Experimentelle Ergebnisse (I)

Datenbank

- Buchstaben von vier Schriftarten wurden verwendet.
- Die Skalenraumabbildungen durften maximal ~20 Grad gedreht werden, um kursive Zeichen zu erkennen.

Herausforderungen

- Texterkennung bei Segmentierungsfehlern:

Experimentelle Ergebnisse (II)

Segmentierungsfehler	Projektionsprofile	Kürzeste-Pfade
Unterteilte Buchstaben	9.9 %	3.8 %
Verbundene Buchstaben	7.5 %	5.4 %
Segmentierungsfehler	17.4 %	9.2 %

Texterkennungsverfahren	Erkennungsergebnisse
Pattern Matching	69 %
Zoning	64 %
Konturprofile	71 %
Skalenraumabbildungen	76 %
Kommerzielle OCR-Software (Scanner)	75 %

Experimentelle Ergebnisse (III)

Fragen ?