Algorithmus zum Graphen-Matching

und

Anwendung zur inhaltsbasierten Bildersuche

von Martin Junghans
auf Grundlage einer Arbeit von Adel Hlaoui, Shengrui Wang [HW02b]

Gliederung

1. Einführung

2. Algorithmus
   - Beschreibung
   - Beispiel
   - Laufzeit

3. Anwendung des Algorithmus

1. Einführung

- Ziel ist Finden und Kategorisieren von Bildern, basierend auf Bildinhalt

- Graphen können den Inhalt von Bildern repräsentieren

- Um Ähnlichkeiten zwischen Graphen/Bildern zu ermitteln, wird ein effizienter Graphen-Matching-Algorithmus benötigt

Graph G

Tupel $G=(V, E, \mu, \nu)$ mit:

V Knotenmenge,

$E \subseteq V \times V$ Kantenmenge,

$\mu : V \to \mathbb{R}$ Knotengewichtung und

$\nu : E \to \mathbb{R}$ Kantengewichtung

Graphen-Matching

- Berechnung der Ähnlichkeit zweier Graphen
- Gesucht wird ein (Teil-) Graph-Isomorphismus

Graph-Isomorphismus

Ist jede Bijektion $b: V_1 \to V_2$, die einen Graph G_1 in einen Graph G_2 transformiert und umgekehrt, d.h. Verbindungen von Knoten durch Kanten bleiben erhalten.

Teilgraph-Isomorphismus („Graph-Matching“)

Man wählt einen Teilgraphen $G_3 \subseteq G_2$ so, dass $|V_1| = |V_3|$ und ein Graph-Isomorphismus zu G_1, G_3 existiert.

Matching-Fehler

Benötigt eine Abstandsfunktion $d: (V_1 \times V_2) \cup (E_1 \times E_2) \to \mathbb{R}$, die Unterschiede von aufeinander abgebildeten Knoten und der entsprechend aufeinander abgebildeten Kanten bzgl. des Graph-Matchings bestimmt.

Im folgenden Beispiel bestimmt sich der Abstand wie folgt:

- Abstand/Unterschied von aufeinander abgebildeten Knoten

  $\forall i \in V_1 \ \forall j \in V_2 : b(i) = j \Rightarrow$

  $d(\mu_1(i), \mu_2(j)) = |\mu_1(i) - \mu_2(j)|$

- Abstand/Unterschied von aufeinander abgebildeten Kanten

  $\forall (i, i') \in E_1 \ \forall (j, j') \in E_2 : b(i) = j \wedge b(i') = j' \Rightarrow$

  $d(\nu_1((i, i')), \nu_2((j, j'))) = |\nu_1((i, i')) - \nu_2((j, j'))|$

2. Neuer Graphen-Matching-Algorithmus [HW02a]

gegeben: 2 Graphen G_1, G_2

gesucht: Graphen-Isomorphismus,
mit geringstem Matching-Fehler

Annahme: gute Knotenabbildungen bilden ähnliche Knoten aufeinander ab

<u>bisherige Graphen-Matching-Algorithmen</u>

- Kombinatorischen Explosion der zu prüfenden Matchings
- Nur anwendbar bis ca. 10 Knoten

<u>Grundidee</u>

- Matching-Prozess wird in Phasen unterteilt
- iteratives Erkunden vielversprechender Knotenabbildungen und Auswahl der besten Abbildung nach jeder Iteration (durch minimalen Matching-Fehler)
- geringe Anzahl von Phasen beschränkt den Suchraum
- findet effizient sehr gute (meist optimale) Graphen-Matchings

Neuer Graphen-Matching-Algorithmus

- 1. Phase: beste Abbildung bzgl. des Matching-Fehlers der Knoten wählen
- 2. Phase: Abbildungen bestimmen, die mindestens einen Knoten mit zweit-kleinsten Matching-Fehler enthalten

  ⋮
- K. Phase: analog

- Jede Abbildung wird auf Bijektivität geprüft
- Falls Matching-Fehler minimal, wird Bijektion gespeichert
- Ergebnis ist die Bijektion mit geringsten Matching-Fehler

$n=|V_1|$, $m=|V_2|$ (o.B.d.A. sei $n \leq m$)

$n \times m$-Matrix $P=(p_{ij})$
enthält Unterschiede $p_{ij}=d(\mu_1(i),\mu_2(j))$
der Knoten $i \in V_1$ und $j \in V_2$

$n \times m$-Matrix $B=(b_{ij})$
enthält alle bisher untersuchten Knotenabbildungen,
markiert durch $b_{ij}=1$, sonst $b_{ij}=0$

$n \times m$-Hilfsmatrix B'
enthält die zu untersuchenden Abbildungen und
vermeidet dadurch mehrfache Untersuchungen

Neuer Graphen-Matching-Algorithmus

Initialisiere P mit $p_{ij} := d(\mu_1(i), \mu_2(j))$

Initialisiere B' mit $b'_{ij} := 0$ für $i = 1, \ldots, n$ und $j = 1, \ldots, m$

$SET\ Phase := 1$

$FOR\ i = 1, \ldots, n$

wähle kleinsten Wert p_{ix} aus Zeile i von P, $SET\ b'_{ix} := 1$

$CALL\ MATCH(B')$

$SET\ Phase := 2$

$WHILE\ Phase < K$

$FOR\ i = 1, \ldots, n$

$SET\ B := B'$

$FOR\ j = 1, \ldots, m\ \ SET\ b'_{ij} := 0$

wähle kleinsten Wert p_{ix} mit $b_{ix} \neq 1$ aus Zeile i

$SET\ b'_{ix} := 1$, $SET\ b_{ix} := 1$

$CALL\ MATCH(B')$

$SET\ B' := B$

IF alle Elemente $b'_{ij} \in B'$ mit 1 markiert

$THEN\ SET\ Phase := K$

$ELSE\ SET\ Phase := Phase + 1$

$MATCH(B')$:

FOR ALL bijektive Knotenabbildung in B'

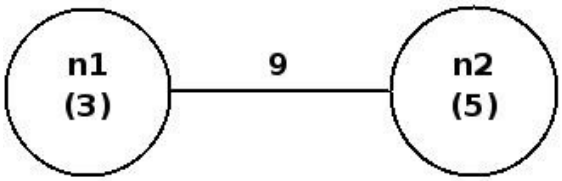
Berechne Matching-Fehler bzgl. der Knoten f_n

Berechne Matching-Fehler bzgl. entsprechender Kanten f_e

IF Matching-Fehler minimal

THEN speichere Bijektion

Beispiel



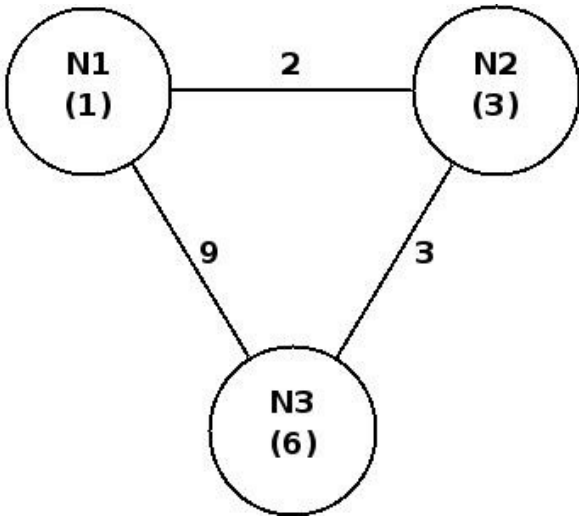


Abbildung 1: Beispiel-Graphen G_1, G_2

$$P := \begin{pmatrix} \mathbf{2} & \mathbf{0} & \mathbf{3} \\ \mathbf{4} & \mathbf{2} & \mathbf{1} \end{pmatrix}$$

$$B' := \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}$$

$Phase = 1$

$i = 1:$ $\quad B' := \begin{pmatrix} 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 \end{pmatrix}$

$i = 2:$ $\quad B' := \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & \mathbf{1} \end{pmatrix}$

$MATCH(B'):$ $\quad$ Bijektion $b_1 = \{(n1 \to N2), (n2 \to N3)\}$
$f_n = 0 + 1$
$f_e = 6$
$f = 7$ minimal, speichere b_1

$Phase = 2$

$i = 1:$ $\quad B := \begin{pmatrix} \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix}$

$j = 1 \ldots 3: B' := \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 \end{pmatrix}$

$B' := \begin{pmatrix} \mathbf{1} & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \; B := \begin{pmatrix} \mathbf{1} & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

$MATCH(B'):$ $\quad b_2 = \{(n1 \to N1), (n2 \to N3)\}$
$f_n = 2 + 1$
$f_e = 0$
$f = 3$ minimal, speichere b_2

$B' := \begin{pmatrix} \mathbf{1} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix}$

$i=2:$ $\quad B:=\begin{pmatrix}\mathbf{1} & \mathbf{1} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{1}\end{pmatrix}$

$j=1...3: B':=\begin{pmatrix}1 & 1 & 0\\ \mathbf{0} & \mathbf{0} & \mathbf{0}\end{pmatrix}$

$B':=\begin{pmatrix}1 & 1 & 0\\ 0 & \mathbf{1} & 0\end{pmatrix},\ B:=\begin{pmatrix}1 & 1 & 0\\ 0 & \mathbf{1} & 1\end{pmatrix}$

$MATCH(B'):$ $\quad b_3=\{(n1\to N1),(n2\to N2)\}$
$f_n=2+2$
$f_e=7$
$f=11$ nicht minimal

$b_4=\{(n1\to N2),(n2\to N2)\}$
keine Bijektion

$B':=\begin{pmatrix}\mathbf{1} & \mathbf{1} & \mathbf{0}\\ \mathbf{0} & \mathbf{1} & \mathbf{1}\end{pmatrix}$

Ergebnis

$b_2=\{(n1\to N1),(n2\to N3)\}$ mit Matching-Fehler $f=3$

Komplexität

$O(n^2 K^n)$ Schritte zur Berechnung eines Matchings

- $O(n^2)$ Kanten in G_1
- K^n mögliche Knotenabbildungen

Vergleich mit anderen Matching-Algorithmen

- Fehlerkorrigierender Teilgraph-Isomorphismus-Algorithmus (A*) benötigt

  $O(n^2 m)$ im best case,

  $O(n^2 m^n)$ im worst case
- neuer Algorithmus benötigt $(m/K)^n$ Schritte weniger

Experiment mit 1000 zufällig erzeugten Graphenpaaren

Graphen mit jeweils 2 bis 10 Knoten

Anzahl der Phasen K	1	2	3	4	5	A*
optimale Knotenabbildungen	609	827	940	971	1000	1000
durchschnittliche Berechnungszeit in Sekunden	2	4	6	11	16	187

Tabelle 1: Vergleich beider Algorithmen; SUN workstation, Ultra 60, zwei 450MHz CPUs, 512MB Arbeitsspeicher

3. Finden von ähnlichen Bildern mit neuem Graphen-Matching-Algorithmus [HW02b]

Künstliche Datenbank mit Bildern

Jedes Bild besteht aus Objekten mit regelmäßigen Formen (Rechtecke, Quadrate, Dreiecke)

Zufällige Anzahl, Form, Farbe, Größe, Position der Objekte

Suchstrategie

gegeben: 1 Anfrage-Bild

gesucht: ähnliche Bilder aus der Datenbank (geringer Matching-Fehler)

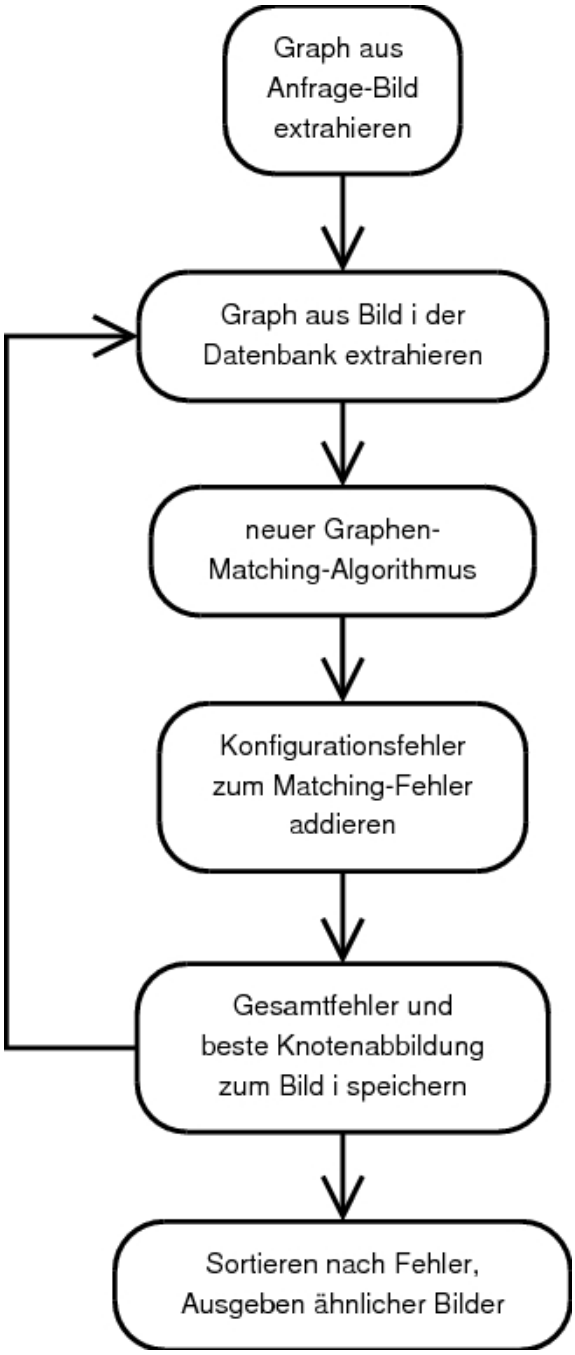


Abbildung 2: Flussdiagramm der Suchstrategie

Darstellung des Bildinhaltes durch Graphen (Schritt 1 und 2)

- Knoten stellen Objekte dar
  Eigenschaften: Form, Größe, Farbe

- Kanten repräsentieren Beziehungen zwischen Objekten
  Eigenschaften: Abstand, relative Position

Anpassung des Matching-Fehlers (Schritt 3)

- Knotenfehler f_n muss für jeden Knoten $v \in V_1$ durch mehrere Eigenschaften gewichtet werden:

$$\begin{aligned} f_n(v) = & c_{Form} \cdot e_{Form}(Form_{Anfrage}, Form_{DB}) \\ & + c_{Größe} \cdot e_{Größe}(Größe_{Anfrage}, Größe_{DB}) \\ & + c_{Farbe} \cdot e_{Farbe}(Farbe_{Anfrage}, Farbe_{DB}) \end{aligned}$$

- Kantenfehler für jede Kante $\epsilon \in E_1$ analog:

$$\begin{aligned} f_e(\epsilon) = & c_{Position} \cdot e_{Position}(Position_{Anfrage}, Position_{DB}) \\ & + c_{Abstand} \cdot e_{Abstand}(Abstand_{Anfrage}, Abstand_{DB}) \end{aligned}$$

- Koeffizienten zur Gewichtung bieten dem Nutzer Möglichkeiten zur Anpassung

$$c_{Form}, c_{Größe}, c_{Farbe}, c_{Position}, c_{Abstand}$$

$$e_{Form}(Form_{Anfrage}, Form_{DB}) = \begin{cases} 0 & falls\ Form_{Anfrage} = Form_{DB} \\ 1 & sonst \end{cases}$$

$$e_{Größe}(Größe_{Anfrage}, Größe_{DB}) = \frac{|Größe_{Anfrage} - Größe_{DB}|}{Größe_{Anfrage} + Größe_{DB}}$$

$$e_{Farbe}(Farbe_{Anfrage}, Farbe_{DB}) = \sqrt{\begin{array}{l}(Farbe \cdot L_{Anfrage} - Farbe \cdot L_{DB})^2 \\ +(Farbe \cdot U_{Anfrage} - Farbe \cdot U_{DB})^2 \\ +(Farbe \cdot V_{Anfrage} - Farbe \cdot V_{DB})^2\end{array}}$$

unter Verwendung des CIE-L*u*v-Farbraums

$$e_{Position}(Position_{Anfrage}, Position_{DB}) = \begin{cases} 0 & falls\ Pos._{Anfrage} = Pos._{DB} \\ 1 & sonst \end{cases}$$

$$e_{Abstand}(Abstand_{Anfrage}, Abstand_{DB}) = \frac{|Abstand_{Anfrage} - Abstand_{DB}|}{Abstand_{Anfrage} + Abstand_{DB}}$$

Konfigurationsfehler (Schritt 4)

ähnliche Bilder mit ungleicher Anzahl von Objekten (Knoten) oder Relationen über Objekten (Kanten)

$$f_{Konfig.} = c_{Konfig.} \cdot \left(\left| |V_{Anfrage}| - |V_{DB}| \right| + \left| |E_{Anfrage}| - |E_{DB}| \right| \right)$$

Gesamter Matching-Fehler

$$f = \sum_{v \in V_1} f_n(v) + \sum_{\epsilon \in E_1} f_e(\epsilon) + f_{Konfig.}$$

Experimentelle Ergebnisse

- Algorithmus findet tatsächlich ähnliche Bilder zum Bild der Anfrage
- Suchstrategie kann für verschiedene Anforderungen genutzt werden

Vorgehen

- Datenbank besteht aus 1000 Bildern
- Bilder bestehen aus je 2 bis 9 Objekten
- Bild der Anfrage ist nicht in der Datenbank enthalten

Suchstrategie bzgl. Formen

gesucht: - Bilder mit 3 Objekten

- Bilder sollen aus 2 Dreiecken und 1 Quadrat bestehen

Koeffizienten sind dementsprechend:

$c_{Form}=1$, $c_{Konfig.}=1$

$c_{Größe}=0$, $c_{Farbe}=0$, $c_{Position}=0$, $c_{Abstand}=0$

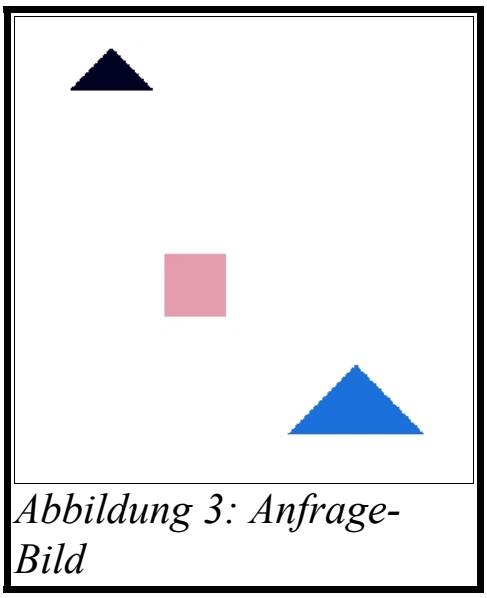


Abbildung 3: Anfrage-Bild

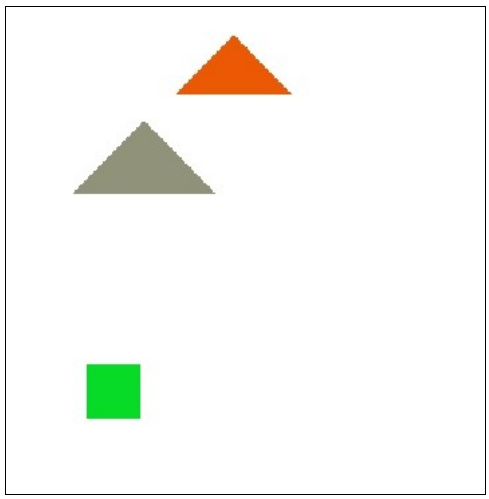


*Abbildung 4: Bild aus DB, **f=0**, alle Objekte haben die gleiche Form*

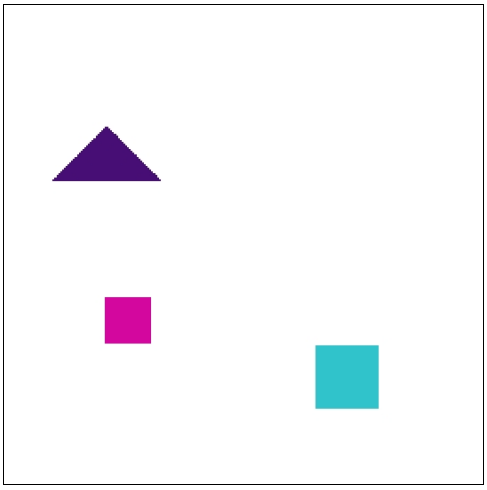


*Abbildung 5: Bild aus DB, **f=1**, nur 2 Objekte haben die gleiche Form*

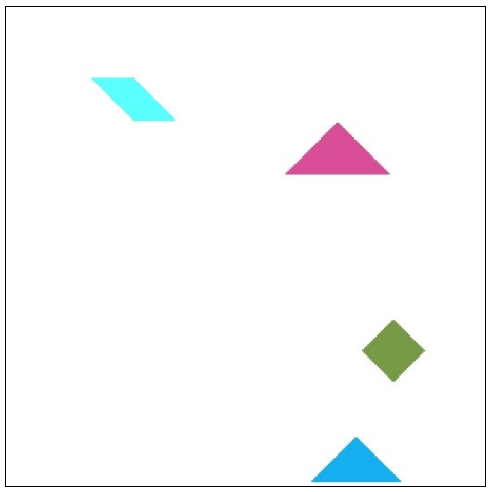


*Abbildung 6: Bild aus DB, **f=5**, ein Objekt mehr als im Anfrage-Bild, nur 2 Objekte haben die gleiche Form*

Suchstrategie bzgl. Formen und relativer Positionen

Anfrage-Bild unverändert

$c_{Form}=0{,}5$, $c_{Position}=0{,}5$, $c_{Konfig.}=1$

$c_{Größe}=0$, $c_{Farbe}=0$, $c_{Abstand}=0$

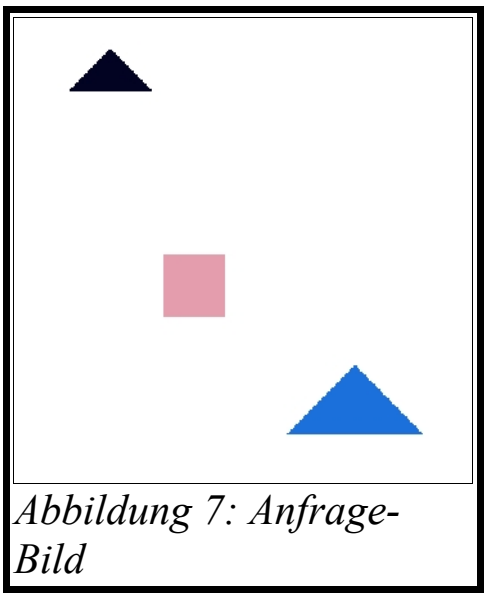


Abbildung 7: Anfrage-Bild

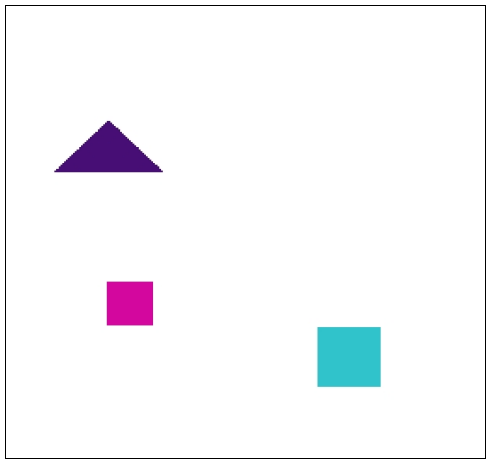


*Abbildung 8: Bild aus DB, **f=1**, Fehler entsteht aus zwei Gründen: Quadrat anstatt eines Dreiecks und die relative Position beider Quadrate*

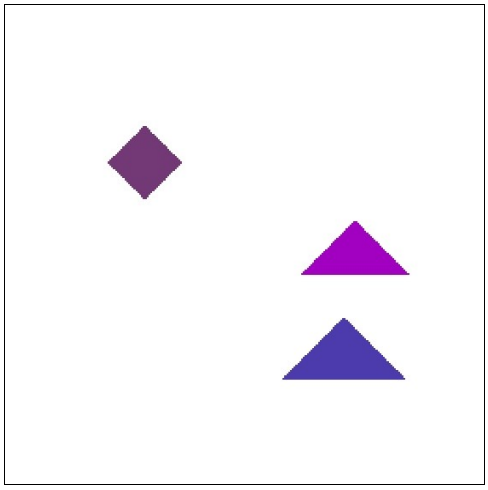


*Abbildung 9: Bild aus DB, **f=1***

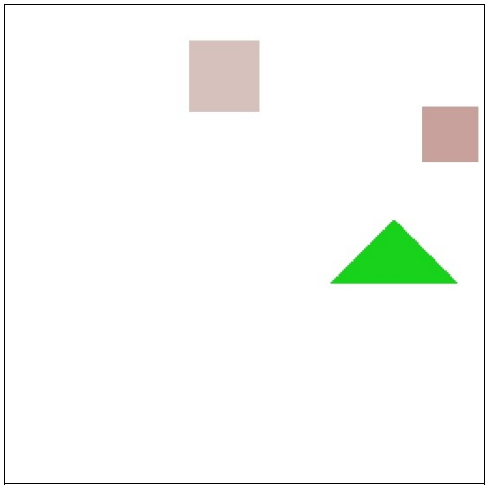


*Abbildung 10: Bild aus DB, **f=1,5***

Zusammenfassung

- neuer Graphen-Matching-Algorithmus
- Suchprozess in K Phasen unterteilt
- die wahrscheinlich erfolgreichen Isomorphismen werden in den ersten Phasen untersucht
- in wenigen Phasen lassen sich gute (meist optimale) Abbildungen finden
- Laufzeit kann wesentlich kleiner sein als mit bisherigen Algorithmen
- Algorithmus kann zur inhaltsbasierten Bildsuche verwendet werden

Literatur

[HW02a] Adel Hlaoui and Sengrui Wang. A New Algorithm for Inexact Graph Matching. In *ICPR (4)*, pages 180-183, 2002.
(http://www.dmi.usherb.ca/~hlaoui/icpr2002.pdf vom 22.11.2006)

[HW02b] Adel Hlaoui and Shengrui Wang. A new algorithm for graph matching with application to content-based image retrieval. In Terry Caelli, Adnan Amin, Robert P. W. Duin, Mohamed S. Kamel, and Dick de Ridder, editors, *SSPR/SPR*, volume 2396 of *Lecture Notes in Computer Science*, pages 291–300. Springer, 2002.
(http://www.dmi.usherb.ca/~hlaoui/sspr2002.pdf vom 22.11.2006)

[SW] Shengrui Wang. Contribution en appariement de graphe pour la reconnaissance de formes structurelles. Un nouvel algorithme pour l'appariement de graphe. Présentation:
(http://www.dmi.usherb.ca/~hlaoui/these2.pdf vom 22.11.2006)