Praktikumsversuch

Mustererkennung

A. Fenske und H. Burkhardt

Inhaltsverzeichnis

1 Mustererkennung

1.1 Einleitung

Der Mustererkennung kommt in der Bildverarbeitung eine große Bedeutung zu, die motiviert wird vom dem Wunsch, das sehr komplexe optische Wahrnehmungsvermögen des Menschen mit Hilfe von Digitalrechnern nachzuahmen. Der Mensch ist in der Lage, aus der ungeheuren Datenflut, die das Auge dem Gehirn liefert, die für ihn wichtigen Informationen zu extrahieren und auch so zu speichern, daß ein Wiedererkennungsprozeß möglich ist.

Die meisten Bildverarbeitungssysteme arbeiten nach dem in der Abbildung 1.1 gezeigten Prinzip. Eine Bildszene wird mit einem Sensor aufgenommen und digitalisiert. Um bessere Daten zu erhalten, können Bildverbesserungsmaßnahmen oder eine Bildrestauration vorgenommen werden. Je nach Anwendung und auch im Hinblick auf eine später zu erfolgende Klassifikation können sehr unterschiedliche Vorverarbeitungsschritte sinnvoll sein. Maßnahmen wie Filterung und Histogrammskalierung werden im Praktikumsversuch **Bildvorverarbeitung** behandelt. Liefert das Bild die zu interessierenden Objekte innerhalb einer komplexen Szene, ist eine Segmentierung des Bildes erforderlich.

Ist das interessante Objekt vom Hintergrund getrennt, kann die eigentliche **Mustererkennung** beginnen. Sie setzt sich aus zwei Schritten zusammen , die **Merkmalsextraktion** und die **Klassifikation**. Je nach Anforderungsgrad erhalten die beiden Stufen ihre Gewichtung. Bei einer bildunterstützten Qualitätskontrolle beispielsweise, bei der das Objekt fest positioniert ist, kommt dem Klassifikator eine größere Bedeutung zu als der Merkmalsextraktion, die sich z.B. auf rein heuristische Messgrößen stützen kann. Sollen Objekte unabhängig von ihrer Lage (z.B. Translation und Rotation) erkannt werden, was das Ziel in diesem Versuch sein soll, ist eine aufwendigere **lageinvariante Merkmalsextraktion** erforderlich.

Dieser Praktikumsversuch beschränkt sich auf die Mustererkennung von einfachen, zusammenhängenden Objekten in Binärbildern, in denen nur zwei Helligkeitsstufen, weiß (1) und schwarz (0), vorhanden sind. Eine sinnvolle Unterscheidung zwischen den Objekten und dem Hintergrund erfolgt durch Binarisieren des Grauwertbildes, welches die Kamera liefert, mit Hilfe einer Schwellwertoperation

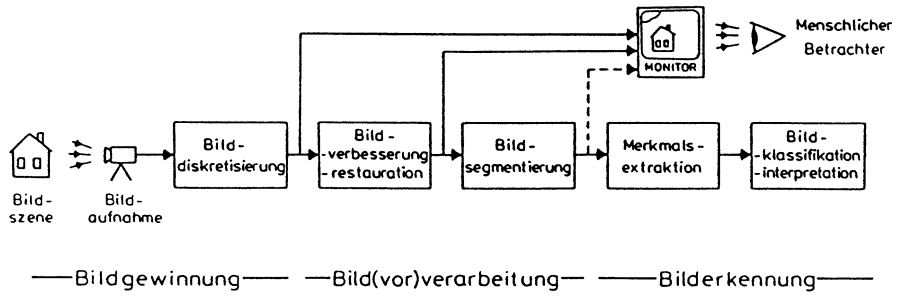


Abbildung 1.1: Stufen der Bildverarbeitung [2]

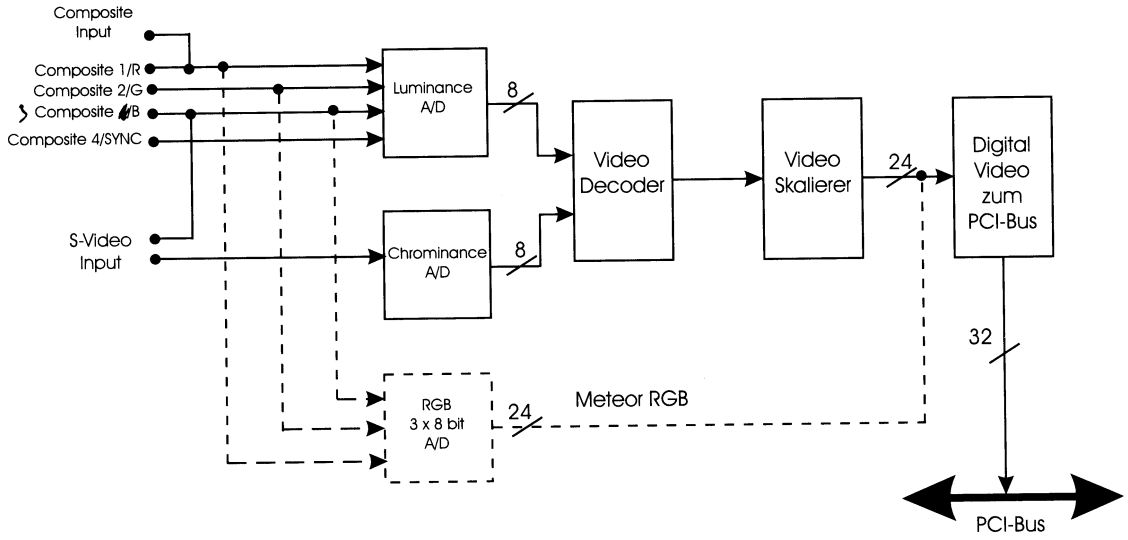


Abbildung 1.2: Bildverarbeitungskarte Matrox^{TM} Meteor

aufgrund des Grauwerthistogramms, wie sie bereits in dem Praktikumsversuch Bildvorverarbeitung behandelt wurde.

Da die Objekte als einfach zusammenhängend vorgegeben sind, sind sie durch ihre Konturlinie vollständig bestimmt. Daher beschäftigt sich Kapitel 1.2 mit der **Konturfindung**. Wichtig ist die Konturfindung auch im Hinblick auf die **Datenreduktion**, die wegen der großen Datenmengen oftmals unerläßlich ist.

In Kapitel 1.3 werden ausgehend von der Konturlinie primitive **geometrische Merkmale** der Objekte besprochen. Neben den Grundlagen der lageinvarianten Mustererkennung werden einige lageinvariante Merkmale vorgestellt, die aber nicht der Forderung der Vollständigkeit genügen, d.h. die nicht das Muster eindeutig kennzeichnen.

Durch einen direkten Vergleich des Testmusters mit allen Referenzmustern (mat-

ching) läßt sich dieser eindeutige Bezug herstellen. Wegen der zugelassenen Lagevarianz der Testobjekte ist dieses Verfahren allerdings zu rechenintensiv. Eine verfeinerte Methode, die charakteristische Merkmale der Muster und deren Beziehungen zueinander ausnutzt, ist das **Graph-matching**, das in Kapitel 1.4 vorgestellt wird. Durch geschickte Wahl der Merkmale und deren Relationen wird ein **Graph** definiert, der das Objekt eindeutig darstellt. Zur Klassifikation wird dann der Testgraph mit den Graphen der Referenzmuster verglichen.

Die in den bisherigen Kapiteln besprochenen Mustererkennungsverfahren versagen, falls die Kontur des Testmusters wesentlich gestört ist, z.B. ein Teil des Objektes fehlt, oder das Objekt teilweise von einem anderen überlappt ist. Das Problem des Segment-Vergleichs und von überlappenden Mustern wird in Kapitel 1.5 durch **Subgraph-matching** gelöst. Die Grunddefinitionen der Graphentheorie werden dabei anwendungsbezogen eingeführt.

Der Versuchsaufbau wird prinzipiell in Abbildung 1.1 deutlich. Analog aufgenommene Bilder werden diskretisiert dem Computer zugänglich gemacht. Dies geschieht durch eine spezielle Bildverarbeitungskarte (frame grabber board), die in Abbildung 1.2 schematisch erklärt ist. Ansonsten ist der Aufbau des Praktikumversuches unmittelbar einsichtig.

1.2 Konturfindung

Die Bestimmung der Kontur eines Objektes in einem Binärbild bereitet keine großen Schwierigkeiten, da eine Kante eines ausgedehnten Objektes einfach durch den Übergang von schwarz auf weiß, bzw. von 0 auf 1, definiert wird. In Abschnitt 1.2.1 wird die **Konturverfolgung** besprochen. Ein Algorithmus liefert die Konturpunkte in einer nach der Bogenlänge geordneten Folge. Der Abschnitt 1.2.2 behandelt die **Konturglättung**, die hochfrequente Störeinflüsse der Konturlinie beseitigt, und durch die eine Datenreduktion erreicht wird. Das hier beschriebene Verfahren approximiert die Objektkontur durch einen Polygonzug, also ausschließlich durch Geradenstücke. Ausgehend von dieser Polygonapproximation lassen sich in Kapitel 1.3 einfache geometrische Merkmale des Objektes berechnen. Eine weitere Möglichkeit zur Datenreduktion, ist die Kettenkodierung, die in Abschnitt 1.2.3 eingeführt wird.

1.2.1 Konturverfolgung

Die Kontur eines Objektes in einem Binärbild läßt sich mit einem Algorithmus, der die Übergänge von 0 zu 1 sucht, finden. Der Konturfinde-Algorithmus hat die Aufgabe, eine Folge von Zahlenpaaren zu ermitteln, welche die Positionen der

Hell-Dunkel-Übergänge am Objektrand in der Reihenfolge eines Umlaufs angibt.

Als Startpunkt wird dem Algorithmus ein bekannter Konturpunkt übergeben, von dem er den Objektrand so verfolgt, daß rechts bzw. links der Suchrichtung dunkle oder helle Bildbereiche liegen. Wird der Startwert im Laufe der Suche wieder erreicht, so ist die Kontur geschlossen und die Aufgabe dieses Algorithmus vollendet. Eine einfache Prozedur (Kontur) ist [1] :

1. Fahre zeilenweise über das Bild, bis ein Objektpunkt als Startpunkt gefunden ist.
2. Man stelle sich vor, daß man auf den zuletzt erreichten Punkt zugegangen ist. Wenn er ein Objektpunkt ist, trage man ihn in die Liste ein und biege nach links ab, sonst biege man nur nach rechts ab.
3. Man wiederhole Schritt 2, bis der Startpunkt erreicht ist.

Die Abbildung 1.3 zeigt das Verfahren an einem Beispiel. Die Prozedur Kontur beschreibt die Objektregion nur konsistent bei einer Vierer-Nachbarschaft (s. Abbildung 1.4). Wie aus Aufgabe 1.1 deutlich wird, ist eine Abfrage nach doppelt gezählten Konturpunkten notwendig.

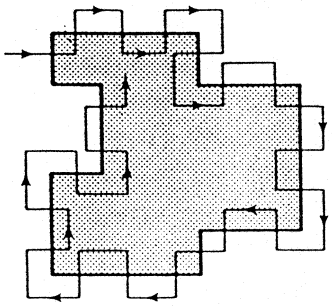


Abbildung 1.3: Konturverfolgung (Prozedur Kontur) [1]

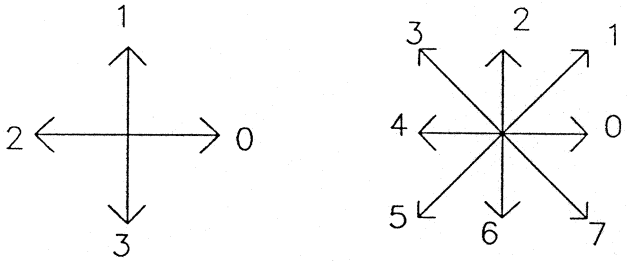
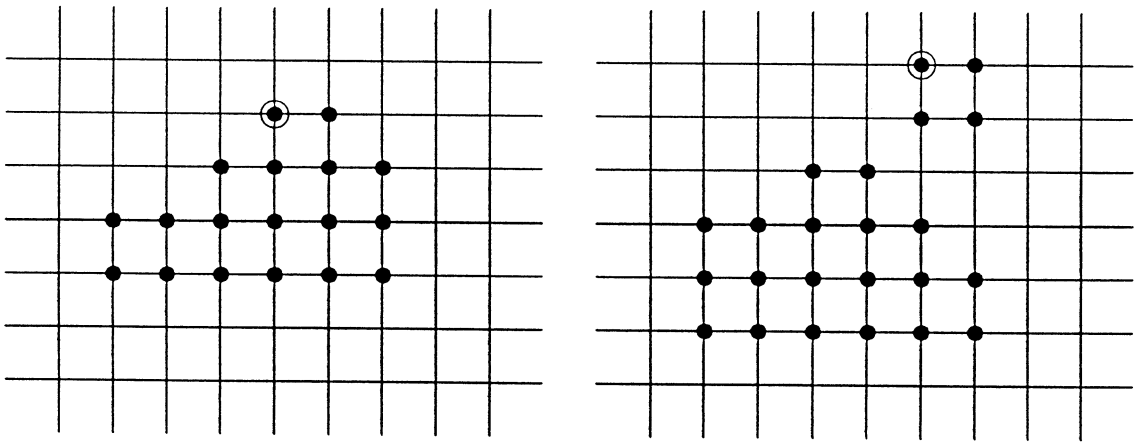


Abbildung 1.4: (a) Vierer- (b) Achternachbarschaft

Aufgabe 1.1 *Simulieren Sie den Algorithmus Kontur in folgenden zwei Beispielen mit ⊙ als Startpunkt, tragen Sie den Pfad ein und markieren Sie doppelt gezählte Konturpunkte.*



1.2.2 Konturglättung

Eine Objektkontur, die aus einem Binärbild extrahiert wird, unterliegt im allgemeinen Störungseinflüssen, die dazu führen daß die Kontur sich mit der tatsächlichen Randlinie nicht an jeder Stelle deckt. Die Störung kommt zustande durch ortsvariante Objektausleuchtung, durch Randpositionen, die sich nicht mit dem Abtastraster decken, und durch additive Rauschstörungen in der Videokamera. Diese Störeinflüsse sind im wesentlichen hochfrequenter Natur. Ein Glättungsverfahren soll derartige Fehler beseitigen. Gleichzeitig wird eine Datenreduktion angestrebt durch Zerlegung der Kontur in vorgegebene Randsegmente wie z.B. Kreisbögen, Winkelstücke, Geradenstücke usw. Es ist dann nicht mehr notwendig, jeden Randpunkt abzuspeichern, sondern nur noch die Art und Lage der einzelnen Segmente.

Das hier beschriebene Konturglättungsverfahren [10] approximiert die Kontur, die durch eine Zahlenfolge repräsentiert wird, durch einen Polygonzug, also ausschließlich durch Geradenstücke. Die Eckpunkte des Polygonzugs stellen den reduzierten Datensatz dar. Der Abbruch der Polygonzugadaption tritt ein, wenn alle Konturpunkte zwischen zwei benachbarten Stützstellen des Polygonzugs innerhalb eines vorgegebenen Abstandmaßes der Breite E um das entsprechende Geradenstück liegen. Für Geradenstücke, bei denen das Kriterium nicht erfüllt ist, wird der Punkt mit maximalem Abstand von dem Geradenstück als zusätzliche Stützstelle des Polygonzugs eingesetzt. Der Algorithmus Polapprox der Polygonapproximation, der die in Kontur ermittelten Konturpunkte benutzt, sieht

folgendermaßen aus :

1. Fahre zeilenweise über das Konturlinienbild, bis ein Konturpunkt als Startpunkt r_1 gefunden ist.
2. Bestimme den Konturpunkt r_2 mit dem größten Abstand davon, lege eine Gerade von r_1, nach r_2 und führe die nächsten Schritte für die beiden Zweige des Polygons $(r_1 \rightarrow r_2)$ und $(r_2 \rightarrow r_1)$ getrennt aus.
3. Wenn der Abstand der Konturpunkte von allen Geraden kleiner ist als ε, dann ist die Approximation beendet, sonst fahre fort mit Schritt 4.
4. Für jede der bisher gefundenen Geraden mit zu großem Approximationsfehler führe Schritt 5 und 6 aus.
5. Ermittle den Punkt P mit dem größtem Abstand von der Geraden.
6. Ersetze diese alte Gerade durch zwei neue Geraden mit P als mittlerem Eckpunkt.
7. Gehe zurück nach Schritt 3.
8. Weise alle Polygonzugpunkte zurück, die einen nicht genügend spitzen Winkel mit ihren zwei Nachbarpunkten bilden.

In Abbildung 1.5 ist ein Beispiel für die sukzessive Approximation durch die Polygonzugpunkte P_i dargestellt. Für die nachfolgenden Matching-Verfahren ist es wichtig, nur die signifikanten Punkte der Kontur zu finden, was bei dem in der Literatur unter *Split+Merge* bezeichnetem Algorithmus [3] nicht ausschließlich der Fall ist. Durch eine Abfrage nach dem Winkel an den Eckpunkten im Schritt 8 läßt sich das erreichen [11].

1.2.3 Kettenkodierung

Eine weitere Repräsentation der Konturlinie ist die Kettenkodierung (chain coding) [8]. Von einem Startpunkt beginnend wird jeweils die Richtung zum nächsten Punkt der Linie angegeben. Dabei werden vier oder acht Richtungen unterschieden, so daß zwei bzw. drei bit zur Kodierung einer Richtungsstrecke (link) ausreichen. Die vier oder acht Richtungen ergeben die in Abbildung 1.4 definierte 4- bzw. 8-Nachbarschaft.

Aufgabe 1.2 *Bilden Sie bei folgenden zwei Beispielen den 8-Kettencode mit* $\odot$ *als Startpunkt (s. Abbildung 1.4), wobei die Laufrichtung im Uhrzeigersinn sein soll. Ist der Kettencode invariant gegenüber Translationen, Rotationen oder Skalierungen (Vergrößerung/Verkleinerung) des Musters?*

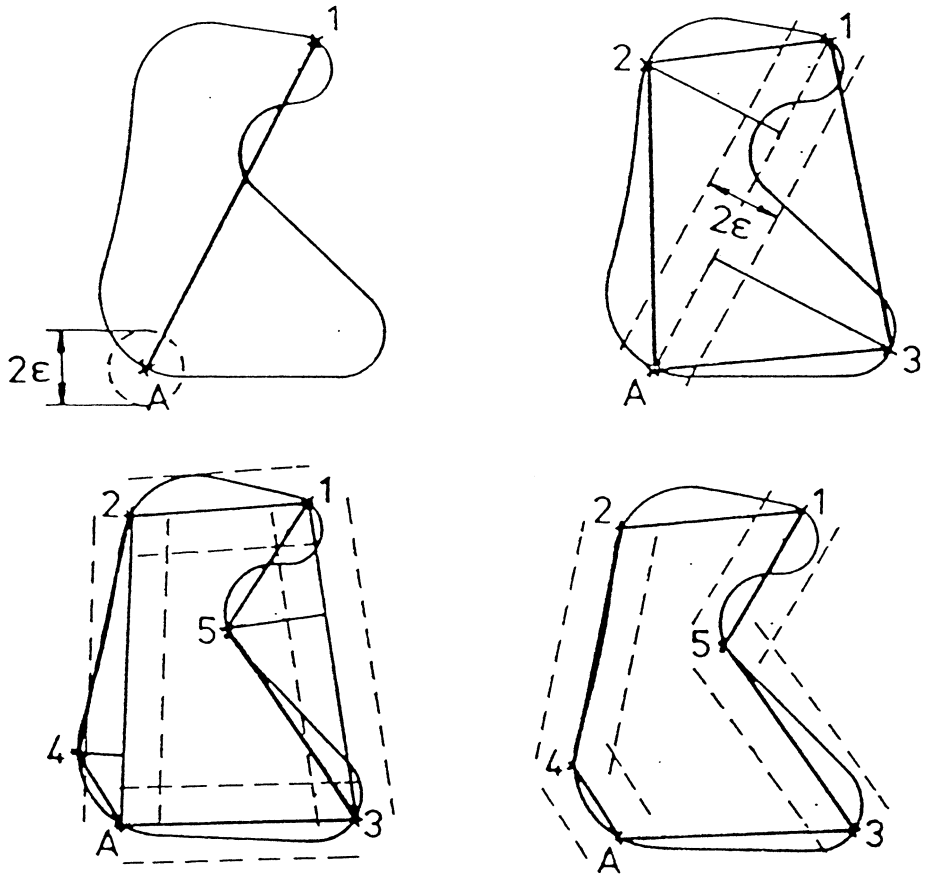
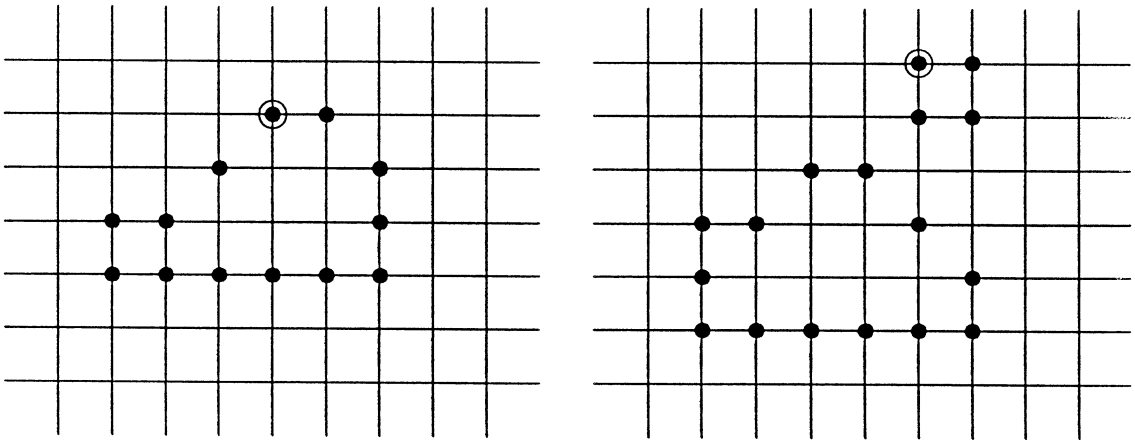


Abbildung 1.5: Polygonapproximation der Kontur [10]



Einige Merkmale des Musters wie Umfang und Fläche lassen sich direkt aus dem Kettencode berechnen. Bedeutsam ist die Kettenkodierung wegen der Datenreduktion von Linienbildern. Da hier aber nicht auf das Gebiet der Kodierung ein-

gegangen werden soll, wird in den anschließenden Kapiteln auf die anschaulichere Polygonapproximation der Konturlinie zur Merkmalsgewinnung der Binärobjekte zurückgegriffen.

1.3 Geometrische Merkmale

Für die Lösung einfacher Bildanalyseprobleme wie die Mustererkennung einzelner Objekte in Binärbildern genügen häufig einfach berechenbare geometrische Merkmale zur Objektbeschreibung. Hierzu gehören beispielsweise die **Fläche** und **Umfang** eines Binärmusters, die in Abschnitt 1.3.2 behandelt werden. Einfache geometrische Merkmale sind **Momente** durch die sich auch die Lage des Objektes bestimmen läßt. Die Momente werden in Abschnitt 1.3.3 besprochen. Die häufig in der industriellen Anwendung benutzte Methode des **Polar-check** wird in Abschnitt 1.3.4 vorgestellt. Da der Praktikumsversuch die lageinvariante Mustererkennung behandelt und die dazugehörige Vorlesung **Mustererkennung** eventuell noch nicht gehört ist, werden die Grundlagen der lageinvarianten Klassifikation im Abschnitt 1.3.1 behandelt, genauso wie sie in der Vorlesung eingeführt werden [2].

1.3.1 Grundlagen der lageinvarianten Mustererkennung

Mustererkennung ist die Theorie der Zuordnung eines unbekannten Musters X_i zu einer Bedeutungs- oder **Äquivalenzklasse** χ_j. Auf der Menge von Mustern $M = \{X, Y, Z, ...\}$ ist im allgemeinen eine Äquivalenzrelation $\sim$ mit folgenden drei Eigenschaften definiert :

1. Für jedes $X \in M$ gilt $X \sim X$ (Reflexivität).
2. Aus $X \sim Y$ folgt $Y \sim X$ (Symmetrie).
3. Aus $X \sim Y$ und $Y \sim Z$ folgt $X \sim Z$ (Transitivität).

Für jedes $X \in M$ ist die dazugehörige Äquivalenzklasse χ_x:

$$\chi_x = \{W | W \in M \qquad \text{und} \qquad X \sim W\}. \tag{1.1}$$

Äquivalenzklassen sind definiert durch alle Repräsentanten von χ oder auch parametrisch durch ein erzeugendes Element X^0 und eine Abbildungsklasse $\Gamma(X)$, die z.B. durch alle Bewegungen in der Ebene gebildet wird.

Bei der lageinvarianten Musterklassifikation muß ein unbekanntes Muster X, welches innerhalb des Gesichtsfeldes der Kamera eine Lagetransformation erfährt,

einer Bedeutungsklasse zugeordnet werden. Dabei soll angenommen werden, daß das Muster X durch eine zulässige, jedoch unbekannte Lagetransformation $\gamma_j \in \Gamma$ aus einem der möglichen Referenzmuster X_i^0 erzeugt werden kann.

$$X = \gamma_j(X_i^0) \tag{1.2}$$

Es ist $\gamma_j \in \Gamma$ unbekannt, wobei Γ die Menge der zulässigen Lagetransformationen sei. Als Menge der zulässigen Lagetransformationen des Musters soll im folgenden die Menge der Bewegungen in der Ebene (Translation und Rotation) und die Menge der Skalierung (Vergrößerung/ Verkleinerung) gelten. In (1.2) ist i auch unbekannt mit $i \in 1,...,n$, wobei n die Anzahl der (bekannten) Referenzmuster ist.

Sind x,y die Koordinaten des Referenzmusters $X^0, X^0 = X^0(x,y)$, und x',y' die des transformierten Musters X, wird Γ beschrieben durch

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = R \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} x_t \\ y_t \end{pmatrix} \tag{1.3}$$

Für den Skalierungsfaktor gilt $R > 0$. Die Translation des Musters wird beschrieben durch x_t, y_t. Der Drehwinkels des Objekts ist ϕ. (1.3) beschreibt die Ähnlichkeitsgruppe.

Die einfachste Möglichkeit der Klassifikation ist der direkte Vergleich des unbekannten Musters X in allen möglichen Lagen mit jedem Referenzmuster X_i^0 (matching). Ein weniger rechenintensives Verfahren, insbesondere bei der zugelassenen Lagevarianz der Muster, ist das **Graph-matching**, welches in den nächsten beiden Kapiteln besprochen wird.

Eine weitere Möglichkeit zur Musterklassifikation ergibt sich durch die Verwendung von **lageinvarianten Transformationen**, deren einfachste Beispiele hier als geometrische Merkmale behandelt werden. Anstatt ein gemessenes Muster in jeder Lage mit allen Referenzmustern zu vergleichen, wendet man zunächst eine Transformation τ auf das Muster X an, um die von der Ähnlichkeitsgruppe unabhängigen, strukturspezifischen Merkmale zu extrahieren. Das transformierte Muster, bzw. der gewonnene Merkmalsvektor, $\tilde{X}$ ergibt sich also aus

$$\tau : X \to \tilde{X}. \tag{1.4}$$

Die Merkmale $\tilde{X}$ werden nun mit den Transformierten der Referenzmuster $\{\tilde{X}_i^0\}$ verglichen, wobei häufig eine starke Reduktion des Merkmalsraums auf wenige, dominante Merkmale möglich ist.

Eine unter der Gruppe Γ, der Menge der zulässigen Lagetransformationen, invariante Transformation τ erfüllt notwendigerweise die Bedingung, daß alle Muster, die durch eine Lagetransformation γ_j hervorgehen, in einen Punkt des Merkmalraums abgebildet werden.

$$X = \gamma_j(X_i^0) \Rightarrow \tau(X_i^0) = \tau(X), \qquad \forall \gamma_j \in \Gamma \tag{1.5}$$

Wenn gilt zusätzlich gilt, daß aus

$$\tau(X) = \tau(X_i^0) \Rightarrow X = \gamma_j(X_i^0), \qquad \gamma_j \in \Gamma \tag{1.6}$$

folgt, ist τ eine **vollständige**, lageinvariante Transformation. Die Vollständigkeit gewährleistet, daß Muster verschiedener Äquivalenzklassen nicht in den selben Punkt im Merkmalraum abgebildet werden. Die strenge Forderung der vollständigen Separierbarkeit aller möglichen Muster ist häufig jedoch nicht erforderlich, und zwar dann, wenn aus Kenntnis der Muster die Separierbarkeit unter der Transformation gesichert ist.

Nach der Merkmalsextraktion, der Bildung des Merkmalsvektors $\tilde{X}$, erfolgt die Klassifikation der Muster anhand der lageunabhängigen, strukturspezifischen Merkmale. Als Maß für die Ähnlichkeit zweier Muster X, Y wird eine Metrik $D(\tilde{X}, \tilde{Y})$ definiert. Die Zuordnung eines Testmusters zu einer Bedeutungsklasse geschieht auf der Basis des kleinsten Abstands gemäß der Metrik D, also

$$X_i \sim X_j^0 \qquad \text{falls} \qquad D(\tilde{X_i}, \tilde{X_j^0}) = \min. \tag{1.7}$$

Dieser einfache Klassifikator heißt **Minimumdistanz-Klassifikator**. Eine Zurückweisungsklasse läßt sich einfach durch einen Schwellwert, der der gewünschte maximale Fehlerabstand zwischen einem Testmuster und seinem dazugehörigen Referenzmuster ist, bilden. Für die Robustheit des Klassifikationsverfahrens ist es wünschenswert, daß die Abbildung der gewählten Transformation τ im Sinne der gewählten Metrik stetig ist, d.h. es gibt zu jedem $\eta > 0$ ein $\varepsilon > 0$, so daß

$$D(X, Y) < \varepsilon \quad \Rightarrow \quad D(\tilde{X}, \tilde{Y}) < \eta. \tag{1.8}$$

1.3.2 Umfang, Fläche und Kompaktheit

Nach der Polygonapproximation liegt die Konturlinie als ein geschlossener Polygonzug mit N Eckpunkten $z(i) = (x(i), y(i)), i = 0, ..., N$ vor, wobei $z(0) = z(N)$ ist. Der Umfang U und die Fläche F des Objektes lassen sich daraus sehr einfach bestimmen.

$$U = \sum_{i=0}^{N-1} |z(i+1) - z(i)| \tag{1.9}$$

$$F = \frac{1}{2} \sum_{i=0}^{N-1} (x(i+1)y(i) - x(i)y(i+1)) \tag{1.10}$$

Aufgabe 1.3 *Leiten Sie die Gl. (1.10) für die Fläche her. Erinnern Sie sich dabei an das Kreuzprodukt zweier Vektoren. Wie transformieren sich der Umfang U und die Fläche F unter der zugelassenen Lagetransformation in Gl. (1.3)? Wird die notwendige Bedingung der Lageinvarianz (3-5) erfüllt?*

Als Ergebnis der Aufgabe 1.3 wird deutlich, daß der Umfang und die Fläche nicht skalierungsinvariant sind. Es kann allerdings ein größeninvarianter Formfaktor K, der Kompaktheit genannt wird [1], definiert werden durch

$$K = \frac{U^2}{4\pi F} \quad . \tag{1.11}$$

Dieser wird näherungsweise gleich 1 für Kreise und nimmt große Werte für linienhafte Objekte an.

1.3.3 Momente

Eine weitere lageinvariante Lagetransformation ist die Methode der **Momente** [12]. Sei das Objekt im Binärbild durch $f(x,y) = 1$ gekennzeichnet, dann sind die Momente von $f(x,y)$ definiert durch

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y), \quad p,q = 0,1,2,... \tag{1.12}$$

$(p+q)$ bezeichnet man als Ordnung des Momentes m_{pq}. Mit den beiden Flächenschwerpunkten $\bar{x} = m_{10}/m_{00}$ und $\bar{y} = m_{01}/m_{00}$ sind die **Zentralmomente** gegeben zu

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q f(x,y), \quad p,q = 0,1,2,... \quad . \tag{1.13}$$

Im Gegensatz zu den in (1.12) gegebenen Momenten sind die Zentralmomente invariant gegenüber Translationen des Bildsignals. Normiert man die Zentralmomente gemäß

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^r}, \quad r = \frac{p+q}{2} + 1 \tag{1.14}$$

mit $p+q = 2,3,...$, so läßt sich ein Satz von Momenten angeben, der invariant ist gegenüber Translationen, Rotationen und Größenveränderungen, also die Bedingung (1.5) erfüllt :

$$\begin{aligned} I_1 &= \eta_{20} + \eta_{02} \\ I_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\ I_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} + \eta_{03})^2 \\ I_4 &= (\eta_{30} - \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \quad . \end{aligned} \tag{1.15}$$

Damit sind neben der Kompaktheit K weitere einfache geometrische Merkmale gegeben, Objekte zu klassifizieren. Außerdem läßt sich die Lage und die Orientierung eines Objektes bezüglich des Referenzmusters durch Momente angeben. Der Schwerpunkt des Objektes z_s wird durch $z_s = (\overline{x}, \overline{y})$ bestimmt, die Orientierung θ ergibt sich aus

$$\theta = \frac{1}{2}\tan^{-1}\left(\frac{2m_{11}}{m_{20} - m_{02}}\right) \quad . \tag{1.16}$$

Aufgabe 1.4 *Erfüllen die Invarianten $I_1, ..., I_4$ die Bedingung der Vollständigkeit einer lageinvarianten Transformation (1.6) ?*

1.3.4 Polar-check

Die Methode des Polar-check zur Mustererkennung ist ein in der industriellen Anwendung häufig benutztes Verfahren [5]. Es besteht darin, einen oder mehrere Kreise mit bestimmten Radien um den Schwerpunkt des Objekts zu ziehen und die Schnittpunkte der Kreise mit der Kontur zu bestimmen (s. Abb. 1.6). Je nach Anforderungsgrad an die Klassifikationsfähigkeit lassen sich Merkmale auf zwei Weisen finden:

1. als Anzahl der Schnittpunkte zu den jeweiligen Radien,
2. als Folge von Winkeldifferenzen, die man erhält, wenn man die Schnittpunkte mit dem Schwerpunkt des Objekts verbindet.

Beim 1. Verfahren wird die Anzahl der Schnittpunkte mit denen der Referenzmuster verglichen. Beim 2. Verfahren wird die maximale Korrelation der Winkeldifferenzfolge des Testobjektes mit denen der Referenzobjekte gesucht. Sind

$f_i, i = 1,...,n$ die Listenelemente des Testmusters, $g_i, i = 1,...,n$ die des Refenzmusters, so wird das Testobjekt der Referenzklasse zugeordnet, für die

$$\frac{\sum_{i=1}^{n} f_i g_{i+\tau}}{\sqrt{\sum_{i=1}^{n} f_i^2}\sqrt{\sum_{i=1}^{n} g_i^2}} \tag{1.17}$$

bei einer zyklischen Permutation τ maximal ist.

Aufgabe 1.5 *Ist bei beiden Verfahren des Polar-check die notwendige Bedingung oder die Vollständigkeit der lageinvarianten Transformation erfüllt ?*

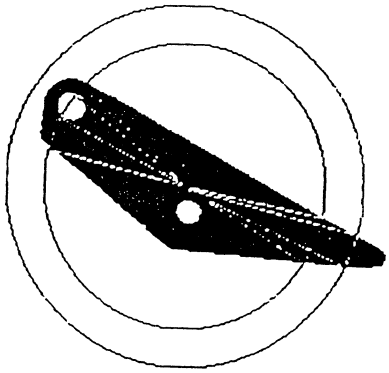


Abbildung 1.6: Polar-check [5]

1.4 Graph-matching

In Abschnitt 1.4.1 werden die Grundlagen der Darstellung eines Binärmusters als Graph besprochen [1, 7, 9]. Der zu definierende **Attributgraph** soll das Muster lageinvariant beschreiben. Die Klassifizierungsaufgabe, zu dem Testgraphen den identischen Referenzgraphen zu finden, ist der **Graph- Isomorphismus**. Die Definition wird in Abschnitt 1.4.2 gegeben. In Abschnitt 1.4.3 wird die Praxis des Graph- matching beschrieben. Die durch die Aufnahmeeinheit verursachten Fehler machen eine Vergleichs- Metrik notwendig, da eine 1:1 Abbildung nur näherungsweise gefunden werden kann. Außerdem ist der Einfluß von Bildvorverarbeitungsschritten, hier de Kantenapproximation, zu beachten.

1.4.1 Grundlagen der Darstellung eines Musters als Graph

Definition 1.1 *Ein gerichteter Graph $G = (V,E)$ besteht aus der Menge $V = \{1,2,...,|V|\}$, deren Elemente die Ecken (vertex,node) von G heißen, und einer*

Teilmenge $E \subseteq V \times V$, deren Elemente Kanten (edge) von G genannt werden. Ein Paar $(v,w) \in E$ ist eine Kante von v zu w. Kanten der Form (v,v) heißen Schlingen.

Für einen ungerichteten Graphen, hier immer Graph genannt, gilt: $(v,w) \in E \Leftrightarrow (w,v) \in E$. Im folgenden sei $n = |V|$ und $e = |E|$. Die Abbildung 1.7 zeigt ein Beispiel eines gerichtet Graphen.

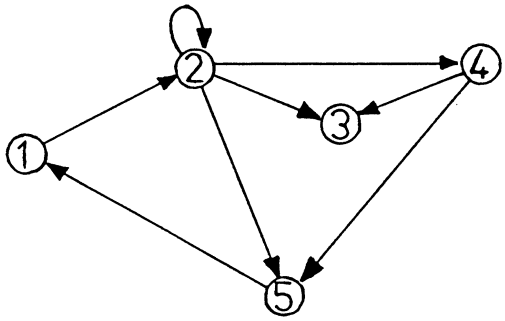


Abbildung 1.7: Gerichteter Graph [7]

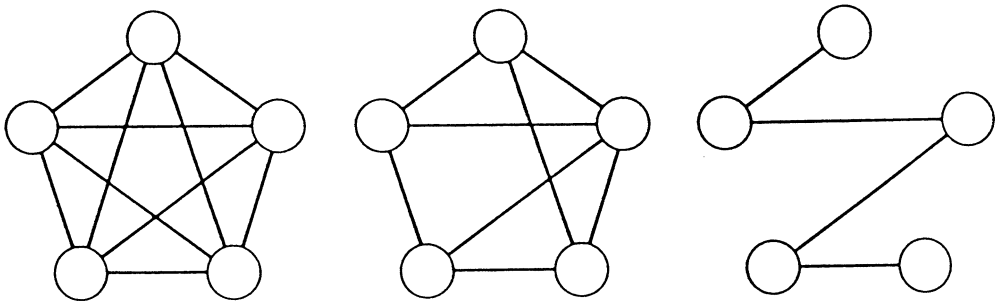


Abbildung 1.8: Graph : (a) 4-fach (b) 3-fach (c) 1-fach-zusammenhängend [1]

Zwei Methoden, einen Graphen zu speichern, sind üblich:

1. Adjazenzmatrix : $G = (V,E)$ wird repräsentiert durch eine $n \times n$ Matrix $A = (a_{ij})$ mit

$$a_{ij} = \begin{cases} 1, & (i,j) \in E \\ 0, & (i,j) \notin E \end{cases} \tag{1.18}$$

2. Liste : Die Speicherung von $G = (V,E)$ erfolgt in folgender Form :

$$n, e, i_1, k_1, i_2, k_2, ..., i_e, k_e \tag{1.19}$$

Der Index der Anfangsecke des j-ten Pfeils ist i_j, der Index der Endecke ist k_j.

Die Gl. (1.20) zeigt die Adjazenzmatrix des Beispiels. Ein Graph heißt *r-fachzusammenhängend*, wenn je zwei Ecken durch mindestens r kreuzungsfreie Wege verbunden sind. Wege zwischen v und w sind kreuzungsfrei, wenn sie außer v und w keine gemeinsamen Ecken haben. Die Abbildung 1.8 zeigt einige Beispiele.

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{1.20}$$

Aufgabe 1.6 *Wie sieht die Adjazenzmatrix des in Abbildung 1.7 gezeigten Graphen aus, falls die Kanten ungerichtet sind.*

Bisher sind den Kanten eines Graphen nur zweiwertig repräsentiert, also nur durch wahr oder falsch, ob eine Kante zwischen den Ecken existiert. Eine Verallgemeinerung ist möglich, indem man den Kanten Attribute zuordnet. Diese können z.B. zur Darstellung einer komplexen Szene Relationen wie UNTER, ÜBER, RECHTS-VON usw. oder geometrische Merkmale sein. Auch den Knoten lassen sich über die Schlingen Attribute zuordnen. Ein solcher Graph wird ein **bewerteter Graph** oder **Attributgraph** genannt [6]. Mit ihm lassen sich beliebige relationale Strukturen darstellen. Im folgenden soll der Attributgraph $G_A = (V, E, , M)$ definiert sein durch:

Definition 1.2 *Sei M eine nichtleere Menge, $G = (V, E)$ ein Graph und eine Abbildung von G in M; $: G \longrightarrow M$. Dann ist $G_A = (V, E, , M)$ ein M-bewerteter Graph.*

Besteht über den Wertebereich M kein Zweifel, so schreiben wir auch kurz $G_A = (V, E,)$. Ähnlich wie ein unbewerteter Graph läßt sich ein Attributgraph als Matrix oder Liste darstellen und speichern. Ziel ist es nun, die Elemente der Mengen V, E und die Abbildung so zu wählen, daß sich ein Binärmuster eindeutig beschreiben läßt und die Freiheitsgrade bezüglich der Lagetransformation eingeschränkt sind.

Aufgabe 1.7 *Versuchen Sie, ein Rechteck mit Hilfe eines Attributgraphen zu beschreiben.*

1.4.2 Graph-Isomorphismus

Durch die Repräsentation eines Testmusters als Attributgraph ist es zur Klassifikation nur noch notwendig, den Graphen des Testmusters mit denen der Referenzmuster zu vergleichen und nicht das Muster selbst. Ist ein Graph mit einem anderen identisch, gibt es eine bijektive Abbildung zwischen den Ecken, Kanten und Attributen der beiden Graphen, ein **Graph-Isomorphismus**

Definition 1.3 *Ein Isomorphismus des Graphen* $G_A = (V,E,,M)$ *auf den Graphen* $G'_A = (V',E',',M)$ *ist eine bijektive Abbildung* $f : V \longrightarrow V'$*, so daß gilt:*

$$\begin{aligned} (v_1,v_2) \in E &\iff (f(v_1),f(v_2)) \in E' && ,v_1,v_2 \in V \\ (v) \in M &\iff (f(v)) \in M && ,v \in V \cup E \end{aligned}$$

D.h., G ist isomorph zu G' genau dann, wenn es eine 1:1 Korrespondenz zwischen der Menge der Knoten von G und G' gibt, und die Verbindungen der Knoten mit den dazugehörigen Attributen erhalten werden.

Algorithmen zum Graph-Isomorphismus sind NP-Algorithmen, das sind *nicht*deterministische polynomial lösbare Algorithmen. D.h. nur nichtdeterministische Algorithmen lösen das Problem in polynomialer Zeit. Deterministische Algorithmen zum Graph-Isomorphismus besitzen einen *exponentiellen* Aufwand. Manche Spezialfälle lassen sich aber mit einem P-Algorithmus, einem deterministischem Algorithmus mit *polynomialer* Zeit, lösen. Sind die Graphen planar, so kann das Problem des Graph- Isomorphismus mit einem P-Algorithmus gelöst werden. Ein Graph heißt planar, wenn er sich kreuzungsfrei in der Zeichenebene darstellen läßt [1, 9].

1.4.3 Attributierung

An dieser Stelle wird eine Darstellung eines einfach zusammenhängenden Binärmusters als Attributgraph vorgestellt, die im Praktikumsversuch realisiert ist. Der Graph ist unter den zugelassenen Lagetransformationen *teilinvariant* so daß sich die Anzahl der Freiheitsgrade reduziert. Im vorliegenden Fall ist der Attributgraph invariant unter der Gruppe der Translation und Skalierung, aber variant unter der Gruppe der Rotation. Diese Varianz kommt in der Abhängigkeit des Graphen von dem Aufpunkt des Konturverfolgers und damit in der Indizierung des Polygonzugs zum Ausdruck. Der Attributgraph ist dadurch variant unter der Gruppe der zyklischen Permutationen der Knoten, was bei der Auswertung berücksichtigt werden muß.

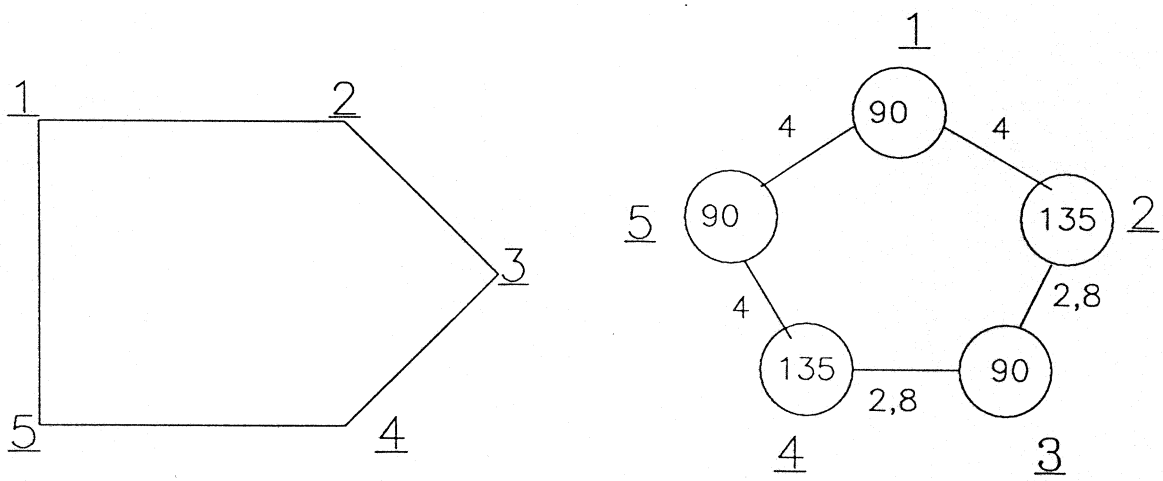


Abbildung 1.9: (a) Polygonzug (b) Attributgraph

Ausgangspunkt ist der Polygonzug mit den Eckpunkten $z_i = (x_i, y_i), i = 0, 1, 2, ..., N$ mit $z_0 = z_N$, durch den die Objektkontur approximiert ist. Die Umlaufrichtung ist entgegengesetzt dem Uhrzeigersinn, so wie der Algorithmus *Polapprox* arbeitet. Aus Abbildung 1.9 wird deutlich, daß der Polygonzug topologisch einer Darstellung als Graph entspricht, faßt man die Eckpunkte als Knoten auf. Die Folge der Eckpunkte läßt sich darstellen als 2-fachzusammenhängender Graph mit Ringstruktur. So, wie ein Eckpunkt des Polygonzugs z_i nur mit den nächsten Nachbarn z_{i+1}, z_{i-1}, verbunden ist, ist der dazugehörige Vertex v_j nur mit den jeweiligen Knoten v_{j+1}, v_{j-1}, verbunden. Die Struktur eines Polygonzugs bleibt also erhalten.

Die zu den Knoten gehörige Attributmenge $|V$ seien die Winkel ϕ_i an den Eckpunkten z_i. Es ist $\phi_0 = \phi_N$. Werden die Kanten zwischen zwei benachbarten Vertices mit den Längen der Geradenstücke $t_{i,i+1}$, zwischen den abgebildeten Eckpunkten attributiert, erhält man eine Graphdarstellung des Musters. Normiert man die Längen $t_{i,i+1}$, mit dem dem Umfang U, so wird die relationale Struktur des Musters lage- und größeninvariant abgebildet, wohingegen sich die Lagevarianz des Musters gegenüber dem Aufpunkt des Konturverfolgers als Varianz der Knoten unter der Gruppe der zyklischen Permutationen ausdrückt. Die Abbildung der relationalen Struktur des Musters auf den Attributgraphen $G_A = (V, E, , R)$ mit $\phi_i = (z_i), t_{i,i+1} = (z_i, z_{i+1})$ ist eindeutig, da jede Deformation des Musters auf einen andere Attributierung führt. Deformationen sind alle möglichen Veränderungen des Musters bis auf die zugelassenen Lagetransformationen. In Abbildung 1.9 ist ein Muster und der dazugehörige Attributgraph gezeigt. Genaugenommen liegt durch den Umlaufsinn der Kontur ein gerichteter Graph vor. Durch den Konturfindealgorithmus ist die Umlaufrichtung festgelegt, so daß sie nicht beachtet werden muß.

Bei der Definition 4.3 eines Graph-Isomorphismus wird davon ausgegangen, daß die Identität der Attribute geprüft wird. In der Praxis ist dies nicht sinnvoll, da sich aufgrund von Fehlern, die von der Aufnahmeeinheit oder äußeren Störeinflüssen herrühren, im allgemeinen die Attributgraphen G_A, G'_A desselben Musters in zwei verschiedenen Aufnahmen unterscheiden. Ein Vergleich der Attribute soll nicht nur bei Identität wahr sein, sondern auch bei Unterschreiten einer Fehlertoleranzgrenze δ im Sinne einer zu definierenden Metrik $D(\phi,\phi')$ und $D(t,t')$.

Weiterhin ist zu beachten, daß die Polygonapproximation nur wirklich markante Punkte der Kontur berechnet, da kleine Fehler in der Kontur nicht zu fehlenden oder zusätzlichen Eckpunkten führen dürfen, da die Graph-Isomorphie als erste Klassifikationsbedingung die gleiche Anzahl von Knoten beim Test- und Referenzgraph fordert. Das Genauigkeitsintervall ε der Polygonapproximation (s. Abschnitt 1.2.2) darf daher nicht zu klein (wegen Rauschen) oder zu groß (wegen fehlender Punkte) gewählt werden (s. Abbildung 1.10), was die Klassifikationsfähigkeit des Graph-matching einschränkt. Größere Störungen der Objektkontur oder überlappende Muster führen folglicherweise zu Fehlklassifikationen, die die abgewandelte Methode des Subgraphmatching erforderlich machen.

Aufgabe 1.8 *Welche Objekte lassen sich aufgrund der Polygonapproximation schlecht durch dieses Verfahren klassifizieren ?*

Ein Binärobjekt wird durch die dargestellte Attributierung als folgende Liste dargestellt :

$$n, \phi_0, t_{0,1}, \phi_1, t_{1,2,\ldots}, \phi_{n-1}, t_{n-1,n} \quad . \tag{1.21}$$

Die notwendige Bedingung für einen Isomorphismus zwischen dem Graphen G_A des Referenzmusters O_{ref} und dem Graphen G'_A des Testmusters O_{test} ist

$$n = n'. \tag{1.22}$$

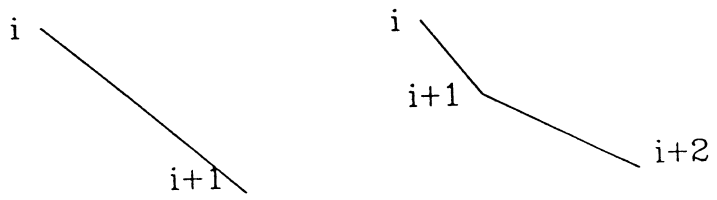


Abbildung 1.10: Fehlermöglichkeit

Ist Gl. (1.22) erfüllt, werden die übrigen Listenelemente des Testmusters mit denen der Referenzmuster verglichen. Da der Startpunkt des Konturfindealgorithmus unbekannt ist, was die Varianz der Transformation des Musters auf den Attributgraphen unter der Gruppe der zyklischen Permutationen der Knoten zur Folge hat, muß der Vergleich mit allen zyklischen Permutationen der Listenpaare (ϕ, t) von O_{test} stattfinden. Stimmen sämtliche Listenelemente innerhalb eines Fehlerintervalls δ_ϕ bzw. δ_t überein, wird das Testobjekt der Klasse des Referenzobjektes zugeordnet.

Aufgabe 1.9 *Welche Ordnung hat der aufgeführte Algorithmus ? Was ist der Grund für die Reduktion der Zeitkomplexität gegenüber einem gewöhnlichen NP-Algorithmus zur Lösung eines Graph-Isomorphismus ?*

1.5 Subgraph-matching

Das im vorigen Kapitel behandelte Verfahren des Graph-matching versagt bei rundlich geformten Konturen oder bei deformierten Objekten, bei denen z.B. ein Teil fehlt oder überdeckt wird, da der Graph- Isomorphismus eine 1:1 Korrespondenz zwischen Test- und Referenzgraph fordert. Für diese Fälle ist ein verfeinertes Matching-Verfahren, das **Subgraph-matching**, notwendig, bei dem Teile des Testgraphen mit den Referenzgraphen verglichen werden und die größte Übereinstimmung (best match) gesucht wird. In Abschnitt 1.5.1 wird der dazu benötigte **Assoziationsgraph** definiert, in dem die maximale **Clique** gesucht wird [1]. Als Attribute zum Aufbau des Assoziationsgraphen werden nicht die in Kapitel 1.4 genannten verwendet, sondern Parameter der Koordinatentransformation von Test- zu Referenzobjekt [4]. Die Parameter werden in Abschnitt 1.5.2 hergeleitet. Der Algorithmus zum Subgraph-matching wird in Abschnitt 1.5.3 behandelt.

1.5.1 Subgraph-Isomorphismus und Assoziationsgraph

Die Abbildung 1.11 illustriert zwei Beispiele, in denen ein Objekt deformiert oder teilweise überlappt ist. Wie sofort ersichtlich ist, kann kein Graph-Isomorphismus zwischen Referenz- und Testgraph existieren, falls der Graph wie in Kapitel 1.4 aus dem Binärbild extrahiert wird. Statt dessen wird versucht, Teile des Referenzgraphen Teilen des Testgraphen zuzuordnen, um so die größte Übereinstimmung festzustellen. Das Problem ist das Finden eines "doppelten" Subgraph-Isomorphismus, das sich durch Bildung eines Assoziationsgraphen mit einem einfachen Subgraph-Isomorphismus-Algorithmus lösen läßt.

Ein Teilgraph (subgraph) ist von einem Untergraph (induced subgraph) zu unterscheiden. Die Definitionen lauten folgendermaßen :

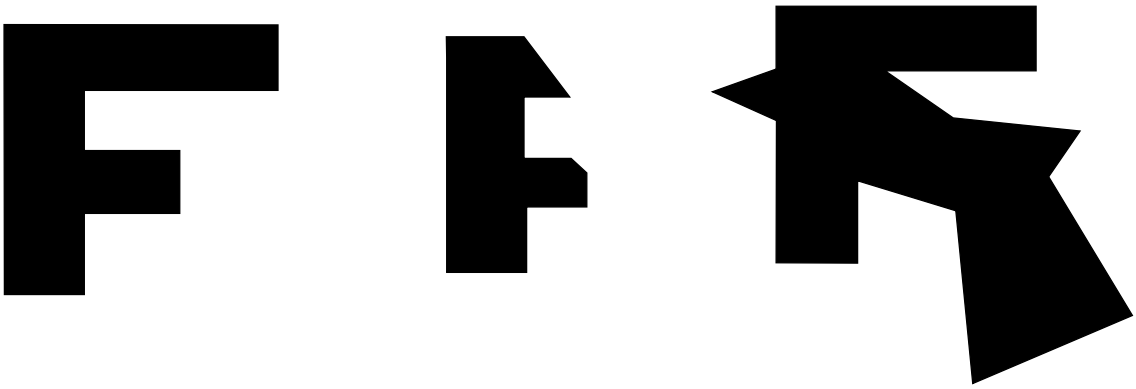


Abbildung 1.11: (a) Referenzmuster (b) deformiertes und (c) überlapptes Testmuster

Definition 1.4 *G, G' seien Graphen. $G' = (V', E')$ heißt Teilgraph (subgraph) von $G = (V, E)$, wenn $V' \subseteq V$ und $E' \subseteq E \cap V' \times V'$ gilt.*

Definition 1.5 *G' heißt Untergraph (induced subgraph) von G, wenn $V' \subseteq V$ gilt und die Menge der Kanten $E' = E \cap V' \times V'$ ist.*

Das Problem des Subgraph-Isomorphismus ist, einen Isomorphismus zwischen einem Graphen G und einem Teilgraphen des Graphen G' zu finden.

Definition 1.6 *G'_S sei Subgraph von G'. Ein Subgraph-Isomorphismus des Graphen $G = (V, E, , M)$ auf den Graphen $G'_S = (V'_S, E'_S, '_S, M)$ ist eine bijektive Abbildung $f : V \longrightarrow V'_S$, so daß gilt :*

$$\begin{aligned} (v_1, v_2) \in E &\iff (f(v_1), f(v_2)) \in E'_S && , v_1, v_2 \in V \\ (v) \in M &\iff (f(v)) \in M' && , v \in V \cup E \quad . \end{aligned}$$

Aus Abbildung 1.11 ist ersichtlich, daß die Forderung nach einem Subgraph-Isomorphismus zwischen Test- und Referenzgraph nicht allgemein genug ist, um das Testmuster einer Referenzklasse zuzuordnen, da jeweils Teile des Testobjektes nur mit Teilen des Referenzobjektes übereinstimmen. Das Problem ist also, einen Isomorphismus zu finden zwischen Teilgraphen vom Testobjekt O_{test} und Teilgraphen von O_{ref}, und zwar den, der die größte Übereinstimmung (best match) liefert. Der zu definierende Assoziationsgraph [1], auch Korrespondenzgraph genannt, ist das geeignete Instrument, die zwei Graphen miteinander zu vergleichen.

Definition 1.7 *$G = (V, E, , M)$ und $G' = (V', E', ', M')$ seien Attributgraphen. Der Assoziationsgraph $A_{G,G'} = (V_A, E_A, _A, M_A)$ wird folgendermaßen konstruiert. Für*

jedes $v \in V$ und $v' \in V'$, bilde einen Knoten $v_A \in V_A$ mit der Kennzeichnung (v, v') genau dann, wenn $(v) = '(v')$ ist. Bilde eine Kante $e_A \in E_A$ zwischen zwei Knoten $v_A^1 = (v_i, v'_j)^1$ und $v_A^2 = (v_k, v'_l)^2$ des Assoziationsgraphen genau dann, wenn $(v_i, v_k) = '(v'_j, v'_l)$ gilt. $_A$ sei eine Abbildung $_A : (V_A, E_A) \longrightarrow M_A$.

Ein Knoten in $A_{G,G'}$ wird gebildet, wenn ein Knoten von G das identische Attribut wie ein Knoten von G' hat. Der Knoten des Assoziationsgraphen drückt also eine Übereinstimmung von G und G' aus, die aber nicht auf die relationale Struktur der Graphen eingeht. Stimmt ein Kantenattribut von G mit einem von G' überein *und* bilden die Anfangs- und Endpunkte der beiden Kanten von G, G' jeweils einen Knoten im Assoziationsgraphen, haben also identische Attribute, so wird eine Kante in $A_{G,G'}$ gebildet. Kanten von $A_{G,G'}$ drücken aus, daß außer den Knoten auch relationale Strukturen beider Graphen identisch sind. Die Abbildung 1.12 zeigt ein Beispiel [1] :

Zwei Muster O, O' seien durch "primitive Elemente "(Kreis ◯, Quadrat ⊓) dargestellt, die als Knoten in den dazugehörigen Graphen G, G' erscheinen. Die Kanten zwischen zwei Elementen beschreiben die relationale Struktur, die Attribute sind also ◯ — ⊓, ◯ — ◯ oder ⊓ — ⊓. Ein Knoten des Assoziationsgraphen $A_{G,G'}$ korrespondiert zu einem Paar von Knoten von G, G', deren Eigenschaften gleich sind. Es werden jeweils die Eckpunkte von $A_{G,G'}$ verbunden, die die gleiche relationale Struktur von G und G' ausdrücken.

Gibt es einen Isomorphismus zwischen einem Teilgraphen G_S von G und einem Teilgraphen G'_S von G', so wird dieser durch eine Menge von vollständig verbundenen Knoten im Assoziationsgraphen repräsentiert. Vollständig verbunden heißt, jeder Knoten ist mit jedem anderen Knoten verbunden.

Definition 1.8 *Eine Clique der Größe N ist ein vollständig verbundener Teilgraph der Größe N.*

Die Abbildung 1.8(a) zeigt eine 5-Clique. Eine Clique der Größe N ist N – 1 - fachzusammenhängend. Am Beispiel der Abbildung 1.12 sieht man, daß ein Isomorphismus zwischen einem Subgraph von (a) und einem Subgraph von (b), also eine 1:1 Korrespondenz zwischen Teilen der Muster, eine maximale Clique im Assoziationsgraphen bildet. Eine Clique ist maximal, wenn ein Hinzufügen eines Knoten die Eigenschaft der vollständige Verbundenheit zerstört. Um die Klassifikationsaufgabe, ein deformiertes oder überlapptes Testmuster bestmöglich einer Referenzklasse zuzuordnen, zu erfüllen, muß die größte maximale Clique, d.h. die maximale Clique mit größtem N, in den Assoziationsgraphen, die vom Testmuster mit allen Referenzmustern gebildet werden, gefunden werden.

1.5.2 Koordinatentransformation

Wie in Kapitel 1.4 beim Graph-matching ist es wichtig, geeignete Attribute zur Beschreibung der Test- und Referenzmuster zu finden, um mit einem Assoziationsgraphen die in Abbildung 1.11 dargestellte Aufgabe zu lösen. Die Innenwinkel und die Kantenlängen des Polygonzugs lassen sich nicht wie in Abschnitt 1.4.3 verwenden, da die Strecken zwischen zwei Punkten nicht einheitlich über den Gesamtumfang des Musters normierbar sind, da sich der Umfang wegen einer Deformation oder Überlappung ändert. Daher sind andere Attribute zur Bildung eines Assoziationsgraphen zu verwenden, die die Klassifikationsaufgabe von deformierten oder überlappten Mustern unter der zugelassenen Lagetransformation 3.3 ermöglicht. Angenommen, zwei Punkte (x_1,y_1) und (x_2,y_2) des Referenzobjektes korrespondieren mit (x'_1,y'_1) und (x'_2,y'_2) des Testmusters. Dann gilt gemäß Gl. 1.3

$$\begin{pmatrix} x'_1 \\ y'_1 \end{pmatrix} = R\begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} x_t \\ y_t \end{pmatrix},$$
$$\begin{pmatrix} x'_2 \\ y'_2 \end{pmatrix} = R\begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}\begin{pmatrix} x_2 \\ y_2 \end{pmatrix} + \begin{pmatrix} x_t \\ y_t \end{pmatrix}. \tag{1.23}$$

Die Parameter $TP = (\phi, R, x_t, y_t)$ der Koordinatentransformation ergeben sich aus Gl. 1.23 zu

$$R = \sqrt{\frac{(x'_1 - x'_2)^2 + (y'_1 - y'_2)^2}{(x_1 - x_2)^2 + (y_1 - y_2)^2}} \tag{1.24}$$

$$\tan\phi = \frac{(y'_1 - y'_2)(x_1 - x_2) - (y_1 - y_2)(x'_1 - x'_2)}{(x'_1 - x'_2)(x_1 - x_2) + (y_1 - y_2)(y'_1 - y'_2)} \tag{1.25}$$

$$x_t = \frac{1}{2}(x'_1 + x'_2 + R(y_1 + y_2)\sin\phi - R(x_1 + x_2)\cos\phi) \tag{1.26}$$

$$y_t = \frac{1}{2}(y'_1 + y'_2 - R(y_1 + y_2)\cos\phi - R(x_1 + x_2)\sin\phi) \tag{1.27}$$

Die Verwendung der Parameter zur Bildung eines Assoziationsgraphen $A_{G,G'}$ wird in Abbildung 1.13 deutlich. Gegeben seien die Eckpunkte $[z_{i-1}, z_i, z_{i+1}]$ des einen (Referenz-) Musters und $[z'_{j-1}, z'_j, z'_{j+1}]$ des (Test-) Musters $(z = (x,y))$. Bei einer Korrespondenz zwischen z_{i-1}, z_i und z'_{j-1}, z'_j ergeben sich $TP^-_{ij} = (R^-_{ij}, \phi^-_{ij}, x^-_{t,ij}, y^-_{t,ij})$ als Transformationsparameter, bei einer Korrespondenz zwischen z_{i+1}, z_i und z'_{j+1}, z'_j ergeben sich $TP^+_{ij} = (R^+_{ij}, \phi^+_{ij}, x^+_{t,ij}, y^+_{t,ij})$. Soll nun die Ecke um z_i, die mit $[z_{i-1}, z_i, z_{i+1}]$ bezeichnet wurde, ein möglicher Matchkandidat für die Ecke um z'_j sein, so ist die notwendige Bedingung dafür die Gleichheit der Parameter :

$$R_{ij}^{-} = R_{ij}^{+}, \qquad \phi_{ij}^{-} = \phi_{ij}^{+}, \qquad x_{t,ij}^{-} = x_{t,ij}^{+}, \qquad y_{t,ij}^{-} = y_{t,ij}^{+} \quad . \tag{1.28}$$

Ist Gl. (1.28) erfüllt, so wird ein Knoten mit der Kennzeichnung (v_i, v'_j) im Assoziationsgraphen gebildet. Als Attribut erhält er die Parameter der Koordinatentransformation der korrespondierenden Polygonzugecken z_i und z'_j, also TP_{ij}. Werden alle Ecken des Testmusters mit allen Ecken eines Referenzmusters verglichen, so wird nach obiger Konstruktionsregel ein Assoziationsgraph erzeugt, in dem die Knoten übereinstimmende Ecken von G und G' darstellen. Sämtliche Knoten mit übereinstimmenden Parametern werden verbunden, so daß eine Clique gebildet wird. Die Größe der Clique sagt aus, wie viele Ecken des Testmusters mit Ecken des Referenzmusters matchen, d.h. unter der jeweiligen Lagetransformation übereinstimmen. Ein Testmuster wird der Referenzklasse zugeordnet, dessen Referenzmuster mit dem Testmuster den Assoziationsgraphen mit der größten maximalen Clique bildet.

1.5.3 Berechnung

In einem unbewerteten Graph ist das Suchen einer Clique ein NP- vollständiges Problem, d.h. ein deterministischer Algorithmus löst das Problem mit exponentiellem Aufwand. Im vorliegenden Fall vereinfacht sich der Aufwand durch die Attributierung der Knoten des Assoziationsgraphen mit den Parametern der Koordinatentransformation TP_{ij}.

Besitzt das Testobjekt n Polygonpunkte, und wird es mit einem Referenzobjekt mit m Punkten verglichen, so werden $n \times m$ Möglichkeiten für Matchkandidaten durchgerechnet. Ist m ungefähr gleich n, so ist der Aufwand für den Algorithmus quadratisch in $n, O(n^2)$. Die jeweiligen Ecken, die zueinander matchen, bilden den Assoziationsgraph, der sich als Liste ab speichern läßt. Seien $z_i;\ i = 1,...,N$ die Polygoneckpunkte des einen Musters $z'_j;\ j = 1,...,M$ die des zu vergleichenden Musters, und gibt es p richtige Eckenzuordnungen, dann repräsentiert folgende Liste den Assoziationsgraph :

$$\begin{aligned}
z_{i_1}, z'_{j_1} \quad &; \quad R_{i_1,j_1}, \phi_{i_1,j_1}, x_{t_{i_1,j_1}}, y_{t_{i_1,j_1}} \\
z_{i_2}, z'_{j_2} \quad &; \quad R_{i_2,j_2}, \phi_{i_2,j_2}, x_{t_{i_2,j_2}}, y_{t_{i_2,j_2}} \\
&\ldots \\
z_{i_p}, z'_{j_p} \quad &; \quad R_{i_p,j_p}, \phi_{i_p,j_p}, x_{t_{i_p,j_p}}, y_{t_{i_p,j_p}}
\end{aligned} \tag{1.29}$$

(z_i, z'_j) repräsentieren die korrespondierenden Ecken und R, ϕ, x_t, y_t sind die Parameter der dazugehörigen Koordinatentransformation. Eine Clique wird von den

Eckpaaren (z_i, z'_j) gebildet, die die gleichen Transformationsparameter haben. Die Liste 5.7 muß also nach gleichen Parametersätzen abgesucht werden, was im schlechtesten Fall einen Aufwand von $(p-1)p/2$ erfordert. Da im Extremfall p gleich n ist, hat der Algorithmus zum Suchen auch einen quadratischen Aufwand in n. Somit hat der hier vorgestellte Algorithmus zum Subgraph-matching insgesamt die Ordnung $O(n^2)$.

Der Match mit der größten Anzahl an identischen Parametersätzen liefert die Zuordnung des Testmusters zu einer Referenzklasse. Wegen der in Abschnitt 1.4.3 erwähnten Fehlerquellen muß natürlich dabei ein Fehlertoleranzintervall berücksichtigt werden.

Aufgabe 1.10 *Machen Sie sich die Bildung des Assoziationsgraphen an einem Beispiel klar. Überlegen Sie, was das Verfahren kann und welche Nachteile es hat.*

1.6 Aufgaben im Praktikum

Aufgabe 1.11 *Binarisieren Sie eines der bereitgestellten Bilder mit der Funktion* `binarisierung`*. Wählen Sie ein Bild mit mehreren dunklen Objekten auf weißem Hintergrund. Zeigen Sie das Original und das binarisierte Bild mit der Funktion* `view` *an. Extrahieren Sie die Konturen mit der Funktion* `konturen` *und zeigen Sie diese mit der Funktion* `zeigekonturen` *an.*

Aufgabe 1.12 *Testen Sie den Algorithmus* `polapprox` *zur Polygonapproximation mit verschiedenen Fehlern* ε*. Bestimmen Sie* ε *so, daß brauchbare Ergebnisse geliefert werden. Dazu wird mit* `unpack` *eine einzelne Kontur aus dem von* `konturen` *zurückgelieferten Array extrahiert und der Funktion* `polapprox` *übergeben. Zum Anzeigen der Approximation gegenüber dem Original kann die Funktion* `zeigekonturen` *verwendet werden.*

Aufgabe 1.13 *Schreiben Sie eine Funktion* `umf`*, die den Umfang eines Polygonzugs berechnet.*

Aufgabe 1.14 *Schreiben Sie eine Funktion* `flae`*, die die Fläche eines Polygonzugs berechnet.*

Aufgabe 1.15 *Schreiben Sie eine Funktion* `kompaktheit`*, die die Kompaktheit eines Polygonzugs berechnet.*

Aufgabe 1.16 *Bestimmen Sie die Kompaktheit K von vier Referenzobjekten. Anschließend klassifizieren Sie unbekannte Muster mit Hilfe eines Minimumdistanz-Klassifikators* `klasskom`.

Aufgabe 1.17 *Schreiben Sie eine Funktion* `schwer`*, die den Schwerpunkt eines Binärbildes berechnet.*

Aufgabe 1.18 *Erlernen Sie die Momenten-Invarianten* $I_1, ..., I_4$ *aus den Binärbildern von vier Mustern durch die Funktion* `moment`*. Anschließend klassifizieren Sie unbekannte Muster mit Hilfe eines Minimumdistanz-Klassifikators* `klassmom`.

Aufgabe 1.19 *Extrahieren Sie die Konturen mehrere Objekte und vereinfachen Sie diese (*`konturen`, `polapprox`*). Zeigen Sie die Knoten mit den dazugehörigen Attributen, also den Winkeln und den Bogenlängen, als Liste auf (*`graphlist`*).*

Aufgabe 1.20 *Verwenden Sie vier verschiedene Objekte als Referenzmuster und klassifizieren Sie überlappte oder deformierte Testmuster mit Hilfe des Programms* `subgraph`.

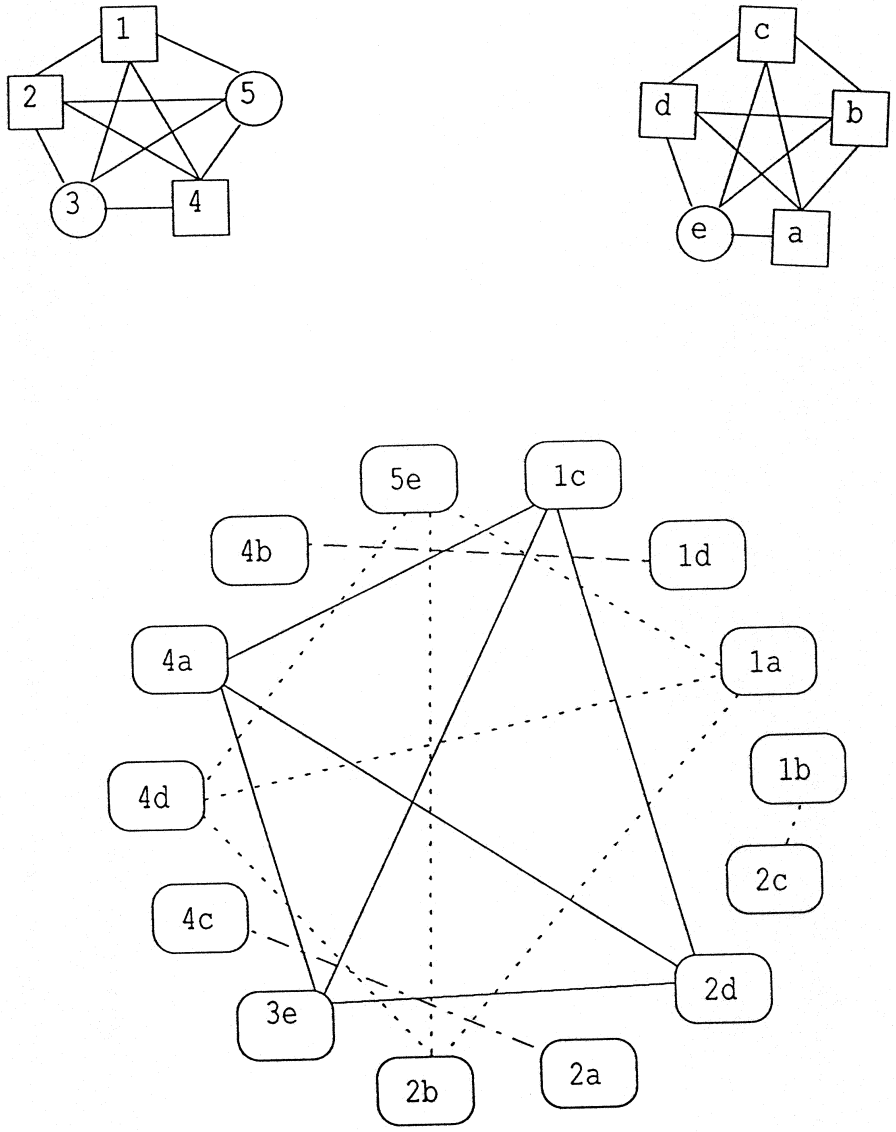


Abbildung 1.12: Beispiel zum Assoziationsgraph. Zwei Graphen (a) und (b) mit ◯ oder ⊓ als Knotenmerkmale und ◯ — ⊓, ◯ — ◯ oder ⊓ — ⊓ als Kantenmerkmale sollen verglichen werden. Den Assoziationsgraph zeigt (c). Die verschiedenen Cliquen sind durch Strichmarkierungen hervorgehoben.

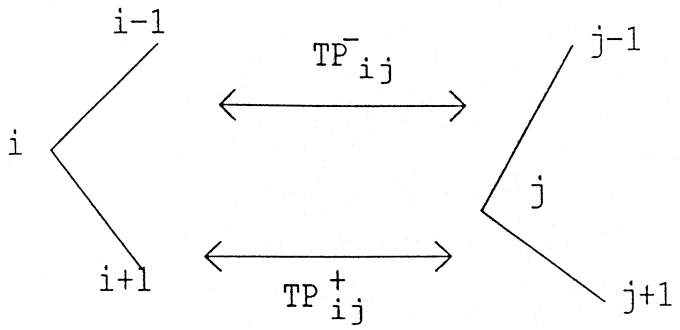


Abbildung 1.13: Vergleich einer Ecke $[z_{i-1}, z_i, z_{i+1}]$ des Testpolygonzugs mit einer Referenzecke $[z'_{j-1}, z'_j, z'_{j+1}]$

Literaturverzeichnis

[1] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, 1982.

[2] H. Burkhardt. Mustererkennung. Vorlesung am Institut für Informatik, Albert-Ludwigs-Universität Freiburg.

[3] B. Bässmann and P. W. Besslich. *Konturorientierte Verfahren in der digitalen Bildverarbeitung*. Springer-Verlag, Berlin, 1989.

[4] L. S. Davis. Shape matching using relaxation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(1):60–72, January 1979.

[5] J. P. Foith. *Intelligente Bildsensoren zum Sichten, Handhaben, Steuern und Regeln*. Springer-Verlag, Berlin, 1982.

[6] R. Halin. *Graphentheorie I*. Wissenschaftliche Buchgesellschaft, Darmstadt, 1980.

[7] K. Mehlhorn. *Graph Algorithms and NP-Completeness*. Springer-Verlag, Berlin, 1984.

[8] H. Niemann. *Klassifikation von Mustern*. Springer-Verlag, Berlin, 1983.

[9] H. Noltemeier. *Graphentheorie mit Algorithmen und Anwendungen*. Walter de Gruyte, Berlin, 1976.

[10] H. Schorb. Modellhafter Einsatz von Mustererkennungsverfahren für Grauwertbilder und Binärbilder. Technical report, Institut für Meß- und Regelungstechnik der Universität Karlsruhe, 1983.

[11] C. Teh and R. T. Chin. On the detection of dominant points on digital curves. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 11-8, pages 859–872, August 1989.

[12] F. M. Wahl. *Digitale Bildsignalverarbeitung*. Springer-Verlag, Berlin, 1984.