

# Distanzfunktionen für Polygone und ihre Anwendung in der Roboterlokalisierung

Diplomarbeit (Informatik)  
Thomas Wahl

Universität Würzburg, 6. Juni 1997



Bayerische Julius-Maximilians-Universität Würzburg  
Lehrstuhl für Informationsstrukturen  
und wissensbasierte Systeme  
(Informatik I)

# Distanzfunktionen für Polygone und ihre Anwendung in der Roboterlokalisierung

Diplomarbeit Thomas Wahl

6. Juni 1997

Betreuer: Prof. Dr. Hartmut Noltemeier  
Dipl.-Inform. Oliver Karch



# Vorwort

Die Selbstlokalisierung von autonomen Fahrzeugen und Robotern ist eine Problematik, mit der sich Konstrukteure solcher Systeme aus verschiedenen Gründen beschäftigen müssen:

Unter *Autonomie* versteht man nicht nur die Fähigkeit, daß sich ein Fahrzeug nach einer Standortinitialisierung selbständig in einem abgegrenzten Gelände bewegt. Es gehört ebenso dazu, daß es diese anfängliche Positionsbestimmung ohne fremde Hilfe durchführen kann. Dabei hat das autonome System zunächst meist keinerlei Information über den aktuellen Standort außer einer Karte, in der das Gelände eingezeichnet ist. Derartige Situationen treten nicht nur zu Beginn des Einsatzes eines Roboters auf, sondern auch nach technischen Ausnahmesituationen, wie Stromausfällen oder einem Versagen der Sensorik.

Selbst wenn das autonome System seinen Standort kennt und nun bei allen Bewegungen die Veränderung der Koordinaten seiner Position protokolliert, sind nach einer gewissen Zeit große Abweichungen des berechneten Standorts vom tatsächlich zutreffenden mit den heutigen Mitteln der Technik nicht zu verhindern. Durch unebene oder weiche Böden, aber auch Ungenauigkeiten im Bewegungsapparat des Fahrzeugs kommt es unweigerlich zu einem Abdriften von der beabsichtigten Bahn. Auch dieses Problem kann durch eine regelmäßige Standort-Neubestimmung zumindest gelindert werden. Indiskutabel ist jedenfalls der Eingriff eines Menschen in immer wiederkehrenden Abständen, der die Position des Roboters korrigiert.

Zur Lokalisation gibt es vor allem zwei wesentliche Grundrichtungen. Die eine Möglichkeit benutzt Landmarken, d.h. Markierungen an Wänden oder auf dem Untergrund, die der Roboter erkennt und danach seinen Kartenstandort ermitteln kann. Die andere Vorgehensweise betont mehr die Unabhängigkeit des Roboters auch von derartigen Kennzeichnungen des Geländes und untersucht die Möglichkeiten der Lokalisation ausschließlich anhand des sichtbaren Teils der Umgebung.

Die vorliegende Diplomarbeit bezieht sich auf die landmarkenfreie Navigation. Dazu gibt es einen theoretischen Ansatz von Guibas, Motwani und Raghavan ([GMR95]), der auf dem Vergleich sogenannter *Sichtbarkeitspolygone* vom Roboterstandort einerseits und von Kartenpositionen andererseits beruht. Sobald dabei eine Übereinstimmung festgestellt wurde, ist ein Kartenstandort gefunden, der bezüglich des einsehbaren Bereiches als Position des Roboters in Frage kommt.

Dieser Ansatz stößt in der praktischen Anwendung auf Schwierigkeiten. Dafür sind vor allem die technischen Möglichkeiten der Aufzeichnung des Sichtbarkeitspolygons verantwortlich. Verwendet man hierzu einen Entfernungssensor irgendeiner Art (z.B. Laserscanner), so gibt es nicht nur Meßungenauigkeiten, sondern auch das Problem einer begrenzten Auflösung des Scanbildes, so daß man weder präzise noch vollständige Information über seine Umgebung erwarten kann. Ein einfacher Vergleich von Sichtbarkeitspolygone schlägt also fehl.

Die Idee zur Bewältigung dieser Schwierigkeiten stellen Distanzfunktionen dar, die die Ähnlichkeit zwischen einem Scan und einem aus der Karte extrahierten Sichtbarkeitspolygon beschreiben sollen. Das Bewertungskriterium für einen Kartenstandort ist dann nicht mehr Identität, sondern nur noch Ähnlichkeit zum realen Standort hinsichtlich des sichtbaren Anteils der Szenerie.

In der vorliegenden Arbeit geht es um die Anforderungen an Distanzfunktionen zwischen Polygonen in diesem konkreten Kontext. Dazu werden in einem einführenden Kapitel zunächst Grundbegriffe erläutert und die bereits zitierte Methode von Guibas et al. zusammen mit ihren Problemen in der Praxis genauer vorgestellt.

Der Hauptteil der Arbeit liegt in Kapitel 2. Dort werden zwei Ansätze zur Definition von Distanzfunktionen zwischen *sternförmigen* Polygonen (mit denen wir es hier stets zu tun haben) untersucht, die intensiv von den Gegebenheiten der Roboterlokalisierung Gebrauch machen. Neben einer theoretischen Analyse der definierten Metriken werden jeweils auch Hinweise zu deren Implementation gegeben.

Kapitel 3 beschreibt Distanzfunktionen anderer Autoren, die sich nicht speziell auf sternförmige Polygone beziehen, aber dennoch vielversprechend erscheinen. Es werden ihre Stärken und Schwächen, auch im Vergleich mit den Distanzen aus Kapitel 2, untersucht.

Nachdem nun etliche Ähnlichkeitsfunktionen zwischen Polygonen angeführt wurden, geht es im abschließenden Kapitel 4 um eine Diskussion der Einsatzmöglichkeiten in der Praxis. Das Vorgehen zur Lokalisation beruht nach wie vor auf dem Verfahren von Guibas et al., welches jedoch für die Praxis stark überarbeitet werden muß. Anschließend faßt dieses Kapitel einige Punkte zusammen, die bis dahin entweder nicht angesprochen wurden oder offengeblieben sind.

Zu dieser Arbeit gehören auch die Implementationen der beiden Distanzen aus Kapitel 2. Im Anhang A sind einige Beispiele für das Verhalten der Metriken in verschiedensten Situationen aufgeführt, die die theoretischen Analysen bestätigen sollen. – Ebenfalls im Anhang ist eine Auswahl von Begriffen und Symbolen verzeichnet, die möglicherweise nicht dem üblichen mathematischen Wortschatz angehören.

Dort sind jedoch keine allgemein bekannten Definitionen und Bezeichnungen enthalten. Gleiches gilt generell für Sachverhalte aus der Mathematik und der Geometrie, die im Laufe dieser Arbeit zitiert werden. Beweise werden bis auf wenige Ausnahmen nur für die hier entwickelten Zusammenhänge dargeboten.

Zum Abschluß sei an dieser Stelle allen gedankt, die mich bei der Anfertigung dieser Diplomarbeit unterstützt und durch schwierige Phasen hindurch begleitet haben. Dazu gehören in erster Linie meine beiden Betreuer, Herr Prof. Hartmut Noltemeier und Herr Oliver Karch. Für die zahlreichen Vorschläge und Diskussionsangebote gilt mein Dank weiterhin Herrn Riko Jacob, der ständig Interesse am Fortgang der Arbeiten bekundet hat.

# Inhaltsverzeichnis

<b>Vorwort</b>	<b>iii</b>
<b>Abbildungsverzeichnis</b>	<b>vi</b>
<b>Tabellenverzeichnis</b>	<b>viii</b>
<b>1 Einführung</b>	<b>1</b>
1.1 Grundbegriffe . . . . .	1
1.2 Ein Ansatz für die Lokalisation . . . . .	2
1.2.1 Sichtbarkeitspolygone und -skelette . . . . .	3
1.2.1.1 Begriffsklärungen . . . . .	3
1.2.1.2 Eigenschaften . . . . .	5
1.2.2 Skizze des Algorithmus' . . . . .	7
1.2.3 Anwendungsaspekte . . . . .	9
1.3 Morphologische Verfahren im Überblick . . . . .	11
1.3.1 Geometrische Transformationen . . . . .	11
1.3.2 Distanz- und Ähnlichkeitsfunktionen . . . . .	13
1.3.3 Matching . . . . .	16
1.3.4 Einbettung (Embedding) . . . . .	17
1.3.5 Vereinfachung (Simplification) . . . . .	17
1.3.6 Morphing . . . . .	18
<b>2 Vergleich sternförmiger Polygone</b>	<b>19</b>
2.1 Ein Ansatz über innere Abstände . . . . .	20
2.1.1 Die Polarkoordinatenfunktion . . . . .	20
2.1.2 Analytische Beschreibung . . . . .	21
2.1.3 Ableitung einer Ähnlichkeitsfunktion . . . . .	23
2.1.3.1 Die Festlegung des Kernpunktes $p$ . . . . .	23
2.1.3.2 Definition der Ähnlichkeitsfunktion . . . . .	25
2.1.4 Berechnung der PKF-Metrik . . . . .	28
2.1.4.1 Lineare Approximation . . . . .	29
2.1.4.2 Startpunktauswahl . . . . .	35
2.1.4.3 Komplexität der Berechnung . . . . .	38
2.1.5 Eigenschaften der PKF-Ähnlichkeitsfunktion; Ausblick . . . . .	39
2.2 Ein flächenorientierter Ansatz . . . . .	40
2.2.1 Die Flächeninhaltsfunktion . . . . .	41

2.2.2	Analytische Beschreibung . . . . .	42
2.2.3	Eigenschaften der Flächeninhaltsfunktion . . . . .	43
2.2.4	Die Flächeninhaltsdistanz . . . . .	45
2.2.4.1	Die FIF als quasiperiodische Funktion . . . . .	45
2.2.4.2	Versuch zur Definition einer Ähnlichkeitsfunktion . . . . .	47
2.2.4.3	Definition der skalierten Flächeninhaltsmetrik . . . . .	52
2.2.4.4	Berechnung der skalierten Flächeninhaltsmetrik . . . . .	56
2.2.5	Abschließendes zur Flächeninhaltsdistanz . . . . .	58
2.3	Zusammenfassung . . . . .	61
2.3.1	Vergleich PKF- und FIF-Distanz . . . . .	61
2.3.2	Rekonstruktion der Polygone aus PKF- und FIF-Darstellungen . . . . .	62
2.3.2.1	Zur Polarkoordinatenfunktion . . . . .	63
2.3.2.2	Zur Flächeninhaltsfunktion . . . . .	63
2.3.2.3	Zur linear approximierten Flächeninhaltsfunktion $FIF_{lin}$ . . . . .	63
2.3.3	Allgemeines . . . . .	66
<b>3</b>	<b>Vergleich beliebiger Polygone</b>	<b>71</b>
3.1	Arkin-Metrik . . . . .	71
3.1.1	Die Turning-Funktion . . . . .	72
3.1.2	Eigenschaften der Turning-Funktion . . . . .	73
3.1.3	Ableitung der Metrik . . . . .	74
3.1.4	Die Arkin-Metrik in der Roboterlokalisierung . . . . .	75
3.1.5	Abschließendes zur Arkin-Metrik . . . . .	77
3.2	Maes-Distanz . . . . .	77
3.2.1	Polygondarstellung . . . . .	77
3.2.2	Eigenschaften der Stringdarstellung . . . . .	79
3.2.3	Stringvergleich über Edit-Distanzen . . . . .	79
3.2.4	Berechnung der Maes-Distanz . . . . .	82
3.2.5	Probleme beim Einsatz der Maes-Distanzfunktion . . . . .	84
3.2.6	Zusammenfassung . . . . .	86
<b>4</b>	<b>Offene Probleme; Ausblick</b>	<b>87</b>
4.1	Einsatz der Metriken in der Roboterlokalisierung . . . . .	87
4.1.1	Skelettbasierter Ansatz . . . . .	88
4.1.2	Polygonbasierter Ansatz . . . . .	89
4.2	Weitere Gesichtspunkte im Überblick . . . . .	90
4.2.1	Referenzpunkte . . . . .	90
4.2.2	Reichweitenbeschränkung, Winkelinkrement, Verdeckung . . . . .	93
4.3	Einzelheiten . . . . .	95
<b>A</b>	<b>Testbeispiele</b>	<b>99</b>
A.1	Beispiele für Spezialfälle . . . . .	99
A.2	Vergleiche von Polygonen gegen ein Muster . . . . .	104
<b>B</b>	<b>Begriffe und Symbole</b>	<b>107</b>
B.1	Definitionen . . . . .	107
B.2	Symbole . . . . .	109
	<b>Literaturverzeichnis</b>	<b>111</b>

# Abbildungsverzeichnis

1.1	Eine polygonale Szene . . . . .	2
1.2	Das Sichtbarkeitspolygon eines Betrachterstandpunktes . . . . .	3
1.3	Sichtbarkeitspolygon und zugehöriges Skelett . . . . .	4
1.4	Eine Szene mit Mehrdeutigkeiten . . . . .	7
1.5	Eine Zerlegung in Sichtbarkeitszellen . . . . .	8
1.6	Scanbild eines Szenenausschnitts; Vergleich ideales – reales Sichtbarkeitspolygon . . . . .	10
1.7	Sehr ähnliche, aber weit entfernte und recht verschiedene, aber nah beieinanderliegende Objekte . . . . .	14
1.8	Vereinfachung eines Weges durch einen Hindernisraum . . . . .	18
2.1	Ein Polygon mit Kernpunkt und die Entfernung der Peripheriepunkte als Funktion vom Winkel . . . . .	20
2.2	Sektorenzerlegung eines Polygons; ein Ausschnitt . . . . .	22
2.3	Veränderung des Kerns ähnlicher Polygone durch verrauschte Kanten . . . . .	24
2.4	Extremfälle der Kernveränderung durch verrauschte Kanten . . . . .	25
2.5	Approximation der PKF durch Geradenstücke bedeutet Approximation der Polygone durch Bogensegmente . . . . .	30
2.6	Umwandlung eines Polygons in eine Figur mit stückweise linearer Polarkoordinatenfunktion (Skizze) . . . . .	31
2.7	Graph der Funktion $y = 1/\sin x$ ; eine Konstellation benachbarter Polygonecken mit schlechter Näherung der PKF durch eine Gerade . . . . .	32
2.8	Einteilung der Zeichenebene in Streifen; Fläche zwischen zwei Geraden in Abhängigkeit von der Lage des begrenzenden Streifens . . . . .	33
2.9	Ungünstige Ecken des Scanbildes als Startpunkte der PKF . . . . .	36
2.10	Ein Polygon und der überstrichene Flächeninhalt als Funktion vom Strahlwinkel . . . . .	41
2.11	Sektorenzerlegung eines Polygons; ein Ausschnitt . . . . .	42
2.12	Konvexer oder konkaver Verlauf der FIF . . . . .	45
2.13	Verschiebung des Meßausgangspunktes und des Meßnullpunktes . . . . .	46
2.14	Zur Verletzung der Dreiecksungleichung . . . . .	48
2.15	Minimale FIF-Distanz der Polygone $\mathcal{A}$ und $\mathcal{C}$ . . . . .	49
2.16	FIF-Distanz der Polygone $\mathcal{A}$ und $\mathcal{B}$ bzw. $\mathcal{B}$ und $\mathcal{C}$ . . . . .	50
2.17	Variierende Verschiebungsrichtungen und Gesamtverschiebung . . . . .	53
2.18	Gleichmäßige Verschiebung des Startpunktes in zwei Polygonen . . . . .	54
2.19	Ein Dreieck und seine sektorenbezogene Flächeninhaltsfunktion . . . . .	59
2.20	Ein verrauschtes Dreieck und seine sektorenbezogene Flächeninhaltsfunktion . . . . .	60

2.21	Rekonstruktion eines Polygons aus der $FIF_{lin}$ : erster Sektor, letzter Sektor	64
2.22	Zwei Achtecke mit gleicher $FIF_{lin}$ -Darstellung	65
2.23	Maximumsmetrik für Funktionen	67
2.24	Überstrichene Fläche in Abhängigkeit von der Bogenlänge	68
2.25	Zwei Polygone mit derselben $FIF$ als Funktion der Bogenlänge	69
2.26	Einseitiges Rauschen eines Polygons	69
3.1	Turning-Funktion eines einfachen Polygons	72
3.2	Turning-Funktion eines einseitig verrauschten Polygons mit Bogenlängenargument und mit Winkelargument	76
3.3	Stringrepräsentation eines einfachen Polygons	78
3.4	Darstellung zweier Strings in einem Graphen. Die Kantengewichte sind Funktionswerte der Kostenfunktionen	82
3.5	Segmentierungsphänomene: Aufspaltung einer Strecke und eines Winkels durch Meßfehler	85
4.1	Ermittlung kollinearere Kanten in der Praxis	89
4.2	Sichtbarkeitspolygonveränderung bei Bewegungen innerhalb einer großen Sichtbarkeitszelle	90
4.3	Der Schwerpunkt ist kein Referenzpunkt für Polygon-Matching	92
4.4	Sichtbarkeitspolygone mit verschiedenartigen Ausbuchtungen	94
4.5	PKF und $FIF$ für ein nicht sternförmiges Polygon	96
A.1	„Rotation“	100
A.2	„Beispielszene“	101
A.3	„skaliert“	101
A.4	„kollinear“	102
A.5	„Kerbe“	102
A.6	„leichtes Rauschen“	103
A.7	„uneinheitliches Rauschen“	103
A.8	„einheitliches Rauschen“	104
A.9	Sortierung gemäß PKF-Metrik	104
A.10	Sortierung gemäß $PKF_{lin}$ -Metrik	105
A.11	Sortierung gemäß $FIF$ -Metrik	105
A.12	Sortierung gemäß $FIF_{lin}$ -Metrik	105
B.1	Zur Definition Verdeckungsecke	108

# Tabellenverzeichnis

2.1	Unterschiedliche Polygondarstellungen . . . . .	19
2.2	Laufzeit der Berechnungen zur PKF-Ähnlichkeitsfunktion . . . . .	38
2.3	Vergleich verschiedener Ähnlichkeitsfunktionen in der Übersicht . . . . .	62
A.1	Vergleichswerte für Polygonpaare . . . . .	100
A.2	Distanzwerte gemäß PKF-Metrik . . . . .	104
A.3	Distanzwerte gemäß $\text{PKF}_{\text{lin}}$ -Metrik . . . . .	105
A.4	Distanzwerte gemäß FIF-Metrik . . . . .	105
A.5	Distanzwerte gemäß $\text{FIF}_{\text{lin}}$ -Metrik . . . . .	105



# Kapitel 1

## Einführung

### 1.1 Grundbegriffe

Die Sichtbarkeit von Objekten ist überall da von Bedeutung, wo es um das Wiedererkennen einer bekannten Umgebung (Szene) oder eines Teils davon geht. Diese ist zumeist in Form einer (zweidimensionalen) Karte gegeben, die alle Merkmale des Geländes in vereinfachter geometrischer Form darstellt. Dazu gehören zwei Dinge:

- a) der Rand der Szene. Dabei handelt es sich um eine Kurve im Jordanschen Sinne, d.h. um einen geschlossenen, überschneidungsfreien, stetigen Kantenzug.
- b) die Hindernisse der Szene, d.h. die Repräsentationen aller Objekte, die in einer realen Umgebung als Gegenstände vorstellbar sind. Auch hier nimmt man zunächst Jordankurven an.

Häufig sind diese Kurven in Form geometrischer Figuren gegeben, z.B. Kreise für Säulen, Rechtecke für Tische etc.

Die Hindernisse sind sämtlich vollständig im Inneren<sup>1</sup> der Randkurve enthalten. Darüberhinaus sollen keine zwei Hindernisse einen inneren Punkt gemeinsam haben, d.h. die Hinderniskurven sollen untereinander überschneidungsfrei sein und sich nicht gegenseitig enthalten<sup>2</sup>. Im folgenden wird der Begriff *Objekt* als Oberbegriff für Rand(objekt) und Hindernis(objekt) verwendet.

Die Unterscheidung zwischen Rand und Hindernissen ist notwendig, da diese Objekte auf unterschiedliche Art und Weise die Szene in zwei Zonen einteilen, und zwar in den *Freiraum* (*free space*) – das ist das Innere des Randes geschnitten mit dem Äußeren eines jeden Hindernisses – und den *Hindernisraum* (*obstacle space*) – das ist die Vereinigung des Inneren aller Hindernisse<sup>3</sup>. Beide Zonen sind also beschränkt; der Hindernisraum ist außerdem unzusammenhängend, falls mindestens zwei Hindernisse vorhanden sind.

Die Auswahl von Algorithmen, die auf solchen Szenen arbeiten, hängt in erster Linie von der Art der Objekte ab. Da der praktische Hintergrund auf dem Gebiet der Roboterlokalisierung vor allem Innenräume sind, z.B. in Fabrikgebäuden, Krankenhäusern usw.,

---

<sup>1</sup>Der Begriff „Inneres“ ist hier nicht im topologisch-mengentheoretischen, sondern im anschaulich-geometrischen Sinne zu verstehen.

<sup>2</sup>Andernfalls spricht man von „Hindernissen mit Löchern“.

<sup>3</sup>Die Verwendung der Begriffe *Äußeres* und *Inneres* ist wegen der Beschränkung auf Jordankurven durch den *Jordanschen Kurvensatz* legitimiert.

erhofft man sich regelmäßige und mathematisch leicht modellierbare Kurven als Begrenzungen der Objekte. Daher nimmt man zunächst „Primitivflächen“ aus der Geometrie zur Beschreibung von Rand und Hindernissen an. Der überaus größte Anteil bisheriger Ergebnisse bezieht sich auf *polygonale Szenen*, bei denen also als Begrenzungskurven nur (einfache) Polygone zugelassen sind. Die vorliegende Diplomarbeit bezieht sich ebenfalls ausschließlich auf diese Art von Umgebungen. Bild 1.1 zeigt einen solchen Fall, in dem sich zwei rechteckige Hindernispolygone berühren.

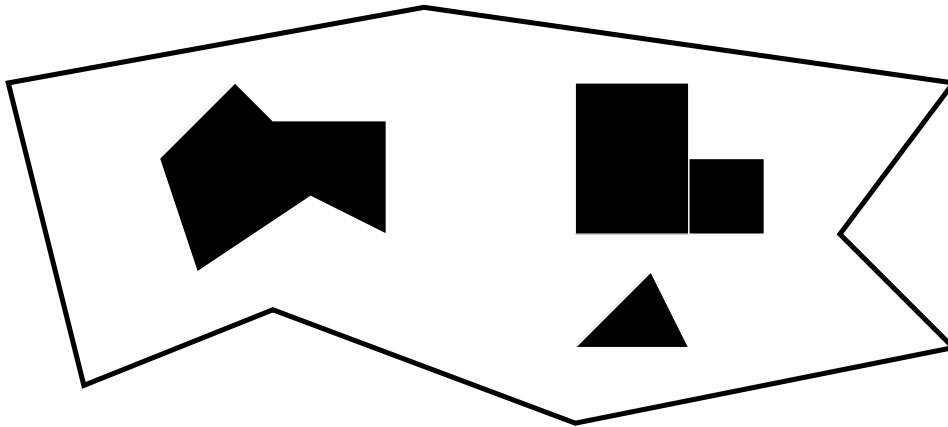


Abbildung 1.1: Eine polygonale Szene. Der Hindernisraum ist schwarz gezeichnet

## 1.2 Ein Ansatz für die Lokalisation

In diesem Abschnitt soll die Beschäftigung mit Distanzfunktionen und Matchingverfahren aus Sicht der Roboterlokalisierung motiviert werden.

Das *Lokalisationsproblem* in der algorithmischen Geometrie ist durch folgende Aufgabenstellung definiert:<sup>4</sup>

Gegeben ist die Karte einer Szene, wie sie in Kapitel 1.1 beschrieben ist. Ein Beobachter (in der Praxis z.B. ein Roboter), der sich im Besitz dieser Karte befindet, hat nur die Information, daß sein aktueller Standort innerhalb des Freiraums dieser Szene liegt. (Dies kann zum Beispiel durch Verlust aller Daten über die bisherige Positionierung geschehen sein, wie etwa nach einem Stromausfall oder einem Versagen der Sensorik.) Er soll nun in einer ersten Phase seinen Standort in der Karte wiedererkennen, ohne sich zu bewegen, d.h. ausschließlich anhand des aktuell sichtbaren Bereiches. Wie wir später im Abschnitt 1.2.2 noch sehen werden, ist das jedoch in vielen Situationen nicht eindeutig möglich. Daher schließt sich im Fall mehrerer *hypothetischer* Lösungen der ersten Phase eine zweite an, in der sich der Beobachter gezielt auf Teile der Karte zubewegen soll, die mehr Sichtbarkeitsinformation als der augenblickliche Standort liefern und daher die Mehrdeutigkeiten aufzulösen helfen.

Dieser Ansatz ist vollständig in dem Sinne, daß nach genügend langen zurückgelegten Erkundungswegen stets eine eindeutige Lösung angegeben werden kann: Hat der Beob-

<sup>4</sup>Ausführliche Beschreibungen finden sich zum Beispiel in [GMR95], [Klb94] und [DRW95].

achter im Extremfall die gesamte Szene abgelaufen, so kennt er die exakte Position in der Karte.

In dieser Diplomarbeit wird es ausschließlich um die erste Phase gehen. Guibas, Motwani und Raghavan stellen dazu in ihrer Arbeit [GMR95] ein Verfahren vor, das Ausgangspunkt für die Beschäftigung mit dem Thema war. Daher soll es im folgenden in leicht vereinfachter Form vorgestellt werden.

## 1.2.1 Sichtbarkeitspolygone und -skelette

### 1.2.1.1 Begriffsklärungen

Zunächst wollen wir den Begriff „sichtbarer Bereich“ etwas näher beleuchten. Darunter versteht man die Menge aller Punkte, die vom gegebenen Betrachterstandpunkt aus sichtbar sind, d.h. für die die Verbindungsstrecke zum Betrachter keine Szenenkante schneidet. Für den Fall, daß alle Objekte der Szene Polygone sind, ergibt sich ein einfacher und wichtiger Zusammenhang:

**Lemma 1.1** *Für eine polygonale Szene und einen beliebigen Betrachterstandpunkt innerhalb des Freiraums ist die Fläche aller sichtbaren Punkte ein Polygon.*

Zum Beweis überprüft man, daß diese Fläche beschränkt, zusammenhängend und geradlinig begrenzt ist.

**Definition 1.2 (Sichtbarkeitspolygon)** *Für eine polygonale Szene und einen beliebigen Betrachterstandpunkt  $p$  innerhalb des Freiraums heißt der sichtbare Bereich **Sichtbarkeitspolygon** von  $p$  und wird mit  $\mathcal{V}(p)$  bezeichnet.*

Daher können auf den Sichtbereich eines Beobachters, der ja zum Wiedererkennen des Standortes innerhalb der Karte dienen soll, Polygonalgorithmen angewendet werden. Abbildung 1.2 zeigt ein Sichtbarkeitspolygon für die Szene aus Bild 1.1. Die Position des Beobachters ist durch einen kleinen Kreis in der Mitte des Bildes gekennzeichnet.

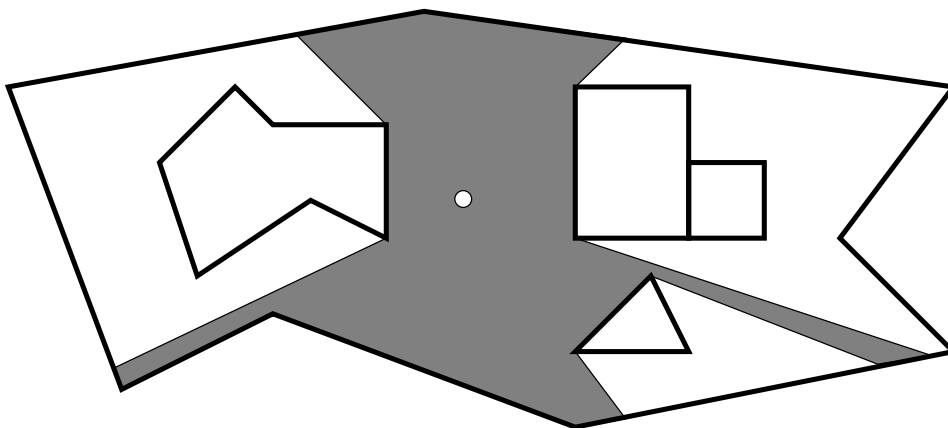


Abbildung 1.2: Das Sichtbarkeitspolygon eines Betrachterstandpunktes

Wie im nächsten Abschnitt beschrieben ist, wird zur Lösung des Lokalisationsproblems noch eine weitere Grundstruktur benötigt: Nach [GMR95] lassen sich die Kanten des Sichtbarkeitspolygons in zwei Klassen einteilen:

- a) die echten Kanten. Sie sind (Teile von) Kanten der Szene, liegen also entweder auf Kanten von Hindernispolygonen oder auf Randpolygonkanten.
- b) die Scheinkanten. Sie entstehen durch Verlängerung eines Sichtstrahles vom Beobachter durch eine verdeckende Szenenecke, bis dieser Strahl auf eine Szenenkante trifft. Der Auftreffpunkt mit der Kante heißt *Scheinecke*.

Der Beobachterstandpunkt ist also der gemeinsame Schnittpunkt aller Geraden entlang der Scheinkanten. In Abbildung 1.3(a) sind die Scheinkanten des Sichtbarkeitspolygons gepunktet gezeichnet, die Scheinecken sind von einem Kreis umschlossen.

Scheinecken haben die Eigenschaft, sich auch bei nur geringer Veränderung des Standorts sofort zu verschieben. Echte Ecken (Szenenecken) hingegen verändern nicht ihre Lage in der Karte; sie können nur verschwinden oder neu sichtbar werden. Letzteres geschieht nicht kontinuierlich, sondern lediglich beim Überschreiten von *Verdeckungslinien*, die von verdeckenden Szenenecken induziert werden, und davon gibt es nur endlich viele. Solange die Beobachterposition in einem Bereich variiert, in dem keine solche Linie überschritten wird, sieht der Beobachter immer die gleichen Szenenecken. Ebenso bleibt die angulare Ordnung dieser Ecken erhalten. Daher liegt es nahe, diese Eckenfolge als Polygon zu interpretieren, welches wir *Skelettpolygon* des Beobachterstandorts nennen wollen. Seine Eckpunkte sind also alle Ecken des Sichtbarkeitspolygons außer die Scheinecken, die durch neue, *künstliche* Kanten überbrückt werden.

Bild 1.3(b) zeigt das Skelettpolygon zum nebenstehenden Sichtbarkeitspolygon. Die künstlichen Kanten sind gestrichelt gezeichnet. Die in Abbildung (a) daneben eingekreisten Scheinecken sind im Skelett nicht mehr vorhanden.

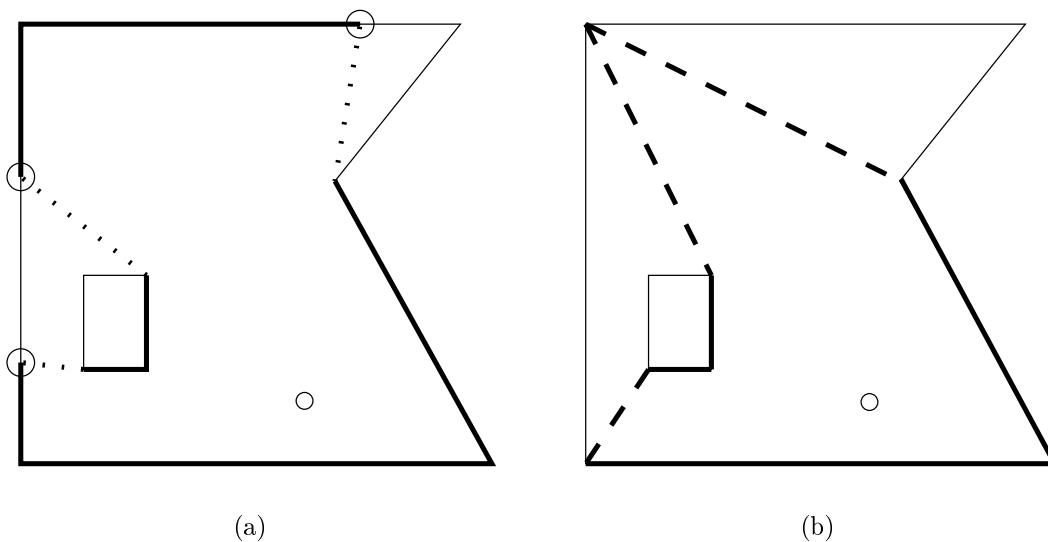


Abbildung 1.3: Sichtbarkeitspolygon (a) und zugehöriges Skelett (b) (jeweils fett)

Nun ist aber von der Szenenkante, auf der eine Scheinecke liegt, immerhin ein Stück sichtbar, so daß das vollständige Weglassen dieser Kante einen Informationsverlust darstellen würde. Daher versieht man jede künstliche Kante mit der Gleichung der Geraden, entlang der die dahinterliegende Szenenkante verläuft. Diese Zusatzinformation heißt *Label*

der künstlichen Kante. Sie verändert sich ebenfalls nicht, solange keine Verdeckungslinie überschritten wird.

Die genannten Bezeichnungen lassen sich zusammenfassen zu der

**Definition 1.3 (Sichtbarkeitsskelett)** *Gegeben seien eine polygonale Szene und das Sichtbarkeitspolygon  $\mathcal{V}(p)$  zu einem Beobachterstandort  $p$ . Das **Sichtbarkeitsskelett** (zu dem vorgegebenen Sichtbarkeitspolygon bzw. dem Standort) besteht aus dem Skelettpolygon, in dem jede künstliche Kante mit ihrem Label in Form einer Geradengleichung versehen ist.*

Die Bedeutung all dieser Strukturen wird in Kapitel 1.2.2 deutlich, in dem ein Lokalisationsalgorithmus vorgestellt wird. In vorliegender Diplomarbeit werden allerdings Polygone im Vordergrund stehen, so daß wir uns im folgenden auf den polygonalen Anteil des Skeletts beschränken werden und daher oft nur das Skelettpolygon meinen, wenn vom Sichtbarkeitsskelett die Rede ist.

### 1.2.1.2 Eigenschaften

Im zweiten Teil dieses Abschnitts werden zwei wichtige geometrische Charakteristiken von Sichtbarkeitspolygon und -skelett aufgeführt, die bei der späteren Behandlung zum Teil von Bedeutung sind.

Zunächst zeigen die Abbildungen, daß weder das Polygon noch das Skelett konvex zu sein brauchen. Vielmehr gilt zunächst folgender Zusammenhang:

**Satz 1.4** *Für Szenen ohne Hindernisse, d.h. mit einem alleinigen Randpolygon  $\mathcal{P}$ , gilt: Das Sichtbarkeitspolygon  $\mathcal{V}(p)$  eines Beobachterstandortes  $p$  ist genau dann konvex, wenn  $\mathcal{P}$  selbst konvex ist.*

Insbesondere hängt also die Konvexität der Sichtbarkeitspolygone nicht von der Auswahl eines Standortes ab.

**Beweis:** Dazu benötigen wir die aus der Geometrie bekannte Tatsache, daß ein Polygon genau dann konvex ist, wenn jede seiner Ecken konvex ist, d.h. der Innenwinkel an jeder Ecke höchstens  $180^\circ$  beträgt.

- a) „ $\Leftarrow$ “: Ist  $\mathcal{P}$  konvex, so ist von  $p$  aus *jeder* Punkt sichtbar, d.h. es ist  $\mathcal{V}(p) = \mathcal{P}$ . Also ist auch  $\mathcal{V}(p)$  konvex.
- b) „ $\Rightarrow$ “: Sei nun  $\mathcal{P}$  nicht konvex. Ist  $\mathcal{V}(p) = \mathcal{P}$ , so ist auch  $\mathcal{V}(p)$  nicht konvex, und der Beweis ist erbracht. Gelte nun  $\mathcal{V}(p) \neq \mathcal{P}$ . wegen  $\mathcal{V}(p) \subset \mathcal{P}$  (betrachtet als Punktmenge der Polygonflächen) gibt es eine Kante  $e$  von  $\mathcal{V}(p)$ , an der sich  $\mathcal{V}(p)$  und  $\mathcal{P} \setminus \mathcal{V}(p)$  berühren. Diese Kante verläuft also im Innern von  $\mathcal{P}$  und trennt lokal sichtbaren und unsichtbaren Bereich.

Dies ist nur für die Scheinkanten des Sichtbarkeitspolygons möglich, da echte Kanten – der andere Kantentyp – nicht im Innern der Szene verlaufen, sondern auf dem Rand. Scheinkanten haben stets genau eine verdeckende Szenenecke als Endpunkt (siehe Beginn dieses Abschnitts). Eine solche Ecke ist jedoch konkav (nichtkonvex): Nach außen gerichtete Ecken können nicht verdecken, denn der Sichtstrahl von  $p$  durch diese Ecke würde das Innere von  $\mathcal{P}$  verlassen.

Diese Konkavecke  $u$  (in  $\mathcal{P}$ ) ist auch eine Ecke in  $\mathcal{V}(p)$ . Da von  $p$  aus genau eine der beiden zu  $u$  adjazenten Szenenkanten sichtbar ist (denn  $u$  ist eine Verdeckungsecke), ist  $u$  auch eine Konkavecke in  $\mathcal{V}(p)$ . Somit ist auch  $\mathcal{V}(p)$  nicht konvex.

□

Für Szenen mit mindestens einem Hindernis ist zunächst der Begriff der konvexen/konkaven Ecke zu verallgemeinern:

**Definition 1.5 (Reflexecke)** *Eine Ecke, die zum Randpolygon gehört und konkav ist oder zu einem Hindernispolygon gehört und konvex ist, heißt **Reflexecke** oder **reflexe Ecke**.*

Anschaulich gesprochen, sind Reflexecken genau die Objektecken, die in Richtung des Freiraums zeigen. Nur diese Ecken können für Verdeckungen verantwortlich sein.

Damit ist sicherlich ein Sichtbarkeitspolygon eines Punktes  $p$  genau dann konvex, wenn von  $p$  aus keine Reflexecke der Szene sichtbar ist. Genau die Reflexecken verursachen nämlich konkave Ecken im Sichtbarkeitspolygon und machen dieses somit nichtkonvex.

Die folgende Argumentation über die Konvexität des Sichtbarkeitspolygons folgt der Richtung „ $\implies$ “ von Satz 1.4. Der zugrundeliegende Freiraum für einen Beobachter soll jetzt abkürzend mit  $\mathcal{S}$  bezeichnet werden. Der Fall  $\mathcal{V}(p) = \mathcal{S}$  tritt hier nicht auf, da durch jedes Hindernis einige Punkte der Sicht des Beobachters entzogen werden. Wegen  $\mathcal{V}(p) \subsetneq \mathcal{S}$  existiert wieder eine Kante von  $\mathcal{V}(p)$ , die an einem Ende durch eine Reflexecke begrenzt ist. Diese Ecke verursacht in  $\mathcal{V}(p)$  eine konkave Ecke und ist verantwortlich für dessen Nicht-Konvexität. Zusammenfassend ergibt sich:

**Satz 1.6** *In einer Szene mit mindestens einem Hindernis ist das Sichtbarkeitspolygon eines jeden Beobachterstandortes nichtkonvex.*

Die Eigenschaft der Konvexität steht also nicht zur Verfügung. Hingegen läßt sich anhand der Bilder vermuten und auch aus der Definition unmittelbar ableiten, daß das Sichtbarkeitspolygon sternförmig ist, und es liegt sogar ein Kernpunkt vor: der Beobachterstandort. Bild 1.3(b) legt selbiges auch für das Skelettpolygon nahe, obwohl es in diesem Fall nicht so selbstverständlich ist. Es gilt jedoch das allgemeinere

**Lemma 1.7** *Gegeben seien ein Polygon  $\mathcal{P}$  und ein Punkt  $p$  im Innern von  $\mathcal{P}$ . Der Punkt  $p$  ist genau dann ein Kernpunkt von  $\mathcal{P}$ , wenn alle Eckpunkte des Polygons von  $p$  aus sichtbar sind.*

Im Fall unserer Sichtbarkeitsskelette sind tatsächlich alle Eckpunkte vom Beobachterstandort aus sichtbar, denn die neuen, künstlichen Kanten verdecken keine (vorher) sichtbaren Szenenecken; sie überbrücken allein Scheinecken. Somit verdecken diese Kanten keine Skelettecken, da letztere niemals Scheinecken sind. – Fassen wir also zusammen:

**Satz 1.8** *Für einen beliebigen Beobachterstandort  $q$  innerhalb des Freiraums sind sowohl das Sichtbarkeitspolygon als auch das Skelettpolygon sternförmig, und  $q$  übernimmt in beiden Fällen die Rolle eines Kernpunktes.*

### 1.2.2 Skizze des Algorithmus'

In diesem Teilabschnitt wird der in [GMR95] vorgestellte Lokalisationsalgorithmus für polygonale Szenen grob demonstriert. Während wir bisher von der Position eines Betrachters als *Beobachterstandort* gesprochen haben, so soll in Zukunft meist von einem *Anfragepunkt* die Rede sein, um deutlich zu machen, daß es um die Lösung der Lokalisationsanfrage für einen Roboter geht.

Das Grundprinzip der Lokalisation ist der Vergleich des Sichtbarkeitspolygons des Beobachterstandortes<sup>5</sup> mit allen denkbaren Sichtbarkeitspolygonen der Szene. Wann immer ein Kartenpunkt einen Sichtbereich der gleichen polygonalen Form induziert, ist eine Lösung gefunden. Wie schon weiter vorne angemerkt, ist auf diese Weise der Standort des Betrachters nicht immer eindeutig bestimmbar: Man stelle sich eine Galerie mit vielen identisch aussehenden Seitenräumen und einigen Vitrinen im Korridor vor (Abbildung 1.4). Ein Beobachter kann die beiden äußeren der Seitenräume unter Umständen nicht voneinander unterscheiden, falls er sich selbst in einem davon befindet, weil sie die exakt gleiche Sicht in den Korridor bieten. Er kann lediglich die Lage relativ zu den Seitenräumen diagnostizieren<sup>6</sup> – Alle Lösungsstandorte des Algorithmus' haben also dasselbe Sichtbarkeitspolygon.

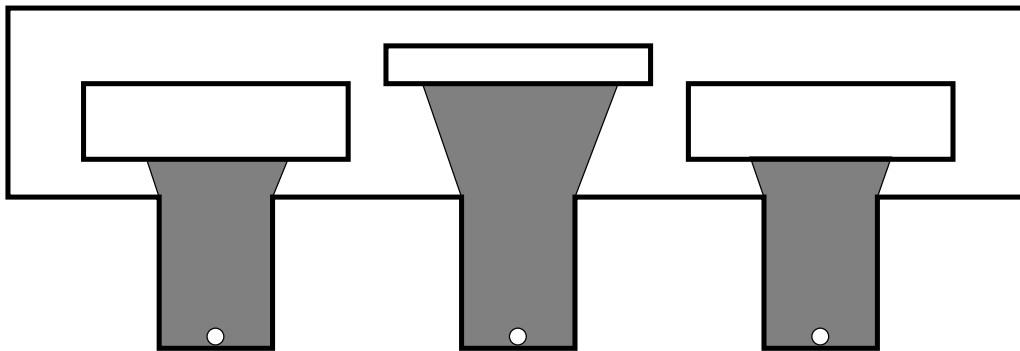


Abbildung 1.4: Eine Szene mit Mehrdeutigkeiten

Nun können freilich nicht alle Kartenstandorte auf das „passende“ Sichtbarkeitspolygon abgetestet werden: Verändert man die Position in der Szene, so ändert sich sofort auch das Sichtbarkeitspolygon; es gibt also unendlich viele verschiedene solche Polygone – die Suche muß auf geeignete Weise diskretisiert werden.

Die Idee hierfür wurde schon im Abschnitt über die Sichtbarkeitsskelette geliefert: Bei einer Standortvariation ändern sich das Skelettpolygon und die Kantenlabel nur, falls eine Verdeckungslinie überschritten wird. Deren Anzahl ist von der Größenordnung  $\mathcal{O}(n^2)$ , wenn  $n$  die Anzahl aller Objektecken der Szene ist.

Zeichnet man also alle Verdeckungslinien in die Karte ein, erhält man eine Zerlegung in Zellen mit der wichtigen Eigenschaft, daß innerhalb einer Zelle das Sichtbarkeitsskelett für alle Standorte dasselbe ist. Man nennt diese Zellen *Sichtbarkeitszellen*. In Bild 1.5 ist

<sup>5</sup>In der Praxis könnte dieses Polygon von einer Sensorik des Roboters geliefert worden sein, die auf Laserscanner-Daten zurückgreift.

<sup>6</sup>Gleichwohl kann man bei einer „hinreichend unregelmäßigen“ Szene mit einer eindeutigen Antwort auf die Lokalisationsanfrage rechnen.

die Sichtbarkeitszellenzerlegung des mittleren Ausschnitts der Galerie in Abbildung 1.4 angegeben. Die Verdeckungslinien sind dünn eingezeichnet. Die beiden markierten Punkte haben dasselbe Sichtbarkeitsskelett, d.h. insbesondere sind von da aus dieselben Sze-  
nenecken (eingekreist) sichtbar; für jeden Standort einer beliebigen Nachbarzelle sind es entweder mehr oder weniger.

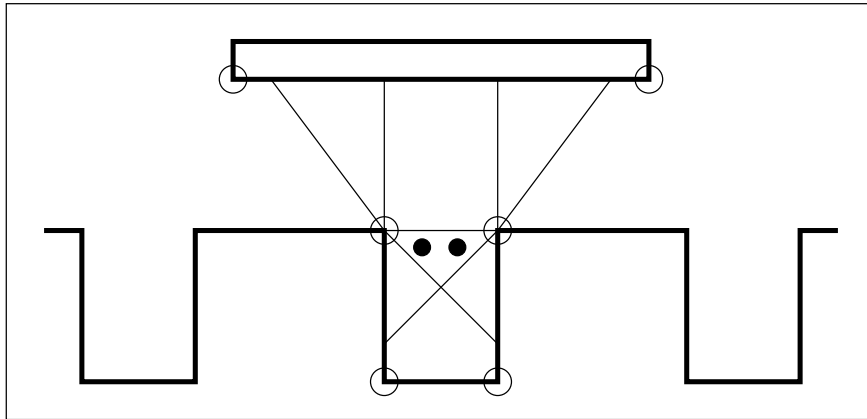


Abbildung 1.5: Eine Zerlegung in Sichtbarkeitszellen

Diese Zellenzerlegung ist die entscheidende Erkenntnis für die geforderte Diskretisierung der Standortsuche in der Karte: Anstatt die Sichtbarkeitspolygone der Kartenpunkte zum Vergleich heranzuziehen, ermittelt man zunächst nur die Zellen, deren (eindeutig bestimmtes) Sichtbarkeitsskelett mit dem Skelett des Beobachterstandorts („Anfrageskelett“) übereinstimmt. Dabei müssen neben den Skelettpolygonen und ihrer Kantentypen (echt/künstlich) auch die Label der künstlichen Kanten identisch sein, und zwar relativ zu einem Bezugspunkt innerhalb des Skeletts als Koordinatenursprung: Die absoluten Koordinaten in einem globalen Koordinatensystem der Karte sind freilich von Standort zu Standort verschieden und machen keinen Sinn, da der Roboter ja nicht „global sieht“, sondern nur relative Lagebeziehungen, wie Entfernungen, ausmachen kann. – In Bild 1.4 ist zum Beispiel der mittlere markierte Standort von den beiden äußeren aufgrund des Labels unterscheidbar: Es ist viel weiter vom Standort entfernt als bei den äußeren Räumen.

Nach der Bestimmung dieser „passenden Zellen“ hat man aber weder irgendwelche Sichtbarkeitspolygone verglichen (die Skelette geben ja noch keinen Aufschluß über den Standort innerhalb der Zelle) noch potentielle Lösungspositionen in der Karte gefunden. Dazu ist folgender letzter Schritt nötig: Der Anfragepunkt  $q$  hat innerhalb des Anfrageskeletts eine feste relative Lage (die wieder mit Hilfe eines Bezugspunktes angegeben werden kann). Hat man eine passende Sichtbarkeitszelle  $\mathcal{C}$  gefunden und bildet das Anfrageskelett zusammen mit dem Anfragepunkt auf das identische Zellskelett ab, so bekommt  $q$  eine Kartenposition  $p$  innerhalb des Zellskeletts zugeordnet. Ob aber  $p$  sogar innerhalb der Zelle  $\mathcal{C}$  liegt, ist offen und mitverantwortlich dafür, ob mit  $p$  eine potentielle Lösung gefunden ist oder nicht. Darüber gibt folgender Satz Auskunft, der in etwas abgewandelter Form auch in [GMR95] zu finden ist:

**Satz 1.9** *Gegeben seien ein Anfragepunkt  $q$  und eine Sichtbarkeitszelle  $\mathcal{C}$  mit passendem Skelett. Sei weiter  $p$  derjenige Kartenpunkt, auf den  $q$  bei der Einbettung des Anfrageskeletts an die Position des Skeletts der Zelle abgebildet wird. Dann gilt:  $p$  und  $q$  erzeugen*

genau dann dasselbe Sichtbarkeitspolygon (und  $p$  ist eine Lösung der ersten Phase der Roboterlokalisierung), wenn die Zelle, in der  $p$  liegt, ebenfalls ein passendes Skelett induziert.

Das ist insbesondere dann der Fall, wenn  $p$  in  $\mathcal{C}$  liegt. Gehört  $p$  jedoch zu einer anderen Zelle  $\mathcal{C}'$ , so wird erst bei der Behandlung dieser Zelle  $\mathcal{C}'$  darüber entschieden, ob  $p$  eine Lösung ist.

### 1.2.3 Anwendungsaspekte

Das Verfahren von Guibas, Motwani und Raghavan basiert auf einem Vergleich von Sichtbarkeits skeletten der Sichtbarkeitszellen (Zellskelette) mit dem Skelett der Anfrageposition (Anfrageskelett). Während die Zellskelette im Preprocessing exakt bestimmt werden, muß das Anfrageskelett mit geeigneten technischen Mitteln bei der Query gemessen werden. Verwendet man dafür einen Laserscanner, so treten zwei Arten von Problemen auf:

**Meßungenauigkeiten:** Die zurückgelieferten Entfernungswerte vom Roboter bis zum Auftreffpunkt des Laserstrahls sind fehlerbehaftet.

**Diskretisierung:** Auf der Peripherie des (theoretischen) Sichtbarkeitspolygons werden nur endlich viele Punkte vom Laserstrahl getroffen, da die Strahlen stets in festen Winkelinkrementen ausgesendet werden. Diese Winkelgröße und die Entfernung des Standorts vom Hindernis bestimmen die Dichtheit der Meßpunkte und damit die Feinheit der Messung. Dies ist in Bild 1.6(a) zu erkennen. Die Länge der eingezeichneten Strahlen soll die gemessene fehlerbehaftete Entfernung repräsentieren (die auch größer sein kann als die exakte Distanz).

Diese beiden Punkte machen einen Einsatz des vorgestellten Lokalisationsverfahrens in der Praxis unter *exaktem* Skelettvergleich unmöglich (siehe Abbildung 1.6(b)). Um die Diskretisierung zu umgehen, kann man zwar die Entfernungswerte in Punkte umrechnen und diese dann (gegebenenfalls mittels Geradenapproximation) zu einem Polygonzug verbinden. Trotzdem bleibt aber die Gestalt der Szene im Winkelintervall *zwischen* zwei ausgesendeten Laserstrahlen verborgen. Das aus dem Scan extrahierbare Sichtbarkeitspolygon ist daher stets nur eine Näherung und läßt sich in der Karte nicht exakt wiederfinden.

Aus diesen Gründen kann der Vergleich des Anfrageskeletts mit dem Zellskelett nur näherungsweise durchgeführt werden. Das motiviert den Einsatz einer Distanzfunktion, der ein Scan und ein Skelett übergeben werden und die als Ausgabe eine reelle Zahl liefert, die in einem bestimmten Sinne die Ähnlichkeit der Eingabegrößen beschreibt. Dabei sollte berücksichtigt werden, daß sich der Scan und das Skelett in ihrer Qualität wesentlich unterscheiden, z.B. hinsichtlich der Anzahl der Eckpunkte und der prognostizierten bzw. tatsächlichen Genauigkeit. Als „passend“ werden dann diejenigen Sichtbarkeitszellen bezeichnet, für die der Skelett-Scan-Distanzwert unter einer bestimmten Schranke liegt oder zu den  $k$  kleinsten gehört (für ein geeignetes  $k$ ).

Im Zusammenhang mit den Laser-Unzulänglichkeiten stellt sich die Frage, ob es Sinn macht, im Preprocessing für jede Sichtbarkeitszelle das Skelett exakt zu berechnen und auch so zu speichern. Dabei entstehen bei komplexen Szenen unter Umständen sehr detaillierte Skelette, die vom Laser so in keinem Fall wahrgenommen werden können. Es könnte also sinnvoll sein, in einem weiteren Preprocessing-Schritt Skelette zu vereinfachen oder

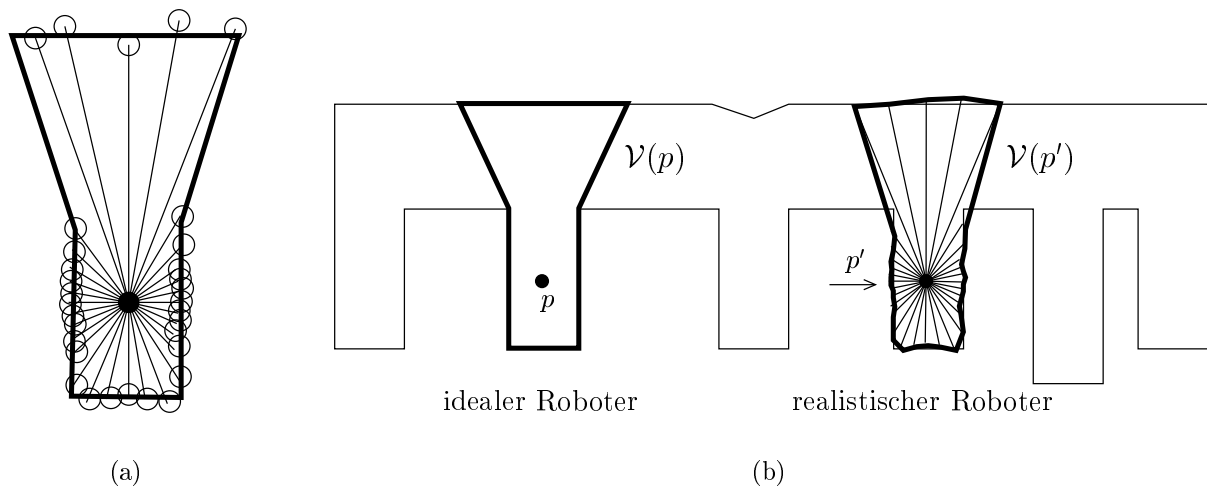


Abbildung 1.6: Ein Scanbild eines Szenenausschnitts (a); ein Vergleich ideales – reales Sichtbarkeitspolygon (b)

sogar benachbarte Sichtbarkeitszellen zu verschmelzen, wenn sich ihre Skelette nur wenig unterscheiden. Dies ist möglich, weil durch den Einsatz von Distanzfunktionen nicht die Kongruenz, sondern nur die (intuitive) Ähnlichkeit von Anfrage- und einem Zellskelett relativ zu anderen Zellskeletten geprüft wird. In vorliegender Arbeit werden derartige Kartenvereinfachungen jedoch nicht behandelt; es wird immer von den exakten Daten des Preprocessings ausgegangen.

Doch die Ähnlichkeitsfunktionen zwischen Scan und Skelett bzw. allgemein zwischen Polygonen müssen noch weitere Aufgaben übernehmen, die sich gut damit verbinden lassen. Sie betreffen weniger die Gestalt, sondern mehr die *Lage* der Polygone und Skelette:

Der Roboter kann mit einem Laser-Radar nur relative Lagebeziehungen bestimmen. Man kann daher nicht erwarten, daß die Eckenkoordinaten des Anfrageskeletts selbst im Fall der Kongruenz mit den (absoluten) Koordinaten eines Zellskeletts übereinstimmen. Vielmehr geht es im Algorithmus um eine korrekte Einbettung des Anfrageskeletts in die Karte (siehe z.B. Satz 1.9). Diese Vorstellung entspricht formal dem (beliebigen) Verschieben eines Polygons, so daß die Ähnlichkeitsfunktion *translationsinvariant* sein muß.

Ein Polygon zeichnet sich aber auch durch eine Orientierung aus: Möglicherweise unterscheidet sich ein Zellskelett vom kongruenten Anfrageskelett nur durch eine Rotation. Ob diese zu ignorieren ist, d.h. ob auch beliebige Drehungen der Eingabepolygone ohne Einfluß auf das Ergebnis des Vergleichs sein sollen, hängt davon ab, ob der Roboter einen Kompaß besitzt. Falls ja, so kann man festlegen, daß nach jeder Scan-Aufnahme das Polygon um den Winkel gedreht wird, um den die Blickrichtung des Roboters z.B. von Nord abweicht. Die Zellskelette aus dem Preprocessing müssen ebenfalls so gespeichert sein, als wären sie vom Roboter mit dieser Blickrichtung aufgenommen worden. Erst nach dieser Einnordung des Anfrageskeletts wird es an die Vergleichsfunktion übergeben. Eine Sichtbarkeitszelle darf nur dann als passend bezeichnet werden, wenn ihr Skelett ausschließlich durch eine Translation auf das Anfrageskelett gut genug (in oben beschriebenem Sinne) gematcht werden kann: Potentielle Lösungsskelette müssen zum Anfrageskelett gleichge-

richtet sein.

Ohne Kompaß ist die Einnordung nicht möglich. Ein kongruentes Zellskelett mit anderer Orientierung als das Anfrageskelett muß als Lösung zugelassen sein. Von der Vergleichsfunktion verlangt man *Rotationsinvarianz*. Die Konsequenz ist, daß durch die fehlende *Kompaßvoraussetzung* im allgemeinen mehr Standortkandidaten vom Lokalisationsalgorithmus zurückgeliefert werden.

*Skalierungen* (zentrische Streckungen) von Polygonen sollen nach Möglichkeit von der Vergleichsfunktion in jedem Fall berücksichtigt werden, da ein Laserscanner Entfernungen messen kann. Unterschiedlich „große“ Polygone und Skelette sind von ihm also immer unterscheidbar.

Zusammenfassend sind also folgende Eigenschaften einer Vergleichsfunktion gefordert:

- ▷ Geringe Empfindlichkeit gegenüber Meßfehlern und verrauschten Eingabedaten („Noise“)
- ▷ Translationsinvarianz
- ▷ Rotationsinvarianz, falls kein Kompaß zur Verfügung steht
- ▷ Sensitivität gegenüber Skalierungen

Selbst wenn man Distanzfunktionen gefunden hat, die allen diesen Anforderungen genügen, gibt es noch etliche Schwierigkeiten bei der Anwendung des Verfahrens in der Praxis. Dies wird eingehend in Kapitel 4.1 am Ende dieser Arbeit beschrieben, wenn einige Erkenntnisse über Eigenschaften verschiedener Ähnlichkeitsfunktionen vorliegen.

## 1.3 Morphologische Verfahren im Überblick

Im Abschnitt 1.2.3 wurden einige Probleme genannt, die im Zusammenhang mit praktischen Anwendungen des zuvor vorgestellten Ansatzes auftreten. Für die Bewältigung derartiger Schwierigkeiten steht eine Reihe von Verfahren zur Verfügung, die in diesem Abschnitt in einer Übersicht vorgestellt werden sollen. Ähnlichkeitsmaße bilden die Grundlage fast aller dieser Verfahren. Deshalb wird zuvor ihre Definition etwas ausführlicher behandelt.

### 1.3.1 Geometrische Transformationen

Zu Anfang ist genau festzulegen, was für Manipulationen auf welchen Objekten zugelassen sind:

*Objekte* können allgemein auf zwei Arten angegeben sein: einmal in einer unstrukturierten Form als Punktmengen, zum Beispiel

$$\{x \in \mathbb{R}^2 : d_2(x, p) \leq 50\}$$

für irgendeinen Punkt  $p$ . Normalerweise liefern Scanner und andere mit „Sinnesorganen“ ausgestattete technische Geräte Teile solcher Punktmengen (z.B. die Auftreffpunkte eines Laserstrahls) als Beschreibung der Umwelt, die sie wahrgenommen haben. Aufgabe eines *Merkmalsinterpreters* oder *-extrahierers* ist es dann, diese Meßpunktewolken in die zweite Form der Beschreibung von Objekten zu überführen: in eine strukturierte Form, in der Objekte nicht ausschließlich quantitativ, sondern auch qualitativ durch Angabe ihrer Eigenschaften beschrieben werden. Zum Beispiel von oben paßt die Angabe

*Kreis um  $p$  mit Radius 50.*

Solche Angaben sind vor allem geometrische Formen, wie Polygone oder Ellipsen, zusammengesetzte Figuren, Kurven stetiger Funktionen u.v.m.

Gleichwohl gibt es für jedes geometrische Objekt eine Darstellung als Punktmenge. Das ist wichtig für die theoretische Behandlung von Objekten, da viele Aussagen Formulierungen über Punktmengen sind.

*Transformationen* sind ganz allgemein Abbildungen  $t: \mathcal{O} \rightarrow \mathcal{O}$ , wobei  $\mathcal{O}$  für die oben beschriebene zulässige Menge von Objekten steht. Man kann sie einteilen anhand der Art und Weise, wie sie auf Objekte wirken:

**Definition 1.10 (Euklidische Transformation)** *Transformationen, unter denen das Bild zum Urbild kongruent ist, heißen **Euklidische** oder **starre** Transformationen.*

Euklidische Transformationen verändern also ausschließlich die Lage eines Objektes. Das trifft zum Beispiel zu für Translationen (Verschiebungen), Rotationen (Drehungen), Reflexionen (Spiegelungen) oder sämtliche Kombinationen aus diesen Grundoperationen. Im Gegensatz dazu verändert eine Skalierung (d.h. eine zentrische Streckung) die „Größe“ eines Objektes. Für ein Polygon beläßt sie jedoch die Verhältnisse der Seiten zueinander sowie sämtliche Innenwinkel unverändert. Man erhält nach der Anwendung einer solchen Operation also *ähnliche* Objekte und verallgemeinert daher obenstehende Definition zu:

**Definition 1.11 (Ähnlichkeitstransformation)** *Transformationen, die ein Objekt auf ein (im streng geometrischen Sinne) ähnliches Objekt abbilden, heißen **Ähnlichkeitstransformationen**.*

Sie verändern Lage und Größe eines Objektes, aber – anschaulich gesprochen – nicht dessen Form und sind aus Euklidischen Transformationen und Skalierungen zusammengesetzt. Die zugehörigen Abbildungen  $t: \mathcal{O} \rightarrow \mathcal{O}$  lassen sich algebraisch durch Vektoraddition, Skalarmultiplikation und Matrixmultiplikation (Drehmatrizen) ausdrücken.

Viele in der Praxis vorkommende Transformationen besitzen eine nützliche Eigenschaft, die wir desöfteren benötigen werden:

**Definition 1.12 (umkehrbare Transformation)** *Eine Transformation  $t \in \mathcal{T}$  heißt **umkehrbar** (in  $\mathcal{T}$ ), wenn es eine Transformation  $t^{-1} \in \mathcal{T}$  gibt, so daß gilt:*

$$\forall_{A \in \mathcal{O}} t^{-1}(t(A)) \text{ ist identisch mit } A.$$

Das trifft zum Beispiel zu für Rotationen, Translationen und Skalierungen, mit denen wir uns hauptsächlich beschäftigen werden.

### 1.3.2 Distanz- und Ähnlichkeitsfunktionen

Unter einer *Distanzfunktion* verstehen wir eine Abbildung  $d: \mathcal{O}^2 \rightarrow \mathbb{R}$ , die wie üblich jedem Paar  $(A, B)$  von Objekten eine reelle Zahl zuordnet mit den Minimaleigenschaften

$$\forall A, B \in \mathcal{O} : d(A, B) \geq 0 \quad (1.1)$$

$$\forall A \in \mathcal{O} : d(A, A) = 0 \quad (1.2)$$

$$\forall A, B \in \mathcal{O} : d(A, B) = d(B, A) \quad (1.3)$$

Zusammen mit der Identitätseigenschaft (1.4) sowie der Dreiecksungleichung (1.5) wird aus der Distanzfunktion eine Metrik:

$$\forall A, B \in \mathcal{O} : d(A, B) = 0 \implies A \equiv B \quad (1.4)$$

$$\forall A, B, C \in \mathcal{O} : d(A, B) + d(B, C) \geq d(A, C) \quad (1.5)$$

Bereits aus (1.2), (1.3) und (1.5) folgt die (schwache) *Positivität* (1.1), denn

$$\forall A, B \in \mathcal{O} \quad d(A, B) = \frac{1}{2}(d(A, B) + d(B, A)) \geq \frac{1}{2}d(A, A) = 0.$$

Die Formel  $A \equiv B$  in (1.4) weist hin auf den Unterschied zwischen *Identität* von Objekten (das ist die Übereinstimmung ihrer Punktmengen in Koordinaten) und deren *gleiche äußere Form* (Kongruenz, Ähnlichkeit). Er geht auf die Erkenntnis zurück, daß Entfernungen zur Modellierung von Ähnlichkeit notwendig, aber unzureichend sind, da sie Verschiebungen und andere formerhaltende Transformationen in unerwünschter Weise berücksichtigen. Daher wird noch ein zweites Maß benötigt. Zunächst können wir wie folgt die „Aufgabe“ von Distanzfunktionen beschreiben:

**Vereinbarung 1.13 (intuitiv)** *Distanzfunktionen messen ausschließlich Entfernungen von Objekten. Die Formel  $A \equiv B$  bedeutet punktweise Identität der Mengendarstellungen von  $A$  und  $B$ , und wir sagen:  $A$  und  $B$  sind identisch.*

So kann zum Beispiel für eine umkehrbare Transformation  $t$  geschrieben werden (vgl. Definition 1.12):  $t^{-1}(t(A)) \equiv A$ .

Die Figuren in Abbildung 1.7(a) haben einen von Null verschiedenen und mehr oder weniger deutlichen Abstand, sind aber kongruent.

Wie kann nun die offensichtliche Ähnlichkeit der beiden Objekte in Abbildung 1.7(a) beschrieben werden? Auch hier macht die Intuition Vorgaben für spätere Definitionen: Die Frage ist offenbar, ob man durch geeignete Manipulationen an den Objekten diese ineinander überführen kann, so daß sie anschließend *identisch* sind in oben beschriebenen Sinne, also übereinanderliegen. Das hängt in entscheidendem Maße davon ab, welche Manipulationen konkret zugelassen sind. Geben wir uns also eine Menge  $\mathcal{T}$  von Transformationen vor, die auf Objekte zur Feststellung ihrer Ähnlichkeit angewendet werden dürfen. In Bild 1.7(a) ist es allein durch starre Transformationen möglich, die Figuren exakt zur Deckung zu bringen. Danach haben sie also den Abstand Null. Die Polygone in Bild 1.7(b) sind hingegen durch derartige Bewegungen nicht ineinander überführbar; im Fall einer *metrischen* Distanzfunktion wird ihr Abstand immer positiv sein. Als Zusammenfassung halten wir fest:

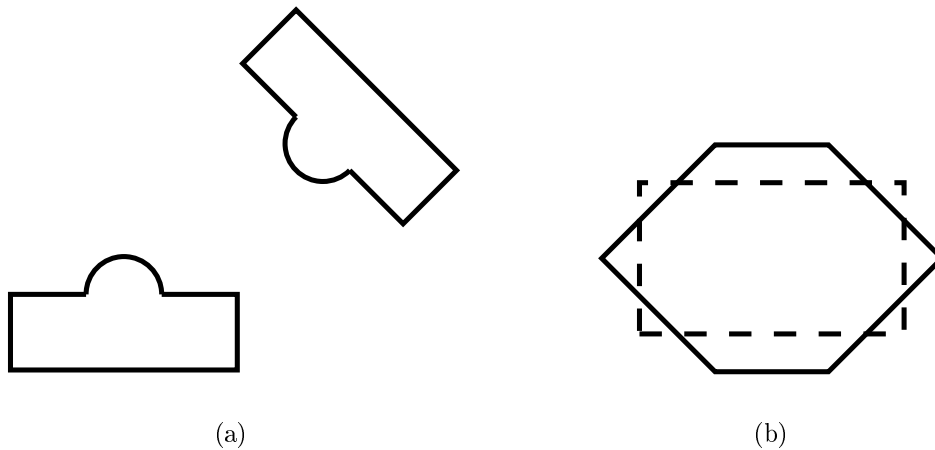


Abbildung 1.7: Sehr ähnliche, aber weit entfernte (a) und recht verschiedene, aber nah beieinanderliegende Objekte (b)

**Definition 1.14 (Ähnlichkeitsfunktion)** Sei  $\mathcal{T}$  eine nichtleere Menge von umkehrbaren Transformationen. Eine Abbildung  $s : \mathcal{O}^2 \rightarrow \mathbb{R}$  heißt **Ähnlichkeitsfunktion** (auf  $\mathcal{O}$ ), wenn sie die folgenden Bedingungen erfüllt:

$$\forall A, B \in \mathcal{O} : s(A, B) \geq 0 \quad (1.6)$$

$$\forall A, B \in \mathcal{O} : A \text{ und } B \text{ sind durch Transformationen} \quad (1.7)$$

$$\text{aus } \mathcal{T} \text{ ineinander überführbar} \implies s(A, B) = 0$$

$$\forall A, B \in \mathcal{O} : s(A, B) = s(B, A) \quad (1.8)$$

Da die Umkehrbarkeit der Transformationen aus  $\mathcal{T}$  gefordert ist, ist es äquivalent, ob  $A$  vermöge der Operationen in  $\mathcal{T}$  in  $B$  überführbar ist oder umgekehrt. Das rechtfertigt die Schreibweise in (1.7).

Wie bei Distanzfunktionen kann man zusätzlich die Umkehrung von (1.7) sowie eine Dreiecksungleichung fordern; dann könnte man von „metrikähnlichen Eigenschaften“ der Ähnlichkeitsfunktion sprechen.

Eine spezielle Form von Ähnlichkeitsfunktionen ist folgendermaßen gegeben:

**Definition 1.15 (induzierte Ähnlichkeitsfunktion)** Seien  $\mathcal{T}$  eine nichtleere Menge von Transformationen und  $d$  eine Distanzfunktion. Die Abbildung

$$s_d : \mathcal{O}^2 \rightarrow \mathbb{R}, \quad s_d(A, B) := \inf_{t_1, t_2 \in \mathcal{T}} d(t_1(A), t_2(B))$$

heißt die **von  $d$  induzierte Ähnlichkeitsfunktion**.

Da die Menge  $\mathcal{T}$  der Transformationen meist in größerem Kontext vorgegeben ist, soll sie in der Bezeichnung der Ähnlichkeitsfunktion nicht explizit genannt werden.

### Bemerkungen zu Definition 1.15.

1.) Für  $A, B \in \mathcal{O}$  ist die Menge

$$M := \{d(t_1(A), t_2(B)) \in \mathbb{R} : t_1, t_2 \in \mathcal{T}\}$$

nicht leer (da  $\mathcal{T} \neq \emptyset$ ) und durch Null nach unten beschränkt, also existiert das in der Definition gebildete Infimum.

- 2.) Es kann im allgemeinen nicht durch das Minimum ersetzt werden, da  $\min M$  nicht zu existieren braucht: Wählt man (als theoretisches Beispiel)  $\mathcal{O} := \mathbb{R}^2$ ,  $A := (1, 1)$ ,  $B := (-2, -2)$ , für  $d$  den Euklidischen Abstand von Punkten und schließlich  $\mathcal{T}$  als die Menge aller Punktverschiebungen, die nicht die  $y$ -Achse schneiden, die also

$$\forall_{x,y \in \mathbb{R}} \begin{cases} x > 0 & \Rightarrow t(x, y) \in \mathbb{R}_{>0} \times \mathbb{R}, \\ x = 0 & \Rightarrow t(x, y) \in \{0\} \times \mathbb{R} \\ x < 0 & \Rightarrow t(x, y) \in \mathbb{R}_{<0} \times \mathbb{R} \end{cases} \quad \text{und}$$

erfüllen, dann gilt  $\inf M = 0 \notin M$ , da die Punkte  $A$  und  $B$  durch die  $y$ -Achse getrennt sind.

- 3.) Die Abbildung  $s_d$  ist tatsächlich eine Ähnlichkeitsfunktion im Sinne von Definition 1.14, denn die geforderten Eigenschaften (1.6), (1.7) und (1.8) folgen auf triviale Weise aus den Eigenschaften (1.1), (1.2) und (1.3) der Distanzfunktion  $d$ . Das gilt nicht allgemein für die Dreiecksungleichung: Es lassen sich leicht „entartete“ Transformationen wie in Bemerkung 2 definieren, die sie verletzen, obwohl die Distanzfunktion selbst ihr genügt. Gleiches gilt für die Umkehrung von (1.7). Ein möglicher Grund ist wiederum, daß das Minimum der Menge  $M$  nicht existieren braucht und daher  $\inf M \notin M$  gelten kann.
- 4.) In vielen Fällen wird  $s_d$  einfacher in der Form

$$s'_d(A, B) = \inf_{t \in \mathcal{T}} d(t(A), B)$$

definierbar sein. Dies zerstört im allgemeinen jedoch die Symmetrie: Betrachten wir wieder das Beispiel aus Bemerkung 2. Hier gilt:

$$s'_d(A, B) = \inf_{t \in \mathcal{T}} d(t(A), B) = 2, \quad s'_d(B, A) = \inf_{t \in \mathcal{T}} d(t(B), A) = \inf_{t \in \mathcal{T}} d(A, t(B)) = 1.$$

Im übrigen sei gesagt, daß eine intuitive Ähnlichkeitsfunktion ein „Unähnlichkeitsmaß“ sein sollte: Je größer der Wert  $s(A, B)$ , desto unähnlicher werden  $A$  und  $B$  sein. Wir werden trotzdem vom *Ähnlichkeitsmaß* oder kurz von der *Ähnlichkeit*  $s(A, B)$  sprechen, um diese Zahl von einer *Entfernung*  $d(A, B)$  abzugrenzen.

In der Literatur (z.B. in [AG96]) werden solche Ähnlichkeitsabbildungen häufig als *invariant unter den Transformationen*  $\mathcal{T}$  bezeichnet: Solange zwei identische Objekte  $A$  und  $B$  nur vermöge der Operationen in  $\mathcal{T}$  zu Bildern  $A'$  und  $B'$  bewegt, verformt etc. werden, gilt  $s(A', B') = 0$ . In Abschnitt 1.2.3 wurde begründet, daß Funktionen für den Vergleich von Sichtbarkeits skeletten translations- und eventuell auch rotationsinvariant sein sollten.

Der Sinn dieses Teilabschnitts 1.3.2 war es, darauf hinzuweisen, daß der Begriff „Distanz“ in verschiedenen Bedeutungen gebraucht werden kann und daß man bei Diskussionen über Distanz- und Ähnlichkeitsfunktionen stets wissen sollte, ob man gerade von einem (Euklidischen) Abstand oder einer Distanz im Sinne von „Ähnlichkeit“ spricht. Es

hat sich allerdings in der algorithmischen Geometrie nicht durchgesetzt, diese Begriffe sauber zu trennen, sondern sich auf die Eindeutigkeit des Kontextes zu verlassen. Daher wird oft der Begriff „Distanzfunktion“ auch im Sinne einer Ähnlichkeitsfunktion gebraucht und damit die Vereinbarung 1.13 nicht streng eingehalten.

### 1.3.3 Matching

Gegeben sei eine feste Ähnlichkeitsfunktion  $s$ . Aufgabe von *Matchingverfahren* ist es, zu einer Objektmenge  $\mathcal{O}$  und einem Objekt  $A \notin \mathcal{O}$  dasjenige Objekt  $B$  aus  $\mathcal{O}$  zu finden, das  $A$  am ähnlichsten ist. Ebenso interessiert dann das Ähnlichkeitsmaß  $s(A, B)$ , d.h. gesucht ist

$$\min_{B \in \mathcal{O}} s(A, B).$$

Zu Beginn von Abschnitt 1.3.1 wurde erläutert, daß Objekte auf zwei verschiedene Arten angegeben werden können. Demgemäß unterteilt man die Matchingverfahren in zwei Gruppen: Beim *Shape-Matching* liegen Objekte als geometrische Strukturen vor. In diesem Fall können andere Verfahren aus der Geometrie, z.B. Polygonalgorithmen, zu Hilfe genommen werden.

Komplizierter ist das *Point-Pattern-Matching*: Hier hat man es mit Objekten in Form (unstrukturierter) Punktmengen zu tun. Dann kommen oft Sweep-Verfahren, lineare Programme und andere Algorithmen zur Anwendung. Manchmal kann das Matching-Problem auf eine Extremwertaufgabe zurückgeführt werden. Das ist besonders dann der Fall, wenn die gegebene Ähnlichkeitsfunktion von einer Distanzfunktion induziert ist; vgl. Definition 1.15.

Eine Variante des Matchings stellt sich die Aufgabe, zu zwei gegebenen Objekten eine Folge von Ähnlichkeitstransformationen anzugeben, die das eine dem anderen möglichst ähnlich machen. Anschaulich gesprochen, soll das erste Objekt solange verschoben, gedreht oder skaliert werden, bis es so über dem zweiten liegt, daß ihr Abstand im Sinne einer Distanzfunktion möglichst klein ist. Auch hierbei ist der Zusammenhang zu Ähnlichkeitsfunktionen deutlich: Die Frage, wie gut sich Objekte ineinander überführen lassen, ist ein Maß für ihre Ähnlichkeit. Umgekehrt liefern „komfortable“ Ähnlichkeitsfunktionen Hinweise darauf, wie ein möglichst gutes Matchen der Argumente zu erfolgen hat: Betrachten wir wieder die Definition 1.15. Dort werden auf die Objekte  $A$  und  $B$  zuerst Ähnlichkeitstransformationen aus einer Menge  $\mathcal{T}$  angewandt, bevor eine Distanzfunktion ihren verbliebenen geometrischen Abstand mißt. Die Folge jener Transformationen könnte sich auch als Ausgabe eines Matchingalgorithmus eignen. In Kapitel 2 werden Beispiele für diese Gemeinsamkeiten genannt.

Allerdings sind die Anforderungen an Matchingverfahren im allgemeinen größer als die an Ähnlichkeitsfunktionen in folgendem Sinne: Letztgenannte haben lediglich die Verpflichtung, intuitive Ähnlichkeit zu modellieren und dabei möglichst metrikähnliche Eigenschaften zu haben. Das läßt noch viele Freiheiten offen, wie die zahlreichen Ansätze zur Definition von Ähnlichkeitsfunktionen beweisen. Ein Matchingalgorithmus muß hingegen genau angeben, wie das Überführen der Objekte vor sich gehen muß, und dafür gibt es oft nur eine Möglichkeit. Der Output ist also deutlicher vorherbestimmt, während die konkrete Zahl, die zwei Polygonen etc. als Ähnlichkeitsmaß zugewiesen wird, immer noch relativ willkürlich ist.

In dieser Arbeit wird es um Matching-Verfahren gehen, wenn im Rahmen der Roboterlokalisierung ein Anfrageskelett mit den Zellskeletten einer Szene zu vergleichen und möglichst ähnliche Skelette zurückzuliefern sind.

### 1.3.4 Einbettung (Embedding)

Gegeben sei wiederum eine Ähnlichkeitsfunktion  $u$ . Weiterhin liegen eine Szene  $\mathcal{S}$  und ein Anfrageobjekt  $A$  vor. Die Frage ist, ob es in  $\mathcal{S}$  einen Ausschnitt gibt, der  $A$  ähnlich ist, und wo sich dieser gegebenenfalls befindet. Außerdem könnte interessieren, wie groß die Ähnlichkeit mit der *Einbettungsstelle* ist. Diese kann etwa als Teilmenge von  $\mathcal{S}$  angegeben werden, so daß die formale Problemstellung lautet:

$$\text{Bestimme } \min_{S \subset \mathcal{S}} u(A, S).$$

Diese Aufgabe ist auch als „partial Mapping“ bekannt. Sie ist gewissermaßen eine realere Variante des Matchings aus dem vorigen Abschnitt: Eine Umgebung ist nicht von vorneherein geclustert in einzelne Objekte, sondern im schlimmsten Fall als *eine* große Punktmenge gegeben. Handelt es sich jedoch um eine Szene wie in Kapitel 1.1, so kann das Einbettungsproblem auf das des Matchings zurückgeführt werden. Dann brauchen nämlich nur die Einzelobjekte der Szene mit  $A$  verglichen werden. Diese Situation liegt zum Beispiel bei der Anfrage im Lokalisationsalgorithmus vor. Im allgemeinen ist es möglich, daß das Anfrageobjekt zusammengesetzt ist und daher nicht auf ein Objekt, sondern auf eine Kombination aus mehreren Objekten der Szene abzubilden ist.

### 1.3.5 Vereinfachung (Simplification)

Viele Meßgeräte der Praxis, wie Entfernungsmesser oder Scanner, liefern zu detaillierte Daten zurück, die zudem mit Ungenauigkeiten behaftet sind. Andererseits kann es sein, daß reale Strukturen, z.B. im Bauwesen, in idealer Form konstruiert waren und dennoch winzige Unregelmäßigkeiten aufweisen, die etwa ein Laserscanner unerwünschterweise erkennt. Man denke hier beispielsweise an eine Wand, die in einer festen Höhe als gerade Linie wiedergegeben werden soll. In Wirklichkeit erhält man vom Scanner Punkte, die nur näherungsweise auf einer Geraden liegen.

Wie schon früher angedeutet, müssen nun Extraktionsprogramme diese Daten auswerten und dabei in der Regel Vereinfachungen oder Approximationen durchführen. Formal ist ein (kompliziertes) Objekt  $A$  gegeben und ein Katalog von Bedingungen, die bei der Vereinfachung nicht verlorengehen dürfen. Gesucht ist ein „möglichst einfaches“ Objekt  $\tilde{A}$ , das (gerade noch) die Bedingungen erfüllt.

„Möglichst einfach“ könnte bedeuten: weniger Punkte (im Fall einer Punktwolke), weniger Ecken (im Fall eines Polygons), konvex (für eine „fast konvexe“ Figur), weniger Kanten (im Fall eines Graphen) etc. Die gestellten Bedingungen ergeben sich meist aus der konkreten Anwendung bzw. daraus, daß laut zugrundeliegender Theorie nur bestimmte Objekte in Frage kommen. Beispiele dafür sind: Kanten müssen zwischen Hindernissen hindurch verlaufen, Zusammenhangseigenschaft eines Graphen (bei einer Kantenreduzierung) etc.

In der Roboterlokalisierung hat man es mit Vereinfachungen zu tun, wenn, wie in Abschnitt 1.2.3 angedeutet, zu detaillierte (kleine) Sichtbarkeitszellen des Preprocessings

oder auch ein mit Meßpunkten überhäufter Scan auf das wesentliche zu reduzieren sind. Das Ziel ist hierbei eine Verringerung der praktischen Komplexität des Algorithmus'. Aber auch die Reduktion des Sichtbarkeitspolygons eines Kartenpunktes auf das Sichtbarkeits skelett ist ein Beispiel einer (Struktur-)Vereinfachung. Die einzuhaltende Bedingung ist hier, daß die Menge aller sichtbaren Szenenecken nicht verändert wird. Der genaue sichtbare Teil einer teils verdeckten Szenenkante wird vereinfacht zur Gleichung der Geraden, auf der die Kante liegt (*Label*).

Eine formale Spezifikation dieses Problems ist schwierig; Bild 1.8 zeigt statt dessen ein Beispiel.

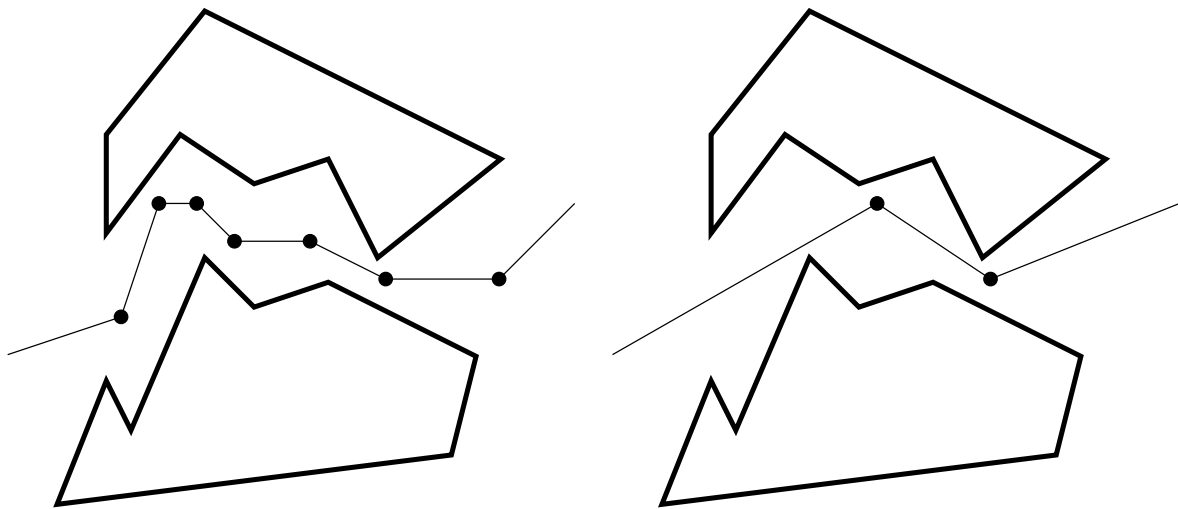


Abbildung 1.8: Vereinfachung eines Weges durch einen Hindernisraum

### 1.3.6 Morphing

Auch beim Morphing („Verwandlung“) ist es das Ziel, aus einem Objekt  $A$  ein Objekt  $\tilde{A}$  zu machen. Im Gegensatz zur Vereinfachung ist aber hier das Zielobjekt  $\tilde{A}$  konkret vorgegeben. Dieses Verfahren kommt zum Beispiel in der Filmindustrie und bei Animationen zur Anwendung: Bewegungen von Trickfiguren sind nichts anderes als Verwandlungen einer Körperhaltung in eine andere bei gleichzeitiger Verschiebung der Figur. Dabei müssen etliche Zwischenzustände angegeben werden, um eine flüssige Bewegung zu erreichen. Formal sind also zwei Objekte  $A$  und  $\tilde{A}$  sowie eine Menge  $\mathcal{T}$  zulässiger Operationen gegeben; gesucht ist eine Folge  $(t_i \in \mathcal{T})_{i=2}^n$  von Transformationen und daraus resultierende Objekte  $C_i$  ( $1 \leq i \leq n$ ) mit

$$C_1 = A, \quad C_i = t_i(C_{i-1}) \quad (2 \leq i \leq n), \quad C_n = \tilde{A}.$$

Man erkennt sofort einen Zusammenhang zu Ähnlichkeitsfunktionen: Je ähnlicher  $A$  und  $\tilde{A}$  sind, desto geringer wird der Aufwand zur Umwandlung von  $A$  in  $\tilde{A}$  sein. Nach [AG96] können daher Morphing-Techniken auch zur Definition von Ähnlichkeitsfunktionen benutzt werden. Das Morphing unterscheidet sich vom Matching aus Kapitel 1.3.3 dadurch, daß die Menge zulässiger Objektmanipulationen a priori nicht beschränkt ist.

# Kapitel 2

## Ähnlichkeitsfunktionen für sternförmige Polygone

In diesem Kapitel werden konkrete Verfahren vorgestellt, wie man die Ähnlichkeit sternförmiger Polygone messen kann. Dabei ist stets auch die Fragestellung zu untersuchen, wie die speziellen Gegebenheiten der Sichtbarkeitspolygone und -skelette bzw. der Roboterlokalisierung überhaupt ausgenutzt werden können, um ein effizientes und qualitativ brauchbares Verfahren zu erhalten.

Das allgemeine Prinzip, Ähnlichkeits- oder Distanzfunktionen für bestimmte (nicht notwendigerweise mathematische) Objekte zu entwerfen, lässt sich wie folgt charakterisieren:

Es existieren bereits Abstandsbegriffe für die verschiedensten Strukturen der Mathematik, so etwa für Zahlen, Punkte der Euklidischen Ebene, für Mengen oder für (stetige) Funktionen. Daher versucht man, andere Objekte, wie zum Beispiel Polygone, in eine Darstellung oben genannter Strukturen zu überführen, um anschließend deren Ähnlichkeit festzustellen.

Einige naive Möglichkeiten, die nur der Veranschaulichung des Prinzips dienen, sind in Tabelle 2.1 aufgeführt. Unter der „Ähnlichkeit“ von Zahlen bzw. Punkten ist dabei ihr Differenzbetrag bzw. ihr Abstand zu verstehen.

Verwendete Struktur	Polygon repräsentiert durch ...
Reelle Zahl	seinen Flächeninhalt
Punkt des $\mathbb{R}^2$	seinen Schwerpunkt
Menge von Punkten des $\mathbb{R}^2$	seine Eckenmenge

Tabelle 2.1: Unterschiedliche Polygodarstellungen

Die Qualität einer auf diese Weise gewonnenen Ähnlichkeitsfunktion für Polygone wird in entscheidendem Maße davon abhängen, wie sehr die gewählte Darstellung auf die Gestalt des Polygons und auf die Belange der Anwendung Rücksicht nimmt. Das ist bei der Repräsentation durch den Flächeninhalt nur sehr eingeschränkt der Fall. Wesentlich bessere Chancen hat man, wenn man die Polygone in eine *funktionale* Darstellung bringt, wie im folgenden demonstriert wird.

## 2.1 Ein Ansatz über innere Abstände

Wie in Kapitel 1 angedeutet, sollten Ähnlichkeitsfunktionen, die in der Roboterlokalisierung angewendet werden, translations- und rotationsunabhängig sein, d.h. Polygone ohne Rücksicht auf deren Lage vergleichen können. Wir definieren daher  $\mathcal{T}$  als die Menge aller Translationen entlang beliebiger Vektoren sowie aller Rotationen um beliebige Punkte und Winkel in der Ebene. Im Sinne von Kapitel 1.3.2 suchen wir nun formal nach einer Ähnlichkeitsfunktion  $s$  der Gestalt

$$s(\mathcal{A}, \mathcal{B}) := \inf_{t_1, t_2 \in \mathcal{T}} d(t_1(\mathcal{A}), t_2(\mathcal{B})).$$

### 2.1.1 Die Polarkoordinatenfunktion

Die Sichtbarkeitspolygone und -skelette, die bei der Roboterlokalisierung auftreten, sind sternförmig und liefern mit dem Anfragepunkt  $p$  auch gleich einen Kernpunkt mit. Der Laser-Radar befand sich bei der Gewinnung dieser Polygone an der Stelle  $p$  und hat in regelmäßigen Winkelabständen von  $0^\circ$  bis  $360^\circ$  die Entfernung zum Auftreffpunkt des Laserstrahles gemessen. Da der Strahl stets auf genau einen Punkt der Szene trifft (abgesehen von dem Fall zu geringer Reichweite, in dem gar keine Auftreffentfernung vorliegt), wird so jedem Winkel im Intervall  $[0, 2\pi]$  eindeutig eine reelle Zahl als Entfernung zugeordnet. Dies legt die Betrachtung als Funktion nahe und ergibt damit eine funktionale Darstellung von sternförmigen Polygonen. Bild 2.1 zeigt ein Beispiel.

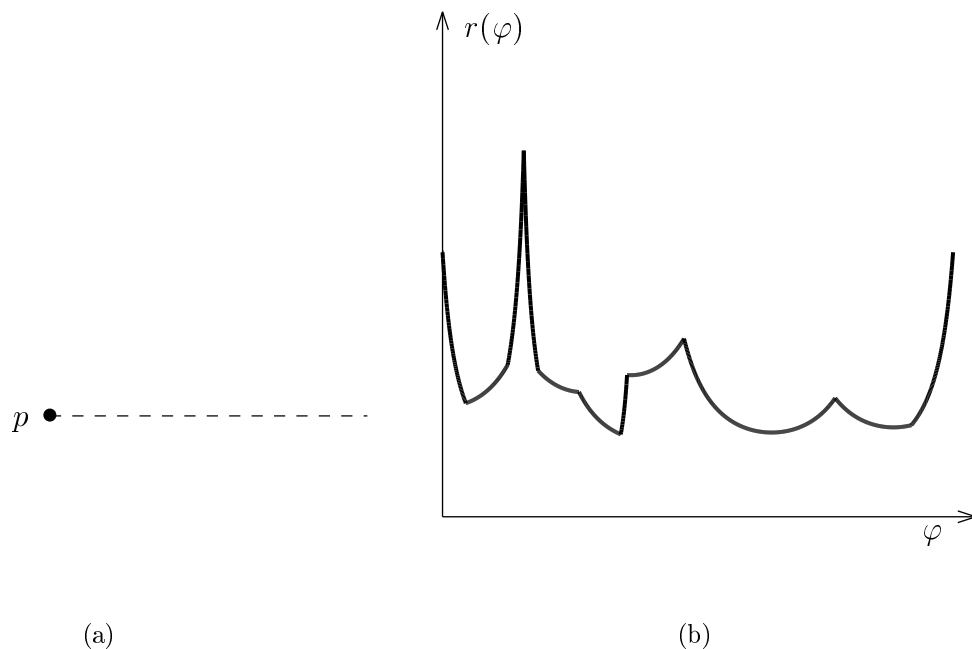


Abbildung 2.1: Ein Polygon mit Kernpunkt (a) und die Entfernung  $r$  der Peripheriepunkte als Funktion vom Winkel  $\varphi$  (b)

Man kann die Funktion auch wie folgt charakterisieren: Stellt man sich den Anfragepunkt  $p$  als Ursprung eines Koordinatensystems vor, so repräsentiert das Paar  $(\varphi, r)$

mit dem Strahlwinkel  $\varphi$  und dem Abstand  $r$  eines Punktes  $q$  von  $p$  auf dem Polygonrand genau die Polarkoordinaten von  $q$ . Daher bezeichnen wir die Funktion wie folgt:

**Definition 2.1 (Polarkoordinatenfunktion)** Seien  $\mathcal{P}$  ein sternförmiges Polygon sowie  $p$  ein Punkt im Kern. Bezeichne weiter für einen Winkel  $\varphi$  der Ausdruck  $q(\varphi)$  den Schnittpunkt desjenigen Strahls mit der Polygonperipherie, der in  $p$  startet und mit der positiven Horizontalrichtung den Winkel  $\varphi$  (gegen den Uhrzeigersinn) einschließt. Dann heißt die Funktion

$$\text{pkf}: \mathbb{R} \longrightarrow \mathbb{R}_{\geq 0}, \quad \text{pkf}(\varphi) = d_2(p, q(\varphi))$$

die **Polarkoordinatenfunktion** (Abk. **PKF**) von  $\mathcal{P}$  (bez.  $p$ ).

### Bemerkungen.

- 1.) In Abschnitt 2.1.3.1 auf Seite 23 wird sich herausstellen, daß es sinnvoll ist, für  $p$  auch Randpunkte des Kerns zuzulassen. In diesem Fall ist  $\text{pkf}$  formal nicht auf ganz  $[0, 2\pi]$  definiert, nämlich an den Stellen nicht, für deren Winkel der Laserstrahl mit einer Kante kollinear verläuft. Um die Notationen nicht zu verkomplizieren, gehen wir – wenn nicht explizit das Thema der Kollinearität behandelt wird – von einem Punkt  $p$  im Kerninnern aus.
- 2.) Für die Bestimmung des Funktionswertes  $\text{pkf}(\varphi)$  wurde der Euklidische Abstand  $d_2$  von Punkten gewählt, da er translations- und rotationsinvariant ist. Diese Eigenschaft gilt zum Beispiel nicht für die ebenfalls gebräuchlichen Abstandsbegriffe  $d_1$  und  $d_\infty$ .
- 3.) Die PKF eines Polygons ist rotationsabhängig, solange man immer die positive  $x$ -Achse als Nullrichtung benutzt. Eine Drehung des Polygons entspricht dann einer Verschiebung des Graphen entlang der  $x$ -Achse (waagrecht).
- 4.) Bis auf Translationen und Rotationen ist die Darstellung eines Polygons durch seine PKF jedoch eindeutig; siehe auch Abschnitt 2.3.1.
- 5.) Die Funktion  $\text{pkf}$  ist natürlich  $2\pi$ -periodisch, weshalb wir uns auf den Ausschnitt  $\text{pkf}|_{[0, 2\pi]}$  beschränken können. Aus technischen Gründen erwies es sich aber als besser,  $\text{pkf}$  auf ganz  $\mathbb{R}$  zu definieren.

### 2.1.2 Analytische Beschreibung

Im folgenden soll die Polarkoordinatenfunktion in Form einer Gleichung angegeben werden. Dazu betrachte man Bild 2.2. Jedes sternförmige Polygon läßt sich anhand eines Kernpunktes in Sektoren zerlegen (triangulieren). Der Funktionswert von  $\text{pkf}$  an den Rändern  $P_i, P_{i+1}$  eines Sektors ergibt sich aus den Abständen  $|pP_i|$  und  $|pP_{i+1}|$ ; im Innern eines Sektors gilt für einen Punkt  $P$  und die im Bild bezeichneten Größen

$$\frac{r(\varphi)}{\sin \beta} = \frac{|pP_i|}{\sin \varepsilon}, \quad \text{d.h. } r(\varphi) = \frac{\sin \beta}{\sin(\pi - (\varphi + \beta))} \cdot |pP_i|.$$

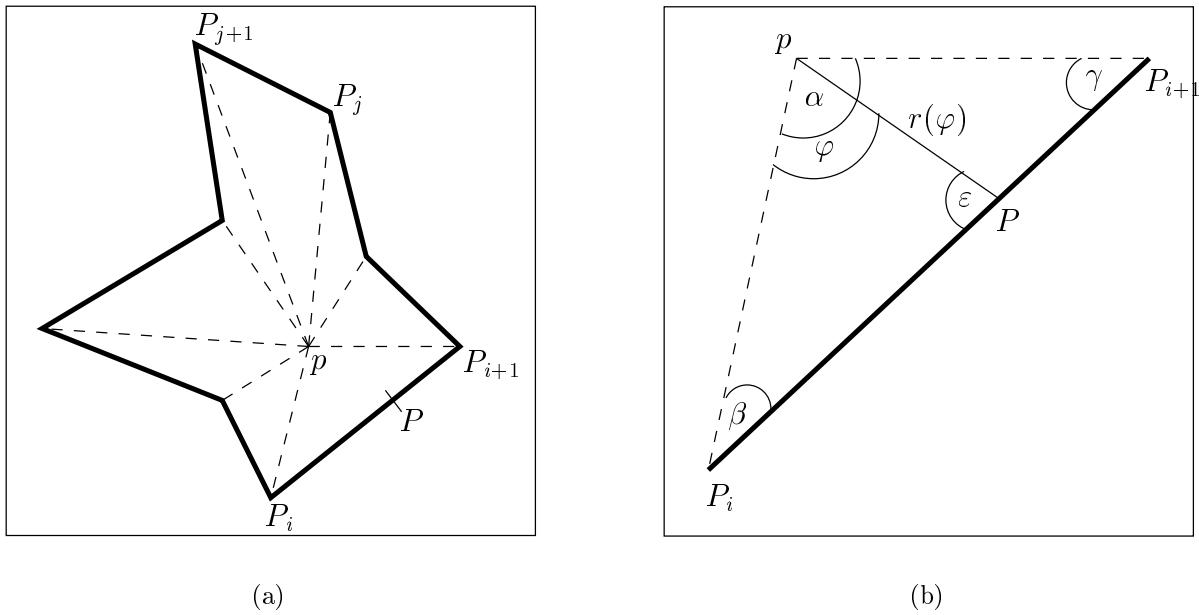


Abbildung 2.2: Sektorenzerlegung eines Polygons (a); ein Ausschnitt (b)

Faßt man alle innerhalb eines Sektors konstanten Größen zu  $c := \sin \beta \cdot |pP_i|$  zusammen, so erhält man

$$r(\varphi) = \frac{c}{\sin(\varphi + \beta)}. \quad (2.1)$$

Man beachte, daß  $\varphi$  hier zunächst ein lokaler Winkel innerhalb des Sektors ist. Es gilt  $r(0) = c / \sin \beta = |pP_i|$  und  $r(\alpha) = \sin \beta / \sin \gamma \cdot |pP_i| = |pP_{i+1}|$ .

Die gesamte Funktion  $\text{pkf}$  ist damit eine *stetige* Funktion, denn sie ist lückenlos aus stetigen Funktionen der Form (2.1) zusammengesetzt; an den Rändern haben angrenzende Sektoren jeweils gleiche Funktionswerte.

Ausnahmen sind diejenigen Stellen  $\varphi$ , für die der Strahlwinkel kollinear mit einer Polygonkante verläuft, wenn also  $p$  auf dem Rand des Kerns liegt. In diesem Fall hat der Graph an jeder solchen Stelle einen Sprung, dessen Höhe genau mit der Länge der kollinearen Kante übereinstimmt.

Zur Analyse des Kurvenverlaufs der Funktion lassen sich die Kanten des Polygons bezüglich  $p$  in zwei Klassen einteilen:

1. Kanten, entlang denen der Abstand zu  $p$  streng monoton wächst oder fällt. Das sind die Kanten, die nicht den Fußpunkt des Lots durch  $p$  auf die unterstützende Gerade enthalten (z.B. Kante  $\overline{P_j P_{j+1}}$  in Abbildung 2.2(a)).
2. Kanten, in deren Innern der Abstand zu  $p$  ein lokales Minimum besitzt. Das sind die Kanten, auf die sich  $p$  senkrecht projizieren läßt (z.B. Kante  $\overline{P_i P_{i+1}}$  in Abbildung 2.2(a)). Die Projektion von  $p$  gibt die Stelle des lokalen Minimums an.

Im Innern von Kanten gibt es keine lokalen Maxima der Abstände.

Betrachtet man die Ecken des Polygons, so läßt sich zeigen, daß lokale Maxima höchstens an Konkavecken auftreten; unter anderem immer dann, wenn der anliegende Innenwinkel kleiner als  $90^\circ$  ist. Lokale Minima können (außer im Innern von Kanten) nur an Konkavecken auftreten; unter anderem immer dann, wenn der Innenwinkel größer als  $270^\circ$  ist. Für alle Winkelgrößen dazwischen kommt es auf die spezielle Lage von Kernpunkt zu Eckpunkt an.

Jede Ecke des Polygons definiert im Graphen eine *Übergangsstelle*, an der sich die Gleichung der Funktion ändert und an der sie (falls der anliegende Innenwinkel von  $180^\circ$  verschieden ist) im allgemeinen nicht differenzierbar ist. An allen anderen Stellen im Intervall  $[0, 2\pi]$  ist die PKF konvex, denn es gilt für die Funktion  $f(x) = 1/\sin x$ :

$$f'(x) = \frac{-\cos x}{\sin^2 x}, \quad f''(x) = \frac{1 + \cos^2 x}{\sin^3 x}.$$

Da  $0 < \varphi + \beta < \pi$  gilt, ist  $f''(\varphi + \beta) > 0$  für alle  $\varphi$ .

### 2.1.3 Ableitung einer Ähnlichkeitsfunktion

Wir wollen nun die Ähnlichkeit sternförmiger Polygone über ihre Polarkoordinatenfunktionen messen. Dazu muß zunächst für beide Polygone der Parameter der Funktionen – der jeweilige Kernpunkt  $p$  – fixiert werden. Ist das geschehen, kann man den Abstand der nun festgelegten Graphen mit einem geeigneten Distanzmaß für Funktionen bestimmen.

Die Auswahl des Funktionsparameters  $p$  ist wesentlich für die Qualität der entstehenden Ähnlichkeitsfunktion, weshalb ihr große Aufmerksamkeit gewidmet werden muß.

#### 2.1.3.1 Die Festlegung des Kernpunktes $p$

Zu Beginn dieses Kapitels wurde daran erinnert, daß durch die speziellen Umstände der Roboterlokalisierung zu einem sternförmigen Anfrageskelett stets auch ein Kernpunkt – der Anfragepunkt – bekannt ist. Dieser ist aber für unsere Zwecke nicht geeignet: Variiert die Anfrageposition innerhalb einer Sichtbarkeitszelle, so ist das zugehörige Skelettpolygon stets dasselbe, während sich die Lage dieses speziellen Kernpunktes relativ zum Polygon verändert. Damit verändert sich auch der Graph der PKF für ein und dasselbe Polygon – ein Umstand, der eine grundlegende Metrikeigenschaft verletzen würde: Identische (d.h. kongruente) Polygone sollten dieselbe Polarkoordinatenfunktion haben, damit später ihr Ähnlichkeitsmaß 0 gesichert ist.

Fassen wir noch einmal die Anforderungen an  $p$  zusammen:

1.  $p$  muß ein Kernpunkt sein.
2. Die Bestimmung von  $p$  muß *deterministisch* sein.
3. Die Lage von  $p$  relativ zum Polygon muß unabhängig sein von Verschiebungen oder Drehungen des Polygons.
4. Die Bestimmung von  $p$  sollte unempfindlich sein gegenüber verrauschten Eingabedaten, z.B. Zickzacklinien anstelle glatter Polygonkanten.

Die ersten drei Punkte legen die Verwendung des *Schwerpunktes des Kerns* nahe. Das hat sich allerdings als unzureichend erwiesen. Man betrachte hierzu Bild 2.3. Der Kern selbst (schattiert) und damit auch sein Schwerpunkt  $K$  sind stark von unglatten Polygonkanten beeinflussbar, auch wenn der Grad des Rauschens gering ist: Für die Kernbestimmung sind die Innenwinkel an den durch das Rauschen entstandenen Ecken entscheidend, weniger die Längen der zusätzlichen Kantenstücke.

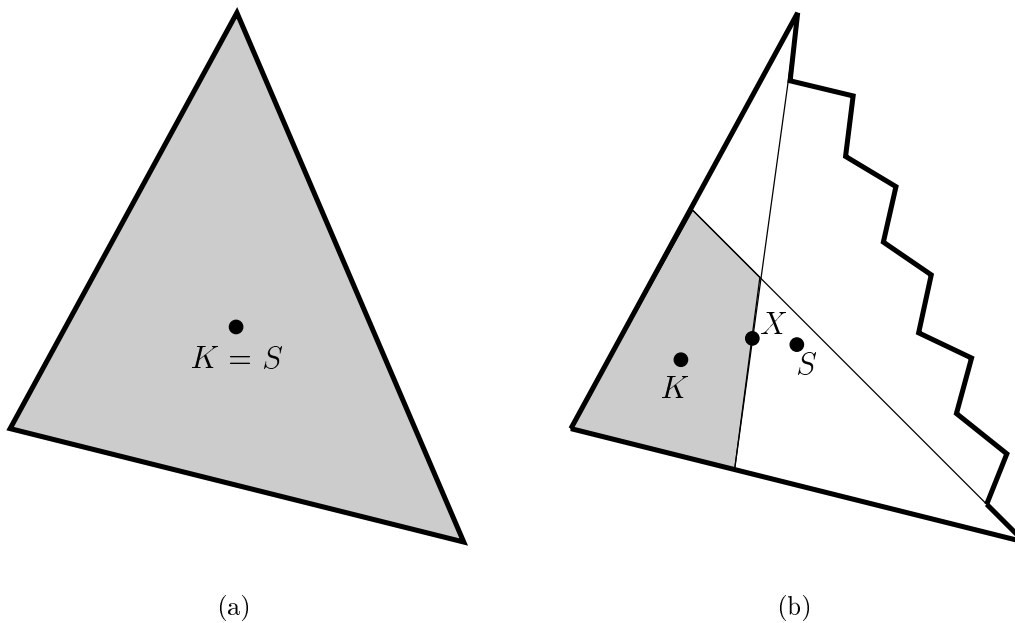


Abbildung 2.3: Veränderung des Kerns ähnlicher Polygone durch verrauschte Kanten

Die Unempfindlichkeit gegenüber ungenauen Eingaben wäre mit dem Schwerpunkt  $S$  (siehe Bild) des gesamten Polygons erreicht, was aber im allgemeinen kein Kernpunkt ist. Es kann folgender Kompromiß gefunden werden:

**Festlegung 2.2 (Kernpunktwahl)** *Liegt der Schwerpunkt  $S$  des Polygons  $\mathcal{P}$  im Kern, so wähle man  $p := S$ ; andernfalls wähle man den Punkt auf dem Rand des Kerns, der zum Schwerpunkt den geringsten Abstand hat.*

Dazu kann man wie folgt vorgehen (Abbildung 2.3(b)):

Man bestimmt parallel in *einem* Durchlauf über die Kanten des Kerns von  $\mathcal{P}$ , ob sich  $S$  im Kern befindet und den Randpunkt  $X$  des Kerns, der  $S$  am nächsten ist. Dazu genügt es, sukzessive den Punkt jeder *Kante* zu bestimmen, der  $S$  am nächsten ist. Stellt sich am Ende heraus, daß  $S$  außerhalb liegt, hat man mit  $X$  bereits den Punkt des Kerns, der zu  $S$  den geringsten Abstand hat (ansonsten ist  $X$  irrelevant).

Ein auf diese Weise bestimmter Kernpunkt erfüllt die Bedingungen 1 bis 3 vollständig und die Bedingung 4 in besserer Qualität, als es der Schwerpunkt des Kerns tut, wie gesehen.

Allerdings können diese Maßnahmen nicht verhindern, daß durch das Rauschen der Kern unter Umständen ganz verschwindet, wodurch die Polarkoordinatenfunktion nicht mehr definiert ist. Wenn man unter *Rauschen* das Vorhandensein zusätzlicher Meßpunkte

entlang einer eigentlich glatten Linie versteht, dann ist die Gefahr des Verschwindens des Kerns um so kleiner, je flacher die neu entstandenen Innenwinkel sind, d.h. je geringer der Wert  $|\pi - \alpha|$  für einen solchen Winkel ist. Sehr kleine ( $< 90^\circ$ ) und sehr große Innenwinkel ( $> 270^\circ$ ) erzeugen sehr kleine Sichtbarkeitskegel, die zusammen mit anderen einen leeren Schnitt ergeben können. Bild 2.4(a) zeigt einen solchen Fall.

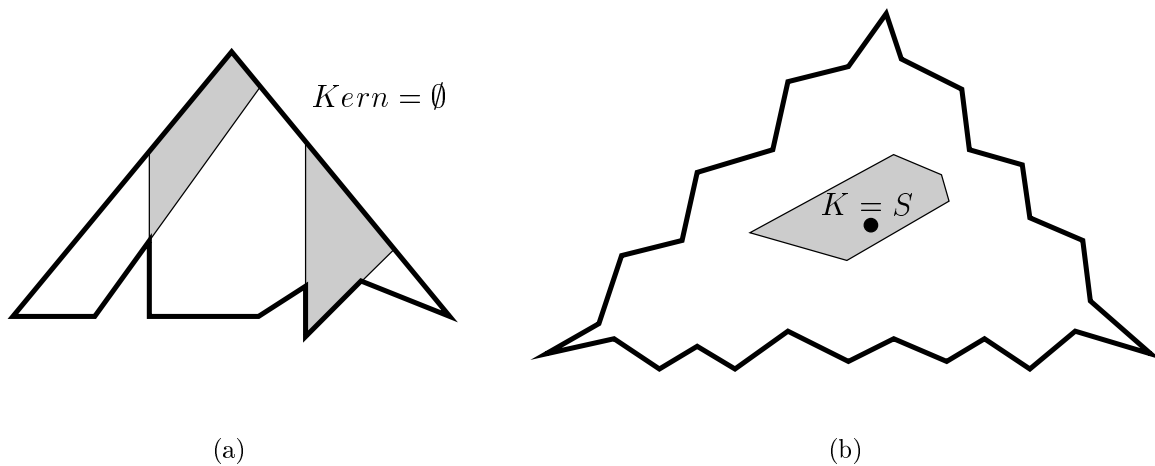


Abbildung 2.4: Extremfälle der Kernveränderung durch verrauschte Kanten

Im Fall der Roboterlokalisierung, wo das Sichtbarkeitspolygon und später das Skelett einem Laserscan entnommen werden, tritt der Extremfall aus Abbildung 2.4(a) allerdings nie auf: Laser-Ungenauigkeiten wirken sich nur auf die gemessenen Entfernungen aus; das Polygon, das durch die Verbindung der Laser-Meßpunkte entsteht, ist natürlicherweise sternförmig.

Die Bestimmung des schwerpunktnächsten Kernpunktes ist besonders bei uneinheitlichem Rauschen sinnvoll, wie in Abbildung 2.3(b). Sind alle Seiten gleichmäßig gestört (Bild 2.4(b)), so ändert sich der ausgewählte Punkt  $p$  zum ursprünglichen glatten Polygon unter Umständen gar nicht (d.h.  $K = S$ ). Auch die Anzahl von fehlerhaften Meßpunkten hat keinen unmittelbaren Einfluß auf die Qualität des Ergebnisses: Zwar können verrauschte Kanten relativ zu glatten Kanten beliebig lang werden, aber der von  $p$  aus abgedeckte Winkelbereich wird durch das Rauschen nicht verändert. Dadurch ist das Teilintervall von  $[0, 2\pi]$ , in dem die Polarkoordinatenfunktion eine verrauschte Kante darstellt, im wesentlichen dasselbe wie im unverrauschten Fall<sup>1</sup>

### 2.1.3.2 Definition der Ähnlichkeitsfunktion

Laut Definition 2.1 beginnt die PKF ihre Abstandsmessung stets mit dem Punkt, der durch einen nach rechts gerichteten Horizontalstrahl (beginnend in  $p$ ) als Schnittpunkt mit der Polygonperipherie gebildet wird. Damit zwei kongruente Polygone dieselbe Funktion erzeugen, muß eines so gedreht werden, daß ein fester Winkel  $\varphi$  als Argument für bei-

<sup>1</sup>Dabei ist es wesentlich, daß das Argument der PKF der *Winkel* ist, den der Laserstrahl mit einer Bezugsrichtung einschließt. In Abschnitt 2.3.3 auf Seite 67 werden wir Alternativen dazu erläutern, die gegenüber uneinheitlichem Rauschen anfälliger sind.

de Funktionen *korrespondierende* Peripheriepunkte induziert. (Danach genügt eine reine Translation, um die Polygone zur Deckung zu bringen.) Natürlich ist dieser Drehwinkel a priori nicht bekannt. Damit die gesuchte Ähnlichkeitsfunktion rotationsunabhängig wird, muß der Abstand beider Polarkoordinatenfunktionen über alle Wahlen solcher Drehwinkel minimiert werden.

Analytisch läßt sich eine Drehung eines Polygons durch eine Argumentverschiebung seiner PKF ausdrücken: Sind  $\mathcal{P}$  und  $\mathcal{P}'$  kongruent und ist  $\mathcal{P}'$  gegenüber  $\mathcal{P}$  um den Winkel  $t$  (gegen den Uhrzeigersinn) gedreht, so gilt

$$\text{pkf}_{\mathcal{P}'}(\varphi) = \text{pkf}_{\mathcal{P}}(\varphi - t)$$

für alle  $\varphi$ . Vereinbaren wir nun noch, daß wir die Integralmetrik für stetige Funktionen ( $L_2$ -Norm) als Abstandsmaß für die Polarkoordinatenfunktionen verwenden, dann läßt sich eine metrische Ähnlichkeitsfunktion definieren:

**Definition 2.3 (PKF-Distanz)** Seien  $\mathcal{A}$  und  $\mathcal{B}$  zwei sternförmige Polygone, für die nach dem Verfahren aus Abschnitt 2.1.3.1 jeweils ein spezieller Kernpunkt festgelegt wurde. Seien  $\text{pkf}_{\mathcal{A}}$  und  $\text{pkf}_{\mathcal{B}}$  die diesbezüglichen Polarkoordinatenfunktionen.

Die Abbildung

$$s(\mathcal{A}, \mathcal{B}) := \min_{t \in [0, 2\pi]} \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t) - \text{pkf}_{\mathcal{B}}(\varphi) \right)^2 d\varphi}$$

heißt **PKF-Distanz** für sternförmige Polygone.

**Satz 2.4 (PKF-Metrik)** Die PKF-Distanz ist eine Ähnlichkeitsfunktion mit Metrikeigenschaften und wird daher im folgenden auch **PKF-Metrik** genannt.

Für den Beweis dieser Aussagen benötigen wir noch einen Hilfssatz:

**Lemma 2.5** Für zwei Polarkoordinatenfunktionen  $\text{pkf}_{\mathcal{A}}$  und  $\text{pkf}_{\mathcal{B}}$  sowie beliebige Parameter  $t_{\mathcal{A}}$ ,  $t_{\mathcal{B}}$  und  $t \in \mathbb{R}$  gilt:

$$\int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}}) - \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}}) \right)^2 d\varphi = \int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}} - t) - \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}} - t) \right)^2 d\varphi.$$

**Beweis:** Da die Funktionen  $\text{pkf}$   $2\pi$ -periodisch sind, ist auch die Funktion  $g(\varphi) := (\text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}}) - \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}}))^2$   $2\pi$ -periodisch. Daher gilt  $\int_0^{2\pi} g(\varphi) d\varphi = \int_0^{2\pi} g(\varphi - t) d\varphi$  für alle  $t$ .  $\square$

### **Beweis zu Satz 2.4:**

0. Da die Funktion

$$f: [0, 2\pi] \longrightarrow \mathbb{R}_{\geq 0}, \quad f(t) = \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t) - \text{pkf}_{\mathcal{B}}(\varphi) \right)^2 d\varphi}$$

stetig und  $[0, 2\pi]$  kompakt ist, existiert das Minimum in der Definition von  $s(\mathcal{A}, \mathcal{B})$ .

- 1.a Sind  $\mathcal{A}$  und  $\mathcal{B}$  kongruent, so existiert ein  $t_0 \in [0, 2\pi]$ , um das  $\mathcal{A}$  gedreht werden kann, so daß die Polygone dann nur durch eine Translation auseinander hervorgehen. Ist  $\mathcal{A}'$  das Bild von  $\mathcal{A}$  unter dieser Drehung, so gilt  $\text{pkf}_{\mathcal{A}}(\varphi - t_0) = \text{pkf}_{\mathcal{A}'}(\varphi) = \text{pkf}_{\mathcal{B}}(\varphi)$  für alle  $\varphi$ , also  $s(\mathcal{A}, \mathcal{B}) = 0$ .
- 1.b Ist umgekehrt  $s(\mathcal{A}, \mathcal{B}) = 0$ , so existiert ein  $t_0 \in [0, 2\pi]$  mit  $\text{pkf}_{\mathcal{A}}(\varphi - t_0) = \text{pkf}_{\mathcal{B}}(\varphi)$  für alle  $\varphi$ . Ist  $\mathcal{A}'$  das Polygon, das aus  $\mathcal{A}$  durch Drehung um  $t_0$  hervorgeht, so gilt  $\text{pkf}_{\mathcal{A}'} \equiv \text{pkf}_{\mathcal{B}}$ . Überlagert man die Kernpunkte von  $\mathcal{A}'$  und  $\mathcal{B}$ , so kommen nach der Definition der PKF sämtliche Peripheriepunkte von  $\mathcal{A}'$  und  $\mathcal{B}$  zur Deckung (sie haben gleiche Polarkoordinaten). Also sind  $\mathcal{A}'$  und  $\mathcal{B}$  und somit auch  $\mathcal{A}$  und  $\mathcal{B}$  kongruent; im Sinne dieses Kontextes gilt „ $\mathcal{A} = \mathcal{B}$ “.
2. Symmetrie: Sei  $t_{\mathcal{A}\mathcal{B}}$  der Winkel, für den das Minimum in  $s(\mathcal{A}, \mathcal{B})$  angenommen wird, und sei  $t_{\mathcal{B}\mathcal{A}} := 2\pi - t_{\mathcal{A}\mathcal{B}}$ . Dann gilt nach Lemma (2.5)

$$\begin{aligned}
s^2(\mathcal{A}, \mathcal{B}) &= \int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}\mathcal{B}}) - \text{pkf}_{\mathcal{B}}(\varphi) \right)^2 d\varphi \\
&= \int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}\mathcal{B}} - t_{\mathcal{B}\mathcal{A}}) - \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}\mathcal{A}}) \right)^2 d\varphi \\
&\stackrel{(*)}{=} \int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi) - \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}\mathcal{A}}) \right)^2 d\varphi \\
&\geq \min_{t \in [0, 2\pi]} \int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi) - \text{pkf}_{\mathcal{B}}(\varphi - t) \right)^2 d\varphi \\
&= s^2(\mathcal{B}, \mathcal{A}),
\end{aligned}$$

wobei in (\*) die Tatsache  $t_{\mathcal{A}\mathcal{B}} + t_{\mathcal{B}\mathcal{A}} = 2\pi$  sowie die  $2\pi$ -Periodizität der Funktion  $\text{pkf}$  benutzt wurden.

Analog zeigt man  $s(\mathcal{B}, \mathcal{A}) \geq s(\mathcal{A}, \mathcal{B})$  und erhält Gleichheit.

3. Dreiecksungleichung: Der Beweis folgt einer Idee von Arkin et. al in [ACH<sup>+</sup>91]:

Seien  $\mathcal{A}$ ,  $\mathcal{B}$  und  $\mathcal{C}$  entsprechende Polygone, und seien  $t_{\mathcal{A}\mathcal{B}}$  und  $t_{\mathcal{B}\mathcal{C}}$  die Werte für  $t$ , bei denen die Minima in  $s(\mathcal{A}, \mathcal{B})$  bzw.  $s(\mathcal{B}, \mathcal{C})$  angenommen werden. Wiederum nach Lemma (2.5) gilt

$$\begin{aligned}
s(\mathcal{A}, \mathcal{B}) + s(\mathcal{B}, \mathcal{C}) &= \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}\mathcal{B}}) - \text{pkf}_{\mathcal{B}}(\varphi) \right)^2 d\varphi} \\
&\quad + \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}\mathcal{C}}) - \text{pkf}_{\mathcal{C}}(\varphi) \right)^2 d\varphi} \\
&= \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}\mathcal{B}} - t_{\mathcal{B}\mathcal{C}}) - \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}\mathcal{C}}) \right)^2 d\varphi} \\
&\quad + \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{B}}(\varphi - t_{\mathcal{B}\mathcal{C}}) - \text{pkf}_{\mathcal{C}}(\varphi) \right)^2 d\varphi}
\end{aligned}$$

$$\begin{aligned}
&\stackrel{(**)}{\geq} \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t_{\mathcal{A}B} - t_{BC}) - \text{pkf}_{\mathcal{C}}(\varphi) \right)^2 d\varphi} \\
&\geq \min_{t \in [0, 2\pi]} \sqrt{\int_0^{2\pi} \left( \text{pkf}_{\mathcal{A}}(\varphi - t) - \text{pkf}_{\mathcal{C}}(\varphi) \right)^2 d\varphi} \\
&= s(\mathcal{A}, \mathcal{C}),
\end{aligned}$$

wobei in (\*\*) die Minkowski-Ungleichung

$$\sqrt{\int_a^b f^2(x) dx} + \sqrt{\int_a^b g^2(x) dx} \geq \sqrt{\int_a^b (f(x) + g(x))^2 dx}$$

benutzt wurde.

□

### 2.1.4 Berechnung der PKF-Metrik

In diesem Abschnitt soll die Implementation der vorgestellten Metrik einschließlich einer Laufzeitanalyse genauer beschrieben werden.

Zur Vorbereitung der eigentlichen Differenzberechnung sind drei Schritte nötig, die für jedes der beiden Eingabepolygone  $\mathcal{P}$  getrennt durchzuführen sind. Sei dabei  $k := \max(m, n)$  die größere der beiden Eckenzahlen.

1. Berechnung des Kerns von  $\mathcal{P}$ . Dies kann mit einem Algorithmus zum Schnitt von Halbebenen in  $\mathcal{O}(k \log k)$  Schritten geschehen. Ist der Kern leer, so ist die Eingabe abzulehnen.
2. Berechnung des Schwerpunkts  $S$  von  $\mathcal{P}$ . Dies kann in Zeit  $\mathcal{O}(k)$  wie folgt geschehen: Mittels eines beliebigen Kernpunktes  $K$  berechnet man eine Triangulation von  $\mathcal{P}$ , indem  $K$  mit allen Ecken verbunden wird. Von den entstandenen Dreiecken kann der Schwerpunkt  $S_i$  über die arithmetischen Mittel der drei Eckpunkte gebildet werden.  $S$  ergibt sich dann als über die Flächeninhalte  $A_i$  gewichtetes Mittel der  $S_i$  zu

$$x_S = \frac{\sum_i x_{S_i} \cdot A_i}{\sum_i A_i}, \quad y_S = \frac{\sum_i y_{S_i} \cdot A_i}{\sum_i A_i}.$$

3. Bestimmung eines speziellen Kernpunktes  $P$  nach dem Verfahren aus Abschnitt 2.1.3.1. Da dieses den Test „Liegt  $S$  im Kern?“ enthält, sind nochmals  $\mathcal{O}(k)$  elementare Schritte nötig.

Dies ergibt zusammen einen Aufwand von  $\mathcal{O}(k \log k)$  Schritten. Falls in einer Szene mit mehreren Objekten gehäufte Ähnlichkeitsanfragen zu erwarten sind, so kann dieser Aufwand als Teil eines Preprocessing-Schritts bestritten werden.

Mit der Festlegung von  $p$  ist die Polarkoordinatenfunktion eines Polygons definiert. Bei der Berechnung der Funktionswerte beachte man, daß Formel (2.1) auf Winkeln operiert, die lokal für einen Sektor gelten (siehe Abschnitt 2.1.2). Bei der Übergabe eines globalen Wertes  $\varphi \in [0, 2\pi]$  als Argument von  $\text{pkf}$  ist vor der Anwendung der Formel der Startwinkel des ersten Peripheriewinkels des Sektors (gegen den Uhrzeigersinn) abzuziehen.

Nun ist gemäß Definition 2.3 die  $L_2$ -Norm der beiden PKF-Kurven über den Verschiebeparameter  $t$  zu minimieren. Aufgrund der relativ komplizierten Struktur der Polarkoordinatenfunktion kann die Auswahl aus *allen* reellen Werten von  $t$  leider nicht so eingeschränkt werden, daß sie effizient implementierbar wird. Daher müssen zugunsten der Praktikabilität Abstriche an den soeben bewiesenen Eigenschaften der Ähnlichkeitsfunktion gemacht werden, indem man sich auf die näherungsweise Berechnung der Metrik beschränkt. Dazu gibt es zwei sehr verschiedene Ansätze.

### 2.1.4.1 Lineare Approximation

Die PKF ist allein durch die Angabe derjenigen Funktionswerte  $\text{pkf}(\varphi)$  eindeutig festgelegt, für die  $\varphi$  Abszisse einer Übergangsstelle ist, also einem Eckpunkt im ursprünglichen Polygon entspricht. Wir wollen diese Werte *Stützstellenwerte* nennen. Der Grund ist, daß zwei benachbarte Stützstellenwerte  $\text{pkf}(\varphi_i)$ ,  $\text{pkf}(\varphi_i + \alpha)$  die Lage zweier benachbarter Polygonecken bestimmen. Die dazwischenliegende Polygonkante ist damit festgelegt:

$$\text{pkf}_{\mathcal{A}}(\varphi_i) = \text{pkf}_{\mathcal{B}}(\varphi_i) \wedge \text{pkf}_{\mathcal{A}}(\varphi_i + \alpha) = \text{pkf}_{\mathcal{B}}(\varphi_i + \alpha) \implies \forall_{\varphi \in [\varphi_i, \varphi_i + \alpha]} \text{pkf}_{\mathcal{A}}(\varphi) = \text{pkf}_{\mathcal{B}}(\varphi)$$

Die Polarkoordinatenfunktionen zweier Polygone sind also genau dann identisch, wenn sie in allen Stützstellenwerten übereinstimmen. Das hat die Konsequenz, daß für *jede beliebige* (stetige) deterministische Approximation der PKF zwischen den Stützstellen die Ähnlichkeitsfunktion aus Definition 2.3 wieder eine Metrik ergibt: Die Identitätseigenschaft gilt aufgrund des eben Gesagten und weil die ursprüngliche PKF-Metrik sie erfüllt. Die Beweise der Symmetrie und der Dreiecksungleichung aus Satz 2.4 haben in keiner Weise den Kurvenverlauf der PKF ausgenutzt, sondern im wesentlichen die Periodizität.

Wir könnten also einfach die Stützstellen durch Strecken verbinden und erhielten eine neue, leichter berechenbare Metrik. In Abschnitt 2.1.2 wurde festgestellt, daß es Kanten gibt, in deren Innern der Abstand zum Kernpunkt  $p$  ein lokales Minimum besitzt. Dies gilt dann auch für den Graphen im zugehörigen Winkelabschnitt. Um die Abweichungen der approximierten zur exakten PKF gering zu halten, fügen wir an diesen Minima zusätzliche Stützstellen ein und verbinden dann jeweils benachbarte Stützstellen durch ein Geradenstück. Damit haben wir erreicht, daß nur monotone Kurvenstücke durch Geraden approximiert werden. Die entstehende stückweise lineare Funktion werde mit  $\mathbf{PKF}_{\text{lin}}$  bezeichnet.

Es ist zu beachten, daß es zwischen benachbarten Stützstellen der ursprünglichen PKF höchstens eine Minimumstelle gibt, da der Abstand der Peripheriepunkte eines Polygons entlang einer Kante zum Kernpunkt höchstens an einer Stelle minimal ist. Sie lassen sich leicht identifizieren:

$$\text{pkf}'(\varphi) = \frac{-c \cdot \cos(\varphi + \beta)}{\sin^2(\varphi + \beta)} = 0 \iff \cos(\varphi + \beta) = 0 \iff \varphi = \frac{\pi}{2} - \beta$$

Da  $\varphi$  im Intervall  $[0, \alpha]$  variiert (vergleiche Abbildung 2.2(b)), liegt in einem Sektor genau dann ein lokales Minimum vor, wenn  $0 < \pi/2 - \beta < \alpha$ . Dazu äquivalent ist die anschaulich sofort einsichtige Bedingung

$$\beta < \frac{\pi}{2} \quad \wedge \quad \gamma < \frac{\pi}{2} .$$

Durch das Approximationsverfahren werden also maximal  $n$  neue Stützstellen in den Graphen eingefügt. Die Entsprechungen dieser Stellen im Polygon sind neue „Eck“punkte, die kollinear zu ihren beiden Nachbarn liegen. Sie beeinflussen weder den Kern noch den Schwerpunkt des Polygons, also auch nicht die Berechnung des speziellen Kernpunktes zu Beginn der Ähnlichkeitsbestimmung.

**Geometrische Interpretation.** Wenn zwei benachbarte Eckpunkte  $P_i$  und  $P_{i+1}$  des Polygons gleichweit vom Kernpunkt  $p$  entfernt sind, so gilt  $\text{pkf}(\varphi_i) = \text{pkf}(\varphi_{i+1}) =: y$  für die zugehörigen Eckpunktwinkel  $\varphi_i$  und  $\varphi_{i+1} = \varphi_i + \alpha$ . Verbindet man die Punkte  $(\varphi_i, y)$  und  $(\varphi_{i+1}, y)$  des Funktionsgraphen durch eine Strecke, so erhält man also eine Waagerechte, d.h. die Abstände zum Kernpunkt ändern sich entlang der Approximationsgeraden nicht. Diese Bedingung ist genau durch Kreisbögen mit dem Zentrum  $p$  erfüllt (siehe Abbildung 2.5 rechts, gepunktete Linien).

Verschiebt man den Punkt  $p$  zu einem Punkt  $p'$ , so daß die Mittelsenkrechte von  $\overline{P_i P_{i+1}}$  nicht mehr durch  $p'$  geht, so ist die Kurve durch  $P_i$  und  $P_{i+1}$ , deren Punktabstand zu  $p'$  sich linear mit dem Winkel  $\varphi$  ändert, kein Kreis mehr, auch nicht mit einem von  $p'$  verschiedenen Mittelpunkt (siehe Abbildung 2.5 links und rechts).

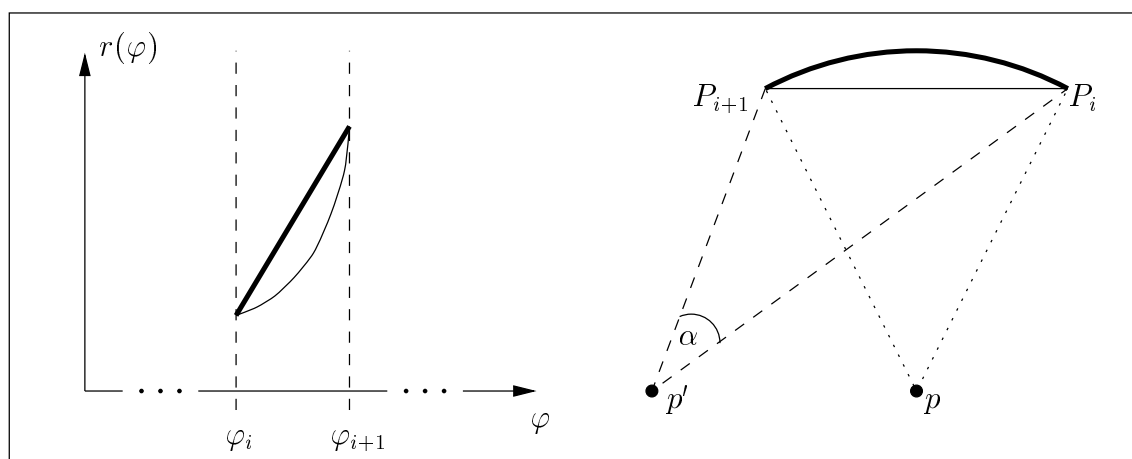


Abbildung 2.5: Approximation der PKF durch Geradenstücke bedeutet Approximation der Polygone durch Bogensegmente

Da die ursprüngliche PKF nach den Überlegungen aus Abschnitt 2.1.2 überall konvex ist, liegen die Approximationsgeraden sämtlich oberhalb der exakten Kurve, geben also zu große Abstände an. Das entspricht genau den nach außen gewölbten Bögen auf der Polygonebene, deren Punkte von  $p$  weiter entfernt sind als die Punkte der Polygonperipherie.

Durch die Geradenapproximation wird also die PKF einer Figur erzeugt, die aus dem Polygon durch das Auftragen von nach außen gerichteten Bögen hervorgeht. Dieser Vorgang ist in Bild 2.6 veranschaulicht. Dort sind die senkrechten Projektionen von  $p$  auf die Kanten gekennzeichnet, falls sie auf den Kanten liegen. Diese Minimalstellen des Abstands der Peripherie zu  $p$  erzeugen bei der Approximation zusätzliche Stützstellen. Sie sind in der rechten Figur durch kleine Kreise markiert.

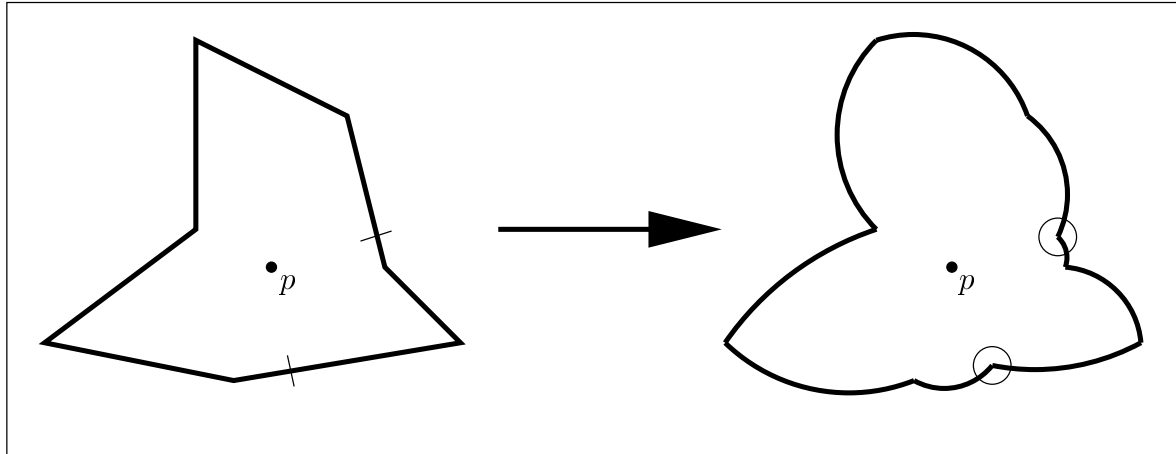


Abbildung 2.6: Umwandlung eines Polygons in eine Figur mit stückweise linearer Polarkoordinatenfunktion (Skizze)

**Fehlerbetrachtung für die Approximation.** Die Abstandswerte  $\text{PKF}_{\text{lin}}$  sind sämtlich größer als die exakten Funktionswerte. Wir haben die Approximation so gewählt, daß nur streng monotone Abschnitte der PKF durch Strecken überbrückt werden.

Die Funktion  $f(x) = 1/\sin x$  hat im Intervall  $[0, \pi]$ , für das wir uns interessieren, etwa den in Abbildung 2.7(a) dargestellten Verlauf. Das Teilintervall  $[\pi/2, \pi]$  ist ein Intervall maximaler Größe, in dem die Funktion monoton ist, in dem sie also komplett durch eine Gerade approximiert wird. Der maximale Fehler tritt auf, wenn die Approximationsgerade  $g_r$  am Punkt  $(\pi/2, 1)$  beginnt und an einem Punkt  $(r, 1/\sin r)$  endet und  $r$  sehr nahe bei  $\pi$  liegt ( $r < \pi$ ). Die Gleichung der Geraden ist parametrisiert über  $r$  gegeben durch

$$g_r(x) = \frac{(2x - \pi)(1 - \sin r)}{(2r - \pi) \sin r} + 1.$$

Den Approximationsfehler  $e$  beider Kurven bestimmen wir mit der  $L_2$ -Norm, da sie später gemäß Definition 2.3 zur Distanzbestimmung zwischen verschiedenen Polarkoordinatenfunktionen genutzt werden soll:

$$e = \lim_{r \rightarrow \pi} e(r) = \lim_{r \rightarrow \pi} \sqrt{\int_{\pi/2}^r \left( g_r(x) - \frac{1}{\sin x} \right)^2 dx} = +\infty.$$

Der Fehler kann also beliebig groß werden, obwohl durch das Einfügen von Zwischenwerten nicht über Extremstellen der Funktion hinweg approximiert wurde. Man beachte allerdings, daß dieser Fehler nur die „Abweichung von der Intuition“ beschreibt, denn die Ähnlichkeitsfunktion von Definition 2.3 mit dieser Näherung der PKF ist ebenfalls eine Metrik. Wie wir gleich sehen werden, kann die  $\text{PKF}_{\text{lin}}$ -Distanz – im Gegensatz zur ursprünglichen PKF – sogar exakt berechnet werden.

Bild 2.7(b) zeigt, daß diese beliebig große Abweichung der  $\text{PKF}_{\text{lin}}$  von der PKF auch in der Praxis auftreten kann. Mit den Bezeichnungen von Abbildung 2.2(b) gilt hier  $\beta = \pi/2$ ,  $\alpha \approx \pi/2$ , d.h. in der Formel  $r(\varphi) = c/\sin(\varphi + \beta)$  bewegt sich das Argument der Sinusfunktion von  $\pi/2$  ( $\varphi = 0$ ) bis etwa  $\pi$  ( $\varphi = \alpha$ ).

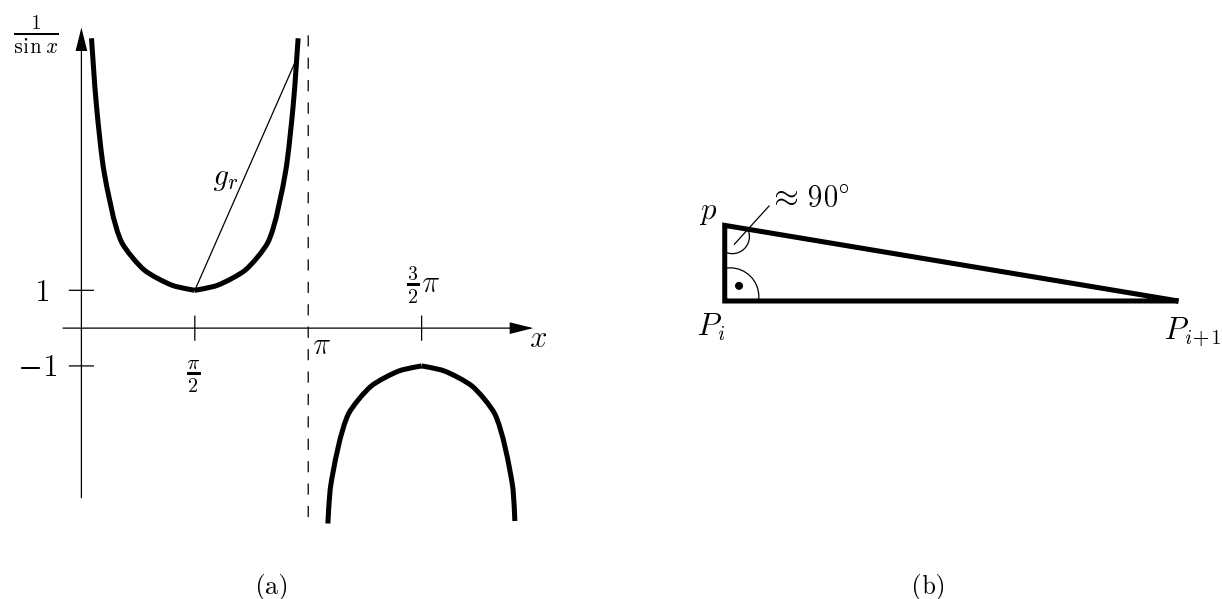


Abbildung 2.7: Graph der Funktion  $y = 1/\sin x$  (a); eine Konstellation benachbarter Polygonecken mit schlechter Näherung der PKF durch eine Gerade (b)

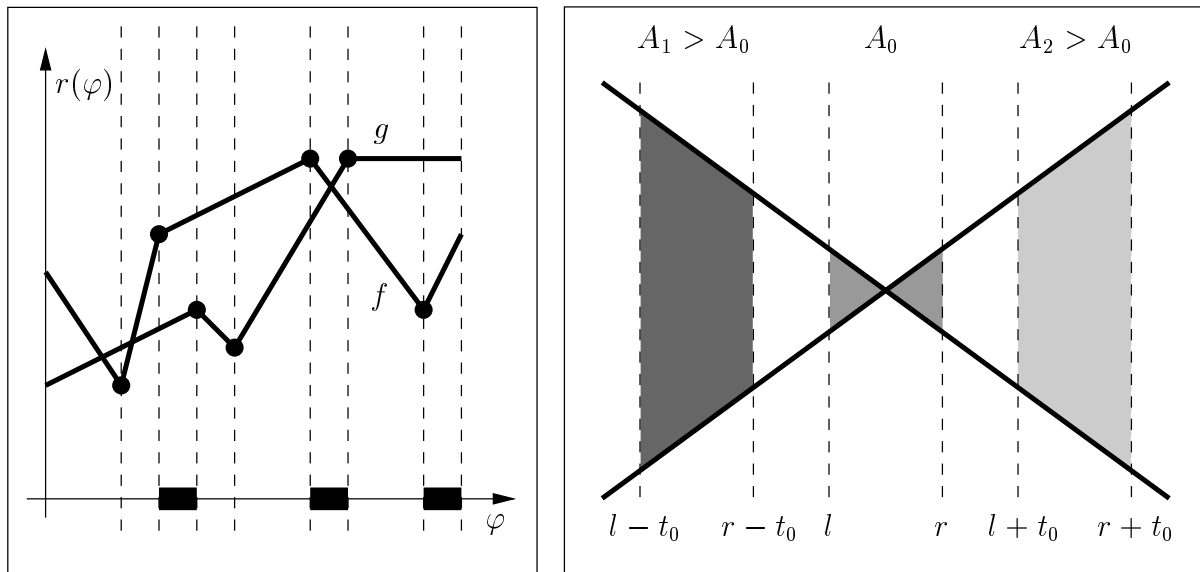
**Eigentliche Differenzbestimmung.** Durch die Geradenapproximation der PKF ist die Minimierung über  $t$  in Definition 2.3 darauf reduziert, eine stückweise lineare Funktion so waagrecht zu verschieben, daß die Flächendifferenz zu einer anderen stückweise linearen Funktion möglichst klein wird.

Seien nun  $f$  und  $g$  die beiden approximierten Polarkoordinatenfunktionen. Die Minimierung über  $t$  und das Wurzelziehen aus Definition 2.3 sind vertauschbar, da die Wurzelfunktion streng monoton ist. Es genügt also, das Integral  $\int_0^{2\pi} (f(x-t) - g(x))^2 dx$  zu minimieren.

Die Idee hierzu ist folgende: Der Graph von  $g$  ist fixiert. Es gibt  $m \cdot n$  *Eventpunkte* beim Verschieben von  $f$ , wenn nämlich zwei Übergangsstellen der Graphen zusammenfallen. Diese müssen explizit vom Algorithmus betrachtet werden, da sich dort die Einteilung des Intervalls  $[0, 2\pi]$  in Streifen mit jeweils geschlossener Funktionsgleichung für  $f$  und  $g$  ändert. Verschiebt man jedoch  $f$  nur *zwischen* zwei Eventpunkten, so läßt sich die Veränderung der Flächendifferenz analytisch sehr schön beschreiben (wenn auch für jeden Streifen separat):

Die Streifeneinteilung ergibt sich durch  $m + n$  senkrechte Linien durch die Übergangsstellen von  $f$  und  $g$  (Abb. 2.8(a)). Wir können ohne Beschränkung der Allgemeinheit festlegen, daß  $f$  zur Minimierung der Flächendifferenz nur nach rechts verschoben wird (wie es die Formel in Definition 2.3 durch  $t \geq 0$  auch andeutet). Denn die Lagen von  $f$  links von der aktuellen Eventpunktlage können durch Verschieben nach rechts aus der vorhergehenden (links befindlichen) Eventpunktlage heraus erreicht werden.

Sind  $l$  und  $r$  die  $x$ -Koordinaten der aktuellen Streifengrenzen, so werden sie zusammen mit  $f$  verschoben, d.h. vermöge des Verschiebeparameters lauten sie parametrisiert  $l + t$  und  $r + t$ .



(a)

(b)

Abbildung 2.8: Einteilung der Zeichenebene in Streifen (a); Fläche zwischen zwei Geraden in Abhängigkeit von der Lage des begrenzenden Streifens (b)

Die Funktionsgleichungen von  $f$  und  $g$  in der Eventpunktlage vor dem Verschieben seien für den aktuellen Streifen durch  $f(x) = ax + b$ ,  $g(x) = cx + d$  gegeben. Die  $L_2$ -Norm zwischen den Kurven im Streifen beträgt dann

$$F(t) := \int_{l+t}^{r+t} \left( a(x-t) + b - (cx+d) \right)^2 dx. \quad (2.2)$$

Dies ist eine quadratische Funktion von  $t$ . In expliziter Darstellung erhält man  $F(t) = x \cdot t^2 + y \cdot t + z$  mit

$$\begin{aligned} x &= c^2 \cdot (r-l) \\ y &= c \cdot (c-a)(r^2-l^2) + 2 \cdot c \cdot (d-b)(r-l) \\ z &= 1/3 \cdot (a-c)^2 \cdot (r^3-l^3) + (a-c)(b-d)(r^2-l^2) + (b-d)^2 \cdot (r-l). \end{aligned} \quad (2.3)$$

An dieser Formel läßt sich unter anderem folgendes ablesen:

1. Ist  $c = 0$ , so ist  $F(t) = z = \text{const}$ . Hat nämlich  $g$  im Streifen den Anstieg  $c = 0$ , so wird  $f$  parallel zu  $g$  verschoben. Also ändert sich der Flächeninhalt gegenüber der Verschiebung nicht.
2. Ist  $c \neq 0$ , so ist  $x > 0$  ( $r > l$  vorausgesetzt). Die Parabel  $F(t)$  ist nach oben geöffnet; für beide Verschieberichtungen  $t \rightarrow \pm\infty$  wächst sie streng monoton. Dies ist in Bild 2.8(b) demonstriert: Die Fläche zwischen zwei Geraden in einem Streifen konstanter Breite  $r-l$  ist umso größer, je weiter der Streifen vom Schnittpunkt der Geraden entfernt ist.

Die quadratische Form (2.2) läßt sich für jeden Streifen aufstellen. Da die Gesamtfläche im Intervall  $[0, 2\pi]$  die Summe der Flächen in jedem Streifen ist, kann auch die Gesamtänderung des Flächeninhalts (bzw. der  $L_2$ -Norm) in Abhängigkeit von  $t$  als Summe von quadratischen Ausdrücken geschrieben werden und ergibt ebenfalls einen quadratischen Term  $F_{\text{ges}}(t) = X \cdot t^2 + Y \cdot t + Z$ . Da sich  $X$  als Summe aus lauter positiven Werten  $x$  darstellt, ist auch  $X$  größer als Null.

Es ist nun abschließend zu klären, welche Werte für  $t$  in der Gesamtformel  $F_{\text{ges}}(t)$  zugelassen sind. Dann kann das globale Minimum des Terms und damit die minimierende Verschiebung von  $f$  bestimmt werden.

Die zulässige Wertemenge ist ein Intervall der Form  $[0, t^+]$ , denn  $t$  beschreibt die Länge einer Verschiebung aus einer initialen Eventpunktlage heraus. Der Wert  $t^+$  entspricht genau dem Betrag, um den  $f$  aus der Initiaallage nach rechts verschoben werden kann, bis erneut ein Eventpunkt erreicht ist. Dann ändert sich die Intervalleinteilung, und die Berechnungen müssen von vorne beginnen.

Formal läßt sich  $t^+$  so beschreiben: Sei  $R_{fg}$  die Menge aller Paare von benachbarten Übergangsstellen von  $f$  und  $g$  in dieser Reihenfolge. (Es interessieren nur die Streifen, die links von einer Übergangsstelle von  $f$  und rechts von einer von  $g$  begrenzt sind, denn  $f$  wird nach rechts verschoben. In Abbildung 2.8(a) gibt es genau 3 Streifen dieser Art; sie sind am unteren Ende schwarz markiert.) Dann ist

$$t^+ = \min\{r - l \in \mathbb{R} : r > l \wedge (l, r) \in R_{fg}\}.$$

Gesucht ist also das Minimum von  $F_{\text{ges}}(t)$  in  $[0, t^+]$ . Dazu ist die lokale Minimumstelle  $t_0$  zu bestimmen. Liegt sie außerhalb von  $[0, t^+]$ , so ist das Ergebnis der kleinere der beiden Funktionswerte an den Intervallrändern.

Die Gesamtprozedur muß für jede Eventpunktlage von  $f$  (bei vorher fest gewählter Lage von  $g$ ) durchgeführt werden. Sie ist im Algorithmus 1 für beliebige stückweise lineare,  $p$ -periodische Funktionen zusammengefaßt.

Für jede Ausgangslage von  $f$  (Zeile 3) ist also ein *Plane Sweep* durchzuführen. Der Wert  $t^+$ , der ja erst am Ende des Sweeps (Zeile 9) benötigt wird, kann parallel im gleichen Durchlauf bestimmt werden, indem für jedes Streifenintervall (Zeile 5) geprüft wird, ob seine Ränder zu  $R_{fg}$  gehören, und der aktuelle Minimalwert  $t^+$  gegebenenfalls aktualisiert wird.

Der Plane Sweep ist in linearer Zeit  $\mathcal{O}(m+n)$  durchführbar. Da es für fest vorgegebenes  $g$  genau  $m \cdot n$  Eventpunktlagen für  $f$  gibt (Schritt 3), erhält man einen Gesamtaufwand von  $\mathcal{O}(mn \cdot (m+n))$ .

Der Algorithmus kann als starke Verallgemeinerung der von Arkin et. al in [ACH<sup>+</sup>91] vorgestellten Minimierung des  $L_2$ -Abstandes stückweise konstanter Funktionen aufgefaßt werden. Während dort gezeigt wurde, daß es für den Fall von Treppenfunktionen genügt, die  $m \cdot n$  Positionen gemeinsamer Übergangsstellen zu betrachten, kann bei allgemeineren stückweise *linearen* Funktionen (die auch Sprünge haben dürfen) das  $L_2$ -Minimum auch zwischen solchen kritischen Stellen liegen. Man erhält den stückweise linearen Fall in den Gleichungen (2.3) mit  $a = c = 0$  zu  $F(t) = (b - d)^2 \cdot (r - l) = \text{const}$ , wobei  $b$  und  $d$  die (lokalen) konstanten Funktionswerte im Streifen von  $l$  bis  $r$  angeben.

---

**Algorithmus 1** Minimierung der  $L_2$ -Norm zwischen stückweise linearen Funktionen unter waagerechten Verschiebungen

---

**Eingabe:**  $f, g$  stückweise linear und  $p$ -periodisch

**Ausgabe:**  $\min_{t \in [0, p]} \int_0^p (f(x-t) - g(x))^2 dx$

- 1: Fixiere eine Lage von  $g$
  - 2:  $min :=$  undefiniert
  - 3: **Für** jede Lage von  $f$ , in der eine gemeinsame Übergangsstelle mit  $g$  existiert
  - 4:  $X := Y := Z := 0$
  - 5: **Für** jedes Streifenintervall  $[l, r]$  benachbarter Übergangsstellen
  - 6: Bestimme  $F(t) := x \cdot t^2 + y \cdot t + z$  gemäß Formel (2.3) /\*  $L_2$ -Norm zwischen den Kurven im aktuellen Streifen als Funktion des Verschiebeparameters \*/
  - 7:  $X := X + x, Y := Y + y, Z := Z + z$
  - 8:  $t_0 := -\frac{Y}{2X}$  /\* Stelle des lokalen Minimums von  $F_{\text{ges}}(t) = X \cdot t^2 + Y \cdot t + Z$ , der  $L_2$ -Norm in  $[0, p]$  als Funktion des Verschiebeparameters \*/
  - 9: **Falls**  $t_0 \notin [0, t^+]$  **dann**
  - 10: **Falls**  $F_{\text{ges}}(0) < F_{\text{ges}}(t^+)$  **dann**
  - 11:  $t_0 := 0$
  - 12: **sonst**
  - 13:  $t_0 := t^+$
  - /\*  $F_{\text{ges}}(t_0)$  ist die minimale  $L_2$ -Norm für die aktuelle Ausgangslage von  $f$  \*/
  - 14: **Falls**  $min =$  undefiniert oder  $F_{\text{ges}}(t_0) < min$  **dann**
  - 15:  $min := F_{\text{ges}}(t_0)$
  - 16: **Ergebnis:**  $min$
- 

### 2.1.4.2 Startpunktauswahl

Die zweite Möglichkeit zur näherungsweisen Berechnung der PKF-Metrik ist, unter den Werten des Rotationsparameters  $t$  in Definition 2.3 eine Auswahl zu treffen.

Blicken wir noch einmal zurück auf den Beginn von Abschnitt 2.1.3.2. Dort wurde festgestellt, daß für zwei kongruente Polygone die Polarkoordinatenfunktionen genau dann identisch sind, wenn eines der Polygone so gedreht wird, daß korrespondierende Punkte auf der Peripherie dem gleichen Winkelparameter  $\varphi$  zugeordnet werden. Sie entsprechen den Startpunkten der (verschobenen) Funktionen  $pkf$ . Es ist naheliegend, in der Praxis die *Eckpunkte* der Polygone als solche Startpunkte zu benutzen und über deren Auswahl zu minimieren. Dafür spricht,

- a) daß sie in einem Laserscan leicht identifizierbar sind (sofern ihr Innenwinkel deutlich verschieden von  $180^\circ$  ist; siehe auch weiter unten Punkt „Spezielle Kriterien für die Startpunktwahl“)
- b) und daß es nur  $\mathcal{O}(n \cdot m)$  solche Startpunktpaare gibt (für Polygone mit  $n$  bzw.  $m$  Ecken), die sich effizient durchprobieren lassen.

Man beachte, daß bisher stets aus der Sicht (näherungsweise) *kongruenter* Polygone argumentiert wurde und was dafür getan werden muß, um ihre Übereinstimmung zu erkennen. Für (intuitiv) nicht ähnliche Polygone macht es keinen Sinn, nach korrespondierenden Startpunktpaaren zu suchen. Die Beschränkung auf Ecken als Startpunkte muß

hier im Rahmen der Freiheit gesehen werden, die eine Ähnlichkeitsfunktion bei der Festlegung der Funktionswerte für ihre Argumente hat (vgl. Abschnitt 1.3.3). Möglicherweise gibt es zwar außer den Eckpunkten noch andere Startpunkte auf der Peripherie, für die sich die Polarkoordinatenfunktionen noch weniger unterscheiden. Im Gegensatz zu einem Matching-Algorithmus muß aber eine Ähnlichkeitsfunktion nicht unbedingt die Polygone bestmöglich aufeinander abbilden.

Bei diesem vereinfachten Ansatz zur Berechnung der PKF-Metrik geht die Dreiecksungleichung als wichtige Metrikeigenschaft verloren. Der Grund ist, daß beim Vergleichen zweier Polygonpaare  $(\mathcal{A}, \mathcal{B})$  und  $(\mathcal{B}, \mathcal{C})$  die Summe der Rotationswinkel  $t_{AB}$  und  $t_{BC}$  zum jeweiligen Ausrichten zweier Eckpunkte aufeinander keinen gültigen Rotationswinkel  $t_{AC}$  darzustellen braucht. Dreht man also  $\mathcal{A}$  um  $t_{AB} + t_{BC}$ , so gibt es im allgemeinen kein Eckpunktpaar von  $\mathcal{A}$  und  $\mathcal{C}$ , das hierdurch bis auf Verschiebung auf demselben (bei  $p$  startenden) Strahl liegen würde.

Daher muß man sich mit der Eigenschaft begnügen, daß die Ähnlichkeitsfunktion zumindest intuitive Ähnlichkeit modelliert. Die Metrikeigenschaften der Identität (1.4) und der Symmetrie (1.3) bleiben auch bei der Einschränkung auf Ecken als Startpunkte erhalten.

**Spezielle Kriterien für die Startpunktwahl.** Um das spezifische Bild, das ein Laserscanner von seiner Umgebung liefert, angemessen zu interpretieren, kann man die Auswahl von Startpunkten weiter einschränken.

Da die Startpunkte durch die Rotation eines der beiden Polygone möglichst gut aufeinander abgebildet werden, werden sie als Kandidaten für korrespondierende Punkte angesehen. Daher kommen für sie nur Punkte in Frage, die nicht durch Rauschen oder andere Laserungenauigkeiten entstanden sind.

Bild 2.9 zeigt Startpunkte, die aufgrund ihrer Umgebung ungeeignet sind. In (a) besteht eine gewisse Wahrscheinlichkeit, daß der eingekreiste Punkt in Wirklichkeit kollinear mit seinen beiden Nachbarn liegt und daher als Eckpunkt nicht vorkommen dürfte. In diesem Fall gäbe es im anderen Polygon höchstwahrscheinlich keine Entsprechung für diesen Punkt, und ein Matchen würde fehlschlagen.

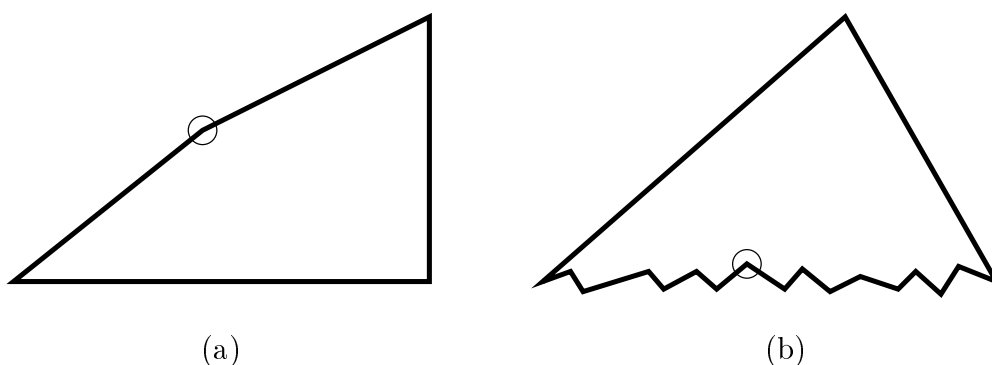


Abbildung 2.9: Ungünstige Ecken des Scanbildes als Startpunkte der PKF

In (b) läßt die hohe Anzahl von Meßpunkten auf engem Raum darauf schließen, daß es sich um eine verrauschte Darstellung handelt. Selbst wenn im zweiten Polygon eine ähnliche Situation vorläge und eine Punktkorrespondenz prinzipiell möglich wäre, so wäre

die Auswahl des entsprechenden Punktes mit großen Risiken bzw. Fehlern behaftet, da vermutlich in einer ganzen Umgebung potentielle Korrespondenzpunkte zu finden wären.

Folgende Punkte sind also als Startpunkte ungeeignet:

1. Punkte, die etwa kollinear mit ihren Nachbarn liegen, d.h. an denen ein Innenwinkel von etwa  $180^\circ$  besteht, und
2. Punkte, die von ihren Nachbarn weniger als eine bestimmte Schranke entfernt sind; letztere sollte sich eventuell an der Genauigkeit des Lasers orientieren.

**Startpunktauswahl nur in einem Polygon.** Eine ganz andere Methode zur Startpunktauswahl betrachtet den häufigsten Vergleichsfall bei der Roboterlokalisierung: Ein Anfrageskelett wird mit einem Zellskelett verglichen. Beide unterscheiden sich gravierend in ihrer Qualität: Zellskelette sind aufgrund exakter Daten (der Fabrikumgebung) im Pre-processing entstanden und liegen damit in unverrauschter Form vor. Das Anfrageskelett ist jedoch aus einem Laserscan hervorgegangen und fehlerbehaftet. Daher bietet es sich an, nicht in beiden Polygonen nach Startpunkten zu suchen, sondern im Zellskelett einen Punkt fest vorzugeben. Dessen Eckpunkte sind „echt“, weshalb ein beliebiger davon sich – im Fall der Ähnlichkeit bzw. Kongruenz – auch im Anfrageskelett wiederfinden wird. Die Umkehrung gilt nicht: Ein Laserscanner-Bild liefert (selbst nach einer entsprechenden Filterung) in der Regel mehr Punkte als real vorhanden.

Mit dieser Vorgehensweise ist die Ähnlichkeitsfunktion zwar nicht mehr symmetrisch, was aber durch die Ungleichheit der Eingabepolygone auch nicht sinnvoll erscheint.

**Eigentliche Differenzbestimmung.** Jede der  $\mathcal{O}(m \cdot n)$  Auswahlen des Verschiebeparameters  $t$  ergibt eine konkrete Lage der Graphen von  $\text{pkf}_A$  und  $\text{pkf}_B$ . Dafür ist nun noch die  $L_2$ -Norm zu bestimmen.

Da die Gleichung der Polarkoordinatenfunktion sich an jeder Übergangsstelle ändert, müssen in einem *Plane Sweep-Verfahren* die überlagerten Funktionen „durchstreift“ werden. Jede Undifferenzierbarkeitsstelle definiert einen Haltepunkt. Damit ist das Intervall  $[0, 2\pi]$  in Abschnitte  $[\varphi_l, \varphi_r]$  aufgeteilt, innerhalb deren beide Funktionen jeweils eine feste Gleichung der Form (2.1) haben. Die gesamte Integraldifferenz kann also über diese Abschnitte aufsummiert werden.

Durch die Verwendung der  $L_2$ -Norm, die das Quadrat der Differenz der beiden Funktionen enthält, muß man sich keine Gedanken über Schnittpunkte der Graphen machen, die für die Integralberechnung weitere Haltepunkte bedeuten würden. Daher kann man das Integral direkt ausrechnen und erhält mit  $c_{1,2} = \sin \beta_{1,2} \cdot |pP_i|_{1,2}$  gemäß Formel (2.1)

$$\begin{aligned} & \int_{\varphi_l}^{\varphi_r} \left( \frac{c_1}{\sin(\varphi + \beta_1)} - \frac{c_2}{\sin(\varphi + \beta_2)} \right)^2 d\varphi = \\ & \int_{\varphi_l}^{\varphi_r} \left( \frac{c_1^2}{\sin^2(\varphi + \beta_1)} + \frac{c_2^2}{\sin^2(\varphi + \beta_2)} - \frac{2c_1c_2}{\sin(\varphi + \beta_1) \cdot \sin(\varphi + \beta_2)} \right) d\varphi = \\ & \left[ -c_1^2 \cdot \cot(\varphi + \beta_1) - c_2^2 \cdot \cot(\varphi + \beta_2) - 2c_1c_2 \cdot \frac{\log \sin(\varphi + \beta_2) - \log \sin(\varphi + \beta_1)}{\sin(\beta_1 - \beta_2)} \right]_{\varphi_l}^{\varphi_r} \end{aligned}$$

für  $\beta_1 \neq \beta_2$  und

$$\int_{\varphi_l}^{\varphi_r} \left( \frac{c_1 - c_2}{\sin(\varphi + \beta)} \right)^2 d\varphi = [-(c_1 - c_2)^2 \cdot \cot(\varphi + \beta)]_{\varphi_l}^{\varphi_r}$$

für  $\beta_1 = \beta_2 =: \beta$ . Die technischen Einzelheiten über das Transformieren des globalen Wertes  $\varphi \in [0, 2\pi]$  auf den lokalen Wert  $\varphi \in [0, \alpha]$  innerhalb eines Sektors sind hier ausgelassen.

Alle Berechnungen zwischen zwei Haltepunkten können aufgrund konstant vieler Schnittpunkte in konstanter Zeit durchgeführt werden. Damit ist die Integralberechnung mit einem Zeitbedarf von  $\mathcal{O}(m + n)$  möglich, denn es gibt  $m + n$  Haltepunkte (jede Ecke eines der beiden Polygone stellt einen solchen Punkt dar). Benutzt man alle Paare von Ecken der Eingabepolygone als Startpunkte der Polarkoordinatenfunktionen, so erhält man  $m \cdot n$  Integralberechnungen.

### 2.1.4.3 Komplexität der Berechnung

Bei beiden Berechnungsmethoden – der linearen Approximation und der diskreten Minimierung durch Startpunktauswahl – sind zunächst der Kern, der Schwerpunkt und der spezifische Kernpunkt der Eingabepolygone zu bestimmen. Wie gesehen, ist dies in der Zeit  $\mathcal{O}(k \cdot \log k)$  möglich ( $k = \max(m, n)$ ).

Die eigentliche Differenzbestimmung kostet ebenfalls für beide Verfahren den gleichen Zeitaufwand: Für  $m \cdot n$  Lagen der Graphen der beiden Polarkoordinatenfunktionen sind in der Zeit  $\mathcal{O}(m + n)$  Integraldifferenzen zu bestimmen. Die Minimierung im Approximationsverfahren benötigt dabei nicht mehr Zeit, da – wie wir gesehen haben – die minimale Integraldifferenz zwischen zwei Eventpunktlagen in einem einzigen Durchlauf über das Intervall  $[0, 2\pi]$  berechnet werden kann, also in Zeit  $\mathcal{O}(m + n)$ .

Der Gesamtaufwand ist in Tabelle 2.2 zusammengefaßt.

Teilschritt	Zeitaufwand
1. Kernberechnung	$2 \cdot \mathcal{O}(k \cdot \log k) = \mathcal{O}((m + n) \cdot \log(m + n))$
2. Schwerpunktberechnung	$2 \cdot \mathcal{O}(k) = \mathcal{O}(m + n)$
3. Bestimmung des spezifischen Kernpunktes	$2 \cdot \mathcal{O}(k) = \mathcal{O}(m + n)$
4. Differenzbestimmung	$mn \cdot \mathcal{O}(m + n) = \mathcal{O}(mn \cdot (m + n))$

Tabelle 2.2: Laufzeit der Berechnungen zur PKF-Ähnlichkeitsfunktion

Der Speicheraufwand ist dabei linear: es muß im wesentlichen der Kern gespeichert werden.

Wie in Abschnitt 2.1.4.2 angedeutet, genügt es bei unterschiedlicher Qualität der Eingabepolygone in der Roboterlokalisierung, nur in einem der Polygone (dem verrauschten) nach korrespondierenden Startpunkten zu suchen, während im anderen Polygon dieser Punkt festgehalten wird. Dann sind nur  $k$  Integralberechnungen notwendig, und der Aufwand reduziert sich auf  $\mathcal{O}(k \log k + k \cdot (m + n)) = \mathcal{O}((m + n)^2)$ .

### 2.1.5 Eigenschaften der PKF-Ähnlichkeitsfunktion; Ausblick

Die Stärken und Schwächen der vorgestellten Ähnlichkeitsfunktion lassen sich wie folgt zusammenfassen:

Die PKF-Metrik mißt die Ähnlichkeit zweier sternförmiger Polygone. Sie greift dabei ausschließlich auf Abstände und Winkelgrößen des Polygons zurück, die bei Translationen und Rotationen nicht verändert werden. Dadurch ist die Metrik **translations- und rotationsinvariant**.

Will man bewußt auf die Rotationsunabhängigkeit verzichten und nur verschobene Polygone als kongruent zulassen, so wählt man einen festen gemeinsamen Startwinkel für beide Polygone. Der Aufwand zur eigentlichen Abstandsberechnung wird dann linear; zusammen mit der (immer noch notwendigen) Kernbestimmung ergibt sich  $\mathcal{O}((m+n) \cdot \log(m+n))$ .

Eine Skalierung verändert als Ähnlichkeitstransformation innere Abstände, nicht aber Winkelgrößen innerhalb der Polygone. Dadurch bleibt der spezielle Kernpunkt derselbe, und der Graph der Polarkoordinatenfunktion wird senkrecht nach oben oder unten verschoben. Möchte man hingegen die Ähnlichkeitsfunktion auch invariant unter Skalierungen machen, so betrachte man als Funktionswert der PKF nicht den (totalen) Abstand vom Kernpunkt zum Randpunkt, sondern den Abstand im Verhältnis zum Polygonumfang:

$$\tilde{r}(\varphi) = \frac{c}{\sin(\varphi + \beta)} \cdot \left( \sum_{i=1}^n |P_i P_{i+1}| \right)^{-1} \quad \text{mit } P_{n+1} := P_1.$$

Denn bei einer Skalierung eines Polygons um einen Faktor  $s$  werden alle Abstände  $|pP_i|$  vom Kernpunkt um den Faktor  $s$  verändert, wie auch die Längen aller Seiten und daher auch der Umfang des Polygons. Somit ist der Wert  $\tilde{r}(\varphi)$  von Skalierungen unberührt.

Anstatt laut Definition über alle Werte des Rotationsparameters  $t$  zu minimieren, wurden für die Praxis zwei vereinfachende Möglichkeiten zur Berechnung gezeigt: Zum einen wurde aus der PKF-Metrik eine zweite Metrik abgeleitet, die sich exakt und effizient berechnen läßt, aber etwas von der Intuition abweicht, die zur PKF geführt hatte. Zum anderen kann man aus allen Werten von  $t$  diejenigen auswählen, die den Ecken der Polygone entsprechen. In der **zweiten Variante** geht die Eigenschaft der **Dreiecksungleichung verloren**. Die Ähnlichkeitsfunktion ist aber **positiv definit** und **symmetrisch**.

Die Berechnung der Ähnlichkeit von Polygonen über die PKF-Metrik erwies sich als sehr **robust gegenüber verrauschten Daten**. Da es sich bei der Fläche unter einer Kurve gewissermaßen um ein *ganzheitliches* Maß handelt, ist der lokale Einfluß von Meßfehlern begrenzt. Durch die heuristische Suche nach einem speziellen Kernpunkt, wie in Kapitel 2.1.3.1 vorgeschlagen, sind dessen Lage und der daraus resultierende Verlauf des Graphen der PKF relativ resistent gegenüber unglatten Polygonkanten oder fehlerhaften zusätzlichen Eckpunkten. Dies hat sich in praktischen Tests bestätigt. Dazu wurden beide Varianten der Ähnlichkeitsfunktion implementiert; einige vergleichende Beispiele sind im Anhang zu finden.

Andere Erfahrungen wurden zum Beispiel mit der in [Maes94] vorgeschlagenen Polygondistanz über String-Matching gemacht. Dort haben redundante Eckpunkte unter Umständen negativen Einfluß auf das Vergleichsergebnis. Siehe dazu auch Abschnitt 3.2.

Zum Abschluß noch ein paar Bemerkungen zum Ausbau der PKF-Metrik zu einem Matching-Verfahren. Die ausgewählten speziellen Kernpunkte werden bei der PKF-Berechnung implizit als Korrespondenzpunkte angenommen. Man stellt sich also zunächst vor, die Polygone werden übereinandergelegt, so daß diese Punkte zur Deckung kommen. Anschließend wird unter den Parametern  $t$  derjenige ausgewählt, für den die Drehung des ersten Polygons  $\mathcal{A}$  um  $t$  die bestmögliche Annäherung von  $\text{pkf}_{\mathcal{A}}$  an  $\text{pkf}_{\mathcal{B}}$  ergibt. Der resultierende Wert  $t_{\min}$  entspricht dann einer Drehung von  $\mathcal{A}$ , bei der die Polygone bestmöglich *gematcht* werden, allerdings nur unter allen Werten von  $t$ , über die minimiert wurde.

Sind also  $p_A$  und  $p_B$  die beiden berechneten speziellen Kernpunkte und  $t_{\min}$  der gefundene Wert für  $t$ , so kann folgende Ausgabe als Resultat eines Matchingalgorithmus' betrachtet werden:

1. Verschiebe  $\mathcal{A}$  um den Vektor  $\overrightarrow{p_A p_B}$
2. Drehe  $\mathcal{A}$  gegen den Uhrzeigersinn um  $t_{\min}$ .

Wie bereits in Abschnitt 2.1.4.2 erläutert, ist bei einer Einschränkung der potentiellen Startwinkel  $t$  auf solche, die Ecken der Polygone entsprechen, nicht garantiert, daß  $t_{\min}$  die optimale Rotation des Polygons  $\mathcal{A}$  zum Matchen auf  $\mathcal{B}$  darstellt.

## 2.2 Ein flächenorientierter Ansatz

Die Polarkoordinatenfunktion eines Polygons ist nach Angabe eines speziellen Kernpunktes  $p$  bereits eindeutig definiert: der Abstand eines Peripheriepunktes zu  $p$  ist von keinen weiteren Funktionsparametern abhängig. Drehungen des Polygons bedeuten nur eine Veränderung der Lage des Horizontalstrahls relativ zum Polygon. Dieser Strahl, dem der Winkel  $0^\circ$  zugewiesen wurde, ist verantwortlich dafür, wo die Messung des Kernpunktabstandes und damit die Aufzeichnung der PKF beginnt. Eine Polygondrehung entsprach dadurch lediglich einer waagerechten Verschiebung des Funktionsgraphen.

Diese einfache Form der Rotationsabhängigkeit ist leider nicht selbstverständlich. Wir werden in diesem Abschnitt eine Beschreibung eines Polygons durch eine Funktion angeben, die von mehreren Parametern abhängt. Deren Veränderung hat wesentlich weitreichendere Konsequenzen auf den Graphen als nur eine Horizontalverschiebung.

Viele Darstellungsfunktionen beschreiben ein Polygon, indem jedem Punkt entlang der Peripherie ein Wert zugewiesen wird. Wir wollen allgemein eine solche Funktion *absolut* nennen, wenn der Funktionswert nur vom Argument und keinem weiteren Parameter abhängt. Andere Funktionen, wie die folgende *Flächeninhaltsfunktion*, legen auf der Peripherie einen Nullpunkt fest (ein Parameter der Funktion) und liefern als Wert einen bis zum aktuellen Peripheriepunkt akkumulierten Betrag zurück, z.B. einen überstrichenen Flächeninhalt. Es wird sich zeigen, daß eine auf diese Weise zusätzlich parametrisierte Funktion nur mit großem Aufwand und erheblichen Einschränkungen handhabbar ist. Die Berechtigung, sich dennoch ausführlich damit zu beschäftigen, liegt offenbar darin, daß der folgende Ansatz intuitiv sehr naheliegend ist.

### 2.2.1 Die Flächeninhaltsfunktion

In Abschnitt 2.1.1 wurde die Polarkoordinatenfunktion aus der Arbeitsweise eines Laser-scanners so abgeleitet: Der Laserstrahl liefert in regelmäßigen Winkelabständen Entfernungsmesspunkte zurück, die als Funktion vom Winkel aufgetragen werden.

Eine andere Interpretation betrachtet den vom Laserstrahl überstrichenen Flächeninhalt. Analog zur Polarkoordinatenfunktion muß dazu ein Startpunkt auf der Peripherie vorgegeben werden, an dem die Messung des Flächeninhalts beginnt. Nach einem vollen Umlauf des Laserstrahls ist die überstrichene Fläche gleich der des gesamten Polygons.

Wir werden erneut den Winkel gegenüber einer vorgegebenen Nullrichtung als die Größe wählen, von der die Meßgröße abhängig gemacht wird. Zu einer Alternative dazu siehe Abschnitt 2.3.3.

Der überstrichene Flächeninhalt wird also als Funktion einer reellen Zahl aus  $[0, 2\pi[$  dargestellt. Ein Beispiel dazu zeigt Abbildung 2.10.

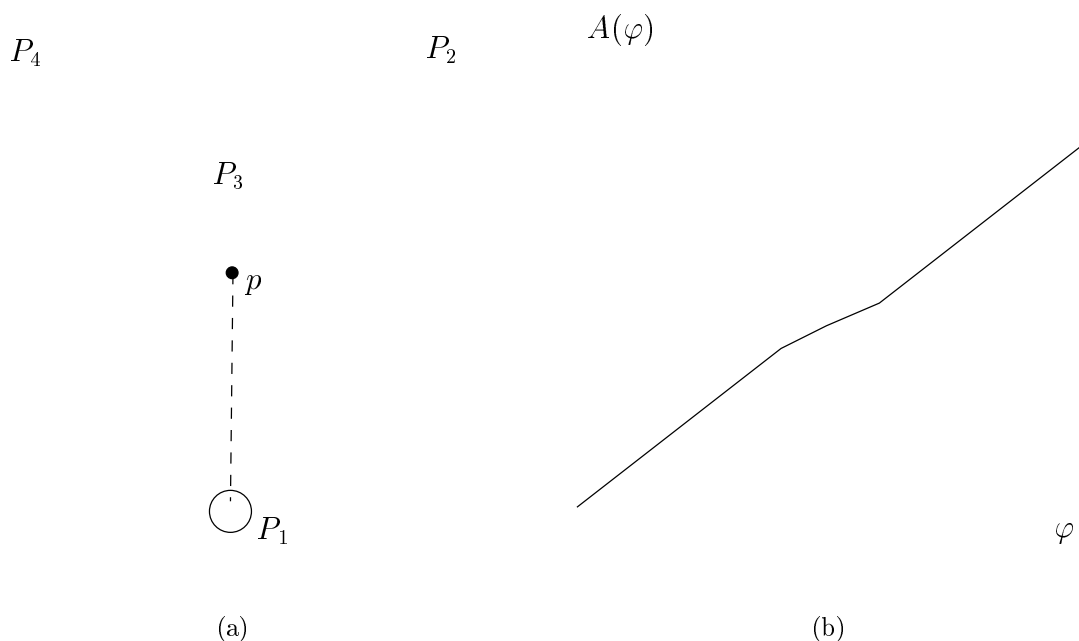


Abbildung 2.10: Ein Polygon (a) und der überstrichene Flächeninhalt  $A$  (startend in  $P_1$ ) als Funktion vom Strahlwinkel  $\varphi$  (b)

Es ergibt sich folgende verbale Definition:

**Definition 2.6 (Flächeninhaltsfunktion)** Sei  $\mathcal{P} = (P_1, \dots, P_n)$  ein sternförmiges Polygon mit Flächeninhalt  $A_{\mathcal{P}}$ , einem Kernpunkt  $p$  und einem Peripheriepunkt  $P_1$  (**Startpunkt**).

Die Abbildung  $f: [0, 2\pi[ \rightarrow [0, A_{\mathcal{P}}[$ , die einem Winkel  $\varphi$  den vom Strahl  $p \rightarrow P_1$  bei der Drehung um  $\varphi$  gegen den Uhrzeigersinn überstrichenen Flächeninhalt zuweist, heißt **Flächeninhaltsfunktion** (Abk. **FIF**) von  $\mathcal{P}$  (bez.  $p$ ).

Für  $\varphi = 2\pi$  ist die Funktion formell nicht definiert. Der Peripheriepunkt, der diesem Winkel entspricht, ist der Punkt  $P_1$ , an dem der Laserstrahl seine Bewegung beginnt.

Daher ist der überstrichene Flächeninhalt dort Null. Wegen  $\lim_{\varphi \rightarrow (2\pi)^-} f(\varphi) = A_{\mathcal{P}}$  setzen wir  $f$  aber durch  $f(2\pi) := A_{\mathcal{P}}$  fort (siehe auch Punkt „Stetigkeit“ in folgendem Abschnitt).

Zur Berechnung einer noch zu definierenden Metrik ist erneut eine lineare Approximation dieser Funktion von Interesse. Dazu verbinden wir wieder benachbarten Übergangsstellen des Funktionsgraphen durch Strecken, d.h. alle die Stellen, die einem Eckpunkt des Polygons entsprechen.

**Definition 2.7 (linear approximierte Flächeninhaltsfunktion)** *Es gelten die Voraussetzung aus Definition 2.6 für ein Polygon  $\mathcal{P}$  mit der Flächeninhaltsfunktion  $f$ . Die linear approximierte Flächeninhaltsfunktion (Abk.  $\mathbf{FIF}_{lin}$ ) entsteht durch das Verbinden der Übergangsstellen des Graphen von  $f$  durch Geradensegmente.*

Die Approximationskurve ist in Bild 2.10(b) dünn eingezeichnet. Zur Frage des Einfügens zusätzlicher Stützstellen und zu Approximationsfehlern siehe Kapitel 2.2.3, Abschnitt „Monotonie“.

## 2.2.2 Analytische Beschreibung

Die Funktionswerte der (nichtapproximierten) FIF erhält man wie folgt: Der Flächeninhalt, den der Strahl von  $p$  bei Drehung gegen den Uhrzeigersinn bis zum Erreichen des Peripheriepunktes  $P$  überstreicht, ist die Summe der Inhalte aller vollständig überstrichenen Dreiecke und des Flächeninhalts  $A(\varphi)$  von Dreieck  $\triangle pP_iP$ . Mit den in Abbildung 2.11(b)

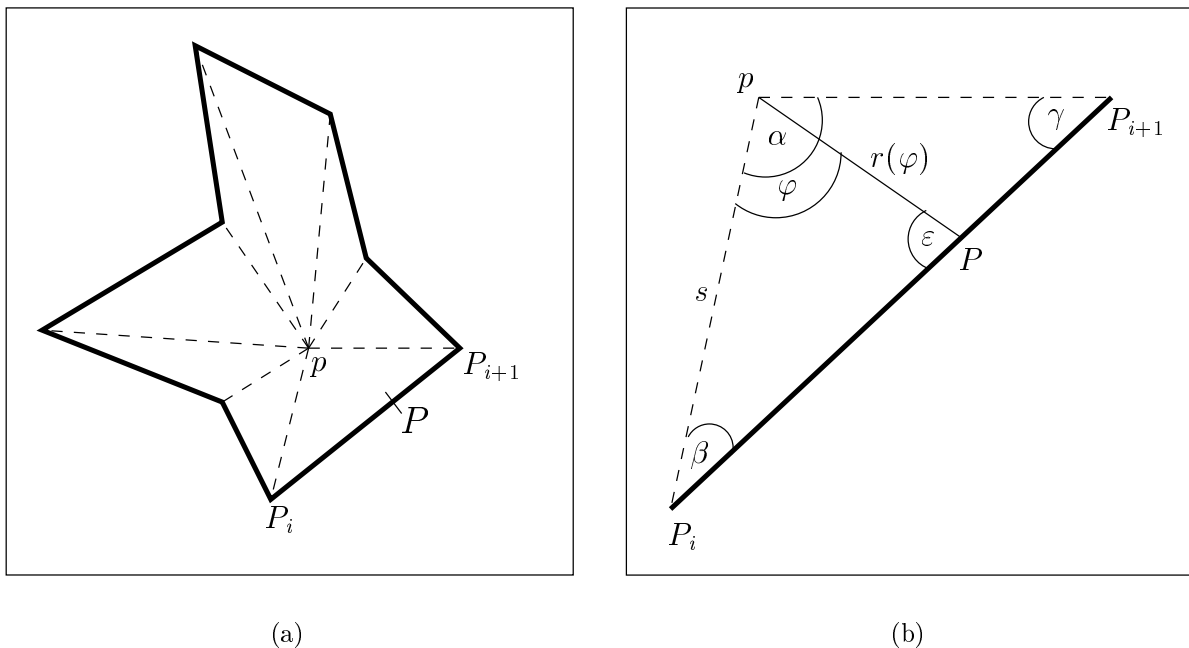


Abbildung 2.11: Sektorenzerlegung eines Polygons (a); ein Ausschnitt (b)

bezeichneten Größen gilt:

$$A(\varphi) = \frac{1}{2} \cdot s \cdot r(\varphi) \cdot \sin \varphi = \frac{1}{2} \cdot s \cdot \frac{s \cdot \sin \beta}{\sin(\varphi + \beta)} \cdot \sin \varphi = c \cdot \frac{\sin \varphi}{\sin(\varphi + \beta)}, \quad (2.4)$$

wobei  $c := 1/2 \cdot s^2 \cdot \sin \beta$  alle innerhalb eines Sektors konstanten Größen enthält. Wie bei der PKF ist  $\varphi$  hier zunächst ein lokaler Winkel im Sektor. Später muß vom Argument aus  $[0, 2\pi]$  der FIF der Anfangswinkel desjenigen Sektors abgezogen werden, in den der Strahl mit Winkel  $\varphi$  zeigt.

Ebenso ist  $A(\varphi)$  nur der lokale Flächeninhalt. Da die FIF eine „akkumulierende“ (d.h. im Sinne der Einleitung zu Abschnitt 2.2 keine *absolute*) Funktion ist, gilt für den Funktionswert eines Winkels  $\varphi$ , dessen zugehöriger Strahl die Polygonperipherie zwischen  $P_k$  und  $P_{k+1}$  schneidet:

$$f(\varphi) = \sum_{i=1}^{k-1} A_i + A(\varphi), \quad (2.5)$$

wenn  $A_i$  der Flächeninhalt des Dreiecks  $\Delta pP_iP_{i+1}$  ist.

Der Ausdruck  $r(\varphi)$  aus Formel (2.4) entspricht genau dem lokalen Funktionswert der Polarkoordinatenfunktion, der in Gleichung 2.1 angegeben wurde. Das läßt auf einen Zusammenhang zwischen der PKF und der FIF schließen, der in Abschnitt 2.3.1 behandelt wird.

### 2.2.3 Eigenschaften der Flächeninhaltsfunktion

Die FIF ist durch sehr spezielle analytische Merkmale gekennzeichnet.

**Stetigkeit, Differenzierbarkeit.** Die Anzahl der Übergangsstellen zwischen verschiedenen Abschnitten des Graphen der Flächeninhaltsfunktion (innere Punkte in Abbildung 2.10(b)) ist gleich der Anzahl der Ecken des Polygons vermindert um eins. Auch an diesen Stellen ist die Funktion stetig und differenzierbar:

Die Stetigkeit im Intervall  $[0, 2\pi]$  folgt daraus, daß für einen Eckpunkt  $P_k$  mit zugehörigem Strahlwinkel  $\alpha_k$  gilt:

$$\lim_{\varphi \rightarrow (\alpha_k)^-} f(\varphi) = \lim_{\varphi \rightarrow (\alpha_k)^+} f(\varphi) = \sum_{i=1}^{k-1} A_i = f(\alpha_k).$$

Mit der Fortsetzung der Funktion im Anschluß an Definition 2.6 gilt für alle Polygone:  $f(0) = 0$ ,  $f(2\pi) = A_p$ .

Die Funktion ist in ganz  $[0, 2\pi]$  sogar (einmal) *stetig differenzierbar*, denn an den Rändern zweier benachbarter Abschnitte gilt für die jeweiligen lokalen Flächeninhaltsfunktionen  $f_1$  und  $f_2$  in den Polygonektoren  $\Delta pP_iP_{i+1}$  und  $\Delta pP_{i+1}P_{i+2}$ :

$$f'_1(\alpha_1) = f'_2(0) = \frac{1}{2} \cdot |pP_{i+1}|^2.$$

Im Gegensatz zur Polarkoordinatenfunktion entstehen also an den Übergängen zwischen benachbarten Definitionsabschnitten der Funktion keine Knickstellen (siehe Bild 2.10(b)). In Abschnitt 2.3.1 werden wir die Ableitung der Funktion genau angeben.

Aus der Stetigkeit der FIF folgt sofort die Stetigkeit der  $FIF_{\text{lin}}$ , denn sie ist aus den Stützstellenwerten der FIF linear zusammengesetzt. Die Differenzierbarkeit geht natürlich verloren.

Ausnahmen der Differenzierbarkeit sind wiederum dann gegeben, wenn der Punkt  $p$  auf dem Rand des Kerns liegt. Der Graph der Flächeninhaltsfunktion hat dann für die Winkelargumente, die den kollinearen Kanten entsprechen, eine Undifferenzierbarkeitsstelle. Die Stetigkeit bleibt dort jedoch erhalten, da der Flächeninhalt sich nicht ändert, wenn der Endpunkt des Laserstrahles (in idealisierter Vorstellung) auf der kollinearen Kante entlangwandert. Man beachte, daß die *Entfernung* zu  $p$  sehr wohl wächst oder fällt, weshalb die PKF an solchen Stellen unstetig ist.

**Monotonie.** Aufgrund der Herkunft der FIF ist klar, daß sie im gesamten Definitionsintervall  $[0, 2\pi]$  streng monoton wachsend ist. Dafür ist es auch von Bedeutung, daß der Punkt  $P_1$  in Abbildung 2.10(a) zwei Rollen bei der Definition der Funktion übernimmt:

- Er ist der *Ausgangspunkt* für die Messung des überstrichenen Flächeninhalts. An diesem Punkt wird die Bogenlänge als Null definiert.
- Er ist der Punkt, dem der Flächeninhalt Null zugewiesen wurde (*Nullpunkt*).

Es wäre auch denkbar, diese Rollen zu trennen. Dann hätte der Graph der FIF an der Stelle  $\varphi$  (mit  $0 < \varphi < 2\pi$ ), an der der Nullpunkt überstrichen wird, einen Sprung auf Null, und es gälte dann  $f(0) > 0$ .

Da es sicher eher der Intuition entspricht, an dem Punkt den Flächeninhalt als Null zu definieren, an dem man dessen Messung beginnt, ist es sinnvoll, die beiden genannten Eigenschaften in einem Punkt („Startpunkt“) zu vereinen. Dennoch hat die Flächeninhaltsfunktion die beiden obenstehenden Parameter; sie ist also keine absolute Funktion im Sinne der Einleitung zu Kapitel 2.2. Das wird negative Folgen für die Effizienz der Berechnung einer daraus resultierenden Ähnlichkeitsfunktion haben.

In Definition 2.6 wurde vom Punkt  $p$  ausdrücklich nicht verlangt, daß er im *Innern* des Kerns liege. Die strenge Monotonie bleibt erhalten, wenn  $p$  kollinear mit einer Polygonkante ist.

Durch die strenge Monotonie treten im Kurvenverlauf keine Minima auf, an denen sich zusätzliche Stützstellen für eine lineare Approximation anbieten würden. Aus der Herleitung der Funktion ist unmittelbar einsichtig, daß der Graph (zwischen zwei Übergangsstellen) die Approximationsgerade genau dann schneidet, wenn  $\beta < \pi/2$  und  $\gamma < \pi/2$  gilt (vergleiche Abbildung 2.11(b)), denn genau dann befindet sich der Fußpunkt des Lotes von  $p$  auf die Gerade durch  $P_i$  und  $P_{i+1}$  im Innern der Strecke  $\overline{P_i P_{i+1}}$ .<sup>2</sup> In Bild 2.10(a) ist das bei den Strecken  $\overline{P_1 P_2}$  und  $\overline{P_4 P_1}$  der Fall.

Der ungünstigste Fall für den Fehler bei der linearen Approximation liegt vor, wenn die angegebene Bedingung nicht erfüllt ist. Dann wird der *Flächenzuwachs* entlang der Polygonkante entweder immer größer ( $\beta > 90^\circ$ , die exakte Kurve ist konvex) oder immer kleiner ( $\gamma > 90^\circ$ , die exakte Kurve ist konkav) und ist nicht konstant, wie es eine Approximationsgerade suggeriert. Bild 2.12 zeigt diese Fälle und den Verlauf der Flächeninhaltskurve in den jeweiligen Sektoren.

---

<sup>2</sup>Dies ist genau die Bedingung dafür, daß bei den Kurvenabschnitten der PKF ein lokales Minimum vorliegt. Zum genauen Zusammenhang siehe Abschnitt 2.3.2.2.

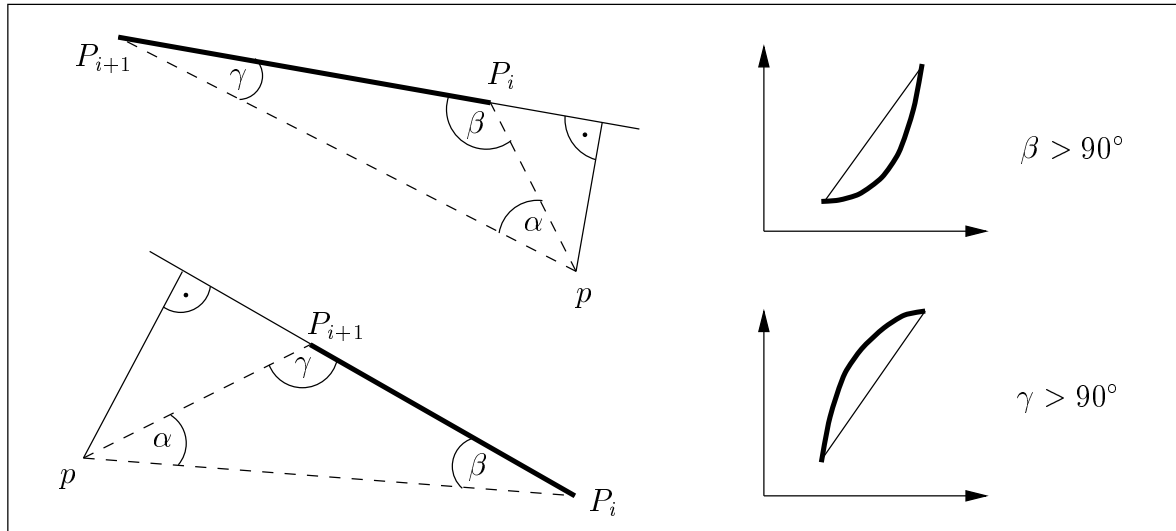


Abbildung 2.12: Konvexer oder konkaver Verlauf der FIF

**Translation, Rotation, Skalierung, Startpunktwahl.** Die Funktion  $f$  ist von einer Translation des Eingabepolygons unberührt, da nur Flächeninhalte betrachtet werden. Gleiches gilt für Rotationen, wenngleich die Rotationsabhängigkeit implizit in Form der Abhängigkeit vom Startpunkt zum Ausdruck kommt. Man hätte auch – wie bei der Polarkoordinatenfunktion – vereinbaren können, daß etwa  $P_1$  stets der Eckpunkt mit kleinstem Fahrstrahlwinkel gegenüber der Horizontalen sein soll. Dann wäre der Startpunkt festgelegt, aber der Funktionsgraph nunmehr rotationsabhängig.

Die Abhängigkeit von der Fixierung eines Startpunktes ist tatsächlich nicht zu beseitigen; wir werden im nächsten Kapitel bei der Definition der *Flächeninhaltsdistanz* daher über die Wahl dieses Punktes minimieren müssen.

Durch eine Skalierung eines Polygons verändert sich zwar  $f(0) = 0$  am linken Rand des Definitionsintervalls der Flächeninhaltsfunktion nicht, aber  $f(2\pi) = A_P$  wächst, wodurch der Graph der Funktion sich nach oben oder nach unten verlagert. Sie ist also skalierungssensitiv, wie es in der Roboterlokalisierung erwünscht ist.

## 2.2.4 Die Flächeninhaltsdistanz

Nachdem wir Polygone in eine flächeninhaltsbezogene funktionale Darstellung gebracht haben, können die resultierenden Funktionsgraphen mit einem bereits existierenden Distanzmaß verglichen werden. Um die Bestimmung der Flächeninhaltsfunktion eines Polygons eindeutig zu machen (abgesehen von der Wahl des Startpunktes  $P_1$ ), ist wiederum der Parameter  $p$  der Funktion zu fixieren – der spezielle Kernpunkt. Dabei gelten genau dieselben Kriterien wie bei der Polarkoordinatenfunktion in Kapitel 2.1.3.1. Festlegung 2.2 auf Seite 24 gilt also entsprechend.

### 2.2.4.1 Die FIF als quasiperiodische Funktion

Bei der Besprechung der Monotonieigenschaften wurde festgestellt, daß die Flächeninhaltsfunktion im Grunde von zwei Parametern abhängt: dem Ausgangspunkt für die

Messung der Fläche und dem Nullpunkt. Diese Punkte, die im allgemeinen irgendwo auf der Peripherie des Polygons liegen, können nicht rotationsunabhängig fixiert werden. Bei der späteren Ableitung einer Metrik müssen also alle Wahlen dieser Punkte berücksichtigt werden. Dabei gilt:

- Eine Veränderung des Meßausgangspunktes bewirkt eine waagerechte Verschiebung des Graphen der FIF
- Eine Veränderung des Meßnullpunktes bewirkt eine senkrechte Verschiebung des Graphen der FIF

Dieser Sachverhalt ist in Bild 2.13 am Beispiel der linear approximierten FIF dargestellt: Bei Bewegungen jeweils *gegen* den Uhrzeigersinn verschiebt sich der Graph von  $f$  nach links bzw. nach unten.

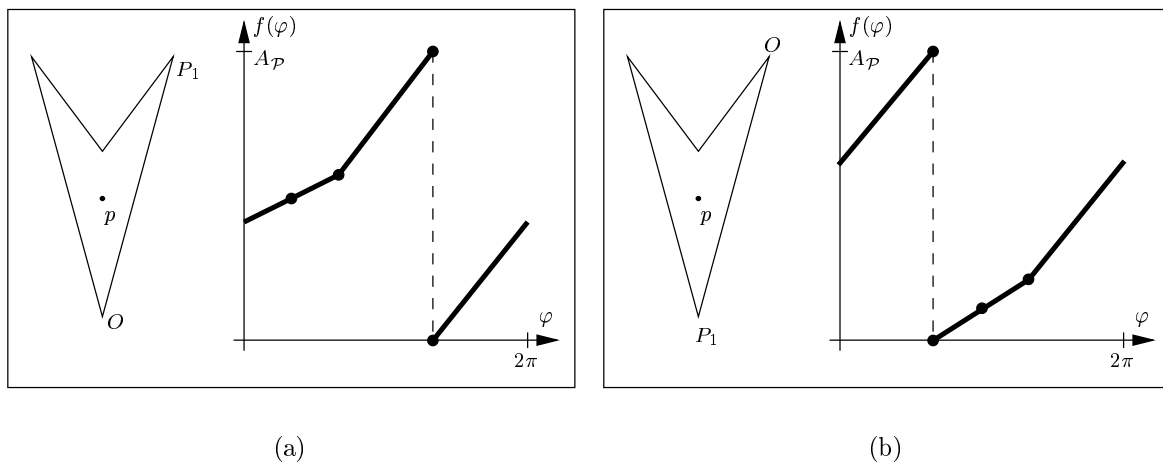


Abbildung 2.13: Verschiebung des Meßausgangspunktes  $P_1$  (a) und des Meßnullpunktes  $O$  (b) gegenüber Abbildung 2.10 (jeweils gegen den Uhrzeigersinn)

Es wurde bereits darauf hingewiesen, daß aus praktischen Gründen der Meßausgangspunkt und der -nullpunkt stets in einem Punkt  $P_1$  vereint sein sollen, wie das auch in Abbildung 2.10 der Fall ist. Eine Veränderung dieses Punktes bewirkt dann eine Überlagerung der beiden in Bild 2.13 dargestellten Bewegungen, also eine diagonale Verschiebung des Graphen von  $f$ . Aufgrund der beiden Funktionen von  $P_1$  muß dabei stets der Punkt  $(0,0)$  zu  $f$  gehören. Daher ist die Verschiebungsrichtung immer durch den Anstieg der Kurve in dem Punkt gegeben, der aktuell im Ursprung liegt; im Fall der  $FIF_{\text{lin}}$  ist es der Anstieg des Segments, das momentan durch den Ursprung geht (*Ursprungssegment*). Dabei wandert der Punkt  $(0,0)$  gegen die Verschiebungsrichtung auf der Kurve entlang. Bei der  $FIF_{\text{lin}}$  ist die Verschiebungsrichtung konstant, bis der anfängliche Punkt  $(0,0)$  mit einem der Stützpunkte zusammenfällt. Dann ändert sich die Verschiebungsrichtung entsprechend dem neuen Ursprungssegment. Auf diese Weise ist sichergestellt, daß der Graph stets durch die Punkte  $(0,0)$  und  $(2\pi, A_P)$  verläuft – zwei Invarianten der Funktion.

Hat der Punkt  $P_1$  einen vollen Umlauf auf der Polygonperipherie zurückgelegt, so ist der anfängliche Punkt  $(2\pi, A_P)$  des Graphen von  $f$  in den Ursprung gewandert.

Um die im Bild zu erkennenden Unstetigkeitsstellen durch das Verschieben zu umgehen, setzen wir die Flächeninhaltsfunktion geeignet fort. Da  $0 = f(0) \neq f(2\pi) = A_p$  gilt, ist eine periodische Fortsetzung ungünstig. Vielmehr muß der Graph z.B. im Intervall  $[2\pi, 4\pi]$  lückenlos an den Punkt  $(2\pi, A_p)$  anschließen. Dies erreicht man durch eine Abschwächung der Periodizität:

**Definition 2.8 (Quasiperiodizität)** Eine Funktion  $f: \mathbb{R} \rightarrow \mathbb{R}$  heißt *p-quasiperiodisch*, wenn es eine Konstante  $c$  gibt mit

$$\forall x \quad f(x+p) - f(x) = c.$$

$p$  heißt *Quasiperiode*,  $c$  *Periodenversatz* von  $f$ .

Es gilt also  $f(x+k \cdot p) = f(x) + k \cdot c$  für alle  $k \in \mathbb{Z}$ . Im Fall  $c = 0$  ist  $f$   $p$ -periodisch. Beispiele quasiperiodischer (aber nicht periodischer) Funktionen sind  $f(x) = \sin x + x$  ( $p = c = 2\pi$ ) und die Integerfunktion  $f(x) = [x]$  ( $p = c = 1$ ).

Wir setzen nun die Flächeninhaltsfunktion quasiperiodisch mit Quasiperiode  $2\pi$  fort; der Periodenversatz beträgt  $f(2\pi) - f(0) = A_p$ :

**Festlegung 2.9** Die Flächeninhaltsfunktion  $f: [0, 2\pi] \rightarrow [0, A_p]$  sei auf ganz  $\mathbb{R}$  durch  $f(x_0 + k \cdot 2\pi) := f(x_0) + k \cdot A_p$  für  $x_0 \in [0, 2\pi]$ ,  $k \in \mathbb{Z}_{\neq 0}$  eindeutig fortgesetzt.

#### 2.2.4.2 Versuch zur Definition einer Ähnlichkeitsfunktion

Die Ähnlichkeit zweier Polygone soll über einen Vergleich ihrer Flächeninhaltsfunktionen bestimmt werden. Dazu verwenden wir wie bei der PKF die Integralnorm der Ordnung 2 für stetige Funktionen ( $L_2$ -Norm).

Dabei ist der Abstand über die Wahl des Startpunktes  $P_1$  in beiden Polygonen zu minimieren. Es genügt nicht, diesen Parameter nur in einem Polygon zu verändern, da die Verschiebungsrichtungen für die beiden Flächeninhaltsfunktionen verschieden sind. Man kann also nicht etwa die Verschiebung des einen Graphen durch eine entgegengesetzte Verschiebung des anderen ausdrücken.

Für einen beliebigen Parameter  $r \in \mathbb{R}$  bezeichne  $f_r$  diejenige Funktion, die aus der Flächeninhaltsfunktion  $f$  (mit fester Wahl von  $P_1$ ) durch eine Verschiebung gemäß Abschnitt 2.2.4.1 nach links unten hervorgeht, und zwar mit einem Verschiebungsbetrag von genau  $r$  in  $x$ -Richtung. Ist also  $(x, y)$  ein Punkt von  $f$ , der durch die Verschiebung zu  $f_r$  auf den Punkt  $(x', y')$  (auf dem Graphen von  $f_r$ ) abgebildet wurde, so gilt:  $x - x' = r$ .

Aufgrund der Quasiperiodizität von  $f$  ist klar, daß  $r$  nur im Intervall  $[0, 2\pi]$  betrachtet zu werden braucht, denn der Term  $f_r$  ist  $2\pi$ -periodisch in  $r$ .

Da sich die Verschiebungsrichtungen bei einem Umlauf von  $P_1$  an allen Eckpunkten (d.h. an allen Übergangsstellen im Graphen) ändern, ist eine geschlossene Darstellung für  $f_r$  mittels  $f$  schwierig. Wie wir in Kapitel 2.2.4.4 sehen werden, genügt es, innerhalb eines Intervalls zweier benachbarter Übergangsstellen  $f_r$  analytisch zu beschreiben. Die Darstellung von  $f_r$  in  $[0, 2\pi]$  ergibt sich dann als Summe solcher Intervallgleichungen.

Bereits hier zeigt es sich, daß durch die beiden Parameter der Flächeninhaltsfunktion – der Meßausgangspunkt und der Nullpunkt – die Ableitung einer sinnvollen Polygondistanz sehr komplex wird. Die Auswirkungen der Veränderung dieser Parameter sind mathematisch recht aufwendig zu formulieren. Daher soll zunächst versuchsweise eine Ähnlichkeitsfunktion definiert werden:

**Definition 2.10 (Versuch)** Seien  $\mathcal{A}$  und  $\mathcal{B}$  zwei sternförmige Polygone, für die nach dem Verfahren vom Beginn des Abschnitts 2.2.4 jeweils ein spezieller Kernpunkt festgelegt wurde. Die diesbezüglichen Flächeninhaltsfunktionen seien  $f$  und  $g$  mit beliebig fixierten Startpunkten  $P_1$ .

Die Abbildung

$$s(\mathcal{A}, \mathcal{B}) := \min_{(r,t) \in [0,2\pi]^2} L_2(f_r, g_t) = \min_{(r,t) \in [0,2\pi]^2} \sqrt{\int_0^{2\pi} (f_r(x) - g_t(x))^2 dx}$$

heißt **Flächeninhaltsdistanz** für sternförmige Polygone.

Im Sinne von Abschnitt 1.3.2 handelt es sich dabei natürlich um eine Ähnlichkeitsfunktion (und keine Distanzfunktion in der strengen Bedeutung). Aus lexikalischen Gründen soll aber hier von der Flächeninhaltsdistanz gesprochen werden.

**Metrikeigenschaften.** Leider lassen sich bei dieser Definition nicht alle Metrikeigenschaften nachweisen. Die Identitätseigenschaft  $s(\mathcal{A}, \mathcal{B}) = 0 \iff \mathcal{A} = \mathcal{B}$  kann wie im Satz 2.13 aus Abschnitt 2.2.4.3 weiter hinten bewiesen werden. Die Symmetrie ergibt sich daraus, daß der Term für  $s(\mathcal{A}, \mathcal{B})$  symmetrisch in  $f_r$  und  $g_t$  ist, es ist also nichts zu zeigen.

Die Dreiecksungleichung gilt im allgemeinen nicht. Wir zeigen diesen Sachverhalt hier nur für die linear approximierte Flächeninhaltsfunktion  $\text{FIF}_{\text{lin}}$ . Man betrachte als Beispiel Abbildung 2.14. Dort sind drei zum Teil verrauschte gleichseitige Dreiecke unterschiedli-

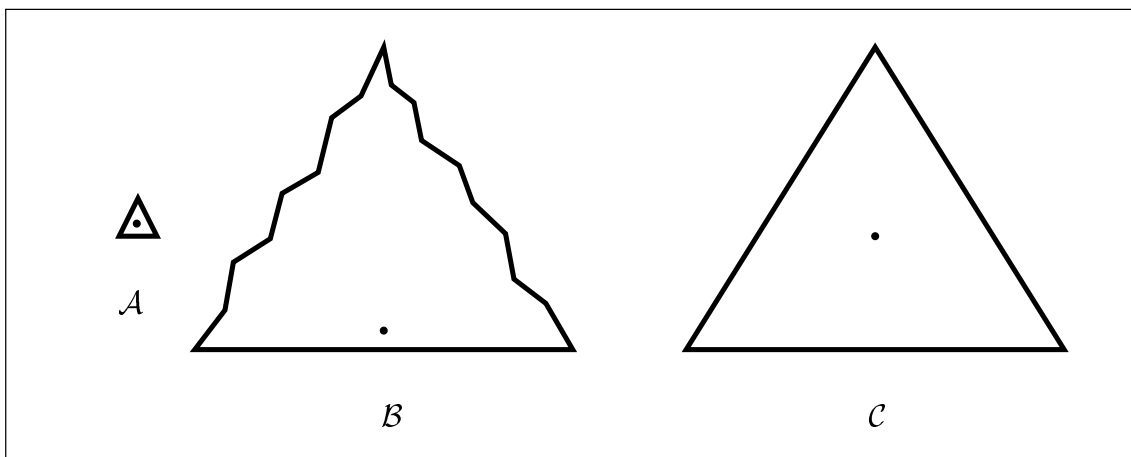


Abbildung 2.14: Zur Verletzung der Dreiecksungleichung

cher Größe angegeben. Der spezielle Kernpunkt  $p$  liege im zweiten Polygon aufgrund des Rauschens nah am Rand der unteren Seite. Abbildung 2.3 auf Seite 24 weiter vorne erläutert die Veränderungen des Kerns, die zu solchen extremen Kernpunktverschiebungen führen. Zur Vereinfachung der Rechnung wird das Polygon  $\mathcal{B}$  jedoch als Dreieck angenommen, was das Endergebnis nur quantitativ beeinflusst; siehe auch folgender Absatz.

Die zugehörigen  $\text{FIF}_{\text{lin}}$ -Graphen  $f$ ,  $g$  und  $h$  von  $\mathcal{A}$ ,  $\mathcal{B}$  und  $\mathcal{C}$  erfüllen die Eigenschaft  $f(2\pi) \ll g(2\pi) = h(2\pi)$ . Dies ist in idealisierter Form in den Abbildungen 2.15 und 2.16 dargestellt. Zur deutlicheren Herausstellung des Problems wurde  $f \equiv 0$  angenommen. Die Funktion  $g$  ist abschnittsweise konstant, was durch die Nähe des speziellen Kernpunktes

von  $\mathcal{B}$  zu einer der Polygonkanten näherungsweise erreicht ist. Die Vereinfachungen haben keinen qualitativen Einfluß auf die Verletzung der Dreiecksungleichung in diesem Szenario, da man sich dem dargestellten „Idealfall“ beliebig nähern kann.

Die Funktionen  $f$  und  $h$  sind nicht nur stückweise, sondern auf ganz  $[0, 2\pi]$  linear; eine Verschiebung bildet sie somit auf sich selbst ab, d.h.  $f_r \equiv f$ ,  $h_t \equiv h$  für alle  $r, t$ . Daher gilt (Abbildung 2.15)

$$\min_{(r,t) \in [0,2\pi]^2} \sqrt{\int_0^{2\pi} (f_r(x) - h_t(x))^2 dx} = \sqrt{\int_0^{2\pi} \left(\frac{x}{2\pi}\right)^2 dx} = \frac{1}{3}\sqrt{6\pi}.$$

Die Funktion  $g$  verändert sich hingegen bei einer abschnittswisen Verschiebung durch den

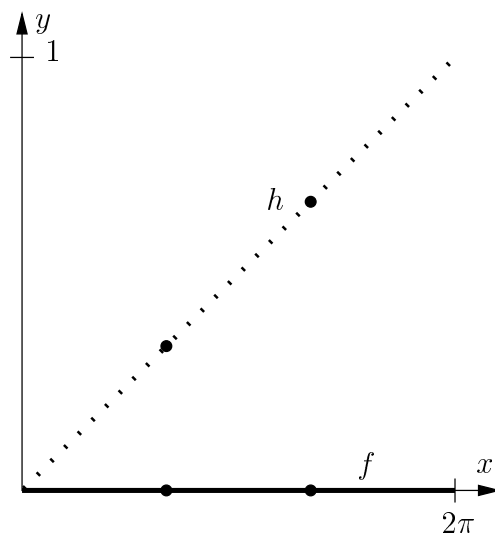


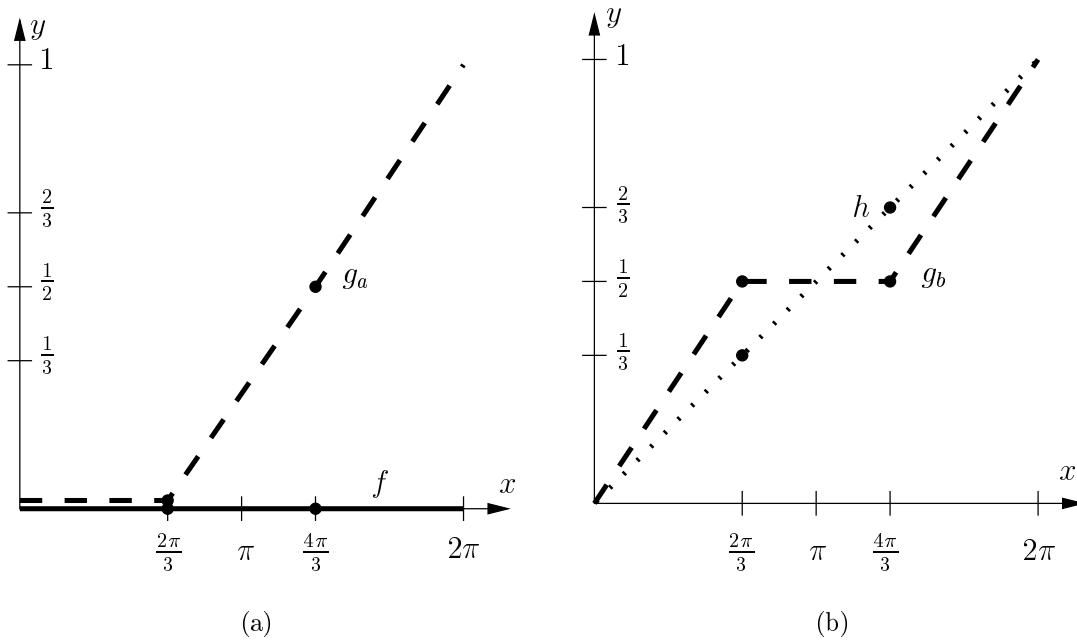
Abbildung 2.15: Minimale FIF-Distanz der Polygone  $\mathcal{A}$  und  $\mathcal{C}$

Ursprung. Die Lagen in den Abbildungen 2.16(a) bzw. (b) seien mit  $g_a$  bzw.  $g_b$  bezeichnet. Es gilt für Bild (a):

$$\sqrt{\int_0^{2\pi} (f(x) - g_a(x))^2 dx} = \sqrt{\int_{2\pi/3}^{2\pi} \left(\frac{3}{4\pi}x - \frac{1}{2}\right)^2 dx} = \frac{2}{3}\sqrt{\pi}.$$

Für Bild (b) erhält man

$$\sqrt{\int_0^{2\pi} (g_b(x) - h(x))^2 dx} = \sqrt{2 \cdot \left( \int_0^{2\pi/3} \left(\frac{3}{4\pi}x - \frac{1}{2\pi}x\right)^2 dx + \int_{2\pi/3}^{\pi} \left(\frac{1}{2} - \frac{1}{2\pi}x\right)^2 dx \right)} = \frac{1}{18}\sqrt{6\pi}.$$

Abbildung 2.16: FIF-Distanz der Polygone  $\mathcal{A}$  und  $\mathcal{B}$  (a) bzw.  $\mathcal{B}$  und  $\mathcal{C}$  (b)

Also folgt

$$\begin{aligned}
 \min_{(r,t) \in [0,2\pi]^2} L_2(f_r, g_t) + \min_{(r,t) \in [0,2\pi]^2} L_2(g_r, h_t) &\leq \frac{2}{3}\sqrt{\pi} + \frac{\sqrt{6}}{18}\sqrt{\pi} \\
 &< \frac{\sqrt{6}}{3}\sqrt{\pi} \\
 &= \min_{(r,t) \in [0,2\pi]^2} L_2(f_r, h_t). \quad \square
 \end{aligned}$$

Da wir später eine Variante der Flächeninhaltsdistanz herleiten wollen, die die Dreiecksungleichung erfüllt, lohnt es sich, darüber nachzudenken, was die Ursachen für ihre Verletzung sind:

Die Lagen  $g_a$  und  $g_b$ , für die die  $L_2$ -Normen von  $f$  und  $g$  bzw.  $g$  und  $h$  minimiert werden, unterscheiden sich wesentlich. Um  $L_2(f, g_a)$  und  $L_2(g_b, h)$  mit  $L_2(f, h)$  zu vergleichen, müßte man  $g_a$  so verschieben, daß es mit  $g_b$  zur Deckung kommt und außerdem  $L_2(f, g_a)$  sich nicht ändert. Letzteres ist mit den in der Definition der FIF-Distanz vorgesehenen Verschiebungen nicht möglich – sie erlaubt nur Verschiebungen entlang der Graphen von  $f$  und  $g$ , die ganz unterschiedlich verlaufen. Die Beweisidee

$$L_2(f, g_a) + L_2(g_b, h) = L_2(f_s, g_b) + L_2(g_b, h) \geq L_2(f_s, h) \geq \min_{(r,t) \in [0,2\pi]^2} L_2(f_r, h_t),$$

wie sie im Fall der Polarkoordinatenmetrik für die Dreiecksungleichung genutzt wurde, funktioniert also hier nicht. In Abschnitt 2.2.4.3 wird gezeigt, unter welchen Einschränkungen dieser Gedanke dennoch zum Erfolg führt. Dazu muß man erreichen, daß die erlaubten Verschiebungen zur Minimierung der  $L_2$ -Norm nicht von Funktion zu Funktion verschieden sind.

**Berechnungskomplexität.** Es gibt aber noch ein algorithmisches Problem mit dieser Definition der Flächeninhaltsdistanz. Es soll ebenfalls unter Verwendung der  $\text{FIF}_{\text{lin}}$ -Distanz demonstriert werden:

Zur Bestimmung der Integraldifferenz gemäß Definition 2.10 muß ein Sweep-Verfahren über das Intervall  $[0, 2\pi]$  angewendet werden, da die Funktionen nur stückweise definiert sind. Die Haltepunkte des Sweeps, festgelegt durch die Übergangsstellen der Funktionen zwischen verschiedenen Geradenabschnitten, verändern ihre Lage beim Verschieben der Graphen. Dabei werden ständig Eventpunkte überstrichen, d.h. solche Punkte, bei denen eines der folgenden Ereignisse eintritt:

1. Zwei Übergangsstellen der beiden Funktionen  $f$  und  $g$  fallen zusammen („kritischer Punkt“). Bei diesem Ereignis verschwindet ein aktueller Streifen, der durch die beiden jetzt identischen Übergangsstellen begrenzt war. Ein neuer tut sich auf mit neuen aktuellen Geradenstücken.
2. Eine Übergangsstelle einer der beiden Funktionen durchstößt den Ursprung  $(0, 0)$ . Bei diesem Ereignis ändert sich das aktuelle Ursprungssegment (vgl. Kapitel 2.2.4.1). Dadurch ändert sich die Verschiebungsrichtung des Graphen der  $\text{FIF}_{\text{lin}}$ .

Zwischen zwei solchen Eventpunkten kann die optimale Lage der Graphen (mit minimaler Fläche zwischen den Kurven) mit einer ähnlichen Methode ausgerechnet werden, wie wir sie in Abschnitt 2.1.4.1 unter Punkt „Eigentliche Differenzbestimmung“ vorgestellt hatten. Bei jedem Überschreiten eines Eventpunktes jedoch verändern sich offensichtlich die aktuellen Verhältnisse drastisch: Bei Ereignis 1 ändert sich die Streifeneinteilung, die für die stückweise Integralberechnung zugrundeliegt. Bei Ereignis 2 ändert sich der Verschiebungsvektor der Form  $(1 \ c)^T$ , mithilfe dessen die lineare Funktion  $f(x) = a \cdot x + b$  eines Abschnitts durch  $f(x + t) - c \cdot t$  über alle Verschiebungen parametrisiert werden kann.

Aus diesem Grund ist die Berechnung der Polygonähnlichkeit gemäß Definition 2.10 mindestens so aufwendig, wie es Eventpunkte bei Verschiebungen der Kurven  $f_r$  und  $g_t$  für  $(s, t) \in [0, 2\pi]^2$  gibt. Diese Anzahl kann man wie folgt ermitteln:

Man stelle man sich zunächst  $f_r$  fixiert vor und betrachte  $g_t$ . Bei einer Variation von  $t$  im Intervall  $[0, 2\pi]$  überstreift jede Übergangsstelle von  $g$  jede der  $m$  Übergangsstellen des Graphen der fixierten Funktion  $f_r$ . Außerdem gelangt jede der  $n$  Übergangsstellen des Graphen von  $g$  einmal in den Ursprung – ein Ereignis vom Typ 2. Also gibt es hierfür bereits  $m \cdot n + n$  Eventpunkte.

Hat der Graph von  $g_t$  für  $t = 2\pi$  wieder seine Ausgangslage erreicht, so muß jetzt auch  $f$  variiert werden, und zwar zunächst so weit, bis erstmals ein Eventpunkt der Lage von  $f_r$  und der Ausgangslage von  $g$  (d.h.  $g_0$ ) erreicht ist. Für diese neue Lage von  $f$  ist nun wieder  $g$  vermöge  $g_t$  im Intervall  $t \in [0, 2\pi]$  zu variieren, um alle Lagekombinationen zu erfassen. Das ergibt wiederum  $m \cdot n + n$  Eventpunkte.

Dies ist nun für jede Lage von  $f$  zu tun, die mit der Ausgangslage von  $g$  einen Eventpunkt (eines der beiden Typen) induziert. Es gibt analog  $m \cdot n + m$  solche Lagen von  $f$ . Auf diese Weise erhält man  $(m \cdot n + n) \cdot (m \cdot n + m)$  Eventpunkte, die bei einer Verschiebung von  $f$  und  $g$  aus beliebiger Ausgangslage heraus erreicht werden. Die Anzahl ist schließlich noch mit dem Aufwand  $\mathcal{O}(m + n)$  für eine konkrete Integralberechnung zu multiplizieren. Damit hat das Auffinden der Lage mit minimaler Flächeninhaltsdifferenz für dieses konkrete Darstellungsmodell eine unakzeptable Komplexität.

**Zusammenfassung.** Die entstandenen Probleme müssen ursächlich auf die Existenz zu vieler Funktionsparameter der  $FIF_{\text{lin}}$  zurückgeführt werden. Ihre Variation führt zunächst zu einer komplizierten Veränderung des Graphen der Funktion. Dies umso mehr, da die Variationen der Parameter in unterschiedlicher Qualität (z.B. Verschieberichtung) und nicht etwa nur quantitativ verschieden auf den Graphen wirken. Die Variationen über diese Parameter müssen deshalb berücksichtigt werden, weil sie bei den Transformationen, die wir zugelassen haben, verändert werden.

Für die nichtapproximierte FIF gilt, daß selbst bei einer diskreten Minimierung nur über die Eventpunkte – wie im Abschnitt „Startpunktauswahl“ bei der PKF – die oben angeführte Komplexität angenommen wird, wenn *alle* Eventpunkte zur Auswahl stehen.

Obwohl die Ausgangsidee intuitiv sehr vernünftig erschien, hat sich also diese Methode nicht als praktikabel erwiesen. Da bei den meisten Vergleichs- und Optimierungsproblemen Extremwertaufgaben auftreten, sollte man daher im Fall funktionaler Darstellungen der Vergleichsobjekte darauf achten, daß die Funktionen wenige Parameter aufweisen, um die Komplexität begrenzt zu halten. Dies gilt in der Roboterlokalisierung erst recht, weil für eine Anfrage des Roboters an den Lokalisationsalgorithmus sehr viele Polygonvergleiche zu erwarten sind, die auch nicht in einem Preprocessing untergebracht werden können.

Man beachte schließlich, daß die Argumentation unabhängig von der gewählten Metrik auf der Menge stetiger Funktionen war, der Integral-Metrik. Bei der Komplexitätsanalyse wurde an keiner Stelle die eigentliche Integralberechnung einbezogen. Es liegt vielmehr an der Struktur der Polygondarstellung. Im folgenden wird gezeigt, wie durch eine (erhebliche) Vereinfachung dieser Struktur die Komplexität der Berechnung verringert und die Qualität der Ähnlichkeitsfunktion erhöht werden kann (Dreiecksungleichung), wenn auch nicht ohne Abstriche an anderen Stellen.

### 2.2.4.3 Definition der skalierten Flächeninhaltsmetrik

In diesem Abschnitt soll eine abgeschwächte Variante der Flächeninhaltsdistanz vorgestellt werden. Sie wird leicht und relativ effizient implementierbar sein und die Dreiecksungleichung erfüllen. Die Nachteile werden sein, daß sie nicht mehr ganz der Intuition entspricht, die zur Definition der FIF geführt hat, und daß sie nicht vollständig die Anforderungen der Roboterlokalisierung an eine Ähnlichkeitsfunktion erfüllt.

Wir hatten gesehen, daß eine Veränderung des Startpunktes  $P_1$  auf der Polygonperipherie grob einer diagonalen Verschiebung des Funktionsgraphen der Flächeninhaltsfunktion entspricht, allerdings mit ständig variierenden Verschiebungsrichtungen. Hier nehmen wir die erste Vereinfachung vor: In Bild 2.17 gilt für die Folge der Verschiebungen:  $\sum_{i=1}^5 \vec{v}_i = (2\pi A_P)^\top =: \vec{v}$ . Der Vektor  $\vec{v}$  kann daher als Gesamtvektor der Verschiebung im Intervall  $[0, 2\pi]$  bezeichnet werden. Er approximiert sowohl die sich ständig verändernde Verschiebungsrichtung der FIF als auch die im Bild dargestellte sukzessive Verschiebung entlang mehrerer Vektoren im Fall der  $FIF_{\text{lin}}$ .

Es sollen jetzt nur noch Verschiebungen entlang dieses Näherungsvektors zulässig sein. Damit verlassen wir etwas die Intuition: Durch dieses Verschieben von  $f$  zur neuen Funktion  $f_r$  gilt im allgemeinen  $f_r(0) \neq 0$ . Die Stelle, an der die Aufzeichnung der Funktion begonnen wird ( $x = 0$ ), bekommt nicht mehr den Flächeninhalt 0 zugewiesen; der Nullpunkt liegt kurz vor oder kurz hinter dem Meßausgangspunkt. Man beachte jedoch, daß dies nicht im Widerspruch zu den Bemerkungen aus Abschnitt 2.2.4.1 steht, wo wir das

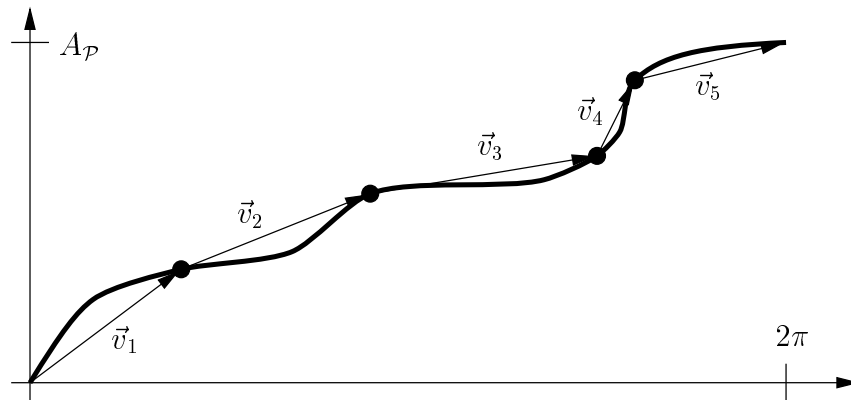


Abbildung 2.17: Variierende Verschiebungsrichtungen und Gesamtverschiebung

Zusammenlegen dieser beiden Punkte favorisiert hatten. Auch jetzt können die Punkte nicht unabhängig voneinander bewegt werden - anhand des Graphen kann man zu jedem potentiellen Meßausgangspunkt ablesen, wo der zugehörige Nullpunkt liegt: Die (quasiperiodisch fortgesetzte) Flächeninhaltsfunktion besitzt eine eindeutig bestimmte Nullstelle  $\varphi_0$ . Dieser Wert kann nun positiv oder negativ sein und gibt die Abweichung des Nullpunktes  $O$  vom Meßausgangspunkt  $P_1$  an. Ist  $\varphi_0 > 0$ , so liegt  $O$  kurz hinter  $P_1$  auf der Polygonperipherie (im Gegenuhreigersinn), im anderen Fall kurz davor. Dieses Verhalten kann man auch aus Abbildung 2.13 ablesen, wo die FIF allerdings noch nicht quasiperiodisch fortgesetzt war.

Damit hängt die Verschiebungsrichtung nur noch vom Polygon an sich ab und nicht mehr von der aktuellen Zwischenlage des Graphen.

Um die Dreiecksungleichung zu „erzwingen“, müssen wir aber erreichen, daß die Verschiebungsrichtung sogar für jedes Polygon dieselbe ist (zur Begründung vgl. Abschlußbemerkungen zum Punkt „Metrikeigenschaften“ auf Seite 50). Dazu führen wir eine Abschwächung der FIF ein:

**Definition 2.11 (skalierte Flächeninhaltsfunktion)** *Es gelten dieselben Voraussetzungen wie in Definition 2.6, und  $f$  sei die (exakte oder linear approximierte) Flächeninhaltsfunktion des angegebenen Polygons mit Flächeninhalt  $A_P$ . Dann heißt die Abbildung*

$$f_s : [0, 2\pi[ \longrightarrow [0, 1[ , \quad f_s(\varphi) := f(\varphi)/A_P$$

(exakte oder linear approximierte) **skalierte Flächeninhaltsfunktion** (Abk. **FIF<sub>s</sub>**).

Im folgenden wird, wenn nichts anderes gesagt ist, die skalierte Variante der Flächeninhaltsfunktion gemeint sein, auch wenn wir das Attribut „skaliert“ auslassen.

Es ist sofort einsichtig, daß dadurch die Skalierungsabhängigkeit der Darstellung verlorengeht, die für die Roboterlokalisierung mittels Laser-Radar (=Entfernungsmesser) natürlicherweise gefordert wird. Am Ende des Kapitels 2.2 wird allerdings eine prinzipielle Möglichkeit gezeigt, wie die Metrik wieder sensibel gegenüber Skalierungen gemacht werden kann.

Das positive Ergebnis ist, daß der Funktionswert an der Stelle  $\varphi = 2\pi$  nun 1 beträgt und damit vom Polygon unabhängig ist. Der Graph der Flächeninhaltsfunktion darf nur entlang des Vektors  $(2\pi \ 1)^T$  verschoben werden, was näherungsweise der Veränderung des

Startpunktes  $P_1$  auf der Peripherie entspricht. Für eine Abschnittsfunktion  $f$  (zwischen zwei Haltepunkten des Sweeps) wird dies durch den Ausdruck  $f(x + 2\pi t) - t$  modelliert.

Bei dieser Vorgehensweise läßt sich erfreulicherweise auch das zweite Problem lösen: das der Komplexität. Angenommen, für die Lage der Punkte  $P_1$  wie in Bild 2.18(a) sei die  $L_2$ -Norm der Differenz der FIF-Kurven minimal. Im Bild (b) daneben ist der Punkt  $P_1$

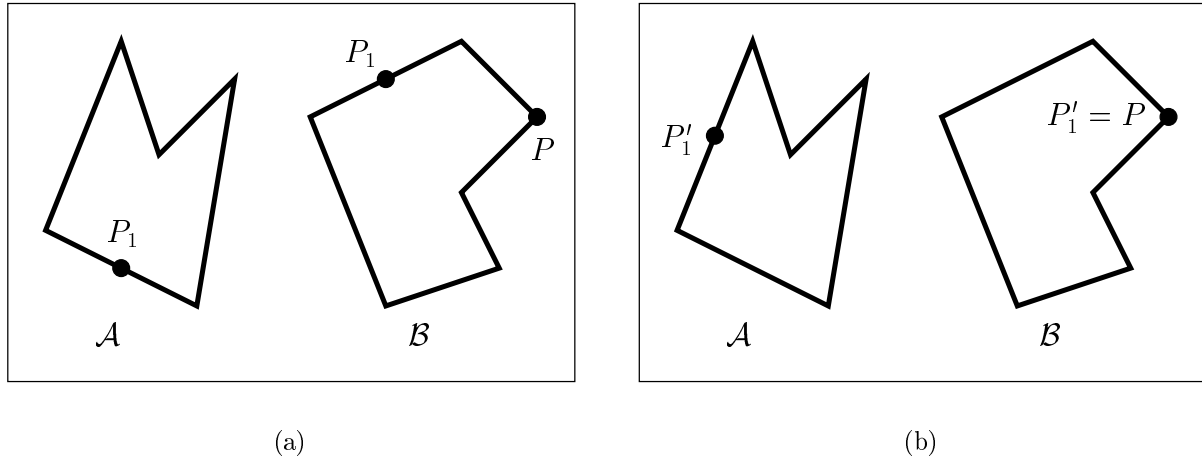


Abbildung 2.18: Gleichmäßige Verschiebung des Startpunktes  $P_1$  in zwei Polygonen

des Polygons  $\mathcal{B}$  um etwa  $90^\circ$  im Uhrzeigersinn in einen vorher bestimmten Eckpunkt  $P$  verschoben worden. Im Polygon  $\mathcal{A}$  ist auf den Punkt  $P_1$  genau dieselbe Verschiebung angewendet worden. Diese Veränderungen des Funktionsparameters  $P_1$  repräsentieren eine Verschiebung der Flächeninhaltsfunktionen  $f_s$  und  $g_s$  um den gleichen Betrag  $\pi/2$  in  $x$ -Richtung. Da schließlich der Verschiebungsvektor der FIFs in beiden Fällen  $(2\pi \ 1)^\top$  ist, ist auch der Verschiebungsbetrag in  $y$ -Richtung derselbe (nämlich  $1/4$ ), wodurch sich einfach eine parallele Veränderung beider Funktionsgraphen ergibt. Der Flächeninhalt dazwischen (d.h. die  $L_2$ -Norm der Differenz) ist unverändert geblieben, also in Abbildung 2.18(b) ebenfalls minimal.

Die Folge ist, daß wir für jede Lage der Graphen, für die die Flächendifferenz zu bestimmen ist, ohne Beschränkung der Allgemeinheit davon ausgehen können, daß ein vorher fixierter Punkt  $P$  des Polygons  $\mathcal{B}$  (wie in Abbildung 2.18 (b)) dem Punkt  $(0,0)$  im Graphen entspricht. Wir brauchen daher nur über alle Verschiebungen der FIFs des Polygons  $\mathcal{A}$  zu minimieren.

Die Anzahl der Eventpunktlagen ist nun deutlich reduziert. Eventpunkte vom Typ 2 spielen keine Rolle mehr: Die Verschieberichtung ändert sich nicht während der Verschiebung. Also verbleiben genau die  $m \cdot n$  Eventpunktlagen, bei denen zwei Übergangsstellen von  $f$  und  $g$  zusammenfallen.

Der Verschiebeparameter  $t$  muß aufgrund der Quasiperiodizität natürlich nur im Intervall  $[0, 1]$  (für den Ausdruck  $f(x + 2\pi t) - t$ ) betrachtet werden.

**Definition 2.12 (FIF-Distanz)** Seien  $\mathcal{A}$  und  $\mathcal{B}$  zwei sternförmige Polygone, für die nach dem Verfahren vom Beginn des Abschnitts 2.2.4 jeweils ein spezieller Kernpunkt festgelegt wurde. Die diesbezüglichen skalierten Flächeninhaltsfunktionen seien  $f$  und  $g$ .

Die Abbildung

$$s(\mathcal{A}, \mathcal{B}) := \min_{t \in [0,1]} \sqrt{\int_0^{2\pi} \left( f(x + 2\pi t) - t - g(x) \right)^2 dx}$$

heißt **skalierte Flächeninhaltsdistanz** für sternförmige Polygone.

(Zur Vereinfachung werden wir aber nur von der Flächeninhaltsdistanz sprechen und meinen, wie auch bei der Flächeninhaltsfunktion, stets die skalierte Variante.)

**Satz 2.13 (FIF-Metrik)** *Die skalierte Flächeninhaltsdistanz ist eine metrische Ähnlichkeitsfunktion und wird daher im folgenden auch (**skalierte**) **Flächeninhaltsmetrik** genannt.*

**Beweis:**

1. Die Eigenschaft  $\mathcal{A} = \mathcal{B} \Leftrightarrow s(\mathcal{A}, \mathcal{B}) = 0$  benutzt sowohl für die FIF als auch für die  $\text{FIF}_{\text{lin}}$  ein interessantes Resultat aus Kapitel 2.3 und wird deshalb erst auf Seite 63 nachgewiesen. Dort geht es um die Rekonstruktion eines Polygons aus der FIF-Darstellung, die einem konstruktiven Beweis der Rückrichtung „ $\Leftarrow$ “ obenstehender Äquivalenz entspricht. (Die andere Richtung stellt keine Schwierigkeit dar.) Man bedenke, daß „ $\mathcal{A} = \mathcal{B}$ “ hier Identität der Punktmengen bis auf Translation, Rotation und Skalierung bedeutet.

Die Symmetrie und die Dreiecksungleichung zeigt man ähnlich wie in Satz 2.4; allerdings sind die hier vorkommenden Funktionen nicht periodisch, sondern quasiperiodisch. Zu beachten ist, daß die beiden folgenden Beweisteile für die FIF wie auch für die  $\text{FIF}_{\text{lin}}$  gleichermaßen gelten, denn sie benutzen nur die Quasiperiodizität der Funktionen, die natürlich auch im Fall der  $\text{FIF}_{\text{lin}}$  gegeben ist:

2. Symmetrie: Seien  $t_{\mathcal{A}\mathcal{B}}$  der minimierende Wert für  $s(\mathcal{A}, \mathcal{B})$  und  $t_{\mathcal{B}\mathcal{A}} := 1 - t_{\mathcal{A}\mathcal{B}} \in [0, 1]$ . Dann gilt:

$$\begin{aligned} s^2(\mathcal{A}, \mathcal{B}) &= \int_0^{2\pi} \left( f(x + 2\pi t_{\mathcal{A}\mathcal{B}}) - t_{\mathcal{A}\mathcal{B}} - g(x) \right)^2 dx \\ &\stackrel{(*)}{=} \int_0^{2\pi} \left( f(x + 2\pi t_{\mathcal{A}\mathcal{B}} + 2\pi t_{\mathcal{B}\mathcal{A}}) - t_{\mathcal{A}\mathcal{B}} - g(x + 2\pi t_{\mathcal{B}\mathcal{A}}) \right)^2 dx \\ &= \int_0^{2\pi} \left( f(x + 2\pi) - t_{\mathcal{A}\mathcal{B}} - g(x + 2\pi t_{\mathcal{B}\mathcal{A}}) \right)^2 dx \\ &\stackrel{(\bullet)}{=} \int_0^{2\pi} \left( f(x) + t_{\mathcal{B}\mathcal{A}} - g(x + 2\pi t_{\mathcal{B}\mathcal{A}}) \right)^2 dx \\ &\geq \min_{t \in [0,1]} \int_0^{2\pi} \left( g(x + 2\pi t) - t - f(x) \right)^2 dx \\ &= s^2(\mathcal{B}, \mathcal{A}). \end{aligned}$$

Für die Gleichung (\*) wurden lediglich die Graphen von  $f$  und  $g$  gemeinsam um  $2\pi t_{\mathcal{B}\mathcal{A}}$  (nach links) verschoben, ohne daß sich dadurch ihr  $L_2$ -Abstand ändert. In

der Gleichung (•) wurden einerseits die Quasiperiodizität  $f(x + 2\pi) = f(x) + 1$  und andererseits  $1 - t_{\mathcal{A}\mathcal{B}} = t_{\mathcal{B}\mathcal{A}}$  ausgenutzt. Diese Beziehung zwischen den Verschiebeparametern besagt natürlich, daß bei einer Vertauschung der Rollen von  $f$  und  $g$  die minimierende Verschiebung in entgegengesetzter Richtung erfolgt.

Mit der analog zu zeigenden Ungleichung  $s(\mathcal{B}, \mathcal{A}) \geq s(\mathcal{A}, \mathcal{B})$  erhält man Gleichheit.

3. Dreiecksungleichung: Seien dazu  $f$ ,  $g$  und  $h$  die Flächeninhaltsfunktionen dreier Polygone  $\mathcal{A}$ ,  $\mathcal{B}$  und  $\mathcal{C}$ , und seien  $t_{\mathcal{A}\mathcal{B}}$  und  $t_{\mathcal{B}\mathcal{C}}$  die minimierenden Werte für  $s(\mathcal{A}, \mathcal{B})$  und  $s(\mathcal{B}, \mathcal{C})$ .

$$\begin{aligned}
s(\mathcal{A}, \mathcal{B}) + s(\mathcal{B}, \mathcal{C}) &= \sqrt{\int_0^{2\pi} \left( f(x + 2\pi t_{\mathcal{A}\mathcal{B}}) - t_{\mathcal{A}\mathcal{B}} - g(x) \right)^2 dx} \\
&\quad + \sqrt{\int_0^{2\pi} \left( g(x + 2\pi t_{\mathcal{B}\mathcal{C}}) - t_{\mathcal{B}\mathcal{C}} - h(x) \right)^2 dx} \\
&= \sqrt{\int_0^{2\pi} \left( f(x + 2\pi t_{\mathcal{A}\mathcal{B}} + 2\pi t_{\mathcal{B}\mathcal{C}}) - t_{\mathcal{A}\mathcal{B}} - g(x + 2\pi t_{\mathcal{B}\mathcal{C}}) \right)^2 dx} \\
&\quad + \sqrt{\int_0^{2\pi} \left( g(x + 2\pi t_{\mathcal{B}\mathcal{C}}) - t_{\mathcal{B}\mathcal{C}} - h(x) \right)^2 dx} \\
&\stackrel{(**)}{\geq} \sqrt{\int_0^{2\pi} \left( f(x + 2\pi(t_{\mathcal{A}\mathcal{B}} + t_{\mathcal{B}\mathcal{C}})) - (t_{\mathcal{A}\mathcal{B}} + t_{\mathcal{B}\mathcal{C}}) - h(x) \right)^2 dx} \\
&\geq \min_{t \in [0, 1]} \sqrt{\int_0^{2\pi} \left( f(x + 2\pi t) - t - h(x) \right)^2 dx} \\
&= s(\mathcal{A}, \mathcal{C}).
\end{aligned}$$

Im Schritt (\*\*) gilt die Minkowski-Ungleichung. □

#### 2.2.4.4 Berechnung der skalierten Flächeninhaltsmetrik

Zunächst ist ein spezieller Kernpunkt nach der Methode von Abschnitt 2.1.3.1 zu bestimmen; der Aufwand hierbei beträgt  $\mathcal{O}(k \cdot \log k)$  für  $k = \max(m, n)$ , wenn die Eingabepolygone  $m$  bzw.  $n$  Ecken haben. Er kann für jedes Polygon separat bestritten werden und ist damit Preprocessing-fähig (geeignet bei großen Szenen mit mehreren zu erwartenden Anfragen).

Für die Bestimmung der  $L_2$ -Norm laut Definition 2.12 hat man – wie bei der PKF – zwei Optionen: Da eine exakte Bestimmung des angegebenen Minimums für die nichtapproximierte FIF<sub>S</sub>-Distanz nicht evident ist, kann man sich auf die  $m \cdot n$  Eventpunkte beschränken, die durch die Fixierung des Graphen von  $g$  (siehe Seite 54) übriggeblieben sind, und nur unter diesen Lagen von  $f$  minimieren. Es gelten dieselben Kriterien zur Startpunktauswahl wie in Abschnitt 2.1.4.2; man kann also durchaus noch weiter einschränken, wenn zum Beispiel die unterschiedliche Qualität der beiden Eingabepolygone bekannt ist, wie im Fall des Zellskeletts aus dem Preprocessing und des Scan-Skeletts.

In jedem Fall sind maximal  $m \cdot n$  Integralberechnungen einer jeweiligen Komplexität von  $\mathcal{O}(m+n)$  durchzuführen. Das Integral innerhalb eines der  $\mathcal{O}(m+n)$  Streifen erhält man für  $c_{1,2} = 1/2 \cdot |pP_i|^2 \cdot \sin \beta_{1,2}$  aus Formel (2.4) wie folgt: Mit

$$\begin{aligned} S(\beta) &= \int_{\varphi_l}^{\varphi_r} \frac{\sin^2 \varphi}{\sin^2(\varphi + \beta)} d\varphi \\ &= \left[ \frac{\cos(\varphi - \beta) - \cos(\varphi + \beta) + \varphi \cdot \sin(\varphi - \beta) + \varphi \cdot \sin(\varphi + 3\beta)}{2 \cdot \sin(\varphi + \beta)} \right. \\ &\quad \left. - \log \sin(\varphi + \beta) \cdot \sin 2\beta \right]_{\varphi_l}^{\varphi_r} \end{aligned}$$

und

$$\begin{aligned} T(\beta_1, \beta_2) &= \int_{\varphi_l}^{\varphi_r} \frac{\sin^2 \varphi}{\sin(\varphi + \beta_1) \cdot \sin(\varphi + \beta_2)} d\varphi \\ &= \left[ \varphi \cdot \cos(\beta_1 + \beta_2) + \frac{\log \sin(\varphi + \beta_2) \cdot \sin^2 \beta_2 - \log \sin(\varphi + \beta_1) \cdot \sin^2 \beta_1}{\sin(\beta_1 - \beta_2)} \right]_{\varphi_l}^{\varphi_r} \end{aligned}$$

ergibt sich

$$\int_{\varphi_l}^{\varphi_r} \left( \frac{c_1 \cdot \sin \varphi}{\sin(\varphi + \beta_1)} - \frac{c_2 \cdot \sin \varphi}{\sin(\varphi + \beta_2)} \right)^2 d\varphi = c_1^2 \cdot S(\beta_1) + c_2^2 \cdot S(\beta_2) - 2c_1 c_2 \cdot T(\beta_1, \beta_2)$$

für  $\beta_1 \neq \beta_2$  und

$$\int_{\varphi_l}^{\varphi_r} \frac{(c_1 - c_2)^2 \cdot \sin^2 \varphi}{\sin^2(\varphi + \beta)} d\varphi = (c_1 - c_2)^2 \cdot S(\beta)$$

für  $\beta_1 = \beta_2 =: \beta$ .

Bei diesen Formeln beachte man jedoch, daß die FIF gemäß Gleichung (2.5) akkumulierend ist, d.h. der Funktionswert in einem Sektor ergibt sich aus einem lokalen Wert zuzüglich des bis dahin überstrichenen Flächeninhalts. Dieser Sachverhalt wurde in den angegebenen Integralen zur Vereinfachung der Darstellung nicht berücksichtigt.

Die andere Option besteht in der Verwendung der linear approximierten (skalierten) Flächeninhaltsfunktion  $FIF_{\text{lin}}$ . Auch wenn die Minimierungsvorschrift – Verschiebung entlang  $(2\pi \cdot 1)^T$  – hier komplizierter ist als bei der PKF, kann das Minimum aus Definition 2.12 mit der  $FIF_{\text{lin}}$  exakt bestimmt werden. Das Verfahren ist mit dem von Algorithmus 1 bis auf einige technische Veränderungen identisch, daher seien nur die wichtigsten davon genannt:

Es gibt also  $m \cdot n$  Eventpunktlagen von  $f$ , während die Lage von  $g$  fixiert ist. Zwischen diesen Lagen sind die Funktionsgleichungen fest, und die  $L_2$ -Norm unterliegt einer Veränderung, die durch ein Polynom zweiten Grades beschrieben und deren Minimalwert daher exakt bestimmt werden kann.

Ohne Beschränkung der Allgemeinheit verschieben wir  $f$  nur nach links unten. Die aktuellen Streifengrenzen, d.h. die Ränder des aktuellen Integrationsintervalls, lauten allgemein  $l - 2\pi t$  und  $r - 2\pi t$ , wenn  $t$  der Verschiebeparameter ist.

Befindet sich  $f$  am rechten Rand des aktuellen Streifens, d.h. vor der Verschiebung, dann seien die beiden FIF<sub>S</sub>-Gleichungen durch  $f(x) = ax + b$ ,  $g(x) = cx + d$  gegeben. Die  $L_2$ -Norm zwischen den Kurven beträgt

$$F(t) := \int_{l-2\pi t}^{r-2\pi t} \left( a(x + 2\pi t) + b - t - (cx + d) \right)^2 dx. \quad (2.6)$$

In expliziter Darstellung erhält man  $F(t) = x \cdot t^2 + y \cdot t + z$  mit

$$\begin{aligned} x &= (2\pi c - 1)^2 \cdot (r - l) \\ y &= (2\pi c - 1) \cdot ((a - c)(r^2 - l^2) + 2 \cdot (b - d)(r - l)) \\ z &= 1/3 \cdot (a - c)^2 \cdot (r^3 - l^3) + (a - c)(b - d)(r^2 - l^2) + (b - d)^2 \cdot (r - l). \end{aligned}$$

Die  $L_2$ -Norm zwischen beiden Kurven ist genau dann von  $t$  unabhängig, wenn die Verschieberichtung (von  $f$ ) mit dem Anstieg von  $g$  übereinstimmt, d.h. wenn  $1/2\pi = c$  ist. Das bestätigt die Formel, denn für diesen Wert von  $c$  ist  $F(t) = z = \text{const}$ . In allen anderen Fällen ist  $x > 0$ , d.h.  $F$  ist wieder eine nach oben geöffnete Parabel. Die Darstellung von Bild 2.8(b) (Seite 33) gilt entsprechend.

Stellt man die quadratische Form (2.6) für jeden Streifen auf und bildet die Summe der Koeffizienten  $x$ ,  $y$  und  $z$  aller Streifen im Intervall  $[0, 2\pi]$ , so erhält man eine Darstellung  $F_{\text{ges}}(t) = X \cdot t^2 + Y \cdot t + Z$ , die die Änderung der  $L_2$ -Norm unter Verschiebungen im gesamten Definitionsbereich angibt. Das zulässige Intervall  $[0, t^-]$  für den Verschiebeparameter errechnet man analog zu Abschnitt 2.1.4.1, indem  $t^-$  als die Breite des schmalsten Streifens genommen wird, der links von  $g$  und rechts von  $f$  begrenzt wird. Um diesen Wert  $t^-$  kann  $f$  aus der aktuellen Initiallage heraus verschoben werden, ohne daß zwei Übergangsstellen von  $f$  und  $g$  kollidieren. Damit kann das globale Minimum von  $F_{\text{ges}}(t)$  in  $[0, t^-]$  leicht bestimmt werden.

Die Gesamtprozedur betrachtet nun jede Eventpunktlage von  $f$ , in der eine gemeinsame Übergangsstelle mit  $g$  existiert. Gemäß der Lagen *aller* Übergangsstellen wird das Intervall  $[0, 2\pi]$  in Streifen eingeteilt und für jeden Streifen nach obiger Vorschrift die Funktion  $F(t)$  gebildet. Die Koeffizientensumme ergibt nach dem Durchlauf über alle Streifen die Formel  $F_{\text{ges}}(t)$  und den Wert  $t^-$ , so daß das globale Minimum von  $F_{\text{ges}}$  bestimmt werden kann. Dieser Wert ist über alle Eventpunktlagen von  $f$  zu minimieren. Das Resultat ist die  $L_2$ -Norm der beiden stückweise linearen Funktionen  $f$  und  $g$  unter allen Verschiebung von  $f$  und  $g$  entlang  $(2\pi - 1)^T$ ; der Gesamtaufwand beträgt – wie bei der PKF-Metrik –  $\mathcal{O}(mn \cdot (m + n))$ .

### 2.2.5 Abschließendes zur Flächeninhaltsdistanz

In diesem Kapitel wurde eine Ähnlichkeitsfunktion vorgestellt, die den vom Laserstrahl überstrichenen Flächeninhalt (bei einem 360°-Rundumscan) als Funktion des Winkels aufträgt, den der Strahl mit einer festen Referenzrichtung bildet. Diese Funktion hängt von zwei Parametern ab, die nicht unabhängig von Rotationen des Eingabepolygons festgelegt werden können: dem Peripheriepunkt, an dem die Messung des Flächeninhalts beginnt, und demjenigen, an dem er auf Null gesetzt wird. Selbst wenn man vereinbart, daß diese Punkte stets identisch sein sollen, wirkt ihre Veränderung unterschiedlich auf den Graphen der Funktion: in Form einer Horizontal- und einer Vertikalverschiebung.

Vereint man diese Punkte zu einem gemeinsamen Startpunkt  $P_1$  auf der Peripherie, so gilt stets  $f(0) = 0$ . Eine Variation von  $P_1$  bedeutet dann eine Verschiebung des Graphen durch den Ursprung hindurch, indem sich die Verschiebungsrichtung an jedem Punkt des Graphen ändert. Diese Parametrisierung ist kompliziert genug, daß eine daraus abgeleitete Ähnlichkeitsfunktion nachweislich nicht die Dreiecksungleichung erfüllt und außerdem kaum effizient berechenbar ist.

**Beseitigung des globalen Nullpunkts.** In Abschnitt 2.2.4.2 (siehe dortige Zusammenfassung) wurde argumentiert, daß die Ursachen dafür darin liegen, daß die Flächeninhaltsfunktion keine *absolute* Funktion ist. Eine einfache Möglichkeit, sie zu einer absoluten Funktion zu machen, ist anscheinend folgende:

Statt den insgesamt überstrichenen Flächeninhalt zu betrachten, legt man als Funktionswert diejenige Fläche fest, die im *aktuellen Sektordreieck* überstrichen wurde (Abbildung 2.19). Damit hat man erreicht, daß für jeden Peripheriepunkt  $P$  der Funktionswert

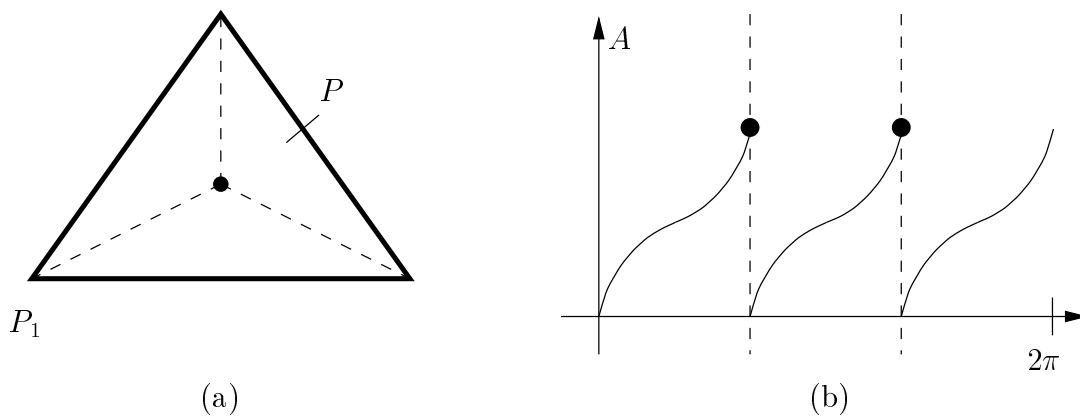


Abbildung 2.19: Ein Dreieck (a) und seine sektorenbezogene Flächeninhaltsfunktion (b)

eindeutig feststeht – unabhängig von einem gewählten Nullpunkt: Man bestimmt den Sektor, zu dem  $P$  gehört, und berechnet den darin überstrichenen Flächeninhalt. Damit ist die Funktion in der Tat nur noch von dem Punkt abhängig, an dem die Messung beginnt. Eine Veränderung dieses Punktes bedeutet eine einfache waagerechte Verschiebung des Graphen, und wir haben eine ähnliche Situation wie bei der Polarkoordinatenmetrik.

Dieser Ansatz schlägt jedoch in der Praxis völlig fehl. In Abbildung 2.20 ist noch einmal das Dreieck aus Bild 2.19 dargestellt, allerdings mit einigen Meßungenauigkeiten. Diese sind noch dazu ziemlich „harmlos“: die gemessenen Entfernungen weichen nur wenig von denen im Bild davor ab; allerdings hat ein Extraktionsprogramm diese geringen Abweichungen bereits zum Anlaß genommen, einige zusätzliche Eckpunkte einzufügen. Dadurch ist die Sektoreinteilung des Dreiecks feiner geworden, und der abgebildete Funktionsgraph hat doppelt so viele Nullstellen.

Durch solche Veränderungen kann die Integraldifferenz der Funktionen aus den Bildern 2.19(b) und 2.20(b) beliebig groß werden. Es müssen nur ausreichend viele falsche Eckpunkte zum Dreieck hinzukommen, die sogar kollinear mit den echten Ecken sein dürfen. Zusätzlich sind diese Abbildungen nun nicht mehr stetig. – All dies gilt für die linear approximierten FIF in analoger Weise.

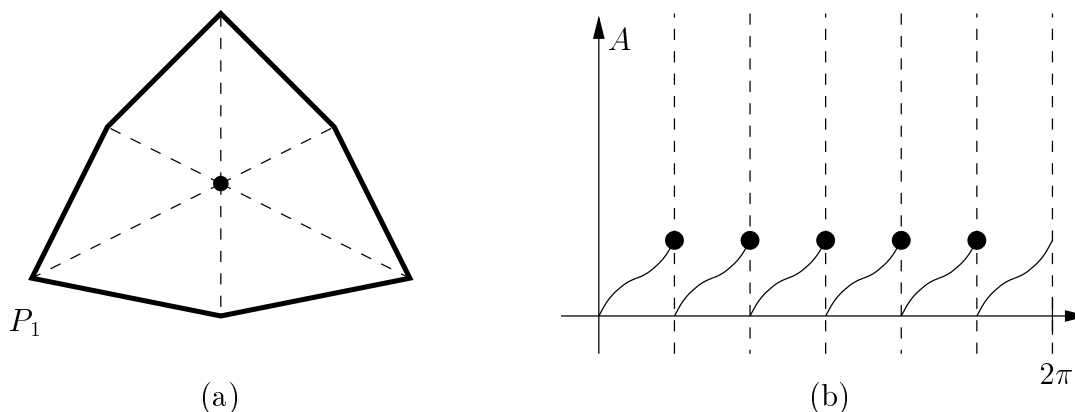


Abbildung 2.20: Ein verrauschtes Dreieck (a) und seine sektorenbezogene Flächeninhaltsfunktion (b)

In Kapitel 2.2.4.3 wurde die Definition der Flächeninhaltsdistanz drastisch vereinfacht. Der Flächeninhalt wurde für jedes Polygon auf 1 normiert, als Verschiebungsvektor nur noch  $(2\pi \ 1)^\top$  zugelassen. Die abgeleitete *skalierte* FIF<sub>S</sub>-Distanz entsprach damit nur näherungsweise der Intuition. Die  $(2\pi \ 1)^\top$ -Verschiebung weicht von der lageabhängigen Verschiebung durch den Ursprung dann stärker ab, wenn die Flächeninhalte der einzelnen Sektoren sehr unterschiedlich sind. Der Verlauf des Graphen (ob linear approximiert oder nicht) ist in diesem Fall nur eine schlechte Näherung der Strecke  $(0, 0)(2\pi, 1)$ .

Aufgrund der durchgeführten Vereinfachungen konnte für die neue Distanz die Dreiecksungleichung gezeigt werden, so daß alle Metrikeigenschaften erfüllt sind. Außerdem ließ sich mit dem Verfahren zur Minimierung der  $L_2$ -Norm (bzw. [zumindest theoretisch] jeder beliebigen  $L_p$ -Norm) zwischen (stückweise) linearen Funktionen der exakte Wert der FIF<sub>lin</sub>-Metrik bestimmen.

Nachteile haben sich in der Anwendbarkeit für die Roboterlokalisierung gezeigt. Zwar ist die Metrik **translations- und rotationsunabhängig**<sup>3</sup>, sie ist aber auch **unabhängig gegenüber Skalierungen** der Eingabepolygone. Die Skalierungsabhängigkeit der Metrik wurde gewissermaßen der Dreiecksungleichung geopfert, deren Nachweis ohne die Normierung der Polygonflächeninhalte auf  $f(2\pi) = 1$  nicht möglich gewesen wäre.

Diesem Umstand kann eventuell wie folgt abgeholfen werden.

**Lemma 2.14** Sei  $s$  eine Metrik auf einer Menge  $\mathcal{O}$  geometrischer Objekte, die durch eine Jordankurve darstellbar sind. Dann ist für jedes Objekt  $o \in \mathcal{O}$  ein Flächeninhalt  $\text{area}: \mathcal{O} \rightarrow \mathbb{R}_{>0}$  definiert. Sei weiter  $\omega$  eine positive reelle Zahl. Die Abbildung

$$s': \mathcal{O} \times \mathcal{O} \rightarrow \mathbb{R}_{\geq 0}, \quad s'(x, y) := s(x, y) + \omega \cdot |\text{area}(x) - \text{area}(y)| \quad (2.7)$$

ist ebenfalls eine Metrik, und sie ist skalierungsabhängig.

**Beweis:** Die Metrikeigenschaften rechnet man unmittelbar nach. Für die Dreiecksunglei-

<sup>3</sup>Falls ein Kompaß zur Verfügung steht, kann die FIF-Metrik auch in einer rotationssensitiven Variante implementiert werden; siehe dazu Abschnitt 2.1.5 auf Seite 39 bei der PKF.

chung gilt zum Beispiel

$$\begin{aligned} s'(x, y) + s'(y, z) &= s(x, y) + s(y, z) + \omega \cdot (|\text{area}(x) - \text{area}(y)| + |\text{area}(y) - \text{area}(z)|) \\ &\geq s(x, z) + \omega \cdot |\text{area}(x) - \text{area}(z)| \\ &= s'(x, z). \end{aligned}$$

□

Die Skalierungsabhängigkeit, die durch den rechten Summanden in Gleichung 2.7 erreicht wird, ist durch den Parameter  $\omega$  beeinflusst. Er gibt an, ob die alte Metrik  $s$  oder die Flächendifferenz eine stärkere Wirkung auf  $s'$  hat.

Eine alternative Definition von  $s'$  wäre

$$s''(x, y) := s(x, y) + \omega \cdot \frac{|\text{area}(x) - \text{area}(y)|}{\text{area}(x) + \text{area}(y)}.$$

Auch  $s''$  ist (als Summe zweier Metriken) eine Metrik<sup>4</sup>. Sie ist toleranter gegenüber Flächeninhaltsdifferenzen bei großen Objekten; bei kleinen fallen sie mehr ins Gewicht.

Die Schwierigkeit besteht natürlich in der Wahl des Parameters  $\omega$ . Dazu liegen noch keine Erkenntnisse vor. Ob die Idee von Lemma 2.14 überhaupt sinnvoll ist, muß ebenfalls noch untersucht werden.

Die skalierte Flächeninhaltsmetrik aus Definition 2.12 wurde in beiden Versionen (FIF und FIF<sub>lin</sub>) implementiert und getestet. Einige Beispiele dazu sind im Anhang aufgeführt. Zu einer theoretischen Betrachtung der Qualität der Metrik und zur Toleranz gegenüber verrauschten Daten siehe den zusammenfassenden Abschnitt 2.3.3.

## 2.3 Zusammenfassung

### 2.3.1 Vergleich PKF- und FIF-Distanz

In den einzelnen Kapiteln über die genannten Ähnlichkeitsfunktionen wurde bereits ausführlich eine Bewertung formuliert. Daher soll ein Vergleich nur noch einmal knapp die wesentlichen Charaktere der Distanzen herausstellen (Tabelle 2.3). Man beachte, daß ein „ja“ in der Zeile „Invariant unter Skalierung“ ein unerwünschtes Resultat ist. Das Kriterium „Minimierung ist intuitionsgemäß“ soll angeben, ob die Variation des in der Distanz verwendeten Minimierungsparameters der Veränderung des korrespondierenden Polygonparameters entspricht, also der Polygonorientierung bei der PKF bzw. der Wahl des Startpunktes  $P_1$  bei der FIF. Es steht in engem Zusammenhang mit dem Kriterium „exakte Berechnung in der Praxis“.

Die vorgestellten Metriken bzw. Distanzen haben sich als recht robust gegenüber verrauschten Daten erwiesen, unabhängig davon, ob das Rauschen gleichmäßig oder ungleichmäßig verteilt ist. Ein allgemeiner Nachteil der linear approximierten Kurven muß

---

<sup>4</sup>Die Dreiecksungleichung – die einzige nicht offensichtliche Metrikeigenschaft von  $s''$  – beweist man durch eine Fallunterscheidung bezüglich der Lagebeziehung von  $x$ ,  $y$  und  $z$  und Ausnutzen von Termsymmetrien.

Kriterium	PKF-Distanz	FIF-Distanz	FIF <sub>S</sub> -Distanz
Metrikeigenschaften	ja	nein	ja
exakte Berechnung in der Praxis*	nein (PKF) ja (PKF <sub>lin</sub> )	nein	nein (FIF) ja (FIF <sub>lin</sub> )
Verwendete Funktionenmetrik	$L_2$	$L_2$	$L_2$
Anzahl der Minimierungsparameter in der Metrikdefinition	1	2	1
Minimierung ist intuitionsgemäß	nein (PKF) näherungsweise (PKF <sub>lin</sub> )	ja	nein (FIF) näherungsweise (FIF <sub>lin</sub> )
Invariant unter Translation Rotation Skalierung	ja ja nein	ja ja nein	ja ja ja
Berechnungskomplexität	$mn \cdot (m + n)$	$(mn + n) \cdot (mn + m) \cdot (m + n)$	$mn \cdot (m + n)$

\*Ein „nein“ in dieser Zeile bedeutet auch, daß die Metrikeigenschaften in der Praxis nicht mehr gelten.

Tabelle 2.3: Vergleich verschiedener Ähnlichkeitsfunktionen in der Übersicht

dennoch erwähnt werden: Da sie immer aus Stützstellen der exakten Kurve entstehen, die Polygonecken entsprechen, können redundante (evtl. auch durch Meßungenauigkeiten entstandene) Peripheriepunkte, die kollinear mit ihren Nachbarn liegen, das Ergebnis negativ beeinflussen: Vergleicht man ein Dreieck mit einem Viereck, das aus dem Dreieck durch Einfügen des Mittelpunkts einer Seite als Scheinecke entsteht, so verändern sich die PKF<sub>lin</sub> und die FIF<sub>lin</sub> leicht, da jetzt eine weitere Stützstelle hinzukommt. Will man diese beiden Polygone als identisch betrachten (was freilich nur in quantitativer Hinsicht, nicht qualitativ, gelten kann), so muß man auf die Verwendung der linear approximierten Darstellungen verzichten.

Der wesentliche konzeptionelle Unterschied zwischen der PKF- und der FIF<sub>S</sub>-Distanz ist, daß die erstgenannte *absolut* ist, während die zweite schon im Ansatz von mehreren Parametern abhängt.

### 2.3.2 Rekonstruktion der Polygone aus PKF- und FIF-Darstellungen

An dieser Stelle soll die Eindeutigkeit der PKF- und der FIF-Repräsentationen für Polygone untersucht werden: Gibt es zwei (bis auf gewisse Ähnlichkeitstransformationen) verschiedene Polygone mit derselben PKF- bzw. FIF-Darstellung? Falls ja, so lassen sich die Polygone aus dem Graphen der Funktion unter Umständen nicht rekonstruieren, und die Bezeichnung „Darstellung“ ist im strengen Sinne gar nicht gerechtfertigt.

### 2.3.2.1 Zur Polarkoordinatenfunktion

Bei der Polarkoordinatenfunktion ist die Antwort auf die Frage der Rekonstruierbarkeit sehr einfach: Man gibt sich den Punkt  $p$  in der Ebene vor und trägt in der Richtung  $\varphi$  gegenüber der Horizontalen einen Peripheriepunkt im Abstand  $\text{pkf}(\varphi)$  von  $p$  ein. Es genügt sogar, die aus der Kurve ersichtlichen Übergangsstellen der Funktion zu betrachten, da sie genau die Ecken des Polygons ergeben. Daher ist die PKF-Darstellung ebenso eindeutig wie die vereinfachte  $\text{PKF}_{\text{lin}}$ -Darstellung.

### 2.3.2.2 Zur Flächeninhaltsfunktion

Dieser Abschnitt behandelt die nichtapproximierte FIF.

Betrachten wir noch einmal die Gleichungen (2.1) für den Abstand  $r(\varphi)$  und (2.4) für den überstrichenen Flächeninhalt  $A(\varphi)$ , die der analytischen Beschreibung der PKF und der FIF in einem Sektor  $\triangle pP_iP_{i+1}$  zugrundeliegen: Mit  $s = |pP_i|$  gilt

$$r(\varphi) = \frac{s \cdot \sin \beta}{\sin(\varphi + \beta)}, \quad A(\varphi) = \frac{1}{2} \cdot s^2 \cdot \sin \beta \cdot \frac{\sin \varphi}{\sin(\varphi + \beta)}.$$

Nach einigen Umformungen erhält man

$$A'(\varphi) = \frac{1}{2} \cdot s^2 \cdot \sin \beta \cdot \frac{\sin \beta}{\sin^2(\varphi + \beta)} = \frac{1}{2} \cdot r^2(\varphi).$$

Die Ableitung der Flächeninhaltsfunktion ist also im wesentlichen gleich dem Quadrat der Polarkoordinatenfunktion. Zum Verständnis dieser Formel beachte man, daß der überstrichene Flächeninhalt bei einer Rotationsbewegung eines Geradenstückes nicht nur von der Länge dieser Strecke, sondern auch direkt proportional vom Abstand (des Mittelpunktes) dieser Strecke vom Drehzentrum abhängt. Diese Größen haben in unserem Fall die Werte  $r(\varphi)$  bzw.  $1/2 \cdot r(\varphi)$ , daher fließt der Ausdruck  $r(\varphi)$  zweimal in den Flächeninhalt ein. Der Übergang von  $A'(\varphi)$  zu  $A(\varphi)$  ist dann noch durch die Integration über die Drehbewegung, d.h. über  $\varphi$ , beschrieben.

Dieser interessante Zusammenhang beantwortet auch die Frage nach der Eindeutigkeit der Flächeninhaltsfunktion: Für eine gegebene Kurve  $A(\varphi)$  sind  $A'(\varphi)$  und damit auch  $r(\varphi) = \sqrt{2 \cdot A'(\varphi)}$  eindeutig bestimmt – zwei Polygone haben genau dann die gleiche Flächeninhaltsfunktion, wenn sie kongruent sind<sup>5</sup>. Die  $L_2$ -Norm gemäß Definition 2.10 ist genau in diesem Fall Null.

### 2.3.2.3 Zur linear approximierten Flächeninhaltsfunktion $\text{FIF}_{\text{lin}}$

Die linear approximierten Flächeninhaltsfunktion verzichtet auf die Information zwischen den Polygonecken. Bei der Definition der  $\text{PKF}_{\text{lin}}$  hatten wir als Legitimation eingeführt, daß durch die Approximation keine Metrikeigenschaft verlorengeht, insbesondere nicht die Eindeutigkeit. Dieser Sachverhalt soll nun auch für die  $\text{FIF}_{\text{lin}}$  überprüft werden. Dazu stellen wir uns vor, wie man bei einer Rekonstruktion eines Polygon  $\mathcal{P}$  vorgehen würde:

Man legt wieder den Punkt  $p$  in der Ebene fest und muß nun zuerst den Punkt  $P_1$  fixieren, der der Stelle  $(0, 0)$  im Graphen entspricht. Dies bedeutet nichts anderes als seinen

<sup>5</sup>Für die skalierte FIF gilt entsprechend Gleichheit bis auf Ähnlichkeitstransformationen.

Abstand zu  $p$  zu ermitteln, denn der exakte geometrische Ort ist durch die Rotationsfreiheit uninteressant. Angenommen, wir hätten diesen Abstand gefunden.

Dann können wir  $P_1$  vorgeben und haben mit dem Strahl  $p \rightarrow P_1$  auch die Referenzrichtung festgelegt (siehe Bild 2.21(a)). Aus dem Graph der Funktion können wir

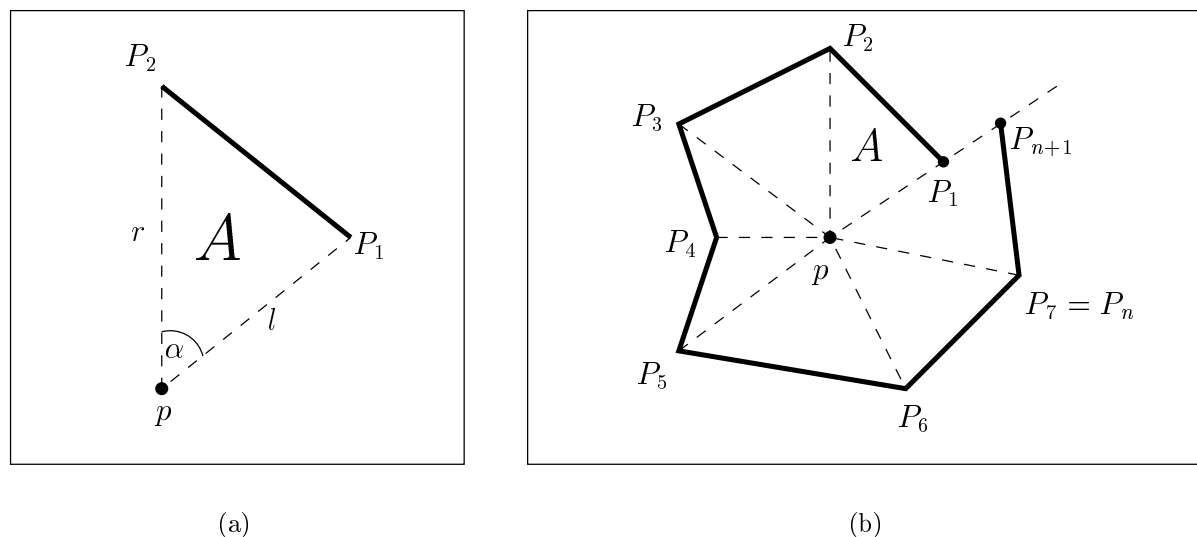


Abbildung 2.21: Rekonstruktion eines Polygons aus der  $FIF_{lin}$ : erster Sektor (a), letzter Sektor (b)

den Winkel  $\alpha$  und den Flächeninhalt  $A$  von Dreieck  $\triangle pP_1P_2$  ablesen (Breite des ersten Teilintervalls; Funktionswert an der ersten [inneren] Stützstelle). Vermöge der Gleichung

$$A = \frac{1}{2} \cdot r \cdot l \cdot \sin \alpha \quad \text{bzw.} \quad r = \frac{2A}{l \cdot \sin \alpha}$$

ist der Abstand  $|pP_2|$  eindeutig bestimmt, und  $P_2$  läßt sich elementar konstruieren. So verfährt man in der Folge mit  $P_3$  und allen weiteren Punkten.

Beim letzten Sektor ergibt sich wiederum eindeutig ein Punkt  $P_{n+1}$ , der auf dem Strahl  $p \rightarrow P_1$  liegt, da die Summe der Größen aller Intervallabschnitte  $2\pi$  beträgt. Im allgemeinen wird nun  $P_1 \neq P_{n+1}$  sein, da der Startpunkt  $P_1$  am Anfang „geraten“ wurde (Abbildung 2.21(b)).

Diese Konstruktionsvorschrift definiert eine Funktion  $z_{\mathcal{P}} : \mathbb{R}^2 \setminus \{p\} \rightarrow \mathbb{R}^2$ , die für jede Wahl von  $P_1$  mit  $P_1 \neq p$  eindeutig einen Punkt  $P_{n+1}$  auf dem Strahl  $p \rightarrow P_1$  liefert. Lösungen des Rekonstruktionsproblems sind alle Fixpunkte von  $z_{\mathcal{P}}$ , d.h. Punkte  $P_1$  mit  $z_{\mathcal{P}}(P_1) = P_1$ . Die  $FIF_{lin}$ -Darstellung ist eindeutig, wenn für jedes Polygon  $\mathcal{P}$  die zugehörige Funktion  $z_{\mathcal{P}}$  höchstens einen Fixpunkt besitzt (die Existenz ist gesichert, falls die gegebene Kurve ein gültiger  $FIF_{lin}$ -Graph ist).

Dazu muß das Verhalten von  $z_{\mathcal{P}}$  genauer untersucht werden. Betrachten wir noch einmal Abbildung 2.21(b). Um einen Fixpunkt zu erhalten, würde man intuitiv vielleicht den Punkt  $P_1$  auf den Punkt  $P_{n+1}$  hinzubewegen, also nach außen verschieben. Dann wird aber der Flächeninhalt von Dreieck  $\triangle pP_1P_2$  größer, der jedoch fest vorgegeben ist. Um dies auszugleichen, muß daher  $P_2$  nach innen wandern, d.h. auf  $p$  zu. Dies verkleinert aber Dreieck  $\triangle pP_2P_3$ , so daß  $P_3$  nach außen bewegt wird, damit der Flächeninhalt

konstant bleibt. (Man beachte, daß bei einer Variation von  $P_1$  die Lagen aller folgenden Eckpunkte  $P_i$  sich auf eindeutig bestimmte Art ebenfalls verändern.) Auf diese Weise werden offenbar alle Eckpunkte mit ungeradem Index (wie  $P_1$ ) nach außen bewegt, alle anderen nach innen. Im Beispiel gilt  $n = 7$ , also wandert  $P_{n+1} = P_8$  nach innen, und es gibt genau eine Wahl von  $P_1$ , für die  $P_8$  und  $P_1$  zusammenfallen – das Polygon ist rekonstruiert.

Dieser Erfolg beruht aber darauf, daß hier  $n$  ungerade ist. Allgemein gilt für das Verhalten der Funktion  $z_{\mathcal{P}}$ :

- Ist die Eckenzahl  $n$  von  $\mathcal{P}$  ungerade, so bewegt sich  $P_{n+1} = z_{\mathcal{P}}(P_1)$  bei einer Veränderung von  $P_1$  entlang  $p \rightarrow P_1$  entgegen der Bewegungsrichtung von  $P_1$ .
- Ist  $n$  gerade, so bewegen sich  $P_1$  und  $P_{n+1}$  stets gleichgerichtet, d.h. entweder beide nach außen oder beide nach innen auf  $p$  zu.

Für gerades  $n$  muß also die Konstruktion theoretisch nicht unbedingt zum Erfolg führen, und tatsächlich findet man relativ leicht ein Gegenbeispiel.

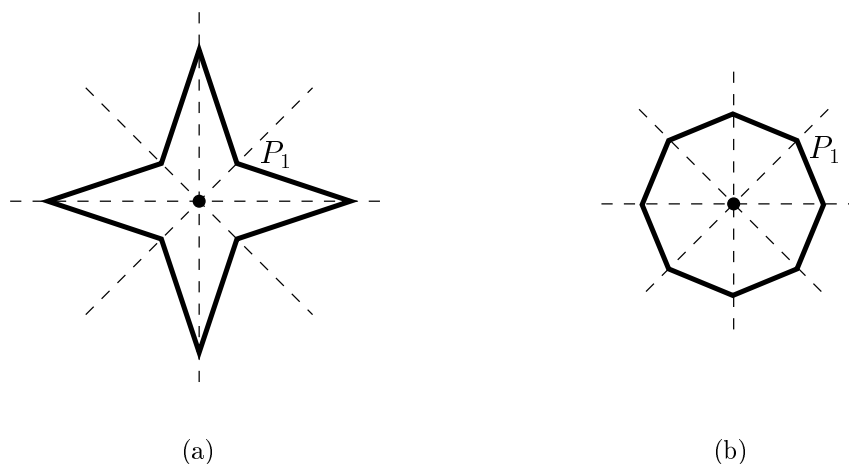


Abbildung 2.22: Zwei Achtecke mit gleicher  $FIF_{\text{lin}}$ -Darstellung

In Abbildung 2.22(a) ist ein sternartiges, achsensymmetrisches Achteck angegeben. Bewegt man hier den Punkt  $P_1$  nach außen, so wandern *alle* konkaven Ecken nach außen, während alle Spitzen sich auf  $p$  zubewegen. Irgendwann sind die Konkavecken kollinear mit ihren Nachbarn, und es ist ein Rhombus entstanden. Wird dann  $P_1$  weiter von  $p$  entfernt, so ergibt sich schließlich das regelmäßige Achteck im Bild rechts daneben. Bei dieser Umwandlung, die ein Beispiel für das in Kapitel 1.3.6 vorgestellte *Morphing* ist, bleiben sämtliche Zentriwinkel des Sterns (d.h. alle Sektorenwinkel mit  $p$  als Scheitel) unverändert, ebenso die Flächeninhalte der Sektoren. Das regelmäßige Achteck hat daher exakt die gleiche Flächeninhaltsfunktion  $FIF_{\text{lin}}$  wie der anfängliche Stern (gleiches gilt für alle Zwischenpolygone dieses Morphings).

Allerdings treten diese Mehrdeutigkeiten nicht bei allen Polygonen mit gerader Eckenzahl auf. Selbst wenn sich  $P_1$  und  $P_{n+1}$  bei der Rekonstruktion in die gleiche Richtung bewegen, kann diese Bewegung mit unterschiedlicher Geschwindigkeit erfolgen, so daß

unter Umständen trotzdem genau eine Lage von  $P_1$  existiert, für die  $P_{n+1}$  mit  $P_1$  zusammenfällt wie im ungeraden Fall. Die Klasse der Polygone, für die die Rekonstruktion aus der  $\text{FIF}_{\text{lin}}$  nicht möglich ist, wurde noch nicht genau identifiziert.

Die Metrikeigenschaft „ $\mathcal{A} = \mathcal{B} \iff s(\mathcal{A}, \mathcal{B}) = 0$ “ gilt jedenfalls für die  $\text{FIF}_{\text{lin}}$  nicht. Da sie jedoch höchstens für Polygone mit gerader Eckenzahl verletzt ist, müssen wir nach einer Möglichkeit suchen, uns auf Polygone mit ungerader Eckenzahl zu beschränken. In der Praxis kann man das wie folgt erreichen:

Als Grund für die Mehrdeutigkeiten hatten wir erkannt, daß zuviel Information *zwischen* den Eckpunkten der Polygone verschenkt wurde. Gibt man nur einen einzigen Zwischenpunkt (d.h. einen zusätzlichen Winkel mit zugehörigem Flächeninhalt) vor, so ist die Lage der Polygonkante fixiert, auf der dieser Punkt liegt, und mit ihr dann wieder das ganze Polygon.

Eine Lösung des Problems besteht also darin, ein Polygon mit gerader Eckenzahl mit zusätzlichen, ungeradzahlig vielen Stützstellen zu versehen.<sup>6</sup> Diese Lösung ist aber in der Praxis nicht angebracht. Da das Ausgangspolygon geradzahlig viele Kanten hat, kann man die zusätzlichen Punkte nicht gleichmäßig auf die Kanten verteilen. Dadurch beeinflussen sie die  $\text{FIF}_{\text{lin}}$  des Polygons ungleichmäßig, und der Distanzwert hängt von der Auswahl der Punkte ab. Zum zweiten wird sich kaum eine rotationsunabhängige Ergänzung der Punkte finden lassen.

Hier nutzt man viel besser die praktischen Gegebenheiten aus: Der Laserscanner vollführt zur Gewinnung des Sichtbarkeitspolygons einen Rundumscan und liefert in regelmäßigen Winkelabständen einen Entfernungswert zurück. Das Winkelinkrement sollte nun so eingestellt werden, daß die Anzahl der Meßpunkte nach einer Drehung ungerade ist.

### 2.3.3 Allgemeines

**Auswahl einer Funktionenmetrik.** In allen bisherigen Ähnlichkeitsabbildungen haben wir zur Messung des Abstands der Funktionsdarstellung die *Integralmetrik* für stetige Funktionen verwendet. Dabei handelt es sich allgemein um die Klasse der Metriken

$$L_p = \sqrt[p]{\int_a^b |f(x) - g(x)|^p dx} \quad (2.8)$$

für das gemeinsame Definitionsintervall  $[a, b]$  von  $f$  und  $g$ . Aus Gründen des Rechenaufwandes wird für  $p$  meistens 1 oder 2 gewählt. Die  $L_1$ -Norm ist oft einfacher zu berechnen, auch wenn  $f$  und  $g$  lineare (nichtkonstante) Funktionen sind. Die Norm  $L_2$  bietet den Vorteil, daß die Betragsstriche in Gleichung (2.8) entfallen und man sich daher bei der Integralberechnung keine Gedanken über Schnittpunkte von  $f$  und  $g$  machen muß. Dies wurde bei beiden bisher behandelten Polygonmetriken ausgenutzt. Der eingesparte organisatorische Aufwand wird bei der Integration wieder eingefordert.

Es sind auch andere Funktionsmetriken denkbar. G. Rote ([Rote92]) erwähnt die *be-*

---

<sup>6</sup>Würde man zu einem Polygon mit *ungerader* Eckenzahl (und daher eindeutiger  $\text{FIF}_{\text{lin}}$ -Darstellung) einen Punkt hinzufügen, so könnte es eventuell eine ganze Reihe von Polygonen geben, die jetzt ebenfalls diese  $\text{FIF}_{\text{lin}}$ -Darstellung haben, die nun also mehrdeutig geworden ist.

schränkte Lipschitz-Metrik  $\|f - g\|_{\text{BL}}$  mit

$$f_{\text{BL}} = \max \left\{ \int_a^b f(x)g(x)dx : |g(x)| \leq M, |g(x) - g(y)| \leq |x - y| \quad \forall x, y \right\}$$

für einen Parameter  $M$ , die Maximumsmetrik  $\|f - g\|_{\infty} = \max_{x \in [a, b]} |f(x) - g(x)|$  und die Diskrepanzmetrik

$$\|f - g\|_{\text{D}} = \max_{a \leq l \leq r \leq b} \left| \int_l^r (f(x) - g(x))dx \right|.$$

(All dies sind Metriken, da sie von Normen abgeleitet sind.) Davon ist die Maximumsmetrik sicher am leichtesten zu berechnen, wäre also ein geeigneter Kandidat. Sie ist allerdings für praktische Belange viel zu grob, wie folgendes Beispiel für die approximierete Flächeninhaltsfunktion zeigt (Abbildung 2.23). Für die Seitenlängen gelten  $a < a'$ ,

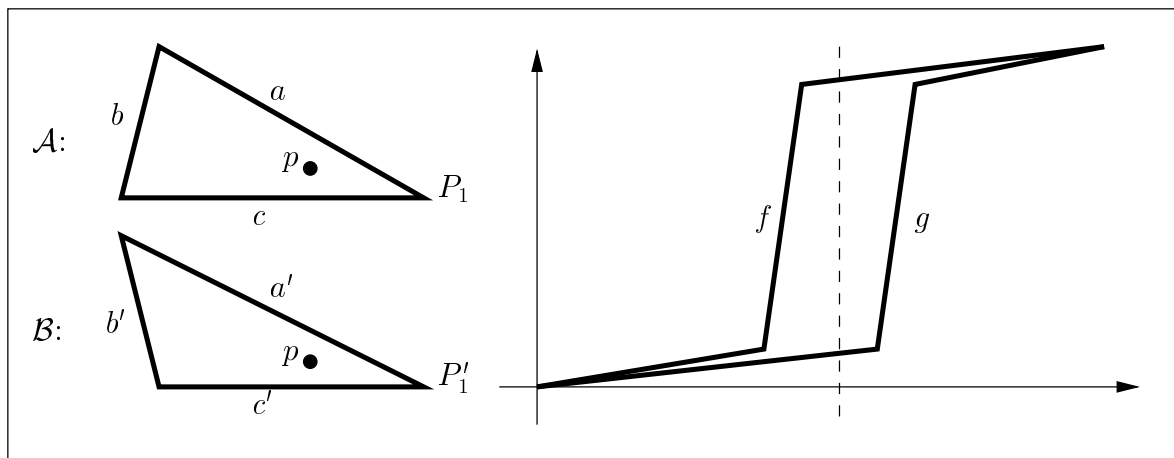


Abbildung 2.23: Maximumsmetrik für Funktionen

$b = b'$ ,  $c > c'$ . Dann ergeben sich für die Polygone  $\mathcal{A}$  und  $\mathcal{B}$  die FIF<sub>lin</sub>-Darstellungen  $f$  und  $g$ . Obwohl die Dreiecke nicht wesentlich variieren, ist der Maximumsabstand der Graphen (angenommen an der gestrichelten Linie) sehr groß.

**Winkelmaß versus Bogenlänge.** Die PKF und die FIF benutzen zum Ermitteln des Funktionswertes eines Peripheriepunktes  $P$  dasselbe Funktionsargument: den Winkel, den der Strahl vom Kernpunkt  $p$  durch  $P$  mit einer bestimmten Referenzrichtung einschließt.

Auch hier gibt es offenbar Alternativen. Die *Bogenlänge* eines Peripheriepunktes  $P$  ist definiert als die Länge des Weges, die auf dem Polygonrand von einem festzulegenden Startpunkt  $P_1$  bis zu  $P$  (meist im Gegenuhrzeigersinn) zurückzulegen ist. Auch diese Darstellung erlaubt eine Bijektion der Menge aller Peripheriepunkte auf ein reelles Intervall, in diesem Fall  $[0, u_{\mathcal{P}}]$  für den Polygonumfang  $u_{\mathcal{P}}$ . Eine andere Gemeinsamkeit beider Varianten ist, daß sie eine Festlegung benötigen, wo der Wert Null (Winkel bzw. Bogenlänge) angenommen werden soll – am Punkt  $P_1$ .

Interessanter sind die Unterschiede. Zunächst ergibt sich in den meisten Anwendungen der Bogenlänge zur funktionalen Polygonrepräsentation die Notwendigkeit einer Normierung von  $[0, u_{\mathcal{P}}]$  auf z.B.  $[0, 1]$ , da das Intervall sonst nicht als Basis bei einem Vergleich

verschiedener Graphen per Integralmetrik verwendbar ist. Das Winkelmaß hat von vorneherein eine feste Definitionsmenge  $[0, 2\pi]$ .

Umgekehrt besteht ein Nachteil der Winkeldarstellung darin, daß sie in der vorgestellten Form nur für sternförmige Polygone möglich ist. Andernfalls gibt es keinen Punkt, von dem aus ein Laserstrahl stets genau einen Peripheriepunkt trifft. Das ist aber für die Anwendung im Zusammenhang mit Sichtbarkeitspolygonen keine Einschränkung.

Die weiteren Unterschiede sind nicht so offensichtlich. Sowohl bei der PKF als auch bei der FIF ergaben sich relativ komplizierte Funktionsbeschreibungen mit gebrochenrationalen Ausdrücken in Winkelfunktionen. Um das in den jeweiligen Metriken gebildete Minimum exakt zu bestimmen, mußten wir uns auf eine lineare Approximation dieser Funktionen beschränken.

Bei der Verwendung des Bogenmaßes als Argument erhält man wesentlich einfachere Strukturen, beispielsweise wie folgt bei der Flächeninhaltsfunktion:

Der überstrichene Flächeninhalt wird als Funktion einer reellen Zahl aus  $[0, 1]$  dargestellt, die vermöge der normierten Bogenlänge eindeutig einen Peripheriepunkt  $P$  bezeichnet. Daraus ergibt sich eine einfache lineare Beziehung, wie Bild 2.24 zeigt.

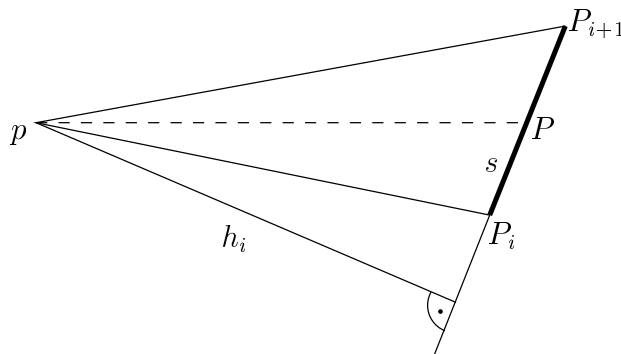


Abbildung 2.24: Überstrichene Fläche in Abhängigkeit von der Bogenlänge

Die Höhe  $h_i$  des Dreiecks  $\Delta pP_iP_{i+1}$  ist unabhängig von der Lage von  $P$  zwischen  $P_i$  und  $P_{i+1}$ . Die Länge  $s$  der Grundseite ist genau durch die (lokale) Bogenlänge des Punktes  $P$  gegeben. Betrachtet man  $s$  als Argument einer Funktion, so erhält man

$$A = s \cdot h_i/2,$$

also eine lineare Beziehung. Die (globale) Flächeninhaltsfunktion ist damit stückweise linear; die (Sektor-)Konstante  $h_i/2$  gibt den Anstieg des Graphen zwischen zwei Stützstellen an. Eine Approximation erübrigt sich, man kann die minimale  $L_2$ -Norm zwischen zwei solchen Kurven bereits in der „Originalversion“ bestimmen.

Leider ist diese Form der Flächeninhaltsfunktion nicht eindeutig: Man betrachte die beiden Polygone in Abbildung 2.25. Der Abstand von  $p$  zu den Dreiecksseiten sei jeweils gleich und stimme auch mit dem Abstand von  $p'$  zu den Quadratseiten überein. Dann ist die FIF im gesamten Intervall  $[0, 1]$  linear,<sup>7</sup> und die beiden Graphen sind nicht (zumindest mit keiner der  $L_p$ - oder der auf Seite 66 vorgestellten Metriken) unterscheidbar.

<sup>7</sup>Die inneren Stützstellen sind im Bild als schwarze Punkte für das Dreieck und als Kreise für das Quadrat dargestellt

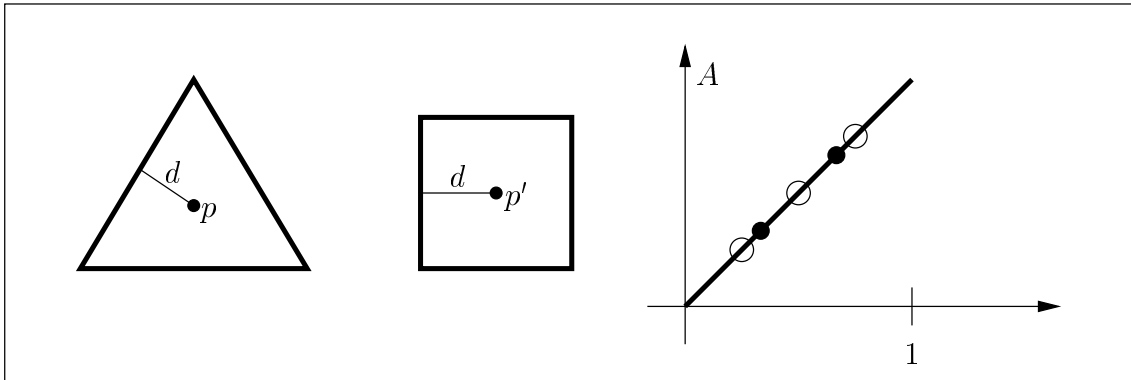


Abbildung 2.25: Zwei Polygone mit derselben FIF als Funktion der Bogenlänge

Die Identitätseigenschaft einer Metrik ist also verletzt. Es hat sogar *jedes regelmäßige  $n$ -Eck* mit Abstand  $d$  des Schwerpunktes (den wir hier einmal als den speziellen Kernpunkt annehmen) von den Seiten dieselbe FIF-Darstellung wie in Bild 2.25. Allgemein gilt:

**Beobachtung 2.15** *Ein Polygon besitzt genau dann einen inneren Punkt  $p$ , der von allen Seiten den gleichen Abstand hat, wenn sich die Winkelhalbierenden der Innenwinkel in einem Punkt schneiden; dieser erfüllt dann die Eigenschaft von  $p$ .*

Dies trifft durchaus nicht nur für regelmäßige Polygone zu, sondern z.B. ebenso für geeignete Trapeze, Rhomben etc. Diese  $n$ -Ecke haben nicht einmal alle den gleichen Flächeninhalt, da der Wert  $f(1) = A$  (siehe Bild) aufgrund der Normierung der Bogenlänge die Fläche nicht exakt wiedergibt.

Beim Kreis fallen übrigens die FIF als Funktion der Bogenlänge und die FIF als Funktion vom Strahlwinkel zusammen, denn beide Argumente sind proportional zueinander mit dem Radius des Kreises als konstantem Faktor. Die FIF-Darstellung des Kreises hat ebenfalls die Gestalt wie in Abbildung 2.25 rechts.

Ein letzter Vorteil des Winkelarguments gegenüber der Bogenlänge ist praktischer Natur. Bild 2.26 zeigt ein einseitig verrauschtes Polygon und das Original dazu.

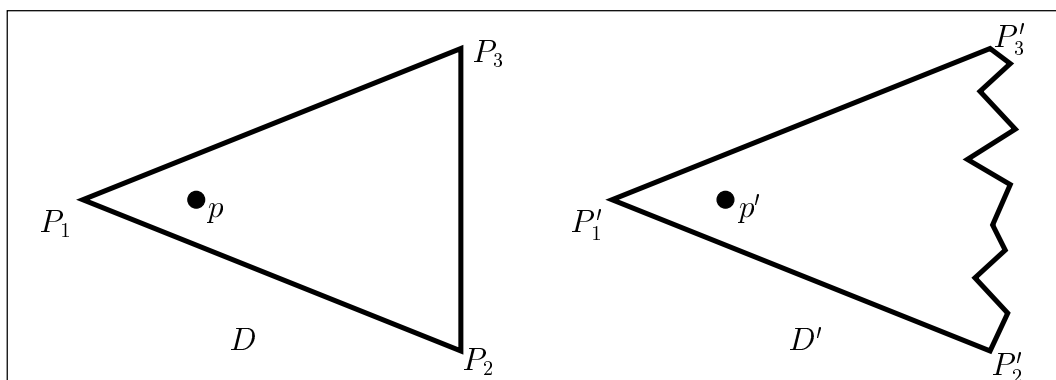


Abbildung 2.26: Einseitiges Rauschen eines Polygons

Die Größe der Teilintervalle, die den Sektoren  $\triangle pP_2P_3$  und  $\triangle p'P'_2P'_3$  entsprechen, hängt stark davon ab, ob das Funktionsargument der Winkel oder die Bogenlänge ist:

Bezeichnet  $\alpha(P)$  (bzw.  $\alpha'(P)$ ) den Winkel, den der Strahl von  $p$  (bzw.  $p'$ ) durch  $P$  mit der Horizontalen einschließt, so gilt  $\alpha(P_2) = \alpha'(P_2)$ ,  $\alpha(P_3) = \alpha'(P_3)$ . Das zugehörige Teilintervall von  $[0, 2\pi]$  ist also zwischen  $P_2$  und  $P_3$  dasselbe wie zwischen  $P'_2$  und  $P'_3$  (Winkel als Argument).

Der Bogenlängenanteil der Strecke  $\overline{P'_2P'_3}$  am Umfang von  $D'$  ist jedoch weit größer als bei  $D$ ; er kann im Stil einer fraktalen Linie beliebig groß werden. Dadurch wird auch der Einfluß der verrauschten Linie auf den Wert der Metrik beliebig hoch.<sup>8</sup> Zu dieser Problematik siehe auch die Beschreibung der *Arkin-Metrik* in Kapitel 3.1 (Seite 71).

Dieser Fall einseitigen Rauschens ist in der Roboterlokalisierung tatsächlich möglich. Man stelle sich, wie im Bild angedeutet, vor, der Roboter stehe in einem dreieckigen Zimmer nahe an einer der Ecken. Dann werden die gemessenen Entfernungswerte für die gegenüberliegende Wand stärker verrauscht sein.

Ist der Roboter in der Mitte plaziert, dann sind die Seiten gleichmäßig verrauscht. Damit erhöht sich zwar der Umfang des Polygons, aber durch das Normieren der Bogenlänge auf  $[0, 1]$  haben die drei Seiten dann den gleichen Anteil am Definitionsintervall wie im unverrauschten Fall.

Insofern hat die Normierung der Bogenlänge auch eine positive Konsequenz: sie verhindert die starke Beeinflussung des Metrikabstandes durch *gleichmäßiges* Rauschen. Für uneinheitlich verteilte Meßfehler ist sie eher von Nachteil und deshalb für die Roboterlokalisierung weniger geeignet.

---

<sup>8</sup>Zwar kommt es auch hier wieder zu einer Veränderung des Kernpunktes, die im Bild nicht dargestellt ist. Das Phänomen der unterschiedlich langen Definitionsintervalle zur Repräsentation des verrauschten Abschnitts ist jedoch davon unabhängig.

# Kapitel 3

## Ähnlichkeitsfunktionen für beliebige Polygone

Dieses Kapitel beschreibt Ähnlichkeitsfunktionen anderer Autoren, die nicht vor dem Hintergrund der Roboterlokalisierung gearbeitet haben. Dadurch sind die definierten Distanzmaße nicht aus der Arbeitsweise eines Laserscanners abgeleitet, sondern benutzen ganz allgemeine intuitive Vorstellungen vom Polygonvergleich. Dennoch gilt auch hier, wie in der Einleitung zu Kapitel 2 festgestellt, daß die Polygone zunächst in eine bestimmte Darstellung zu überführen sind, in der sie sich dann leichter vergleichen lassen. Wir werden eine weitere funktionale sowie eine zeichenkettenorientierte Polygonrepräsentation vorstellen.

Es wird sich zeigen, daß zumindest teilweise die speziellen Vorgaben der Roboterlokalisierung ausgenutzt werden können, um die allgemeinen Distanzen für diese konkrete Anwendung etwas zu verbessern.

Im folgenden gehen wir also von einfachen, aber sonst beliebigen, insbesondere nicht unbedingt sternförmigen Polygonen aus. Da die meisten Sachverhalte aus den jeweils angegebenen Quellen zitiert sind, werden zu Sätzen und Lemmata keine Beweise angeführt.

Die Problematik verrauschter Eingabedaten besteht in vielen Anwendungen und nicht nur bei der Arbeit mit einem Laserscanner. Daher werden wir uns auch in diesem Abschnitt mit Ungenauigkeiten der Eingabedaten befassen.

### 3.1 Arkin-Metrik

Lösen wir uns nun von der Vorstellung, daß ein Innenpunkt existiert, von dem das gesamte Polygon aus eingesehen werden kann. Möchte man trotzdem eine mathematische Größe haben, die eine Art Parametrisierung des Polygons ergibt (analog zum Winkel des Laserstrahls mit der Horizontalen), so kann man auf die Bogenlänge zurückgreifen, auf die bereits kurz in Abschnitt 2.3.3 auf Seite 67 eingegangen wurde. Dazu legt man einen Punkt  $P_0$  auf der Peripherie des Polygons fest, dem die Bogenlänge Null zugewiesen wird, und ordnet jedem Punkt  $P$  die Wegstrecke zu, die gegen den Uhrzeigersinn zurückzulegen ist, um  $P$  von  $P_0$  aus zu erreichen. Um diese Parametrisierung später als Argument einer Funktion verwenden zu können, die mit anderen Funktionen verglichen werden soll, normiert man die Bogenlänge auf ein konstantes Intervall, das wir hier als  $[0, 1]$  wählen.

Umgekehrt definiert dadurch jede Zahl aus  $[0, 1]$  eindeutig einen Punkt auf der Polygonperipherie. Für eine Zahl  $b > 1$  kann man sich einen eventuell mehrfachen Umlauf um das Polygon vorstellen, nach dem dennoch wieder eindeutig ein Zielpunkt bestimmt ist. Die so definierte Bogenlängenfunktion  $f$  von  $\mathbb{R}_{\geq 0}$  in die Menge der Peripheriepunkte von  $\mathcal{P}$  ist periodisch; es gilt  $f(b+1) = f(b)$  für alle  $b$  und insbesondere  $f(n) = f(0) = P_0$  für alle  $n \in \mathbb{N}$ .

### 3.1.1 Die Turning-Funktion

Nun ist eine Größe festzulegen, deren Änderung gegenüber der Bogenlänge möglichst eindeutig Auskunft über die Gestalt des Polygons gibt. Arkin et al. schlagen in [ACH<sup>+</sup>91] dazu den Anstieg der Polygonkanten gegenüber einer Referenzrichtung, etwa der Horizontalen, vor. Genauer:

**Definition 3.1** Seien  $\mathcal{P}$  ein beliebiges (einfaches) Polygon,  $P_0$  ein Punkt auf der Peripherie sowie  $f: [0, 1] \rightarrow \partial\mathcal{P}$  die Bogenlängenparametrisierung von  $\mathcal{P}$  bezüglich  $P_0$  als Startpunkt.

Die Abbildung  $\Theta: [0, 1] \rightarrow \mathbb{R}$  bildet eine reelle Zahl  $b \in [0, 1]$  auf den Winkel zwischen derjenigen orientierten Polygonkante und der  $b$ -Achse ab, auf der der Punkt  $f(b)$  liegt, und heißt **Turning-Funktion** von  $\mathcal{P}$  (bezüglich  $P_0$ ).

Ist  $f(b)$  ein Eckpunkt  $P_k$  des Polygons, so sei festgelegt, daß er auf der Kante  $\overline{P_k P_{k+1}}$  liegt.

Wird bei dem gedachten Umlauf um das Polygon eine Ecke überstrichen (und nur dann ändert sich der Wert der Turning-Funktion), so soll der Funktionswert um den Außenwinkel an dieser Ecke wachsen, wenn die Ecke konvex ist, ansonsten um diesen Betrag fallen. Nach einem vollen Umlauf um das Polygon hat der Funktionswert daher nicht wieder den Ausgangswert angenommen, sondern ist genau um  $2\pi$  größer:  $\Theta(1) = \Theta(0) + 2\pi$ .

Diese Zusatzfestlegung ist in Bild 3.1 zu erkennen: Der Funktionswert an der Stelle  $P = f(b_1)$  beträgt nicht 0, sondern  $2\pi$ . Dadurch verdient die Funktion eher den Namen

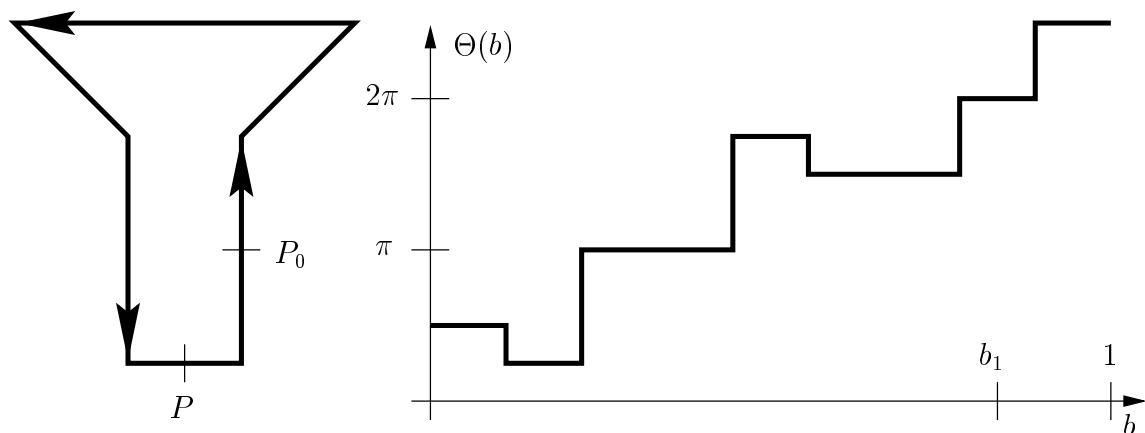


Abbildung 3.1: Turning-Funktion eines einfachen Polygons

„Turning-Funktion“: Sie beschreibt das Verhalten der Polygon*krümmung* an den Wendepunkten der Polygonperipherie, den Ecken.

### 3.1.2 Eigenschaften der Turning-Funktion

**Kurvenverlauf.** Da sich der Anstieg der Polygonkanten gegenüber einer Bezugsrichtung nur an den Ecken des Polygons ändert, ist die Turning-Funktion stückweise konstant. Die Unstetigkeitsstellen liegen genau an den Parameterwerten  $b$  vor, die den Ecken entsprechen. Die Höhe einer Sprungstelle gibt den an der betreffenden Ecke vorliegenden Außenwinkel an; ist eine Ecke kollinear mit ihren beiden Nachbarn („redundanter Eckpunkt“), so findet kein Sprung statt.

Die Funktion kann beliebig kleine und beliebig große Werte annehmen: Ein spiralförmiges Polygon bewirkt eine Verkleinerung des Funktionswertes, solange es sich im Uhrzeigersinn windet, und entsprechend eine Vergrößerung, wo es positiv orientiert ist. Da der Funktionswert  $\Theta(0)$ , also der Anstieg der ersten Polygonkante, stets im Intervall  $[0, 2\pi]$  liegt, gibt es kein Polygon, dessen Turning-Funktion ausschließlich negative Werte annimmt.

Die Längen der Polygonseiten (relativ zum Umfang) sind aus den Breiten der Streifen abzulesen, die durch die Sprungstellen definiert sind.

**Monotonie.** Oben erwähnte Zusatzfestlegung hat folgende Konsequenz:

1. Ist  $\mathcal{P}$  konvex, so ist  $\Theta$  für jeden Startpunkt  $P_0$  monoton.
2. Ist  $\mathcal{P}$  nicht konvex, so ist  $\Theta$  monoton, falls  $\mathcal{P}$  genau eine konkave Ecke  $P$  hat und  $P_0 = P$  gewählt wurde; in allen anderen Fällen ist  $\Theta$  nicht monoton.

Es gilt also:

$$\mathcal{P} \text{ konvex} \iff \Theta \text{ monoton für jeden beliebigen Startpunkt auf der Peripherie.}$$

Allgemeiner ist eine Ecke  $P$  genau dann eine Konvexecke, wenn an der entsprechenden Sprungstelle im Graph  $\Delta\Theta$  größer als Null ist.

**Polygontransformationen, Startpunktwahl.** Die Turning-Funktion ist sowohl gegenüber Translationen als auch Skalierungen des Eingabepolygons unabhängig: Weder die Anstiege der Kanten noch die Seitenlängen *relativ zum Umfang* (Normierung auf  $[0, 1]$ ) ändern sich unter diesen Ähnlichkeitstransformationen.

Eine Rotation des Eingabepolygons um einen Winkel  $\theta$  gegen den Uhrzeigersinn bedeutet, daß sich der Anstieg jeder Polygonseite um  $\theta$  ändert. Dadurch wird der Funktionsgraph parallel nach oben ( $\theta > 0$ ) oder nach unten ( $\theta < 0$ ), d.h. in Richtung von  $\Theta$ , verschoben.

Schließlich besitzt die Funktion einen Startpunktparameter  $P_0$ . Verschiebt man diesen Punkt entlang der Peripherie, so werden alle Anstiege an bestimmten Peripheriepunkten zeitlich verzögert oder im Voraus aufgezeichnet, wenn man sich die  $b$ -Achse als Zeitachse vorstellt. Das entspricht, ähnlich wie bei der Polarkoordinatenfunktion, einer Verschiebung des Graphen in Richtung dieser Achse, also horizontal. Genauer wird der Graph nach rechts verschoben, wenn  $P_0$  gegen den Uhrzeigersinn bewegt wird, ansonsten nach links.

### 3.1.3 Ableitung der Metrik

Das Prinzip zur Ableitung einer Metrik ist wiederum, die beiden Funktionsgraphen zweier Polygone zu vergleichen. Um die Ähnlichkeitsbestimmung unter Rotationen und der (willkürlichen) Wahl des Startpunktes  $P_0$  invariant zu machen, muß über diese beiden Parameter minimiert werden.

Die Rotation um  $\theta$ , die einer Verschiebung des Graphen in  $\Theta$ -Richtung entsprach, kann dabei durch den Ausdruck  $\Theta(x) + \theta$  modelliert werden. Eine Startpunktveränderung verschiebt den Graphen horizontal, was mittels  $\Theta(x + b)$  parametrisierbar ist. Der Parameter  $b$  gibt dabei die Weglänge (relativ zum Umfang) an, um die  $P_0$  verschoben wurde:  $b > 0$  bedeutet Verschiebung gegen den Uhrzeigersinn. Kombiniert man diese beiden Parametrisierungen, so erhält man folgende

**Definition 3.2** *Seien  $\Theta_{\mathcal{A}}$  und  $\Theta_{\mathcal{B}}$  die Turning-Funktionen zweier Polygone  $\mathcal{A}$  und  $\mathcal{B}$ . Die Abbildung*

$$d(\mathcal{A}, \mathcal{B}) = \sqrt{\min_{\theta \in \mathbb{R}, b \in [0,1]} \int_0^1 (\Theta_{\mathcal{A}}(x + b) - \Theta_{\mathcal{B}}(x) + \theta)^2 dx}$$

heißt **Arkin-Metrik** auf der Menge aller einfachen Polygone.

Statt der hier verwendeten  $L_2$ -Norm wäre auch jede andere  $L_p$ -Norm denkbar gewesen. Bei der Beschreibung der Berechnung dieser Metrik konzentrieren sich Arkin et al. jedoch auf den Fall  $p = 2$ , auf den wir uns daher beschränken wollen.

Der Beweis der Metrik-Eigenschaften, der von der Wahl von  $p$  unabhängig ist, ist in [ACH<sup>+</sup>91] nachzulesen.

Damit hat man zunächst ein zweidimensionales Minimierungsproblem vor sich. Bei der Flächeninhaltsfunktion gelang es nicht, ohne deutliche Vereinfachungen an der Definition die Minimierung über die beiden Parameter effizient zu berechnen. Da die Turning-Funktion jedoch stückweise konstant ist und damit eine sehr einfache Gestalt hat, läßt sich die Menge aller Stellen aus  $\mathbb{R} \times [0, 1]$ , die für die minimierenden Werte  $(\theta, b)$  in Frage kommen, stark reduzieren:

**Lemma 3.3** *Sei für das Integral aus Definition 3.2 der Parameter  $b$  fest gewählt. Der minimierende Wert für  $\theta$  ist dann gegeben durch*

$$\theta^* = c - 2\pi b \quad \text{mit} \quad c = \int_0^1 \Theta_{\mathcal{B}}(x) dx - \int_0^1 \Theta_{\mathcal{A}}(x) dx.$$

Ersetzt man also in Definition 3.2 den Parameter  $\theta$  durch  $\theta^*$ , so erhält man ein eindimensionales Minimierungsproblem in  $b$ .

Wiederum da  $\Theta_{\mathcal{A}}$  und  $\Theta_{\mathcal{B}}$  stückweise konstant sind, kann man zeigen, daß der minimierende Wert  $b$  einer horizontalen Verschiebung des Graphen von  $\Theta_{\mathcal{A}}$  entspricht, bei der eine Unstetigkeitsstelle von  $\Theta_{\mathcal{A}}$  und eine von  $\Theta_{\mathcal{B}}$  zusammenfallen. Davon gibt es nur  $m \cdot n$  viele (für Polygone mit  $m$  bzw.  $n$  Ecken), so daß ein  $\mathcal{O}(mn \cdot (m + n))$ -Algorithmus zur Berechnung der Metrik unmittelbar folgt: Genau wie bei der (nichtapproximierten) Polarkoordinaten- und der Flächeninhaltsfunktion ist für jede der  $m \cdot n$  Lagen der beiden Graphen das Gesamtintegral durch Aufsummieren der Einzelwerte in den  $m + n$  Streifen

zu bilden. Es sei aber nochmals der Unterschied erwähnt, daß bei den beiden anderen Metriken der globale Minimalwert nicht unbedingt für ein Zusammenfallen zweier Übergangsstellen angenommen werden muß, sondern möglicherweise für eine Zwischenlage.

Schließlich läßt sich der Algorithmus zur Berechnung der Metrik noch verbessern, so daß man eine Laufzeitkomplexität von  $\mathcal{O}(mn \cdot \log mn)$  erhält, was erneut die stückweise Konstanz der beiden beteiligten Funktionen inhärent ausnutzt.

### 3.1.4 Die Arkin-Metrik in der Roboterlokalisierung

In Abschnitt 3.1.2 wurde festgestellt, daß die Darstellung eines Polygons durch seine Turning-Funktion skalierungsunabhängig ist. Das ist in der Roboterlokalisierung unerwünscht. Da dieses Problem auch im Zusammenhang mit der Flächeninhaltsfunktion FIF auftrat, sei auf Kapitel 2.2.5 auf Seite 60 verwiesen, wo eine prinzipielle Lösungsmöglichkeit angegeben ist.

Das Argument der Turning-Funktion ist die Bogenlängenposition des Peripheriepunktes, in dem der Tangentialanstieg gegenüber der Horizontalen gemessen wird. In Kapitel 2.3.3 auf Seite 67 wurden einige kritische Anmerkungen zur Bogenlänge als Argument der FIF gemacht, die leider in dieser Form auch für die Turning-Funktion gelten. Uneinheitliches Rauschen, wie es in Bild 2.26 (Seite 69) dargestellt ist, verändert nicht nur die Turning-Funktion sehr, ohne daß das Polygon gegenüber dem Original sehr unähnlich ist. Es läßt darüberhinaus den Einfluß gerade der einen verrauschten Seite beliebig stark werden, da das Teilintervall von  $[0, 1]$ , das diese Seite repräsentiert, im Verhältnis zu den anderen Seiten beliebig lang werden kann.

Arkin et al. bemerken zu diesem Problem ([ACH<sup>+</sup>91, Seite 210]), daß in Anwendungen der *Computer Vision* hauptsächlich einheitliches Rauschen zu erwarten ist. Das ist in der Roboterlokalisierung mittels Laser-Radar leider nicht der Fall, wie ebenfalls im Zusammenhang mit Abbildung 2.26 erläutert wurde.

In oben erwähntem Kapitel wurde diese Problematik als ein spezieller Nachteil der Bogenlängenrepräsentation der Peripheriepunkte aufgeführt. Da wir in der Roboterlokalisierung von der zusätzlichen Eigenschaft der Sternförmigkeit der Eingabepolygone ausgehen können, bietet es sich an, statt der Bogenlänge wiederum den Strahlwinkel als Argument zu benutzen (ansonsten aber die Definition der Turning-Funktion beizubehalten).

**Definition 3.4** Sei  $\mathcal{P}$  ein sternförmiges Polygon, für das wie im Fall der PKF und der FIF ein spezieller Kernpunkt  $p$  festgelegt sei. Dann ist als Parametrisierung der Polygonperipherie sowohl die Bogenlänge als auch der Winkel eines Strahls startend in  $p$  geeignet. Die Turning-Funktionen, die die jeweilige Parametrisierung als Argument benutzen, seien mit  $\mathbf{TF}_{BL}$  und  $\mathbf{TF}_{WKL}$  bezeichnet.

Für alle regelmäßigen  $n$ -Ecke sind die  $\mathbf{TF}_{BL}$  und die  $\mathbf{TF}_{WKL}$  identisch (abgesehen von der unterschiedlichen Länge des Definitionsintervalls: 1 bzw.  $2\pi$ ). Dort sind nicht nur alle Seiten gleich lang, sondern auch die Zentriwinkel aller Sektoren gleich groß, denn der spezielle Kernpunkt ist hier der Umkreismittelpunkt. Daher sind die Anteile aller Polygonkanten am Graphen jeweils gleich ( $1/n$  bzw.  $2\pi/n$ ).

Für unregelmäßige Polygone unterscheiden sich die beiden Varianten der Turning-Funktion. Bild 3.2 zeigt die Graphen der  $\mathbf{TF}_{BL}$  und der  $\mathbf{TF}_{WKL}$  für das einseitig verrauschte Dreieck  $D'$  aus Abbildung 2.26. Die Graphen für das unverrauschte Dreieck  $D$  in jenem

Bild links sind zum Vergleich jeweils dünn daruntergezeichnet, allerdings etwas versetzt, um sie von den anderen Graphen unterscheiden zu können. Der Startpunkt  $P_0$  liege in  $D$  bei  $P_2$  und in  $D'$  bei  $P'_2$ .

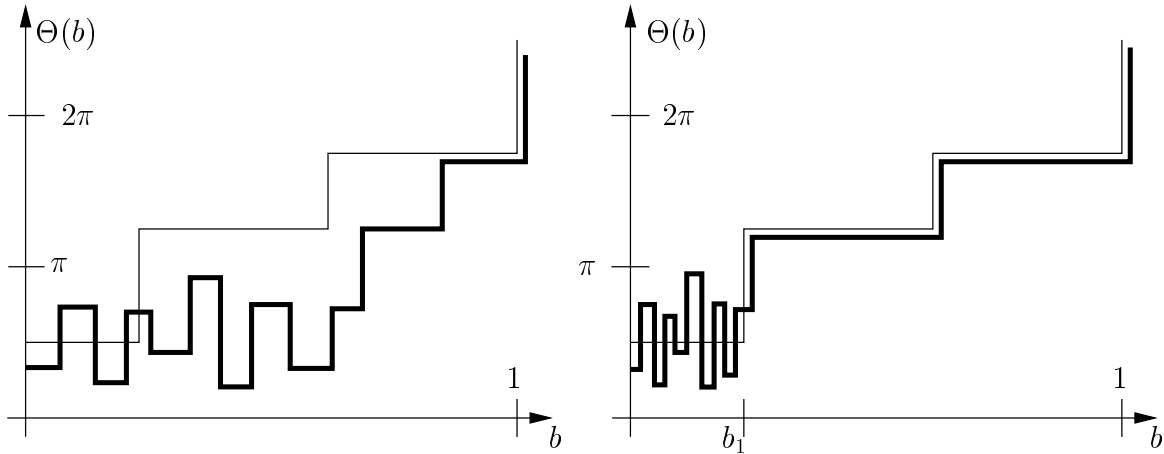


Abbildung 3.2: Turning-Funktion des einseitig verrauschten Polygons aus Abbildung 2.26 (Seite 69) mit Bogenlängenargument (links) und mit Winkelargument (rechts) im Vergleich zum unverrauschten Polygon (dünn)

Man erkennt deutlich, daß bei der Turning-Funktion mit Bogenlängenargument die Verhältnisse durch das Rauschen stark verzerrt werden, wodurch der  $L_2$ -Abstand der beiden Graphen sehr groß wird. Im Bild rechts daneben tritt dieser Effekt nicht auf.

Allerdings ist selbst bei gleichmäßigem Rauschen ein erheblicher Distanzwert zu erwarten. Man betrachte dazu das Teilintervall  $[0, b_1]$  auf der  $b$ -Achse im rechten Bild, wo also zumindest die *Verzerrungen* durch das Rauschen ausgeschaltet sind. In diesem Intervall oszilliert der Graph der Turning-Funktion stark um das Geradenstück, daß die exakte Kurve wiedergibt (dünne Linie im Hintergrund). Die Ursache ist in der Art des Entstehens des Rauschens zu suchen:

Ein Laserscanner bestimmt Entfernungswerte, keine Anstiegswinkel gegenüber Bezugslinien. Dadurch wird sich Rauschen auch in erster Linie durch falsche Entfernungen bemerkbar machen. Die dadurch ebenfalls auftretenden Schwankungen von Kantenorientierungen sind nur mittelbar auf das Rauschen zurückzuführen und daher in ihrer Quantität nicht abschätzbar: Selbst bei geringen Fehlern in der Entfernungsmessung kann sich der Winkel zwischen inzidenten Polygonkanten beliebig stark ändern. Dieser Umstand tritt um so stärker hervor, je kürzer die betroffenen Seiten sind. Da aber bei verrauschten Polygonen meist kürzere Segmente zu erwarten sind als real vorhanden (durch die größere Anzahl von Eckpunkten), dürfte der Einfluß von *Winkelschwankungen* erheblich sein.

Auf diese Fehlerquelle nimmt die Turning-Funktion wenig Rücksicht, da sie explizit auf Winkelgrößen aufbauend definiert wurde. Für den Fall von „Entfernungsrauschen“ wie bei der Roboterlokalisierung sind die PKF-Metrik und auch die FIF-Metrik weniger anfällig: Die PKF mißt direkt Entfernungen, die dann nur den direkten Fehlern des Scanners unterliegen und nicht durch geometrische Sachverhalte unnötig vergrößert werden. Ähnlich ist der Flächeninhalt der Sektordreiecke in der Summe bzw. im Verhältnis zum aktuellen Winkel des Fahrstrahls wenig beeinflußbar durch *geringe* Entfernungsschwankungen, wie sie in Bild 2.26 vorlagen.

### 3.1.5 Abschließendes zur Arkin-Metrik

Zusammenfassend läßt sich über die Arkin-Metrik sagen: Die Idee, Polygonkantenanstiege in Abhängigkeit eines Peripherieparameters aufzuzeichnen, ergibt eine analytisch sehr einfache Funktionsdarstellung von Polygonen. Die abgeleitete Metrik ist daher sehr effizient berechenbar und leicht implementierbar. Außerdem ist sie anwendbar für *alle* Polygone (mit einigen Zusatzfestlegungen sogar für überschlagene). In diesen Punkten ist sie den in Kapitel 2 vorgestellten Distanzen deutlich überlegen.

Ähnlich wie die Flächeninhalts- und die Polarkoordinatenmetrik liefert die Arkin-Metrik minimierende Werte für  $\theta$  und  $b$  zurück, die zur Ableitung eines Polygon-Matchings genutzt werden können: Das optimale  $\theta^*$  gibt die Drehung des Polygons  $\mathcal{A}$  gegen den Uhrzeigersinn an, der Wert  $b^*$  bestimmt eindeutig einen Peripheriepunkt von  $\mathcal{A}$ , der durch eine Translation auf den vorher fest gewählten Startpunkt  $P_0$  von  $\mathcal{B}$  zu verschieben ist. Über die Güte dieses Matchings werden in [ACH<sup>+</sup>91] keine Angaben gemacht.

Speziell in der Roboterlokalisierung ist jedoch Rauschen ein derart bedeutender Faktor, daß etwas mehr Rechen- und Zeitaufwand in Kauf genommen werden wird, wenn dadurch Ungenauigkeitseffekte begrenzt werden können. Die Arkin-Metrik führt bei ungleichmäßigem Rauschen zu einer mehr oder weniger starken Verzerrung der Polygondarstellung in waagerechter Richtung; die Zuordnung von Teilintervallen von  $[0, 1]$  und Polygonkanten ändert sich vollkommen. Diesen Effekt kann man bei *sternförmigen* Polygonen (und in der vorgestellten Form nur bei diesen) noch leicht beseitigen, indem man zum Winkelargument übergeht. Da aber Entfernungsruschen selbst bei kleinen Fehlern zu sehr hohen Winkelschwankungen führen kann, würde diese Lösung kein befriedigendes Resultat im Hinblick auf die Meßfehlerabhängigkeit ergeben.

Außerdem ist die Skalierungsunabhängigkeit der Metrik eine unerwünschte Eigenschaft, die nicht – wie bei der FIF-Metrik – durch eine per Definition durchgeführte Skalierung entstanden ist, sondern durch die Skalierungsunabhängigkeit der Meßgröße, nämlich der Anstiegswinkel.

## 3.2 Maes-Distanz

Während alle bisher vorgestellten Polygondistanzen auf einer funktionalen Darstellung der Eingabepolygone beruhen, soll nun zum Abschluß eine ganz andere Methode vorgestellt werden.

### 3.2.1 Polygondarstellung

Ein Polygon läßt sich als eine endliche Folge von Punkten oder Strecken beschreiben. Daher haben einige Autoren versucht, Polygondistanzen über den Vergleich von Folgen bestimmter (mathematischer) Objekte zu bestimmen (siehe zum Beispiel die *Kedem-Distanz* in [HK90]). Eine spezielle Art solcher Folgen sind Zeichenketten.

Die Darstellung eines Polygons als String wirft zunächst die Frage nach einem Alphabet auf. Über einem endlichen Zeichenvorrat können endliche Strings auch nur endlich viele Objekte repräsentieren. Polygone unterscheiden sich aber selbst bei vorgegebener Eckenzahl durch quantitative Merkmale wie Seitenlängen und Innenwinkel.

Als Lösung dieses Problems könnte man einfach ein unendliches Alphabet zulassen, etwa  $\mathbb{R}$ , und ein Polygon als Folge seiner Kantenlängen und Innenwinkel darstellen. Um Streckenlängen und Winkel leichter voneinander unterscheiden zu können, schlägt *Maes* jedoch vor, als Alphabet eine Menge zweier Symbole  $\Sigma = \{W, L\}$  zu benutzen, die einen Winkel bzw. eine Strecke repräsentieren ([Maes94]). Größe und Länge werden durch ein reelles Attribut angegeben, das jedem Vorkommen von  $W$  und  $L$  in einem String angefügt wird.

Eine andere Möglichkeit der Unterscheidung von Strecken und Winkeln ist die Vereinbarung, in der Polygonrepräsentation Streckenlängen an allen ungeraden Positionen, Winkelgrößen an allen geraden Positionen des Strings zu notieren. Dies ist möglich, da sich Winkel und Strecken entlang der Polygonperipherie abwechseln.

Die Notwendigkeit der Unterscheidung ergibt sich aus dem Ziel, daß die Ähnlichkeit von Polygonen später über ihre Stringdarstellungen bestimmt werden soll. Dabei dürfen natürlich stets nur Strecken mit Strecken sowie Winkel untereinander verglichen werden.

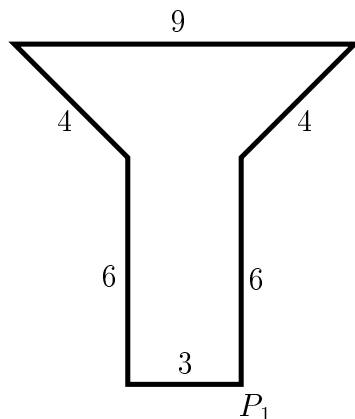
**Definition 3.5** Seien  $\mathcal{P}$  ein einfaches Polygon und  $P_1$  einer seiner  $n$  Eckpunkte. Seien weiter  $(\bar{\alpha}_1, \dots, \bar{\alpha}_n)$  die Folge seiner Innenwinkelgrößen und  $(l_1, \dots, l_n)$  die Folge seiner Seitenlängen, jeweils beginnend am Punkt  $P_1$ . Dann heißt die Zeichenkette

$$A_{\mathcal{P}} := (\alpha_1, l_1, \dots, \alpha_n, l_n) \quad \text{mit} \quad \alpha_i := 180^\circ - \bar{\alpha}_i$$

die **Stringrepräsentation** von  $\mathcal{P}$  mit Startpunkt  $P_1$ .

Um obiger Unterscheidung von Streckenlängen und Winkelgrößen gerecht zu werden, gehen wir davon aus, daß jedem Symbol des Strings angesehen werden kann, ob es sich um eine Länge oder einen Winkel handelt. Strenggenommen müßte jedes solche Symbol ein geordnetes Paar aus einem „Typspezifikator“ und einer reellen Zahl sein. – Der Grund für die Verwendung von  $180^\circ - \bar{\alpha}_i$  statt der Innenwinkel  $\bar{\alpha}_i$  wird in Abschnitt 3.2.3 klarwerden.

Abbildung 3.3 zeigt für das bekannte Beispielpolygon aus Bild 3.1 nun die Stringrepräsentation.



Stringrepräsentation:

$$A_{\mathcal{P}} = (90^\circ, 6, -45^\circ, 4, 135^\circ, 9, \\ 135^\circ, 4, -45^\circ, 6, 90^\circ, 3)$$

Abbildung 3.3: Stringrepräsentation eines einfachen Polygons  $\mathcal{P}$

### 3.2.2 Eigenschaften der Stringdarstellung

Jeder String, der nach Definition 3.5 ein Polygon repräsentiert, hat gerade Länge, und die Symbole des Strings bezeichnen abwechselnd Seitenlängen und Winkelgrößen. Für die Winkel  $\alpha_i$  an der Ecke  $i$  gilt:

$$\begin{aligned}\alpha_i > 0 &\implies i \text{ ist konvex,} \\ \alpha_i = 0 &\implies i \text{ ist redundant, d.h. kollinear mit ihren beiden Nachbarn,} \\ \alpha_i < 0 &\implies i \text{ ist konkav.}\end{aligned}$$

Offensichtlich ist die Darstellung eines Polygons durch Strings gemäß obiger Definition unabhängig von Translationen und Rotationen, denn es werden nur Seitenlängen sowie relative Winkelgrößen im Innern des Polygons gemessen.

Die Stringdarstellung hat einiges mit der Repräsentation durch die Turning-Funktion aus Kapitel 3.1.1 gemeinsam: Auch dort werden Winkel (in Form von Kantenanstiegen) und Seitenlängen (durch die Längen der Teilintervalle von  $[0, 1]$ ) zur Beschreibung benutzt. Die Peripherie des Polygons ist in Definition 3.5 gewissermaßen durch die Zahlenfolge  $(1, 2, \dots, 2n - 1, 2n)$  parametrisiert; der Parameter gibt die Position der  $i$ -ten Kantenlänge bzw. des  $j$ -ten Winkels im String an.

Ein wichtiger Unterschied ist allerdings, daß hier keine Normierung des Polygonumfangs auf 1 notwendig ist, denn wir haben es nicht mit einer funktionalen Darstellung zu tun. Dadurch brauchen wir kein normiertes Definitionsintervall, um etwa eine Integraldistanz ausrechnen zu können.

Die Folge ist, daß eine später abgeleitete Ähnlichkeitsfunktion über Stringvergleich nicht automatisch skalierungsinvariant ist, wie es bei der Arkin-Metrik der Fall war. Es sei erneut daran erinnert, daß wir für die Roboterlokalisierung skalierungssensitive Distanzen brauchen!

Die Darstellung hängt aber vom Startpunkt  $P_1$  ab. Die Wahl einer neuen Startecke bewirkt ein zyklisches Verschieben des Strings aus Definition 3.5.

### 3.2.3 Stringvergleich über Edit-Distanzen

Zur Ableitung einer Distanzfunktion für Polygone vergleicht man nun also ihre Stringdarstellungen. Die *Edit-Distanz* setzt dazu drei Operationen zur String-Manipulation voraus und beschreibt die Ähnlichkeit von Strings durch die Gesamtkosten an Operationen, die zur Überführung des einen Strings in den anderen nötig sind.

**Definition 3.6** *Seien  $\Sigma$  ein beliebiges Alphabet und speziell  $\lambda$  das leere Wort. Dann lassen sich durch  $x \rightarrow y$ ,  $x \rightarrow \lambda$  und  $\lambda \rightarrow y$  für  $x, y \in \Sigma$  Änderungs-, Lösch- und Einfügeoperationen beschreiben, denen jeweils eine nichtnegative Zahl  $c$  als Kostenmaß zugeordnet sei.*

*Für zwei Wörter  $v, w \in \Sigma^*$  ist eine **Edit-Sequenz von  $v$  nach  $w$**  definiert als eine Folge von Edit-Operationen obiger Art, die  $v$  in  $w$  überführt. Die Kosten dieser Sequenz*

---

<sup>1</sup>Gleichwohl wurde in [Maes94] die Polygondarstellung in der skalierten Version eingeführt, wie es in vielen anderen Anwendungen angemessen sein mag.

sind definiert als die Summe der Kosten aller Sequenzoperationen. Die **Edit-Distanz** zweier Wörter  $v, w \in \Sigma^*$  ist

$$d_{\text{edit}}(v, w) = \min\{C \in \mathbb{R}_{\geq 0} : C \text{ sind die Kosten einer Edit-Sequenz, die } v \text{ in } w \text{ überführt.}\}.$$

Ob  $d_{\text{edit}}$  eine Metrik ist, hängt von den Kostenfunktionen für die einzelnen Operationen ab. Definiert man zum Beispiel

$$c(x \rightarrow \lambda) := 1, \quad c(\lambda \rightarrow y) := 1 \quad \text{und} \quad c(x \rightarrow y) := \begin{cases} 1 & \text{für } x \neq y \\ 0 & \text{für } x = y \end{cases},$$

so ist  $d_{\text{edit}}(v, w)$  genau die minimale Anzahl von Edit-Operationen zur Überführung von  $v$  in  $w$ , und  $d_{\text{edit}}$  ist eine Metrik. Diese *diskrete Kostenfunktion* ist aber für unsere Zwecke nicht geeignet, da es zum Beispiel bei der Überführung einer Strecke eines Polygons in eine andere durchaus auf die genauen Längen der Strecken ankommen soll und nicht nur darauf, ob sie gleichlang sind oder nicht. Dies kann man wie folgt erreichen:

**Definition 3.7** Für alle  $x, y \in \Sigma$  seien Kostenfunktionen der Editoperationen definiert durch

$$c(x \rightarrow \lambda) = c(\lambda \rightarrow y) := \begin{cases} |x| & \text{falls } x \text{ ein Winkel ist} \\ \omega \cdot x & \text{falls } x \text{ eine Seitenlänge ist} \end{cases}$$

und

$$c(x \rightarrow y) := \begin{cases} |x - y| & \text{für Winkel } x \text{ und } y \\ \omega \cdot |x - y| & \text{für Seitenlängen } x \text{ und } y \\ \infty & \text{sonst} \end{cases},$$

wobei  $\omega \in \mathbb{R}_{>0}$  ein Gewichtungsfaktor ist, mit dem man regeln kann, wie die Kosten für Operationen auf Seiten und Winkeln relativ zueinander die Gesamtkosten beeinflussen.

Laut [Maes94] hängt die Wahl des Parameters  $\omega$  stark von der aktuellen Anwendung ab, aber zum Beispiel auch davon, ob die Winkel in Definition 3.5 in Grad oder Radiant gemessen werden.

Hier wird nun auch deutlich, warum in Definition 3.5 als Winkelgrößen die Werte  $180^\circ - \bar{\alpha}_i$  statt der Innenwinkel  $\bar{\alpha}_i$  gespeichert wurden: Die Kosten des Einfügens oder Löschens eines Winkels sind genau dann Null, wenn der Winkel  $180^\circ$  beträgt; sie sind maximal, wenn der Winkel nahe bei  $0^\circ$  oder nahe bei  $360^\circ$  liegt. Unter „Einfügen eines Winkels“ ist dabei das Splitten einer bestehenden Polygonkante zu verstehen, und zwar so, daß die beiden entstandenen Seiten nach der Operation einen Winkel wie angegeben miteinander einschließen. Ist dieser Winkel  $180^\circ$ , so bedeutet das Splitten nur das Einfügen eines redundanten Eckpunktes ohne Veränderung der äußeren Form des Polygons. Es ist daher sinnvoll, genau in diesem Fall die Kosten als 0 zu definieren.

Für die Definition der Polygondistanz ist nun wieder über die Wahl aller Startpunkte zu minimieren. Um ein exaktes Matching zu erhalten, müßte als Startpunkt jeder Punkt der Polygonperipherie zugelassen werden, nicht nur Eckpunkte. Dazu könnte man an der

betreffenden Peripheriepunktstelle einfach einen weiteren (redundanten) Eckpunkt einfügen. Dann wäre die Stringrepräsentation auch für die ausgewählte Stelle als Startpunkt definiert.

Dies verursacht aber zwei entscheidende Schwierigkeiten. Die erste ist als das *Segmentierungsphänomen* bekannt und wird weiter unten beschrieben (Abschnitt 3.2.5). Das andere Problem besteht darin, daß für das spätere Stringmatching nur dann ein effizienter Algorithmus bekannt ist, wenn die Startpunktwahl auf die Polygonecken beschränkt ist. Eine Startveränderung entspricht dann, wie gesagt, einem zyklischen Shiften, was das String-Matching gegenüber dem für zwei feste Strings lediglich um einen logarithmischen Faktor teurer macht. (Zu Einzelheiten des Algorithmus' siehe ebenfalls weiter unten.)

Damit leidet die Maes-Distanz an einem ähnlichen Umstand wie die nichtapproximierte PKF-Distanz: Eine kontinuierliche Minimierung über alle möglichen Startpunkte ist bisher nicht effizient gelöst, und man beschränkt sich auf die Werte für spezielle Startpunkte. Dadurch sind auch bei der Maes-Distanz keine Metrikeigenschaften zu erwarten.

Um aber zumindest Eckpunkte als Startpunkte zuzulassen, beschreibt man zunächst ein Polygon wie in Definition 3.5 angegeben und repräsentiert es dann durch einen *zyklischen String*  $[A_{\mathcal{P}}]$ , der definiert ist als die Äquivalenzklasse aller zyklischen Vertauschungen von  $A_{\mathcal{P}}$ :

**Beobachtung 3.8** Sei  $S_n$  für  $n \geq 0$  die Menge aller Strings der Länge  $n$  über einem beliebigen Alphabet. Dann wird durch

$$\begin{aligned} s_1 \sim s_2 & : \iff \text{Es existiert eine zyklische Vertauschung } \tilde{s}_1 \text{ von } s_1 \text{ mit } \tilde{s}_1 \equiv s_2 \\ & (\iff \text{Es existiert eine zyklische Vertauschung } \tilde{s}_2 \text{ von } s_2 \text{ mit } \tilde{s}_2 \equiv s_1 \end{aligned}$$

eine Äquivalenzrelation auf  $S_n$  definiert, deren Äquivalenzklassen  $[s]$  als **zyklische Strings** bezeichnet werden.

Die Darstellung  $[A_{\mathcal{P}}]$  eines Polygons ist nun unabhängig von Translation, Rotation und Startpunktwahl, und wir können die Ähnlichkeitsfunktion für Polygone aufbauend auf String-Vergleich definieren:

**Definition 3.9** Seien  $\mathcal{P}$  und  $\mathcal{Q}$  zwei Polygone und  $[A_{\mathcal{P}}]$  und  $[A_{\mathcal{Q}}]$  ihre zyklischen Stringdarstellungen. Bezeichne weiter für einen String  $A$  der Länge  $n$  und eine natürliche Zahl  $k$  der Ausdruck  $\sigma^k(A)$  die  $k$ -te zyklische Rotation von  $A$ , also insbesondere  $\sigma^n(A) = A$ . Dann definiert

$$m(\mathcal{P}, \mathcal{Q}) := d_{edit}([A_{\mathcal{P}}], [A_{\mathcal{Q}}])$$

die **Maes-Distanz** der Polygone  $\mathcal{P}$  und  $\mathcal{Q}$ . Dabei ist

$$d_{edit}([A], [B]) := \min\{d_{edit}(\sigma^k(A), B) : k \in \mathbb{N}\}$$

die *Edit-Distanz* zweier zyklischer Strings  $A$  und  $B$ .

Es ist leicht einzusehen, daß es genügt, nur die zyklischen Rotationen *eines* der beiden Strings  $A$  und  $B$  zu betrachten. Man kann für  $A$  sogar den kürzeren der beiden Strings nehmen und dadurch die Laufzeit des folgenden Algorithmus' positiv beeinflussen.

### 3.2.4 Berechnung der Maes-Distanz

Betrachten wir zunächst das *statische Stringmatching-Problem*, bei dem es darum geht, für zwei Strings (und nicht für ihre zyklische Äquivalenzklasse) die Edit-Distanz und eine minimierende Edit-Sequenz zu bestimmen. In [WF74] wird dazu ein effizienter Algorithmus der Laufzeit  $\mathcal{O}(m \cdot n)$  für Zeichenketten der Länge  $m$  bzw.  $n$  vorgestellt. Dabei werden die beiden Strings in eine Graphdarstellung gebracht, wie es Bild 3.4 beispielhaft zeigt.

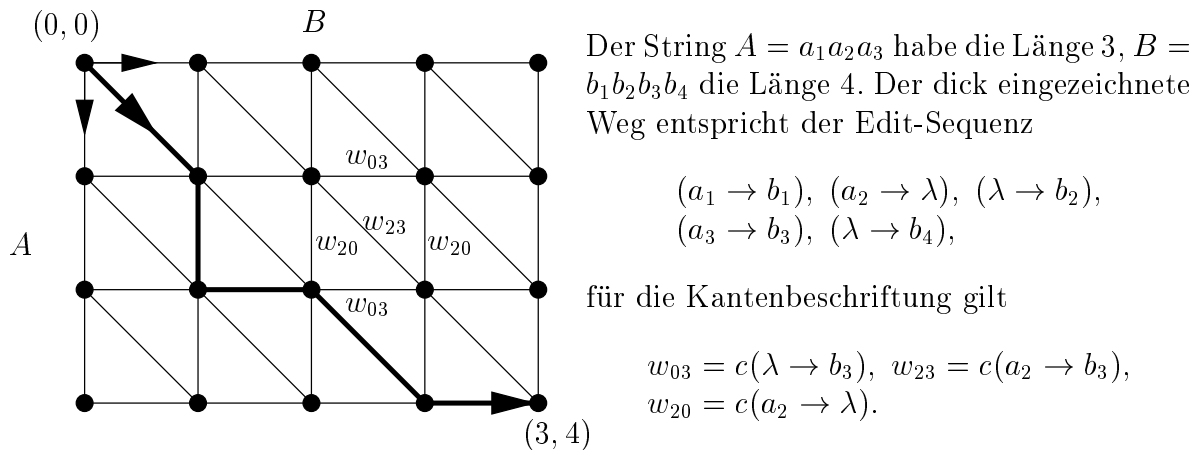


Abbildung 3.4: Darstellung zweier Strings in einem Graphen. Die Kantengewichte sind Funktionswerte der Kostenfunktionen

Jeder Knoten dieses Graphen entspricht einem Zwischenstand des Strings bei seiner Umformung von  $A$  nach  $B$ . So steht am Knoten  $(0, 0)$  der String  $A$ , am Knoten  $(3, 4)$  der String  $B$ . Jede Kante entspricht einer Edit-Operation, und zwar:

- Eine „senkrechte“ Kante  $(i-1, j) \rightarrow (i, j)$  steht für die Operation  $a_i \rightarrow \lambda$ , unabhängig von  $j$ . Es wird also ein Symbol aus  $A$  gelöscht.
- Eine „waagerechte“ Kante  $(i, j-1) \rightarrow (i, j)$  steht für die Operation  $\lambda \rightarrow b_j$ , unabhängig von  $i$ . Es wird also ein Symbol aus  $B$  hinzugefügt.
- Eine „diagonale“ Kante  $(i-1, j-1) \rightarrow (i, j)$  steht für die Operation  $a_i \rightarrow b_j$ . Es wird also ein Symbol aus  $A$  durch ein Symbol aus  $B$  ersetzt.

Andere Pfeile außer die zwischen Knoten mit benachbarten Indexpositionen gibt es nicht, und alle vorhandenen Pfeile sind nach rechts, nach unten oder diagonal nach rechts unten gerichtet.

Desweiteren ist jede Kante mit den Kosten der Edit-Operation beschriftet, der sie entspricht. Diese Beschriftung ist für einen Teil des Graphen im Bild angegeben.

Nun gilt: Jeder Weg des Graphen, der in  $(0, 0)$  startet und in  $(3, 4)$  endet, beschreibt eine gültige Edit-Sequenz, die  $A$  in  $B$  überführt. Ein solcher Weg ist zusammen mit der zugehörigen Sequenz im Bild angedeutet. Umgekehrt gibt es natürlich Edit-Sequenzen, die  $A$  in  $B$  überführen, aber nicht durch einen Weg in jenem Graphen darstellbar sind. Man überlegt sich, daß eine kostenminimale Edit-Sequenz (vgl. Definition der Edit-Distanz) mit

Sicherheit in obigem Graph repräsentiert ist, solange nur alle Kantengewichte nichtnegativ sind?<sup>2</sup>

Das Problem besteht nun also darin, in dem Graphen einen Weg von  $(0, 0)$  nach  $(3, 4)$  mit in der Summe minimalen Kantengewichten zu finden. Dies kann in der Art eines Greedy-Algorithmus' wie folgt geschehen: Bezeichnet  $D(i, j)$  die minimalen Kosten eines Weges von  $(0, 0)$  nach  $(i, j)$ , so gilt

$$D(0, 0) = 0, \quad D(i, 0) = D(i - 1, 0) + w_{i0}, \quad D(0, j) = D(0, j - 1) + w_{0j} \quad \text{für } i, j > 0$$

sowie

$$D(i, j) = \min\{D(i - 1, j) + w_{i0}, D(i, j - 1) + w_{0j}, D(i - 1, j - 1) + w_{ij}\} \quad \text{für } i, j > 0.$$

Man berechnet nun zuerst alle Werte  $D(i, j)$  in Zeit  $\mathcal{O}(m \cdot n)$ , indem man den Graphen von links oben nach rechts unten durchläuft. Anschließend enthält die Variable  $D(n, m)$  bereits die Edit-Distanz der beiden Eingabestrings. In nochmals  $\mathcal{O}(m + n)$  Zeit kann eine zugehörige Edit-Sequenz konstruiert werden. Dazu ist der Graph diesmal von rechts unten aufsteigend nach links oben zu durchlaufen. Für jede Position  $(i, j)$  bestimmt man die Vorgängerposition als eine der drei Positionen  $(i - 1, j - 1)$ ,  $(i - 1, j)$  und  $(j, i - 1)$  mit minimalem Wert  $D(\cdot, \cdot)$ . Genauer gesagt, ist die Laufzeit dieser zweiten Phase output-sensitiv  $\mathcal{O}(k)$ , wobei  $k$  die Länge der minimierenden Edit-Sequenz ist. Es gilt  $k \in \mathcal{O}(m+n)$ .

Da wir aber gemäß Definition 3.9 die Distanz zwischen *zyklischen* Strings finden müssen, ist dieser Algorithmus noch etwas zu erweitern. Es wurde bereits argumentiert, daß es genügt, die zyklischen Shifts eines der beiden Strings zu betrachten, etwa des kürzeren. Sei dazu  $m := |A| \leq |B| =: n$ . Da es nur  $m$  Repräsentanten in der Äquivalenzklasse  $[A]$  gibt, folgt sofort ein Durchprobier-Algorithmus der Laufzeit  $\mathcal{O}(m^2n)$  zur Bestimmung von  $d_{\text{edit}}([A], [B])$ . In [Maes90] wird aber noch ein besseres Verfahren vorgestellt:

**Lemma 3.10** *Für einen String  $A$  der Länge  $m$  und einen beliebigen String  $\tilde{A}$  gilt:*

$$\tilde{A} \text{ ist zyklische Rotation von } A \iff \tilde{A} \text{ ist ein Teilstring von } AA \text{ (Konkatenation)} \\ \text{der Länge } m.$$

Daher konstruiert man den Graphen aus Abbildung 3.4 für die Strings  $AA$  und  $B$  und sucht nun einen Pfad mit in der Summe minimalen Kantengewichten von einer Position  $(0, p)$  zu einer Position  $(m, p + n)$  für ein  $p \in \{0, \dots, m - 1\}$ . Dies ist in der Zeit  $\mathcal{O}(nm \cdot \log m)$  möglich; für Details zum Algorithmus sei auf [Maes90] verwiesen. Dieser Algorithmus liefert die Edit-Distanz  $d_{\text{edit}}([A], [B])$ , eine zugehörige Edit-Sequenz sowie den Wert  $p$  zurück, der angibt, um welchen Betrag  $A$  zu rotieren ist, um es mit geringsten Kosten in  $B$  zu überführen.

Abschließend noch einige Bemerkungen zu diesem Verfahren. Wendet man es auf das Überführen zweier Strings an, die Polygone wie in Definition 3.5 repräsentieren, so treten im Laufe der Transformation Strings auf, die keine Polygone darstellen, und zwar genau dann, wenn eine Lösch- oder eine Einfügeoperation ausgeführt wurde. Dadurch wird die Länge des Strings vorübergehend ungerade. Da jedoch der Zielstring stets für ein gültiges Polygon steht, hat diese Erscheinung keine weiteren Konsequenzen.

---

<sup>2</sup>In diesem Fall kann man nämlich Teilwege der Edit-Sequenz, die aus dem Graphen herausführen, streichen und erhöht damit die Gesamtkosten nicht.

In der *optimalen* Stringtransformation wird niemals ein Winkel gegen eine Seitenlänge ersetzt oder umgekehrt, denn die Kosten für eine solche Operation sind unendlich (siehe Definition 3.7)<sup>3</sup>. Prinzipiell kann man natürlich Sequenzen (mit unendlichen Kosten) angeben, die eine gültige Überführung im Sinne der Edit-Sequenzen darstellen und dabei Winkel mit Seitenlängen vergleichen.

Eine zurückgelieferte Edit-Distanz (ob kostenminimal oder nicht) kann als ein *Morphing-Verfahren* verstanden werden (vergleiche dazu Abschnitt 1.3.6). Die einzelnen Edit-Operationen geben an, welche Stücke des einen Polygons durch welche des anderen zu ersetzen sind. So entsteht im Laufe der Ersetzungen nicht nur auf der Ebene der Strings, sondern auch anschaulich das Zielpolygon aus dem anderen.

### 3.2.5 Probleme beim Einsatz der Maes-Distanzfunktion

In [Maes94] sind keine konkreten Vergleichsbeispiele angegeben, die das Verhalten der Distanzfunktion bei intuitiver Ähnlichkeit, aber verrauschten Daten etc. beschreiben. Stattdessen ist ein typisches Phänomen aufgeführt, welches leider auch in der Roboterlokalisierung bzw. allgemein beim Aufzeichnen von polygonalen Hindernissen mit einem Laserradar zu beobachten ist:

Ein Laserscanner nimmt seine Umgebung durch das Aussenden von Strahlen in regelmäßigen Winkelabständen auf. Er kann nicht schon a priori Rücksicht auf die Gestalt der Umgebung nehmen und etwa potentielle Ecken von Hindernissen erkennen. Dadurch wird es häufig zu beobachten sein, daß eine an sich glatte Hinderniskante durch zahlreiche Meßpunkte im Scan vertreten ist, obwohl ihre beiden Endpunkte ausreichen würden. Die Kante ist also in unerwünschter Weise *segmentiert*.

Ähnliches kann auch mit Winkeln auftreten. Da der Scanner normalerweise mehrere Meßpunkte in der Umgebung eines Winkel-Scheitelpunktes aufnimmt, werden dort bei einer Polygonextraktion auch mehrere Ecken entstehen – der Winkel ist in etliche „flachere“ Winkel aufgespalten.

Diese Phänomene werden in [Maes94] als *Segmentierungsphänomene* bezeichnet. Sie sind in Bild 3.5 illustriert.

Während die bisher vorgestellten Ähnlichkeitsfunktionen durch die Verwendung einer Funktionenmetrik die Polygone gewissermaßen punktweise vergleichen, geht die Maes-Distanz *stückweise* vor, d.h. sie kann immer nur eine ganze Strecke mit einer anderen Strecke in Beziehung setzen, so auch für Winkel. Das ist nun in den im Bild gezeigten Fällen sehr ungünstig. Der vorgestellte Algorithmus würde für das verrauschte und das glatte Rechteck im oberen Bild die Seiten  $a$ ,  $b$  und  $c$  unverändert übernehmen und dann die Seite  $l$  in die Seite  $l_2$  (da sie länger ist als  $l_1$ ) überführen. Die Seite  $l_1$  und der neue Innenwinkel müßten per Einfügeoperation neu aufgenommen werden. Obwohl hier also  $l_1 + l_2 \approx l$  gilt, kommen beide Längen  $l_1$  und  $l - l_2$  als Kosten zum Ausdruck, zusätzlich noch die Kosten für den neuen Winkel (die allerdings gering sind, denn er beträgt etwa  $180^\circ$ ).

Ganz ähnlich verhält es sich im unteren Bild, nur das Winkel und Seitenlängen ihre Rollen tauschen. Als Verschärfung des Phänomens wurde in diesem Beispiel der ursprüngliche Winkel  $\alpha$  in drei statt zwei Teile aufgespalten. Je kleiner die Winkelintervalle

---

<sup>3</sup>Man beachte, daß es stets eine Überführung der beiden Polygon-Strings ineinander mit endlichen Kosten gibt.

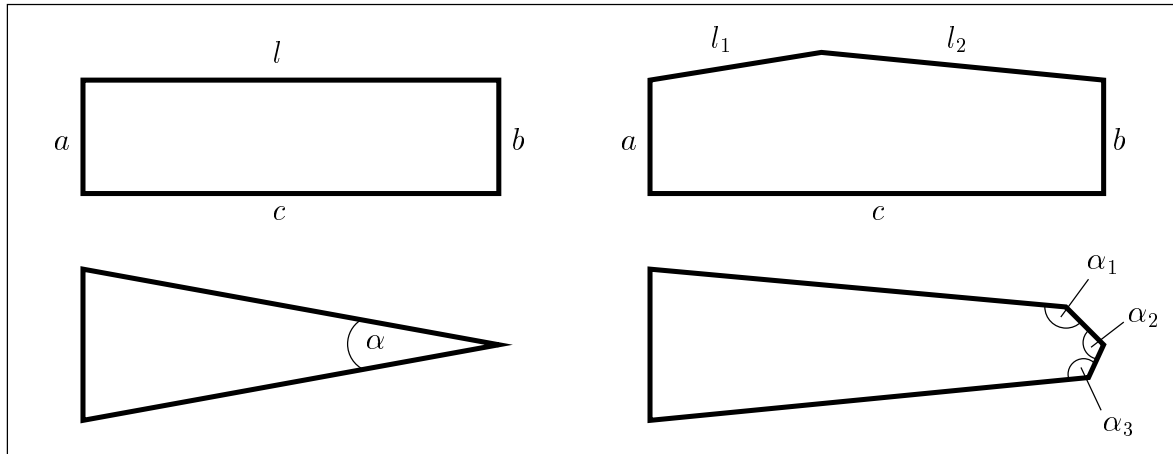


Abbildung 3.5: Segmentierungsphänomene: Aufspaltung einer Strecke (oben) und eines Winkels (unten) durch Meßfehler

sind, in denen die Laserstrahlen ausgesendet werden, desto schlimmer werden diese Segmentierungsphänomene. Paradoxerweise wird ein Anwender bei einer Verkleinerung der Intervalle und damit einer Erhöhung der Meßpunktanzahl davon ausgehen, daß seine Umgebungsextraktion mit dem Scanner und alle darauf aufbauenden Algorithmen genauer werden. Durch den abschnittswisen Polygonvergleich bei der Maes-Metrik ist genau das Gegenteil der Fall.

Eine Lösungsmöglichkeit für das Problem besteht offensichtlich darin, die starke Einschränkung aufzulockern, daß stets genau ein Stück des einen Polygons mit einem entsprechenden des anderen verglichen werden muß. Eine Auflockerung in die Richtung, daß auch *Teile* von Polygonstücken betrachtet werden dürfen (also zum Beispiel nur ein Teil der Strecke  $l$  mit  $l_2$ ), ist eine Möglichkeit. Dies hat aber den methodischen Nachteil, daß dann die fehlerhafte Segmentierung der oberen Strecke im verrauschten Polygon durch eine weitere Segmentierung, diesmal im unverrauschten, exakten Polygon, kompensiert werden soll.

Umgekehrt sollte man der Segmentierung entgegentreten, indem es gestattet wird, aufeinanderfolgende Teile eines Polygons (die ja auch in der Stringrepräsentation aufeinanderfolgen) zusammenfassend mit einem Teil des anderen zu vergleichen, also im Bild etwa die Strecken  $l_1$  und  $l_2$  und ihren eingeschlossenen Winkel mit der Strecke  $l$  im Polygon daneben. Da man auch hier beachten muß, daß nur in Bezug auf den „Datentyp“ sich entsprechende Teile betrachtet werden, könnte man allgemeiner gestatten,

- eine Teilfolge „Winkel – Seite – Winkel“ in einem Polygon mit einem Winkel des anderen oder
- eine Teilfolge „Seite – Winkel – Seite“ in einem Polygon mit einer Seite des anderen

zu vergleichen.

Damit ist die Liste der Edit-Operationen erweitert worden, und man muß sich Kostenfunktionen für die neuen Operationen ausdenken. Zum anderen ist zu bedenken, daß nun in jedem Edit-Schritt neben dem Vergleich der beiden aktuellen Polygonstücke auch ein Vergleich von Kombinationen der obigen Art in Erwägung zu ziehen ist. Da es verschiedene Kombinationen von Seite – Winkel – Seite bzw. Winkel – Seite – Winkel gibt, die in

Frage kommen, erhöht sich die Komplexität des Suchalgorithmus' aus Kapitel 3.2.4, laut [Maes94] immerhin auf  $\mathcal{O}(n^2m^3)$  statt  $\mathcal{O}(mn \cdot \log m)$ .

Genauer haben sich *Tsai* und *Yu* mit dem Thema beschäftigt, das sie als „Stringmatching mit Merging“ bezeichnen (siehe [TY85]).

### 3.2.6 Zusammenfassung

Die Maes-Distanz ist ein String-orientiertes Verfahren zum Polygon-Vergleich. Der resultierende Vergleichswert ist unabhängig von Translationen und Rotationen der Eingebepolygone, aber sensitiv gegenüber Skalierungen. Damit erfüllt es genau die Transformationsbedingungen der Roboterlokalisierung.

Die Rotationsunabhängigkeit wurde durch eine Veränderung eines Startpunktparameters erreicht. Der Startpunkt kann auf der Polygonperipherie allerdings nicht frei variieren, sondern nur Eckpunktpositionen annehmen. Dadurch leidet diese Distanz – wie die nichtapproximierten PKF- und FIF-Metriken – darunter, daß ein davon abgeleitetes Matching-Verfahren für zwei nicht kongruente Polygone im allgemeinen nicht die optimale Rotation finden wird. Zum Matchen zweier Polygone mittels Maes-Distanz müßte man zunächst *Referenzpunkte* (siehe Abschnitt 4.2.1) der Polygone bestimmen und anschließend eines so drehen, daß die durch die zyklische Rotation bestimmten korrespondierenden Startpunkte auf der Peripherie den gleichen Winkel (mit dem Referenzpunkt als Scheitel) mit einer Bezugslinie, z.B. der Horizontalen, einschließen.

Andererseits definiert aber die Maes-Distanz ein Morphing-Verfahren, da sie nicht nur ein Ähnlichkeitsmaß zurückliefert, sondern in Form einer Edit-Sequenz auch einen Vorschlag zum Überführen der beiden Polygone ineinander macht.

Die Berechnung des Vergleichswertes erfolgt über ein graphentheoretisches Verfahren zum Suchen kostengünstigster Pfade, das sehr effizient in Zeit  $\mathcal{O}(mn \cdot \log \min(m, n))$  ausgeführt werden kann. Dabei seien  $m$  und  $n$  die Eckenzahlen der beiden Polygone.

Durch sogenannte Segmentierungsphänomene kann das hier vorgestellte Verfahren zum Polygonvergleich erheblich beeinträchtigt werden. Diese lassen sich zwar durch eine algorithmische Erweiterung einschränken. Dadurch steigt aber der Rechenaufwand bei Benutzung der bisher bekannten Algorithmen erheblich auf  $\mathcal{O}(\max^2(m, n) \cdot \min^3(m, n))$ . Über das Verhalten der Maes-Distanz bei sonstigen Meßfehlern sind in [Maes94] nur wenige Aussagen gemacht worden. Allerdings sind verrauschte Kanten, wie in Bild 2.26 auf Seite 69 oder in Abbildung 2.4 auf Seite 25, im Grunde Extremfälle der Segmentierungsphänomene: Dort ist eine Seite in viele kleine aufgeteilt, von denen genau eine (sinnvollerweise die längste) mit der ursprünglichen Seite verglichen und alle anderen Seiten erst eingefügt werden müssen. Aus diesem Grund leidet auch die Maes-Distanz stark unter dem für die Arbeit mit Laserscannern typischen Rauschen.

# Kapitel 4

## Offene Probleme; Ausblick

Im letzten Kapitel vorliegender Arbeit geht es darum, wie konkret eine praxistaugliche Variante des Lokalisationsverfahrens von Guibas et al. aussehen könnte und welche Rollen die verschiedenen zur Sprache gekommenen Distanzfunktionen darin spielen. – Im zweiten Teil werden einige Gesichtspunkte aufgeführt, die in vorliegender Arbeit nicht behandelt wurden, sich aber dennoch als nützlich erweisen könnten. Neben der Frage sogenannter *Referenzpunkte* geht es dabei vor allem um spezielle Eigenheiten der Arbeit mit einem Laser-Radar und ihre eventuelle Ausnutzung.

Eine Aufzählung der in den vergangenen Kapiteln offengebliebenen Probleme beschließt diese Diplomarbeit.

### 4.1 Einsatz der Metriken in der Roboterlokalisierung

In diesem Abschnitt soll noch einmal etwas näher auf den Lokalisationsalgorithmus von Guibas et al. eingegangen und diskutiert werden, wie die Distanzen aus den vergangenen Kapiteln dabei eingesetzt werden können.

Erinnern wir uns daran, daß im Preprocessing des erwähnten Algorithmus' zu jeder Sichtbarkeitszelle ein Skelett abgespeichert wird. Dieses enthält die Informationen, die alle Sichtbarkeitspolygone gemeinsam haben, die von Standorten innerhalb der Zelle induziert werden. Die Abspeicherung eines Sichtbarkeits*polygons* ist in dem Sinne nicht evident, daß es stets an einen festen Standort gebunden ist. Ziel des Preprocessings war es aber gerade, möglichst viele Standorte in geeigneten Äquivalenzklassen zusammenzufassen.

Bei der Anfrage liefert der Laserscanner zunächst eine Menge von Zahlen (=Entfernungswerten) zurück, die sich leicht in Punktkoordinaten mit dem aktuellen Standort als Ursprung umrechnen lassen („relative Koordinaten“). Verbindet man Punkte, die bei einem zirkularen Sweep um den Standort als Mittelpunkt benachbart sind, durch ein Geradensegment, so erhält man das (bzw. eine Approximation des) Sichtbarkeitspolygon(s).

Man hat also bei der nun folgenden Suche nach einem möglichen Standort Skelette (aus dem Preprocessing) und ein Sichtbarkeitspolygon (aus der Anfrage) zur Verfügung. Es können aber höchstens Skelette mit Skeletten oder Polygone mit Polygonen verglichen werden!<sup>1</sup> Daraus ergeben sich prinzipiell zwei Ansätze:

---

<sup>1</sup>Selbst wenn man zwei Skelette vor sich hat, ist ein Vergleichsverfahren nicht unmittelbar klar; siehe dazu Abschnitt 4.1.1.

1. Erzeuge aus dem Anfrage-Sichtbarkeitspolygon ein Skelett und vergleiche dieses dann mit den Zellskeletten aus dem Preprocessing. Oder:
2. Erzeuge aus jedem Zellskelett ein Sichtbarkeitspolygon und vergleiche dieses mit dem Anfragepolygon.

Wir wollen den ersten Ansatz **skelettbasiert** und den zweiten **polygonbasiert** nennen.

### 4.1.1 Skelettbasierter Ansatz

Der skelettbasierte Ansatz entspricht genau dem Verfahren, wie es Guibas et al. für die Theorie vorgeschlagen haben. Man geht dabei davon aus, daß man in einem (Sichtbarkeits-)Polygon die Scheinkanten identifizieren kann, also die Kanten, welche kollinear mit dem Betrachterstandort liegen. Ist das gelungen, so läßt sich problemlos jede Scheinecke als diejenige Ecke einer Scheinkante bestimmen, die von  $p$  weiter entfernt liegt als die andere Ecke der Scheinkante. Nun überbrückt man schließlich Scheinecken durch neue, künstliche Kanten und gelang auf diese Weise zum Sichtbarkeits skelett. Der weitere Ablauf des Algorithmus' bereitet in der Praxis nur insofern Schwierigkeiten, als man den Skelettvergleich nur näherungsweise durchführen kann und daher Distanzfunktionen einsetzen muß. Da künstliche Kanten eine andere Bedeutung haben als echte Kanten (andere Kantentypen gibt es im Sichtbarkeits skelett nicht), muß eine polygonale Distanzfunktion daraufhin angepaßt werden, daß sie diese Kanten auch unterschiedlich behandelt.

Diese Frage ist in vorliegender Arbeit nicht behandelt worden. Die PKF- und die FIF-Metrik sind für diese Aufgabe möglicherweise ebenso ungeeignet wie die in Abschnitt 3.1 (Seite 71) vorgestellte *Arkin*-Distanz. Der Grund ist, daß sie funktionale Darstellungen der Polygone per Integralmetrik vergleichen, die keine Rücksicht auf Segmentierungen der Funktion (durch Kanten im ursprünglichen Polygon) nimmt. Hier ist eventuell die in Kapitel 3.2 auf Seite 77 eingeführte *Maes*-Ähnlichkeitsfunktion mehr von Nutzen, da sie Polygone *abschnittsweise* vergleicht. Für jeden Abschnitt, d.h. für jede Kante(nlänge) und jeden Winkel, können Gewichtungsfaktoren angegeben werden, die bestimmen, wie stark der Vergleich solcher Abschnitte mit Teilen des anderen Polygons Einfluß nehmen soll auf die Gesamtdistanz. Dort werden zum Beispiel die Kosten zum Vergleich einer Winkelgröße mit einer Streckenlänge auf  $\infty$  gesetzt. Da künstliche und echte Kanten ebenso „unvergleichbare“ Stücke eines Skeletts sind, müßte man eine Distanz zwischen solchen Segmenten ebenfalls auf  $\infty$  setzen. Eine genaue Betrachtung dieser Fragen ist ein offenes Problem für die Zukunft.

Selbst wenn wir einmal davon ausgehen, daß wir Skelette mit Skeletten vergleichen können, liegt eine große Schwierigkeit noch darin, wie die Scheinkanten zu identifizieren sind. In der Theorie benutzt man dazu das exakte Kollinearitätskriterium, was sich so in der Praxis nicht überprüfen läßt. Eine fast kollineare Kante kann bereits eine echte Kante sein. Wenn wir sie näherungsweise als kollinear und damit als Scheinkante annehmen, verändert das die Topologie der Szene vollständig, denn entlang einer Scheinkante verläuft kein Hindernis.

An diesem Mangel leidet auch die Möglichkeit, einen deutlichen Sprung der gemessenen Entfernungen zwischen zwei benachbarten Strahlwinkeln zu registrieren und daraus eine Scheinkante abzuleiten. Dies ist in Abbildung 4.1 zu sehen. Eine fast kollineare Kante erzeugt genauso einen Entfernungssprung wie eine tatsächliche Scheinkante. Daran ändert

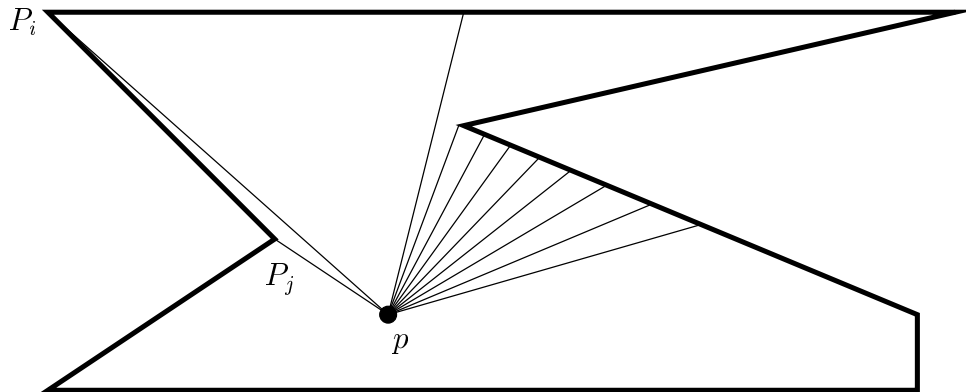


Abbildung 4.1: Ermittlung kollinearier Kanten in der Praxis

prinzipiell auch eine Verkleinerung des Winkelintervalls nichts. Das würde zwar im Bild einen weiteren Entfernungswert zwischen  $P_i$  und  $P_j$  ergeben. Da aber der Roboter beliebig nahe an der gedachten Linie durch  $P_i$  und  $P_j$  stehen kann, reicht irgendwann ein noch so kleiner Winkel nicht mehr aus, um Scheinkanten zuverlässig zu identifizieren.

Es ist daher offen, ob das von Guibas et al. für die Theorie entwickelte skelettbasierte Verfahren in der Praxis umsetzbar ist.

#### 4.1.2 Polygonbasierter Ansatz

Die Probleme der Identifizierung von Scheinkanten umgeht man, wenn man direkt das Sichtbarkeitspolygon, das aus dem Scan extrahiert wurde, verwendet. Im Algorithmus muß man es dann mit Repräsentanten von Sichtbarkeitspolygone der Zellen des Pre-processings vergleichen. Dabei geht eine wesentliche Idee des vorgestellten Algorithmus' nur noch teilweise ein: Die Zellen haben die Eigenschaft, daß sich innerhalb davon das Sichtbarkeitspolygon eines Beobachterstandorts nur wenig ändert. Dies gilt jedoch vornehmlich in qualitativer Hinsicht: Es werden bei einer Veränderung des Standorts in der Zelle tatsächlich keine neuen Szenenecken sichtbar oder verschwinden aus dem Blickfeld. *Quantitativ* kann sich jedoch einiges ändern, wie Abbildung 4.2 zeigt.

Dort ist eine Szene mit genau einer Reflexecke angegeben, so daß es nur wenige und daher recht große Sichtbarkeitszellen gibt. In einer davon sind zwei weit auseinanderliegende Roboterstandorte  $p_1$  und  $p_2$  sowie deren Sichtbarkeitspolygone markiert. Diese unterscheiden sich nur in der am weitesten links gelegenen Scheinecke (und vor allem in keiner echten Ecke). Die betreffende Zelle ist durch die Szenenkanten  $k_1$  und  $k_2$  sowie nach oben durch die eingezeichnete gestrichelte Linie begrenzt. Führt man  $p_1$  nah an diese Linie heran, so wird die nach links herausragende Spitze des Sichtbarkeitspolygons immer größer.

Wie sehr diese Veränderungen auf eine Polygondistanz wirken, hängt natürlich in starkem Maße von der Ähnlichkeitsfunktion selbst ab. Bei der PKF-Distanz zum Beispiel, die ja direkt die Entfernung der linken Spitze des Sichtbarkeitspolygons zum Standort  $p_i$  einfließen läßt, kann der Unterschied recht groß werden, denn auch der Winkelbereich, der dem Spalt am oberen Rand der Szene zugewiesen wird, ist von  $p_1$  aus größer als von  $p_2$  aus. Ähnliches gilt für die von der Flächeninhaltsfunktion abgeleitete Metrik.

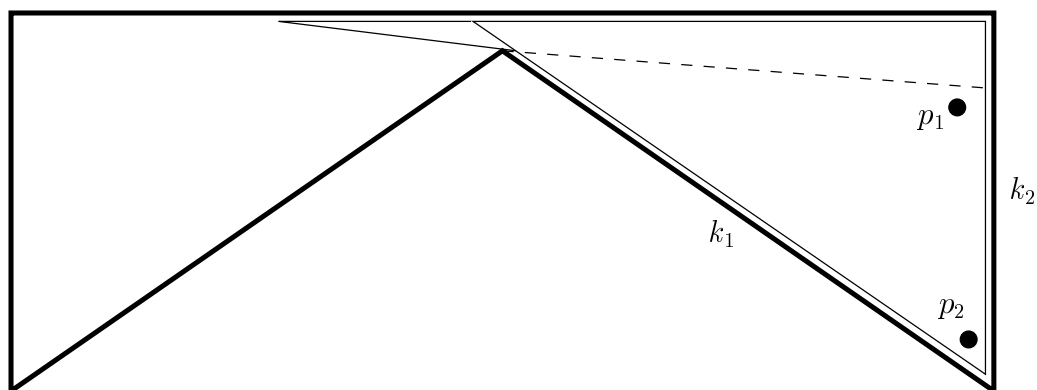


Abbildung 4.2: Sichtbarkeitspolygonveränderung bei Bewegungen innerhalb einer großen Sichtbarkeitszelle

Da die beiden Schenkel der nach links weisenden Spitze sehr lang werden können, haben sie auch einen großen Bogenlängenanteil am Umfang des Sichtbarkeitspolygons. Das bewirkt auch einen großen Einfluß der gemessenen Distanz mit der Arkin-Metrik.

Alles in allem ist es sehr schwierig, einen geeigneten Repräsentanten einer Zelle zu finden, dessen Sichtbarkeitspolygon sich sinnvollerweise mit dem Anfragepolygon vergleichen läßt. Selbst Wahlen wie der Schwerpunkt der Zelle oder des Skeletts, das zu dieser Zelle gehört, sind letztendlich willkürlich. Denn das Ziel ist es ja hierbei nicht, einen von *Rauschen* relativ unabhängigen Repräsentanten zu finden, sondern einen, der dem realen Standort möglichst nahe kommt. Diese wirkliche Position steht in keinem Zusammenhang mit dem Schwerpunkt oder einem anderen festen Bezugspunkt der Zelle. Der einzige Vorteil, hier den Schwerpunkt o.ä. als Vertreter zu verwenden, ist der, daß dann solche Extremlagen wie in Bild 4.2 nicht vorkommen können. Denn der Repräsentant wird dann relativ in der Mitte der Zelle liegen, und die maximale Distanz zwischen Anfragepunkt bei einer Einbettung und gewähltem Vertreter ist der halbe Durchmesser der Zelle. Eine Reduktion des Abstands dieser beiden Standorte läßt auch eine Verkleinerung der Distanz der Sichtbarkeitspolygone erwarten.

## 4.2 Weitere Gesichtspunkte im Umfeld der Roboterlokalisierung

Dieses Kapitel beschreibt im Überblick Themen, die entweder im Zusammenhang mit Distanzfunktionen oder im Umfeld der Roboterlokalisierung von Bedeutung sind, aber im Rahmen dieser Arbeit nicht mehr erfaßt wurden. Es stellt somit mögliche Ansatzpunkte für eine weitere Beschäftigung mit dem Themenkomplex dar.

### 4.2.1 Referenzpunkte

Wie am Ende von Abschnitt 1.3.3 auf Seite 16 erläutert wurde, hat man bei der Definition von Distanzfunktionen recht große Freiheiten, welche Zahl einem Paar von Eingabepolygonen als konkrete Distanz zugeordnet werden soll. Auferlegte Bedingungen sind neben

eventuellen Metrik-Eigenschaften auch die Unabhängigkeit gegenüber gewissen Polygontransformationen sowie ein gutes Modellieren intuitiver Ähnlichkeit.

Ein Matchingverfahren hingegen muß eine Vorschrift angeben, wie eines der Polygone zu verschieben, zu drehen oder zu skalieren (d.h. zu „matchen“) ist, damit die (statische) Distanz des transformierten Polygons zu dem anderen, unveränderten möglichst gering ist. Sind die erlaubten Transformationen sowie die verwendete Polygondistanz einmal festgelegt, so ist das optimale Matching zweier Polygone ebenfalls im wesentlichen fixiert.

Ist das theoretische Optimum des Matchens gesucht, so erhält man oft nicht nur sehr komplizierte Algorithmen, die aufwendig zu implementieren sind, sondern auch die Laufzeit einer Implementation ist zumeist recht hoch. Daher erhebt sich die Frage nach approximativen Algorithmen. Dies macht jedoch im allgemeinen nur Sinn, wenn man die dabei auftretenden Fehler kennt oder zumindest abschätzen kann.

Eine Möglichkeit dazu liefern *Referenzpunkte*:

**Definition 4.1** *Seien  $A$  und  $B$  zwei Elemente einer vorgegebenen Objektmenge. Die Punkte  $r_A$  und  $r_B$  heißen **Referenzpunkte** von  $A$  und  $B$ , wenn sie die folgende Eigenschaft besitzen:*

*Wird  $B$  derart transformiert, daß die Distanz zu  $A$  minimal ist (optimales Matching), so ist der (Euklidische) Abstand des (mit-)transformierten Punktes  $r_B$  zu  $r_A$  beschränkt durch die Distanz  $\delta := \min_{t \in \mathcal{T}} d(A, t(B))$  des Matchings multipliziert mit einem konstanten Faktor  $q$ :*

$$\exists_{q \in \mathbb{R}_{>0}} \forall_{A,B} \forall_{t \in \mathcal{T}} \quad d(A, t(B)) = \delta \quad \implies \quad d_2(r_A, t(r_B)) \leq \delta \cdot q.$$

$q$  heißt **Qualität** des Referenzpunktes.

Man beachte, daß ein Referenzpunkt einer Menge im allgemeinen nicht darin enthalten ist, wie wir auch gleich an einem Beispiel sehen werden.

Der Begriff *Referenzpunkt* hängt vor allem von zwei wichtigen Festlegungen ab:

1. von der vorgegebenen Ähnlichkeitsfunktion  $d$  und
2. von der vorgegebenen Menge erlaubter Transformationen  $\mathcal{T}$ .

Ein Beispiel eines Referenzpunktes für ein relativ einfaches Szenario zeigt folgender

**Satz 4.2 (vgl. [AG96])** *Seien  $\mathcal{T}$  die Menge aller Translationen und  $\delta_H$  die Hausdorff-Distanz<sup>2</sup> auf kompakten Teilmengen des  $\mathbb{R}^2$ . Für das dadurch definierte Punktmengen-Matching ist ein Referenzpunkt einer Menge  $A$  gegeben durch  $r_A = (x_{\min}^A, y_{\min}^A)$ , wobei  $x_{\min}^A$  bzw.  $y_{\min}^A$  die minimale in (einem Punkt in)  $A$  vorkommende  $x$ - bzw.  $y$ -Koordinate ist.  $r_A$  hat die Qualität  $\sqrt{2}$ .*

Man erhält  $r_A$  als die linke untere Ecke eines minimalen achsenparallelen Rechtecks, das  $A$  vollständig umfaßt. – Alle weiteren Sätze aus dem Kapitel beziehen sich auf diesen Referenzpunkt.

Wie kann man nun Referenzpunkte verwenden? Wir hatten eingangs festgestellt, daß es aus Effizienzgründen nützlich wäre, statt exakten Matching-Algorithmen Approximationen anzuwenden, deren Fehler abschätzbar ist. Dies ist mit Referenzpunkten möglich, indem man einfach die beiden Referenzpunkte der Objekte  $A$  und  $B$  matcht und nicht die Objekte selbst. Der auftretende Fehler ist tatsächlich relativ gering:

<sup>2</sup>Eine genaue Definition dieser Distanz findet sich im Anhang.

**Satz 4.3 (vgl. [AG96])** *Es liege die Situation aus Satz 4.2 vor. Seien speziell  $A$  und  $B$  zwei Punktmengen mit Referenzpunkten  $r_A$  und  $r_B$  der Qualität  $q$ .  $B_{opt}$  sei das optimale Matching-Resultat von  $A$  und  $B$ , d.h. es gelte*

$$\delta_H(A, B_{opt}) = \min_{t \in \mathcal{T}} \delta_H(A, t(B)) =: \delta.$$

*Schließlich sei  $\tilde{B} = B + \overline{r_B r_A}^\lambda$ , d.h.  $\tilde{B}$  erhält man aus  $B$  durch Verschieben entlang  $\overline{r_B r_A}^\lambda$ . Dann gilt:*

$$\delta_H(B_{opt}, \tilde{B}) \leq q \cdot \delta.$$

**Korollar 4.4** *Für die Qualität des näherungsweise Matchens von  $A$  und  $B$  vermöge  $\tilde{B}$  gilt*

$$\begin{aligned} \delta_H(A, \tilde{B}) &\leq \delta_H(A, B_{opt}) + \delta_H(B_{opt}, \tilde{B}) \\ &\leq \delta + q \cdot \delta = (q + 1) \cdot \delta. \end{aligned}$$

Speziell für die Hausdorffdistanz und Translationen als zugelassene Transformationen sowie den speziellen Referenzpunkt aus Satz 4.2 ist also das approximative Matchen höchstens um den Faktor  $\sqrt{2} + 1 \approx 2.4$  schlechter als das optimale. Vor allem ist es jedoch mit geringem Zeitaufwand realisierbar, denn sowohl das Bestimmen der Referenzpunkte als auch das Konstruieren von  $\tilde{B}$  aus  $B$  sind in *linearer Zeit* möglich.

Auch wenn eine Ähnlichkeitsfunktion nicht die Aufgabe hat, den Distanzwert zweier Objekte nach einem optimalen Matching zurückzuliefern, dient die Vorstellung des Matchens doch oft als Ansatzpunkt für die Definition eines Distanzmaßes. In der Anwendung für die Roboterlokalisierung war es das Ziel, *rotationsunabhängige* Ähnlichkeitsfunktionen zu finden. Die Rotation haben wir bei den PKF- und FIF-Metriken durch die Minimierung über den Winkelparameter  $t$  berücksichtigt.

Möchte man aus diesen Distanzen ein Matching-Verfahren ableiten, so bestimmen die speziellen Kernpunkte  $p_A$  und  $p_B$  den translatorischen Anteil der Überführung der Polygone ineinander (zu den Einzelheiten siehe den Abschluß der Zusammenfassung auf Seite 39). Wie oben begründet wurde, ist es dann wünschenswert, daß  $p_A$  und  $p_B$  Referenzpunkte sind, und zwar bezüglich Translation *und* Rotation. Die in Kapitel 2.1.3.1 vorgeschlagene Wahl der Punkte erfüllt diese Bedingung nicht, wie folgendes Bild zeigt:

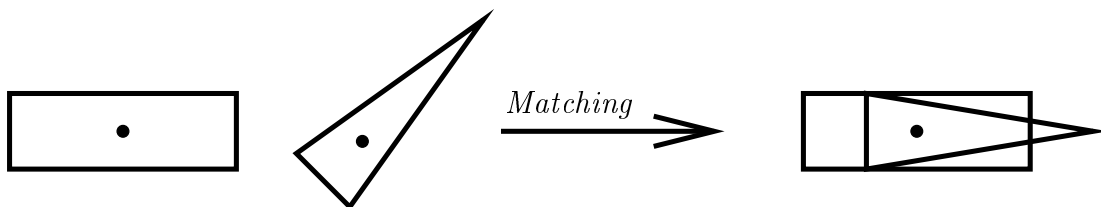


Abbildung 4.3: Der Schwerpunkt ist kein Referenzpunkt für Polygon-Matching

Verschiebt man hier die beiden speziellen Kernpunkte ineinander, so erhält man offenbar ein sehr schlechtes Matching.

In der Tat kann man zeigen, daß weder der in Satz 4.2 vorgeschlagene Punkt noch der Schwerpunkt die Referenzpunkt-Eigenschaft für beliebige Euklidische Transformationen erfüllen. In [AG96] wird der sogenannte Steinerpunkt einer beschränkten Menge  $A$  als Referenzpunkt erwähnt. Er ist sogar für beliebige *Ähnlichkeitstransformationen* in *beliebigen Dimensionen* gültig und hat im  $\mathbb{R}^2$  die Qualität  $4/\pi \approx 1.27$ .

H. Alt et al. zeigen in [AFRW96] einen Referenzpunkt für konvexe Polygone (bzw. allgemein konvexe Mengen) mit der symmetrischen Differenz als Distanzfunktion. In der Roboterlokalisierung ist aber die Konvexität (des Sichtbarkeitspolygons) nach Lemma 1.7 (Seite 6) sogar nachweislich normalerweise nicht gegeben, dafür die Sternförmigkeit (Lemma 1.8). Zu der Frage nach Referenzpunkten für sternförmige Polygone ist offenbar noch wenig bekannt. Ergebnisse in dieser Richtung könnten die Qualität der oben genannten Metriken im Hinblick auf Erweiterung zu einem Matchingverfahren verbessern und eventuell Abschätzungen der Approximationsfehler ermöglichen.

### 4.2.2 Reichweitenbeschränkung, Winkelinkrement, Verdeckung

Bei der Definition sowohl der PKF- wie auch der FIF-Metrik haben wir insofern von dem Spezialfall der Polygonaufzeichnung durch einen Laser-Radar Gebrauch gemacht, als wir ausgenutzt haben, daß diese Polygone sternförmig sind. Die Art, wie der Laserscanner seine Daten aufnimmt, hat zu der Idee der Definition der Distanzen geführt. Dabei gibt es jedoch noch weitere Einzelheiten bzw. Besonderheiten, die bisher noch nicht in die Arbeit eingeflossen sind. Sie sollen im folgenden in aller Kürze angesprochen werden.

**Reichweitenbeschränkung.** Ein Laserscanner besitzt eine begrenzte Reichweite, die zum Beispiel in der Größenordnung von 50m liegen kann. Damit können alle Räume vermessen werden, deren *Diameter*<sup>3</sup> maximal 50m beträgt.

In größeren Räumen ist es möglich (vor allem, wenn der Roboter, auf dem der Laser montiert sein könnte, nah an einer Wand steht), daß für viele Richtungen des Laserstrahls keine Entfernung zurückgeliefert wird. Damit kann das Sichtbarkeitspolygon eines entsprechenden Standorts nicht vollständig konstruiert werden.

Die Anwendung einer Metrik, die auf der Integraldistanz für stetige Funktionen beruht, ist in diesem Falle schwierig, denn das Integral kann für den fehlenden Winkelabschnitt nicht bestimmt werden. Bei der Maes-Metrik könnte man zumindest den Winkelabschnitt einfach weglassen und erhielte einen verkürzten String als Darstellung des unvollständigen Polygons. Damit ist rein syntaktisch das String-Matching mit der Maes-Distanz durchführbar.

Diese Vorgehensweise ist jedoch unbefriedigend, denn man weiß zumindest, daß das Polygon in den Richtungen, für die kein Entfernungswert zurückgeliefert wurde, einen Peripheriepunktabstand von mindestens  $r$  hat, wenn  $r$  die (technisch vorgegebene) Reichweite des Scanners ist. Also könnte man die Polygone in diesen Winkelabschnitten durch linear approximierten Kreisbögen mit Radius  $r$  vervollständigen. Dies hat aber den Nachteil, daß für zwei unvollständige Polygone dann suggeriert wird, sie würden in diesen

---

<sup>3</sup>Der Diameter einer beschränkten Fläche ist die Länge der längsten Sehne, die man vollständig in die Fläche legen kann. Für einen Raum sei der Diameter hier der der Grundfläche, da wir zumeist von prismenförmigen Räumen ausgehen.

Winkelabschnitten übereinstimmen. In Wirklichkeit weiß man jedoch nichts über diese Bereiche.

Wie solche unbekanntem Abschnitte in eine Metrikberechnung einbezogen werden sollen, bleibt noch zu untersuchen.

**Verdeckungen (Occlusion).** Objekte einer Szene könne teilweise verdeckt sein, entweder durch dynamische (bewegte, d.h. unvorhergesehene) Hindernisse oder durch andere statische Objekte. Der Fall statischer Hindernisse stellt kein Problem dar, solange alle unbeweglichen Objekte in der Karte, die der Roboter zur Verfügung hat, eingetragen sind. Denn die entstehenden Sichtbarkeitspolygone von Standorten sind dann in jedem Fall in der Karte wiederzuerkennen.

Schwieriger ist es mit bewegten Hindernissen. Diese können entweder gar nicht in der Karte verzeichnet sein (z.B. Personen) oder an einem anderen Ort, als es im Moment der Aufnahme des Laserscans der Fall ist. Daraus erwachsen zwei Probleme:

1. Es muß überhaupt *erkannt* werden, daß es sich um ein dynamisches, d.h. nichtverzeichnetes Hindernis handelt.
2. Ist dies gelungen, so ist entweder der Winkelbereich, in dem das Hindernis liegt, in geeigneter Weise auszufüllen oder wegzulassen, da in jedem Fall das, was sichtbar ist (das dynamische Hindernis), nicht in der Karte wiedererkannt werden kann.

Für den zweiten Punkt gelten ähnliche Kriterien wie für die Problematik der Reichweitenbeschränkung. Zum ersten Punkt kann man als Entscheidungsheuristik folgende Überlegung zu Hilfe nehmen: In Bild 4.4 ist links ein Sichtbarkeitspolygon angegeben, das möglicherweise auf eine lange Wand im Hintergrund hinweist.

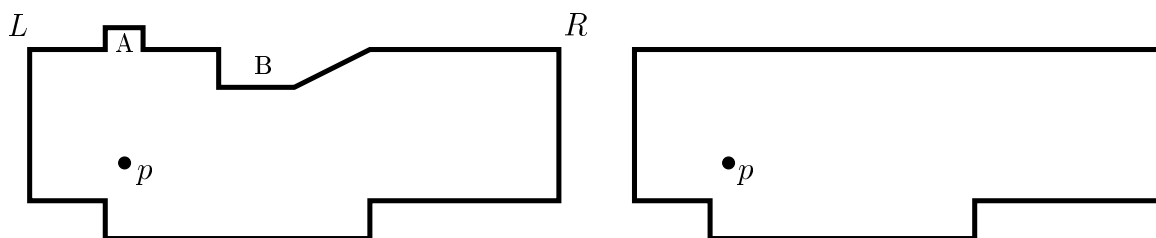


Abbildung 4.4: Sichtbarkeitspolygone mit verschiedenartigen Ausbuchtungen

Diese Wand ist jedoch von zwei Ausbuchtungen unterbrochen, die in verschiedene Richtungen zeigen. Bei A handelt es sich sicher nicht um ein Hindernis, sondern um einen Einlaß in der Wand o.ä.<sup>4</sup>, der nicht beweglich ist. Daher wird der Teil A der Szene in der Karte verzeichnet sein.

Die Ausbuchtung B hingegen ist keine nach innen gerichtete bauliche Veränderung der Wand, sondern höchstwahrscheinlich ein Hindernis. Natürlich sagt dies nichts darüber aus, ob es sich um ein *dynamisches* Hindernis handelt, aber im Gegensatz zu A besteht hier immerhin die Möglichkeit dazu.

<sup>4</sup>A könnte auch eine offenstehende Tür sein. Dann wäre es ebenfalls als ein dynamisches Hindernis zu werten und erschwert das Wiedererkennen der Szene in der Karte. Man kann Türen eventuell an einer spezifischen Breite der Wandöffnung erkennen.

Im Vergleichspolygon rechts gibt es diese Ausbuchtungen nicht. Daher ist die (wie auch immer gemessene) Distanz der Polygone auf  $A$  und  $B$  zurückzuführen. Dabei könnte man den durch  $A$  verursachten Distanzanteil stärker gewichten, denn er geht mit einiger Sicherheit nicht auf ein dynamisches Hindernis zurück.

Der geometrische Unterschied zwischen  $A$  und  $B$  ist, daß die Punkte des Sichtbarkeitspolygons, die zu  $A$  gehören, weiter entfernt sind als die gedachte Linie von  $L$  nach  $R$ . Beim Hindernis  $B$  liegt der umgekehrte Fall vor. Eine Heuristik könnte also lauten, alle „Unregelmäßigkeiten“ mit größerer Entfernung als an der entsprechenden Stelle in der Karte stärker einfließen zu lassen als solche mit geringerer Entfernung, da letztere unter Umständen nur auf ein nichtverzeichnetes, temporäres Hindernis zurückgehen.

Diese Vorgehensweise muß noch untersucht werden. Zum Beispiel hängt die Gewichtung auch davon ab, wie wahrscheinlich das Auftreten von dynamischen Hindernissen ist. Ist die Wahrscheinlichkeit gering, so wachsen die Chancen dafür, daß es sich auch bei  $B$  nur um ein statisches Hindernis handelt, das dann genauso zu behandeln ist wie  $A$ .

Zum Verhalten der Arkin-Metrik beim Auftreten von Verdeckungen siehe [ACH<sup>+</sup>91, Seite 214].

**Winkelinkrement.** Beim Aufzeichnen seiner Umgebung sendet ein Laserscanner seine Strahlen stets in festen Winkelinkrementen aus, abhängig von der gewünschten Auflösung in der Größenordnung von wenigen Grad oder Bruchteilen eines Grades. Aus den Auftreffpunkten generiert man später das Sichtbarkeitspolygon, z.B. einfach durch eine Verbindung der angular benachbarten Punkte. Diese Nachbarpunkte haben also alle einen festen und bekannten Winkelabstand voneinander.

Inwieweit diese geometrische Tatsache nutzbar ist, wurde noch nicht untersucht. Wir haben bisher nur an einer Stelle von dieser Gegebenheit profitiert, als es um die Auflösung der Mehrdeutigkeiten der  $FIF_{lin}$  ging (siehe Seite 66). Überhaupt nicht berücksichtigt wurde sie bei der Definition der Metriken. Hier besteht wieder das Problem, daß mit Ausnahme der Maes-Distanz alle Ähnlichkeitsfunktionen, die in dieser Arbeit behandelt wurden, auf der Integral-Metrik für stetige Funktionen beruhen. Dabei handelt es sich um ein kontinuierliches Maß, das Segmentierungen der Eingabe ignoriert. Ein Ausnutzen der konstanten Winkelabstände müßte also früher ansetzen, etwa bei der Gewinnung der Polygondarstellung.

## 4.3 Einzelheiten

Folgende Einzelheiten waren in den vergangenen Kapiteln noch offengeblieben:

### Zu PKF und FIF.

1. Die lineare Approximation der PKF bzw. der FIF hat zu der Möglichkeit geführt, die Integraldistanz in der Definition der jeweiligen Metrik exakt auszurechnen (siehe Algorithmus 1 auf Seite 35). Ist dies auch *ohne* eine Approximation exakt möglich? D.h., gibt es ein Verfahren, das für die PKF- bzw. die FIF-Kurven das Minimum in der Metrik-Definition berechnet und dabei nicht nur die  $m \cdot n$  vielen „kritischen“ Stellen heranzieht, an denen zwei Übergangsstellen der Graphen zusammenfallen? Wie aufwendig ist ein solches Verfahren?

2. In den Kapiteln über die linearen Approximationen von PKF und FIF wurden die Fälle aufgezeigt, in denen die Approximationsfehler am größten sind und daß sie gegebenenfalls sogar beliebig groß werden können (für die PKF siehe etwa Absatz „Fehlerbetrachtung für die Approximation“ auf Seite 31). Kann man außer für diese Extremfälle noch weitere Abschätzungen für den allgemeinen Approximationsfehler durchführen? Lassen sich Vorhersagen treffen, unter welchen Umständen das Minimum in der Metrik-Definition *nicht* an einem kritischen Punkt (siehe oben) angenommen wird?
3. Eine Vorbereitung zum Einsatz von PKF- und FIF-Metrik besteht zur Zeit noch darin, den speziellen Kernpunkt aus Abschnitt 2.1.3.1 (Seite 23) zu bestimmen (eventuell im Preprocessing für jedes Polygon). Es wurde diskutiert, daß der Schwerpunkt des Polygons ein besserer und ein Referenzpunkt (Abschnitt 4.2.1) ein noch besserer Punkt für diese Rolle wären. Diese Punkte sind aber im allgemeinen keine Kernpunkte mehr. Wie lassen sich daher die PKF und die FIF auf Darstellungen nicht sternförmiger Polygone erweitern?

Ist  $p$  kein Kernpunkt, so stimmt die Umlaufordnung der Ecken im Polygon nicht mehr mit der angularen Sortierung der Ecken bezüglich  $p$  als Zentrum überein. Sortiert man die Ecken daher neu um  $p$  (was den Aufwand  $\mathcal{O}(n \cdot \log n)$  wie für die Kernbestimmung zur Folge hat), so ignoriert man die Tatsache, daß eine Strecke  $\overline{P_i P_j}$  eines Sektors  $\Delta p P_i P_j$  unter Umständen keine Polygonkante ist (vergleiche dazu folgende Abbildung).

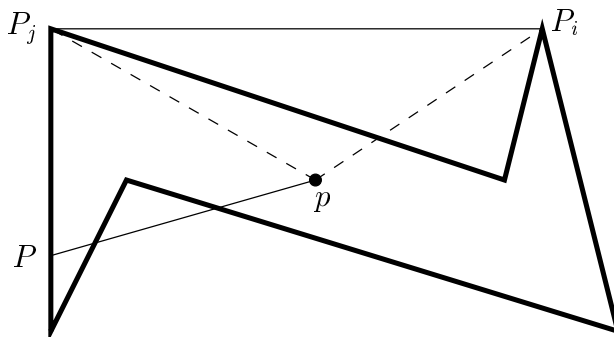


Abbildung 4.5: PKF und FIF für ein nicht sternförmiges Polygon

In jedem Fall befinden sich die Sektorendreiecke, deren Flächeninhalte bei der FIF zu bestimmen sind, teilweise außerhalb des Polygons. Es ist möglicherweise sinnvoll, die außerhalb liegenden Teile mit einem negativen Vorzeichen zu versehen.

Für die PKF ist es schwieriger, einen Lösungsansatz zu finden. Um einem Winkel, in dem ein Strahl ausgesendet wird, eine Entfernung zuzuordnen, muß dieser Strahl einen eindeutigen Peripheriepunkt bezeichnen. Dies ist nicht mehr gegeben, wenn  $p$  kein Kernpunkt ist. Zum Beispiel schneidet der Strahl  $p \rightarrow P$  in obiger Abbildung die Peripherie dreimal. Es ist offen, ob man hier den Schnittpunkt des Strahls mit der Peripherie benutzen sollte, der minimale oder maximale oder mittlere Entfernung

zu  $p$  hat, oder ob in einem solchen Fall gar kein Schnittpunkt mit der Peripherie als Bezugspunkt für die Entfernungsmessung sinnvoll ist.

Es sei allerdings daran erinnert, daß die Motivation des Punktes 3 nicht in dem Mangel an einem Kernpunkt besteht. Dieser ist in der Lokalisation mittels Laser-Radar stets in Form des Standorts gegeben. Es geht vielmehr darum, die Unzulänglichkeiten eines Kernpunktes zu beseitigen, indem zu einem passenderen Punkt, bestenfalls einem Referenzpunkt, übergegangen wird, der im allgemeinen die Kernpunkteigenschaft aber nicht mehr erfüllt.

#### Zur FIF.

4. In Abschnitt 2.3.2.3 wurde gezeigt, daß es Polygone gibt, die sich aus der  $FIF_{\text{lin}}$ -Darstellung nicht eindeutig rekonstruieren lassen und wie man in der Praxis dieses Problem umgeht (aber nicht beseitigt). Es bleibt die Frage, für welche Polygone genau dieses Phänomen auftritt. Die beiden Beispiele in Abbildung 2.22 auf Seite 65 erscheinen im intuitiven (nicht im formal-geometrischen) Sinne regelmäßig. Insbesondere sind sie rotationssymmetrisch.
5. Da wir in der Definition der Flächeninhaltsmetrik aus mehreren Gründen gezwungen waren, durch den Gesamtflächeninhalt zu dividieren und so die Darstellung von Skalierungen unabhängig war, wurde mit dem Lemma (2.7) (Seite 60) eine prinzipielle Möglichkeit zur Beseitigung dieses unerwünschten Effekts angegeben. Es blieben aber noch die Fragen offen, ob der Ansatz in der Praxis sinnvoll ist und wie in diesem Fall der Gewichtungsfaktor  $\omega$  zu wählen ist. Das Hauptproblem wird sein, daß durch die globale Vorgabe von  $\omega$  meist doch einer der beiden Summanden der neuen Metrik  $s'$  oder  $s''$  (siehe erwähntes Lemma), d.h. die alte Metrik  $s$  oder der Korrektursummand, größeren Einfluß hat. Wären die beiden Anteile als *Faktoren* eingegangen, d.h.  $s'$  als Produkt definiert worden, so hätte man eine gleichmäßige Beeinflussung des Gesamtausdrucks, allerdings wäre dann  $s'$  keine Metrik mehr (Dreiecksungleichung).

#### Zum praktischen Einsatz des Lokalisationsalgorithmus<sup>5</sup>.

6. In Kapitel 1.2.3 auf Seite 9 wurde davon gesprochen, daß im Preprocessing zum Lokalisationsalgorithmus von Guibas et al. viele Sichtbarkeitszellen entstehen können, deren Skelette sich unter Umständen nur wenig unterscheiden<sup>5</sup>, z.B. wenn die Szene aus vielen Hindernissen mit verhältnismäßig vielen Reflexecken besteht. Wenn solche Zellen auch noch benachbart sind, dann kann es sinnvoll sein, sie zu einer größeren zu verschmelzen. Solche Vereinfachungsoperationen im Sinne von Abschnitt 1.3.5 stellen allerdings immer auch eine *topologische* Veränderung an der Karte dar, die den theoretischen Ansatz zur Lösung des Lokalisationsproblems sofort zum Scheitern bringen würde. Deshalb ist es in diesem Fall sogar *notwendig*, Distanzfunktionen

---

<sup>5</sup>Man beachte hier, daß eine geringe räumliche Ausdehnung einer Sichtbarkeitszelle kein Kriterium dafür ist, daß Skelette benachbarter Zellen sehr ähnlich sind: Das Überschreiten einer Sichtlinie zwischen zwei Zellen bewirkt zunächst, daß eine neue Ecke sichtbar wird. Diese kann weit entfernt von den bisher sichtbaren Ecken liegen, so daß das neue Skelett sich quantitativ stark von dem vorherigen unterscheiden kann.

einzusetzen, die gegenüber (geringfügigen) Kartenvereinfachungen weniger sensibel sind als der theoretische Algorithmus von Guibas et al. Das genaue Vorgehen hierbei ist ebenfalls eine weitere offene Frage.

Viele der angesprochenen Punkte beschreiben Probleme in der Praxis, die durch Unzulänglichkeiten der eingesetzten Technik zustande gekommen sind, besonders aufgrund von Meßungenauigkeiten. Diese werden sich auch in Zukunft nie vollständig vermeiden lassen. Es ist zwar (unter Erhöhung der Laufzeit eines Algorithmus') möglich, Ungenauigkeiten der Rechnerplattform in ihrem Einfluß zurückzudrängen, indem man zum Beispiel auf exakte Arithmetik ausweicht. Das Entscheidende wird aber immer die Schnittstelle zwischen Computer und Umwelt sein, die nicht ohne einen Verlust an Präzision zu passieren ist.

Leider verhalten sich die Ungenauigkeiten nicht *stetig* in dem Sinne, daß leicht veräuschte Eingabedaten stets auch nur ein leicht abweichendes Endergebnis (zum Beispiel einer Lokalisationsanfrage) zur Folge hätten. Wir haben gesehen, daß das Verfahren von Guibas in der theoretischen Version für die Praxis nicht sofort anwendbar ist, da rein algorithmisch viele der Schritte nur bei exakten Berechnungen möglich bzw. sinnvoll sind. Dies gilt zum Beispiel für die Entscheidung darüber, ob eine Kante eines Sichtbarkeitspolygons eine Scheinkante ist, dadurch auch für die Extraktion des Sichtbarkeits skeletts aus einem Scan etc.

Rechenungenauigkeiten lassen sich also nicht allein dadurch in ihrem Einfluß reduzieren, daß man überall dort, wo in einem Algorithmus Vergleiche anstehen, Distanzfunktionen einsetzt. Vielmehr sind in den meisten Fällen auch algorithmische Veränderungen notwendig, wie wir es zum Beispiel in Abschnitt 4.1.2 über den polygonbasierten Ansatz bei einer Lokalisationsanfrage gesehen haben.

# Anhang A

## Testbeispiele

Dieser Teil des Anhangs stellt einige Testbeispiele vor, die mit den in Kapitel 2 vorgestellten Ähnlichkeitsfunktionen gewonnen wurden. Sie dienen einer graphischen Veranschaulichung der hergeleiteten bzw. prognostizierten Verhaltensweise der Metriken bei bestimmten Arten von Rauschen etc. Für eine genauere Analyse siehe in den jeweiligen Kapiteln.

Die Beispiele (zur PKF- und zur FIF-Metrik) sind wie folgt aufgebaut: Zunächst werden einige Paare von Polygonen direkt miteinander verglichen. Sie zeichnen sich durch spezielle Eigenschaften (vor allem gewisse Fälle von Rauschen) aus, die jeweils diskutiert werden.

Im zweiten Teil wird ein Musterpolygon vorgegeben und mit verschiedenen mehr oder weniger ähnlichen Polygonen verglichen. Die Ergebnisse sind in der Reihenfolge ihrer Ähnlichkeit zu dem Muster aufgeführt. Daraus kann man ablesen, was für Polygone die jeweilige Distanz als ähnlich ansieht.

In den Beispielen aus Abschnitt A.1 ist jeweils der spezielle Kernpunkt angegeben. Da er als kernnächster Punkt gewählt wird, ist er oft kollinear mit einer der Polygonkanten.

### A.1 Beispiele für Spezialfälle

Die Kopfzeile folgender Tabelle enthält die Bezeichnungen der Funktionsdarstellungen, auf denen die jeweilige Metrik beruht. Vor einer Betrachtung der Zahlenwerte empfiehlt sich ein Blick auf die zugehörigen Abbildungen, da dort auf die Besonderheiten des aktuellen Polygonpaares hingewiesen wird. Der Vergleich der PKF-Distanzwerte mit den absoluten FIF-Werten macht wenig Sinn, da die Funktionen verschiedene Wertebereiche haben. Man kann aber relative Vergleiche anstellen.

Für die Polygone aus Abbildung A.1 („Rotation“) wurden auch die Varianten der PKF-Metrik ausprobiert, die nur Translationen als Polygontransformationen zulassen. Es ergab sich ein Distanzwert von 19.5433 für die (nichtapproximierte) PKF-Distanz und von 21.7838 für die  $\text{PKF}_{\text{lin}}$ -Distanz.

Abbildung	PKF	PKF <sub>lin</sub>	FIF	FIF <sub>lin</sub>
A.1 („Rotation“)	0.0217819	0.021772	0.0186525	0.000314677
A.2 („Beispielszene“)	31.9157	33.5932	3.718	5.9617
A.3 („skaliert“)	50.5964	53.5509	0	0
A.4 („kollinear“)	0	3.17972	0.0221669	2.99818
A.5 („Kerbe“)	0.491679	2.05269	3.90533	1.34039
A.6 („leichtes Rauschen“)	3.63465	1.9760	3.62581	2.73868
A.7 („uneinheitliches Rauschen“)	4.92681	4.20746	4.7174	3.14649
A.8 („einheitliches Rauschen“)	7.74074	4.50825	8.02348	3.94091

Tabelle A.1: Vergleichswerte für die Polygonpaare aus den folgenden Abbildungen

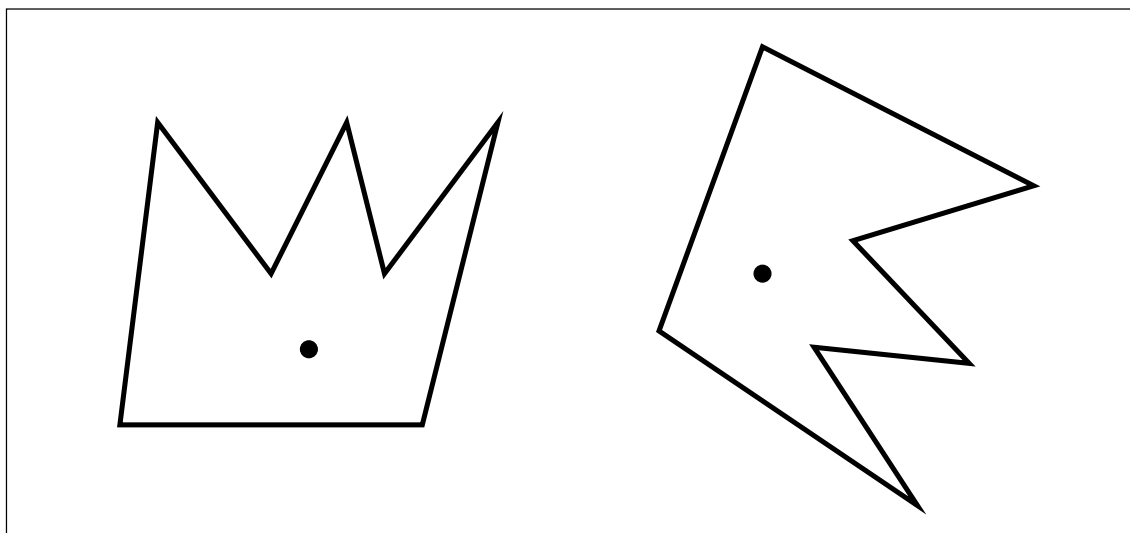


Abbildung A.1: („Rotation“) Zwei Polygone, die sich nur durch eine Rotation unterscheiden. Alle vier Distanzwerte sind nahezu 0

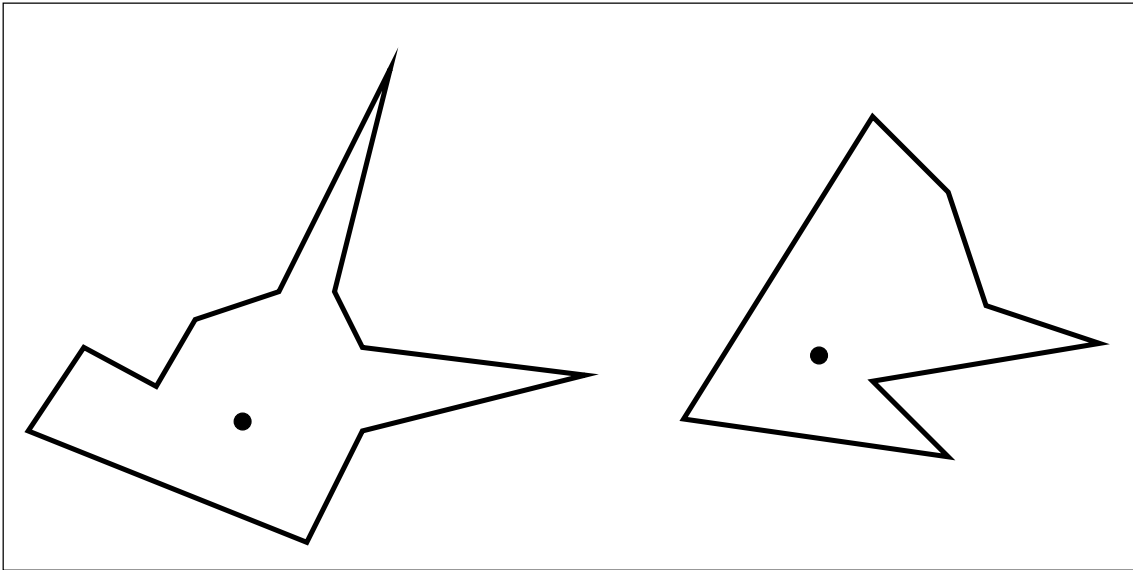


Abbildung A.2: („Beispielszene“) Ein Polygonpaar ohne spezielle Eigenschaften. Die FIF-Distanzwerte sind allgemein kleiner, da sie den Flächeninhalt der Polygone auf 1 skalieren

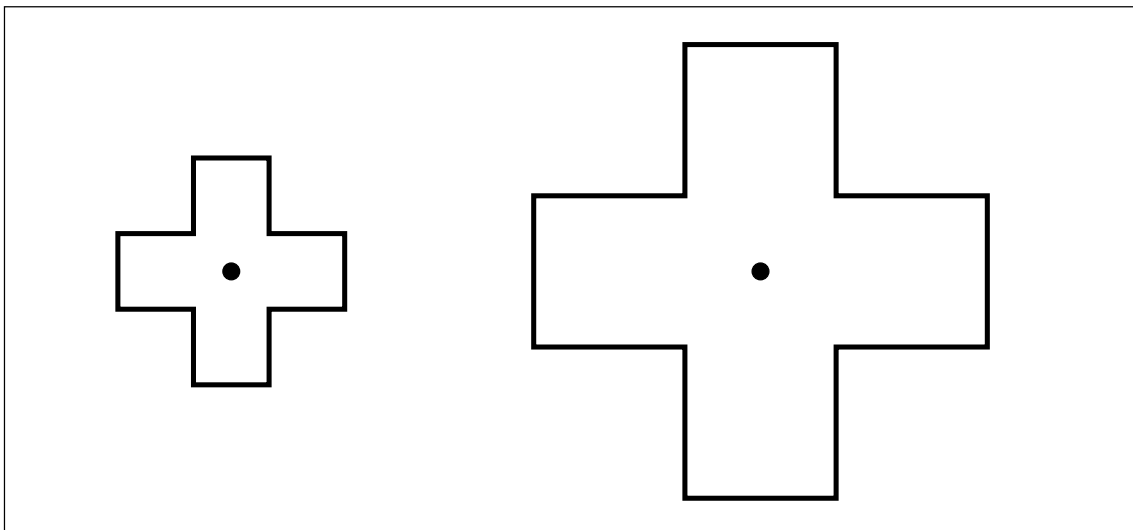


Abbildung A.3: („skaliert“) Zwei Polygone, die sich nur durch eine Skalierung unterscheiden. Da die FIF-Metriken in der vorgestellten Form skalierungsinvariant sind, liefern sie für diese Polygone die Distanz 0 zurück – ein unerwünschter Effekt, der bei den PKF-Metriken nicht auftritt

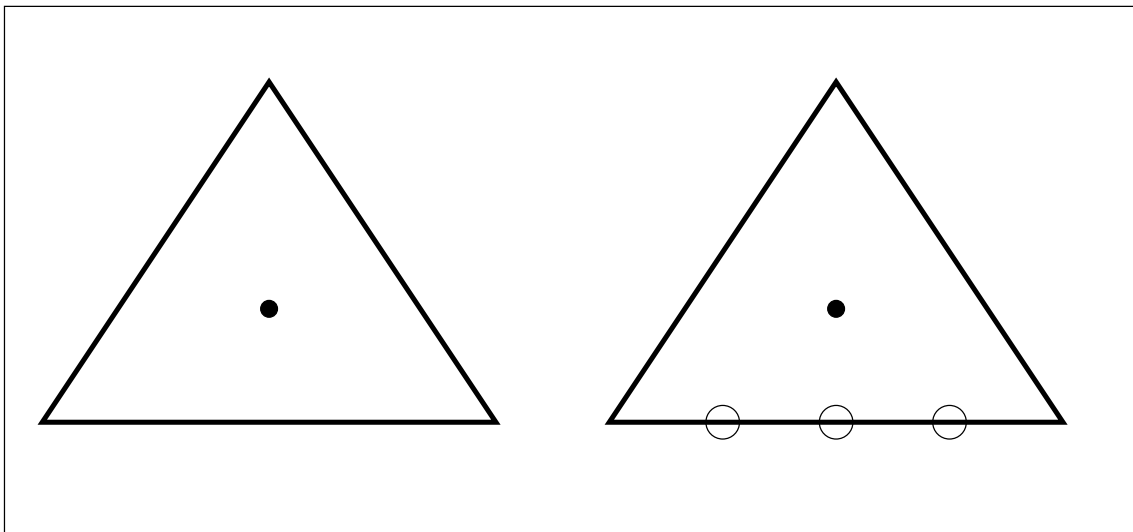


Abbildung A.4: („kollinear“) Das rechte Dreieck wurde als „Sechseck“ mit fünf kollinearen Punkten auf der Unterseite eingegeben (die Kreise stellen redundante Punkte dar). Die  $\text{PKF}_{\text{lin}}$  und die  $\text{FIF}_{\text{lin}}$  haben mit diesen Polygonen Schwierigkeiten: Die zusätzlichen Eckpunkte bedeuten zusätzliche Stützstellen bei der Approximation und daher einen anderen Verlauf der linear approximierten Kurve als im linken Dreieck. Dadurch ist der Wert bei diesen beiden Distanzen von Null verschieden. Die nichtapproximierten Ähnlichkeitsfunktionen liefern wie gewünscht 0

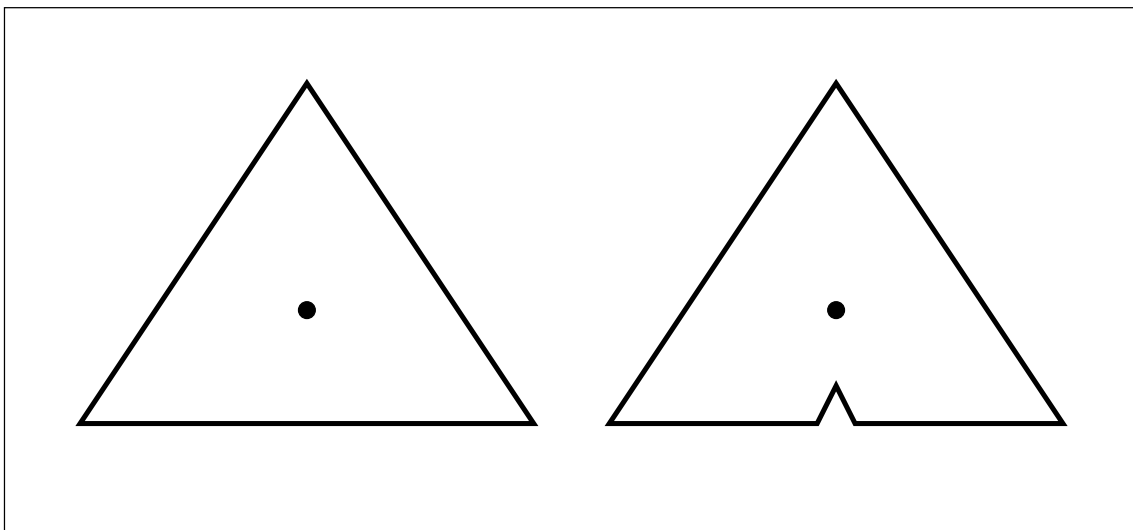


Abbildung A.5: („Kerbe“) Das rechte Polygon besitzt an der Unterseite eine Kerbe, die durch einen (groben) Messfehler entstanden sein könnte. Dies verändert zwar nicht den speziellen Kernpunkt (der Schwerpunkt liegt in beiden Polygonen im Kern), aber doch die Kurven der Polygodarstellungen. Dadurch ergibt sich in allen Fällen ein von Null leicht abweichender Wert

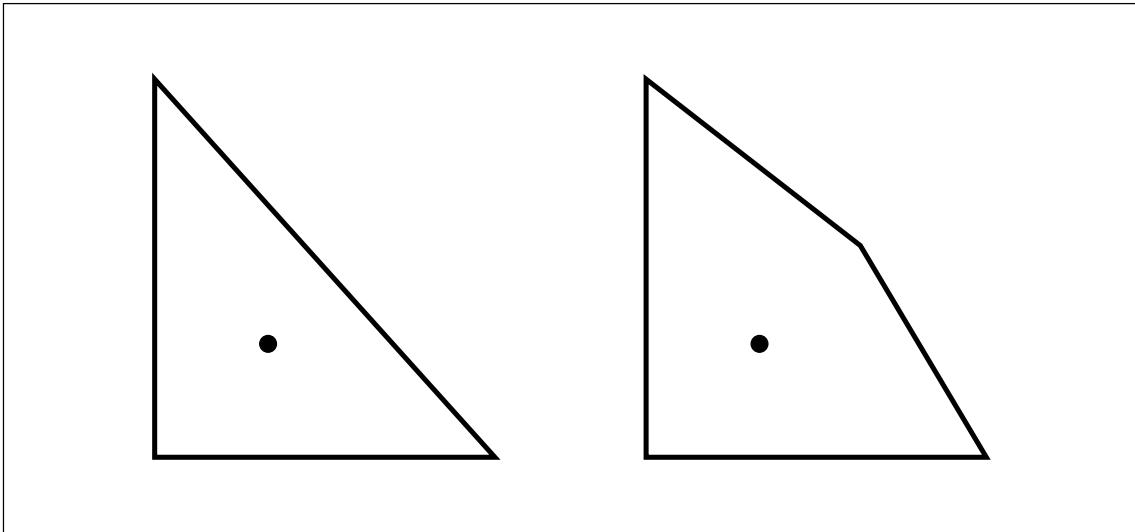


Abbildung A.6: („leichtes Rauschen“) Das rechte Polygon ist leicht verrauscht, indem an der diagonalen Seite ein zusätzlicher Meßpunkt eingefügt wurde

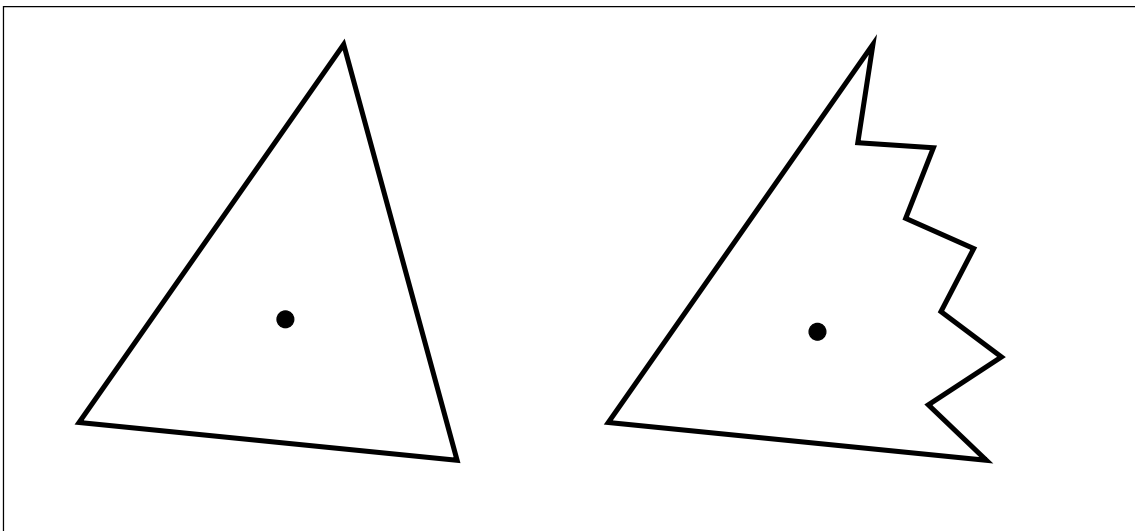


Abbildung A.7: („uneinheitliches Rauschen“) Das rechte Polygon hat eine stark verrauschte Seite, die Kernpunkte unterscheiden sich jedoch kaum

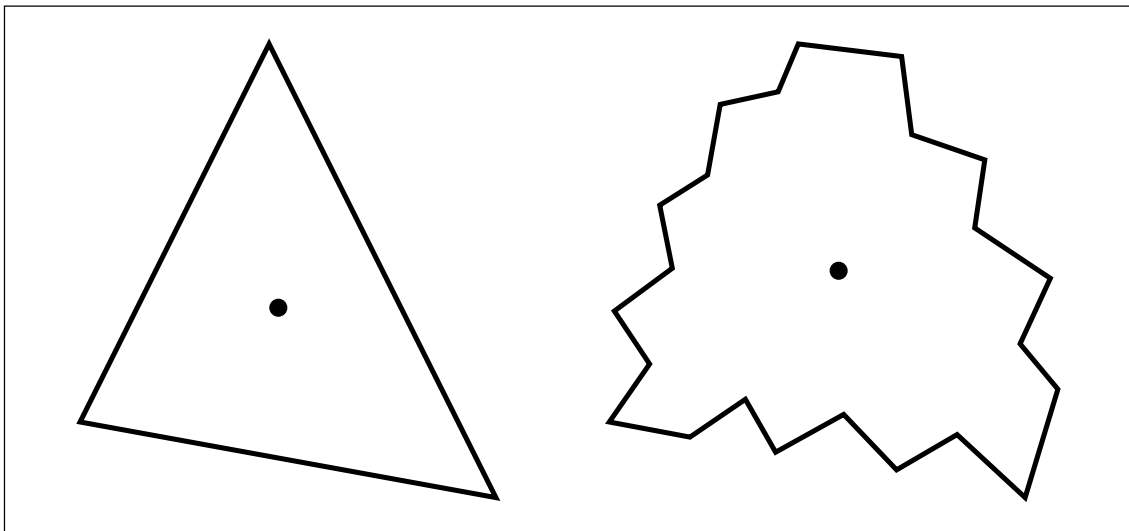
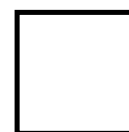


Abbildung A.8: („einheitliches Rauschen“) Ein einheitlich verrauschtes Dreieck, z.B. von einem Laserscanner aufgenommen, und ein unverraushtes Dreieck, z.B. aus dem Preprocessing gewonnen

## A.2 Vergleiche von Polygonen gegen ein Muster

Die Idee zu folgendem Vergleichsverfahren stammt aus [ACH<sup>+</sup>91]. Auch die Polygone, die hier verglichen werden, sind in etwa diesem Artikel (Seite 214) entnommen.

Als Vergleichsmuster dient jeweils das nebenstehende Quadrat. Die Tabelle enthält das Ähnlichkeitsmaß der letzten sieben Polygone im Vergleich dazu.



Die Werte der beiden FIF-Metriken sind erneut kleiner als die der PKF-Distanzen und liegen dadurch auch enger zusammen. Das ist wiederum auf die Stauchung der Polygone auf den Flächeninhalt 1 zurückzuführen.



Abbildung A.9: PKF-Metrik

2.21256	4.42994	5.81967	10.0243	15.4387	16.4494	18.1911
---------	---------	---------	---------	---------	---------	---------

Tabelle A.2: PKF-Metrik



Abbildung A.10:  $PKF_{lin}$ -Metrik

2.16822	4.75083	8.32637	8.90252	13.3249	15.8284	18.7481
---------	---------	---------	---------	---------	---------	---------

Tabelle A.3:  $PKF_{lin}$ -Metrik

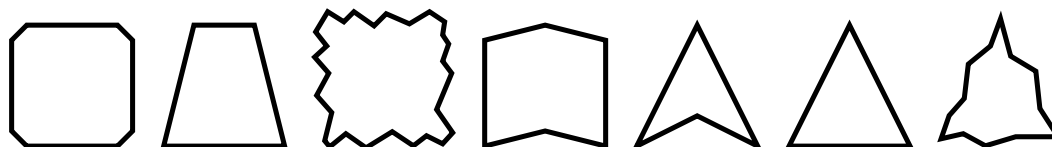


Abbildung A.11: FIF-Metrik

1.83816	3.84395	3.87478	3.93427	3.97508	4.93078	5.26266
---------	---------	---------	---------	---------	---------	---------

Tabelle A.4: FIF-Metrik

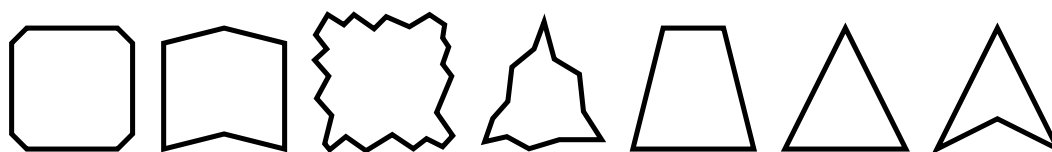


Abbildung A.12:  $FIF_{lin}$ -Metrik

1.54779	2.50141	3.29697	4.28675	6.61712	8.30526	9.04502
---------	---------	---------	---------	---------	---------	---------

Tabelle A.5:  $FIF_{lin}$ -Metrik



# Anhang B

## Begriffe und Symbole

Dieser Teil des Anhangs enthält eine Übersicht über Bezeichnungen, die möglicherweise in anderen Kontexten anders definiert oder generell sonst nicht verwendet werden. Allgemein bekannte Sachverhalte und Begriffe sind jedoch nicht aufgeführt, um die Darstellung übersichtlich zu halten. Ebenso sind keine Termini aufgeführt, denen im laufenden Text eine eigene Definition gewidmet war.

### B.1 Definitionen

**Autonomie** Eigenschaft eines mobilen Systems, sich ohne menschliche Eingriffe selbstständig zu lokalisieren und zu bewegen. Vollständige Autonomie eines Systems verlangt daher auch die Fähigkeit, auf unvorhergesehene Situationen und Bahnabweichungen bei der Bewegung zu reagieren.

**Hausdorffdistanz** Distanz  $\delta_H$  zwischen kompakten Punktmenge  $A$  und  $B$ , die eine Distanzfunktion  $\|\cdot\|$  für Punktmenge voraussetzt und wie folgt definiert ist:

$$\begin{aligned}\delta_H(A, B) &= \max(\tilde{\delta}_H(A, B), \tilde{\delta}_H(B, A)) \quad \text{mit} \\ \tilde{\delta}_H(A, B) &= \max_{a \in A} \min_{b \in B} \|a - b\|.\end{aligned}$$

Dabei heißt  $\tilde{\delta}_H$  die **gerichtete Hausdorffdistanz** der Punktmenge  $A$  und  $B$ .  $\delta_H$  ist eine Metrik, falls  $\|\cdot\|$  eine Metrik ist.

**Invarianz unter geometrischen Transformationen** Eigenschaft von Distanz- und Ähnlichkeitsabbildungen, unabhängig von diesen Transformationen zu sein, d.h. der Distanzwert zweier Objekte ändert sich nicht, wenn sie mittels der Transformationen verändert werden. Insbesondere haben Objekte die Distanz Null, wenn sie sich nur durch solche Transformationen unterscheiden.

**Kernpunkt** Punkt  $p$  im Innern oder auf dem Rand eines Polygons  $\mathcal{P}$ , von dem aus jeder innere und jeder Randpunkt des Polygons sichtbar ist, d.h. die Verbindungslinie von  $p$  zu einem solchen Punkt schneidet keine Polygonkante. Falls für  $\mathcal{P}$  ein solcher Punkt existiert, heißt  $\mathcal{P}$  **sternförmig**.

**kritischer Punkt, kritische Stelle** Beim Verschieben der Graphen zweier zusammengesetzter Funktionen: Lage der Graphen, bei der zwei  $\rightarrow$  *Übergangsstellen* zusammenfallen, d.h. die gleiche Abszisse haben.

**korrespondierende Punkte** Zwei Punkte von Polygonen  $\mathcal{P}$  und  $\mathcal{Q}$ , die bei einem Matching von  $\mathcal{P}$  und  $\mathcal{Q}$  aufeinander abgebildet werden.

**künstliche Kante** Kante eines Sichtbarkeits skeletts, die für die Überbrückung einer  $\rightarrow$  *Scheinecke* eingefügt wurde. Eine künstliche Kante kommt nicht als Kante in der Szene vor, auch nicht als Teil davon. Mindestens eine der beiden Ecken der künstlichen Kante ist eine Reflexecke der Szene.

**redundanter Eckpunkt** Eckpunkt eines Polygons, der kollinear mit seinen beiden Nachbarn ist und dadurch einen Innenwinkel von  $180^\circ$  erzeugt.

**Scheinecke** Ecke eines Sichtbarkeitspolygons eines Standorts  $p$ , die als erster Schnittpunkt der Verlängerung der Strecke von  $p$  zu einer sichtbaren  $\rightarrow$  *Verdeckungsecke*  $v$  der Szene über  $v$  hinaus mit einer Szenekante entsteht. Scheinecken liegen also „hinter“ (d.h. sind insbesondere kollinear mit) den Verdeckungsecken, von  $p$  aus betrachtet. Sie sind keine Szenenecken.

**Scheinkante** Kante eines Sichtbarkeitspolygons zwischen einer  $\rightarrow$  *Scheinecke* und einer benachbarten  $\rightarrow$  *Verdeckungsecke*. Sie ist stets kollinear mit dem Betrachterstandort  $p$ . Gibt es in einem Sichtbarkeitspolygon mehrere Scheinkanten, so schneiden sie sich alle in  $p$ .

**Sichtbarkeitszelle** Zusammenhängender, konvexer, polygonaler Teilbereich des Freiraums einer Szene, in dem alle Standorte dasselbe Sichtbarkeits skelett besitzen, d.h. in dem von allen Standorten aus dieselbe Menge von Szenenecken sichtbar ist (auch in derselben angularen Ordnung) und in der sich das Sichtbarkeitspolygon daher qualitativ nur wenig ändert.

**Stütz- oder Übergangsstelle** Abszisse des Graphen einer stückweise definierten Funktion, an der sich die Definitionsgleichung ändert. Auch der Punkt des Graphen an einer solchen Stelle wird als Übergangsstelle (des Graphen) bezeichnet.

**Verdeckungsecke** Reflexecke einer Szene (siehe Definition 1.5 auf Seite 6), für die gilt: Der aktuelle Betrachterstandort liegt im *Nebenwinkelbereich* der Ecke, d.h. so, daß genau eine der beiden adjazenten Kanten vom Standort aus verdeckt ist.

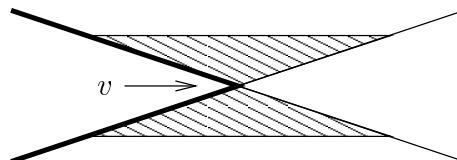


Abbildung B.1: Zur Definition Verdeckungsecke. Der Roboterstandort liegt im schraffierten Nebenwinkelbereich, wenn  $v$  eine Verdeckungsecke ist

**Verdeckungslinie** Verlängerung des Strahls von einer beliebigen Szenenecke  $u$  zu einer sichtbaren  $\rightarrow$  *Verdeckungsecke*  $v$ , wenn der Roboterstandort als  $u$  angenommen wird, über  $v$  hinaus, bis er die Szene in einer Kante trifft. (Dieser Auftreffpunkt ist eine  $\rightarrow$  *Szenecke* des von  $u$  aufgenommenen Sichtbarkeitspolygons.) Diese Linien sind die Grenzen der  $\rightarrow$  *Sichtbarkeitszellen*.

## B.2 Symbole

$\overline{AB}$	Strecke zwischen $A$ und $B$
$ AB $	Länge der Strecke zwischen $A$ und $B$ , auch $d_2(A, B)$ (Euklidischer Abstand)
$\overrightarrow{AB}$	Vektor oder orientierte Strecke von $A$ nach $B$
$\triangle ABC$	Dreieck der Punkte $A, B, C$ ; in der Form $\triangle pP_iP_{i+1}$ auch Sektor eines Polygons mit Kernpunkt $p$ und benachbarten Eckpunkten $P_i$ und $P_{i+1}$
$\partial\mathcal{P}$	Menge aller Randpunkte des Polygons $\mathcal{P}$
$f^2(x)$	Abkürzend für $(f(x))^2$ ( $f$ reelle Funktion)
$\mathbb{M}_P$	Menge aller $x$ aus $\mathbb{M} \subset \mathbb{R}$ mit der Eigenschaft $P$ (einstelliges Prädikat), d.h. $\mathbb{M}_P := \{x \in \mathbb{M} : P(x)\}$ . Beispiele dafür sind $\mathbb{R}_{>0}$ , $\mathbb{N}_{\neq a}$ etc.
$p \rightarrow X$	In $p$ ausgehender und durch $X$ verlaufender Strahl ( $p \neq X$ )
$\mathcal{P} = \mathcal{Q}$	Gleichheit der geometrischen Objekte $\mathcal{P}$ und $\mathcal{Q}$ , d.h. abhängig vom Kontext Kongruenz, Ähnlichkeit etc., allgemein Identität (siehe $\mathcal{P} \equiv \mathcal{Q}$ ) bis auf Anwendung bestimmter Transformationen
$\mathcal{P} \equiv \mathcal{Q}$	Identität der geometrischen Objekte $\mathcal{P}$ und $\mathcal{Q}$ , d.h. Übereinstimmung ihrer Punktmengendarstellungen in Koordinaten
$[x]$	Ganzer Anteil von $x \in \mathbb{R}_{\geq 0}$ , d.h. $[x] := \max\{n \in \mathbb{N} : n \leq x\}$



# Literaturverzeichnis

- [AAR94] H. Alt, O. Aichholzer, G. Rote: MATCHING SHAPES WITH A REFERENCE POINT. In: *Proceedings of the 10th Annual ACM Symposium on Computational Geometry*, S. 85-92, 1994.
- [ACH<sup>+</sup>91] E. M. Arkin, L. P. Chew, D. P. Huttenlocher, K. Kedem, J. S. B. Mitchell: AN EFFICIENTLY COMPUTABLE METRIC FOR COMPARING POLYGONAL SHAPES. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Nr. 13, S. 209-216, 1991.
- [AFRW96] H. Alt, U. Fuchs, G. Rote, G. Weber: MATCHING CONVEX SHAPES WITH RESPECT TO THE SYMMETRIC DIFFERENCE. *Bericht Nr. 87*, Bereich Diskrete Optimierung, Karl-Franzens-Universität Graz & Technische Universität Graz, Oktober 1996.
- [AG96] H. Alt, L. Guibas: DISCRETE GEOMETRIC SHAPES: MATCHING, INTERPOLATION, AND APPROXIMATION. *Handbook of Computational Geometry*, 1997 (zu erscheinen).
- [BS57] I. N. Bronstein, K. A. Semendjajew: TASCHENBUCH DER MATHEMATIK. Herausgegeben von G. Grosche, V. Ziegler, D. Ziegler, BSB B. G. Teubner Verlagsgesellschaft, Leipzig, und Verlag Nauka, Moskau, 1979.
- [DRW95] G. Dudek, K. Romanik, S. Whitesides: LOCALIZING A ROBOT WITH MINIMUM TRAVEL. In: *Proceedings of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms*, S. 437-446, 1995.
- [GMR95] L. Guibas, R. Motwani, P. Raghavan: THE ROBOT LOCALIZATION PROBLEM. In: Ken Goldberg, Dan Halperin, Jean-Claude Latombe, Randall Wilson (Hrsg.): *Algorithmic Foundations of Robotics*, S. 269-282, AK Peters, 1995.
- [HK90] D. P. Huttenlocher, K. Kedem: COMPUTING THE MINIMUM HAUSDORFF DISTANCE FOR POINT SETS UNDER TRANSLATION. In: *Proceedings of the 6th Annual ACM Symposium on Computational Geometry*, S. 340-349, 1990.
- [Karch96] O. Karch: A SHARPER COMPLEXITY BOUND FOR THE ROBOT LOCALIZATION PROBLEM. In: *Technischer Bericht Nr. 139*, Lehrstuhl für Informatik I, Universität Würzburg, Juni 1996.

- [Klb94] J. Kleinberg: THE LOCALIZATION PROBLEM FOR MOBILE ROBOTS. In: *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, 1994.
- [KNW97] O. Karch, H. Noltemeier, Th. Wahl: ROBOT LOCALIZATION – THEORY AND IMPLEMENTATION. In: *Proceedings of the 13th European Workshop on Computational Geometry (CG '97)*, Universität Würzburg, März 1997.
- [LEDA3.4] K. Mehlhorn, S. Näher, C. Uhrig: THE LEDA USER MANUAL, VERSION R 3.4. Max-Planck-Institut für Informatik, Saarbrücken, und Martin-Luther-Universität Halle-Wittenberg, 1996.
- [Maes90] M. Maes: ON A CYCLIC STRING-TO-STRING CORRECTION PROBLEM. In: *Information Processing Letters*, Nr. 35, S. 73-78, 1990.
- [Maes94] M. Maes: POLYGONAL SHAPE RECOGNITION USING STRING MATCHING TECHNIQUES. In: *Pattern Recognition*, Nr. 24, S. 433-440, 1994.
- [Mark95] R. Mark: VERGLEICH POLYGONALER FIGUREN. Studienarbeit, Universität Würzburg, 1995.
- [MATH] St. Wolfram: MATHEMATICA – A SYSTEM FOR DOING MATHEMATICS BY COMPUTER. Addison-Wesley Publishing Company Inc., 2nd Edition, 1991.
- [Rote92] G. Rote: A NEW METRIC BETWEEN POLYGONS, AND HOW TO COMPUTE IT. In: *Proceedings of the 19th ICALP Colloquium, Lecture Notes of Computer Science*, Nr. 623, S. 404-415, Springer-Verlag, 1992.
- [TY85] W. H. Tsai, S. S. Yu: ATTRIBUTED STRING MATCHING WITH MERGING FOR SHAPE RECOGNITION. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Nr. 7, S. 453-462, 1985.
- [WF74] R. A. Wagner, M. J. Fischer: THE STRING-TO-STRING CORRECTION PROBLEM. In: *Journal of the ACM*, Nr. 21(1), S. 168-173, 1974.

# Erklärung

Ich erkläre hiermit, daß ich die vorliegende Arbeit selbständig und nur unter Zuhilfenahme aller aufgeführten Quellen verfaßt habe.

Thomas Wahl