

Dynamische kartographische Visualisierung von Location Based Services mittels SVG

DIPLOMARBEIT

ausgeführt am
Geographischen Institut der Universität Zürich

von
Jürg Mannes

Zürich, Februar 2004

Betreuung: Dr. Dirk Burghardt
Begutachtung: Prof. Dr. Robert Weibel

Summary

Location based services provide information which is adapted to the current position of a user. The content of the map is determined by the interests of the user; the map shows only one of many combinations of data. Thus the use of precalculated visualisations is not possible; the map has to be produced dynamically in a short time. To allow changes of scale, an on-the-fly generalization that is carried out in real-time, is necessary.

A goal of this work was the development of methods which can be used for the real-time generalization of point data. It was recognized, that there are two types of point data which require different generalisation. The use of clustering techniques for an aggregation of point data was then examined in detail. A number of hierarchical, partitioning and graphbased algorithms were implemented. These methods were evaluated through a qualitative comparison. The hierarchical clustering method with single linkage produced the best results. Time measurements to see whether application in real time is possible were also made. For small datasets this seems realistic, but the computation time increases strongly with a growing number of points. Therefore a method to improve performance through a pre-processing is presented. Furthermore, a simple solution for automatic determination of an appropriate number of clusters is proposed.

Some possibilities for visualizing the results of generalisation are shown. The goal is a cartographically acceptable portrayal presenting information to users in an easily understandable way.

The third goal of this work was to find solutions, how generalisation can be integrated in an architecture for dynamic cartographic visualization. This system is built with technologies basing on open standards from the Open GIS Consortium and the W3C. The generalization was designed as service which makes use of XSLT to read the data and transform them to SVG. The generalization is performed with Java classes that were developed for this work. Examples of XSLT- and SVG-files as well as the generalization component's source code can be found in the annex and on the attached CD.

Zusammenfassung

Location Based Services vermitteln Informationen, die auf den aktuellen Aufenthaltsort des Benutzers abgestimmt sind. Der Inhalt der Karte wird von den Interessen der Nutzer bestimmt; die Karte stellt also nur eine von vielen Kombinationen von Daten dar. Die Verwendung von vorberechneten Visualisierungen ist deshalb nicht möglich; die Karten müssen in kürzester Zeit dynamisch erzeugt werden. Um dabei auch Änderungen des Masstabs zu erlauben, ist eine on-the-fly Generalisierung, die in Echtzeit ausgeführt wird, notwendig.

Ein Ziel dieser Arbeit war die Entwicklung von Methoden, mit welchen sich eine Echtzeitgeneralisierung von Punktdaten realisieren lässt. Es wurde erkannt, dass sich zwei Typen von Punktdaten unterscheiden lassen, die eine unterschiedliche Behandlung bei der Generalisierung erfordern. Detailliert untersucht wird die Eignung von Verfahren der Clusteranalyse zur Aggregation. Dazu wurden mehrere hierarchische, ein partitionierender (k-means) und ein graphbasierter Clusteringalgorithmus implementiert. Evaluert wurden diese Verfahren mit einem qualitativen Vergleich und mit Zeitmessungen, die Auskunft darüber geben, ob eine Anwendung in Echtzeit möglich ist. Die besten Ergebnisse wurden mit hierarchischem Clustering mit Single Linkage erzielt. Die Zeitmessungen ergaben, dass eine Anwendung in Echtzeit für kleinere Datensätze denkbar ist. Mit wachsender Anzahl Punkte steigt jedoch der Rechenaufwand stark an, weshalb eine Möglichkeit diskutiert wird, die Leistung durch eine im Voraus berechnete Generalisierung zu verbessern. Weiter wird eine einfache Methode vorgeschlagen, eine geeignete Anzahl von Clustern automatisch zu bestimmen.

Es werden verschiedene Varianten vorgestellt, die Ergebnisse der Generalisierung zu visualisieren. Das Ziel ist eine kartographisch gute Darstellung, die den Nutzern auf einfache Weise relevante Informationen gibt.

Das dritte Ziel war das Finden einer Möglichkeit, wie Generalisierung in eine Architektur zur dynamischen kartographischen Visualisierung integriert werden kann. Dieses System wurde mit Technologien aufgebaut, die auf offenen Standards des Open GIS Consortiums und des W3C basieren. Die Generalisierung wurde als Dienst gestaltet, der XSLT nutzt, um die Daten zu lesen und in SVG umzuwandeln. Die Generalisierungsverfahren wurden in einer Gruppe von Java-Klassen implementiert, die von XSLT mittels Erweiterungsfunktionen genutzt werden. Im Anhang und auf der beigelegten CD finden sich Beispiele von XSLT- und SVG-Dateien sowie der Quellcode der Generalisierungskomponente.

Dank

Ganz herzlich bedanken möchte ich mich bei allen, die mich bei meiner Arbeit unterstützt haben. Ganz besonders möchte ich meiner Familie danken, die mir mein Studium ermöglicht hat und mir stets ein guter Rückhalt war.

Dank geht auch an Dr. Dirk Burghardt für die Betreuung meiner Arbeit, die Ideen und die Motivation in schwierigeren Phasen. Ebenfalls für Betreuung und Ideen bedanken möchte ich mich bei Prof. Robert Weibel.

Vielmals bedanken möchte ich mich bei Lea Spahn, die mich immer motiviert und unterstützt hat.

Dank geht auch an Martin Beusch für die langen Diskussionen über Studium und Diplomarbeit.

Schliesslich möchte ich mich bei allen Institutsmitgliedern bedanken, die mir Hinweise und Anregungen gegeben haben, besonders bei Alistair Edwardes, der Ideen und kritische Anmerkungen einbrachte.

Maienfeld, im Februar 2004

Jürg Mannes

Inhalt

Summary	III
Zusammenfassung	IV
Dank	V
Verzeichnis der Abbildungen	VIII
Verzeichnis der Tabellen	VIII
1 Einleitung	1
1.1 Motivation und Problemstellung.....	1
1.2 Forschungsfragen.....	2
1.3 Abgrenzung.....	2
1.4 Aufbau der Arbeit.....	3
2 Grundlagen	5
2.1 Location Based Services.....	5
2.2 Generalisierung.....	6
2.2.1 Verschiedene Sichten der Generalisierung	7
2.2.2 Begriffsrahmen der Generalisierung	9
2.2.2.1 Das Modell von Brassel und Weibel.....	10
2.2.2.2 Das Modell von McMaster und Shea.....	10
2.2.3 Operatoren der Generalisierung	11
2.2.4 On-the-fly Generalisierung	13
2.3 Typologie von Punktdaten.....	14
2.3.1 Ortsfeste Punktdaten	15
2.3.2 Nicht ortsfeste Punktdaten	15
2.4 Geeignete Generalisierungsmethoden für Punktdaten.....	16
2.4.1 Auswahl/Elimination	16
2.4.2 Aggregation (Typisierung und Kombination)	17
2.5 Clusteranalyse.....	17
2.5.1 Ähnlichkeitsmasse	18
2.5.2 Clusteringverfahren	19
2.5.2.1 Partitionierendes Clustering.....	19
2.5.2.2 Hierarchisches Clustering.....	20
2.5.2.3 Graphbasiertes Clustering.....	22
2.5.3 Clustering und Generalisierung	23
2.5.3.1 Kartographische Generalisierung.....	23
2.5.3.2 Modellgeneralisierung.....	23
2.5.4 Vorstellung der verwendeten Algorithmen	24
2.5.4.1 Hierarchisches Clustering mit Single Linkage.....	24
2.5.4.2 Hierarchisches Clustering mit Complete Linkage.....	24
2.5.4.3 Maschenvereinfachung.....	25
2.5.4.4 k-means.....	25
2.5.4.5 Graphbasiertes Clustering mit dem Minimum Spanning Tree.....	26
2.5.5 Bestimmung der Anzahl Cluster	26
3 Eingesetzte Technologien	29
3.1 Interoperabilität.....	29
3.1.1 Das Open GIS Consortium	29

3.1.2 Dienste.....	30
3.1.2.1 Verkettung von Diensten.....	31
3.1.2.2 Geographische Dienste.....	32
3.2 Web Feature Server.....	32
3.3 XML.....	33
3.3.1 Geography Markup Language.....	34
3.3.2 Scalable Vector Graphics.....	34
3.3.3 Extensible Stylesheet Language Transformations.....	36
3.3.3.1 Erweiterungsfunktionen.....	36
4 Implementation.....	39
4.1 Integration von Generalisierung.....	39
4.2 Aufbau der Architektur.....	39
4.3 Diskussion der Architektur.....	42
4.3.1 Generalisierungsdienst.....	42
4.3.2 Andere Möglichkeiten.....	43
4.3.3 Location Based Services.....	43
4.4 Implementation der Generalisierung.....	43
4.5 Verwendete Daten.....	46
4.6 Symbolisierung.....	48
4.6.1 Darstellung durch lokale Signaturen.....	48
4.6.2 Darstellung durch Flächen.....	50
5 Resultate.....	53
5.1 Evaluation des Clusterings.....	53
5.1.1 Kriterien zur qualitativen Auswertung.....	53
5.1.1.1 Kritische Anmerkungen.....	53
5.1.2 Vergleich der Algorithmen.....	55
5.1.2.1 Hierarchisches Clustering.....	55
5.1.2.2 Maschenvereinfachung.....	56
5.1.2.3 K-means.....	56
5.1.2.4 Graphbasiertes Clustering mit dem Minimum Spanning Tree.....	57
5.1.2.5 Versuche mit weiteren Daten.....	58
5.2 Zeitbedarf.....	61
5.2.1 Generalisierung und Visualisierung.....	61
5.2.2 Architektur mit Server.....	63
5.3 Einsatz für Modellgeneralisierung.....	64
5.4 Evaluation der Bestimmung der Clusterzahl.....	64
5.5 Diskussion der Resultate.....	66
6 Schlussfolgerungen und Ausblick.....	69
6.1 Schlussfolgerungen.....	69
6.2 Ausblick.....	71
7 Literatur.....	73
8 Anhang.....	77
GML.....	77
XSLT.....	78
SVG.....	85

Verzeichnis der Abbildungen

Abb. 2.1: Generalisierung als Abfolge von Modelltransformationen.....	8
Abb. 2.2: Begriffsrahmen für die Kartengeneralisierung.....	9
Abb. 2.3: Modell der digitalen Generalisierung nach McMaster und Shea.....	11
Abb. 2.4: Typologie von Clusteringverfahren mit ausgewählten Algorithmen.....	20
Abb. 2.5: Hierarchisches Clustering (Single Linkage) mit Dendrogramm.....	20
Abb. 2.6: Darstellung einiger Definitionen von Ähnlichkeit zwischen zwei Clustern.....	21
Abb. 2.7: Darstellung von Begriffen der Graphentheorie.....	22
Abb. 2.8: Beispiel eines Evaluationsgraphen zur Bestimmung einer geeigneten Anzahl Cluster.....	27
Abb. 3.1: Interoperabilität durch Schnittstellen.....	30
Abb. 3.2: Undurchsichtige Verkettung von Diensten („Opaque chaining“)......	31
Abb. 3.3: Mehrstufiges System von OGC Web Services für Webmapping.....	33
Abb. 3.4: Beispiel für Umwandlung von GML zu SVG mittels XSLT.....	35
Abb. 4.1: Schematische Darstellung des aufgebauten Systems.....	40
Abb. 4.2: Sequenzdiagramm der Herstellung einer Karte im Rahmen der verwendeten Architektur.....	41
Abb. 4.3: Klassendiagramm der Generalisierungskomponente.....	44
Abb. 4.4: Darstellung der verwendeten Datensätze.....	47
Abb. 4.5: Darstellung von generalisierten Steinbockbeobachtungen mittels skalierten lokalen Signaturen.....	48
Abb. 4.6: Darstellung von generalisierten Steinbockbeobachtungen durch konvexe Hüllen.....	49
Abb. 4.7: Darstellung von generalisierten Steinbockbeobachtungen durch Standardabweichungsellipsen.....	50
Abb. 5.1: Aufteilung von Beobachtungen in zwei Cluster.....	54
Abb. 5.2: Vorschlag für ein manuelles Clustering der Beobachtungen von Mardern.....	55
Abb. 5.3: Beobachtungen von Mardern zwischen 1995 und 2002: Generalisiert mit Single Linkage und Complete Linkage.....	56
Abb. 5.4: Beobachtungen von Mardern zwischen 1995 und 2002, generalisiert mit Maschenvereinfachung.....	56
Abb. 5.5: Beobachtungen von Mardern zwischen 1995 und 2002, generalisiert mit dem k-means Algorithmus und graphbasiertem Clustering.....	57
Abb. 5.6: Generalisierte Beobachtungen von Steinböcken im Januar zwischen 1998 und 2002 im Gebiet Ofenpass.....	58
Abb. 5.7: Generalisierte Beobachtungen von Gemsens im August 2002.....	59
Abb. 5.8: Beobachtungen von Gemsens im August 2002 im Gebiet Il Fuorn.....	60
Abb. 5.9: Zeitbedarf für Generalisierung und Visualisierung mittels XSLT.....	62
Abb. 5.10: Zeitbedarf für Generalisierung und Visualisierung.....	63
Abb. 5.11: Mit Single Linkage berechnete Cluster in Beobachtungen von Mardern.....	65

Verzeichnis der Tabellen

Tab. 2.1: Typologie von Generalisierungsoperatoren.....	12
Tab. 2.2: Typologie von Punktdaten.....	15
Tab. 4.1: Vergleich von automatisch und manuell ermittelten Clusterzahlen für verschiedene Testfälle.....	66

1 Einleitung

1.1 Motivation und Problemstellung

Location Based Services könnten zu einem grossen Wachstumsgebiet im Telekommunikationsmarkt werden. Neben der Industrie ist auch die Politik an der Entwicklung dieses Gebiets interessiert: Die EU fördert Technologien für die Informationsgesellschaft in Forschungs-Rahmenprogrammen. Einige der unterstützten Projekte beschäftigen sich mit Location Based Services, so z.B. das Projekt Webpark (2003), an dem unter anderem das Geographische Institut der Universität Zürich als Partner beteiligt ist.

Mit der grösser werdenden Verbreitung von Karten im Internet ist das Bedürfnis der Benutzer gewachsen, die gezeigten Inhalte selbst zu bestimmen. Neben topographischen Daten sollte dabei auch ein Zugriff auf thematische Daten möglich sein. Die Daten, die dargestellt werden, sind nur eine von vielen Möglichkeiten. Eine Verwendung von vorberechneten Visualisierungen ist deshalb nicht möglich; die Karten müssen in kürzester Zeit dynamisch erzeugt werden. Um dabei auch Änderungen des Massstabs zu ermöglichen, ist eine on-the-fly Generalisierung, die in Echtzeit ausgeführt wird, notwendig.

Im Rahmen von Location Based Services werden oft Karten zur Vermittlung von Information verwendet. Der Inhalt dieser Karten ist neben den Interessen eines Nutzers zusätzlich von seiner (sich ändernden) Position abhängig. Eine dynamische Erzeugung der Karten ist also speziell wichtig. Dazu kommt, dass auf den kleinen Bildschirmen, die zum Einsatz gelangen, nur eine beschränkte Menge von Information vermittelt werden kann. Es gilt also, durch eine Generalisierung eine Vereinfachung der Daten zu erreichen, um die Fragestellungen des Nutzers zu beantworten.

Ein erstes Ziel dieser Arbeit ist der Aufbau einer für dynamische kartographische Visualisierung geeigneten Architektur. Ein solches Framework kann als Komponente in einem Location Based Service eingesetzt werden, eignet sich aber auch für Webmapping ohne mobilen Kontext. Für diese Architektur sollen aktuelle Technologien der Internet- und Computerkartographie Verwendung finden. Die Architektur dient in erster Linie als Test- und Entwicklungsumgebung für die in dieser Arbeit durchgeführten Analysen. Daneben liefert sie auch Erkenntnisse über die Verwendbarkeit der eingesetzten Technologien im Rahmen von Location Based Services.

Ein zweites Ziel ist die Untersuchung von geeigneten Techniken der Generalisierung für ortsbasierte Dienste. Die Generalisierung soll für Punktdaten untersucht werden, wobei das Hauptaugenmerk auf der Verwendung von Clusteringverfahren zur Aggregation liegt.

Weiter sollen Wege aufgezeigt werden, wie die Resultate der Generalisierung im Rahmen eines

Location Based Service visualisiert werden können. Das Ziel ist eine kartographisch gute Darstellung, wobei die kleine Bildschirmgröße der für ortsbasierte Dienste verwendeten Geräte zu berücksichtigen ist.

1.2 Forschungsfragen

Generalisierung ist ein wichtiger Bestandteil einer guten Visualisierung. Es ist deshalb wichtig, Methoden zu finden, mit denen Daten für ortsbasierte Dienste generalisiert werden können. Für diese Arbeit stand die Suche von Methoden im Vordergrund, mit denen eine Generalisierung in Echtzeit durchgeführt werden kann.

Wie kann eine Echtzeit-Generalisierung in einem Location Based Service aussehen?

Das Ziel von Webpark (2003) ist es, ein interaktives Informationssystem für Naturparks zu entwickeln. In dieser Arbeit wird als Teilaspekt eines solchen Systems die Visualisierung von Wildtierbeobachtungsdaten untersucht. Als Anwendungsfall für diese Arbeit soll ein Dienst für die „Auswahl eines Standortes zur Beobachtung von Tieren und Pflanzen“ entwickelt werden, der auf Beobachtungsdaten des Schweizerischen Nationalparks, der ebenfalls am Webpark-Projekt beteiligt ist, zurückgreift. Das Ziel ist es, die Daten in Echtzeit zu generalisieren und sie anschliessend kartographisch gut darzustellen.

Wie können die Daten auf kleinen Displays optimal dargestellt werden?

Diese Untersuchungen sollen im Rahmen einer Architektur durchgeführt werden, wie sie allenfalls für einen Location Based Service verwendet werden könnte. Es sollen auch Möglichkeiten für die Integration einer Generalisierungskomponente in diese Architektur gefunden werden.

Wie könnte eine Architektur für ortsbasierte Dienste aussehen? Wie lässt sich darin eine Generalisierung integrieren?

1.3 Abgrenzung

Es wird in dieser Arbeit darauf verzichtet, einen vollständigen Prototypen des oben angesprochenen Dienstes zu entwickeln. Der zusätzliche Aufwand für die Entwicklung einer vollständigen Applikation, besonders einer Benutzeroberfläche, wäre nicht gerechtfertigt gewesen, da die Fragestellung dieser Arbeit im Bereich der Visualisierung und der Technik liegt. Um aus den Resultaten dieser Arbeit ein fertiges Produkt zu entwickeln, ist, neben der Integration einer Datenbank, vor allem eine gute Benutzerschnittstelle nötig.

Weiter wurde von den in Kapitel zwei skizzierten Möglichkeiten der Generalisierung nur Methoden implementiert, die auf Verfahren der Clusteranalyse beruhen.

1.4 Aufbau der Arbeit

Die Arbeit ist in sechs Kapitel gegliedert. In diesem ersten Kapitel wird ein kurzer Überblick über das Thema gegeben und auf Zielsetzung und Fragestellungen der Arbeit eingegangen.

Das Kapitel „Grundlagen“ beginnt mit einer kurzen Vorstellung von Location Based Services mit einer kleinen Zusammenstellung von offenen Forschungsfragen in diesem Gebiet. Anschliessend werden Grundlagen der automatischen Generalisierung, Operatoren und on-the-fly Generalisierung diskutiert. Weiter wird eine Einteilung von Punktdaten in zwei Kategorien vorgenommen. Bei der ersten Kategorie ist für die Visualisierung einzig die absolute Position massgebend. Für eine zweite Kategorie von Punktdaten spielt neben der absoluten auch die relative Position eine Rolle, d.h. die Nachbarschaft zu anderen Objekten und durch die Objekte gebildete Muster müssen bei der Darstellung berücksichtigt werden. Es folgen Vorschläge von geeigneten Generalisierungsmethoden für beide Klassen. Ausführlich besprochen werden dann verschiedene Clusteringverfahren, die in dieser Arbeit zur Aggregation von Punktdaten Verwendung finden. Den Abschluss dieses Kapitels bilden Überlegungen zur automatischen Bestimmung einer geeigneten Anzahl von Clustern.

Das dritte Kapitel präsentiert zunächst einige Gedanken zur Interoperabilität im GIS-Bereich und erläutert die Grundidee von Diensten. Weiter werden die verwendeten Technologien vorgestellt: Web Feature Server (WFS), Geography Markup Language (GML), Extensible Stylesheet Language Transformations (XSLT) und Scalable Vector Graphics (SVG).

Das Kapitel „Implementation“ beschreibt den Aufbau eines offenen Systems mit den diskutierten Technologien und deren Zusammenspiel beim Bereitstellen einer Karte. Es werden verschiedene Möglichkeiten besprochen, wie eine Generalisierung in ein solches System integriert werden kann. Nach der Beschreibung der Generalisierungskomponente werden die für diese Arbeit verwendeten Daten vorgestellt. Verschiedene Möglichkeiten der Visualisierung dieser Daten schliessen das Kapitel ab.

Im fünften Kapitel werden die Resultate der Arbeit präsentiert. Zum einen werden die Resultate der verwendeten Clusteringverfahren bei der kartographischen Generalisierung miteinander verglichen. Kurz eingegangen wird auf eine Möglichkeit, Clustering in der Modellgeneralisierung zu nutzen. Weiter erfolgt eine Evaluation der automatischen Bestimmung der Clusteranzahl.

Das sechste Kapitel enthält Schlussfolgerungen und einen Ausblick.

Im Anhang finden sich kommentierte Beispiele von GML, XSLT und SVG.

Zusätzlich liegt dieser Arbeit eine CD bei, auf der sich der komplette für diese Arbeit entwickelte Code befindet.

2 Grundlagen

Dieses Kapitel beschäftigt sich mit den Grundlagen für diese Arbeit und beginnt mit einem Einblick in Location Based Services. Anschliessend wird auf Generalisierung eingegangen, wobei der Schwerpunkt auf der automatischen Generalisierung liegt. Zu diesem Zweck werden konzeptionelle Modelle und eine Typologie von Operatoren vorgestellt.

Ein Ziel dieser Arbeit ist die Generalisierung von Punktdaten: Dazu wird eine Unterteilung von Punktdaten in zwei Kategorien vorgenommen, die unterschiedliche Verfahren der Generalisierung erfordern.

Weiter werden in diesem Kapitel verschiedene Clusteringverfahren vorgestellt, die in dieser Arbeit zur Aggregation von Punktdaten Verwendung finden. Zum Schluss wird noch kurz auf Möglichkeiten zur Ermittlung einer optimalen Zahl von Clustern eingegangen.

2.1 Location Based Services

Unter Location Based Services (LBS), auf deutsch als standortbasierte Dienste oder auch ortsbezogene Anwendungen bezeichnet, versteht man die auf den aktuellen Aufenthaltsort des Benutzers abgestimmte Informationsvermittlung mittels mobiler Endgeräte. Die gelieferte Information soll also relevant sein für die gegenwärtige Position des Nutzers. Eine wichtige Komponente eines LBS ist die Adaption, d.h. die Anpassung, der Information an den Kontext. Reichenbacher (2002:128ff) schlägt ein Rahmenwerk für eine adaptive Visualisierung für mobile Informationssysteme vor: *„Ziel ist, die Visualisierung von Geoinformation in Bezug auf den Benutzer zu personalisieren und an das verwendete Gerät sowie den aktuellen Kontext anzupassen und zwar möglichst so, dass es keiner weiteren Anpassung durch den Benutzer mehr bedarf, weil die Visualisierung schon optimiert ist.“* Die Adaption erfolgt also auf drei Ebenen: Zuerst erfolgt eine Personalisierung, bei der die Daten aufgrund des Vorwissens und der Interessen des Benutzers und der aktuellen Aufgabe selektiert werden. Die Information wird dann den Erfordernissen des verwendeten Geräts (Displaygrösse, Speicher, Prozessor) angepasst. Schliesslich soll die Darstellung der Information wiederum an Benutzer, Aufgaben und Kontext angepasst werden.

Ermöglicht wird die Verbreitung von komplexeren standortbasierten Diensten durch die Einführung von neuen Telekommunikationsnetzen wie z.B. UMTS, die höhere Übertragungsraten ermöglichen. Die Betreiberfirmen haben grosse Investitionen getätigt, die sie nun wieder amortisieren müssen, was nur mit attraktiven Angeboten möglich ist, die neue Kunden anziehen. *„Einnahmen können nur durch attraktive, vom Anwender akzeptierte mobile Dienste generiert werden, für die der Nutzer auch zu zahlen bereit ist“* (Zipf, 2003:6).

Die Technologie für die Verbreitung von Location Based Services ist vorhanden. Die bisherige Entwicklung von LBS war jedoch stark technologiegetrieben und berücksichtigte kartographische Anforderungen nur ungenügend, z.B. Überlagerung von Symbolen, fehlende Generalisierung (Reichenbacher, 2002:126). Es besteht ein erheblicher Forschungsbedarf bezüglich verschiedenster Aspekte von standortbasierten Diensten. An dieser Stelle soll ein kurzer Überblick über Fragestellungen im Bereich von Location Based Services gegeben werden (nach Zipf 2003; Reichenbacher, Meng 2003):

- **Mobilität und Kartographie:** Was für Anforderungen stellen mobile Nutzer an Karten und welche Kartenfunktionalität ist deshalb notwendig? Welche anderen Möglichkeiten als Karten gibt es zur Darstellung von Information im mobilen Umfeld und wie gut sind diese für bestimmte Aufgaben geeignet?
- **Nutzungssituation:** Gibt es typische Nutzungssituationen? Kann der Kontext von Nutzungssituationen formalisiert werden? Wie kann die Information an diesen Kontext adaptiert werden?
- **Benutzermodellierung:** Welche Eigenschaften eines Benutzers haben einen Einfluss auf Inhalt und Design einer Karte? Wie gross ist der Einfluss einer Eigenschaft und wie beeinflusst dies die Adaption der Karte? Ist es möglich, typische Benutzergruppen zu identifizieren?
- **Adaption:** Welche Elemente einer Karte sind adaptierbar? Welche davon sollen adaptiert werden? Was sind geeignete Adaptionmethoden und wie lassen sich diese implementieren?

Die Fragestellungen in der vorliegenden Arbeit betreffen vor allem den Bereich der Adaption.

2.2 Generalisierung

Eine Karte stellt eine verkleinerte Abbildung der realen Welt dar. In der Verkleinerung können aber niemals alle Einzelheiten enthalten sein, die wir in der Wirklichkeit finden, da dazu schlicht der Platz fehlt. Die in einer Karte dargestellte Information kann also immer nur eine Abstraktion der Realität sein. Um diese Abstraktion zu erreichen, die je nach Kartenmassstab verschieden stark sein muss, ist

Generalisierung erforderlich. Dabei wird das Ziel verfolgt, Wichtiges zu erhalten, Unwichtiges wegzulassen und Charakteristisches zu betonen. Was in einer Karte wichtig ist, wird durch deren Verwendungszweck bestimmt. Generalisierung in der traditionellen Kartographie kann man also zusammenfassen als „selection and simplified representation of detail appropriate to the scale and/or purpose of the map“¹ (Neumann 1997). Ein Überblick über Generalisierung in der herkömmlichen Kartographie findet sich beispielsweise bei Hake und Grünreich (1994: 110-117).

Nicht nur die Visualisierung von Daten erfordert eine Generalisierung: Für Müller (1991: 457) ist sie vor allem auch ein Prozess, welcher den Informationsgehalt maximiert und damit analytischen Zwecken dient. Generalisierung, Abstraktion und Reduktion von Komplexität, ist eine fundamentale Aktivität des Menschen und sowohl Teil des wissenschaftlichen Arbeitens wie auch des alltäglichen Verhaltens (Brassel und Weibel 1988: 230). Besonders im digitalen Bereich mit seinen riesigen Datenmengen ist deshalb Generalisierung nötig, um mehr Informationen zu gewinnen. Eine weitere Motivation für Generalisierung ist die Tatsache, dass räumliche Phänomene und Prozesse meistens massstabsabhängig sind und auf der jeweils geeignetsten Stufe untersucht werden sollten (Müller et al. 1995: 3).

2.2.1 Verschiedene Sichten der Generalisierung

Da unterschiedliche Motivationen für Generalisierung existieren, gibt es auch verschiedene Ansichten des ganzen Prozesses. Eine erste betrachtet Generalisierung als einen Prozess, der Übergänge zwischen verschiedenen Modellen der realen Welt ermöglicht. Dabei nehmen Details ab, während der Informationsgehalt für einen bestimmten Zweck maximiert wird. Grünreich (1992) entwickelte ein Modell für diese Sicht der Generalisierung, das seitdem von anderen Autoren übernommen wurde (Weibel und Dutton 1999: 127), und das die Übergänge zwischen verschiedenen Modellen zeigt (vgl. Abb. 2.1):

- *Objektgeneralisierung* findet bei der Konstruktion des Primärmodells, der Originaldatenbank statt. Da die Datenbank stets ein vereinfachtes Modell der Realität ist, ist ein gewisser Grad an Generalisierung notwendig. Dieser Prozess ist sehr ähnlich auch bei der Datenaufbereitung für eine traditionelle Karte nötig.
- *Modellgeneralisierung* existiert hingegen nur im digitalen Bereich. Hier wird Generalisierung nicht auf die graphische Repräsentation der Daten angewendet, sondern auf die Daten selbst. Das Ziel ist dabei eine kontrollierte Reduktion der Daten für einen bestimmten Zweck,

¹ Die englische Definition wurde der deutschen vorgezogen, da diese etwas umständlich erscheint: „Gesamtheit der beim Aufnehmen oder bei kartographischer Darstellung auftretenden Generalisierungsvorgänge, mit denen - massstabs oder themenbedingt – geometrisch, begrifflich und zeitlich die unwesentlichen Einzelheiten vernachlässigt werden und Wesentliches erhalten bleibt oder in übergeordnete Begriffe überführt wird“ (Neumann 1997).

beispielsweise um Speicherplatz zu sparen oder Berechnungen schneller ausführen zu können. Modellgeneralisierung kann als Vorstufe der kartographischen Generalisierung eingesetzt werden, orientiert sich aber nicht an der graphischen Darstellung der Daten.

- Unter *kartographischer Generalisierung* versteht man die Generalisierung von räumlichen Daten für die graphische Darstellung. Das Resultat sind Visualisierungen der Daten, wobei die Ziele dieser Generalisierung prinzipiell gleich denjenigen der konventionellen Kartographie sind.

Das oben präsentierte Modell folgt einer *prozessorientierten Sicht*, in der Generalisierung als Transformation einer detaillierten Datenbank in eine Karte oder Datenbank mit geringerer Komplexität verstanden wird. Dabei gilt es zu beachten, dass für eine automatische Durchführung dieser Transformationen noch viel Entwicklungsarbeit notwendig ist. Daher wird auch ein anderer Ansatz verfolgt, der als *repräsentationsorientierte Sicht* bezeichnet wird. Dazu werden Multiskalendatenbanken entwickelt, die Repräsentationen von verschiedenen Massstäben zu einer konsistenten multiplen Repräsentation zusammenfassen. Ein Problem der multiplen Repräsentationen ist die Erhaltung der für eine Datenbank zentralen Konsistenz, da man redundante Daten speichert.

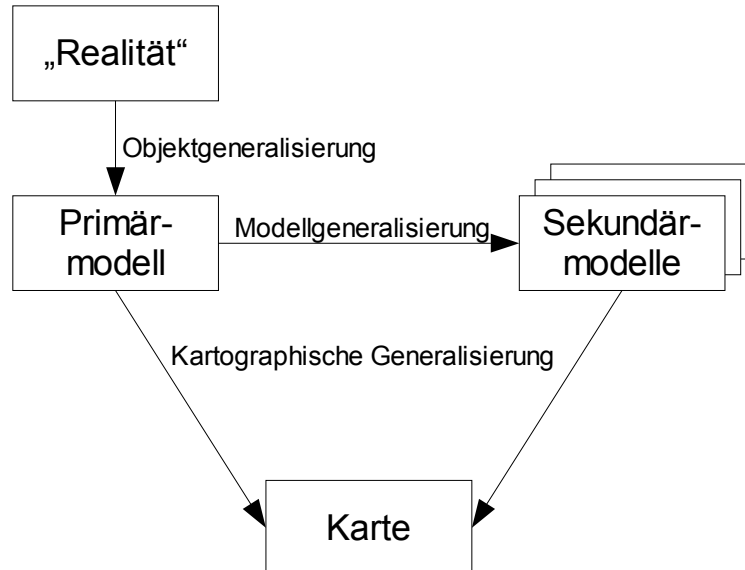


Abb. 2.1: Generalisierung als Abfolge von Modelltransformationen (nach Grünreich (1985) in Weibel und Dutton (1999)).

Es gibt verschiedene Möglichkeiten, um mit der oben angesprochenen Redundanz umzugehen. Eine Möglichkeit ist ein preprocessing der Daten und die Speicherung der Generalisierungsergebnisse, damit nachher Karten in verschiedenen Massstäben on-the-fly erzeugt werden können. Damit werden keine expliziten Repräsentationen verschiedener Massstäbe benötigt, man spricht daher von impliziten multiplen Repräsentationen. Um bei der Kartenerstellung lange Wartezeiten zu vermeiden,

werden die Resultate der Generalisierung in speziellen Datenstrukturen gespeichert (van Oosterom 1995: 120).

Eine andere Möglichkeit ist die explizite multiple Repräsentation. Dabei integriert man bestehende Repräsentationen jeweils eines Masstabs und modelliert die Übergänge zwischen den Masstäben. Das Problem dieses Ansatzes ist der Unterhalt der Datenbank, besonders bei Änderungen von Objekten, die in allen Masstäben vorgenommen werden müssen.

Cecconi (2003: 14) schlägt als dritte Möglichkeit die ableitungsorientierte Sicht vor, die eine Kombination von prozess- und repräsentationsorientierter Sicht darstellt. Von einem Datensatz werden durch einen Generalisierungsprozess verschiedene Detailstufen abgeleitet. Dadurch bleibt der Datensatz konsistent und die verschiedenen Repräsentation eines Objektes sind miteinander verbunden.

Das Kapitel 2.2.1 stützt sich ab auf Weibel und Dutton (1999: 127-131).

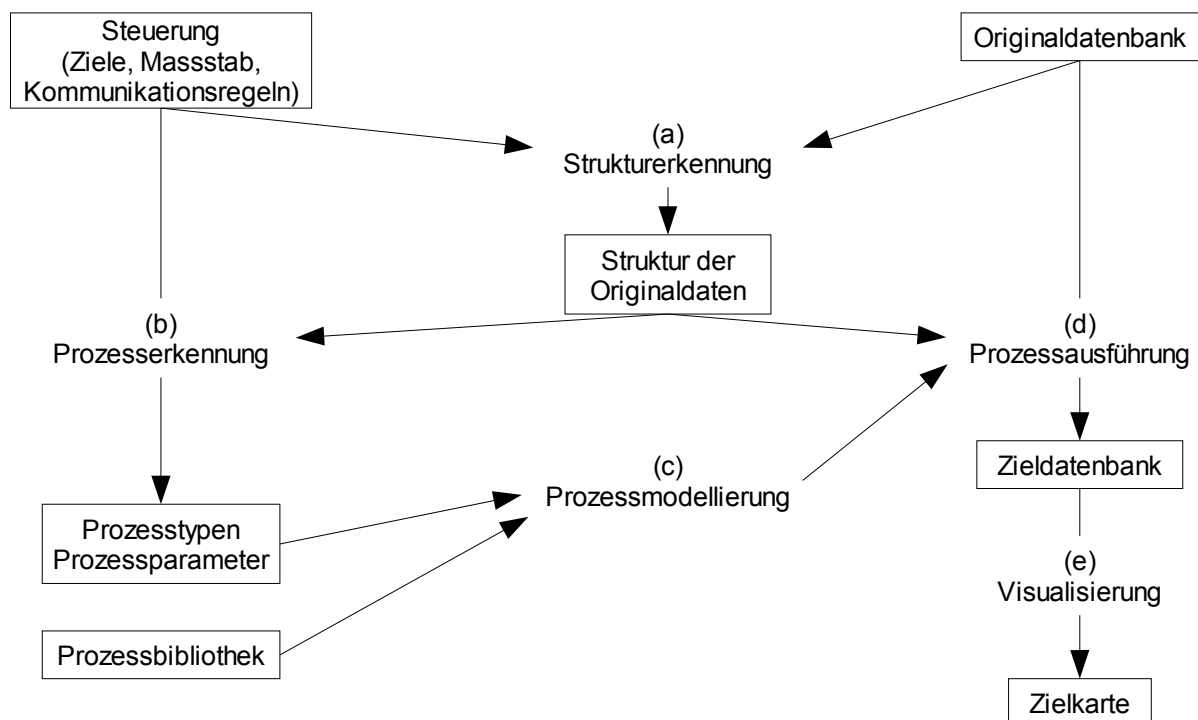


Abb. 2.2: Begriffsrahmen für die Kartengeneralisierung (nach Brassel und Weibel 1988: 231).

2.2.2 Begriffsrahmen der Generalisierung

Um ein voll funktionsfähiges System zur automatischen kartographischen Generalisierung zu entwickeln, sind gemäss McMaster (1991: 21) drei komplexe Probleme zu lösen: Erstens muss eine Einigung über einen umfassenden, formalen Begriffsrahmen erreicht, zweitens müssen die Verfahren für die einzelnen Generalisierungsoperatoren entwickelt, programmiert und getestet werden und drittens muss kartographisches Wissen aus Karten und von Experten in Regeln umgesetzt werden.

Im Folgenden werden zwei Begriffsrahmen vorgestellt.

2.2.2.1 Das Modell von Brassel und Weibel

Brassel und Weibel (1988) entwickelten den in Abb. 2.2 wiedergegebenen Begriffsrahmen des Generalisierungsprozesses. Die Generalisierung beginnt mit der Anwendung einer Strukturerkennung (a) auf die Originaldatenbank. Das Ziel ist die Erkennung von Objekten und ihren räumlichen Beziehungen sowie Messungen der relativen Wichtigkeit. Gesteuert wird dieser Prozess unter anderem durch die Ziele der Generalisierung und den Massstab der Zielkarte. Basierend auf den Strukturen, die bei Schritt (a) gefunden werden, kann man nun den Generalisierungsprozess definieren (b). Dazu müssen verschiedene Parameter definiert werden. Ist ein Generalisierungsprozess definiert, kann er als Abfolge von einzelnen Operationen modelliert werden, was als Prozessmodellierung (c) bezeichnet wird. Dabei werden Regeln und Verfahren aus einer Prozessbibliothek mit den vorher bestimmten Parametern zu einem Prozess zusammengestellt. Dieser wird anschliessend auf die Originaldatenbank angewendet (d). Die Prozessausführung besteht aus einer Abfolge von Operationen wie sie im Schritt (c) festgelegt wurde und liefert als Resultat die Zieldatenbank. Die darin enthaltenen Daten werden anschliessend visualisiert (e).

Wesentlich am Modell ist laut McMaster (1991: 26-27) die Prozessbibliothek genannte Komponente, welche die Regeln und Verfahren für die Generalisierung enthält. Bei der Entwicklung einer solchen Bibliothek müssen kritische Entscheidungen getroffen werden, welche Generalisierungsoperatoren (vgl. 2.2.3) notwendig sind, welches Wissen in Form von Regeln vom System erfasst sein soll und wie dieses strukturiert wird, sowie welche Parameter für die Implementation der Regeln und Operatoren notwendig sind.

2.2.2.2 Das Modell von McMaster und Shea

Ein noch umfassenderer Begriffsrahmen stammt von McMaster und Shea (1992). Ihr Modell (Abb. 2.3) teilt den Generalisierungsprozess in drei Teilgebiete auf. Neben den vorwiegend technischen Punkten der kartometrischen Auswertung und den räumlichen und Attributtransformationen enthält das Modell auch die philosophischen Ziele der digitalen Generalisierung, d.h. warum überhaupt generalisiert werden soll. Die Gründe für die Generalisierung werden in die folgenden drei Gruppen eingeteilt: allgemeine kartographische Prinzipien (theoretische Elemente), spezielle Ansprüche an das betrachtete Generalisierungsproblem (anwendungsspezifische Elemente) und rechnerische Anforderungen. Der zweite Bereich ist die kartometrische Auswertung, welche die Bedingungen ermittelt, wann generalisiert werden soll. Das dritte Teilgebiet umfasst schlussendlich die Auswahl von geeigneten Transformationen, die bestimmen, wie generalisiert wird.

Der zweite Bereich im Begriffsrahmen von McMaster und Shea verfolgt im Wesentlichen die

gleichen Ziele wie die Schritte Strukturerkennung und Prozesserkennung im Modell von Brassel und Weibel (vgl. Kapitel 2.2.2.1). Räumliche und holistische Masse werden verwendet, um Strukturen und Formen in den Originaldaten zu beschreiben. Aufgrund der gemessenen Werte kann bestimmt werden, ob bestimmte geometrische Bedingungen erfüllt sind, und eine Generalisierung ausgelöst werden muss. Die Transformationssteuerung ist dann verantwortlich für die Auswahl geeigneter Operatoren, Algorithmen und Parameter (Weibel und Dutton 1999: 133).

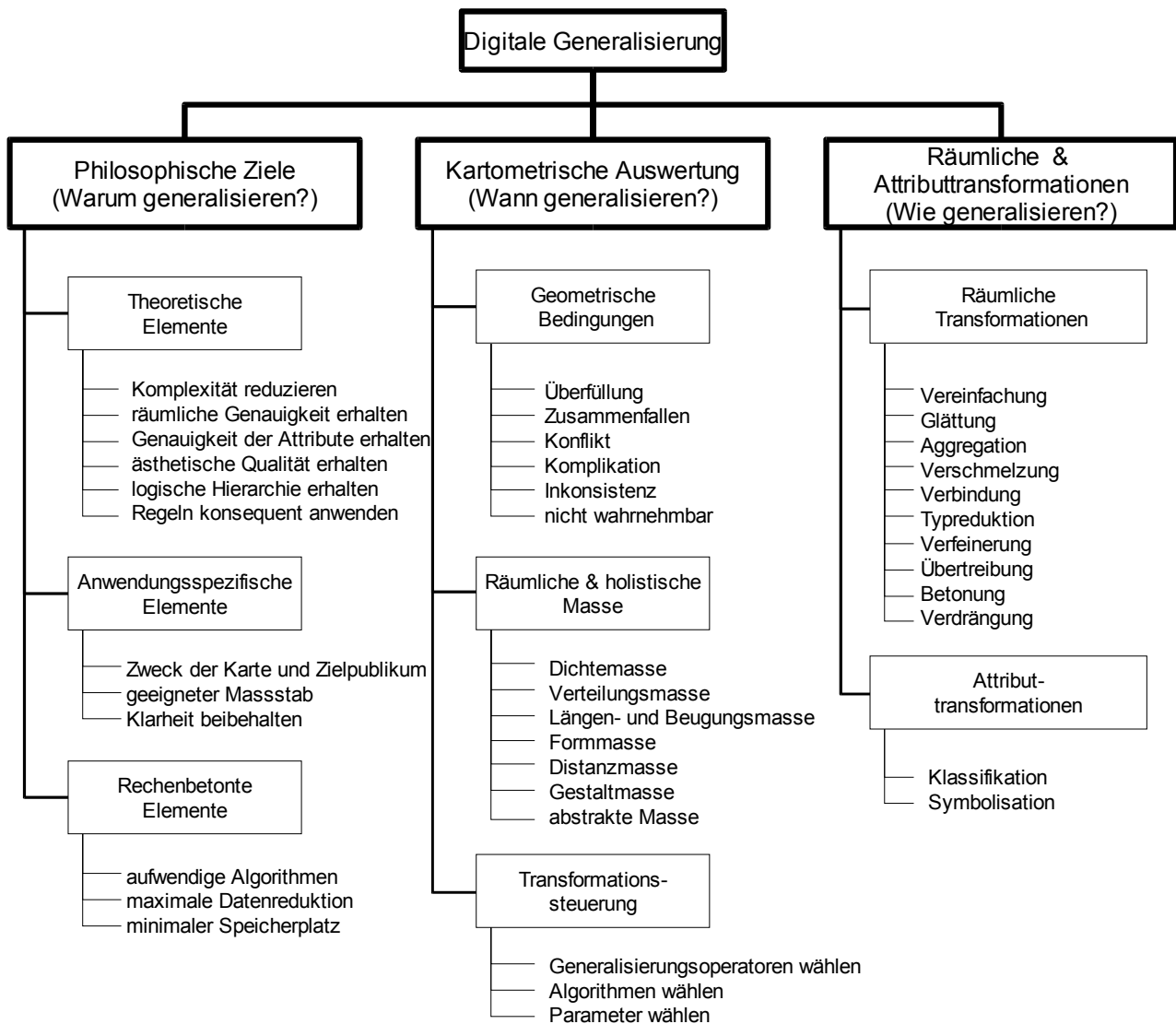


Abb. 2.3: Modell der digitalen Generalisierung nach McMaster und Shea (1992: 68).

Einen Überblick über weitere Modelle der Generalisierung gibt McMaster (1991).

2.2.3 Operatoren der Generalisierung

Der Gesamtprozess der Generalisierung wird oft in einzelne Teilprozesse, die meistens als Operatoren bezeichnet werden, aufgespalten. Im digitalen Bereich ist eine solche Unterteilung noch wichtiger als in der herkömmlichen Kartographie, da sie bei der Identifikation der Grundbestandteile der Generalisierung hilft und die Entwicklung von Lösungen für die einzelnen Teilprobleme

ermöglicht (Weibel 1997: 109). Allerdings besteht in der Literatur keine Einigkeit, wieviel und welche Operatoren nötig sind und wie diese bezeichnet werden sollen. Definitionen der Operatoren sind oft unterschiedlich oder fehlen teilweise ganz. Gleiche Begriffe werden unterschiedlich definiert oder es werden verschiedene Bezeichnungen für die gleiche Operation benutzt (Rieger und Coulson 1993: 69). Einige Begründungen für diese Differenzen liefert Ruas (2002: 79).

Diese Unstimmigkeiten aufzulösen, kann nicht das Ziel dieser Arbeit sein. Wichtig ist auf jeden Fall, dass die verwendeten Begriffe klar definiert sind und konsequent verwendet werden. Abgestützt auf die Typologie von Tab. 2.1 werden im Folgenden die Bedeutungen der für diese Arbeit wesentlichen Operatoren festgelegt.

		Traditionelle Operatoren	Digitale Operatoren
Semantische Generalisierung		Klassifikation	Thematische Auswahl Thematische Aggregation
Geometrische Generalisierung	Einzelobjekte	Vereinfachung	Glätten freie Vereinfachung
		Typreduktion	Typreduktion
		Wesentliches Betonen	Vergrößerung Betonen Ausrichtung rechter Winkel
	einzelne oder mehrere Objekte	Auswahl / Elimination	Auswahl / Elimination
		Verdrängung	Verdrängung
	mehrere Objekte (kontextabhängige Generalisierung)	Aggregation (Objekte zusammenfassen)	Verbinden Kombination
			Typisierung

Tab. 2.1: Typologie von Generalisierungsoperatoren (vereinfacht nach AGENT 1999: 7). Viele Operatoren können nur zur Generalisierung von Linien und/oder Flächen verwendet werden. Geeignet für Punktdaten sind Auswahl bzw. Elimination, Typisierung und Kombination.

- Auswahl/Elimination: Auswahl und Elimination bezeichnen denselben Vorgang aus einem unterschiedlichen Gesichtspunkt. Gemeint ist damit die Reduktion der Anzahl von Objekten einer Gruppe.
- Typisierung: Aggregation bezeichnet allgemein das Zusammenfassen von Objekten einer Klasse zu einem oder mehreren neuen Objekten. Bei Typisierung handelt es sich um einen Spezialfall von Aggregation: Eine Gruppe von Objekten wird durch eine neue, verkleinerte Gruppe wiedergegeben. Diese neue Gruppe soll ähnliche Eigenschaften bezüglich Dichte, Orientierung usw. aufweisen wie die Originalobjekte. Typisierung wird oft basierend auf einer Kombination von einfacheren Operatoren durchgeführt, z.B. Auswahl oder Kombination. Die aus der Typisierung resultierenden Objekte können eine andere Dimension haben als die Ursprünglichen. So werden in dieser Arbeit Gruppen von Punkten zu Flächen zusammengefasst.

- Kombination: Darunter versteht man das Zusammenfassen von Objekten einer Klasse zu einem Objekt mit höherer Dimension. Im Gegensatz zur Typisierung ist das Resultat einer Kombination ein einzelnes Objekt, die Eigenschaften der ursprünglichen Gruppe werden nicht berücksichtigt. Kombination kann aber als Teiloperator der Typisierung verwendet werden.

Unabhängig von der Definition der Generalisierungsoperatoren besteht eine hierarchische Beziehung zu Generalisierungsalgorithmen. Operatoren beschreiben die auszuführende Umwandlung; Algorithmen werden dann benutzt, um sie zu implementieren, wobei für einen Operator meistens mehrere Algorithmen möglich sind (Weibel 1997: 110-111). Algorithmen werden aufgeteilt in unabhängige und kontextabhängige Algorithmen (Ruas 2002: 82): Erstere werden auf einzelne Objekte angewendet ohne, dass deren Umgebung berücksichtigt wird (z.B. Vereinfachung). Kontextabhängige Algorithmen werden auf eine Gruppe von Objekten angewendet (z.B. Aggregation).

2.2.4 On-the-fly Generalisierung

Mit dem starken Wachstum des Internets hat auch die Nutzung von Karten im Web zugenommen. Entsprechende Untersuchungen zeigen einen starken exponentiellen Anstieg (Peterson 2003: 9). Mit diesem Wachstum sind auch neue Bedürfnisse entstanden: Nutzer und Nutzerinnen möchten selbst bestimmen, was sie wie dargestellt haben wollen. Zusätzlich eröffnen neue Technologien Interaktionsmöglichkeiten, die mit statischen Karten nicht erreichbar sind. Entsprechend müssen Karten innert kürzester Zeit dynamisch erzeugt werden können. Dazu sind Änderungen des Massstabes in Echtzeit nötig, was eine on-the-fly Generalisierung erfordert, die ebenfalls in Echtzeit erfolgt (Weibel et al 2002: 319).

On-the-fly Generalisierung kann als Teilgebiet von on-demand mapping angesehen werden (Weibel et al. 2002: 320). Dieses lässt sich definieren als Produktion eines kartographischen Produkts mit entsprechendem Massstab und Zweck auf Anfrage eines Benutzers (Cecconi 2003: 20). Eine solche Karte kann durchaus auch auf Papier gedruckt sein; wesentlich ist, dass die Kartenproduktion den Bedürfnissen der Nutzerin folgt, die den Inhalt der Karte, dessen Visualisierung und den Zeitpunkt der Kartenerstellung bestimmt. Die on-the-fly Generalisierung verfolgt den gleichen Zweck, wobei hier die Karte in Echtzeit berechnet und an die Benutzerin geliefert wird. Charakteristisch ist, dass das Ergebnis der Generalisierung nicht gespeichert wird, dass also eine temporäre Generalisierung aus einem detaillierten Datenbestand erzeugt wird (van Oosterom 1995: 120). Ein Massstab ist dabei nicht vorgegeben. Die Berücksichtigung der Präferenzen eines Nutzers und die Verfügbarkeit des Resultats innert weniger Sekunden sind weitere Merkmale (Weibel et al. 2002: 320).

Um eine Echtzeit-Generalisierung zu implementieren, kann entweder ein repräsentationsorientierter

oder ein prozessorientierter Zugang gewählt werden. Ersterer basiert auf einer multiplen Repräsentation, die in einer Multiskalendatenbank gespeichert ist. Diese Lösung ist technisch einfacher, aber auch unflexibler, da nur die vorhandenen Detailstufen verwendet werden können. Die zweite Möglichkeit beruht auf einer Datenbank in einem Massstab, von der aus die gewünschten Karten in Echtzeit generalisiert werden. Da die Zeit ein kritischer Faktor ist, müssen die verwendeten Algorithmen schnell sein und/oder durch Attribute und vorberechnete Strukturen unterstützt werden (Weibel et al. 2002: 325). Beispiele dafür sind der GAP-, BLG- und Reactive-tree (van Oosterom 1995). Dieser Ansatz ist flexibler, da man nicht auf die vordefinierten Detailstufen beschränkt ist, stellt aber höhere technische Anforderungen. Die besten Resultate lassen sich wahrscheinlich mit einer Kombination der beiden Ansätze realisieren (vgl. Cecconi 2003).

Die Vorteile des Einsatzes einer Echtzeitgeneralisierung in der Internetkartographie sind nach Weibel et al. (2002: 323), dass Karten besser an die spezifischen Wünsche der Benutzer angepasst werden können und dass die Herstellung der gewünschten Karte beschleunigt wird. Diese Vorteile gelten auch bei einer Verwendung im Rahmen von Location Based Services, da dort die Adaption an Aufenthaltsort und Bedürfnisse der Nutzerin zentral ist. Ein Nachteil der on-the-fly Generalisierung soll hier jedoch nicht unerwähnt bleiben: Die automatisch erzeugten Karten können vor der „Publikation“, d.h. der Anzeige beim Benutzer, nicht auf ihre kartographische Qualität geprüft werden.

2.3 Typologie von Punktdaten

Es gibt diverse Möglichkeiten, um eine Klassifikation von Punktdaten vorzunehmen. Die hier vorgestellte Einteilung (Tab. 2.2) geschieht aus der Sicht der Generalisierung und basiert auf den unterschiedlichen Möglichkeiten graphischer Darstellung der unterschiedlichen Typen. Diese Unterschiede in der Darstellung bedingen auch unterschiedliche Methoden, um die Daten zu generalisieren. Die gezeigte Typologie ist vom Massstab abhängig, in dem die Daten betrachtet werden.

Eine erster Typ von Punktdaten bezeichnet Objekte, deren kartographische Darstellung nur von der absoluten Lage des jeweiligen Punktes abhängig ist. Diese Daten werden im Folgenden als ortsfeste Daten bezeichnet. Ein anderer Typ von Punktdaten kann zwar ebenfalls wie die ortsfesten Daten dargestellt werden, es ist aber auch möglich, Punkte zu anderen Objekten, wie Linien oder Flächen, zusammenzufassen und sie so darzustellen. Bei diesen nicht ortsfesten Punktdaten ist nicht allein die absolute Lage im Raum entscheidend für die Informationsvermittlung, sondern auch die relative

Lage, d.h. die Nachbarschaft zu anderen Objekten. Wichtig ist bei diesem Typ die räumliche Verteilung, z.B. unterschiedliche Dichten, oder Muster.

	Ortsfeste Punktdaten	Nicht ortsfeste Punktdaten
Darstellung abhängig von	absoluter Lage	absoluter und relativer Lage
Generalisierung	Auswahl / Elimination	Auswahl / Elimination Aggregation (Kombination und Typisierung)
Beispiele	- Points of Interest - Siedlungen als Punkte - Gebäude bis 1:25'000	- Tierbeobachtungsdaten - Signaturen in thematischen Karten - Gebäude in kleinen Massstäben

Tab. 2.2: Typologie von Punktdaten.

2.3.1 Ortsfeste Punktdaten

Ein Beispiel für ortsfeste Punktdaten ist die Darstellung von Points of Interest. Ein solcher POI kann je nach Interesse des Kartenbenutzers z.B. eine Haltestelle des öffentlichen Verkehrs, eine Sehenswürdigkeit oder ein Restaurant sein. Weitere Beispiele für diesen Typ stellen Gebäude in grossen Massstäben, bis ca. 1:25'000, oder Siedlungen, die in kleinen Massstäben nur als Punkte wiedergegeben werden, dar.

Die hier wichtigste Eigenschaft eines solchen Punktes ist, dass eine Darstellung nur an der Originalposition möglich ist. Falls dies wegen dem Kartenmassstab nicht in Frage kommt, muss auf die Darstellung dieses Punktes verzichtet werden. Es ist auch nicht möglich, verschiedene Punkte zusammenzufassen und durch ein einzelnes Objekt wiederzugeben.

Für diese Art von Daten kann der Generalisierungsoperator Auswahl/Elimination verwendet werden. Die Auswahl wird wahrscheinlich auf der Basis einer Hierarchie innerhalb des Datensatzes durchgeführt. Diese auf Attributen basierende Hierarchie kann explizit oder implizit gegeben sein: Im Fall von Siedlungen ist die Rangfolge der Objekte explizit durch die Einwohnerzahl oder durch ein anderes Mass, wie wirtschaftliche oder verkehrstechnische Bedeutung gegeben. Bei anderen Punkten muss eine solche Hierarchie zuerst erstellt werden, was im Rahmen von Location Based Services durch den Einbezug von Benutzerposition und -interessen geschehen kann. Restaurants können beispielsweise nach Vorlieben und Distanz zur Nutzerin gewichtet werden.

2.3.2 Nicht ortsfeste Punktdaten

Beispiele für diesen Typ von Punktdaten können z.B. die beobachteten Positionen von Tieren sein, wie sie für diese Arbeit Verwendung fanden. Ein weiteres Beispiel sind Punktdichtekarten. In thematischen Karten finden sich auch oft Signaturen für lokale Diskreta, die man ebenfalls zu diesem Typ von Punktdaten zählen kann. Ähnlich können auch Gebäude in kleineren Massstäben behandelt

werden, auch wenn es sich dabei genau genommen nicht um Punktdaten handelt.

Bei diesen Daten spielen einzelne Objekte eine weniger wichtige Rolle als bei den ortsgebundenen Daten. Wichtiger ist die räumliche Verteilung aller Punkte und sich daraus möglicherweise ergebende Muster. Es ist möglich, eine Gruppe von Punkten durch ein oder mehrere Objekte zu ersetzen oder die räumliche Verteilung durch eine flächenhafte Darstellung wiederzugeben. Wichtig ist, dass die räumliche Verteilung erhalten bleibt.

Mit diesem Typ Daten kann Auswahl bzw. Elimination verwendet werden. Es ist aber auch möglich, Aggregation zu verwenden. Bei der Aggregation steht der Operator Typisierung im Vordergrund, bei dem die Eigenschaften der zusammengefassten Objektgruppen erhalten bleiben.

2.4 Geeignete Generalisierungsmethoden für Punktdaten

Betrachtet man eine Auflistung von Algorithmen zur automatischen Generalisierung, fällt auf, dass die Zielobjekte meist linear, manchmal flächenhaft sind. Methoden zur Generalisierung von Punktdaten finden sich dagegen nur wenige. Der Grund dafür ist laut McMaster und Shea (1992: 71), dass bis zu 80 Prozent der Information in einer digitalen Karte aus Linien besteht. In einer entsprechenden Liste des AGENT-Projekts (1999:14) findet sich beim Operator Auswahl nur das Wurzelgesetz von Töpfer (1974), das für alle Typen von Objekte angewendet werden kann. Dieses Gesetz liefert eine geeignete Anzahl von Objekten, die in einem bestimmten Kartenmassstab dargestellt werden können; es gibt aber keine Hilfe bei der Auswahl dieser Objekte.

2.4.1 Auswahl/Elimination

Existiert innerhalb der Daten bereits eine Hierarchie oder kann eine solche erstellt werden, wird sich die Selektion auf diese Rangfolge abstützen müssen. Eine Möglichkeit ist die Einteilung der Objekte in Kategorien aufgrund der Hierarchie. Anschliessend wird eine Auswahl durchgeführt, wobei mit abnehmender Bedeutung immer mehr Punkte eliminiert werden. Die Gesamtzahl der dargestellten Objekte kann mit dem Wurzelgesetz bestimmt werden. Bei der genauen Ausgestaltung dieses Verfahrens stellen sich Fragen: Wie soll die Anzahl Kategorien bestimmt werden? Wie wird der Anteil von zu eliminierenden Objekten in den einzelnen Kategorien bestimmt?

Eine andere Methode könnte an das automatische Label Placement angelehnt werden, das zur Platzierung von Kartenschrift verwendet wird. Entsprechend der Rangfolge werden Objekte platziert, wobei immer überprüft wird, dass sich Punkte nicht überdecken. Ist das der Fall, wird das Objekt eliminiert.

2.4.2 Aggregation (Typisierung und Kombination)

Unter dem Operator Aggregation finden sich in der Algorithmenliste von AGENT (1999:14) unter Typisierung und Verbinden neben Methoden, die für Linien und Flächen geeignet sind, nur speziell für die Gebäudegeneralisierung geeignete Verfahren (vgl. beispielsweise Regnauld 1997). Eine Reihe von Algorithmen, mit denen sich Punkte zu Flächen zusammenfassen lassen, finden sich beim Operator Kombination: Neben zwei Algorithmen der Firma LaserScan werden graphtheoretische Methoden zur Clustererkennung und die Aggregation von Punkten mittels Delaunay-Triangulation (DeLucia und Black 1987) aufgezählt. Die letzte Möglichkeit ist auch die einzige zur Aggregation von Punkten, die McMaster und Shea (1992: 96-98) in ihrem Buch beschreiben. Es muss beachtet werden, dass diese Methoden Gruppen von Objekten zusammenfassen, aber keine Gewähr für die Erhaltung der Eigenschaften dieser Gruppen geben.

De Berg et al. (2004) beschreiben die Vereinfachung von Punktkarten und diskutieren Möglichkeiten, wie die Qualität der Vereinfachung gemessen werden kann. Sie ersetzen die Originalpunktmenge P durch eine neue Menge Q von Punkten, die eine ähnliche Verteilung wie P aufweist. Zur Konstruktion der vereinfachten Punktmenge verwenden sie zwei Typen von Heuristiken: Iterative und Clustering-Algorithmen. Erstere beginnen mit einer Zufallslösung und versuchen diese schrittweise zu optimieren. Verwendet werden dazu Optimierungsalgorithmen wie beispielsweise Simulated Annealing. Die verwendeten Clustering-Algorithmen (im weiteren Sinn) teilen die Punkte in eine Anzahl Gruppen auf und ersetzen die Gruppe dann durch einen oder mehrere Repräsentanten. Von de Berg et al. (2004) wird beispielsweise ein Quadtree verwendet.

Die vorliegende Arbeit konzentriert sich auf die Anwendung von Verfahren der Clusteranalyse, um den Operator Typisierung für Punktdaten in Echtzeit durchzuführen.

2.5 Clusteranalyse

Die Clusteranalyse stammt aus der multivariaten Statistik und umfasst Verfahren zur Gruppenbildung. Ihr Ziel ist es, eine Vielzahl von Objekten aufgrund ihrer Eigenschaften zu Gruppen zusammenzufassen. *„Die Mitglieder einer Gruppe sollen dabei eine weitgehend verwandte Eigenschaftsstruktur aufweisen; d.h. sich möglichst ähnlich sein. Zwischen den Gruppen sollen demgegenüber so gut wie keine Ähnlichkeiten bestehen“* (Backhaus et al. 1996: 262).

Der Ablauf der Gruppenbildung erfolgt in zwei Schritten:

1. Wahl des Ähnlichkeitsmasses (auch Proximitätsmass): Es wird versucht, durch einen Zahlenwert die Unterschiede bzw. Übereinstimmung zwischen den Eigenschaften zweier

Objekte auszudrücken. Die berechnete Zahl symbolisiert die Ähnlichkeit der Objekte hinsichtlich der untersuchten Merkmale (Backhaus et al. 1996: 262).

2. Anwendung einer Methode, welche die Daten aufgrund des Ähnlichkeitsmasses in Gruppen einteilt.

Die Klassifikation von ähnlichen Objekten in Gruppen ist eine wichtige menschliche Aktivität und spielt auch in der Wissenschaft eine bedeutende Rolle. Verfahren der Clusteranalyse werden deshalb in den verschiedensten Gebieten, wie z.B. Medizin, Biologie, Geschichte und Wirtschaft, eingesetzt (Kaufman und Rousseeuw 1990: 1). Ein wichtiges Anwendungsgebiet ist Data Mining, für das Berkhin (2002:4) folgende Einsatzgebiete beschreibt: „*information retrieval and text mining, spatial database applications, [...], sequence and heterogeneous data analysis, Web applications, DNA analysis in computational biology.*“ Das Ziel ist es, ohne Vorkenntnisse über den eventuell sehr umfangreichen Datenbestand versteckte Zusammenhänge in diesem zu finden.

In der Geographie wird die Clusteranalyse vor allem zum Erkennen von Raumtypen und Regionen eingesetzt (Bahrenberg, Giese, Nipper 1992; O'Sullivan, Unwin 2003: 335).

2.5.1 Ähnlichkeitsmasse

Weisen die betrachteten Objekte ausschliesslich Eigenschaften auf *metrischem Skalenniveau* (Intervall- oder Rationalskala) auf, können sie als Punkte in einem Variablenraum betrachtet werden. Ein Ähnlichkeitskriterium ist dann der Abstand zwischen zwei Punkten: Je näher sie zusammenliegen, desto ähnlicher sind sie. Als Mass für die Nähe der Punkte werden gewöhnlich sogenannte Distanzmasse verwendet, die eigentlich Unähnlichkeitsmasse darstellen (Bahrenberg, Giese, Nipper 1992: 281). Je grösser nämlich ihr Wert, desto unähnlicher sind sich die Punkte.

Ein anschauliches Beispiel ist die euklidische Distanz. In Erweiterung des zweidimensionalen Falls ist sie für zwei Objekte j, k bei m Variablen folgendermassen definiert:

$$d_{jk} = \sqrt{\sum_{i=1}^m (x_{ij} - x_{ik})^2}$$

Eine andere Möglichkeit ist die City-Block oder Manhattan-Distanz:

$$d_{jk} = \sum_{i=1}^m |x_{ij} - x_{ik}|$$

Sie entspricht der Länge des Weges zwischen j und k , wenn man sich entlang der Koordinatenachsen bewegt (Weglänge mit rechtwinkligem Strassensystem).

Bei diesen beiden Distanzen handelt es sich um Spezialfälle der sogenannten Minkowski-Metriken

$$d_{jk} = \sqrt[r]{\sum_{i=1}^m |x_{ij} - x_{ik}|^r}$$

mit $r = 1$ (Manhattan-Distanz) und $r = 2$ (Euklidische Distanz). Die Wahl des Distanzmasses beeinflusst die Ähnlichkeitsreihenfolge der Untersuchungsobjekte. Sollen Minkowski-Metriken angewendet werden, ist darauf zu achten, dass den Variablen vergleichbare Masseinheiten zugrunde liegen (Backhaus et al. 1996: 275f).

Neben den Distanzmassen können auch direkte Ähnlichkeitsmasse, z.B. verschiedene Korrelationskoeffizienten, verwendet werden. Nach Backhaus et al. (1996: 278) sind diese vor allem geeignet, wenn der primäre Ähnlichkeitsaspekt im Gleichlauf zweier Profile zu sehen ist, unabhängig davon, auf welchem Niveau die Objekte liegen. Für Ähnlichkeitsmasse, die sich für nominal- und ordinalskalierte Variablen eignen sowie die Behandlung von gemischten Variablen sei an dieser Stelle auf die Literatur verwiesen.

2.5.2 Clusteringverfahren

In der Literatur existiert eine Vielzahl von Algorithmen, wobei die Mehrheit entweder zu den hierarchischen oder den partitionierenden Verfahren gehört (Kaufman und Rousseeuw 1990: 37f). Auf anderen Überlegungen beruht das graphbasierte Clustering, welches hier ebenfalls diskutiert wird.

2.5.2.1 Partitionierendes Clustering

Partitionierende Verfahren gehen von einer vorgegebenen Gruppeneinteilung aus und versuchen diese durch Verlagerung von Objekten in andere Gruppen schrittweise zu verbessern (Backhaus et al. 1996: 282). Die Unterschiede zwischen den Verfahren liegen darin, wie die Verbesserung gemessen und nach welchen Regeln die Gruppenzugehörigkeit der Objekte verändert wird. Ein partitionierendes Verfahren teilt die Daten in k Cluster ein, wobei k meist vom Anwender vorgegeben wird. Dabei muss jede Gruppe mindestens ein Objekt enthalten und ein Objekt muss genau zu einer Gruppe gehören (Kaufman und Rousseeuw 1990: 38). Im Gegensatz zu den hierarchischen Verfahren, bei denen ein einmal gefundenes Cluster nicht mehr verändert werden kann, erlauben es partitionierende Methoden, Elemente beliebig zwischen Gruppen zu verschieben. Sie besitzen also eine grössere Flexibilität; trotzdem haben sie bei praktischen Anwendungen aus folgenden Gründen nur wenig Verbreitung gefunden (Backhaus et al. 1996: 284):

- Die Ergebnisse werden stark durch die der Umordnung zugrunde liegende Zielfunktion beeinflusst.
- Die Wahl der Startpartition, welche die Ergebnisse ebenfalls beeinflusst, ist häufig subjektiv

(oder zufällig) begründet.

- Vielfach gelangt man nur zu lokalen statt globalen Optima und eine Berechnung aller möglichen Varianten ist normalerweise nicht möglich.

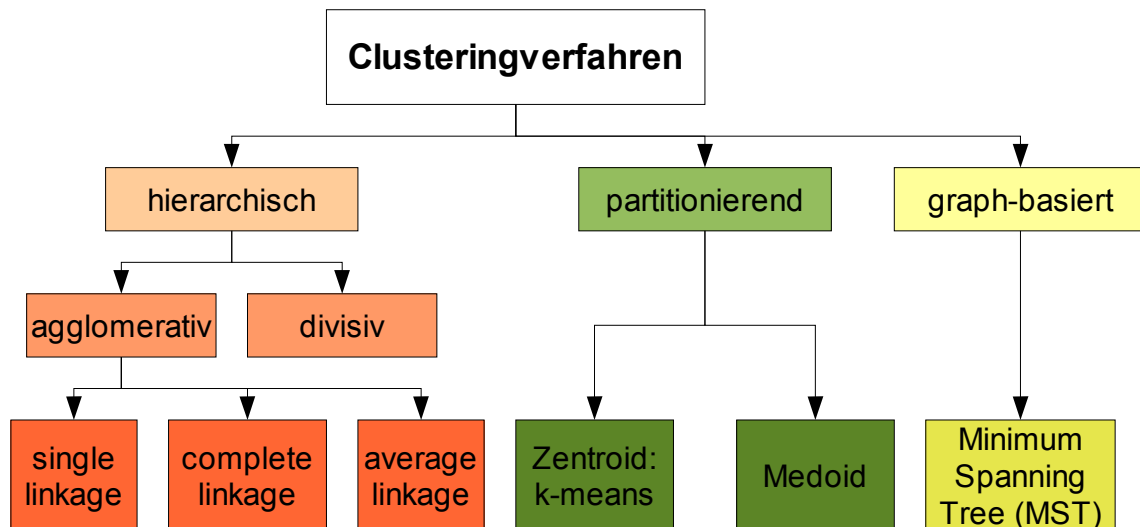


Abb. 2.4: Typologie von Clusteringverfahren mit ausgewählten Algorithmen.

Innerhalb der partitionierenden Verfahren kann man Zentroid- und Medoid-basierte Methoden unterscheiden. Bei Zentroid-basierten Verfahren, zu denen z.B. der k-Means-Algorithmus gehört, sind die Abstände zwischen einem Objekt und den Clusterzentren (Schwerpunkt) entscheidend für die Unterteilung. Medoid-basierte Verfahren benutzen ein Objekt als Clusterzentrum, womit die Anfälligkeit gegenüber Ausreißern verringert wird. Es wird versucht, die Abstände zwischen den Objekten und diesem Medoid zu minimieren.

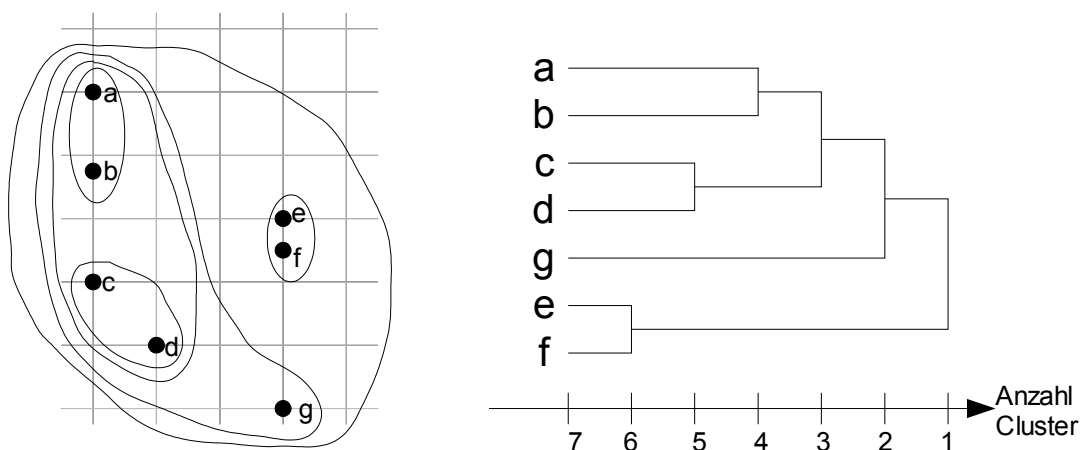


Abb. 2.5: Hierarchisches Clustering (Single Linkage) mit Dendrogramm.

2.5.2.2 Hierarchisches Clustering

Hierarchische Algorithmen konstruieren nicht eine einzige Unterteilung in k Cluster sondern eine verschachtelte Hierarchie von Gruppeneinteilungen. Unterschieden werden agglomerative und

divisive Verfahren (Kaufman und Rousseeuw 1990: 44).

Agglomerative Techniken beginnen mit einem Cluster für jedes Objekt im Datensatz. Bei jedem Schritt werden die zwei ähnlichsten Cluster zu einem neuen zusammengefasst und die Gesamtzahl der Cluster verkleinert sich um eins. Das Verfahren wird fortgesetzt bis eine vorgegebene Anzahl von Clustern erreicht ist, bis nur noch ein Cluster übrigbleibt oder der Abstand zwischen zwei zu vereinigenden Clustern eine vorgegebene Grösse überschreitet. Divisive Techniken funktionieren genau umgekehrt: Sie beginnen mit einem Cluster, der alle Objekte umfasst und unterteilen diesen schrittweise.

Es stellt sich die Frage, durch welche(s) seiner Elemente ein Cluster repräsentiert werden soll, um die *Ähnlichkeit zwischen zwei Clustern* zu bestimmen. Bei Linkage-Verfahren sind alle Objekte, die zu einem Cluster gehören, Kandidaten für die Distanzberechnung, während andere Verfahren nur auf eine Stichprobe der enthaltenen Elemente zurückgreifen. Je nach Algorithmus wird aus diesen Kandidaten einer oder mehrere ausgewählt, um die Distanz zwischen zwei Clustern zu berechnen:

- Single Linkage: Massgebend ist die kürzeste Distanz zwischen den Clustern, d.h. der Abstand zwischen den zwei nächsten Punkten (nächste Nachbarn) aus beiden Clustern.
- Complete Linkage: Der massgebende Abstand ist die längste Distanz zwischen den Clustern, d.h. der Abstand zwischen den zwei entferntesten Punkten aus beiden Clustern.
- Average Linkage: Für den massgebenden Abstand wird der Durchschnitt aus allen paarweisen Distanzen zwischen den zwei Clustern berechnet.

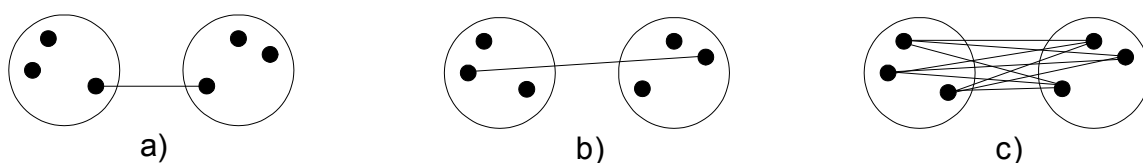


Abb. 2.6: Darstellung einiger Definitionen von Ähnlichkeit zwischen zwei Clustern: a) Single Linkage, b) Complete Linkage, c) Average Linkage (nach Kaufman und Rousseeuw 1990: 47).

Neben diesen drei Beispielen existieren weitere Möglichkeiten, wie die Ähnlichkeit zwischen zwei Clustern bestimmt werden kann. Die Ergebnisse eines hierarchischen Clusterings werden oft als Baumstruktur, auch Dendrogramm genannt, dargestellt (Abb. 2.5). Die Wurzel des Baumes wird von einem Cluster gebildet, das alle Objekte beinhaltet. Die Blätter des Baumes repräsentieren Cluster, die nur aus einem einzigen Objekt bestehen.

Mit hierarchischen Verfahren werden die Daten auf eine vollständig andere Weise beschrieben als mit partitionierenden Algorithmen, welche die bestmögliche Unterteilung für eine vorgegebene Anzahl Cluster suchen. Das Gewicht liegt auf dem schrittweisen Vorgehen, welches gewissermassen den

Verwandschaftsgrad zwischen Objekten aufzeigt. Die Wahl einer geeigneten Unterteilung bzw. Anzahl Cluster wird oft dem Anwender überlassen.

Eine Schwäche der hierarchischen Verfahren ist, dass eine einmal gemachte Unterteilung nicht mehr rückgängig gemacht werden kann. Demgegenüber steht unter anderem der Vorteil, dass verschiedene Distanz- und Ähnlichkeitsmasse verwendet werden können und sich die Verfahren folglich für Variablen auf verschiedenen Skalenniveaus eignen (Backhaus et al. 1996: 287).

2.5.2.3 Graphbasiertes Clustering

Ein Graph ist ein Paar $G = (V, E)$ disjunkter Mengen, wobei man die Elemente von V als Knoten (oder Ecken), die Elemente von E als Kanten bezeichnet (Diestel, 2000:2). Für die graphische Darstellung eines Graphen zeichnet man die Knoten als Punkte und die Kanten als Linien zwischen den Punkten. In einem ungerichteten Graph verbindet eine Kante zwei Knoten ohne eine Richtung vorzugeben, während in einem gerichteten Graph die Knoten in einer bestimmten Richtung verbunden sind. Ein Graph kann auch gewichtet sein: In diesem Fall wird jeder Kante ein Gewicht zugeordnet, z.B. Kosten oder Distanz.

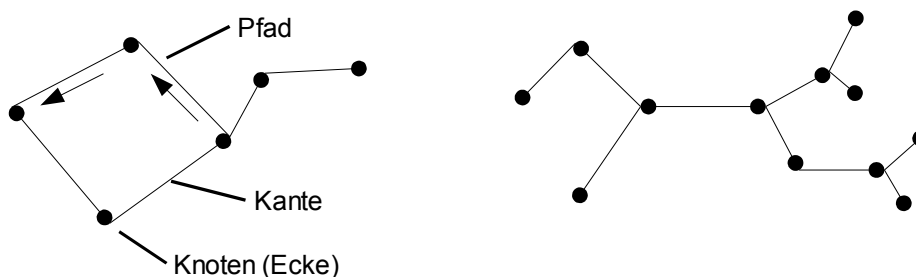


Abb. 2.7: Darstellung von Begriffen der Graphentheorie. Links ein ungerichteter Graph mit einem Zyklus, rechts ein Baum.

Ein Pfad ist eine Liste von aufeinanderfolgenden Knoten, welche durch Kanten verbunden sind. Hat ein solcher Pfad den gleichen Anfangs- und Endknoten wird er auch als Zyklus bezeichnet. Einen Graphen ohne Zyklen nennt man Baum und eine Kombination solcher Graphen kann als Wald bezeichnet werden.

Um ein auf Graphen basierendes Clustering durchzuführen, wird zuerst ein Nachbarschaftsgraph, z.B. der Minimum Spanning Tree, aus den Originaldaten berechnet. In diesem Graph werden anschliessend die nach einem gewählten Kriterium längsten Kanten eliminiert. Das Ergebnis der Kantenelimination ist ein Wald, in dem jeder Baum einen Cluster darstellt (Anders 2002: 35).

Hierarchisches Clustering arbeitet häufig auf der Basis einer Ähnlichkeitsmatrix, wobei jedes Element der Matrix die Ähnlichkeit zwischen zwei Objekten beschreibt. Diese Matrix kann als Graph betrachtet werden: Die Knoten repräsentieren die Objekte, die Kanten und ihre Gewichte sind Paare

von Punkten und die entsprechenden Elemente der Matrix. Auf diese Weise sind hierarchische und graphbasierte Verfahren verbunden (Berkhin 2002: 7).

2.5.3 Clustering und Generalisierung

Schwerpunktmässig wird in dieser Arbeit die Verwendung von Clusteringverfahren zur kartographischen Generalisierung untersucht. Eine Benutzung dieser Techniken ist auch für die Modellgeneralisierung möglich, worauf kurz in Abschnitt 2.5.3.2 eingegangen wird.

Die Wahl eines bestimmten Clusteringverfahrens hängt von den Datentypen und dem Verwendungszweck ab. Für viele Aufgaben können verschiedene Verfahren angewendet werden und eine Auswahl aufgrund theoretischer Überlegungen ist schwierig. Oft ist es deshalb sinnvoll, verschiedene Methoden zu probieren und die Ergebnisse zu analysieren (Kaufman und Rousseeuw 1990: 37). Aus diesem Grund werden für diese Arbeit eine Auswahl von Algorithmen, die auf verschiedenen Prinzipien basieren, verwendet.

2.5.3.1 Kartographische Generalisierung

Bei der kartographischen Generalisierung kann Clustering verwendet werden, um den Generalisierungsoperator Typisierung für Punktdaten zu realisieren. Wie in Kap. 2.4.2 gezeigt, werden die Punkte zu Gruppen zusammengefasst, für die anschliessend ein oder mehrere Repräsentanten bestimmt werden. Das Resultat ist eine Menge von Objekten, die eine möglichst ähnliche Verteilung wie die Originalpunkte aufweisen sollten. Clustering ist also ein kontextabhängiges Verfahren der Generalisierung. Genauer betrachtet führen Clusteringverfahren bei der kartographischen Generalisierung eine Strukturerkennung bzw. eine kartometrische Evaluation durch (vgl. Kap. 2.2.2); ihr Ziel ist es, Muster in der Punktverteilung zu entdecken. Anschliessend wird ein Prozess bestimmt und ausgeführt, welcher, basierend auf den gefundenen Gruppen, zu einer Vereinfachung der ursprünglichen Punktmenge führt. Zum Beispiel kann man die Punkte durch den Generalisierungsoperator Kombination zu einer Fläche zusammenfassen.

Für die kartographische Generalisierung werden in dieser Arbeit als Variablen nur die x- und y-Koordinaten der Punkte benutzt. Die Clusteringverfahren werden also auf eine bivariate Verteilung angewendet. Als (Un-)Ähnlichkeitsmass wurde die euklidische Distanz verwendet.

2.5.3.2 Modellgeneralisierung

Eine Anwendung von Clusteringverfahren in der Geographie zielt auf die Identifikation von Raumtypen und Regionen ab (Bahrenberg, Giese, Nipper 1992; O'Sullivan, Unwin 2003: 335). Dabei wird versucht, mögliche Klassifikationen in statistischen Daten zu finden. Man führt also eine Modellgeneralisierung zu analytischen Zwecken durch.

Interessanter ist eine Modellgeneralisierung als Vorbereitung für eine kartographische Echtzeitgeneralisierung. Wie in Kap. 2.2.4 ausgeführt, müssen Algorithmen für eine on-the-fly Generalisierung schnell sein und/oder durch Attribute der Daten und geeignete Datenstrukturen unterstützt werden. Das Ziel der Modellgeneralisierung ist dann die Anreicherung der Daten mit entsprechenden Informationen. Algorithmen, die eine hierarchische Datenstruktur ausnutzen, eignen sich deshalb besonders für eine Echtzeitgeneralisierung. Ein Beispiel dafür ist der Douglas-Peucker-Algorithmus (Douglas und Peucker 1973), dessen Ergebnisse im BLG-tree (van Oosterom 1995: 125) gespeichert werden können. Der Algorithmus muss in der Folge nicht mehr für jede Liniengeneralisierung ausgeführt werden. Es stellt sich nun die Frage, ob die hierarchischen Clusteringverfahren bzw. das resultierende Dendrogramm als hierarchische Struktur, ebenfalls geeignet sind für ein pre-processing. In diesem Fall könnte das Ausführen der aufwendigen hierarchischen Verfahren bei jeder Kartenbereitstellung vermieden werden.

2.5.4 Vorstellung der verwendeten Algorithmen

2.5.4.1 Hierarchisches Clustering mit Single Linkage

Das Single Linkage Verfahren wurde bereits oben bei der Vorstellung der hierarchischen Verfahren beschrieben. Das Ergebnis eines hierarchischen Clusterings ist normalerweise ein Dendrogramm, das graphisch den Agglomerationsprozess veranschaulicht. Bei der Verwendung zur kartographischen Generalisierung liegt das Interesse aber weniger auf dem Prozess als auf den resultierenden Clustern auf einer ausgewählten Ebene der Agglomeration. Deshalb wurde für diese Arbeit der Algorithmus so implementiert, dass eine Anzahl k von Clustern als Abbruchkriterium angegeben werden muss. Das Resultat des Clusterings sind dann k Cluster. Der Bestimmung einer geeigneten Grösse von k ist Kapitel 2.5.5 gewidmet.

Das Single Linkage Verfahren neigt zur Bildung vieler kleiner und weniger grosser Gruppen. Es wird deshalb auch als kontrahierendes Verfahren bezeichnet. Da eine einzige Verbindung (single link) entscheidet, welche Cluster vereinigt werden, neigt diese Methode zur Kettenbildung, d.h. es entstehen langgezogene Cluster (Backhaus et al. 1996: 290). Der Single Linkage Algorithmus besitzt die Laufzeitverhalten $O(n^2)$ (Berkhin 2002: 8).

2.5.4.2 Hierarchisches Clustering mit Complete Linkage

Das Verfahren mit Complete Linkage wurde ebenfalls bereits vorgestellt. Ein Unterschied zum Single Linkage Verfahren ist, dass Complete Linkage eher kompakte, in sich homogene Gruppen bildet, da es die Ähnlichkeit der unähnlichsten Paare maximiert (Bahrenberg, Giese, Nipper 1992: 286). Ein solches Verfahren, das etwa gleich grosse Gruppen ermittelt, wird als dilatierend

bezeichnet.

2.5.4.3 Maschenvereinfachung

Wegen dem grossen Aufwand der Linkage Verfahren wurde eine Möglichkeit gesucht, diesen zu minimieren. Ein Mittel dazu ist die Verwendung eines Repräsentanten für jeden Cluster: Damit entfällt das Suchen der kürzesten Distanz zwischen zwei Clustern, bei der alle Punkte aus beiden Clustern berücksichtigt werden müssen. Als Repräsentant wurde der Schwerpunkt der Cluster gewählt. Der übrige Ablauf des Algorithmus ist genau gleich wie bei den anderen hierarchischen Verfahren: Bei jedem Schritt werden die beiden Cluster, zwischen denen die kürzeste Distanz gemessen wird, vereinigt. Die Distanz zwischen den Clustern wird durch den Abstand der jeweiligen Schwerpunkte bestimmt.

Dieses Verfahren entspricht einer Maschenvereinfachung durch die Elimination von Kanten, wie sie von Cecconi (2003:126) für die Generalisierung von Gebäuden angewendet wurde.

2.5.4.4 k-means

Die k-means Methode ist möglicherweise die meistgenutzte nichthierarchische Technik (Kaufman und Rousseeuw 1990: 113). Für das Clustering muss zuerst die Anzahl k der zu bestimmenden Cluster angegeben werden. Aus allen Punkten werden dann zufällig k Punkte ausgewählt, die als erste Clusterzentren dienen. Die restlichen Punkte werden dem jeweils nächstgelegenen Clusterzentrum zugeteilt, um mit diesem ein Cluster zu bilden. Anschliessend wird von jeder Gruppe der Schwerpunkt berechnet, und dieser als neues Clusterzentrum (Zentroid) verwendet. Dann erfolgt erneut eine Verteilung aller Punkte zum nächstgelegenen Cluster. Das Verfahren wird wiederholt, bis keine Veränderungen mehr auftreten. K-means besitzt die Komplexität $O(rkN)$, wobei k die gewählte Clusterzahl und r die Anzahl Durchgänge bedeutet.

Bei diesem Algorithmus hat die Startkonfiguration, d.h. die zufällige Auswahl der Punkte zu Beginn, einen Einfluss auf das Resultat. Zusätzlich muss berücksichtigt werden, dass gewisse Werte von k keine natürlichen Cluster ergeben. Um diesen beiden Faktoren Rechnung zu tragen, wäre es nötig, den Algorithmus für verschiedene Werte von k jeweils mehrmals auszuführen, die verschiedenen Resultate auf ihre Qualität zu analysieren, um sich dann für die beste Lösung zu entscheiden. Das bedeutet aber einen hohen Rechenaufwand, der im Kontext einer Echtzeit-Generalisierung zu gross sein könnte. Weitere Nachteile dieses Algorithmus liegen darin, dass er anfällig gegenüber Ausreissern ist und nur mit metrischen Variablen verwendet werden kann.

Die Vorteile des k-means Verfahren sind, dass es einfach und geradlinig ist, und sich auf die festen Grundlagen der Varianzanalyse abstützt (Berkhin 2002:17). Zusätzlich ist der Algorithmus im

Vergleich zu den anderen verwendeten Algorithmen schnell.

2.5.4.5 Graphbasiertes Clustering mit dem Minimum Spanning Tree

Ein Baum $T \subseteq G$ heisst Spannbaum von G , wenn er ganz G aufspannt, d.h. wenn $V(T) = V(G)$ ist (Diestel 2000:14). Der aufspannende Baum beinhaltet also alle Knoten eines Graphen und verbindet diese miteinander. Der Baum beinhaltet aber nur die minimale Teilmenge der Kanten von G , die nötig ist, um alle Knoten miteinander zu verbinden. Ein Spezialfall der Spannbäume ist der Minimum Spanning Tree (MST), der minimal aufspannende Baum. Werden den Kanten eines Graphen Gewichte zugewiesen, kann die Gewichtung des gesamten Graphen als Summe der Kantengewichte berechnet werden. Der MST ist der Graph, der von allen aufspannenden Bäumen die kleinste Gesamtgewichtung aufweist.

Für diese Arbeit wurde der MST mittels des Prim Algorithmus berechnet. Dieser benötigt als Eingabe einen gewichteten Graphen, der alle Punkte verbindet. Dafür wurde der Gabriel Graph gewählt, da dieser einfach zu berechnen ist und den MST einer Menge von Punkten enthält. Als Gewicht wird naheliegenderweise die Distanz zwischen den Punkten bzw. Knoten verwendet. Der Prim Algorithmus beginnt bei einem beliebigen Knoten des Ausgangsgraphen, sucht die Kante mit dem kleinsten Gewicht und fügt diese dem MST hinzu. Man iteriert das Verfahren, bis alle Knoten des Originalgraphen im MST enthalten sind, wobei natürlich darauf zu achten ist, dass ein Knoten nicht zweimal aufgenommen wird.

Als Mittel zur Bestimmung der relativ längsten Kanten, die aus dem Baum gelöscht werden, wurde das von Regnaud (1996: 5) vorgestellte Kriterium verwendet, das für die Erkennung von Gebäudeclustern entwickelt wurde: Von jeder Kante werden Mittelwert und Standardabweichung der angrenzenden Kanten auf beiden Seiten berechnet. Für jede Seite wird dann das folgende Mass berechnet: $p_1 * \text{mittelwert} + p_2 * \text{standardabweichung}$, wobei p_1 und p_2 zwei empirisch wählbare Parameter sind. Ist eine Kante länger als dieses Mass, wird sie gelöscht.

2.5.5 Bestimmung der Anzahl Cluster

Die meisten Clusteringverfahren benötigen als Parameter die Anzahl zu findender Cluster. Bei den in dieser Arbeit verwendeten Algorithmen ist das für k-means und die hierarchischen Verfahren der Fall, während das Clustering mit dem Minimum Spanning Tree zwei andere Parameter braucht, die empirisch bestimmt werden müssen. Eine Bestimmung dieser Anzahl kann auf der Basis von Vorwissen erfolgen, sofern dieses vorhanden ist. Eine andere Variante ist das Ausprobieren von verschiedenen Werten für die Clusteranzahl k und anschliessendes manuelles Suchen der besten Lösung. Wenn das Clustering in einer Echtzeit-Generalisierung benutzt werden soll, sind beide

Lösungen nicht praktikabel, da sie ein manuelles Eingreifen erfordern.

Für ein automatisches Verfahren kann die Anzahl der Cluster ermittelt werden, indem verschiedene Parameterwerte getestet werden und anschliessend die Auswahl der besten Lösung mittels eines mathematischen Kriteriums erfolgt (Salvador und Chan 2003:1; Kaufman und Rousseeuw 1990: 38). Solche Verfahren benötigen aber viel Zeit.

Eine andere Möglichkeit ist das Finden des „Knies“ in einem Evaluationsgraphen. Als „Knie“ bezeichnet man den Punkt der Kurve mit der maximalen Krümmung (Salvador und Chan 2003:2). Für diese Arbeit wurde ein Evaluationsgraph basierend auf hierarchischem Clustering verwendet, wobei die Distanz zwischen den bei einem Schritt vereinigten Cluster entscheidend ist (vgl. Bahrenberg, Giese, Nipper 1992: 288-293). Für die Kurve berechnet man die Differenz zwischen der Vereinigungsdistanz beim aktuellen Schritt und der Distanz des vorhergehenden Schritts und trägt diese gegen die Anzahl Cluster auf. Das Maximum dieser Kurve sollte eine geeignete Anzahl Cluster liefern.

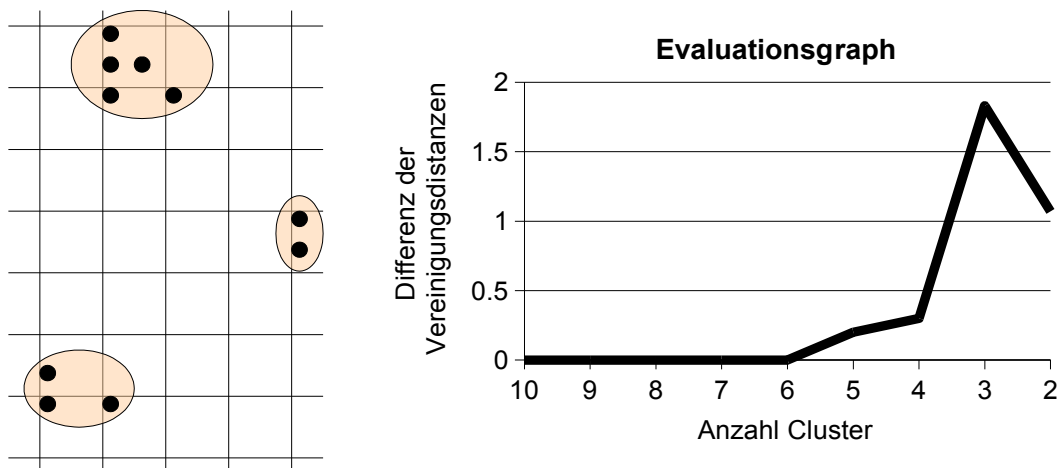


Abb. 2.8: Beispiel eines Evaluationsgraphen zur Bestimmung einer geeigneten Anzahl Cluster. Das Maximum der Kurve, welche die Differenz der Vereinigungsdistanzen auf die Clusterzahl abbildet, zeigt eine geeignete Anzahl Cluster an, in diesem Beispiel drei.

3 Eingesetzte Technologien

In diesem Kapitel werden zuerst einige Grundlagen der Interoperabilität im GIS-Bereich präsentiert. Anschliessend werden die in dieser Arbeit genutzten Technologien vorgestellt. Ausführliche Beispiele für GML, SVG und XSLT mit Erläuterungen zum Code finden sich im Anhang.

3.1 Interoperabilität

Die ISO definiert Interoperabilität als Fähigkeit zu Kommunikation, Programmausführung und Datentransfer zwischen verschiedenen funktionalen Einheiten auf eine Weise, in der ein Nutzer keine oder nur geringe Kenntnis dieser Einheiten benötigt (zitiert in Open GIS Consortium 2002b:2). Interoperabilität ist im Bereich der Informationstechnologie von ständig wachsender Bedeutung. Mit der immer grösseren Verbreitung von Computern steigt das Bedürfnis, Daten auszutauschen und andere Ressourcen, wie z.B. Programme, gemeinsam zu nutzen.

Dabei lassen sich verschiedene Stufen der Zusammenarbeit identifizieren (vgl. beispielsweise Sondheim, Gardels und Buehler 1999; Vckovski 1998: 9-33): Die einfachste Möglichkeit des Datentransfers ist der Einsatz von Konvertern, die ein Datenformat in ein anderes umwandeln. Um die Anzahl der benötigten Konversionsprogramme zu verringern, wurden Anstrengungen unternommen, spezielle Formate für den Datentransfer zu definieren. Damit müssen nicht mehr Umwandlungen zwischen allen Formaten ermöglicht werden, sondern nur noch vom Ausgangsformat in das Transferformat und von dort ins Zielformat. Ein Problem bei diesen Umwandlungen stellen jedoch die Datenmodelle der verwendeten Formate dar: Sind sie nicht gleich, kann ein Informationsverlust auftreten.

Oft wird eine weitergehende Integration der Systeme angestrebt, um nicht nur Daten auszutauschen, sondern auch Funktionalität zugänglich zu machen. Das Ziel der Bemühungen sind offene Systeme, im Bereich der Geoinformatik also ein offenes GIS: Durch Netzwerke soll eine *verteilte Nutzung von Geodaten und GIS-Funktionalität* ermöglicht werden, um von einem System aus Zugriff zu einer umfassenden GIS-Lösung zu haben. Um dieses Ziel zu erreichen, werden Schnittstellen definiert, über die verschiedene Dienste miteinander kommunizieren können.

3.1.1 Das Open GIS Consortium

Das Open GIS Consortium (OGC) ist ein 1994 gegründeter Zusammenschluss von Universitäten, Herstellern und Nutzern von GIS, der die Interoperabilität im Bereich der geographischen Informationsverarbeitung fördern soll. Das Ziel des Open GIS Consortiums (1999:1) ist „the full integration of geospatial data and geoprocessing into mainstream computing and the widespread use

of interoperable geoprocessing software and geodata products throughout the information infrastructure.“ Möglichst viele Leute sollen unabhängig von Plattformen und Anwendungen geographische Daten aus verschiedenen Quellen nutzen und von raumbezogener Datenverarbeitung profitieren können. Dazu erarbeitet und definiert das OGC in einem Konsensprozess seiner Mitglieder² Schnittstellen und veröffentlicht deren Spezifikationen. Diese Schnittstellen ermöglichen den Austausch von Daten und Funktionen zwischen Systemen wie in Abb. 3.1 gezeigt. Die einzelnen Systeme und Technologien können auf dieser Basis als ein System zusammenarbeiten.

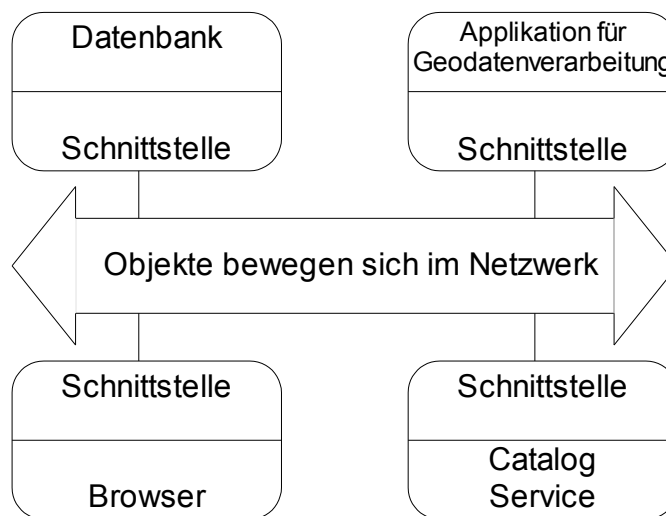


Abb. 3.1: Interoperabilität durch Schnittstellen (vereinfacht nach Sondheim, Gardels und Buehler (1999: 352)).

Zur Open GIS Spezifikation gehören drei Bestandteile: Das Open Geodata Model, das Service Model und das Information Communities Model. Ersteres ist ein für die Integration nötiges Datenmodell, das grundlegende geographische Datentypen beschreibt. Das Service Model ist eine Modellspezifikation zur Implementation von Diensten, mit denen ein Zugriff auf Geodaten, deren Verwaltung, Bearbeitung und Darstellung und möglich ist. Das Information Communities Model stellt ein Rahmenwerk dar, das die beiden ersten Modelle nutzt, um neben der technischen auch eine institutionelle Interoperabilität zu erreichen. Die Hindernisse dazu sind beispielsweise semantische Unterschiede (Vckovski 1998: 46-47).

Neben diesen abstrakten Spezifikationen veröffentlicht das Open GIS Consortium mit den Implementationsspezifikationen auch konkrete Standards für die Realisierung von Software. Beschrieben wird dabei jeweils nur die Schnittstelle, der Rest wird der Implementation überlassen.

3.1.2 Dienste

Allgemein wird als Dienst oder Service Funktionalität bezeichnet, die von einem Objekt durch eine

² Zum Arbeitsweise des OGC am Beispiel des Web-Mapping-Testbeds siehe Joos (2003).

Schnittstelle zur Verfügung gestellt wird (nach ISO, zitiert in Open GIS Consortium 2002b: 2). *Web Services* erlauben es Anwendungen, über das Internet plattform- und programmiersprachenunabhängig zu kommunizieren. Ein Web Service ist eine Softwareschnittstelle, die eine Anzahl von Funktionen beschreibt, die über ein Netzwerk mittels Standardprotokollen zugänglich sind (IBM 2004). Die Ziele solcher Dienste decken sich mit der oben erwähnten Vision eines offenen GIS, da damit Funktionalität (und nicht nur Daten) über ein Netzwerk zugänglich gemacht wird.

3.1.2.1 Verkettung von Diensten

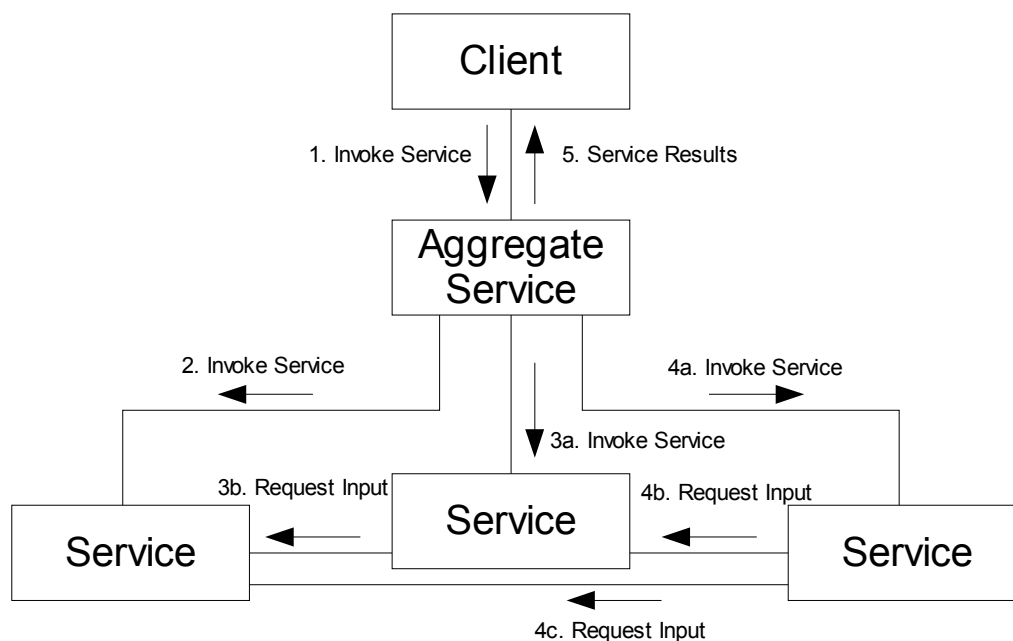


Abb. 3.2: Undurchsichtige Verkettung von Diensten („Opaque chaining“). Für den Nutzer ist nur der Aggregate Service sichtbar (aus Open GIS Consortium 2002b:18).

Um komplexere Aufgaben zu bewältigen, können Dienste kombiniert werden, was als Verkettung von Diensten („Service Chaining“) bezeichnet wird. Das Open GIS Consortium (2002b: 13-19) zeigt unterschiedliche Möglichkeiten auf, wie eine solche Kombination aussehen kann. Eine erste Variante ist die benutzerdefinierte Verkettung, bei der ein Nutzer die einzelnen Dienste selbst aufruft, wozu natürlich Kenntnisse der Services erforderlich sind. Da die Einzelheiten der Dienste für Nutzer nicht versteckt sind, wird dieser Ansatz auch als durchsichtige („transparent“) Verkettung bezeichnet. Eine zweite Möglichkeit ist die arbeitsflussgesteuerte („workflow-managed“), auch als lichtdurchlässig („translucent“) bezeichnete, Verkettung. Hier ruft ein Nutzer einen Dienst auf, der anschliessend eine definierte Abfolge anderer Dienste abrufen, wobei diese für den Nutzer sichtbar bleiben. Eine dritte Variante schliesslich stellt die undurchsichtige („opaque“) Verkettung dar, bei der ein Dienst (Aggregate Service) die Aufrufe der anderen Services kontrolliert (Abb. 3.2). Für den Nutzer ist

nicht sichtbar, dass verschiedene Dienste aufgerufen werden.

3.1.2.2 Geographische Dienste

Das Open GIS Consortium (2002b: 24-31) beschreibt in der „Abstract Specification“ eine Taxonomie von geographischen Diensten und nennt eine Auswahl von Diensten für die einzelnen Kategorien. Die Taxonomie wird im Folgenden gekürzt mit einigen Beispielen wiedergegeben:

- Geographische Benutzerinteraktionsdienste: diverse Viewer und Editoren
- Geographische Modell-/Informationsverwaltungsdienste: Zugriffsdienste auf Karten und Objekte (Map Service, Feature Service), Kataloge
- Geographische Arbeitsfluss-/Aufgabenverwaltungsdienste: Verkettungsdefinitionsdienst
- Geographische Verarbeitungsdienste:
 - räumlich: Koordinatentransformation, Objektmanipulations- und *generalisierungsdienste*, *Positionierungsdienst*
 - thematisch: Klassifikationsdienst, Objekt*generalisierungsdienst*
 - zeitlich
 - Metadaten: Statistikberechnungsdienst
- Geographische Kommunikationsdienste: Transferdienst

Auf der Basis der abstrakten Spezifikation wurden verschiedene Implementationspezifikationen, z.B. für einen Web Map Service oder einen Web Feature Service, veröffentlicht.

3.2 Web Feature Server

Der Zugriff auf einzelne (Vektordaten-) Objekte kann in einem offenen GIS-System über einen Web Feature Server (WFS) geschehen. Nach dem OPEN GIS CONSORTIUM (2002a), welches die Implementationspezifikation des WFS veröffentlicht hat, besteht die Grundfunktionalität eines WFS darin, die angefragten Daten in Geography Markup Language (GML) kodiert zurückzuliefern. Im Gegensatz zu einem Web Map Server, der einen Zugriff auf Karten ermöglicht, erlaubt es einem der WFS, auf Vektordaten zuzugreifen.

Der Server muss fähig sein, Anfragen nach seinen Fähigkeiten (GetCapabilities request) zu beantworten und Beschreibungen der gespeicherten Objekte (DescribeFeatureType request) zu geben. Ein WFS der diese Minimalanforderungen erfüllt, wird vom OPEN GIS CONSORTIUM als Basic WFS bezeichnet. Die Erweiterung davon wird als Transaction WFS bezeichnet und zeichnet sich dadurch aus, dass Daten, d.h. Vektorobjekte, auf dem Server verändert und gelöscht werden

können (Transaction request).

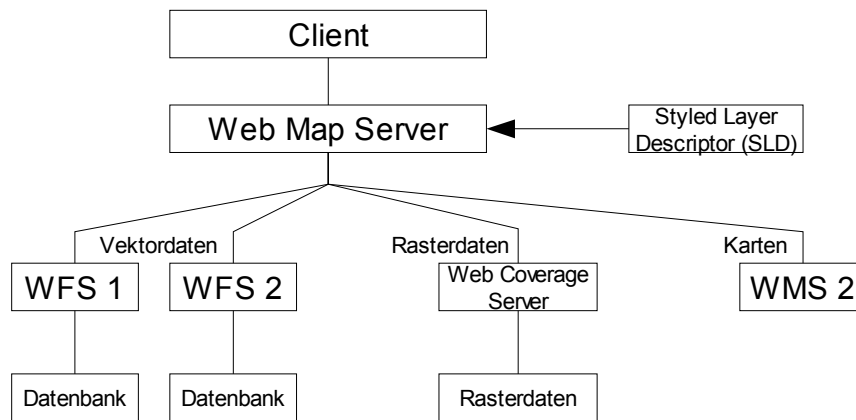


Abb. 3.3: Mehrstufiges System von OGC Web Services für Webmapping. Ein Web Map Server (WMS) bezieht Daten aus verschiedenen Quellen: Vektordaten von Web Feature Servern, Rasterdaten von einem Web Coverage Server und Karten von einem anderen WMS. Der WMS verwendet die in einem Styled Layer Descriptor festgelegte Symbolisierung zur Darstellung der Daten und liefert sie an den Client.

In dieser Arbeit wurde der Web Feature Server aus dem deegree Projekt verwendet. Deegree (2003) ist ein in Java programmiertes Framework um Applikationen zur Lösung von geographischen Problemen zu entwickeln. Dieses Framework steht im Quellcode unter der GNU Lesser General Public License zur Verfügung. Die Schnittstelle des Servers ist als Servlet realisiert; es wurde auf einer Apache Tomcat Version 4.1 eingesetzt.

3.3 XML

XML, Extensible Markup Language, ist ein Standard des World Wide Web Consortiums (W3C) zur Auszeichnung von Dokumenten. Er definiert eine Syntax, die benutzt wird, um Daten mit einfachen, von Menschen lesbaren Tags zu kodieren. Daten und die dazugehörigen Auszeichnungen, welche die Daten beschreiben, bilden zusammen Elemente, den Grundbaustein von XML. Die Elemente sind in den Dokumenten als Textstrings enthalten. XML ist eine Metasprache, d.h. sie definiert keine festgelegte Menge von Tags für alle möglichen Bereiche. XML gibt strenge Regeln vor, beispielsweise wo Tags platziert werden müssen und wie sie auszusehen haben. Um die *Interoperabilität* zu verbessern, können Organisationen oder Individuen auf dieser Basis eine Menge von Tags definieren, die für sie von Interesse sind. Damit werden XML-Sprachen, auch XML-Applikationen genannt, geschaffen, die es inzwischen für alle möglichen Bereiche gibt (zusammengefasst aus Harold und Means 2002: 3-7).

XML ist ein offener, gut dokumentierter Standard und eröffnet als plattformunabhängiges Datenformat grosse Möglichkeiten. Da die Daten beschrieben und durch Menschen lesbar sind,

können sie auch nach sehr langer Zeit noch genutzt werden, was bei anderen Formaten normalerweise nicht der Fall ist. Auf XML basierende Formate sind beispielsweise die nachfolgend besprochenen Geography Markup Language als Format für Geodaten, SVG als Vektorgrafikformat und XSLT.

3.3.1 Geography Markup Language

Mit der Spezifikation der Geography Markup Language definiert das Open GIS Consortium eine XML-Sprache für Geodaten: *“GML is an XML encoding for the modeling, transport and storage of geographic information including both the spatial and non-spatial properties of geographic features.”* (Open GIS Consortium 2003: 1). *„GML provides a variety of kinds of objects for describing geography including features, coordinate reference systems, geometry, topology, time, units of measure and generalized values“* (Open GIS Consortium 2003: xviii).

Die Daten sind in GML unabhängig von einer Darstellung gespeichert, d.h. die Visualisierung wird den Programmen überlassen, welche die Darstellung den Gegebenheiten anpassen können. Dabei kann es allerdings zu Problemen, wie beispielsweise Überlagerungen von Objekten, kommen, was eine Generalisierung nötig macht.

In Netzwerken ermöglichen Dienste die verteilte Nutzung von Funktionalität; GML ermöglicht die verteilte Nutzung von Geodaten. Es gibt die Möglichkeit sogenannte Profile (anwendungsspezifische Variationen) von GML zu schaffen, wobei aber die Gefahr besteht, dass neue XML Sprachen entstehen, die keine Profile von GML sind.

3.3.2 Scalable Vector Graphics

Scalable Vector Graphics (SVG) ist eine Spezifikation des W3C. SVG ist eine XML-Sprache, um zweidimensionale Graphiken - bestehend aus Vektorobjekten, Rasterbildern und Text - zu beschreiben. SVG-Graphiken können interaktiv und dynamisch gestaltet werden (W3C 2003). Eine clientseitige Darstellung von Karten mittels SVG liegt nahe, wenn die Ausgangsdaten in GML bzw. XML vorliegen, da diese mittels einer Extensible Stylesheet Language Transformation leicht in eine andere XML-Sprache konvertiert werden können.

Mit SVG kann man, da es sich um ein Vektorgrafikformat handelt, objektorientiert arbeiten, andererseits bietet SVG diverse Möglichkeiten für eine gute kartographische Darstellung. Eine Betrachtung der Eignung von SVG zur kartographischen Darstellung findet sich bei Neumann und Winter (2000).

Um SVG auf dem Client anzuzeigen, muss auf diesem ein Programm installiert sein, das die übermittelte XML-Datei graphisch darstellen kann. Dieses Programm, das man als SVG-Viewer

bezeichnet, ist häufig als Plugin für einen Webbrowser realisiert. Inzwischen sind auch für mobile Geräte eine Reihe solcher Produkte entwickelt worden, z.B. der PocketSVGViewer von CSIRO, der Mobile SVG Player von BitFlash oder der Open SVG Viewer, der kostenlos heruntergeladen werden kann und im Quellcode vorliegt.

<pre><gml:featureMember> <Shape gid="1"> <gml:pointProperty> <gml:Point srsName="null"> <gml:coordinates cs="," decimal="." ts=""> 50,100 </gml:coordinates> </gml:Point> </gml:pointProperty> </Shape> </gml:featureMember></pre>	<pre><xsl:template name="getPoint" match="//gml:Point/gml:coordinates"> <xsl:variable name="point" select="."/> <xsl:variable name="x" select="number(substring-before(\$point,','))"/> <xsl:variable name="y" select="number(substring-after(\$point,','))"/> <xsl:element name="circle" > <xsl:attribute name="cx"> <xsl:value-of select="\$x"/> </xsl:attribute> <xsl:attribute name="cy"> <xsl:value-of select="\$y"/> </xsl:attribute> <xsl:attribute name="r"> 10 </xsl:attribute> </xsl:element> </xsl:template></pre>
<pre><circle cx="50" cy="100" r="10" /></pre>	

Abb. 3.4: Beispiel für Umwandlung von GML (links) zu SVG (unten) mittels XSLT (rechts). Die Schablone des XSLT-Stylesheets mit dem Namen „getPoint“ wird aufgerufen, wenn im GML-Dokument der entsprechende Knoten (`//gml:Point/gml:coordinates`) gefunden wird. Die grüne Zeile liest die Koordinatenwerte (50,100), die anschliessend in x und y aufgeteilt werden (blau). Der rot markierte Teil konstruiert mit diesen Werten ein SVG-Element (Kreis mit Radius 10).

Bei der Darstellung von geographischen Daten mit SVG muss das Koordinatensystem, das in SVG verwendet wird, berücksichtigt werden: Dieses hat seinen Ursprung in der linken oberen Ecke der Zeichenfläche, wobei der positive Teil der y-Achse abwärts verläuft. Bei geographischen Koordinaten (auf der Nordhemisphäre) muss deshalb eine Transformation der y-Achse durchgeführt werden.

Neumann und Winter (2000) erwähnen einige Problembereiche von SVG, von denen hier der Daten- und Kopierschutz kurz Erwähnung finden soll, da er für die Internetkartographie besondere Bedeutung hat: SVG ist als XML-Sprache definiert und somit für Menschen leicht lesbar. Es besteht die Gefahr, dass anstelle der alleinigen Nutzung der graphischen Darstellung die dafür verwendeten Daten extrahiert und anderweitig verwendet werden. Diese Urheberrechtsverletzungen sind im Internet nur schwer zu unterbinden.

Ein Gegenmittel kann die Generalisierung der Daten sein. Damit werden Rückschlüsse auf die Originaldaten erschwert oder verhindert, die gewünschte Information kann aber trotzdem vermittelt

werden. Die Generalisierungsoperation Typisierung mit Verwendung von Clusteringverfahren scheint dazu gut geeignet zu sein.

3.3.3 Extensible Stylesheet Language Transformations

Die Extensible Stylesheet Language (XSL) ist wie SVG eine auf XML basierende Spezifikation des W3C. Sie besteht aus zwei Teilen: XSL *Transformations* (XSLT) und XSL Formatting Objects (XSL-FO), das die graphische Darstellung von Daten beschreibt und hier nicht weiter interessiert. XSLT dient dazu, Dokumente in einer XML-Sprache in eine andere XML-Sprache umzuwandeln. Eine solche Transformation basiert auf Regeln, die in einem Stylesheet beschrieben werden und die vorgeben, wie die Baumstruktur des Ausgangsdokuments in den Baum des resultierenden Dokuments umzuwandeln ist. Umgesetzt wird das mittels Mustern („pattern“), die mit sogenannten Schablonen („templates“) verbunden sind. Die Muster werden mit Elementen im Baum des Ausgangsdokuments verglichen und bei einer Übereinstimmung wird die entsprechende Schablone aufgerufen, die einen Teil des Resultatbaums erzeugt (nach W3C 1999).

Um solche Transformationen ausführen zu können, ist ein XSLT Prozessor nötig. Für diese Arbeit wurde der Xalan-Prozessor Version 2.4.1 aus dem Apache-Projekt (2004) verwendet.

3.3.3.1 Erweiterungsfunktionen

XSLT kann in seiner Funktionalität auf zwei Arten erweitert werden: Mittels Erweiterungselementen („extension elements“) und Erweiterungsfunktionen („extension functions“). In der Empfehlung zu XSLT 1.0 wird auf diese Möglichkeit hingewiesen, allerdings ist die Implementation dieser Erweiterungen dort nicht definiert. Verschiedene XSLT-Prozessoren bieten trotzdem die Möglichkeit, solche Erweiterungen zu verwenden, allerdings sind die Stylesheets dann nicht mehr uneingeschränkt portierbar, da die Implementationen unterschiedlich sind. Xalan unterstützt Erweiterungsfunktionen, die in Java implementiert werden.

Der Vorteil dieser Erweiterungen ist, dass viele Aufgaben in einer kompletten Programmiersprache wie z.B. Java einfacher zu lösen sind. XSLT kann zwar auch als umfassende Programmiersprache betrachtet werden³, allerdings sind kompliziertere Berechnungen mit viel Programmieraufwand verbunden, da Stylesheets wohlgeformte XML-Dokumente sein müssen. Zusätzlich fehlen XSLT Standardelemente von Programmiersprachen, wie z.B. Iterationen; hier muss man über Umwege zum Ziel gelangen.

Ein Problem bei der Arbeit mit Erweiterungsfunktionen stellen Datentypen und -strukturen dar. XSLT kennt folgende Datentypen, die in XPath definiert sind: boolean, number für die Darstellung

³ Es wurde bewiesen, dass XSLT Turing-komplett ist (Harold und Means 2002: 5).

aller Zahlen, string und node-set. Es gibt keine Datenstruktur, die wie ein Array eine bestimmte Anzahl von Werten speichern kann. Werden Erweiterungsfunktionen verwendet, können diese Werte zurückgeben, die von XSLT verstanden werden. Dabei wird eine Typkonvertierung auf den Datentyp durchgeführt: So werden z.B. die verschiedenen Datentypen, die Java zur Speicherung von Zahlen benutzt, in den Datentyp number umgewandelt. Eine Erweiterungsfunktion kann aber auch Werte bzw. Objekte eines beliebigen Typs zurückgeben. XSLT hält in diesem Fall eine Referenz zu diesem Objekt. Mittels dieser kann das Objekt als Argument für weitere Erweiterungsfunktionen verwendet werden. Zuletzt muss jedoch eine Methode die Daten in einer Form liefern können, die von XSLT verstanden wird: Dabei kann nicht auf einzelne Elemente eines Arrays zugegriffen werden und auch der Zugriff auf Elemente `java.util.Vector` ist nicht möglich, da dort der Datentyp `java.Object` zurückgegeben wird, von dem in XSLT keine Typkonvertierung möglich ist.

4 Implementation

In diesem Kapitel wird der Einbau von Generalisierungstechniken in ein offenes System, das die in Kapitel drei vorgestellten Technologien nutzt, diskutiert. Ein zweiter Teil dieses Kapitels ist der Vorstellung der verwendeten Daten und deren Visualisierung gewidmet.

4.1 Integration von Generalisierung

Dienste stellen Funktionalität zur Verfügung, womit über Netzwerke eine verteilte Nutzung möglich wird. Die Definition der Schnittstellen ermöglicht eine Zusammenarbeit zwischen verschiedenen Diensten. Zur Lösung eines Problems können anstelle eines einzigen grossen Programms, das auf einem lokalen Rechner läuft, verschiedene Dienste, die über das Internet zugänglich sind, flexibel kombiniert werden. In einem solchen System ist es naheliegend, Generalisierung als Service zu integrieren. Generalisierungsdienste werden vom OGC auch in der Taxonomie geographischer Dienste erwähnt (vgl. Kap. 3.1.2.2).

Um einen Service zu spezifizieren, muss seine Schnittstelle festgelegt werden. Naheliegend ist es, GML als Datenformat sowohl für Eingangs- wie Ausgangsdaten des Dienstes zu verwenden. Schwieriger wird die Festlegung der Funktionen, die der Dienst anbieten soll. Betrachtet man die herrschenden Differenzen, beispielsweise bezüglich Generalisierungsmodellen und -operatoren (vgl. Kap. 2), dürfte es nicht allzu leicht sein, einen Konsens zu finden. Vor der Spezifikation eines Generalisierungsdienstes durch das OGC sind demnach noch einige Hindernisse zu überwinden.

4.2 Aufbau der Architektur

Anhand einer schematischen Darstellung (Abb. 4.1) soll das Zusammenspiel der besprochenen Technologien erläutert werden. Der genaue zeitliche Ablauf ist aus dem Sequenzdiagramm (Abb. 4.2) ersichtlich.

Bei einer Anwendung für Webmapping oder Location Based Services ist davon auszugehen, dass die Nutzer nur wenig technische Kenntnisse haben. Die Interaktion mit dem System muss deshalb möglichst einfach sein und technische Details sollten vom Benutzer ferngehalten werden. Die aufgebaute Architektur ist deshalb nach dem Prinzip der undurchsichtigen Verkettung aufgebaut (vgl. Kap. 3.1.2.1), bei dem eine Nutzerin nur mit einem einzigen Dienst, dem Aggregate Service, in Kontakt kommt. Dieser ist implementiert mit einem Servlet, das einerseits für die Interaktion mit der Anwenderin verantwortlich ist, andererseits die Aufrufe der Dienste steuert.

Die Benutzeroberfläche besteht neben dem Servlet aus einer einfachen HTML-Seite. Ein Nutzer wählt auf dieser mittels verschiedener Auswahlmöglichkeiten die Daten aus, die er dargestellt haben möchte. Ausserdem kann er Einfluss darauf nehmen, wie diese Daten visualisiert werden sollen. Das Servlet setzt die vom Nutzer gemachten Angaben zu einer Anfrage an den Web Feature Server zusammen. In dieser Anfrage sind neben der Bezeichnung der gewünschten Daten auch Filter definiert, mit denen eine Auswahl der Daten erfolgt.

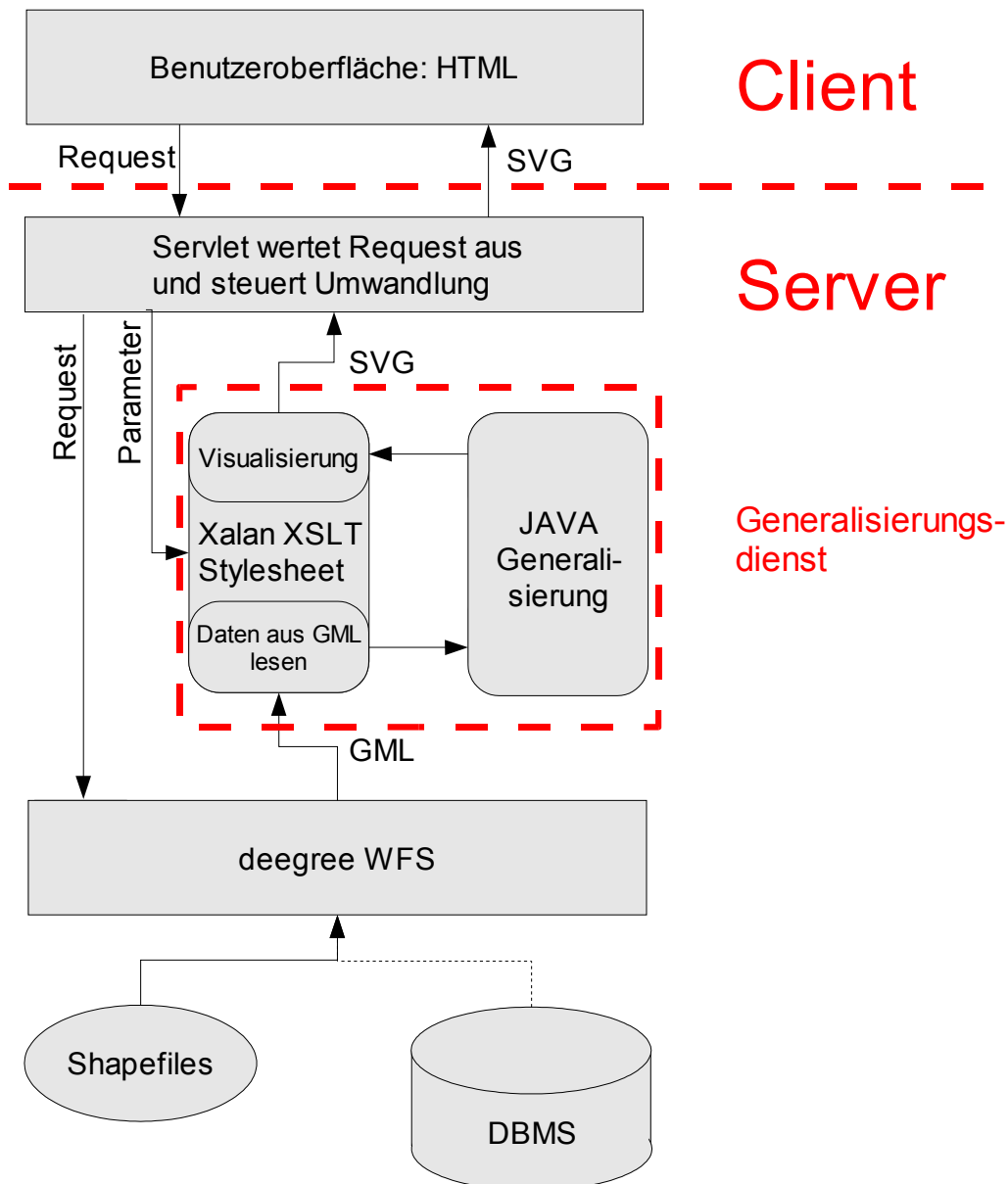


Abb. 4.1: Schematische Darstellung des aufgebauten Systems.

Die erhaltene Anfrage veranlasst den Server, sich die benötigten Daten aus dem Speicher zu holen. Die gelesenen Daten wandelt der Server dann in GML um. Der verwendete WFS von deegree (2003) unterstützt als Datenquellen Datenbanken und ESRI Shapefiles. Neben der schlechteren Zugriffsleistung gegenüber einer Datenbank ist man mit Shapefiles bei der Selektion der Daten

eingeschränkt. Der einzige Filter, der zur Auswahl der Daten verwendet werden kann, ist eine „bounding box“. Es ist also keine Datenauswahl über Attribute möglich. Da die Anwendung hier nur Testzwecken dient, wurden der Einfachheit halber trotzdem Shapefiles als Datenquelle benutzt. Für einen produktiven Einsatz ist deshalb die Verwendung einer Datenbank unumgänglich.

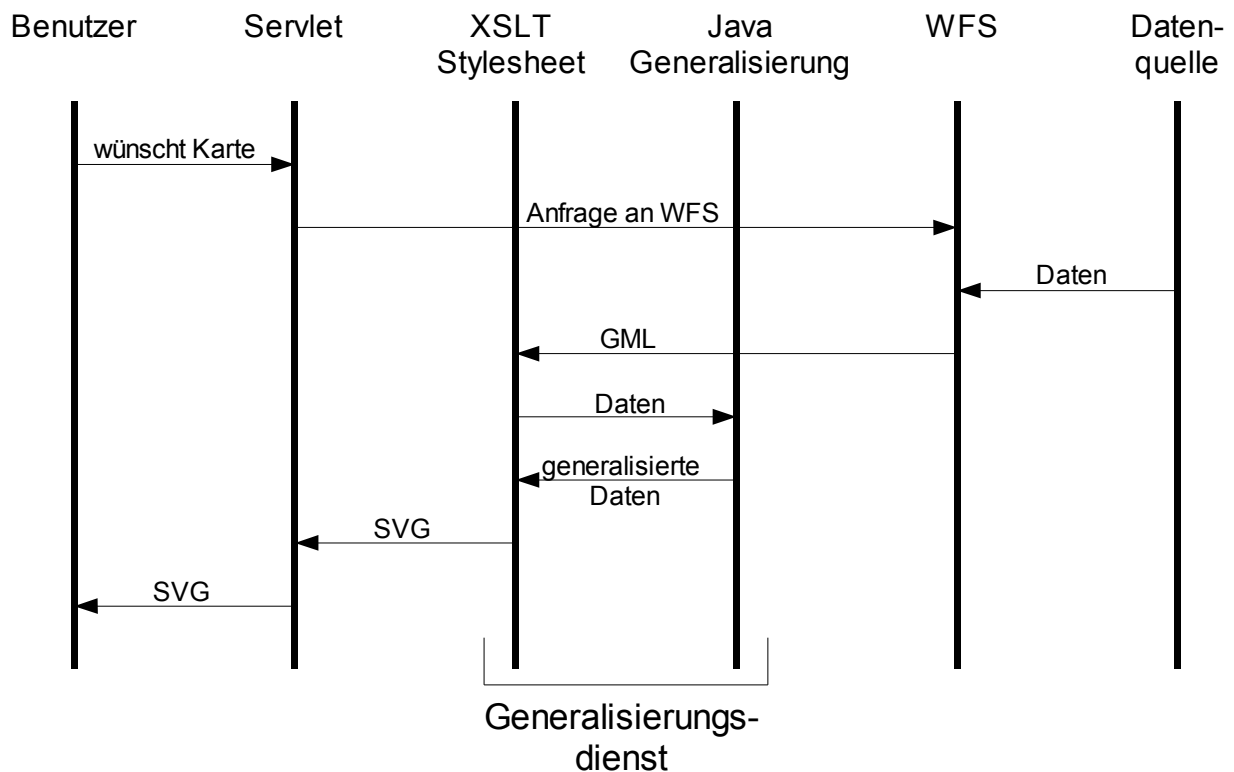


Abb. 4.2: Sequenzdiagramm der Herstellung einer Karte im Rahmen der verwendeten Architektur.

Das Servlet der Benutzerschnittstelle wählt entsprechend der Visualisierungswünsche des Benutzers das geeignete Stylesheet aus. Diesem werden verschiedene Parameter übergeben: Die Bildschirmauflösung des Clients und die bounding box der Daten, die vom Benutzer gewünscht werden. Damit kann die Grösse der entstehenden SVG-Datei an das beim Benutzer vorhandene Display angepasst werden.

Zusammen mit einer Gruppe von Java-Klassen bildet das Stylesheet den Generalisierungsdienst. Es liest die Daten aus GML und ruft dann verschiedene Methoden bzw. Erweiterungsfunktionen aus den Java-Klassen auf. Mit diesen Methoden werden die Daten den Klassen zugänglich gemacht und die Generalisierung ausgelöst. Ist diese beendet, werden die Resultate zurück an das Stylesheet geliefert. Dieses konstruiert nun den Resultatbaum bzw. die SVG-Datei, indem die Daten zur Konstruktion von geometrischen Elementen verwendet werden. Das Stylesheet ergänzt ausserdem die SVG-Datei mit einer Hintergrundkarte, die als Rasterbild hinter die Vektordaten gelegt wird. Die SVG-Datei wird dann via Servlet an den Client übermittelt.

4.3 Diskussion der Architektur

Die vorgestellte Architektur hat sich in der Testphase dieser Arbeit bewährt. Es handelt sich dabei aber nur um eine Testumgebung, die für einen produktiven Einsatz zahlreiche Verbesserungen erfordert. Besonders erwähnt werden soll in diesem Zusammenhang vor allem die Benutzerschnittstelle.

Da die Rechenleistungen von kleinen, mobilen Geräten immer noch ziemlich beschränkt sind, wurde entschieden, mit einem dünnen Client zu arbeiten. Dies bedeutet, dass der Grossteil der Funktionalität serverseitig implementiert wird. Im Falle dieser Arbeit zeigt der Client nur die HTML-Seite der Benutzerschnittstelle an und stellt die gelieferte SVG-Datei mittels eines Viewers graphisch dar.

4.3.1 Generalisierungsdienst

Ein möglicher Generalisierungsdienst wurde bereits im Kapitel 4.1 skizziert. In Abweichung davon wurde für diese Arbeit Generalisierung und Visualisierung in einem Dienst zusammengefasst, weshalb das Ausgabeformat des Dienstes SVG ist. Auf diese Weise kann das Ziel, dem Benutzer innert kürzester Zeit eine generalisierte Karte zu liefern, besser erreicht werden. Ein flexiblerer Ansatz sollte aber mit einem reinen Generalisierungsservice arbeiten. In diesem Fall kann ein Web Map Server zur Visualisierung der Daten verwendet werden (wie in Abb. 3.3 gezeigt), worauf im Rahmen dieser Architektur verzichtet wurde.

Technisch besteht der Generalisierungsdienst aus einem XSLT-Stylesheet, das die Daten aus GML in SVG transformiert, und den Java-Klassen, mit deren Methoden die Generalisierung durchgeführt wird. Lehto und Kilpeläinen (2001) verwendeten ebenfalls XSLT-Stylesheets mit Erweiterungsfunktionen zur Echtzeit-Generalisierung von geographischen Daten. Sie kommen zum Schluss, dass XSLT ein vielversprechendes Werkzeug zur Generalisierung von räumlichen Datenbeständen in Echtzeit ist. Die gleiche Technik beschreiben auch Harrie, Sarjakoski und Lehto (2002). Sie bemerken, dass für komplexe Probleme XSLT möglicherweise zu wenig leistungsfähig ist und schlagen für einen solchen Fall die Verwendung einer geometrischen und topologischen Java-Bibliothek vor (z.B. Java Topology Suite).

Eine technische Umsetzung ohne XSLT muss mit der Generalisierungskomponente direkt auf den GML-Datenstrom aus dem Server zuzugreifen. Dies ist mit Java unter Verwendung von DOM oder SAX möglich. Die Komponente kann dann als Resultat der Generalisierung wieder einen GML-Datenstrom zurückgeben, der mittels XSLT visualisiert oder an einen Web Map Server weitergeleitet wird. Auf diese Weise könnte eventuell die Performance verbessert werden. Allerdings

ist man mit dieser Variante gezwungen, Funktionalität zu implementieren, die in einem XSLT-Prozessor vorhanden ist. Möglich ist auch, die Visualisierung in der gleichen Java-Applikation durchzuführen wie die Generalisierung, so dass direkt SVG zurückgegeben wird. Für diese Möglichkeit gilt ebenfalls, dass bereits vorhandene Funktionalität neu implementiert werden muss. Dies bedeutet einen höheren Entwicklungsaufwand, der sich im Rahmen dieser Arbeit nicht rechtfertigen liess.

4.3.2 Andere Möglichkeiten

Denkbar wäre auch, eine Generalisierung direkt in den Web Feature Server einzubauen. Der Vorteil liegt dabei in der Geschwindigkeit, weil direkt mit dem vom WFS verwendeten Datenmodell gearbeitet werden kann. Dieser Ansatz erscheint vom Gesichtspunkt der Flexibilität aus betrachtet nicht sinnvoll: Spezifiziert ist nur die Schnittstelle des Servers, die Implementationen können sehr unterschiedlich sein und auch unterschiedliche Datenmodelle verwenden. Eine Generalisierungskomponente müsste daher für jeden Server angepasst werden.

Ein ganz anderer Ansatz wäre die repräsentationsorientierte Sichtweise: Eine multiple Repräsentation wird in der Datenbank gespeichert. Der Server kann dann auf generalisierte Daten zugreifen, was den prozessorientierten Generalisierungsdienst überflüssig macht (ausser bei einer Kombination der beiden Sichtweisen).

4.3.3 Location Based Services

Eine Architektur zur Verwendung mit Location Based Services muss für einen produktiven Einsatz das Problem der unter Umständen unterbrochenen Verbindung zum Client berücksichtigen können. Beim Vorhandensein einer Verbindung sollten mehr Informationen und Daten übertragen werden können, als im Moment gebraucht werden, damit der Benutzer im Falle einer unterbrochenen Verbindung trotzdem noch auf gewisse Informationen zugreifen kann⁴. Dies ist auch aus Sicherheitsüberlegungen heraus sinnvoll: Verlässt sich ein Benutzer nur auf einen Location Based Service, könnte ein Ausfall der Verbindung z.B. im Gebirge unangenehme Folgen haben.

4.4 Implementation der Generalisierung

Die Generalisierungskomponente wurde in Java implementiert. Die wichtigsten Eigenschaften der Klassen, aus denen die Applikation besteht, werden im Folgenden vorgestellt (vgl. auch Abb. 4.3). Die komplette Dokumentation der Klassen mit Javadoc findet sich auf der beigelegten CD.

⁴ Bei der Verwendung von GPRS entstehen so aber zusätzliche Kosten, da nach übertragener Datenmenge abgerechnet wird.

- Point: Diese Klasse repräsentiert einen einzelnen Punkt. Sie besitzt dazu die Attribute, in denen x- und y-Koordinate gespeichert werden. Die Klasse kann die Distanz von diesen Koordinaten zu einem anderen gegebenen Punkt bestimmen.

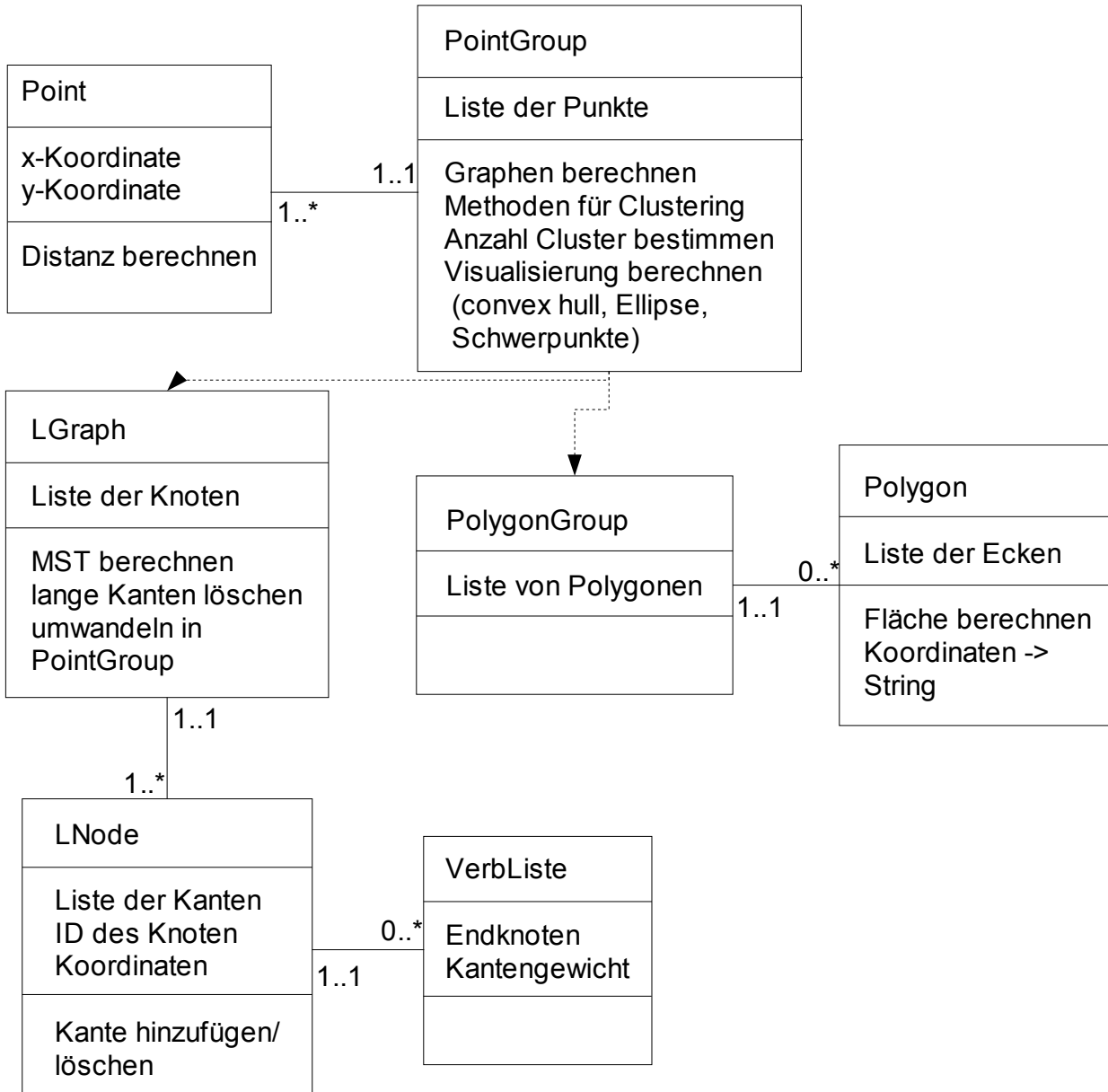


Abb. 4.3: Klassendiagramm der Generalisierungskomponente.

- **PointGroup**: Diese Klasse steht für eine Gruppe von Punkten, die in einer Liste gespeichert sind. Diese Klasse besitzt Methoden, um Cluster in einer solchen Punktgruppe zu erkennen: alle Clusteringverfahren, welche nicht auf Graphen basieren, sind in dieser Klasse implementiert. Daneben finden sich in der Klasse auch Methoden, mit der die Anzahl Cluster bestimmt werden können, sowie die Möglichkeit, aus der Gruppe von Punkten einen Graphen zu berechnen. Wichtig in dieser Klasse sind die Methoden, die aus den Resultaten der Generalisierung die Daten für die Visualisierung berechnen: Aus einem Array von PointGroups

kann für jede einzelnen Gruppe, d.h. für jeden Cluster, z.B. der Schwerpunkt oder die Standardabweichungsellipse berechnet werden.

- **VerbListe:** Eine Kante eines Graphen wird durch diese Klasse repräsentiert. Sie speichert den Endknoten und das Gewicht der Kante.
- **LNode:** Diese Klasse steht für einen Knoten eines Graphen. Wichtigstes Attribut ist eine Liste der Kanten (**VerbListe**), die von diesem Knoten ausgehen. Der Knoten verfügt über Methoden, die es ihm erlauben, Kanten zu dieser Liste hinzuzufügen oder daraus zu löschen.
- **LGraph:** Implementation eines Graphen mit Knotenliste. In dieser Arbeit wurde zuerst ein Graph mit einer Matrix implementiert (**WeightGraphMatrix**), welcher aus Performancegründen durch **LGraph** ersetzt wurde. Diese Klasse verfügt über Methoden, um aus einem gegebenen Graphen einen Minimum Spanning Tree zu berechnen und um aus diesem die längsten Kanten zu löschen. Ebenfalls vorhanden ist eine Methode, um einen Graph wieder in eine **PointGroup**, bzw. in einen Array von **PointGroups** umzuwandeln.
- **Polygon:** Diese Klasse repräsentiert ein Polygon und kann dessen Fläche berechnen.
- **PolygonGroup:** Repräsentiert eine Gruppe von Polygonen. Benötigt wird diese Klasse bei der Umwandlung eines Arrays von **PointGroups** in Polygone.

Im Prinzip müssten auch noch eine Klassen für Ellipsen und eine für eine Gruppe von Ellipsen existieren, doch wurde aus Zeitgründen darauf verzichtet, in diesem Fall konsequent objektorientiert zu bleiben.

Wegen der fehlenden Unterstützung von komplexen Datentypen in XSLT war es nötig, Lösungen zu finden, mit denen die generalisierten Daten für XSLT zugänglich gemacht werden können:

- Der Zugriff auf die Koordinaten von Einzelpunkten, z.B der Schwerpunkte, ist in XSLT über Zugriffsmethoden, die einen `double` Wert zurückgeben, möglich. Ein `double` Wert wird automatisch in den Typ `number` konvertiert.
- Die Koordinaten der Polygonecken sind in einem `Vector` gespeichert, mit dem man in XSLT nicht arbeiten kann. Die `elementAt()` Methode funktioniert zwar, gibt aber `Object` zurück, was ohne die Möglichkeit einer Typenkonvertierung wertlos ist. Eine erste Möglichkeit, um auf die Koordinaten der Ecken zugreifen zu können, ist eine Methode in der Klasse `Polygon`, mit der man auf die einzelnen Ecken zugreifen kann, und die eine Instanz der Klasse `Point` liefert: `public Point getCorner(int index)`. Eine zweite Möglichkeit ist eine Methode in `Polygon`, welche die Koordinaten aller Ecken in einen `String` schreibt. Der Vorteil davon ist, dass dieser `String` ohne weitere Bearbeitung in das `polygon`-Tag von `SVG` eingesetzt werden kann. In dieser Arbeit wurde deshalb die zweite Möglichkeit verwendet.

- Bei den Ellipsen wurde die gleiche Lösung verwendet wie bei den Polygonen: Es wird ein String zurückgegeben, der die Koordinaten des Mittelpunkts, die Länge der Haupt- und Nebenachse sowie den Rotationswinkel enthält. Hier ist es jedoch nötig, die einzelnen Zahlen aus dem String herauszulesen, um sie an die passenden Stellen im SVG-Dokument zu schreiben.

4.5 Verwendete Daten

Zum Testen der Generalisierung standen Daten des Schweizerischen Nationalparks (SNP) zur Verfügung. Im Vordergrund standen dabei Beobachtungen von Huftieren, also von Steinböcken, Gamsen, Rothirschen und Rehen, da für diese das Interesse der Parkbesucher am grössten sein dürfte. In zwei Gebieten des Parks (Il Fuorn, Trupchun) werden viermal jährlich die Huftiere gezählt und kartiert. Der für diese Arbeit verwendete Datensatz enthält alle Beobachtungen seit 1997. Verwendet werden die Daten der Steinböcke, Gamsen und Rothirsche.

Aus diesen Beobachtungen wurden Testfälle ausgewählt, um verschiedene Möglichkeiten zur Visualisierung der Daten zu zeigen und die Generalisierung zu untersuchen. Die Auswahl erfolgte einerseits über den Zeitpunkt, andererseits über die Geometrie der Beobachtungen. Nötig ist eine solche Selektion, weil sehr viele Daten vorliegen, z.B. im Fall der Gamsen rund 2100 Beobachtungen. Ein Blick auf die Darstellung der Datensätze in Abb. 4.1 zeigt, dass zwar teilweise ein Clustering möglich, mit Blick auf die Zielsetzung aber kaum sinnvoll wäre. Diese ist, einen Parkbesucher beim Suchen von Tieren⁵ zu unterstützen. Eine Angabe, dass praktisch überall (in den Beobachtungsgebieten) schon einmal Tiere gesehen wurden, hilft dabei nicht weiter. Interessanter ist, wo sich die Tiere in einer bestimmten Jahreszeit häufig aufhalten oder, falls sehr aktuelle Daten vorliegen, wo sie vor kurzem gesehen wurden.

Ein Nachteil dieser Daten ist, dass sie nur zwei Teilgebiete des Parks abdecken. Die Ergebnisse der Clusteringalgorithmen können durch diese Tatsache verfälscht werden, wenn man den ganzen Nationalpark als Untersuchungsgebiet für die Generalisierung verwendet. Weiter sind die Beobachtungen teilweise ziemlich regelmässig über die Fläche verteilt. Dabei stellt sich die Frage, ob ein Clustering mit allen Daten möglich und sinnvoll ist.

Um auch eine Evaluation des Clusterings über die ganze Parkfläche durchführen zu können, wurde ein weiterer Datensatz, in dem sich Standorte aus dem ganzen Park finden, gesucht. Zusätzlich sollten diese Beobachtungen deutlich auf bevorzugte Aufenthaltsorte der Tierart hinweisen. Diese

⁵ mit dem Fernglas.

Bedingungen erfüllen z.B. die Beobachtungen von Mardern.

Ein Punkt steht für eine gemachte Beobachtung, wobei diese mehrere Tiere umfassen kann. Die Anzahl der Tiere ist als Attribut abgespeichert, wenn möglich aufgeschlüsselt nach männlichen, weiblichen sowie Jungtieren. Möchte man für diese Daten statistische Masse berechnen, müssen die Punkte entsprechend gewichtet werden. Eine Gewichtung ist aber in der verwendeten Implementation der Generalisierung und Visualisierung nicht vorgesehen, weil vorausgesetzt wurde, dass ein Punkt einem einzelnen Objekt in der realen Welt entspricht.

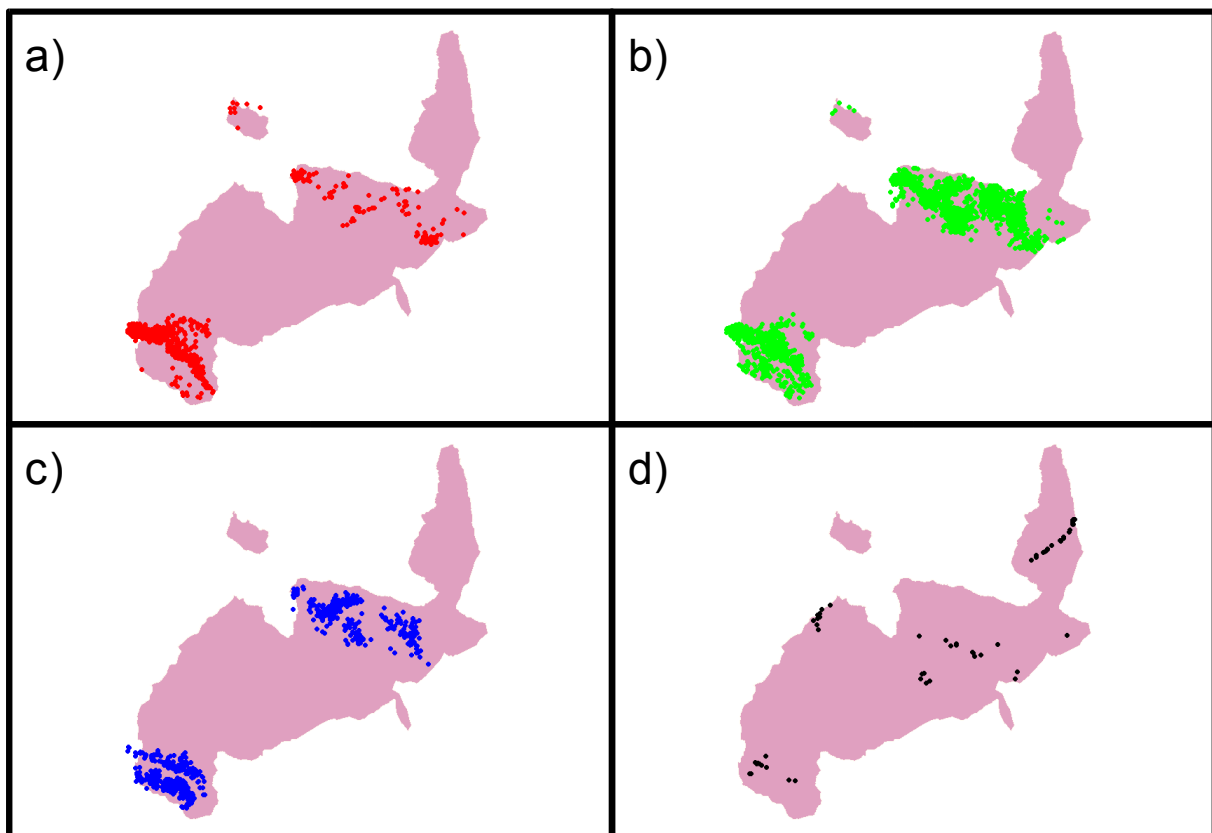


Abb. 4.4: Darstellung der verwendeten Datensätze. Beobachtungen zwischen 1997 und 2002 von a) Steinböcken, b) Gemsen und c) Rothirschen; d) zeigt die Beobachtungen von Mardern zwischen 1995 und 2002.

Die im Folgenden vorgestellten Resultate sind dadurch teilweise verfälscht. Von den Generalisierungsverfahren sind die Maschenvereinfachung und der k-means Algorithmus betroffen, da bei diesen der Schwerpunkt zur Berechnung der Cluster verwendet wird. Würde eine Gewichtung verwendet, könnte sich dessen Position und damit auch die Unterteilung der Cluster ändern. Die anderen Verfahren liefern die gleichen Resultate, da dort nur die Distanz zwischen den Punkten massgebend ist. Bei der Symbolisierung der Daten sind lokale Signaturen und Standardabweichungsellipse von der fehlenden Gewichtung der Punkte beeinflusst.

Eine Möglichkeit zur Korrektur ist, die Daten zu verändern. Beispielsweise können anstelle eines

Punktes, der die Beobachtung von drei Tieren wiedergibt, drei Punkte mit denselben Koordinaten gespeichert werden, die jeweils für ein Tier stehen.

4.6 Symbolisierung

Clusteringverfahren fassen Objekte, in diesem Fall Punkte, zu Gruppen zusammen. Dadurch wird eine bessere Übersicht möglich, indem Muster innerhalb der Daten sichtbar gemacht werden; im Fall dieser Arbeit Gebiete, in denen eine bestimmte Tierart häufig beobachtet wurde. Um diese Information aber an einen Kartennutzer weitergeben zu können, muss sie visualisiert werden. Dazu muss man den Clustern eine graphische Repräsentation zuweisen.

4.6.1 Darstellung durch lokale Signaturen

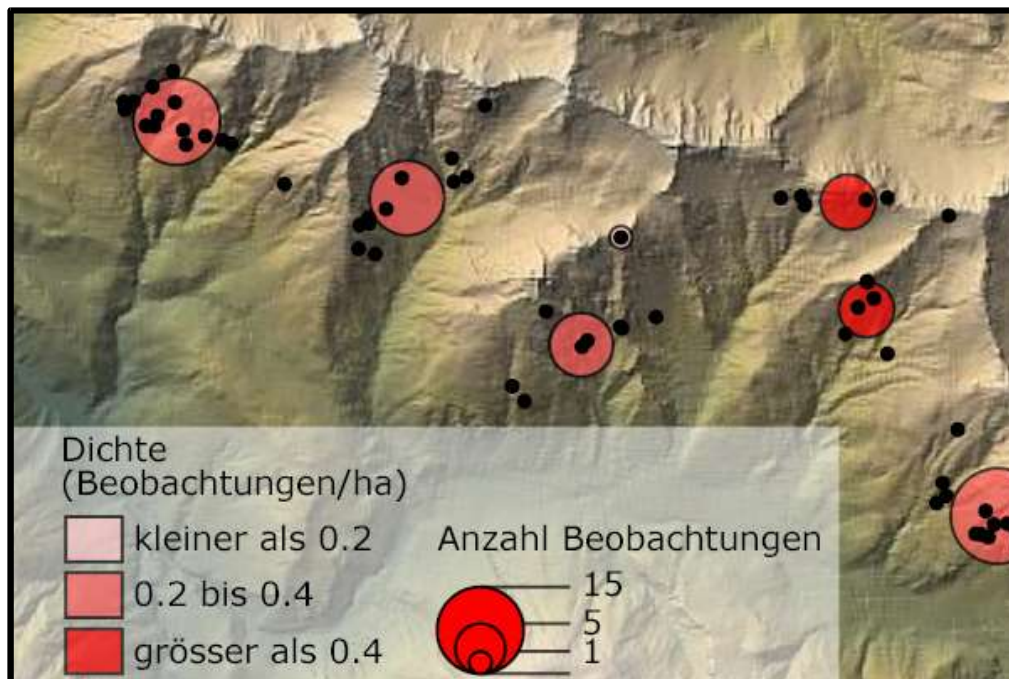


Abb. 4.5: Darstellung von generalisierten Steinbockbeobachtungen in den Monaten Mai und August der Jahre 1997 bis 2002 im Gebiet des Ofenpasses. Gezeigt wird die Wiedergabe von Clustern mittels skalierten lokalen Signaturen (Skalierung: $r = 100 * (\text{Anzahl Beobachtungen})^{0.57}$). Zum Vergleich sind die Originalbeobachtungen als schwarze Punkte eingezeichnet.

Eine erste Möglichkeit zur Visualisierung von Clustern besteht in der Verwendung einer lokalen Signatur, welche jeweils ein Cluster repräsentiert, d.h. es wird der Generalisierungsoperator Typreduktion ausgeführt. In einer einfachen Variante erfolgt die Darstellung durch eine geometrische Signatur, möglich ist aber auch eine bildhafte Signatur, welche die jeweilige Tierart durch eine stilisierte Abbildung wiedergibt. Durch Variation der graphischen Variablen ist es möglich,

verschiedene Qualitäten der Objekte anzuzeigen. In Abb. 4.2 wird beispielsweise die Grösse der Signaturen verwendet, um die Anzahl Beobachtungen in einem Cluster darzustellen. Mit der Variation von Farbe oder Form können auch die Standorte mehrerer Tierarten in derselben Karte abgebildet werden. Zur Platzierung des Symbols bietet sich der Schwerpunkt des Clusters an.

Zwischen Clustern gibt es grosse Unterschiede: Einige enthalten nur wenige Objekte und erstrecken sich über eine grosse Fläche⁶, während für andere das Gegenteil zutrifft. Aus dieser Information kann man auf die Bedeutung der einzelnen Cluster schliessen. Dazu berechnet man aus der Anzahl der Beobachtungen und der Fläche einen Dichtewert, der durch eine Einfärbung der Signaturen dargestellt wird. Allerdings ist dieser Wert nicht sehr aussagekräftig, da die gewählte Flächendefinition nicht optimal ist.

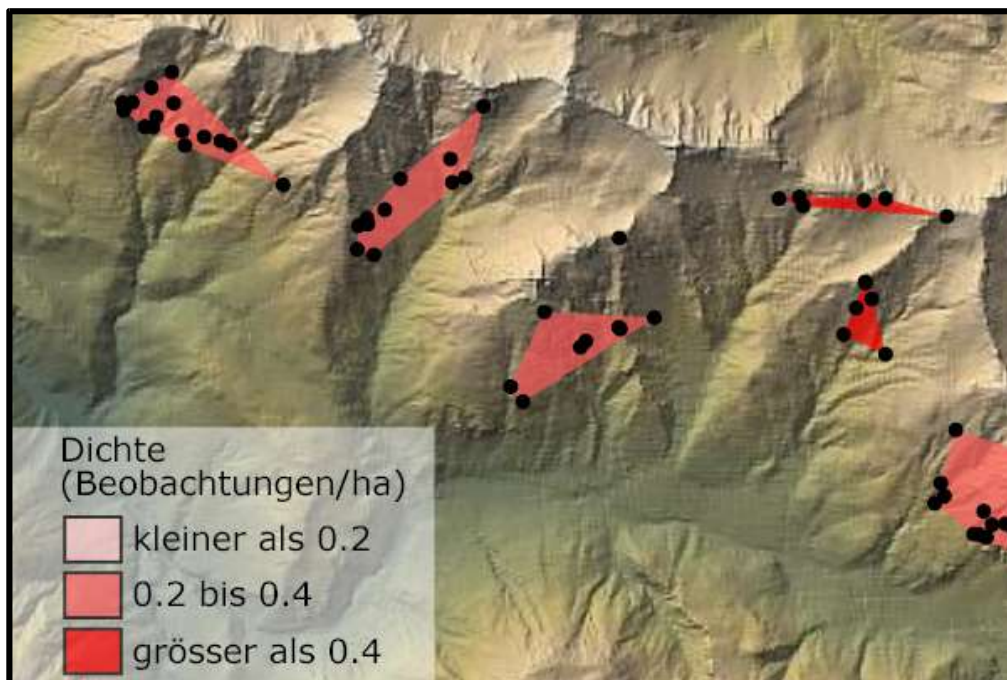


Abb. 4.6: Darstellung von generalisierten Steinbockbeobachtungen in den Monaten Mai und August der Jahre 1997 bis 2002 im Gebiet des Ofenpasses. Ein Cluster wird durch die konvexe Hülle um alle enthaltenen Punkte dargestellt. Zum Vergleich sind die Originalbeobachtungen als schwarze Punkte eingezeichnet.

Die Wiedergabe eines Clusters mittels einer lokalen Signatur am Schwerpunkt ist nur wenig von Ausreissern in den Punkten beeinflusst. Ein weiterer Vorteil ist, dass viele Variationen der Symbole möglich sind, womit Informationen über Tierart und Dichte innerhalb des Clusters vermittelt werden können. Als Nachteil ist zu vermerken, dass mit dieser Darstellung das Gebiet, in dem die Beobachtungen gemacht wurden, nicht abgegrenzt wird. Es ist zwar bekannt, dass um die Signatur herum Tiere gesehen wurden, aber nicht in welcher Distanz dazu.

⁶ Mit der Fläche eines Clusters ist die Fläche der konvexen Hülle um alle Punkte eines Clusters gemeint.

4.6.2 Darstellung durch Flächen

Cluster können durch Flächen dargestellt werden. Eine Möglichkeit dazu bietet die konvexe Hülle um alle Punkte eines Clusters, womit die ganze Fläche eines Clusters sichtbar gemacht wird. Der Nachteil dieser Darstellung ist, dass einzelne, weit abseits liegende Punkte diese Fläche sehr gross machen können. Dies führt zu Clustern mit grossen Gebieten, in denen keine Beobachtungen gemacht wurden. Ein solcher Cluster ist ungünstig, da er ein falsches Bild der räumlichen Verteilung zeichnet.

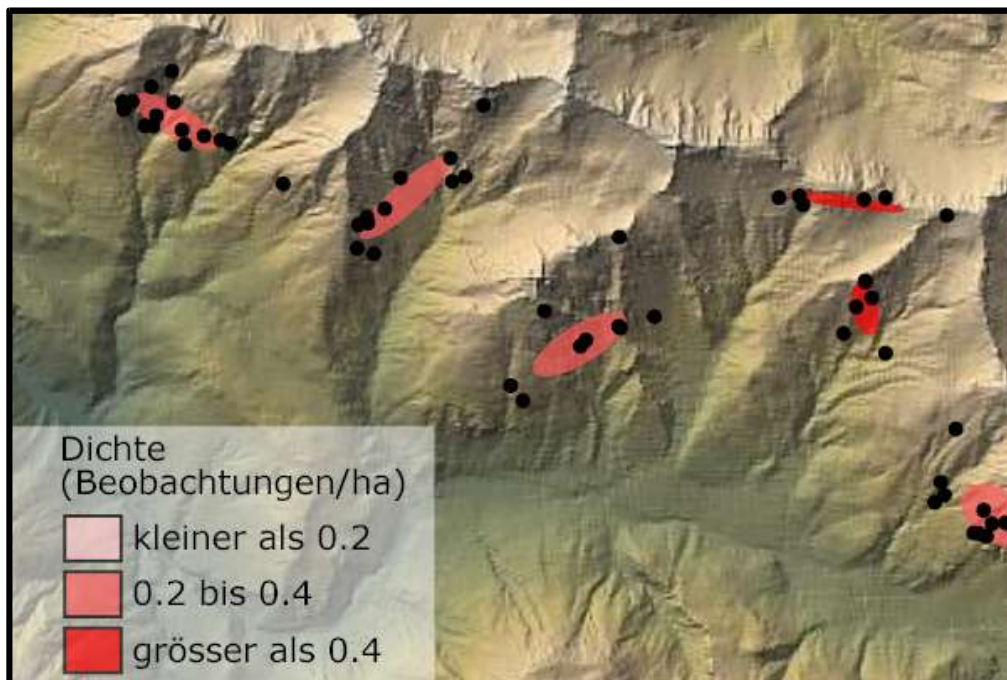


Abb. 4.7: Darstellung von generalisierten Steinbockbeobachtungen in den Monaten Mai und August der Jahre 1997 bis 2002 im Gebiet des Ofenpasses. Die Cluster werden durch eine Standardabweichungsellipse repräsentiert. Zum Vergleich sind die Originalbeobachtungen als schwarze Punkte eingezeichnet.

Besser geschieht die Repräsentation eines Clusters durch eine Fehlerellipse, wobei hier die Standardabweichungsellipse verwendet wird. Mit der Standarddistanz steht ein Mittel zur Beschreibung der räumlichen Streuung zur Verfügung. Dabei wird jedoch nicht berücksichtigt, dass die Dispersion in verschiedene Richtungen vom Mittelzentrum aus nicht gleich sein muss. Die Standardabweichungsellipse stellt die Streuung nun als Ellipse um das arithmetische Mittelzentrum dar. Die Hauptachse der Ellipse zeigt die maximale, während die Nebenachse die minimale Streuung wiedergibt. Berechnet werden muss also neben dem Schwerpunkt des Clusters die Länge der Haupt- und Nebenachse sowie der Rotationswinkel. Die Standardabweichungsellipse wird von Ausreissern weniger beeinflusst als die konvexe Hülle. Sie ist als räumlicher Dispersionsparameter statistisch gut abgestützt.

Ein Nachteil aller flächenhaften Darstellungen sind die im Vergleich zu lokalen Signaturen kleineren Möglichkeiten zur Variation der graphischen Variablen. Grösse, Richtung und Form sind festgelegt, verändert werden können nur Farbe, Muster und Helligkeit. Damit wird es schwieriger, Objektqualitäten darzustellen. Beachtet werden sollten auch die Ränder der Flächen. Meiner Ansicht nach kann besonders bei der konvexen Hülle der Eindruck entstehen, man könne Gebiete mit Tieren gegen solche ohne deutlich abgrenzen, was natürlich nicht der Fall ist. Möglicherweise könnte man mit einem abgestuften Übergang der Fläche zur Umgebung diesem Eindruck entgegenwirken.

5 Resultate

In diesem Kapitel werden zuerst die Ergebnisse der verschiedenen Clusteringverfahren miteinander verglichen. Untersuchungen zum Zeitbedarf der Generalisierung dienen dem Abklären der Eignung von Clusteringverfahren zur Echtzeit-Generalisierung. Dabei ist die Verwendung innerhalb der für diese Arbeit aufgebauten Architektur berücksichtigt. Vor der Evaluation der automatischen Bestimmung einer geeigneten Clusterzahl wird die Verwendung von Clustering zur Modellgeneralisierung angesprochen.

5.1 Evaluation des Clusterings

Im Folgenden wird untersucht, welches Clusteringverfahren mit den verwendeten Daten die besten Resultate erbringt. Dazu müssen Kriterien definiert werden, die zur qualitativen Auswertung der Generalisierung Verwendung finden. Untersucht wird auch, welche Daten verwendet werden können.

5.1.1 Kriterien zur qualitativen Auswertung

Die Visualisierung von Wildtierbeobachtungsdaten verfolgt hier den Zweck, einem Parkbesucher Informationen über bevorzugte Tierstandorte zu liefern. Das Ziel ist, einem Besucher Gebiete zu zeigen, wo er grössere Chancen hat, eine bestimmte Tierart zu beobachten. Das Resultat der Generalisierung sollte also folgende Eigenschaften erfüllen:

- Wenn Habitate erkennbar sind, d.h. Gebiete in denen viele Beobachtungen dieser Tierart gemacht werden, sollen diese zu Clustern zusammengefasst werden. Idealerweise sind die Cluster ähnlich denjenigen, die man manuell bestimmen würde.
- Flächen, in denen keine Beobachtungen gemacht wurden, sollen möglichst nicht zu einem Cluster gehören.

Eine Prüfung, ob die gefundenen Gruppen über einen Zeitraum verallgemeinert werden können, ist nicht das Ziel dieser Arbeit. Die verwendeten Daten entstammen vier Beobachtungstagen pro Jahr und es ist fraglich, ob auf dieser Basis bevorzugte Standorte, falls solche überhaupt existieren, ermittelt werden können. Beispielsweise ist eine, je nach Tierart verschieden starke, saisonale Wanderung zu erwarten. Es ist klar, dass es für einen Parkbesucher nie eine Garantie geben kann, ein bestimmtes Tier zu sehen. Die Vermittlung von Informationen über beobachtete Tierstandorte kann mithelfen, die Chancen dafür zu erhöhen.

5.1.1.1 Kritische Anmerkungen

Der oben diskutierte Ansatz, räumliche Muster, in diesem Fall Habitate, aufgrund der Geometrie der

Beobachtungen zu ermitteln, soll an dieser Stelle eingehender betrachtet werden. Ein Ziel der Generalisierung ist es, räumliche Beziehungen zu erhalten. Als Beispiel kann man sich zwei Habitate einer Tierart vorstellen, die durch einen Fluss getrennt sind. Werden die Beobachtungen generalisiert, sollte die Trennung durch den Fluss erhalten bleiben (Abb. 5.1). Bei der Anwendung von Clusteringverfahren, die nur auf der Geometrie der Beobachtungen arbeiten, ist der Erhalt von solchen charakteristischen Eigenschaften nicht garantiert.

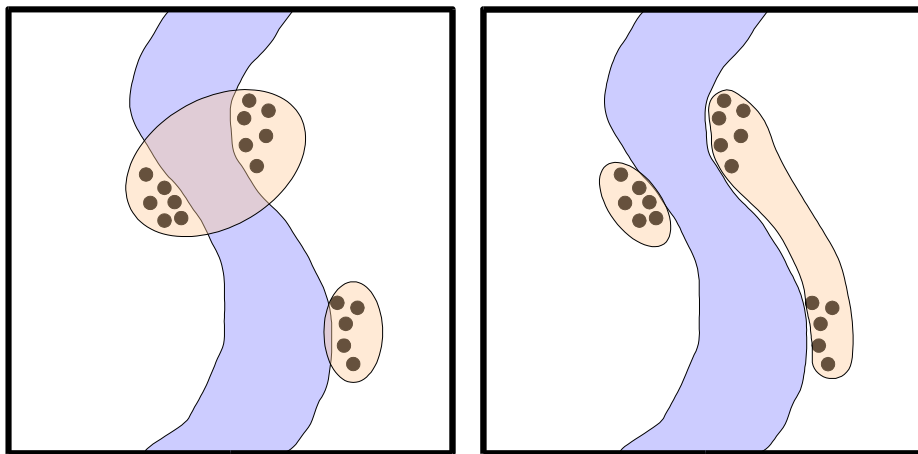


Abb. 5.1: Aufteilung von Beobachtungen in zwei Cluster: Links ohne Berücksichtigung der Trennung durch einen Fluss; rechts bleibt diese charakteristische Trennung erhalten.

Mit der Beschreibung von räumlichen Mustern der Variation wird die Struktur der räumlichen Autokorrelation beschrieben (O'Sullivan und Unwin 2003: 29 u. 65). Um diese zu analysieren, spaltet man sie in zwei Teile auf: Von räumlicher Variation *erster Ordnung* spricht man, wenn die Verteilung der Beobachtungen von Unterschieden in der Umwelt bestimmt ist. Beispielsweise ist eine Pflanze dort häufig anzutreffen, wo sie geeignete Bedingungen bezüglich Temperatur, Licht, Wasser usw. vorfindet. Die räumliche Variation *zweiter Ordnung* dagegen beruht auf lokalen Interaktionen zwischen den Beobachtungen. Wenn man eine bestimmte Pflanze findet, ist aufgrund der Fortpflanzung die Wahrscheinlichkeit gross, in der Nähe noch weitere Exemplare dieser Art zu entdecken. In der Realität ist eine Trennung dieser beiden Typen schwierig.

Um charakteristische Beziehungen zu erhalten, ist eine Untersuchung nötig, welche die Ursachen der räumlichen Verteilung aufzeigt. Das Ziel ist die Analyse der Variation erster Ordnung, d.h. die Identifikation von Faktoren, welche die Grenzen der Habitate bestimmen. Die ermittelten Grenzen können dann als Randbedingungen bei der Generalisierung berücksichtigt werden, womit eine Verbesserung der Gruppenunterteilung möglich sein sollte. Eine solche Analyse wurde für diese Arbeit nicht vorgenommen. Bei der Auswertung der gefundenen Cluster wird deshalb nicht darauf geachtet, ob die entstandenen Gruppen auch biologisch sinnvoll sind.

5.1.2 Vergleich der Algorithmen

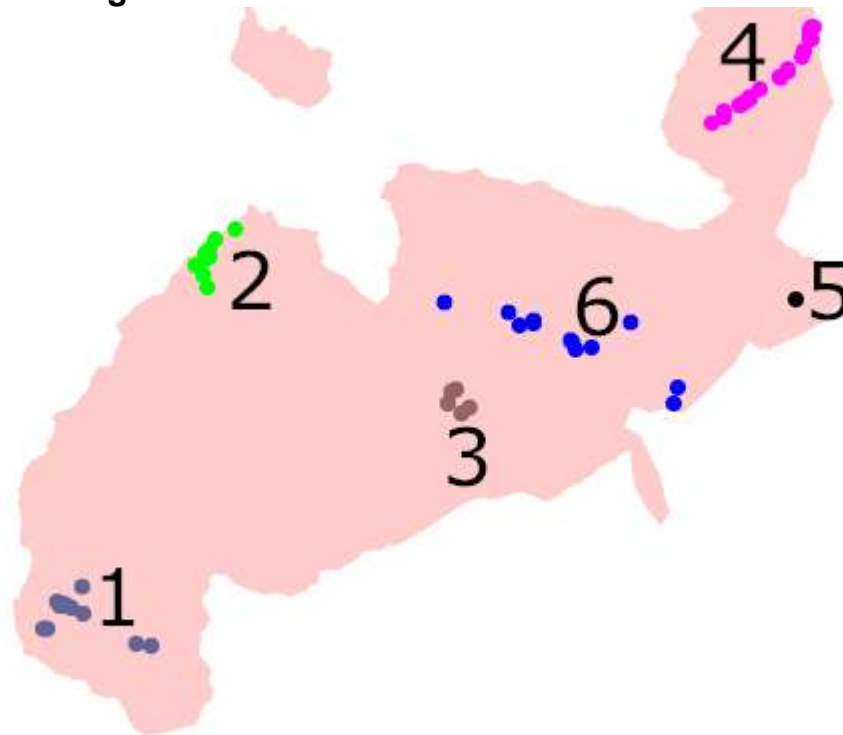


Abb. 5.2: Vorschlag für eine manuelle Aufteilung der Beobachtungen von Mardern in sechs Cluster.

In einer ersten Untersuchung soll festgestellt werden, wie die verwendeten Verfahren die Daten unterteilen, wenn eine Clusterzahl vorgegeben wird. Dazu sollen die Resultate mit einem manuell durchgeführten Clustering verglichen werden. Als Datensatz dienten die Beobachtungen von Mardern im ganzen Park. Die Darstellung der Cluster erfolgt mittels der konvexen Hülle, damit klar ersichtlich ist, welche Punkte zu einem Cluster gehören. Zu Vergleichszwecken sind die Originalpunkte ebenfalls eingezeichnet.

Als Vergleichsobjekt dient eine manuell durchgeführte Unterteilung in sechs Cluster, die in Abb. 5.2 gezeigt wird. Zur Orientierung ist jeweils die Grenze des Nationalparks eingezeichnet.

5.1.2.1 Hierarchisches Clustering

Das Single Linkage Verfahren ermittelt dieselben Cluster, die auch bei der manuellen Unterteilung von Abb. 4.3 entstehen.

Wird das hierarchische Clustering mit Complete Linkage durchgeführt, resultieren leicht andere Cluster. Hier wird der Cluster sechs (vgl. Abb. 4.3) in zwei Gruppen aufgeteilt, wobei eine auch die Punkte von Cluster drei beinhaltet

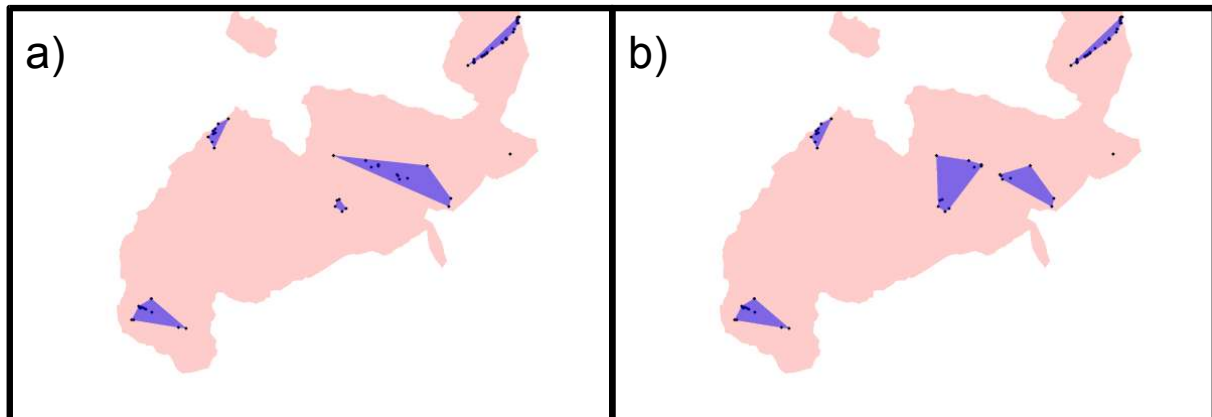


Abb. 5.3: Beobachtungen von Mardern zwischen 1995 und 2002. Berechnet wurden sechs Cluster mit a) Single Linkage und b) Complete Linkage.

Gemäss den oben festgelegten Kriterien für die qualitative Auswertung liefern beide Verfahren gute Ergebnisse. Gebiete, in denen häufig Tiere gesichtet wurden, werden zusammengefasst, während Gebiete ohne Beobachtungen zu keinem Cluster gehören. Die beiden Linkage Verfahren ergeben nur kleine Unterschiede in der Generalisierung, die darin begründet sind, dass Single Linkage zur Verkettung von Clustern neigt und Complete Linkage eher gleich grosse Cluster bildet.

5.1.2.2 Maschenvereinfachung

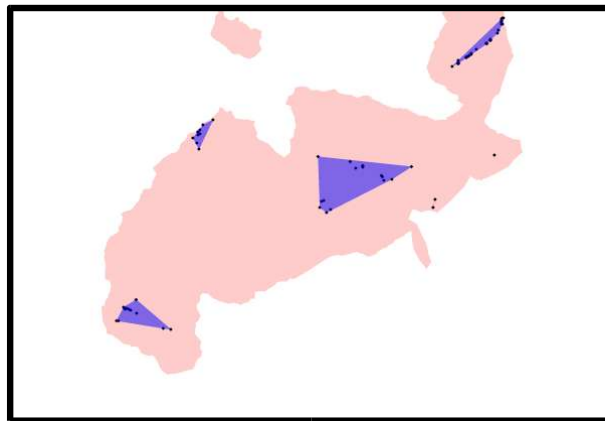


Abb. 5.4: Beobachtungen von Mardern zwischen 1995 und 2002, generalisiert mit Maschenvereinfachung. Bei der Generalisierung mittels Maschenvereinfachung werden vom Cluster sechs zwei Punkte am rechten Rand abgetrennt. Der Rest des Clusters bildet zusammen mit den Punkten von Cluster drei eine Gruppe. Dieses Verfahren liefert ebenfalls gute Ergebnisse.

5.1.2.3 K-means

Die Resultate dieses Algorithmus weichen bereits deutlicher vom Vergleichsobjekt ab. Der Cluster vier wird in zwei Gruppen aufgespalten und ein Teil von Cluster sechs mit den Punkten des Clusters drei vereinigt. Ungünstig ist vor allem die Gruppe, die den Einzelpunkt fünf beinhaltet. Dieser

Cluster besitzt eine grosse Fläche, die aber nur wenige Beobachtungen beinhaltet. Dies vermittelt einen falschen Eindruck der räumlichen Verteilung, da ein Nutzer überall in diesen Flächen Tiere vermutet, obwohl im grössten Teil davon nie welche gesehen wurden. Der Grund liegt wahrscheinlich in der Verwendung des arithmetischen Mittelpunkts (Zentroid) als Repräsentant eines Clusters, der stark von Ausreissern beeinflusst wird.

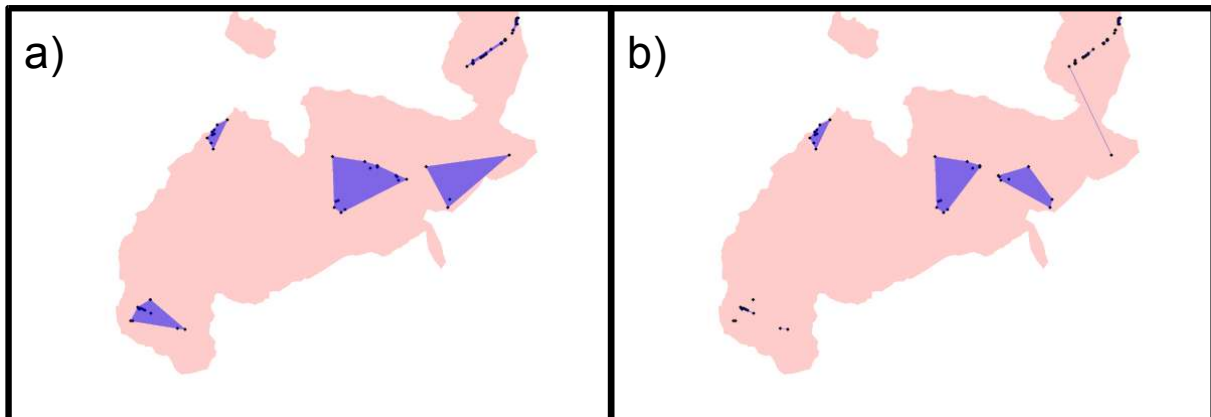


Abb. 5.5: Beobachtungen von Mardern zwischen 1995 und 2002. a) zeigt sechs Cluster, die mit dem k-means Algorithmus berechnet wurden, während b) die Resultate eines graphbasiertes Clusterings mit dem Minimum Spanning Tree und den Parameterwerten $p_1 = 1.2$ und $p_2 = 2$ wiedergibt.

Es soll hier noch einmal erwähnt werden, dass dieses Verfahren nicht deterministisch ist, die Resultate also also stark von der Startkonfiguration abhängen. Dies führt dazu, dass die Qualität der Ergebnisse sehr unterschiedlich ist.

5.1.2.4 Graphbasiertes Clustering mit dem Minimum Spanning Tree

Zu diesem Verfahren ist anzumerken, dass hier keine feste Clusterzahl vorgegeben werden kann. Von den verglichenen Algorithmen liefert es deutlich die schlechtesten Ergebnisse. Die meisten manuell gefundenen Cluster sind hier mehrmals unterteilt, im Gegenzug sind weit auseinander liegende Objekte zu einem Cluster zusammengefasst. Weitere Versuche mit veränderten Parameterwerten lieferten vergleichbare Resultate.

Das verwendete Kriterium zur Aufspaltung des Minimum Spanning Trees in einen Wald scheint für die Anwendung mit diesen Daten ungeeignet. Die Ursache dürfte darin liegen, dass der Abstand zwischen einigen Punkten sehr klein ist. Berechnet man nun die mittlere Länge und Standardabweichung von angrenzenden Kanten, sind diese ebenfalls klein. Kanten zwischen Punkten die, über das Ganze betrachtet, nahe beieinander liegen können deshalb um ein Vielfaches länger sein, als ihre angrenzenden Kanten und werden deshalb gelöscht. Je höher man aber die Parameter wählt, welche die Grenze festlegen, desto grösser ist auch die Wahrscheinlichkeit, dass weit auseinander liegende Punkte zu einem Cluster vereinigt werden.

5.1.2.5 Versuche mit weiteren Daten

Um die gezogenen Schlussfolgerungen zu bestätigen, wurden weitere Daten mit den verschiedenen Verfahren generalisiert. Ein Versuch mit Beobachtungen von Steinböcken im Gebiet des Ofenpasses bestätigt die gewonnenen Erkenntnisse (vgl. Abb. 4.9). Die Resultate der Generalisierung fallen bei diesem Beispiel aber insgesamt besser aus.

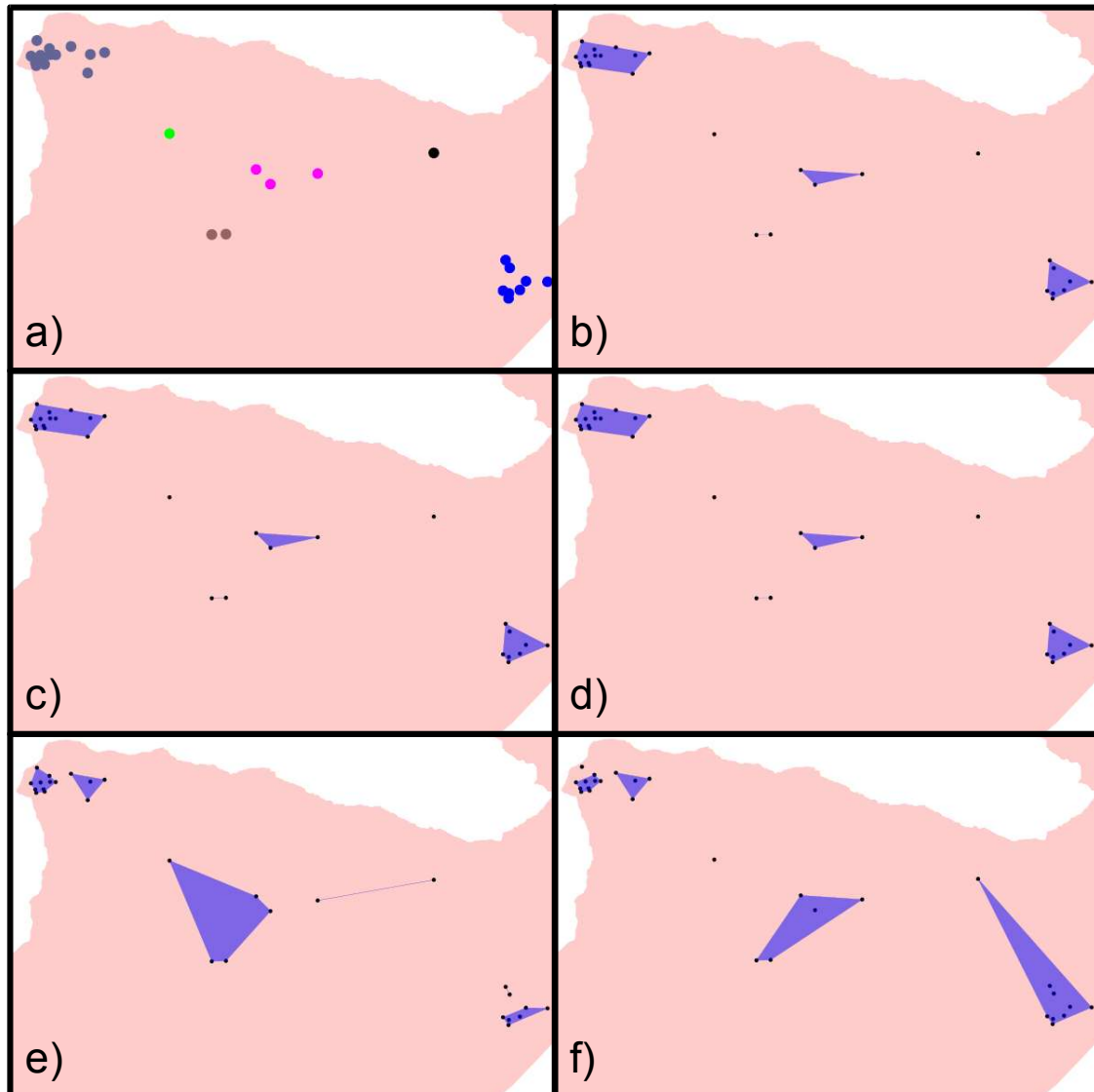


Abb. 5.6: Beobachtungen von Steinböcken im Januar zwischen 1998 und 2002 im Gebiet Ofenpass. a) zeigt die manuelle Unterteilung mit der die Resultate der Generalisierung verglichen wurden. Die Resultate der verschiedenen Generalisierungsverfahren sind mit konvexen Hüllen dargestellt: b) hierarchisch, Single Linkage; c) hierarchisch, Complete Linkage; d) Maschenvereinfachung; e) k-means; f) graphbasiertes Clustering mit dem Minimum Spanning Tree ($p_1=1.2$; $p_2=2$). Berechnet wurden bei b) bis e) jeweils sechs Cluster.

Die hierarchischen Verfahren sowie die nach dem gleichen Prinzip aufgebaute Generalisierung mittels Maschenvereinfachung liefern alle dieselbe Unterteilung in Gruppen, wie sie auch manuell vorgenommen wurde. Der k-means Algorithmus teilt die Punkte oben links und unten rechts jeweils

in zwei Cluster auf. Im Zentrum der Karte gibt es zwei grossflächige Cluster mit wenigen Beobachtungen. Das Clustering mit dem Minimum Spanning Tree liefert hier etwas bessere Resultate als beim vorhergehenden Beispiel. Negativ ist das Aufteilen der Punkte links oben in mehrere Cluster und der grosse Cluster rechts unten, der einen weit entfernten Einzelpunkt integriert.

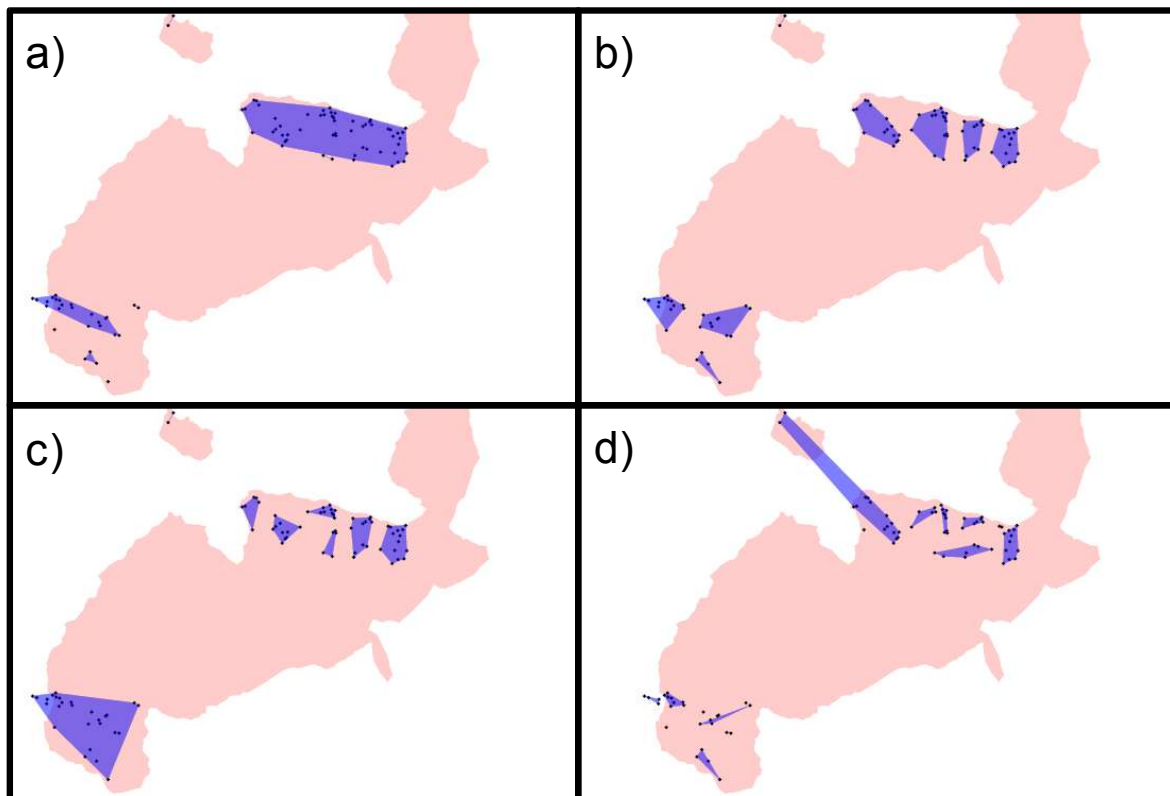


Abb. 5.7: Beobachtungen von Gemen im August 2002 und durch verschiedene Algorithmen ermittelte Cluster. a) Hierarchisch, Single Linkage; b) Hierarchisch, Complete Linkage; c) K-means; d) Graphbasiertes Clustering mit dem Minimum Spanning Tree. Berechnet wurden bei a), b) und c) jeweils acht Cluster.

Ein weiteres Beispiel zeigt die Grenzen der Anwendung von Clusteringverfahren auf. Verwendet wurden dabei die Beobachtungen von Gemen im August 2002 im ganzen Parkgebiet. Es ist anzumerken, dass die Beobachtungen von Huftieren nur in zwei Gebieten des Parks kartiert werden. Bei einem Kartennutzer ohne dieses Wissen kann der Eindruck entstehen, die Tiere würden nur in diesen zwei Gebieten vorkommen. Dieses Missverständnis kann durch das Zeichnen von Clustern als Polygone noch verstärkt werden. Dieses Beispiel zeigt, dass Daten existieren, die für die Generalisierung mit Clusteringverfahren weniger geeignet sind als andere.

Auf eine manuelle Unterteilung wurde für dieses Beispiel verzichtet. Das Single Linkage Verfahren belässt die Punktansammlungen weitgehend im selben Cluster. Daneben bilden wenige Punkte, die abseits liegen, eigene Cluster. Besonders der grosse Cluster oben bildet in erster Linie das Gebiet ab, in dem überhaupt Beobachtungen kartiert wurden. In welchen Teilen davon sich die Tiere tatsächlich

aufhalten, lässt sich nicht herausfinden, wenn die Originalpunkte nicht eingezeichnet wären. Complete Linkage teilt diese grossen Cluster in kleinere auf. Allerdings vermitteln auch diese Cluster keinen genauen Eindruck, wo die Wahrscheinlichkeit einer Gemsbeobachtung höher ist. Das k-means Verfahren unterteilt das obere Beobachtungsgebiet am Ofenpass ebenfalls in kleinere Cluster auf. Das untere Gebiet bildet jedoch eine einzige grosse Gruppe. Das graphbasierte Clustering erzeugt im oberen Teil einige brauchbare Cluster, die anderen Gruppen bestätigen aber die oben gezogenen Schlussfolgerungen.

Zusammenfassend lässt sich sagen, dass die für diesen dritten Datensatz berechneten Cluster nur eine kleine Hilfe bei der Beobachtung von Tieren sein können. Viele beinhalten zu grosse Flächen ohne Beobachtungen und genügen deshalb den Anforderungen nicht. Damit bestätigt sich die Vermutung, dass gewisse Daten für diese Art der Generalisierung eher ungeeignet sind. Man muss aber berücksichtigen, dass dies auch vom betrachteten Ausschnitt abhängt. Schaut man nur einen Ausschnitt derselben Daten an (vgl. Abb. 4.11), können dort möglicherweise sinnvolle Cluster berechnet werden.

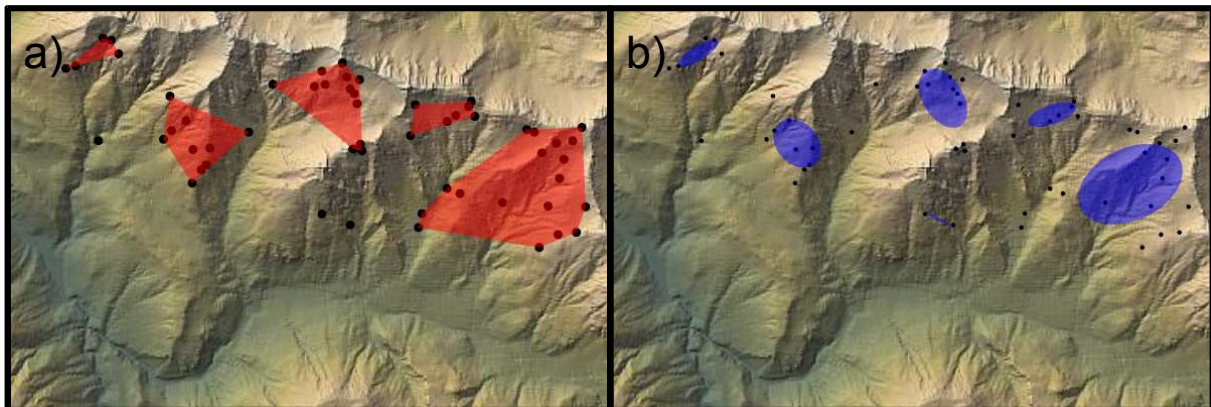


Abb. 5.8: Beobachtungen von Gemsen im August 2002 im Gebiet II Fuorn. Berechnet wurden mit hierarchischem Clustering (Single Linkage) acht Cluster, die mit a) konvexen Hüllen und b) Standardabweichungsellipsen dargestellt sind.

Verbessert werden kann die Informationsvermittlung vor allem durch eine geeignete Symbolisierung. Werden Cluster durch Standardabweichungsellipsen anstatt durch die konvexe Hülle repräsentiert, beeinflussen einzelne Ausreisser den Gesamteindruck viel weniger. Dadurch kann eher vermieden werden, dass durch Cluster mit grossen beobachtungsfreien Flächen ein falsches Bild der Tierstandorte vermittelt wird.

Das Beispiel von Abb. 5.8 verdeutlicht die oben angesprochenen Mängel der Generalisierung. Es werden keine Randbedingungen wie beispielsweise Geländekammern berücksichtigt. Die berechneten Cluster stellen daher Habitate dar, die mit der biologischen Realität nicht zwingend übereinstimmen.

5.2 Zeitbedarf

Es ist ein Ziel dieser Arbeit, Clusteringverfahren auf ihre Verwendbarkeit in einer Echtzeit-Generalisierung zu testen. Entscheidend dabei ist unter anderem die Zeit, die für die Generalisierung benötigt wird.

Wenn ein Benutzer im Internet eine Karte wünscht, sollte ihm diese in kurzer Zeit geliefert werden. Ist das nicht der Fall, wird der Nutzer ungeduldig und bricht die Übertragung ab. Um nun die Leistungsfähigkeit der verwendeten Algorithmen einerseits und der aufgebauten Architektur zur dynamischen kartographischen Visualisierung andererseits zu untersuchen, wurden Zeitmessungen durchgeführt.

Die Messungen wurden lokal auf einem PC mit einem Pentium 4 Prozessor mit 1,6 GHz und 512 MB Arbeitsspeicher durchgeführt. Nicht berücksichtigt ist dabei die Zeit für die Übertragung der Daten über ein Netz zum Client, wo die Darstellung erfolgt. Die Messungen geben aber trotzdem einen Eindruck von den Möglichkeiten und Grenzen der Architektur und der Generalisierung mit Clusteringverfahren.

5.2.1 Generalisierung und Visualisierung

In einer ersten Messung wurde untersucht, wie lange die Generalisierung und Visualisierung einer GML-Datei von der Kommandozeile aus dauert. Untersucht wurde also, wie lange das XSLT-Stylesheet, inklusive Benutzung der Generalisierungskomponente, braucht, um GML in SVG umzuwandeln. Der Zeitverbrauch wurde mit der Java-Methode `System.currentTimeMillis()` gemessen, die jeweils am Anfang und Ende der Umwandlung aufgerufen wurde. Die benötigte Zeit ergibt sich dann als Differenz der beiden gemessenen Zeiten.

Aus den Beobachtungsdaten der Gelsen im SNP wurden Testdatensätze generiert, indem jeweils eine bestimmte Anzahl Punkte zufällig ausgewählt wurde. Mit diesen Daten wurde anschliessend der Zeitaufwand für Generalisierung und Visualisierung mit den verschiedenen implementierten Algorithmen ermittelt. Da die beiden hierarchischen Algorithmen genau den gleichen Rechenaufwand benötigen, wurde hier nur Single Linkage berücksichtigt. Berechnet wurden jeweils zehn Cluster, mit Ausnahme des Datensatzes mit 25 Punkten (fünf Cluster), unabhängig davon, ob diese Anzahl für die ausgewählten Daten sinnvoll wäre. Nicht berücksichtigt ist in dieser Messung auch das vorgängige Bestimmen einer geeigneten Anzahl Cluster. Dieses benötigt zusätzlich die Zeit, die für das hierarchische Clustering gebraucht wird, da sie auf diesem aufgebaut ist.

Bei kleinen Datensätzen bewegen sich die Leistungen der Verfahren in einem ähnlichen Bereich, während bei grösseren Datensätzen nur k-means in einer sinnvollen Zeit Cluster berechnet. Dieses

Resultat ist auch zu erwarten, wenn man die Laufzeitverhalten der Algorithmen betrachtet. Zum guten Abschneiden des k-means-Verfahren ist anzumerken, dass man den Algorithmus mehrmals mit unterschiedlichen Anfangskonfigurationen starten müsste, um eine möglichst gute Lösung zu erhalten. Dies würde aber die gemessene Zeit vervielfachen.

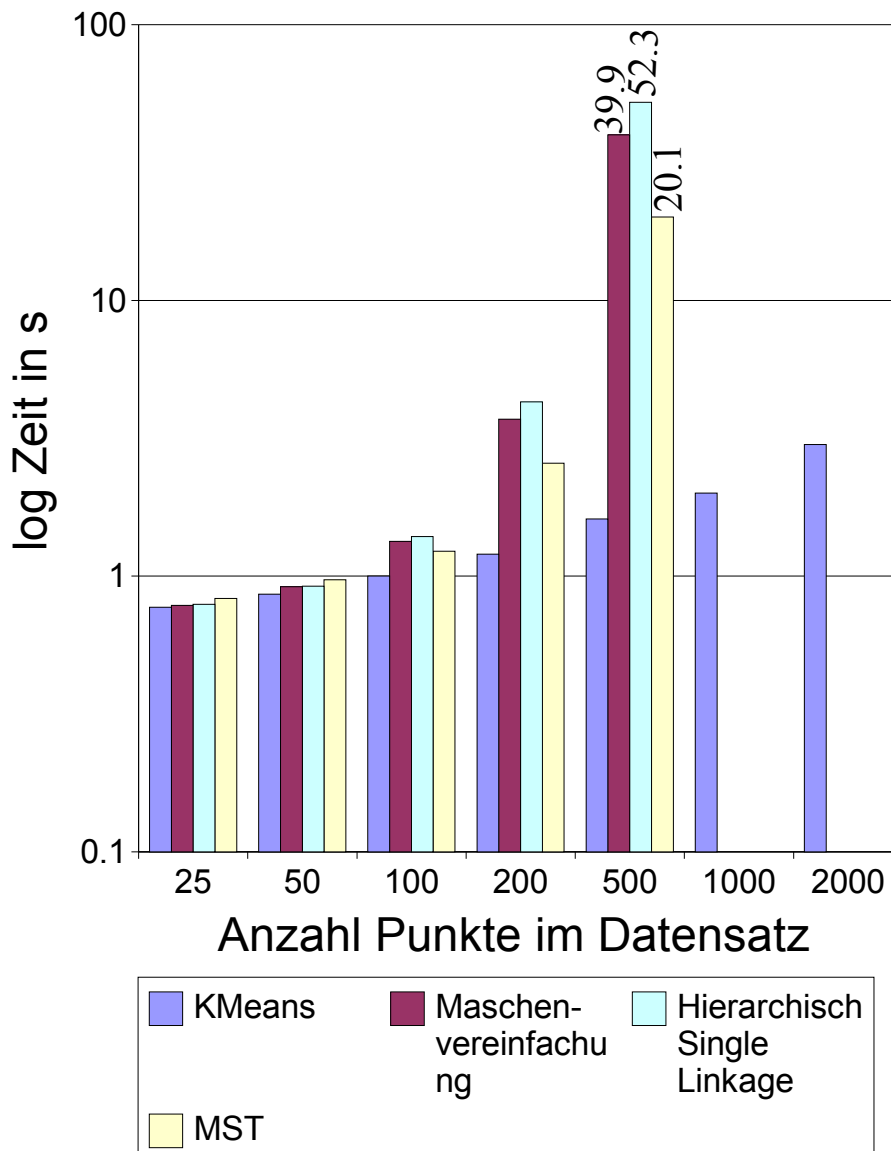


Abb. 5.9: Zeitbedarf für Generalisierung und Visualisierung mittels XSLT in Abhängigkeit der Grösse des verwendeten Datensatzes und des verwendeten Clusteringverfahrens.

Für die Wartezeit eines Benutzers auf eine Karte muss zusätzlich zum Zeitbedarf für Generalisierung und Visualisierung auch derjenige für andere Aufgaben berücksichtigt werden: Die Umwandlung der Daten in GML durch den Web Feature Server, die Übertragung der Daten zum Nutzer und das Zeichnen der Daten auf dem Client. Zählt man diese Zeit zur Gemessenen hinzu, kann man eine Anzahl Punkte bestimmen, die verarbeitet werden kann ohne dass die Wartezeit für einen Nutzer zu gross wird. Diese Grenze liegt zwischen 200 und 500 Punkten. Es gibt verschiedene Reaktionen,

wenn ein Benutzer bei der Auswahl der Karte einen Suchbereich angibt, in dem zuviele Punkte ausgewählt werden. Entweder wird dem Benutzer eine Warnung ausgegeben, die ihm die ungefähre Zeit zur Berechnung der Karte mit so vielen Punkten mitteilt, und ihm die Möglichkeit lässt, seine Auswahl zu ändern. Eine Möglichkeit besteht auch darin, ab einer bestimmten Anzahl Punkte ein anderes Generalisierungsverfahren einzusetzen.

Eine Möglichkeit, diese Generalisierung für eine grössere Anzahl Punkte in Echtzeit zu ermöglichen, besteht in der Aufteilung des Untersuchungsgebietes. Die Generalisierung wird danach für jedes Teilgebiet einzeln durchgeführt.

5.2.2 Architektur mit Server

In einem zweiten Versuch wurde die Zeit gemessen, die Generalisierung und Visualisierung benötigen, wenn sie innerhalb der in Kapitel drei vorgestellten Architektur eingesetzt werden. Gemessen wurde die Zeit, die das Servlet, das den Prozess der Generalisierung und Visualisierung der Daten steuert, für die Bearbeitung einer Anfrage benötigt.

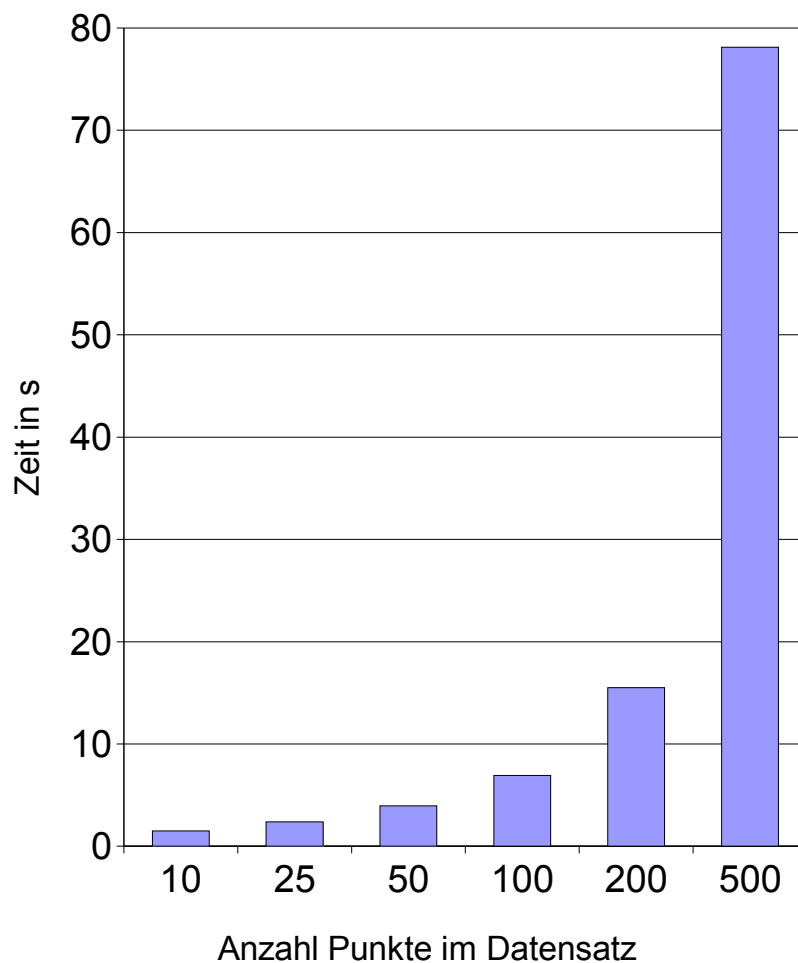


Abb. 5.10: Zeitbedarf für Generalisierung und Visualisierung innerhalb der aufgebauten Architektur. Zur Generalisierung wurde hierarchisches Clustering (Single Linkage) verwendet.

Verwendet wurden die gleichen Datensätze wie bei der vorhergehenden Messung. Untersucht wurde

in diesem Fall aber nur das hierarchische Clustering: Einerseits benötigt die Ermittlung der Cluster mit diesem Verfahren mehr Zeit als bei den anderen Methoden, die in dieser Arbeit verwendet wurden. Andererseits scheint das hierarchische Verfahren die besten Resultate zu liefern. Es wurde gleich wie oben mit einer gegebenen Zahl von Clustern gearbeitet.

Die Resultate dieser Messung bestätigen die Erkenntnisse des vorhergehenden Kapitels. Die Wartezeit auf die Karte scheint akzeptabel für Datensätze mit einer Obergrenze zwischen 200 und 500 Punkten.

5.3 Einsatz für Modellgeneralisierung

Eine Möglichkeit, die Zeit zur Berechnung einer Karte zu verkürzen, ist eine vorgängige Generalisierung und das Speichern der Ergebnisse in hierarchischen Strukturen (van Oosterom 1995). Die Zeit, die ein System zur Berechnung einer Karte braucht, kann so verkürzt werden, weil das Generalisierungsverfahren nicht mehr komplett ausgeführt werden muss.

Die Ergebnisse eines hierarchischen Clusteringverfahrens können als Dendrogramm dargestellt werden. Dieses Dendrogramm ist ein binärer Baum. Sind die gruppierten Objekte Punkte im Raum, kann man das Dendrogramm als hierarchische Datenstruktur betrachten, die auf einer objektgesteuerten Zerlegung des Raumes basiert. Ein binärer Baum kann relativ einfach als rekursive Datenstruktur implementiert werden. Die hierarchische Struktur des Baums erlaubt es, auf die verschiedenen Aggregationsschritte zuzugreifen; man kann die Anzahl der gezeigten Objekte festlegen und die entsprechende Stufe der Aggregation darstellen.

5.4 Evaluation der Bestimmung der Clusterzahl

Das Ermitteln einer sinnvollen Zahl von Clustern ist ein wichtiger Bestandteil einer automatischen Generalisierung, die sich auf Clusteringverfahren abstützt. In Kapitel 2.4.3 wird eine einfache Methode zur Bestimmung dieser Anzahl vorgeschlagen. Inwiefern sie sich im Einsatz bewährt, wird im Folgenden untersucht.

Es muss festgehalten werden, dass bei der Verwendung von realen Daten normalerweise nicht eine einzige Zahl als Lösung in Frage kommt. Die Lösung besteht vielmehr aus einem Intervall, aus dem eine Lösung ausgewählt wird. Illustriert wird dies durch Abb. 4.14, die verschiedene Lösungen miteinander vergleicht. Welche davon als optimal erachtet wird, hängt auch von persönlichen Vorlieben ab.

Um die automatische Bestimmung der Clusterzahl zu testen, wurden aus den Daten eine Reihe von Testfällen generiert. Vor der Generalisierung wurde für jeden dieser Fälle manuell eine sinnvolle Clusterzahl bestimmt. Diese wird anschliessend mit der automatisch Ermittelten verglichen.

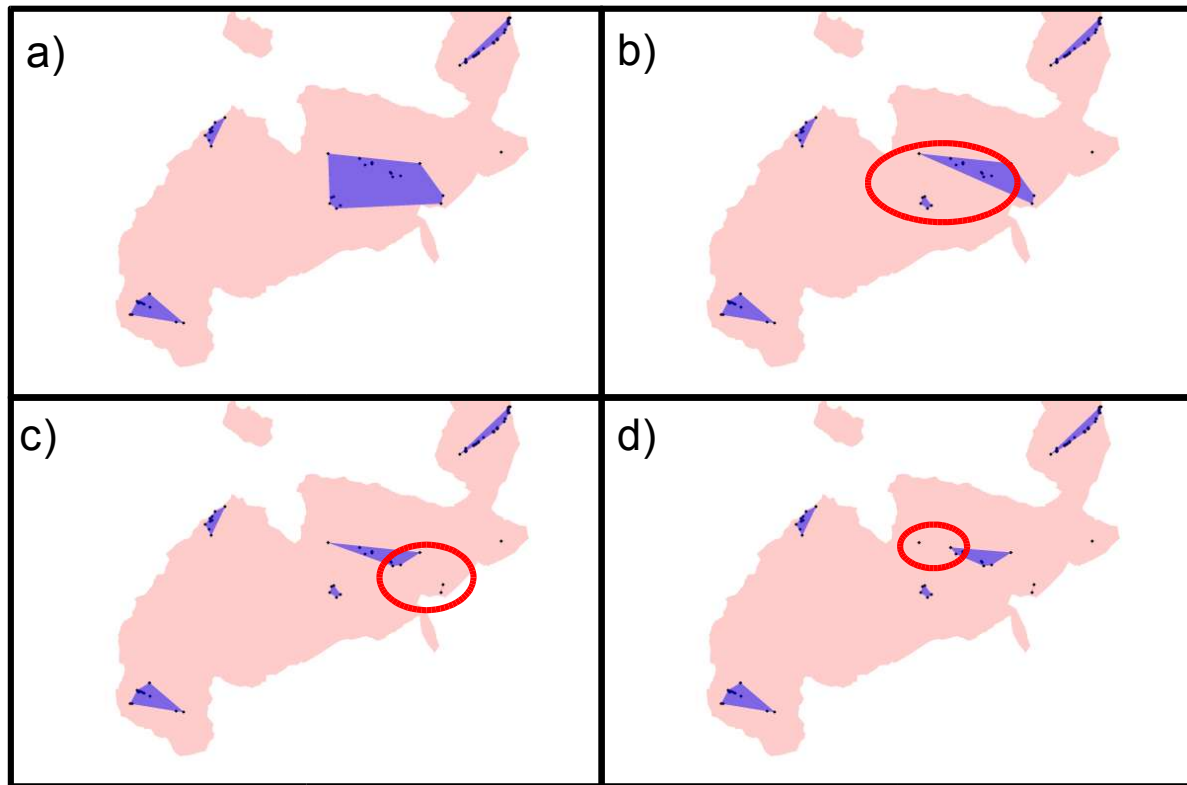


Abb. 5.11: Mit Single Linkage berechnete Cluster in Beobachtungen von Mardern. a) zeigt fünf, b) sechs, c) sieben und d) acht Cluster. Rot hervorgehoben sind die Cluster, die bei der Verkleinerung der Clusterzahl vereinigt werden.

Bis auf zwei Fälle ist die automatisch ermittelte Zahl stets tiefer als die manuelle. Der Mittelwert der Differenzen zwischen den beiden Werten beträgt -1.27 , wobei zu beachten ist, dass die Unterschiede zwischen den Testfällen gross sind. In machen Fällen ist der automatisch bestimmte Wert bloss um eins tiefer, während in anderen die Differenz fünf Cluster ausmacht. Bei der kleinen Zahl an Clustern in den Beispielen ist dieser Unterschied sehr gross.

Berücksichtigt man, dass die optimale Clusterzahl, wie oben erwähnt, subjektiv ist, liefert das Verfahren in den meisten Fällen befriedigende Resultate. In einzelnen Fällen sind die Ergebnisse aber ungenügend. Da die berechneten Werte meistens zu klein sind, könnte vielleicht eine Korrektur um einen bestimmten Faktor bessere Resultate liefern. Dazu wären aber ausführlichere Tests, insbesondere mit einer grösseren Anzahl von Clustern notwendig.

Tierart	Beobachtungen	Gebiet	Anzahl automatisch	Anzahl manuell	Differenz
Gemse	August; 1997-2002	Il Fuorn	2	1	1
Gemse	August; 2002	Trupchun	7	5	2
Gemse	August; 2002	Ganzer Park	2	3	-1
Steinbock	Mai, August; 1997-2002	Il Fuorn	3	6	-3
Steinbock	Januar; 1997-2002	Il Fuorn	5	6	-1
Steinbock	Mai; 1997-2002	Il Fuorn	4	5	-1
Steinbock	November; 1997-2002	Il Fuorn	2	7	-5
Rothirsch	August; 1997-2002	Il Fuorn	2	5	-3
Rothirsch	Oktober, November; 1997-2002	Ganzer Park	2	3	-1
Marder	1995-2002	Ganzer Park	2	6	-4
Marder	1995-2002	Il Fuorn	6	4	2
				Mittelwert	-1.27

Tab. 4.1: Vergleich von automatisch und manuell ermittelten Clusterzahlen für verschiedene Testfälle.

5.5 Diskussion der Resultate

Die verschiedenen für diese Arbeit durchgeführten Versuche haben gezeigt, dass eine Anwendung von Clusteringverfahren für die automatische Generalisierung von Punktdaten möglich ist. Die Qualität der berechneten Unterteilungen unterscheidet sich je nach verwendetem Verfahren deutlich. Erwartungsgemäss unterscheidet sich auch der Rechenaufwand für die einzelnen Clusteringmethoden.

- Das Single Linkage Verfahren lieferte beim Vergleich mit manuellen Unterteilungen die besten Ergebnisse. Die berechneten Cluster sind gemäss den in Kap. 4.3.0.1 definierten Anforderungen von guter Qualität. Allerdings benötigt diese Methode von allen getesteten den höchsten Rechenaufwand.
- Complete Linkage liefert ebenfalls gute Resultate, die denen von Single Linkage meist ähnlich sind. Die Unterschiede zu den manuellen Unterteilungen sind hier aber bereits grösser. Dieser Algorithmus benötigt den selben Aufwand zur Berechnung wie der vorhergehende.
- Um ein Verfahren zu haben, dass ähnliche Ergebnisse wie die hierarchischen Verfahren berechnet, aber schneller ist, wurde eine Generalisierung mit Maschenvereinfachung durchgeführt. Die Rechenzeit ist hier kürzer, allerdings nicht in einem riesigen Ausmass. Die berechneten Unterteilungen sind qualitativ gut.
- Mit dem k-means Algorithmus konnten nur mittelmässige Ergebnisse erzielt werden. Gruppen von Punkten werden oft in zwei oder mehr Cluster aufgespalten während weit voneinander entfernt liegende Punkte zu einem Cluster vereinigt werden. Ein Problem dieses Verfahrens ist, dass man für die gleiche Ausgangslage verschiedene Unterteilungen mit unterschiedlicher Qualität erhält, wenn man die Berechnung mehrmals durchführt. Um gute Resultate zu

erhalten, benötigt man also ein Mass, dass aus verschiedenen Lösungen zuverlässig die Beste auswählt. Ansonsten kann dieser Algorithmus nicht für eine unüberwachte Generalisierung verwendet werden. Durch das mehrmalige Rechnen erhöht sich auch der Rechenaufwand, der allerdings von allen verwendeten Verfahren mit Abstand am geringsten ist.

- Mit dem graphbasierten Clustering wurden die schlechtesten Ergebnisse erzielt. Für die Qualität der berechneten Cluster gilt das beim vorhergehenden Verfahren gesagte noch verstärkt. Das Problem ist wahrscheinlich das Kriterium, welches für die Unterteilung des Baums verwendet wurde. Es scheint für die Verwendung mit diesen Daten nicht geeignet zu sein.

Der Anwendung von Clusteringverfahren zur Generalisierung sind aber Grenzen gesetzt. Zum einen kann diese Generalisierung nicht für alle Punkte verwendet werden, sondern nur für den in Kapitel 2.2 vorgestellten zweiten Typ von Punktdaten, bei dem die Darstellung von der relativen Lage abhängig ist.

Eine zweite Einschränkung betrifft die Verwendbarkeit zur Echtzeit-Generalisierung. Die besten Ergebnisse wurden mit den aufwendigsten Verfahren erreicht; der dafür nötige Rechenaufwand bestimmt eine Obergrenze der Anzahl Punkte in einem Datensatz. Mit dem verwendeten Rechner lag diese Grenze für die hierarchischen Verfahren zwischen zwei- und fünfhundert Punkten. Da diese Berechnungen serverseitig gemacht werden sollen, dürfte sich diese Grenze nach oben verschieben.

Die dritte Einschränkung ist trivial: Die Punkte im zu generalisierenden Datensatz dürfen nicht zu regelmässig über die Fläche verteilt sein, damit überhaupt sinnvolle Cluster gefunden werden können. Dabei spielt aber auch der Massstab eine Rolle.

Eine Problem bleibt die Bestimmung einer geeigneten Zahl von Clustern, die als Abbruchbedingung für die Algorithmen verwendet wird. Es gibt zwar eine Reihe von Verfahren zur Berechnung dieser Anzahl, allerdings sind dieses oft sehr aufwendig. Für diese Arbeit wurde ein einfaches Verfahren umgesetzt, welches zufriedenstellend funktioniert. Da die berechneten Clusterzahlen jedoch fast durchwegs zu tief sind, bleibt hier noch Spielraum für Verbesserungen.

Eine Möglichkeit, wie die Generalisierung durch ein pre-processing eventuell beschleunigt werden kann, wurde kurz skizziert. Die Idee ist, das beim hierarchischen Clustering resultierende Dendrogramm als hierarchische Datenstruktur zu nutzen.

6 Schlussfolgerungen und Ausblick

6.1 Schlussfolgerungen

Für diese Arbeit wurde eine Architektur aufgebaut, die eine dynamische kartographische Visualisierung ermöglicht. Verwendet wurden Technologien, die auf offenen Standards des Open GIS Consortiums und des W3C beruhen. Eine Frage dieser Arbeit war, wie eine Generalisierung in dieses System integriert werden kann. Entschieden wurde, die Generalisierungskomponente als Dienst zu entwickeln, was einen flexiblen Einsatz ermöglicht. Generalisierungsdienste sind beim Open GIS Consortium als Idee skizziert, so dass in Zukunft mit einer Standardisierung gerechnet werden kann, auch wenn dazu noch einige Hindernisse zu überwinden sind.

Implementiert wurde der Generalisierungsdienst durch eine XSL-Transformation. Die XSLT-Stylesheets lesen die Daten und wandeln sie in Scalable Vector Graphics (SVG) um; für die Generalisierung wird mittels Erweiterungsfunktionen eine Gruppe von Java-Klassen genutzt, die für diese Arbeit implementiert wurden. Für die beschränkte Aufgabe, Punktdaten zu aggregieren, ist diese Lösung sehr gut geeignet. Für komplexere Aufgaben müssen andere Lösungen, wie z.B. ein direkter Zugriff von Java auf die GML, gewählt werden. Dies ist der Fall, wenn die Ausgangsdaten neben Punkten auch Linien und Flächen enthalten und weitere Generalisierungsoperationen möglich sein müssen.

Es war das Ziel, dass die aufgebaute Architektur für Location Based Services (LBS) verwendbar sein sollte. Vom technischen Standpunkt gesehen ist diese Verwendung möglich, da der mobile Client nur in der Lage sein muss, eine SVG-Datei darzustellen. Im Rahmen dieser Arbeit wurden jedoch keine Versuche mit mobilen Geräten gemacht. Ein wichtiges Element eines LBS, die aktuelle Position des Nutzers, ist in der Architektur bisher nicht berücksichtigt. Das verwendete System benötigt eine Reihe von Verbesserungen, damit eine Benutzung für LBS möglich ist.

Ein zweites Ziel dieser Arbeit war die Suche von Methoden für eine Echtzeitgeneralisierung von Punktdaten. Es wurde erkannt, dass es zwei Typen von Punkten gibt: Beim ersten Typ ist die kartographische Darstellung nur von der absoluten Lage abhängig. Der zweite Typ erfordert die Berücksichtigung der Beziehungen zu benachbarten Punkten; räumliche Muster der Verteilung sind hier wichtiger als die absolute Position einzelner Punkte. Eine Erkenntnis dieser Unterscheidung ist, dass die beiden Typen bei einer Generalisierung eine unterschiedliche Behandlung erfordern. Eine Aggregation ist beispielsweise nur beim zweiten Typ sinnvoll durchführbar.

Diese Arbeit konzentrierte sich auf die Realisierung des Generalisierungsoperators Typisierung: Eine Gruppe von Objekten wird ersetzt durch eine kleinere Gruppe, die ähnliche Eigenschaften bezüglich

Dichte, Orientierung usw. aufweist. Wichtig ist, dass Typisierung im Gegensatz zu anderen Verfahren, z.B. Auswahl (Selektion), den Kontext der Objekte berücksichtigt. Es wurde in dieser Arbeit versucht zu beantworten, ob sich Methoden der Clusteranalyse zur Realisierung einer Typisierung eignen. Zu diesem Zweck wurden eine Reihe von verschiedenen Verfahren implementiert: mehrere hierarchische, ein partitionierendes (k-means) und ein graphbasiertes Verfahren. Als Daten wurden Beobachtungen von verschiedenen Tieren des Schweizerischen Nationalparks verwendet.

Es wurde eine qualitative Auswertung der Ergebnisse der einzelnen Algorithmen durchgeführt. Dies geschah durch einen visuellen Vergleich der gefundenen Gruppen mit den Beobachtungen einerseits, mit manuell vorgenommenen Gruppeneinteilungen andererseits. In der Arbeit werden bereits die Mängel dieser Evaluation angesprochen: Ein Ziel der Generalisierung ist der Erhalt von räumlichen Beziehungen. Dies ist nicht garantiert, wenn man ein Clustering durchführt, das nur auf der Geometrie der Objekte basiert. Die Evaluation liefert Erkenntnisse über die Qualität der gefundenen Gruppen, aber nur beschränkt über die Qualität der Generalisierung.

Um zu prüfen, ob mit den Clusteringverfahren auch eine Generalisierung in Echtzeit möglich ist, wurden Zeitmessungen durchgeführt. Dabei stellte sich heraus, dass für relativ kleine Punktmengen die Zeit zur Berechnung einer Karte akzeptabel ist, für mehr Punkte aber stark wächst. Eine Möglichkeit, die Zeit zur Berechnung einer Karte zu verkürzen, ist eine vorgängige Generalisierung und das Speichern der Ergebnisse in hierarchischen Strukturen. In Kap. 5.3 wird die Verwendung des Dendrogramms des hierarchischen Clusterings zu diesem Zweck skizziert. Ein Problem bei der Nutzung von Clustering zur automatischen Generalisierung ist die Bestimmung einer geeigneten Anzahl von Clustern. Für diese Arbeit wurde eine einfache Lösung umgesetzt, die Evaluation zeigt allerdings, dass hier noch Verbesserungspotential vorhanden ist.

Zusammenfassend lässt sich sagen, dass Verfahren der Clusteranalyse durchaus für eine Generalisierung eingesetzt werden können. Ein grosser Vorteil ist, dass der Kontext der Objekte explizit berücksichtigt wird. Besonders mit den hierarchischen Verfahren konnten gute Resultate erzielt werden. Anzumerken ist, dass in dieser Arbeit sehr einfache Algorithmen verwendet wurden. Mit komplexeren Verfahren lassen sich die Ergebnisse möglicherweise verbessern.

Auf kleinen Bildschirmen kann nur eine beschränkte Menge von Information dargestellt werden. Eine weitere Fragestellung dieser Arbeit war, wie die verwendeten Daten auf diesen Displays kartographisch gut dargestellt werden können. Es wurden drei Möglichkeiten präsentiert, die durch das Clustering ermittelten Gruppen zu visualisieren. Eine Gruppe wird dabei durch eine lokale Signatur oder eine Fläche repräsentiert. Durch die Variation der graphischen Variablen wurde

versucht, weitere Information, beispielsweise die Anzahl Beobachtungen pro Cluster, darzustellen. Die Idee ist, mit möglichst wenig Signaturen möglichst viel Information über die Verteilung der Beobachtungen zu vermitteln. Um zu bestimmen, wie gut sich die drei vorgestellten Möglichkeiten dafür eignen, muss eine Untersuchung mit potentiellen Nutzern und Nutzerinnen durchgeführt werden. Da eine solche im Rahmen dieser Arbeit nicht stattfand, können keine entsprechenden Aussagen gemacht werden.

6.2 Ausblick

Diese Arbeit enthält eine Reihe von Themen, bei denen weiterer Forschungsbedarf besteht. Ein solches Gebiet sind die vorgestellten Visualisierungen: Das Ziel ist es, die Daten so darzustellen, dass eine optimale Informationsvermittlung auch auf kleinen Bildschirmen erreicht wird. Um gute Lösungen zu finden, muss die Aufnahme der Karten durch potenzielle Benutzer und Benutzerinnen untersucht werden: Wie werden die Karten interpretiert? Welche Variante ist am leichtesten verständlich und wird von den Nutzern bevorzugt? Bei dieser Untersuchung könnte nach einem ähnlichen Muster vorgegangen werden, wie es von Brodersen (1986) zur Untersuchung von thematischen Atlaskarten verwendet wurde.

Bei der Generalisierung stellen sich zwei Fragen. Eine erste betrifft die Beschleunigung der Kartenberechnung durch eine vorgängige Generalisierung (vgl. Kap. 5.3). Hier geht es darum, die Verwendbarkeit des Dendrogramms detailliert zu untersuchen. Die zweite Frage dreht sich um das Finden von Randbedingungen (constraints) der Generalisierung. Konkret müssen dazu die dargestellten Tierarten und ihre Lebensräume analysiert werden. Damit wird es einerseits möglich, die in dieser Arbeit gefundenen Cluster auf ihre biologische Plausibilität zu überprüfen, andererseits können bekannte Habitatgrenzen als Randbedingungen bei der Generalisierung berücksichtigt werden, um so qualitativ bessere Gruppeneinteilungen zu erreichen.

Im Bereich der Architektur dürfte neben der Lösung von technischen Problemen vor allem die weitere Untersuchung eines Generalisierungsdienstes von Interesse sein. Eine wichtige Frage ist beispielsweise, wie die Schnittstelle eines solchen Dienstes definiert werden soll.

7 Literatur

AGENT (1999): Selection of Basic Algorithms, Report for the AGENT project. <http://agent.ign.fr/deliverable/DD2.pdf>, 4.2.2004.

APACHE-PROJEKT (2004): Website des XSLT-Prozessors Xalan (Java-Version). <http://xml.apache.org/xalan-j/index.html>, 30.1.2004.

ANDERS K.-H. (2002): Parameterfreies hierarchisches Graph-Clustering Verfahren zur Interpretation raumbezogener Daten, Hannover.

BACKHAUS K. ET AL. (1996): Multivariate Analysemethoden, 8. Auflage, Berlin/Heidelberg.

BAHRENBURG G., GIESE E., NIPPER J. (1992): Statistische Methoden in der Geographie 2, Stuttgart/Leipzig.

BERKHIN P. (2002): Survey Of Clustering Data Mining Techniques. Accrue Research Papers (<http://www.accrue.com/products/researchpapers>, 14.9.2003).

BRASSEL K., WEIBEL R. (1988): A review and conceptual framework of automated map generalization. In: International Journal of Geographical Information Systems, 1988, Vol. 2, No. 3, S.229-244.

BRODERSEN L. (1986): Aspekte der graphischen Gestaltung komplexer Wirtschaftskarten in Schulatlanten, Dissertation, Zürich.

CECCONI A. (2003): Integration of Cartographic Generalization and Multi-Scale Databases for Enhanced Web Mapping, Dissertation, Zürich.

DE BERG M. ET AL. (2004): On simplifying dot maps. In: Computational Geometry: Theory and Applications, 27(1), S. 43-62.

DEGREE (2003): Website des degree-Projekts. <http://degree.sourceforge.net>, 24.7.2003.

DELUCIA A., BLACK R. (1987): A comprehensive Approach to Automatic Feature Generalization. Proceedings of the 13th International Cartographic Conference, Morelia, Mexico, 169-192.

DIESTEL R. (2000): Graphentheorie, 2. Auflage, Heidelberg.

DOUGLAS D., PEUCKER T. (1973): Algorithms for the reduction of points required to represent a digitized line or its caricature. In: Canadian Cartographer, Vol. 30, S. 112-122.

FIBINGER I. (2002): SVG – Scalable Vector Graphics, Praxiswegweiser und Referenz für den neuen Vektorgrafikstandard, München.

GRÜNREICH D. (1992): ATKIS – a topographic information system as a basis for GIS and digital

- cartography in Germany. In: Geologisches Jahrbuch Reihe A, Heft 122, Hannover, S. 207-216.
- GRÜNREICH D. (1985): Computer-assisted generalisation, Papers CERCO Cartography Course, Frankfurt a.M.
- HAKE G., GRÜNREICH D. (1994): Kartographie, 7. Auflage, Berlin/New York.
- HAROLD E., MEANS W. (2002): XML in a Nutshell, 2. Auflage, Sebastopol.
- HARRIE L., SARJAKOSKI T., LEHTO L. (2002): A variable-scale map for small-display cartography. In: Proceedings of the Joint International Symposium on Geospatial Theory, Processing and Application, Ottawa.
- IBM (2004): New to Web services. <http://www-106.ibm.com/developerworks/webservices/newto/>, 29.1.2004.
- JOOS G. (2003): Das Web-Mapping-Testbed des Open-GIS-Konsortiums. In: Asche H., Herrmann C. (Hrsg.) (2003): Web Mapping 2, Heidelberg, S. 181-189.
- KAUFMAN L., ROUSSEEUW P. (1990): Finding groups in data – an introduction to cluster analysis, New York.
- LEHTO L., KILPELÄINEN T. (2001): Generalizing XML-Encoded Spatial Data on the Web. In: Proceedings of The 20th International Cartographic Conference, Beijing, 2001, Vol. 4, S. 2390-2396.
- MCMASTER R., SHEA K. (1992): Generalization in Digital Cartography, Washington.
- MCMASTER R. (1991): Conceptual frameworks for geographical knowledge. In: Battenfield B., McMaster R. (Hrsg.) (1991): MapGeneralisation: making rules for knowledge representation, London, S. 21-39.
- MÜLLER J.-C. ET AL. (1995): Generalization: state of the art and issues. In: Müller J.-C., Lagrange J.-P., Weibel R. (Hrsg.) (1995): GIS and Generalization – Methodology and Practice, London, S. 3-17.
- MULLER J.-C. (1991): Generalization of Spatial databases. In: Maguire D., Goodchild M., Rhind D. (Hrsg.) (1991): Geographic Information Systems, Volume 1: Principles, Harlow/New York, S. 457-475.
- NEUMANN A., WINTER A. (2000): Kartografie im Internet auf Vektorbasis, mit Hilfe von SVG nun möglich. <http://www.carto.net/papers/svg>, 22.2.2003.
- NEUMANN J. (1997): Enzyklopädisches Wörterbuch Kartographie in 25 Sprachen, 2. erweiterte Ausgabe, München.

- OPEN GIS CONSORTIUM OGC (2003): OpenGis® Geography Markup Language (GML) Implementation Specification (02-023r4). <http://www.opengis.org/techno/documents/02-023r4.pdf>, 22.2.2003.
- OPEN GIS CONSORTIUM OGC (2002a): Web Feature Service Implementation Specification (02-058). <http://www.opengis.org/techno/specs/02-058.pdf>, 22.2.2003.
- OPEN GIS CONSORTIUM OGC (2002b): The OpenGIS Abstract Specification. Topic 12: OpenGIS Service Architecture, Version 4.3 (02-112). <http://www.opengis.org/docs/02-112.pdf>, 25.1.2004.
- OPEN GIS CONSORTIUM OGC (1999): The OpenGIS Abstract Specification. Topic 0: Abstract Specification Overview, Version 4 (99-100r1-1). <http://www.opengis.org/docs/99-100r1.pdf>, 25.1.2004.
- O'SULLIVAN D., UNWIN D. (2003): Geographic Information Analysis, Hoboken.
- REGNAULD N. (1996): Recognition of Building Clusters for Generalization. In: Proceedings of the 7th International Symposium on Spatial Data Handling, Delft, 1996, Vol. 1, S. 4B.1-14.
- PETERSON M. (2003): Webmapping at the start of the new Millenium – state of the art. In: Asche H., Herrmann C. (Hrsg.) (2003): Web Mapping 2, Heidelberg, S. 3-17.
- REICHENBACHER T. (2002): Adaptive Visualisierung von Geodaten für Location Based Services – ein konzeptionelles Framework. In: Geowissenschaftliche Mitteilungen, Heft Nr. 58, S. 125-134, Wien.
- REICHENBACHER T., MENG L. (2003): Forschungsfragen der mobilen Kartographie. In: Kartographische Nachrichten, Heft 2, 53. Jg., S. 67-70, Bonn.
- RIEGER M., COULSON M. (1993): Consensus or Confusion: Cartographers' knowledge of generalization. In: Cartographica, Vol. 30 (2+3), S. 69-80.
- RUAS A. (2002): Les problématiques de l'automatisation de la généralisation. In: Ruas A. (Hrsg.) (2002): Généralisation et représentation multiple, Paris, S. 75-90.
- SALVADOR S., CHAN P. (2003): Determining the Number of Clusters/Segments in Hierarchical Clustering/Segmentation Algorithms. <http://www.cs.fit.edu/~tr/cs-2003-18.pdf>, 2.10.2003.
- SONDHEIM M., GARDELS K., BUEHLER K. (1999): GIS interoperability. In: Longley P.A. et al. (Hrsg.) (1999): Geographic Information Systems, Second Edition, Volume 1 – Principles and Technical Issues, S. 347-358.
- TÖPFER F. (1974): Kartographische Generalisierung, Gotha/Leipzig.

-
- VAN OOSTEROM P. (1995): The GAP-tree, an approach to 'on-the-fly' map generalization of an area partitioning. In: Müller J.-C., Lagrange J.-P., Weibel R. (Hrsg.) (1995): GIS and Generalization – Methodology and Practice, London, S. 120-132.
- VCKOVSKI A. (1998): Interoperable and Distributed Processing in GIS, Dissertation, Zürich.
- W3C (2003): Scalable Vector Graphics (SVG) 1.1 Specification. <http://www.w3.org/TR/SVG11>, 25.7.2003.
- W3C (1999): XSL Transformations (XSLT) Version 1.0. <http://www.w3.org/TR/xslt>, 25.7.2003.
- WEBPARK (2003): Website des Webpark-Projekts. <http://www.webparkservices.info>, 13.10.2003.
- WEIBEL R. ET AL. (2002): La généralisation à la volée. In: Ruas A. (Hrsg.) (2002): Généralisation et représentation multiple, Paris, S. 319-335.
- WEIBEL R., DUTTON G. (1999): Generalising spatial data and dealing with multiple representations. In: Longley P.A. et al. (Hrsg.) (1999): Geographic Information Systems, Second Edition, Volume 1 – Principles and Technical Issues, S. 125-156.
- WEIBEL R. (1997): Generalization of Spatial Data: Principles and Selected Algorithms. In: Van Kreveld M. et al. (Hrsg.) (1997): Algorithmic Foundations of Geographic Information Systems, S. 99-152.
- ZIPF A. (2003): Forschungsfragen zur benutzer- und kontextangepassten Kartengenerierung für mobile Systeme. In: Kartographische Nachrichten, Heft 1, 53. Jg., S. 6-11, Bonn.

8 Anhang

In diesem Anhang werden Beispiele für Dateien in GML, XSLT und SVG gezeigt. Diese kommentierten Beispiele sollen den genauen technischen Ablauf verständlicher machen. Alle für diese Arbeit entwickelten Dateien finden sich auf der CD, die dieser Arbeit beiliegt.

GML

Bei der im Folgenden abgebildeten Datei handelt es sich um einen Datensatz in Geography Markup Language, wie sie in einer Antwort des Web Feature Servers geliefert wird.

```
<?xml version="1.0" encoding="UTF-8"?>
<collection xmlns:gml="http://www.opengis.net/gml" >
  <gml:boundedBy><gml:box><gml:coord><gml:X>808515.062</gml:X>
  <gml:Y>171109.969</gml:Y></gml:coord>
  <gml:coord><gml:X>816777.75</gml:X>
  <gml:Y>175471.25</gml:Y></gml:coord></gml:box></gml:boundedBy>
```

Nach der XML-Deklaration mit der Angabe der Codierung, wird der GML-Namensraum definiert. Anschliessend wird die räumliche Ausdehnung des Datensatzes angegeben, bevor die Auflistung der einzelnen Objekte beginnt.

```
<gml:featureMember>
```

Jedes Objekt ist in einem gml:featureMember-Tag eingeschlossen und besitzt eine eindeutige Identifikationsnummer.

```
<Shape gid="0">
  <ART><![CDATA[Steinbock]]></ART>
  <YEAR><![CDATA[1997.0]]></YEAR>
  <MONTH><![CDATA[5.0]]></MONTH>
```

In CDATA-Abschnitten werden die Attribute der Objekte angegeben: Information ist in solchen Abschnitten als Text abgelegt, der nicht ausgezeichnet ist. Als nächstes wird die Geometrie des Objekts beschrieben. Im Falle der verwendeten Tierbeobachtungsdaten ist dies einfach, da es sich nur um Punkte handelt.

```
<gml:pointProperty>
  <gml:Point srsName="null">
    <gml:coordinates cs="," decimal="." ts="">
      808786.438,174982.281
    </gml:coordinates>
  </gml:Point>
</gml:pointProperty>
</Shape>
```

```
</gml:featureMember>
```

Anschliessend an die Beschreibung der Geometrie wird das `gml:featureMember`-Tag geschlossen und das nächste Objekt kann aufgelistet werden.

```
<gml:featureMember>
  <Shape gid="1">
    <ART><![CDATA[Steinbock]]></ART>
    <YEAR><![CDATA[1997.0]]></YEAR>
    <MONTH><![CDATA[5.0]]></MONTH>
    <gml:pointProperty>
      <gml:Point srsName="null">
        <gml:coordinates cs="," decimal="." ts="">
          808771.062,175334.062
        </gml:coordinates>
      </gml:Point>
    </gml:pointProperty>
  </Shape>
</gml:featureMember>
</collection>
```

XSLT

Das folgende XSLT-Stylesheet führt auf den Daten in einer GML-Datei ein hierarchisches Clustering durch und schreibt die Resultate in eine SVG-Datei. Die graphische Darstellung der Cluster erfolgt durch Standardabweichungsellipsen.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- stylesheet to cluster point data with hierarchical algorithm (single
linkage)-->
<!-- draws standard deviation ellipse to represent clusters-->
<!-- written by Juerg Mannes -->
```

Im Anschluss an einige einleitende Kommentare werden die Namensräume des Stylesheets definiert, wobei auch die Sprachen des Ausgangs- und des Resultatdokuments berücksichtigt werden müssen. Zusätzlich wird ein Namensraum für Java definiert, damit der (Xalan-)XSLT-Prozessor die Aufrufe der Java-Methoden erkennt.

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:gml="http://www.opengis.net/gml" xmlns:svg="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:java="http://xml.apache.org/xslt/java">
```

Im `xsl:output`-Tag wird der Dateityp und die Codierung der Resultatdatei festgelegt.

```
<xsl:output method = "xml" indent="yes" encoding="iso-8859-1" media-
type="image/svg+xml"/>
```

Mittels `xsl:param` lassen sich Parameter definieren, deren Werte einem Stylesheet übergeben werden, wenn es aufgerufen wird. Hier handelt es sich um globale Parameter für die Grösse der Karte, die gezeichnet wird. `x` und `y` sind die Koordinaten der Ecke links oben, `l` und `b` die Länge respektive die Breite der Karte. Diese Angaben beziehen sich auf die Kartenkoordinaten, während `width` und `height` die Breite und Höhe des Displays des Ausgabegeräts in Pixel repräsentieren.

```
<xsl:param name="x"/>
<xsl:param name="y"/>
<xsl:param name="b"/>
<xsl:param name="l"/>
<xsl:param name="width"/>
<xsl:param name="height"/>
```

Weiter wird eine globale Variable definiert, die als „Instanzvariable“ für die Aufrufe von Java-Methoden dient. Mit `name` wird der Name der Variable festgelegt, der mit `select` ein Wert zugewiesen wird. Diese Variable enthält eine Referenz auf eine Instanz der Java-Klasse `PointGroup`. Es ist auch ersichtlich, wie der Aufruf von Java-Methoden aus XSLT erfolgt, wenn als XSLT-Prozessor Xalan verwendet wird: Der Namensraum `java:` teilt dem Prozessor mit, dass nun der Aufruf einer Java-Methode folgt. Die Methode `new()` ruft den Konstruktor der angegebenen Klasse auf.

```
<!-- globale "Instanzvariable" fuer eine PointGroup -->
<xsl:variable name="pointgroup" select="java:PointGroup.new()"/>
```

Eine weitere globale Variable enthält einen Wert für die Symbolgrösse in der Karte. Dieser Wert wird in einem Template, auf deutsch Schablone, berechnet, das den Namen `sSize` trägt und mittels `call-template` aufgerufen wird. Templates können als Methoden betrachtet werden: Es ist möglich, diese Methoden mit einem Muster zu verbinden, das auf Knoten im Ausgangsdokument zeigt, und dann im Template Regeln zu definieren, um die Daten dieses Knotens umzuwandeln.

```
<!-- berechnung einer geeigneten Symbolgroesse für diesen Massstab -->
<xsl:variable name="syms">
  <xsl:call-template name="sSize"/>
</xsl:variable>
```

Als nächstes folgt ein erstes Template, das mit einem Muster oder Pattern verbunden ist. Im Attribut `match` des `xsl:template`-Tags kann mittels XPath ein Knoten angegeben werden, der dann im Ausgangsdokument gesucht wird. Für jeden dort gefundenen Knoten, werden die im Template


```
<svg xmlns:java="http://xml.apache.org/xslt/java"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:svg="http://www.w3.org/2000/svg"
xmlns:gml="http://www.opengis.net/gml" xml:space="preserve"
width="600px" height="400px" viewBox="808000.0 -176000.0 8000.0 6000.0"
preserveAspectRatio="xMidyMid">
```

Als Basiskarte kann, wie hier, ein Rasterbild verwendet werden. Es ist auch möglich, Vektordaten zur als Basiskarte zu verwenden.

```
<image x="798125" y="-180600" width="21740" height="18500"
xlink:href="file:///D:/Data/Studium/diplomarbeit/snpdata/first/snp100.jpg"/>
```

Die Ellipsen, die zur Darstellung der Cluster verwendet werden, müssen geometrischen Transformationen unterzogen werden.

```
<ellipse cx="0" cy="0" rx="427.26623720120944" ry="139.83933647541247"
fill="blue" opacity="0.5" transform="matrix(1 0 0 -1 0 0) translate
(808989.3251333332,175024.90846666667) rotate(148.71568739646654)"/>
<ellipse cx="0" cy="0" rx="533.3041563292556" ry="130.31407400392928"
fill="blue" opacity="0.5" transform="matrix(1 0 0 -1 0 0) translate
(811044.0793636363,174341.64763636363) rotate(42.06268448482276)"/>
<ellipse cx="0" cy="0" rx="474.17046123562733" ry="168.8527892731023"
fill="blue" opacity="0.5" transform="matrix(1 0 0 -1 0 0) translate
(812600.929375,173032.642625) rotate(31.367666664290162)"/>
<ellipse cx="0" cy="0" rx="117.66759520239131" ry="235.2355716202672"
fill="blue" opacity="0.5" transform="matrix(1 0 0 -1 0 0) translate
(815147.677,173347.04433333332) rotate(13.596594648482778)"/>
<ellipse cx="0" cy="0" rx="357.0948337579275" ry="226.68154635686227"
fill="blue" opacity="0.5" transform="matrix(1 0 0 -1 0 0) translate
(816324.3611666667,171506.39227777778) rotate(142.02121592958403)"/>
<ellipse cx="0" cy="0" rx="0.0" ry="0.0" fill="blue" opacity="0.5"
transform="matrix(1 0 0 -1 0 0) translate(812954.438,173990.219) rotate
(NaN)"/>
<ellipse cx="0" cy="0" rx="523.4139002016489" ry="53.42892038574367"
fill="blue" opacity="0.5" transform="matrix(1 0 0 -1 0 0) translate
(814983.3956666667,174304.844) rotate(172.29851543234878)"/>
</svg>
```