Graphical Abstraction and Progressive Transmission in Internet-based 3D-Geoinformationsystems

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

DISSERTATION

zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
von

Dipl.-Inform. Volker Coors
aus Walsrode

Referenten der Arbeit:	Prof. Dr. J.L. Encarnação
	Prof. J. Rossignac, PhD

Tag der Einreichung:	17.12.2002
Tag der mündlichen Prüfung:	29.01.2003

D17
Darmstädter Dissertationen 2003

Acknowledgements

Many people have helped in one way or another to make this dissertation happen. I would like to thank everybody who supported me.

My special thanks go to my advisor José Encarnação, for giving me the freedom to explore this interesting subject while helping me to stay on the right track. I would like to express my deepest gratitude to Jarek Rossignac for accepting to be on my thesis committee. I owe him a lot. His invitation to Atlanta has been a remarkable experience.

Many of the people I have been working with have helped me. In particular, “big boss” Uwe Jasnoch, and my “fellow sufferers” Dirk Balfanz and Stefan Göbel have been a source for countless valuable discussions. I would also like to thank my former colleagues Heiko Blechschmied, Dirk Burmeister, Sascha Flick, Elfriede Fitschen, Christine Giger, Jörg Haist, Monika Heidemann, Volker Jung, Ursula Kretschmer, Daniel Holweg, Karen Lutze, Isabel Sobon, and Thorsten Schulz. Their friendship and encouragement made previous years enjoyable and tough times more bearable.

I am deeply indepted to my parents. Where I am today is no small part due to their encouragement and support. Thousand kisses to Susy for support, love, and for her understanding that this work took much of my time. And special thanks to Florian, simply for being so cute, and baby Tabea for not crying all night long while daddy is working on this thesis.

Deutsche Kurzfassung

Motivation

Mit Hilfe eines Geo-Informationssystems (GIS) werden Informationen über die Erde erfaßt, verwaltet, analysiert und auf unterschiedlichste Art und Weise präsentiert. Eine Vielfalt an Modellen spiegelt dabei unser Wissen über die Erdoberfläche und deren Bebauung sowie über die auf ihr lebenden Menschen, Pflanzen und Tiere wieder. Kommerzielle GIS unterstützen in der Regel digitale Höhenmodelle (DHM) der Erdoberfläche, während sie die Speicherung der räumlichen Ausdehnung von Objekten auf der Erde nur zweidimensional zulassen.

Durch die rasch voranschreitende Hardwareentwicklung, vor allem aber durch Fortschritte in der (semi-)automatischen Datenerfassung ist es heute jedoch möglich geworden, raumbezogene Objekte in ihrer dreidimensionalen Ausdehnung zu erfassen. Vorangetrieben wurden diese Entwicklungen durch ein breites Anwendungsspektrum dreidimensionaler raumbezogener Informationen u.a. in der Archäologie, der Telekommunikation und des Umweltschutzes, aber auch in Tourismus und Edutainment. Mit Hilfe dreidimensionaler Modelle können Umweltsituationen wie die Abgas- und Lärmausbreitung in Städten simuliert und für den Bürger verständlich präsentiert werden [Knol98]. In der Stadtplanung wird der eigentliche Planungsprozeß durch die Verfügbarkeit dreidimensionaler Computermodelle stark unterstützt. Alternativen können besser beurteilt werden, und auch Laien können die Folgen der vorgeschlagenen Änderungen nachvollziehen [Aria94], [Coor99]. Das kulturelle Erbe einer Stadt wird für Touristen erfahrbar [Coor00b].

Auch in der Telekommunikation wächst der Bedarf an Modellen, die zusätzlich zu der Erdoberfläche auch urbane Regionen dreidimensional abbilden. In der Tat sind in Deutschland derzeit die Mobilfunknetzbetreiber die treibende Kraft bei der Erfassung flächendeckender dreidimensionaler Stadtmodelle in Ballungsgebieten. Vorrangiges Ziel der Mobilfunktnetzbetreiber ist ein möglichst hoher Netzabdeckungsbereich. Zur Planung dieses Netzes werden dreidimensionale Stadtmodelle eingesetzt, um Bebauungseffekte wie Funkschatten und Reflexionen elektromagnetischer Wellen durch Gebäude berücksichtigen zu können [RaLa00].

Die Nachfrage nach entsprechenden dreidimensionalen Modellen wird durch eine OEEPE Studie zu 3D-Stadtmodellen der Europäischen Organisation für Experimentelle Photogrammetrische Forschung [Fuch98] bestätigt: 95% der Teilnehmer sehen den größten Bedarf an dreidimensionalen Gebäudedaten in einem digitalen Stadtmodell, gefolgt von Informationen über das Verkehrsnetz (etwa 85%) und 3D-Informationen über die Vegetation (etwa 75%).

Wie aus den oben genannten Beispielen deutlich wird, sind neben einem digitalen Geländemodell der Erdoberfläche insbesondere dreidimensionale Modelle urbaner Regionen und Ballungszentren von großer Wichtigkeit. Die Erfassung dieser Daten ist heutzutage durch photogrammetrische Verfahren möglich [Förs99], [Maye99], [BrHa00]. Es fehlen jedoch geeignete Werkzeuge, um diesen Datenbestand geeignet zu verwalten und einer großen Zahl von Nutzern zur Verfügung zu stellen.

Neue Chancen, dem Nutzer qualitativ hochwertige Daten einfach zugänglich zu machen, sind durch moderne Kommunikationsmedien gegeben, insbesondere durch das Internet, welches eine sehr einfache Verbreitung von Daten erlaubt. Aber nicht nur die Verfügbarkeit der Informationen wird durch eine entsprechende Infrastruktur wesentlich erhöht, auch neue Formen der Kommunikation werden möglich. So können beispielweise Planungsvorhaben in der Stadt vorab verständlich dargestellt und diskutiert werden. Dem steigenden Informationsbedarf bei Entscheidungsträgern und der Bevölkerung wird somit Rechnung getragen. Auch die rasante Entwicklung im Bereich von persönlichen digitalen Assistenten (PDA) bringt immer kleinere und leistungsfähigere Geräte hervor. Die Nutzung von dreidimensionalen Modellen auf diesen ultraportablen Geräten beispielsweise in Systemen zur Personennavigation rückt in greifbare Nähe. Durch die steigende Datenrate der mobilen Kommunikationsnetze sind digitale Informationen allgegenwärtig und können an jedem Ort zu jeder Zeit abgerufen werden. Die Kombination raumbezogener Informationssysteme und moderner Datennetze wird in Zukunft nicht mehr aus unserem Alltag wegzudenken sein.

3D-GIS als Forschungsgebiet

Eine verbreitete Vorgehensweise im Umgang mit dreidimensionalen raumbezogenen Daten ist die kombinierte Anwendung von 2D-GIS, CAD- und Visualisierungssystemen. CAD-Systeme aus der Architektur sind spezialisiert auf Planung und Konstruktion dreidimensionaler Gebäude. Im Gegensatz zu GIS wird in CAD-Systemen aber in einem sehr viel kleinerem Maßstab gearbeitet, üblicherweise 1:1 bis 1:50. Im Umgang mit großmaßstäbigen flächendeckenden Modellen weisen diese Systeme jedoch Schwächen auf. Visualisierungssysteme dienen, wie der Name vermuten läßt, in erster Linie der Präsentation eines Datenbestandes. Visualisierungssysteme, die mit dem hohen Datenvolumen aus dem GIS Bereich geeignet umgehen können, haben ihre Ursprünge in der virtuellen Realität (VR). Ziel dieser Systeme ist die interaktive Präsentation einer möglichst realen Umgebung, wobei Stereodarstellung und spezielle Präsentationsumgebungen den Grad der Immersion noch erhöhen. Solche VR-Systeme haben oft hohe Anforderungen an die zugrundeliegende Hardwareausstattung und arbeiten mit einem lokalen Datenbestand. Eine Verknüpfung der Präsentation mit thematischen Daten wird nicht unterstützt.

Im Kontext eines 3D-GIS spielt jedoch die Integration eines dreidimensionalen geometrischen Datenbestandes und thematischer Fachdaten eine wichtige Rolle. Erst durch diese Verknüpfung lassen sich kombinierte raumbezogene und thematische Suchanfragen beantworten. Beispielhaft seien hier die Suche nach Hotels in der Nähe des Bahnhofs die Suche nach verfügbaren Immobilien mit Kindergarten in der Umgebung genannt. Entsprechende raumbezogene Anfragen können mit Hilfe eines dreidiemnsionalen Stadtmodells bearbeitet werden. Ob es sich bei einem Gebäude jedoch um ein Hotel oder einen Kindergarten handelt, kann der geometrischen Beschreibung in der Regel nicht entnommen werden. Hier ist die Integration des dreidimensionalen Modells mit in der Regel vorhandenen fachbezogenen Daten notwendig.

Damit die einmal erhobenen Daten einem möglichst großen Nutzerkreis verfügbar gemacht werden können, ist der Zugriff des Datenbestandes über Kommunikationsnetze wie das World Wide Web oder auch mobile Netze von großer Bedeutung. Aus diesen Anforderungen ergeben sich die folgenden Kernthemen im Kontext 3D-GIS:

Konzeptionelles Datenmodell

Ein wesentliches Manko bestehender CAD- und VR-Systeme ist die mangelnde Unterstützung typischer funktionaler Anforderungen an geographische Informationssysteme wie kombinierte Suche nach raumbezogenen und semantischen Kriterien oder Analyse von Nachbarschaftebeziehungen. Zur effizienten Bearbeitung dieser Anfragen muss einem 3D-GIS ein geeignetes konzeptionelles Datenmodell zugrundeliegen. Ein solches Datenmodell muss Informationen über die dreidimensionale Geometrie und Semantik raumbezogener Objekte, aber auch die topologische Beziehungen der Objekte untereinander abbilden. Neben der Semantik spielt gerade diese 3D-Topologie bei der Analyse des Datenbestandes eine entscheidende Rolle. Analysefunktionalität und konzeptionelles Datenmodell stehen dabei in direktem Zusammenhang. Zusätzlich zur reinen Geometrie ist es erforderlich, Darstellungsattribute zu verwalten, um aus dem Datenbestand kontextabhängig Präsentationen generieren zu können. Gleichzeitig muss aber aufgrund der Vielzahl der zu verwaltenden Objekte das Datenvolumen eines Einzelobjekts so gering wie möglich gehalten werden.

In verschiedenen Arbeiten werden konzeptionelle Modelle für 3D-GIS vorgeschlagen, wobei der Schwerpunkt in der Regel bei der Abbildung der 3D-Topologie liegt (vgl. [Mole92], [Pilo96], [Flic98], Zlat[00]). Diese Modelle decken einen Teilaspekt der Anforderungen sehr gut ab, unterstützen andere Aspekte jedoch nur rudimentär. Eine Diskussion der wichtigsten Ansätze findet sich in Abschnitt 3.3.

Nutzung des Datenbestandes in heterogener Umgebung

Die Wirtschaftlichkeit eines 3D-GIS wird sehr davon abhängen, ob eine Nutzung der Daten durch einen großen Anwenderkreis möglich ist. Dazu ist es notwendig, die Daten so zu speichern und zu verwalten, dass eine große Anzahl von Nutzern über ein gemeinsames Netzwerk auf diesen Datenbestand zugreifen kann. Durch dieses Konzept eines Geodatenservers, der neben dem Datenbestand auch Anfrage- und Analysefunktionalität in einem Netzwerk zur Verfügung stellt, ergeben sich folgende Vorteile:

- Kostenersparnis bei der Datenerfassung, -verwaltung, -speicherung und beim Datenaustausch
- Kostenersparnis für Anwender durch hohe Aktualität des Datenbestands und Vermeidung der zeitlich aufwendigen Konvertierung und Integration verschiedener Datenbestände
- Höchstmögliche Verfügbarkeit der Daten zu jeder Zeit und - durch voranschreitende Entwicklung der mobilen Netze - an jedem Ort

Für eine effiziente Nutzung eines 3D-Geodatenservers sind aufgrund des zu erwartenden hohen Datenvolumens neue Konzepte zur Datenübertragung zu entwickeln. Die Ergebnisse einer Anfrage sollen dem Benutzer in einer geeigneten Form präsentiert werden. Die Ergebnismenge typischer Anfragen wie einer Region-Query, bei denen nach Objekten innerhalb einer gegebenen Region gesucht wird, beinhaltet eine große Anzahl raumbezogener Daten. Hierbei stellt sich die Frage nach einer geeigneten Aufbereitung dieses Anfrageergebnisses, das dem Benutzer visuell vermittelt werden soll. Neben dem Informationsziel des Benutzers sind hier auch die technischen Möglichkeiten des Ausgabegerätes und technische Aspekte der Datenübertragung zu berücksichtigen. Verfahren zur ressourcen-adaptiven Visualisierung sind im Kontext eines 3D-Geodatenservers von entscheidender Bedeutung, da der Zugriff auf den Datenbestand mit sehr unterschiedlichen Endgeräten erfolgen kann. Um eine größtmögliche Nutzung der Daten sicherzustellen, müssen sowohl leistungsstarke Visualisierungssysteme aus dem Bereich der virtuellen Realität als auch ultraportable Endgeräte wie PDAs unterstützt werden. In direkten Zusammenhang damit steht auch eine sehr unterschiedliche Bandbreite der genutzten Kommunikationsnetze.

Bisherige Forschungsarbeiten im 3D-GIS Kontext gehen über ein Level-of-Detail Konzept zur die Visualisierung des Datenbestandes nicht hinaus [Kofl98], [Zlat00]. Dieses Konzept allein ist jedoch nicht ausreichend, um die oben skizzierte Problematik der Ergebnisvisualisierung innerhalb eines heterogenen Netzwerks zu lösen.

Ziele der vorliegenden Arbeit

Zusammenfassend liegt das Ziel dieser Forschungsarbeit darin, einen Mangel bisheriger 3D-GIS zu beseitigen: die Generierung einer aussagekräftigen interaktiven dreidimensionalen Präsentation eines ausgewählten Datenbestandes unter spezieller Berücksichtigung der graphischen Abstraktion und progressiver Übertragung in einer heterogenen Netzwerkumgebung. Dabei soll das Konzept einer offenen verteilten GIS-Infrastruktur des OpenGIS Consortiums (OGC) [BüMc96] um dreidimensionale Geodaten erweitert werden. Hierzu soll ein 3D-Geodatenserver konzipiert und entwickelt werden. Kernbestandteile dieses 3D-Geodatenservers sind

- ein topologisches Datenmodell zur Speicherung dreidimensionaler Geometrie,

- Methoden zur graphischen Abstraktion von 3D-Stadtmodellen
- progressive Übertragung und Kompression von 3D-Stadtmodellen

Im Gegensatz zu vorangegangenen Arbeiten wie [Lang99] liegt der Schwerpunkt nicht auf einer möglichst photorealisistischen Darstellung einer virtuellen Landschaft bzw. urbanen Region. Die visuelle Aufbereitung eines Datenbestandes soll vielmehr das Informationsziel des Benutzers möglichst gut unterstützen. Bei dieser Form der Visualisierung geht es im Gegensatz zum Rendering in der Computergraphik nicht darum, ein zugrundeliegendes geometrisches Modell perspektivisch korrekt auf eine zweidimensionale Fläche abzubilden. Vielmehr dient die Visualisierung dem Ziel, dem Betrachter Informationen zu vermitteln [Fole92]. Bestandteil eines solchen Prozesses ist nicht nur das eigentliche Rendering, sondern auch ein Selektionsprozess, der entscheidet, welche Bestandteile der Szene zum Informationsgehalt beitragen. Diese Art der Visualisierung wird graphische Abstraktion genannt.

Der Prozess der graphischen Abstraktion ist in GIS als Generalisierung bekannt und hat eine traditionelle Anwendung in der Kartographie bei der Überführung einer Karte in einen anderen Maßstab. Bei der Kartenherstellung wird eine Generalisierung immer für eine große Zielgruppe angefertigt, die entsprechende Karten benötigt. Ein typisches Beispiel sind Straßenkarten für Autofahrer. Autobahnen spielen für diese Zielgruppe eine große Rolle. Im Kartenwerk sind sie entsprechend stark vergrößert dargestellt, während andere Daten wie Wald- und Fahrradwege völlig fehlen. Die Verfügbarkeit digitaler Geodaten beispielsweise im Internet erfordert jedoch eine zunehmende Individualisierung und Automatisierung des Generaliserungsprozesses. Eine Karte wird nicht mehr für eine Nutzergruppe erzeugt, sondern individuell für den Benutzer, der gerade einen Informationswunsch geäußert hat. Die Generalisierung muss entsprechend den individuellen Zielen in sehr kurzer Zeit erfolgen.

In dieser Arbeit werden Techniken zur graphischen Abstraktion in 3D-GIS entwickelt. Diese Techniken sollen durch ein konzeptionelles Datenmodell unterstützt werden, um den Informationsraum bestmöglich zu repräsentieren. Ein entsprechendes Modell für ein 3D-GIS wird als Grundlage für die Bearbeitung von Anfragen und die graphische Abstraktion und Übertragung des Anfrageergebnisses entwickelt.

In einer heterogenen Netzwerkumgebung kommt neben dieser graphischen Abstraktion noch die Problematik der ressourcen-adaptiven Datenübertragung und Visualisierung hinzu. Es stellt sich dann nicht nur die Frage nach einer geeigneten Darstellung eines ausgewählten Datenbestandes, sondern auch nach einer adäquaten Übertragung der Daten, die dem Benutzer in möglichst kurzer Zeit ein erstes visuelles Ergebnis präsentiert. Zusätzliche Daten können nachträglich übertragen werden. Diese progressive Übertragung der Daten ist für die Nutzung von 3D-Informationen im Internet entscheidend [Ross98].

Ein Verfahren zur Datenübertragung innerhalb eines 3D-GIS bildet neben der graphischen Abstraktion den Schwerpunkt dieser Arbeit. Dabei sind graphische Abstraktion und Datenübertragung nicht unabhängig voneinander zu betrachten. Eine graphische Abstraktion führt im Allgemeinen auch zu einer Reduktion des darzustellenden Datenvolumens. Eine Abstraktion kann daher auch eingesetzt werden, um ein 3D-GIS innerhalb eines Netzwerkes mit geringer Bandbreite zu nutzen.

Zusammenfassung der Ergebnisse

Vom wissenschaftlichen Standpunkt sind die wichtigsten Ergebnisse dieser Forschungsarbeit:

- die Entwicklung eines Query-Datenmodells zur effizienten Analyse topologischer Beziehungen dreidimensionaler raumbezogener Daten,
- eine Methode zur Bewertung der Relevanz einzelner Features bezogen auf eine benutzerspezifische Anfrage und darauf aufbauend die Generierung einer graphischen Abstraktion des Anfrageergebnisses,
- die Entwicklung eines Kompressionsverfahrens für Dreiecksnetze zur progressiven Übertragung dreidimensionaler Modelle in einer Netzwerkumgebung
- und die prototypische Realisierung dieser Konzepte innerhalb eines 3D-Geodatenservers

Query-Datenmodell

Im Rahmen dieser Arbeit wurde das Urban Data Model (UDM) als Query-Datenmodell zur Verwaltung und Analyse urbaner Daten entwickelt (Kapitel 5.1). In diesem Datenmodell werden diskrete raumbezogene Weltobjekte als Erweiterung des OpenGIS-Datenmodells als Features modelliert. Es bildet die Grundlage zur datenbankgestützten Verwaltung und Analyse dreidimensionaler Stadtmodelle. UDM zeichnet sich insbesondere durch die folgenden vier Eigenschaften aus:

- *Feature-Geometrie:* Repräsentation der räumlichen Ausdehnung elementarer Feature durch genau ein *n*-dimensionales geometrisches Primitiv. Ein *Point* repräsentiert dabei ein 0-dimensionales Objekt, 1-, 2- und 3-dimensionale Objekte werden durch *Line*-, *Surface*- bzw. *Body*-Primitiv realisiert.
- *Konstruktionselemente:* Modellierung der Primitive durch die Konstruktionselemente *Node*, *Arc*, *Face* und *Solid*. Durch die Beschränkung von der *Face*-Geometrie auf konvexe Polygone kann auf die explizite Speicherung von Kanten verzichtet werden. Dadurch wird das Datenvolumen etwa halbiert, ohne topologische Beziehungen zu verlieren.

- *3D-Topologie*: Explizite Speicherung dreidimensinaler topologischer Relationen im Datenmodell und Realisierung sämtlicher topologischer Operationen nach dem Egenhofer'schen *9-Intersection* Modell. Die topologischen Operatoren können aufgrund des zugrundeliegenden Datenmodells ohne aufwendige geometrische Verschneidungen verwirklicht werden.

- *View-Konzept:* Trennung von Objektgeometrie und Darstellungsgeometrie und damit verbunden ein flexibles Konzept zur dreidimensionalen Visualisierung der Features.

Graphische Abstraktion

Ein grundsätzliches Problem vieler Anfragen innerhalb eines 3D-Geodatenservers ist das große Datenvolumen des resultierenden Anfrageergebnisses. In der vorliegenden Arbeit wurde ein Konzept zur graphischen Abstraktion erarbeitet (Kapitel 5.2), das auf einer räumlichen Zugriffsstruktur basiert. Innerhalb dieses Konzeptes wird die Semantik der Features berücksichtigt, um die zu visualisierende Datenmenge zu reduzieren, ohne dabei die wesentlichen Informationen in der Darstellung zu verlieren. Dazu wird bezogen auf eine konkrete Benutzeranfrage eine Dominanz der einzelnen Feature bewertet. Diese Dominanz spiegelt die Wichtigkeit des Features bei der Kommunikation des Anfrageergebnisses wieder. Einflußfaktoren bei der Berechnung dieses Dominanzwertes sind

- die benutzerspezifisch gewichtete Distanz von Anfrageparametern und den korrespondierenden Merkmalen des jeweiligen Features,

- die Bedeutung des Features als Referenzobjekt in der Anfrage und

- die allgemeine Bedeutung des Features als Landmarke für verschiedene Benutzergruppen.

Ausgehend von diesen Feature-spezifischen Dominanzwerten wird die graphische Abstraktion durchgeführt. In Abhängigkeit des Dominanzwertes wird jedem Feature ein Abstraktionsgrad zugeordnet. Eine Entscheidungsfunktion wählt daraufhin eine geeignete Darstellungsgeometrie für jedes Feature aus. Dabei werden zusätzlich technische und kognitive Ressourcen wie Größe und Auflösung des Displays bzw. Fokus berücksichtigt. Zur Aggregation von Hintergrundobjekten wurde in dieser Arbeit der *P-Tree* konzipiert. Dabei handelt es sich um eine Erweiterung einer raumbezogenen baumartigen Zugriffsstruktur, in deren inneren Knoten zusätzliche Darstellungsgeometrien definiert werden können. Diese sog. Views präsentieren jeweils die Gesamtheit der Darstellungsgeometrien des darunterliegenden Teilbaums und entsprechen somit einem hierarchischen Level-of-Detail. Zur automatisierten Generierung dieser inneren Views können beispielsweise Geometrie-Verfeinfachungsverfahren aus der Computergraphik eingesetzt werden.

Progressive Datenübertragung

Der mittels graphischer Abstraktion generierte P-Tree wird zur progressiven Übertragung des dreidimensionalen Modells in einem heterogenen Netzwerk herangezogen. Mit Hilfe des in dieser Arbeit entwickelten Delphi-Kompression für Dreiecksnetze läßt sich das zu übertragende Datenvolumen einer Darstellungsgeometrie um etwa 95% reduzieren. Das Verfahren basiert auf dem von Rossignac entwickelten Edgebreaker Algorithmus, verwendet aber zur Kompression der Connectivity eines Dreiecksnetzes zusätzliche geometrische Information. Dadurch ist die Delphi-Kompression um bis zu 3 mal so kompakt wie bei Edgebreaker. Die obere Schranke von 4 Bit pro Vertex bleibt auch bei der Delphi-Kompression erhalten. Diese Schranke ist zur Abschätzung des zu erwartenden Datenvolumens und damit zur Entscheidung, ob die zur Verfügung stehende Bandbreite eine (online-) Kompression rechtfertigt von großer Bedeutung.

CityServer3D

Mit dem Softwaresystem CityServer3D wurde die in dieser Arbeit entwickelten Konzept innerhalb eines 3D-Geodatenserver prototypisch realisiert. Den Kern dieses Systems bildet das Query-Datenmodell UDM, welches in unterschiedlichen Datenbanksystemen (Oracle8i, Oracle 9i und Cloudscape) implementiert wurde. Der Zugriff auf die entsprechende Datenbank erfolgt über eine Java-Schnittstelle. Benutzeranfragen werden aus dem Datenbestand heraus beantwortet, wobei geometrische Anfragen durch einen R*-Baum als raumbezogenen Index unterstützt werden. Das Anfrageergebnis kann mit den in der Arbeit vorgestellten Techniken zur graphischen Abstraktion aufbereitet werden. Zur progressiven Übertragung der dreidimensionalen Modelle wird in der vorliegenden Arbeit entwickelte P-Tree eingesetzt. Zur online-Kompression der 3D-Geometrie bei geringer Bandbreite des Netzwerkes wurde das Delphi-Verfahren implementiert.

Mit der Realisierung des CityServer3D konnte gezeigt werden, daß ein 3D-Geodatenserver mit der heute verfügbaren Technologie zu realisieren ist. Durch neue Konzepte in der graphischen Abstraktion und Kompression bei der progressiven Datenübertragung können 3D-Modelle auch in einer heterogenen Netzwerkumgebung performant genutzt werden.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

A geoinformation system (GIS) is used to collect and analyze information about the earth and to present this in various ways. A variety of models reflect our knowledge of the earth's surface and its buildings, of the people, animals and plants living on it. As a rule, commercial GIS support digital elevation models (DEM) of the earth's surface, although they only allow the spatial extend of objects on the earth's surface to be stored two-dimensionally.

As a result of the rapid development of new hardware and the progress achieved in the field of (semi)-automatic data acquisition, it is now possible to efficiently collect spatial objects in three dimensions. This development was promoted by a broad application spectrum of three-dimensional spatial information, e.g. in archeology, telecommunications and ecology, as well as in tourism and entertainment. Using three-dimensional models, environmental situations such as the spread of emissions and noise in cities can be simulated and presented to the citizen in a comprehensible way [Knol98]. In the field of city development, the real planning process is strongly supported by the availability of three-dimensional computer models. Alternatives can be better weighted, and laypersons can also understand the outcome of suggested modifications [Aria94], [Coor99]. A city's cultural heritage can be experienced by tourists [Coors00b].

In the telecommunications industry, the demand is growing for models that map urban regions as well as the earth's surface in three dimensions. In fact, in Germany the cell phone network carriers are the driving force behind the collection of comprehensive three-dimensional city models in metropolitan areas. Their first goal is a maximum net coverage for their cell phone network. In planning a network such as this, three-dimensional city models are used so that effects such as deadspots and reflections of electromagnetic waves by buildings can be taken into account [RaLa00].

The demand for appropriate three-dimensional models is confirmed by an OEEPE study of 3D city models conducted by the European Organization for Experimental Photogrammetric Science [Fuch98]; 95% of the participants believe that the greatest need for three-dimensional building data is in the field of a digital city model, followed by information about the traffic system (approx. 85%) and 3D information about the vegetation (approx. 75%).

As the above-mentioned examples make clear, digital terrain models of the earth's surface as well as three-dimensional models of urban regions and metropolitan areas are of particularly great importance. Nowadays, this data is collected using photogrammetric methods [Förs99], [Maye99], [BrHa00]. However, there is a lack of suitable tools for managing the data and making the data available to a large number of users.

Modern communication media provide new opportunities for making high-quality data accessible to users. The distribution of data is particularly facilitated by the Internet. Apart from increasing data accessibility, the Internet opens up new modes of communication. Urban planning schemes can be comprehensively described and discussed, thus meeting the growing need for information by decision-makers and citizens. As a result of the rapid development in the field of personal digital assistants (PDA) increasingly smaller and more powerful devices are being produced. The use of three-dimensional models on these ultra-portable devices, e.g. for purposes of personal navigation, is approaching reality. Because of the increasing data rate of mobile communication networks, digital information is omnipresent and can be accessed anytime, anywhere. The combination of spatial information systems and modern data networks will be an essential part of our everyday life in the future.

1.2 3D-GIS as a research field

A common approach for dealing with three-dimensional data is the combined use of 2D-GIS, CAD and visualization systems. CAD systems from architecture specialize in planning and constructing three-dimensional buildings. In contrast to GIS, work in CAD systems is done on a much smaller scale, usually between 1:1 and 1:50. These systems ae intrinsically inefficient for dealing with large-scale, wide-area models. Data collections are presented using specially made visualization systems. Visualization systems which are able to cope with a large amount of GIS data are derived from virtual reality (VR) applications. The aim of these systems is an interactive presentation of a very realistic environment, in which stereo sound and special presentation environments increase the level of immersion. Such VR systems often make great demands on the underlying hardware configuration and work with a local database. .

The integration of a three-dimensional geometric database and specialized thematic data plays an important role in 3D GIS. Combined spatial and thematic search queries can only be answered if this link is established. The search for a hotel near the train station and the search for available houses or apartments near child care facilities are two examples. Such search requests can be handled using a three-dimensional city model. As a general rule, the type of the building (e.g. hotel or child care facility) cannot be seen from the geometric description. What is necessary here is the integration of the three-dimensional model with existing data on the topic.

In order to distribute the collected data to a large user group, the data base must be accessible using communication networks such as the World Wide Web and mobile networks. The following core topics can be derived from this fundamental requirement, in relation to 3D GIS:

Conceptual data model

An important deficit of existing CAD and VR systems, is the lack of support for typical functional demands on geographic information systems, such as combined searches for spatial and semantic criteria or analyses of neighborhood relationships. For an efficient handling of these requests, a 3D GIS must be based on a suitable conceptual data model. This data model must depict information on the three-dimensional geometry and semantics of spatial objects, but also on the topological relations between the objects themselves. Along with semantics, this 3D topology plays a decisive role in the analysis of the database. The analysis function and the conceptual data model are thus directly connected. In addition to the pure geometry, it is necessary to administer descriptive attributes, in order to generate context-dependent presentations from the database. At the same time, because of the large amount of objects that must be managed, the storage space required by each individual object must be minimized.

Various authors have suggested conceptual models for 3D GIS, focussing mainly on 3D topology (see [Mole92], [Pilo96], [Flic98], Zlat[00]). All these models satisfy some of the requirements, but only support other aspects rudimentarily. A discussion of the most important approaches can be found in Section 3.3.

Use of the database in a heterogeneous environment

The economic efficiency of a 3D GIS depends greatly on whether the data can be utilized by a large number of users. Therefore, it is necessary to store and administer the data so as to provide data base access to a large number of users via a common network. The geodata server concept, which makes query and analysis functions and the database available via a network, has the following advantages:

- Cost-saving in the collection, administration, storage and exchange of data
- Savings of time and money for users due to a up-to-date database and to avoidance of time-consuming conversions and integration of different databases

- Greatest possible availability of data at all times and - with continuing development of mobile phone networks - in any place.

Because of the expected amounts of data, new concepts for data transmission must be developed to ensure to facilityte the efficient use of a 3D geodata server. The results of a query should be presented to the user in a suitable way. The quantity of results of typical queries such as region queries (that is, the search for objects in a particular region) includes a large amount spatial data. The question is how to suitably prepare the query result in order to communicate it to the user visually. This requires consideration of the informational goal of the user query, but also of the technical possibilities of the output device as well as the technical aspects of data transfer. Resource-adaptive visualization methods are of crucial importance in the context of a 3D geodata server, since database access can be realized using a large variety of end devices. To ensure the greatest possible data utilization, both high-performance visualization systems from the virtual reality field and ultra-portable devices such as PDAs must be supported. Thus different bandwidths of the communication networks used must be considered.

Previous publications in the 3D GIS context do not go beyond a level-of-detail concept for the visualization of the database [Kofl98], [Zlat00]. This concept is, in itself, not sufficient to solve the problem described above of result visualization in a heterogeneous network.

1.3 Aims of the thesis

In summary, the aim of this research thesis is to fill a gap left by existing 3D GIS: the generation of a meaningful, interactive, three-dimensional presentation of a selected database, especially in terms of the graphical abstraction and progressive transfer in a heterogeneous network environment. The intention here is to expand the concept of an openly distributed GIS infrastructure of the OpenGIS Consortium (OGC) [BüMc96] to three-dimensional geodata. For this purpose, a 3D geodata server will be designed and developed.

Core components of this 3D geodata server are

- a topological data model for the storage of three-dimensional geometry, and efficently supports 3D queries,
- methods for the presentation of query results and 3D city models in general
- progressive transfer and compression of 3D city models.

In contrast to previous work such as [Lang99], the main goal is not to depict a virtual landscape and/or an urban region as photorealistically as possible. Rather, the visual processing of the database should strive to support the user’s information needs. In contrast to computer-graphics rendering, in which a geometric model is depicted on a two-dimensional surface in proper perspective, the goal of the visualization is to transmit information to the user [Fole92]. Apart from the actual rendering, this process also requires a selection procedure that decides which elements of a scene contribute to the information content. This kind of visualization is called graphical abstraction.

The process of graphical abstraction is known as generalization in GIS and is traditionally applied in cartography for converting a map into another scale. During the creation of a map, a generalisation is always produced for a large target group that needs such maps. Road maps for car drivers are a typical example. Highways play an important role for this target group. On the map, they are enlarged, while other data such as paths through the woods or bicycle paths are completely left out. The availability of digital geodata e.g. on the Internet requires an increasing individualization and automation of the generalization process. A map is no longer produced for a user group, but individually created for the user with a specific need for information. The generalization must be completed in a very short time, according to individual aims.

For this thesis, techniques for graphical abstraction in 3D GIS have been developed. These techniques will be supported by a conceptual data model for an optimal depiction of the information. A corresponding model for a 3D GIS is developed as a basis for the processing of queries and the graphical abstraction and transfer of the query result.

In a heterogeneous network environment, the problem of resource-adaptive data transfer and visualization is second to the graphical abstraction problem. The question is not only how to suitably describe the selected data, but also how to develop an adequate data transfer that presents the viewer with an initial visual result as quickly as possible. Additional data can be transferred later. This progressive data transfer is crucial for the use of 3D data on the Internet [Ross98].

Besides graphical abstraction, a method for data transfer in a 3D GIS is the focal point of this thesis. Graphical abstraction and data transfer cannot, therefore, be considered independently of one another. Generally, graphical abstraction leads to a reduction of the data volume to be depicted. Consequently, an abstraction can also be used for 3D GIS applications in a network with low bandwidth.

1.4 Chapter Overview

In Chapter 2, the basics and concepts used in this thesis are explained. Furthermore, different approaches to graphical abstraction and 3D geometry transfer from related research fields are presented.

Chapter 3 provides the reader with a detailed introduction to the problem of graphical abstraction and Internet-based transfer in 3D GIS. This chapter also includes an overview of important work that has already been done and the state of the technology in the field. The chapter ends by evaluating existing approaches.

Chapter 4 introduces the overall concept of a 3D geodata server in a heterogeneous network environment. The integration of the 3D geodata server in to the general architecture of an open, distributed GIS infrastructure of the OGC will be discussed.

In Chapter 5 the core components of the 3D geodata server designed for this thesis will be examined in detail.

In Section 5.1, the data model for the storage of three-dimensional, spatial objects, which serves as the basis for the geodata server, is introduced. An important characteristic of this data model is the suitable presentation of the 3D topology, which allows efficient responses to queries.

In 5.2 other methods of graphical abstraction are developed, all of which are based on this data model. A model for evaluating the importance of objects in relation to the queries will be introduced. A graphical abstraction of the query results reflects the objects' importance on different output devices.

In Section 5.3, methods for transferring the generated graphical abstraction to a network will be introduced, focusing on the quickest possible visual feedback for the user and a progressive visualization refinement. Additionally, a new data compression method for 3D geometry developed in this thesis is used to reduce the data volume to be transferred.

Chapter 6 describes the prototypical realization of the concepts developed in chapter 5. The results are evaluated using this prototype.

Chapter 7 recapitulates the results of the thesis, evaluates them and suggests perspectives for future research on this topic.

Chapter 2

Basics

2.1 Geoinformation systems

Geoinformation systems (GIS) are used for administrating and analyzing spatial information. The geographic information system concept was first introduced in the 1960s to describe a novel, space-oriented information system. The complexity of these systems is clarified in the following definition, which specifies all essential components and functionalities:

„A geo information system is a computer-aided system, which consists of hardware, software, data and applications. It can be used to digitally collect, edit, store, shape and analyze space-oriented data and represent them alpha-numerically and graphically“ [BiFr99].

The spatial data managed in a GIS, also known as geo data, represent a cutaway of the real world. They differ from CAD system model data by their scale or by the size of the represented cutaway. Geo data are typically based on a scale between 1:1,000 and 1:1,000,000.

Typical application areas of GIS are planning and public administration. GIS supports all stages of spatial planning. A GIS project on a planning subject begins with the collection of data in the planning area. The situation is analyzed using this data and several problem solving possibilities are documented. The GIS also helps to evaluate alternative scenarios and can be used to make a graphical presentation of the planning results. GIS are used in many fields of city, regional, environment, marketing, routing and resource planning.

The earliest applications of GIS can be found in public administration. Two systems for collecting and updating of wide-area geo data are in use in the Federal Republic of Germany today:

- The automated real estate map (ALK) is the digital continuation of the real estate register and covers the scales between 1:1,000 and 1:100,000. The basic unit is the „Flurstück“ [Stöp87].
- The official topographic-cartographic information system (ATKIS) covers the middle scales between 1:10,000 and 1:100,000. It includes topographical as well as altitude information for the terrain [AdV99].

New GIS application fields

Global data networks and the cumulative advancement of information and communication networks to the private sector have increased the scope of for GIS applications in the last years. In particular, three new types of GIS application have emerged and will have a notable potential for the GIS market in the future:

- Navigation systems: Navigation systems integrated in vehicles ease orientation by providing specific hints for routes, redirections and interesting spots for tourists. There is also a strong trend towards navigation systems for pedestrians, trekkers or yachting tourists and ultra portable devices such as PDAs are emerging beneath navigation systems in cars.
- Points-of-information: These information columns are increasingly being installed in railway stations, airports and comparable locations. They can be used as a resource for up- to-date location-related information about traffic connections, routes or sites.
- WWW based GIS: Using the World Wide Web, current location-related information can be made available worldwide. For this application field, data are collected from various sources such as city planning, environmental protection and the tourist industry.

2.2 Data Models used in GIS

There are different levels for processing geo data in GIS. The first level includes data collection by remote sensing or photogrammetric surveys and image processing. In the second stage, the collected data are managed in a database system and prepared for analysis and specific queries. The third stage involves the suitable presentation of the query and analysis results. Each stage has special data management requirements. Consequently, Molenaar [Mole98] suggests using a special data model for each of the three levels. Additional methods are also necessary for transferring data from one level in the workflow to the next level, represented by arrows in fig. 2-1.

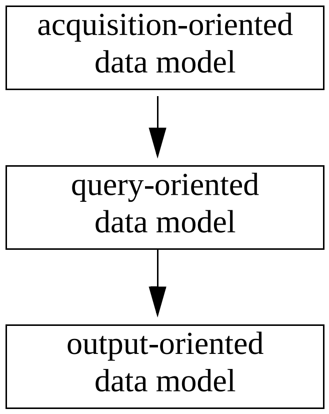

acquisition-oriented data model
query-oriented data model
output-oriented data model

Fig. 2-1: Specialized data models for different levels of a GIS according to Molenaar [Mole98]

- The first level requires a data model that efficiently supports the acquisition of data by various methods and also accelerates data collection.
- The second level requires a data model that permits the efficient processing of typical queries and analyses. In GIS this involves the extraction of thematic and geometric aspects of spatial objects and their spatial and topological relation to each other.
- The last stage requires a data model specifically designed for the presentation of data. Traditionally, spatial data are presented in the form of reports and 2D maps in GIS. However, the aim of the present study is to develop a three dimensional presentation of the underlying database in a heterogenically distributed network.

Each of these data models uses thematic and geometric data to model real world entities. [BiFr99] contains many examples. Thematic and geometric data can be linked by two different approaches:

- In the so-called field-based approach the space to be modeled is viewed as a spatial (and temporal) continuum. The various thematic aspects are assigned as attributes to a position in this continuum. Generally, the continuum is discretized with the aid of a regular two or three dimensional grid. Thematic data is then allocated to each element in the form of attribute quantities.
- In the entity-based approach, clearly identifiable discreet entities are represented as so-called *features*. A feature can exist in 0-, 1-, 2- or 3 dimensional space. The semantics of the feature is represented by a thematic description. The thematic and geometric data are allocated according to the identification of the feature.

In addition to the actual geometry, topological structures play a significant role in the modeling of geometric data. Topological models are used in the acquisition model for plausibility tests and for control purposes. They are also used in the query model for the efficient processing of contiguity of two features. These queries, which are not directly related to the actual geometry are also known as topological queries. The following section summarizes the basics of geometric and topological modeling of three dimensional objects.

2.3 3D-Representation Scheme

In contrast to traditional 2D GIS systems, the real world is modeled by a three dimensional representation in 3D-GIS. The problems encountered in creating the model are comparable to those experienced in solid body modeling in modern CAD systems, also known as Solid Modeling. Solid body modeling involves the creation of a solid model for each real object, which describes the closed three dimensional body exactly and completely. This model contains the information about the shape and closeness of the three dimensional solid, as well as the geometric connectivity, meaning the number of connected components and handles. The term solid model embraces numerous representation schemes. [Mänt88] describes the following criteria for evaluating representation schemes:

- How exactly can complicated solids be modeled?
- Does each model have just one representation or are there several descriptions per model?
- How much memory is required for representing practically relevant models?
- How general are the operations used for manipulating representations?
- How complicated are the algorithms for creating and manipulating objects?

Creating models is just one aspect of 3D-GIS. Another important function is the ability to analyze models. In view of this, the criteria for GIS representation schemes must be extended:

- Are discreet objects handled in different spatial reference systems?
- Can 0-, 1-, 2- and 3-dimensional objects be represented simultaneously?
- How efficient is the processing of thematic, geometric and topological queries?
- Is a flexible and efficient visualization of query results from the data model possible?

Encarnação et al. [Enca97] classify these representations as boundary representations (BRep), Constructive Solid Geometry (CSG), cell models and hybrid models. Fig. 2-2 illustrates how to represent a simple building using the different representations.

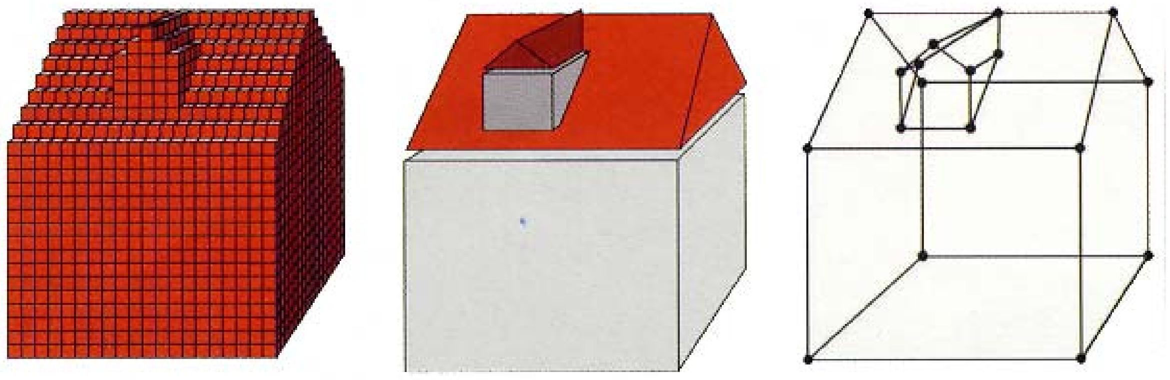


Fig. 2-2: Illustration of a building using cells, CSG, and BRep [Pfun02]

2.3.1 Topological Principles

The underlying concepts of topology as a branch of mathematics were first laid down by Euler in 1736 in his publication about the Königsberger Bridge Problem: Königsberg had seven bridges across the river Pregel, dividing the town in to three regions. The questions was: Is it possible to visit each region of the town by using each bridge just once? Euler represented the town regions as knots and the bridges as arcs connecting the knots. Using this model he was able to prove that the Königsberger Bridge problem could not be solved.

Generally, topology is used in GIS for modeling spatial relationships such as contiguities that can be manipulated independently from the actual geometry of the object. The topological basic elements are often described as nodes, edge or arc and face. The concepts of nodes and arcs are the basic components of graph theory.

Some more basic topological terms are described here. Please refer to [Will70] for a detailed explanation.

Manifolds

A n-dimensional manifold or n-manifold is a coherent set, where the surroundings of each point has the same topological structure as the n-dimensional Euclidean space. A 2-manifold has the property that each of the points is surrounded by space that is homeomorph to a circular disc in a plain. A 3-manifold is a coherent set, where each point is surrounded by space that is equivalent to the inside of a sphere in three dimensional Euclidean space.

Adjacency and Incidence

The relationship between nodes and edges within a graph are described by the terms adjacency and connectivity. According to Bill and Fritsch, these terms are used as follows in the GIS context [BiFr99]:

- "Incidence describes the 'interlocking' or 'nested' arrangement of the elements of a graph. For example an edge is connected to the start node and end node and consequently, all edges radiating from a node are connected to this node. Thus connectivity describes the relationship between the different elements of a graph"
- "Adjacency describes the contact between corresponding structural elements. Two nodes are adjacent when both are connected via an edge. Furthermore, two edges are adjacent if they meet in a single node. Thus adjacency describes the relationship between similar elements of a graph."

This understanding of incidence and adjacency are applied to higher dimensions. Thus incidence always describes the relationship between n- and (n+1)-dimensional structural elements and adjacency describes the relationship between adjacent elements of the same dimension.

2.3.2 Spatial Access Structures

GIS can handle two types of queries: thematic and spatial queries. Suitable indexing methods are used for finding relevant data. These methods avoid a sequential search of the complete data base. The data structures used for indexing thematic data only allow for a one dimensional search area. Consequently, spatial data requires special data structures to provide a multidimensional search volume.

Spatial indexing methods for two dimensional geometries are established in GIS. However, three dimensional geometries and 3D data queries require access structures for 3D geometries. Many of the spatial access structures used in GIS can be extended to include the third dimension.

Spatial access methods split space in to hierarchical regions. An example of such a spatial access method is the quadtree often used in GIS and introduced in the previous section. A quadtree [Same84], [Same94] splits space into non-overlapping regions. In contrast, the R-tree [Gutt94] splits space in to overlapping regions. In section the R-tree concept is expanded to form a hierarchical method for aggregating three dimensional spatial objects. For this reason the following section is a brief introduction to the R-tree and the R*-Tree based on Gaede and Günther’s survey of Multidimensional Accss Methods [GaGü98].

R-Tree

The rectangle tree or R-Tree was defined as a access structure for spatial data such as points, polylines and polygons in two dimensional space [Gutt94]. A R-Tree corresponds to a hierarchy of nested *d*-dimensional intervals (boxes). Each node *v* of the R-Tree corresponds to an object o_v and a d-dimensional interval $I^d(v)$. If v is an interior node, then the intervals corresponding to the descendants v_i of v are contained in $I^d(v)$. Intervals at the same tree level may overlap. If v is a leaf node, $I^d(v)$ is the d-dimensional axis aligned minimum bounding boxof the geometry stored in v. Other properties of the R-Tree include the following:

- Every node contains between *m* and *M* entries unless it is the root. The lower bound *m* prevents the degeneration of trees and ensures an efficient storage utilization. Whenever the number of a node's descendants drops below m, the node is deleted and its descendants are distributed among sibling nodes.
- The root node has at least two entries unless it is a leaf.
- The R-Tree is height-balanced; that is, all leaves are at the same level.

An appropriate R-tree is illustrated in fig. 2-3.

The advantage of a R-tree is the hierarchical splitting of space without fragmentation of the individual objects. The consequence of this very useful property is that the bounding rectangles may overlap on all levels. The tree is height-balanced, which means that all leaves are located on the same level and therefore are equally distant from the root. The height of the R-tree grows logarithmically as the number of entries grows.

One disadvantage of the R-tree is that the tree depends on the sequence in which objects are added. The reason for this is that only the minimization of the bounding rectangle is observed when objects are added.

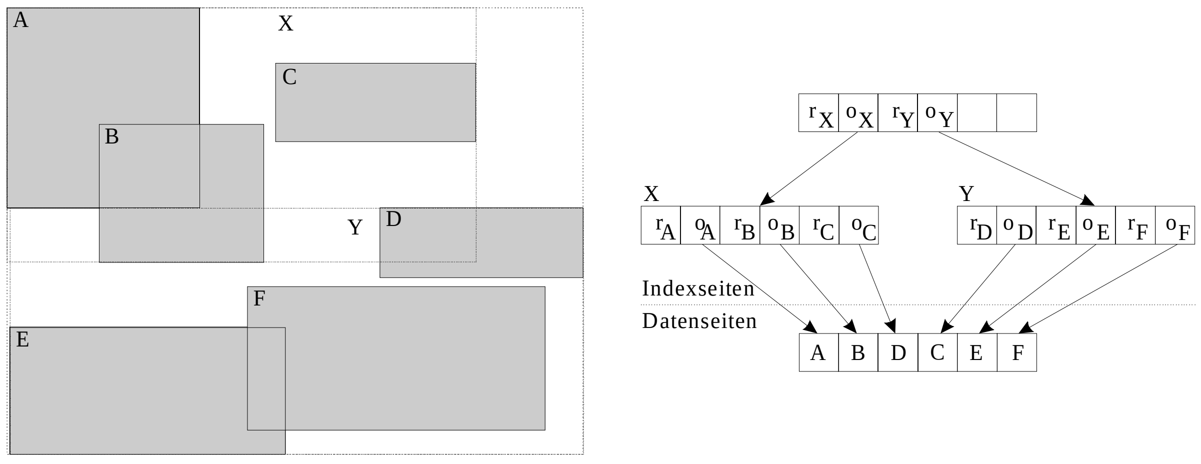

A
X
C
B
Y
D
F
E
r_X o_X r_Y o_Y
X
Y
r_A o_A r_B o_B r_C o_C
r_D o_D r_E o_E r_F o_F
Indexseiten
Datenseiten
A B D C E F

Fig. 2-3: Example of a R-tree where M=3 and m=2

Based on a careful study of R-Tree behavior, Beckman et al. [Beck90] identified several weaknesses of the original R-Tree. Inparticular, the insertion phase of the R-Tree has room for improvement. The design of the R*-Tree therefore introduces a policy called forced reinsert: If a node overflows, it is not split right away. Rather, *p* entries are removed from the node and reinserted into the tree. The parameter *p* may vary; Beckmann et al. suggest it should be about 30% of the maximal number of entries *M*.

Another issue investigated be Beckmann et al. concerns the node-splitting policy. Although Guttmann's R-Tree algorithms tried only to minimize the area covered by the intervals, the R*-Tree algoirthms also take the following objectives into account:

- Overlap between intervals at the same tree level should be minimized. The less overlap, the smaller the probability that one has to follow multiple search paths.
- Interval perimeters should be minimized.

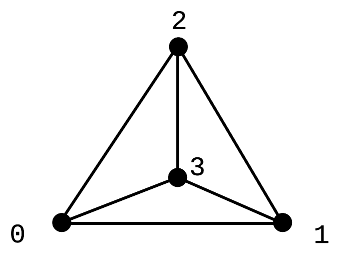
In summary, the R*-Tree differs from the R-Tree mainly in the insertion algorithm; deletion and searching are essentially unchanged. Beckmann et al. report performance improvements of up to 50% compared to the basic R-Tree.

2.4 Interactive 3D-Visualization in Networks

The magnitude of models in three dimensional computer graphics is increasing continuously. On the hand, this is due to the improved 3D design tools and data acquisition methods thus accelerating the creation of complex models and making these models cheaper. On the other hand, the required quality of such models grows as graphical hardware performance especially with regards to the cheaper workstations, but also in the field of high performance visualization increases. In the past, the scope for designers were limited by computer performance and the memory available to workstations. These technical limitations are reduced by further development of the hardware, so that complex models used in car construction, architecture and entertainment can contain several million triangles.

Databases can be accessed at any time and from any place with modern communication networks such as the internet. Mobile networks will have sufficient bandwidth in the future for usefully transferring three dimensional data. It is expected that the available bandwidth in the mobile network will increase from currently 9,6 Kbit/s (GMS) to a maximum of 2 Mbit/s following the introduction of the UMTS standard. The increasing bandwidth of the transmitting media will result in a further increase in the size of the models. This is why methods for compressing and simplifying 3D graphics have become a central part of computer graphic research in the past few years. The problem at issue will be explained with the help of an example, before the current methods for reducing data volumes are introduced.

The de facto standard for storing and visualizing 3D models are *triangular meshes*. For this reason the number of triangles in a model is an adequate measure of the complexity of the model. Widespread graphic libraries (such as OpenGL, DirectX and Java3D) and data exchange formats (such as VRML) usually store the vertices and the connectivity between triangle surfaces of these triangular meshes in the form of a vertex reference. A polygonal model with *V* vertices contains approximately *2V* triangles. If the memory space required for each of the three coordinates of a vertex is 4 Bytes and the reference also requires 4 Bytes then a total of 12*V + 12*2*V Byte is required to store the model. This is equivalent to 36 Byte per vertex or 18 Byte per triangle. This does not include color and texture data.



```
IndexedFaceSet {
   coord Coordinate {
      point [ 0 0 0, 2 0 0, 1 2 0, 1 0.5 1 ]
   }
   coordIndex [ 0, 1, 3, -1, 1, 2, 3, -1,
                2, 0, 3, -1, 0, 2, 1, -1 ]
}
```

Fig. 2-4: Tetrahedron in VRML97

fig. 2-4 Shows the display of a tetrahedron in VRML97.

Any private user who wants to access a model via the internet using a modem connection with a bandwidth of 64 Kbit, will experience the following conditions. The cost of 18 Byte per triangle allow a transmission of 444 triangles per second via a 64 Kbit cable. Studies on software ergonomics have shown that internet users accept an idle time of 10 seconds before becoming impatient. This results in a model size of 4440 triangles for internet based work.

According to an appraisal by Flick, a model of downtown Frankfurt consists of 13.5 million triangles. This is based on an estimated 90 buildings per hectare consisting of 250 triangles per building. Therefore the data volume for a purely geometrical description would be approx. 250 Mbyte. The transfer of data required for publishing this model on the internet would take about 9 hours with a 64 Kbit/s ISDN connection and two and a half days with a mobile net permitting 9.6 Kbit/s. In a high speed net with 100 Mbits/s it would still take 20 s. These estimates are only valid for optimum bandwidth. As a rule this is not available since other users use the network. To improve user acceptance measures must be taken to increase data transmission rates even in this case of a limited model that does not cover a whole area completely. To put the size of the model in to perspective, Kofler estimates that a complete city model of Vienna would contain 500 Gbyte of data [KoGr98]. Consequently, there is a strong need for methods for using complex 3D models in networks with low bandwidth.

2.4.1 Approaches for Reducing Data Volumes

Different approaches exist for reducing the transmitted data volume. The methods can be categorized as follows. As a rule, these methods do not preclude each other, but can be combined.

- **Geometric compression:** Compression involves the reduction of the number of bits of a 3D geometry to be transmitted. The compression of the geometry is a lossless process, so that the original geometry can be reconstructed completely.
- **Geometric simplification:** The process of geometric simplification of 3D models is based on the fact that the complete object geometry is not always required in order to visualize the model. Geometric simplification aims to reduce the number of triangles that represent a 3D object. Consequently, this compression method will result in some loss of information. The 3D object is depicted more or less accurately depending on the level of detail. The use of geometric simplification methods of 3D models is discussed in more detail in section 2.5.3. Of course, the levels of detail can be further compressed.
- **Progressive transmission:** The method of progressive transmission of a 3D model initially transmits a rough model. Data for adding details to this rough model are sent in the next stage, until the required level of detail has been achieved. Progressive transmission uses both geometric simplification and compression procedures.

2.4.2 Geometric Compression Methods

With the use of specialized methods for compressing 3D geometries, such as those introduced by [GuSt98], [ToGo98], [TaRo98] and [Ross99], much higher compression rates can be achieved than with general compression tools such as *gzip*.

Normally the compression of 3D geometries assumes that the geometry exists as a triangular mesh. The topology of the triangular meshes is compressed separately from the geometry. In this section, several techniques for the lossless compression of the connectivity of triangle meshes will be reviewed.

The connectivity of a “simple” mesh (defined as connected, zero-genus, manifold triangle mesh) may be stored as a sequence of t triangle descriptors, each triangle been represented by 3 integer labels. Each label identifies one amongst v vertices and hence requires $\lceil log_2 v \rceil$ bits.

Organizing triangles into strips [Evan96], where each new triangle shares an edge with the previous one, reduces the above storage by nearly half. The use of a buffer to cache a small number of labels [Deer95] may further reduce the expected cost.

The **Topological Surgery** method of Taubin and Rossignac [TaRo98] compresses both a triangle-spanning tree and its dual vertex-spanning tree by encoding the lengths of consecutive single-child nodes. Both trees suffice to decode the connectivity of the simple mesh. For complex and reasonably regular meshes, the expected cost of encoding both trees amounts to less than two bits per triangle. However, the overhead of the run length encoding may result in a significantly higher average cost for irregular or small meshes.

Rossignac's **EdgeBreaker** compression scheme [Ross99] was the first one to propose a tight linear worst-case bound of the connectivity compression. The original method guarantees 4 bit per vertex (denoted b/v for simplicity). This guaranteed upper bound was later improved to 3.67 b/v [KiRo99] and to 3.60 b/v [Gumh00]. This upper-bound on storage does not rely on statistic-based entropy or arithmetic coding schemes, which are discussed below and, in general, perform poorly on small or irregular meshes.

EdgeBreaker visits all the triangles of a mesh one at a time, walking from a previously visited triangle to one of its not-yet visited neighbors through their common edge, called the "gate". For manifold meshes, the tip of the new triangle is either a "new" vertex (case C) that has not yet been encountered or an "old" vertex of the boundary separating the previously visited portion of the mesh from the rest. EdgeBreaker distinguishes four types of "old" vertices, depending on whether they appear in that boundary immediately before the gate (case L), immediately after the gate (case R), both (case E), or neither (case S).

Because half of the triangles correspond to case C in a simple mesh, one may chose to encode them using a single bit (for instance 0), while the remaining four cases may be unambiguously encoded using 3 bits each (110 for L, 101 for R, 111 for E, and 100 for S). This simple code guarantees to compress the EdgeBreaker-generated *clers* sequence of any zero-genus manifold mesh down to 4.0 bits per vertex. The *clers* sequence of meshes encountered in practice may be compressed even further, sometimes to less than 1.8 bits per vertex, using variable-length entropy codes [RoSz99]. When the mesh has a sufficiently large number of vertices and most of them have exactly six neighbors, the *clers* sequence can provably be compressed down to 1.62 bits per vertex [Szym01]. A detailed description of the EdgeBreaker compression is given in section 5.3.2.

Although developed independently, EdgeBreaker bears strong similarities with Gumhold and Strasser's **cut-border machine** [GuSt98]. For manifold meshes without holes, the principal difference lies in the fact that cut-border explicitly encodes an offset with each S symbol, while EdgeBreaker does not.

Similarly to the Topological Surgery, to EdgeBreaker, and to Cut-Border, Touma and Gotsman's **valence coding** [ToGo98] also encode the vertices along a vertex-spanning tree. However, instead of the five EdgeBreaker cases, they distinguish only two cases, split and add, which correspond to the EdgeBreaker's cases S and C. Instead of the other symbols, they encode the valence of each vertex, i.e., the number of incident triangles. Thus, each C symbol is associated with an integer valence and each S symbol (as in the cut-border machine) is associated with an integer offset. Their decoder keeps track of the valence defect of all border vertices. When the defect of a vertex v reaches 1, a new triangle is automatically inserted in the star of v, turning it into an interior vertex. Thus L, R, and E triangles are recovered automatically and need not be encoded. Experimental results show that this approach compresses connectivity down to less than 0.2 b/v for very regular meshes, and between 2.0 and 3.5 b/v for the less regular meshes found in practice.

Alliez and Desbrun [AlDe01] replace the deterministic traversal of Touma and Gotsman's approach by an adaptive traversal that attempts to minimize the number of split operations (EdgeBreaker's S cases). To do so, they choose for gate the border edge bounded by vertices with the smallest number of free edges.

This optimized gate selection was further improved in **Angle-Analzyer** [Lee02], where Lee, Alliez and Desbrun use both connectivity and geometry to direct the mesh traversal. They select the gate using connectivity and geometry criteria, as the border-edge having the highest number of incident triangles and also the minimum angle with a neighboring border edge. The optimal combination of these two criteria dramatically decreases the frequency of S symbols. Thus, by combining the simplicity of EdgeBreaker with the statistical benefits of valence-based coding, they improve connectivity compression by 40% over [ToGo98].

2.4.3 Geometric Simplification Methods

Geometric simplification methods were developed to accelerate the visualization of very large three dimensional scenes and to facilitate their interactive use in virtual reality applications. Different levels of detail of a complex model can be achieved with special methods, which are used according to the size of the object and the distance from the viewer.

This concept of different levels of object detail can also be used for data transfer. Levels of detail that are not used for visualization do not need to be transferred to the client. This means that the transferred data volume of large scenes including many complex objects can be reduced substantially.

Reducing Data using the Level-of-Detail-Concept

The basic idea behind the level of detail concept is that distant, small or less important objects do not need to be visualized in all detail. The degree of abstraction can be individually determined for each object according to the visibility. A measure of the visibility is given by the size of the object and distance from the viewer. In a perspective projection, small distant objects are projected on a very small area of the viewing plane. As a consequence the details are reduced to a few pixels and the use of a detailed geometric model is unsuitable in these cases.

In the next stage, the visualization of the scene requires the use of representations that combine a suitable level of detail with a low degree of geometric complexity. This method should allow real-time navigation within a virtual scene while including as many objects in the visualization as possible. The method required for this purpose must provide a suitable compromise between the quality of the representation and the absolute recall speed.

Level of detail methods can and should be used to to avoid delays during interactive navigation through a large scene. Using this method several geo-object representations with varying degrees of detail can be generated.

Generating the Levels of Detail

The levels of detail used in the visualization can be generated by:

- hand,
- automatically from complex primitives or
- automatically from complex models.

Generating levels of detail manually usually delivers the most appealing visual result, because the author of the scene is in complete control. Under these conditions, the level of detail can be determined individually for each point for each polygon model. However, this is a very laborious task when millions of polygons are involved.

If the geometry is present in the form of a triangular mesh, the second method is also useless. The only primitive present in the geometric description is the triangle. There are several approaches for simplifying triangular meshes. The two most important approaches will be outlined in the following. [HeGa97] contains an overview of further approaches for simplifying the geometry automatically.

Vertex Clustering

One of the first and most famous procedures for reducing the complexity is vertex clustering, developed by Rossignac and Borell, which is based on a condensation of space [RoBo93].

This procedure simplifies any polygon model while maintaining the original appearance of the model according to the level of abstraction. The method consists of the following four phases that are described in more in the next section:

- grading
- clustering
- synthesis
- elimination or replacement

Starting with a triangular mesh, each vertex of triangle is graded which determines the visual result in the synthesis phase.

In the next step the bounding box parallel to the axis is split into a regular user-defined grid in a process called clustering. All vertices that lie within a so-called cluster are grouped. The size of the cluster determines the level of abstraction: the larger the clusters, the coarser the final approximation.

In the next synthesis the points of a cluster are moved to a new point. The position of this representative is calculated from the graded average of all vertices in the cluster. This process is also known as the fusion of vertices.

In the last phase degenerated triangles are eliminated or replaced. Two or all three vertices of a triangle can be projected on to a single point in the synthesis phase. Such a triangle does not have an area and is therefore known as a degenerate triangle. These triangles are either deleted or replaced by an edge or a point. Note: the complexity of the model is reduced in this phase. After this process the geometry is updated by projecting the vertices on to the appropriate replacements.

A case example of this method is depicted in fig. 2-5. In this case, degenerate triangles are not eliminated but replaced by points. As fig. 2-5 illustrates, the algorithm is based on a filtering procedure which processes the points of a polyhedron according to their position within a a spatial grid.

The distance between a vertex and its replacement is governed by the size of the particular cluster. This relationship results in a number of disadvantages for this process. Thus the abstraction of triangulated planar surfaces can fail if the cluster is too small to enable the grouping of the vertices. In some cases this may result in unsatisfactory visual effects, since vertices that are connected via an edge to each other will be fused (simplification of the topology). This effect is also illustrated in fig. 2-5. Here a triangle and a polygonal surface are fused to a single object. Consequently, not every detail is preserved.

Rossignac and Borell use a regular grid for their algorithm to determine clusters. Howev-

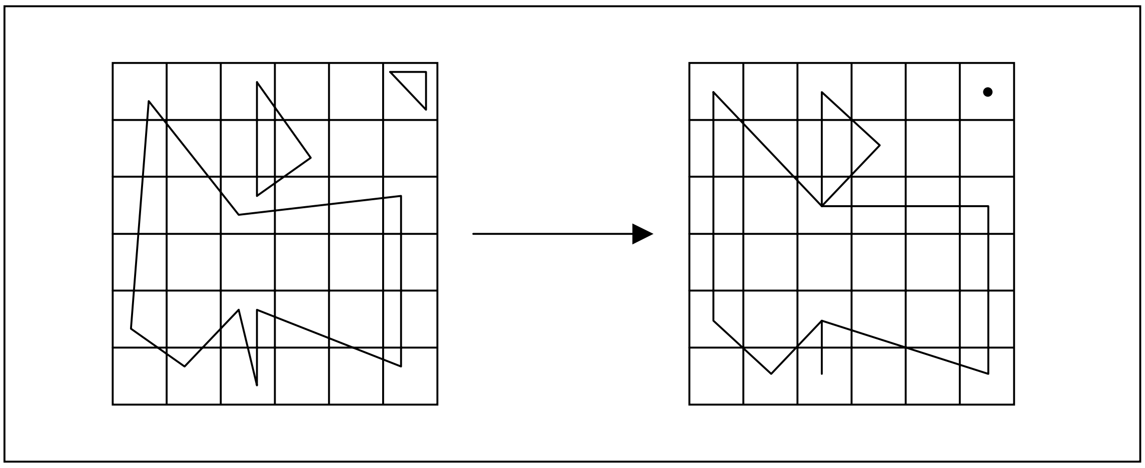


Fig. 2-5: Vertex Clustering

er, many systems use an extension of this principle by using a data sensitive determination of the cluster size, which also enables the abstraction of planar surfaces. Other modifications of the procedure only concern the grading of the vertices. Principle possibilities are, for example: a high grading of all vertices of the silhouette of the model or a grading according to the particular edge length.

The vertex clustering method is very general, meaning that the entry data can be any set of points, edges or triangles. A topological restriction of the entry data is not necessary. Thus three or more triangles can share a common edge, any polygon section is possible (for example surfaces with self penetration). Only a few simplification procedures can process this type of general model. A selection of the particular procedures are introduced in the following section. In addition, this algorithm is extremely fast, allowing calculations in real time applications.

The algorithm is particularly useful for eliminating holes, simplifying thin objects in to planar objects and elongated objects in to line segments. Any number of levels of detail can be generated. Small objects can be eliminated, however if elimination is not desirable then precautions must be taken.

This procedure is regarded to be efficient, simple and robust. For these reasons, it has been used in several commercial products such as "OpenGL Optimizer" by SGI and "3D Interaction Accelerator" by IBM.

Edge Collapse

The edge collapse method is based on the condensation of the geometry. A edge collapse involves projecting the two vertices of an edge on to a single position. During this process, triangles that are limited by this edge are eliminated. Generally this involves two triangles. In the case of a marginal edge, meaning an edge that is assigned to only one triangle, only a single triangle is removed.

Fig. 2-6 shows an edge collapse and the splitting of a vertex. The vertex split is the reverse of an edge collapse.

The result of an edge collapse is comparable to the fusing of two vertices in a cluster. The fusion results in the degeneration of two triangles that are then eleminated.

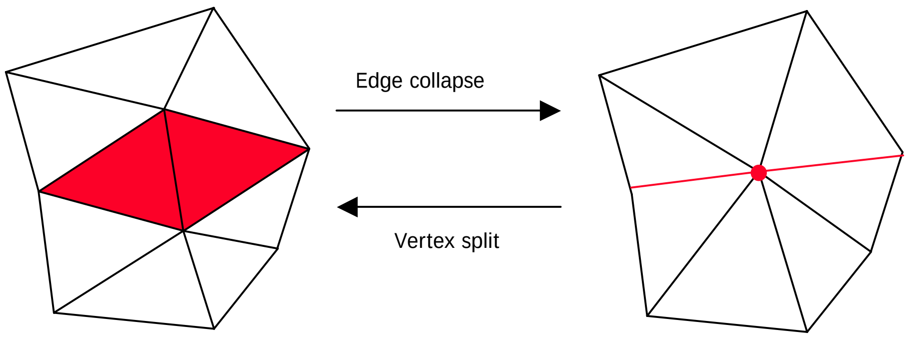


Fig. 2-6: Edge Collapse and Vertex Split

The process collapses one edge after the other. The sequence is determined by the cost associated with the collapses. For example, the operation will always eliminate the particular edge which causes the least geometric discrepancy with regards to the original model.

Edge collapses are carried out until a given error margin has been surpassed or the desired number of triangles has been attained.

Edge collapses are equivalent to surjective images, which assign a representative for the vertices. The various procedures differ only in the method for determining the resulting error, which is limited by the level of accuracy set by the user. More information on this subject can be found in [Hopp96] and [Ross98].

Evaluation

Vertex clustering is a very robust and fast procedure for generating several levels of detail for geometric objects. The method does not impose restrictions on the original geometry. However, vertex clustering delivers relatively poor results for highly simplified objects.

Methods based on edge collapse yield a noticeably higher quality in the level of detail than vertex clustering. However, edge collapse methods can only simplify coherent geometries. These methods fail in models containing many small separate objects. Extended procedures such as [GaHe97] solve this problem by utilizing a distance operator. This operator also lets separate points collapse, providing they lie sufficiently close to each other.

The methods for geometric simplification generally create several, independent levels of detail. These static levels of detail store geometric data as redundants, which leads to a higher data volume transfer in a network. However, slight modifications enable these procedures to generate so-called multi-resolution models necessary for a progressive transfer.

2.4.4 Procedures for the Progressive Transfer of 3D Models

The progressive transfer of 3D geometries initially involves the sending of a coarse model to give the user a first impression of the model early in the proceedings. Then data are transferred for adding the detail to this coarse model until the required level of detail of the model has been attained. This improvement of the model is often based on the *Load-on-demand* paradigm, which means that only those data are transferred if they are required for visualization.

Apart from providing a quick first impression of the model, progressive data transfer also has advantages with regards to the safety of transfer. The user will have at least an approximation of the model to work with, even if the data stream has been interrupted.

Redundant Transfer of Several Levels of Detail

The simplest solution for progressive data transfer of 3D models is the generation of a simplified representation geometry using a *level-of-detail* procedure. The user can be provided with this simplified model from the start. The original model will then be transferred in full detail at a later point in time. In the intermediate period, the user can interact with the simplified model. Once the original geometry is completely transferred, the simplified model is replaced by the more complex one. Various *levels of detail* between the simplified model and the original geometry can be transferred by the same means. This procedure is similar to the usual implementation of the *level-of-detail* concept for visualization. In this concept, an object will be represented in more or less detail according to the distance between it and the viewer. In contrast to progressive transfer, all levels of detail are stored in the main memory in purely visualization procedures.

The disadvantage of the procedure is the complete independence of the various *level- of-detail* geometries. The information transferred in the coarse model is not taken in to account in the complex model. The high redundancy of this procedure means that the volume of data to be transferred accumulates for all levels of detail. If the highest level of detail is required for all objects, the total data volume is doubled.

The procedure of pointwise refinement of a model begins with the addition of single points to an initial coarse model. The accuracy of a model increases incrementally and each small step results in an improved resolution for the user. However, the coding of the pointwise refinement is very complex. The flexibility of the procedure is bought with a concomitant loss of compression rate. The transfer time is very much higher for a pointwise refinement in comparison with a non-progressive procedure.

However, the refining allows for a greater control of the accuracy and can be carried out in relation to the point of view and the direction of view. A well known procedure for pointwise refinement is Hoppe's "Progressive meshes" algorithm [Hopp96].

The procedure begins with the transfer of a coarse model. This model is then progressively refined by a series of vertex splits (see. fig. 2-6). An extension of Hoppe's *Progressive Meshes* utilizes "adaptive" refinements in relation to the direction of view [Hopp97]. This avoids that invisible areas of the model are refined such as those at the back of the model.

Refining the 3D Geometry in Batches

Techniques for refining models in batches have beneficial properties. The accuracy of the model is unchanged, while the next batch is loaded. The transferred data volume is reduced in comparison to pointwise refining procedures, since large packets can be compressed more efficiently. Slightly more storage space is required than for a model without levels of detail. For example, the progressive data structure according to Guéziec [Guiz99] is maximally 10% larger than the original model without levels of detail.

Progressive Meshes

The *Compressed Progressive Meshes* (CPM) introduced by Pajarola and Rossignac [PaRo99] is a procedure for refining 3D geometries in batches. The procedure is based on Hoppe's Progressive Meshes [Hopp96]. However, a much reduced transferred data volume is achieved by forming batches and compression.

The original geometry M_∞ is progressively simplified by edge collapse which results in a set of triangular meshes M_∞,...,M_{i+1}, M_i, ..., M_1, M_0 with diminishing accuracy. The difference between two adjacent levels of detail M_{i+1} and M_i are stored in a batch $M_i \rightarrow M_{i+1}$. The simplification of the geometry of a coarse model M_0 ais stopped when the error threshold has been achieved. This inaccurate model is transferred initially using any effective procedure for compressing simple triangular meshes.

The following restrictions apply for M_{i+1} edge collapses during CPM:

- No more than two vertices may be collapsed in to one
- No more than two edges may be collapsed in to one
- All edge collapses are represented by the middle point of the edge

In order to fulfil these restrictions no edge may be collapsed if it is adjacent to an edge that has been collapsed in the same batch. Additionally, no edge of a triangle may be collapsed if the triangle is adjacent to a triangle collapsed in thie same batch.

The following information is required for the transfer of a batch in order to proceed with a *vertex split* for refining the model:

- All vertices that were created by edge collapse in this batch (*split-vertex*).
- All edges that were created by edge collapses of degenerated triangles in this batch (*cut-edges*).
- Position of the vertices A and B of the collapsed edge.

The position of the vertices A and B of a collapsed edge are coded on the basis of previously adjacent vertices. The position of the vertices is predicted by a linear combination of the adjacent vertices (*geometry prediction*). the restrictions mentioned above mean that the geometry of adjacent vertices in the coarse model is always known. Consequently, all that is required is the coding of a factor and a potentially occuring error. A detailed description of the CPM format as well as coding and decoding can be found in [PaRo00].

Evaluation

The CPM procedure requires approx. 3.5 bits per triangle for *connectivity coding*. Geometry coding requires 7 to 10 bits per triangle. fig. 2-7 shows the results of a CPM procedure:

Compression means that the CPM procedure requires 50% less storage space than the Hoppe's original *Progressive Meshes* procedure. Experimental results have shown that the CPM format achieves a similar degree of compression to well known non-progressive formats [TaRo98, TuGo98].

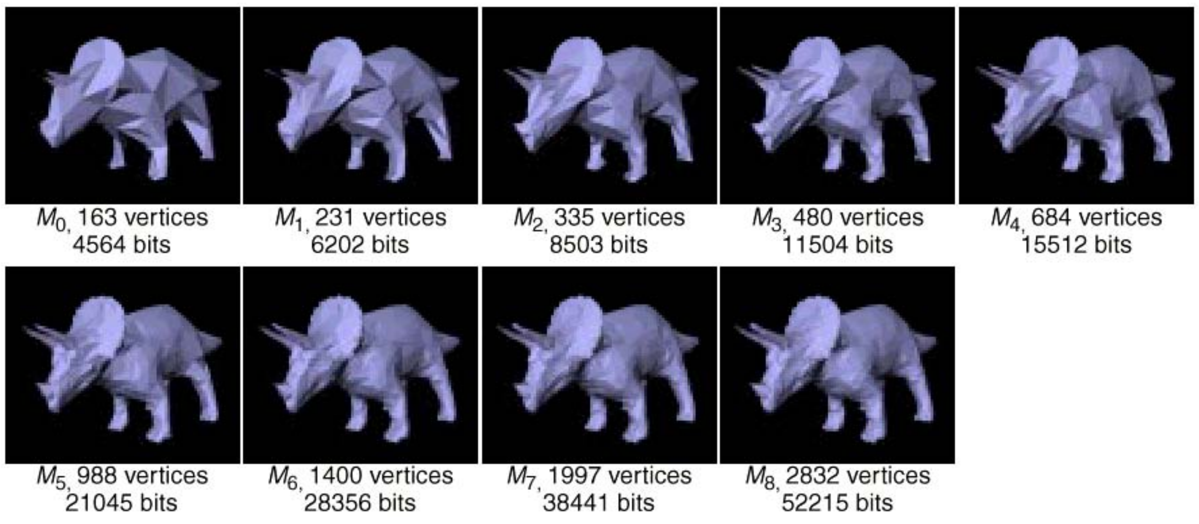


Fig. 2-7: Compressed Progressive Meshes

2.5 Graphical Abstraction

2.5.1 Perception Systems in Humans

Extensive knowledge of the human visual system form the basics for a precise graphic abstraction. Intensity, wavelength and spatial distribution of the incoming light are all processed by the human retina. Accurate vision with high resolution is only possible in a circular area approx. 4 degrees in diameter around the so-called yellow spot (Fovea) [Wand92]. The fovea is the location on the retina on to which fixed points are projected. A foveal distance of two degrees angle of vision is equivalent to a distance of approx. 20 mm from the fixed point on the screen, when the viewing distance is about 50 or 60 cm. The foveal distance of a perceived object and the relative size of the retinal projection is illustrated in fig. 2-8.

Peripheral vision occurs when a perceived object is displaced by more than two degrees from the fovea and projected on to the retina. Peripheral vision is out of focus in comparison with foveal vision. Nevertheless, peripheral vision enables the perception and localization of stimuli and delivers data required for spatial orientation and the control of eye movements.

Ballistically controlled eye movements, the so-called *Sakkaden* fix the perceived object and bring it in to the foveal area. Directly after a *Sakkade*, which lasts approx. 30 ms, the eye remains fixed for about 60 to 700 ms for recording information.

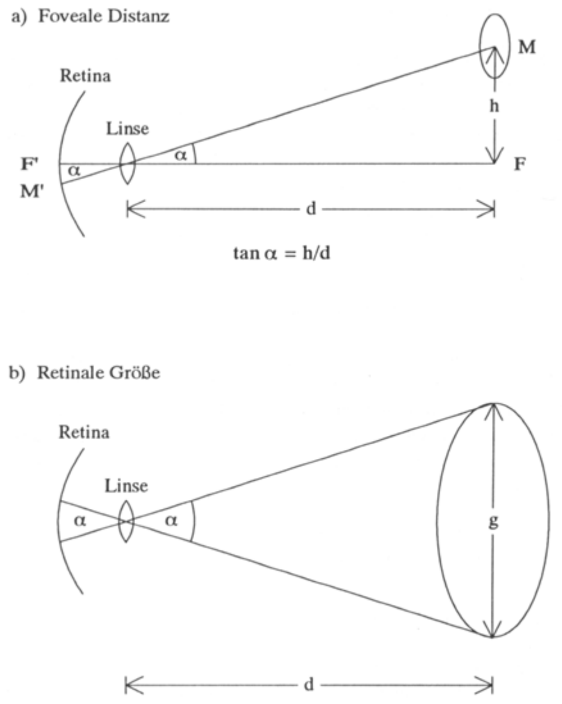


Fig. 2-8: a) Foveal Distance of a perceived object in degrees angle of vision. F is the point of fixation and M is the center of the circular perceived object. F’ and M’ are the projections of F and M on to the retina. b) Size of the retinal projection of a perceived object (circle) in degrees angle of vision. [Wand92]

Context Information

The interaction between characteristics and context in visual perception are extremely important for graphical abstractions. Objects seen in a context require only a small amount of information about the characteristics in order to be identified. If the particular objects are shown in isolation, more visual details are required [Ande96]. This is exemplified in in fig. 2-9 by a door shown with and without context. In the context of a house, a simple rectangle suffices to classify the shape as a door. However, the same geometry in isolation is not easily recognized as a door. More details are required in order to classify the geometry. As a result the identification of objects is based on their context. Context information is added to characteristic information.

According to Biederman, Beiring, Ju and Blickle [Ande96], the recognition of objects is based on the identification of the components of the particular object. Consequently, objects are recognized as configurations of object component and it is not suitable to remove these components, for example a roof of a house, for simplification purposes. It is possible to simplify components, but they should still remain in the representation.

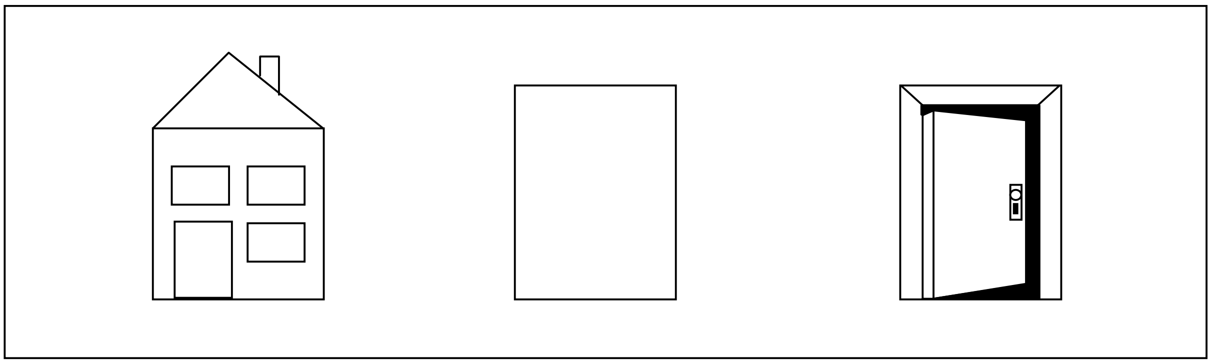


Fig. 2-9: Interaction between characteristics and context: exemplified by a door [Schu99]

2.5.2 Abstraction in Computer Graphics

In computer graphics, rendering means the perspectively correct projection of a 3D model on to a 2D plane. A geometric object will generate as much information as is necessary and required. The visualization of an object is generated with the help of standard software such as VRML-browsers. Apart from the possibility for selecting the point of view, rendering proceeds without the possibility for intervention by the user. This means that a model can normally only be used for the purpose it was created for. A model of a workpiece or product from a CAD system that was modeled for planning purposes, cannot be used in technical documentation, since the requirements for applications are different. Whereas planning often requires a photorealistic impression of the product to be developed, technical documentation requires the depiction of the functional relationship of the different components. In this case, important elements are emphasized in order to attract the users attention. These requirements must be carried out at the level of the model in the usual rendering pipeline used in computer graphics.

In contrast to this, graphical abstraction in computer graphics is aimed at deriving and representing the most important aspects of a model. Strothotte defines abstraction in computer graphics as follows:

"*a process by which an extract of an information space is refined so as to reflect the importance of the features of the underlying model for the dialog context and visualization goal at hand*" [Stro98].

Such a graphical abstraction cannot be carried out on a purely geometric basis. Apart from the actual representation it also requires a decision about which elements are important to the user in a specific context. These will then be emphasized in the representation. Some knowledge of the importance of the spatial objects and the context of the presentation is important. Strothotte distinguishes between geometric and symbolic models. While the geometric model describes the form and the appearance of the model, the symbolic model contains all the data that do not directly influence the appearance of the object. Together, the geometric and the symbolic model form the information space. This distinction is carried out in a similar manner in GIS. Here a spatial object is specified according to the three component model for object types or objects by combining the geometric data, graphic data and content data [BiFr99]. Geometric and graphic data are equivalent to the geometric model, content data is analogous to the symbolic model.

Methods for reducing model complexity are ubiquitous in computer graphics. These involve the reduction of the number of polygons on the basis of graphical and geometric attributes to an extent the user cannot visually distinguish in the representation. Appropriate procedures were discussed in the preceding section. However, only a few approaches allow the simplification of the model that can take in to account a specific selection of the important elements to be visualized as required by the user. The necessity for such a graphic abstraction was mentioned by Feiner [Fein85]. His APEX system generates a graphic image in accordance with a specified communicative target. This requires abstraction methods in order to select the degree of detail of the object to be represented in relation to the importance in fulfilling this target.

Butz and Krüger [BuKr97] investigated, which areas of an image are important recognizing the represented information effectively. For example the silhouette of an object and significant attributes play a central role in the recognition and must be retained for important objects. It is assumed that different objects contribute to varying degrees towards achieving the presentation target and consequently demand different levels of attention from the user.

Objects within the content focus are especially important for the graphic and must be represented in full detail. All objects that are not within the content focus are called background objects. These objects are further distinguished according to their importance for the presentation. A focus object is identified by landmarks. Consequently, these landmarks must be presented in a way that enables the orientation of the object. In case of the remaining objects, a decision must be made whether a visible presentation is necessary or whether the objects can be omitted in the presentation. Fig. 2-10 shows an exemplary abstraction focussing on part of the image of a motorblock.

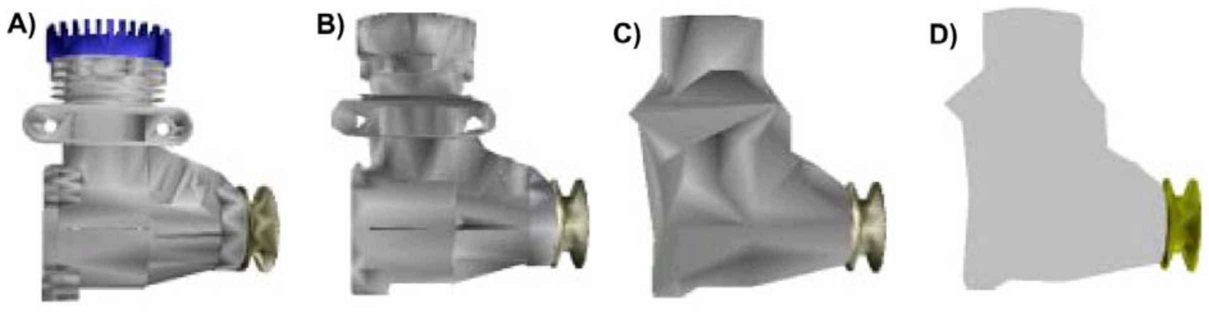


Fig. 2-10: Abstraction motorblock [Krue98]

In an animation this focus structure is defined by the presentation planner according to specific rules [Krüg98]. The consistency rule for example, defines the maximum degree of abstraction for objects according to their importance for the presentation. This prevents objects that gain importance during a presentation from appearing suddenly from nowhere.

In contrast to this, the focus structure is not fully planable in interactive presentations. The presentation must react to user interaction, which can be equated with user interest, by changing [Hart99] the focus structure. The nodes of this network represent the objects in the presentation. Each node is assigned a dominance, which defines the importance of the particular object in the presentation. The relationship between values of dominance for different objects are represented by the edges of the semantic net. The interactions of the user, for example by selecting a text section, can modify the values of dominance for particular objects. The main problem of this approach is the favorable parametrization of the semantic net, either by training with predefined examples or by using a rule based approach.

2.6 Generalizations in Cartography

Maps are more or less abstracted models of the real world or reality. The reality is represented symbolically and in generalized form. According to [Hake02] the reasons for a generalization are:

- Even the initial acquisition and processing of the object information requires generalized procedures (e.g. metrological simplifications, creation of classes)
- The design of the map has an effect on the scale and the dedication of the cartographic appearance and therefore affects the extent and type of presentation of the object information.

It is possible to differentiate between *acquisition generalization* and *cartographic generalization* according to the different circumstances.

Acquisition generalizations occur en route between object and map. For example a survey will neglect small projections on buildings and local objects such as masts are only described by their center.

Cartographic generalizations occur en route from a detailed map to a smaller scaled follow-up map. As the scale of the map is reduced, every geometric presentation of an object shrinks until it falls below the minimum threshold value required for graphical presentation. In this case the object will not be depicted. Since this is impractical and contrasts with the actual demands made on the map, it is necessary generalize the objects to be represented.

The first stage of this process involves the selection of the information to be included on the map. This selection will depend on the theme of the map and the map scale. The selected infor-

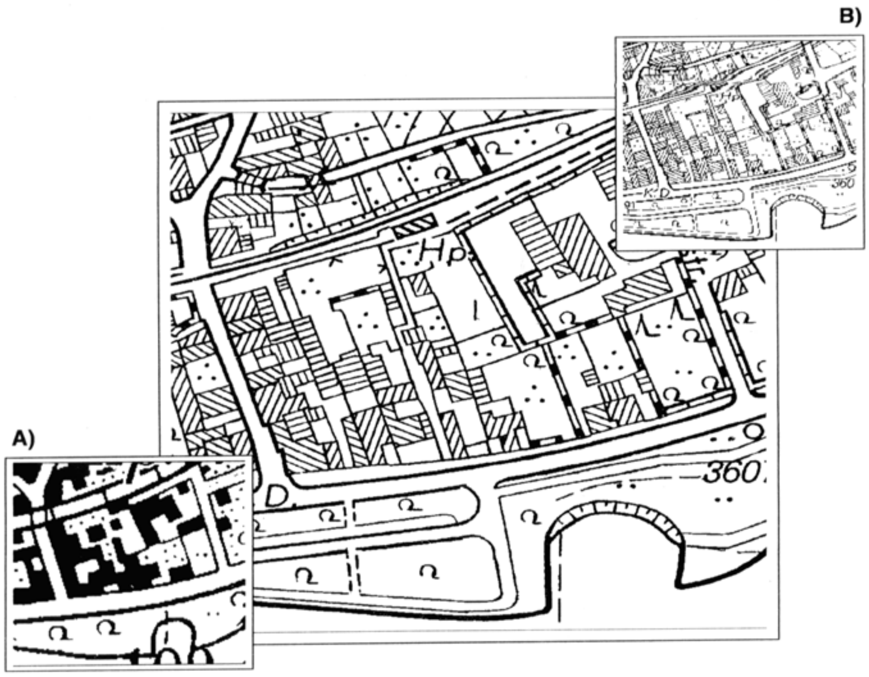
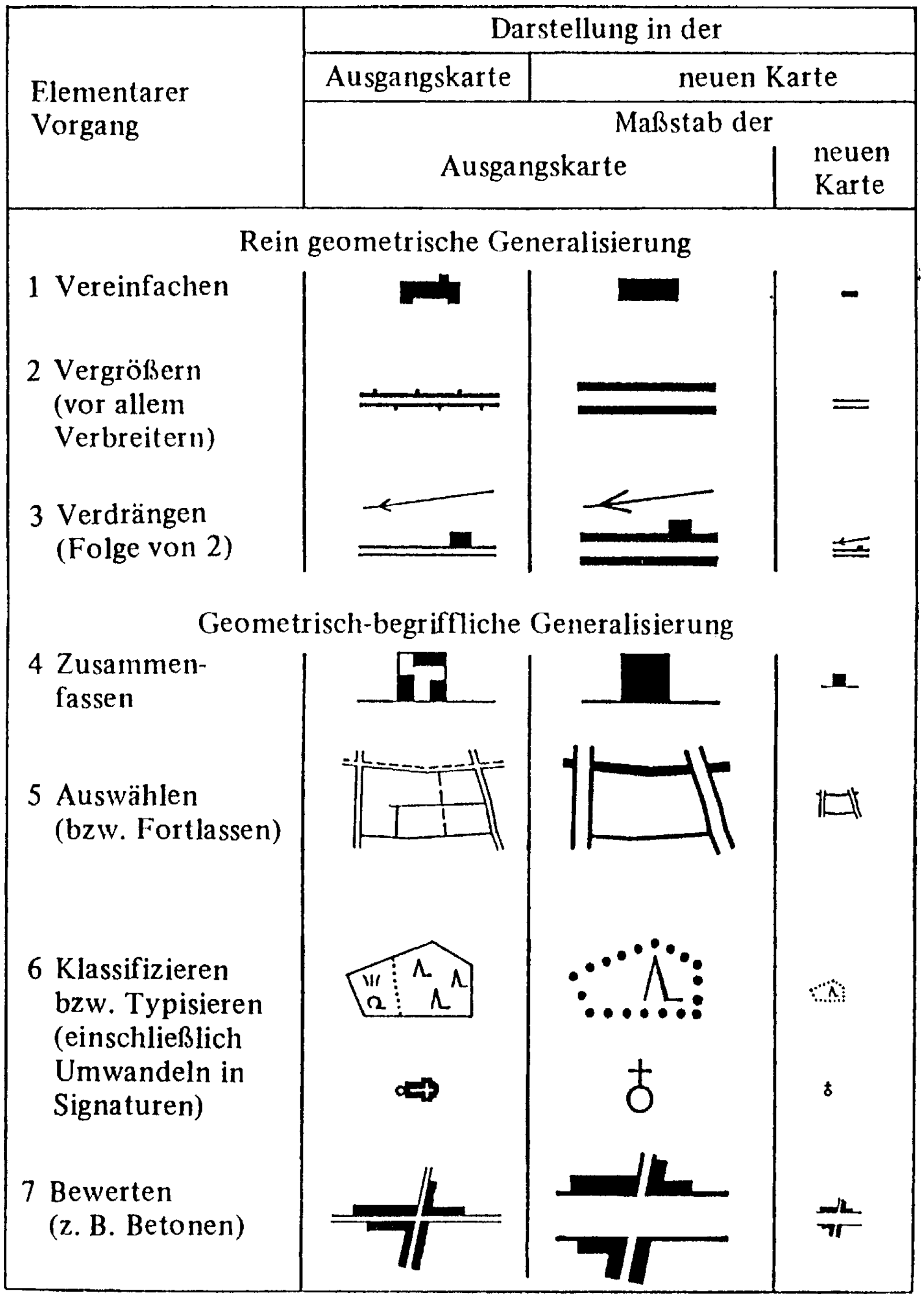


Fig. 2-11: Example of cartographic generalization [Hake02]: In the center a cutout of a 1:500 scale map. Part A shows the same cutout with a 1:25,000 scale. For comparison, part B is a reduction of the central cutout.

mation is usually abstracted in the next stage of the process, to enable the presentation with the limited means of the output device. The generalization procedure can be divided in to three sub-processes (see [BuMa91], [Jung98]):

- **Simplification** includes the reduction of the representation precision of the map elements, the selection of relevant elements and element relationships and the shifting of elements to eliminate unimportant details.
- **Classification** includes the aggregation of adjacent elements in to units and the partitioning of metric sizes in to classes.
- **Improvement** involves the specific addition of new information to the map to simplify interpretation, such as interpolation and reconstruction of elements.

Fig. 2-11 shows a cartographic generalization of a town plan. Fig. 2-12 illustrates the important generalization operators.



Elementarer Vorgang	Darstellung in der		
	Ausgangskarte	neuen Karte	
	Maßstab der		
	Ausgangskarte		neuen Karte
Rein geometrische Generalisierung			
1 Vereinfachen			
2 Vergrößern (vor allem Verbreitern)			
3 Verdrängen (Folge von 2)			
Geometrisch-begriffliche Generalisierung			
4 Zusammenfassen			
5 Auswählen (bzw. Fortlassen)			
6 Klassifizieren bzw. Typisieren (einschließlich Umwandeln in Signaturen)			
7 Bewerten (z. B. Betonen)			

Fig. 2-12: Generalization operators according to [Hake02]

2.6.1 Approaches for Automating the Generalization Process

In the past twenty years the automation of the cartographic generalization processes has played a major role in cartographic research. Automatic generalization is aimed at creating a database in GIS that is independent of scale, which enables the user to generate maps of any scale in real-time. The three most important advantages of such a database are summarized by Müller [Müll91] and [Müll95]:

- Redundant data storage is avoided
- Generation of an output independent of scale, not limited to conventional scales such as 1:10.000 and 1:50.000
- Maintaining consistency between different outputs and consequently, simplified continuation of the database.

A number of different approaches were developed for automating the map generalization process. To date these have only succeeded in automating parts of the generalization process such as labeling [Kres94] or smoothing of lines and building plans [Powi93].

The example of a generalization shown in fig. 2-11 illustrates the problems confronting automatic approaches for generating generalizations. Using a cutout of a 1:5.000 and 1:25.000 scale cartographic presentation of the town Bernkastel-Kues the diagram illustrates the abstractions associated with a change in scale. Small roads are no longer shown, buildings are fused to larger units, the size of the important roads remains unchanged and distort other planar elements. Some objects such as the bay are not depicted true to form but in an idealized fashion.

The following sections give a brief overview of the use of expert systems in cartographic generalization.

Expert systems

The complexity of cartographic generalization has led to a use of knowledge based systems. These base their decision for using different generalization operators on formalized expert knowledge. The approach is based on modules for solving specific generalization subtasks. Important systems in this field are CHANGE [Grün93], STRATÈGE [RuPl97] and MAGE [Bund95].

For example, such a module in CHANGE processes the generalization of buildings [Powi93]. This approach uses an ordinary database to create plan-like presentations that take in to account the target scale. This procedure is divided in to two important steps, *edge simplification* and *fusing* of building plans.

Edge simplification takes a number of building characteristics in to account. This includes the number of corners and sides of a building, distinct building angles as well as building side lengths, maximum extent of the building, the center of gravity of the building and the main direction of the building. In addition basic building forms are distinguished such as T, U and Z forms.

Individual building plans are fused if their distance falls below a specific threshold value. The procedure follows the following basic strategy: both objects will be linked either by filling the gap between their ground plans or alternatively, if the resultant increase in surface area is too much by bridging the gap by moving the smaller building towards the larger one.

Amplified Intelligence

Weibel [Weib91] considers the development of complete cartographic expert systems for automatic generalization to be extremely expensive and not feasible in the middle term. His *amplified intelligence* approach begins at a lower level: he leaves the most important graphic decisions to the user and supports this by providing a number of knowledge based tools that are able to carry out design subtasks. Appropriate modules can solve these subtasks automatically. The user is subsequently presented with a partially generalized map. Any further steps can be carried out interactively by the user.

2.6.2 Fisheye-Lenses

One method for emphasizing is the use of so-called fisheye-lenses, which can link detail and context in the same presentation. The fisheye concept is based on the fact that humans focus on a small area of an image. Details are discerned only in the focal area, while vision is out of focus in the peripheral area. A general procedure for creating fisheye images was suggested by [Furn86]. Each object of an information space is assigned a degree of interest (DOI). The DOI quantifies the current interest of the user in this object.

The DOI contains a static and a dynamic component. The static component defines the general importance of an object and is independent of the interactions by the user. This is why this is also known as a priori importance (API). The dynamic component is derived from the interactions of the user. For this purpose, the distance between each object x and the current user focus (called focal point FP) is computed. The calculation of this distance function is user specific and is a measure of the reduced interest between x and FP. The DOI is obtained by the subtraction of both components:

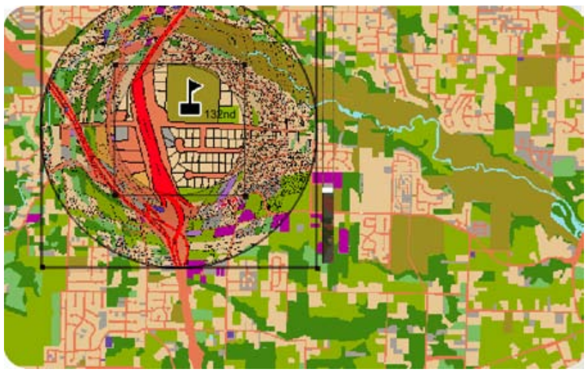
$$DOI = API(x) - dist(x, FP)$$

Once the DOI values have been computed for all objects, the question arises, which presentation technique will be used to emphasize those objects with a high DOI. Noik identifies three groups of object emphasizing techniques [Noik94]:

- Filter: objects with low DOI are omitted from the presentation.
- Distorted: reality is distorted in analogy to fisheye lenses used in photography. Size, position and form of the object are manipulated according to their DOI.
- Adorned: technique for emphasizing or minimizing objects by changing graphical variables such as color, line thickness and transparency.

Interactive fisheye presentations are also found in GIS. For example they are used in map visualizations to view details on a map without losing the context. In the mean time these presentation methods are available in commercial GIS [Plia01].

Rauschenbach has developed a wavelet based method for progressive transmission of raster images which can be viewed interactively through a rectangular fisheye view [RaSc98], [Raus01]. Wavelet compression takes in to account Regions-of- Interest (ROI). These ROIs are presented in great level of detail and are equivalent to the focal area of a fisheye view. The transmission bandwidth can be specified both before and during transmission, either by requesting refinement to define new ROIs or by modifying the level of detail or priority of existing ROIs [Raus00].



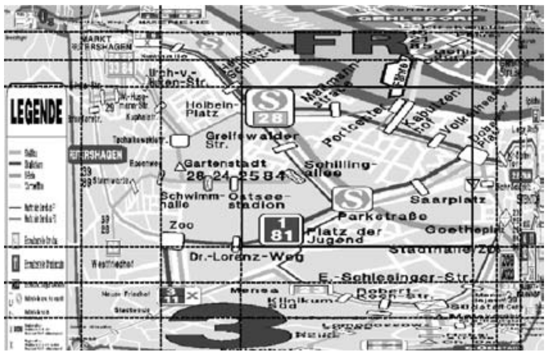


Fig. 2-13: FishEye-View in GIS [Idel01], [Raus01]

Chapter 3

State of the Art: 3D-GIS on the Internet

In this chapter, the state of the technology in terms of graphical abstraction and progressive data transfer in 3D-GIS will be discussed and evaluated. For this purpose, the VRML Topographic Map Generator and GeoVrml approaches, which allow access to a three-dimensional model of the earth's surface within a network-based environment, have been selected. In addition, existing 3D-GIS that focus on storage, analysis and visualization of three-dimensional city models, namely the KARMA VI system and the Vienna Walkthrough system, will be introduced. In addition, the most important proposals of topological 3D data models in 3D-GIS will be compared and discussed. Geological approaches that aim at modeling natural phenomena inside the earth are beyond the scope of this thesis and are thus not analyzed in detail.

Before going into the individual approaches in detail, the requirements should be formulated that 3D-GIS is meant to fulfill and according to which existing systems should be evaluated. The emphasis will be on the use of these systems in the context of three-dimensional city models.

3.1 Internet-based 3D-GIS in an Urban Context

A 3D-GIS in an urban context serves to compile, manage, analyze and present three-dimensional city models. The development of automatic data acquisition processes in an urban environment is a dynamic field of research in the area of photogrammetry. Apart from data transfer problems, the existing data acquisition systems are independent of the other 3D-GIS components. The emphasis in this thesis is not on data acquisition, but on the use of corresponding databases. For the data acquisition, it is assumend that a three-dimensional city model was aquired in an object-oriented approach and a single object has a polygonal geometric shape.

The 3D-GIS database should be available to as many users as possible. The maximal use of a 3D city model can only be ensured when efficient access to and analysis of the database are possible from any location. The necessary technical infrastructure is already available on the Internet and the so-called mobile Internet, and it is likely that the number of available bandwidths will go up substantially, especially on the mobile networks. To make use of all this potential, new possibilities of presenting query results in a network-based environment should be taken advantage of, along with the requirements to be fulfilled by database analysis.

3.1.1 Criteria for the Query-Oriented Data Model

An essential feature of GIS is the efficient analysis of spatial data. Queries within a GIS are characterized by their spatial nature. Many analyses deal with the spatial shape of features entered in the GIS and their relation to one another. Normally, three different types of queries can be distinguished (see [BiFr99] and [Laur93]), which can of course be combined with one another:

- thematic / semantic
- geometric
- topologic

Semantic queries refer to the features' thematic data. The data's spatial reference plays a less important part. Today, the efficient processing of semantic queries is supported very well by standard technologies such as databases.

Geometric queries describe the spatial reference and the concrete spatial extent of a feature. A typical geometric query is the window query, or, more generally, the region query. The user specifies a region and initiates a search for certain features in that region. In the case of a window query, this region is a rectangle with parallel axes; otherwise it is a polygonal region. Spatial indices such as *k-d-Tree* [Bent79], *Region Quadtree* [Same84] and *R-Tree* [Gutt84], [Gree89], [Beck90] have been established to efficiently process geometric queries. [GaGü98] gives an overview and a comparison of the special characteristics of these spatial indices.

Topologic queries do not refer directly to a feature's spatial extent, but to topological relations such as inclusion and neighborhood of two or more features. Typical topologic queries come from route planning. Here the user is interested in the road network and not the exact geometry of a street. A database in which frequently-searched topological relations are explicitly stored is necessary for an efficient processing of these topologic queries. The three main topological properties that are to be given by the model are adjacency, incidence and inclusion [Pig91].

In summary, the following criteria can be used to evaluate the query-oriented data model of a 3D-GIS. In contrast to CAD systems, the focus is on the analysis and efficient processing of the data and not on the modeling itself.

- Can discrete objects be treated as a unit?
- Can zero, one, two, and three-dimensional objects be managed?
- How efficient is the processing of thematic, geometric and topological queries?
- How much memory is needed to display models that are interesting from a practical point of view?
- Can data acquisition models be converted from other systems such as CAD to the query-oriented data model without losses?

While various solutions to some of these requirements have been found, some of them, especially data models for the efficient processing of topological queries in 3D-GIS, are the topics of current research. Various approaches will be described in Section 3.3.

3.1.2 Criteria for the Presentation of Query Results

Apart from the database analysis, the presentation of the query results also plays a crucial role in 3D-GIS. The increasing availability of information networks in private homes and the rising bandwidth in the communications infrastructure requires individual tailoring of the database to the user's needs and available resources. In the past, GIS was used by experts who performed tasks such as creating city maps for a certain target group. The scale and content of these maps were chosen by experts, in this case cartographers, with the target group's needs in mind, and were then interactively designed using a GIS. Today's communication networks allow a much more flexible and thus more individual use of the underlying database. The user's current position and line of vision, as well as his personal needs, can be taken into account in the presentation. A map no longer has to be created for a large target group, but can be generated specifically for one user.

The individual tailoring of a query result requires a database for terrain and features that is independent of scale. The presentation must utilize various scales depending on the query parameters, such as the size of a selected area. A combination of scales can also be useful when it is necessary to show details without losing the overall view. Generalization and graphical abstraction methods are absolutely necessary to generate a suitable presentation from a database that is independent of scale. This graphical abstraction must be carried out according to the query and to the user's individual needs.

Because of the heterogeneous network structure, technical resources must be taken into account when transferring query results. Thus, data compression and progressive transfer of 3D models must be supported by 3D-GIS.

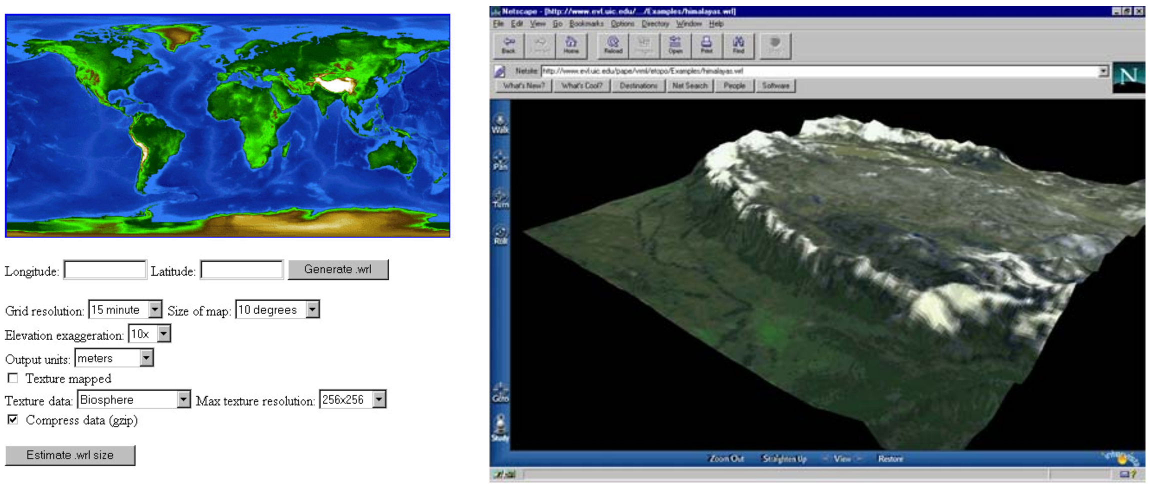
3.2 Selected Systems and Prototypes in the Context of 3D-GIS

3.2.1 VRML Topographic Map Generator

An early example of three-dimensional terrain models on the Internet was created by Pape’s VRML Topographic Map Generator [Pape96]. Using a HTML form, the area, accuracy, the elevation of the terrain and appearance (color, texture) of the desired map are entered by the user. On the server, the corresponding model is then selected from a global database [ETOPO5] using a CGI script. It is converted to VRML and sent to the user.

This system is not a GIS in the true sense, however, because no query or analysis functions are offered besides the window query. The scene, once generated, remains static. Neither data compression nor incremental loading of the terrain model were implemented.

The interaction of this system includes only the generation of a topographical map. When this has been produced, no other interactions are possible except for navigation in the VRML scene.


Longitude: Latitude:

Grid resolution: 15 minute Size of map: 10 degrees

Elevation exaggeration: 10x

Output units: meters

☐ Texture mapped

Texture data: Biosphere Max texture resolution: 256x256

☑ Compress data (gzip)

Fig. 3-1: VRML Topographic Map Generator from [Pape96]

3.2.2 GeoVRML

GeoVRML is an extension of the VRML97 standard for the visualization of geographical three-dimensional models on the Internet. The aim of GeoVRML is to publish and use three-dimensional geographical data simply, dynamically and interactively on the World Wide Web. In contrast to VRML97, GeoVRML supports the direct use of geographical coordinate systems such as Universal Transverse Mercator (UTM) and WGS84. GeoVRML’s focus is on visualization of large areas of the earth’s surface. This requires special mechanisms that can cope with a large data volume. A digital elevation model (DEM) of the earth's surface is generally stored in a grid. The elevation data of each area are kept and managed in a separate file. A file such as this with a simple visualization process for the US Geological Survey results in a geometric model with about 1.4 million polygons [Redd99]. Thus, GeoVRML supports a multiresolution terrain model which allows progressive loading of the model [Redd00]. In addition, the terrain model is subdivided into a hierarchical quadtree structure. On the highest level of this structure there is a low resolution DEM. For each quadrant, appropriately detailed DEM are storeed and then recursively refined. An analogous structure is created for terrain texture, if applicable. The regular hierarchical subdivision of the terrain is also called a *terrain pyramid.* Each individual terrain model in the terrain pyramid is called a tile. Depending on the location and the line of vision, tiles are selected in suitable resolution for DEM and texture. These tiles can be interactively refined and missing data can be loaded. This hierarchical model is shown in fig. 3-2a. A so-called *tree file* manages the quadtree structure for terrain and textures. Terrain model and texture in appropriate resolution are assigned to each tile in the quadtree. A large number of features can be assigned to the tiles with the highest resolution. For example, fig. 3-2b shows an example of an image pyramid for textures and a graph oriented to the line of vision, fig. 3-2c, which was generated from this image pyramid.

Evaluation

GeoVRML is a purely presentational model. No database analysis functions are supported. The hierarchical structure for textures and terrain is suited to generate a multiresolution model for a regular elevation model. The model is very usable when features are only visualized at the most accurate level of resolution. An abstraction of features is not possible, so that its use for e.g. creating a digital city model with a large number of buildings does not lead to a substantial reduction in the amount of data.

3.2.3 Karma VI

GIS and VR technologies are integrated in the Karma VI system, which was developed at Delft University of Technology [Verb99]. Various existing systems were combined to allow the design, development and presentation of infrastructure maps covering large areas in the Netherlands. From this application, the following criteria for the system were derived:

- **Orientation**: In this first phase, the planning task is analyzed based on the existing situation and initial ideas are sketched. Usually, maps with a scale of 1:10,000 to 1:100,000 are used. In this orientation phase, the standard 2D-GIS function is sufficient. The underlying data and maps are two-dimensional.

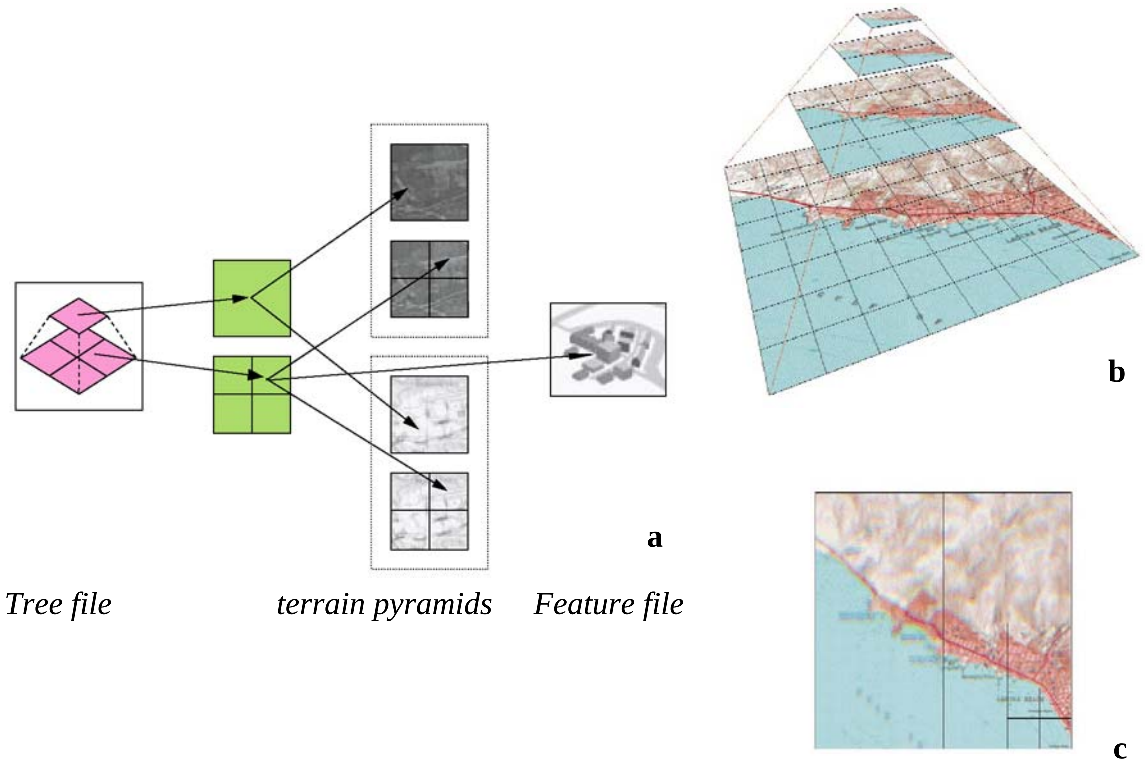


Fig. 3-2: Hierarchical Multiresolution Model for DHM from [Redd99]

- **Design and Modeling:** In the design phase, two or three promising possibilities from the orientation phase are planned in detail. To evaluate the design, a simple 3D model is created, so that the relations between the objects and the situation as a whole can be judged more easily. In this phase, analysis and interaction functions, including the possibility to modify objects, are important. The 3D models of individual buildings are derived from the GIS sketch and thematic data, such as the number of floors, or are modeled manually in a CAD system. The CAD models are assigned to the GIS features; a CAD model often contains several features.
- **Presentation:** In this phase, the designs are presented to those responsible for making decisions. For these purposes, the design should be communicated comprehensibly, which is why the visual presentation should be as realistic as possible. The objects used in the presentation are modeled using CAD tools; a VR system is used for the presentation. The presentation can be made on typical VR technology devices such as Virtual Table, CAVE or Head Mounted Displays.

Fig. 3-3 shows the architecture of the Karma system. Access to GIS data is given by spatial access methods; for these purposes, the commercial software program SDE from the company ESRI is used [SDE97]. The World Toolkit from Sense8 [Sens98] is used as a VR environment. In Karma VI, the models for the three different phases are used: the

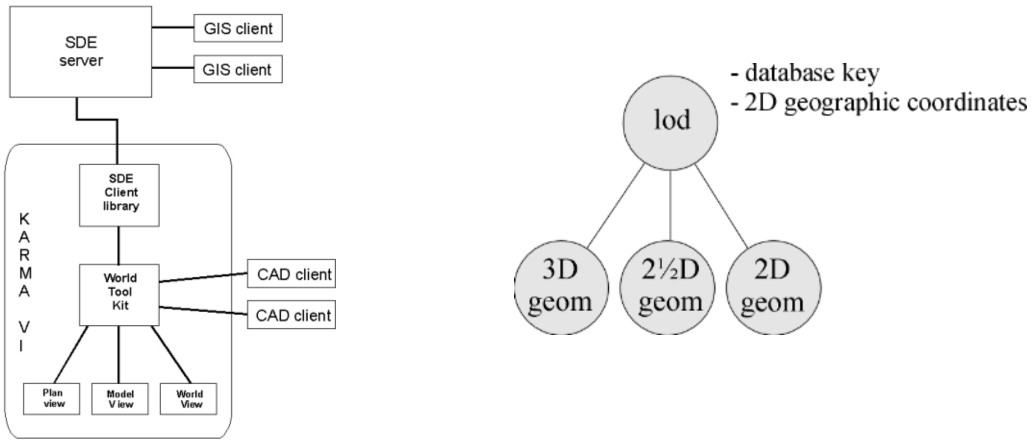


Fig. 3-3: Architecture and View Concept in Karma VI from [Verb99]

two-dimensional plan view, the model view that shows a 2.5D model with only a small amount of detail, and the world view, which approaches visual reality. The graphs in the three different views are object-oriented. For each object, there are three levels of detail which correspond to the views. Fig. 3-4 shows these views using the example of the Delft Institute's campus.

Evaluation

The Karma system makes use of the strengths of various standard technologies and integrates these into a 3D-GIS. However, the scope of geometric queries is limited to ground-plan geometry by the use of SDE. It is not a 3D-GIS in the strict sense, but a 2D-GIS with 3D visualization components. Corresponding analysis functions for three-dimensional geometries are missing. The focus of the visualization is on VR. Use of the maps on the Internet is not explicitly supported. A web-based interface is, therefore, not available, and neither is a function for the incremental loading of the model. Although Verbree believes that there is a need for generalization technologies in addition to the three views [Verb99], no generalization is supported in Karma VI .

3.2.4 Vienna Walkthrough

The Vienna Walkthrough System was developed by Kofler as part of his dissertation [Kofl98]. The aim of the system is an interactive exploration of a large 3D-GIS database. The underlying database is a 3D model of Vienna, of which an area of 150 km^2 with about 20,000 buildings is modeled. For an efficient search for buildings in the visible area, a so-called perspective query, an R-tree is used as a spatial index. There are three levels of detail per building, which are used depending on the distance between the viewer and the graph. A building geometry with little detail, a bounding box and a single point are used as levels of detail. As shown in fig. 3-5a, the view pyramid is divided into three

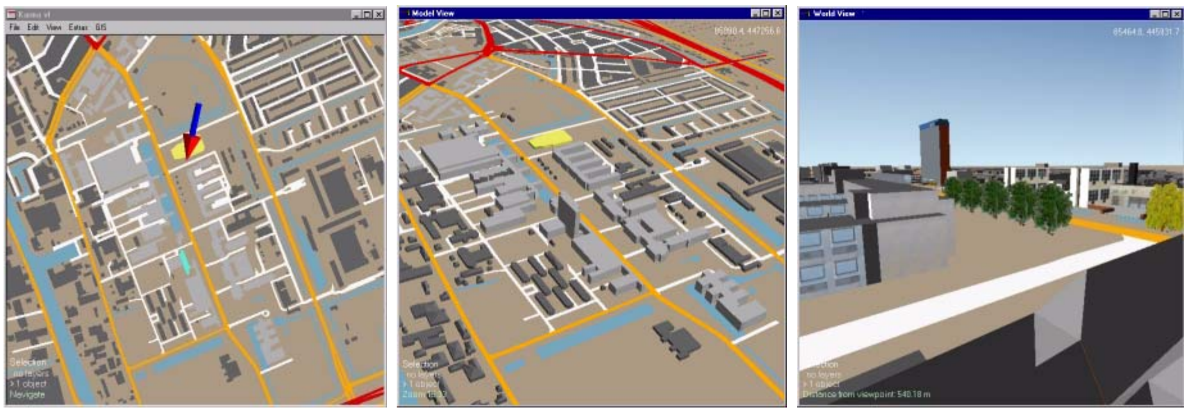


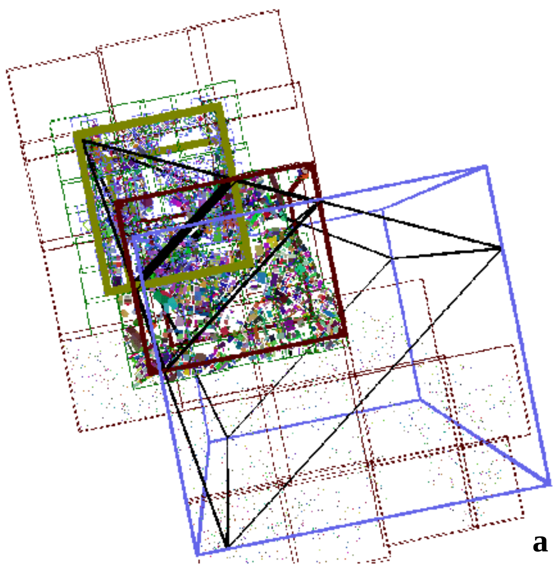
Fig. 3-4: Plan, Model und World View in Karma VI [Verb98]

segments for the selection of suitable levels of detail. This division can be adapted to available resources in order to guarantee a frame rate of 10 fps. If the user provides no input, the model is refined so that more details can be seen in the background, as shown in fig. 3-5b and c. However, all levels of detail are initially loaded into the main memory when the system is started. As no progressive reloading of the scene is possible, the use of a database system to manage the 3D building geometry in the Vienna Walkthrough System was not implemented.

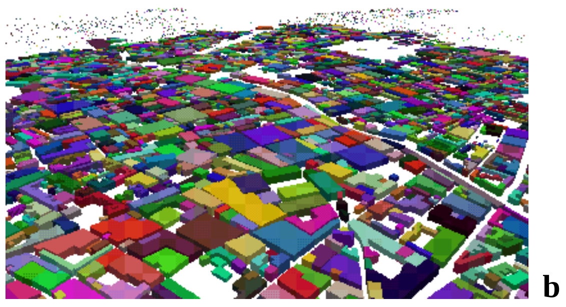
Evaluation

With its capability to produce progressive renderings, Kofler's procedure is well-suited to the resource-adaptive visualization of a large-area, three-dimensional city model. However, a progressive reloading of the detail levels is not possible. Zlatanova extended Kofler's approach by using the R-tree as a hierarchical level-of-detail structure; the individual buildings in the inner node of the R-tree are aggregated using a bounding box [Zlat00]. This procedure is similar to that developed by Coors and Schulz and also allows a progressive reloading of refinements [CoSc99].

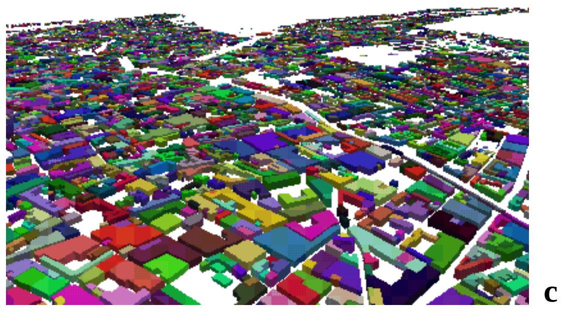
A database analysis function is not supported by Kofler, so that the system is a visualization system rather than a true 3D-GIS. A connection to one of the query-oriented data models discussed in the next chapter is possible, however, through the use of a spatial index for the selection of detail levels.



Perspective Query with three LoD areas in the view pyramid



Initial Exploration Model



Progressive Refinement

Fig. 3-5: Perspective query and progressive rendering from [Kolf98]

3.3 Query-Oriented Data Models in 3D GIS

3.3.1 Molenaar's 3D-FDS

Molenaar's *Single Vector Value Map (svvm)* is a formal model that describes topological relations of two-dimensional geographical objects [Mole98]. This concept is the basis of many established GIS data models in public administration. An example is the geometric model of the *Amtliches Topographisch-Kartographisches Informationssystem* (ATKIS) in Germany [AdV99].

The basic concepts of the *svvm* are

- object-oriented management of thematic and geometric data,
- vector model for the modeling of the geometry, and
- explicit management of the topology using an incidence relation between points, lines and surfaces and point-surface inclusion.

A fundamental assumption of the *svvm* is a complete subdivision of the space by all objects. This implies that not only relevant objects must be modeled; "empty space" must also be considered an object and taken into account. The second fundamental feature is the *single value map*; that is, a geometric primitive can only represent an elementary object of the same dimension. A detailed discussion and a formal definition of the *svvm* can be found in [Mole98].

An extension of this conceptual data structure to the third dimension was suggested as *3D Formal Data Structure (3D-FDS)* in [Mole92] and was implemented as a relational model in [Rikk94] and [Pilo96]; it was analyzed and tested with regard to spatial queries. Fig. 3-6 shows an overview of this 3D-FDS. In essence, the 3D-FDS is a boundary representation of solids, expanded by 0D, 1D, and 2D primitives to represent point-like, line-like and surface-like elements. The basic topological elements *node*, *arc*, *edge*, and *face* are available as construction elements for geometric primitives. The edges serve to orient the faces. Conceptually, any 2-manifold can be modeled, but only a polyhedron model was implemented. There is no detailed specification for the storage of the geometry of curved surfaces.

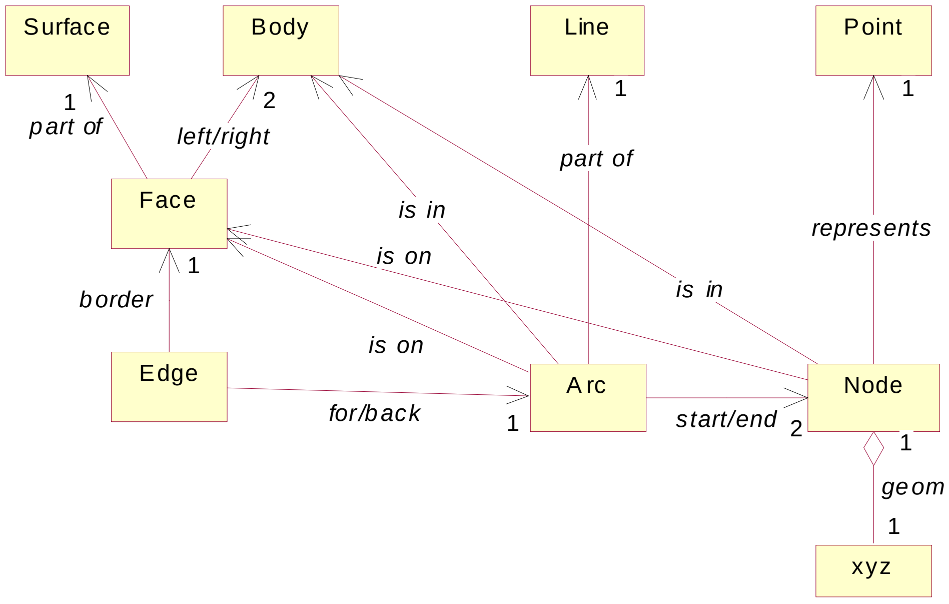


Fig. 3-6: Conceptual data model 3D-FDS by Molenaar [Mole92]

The topological primitives are linked to thematic data by the introduction of features using a classification. This feature concept permits a discrete view of individual objects. Using the *svvm* concept, a feature can be assigned to only one topological primitive. Thus, an explicit distinction between point-like, line-like, surface-like, and volume-like features is absolutely necessary. Accordingly, a feature is represented by a *point*, a *line*, a *surface* or a *body* primitive.

Evaluation

Topological relations between the primitives can be derived from the data model. Because of the oriented relations, queries regarding solids can only be answered in terms of the boundary faces. For example, the boundary representation of a feature that is represented by a body *B* is given by:

$$\partial B = \{FaceF | ((F.left = B \wedge F.right \neq B) \vee (F.right = B \wedge F.left \neq B))\}$$

Due to the explicit treatment of arcs and edges, the data volume necessary for the storage of polygonal solids is very large. For a convex polyhedron the number of the number of faces F and nodes V together is approximately as large as the number of arcs E, or more precisely |F| + |V| = |E| + 2 [Hen79]. Thus, the number of arc elements in the *3D-FDS* is approximately as large as the sum of the face elements and node elements. In a solid model, each arc has two edges, so that the number of edge elements is twice as high as the number of arc elements.

Zlatanova and Tempfli have developed a web-based visualization of a database based on the *3D-FDS* [ZlTe98]. As the 3D-FDS is an boundary representation of polyhedrons, a direct visualization of the model is possible. Nevertheless, the following problems have been reported, among others:

- Lack of links to face per solid object and thus reduced efficiency in viewing individual objects. All faces must be checked each time.
- There are sometimes problems in the use of textures for individual faces. In some cases, many textures must be entered for a face.
- The implicit description of hole slows the generation process of the visualization.

If a CSG or BRep model is generated in the course of data acquisition, the data can be transferred to the 3D-FDS. The conversion of cell models is possible but generally leads to losses.

3.3.2 3D GIS Data Model by Flick

In [Flic96] and [Flic98], Flick suggests an extension of the 3D-FDS. In this data model, four basic objects with 0, 1, 2, and 3 cells are used to manage topological relations and also the geometry and visualization. Incidence ("bounds") between *n* and *(n+1)* cells and inclusion ("is-in" in 3-cells, "is-on" in 2-cells) are explicitly stored bidirectionally. The four basis objects are subject to restrictions that are less limiting than in the 3D-FDS concept:

- *n*-cells cannot intersect themselves or other *n*-cells (*n*=1, 2, 3).
- 1-cells cannot intersect with 2-cells.
- Neither 0-cells nor 1-cells can ever have a "is-in" and "is-on" relation at the same time.

Flick rejects the *svvm* concept and implements the assignment of geometric and topological primitives and concrete geo-objects using association. Using a so-called view concept, various visualization data can be assigned to a geo-object independent of the geometric data. This flexible concept is very well suited to the requirements of a 3D geoinformation system and to the requirements of a visualization system as well. To give an example, the view concept can be used to implement an LoD structure [CoFl98].

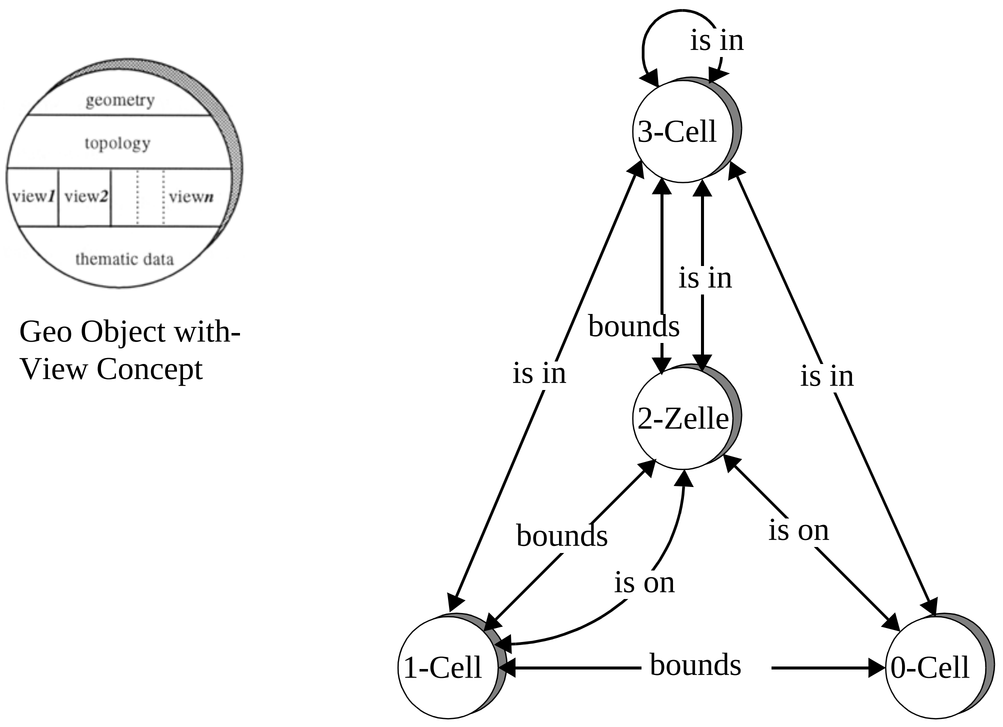


Fig. 3-7: 3D-GIS data model from Flick [Flic98]

Discrete objects of different dimensions can be handled discretely in the data model. The designation of the four base objects as cells is misleading, however. For example, a cell complex allows no 0-cells within a 2-cell; rather, the geometry model is a geometric complex [RoCo89]. The conceptual model was implemented using a hybrid modeler from the CAD system as a basis [Flic98].

Evaluation

The explicit storage of all bidirectional relations permits a very efficient and flexible processing of topological queries. Nevertheless, in this approach, many relations that can be derived from other relations are stored several times. Substantially more data volume is necessary. A database is not used to store the geometry. Mapping the 3D data model to a database cannot be simply performed, due to the large number of stored relations and the use of a hybrid modeler from the CAD system.

A web-based visualization of the query result was implemented by Coors [Coor97]. The boundary representation allows the efficient processing of the visualization. Various levels of detail of individual features are supported by the view concept. However, the preservation of the object structure at the cell level, which is desirable for purposes of analysis and interaction, creates a large data volume. This has disadvantages for the use of the model on the Internet.

3.3.3 De la Losa’s 3D Topological Model

An alternative approach to a database for a 3D-GIS, developed by de la Losa and Cervelle [LoCe99], expands the topological model of a map by David [Davi91] into the third dimension. Fig. 3-8 shows an overview of the complete 3D-GIS data model. A detailed description can be found in [Losa98] and [LoCe99]. In the model, a distinction is made between *arcs* and *oriented arcs*, as well as between *faces* and *oriented faces*. Because of the explicit linkage of the face to the oriented arc, various operators such as "*Next Couple with the same Face*" (NCF), "*Next Couple with the same oriented Arc*" (NCA), and "*inverses couple*" (INV) can be defined as part of this composite element. These operators form the core of the topological query processing. Holes and cavities are explicitly stored, so substantially more information can be derived directly at the model level. The possible topological relations of the primitives node, arc, face and volume are examined in detail in [LoCe99] using Egenhofer’s *9-Intersection* model [EgHe90]. The geometric implementation of the topological model is carried out using 0-, 1-, and 2-simplexes, that is, a triangulated boundary representation.

Evaluation

The data model was created in an object-oriented database. Simple topological and geometric queries such as the distance between two objects can be answered in the prototypical implementation. The search result is visualized using VRML. Unfortunately, no research results regarding the data volume and effort necessary to generate the visualization have so far been published. As regards visualization, there is no possibility in the data

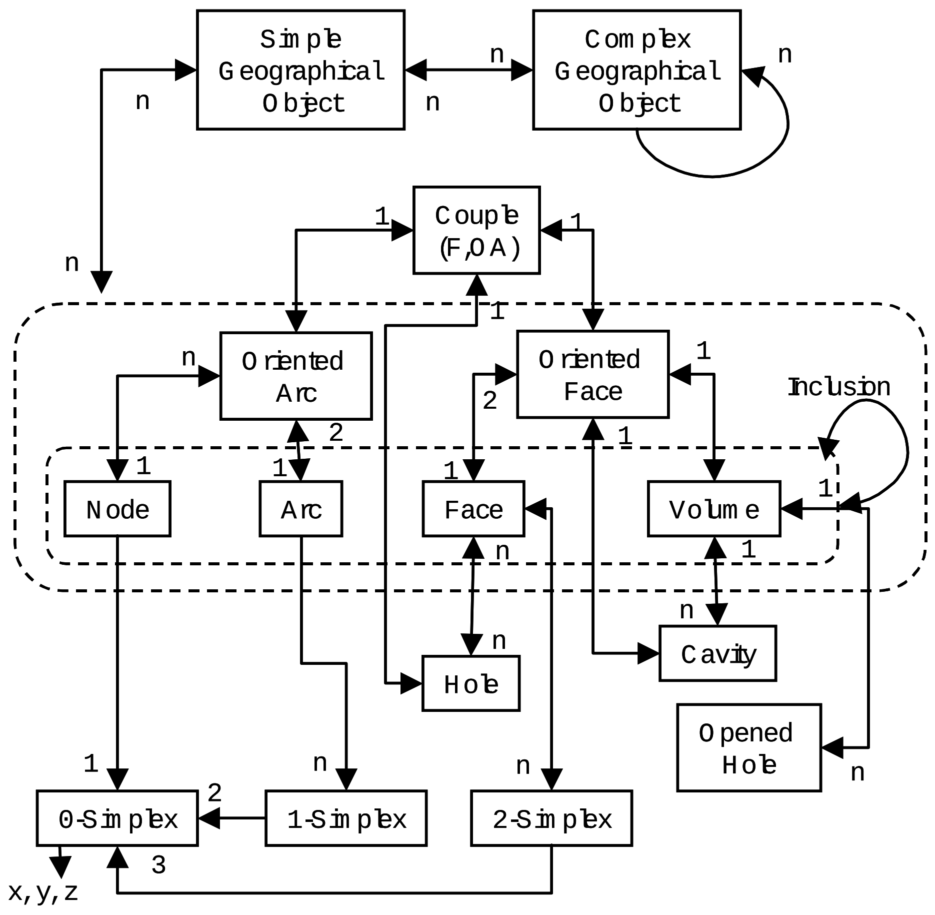


Fig. 3-8: de la Losa's 3D-GIS data model [LoCe99]

model to assign textures to faces. It is also unclear how holes can be identified from a given boundary representation of a feature. An example would be a torus which, according to de la Losa and Cervelle's model, has a hole. Using an boundary representation, however, the hole cannot be simply located.

3.3.4 Simplified Spatial Schema

Zlatanova and Tempfli have suggested the *Simplified Spatial Schema* (SSS), a data model based on Molenaar's 3D-FDS. However, they do not explicitly store arcs nor edges [ZlTe98]. Surfaces and lines are represented by vertices only. This model is limited to an boundary representation of polyhedrons. Due to this limitation, however, the number of construction elements is reduced substantially. In [Zlat00], Zlatanova proves that this topological model is complete; that is, that all topological relations between two simple point-like, line-like, surface-like or volume-like primitives used by Egenhofers *9-Intersection* model are distinguishable in SSS. The topological data model is shown in fig. 3-9.

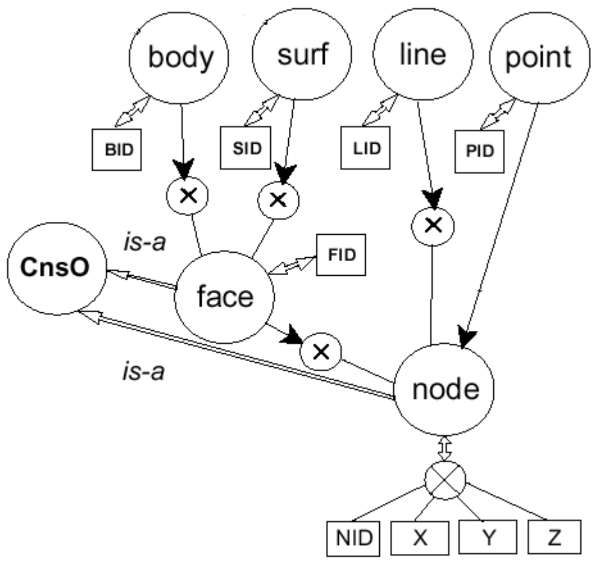


Fig. 3-9: Topological model by Zlatanova [Zlat00]

In SSS, both thematic data and a geometric appearance, as well as a geometric behavior, are assigned to a spatial object. The appearance, like the visualization system, is used to describe color, texture and material properties. The behavior defines possible interactions with the object in question, such as the opening of a door, and triggers that activate this behavior.

Evaluation

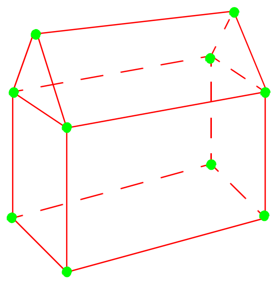
The SSS permits a very efficient storage of polygonal models. Because arcs are implicitly stored, however, only straight lines can be used for line-features. This is often not sufficient for the modeling of streets and roads, for example. Topological relations are completely supported in SSS.

A web-based visualization can be generated directly from the SSS, as only convex polygons are permitted as faces and face definition concepts using nodes are also found in visualization formats such as VRML. According to Zlatanova, the visualization of a database of 1600 buildings with approximately 20,000 polygons can be generated in under a minute [ZlVe00].

Behavior can also be used in SSS to represent objects differently according to their degree of immersion. This corresponds to a level of detail concept; however, only the selection of a different form of representation is defined in the behavior. Zlatanova herself views this method as being only of limited use, however, and refers to automatic generalization as one of the future research topics in the 3D-GIS field.

3.3.5 Comparison of Existing Query-Oriented Models

In order to evaluate the 3D-GIS models described here, a very simple model of a building was created using the different models, and the data volume was estimated. Both the number of objects and the number of links between the objects were taken into account. The example model and the results are shown in fig. 3-10.



Data Model	*Coordinates*	*Objects*	*Links*
3D-FDS	*30*	*71*	*120*
Flick, concept	*30*	*71*	*153*
Flick, ACIS	*30*	*152*	*~300*
de la Losa	*30*	*164*	*227*
SSS	*30*	*20*	*43*

Fig. 3-10: Comparison of data volume needed for the model of a building

In the 3D-FDS, the sample building is modeled with 10 nodes, 17 arcs, 34 edges, 9 faces and 1 body. Three coordinates were stored per node. For each arc the beginning and ending nodes are stored, i.e. 34 links total. For an edge, the corresponding arc is handled in the direction of circulation; there is also a link to the face which borders on the edge in question. For 34 edges, there are 68 links. Each face is linked to exactly two bodies; an element AIR is added as a default body. Thus, for each face 18 links are stored. Accordingly, a total of 30 coordinates, 71 objects and 120 links are needed to describe the model.

Flick’s data model is analogous to the 3D-FDS in that it requires 10 0-cells, 17 1-cells, 9 2-cells and 1 3-cell. The bounding relations between the cells are stored bidirectionally, however. That means there are 9 links for the 3-cells to the bordering 2-cells, for each 2-cell a link to the bounded 3-cell and 3 or 4 links to the bordering 1-cells. For the 2-cells there are thus 43 links in all. For 1-cells there is a link to each of the two bordering 0-cells and to each of the two bounded 2-cells. Thus for 1-cells 68 links are stored. A 0-cell borders on three or four 1-cells; here 34 links are stored. Therefore, to store the sample model, theoretically 10 coordinates, 71 objects and 153 links are used. To implement the data model using the hybrid modeler ACIS as a basis, the 81construction elements used there, as well as their relations to each other, must be added.

In de la Lola und Cervelle’s data model, only triangles can be used as two-dimensional construction elements. Thus, the model must first be triangulated, which leads to 10 0-simplexes, 24 1-simplexes and 16 2-simplexes. A 1-simplex is linked to 2 0-simplexes and a 2-simplex is linked to 3 0-simplexes. Thus 50 objects and 96 links are stored for the construction elements. The topological model is generated using these simplexes. Ten nodes with links to the corresponding 0-simplex, 17 arcs with links to a corresponding 1-simplex and 9 faces with links to one or two 2-simplexes and a volume are stored. In addition, there are 34 oriented arcs with bidirectional relations to the corresponding arcs and nodes and 9 oriented faces with a corresponding link to faces. Also, one couple object is formed for each oriented face and bordering oriented arc. In all, 114 objects and 181 links are created in addition to those listed above.

In Zlatanova’s Simplified Spatial Schema (SSS), arcs are not stored explicitly. Thus the model is limited to polygonal models, but the data volume is substantially smaller. For the sample model, 10 nodes, 9 faces and 1 body are stored. There are links from each face to the bordering nodes. The body object manages the bordering surfaces. Thus a total of 43 links are required.

3.4 Summary and Unsolved Problems

In this chapter, different 3D-GIS systems and concepts were described. No attempt has been made to conduct a complete survey; however, systems that represent a typical class of problem-solving approaches in 3D-GIS were selected. The integration of 2D-GIS and interactive three-dimensional visualization in the Karma VI system is under development in similar form by commercial GIS manufacturers such as ESRI. Systems based on this approach have similar features, especially the fact that only a two-dimensional query-oriented data model is available for the analysis of the database.

In Table 3-1, the properties necessary in a 3D-GIS are compared to the possibilities offered by existing solutions. It is clear that the focal point of current research is the area of suitable query-oriented data models. A corresponding data model is a prerequisite for reaching the aim of database use in a heterogeneous network environment; on its own, however, it is insufficient to achieve this.

There is great need for new approaches to the user-specific visual presentation of query results and the related transfer of the presentation ina network. The presentation of results must satisfy the user’s need for information, while at the same time taking into account the technical resources, such as bandwidth and power of the output device, which are available. In addition, techniques for automatic graphical abstraction must be developed that are capable of presenting the essential elements of a query result in a suitable way. Generalization techniques from cartography, which are used to emphasize important map features, can be applied to three-dimensional geometries. The user must be able to explore the database interactively and, at the same time, formulate new queries on the basis of the situation as it is presented.

In a network environment, quick feedback to a user’s query is crucial. Most systems do not have suitable mechanisms for progressive data transfer needed for this purpose. Despite continual improvements in telecommunications technology, methods of compressing the generated presentation should be supported so that the available bandwidth is used as efficiently as possible.

		VRML Map Gen.	*Geo VRML*	*Karma VI*	*Vienna Walk*	*3D-FDS*	*Flick*	*de la Losa*	*Zlat. (SSS)*
Query DM	*3D Topology*	--	--	*o(2D)*	--	+	+	+	+
	Database	--	--	--	--	+	--	+	+
	Multires. DHM	--	+	--	--	--	+	--	--
	Multires. Feature	--	--	*3 LoD*	*3 LoD*	--	*LoD*	--	*LoD*
Autom. Graph. Abstract.	*Simplification*	--	--	--	*o*	--	*DHM*	--	--
	Aggregation	--	*DHM*	--	*(+)*	--	--	--	+
	Query/User specific	--	--	--	--	--	--	--	--
Network	*Resource adapt. Visualization*	--	+	--	+	--	--	--	--
	Progressive Data transfer	--	+	--	*(+)*	--	--	--	+
	Compression	--	--	--	--	--	--	--	--

Table 3-1: Evaluation of existing 3D-GIS systems

In the following chapters, a 3D-GIS concept that meets these new challenges will be elaborated. To this end, a query-oriented data model based has been developed, and serves as the basis for a user-specific graphic abstraction of query results. The generation of this graphic abstraction and the progressive transfer and compression of the presentation will be discussed in detail in the following chapters.

Chapter 4

Concept of a 3D Geodata Server

This chapter introduces the general concept of a 3D geodata server for administration and supply of space-oriented three-dimensional data in a heterogeneous network environment. Due to the application field of this 3D geodata server, the main focus is on wide-area, three-dimensional city models. The chapter starts with a detailed requirement analysis of a 3D geodata server. On the basis of these requirements, a concept for the realisation of a 3D geodata server will be proposed. In chapter 5 the main components within the general concept are explained and specified in detail. A prototype implementation of the general concept focused on the core components can be found in chapter 6.

Collecting and maintaining the database is not a part of the 3D geodata server. It is assumed that all data are provided by a specialised third-party system. Corresponding systems for (semi-)automatic collection of three-dimensional city models do already exist. The integration of collection system and 3D geodata server meets all criteria for a 3d GIS in this respect.

4.1 Requirement analysis

In this section, the requirements for a 3D geodata server are analysed. The application field is a wide-area, three-dimensional city model. This 3D city model is provided by the geodata server for a large number of users. This analysis will distinguish between the requirements for the database and the functional requirements of the 3D geodata server.

The requirements for the database are application specific. Essential criteria are:

- Level of detail,
- Accuracy and
- Up-to-dateness of the underlying database.

A geodata server has to meet the functional criteria independently from the underlying database. The specific criteria are

- efficient processing of space-oriented requests,
- interoperability with other geodata servers,
- online database access, especially efficient data transmission
- presentation of the query results.

4.1.1 Requirements for a 3D city model database

In this section begins with a description of the critera for the underlying database. As a result of intensive research, (semi-) automatic collection of urban data has advanced to such an extent in the last years that wide-area 3D city models are readily available and affordable. Some German cities like Hamburg and Berlin are currently building such 3D models. Analyses and user surveys from concrete projects were used to determine the aforementioned criteria for a 3D city model database.

Level of detail

The required level of detail for the three-dimensional city model depends on the application. A 3D model with 1:1,000 scale and standard roof shapes is enough for most costumers. The constumer must be able to expand this model using his own database. Special applications require a significant higher level of detail for certain objects. Due to these application-dependent differences for the required level of detail, a suitable data model for 3D city models must support a multiple object representation. Therefore, there must be an opportunity to represent a building by several models that differ in the level of detail (fig. 4-1). Multiple representations are well known as level-of-detail concepts in computer graphics .

Accuracy

The accuracy rule is: the more exact, the better. Qualitative accuracy may be adequate for a pure visualisation exercise (e.g. real estates), but for planning and simulation purposes quantitative accuracies must be guaranteed. For example, for planning a cell phone net, a position accuracy of 1.0m and an altitude accuracy of 0.5m for buildings smaller than 5m and 2.0m for higher buildings is required [Ruff00].

Fig. 4-2 contrasts both requirements, level of detail and level of accuracy in a chart. As an example, some applications are included and placed in the chart corresponding to their requirements on level of detail and level of accuracy. The coloured area in the chart highlights the level of detail and level of accuracy area, which can be provided by the city in a first realisation stage of the 3D information infrastructure. In this case, it is a simple block model with building heights that were calculated from the number of storeys. This block model can also be used to transform or expand it for other applications with higher level of detail or higher accuracy requirements. These applications are costumer specific extensions. For city marketing and tourism applications, special buildings will be more de-

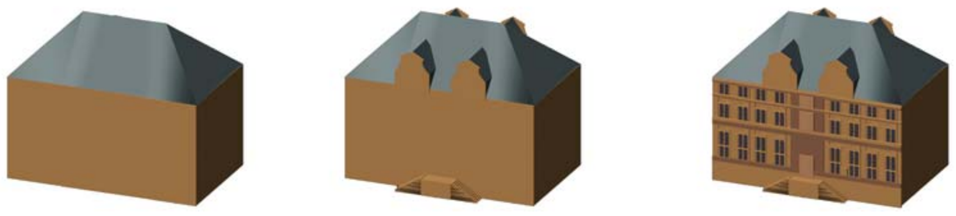


Fig. 4-1: different level of details of the same building

tailed by providing textures, for example. Models for architects for planning purposes have to be more detailed and more accurate than a simple block model.

Up-to-dateness

For all applications, the data must be as up-to-date as possible. Most of the interviewed potential users want to have an model up-to-dateness between 1 and 2 years. This corresponds with experiences from cell phone network planning, which aims at a 2-year periodic update [Ruff00]. The ALK represents the current state-of-law. The up-to-dateness derived from this model is known as legal up-to-dateness. The physical reality corresponds to the buildings at a specific time. There is a difference between the legal and the physical up-to-dateness.

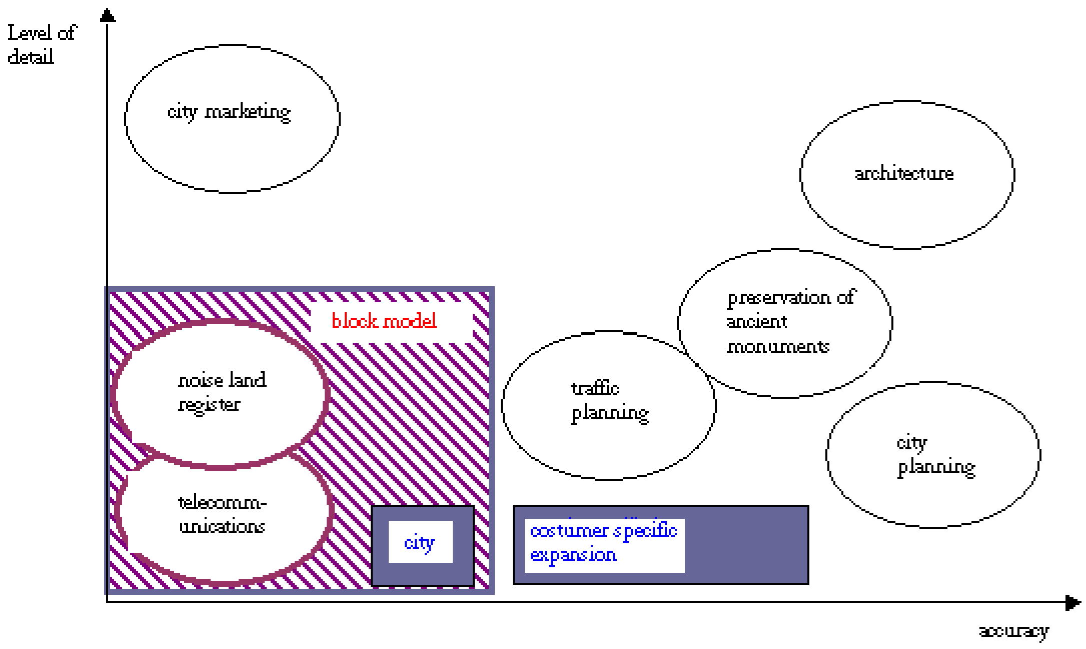


Fig. 4-2: Accuracy and level of detail (DVV 1995, extended)

4.1.2 Functional requirements for a 3D geodata server

Interoperability

There is no advantage in providing isolated special databases such as 3D city models outside of special applications. A modern geodata infrastructure requires the integration of all databases. This ensures that different databases can be used mutually by many applications. The integration of databases does not imply a physical integration of data. To be precise the integration of data takes place on the basis of interoperable geodata servers.

Hence, interoperability with other data sources must be guaranteed when concieving a 3D geodata server. Only this makes it possible to connect 3D city models and subject specific data sources such as costumer files online.

Online data access

A 3D geodata server should provide access to the underlying database even in a heterogeneous network, thus ensuring highest possible data utilisation. Applications such as city marketing and location based services use 3D geodata servers via the world wide web and mobile end devices. Usually, only a small slice of the model is accessed here.

The transmission of complex 3D models on the internet, requires consideration of the comparatively small bandwith. This means that particular features are essential for data transfer (see [Ross98]):

- **Model access:** A rough model must be presented after a maximum of 10 seconds. This model enables the user to decide on the next steps required for navigating the model and accomplishing his tasks.
- **Model navigation:** A screen refresh rate of 10 Hz must be guaranteed when navigating the model or the scene. Lower refresh rates mean that moves in the model are discontinuous. Jerky movements caused by low refresh rates exceed the user's frustration level. When navigating a scene, the presentation quality must be sufficient for estimating the relative position and to facilitate orientation during navigation.
- **Model Refinement:** While the user focuses a particular area in the model, the system refines the objects in the spectator's view. The detailed objects are the decision basis for the user's further actions.

Space-oriented queries

A 3D geo data server must be able to process different user queries using the underlying database. These user queries can be split up in three main categories:

- area-oriented queries,
- topological queries and

- thematic queries.

Area-oriented queries are used to select objects in a specified region. A 3D geo data server must support two-dimensional queries such as *point, intersection* and *containment query* [GaGü98], as well as three-dimensional queries such as *viewcone query.*

A *point query* is a tool for selecting an object by the user. For example, the user specifies a point P on the screen using a mouse. The point query now selects all objects touched or sliced by the ray defined through the user's visual focus and P.

Intersection and *containment query* select all objects located in an area specified by the user. The *intersection query* selects all objects which partially or completely belong to the area, while the *containment query* only refers to objects, which completely belong to the area. The specified area is a two- or three-dimensional region. Using a two-dimensional query, a three-dimensional presentation can be created from an area that can be chosen on a map or from an aerial view. The 3D geo data server uses the intersection query to select all objects belonging to this two-dimensional region.

The *viewcone query* is a special type of *intersection query*, where the queried region is confined by a cone. This cone is defined by the *view point*, a direction and an apex angle. The viewcone query is used for visibility analysis and for visualisation.

Contrary to area-oriented requests, topological request do not refer to the real position geometry, but to topological relations such as neighbourhood and inclusion. Zlatonova collected topological requests in the context of a user survey [Zlat00]. Typical requests were:

- building next to building (neighbourhood)
- building in street (neighbourhood)
- door, window in wall (inclusion)
- all neighbouring lots (neighbourhood)
- building on lot (neighbourhood)
- tree on lot (neighbourhood / inclusion)

In order to decrease the output data volume, all space-oriented queries can be combined with thematic queries. For example, an *intersection query* can be limited to select buildings only.

Presentation

In a space-oriented query, the 3D geodata server is required to create a meaningful, interactive, three-dimensional presentation of the selected data. The presentation of a query result by the 3D geodata server, is based on similar principles as for the creation of a two-dimensional map. Three-dimensional visualisation is model-characteristic, i.e. the representations of real world objects should be located in the right place and should have the exact shape (geometry). Furthermore, as a communication medium, visualisation requires an adequate level of readability. Hence the basic principles of cartography such as exact geometrical presentation, emphasis of characteristics and adequate graphical distinction [Hake02] must be taken in to account in three-dimensional visualisation.

In contrast to the carthographic generalisation an interactive, a three-dimensional visualisation can be further refined. This is also possible in a heterogeneous network environment. The special requirements for online access to the network have already be mentioned.

Furthermore, the user's intention, which is expressed by the concrete query, has to be considered when generating a presentation. Therefore, a photorealistic presentation of the model is not the intention of the request in every case. In many cases, a photorealistic presentation can distract the user's focus from the presentation's essential elements. An abstract presentation is more suitable for focus guidance [Krüg00].

Additionally, the user has to be able to navigate in the 3D space in real time, so that a good spacial impression of the 3D representation and the spacial relationship between the object may arise [Frey97]. The object's case data have to be accessible from the presentation to use the three-dimensional representation not only as sheer visualisation, but also as interface to the information system.

4.2 Overall concept

The geodata server provides geodata, especially three-dimensional city models and guarantees the highest possible availability of the database in a network environment. The 3D geodata server should be accesible at anytime and - due to advancing development of the mobile internet - from every place. In order to achieve this target, a suitable data model for administration and analysis is necessary. In addition to this, concepts for graphical abstraction and progressive transmission must be developed to allow a suitable visual processing of a query result. For this, the user's intention, the technical ressources of the output device and the communication infrastructuremust be taken in to consideration.

These demands for data integration and online data access are not specific for three-dimensional city models, but apply generally to all modern GIS applications. For this reason, the OpenGIS Consortium (OGC) was founded in 1996 with the aim to establish interoperability between different GIS systems and to create an open, distributed infrastructure for the use of geodata: „*The Open Geodata Interoperability Specification (OGIS) provides a framework ... that enables their users to access and process geographic data from a variety of sources across a generic computing interface within an open information technology foundation.*“ [BüMc96]. The OpenGIS concept is based on a distributed GIS architecture which is shown in fig. 4-3. In this structure, several layers for presentation (tier 1), data access (tier 2) and data management (tier 3) are defined. In this concept, data providers are located in tier 3, while mere data users only see tier 1. Real geodata,

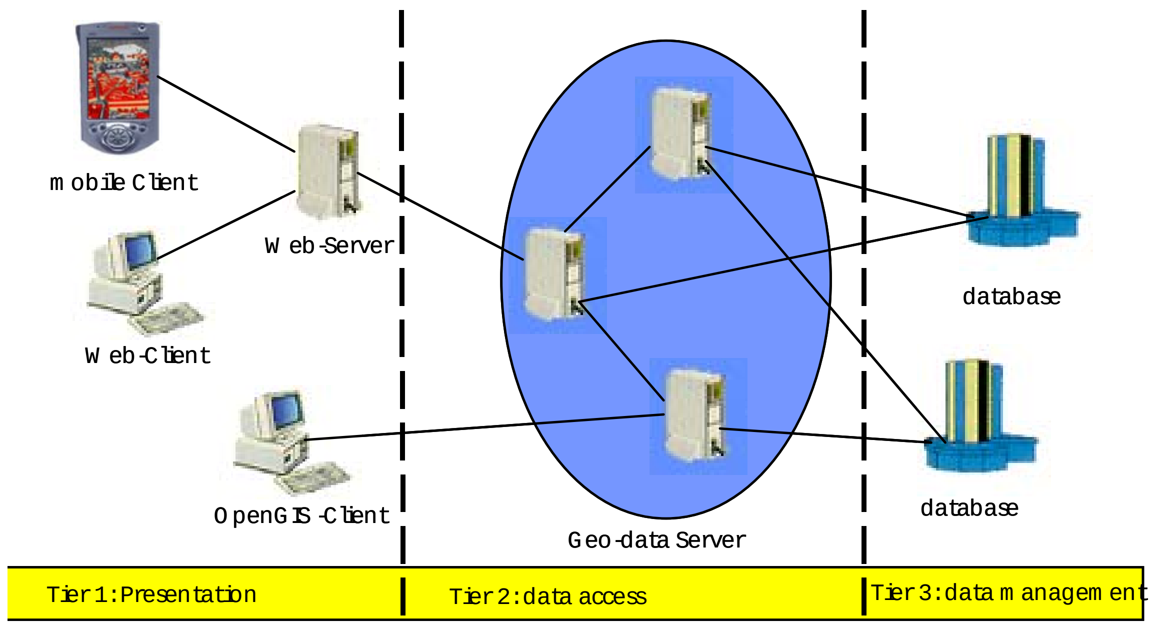


Fig. 4-3: 3-tier architecture of an open integrated GIS infrastructure

which may be collected from different data sources, is accessed in tier 2 via the geodata server. The present study includes the design and realization of prototypes of the necessary components of a 3D geodata server within this open GIS structure. Fig. 4.4 illustrates these components and their classification within the 3-tier architecture .

A data model for 3D geodata will be developed for data management, which ensures efficient processing of thematical, geometric and topological queries. The categorisation in section 2.2 shows, that our model is a query data model. The real queries are processed in a so-called query layer and form the interface between the data access (tier 2) and the data management (tier 3). Of course, in an interactive system the query layer has to provide a standardised interface to the presentation step, so that queries can be formulated and filed by the user. In this query layer, the evaluation of the objects' importance in connection with a query is processed. This evaluation is the basis for the graphical abstraction of the query result.

The real graphical abstraction of the relevant database and the necessary data transmission to the client form the interface between data access (layer 2) and data presentation (layer 1). Several abstraction methods for the visual processing of query results are discussed in section 5.2.3. The result of the graphical abstraction leads to a hierarchical structure, the *progressive tree*, or abbreviated: *P-tree*. The P-tree developed by Coors and Schulz allows interactive exploration and progressive data transmission of the visually processed query result in a heterogeneous network environment [CoSc99].

Fig. 4-5 shows the standard processing of a user's query to the geodata server. The user formulates the query interactively in a network-based 3D viewer. A typical query is the

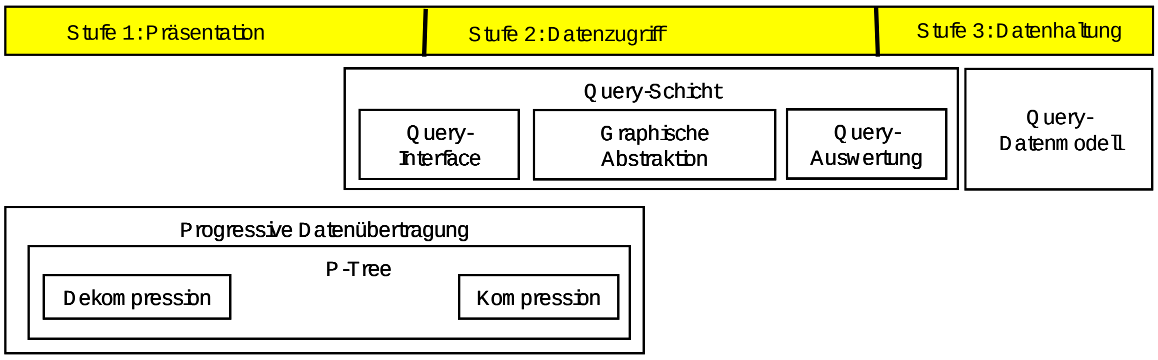


Fig. 4-4: Components of a 3D geodata server and their categorisation within the 3-layer architecture of a GIS developed in this thesis.

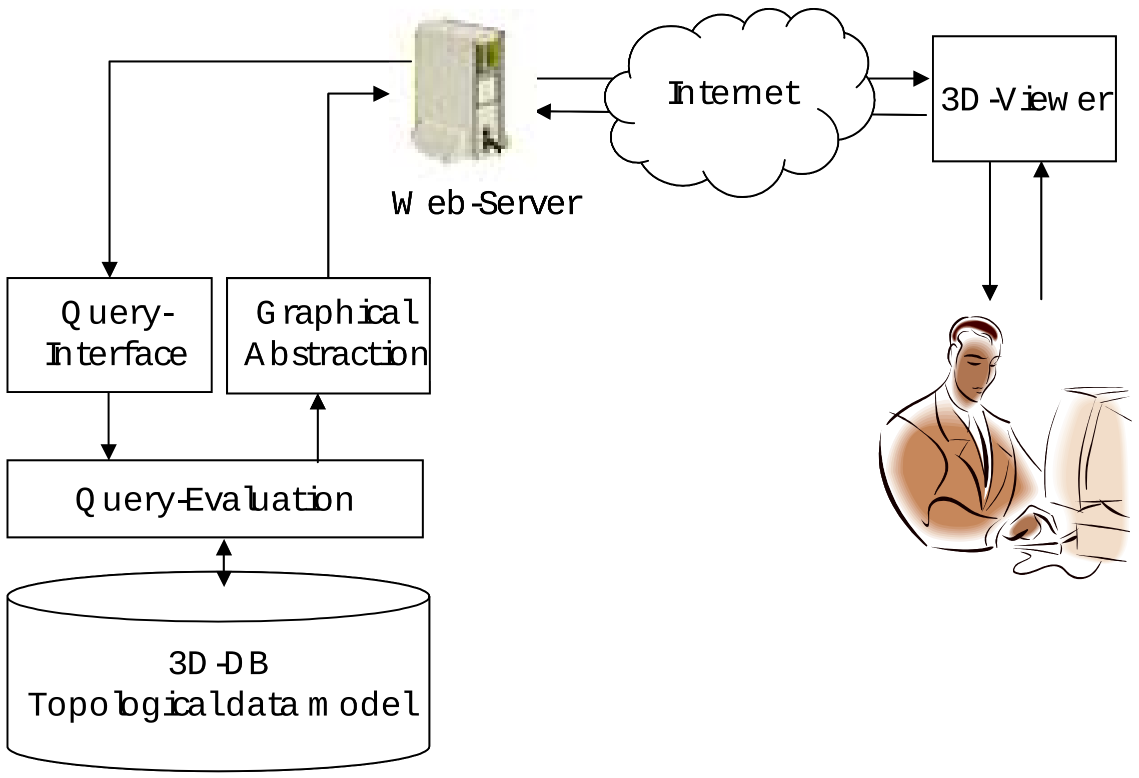


Fig. 4-5: Query excecution and visual processing of query results using graphical abstraction.

selection of specific spatial features in a specific region, e.g. all hotels in the inner city of Darmstadt. The query will be passed to the webserver through the internet, which forwards the query to the 3D geodata server. Inside the 3D geodata server, the query will be received by the query interface, interpreted and forwarded to the query analysis for further processing. At this point, the underlying database will be analysed according to the query. The query data model which is used for the storage of the 3D database ensures an efficient evaluation of typical requests in connection with spatial access structures. Query results are visually processed for the user using graphical abstraction. The resulting three-dimensional scene will be progressively transmitted to the internet client and interactively visualised there.

In order to use the available bandwith more efficiently, the model to be transmitted can be compressed. This is illustrated in fig. 4.6. The query result will be stored on the 3D geo-data server in a hierarchical structure, the so-called P-tree. Graphical abstraction makes use of the hierarchical structure of the P-tree and can process the three-dimensional scene for a user specific focus. This is similar to the level-of-detail concept. The procedure significantly reduces the initial data volume to be transmitted, because the P-tree is transmitted progressively to the 3D viewer. This means, that an initial abstract model is transmitted to the client, which is consecutively refined depending on the user's interaction. Since the P-tree is not usually transmitted completely to the 3D viewer in this first stage, a partial P-tree must be administrated on the client side which can be consecutively completed. The compression of the partial P-tree or the single elements in the P-tree further reduces the data volume. For this purpose, a mesh compression method developed by Coors and Rossignac is implemented [CoRo02]. The method and the use of the P-tree for progressive data transmission are introduced in chapter 5.3.

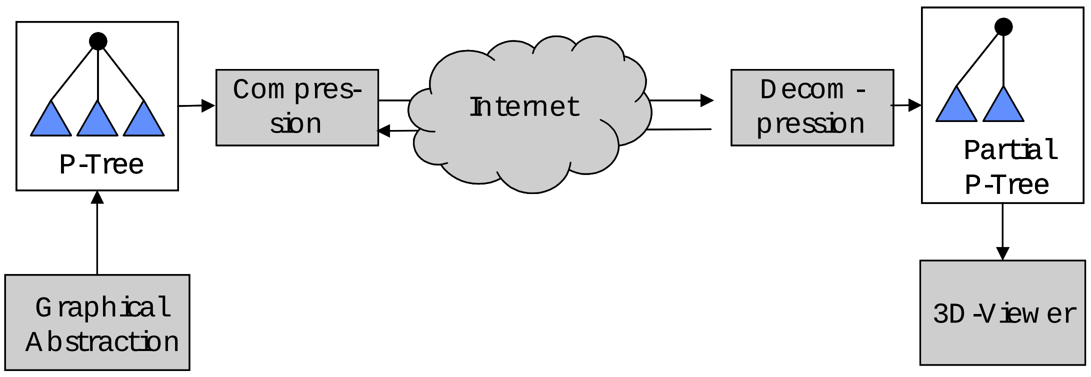


Fig. 4-6: P-Tree und Compression for efficient transmission of the 3D model.

Chapter 5

Central Components of the 3D Geodata Server

The 3D geodata server manages 3D geospatial data using a database. In order to ensure efficient processing of thematic, geometric and topological queries, a specialized data model for 3D geodata has to be developed. According to the categorization described in section 2.2., this data model is a query-oriented data model. The actual queries occur in a so-called query layer and form the interface between tier 2 and tier 3. Of course, in an interactive system, the query layer also has to provide a standardized interface to the presentation tier, so that the user can formulate and pose queries. Within the query tier, the importance of objects concerning a query is also evaluated. This evaluation forms the basis for graphical abstraction of the query result.

The actual graphical preparation of the relevant data and the necessary transfer of the data to the client form the interface between the data access (tier 2) and the presentation of the data (tier 1). Different abstraction methods for the visual preparation of a query result are discussed section 5.2.3. The result of this graphical abstraction leads to a hierarchic structure, the *Progressive Tree*, or P-Tree for short. The P-Tree developed by Coors and Schulz allows an interactive exploration and progressive transfer of the visually prepared query result in a heterogeneous network environment [CoSc99]. In order to better utilize the available bandwidth, the presentation geometry can additionally be compressed. For this, a procedure developed by Coors and Rossignac for compressing meshes is used [CoRo02]. These procedures and the use of the P-Tree for progressive data transfer are described in chapter 5.3.

In the following section, the query-oriented data model is developed, which is used within the 3D geodata server for managing the database and which forms the basis for the analysis functionality of the 3D geodata server.

5.1 Query-Oriented Data Model UDM

Before describing in detail the query-oriented data model for managing and analyzing urban data that was developed within the context of this thesis, the *Urban Data Model*, or *UDM* for short, the requirements for such a model, which were already discussed in the previous chapter, will be briefly recapitulated:

- Feature-based structurization of individual objects
- 3D topology for efficient processing of topological queries
- Low data volume for storing practically-relevant data
- Use of the query-oriented data model in standard databases
- Scale-independent storage of data
- Data transfer from and to collection data models

In the data model described here, discrete real world objects are modelled as features. Here, in analogy to the OpenGIS data model, a difference is made between elementary (*simple*) and composed (*complex*) features [BüMc96]. The spatial dimensions of elementary features are described by exactly one geometric primitive. A *point*-primitive represents a 0-dimensional object, 1-, 2- and 3-dimensional objects are realized through *line, surface* and *body*-primitives. These primitives are modelled using the construction elements *node*, *arc*, *face* and *solid*. A composed feature is formed by the aggregation of several elementary features.

The explicit management of the topological relations between primitives and construction elements, as proposed for example by Flick (see section 3.3.2), is often contradictory to compactly storing the spatial dimensions of a feature. Due to the (semi)automatic data collection, three-dimensional entities of the real world are represented as polygonal models or as CSG models with polyhedral base primitives. It is therefore justifiable to restrict the data model to the effect that *face* and *solid* only describe polygonal or polyhedral geometries. In polygonal models, edges can be defined implicitly by the order of the vertcies. Explicit storing of the edges is not necessary. As the number of edges in a polyhedron approximately corresponds to the sum of vertices and faces, the storage volume can thus be significantly reduced. Zlatanova has proven that all topological relations between the primitives can also be stored when using implicit storing of the edges [Zlat00]. The relations between the primitives are calculated based on the topology of the construction elements. Fig. 5-1 shows the conceptional data model with the relations between primitives and construction elements. The topology of the construction elements is discussed in detail in section 5.1.3.

Contrary to the SSS model according to Zlatanova, edges for line-shaped objects are, however, stored explicitly. Thus the model space for linear features is conceptionally

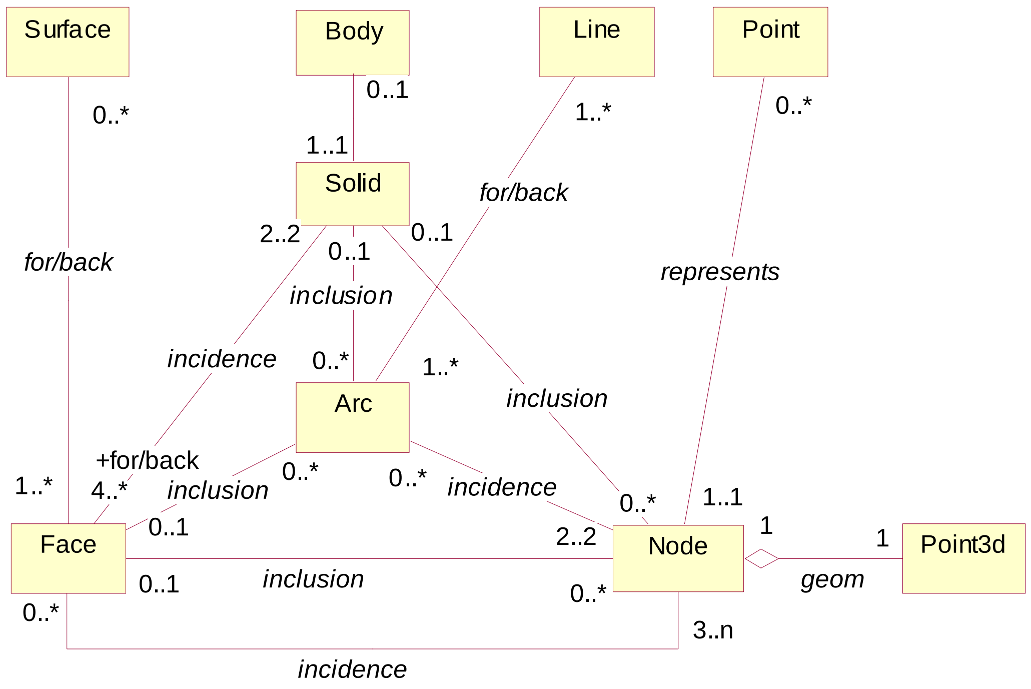


Fig. 5-1: Conceptional query-oriented data model in UDM

extended. It is thus possible to model streets, for example, not just by piecewise linear geometries, but also by using arc segments or other non-linear elements.

This geometric model is explained in detail in the following section. The point sets interior, boundary, closure and exterior of a construction element or of a primitive always concern the geometry of the element. Thus, the interior of an arc, for example, strictly speaking means the interior of the arc geometry. For reasons of better readability, the short form is always used in the following. The most important properties, however, are described here in advance:

- All points that are necessary for describing the geometry are treated as *nodes*.
- Two *nodes* are connected by a maximum of one *arc*.
- An *arc* is only stored explicitly if it is required for describing a *line* feature. *Arc* elements whose only function is to delimit *faces* are not stored explicitly.
- An *arc* is geometrically represented by a straight line. Conceptually, a non-linear geometry is also conceivable for explicitly stored *arcs*.
- Two *arcs* are either disjoint or they intersect in a common *node*. The intersection of the interior of two *arcs* is the empty set.
- *Face* elements are geometrically represented by planar convex polygons. The intersection of the interior of two *faces* is the empty set.

- A *face* is described by a list of *nodes*.
- A *face* has an orientation and thus an unambiguously defined right and left side. The order of the *nodes* defines the orientation of the face.
- The intersection of the interior of *arc* and *face* is either the empty set or the interior of the *arc* itself.
- A *solid* is defined by the faces that delimit it. A *face* bounds exactly one *solid* on the right and one *solid* on the left side. This convention presupposes that even unoccupied space is represented as solid.
- A *point* primitive is represented by exactly one *node*.
- A *line* primitive consists of a simply successional set of *arcs*.
- In analogy, a *surface* primitive is described by a simply successional set of *faces*.
- A *body* primitive is represented by exactly one *solid* element.

The geometrical construction elements and primitives are formally described in the following sections. This is followed by an analysis of the topological relations between the construction elements, from which the topology of the primitives can be derived.

5.1.1 Geometrical Construction Elements

When defining the geometrical construction elements and in particular when deriving topological relations between construction elements and primitives, the actual geometry as well as the interior, exterior, closure and boundary of this geometry must also be taken into account. The formalism employs fundamental concepts of set theory [Will70].

Node

Nodes represent points. The geometry *G(n)* of a node *n* is defined by a point $P \in \Re^3$ with the coordinates $P = (x, y, z)$: *G(n)=P.*

Two different *nodes* n_j and n_k cannot represent the same point *P*, i.e. $G(n_j)=G(n_k) \Rightarrow j=k$.

The interior $°n$ of a *node n* is the empty set. The boundary ∂n is the geometry of the *node* itself, i.e. $\partial n = G(n)$.

Arc

Two *nodes* can be connected by the topologically one-dimensional element *arc*. An *arc* $a=(n_b, n_e)$ has exactly one starting-*node* n_b and exactly one end-*node* n_e. The functions *Begin(a, n)* or *End(a, n)* define whether a given *node n* is a starting or end-*node* of the *arc a*:

$$Begin(a, n) = \begin{cases} 1, n = n_b \\ 0, n \neq n_b \end{cases},$$

$$End(a, n) = \begin{cases} 1, n = n_e \\ 0, n \neq n_e \end{cases}$$

The starting and end-*node* of an *arc a* may not be identical, i.e.

$Begin(a, n_j) = 1 \wedge End(a, n_k) = 1 \Rightarrow j \neq k$.

Two *nodes* can only be connected by one *arc* maximally. If there is an *arc* $a_1=(n_b, n_e)$, then there is no *arc* $a_2=(n_e, n_b)$.

The geometry *G(a)* of an *arc a* is defined implicitly by a distance between starting and end-point, i.e. $G(a) = \{ P \in \Re^3 \mid P=tG(n_j) + (1-t)G(n_k),\ 0 \leq t \leq 1 \}$. The boundary ∂a of an *arc a* contains the starting and the end point, i.e. $\partial a = G(n_b) \cup G(n_e)$.

The interiors of two *arcs* a_j and a_k may not intersect, i.e. $°a_j \cap °a_k = \emptyset$. A *node n* may not be located in the interior of an *arc a*: $°a \cap G(n) = \emptyset$. A breach of these conventions can be solved by suitable division of the affected *arcs* and, if necessary, by inserting new *nodes*. Fig. 5-2 illustrates these conventions.

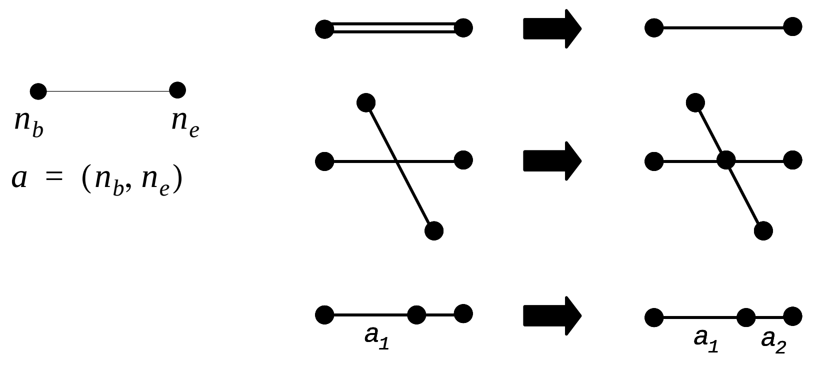


Fig. 5-2: Conventions *node/arc*

Face

The topologically two-dimensional element *face* $f=(n_0,....,n_k)$ is defined by a list of delimiting *nodes*. The order of the *nodes* implies the geometry and the orientation of the *face*.

The requirements for the delimiting *nodes* n_i of a *face f* are that their geometries $G(n_i)$ are coplanar. Then the geometry *G(f)* of a *face f* is defined by the intersection of the half-planes spanned between each two successive *nodes*. It is required that the *face*-geometry has to be a convex polygon, so that $\forall i.(G(n_i) \subset Gf))$.

The boundary ∂f of *f* contains the geometry of all delimiting *nodes* and the linear connections between each two successive *nodes,* i.e.

$$\partial f = \bigcup_{n_i \in f} \{P \in \Re^3 | P = tG(n_i) + (1-t)G(n_{i+1}), 0 \le t \le 1\}$$

The orientation of a *face f* is defined by the cross product of the difference vectors of three successive *nodes* that delimit *f*:

$$l = (G(n_{(i \bmod k)}) - G(n_{((i-1) \bmod k)})) \otimes (G(n_{((i+1) \bmod k)}) - G(n_{(i \bmod k)})), 0 < i \le k$$

If you traverse the delimiting *nodes* of *f* in an counter-clockwise direction, *f* is always on the left side of this path. In the following, the direction of this vector *l* is therefore called *left*, the opposite direction *right*. Fig. 5-3 illustrates the orientation of a *face f.*

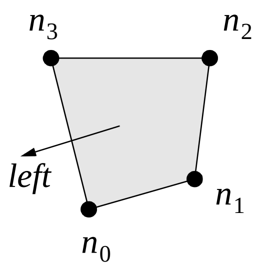


Fig. 5-3: Orientation of a face $f = (n_0, n_1, n_2, n_3)$

Two *faces* f_1 and f_2 may not overlap or cross one another. The intersection of the interior is the empty set: ${}^{o}f_1 \cap {}^{o}f_2 = \varnothing$. An *arc a* must be either entirely contained in a face *f* or it may only touch *f* in one *node,* i.e. $°a \cap G(f) \neq \varnothing \Rightarrow G(a) \subset f$.

A breach of these two conventions can be corrected by suitable division of the *faces* and, if necessary, by inserting new *nodes,* as illustrated in fig. 5-4.

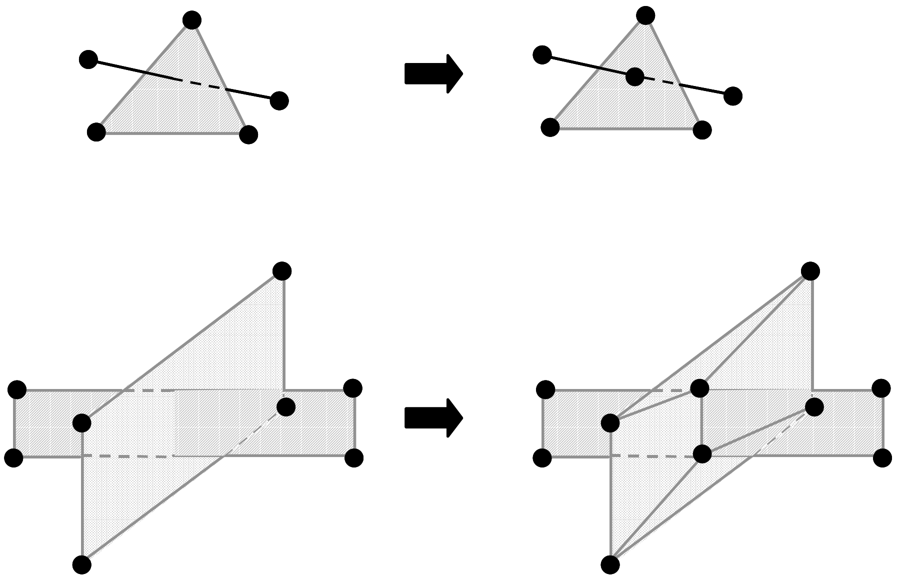


Fig. 5-4: Conventions *face*

A *node n* delimits a *face f* when the geometry of the *node* is a subset of the boundary of *f*. A decision function *b(n, f)* verifies this relation:

$$b(n, f) = \begin{cases} 1, G(n) \subset \partial f \\ 0, G(n) \not\subset \partial f \end{cases}$$

The set *B(f)* of all *nodes* that delimit a *face f* is then given by *B(f)={n|b(n,f)=1}*.

A further relation between *node* and *face* is inclusion. A *node n* is contained in a *face f* if it is located in the interior of the *face*. The boolean decision function *isin(n,f)* computes this property:

$$isin(n,f)=1 \leftrightarrow G(n) \cap {}^{\circ}f \neq \emptyset$$

If a *node* is located within a *face*, it may not delimit another *face*. This restriction avoids situations as shown in fig. 5-5a from occurring.

$$\exists k.isin(n, f_k) = 1 \rightarrow \forall i \neq k.b(f_i, n) = 0$$

The adjacency of two *faces* can only be deduced from the delimiting *nodes*. Two *faces* f_1 and f_2 are adjacent to one another if they are delimited by at least two common nodes: $|B(f_1) \cap B(f_2)| \geq 2$. This applies due to the required convex geometry of the *faces* and the above-mentioned restrictions for the *node*/*face* inclusion. Without these restrictions, the *faces* depicted in fig. 5-5a would not be recognized as being adjacent to one another, as they do not dispose of any common *nodes*.

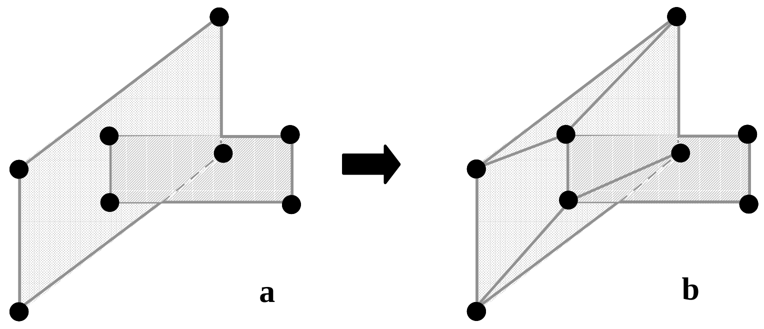


Fig. 5-5: Convention *node/face* inclusion

If the *node*/*face* inclusion convention is observed, two further relations between faces can be derived. If $|B(f_1) \cap B(f_2)| = 1$, the *faces* are touching in exactly this *node*. If the intersection is empty, then the two *faces* are not connected.

Solid

A *solid* $s=\{(f_0, o_0), (f_1, o_1),..., (f_n, o_n)\}$ is described by a set of *faces* that delimit the *solid*. Using the direction $o_i \in \{forward, backward\}$, each *face* is oriented so that *left* points towards the interior of the *solid*.

The geometry of the *solid* is the intersection of the half-planes spanned between (oriented by) the delimiting *faces*. The boundary of a *solid* is identical to the geometry of the *faces* that delimit the solid, i.e. $\partial s = \{G(f)|(f, o) \in s\}$.

The space that is to be depicted is divided in such a way that each *face* has exactly one *solid* on the left and exactly one *solid* on the right side. A boolean decision function *bleft(f,s)* or *bright(f,s)* computes this relation. Since unoccupied space must be modeled as well a special *solid* AIR is introduced that represents no matter to maintain this relation.

The function *b(f,s)* decides, whether a *face f* delimits a *solid s*:

$$b(f,s)=bleft(f,s) \text{ xor } bright(f,s).$$

The set B_f of all *faces* that delimit *s* is then given by $B_f(s)=\{f|b(f,s)=1\}$. The set B_n of all *nodes* that are located on the boundary of *s* is given by

$$B_n(s) = \bigcup_{f_i \in B_f(s)} b(n, f_i)$$

5.1.2 Geometric Primitives

Point

A *point* can be used to model a topologically 0-dimensional *feature*, a so-called *point-feature*. A *point P* is represented by exactly one *node n*: $P = n$.

Contrary to the *svvm*-concept according to Molenaar [Mole98], a *node* can represent different *points*. In this way, two *point-features* can be modelled by two different *point-primitives* that have the same location and the same topological relations, i.e. $P_1=n_0$ und $P_2=n_0$.

Line

A *line* $L=\{(a_0, o_0), (a_1, o_1),..., (a_n, o_n)\}$ is defined by a set of simply successional *arc* elements. The orientation of a *line* can be modelled by specifying a direction $o \in \{forward, backward\}$ for each *arc* $a=(n_b, n_e)$. A *forward arc* stretches from n_b to n_e, a *backward arc* is oriented in the opposite direction. Each *arc* is contained in at least one *line,* but contrary to the *svvm*-concept, it can also be contained in several *line* primitives. A *line* can be used to model topologically one-dimensional features such as streets, pipelines and similar objects.

The boundary of a *line l* is the geometry of the *nodes n* that delimit exactly one *arc* contained in *l.*

$$\partial L = \{G(n) | \exists a_i \in L.\ G(n) \in \partial a_i \wedge \forall a_k \in L, a_k \neq a_i.\ n \notin \partial a_k\}$$

If the *line* is closed, the boundary is the empty set, otherwise, due to the simple relation, there are exactly two *nodes* that fulfill this property.

Surface

A *surface* $S=\{(f_0, o_0), (f_1, o_1),..., (f_n, o_n)\}$ is represented by a set of simply connected *face* elements. A *surface* can be oriented by specifying a direction $o_i \in \{forward, backward\}$ for each *face* f_i. A *forward face* is oriented in the direction of the normal vector *l,* a *backward face* in the opposite direction. This means that in a *backward face,* the *left* and *right* relations are switched. A *surface* can be used to model topologically two-dimensional *features,* such as a roof or the ground plan of a building.

The boundary of a *surface S* is given by the set of the boundaries of all *faces* f_i that are not simultaneously boundary of another *face* f_k of *S.*

$$\partial S = \{\partial f | \exists f_i \in S.\ \partial f_i = \partial f \wedge \forall f_k \in S, f_k \neq f_i.\ \partial f_k \neq \partial f\}$$

Body

A *body* is used to model a topologically 3-dimensional *feature*, a so-called *body feature*. A *body B* is represented by exactly one *solid s*: *B*=*s*.

5.1.3 Topological Relations Between the Construction Elements

The description of topological relations between two construction objects in the data model is carried out using the *9-intersection* model according to Egenhofer and Herring [EgHe90]. In this model, the relation of two simple spatial objects *A* and *B* is differentiated according to their interior, their boundary and their exterior. The relation *R(A,B)* is defined using the intersections of two of these point sets each, whereby it is only of importance whether the intersection is the empty set or not.

$$R(A, B) = \begin{bmatrix} {}^{\circ}A \cap {}^{\circ}B & {}^{\circ}A \cap \partial B & {}^{\circ}A \cap B^{-} \\ \partial A \cap {}^{\circ}B & \partial A \cap \partial B & \partial A \cap B^{-} \\ A^{-} \cap {}^{\circ}B & A^{-} \cap \partial B & A^{-} \cap B^{-} \end{bmatrix}$$

The 9-intersection model does not distinguish the dimensionality of the intersections of two point sets. The two Face-Face situations shown fig. 5-7, have the same relation in the 9-intersection model. More complex models Clementini Operator [Clem93], or CNRG [RoRe91] distingish these situations. However, the 9-intersection model it is sufficient for all typical topological queries posed to a 3D geodata server.

The individual elements of the matrix can also be written in succession in the following order:

$$(\partial A \cap \partial B)({}^{\circ}A \cap {}^{\circ}B)(\partial A \cap {}^{\circ}B)({}^{\circ}A \cap \partial B)(A^{-} \cap B^{-})(A^{-} \cap \partial B)(A^{-} \cap {}^{\circ}B)(\partial A \cap B^{-})({}^{\circ}A \cap B^{-})$$

If the empty set is represented by a 0 and the non-empty set by a 1, this notation can be interpreted as a binary number. This facilitates the notation of the relations. For example, for two disjoint elements *A* and *B*, the binary number d=$000011111_{\text{bin}} = 31_{\text{dec}}$ is obtained. The relation is thus called R031.

***Node / Node* Relations**

There are only two different topological relations possible between two *node* elements n_0 and n_1 , R026 (*disjoint*) and R272 (*meet*). The interior of a *node* ${}^{\circ}n$ is by definition the empty set. If the intersection of the boundary is not the empty set, i.e. $\partial n_0 \cap \partial n_1 \neq \varnothing$, then $n_0^{-} \cap \partial n_1 = \varnothing$ and $\partial n_0 \cap n_1^{-} = \varnothing$ and thus *R*(n_0, n_1)=100010000=R272. Otherwise, in analogy, $R(n_0, n_1)$=000011010=R026. In the data model, these relations can be derived using the identity, as two different nodes must have different geometrical characteristics. Fig. 5-7 illustrates these two relations.

Arc / Node Relations

Only two different topological relations are possible between an *arc a* and a *node n*, *disjoint* and *meet*. The interior of a *node* $°n$ is by definition the empty set. Due to the convention between *arc* and *node* defined in section 5.1.1, $°a \cap \partial n = \emptyset$., the intersection between the exterior of *n* and the interior or boundary of *a* is always a non-empty set, i.e. $°a \cap n^- \neq \emptyset$ or $\partial a \cap n^- \neq \emptyset$. As with *node*/*node*, the relation of exterior and boundary depends directly on the relation of the boundaries. Thus, the two cases *R(a,n)*=000011011=R027 (*disjoint*) or *R(a,n)*=100010011=R275 (*meet*) apply.

Both relations can be derived from the data model. An incidence-relation (*meet*) is the case when *Begin(a,n)+End(a,n)=1*. Otherwise, *a* and *n* are independent.

Arc / Arc Relations

There are three different relations possible between two *arc* elements a_0 und a_1. Due to the convention $°a \cap \partial n = \emptyset$ agreed upon in section 5.1.1, the interior of a_0 and the boundary of a_1 are always disjoint. Based on $°a_j \cap °a_k = \emptyset$, one can conclude that if the intersection of the interiors of a_0 and a_1 is a non-empty set, the two *arcs* are identical. Thus exterior of a_0 and boundary or interior of a_1 and vice versa are each disjoint. In this case, the relation R(a_0, a_1)=110010000=R400 (*equal*) applies. If $°a_0 \cap °a_1 = \emptyset$, then it immediately follows that the intersection of the exterior of a_0 with the boundary or the interior of a_1 are both not empty and vice versa. Depending on the relation of the boundaries, two further relations can occur: R(a_0, a_1)=000011111=R031 (*disjoint*) and R(a_0, a_1)=100011111=R287 (*meet*).

The relation between two *arcs* $a_0=(n_0^b, n_0^e)$ and $a_1=(n_1^b, n_1^e)$ can be derived in the data model from the function

$$r(a_0, a_1) := Begin(a_0,n_1^b)+End(a_0,n_1^b)+Begin(a_0,n_1^e)+End(a_0,n_1^e)$$

If r=0, then R(a_0, a_1)=R031, if r=1 then R(a_0, a_1)=R287 (*meet*), and if r=2, then a_0 *equal* a_1.

Face / Node Relations

There are three distinguishable topological relations *R(f,n)* for a *face* and a *node*. The interior of a *node* $°n$ is the empty set. The intersection of the exterior of a *node* with the interior or the boundary of a *face* is always a non-empty set. If the boundary of *n* is contained in the boundary of *f*, i.e. $\partial f \cap \partial n \neq \emptyset$, then it immediately follows that the intersection of the interior or the exterior of *f* with the boundary of *n* is the empty set. In this case the relation *R(f,n)*=R275 (*meet*) applies. Otherwise, the two boundaries are disjoint and a difference is made on whether the boundary of *n* is contained in the interior of *f* or not. This leads to the relations *R(f,n)*=R051 (*contain*) or *R(f,n)*=R027 (*disjoint*).

In the data model, the *face*/*node* relations are derived from the decision functions *b(n,f)* und *isin(n,f)*. If *b(n,f)=1*, then *R(f,n)*=R275. This corresponds to an incidence relation between *f* and *n*. If *b(n,f)=0* and *isin(n,f)=1*, then the inclusion *R(f,n)*=R051 applies. Otherwise, the relation R027 applies.

Face/Arc Relations

The possible topological relations between a *face f* and an *arc* $a=(n_b, n_e)$ according to [Bric93] are shown in fig. 5-6. Due to the convention $°a \cap f \neq \emptyset \Rightarrow a \subset f$, the relations R159, R223, R255 and R479 are not possible in the data model. Thus 7 distinguishable topological relations between *face* and *arc* remain.

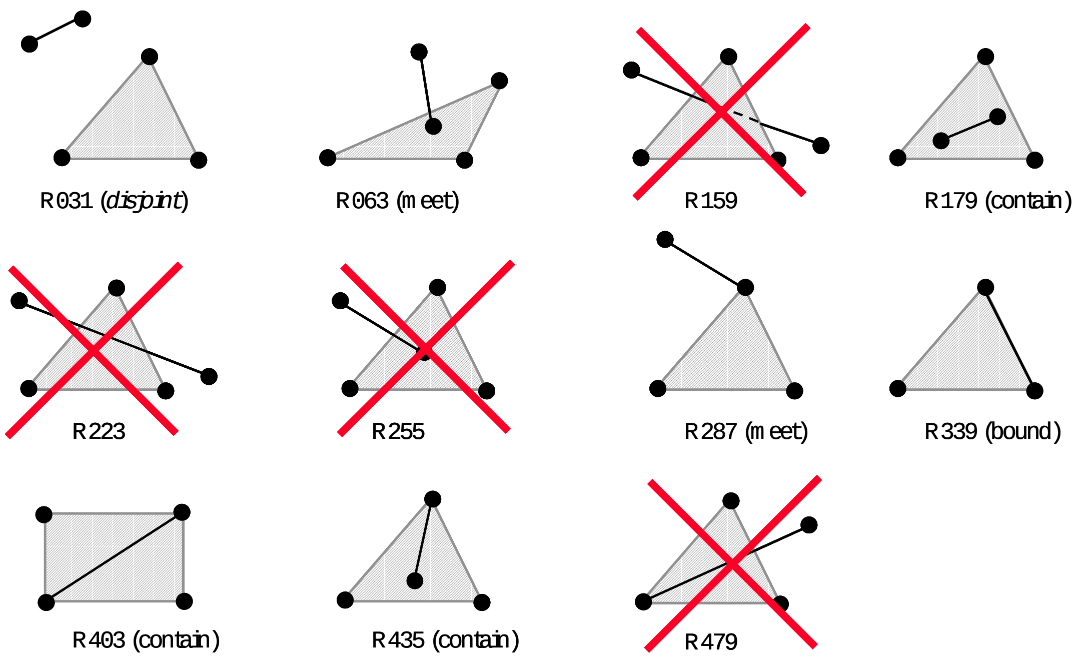


Fig. 5-6: *Face/arc* relations according to [Bric93]. The relations R159, R223, R255 and R479 are not possible due to the conventions in UDM.

In the data model, the different relations can be unambiguously identified using the functions *b(n,f)* and *isin(n,f)*. For reasons of clarity, the sum of the common delimiting *nodes* is called *sb*, so that $sb=b(n_b,f)+b(n_e,f)$. Similarly, $sin=isin(n_b,f)+isin(n_e,f)$. The order of the *nodes* delimiting *f* is only necessary[1] in addition for differentiating between the relations R339 and R403. It is not necessary to explicitly store the incidence between *face* and *arc*.

A *face f* and an *arc a* are disjoint, i.e. *R(f,a)*=R031, if *sb*+*sin*=0. The incidence relation R063 applies when *sb*=0 and *sin*=1. The inclusion R179 (*contain*) is unambiguously de-

1.Relation R403 is not possible if the *face* geometry has been restricted to triangles. However, when inserting an *arc*, a retriangulation may be necessary.

fined by *sb*=0 and *sin*=2. The second incidence relation R287 (*meet*) follows from *sin*=0 and *sb*=1. For the two relations R339 (*bound*) and R403 (*contain*), *sin*=0 and *sb*=2. These two cases can only be distinguished by the order of the *nodes* in the *face* definition. If n_b is the direct predecessor or successor of n_e on the boundary of the *face f*, situation R339 is the case, otherwise R403. The inclusion R435 applies when *sb*=1 and *sin*=1.

Face / Face Relations

Due to the conventions, the intersection of the interiors of two *faces* f_1 and f_2 is always the empty set. The only exception to this rule is the identity relation, i.e. $R(f_1, f_2)$=R400. Furthermore, two *faces* can be disjoint, which is described by R031, or can be touching in at least one *node*. In the latter case, one has to distinguish between the two relations R287 and R351 or R319. In the case of R287, the *faces* are only touching at the boundaries, the intersection of the interior of f_1 with the boundary of f_2 is the empty set. Otherwise, the boundary of one *face* is partly located in the interior of the other *face*. If R351 and R 319 are symmetrical, i.e. if $R(f_1, f_2)$=R351, then it follows that $R(f_2, f_1)$=R319 and vice versa. Examples of *face*/*face* relations are depicted in fig. 5-7.

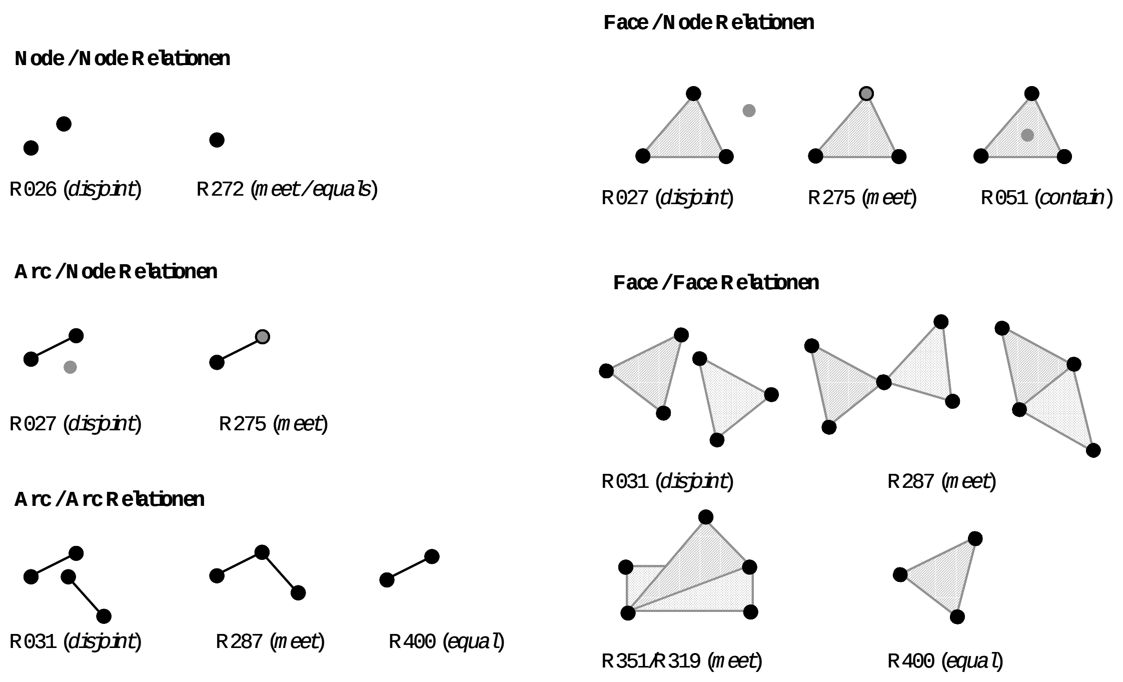


Fig. 5-7: Relations of construction elements

In the data model, the different relations between two *faces* f_1 and f_2 can be derived from the delimiting *nodes*. f_1 and f_2 are disjoint when the two sets of delimiting *nodes* $B(f_1)$ and $B(f_2)$ are disjoint. If $B(f_1)$ and $B(f_2)$ are identical, this also applies to the two *faces*. Otherwise, the order of the common *nodes* distinguishes between R287 and R351. If these are neighboring in both *faces*, R287 is the case, otherwise R351. It should be noted that relation R351 is not possible if the *face* geometry is restricted to triangles. Two faces f_1 and

f_2 are adjacent to one another if they are in relation R287 and are delimited by at least two common *nodes*: $|B(f_1) \cap B(f_2)| \geq 2$. If $|B(f_1) \cap B(f_2)| = 1$, the *faces* are touching in this *node*.

Solid/Solid

In the *9-intersection* model, there are exactly 8 distinguishable relations between two *solid* primitives [Zlat00]. Examples for each of these relations are illustrated in fig. 5-8. In the underlying data model, these relations are derived as follows.

Three cases can be distinguished if all delimiting *faces* of the two *solids* s_1 and s_2 are disjoint in pairs, i.e. $R(f_1,f_2)$=R031 applies for all f_1 of $B_f(s_1)$ and for all f_2 of $B_f(s_2)$. If there is a *node n* of $B_n(s_1)$ that is contained in the *solid* s_2, i.e. $isin(n,s_2)=1$, then the relation R220 (s_1 *inside* s_2) is the case. If, reciprocally, there is a *node n* of $B_n(s_2)$ with $isin(n,s_1)=1$, then R179 (s_1 *contains* s_2) is the case. Otherwise, s_1 and s_2 are disjoint (R031).

For the remaining 5 relations, the delimiting *faces* are not disjoint in pairs. The two *solids* are identical (R400) if each *node n* of $B_n(s_1)$ is also contained in $B_n(s_2)$. If no *node* n_1 of $B_n(s_1)$ is contained in s_2 and, conversely, no *node* n_2 of $B_n(s_2)$ is contained in s_2, i.e. if $isin(n_1,s_2)=0$ applies for all n_1 and $isin(n_2,s_1)=0$ applies for all n_2, then R287 (*meet*) is the case. In analogy to *face*, no difference is made here whether s_1 and s_2 are touching in a plane or only at a boundary. This adjacency can be distinguished in the data model, as only in the former case does a *face f* exist for which $left(f,s_1)+right(f,s_2)+left(f,s_2)+right(f,s_1)=2$ applies.

If there is a *node* n_2 of $B_n(s_2)$ that is located within s_1, i.e. $isin(n_2,s_1)=1$, but no node n_1 of $B_n(s_1)$ is located within s_2, i.e. $isin(n_1,s_2)=0$, then the relation s_1 *covers* s_2 (R435) is the case. If, reciprocally, a *node* n_1 of $B_n(s_1)$ lies within s_2, but no *node* n_2 of $B_n(s_2)$ within s_2, R476 (s_1 *covered by* s_2) is the case. The last relation, s_1 *overlap* s_2 (R511) is characterized by the fact that a *node* n_1 of $B_n(s_1)$ is located within s_2 as well as a *node* n_2 of $B_n(s_2)$ within s_2.

5.1.4 Presentation Model

The query-oriented data model described in the previous section is mainly characterized by a geometrical model, in which topological relations are represented appropriately, ensuring efficient processing of topological queries. Visualization and simulation aspects such as, for example, color properties, material properties for lighting simulations, textures for optical improvement of the model have not been considered yet. The concept of additional representations described in the following supports these aspects, which are primarily used in visualization.

Requirements

An important concept in 2D-GIS is the separation of graphic-describing data and object geometry. According to Bill and Fritsch, graphic-describing data is "information on the

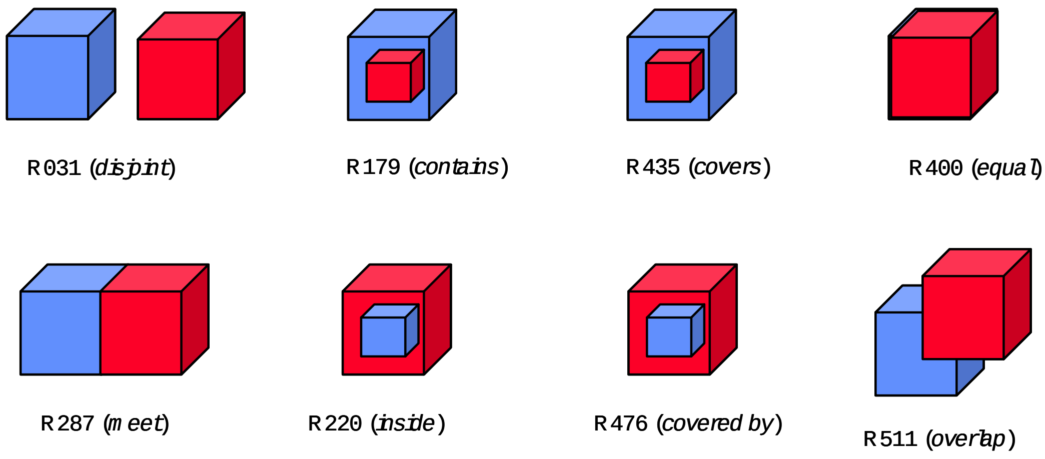


Fig. 5-8: 8 solid/solid relations

way in which a spatial object (i.e. geometry and attributes) is to be visualized for a certain topic and on a certain output device" [BiFr99]. This concept is implemented, for example, in the government land cadastral information system ATKIS [AdV99] using a so-called signature catalog. This is a basis of rules with which the graphical depiction of an object can be derived from its geometry and semantics. This signature catalog is supplemented by presentation objects such as labels, which cannot be created fully automatically for a certain target scale, and map-geometry objects, which replace the shape and/or position of a feature with increasing cartographic generalization.

The separation of presentation and geometry is a flexible concept that also must be supported in 3D-GIS. The difficulty in applying this concept in a three-dimensional visualization is the much larger range of requirements for the optical quality of the visualization, which ranges from an abstract depiction, as it is often used in environmental simulations [Knol98], to photorealism for decision support in planning.

The data required for an interactive three-dimensional visualization can be generally distinguished according to

- object-specific attributes such as color and texture and
- general properties of the environment.

It is often insufficient to assign object-specific attributes to an elementary feature, in other words a re-presentation of a real world object that is not further subdivided semantically. In a 3D land register, it may, for example, be desirable to regard buildings as elementary features. The assignment of one color per elementary feature, however, then only allows single-colored depiction of the building. This may be sufficient for an abstract presentation, but hardly fulfills the requirements of a photorealistic visualization.

In spite of this, this approach is often found in existing 3D-GIS. In [Zlat00], for example, the geometric appearance of a feature (*spatial object*) is described by a set of geometrical attributes that apply to the entire geometrical description of this feature. When using this approach for semantic modelling, appearance characteristics must also be considered. Geometry areas with different presentation attributes must be modelled as different elementary features and then be combined using aggregation to form a complex feature. This procedure is common in modelling systems. Within a GIS, however, features represent semantic units that have a meaning in the real world. These elementary features are predefined by an application-specific generalization in acquisition. Thus an approach that allows only one assignment of presentation attributes on the feature tier is too restrictive in GIS and has undesirable effects on semantic modelling. The assignment of object-specific attributes has to be possible on the level of the geometric construction elements.

Furthermore, the assignment of exactly one graphical representation corresponding to the object geometry is not sufficient. With only one graphical representation for example trees as 3D objects would have to be modeled as a body feature in the query-oriented data model. In most applications, however, trees are modelled as topologically zero-dimensional point features, as further analyses on the relation of the tree top to neighboring objects are not required. The visual representation of a feature should also be dependent on the scale, or, in an interactive visualization, be dependent on the distance of the viewer. In visualization systems, this is supported by a level-of-detail concept. The management of the different representations, however, should take place in the data model.

Flick has proposed a view-concept that allows a separation of geometry and presentation within a 3D-GIS [Flic98]. A geo-object then disposes of a set of different presentations, so-called views. These are independent of the topological dimensions and the geometry of the geo-object. Of course, the possibility for using a view derived from the geometry still exists. This concept is very flexible and can be used, for example, for realizing a level-of-detail concept within a web-based information system with interactive three-dimensional visualization [CoFl98].

In Flicks approach, however, the same data model is used for modeling graphical representations and for storing the actual geometry. This makes the management of the graphical representations very storage-intensive. Information is stored that is never used in analyses and only required for visualization. It is therefore not necessary to explicitly store topological relations for graphical representations. Besides, the view-concept according to Flick does not offer the possibility of defining conditions between presentation (view) and geometry. In a city model, for example, a building can be presented with different levels of detail, e.g. as a colored cuboid, as a cuboid using wall textures, or modeled as a highly detailed object. Nevertheless, the accuracy of the ground plan is to be preserved within a certain error tolerance. Flick’s conceptalso lacks metadata describing the different presentations and enabling automated selection of a suitable visualization model.

In the concept of additional representations proposed in this thesis, the separation of presentation and geometry is carried out based on the view-concept according to Flick. However, the concept is significantly extended in order to overcome the weaknesses described in the previous section.

View concept in UDM

This section describes the concept developed for managing the graphic-describing data in UDM. First, the requirements for the presentation of the data model, which were worked out in the previous section, are summed up:

- Separation of graphical attributes and object geometry
- Object-related graphical attributes not only related to one feature, but also on the level of the geometric modeling elements.
- Modeling environment properties such as light, fog or rain
- Different graphical attributes depending on the presentation form
- Dimension of the graphical representation independent of the topological dimension
- Specifications of boundary conditions (*constraints*) between graphic-describing data and geometry, such as, for example, accuracy of ground plan for buildings
- Metadata for automated selection of a suitable graphical representation

In order to fulfill these requirements, a set of additional representations, so-called views, $v=\{v_0, v_1,..., v_n\}$ can be stored for each spatial object or feature *f*. Each view contains a complete set of graphical attributes representing the spatial form and the appearance of the feature in this view. Each feature has at least one view v_0 that can be generated directly from the geometry using feature-related attributes such as color. In analogy to a signature catalog, a system of rules is used for assigning standard attributes to a feature class. If deviating attributes are to be used, these can be managed view-specifically.

However, feature-related graphical attributes are not sufficient for many visualizations. Thus, in further views, it is possible to change the attributes as well as the entire geometry of the feature. This depiction geometry is not available for topological queries. It is therefore not managed in the query-oriented data model, but in a model that is tailored to the requirements of an efficient visualization. In this case, the explicit storage of topological relations is not necessary. Suitable visualization models are irregular triangle *meshes* that can also be stored in compressed form as *compressed meshes*. The advantage of the *compressed mesh* is in efficient storage and low data volume for transmission. Within a mesh it is possible to use textures of attributes on the triangle and even on the point level. Furthermore, the use of meshes is advantageous because today’s graphic processors are optimized for the depiction of triangles.

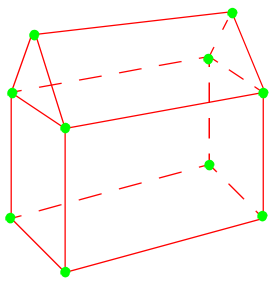
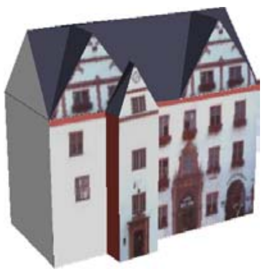
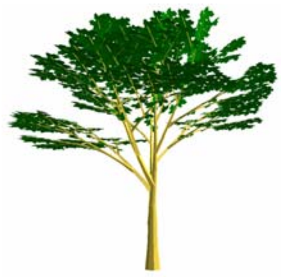
Feature Type	Geometry (Query-Oriented Data Model)	View (Depiction Geometry)
Building		
Tree		

Fig. 5-9: Different levels of detail and dimensionality in geometry and view

This means that a view *v* can change either feature-related attributes only, or the entire geometry including attributes on the level of the construction elements. The view *v* is therefore defined by

- a set of attributes $A=(a_0,..., a_n)$, whereby a_i are the graphical attributes of the corresponding feature class defined in the signature catalog, or
- a 2-tuple (M, A), whereby the mesh M contains the depiction geometry of the represented feature and A defines the graphical attributes related to the entire mesh. A mesh is represented by an irregular triangle network $M(V, T, A_V, A_T)$ with a set V of vertices, a set T of triangles, a set A_V of vertex attributes and a set of area attributes A_T.

When creating meshes as the view of a feature, it has to be possible to consider certain constraints. This is a conscious restriction of the flexibility of the view-concept that is necessary in order to ensure a certain correlation between feature geometry and representation. This can, as already mentioned, force an accurate visualization of a building's ground plan but can also prevent a building being visualized as a tree. If a detailed building geometry is available, a view can also be derived from this geometry using simplification procedures.

A view requires specification of the metadata describing the origin, the visual accuracy, the data volume and the like. Metadata play an important role in the automated selection of a suitable view as well as in graphical abstraction and network-based visualization. This will be detailed in the following chapter.

Fig. 5-10 displays the connection of the view-concept to the geometrical model.

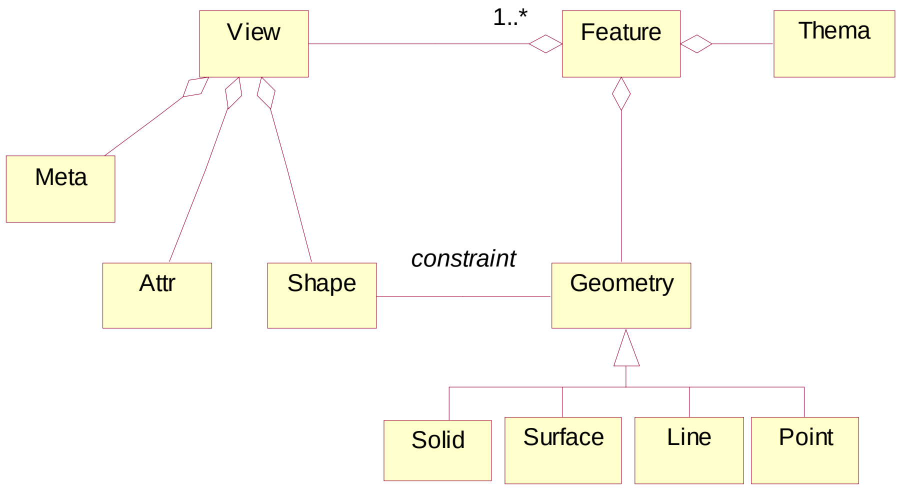


Fig. 5-10: Separation of geometry and presentation in UDM

5.2 Graphical Abstraction

In the previous chapter, the basic data model of a 3D geodata server was described. For efficient processing of topological queries, a topological data model was designed that supports typical topological queries with a small additional storage expense. Geometric queries can be additionally supported using a spatial access structure. A basic problem of many queries within a 3D model, however, is the large data volume of the resulting query result. Here the need for suitable visual processing of the results arises. The concept of separating geometry and presentation using views that was described in the previous chapter offers a solution to this problem: however, it does not help in deciding how a certain result set can be suitably processed visually. In this chapter, a concept for graphical abstraction is proposed. The objective of the abstraction procedure is to reduce the data volume to be visualized without reducing the information content. The procedure is based on a spatial access structure and considers the semantics of the features during the abstraction.

5.2.1 Requirements

In existing geographical information systems, the reality - the real world - is depicted in more or less abstracted models. In maps, for example, depending on the scale, important elements of a map are emphasized, others are summarized, represented by a symbol or completely left out. Similar to the two-dimensional abstraction of reality, the virtual landscape or the city model can also be understood as a three-dimensional abstraction of reality [Lang99]. In both cases, reality is presented symbolically and in a generalized form. Here, one must distinguish between a generalization in acquisition and a generalization in depiction. The generalization in acquisition is carried out application-specifically during modeling and data collection and thus forms the basis of the database. The goal of a GIS is to support this with a flexible data model. The actual generalization in acquisition, however, takes place outside of the GIS and is therefore not subject of further study.

The generalization in depiction, however, tries to depict a given data stock in a suitable way on a given output device. In cartography, this is the presentation of a basic data stock in additional maps of a smaller scale [Hake02]. Existing approaches for automated cartographic generalization were already discussed in section 2.6.

Similar principles apply to the visualization of a query result in 3D-GIS and to the design of two-dimensional maps. The three-dimensional visualization has a model character, i.e. the represented objects of the real world should be represented geometrically accurate and in the correct position. Additionally, visualization as a means of communication requires a sufficient degree of readability. This results, as in cartography, in the following principles:

- Geometrically accurate depiction.
- Depict same objects in the same way, dissimilar objects in dissimilar ways.

- Maintain important objects, leave out unimportant ones.
- Emphasize characteristic objects, force back random ones.
- Objects must have a minimum dimension to be recognized visually.
- The graphical differentiation has to be sufficient.
- The graphical density must not be too high.

These principles are partly contradictory. For example, a geometrically accurate depiction of all geographic objects of a city model on a standard display will immediately lead to a high graphic density. The graphic differentiation of individual objects becomes hard to fulfill.

The graphical abstraction within a 3D-GIS tries to solve this difficulty in the visualization of query results. One has to consider that, contrary to cartographic generalization, a targeted refinement of the model is possible in interactive three-dimensional visualization. There is therefore no fixed target scale for graphical abstraction in an interactive system. In this context, one also has to consider that a refinement of the model or a change in scale may occur via a network. In this case, adequate technology is required in order to avoid long waiting times. On the other hand, the user's target is immediate, as the visualization is always preceded by a concrete query. This user target should be considered in the graphical abstraction. Similar requirements can also be found in other application areas of graphical abstraction, for example in technical documentations [Krüg00].

5.2.2 Concept for Automated Graphical Abstraction in 3D-GIS

In this section, a concept for the automated graphical abstraction of a query result under consideration of the requirements discussed above is developed. For this, it needs to be clarified to what extent the abstraction targets can be formalized. Moreover, the parameters that influence the abstraction result need to be identified. These are mainly technical resources such as the output device used and cognitive resources of the viewer. Building on this, methods are described that can be used in abstraction.

Specification of Abstraction Targets

The goal of a GIS is to convey information to the user. For this, besides the actual data that is stored in the GIS, a suitable man-machine communication interface is also required that allows user-specific queries to the system and displays results in a suitable fashion. Thus, visual processing of query results is about the creation of a visualization with a topical focus. This topical focus has to be derivable from the preceding dialog with the user.

Creating a graphical abstraction with a given topical focus requires specification of all contributions of the individual features within this focus. One approach for this is to devide the objects to be depicted into foreground objects, which are in the content focus and therefore of significant importance to the graphic, and background objects that are not in the content focus. This approach was followed by Andre, for example, for generating multimedia presentations [Andr95].

Krüger distinguishes between four depiction classes: *identifiable, classifiable, discriminable* and *visible* [Krüg00]. For clarifying this division, imagine the visualization of a building. If significant details enable the viewer to recognize that the depicted building represents, for example, his own house, this is an *identifiable* depiction of this feature. If the depiction cannot be accurately associated with an object of the real world, but it is clearly recognizable as the depiction of house, this is a *classifiable* depiction. The class of houses is represented by the depiction. If a classification is no longer possible using the depiction, but the depiction can still be spatially delimited from other objects, this is a *discriminable* depiction. If even this property is lost, it is a *visible* depiction. Using a rule-based approach, Krüger assigns all objects with a depiction class that corresponds to the topical focus. This so-called *focus structure* is used as a starting basis for the process of graphical abstraction.

Objects in the content focus are of special importance for the visualization and often have to be depicted as identifiable or classifiable. Background objects can be of varying importance for the attainment of the abstraction target. Butz and Krüger propose to depict *landmarks*, which often enable the identification of objects in the content focus, as discriminable and to depict the remaining background objects as visible or to leave them out [BuKr97].

The division between identifiable and classifiable depictions is, however, determined not only by the depiction itself, but also by the knowledge of the user. Thus, even in a photo-realistic depiction, for a user that does not know Darmstadt, the Darmstadt Bell Tower is not identifiable, only classifiable as a tower or bell tower. For a native of Darmstadt, however, even a considerably more abstract depiction can be enough for identifying the model as the Darmstadt Bell Tower. Moreover, when demanding classifiable depictions, the possibility of specifying which class of objects is to be recognized is missing. Thus the depiction of a maple tree can represent the class of all maple trees but just as well the class of all trees. For classifiable depictions, a legend should therefore always be provided.

In this thesis, similar to the approach used by Hartmann et al. [Hart99], for specifying the abstraction target, each feature is assigned a dominance value that reflects the importance of this feature in the communication of the query result. The assignment of this dominance value is consciously carried out on the feature level and not on the view or geometry level. On the feature level, relations between different features can be modeled depending on the dominance. For example, if the chime in the Darmstadt Castle is explained in a city information system, then this chime has a high dominance value. Connected to this is an increase in dominance for the features containing the chime. The chime is part of the Bell Tower, which in turn is part of the Castle. These relations, however, are only available on the feature level. On the other hand, the depiction of special attributes is enough to succinctly depict this feature. For example, if a large building such as the Darmstadt Castle has a medium dominance value, then significant parts of the Castle should have a considerably higher dominance value, because these significant areas are usually sufficient for adequately depicting the Castle. A dominance function assigns each feature with a dominance value depending on the query [Coor02a]:

$$dom: F \times Query \rightarrow \Re$$

Calculation of the Dominance Function

A significant influence on the dominance of a feature is the relevance of this feature with regard to the query. The calculation of this relevance factor is detailed in the following. It is based on approaches for information visualization [KeKr94]. First, the distance between the query parameters and the feature characteristics is calculated for each feature. The distance between the characteristics and the corresponding query parameters depends both on the type of characteristic as well as on the application. The difficulty in this approach is to find suitable distance functions for the different data types. For metric data types, the numerical difference seems to be a suitable measure for the distance. Non-metric data types have a non-interpretable or unapparent distance. Here, application-specific distances must be defined depending on the characteristic. For spatial data, the spatial distance is always given, but often only the topological relations such as inclusion between two areas are of interest.

The distance of a query parameter p_q and the attribute value p_m of the corresponding characteristic m of a feature is calculated using a characteristic-specific distance function $dist(p_q,p_m)$. The distance between a feature and a query q is then given by the distance vector $d=(d_0, d_1, ..., d_n)^T$ with $d_i = dist(p_q{}^i,p_m{}^i)$.

Query parameters can be differently weighted depending on the user. For example, two tourists may be looking for accommodation in Darmstadt close to the Castle. Both also specify that the room should cost no more than EUR 75. With this query, all hotels are of the same relevance to both users. However, the price may play a much larger role than the location for tourist A, while tourist B attaches much more importance to the central location. Thus the distances between query parameters and characteristics are additionally multiplied with a user-specific weighting factor. For each query parameter j, a user-specific weighting factor w_j is used.

The relevance r_f of a feature f related to a query q with n query parameters is now calculated as the sum of the weighted distances of query parameters and corresponding feature characteristics:

$$r_f = \sum_j^n w_j \cdot d_j^f$$

Distance Functions for Different Order Types

The calculation of the distance between query parameter and corresponding characteristic depends on the order type of the characteristic. One distinguishes between the order types quantitative, ordinal, nominal and geographic. In order to avoid uneven weighting of different parameters, the distance function is standardized and the parameter always takes a value between 0 and 1. 0 means that the characteristic fulfills the query parameter. In the following, examples for distance functions of the different order types are given:

- *quantitative*: the value range of the parameter is ordered linearly and has a precise, numerical dimension. A distance function for quantitative characteristics evaluates, for example, 0 to query that searches for features whose value p_m of the characteristic *m* is smaller than or equal to a given value p_q. In this case, for example the following function can be used:

$$dist(p_m, p_q) = \begin{cases} 0, p_m \le p_q \\ 1 - \dfrac{p_q}{p_m}, otherwise \end{cases}$$

- *ordinal*: the value range of the parameter is ordered linearly. An example would be the comfort of a hotel, expressed in stars. A distance function for ordinal order types has to reflect the order of the markedness of the characteristic. Let p_m^1 and p_m^2 each be a markedness of the characteristic *m* with $p_m^1 < p_m^2$. Then the relation between the markedness of the characteristics must also apply to the distances to a query parameter p_q: $dist(p_m^1, p_q) < dist(p_m^2, p_q)$. An example for such a distance function is:

$$dist(p_m, p_q) = \begin{cases} 0, p_m \le p_q \\ 0.2(p_m - p_q), otherwise \end{cases}$$

- *nominal*: the value range of the parameter does not have a natural order. A special case of the nominal order type is the *boolean* type, which only takes the values *true* and *false*. For simplification purposes, only a valid markedness of the corresponding characteristic is permitted as query parameter. A distance function is generally defined using a matrix that assigns a distance to each two markednesses of a characteristic. Let *A* be the set of all markednesses of the characteristic *m*. The distance function assigns a distance to each pair of markednesses. This distance strongly depends on the semantic of the markednesses and is hard to generalize: $dist: A \times A \rightarrow [0, 1]$.

- *geographic*: the value range of the parameter is ordered by the geographic location. The geographic relation is given in space. Examples are coordinates, polygonal regions, but also addresses. In the sense of the order theory, this may not be a new order type (there it is equivalent to the quantitative order type), but for practical reasons, geographic parameters are a very useful extension of the classical trisection of order types. In the GIS context, it plays an outstanding role. An example for evaluating a query according to geographical characteristics is point-area inclusion. The user is looking for a hotel close to the train station. The spatial concept "close to" is expressed using a polygonal region *B*. The resulting query *q* thus has a parameter $p_q^L = B$, which relates to the geographical location. All hotels that are located in this region fulfill this condition, i.e. the distance between the location of a hotel $p_m^L = P$ and the query parameter p_q^L is 0. Hotels that are not located in the given region are assigned with a larger distance depending on the size of the region. For this, a threshold value τ is determined first, for example the shortest distance r_S of the center of gravity S_B of the region *B* to the boundary of *B*. If the distance r_P of *P* is larger than the threshold value τ, then the hotel is located too far away from the region to be of importance to the query, the distance between the location of the hotel and the query parameter is evaluated with 1. Otherwise, the distance is calculated using the ratio of r_P and τ:

$$dist(p_m, p_q) = \begin{cases} 0, p_m \in p_q \\ min\left(\frac{\|r_p, \tau\|}{\tau}, 1\right), otherwise \end{cases}$$

Examples for the use of the distance function for evaluating hotel queries are described in section 6.2.

However, features that do not fulfill the query criteria can also be important in the visualization of the result. This applies on one hand to landmarks that are necessary for orientation. For example, in a city model these are prominent buildings such as a historical building. Furthermore, reference objects are also of special importance if they were referred to in the query. In a query for all hotels close to the train station, the train station of course does not fulfill the query criteria, as it is not a hotel. In spite of this, it is of importance for the visualization and should therefore receive a high dominance value.

So, the dominance value of a feature is composed of the query-specific relevance *R(Query, User)*, the significance of the feature as a reference object in the query *O(Query)* and the general significance of the object as a landmark *L(User)*. The significance of a feature as a landmark to a certain degree depends on the user. In order to be able to model this dependence, stereotypes are used, to which specific values of the landmarks are assigned. Using the parameters a_0, a_1 and a_2, the individual values relevance, reference object and landmark that compose the dominance value of a feature, can be additionally weighted:

$$dom = a_0 * (1 - R(Query, User)) + a_1 * O(Query) + a_2 * L(User)$$

Resource Model

The term resources comprises all software, hardware and network components of the environment that are used for the visualization of the query result. The technical limitations of the output media have a substantial influence on the graphical abstraction.

Output Media

The properties of the display must be considered when generating the abstraction. The display resolution, i.e. the height and width in pixels, is a significant parameter for minimum size and number of depictable objects. Connected with this is the dimension of the output medium, i.e. the area available for depicting the visualization. The depiction density, which is often specified in DPI (dots per inch), can be calculated from the resolution and the dimension. The density is a significant limitation for making objects distinguishable.

A further important criterion is the color ability of the output media. A significant parameter for this is the color capability, which can range from black and white and gray-scales to color tables and full color. If you want to ensure that the color sensation for the visualization is identical on different monitor types, parameters such as color rendition and gamma correction of the monitors have to be stored in the resource model.

A further criterion is the stereo ability or, more generally, the degree of immersion of the output medium. The effects of an immersive environment in GIS have hardly been studied. One can, however, assume that a realistic and less abstract depiction is sensible in an immersive environment with stereo projection. Closely connected with this are the system resources of the output device such as processor performance, graphic acceleration and central memory. These factors significantly influence the rendering speed, which is measured in polygons per second. If a high refresh rate is necessary in order to give the user the impression of real-time, the rendering speed significantly limits the depictable data volume.

Color ability, resolution and display dimensions are critical factors when using ultra-portable mobile end devices. When dealing with workstations and PCs, due to modern graphic cards that support rendering in hardware, significantly higher rendering speeds can be assumed. Creating a high degree of immersion requires special output media such as head mounted displays (HMD) or projections, which are beyond the metaphor of a window to the virtual world that is common in desktop computers.

Interaction

Contrary to static output media such as paper maps, electronic media are characterized by a high level of interactivity. These interaction possibilities are important in particular in explorative data analysis. By zooming in on a part of the depiction, the interplay between detail and overview can be realized very dynamically using an appropriate visualization. The interactive ability can be measured in different levels that build on one another:

- Variation of view: main functions are zoom in, zoom out, pan, rotate, change viewpoint and change 3D-viewing parameters
- Static changes to the visualization technique such as color palette and symbolization.
- Dynamic interventions in the visualization such as interactive positioning of particle clouds (particle tracing) in connection with a preceding simulation
- Control of the simulation process preceding the visualization

Network
A further limitation for graphical abstraction is the network bandwidth, which limits the response time behavior and the transferable data volume. A further influencing factor of the response time behavior is the complexity of the generation process of the abstraction. This should not be underestimated, as an abstraction is usually generated specifically for a user query. The use of offline calculations is only reasonable for often recurring standard queries.

In the following, a short recapitulation of the model of technical resources:

- Output media / display
    - Resolution: number of pixels in height and width
    - Dimension: height and width in cm
    - Color ability: bi-tone, gray-tone, color and color depth
    - Degree of immersion: window, stereo, immersive
- Rendering: polygons per second
- Degree of interaction
- Network: bits per second

User Model
Information on the user or target group can be used in order to adjust the abstraction to special, personal requirements. This personalization is desirable if the visualization is designed merely for personal use or for a limited user group. Output devices such as internet terminals and mobile devices in particular, enable individual access to a geo-data stock. Personalized visualizations should therefore be supported especially in these areas.

User models have some tradition in AI research and literature on this subject is correspondingly extensive. Allen has established a taxonomy that pinpoints the three main dimensions of user models [Alle90]. Every user model can be characterized by a position or an interval in the following three dimensions:

- canonical vs. individual model: does the system contain a model for a number of canonical users (i.e. stereotypes) or for each individual user?
- explicit vs. implicit model: are the user models explicitly described in the system design or are they derived from the users' behavior?
- long-term vs. short-term model: do the models rather describe long-term characteristics (e.g. abilities and knowledge) or more short-term characteristics?

An extensive treatment of the subject of user models in GIS and in graphical abstraction could fill an entire scientific study and exceeds the scope of this thesis. Here, the subject of user models in GIS will only be considered exemplarily and exclusively from the viewpoint of graphical abstraction. A suitable user model contains an individual user model. However, this is only possible on personalized end devices in which the user has created and shares a corresponding, explicit profile. An explicit interview for creating the user profile does not appear suitable, as users usually want to receive certain information as quickly as possible when they are looking for them in the internet. In this case, an explicit interview would be counterproductive and would scare off occasional users. An implicit model can be created during the explorative analysis of the database. Few long-term characteristics such as color perception and spatial perception are modeled. The user model rather contains short-term characteristics that can be derived from the search queries that are made.

Knowledge Level of the User

A generally applicable formula for the knowledge level cannot be defined. The parameters with which the knowledge level is modelled strongly depends on the application. In the abstraction of an urban environment, the knowledge level defines, for example, which features are used as landmarks. In this case stereotypes such as *native*, *frequent visitor, rare visitor* and *stranger* are suitable. Lange uses similar categories in landscape visualization [Lang99].

Krüger suggests that the knowledge level for dealing with graphical abstractions should also be modeled using the stereotypes *expert, layman, half-expert*.

The concentration ability of the human cognitive system is limited. Therefore, a graphical abstraction must consider these limitations by using clear focus structures. The two main influence factors for this are the viewing time and the viewer's background knowledge.

5.2.3 Creating the Graphical Abstraction

The generation of a graphical abstraction requires the assignment of dominance values of the features on the basis of the query that was made. For this, the dominance function is evaluated for each feature. The abstraction is carried out based on these dominance values. Of course technical and cognitive resources must be considered as well, besides the dominance values. In order to be able to consider technical resources, it is necessary to have metadata of the individual views of a feature that describe the technical requirements of the view, such as for example the data volume that needs to be transferred.

In consideration of these factors, suitable abstraction operators are selected that are then used for generating the visualization model.

View-Related Metadata

For each view of a feature, metadata is defined that is required for the selection of a view for given technical resources. In triangulated triangle meshes, the costs for visualization and for transferring a view are mainly determined by the number of triangles used. If compression techniques are used, the necessary compression and decompression times also have to be considered in the transfer time.

The degree of abstraction contains information on how abstractly a view depicts a feature. Here, geometric accuracy and detail level, but also the use of textures play a role. The categories proposed by Krüger *identifiable*, *classifiable* and *discriminable* are used to formally describe the degree of abstraction. For classifiable depictions, the type of classification is additionally specified, so that a feature can have several different classifiable depictions, which represent the feature in a mostly hierarchical semantic. For example, take the use of symbols that classify a building as a 5-star hotel, a hotel in general or as a building without specifying its use. However, the degree of abstraction of a view cannot be described independent of the user group for the semantic categories *identifiable* and *classifiable*. Even the use of symbols as they are common in map depictions may have a completely different meaning in different culture groups and thus lead to misunderstandings. Different background knowledge may also lead to a different understanding of classifiable depictions. Thus, for classifiable views, besides the type of classification, the user group for which this classification is understandable also has to be specified. In identifiable depictions, the familiarity of the feature usually plays an important role. If a user is familiar, for example, with a building, small details often make the difference between an identifiable and a classifiable depiction. These details are also called significant attributes and have to be maintained for identifiable depictions. Which attributes are significant often depends on the user and the familiarity of the user with the feature. For the formal specification of these influence factors, four user groups are formed: *altogether undiscerning, layman, half-expert, expert*. Conferred on an application in city tourism, these groups could also be called *stranger, rare visitor, frequent visitor* and *native*.

The degree of possible interaction also plays a significant role where two-dimensional depictions such as panorama images and so-called billboards are also permitted in addition to three-dimensional depictions. Billboards are textures that always face the viewer and thus create a three-dimensional impression although they are two-dimensional. The interaction with billboards is limited however, due to its two-dimensional form.

To recapitulate, the following metadata are required per view for selecting a suitable degree of abstraction:

- Costs for visualization
- Costs for data transfer
- Degree of abstraction
    - Identifiability by stereotypical viewer
    - Classifiability by stereotypical viewer
    - Discriminability
- Degree of interaction

If a desired view is not available, the possibility exists for generating a corresponding view online. For this, it is possible to maintain significant attributes that are of importance for the current abstraction. A procedure suitable for this was developed within the scope of this thesis [Coor01c] and is explained in section 5.2.4.

Selection of Suitable Views

Different factors play a role when selecting a view for a feature. The degree of abstraction is inversely proportional to the dominance of the feature. The more important the feature is as a query result, the less abstractly it should be displayed. Unimportant features that do not belong to the topical focus can be summarized. This aggregation is carried out using a spatial access structure. The complexity of the model in visualization is limited by the display abilities of the output device. If the network bandwidth is low, the total data volume that is to be transferred should also be small. In this case progressive visualization that allow incremental loading of the query result should be preferred.

Further important factors for the selection of the views are the size of the depiction object on the screen and the visibility of the feature. Even very important features do not have to be depicted in a complex form if they are hardly visible on the display.

Using these criteria, which will be formalized in the following, the views used for the graphical abstraction of the query result are based on the focus structure that defines the importance of individual features. This requires that a decision is made on whether a view may be aggregated or not, depending on the dominance of a feature. Additionally, for interactive exploration of the scene, one has to decide on when a view may be refined or an aggregation may be broken up. The main criterion for this is the distance of the view to the viewpoint of the viewer and connected with this, the size of the view on the screen. The result of this graphical abstraction is managed in the P-Tree, which was developed specially for this purpose [CoSc99] and [Coor01c]. In particular, the P-Tree supports incremental loading of the abstraction result in a distributed environment.

A decision function *degreeOfAbstraction* derives a suitable degree of abstraction Γ_f for depicting a feature *f* from the dominance value dom_f of the feature.

$$degreeOfAbstraction\colon dom_f \rightarrow \Gamma_f$$

This can take place, for example using threshold values. However, for the decision function, it must be ensured that features with a high relevance factor regarding a query are also assigned with a *classifiable* depiction corresponding to this query. In general, reference objects should be assigned with *identifiable* depictions. Krüger and Butz suggest a discriminable depiction in general for landmarks as background objects, as long as they are not within the topical focus [KrBu97], but this seems too restrictive for visualizations in urban environments. A decision function should, however, ensure that landmarks are assigned with a *discriminable* depiction at the least. In many situations, for example in routing, a semantic depiction category for landmarks is also reasonable. For features with a dominance value of 0, the degree of abstraction *visible* will usually suffice.

For example, for the query "show all hotels within a radius of 2 km of the Darmstadt Castle with more than 50 rooms", a suitable function would assign a view to the corresponding features that depicts them so that they are *classifiable* as a hotel. The Darmstadt Castle as a reference object should be assigned with a *identifiable* depiction. Any existing landmarks can be correspondingly *classifiable*, so that they can be used for a rough orintation.

Based on the selected degree of abstraction, a concrete depiction must be found for the feature. This, requires the introduction of a further decision function *chooseView*, which assigns a suitable view v_f to a feature *f* depending on the degree of abstraction Γ_f. However, the selection of a suitable view also depends on the size and position of the depiction on the output device. In an interactive three-dimensional visualization, this directly depends on the viewpoint *vp* from which the scene is being viewed. The choice of this viewpoint is usually not specified by the user, but defined by the generating system itself. A further influencing factor for the selection of a view is the desired degree of immersion G_I, which, however, is limited by the rendering speed of the visualization system.

$$chooseView\colon \Gamma_f \times vp \times G_I \rightarrow v_f$$

For this decision function, it must be ensured that the desired degree of abstraction Γ_f is identical to the degree of abstraction Γ_v of the view v_f, i.e. $\Gamma_f = \Gamma_v$. An exception applies to features with the degree of abstraction *visible*. This degree of abstraction does not occur in views, as a view that is assigned to feature always represents this feature at least as *discriminable*. Features with the degree of abstraction *visible* can be assigned with discriminable views.

The result of this selection of views is the targeted visualization result. Here, conflicts may occur both with the technical resources, such as available bandwidth, as well as with the depiction principles, such as the rule of readability. These conflicts can be solved using two abstraction techniques, aggregation and emphasis.

Using aggregation, views are summarized according to certain criteria such as spatial proximity and similar semantics of the features. Aggregation supports depiction principles such as readability by depicting fewer discriminable objects and thus causing the viewer to focus on important areas of the scene. Usually, aggregation also leads to a reduction in data volume, so that technical boundary conditions can also be fulfilled. Aggregation should only be applied to the view of those features whose degree of abstraction is *visible*. There are, however, situations, in which this is not sufficient for solving all conflicts. In this case, features with other degrees of abstraction must also be aggregated, whereby this is carried out in the order of ascending dominance values. The lower the dominance value of a feature, the more likely it is to be aggregated. Features that serve as landmarks or reference objects should not be aggregated under any circumstance, as orientation in the depiction would otherwise become very difficult. A method for aggregation is explained in the following section.

In an interactive three-dimensional visualization, it is possible that dominant features are not visible. In a realistic, perspective visualization, this can occur due to the distance between the viewpoint and the feature, because the feature takes up too little space on the output device or because it is hidden behind other features or aggregated depictions. A perspective that is unusual for the viewer can also cause an identifiable depiction to lose its meaning. In order to solve these conflicts, important features can be depicted over-proportionally, similar to highways in a road atlas, so that they remain classifiable or identifiable even from a distant viewpoint. Following Michael Ende, a corresponding view is also called 'Scheinriese' (apparent giant). If the *view* is far away from the viewer, then it is depicted as large, and the closer it gets, the smaller it becomes, until it has shrunk to its normal size. An alternative is the use of several perspectives so that, although the depiction no longer corresponds to reality, it offers a larger information content. Fig. 5-11, for example, shows a different perspective depiction of the Darmstadt Archive, in order to make it identifiable even in a view from above. For this, the roof alone is not enough. The depiction geometries that are used for the emphasis are characterized by the fact that they are scaled, transformed and deformed depending on the viewpoint.

Results of the Graphical Abstraction

The calculation of dominance values will be clarified using a three-dimensional route visualization. For visualizing the route, for example the path from the main station to the Fraunhofer Institute for Computer Graphics Research (IGD) in Darmstadt, all buildings along the desired route are extracted using a spatial query. Additionally, all features along this route are assigned with a relevance of r_f=0. The remaining features are not relevant for the query (r_f=1) and therefore do not contribute to the dominance value *dom* of the corresponding feature. The reference objects *O* of the query are starting and end point of the route, in this case the main station and the Fraunhofer IGD. The dominance values of

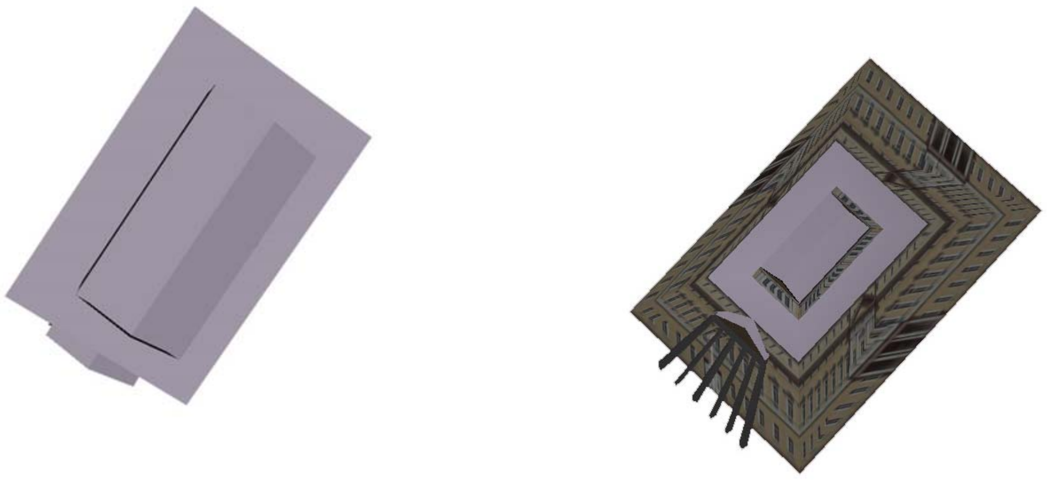


Fig. 5-11: Emphasis by changed perspective, e.g. for a view from above. In this example, the roof shapes alone are not enough for making this feature identifiable.

these two features are increased correspondingly. Prominent buildings in Darmstadt such as the Castle, the Archive and conspicuous buildings along the route are considered as landmarks. The landmarks receive an increased dominance value using the addend L. In route visualization, landmarks are valuated considerably more highly by the weighting factor a_2 than the relevance R ($a_0 << a_2$).

The dominance of the individual features is reflected in the depiction by transparency and level of detail. Features with a dominance *dom=0* are not displayed. Features with a low dominance are visualized single-colored and semi-transparent, while features with a high dominance value are depicted textured in order to achieve visual identification. Fig. 5-12 shows individual sequences of the corresponding route visualization. The high weighting of landmarks leads to the fact that features such as the Archive in fig. 5-12 (third image) have a high dominance, although they are not located directly along the route (r_f=1).

5.2.4 Aggregation Procedure

In this section, a procedure for aggregation is proposed that is based on a spatial access structure, which at the same time can be used for efficient answering to spatial queries. Based on this hierarchical index, nodes are assigned with depiction geometries of simple and aggregated features. According to certain criteria, which were discussed in the previous section, views of a subtree can be summarized and represented by an aggregated depiction geometry in the root of this subtree. The resulting tree then contains the graphical abstraction of the query result that is to be visualized and, at the same time, can also be used to progressively transfer the scene that is to be depicted in a network-based environment. Due to this property, the resulting data structure is also called tree for progressive transfer, or *P-Tree* for short. In this section, the structure of this tree will be detailed. The

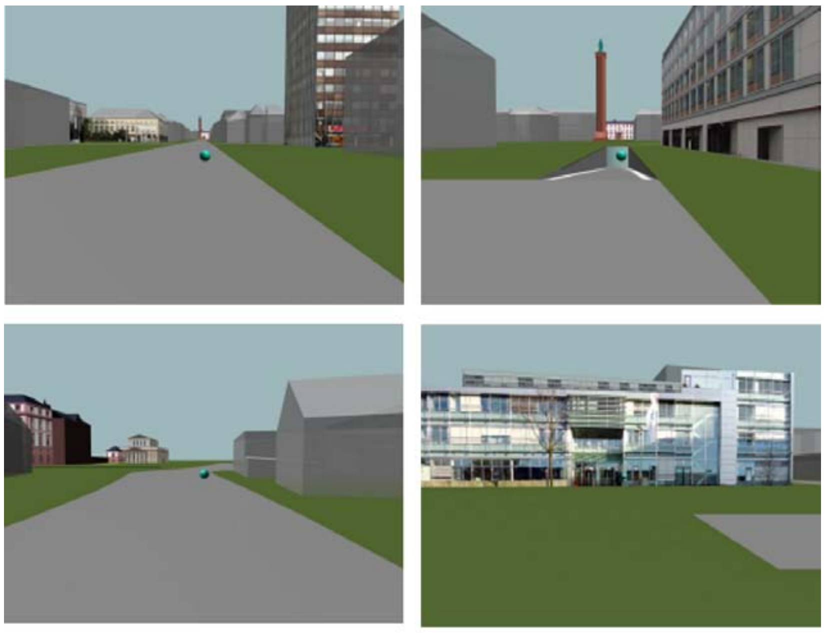


Fig. 5-12: Sequences from a three-dimensional directions sketch to the Fraunhofer IGD Darmstadt. Navigation landmarks are visualized in detail, while buildings with low dominance are displayed in gray and semi-transparent [Coor02b].

following chapter is dedicated to the data transfer and, besides the progressive transfer, also deals with geometry compression.

Data Structure P-Tree

The P-Tree is based on the spatial access structure R*-Tree, a variation on the R-Tree according to [Gutt84]. The R*-Tree is, as was experimentally shown in [Brin93] and [Köse97], significantly more efficient than the original R-Tree for searches for spatial objects. The R*-Tree also offers advantages as a basis for graphical abstraction, as overlaps of the envelope rectangles are minimized, which usually leads to the fact that the objects of a subtree are spatially not very far apart. It is therefore justifiable to also merge spatial objects of the same subtree in aggregation.

The P-Tree has all the properties of the R*-Tree and extends this structure with depiction geometries on all levels of the tree. It is thus a hierarchy of nested *d*-dimensional intervals. Traditionally, two-dimensional intervals are used in R-Trees, an extension to the third dimension is possible without problems [Schu99]. A node k_l of the level *l* of the P-tree consists of *n* entries e_i. Each entry *e* consists of a content *c(e)*, a three-dimensional interval *q(e)* and a graphical representation of the subtree *v(e)* below this point. If *e* is an internal node of the of the P-Tree, then *c(e)* is a reference to a node k_{l-1} on the next-lower level of the tree. The interval *q(e)* is the minimum surrounding axis-parallel envelope cuboid that encloses the intervals of all entries of the child node *c(e)*. A leaf node contains entries of the same type, but the reference now points to a geometry elements of the database. For three-dimensional spatial objects, this is a *body*. The interval thus is the minimum surrounding envelope cuboid of this *body*. In the data model, the geometry of an elementary feature is described by exactly one *body* primitive. For the graphical representation *v(e)* of this node, the depiction geometry v_f selected by the decision function *chooseView* is therefore used.

The properties of the R*-Tree are maintained even in the P-Tree (cf. [Köse97]):

- The root has at least two sons, as long as it is not a leaf node.
- Each node contains between m and M entries, as long as it is not a root.
- The tree is a height-balanced multipath tree, i.e. all leaves are on the same level of the P-Tree and thus have the same distance to the root.
- Every envelope cuboid of an internal node encompasses all envelope cuboids of its sons. The envelope cuboid is the minimum surrounding axis-parallel envelope cuboid. The elements in the P-Tree are summarized in such a way that the overlaps of the envelope cuboids are minimized.

The maximum number *M* of entries in a node is also called node capacity. For the minimum number m of entries per node, $2 \leq m \leq M$ applies. The utilization of an internal node thus always amounts to at least m/M and the height of the P-Tree grows logarithmically with the number of entries. The P-Tree is completely dynamic, i.e. insertion and deletion operations are identical to those in the R*-Tree. As graphical representations for internal nodes are specifically generated for a query result, these do not have to be specially considered for modifications of the tree. For clarification, fig. 5-13 shows a three-dimensional P-Tree in which the graphical representations in the internal nodes are not displayed.

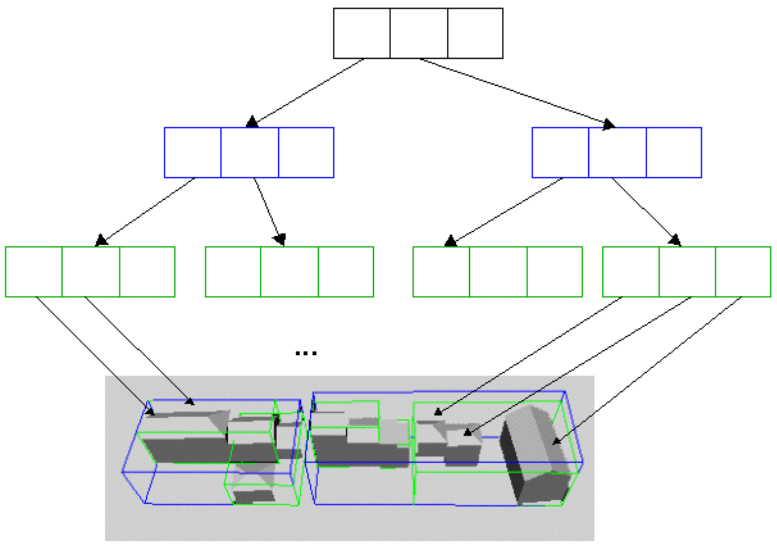


Fig. 5-13: P-Tree as spatial index

Aggregation Techniques on the P-Tree

After the basic structure of the P-Tree has been described, this section explains how the P-Tree can be used for aggregation. Depiction geometries of elementary features are stored in the leaf nodes of the P-Tree. Using certain criteria such as the degree of abstraction of the features, a function *doAggregate* decides, whether several views should be aggregated. If the decision is positive, the corresponding views are summarized using the method *amalgamate*. For this, the method *amalgamate* creates a new depiction geometry that merges two views. By applying this method recursively, all views of the children of an internal node k_l in the P-Tree can be aggregated. The resulting view is registered as the depiction geometry of the corresponding entry *v(e)* of the parent node k_l.

What remains is a clarification of how the merging of different views is carried out. The goal of the *amalgamate* method is to reduce the complexity of two depiction geometries. In principle, a view according to section 5.1.4 can contain a signature or a mesh. In the following, the analysis of aggregation techniques will be restricted to meshes. The complexity of a depiction geometry in this context is given by the number of triangles used in the meshes. Numerous procedures are known for this in computer graphics (cf. [HeGa97]). In general, these are procedures that are based on syntactic criteria such as geometry and surface properties. Important simplification operators for this are, on one hand, the summarization of spatially proximate vertices (vertex clusters). A second operator, which is almost exclusively used in modern simplification procedures, is the edge collapse. For this, an edge with its delimiting vertices is replaced by one vertex. The advantage of edge collapse lies in the fact that this operation is reversible. This inverse function, called vertex split, allows progressive storage of the depiction geometry. A detailed description of these methods can be found in section 2.5.2.

Simplification procedures are very well-suited for reducing the complexity of the depiction of a single feature. They can be used, for example, for generating a simple view from a detailed geometry description of a feature. In many applications however, such as 3D land registers, a large part of the features have a quite simple, little detailed geometry. The goal of a 3D land register is to make an area-wide model available at a justifiable price. Therefore, only (semi)automatic acquisition methods can be used, which, at least not for the time being, are not able to create a detailed geometry of all the buildings in a city. In spite of this, and for different reasons, the depiction of a large part of such a model requires the aggregation of individual features. Simplification procedures based on edge collapse are usually topology-conserving. This is not desirable for an aggregation, as in particular not joined components always remain disjoint after edge collapse. Merging in the actual sense of the word does not take place.

Garland and Heckbert have proposed a procedure that allows the collapse of even non-existing edges [GaHe97]. For this, vertex pairs are connected by temporary edges according to criteria such as spatial proximity. During simplification, the temporary edges are dealt with as normal edges. After the simplification process, temporary edges that were not collapsed are removed again. This procedure allows merging of disjoint components. However, when temporary edges are collapsed, internal faces may be created that no longer belong to the surface face depiction. These faces are not required for depiction and can therefore be removed. It is, however, very costly to determine these faces, which has a negative effect on the running time of the procedure [GaHe98].

An alternative approach is the use of envelope elements. In this way, the convex envelope, for example, can be used as the abstraction of two views. The calculation of the convex envelope, however, needs to be carried out for a large number of points. Such a procedure should therefore only be used for abstracting simple views. Even euclidian elements are suitable envelopes for abstraction. Especially in the urban area, many features can be approximated well using euclidian elements. If only envelope cuboids are used in the aggregation, these are identical to the intervals of the P-Tree. This property can be utilized if the query result needs to be depicted as quickly as possible, as is often the case for web-based applications. When the entries are to be aggregated, instead of generating an abstraction, the intervals can initially be used as depiction geometry. Crosnier and Rossingnac propose a tribox as a tighter bound as the envelope cuboid but also cheap to calculate and to display [CrRo99].

This type of aggregation is used by Zlatanova and generally summarizes all features or subtrees [Zlat00]. This, however, leads to the fact that landmarks do not remain visible and that orientation is quickly lost in a bird’s eye view. In abstraction, it should be generally avoided to summarize or hide landmarks or reference objects behind aggregated depiction geometries.

Fig. 5-14 shows the different methods for merging two views using the example of two buildings. The QSlim software by Garland was used to generate the simplification using edge collapse. The difficulty herein lies with the definition of a suitable abort condition. Edge collapse was carried out until the number of faces had been reduced by 50%. In spite of temporary edges, aggregation is only carried out once the building on the right has already been reduced to such a degree that it is no longer classifiable as a building. The use of the convex envelope leads to significantly improved abstraction results. The disadvantage with the use of the convex envelope is that the number of triangles used is not controllable. If a maximum number of triangles is to be observed for depiction, the convex envelope can be additionally simplified using edge collapse. The use of envelope cuboids in the prototypical realization is shown in section 6.3 using an area-wide city model of Darmstadt.

Simplification With Consideration of the Semantics

The procedures presented up to now work according to syntactic criteria such as geometry and surface attributes. Semantic criteria such as the topical focus cannot be considered. But for the graphical abstraction of a feature’s depiction geometry, it is absolutely necessary that these semantic criteria are considered. This shall be clarified using a model of the Darmstadt Bell Tower building. This model is an object composed of different features, as depicted in fig. 5-15. The depiction geometry for the entire Bell Tower building is very detailed and comprises about 100,000 triangles. Just as the feature itself, the depiction geometry is also organized hierarchically. If the Bell Tower building is to be de-

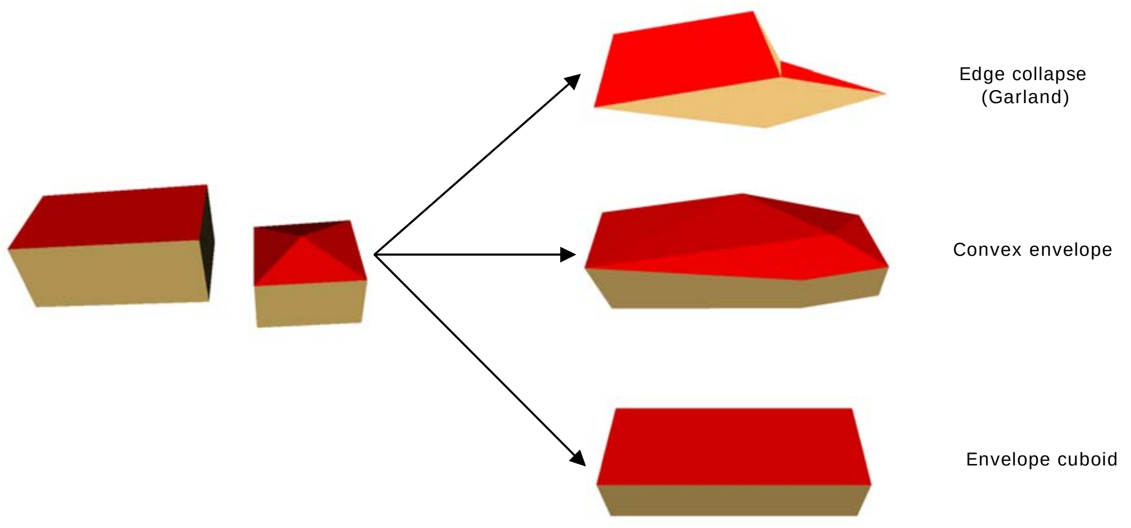


Fig. 5-14: P-Tree aggregation: comparison of different methods

picted identifiably as a landmark or as a reference object, then the use of this view is surely not suitable. On one hand, a quite significant data volume is required, which, depending on the context of the application, may also have to be transferred. On the other hand, the detailed depiction of the Bell Tower attracts the viewer's focus to this building. This entails the danger of distracting the viewer's attention from the topical focus of the entire depiction.

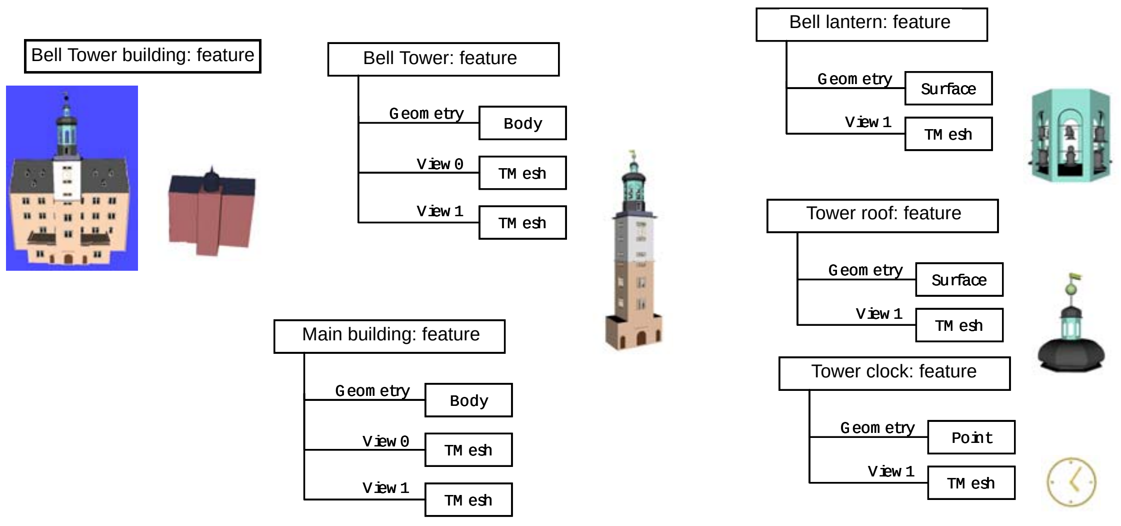


Fig. 5-15: Data model Bell Tower building

The different components of the Bell Tower building are of different importance for an *identifiable* depiction. A significant characteristic of the Bell Tower building is the bell lantern. The detailed depiction of this component alone is usually sufficient for identifying the Bell Tower building. In a graphical abstraction, it is therefore sufficient to depict the bell lantern as *identifiable*, while the remaining components should only be kept *visible*, in order to ensure the context. In order to generate such an abstraction automatically, this knowledge about the significance of the individual components has to be used. In principle, two different approaches are possible for this.

On one hand, the individual components or depiction geometries can be simplified independent from one another according to their importance. This approach is used, for example by Martin, but leads to steps in the model, as the components do not fit together seamlessly anymore after simplification [Mart00]. In order to avoid this, Hubeli and Gross introduce so-called feature edges, which may not be modified during the simplification process [HuGr00]. However, in particular when dealing with many small components, this unnecessarily restricts the attainable degree of abstraction.

The approach favored in this thesis does not work on the basis of individual components, but considers the entire depiction geometry as a unit. The meaning of the individual components is considered as an additional face attribute. Initially, a valid triangle mesh is created from the different components, in which double vertices and edges are eliminated. This avoids cracks from being produced between the individual components during simplification. The faces of the resulting mesh are assigned with dominance values corresponding to their importance, which can be derived from the individual components' desired degree of abstraction. This additional face attribute is then considered during the actual simplification procedure. In edge collapse, for example, edges that delimit triangles with a high dominance are assigned with high costs. Such a procedure will be presented in the following. Like the approach of Garland and Heckbert [GaHe98], this procedure is based on a square error matrix for determining the costs of an edge collapse.

Starting point for the procedure is a mesh M(V, T, A_V, A_T) that is described by a set V of vertices, a set T or triangles and the attribute sets A_V and A_T. The set A_V contains vertex-related, A_T correspondingly triangle-related attributes. The dominance value of a component is assigned to the corresponding areas in the mesh as a triangle attribute. The algorithm for the mesh abstraction with consideration of semantic properties then is as follows:

- Calculate the matrix Q for the square error for all initial vertices (see below). Assign each vertex v with a dominance value v_d. Here it has to be ensured that $v_d \geq 0$, whereby $v_d=0$ means that v does not delimit a triangle that is part of the representation of a relevant feature.
- Select all valid vertex pairs as candidates for a collapse. A vertex pair (v_1, v_2) is valid when v_1 and v_2 are connected to one another by an edge or when the distance dist(v_1, v_2) is smaller than a given threshold value t.

- Calculate a representative vertex v for each valid vertex pair (v_1, v_2). When the vertex pair is removed from the mesh, v_1 und v_2 coincide in v. The error $Q_1 + Q_2$ of this vertex defines the costs of the corresponding edge collapse. These costs are increased with the weighting $a{*}v_d$, which reflects the semantic relevance of this vertex.
- Iteratively remove the vertex pair with the lowest costs from the heap, carry out the corresponding edge collapse and recalculate the costs of all neighboring vertices.

The initial matrix Q is calculated for each vertex on the basis of a heuristic according to [RoRo96], which characterizes the geometric error. Each vertex is assigned with a set of levels. The error $\Delta(v)$ of a vertex regarding the levels is calculated using the square distance of the vertex to these levels, whereby the level $p = [n_x n_y n_z d]^T$ corresponds to the normal form according to Hesse $xn + d = 0, \|n\| = 1$.

$$\Delta(v) = \Delta([v_x v_y v_z 1]^T) = \sum_p (p^T v)^2$$

The set of levels at a vertex is defined by the number of triangles that have this vertex in common.

Fig. 5-16 shows the implementation of this procedure using the abstraction of the Darmstadt Bell Tower building. The bell lantern serves as the significant feature. In this model, only edge collapses between vertices that are connected by an edge were permitted. This avoids the window openings from being closed. In the second example, the model of a cow was used. So that there are no misunderstandings, the cow should remain identifiable as a cow. For this, the udder was specified as the significant component. Even in strong reductions, this feature remains clearly visible. The results of this abstraction are depicted in fig. 5-17.

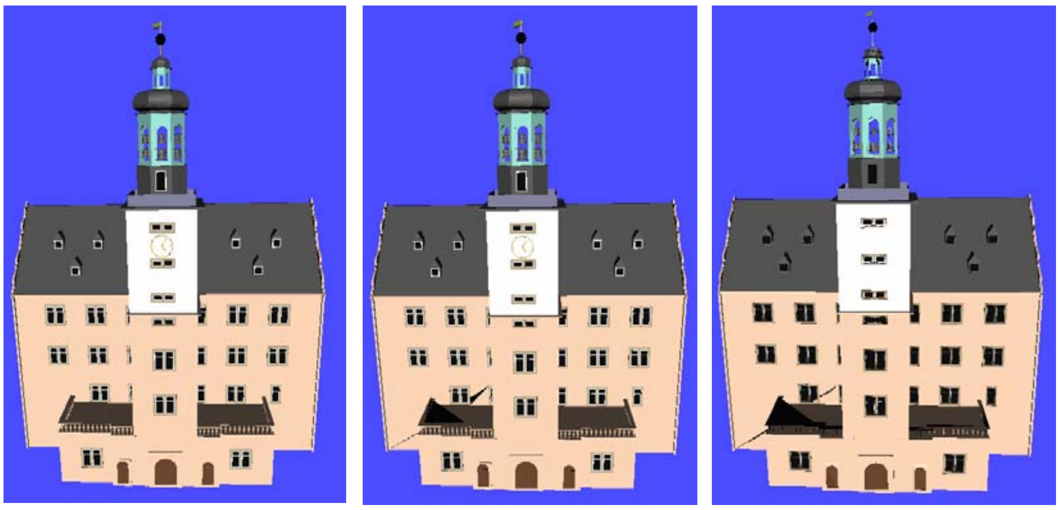


Fig. 5-16: Abstraction of the Darmstadt Bell Tower building

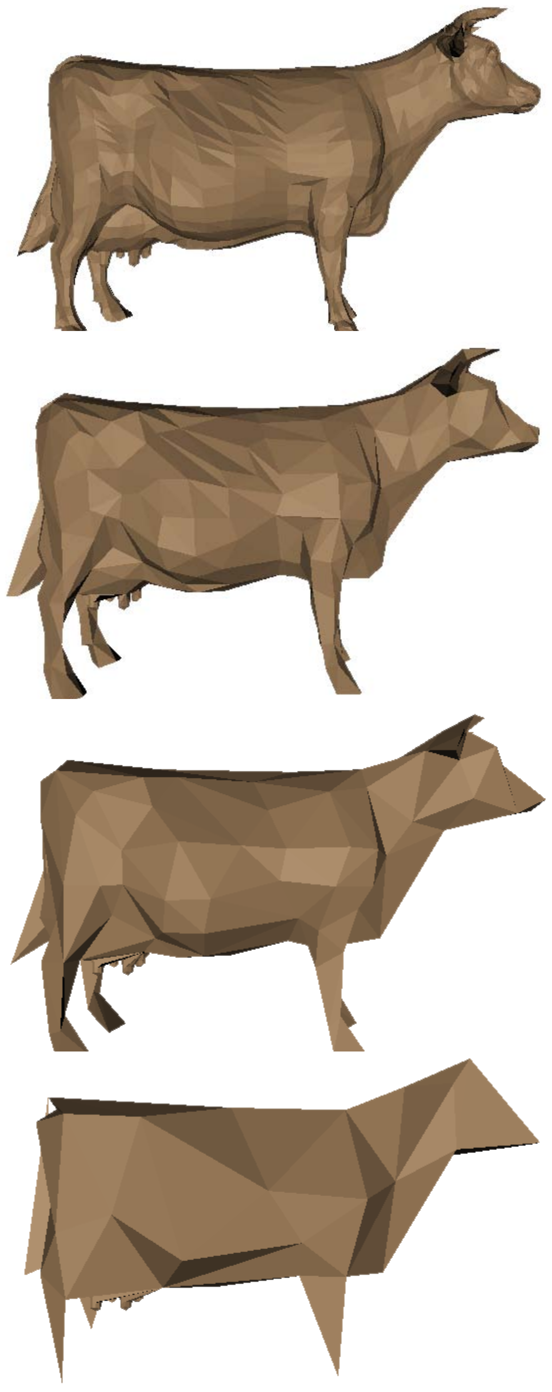


Fig. 5-17: Abstraction of the model of a cow. The udder was maintained during abstraction as a significant attribute. The original model has 5804 triangles, the simplified models have 1000, 532 and 300 triangles respectively.

Estimation of Applicability of the P-Tree

In the following, the data volume of a 3D city model that is created by hierarchic aggregation of the building geometries is to be estimated. It is assumed that each aggregation level in the P-Tree reduces the number of triangles by 50%. Each internal node summarizes between m and M depiction geometries and depicts them using a mesh with 50% fewer triangles.

Let *t* be the average number of triangles that are used for representing the spatial characteristics *g* of a feature *f*. Let *n* be the number of features that are administered in the P-Tree. Then *t* is equal to the average value of the triangles per feature representation *f.g*:

$$t = \frac{1}{n}\sum_{i=1}^{n} \#triangles(f_i.g)$$

The leaf nodes of the P-Tree are located on level 0, the root on level *h-1*. A depiction geometry V^0 on the level 0 (leaf node) then represents between *m* and *M* features. The number of triangles used for this can be estimated as follows:

$$\frac{m}{2}t \le \#t(V^0) \le \frac{M}{2}t$$

For this, an average complexity of *t* triangles per feature is assumed.

The number of triangles of a depiction geometry V^l on a level $l>0$, which represent the corresponding subtree, is given by:

$$\#t(V^l) = \frac{1}{2}\sum_{i=1}^{c} \#t(V^{l-1}) \text{ , } c \in [m,M]$$

The number of nodes k^l per level *l* at a height *h* of the P-Tree can be estimated by:

$$k^l \in \left[\left\lceil \frac{k^{l-1}}{M} \right\rceil, \left\lceil \frac{k^{l-1}}{m} \right\rceil\right] \text{ with } k^{-1} = n$$

For the number *k* of nodes in the tree, the following thus applies:

$$k = \sum_{i=0}^{h-1} k^i$$

In the fully occupied tree, $h = log_M n$ or $n=M^h$ applies, and furthermore, for the number of nodes k^l of a level *l* (l=0,...h-1):

$$k^l = \frac{k^{l-1}}{M} = \frac{k^{l-2}}{M^2} = \dots = \frac{k^{l-l-1}}{M^{l+1}} = \frac{n}{M^{l+1}} = M^{h-l-1}$$

The number of nodes in the fully occupied tree is then given by:

$$k = \sum_{i=0}^{h-1} k^i = \sum_{i=0}^{h-1} \frac{n}{M^{i+1}} = \sum_{i=0}^{h-1} \frac{M^h}{M^i} = \sum_{i=0}^{h-1} M^{h-i} = \sum_{i=0}^{h-1} M^i = \frac{M^h - 1}{M - 1}$$

At an average of *t* triangles per feature, the number of triangles of a depiction geometry *V* on the level *l* in the fully occupied P-Tree amounts to:

$$\#t(V^l) = \frac{1}{2} \sum_{i=0}^{M} \#t(V^{l-1}) = t\left(\frac{M}{2}\right)^{l+1}$$

Thus, for the depiction geometries additionally used in the P-Tree, the following number of triangles is necessary:

$$\#t(P\text{-}Tree) = \sum_{l=0}^{h-1} \#t(V^l)k^l = t\sum_{l=0}^{h-1} \left(\frac{M}{2}\right)^{l+1} M^{h-l-1} = tM^h \sum_{l=0}^{h-1} 2^{-l-1} = tn(1-2^{-h})$$

Thus the data volume caused by the additional depiction geometries in the P-Tree is approximately doubled. So, if all depiction geometries, starting from the root of the P-Tree to the level 0, are transferred, one could also have transferred the depiction geometries of the features in the same time. The use of the P-Tree therefore only makes sense when it can be assumed that many areas of the model are not going to be resolved down to the most detailed depiction geometry, but remain on a high aggregation level.

However, if, for aggregation, an implicit model is used as the depiction geometry in the P-Tree, the additional transfer volume does not apply. An example for this implicit model are the envelope cuboids of the corresponding subtree. This information is given without any additional geometric model.

5.3 Adaptive Transfer in a Network

In the previous section, with the P-Tree, a data structure was introduced that combines the concept of a spatial index for effective access to spatial data with the concept of hierarchical visualization geometry. This data structure was developed with the goal of using geometry data from a 3D GIS in a heterogeneous, distributed environment. In the following chapter, the actual transfer of the data within the network is discussed. Since prior knowledge on the existing client hardware and the bandwidth of the network connection is usually not available, particular emphasis has to be laid on resource adaptivity when designing a 3D GIS server. Thus, besides the presentation goal, the technical resources of the output system and of the network also have to be taken into account. For reasons of scalability, the current server load should also be taken into account.

5.3.1 Transfer of the P-Tree in the Network

In this section, different transfer strategies for a P-Tree are worked out and discussed. It is assumed that the P-Tree is completely built, i.e. suitable views for internal and external nodes of the tree were already generated using the abstraction procedure discussed in the previous chapter. The task at hand is to transfer and render the P-Tree within a network as efficiently as possible. This process is mainly influenced by the technical resources within the network. This comprises all aspects that are of significance to the entire process of data transfer and to the interactive visual presentation of the 3D geometry. Correspondingly, the technical limitations are divided into the subclasses

- resources of the output medium
- resources of the transfer medium

The possibilities for depicting a scene that is administered in a P-Tree vary from complete server-side rendering to complete transfer and client-side rendering.

Server-Side Rendering

In this alternative, the P-Tree is entirely rendered on the server side. Only an image as the rendering result is transferred. The viewer's position and view direction, as well as the display resolution, must be transferred to the server. On the basis of these parameters, the P-Tree is rendered to an image. This image is transferred to the client, whereby additional compression of the image is possible. The costs t_F for this process result from the time required for rendering an image t_{RS}, time for compression t_C and decompression t_{DC} and transfer t_T of the resulting image. For an interactive exploration of the scene, a frame rate of about 10 fps (frames per second) should be achieved. The rendering and transfer process per image may therefore not exceed 0.1s.

$$t_F = t_{RS} + t_C + t_T + t_{DC} <= 0.1\ s$$

The rendering costs depend on the complexity of the scene that is to be depicted and on the technical resources of the server. The transfer costs result from the image size s_I and the available bandwidth b of the network. s_I is determined by the resolution and color depth of the output medium. Thus, the transfer costs are given by $t_T = s_I / b$. Using compression, the image size can be reduced to $s_C << s_I$, which correspondingly reduces the transfer costs. Additionally, however, costs t_C for compression and t_{DC} for decompression arise. Image compression is therefore only economical if the cost $t_C + t_{DC}$ for compression is smaller than the saving in data transfer $(s_I - s_C) / b$:

$$t_C + t_{DC} < \frac{s_I - s_C}{b}$$

This procedure will be clarified using an example. A three-dimensional city map is to be used for navigation support on a mobile end device with a 200 x 300 pixel display and 8 bit color depth. The processing of the corresponding query, the building of the P-Tree and the entire rendering is carried out on the 3D geodata server. With a high-performance server, one can assume that the rendering costs for an image amount to about $t_{RS} = 0.04$ s. As the result, a 200x300x8 bit = 60 KB large image is sent back to the mobile end device. With a bandwidth of 11Mbit/s in WLAN, the costs for transfer amount to t_T =0.044 s. Thus, the total costs for rendering and transfer without compression amount to $t_F= t_{RS}+ t_T$=0.084 s, which corresponds to a frame rate of about 12 fps. With a smaller bandwidth of 384 Kbit/s using UMTS, the costs are significantly higher and amount to t_T =1.25 s. For an uncompressed image, total costs amounting to t_F = 1.65 s or an frame rate of 0.6 fps result.

Server-side rendering is well-suited for end devices, which cannot achieve high frame rates in rendering, for example due to lack of hardware support. For server-side rendering, a high image frequency on the server side and a large bandwidth for transferring the result are necessary, as each individual frame has to be transferred to the client. If the available bandwidth is not sufficient, the images can be compressed as single images or , as proposed in [Cohe01], as a video stream using layering techniques to yield a lighter stream. An alternative would be a progressive procedure in which rendering is carried out with a low resolution as long as the user is in motion. Once the user stops in one location, the scene is rendered with a higher resolution, whereby the rough image only has to be refined.

Complete Transfer of the P-Tree

Another extreme alternative for the rendering is the complete transfer of the P-Tree to the client. For this, the entire database is initially transferred and then rendered locally. This procedure is common, for example, in virtual reality but also for small scenes in the WWW, as this allows direct response to user interactions. For example, aggregations can be disintegrated and individual buildings can be depicted without delay when the viewer comes closer. All data required for this are available in the client memory.

The costs for the complete transfer of the P-Tree are the costs of the initial transfer t_I. The costs for the initial transfer are composed of the writing of the P-Tree t_W, the actual transfer t_T and the renewed parsing of the structure t_P. If the P-Tree is compressed, additional costs for compression t_C and decompression t_{DC} arise.

$$t_I = t_W + t_C + t_T + t_{DC} + t_P$$

In the interactive exploration of the scene, no further costs arise from data transfer, as all required data is available locally. The frame rate thus only depends on the client's resources. The costs t_F for rendering an image correspond to the time t_{RC} that is required for locally rendering the P-Tree:

$$t_F = t_{RC}$$

The disadvantage of this procedure is the initial transfer of the entire P-Tree. The data volume of an area-wide city model, for example, is surely not suitable for this form of data transfer. On one hand the waiting time for the user increases drastically, and on the other hand, in most cases, data is transferred that is never depicted because the user does not enter the corresponding areas. A further disadvantage in the transfer of the complete model is the high requirements made to the client's memory capacity. Especially on mobile end devices with a memory capacity of currently 8 to 64 MB, this can be very problemtaic.

Progressive Transfer and Load-on-Demand

In data transfer according to the load-on-demand paradigm, only data that is required for the visualization of the current situation are transferred. Related to the P-Tree, this paradigm means that only a part of the P-Tree is stored on the client side. This partial P-Tree is selected from the P-Tree by the server, in dependence on viewpoint and view direction, and then transferred to the client. Interactive exploration in the database, but also new user queries, cause the partial subtree to be supplemented with new nodes. The costs for this type of data transfer are on one hand the initial transfer costs. Besides the costs for writing t_{Wp}, transferring t_T and parsing t_{Pp} the partial P-Tree, an additional cost t_S arises from the selection of the data that are to be transferred. In very large models, however, using suitable selection, both the costs for the data transfer as well as the local rendering costs are much lower than they would be for the complete transfer of the P-Tree.

$$t_I = t_S + t_{Wp} + t_C + t_T + t_{DC} + t_{Pp}$$

The selection of the partial P-Tree has to be carried out efficiently, in order to be able to give the user a first feedback as quickly as possible. All objects visible from the current location should be contained in the P-Tree. An efficient decision function is required for this.

In the load-on-demand procedure, when exploring the database, the partial P-Tree needs to be supplemented. The costs for this so-called refinement of the local P-Tree t_r are similar to the initial transfer, only the costs t_S, for selecting the data that are to be transferred and the data volume that is to be transferred are different. As data is already available locally, only supplements to the data stock have to be sent.

The disadvantage of the load-on-demand concept is the generally longer waiting time, when refinement data have to be loaded. Especially when the view direction is changed drastically, many objects that were not visible before suddenly have to be depicted. In order to solve this problem, a strategy is required that foresightedly loads data, which, with a high probability, will soon be located in the viewer's field of vision. This foresighted loading of potentially relevant data is also called *prefetching*. Corresponding predictions can be made, for example based on the interactions made up to that point. Another *prefetching* strategy is to load all data within range of the user, independent of whether they are visible or not. Important features that are not in the field of vision should also be transferred using *prefetching,* so that a rough scene for orientation can immediately be offered in interactions, in particular when rotations are carried out. Algorithms for selecting the initial partial P-Tree and for prefetching are presented in the following section.

Selecting the Initial Partial P-Tree

In the selection of the initial partial P-Tree, initially all depiction objects are sought that are located within a given view pyramid. In dependence on visibility, distance to the viewpoint and dominance of the features represented in the corresponding view, nodes of the P-Tree are selected for transfer. For this, the P-Tree is traversed as in the algorithm described in the following. Starting from the root, for all entries e_i of an internal node, it is decided whether the interval $q(e_i)$ intersects with the view pyramid. If this is the case, depending on the distance of the interval $q(e_i)$ and the dominance of the view $v(e_i).dom$, the view is included in the partial P-Tree or the selection is applied recursively on the child nodes $c(e_i)$ of this entry. Once all visible depiction geometries have been selected by this procedure in a suitable abstraction, a *prefetching* for currently not visible depiction geometries is carried out. This takes place in dependence on distance and dominance. Depending on the transfer strategy, the partial P-Tree can be selected first and then be transferred to the client in its entirety. Alternatively, the depiction geometries can be transferred progressively by immediately sending each selection that is made to the client. The partial P-Tree is then rebuilt by the client. The advantage of this progressive transfer is that visual feedback to the user takes place as quickly as possible.

5.3.2 Compression of the Depiction Geometry

For transferring the depiction geometry, it is usually reasonable to reduce the data volume by compression. A general procedure for compression is based on the LZW algorithm, which uses the different frequencies of characters for efficiently storing a text, for example. This algorithm is used in gzip, for example. As we are dealing with triangle meshes in depiction geometries, significantly higher compression rates can be achieved using special compression procedures for three-dimensional geometries, such as topological surgery (see section 2.4.2), which take into account the inner structure of a mesh. The data volume that needs to transferred for a mesh can thus be reduced by about 95%.

The compression procedure developed in this thesis is based on the Edgebreaker algorithm [Ross99]. The compression of the *connectivity* of a mesh in Edgebreaker is carried out by traversing the triangles in a defined path, whereby each triangle is used only once. Five different situations are distinguished in traversing, which are marked by one of the symbols {C, L, E, R, S}. The traversal will be explained in detail in the following section. Half of the symbols are C situations. A triangle with a vertex that was not yet viewed in the traversal up to that point is always marked with a C. As the number of triangles T in a mesh is about double the number of vertices V, i.e. T ≈ 2V, 50% of the triangles are of the C tape. A simple coding of the symbols (C=0, L=110, E=111, R=101, S=100) thus guarantees an upper limit of 2 bits per triangle. Using a more efficient code, the upper limit can be reduced to 1.84 bit per triangle [KiRo99]. Contrary to other compression procedures, this upper limit is independent of statistical procedures such as entropy or arithmetic coding, which are usually not very efficient for small meshes with a small number of triangle meshes. The Edgebreaker procedure is therefore very well suited for compressing large data catalogs with many small individual models. The same situation is present in the P-Tree. Here, too, we are dealing with a large collection of depiction geometries which each have a small number of triangles.

Within the scope of this thesis, the procedure for *connectivity* compression was improved to the effect that the five different situations are not rigidly encoded on the basis of the traversal, but are predicted using the geometry of the part of the mesh that was already encoded. If the prediction is correct, only this information is transferred to the client, and the client decodes the triangle using the same prediction. Otherwise, correction data needs to be sent in order to correct the prediction. Experimental results have achieved correct predictions of up to 97% and thus allow a compression of about 0.2 bit per triangle, or 0.4 bit per vertex. In order to explain the Delphi compression procedure in detail, the Edgebreaker traversal and *connectivity* compression will be presented first.

Edgebreaker Compression

Edgebreaker is a finite state machine with which a triangle mesh is traversed. In this traversal, each triangle is marked with one of the symbols CLERS, as depicted in fig. 5-18. In each state, starting from a triangle *F*, an adjacent triangle *T* is encoded and all vertices of *F* and *T* are marked as viewed. Let *left* and *right* be the other two neighboring triangles of *T* and let *v* be the common vertex of *T*, *left* and *right*. If *v* was not yet viewed, then *left* and *right* have not been traversed either. This is situation C. The traversal is continued with the triangle *right*. If *v* was already viewed, the four cases LERS are distinguished. If *right* was traversed but *left* was not, then situation R is the case and encoding is continued with the triangle *left*. Situation L is the analog case where *left* was traversed but *right* was not. If both *left* as well as *right* were traversed, situation E is the case, if neither *left* nor *right* have been traversed, situation S is the case. In the case of an S situation, the traversal is initially continued with the triangle *right*. If an E situation is encountered, the traversal is continued with the triangle *left* of the last S situation. A detailed description of the Edgebreaker Algorithm can be found in [Ross99] and [Ross01].

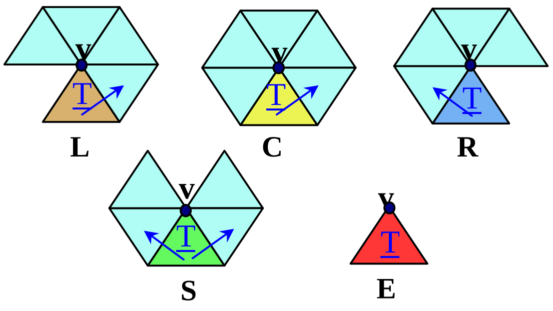


Fig. 5-18: Edgebreaker CLERS symbols

With this algorithm, a corresponding symbol is created for each triangle. The list of symbols created in this way is also called *clers* sequence.

An efficient decompression procedure of linear order, called Wrap&Zip [RoSz99], at first creates a *triangle spanning tree* from the *clers* sequence. Then the free edges are closed to form fitting pairs in a bottom-up approach. An alternative decompression procedure, the so-called Spirale Reversi [IsSo01], interprets the *clers* sequence in the opposite order and builds the *triangle spanning tree* starting from the bottom.

Delphi Compression

The newly developed procedure [CoRo02] in this thesis for *connectivity* compression is based on predicting the situation of the triangle T. Before this procedure can be described in detail, a few conventions must be introduced first. A corner c is the association of a triangle c.t with a vertex c.v of this triangle. A corner belongs to exactly one triangle, each triangle has exactly three corners. The next corner around a triangle in anti-clockwise direction is called c.n, the previous c.p. The corner c.o opposite to c is located in a triangle adjacent to c.t. The position or geometry of a vertex c.v is called c.v.g. Fig. 5-19a clarifies these relations using an example.

In the Delphi compression, as in Edgebreaker, in each situation an adjacent triangle T is encoded from a triangle F. F and T have a common edge E. Let c be the edge of T that is opposite to E and let g(c) be the predicted position of the vertex c.v. In order to make the prediction of the vertex geometry calculable both during compression as well as during decompression, only the position of vertices traversed up to now may be used. In fig. 5-19a, for example, g(c) is predicted from the three corner points of F using the parallelogram rule [ToGo98]:

$$g(c)=c.n.v.g+c.p.v.g-c.o.v.g$$

Let B be the set of all the vertices of the triangles of the mesh encoded so far that are not located in the interior of this submesh. Assuming that the mesh is simply connected, the vertices of B form an oriented border *{c.p.v,* V_r, V_2,..., V_l, *c.n.v}*. Let B'=B\{c.p.v, c.n.v} be the set B without the delimiting vertices of the edge E. B' is also called active border. The vertex X indicates the vertex of B' closest to g(c) and *d:=dist(X, g(c))* the distance between X and g(c). In fig. 5-19b these conventions are depicted in an example.

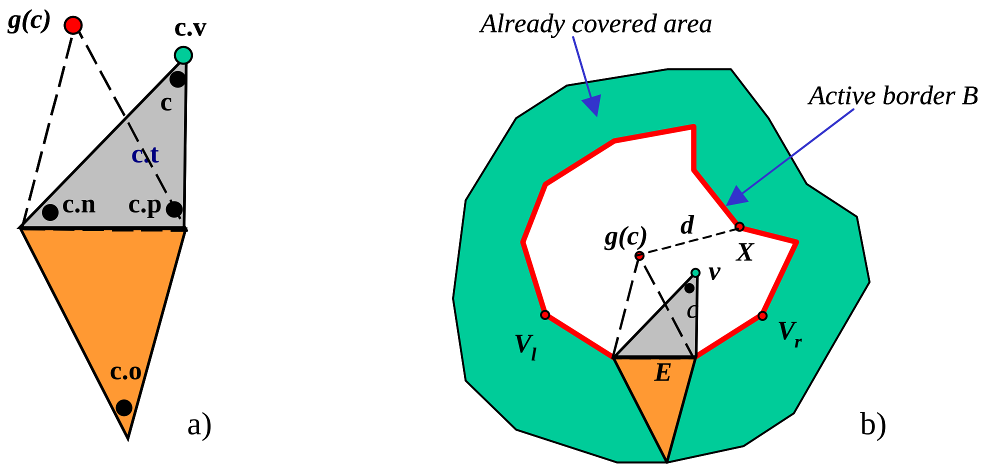


Fig. 5-19: Connectivity predicted using parallelogram rule

The situation of the triangle T can now be predicted using simple decision rules. If the distance between g(c) and B' is too large, a C situation is assumed. Otherwise, X is assumed to be the missing vertex c.v and the situation is predicted as L, E, R or S depending on X, V_l and V_r. Defined more precisely, a triangle T is assumed to be of the C type if $d > \tau\|E\|$ with a constant $\tau < 1$. Otherwise the four following cases are distinguished:

- If X is both V_l as well as V_r, then T is a triangle of type E
- If X is only identical to V_r, then T is of type R
- If X is only identical to V_l, then T is of type L
- Otherwise, situation S is the case

The different cases are displayed in fig. 5-20. One bit in the data flow is sufficient for indicating to the decoder whether the prediction is correct. If the prediction is correct, the decoder not only knows the situation of the current triangle, but also the missing vertex, so that the triangle can be immediately inserted into the mesh. The delayed zipping necessary in Edgebreaker [RoSz99] can thus be avoided without having to code an offset in situation S for identifying the missing vertex, as is the case with other procedures [GuSt98], [ToGo98].

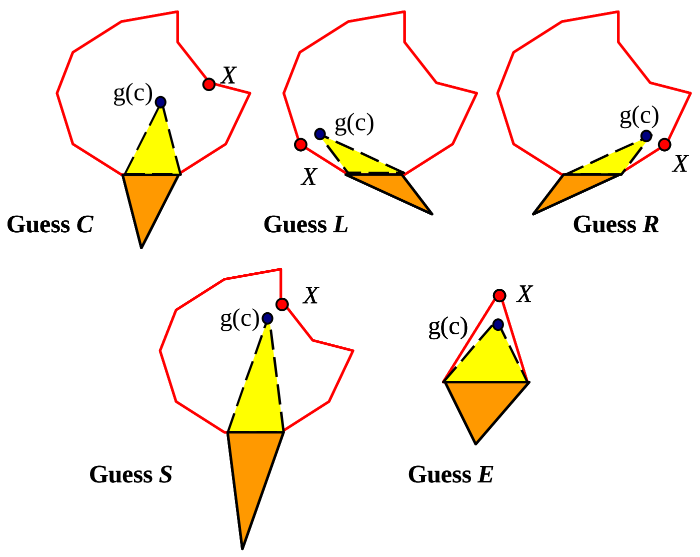


Fig. 5-20: Prediction of the five *clers* situations

Of course the prediction can also be wrong. In this case additional information is required in order to correct the prediction. The necessary correction data depend on the predicted symbol and on the length of the active border. The different cases are discussed in the following.

If a C situation was predicted although T is not a C triangle, the free vertex *c.v* of T must have already been visited, i.e. $c.v \in B'$. Corresponding examples are shown in fig. 5-21b. In order to correct the prediction, the triangle type has to be corrected first. If the active border B' only contains one vertex, i.e. $|B'|=1$, then the triangle has to be of type E. In this case, L, R and S situations are not possible. In this case, no further information is necessary for correcting the prediction. If $|B'|=2$, only situations L and R have to be distinguished for the correction. Otherwise, one of the three symbols {L, R, S} has to be transferred for correction. An E situation can only occur if the length of the active border is 1 and therefore does not have to be considered in the other cases.

In an S situation, the vertex c.v. also has to be encoded, as immediate zipping is necessary during decompression. As $c.v \in B'$ applies, at most $log_2 B'$ bits are required for this.

If T was not prdicted as situation C and thus *c.v* was predicted as X and if this prediction is wrong, this can have two different causes. On one hand, *c.v* can be a new vertex that is close to the active border (situation C). On the other hand, *c.v* may have been predicted as being part of the active border, only the assignment of *c.v* and X is incorrect. Fig. 5-22 shows the corresponding cases. Depending on the prediction, different correction data is required in order to rectify the situation:

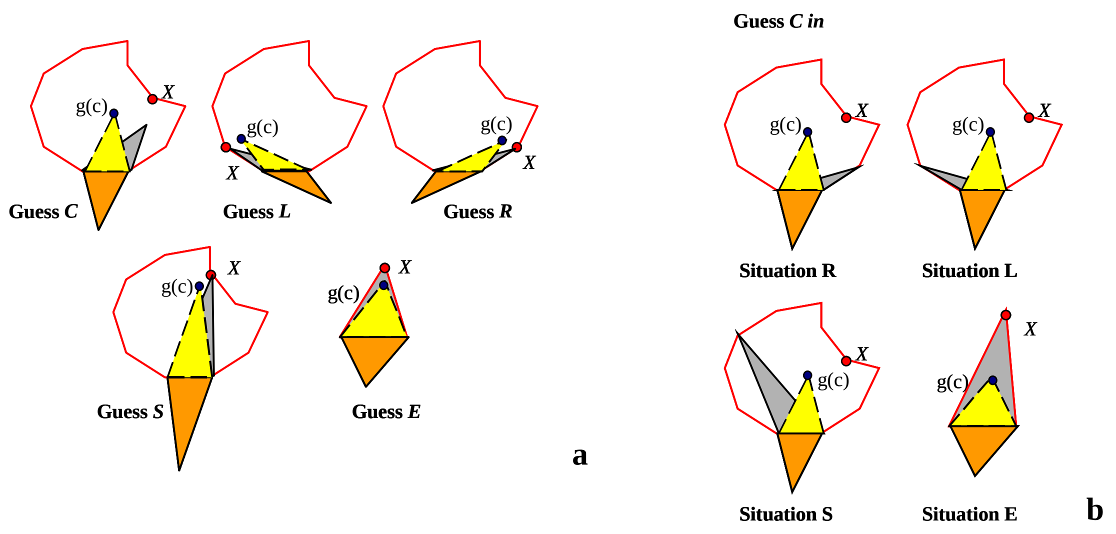


Fig. 5-21: correct prediction and incorrect prediction of a C situation

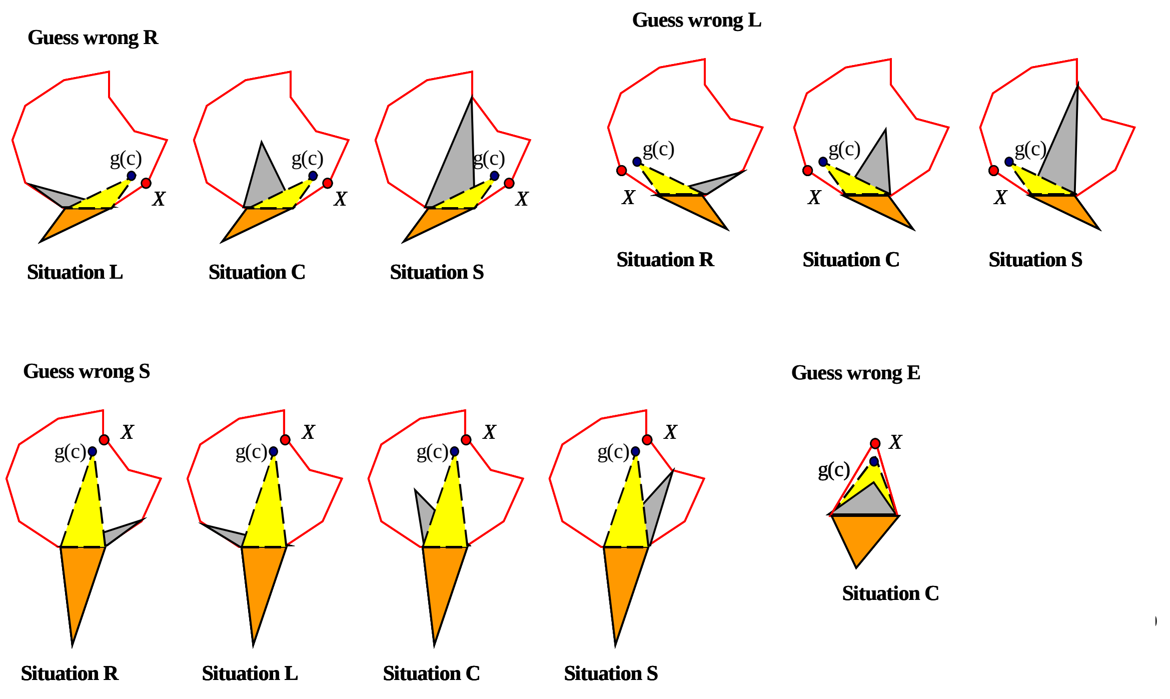


Fig. 5-22: Incorrect predictions of a non-C situation

- If an E situation was predicted, no additional data is required besides the information that the prediction is incorrect. An E situation is only predicted if X is identical with both V_l as well as with V_r. This only occurs when the active border contains only one vertex, i.e. $|B'| = 1$ applies. In the case of an active border of this type, in general only situations C and E are possible. As it is known that situation E is not the case, T has to be of type C.
- In the case of a wrongly predicted R situation, T has to be a triangle of type C, L or S. Type E is not possible, as an E situation can only occur if $|B'| = 1$. In the case of an active border with exactly one vertex, either E or C is predicted, but never situation R. In the case of a corrected S situation, the vertex *c.v* must also be encoded in order to allow immediate zipping.
- Similarly, it applies that for an incorrectly predicted L situation, T has to be of type C, R or S. Only these three cases have to be distinguished.
- If an S situation is predicted incorrectly, an S situation may not be excluded from the consideration. The error cause may also lie in the fact that an S situation is indeed the case, but *c.v* was not assigned to the correct vertex of the active border. In this case, the triangle T can be of type C, L, R or S. In the case of a corrected S situation, as with the incorrect R and L situations, the vertex *c.v* has to be encoded.

If, besides the predicted situation, the number of vertices of the active border |B’| is also taken into account, the necessary correction data can be further restricted. As mentioned above, only situations C and E are possible if the active border consists of only one vertex. In the case of an incorrect prediction when |B’|=1, no additional correction data is therefore necessary.

If the active border consists only of the two vertices V_l and V_r, only the situations C, L and R are possible. Situations E and S cannot occur when |B’|=2. In this case one bit is sufficient for correcting an incorrect prediction, as only the two remaining situations need to be distinguished.

A further restriction can be made for |B’|=3. In this case, for an incorrectly predicted S situation, T is a triangle of type C, L or R. As the active border consists of only three vertices, the type S can be excluded as a possible correction for T.

Using predictions with possible corrections, the *connectivity* of a mesh that is traversed using the Edgebreaker algorithm can be uniquely described by a sequence of 3-tuples A=(G, R, S_O). G is the information on whether the prediction is correct, R is the correction symbol in the case of an incorrect prediction and S_O is the vertex *c.v* for direct *zipping* in the case of the correction symbol S. This sequence of 3-tuples is also called Apollo sequence[1]. This Apollo sequence is equivalent to the *clers* sequence. For statistical

1.The name is inspired by the Greek god Apollo, who is bound to the truth and cannot lie. Just the right one to correct an oracle, if need be.

purposes, the number of vertices in the active border |B'| and the predicted symbol G_S can also be stored in the Apollo sequence. These two values, however, do not have to be transmitted, as they are predicted from the current situation during decoding, similar as in encoding.

In fig. 5-23, the principle of connectivity prediction is shown in the example. The sequence always depicts the same mesh, which is encoded step by step. For simplification, it is assumed that the mesh has already been traversed except for a small region. The red line indicates the remaining active border. Starting from the coordinates of the previous triangle, the position of the unknown vertex c.v is predicted as g(c) using the parallelogram rule. Due to the distance between g(c) and the active border, a C situation is predicted. During compression, this prediction is compared to the actual situation. As the triangle is an R situation, the prediction is marked as wrong and a corresponding correction symbol is stored in the Apollo sequence as a tuple (f, R). As the triangle in the mesh was of type R, the traversal of the mesh is continued towards the left.

The prediction is repeated for all triangles until the entire mesh has been traversed. In the example, the following Apollo sequence is the result: ((t), (f,R), (t), (t), (t), (t), (t), (t), (t), (f,R), (t), (t)). Using a trivial encoding scheme, this sequence can be stored with 16 bits. This is opposed to 32 bits for the corresponding *clers* sequence (*CRSRLECRRRLE*).

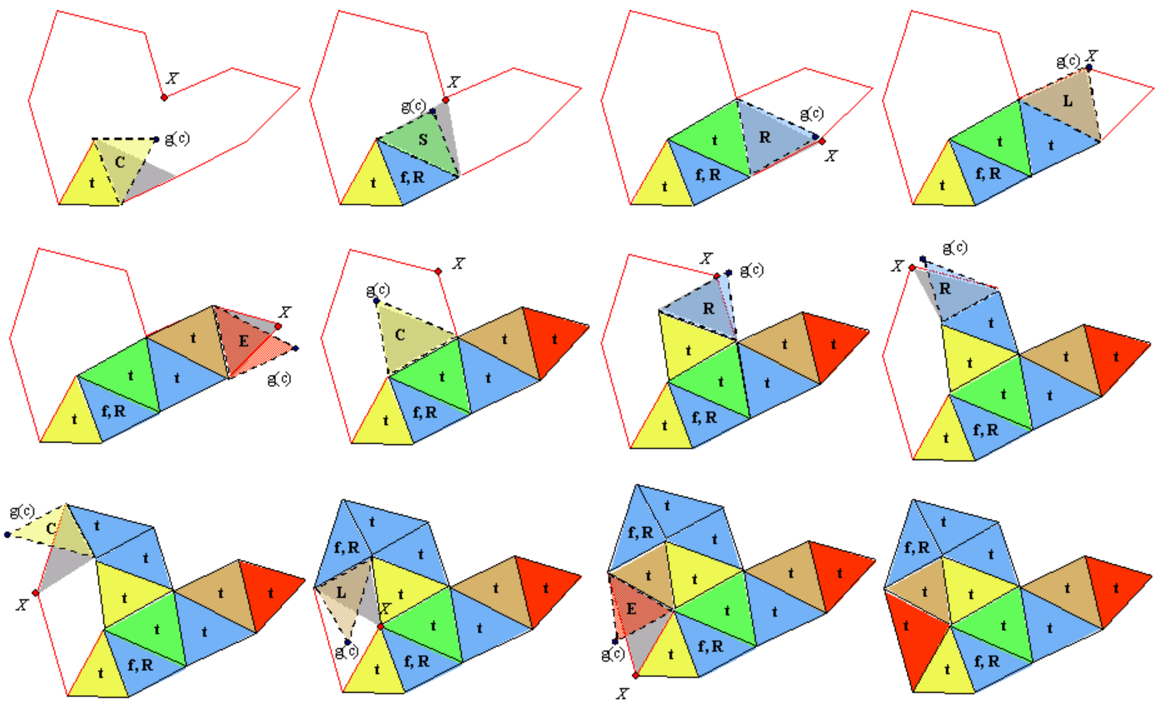


Fig. 5-23: Example Delphi compression

In the section that now follows, a compression scheme for the Apollo sequence is introduced and analyzed. This scheme is to efficiently store and transmit the Apollo sequence.

Compression of the Apollo Sequence

Using the scheme described above, a simply connected mesh with t triangles can be encoded as an Apollo sequence of the length t-1. For transfer, this Apollo sequence should be stored as efficiently as possible, in order to keep the data volume as small as possible. For this, the Apollo sequence is divided into three layers, which can be compressed independently:

- The prediction layer contains the information on whether a prediction is correct
- the correction layers correct the type of a triangle in the case of an incorrect prediction. Here, different cases are distinguished depending on the predicted situation and the length of the active border.
- The offset layer contains the vertex c.v in the case of an S situation that was predicted incorrectly.

The different layers can be compressed, for example using simple Huffman codes or using an adaptive entropy encoder such as the *range encoder* used in [AlDe01]. The achievable compression rates strongly depend on the compression techniques used. In order to ensure that the results are comparable, a method that is independent from the compression technique needs to be found. The entropy is a lower limit for the number of bits necessary for encoding a symbol in a sequence. A good entropy encoding scheme should come close to this lower limit. The entropy of the different layers of the Apollo sequence will thus be determined for measuring the efficiency of the mesh compression.

Entropy

Each layer of the Apollo sequence can be regarded as a homogeneous markov source of order n [TsHa93], as the layers are time-independent. The conditional entropy $H(M|\sigma_k)$ of these markov sources is given by

$$H(M|\sigma_k) = -\Sigma p(m_i|\sigma_k) \log_2(p(m_i|\sigma_k))$$

where $\sigma_k = s_{k1}...s_{kn}$ is the state of the markov source. The transition probabilities $(m_i|\sigma_k)$ indicate the probability of the next output symbol and thus the subsequent state $\underline{\sigma}_k = s_{k2}...s_{kn}m_i$. States that recur in the further course of the process with a probability of 1 are called recurrent. If the return to a state requires on average a finite number of steps, this state is called positively recurrent. States are called periodic if a return is only possible in numbers of steps that are a multiple of a whole number $z > 1$. If every state of a markov source can be reached from every other state in one or more transitions, then the markov source is called irreducible. An irreducible markov source whose states are positively recurrent and aperiodic is called ergodic.

The entropy of an ergodic markov source of order n is given by

$$H(M) = \Sigma p(\sigma_k)\ H(M|\sigma_k).$$

The entropy H(M) of an ergodic markov source is smaller or equal to the entropy of an adjoint memory-less source $H(\underline{M}) = -\Sigma p(m_i)\ \log_2(p(m_i))$.

In the following, prediction and correction layers will be treated as ergodic first order markov sources (n=1).

Compression of the Prediction Layer

The Apollo sequence contains t-1 tuples with one symbol $m_i \in \{t,f\}$ each in the prediction layer M_G. Let p(t|t) be the conditional probability for a correct prediction with the condition that the previous prediction was also correct. The meanings of the other conditional probabilities p(f|t), p(t|f) und p(f|f) are analogous. These probabilities, which can be derived from the Apollo sequence, are used to calculate the conditional entropies $H(M_G|t)$ and $H(M_G|f)$. In order to determine the first order entropy $H(M_G)$, the probabilities p(t) and p(f) have to determined first by solving the following system of equations:

$$p(t)=p(t)p(t|t)+p(f)p(t|f)$$

$$p(f)=p(t)p(f|t)+p(f)p(f|f)$$

$$p(t)+p(f)=1$$

$H(M_G)$ is then given by the sum of the symbol probabilities multiplied by the corresponding conditional entropy:

$$H(M_G)=p(t)H(M_G|t)+p(f)H(M_G|f)$$

A lower limit for the bits necessary for compressing the prediction layer is thus given by

$$b_G=(t-1)\ H(M_G)\ \text{bit}$$

Compression of the Correction Layer

The correction symbols of the Apollo sequence are divided into the four different layers. The assignment of the symbols to these layers is carried out depending on the predicted symbol $G_S \in \{C, L, R, S\}$and the length of the active border B'. In the following, these layers are denoted M_C, M_L, M_R and M_S. A layer for predicted E symbols is not necessary, as the triangle in this case is always of type C and no further information is necessary. Reciprocally, it is not necessary to include the correction symbol E in a layer. An E symbol is only possible for an active border with exactly one vertex. In this situation, however, only either E or C is predicted. If the prediction G_S=C is incorrect, then, accordingly, situation E has to be the case. Additional information is not required. This also means that if the length of the active border |B'|=1, generally no correction symbol is stored.

The number of symbols $m_i \in \{L, R, S\}$ of the layer M_C thus amounts to

$$|M_C|=(t-1)*p(G=f \wedge G_S=C \wedge |B'|>1)$$

A lower limit for the bits necessary for encoding this layer, using the first order entropy, is given by

$$b_C = |M_C|\ H(M_C) \text{ bit.}$$

In analogy, the number of symbols and a corresponding lower limit for the other layers are calculated as follows:

$$b_L = |M_L|\ H(M_L) \text{ bit}$$

$$b_R = |M_R|\ H(M_R) \text{ bit}$$

$$b_S = |M_S|\ H(M_S) \text{ bit}$$

Note that in the layer M_S, all four correction symbols {C, L, R, S} can occur.

A more effective comparison can be achieved if it is taken into account that when the length of the active border |B'|=2, only two symbols have to be distinguished, as no S is possible in this case. A further restriction is that an S cannot occur in the layer M_S if |B'|=3. Experimental results have shown, however, that these special cases occur very rarely and thus hardly improve the compression rate. They are therefore not specially considered here.

Compression of the Offset Layer

If the correction symbol R_S is an S when the length of the active border |B'|>3, then the vertex *c.v* on the active border has to be encoded in order to allow immediate *zipping*. A vertex of the active border is identified by its index s. This index is numbered serially beginning with V_r =1. In any case, a vertex can be encoded using this index with $log_2|B'|$ bits. Experimental results have shown, however, that the desired vertex very often is close to the vertex V_r. Thus, similar to a Huffman code, the offsets are encoded with 0 for s=2, or 10 for s=3, while $2 + log_2|B'|$ bits are used for the remaining offsets. The bits required for compressing the offset layer can be estimated as follows:

$$b_O \le |M_O|*(1+p(s\ge3)+p(s>3)*\log_2(\max|B'|)) \text{ bit,}$$

where $|M_O|$ is the length of the offset layer sequence and max |B'| is the maximum length of the active border in the case of a necessary offset s>3. The encoding given here is very simple and can be improved using entropy-based procedures.

In total, the following costs in bits per vertex (bpv) are estimated for compressing the Apollo sequence:

$$bpv = (b_G + b_C + b_L + b_R + b_S + b_O)/v,$$

where v is the number of vertices in the mesh that is to be compressed.

Decompression

The decompression algorithm of the Apollo sequence is in analogy to the oracle of Delphi. Pythia, the oracle's medium, answers the questions of the seekers in a language that is unintelligible to all but the priests of Apollo. The priests interpret the oracle and give the seekers a truthful answer.

In the decompression of the Apollo sequence, Pythia predicts the *clers* symbol for the new triangle. Here the same algorithm as in compression must be used. This prediction is interpreted using the corresponding Apollo tuple. If the prediction was correct, i.e. G=t, the triangle can be directly inserted into the mesh, as not only the *clers* symbol, but also the third vertex on the active border was predicted correctly. Otherwise, the correction symbol R_S and in the case of R_S=S the offset S_O has to be used to form the mesh. Delayed zipping as with Wrap&Zip decompression of a *clers* sequence must be avoided, as the Pythia prediction is based on the mesh geometry decoded so far.

5.3.3 Compression Results

In this section, the Apollo sequence and in particular the probabilities of the different situations requiring correction are analyzed exemplarily for a model. To conclude, the efficiency of the Delphi compression will be measured for a series of meshes. In order to make this comparable to other techniques, typical models in computer graphics will be used.

Analysis of the Apollo Sequence

In the following, the Apollo sequence will be analyzed using the example of a triangulated model of a horse. The model is a simply connected mesh with 96966 triangles and 48485 vertices. The parallelogram rule is used for predicting the clers symbols, the constant τ for deciding a C situation is defined with $\tau = 0.6$. Fig. 5-25 displays the probabilities of a correct prediction, the length of the active border and the rectification symbol in the case of a wrong prediction. Using the parallelogram rule, about 83% of the triangles are predicted correctly. For the incorrect predictions with |B'|=1, which amount to 1.5%, no further correction data is necessary. The remaining 98.5% of the incorrectly predicted symbols, however, require a correction symbol. As the case |B'|=2 is rare, this situation is not specially distinguished during compression of the correction data.

For the prediction layer, these probabilities and the distribution of the symbols lead to an entropy of $H(M_G) = 0.66$. Thus

$$b_G=(t-1)\,H(M_G) = 96965*0.66 = 63997 \text{ bits}$$

are necessary for the compression of this layer.

The second table displays the probability P_{XY} of an incorrectly predicted symbol G_S=Y and the corresponding correction symbol R_S=X for a length of the active border of $|B'| \geq 3$:

$$P_{XY} = P(R_S=X \mid G=f \wedge G_S=Y \wedge |B'| \geq 3)$$

As can be seen in fig. 5-25, the largest part of the incorrectly predicted C situations (G_S=C) is corrected by an R, i.e. the C layer contains to the most part R symbols. This is utilized in entropy encoding and leads to a compact code. The same applies in the analogous situation (G_S=R and R_S=C) for the R layer.

Thus, the following volume can be expected for the compression of these four layers:

$$b_C = |M_C|\ H(M_C) = 8758*0.23 = 2014 \text{ bit}$$

$$b_L = |M_L|\ H(M_L) = 250*1.14 = 285 \text{ bit}$$

$$b_R = |M_R|\ H(M_R) = 4761*0.35 = 1666 \text{ bit}$$

$$b_S = |M_S|\ H(M_S) = 2498*1.04 = 2598 \text{ bit}$$

For the compression of the offset layer, it is taken into account that s=2 or s=3 applies in 96% of the cases for the index s of the vertex *c.v. c.v* has a higher index in only 4% of the cases. The maximum length of the active border in such a situation amounts to max|B'| = 560. The offset layer is compressed with $b_O \leq 783$ bit.

In total, the following costs arise for the compression of the Apollo sequence as proposed here:

$$bpv = (b_G + b_C + b_L + b_R + b_S + b_O)/v = 1.47 \text{ bits per vertex},$$

for encoding the mesh connectivity. The corresponding *clers* sequence has an entropy of about 1.75 bpv. The Delphi compression thus improves the Edgebreaker compression here by about 15%.

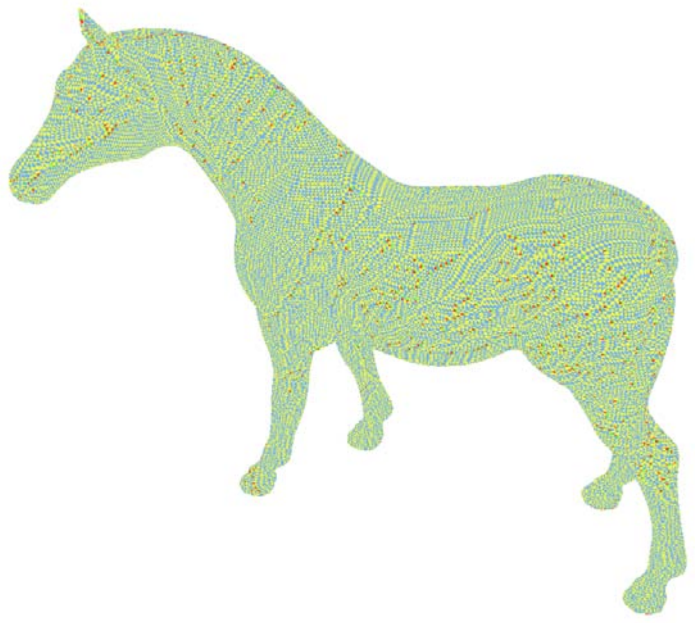


Fig. 5-24: An Apollo sequence for a horse model

P(G)	0.8290
P(¬G)	0.1710
P(|B'|=1|¬G)	0.0162
P(|B'|=2|¬G)	0.0025
P(|B'|>=3|¬G)	0.9813

P_{XY}	***R_S=C***	***R_S=L***	***R_S=R***	***R_S=S***	***Total***
G_S=C	--	*0.0033*	*0.5209*	*0.0143*	*0.5385*
G_S=L	*0.0082*	--	*0.0067*	*0.0004*	*0.0153*
G_S=R	*0.2751*	*0.0013*	--	*0.0163*	*0.2927*
G_S=S	*0.0580*	*0.0003*	*0.0932*	*0.0020*	*0.1535*
Total	*0.3413*	*0.0049*	*0.6208*	*0.0330*	*1.0000*

Fig. 5-25: Statistics of the compression of the horse model

Experimental Results and Comparison

In this section, the results from a series of meshes, which were achieved using the newly developed Delphi compression, are presented. As before, the parallelogram rule and $\tau = 0.6$ was used for prediction. In order to improve the comparability, the compression rate is specified as the first order entropy. The results are compared with the first order entropy of the Edgebreaker *clers* sequence and with the results published in [AlDe01] by Touma and Godsman (TG) [ToGo98], and Alliez and Desbrun [AlDe01]. The technique by Alliez and Desbrun currently creates the most compact lossless compression of mesh *connectivity.*

The results in fig. 5-26 show that with a good prediction of the *clers* symbols, the Delphi compression is clearly more effective than the Edgebreaker compression. Even when only 60% to 65% are predicted correctly, the results are clearly improved. For regular meshes such as the ***Mannequin2*** model, 97% of the predictions are correct, which increases the compression rate compared to Edgebreaker by 300%. For irregular meshes such as the models ***body*** and ***neferiti***, depending on the percentage of correct predictions attained, the Delphi compression is even more efficient than the technique introduced by Alliez and Desbrun, which is considered to be the currently most effective lossless compression technique for mesh connectivity. Especially such irregular models often are the result of mesh simplification and form the initial model of a progressive transfer both for hierarchical level of detail as well as for progressive meshes. A compact representation of this initial mesh ensures a fast first impression for the user.

As the Delphi compression traverses the mesh in the same way that Edgebreaker does, one can discard compression of the Apollo sequence if only about 50% of the predictions are correct. Instead, the mesh is then encoded using the *clers* sequence. Thus the same upper limit is maintained in the Delphi compression as in Edgebreaker and an a priori estimation of the data volume to be transferred remains possible. This a priori estimation is necessary in online compression, as it is intended for the geodata server, in order to be able to decide whether the costs of a corresponding compression are worth it, depending on the available bandwidth.

	#V	#T	Apollo	corret	Edgebreaker	ToGo	AlDe
			bpv	guesses(%)	bpv	bpv	bpv
Mannequin2	11704	23402	0,41	97	1,2	0,46	0,37
horse	48485	96966	1,47	83	1,75		
cow	2904	5804	1,6	83	2,26		
neferiti	299	562	2,17	69	2,58	2,83	2,37
body	711	1396	2,25	65	2,59	2,62	2,35
foot	10016	20028	2,63	60	2,55	2,33	2,2
Venus	8268	16532	2,84	59	2,86	2,82	2,71
david 4	24085	47753	2,9	58	2,79	2,69	2,52
Mannequin1	428	839	2,75	53	2,75	2,97	2,51
david 3	6035	11820	3,1	53	2,9	2,92	2,7
david 1	328	586	3,03	51	2,87	3,58	2,96
david 2	1512	2924	3,15	51	3	3,15	2,88

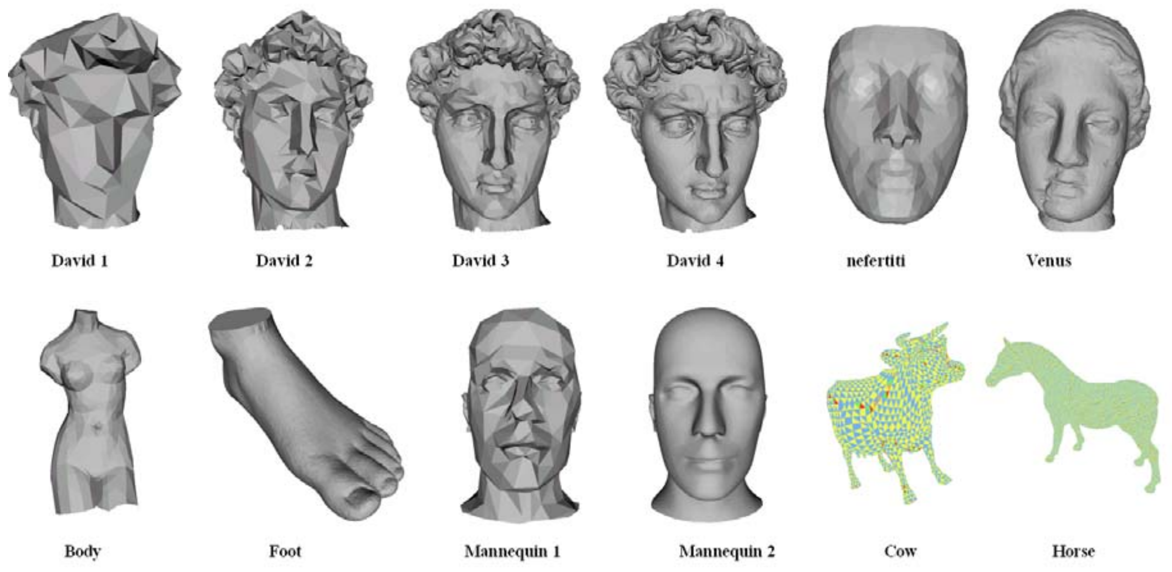


Fig. 5-26: Comparison of compression techniques (table)

Chapter 6

Prototypical Implementation

The following chapter will illustrate several applications using the introduced concept .

6.1 VRML-based 3D GIS interface

6.1.1 Web-based interface

The aim of the web-based 3D GIS interface was a user interface for a 3D GIS for both interactive exploration of the database and thematic queries addressed to the GIS, including result visualization. Based on the Flick 3D GIS web-based interface (see fig. 3.3.2 and/or [Coor97], a SQL-node is developed that allows access to the database from a VRML scene.

Initially, a VRML representation of the database is sent to the web-based user interface, which is displayed in an interactive 3D viewer. For example, Fig. 6-1 shows the multi-story bank buildings in downtown Frankfurt. Additional information and topological queries about the displayed features can be demanded interactively. The primary focus is on two typical GIS request types. The first type of query demands additional thematic information for the feature displayed. The feature is selected interactively and identified internally with an unique name. Generally, the result of this query is textual or multi-media information, which is processed in a table or a normal web page.

The other type of query results goes in the other direction. Here features that meet specific criteria are searched. The result of this query is a list of features that can be identified by their names. Exemplary results of both query types are displayed in fig. 6-2.

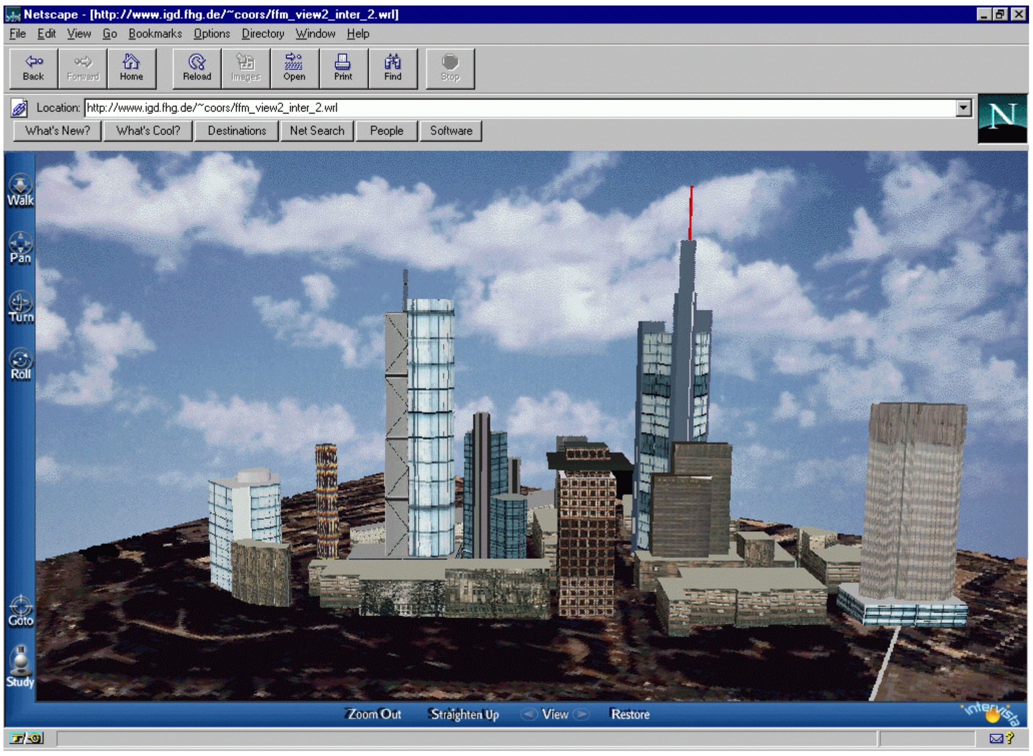
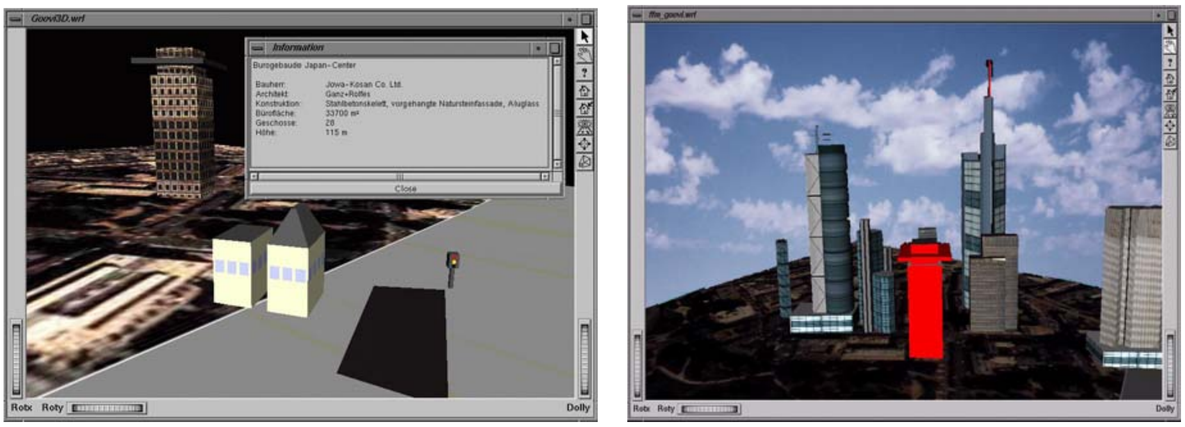

Netscape - [http://www.igd.fhg.de/~coors/ffm_view2_inter_2.wrl]
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location: http://www.igd.fhg.de/~coors/ffm_view2_inter_2.wrl
What's New? What's Cool? Destinations Net Search People Software
Walk Pan Turn Roll Goto Study
Zoom Out Straighten Up View Restore

Fig. 6-1: Web-based user interface of a 3D-GIS

The visualization of the query result requires a mapping of feature identifiers and nodes in the VRML scene representing this feature. For this reason, the name dictionary for the VRML scene was implemented. This can be used to carry out this mapping with an algorithm order *O(m log n)*, where *n* is the number of features represented in the VRML scene and *m* is the number of features in the results. Without a name dictionary, a mapping in VRML scene graphs is only possible with order *O(n log m)*, which is unfavorable for *n>>m*.

For query specification, an embedded SQL node for the VRML scene graph has been developed, with which an arbitrary SQL statement can be specified inside the SQL node. Input parameters and the SQL query result have been made available by using events in the scene graph. The node was realized using Java; database access is provided by JDBC. The access was tested successfully for an Oracle and an Access database. Further explanations and details for accessing data from VRML can be found in [CoJu98]. This concept was picked up in the VRML Database Working Group and adopted as part of the recommended practice for database access for VRML97 Standard [Lipk98].



Burogebaude Japan-Center

Bauherr:	Jowa-Kosan Co. Ltd.
Architekt:	Ganz+Rolfes
Konstruktion:	Stahlbetonskelett, vorgehangte Natursteinfassade, Aluglass
Bürofläche:	33700 m²
Geschosse:	28
Höhe:	115 m

Fig. 6-2: Interactive Queries and Viewing of Results

6.1.2 Cooperative City Planning at the virtual table

The approach of a web-based interface for the 3D geo data server, which was described in the previous section, was extended to support cooperative planning in a virtual environment. Cooperative work by planners, specialists and citizens who are affected can substantially enhance the quality of a planning process. Experimental approaches of citizen involvement in building plans in an urban environment show that decisions are accepted more readily by citizens who are affected when they are involved in the planning process than when the planning approach is centralized, non-transparent and does not involve them [Aria94]. On the other hand, a planning process such as this can be time-consuming, because generally citizen representatives have to be trained in using the planning process first, and all participants have to develop a common understanding of the planning task. Geographic information systems aid decision-making by providing and administrating a multitude of data which are needed for this purpose. Also, data can be analyzed and alternatives concerning different factors can be evaluated. A three-dimensional visualization helps non-experts, especially, to picture the planned project in their minds, particularly when architectural measures are to be taken.

To make these information systems usable for citizens in a planning process, tools which support a distributed working environment, where several persons can plan alternatives together at the same time, are necessary. A horizontal, table-like environment, similar to that ordinarily used for drafts and physical models, is preferable. A useful platform seems to be the so-called Virtual Table, a back-projection technique which allows a stereoscopic output of a three-dimensional virtual model on a projection surface which can be also used horizontally. Back-projection tables like the Barco Baron [Barc98] or the Responsive Workbench [KrFr94] support large-area displays which fulfill the specified requirements.

The objective of this work was the development of a simple interaction possibility at the Virtual Table which can be used intuitively and which gives several participants the opportunity to actively intervene in the planning process. For this purpose, small physical objects are placed on the surface of the table. Contrary to the Ulmer and Ishii approach [UlIs97], these objects have small sources of light on their bottom sides that can be recognized by a video camera. Because of this, these physical objects are called *Active Pieces*. By using different light patterns, different meanings can be assigned to them, e.g. a query activation or a building selection. Of course, besides interactive meanings, different users can be identified. For further information on this interaction technique at the Virtual Table, see [Coor99].

The real interaction and the processing of queries is executed according to the same principle as web-based access on a 3D GIS. The user interface described in the previous section could be used as a frontend on the Virtual Table. The user interface must simply be extended by an interface for the recently developed form of interaction called *Active Pieces*. For this purpose, a VRML node was developed which responds to newly added Active Pieces or to their changing positions. It sends corresponding events which can be further processed in the VRML scene graph. Fig. 6-3 shows the connection of the VRML-based user interface to the 3D GIS. Fig. 6-4 illustrates the mode of operation with active objects on the Virtual Table.

It has been shown that the concept of a 3D geo data server with a loosely connected user interface is very flexible and portable to other applications with low expenses.

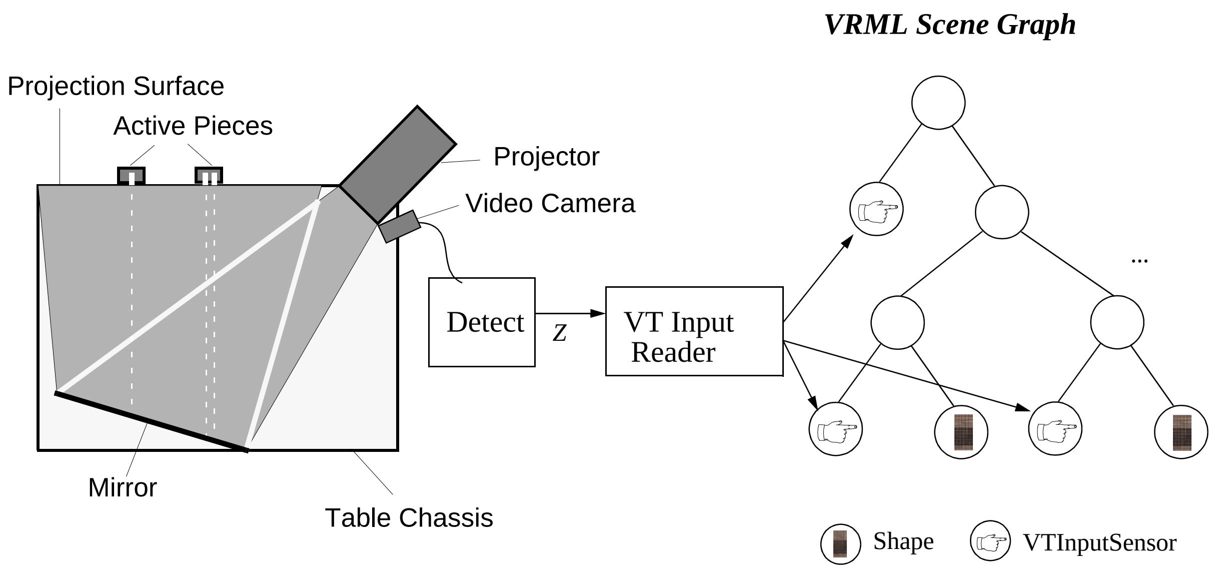


Fig. 6-3: Connection of the *Active Pieces* on the Virtual Table with the VR user interface of the 3D GIS

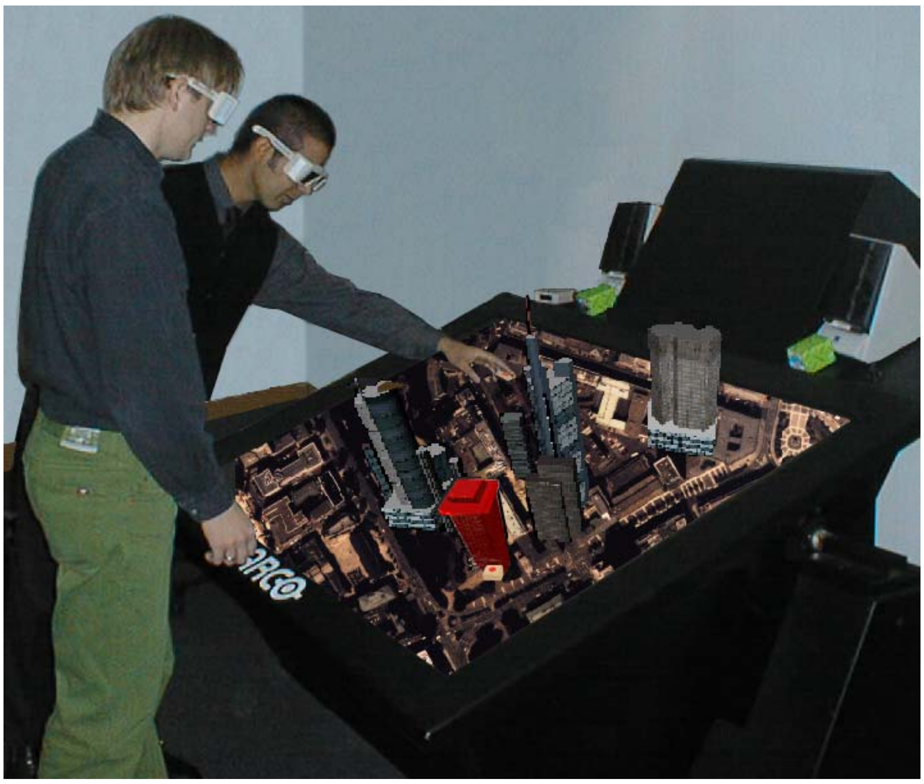


Fig. 6-4: Usage of *Active Pieces* for interaction at the Virtual Table (Montage: the skyscrapers in the images are manipulated to show the stereoscopic impression)

6.2 Integrated Map

The concept of query-specific feature dominance as a basis for graphical abstraction (see section 5.2.2) was implemented for a hotel reservation system as part of the project Integrated Map. The goal of this cooperative project between Fraunhofer Institute for Computer Graphics and the European Media Laboratory was the integration of heterogeneous distributed data sources and external services.

The concept of Integrated Map for the integration of external services was tested using a component for hotel reservation within the intelligent tourist information system Deep-Map [MaZi00]. The hotel-service component integrates several external hotel booking systems and provides standardized access to several hotels on the Internet, without having to formulate search requests explicitly for every single system. Besides the simple search for hotels, the integration of several systems makes a price comparison possible.

In this thesis, only the evaluation and visualization of the query results will be discussed, because the concept of dominance function introduced in section 5.2.2 was used. More information about the Integrated Map project can be found in [Jasn00], [Coor00a] und [Coor01a].

6.2.1 Calculation of the dominance function

After a search request, the evaluation of the result is of basic importance. When searching for a specific attribute such as "search for a single room, with breakfast included, near the station" some hotels meet all requirements, while others only meet some criteria. A rigorous decision function would not add hotels to the result that do not meet at least one attribute. This approach is too restrictive in most cases, because the importance of a specific attribute is not taken into consideration. Because of this, the dominance function suggested in Section 5.2.2 determines the relevance of every single hotel regarding the query for evaluating the single search parameters. For a mere evaluation of the query, only the relevance factor R is important, but for the visualization of the query result, reference objects and landmarks can be taken into account. For calculating the dominance function, the conceptional model in Section 5.2.2 and [CoGö99] should be considered. At this point, only the distance function developed especially for the domain of hotel reservation will be introduced.

Calculation of distance

The calculation of distances between hotel attributes and the concrete query requires the determination of the distance between every single attribute and the corresponding query parameter. The required distance functions depend on the attribute type and the application.

The task of finding suitable distance functions for the individual possible data types can be difficult. For metric data types, the numeric difference may be suitable as distance measure. Non-metric data types such as ordinal (e.g. a hotel’s comfort) and nominal (e.g. kind of breakfast) include no distance which can be interpreted or which is obvious. In this case, application-specific distances dependent on the attribute must be defined. In the case of spatial data, the spatial distance is always given, but is often irrelevant, since it is merely a topological relation between two areas (e.g. inclusion). In the following, the distance functions used in the hotel reservation system are defined.

To set up the distance functions of the query parameters, an order type is assigned to each parameter.

Below, the HotelService distance functions with the order types nominal and boolean are defined. For the quantitative, ordinal and geographic order types, see the distance functions in Section 5.2.2.

parameter	**order type**	**description**
Region	geographic	Spatial reference in Gauss-Krüger coordinates
Pets allowed	boolean	Pets allowed?
Has airport shuttle	boolean	
Smoking	boolean	Smoker / Non-Smoker
Quiet	boolean	Quiet room
Single/Double room	quantitative	Number of desired single/ double rooms
Comfort	ordinal	Hotel’s comfort level (stars)
Price	quantitative	User’s price range
Breakfast	nominal	Kind of breakfast

Fig. 6-5: Order types of request parameters in HotelService

The only request parameter of the *nominal* order type which is included in the distance vector is *breakfast*. The following values are possible for this parameter: UNDEFINED; NONE, BUFFET, AMERICAN_STYLE und CONTINENTAL.

$$dist(p_m, p_q) = \begin{cases} \begin{array}{cccccc} & undef & none & buffet & amer. & conti \\ none & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ buffet & 0.8 & 1.0 & 0.0 & 0.8 & 0.8 \\ amer. & 0.8 & 1.0 & 0.2 & 0.0 & 1.0 \\ conti & 0.8 & 1.0 & 0.1 & 1.0 & 0.0 \end{array} \end{cases}$$

The order type *boolean* is a special case of nominal order type. The distance function between request parameter p_q and corresponding hotel attribute p_m is given by

$$dist(p_m, p_q) = \begin{cases} 0, if\ p_m = p_q \\ 1, else \end{cases}$$

6.2.2 Visualization of Results

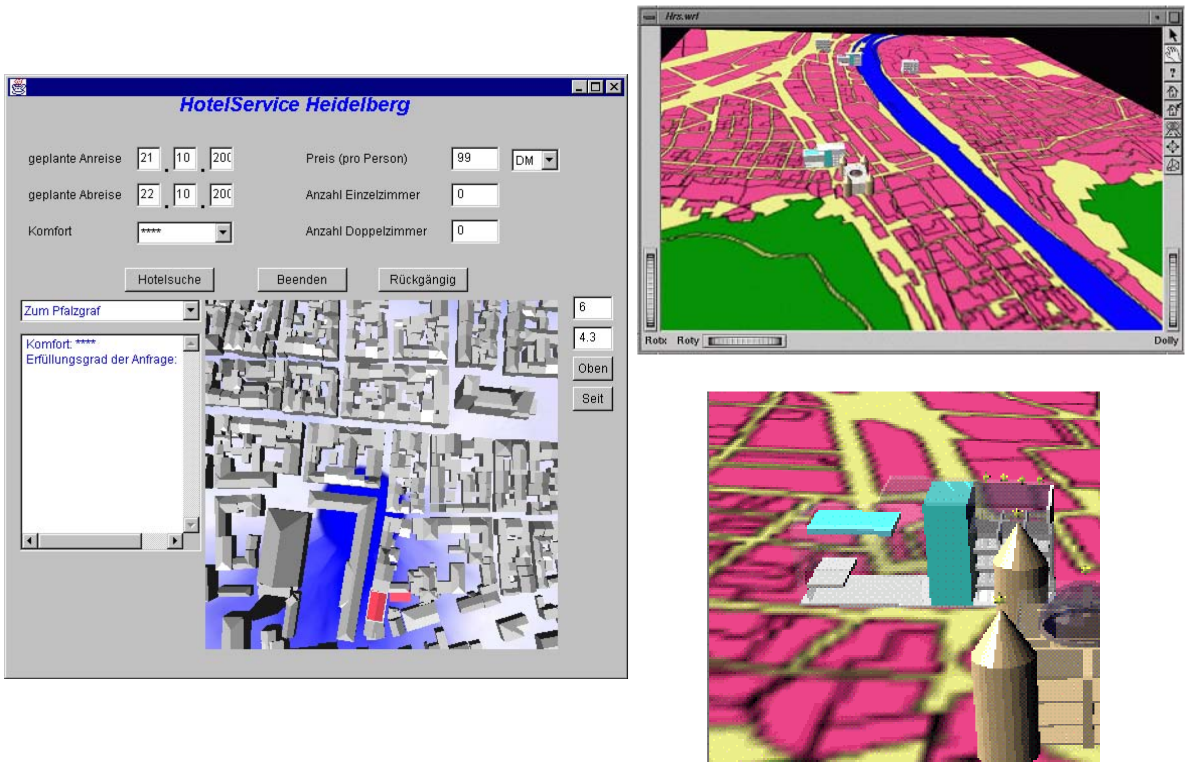
Two alternatives are supported in HotelService for visualizing the query results. First, a three-dimensional city model can be used in which the hotels are color-coded. Such a model is shown in Fig. 6-6 which is embedded in a web-based user interface. A more abstract kind of visualization is shown in the figure on the right. This is a 2.5D landscape model with a cartographic texture, in which the buildings are placed as 3D symbols. Cartographic presentations such as this which include altitude information have moderate data volumes, give an idea of the terrain relief and are able to use common map symbols. In contrast to realistic object presentation, the presentation of individual objects as 3D symbols has the advantage of being able to code additional information. The symbols used in the example are signs that classify the level of hotel comfort.

The relevance factor of every hotel is represented on the transparency of the 3D symbol, so that the user can intuitively recognize to what extent the hotel meets his wishes. The strength of each search parameter can be defined user-specifically. The hotel price can be defined as a guide value which is not strictly evaluated. Hotels are also shown in the query results whose price exceeds this value. During the interactive result visualization, these search parameter weights can be altered. For this reason, an interactive evaluation of the search results is possible, which is very helpful for the user searching for a suitable hotel.

6.3 CityServer3D

The previous examples used the concept of feature geometry and view separation. However, the database was small, so that there was no need to load the data incrementally. In the hotel reservation system, the allocation of feature dominance values depending on a query was shown using examples. The graphical abstraction was restricted to the use of symbols for the hotel comfort classification and the representation of the dominance value on the presentation transparency of an individual feature. Reference objects and background objects were not considered. In the CityServer3D system that was prototypically developed for this thesis, the concepts introduced in this thesis were universally realized.

The core of this system is the *Urban Data Model* (UDM) for the administration of three-dimensional models developed in this thesis. The UDM was implemented as a relational model in Oracle8i and, for mobile applications, in Oracle Light. Database access is provided independent of the underlying database via a Java interface using JDBC. For efficient processing of geometric requests like region queries, an R-tree is used as spatial index. A spatial index of the underlying database can also be used if the query is related to a 2.5-dimensional geometry only, e.g. when selecting regions. Typical three-dimensional geometric queries are visibility analyses. In this case, a spatial index in the database generally is not sufficient, because these refer to two-dimensional objects only. An example is Oracle Spatial, which is an extension of Oracle8i for the management of spatial data. Besides geometric queries, special topological queries are supported which can be de-


HotelService Heidelberg

geplante Anreise	21 . 10 . 200	Preis (pro Person)	99 DM
geplante Abreise	22 . 10 . 200	Anzahl Einzelzimmer	0
Komfort	****	Anzahl Doppelzimmer	0

Zum Pfalzgraf

Komfort: ****
Erfüllungsgrad der Anfrage:

6
4.3

Fig. 6-6: Visualization of a query result (transparency is equivalent to relevance factor)

rived from UDM directly without geometric intersections. In this case, the database functions are also restricted to two-dimensional geometries.

The query result can be processed using the techniques for graphic abstraction introduced in this thesis. As in the previous example (the hotel reservation system), for each feature a dominance value is calculated which results from the user's query and criteria. Starting from these dominance values, the abstraction degree for all features is determined, which is the basis for the aggregation of irrelevant objects. For the actual abstraction, the P-tree developed in this thesis is created. The inner nodes of the P-tree include the graphical presentation of the corresponding subtree. This presentation is generated by aggregation and - if necessary - simplification, depending on the abstraction degree of the subtree features.

The P-tree thus generated is used for progressive transmission of three-dimensional models in a network. Online compression of 3D geometry with small network bandwidth is accomplished using the Delphi compression method introduced in this thesis.

In the following, some aspects of prototypical realization, such as the UDM illustration in a relational model, visibility analyses and the usage of the P-tree are examined in more detail.

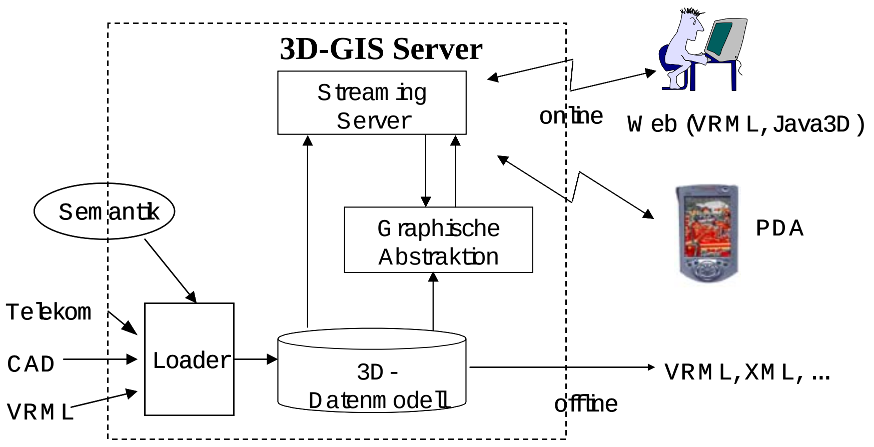


Fig. 6-7: CityServer3D

6.3.1 Logical Database Model

For the storage of 3D geometry data in a database, the data model discussed in Section 4.1 was transferred into a relational database model. For this, an additional restriction was introduced in CityServer3D: a face can only be a planar triangle. This restriction simplifies data storage in a relational data base model [Coor01c]. Because triangles are always convex polygons, all of the conceptional data model's rules are valid without limitations. Figure 6-8 shows the logical model derived from the conceptional query data model for building geometries. Each individual building is realized as a body feature. The ground plan, exterior walls and roof of each building are represented as surface features and can be identified directly. For example, it is possible to extract only the roof landscape in a certain area from the database. Using the ground plans of the buildings without exterior walls or roofs, the same model can be calculated either from a three-dimensional data base or from an intersection of two-dimensional ground plans and a digital terrain model. Taking the relations between buildings and parts of the building, it is possible to aggregate buildings and their parts to higher semantic units. This is necessary when processing large building complexes such as the Darmstadt Castle.

The logical data model was implemented both in Oracle8i and Oracle Lite 3.5 and tested on a 3D city model of Darmstadt. It was a comprehensive model, which covers an area of about 35 km^2 and in which every building is represented by a three-dimensional geometry. When entering building geometries, standard roof shapes such as flat roof, pent roof, span roof, hipped roof and tent roof were used. Other roof shapes such as dome roofs were approximated by the standard shapes. More complex buildings were compiled using different parts of buildings, superstructures and courtyards. In all, the database consists of 20,000 individual buildings. Besides the building geometry, walls, vegetation and a digital terrain model are included in the database. The model was made available by T-Mobile. This database was completed by models of varying textures and levels of detail which were assigned to the corresponding features as views.

To manage this building portfolio in the logical data model presented here, about 20,000 body features, 60,000 surface features, 420,000 faces and 250,000 nodes were created.

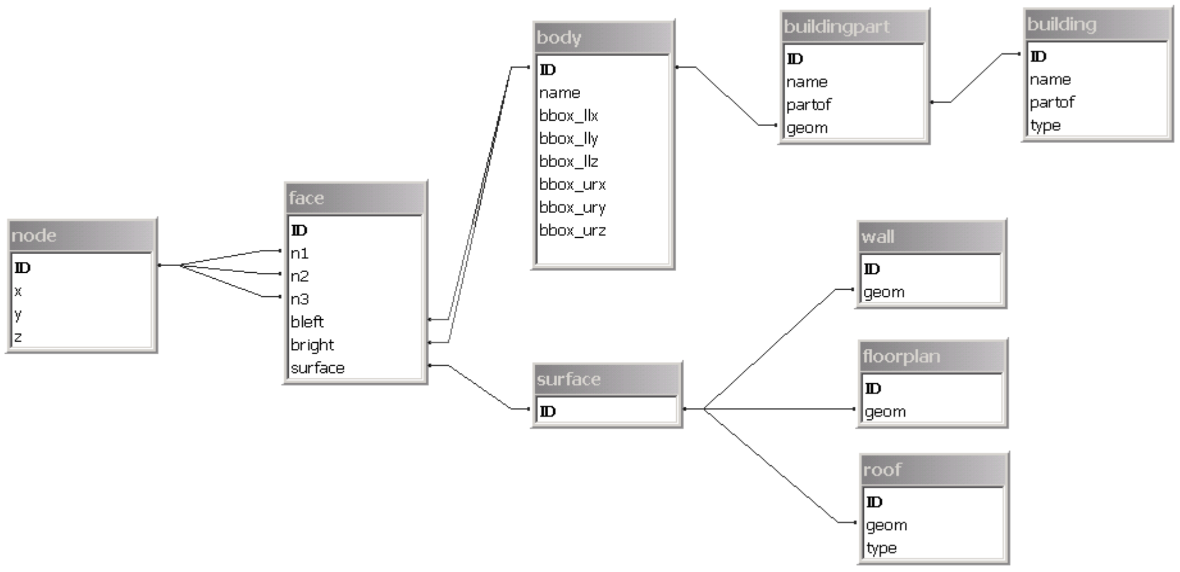


Fig. 6-8: Logical data model

About 83 MB are necessary for the storage of the building geometry alone in the logical query data model. The duration of a region query depends on the size of the area and the number of the buildings included. Experimental queries required an extraction time of 3.5 seconds for 40 and for 130 buildings, and about 15 seconds for 400 buildings.

6.3.2 P-Tree for visualization of a query result

Before the P-tree application for visualization of a region query is described in more detail, the client software presented in Fig. 6-9, which was developed for access to the CityServer3D, should be briefly explained. It is an application in Java/Java3D which can be used both as application and as applet in a web browser. The user selects a region from an aerial view and can specify further query parameters such as the object type. The selection is sent to the CityServer3D as a region query and is processed there. If adequate bandwidth is available, or if a small section of the model has been chosen, the result of the region query is immediately transmitted and visualized. The user can interactively select each individual feature and can use it for further queries, e.g. for obtaining additional information about the object. The model can interactively explored using various navigation metaphors such as terrain-following, orbit-behavior or panorama-sight. To improve orientation, the user's position is represented by a small point in the 2D viewer. For the visualization of a large region, or if only a low bandwidth is available on the Internet, the entire result is not transferred at once; only a highly abstracted model is transferred. This fulfills the user's expectation of quick feedback. For this purpose, the P-tree bounding boxes are used as levels of abstraction. Using this abstraction, the dominance of the features can be considered for better orientation [Coor01b]. If it is only a region query without additional query parameters as in the example shown in Fig. 6-10, only landmarks

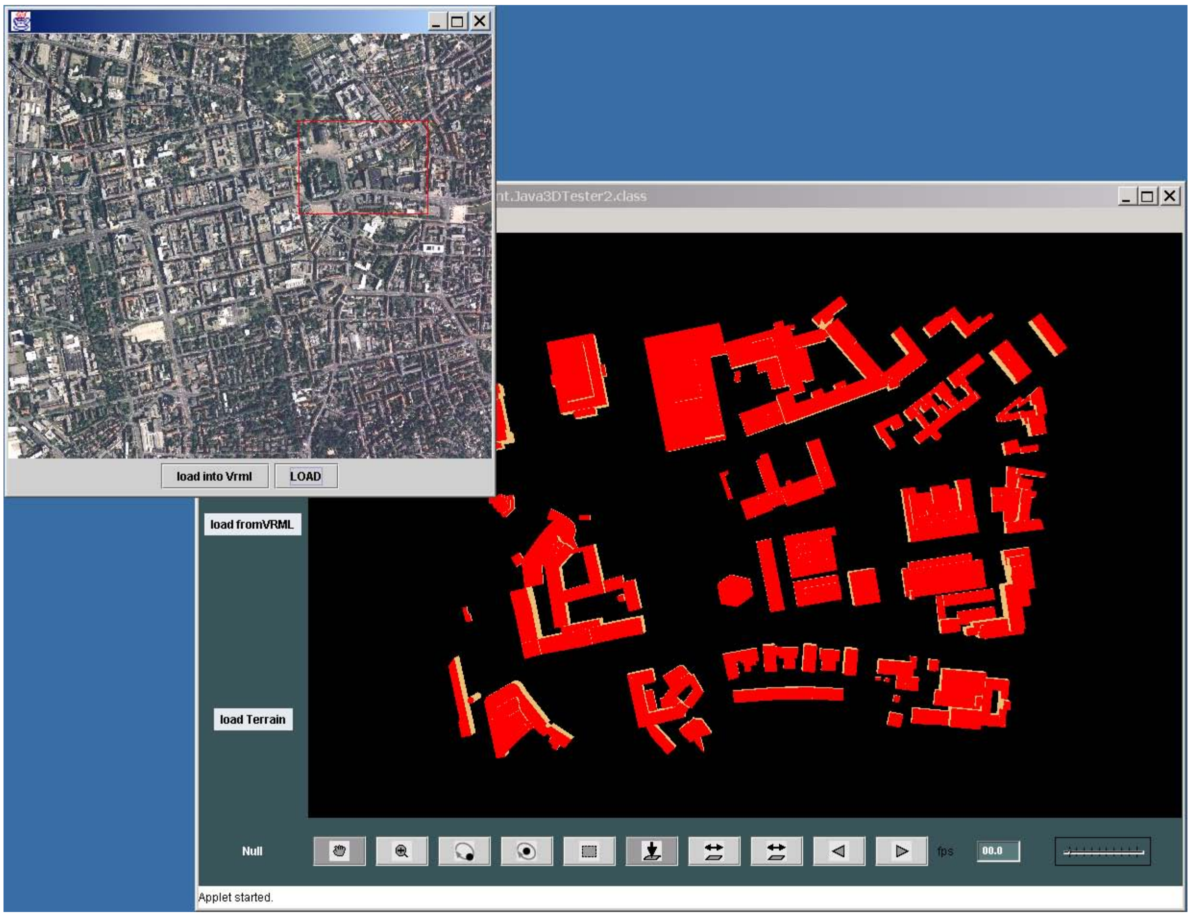


Fig. 6-9: Access software for CityServer3D

have a higher dominance. The illustration on the left shows the P-tree aggregation when dominance values are not taken into account, while the example on the right represents the same scene with the feature dominance taken into account. Landmarks such as the Darmstadt Castle and the City Church remain visible and serve as orientation points in the scene.

6.3.3 Visibility query at the P-Tree

Besides region queries, visibility queries play a crucial role in a three-dimensional model. "Visibility query" means a search for visible geometries when a view pyramid is given. For efficiency reasons, the visibility query extracts all potentially visible buildings. An occlusion of buildings by other geometries (occlusing-culling) is not processed here; this is a conservative visibility query which guarantees that all visible buildings are shown in the result. Nevertheless, it is possible for hidden and thus non-visible buildings to be shown as part of the result. The visibility query is necessary to ensure a fast feedback by a partial P-tree and to download additional geometries in case of focal point rotations, among other things.

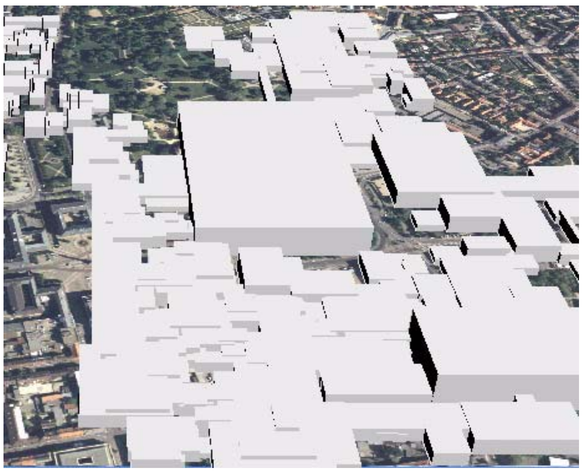
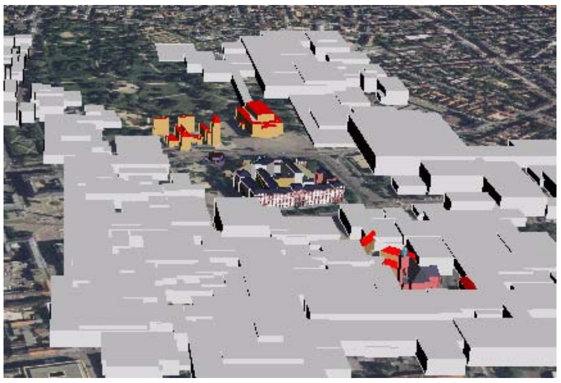



Fig. 6-10: Aggregation with and without focus structure

To process the visibility query efficiently, the view pyramid is represented by an enveloping cone. The test whether an object intersects this cone is not conducted using the object geometry; a bounding sphere, which encloses the geometry, is used instead. This bounding sphere test is a conservative test of linear complexity which can be calculated with low cost [Gärt99]. Because the buildings are relatively small, the difference between the bounding sphere and the object geometry generally can be disregarded.

The view cone K is defined by the eye point P_{eye}, a line of vision v_d and a field of view α. A P-tree interval I is approximated by an optimal bounding sphere S_I. The function *intersect* tests a bounding sphere against a view cone and yields *true* if the intersection of the bounding sphere and *K* is not empty.

To compute *intersect*, the program first tests whether S_I is located entirely in the half space spanned by $-v_d$, that is, behind the eye point:

$$\text{if } (c - p_{eye})v_d < -r \text{ reject}$$

If this is not the case, the program tests whether the projection of $S_I = (C_I, r_I)$ onto an arbitrary projection plane E which is parallel to the plane given by P_{eye} and v_d by the cut of the view cone and this plane. This is the intersection of two circles. Because the projection plane is arbitrary, the plane is chosen that is given by the central point of the envelope sphere and the line of vision.

When $d = (c_I - p_{eye})v_p$ is the distance between the bounding sphere's central point and the plane spanned by the eye point and the line of vision, the intersection of the view cone *K* and *E* as a circle S_K with the central point $C_K = P_{eye} + dv_d$, $(\|v_d\| = 1)$ and the radius $r_K = d\sin\alpha$

$\|C_K - C_I\| \le r_K + r_I$ follows $S_I \cap K \neq \emptyset$

This algorithm has a linear complexity O(n), and the necessary operations for each test are low-cost.

A result of the visibility query is the refinement of a scene with interactive exploration as shown in Fig. 6-11. The aggregated geometry is broken down as soon as the distance to the view point falls below a certain distance, and thus the aggregated geometry has a certain size on the projection plane.

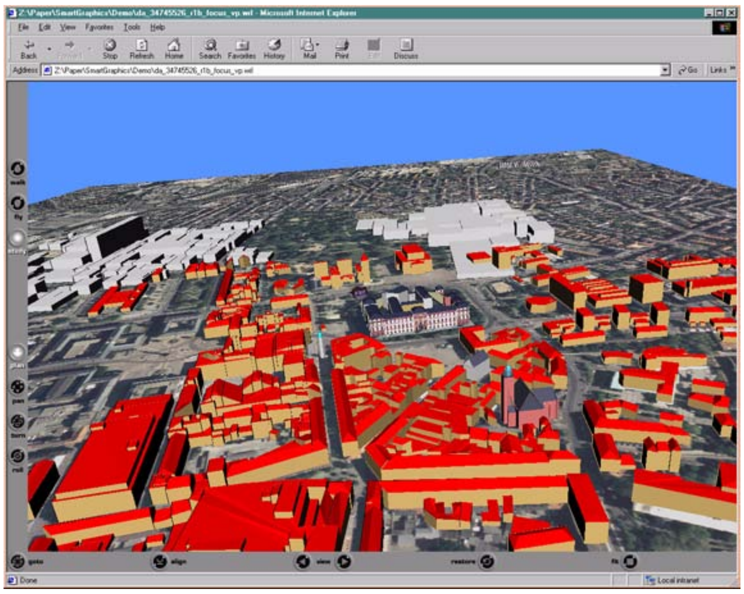


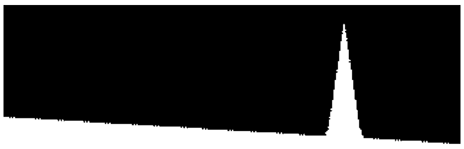
Fig. 6-11: Load-on-demand while exploring interactively

6.3.4 Compression

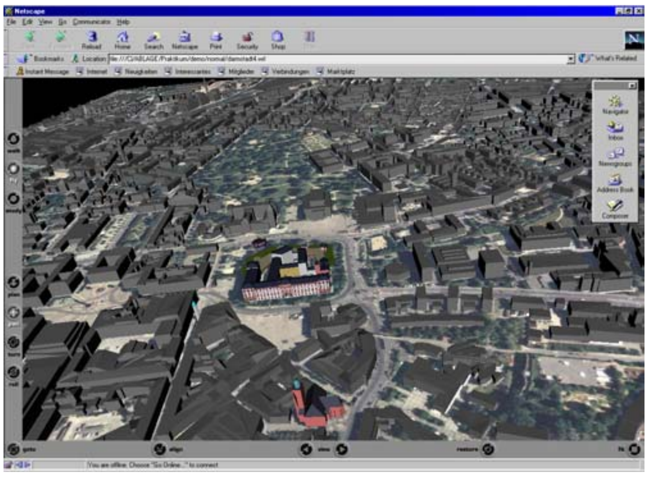
When only a low network bandwidth is available, it is sensible to compress geometry transfers when using the P-tree. The power of the compression method developed in this thesis will be proved using two examples.

The first example is the transfer of the entire city model of Darmstadt. For reasons of simplification, no progressive transfer is to be used. Rather, the entire model is sent to the client without aggregation. This makes sense if the client has enough capacity to render large models.

As a visualization model, the entire Darmstadt model includes about 250,000 vertices and 420,00 triangles. When analyzing the compression rate in this example, only the geometry is considered. Surface attributes such as color and normals are not taken into account in this example. When storing the model in VRML, three *float*-values per vertex and three *integer*-values per triangle are saved. This means 12 bytes per vertex and 12 bytes per triangle. This means a storage requirement of about 8 MB for the entire model. Using a standard compression method like gzip, the data volume can be reduced to 3.12 MB. Using the Delphi compression method for 3D-Meshes developed in this thesis, the Mesh-



Aschaffenburg collegiate



Darmstadt city model

Fig. 6-12: Example data for compression

es connectivity is compressed to a maximum of 2 bits per triangle. The actual compression rates have an average of 1 bit per triangle, and are therefore well below this upper limit. The compression of the vertex coordinates uses the same prediction scheme as for the coding of connectivity. When additional quantization of the coordinates is carried out, there is a storage requirement of 10 bits per vertex. Thus, using the Delphi compression, the complete Darmstadt model has a storage requirement of 105 KB as an upper limit for connectivity and about 312.5 KB for the vertex geometry. The actual compression requires 360 KB which is well under this limit. Thus, a compression rate of 22:1 was achieved and the data volume was reduced by 95.5%. Transfer of the compressed model can be achieved in about 50 seconds on a 56 kbit/s modem connection. But in spite of the high compression rate, without progressive transmission for this model there will be no adequately fast feedback on the Internet. Model compression alone is only sufficient with a bandwidth of 500 KBit/s or more. In contrast, the transmission of the model compressed using gzip would require 8 minutes. The results are compared again in Fig. 6-13.

These results are confirmed by a second example - a model of the Aschaffenburg collegiate church. As part of a cooperation with the Stiftsmuseum Aschaffenburg during a special exhibition in the Roman year 2001, the Aschaffenburg collegiate church was reconstructed as it appeared in different eras [Coor00b]. To make the model available to a multitude of interested people on the web, a compression of the model was required The church model displayed in Fig. 6-12 was created using the modeling tool 3D Studio MAX and exported in VRML format. The size of the VRML model was approx. 7 MB. To process this model for the Internet, the VRML model was optimized by first eliminating standard values and redundant point coordinates. For this, the VRML optimizer Chisel was used. By itself, this optimization of the VRML file reduced the data volume to 3.9 MB. This optimized VRML model was reduced to approx. 1 MB using gzip. The Delphi compression method introduced here reduced the same model to 180 KB - a reduction of more than 95%, as in the previous example.

	Darmstadt city model	**Aschaffenburg collegiate church**
Number of triangles	420,000	155,000
VRML model	8 MB (CityServer3D)	7 MB (3D-Studio Max) 3.9 MB (optimized)
gzip	3.12 MB	0.98 MB
Compression rate	61 %	75 %
Transmission 56 KBit/s	8 min	2 min 20 s
Delphi compression	360 KB	180 KB
Compression rate	95.5%	95.5%
Transmission 56 KBit/s	50 s	25 s

Fig. 6-13: Compression rates for example applications

Chapter 7

Summary and Perspectives

The aim of this thesis was to eliminate essential deficits in the use of previous 3D GIS in an openly distributed GIS infrastructure. The increasing availability of three-dimensional city models poses a special problem, because previous GIS were not able to use such a database sensibly. An additional difficulty occurs in computer graphical representations, where problems exist when generating a meaningful, interactive, three-dimensional presentation of these large models in a heterogeneous network environment. The use of three-dimensional geodata in new applications like Location Based Services is only feasible when these basic technical problems are solved.

This chapter recapitulates the results of the thesis and their significance for GIS research and summarizes the applications. The perspective on future work in this research field is developed subsequently.

7.1 Summary of results

From the scientific view, the most important results of the research thesis are:

- development of a query-oriented data model for efficient analysis of topological relations in three-dimensional, spatial data,
- a method for evaluating the relevance of single feature based on a user specific query and, based on this, the generation of a graphical abstraction of the query result,
- development of a compression method for triangle meshes for progressive transmission of three-dimensional models in a network environment

and the prototypical implementation of these concepts within a 3D geodata server.

Query-oriented data model

The Urban Data Model (UDM) was developed as a query-oriented data model as part of this thesis project (Chapter 5.1) for the administration and analysis of urban data. This data model is used for modeling features from discrete spatial world objects as an extension of OpenGIS data models. The data model is the basis for database-supported administration and analysis of three-dimensional city models. UDM is distinguished by the following four attributes:

- *Feature geometry:* The representation of the spatial dimension of elementary features by exactly one n-dimensional geometric primitive. Here, a point represents a 0-dimensional object, whereas 1-, 2- and 3-dimensional objects are modeled by Line-, Surface- and Body-primitives.

- *Construction elements:* Modeling of primitives by construction elements *Node*, *Arc*, *Face* and *Solid*. By restricting the *face*-geometry to convex polygons, the explicit storage of edges is not necessary. In this way, data volume is approximately halved.

- *3D topology:* Explicit storage of three-dimensional topological relations in the data model and implementation of every topological operation following the Egenhof *9-intersection* model. The topological operators can be created without costly geometric intersections due to the underlying data model.

- *Concept of view:* Separation of object geometry and presentation geometry in connection with a flexible concept for three-dimensional feature visualization.

Graphic Abstraction

A basic problem of many queries within a 3D geodata server is the large data volume of the resulting query result. In this thesis, a concept for graphic abstraction was developed (Chapter 5.2), which is based on a spatial access structure. Within this concept, the feature semantics are taken into consideration in order to reduce the data volume that has to be visualized without losing any essential presentation information. For this purpose, the dominance of a single feature is evaluated related to a concrete user query. This dominance reflects the importance of features to the communication of the query result. When calculating these dominance values, important factors are:

- the user-specific weighted distances of query parameters and the corresponding attributes of the feature,

- the meaning of the feature as reference object in the query and

- the general meaning of the feature as a landmark for different user groups.

The graphic abstraction is carried out using these feature-specific dominance values as a starting point. Depending on the dominance value, each feature is assigned to an abstraction degree. A decision function then selects the proper presentation geometry for every feature; additional technical and cognitive resources such as size and resolution of the display and the focus are considered. For aggregation of background objects, the *P-tree* was developed as part of this thesis. This is an extension of a spatial, tree-like access structure in which additional presentation geometries can be defined in the inner nodes. Each of these presents the entirety of presentation geometries of the whole subtree and complies with a hierarchical level of detail. For example, geometry-simplifying methods from computer graphics can be used for the automated generation of these inner nodes geometry.

Progressive data transfer

The P-Tree generated by graphical abstraction is used for the progressive transfer of three-dimensional models in a heterogeneous network. Using the Delphi compression for triangle meshes developed in this thesis, the data volume of a presentation geometry can be reduced by 95%. The method is based on the Edgebreaker algorithm developed by Rossignac, but uses additional geometric information for compressing the connectivity of a triangle network. In this way, Delphi compression is up to 3 times as compact as when Edgebreaker is used. However, the upper bound is the same as using Edgebreaker compression. This limit is of great importance when estimating the expected data volume and therefore in deciding if the available bandwidth justifies an (online) compression.

CityServer3D

A prototypic concept of a 3D geodata server developed in this thesis was realized using the software system CityServer3D. The system core consists of the query-oriented data model UDM, which was implemented in different data base systems (Oracle8i, Oracle 9i and Cloudscape). The database is accessible via a Java interface. User queries are answered from the database; geometric queries are supported by an R-tree as a spatial index. The query result can be processed with the techniques of graphical abstraction introduced here. For the progressive transfer of three-dimensional models, the P-tree developed in this theses is used. For online compression of 3D geometry using a low network bandwidth, the Delphi method was implemented.

The creation of the CityServer3D proves that a 3D geodata server can be realized with current technology. Using new concepts in graphical abstraction and compression in progressive data transfer, 3D models can also be used performantly in a heterogeneous network environment.

7.2 Perspectives

7.2.1 Future work

The need for further research can be identified for each of the three thesis levels.

The data model for the administration of general three-dimensional geometry models can be extended by using non-polygonal models. In many application fields, besides three-dimensional geometry, the administration of the time dimension is also necessary. The data model introduced here can be used as basis and can be extended to a 4D data model.

The methodology of automatic graphical abstraction can be extended and deepened in several directions. The most important fields of further research might be the integration of methods for cartographic generalization and their extension to three-dimensional geometries and empirical analysis of the impact that graphical abstraction has on the viewer. These research topics may produce valuable findings about the usage of abstract three-dimensional models, especially in map presentation.

Further research areas in the field of progressive data transfer are resource-adaptive methods which focus on the loss of data packets during compression or on changing bandwidths, the goal being to increase the robustness of corresponding solutions with specific protocols. The research project TellMaris briefly described here deals with the resource-adaptive transfer of 3D geometries.

7.2.2 TellMaris

The aim of the research and development project TellMaris is to develop Location Based Services for boat tourists. TellMaris was funded by the European Commission as part of IST-2000 since June 2001 with a duration of 31 months. Besides the Fraunhofer Institute for Computer Graphics, SINTEF (Norway), Nokia (Finland), Helsinki University of Technology (Finland), Pouliadis (Greece), Tellus (Norway) and six local tourism organization are involved in the project.

Within the project framework, the 3D geodata server developed in this thesis is integrated into a geodata infrastructure and used for the generation of interactive three-dimensional city maps. The focal point from the view of research and 3D geodata server development is the use of mobile end devices as visualization platforms. Especially the concepts of graphical abstraction and progressive data transfer are becoming more important in this area; methods for resource-adaptive transfer and rendering must be developed further.

7.2.3 GEIST

GEIST is a conceptual draft which has received funding since March 2001 with a duration of three years. It was developed for an Ideas Competition for Virtual and Augmented Reality, sponsored by the German Federal Ministry of Education and Research. The overall project volume is 5.4 million DM. The Fraunhofer Institut for Computer Graphics, the COmputer Graphics Center and the European Media Laboratory are involved in the project consortium.

The aim of the GEIST project is the research and development of a mobile augmented reality information system for the experience of historic areas and events in an urban environment, using digital storytelling methods. The communication of cultural heritage during a city tour or when visiting historic cultural landmarks thus becomes an unforgettable experience. Further, the attractiveness of this method of communication is expected to enlarge the target group and to excite more interest in history than traditional tours do.

The exact visual superimposition of digital reconstructions on the actual existing buildings is crucial for an on-the-spot experience. The tracking system needed for this is being developed at Fraunhofer IGD. For this, a hybrid approach is used, in which a gyroscope-based tracking is refined by a video-based method. A small video camera will be installed in the display, ensuring that the video picture corresponds to the visitor's view. The video stream is matched using a virtual city model. A 3D geodata server should be used for the administration of this city model. This geodata server can be used for the explicit storage of significant characteristics such as the edges of buildings, as well as the actual building geometry. For the matching, a geometric model in the area surrounding a given point is extrapolated within the 3D geodata server. The significant corners and edges are extrapolated in the video picture and are matched to the corresponding data in the 3D model. Preliminary research in this area was conducted in [Coor00c] and [Coor00d].

Bibliography

[AdV99] Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland (AdV), *AdV-Konzept für die Modellierung der Geoinformationssysteme ALKIS und ATKIS*, 1999.

[AlDe01] Alliez, P. and Desbrun, M., *Valence-Driven Connectivity Encoding for 3D Meshes*. Eurographics 2001 Conference Proceedings, 2001.

[Alle90] Allen, R.B., *User models: theory, method and practice*. Int. Journal of Man-Machine Studies, Vol. 32, 1990, pp 511-543.

[Ande96] Anderson, J., *Kognitive Psychologie*. 2nd edition, Spektrum Akademischer Verlag, 1996.

[Andr95] Andre, E., *Ein planbasierter Ansatz zur Generierung multimedialer Präsentationen*. Dissertationen zur Künstlichen Intelligenz, Bd. 108, infix, 1995.

[Aria94] Arias, E.G., *Decision support in locational analysis for urban growth management: an integrated approach*. Proceedings of GIS 94, Wiesbaden, Germany, 1994.

[Barc98] Barco Systems, *Baron Product Description*. http://www.barco.com/projecti/products/bsp/baron.htm, 1998.

[Beck90] Beckmann, N. Kriegel, H.-P., Schneider, R. and Seeger, B., *The R*-tree: An efficient and robust access method for points and rectangles.* In Proceedings of ACM SIGMOD International Conference on Management of Data, 1990, pp 322-331.

[Bent79] Bentley, J.L., *Multidimensional binary search in database applications*. IEEE Trans. Softw. Eng. 4, 5, 333-340.

[BiFr99] Bill, R, and Fritsch, D., *Grundlagen der Geo-Informationssysteme*. Vol. 1, Wichmann, 4. Auflage, 1999.

[Birc93] Bric, V., *3D Vector data structures and modelling of simple objects in GIS.* MSc Thesis, ITC, The Netherlands, 1993.

[BrHa00] Brenner, C. and Haala, N., *Erfassung von 3D Stadtmodellen*. PFG - Photogrammetrie, Fernerkundung, Geoinformation, Vol. 2/2000, pp.109 - 118, 2000.

[Brin93]	Brinkhoff, T., Kriegel, H., Seeger, B., *Efficient Processing of Spatial Joins Using R-Trees,* Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'93), Washington DC, 1993, pp. 237-246.
[BuKr97]	Butz, A. and Krüger, A., *Zur Auswahl von Abstraktionsgraden*, In: Deussen, O., Lorenz, P. (Ed.) "Simulation und Animation 97, Madgeburg, 1997, ISBN 1-56555-111-7.
[BuMa91]	Buttenfield, B.P. and Mark, D.M., *Expert systems in cartographic design*. In: D.R. Fraser Taylor (Ed.) Geographic Information Systems: The Microcomputer and Modern Cartography. Pergamon Press, Oxford, 1991, pp. 129-150.
[BüMc96]	Bühler, McKee, *The OpenGIS Guide*, OpenGIS Consortium, Inc, 1996.
[Bund95]	Bundy, G., Jones, C. and Furse, *E., Holistic generalization of large-scale cartographic data*. In: Müller, J.C., Weibel, R., Lagrange, J.P. GIS and Generalisation: Methodology and Practice, London Taylor & Francis, 1995, pp 106-119.
[Clem93]	Clementini, E., Di Felice, P., Van Oosterom, P., *A Small Set of Formal Topological Relationships Suitable for End-User Interaction*. In: Third International Symposium on Spatial Data Handling. Proceedings, Singapore, 1993, pp. 277-295.
[CoFl98]	Coors, V. and Flick, S., *Integrating Levels of Detail in a Web-based 3D-GIS*. 6th ACM Symposium on Geographic Information Systems (ACM GIS 98), Washington D.C., USA, 1998.
[Cohe01]	D. Cohen-Or, Y. Noimark, and T. Zvi, *A Server-based Interactive Remote Walkthrough*. 6th Euographics Workshop on Multimedia (EGMM), Manchester, UK, 2001.
[CoGö99]	Coors, V., Göbel, S. and Balfanz, D., *Einsatz von Visualisierungtechniken in Hotelreservierungssystemen,* In: B. Schmidt, C. Uhlenküken (Ed.) „Visualisierung raumbezogener Daten. Method.
[Cohe99]	Cohen-Or, D., Levin, D., and Remez, O., *Progressive Compression of Arbitrary Triangular Meshes*. Visualization 99 Conference Proceedings, pp. 67-72, 1999.
[CoJu98]	Coors, V. and Jung, V., *Using VRML as an Interface to the 3D Data Warehouse*. Symposium on the Virtual Reality Modeling Language (VRML98), Monterey, CA, USA, Feb. 16-19, 1998.
[Coor97]	Coors, V., *Konzeption und Entwicklung einer Client/Server-Architektur für den Zugriff und die Visualisierung von gegraphischen 3D-Informationen via World Wide Web,* Diplomarbeit, Technische Hochschule Darmstadt, FB Informatik, Fachgebiet Graphisch-Interaktive Systeme.
[Coor99]	Coors, V., Jung, V. and Jasnoch U., *Using the Virtual Table as an interaction platform for collaborative urban planning*. Computers & Graphics, Vol. 23 No.4, 1999, Elsevier Sciene Ltd., U.K.

[Coor00a] Coors, V., *Integrated Map Abschlußbericht*. Fraunhofer IGD, Darmstadt, 2000.

[Coor00b] Coors, V, Jasnoch, U. and Joeckle, K., *A Virtual Visit to the Collegiate Church St. Peter and Alexander, Aschaffenburg*. Proceedings of High Performance Graphics Systems and Applications, European Workshop, Bologna, Italy, Oktober 2000.

[Coor00c] Coors, V., Huch, T. and Kretschmer, U., *Videobasierte Blickrichtungsbestimmung in einem urbanen Umfeld*. GeoViSC 2000, Universität Münster, September 2000.

[Coor00d] Coors, V., Huch, T. and Kretschmer, U., *Matching Buildings: Pose Estimation in an Urban Environment*. IEEE International Symposium on Augmented Reality (ISAR), München, Oktober 2000.

[Coor01a] Coors, V., *Integrated Map*. European Media Laboratory Annual Report 2000, Heidleberg, pp 67-69, 2001.

[Coor01b] Coors, V., *Feature-preserving Simplification in Web-based 3D-GIS*. Proceedings of International Symposium on Smart Graphics, New York, USA, ACM Press, March 2001.

[Coor01c] Coors, V., *3D-GIS in Networking Environments*. Proccedings of International Workshop of 3D Cadastre, Delft, November 2001.

[Coor02a] Coors, V., *Resource-adaptive interactive 3D maps.* 2nd International Symposium on Smart Graphics, Hawthorne, New York, USA, ACM Press, 2002.

[Coor02b] Coors, V., *Resource-adaptive 3D-Maps for Location Based Services*. UDMS 2002, Prag, Oktober 2002.

[Coor02c] Coors, V., *Dreidimensionale Karten für Location Based Services*. in: Zipf/ Strobl (Eds.) Geoinformation mobil, Wichmann, pp. 14-25, 2002.

[Coor02d] Coors, V., *3D-GIS in Networking Environments*. accepted for pulication in CEUS Journal for Computer, Environments and Urban Systems, Special Issue on 3D Cadastre, 2002.

[CoRo02] Coors, V, and Rossignac, J., *Guess Connectivity: Delphi Encoding in Edgebreaker*. GaTech Technical Report, 2002.

[CoSc99] Coors, V., und Schulz, T., *Progressive Datenübertragung in einem WWW-basierten 3D-Geoinformationssystem*. Proceedings GeoViSC 99, IfGI prints Nr. 6, Universität Münster, 1999

[CrRo99] A. Crosnier, and J. Rossignac, *Tribox bounds for three-dimensional objects*. Computers & Graphics 23, Elsevier Science, pp. 429-437, 1999

[Davi91] David, B., *Modélisation, représentation et gestion d'informatique géographique. Une approche en relationnel étendu*. Thèse de doctorat, Université de Paris 6, 1991.

[Deer95]	Deering, M., *Geometry Compression*. SIGGraph Conference Proceedings 1995, pp. 13-20, 1995.
[EgHe90]	Egenhofer, M.J. and Herring, J.R., *A mathematical framework for the definition of topological relationships*, Proceedings of Fourth International Symposium on SDH, Zurich, Switzerland, pp. 803-813, 1990.
[Enca97]	Encarnacao, J., Straßer, W., Klein, R., *Graphische Datenverarbeitung, Bd. 2 Modellierung komplexer Objekte und photorealistische Bilderzeugung*. 4th edition, 1997.
[ETOPO5]	*5 Minute Gridded Earth Topography Data*. http://edcsgs9.cr.usgs.gov/glis/hyper/guide/etopo5, 2003
[Evan96]	Evans, F., Skiena, S., and Varshney, A., *Optimizing Triangle Strips for Fast Rendering*. Proceedings of IEEE Vizualization'96, pp. 319-326, 1996.
[Fein85]	Feiner, S., *Apex: An experiment in the automated creation of pictorial explanations*. IEEE Computer Graphics & Applications, 5(11), pp. 117-123, 1985.
[Flic96]	Flick, S., *An object-oriented framework for the realisation of 3D Geographic Information Systems*, Proceedings of 2th joint European conference and exhibition on Geographical Information, Barcelona, Spain, pp. 187-196., 1996
[Flic98]	Flick, S., *Konzeption eines adaptiven Frameworks für 3D-Geo-Informationssysteme*. Dissertation TU Darmstadt, FB Informatik, Fraunhofer IRB Verlag, 1998.
[Frey97]	Freykamp, F. *Erweiterung eines Geo-Informationssystems um 3D-Darstellung*. Diplomarbeit, Universität Paderborn, FB Mathematik-Informatik, 1997.
[Fuch98]	Fuchs, C., Gülch, E. and Förstner, W., *OEEPE Survey on 3D-City Models*. OEEPE Publication, No 35, Bundesamt für Kartographie und Geodäsie, Frankfurt, 1998.
[Furn86]	Furnas, G.W., *Generalized fisheye views*. Proceedings of CHI 86, Human Factors in Computing Systems, ACM SIGCHI 86, pp 16-23, 1986.
[Förs99]	Förstner, W., *3D-City Models: Automatic and semiautomatic Acquisition Methods*. Fritsch, D. and Spiller, R. (Ed.): Photogrammetrische Woche 1999, Wichmann, Heidelberg, pp. 291-304, 1999.
[GaGü98]	Gaede, V. and Günther, O., *Multidimensional Access Methods*, ACM Computing Surveys, Vol. 30, 2, pp. 170-231, 1998.
[GaHe98]	Garland, M., and Heckbert, P., *Simplifying Surfaces with Color and Texture using Quadric Error Metrics*. Proceedings of IEEE Visualization 98, 1998.
[GaHe97]	Garland, M., and Heckbert, P., *Surface Simplification using quadric error metrics*. Proceedings of ACM SIGGRAPH 97, 1997.

[Gärt99] Gärtner, B., *Fast and robust smallest enclosing balls*, Proc. 7th Annual European Symposium on Algorithms (ESA), Lecture Notes in Computer Science 1643, Springer-Verlag, Berlin, pp. 325-338, 1999.

[Gree89] Greene, D., *An Implementation and performance analysis of spatial data access methods*. Proceedings of the fifth IEEE International Conference on Data Enginering, pp. 606-615, 1989.

[Grün93] Grünreich, D., *Generalization in GIS environment*. Proceedings of the 16th International Cartographic Conference, Cologne, pp. 203-210, 1993.

[Gumh00] Gumhold, S., *Towards optimal coding and ongoing research*. 3D Geometry Compression Course Notes, Siggraph, 2000.

[GuSt98] Gumhold, S., and Strasser, W., *Real Time Compression of Triangle Mesh Connectivity*. Proceedings of ACM Siggraph, pp. 133-140, 1998.

[Gutt84] Guttmann, A., *R-Trees: A dynamic index structure for spatial searching.* Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 47-54, 1984.

[Hake02] Hake, G., Grünreich, D., Meng, L., *Kartographie*, 8. Auflage, de Gruyter, Berlin, 2002.

[Hart99] Hartmann, K. et al., *Interaction and Focus: Towards a Coherent Degree of Details in Graphics, Captions and Text,* In: Lorenz, P. and Deussen, O. (Eds.): Simulation and Visualization 99, Society for Computer Simulation, Erlangen, 1999.

[HeGa97] Heckbert, P., Garland, M., *Survey of Polygonal Surface Simplification Algorithmn,* School of Computer Science, Carnegie Mellon University, Pittsburgh, 1997.

[Hen79] Henle, M., *A Combinatorial Indroduction to Topology*, W.H. Freeman and Company San Francisco, 1979.

[Hopp96] Hoppe, H., *Progressive Meshes*, Proceedings of ACM SIGGRAPH 96, pp. 99-108, 1996

[Hopp97] Hoppe, H., *View-Dependent Refinement of Progressive Meshes*. Proceedings of ACM SIGGRAPH 97, 1997.

[HuGr00] A. Hubeli, M. Gross, *Fairing of Non-Manifolds for Visualization*. Proceedings of IEEE Visualization 2000, IEEE Computer Society Press, Visualization 2000, pp. 407-414, 2000.

[IsSo01] Isenburg, M. and J. Snoeyink, J., *Spirale Reversi: Reverse decoding of the Edgebreaker encoding*, Journal of Computational Geometry: Theory and Applications, 20 (1-2), pp. 39-52, 2001.

[Jasn00] Jasnoch, U., Balfanz, D. and Coors, V., *Managing Distributed Heterogeneous Information Spaces*. IEEE FAIM conference 2000, Washington D.C., USA, pp 431-440, 2000.

[Jung98] Jung, V., *Integrierte Benutzerunterstützung für die Visualisierung in Geo-Informationssystemen*. Dissertation TU Darmstadt, FB Informatik, Fraunhofer IRB Verlag, 1998.

[KeKr94] Keim, D. and Kriegel, H.-P., *VisDB: Database Exploration using Multidimensional Visualization*. IEEE Computer Graphics and Applications, 14(5), pp. 40-49, 1994.

[KiRo99] King, D. and Rossignac, J., *Guaranteed 3.67V bit encoding of planar triangle graphs*. 11th Canadian Conference on Computational Geometry (CCCG''99), pp. 146-149, Vancouver, CA, August 15-18, 1999.

[Knol98] Knoll, A., *Integration eines Umweltinformationssystems in ein 3D-GIS zur Entscheidungsunterstützung in der Stadtplanung*. Diploma thesis, TU-Darmstadt, supervisior Prof. Dr. Encarnação, 1998.

[Kofl98] Kofler, M., *R-Trees for Visualizing and Organizing Large 3D GIS Databases*. Dissertation, TU Graz, 1998.

[KoGr98] Kofler, M. and Gruber, M., *Verwaltung und Visualisierung dreidimensionaler Stadtmodelle*, ZPF Zeitschrift für Photogrammetrie und Fernerkundung, 2/98, Wichmann, pp. 44-57, 1998.

[Köse97] Köser, D., *Untersuchung zur Datenbankanbindung für ein 3D Geo-Informationssystem im Städtebau*, Diplomarbeit Universität Rostock, Fachbereich Informatik, Prof. H. Schumann, 1997.

[Kres94] Kresse, W., *Platzierung von Schrift in Karten,* Dissertation, Landwirtschaftliche Fakultät, Universität Bonn, 1994.

[KrFr94] Krüger, W., and Fröhlich, B., *The responsive workbench*. *I*EEE Computer Graphics and Applications, 14 (3), pp. 12–15, 1994.

[Krüg98] Krueger, A., *Graphical Abstraction - How to adapt the detail of graphics to limited resources,* Proceedings of ECAI 98, Brighton, UK, 1998.

[Krüg00] Krüger, A., *Automatische Abstraktion in 3D-Graphiken,* Dissertationen zur Künstlichen Intelligenz, Bd. 232, infix, 2000.

[Lang99] Lange, E., *Realität und computergestützte visuelle Simulation*. Berichte zur Orts-, Regional- und Landesplanung, Bd. 106, vdf Hochschulverlag AG an der ETH Zürich, 1999.

[Laur93] Laurini, Robert, and Thompson, Derek, *Fundamentals Of Spatial Information Systems.* Apic Series, Harcourt Publishers Ltd, 1993.

[Lee02] Lee, H., Alliez, P., and Desbrun, M., *Angle-Analyzer: A Triangle-Quad Mesh Codec*, Proceedings of Eurographics 2002.

[Lipk98] Lipkin, D., *Recommended Practices for SQL Database Access in VRML,* http://www.web3d.org/Recommended/vrml-sql/vrml-sql-ex.html, 1998.

[Losa98] de la Losa, A., *Towards a 3D GIS. From 2D GIS and 3D CAD, Conception of a 3D GIS with a complete Topological Management*. Proceedings of GISPlanet 98, Lisbon, Portugal, 1998.

[LoCe99] de la Losa, A. and Cervelle, B., *3D Topological modeling and visualization for 3D GIS*. Computers & Graphics 23/4, Pergamon, pp. 269-478, 1999.

[Mänt88] Mäntylä, M., *An Introduction to Solid Modeling*. Computer Science Press, Inc. 1988.

[Mart00] Martin, I. M., *ARTE - An Adaptive Rendering and Transmission Environment for 3D Graphics*. Proceedings of ACM Multimedia Conference, Los Angeles, CA, 2000.

[Maye99] Mayer, H., *Automatic Object Extraction from Aerial Imagery - A Survey Focusing on Buildings*. Computer Vision and Image Understanding 74(2), pp.138-149, 1999.

[MaZi00] Malaka, R. and Zipf, A., *DEEP MAP - Challenging IT research in the framework of a tourist information system*. Fesenmaier, D. Klein, S. and Buhalis, D. (Eds.): Information and Communication Technologies in Tourism 2000. Proceedings of ENTER 2000, 7th. International Congress on Tourism and Communications Technologies in Tourism. Barcelona. Spain. Springer Computer Science, Wien, New York. pp. 15-27, 2000.

[Mole92] Molenaar, M., *A topology for 3D vector maps*, ITC Journal 1992-1, pp. 25-33, 1992.

[Mole98] Molenaar, M., *An Introduction to the Theory of Spatial Object Modelling for GIS,* Taylor & Francis, 1998.

[Müll91] Müller, J.C., *Generalization of spatial databases*. Maguire, D, Goodchild, M. and Rhind, D. (Eds.) Geographical Information Systems: Principles and Applications, Vol. 1, Harlow: Longman, 1991, pp 457-75.

[Müll95] Müller, J.C., Weibel, R., Lagrange, J.P., *GIS and Generalisation: Methodology and Practice*, London Taylor & Francis, pp. 3-17, 1995.

[Noik94] Noik, E.G., *A Space of Presentation Emphasis Techniques for Visualizing Graphs*. Proceedings of Graphics Interface '94, 1994.

[Pape96] Pape, D., *VRML Topographic Map Generator*, http://evlweb.eecs.uic.edu/pape/vrml/etopo, Electronic Visualization Laboratory, University of Illinois, Chicago, USA, 1996

[PaRo00] Pajarola, R. and Rossignac, J., *Compressed Progressive Meshes*. IEEE Transactions on Visualization and Computer Graphics, 6(1), pp. 79-93, 2000.

[Pfun02] Pfund, M., *3D GIS Architecture - A Topological Data Structure*, GIM International, 16 (2), pp. 35-37, 2002.

[Pig91] Pigot, S., Topological Models for 3D Spatial Information Systems. *Proc. 10th International Conference on Computer Assisted Cartography, Auto-Carto*, Vol.10, 368-392, 1991.

[Pilo96] Pilouk, M., *Integrated Modeling for 3D GIS*, PhD thesis, ITC, 1996.

[Plia01] PliableGIS, Extension for ArcView 8.1, http://www.idelix.com, 2001

[Powi93] Powitz, B., *Zur Automatisierung der kartograpischen Generalisierung topographischer Daten in GIS*. Doktorarbeit, Universität Hannover, 1993.

[RaLa00] Ragia, L. and Laing, R., *Qualitätsprüfung von Gebäudedaten aus photogrammetrischen Auswertungen*. PFG - Photogrammetrie Fernerkundung Geoinformation 3/2000, Schweizerbart'sche Verlagsbuchhandlung, Stuttgart, pp . 212-220, 2000.

[RaSc98] Rauschenbach, U. and Schumann, H., *Flexible Embedded Image Communication using Levels of Detail and Regions of Interest,* Proceedings of IMC '98 - Interactive Applications of Mobile Computing, Rostock, Germany, 1998.

[Raus00] Rauschenbach, U., *Bedarfsgesteuerte Bildübertragung mit Regions of Interest und Levels of Detail für mobile Umgebungen*, Dissertation, Universität Rostock, Ingenieurwissenschaftliche Fakultät, 2000.

[Raus01] Rauschenbach, U., Jeschke, S. and Schumann, H.: *General Rectangular FishEye Views for 2D Graphics*, in: Computers and Graphics, 25(4), Elsevier Science, 2001.

[Redd99] Reddy, M., Leclerc, Y.G., Iverson, L., and Bletter, N., *TerraVision II: Visualizing Massive Terrain Databases in VRML*. IEEE Computer Graphics and Applications (Special Issue on VRML), 19(2), pp. 30-38.

[Redd00] Reddy, M., Leclerc, Y.G., and Iverson, L., *Under the Hood of GeoVRML 1.0*", Proceedings of The Fifth Web3D/VRML Symposium. Monterey, California, USA, 2000.

[Rikk94] Rikkers, R., Molenaar, M., and Stuiver, J., *A Query Oriented Implementation of a topologic data structure for 3-dimensional vector maps,* International Journal of Geographic Information Systems, 8, pp. 243-260, 1994.

[RoCo89] Rossignac, J. and O'Connor, M., *SGC: A Dimension-independent Model for Pointsets with Internal Structures and Incomplete Boundaries.* Geometric Modelling for Product Engineering, North Holland, pp.145 - 180, 1989.

[RoRe91] Rossignac, J. and Requicha, A., *Constructive Non-Regularized Geometry*. Computer Aided Design 23(1), pp. 21-32. 1991.

[RoRo96] Ronfard, R., and Rossignac, J., *Full-range approximation of triangulated polyhedra*. Computer Graphics Forum, Proceedings of Eurographics '96, 1996.

[RoBo93] Rossignac, J., and Borell, P., *Multi-resolution 3D approximations for rendering complex scenes,* Modeling in Computer Graphics, B. Falicieno & T.L. Kunni (eds.), Springer-Verlag, pp. 455-465, 1993.

[Ross98] Rossignac, J., *Interactive Exploration of Distributed 3D Databases over the Internet.* GVU Center and Collage of Computing, Georgia Institute of Technology, 1998.

[Ross99] Rossignac, J., *Edgebreaker: Connectivity compression for triangle meshes*. IEEE Transactions on Visualization and Computer Graphics, 5(1), pp. 47-61, 1999.

[Ross01] Rossignac, J., Safonova, A. and Syzmczak, A., *3D Compression Made Simple: Edgebreaker on a Corner-Table*, Invited lecture at the Shape Modeling International Conference, Gemoa, Italy, 2001.

[RoSz99] Rossignac, J. and Szymczak, A., *Wrap&Zip decompression of the connectivity of triangle meshes compressed with Edgebreaker*, Computational Geometry, Theory and Applications, 14(1/3), pp. 119-135, November 1999.

[RuPl97] Ruas, A. and Plazanet, C. *Strategies for automated generalization*. Kraak, M. J., Molenaar, M. and Fendel, E. (Eds.) Advances in GIS Research II (Proceedings of the 7th International Symposium on Spatial Data Handling), Taylor & Francis, London, pp 319-36, 1997.

[Ruff00] Ruff, *3D-Stadtmodelle*. Workshop Proceedings, Fraunhofer IGD, Darmstadt, 2000.

[Same84] Samet, H., *The quadtree and related hierarchical data structure.* ACM Comput. Survey 20, 4, pp. 271-309, 1984.

[Same94] H. Samet. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley, 1994.

[SDE97] SDE Technical Documentation http://www.esri.com/base/products/sde/sde_api.html, 1997.

[Sens98] Homepage of the Sense8 Corporation http://www.sense8.com., 1998.

[Schu99] Schulz, T., *Konzeption und Implementierung eines Verfahrens zur progressiven Datenübertragung in einem WWW-basierten 3D-Geo-Informationssystem*, Diplomarbeit, TU-Darmstadt, supervisior Prof. Dr. Encarnação 1999.

[Stöp87] Stöppler, H.W., *Die automatisierte Liegenschaftskarte (ALK) - Überblick.* Nach Öffentl. Verm. Dienst (NÖV) NRW, pp 64-88, 1987.

[Stro98] Strothotte, T., *Computational Visualisation. Graphics, Abstraction and Interactivity.* Springer Verlag, Berlin - Heidelberg - New York, 1998.

[Szym01] Szymczak, A., King, D. and Rossignac, J., *An Edgebreaker-based efficient compression scheme for regular meshes*, Journal of Computational Geometry: Theory and Applications, 20(1-2), pp. 53-68, 2001.

[TaRo98] Taubin, G., and Rossignac, J., *Geometric Compression through Topological Surgery*, ACM Transactions on Graphics, 17(2), pp. 84-115, 1998.

[TsHa93] Tzschach, H., and Haßlinger, G., *Codes für den störungsfreien Datentransfer*, Oldenburg, 1993.

[ToGo98] Touma, C., and Gotsman, C., *Triangle Mesh Compression*, Proceedings Graphics Interface 98, pp. 26-34, 1998.

[UlIs97]	Ullmer, B., and Ishii, H., *The metaDESK: Models and prototypes for tangible user interfaces*. Proceedings of Symposium on User Interface Software and Technology (UIST ’97), ACM Press, New York, pp. 223-232, 1997.
[Verb99]	Verbree, E., van Maren, G., Germs, R., Jansen, F. and Kraak, M.-J. *Interactino in virtualk world views - Linking 3D GIS with VR*, International Journal of Geographic Information Science, 13(4), 1999.
[Wand92]	Wandmacher, J., *Software-Ergonomie*. de Gruyter, 1992
[Weib91]	Weibel, R., *Amplified Intelligence and Rule Based Systems*. Buttenfield, B.P. and McMasters, R.B. (Eds) Map Generalization: Making Rules for Knowledge Representation, Longman, London, pp 172-186, 1991.
[Weib94]	Weibel, R., *Geographic Information Systems (GIS) and visualization*. State-of-the-Art Report, Eurographics 94, Oslo, 1994.
[Will70]	Willard, S., *General Topology*, Reading, Addison-Wesley, 1970
[Zlat00]	Zlatanova, S., *3D-GIS for Urban Environments*, ITC Dissertation Number 69, Delft, 2000.
[ZlTe98]	Zlatanova, S., and Tempfli, K., *Data Structuring and Visualization of 3D Urban Data*,in: Proceedings of AGILE conference, Enschede, The Netherlands, 1998.
[ZlVe00]	Zlatanova, S. and Verbree, E., *A 3D topological model for augmented reality*, Proceedings of the Second International Symposium on Mobile Multimedia Systems & Applications, Delft, The Netherlands, pp.19-26, 2000.

Curriculum Vitae

Volker Coors
Bessunger Str. 12 A
64347 Darmstadt, Germany

Date of birth: 21 September 1968
Place of birth: Walsrode

Family status:	Married, 2children

School education

June 1987:	Allgemeine Hochschulreife

University education

Oct. 1992 - May 1997:	Studied computer science with a minor in civil engineering
Sept. 30, 1994:	Vordiplom (two-year degree)
May 28, 1997:	Diplom (equivalent to a master’s degree); overall evaluation "passed with distinction"

Post-University

Aug. 1997 - Sept.2002:	Scientific employee at Fraunhofer Gesellschaft, Institute for Computer Graphics, Department Graphic Information Systems
July 2000:	Winner of the BMBF Contest Virtual and Augmented Reality for the GEIST project
Nov. 2000 - May 2001:	German Academic Exchange Service scholarship for research at Georgia Institute of Technology, Atlanta, USA
since Sept. 2002:	Professor at the Stuttgart University of Applied Sciences in the subjects of surveying, computer science and mathematics