Topologische Konsistenzanalyse von großen multidimensionalen geographischen Datenbeständen

Zur Erlangung des akademischen Grades einer
DOKTOR-INGENIEURIN
von der KIT-Fakultät für Bauingenieur-, Geo- und Umweltwissenschaften des
Karlsruher Instituts für Technologie (KIT)
genehmigte

DISSERTATION

von
Frau Dipl.-Ing. Anna Katarina Giovanella
aus Erbach/Odw.

Tag der mündlichen Prüfung: 22.10.2021

Referent: Prof. Dr.-Ing. Stefan Hinz
Korreferent: Prof. Dr. rer. nat. Martin Breunig

Karlsruhe 2022

Danksagungen

Stefan Hinz, Patrick Bradley und Sven Wursthorn sei für die Betreuung der Arbeit und die Unterstützung in Form von wertvollen Diskussionen und Anregungen gedankt.

Diese Arbeit leistet einen Beitrag zum Forschungsprojekt „Topologisch konsistentes Modellieren im raumzeitlichen Kontext mit bewegten Objekten im Stadtmodell“, welches von der Deutschen Forschungsgemeinschaft unter dem Förderkennzeichen BR 3513/12-1 gefördert wird. An dieser Stelle sei der Deutschen Forschungsgemeinschaft für die Unterstützung gedankt.

Kurzfassung

Die Topologie von geographischen Datenbeständen, wie beispielsweise Gebäude- oder Stadtmodellen, wird häufig durch sogenannte *Boundary representation*-Modelle beschrieben. Dabei werden Objekte durch ihre begrenzenden Oberflächen beschrieben. Dies führt in Kombination mit der Geometrie der Objekte zu einem Problem, da dem geometrischen Modell notwendigerweise ebenfalls eine zugrunde liegende Topologie zugeordnet ist. Um die Topologie korrekt darzustellen, ist jedoch die Korrektheit des Inzidenzgraphen des *Boundary representation*-Modells erforderlich. Dies ist nur dann der Fall, wenn die dem geometrischen Modell zugrunde liegende Topologie mit der dem *Boundary representation*-Modell zugrunde liegenden Topologie übereinstimmt, also die Daten im Sinne dieser Arbeit topologisch konsistent sind.

Um zu ermöglichen, dass sich topologische Abfragen nur auf den Inzidenzgraphen stützen, wird ein neuer Begriff der topologischen Konsistenz eingeführt, der mögliche topologische Unterschiede zwischen dem Inzidenzgraphen und der aus der Geometrie stammenden Topologie erfasst. Schnittmatrizen beschreiben dann mögliche Typen topologischer Konsistenz und Inkonsistenz. Als Anwendung wurde untersucht, welche Matrizen als Schnittmatrizen auftreten können und wie Matrizen aus topologisch konsistenten Daten aussehen und es wurde gezeigt, dass Schnittmatrizen topologisch konsistenter Daten Diagonalmatrizen sind. Aus Nebendiagonal-Elementen lässt sich auf Inkonsistenzen rückschließen, aber zur Unterscheidung verschiedener Formen topologischer Inkonsistenz ist die hier definierte Schnittpunktmatrix nur ein erster Indikator. Einige Matrizen sind mehrdeutig: Sie können sowohl aus konsistenten als auch aus inkonsistenten Konfigurationen stammen.

Für mehrere CityGML-Datensätze wurde die topologische Konsistenz überprüft und eine erste Klassifizierung topologischer Inkonsistenzen durchgeführt.

Die Analyse wurde auf einem räumlichen Datenbanksystem durchgeführt, in das die Datensätze importiert wurden. Die Analyse von CityGML-Datensätzen hat gezeigt, dass viele Datensätze der realen Welt viele topologisch inkonsistente Polygonpaare enthalten. Es wurde beobachtet, dass selbst wenn Daten den val3dity-Test erfüllen, sie immer noch topologisch inkonsistent sein können. Das heißt im Fall von CityGML ist es möglich, nach dem Standard korrekt zu modellieren und trotzdem ein topologisch inkonsistentes Modell zu haben. Andererseits wird gezeigt, dass der ISO 19107-Standard dem Begriff der topologischen Konsistenz in dieser Arbeit äquivalent ist. Eine Folge davon ist, dass CityGML-Datensätze nicht dem ISO 19107-Standard entsprechen, wenn nicht von Anwenderseite auf die korrekte Modellierung geachtet wird.

Da effiziente topologische Abfragen auf die Korrektheit des Inzidenzgraphen angewiesen sind, sind die hier untersuchten Daten für eine über die bloße Visualisierung hinausgehende Analyse nicht geeignet. Folglich können sich topologische Abfragen in aktuellen CityGML-Daten nicht nur auf den Inzidenzgraphen verlassen, sondern müssen immer aufwändige geometrische Berechnungen durchführen, wenn korrekte Ergebnisse erwartet werden. Wird ein Geometriemodell in CityGML aus Punktwolkendaten oder auf andere Weise automatisch erstellt, ist eine Prüfung auf topologische Konsistenz im Sinne dieser Arbeit erforderlich. Darüber hinaus ist es notwendig, Wege zu finden, um topologisch inkonsistente Daten zu „heilen“, möglicherweise abhängig von der Art der angetroffenen topologischen Inkonsistenzen.

Abstract

The topology of geographic data, such as building or city models, is often described by so-called boundary representation models. Objects are defined by their limiting surfaces. In combination with the geometry of the objects, this leads to a problem since the geometric model necessarily also has an underlying topology associated with it. However, in order to correctly represent the topology, the correctness of the incidence graph of the boundary representation model is needed. This is only the case if the topology underlying the geometric model coincides with the topology underlying the boundary representation model – in other words, if the data are topologically consistent in the sense of this work.

In order to allow topological queries to rely on the incidence graph only, a new notion of topological consistency is introduced that captures possible topological differences between the incidence graph and the topology coming from geometry. Intersection matrices then describe possible types of topological consistency and inconsistency. As an application, it was examined which matrices can occur as intersection matrices, and how matrices from topologically consistent data look and shown that intersection matrices of topologically consistent data are diagonal matrices. Inconsistencies can be inferred from off-diagonal elements, but the intersection matrix defined here is only a first indicator for distinguishing different forms of topological inconsistency. Some matrices are ambiguous: they can come from both, consistent and inconsistent configurations.

The topological consistency was checked for several CityGML datasets, and a first classification of topological inconsistencies is performed. The analysis was carried out on a spatial database system into which the datasets have been imported. The analysis of CityGML data sets has shown that many

real-world data sets contain many topologically inconsistent pairs of polygons. It was observed that even if data satisfy the val3dity test, they can still be topologically inconsistent. That means in the case of CityGML, it is possible to model correctly according to the standard and still have a topologically inconsistent model. On the other hand, it is shown that the ISO 19107 standard is equivalent to the notion of topological consistency in this work. A consequence of this is that CityGML datasets do not correspond to the ISO 19107 standard if the user does not pay attention to correct modeling.

As efficient topological queries rely on the correctness of the incidence graph, it follows that the data studied here are not suitable for analysis which goes beyond mere visualisation. Consequently, topological queries in present CityGML data cannot rely on the incidence graph only, but must always make costly geometric computations if correct results are to be expected. When producing a geometry model in CityGML from point cloud data or in some other way automatically, it is necessary to include a check for topological consistency in the sense of this work. Furthermore, a desideratum is to find ways of healing topological inconsistent data, possibly depending on the type of topological inconsistencies encountered.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

In diesem Kapitel wird als Einführung ins Thema zunächst auf die Motivation und Relevanz der Arbeit eingegangen und anschließend wird der Stand der Forschung vorgestellt. Danach werden die Zielsetzung und Beiträge dieser Arbeit erarbeitet und abschließend wird die Gliederung der Arbeit beschrieben.

1.1 Motivation und Relevanz der Arbeit

Heutezutage erleben wir, dass es seit mehr als dreißig Jahren zu einer technologischen Erweiterung in Bezug auf die Erfassung und Verarbeitung von raumbezogenen Daten gekommen ist, die eine digitale Grundlage für das Management der Raumentwicklung bildet. Dies hat zu einer Erweiterung des Geodatenmarktes geführt. Neue Technologien, die in Computeranwendungen vorgestellt werden, haben die Anzahl der Benutzer dieser Produkte erheblich erweitert. Die Philosophie der Sammlung von Geodaten hat sich geändert. Analoge Karten und Pläne, die auf Papier gedruckt wurden, wurden durch digitale Datenbanken ersetzt, die ihre Darstellung auf die für einen bestimmten Benutzer optimale Weise ermöglichen. Darüber hinaus bieten digitale räumliche Datenbanken die Möglichkeit einer weiteren Aktualisierung durch Benutzer.

Ohne das Geodatenmanagement könnten die heutigen Herausforderungen bei Big-Data-Anwendungen wie Erdbeobachtung, Integration von Geoinformationssystemen (GIS) oder Gebäudeinformationsmodellen (BIM) und 3D/4D-Stadtplanung nicht gelöst werden. Darüber hinaus spielt das Geodatenmanagement eine wichtige Rolle im Zusammenspiel von Datenerfassung, Datenmo-

dellierung, Datenvisualisierung und Datenanalyse. Es ermöglicht die kontinuierliche Verfügbarkeit von Geodaten und die Reproduzierbarkeit der Geodatenanalyse. In Breunig et al. (2020) werden fünf Meilensteine der Geodatenmanagementforschung vorgestellt, die im letzten Jahrzehnt erreicht wurden und es wird ein Ausblick auf deren zukünftige Ausrichtung gegeben. Einen dieser Meilensteine stellt der theoretische Fortschritt dar, welcher durch die Einführung der Topologie als Schlüsselkonzept des Geodatenmanagements erreicht wird. Die Topologie wird als leistungsstarkes und allgemeines Konzept zur Analyse von GIS- und BIM-Datenstrukturen und räumlichen Beziehungen vorgestellt, die in neuen Anwendungen wie Smart Cities und digitalen Zwillingen von großer Bedeutung sein werden.

In den letzten Jahrzehnten wurden 3D-Stadtmodelle überwiegend zur Visualisierung verwendet. Heute werden sie jedoch zunehmend in einer Reihe von Bereichen und für eine Vielzahl von Aufgaben eingesetzt, die über die Visualisierung hinausgehen, wie zum Beispiel Stadt- und Raumplanung, Facility Management, Simulation von Lärm-, Abgas- oder Hochwasserausbreitung und viele weitere. Die ersten Anwendungen von 3D-Stadtmodellen waren oft Insellösungen, die für spezielle Anwendungen und Vorhaben realisiert wurden. Das jeweilige Stadtmodell war nur für eine bestimmte Gruppe oder ein spezifisches Einsatzgebiet erstellt, dabei in der Regel auf einen kleinen Stadtraum begrenzt und nur mit einer bestimmten Soft- und Hardware nutzbar. Somit wurde das Stadtmodell meist nach Projektabschluss wertlos.

Mit der Zeit entstand der Wunsch, flächendeckende 3D-Stadtmodelle vorzuhalten, die verwaltungsweit als Grundlage für unterschiedliche Projekte, Aufgaben und Systeme genutzt werden könnten. Zunächst fehlte jedoch ein Datenmodell, welches die unterschiedlichen Anforderungen abdecken konnte.

Die Lösung wurde in der Schaffung eines offenen Datenmodells, das die unterschiedlichen Anforderungen an die Modellierung der Stadttopografie vereint, gesehen. Aus diesem Grund hat sich 2002 in Deutschland eine Arbeitsgruppe, die Special Interest Group 3D (SIG3D), gebildet. In der SIG3D kamen über siebzig Vertreter aus Wirtschaft, Forschung und öffentlichem Dienst zusammen, um ein geeignetes Informationsmodell zu entwickeln. 2004 brachte die SIG3D ihren Vorschlag für ein Datenmodell beim Open Geospatial Consortium (OGC) ein. Seit August 2008 ist CityGML als Standard des OGC

anerkannt und im April 2012 in der Version 2.0.0 verabschiedet.

Modelliert werden Stadt- und Landschaftsobjekte, insbesondere das Gelände, Gebäude, Wasser- und Verkehrsflächen, Vegetation, Stadtmöblierung und Landnutzungen. Jedes Objekt kann in unterschiedlichen Level of Detail (LoD) vorkommen. Dabei werden neben der Geometrie und dem Aussehen insbesondere auch die Semantik und die Topologie der Objekte beschrieben. Eine ausführliche Beschreibung des geometrisch-topologischen Modells vom CityGML folgt in Abschnitt 2.2.1.

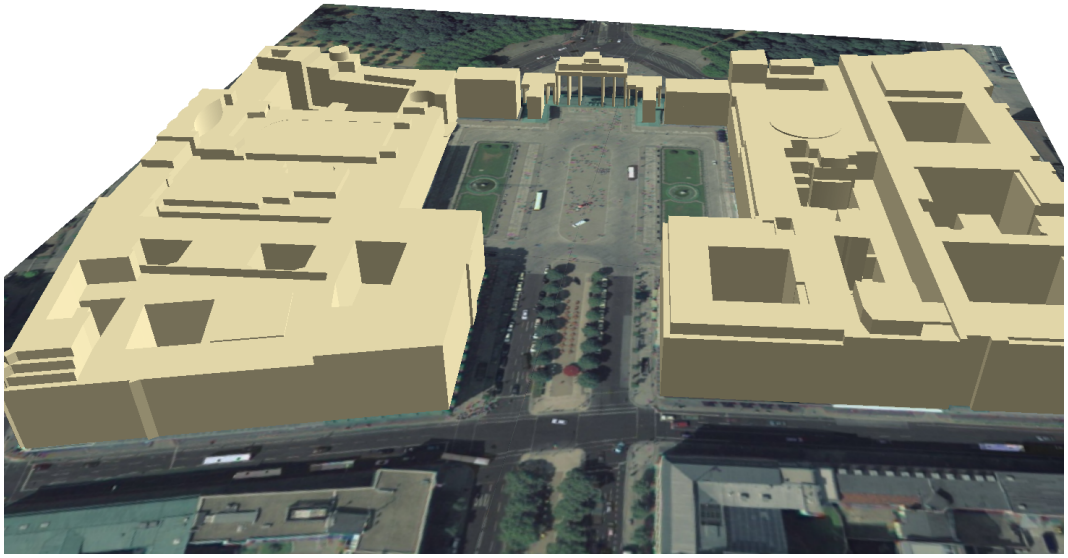


Abbildung 1.1: CityGML-Modell in LoD2 vom Pariser Platz in Berlin mit dem Brandenburger Tor

Biljecki et al. (2015b) geben einen Überblick über den Stand der Technik bezüglich der Verwendung von 3D-Stadtmodellen in mehreren Bereichen, basierend auf einer umfassenden Literaturstudie, die Hunderte von Forschungsarbeiten, technischen Berichten und Online-Ressourcen umfasst. Sie definieren eine hierarchische Terminologie (*räumliche Operationen, Anwendungsfälle, Anwendungen*) und entwickeln eine theoretische Begründung, um die vielfältigen Einsatzmöglichkeiten von 3D-Stadtmodellen zu segmentieren und zu kategorisieren. Im Anschluss an dieses Framework stellen sie eine Liste identifizierter Anwendungsfälle von 3D-Stadtmodellen (mit einer Beschreibung der einzelnen Modelle) und deren Anwendungen bereit. Die Studie zeigt, dass 3D-Stadtmodelle in zahlreichen Anwendungsfällen eingesetzt werden, die Teil von

mehr als hundert Anwendungen sind.

Deutschlandweit erhebt jedes Bundesland die Gebäude seiner Landesfläche als CityGML Datensatz. Diese Daten geben die Bundesländer an die „Zentrale Stelle für Hauskoordinaten und Hausumringe“ (ZSHH) als LoD1 und LoD2 im CityGML Format ab und vertreiben diese selbst an Architekten, Stadtplaner und ähnliche Nutzer. Zunehmend bieten die Bundesländer oder einzelne Städte ihre CityGML Daten als Open Data an. Europaweit hat sich der CityGML Standard so weit verbreitet, dass er nicht mehr aus den 3D-Stadtmodellen der Beneluxstaaten, Skandinaviens, der Schweiz, Österreichs, Italiens, Spaniens, Frankreichs, Großbritanniens, der Türkei, Tschechiens, Polens, des Baltikums sowie aus Teilen Russlands und vielen weiteren Staaten wegzudenken ist. Weltweit wird CityGML unter anderem in Singapur, USA, Kanada, Australien, China, Japan und Indien genutzt.

In Biljecki et al. (2016a) wird die Qualität der derzeit verfügbaren CityGML-Datensätze untersucht, das heißt sie validieren die Geometrie und die Topologie der 3D-Grundelemente (*Solid* und *MultiSurface*) und prüfen, ob die Semantik der Gebäuderandflächen korrekt ist oder nicht. Ihre Ergebnisse zeigen, dass CityGML-Datensätze ohne Fehler selten sind und die (nahezu) fehlerlosen meist einfache LoD1-Modelle sind, was sich mit den eigenen Erfahrungen, die im Rahmen der vorliegenden Arbeit gemacht wurden, deckt. Eine weitere Hauptbeobachtung von Biljecki et al. (2016a) ist, dass viele dieser Fehler durch einfache Änderungen an der Modellierungssoftware automatisch behoben oder verhindert werden könnten. Sie finden außerdem heraus, dass die häufigsten topologischen Fehler darin bestehen, dass Polygone nicht richtig orientiert sind und dass Geometrien nicht richtig „gefangen“ werden.

Im Unterschied dazu stellt der Ansatz der vorliegenden Arbeit einerseits eine weitere Differenzierung dieses Fehlertyps dar und andererseits muss hier kein Gebäude nur aus *Solids* bestehen, solange sich die Polygone in gemeinsamen Randelementen schneiden. Zum Beispiel Balkone, Vordächer oder Unterstände und Ähnliches weisen oft Geometrien auf, die keine geschlossene Hülle bilden, das heißt es sind nicht geschlossene Oberflächen.

CityGML hat sich demnach zu einem weit verbreiteten Format für städtebauliche Daten in verschiedenen Detailebenen (LoDs) entwickelt. Wenn die in CityGML gespeicherten Daten für eine effiziente Analyse jenseits der Visuali-

sierung verwendet werden sollen, müssen sie topologisch konsistent sein. Andernfalls ergeben topologische Abfragen falsche Ergebnisse. In der vorliegenden Arbeit zeigt sich jedoch, dass reale CityGML-Datensätze meist unterschiedliche topologische Inkonsistenzen aufweisen und dass es in CityGML möglich ist, gemäß dem Standard korrekt zu modellieren und dennoch ein topologisch inkonsistentes Modell zu haben.

1.2 Stand der Forschung

In diesem Abschnitt wird der Stand der Forschung zum Thema topologische Konsistenz vorgestellt. Dies ist zunächst nur ein allgemeiner Überblick, auf Literatur, welche konkrete Relevanz für die vorliegende Arbeit hat, wird an anderer Stelle noch ausführlicher eingegangen.

Die Literatur enthält verschiedene unterschiedliche Begriffe topologischer Konsistenz. In Dušan and Branislav (2004) geht es um Elemente der Geodatenqualität als informationstechnische Unterstützung für eine nachhaltige Entwicklungsplanung. Dort wird topologische Konsistenz schlicht als Fehlen topologischer Fehler (z.B. nicht geschlossene Polygone, hängende Knoten usw.) definiert.

In Li (2006) wird ein direkter und einfacher Algorithmus zur Generierung von pfadkonsistenten Netzwerken von RCC8-Basisbeziehungen (Randell et al., 1992) bereitgestellt. Als Ergebnis wird auch gezeigt, dass jedes konsistente Netzwerk von RCC8-Beziehungen eine Realisierung in der digitalen Ebene (mit entweder 4- oder 8-Verbindungen) und in einem beliebigen RCC-Modell hat.

Kang and Li (2005) schlagen eine Reihe von Regeln vor, um die topologische Konsistenz für den Minimierungsvorgang bei der Generalisierung räumlicher Datenbanken zu bewerten. Die Regeln basieren auf einer strengen Klassifizierung topologischer Eigenschaften in einem Minimierungsvorgang. Mit diesen Regeln können sie inkonsistente topologische Änderungen im Generalisierungsprozess erkennen und die Qualität der abgeleiteten Datenbanken verbessern.

Rodriguez et al. (2010) schlagen Maßzahlen vor, um den Grad der Verletzung einer topologischen Abhängigkeitsbedingung durch in einer räumlichen Datenbankinstanz gespeicherte Geometrien zu bewerten. Sie schlagen auch vor, wie diese Maßzahlen aggregiert werden können, um die Datenqualität einer Da-

tenbankinstanz global zu bewerten, sodass sie einen Vergleich der Datenbankinstanzen im Hinblick auf ihre Bedingungserfüllung ermöglichen. Sie liefern eine experimentelle Bewertung dieser Maßzahlen anhand von synthetischen und realen Daten. Sie validieren ihre Maßzahlen durch Analyse ihrer Korrelation mit der semantischen Distanz der topologischen Beziehungen und prüfen, dass je mehr Geometrien zufällig geändert werden, um Datenbankinstanzen inkonsistent zu machen, ihr globales Datenqualitätsmaß umso mehr abnimmt, um seine Sensibilität für die eingeführten Bedingungsverletzungen zu zeigen.

Bradley (2015) gibt einen Überblick über topologische Datenmodelle und führt topologische Konsistenz im Kontext von Smart Cities ein. Es wird eine genaue Definition der topologischen Konsistenz im zweidimensionalen Fall gegeben und eine Anwendung auf Datenmodelle diskutiert.

Jahn et al. (2017) führen eine erste Definition der topologischen Konsistenz ein, die die Geometrie und den Inzidenzgraphen im Kontext verteilter großer geographischer Daten in Beziehung setzt und ein Maß für topologische Inkonsistenz basierend auf Betti-Zahlen von endlichen, partiell geordneten Mengen definiert.

Gröger and Plümer (2011) liefern die theoretischen Grundlagen für effiziente und zuverlässige Werkzeuge für die Konsistenzprüfung, indem sie eine axiomatische Charakterisierung von 3D-Stadtmodellen definieren. Sie fordern ein konsistentes Modell, um eine endliche Tessellation vom $\mathbb{R}^3$ darzustellen. Es wird gezeigt, wie ein 3D-Stadtmodell in Komponenten zerlegt werden kann, die entweder zu einer Kreisscheibe, einer Kugel oder einem Torus topologisch äquivalent sind und die Modellierung des Geländes, von Gebäuden und anderen Konstruktionen sowie von Brücken und Tunneln ermöglichen. Dies schließt Polygone aus, die nicht an einen Volumenkörper (*Solid*) angrenzen, wie zum Beispiel ein Gebäude mit freien Wänden.

Ledoux and Meijers (2011) stellen ein Extrusionsverfahren vor, um topologisch konsistente 3D-Stadtmodelle aus planaren Polygonen zu erstellen. Es basiert auf der Verwendung einer bedingten Triangulation, ist konzeptionell einfach und bietet große Flexibilität zum Erstellen von Stadtmodellen in verschiedenen Formaten (zum Beispiel CityGML oder einer oberflächenbasierten Darstellung). Sie haben das Verfahren implementiert, mit realen Datensätzen getestet und validiert. Es wird ein Begriff der topologischen Konsistenz de-

finiert, der ein Sonderfall der in dieser Arbeit betrachteten Definition ist. Es werden zum Beispiel keine Polygone mit Löchern oder Durchstößen zugelassen.

Anwendungen einer solchen topologischen Konsistenz werden zum Beispiel in Steuer et al. (2015) beschrieben, wo das Volumen von Gebäuden in CityGML durch Überwindung topologischer Fehler approximiert wird. Die Methode basiert auf einer Voxelisierung und einer Verallgemeinerung des Punkt-in-Polygon-Tests auf drei Dimensionen. Abgesehen von der Berechnung von Volumina könnte der vorgeschlagene Ansatz möglicherweise nützlich sein für zahlreiche Anwendungen wie das Heilen von Gebäudemodellen, das Indoor-Routing oder die Modelltransformation.

In Ledoux (2013) und in Ledoux (2018) ist eine Open-Source-Software (val3dity) zum Prüfen von Solids gegen die ISO/OGC-Spezifikationen beschrieben. Dieses Werkzeug überprüft auch, ob sich Paare unterschiedlicher Solids in ihren Innenräumen schneiden. Es wird jedoch nicht überprüft, ob sie sich in ihren Rändern topologisch inkonsistent schneiden, da dies vom CityGML-Standard nicht gefordert wird. Dies würde die Überprüfung von sich schneidenden Polygonpaaren erfordern.

Im Unterschied zu val3dity ist es das Ziel der vorliegenden Arbeit, die topologische Konsistenz unabhängig von der Konformität mit den entsprechenden Standards zu überprüfen. Zum Beispiel wurde das topologisch inkonsistente Beispielhaus, das hier zur Veranschaulichung verwendet wird (siehe Abbildung 3.3), durch val3dity laufen gelassen und wurde als „valid“ befunden, wenn es als eine Kombination von *MultiSurfaces* modelliert war, was gemäß dem CityGML-Standard korrekt ist. Der Vergleich von val3dity mit der in der vorliegenden Arbeit verwendeten Methodik wurde in Giovanella et al. (2018) versucht, ist jedoch nur möglich, wenn Gebäudehüllen als *Solid* modelliert werden, was eine äußere Schale minus mögliche innere Schalen bedeutet. Die Zuordnung von Polygonen zu Gebäuden wird problematisch, wenn die Gebäudehülle als *MultiSurface*-Geometrien anstelle von einem *Solid* modelliert wird. Aus diesem Grund können Gebäude mit inkonsistenten Polygonen und Polygonpaaren nur dann bestimmt werden, wenn sie als *Solid* modelliert sind. In Giovanella et al. (2019) wird nochmal etwas ausführlicher auf den Vergleich der hier verwendeten Methodik mit val3dity eingegangen.

Ein weiteres Tool zur Validierung von 3D-Stadtmodellen stellt das Projekt

CityDoctor (Wewetzer et al., 2013) dar. Ziel des Projekts war es einen gemeinsamen Prüf- und Heilkern zur Validierung und Reparatur von CAD- und Stadtmodellen zu entwickeln. Im Rahmen des Projektes wurden verschiedene Validierungsregeln erarbeitet und mathematisch fixiert. Darauf aufbauend werden in Wagner et al. (2013) und in Wagner et al. (2014) Aufbau und Spezifikation eines Prüfplans beschrieben, der anwendungsspezifische Erfordernisse, wie etwa Geschlossenheit von Volumenkörpern (Wasserdichtigkeit) oder die Verwendung bestimmter Geometriearten berücksichtigt. Ein Modell zur standardisierten Dokumentation und Speicherung von Prüfergebnissen wird vorgestellt. Damit kann die Eignung des Datensatzes für die vorgesehene Anwendung bewertet werden.

In Coors et al. (2020) wird am Beispiel der Heizungsbedarfssimulation eine neue Methode zur Spezifikation anwendungsabhängiger Anforderungen an 3D-Stadtmodelle vorgestellt. Ein modularer Satz von geometrischen und semantischen Anforderungen wurde definiert. Der Artikel schlägt einen Validierungsplan vor, der abhängig von der Anwendung jeweils einen Satz von Anforderungen festlegt. Zunächst erfolgt die Validierung dort ähnlich wie bei val3dity nur bezüglich Konformität zur *XML Schema Definition* von CityGML, allerdings ist es aufgrund der Modularität des Ansatzes denkbar den Satz der Prüfbedingungen so zu erweitern, dass auch auf topologische Konsistenz im Sinne der hier vorliegenden Arbeit geprüft wird.

Im Folgenden sollen aufbauend auf dem hier umrissenen Stand der Forschung, die Zielsetzung und die Beiträge dieser Arbeit beschrieben werden.

1.3 Zielsetzung und Beiträge der Arbeit

Aufbauend auf den beiden vorherigen Abschnitten werden in diesem Abschnitt die Zielsetzung und die Beiträge der Arbeit herausgearbeitet.

Räumliche Datenmodelle, die zu Analysezwecken über die reine Visualisierung hinaus entwickelt wurden, müssen folgende Eigenschaften erfüllen:

- Korrektheit
- Redundanzfreiheit
- Konsistenz

Korrektheit bezieht sich hierbei auf die Korrektheit der Geometrie und der Topologie. Wenn Daten redundant gespeichert werden, entstehen Konsistenzprobleme. Das Speichern von Daten ohne Redundanz alleine garantiert jedoch keine Konsistenz. Beispielsweise verfügt CityGML über ein Geometriemodell und über ein separates Topologiemodell. Da der Geometrie selbst eine Topologie zugrunde liegt, kann CityGML nicht das Fehlen von Widersprüchen zwischen der aus der Geometrie stammenden Topologie und seinem topologischen Modell garantieren. Es ist ein Ziel dieser Arbeit, die mögliche Nachlässigkeit in Bezug auf diese Art von topologischer Konsistenz aufzuzeigen, wenn CityGML zur Modellierung räumlicher Daten verwendet wird. Diese Inkonsistenzen sollen, mit Hilfe einer in dieser Arbeit definierten Schnittmatrix, automatisiert gefunden und klassifiziert werden. Die Schnittmatrix wird in Abschnitt 3.1 im Detail vorgestellt.

Topologische Abfragen wie „finde alle Randobjekte des Objekts A“ oder „wie nahe sind die Objekte A und B topologisch“, können erwartungsgemäß am effizientesten unter Verwendung des Inzidenzgraphen des topologischen Modells für gegebene räumliche Daten beantwortet werden.

Der Inzidenzgraph ist eine endliche Repräsentation der Topologie eines räumlichen Modells. In Bradley and Paul (2010) wird, motiviert durch Forschungen darüber, wie Topologie eine hilfreiche Grundlage für das *Building Information Modeling (BIM)* bilden kann, eine relationale Datenbankversion der Begriffe Kettenkomplex und Kettenkomplexmorphismus definiert und zum Speichern von CW-Komplexen und deren Morphismen verwendet. In vielen Fällen kann dies ohne Verlust topologischer Informationen erfolgen. Die Äquivalenz von Kategorien zwischen Mengen mit binären Relationen und Alexandrov-Räumen (Alexandrov, 1937) wird nachgewiesen und verwendet, um die relationalen Komplexe in den allgemeineren Rahmen topologischer Datenbanken aufzunehmen. Für letzteres wird eine topologische Version einer relationalen Abfragesprache definiert, indem die üblichen relationalen Algebraoperatoren in topologische Konstruktionen übertragen werden. Letztendlich wird bewiesen, dass eine solche topologische Version der relationalen Algebra im Allgemeinen in der Lage sein muss, den transitiven Abschluss einer Beziehung zu berechnen. Der Inzidenzgraph hat eine demnach einfache relationale Datenbankdarstellung durch eine Tabelle für die Objekte und eine andere für die

topologiedefinierenden Relationen. Es wird auch gezeigt, dass seine Speicherkomplexität in der Anzahl der Objekte quadratisch ist und dies ist im Allgemeinen die effizienteste zu erwartende Komplexität (Paul, 2008). Darüber hinaus ist dieses Datenmodell insofern universell, als es eine beliebige endliche topologische Repräsentation von Daten erfasst (Bradley and Paul, 2010).

Topologische Nähe von A und B bedeutet hierbei, dass es eine Sequenz $A = x_0,x_1,\ldots,x_m = B$ gibt, so dass x_i der Rand von x_{i+1} oder x_{i+1} der Rand von x_i ist. Der Inzidenzgraph ist eine Struktur, die die Beziehung „ist begrenzt durch“ modelliert, und die Beantwortung von Abfragen muss idealerweise nicht auf die Anwendung geometrischer Operationen wie die Schnittberechnung zurückgreifen, da der Inzidenzgraph die Topologie korrekt modelliert. Geometrische Operationen werden besonders dann rechenaufwändig, wenn sich viele Objekte geometrisch nahe sind, aber nicht topologisch. Sind Objekte hingegen topologisch nahe, aber nicht geometrisch, dann werden sie bei der topologischen Abfrage nicht berücksichtigt, wenn die Geometrie als Grundlage verwendet wird. Indexstrukturen, die auf euklidischer Geometrie basieren wie zum Beispiel der R-Baum, werden suboptimal, weil sie Objekte berücksichtigen müssen, die weiter weg als notwendig sind. Ein Erfordernis ist also ein topologischer Index, der nur auf dem topologischen Modell beruht und die zugrunde liegende Geometrie ignoriert. Eine notwendige Voraussetzung für die Korrektheit eines solchen Ansatzes ist, dass die dem geometrischen Modell zugrunde liegende Topologie mit der des topologischen Modells übereinstimmt. Mit anderen Worten, es wird angenommen, dass das Modell *topologisch konsistent* ist, ein Begriff, der in dieser Arbeit präzisiert werden soll. Mögliche Anwendungen hierfür sind die Berechnung des Volumens, des Volumen-Adjazenz-Graphen für Pfadabfragen oder die Wärmeausbreitung in Gebäuden.

Im Allgemeinen ist zu betonen, dass jede topologische Abfrage auf die eine oder andere Weise die zugrunde liegende Topologie verwendet und daher natürlich auf den Inzidenzgraphen angewendet werden kann, falls die Daten im Sinne dieser Arbeit topologisch konsistent sind.

Alam et al. (2014) enthält eine Liste von Konsistenzregeln für Topologie und Semantik, in denen nicht mehr als zwei Polygone eine gemeinsame Kante haben dürfen. Validierungsergebnisse realer Stadtmodelle werden vorgestellt, um das Potenzial des Ansatzes zu demonstrieren. Ziel ist es, Defekte der Mo-

delle automatisch zu heilen und ein korrigiertes CityGML-Modell zu exportieren. Es wird gezeigt, dass Probleme auch dann bestehen, wenn die Daten standardkonform sind, was einer der Ansatzpunkte dieser Arbeit ist. Ein Ziel der vorliegenden Arbeit ist es, topologische Konsistenz unabhängig von der Konformität zu Standards der ISO beziehungsweise OGC oder Ähnlichem zu prüfen.

Der Beitrag dieser Arbeit kann wie folgt zusammengefasst werden: Es wird eine neue Definition der topologischen Konsistenz eingeführt, die die Übereinstimmung des Inzidenzgraphen mit der aus der Geometrie stammenden Topologie gewährleistet. Im Gegensatz zu den Definitionen in der Literatur werden alle möglichen Inzidenzgraphen als topologisch konsistent akzeptiert, wenn die obige Anforderung erfüllt ist. Eine Definition von Schnittmatrizen zur Erfassung der möglichen Arten von topologischer Konsistenz und Inkonsistenz wird vorgestellt und es wird gezeigt, welche Schnittmatrizen möglicherweise auftreten können und welche Schnittmatrizen aus topologisch konsistenten Daten stammen. Eine Analyse von Schnittmatrizen für Polygonpaare in verschiedenen CityGML-Datensätzen wurde durchgeführt.

1.4 Gliederung der Arbeit

Nach dem im Kapitel 1 die Motivation (in Abschnitt 1.1) und die Zielsetzung der Arbeit (in Abschnitt 1.3), sowie der Stand der Forschung (in Abschnitt 1.2) vorgestellt wurden, werden zunächst in Kapitel 2 die mathematischen Grundlagen dieser Arbeit eingeführt. Dort findet sich zunächst in Abschnitt 2.1 eine Einführung in die Grundlagen der Topologie, danach wird in Abschnitt 2.2 erklärt wie Topologie und Geometrie in CityGML modelliert werden, gefolgt von einer detaillierten Einführung in den Begriff der topologischen Konsistenz in Abschnitt 2.3. Im darauf folgenden Kapitel 3 werden dann die Methodik, die technischen Grundlagen und die Umsetzung beschrieben. In Abschnitt 3.1 wird dort die Schnittmatrix eingeführt. Diese ist ein erstes Mittel zum Erkennen der topologischen Konsistenz und zum Unterscheiden zwischen verschiedenen Arten von topologischen Inkonsistenzen, wenn die Konfiguration aus zwei Polygonen besteht. Im Abschnitt 3.2 wird das Tool *3D City Database* (3DCityDB) kurz vorgestellt. Dann wird in Abschnitt 3.3 die Implementierung

erklärt. Folgend werden in Kapitel 4 zunächst im Abschnitt 4.1 die verwendeten CityGML-Datensätze beschrieben und danach werden die Ergebnisse in Abschnitt 4.2 vorgestellt und diskutiert. Abschließend folgt eine Zusammenfassung in Abschnitt 5.1 und ein Ausblick in Abschnitt 5.2 in Kapitel 5.

Kapitel 2

Grundlagen der topologischen Modellierung

Nach einer allgemeinen Einführung in die Topologie in Abschnitt 2.1 und einer Erläuterung, wie die Topologie in CityGML modelliert wird in Abschnitt 2.2, wird in Abschnitt 2.3 eine Definition der topologischen Konsistenz eingeführt, die genau dann gegeben ist, wenn die Topologie des Inzidenzgraphen, der aus Randbeziehungen der Geometrie stammt, mit der der Geometrie zugrunde liegenden Topologie übereinstimmt.

2.1 Einführung in die Topologie

Durch diesen Abschnitt sollen die benötigten mathematischen Grundlagen dieser Arbeit aus dem Gebiet *Topologie* zur Verfügung gestellt werden. Diese Grundlagen wurden zum Teil bereits in meiner Studienarbeit und Diplomarbeit erarbeitet und wurden teilweise daraus übernommen. Weitere Inhalte kommen aus Mitschriften aus einem Topologiekurs, welcher von Dr. rer. nat. Patrick Erik Bradley durchgeführt wurde und aus Jänich (1990), Paul (2008) und Hatcher (2002).

2.1.1 Grundbegriffe

In diesem Abschnitt werden einige Grundbegriffe definiert, welche für das Verständnis des mathematischen Begriffes Topologie grundlegend sind oder welche in dieser Arbeit später noch benötigt werden.

Topologische Begriffe

In diesem Abschnitt werden einige topologische Begriffe definiert.

Zunächst soll die Definition des *topologischen Raumes* in Erinnerung gerufen werden:

Definition 2.1.1 (topologischer Raum)**.** Ein Paar $(\mathcal{X}, \mathcal{T})$, bestehend aus einer Menge $\mathcal{X}$ und einer Menge $\mathcal{T}$ ($\mathcal{T} \subseteq \mathcal{P}(\mathcal{X}) := \{\mathcal{A} \mid \mathcal{A} \subseteq \mathcal{X}\}$) von Teilmengen von $\mathcal{X}$, heißt *topologischer Raum*, wenn die folgenden Axiome gelten:

1. Die Menge $\mathcal{X}$ und die leere Menge $\emptyset$ seien in $\mathcal{T}$ enthalten. $(\mathcal{X}, \emptyset \in \mathcal{T})$
2. Die Vereinigung beliebig vieler in $\mathcal{T}$ enthaltener Mengen sei Teilmenge von $\mathcal{T}$. $(\mathcal{A} \subseteq \mathcal{T} \Rightarrow \bigcup_{A \in \mathcal{A}} A \in \mathcal{T})$
3. Der Durchschnitt zweier Mengen aus $\mathcal{T}$ sei wieder in $\mathcal{T}$ enthalten. $(A, B \in \mathcal{T} \Rightarrow A \cap B \in \mathcal{T})$

Sind diese drei Axiome erfüllt, so wird das Paar $(\mathcal{X}, \mathcal{T})$ als *topologischer Raum* und $\mathcal{T}$ als *Topologie* bezeichnet. Eine Menge $\mathcal{O}$ aus $\mathcal{T}$ ($\mathcal{O} \in \mathcal{T}$) heißt *offene* Menge.

Benutzt man den Begriff der offenen Menge, lassen sich die drei Axiome aus der Definition 2.1.1 für den topologischen Raum folgendermaßen ausdrücken:

1. Die Menge $\mathcal{X}$ und die leere Menge $\emptyset$ seien offene Mengen.
2. Die Vereinigung beliebig vieler offener Mengen sei wieder offen.
3. Der Durchschnitt zweier offener Mengen sei wieder offen.

Ein Beispiel für einen topologischen Raum ist das Paar $(\mathbb{R}, \mathcal{T})$, wobei $\mathcal{T}$ die Vereinigung „offener“ Intervalle (a, b) ist. Weitere Beispiele für topologische Räume stellen *metrische Räume* (siehe Definition 2.1.13) dar.

Zur Ergänzung des Begriffes der offenen Menge soll hier als Nächstes die Definition einer *abgeschlossenen Menge* wiederholt werden:

Definition 2.1.2 (abgeschlossene Menge)**.** Sei das Paar $(\mathcal{X}, \mathcal{T})$ ein topologischer Raum. Dann heißt eine Teilmenge $\mathcal{A}$ von $\mathcal{X}$ ($\mathcal{A} \subseteq \mathcal{X}$) *abgeschlossen*, falls das Komplement $(\mathcal{X} \setminus \mathcal{A})$ von $\mathcal{A}$ offen ist.

Es folgen die Definitionen einiger weiterer Begriffe, welche sich auf Mengen und deren Elemente beziehen:

Definition 2.1.3 (innerer Punkt)**.** Ein Element x einer Menge $\mathcal{X}$ ($x \in \mathcal{X}$) heißt *innerer Punkt* einer Teilmenge $\mathcal{A}$ von $\mathcal{X}$ ($\mathcal{A} \subseteq \mathcal{X}$), falls eine offene Teilmenge $\mathcal{U}$ von $\mathcal{X}$ ($\mathcal{U} \subseteq \mathcal{X}$) existiert, sodass x Element von $\mathcal{U}$ ($x \in \mathcal{U}$) und $\mathcal{U}$ Teilmenge von $\mathcal{A}$ ($\mathcal{U} \subseteq \mathcal{A}$) ist. Die Menge aller inneren Punkte von $\mathcal{A}$ heißt *Inneres* $\mathcal{A}^\circ$ von $\mathcal{A}$.

Definition 2.1.4 (Randpunkt)**.** Ein Element x einer Menge $\mathcal{X}$ ($x \in \mathcal{X}$) heißt *Randpunkt* einer Teilmenge $\mathcal{A}$ von $\mathcal{X}$ ($\mathcal{A} \subseteq \mathcal{X}$), wenn jede offene Teilmenge von $\mathcal{X}$, die x enthält, sowohl Punkte aus $\mathcal{A}$, als auch Punkte aus dem Komplement von $\mathcal{A}$ enthält. Die Menge aller Randpunkte von $\mathcal{A}$ heißt *Rand* $\partial\mathcal{A}$ von $\mathcal{A}$.

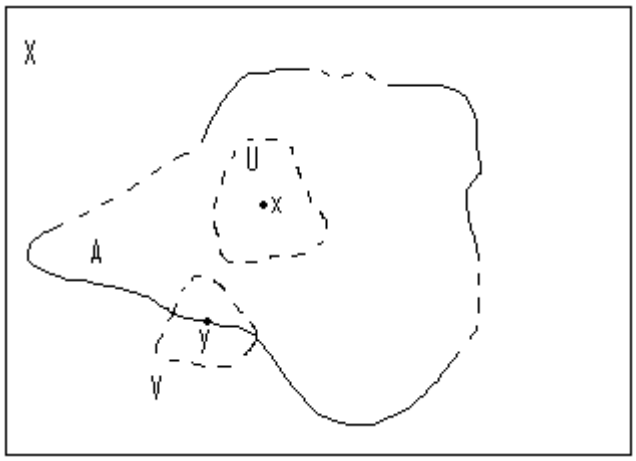


Abbildung 2.1: Innerer Punkt x und Randpunkt y

Die Definitionen 2.1.3 und 2.1.4 werden in Abbildung 2.1 nochmals grafisch verdeutlicht. Aus der Definition 2.1.4 und der Abbildung 2.1 lässt sich leicht einsehen, dass der Rand vom Rand $\partial(\partial\mathcal{A})$ (oder auch $\partial^2\mathcal{A}$) einer Menge $\mathcal{A}$ die leere Menge $\emptyset$ ist beziehungsweise, dass der Rand $\partial\mathcal{A}$ keinen Rand hat. Diese anschauliche Vorstellung davon, warum der Rand vom Rand nicht existiert, kann bei der Veranschaulichung der im Abschnitt 2.1.4 behandelten *Kettenkomplexe* hilfreich sein und wird hier aus diesem Grund explizit erwähnt.

Definition 2.1.5 (Abschluss)**.** Als *Abschluss* $\bar{\mathcal{A}}$ von $\mathcal{A}$ bezeichnet man die Vereinigung vom Inneren $\mathcal{A}^\circ$ von $\mathcal{A}$ und dem Rand $\partial\mathcal{A}$ von $\mathcal{A}$. ($\bar{\mathcal{A}} = \mathcal{A}^\circ \cup \partial\mathcal{A}$)

Hierzu ist anzumerken, dass das Innere $\mathcal{A}^\circ$ und der Rand $\partial\mathcal{A}$ einer Menge $\mathcal{A}$ immer disjunkt zueinander sind. Dies lässt sich leicht veranschaulichen, da jeder Punkt x immer nur entweder innerer Punkt oder Randpunkt der Menge $\mathcal{A}$ sein kann und nicht beides gleichzeitig. Die Menge aller inneren Punkte $\mathcal{A}^\circ$ und die Menge aller Randpunkte $\partial\mathcal{A}$ haben also keine gemeinsamen Punkte, das heißt sie sind disjunkt.

Weiter geht es mit einigen Definitionen, welche sich mit den Eigenschaften von Topologien beschäftigen:

Definition 2.1.6 (von $\mathcal{A}$ erzeugte Topologie)**.** Seien eine Menge $\mathcal{X}$ und eine Teilmenge $\mathcal{A}$ der Potenzmenge $\mathcal{P}(\mathcal{X})$ (Menge aller Teilmengen von $\mathcal{X}$) ($\mathcal{A} \subseteq \mathcal{P}(\mathcal{X})$) gegeben. Dann sei die Topologie $\mathcal{T}(\mathcal{A})$ definiert als die kleinste Topologie $\mathcal{T}$ auf $\mathcal{X}$ mit der Eigenschaft, dass $\mathcal{A}$ Teilmenge der Topologie $\mathcal{T}$ ist ($\mathcal{A} \subseteq \mathcal{T}$). Diese Topologie $\mathcal{T}(\mathcal{A})$ heißt die *von $\mathcal{A}$ erzeugte Topologie*.

Man erhält die Topologie $\mathcal{T}(\mathcal{A})$, indem man alle Topologien $\mathcal{T}$, welche $\mathcal{A}$ enthalten, schneidet ($\mathcal{T}(\mathcal{A}) = \bigcap\limits_{\mathcal{T}\supseteq\mathcal{A}} \mathcal{T}$). Denn der Durchschnitt von Topologien $\mathcal{T}$ auf $\mathcal{X}$ ergibt wieder eine Topologie $\mathcal{T}$ auf $\mathcal{X}$. Warum diese beiden Aussagen sinnvoll sind, wird in meiner Studienarbeit ausführlich behandelt und soll deshalb hier nicht weiter vertieft werden.

Definition 2.1.7 (triviale Topologie)**.** Als *triviale Topologie* $\mathcal{T}$ wird diejenige Topologie $\mathcal{T}$ auf der Menge $\mathcal{X}$ bezeichnet, welche nur die Menge $\mathcal{X}$ selbst und die leere Menge $\emptyset$ enthält ($\mathcal{T} = \{\mathcal{X}, \emptyset\}$).

Definition 2.1.8 (diskrete Topologie)**.** Diejenige Topologie $\mathcal{T}$ auf der Menge $\mathcal{X}$, welche die gesamte Potenzmenge $\mathcal{P}(\mathcal{X})$ der Menge $\mathcal{X}$ darstellt, heißt *diskrete Topologie* ($\mathcal{T} = \mathcal{P}(\mathcal{X})$).

Definition 2.1.9 (Spurtopologie)**.** Sei das Paar $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ ein topologischer Raum und die Menge $\mathcal{A}$ eine Teilmenge von $\mathcal{X}$ ($\mathcal{A} \subseteq \mathcal{X}$), dann ist die *Spurtopologie* $\mathcal{T}_\mathcal{X}|_\mathcal{A}$ (oder kurz $\mathcal{T}_\mathcal{A}$) definiert als die Topologie, welche gebildet wird aus dem Durchschnitt der Elemente O aus $\mathcal{T}_\mathcal{X}$ und $\mathcal{A}$ ($\mathcal{T}_\mathcal{X}|_\mathcal{A} := \mathcal{T}_\mathcal{A} := \{O \cap \mathcal{A} \mid O \in \mathcal{T}_\mathcal{X}\}$). Das Paar $(\mathcal{A}, \mathcal{T}_\mathcal{A})$ heißt dann *Teilraum* des topologischen Raumes $(\mathcal{X}, \mathcal{T}_\mathcal{X})$.

Der Begriff der Spurtopologie wird in Abbildung 2.2 nochmal grafisch verdeutlicht. Außerdem werden die Begriffe *stetige Abbildung* und *Homöomorphis-*

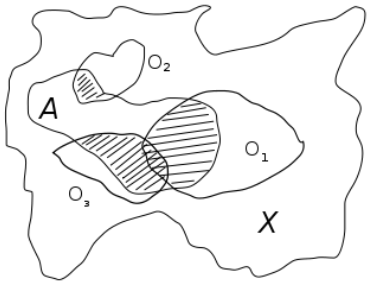


Abbildung 2.2: Spurtopologie

mus an anderer Stelle noch benötigt und sollen daher als Nächstes definiert werden:

Definition 2.1.10 (stetige Abbildung)**.** Seien $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ und $(\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ topologische Räume. Dann ist eine *stetige Abbildung* $(\mathcal{X}, \mathcal{T}_\mathcal{X}) \to (\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ definiert als eine Abbildung $f : \mathcal{X} \to \mathcal{Y}$, für die gilt, dass die Urbilder offener Mengen wieder offen sind (das heißt für jedes $\mathcal{U} \in \mathcal{T}_\mathcal{Y}$ existiert ein $f^{-1}(\mathcal{U}) \in \mathcal{T}_\mathcal{X}$).

Sind die Abbildung $f : \mathcal{X} \to \mathcal{Y}$ und die Abbildung $g : \mathcal{Y} \to \mathcal{Z}$ beide stetig, dann ist auch $g \circ f : \mathcal{X} \to \mathcal{Z}$ stetig.

Definition 2.1.11 (zusammenhängend)**.** Eine Teilmenge $\mathcal{Z}$ der Menge $\mathcal{X}$ heißt *zusammenhängend*, falls $\mathcal{Z}$ nicht die disjunkte Vereinigung zweier (nicht leerer) offener Mengen aus $\mathcal{X}$ ist.

Mit Hilfe der Definition 2.1.11 lässt sich eine weitere Eigenschaft stetiger Abbildungen festlegen. Ist eine Abbildung f stetig und ist die Menge $\mathcal{Z}$ zusammenhängend, dann folgt daraus, dass auch das Bild $f(\mathcal{Z})$ zusammenhängend sein muss.

Definition 2.1.12 (Homöomorphismus)**.** Ein *Homöomorphismus* wird eine stetige Abbildung $f : (\mathcal{X}, \mathcal{T}_\mathcal{X}) \to (\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ genannt, zur der eine stetige Umkehrabbildung $g = f^{-1} : (\mathcal{Y}, \mathcal{T}_\mathcal{Y}) \to (\mathcal{X}, \mathcal{T}_\mathcal{X})$ existiert. Zwei topologische Räume $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ und $(\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ heißen *homöomorph*, wenn ein *Homöomorphismus* zwischen ihnen existiert.

Sonstige Begriffe

Im folgenden Abschnitt werden einige weitere Begriffe definiert, welche in der weiteren Arbeit benötigt werden.

Definition 2.1.13 (metrischer Raum)**.** Ein *metrischer Raum* ist ein Paar (X, d), bestehend aus einer Menge X und einer Distanzfunktion $d : X \times X \to \mathbb{R}_{\geq 0}$, welche die folgenden Bedingungen erfüllt:

1. Ist die Distanz $d(x, y)$ gleich Null, dann folgt daraus, dass x und y gleich sind. Sind umgekehrt x und y gleich, dann muss die Distanz $d(x, y)$ gleich Null sein. $(d(x, y) = 0 \Leftrightarrow x = y)$

2. Die Distanz $d(x, y)$ sei gleich der Distanz $d(y, x)$ $(d(x, y) = d(y, x))$.

3. Die Distanz $d(x, z)$ sei kleiner oder gleich der Summe der Distanzen $d(x, y)$ und $d(y, z)$ $(d(x, z) \leq d(x, y) + d(y, z))$. Diese Bedingung wird als *Dreiecksungleichung* bezeichnet (siehe Abbildung 2.3).

Sind diese drei Bedingungen erfüllt, dann wird (X, d) *metrischer Raum* genannt.

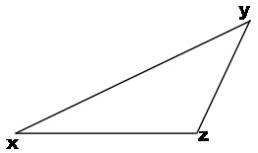


Abbildung 2.3: Grafische Verdeutlichung der Dreiecksungleichung

Offene Mengen im Sinne der Definition topologischer Räume sind bei metrischen Räumen r-Kugeln $(B_r(a) := \{x \in X \mid d(x, a) < r\})$. Topologien auf einem metrischen Raum enthalten somit beliebig viele solcher Kugeln, deren Vereinigungen, Durchschnitte endlich vieler solcher Kugeln und diverse weitere Kombinationen von Vereinigungen und Durchschnitten, der so entstandenen offenen Mengen. Die von den r-Kugeln erzeugte Topologie $\mathcal{T}_d$ auf X heißt die *von der Metrik d erzeugte* Topologie.

Definition 2.1.14 (Relation)**.** Eine *(binäre) Relation* R ist eine Teilmenge des kartesischen Produkts zweier Mengen A und B, also $R \subseteq A \times B$ mit $A \times B := \{(a, b) \mid (a \in A) \text{ und } (b \in B)\}$. Ist $A = B$ und somit $R \subseteq A \times A$, dann nennt man die Relation *homogen*.

2.1.2 Verbinden von Räumen

In diesem Abschnitt sollen zwei Möglichkeiten aufgezeigt werden topologische Räume miteinander zu verbinden. Dabei handelt es sich zum einen um die *topologische Summe* und zum anderen um das *Verkleben*.

Topologische Summe

Um die topologische Summe definieren zu können, muss erstmal die *Summe von Mengen* definiert werden:

Definition 2.1.15 (Summe von Mengen)**.** Seien X und Y Mengen, so erklärt man ihre *disjunkte Vereinigung* oder *Summe* durch einen formalen Trick. Man definiert einfach $X + Y := X \times \{0\} \cup Y \times \{1\}$.

Anschaulich bedeutet dieser Vorgang weiter nichts als das disjunkte Nebeneinanderstellen je eines Exemplars von X und von Y. Das darf man nicht als $X \cup Y$ schreiben, denn X und Y müssen ursprünglich nicht disjunkt sein, zum Beispiel wäre $X \cup X$ wieder nur ein Exemplar von X, während sich aus $X + X$ zwei nebeneinandergestellte Exemplare von X ergeben (siehe Abbildung 2.4).

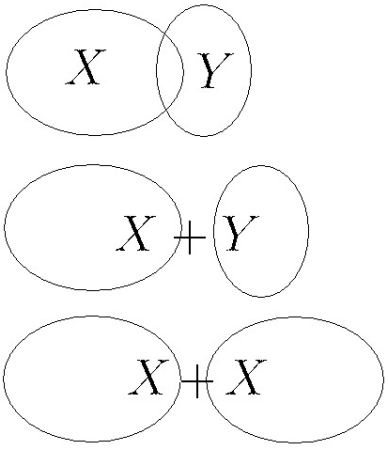


Abbildung 2.4: Veranschaulichung der Summe von Mengen

Auf der Grundlage der Summe von Mengen kann nun auch die *topologische Summe* definiert werden:

Definition 2.1.16 (topologische Summe)**.** Seien $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ und $(\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ topologische Räume. Dann heißt das Paar $(\mathcal{X} + \mathcal{Y}, \mathcal{T}_{\mathcal{X}+\mathcal{Y}})$ die *topologische Summe* der

topologischen Räume $\mathcal{X}$ und $\mathcal{Y}$. Die in $\mathcal{X}+\mathcal{Y}$ offenen Mengen sind Summen von Mengen aus $\mathcal{T}_\mathcal{X}$ und $\mathcal{T}_\mathcal{Y}$.

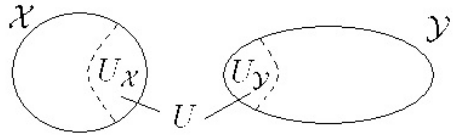


Abbildung 2.5: Veranschaulichung einer in $\mathcal{X}+\mathcal{Y}$ offenen Menge $U \subseteq \mathcal{T}_{\mathcal{X}+\mathcal{Y}}$

Eine in $\mathcal{X}+\mathcal{Y}$ offene Menge $U \subseteq \mathcal{T}_{\mathcal{X}+\mathcal{Y}}$ ergibt sich damit aus der Summe zweier Mengen $U_\mathcal{X}$ und $U_\mathcal{Y}$, wobei $U_\mathcal{X}$ offen in $\mathcal{X}$ und $U_\mathcal{Y}$ offen in $\mathcal{Y}$ ist. Es gilt also $U = U_\mathcal{X} + U_\mathcal{Y}$ mit $U_\mathcal{X} \in \mathcal{T}_\mathcal{X}$ und $U_\mathcal{Y} \in \mathcal{T}_\mathcal{Y}$ (siehe Abbildung 2.5).

Daran erkennt man, dass die topologische Summe die Topologie-Axiome aus Definition 2.1.1 erfüllt, also dass es sich bei dem Paar $(\mathcal{X}+\mathcal{Y}, \mathcal{T}_{\mathcal{X}+\mathcal{Y}})$ in der Tat um einen topologischen Raum handelt:

1. Die Menge $\mathcal{X}$ und die leere Menge $\emptyset$ seien offene Mengen: Da $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ und $(\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ topologische Räume sind, gilt für diese das 1. Axiom und die Mengen $\mathcal{X}$ und $\mathcal{Y}$ sind somit offen. Also ist auch $\mathcal{X}+\mathcal{Y}$ offen. Bei der leeren Menge $\emptyset$ verhält es sich genauso. Die leere Menge $\emptyset$ ist offen in $\mathcal{X}$ und in $\mathcal{Y}$ und somit ist sie auch offen in $\mathcal{X}+\mathcal{Y}$.

2. Die Vereinigung beliebig vieler offener Mengen sei wieder offen: Gegeben seien zwei in $\mathcal{X}+\mathcal{Y}$ offene Mengen U und V. Die Menge W sei die Vereinigung von U und V, also $W = U \cup V$. Da U und V offen in $\mathcal{X}+\mathcal{Y}$ sind, lassen sie sich als $U = U_\mathcal{X} + U_\mathcal{Y}$ und $V = V_\mathcal{X} + V_\mathcal{Y}$ schreiben, wobei $U_\mathcal{X}$ bzw. $V_\mathcal{X}$ dann offen in $\mathcal{X}$ und $U_\mathcal{Y}$ bzw. $V_\mathcal{Y}$ dann offen in $\mathcal{Y}$ sind. Man erhält die Menge W dann auch, wenn man zuerst die Vereinigung ausführt und dann die Summe bildet. Man kann W also auch als $W = W_\mathcal{X} + W_\mathcal{Y}$ mit $W_\mathcal{X} = U_\mathcal{X} \cup V_\mathcal{X}$ und $W_\mathcal{Y} = U_\mathcal{Y} \cup V_\mathcal{Y}$ schreiben. Weil $U_\mathcal{X}$ und $V_\mathcal{X}$ beziehungsweise $U_\mathcal{Y}$ und $V_\mathcal{Y}$ offen in $\mathcal{X}$ beziehungsweise $\mathcal{Y}$ sind, ist auch $W_\mathcal{X}$ offen in $\mathcal{X}$ und $W_\mathcal{Y}$ offen in $\mathcal{Y}$ und somit ist auch W offen in $\mathcal{X}+\mathcal{Y}$. Diese hier beispielhaft für zwei Mengen ausgeführte Beweisidee, lässt sich auch auf die Vereinigung beliebig vieler Mengen übertragen. Man kann alle zu vereinigenden Mengen als Summe zweier Teilmengen schreiben, von denen eine Teilmenge offen in $\mathcal{X}$ und die andere Teilmenge

offen in $\mathcal{Y}$ ist. Dann entstehen, durch die Vereinigung jeweils aller in $\mathcal{X}$ bzw. $\mathcal{Y}$ offenen Mengen, zwei neue Mengen, von denen die eine offen in $\mathcal{X}$ und die andere offen in $\mathcal{Y}$ ist. Die Summe dieser beiden Mengen ist dann offen in $\mathcal{X} + \mathcal{Y}$.

3. Der Durchschnitt zweier offener Mengen sei wieder offen: Dieser Fall verhält sich analog zum 2. Axiom. Gegeben seien wieder zwei in $\mathcal{X} + \mathcal{Y}$ offene Mengen U und V, welche sich als $U = U_\mathcal{X} + U_\mathcal{Y}$ und $V = V_\mathcal{X} + V_\mathcal{Y}$ schreiben lassen. Dabei gilt auch wieder, dass $U_\mathcal{X}$ und $V_\mathcal{X}$ offen in $\mathcal{X}$ und $U_\mathcal{Y}$ und $V_\mathcal{Y}$ offen in $\mathcal{Y}$ sind. Den Durchschnitt $W = U \cap V$ bekommt man dann ebenfalls, indem man W als $W = W_\mathcal{X} + W_\mathcal{Y}$ schreibt, wobei $W_\mathcal{X} = U_\mathcal{X} \cap V_\mathcal{X}$ und $W_\mathcal{Y} = U_\mathcal{Y} \cap V_\mathcal{Y}$ ist. $W_\mathcal{X}$ ist dann offen in $\mathcal{X}$ und $W_\mathcal{Y}$ ist offen in $\mathcal{Y}$ und somit ist $W = W_\mathcal{X} + W_\mathcal{Y}$ offen in $\mathcal{X} + \mathcal{Y}$.

Verkleben

Damit man verstehen kann wie *Verkleben* funktioniert, müssen zuerst einige Begriffe wie *Äquivalenzrelation*, *Äquivalenzklasse*, *Quotientenmenge* und *kanonische Projektion* erklärt werden. Zunächst also die Definition der *Äquivalenzrelation*:

Definition 2.1.17 (Äquivalenzrelation)**.** Sei X eine Menge. Eine Relation $R \subseteq X \times X$ heißt *Äquivalenzrelation* auf X, wenn sie die folgenden drei Eigenschaften aufweist:

1. Reflexivität: Jedes Objekt ist zu sich selbst äquivalent. ($x \sim x$)
2. Symmetrie: Wenn x zu y äquivalent ist, dann ist auch y äquivalent zu x (und umgekehrt). ($x \sim y \Leftrightarrow y \sim x$)
3. Transitivität: Wenn x zu y und y zu z äquivalent ist, dann ist x auch äquivalent zu z. ($x \sim y$ und $y \sim z \Rightarrow x \sim z$)

Für eine Äquivalenzrelation wird üblicherweise das Symbol $\sim$ verwendet.

Im nächsten Schritt kann jetzt der Begriff der *Äquivalenzklasse* definiert werden:

Definition 2.1.18 (Äquivalenzklasse)**.** Sei X eine Menge und $\sim$ eine Äquivalenzrelation auf X. Dann heißt für jedes $x \in X$ die Menge $[x] := \{y \in X \mid y \sim x\}$ *Äquivalenzklasse* von x bezüglich $\sim$, das heißt sie enthält jedes $y \in X$, welches äquivalent zu x ist. x wird dann auch als *Repräsentant* für $[x]$ bezeichnet.

Darauf aufbauend folgt nun die Definition der *Quotientenmenge*:

Definition 2.1.19 (Quotientenmenge)**.** Sei X eine Menge und $\sim$ eine Äquivalenzrelation auf X. Die Menge aller Äquivalenzklassen $X/\sim := \{[x] \mid x \in X\}$ bezüglich $\sim$ heißt dann die *Quotientenmenge* von X nach $\sim$ oder kurz „X modulo $\sim$“.

Jetzt kann auch geklärt werden, worum es sich bei einer *kanonischen Projektion* handelt:

Definition 2.1.20 (kanonische Projektion)**.** Sei X eine Menge und $\sim$ eine Äquivalenzrelation auf X. Die *kanonische Projektion* nennt man die Abbildung $\pi : X \to X/\sim$ mit $x \mapsto [x]$ der Menge X auf die Quotientenmenge $X/\sim$, wobei jedes $x \in X$ auf seine Äquivalenzklasse $[x] \in X/\sim$ abgebildet wird.

Weiterführend kann hiermit der Begriff des *Quotientenraumes* definiert werden:

Definition 2.1.21 (Quotientenraum)**.** Sei $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ ein topologischer Raum, $\sim$ eine Äquivalenzrelation auf $\mathcal{X}$ und π die kanonische Projektion. Dann ist $(\mathcal{X}/\sim, \mathcal{T}_{\mathcal{X}/\sim}) := (\mathcal{X}, \mathcal{T}_\mathcal{X})/\sim$ ebenfalls ein topologischer Raum, wenn man die Topologie $\mathcal{T}_{\mathcal{X}/\sim}$ derart definiert, dass eine Teilmenge $U \subseteq \mathcal{X}/\sim$ der Quotientenmenge $\mathcal{X}/\sim$ offen in $\mathcal{X}/\sim$ ist, wenn ihr Urbild $\pi^{-1}(U)$ offen in $\mathcal{X}$ ist. Das heißt die kanonische Projektion $\pi : \mathcal{X} \to \mathcal{X}/\sim$ ist stetig. Der topologische Raum $(\mathcal{X}, \mathcal{T}_\mathcal{X})/\sim$ heißt *Quotientenraum* und die Topologie $\mathcal{T}_{\mathcal{X}/\sim}$ dazu heißt *Quotiententopologie*.

Ein anschauliches und nützliches Beispiel für die in den Definitionen 2.1.17 bis 2.1.21 definierten Begriffe ist das Zusammenschlagen eines Teilraumes zu einem Punkt. Gegeben sei hierfür ein topologischer Raum $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ und eine nichtleere Teilmenge $A \subset \mathcal{X}$ von $\mathcal{X}$. Dazu wird eine Äquivalenzrelation $\sim_A$ folgendermaßen definiert: $x \sim_A y :\Leftrightarrow x = y$ oder $x, y \in A$. Das heißt x ist

äquivalent zu y, wenn x gleich y ist oder wenn x und y beide in A enthalten sind. Man könnte sich an dieser Stelle natürlich fragen, ob es sich bei $\sim_A$ tatsächlich um eine Äquivalenzrelation handelt, aber das lässt sich recht einfach anhand der Eigenschaften einer Äquivalenzrelation aus Definition 2.1.17 überprüfen:

1. Reflexivität: Jedes Objekt soll zu sich selbst äquivalent sein, also $x \sim_A x$ soll gelten. Das ist erfüllt, da immer $x = x$ gilt, unabhängig davon, ob x in A enthalten ist oder nicht.

2. Symmetrie: Wenn x zu y äquivalent ist, dann soll auch y äquivalent zu x sein (und umgekehrt), also $x \sim_A y \Leftrightarrow y \sim_A x$ soll gelten. Wenn $x \sim_A y$ gilt, dann ist entweder $x = y$ oder x und y sind beide in A enthalten. Wenn $x = y$ gilt, dann gilt auch $y = x$ und wenn x und y beide in A enthalten sind, dann sind auch y und x beide in A enthalten, da es bei der Beziehung „ist enthalten in“ nicht auf die Schreibreihenfolge der Elemente ankommt. Die Eigenschaft der Symmetrie ist also ebenfalls erfüllt.

3. Transitivität: Wenn x zu y äquivalent und y zu z äquivalent ist, dann soll x auch äquivalent zu z sein, also $x \sim_A y$ und $y \sim_A z \Rightarrow x \sim_A z$ soll gelten. Hier gibt es mehrere denkbare Möglichkeiten. Die einfachste Möglichkeit ist, wenn $x = y$ und $y = z$ gilt, dann gilt natürlich auch $x = z$ und die Transitivität ist erfüllt. Die zweite Möglichkeit wäre, dass $x, y \in A$ und $y, z \in A$ gilt. In diesem Fall sind alle drei Elemente x, y und z in A enthalten und damit gilt auch $x, z \in A$. Auch für diese Möglichkeit ist die Transitivität erfüllt. Als dritte Möglichkeit ist noch ein „Mischfall“ denkbar, also wenn $x = y$ und $y, z \in A$ oder auch $x, y \in A$ und $y = z$ gilt. In diesen Fällen gilt jeweils auch $x, z \in A$, also $x \sim_A z$. Die Transitivität ist also für alle drei Möglichkeiten erfüllt.

Nachdem geklärt ist, dass $\sim_A$ alle drei Eigenschaften, Reflexivität, Symmetrie und Transitivität aufweist, ist klar, dass es sich bei $\sim_A$ wirklich um eine Äquivalenzrelation auf $\mathcal{X}$ handelt.

Schaut man sich als Nächstes an, welche Äquivalenzklassen sich bezüglich $\sim_A$ bilden, erkennt man, dass die Menge A eine Äquivalenzklasse ist, da alle in ihr enthaltenen Elemente zueinander äquivalent sind. Außerdem sind die nicht

in A enthaltenen einelementigen Teilmengen von $\mathcal{X}$ auch Äquivalenzklassen. In der Quotientenmenge $\mathcal{X}/A := \mathcal{X}/\sim_A$ ist daher A ein Punkt, während das Komplement $\mathcal{X} \setminus A$ praktisch unverändert bleibt. Mit Hilfe der kanonischen Projektion $\pi_A : \mathcal{X} \to \mathcal{X}/A$ mit $x \mapsto [x]$ hat man somit eine stetige Abbildung zur Hand, welche einen topologischen Raum $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ in einen anderen topologischen Raum (den Quotientenraum $(\mathcal{X}/A, \mathcal{T}_{\mathcal{X}/A})$) abbildet und dabei einen Teilraum $A \subset \mathcal{X}$ auf einen Punkt zusammenschlägt, während der Rest unverändert bleibt. In Abbildung 2.6 wird dieses Beispiel grafisch verdeutlicht.

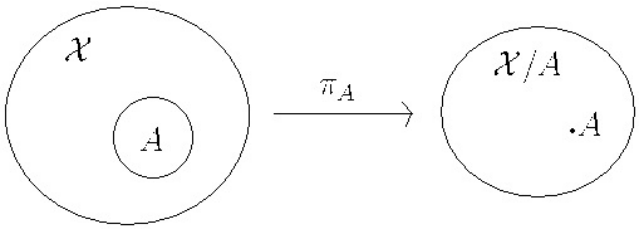


Abbildung 2.6: Veranschaulichung des Beispiels zu den Definitionen 2.1.17 bis 2.1.21

Jetzt ist das nötige Handwerkszeug vorhanden, um endlich das *Verkleben* definieren zu können:

Definition 2.1.22 (Verkleben)**.** Seien $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ und $(\mathcal{Y}, \mathcal{T}_\mathcal{Y})$ topologische Räume, $\mathcal{X}_0 \subseteq \mathcal{X}$ ein Teilraum von $\mathcal{X}$ und $\varphi : \mathcal{X}_0 \to \mathcal{Y}$ eine stetige Abbildung. Dann bezeichnet man mit $\mathcal{Y} \cup_\varphi \mathcal{X}$ den Quotientenraum $(\mathcal{X} + \mathcal{Y})/\sim_\varphi$. Die Äquivalenzrelation $\sim_\varphi$ ist hierbei folgendermaßen definiert: $x \sim_\varphi y :\Leftrightarrow (x = y)$ oder $(x \in \mathcal{X}_0$ und $y = \varphi(x))$ oder umgekehrt $(y \in \mathcal{X}_0$ und $x = \varphi(y))$ oder $(x, y \in \mathcal{X}_0$ und $\varphi(x) = \varphi(y))$. Man sagt auch, $\mathcal{Y} \cup_\varphi \mathcal{X}$ entstehe durch *Verkleben* bzw. *Anheften* von $\mathcal{X}$ an $\mathcal{Y}$ mittels der *Verklebungs-* bzw. *Anheftungsabbildung* φ.

Ähnlich wie bei $\sim_A$ kann man an Hand der Eigenschaften einer Äquivalenzrelation aus Definition 2.1.17 leicht überprüfen, dass $\sim_\varphi$ dabei wirklich eine Äquivalenzrelation ist:

1. Reflexivität: Jedes Objekt soll zu sich selbst äquivalent sein, es soll also $x \sim_\varphi x$ gelten. Das ist erfüllt, da immer $x = x$ gilt, unabhängig davon, in welcher Menge x liegt.

2. Symmetrie: Wenn x zu y äquivalent ist, dann soll auch y äquivalent zu x sein (und umgekehrt). Es soll also $x \sim_\varphi y \Leftrightarrow y \sim_\varphi x$ gelten. Wenn $x \sim_\varphi y$ gilt, dann ist entweder $x = y$ oder ($x \in \mathcal{X}_0$ und $y = \varphi(x)$) oder ($y \in \mathcal{X}_0$ und $x = \varphi(y)$) oder ($x, y \in \mathcal{X}_0$ und $\varphi(x) = \varphi(y)$). Wenn $x = y$ gilt, dann gilt auch $y = x$ und die Symmetrieeigenschaft ist erfüllt. Wenn $x \in \mathcal{X}_0$ und $y = \varphi(x)$ gilt, dann ist auch $y \sim_\varphi x$, da die Schreibreihenfolge der beiden Bedingungen egal ist und man auch schreiben könnte $y = \varphi(x)$ und $x \in \mathcal{X}_0$. Das Gleiche gilt für $y \in \mathcal{X}_0$ und $x = \varphi(y)$. Die Symmetrieeigenschaft ist also auch für diese beiden Fälle erfüllt. Wenn $x, y \in \mathcal{X}_0$ und $\varphi(x) = \varphi(y)$ gilt, dann ist die Symmetrie ebenfalls gegeben, weil dann auch $y, x \in \mathcal{X}_0$ und $\varphi(y) = \varphi(x)$ ist. Die Symmetrieeigenschaft ist also in allen vier Fällen erfüllt.

3. Transitivität: Wenn x zu y äquivalent und y zu z äquivalent ist, dann soll x auch äquivalent zu z sein. Es soll also $x \sim_\varphi y$ und $y \sim_\varphi z \Rightarrow x \sim_\varphi z$ gelten. Dies überprüft man am besten, wenn man sich alle Möglichkeiten, für die $x \sim_\varphi y$ gilt, nacheinander vornimmt und sich dann überlegt, unter welchen Bedingungen dann $y \sim_\varphi z$ sein kann und was sich daraus jeweils für x und z ergibt. Der einfachste Fall ist, wenn $x = y$ und $y = z$ gilt, dann ist natürlich auch $x = z$ und die Transitivität ist gegeben. Die nächste Variante wäre, dass $x = y$ ist und für $y \sim_\varphi z$ eine beliebige der anderen Bedingungen gilt. Dann gilt diese andere Bedingung auch für x und z und somit ist die Transitivitätseigenschaft erfüllt. Ebenso verhält es sich, wenn für $x \sim_\varphi y$ eine der Bedingungen gilt und $y = z$ ist. Nun sei $x \in \mathcal{X}_0$ und $y = \varphi(x)$. Da y in diesem Fall ein Element von $\varphi(\mathcal{X}_0)$ ist, kann $y \sim_\varphi z$ nur sein, wenn $z \in \mathcal{X}_0$ und $y = \varphi(z)$ gilt. Daraus ergibt sich dann $x, z \in \mathcal{X}_0$ und $\varphi(x) = \varphi(z)(= y)$, das heißt $x \sim_\varphi z$. Im nächsten Fall sei $y \in \mathcal{X}_0$ und $x = \varphi(y)$. Da $y \in \mathcal{X}_0$ ist, gibt es dann zwei Möglichkeiten, wie $y \sim_\varphi z$ gelten kann. Erstens kann $y \in \mathcal{X}_0$ und $z = \varphi(y)$ sein. Daraus ergibt sich, dass $x = z$ sein muss und damit ist auch $x \sim_\varphi z$. Zweitens kann $y, z \in \mathcal{X}_0$ und $\varphi(y) = \varphi(z)(= x)$ sein. Dann gilt auch $z \in \mathcal{X}_0$ und $x = \varphi(z)$ und somit ist $x \sim_\varphi z$. Eine Variante für $x \sim_\varphi y$ bleibt nun noch übrig, nämlich dass $x, y \in \mathcal{X}_0$ und $\varphi(x) = \varphi(y)$ gilt. Dann ist $y \sim_\varphi z$ nur möglich, wenn $y \in \mathcal{X}_0$ und $z = \varphi(y)(= \varphi(x))$ ist. Dann gilt ebenfalls $x \in \mathcal{X}_0$ und $z = \varphi(x)$, also $x \sim_\varphi z$. Man erkennt also, dass sich für alle

möglichen Varianten $x \sim_\varphi z$ ergibt und somit die Transitivität gegeben ist.

Anschaulich kann man sich das Verkleben vorstellen als würden die Räume $\mathcal{X}$ und $\mathcal{Y}$ an einer bestimmten Stelle „zusammengeklebt“ oder „geheftet“, daher auch die Ausdrücke Verkleben beziehungsweise Anheften. Die „Klebestellen“ werden durch die Verklebungsabbildung φ bestimmt. Es handelt sich dabei um $\mathcal{X}_0$ aus $\mathcal{X}$ und $\varphi(\mathcal{X}_0)$ aus $\mathcal{Y}$. Das Verkleben wird in Abbildung 2.7 noch mal grafisch veranschaulicht.

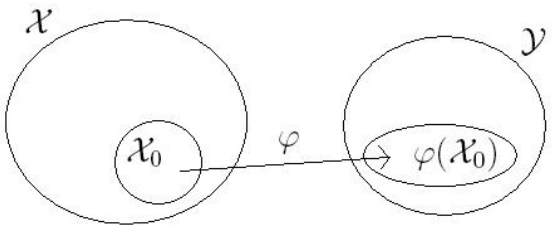


Abbildung 2.7: Veranschaulichung des Verklebens

2.1.3 CW-Komplexe

Mit Hilfe des Verklebens kann bald der Begriff *CW-Komplex* definiert werden. Dafür werden allerdings zuerst noch einige weitere Begriffe benötigt. Begonnen werden soll mit der *Zerlegung* einer Menge:

Definition 2.1.23 (Zerlegung)**.** Unter einer *Zerlegung* einer Menge X versteht man eine Menge paarweise disjunkter Teilmengen von X, deren Vereinigung ganz X ist. Jedes Element von X liegt also in genau einer dieser Mengen.

Weiter geht es mit der *n-Zelle* und der *Zellzerlegung*:

Definition 2.1.24 (n-Zelle)**.** Ein topologischer Raum heißt *n-Zelle*, wenn er zu einer offenen Vollkugel D_n° im $\mathbb{R}^n$ homöomorph ist.

Einige Beispiele für n-Zellen werden in den Abbildungen 2.8 bis 2.10 dargestellt. In Abbildung 2.11 sieht man ein Beispiel für eine Wandfassade, welche keine Zelle ist, da sie „Löcher“ (zum Beispiel Fenster) hat und somit nicht homöomorph zur offenen Vollkugel im $\mathbb{R}^2$ ist. Möchte man die Wand dennoch mit Hilfe einer 2-Zelle darstellen, dann muss man, wie in Abbildung 2.12 zu sehen, Verbindungslinien von den „Löchern“ nach außen einfügen.

Abbildung 2.8: 0-Zelle Abbildung 2.9: 1-Zellen

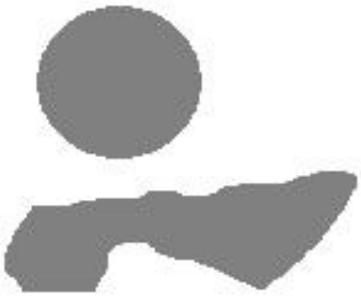


Abbildung 2.10: 2-Zellen

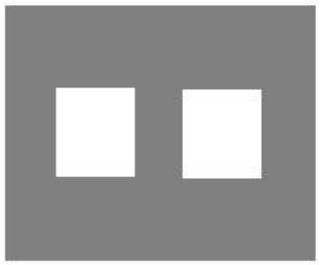


Abbildung 2.11: keine Zelle

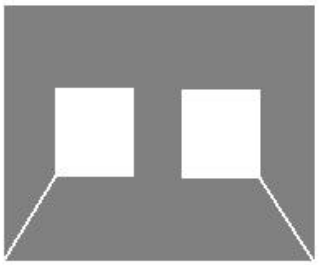


Abbildung 2.12: 2-Zelle

Definition 2.1.25 (Zellenzerlegung)**.** Eine *Zellenzerlegung* $\mathcal{E}$ eines topologischen Raumes $(\mathcal{X}, \mathcal{T}_\mathcal{X})$ ist, wie der Name sagt, eine Zerlegung von $\mathcal{X}$ in Teilräume, welche Zellen sind. Eine Zellenzerlegung $\mathcal{E}$ wird *endlich* genannt, wenn sie aus endlich vielen Zellen besteht.

Außerdem wird zur Definition des Begriffes *CW-Komplex* auch noch der

Begriff des *Hausdorff-Raumes* benötigt:

Definition 2.1.26 (Hausdorff-Raum)**.** Ein topologischer Raum heißt *Hausdorff-Raum*, wenn es für beliebige Punkte $x \neq y$ des Raums disjunkte offene Umgebungen U_x und U_y gibt, so dass $x \in U_x$ und $y \in U_y$.

Der $\mathbb{R}^n$ ist zum Beispiel ein Hausdorff-Raum, da man für jeden Punkt x eine offene n-Kugel $\sum_{i=1}^{n} x_i^2 < \varepsilon$ finden kann, die sich nicht mit einer benachbarten offenen n-Kugel schneidet. Des Weiteren sind alle metrischen Räume Hausdorff-Räume, denn ist d eine Metrik und $d(x, y) = \varepsilon > 0$, dann sind zum Beispiel $U_x := \{z \mid d(x, z) < \frac{\varepsilon}{2}\}$ und $U_y := \{z \mid d(y, z) < \frac{\varepsilon}{2}\}$ disjunkte Umgebungen.

Jetzt kann endlich auch der Begriff *CW-Komplex* definiert werden. Hierbei sei $D_n := \{x \mid \sum_{i=1}^{n} x_i^2 \leq 1\}$.

Definition 2.1.27 (CW-Komplex)**.** Ein Paar $(\mathcal{X}, \mathcal{E})$, bestehend aus einem Hausdorff-Raum $\mathcal{X}$ und einer (endlichen) Zellenzerlegung $\mathcal{E}$ von $\mathcal{X}$, heißt *(endlicher) CW-Komplex* (oder auch *Zellkomplex*), wenn es zu jeder n-Zelle $e \in \mathcal{E}$ eine stetige Abbildung $\varphi_e : D_n \to \mathcal{X}$ gibt, welche die offene Vollkugel D_n° homöomorph auf die Zelle e und die $(n-1)$-Sphäre $S^{n-1} := \partial D_n$ in die Vereinigung der höchstens $(n-1)$-dimensionalen Zellen abbildet. Die Abbildungen φ_e werden *charakteristische Abbildungen* genannt.

Hier wurde bewusst nur der endliche CW-Komplex definiert, da man bei der Anwendung auf CityGML beziehungsweise allgemein auf 3D-Stadtmodelle immer davon ausgehen kann, dass es sich um endlich viele Zellen handelt. Möchte man *unendlich* viele Zellen zulassen können, dann müssen noch zwei weitere Bedingungen erfüllt sein, die *Hüllenendlichkeit* und die *schwache Topologie*:

- *Hüllenendlichkeit*: Die abgeschlossene Hülle $\bar{e}$ jeder Zelle $e \in \mathcal{E}$ trifft nur endlich viele andere Zellen.

- *Schwache Topologie*: $A \subset \mathcal{X}$ ist genau dann abgeschlossen, wenn jedes $A \cap \bar{e}$ abgeschlossen ist.

Von diesen beiden Bedingungen rührt auch die Bezeichnung „CW“-Komplex her. Es steht nämlich „C“ für „closure finite“ (hüllenendlich) und „W“ für „weak topology“ (schwache Topologie).

Anschaulich kann man sich den Aufbau eines CW-Komplexes $\mathcal{X}$ so vorstellen, dass man mit einer Menge X_0 beginnt, welche alle 0-Zellen, also alle Punkte, enthält. Diese Menge X_0 wird auch als *0-Skelett* bezeichnet. Dieses 0-Skelett wird nun sukzessive durch Verkleben mit den 1-Zellen verbunden. Als Verklebungsabbildungen werden die charakteristischen Abbildungen verwendet. Der so entstandene Raum X_1 heißt *1-Skelett*. In dieses 1-Skelett werden dann wiederum die 2-Zellen „geklebt“ und man erhält das *2-Skelett* X_2. Dieser Vorgang, also das Verkleben der $(i+1)$-Zellen mit dem i-Skelett, wird solange wiederholt bis das *n-Skelett* $X_n = \mathcal{X}$ erreicht wird. Die Zahl n ist beim endlichen (oder mindestens endlich dimensionalen) CW-Komplexen die Dimension des CW-Komplexes, falls $X_m = X_n$ für $m \geq n$ und $X_m \neq X_n$ für $m < n$ gilt. Anschaulich bedeutet das, dass zum n-Skelett beim Erhöhen der Dimension nichts mehr hinzu kommt, während es sich aber noch von den m-Skeletten niedrigerer Dimensionen unterscheidet, das heißt n-Zellen sind die Zellen der höchsten Dimension, welche im CW-Komplex vorkommen.

Ein Beispiel für 1-dimensionale CW-Komplexe sind Graphen. Ein Beispielgraph ist in Abbildung 2.13 abgebildet.

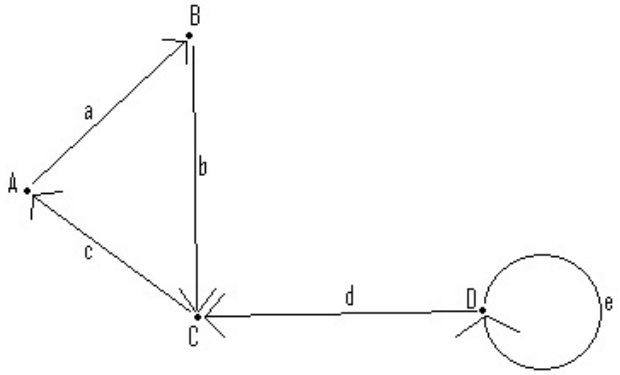


Abbildung 2.13: Beispielgraph

2.1.4 Kettenkomplexe

Graphen sind auch ein guter Anfang um den Übergang von CW-Komplexen zu *Kettenkomplexen* zu veranschaulichen. Deshalb soll der Beispielgraph aus Abbildung 2.13 etwas genauer betrachtet werden.

Zuerst wird hierfür jedoch der Begriff der *direkten Summe* von Vektorräumen gebraucht:

Definition 2.1.28 (direkte Summe)**.** Sei V ein Vektorraum. Eine Summe $\sum_{i=1}^{n} U_i$ von Untervektorräumen $U_1 \dots U_n \subset V$ des Vektorraumes V heißt *direkt*, wenn jedes Element $x \in \sum_{i=1}^{n} U_i$ eine eindeutige Darstellung $x = x_1 + \dots + x_n$ besitzt, bei der, für alle $i = 1 \dots n$, $x_i \in U_i$ gilt. Die *direkte Summe* schreibt man $\bigoplus U_i$ oder $U_1 \oplus \dots \oplus U_n$.

Jetzt sind die Voraussetzungen geschaffen, welche nötig sind, um den Graphen Γ aus Abbildung 2.13 algebraisch zu betrachten. Hierfür sei C_0 die Menge aller Knoten (also 0-Zellen) des Graphen und C_1 sei die Menge aller Kanten (also 1-Zellen) des Graphen. C_0 und C_1 lassen sich dann als direkte Summe von $\mathbb{R}$-Vektorräumen schreiben. Es ist also $C_0 = \mathbb{R}A \oplus \mathbb{R}B \oplus \mathbb{R}C \oplus \mathbb{R}D$ und $C_1 = \mathbb{R}a \oplus \mathbb{R}b \oplus \mathbb{R}c \oplus \mathbb{R}d \oplus \mathbb{R}e$, da allgemein der $\mathbb{R}$-Vektorraum $\mathbb{R}B$ zu einer Menge B als direkte Summe der $\mathbb{R}$-Vektorräume $\mathbb{R}b$ der Elemente $b \in B$ von B geschrieben werden kann, also $\mathbb{R}B = \bigoplus_{b \in B} \mathbb{R}b$ gilt. Abgekürzt kann man dann also auch $C_0 = \mathbb{R}\{A, B, C, D\}$ und $C_1 = \mathbb{R}\{a, b, c, d, e\}$ schreiben.

Was nun noch fehlt ist eine Verknüpfung von C_0 und C_1. Diese ist mit Hilfe des Randoperators $\partial_1 : C_1 \to C_0$ möglich, welcher C_1 auf seinen Rand C_0 abbildet. Für das Beispiel könnte der Randoperator ∂_1 zum Beispiel folgendermaßen definiert werden: $\partial_1 : C_1 \to C_0$ mit $a \mapsto B - A$, $b \mapsto C - B$, $c \mapsto A - C$, $d \mapsto C - D$ und $e \mapsto D - D = 0D$. Die Vorzeichen sind dabei so gewählt, dass sie dem entsprechen, was man erhält, wenn man sich die Großbuchstaben als die Ortsvektoren der Knoten vorstellt. Man könnte das aber auch umgekehrt machen, wichtig ist nur, dass die Richtung der Kanten erkennbar ist.

Man kann den Graphen also als Folge $\mathcal{C}$ von Randoperatoren ∂_i darstellen:

$$\mathcal{C} : \dots \longrightarrow 0 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0 \longrightarrow \dots$$

Hier kann man die (triviale) Beobachtung machen, dass sich als Rand vom Rand Null ergibt. Es gilt also $\partial^2 = 0$ beziehungsweise $\partial_0 \circ \partial_1 = 0$. Außerdem kann man beobachten, dass das Bild von ∂_1 eine Teilmenge vom Kern von ∂_0 ist ($\text{bild}(\partial_1) \subseteq \ker(\partial_0)$). Eine derartige Folge $\mathcal{C}$ wird als *Kettenkomplex* bezeichnet.

Diese Darstellungsform als Kettenkomplex $\mathcal{C}$ lässt sich auf beliebige n-dimensionale CW-Komplexe übertragen. Man erhält dann einen Kettenkom-

plex $\mathcal{C}$ der Form

$$\mathcal{C} : \ldots \to 0 \xrightarrow{\partial_{n+1}} C_n \xrightarrow{\partial_n} C_{n-1} \xrightarrow{\partial_{n-1}} C_{n-2} \xrightarrow{\partial_{n-2}} \ldots \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0 \to \ldots,$$

wobei es sich bei den C_i wieder jeweils um die direkte Summe der i-Zellen handelt und bei ∂_i um die dazugehörigen Randoperatoren. Zusätzlich muss, wie beim Graphen, wieder gelten, dass der Rand vom Rand Null ist. Es muss also $\partial^2 = 0$ beziehungsweise $\partial_i \circ \partial_{i+1} = 0$ sein.

Nun könnte man sich die Frage stellen: Wie erhält man diese Randoperatoren ∂_i für einen gegebenen CW-Komplex? Um diese Frage zu beantworten betrachtet man zunächst wieder den Beispielgraphen Γ aus Abbildung 2.13. Der Randoperator ∂_1 kann folgendermaßen als Matrix dargestellt werden:

∂_1	a	b	c	d	e
A	-1	0	1	0	0
B	1	-1	0	0	0
C	0	1	-1	1	0
D	0	0	0	-1	0

Diese Matrix sieht dem *topologischen Datentyp* recht ähnlich, durch welchen sich die Relation $D_\Gamma \subseteq E \times V$ zwischen Knoten und Kanten eines Graphen Γ in Matrizenform darstellen lässt (siehe Abbildung 2.14).

D_Γ	a	b	c	d	e
A	x		x		
B	x	x			
C		x	x	x	
D				x	x

Abbildung 2.14: topologischer Datentyp des Graphen aus Abbildung 2.13

Dort wo im topologischen Datentyp ein x steht, findet man in der Matrix des Randoperators ∂_1 eine Zahl $x \neq 0$ (im Beispiel immer 1 und -1) und dort wo im topologischen Datentyp kein Eintrag ist, findet man in der Matrix des Randoperators ∂_1 eine Null. Die einzige Stelle, welche von dieser Regel abweicht, ist der Eintrag $(e : D)$, welcher im topologischen Datentyp ein x und in der Matrix des Randoperators ∂_1 jedoch eine Null ist. Dieser Unterschied kommt durch die Beachtung der Richtung der Kanten im Randoperator ∂_1, wobei im

topologischen Datentyp nur die Beziehung Kante „hängt an“ Knoten modelliert wird. Durch die Vorschrift $e \mapsto D - D = 0D$ kommt der Nulleintrag in der Matrix des Randoperators ∂_1 zustande, während im topologischen Datentyp nur erfasst wird, dass die Kante e am Knoten D „hängt“, wodurch sich ein Eintrag x ergibt. Das bedeutet also, dass man in der Matrix des Randoperators ∂_1 schleifenförmige Kanten nicht erkennen kann. Diese Tatsache kann man aber recht leicht umgehen, indem man die Matrix des Randoperators ∂_1 folgendermaßen schreibt:

∂_1	a	b	c	d	e
A	-1		1		
B	1	-1			
C		1	-1	1	
D				-1	0

Jetzt gibt es nur noch überall dort einen Eintrag, wo es auch im topologischen Datentyp einen Eintrag gibt, das heißt hängen eine Kante und ein Knoten nicht zusammen, dann gibt es gar keinen Eintrag, statt dem Eintrag Null. Ein Null-Eintrag lässt somit eindeutig auf eine schleifenförmige Kante schließen. Jetzt muss man nur noch festlegen, wie bei der Matrixmultiplikation mit dem Nicht-Eintrag umgegangen werden muss, damit weiter $\partial^2 = 0$ gilt. Das lässt allerdings recht einfach handhaben, indem man den Nicht-Eintrag als neutrales Element ähnlich der Null behandelt, das heißt man legt $nichts + a = a$ und $nichts \cdot a = nichts$ fest.

Kettenkomplexe, deren Randoperatoren ∂_i auf diese Weise in Matrizenform dargestellt werden können, werden als *relationale Kettenkomplexe* bezeichnet.

Aus den Matrizen der Randoperatoren ∂_i lassen sich auch die Bettizahlen b_0 und b_1 für den Graphen Γ berechnen. Die Anzahl der Zusammenhangskomponenten ergibt sich als $b_0 = \dim(\ker(\partial_0)) - \dim(\text{bild}(\partial_1))$ und die Anzahl der minimalen Rundwege b_1 ergibt sich als $b_1 = \dim(\ker(\partial_1)) - \dim(\text{bild}(\partial_2))$. Dieses Thema soll hier allerdings nicht weiter vertieft werden.

Nun zurück zur Frage, wie man die Randoperatoren ∂_i für einen gegebenen CW-Komplex bestimmen kann. Mit Hilfe der Darstellung der Randoperatoren ∂_i in Matrizenform lässt sich erkennen, dass man die Randoperatoren ∂_i durch eine Bestimmung der Matrixeinträge erhält. Das Problem kann also auf eine Bestimmung der Matrixeinträge reduziert werden.

Um die Matrixeinträge zu bestimmen, kann das bereits als Beispiel im Abschnitt 2.1.2 eingeführte Zusammenschlagen eines Teilraumes zu einem Punkt verwendet werden. Dazu wird der komplette CW-Komplex außer einer n-Zelle auf eine 0-Zelle, also einen Punkt, kontrahiert. Was hierbei passiert, lässt sich wieder gut am Beispiel des Graphen verdeutlichen. Nimmt man den Beispielgraphen Γ aus Abbildung 2.13 und kontrahiert alles, hier mal mit $\mathcal{B}$ bezeichnet, außer einer 1-Zelle, nämlich der Kante e, auf eine 0-Zelle, dann erhält man einen Punkt $\mathcal{B}$ mit einer Kreislinie, welche der Kante e entspricht (siehe Abbildung 2.15).

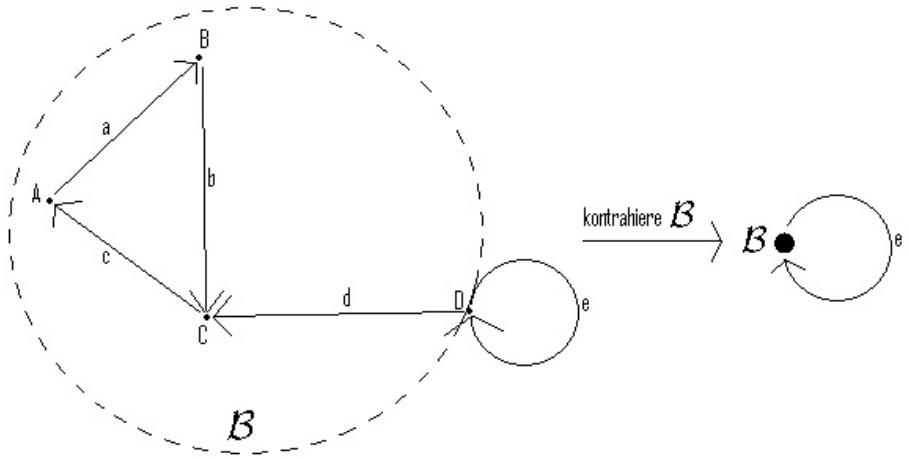


Abbildung 2.15: kontrahiere alles außer Kante e

Allgemein ergibt sich bei einem Graphen Γ immer ein Punkt mit einer Kreislinie, wenn man den kompletten Graphen Γ außer einer beliebigen Kante x auf einen Punkt kontrahiert. Die erhaltene Kreislinie kann topologisch als 1-Sphäre S^1 interpretiert werden.

Geht man jetzt einen Schritt weiter und kontrahiert bei einem CW-Komplex alles außer einer 2-Zelle, dann erhält man einen Punkt mit einer Kugel, also eine 2-Sphäre S^2 und so weiter. Man erhält also beim Kontrahieren eines kompletten CW-Komplexes außer einer n-Zelle immer einen Punkt mit einer n-Sphäre S^n.

Außerdem benötigt man zur Bestimmung der Randoperatoren ∂_i noch den *Grad* $\deg(f)$ von stetigen Abbildungen $f : S^n \to S^n$ zwischen n-Sphären. Dabei seien die n-Sphären S^n jeweils mit einer Orientierung ausgestattet. Für die allgemeine Definition des *Grades* $\deg(f)$ wird allerdings der Begriff *Homologie* benötigt, auf dessen Einführung hier der Einfachheit halber verzichtet wird.

Jedoch lässt sich der *Grad* $\deg(f)$ in vielen Fällen folgendermaßen bestimmen:

- Ist die Abbildung f nicht surjektiv, dann ist $\deg(f) := 0$ definiert.
- Sei f surjektiv. Stimmen die Orientierungen von S^n und $f(S^n)$ überein, dann ist $\deg(f) \geq 0$, andernfalls ist $\deg(f) \leq 0$.
- Hat f für alle $y \in f(S^n)$ stets d Urbilder, dann ist $\deg(f) = \pm d$.
- Sei $f : S^n \to S^n$ lokal ein Homöomorphismus, das heißt es gibt für jedes $x \in S^n$ und für jedes $y \in f(S^n)$ offene Umgebungen für die f ein Homöomorphismus ist und für ein $y \in f(S^n)$ ist $f^{-1}(y)$ endlich. Die Anzahl der Urbilder sei dabei d. Dann lässt sich f in d einzelne stetige Abbildungen f_i aufteilen, wobei $\deg(f) = \sum_{i=1}^{d} \deg(f_i)$ gilt.

Einige Beispiele für den Grad einer Abbildung f kann man sich für die Abbildung zwischen 1-Sphären, also Kreislinien, überlegen. Diese sind noch recht gut anschaulich vorzustellen, was bei höheren Dimensionen immer schwieriger wird.

Nun also drei Beispiele für Abbildungen f zwischen 1-Sphären:

- $f = const.$: Der Grad $\deg(f)$ ist Null, weil f nicht surjektiv ist.
- f „wickelt auf“: Die Abbildung f durchläuft hierbei die Kreislinie n mal in die gleiche Richtung. Daraus ergibt sich der Grad $\deg(f) = \pm n$, je nachdem, ob die Orientierung von S^1 und $f(S^1)$ übereinstimmt oder nicht.
- f „macht Knicke“: Damit ist anschaulich gemeint, dass die Abbildung die Kreislinie erst n mal in eine Richtung durchläuft und dann „abknickt“ und die Kreislinie nochmals genauso oft in die andere Richtung durchläuft. Hieraus ergibt sich auf Grund der lokalen Eigenschaft ebenfalls der Grad $\deg(f) = 0$. Verallgemeinert man dieses Beispiel, sodass die Abbildung die Kreislinie zuerst n_1 mal in die eine Richtung durchläuft, dann n_2 mal in die andere Richtung und danach wieder n_3 mal in die erste Richtung und so weiter, dann erkennt man, dass der Grad $\deg(f)$ für diesen Fall nicht immer gleich Null sein muss. Aus der lokalen Eigenschaft von f ergibt sich nämlich $\deg(f) = \sum n_i$, wobei die n_i aus der Menge

der ganzen Zahlen $\mathbb{Z}$, das heißt vorzeichenbehaftet, sind. Jeweils in die gleiche Richtung durchlaufene n_i haben dabei das gleiche Vorzeichen.

Jetzt sind alle Begriffe bekannt, welche zur Bestimmung der Randoperatoren ∂_i notwendig sind.

Gegeben sei also ein CW-Komplex $\mathcal{X}$ und ein n-Skelett X_n als Vereinigung aller k-Zellen b, welche die Eigenschaft $k \leq n$ erfüllen, aus der Menge B_k aller k-Zellen in $\mathcal{X}$. Nun heftet man mit Hilfe der Abbildung $f_c : S^n \to X_n$ eine $(n+1)$-Zelle c an das n-Skelett X_n. Dabei entspricht die n-Sphäre S^n dem Rand ∂_c der $(n+1)$-Zelle c. f_c kann also als Verklebungsabbildung der $(n+1)$-Zelle mit dem n-Skelett interpretiert werden.

Als Nächstes sei $b \in B_n$ eine n-Zelle aus der Menge B_n aller n-Zellen in $\mathcal{X}$. Man definiert jetzt eine Abbildung $f_{cb} : S^n \xrightarrow{f_c} X_n/(X_n \setminus b)$, welche die n-Sphäre S^n auf den Quotientenraum $X_n/(X_n \setminus b)$ abbildet. Der Quotientenraum $X_n/(X_n \setminus b)$ entsteht dabei dadurch, dass man alles außer der n-Zelle b auf einen Punkt kontrahiert. Der Quotientenraum $X_n/(X_n \setminus b)$ ist also homöomorph zur n-Sphäre. Nun bestimmt man den Grad $\deg(f_{cb})$ von f_{cb} und bezeichnet ihn als α_{cb}.

Im nächsten Schritt findet der Übergang vom CW-Komplex $\mathcal{X}$ zum Kettenkomplex $\mathcal{C}$ für $\mathcal{X}$ statt. Hierbei sei, für alle n, $C_n = \mathbb{R}B_n$ mit B_n als Menge aller n-Zellen in $\mathcal{X}$. Die C_n werden auch als n-Ketten bezeichnet. Dann ist der Randoperator $\partial_c : C_{n+1} \to C_n$ für $c \in B_{n+1}$ definiert als $\partial_c := \sum_{b \in B_n} \alpha_{cb} b$. Es handelt sich also bei den zuvor bestimmten α_{cb} um die gesuchten Matrixeinträge der Randoperatoren ∂_i. Abschließend wäre noch zu beweisen, dass der Rand vom Rand in diesem Fall wirklich Null ist, dass also in der Tat $\partial_i \circ \partial_{i+1} = 0$ gilt. Darauf soll hier verzichtet werden. Der Beweis ist unter anderem in Hatcher (2002) im Kapitel 2 nachzulesen.

2.1.5 Zelluläre Abbildungen

Nun als Letztes noch zur Definition der *zellulären Abbildungen*, welche zur Modellierung von Detaillierungsabbildungen benötigt werden.

Definition 2.1.29 (zelluläre Abbildung)**.** Seien X und Y CW-Komplexe. Dann heißt eine stetige Abbildung $f : X \to Y$ *zellulär*, wenn für jedes $n \leq m$ jede m-Zelle b aus X in das n-Skelett von Y abgebildet wird.

Ist eine Abbildung $f : X \to Y$ zwischen zwei CW-Komplexen X und Y zellulär, dann kann man diese auch als Abbildung zwischen den zugehörigen Kettenkomplexen $\mathcal{C}(X)$ und $\mathcal{C}(Y)$ folgendermaßen schreiben:

$$\begin{array}{lccccccc} \mathcal{C}(X): & \ldots & \xrightarrow{\partial_{m+1}} & C_m(X) & \xrightarrow{\partial_m} & C_{m-1}(X) & \xrightarrow{\partial_{m-1}} & \ldots \\ & \downarrow f & & \downarrow f_m & // & \downarrow f_{m-1} & & \\ \mathcal{C}(Y): & \ldots & \xrightarrow{\partial'_{m+1}} & C_m(Y) & \xrightarrow{\partial'_m} & C_{m-1}(Y) & \xrightarrow{\partial'_{m-1}} & \ldots \end{array}$$

Dabei bilden die Abbildungen f_i die i-Zellen aus X ins i-Skelett von Y ab und alle n-Zellen aus X mit $n < i$ werden auf den Nullvektor abgebildet.

Man erhält dabei dasselbe Ergebnis, egal ob man zuerst den Randoperator anwendet und dann die Abbildung ausführt ($f_{m-1} \circ \partial_m$) oder umgekehrt zuerst die Abbildung durchführt und dann den Randoperator anwendet ($\partial'_m \circ f_m$), das heißt also in diesem Fall ist $f_{m-1}(C_{m-1}(X)) \circ \partial_m(C_m(X)) = \partial'_m(C_m(Y)) \circ f_m(C_m(X)) = C_{m-1}(Y)$. Man sagt dazu, dass die Abbildung *kommutiert*. Dies wird durch die Schraffur innerhalb des Vierecks symbolisiert.

Auf diese Weise kann man zelluläre Detaillierungsabbildungen zwischen CW-Komplexen modellieren, bei welchen der weniger detaillierte CW-Komplex zellulär auf den detaillierteren CW-Komplex abgebildet wird. Dabei werden außerdem Zyklen auf Zyklen und Ränder auf Ränder abgebildet.

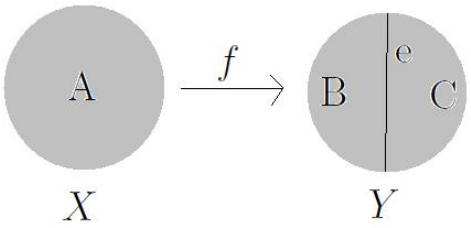


Abbildung 2.16: Beispiel für eine zelluläre Detaillierungsabbildung

Zur Veranschaulichung dieses Sachverhalts soll das folgende Beispiel dienen (siehe Abbildung 2.16). Hierbei wird die 2-Zelle A auf die zwei 2-Zellen B und C und die 1-Zelle e abgebildet. Man kann also eine zelluläre Detaillierungsabbildung f zwischen den beiden CW-Komplexen X und Y wie folgt definieren: $f : X \to Y$ mit $A \mapsto B \cup C \cup e$. Die umgekehrte Generalisierungsabbildung bei der B, C und e alle auf A abgebildet werden, wäre übrigens nicht zellulär, weil dabei die 1-Zelle e auf die höherdimensionale 2-Zelle A abgebildet wird.

Als Abbildung zwischen den zugehörigen Kettenkomplexen $\mathcal{C}(X)$ und $\mathcal{C}(Y)$ lässt sich die Detaillierungsabbildung f auch folgendermaßen schreiben:

$$\begin{array}{ccccccccc} \mathcal{C}(X): & \ldots & \xrightarrow{\partial_3} & C_2(X) & \xrightarrow{\partial_2} & C_1(X) & \xrightarrow{\partial_1} & \ldots \\ & \downarrow f & & \downarrow f_2 & // & \downarrow f_1 & & \\ \mathcal{C}(Y): & \ldots & \xrightarrow{\partial'_3} & C_2(Y) & \xrightarrow{\partial'_2} & C_1(Y) & \xrightarrow{\partial'_1} & \ldots \end{array}$$

Dabei sind $C_2(X) = \mathbb{R}\{A\}$, $C_2(Y) = \mathbb{R}\{B, C\}$, $C_1(X) = \mathbb{R}\{0\}$ und $C_1(Y) = \mathbb{R}\{e\}$ und es gilt $f_2 : C_2(X) \to C_2(Y)$ mit $A \mapsto B + C + 0 \cdot e$ und $f_1 : C_1(X) \to C_1(Y)$ mit $0 \mapsto e$.

2.2 Topologie in CityGML

Die geometrischen und topologischen Modelle von CityGML stehen in enger Beziehung zueinander (Gröger and Plümer, 2012; Gröger et al., 2012). Die räumlichen Eigenschaften von CityGML-Objekten werden durch Objekte des Geometriemodells der Geography Markup Language (GML3) (Cox et al., 2002) dargestellt. Dieses Modell basiert auf dem ISO-Standard 19107 „Spatial Schema“ (iso, 2019), der dreidimensionale Geometrien nach dem bekannten Prinzip der *Boundary Representation* (Foley et al., 1996) darstellt. Das GML3-Geometriemodell besteht aus Grundelementen, die zu Komplexen, Verbundgeometrien oder Aggregaten kombiniert werden können. Für jede Dimension gibt es ein geometrisches Grundelement wie *Point*, *Curve*, *Surface* und *Solid*. Die Darstellung von Flächen und Kurven ist auf planare Polygone beschränkt: Alle Koordinaten der äußeren Begrenzung und der optionalen inneren Grenzen (die Löcher im Polygon bilden) müssen in derselben Ebene liegen. Ebenso sind nur gerade Linien (gemäß der GML3-Klasse *LineString*) zulässig. CityGML stellt die explizite Modellierung der Topologie bereit, zum Beispiel durch die gemeinsame Nutzung von Geometrieobjekten durch Features oder andere Geometrien. Ein Teil des Raums sollte nur einmal von einem Geometrieobjekt dargestellt und von allen Features oder komplexeren Geometrien referenziert werden, die von diesem Geometrieobjekt definiert oder begrenzt werden. So soll Redundanz vermieden werden und explizite topologische Beziehungen zwischen den Teilen sollten erhalten bleiben. Anstatt Topologie mit eigenen XML-Tags zu implementieren, verwendet CityGML das XML-Konzept von *XLinks*, das

von GML3 bereitgestellt wird. Es ist jedoch nicht erforderlich, die Topologie auf diese Weise zu modellieren, um eine gültige CityGML-Datei zu erhalten.

2.2.1 Geometrisch-topologisches Modell

Die räumlichen Eigenschaften der CityGML-Objekte werden durch Objekte des GML3-Geometriemodells repräsentiert. Dieses Modell basiert auf dem ISO-Standard 19107 „Spatial Schema“, welche 3D-Geometrien nach der bekannten *Boundary Representation* darstellt, welche Geometrien mittels ihres Rands beschreibt. CityGML nutzt tatsächlich nur einen Teil des GML3-Geometriepakets, ein GML3-Profil definierend. Diese Teilmenge ist in Abbildung 2.17 und Abbildung 2.18 dargestellt.

Darüber hinaus wird die explizite *Boundary Representation* aus GML3 durch sogenannte *scene graph* Konzepte erweitert, welche die implizite Darstellung der Geometrie von Objekten derselben Form ermöglichen und die 3D-Stadtmodelle somit speicherplatzeffizienter machen sollen.

Das GML3-Geometriemodell besteht aus Primitiven, die kombiniert werden können, um Komplexe, zusammengesetzte Geometrien oder Aggregate zu bilden. Für jede Dimension gibt es ein entsprechendes geometrisches Primitiv: ein nulldimensionales Objekt ist ein Punkt (*Point*), ein eindimensionales Objekt ist eine Kurve beziehungsweise eine Linie (*Curve*), ein zweidimensionales Objekt ist eine Fläche (*Surface*) und ein dreidimensionales Objekt ist ein Volumenkörper (*Solid*) (siehe Abbildung 2.17). Jede Geometrie kann ihr eigenes Koordinatensystem besitzen. Ein Volumenkörper wird durch Flächen begrenzt und eine Fläche von Kurven. In CityGML dürfen diese Kurven nur gerade Linien sein, weshalb nur die GML3-Klasse *LineString* verwendet wird. Ein *LineString* ist, auch wenn der Name etwas irreführend gewählt wurde, die geradlinige Verbindung zwischen zwei Punkten. Flächen werden in CityGML durch Polygone (*Polygons*) dargestellt, welche eine ebene Geometrie definieren, das heißt, dass der Rand und alle inneren Punkte in einer Ebene liegen müssen.

Zusammengesetzte Geometrien können Aggregate, Komplexe oder Composites der Primitive sein (siehe Darstellung in Abbildung 2.19). Bei einem Aggregat (*Aggregate*) ist die räumliche Beziehung zwischen den Komponenten nicht eingeschränkt. Sie können disjunkt sein, sich überschneiden, sich berühren oder

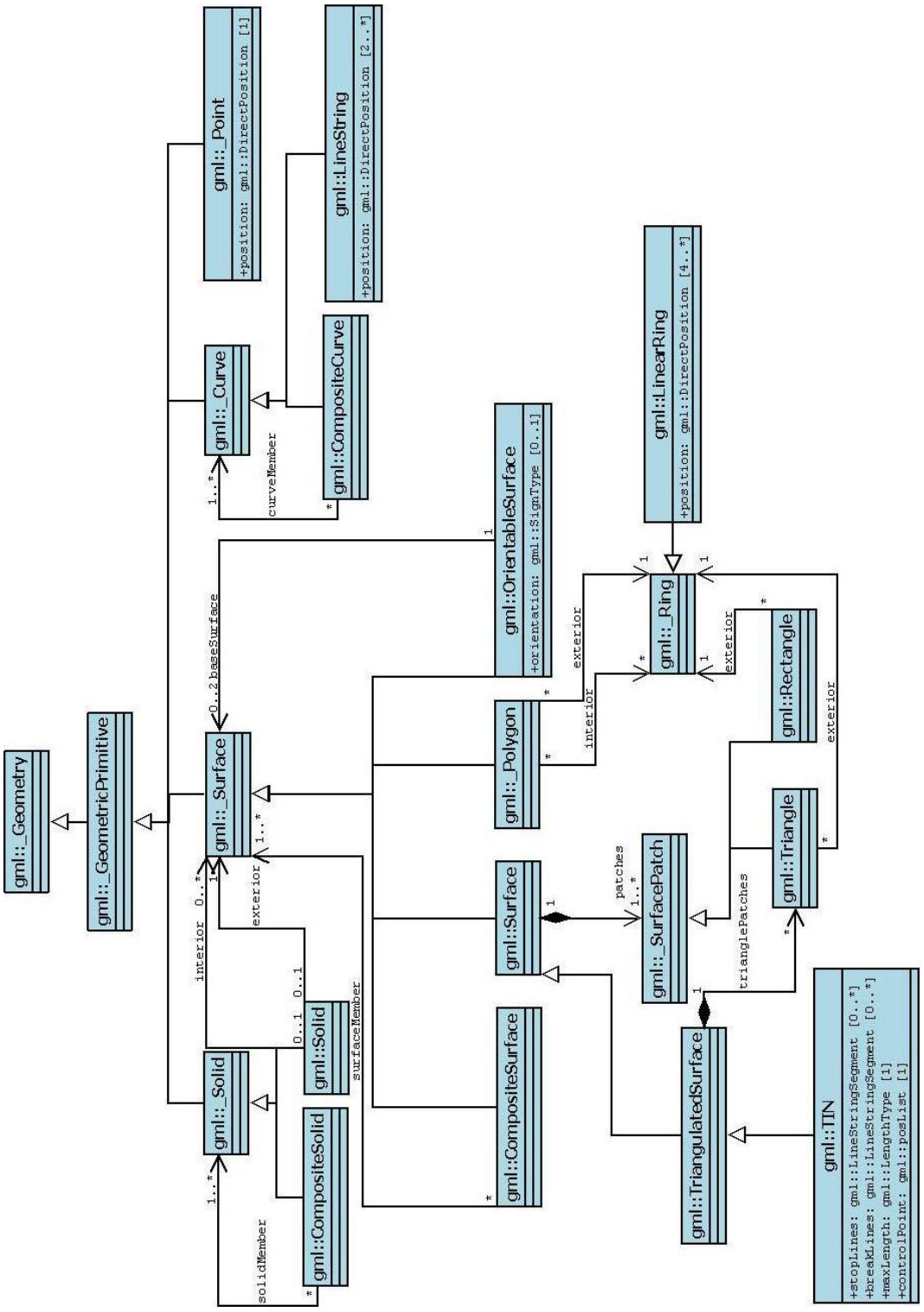


Abbildung 2.17: UML-Diagramm des CityGML-Geometriemodells: Primitive und zusammengesetzte Geometrien (Inhalt aus Gröger et al. (2012))

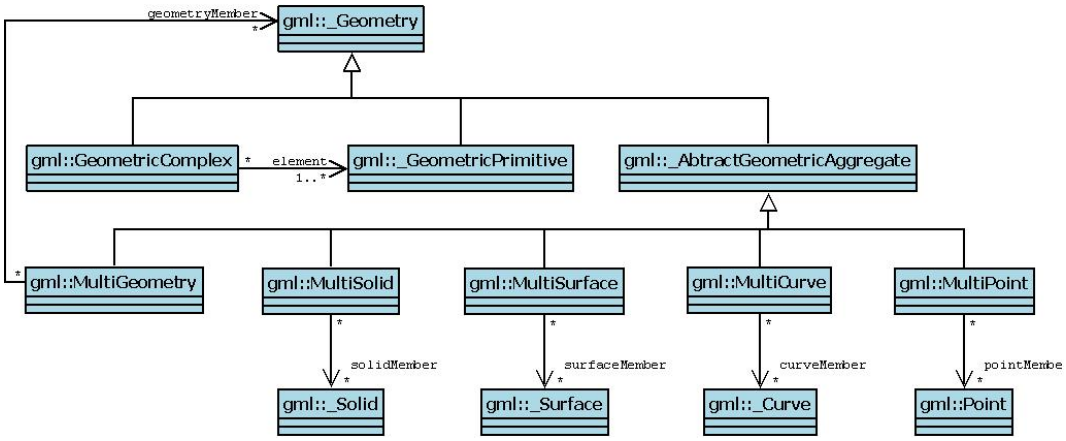


Abbildung 2.18: UML-Diagramm des CityGML-Geometriemodell: Komplexe und Aggregate (Inhalt aus Gröger et al. (2012))

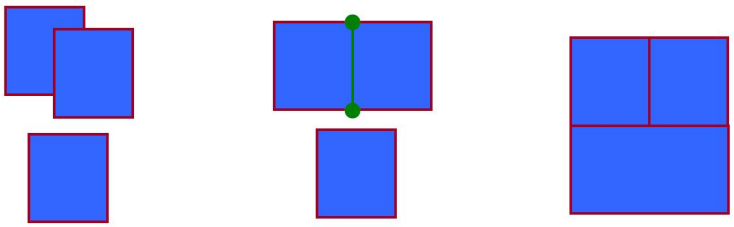


Abbildung 2.19: zusammengesetzte Geometrien (Quelle Gröger et al. (2008))

unzusammenhängend sein (siehe linkes Bild in Abbildung 2.19). GML3 stellt ein spezielles Aggregat für jede Dimension bereit (*MultiPoint*, *MultiCurve*, *MultiSurface* und *MultiSolid*, siehe Abbildung 2.18). Im Gegensatz zu Aggregaten sind Komplexe topologisch strukturiert: ihre Teile müssen disjunkt sein, dürfen sich nicht überschneiden und dürfen sich allenfalls an ihren Rändern berühren oder Teile ihrer Ränder gemeinsam haben (siehe mittleres Bild in Abbildung 2.19). Ein *Composite* ist ein spezieller Komplex, welcher von GML3 zur Verfügung gestellt wird. Es können nur Elemente enthalten sein, die die gleiche Dimension haben. Seine Elemente müssen ebenfalls disjunkt sein, aber sie müssen außerdem entlang ihrer Ränder verbunden werden (siehe rechtes Bild in Abbildung 2.19). Ähnlich wie bei den Aggregaten wird von GML3 auch für jede Dimension (außer nulldimensional, weil das keinen Sinn ergeben würde) ein Composite zur Verfügung gestellt (*CompositeSolid*, *CompositeSurface* und *CompositeCurve*, siehe Abbildung 2.18).

Eine *OrientableSurface* ist eine Fläche mit einer expliziten Orientierung, das

heißt zwei Seiten (Vorder- und Rückseite) können unterschieden werden. Dies kann zum Beispiel verwendet werden, um Texturen auf eine bestimmte Seite einer Fläche zu legen, oder um die Außen- und die Innenseite der Oberfläche zu unterscheiden, wenn ein Volumenkörper durch die Fläche begrenzt wird. Zu beachten gilt, dass Kurven und Flächen eine standardmäßige Orientierung in GML3 haben, die sich aus der Definitionsreihenfolge der Punkte ergibt. Deshalb muss eine *OrientableSurface* nur verwendet werden, wenn die Orientierung einer bestimmten GML3-Geometrie umgekehrt werden soll.

Das GML3-Modell ermöglicht ein rekursives Aggregationsschema für jeden primitiven Typ der entsprechenden Dimension. Dieses Aggregationsschema erlaubt die Definition von verschachtelten Aggregationen (Hierarchie der Komponenten). Zum Beispiel kann die Geometrie eines Gebäudes (*CompositeSolid*) zusammengesetzt sein aus der Geometrie des Hauses (*CompositeSolid*) und der Geometrie der Garage (*Solid*), während die Geometrie des Hauses wiederum weiter unterteilt ist in die Geometrie des Daches (*Solid*) und die Geometrie des Hauskörpers (*Solid*).

CityGML ermöglicht die explizite Modellierung der Topologie und will damit zum Beispiel die gemeinsame Nutzung von einzelnen Geometrieobjekten durch mehrere Objekte oder andere Geometrien möglich machen. Ein Teil des Raumes soll immer nur von einem Geometrieobjekt repräsentiert werden, auf welches sich dann alle Objekte oder komplexere Geometrien beziehen, die durch dieses Geometrieobjekt definiert sind oder begrenzt werden. So soll Redundanz vermieden werden und explizite topologische Beziehungen zwischen den Teilen sollen erhalten bleiben. Wie allerdings die Überprüfung der Forderung nach Redundanzfreiheit aussehen soll, wird in der Spezifikation nicht geklärt. Diese Frage soll unter anderem im Abschnitt 2.2.2 diskutiert werden.

Grundsätzlich gibt es drei Fälle. Erstens können zwei Objekte räumlich durch die gleiche Geometrie definiert werden. Wenn zum Beispiel ein Weg sowohl eine Transportfunktion als auch eine Vegetationsfunktion erfüllt, sollte die Oberflächengeometrie des Weges durch das Transportobjekt und ebenfalls durch das Vegetationsobjekt benutzt werden. Zweitens kann eine Geometrie von einem Objekt und einer weiteren Geometrie gemeinsam genutzt werden. Eine Geometrie, welche die Wand eines Gebäudes definiert, kann zum Beispiel zweimal referenziert werden: durch den Volumenkörper, der die Geometrie des

Gebäudes festlegt und durch das Wandobjekt. Drittens können zwei Geometrien die gleiche Geometrie referenzieren, welche Teil der Grenze von beiden ist. Zum Beispiel können ein Gebäude und eine angrenzende Garage mit Hilfe von zwei Volumenkörpern dargestellt werden. Die Fläche, die den Bereich beschreibt, wo sich beide Volumenkörper berühren, wird dann nur einmal beschrieben und von beiden Volumenkörpern benutzt. Wie aus Abbildung 2.20 ersichtlich, erfordert dies eine Aufteilung der jeweiligen Flächen.

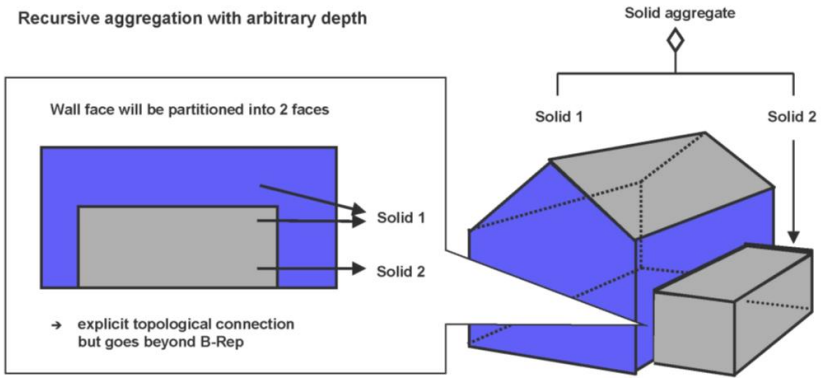


Abbildung 2.20: rekursive Aggregation von Objekten und Geometrien in CityGML (Quelle: IGG Uni Bonn)

Im Allgemeinen berücksichtigt die *Boundary Representation* nur sichtbare Flächen. Allerdings werden, um explizite topologische Nachbarschaft zu ermöglichen und die Möglichkeit des Löschens eines Teils eines zusammengesetzten Objekts zu schaffen, ohne Löcher in den verbleibenden Teilen zu hinterlassen, sich berührende Elemente erlaubt. Während Berühren erlaubt ist, ist es die Durchdringung von Objekten nicht, um die mehrfache Darstellung des gleichen Raumes zu vermeiden. Allerdings ist der Einsatz von Topologie in CityGML optional.

Statt Topologie zu implementieren nutzt CityGML das XML-Konzept von *XLinks*, welches von GML3 Verfügung gestellt wird. Jedem Geometrieobjekt, auf das von verschiedenen geometrischen Aggregaten oder verschiedenen thematischen Objekten zugegriffen werden soll, wird eine eindeutige Kennung (ID) zugeordnet, auf welche die GML-Geometrie-Eigenschaften mit Hilfe des Attributs *href:xlink* referenziert werden können. CityGML benutzt nicht das integrierte Topologie-Paket von GML3, welches separate Topologie-Objekte begleitend zur Geometrie liefert. Diese Art der Topologie ist sehr komplex und

aufwändig. Dennoch fehlt es an Flexibilität, wenn Datensätze, die Topologie enthalten können oder auch nicht, durch das gleiche Datenmodell erfasst werden sollen. Die XLink-Topologie ist einfach, flexibel und fast so mächtig, wie das explizite GML3-Topologiemodell. Im Abschnitt 2.2.2 wird untersucht, wie das vorgestellte topologische Modell im Schema tatsächlich umgesetzt wurde und was man damit zusätzlich modellieren kann.

In der Spezifikation (Gröger et al., 2012) wird zusätzlich behauptet, es sei ein Nachteil der XLink-Topologie, dass die Navigation zwischen den topologisch verbundenen Objekten nur in einer Richtung (von einem Aggregat, zu seiner Komponente) durchgeführt werden könne und nicht (unmittelbar) bidirektional möglich sei, wie es bei der in GML3 integrierten Topologie der Fall ist. Warum das so sein soll, erschließt sich nicht direkt. Dieses Thema wird im Abschnitt 2.2.2 nochmals diskutiert.

2.2.2 Modellierung der Topologie

In diesem Abschnitt soll untersucht werden, wie Topologie in CityGML tatsächlich umgesetzt ist und was damit zusätzlich modelliert werden kann. Zunächst wird festgestellt, wie das in Abschnitt 2.2.1 theoretisch beschriebene geometrisch-topologische Modell der XLink-Topologie umgesetzt wurde. Hierfür ist es wichtig herauszufinden, wie man Objekten eine ID zuweist und wie man auf diese ID dann wiederum zugreift. Außerdem ist es wichtig zu wissen, wie Knoten, Kanten oder Flächen als einzelne Objekte modelliert werden können. Zusätzlich gilt es zu klären, was mit redundanten Knoten, Kanten oder Flächen passiert. Es soll also untersucht werden, ob gewährleistet ist, dass ein Teil des Raumes nicht mehrmals repräsentiert werden kann. Abschließend soll die, in der Spezifikation (Gröger et al., 2012) aufgestellte, Behauptung, dass die Navigation zwischen den, in der XLink-Topologie topologisch verbundenen, Objekten nur in einer Richtung (von einem Aggregat, zu seiner Komponente) durchgeführt werden könne und nicht (unmittelbar) bidirektional möglich sei, diskutiert werden.

ID-Vergabe und Referenzierung

Begonnen werden soll damit, zu klären wie IDs vergeben werden können und wie auf diese zugegriffen werden kann. Um IDs zu vergeben wird das GML3-Attribut *gml:id* verwendet. Dies wird ganz einfach beim Öffnen des jeweiligen Tags definiert und kann eine beliebige Zeichenkette (*string*) sein. Hierzu soll ein einfaches Beispiel betrachtet werden (siehe Abbildung 2.21).

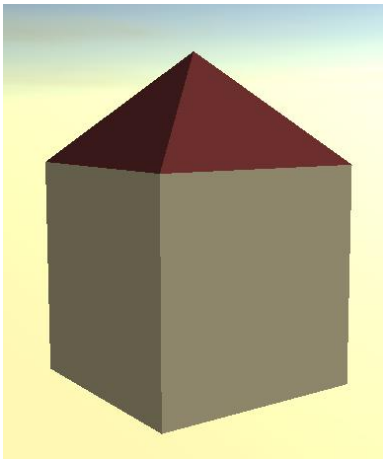


Abbildung 2.21: Bild des Hauses aus dem Beispiel

Dazu hier folgt ein gekürzter Ausschnitt aus der zugehörigen XML-Datei:

```
<?xml version="1.0" encoding="UTF-8"?>
<core:CityModel xmlns="http://www.citygml.org/citygml/1/0/0"
                xmlns:gml="http://www.opengis.net/gml"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:bldg="http://www.opengis.net/citygml/building/1.0"
                xmlns:core="http://www.opengis.net/citygml/1.0"
                xmlns:xlink="http://www.w3.org/1999/xlink"
                xsi:schemaLocation="http://www.citygml.org/citygml/1/0/0
                                    http://www.citygml.org/citygml/1/0/0/CityGML.xsd">
 <core:cityObjectMember>
  <bldg:Building gml:id="building1">
  <bldg:roofType>1120</bldg:roofType>
   <bldg:GroundSurface>
    <bldg:lod2MultiSurface>
     <gml:MultiSurface>
      <gml:surfaceMember>
       <gml:Polygon gml:id="poly1">
        ...
       </gml:Polygon>
      </gml:surfaceMember>
     </gml:MultiSurface>
    </bldg:lod2MultiSurface>
```

```
   </bldg:GroundSurface>
   <bldg:WallSurface>
    <bldg:lod2MultiSurface>
     <gml:MultiSurface>
      <gml:surfaceMember>
       <gml:Polygon gml:id="poly2">
        ...
       </gml:Polygon>
      </gml:surfaceMember>
      ...
     </gml:MultiSurface>
    </bldg:lod2MultiSurface>
   </bldg:WallSurface>
   <bldg:RoofSurface>
    ...
   </bldg:RoofSurface>
  </bldg:Building>
 </core:cityObjectMember>
</core:CityModel>
```

Im Tag `<core:CityModel>` wurde die Zeile `xmlns:xlink="http://www.w3.org/1999/xlink"` hinzugefügt. Diese enthält den Verweis auf die Quelle der XLink-Schema-Datei. Bei den Objekten, welche eine ID bekommen sollten, wurde diese beim Öffnen des Tags festgelegt, wie zum Beispiel beim Polygon, welcher die Bodenfläche beschreibt (`<gml:Polygon gml:id="poly1">`).

Als Nächstes gilt es die Frage zu klären, wie man auf die so vergebenen IDs und die dazugehörigen Objekte beziehungsweise Geometrien an anderer Stelle zugreifen kann. Dazu wird das Attribut *xlink:href* benutzt. Anstatt der ausführlichen Definition des zu referenzierenden Objekts beziehungsweise der zu referenzierenden Geometrie steht in der Tag-Definition dann nur `xlink:href="#ID"`, die komplette Tag-Definition würde also so aussehen: `<tag xlink:href="#ID"/>`. Wollte man zum Beispiel auf das Polygon, welches die Bodenfläche des Beispiels von oben beschreibt, nochmals zugreifen, dann müsste man zum Beispiel `<gml:surfaceMember xlink:href="#poly1"/>` schreiben.

Gelegenheiten dieses anzuwenden ergeben sich unter anderem, wenn man ein Gebäude nicht nur in einem LoD definieren möchte, sondern in zwei oder mehreren LoDs, weil dann häufig Teile der Geometrie identisch sind. Deshalb folgt nun eine Erweiterung unseres Beispiels, in der das Häuschen auch noch im LoD1 definiert wird. Hier kommen also Teile aus der entsprechenden XML-Datei:

```
<?xml version="1.0" encoding="UTF-8"?>
<core:CityModel ...
                xmlns:xlink="http://www.w3.org/1999/xlink"
                ...
 <core:cityObjectMember>
  <bldg:Building gml:id="building1">
  <bldg:roofType>1120</bldg:roofType>
   <bldg:GroundSurface>
    <bldg:lod2MultiSurface>
     <gml:MultiSurface>
      <gml:surfaceMember>
       <gml:Polygon gml:id="poly1">
        ...
       </gml:Polygon>
      </gml:surfaceMember>
     </gml:MultiSurface>
    </bldg:lod2MultiSurface>
   </bldg:GroundSurface>
   <bldg:WallSurface>
    <bldg:lod2MultiSurface>
     <gml:MultiSurface>
      <gml:surfaceMember>
       <gml:Polygon gml:id="poly2">
        ...
       </gml:Polygon>
      </gml:surfaceMember>
      <gml:surfaceMember>
       <gml:Polygon gml:id="poly3">
        ...
       </gml:Polygon>
      </gml:surfaceMember>
      <gml:surfaceMember>
       <gml:Polygon gml:id="poly4">
        ...
       </gml:Polygon>
      </gml:surfaceMember>
      <gml:surfaceMember>
       <gml:Polygon gml:id="poly5">
        ...
       </gml:Polygon>
      </gml:surfaceMember>
     </gml:MultiSurface>
    </bldg:lod2MultiSurface>
   </bldg:WallSurface>
   <bldg:RoofSurface>
    ...
   </bldg:RoofSurface>
   <bldg:lod1Solid>
    <gml:Solid>
     <gml:exterior>
      <gml:CompositeSurface>
       <gml:surfaceMember>
```

```
        <gml:Polygon gml:id="poly10">
         ...
        </gml:Polygon>
       </gml:surfaceMember>
       <gml:surfaceMember xlink:href="#poly1"/>
       <gml:surfaceMember xlink:href="#poly2"/>
       <gml:surfaceMember xlink:href="#poly3"/>
       <gml:surfaceMember xlink:href="#poly4"/>
       <gml:surfaceMember xlink:href="#poly5"/>
      </gml:CompositeSurface>
     </gml:exterior>
    </gml:Solid>
   </bldg:lod1Solid>
  </bldg:Building>
 </core:cityObjectMember>
</core:CityModel>
```

Zunächst wird hier das Haus im LoD2 genauso wie in den vorherigen Beispielen definiert. Den Polygonen, welche die einzelnen Flächen beschreiben, werden dabei IDs zugewiesen. Dies geschieht in den entsprechenden Zeilen der Form `<gml:Polygon gml:id="polyX">`. Dann folgt, innerhalb derselben Gebäudedefinition, die Definition des Hauses im LoD1. Dabei soll das Haus hier nur noch durch einen Würfel ohne Dach, entsprechend dem für LoD1 in der Spezifikation beschriebenen Blöckemodell, repräsentiert werden. Auch die semantischen Informationen zu den Flächen, ob es sich um eine Boden-, Wand- oder Dachfläche handelt, fallen hier weg, da diese sowohl in der Spezifikation als auch im XML-Schema für LoD1 ebenfalls nicht vorgesehen sind.

Als Geometrietyp wurde für das Haus im LoD1 ein Volumenkörper (*lod1-Solid*) gewählt. Auch in GML3 heißt der entsprechende Datentyp *Solid*. Sein Äußeres (*exterior*) wird durch eine *CompositeSurface* repräsentiert, welche sich wiederum aus einzelnen *surfaceMembers* zusammensetzt. Von diesen Umgebungsflächen muss jetzt nur noch eine einzige explizit definiert werden. Dies ist die obere Fläche des Würfels, an deren Stelle sich im LoD2 das Dach befunden hat. Alle anderen Flächen, also die seitlichen Begrenzungsflächen und die Bodenfläche, wurden bereits vorher innerhalb der Gebäudedefinition im LoD2 definiert und werden hier über die vergebene ID wieder aufgerufen. Die Zeilen der Form `<gml:surfaceMember xlink:href="#polyX"/>` werden dazu verwendet. Eine Definition auf diese Art spart viel Schreibarbeit und sorgt außerdem für Redundanzfreiheit. Allerdings ist es so, dass die Kontrolle der Redundanzfreiheit dem Nutzer beziehungsweise der Anwendung überlassen wird.

Zu diesem Thema folgt an anderer Stelle noch eine weitere Diskussion.

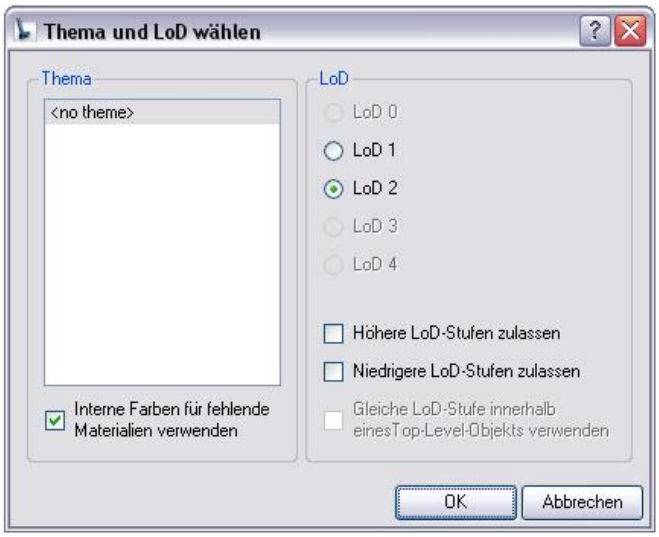


Abbildung 2.22: Auswahlmenü von LandXplorer für das Beispiel in LoD1 und LoD2

Definiert man ein Gebäude innerhalb einer Gebäudedefinition in mehreren LoDs, dann wird man beim Öffnen des Modells mit einem Viewer dazu aufgefordert zu wählen in welchem LoD das Modell angezeigt werden soll. In Abbildung 2.22 ist das entsprechende Fenster zu sehen, welches auftaucht, wenn man das Beispiel mit dem *LandXplorer CityGML Viewer* von *Autodesk* öffnet. Wählt man hier LoD2 aus, dann erhält man das Haus aus Abbildung 2.21, wie in allen bisherigen Beispielen. Wählt man LoD1 wird das Haus nur als Würfel angezeigt (siehe Abbildung 2.23). Den gleichen Effekt kann man mit der Beziehung *generalizesTo* erreichen.

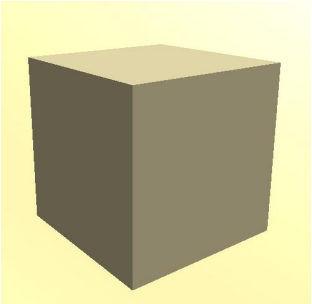


Abbildung 2.23: Haus aus dem Beispiel im LoD1

Mit Hilfe dieser Vergabe und Referenzierung von IDs lassen sich schon eini-

ge topologische Beziehungen modellieren. Zwei Gebäude sind zum Beispiel benachbart, wenn sie eine gemeinsame Wandfläche haben. Allerdings funktioniert das nur zuverlässig, wenn die Redundanzfreiheit gewährleistet ist und das muss, wie bereits erwähnt, nicht immer der Fall sein. In der Spezifikation steht zwar, dass kein Teil des Raumes zweimal repräsentiert werden soll, aber es wird dafür kein Kontrollmechanismus im XML-Schema zur Verfügung gestellt. Das XML-Schema lässt mehrfache Definitionen derselben Geometrieobjekte also zu. Man kann zum Beispiel das Haus aus dem Beispiel im LoD1 nochmals komplett neu, ohne Verwendung der IDs definieren. In diesem Fall wären dann insgesamt fünf Flächen doppelt definiert worden.

Modellierung von Punkten

Bisher war die kleinste geometrische Einheit auf die zugegriffen werden konnte, immer das Polygon, welches eine Fläche beschreibt. Um die Topologie vollständig modellieren zu können, ist es allerdings auch wünschenswert auf kleinere geometrische Elemente, wie Kanten oder Knoten (Punkte), direkt zugreifen zu können. Deshalb soll als Nächstes geklärt werden, ob und wie das funktioniert.

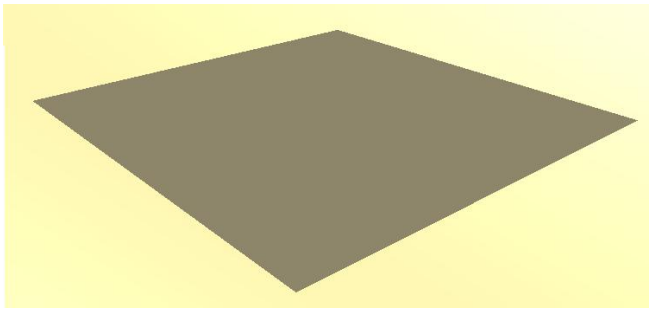


Abbildung 2.24: Bodenfläche des Hauses

Begonnen werden soll damit herauszufinden, wie man in CityGML einzelne Punkte modellieren kann. Hierzu soll ein noch einfacheres Beispiel verwendet werden, als das Haus aus den bisherigen Beispielen, nämlich nur eine quadratische Fläche, bei der es sich um die Bodenfläche des Häuschens handelt (siehe Abbildung 2.24). Nach dem bisherigen Wissensstand würde man diese folgendermaßen definieren:

```
<?xml version="1.0" encoding="UTF-8"?>
<core:CityModel ...>
```

```
<core:cityObjectMember>
 <bldg:Building>
  <bldg:lod2MultiSurface>
   <gml:MultiSurface>
    <gml:surfaceMember>
     <gml:Polygon>
      <gml:exterior>
       <gml:LinearRing>
        <gml:pos>0 0 0</gml:pos>
        <gml:pos>0 2 0</gml:pos>
        <gml:pos>2 2 0</gml:pos>
        <gml:pos>2 0 0</gml:pos>
        <gml:pos>0 0 0</gml:pos>
       </gml:LinearRing>
      </gml:exterior>
     </gml:Polygon>
    </gml:surfaceMember>
   </gml:MultiSurface>
  </bldg:lod2MultiSurface>
 </bldg:Building>
</core:cityObjectMember>
</core:CityModel>
```

Hier ist der *LinearRing* wieder als Liste von Koordinatentripeln definiert worden. Allerdings wäre es wünschenswert diesen aus Punkten aufzubauen. Um herauszufinden, ob und wie sich das umsetzen lässt, schaut man am Besten ins XML-Schema von GML3, wo herauszufinden ist, wie ein *LinearRing* definiert werden kann.

Auch das XML-Schema von GML3 ist, ebenso wie das CityGML-Schema, in mehrere Schemadateien unterteilt. In diesen Schemadateien von GML3 finden sich häufig erläuternde Anmerkungen, welche beim Verständnis sehr hilfreich sein können. Die Definitionen des Elements *LinearRing* und des zugehörigen Datentyps *LinearRingType* findet man in der Datei *geometryBasic2d.xsd*. Auf das Abdrucken des kompletten XML-Codes dieser Definitionen soll hier verzichtet werden, aber auch hierzu gibt es eine der erwähnten erläuternden Anmerkungen, welche schon ganz gut weiterhilft. Sie lautet:

„GML supports two different ways to specify the control points of a linear ring. 1. A sequence of “pos” (DirectPositionType) or “pointProperty” (PointPropertyType) elements. “pos” elements are control points that are only part of this ring, “pointProperty” elements contain a point that may be referenced from other geometry elements or reference another point defined outside of this ring (reuse of existing points). 2. The “posList” element allows for a compact

way to specifiy the coordinates of the control points, if all control points are in the same coordinate reference systems and belong to this ring only. The number of direct positions in the list must be at least four.“

Man kann einen *LinearRing* also auf verschiedene Arten definieren. Eine dieser Möglichkeiten, nämlich über den Datentyp *pointProperty*, ist genau das, was für das Beispiel gebraucht wird. Der Datentyp *pointProperty* enthält einen Punkt, welcher entweder dort definiert wird und dann an anderer Stelle referenziert werden kann oder welcher umgekehrt woanders bereits definiert wurde und dort referenziert wird. Stöbert man noch ein bisschen mehr in den Schemadateien von GML3, dann überblickt man schnell, dass man den Datentyp *pointProperty* überall dort auch anwenden kann, wo Geometrien aus Punkten aufgebaut werden sollen. Den *LinearRing* für die Fläche aus dem Beispiel könnte man also auch wie folgt definieren:

```
<gml:LinearRing>
 <gml:pointProperty>
  <gml:Point gml:id="punktA">
   <gml:pos>0 0 0</gml:pos>
  </gml:Point>
 </gml:pointProperty>
 <gml:pointProperty>
  <gml:Point gml:id="punktB">
   <gml:pos>0 2 0</gml:pos>
  </gml:Point>
 </gml:pointProperty>
 <gml:pointProperty>
  <gml:Point gml:id="punktC">
   <gml:pos>2 2 0</gml:pos>
  </gml:Point>
 </gml:pointProperty>
 <gml:pointProperty>
  <gml:Point gml:id="punktD">
   <gml:pos>2 0 0</gml:pos>
  </gml:Point>
 </gml:pointProperty>
 <gml:pointProperty xlink:href="#punktA"/>
</gml:LinearRing>
```

Hier wird nun der *LinearRing* mittels des Datentyps *pointPropertys* definiert. Die in den *pointProperty*-Tags definierten Punkte erhalten jeweils eine ID. Der letzte Punkt ist beim *LinearRing* immer identisch zum ersten Punkt und muss in diesem Fall nicht nochmals neu definiert werden, sondern kann mit Hilfe der vergebenen ID wieder aufgerufen werden. Die ID-Vergabe und Referenzierung funktioniert dabei genauso wie im vorigen Beispiel.

Durch die Möglichkeit Punkte explizit zu definieren, an anderer Stelle zu referenzieren und die Polygone daraus aufzubauen, lässt sich die Topologie schon viel detaillierter modellieren, als wenn die Polygone die kleinste Einheit wären, auf die zugegriffen werden kann.

In diesem Fall gibt es allerdings dasselbe bereits im Abschnitt zuvor erwähnte Problem mit der Redundanzfreiheit. Diese ist nicht immer zwangsläufig gewährleistet. Auch bei diesem Beispiel kann man den letzten Punkt des Polygons, welcher identisch zum ersten Punkt sein sollte, einfach noch mal neu definieren. Es liegt also auch hier in der Verantwortung des Nutzers, darauf zu achten, dass keine Punkte redundant definiert werden.

Ist die Redundanzfreiheit nicht gegeben, so kann dies zu Problemen beim Modellieren der Topologie führen, weil die topologischen Beziehungen allein über die XLink-Beziehungen hergestellt werden, das heißt zwei Objekte, welche ein gemeinsames Geometrieelement enthalten, werden miteinander verknüpft, indem nur in einem von beiden Objekten das gemeinsame Geometrieelement definiert wird und das andere Objekt dieses dann mittels XLink ebenfalls benutzt. Wäre dieses Geometrieelement nun in beiden Objekten redundant definiert worden, dann gäbe es keine XLink-Beziehung zwischen den Objekten und auch die topologische Beziehung zwischen ihnen wäre nicht erkennbar.

Modellierung von Kanten

Als Nächstes wäre es zur Modellierung der Topologie noch wünschenswert, dass auch das Modellieren von Kanten in CityGML möglich wäre. Man könnte die Objekte dann aus den geometrischen Primitiven (Knoten beziehungsweise Punkten, Kanten beziehungsweise Linien und Flächen) zusammensetzen und sie so beispielsweise als CW-Komplexe beziehungsweise als Kettenkomplexe interpretieren.

Linienförmige Objekte werden in CityGML mittels des GML3-Datentyps *LineString* definiert, welcher, auch wenn sein Name anderes vermuten lassen könnte, die geradlinige Verbindung zwischen zwei Punkten darstellt. Diese Objekte sind ab LoD2 verfügbar und dazu vorgesehen, um zum Beispiel linienförmige Gebäudeteile, welche über den Volumenkörper hinaus stehen, zu modellieren. Dabei könnte es sich zum Beispiel um Antennen, Leitungen oder Ähnliches handeln. Deshalb wurde als Veranschaulichungsbeispiel dem Häus-

chen aus den bisherigen Beispielen eine senkrecht nach oben stehende Antenne auf das Dach gesetzt. Es folgt der zugehörige Ausschnitt aus der XML-Datei:

```
<?xml version="1.0" encoding="UTF-8"?>
<core:CityModel ...>
 <core:cityObjectMember>
  <bldg:Building gml:id="building1">
   ...
   <bldg:outerBuildingInstallation>
    <bldg:BuildingInstallation>
     <bldg:lod2Geometry>
      <gml:LineString>
       <gml:pos>1 1 3</gml:pos>
       <gml:pos>1 1 4</gml:pos>
      </gml:LineString>
     </bldg:lod2Geometry>
    </bldg:BuildingInstallation>
   </bldg:outerBuildingInstallation>
  </bldg:Building>
 </core:cityObjectMember>
</core:CityModel>
```

Das Haus wird darin zunächst wie üblich definiert und danach folgt die Definition der Antenne als *outerBuildingInstallation*. Es handelt sich dabei um eine LoD2-Geometrie und es wird der erwähnte Datentyp *LineString* verwendet. Auch Objekte dieses Datentyps kann man mit Hilfe des Datentyps *pointProperty* definieren, welcher oben schon zur Definition der Punkte eines *LinearRing* genutzt wurde. Das Ergebnis ist wie gewünscht ein Haus mit Antenne auf dem Dach (siehe Abbildung 2.25).

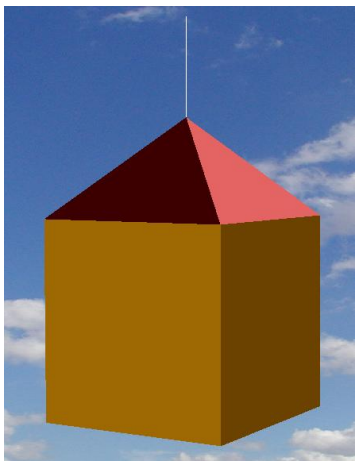


Abbildung 2.25: Das Haus mit Antenne (mit Aristoteles angezeigt)

Es ist also möglich geradlinige Objekte in CityGML zu modellieren. Was allerdings nicht möglich ist, ist Polygone, über welche in CityGML fast alle flächenförmigen Objekten beschrieben werden, aus solchen Objekten aufzubauen. Dieser Sachverhalt lässt sich zum Beispiel aus dem UML-Diagramm des GML3-Geometriemodells (Abbildung 2.17) erkennen. Ein Polygon kann immer nur durch innere (*interior*) und äußere (*exterior*) Ringe (abstrakte Klasse *Ring*) definiert werden und ein Ring wird immer durch einen *LinearRing* festgelegt, welcher wiederum nur über punktförmige Datentypen (*pos* oder *pointProperty*) definiert werden kann, wie bereits im Abschnitt zuvor an Hand des XML-Schemas nachvollzogen wurde.

Es ist also nicht ohne Weiteres möglich die Objekte, wie oben als wünschenswert zur Modellierung der Topologie erwähnt, direkt aus geometrischen Primitiven, wie Knoten, Kanten und Flächen zusammenzusetzen. Braucht man diese Aufteilung zur Modellierung der Topologie trotzdem, dann muss man einen kleinen Trick anwenden. Man könnte, als separates Objekt, das Kantengerüst des gewünschten Objektes definieren, wobei man zur Definition der Kanten, wie auch der Polygone für die Flächen des Objekts, jeweils den Datentyp *pointProperty* einsetzen müsste. Dann würden die Punkte entweder im Kantengerüst oder im Originalobjekt definiert und von dem entsprechenden anderen Objekt ebenfalls genutzt. So würde mittels XLink eine Beziehung zwischen dem Kantengerüst und dem Originalobjekt hergestellt.

Hierzu soll als Beispiel wieder die Bodenfläche des Hauses dienen. Die Definition des *LinearRing* für diese Fläche sieht so aus:

```
<gml:LinearRing>
 <gml:pointProperty>
  <gml:Point gml:id="punktA">
   <gml:pos>0 0 0</gml:pos>
  </gml:Point>
 </gml:pointProperty>
 <gml:pointProperty>
  <gml:Point gml:id="punktB">
   <gml:pos>0 2 0</gml:pos>
  </gml:Point>
 </gml:pointProperty>
 ...
</gml:LinearRing>
```

Die Punkte, die diesen *LinearRing* beschreiben, werden mit Hilfe des Datentyps *pointProperty* definiert und erhalten IDs. Man könnte jetzt die Kante

zwischen den Punkten *punktA* und *punktB* zusätzlich als eigenes Objekt folgendermaßen definieren:

```
<bldg:lod2Geometry>
 <gml:LineString>
  <gml:pointProperty xlink:href="#punktA"/>
  <gml:pointProperty xlink:href="#punktB"/>
 </gml:LineString>
</bldg:lod2Geometry>
```

Um die Punkte *punktA* und *punktB* aufzurufen, welche bereits innerhalb der Definition des *LinearRing* definiert worden sind, wird hier die XLink-Beziehung verwendet. Ebenso könnte man alle anderen Außenkanten der Fläche definieren. Diese Kanten stehen dann in Beziehung zum Polygon und somit zur Fläche, da die XLink-Beziehung verwendet wurde.

Auf diese Art ist es also (theoretisch) möglich, CityGML-Objekte in geometrische Primitive wie Knoten, Kanten und Flächen aufzuteilen. Interpretiert man diese Primitive nun als n-Zellen, so kann man die CityGML-Objekte als CW-Komplexe beziehungsweise als Kettenkomplexe auffassen. Die Matrizen der Randoperatoren ∂_i haben dann als Einträge allerdings keine Zahlenwerte, sondern nur x oder keinen Eintrag, wie der topologische Datentyp bei Graphen, da sich hier nur die Beziehungen „ist Rand von“, woraus sich dann der Eintrag x ergibt, und „ist nicht Rand von“, woraus sich kein Eintrag ergibt, modellieren lassen.

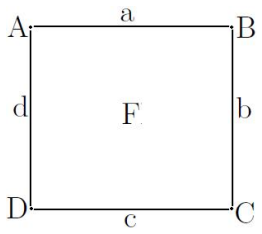


Abbildung 2.26: rechteckige Fläche als Beispiel

Eine rechteckige Fläche (siehe Abbildung 2.26), wie im Beispiel oben die Bodenfläche des Hauses, könnte man also als Kettenkomplexe der Form

$$\mathcal{C} : \ldots \longrightarrow 0 \xrightarrow{\partial_3} C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0 \longrightarrow \ldots$$

mit $C_2 = \mathbb{R}\{F\}$, $C_1 = \mathbb{R}\{a, b, c, d\}$ und $C_0 = \mathbb{R}\{A, B, C, D\}$ interpretieren. Für die Matrizen der Randoperatoren ∂_2 und ∂_1 ergibt sich dann folgendes:

∂_2	F	∂_1	a	b	c	d
a	x	A	x			x
b	x	B	x	x		
c	x	C		x	x	
d	x	D			x	x

Diskussion der Navigationsmöglichkeiten

In diesem Abschnitt soll nun die, in der Spezifikation aufgestellte Behauptung, dass die Navigation zwischen den in der XLink-Topologie topologisch verbundenen Objekten nur in einer Richtung (nämlich von einem Aggregat zu seiner Komponente) durchgeführt werden könne und nicht (unmittelbar) bidirektional möglich sei, diskutiert werden.

Es erschließt sich nicht sofort, warum das so sein sollte und was mit dieser Behauptung überhaupt genau gemeint ist. Deshalb soll hier zunächst einmal erläutert werden, welche Möglichkeiten der Navigation sich aus der Modellierung der Topologie mittels XLink ergeben. Man denke sich hierzu ein beliebiges CityGML-Objekt, dessen Beziehungen zu anderen CityGML-Objekten es herauszufinden gilt. Es sind dann grundsätzlich zwei Fälle denkbar. Entweder die Definition des CityGML-Objekts enthält ein oder mehrere Geometrieobjekte, welche woanders bereits definiert wurden und an dieser Stelle nur referenziert werden, oder es enthält nur Definitionen neuer Geometrieobjekte, welche allerdings eine ID erhalten. Leicht zu erkennen ist hierbei nun die Tatsache, dass die Navigation innerhalb der XLink-Topologie nicht in beiden Richtungen gleich einfach ist.

Betrachtet man den ersten Fall, bei dem in der Definition des CityGML-Objekts ein oder mehrere Geometrieobjekte referenziert werden, dann ist ersichtlich, dass sich in diesem Fall recht leicht die Beziehung zu den anderen CityGML-Objekten bestimmen lässt, in welchen die referenzierten Geometrieobjekte definiert wurden. Man muss einfach nur nach den CityGML-Objekten suchen, in deren Definitionen die IDs der entsprechenden Geometrieobjekte vergeben wurden. Es findet also ein Abgleich der IDs statt. Man kann sich in diesem Fall auch sicher sein, dass diese CityGML-Objekte existieren, da sonst nicht die XLink-Beziehung verwendet werden könnte, sondern die Geometrieobjekte neu definiert werden müssten. Will man im nächsten Schritt aber

zusätzlich wissen, ob diese Geometrieobjekte noch von weiteren CityGML-Objekten referenziert werden, dann kommt man automatisch zur gleichen Konstellation, wie im zweiten oben genannten Fall.

Deswegen nun zur Betrachtung dieser Variante, bei der in der Definition des CityGML-Objekts nur neue Geometrieobjekte definiert werden, welche allerdings eine ID erhalten. Würden keine IDs vergeben werden, dann könnte man sich, zumindest wenn man von Redundanzfreiheit ausgehen kann, sicher sein, dass diese Geometrieobjekte von keinem weiteren CityGML-Objekt genutzt werden und deshalb auch keine topologischen Beziehungen zu einem anderen CityGML-Objekt bestehen. Werden IDs vergeben, dann besteht zumindest die Möglichkeit, dass diese Geometrieobjekte noch von weiteren CityGML-Objekten referenziert werden und somit eine topologische Beziehung zwischen diesen CityGML-Objekten und dem betrachteten CityGML-Objekt besteht. Auch in diesem Fall muss wieder ein ID-Abgleich stattfinden. Es müssen hierbei alle anderen CityGML-Objekte durchsucht werden, wobei man sich dennoch nicht sicher sein kann, ob man fündig wird, da die reine ID-Vergabe noch nicht unbedingt bedeutet, dass diese auch irgendwo referenziert wurden. In diesem Fall ist der Suchaufwand wesentlich höher als im ersten Fall, allerdings lassen sich topologische Beziehungen auch in diesem Fall bestimmen.

Hieran kann man erkennen, dass es grundsätzlich möglich ist alle topologischen Beziehungen zwischen CityGML-Objekten zu finden, weshalb nicht nachvollziehbar ist, warum eine Navigation nur in einer Richtung möglich sein soll.

Allerdings kann es bei der Navigation innerhalb eines CityGML-Objekts zu Schwierigkeiten kommen. Es lässt sich immer zweifelsfrei feststellen, aus welchen Komponenten ein bestimmtes CityGML-Objekt besteht. Die Navigation vom Aggregat zu seiner Komponente beziehungsweise vom CityGML-Objekt zu einem der es definierenden Geometrieobjekte ist also immer möglich. Umgekehrt kann es allerdings zu Problemen kommen. Betrachtet man ein bestimmtes Geometrieobjekt und möchte wissen, zu welchem CityGML-Objekt dieses gehört, dann ist die Lösung nicht immer eindeutig, da ein Geometrieobjekt von mehreren CityGML-Objekten genutzt werden kann. In der Richtung von der Komponente zum zugehörigen Aggregat ist die Navigation also problematisch, weil das Ergebnis nicht eindeutig sein muss. Das ist wohl der Sachverhalt auf

den die Bemerkung in der Spezifikation, ein Nachteil der XLink-Topologie sei es, dass die Navigation zwischen den topologisch verbundenen Objekten nur in einer Richtung (von einem Aggregat, zu seiner Komponente) durchgeführt werden könne und nicht (unmittelbar) bidirektional möglich sei, hinweisen will.

Es ist allerdings so, dass die Navigation von der Komponente zum Aggregat nicht in allen Fällen völlig unmöglich ist, da eben nicht alle Geometrieobjekte mehrmals genutzt werden. Außerdem ist auch die Uneindeutigkeit nicht zwingend problematisch. Findet man zu einem Geometrieobjekt mehrere CityGML-Objekte, dann weiß man, dass diese CityGML-Objekte alle dieses Geometrieobjekt referenzieren und somit weiß man auch, dass zwischen diesen CityGML-Objekten eine topologische Beziehung besteht und das ist eigentlich genau das, was mit Hilfe der XLink-Topologie ermöglicht werden soll.

2.3 Topologische Konsistenz

In diesem Abschnitt wird die in dieser Arbeit verwendete Definition der *topologischen Konsistenz* eingeführt. Topologische Konsistenz ist vor allem wichtig, um teure geometrische Operationen bei topologischen Abfragen zu vermeiden.

Räumliche geometrische Modelle können nur endlich sein. Dazu wird die Geometrie K in Teilstücke unterteilt (häufig sind dies Zellen) und dann wird jedes Teilstück als Element einer endlichen Menge X betrachtet. Um die Randbeziehung zwischen den Elementen zu modellieren, wird auf der endlichen Menge X eine Topologie definiert, die diese Randbeziehung repräsentiert. Es stellt sich heraus, dass X eine partiell geordnete Menge ist. Dies ermöglicht die alleinige Verwendung der endlichen Menge X für topologische Abfragen auf K, ohne auf die Geometrie in K zurückgreifen zu müssen.

Abbildung 2.27 veranschaulicht diesen Prozess, bei dem eine Konfiguration $\mathcal{K}$ aus zwei Polygonen eine Aufteilung der Geometrie K (links) in Punkte, Linien und Flächen bildet (des weiteren als *Teilstücke* bezeichnet). Die endliche partiell geordnete Menge $X = X(\mathcal{K})$ ist in Abbildung 2.27 (rechts) dargestellt, in der die einzelnen Flächen nun als Elemente ohne weitere Struktur für sich betrachtet werden. Die Struktur ergibt sich aus der Teilordnung, die als „begrenzt durch“ (*bounded by*) bezeichnet wird. Hierbei wird ein Element $f \in X$ durch eine anderes Element $f' \in X$ „begrenzt“, wenn das Teilstück F' aus K,

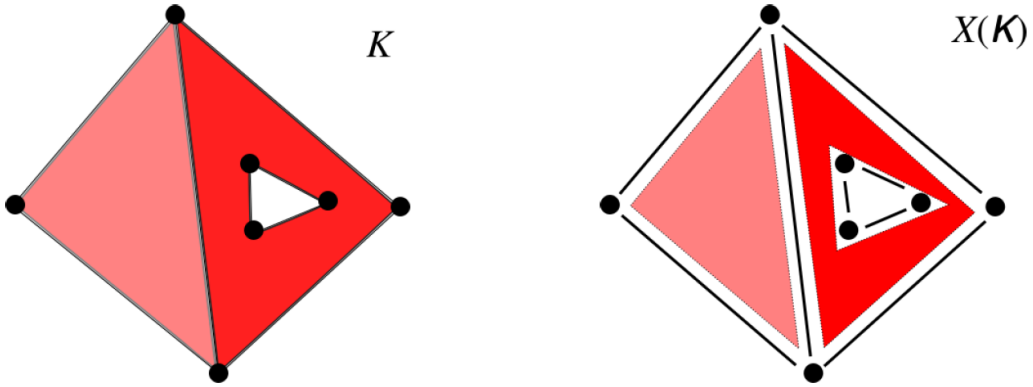


Abbildung 2.27: Eine topologisch konsistente Konfiguration, die als unterteilter Raum K (links) und dessen partiell geordnete Menge $X(\mathcal{K})$ (rechts) dargestellt ist

das f' in $X(\mathcal{K})$ entspricht, Teil des Randes des Teilstücks F aus K ist, das f in $X(\mathcal{K})$ entspricht. Konsistenzprobleme treten dann auf, wenn eine Geometrie K durch eine Konfiguration mehrerer geometrischer Objekte gegeben ist, da es dann vorkommen kann, dass K keine Partitionierung ist, weil sich Teilstücke überlappen.

Man betrachte nun ein topologisches Modell räumlicher Objekte, die als ein Polytopenkomplex modelliert sind, das heißt als ein Zellkomplex, dessen Zellen Polytope verschiedener Dimensionen sind. Angenommen, alle Knoten haben gegebene Koordinaten. Der Inzidenzgraph stellt dann die Topologie des Modells korrekt dar, wenn und nur wenn der Schnitt zweier unterschiedlicher offener Zellen leer ist. Die Topologie des Inzidenzgraphen ist die einer endlichen, partiell geordneten Menge X, wobei die partielle Ordnung durch die *bounded-by*-Beziehung gegeben ist:

$$x \leq y \;\Leftrightarrow\; y \text{ is bounded by } x$$

Dies ist eine sogenannte T_0-*Topologie*. Es ist bekannt, dass die T_0-Topologien auf einer endlichen Menge in einer Eins-zu-eins-Beziehung mit den partiellen Ordnungen auf dieser Menge stehen (Alexandrov, 1937). Nun kann man topologische Konsistenz wie folgt definieren:

Definition 2.3.1 (topologisch konsistent)**.** Das Modell sei *topologisch konsistent*, wenn für alle Paare von geschlossenen Zellen A und B gilt, dass der

Schnitt eines Randobjekts von A mit einem Randobjekt von B ein gemeinsames Randobjekt von A und B ist oder wenn $A = B$ gilt.

Diese Definition ist äquivalent zur Definition topologischer Konsistenz in Giovanella et al. (2019), wo ein Modell als topologisch konsistent definiert ist, wenn für je zwei atomare Objekte aus dem Modell gilt: entweder sind sie disjunkt oder gleich. Das heißt also es gilt

$$a \cap b = \emptyset \text{ oder } a = b$$

für alle $a, b \in X(\mathcal{K})$.

Diese Beschreibung erfüllt auch den ISO 19107 Standard (iso, 2019), wo der Begriff des geometrischen Komplexes definiert ist als eine Menge disjunkter geometrischer Primitive, wobei der Rand jedes geometrischen Primitives als Vereinigung anderer geometrischer Primitive kleinerer Dimension innerhalb derselben Menge dargestellt werden kann („set of disjoint geometric primitives where the boundary of each geometric primitive can be represented as the union of other geometric primitives of smaller dimension within the same set“).

Die Definition der topologischen Konsistenz in dieser Form hier erweitert die Definition von Bradley (2015) und unterscheidet sich von der von Jahn et al. (2017). Man beachte, dass diese Definition der topologischen Konsistenz hier auch auf die Situation angewendet werden kann, in der die „Zellen“ des Komplexes polytopisch geformte Löcher haben dürfen. In diesem Fall ist es die Topologie des Inzidenzgraphen, die vom Modell genau dann repräsentiert wird, wenn es topologisch konsistent ist. Man beachte außerdem, dass das Modell aus einem einzelnen, wenigen oder vielen Objekten bestehen kann, die ein oder mehrere Gebäude bilden können. Es wird in der weiteren Arbeit gezeigt, dass die Einhaltung des CityGML-Standards bei der Modellierung nicht unbedingt bedeutet, dass der ISO-Standard eingehalten wird und dass das Modell topologisch konsistent ist.

Die Philosophie hinter dieser Definition der topologischen Konsistenz ist, dass der aus den Randbeziehungen stammende Inzidenzgraph die „gewünschte“ Topologie modelliert, wohingegen die tatsächlichen geometrischen Einbettungen der Grundelemente, welche durch ihre Eckkoordinaten definiert werden, eine Topologie erzeugen, die sich von der Inzidenzgraph-Topologie unterscheiden könnte.

Diese Idee wird von der „geometrischen Realisierung“ des ISO 19107 Standards abgedeckt, welche als geometrischer Komplex definiert ist, dessen geometrischen Primitive in einer Ein-zu-eins-Entsprechung zu den topologischen Primitiven eines topologischen Komplexes stehen, so dass die Randbeziehungen in den beiden Komplexen übereinstimmen. („geometric complex whose geometric primitives are in a 1-to-1 correspondence to the topological primitives of a topological complex, such that the boundary relations in the two complexes agree“). Das ist auch gemeint, wenn es heißt, dass die aus der Geometrie stammende Topologie mit der Topologie des Inzidenzgraphen übereinstimmen soll.

Eine Analyse der CityGML-Daten zeigte jedoch, dass die Einhaltung des ISO-Standards nicht unbedingt notwendig ist, um dem CityGML-Standard konform zu modellieren. Somit kann es zu im Sinne dieser Arbeit topologisch inkonsistenten Modellen kommen. Man beachte auch, dass in CityGML topologisch inkonsistente Modelle mit oder ohne Verwendung von *XLink* möglich sind. (siehe Abschnitt 2.2).

Kapitel 3

Methodik

In diesem Kapitel werden der grundsätzliche Arbeitsablauf dieser Arbeit und die Umsetzung in eine Implementierung sowie die dafür verwendeten Tools beschrieben.

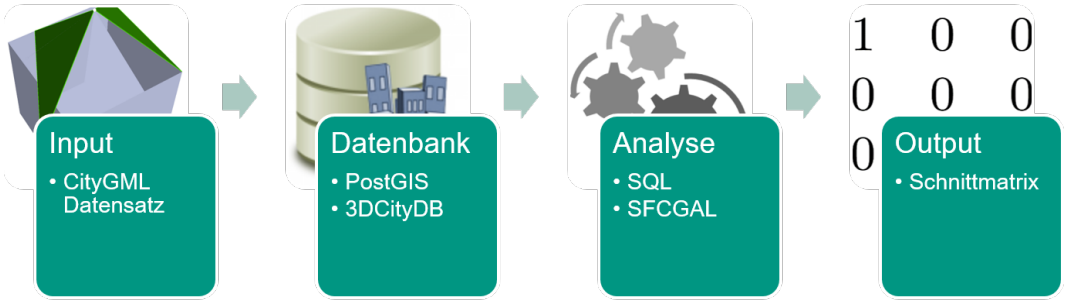


Abbildung 3.1: Workflow dieser Arbeit

Abbildung 3.1 zeigt den Workflow der Arbeit. Zunächst werden die vorhandenen CityGML-Datensätze in eine PostGIS-Datenbank importiert. Dabei wurde ein *3D City Database* (3DCityDB) Schema verwendet, welches in Abschnitt 3.2 genauer erläutert wird. Im nächsten Schritt folgt die Analyse der Daten mit Hilfe von SQL und SFCGAL. Die Implementierung dieser Analyse wird in Abschnitt 3.3 beschrieben. Als Ergebnis erhält man die Schnittmatrix, welche im Abschnitt 3.1 eingeführt wird.

3.1 Schnittmatrix

In diesem Abschnitt wird die Schnittmatrix vorgestellt. Nach der Definition der Schnittmatrix wird zunächst auf die diagonalen Schnittmatrizen eingegangen. Es folgt ein Abschnitt über die Schnittmatrizen für den Schnittgeometrietyp *Punkt* und abschließend werden die Schnittmatrizen für beliebige Polygonpaare betrachtet.

3.1.1 Definition

Basierend auf der Definition der topologischen Konsistenz in Abschnitt 2.3 wird klar, dass es notwendig ist, jedes Polygon P mit jedem anderen Polygon P' zu schneiden, um die topologische Konsistenz zu überprüfen. Zu diesem Zweck wurde eine Schnittmatrix der folgenden Form definiert:

Sei $\mathcal{P} = \{P_1, \ldots, P_n\}$ ein Satz primitiver Typen (zum Beispiel Punkte, Kanten, Flächen, ...) und sei G, H ein Paar von Geometrien gebildet aus den Untermengen $\mathcal{P}_G, \mathcal{P}_H$ von $\mathcal{P}$. Die asymmetrische Schnittmatrix ist dann durch die Abbildung

$$\iota_{G,H} \colon \mathcal{P}_G \times \mathcal{P}_H \to \{0,1\},\ (P, P') \mapsto \begin{cases} 1, & P \cap P' \neq \emptyset \\ 0, & \text{sonst} \end{cases}$$

gegeben. Mit $P \cap P' \neq \emptyset$ ist gemeint, dass ein Objekt O in der Geometrie G vom primitiven Typ P und ein Objekt O' in der Geometrie H vom primitiven Typ P' vorhanden ist, so dass

$$O \cap O' \neq \emptyset$$

gilt. Im Fall $\mathcal{P}_G = \mathcal{P}_H$ wurde festgestellt, dass die Schnittmatrix symmetrisch ist, und sie wird anstelle von $\iota_{G,H}$ als

$$I_{G,H}$$

geschrieben. Der Grund wird später ersichtlich.

In dieser Arbeit interessiert uns hauptsächlich die (symmetrische) Schnittmatrix $I_{\mathcal{P},\mathcal{P}'}$, wobei $\mathcal{P}$ beziehungsweise $\mathcal{P}'$ die Konfiguration der Flächen, Kanten und Knoten von Polygonen P beziehungsweise P' ist. Zu beachten ist, dass man schreiben kann

$$\delta(\mathcal{P}) = \delta(\mathcal{P}') = \{V, E, F\}$$

wobei $V = 0$ für Knoten (*vertex*), $E = 1$ für Kante (*edge*) und $F = 2$ für Fläche (*face*) steht. Dies bedeutet, dass $I_{\mathcal{P},\mathcal{P}'}$ eine Tabelle der Form

	V	E	S
V	a	x	y
E	x	b	z
S	y	z	c

mit $a, b, c, x, y, z \in \{0, 1\}$ ist. Ein Eintrag $I(O, O') = 1$ bedeutet, dass es einen nicht leeren Schnitt zwischen den jeweiligen geometrischen Objekten gibt. Man beachte, dass diese Schnittmatrix nicht mit Egenhofers 9-insection-Matrix aus (Egenhofer, 1991) zusammenhängt. Man beachte außerdem, dass es keine Rolle spielt, ob zum Beispiel eine Kante von P einen Knoten von P' enthält und nicht umgekehrt oder die umgekehrte Situation gilt: Beide Konfigurationen haben dieselbe (symmetrische) Schnittmatrix. Für Polygone P, P' wird im Folgenden auch $I_{P,P'}$ statt $I_{\mathcal{P},\mathcal{P}'}$ und I anstelle von $I_{P,P'}$ geschrieben, wenn klar ist, welches Paar von Polygonen betrachtet wird.

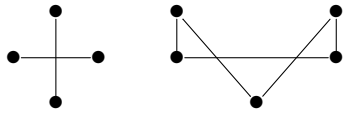


Abbildung 3.2: Zwei topologisch inkonsistente Situationen.

Als Beispiel betrachte man die Situation in Abbildung 3.2. Obwohl das hauptsächliche Interesse auf Polygonen liegt, betrachte man dieses einfache Beispiel bestehend aus Knoten und Kanten (ohne Flächen). Daher sind die Schnittmatrizen 2×2-Matrizen. Beide Konfigurationen von Punkten und Liniensegmenten sind topologisch inkonsistent, da jeweils zwei Linienobjekte vorhanden sind, deren Schnittpunkt ein Punkt ist, der kein Objekt der Konfiguration ist. Die Schnittmatrizen für die Konfigurationen, die aus Liniensegmenten bestehen, sind

$$\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{und} \quad \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

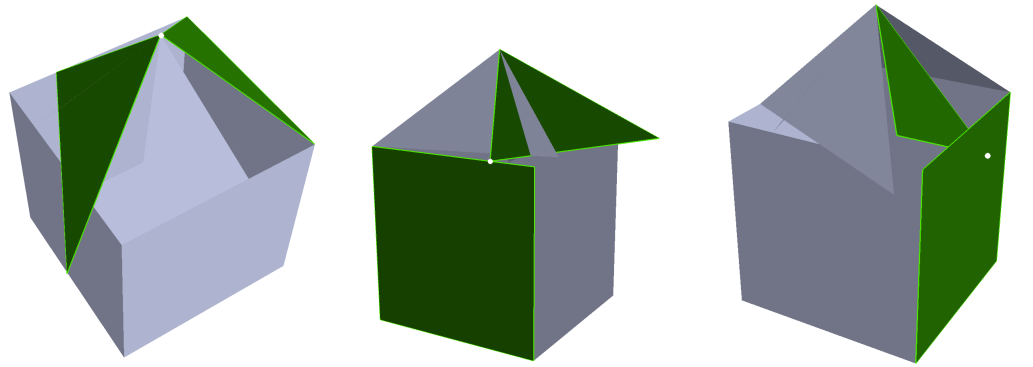
Die Konfiguration rechts in Abbildung 3.2 kann als Rand eines topologisch inkonsistenten Polygons betrachtet werden. Eine andere Art topologisch inkonsistenter Polygone liegt vor, wenn ein Knoten im Inneren einer Kante liegt.

Dann ist die Schnittmatrix der Randkonfiguration

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \quad \text{oder} \quad \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

je nachdem, ob sich zwei Kanten in ihrem Inneren schneiden oder nicht.

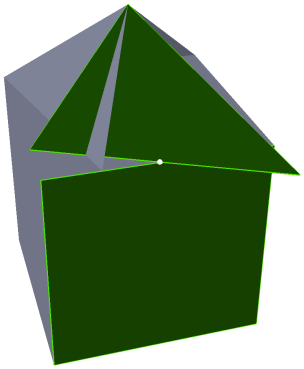
Die Konfiguration links in Abbildung 3.2 kann mit zwei verschiedenen Polygonen in 3D realisiert werden, die sich nur in einem Kantenpaar schneiden, was eine topologisch inkonsistente Konfiguration darstellt.



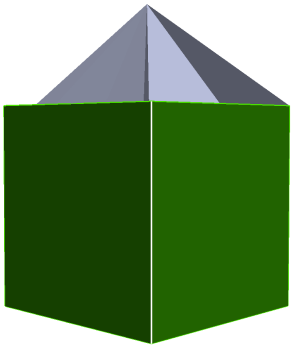
(a) point-point (konsistent)

(b) point-line (inkonsistent)

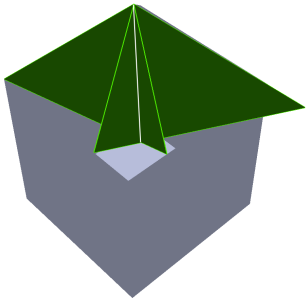
(c) point-area (inkonsistent)



(d) line-line (inkonsistent)



(e) Liniensegment (konsistent)



(f) Liniensegment (inkonsistent)

Abbildung 3.3: Einfaches synthetisches Beispiel eines Hauses mit verschiedenen Arten von topologischen Inkonsistenzen. Die grünen Geometrien zeigen die verschiedenen Arten von Schnittkonstellationen.

Ein Beispiel dafür befindet sich in Abbildung 3.3(d). Die Schnittmatrix ist

dann

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

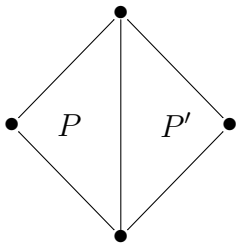


Abbildung 3.4: Eine topologisch konsistente Konfiguration von zwei verschiedenen Dreiecken.

Eine topologisch konsistente Konfiguration von zwei verschiedenen Dreiecken (in diesem Fall mit Flächen) wird in Abbildung 3.4 gezeigt. Die entsprechende 3×3-Schnittmatrix ist

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} = \mathrm{diag}(1, 1, 0)$$

In Abbildung 3.3(e) wird eine dreidimensionale Konstellation zweier Polygone dargestellt, die die gleiche Schnittmatrix haben.

3.1.2 Diagonale Schnittmatrizen

Man wird im Folgenden sehen, dass beim Schneiden zweier topologisch konsistenter planarer Polygone P und P' in 3D sechzig Möglichkeiten bestehen, wie die Schnittmatrix I gefüllt werden kann. Von diesen können genau vier mögliche Konfigurationen topologisch konsistent sein. In diesen Fällen ist I eine Diagonalmatrix:

Satz 3.1.1. *Wenn eine Konfiguration von zwei Randdarstellungsgeometrien topologisch konsistent ist, ist ihre Schnittmatrix eine Diagonalmatrix.*

Beweis. Wenn die Schnittmatrix keine Diagonalmatrix ist, bedeutet dies, dass sich geometrische Objekte mit unterschiedlichen Dimensionen schneiden. Dies

bedeutet jedoch, dass die Schnittgeometrie unmöglich eine Vereinigung von Randobjekten beider Geometrien sein kann, das heißt die Konfiguration ist nicht topologisch konsistent. □

Die Schnittmatrix I als diagonale Matrix bedeutet jedoch umgekehrt nicht, dass die Konfiguration topologisch konsistent ist, wie der folgende Satz 3.1.2 zeigt.

Satz 3.1.2. *Es sei eine Konfiguration von zwei planaren Polygonen P, P' in $\mathbb{R}^3$ gegeben und I sei die Schnittmatrix. Dann gelten folgende Aussagen:*

1. $I \neq \text{diag}(0, 0, 1)$.
2. *Wenn $I = \text{diag}(0, 0, 0)$ oder $I = \text{diag}(1, 0, 0)$, dann ist diese Konfiguration topologisch konsistent.*
3. *Wenn $I = \text{diag}(0, 1, 0)$, $I = \text{diag}(0, 1, 1)$, oder $I = \text{diag}(1, 0, 1)$, dann ist diese Konfiguration topologisch inkonsistent.*
4. *Wenn $I = \text{diag}(1, 1, 0)$ oder $I = \text{diag}(1, 1, 1)$ und P, P' liegen in der gleichen Ebene, dann ist diese Konfiguration topologisch konsistent. Im letzteren Fall folgt $P = P'$.*
5. *Wenn $I = \text{diag}(1, 1, 1)$ und P, P' liegen nicht in derselben Ebene, dann ist diese Konstellation topologisch inkonsistent.*
6. *Wenn $I = \text{diag}(1, 1, 0)$ und P, P' liegen nicht in derselben Ebene, dann ist diese Konstellation nicht eindeutig topologisch konsistent oder inkonsistent.*

Beweis. 1. P und P' sind abgeschlossene Teilmengen von $\mathbb{R}^3$. Daher ist auch deren Schnitt abgeschlossen. Wenn jedoch $I = \text{diag}(0, 0, 1)$ ist, dann entspricht dieser Schnitt dem nicht leeren Schnitt der Innenräume von P und P', welcher im verbundenen Raum $P \cup P'$ offen ist. Die einzigen Untermengen von $P \cup P'$, die abgeschlossen und offen sind, sind die leere Menge und der gesamte Raum $P \cup P'$. Das kann nicht sein.

2. Wenn $I = \text{diag}(0, 0, 0)$ ist, ist der Schnitt leer. Daher ist die Konfiguration topologisch konsistent. Wenn $I = \text{diag}(1, 0, 0)$ ist, wird der Schnitt durch

den Schnitt von Knoten von P mit Knoten von P' gegeben. Daher ist dieser Schnitt eine Vereinigung von Randobjekten sowohl von P als auch von P', das heißt die Konfiguration ist topologisch konsistent.

3. Angenommen, $I = \text{diag}(0, 1, 0)$. In diesem Fall haben nur Kantenpaare einen nicht leeren Schnitt, und dies muss ein einzelner Punkt sein, der in ihrem Inneren liegt. Dies sind keine Randelemente von P oder P'. Jetzt angenommen, $I = \text{diag}(0, 1, 1)$. Dieser Fall ist dem Fall $I = \text{diag}(0, 1, 0)$ ähnlich, außer dass sich auch die Innenräume von P und P' schneiden. Auch hier müssen sich alle sich schneidenden Kantenpaare in ihrem Inneren in einzelnen Punkten schneiden. Nehmen wir schließlich an, dass $I = \text{diag}(1, 0, 1)$ ist. In diesem Fall können die Innenräume von P und P' nicht zusammenfallen, da sonst die beiden Randkurven zusammenfallen müssten. Aber dann hat auch eine Kante einen Schnitt mit einem anderen Randobjekt. In allen drei Fällen folgt daraus, dass die Konfiguration topologisch inkonsistent ist.

4. Wenn $I = \text{diag}(1, 1, 0)$ oder $I = \text{diag}(1, 1, 1)$ ist und P, P' in derselben Ebene liegen, dann schneiden Kanten andere Kanten entlang gesamter Kanten oder gar nicht. Dies reicht aus, um zu sehen, dass die Konfiguration topologisch konsistent ist. Im letzteren Fall folgt $P = P'$.

5. Der Fall $I = \text{diag}(1, 1, 1)$ und P, P' liegen nicht in einer Ebene, ist nur möglich, wenn sich P und P' sich entlang einer Kante schneiden, die keine Randkante beider Polygone ist. Diese Konstellation ist topologisch inkonsistent.

6. Für den Fall $I = \text{diag}(1, 1, 0)$ und P, P' liegen nicht in einer Ebene, lassen sich sowohl konsistente als auch inkonsistente Beispielkonstellationen finden. Ein Beispiel für eine konsistente Konstellation in diesem Fall ist in Abbildung 3.3(e) dargestellt. Für eine inkonsistente Konstellation sei P eine ebenes Dreieck und P' ein nichtkonvexes ebenes Polygon in einer Ebene orthogonal zu P, dessen einer Eckpunkt mit einem Eckpunkt A von P übereinstimmt und dessen eine Kante die gegenüberliegende Kante a von P schneidet und ansonsten kein Durchschnitt erfolgt. Daher ist dieser Fall uneindeutig.

□

In Giovanella et al. (2018) wird noch fälschlicherweise davon ausgegangen, dass die Schnittmatrix $\mathrm{diag}(1, 1, 0)$ immer aus einer topologisch konsistenten Konfiguration von dreidimensionalen Polygonpaaren stammt.

Satz 3.1.3. *Alle Diagonalmatrizen außer* $\mathrm{diag}(0, 0, 1)$ *treten als Schnittmatrizen auf. Die beiden mehrdeutigen Matrizen nach Satz 3.1.2 können sowohl mit topologisch konsistenten als auch mit topologisch inkonsistenten Konfigurationen von Polygonpaaren in* $\mathbb{R}^3$ *realisiert werden.*

Beweis. $\mathrm{diag}(0, 0, 0)$ kann natürlich als Schnittmatrix realisiert werden. Es wurden vier diagonale Schnittmatrizen in CityGML realisiert: Realisierungen inkonsistenter Konfigurationen, die in CityGML modelliert wurden mit den Schnittmatrizen $\mathrm{diag}(0, 1, 0)$ und $\mathrm{diag}(1, 1, 1)$, sind in Abbildung 3.3(d) und in Abbildung 3.3(f) dargestellt. Die topologisch konsistente Konfiguration mit der letzteren Schnittmatrix ist durch $P = P'$ gegeben. Topologisch konsistente Konfigurationen mit den Schnittmatrizen $\mathrm{diag}(1, 0, 0)$ und $\mathrm{diag}(1, 1, 0)$ sind in den Abbildungen 3.3(a) und 3.3(e) dargestellt. Um $\mathrm{diag}(1, 1, 0)$ auf topologisch inkonsistente Weise zu realisieren, sei P ein Polygon und P' ein nicht konvexes Polygon mit zwei Knoten, die P jeweils an einem Knoten und einer Linie berühren. Die Matrix $\mathrm{diag}(0, 1, 1)$ kann wie folgt realisiert werden: Seien P und P' Quadrate, die die gleiche Seitenlänge haben und nicht in derselben Ebene liegen. Diese Konfiguration kann so vorgenommen werden, dass der Schnitt ihrer Innenräume ein Liniensegment ohne Endpunkte ist. Diese fehlenden Endpunkte können mit Schnitten der Kanten von P und P' zusammenfallen, indem P' parallel zu P entlang der Richtung eines Paars gegenüberliegender Kanten von P verschoben wird. Die gleiche Methode realisiert auch $\mathrm{diag}(1, 0, 1)$: Die Quadrate P und P' haben zwei gemeinsame Knoten, die sich in P sowie in P' gegenüberliegen. □

3.1.3 Schnittgeometrietyp Punkt

Wenn der Schnitt zweier Polygone ein Punkt ist, können vier verschiedene Schnittmatrizen auftreten. Diese vier Matrizen können die folgenden beschrei-

benden Namen erhalten:

$$\text{point-point} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \text{point-line} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\text{point-area} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad \text{line-line} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Wie oben gezeigt wurde, beschreibt „point-point“ (siehe Abbildung 3.3(a)) eine topologisch konsistente Konstellation, während „point-line“ (siehe Abbildung 3.3(b), „point-area“ (siehe Abbildung 3.3(c)) und „line-line“ (siehe Abbildung 3.3(d)) topologisch inkonsistente Konfigurationen von zwei unterschiedlichen Polygonen beschreiben. Diese vier Schnittkonstellationen sind also in den Abbildungen 3.3(a) bis 3.3(d) dargestellt, wo ein einfaches synthetisches Beispiel eines Hauses mit verschiedenen topologischen Inkonsistenzen gezeigt wird. Die Schnittmatrix von Abbildung 3.3(f) lautet $\operatorname{diag}(1, 1, 1)$.

3.1.4 beliebige Polygonpaare

Satz 3.1.4. *Aus den* 2^6 *symmetrischen* 3×3*-Matrizen mit Einträgen in* $\{0, 1\}$ *können genau sechzig als Schnittmatrizen für Paare planarer Polygone in* $\mathbb{R}^3$ *auftreten.*

Beweis. Seien P, P' planare Polygone im $\mathbb{R}^3$. Wir können $I_{\partial P, \partial P'}$ als die 2×2-Schnittmatrix für den Rand definieren, $\iota_{\partial P, \operatorname{int}(P')}$ als eine 2×1-Schnittmatrix und $I_{\operatorname{int}(P), \operatorname{int}(P')}$ als eine 1×1-Schnittmatrix. Jetzt können wir definieren

$$J_{\partial P, \partial P'} = \begin{pmatrix} a & b & 0 \\ b & c & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$J_{\partial P, \operatorname{int}(P')} = \begin{pmatrix} 0 & 0 & a \\ 0 & 0 & b \\ a & b & 0 \end{pmatrix}$$

$$J_{\operatorname{int}(P), \operatorname{int}(P')} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & a \end{pmatrix}$$

mit $a, b, c \in \{0, 1\}$ und

$$I_{\partial P, \partial P'} = \begin{pmatrix} a & b \\ b & c \end{pmatrix}$$
$$\iota_{\partial P, \mathrm{int}(P)} = \begin{pmatrix} a \\ b \end{pmatrix}$$
$$I_{\mathrm{int}(P), \mathrm{int}(P')} = \begin{pmatrix} a \end{pmatrix}$$

Wenn also $I_{P,P'}$ eine gültige Schnittmatrix ist, dann ist

$$I_{P,P'} = J_{\partial P, \partial P'} + J_{\partial P, \mathrm{int}(P')} + J_{\mathrm{int}(P), \mathrm{int}(P')}$$

und die Stützen der drei Matrizen sind paarweise disjunkt.

1. Man beachte zunächst, dass alle Möglichkeiten für jedes von $I_{\partial P, \partial P'}$, $\iota_{\partial P, \mathrm{int}(P')}$, $I_{\mathrm{int}(P), \mathrm{int}(P')}$ durch Paare planarer Polygone in $\mathbb{R}^3$ realisiert werden können, so dass ihre Anzahlen $\#I_{A,B}$ beziehungsweise $\#\iota_{A,B}$ sind:

$$\#I_{\partial P, \partial P'} = 2^3$$
$$\#\iota_{\partial P, \mathrm{int}(P')} = 2^2$$
$$\#I_{\mathrm{int}(P), \mathrm{int}(P')} = 2$$

2. Man beachte, dass jedes $I_{\partial P, \partial P'}$ und jedes $I_{\mathrm{int}\, P, \mathrm{int}(P')}$ gleichzeitig realisiert werden können. Es kann nämlich jedes $I_{\partial P, \partial P'}$ so realisiert werden, dass $\mathrm{int}(P) \cap \mathrm{int}(P') = \emptyset$ ist. Und wenn wir $\mathrm{int}(P) \cap \mathrm{int}(P') \neq \emptyset$ wollen, dann kann jedes $I_{\partial P, \partial P'}$ außer

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

mit P und P' in derselben Ebene realisiert werden. Die letztere Schnittmatrix kann mit P und P' in nicht parallelen Ebenen und mit sich schneidenden Innenräumen realisiert werden.

3. Gleichzeitige Realisierung von $\iota_{\partial P, \mathrm{int}(P)}$ und $I_{\mathrm{int}(P), \mathrm{int}(P')}$. Der Fall

$$\iota_{\partial P, \mathrm{int}(P)'} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

ist der einzige, der besondere Aufmerksamkeit erfordert: In diesem Fall müssen die Randpunkte jeder Kante, die $\mathrm{int}(P')$ schneidet, einen nicht

leeren Schnitt mit dem Rand von P' haben. Dies kann mit $\text{int}(P) \cap \text{int}(P') \neq \emptyset$ in einer gemeinsamen Ebene oder mit $\text{int}(P) \cap \text{int}(P') = \emptyset$ in $\mathbb{R}^3$ realisiert werden. In allen anderen Fällen sind alle möglichen zwei Werte von $I_{\text{int}(P),\text{int}(P')}$ realisierbar.

4. Gleichzeitige Realisierung von $\iota_{\partial P,\text{int}(P')}$ und $I_{\partial P,\partial P'}$. Wenn

$$\iota_{\partial P,\text{int}(P')} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

dann liegen zwei Knoten von P auf dem Rand von P'. Dies bedeutet, dass die beiden Matrizen

$$\begin{pmatrix} 0 & 0 \\ 0 & * \end{pmatrix}$$

unmöglich sind. Aber alle anderen sechs sind in diesem Fall realisierbar. In allen anderen drei Fällen von $\iota_{\partial P,\text{int}(P')}$ sind alle acht Möglichkeiten für $I_{\partial P,\partial P'}$ realisierbar.

Zusammenfassung. Insgesamt gibt es zwölf Möglichkeiten für den Fall

$$\iota_{\partial P,\text{int}(P')} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

und in allen drei anderen Fällen gibt es $2 \times 8 = 16$ mögliche Realisierungen. Alles in allem gibt es also

$$12 + 3 \times 16 = 60$$

mögliche Schnittmatrizen für planare Polygone im $\mathbb{R}^3$. □

3.1.5 Schnittmatrizen von CityGML-Daten

CityGML bietet viele verschiedene Möglichkeiten zum Definieren von Entitäten in verschiedenen Kontexten. Im Fall von Flächen in $\mathbb{R}^3$ wird jedoch die Geometrie von Polygonen mit Löchern verwendet, indem Listen von Koordinatentripeln (Punkte in $\mathbb{R}^3$) definiert werden, die die Knoten darstellen. Aus diesen Informationen können die Knoten, Kanten und Flächen eines Polygons extrahiert werden, um Schnittmatrizen zu berechnen. Es ist beispielsweise möglich, dieselbe Konfiguration als *CompositeSurface* oder als *MultiSurface* zu

speichern, wobei letztere nicht alle der ersteren Art sein dürfen, wenn die Spezifikationen von CityGML korrekt befolgt werden. Es sei darauf hingewiesen, dass es nicht das Ziel dieser Arbeit ist zu überprüfen, ob die Datensätze den Spezifikationen von CityGML entsprechend modelliert wurden, sondern dass das Hauptinteresse daran besteht, die teilweise geordneten Zellmengen von Polygonen aus den vorliegenden Datensätzen zu extrahieren, um Schnittmatrizen zu berechnen. Somit hängt die Schnittmatrix oder die Tatsache, ob ein Polygonpaar topologisch konsistent ist oder nicht, nicht von seiner (möglicherweise falschen) Darstellung innerhalb einer CityGML-Datei ab.

3.2 3D City Database

Um topologische und geometrische Abfragen durchführen zu können, wurden die CityGML-Daten in ein *3D City Database* (3DCityDB) Schema (Kolbe et al., 2016; 3DCityDB, 2018; Stadler et al., 2009) importiert. Somit konnten Datenbankabfragen zur Analyse genutzt werden, was zum Beispiel durch (räumliche) Indizes verglichen mit dem nur sequenziell möglichen Durchsuchen eines XML-Dokumentes einen großen Vorteil darstellt.

3DCityDB ist ein kostenloses Open-Source-Paket, das aus einem Datenbankschema und einer Reihe von Softwaretools zum Importieren, Verwalten, Analysieren, Visualisieren und Exportieren von virtuellen 3D-Stadtmodellen gemäß dem CityGML-Standard besteht. Das Datenbankschema ergibt sich aus einer Abbildung des objektorientierten Datenmodells von CityGML 2.0 auf die relationale Struktur eines räumlich erweiterten relationalen Datenbankmanagementsystems (*spatially-enhanced relational database management system* (SRDBMS)). Die 3DCityDB unterstützt das kommerzielle SRDBMS Oracle (mit „Spatial“ oder „Locator“ Lizenzoptionen) und das Open Source SRDBMS PostGIS, welches eine Erweiterung des freien RDBMS PostgreSQL ist und für diese Arbeit verwendet wurde. 3DCityDB nutzt die spezifischen Repräsentations- und Verarbeitungsfunktionen des SRDBMS in Bezug auf die Geodatenelemente. Es kann auch sehr große Modelle in mehreren LoDs verarbeiten, die aus Millionen von 3D-Objekten mit Hunderten von Millionen von Geometrien und Texturbildern bestehen.

3DCityDB ist an vielen Orten der Welt in realen Produktionssystemen in

Benutzung und wird auch in einer Reihe von Forschungsprojekten eingesetzt. Man betrachte als Beispiel Chaturvedi et al. (2015). Nach 3DCityDB (2018) und Kolbe et al. (2016) halten und verwalten die Städte Berlin, Potsdam, München, Frankfurt und Zürich ihre virtuellen 3D-Stadtmodelle innerhalb einer Instanz von 3DCityDB. Die Unternehmen virtualcitySYSTEMS und M.O.S.S., die auch Entwicklungspartner sind, verwenden 3DCityDB als Kern ihrer kommerziellen Produkte und Dienstleistungen, um virtuelle 3D-Stadtmodelle zu erstellen, zu verwalten, zu visualisieren, umzuwandeln und zu exportieren. Darüber hinaus lagern und verwalten die Landesvermessungsämter aller 16 Bundesländer die landesweit gesammelten 3D-Gebäudemodelle in CityGML LoD1 und LoD2 mit 3DCityDB (Kolbe et al., 2016).

Das mitgelieferte Import/Export-Softwaretool ermöglicht einen leistungsstarken Import und Export von CityGML-Datensätzen gemäß den CityGML-Versionen 2.0 und 1.0. Das Tool ermöglicht die Verarbeitung von sehr großen Datensätzen, auch wenn die XLinks zwischen CityGML-Features oder XLinks zu dreidimensionalen GML-Geometrieobjekten enthalten (Kunde, 2012; Kunde et al., 2013).

Da 3DCityDB auf CityGML basiert, kann der interoperable Datenzugriff von Benutzeranwendungen auf die Datenbank auf mindestens zwei Arten erreicht werden:

1. durch Nutzung des mitgelieferten CityGML-Import/Export-Tools oder den enthaltenen Basis-WFS (*Web Feature Service*), um die Daten im CityGML-Format auszutauschen und

2. durch direkten Zugriff auf die Datenbanktabellen. Es ist einfach, ein 3D-Stadtmodell durch Hinzufügen von Informationen zu den Datenbanktabellen in einer Benutzeranwendung anzureichern. Der angereicherte Datensatz kann dann ausgetauscht oder archiviert werden, indem das Stadtmodell ohne Informationsverlust nach CityGML exportiert wird. Analog kann 3DCityDB verwendet werden, um ein CityGML-Datensatz zu importieren und anschließend auf das Stadtmodell zuzugreifen und mit diesem zu arbeiten, indem direkt aus einigen Anwendungsprogrammen auf die Datenbanktabellen zugegriffen wird.

3.3 Implementierung

In diesem Abschnitt wird zunächst die Implementierung beschrieben, wie sie im Rahmen dieser Arbeit für die Untersuchung der CityGML-Datensätze genutzt wurde und anschließend wird noch kurz auf eine möglich andere Lösung eingegangen.

Die Implementierung verwendet SFCGAL-Funktionen (SFCGAL, 2018). SFCGAL ist ein Wrapper um die *Computational Geometry Algorithms Library* (CGAL, 2018), die zwei- und dreidimensionale Operationen an OGC-Standardmodellen (Simple Feature Access, CityGML, ...) implementieren will. Mit Hilfe der C-API von SFCGAL stellt PostGIS einige Funktionen von SFCGAL in räumlichen Datenbanken zur Verfügung und kann um weitere Funktionen gepatcht werden. Diese SFCGAL-Funktionen können somit auch in einer 3DCityDB-Datenbank genutzt werden.

Der erste Teil der Schnittanalyse wurde direkt in der Datenbank unter Verwendung von SQL-Abfragen und somit unter Ausnutzung der Vorteile räumlicher Indizes durchgeführt. Um dies zu erreichen, wurde die *Procedural Language/PostgreSQL Structured Query Language* (PL/pgSQL) (Eisentraut, 2003) verwendet, um eine Funktion zu schreiben. PL/pgSQL wurde eingeführt, um die SQL-Fähigkeiten von PostgreSQL zu erweitern. PL/pgSQL-Code kann als *Stored Procedure* in der Datenbank selbst gespeichert werden. Es unterstützt Variablen, Bedingungen, Schleifen, Funktionen, Datenbankcursor und Ausnahmebehandlung. PL/pgSQL-Code kann sowohl von SQL-Befehlen als auch von Datenbank-Triggern aufgerufen werden.

Für jedes sich schneidende Polygonpaar wurde die Schnittgeometrie berechnet und der Geometrietyp der Schnittgeometrie ermittelt und die Ergebnisse in eine neu erstellte Tabelle in der Datenbank geschrieben. Zu diesem Zweck wurden entsprechende SFCGAL-Funktionen verwendet, die von PostGIS bereitgestellt werden. PostGIS zielt darauf ab, die SQL-Option des OGC Simple Features Access-Standards (Herring, 2010) zu unterstützen. Zuvor wurden alle Polygone auf ihre Validität überprüft, das heißt sie wurden auf Planarität und Selbstüberschneidung getestet und die Position und Orientierung der möglichen inneren Ringe wurden überprüft. Für die Validitätsprüfung wurde die SFCGAL-Funktion *isValid3d* verwendet, die zu PostGIS gepatcht werden

musste, da die von PostGIS bereitgestellte Funktion *st_isValid* nur zweidimensionale Geometrien verarbeiten kann.

Ein ungelöstes Problem mit diesem ersten Teil der Implementierung war die von PostGIS bereitgestellte SFCGAL-Funktion *st_3dintersection*, die bei bestimmten Polygonkonfigurationen zu einer Unterbrechung der Datenbankverbindung führte, aufgrund eines Absturzes des verantwortlichen Codes der Funktion. Dies führte dazu, dass einige meist größere Datensätze nicht vollständig verarbeitet werden konnten. Die einzige Möglichkeit, diese Abstürze zu umgehen, bestand darin, die entsprechenden Polygone händisch aus der Datenbank zu löschen.

Ein weiteres Problem, das ebenfalls zu Abstürzen führte, trat bei zwei synthetischen Datensätzen mit „topologischen Fehlern“ auf. Das Problem betraf die PostGIS-Funktion *st_3dintersects*, die zuerst prüft, ob sich zwei Polygone schneiden, bevor die eigentliche Schnittanalyse gestartet wird. Dieses Problem konnte bisher nicht gelöst werden, weshalb die betroffenen Datensätze nicht verwendet werden konnten.

Die Schnittmatrixoperatoren wurden dann direkt in C++ innerhalb des SFCGAL-Frameworks implementiert, da die von PostGIS bereitgestellten SFCGAL-Funktionen nicht ausreichten, um die notwendigen Abfragen direkt in der Datenbank durchzuführen. Zu diesem Zweck wurden die Polygonpaare, deren Schnittgeometrietyp *Point* ist oder ein Liniensegment (das heißt ein *LineString*, das nur aus zwei Punkten besteht) zuerst aus der Datenbank exportiert und dann von einer C++-Funktion weiterverarbeitet. Derzeit werden nur diese beiden Arten von Schnittgeometrien berücksichtigt, da sie am häufigsten in den verwendeten CityGML-Datensätzen vorkommen und es für sie recht einfach ist, die Schnittkonstellation zu bestimmen. Wenn die Schnittgeometrie ein Punkt ist, dann gibt es, wie in Abschnitt 3.1 beschrieben, vier mögliche Schnittmatrizen, von denen genau eine von einer topologisch konsistenten Konfiguration stammt. Um die Schnittmatrix zu bestimmen, wurde zuerst überprüft, ob der Schnittpunkt gleich einem der Knoten eines oder beider der geschnittenen Polygone ist. Wenn ein übereinstimmender Knoten auf beiden Polygonen gefunden wurde, bedeutet dies, dass sich beide Polygone an diesem Punkt schneiden. Für diese Konfiguration entspricht die Schnittmatrix „point-point“. Dieser Fall ist topologisch konsistent. Wenn auf keinem der bei-

den Polygone ein übereinstimmender Knoten gefunden wurde, entspricht die Schnittmatrix „line-line“, da dies nur möglich ist, wenn sich zwei Kanten der Polygone schneiden. Wenn der Schnittpunkt mit einem Knoten eines der beiden geschnittenen Polygone identisch war, wurde weiter geprüft, ob er auf einer Kante oder innerhalb des anderen Polygons lag. Liegt der Schnittpunkt auf einer Kante des anderen Polygons, so entspricht die Schnittmatrix „point-line“ und wenn er innerhalb des anderen Polygons liegt, entspricht die Schnittmatrix „point-area“. Die den jeweiligen Begriffen zugeordneten Schnittmatrizen können dem Abschnitt 3.1.3 entnommen werden.

Für den Schnittgeometrietyp *LineString* wurde nur zwischen konsistent und inkonsistent unterschieden, da es sehr aufwändig wäre, die exakten Schnittmatrizen für alle möglichen Konfigurationen zu bestimmen. Tatsächlich wurde die Menge aller möglichen Konfigurationen von zwei verschiedenen Polygonen für eine gegebene Schnittmatrix noch nicht gefunden, außer in dem Fall, wenn der Schnitt ein Punkt ist. Um zwischen konsistenten und inkonsistenten Schnittkonstellationen eines Liniensegments zu unterscheiden, ist es ausreichend, zu überprüfen, ob beide Polygone die Schnittgeometrie enthalten, das heißt ob das Liniensegment mit einer Kante beider Polygone identisch ist. Wenn dies der Fall ist, ist die Konfiguration topologisch konsistent, ansonsten inkonsistent.

Die statistische Auswertung der Ergebnisse wurde ebenfalls in dem geschriebenen C++-Programm durchgeführt.

Eine weitere Möglichkeit der Implementierung ist das Patchen der fehlenden SFGCAL-Funktionen in PostGIS. Dann kann für den zuvor in C++ ausgeführten Teil der Analyse eine weitere PL/pgSQL-Funktion geschrieben werden, wobei der Programmablauf dem für C++ beschriebenen Ablauf entspricht. Der Vorteil dieser Vorgehensweise ist, dass somit die komplette Analyse „in einem Rutsch“ stattfinden kann und außerdem die Ergebnisse direkt wieder in der Datenbank gespeichert und weiterverarbeitet werden können. Da bereits alle Datensätze mit der ersten beschriebenen Implementierung prozessiert waren, als die Umsetzung für den kompletten Ablauf in der Datenbank gelang, wurde diese Lösung nur noch stichprobenartig für wenige Datensätze durchgeführt, was erwartungsgemäß zu den gleichen Ergebnissen führte, weshalb nicht alle Datensätze nochmals prozessiert wurden. Für mögliche zukünftige Analysen und Anwendungen, kann es jedoch sehr nützlich sein, dass die zweite Imple-

mentierung dennoch noch umgesetzt wurde.

Kapitel 4

Untersuchungen

In diesem Kapitel werden zunächst in Abschnitt 4.1 die verwendeten Datensätze eingeführt und anschließend in Abschnitt 4.2 die Ergebnisse vorgestellt und erläutert.

4.1 Datensätze

Für diese Arbeit wurden elf „Realwelt“-Datensätze, fünf synthetische Datensätze, sowie ein fiktiver Datensatz verwendet. Eine Übersicht über die verwendeten Datensätze ist in Tabelle 4.1 zu sehen.

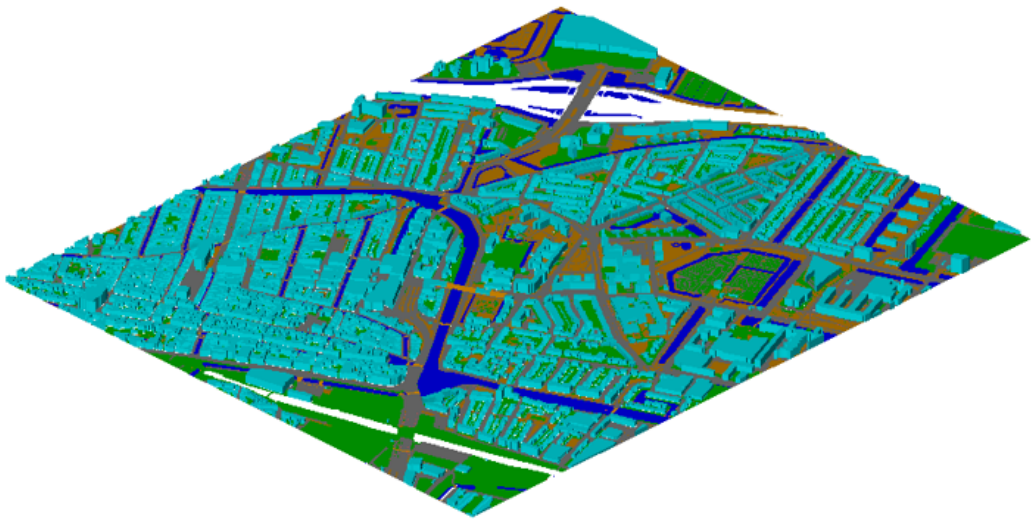


Abbildung 4.1: CityGML-Modell der Stadt Delft

Der größte Datensatz enthält die gesamte Stadt Delft in LoD1 (siehe Abbildung 4.1). Er wurde von (TUDelft 3D geoinformation, 2018) heruntergeladen. Dort gibt es einige 3D-Stadtmodelle niederländischer Städte zum Download, welche alle unter der *Creative Commons CC0 Public Domain Dedication*-Lizenz verfügbar sind. Die Datensätze wurden mit *3dfier* (Ledoux et al., 2021) automatisiert konstruiert. Dieses Tool nimmt geographische 2D-Datensätze (zum Beispiel topografische Datensätze), die aus Polygonen bestehen, und "3dfiziert" sie, was soviel heißt wie "dreidimensional machen". Die Höhe wird aus einem luftgetragenen Punktwolken-Datensatz gewonnen und die Semantik der Polygone wird verwendet, um das Heben in die dritte Dimension so durchzuführen, dass es realistisch ist. Für den vorliegenden Datensatz der Stadt Delft wurden der BGT-Datensatz (großer topografischer Datensatz der Niederlande) und die AHN3-Datensätze (Laseraltimetrie) kombiniert.

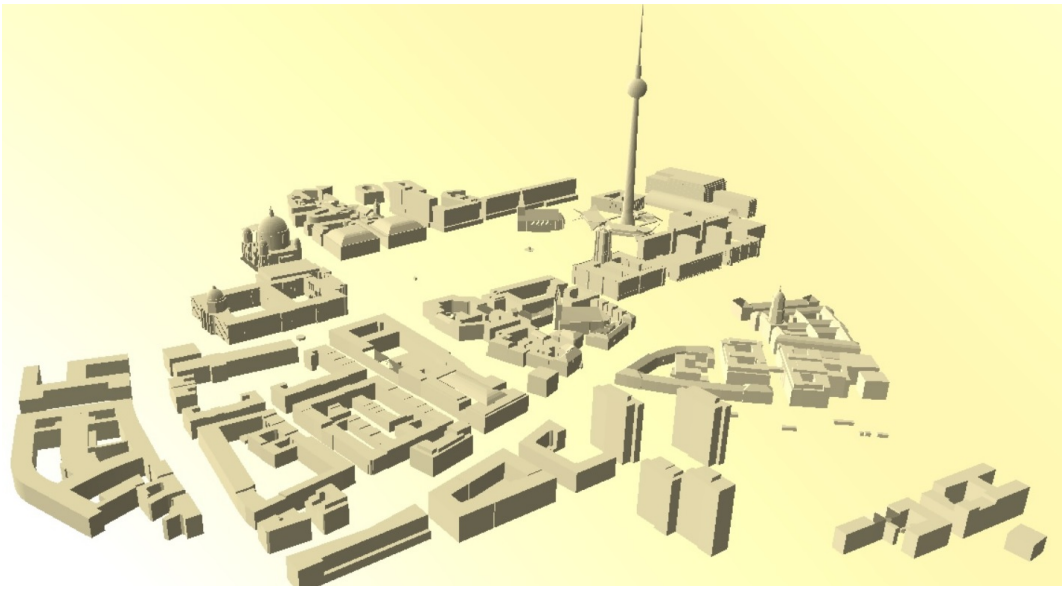


Abbildung 4.2: CityGML-Modell des Alexanderplatzes in Berlin

Zwei der Datensätze enthalten kleine Teile von Berlin in LoD1 und LoD2. Dies sind die Datensätze „Alexanderplatz“ (siehe Abbildung 4.2) und „Pariser Platz“ (siehe Abbildung 1.1). Dies ist der Platz, an dem sich das Brandenburger Tor befindet. Beide Datensätze sowie das ebenfalls verfügbare CityGML-Modell von ganz Berlin wurden aus extrudierten Katasterdaten generiert. Alle verfügbaren CityGML-Datensätze der Stadt Berlin können von (Berlin Business Location Center, 2018) heruntergeladen werden.

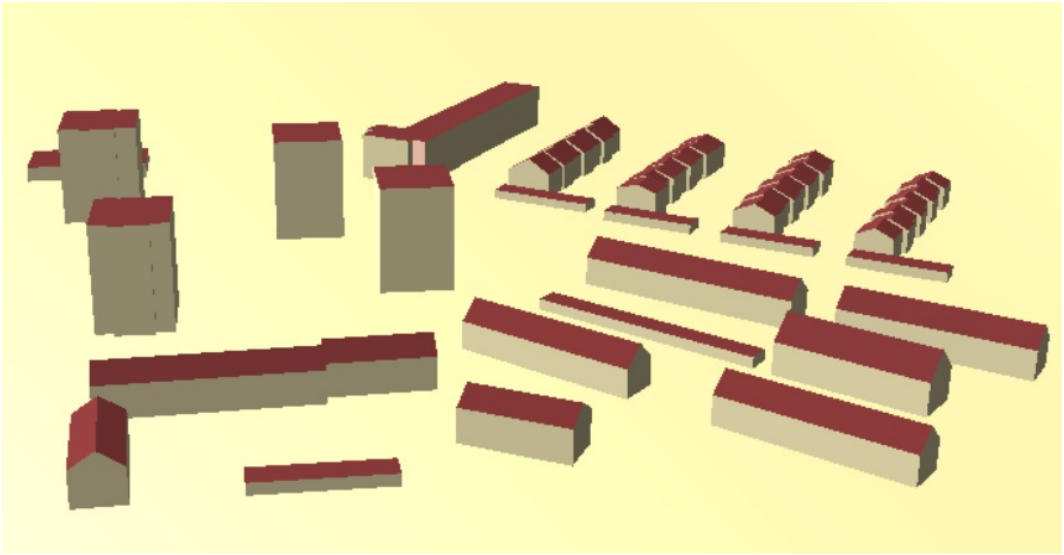


Abbildung 4.3: CityGML-Modell Lindenallee und Kranichweg in Karlsruhe

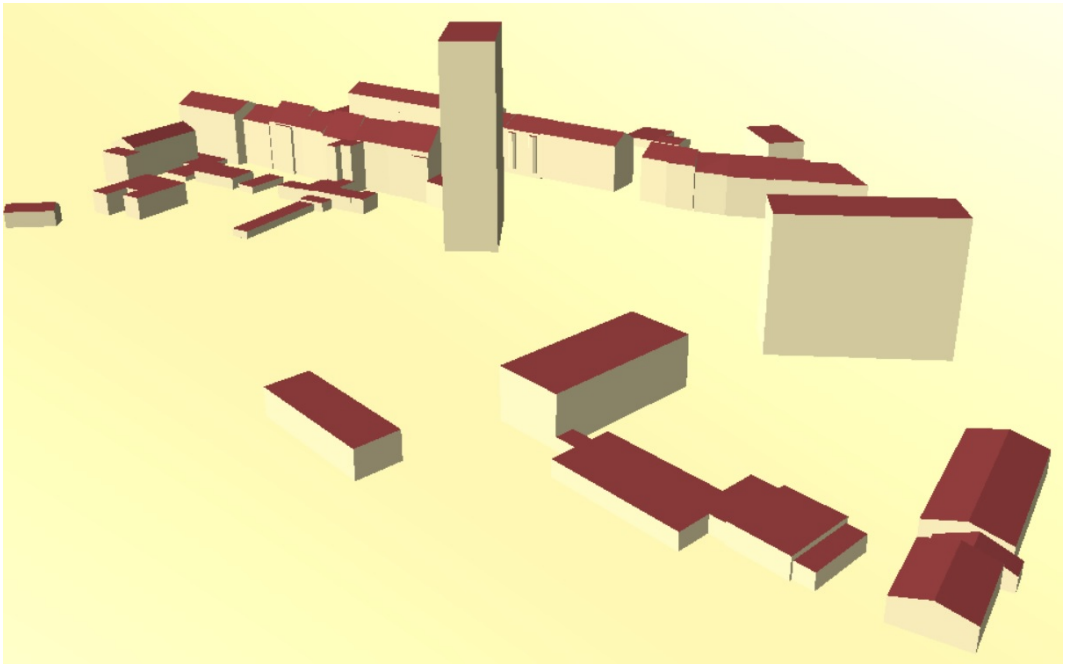


Abbildung 4.4: CityGML-Modell Rheinstraße in Karlsruhe

Außerdem wurden fünf Datensätze aus Karlsruhe untersucht. Dabei handelt es sich um die Datensätze „Lindenallee und Kranichweg“ (siehe Abbildung 4.3), „Rheinstrasse“ (siehe Abbildung 4.4), „Rintheim“ (siehe Abbildung 4.5), „Tennesseeallee“ (siehe Abbildung 4.6) und „Weidenweg“ (siehe Abbildung 4.7). Diese Datensätze stammen vom Liegenschaftsamt der Stadt Karlsruhe.

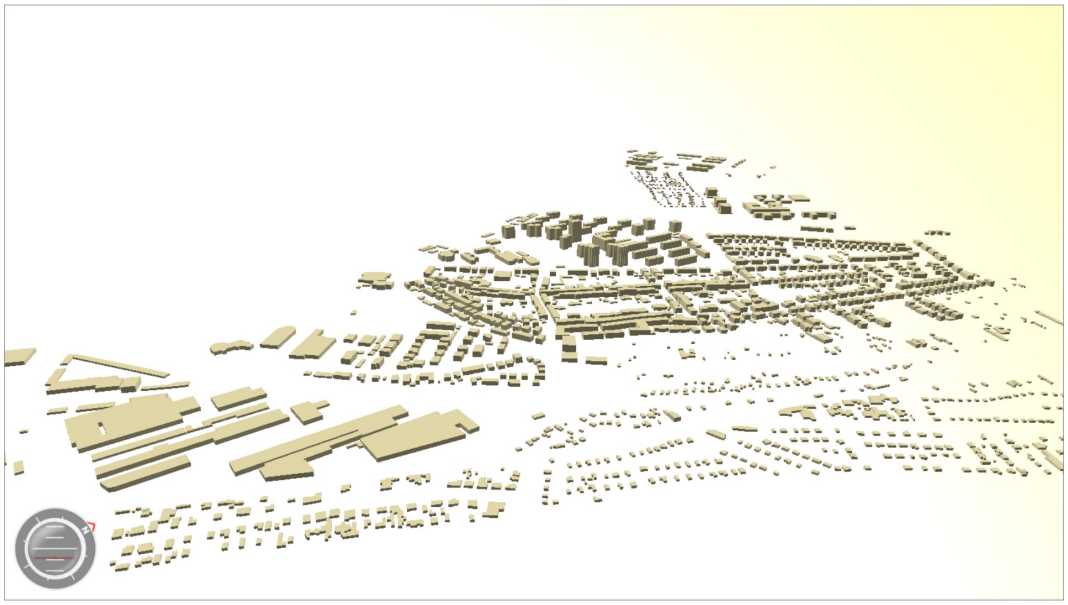


Abbildung 4.5: CityGML-Modell Rintheim, Stadtteil von Karlsruhe

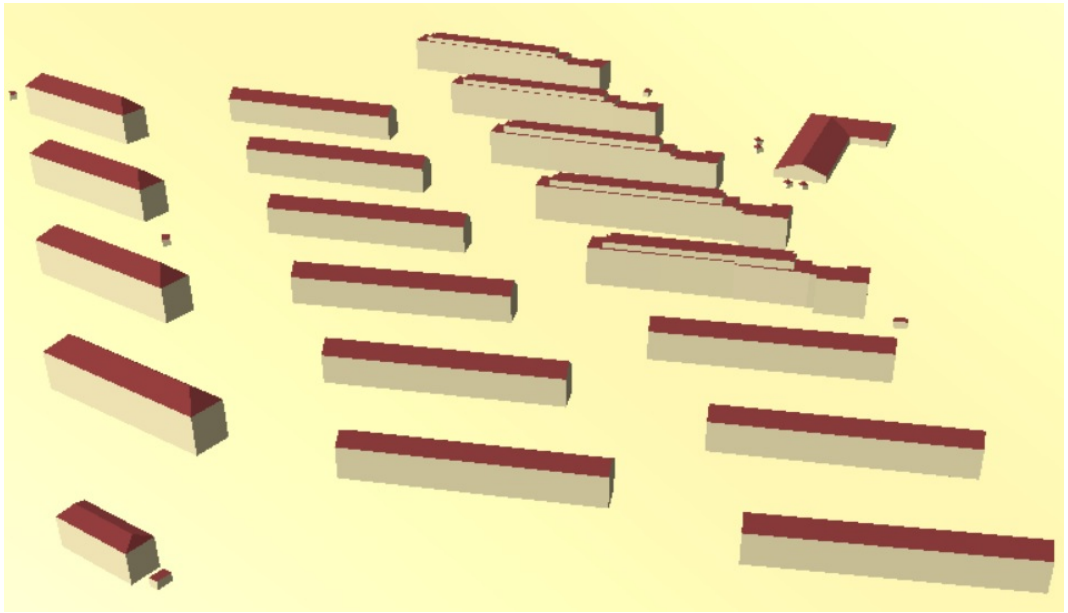


Abbildung 4.6: CityGML-Modell Tennesseeallee in Karlsruhe

Sie enthalten einzelne Straßen oder kleine Wohngebiete der Stadt Karlsruhe, die aus LIDAR-Daten generiert und in LoD2 modelliert wurden. Weitere Informationen zum 3D-Stadtmodell der Stadt Karlsruhe findet man in Hauenstein (2017) und auf der Internetseite der GeodatenInfrastruktur Karlsruhe (GDI-KA) (GeodatenInfrastruktur Karlsruhe (GDI-KA), 2021).

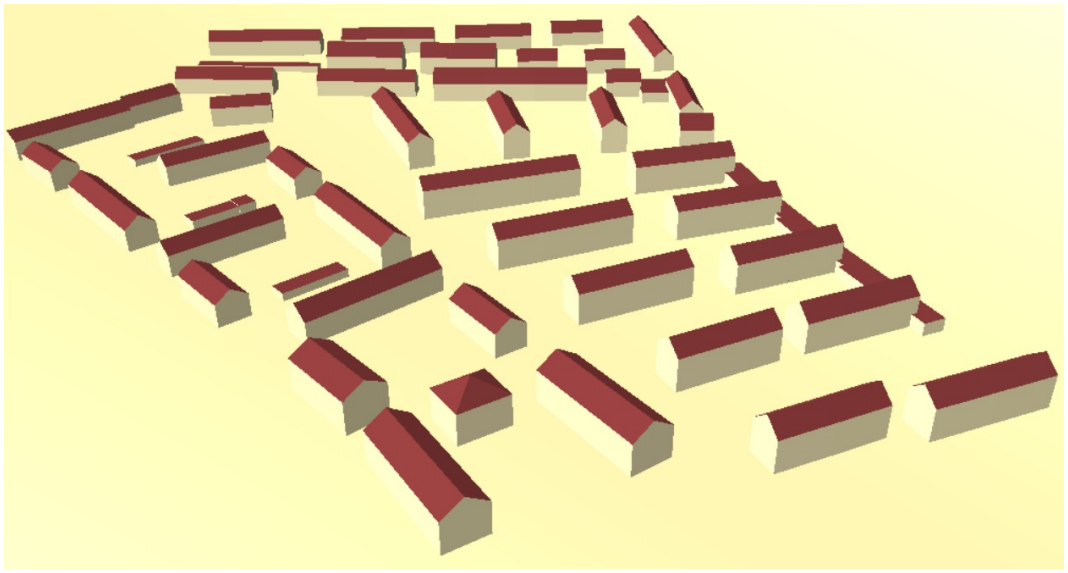


Abbildung 4.7: CityGML-Modell Tennesseeallee in Karlsruhe

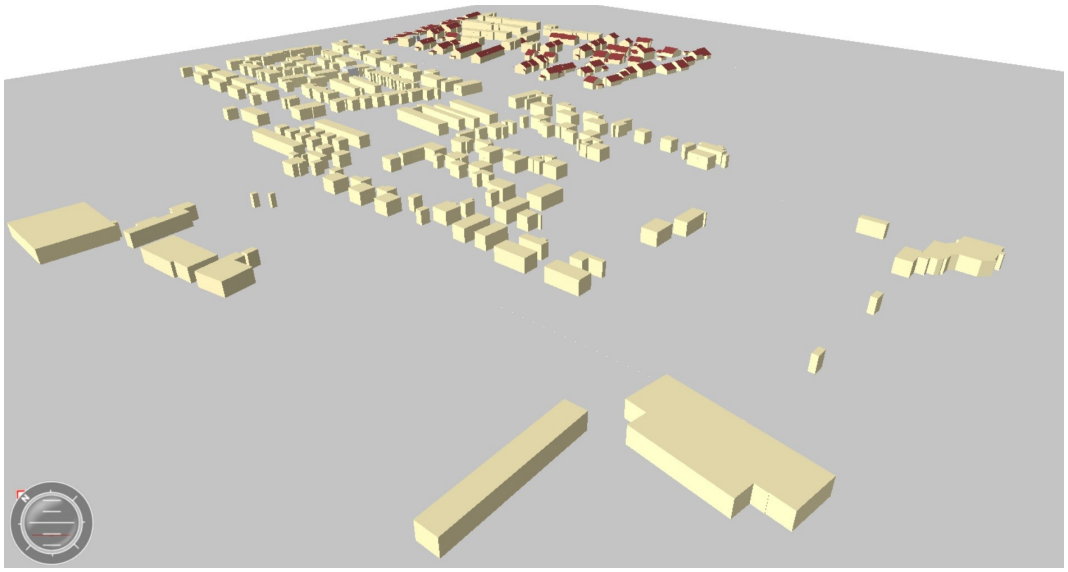


Abbildung 4.8: CityGML-Modell des Dorfes Waldbrücke, Gemeinde Weingarten bei Karlsruhe

Drei der Datensätze wurden von der CityGML-Homepage heruntergeladen. Dies sind die Dörfer Waldbrücke (siehe Abbildung 4.8), das Teil der Gemeinde Weingarten bei Karlsruhe ist, und Ettenheim (siehe Abbildung 4.9) im Schwarzwald, sowie ein fiktiver Datensatz, der eine Gebäudegruppe im LoD3 (siehe Abbildung 4.10) enthält. Der letztgenannte CityGML-Datensatz zeigt drei LoD3-Gebäudeobjekte einschließlich der folgenden Feature-Typen:

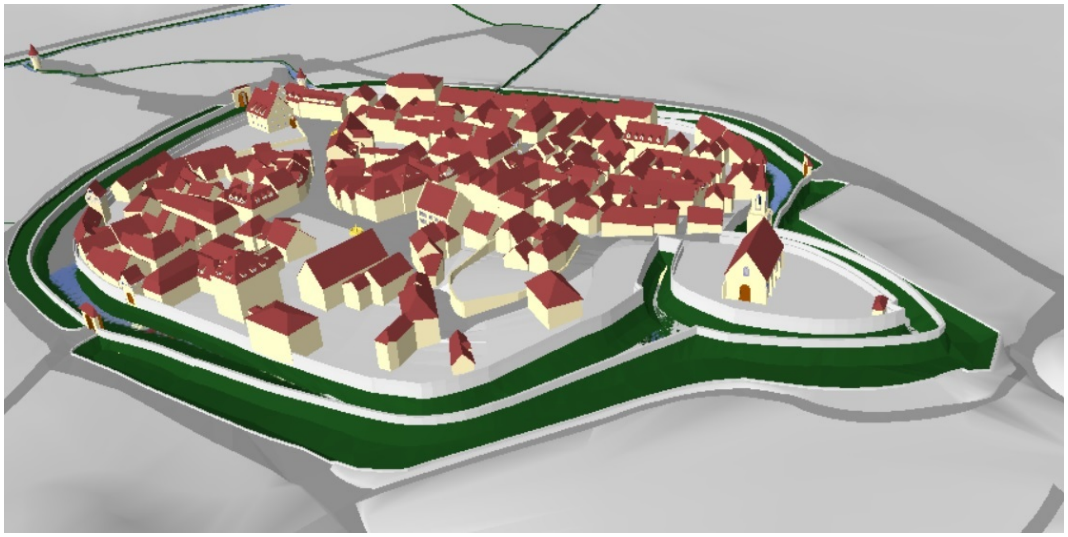


Abbildung 4.9: CityGML-Modell des Dorfes Ettenheim im Schwarzwald

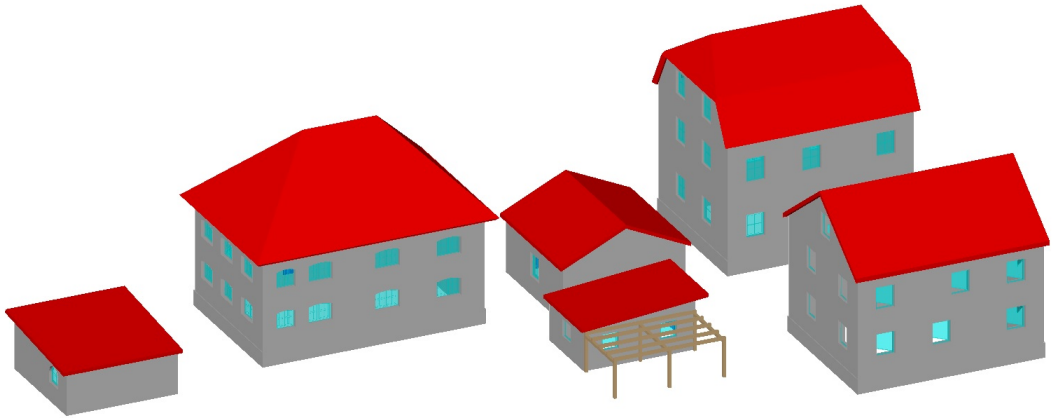


Abbildung 4.10: CityGML-Modell einer Gebäudegruppe im LoD3

GroundSurface, *WallSurface*, *RoofSurface*, *BuildingInstallation*, *Window*, *Door*. Der Datensatz wurde durch Export aus *Google Sketchup* erzeugt. Der Datensatz von Waldbrücke liegt im LoD1 vor und enthält 606 Gebäude und Gebäudeteile auf 525 verschiedenen Grundstücken, darunter Transport-, Gewässer- und einige einzelne Vegetationsobjekte. Dieser Datensatz wurde aus Katasterdaten erstellt. Das virtuelle 3D-Stadtmodell von Ettenheim liegt im LoD3 vor und wurde automatisch abgeleitet aus einem IFC-Datensatz und manuell angereichert in Bezug auf die verwendeten CityGML-Feature-Typen.

Der Potsdam-Datensatz (siehe Abbildung 4.11) ist im Download-Paket der 3DCityDB enthalten. Dieser enthält die Teile der Stadt Potsdam in LoD1. Die

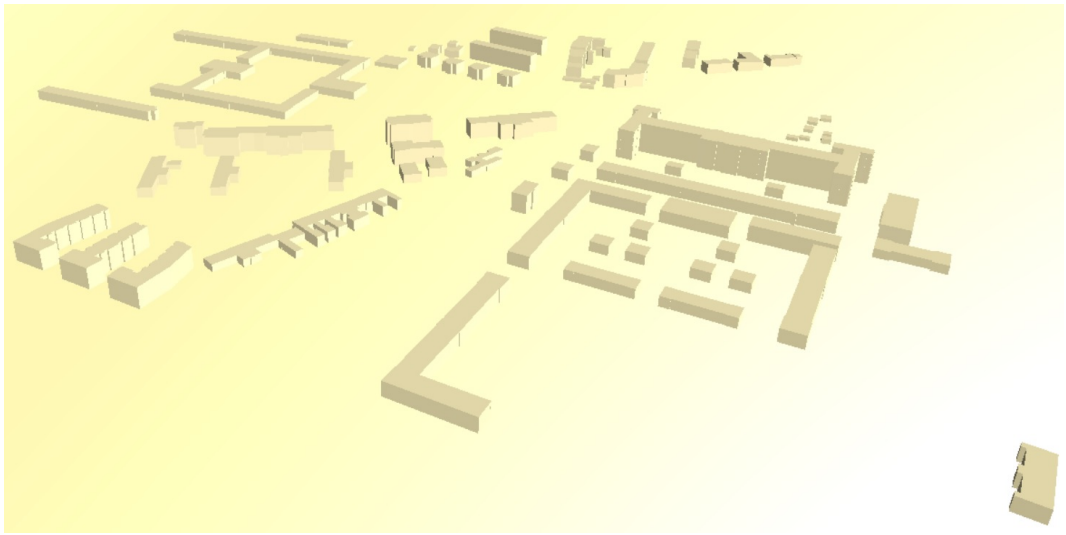


Abbildung 4.11: CityGML-Modell eines Teils von Potsdam

Art der Datenerfassung beziehungsweise die Erstellung des Modells sind in diesem Fall unbekannt.

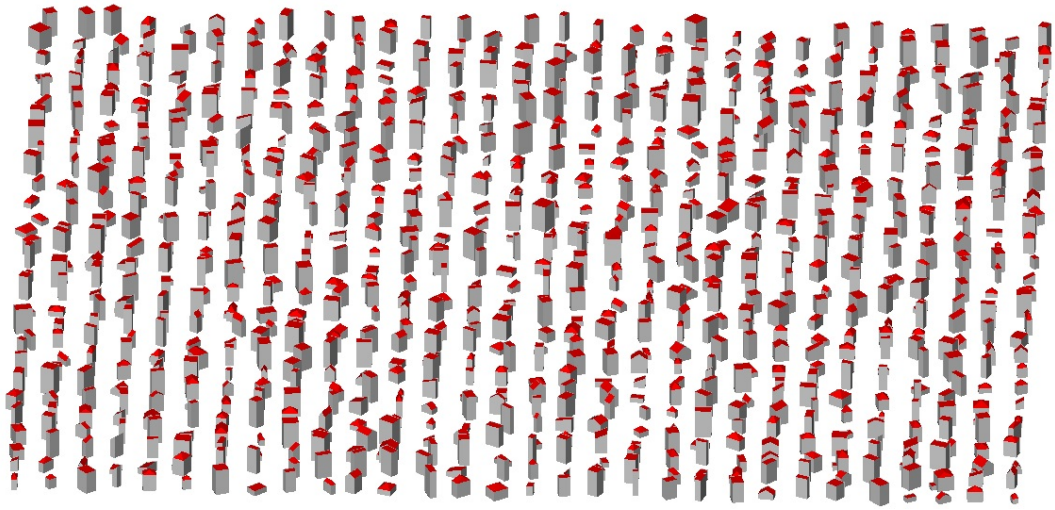


Abbildung 4.12: synthetisches CityGML-Modell von Random3DCity im LoD2 (1)

Zusätzlich wurden fünf synthetische Datensätze verwendet, die mit dem Tool Random3DCity (Biljecki et al., 2016b) erzeugt wurden und von der Projekthomepage (Biljecki, 2018) heruntergeladen werden können. Dabei handelt es sich um die Datensätze „Random3DCity LOD2_0_F0“ (siehe Abbildung 4.12), „Random3DCity LOD2_3_F0“ (siehe Abbildung 4.13), „Random3DCi-

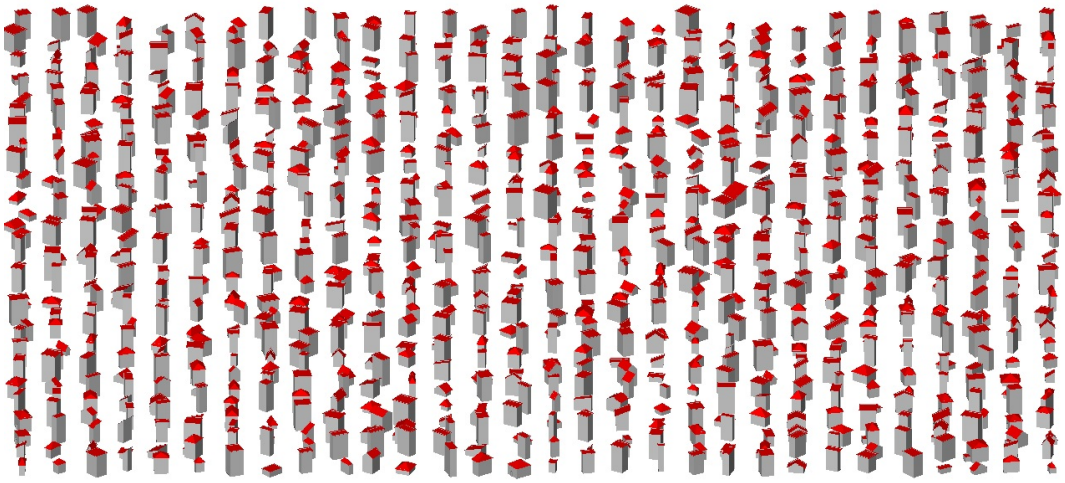


Abbildung 4.13: synthetisches CityGML-Modell von Random3DCity im LoD2 (2)

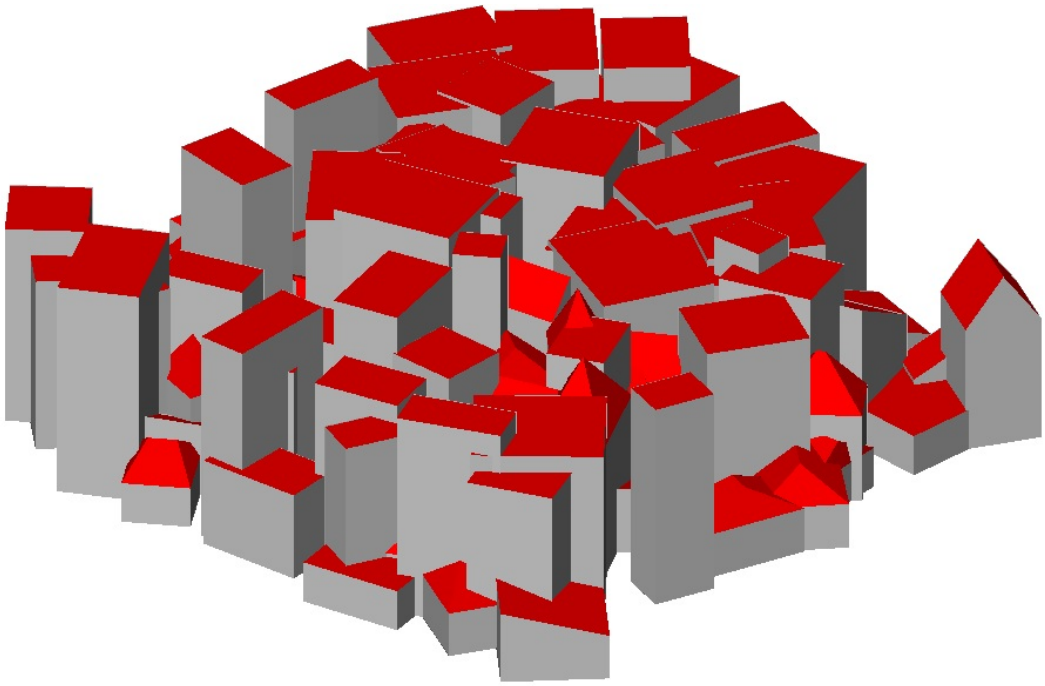


Abbildung 4.14: synthetisches CityGML-Modell von Random3DCity im LoD2 mit Fehlern

ty LOD2_overlap“ (siehe Abbildung 4.14), „Random3DCity LOD3_2“ (siehe Abbildung 4.15) und „Random3DCity LOD3_overlap“ (siehe Abbildung 4.16). Drei dieser Datensätze enthalten Gebäude in LoD2 und die anderen in LoD3. Zwei dieser Datensätze, jeweils ein Datensatz in LoD2 und ein Datensatz in

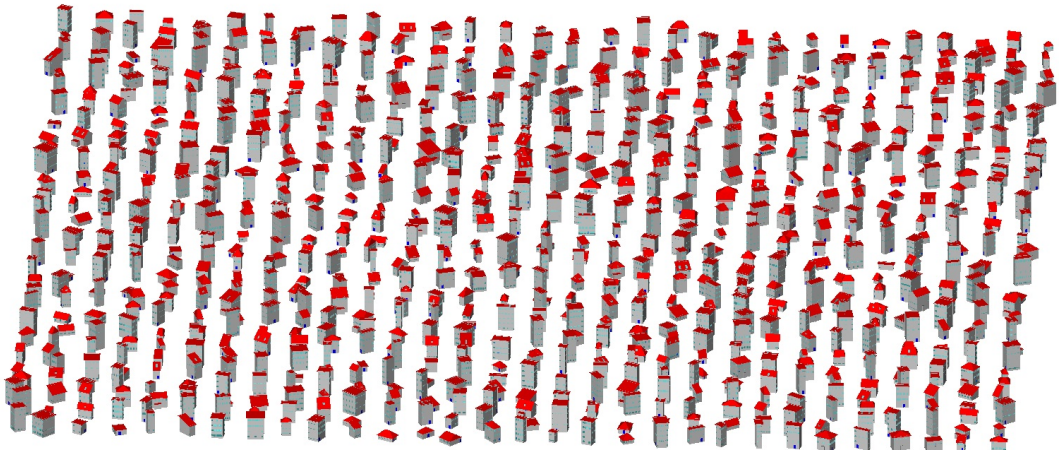


Abbildung 4.15: synthetisches CityGML-Modell von Random3DCity im LoD3

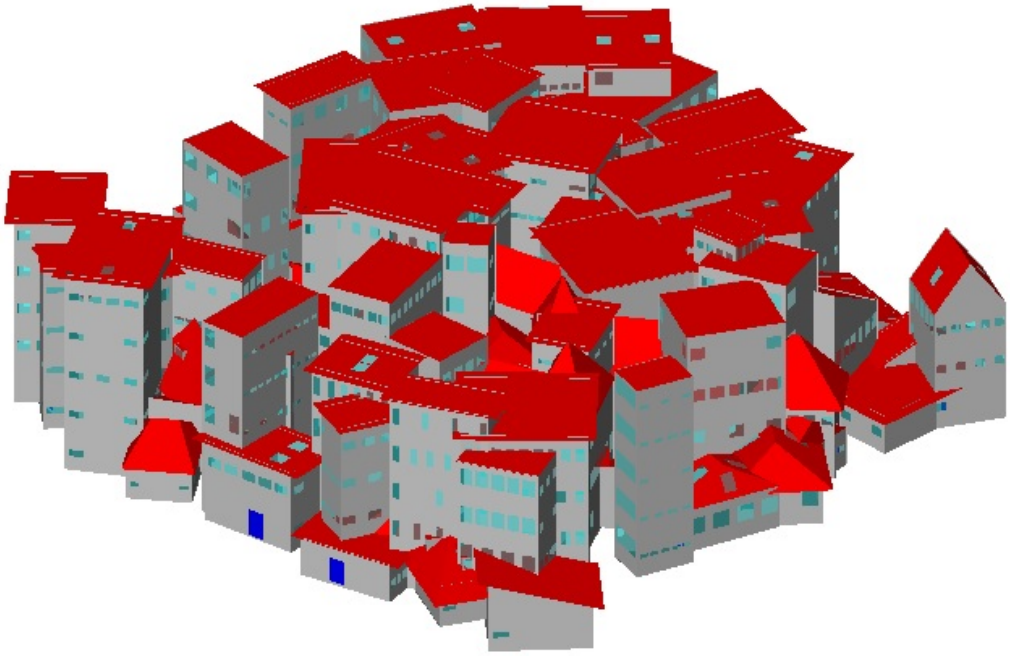


Abbildung 4.16: synthetisches CityGML-Modell von Random3DCity im LoD3 mit Fehlern

LoD3 wurden mit Fehlern modelliert. Bei den Fehlern handelt es sich um überlappende Gebäude.

Wie bereits in Abschnitt 3.3 erwähnt, ist ein ungelöstes Problem in der Implementierung, dass die von PostGIS bereitgestellte SFCGAL-Funktion *st_3d-intersection* bei bestimmten Polygonkonfigurationen zu einer Unterbrechung

der Datenbankverbindung führt. Aus diesem Grund wurden aus drei Datensätzen einzelne Polygone gelöscht um diese Datensätze prozessieren zu können. Dies betrifft acht Polygone aus dem Datensatz „Pariser Platz“, sechs Polygone aus dem Datensatz „Delft“ und zwölf Polygone aus dem Datensatz „Gebäudegruppe LoD3“.

4.2 Ergebnisse

In diesem Abschnitt werden die Ergebnisse der durchgeführten Analyse der oben beschriebenen Datensätze vorgestellt.

Tabelle 4.1 zeigt eine Liste der CityGML-Datensätze, die in dieser Arbeit verwendet wurden, zusammen mit der Anzahl der Polygone und der Anzahl der nicht leeren Schnitte der einzelnen Polygone miteinander. Man sieht, dass die überwiegende Mehrheit der Polygone *valid* ist, das heißt, dass diese Polygone eben sind und keine Selbstüberschneidungen aufweisen und wenn ein innerer Ring existiert, sind die Position und Orientierung des inneren Rings korrekt. Insbesondere sind diese Polygone somit topologisch konsistent.

Abbildung 4.17 zeigt die Prozentsätze der vier häufigsten Geometrietypen (*Point*, *LineString*, *MultiLineString* und *TIN*) der Schnitte jeweils zweier Polygone, wenn die Schnittmenge nicht leer ist und die beiden Polygone verschieden sind. Beispiele für Schnitte mit den Geometrietypen *Point* und *LineString* sind in Abbildung 3.3 zu sehen. Diese beiden Typen können topologisch konsistent sein oder auch nicht. Das wurde in Abschnitt 3.1 bei der Einführung der Schnittmatrix bereits ausführlich diskutiert. Ein *MultiLineString* ist hierbei eine Vereinigung von *LineStrings* mit mindestens zwei Komponenten. Dieser Schnittgeometrietyp kann nur dann konsistent sein, wenn beide Polygone identisch sind. Die Abkürzung *TIN* steht für *Triangulated Irregular Network*. Ist der Schnittgeometrietyp *TIN* bedeutet das, dass die Konfiguration topologisch inkonsistent ist, da die Schnittmenge eine Fläche enthält, die ebenfalls in den Flächen beider Polygone enthalten sein muss. Die exakten Schnittmatrizen in den Fällen der beiden letztgenannten Schnittgeometrietypen exakt zu bestimmen, wäre mit sehr großem Aufwand verbunden gewesen und hätte kaum Relevanz für das Ergebnis der Analyse, da diese Fälle in der überwiegenden Mehrheit topologisch inkonsistent sind (einzige Ausnahme identische Polygone

Name	Polygone	valid [%]	Schnitte	KS
Alexanderplatz	57.831	99,7	1.316.110	lokal
Pariser Platz	6.008	99,7	52.368	Soldner
Delft	112.888	100	1.168.248	UTM
Ettenheim	46.145	86,1	339.402	GK
Gebäudegruppe LoD3	18.993	99,7	304.343	GK
Lindenallee u. Kranichweg	745	86,7	4.564	GK
Rheinstrasse	643	92,5	3.530	GK
Rintheim	18.510	100	120.198	GK
Tennesseeallee	624	90,4	4.410	GK
Weidenweg	918	80,8	5.066	GK
Potsdam	1.547	100	9.074	UTM
Random3DCity LOD2_0_F0	6.341	92,9	24.962	lokal
Random3DCity LOD2_3_F0	7.890	88,9	36.710	lokal
Random3DCity LOD2_overlap	679	99,6	8.642	lokal
Random3DCity LOD3_2	40.263	88,7	8.212	lokal
Random3DCity LOD3_overlap	4.663	88,8	2.206	lokal
Waldbrücke	4.038	93,6	20.644	lokal

Tabelle 4.1: Liste der CityGML-Datensätze, die in dieser Arbeit verwendet wurden. In der Spalte „Polygone“ wird die Anzahl der im jeweiligen Datensatz enthaltenen Polygone aufgelistet und unter „Schnitte“ findet man die Anzahl der Polygonschnitte. Die Spalte „valid“ gibt den Anteil von validen Polygonen innerhalb des Datensatzes an. „KS“ steht für Koordinatensystem, lokal = lokales Koordinatensystem, GK = Gauß-Krüger-Koordinatensystem, UTM = Universal Transverse Mercator-System

bei *MultiLineString*). Daher wurde darauf verzichtet. Es ist ersichtlich, dass der Schnitt in der großen Mehrheit entweder ein Punkt oder ein Liniensegment ist. Dies ist ein weiterer Grund, warum diese zwei Arten von Schnitten weiter untersucht wurden. Somit konnten zwischen $48,4\%$ und $99,9\%$ der Schnitte für die untersuchten Datensätze analysiert werden. Eine Ausnahme bildet der Datensatz „Random3DCity LOD3_overlap“ (detaillierte Ergebnisse siehe Tabelle 4.2), welcher einer der synthetischen Datensätzen mit topologischen

Fehlern ist.

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	955 davon: 648 2P. (67, 85%) konsistent: 388 (59, 88%) inkonsistent: 260 (40, 12%) 265 3P. (27, 75%) 20 4P. (2, 09%) 20 5P. (2, 09%) 2 6P. (0, 21%)	43, 29%
MULTILINESTRING	549	24, 89%
TIN	486	22, 03%
POINT	112 davon: point-point: 110 (98, 21%) point-line: 0 (0%) line-line: 2 (1, 79%) point-area: 0 (0%)	5, 08%
TRIANGLE	92	4, 17%
MULTIPOINT	8	0, 36%
GEOMETRYCOLLECTION	4	0, 18%
Gesamt	4663 Polygone 523 (11, 21%) invalid 2206 Schnitte	

Tabelle 4.2: Detaillierte Ergebnisse der Analyse für den Datensatz „Random-3DCity LOD3_overlap“. Auffällig ist, dass hier nicht die Mehrheit der Schnitte vom Geometrietyp *Point* oder *LineString* ist.

Auffällig ist auch, dass die Verteilung im Datensatz „Alexanderplatz“ signifikant von den anderen Datensätzen abweicht. Die detaillierten Ergebnisse für diesen Datensatz sind in Tabelle 4.3 zu sehen. Ein Grund hierfür könnte in der Art der Erstellung des Datensatzes sein. Dieser wurde als einziger betrachteter

Datensatz automatisch aus Katasterdaten generiert, ohne dass eine manuelle Nachbearbeitung stattfand.

Geometrietyp Schnitt	**Anzahl**	**Anteil**
POINT	931734 davon: point-point: 897556 (96, 33%) point-line: 22856 (2, 45%) line-line: 1330 (0, 14%) point-area: 9992 (1, 07%)	70, 79%
LINESTRING	290722 (alle 2 P.) konsistent: 196700 (67, 66%) inkonsistent: 94022 (32, 34%)	22, 09%
TIN	65840	5, 00%
TRIANGLE	27814	2, 11%
Gesamt	57831 Polygone 148 (0, 26%) invalid 1316110 Schnitte	

Tabelle 4.3: Detaillierte Ergebnisse der Analyse für den Datensatz „Alexanderplatz“. Auffällig ist hier, dass die Verteilung der Schnittgeometrietype in diesem Datensatz signifikant von den anderen Datensätzen abweicht.

Auch die Datensätze aus Karlsruhe, sowie die beiden synthetischen Datensätze weisen einen höheren Anteil an Schnitten vom Typ *Point* auf. Für die Datensätze aus Karlsruhe lässt sich diese Gemeinsamkeit vermutlich ebenfalls auf die gleiche Art der Datenerhebung, in diesem Fall aus LIDAR-Daten, zurückführen.

Im Fall der zwei fehlerhaften synthetischen Datensätze ist der Anteil der Schnittgeometrietypen *Point* und *LineString* niedriger und der Anteil von *MultiLineString* und *TIN* ist höher.

Abbildung 4.18 zeigt die relativen Häufigkeiten des Auftretens der vier Schnittmatrizen, wenn der Schnitt zweier unterschiedlicher Polygone ein Punkt ist. Mit Ausnahme des Datensatzes von Delft (detaillierte Ergebnisse siehe Ta-

belle 4.4) besteht jeweils die Mehrheit aus dem topologisch konsistenten Fall „point-point“. Die meisten Datensätze haben jedoch auch einen großen Anteil an topologisch inkonsistenten Schnittmatrizen vom Typ „point-line“. Nur die synthetischen Datensätze mit Ausnahme von „Random3DCity LOD3_overlap“ enthalten keine topologisch inkonsistenten Schnitte vom Typ *Point*.

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	953268 davon: 929262 2P. (97, 48%) konsistent: 840678 (90, 47%) inkonsistent: 88584 (9, 53%) 23954 3P. (2, 51%) 46 4P. (0, 005%) 4 6P. 2 5P.	81, 60%
POINT	122336 davon: point-point: 27560 (22, 53%) point-line: 94776 (77, 47%) line-line: 0 (0%) point-area: 0 (0%)	10, 47%
TIN	49988	4, 28%
MULTILINESTRING	42440	3, 63%
GEOMETRYCOLLECTION	84	0, 01%
MULTIPOINT	68	0, 01%
TRIANGLE	64	0, 01%
Gesamt	112888 Polygone (alle valid) 6 gelöscht 1168248 Schnitte	

Tabelle 4.4: Detaillierte Ergebnisse der Analyse für den Datensatz „Delft“. Auffällig ist, dass die Mehrheit der Schnitte vom Geometrietyp *Point* inkonsistent sind.

Abbildung 4.19 zeigt den Anteil topologisch konsistenter oder inkonsistenter Polygonpaare, wenn die Schnittmenge ein Punkt oder ein Liniensegment ist (Spezialfall des Typs *LineString*). Dies sind bei weitem die häufigsten Schnittgeometrietypen, zumindest in den „Realwelt"-Datensätzen. Für die drei synthetischen Datensätze ohne absichtlich eingebaute topologische Fehler, stellt sich heraus, dass alle diese Konfigurationen topologisch konsistent sind. Für die realen Datensätze sind meist die Mehrheit, aber keineswegs alle Konfigurationen topologisch konsistent. Der häufigste inkonsistente Schnittgeometrietyp für diese Datensätze ist *LineString*.

Bemerkenswert ist an dieser Stelle auch noch, dass auch der Datensatz „Delft" topologische Inkonsistenzen enthält, obwohl dieser Datensatz mit *3dfier* erstellt wurde. In Ledoux et al. (2021) heißt es die Ausgabe der Software sei eine „wasserdichte" Oberfläche ohne sich überschneidende Dreiecke und ohne Löcher. Dies müsste zur Abwesenheit topologischer Inkonsistenzen führen.

Detaillierte Ergebnisse für die übrigen Datensätze in Form von Tabellen findet man im Anhang A.

Betrachtet man alle Ergebnisse fällt auf, dass zwischen den Datensätzen eine hohe Inhomogenität besteht. Die Verteilung der Schnittgeometrietypen und der Anteil der topologisch inkonsistenten Konstellationen scheint von der Art der Datenerhebung und dem LoD abhängig zu sein. Darauf und auf weitere Ergebnisse wird im Detail in Giovanella et al. (2018) und in Giovanella et al. (2019) eingegangen.

Abschließend sei erwähnt, dass im Vorfeld der eigentlichen Untersuchung der Datensätze auf topologische Fehler, alle Datensätze genau analysiert wurden. Alle Datensätze wurden visualisiert und es wurde in den dazugehörigen XML-Files erhoben, welches Koordinaten- und Höhensystem laut EPSG-Code jeweils vorliegt. Außerdem wurden jeweils einige der tatsächlichen Koordinatenwerte überprüft, um festzustellen, ob diese plausibel sind, das heißt ob diese zum angegebenen Koordinatensystem passen. Zusätzlich wurde für alle Datensätze, außer bei lokalem Koordinatensystem, kontrolliert, ob die Koordinaten nicht nur plausibel, sondern auch korrekt verortet sind.

Bei diesen Untersuchungen stellten sich einige Ungereimtheiten in vielen Datensätzen dar, wie zum Beispiel fehlerhaft verortete oder unplausible Koordinatenwerte, unplausible oder ganz fehlende Höhenwerte oder fehlender EPSG-

Code im XML-File. Da diese Probleme keinen Einfluss auf das Ergebnis der Konsistenzprüfung hatten, wird an dieser Stelle nicht im Detail darauf eingegangen. Der Vollständigkeit halber wurden die ermittelnden Koordinatensysteme dennoch in Tabelle 4.1 eingefügt. Es sei zusätzlich erwähnt, dass sich auch hier zeigt, wie stark fehlerbehaftet reale CityGML-Datensätze derzeit noch sind und dass noch einige Schwierigkeiten überwunden werden müssen um diese Datensätze für weitergehende Analysen möglichst sinnvoll nutzen zu können. Dies führt uns direkt zum folgenden Kapitel 5, wo die Erkenntnisse der Arbeit in Abschnitt 5.1 nochmal abschließend zusammengefasst werden und in Abschnitt 5.2 ein Ausblick auf mögliche zukünftige Schritte gegeben wird.

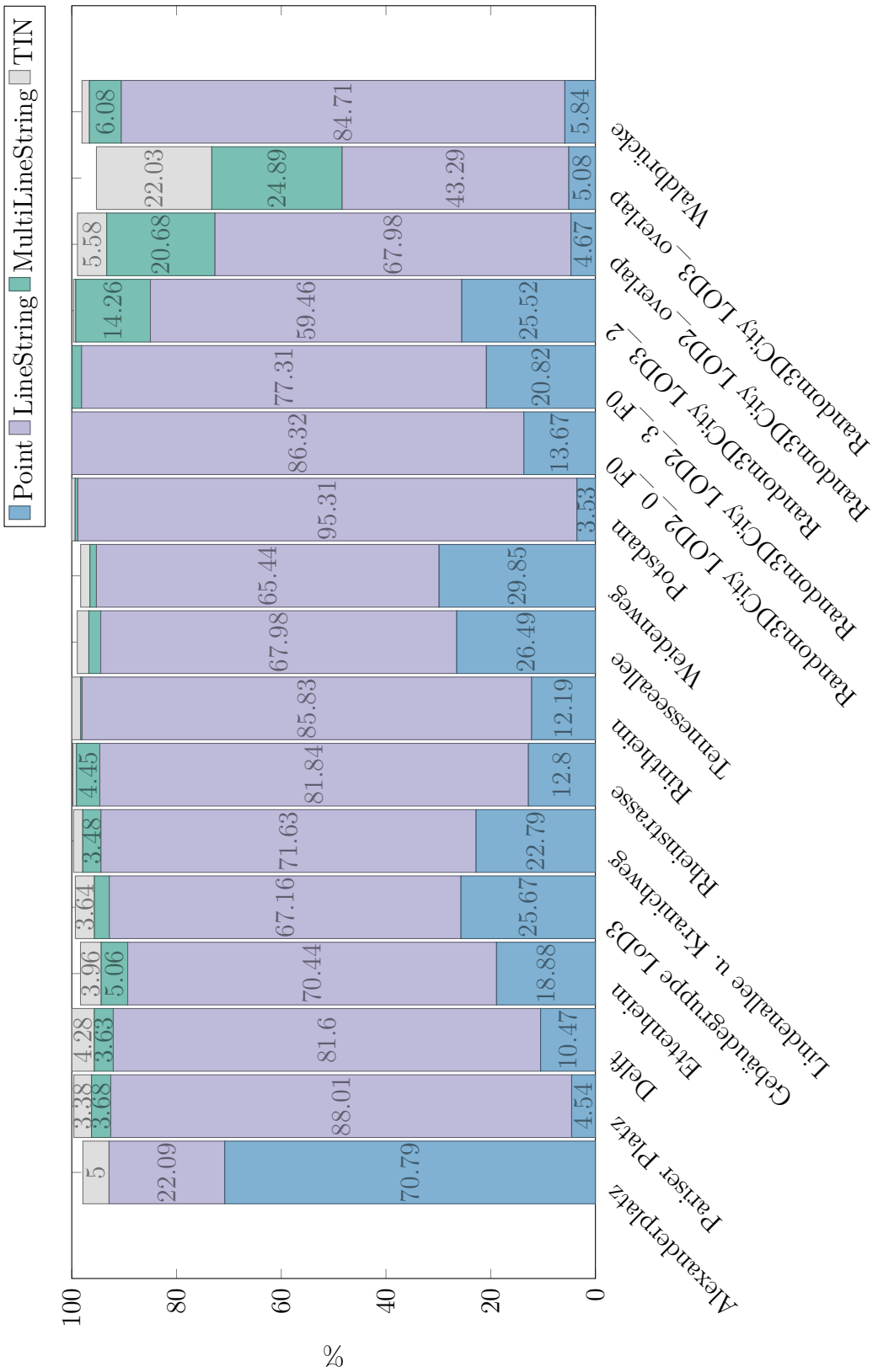


Abbildung 4.17: Anteile der häufigsten nicht leeren Polygon-Schnitttypen

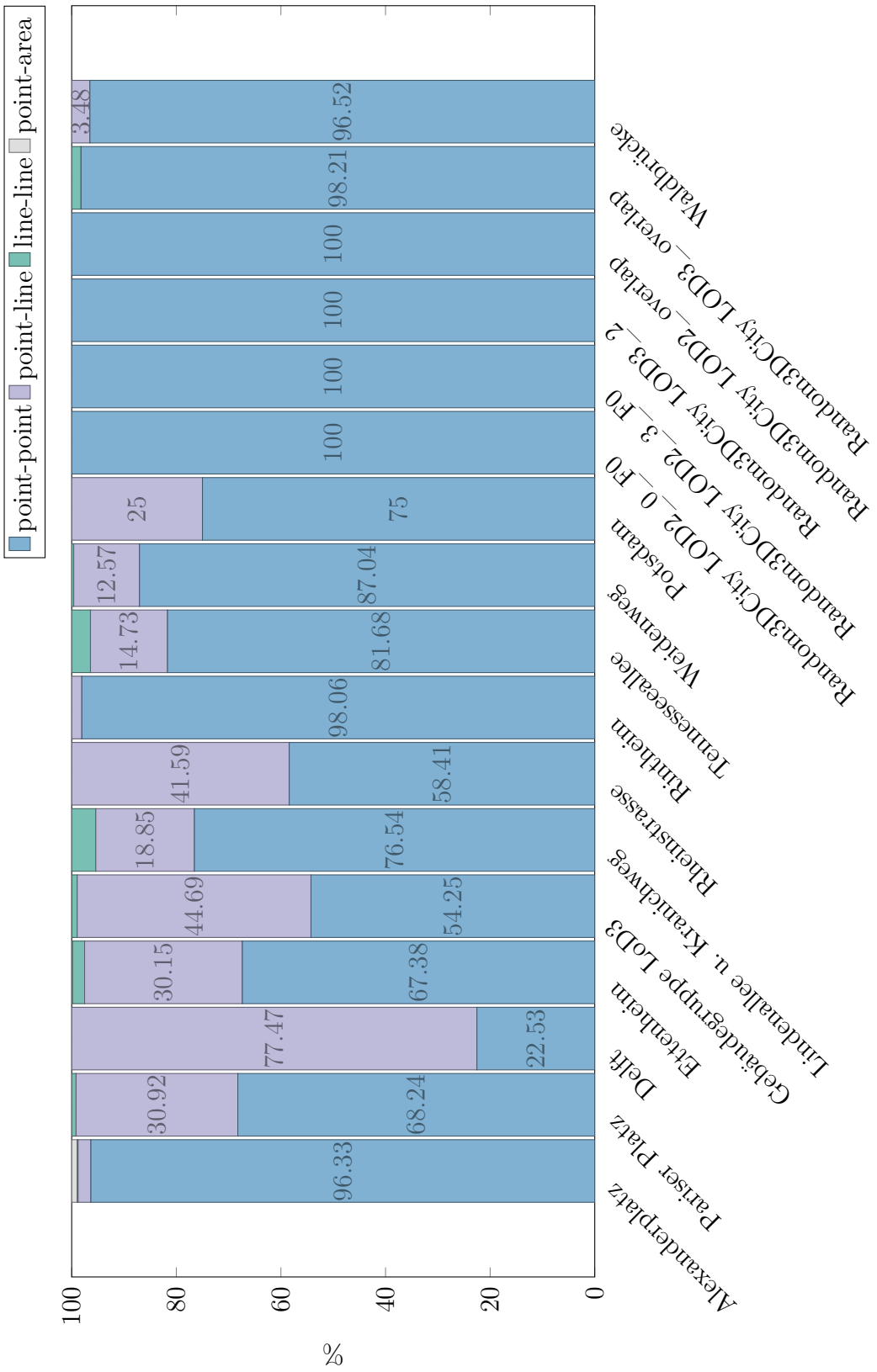


Abbildung 4.18: Schnittmatrizen, wenn die Schnittmenge ein Punkt ist.

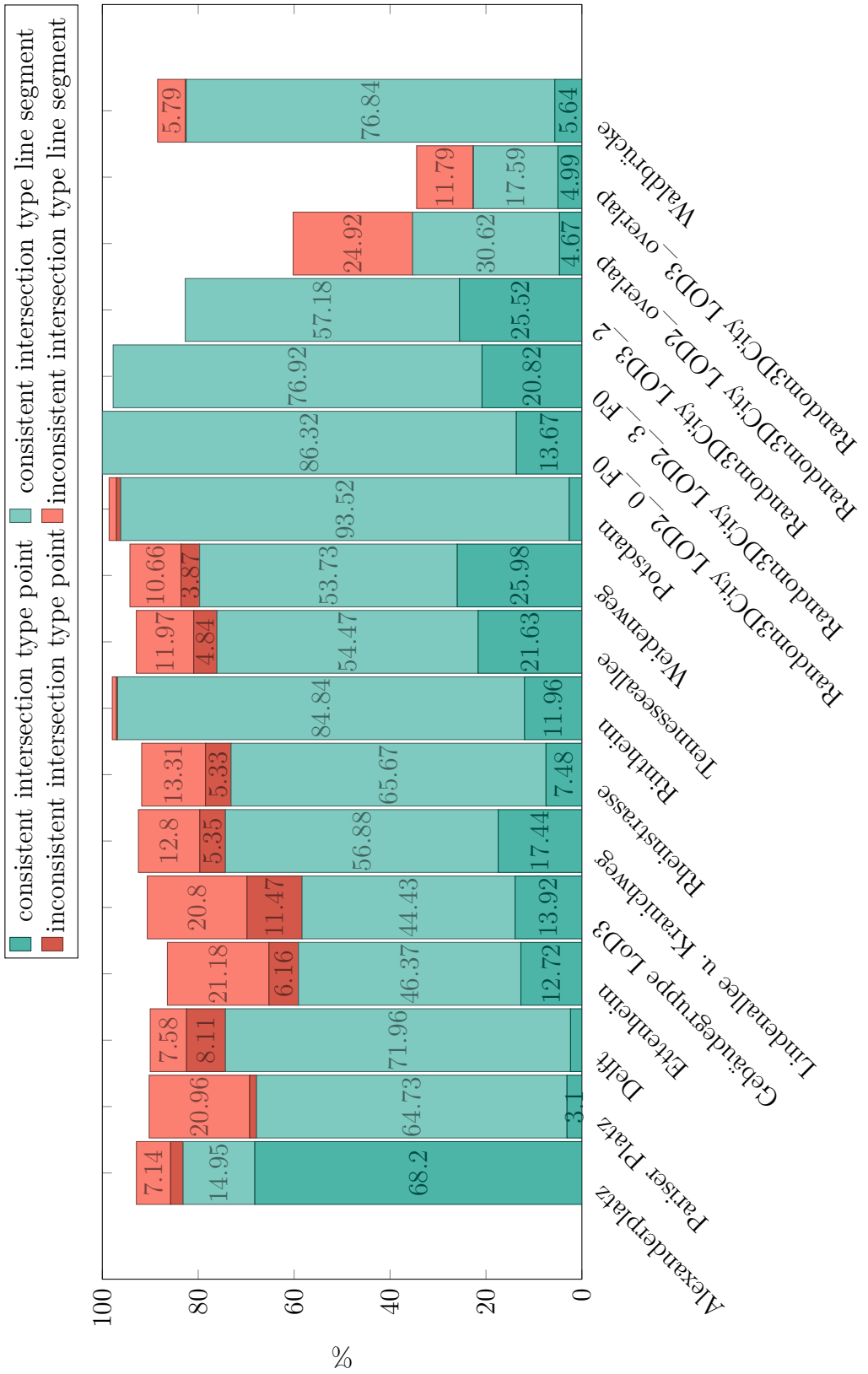


Abbildung 4.19: Topologisch konsistente und inkonsistente Konfigurationen mit Punkt- und Liniensegmentschnitten.

Kapitel 5

Fazit und Ausblick

In diesem Kapitel werden zunächst die Erkenntnisse dieser Arbeit in Abschnitt 5.1 resümiert und diskutiert und anschließend wird in Abschnitt 5.2 ein Ausblick auf mögliche zukünftige Schritte gegeben.

5.1 Fazit

Die Topologie ist ein leistungsfähiges und allgemeines Konzept zur Analyse von GIS- und BIM-Datenstrukturen und räumlichen Beziehungen, das in neuen Anwendungen wie Smart Cities (Meijer and Bolívar, 2016) von großer Bedeutung sein wird. Neben Semantik und Geometrie wird die Topologie wahrscheinlich zu einem Schlüsselkonzept, um die Modellierung und Verwaltung von Geodaten effizient zu unterstützen.

Die topologische Konsistenz ist eine Voraussetzung für die Analyse und Simulation von Geodatenmodellen, da aufwändige geometrische Operationen wie Schnittpunktberechnungen vermieden werden können und die Simulation auf dem topologischen Modell selbst ausgeführt werden kann. Ein räumliches Modell, das aus Punkten, (offenen) Liniensegmenten, (offenen) Polygonen, (offenen) Polytopen etc. besteht, ist topologisch konsistent, wenn alle Objekte paarweise disjunkt sind. Diese Definition erscheint zuerst in Giovanella et al. (2019) und entspricht der Definition in der vorliegenden Arbeit. Diese unterscheidet sich von früheren Definitionen der topologischen Konsistenz in der Literatur. Unter den vielen verschiedenen Begriffen der topologischen Konsistenz bezieht sich die hier betrachtete auf die Geometrie und den Inzidenzgraphen, so dass

topologische Konsistenz bedeutet, dass der Inzidenzgraph die Topologie modelliert, die dem geometrischen Modell zugrunde liegt. Das Ziel von *Boundary Representation* Modellen ist es also, die Topologie durch die zugrunde liegenden geometrischen Modelle darzustellen. Um die Topologie korrekt darzustellen, ist jedoch die Richtigkeit des Inzidenzgraphen des *Boundary Representation* Modells erforderlich. Das heißt, die Verwendung des *Boundary Representation* Modells bedeutet, dass man sich auf die Korrektheit des Inzidenzgraphen verlassen muss. Dies ist nur dann der Fall, wenn die dem geometrischen Modell zugrunde liegende Topologie mit der dem *Boundary Representation* Modell zugrunde liegenden Topologie übereinstimmt – mit anderen Worten, wenn die Daten im Sinne dieser Arbeit topologisch konsistent sind. Der Vorteil ist, dass Geometrie und Topologie so sinnvoll miteinander verknüpft werden.

Der Abschnitt 1.2 bezieht sich auf verschiedene Begriffe der topologischen Konsistenz in der Literatur und auf einige mögliche, aber konsistente Situationen, die sie nicht erfassen. Daraus folgt, dass der Begriff der topologischen Konsistenz in der Literatur eine topologische Konsistenz im Sinne der vorliegenden Arbeit impliziert, aber nicht jede Konfiguration, die im Sinne der vorliegenden Arbeit topologisch (in)konsistent ist, in der Literatur topologisch (in)konsistent ist.

Ein weiteres Problem ist die Art der topologischen Konsistenz, die durch den ISO 19107 Standard vorgegeben wird. Ein Validierungswerkzeug für ISO-/OGC-Standards ist das Tool val3dity (Ledoux, 2018). Die Konsultation der Dokumentation von val3dity zeigt eine lange Liste möglicher Fehler. Eine Überprüfung der Liste möglicher Fehler, die val3dity finden kann, zeigt allerdings, dass nicht alle Konfigurationen, die den val3dity-Test bestehen, im Sinne der vorliegenden Arbeit topologisch konsistent sind. Wenn beispielsweise ein Paar A, B verschiedener Hüllen genommen wird, werden die folgenden Einschränkungen nicht erfasst:

- *Knoten mit Fläche.* Ein Knoten von A darf keinen nicht leeren Schnittpunkt mit einer Fläche von B haben.

- *Kante mit Kante.* Ein nicht leerer Schnitt einer Kante von A mit einer Kante von B muss eine Kante sowohl von A als auch von B sein.

- *Kante mit Fläche.* Eine Kante von A darf keinen nicht leeren Schnitt mit

einer Fläche von B haben.

- *Fläche mit Fläche.* Ein nicht leerer Schnitt einer Fläche von A mit einer Fläche von B muss eine Fläche von A und B sein.

Man kann leicht erkennen, dass diese Konsistenzregeln erfüllt sein müssen, wenn man den Inzidenzgraphen verwenden möchte, zum Beispiel um den Zusammenhang abzufragen: Wenn es in A und B keine gemeinsamen Eckpunkte, Kanten oder Flächen gibt, sind die beiden Hüllen nicht zusammenhängend, es sei denn, sie verstoßen gegen die obigen Regeln. Diese Konsistenzregeln werden durch den hier verwendeten Begriff der topologischen Konsistenz erfüllt.

Zwölf reale und sechs synthetische CityGML-Datensätze wurden untersucht, um die topologische Konsistenz in diesem Sinne zu überprüfen und die häufigsten topologischen Inkonsistenzen zu klassifizieren. Es stellte sich heraus, dass echte CityGML-Daten größtenteils topologisch inkonsistent sind und die Verteilung ihrer Inkonsistenztypen variiert. Der häufigste inkonsistente Fall ist, wenn der Schnitt zweier Polygone ein Liniensegment ist. In dem Fall, dass der Schnitt ein Punkt ist, ist die häufigste Inkonsistenz, wenn ein Knoten im Inneren eines Liniensegments liegt.

Die Ergebnisse dieser Arbeit zeigen also, dass viele in CityGML modellierte Gebäude eine gewisse Anzahl von Polygonpaaren enthalten, die eine topologisch inkonsistente Konfiguration bilden. Es ist in CityGML möglich, gemäß dem Standard korrekt zu modellieren und dennoch ein topologisch inkonsistentes Modell zu haben. Dies hat zur Folge, dass fehlerhaft modellierte CityGML-Datensätze dann nicht dem ISO 19107 Standard entsprechen. Daher sind die bestehenden CityGML-Daten in diesen Fällen für eine effiziente Analyse jenseits der Visualisierung nur bedingt geeignet, da topologische Abfragen zwangsläufig falsche Ergebnisse liefern, wenn sie sich nur auf den Inzidenzgraphen stützen, um kostspielige geometrische Berechnungen zu vermeiden. Dies sollte den Anwendern zumindest bewusst sein und beachtet werden, in dem zum Beispiel verwendete Datensätze zuvor auf topologische Fehler überprüft werden.

Daher ist bei der automatischen Erstellung eines Geometriemodells (zum Beispiel mit Tools wie *3dfier* (Ledoux et al., 2021)) in CityGML aus Punktwolkendaten oder aus einer anderen Weise der automatischen Erstellung, eine

Prüfung auf topologische Konsistenz im Sinne dieser Arbeit sinnvoll und notwendig. Darüber hinaus ist es erforderlich, Wege zu finden, um topologisch inkonsistente Daten zu korrigieren, möglicherweise abhängig von der Art der angetroffenen topologischen Inkonsistenzen. Darauf wird im folgendem Abschnitt detailliert eingegangen.

5.2 Ausblick

Zur Unterscheidung zwischen verschiedenen Formen topologischer Inkonsistenz ist die in der vorliegenden Arbeit definierte Schnittmatrix ein erster Indikator. Einige Matrizen sind jedoch nicht eindeutig: Sie können sowohl aus konsistenten als auch aus inkonsistenten Konfigurationen stammen. Eine Klassifizierung dieser Matrizen wurde eingeführt. Insbesondere wurde gezeigt, welche Matrizen als Schnittmatrizen auftreten können und dass Schnittmatrizen topologisch konsistenter Daten Diagonalmatrizen sind.

Ein möglicher Ansatz, um unter anderem uneindeutige Fälle weiter zu klassifizieren, könnte eine Erweiterung von Zhou et al. (2021) in den dreidimensionalen Raum sein. Diese Ergebnisse könnten mit der Schnittmatrix kombiniert werden oder die Schnittmatrix dadurch erweitert beziehungsweise verfeinert werden.

In Zhou et al. (2021) werden verfeinerte topologische Beziehungen zwischen Linie und Region und zwischen Linie und Linie im zweidimensionalen Raum einheitlich durch die Reihenfolge, Dimension und den topologischen Typ der Schnittkomponenten dargestellt. Um die relevanten Definitionen an die traditionellen Erkenntnisse im euklidischen zweidimensionalen Raum anzupassen, wird das (einfache) räumliche Objekt basierend auf einer Mannigfaltigkeitstopologie definiert und die räumlichen Schnittkomponenten werden basierend auf der Gesamtmenge der Objektschnittpunkte definiert. Dann wird die topologische Invariante des Knotengrads eingeführt und die benachbarten Punktarten (z.B. *Null*, *On*, *In* und *Out*) werden definiert, um die Schnittkomponententypen zu unterscheiden. So werden im zweidimensionalen Raum nach Ausschluss von unmöglichen und symmetrischen Typen 29 Arten von Schnittlinien klassifiziert, welche 21 Schnittlinienarten zwischen Linien und Regionen und acht Schnittlinienarten zwischen Linien und Linien einschließen. Zusätzlich werden

sechs Arten von Schnittpunkten, einschließlich zwei Schnittpunktarten zwischen Linien und Regionen und vier Schnittpunktarten zwischen Linien und Linien, klassifiziert.

Biljecki et al. (2015a) untersuchen die Verknüpfung entsprechender geometrischer Merkmale über mehrere Darstellungen hinweg. Sie beschreiben die möglichen topologischen Fälle und zeigen, wie diese Zusammenhänge erkannt und explizit gespeichert werden können. Ein Software-Prototyp wurde implementiert, um übereinstimmende Merkmale in und zwischen Levels of Detail (LoDs) zu erkennen und diese automatisch durch die Einrichtung expliziter topologischer Beziehungen (mit *XLink*) zu verknüpfen. Die mit ihren synthetischen Testdatensätzen durchgeführten Experimente zeigen eine beträchtliche Anzahl aufeinander abgestimmter Geometrien. So werden Redundanzen in synthetischen CityGML-Datesätzen reduziert und damit die topologische Konsistenz verbessert und auch der Speicherbedarf der Daten kann verringert werden. Die Software wurde bisher nicht mit realen Datensätzen getestet, es wäre jedoch zukünftig denkbar, eine derartige Analyse als eine Art Vorverarbeitungsschritt der vorliegenden Arbeit einzusetzen.

Wenn der Schwerpunkt bei der Erstellung des Modells eher auf der Analyse oder sogar Simulation statt nur auf der Visualisierung liegt, müssen alle n-dimensionalen geometrischen Körper und ihre topologischen Beziehungen auf effiziente und eindeutige Weise deklariert oder berechnet werden. Die Neuberechnung aller topologischen Inkonsistenzen durch Auffinden aller n-dimensionalen Schnittpunkte zur Laufzeit ist eine komplexe Aufgabe, die vermieden werden sollte. Das Problem wird noch komplexer, wenn weitere Dimensionen wie Zeitkoordinaten oder andere thematische oder semantische Koordinaten hinzugefügt werden. Bei der Erstellung eines Geometriemodells in CityGML aus Punktwolkendaten oder durch andere automatisierte Methoden wie zum Beispiel in Girindran et al. (2020) vorgestellt, müsste daher eine Überprüfung der topologischen Konsistenz im Sinne dieser Arbeit von vorne herein durchgeführt werden. Diese könnte beispielsweise durch Tools wie *CityDoctor* (Wewetzer et al., 2013) geschehen. Durch die Modularität des in Coors et al. (2020) beschriebenen Ansatzes ist es denkbar den Satz der Prüfbedingungen so zu erweitern, dass auch auf topologische Konsistenz im Sinne der hier vorliegenden Arbeit geprüft wird.

Darüber hinaus ist es erforderlich, Wege zu finden, um topologisch inkonsistente Daten zu heilen, möglicherweise abhängig von der Art der aufgetretenen topologischen Inkonsistenzen. Dies hat zum Ziel, nur topologisch konsistente Datensätze in topologischen Datenbanken zu speichern und weiter zu verwenden. Auch die verschiedenen Arten der Datenerhebung nach Art der topologischen Inkonsistenz zu unterscheiden, ist ein Thema der zukünftigen Arbeit. Die Aufgaben der Parallelisierung und des Streamings großer topologischer Datensätze sind ebenfalls Schritte für die Zukunft.

Insgesamt hat sich im Rahmen dieser Arbeit gezeigt, wie stark fehlerbehaftet reale CityGML-Datensätze derzeit nicht nur in Bezug auf topologische (In)Konsistenz häufig noch sind und dass noch einige Schwierigkeiten überwunden werden müssen um diese Datensätze für weitergehende Analysen sinnvoll nutzbar zu machen.

Literaturverzeichnis

(2019). ISO 19107:2019 Geographic information - Spatial schema.

3DCityDB (2018). 3DCityDB database schema for CityGML. `https://www.3dcitydb.org/3dcitydb/3dcitydbhomepage/`. Last visited on 15/05/2018.

Alam, N., Wagner, D., Wewetzer, M., von Falkenhausen, J., Coors, V., and Pries, M. (2014). *Towards Automatic Validation and Healing of CityGML Models for Geometric and Semantic Consistency*, pages 77–91. Springer International Publishing, Cham.

Alexandrov, P. S. (1937). Diskrete Räume. *Matematicheskii Sbornik (N.S.)*, 2:501–518.

Berlin Business Location Center (2018). Berlin 3D - Download Portal. `http://www.businesslocationcenter.de/en/downloadportal`. Last visited on 15/05/2018.

Biljecki, F. (2018). Random3Dcity - Synthetic CityGML data and procedural modelling engine. `https://3d.bk.tudelft.nl/biljecki/Random3Dcity.html`. Last visited on 15/05/2018.

Biljecki, F., Ledoux, H., Du, X., Stoter, J., Soon, K. H., and Khoo, V. H. S. (2016a). The most common geometric and semantic errors in CityGML datasets. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W1:13–22.

Biljecki, F., Ledoux, H., and Stoter, J. (2015a). *Improving the Consistency of Multi-LOD CityGML Datasets by Removing Redundancy*, pages 1–17. Springer International Publishing, Cham.

Biljecki, F., Ledoux, H., and Stoter, J. (2016b). GENERATION OF MULTI-LOD 3D CITY MODELS IN CITYGML WITH THEPROCEDURAL MODELLING ENGINE RANDOM3DCITY. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-4/W1:51–59.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., and Çöltekin, A. (2015b). Applications of 3D City Models: State of the Art Review. *ISPRS International Journal of Geo-Information*, 4(4):2842–2889.

Bradley, P. E. (2015). Supporting Data Analytics for Smart Cities: An Overview of Data Models and Topology. In Gammerman, A., Vovk, V., and Papadopoulos, H., editors, *Statistical Learning and Data Sciences SLDS 2015*, LNCS 9047, pages 406–413. Springer.

Bradley, P. E. and Paul, N. (2010). USING THE RELATIONAL MODEL TO CAPTURETOPOLOGICAL INFORMATION OF SPACES. *The Computer Journal*, Vol. 53:69–89.

Breunig, M., Bradley, P. E., Jahn, M., Kuper, P., Mazroob, N., Rösch, N., Al-Doori, M., Stefanakis, E., and Jadidi, M. (2020). Geospatial Data Management Research: Progress and Future Directions. *ISPRS International Journal of Geo-Information*, 9(2).

CGAL (2018). Computational Geometry Algorithms Library. `http://www.cgal.org/`. Last visited on 15/05/2018.

Chaturvedi, K., Yao, Z., and Kolbe, T. H. (2015). Web-based Exploration of and Interaction with Large and Deeply Structured Semantic 3D City Modelsusing HTML5 and WebGL. In *35. Wissenschaftlich-Technische Jahrestagung der DGPF*. Deutsche Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation e.V.

Coors, V., Betz, M., and Duminil, E. (2020). A Concept of Quality Management of 3D City Models Supporting Application-Specific Requirements. *PFG - Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88:3–14.

Cox, S., Daisey, P., Lake, R., Portele, C., and Whiteside, A. (2002). Geography Markup Language (GML) Encoding Specification.

Dušan, J. and Branislav, B. (2004). ELEMENTS OF SPATIAL DATA QUALITY AS INFORMATION TECHNOLOGY SUPPORT FOR SUSTAINABLE DEVELOPMENT PLANNING. *Spatium*, 11:88–83.

Egenhofer, M. J. (1991). Reasoning about binary topological relations. In Günther, O. and Schek, H.-J., editors, *Advances in Spatial Databases*, pages 141–160, Berlin, Heidelberg. Springer Berlin Heidelberg.

Eisentraut, P. (2003). *PostgreSQL: Das offizielle Handbuch*. verlag moderne industrie.

Foley, J. D., van Dam, A., Feiner, S. K., and Hughes, J. F. (1996). *Computer Graphics: Principles and Practice in C, 2nd Edition*. Addison-Wesley Professional.

GeodatenInfrastruktur Karlsruhe (GDI-KA) (2021). Geoportal Karlsruhe. `https://www.karlsruhe.de/b3/bauen/geodaten.de`. Last visited on 25/06/2021.

Giovanella, A., Bradley, P. E., and Wursthorn, S. (2018). Detection and evaluation of topological consistency in CityGML datasets. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-4:59–66.

Giovanella, A., Bradley, P. E., and Wursthorn, S. (2019). Evaluation of Topological Consistency in CityGML. *ISPRS International Journal of Geo-Information*, 8(6).

Girindran, R., Boyd, D., Rosser, J., Vijayan, D., Long, G., and Robinson, D. (2020). On the Reliable Generation of 3D City Models from Open Data. *Urban Science*, 4(4).

Gröger, G., Kolbe, T. H., Czerwinski, A., and Nagel, C. (2008). OpenGIS City Geography Markup Language (CityGML) Encoding Standard.

Gröger, G., Kolbe, T. H., Nagel, C., and Häfele, K.-H. (2012). OGC City Geography Markup Language (CityGML) Encoding Standard.

Gröger, G. and Plümer, L. (2011). How to achieve consistency for 3D city models. *Geoinformatica*, 15. Jg., Heft 1:137–165.

Gröger, G. and Plümer, L. (2012). CityGML - Interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71:12–33.

Hatcher, A. (2002). *Algebraic topology*. Cambridge University Press.

Hauenstein, T. (2017). Das 3D-Stadtmodell Karlsruhe: 3D-Daten für die kommunalen Aufgaben der Gegenwart und Zukunft.

Herring, J. R. (2010). OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 2: SQL option. Technical Report OGC 06-104r4, Open Geospatial Consortium Inc.

Jahn, M., Bradley, P., Al-Doori, M., and Breunig, M. (2017). TOPOLOGICALLY CONSISTENT MODELS FOR EFFICIENT BIG GEO-SPATIO-TEMPORAL DATA DISTRIBUTION. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-4/W5.

Jänich, K. (1990). *Topologie*. Springer-Verlag Berlin Heidelberg, 3 edition.

Kang, H.-K. and Li, K.-J. (2005). Assessing Topological Consistency for Collapse Operation in Generalization of Spatial Databases. In Akoka, J., editor, *Perspectives in Conceptual Modeling*, LNCS 3770, Berlin. Springer.

Kolbe, T. H., Yao, Z., Nagel, C., Redweik, R., Willkomm, P., Hudra, G., Müftüoglu, A., and Kunde, F. (2016). 3D City Database for CityGML, Version 3.3.0, Documentation.

Kunde, F. (2012). CityGML in PostGIS - Portierung, Anwendung und Performanz-Analyse am Beispiel der 3DCityDB Berlin. Masterarbeit, Universität Potsdam.

Kunde, F., Nagel, C., Herrerruela, J., Ross, L., and Kolbe, T. H. (2013). 3D-Stadtmodelle in PostGIS mit der 3D City Database. In *Tagungsband der FOSSGIS-Konferenz*.

Ledoux, H. (2013). On the validation of solids represented with the international standards for geographic information. *Computer-Aided Civil and Infrastructure Engineering*, 28(9):693–706.

Ledoux, H. (2018). val3dity: validation of 3D GIS primitives according to the international standards. *Open Geospatial Data, Software and Standards*, 3(1):1.

Ledoux, H., Biljecki, F., Dukai, B., Kumar, K., Peters, R., Stoter, J., and Commandeur, T. (2021). 3dfier: automatic reconstruction of 3D city models. *Journal of Open Source Software*, 6(57):2866.

Ledoux, H. and Meijers, M. (2011). Topologically consistent 3D city models obtained by extrusion. *International Journal of Geographical Information Science*, 25(4):557–574.

Li, S. (2006). On Topological Consistency and Realization. *Constraints*, 11(1):31–51.

Meijer, A. and Bolívar, M. P. R. (2016). Governing the smart city: a review of the literature on smart urban governance. *International Review of Administrative Sciences*, 82:392–408.

Paul, N. (2008). *Topologische Datenbanken für Architektonische Räume*. Dissertation, Universität Karlsruhe (TH).

Randell, D. A., Cui, Z., and Cohn, A. G. (1992). A Spatial Logic based on Regions and Connection. In *Principles of Knowledge Representation and Reasoning: Proceedings of the 1st International Conference*, pages 165–176.

Rodriguez, M. A., Brisaboa, N., Meza, J., and Luaces, M. R. (2010). Measuring Consistency with respect to Topological Dependency Constraints. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 182–191.

SFCGAL (2018). Simple Features wrapper library for CGAL. `http://www.sfcgal.org`. Last visited on 15/05/2018.

Stadler, A., Nagel, C., König, G., and Kolbe, T. H. (2009). Making interoperability persistent: A 3D geo database based on CityGML. In Lee, J. and Zlatanova, S., editors, *3D Geo-Information Sciences*, Lecture Notes in Geoinformation and Cartography, pages 175–192. Springer.

Steuer, H., Machl, T., Sindram, M., Liebel, L., and Kolbe, T. H. (2015). *Voluminator - Approximating the Volume of 3D Buildings to Overcome Topological Errors*, pages 343–362. Springer International Publishing, Cham.

TUDelft 3D geoinformation (2018). Open datasets created with 3dfier. `https://3d.bk.tudelft.nl/opendata/3dfier/`. Last visited on 15/05/2018.

Wagner, D., Alam, N., and Coors, V. (2013). *Geometric validation of 3D city models based on standarized quality criteria*, pages 197–210. CRC Press.

Wagner, D., Kolbe, T., and Coors, V. (2014). Spezifikation von Prüfplänen und Prüfergebnissen zur Validierung von 3D-Stadtmodellen. In Seyfert, E., Gülch, E., Heipke, C., Schiewe, J., and Sester, M., editors, *Publikationen der Deutschen Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation e.V.*, volume 23.

Wewetzer, M., von Falkenhausen, J., Wagner, D., Alam, N., Pries, M., Coors, V., and Fischer, J. W. (2013). *Verbundprojekt CityDoctor - Entwicklung von Methoden und Metriken zum Qualitätsmanagement virtueller Stadtmodelle*, pages 15–21. Logos Verlag, Berlin.

Zhou, X., He, H., Hou, D., Li, R., and Zheng, H. (2021). A Refined Lines/Regions and Lines/Lines Topological Relations Model Based on Whole-Whole Objects Intersection Components. *ISPRS International Journal of Geo-Information*, 10(1).

Anhang A

Detaillierte Ergebnisse für die übrigen Datensätze

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	46090 davon: 44878 2P. (97, 37%) konsistent: 33900 (75, 54%) inkonsistent: 10978 (24, 46%) 1150 3P. (2, 50%) 54 4P. (0, 12%) 2 7P., 9P., 10P., 15P. (0, 004%)	88, 01%
POINT	2380 davon: 100000000: 1624 (68, 24%) 010100000: 736 (30, 92%) 000010000: 20 (0, 84%) 001000100: 0 (0%)	4, 54%
MULTILINESTRING	1928	3, 68%
TIN	1770	3, 38%
GEOMETRYCOLLECTION	86	0, 36%
MULTIPOINT	60	0, 13%
TRIANGLE	54	0, 12%
Gesamt	6008 Polygone 8 gelöscht 20 (0, 33%) invalid 52368 Schnitte	

Tabelle A.1: Pariser Platz

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	239067 davon: 229268 2P. (95, 90%) konsistent: 157374 (68, 64%) inkonsistent: 71894 (31, 36%) 9534 3P. (3, 99%) 255 4P. (0, 11%) 3 5P. (0, 001%) 1 16P. 1 17P. 5 18P. (0, 002%)	70, 44%
POINT	64076 davon: 100000000: 43174 (67, 38%) 010100000: 19320 (30, 15%) 000010000: 1488 (2, 32%) 001000100: 86 (0, 13%) numerischer Fehler: 8 (0, 01%)	18, 88%
MULTILINESTRING	17179	5, 06%
TIN	13440	3, 96%
TRIANGLE	3222	0, 95%
GEOMETRYCOLLECTION	1898	0, 56%
MULTIPOINT	520	0, 15%
Gesamt	46145 Polygone 6427 (13, 93%) invalid 339402 Schnitte	

Tabelle A.2: Ettenheim

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	204386 davon: 198534 2P. (97, 14%) konsistent: 135234 (68, 12%) inkonsistent: 63300 (31, 88%) 5784 3P. (2, 83%) 3 4P. (0, 001%) 9 5P. (0, 004%) 16 6P. (0, 01%) 10 7P. (0, 005%) 18 8P. (0, 01%) 12 9P. (0, 01%)	67, 16%
POINT	78112 davon: 100000000: 42376 (54, 25%) 010100000: 34910 (44, 69%) 000010000: 810 (1, 04%) 001000100: 16 (0, 02%)	25, 67%
TIN	11084	3, 64%
MULTILINESTRING	8668	2, 85%
TRIANGLE	1381	0, 45%
GEOMETRYCOLLECTION	757	0, 25%
MULTIPOINT	18	0, 01%
Gesamt	18993 Polygone 64 invalid (0, 34%) 12 gelöscht 304343 Schnitte	

Tabelle A.3: Gebäudegruppe LoD3

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	3269 davon: 3180 2P. (97, 28%) konsistent: 2596 (81, 64%) inkonsistent: 584 (18, 36%) 86 3P. (2, 63%) 3 4P. (0, 09%)	71, 63%
POINT	1040 davon: 100000000: 796 (76, 54%) 010100000: 196 (18, 85%) 000010000: 48 (4, 62%) 001000100: 0 (0%)	22, 79%
MULTILINESTRING	159 davon: 97 2G. (61, 01%) 62 3G. (38, 99%)	3, 48%
TIN	80	1, 75%
TRIANGLE	16	0, 35%
Gesamt	745 Polygone 99 (13, 29%) invalid 4564 Schnitte	

Tabelle A.4: Lindenallee u. Kranichweg

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	2889 davon: 2788 2P. (96, 50%) konsistent: 2318 (83, 14%) inkonsistent: 470 (16, 86%) 94 3P. (3, 25%) 7 4P. (0, 25%)	81, 84%
POINT	452 davon: 100000000: 264 (58, 41%) 010100000: 188 (41, 59%) 000010000: 0 (0%) 001000100: 0 (0%)	12, 80%
MULTILINESTRING	157 davon: 83 2G. (52, 87%) 73 3G. (46, 50%) 1 4G. (0, 64%)	4, 45%
TIN	24	0, 68%
TRIANGLE	8	0, 23%
Gesamt	643 Polygone 48 (7, 45%) invalid 3530 Schnitte	

Tabelle A.5: Rheinstrasse

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	103164 davon: 103028 2P. (99, 87%) konsistent: 101970 (98, 97%) inkonsistent: 1058 (1, 03%) 130 3P. (0, 13%) 6 4P. (0, 01%)	85, 83%
POINT	14654 davon: 100000000: 14370 (98, 06%) 010100000: 284 (1, 94%) 000010000: 0 (0%) 001000100: 0 (0%)	12, 19%
TIN	2024	1, 68%
MULTILINESTRING	326	0, 27%
GEOMETRYCOLLECTION	16	0, 01%
MULTIPOINT	12	0, 01%
TRIANGLE	2	0, 002%
Gesamt	18510 Polygone (alle valid) 120198 Schnitte	

Tabelle A.6: Rintheim

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	2998 davon: 2930 2P. (97, 73%) konsistent: 2402 (81, 98%) inkonsistent: 528 (18, 02%) 67 3P. (2, 23%) 1 4P. (0, 03%)	67, 98%
POINT	1168 davon: 100000000: 954 (81, 68%) 010100000: 172 (14, 73%) 000010000: 42 (3, 60%) 001000100: 0 (0%)	26, 49%
MULTILINESTRING	100 davon: 84 2G. (84, 00%) 16 3G. (16, 00%)	2, 27%
TIN	98	2, 22%
TRIANGLE	44	1, 00%
GEOMETRYCOLLECTION	2 (2G.)	0, 05%
Gesamt	624 Polygone 60 (9, 62%) invalid 4410 Schnitte	

Tabelle A.7: Tennesseeallee

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	3315 davon: 3262 2P. (98, 40%) konsistent: 2722 (83, 45%) inkonsistent: 540 (16, 55%) 53 3P. (1, 60%)	65, 44%
POINT	1512 davon: 100000000: 1316 (87, 04%) 010100000: 190 (12, 57%) 000010000: 6 (0, 40%) 001000100: 0 (0%)	29, 85%
TIN	90	1, 77%
TRIANGLE	86	1, 70%
MULTILINESTRING	63 davon: 51 2G. (80, 95%) 12 3G. (19, 05%)	1, 24%
Gesamt	918 Polygone 176 (19, 17%) invalid 5066 Schnitte	

Tabelle A.8: Weidenweg

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	8648 davon: 8626 2P. (99, 75%) konsistent: 8486 (98, 38%) inkonsistent: 140 (1, 62%) 22 3P. (0, 25%)	95, 31%
POINT	320 davon: 100000000: 240 (75%) 010100000: 80 (25%) 000010000: 0 (0%) 001000100: 0 (0%)	3, 53%
TIN	60	0, 66%
MULTILINESTRING	46 (alle 2G.)	0, 51%
Gesamt	1547 Polygone (alle valid) 9074 Schnitte	

Tabelle A.9: Potsdam

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	21546 (alle 2P., alle konsistent)	86, 32%
POINT	3412 davon: 100000000: 3412 (100%) 010100000: 0 (0%) 000010000: 0 (0%) 001000100: 0 (0%)	13, 67%
MULTILINESTRING	4	0, 02%
Gesamt	6341 Polygone 452 (7, 13%) invalid 24962 Schnitte	

Tabelle A.10: Random3DCity LOD2_0_F0

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	28381 davon: 28236 2P. (99, 49%) (alle konsistent) 3 4P. (0, 01%) 142 5P. (0, 05%)	77, 31%
POINT	7642 davon: 100000000: 7642 (100%) 010100000: 0 (0%) 000010000: 0 (0%) 001000100: 0 (0%)	20, 82%
MULTILINESTRING	661	1, 80%
MULTIPOINT	26	0, 07%
Gesamt	7890 Polygone 876 (11, 10%) invalid 36710 Schnitte	

Tabelle A.11: Random3DCity LOD2_3_F0

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	5875 davon: 4800 2P. (81, 70%) konsistent: 2646 (55, 12%) inkonsistent: 2154 (44, 88%) 956 3P. (16, 27%) 116 4P. (1, 97%) 3 5P. (0, 05%)	67, 98%
MULTILINESTRING	1787	20, 68%
TIN	482	5, 58%
POINT	404 davon: 100000000: 404 (100%) 010100000: 0 (0%) 000010000: 0 (0%) 001000100: 0 (0%)	4, 67%
TRIANGLE	92	1, 06%
GEOMETRYCOLLECTION	2	0, 02%
Gesamt	679 Polygone 3 (0, 44%) invalid 8642 Schnitte	

Tabelle A.12: Random3DCity LOD2_overlap

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	4883 davon: 4696 2P. (96, 17%) (alle konsistent) 2 4P. (0, 04%) 185 5P. (3, 79%)	59, 46%
POINT	2096 davon: 100000000: 2096 (100%) 010100000: 0 (0%) 000010000: 0 (0%) 001000100: 0 (0%)	25, 52%
MULTILINESTRING	1171	14, 26%
TIN	34	0, 41%
MULTIPOINT	24	0, 29%
GEOMETRYCOLLECTION	4	0, 05%
Gesamt	40263 Polygone 4553 (11, 31%) invalid 8212 Schnitte	

Tabelle A.13: Random3DCity LOD3_2

Geometrietyp Schnitt	**Anzahl**	**Anteil**
LINESTRING	17487 davon: 17058 2P. (97, 55%) konsistent: 15862 (92, 99%) inkonsistent: 1196 (7, 01%) 407 3P. (2, 32%) 18 4P. (0, 10%) 4 5P. (0, 02%)	84, 71%
MULTILINESTRING	1255 davon: 908 2G. (72, 35%) 335 3G. (26, 69%) 12 4G. (0, 96%)	6, 08%
POINT	1206 davon: 100000000: 1164 (96, 52%) 010100000: 42 (3, 48%) 000010000: 0 (0%) 001000100: 0 (0%)	5, 84%
TRIANGLE	328	1, 59%
TIN	294	1, 42%
GEOMETRYCOLLECTION	74 davon: 56 2G. (75, 68%) 14 3G. (18, 92%) 4 4G. (5, 41%)	0, 36%
Gesamt	4038 Polygone 259 (6, 41%) invalid 20644 Schnitte	

Tabelle A.14: Waldbrücke

Eidesstattliche Versicherung

Eidesstattliche Versicherung gemäß § 6 Abs. 1 Ziff. 4 der Promotionsordnung des Karlsruher Instituts für Technologie für die Fakultät für Bauingenieur-, Geo- und Umweltwissenschaften

1. Bei der eingereichten Dissertation zu dem Thema
„Automatische redundanzfreie Generierung von Topologie aus multidimensionalen geographischen Datenbeständen“
handelt es sich um meine eigenständig erbrachte Leistung.

2. Ich habe nur die angegebenen Quellen und Hilfsmittel benutzt und mich keiner unzulässigen Hilfe Dritter bedient. Insbesondere habe ich wörtlich oder sinngemäß aus anderen Werken übernommene Inhalte als solche kenntlich gemacht.

3. Die Arbeit oder Teile davon habe ich wie bislang nicht an einer Hochschule des In- oder Auslands als Bestandteil einer Prüfungs- oder Qualifikationsleistung vorgelegt.

4. Die Richtigkeit der vorstehenden Erklärungen bestätige ich.

5. Die Bedeutung der eidesstattlichen Versicherung und die strafrechtlichen Folgen einer unrichtigen oder unvollständigen eidesstattlichen Versicherung sind mir bekannt.

Ich versichere an Eides statt, dass ich nach bestem Wissen die reine Wahrheit erklärt und nichts verschwiegen habe.

Ort und Datum Unterschrift