Kartographische Darstellung semantisch strukturierter 2D-Daten in einer 3D-Szene

STEFFEN HEMPEL[1], JOACHIM BENNER[2], KARL-HEINZ HÄFELE[3], ANDREAS GEIGER[4]

Es wird eine allgemeine Methode zur kartographischen Visualisierung verschiedener semantischer Datenformate in einer 3D-Szene beschrieben. Diese Methode transformiert unterschiedliche Datenformate wie XPlanGML oder OpenStreetMap in ein für die Darstellung geeignetes Format und stellt diese mit Hilfe hinterlegter Darstellungsvorschriften dar. Somit wird es ermöglicht, Geometrieobjekte mit 3D-Renderern kartographisch anspruchsvoll darzustellen. Die vorgestellte Methode erlaubt die Integration von 2D- und 3D-Daten.

1 Einleitung

Raumbezogene Daten in semantischen Datenformaten beinhalten in der Regel keine vollständigen Informationen, wie sie kartographisch darzustellen sind. Daher müssen für die kartographische Darstellung passende Darstellungsvorschriften zugeordnet werden. Diese Darstellungsvorschriften müssen verschiedene Symbole, Linienarten, Randsignaturen, Schraffuren und Flächensignaturen unterstützen. Beispielsweise lässt sich XPlanGML (BENNER et al. 2007) nach Darstellungsvorschriften basierend auf der Planzeichenverordnung 1990 (PLANZV 1990) darstellen.

Wird für eine bestimmte Auswertung ein Verschnitt von 2D- und 3D-Daten benötigt und müssen diese Daten zusammen visualisiert werden, dann müssen 2D-Daten von einem 3D-Renderer dargestellt werden.

Eine solche gemeinsame Auswertung erlaubt es neue Anwendungsfelder für die öffentlich verfügbaren Geodaten zu erschließen. Ein Beispiel dafür ist die Integration von Katasterdaten (Alkis/NAS), digitalen Bebauungsplänen (XPlanGML) und 3D-Stadtmodellen (CityGML) zur Unterstützung eines digitalen Bauantragsverfahrens (BENNER et al. 2012).

[1] Steffen Hempel, Institut für Angewandte Informatik, KIT, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen
[2] Joachim Benner, Institut für Angewandte Informatik, KIT, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen
[3] Karl-Heinz Häfele, Institut für Angewandte Informatik, KIT, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen
[4] Andreas Geiger, Institut für Angewandte Informatik, KIT, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen

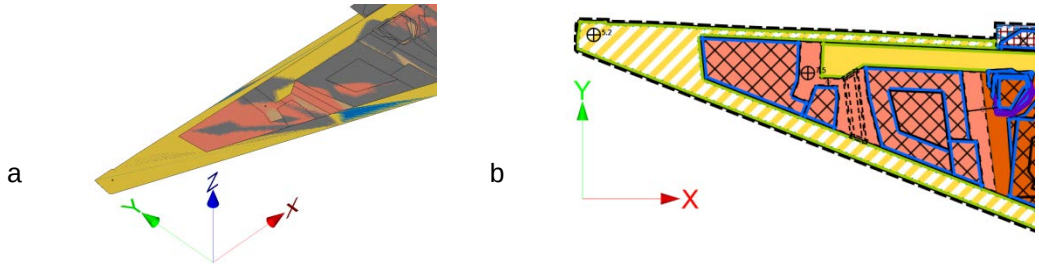

Abb. 1: 3D-Darstellung und 2D-Darstellung (Quelle: LGV der FHH)

Abbildung 1 zeigt zwei verschiedene Darstellungen desselben XPlanGML-Datensatzes, der einen Bebauungsplan der Hamburger HafenCity repräsentiert. Für Abbildung 1a wurden sehr einfache Darstellungsvorschriften verwendet. Alle Flächen-, Linien- und Punktobjekte werden ohne Berücksichtigung der Überlagerung von Objekten in einer klassenspezifischen Farbe dargestellt. Derartige Darstellungen lassen sich mit einem 3D-Renderer einfach erzeugen. Sie sind aber für eine problemgerechte Darstellung nicht geeignet.

In Abbildung 1b wird der Inhalt dieses Planes mit 2D-Vektorgrafik unter Benutzung der Darstellungsvorschriften der Planzeichenverordnung angezeigt. Diese Darstellung ist kartographisch weitaus aussagekräftiger, lässt sich aber mit einem 3D-Renderer nicht ohne weiteres erzeugen.

In diesem Beitrag wird eine allgemeine Methode beschrieben, derartige kartographische Darstellungen für unterschiedliche semantische Datenformate in einer 3D-Szene zu erzeugen.

2 Konzept

Für die 3D-Darstellung semantischer, raumbezogener Daten unter Berücksichtigung von komplexen Darstellungsvorschriften werden die in Abbildung 2 dargestellten Schritten durchlaufen.

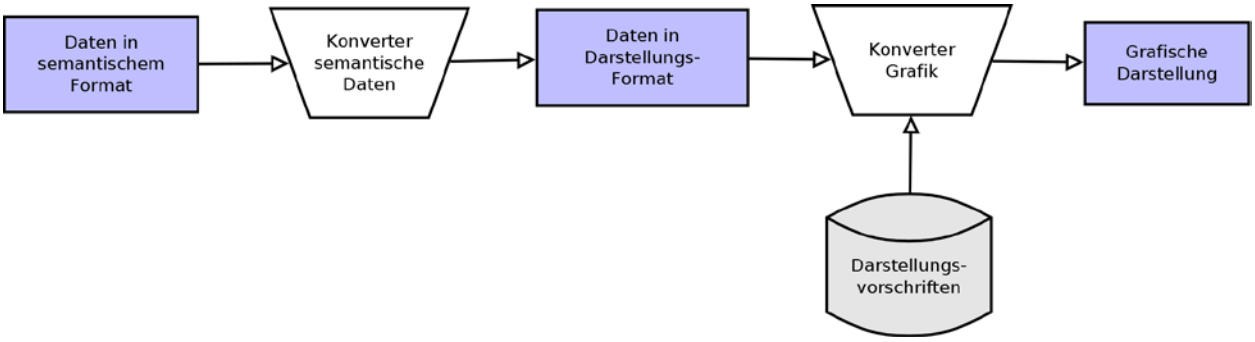

Abb. 2: Schematische Darstellung der notwendigen Verabeitungsschritte

Als Vorverarbeitung müssen die Objekte des Ausgangsdatensatzes (Daten in semantischem Format) anhand ihrer Klassenzugehörigkeit und Attribute in eine für die Darstellung besser geeignete Repräsentation (Darstellungsformat) transformiert werden (siehe Abbildung 2 „Konverter semantische Daten“). Die Darstellungsobjekte werden anhand zugeordneter

Darstellungsvorschriften zu geometrischen Primitiven im 3D-Raum umgesetzt (siehe Abbildung 2 „ Konverter Grafik“).

Im Folgenden wird das Darstellungsformat, sowie die entsprechende Transformation präsentiert.

3 Darstellungsklassen

Die Transformation von Objekten der semantischen Klassen in Objekte der Darstellungsklassen erlaubt die Darstellung verschiedener semantischer Klassen, ohne für jede einzelne dieser semantischen Klassen eine eigene Darstellungsmethode entwickeln zu müssen.

Für diese Transformation werden zuerst die gemeinsamen Eigenschaften semantischer Klassen aufgezeigt. Aus diesen gemeinsamen Eigenschaften kann der Aufbau der Darstellungsklassen bestimmt werden. Ist der Aufbau der Darstellungsklassen festgelegt, kann bestimmt werden, wie die Transformation von Objekten der semantischen Klassen in Objekte der Darstellungsklassen aussehen muss.

3.1 Definition der Darstellungsklassen

Als Vorbild für die Struktur der Darstellungsklassen dient OpenStreetMap XML (OPENSTREETMAP COMMUNITY 2014). OSM XML ist allgemein formuliert und bietet viele Freiheiten, wie zum Beispiel das Hinzufügen beliebiger kartographischer Objekte. Bereits implementierte Renderer zeigen, dass es zur direkten grafischen Darstellung geeignet ist; allerdings würden diese Renderer natürlich an unbekannten kartographischen Objekten scheitern, die aus anderen semantischen Datenformaten übernommen wurden.

OSM XML definiert „Knoten“, „Relationen“ und „Tags“. Eine Relation ist eine geordnete Aggregation von Knoten, die beispielsweise einen offenen oder geschlossener Linienzug oder ein ausgefülltes Polygon beschreibt. Tags werden verwendet, um zusätzliche semantische Information, wie Objekttyp, Zweckbestimmungen oder URLs, zu einem Element hinzuzufügen (RAMM et al. 2010).Diese Tags sind Schlüssel-Wert-Paare und frei definierbar. Es gibt jedoch eine umfangreiche Liste von sogenannten Features als vordefinierte Schlüssel mit entsprechenden Einschränkungen für ihre Werte.

Die für das hier präsentierte Konzept definierten Darstellungsklassen sind die in Abbildung 3 dargestellten Klassen `Relation` und `Node`.

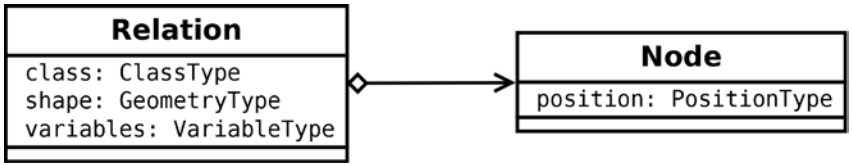

Abb. 3: UML-Diagramm der Darstellungsklassen

Teil jeder `Relation` ist die benötigte Geometrie, sowie das Attribut `class` welches bestimmt, von welcher Art (Straße, Grünfläche, …) das beschriebene Objekt ist. Die Ausprägung der Geometrie wiederum setzt sich aus der Art der geometrischen Form (`shape`: Linie, Fläche, Punkt), sowie aus den zugehörigen Knoten (`Node`) zusammen.

Eine Position innerhalb einer Karte wird als Knoten (`Node`) gespeichert und kann von mehreren `Relation` geteilt werden. So ist es möglich komplexere, sich verzweigende Elemente, aus mehreren `Relation` zusammenzusetzen (zum Beispiel eine Kreuzung), oder Übergängen zwischen `Relation` verschiedener Art einen gemeinsamen Übergangspunkt zu geben (zum Beispiel eine Autobahn, von der eine Abfahrt abgeht).

Im Attribut `variables` werden Texte als Schlüssel-Wert-Paare vorgehalten, die als Zusatzangabe ergänzend dargestellt werden sollen (Beispielsweise Straßennamen).

3.2 Beschreibung der Transformation

Ein semantisches Objekt wird in ein Darstellungsobjekt transformiert. Dabei werden darstellungsrelevante Informationen in das Attribut `class`, geometrische Informationen in das Attribut `shape` und in `Node`-Objekte, sowie darstellungsrelevante Texte in das Attribut `variables` abgebildet.

Der Wert des Attributs `class` setzt sich aus dem Typ der semantischen Klasse selbst und optional aus Attributwerten zusammen. Der Wert von `class` wird verwendet, um eine Darstellungsvorschrift zuzuordnen.

Eine Rücktransformation von Darstellungsklassen in semantische Klassen ist nicht mehr möglich, da für die Darstellung nicht relevante semantische Informationen verloren gehen.

Ausgehend von den semantischen Formaten werden die Objekte der semantischen Klassen jeweils von einem spezifischen Konverter in das Zwischenformat der Darstellungsklassen transformiert.

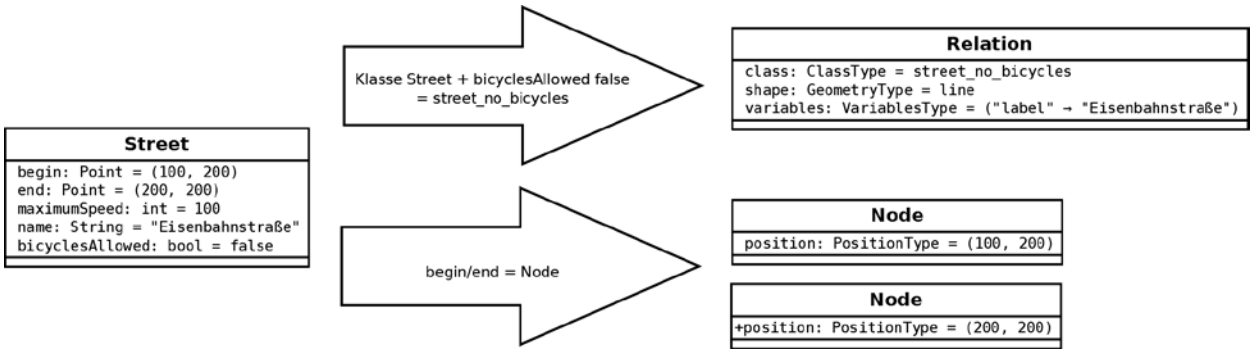

Abb. 4: Transformation einer einzelnen semantischen Instanz

Ein Beispiel für eine solche Transformation findet sich in Abbildung 4. Ein komplexes Element vom Typ `Street` wird in Instanzen der allgemeinen Struktur der Darstellungsklassen `Relation` mit zwei `Node`-Objekten transformiert. Hierbei ist insbesondere der häufig vorkommende Fall zu beachten, dass die Art der `Relation`, beschrieben vom Attribut `class`, sowohl aus der Klasse, als auch aus den Attributen des Ausgangsformats zusammengesetzt wird (`Klasse Street + bicyclesAllowed false = street_no_bicycles`). Der Wert des Attributs `name` der Klasse `Street` wird mit dem Schlüssel `label` in das Attribut

`variables` der `Relation` übernommen. Ob und wie der Wert `“Eisenbahnstraße“` dargestellt wird hängt von der Darstellungsvorschrift (siehe Kapitel 4.1) ab.

Durch die Transformation erhält man die Möglichkeit auf den Objekten der Darstellungsklassen Algorithmen anzuwenden, die neue Informationen erzeugen. Ein Beispiel für einen solchen Algorithmus ist die Umwandlung in eine polygonale Darstellung für OpenGL.

Es können jetzt auch neue semantische Klassen hinzugefügt werden, es ist aber nicht notwendig neue Darstellungsklassen und damit neue Darstellungsalgorithmen zu schreiben. Stattdessen wird lediglich die Transformation von Objekten der semantischen Klassen in Objekte der Darstellungsklassen durchgeführt.

4 Darstellung in einer 3D-Szene

Für die grafische Darstellung werden Darstellungsvorschriften in Form grafischer Teilelemente auf das Attribut `class` einer `Relation` abgebildet. Hierbei lässt sich auch eine Darstellungsvorschrift mehrfach für verschiedene Attributwerte von `class` verwenden.

Nun können alle Instanzen von `Relation` in die eigentliche grafische Darstellung übersetzt werden. Für die Darstellung gibt es zwei Möglichkeiten:

- Das Rendern auf eine Textur
- Das Erzeugen einer polygonalen Darstellung

Eine Darstellung als Textur kommt aus verschiedenen Gründen nicht in Frage:

- Die Darstellung soll auch bei starker Vergrößerung nicht verzerrt werden. Das ist insbesondere dann wichtig, wenn sehr feine Strukturen verwendet werden, die nur in der Vergrößerung sichtbar werden.
- Einzelne Objekte sollen im 3D-Raum auswählbar sein.
- Bestimmte Arten von Objekten sollen unabhängig voneinander sichtbar und unsichtbar gemacht werden.
- Echte 3D-Geometrie kann mit Postprocessing-Algorithmen bearbeitet werden, um wünschenswerte Effekte wie beispielsweise eine Hervorhebung zu erzeugen.

4.1 Beschreibung grafischer Teilelemente

Grafische Teilelemente beschreiben verschiedene Strukturen wie dekorierte Linien, Flächen und Symbole. Dabei ist die Beschreibung eines grafischen Teilelements so allgemein gehalten, dass alle darzustellenden Signaturen oder Symbole erzeugt werden können.

Dies wird erreicht, indem man über einfache grafische Flächenelemente (Kreis, gefüllte Polygone, Linien und Text) einen sich nicht wiederholenden Teilausschnitt eines grafischen Elements modelliert. Dieses grafische Teilelement wird Prototyp genannt.

Die Einfachheit von Prototypen erlaubt es, diese problemlos in Geometrie für 3D-Darstellung umzuwandeln.

In Abbildung 5 findet sich eine aus vier hintereinander gelegten Prototypen zusammengesetzte Linie. Ein einzelner Prototyp ist hervorgehoben.

Prototypen erfüllen bestimmte Eigenschaften:

- Sie sind einfach zusammensetzbar, um sich wiederholende Muster und eine Platzierung entlang einer Linie zu ermöglichen.
- Sie sind skalier-, verschieb- und drehbar.

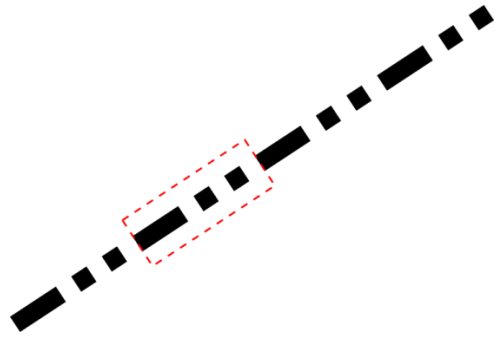

Abb. 5: Prototyp

Die Positionierungswerte für die Teile eines Prototyps werden in einem normierten Koordinatensystem (zum Beispiel von 0 bis 1 oder -1 bis 1) angegeben, sodass eine Skalierung anhand äußerer Faktoren (zum Beispiel eines Feinheitsfaktors) ermöglicht wird.

Ein Spezialfall in einer Prototyp-Beschreibung ist das Einfügen von Text. Es können statische Texte dargestellt werden (wie beispielsweise das P innerhalb eines Parkplatz-Symbols) oder Texte die mit einem Platzhalter versehen sind. Dieser Platzhalter wird aus den Variablen (`variables`, siehe Kapitel 3.1) einer `Relation` befüllt.

4.2 Zeichnen von Prototypen

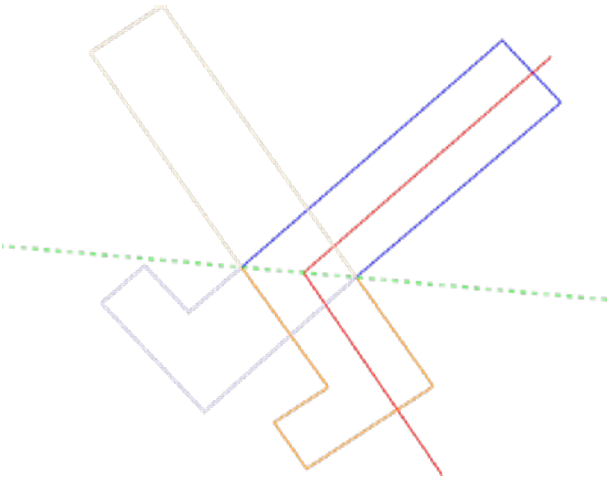

Abb. 6: Zurechtschneiden von Prototypen für Linienzüge

In Abhängigkeit des Attributs `shape` (Punkt, Linie, Fläche) der Klasse `Relation` werden die Prototypen unterschiedlich angewendet.

Die einfachste Möglichkeit ist das Zeichnen von Prototypen als Punktobjekt. Hierbei wird der Prototyp an die passenden Koordinaten verschoben, skaliert und dort gezeichnet.

Für das Rendering eines Linienzugs werden in die entsprechende Richtung gedrehte Prototypen auf dem Linienzug hintereinander gelegt. In der Regel passt die Vervielfältigung von Prototypen nicht zur Linienlänge. Um dieses Problem zu lösen, müssen Prototypen so auseinandergeschnitten werden, dass diese wieder aneinander gelegt werden können. Abbildung 6 zeigt diesen Schnitt als gestrichelte Linie.

Um ein Flächenobjekt zu zeichnen wird zuerst eine das Flächenobjekt umgebende Bounding-Box erzeugt. Diese Bounding-Box wird mit neben- und untereinander gesetzten Prototypen

gefüllt. Danach wird eine boolesche Schnittoperation angewandt, um das entstandene Rechteck an die Form der benötigten Fläche anzupassen.

5 Ergebnisse

Bisher wurden die zwei Formate XPlanGML und OpenStreetMap implementiert. Für beide Formate wurden entsprechende Prototypen anhand der Darstellungsvorschriften entworfen.

In Abbildung 7 findet sich die in der Einleitung dargestellte XPlanGML-Karte als 3D-Rendering links mit einfachen geometrischen Flächen, gegenübergestellt zum 3D-Rendering rechts mit einer kartographischen Darstellung.

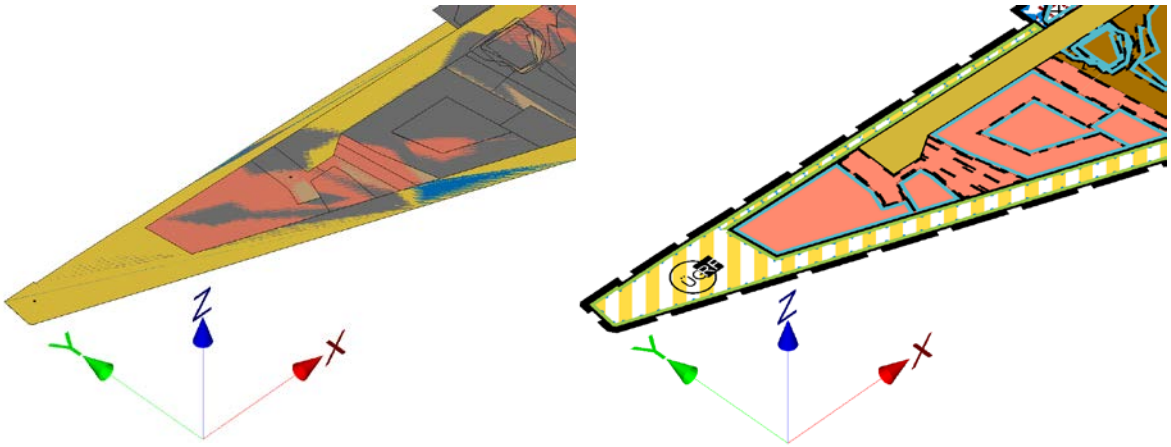

Abb. 7: Primitive 3D-Darstellung und 3D-Darstellung mit Signaturen (Quelle: LGV der FHH)

Abbildung 8 zeigt verschiedene Ansichten auf den gleichen Ausschnitt aus OSM-Daten. Der linke Bildausschnitt zeigt die Features als Liniendarstellung. Im mittleren Ausschnitt wurde das Rendering der Karte von einem Map-Server heruntergeladen. Die Features aus dem linken Bildausschnitt wurden im rechten Bildausschnitt mit hinterlegten Darstellungsvorschriften für OSM als 3D-Geometrie gerendert.

Abb. 8: OSM Features, OSM Mapservice-Rendering, OSM-Features mit Signaturen

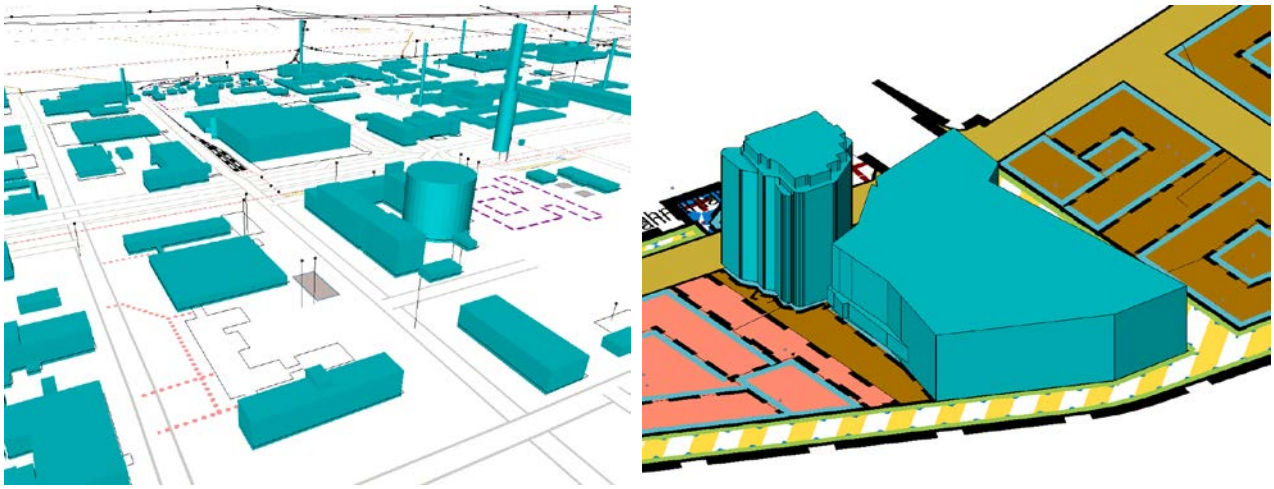

Abb. 9: 2D- und 3D-Daten in einer Szene (links OSM, CityGML und BoreholeML, rechts XPlanGML und CityGML [Quelle: LGV der FHH])

Schließlich ist in Abbildung 9 links die Integration von Daten aus CityGML, BoreholeML und OSM dargestellt. Bestimmte Features für OSM wie Grasflächen wurden für diese Darstellung der Übersichtlichkeit wegen deaktiviert. Auf der rechten Seite werden CityGML- und XPlanGML-Daten zusammen dargestellt.

6 Fazit

Die in diesem Beitrag beschriebene Methode ermöglicht eine Visualisierung von 2D-Daten in einer 3D-Szene unter Berücksichtigung der dazugehörigen Darstellungsvorschriften. Dabei werden für jedes Objekt des Ausgangsdatensatzes, unter Berücksichtigung aller darstellungsrelevanten Attribute, Darstellungsobjekte erzeugt. Das Rendering der Darstellungsobjekte erfolgt mit Hilfe von Prototypen. Zur Erzeugung von Prototypen wurde ein Editor entwickelt, mit dem eine Prototypen-Bibliothek erzeugt werden kann.

Die beschriebene Methode wurde als Machbarkeitsstudie für die Software IFCExplorer (BENNER et al. 2013) des KIT implementiert. Der Funktionsnachweis wurde mit den Formaten XPlanGML und OpenStreetMap erbracht. Als nächstes soll eine prototypische Implementierung erfolgen, um XPlanGML möglichst vollständig zu unterstützen.

7 Literaturverzeichnis

BENNER, J., KRAUSE, K.-U., 2007: XPlanung – Ein GIS-Standard zum Austausch digitaler Bauleitpläne. In: Flächenmanagement und Bodenordnung (fub), Band 6/2007, S. 274 – 280, 2007.

BENNER, J., HÄFELE, K.-H., GEIGER, A., 2012: Integration raumbezogener Daten in einer CityGML Application Domain Extension (ADE) zur Unterstützung des digitalen Bauantragsverfahrens. In: Geoinformatik 2012 „Mobilität und Umwelt“, 28. – 30. März 2012, Braunschweig, S. 163 – 170.

BENNER, J., GEIGER, A., HÄFELE, K.-H., KNÜPPEL, H., 2013: IFCExplorer – Ein Werkzeug für die Integration unterschiedlicher raumbezogener semantischer Daten, GeoInformatik 2013, Heidelberg

PLANZV 1990, 1990: Planzeichenverordnung 1990 - PlanzV

OPENSTREETMAP COMMUNITY, 2014: *OSM XML - OpenStreetMap Wiki*, http://wiki.openstreetmap.org/wiki/OSM_XML, zuletzt aufgerufen 31.01.2014

RAMM F., TOPF J., 2010: OpenStreetMap – Die freie Weltkarte nutzen und mitgestalten, Lehmanns Media, 3. Auflage, 2010