

Verbesserung der Modellierung grafischer Attribute in CityGML

VORSCHLAG

Haik Lorenz und Dirk Dörschlag

Motivation: Bisherige Grenzen

2

- TriangulatedSurface/TIN/Grid kann nicht gefärbt oder texturiert werden
- Keine georeferenzierten Luftbilder
- Keine klassenweiten Farbeinstellungen
- Keine alternativen grafischen Darstellungen



Motivation: Bisherige Schwächen

3

- Verwendung neuer Geometrieklasse (TexturedSurface) für Färbung
 - Klassenkonvertierung bei Einfärbung nötig
- Implizite Vermischung mehrerer Geometrieobjekte bei Anwendung von Texturierung
- 1:1-Relation zw. Textur und Texturkoordinaten
 - Jede Fläche braucht eigene Textur mit allen Attributen (Linking nicht möglich)
- Häufig Materialduplikate, da gesamtes Material pro Fläche angegeben wird (Linking nicht genutzt)

Ziele für neuen Entwurf

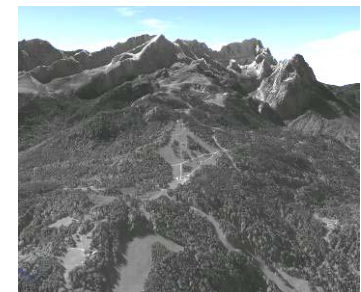
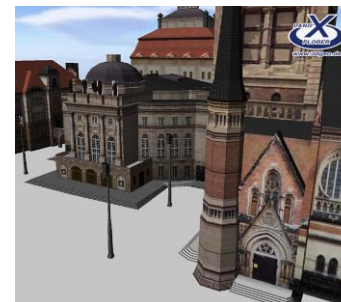
4

- Komplettierung der grafischen Attributierung
- **ABER: Beschränkung auf häufig gebrauchte Anwendungsfälle** – klare Abgrenzung zu CG-Formaten wie X3D, Collada, U3D
- Flexibilität bei geringem Schreibaufwand
- Einfache und definierte Konvertierung nach/aus Fremdformaten
- Verständlichkeit

Anwendungsfälle [1/2]

5

1. Geometrie ohne grafische Attribute
2. Geometrie mit klassenspezifischen (und LoD-abhängigen) Farben (z.B. rote Dächer, weiße Wände, transparente Fenster und blaues Wasser)
3. Gefärbte und texturierte Geometrie
4. Georeferenziertes Luftbild auf Gelände (Grid/TIN)



Anwendungsfälle [2/2]

6

5. Mehrere Darstellungsvarianten
(z.B. realistische Erscheinung
und Wärmebild)
6. Vorberechnete Beleuchtung
abspeichern und mit anderen
Darstellungsvarianten
kombinieren
7. LoD-abhängige Texturen



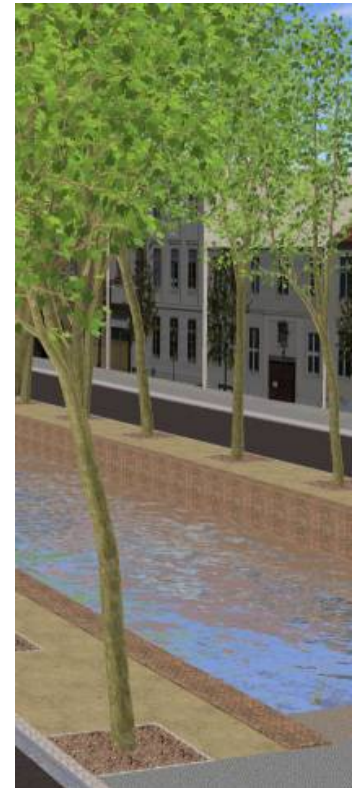
Beleuchtung

- Sehr komplexes Thema
→ Licht nicht in CityGML spezifizieren
- Viewer übernimmt Beleuchtung nach eigenen Möglichkeiten
- Konkrete Beleuchtungsergebnisse können jedoch abgespeichert werden
- Materialien sehr stark von konkreten Beleuchtungsverhältnissen abhängig
→ ohne definierte Beleuchtung präzise Materialien nicht sinnvoll



Spezialeffekte

- Nicht berücksichtigen, da plattformabhängig sehr unterschiedliche Umsetzungen oder zusätzliche Daten nötig
- Z.B. Bumpmapping: Je nach verfügbarer Technologie unterschiedliche Algorithmen mit unterschiedlicher Qualität und Eingabedaten
- Z.B. Environmentmapping: Als Spiegelungersatz (z.B. in Wasser) unbrauchbar, da kein Himmel definiert wird



Multitexturing

- Dient hauptsächlich zur Umsetzung von Spezialeffekten / Ersatz für programmierbare Effekte
- Beschränkung auf wirklich notwendige Varianten

Programmierbare Effekte

- noch kein gemeinsamer Standard → sehr plattformspezifisch, ständige Erweiterung der technischen Möglichkeiten
- Nicht in CityGML aufnehmen

Werden CityGML-Darstellungen langweilig?

NEIN!

- Aber: Viewer muß visuelle Details aus Semantik ableiten und einfügen
- Z.B. qualitativ hochwertige Beleuchtung, reflektierende Fenster, spiegelnde bewegte Wasseroberflächen, spezialisierte Beleuchtung von Bäumen, Schatten...
- CityGML liefert Basisdaten wie Fassadentexturen, Luftbilder, Grundfarben



Möglichkeiten unseres Vorschlags

- Universelle Texturierung und Färbung
- Einfache Materialien (z.B. zur Unterscheidung Glas – Beton)
- Klassenweite Farben/Materialien
- LoD-abhängige Texturen und Farben
- Georeferenzierung bzw. Texturkoordinatengenerierung
- Mehrere (beliebig kombinierbare) grafische Darstellungen
- Separierte Texturen (z.B. Backsteintextur plus Fenster)
- Separate Verschmutzungen und vorberechnete Beleuchtung

Konzepte

12

- **Attributbibliotheken:** Grafische Attribute werden in CityGML-Features gesammelt
- **Reverse Linking:** Grafische Attribute müssen Geometrie, auf die sie angewendet werden, referenzieren
- **Layer:** Definition mehrerer, einzeln zuschaltbarer Attributbibliotheksgruppen
- **Standardattribute:** Bindung von Farben/Materialien an Featureklassen
- **Texturierung:** Trennung zw. wiederverwendbarer Textur und geometriespezifischen Texturkoordinaten

Konsequenzen

- Geometrie in 2 Hierarchien eingebettet:
Semantik und Attributbibliotheken
- Jedes attributierte Objekt braucht eindeutige
gml:id (dateiübergreifend, wenn 2 Dateien
zusammengeführt werden)
- Ein (möglicher) zukünftiger GML-Standard
zur grafischen Attributierung kann direkt
übernommen werden
- Dateigröße texturierter Modelle schrumpft
deutlich (teilweise auf unter 50%)

Relation zu Fremdformaten

14

- Konvertierung von CityGML in andere Formate unkritisch
 - Layer können Probleme bereiten
- Konvertierung nach CityGML meist verlustbehaftet (insb. Beleuchtung)
 - Bewußte Entscheidung gegen komplexe Beleuchtungsbeschreibung
 - Grundstimmungen können übertragen werden; speziell designte Lichtverhältnisse nur mittels vorberechneter Beleuchtung
 - Für verbreitete Fremdformate (X3D, Collada, U3D, 3DS) Konvertierungsrichtlinien möglich

Offene Fragen

- Wo sollen grafische Attribute eingebettet werden?
- Wie präzise sollen Materialien spezifiziert werden?
- Wie werden mehrere Layer grafisch verknüpft?
- Soll es Layergruppen geben?
- Sollen Informationen zur optischen Glättung von Polygonnetzen eingebettet werden?
- ...

Wie gehts weiter?

- TexturedSurface soll komplett ersetzt werden
 - Als deprecated erklären und in CityGML 1.0 ganz entfernen
- Modellierung weit fortgeschritten
- Heute: Diskussion technischer Details in der AG Modellierung



Danke!

Fragen, Kritik & Anregungen sind herzlich willkommen!

haik.lorenz@hpi.uni-potsdam.de
dirk.doerschlag@ikg.uni-bonn.de

Dank an Henrik Buchholz, Prof. Kolbe, Prof. Döllner

17.11.2006