Zusammenführung und interaktive Analyse von hochauflösenden 3D-Geodaten

Bernd LEITNER, Gerd HESINA und Friedrich BRIMMER

Dieser Beitrag wurde nach Begutachtung durch das Programmkomitee als „reviewed paper" angenommen.

Zusammenfassung

Aufgrund des üblicherweise enormen Umfangs von Laserscandaten ist deren Visualisierung und Bearbeitung nach wie vor eine anspruchsvolle Aufgabe. Neben speicherbedingten Limitierungen handelsüblicher Personal Computer ist auch die interaktive Darstellung großer Datenmengen eine der größten Herausforderungen.

In dieser Arbeit präsentieren wir eine in drei Schichten gegliederte, prototypenhafte Visualisierungspipeline für LIDAR Daten. Der Prototyp erlaubt eine Zusammenführung und Homogenisierung von Geodaten unterschiedlicher Art und Herkunft. Um interaktive Frameraten zu gewährleisten, verwendet die Pipeline multi-resolution Renderingmethoden und nutzt Out-of-Core Datenstrukturen zur Verwaltung großer Datenmengen. Die vorgestellte Herangehensweise dient als Basis für die Entwicklung weiterer interaktiver Analysewerkzeuge und ist zur Planung von Infrastrukturprojekten, als Werkzeug für Dokumentation und Qualitätsmanagement oder zur Unterstützung von Nofallsystemen gedacht.

1 Einleitung

Die Bearbeitung und Visualisierung von hochauflösenden Daten, wie sie unter anderem durch LIDAR (Light Detection and Ranging) gewonnen werden, stellen für die Entwicklung entsprechender Anwendungen nach wie vor eine große Herausforderung dar. Selbst ein aktuelles Computersystem mit beispielsweise 8 GB Arbeitsspeicher, sowie einer modernen Grafikkarte mit 1 GB Grafikspeicher, ist mit der Darstellung einer Punktwolke mit einer Größe von mehreren Hundert Mio. Punkten bei weitem überfordert.

Allerdings vermitteln gerade diese umfangreichen Datensätze ein glaubhaftes und genaues Abbild der Umwelt. Aus diesem Grund wurde in den letzten Jahren vermehrt nach Methoden gesucht, um hochauflösende LIDAR-Daten effektiv mit anderen Datenquellen kombinieren zu können (wie zum Beispiel: SCHENK & CSATHÓ 2007, MCGAUGHEY & CARSON 2003). Besonders interessant ist hierbei die Zusammenführung von Punktwolken mit Daten aus vorhandenen GI Systemen, Straßen- und Schienennetzen sowie die automatische Triangulierung und Texturierung ungeordneter 3D-Punkte. Eine Applikation, die es erlaubt solche Geodaten zu fusionieren, kann sich beispielsweise bei der Planung von Infrastrukturprojekten oder zur Unterstützung von Notfallsystemen als hilfreiches Werkzeug erweisen.

Strobl, J., Blaschke, T. & Griesebner, G. (Hrsg.) (2011): Angewandte Geoinformatik 2011.

Wie bereits zuvor erwähnt, bringt jedoch bereits eine umfangreiche Punktwolke einen modernen PC an seine Grenzen. Wie kann ein solcher Datensatz nun in Kombination mit weiteren Daten visualisiert werden? Eine häufige Herangehensweise, um diesem Problem entgegenzuwirken, ist die Ausdünnung und Vereinfachung der Originaldaten. Zusätzlich werden unstrukturierte Punktwolken oftmals gefiltert und durch diverse Interpolationsmethoden in reguläre, Raster basierende Punktwolken umgewandelt, um beispielsweise Berechnungen zu vereinfachen (VILANOVA et al. 2008). Auch die erste Version des hier vorgestellten Visualisierungssystems (HESINA et al. 2009) beinhaltete solche Optimierungen, um eine Darstellung mit den Daten in Echtzeit zu gewährleisten. Eine Vereinfachung der Originaldaten verursacht jedoch nicht nur visuelle Qualitätseinbußen, es können dadurch auch Ergebnisse von Messungen beeinflusst werden. In weiterer Folge wird durch den Genauigkeitsverlust die Aussagekraft des Modells sowie aller darauf basierenden Berechnungen gemindert (VILANOVA et al. 2008).

Die vorliegende Arbeit beschäftigt sich mit der Entwicklung eines Prototyps zur Bearbeitung und Visualisierung von hochauflösenden Geodaten. Das Hauptaugenmerk liegt hierbei auf der Darstellung von ungefilterten Punktwolken und der Möglichkeit, für den Benutzer mit den Daten auf einfache Art und Weise zu interagieren. Dazu war die Entwicklung einer robusten *Out-of-Core* fähigen Datenstruktur notwendig. Diese Datenstruktur muss einerseits die Visualisierung in unterschiedlichen Detailgraden erlauben, um hohe Rendergeschwindigkeiten zu gewährleisten (*multi-resolution*), und andererseits muss der Detailgrad zur Laufzeit anpassbar sein (*view-dependent*). Darüber hinaus soll das System in der Lage sein, Daten unterschiedlicher Quellen und Formate mit der Punktwolke zu fusionieren. Ein wichtiger Aspekt dabei ist die Möglichkeit, jegliche Daten nicht ausschließlich vom lokalen Dateisystem, sondern zusätzlich über das Netzwerk von einem zentralen Server laden zu können.

Im folgenden Abschnitt präsentieren wir einen Überblick über die Architektur des Prototyps. Abschnitt 3 beschäftigt sich mit dem Aufbau der LIDAR-Datenverwaltungsschicht. In Abschnitt 4 werden die Herausforderungen sowie Lösungsvorschläge für die Verarbeitung großer Mengen von Geodaten präsentiert. Abschnitt 5 zeigt einige Methoden zur Visualisierung von Punktwolken und Vektordaten und Abschnitt 6 geht abschließend auf einige Probleme und mögliche Verbesserungen ein.

2 Systemüberblick

Der Prototyp wurde in C#/.NET 4.0 entwickelt und basiert auf dem Szenegraph-orientierten 3D-Rendering-Framework *Aardvark*. Dieses Rendering-Framework ist eine Eigenentwicklung der VRVis GmbH und bildet die Basis für Projekte mit dem Ziel, große Datenmengen interaktiv darzustellen.

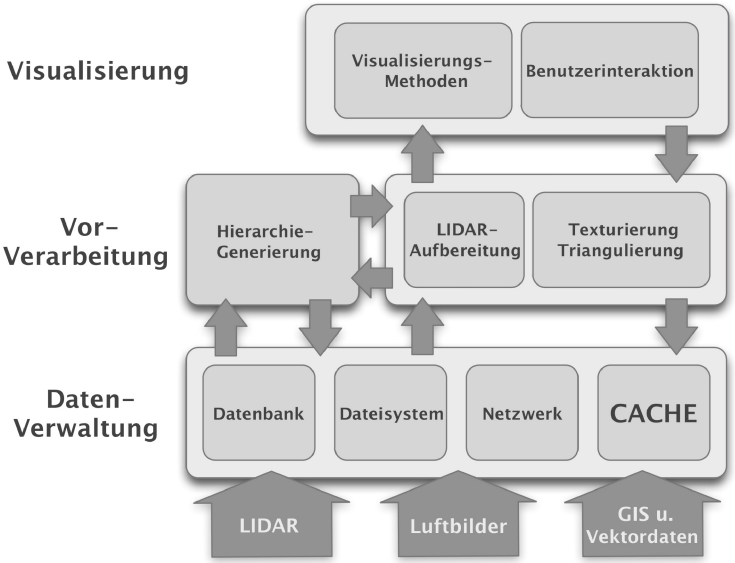

Abb. 1: 3-Schicht-Architektur der Visualisierungspipeline

Die Visualisierungspipeline ist an das Design einer klassischen 3-Schichten-Architektur angelehnt und lässt sich in die Ebenen *Datenverwaltung*, *Vorverarbeitung* sowie *Visualisierung* gliedern (siehe Abb. 1). Die Datenakquise ist genaugenommen ebenfalls ein Teil der Visualisierungspipeline. Diese Arbeit beschäftigt sich jedoch mit der Aufbereitung und Visualisierung von LIDAR-Daten. Aus diesem Grunde wird der Datenerfassungsprozess an dieser Stelle nicht behandelt.

Ein Datenaustausch zwischen direkt aneinandergrenzenden Schichten findet ausschließlich über fest definierte Schnittstellen und Dienste statt. Eine Sonderstellung in der Pipeline wird der Punkthierarchiegenerierung beigemessen. Obwohl dieser Teil ebenfalls zur Vorverarbeitungsschicht zählt, wurde die Funktionalität vom Rest der Schicht abgekapselt. Dadurch ist es möglich, direkt im Backend der Applikation auf die Logik der Hierarchiegenerierung zuzugreifen, um die zur Visualisierung notwendigen Datenstrukturen zu generieren. Bei großen Datensätzen, deren Bearbeitung mehrere Stunden dauert, ergibt sich für den Benutzer der Vorteil, diesen Vorgang im Offlinemodus oder gesteuert durch Skripte erledigen zu können.

3 Datenverwaltung

Die Datenverwaltungsschicht bildet eine Abstraktion unterschiedlicher Datenquellen. Aktuell werden drei Optionen zum Import von LIDAR Daten, Luftbildern und Vektordaten unterstützt: *Lokales Dateisystem*, *Netzwerk-Ressource*, *Oracle Geodatenbank.* Eine weitere Aufgabe dieser Schicht ist das Speichern und die Verwaltung des Applikations-Caches. Nachdem die Eingabedaten die Vorverarbeitungsschicht durchlaufen haben, wird das Resultat in den Cache geschrieben. Dadurch muss ein Datensatz die Prozedur nur einmal

durchlaufen und kann bei der nächsten Verwendung ohne viel Zeitaufwand direkt aus dem Cache geladen werden. Die Verwendung solcher zwischengespeicherten Datensätze garantiert einen äußerst schnellen Start der Visualisierung, sowie schnelles ein- und ausblenden von einzelnen Modellen.

Da hochauflösende Luftbilder sowie LIDAR-Daten tendenziell einen äußerst hohen Speicherbedarf haben, halten wir es für sinnvoll, diese Daten nicht lokal auf einem PC zu speichern. Deshalb fließt derzeit noch ein großer Teil unserer Arbeit in die Verbesserung der direkten Kommunikation zwischen Vorverarbeitungsschicht und Datenbank. Ziel ist es, zukünftig auf Datenimporte vom Dateisystem verzichten zu können und die Werkzeuge zur visuellen Analyse von Punktwolken direkt auf der Datenbank operieren zu lassen.

4 Datenaufbereitung und Zusammenführung

Der erste Schritt, um letztendlich eine homogenisierte Darstellung unterschiedlicher Geodaten zu erreichen, ist die Überführung der Datensätze in ein gemeinsames Referenzsystem. Vor allem bei der Verarbeitung von GI Daten unterschiedlicher Herkunft ist dieser Schritt notwendig, da hier sehr oft unterschiedliche Koordinatensysteme verwendet werden. Zur Lösung dieses Problems mussten in der ersten Version des Protoyps noch externe Werkzeuge verwendet werden. Mittlerweile lässt sich dieser Vorgang aufgrund einer parametrisierbaren Vorverarbeitung vollständig in der Visualisierungsapplikation durchführen. Jeder Datensatz kann entweder vor dem eigentlichen Datenimport oder zur Laufzeit anhand von Verschiebevektoren und Rotationsmatrizen transformiert werden. Zusätzlich werden die gewonnenen LIDAR-Daten automatisch in den Koordinatenursprung des Visualisierungssystems verschoben, da aufgrund der Ausdehnung der vorhandenen Punktwolken die zur Verfügung stehende numerische Genauigkeit oft nicht mehr ausreicht und dadurch Fehler in der Visualisierung verursacht. Der Grund dafür liegt in der Architektur moderner Grafikhardware, da diese intern lediglich mit 32-bit-Genauigkeit arbeitet. In den meisten Fällen ist dieser Bereich ausreichend, doch gerade bei der Darstellung von räumlich weit ausgedehnten Daten ist eine 32-bit-Genauigkeit häufig der Grund für Rendering-Artefakte.

Im anschließenden Teil der Visualisierungspipeline werden die Laserscandaten hierarchisch unterteilt und in eine darstellbare Datenstruktur eingeordnet. Zusätzlich kann hier aus Teilen der Punktwolke automatisch ein texturiertes Triangle-Mesh erzeugt werden, wenn der Benutzer diese Darstellungsform ausgewählt hat. In den folgenden 2 Abschnitten werden diese Vorgänge näher erläutert.

4.1 Hierarchiegenerierung/Out-of-Core Datenstruktur

Bei der Entwicklung der Hierarchiegenerierung gab es einige Aspekte, die für uns besonders wichtig waren: Zum einen muss die Datenstruktur multi-resolution fähig sein, um hohe Frameraten zu erreichen. Zusätzlich muss es möglich sein die Hierarchie aufzubauen, ohne Beschränkungen seitens des Arbeitsspeichers und der Punktmenge. Ein weiterer wichtiger Punkt für uns war die Unabhängigkeit vom Format der LIDAR-Daten.

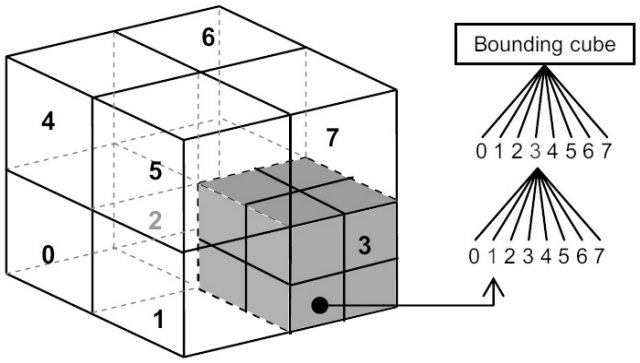

Abb. 2: Prinzip der Octree-Generierung (GIRARDEAU-MONTAUT et al. 2005)

Aus diesem Grunde verwenden wir ein eigenes, binäres Punktformat mit dem es möglich ist, beliebige Attribute (zum Beispiel Punktfarbe, Intensität, etc.) zu jeder 3D-Position zu speichern. Um dies zu erreichen, muss nun lediglich eine Beschreibung des Punktformats angegeben werden und anhand dieser Spezifikation kann ein 3D-Punkt inklusive Attribute serialisiert werden. Abhängig von der Beschaffenheit der Punktwolke ist es für den Benutzer möglich, einen Unterteilungsalgorithmus zu wählen. Derzeit werden zur räumlichen Unterteilung zwei Formate unterstützt: *Quadtree* und *Octree*.

Beide Datenstrukturen ähneln sich in ihrem Unterteilungsschema. Der Raum wird, ausgehend von der minimalen *Bounding-Box* der Punktwolke, rekursiv entlang der Hauptachsen des Koordinatensystems in vier bzw. acht gleich große Würfel unterteilt, bis eine bestimmte Abbruchbedingung erfüllt ist. Da die Unterteilung beim Quadtree nur auf zwei Achsen durchgeführt wird, eignet sich dieses Schema eher für Punkwolken, die vermehrt in der Ebene ausgedehnt sind. Je höher die Dichte und Ausdehnung der Punktwolke in *X*, *Y* sowie *Z* Richtung, desto eher wird die Wahl auf eine Octree Datenstruktur fallen. Jeder Rekursionsschritt entspricht einer Ebene in der Hierarchie mit einer Teilmenge der Punkte des Elternknotens. Diese reduzierte Teilmenge wird anhand eines Ausdünnungsfaktors über die Punktmenge einer Ebene berechnet. Dadurch ist es später möglich, den Detailgrad zur Laufzeit sehr einfach anzupassen. Nähere Details zu diesem Thema werden in Abschnitt 5 präsentiert. Die Abbruchbedingung der Rekursion ist erfüllt, sobald entweder alle Punkte in den Baum eingeordnet wurden, oder in einem Rekursionsschritt die Mindestanzahl von Punkten pro Baumknoten unterschritten wurde. Wie der Name der Datenstruktur bereits deutlich hervorhebt, entsteht durch die rekursive Unterteilung des Raums eine baumartige Struktur. Oft werden beim Aufbau einer solchen Datenstruktur Daten ausschließlich in den Blattknoten gespeichert. Das bedeutet allerdings, dass manche Objekte in mehreren Blattknoten abgelegt werden. Um dieses Verhalten zu umgehen, speichern wir Punktdaten nicht nur in Blattknoten, sondern auch in den inneren Knoten.

Um überhaupt in der Lage zu sein, beliebig große Punktwolken verarbeiten zu können, nutzen wir eine weitere Funktionalität der Datenverwaltungsschicht, die *PointSource*. Eine PointSource ist eine weitere Abstraktion einer Datenquelle. Diesmal mit der Aufgabe, einen Strom von Punkten darzustellen, unabhängig von Punktmenge und Anzahl einzelner Quelldateien. Die PointSource gruppiert Punkdaten in Blöcke im Speicher, die nacheinander abgearbeitet werden. Sobald ein Block in die Hierarchie einsortiert wurde, wird der Spei-

cher freigegeben und der nächste Block kommt an die Reihe. Selbiges Konzept wird auch zur Generierung des Hierarchie-Caches angewandt. Jeder Knoten einer Punkthierarchie besitzt eine sogenannte *PointSink*. Eine PointSink verwendet einen Zwischenspeicher, dessen Größe vom Benutzer anpassbar ist um Punkte zu speichern die dem Knoten zugeordnet wurden. Ist dieser Zwischenspeicher voll, wird dessen Inhalt in eine temporäre Datei auf die Festplatte geschrieben und der Speicher freigegeben. Sollte für den entsprechenden Knoten bereits eine Datei existieren, werden die neuen Punkte am Ende hinzugefügt. Ist die Generierung abgeschlossen, werden alle temporären Dateien eingesammelt, in den Hierarchie-Cache geschrieben und die Dateien entfernt. Auf diese Weise ist es möglich Hierarchien zu generieren, deren Größe die des Arbeitsspeichers deutlich übersteigt.

4.2 Dynamische Triangulierung und Texturierung

In der einfachsten Form enthalten Laserscandaten lediglich die georeferenzierte 3D-Position eines Messpunktes. Allerdings werden während eines Scanvorgangs zusätzliche Daten gesammelt, die in der Nachbearbeitung zum Beispiel zur Klassifizierung des 3D-Punktes verwendet werden können. Wie schon im letzten Abschnitt erwähnt wurde, ist die Verwertung solcher Zusatzattribute ein Teil unserer Visualisierungspipeline. Die Nachbearbeitung von LIDAR-Daten, bzw. das Klassifizieren von Punktwolken, ist nicht Thema dieser Arbeit. Wir wollen jedoch eine Bearbeitungsmöglichkeit unseres Prototyps vorstellen, die sich durch eine klassifizierte Punktwolke ergibt.

Die uns zur Verfügung stehenden Testdaten sind anhand verschiedener Objekttypen wie etwa Grundfläche, Gebäude oder Vegetation klassifiziert. Dadurch ist es zur Laufzeit möglich, gezielt Operationen auf Punkte anzuwenden, die einer bestimmten Kategorie angehören. So können etwa Punkte gefiltert oder auch in ein eine neue Abfolge von Verarbeitungsschritten umgeleitet werden. Ein konkretes Beispiel hierfür ist die automatische Generierung eines Drahtgittermodells aus Laserscanpunkten, die als „Grundfläche“ klassifiziert wurden. Aufgrund der hohen Dichte der Grundfläche bietet es sich an, diesen Teil des Datensatzes abweichend stark auszudünnen, um mittels 2D-Delaunay-Triangulation mehrere Modelle mit unterschiedlichem Detailgrad zu erstellen. Da die lokalen Höhenunterschiede zwischen den Grundflächen-Punkten nicht besonders groß sind, garantiert der verwendete Triangulationsalgorithmus eine vernachlässigbare Abweichung zur gescannten Repräsentation.

Um dem Drahtgittermodell Farbinformationen zuzuweisen, wird das vorhandene Luftbildmaterial verwendet. Meistens enthalten orthofotografisch gewonnene Luftbilder neben Farbinformationen zusätzliche Erfassungsattribute, wie zum Beispiel geografisch zusammenhängende Bildkoordinaten, tatsächliche Pixelgröße, Verzerrung und Rotation. Ein häufiges Problem beim Verwenden von Luftbildern in der Visualisierung sind inkonsistente Pixeldimensionen und natürlich der Speicher. Hochauflösende Bilder bestehen in der Regel aus Hunderten von Megabytes. Dadurch wird es schwierig, diese in Echtzeit Visualisierungen einzusetzen. Um dieses Problem zu lösen, werden die angesprochenen Metadaten aus den Bildern genutzt. Es wurde eine Datenstruktur entwickelt, die eine Menge von Bildern virtuell, unter Berücksichtigung möglicher Drehungen und Verzerrungen, auf eine zusammenhängende, große Textur abbildet. Diese Datenstruktur dient als Fassade zwischen dem Visualisierungssystem und einer beliebigen Menge von Bildern – unabhängig Ihrer Auflösung und Größe. Zusätzlich bietet die Datenstruktur praktische Werkzeuge, um etwa beliebige Bereiche in einer gewünschten Auflösung aus der Textur zu extrahieren, was bedeutet,

dass auf diese Weise sehr einfach eine Reihe homogener Kacheln aus den Luftbildern erstellt werden können. In weiterer Folge werden diese Kacheln zur Texturierung des Drahtgittermodells verwendet. Zusätzlich werden die eigentlichen Bilddaten bei Bedarf dynamisch von der Festplatte geladen, was den Speicherverbrauch drastisch reduziert.

5 Visualisierung

Um die Performanz während der Darstellung so hoch als möglich zu halten, wurde für die Punktvisualisierung ein *Level of Detail* (LOD)-System entwickelt, das den Detailgrad der dargestellten Punkte – abhängig von der Distanz zum Betrachter – dynamisch anpasst. Grundlage dafür war die hierarchische Datenstruktur, die in Abschnitt 4.1 vorgestellt wurde. Um einen Datensatz von vielen Mio. Punkten darzustellen, muss dieser zunächst in den Hauptspeicher und von dort in den Grafikspeicher geladen werden. Normalerweise ist es jedoch nicht notwendig, die gesamte Menge an Punkten auf einmal darzustellen, da der Betrachter ab einer gewissen Distanz die Menge an Punkten nicht mehr erkennt. Mithilfe der Datenstruktur, welche die Szene räumlich unterteilt, ist es möglich durch Verschneidung der *Bounding-Boxen* der Baumknoten mit dem *Kamera-View-Frustum* jene Teilmenge der Punktwolke zu bestimmen, die vom Betrachter potenziell gesehen werden kann.

Diese Menge an Punkten wird dynamisch in den Speicher geladen, und abhängig von der Distanz zum Betrachter die Dichte der Punktwolke angepasst. Das LOD-System wird zur Laufzeit von einem Speichermanager überwacht. Sobald ein bestimmter Detailgrad in den Speicher geladen wurde, da die zugehörigen Baumknoten in das Sichtfeld des Benutzers gelangt sind, markiert der Speichermanager diese Knoten in der in der LOD-Datenstruktur. Knoten, die seit Längerem nicht mehr in das Sichtfeld des Benutzers gelangt sind, können nun ganz einfach aus dem Speicher entfernt und somit der Speicherverbrauch gesteuert werden. Je näher der Betrachter einem Objekt in der Szene kommt, desto detaillierter wird dieses dargestellt; an anderer Stelle nimmt der Detailgrad gleichermaßen ab. Dadurch ist es möglich, eine äußerst hohe Qualität in der Darstellung zu erreichen und Segmente der Punktwolke in Originalauflösung zu analysieren.

5.1 Visualisierungsmethoden

Die entwickelten Datenstrukturen erlauben es, in Echtzeit mit dem 3D-Modell zu interagieren. Beispielsweise können alle Laserscanpunkte, die einer bestimmten Kategorie angehören, per Mausklick mit einer beliebigen Farbe eingefärbt (siehe Abbildung 4) werden (um etwa risikoreiche Gegebenheiten in der Szene hervorzuheben oder um dadurch die Sicht auf einen anderen Teil der Szene zu verbessern). Auch das komplette Ausblenden von Teilbereichen der Punktwolke ist möglich.

Eine weitere Möglichkeit, die Übersicht in der Szene zu verbessern, ergibt sich mit dem Höhenfilter. Hier hat der Benutzer die Möglichkeit, interaktiv Schnittebenen durch die Punktwolke entlang der Höhenachse zu legen. Ein Effekt, der bei der Darstellung von Punktwolken deutlich zum Vorschein kommt, ist der Eindruck von „Löchern“ oder stark ausgedünnten Bereichen, wenn diese aus kurzer Distanz betrachtet werden. Die Ursache dafür ist die Größe, in der ein Punkt dargestellt wird. Diese ist zunächst für alle Punkte eines Datensatzes gleich. Wenn der Laserscan aus einer großen Entfernung betrachtet wird,

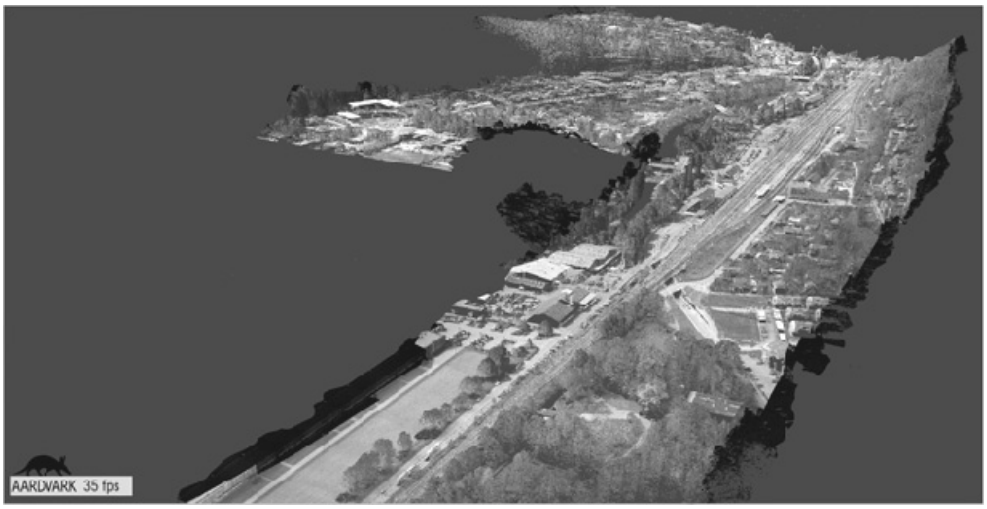

Abb. 3: Punktwolke mit rund 50 Mio. Laserscanpunkten

Abb. 4: In dieser Ansicht wurden die Punktdaten anhand Ihrer Kategorie eingefärbt

überlagern sich Punkte und es entsteht der Eindruck, dass diese Punkte eine Einheit bilden. Verringert man die Distanz dann schwindet dieser Eindruck und die Punktwolke wirkt „dünn“. Um dieses Problem zu lösen, wird in unserer Visualisierung der Durchmesser eines Laserscanpunktes dynamisch an die Distanz des Betrachters angepasst. So entsteht ein ausgewogenes, glaubwürdigeres Bild der Umgebung. Verfügbare Daten aus GI Systemen wie Straßen, Bahnlinien oder Grundstücksgrenzen werden als Polylines direkt auf die Punktwolke oder das Drahtgittermodell gerendert. Polylines sind triangulierte Linienzüge, die automatisch aus den Vektordaten errechnet werden. Diese Linienzüge werden ähnlich wie der Durchmesser eines Laserpunktes abhängig zur Distanz des Betrachters skaliert.

Hier jedoch mit dem Unterschied, dass die Linienzüge durch die Skalierung stets denselben Durchmesser beibehalten, unabhängig von der Entfernung zum Betrachter. Alle Visualisierungsmethoden wurden als *HLSL Shader* Programme in die Visualisierungspipeline integriert und arbeiten somit in Echtzeit direkt auf der Grafikhardware.

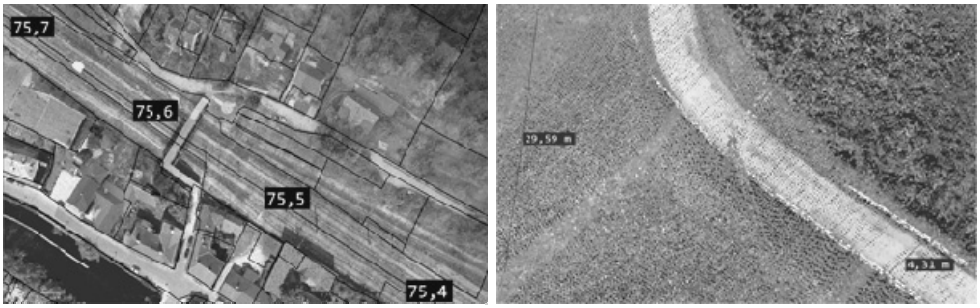

Abb. 5, 6: Visualisierung von GIS- und Bahndatendaten sowie Distanzmessungen

5.2 Benutzerinteraktion

Benutzern des Systems bieten sich momentan zwei Optionen, die Visualisierung zu betrachten. Die erste und vermutlich intuitivste Variante erlaubt eine Maus und Tastatur gesteuerte Bewegung in der Ego-Perspektive durch den gesamten Datensatz. Bei Verwendung der zweiten Variante kann zunächst eine Position in der Szene als Zentrum definiert werden. Danach kann die gesamte Szene durch Maussteuerung um den gewählten Mittelpunkt rotiert werden. Diese Art der Navigation eignet sich ausgezeichnet für kleinere Datensätze, die der Benutzer rasch von verschiedenen Seiten betrachtet möchte.

Zusätzlich kann der Benutzer in der höchsten Darstellungsform der Punktwolke direkt Punkte für eine Distanzmessung selektieren. Aufgrund dieser Selektion im 2D-*Screenspace* wird ein Strahl im 3D-Raum durch die Wolke berechnet. Danach wird anhand dieser Strahlen eine Schnittberechnung mit den einzelnen Knoten der Hierarchie durchgeführt. So ist es möglich, die Menge der Punkte, die für die Distanzberechnung infrage kommen, in sehr kurzer Zeit von mehreren Millionen auf einige Zehntausend zu reduzieren.. Aus diesen Teilbereichen der Punktwolke werden in weiterer Folge jene Positionen ermittelt, die vom Benutzer zur Berechnung selektiert wurden.

6 Fazit/Ausblick

In der aktuellen Version des Prototyps ist es möglich, umfangreiche Punktwolken mit unter-schiedlichen Attributen in Echtzeit darzustellen. Die Hierarchiegenerierung, Datenzusammenführung sowie die Erstellung eines Drahtgittermodells, basierend auf Punktklassifizierungen, ist ein Prozess, den der Benutzer durch Parameter steuern kann. Das heißt, dass dieser Teil der Vorverarbeitung noch nicht vollkommen automatisiert ist. Zukünftig soll in der Vorverarbeitungsschicht die optimale Baumtiefe und Punktmenge für jeden Knoten der Hierarchie berechnet werden, um die Ausbalancierung des Baumes zu verbessern. Eine

weitere Verbesserung wäre, die Skalierung der Laserscanpunkte auch von der lokalen Punktdichte abhängig zu machen. Somit könnten Bereiche der Punktwolke mit variierender Scandichte durch eine Anpassung der Punktgröße deutlich glaubwürdiger dargestellt werden. Auch die Integration zusätzlicher Analyse- und Visualisierungsmethoden ist ein Anliegen, um dem Benutzer neue Möglichkeiten der Interpretation von LIDAR-Daten zu ermöglichen.

7 Danksagung

Diese Arbeit wurde in Zusammenarbeit mit der Firma ÖBB Infrastruktur Bau im Zuge des Projekts GECOV erstellt. Unterstützt wurden wir dabei von der österreichischen Forschungsförderungsgesellschaft (FFG) im Rahmen des Förderungsprogrammes COMET.

Literatur

GIRARDEAU-MONTAUT, D., ROUX, M., MARC, R. & THIBAULT, G. (2005): Change Detection on Points Cloud Data acquired with a Ground Laser Scanner. International Achrives of Photogrammetry

HESINA, G., LEITNER, B., MANTLER, S. & BRIMMER, F. (2009): Infrastructure Acquisition and 3D Virtual Integration. Real Corp Proceedings. Barcelona, Remote Sensing, 36 (3/W19), S. 30-25.

MCGAUGHEY, R. K. & CARSON, W. W. (2003): Fusing LIDAR Data, Photographs, and other Data Using 2d and 3d Visualization Techniques. Proceedings of Terrain Data: Applications and Visualization – Making the Connection, S. 16-24. Charleston, SC.

SCHENK, T. & CSATHÓ, B. (2007): Fusing Imagery and 3D Point Clouds for Reconstructing Visible Surfaces of Urban Scenes. IEEE GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas.

VILANOVA, A., TELEA, G., SCHEUERMANN, G. & MÖLLER, T. (2008): Visual Analysis and Semantic Exploration of Error Aware Urban Change Detection. Eurographics/IEEE-VGTC Symposium on Visualization.