PFG 2011 / 3, 145 – 154
Stuttgart, June 2011

Article

Integrierte Echtzeit-Visualisierung von massiven 3D-Punktwolken und georeferenzierten Texturdaten

Rico Richter & Jürgen Döllner, Potsdam

Keywords: 3D Point Clouds, LiDAR, Point-based Rendering, Out-of-Core Visualization

Summary: *Integrated real-time visualization of massive 3D point clouds and geo-referenced texture data*. We present a technique that enables the integrated real-time visualization of massive 3D point cloud data in combination with geo-referenced texture data. Both represent essential geo data categories that become more and more applied due to increasing availability and decreasing costs generated by airborne and mobile scanning systems. The massivity of the generated data poses challenges both for algorithmic issues and hardware resources. The efficient processing and real-time visualization require out-of-core algorithms and LOD structures. The proposed out-of-core and LOD concept for unlimited large 3D point cloud data enables the real-time visualization of that data combined with geo-referenced texture data such as digital aerial images. This approach allows us to attribute arbitrary geo-referenced thematic information to the 3D point cloud data. This way, objects and regions cannot only be geometrically, but also thematically be presented, explored, and analyzed. The data is prepared in a preprocessing step and transferred into a spatial LOD structure, which is used by the real-time 3D point-based rendering technique. The technique can be used to develop interactive 3D analysis tools and simulation techniques that can directly and efficiently access raw 3D point clouds data.

Zusammenfassung: Dieser Beitrag stellt ein Verfahren vor, das die integrierte Echtzeit-Visualisierung von massiven 3D-Punktwolkendaten zusammen mit georeferenzierten Texturdaten ermöglicht. Beides sind wesentliche Kategorien von Geodaten, die insbesondere aufgrund ihrer wachsenden Verfügbarkeit und sinkender Kosten, generiert mit flugzeuggestützten oder mobilen Scan-Systemen, zunehmend Einsatz finden. Die Massivität der entstehenden Daten stellt eine Herausforderung sowohl für die Algorithmik wie auch für die Hardware-Ressourcen dar. Um die Daten effizient verarbeiten und in Echtzeit visualisieren zu können, werden sog. Out-of-Core-Algorithmen und Level-of-Detail-Strukturen benötigt. Das vorgestellte Out-of-Core- und LOD-Konzept für unbeschränkt große 3D-Punktwolkendaten ermöglicht deren Echtzeit-Visualisierung zusammen mit georeferenzierten Texturdaten, z. B. digitalen Luftbildern. Dieser Ansatz erlaubt es, 3D-Punktwolkendaten in der Visualisierung mit beliebigen georeferenzierten thematischen Information zu attributieren, sodass z. B. Objekte und Bereiche nicht nur geometrisch, sondern auch thematisch präsentiert, exploriert und analysiert werden können. In einem Vorverarbeitungsschritt werden dazu die Daten aufbereitet und in eine räumliche LOD-Struktur überführt, auf der die echtzeitfähige, punktbasierte 3D-Rendering-Technik operiert. Mit dem Ansatz wird es möglich, interaktive 3D-Analyse-Werkzeuge und Simulationsverfahren zu realisieren, die direkt und effizient auf 3D-Punktwolkenrohdaten zugreifen.

1 Einleitung

3D-Punktwolkendaten finden eine wachsende Verwendung in nahezu allen Bereichen, die zu einem bestimmten Zeitpunkt eine exakte Erfassung der räumlichen Strukturen, insbesondere der Oberflächenstrukturen, benötigen. Dabei kommen neben flugzeug- und satellitengestützten Erfassungssystemen (z. B. LiDAR) zunehmend auch stationäre Scan-Systeme zum Einsatz. Nicht nur durch die Größe des Erfassungsgebietes, sondern auch durch

www.schweizerbart.de
DOI: 10.1127/1432-8364/2011/0079
1432-8364/11/0079 $ 2.50

steigende Auflösung, Präzision und Geschwindigkeit bei der Erfassung entstehen so massive Datenmengen.

3D-Punktwolkendaten repräsentieren insbesondere in den Bereichen Planung, Konstruktion und Monitoring eine grundlegende Geodatenquelle. Sie haben vielfältige Anwendungen, z. B. zur Ableitung von 3D-Bauwerksmodellen, 3D-Geländemodellen (DGM), 3D-Oberflächenmodellen (DOM), 3D-Umgebungsmodellen oder 3D-Infrastrukturelementen. Deshalb werden 3D-Punktwolkendaten häufig als Ausgangsbasis für die Erstellung, Aufbereitung und Fortführung von virtuellen 3D-Stadtmodellen genutzt (Brenner 2005, Zhou et al. 2008, Lafarge et al. 2010). Ergänzt werden virtuelle 3D-Stadtmodelle bzw. allgemeine 3D-Raummodelle im Allgemeinen mit digitalen, georeferenzierten Fotodaten, z. B. in Form von Luftbildern oder terrestrisch bzw. aus Schrägluftbildern abgeleitete Fassadenaufnahmen (Früh et al. 2004). In geovirtuellen 3D-Umgebungen (Geovirtual 3D Environments, GeoVEs) repräsentieren digitale, georeferenzierte Fotodaten eine grundlegende Datenkategorie für die photorealistische bzw. wirklichkeitsnahe Ausgestaltung (Tanner et al. 1998, MacEachren et al. 1999, Döllner et al. 2000).

Die Echtzeit-Visualisierung von 3D-Punktwolkendaten ist eine grundlegende Anforderung, um diese Daten zu verwalten und zu nutzen. So ist z. B. eine Qualitätskontrolle des Datenbestandes erforderlich, indem alle erfassten Daten schnell und vollständig visualisiert und somit durch den Nutzer exploriert werden können. Insbesondere müssen hierbei die 3D-Punktwolkendaten auf Vollständigkeit, Genauigkeit und Deckungsgenauigkeit der einzelnen Flugstreifen überprüft werden, bevor eine aufwändige Weiterverarbeitung durchgeführt wird. Für die Visualisierung ist es darüber hinaus erforderlich, dass weitere georeferenzierte Daten, wie z. B. digital Luftbilder, topographische Karten oder thematische Karten, nahtlos in die Visualisierung einbezogen werden können, z. B. indem sie als grafische Attribute den 3D-Punktwolken zugeordnet werden.

In diesem Beitrag stellen wir ein Verfahren vor, das massive, in der Größe konzeptionell unbeschränkte 3D-Punktwolkendaten zusammen mit georeferenzierten Texturdaten in Echtzeit visualisiert. Die Größe der Daten ist in der Praxis lediglich durch die Kapazität der verfügbaren Speichermedien limitiert. Das Verfahren baut dazu in einem Vorverarbeitungsschritt eine LOD-Datenstruktur auf, die einen effizienten Zugriff auf die 3D-Punktwolke in unterschiedlichen Auflösungen ermöglicht. Eine darauf abgestimmte punktbasierte 3D-Renderingtechnik operiert auf dieser LOD-Datenstruktur und wählt dabei sicht- und qualitätsabhängig für jeden Raumbereich entsprechend aufgelöste 3D-Teilpunktwolken aus. Die Aktualisierung eines zugeordneten Cache übernimmt ein separater Thread, sodass die Cache-Aktualisierung bzw. das Nachladen der LOD-Daten die Interaktivität des Verfahrens nicht beeinflusst. Die vollständige, echtzeitfähige und von Hardware-Ressourcen unabhängige Visualisierung ermöglicht es, dass sich Nutzer interaktiv in massiven 3D-Punktwolken unabhängig von deren Datenmenge bewegen können; die Integration von georeferenzierten Texturdaten erlaubt es zusätzlich die 3D-Punktwolke grafisch zu attributieren.

Das vorgestellte Verfahren ermöglicht allgemein die Realisierung interaktiver Analysefunktionen, Simulationstechniken und Explorationswerkzeuge, die direkt und effizient auf massive 3D-Punktwolkenrohdaten Zugriff erhalten. So können z. B. 3D-Punktwolkendaten vor weiterer Verarbeitung durch 3D-Stadtmodellwerkzeuge visuell evaluiert werden und es lassen sich zielgerichtet 3D-Datenbestände aktualisieren (Gröger & Plümer 2009), indem bestehende Geodaten zusammen mit den neu erfassten 3D-Punktwolkendaten visuell untersucht und verglichen werden (Richter & Döllner 2010b).

Im Folgenden werden verschiedene Rendering-Techniken für 3D-Punktwolken vorgestellt (Abschnitt 2). In Abschnitt 3 wird die Aufbereitung der Daten und die verwendete räumliche Datenstruktur präsentiert. Abschnitt 4 erläutert die Rendering-Technik um die Daten zu visualisieren. Abschließend werden die Ergebnisse (Abschnitt 5) sowie Diskussion und Ausblick (Abschnitt 6) präsentiert.

2 Rendering-Techniken für 3D-Punktwolken

Die echtzeitfähige Visualisierung massiver 3D-Punktwolken beruht technisch auf einer *punktbasierten 3D-Rendering-Technik*, bei der im Gegensatz zum klassischen 3D-Rendering Punkte anstelle von Dreiecken als computergrafische Primitive verwendet werden (Kobbelt & Botsch 2004). Die ersten Verfahren für punktbasiertes Rendering wurden in Surfels (Pfister et al. 2000) und QSplat (Levoy & Rusinkiewicz 2000) vorgestellt. Basierend auf einer räumlichen Datenstruktur wird eine Hierarchie aufgebaut, die verschiedene Abstraktionsstufen für das Rendering bereitstellt. Sainz et al. (2004) geben einen Überblick über verschiedene punktbasierte Rendering-Techniken.

Scheiblauer et al. (2009) stellen ein interaktives Werkzeug vor, um das Bearbeiten von massiven 3D-Punktwolken zu ermöglichen. Das Werkzeug wird unter Anderem für die Aufbereitung, Dokumentation und Analyse von archäologischen Stätten verwendet. Die bei der Erfassung entstehenden 3D-Punktwolken sind im Allgemeinen zu groß, um eine traditionelle Verarbeitung und Überführung in dreiecksbasierte 3D-Modelle durchführen zu können. In einer räumlichen Datenstruktur werden nur die Originalpunkte der 3D-Punktwolke gespeichert, sodass eine Manipulation der Daten schnell umgesetzt werden kann (Scheiblauer et al. 2009). In unserem Verfahren werden die Daten ausschließlich visualisiert, d.h. das Editieren und Entfernen von Punkten aus der Datenstruktur sowie das Erkennen von Strukturen und Objekten innerhalb der 3D-Punktwolke (Kreylos et al. 2008) ist nicht erforderlich. Somit kann eine statische LOD-Datenstruktur verwendet werden, die die Performance des Renderingsystems verbessert. Da keine Änderungen der Daten vorgenommen werden, kann die gesamte LOD-Struktur in einem Vorverarbeitungsschritt erzeugt werden.

Aufgrund der massiven Datenmenge ist die Realisierung von Out-of-Core-Strategien ein Schwerpunkt des von uns umgesetzten Verfahrens (Kreylos et al. 2008). Das Speichermanagement muss den limitierten Haupt- und Grafikspeicher optimal ausnutzen und sicherstellen, dass eine interaktive Exploration der Daten durchgeführt werden kann (Richter & Döllner 2010a).

Weiter macht unser Verfahren keine Annahmen über geschlossene Oberflächen innerhalb der 3D-Punktwolke oder deren räumliche Verteilung. Punktbasierte Rendering-Techniken für abgetastete Objekte mit einer konstanten Verteilung der Oberflächenpunkte können die Bildqualität beim Rendering beispielsweise durch die Verwendung von orientierten Splats erhöhen (Dachsbacher et al. 2003, Botsch et al. 2005). Um die Datenstruktur kompakt zu halten, werden in unserem Verfahren keine Annahmen und Berechnungen bezüglich der Abtastdichte von Punkten oder zur Existenz von Oberflächennormalen für jeden Punkt gemacht.

Wimmer & Scheiblauer (2006) stellen ein Renderingsystem vor, das für die Visualisierung von 3D-Punktwolken ohne zeitaufwändige Vorberechnung entwickelt wurde. Die schnelle Verarbeitung und Visualisierung ist ein Schwerpunkt, sodass die Bildqualität beim Rendering nicht durch zusätzliche Informationen pro Punkt verbessert werden kann. Die Punkte werden ausschließlich mit Farbinformationen dargestellt, vergleichbar mit unserem Ansatz, bei dem die Farbinformationen aus zugeordneten georeferenzierten Texturen in einem Vorverarbeitungsschritt ermittelt werden. Die Berechnung von Oberflächennormalen wäre aufgrund der unregelmäßigen Punktdichte aufwändig und würde die Vorverarbeitungszeit sowie die Speicheranforderung der Daten für das Rendering erhöhen (Kobbelt & Botsch 2004).

Goswami et al. (2010) verwenden in ihrem Renderingsystem eine räumliche Datenstruktur, die für den Datentransfer zwischen CPU und GPU optimiert wurde. Die verwendete Baumstruktur ist immer ausbalanciert, was insbesondere für die Verarbeitung von 3D-Punktwolken von Einzelobjekten von Vorteil ist, da die räumliche Verteilung der Punkte im Allgemeinen stark variiert. 3D-Punktwolken aus Befliegungen sind in der Regel gleichmäßig über das Erfassungsgebiet verteilt, sodass eine räumliche Datenstruktur mit einer gleichmäßigen räumlichen Unterteilung verwendet werden kann (z. B. Octree), die in einer kürzeren Vorverarbeitungszeit generiert werden kann.

3 Aufbereitung von 3D-Punktwolkendaten und Texturdaten

In diesem Abschnitt wird die Aufbereitung der 3D-Punktwolkendaten und Texturdaten erläutert. Neben der Datencharakteristik wird die Umsetzung des Speichermanagements erläutert, um die effiziente Verarbeitung und Aufbereitung für die Echtzeit-Visualisierung durchzuführen.

3.1 *Charakteristik der Daten*

3D-Punktwolken bestehen im Allgemeinen aus einer Menge von ungeordneten georeferenzierten 3D-Punkten. Diese können in verschiedenen Formaten (z. B. *las*, *xyz*) vorliegen und sind in der Regel in mehrere Bereiche, basierend z. B. auf den Befliegungsstreifen oder der Kachelung des Erfassungsgebietes, unterteilt. Die Anordnung und Dichte der Punkte in einer 3D-Punktwolke kann stark variieren und hängt von der Scan-Technologie, zahlreichen Faktoren während der Erfassung sowie der Charakteristik und Beschaffenheit des Erfassungsgebietes ab.

Zum Beispiel umfasst eine 3D-Punktwolke für ein Stadtgebiet mehrere Milliarden Punkte und benötigt nicht selten mehrere hundert Gigabyte an Speicherkapazität. Texturdaten, z. B. Luftbilder für ein Stadtgebiet, sind in der Regel in Kacheln angeordnet, deren gesamte Größe in Abhängigkeit der Bildauflösung mehrere hundert Gigabyte betragen kann. Folglich ist eine direkte Verarbeitung dieser Daten im Haupt- und Grafikspeicher nicht möglich. Für die Verarbeitung und Visualisierung dieser Daten sind daher *Out-of-Core-Algorithmen* (bzw. External-Memory Algorithms) notwendig. Deren spezielle Speichermanagement-Techniken sorgen für eine optimale Ausnutzung der verfügbaren endlichen Hardware-Ressourcen und ermöglichen durch die Entkopplung der Datenaktualisierung im Speicher von dem Rendering des momentan verfügbaren Datenbestands eine echtzeitfähige Visualisierung.

3.2 *Grafische Attributierung der 3D-Punktwolke*

Im ersten Schritt der Aufbereitung wird die 3D-Punktwolke mit RGB-Farbinformationen versehen, indem für jeden Punkt die RGB-Farbe aus den zugehörigen Texturdaten ermittelt wird. Die Herausforderung besteht darin, dieses Verfahren für mehrere Milliarden Punkte umzusetzen, um beispielsweise ein ganzes Stadtgebiet aufbereiten zu können.

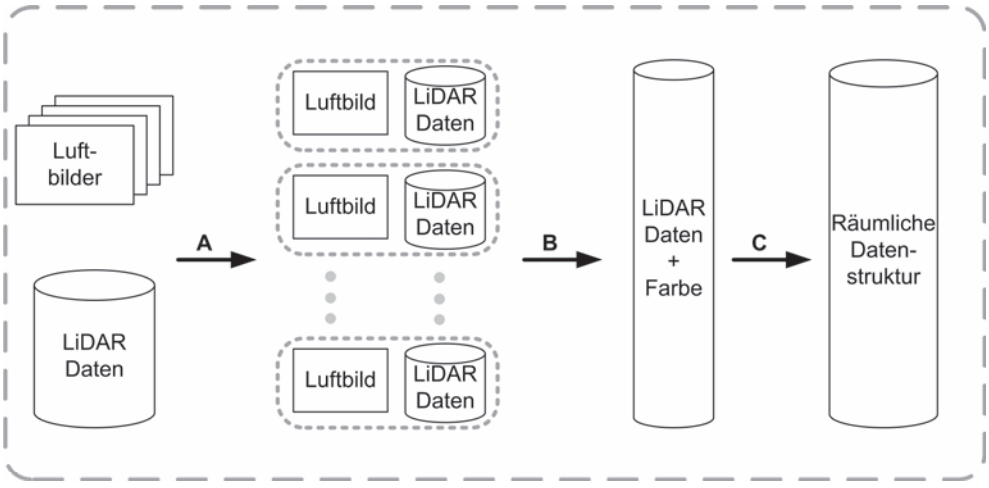

Abb. 1: Arbeitsweise am Beispiel von LiDAR-Daten und Luftbildern: Die Daten werden aufgeteilt (A), mit Farbattributen versehen (B) und in eine räumliche Datenstruktur überführt (C).

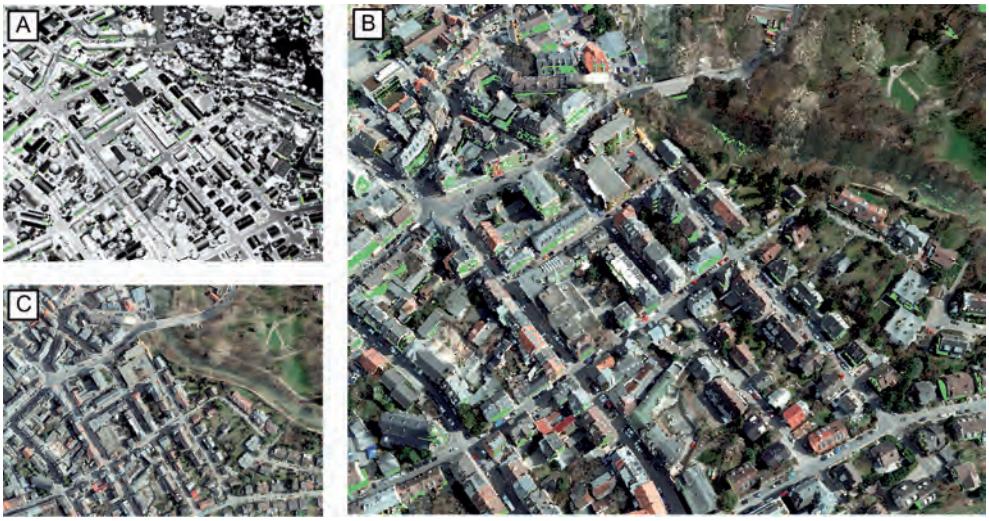

Abb. 2: 3D-Punktwolke ohne Farbinformationen (A) wird basierend auf einem Luftbild (B) mit Farbattributen angereichert (C).

Eine schnelle und effiziente Verarbeitung ist nur dann möglich, wenn der Datentransfer zwischen dem Hauptspeicher und dem sekundären Speicher (z. B. Festplatte) so gering wie möglich gehalten wird.

Die 3D-Punktwolke wird, basierend auf den Kacheln der Luftbilder, vorsortiert (Abb. 1A). Die Kacheln für die im Beispiel verwendeten LiDAR-Daten werden dabei aus der Lage und Ausdehnung der Luftbilder abgeleitet. Danach werden alle Kachelpaare, bestehend aus Luftbild und LiDAR-Daten, sequentiell im Hauptspeicher verarbeitet (Abb. 1B). Für jede Kachel werden das Luftbild und der zugehörige Teil der 3D-Punktwolke in den Hauptspeicher geladen. Jeder Punkt wird in das Luftbild hinein projiziert, um die Farbinformation zu ermitteln. Abb. 2 veranschaulicht die Ergebnisse für eine Kachel indem eine 3D-Punktwolke ohne Farbinformationen (dargestellt mit einem Farbverlauf basierend auf der Punkthöhe) und das zugehörige Luftbild sowie die resultierende farbige 3D-Punktwolke gegenübergestellt werden.

Die Attributierung der 3D-Punktwolkendaten wird in einem Vorverarbeitungsschritt durchgeführt, da bei einer Attributierung während des Renderings zusätzliche Speicher-Ressourcen für die Texturen notwendig wären. Der benötigte Speicher für die Texturinformationen pro Punkt ist generell geringer, als der benötigte Speicher für das Vorhalten von Texturen für die sichtbaren Bereiche der 3D-Punktwolkendaten während des Renderings. Darüber hinaus wäre ein zusätzliches LOD-Verfahren für Texturen erforderlich (z. B. Clipmapping).

3.3 *Erzeugen der räumlichen LOD-Datenstruktur*

Für eine interaktive Visualisierung der 3D-Punktwolkendaten wird die Auswahl von Punkten in Abhängigkeit des Sichtabstandes und Sichtbereiches getroffen. Um eine schnelle Auswahl der darzustellenden Punkte zu ermöglichen, werden im zweiten Schritt der Aufbereitung die gesamten 3D-Punktwolkedaten in eine räumliche LOD-Datenstruktur überführt (Abb. 1C). Die verwendete Baumstruktur ermöglicht den schnellen Zugriff auf benötigte Bereiche der 3D-Punktwolke und beinhaltet zusätzlich verschiedene Abstraktionen für die 3D-Punktwolke, um den Detailgrad in Abhängigkeit der zur Verfügung stehenden Hardware-Ressourcen sowie Nutzerinteraktionen anpassen zu können.

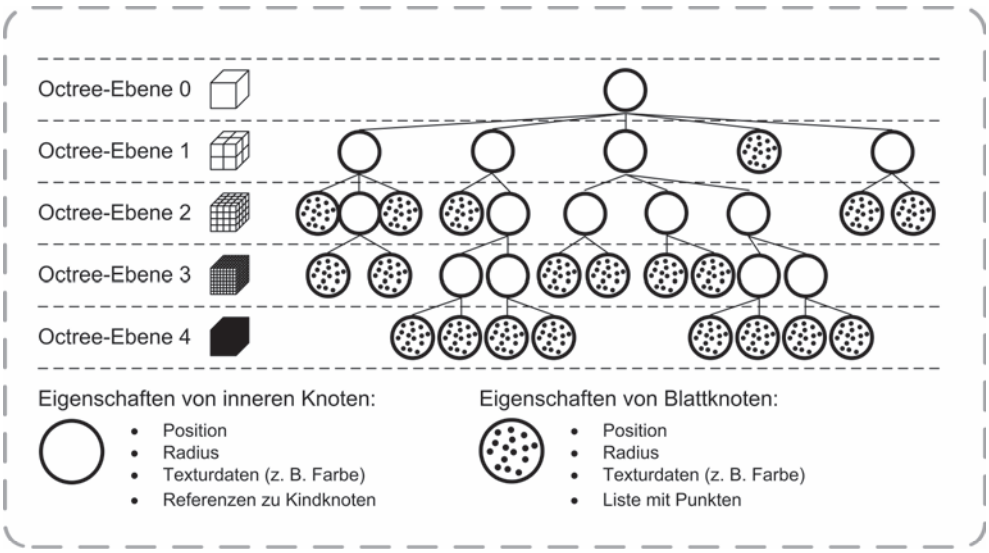

Abb. 3: Aufbau der LOD-Datenstruktur basierend auf einem Octree der Knoten mit 3D-Punktwolkendaten speichert.

Als räumliche LOD-Datenstruktur wird ein Octree verwendet, da dieser im Gegensatz zu anderen Datenstrukturen (z. B. kdTree) eine schnelle und einfache Unterteilung der unsortierten 3D-Punktwolke ermöglicht. Alle Punkte der 3D-Punktwolke werden ausschließlich in den Blattknoten gespeichert. Jeder Blattknoten speichert eine definierte Anzahl an Punkten um die Tiefe der Baumstruktur zu beschränken. Jeder innere Knoten speichert die mittlere Position aller Kindknoten sowie einen Radius, sodass eine Umkugel entsteht, die alle Punkte der Kindknoten umschließt. Abb. 3 veranschaulicht den Aufbau der LOD-Datenstruktur.

Um den Aufbau der Baumstruktur unabhängig von den Hauptspeicherressourcen durchzuführen, wird die 3D-Punktwolke mehrfach in acht Zellen unterteilt, bis die Anzahl der Punkte in einer Zelle im Hauptspeicher verarbeitet werden kann. Für jede dieser Zellen wird die Baumstruktur im Hauptspeicher erzeugt und im Sekundärspeicher abgelegt. Die separat im Hauptspeicher erzeugten Baumstrukturen jeder einzelnen Zelle werden zusammengeführt, sodass eine gesamte Baumstruktur entsteht, die alle Punkte mit Farbinformationen für eine beliebig große 3D-Punktwolke beinhaltet.

4 Echtzeit-Visualisierung von 3D-Punktwolken

In diesem Abschnitt wird das Verfahren vorgestellt, das die interaktive Visualisierung der vorprozessierten 3D-Punktwolkendaten realisiert.

Da die Datenmenge in der Regel die Kapazität des Hauptspeichers um Größenordnungen übersteigt, kann stets nur ein kleiner Ausschnitt im Hauptspeicher resident sein. Die räumliche LOD-Datenstruktur ermöglicht den Zugriff auf ein beliebiges Detaillevel der 3D-Punktwolke. Jeder Knoten im Baum repräsentiert einen bestimmten Bereich der 3D-Punktwolke, für den festgestellt werden kann, ob er im Sichtbereich des Nutzers (View-Frustum) liegt. Knoten außerhalb des Sichtbereiches werden nicht weiter betrachtet. Sichtbare Knoten werden in den Bildraum projiziert, sodass für einen Knoten entschieden werden kann, ob der repräsentierte Bereich für die Darstellung ausreichend ist. Ist der Bereich zu groß, wird die Baumstruktur solange in einem Tiefendurchlauf traversiert, bis die ermittelten Knoten nur noch einen kleinen Bereich auf dem Bildschirm abdecken (Levoy & Rusinkiewicz 2000). Dieser durch wenige Pixel dargestellte Bereich kann mit wenigen Punkten vi-

sualisiert werden, obwohl im Datenbestand selbst ggf. mehrere Millionen Punkte vorhanden sind. Abb. 4 veranschaulicht das punktbasierte Rendering für unterschiedliche Schwellwerte.

Der Schwellwert beeinflusst die Anzahl der darzustellenden Punkte sowie die benötigte Zeit für die Darstellung eines Bildes. Wenn der Speicher der Grafikkarte für die Darstellung der Punktmenge nicht ausreicht, vergrößert das Renderingsystem den Schwellwert, sodass weniger Punkte zur Grafikkarte übertragen werden müssen. Der Schwellwert wird auch verändert, wenn der Nutzer in der 3D-Punktwolke navigiert, um eine interaktive Nutzung zu ermöglichen. Zu Beginn des Verfahrens wird der Schwellwert auf 60 Pixel gesetzt. Dieser Wert wird nach den ersten Renderingdurchläufen angepasst, sodass der verfügbare Speicher und die maximale Rechenzeit optimal ausgenutzt werden, um die Interaktivität der Visualisierung zu gewährleisten.

Lücken in der Visualisierung, die bei der Darstellung von weniger Punkten auftreten (Abb. 4A und 4B), können durch die Verwendung von größeren Renderingprimitiven (z. B. Splats) vermieden werden. Details zur Implementierung der Rendering-Technik für die Echtzeit-Visualisierung von massiven 3D-Punktwolken werden in der Arbeit von Richter & Döllner (2010a) vorgestellt. Insbesondere das Verwenden einer *Renderingfront* als Cache für die dargestellten Bereiche der 3D-Punktwolke steigert die Performance des Renderings. Die Renderingfront ist eine Liste, die alle Elemente der räumlichen LOD-Datenstruktur enthält, die für das Rendering eines Bildes verwendet wurden. Basierend auf dem Rendering-Ergebnis eines Bildes kann für das nächste Bild ermittelt werden, welche Bereich der 3D-Punktwolke beibehalten oder in ihrem Detailgrad vergrößert bzw. verringert werden müssen. Darüber hinaus ermöglicht die Renderingfront eine Aufwandsabschätzung im Bezug auf die Renderingdauer sowie den benötigten Speicher während des Renderings.

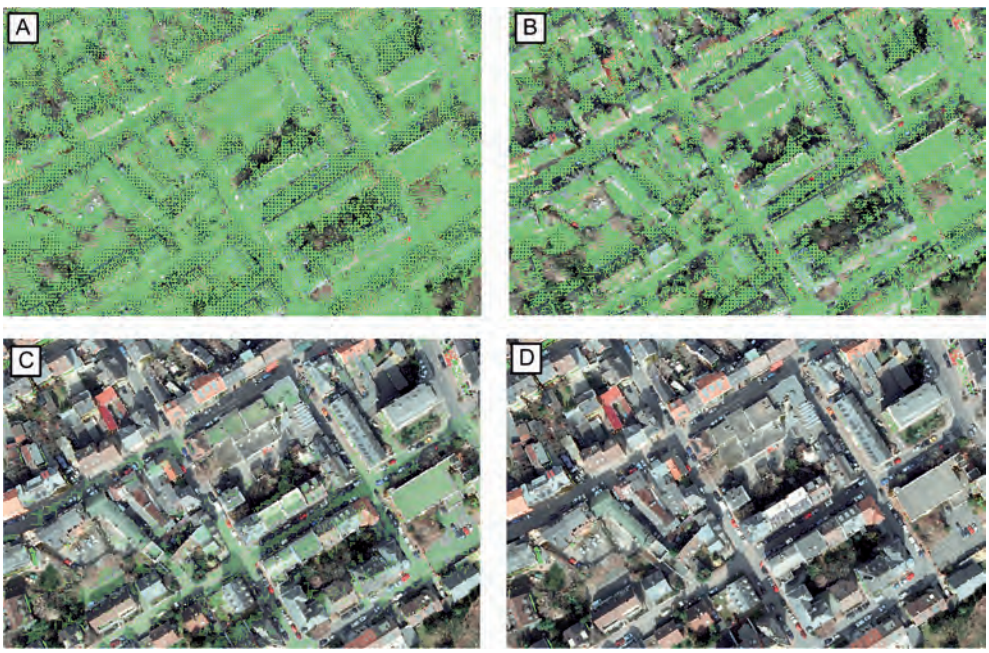

Abb. 4: Darstellung der 3D-Punktwolke mit unterschiedlichen LOD-Stufen in Abhängigkeit der projizierten Knotengröße für die Begrenzung des Tiefendurchlaufes (800 (A), 400 (B), 100 (C) und 1 Pixel (D)).

5 Ergebnisse

In diesem Abschnitt werden die Ergebnisse der Verarbeitung und Visualisierung anhand der Daten einer Befliegung der Stadt Frankfurt am Main vorgestellt. Die erfassten 3D-Punktwolkendaten beinhalten ca. 7,5 Milliarden Punkten und benötigen in einem Binärformat 180 GB Speicher. Die Luftbilder für das Erfassungsgebiet liegen in 2.280 Kacheln vor und benötigen 161 GB Speicher.

Das Einfärben der 3D-Punktwolke benötigt 4,5 Stunden Rechenzeit und ergibt eine 106 GB große Datei (Abb. 1B). Das Überführen in die räumliche LOD-Datenstruktur benötigt 20 Stunden Rechenzeit und erfordert 116 GB Speicher für den Beispieldatensatz (Abb. 1C). Abb. 5 zeigt die komplette 3D-Punktwolke des Frankfurter Stadtgebietes, dargestellt mit ca. 10 Millionen Punkten.

Bei der in Abb. 5 verwendeten Ansicht werden die Daten aus einer größeren Entfernung betrachtet, um einen Überblick über den gesamten Datenbestand zu erhalten. Aufgrund der Entfernung muss der Rendering-Algorithmus die LOD-Datenstruktur nicht bis zu den Blattknoten traversieren, da bereits geringe LOD-Level eine ausreichende Punktdichte für die Visualisierung ermöglichen. Die tatsächlich dargestellten Punkte repräsentieren teilweise mehrere tausend Einzelpunkte, die sich in der LOD-Datenstruktur unterhalb des LOD-Levels in den Blattknoten befinden. Bei Detailansichten, wie z. B. in Abb. 4D, wird die LOD-Datenstruktur in der Regel bis zu den Blattknoten traversiert, sodass alle verfügbaren Punkte für den ausgewählten Ausschnitt der 3D-Punktwolke für die Darstellung vom Rendering-Algorithmus ausgewählt werden.

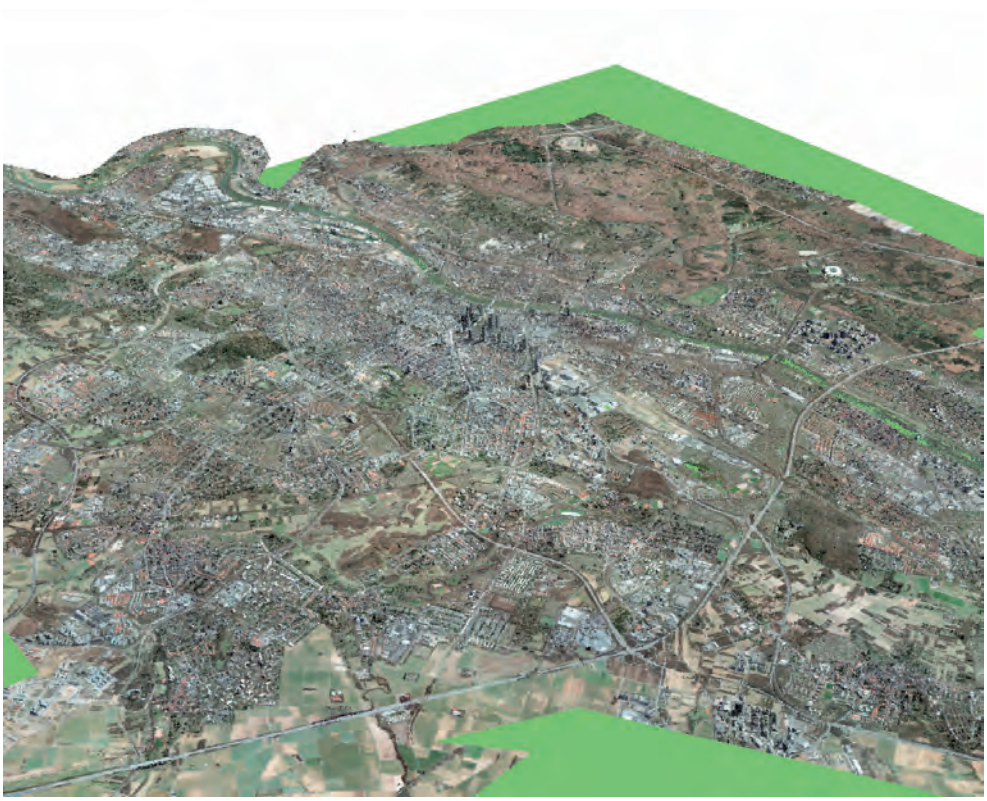

Abb. 5: 3D-Punktwolke des gesamten Frankfurter Stadtgebietes in einer Überblicksvisualisierung als 3D-Punktwolke mit 10 Millionen Punkten.

6 Diskussion und Ausblick

Aufgrund der hinzugefügten thematischen Information aus georeferenzierten Texturen ist eine gezielte Aufbereitung und Gestaltung möglich, die z. B. eine bessere Orientierung sowie Zuordnung von Bereichen innerhalb der 3D-Punktwolke zu realen Objekten erlaubt. Eine Schwäche des Ansatzes resultiert aus der Datenerfassung und zeigt sich dann, wenn 3D-Punktwolkendaten und georeferenzierte Texturdaten nicht zum gleichen Zeitpunkt aufgenommen wurden, sodass beispielsweise Farbinformationen für bewegliche Objekte wie Fahrzeuge und Flugzeuge (Abb. 6A) sowie feingranulare Objekte wie Vegetation (Abb. 6B) nicht korrekt abgebildet werden. In Abb. 6A ist zu erkennen, dass das im Luftbild befindliche Flugzeug nicht durch die 3D-Punktwolke repräsentiert wird und die in der 3D-Punktwolke vorhandenen Flugzeuge mit falschen Farbinformationen versehen werden. Abb. 6B zeigt einen Bereich mit Vegetation. Die Bodenpunkte unterhalb der Baumkrone werden mit Farbinformationen des Baumes eingefärbt.

Aufgrund der Erfassung durch eine Befliegung ist die Punktdichte auf Dachflächen signifikant höher als an Gebäudefassaden. Insbesondere bei hohen Gebäuden entstehen deshalb Löcher an senkrechten Gebäudebestandteilen in der durch die 3D-Punktwoke beschriebenen Oberfläche. Die Visualisierung und damit verbundene Wahrnehmung der Punkte als Gesamtgebäude wird somit aus einer nicht orthogonalen Detailansicht erschwert. Fassaden beispielsweise werden kaum sichtbar dargestellt, sodass Lücken in der Visualisierung entstehen (Abb. 6C). Eine Kombination aus terrestrischen oder mobilen Erfassungsmethoden mit Fahrzeugen und Befliegungen könnte die fehlenden Oberflächeninformationen an Gebäudefassaden teilweise ergänzen.

Die interaktive Visualisierung massiver 3D-Punktwolkendaten eröffnet einen direkten und schnellen Zugang zu den Original-Daten, wodurch nicht nur bessere Werkzeuge zur Analyse und Exploration von 3D-Punktwolken, sondern auch zugehörige Workflows, z. B. bei der Fortführung von 3D-Stadtmodellen, beschleunigt werden können (Richter & Döllner 2010b). Damit erhalten Anwendungen und Systeme Funktionalitäten, auf deren Basis leicht Analysefunktionalität integriert werden kann. In Zukunft arbeiten wir an Werkzeugen, die eine Out-of-Core-Differenzanalyse für 3D-Punktwolkendaten ermöglichen, um z. B. Veränderungen an Bauwerken, Infrastrukturelementen und Geländeoberflä-

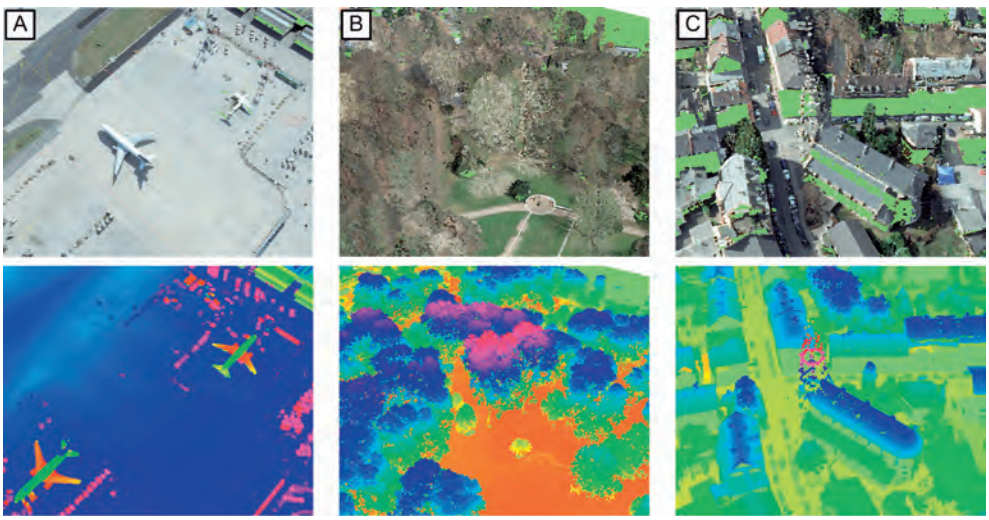

Abb. 6: Ausgewählte Ausschnitte der 3D-Punktwolke mit Farbinformationen (oben) sowie Höheninformationen (unten) um Unterschiede zwischen Geometrie- und Farbrepräsentation sichtbar zu machen.

chen von Städten und Landschaften visualisieren, bestimmen und quantifizieren zu können.

Danksagung

Dieses Forschungsprojekt wurde mit Mitteln des Bundesministeriums für Bildung und Forschung (BMBF) gefördert und ist Teil der InnoProfile Nachwuchsforschergruppe 3D-Geoinformationen (www.3dgi.de). Die verwendeten Datensätze wurden vom Stadtvermessungsamt der Stadt Frankfurt am Main sowie von Fa. virtualcitySYSTEMS, Berlin zur Verfügung gestellt.

Referenzen

Botsch, M., Hornung, A., Zwicker, M. & Kobbelt, L., 2005: High-Quality Surface Splatting on Today's GPUs. – Symposium on Point-Based Graphics: 17–24.

Brenner, C., 2005: Building Reconstruction from Images and Laser Scanning. – International Journal of Applied Earth Observation and Geoinformation **6** (3–4): 187–198.

Dachsbacher, C., Vogelgsang, C. & Stamminger, M., 2003: Sequential point trees. – ACM Transactions on Graphics **22:** 657–662.

Döllner, J., Baumann, K. & Hinrichs, K., 2000: Texturing Techniques for Terrain Visualization. – VIS '00: Conference on Visualization '00: 227–234.

Früh, C., Sammon, R. & Zakhor, A., 2004: Automated Texture Mapping of 3D City Models With Oblique Aerial Imagery. – 3DPVT: 396–403.

Goswami, P., Zhang, Y., Pajarola, R. & Gobbetti, E., 2010: High Quality Interactive Rendering of Massive Point Models using Multi-way kd-Trees. – Pacific Graphics Poster Papers.

Gröger, G. & Plümer. L., 2009: Updating 3D city models: how to preserve geometric-topological consistency. – 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems: 532–535.

Kobbelt, L. & Botsch, M., 2004: A Survey of Point-Based Techniques in Computer Graphics. – Computers & Graphics **28** (6): 801–814.

Kreylos, O., Bawden, G.W. & Kellogg, L.H., 2008: Immersive Visualization and Analysis of LiDAR Data. – International Symposium on Advances in Visual Computing: 846–855.

Lafarge, F., Descombes, X., Zerubia, J. & Pierrot-Deseilligny, M., 2010: Structural Approach for Building Reconstruction from a Single DSM. – IEEE Transactions on Pattern Analysis and Machine Intelligence **32** (1): 135–147.

Levoy, M. & Rusinkiewicz, S., 2000: QSplat: A Multiresolution Point Rendering System for Large Meshes. – ACM SIGGRAPH: 343–352.

MacEachren, A.M., Edsall, R., Haug, D., Baxter, R., Otto, G., Masters, R., Fuhrmann, S. & Qian, L., 1999: Virtual Environments for Geographic Visualization: Potential and Challenges. – 1999 Workshop on new Paradigms in Information Visualization and Manipulation: 35–40.

Pfister, H., Zwicker, M., Baar, J.V. & Gross, M., 2000: Surfels: Surface elements as rendering primitives. – ACM SIGGRAPH: 335–342.

Richter, R. & Döllner, J., 2010a: Out-of-Core Real-Time Visualization of Massive 3D Point Clouds. – 7th International Conference on Virtual Reality, Computer Graphics, Visualization and Interaction in Africa: 121–128.

Richter, R. & Döllner, J., 2010b: Bestandsaktualisierung von 3D-Stadtmodellen durch Analyse von 3D-Punktwolken. – Tagungsband der 3-Ländertagung DGPF, Wien.

Sainz, M., Pajarola, R. & Lario, R., 2004: Points reloaded: Point-based rendering revisited. – Symposium on Point-Based Graphics: 121–128.

Scheiblauer, C., Zimmermann, N. & Wimmer, M., 2009: Interactive Domitilla Catacomb Exploration. – Symposium on Virtual Reality, Archaeology and Cultural Heritage: 65–72.

Tanner, C.C., Migdal, C.J. & Jones, M.T., 1998: The clipmap: a virtual mipmap. – SIGGRAPH 98: 151–158.

Wimmer, M. & Scheiblauer, C., 2006: Instant Points: Fast Rendering of Unprocessed Point Clouds. – Eurographics Symposium on Point-Based Graphics: 129–136.

Zhou, Q. & Neumann, U., 2008: Fast and extensible building modeling from airborne LiDAR data. – Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems: 1–8.

Adresse der Autoren:

Rico Richter und Prof. Dr. Jürgen Döllner, Hasso-Plattner-Institut, Universität Potsdam, Fachgebiet Computergrafische Systeme, D-14482 Potsdam, Tel.: +49-331-5509-170, Fax: +49-331-5509-172, e-mail: rico.richter@hpi.uni-potsdam.de, juergen.doellner@hpi.uni-potsdam.de

Manuskript eingereicht: Januar 2011
Angenommen: März 2011