# Verteilte Prozessierung von ALS-Daten mittels Cloud-Computing - Erste Ergebnisse einer Pilotstudie

Johannes OTEPKA, Gottfried MANDLBURGER, Willi KAREL und Bruno WÖHRER

## 1 Einleitung

Airborne Laserscanning (ALS) hat sich als Standardmethode zur Erfassung topographischer Daten etabliert. Aufgrund der steigenden Messfrequenz moderner Sensoren werden immer höhere Punktdichten erreicht, was zu einem stetigen Anstieg der Punktdatenmengen führt. Dadurch steigen die Anforderungen an die EDV-Infrastruktur – hinsichtlich Hardware und Software – zur Prozessierung der Daten. Im Falle großer Laserscanningprojekte ist die Bearbeitung auf einem Einzelplatzsystem kaum mehr in adäquatem Zeitrahmen möglich.

Eine Alternative dazu bietet eine Verteilung der Rechenleistung auf mehrere Rechner. Als Schlagwort steht gegenwärtig der Begriff „Cloud-Computing“ im Raum. Darunter versteht man allgemein das Bereitstellen von konfigurierbaren Computerressourcen über interne und/oder externe Netzwerke (i.A. das Internet). Mittlerweile bieten zahlreiche Anbieter (Amazon, Google, Mircosoft, etc.) diverse Modelle von „Cloud-Computing“ an, welche Anwender dynamisch und je nach Bedarf nutzen können. Die Steuerung erfolgt zentral über proprietäre Tools oder Programmierschnittstellen (APIs). Darüber hinaus existieren auch im Open-Source-Bereich zahlreiche Tools, die eine verteilte Prozessierung unterstützen; ein Beispiel dafür ist das Open-Source-Projekt *gearman*. Das am Institut für Photogrammetrie und Fernerkundung (I.P.F.) entwickelte Programmsystem OPALS (MANDLBURGER ET AL, 2010) zur Prozessierung von ALS-Daten verfügt über eine C++- und eine Python-Schnittstelle. In der vorliegenden Pilotstudie wird evaluiert, inwieweit sich die hochstehenden Algorithmen von OPALS mit den Verteilungsmöglichkeiten von Python effizient verknüpfen lassen. Dazu wurde am I.P.F. eine „private Wolke“ mittels *gearman* eingerichtet. Die Ergebnisse dieser Studie und die ersten Erfahrungen mit Cloud-Computing hinsichtlich der Anwendung im Bereich des Airborne Laserscanning werden vorgestellt.

## 2 Cloud-Computing

### 2.1 Definition

Cloud-Computing ist derzeit das zentrale Thema für viele Bereiche der gesamten IT-Branche. Hardwarehersteller, Softwareproduzenten, Provider und Dienstleister nähern sich dem Thema naturgemäß auf ganz unterschiedliche Weise, weshalb eine knappe und klare Definition von Cloud-Computing schwierig ist. *Das National Institute for Standards and Technology* (NIST) veröffentliche eine weitergehend anerkannte Definition (MELL & GRANCE, 2009), welche verschiedene Definitionsansätze bündelt (WIKI DE, 2010; CT, 2010). Neben fünf speziellen Charakteristika für Cloud Computing werden drei verschiedenen Servicemodelle und vier Liefermodelle angeführt.

Das Verteilen der Rechenlast auf verschiedene Server (wie beim Grid-Computing oder bei Peer-to-Peer-Netzwerken) ist nur ein Teilaspekt. Das zentrale Ziel beim Cloud-Computing ist das Auslagern von Rechenlast. Anstatt eigene Rechner-, Server- oder Softwareressourcen zu nutzen, werden die Ressourcen von Cloud-Anbietern genutzt. Ein klarer Paradigmenwechsel.

## 2.2. Servicemodelle

Im Allgemeinen werden drei verschiedene Servicemodelle differenziert, die auch unterschiedliche Anwendergruppen ansprechen sollen.

- *Infrastructure-as-a-Service (IaaS):* Stellt dem Benutzer virtuelle Server zur Verfügung. Je nach Bedarf kann um beliebige Server-Instanzen erweitert oder verkleinert werden (z.B. Elastic Compute Cloud (EC2) von Amazon, (AMAZON, 2010)). Der Benutzer hat vollen Zugriff auf die virtuelle Hardware und kann auch selbst Anwendungen installieren.
- *Platform-as-a-Service (PaaS):* Der Entwickler kann Anwendungen hochladen, wobei die Verteilung auf physische Server von der entsprechenden Plattformsoftware erledigt wird. (z.B. Windows Azure von Microsoft (MICROSOFT WINDOWSAZURE, 2010) oder App Engine von Google (GOOGLE APPENGINE, 2010)). Der Nutzer hat wenig bis keine Möglichkeit die Server zu konfigurieren
- *Software-as-a-Service (SaaS):* Stellt dem Nutzer Software zur Verfügung (z.B. Google Docs (GOOGLE DOCS, 2010), Microsoft Skydrive Office Web (MICROSOFT SKYDRIVE, 2010), etc.)

Für die Prozessierung von ALS Daten können unterschiedliche Cloud-Computing Servicemodelle eingesetzt werden. Exemplarisch sollten 2 Szenarien skizziert werden

- *Szenario 1 - ALS-Dienstleister*: Nutzung von *IaaS*-Clouds zur verteilten Prozessierung von ALS-Projekten. Vorteile: Die Anzahl der verwendeten Server-Instanzen kann an die Größe des ALS-Projekts dynamisch angepasst werden. Kosten für Leerlaufzeiten, Hardware und deren Administration entfallen.
- *Szenario 2 - Hersteller von ALS-Software*: Bereitstellen einer *SaaS*-Cloud für die ALS-Prozessierung, welche ihrerseits auf eine *PaaS*- oder *IaaS*-Cloud zurückgreift. Nachdem der Softwarehersteller die Projektauslastung seiner Nutzer nicht kennt, lassen sich benötigte Prozessierungsressourcen nur schwer abschätzen bzw. planen. Bei Verwendung von PaaS- oder IaaS-Clouds können allerdings benötigte Ressourcen dynamisch skaliert werden. Dadurch werden (unbekannte) Maximallasten abgedeckt und Serverleerlaufzeiten vermieden. Zusätzlich erhöht sich die Ausfallsicherheit des Prozessierungsservices.

## 2.3. Liefermodelle

Liefermodelle (Englisch: deployment models) spiegeln den Öffentlichkeitsgrad eines Cloud-Services wieder. Vier Modelle werden unterschieden:

- *public clouds* sind öffentliche Services, die von beliebigen Personen oder Unternehmen genutzt werden können. Hauptproblem dieses Liefermodells ist die Datensicherheit, da der Nutzer Daten außerhalb seiner unmittelbaren Kontrolle bereitstellt. Es gibt bereits einige Ansätze, dieses Problem zu lösen. Höchste Sicherheit mit entsprechender Transparenz für den Nutzer gibt es aber noch nicht.
- Bei *private clouds* befinden sich Dienstleister und Anbieter im selben Unternehmen, weshalb die Problematik der Datensicherheit weitgehend entfällt.

- *hybrid clouds* stellen eine Mischform aus einer privaten und einer öffentlichen „Wolke“ dar. Die Anbindung an ein öffentliches Service sichert dem „private cloud“-Betreiber bessere Ausfallssicherheit und erhöht die Maximallast.
- Bei *community clouds* ist die Serviceinfrastruktur auf mehrere Organisationen verteilt, welche ein gemeinsames Interesse oder Ziel verfolgen. Die „Wolke“ wird entweder von den Organisationen selbst oder von Drittanbietern administriert.

# 3 Die Pilotstudie

Für die Prozessierung von ALS-Projekten mittels Cloud-Computing können die Berechnungszeiten deutlich reduziert werden. Grundsätzlich eignen sich ALS-Daten hervorragend für die verteilte Prozessierung, da viele Berechnungsschritte streifenweise ausgeführt werden (z.B. Qualitätskontrolle). In diesem Fall müssen die Streifen lediglich auf die einzelnen Berechnungsinstanzen verteilt werden. Sind Berechnungsschritte notwendig, welche die gesamten Daten auf einmal verarbeiten, ist eine verteilte Prozessierung nur deutlich aufwändiger zu erreichen. Dazu müssen Berechnungsalgorithmen und die Datenverwaltung eine Verarbeitung in Teilgebieten unterstützen. Zusätzlich wird noch ein Post-Prozess benötigt, der die einzelnen Teilergebnisse zusammensetzt. Aus diesem Grund wurde bei der Pilotstudie nur die streifenweise Verteilung umgesetzt.

Durch die Verknüpfung der LIDAR-Software OPALS mit dem Open-Source-Projekt *gearman* zur verteilten Prozessierung, wurde am I.P.F. eine private Wolke zur ALS-Prozessierung realisiert.

## 3.1. OPALS

OPALS (Orientation and Processing of Airborne Laser Scanning data) ist eine neue Softwarelösung des I.P.F. zur Begegnung der Herausforderungen des komplexen Themas Airborne Laserscanning. OPALS wird eine vollständige Prozessierungskette für ALS-Daten beginnend bei der Full Waveform Signalanlyse (WAGNER ET AL, 2006), über Qualitätskontrolle (RESSL, KAGER & MANDLBURGER, 2008), Streifenanpassung (KAGER, 2004; RESSL, MANDLBURGER & PFEIFER, 2009), Modellierung von Geländekanten (BRIESE, 2004), Filterung der ALS Punktwolke (KRAUS & PFEIFER, 1998) bis hin zum Digitalen Geländemodell (DGM) bereitstellen. Darüber hinaus sind auch weiterreichende Applikationen wie automatische Rekonstruktion von Dachformen (DORNINGER & PFEIFER, 2008) sowie Anwendungen im Forstbereich (HOLLAUS ET AL, 2007 ), in Hydrologie und Hyraulik (MANDLBURGER ET AL, 2009) und viele weitere mehr vorgesehen. Der aktuelle Stand der Implementierung umfasst das Paket *opalsQuality* (Qualitätskontrolle) sowie die teilweise fertig gestellten Pakete *opalsPreprocess* (Full Waveform Signalanalyse, radiometrische Kalibrierung, etc. ), *opalsGeoref* (Streifenausgleichung) und *opalsForest* (fortwirtschaftliche Anwendung).

Konzeptionell besteht OPALS aus einer Reihe von kleinen Modulen mit jeweils klar abgegrenzter Funktionalität, wobei die einzelnen Module sowohl als Kommandozeilenprogramme als auch als Bibliotheksfunktionen ausgeprägt sind. Die Funktionalität jedes einzelnen Moduls steht als Interface für die objekt-orientierte Programmiersprache C++ aber auch für die Scriptsprache *Python* (PYTHON, 2010) zur Verfügung. Die Anbindung an Python ermöglicht die Einbettung der einzelnen OPALS Module in komplexere Scripts, mit welchen individuelle workflows zusammengestellt werden können. Die modulare Struktur von OPALS sowie die Anbindung an Python ermöglichten die rasche Umsetzung einer verteilten Prozessierungskette.

## 3.2. gearman

Das Open-Source-Projekt *gearman* (GEARMAN, 2010) ist ein generisches Framework zur Verteilung von Rechenaufgaben. Neben der Parallelisierung von Aufgaben unterstützt *gearman* einen Lastausgleich zwischen Berechnungsinstanzen und bietet Interfaces zu verschiedenen Programmiersprachen (C, Python, Java, PHP, ...). Das Framework ist plattformunabhängig und erlaubt auch den gemischten Betrieb verschiedener Plattformen (Windows, Linux, MacOS, …)

Zum besseren Verständnis sei kurz die Architektur von *gearman* erläutert, welche sich aus drei Teilen zusammensetzt: (i) der Client erstellt Aufgaben und schickt diese an den Job-Server, (ii) der Job-Server wählt einen geeigneten Worker aus und übermittelt diesem die übertragene Aufgabe und (iii) nach Erledigung der Arbeit retourniert der Worker seine Ergebnisse via Job-Server an den Client.

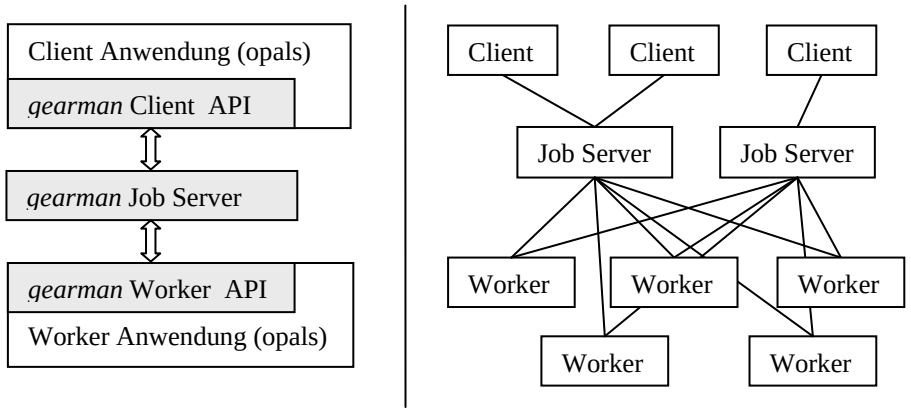


**Abb. 1:** Dreiteilige Architektur von *gearman*

Wie in Abbildung 1 angedeutet, lässt sich ein komplexes *gearman*-Netzwerk aufbauen, das durch die fehlertolerante Implementierung des Frameworks Ausfallsicherheit garantiert.

## 3.3. Anbindung von OPALS an gearman

Nachdem Python die favorisierte Scripting-Sprache für OPALS ist, wurde das Python API von *gearman* zur Anbindung herangezogen. Die bereits vorhandenen Prozessierungsscripts sollten so umgestellt werden, dass sowohl lokal als auch in einem *gearman*-Netzwerk prozessiert werden kann. Die Anpassungen gelangen in kurzer Zeit, sodass die aktuelle Version von OPALS (1.1.0) bereits eine verteilte Prozessierung (im Beta-Stadium) unterstützt. Die momentane Implementierung setzt noch voraus, dass der Client und sämtliche Worker die LIDAR-Daten unter einheitlichen Pfaden erreichen.

## 3.4. Aufbau der Pilotstudie

Die Pilotstudie sollte zeigen, wie gut der vorgestellte Ansatz mit zusätzlichen Recheninstanzen skaliert bzw. ob es sich um einen praktikablen und einfach zu bedienenden Ansatz handelt. Als Datensatz wurde eine ALS-Befliegung von Schloss Schönbrunn aus dem Jahr 2004 bestehend aus 7 Streifen (ca. 1 Mio. Punkte pro Streifen) ausgewählt. Für dieses Testgebiet sollten streifenweise Oberflächenmodelle (DOM) sowie ein mosaikiertes Gesamtmodell berechnet werden. Für sämtliche Modelle waren darüber hinaus je eine Höhenko-

dierung und eine Schummerung zu erstellen. Die ausgewählten Berechungsaufgaben sind typische Teilschritte für die Qualitätskontrolle und -dokumentation eines ALS-Projektes.

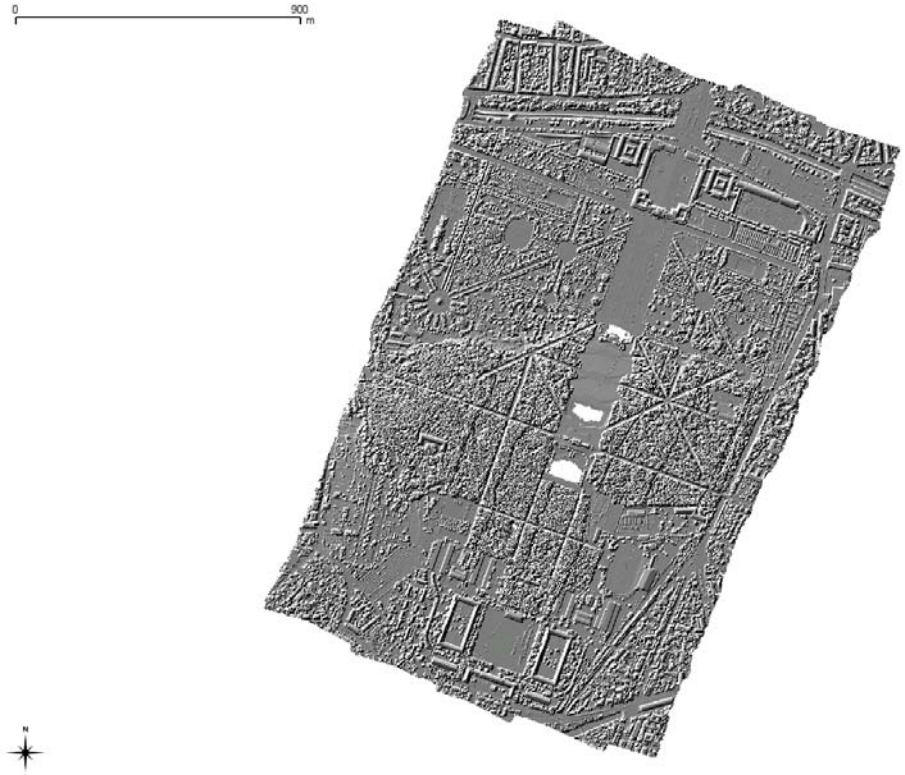


**Abb. 2:** DOM-Schummerung des Testgebietes (Schloss Schönbrunn mit Schlosspark)

Für die Umsetzung wurden die OPALS-Module *opalsImport* (Datenimport), *opalsGrid* (DOM Berechnung), *opalsZColor* (Höhenkodierung), *opalsShade* (Schummerung) und *opalsAlgebra* (Mosaikierung) verwendet.

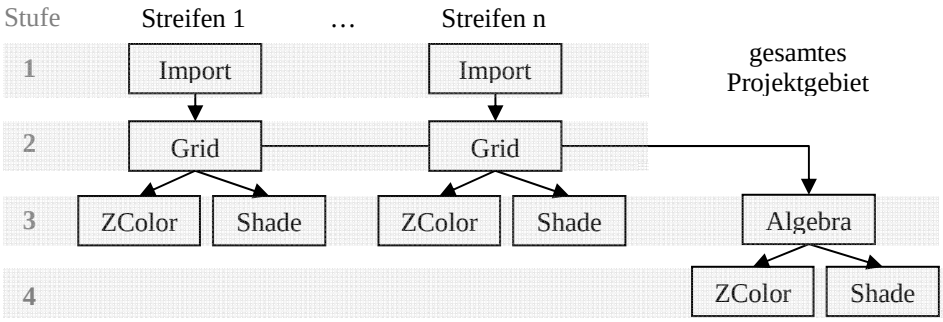


**Abb. 3:** Prozessierungskette nach Parallelisierungsmöglichkeiten ausgerichtet

Wie in Abbildung 3 ersichtlich, lässt sich die Prozessierungskette in vier Stufen zerlegen. Jede Stufe ist von der jeweils vorhergehenden abhängig, aber alle Berechnungsaufgaben innerhalb einer Stufe können vollständig parallelisiert werden. Damit wird klar ersichtlich, welch großer Performancegewinn mit verteilter Prozessierung erreichbar ist.

Anhand unterschiedlicher Konfigurationen (Anzahl der Recheninstanzen) wurde der praktische Performancegewinn für das vorgestellte Testprojekt evaluiert. Die genauen Ergebnisse werden im Folgenden beschrieben.

## 3.5 Ergebnisse und Erfahrungen

Auf einem moderat ausgestatten Notebook (Dual-Core mit 2.4 GHz) benötigt die Berechnung des Testprojekts 6'43" (Datenspeicherung auf lokaler Disk). Diese Berechnungszeit wurde als Bezugsbasis für die Tabelle 1 herangezogen. Für die verteile Prozessierung muss alles über einen Fileserver abgewickelt werden, weshalb Fileserverperformance und Netzwerkgeschwindigkeit nennenswerten Einfluss auf die Prozessierungsdauer haben. Das I.P.F. verfügt über einen sehr schnellen Fileserver (160 TB) und ein GBit-Netzwerk, sodass der zusätzliche Overhead im Mittel (je nach aktueller Auslastung) bei nur 10 % liegt (siehe „Notebook mit Fileserver“ in Tabelle 1).

Bei der verteilten Prozessierung lief ein *gearman*-Worker am besagten Notebook und die anderen Worker auf ähnlicher Hardware. Wie in Tabelle 1 aufgelistet, lässt sich die Prozessierungszeit mit 4 *gearman*-worker um rund 2/3 reduzieren, was ein beachtliches Ergebnis darstellt.

An dieser Stelle sei darauf hingewiesen, dass die vorgestellten Performancegewinne keine absolute Gültigkeit haben, sondern stark von Projektgebiet, Berechnungsaufgaben, Hardware und Netzwerkauslastung beeinflusst werden. So kann z.B. eine besser ausgerüstete Workstation (Quad-Core Maschine mit Hyperthreading (=8 logische Kerne) und schneller Festplatte) das Testprojekt in 3'02" prozessieren (45 % der Referenzzeit). Greift dieser Rechner per Fileserver auf die Daten zu, steigt der Overhead von 10% auf knapp 20%. D.h. je schneller der *gearman*-Worker, desto eher wird das Netzwerk zum Flaschenhals.

**Tabelle 1:** Berechnungszeiten in Abhängigkeit der Berechnungsinstanzen

| **Prozessierungsart** | **Berechnungsinstanzen** | **Berechnungszeiten** | |
|---|---|---|---|
| | | **Absolut** | **Relativ** |
| Standard (ohne *gearman*) | Notebook lokal (Referenz) | 06'43" | 100% |
| | Notebook mit Fileserver | 07'23" | 110% |
| *gearman* (mit Fileserver) | 2 *gearman*-Worker | 04'25" | 66% |
| | 3 *gearman*-Worker | 03'23" | 50% |
| | 4 *gearman*-Worker | 02'23" | 35% |

Von der praktischen Seite stellte sich die verteilte Prozessierung mit *gearman* als unproblematisch heraus. Beim Aufruf der OPALS-Skripts musste lediglich auf die Dateipfade (computerunabhängig) entsprechend Rücksicht genommen werden. Sobald diese Hürde aber genommen war, liefen die Skripts genau so zuverlässig wie bei der rein lokalen Prozessierung durch.

# 4 Diskussion & Ausblick

Durch die Anbindung von OPALS an *gearman* konnte am I.P.F. eine „private Wolke“ zur verteilten ALS-Prozessierung realisiert werden. Momentan müssen die Berechnungsinstanzen (*gearman*-Worker) noch händisch gestartet werden, die Implementierung eines automatischen (Windows-)Services ist derzeit in Arbeit. Nicht desto trotz konnte mit der Pilotstudie das große Performancepotential aufgezeigt werden. Möglichkeiten zur Optimierung ist noch vorhanden und sollen in weiterführenden Teststudien ausgelotet werden.

Der präsentierte Prozess stellt nur den erst Schritt zur ALS-Prozessierung mittels Cloud-Computing dar. Bei einem öffentlichen über das Internet laufenden Cloud-Service gibt es momentan noch zwei problematische Punkte: Datenmenge und Datensicherheit. Die bei ALS-Projekten anfallende Datenmenge kann selbst bei lokalen Netzwerken zu Engpässen führen. Die Sinnhaftigkeit eines ALS-Cloud-Services steht und fällt mit der Verfügbarkeit einer sehr schnellen Internetverbindung. Die von Providern angebotene Bandbreite steigt stetig an, weshalb sich dieses Problem in Zukunft deutlich entschärfen wird. Andererseits ist allerdings durch die Entwicklung in der Sensortechnik mit steigenden Datenmengen zu rechnen.

Während schnelle Internetanbindungen z.B. durch Bündelung mehrer Anschlüsse realisierbar sind, gilt das Thema Datensicherheit als noch weitgehend ungelöst. Vor allem für Unternehmen ist dies ein Hauptgrund, um auf externe Cloud-Services zu verzichten. Serviceanbieter haben diese Problematik erkannt und bieten bereits erste Ansätze zur Entschärfung. In diesem Punkt ist aber sicherlich noch Forschungs- bzw. Standardisierungsarbeit notwendig.

Momentan gibt es noch keine Standards zur Steuerung von Cloud-Services. Dies ist vor allem aus Anwendersicht unbefriedigend, da jeder Service-Provider proprietäre Tools oder APIs zur Verfügung stellt. Hat man sich mal für ein spezielles Cloud-Service entschieden, ist ein Service-Wechsel nur mit viel Aufwand umsetzbar. Nicht zuletzt deshalb wurde die OPEN DATA CENTER ALLIANCE (2010) gegründet, in dem über 70 Datenzentrumsbetreiber Mitglied sind. Ziel dieser Organisation ist es, Standards und Normen für Cloud-Services zu definieren und damit die Verbreitung dieser – auch für geodätische Anwendungen – zukunftsträchtigen Technologie voranzutreiben.

# Literatur

BRIESE, C. (2004): Breakline Modelling from Airborne Laser Scanner Data, Dissertation, Technische Universität Wien.

CT (2010): Prozessorgeflüster. c’t, Magazin für Computer Technik, 24/2010, S. 18ff

DORNINGER, P. & PFEIFER, N., (2008): A Comprehensive Automated 3D Approach for Building Extraction, Reconstruction, and Regularization from Airborne Laser Scanning Point Clouds, Sensors, 8(11), 7323-7343.

HOLLAUS, M, W. WAGNER, B. MAIER & K. SCHADAUER (2007): Airborne Laser Scanning of Forest Stem Volume in a Mountainous Environment, Sensors, 7 / http://www.mdpi.com/journal/sensors (2007), 8; Letzter Zugriff: 2010-12-06

KAGER, H. (2004): Discrepancies Between Overlapping Laser Scanning Strips - Simultaneous Fitting of Aerial Laser Scanner Strips, in: International Archives of Photogrammetry and Remote Sensing, XXXV, B/1, 555–560, Istanbul, Turkey.

KRAUS, K. & PFEIFER, N. (1998): Determination of Terrain Models in Wooded Areas with Airborne Laser Scanner Data, ISPRS Journal of Photogrammetry and Remote Sensing, 53, 193–203.

MANDLBURGER, G., C. HAUER, B. HÖFLE, H. HABERSACK, AND N. PFEIFER (2009): Optimisation of LiDAR derived terrain models for river flow modelling. Hydrology and Earth System Sciences 13(8), 1453–1466.

MANDLBURGER, G., OTEPKA, J., KAREL, W., WÖHRER, B., WAGNER, W., PFEIFER, N. (2010): OPALS (Orientation and Processing of Airborne Laser Scanning data) - Konzept und Anwendungsbeispiele einer wissenschaftlichen Laserscanning Software, Dreiländertagung - 30. Wissenschaftlich-Technische Jahrestagung der DGPF, Wien

MELL, P. & GRANCE T. (2009): The NIST Definition of Cloud Computing, Version 15, http://csrc.nist.gov/groups/SNS/cloud-computing, Letzter Zugriff: 2010-12-06

RESSL, C., KAGER, H. & MANDLBURGER, G. (2008): Quality Checking Of ALS Projects Using Statistics Of Strip Differences, IAPRS, Vol. XXXVII. Part B3b, ISSN: 1682-1750; 253 - 260.

RESSL, C., MANDLBURGER, G. & PFEIFER, N. (2009): Investigating Adjustment Of Airborne Laser Scanning Strips Without Usage Of GNSS/IMU Trajectory Data, IAPRS, Vol. XXXVIII, Part 3/W8 (2009), ISSN: 1682-1750; 195 - 200.

WAGNER, W, A. ULLRICH, V. DUCIC, T. MELZER & N. STUDNICKA (2006): Gaussian Decomposition and Calibration of a Novel Small-Footprint Full-Waveform Digitising Airborne Laser Scanner, ISPRS Journal of Photogrammetry and Remote Sensing, 60 (2006), 2; 100 - 112.

## Web-Referenzen

AMAZON (2010): http://aws.amazon.com/ec2, Letzter Zugriff 2010-12-06

GEARMAN (2010): http://gearman.org, Letzter Zugriff: 2010-12-06

GOOGLE APPENGINE (2010): http://code.google.com/appengine, Letzter Zugriff: 2010-12-06

GOOGLE DOCS (2010): http://www.google.com/google-d-s/documents, Letzter Zugriff: 2010-12-06

MICROSOFT SKYDRIVE (2010): http://explore.live.com/windows-live-skydrive, Letzter Zugriff: 2010-12-06

MICROSOFT WINDOWSAZURE (2010): http://www.microsoft.com/windowsazure, Letzter Zugriff: 2010-12-06

OPEN DATA CENTER ALLIANCE (2010): http://www.opendatacenteralliance.org, Letzter Zugriff: 2010-12-06

PYTHON (2010): http://www.python.org, Letzter Zugriff: 2010-12-06

WIKI DE (2010): http://de.wikipedia.org/wiki/Cloud_Computing, Letzter Zugriff: 2010-12-06