Julia Hauczinger, BSc

Generierung von digitalen Geländemodellen auf Basis von Laserscan - Punktwolken für das Land Steiermark am Fallbeispiel Sankt Lorenzen im Paltental

MASTERARBEIT

zur Erlangung des akademischen Grades

Master of Science

Masterstudium Geospatial Technologies

eingereicht an der

Technischen Universität Graz

Betreuer

Ass.Prof. Dipl.-Ing. Dr.techn. Konrad Rautz

Institut für Geodäsie

Graz, Juni 2015

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Datum

Unterschrift

Danksagung

Ganz besonders möchte ich mich bei meinem Freund Helmut bedanken, der immer für mich da war, mich bei allem unterstützt hat und mich durch meine ganze Studiumzeit begleitet hat. Für die Betreuung meiner Masterarbeit bedanke ich mich bei Herrn Ass.Prof. Dipl.-Ing. Dr.techn. Konrad Rautz. Weiterer Dank gilt Herrn Dipl.-Ing. Rudolf Hütter und Frau Nicole Kamp, MSc, die es mir ermöglichten, diese Masterarbeit für das Referat Statistik und Geoinformation des Landes Steiermark zu schreiben.

Zusammenfassung

Diese Masterarbeit wurde im Auftrag der Abteilung 7 Landes- und Gemeindeentwicklung, Referat Statistik und Geoinformation des Landes Steiermark durchgeführt und beschäftigt sich mit dem Thema der digitalen Geländegenerierung aus LAS Dateien. Zur Evaluierung der Ergebnisse wurde das Gebiet um den Ort Sankt Lorenzen im Paltental in der Steiermark gewählt, da es mehrere typische Geländestrukturen besitzt, wie zum Beispiel ein flaches Talgebiet, eine Mittelgebirgsregion und eine Hochgebirgsregion, und somit die Genauigkeit der generierten DTMs bei unterschiedlichen Topographien vergleichen zu können.

Die Inverse Distance Weigthing (IDW) Methode stellt sich in verschiedenen Studien unter anderem als geeigneter Interpolator für LIDAR Daten dar. Die IDW Methode lässt sich ebenfalls ohne Probleme programmieren und damit ein Tool zur Geländeinterpolation aus LAS Daten entwickeln.

Im Vergleich zu den lizenzpflichtigen Tools, wie LASTools und LP360 konnte vor allem die programmierte IDW Methode gute Ergebnisse zeigen. Der Root-mean-square Fehler (RMSE) liegt bei komplexem Gelände nur im Zentimeterbereich höher als bei den lizenzpflichtigen Tools. Bei flachem Gelände liefert die IDW Methode genauso gute Ergebnisse. Der Unterschied zwischen allen Tools sieht man vor allem in Fällen von Geländeänderungen, von kleinen Gräben bis hin zu großen Höhendifferenzen mit Steilhängen, sowie bei fehlender bis geringer LIDAR Bodenpunktabdeckung, wo sehr unterschiedlich von den Tools interpoliert wird.

Abstract

This master thesis was an assignment of the “Abteilung 7 Landes- und Gemeindeentwicklung, Referat Statistik und Geoinformation des Landes Steiermark“ and deals with the subject of digital terrain generation from LAS files. The area around the town of Sankt Lorenzen im Paltental in Styria was selected to evaluate the results, because it has several typical terrain textures, such as a flat valley area, a low mountain range and a high mountain region. Therefore, the accuracy of the generated DTM can be compared with different topographies. The Inverse Distance Weigthing (IDW) method was mentioned in various studies as appropriate interpolator for LIDAR data. The IDW method can be programmed very easily to develop a tool for terrain interpolation from LAS data.

Especially the programmed IDW method showed very good results compared to the licensed tools like LASTools and LP360. The root-mean-square error (RMSE) of the IDW method is only a few centimeters bigger than that of the licensed tools for complex terrain. The IDW method gives equally good results in flat terrain. The differences between the tools mentioned can be seen mainly when working with terrain changes, from small trenches to large height differences with steep slopes, and in cases of missing and bad LIDAR ground point cover, which the tools interpolate very differently. The programmed IDW delivered very good results in the difference model of debris flow 2011-2012 compared to the existing model.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ALS	Airborne Laserscanner
DEM	Digital Elevation Model
DGM	Digitale Geländemodelle (englisch: Digital Terrain Model)
DHM	Digitales Höhenmodell (englisch: Digital Elevation Model)
DTM	Digital Terrain Model
GIS	Geographische Informationssysteme
GPS	Global Positioning System
IDW	Inverse Distance Weighting (Inverse Distanzgewichtung)
IMU	inertial measurement unit
KT	Kringing with trend model
LIDAR	Light Detection and Ranging
MAE	Mean Absolut Error
ME	Mean Error
MRBF	Multiquadratic radial basis function
NN	Nearest-Neighbor Interpolation
OK	Ordinary Kriging
RMSE	Root Mean Square Error
RST	Regularized Spline with Tension
TIN	Triangulated Irregular Network
UG	Untersuchungsgebiet
UK	Universal Kriging

1 Einleitung

Digitale Geländemodelle (DGM, englisch: Digital Terrain Model, DTM) spielen eine sehr wichtige Rolle in der Modellierung von verschiedenen geographischen Phänomenen. Neben den traditionellen Methoden zur Generierung von DTMs, wie zum Beispiel der Photogrammetrie, hat sich in den letzten Jahren das LIDAR (Light Detection and Ranging) als eine sehr genaue Methode für die Akquisition von dichten Höhendaten etabliert. Die Steiermark wurde vollständig mit hochgenauen 3D Airborne Laserscanner (ALS) überflogen und die Laserscandaten liegen flächendeckend für das ganze Bundesland vor. Ergebnis eines Airborne Laserscannings ist eine sogenannte Punktwolke, die sowohl Objekte als auch den Boden repräsentiert (mehr dazu in Kapitel 2.1.). Anschließend werden die Punkte in Boden- und Nichtbodenpunkte sortiert. Nach dieser Klassifizierung der Bodenpunkte stellt die Punktwolke das Gelände dar, welche dann zu einem DTM generiert werden kann.

Für die verschiedenen Anwendungen, wie zum Beispiel in der Wasserwirtschaft, Raumplanung, Katastrophenschutz und in vielen weiteren Bereichen, ist es wichtig, dass die Verarbeitung von hochauflösenden Airborne LIDAR Bodenpunkten in ein Digitales Geländemodell sehr genau ist.

Die Wahl der Interpolationsmethode zur DTM Generierung aus der bereits klassifizierten LIDAR Punktwolke ist ein entscheidender Faktor, der die DTM Genauigkeit beeinflussen kann (GUO ET AL., 2010). Daher ist es notwendig, dass man aus der Vielzahl an Interpolationsmöglichkeiten geeignete Methoden zur DTM Generierung in Hinblick auf Genauigkeit und Rechenzeit auswählt. Überdies benötigt die Geländeerstellung von LIDAR Daten oft kostenpflichtige Lizenzen. Daher ist die Entwicklung eines Tools mit einer geeigneten Interpolationsmethode zur DTM Interpolation aus LAS Daten erstrebenswert.

1.1 Zielsetzung

Primärziel der Masterarbeit ist die Generierung von DTMs aus bereits klassifizierten LIDAR Punktwolken in LAS Format.

Sekundärziel dieser Masterarbeit wird sein, ein Tool für die Abteilung 7, Landes- und Gemeindeentwicklung, Referat Statistik und Geoinformation des Landes Steiermark zu entwickeln, das aus einer bereits klassifizierten LIDAR Punktwolke in LAS Format DTMs

interpolieren kann. Das Tool soll in Python entwickelt werden, damit es unter anderem in ArcGIS integrierbar ist.

1.2 Forschungsfragen

Für das Primärziel soll die Masterarbeit folgende Forschungsfragen beantworten:

- Welche Interpolationsmöglichkeiten gibt es zum Generieren von LIDAR?
- Welche Methode ist die optimalste für die Interpolation von Lidar Punktwolken zu einem DTM?

Für das Sekundärziel sollen folgende Fragen beantwortet werden:

- Welche Verarbeitungsmöglichkeiten gibt es derzeit in Python für klassifizierte Punktwolken in LAS Format? Welche Möglichkeiten gibt es, um DTMs mit Python zu generieren?
- Kann mit Hilfe von Python ein Tool zur Geländeinterpolation aus LAS Daten entwickelt werden oder ist die Verwendung von anderen Softwarepaketen besser geeignet?

Zur Evaluierung sollen die entwickelten Tools mit anderen Softwarelösungen verglichen werden. Dabei soll folgende Frage beantwortet werden:

- Wo liegen die Unterschiede zwischen den auf verschiede Art und Weise generierten DTMs?

2 Grundlagen

2.1 LIDAR

LIDAR (Light Detection and Ranging) ist ein aktives Fernerkundungsverfahren. Ein Laser, welcher den optischen Teil des elektromagnetischen Spektrums zwischen dem ultravioletten und dem nahen Infrarot-Bereich benutzt, sendet Strahlung in Pulsen oder kontinuierlich durch eine fokussierende Optik aus. Ein optisches System fokusiert dann die zurückkommende Strahlung auf einem Detektor. Es gib drei verschiedene Gruppen und Anwendungen von LIDAR-Systemen (ALBERTZ & WIGGENHAGEN, 2009, S. 125):

- LIDAR-Systeme zur räumlichen Erfassung von Stoffen in der Atmosphäre und im Meer
- Terrestrisches Laserscanning
    - Diese Systeme werden fest aufgestellt und ein mechanisches oder elektronisches Element erzeugt ein Scanraster in zwei Richtungen.
- Airborne Laserscanning

2.1.1 Airborne Laserscanning

Das Airborne Laserscanning ist ein aktives Fernerkundungsverfahren zur punktweisen Erfassung der Geländeoberfläche mit einem Laserscanner von einem Flugzeug aus. Dabei dient der Laserstrahl zur Messung der Entfernung zwischen dem Sensor an Bord des Flugzeuges und der Geländeoberfläche (ALBERTZ, 2009, S. 54). Der Laser arbeitet für topographische Anwendungen meist im Nahen Infrarotbereich, zwischen 0,5 und 1,5 µm des elektromagnetischen Spektrums (ALBERTZ & WIGGENHAGEN, 2009, S. 127). Die Scantechnik erfasst durch die Kombination mit der Bewegung des Flugzeuges quer zur Flugrichtung einen Geländestreifen in zahlreichen Messpunkten (ALBERTZ & WIGGENHAGEN, 2009, S. 127; ALBERTZ, 2009, S. 54; JENSEN, 2009). Die räumliche Lage dieser erfassten Geländepunkte wird durch die GPS/ INS-Systeme bestimmt, indem die Orientierung des Sensors mit hoher Genauigkeit gleichzeitig ermittelt wird (Abbildung 1). Die so gewonnenen Daten ermöglichen dann die Berechnung der räumlichen Koordinaten der vom Laserstrahl reflektierten Geländepunkte. Ergebnis dieser Berechnung ist eine sogenannte Punktwolke, wobei jeder Punkt die Raumkoordinaten x, y, und z besitzt (ALBERTZ, 2009, S. 54).

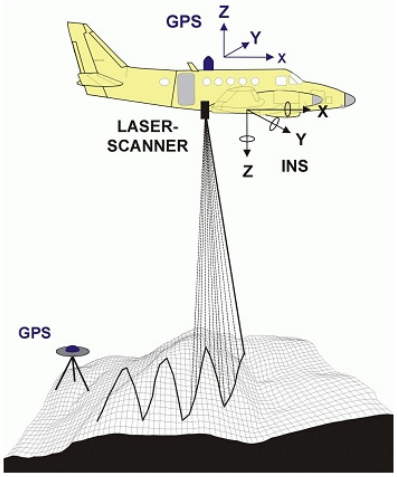

Abbildung 1: Prinzip einer Airborne Laserscan Aufnahme (MAGISTRAT DER STADT WIEN, 2014)

2.1.2 Prinzip des Airborne Lasersanners

Die Impulse, die durch den Laserscanner ausgesendet werden, werden sowohl von Objekten, wie zum Beispiel Bäume oder Gebäude usw., als auch von der Bodenoberfläche reflektiert. Ein ausgesendeter Laserpuls kann - abhängig vom lokalen Gelände auf dem ein Laserfußabdruck auftrifft - einfach oder auch mehrfach reflektiert und zum LIDAR-Sensor zurückgegeben werden. Abbildung 2 zeigt, wie aus einem einzigen ausgesendeten Laserimpuls mehrere Rückgabeimpulse entstehen können:

Die gesamte Energie innerhalb des Laserimpulses „A“ interagiert mit dem Boden. Trifft ein ausgesendeter Laserimpuls jedoch mehrere Reflexionsoberflächen (Laserpuls „B“ in der Abbildung), zum Beispiel auf zwei unterschiedliche Höhen von einem Baum und dann auf den Erdboden, dann wird der Laser ebenso oft reflektiert (JENSEN, 2009, S. 339). Die erste Reflexion (1st return) des Laserpulses wird dem höchsten Merkmal einer Landschaft zugeordnet, zum Beispiel einem Baumwipfel oder dem Dach eines Gebäudes. Die letzte Rückgabe (last return) stammt meistens vom Erdboden. Jedoch muss die letzte Rückgabe nicht immer vom Boden stammen, denn ein ausgesendeter Puls kann auch auf einen dicken Ast treffen und den Boden gar nicht erreichen, so stammt dann die letzte Rückgabe nicht vom Boden, sondern von dem Ast (ESRI, 2014).

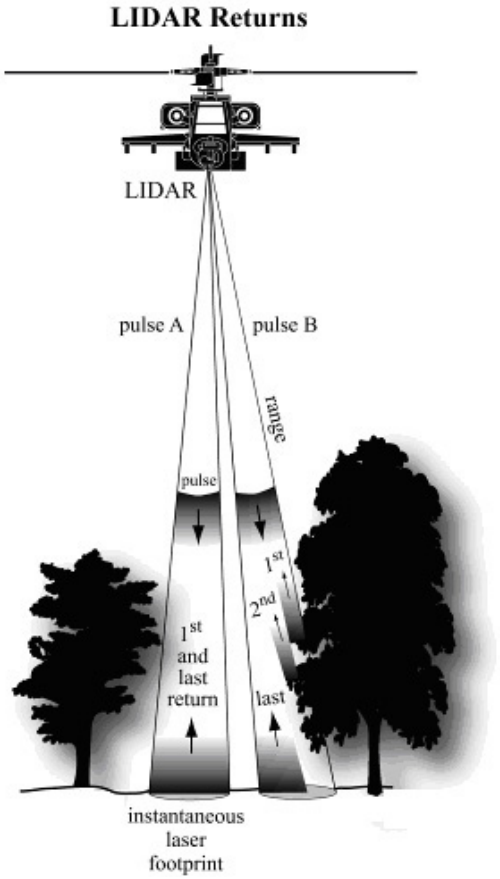

Abbildung 2: Rückgabe des LIDAR Lasers (JENSEN, 2009, S. 339)

2.1.3 Vor- und Nachteile von Laserscandaten

Die LIDAR Technik ist gegenüber den traditionellen Methoden, wie zum Beispiel der Photogrammetrie, zur Generierung von DTMs überlegen. Das liegt vor allem an den vielen Vorteilen der LIDAR Technik (PFEIFER, 2003; MENG ET. AL., 2010):

1. Die hohe Punktdichte der LIDAR Punktwolke ermöglicht die Generierung von hochgenauen und hochauflösenden Digitalen Höhenmodellen (DHM, englisch: Digital Elevation Model DEM), was eine sehr genaue Beschreibung der Geländefläche ermöglicht.
2. Oberflächenmerkmale können durch eine Höhenkontextanalyse von LIDAR Punkten extrahiert werden. Dies ermöglicht Abbildungen von Oberflächenmerkmalen, wie zum Beispiel Häuser, Bäume oder sogar Stromleitungen.
3. Es ist möglich auch geringe Höhenänderungen durch die dichte LIDAR Punktwolke zu identifizieren, dies können einmal Änderungen in der Oberfläche sein oder auch Variationen in der Vegetationsdecke.
4. Die Vegetationsstruktur kann bestimmt werden, weil die LIDAR Pulse die Baumkronendecke durchdringen kann und meherere Rückgaben liefert.
5. LIDAR ermöglicht es die Bodenhöhe abzubilden, sogar in Regionen mit einer sehr hohen Vegetationsdichte, wie zum Beispiel in bewaldeten Gebieten.

6. LIDAR ermöglich einen hohen Automatisierungsgrad, der von der Datenerfassung bis zur Erstellung des Digitalen Geländemodells reicht.
7. Die Genauigkeit in der Höhe liegt bei ca. 10 cm.
8. Das aktive System ermöglicht die Messungen während der Nacht oder über texturlosen Bereichen (z.B. Schnee).

Nachteil beim LIDAR ist jedoch, dass die Qualität des Geländemodells mit bewaldeten Gebieten mit zunehmender Dichte der Vegetation abnimmt, und zwar durch die geringere Anzahl der Punkte, die am Boden gemessen werden. Dadurch wird auch die Beschreibung der Geländefläche unzuverlässiger. Weiterhin könne auch keine Geländekanten gemessen werden (PFEIFER, 2003).

2.1.4 Verarbeitung von Laserscandaten

WEHR & LOHR (1999) beschreiben die Verarbeitung der Laserscandaten folgendermaßen (siehe Abbildung 3): Nach dem Vermessungsflug sind zwei Datensätze verfügbar, die Positionsdaten und die Entfernungsmessungen mit dem Scanwinkel. Zusätzlich werden Kalibrierungsdaten benötigt, die zum Beispiel die Position des Laserscanners zum IMU beschreiben, oder die Position des IMU hinsichtlich zum GPS. Aus diesen drei Datensätzen können die Punkte im WGS84 Koordinatensystem berechnet werden. Danach können die Laserentfernungen von WGS84 in ein gewünschtes lokales Kartenkoordinatensystem umgewandelt werden. Das Ergebnis ist dann eine Wolke von unstrukturierten und in Höhe und Position zufällig verteilten Laserpunkten. Die Höhenmessungen werden hinsichtlich ihrer Position sortiert. Nach dem Sortieren müssen die Punkte in Boden- und Nichtbodenpunkte, wie Gebäude und Vegetation, klassifiziert werden. Dazu wurden in der Vergangenheit eine Vielzahl von Filtern entwickelt (VOSSELMAN, 2000; SITHOLE, 2001; AXELSSON, 2000; KRAUS & PFEIFER, 2001; BROVELLI ET AL., 2002 und viele andere). Anschließend werden die Punkte zu einem Raster interpoliert. Entweder können die Raster zu einem sogenannten Digitalen Oberflächenmodell interpoliert werden, welches nicht nur das Gelände beinhaltet, sondern auch Gebäude, Vegetation und andere Oberflächenmerkmale, oder die LIDAR Punkte werden durch die Filterung in Nichtbodenpunkte zu einem digitalen Geländemodell interpoliert. Die Interpolation von LIDAR Daten zu einem DTM wird in Kapitel 3 genauer beschrieben.

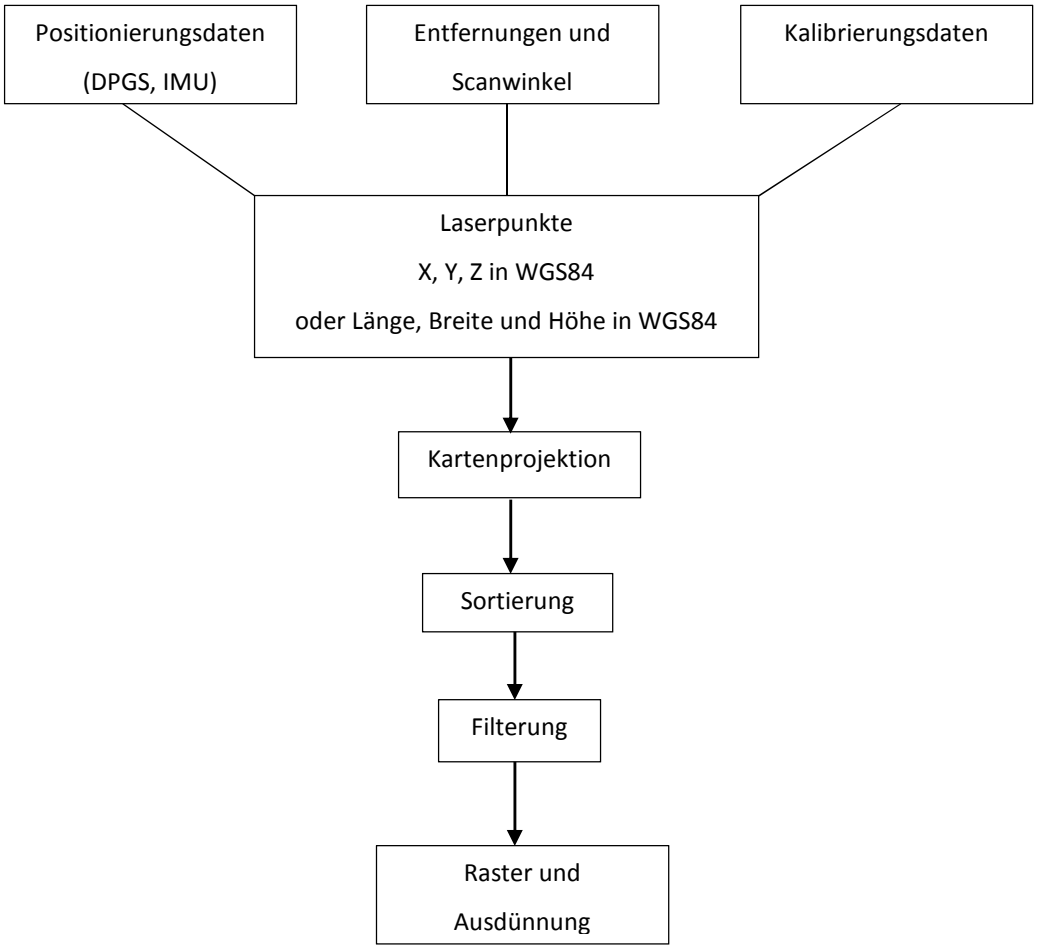

Abbildung 3: Typischer Verlauf der Verarbeitung von Laserscandaten (nach WEHR & LOHR, 2000)

2.1.5 Produkte und Anwendungen

Aus der LIDAR Punkwolke können, wie in 2.1.4 beschrieben, entweder ein DTM oder ein DSM generiert werden. Weitere Anwendungen finden LIDAR Daten beziehungsweise die daraus generierten Rasterdaten in weiteren Bereichen (LAND STEIERMARK, 2010):

- Wasserwirtschaft für Überflutungssimulationen und Abflussberechnungen
- Schutzmaßnahmen und die Gefahrenzonenplanung im Bereich der Wildbach- und Lawinenverbauung
- Verkehrswesen zur Projektvorplanung im Straßenbau
- Umweltschutz für Lärmemissionsberechnungen
- Schadstoffausbreitungssimulationen
- Energieplanung zur Berechnung Objektgenauer Besonnungs- und Beschattungsverhältnisse im Zusammenhang mit Solarenergie
- Forstwesen für die Projektierung und Erfassung von Forststraßen, die Sicherung des Schutzwaldes
- Ermittlung von Bewuchsdaten, insbesondere zur Ermittlung von Biomasse
- Modellierung der Bodenerosion in der Landwirtschaft

- Naturschutz für Renaturierungsmaßnahmen
- Raumplanung als Planungsinstrument bzw. zur Modellierung von Planungsvarianten
- Katastrophenschutz für ein Naturgefahrenmanagement bei Rutschungen, Bergstürzen, Massenbewegungen sowie zur Lawinensimulation

2.2 LASTools und LP360

LASTools ist eine Sammlung von hocheffizienten, leicht skriptbaren, multi-core Kommandozeilentools, um mit LIDAR Daten zu arbeiten. Es gibt Tools, die LIDAR Daten klassifizieren, konvertieren, filtern, rastern, triangulieren, ausschneiden und vieles mehr können. Alle Tools können sowohl in der Kommandozeile ausgeführt werden, als auch über eine Toolbox für ArcGIS ab Version 9.3 (RAPIDLASSO GMBH 2014). Tools, die zur Generierung von LIDAR Daten in ein Raster dienen, sind:

- las2dem trianguliert LIDAR Punkte vom LAS/LAZ Format oder auch von einigen ASCII Formaten in ein temporäres TIN, welches dann gerastert wird, um ein DEM zu kreieren.
- blast2dem ist fast identisch zu las2dem, es kann nur mit viel mehr Input Daten arbeiten. Las2dem kann bis zu 20 Millionen Punkten verarbeiten, während blast2dem mit bis zu 2 Milliarden Punkten arbeiten kann.

LP360 ist eine Erweiterung für ArcMap zur Visualisierung und Verarbeitung von sehr großen Punktwolken in einer ArcGIS Umgebung. LP360 liefert Werkzeuge zur schnellen Visualisierung von Laserscandaten, automatischen Bodenklassifizierung und Extrahierung von Gebäudepunkten. LP360 gibt es mit drei Lizenzen: Basis, Standard und Erweitert (QCOHERENT, 2015). Zur Generierung von LIDAR Daten stehen die Inverse Distance Weigthing Methode und Triangulierungsmethode zur Verfügung.

3 Digitale Geländemodelle

3.1 Definition DHM/DTM

BROVELLI ET AL. (2004) definiert das digitale Höhenmodell (DHM) als ein kontinuierliches mathematisches Model, das die Form der Oberfläche repräsentiert, z.B. die Höhe als Funktion von geografischen Koordinaten Länge und Breite. BROVELLI unterscheidet dabei zwei Arten von DHMs:

- das Digitale Oberflächenmodell DOM (englisch: Digital Surface Model DSM), welches die Erdoberfläche inklusive aller Objekte beschreibt und
- das Digitale Geländemodell (DGM/ DTM), welches die natürliche Bodenoberfläche repräsentiert, z.B. die kahle Erdoberfläche.

Ein DGM beschreibt also, so wie ein DHM, die Höhen der Erdoberfläche. Der Begriff DGM ist spezifischer als DHM, weil explizit festgelegt wird, welche Höhen, nämlich die Geländehöhen, modelliert werden. Teilweise wird der Unterschied zum DHM auch darin gemacht, dass ein DGM im Unterschied zum DHM oft Geländekanten und Strukturlinien enthält (PFEIFER, 2003).

3.2 Grundsätzliches zur räumlichen Interpolation

Das Problem bei der räumlichen Interpolation wird wie folgt beschrieben: Für gegebene räumliche Daten in Form von diskreten Punkten muss eine Funktion gefunden werden, die am besten die gesamte Oberfläche repräsentiert und die Werte für andere Punkte oder Teilgebiete voraussagt (LAM, 1983, S. 129).

Interpolationsmethoden werden verwendet, um Punkte in eine Flächenrepräsentation zu konvertieren. Der Interpolationsprozess beinhaltet die Schätzung des Wertes der modellierten Variablen als eine Abfolge von Punktlagen, gewöhnlicherweise auf einem quadratischen Gitter. Dieser Prozess wird als „gridding“ bezeichnet. Die „gridded“ Werte werden dann als Pixel von einem Rasterbild behandelt. Alternativen sind zum Beispiel, dass die Daten als Konturlinien im Vektorformat vorliegen. Neben dem „gridding“ können auch viele GIS Punktdaten als Triangulated Irregular Network (TIN) repräsentiert werden (BONHAM-CARTER, 1994, S. 148 bis 150). Aus Sicht der digitalen Datenspeicherung sind Raster jedoch effizienter als TINs oder Konturlinien (RAMIREZ, 2006). Außerdem sind Visualisierung, Analyse und Modellierung im GIS gewöhnlich in einem Raster (MITAS & MITASOVA, 1999).

In der digitalen Geländemodellierung wird die Interpolation verwendet, um den Höhenwert von einem Punkt durch die Verwendung von bekannten Höhen von Nachbarpunkten zu bestimmen. Es gibt zwei indirekte Annahmen hinter den Interpolationstechniken (LI ET AL., 2005, S. 115):

- die Geländeoberfläche ist kontinuierlich und glatt und
- es gibt eine starke Korrelation zwischen den Nachbardatenpunkten.

Interpolation ist eine der Kerntechniken in der digitalen Geländemodellierung, da es in vielen Phasen der Modellierungsprozesse involviert ist, wie zum Beispiel in der Qualitätskontrolle, Oberflächenrekonstruktion, Genauigkeitsbeurteilung, Geländeanalyse und -anwendungen (LI ET AL., 2005, S. 115).

Räumliche Interpolationsmethoden können generell in folgende Kategorien klassifiziert werden (ALI, 2004):

- lokale oder globale Methoden
- exakte oder approximierte Methoden
- deterministische oder stochastische Methoden
- stetige oder unstetige Methoden
- flächenbasierte oder punktbasierte Methoden

lokal und global

Lokale Interpolationsmethoden arbeiten nur mit einer Teilmenge aller Datenpunkte bei der Interpolation, während globale Methoden alle vorhandenen Datenpunkte zur Interpolation verwenden. Globale Interpolationsmethoden sind zum Beispiel die Trend Surface Analysis oder das Regressions Model. Lokale Interpolationsmethoden sind Inverse Distance Weighting (IDW), thine plate-splines oder Kriging (WANG, 2010, S. 42/43).

exakt und approximiert

Exakte Interpolationsmethoden generieren Geländeoberflächen, die durch alle Datenpunkte gehen, somit wird der ursprüngliche Wert des Datenpunktes exakt wiedergegeben. Zu den exakten Methoden gehört zum Beispiel Kriging und einige Spline Methoden.

Approximierte Methoden generieren Geländeoberflächen, die nur einem globalen Trend der Datenpunkte folgen. Sie nähern sich den Daten nur an und geben nur eine geglättete Oberfläche wieder, somit gibt es einen gewissen Grad von Ungewissheit (ALI, 2004).

deterministisch und stochastisch

Stochastische Interpolationsmethoden integrieren das Konzept der Geostatistik, um Geländeoberflächen mit spezifischem Level von Ungewissheit zu generieren. Beispiele für stochastische Methoden sind die Fourier Analyse und Kriging (ALI, 2004).

stetig und unstetig

Bei einer unstetigen Oberfläche erscheint diese in Stufen, wohingegen eine stetige Oberfläche kontinuierlich ist (LI ET AL., 2005, S. 115/116).

flächenbasiert und punktbasiert

LAM (1983, S. 129) klassifiziert die räumlichen Interpolationsmethoden folgendermaßen:
Punkt-Methoden, die sich mit Daten befassen, die sammelbar an einem Punkt sind, wie Temperaturwerte oder Höhen. Zu dieser Methoden gehört die weitere Unterteilung in „exakt" und „approximiert". Flächige-Methoden befassen sich mit Daten, wie z.B. Bevölkerung nach Zählbezirke, die für ein gesamtes Gebiet addiert werden.

In der Vergangenheit wurden viele Punktinterpolationsmethoden entwickelt. Aber keine Methode ist besser für alle Anwendungen geeignet als die anderen. Die Wahl der verwendeten Interpolationsmethode ist abhängig von der Datenart, vom Grad der gewünschten Genauigkeit und vom Rechenaufwand, der erforderlich ist, da einige Methoden sehr zeitintensiv und teuer sind, sodass sie für gewisse Anwendungen verwendet werden können. Alle Methoden haben das Problem, dass jede Methode eine Art von Hypothese über die Oberfläche darstellt und die Hypothesen können oder können auch nicht wahr sein (LAM, 1983).
Typische Interpolationsmethoden in GIS, die entwickelt wurden, um unregelmäßige Punktdaten zu einem Raster transformieren, sind (MITAS & MITASOVA, 1999).:

- Lokale Nachbarschafts Anwendungen:
    - Inverse distance weighted IDW
    - Natural neighbour interpolation
    - Interpolation based on a triangulated irregular network (TIN)
- Geostatistische Methoden
    - Kriging
- Variationsmethoden
    - Spline

Eine Übersicht weiterer Interpolationsmethoden nach BURROUGH ET AL. 1998 ist in der Tabelle 1 dargestellt.

Tabelle 1: Übersicht über Interpolationsmethoden und ihre Eigenschaften (nach BURROUGH ET AL. 1998)

Methode	**Deterministisch/stochastisch**	**Lokal/Global**	**Exakter Interpolator**
Trendoberfläche	Grundsätzlich Deterministisch (empirisch)	Global	Nein
Regressionsmodel	Grundsätzlich Deterministisch (empirisch-statistisch)	Global mit lokalen Verfeinerungen	Nein
Thiessenpolygone	Deterministisch	Lokal	Ja
Pycnophylactic Interpolation	Deterministisch	Lokal	Nein
Lineare Interpolation	Deterministisch	Lokal	Ja
Moving Averages und Inverse Distance Weighting	Deterministisch	Lokal	Nicht bei einem regelmäßig geglätteten Fenster, kann aber erzwungen werden
Thin Plate Splines	Deterministisch mit lokalen stochastischen Komponenten	Lokal	Ja innerhalb geglätteter Grenzen
Kriging	Stochastisch	Lokal mit globalen oder lokalen Variongrammen oder mit globalen Trends	Ja
Conditional Simulation	Stochastisch	Lokal mit globalen oder lokalen Variongrammen oder mit globalen Trends	Nein

3.3 Einflüsse auf die DEM Genauigkeit

GUO ET AL. (2010) beschreibt vier Faktoren, die die Genauigkeit eines DEMs beeinflussen können:

- Topographische Variabilität
- Dichte der LIDAR Punkte
- Wahl der Interpolationsmethoden
- Räumliche Auflösung des Ausgaberasters

Für die topographischen Variabilität gilt, je stärker sich das Gelände ändert, wie es besonders zum Beispiel im Gebirge der Fall ist, desto ungenauer wird das interpolierte Raster (GUO ET AL., 2010). Weiters gilt, je größer die Datendichte ist, desto weniger hat die Interpolationstechnik auf die Genauigkeit vom DTM Einfluss, da der Raum zwischen den bekannten Punkten sehr klein ist. Je geringer die Datendichte ist, desto mehr ist die Höhenschätzung abhängig von der Interpolationsmethode (CHAPLOT ET AL., 2006).

Airborne LIDAR Untersuchungen haben gewöhnlich sehr dichte Punktaufnahmen. Eine zu hohe Dichte kann jedoch auch zu überflüssigen Daten und unnötiger Bearbeitungszeit führen. Die Bodensicht kann aber bei hoher Blattflächenüberdachung sehr reduziert sein, dadurch beinhaltet der Datensatz überwiegend Vegetationsrückgaben und sehr wenig Geländeinformationen. Das hat Auswirkungen auf die Qualität des abgeleitenden DEMs und der Repräsentierung der Geländemorphologie (BATER & COOPS, 2009). REUTEBUCH ET AL. (2003) untersuchte in einer Studie die Genauigkeit eines DTM aus LIDAR Daten unter einer Nadelbaumkrone. Mit traditionellen Bodenuntersuchungsmethoden wurden Koordinaten an mehreren Kontrollpunkten unter der Nadelbaumdecke gesammelt. Diese Punkte wurden zur Validierung der DTM Genauigkeit verwendet. Die LIDAR-Fehlerwerte in dieser Studie zeigen einen geringen Anstieg je dichter das Baumkronendach wird, aber die Unterschiede sind auffallend gering. Generell sind die LIDAR Daten in dieser Untersuchung sehr genau und auch sehr nützlich für das Waldgebiet. Das LIDAR System liefert mit ca. 1 Rückgabe/m^2 für diese Studie immer noch genug Reflexionen vom Boden.

Weitere Studien zur Datenreduktion von LIDAR Daten zeigen, dass die LIDAR Daten bis auf 70% der originalen Datenmenge reduziert werden können, ohne eine signifikante Verschlechterung der DTM Genauigkeit zu erhalten (GUO ET AL., 2010). Wie verschiedene Interpolationsmethoden sich auf die Genauigkeit der DTMs auswirken können wird in Kapitel 3.4 genauer beschrieben, in dem einige Studien präsentiert werden, die sich speziell mit der DTM Generierung aus LIDAR Daten beschäftigen. Weiters entscheidet die räumliche Auflösung des Rasters über die Genauigkeit. Je kleiner die Rasterzellen sind, desto genauer kann das Gelände wiedergegeben werden; je geringer die räumliche Auflösung, desto mehr Geländefläche muss von einer Rasterzelle repräsentiert werden. Dies wurde bereits in zahlreichen Studien untersucht (GUO ET AL., 2010; BATER & COOPS, 2009; GARNERO & GODONE, 2013). GUO ET AL. (2010) fand dabei heraus, dass je nach räumlicher Auflösung sich die topographische Variabilität und die Datendichte unterschiedlich auf die DTM Genauigkeit

auswirken. Bei einer sehr hohen Auflösung (0,5 m und 1 m) haben die topographische Variabilität und die Datendichte gleichermaßen zum RMSE beigetragen, während bei geringen Auflösungen (5 m und 10 m) die topograpische Variabilität einen größeren Effekt als die Datendichte hat.

3.4 Vergleiche von Interpolationsmethoden für die DTM Generierung aus LIDAR LIDAR Daten

Es gibt zahlreiche Studien, die sich mit der Wahl der Interpolationsmethode aus LIDAR Daten beschäftigen. Im Folgenden werden einige dieser Studien vorgestellt. LIU ET AL. (2009) haben die Leistung von folgenden Methoden zur Interpolation eines DEMs von LIDAR Daten getestet:

- Inverse Distance Weighted (IDW)
- Kriging
- Local Polynomial

Die LIDAR Daten aus dieser Studie hatten einen mittleren Abstand von 2,2 m. Die Ergebnisse dieser Studie zeigen die Leistung von jedem Interpolationsalgorithmus für zwei Testgebiete mit unterschiedlichen Geländetypen. Das erste Testgebiet im Untersuchungsraum ist relativ eben mit mehreren flachen Rinnen. Das zweite Gebiet ist ein raues Gelände, welches durch vulkanisch bedingte steinige Erhebungen dominiert wird.

Diese Studie verdeutlicht, dass die drei Interpolationsalgorithmen im flachen Testgebiet deutlich kleinere RMS Fehler (RMSE) aufweisen als im komplexeren Gelände. Bei flachem Gelände wurden die Interpolatoren gut ausgeführt, jedoch kam es beim komplexen Gelände zu mehr Fehlern während des Interpolationsprozesses, auch wenn es eine sehr hohe Datendichte gab.

Die größten Fehler lieferte bei dieser Studie das Kriging, welches für LIDAR Daten weder auf dem flachen, noch auf dem komplexen Gelände gut gearbeitet hat. Weiters benötigt Kriging viel Rechenzeit, besonders bei großen Datenmengen, wie LIDAR Daten.

Die Local Polynomial Interpolationsmethode lieferte im Vergleich zum IDW und zum Kriging die besten Ergebnisse im flachen Testgebiet. Die IDW Methode hat in dieser Studie ganz gute Ergebnisse auf dem flachen und auf dem komplexen Testgebiet gezeigt, besonders bei hoher Datendichte. Im komplexen Gelände gab es kaum signifikante Unterschiede zum Local Polynomial. IDW Methoden arbeiten also gut, wenn die Datendichte gut ist, sogar bei komplexen Gelände.

Für die Autoren ist die IDW Methode daher die passendste Interpolationsmethode, sie stellt einen Kompromiss zwischen Genauigkeit und Rechenzeit für große LIDAR Datenmengen dar, da sie einfach zu handhaben ist, eine schnelle Rechenzeit aufweist und auch in den meisten GIS Anwendungen verfügbar ist.

BATER & COOPS (2009) untersuchten sieben Methoden, um die beste Kombination zwischen Interpolationsalgorithmus und räumlicher Auflösung für die DEM Generierung zu identifizieren. Zu diesen untersuchten Methoden gehören:

- linear
- quintic
- Natural Neighbour
- Spline with tension
- regulaized spline
- IDW
- ANUDEM

Das Untersuchungsgebiet zu dieser Studie befindet sich in Clayoquot Sound auf Vancover Island, British Columbia. Die Topographie ist vor allem durch pleistozäne Gletscherablagerungen geprägt. Das Untersuchungsgebiet besteht sowohl aus Kräutern, Büschen, Jungbäumen, jungem Wald und altem Wald. Der mittlere Fußabdruckdurchmesser der LIDAR Daten beträgt 0,4 m. Die Interpolationsmethoden wurden dabei für 0,5 m, 1 m und 1,5 m Auflösung angewendet. In dieser Forschungsarbeit ergibt sich, dass alle untersuchten Methoden sehr ähnlich in Hinblick auf globalen RMS und den mittleren absoluten Fehler waren. Dabei sind die lineare und die Natural Neighbour Methode die konservativsten Algorithmen und hatten auch insgesamt die niedrigsten Fehlerbandbreiten. Weiters ist aber auch die Genauigkeit für Splining und IDW stark beeinflusst durch ihr Parametrisierung. Außerdem hat auch die Auswahl der räumlichen Auflösung (die Zellgröße) für die DEM Generierung eine ähnlich große Bedeutung auf die Qualität des DEMs.

Die Autoren bevorzugen dabei die Natural Neighbour Interpolation für ihr Untersuchungsgebiet hinsichtlich seiner Leistung, da dieser Algorithmus leicht zu gebrauchen ist, eine einfache Parametrisierung aufweist, generell glatte und visuell ansprechende Oberflächen generiert und eine gleichmäßige Genauigkeit hat. ANUDEM, quintic oder spline sind jedoch für die Autoren gute Alternativen.

IDW arbeitete für die Autoren verglichen zu den anderen Methoden nicht gut genug und auch die Oberfläche war ihrer Meinung nach die am wenigsten realistische Repräsentation des

Geländes. Weiterhin waren vor allem IDW und Spline in dieser Studie anfällig für Veränderungen im Gelände. Die meisten Interpolationsfehler tauchten in dieser Studie bei starken Neigungen und bei geringer Punktdichte auf.

GARNERO & GODONE (2013) untersuchten folgende Interpolationsmethoden zur Generierung von DTM:

- Inverse distance weighing IDW
- Spline
- Natural Neighbours

Diese Algotithmen wurden in dieser Studie auf zwei Datensätze angewendet. Der erste Datensatz bezieht sich auf ein Gebirgsgebiet in den westlichen Alpen (Bardonecchia). Der zweite Datensatz repräsentiert ein flaches und urbanisiertes Gebiet (Grugliasco). Die LIDAR Daten haben eine mittlere Datendichte von 0,22 Punkten/m^2 und einen mittleren Punktabstand von 2,12 m. Alle Interpolationsmethoden wurden für eine 5x5, 10x10 und 20x20 m Zellengröße angewendet.
Spline zeigte schon Schwierigkeiten bei der Interpolation dichter Datensätze und konnte daraus kein DTM generieren, während IDW und Natural Neighbours keine Probleme hatten. Der IDW Algorithmus zeigte bei einer 100% Datendichte bei allen Auflösungen (5 m, 10 m und 20 m Zellengröße) die besten Ergebnisse für diese Studiengebiete.

Ziel der Studie von GUO ET AL. (2010) war es, den Effekt von topographischer Variabilität, LIDAR Datendichte unter verschiedenen Interpolationsmethoden auf die DEM Genauigkeit zu quantifizieren. Die untersuchten Interpolationsmethoden waren:

- Natural Neighbor (NN)
- Inverse Distance Weighted (IDW)
- Triangulated Irregular Network (TIN)
- Spline
- Ordinary Kriging (OK)
- Universal Kriging (UK)

Das Untersuchungsgebiet in dieser Studie umfasst ein Gebiet nordöstlich von Oakhurst in Kalifornien. Die mittlere Höhe des Untersuchungsgebietes liegt bei 1631 m. Die minimale Höhe beträgt 758 m und die maximale Höhe 2652 m. Die Dichte der LIDAR Daten beträgt 1,32 Punkte pro m^2.

Diese Studie ergab, dass OK und UK die genausten DEMs produziert haben mit den kleinsten RMS Fehlern und Standardabweichungen. Jedoch ist ihr Nachteil, dass sie sehr rechenintensiv und damit auch sehr zeitintensiv sind.

Der IDW Algorithmus produziert die größten RMS Fehler und Standardabweichungen bei einer Auflösung von 0,5 und 1 m im Vergleich zu den anderen Algorithmen. Bei einer Auflösung von 5 m und 10 m ist die erreichte DEM Genauigkeit fast so gut wie für OK und UK.

Spline, TIN und NN liefern relativ kleine RMS Fehler und Standardabweichungen bei einer hohen Auflösung, aber die RMS Fehler und die Standardabweichungen werden bei niedrigen Auflösungen deutlich höher. Bei Auflösungen von 5 m und 10 m lieferte Spline weitaus die schlechtesten Ergebnisse und ist daher ziemlich unzuverlässig bei niedrigen Auflösungen.

Auch diese Studie verdeutlicht, dass alle Methoden sehr empfindlich gegenüber Datenreduktionen sind.

Zusammenfassend ergibt sich aus dieser Studie, dass IDW, NN und TIN eine hohe Effizienz bei der Generierung von DEMs aus LIDAR Daten aufweisen, da sie sehr einfach sind, schnelle Rechenzeit besitzen und auch relativ genaue DEMs generieren können. OK und UK sind aber verlässlicher in Hinblick auf die Genauigkeit und generieren DEMs mit der höchsten Qualität, sie sind jedoch dafür auch rechenintensiver.

ALI (2004) verwendete in seiner Studie für die Interpolation von LIDAR Daten zu einer Geländeoberfläche folgende Methoden:

- Inverse distance interpolation
- Kriging
- TIN

Das Untersuchungsgebiet in dieser Studie liegt an der Erieseeküste. Das Untersuchungsgebiet ist relativ flach. Die Dichte der LIDAR Daten beträgt 30 Punkte pro 100 m^2.

Im Vergleich zwischen IDW und Kriging scheint für diese Studie das IDW besser zu arbeiten als das Kriging, weil die LIDAR Daten sehr dicht sind, was am besten zur IDW Methode passt. TIN lieferte jedoch die beste Wahl für die Interpolation von LIDAR Daten für dieses Gebiet, jedoch kann es sein, dass es nicht die besten Profile produziert.

Für den Autor ist IDW trotzdem eine gute Wahl. Jedoch zeigt IDW an einzelnen Stellen keine gute Genauigkeit bei den Höhenwerten. ALI (2004) empfiehlt jedoch, nicht Kriging auf LIDAR Daten anzuwenden, basierend auf seinen Ergebnissen, die er in diesen Experiment erhalten hat.

CHAPLOT ET AL. (2006) untersuchte die DEM Generierung für verschiedene Geländeformen, Datendichten, räumliche Skalen und Interpolationsmethoden. Drei Untersuchungsbiete liegen in den Bergregionen von Nordlaos und drei in der sanften Landschaft von Westfrankreich, welche auch weiterhin auf drei unterschiedlichen Skalen getestet wurden. Nordlaos ist gekennzeichnet durch steile Bergneigungen mit den höchsten Spitzen bei 2500 m. Die Algorithmen für diese Studie sind:

- Inverse distance weighting IDW
- OK
- UK
- multiquadratic radial basis function (MRBF)
- regularized Spline with tension (RST)

Dabei beobachteten die Autoren, dass, je größer die Datendichte ist, desto weniger Einfluss die Interpolationstechnik auf die Genauigkeit vom DTM hat, da die Entfernung zwischen den bekannten Punkten sehr klein ist. Je geringer die Datendichte ist, desto mehr ist die Höhenschätzung abhängig von der Interpolationsmethode. Ist die Dichte der Probenpunkte hoch, so gibt es nur wenige Unterschiede zwischen den Interpolationstechniken. MRBF hat nur ein wenig besser gearbeitet als die anderen Interpolationstechniken.
Bei geringerer Datendichte erzielte RST die besten Ergebnisse bei Landschaften mit geringen Variationskoeffizienten der Höhe und schwacher räumlicher Struktur. Bei einem hohen Variationskoeffizienten der Höhe und starker räumlicher Struktur hat IDW gering besser gearbeitet als die anderen Methoden. Kriging erzielte hingegen die besten Ergebnisse für Landschaften mit starker räumlicher Struktur aber geringen Variationskoeffizienten der Höhe.

LLOYD & ATKINSON (2006) haben in ihrer Studie folgende Interpolationsmethoden verglichen:

- Inverse Distance Weighting (IDW)
- Ordinaty Kriging (OK)
- Kringing with Trend Model (KT)

Es wurden drei Untersuchungsgebiete in England gewählt (Humberside, Shropshire, Lincolnshire). Diese drei Untersuchungsgebiete weisen keine großen Höhenunterschiede auf. Die Autoren beurteilten für diese Untersuchungsgebiete, dass die OK Methode größere Vorhersagegenauigkeit als die IDW Methode bietet, während KT noch größere Vorteile gegenüber OK zeigte. Die absoluten Unterschiede sind zwar nicht groß, aber um das Beste von der hohen Qualität der LIDAR Daten auszumachen, bietet KT die geeignetste Technik in diesem Fall.

Kriging erzielte bessere Schätzungen der Höhe als IDW, besonders wenn die Bodenpunktdichte sehr gering wird. Kriging war eine genauere Methode, wenn die Datendichte sehr gering war, aber LLOYD & ATKINSON beurteilten, dass es keine Vorteile gab, wenn man die anspruchsvollere geostatischtische Methode verwendet hat, wenn eine große Datenmenge zur Verfügung steht.

ABRAMOV & MCEWEN (2004) wählten vier Interpolationsmethoden aus, um die Mars Orbiter Laser Altimeter (MOLA) Daten zu testen:

- Delaunay-based linear interpolation,
- splining,
- Nearest Neighbour,
- Natural Neighbour

Diese Methoden wurden für MOLA Daten vom Korolev Krater (Mondkrater) für eine qualitative Analyse (visueller Vergleich auf visulle Artefakte) und für Daten eines Teilgebietes von Island für eine quantitative Analyse (Vergleich der interpolierten DEMs mit einem DEM vom celandic Geodetic Survey) verwendet.
Der Natural Neighbour Algorithmus erzielte dabei exzellente Ergebnisse bei einer Anwendung auf MOLA Daten. Die Natural Neighbour Methode lieferte quantitativ und qualitativ die besten Ergebnisse und auch die wenigste Anzahl an sichtbaren Interpolationsartefakten, es benötigte allerdings auch die meiste Rechenzeit.

SU & BORK (2006) verglichen folgende Interpolationsmethoden:

- Inverse Distance Weighting IDW
- Kriging
- Splining

Die Studie fand im Untersuchungsgebiet Aspen Parkland statt, einer Hügellandschaft mit Geländeerhebungen von fünf bis zehn Meter. Vegetationstypen sind Auenwiesen, hochgelegene Grasflächen und Wald.
Ergebnis dieser Studie war es, dass IDW im Vergleich zu den beiden anderen Methoden eine Oberfläche mit insgesamt weniger Fehlern erstellt hat. Sogar bei Daten mit einer Neigung größer als 15° zeigte IDW die geringsten Fehler. IDW war eine einfachere und genauere Interpolationsmethode als Kriging für die DEM Generierung. Die IDW Methode liefert also in dieser Studie bei einer hohen Dichte an LIDAR Punkten, diese liegt in diesem Untersuchungsgebiet bei >0,75 Punkte pro m^2, bessere Ergebnisse als Kriging.

Aus diesen Studien wird deutlich, dass keine dieser Interpolationsmethoden universal für alle Arten von Datenquellen, Geländemuster oder Zielen ist (LIU ET AL., 2009; LIU, 2008). Alle Studien zeigen auch ganz unterschiedliche Meinungen der Autoren, welche Interpolationsmethoden am besten für LIDAR Daten geeignet sind. Da auch nicht immer exakt die selben Methoden miteinander verglichen werden und auch die Untersuchungsgebiete verschiedene Charakteristika aufweisen, sowie auch die Punktdichte der LIDAR Daten, kann keine Methode als die beste Methode identifiziert werden. Dennoch lässt sich festhalten, dass IDW bei sehr dichten Daten, wie es einmal LIDAR Daten sind, oft gute Ergebnisse geliefert hat. Da diese Methode auch eine sehr einfache Methode ist, die nicht sehr viel Rechenzeit benötigt und daher sehr effizient ist, wurde diese im praktischen Teil der Arbeit weiter berücksichtigt.

4 Datengrundlage

Das Land Steiermark hat in einem mehrjährigen Projekt hochgenaue 3D-Laserscandaten für die Steiermark erfasst, die seit 2013 auch flächendeckend vorhanden sind.
Die ALS Punktwolke umfasst mindestens eine mittlere Punktdichte:

- für Bereiche unter 2000 m Seehöhe von 4 Punkten/m^2
- für Bereiche über 2000 m Seehöhe von 2 Punkten/m^2

Die mittlere Punktdichte bezieht sich dabei auf den Bereich des Flugstreifens ohne Überlappung. Die Anforderung der Mindestgenauigkeit für die LIDAR Daten liegen im Nicht-Überlappungsbereich für:

- die absolute Lagegenauigkeit der Laserpunkte bei σ +-40 cm
- die absolute Höhengenauigkeit auf ebener, befestigter Fläche bei σ +- 15 cm

Im Überlappungsbereich liegt die Genauigkeit für:

- die Lagegenauigkeit bei +-20 cm
- die Höhengenauigkeit bei +-10 cm

5 ASPRS LAS Format

Die American Society for Photogrammetry and Remote Sensing (ASPRS) hat ein einheitliches Standarddateiformat, das LAS-Dateiformat, für LIDAR Daten erstellt und beschreibt dieses Format folgendermaßen (AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING, 2012):
Bei dem LAS-Dateiformat handelt es sich um ein öffentliches Format, das den Austausch von dreidimensionalen LIDAR-Punktwolkendaten zwischen Datennutzern vereinfachen soll. Dieses binäre Dateiformat soll vor allem eine Alternative sein, da von vielen Unternehmen häufig entweder firmeneigene Systeme oder auch generische ASCII-Dateiaustauschsysteme eingesetzt werden. Das Problem bei firmeneigenen Systemen ist dabei, dass die Daten nicht einfach von einem System zum anderen ausgetauscht werden können. Auch ASCII Dateiformate für LIDAR-Daten bringen zwei große Probleme mit sich. Das erste Problem ist die Leistung, da das Lesen und die Interpretation der ASCII Höhendaten sehr langsam sein kann. Das zweite Problem ist, dass alle spezifischen Informationen zu den LIDAR-Daten beim ASCII Format verloren gehen. Das LAS-Dateiformat ist hingegen ein binäres Dateiformat, das spezifische Angaben über den LIDAR Charakter der Daten beinhaltet, aber nicht übermäßig komplex ist. Bereits seit September 2011 gibt es die derzeit aktuelle LAS 1.4 Spezifikation (Stand 2014).
Das Land Steiermark verwendet jedoch noch die LAS Format Version 1.1, daher wird im Folgenden etwas darauf eingegangen. Das LAS Format besteht aus (AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING, 2005):

- Header Block (beinhaltet allgemeine Informationen wie Punktnummern oder die Koordinaten der Grenzen)
- Variablen Length Record (beinhaltet Informationen über die Projektion, Metadaten und allgemeine User-Informationen)
- Point Data Record (beinhaltet die aufgenommen LIDAR Punkte)

5.1 Public Header Block

Der Inhalt des Publich Header Blocks des LAS Formates wird in Tabelle 2 veranschaulicht und genauer beschrieben.

Tabelle 2: Aufbau des Public Header Block im LAS Format (Selbst erstellt nach AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING, 2005)

Item	**Erklärung**
File Signature (“LASF”)	Muss „LASF“ beinhalten und bestimmt die LAS Spezifikation.
(1.1) File Source ID	Beinhaltet die Flugstreifennummer, wenn diese Datei von einem originalen Flugstreifen abgeleitet worden ist.
(1.1) Reserved	Dieses Datenfeld ist reserviert und muss mit einer Null von der generierenden Software gefüllt werden.
(1.1) Project ID - GUID data 1	Diese vier Felder sind optional und reserviert für eine Projekt ID, um die LAS Datei einem Projekt zuzuordnen.
(1.1) Project ID - GUID data 2	
(1.1) Project ID - GUID data 3	
(1.1) Project ID - GUID data 4	
Version Major	Dieses Feld beinhaltet die Versionsnummern. Für Version 1.4 steht zum Beispiel in Version Major die führende Versionsnummer, wie 1, und in Version Minor steht die untere Versionsnummer, wie 4.
Version Minor	
(1.1) System Identifier	Dieses Feld gibt an, ob die Datei das Ergebnis von Extraktion, Zusammenführung oder Modifizierung von Daten ist.
Generating Software	Dieses Feld beschreibt die generierende Software (Name der Software und Version).
(1.1) File Creation Day of Year	Beschreibt den Tag an dem die Datei erstellt wurde.
(1.1) File Creation Year	Beschreibt das Jahr in dem die Datei erstellt wurde.
Header Size unsigned	Beschreibt die Größe des Header Blocks in Bytes.
Offset to point data	Beschreibt die tatsächliche Anzahl an Bytes vom Dateianfang bis zum ersten Point Record Datenfeld.
Number of variable length records	Dieses Feld beinhaltet die aktuelle Anzahl der Varaible Length Records.
Point Data Format ID (0-99 for spec)	Beinhaltet die Punktdatenformat ID entsprechend zum Punktdatenaufnahmeformattyp.
Point Data Record Length	Keine Beschreibung vorhanden.
Number of point records	Dieses Feld beinhaltet die gesamte Anzahl von Punktaufnahmen innerhalb der Datei.
Number of points by return	Dieses Feld beinhaltet ein Array der gesamten Punktaufnahme pro Rückgabe. Der erste Wert entspricht der gesamten Anzahl an

	Aufnahmen der ersten Rückgabe und der zweite Wert bezeichnet die gesamte Anzahl an Aufnahmen der zweiten Rückgabe. Weitere Werte können bis zur fünften Rückgabe folgen.
X scale factor	Diese Felder beinhalten Werte, die die dazugehörigen X, Y und Z Werte innerhalb der Point Data Record skalieren können. Die jeweiligen X, Y und Z Werte müssen mit dem jeweiligen Skalierungsfaktor multipliziert werden. Wenn zum Beispiel die X, Y und Z Koordinaten zwei Dezimalstellen haben, so beinhaltet jeder Skalierungsfaktor die Zahl 0,01.
Y scale factor	
Z scale factor	
X offset	Die Offset Felder werden benötigt, um den Offset der Punktaufnahmen zu setzen.
Y offset	
Z offset	
Max X	Beinhaltet die maximale und minimale Koordinatenausdehnung der LAS Datei.
Min X	
Max Y	
Min Y	
Max Z	
Min Z	

5.2 Variable Length Record

Der Public Header Block wird von einem oder mehreren Variable Length Records gefolgt. Der Aufbau jedes Variable Length Records wird in Tabelle 3 veranschaulicht.

Tabelle 3: Aufbau des Variable Length Records (Selbst erstellt nach AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING, 2005)

Item Format	**Erklärung**
(1.1) Reserved	
User ID	Beinhaltet den Namen des Users, der den Variable Length Record erstellt hat.
Record ID	Dieses Feld ist abhängig vom User ID Feld. Die LAS Spezifikation managen die Record ID selbstständig. Ansonsten werden die Record IDs vom Besitzer der User ID gemanagt.
Record Length After Header	Dieses Feld gibt die Anzahl der Bytes an, die nach dem Standardteil des Headers stehen.
Description	Beinhaltet eine optionale Textbeschreibung der Daten.

5.3 Point Data Record

Das Grundformat bei LAS 1.1 ist das Format 0 (Tabelle 4). Weitere Formate werden von diesem Format abgeleitet und können zusätzliche Datenfelder besitzen. Zum Beispiel wird dem Format 1 das Datenfeld GPS Zeit angehängt.

Tabelle 4: Aufbau des Point Data Record Formats 0 (Selbst erstellt nach AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING, 2005)

Item	**Erklärung**
X	Die X, Y und Z Werte werden als ganzzahlige Datentypen abgespeichert. Die dazugehörigen X, Y und Z Skalierungsfaktoren aus dem Public Header Block verändern diese ganzzahligen Datentypen in ihre wahren Gleitkommazahlwerte.
Y	
Z	
Intensity	Der Intensitätwert repräsentiert die Pulsrückgabemenge.
Return Number	Gibt die Rückgabeimpulsnummer des ausgehenden Impulses an.
Number of Returns (given pulse)	Ist die Gesamtzahl der Returns für einen gegebenen Impuls.
Scan Direction Flag	Gibt an in welcher Richtung der Scannerspiegel während des Laserpulses stand.
Edge of Flight Line	Dieser Wert wird eine „1“ zugewiesen, wenn es sich um den letzten Punkt eines Flugstreifens handelt.
(1.1) Classification	Die Punkte werden einer standardisierten ASPRS Klassifikation zugewiesen, dies wird in Tabelle 5 veranschaulicht.
(1.1) Scan Angle Rank (-90 to +90) – Left side	Dieses Feld gibt den Winkel an, in dem der Laserstrahl ausgesendet wurde, inklusive dem Roll-Winkel des Flugzeuges; er liegt zwischen +90° und -90°. Der Nadir entspricht dem Wert „0“ und die linke Seite des Flugzeuges in Flugrichtung entspricht dem Wert „-90“.
(1.1) User Data	Dieses Feld kann je nach Ermessen des Anwenders beschrieben werden.
(1.1) Point Source ID	Dieser Wert zeigt die Datei, von welcher dieser Punkt abstammt.

Die X, Y und Z Koordinaten ergeben sich aus den X, Y und Z Aufnahmen im Point Data Record, sowie aus den Skalierungs- und Offset Faktoren im Public Header Block. Die X Koordinaten ergeben sich so zum Beispiel aus der Punktaufname von X im Point Data Record multipliziert mit dem Skalierungsfaktor für X und addiert mit den Offset Wert für X aus dem Public Header Block.

$$X_{Koordinate} = (X_{Aufnahme} * X_{Skalierung}) + X_{Offset} \quad [5.1]$$

$$Y_{Koordinate} = (Y_{Aufnahme} * Y_{Skalierung}) + Y_{Offset} \quad [5.2]$$

$$Z_{Koordinate} = (Z_{Aufnahme} * Z_{Skalierung}) + Z_{Offset} \quad [5.3]$$

Tabelle 5: ASPRS Standard für LIDAR Punktklassen (nach AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING, 2005)

Klassenwert	**Bedeutung**
0	Nie klassifiziert
1	Nicht zugewiesen
2	Erde
3	Niedere Vegetation
4	Mittlere Vegetation
5	Hohe Vegetation
6	Gebäude
7	Niedrige Punkte (Rauschen)
8	Modellschlüssel
9	Wasser
10	Reserviert für die ASPRS-Definition
11	Reserviert für die ASPRS-Definition
12	Überlappung
13–31	Reserviert für die ASPRS-Definition

6 Untersuchungsgebiet

Sankt Lorenzen im Paltental liegt in der Gemeinde Trieben im Bezirk Liezen in der Steiermark. Der Ort liegt im Paltental, dem südöstlichen Seitental des oberen Ennstals und gehört somit zu den Niedern Tauern mit ihrer Untergruppe der Rottenmanner Tauern. St. Lorenzen liegt auf ca. 749 m ü. A. am Fuß des Hohenbühel (1275 m ü. A.). Das Gelände ist gekennzeichnet durch steile und bewaldete Hänge und unwegsames Gelände.

Das Gebiet um St. Lorenzen im Paltental ist sehr repräsentativ für verschiedene Geländestrukruren. Deshalb wurden um St. Lorenzen drei charakteristische Untersuchungsgebiete gewählt, die jeweils verschiedene Geländemerkmale aufweisen. Abbildung 4 zeigt dabei die Lage der Untersuchungsgebiete in der Gemeinde Trieben.

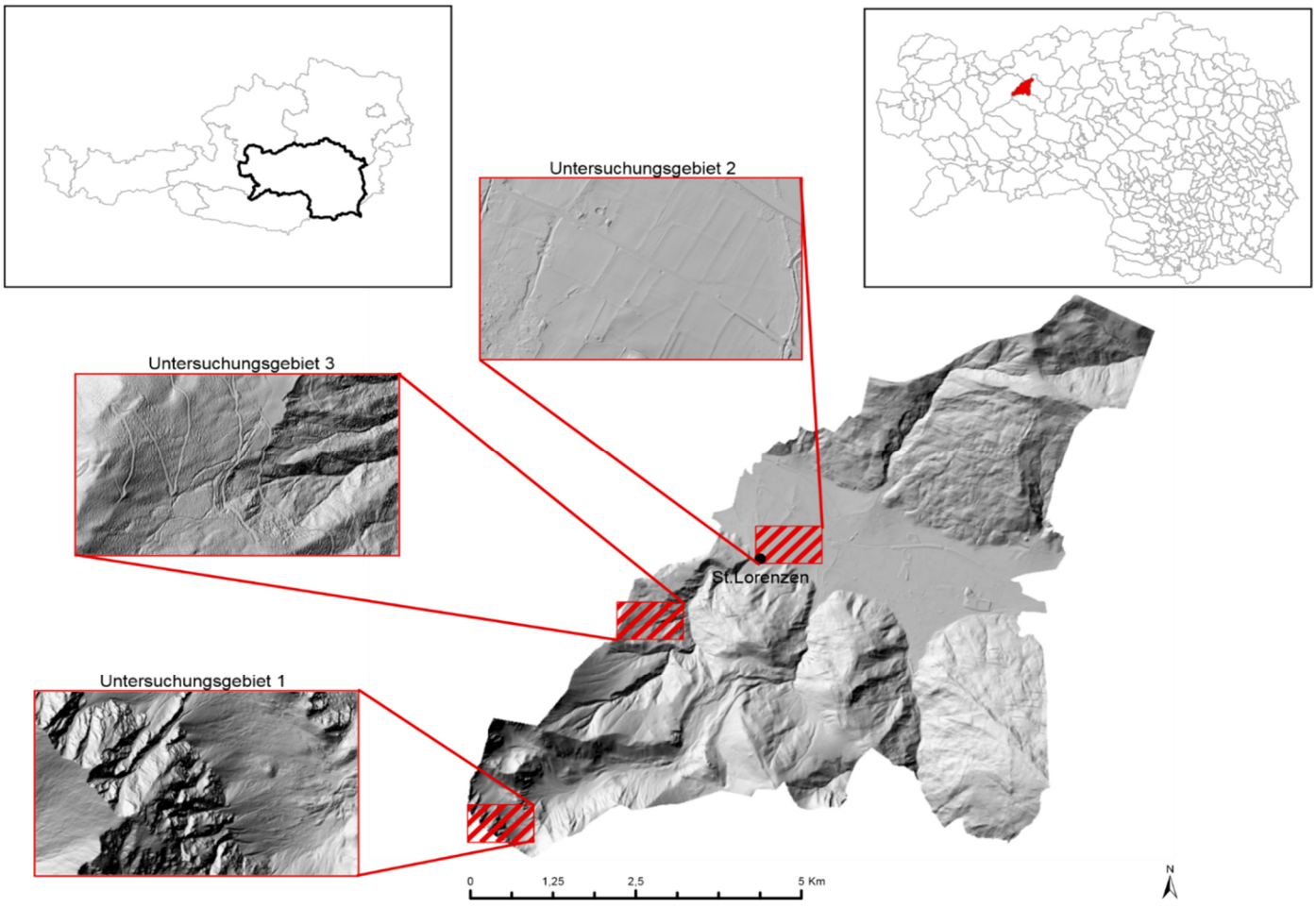

Abbildung 4: Relief der Gemeinde Trieben. Die roten Polygone zeigen die Positionen der Untersuchungsgebiete in der Gemeinde.(Daten: ALS Daten vom Land Steiermark, eigene Darstellung).

Untersuchungsgebiet 1

Untersuchungsgebiet 1 liegt in den Rottenmanner Tauern und ist charakterisiert durch sehr starke Reliefenergie. Der Höhenunterschied erstreckt sich von 1852,12 m bis 2429,44 m. Die Hangneigungen in diesem Gebiet reichen von 0° bis zu 88°. In diesem Untersuchungsgebiet wurden durch die ALS Befliegung insgesamt 3 356 144 Punkte erfasst, davon sind 2 793 490 Bodenpunkte (Tabelle 6). Das Untersuchungsgebiet bedeckt ein 1000 m x 600 m großes Gebiet.

Das Untersuchungsgebiet wurde gewählt als ein Beispiel für eine Hochgebirgsregion in der Steiermark mit einer unwegsamen Geländeoberfläche und Steilhängen. Das Untersuchungsgebiet besteht fast nur aus Gestein (Abbildung 5).

Untersuchungsgebiet 2

Das Untersuchungsgebiet 2 liegt im Paltental und ist daher eine ebene Fläche mit sehr geringen Höhenunterschieden von 741,29 m bis 801,1 m und flachen Hangneigungen im Mittel von 4,62°. In diesem Untersuchungsgebiet wurden durch die ALS Befliegung insgesamt 3 744 218 Punkte erfasst, davon sind 3 212 496 Bodenpunkte (Tabelle 6). Das Untersuchungsgebiet bedeckt ebenfalls eine Fläche von 1000 m x 600 m.

Das Untersuchungsgebiet wurde gewählt als ein Beispiel für ein flaches Talgebiet in der Steiermark. Es ist hauptsächlich durch Siedlungsstrukturen und landwirtschaftlichen Anbau geprägt (Abbildung 6) und zeigt deutlich den Einfluss einer ebenen Fläche auf die Datenaquisitation. So wurden in diesem Untersuchungsgebiet die meisten LIDAR Bodenpunkte aufgenommen.

Untersuchungsgebiet 3

Das Untersuchungsgebiet 3 liegt in einem Waldstück nahe St. Lorenzen und ist gekennzeichnet von hoher Reliefenergie mit Höhenunterschieden von 951,96 m bis zu 1441,97 m und Hangneigungen von 0° bis 79,11°. In diesem Untersuchungsgebiet wurden durch die ALS Befliegung insgesamt 2 802 973 Punkte erfasst, davon sind 668 042 Bodenpunkte (Tabelle 6). Das Untersuchungsgebiet bedeckt ebenfalls ein 1000 m x 600 m großes Gebiet. Es wurde gewählt als ein Beispiel für eine Mittelgebirgsregion in der Steiermark und ist hauptsächlich mit Wald bedeckt (Abbildung 7). Bei diesem Untersuchungsgebiet wird deutlich welchen

Einfluss Waldgebiet auf die Datenaquisitation hat. Es wurden hier deutlich weniger Bodenpunkte klassifiziert als in den anderen Gebieten.

Tabelle 6: Überblick über die Flächen-, Höhen-, Hangneigungs- und ALS Punktparameter in den drei Untersuchungsgebieten (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

	Fläche [km²]	**Höhe [m]**			**Hangneigung [°]**			**ALS Punkte Gesamt**	**ALS Boden-punkte**
		Max	Min	Mittel	Max	Min	Mittel		
UG 1	0,6	2429,44	1852,12	2132,77	88,17	0	35,62	3 356 144	2 793 490
UG 2	0,6	801,1	741,29	762,84	58,36	0	4,62	3 744 218	3 212 496
UG 3	0,6	1441,97	951,96	1196,57	79,11	0	26,44	2 802 973	668 042

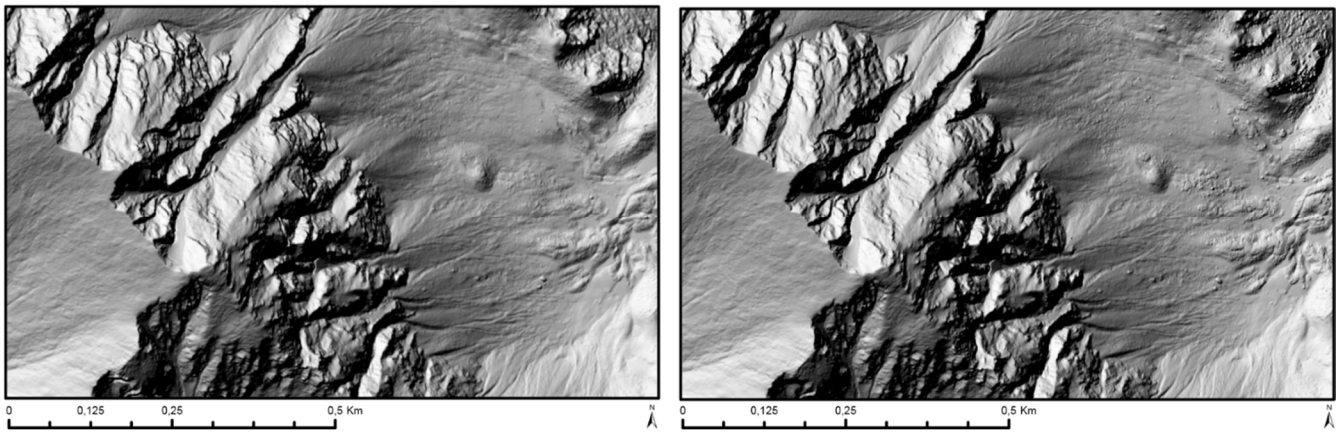

Abbildung 5: Untersuchungsgebiet 1. Links: DTM. Rechts: DSM (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

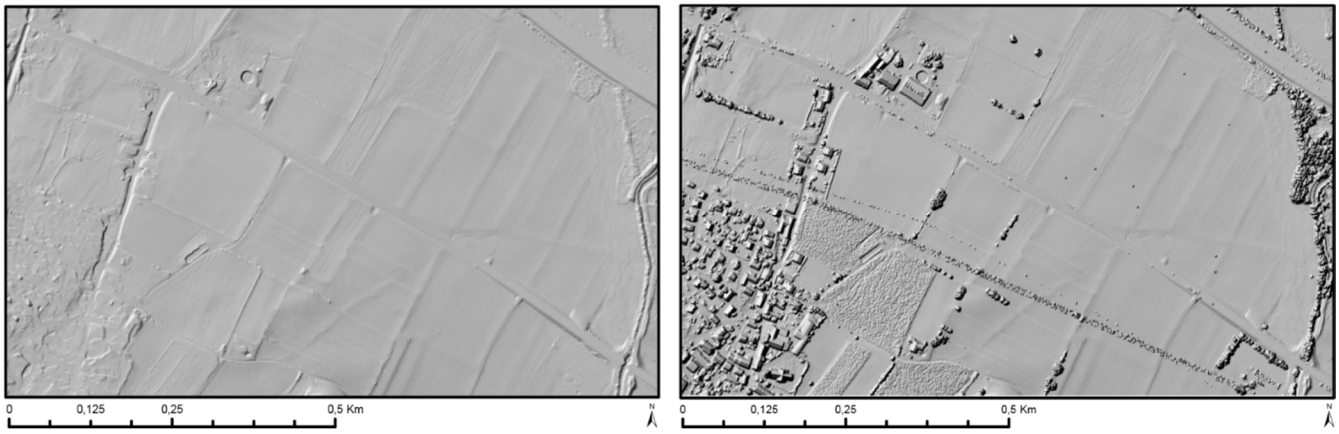

Abbildung 6: Untersuchungsgebiet 2. Links: DTM. Rechts: DSM (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

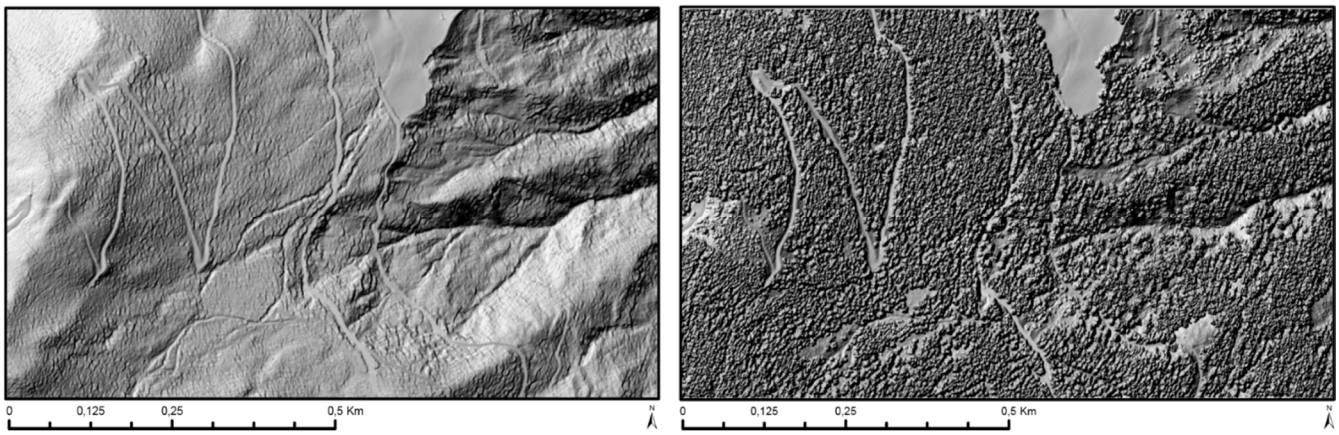

Abbildung 7: Untersuchungsgebiet 3. Links: DTM. Rechts: DSM (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

7 Arbeiten mit LIDAR Daten in Python

LIDAR Daten haben ein spezielles Format, wie bereits in Kapitel 5 beschrieben. Daher können diese nicht mit den allgemeingültigen Pythonbefehl „open“ geöffnet werden. Speziell für das LAS Format gibt es freie Bibliotheken im Internet, die mit LAS-Dateien arbeiten können. Beispiele hierfür sind die quelloffenen Bibliotheken libLAS, LASlib und Laspy.

libLAS

libLAS ist eigentlich eine C/C++ Bibliothek für das Lesen und Schreiben des LAS Formates, welche aber auch mit Python verwendet werden kann. Die anfängliche Entwicklung von libLAS wurde 2007-2008 durch die Iowa Geological Survey Bureau (IGSB) in ihrem landesweiten LIDAR Projekt unterstützt. Die weitere Unterstützung von libLAS wird durch eine Vielzahl von Organisationen, wie zum Beispiel U.S. Army Cold Regions Research und Engineering Laboratory, gewährleistet. Ab 2014 wurde libLAS jedoch fast vollständig von Martin Isenburgs LASlib Bibliothek abgelöst. Die einzige Funktionalität, die libLAS über LASlib bietet, ist die Fähigkeit, mit den von Geospatial Data Abstraction Library (GDAL) abgeleiteten Koordinatensysteminformationen zu arbeiten und in selten verwendete Koordinatensysteme zu projizieren. Weiterhin werden die Python Funktionalitäten von libLAS viel besser durch die pure-Python-Bibliothek Laspy ausgedrückt. Außerdem ist mit Laspy die Manipulation von LAS-Daten in Python möglich (BUTLER ET. AL., 2014).

Laspy

Laspy ist eine Pythonbibliothek für das Lesen, Modifizieren und Erstellen von LAS Formaten. Laspy wurde für die Pythonversion 2.7. erstellt (BROWN & BUTLER, 2012). Laspy ist sehr einfach anzuwenden und auch sehr schnell. Laspy gibt es bereits in der Version 1.2.5 (Stand 2015) und besitzt eine anwenderfreundliche Dokumentation auf http://pythonhosted.org/laspy/. Da Laspy eine reine Python Bibliothek ist und auch im Gegensatz zu libLAS die LAS-Daten modifiziert werden können, wurde in dieser Masterarbeit mit Laspy gearbeitet.

7.1 Einlesen der LAS Daten mit Laspy

Laspy bietet eine schnelle und umkomplizierte Funktion, um LIDAR Daten in LAS Format einzulesen (vgl. BROWN & BUTLER, 2012):

```
from laspy.file import File

#Öffne .LAS Datei
inFile = File("D:/Beispiel_lasfile.las", mode = "r")
```

Die Daten, die in der LAS Datei gelesen werden können, sind:

- Header Informationen
- Punktdaten
- oder Inhalte der Variablen Length Record (VLR)

Beim Einlesen der Punkte werden diese in einem Array verspeichert. Dies ist ein weiterer deutlicher Unterschied zu libLAS, wo die Punktdaten mittels Listen-Abstraktion (List Comprehension) eingelesen werden. Listen-Abstraktion ist eine weitere Methode, um Mengen in Python zu definieren.

Arrays (im Deutschen: Feld) sind ebenfalls Datenstrukturen zur Speicherung und Organisation von vielen Daten gleichen Types. Arrays sind in Python bedeutend effizienter und auch schneller als Listen. Bei einem Array müssen alle Elemente im Gegensatz zu Listen den gleichen Typ haben. Arrays sind in Python Bestandteil des OpenSource Erweiterungsmoduls NumPy. NumPy stellt dabei mit dem weiteren OpenSource Erweiterungsmodul SciPy eine kostenlose Alternative zu MATLAB dar. Python ist eine modernere und allgemeinere Programmiersprache als MATLAB (KLEIN, 2013).

Da die Punktdaten in Point Data Record nur als ganzzahliger Wert verspeichert sind, ist es nötig die Werte in Koordinaten umzurechnen, wie bereits in Kapitel 5.3 beschrieben wurde. Dazu gibt es in Laspy zwei Möglichkeiten:

- Die Koordinaten werden mit Hilfe der Punktdaten, dem Skalierungsfaktor und dem Offsetwert aus dem Public Header Block berechnet (siehe Formeln [5.1] bis [5.3]) Dabei kann mit Laspy auf die Informationen im Public Header Block mit bestimmten Methoden zugegriffen werden:
  - `inFile.header.scale` liefert eine Liste, die die drei Skalierungswerte für X, Y und Z enthält.
    - Für X `inFile.header.scale[0]`
    - Für Y `inFile.header.scale[1]`
    - Für Z `inFile.header.scale[2]`

- `inFile.header.offset` liefert eine Liste, die die drei Offset Werte für X, Y und Z enthält
  - Für X `inFile.header.offset[0]`
  - Für Y `inFile.header.offset[1]`
  - Für Z `inFile.header.offset[2]`
- Kleingeschriebene Dimensionen, wie las_file.x, las_file.y, las_file.z, geben die skalierten Werte wieder.

7.2 Filtern von Boden- und Wasserpunkte

Da in den LAS Formaten bereits die Punkte in Klassen eingeteilt sind, können die Punkte ganz einfach zwischen Boden- und Nichtbodenpunkte unterschieden werden. Nach
Tabelle 5 sind Bodenpunkte mit der Klassennummer zwei verspeichert. Weiterhin werden auch Wasserpunkte (Klasse 9) mitgefiltert, damit Löcher in den Daten vermieden werden. Die Filterung sollte mit Matrizenfunktionen erfolgen, da Filterungen, die eine Schleife durchlaufen, bei den riesigen Datenmengen, die LIDAR Daten besitzen, extrem lange dauern können. Eine Filterung dieser großen Datenmengen sollte mit einer sogenannten Boolean Array und Maskierung erfolgen. Ziel dabei ist es, alle Punkte beziehungsweise Zeilen zu löschen, die keine Klasse zwei oder neun zugewiesen haben.
Ist der Klassenwert eines Punktes „2“ für Boden oder „9“ für Wasser, so wird für den Punkt bzw. für die Zeile ein „True“ in den Boolean Array geschrieben. Ist die Bedingung nicht erfüllt, wird „False“ für diesen Punkt in den Boolean Array verspeichert. Da es sich um zwei Bedingungen handelt, für Boden- und Wasserpunkte, wird die Built-in Funktion „Any“ verwendet. „Any“ überprüft, ob eine der zwei Bedinungen erfüllt ist, also ob es sich entweder um einen Bodenpunkt oder um einen Wasserpunkt handelt. Diese Boolean Array wird mit dem originalen Array maskiert. So werden alle Zeilen des originalen Arrays gelöscht, die in der Boolean Array mit einem „False“ übereinstimmen.

7.3 Berechnung der Zielrasters

Für die spätere Interpolation müssen regelmäßige Gitterpunkte berechnet werden, an denen die interpolierten Werte aus den unregelmäßig verteileten LIDAR Punktdaten berechnet werden. Dabei sind die Punkte für ein regelmäßiges Gitter abhängig von der Ausdehnung der Eingabedatei und von der Anzahl an Zeilen und Spalten des Zielrasters. Wie viele Zeilen und

Spalten ein Zielraster haben soll ist auch abhängig von der gewünschten Zellengröße. Die Anforderung in dieser Masterarbeit wird es sein, dass das interpolierte Raster eine Mindestzellengröße von 1 m hat.

Die Informationen zur Ausdehnung der LAS Datei befindet sich im Public Header Block (siehe Kapitel 5.1). Dazu liefert Laspy Methoden, um diese Informationen einfach lesen zu können. `inFile.header.min` liefert eine Liste, die die Minimum Koordinaten von X, Y und Z enthält und `inFile.header.max` liefert eine Liste, die die Maximum Koordinaten und Höhenwerte für X, Y und Z beinhaltet:

- Für X `inFile.header.min[0]` bzw. `inFile.header.max[0]`
- Für Y `inFile.header.min[1]` bzw. `inFile.header.max[1]`

Anzahl der Zeilen und Spalten lassen sich mit den Formeln [7.1] und [7.2] errechnen:

$$Anzahl\ an\ Spalten = \frac{X_{Max} - X_{Min}}{Zellengröße + 1} \qquad [\ 7.1\ ]$$

$$Anzahl\ der\ Zeilen = \frac{Y_{Max} - Y_{Min}}{Zellengröße + 1} \qquad [\ 7.2\ ]$$

Mit der Python *linespace*-Funktion können nun die regelmäßig verteilten Koordinaten über ein Intervall generiert werden. Dazu wird die Angabe der minimalen und maximalen Koordinaten von X und Y (X/Y_{Max} und X/Y_{Min}) benötigt, die das Intervall begrenzen. Weiters ist die Anzahl der Punkte (Anzahl der Zeilen und Spalten), die im Intervall generiert werden sollen, nötig. Dies wird separat für die Koordinaten X und Y durchgeführt (Vgl. Tabelle 7). Für X spielt dabei die Anzahl der Spalten eine Rolle, wohingegen für Y die Zahl der Zeilen berücksichtigt werden muss (Abbildung 8). Startwert für das jeweilige Intervall muss die obere linke Ecke der Datei sein. Ergebnisse sind allerdings Koordinatenvektoren, daher müssen diese mit der Numpy-Funktion *meshgrid* in Matrizen umgewandelt werden (Tabelle 7).

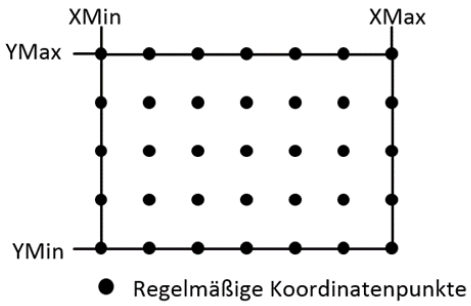

Abbildung 8: Erstellung des regelmäßigen Koordinatengitters mit den Python Funktionen linespace und meshgrid (eigene Darstellung)

Tabelle 7: Verspeicherung der Koordinaten mit den Python Funktionen *linespace* und *meshgrid* (eigene Darstellung)

Vektorformat der Koordinaten mit *linespace*:	**Matrizenformat der Koordinaten mit *meshgrid:***
$[X_{Min} \ldots\ldots X_{Max}]$	$\begin{bmatrix} X_{Min} & \cdots & X_{Max} \\ \vdots & \ddots & \vdots \\ X_{Min} & \cdots & X_{Max} \end{bmatrix}$
$[Y_{Max} \ldots\ldots Y_{Min}]$	$\begin{bmatrix} Y_{Max} & \cdots & Y_{Max} \\ \vdots & \ddots & \vdots \\ Y_{Min} & \cdots & Y_{Min} \end{bmatrix}$

7.4 Interpolation mit Python

Zunächst wird untersucht, welche Interpolationsmethoden bereits in Python als Methode bereitgestellt werden. Python stellt in den OpenSource Erweiterungsmodulen fertige Interpolationsfunktionen zur Verfügung. Erweiterungsmodule für Interpolationen sind SciPy und Matplotlib. Folgende Funktionen werden bei den jeweiligen Modulen bereitgestellt:

- SciPy
  - Multivariate interpolation
    - Multivariate data interpolation (griddata)
      - Linear
      - Nearest
      - Cubic
    - Spline Interpolation
    - Radial basis functions
      - Multiquadric
      - Inverse
      - Gaussian
      - Linear
      - Cubic

        - Quintic
        - Thin Plate
- Matplotlib
    - Matplotlib.mlab.griddata
        - Natural neighbor
        - Linear Interpolation

Diese Funktionen eignen sich ideal für eine Interpolation von unregelmäßig verteilten Punkten auf ein regelmäßig verteiltes Raster. Allerdings konnten bei der Verarbeitung in hochauflösende Raster nur die *Gridata* Funktionen Ergebnisse liefern. Sowohl die *Radial basis* Funktionen als auch die *Matplotlib* Funktion konnten keine Raster in 1 m Auflösung erstellen. Ohne eine Fehlermeldung konnten nur Raster ab 10 m Auflösung erstellt werden.
Deshalb wird zur Interpolation der LIDAR Daten die *Griddata* Funktion von SciPy verwendet. Diese Funktion interpoliert unstrukturierte D-dimensionale Daten. Diese Funktion bietet drei Methoden, um die Daten zu interpolieren. Mit der Methode „nearest“ kann eine Nearest Neigbour Interpolation durchgeführt werden. Diese gibt den Wert der Datenpunkte wieder, die am nähesten zum Interpolationspunkt liegen. Die Methode „linear“ interpoliert die Daten mit einer stückweisen linearen Interpolation. Mit der „cubic“ Methode ist eine stückweise kubische Interpolation möglich.
Für die *Griddata* Funktion benötigt man Koordinaten der Datenpunkte in einem Array, wobei die erste Spalte die X-Koordinate ist und die zweite Spalte die Y-Koordinate beinhaltet. So entspricht jede Zeile im Array einem Datenpunkt. Als zweiter Input werden die Höhenwerte der Datenpunkte benötigt, die auch in einem Array abgespeichert werden. Dabei hat dieses Array nur eine Spalte mit den Höhenwerten. Jede Zeile entspricht dem Höhenwert der jeweiligen Koordinatenpunkte. Außerdem benötigt diese Funktion das berechnete regelmäßige Raster, an denen die Koordinaten interpoliert werden sollen. Weiterhin ist die Wahl der Methode erforderlich. Optional kann auch ein „Füllwert“ angegeben werden, der die nicht berechneten Werte ersetzt, Standardeinstellung ist Nan (Not A Number). Diese Option hat jedoch keinen Effekt auf die „nearest“ Methode.

7.5 Inverse Distance Weighting Interpolation mit Python

Da in der Literatur (Kapitel 3.4) unter anderem die Inverse Distance Weighting Methode gute Ergebnisse bei der Interpolation von LIDAR Daten lieferte, wird diese aus diesem Grund für die Masterarbeit in Python programmiert.

Die Inverse Distance Weighting Methode wird von BONHAM-CARTER (1994, S. 152/153) ausführlich definiert:

Das IDW ist eine der einfachsten Methoden, die die gewichteten bewegenden Mittelwerte von Punkten innerhalb einer Einflusszone, die gewöhnlich kreisförmig ist, verwenden. IDW basiert auf Gewichtungen, die inverse proportional zum Quadrat der Distanz vom Zentrum der Einflusszone sind. Abbildung 9 zeigt das Prinzip des IDWs. Die generelle Formel von IDW lautet:

$$\hat{Z}_0 = \frac{\sum_{i=1} w_i z_i}{\sum_{i=1}^{n} w_i} \qquad [7.3]$$

Das Dach über dem Z zeigt einen Schätzwert der Oberflächenhöhe. Die tiefgestellte 0 bezieht sich auf den Schätzwert (estimation point) und das tiefgestellte i auf die Probenpunkte (sample points), die sich innerhalb der Einflusszone befinden. Die Gewichtung bezieht sich auf die Distanz mit $w_i = 1/d_{i0}^2$, wobei d_{i0} die Distanz zwischen den Punkt i und dem Punkt 0 darstellt.

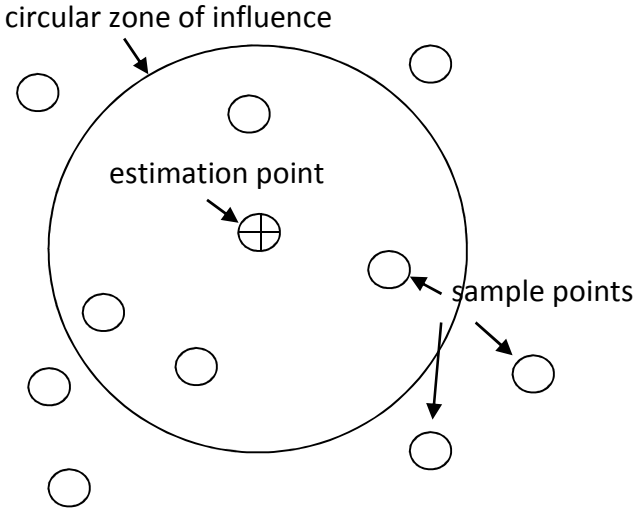

Abbildung 9: Prinzip des IDWs (nach BONHAM-CARTER, 1994, S. 152/153)

Diese Gewichtungsbeziehung hat den Effekt, dass Datenpunkte nahe dem zu interpolierenden Punkt einen größeren Einfluss auf den Schätzwert haben, wohingegen Punkte, die weit weg liegen, weniger Einfluss ausüben. Man muss dabei nicht unbedingt die inverse Distanz auf die Potenz 2 anheben, es können auch andere Exponenten verwendet werden, bei welchen sich die Zerfallsrate der gewichteten Funktion mit wachsender Distanz verändert.

IDW sollte verwendet werden, wenn die Menge und die Punkte dicht genug sind, um den Umfang der lokalen Oberflächenvariation zu erfassen, die für die Analyse benötigt wird (CHILDS, 2004).

Um nun in Python auf die Nachbarn eines Punktes zugreifen zu können, gibt es die im SciPy Erweiterungsmodul enthaltene Klasse *cKDTree*. Diese Klasse liefert einen Index in einer Menge von k-dimensionalen Punkten, der verwendet werden kann, um schnell die Nachbarn eines beliebigen Punktes zu finden (SCIPY.ORG, 2014).

7.5.1 Kd-Baum

Für den Inverse Distance Weighting Algorithmus sind zwei Klassen von Anfragen wichtig, die „nächste Nachbar"-Frage und die Entfernungsfrage. Im ersten Fall wird von einem Punkt aus verlangt, dessen nächste Nachbarn zu finden. Im zweiten Fall sollen alle Punkte, die innerhalb einer definierten Entfernung zu einem bestimmten Punkt liegen, gezählt werden (MANEEWONGVATANA & MOUNT, 1999). Solche Anfragen können durch eine einfache und meist bekannte Datenstruktur beantwortet werden: den kd-Baum von BENTLEY (1975).

BENTLEY (1975) beschreibt den kd-Baum als einen binären Suchbaum, wobei „k" die Dimensionalität des Suchraumes beschreibt. Der kd-Baum ist dabei eine Datenstruktur für die Speicherung von Informationen, die durch assoziative Suche abgefragt werden. Abbildung 10 verdeutlicht den Grundgedanken eines kd-Baumes (MANEEWONGVATANA & MOUNT, 1999; BENTLEY, 1975):

Wird eine Datei in einem kd-Baum repräsentiert, dann wird jede Koordinate in der Datei als Knoten im Baum verspeichert. Der Raum wird dabei hierarchisch durch Teilungshyperebenen in zwei Hälften unterteilt. Dabei befindet sich die Hyperebene orthogonal zu einer Dimensionsachse. So wird zum Beispiel im zweidimensionalen Fall (k = 2) der Raum entweder entlang der X-Achse oder entlang der Y-Achse geteilt. Eine der beiden Koordinaten im Punkt wird also als Teilungsebene verwendet. Die zur Teilung verwendeten Koordinaten werden als Diskriminator verwendet. Der Diskriminator ist eine ganze Zahl zwischen 0 und k-1. Der Diskriminator ist für alle Knoten auf einer Ebene des Baums gleich.

Jeder Knoten hat zwei Zeiger, die entweder Null annehmen können oder zu einem anderen Knoten im kd-Baum zeigen. Punkte auf der linken Seite dieser Hyperebene repräsentieren den linken Baumzweig, Punkte auf der rechten Seite der Hyperebene repräsentieren hingegen den

rechten Baumzweig. Wird also der Raum entlang der X-Achse geteilt (entspricht also dem Diskriminator 0), so werden alle Punkte mit kleinerem X-Wert als Knoten im linken Zweig und alle Punkte mit größeren X-Wert als Knoten im rechten Baumzweig verspeichert. Die zwei Knotensöhne teilen den Raum durch die jeweilige andere Achse. Das bedeutet für diese Knotensöhne, dass diese den Raum entlang der Y-Achse (Diskriminator 1) teilen (BENTLEY, 1975). Dieser Teilungsprozess wird rekursiv wiederholt. Der kd-Baum in der Abbildung 10 entsteht, wenn die Datensätze beispielsweise in alphabetischer Reihenfolge A bis G eingeführt werden. Das Aussehen des Baumes wird daher im Wesentlichen durch die Einfügereihenfolge und Verteilung der Punkte bestimmt (HÄRDER & RAHM, 2001, S.257).

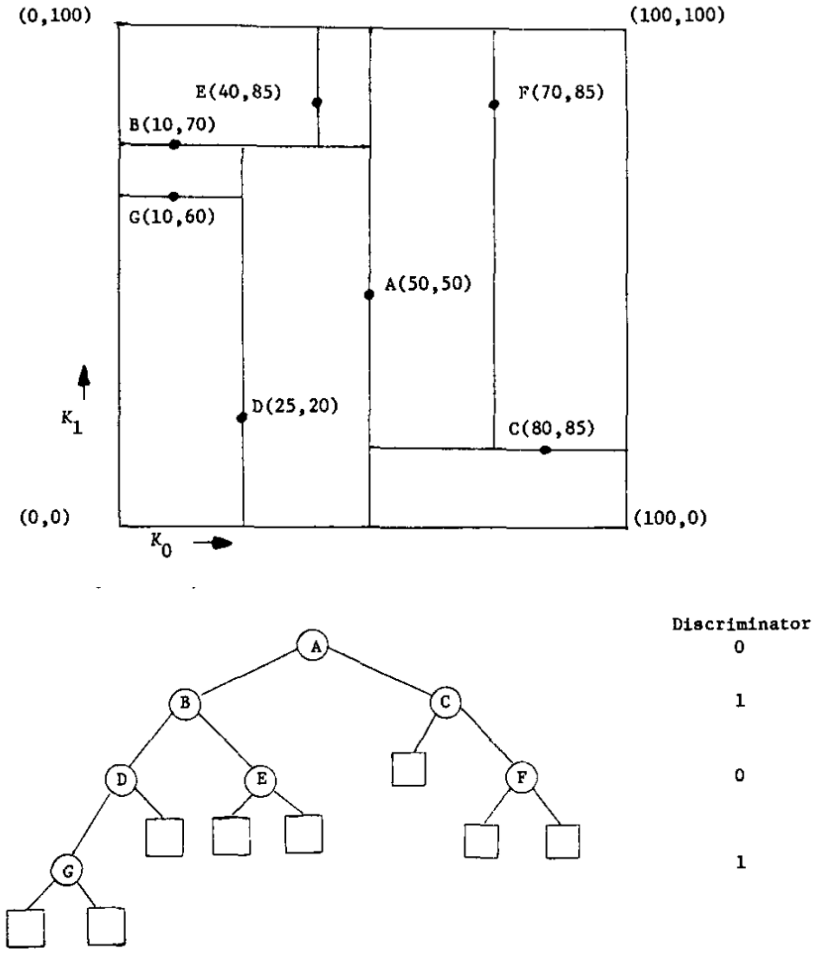

Abbildung 10: Koordinaten in einem zweidimensionalen Raum, die als Knoten in einem 2-d Baum gespeichert werden (BENTLEY, 1975)

Der Algorithmus in Python basiert dabei auf der sogenannten der Teilungsregel „Midpoint splitting rule“ von MANEEWONGVATANA & MOUNT (1999). Bei dieser Regel teilt die Teilungshyperebene den Raum in der Mitte von der längsten Achse. Gibt es mehrere Achsen

mit der gleichen Länge, dann wird irgendeine Achse als erstes gewählt, z.B. die Achse mit dem kleinsten Koordinatenindex.

Eine andere Teilungsregel ist zum Beispiel der „Standard split“. Da wird die Teilungsdimension so gewählt, dass sie für die Datenpunkte am weitesten auseinander liegt (Differenz zwischen Maximum- und Minimumwerten) und der Teilungswert ergibt sich aus dem Koordinatenmedian der Punkte in dieser Dimension (MANEEWONGVATANA & MOUNT, 1999).

7.5.2 Kd-Baum Methoden in Python

Das OpenSource Erweiterungsmodul SciPy liefert für die Klasse cKDTree zwei Methoden, um die nächsten Nachbarn eines Punktes zu finden (Tabelle 8).

Tabelle 8: Methoden für die cKDTree Klasse in SciPy (Selbsterstellt nach SCIPY.ORG, 2014)

Methode	**Erklärung**
query(self, x[, k, eps, p, distance_upper_bound])	Diese Methode sucht die nächsten Nachbarn von einen oder mehreren Punkten (=x), abhängig von: • Anzahl der Nachbarn, die wiedergegeben werden sollen (= k) • der Distanz der Nachbarn zum Punkt x (= distance_upper_bound) Wiedergabe sind die Indexwerte aller Nachbarpunkte in einem Array und deren Distanz zum Punkt x in einem zweiten Array.
query_ball_point(self, x, r, p, eps)	Diese Methode gibt Indexwerte von einem oder mehreren Punkten wieder, die sich innerhalb einer bestimmten Distanz (= r) von Punkt x befinden. Werden nur die Nachbarn eines Punktes gesucht, so ist die Wiedergabe eine Liste. Werden von mehreren Punkten die Nachbarn gesucht, dann ist die Wiedergabe ein Array.

Abbildung 11 verdeutlicht die Unterschiede beider Methoden. Die Methode *query_ball_point* gibt die Nachbarn in einen bestimmten Radius wieder, also auch die Punkte, die noch auf dem Radius liegen. Die Methode *query* gibt die Nachbarn innerhalb eines Radius wieder und keine Punkte, die genau auf dem Radius liegen. Außerdem wird in Abbildung 11 noch eine weitere Besonderheit der Methode *query* verdeutlicht. Die Methode gibt anstatt fünfzehn nur neun Nachbarn wieder, da nur neun Nachbarn innerhalb der Distanz distance_upper_bound = 2 liegen. In Python wird im Ergebnis für diese nicht besetzten Nachbarn einfach ein ungültiger Indexwert angegeben. Im Beispiel hat das Array sechzehn Elemente und da das erste Element mit dem Index Null beginnt, hat das Array fünfzehn Indexe, und so werden nichtbesetze

Nachbarfelder im Array mit dem Index „16“ besetzt. In der dazugehörigen Distanzarray werden die nichtbesetzten Nachbarn mit unendlich („Inf“) belegt. Jede Zeile im Ergebnisarray entspricht dabei den Nachbarn eines Punktes (Abbildung 12).

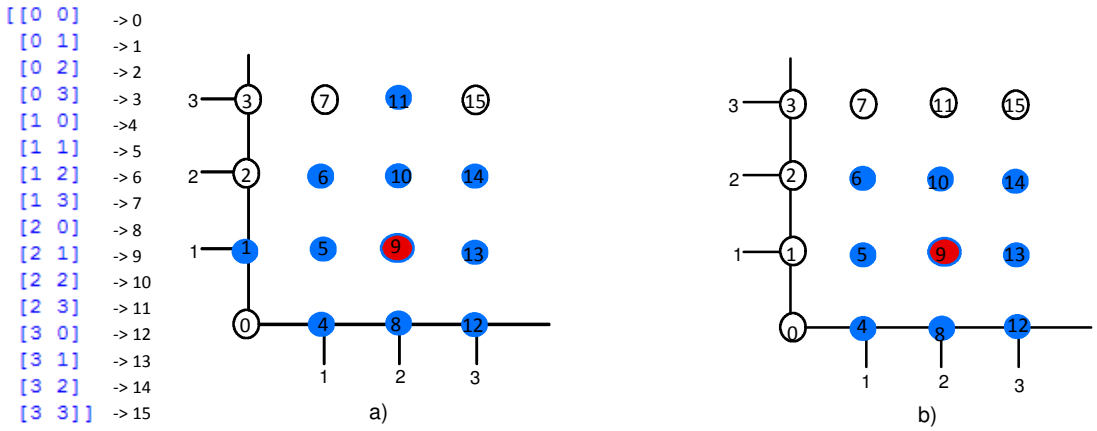

Abbildung 11: Links: Beispielkoordinaten als Array in Python, in denen die Nachbarn für z.B. den Punkt (2,1) gesucht werden sollen (eigene Darstellung).

a) Suche nach dem nächsten Nachbarn (blaue Punkte) mit der Methode query_ball_point mit r = 2 für den Punkt (2,1) (roter Punkt). Ergebnisse in Python als Indexzahlen (vgl. mit Array links): [1, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14] (Hinweis: Indexbeginn ist Null).

b) Suche nach dem nächsten Nachbarn (blaue Punkte) mit der query Methode mit k = 15 und distance_upper_bound = 2 für den Punkt (2,1) (roter Punkt). Ergebnisse in Python als Indexzahlen (vgl. mit Array links): [9, 10, 8, 13, 5, 4, 14, 6, 12, 16, 16, 16, 16, 16, 16] (Hinweis: Indexbeginn ist Null) und Ergebnisse der Distanzen der Nachbarpunkte zu dem Punkt (2, 1) in Python: [0, 1, 1, 1, 1, 1.414, 1.414, 1.414, 1.414, inf, inf, inf, inf, inf]

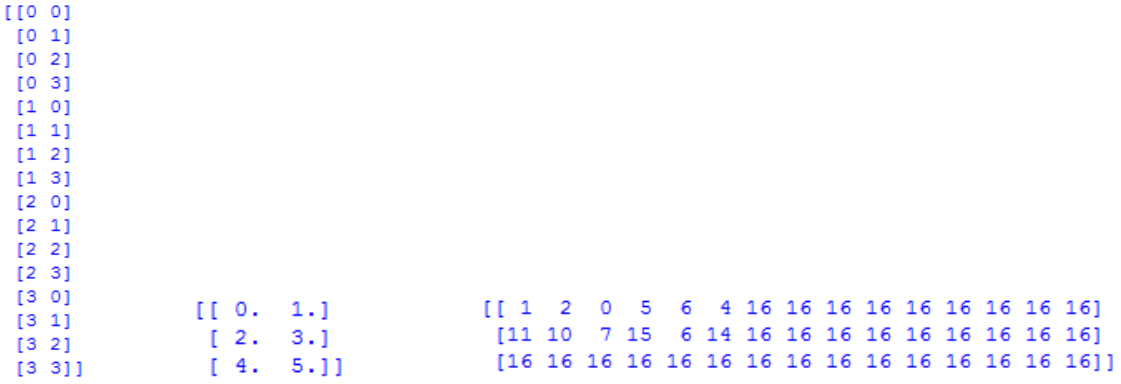

Abbildung 12: a) Beispielkoordinaten als Array in Python. b) Punkte, dessen Nachbarn gesucht werden sollen. c) Ergebnisse in Python als Indexzahlen in einem Array mit der Methode „query“ (k = 15, distance_upper_bound = 2) (eigene Darstellung)

7.5.3 IDW Berechnung mit der Methode „query ball point“ in Python

Wird die Methode „*query ball point*“ zur IDW Berechnung gewählt, so liefert diese die Indexzahlen aller Nachbarn wieder, jedoch nicht die Distanzen, wie bei der Methode „*query*“.

Werden nun mit Hilfe einer Schleife durch die regelmäßigen Gitterwerte (`interp`) für jeden Gitterpunkt (`i`) einzeln die Nachbarn ermittelt, so erhält man kein Array der Nachbarn, sondern eine Liste von Indexen pro Gitterpunkt (`neigh_ind`). Mit Hilfe dieser Liste können innerhalb dieser Schleife durch „Advanced Indexing“ nun die Koordinatenwerte aus der Punktwolke (`obs`) für den jeweiligen Gitterpunkt „ausgeschnitten“ werden, so erhält man die Koordinaten der Nachbarn in einem Array (`neigh`).

Die Distanzen der Nachbarn zu diesem Gitterwert ergeben sich aus dem Satz des Pythagoras. Dazu werden vom Gitterwert, der gerade mit der Schleife durchlaufen wird, und von seinen Nachbarn (`neigh`) die Differenzwerte ihrer X-Koordinaten (Spalte „0“) und Y-Koordinaten (Spalte „1“) mit der Python Funktion „*numpy.substract.outer*“ berechnet. Ergebnisse sind Arrays (`delta_x, delta_y)`, die die Differenzen der Koordinaten jedes Nachbarn zum Punkt enthalten. Mit der Funktion „*numpy.hypot*“ kann nun aus den Differenzwerten der Koordinaten die Hypertenuse jedes Nachbarn zum Punkt berechnet werden. Ergebnis ist ein Distanzarray (`distance`), das die Distanzen jedes Nachbarn zum Punkt enthält.

```
    for i in interp:
        #Finde Indexe für jeden regelmäßigen Gitterpunkt
        neigh_ind = tree.query_ball_point(i, radius)
        neigh = obs[neigh_ind]
        #Distanz berechnen
        delta_x = np.subtract.outer(neigh[:,0], i[0])
        delta_y = np.subtract.outer(neigh[:,1], i[1])
        distance = np.hypot(delta_x, delta_y)
```

Mit Hilfe dieser Distanzen und der Nachbarn ergibt sich durch die IDW Formel mit der „*numpy.sum*“ Funktion der interpolierte Wert an einem Gitterpunkt, indem man den Zähler (`num`) durch den Nenner (`denum`) teilt:

```
        num = np.sum(values[neigh_ind]/(distance**power))
        denom = np.sum(1/distance**power)
```

Ist allerdings der Nenner Null, wenn keine Nachbarn gefunden werden, so wird der gesamte Punkt mit “-9999” ersetzt.

Diese Methode hat jedoch den Nachteil, dass sie sehr lange Rechenzeit benötigt, da diese mit einer Schleife durch jeden Gitterpunkt geht und den interpolierten Wert einzeln berechnet. Handelt es sich also um ein sehr großes Gitter, so ist ein Schleifendurchlauf ungünstig.

Außerdem kann bei dieser Methode nur der Radius eingegrenzt werden, es kann jedoch nicht spezifiziert werden, wie viele Nachbarn gesucht werden sollen.

7.5.4 IDW Berechnung mit der Methode „query“ in Python

Für die Berechnung des IDWs ist die Methode *query* am besten geeignet, da der Benutzer nicht nur die Distanz bestimmen kann, sondern auch wie viele Nachbarn innerhalb einer bestimmten Entfernung gefunden werden sollen. Außerdem ist es nicht nötig, einen interpolierten Wert für jeden Gitterpunkt mit einer Schleife einzeln zu berechnen. Dabei ist zu beachten, dass aufgrund möglicher Unendlichwerte im Distanzarray alle „Inf“ mit „Not a Number“ („nan“) ersetzt werden, da eine Division durch Unendlich zu Fehlermeldungen führt.

```
w = 1.0 / distance**power
idw = np.nansum(w * values[index], axis=1) /np.nansum(w, axis=1)
```

Der Nachteil besteht darin, dass es bei dieser Methode sehr schnell zu Problemen kommen kann, wenn sofort für alle Punkte im regelmäßigen Rastergitter die Nachbarn berechnet werden sollen. Da dieses regelmäßige Raster sehr groß sein kann, kommt es sehr schnell zu einem „Memory Error“. Daher sollte das regelmäßige Raster in Untergebiete geteilt werden. Von diesen Untergebieten werden die interpolierten Werte berechnet und später wieder zusammen gesetzt. Dazu sollten aber trotzdem alle Beobachtungspunkte in einem kd-Baum gespeichert werden.

Die Anzahl der regelmäßigen Gitterpunkte, die bei der nächsten Nachbarsuche ohne eine Fehlermeldung verarbeitet werden können, ist abhängig von der Größe des Gebietes und der Anzahl der zu suchenden Nachbarn.

Das Untersuchungsgebiet wird daher in Abhängigkeit dieser Parameter geteilt, so dass für jede neue Datei eine individuelle Zahl an Untergebieten entsteht. Die Größe eines Untergebietes ergibt sich aus:

$$x = \frac{maximale\ Ausgangsgöße}{k} \qquad [7.4]$$

x ist dabei die Anzahl der regelmäßigen Gitterpunkte in einem Untergebiet, k wiederum die Anzahl der zu suchenden Nachbarn. Die maximale Ausgangsgröße wurde durch Testen gefunden.

7.6 Speichern der Interpolationsergebnisse

Zum Schluss wird das interpolierte Raster im ASCII Grid Format verspeichert. Dieser Datentyp wird oft für Höhendaten verwendet. Vorteil des ASCII Grid Formates für ein Raster ist, dass dieses Format weit verbreitet ist und von den meisten Softwarepaketen unterstützt wird. Außerdem kann das ASCII Grid Format auch wieder direkt von NumPy gelesen werden.
Dieses Rasterformat speichert Daten als Text und dabei wird jede quadratische Rasterzelle in einer Zeile und Spalte durch einen einzelnen numerischen Wert dargestellt, der die Eigenschaften des Geländes, wie zum Beispiel die Höhe, repräsentiert. Die räumliche Information für das Raster wird in die Kopfzeile geschrieben, die aus sechs Zeilen besteht (LAWHEAD, 2013):

- ncols (beschreibt die Anzahl an Spalten und repräsentiert die X-Achse)
- nrows (beschreibt die Anzahl an Zeilen und repräsentiert die Y-Achse)
- xllcorner (beschreibt die X Koordinate in der linken unteren Ecke des Rasters, welches der minimum X-Wert ist)
- yllcorner (beschreibt die Y Koordinate in der linken unteren Ecke des Rasters, welches der minimum Y-Wert ist)
- cellsize (beschreibt die Zellengröße bzw. die Auflösung des Rasters)
- NODATA_value (beschreibt eine Zahl, die zu irgendeiner Zelle zugeordnet sein kann, die keinen Wert hat, z.B. -9999)

7.7 Erstellen eines Werkzeuges zur Interpolation

Der Programmablauf zur Verarbeitung von LAS-Dateien zu einem digitalen Geländemodell wird in der Abbildung 13 dargestellt. Der gesamte Code befindet sich im Anhang. Für die Verabeitung der LAS-Dateien in Python müssen die OpenSource Erweiterungsmodule NumPy, SciPy und Laspy installiert und im Pythonskript importiert werden.
Der Anwender kann dabei eine Eingabedatei in LAS Format und die Zellengröße selbst festlegen. Weiterhin kann zwischen der IDW Methode und den multivariaten Methoden gewählt werden. Dem Nutzer ist es möglich, auch die IDW Parameter auszuwählen, wie die Wahl des Exponenten, die Zahl der Nachbarn, die gesucht werden sollen und den Suchradius. Jede ungültige Benutzereingabe wird mit Schleifen abgefangen, zum Beispiel, wenn der Pfad oder die angegebene Datei nicht existiert, oder wenn Buchstaben statt Zahlen für die Zellengröße eingegeben werden, oder wenn die Befehle falsch geschrieben werden.

Abbildung 14 und Abbildung 15 zeigen die Programmausführung in der Python Kommandozeile. Vorteil eines Pythonskriptes in der Kommandozeile ist, dass es rein aus Opensource Erweiterungsmodulen besteht und unabhängig von kostenpflichtigen Lizenzen, wie zum Beispiel ArcGIS, ausgeführt werden kann. Nachteil ist hingegen, dass die Ausführung in der Kommandozeile für Benutzer, die sich damit nicht so gut auskennen, benutzerunfreundlicher ist.

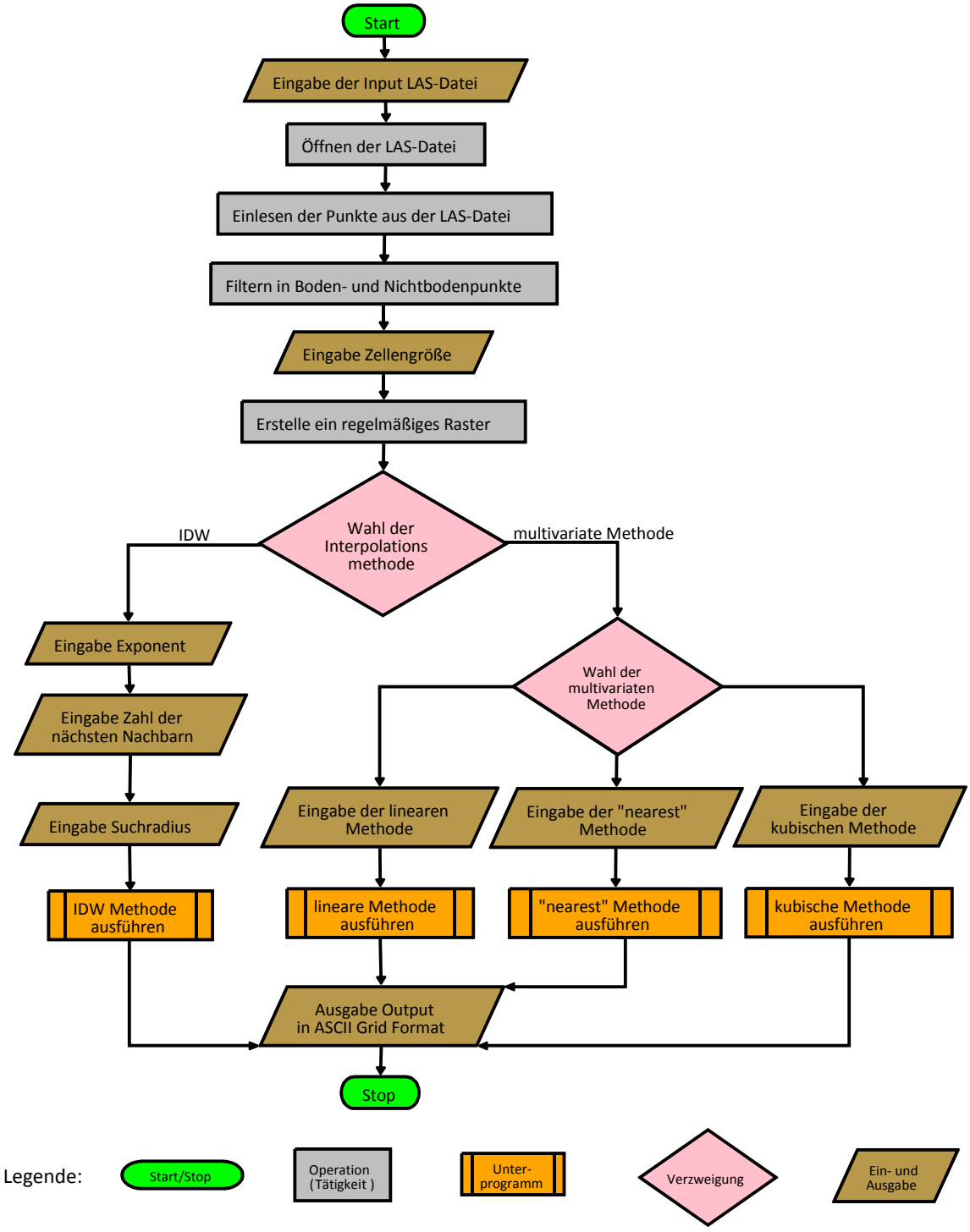

Abbildung 13: Programmablauf zur Generierung eines digitalen Geländemodells aus LAS-Daten mit Python (eigene Darstellung)

```
>>>
Pfad zum Ordner:
D:\JULIA\Masterarbeit\Daten\lorenzen_laserscann_testdaten
Name der Eingabe-LAS-Datei:
28002
Die Zellengröße im zu erstellenden Ausgabe-Raster:
1
Die Interpolationsmethode, die zum Erstellen des Rasters verwendet wird:
 Wähle: 'idw' oder 'multivariat'
     idw - Verwendet inverse Entfernungsgewichtung (Inverse Distance Weighted)-Interpolation
     multivariat - Verwendet multivariate Interpolationsmethoden
idw
Exponent der Entfernung:
 Je höher der Exponent,desto weniger Einfluss haben entferntere Punkte.
 Gültige Eingabe ist jede reelle Zahl größer 0
 Die besten Ergebnisse liefern Werte zwischen 0.5 und 3
 Standardwert: 2
2
Die Zahl der nächsten Nachbarn, die wieder gegeben werden sollen. z.B. 12:
12
Suchradius z.B. 15:
 Gibt nur Nachbarn innerhalb dieser Distanz wieder.
15

Raster erfolgreich berechnet.

Laufzeit der Methode in Sekunden: 50.0
Name der Outputdatei:
28002_Raster
>>>
```

Abbildung 14: Ausführung des Programms in der Kommandozeile mit Wahl der IDW Methode (eigene Darstellung)

```
>>>
Pfad zum Ordner:
D:\JULIA\Masterarbeit\Daten\lorenzen_laserscann_testdaten
Name der Eingabe-LAS-Datei:
28002
Die Zellengröße im zu erstellenden Ausgabe-Raster:
1
Die Interpolationsmethode, die zum Erstellen des Rasters verwendet wird:
 Wähle: 'idw' oder 'multivariat'
     idw - Verwendet inverse Entfernungsgewichtung (Inverse Distance Weighted)-Interpolation
     multivariat - Verwendet multivariate Interpolationsmethoden
multivariat
Multivariate Interpolationsmethoden:
 Wähle: 'nearest', 'linear' oder 'cubic'
 nearest - Nearest-neighbour Interpolation
 linear - stückweise lineare Interpolation
 cubic - stückweise kubische Interpolation
nearest
Raster erfolgreich berechnet.

Laufzeit der Methode in Sekunden: 49.0
Name der Outputdatei:
28002_Raster_nearest
>>>
```

Abbildung 15: Ausführung des Programms in der Kommandozeile mit Wahl der Nearest Neighbour Methode (eigene Darstellung)

Weiters wurde das Pythonskript als Werkzeug in ArcGIS 10 integriert. Dazu wird das kostenpflichtige Python-Site Paket ArcPy benötigt. ArcPy muss am Anfang des Skripts mit dem Befehl `import arcpy` eingebunden werden. Die Eingabeparameter können dadurch über das Werkzeugdialogfeld angegeben werden, wodurch das Tool benutzerfreundlicher wird. Die

Ergebnisdatei ist ebenfalls eine ASCII Datei. Mit Hilfe des ToolValidators (Eigenschaften des Tools unter der Registerkarte Validation) kann das Verhalten des Werkzeuges angepasst werden. So können - abhängig nach Wahl der Methode - Felder aktiviert bzw. deaktiviert werden. Wählt der Benutzer die IDW Methode, werden automatisch die Felder für die Parameter der IDW Methode aktiviert. Mit Hilfe des ToolValidators können auch Fehlermeldungen während der Eingabe in die Werkzeugdialogfelder angezeigt werden. So wird zum Beispiel ein Fehler angezeigt, wenn der Benutzer sich für die multivariate Methode entscheidet, jedoch weder die „lineare“, „nearest“ noch die „cubic“ Methode auswählt.

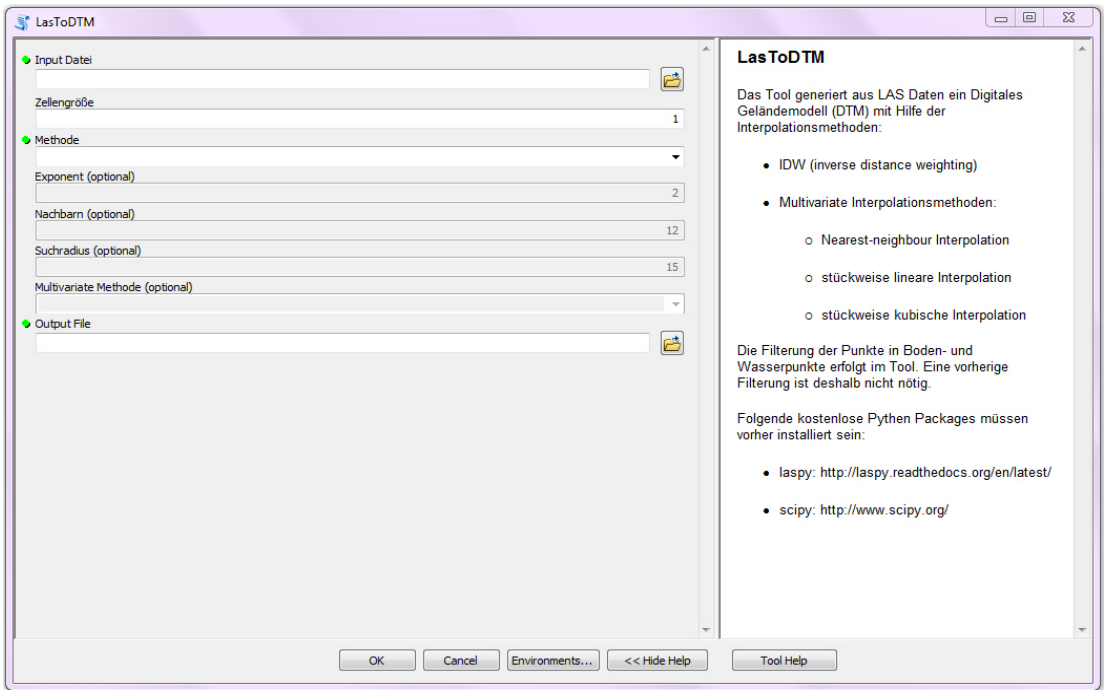

Abbildung 16: Integration des Pythonskriptes zur Generierung von digitalen Geländemodellen aus LAS Formaten als Werkzeug „LasToDTM“ in ArcGIS. Graue Felder werden je nach Wahl der Methode aktiviert (eigene Darstellung)

Tabelle 9 zeigt, dass die Laufzeiten der Methoden als ArcGIS Tools etwas schneller sind, als bei der Ausführung der Methoden in der Kommandozeile. Dabei ist die „nearest“ Methode die schnellste Methode.

Tabelle 9: Laufzeitvergleich zwischen Python Kommandozeile und ArcGIS Tool anhand des Untersuchungsgebietes 2 (eigene Darstellung)

Methode	**Python Kommandozeile**	**Werkzeug in ArcGIS**
IDW	12 sec.	13 sec.
linear	47 sec.	46 sec
nearest	10 sec.	9 sec.
cubic	80 sec.	77 sec.

8 Validierung

Um die Interpolationsmethoden zu beurteilen, werden die Genauigkeiten der erstellten DTMs mit Hilfe von quantitativen Fehlerwerten bestimmt (vgl. CHAPLOT ET AL., 2006; BATER & COOPS, 2009).:

- Root Mean Square Error (RMSE)
- Mean Error (ME)
- Mean Absolut Error (MAE)

Der Root Mean Square Error (RMSE) ist dabei einer der meist verbreiteten Angaben zum Messen von Fehlern zwischen zwei Datensätzen und ergibt sich aus folgender Formel:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(z_i^{interpolated} - z_i^{observed})^2}{n}} \quad [8.1]$$

$z^{interpolated}$ ist dabei der durch die Interpolation berechnete Wert, $z^{observed}$ ist der gemessene Höhenwert der LIDAR Bodenpunkte und n ist die Gesamtzahl aller Bodenpunkte. Diese Masterarbeit beschäftigt sich nur mit den Interpolationsfehlern, nicht jedoch mit dem LIDAR Messfehlern.

Der Mean Absolut Error ergibt sich aus:

$$MAE = \frac{\sum_{i=1}^{n}\left|\left(z_i^{interpolated} - z_i^{observed}\right)\right|}{n} \quad [8.2]$$

Der Mean Error ergibt sich aus der Gleichung:

$$ME = \frac{\sum_{i=1}^{n}(z_i^{interpolated} - z_i^{observed})}{n} \quad [8.3]$$

Um die interpolierten Werte mit den gemessenen Werten vergleichen zu können, werden die LIDAR Punktdatensätze der Untersuchungsgebiete in Trainings- und Testgebiete geteilt. Mit Hilfe der LIDAR Punktdaten im Trainingsgebiet werden die Interpolationen durchgeführt. Mit den LIDAR Punktdaten als gemessene Werte im Testgebiet werden die Genauigkeiten der Interpolationen bestimmt. Die Trennung der Datensätze in Trainings- und Testgebieten wird mit einer einfachen Validierung und mit einer 10-fach-Kreuzvalidierung beurteilt.

Die LIDAR Datenpunkte für die Untersuchungsgebiete werden zuerst zufällig ausgewählt und in zwei Datensätze geteilt: 90% für den Trainingsdatensatz und 10% für die Testdatensätze. Die Teilung der Datensätze erfolgt mit Hilfe von Python. Der Programmablauf wird in Abbildung 17 dargestellt. Ergebnis dieser Teilung ist das Trainingsgebiet in LAS-Format und

Testgebiete in Textformat mit X-, Y- und Z- Koordinaten. Diese Textdatei wird als Punkt-Feature mit dem Z-Höhenwert als Eigenschaft abgespeichert.

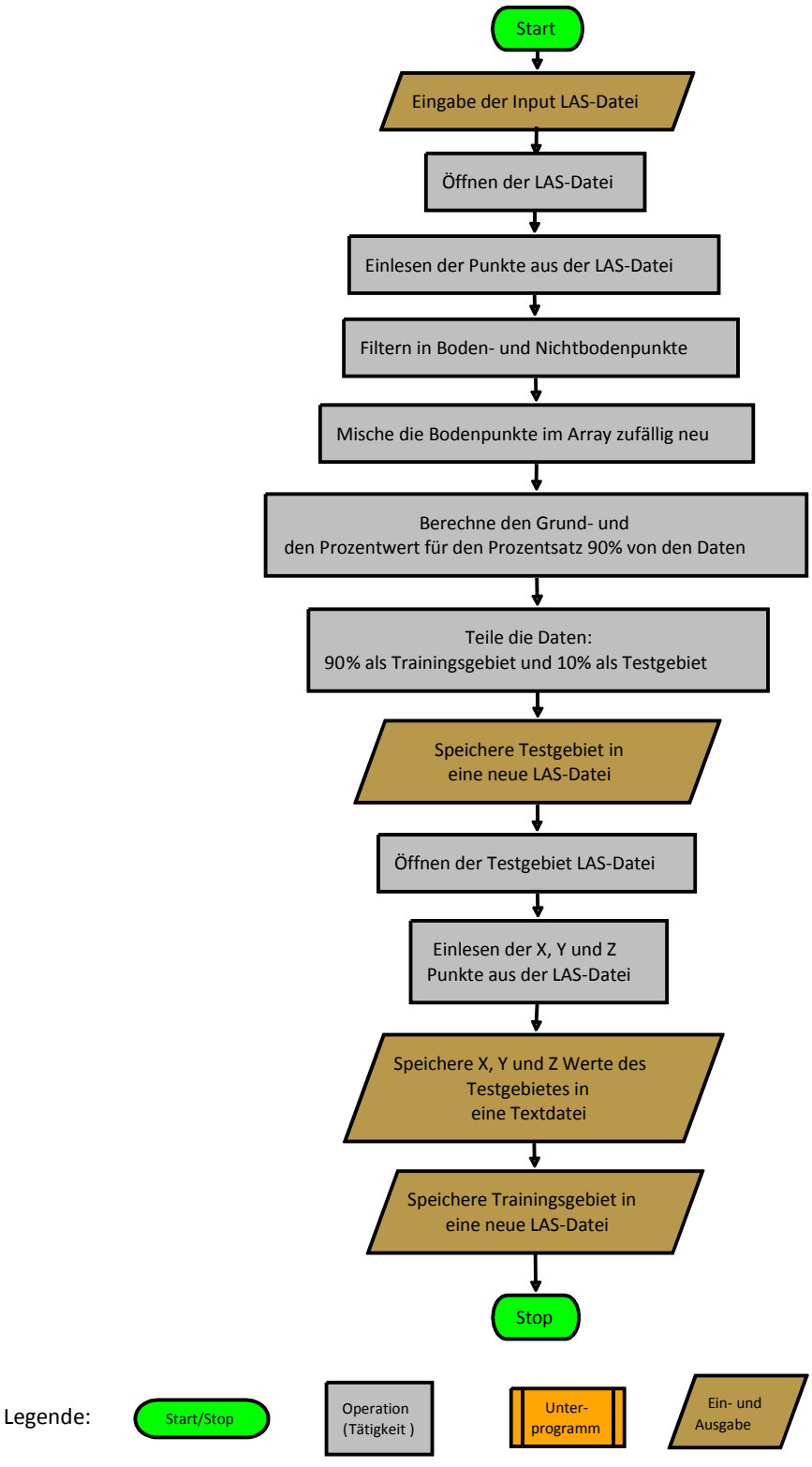

Abbildung 17: Programmablauf für die Teilung der Daten in Trainings- und Testgebiet mit Python (eigene Darstellung)

Bei der 10-fach-Kreuzvalidierung werden die LIDAR Punktdaten zuerst in zehn zufällige, möglichst gleich große Teilmengen geteilt. Eine der zehn Teilmengen wird als Testgebiet zur Validierung beibehalten, die restlichen neun Teilmengen werden als Trainingsdatensatz für die DTM - Interpolation verwendet. Dieser Prozess wird nun zehnmal wiederholt, so dass alle zehn Teilmengen für beides, als Trainingsgebiet und als Testgebiet, vewendet werden. Die Teilung der Datensätze für eine 10-fach-Kreuzvalidierung erfolgt mit Hilfe von Python und wird vereinfacht in Abbildung 18 dargestellt. Ergebnis dieser Teilung der Datensätze sind zehn Trainingsgebiete in LAS-Format und zehn Testgebiete in Textformat mit X-, Y- und Z-Koordinaten. Diese Textdatei wird als Punkt-Feature mit dem Z-Höhenwert als Eigenschaft abgespeichert.

Sowohl die einfache Validierung als auch die 10-fach-Validierung wird mit Hilfe von ArcGIS durchgeführt. Beide Methoden verfolgen das gleiche Prinzip (siehe Abbildung 19), wobei die Höheninformation des erstellten ASCII Grid Files mit den Höheninformationen des Testgebietes mit Hilfe des Tools „Add Surface Information“ verglichen wird. Da diese Information in die Eigenschaftstabelle des Punkt-Features vom Testgebiet übernommen wird, können die Differenzen zwischen den interpolierten Werten und den gemessenen Werten berechnet werden und als Datei exportiert werden, sodass der RMSE, MAE oder ME berechnet werden können.

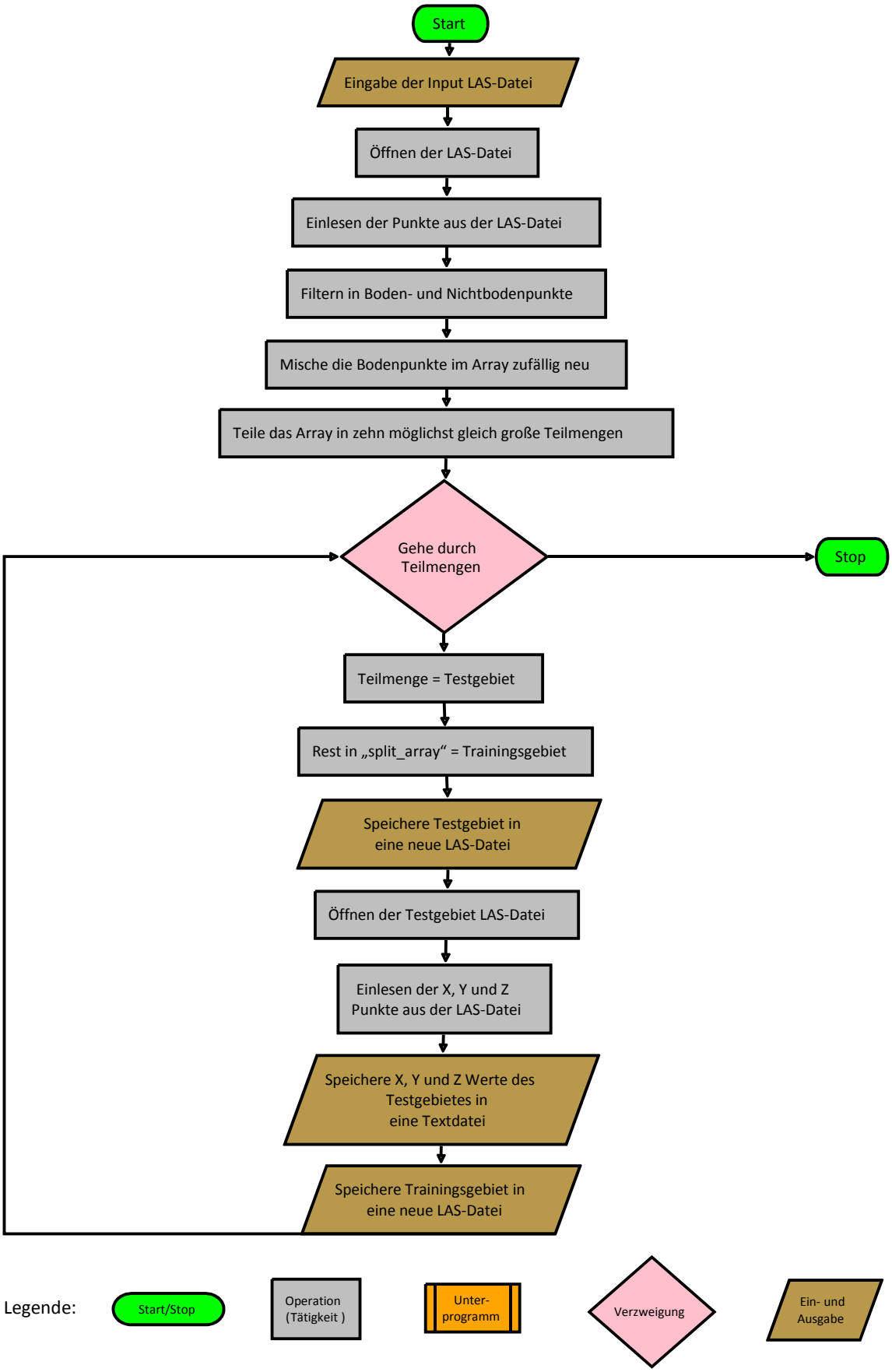

Abbildung 18: Programmablauf für die Teilung der Daten in Trainings- und Testgebiet für die 10-fach-Kreuzvalidierung mit Python (eigene Darstellung)

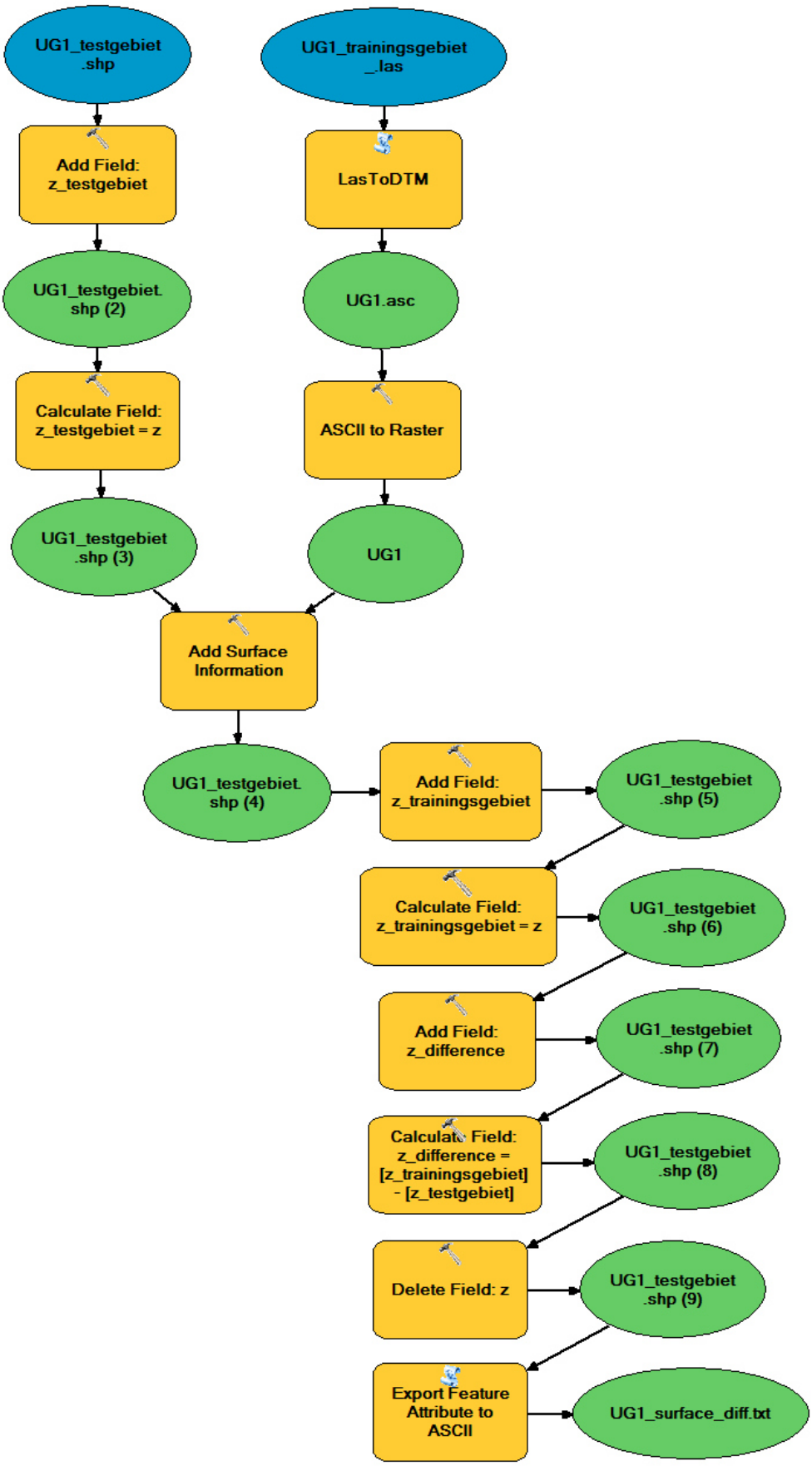

Abbildung 19: Einfache Validierung der Daten mit Trainings- und Testgebiet am Beispiel des Untersuchungsgebietes 1 (Model Builder ArcGIS: eigene Darstellung)

Für die Validierung wurde aus den Untersuchungsgebieten mit der programmierten IDW Methode und auch den multivariaten Methoden ein DTM berechnet. Zusätzlich wurden auch andere Tools, wie LASTools und LP360, die aus LIDAR Daten ein digitales Geländemodel interpolieren, mit validiert, um die Leistung des programmierten Werkzeuges mit diesen vergleichen zu können.

9 Ergebnisse

Abbildung 20 bis Abbildung 22 zeigen die Ergebnisse der einzelnen Interpolationswerkzeuge für die Untersuchungsgebiete. Im Vergleich zwischen den programmierten OpenSource Methoden zeigt die IDW Methde und die Nearest Neighbour Methode im Vergleich zu der linearen und kubischen Methode die besten Ergebnisse. Die IDW und die Nearest Methoden zeigen viele detaillierte Geländestrukturen, während die lineare und die kubische Methode das Gelände viel unschärfer darstellen. So sind in den Ergebnissen der linearen und der kubischen Methode Geländestrukturen und -änderungen weniger deutlich erkennbar. Im UG 1 wird dies unter anderem auch sehr deutlich bei den Hanglagen, aber auch bei sehr kleinen Strukturen, wie Rillen in etwas ebeneren Flächen. Im UG 2 sind bei diesen Methoden auch die Straßen und auch kleinere Geländeänderungen, wie Gräben, viel grober berechnet worden. Im UG 3 sind zum Beispiel besonders die Waldwege viel schlechter sichtbar.
Im Vergleich zum LASTools und LP360 gibt es für die IDW und Nearest Methode kaum optische Unterschiede für das UG 1 und das UG 2. Bei allen Methoden wird das Gelände sehr detailliert dargestellt und auch kleinste Geländeänderungen, sowohl in den Hanglagen als auch im flachen Talgebiet sind deutlich. In Abbildung 22 wird der Einfluss des Waldes und die daraus resultierende geringere Verfügbarkeit der Bodenpunkte erkennbar, denn die IDW Methoden (OpenSource Tool und LP360) und die Nearest Neighbour Methode zeigen deutliche Unterschiede zu LASTools und der Triangulation. Hier wird das Gelände im Wald bei LASTools und Triangulation eher glatter dargestellt, während alle IDW Methoden das Gelände eher rauer berechnen.

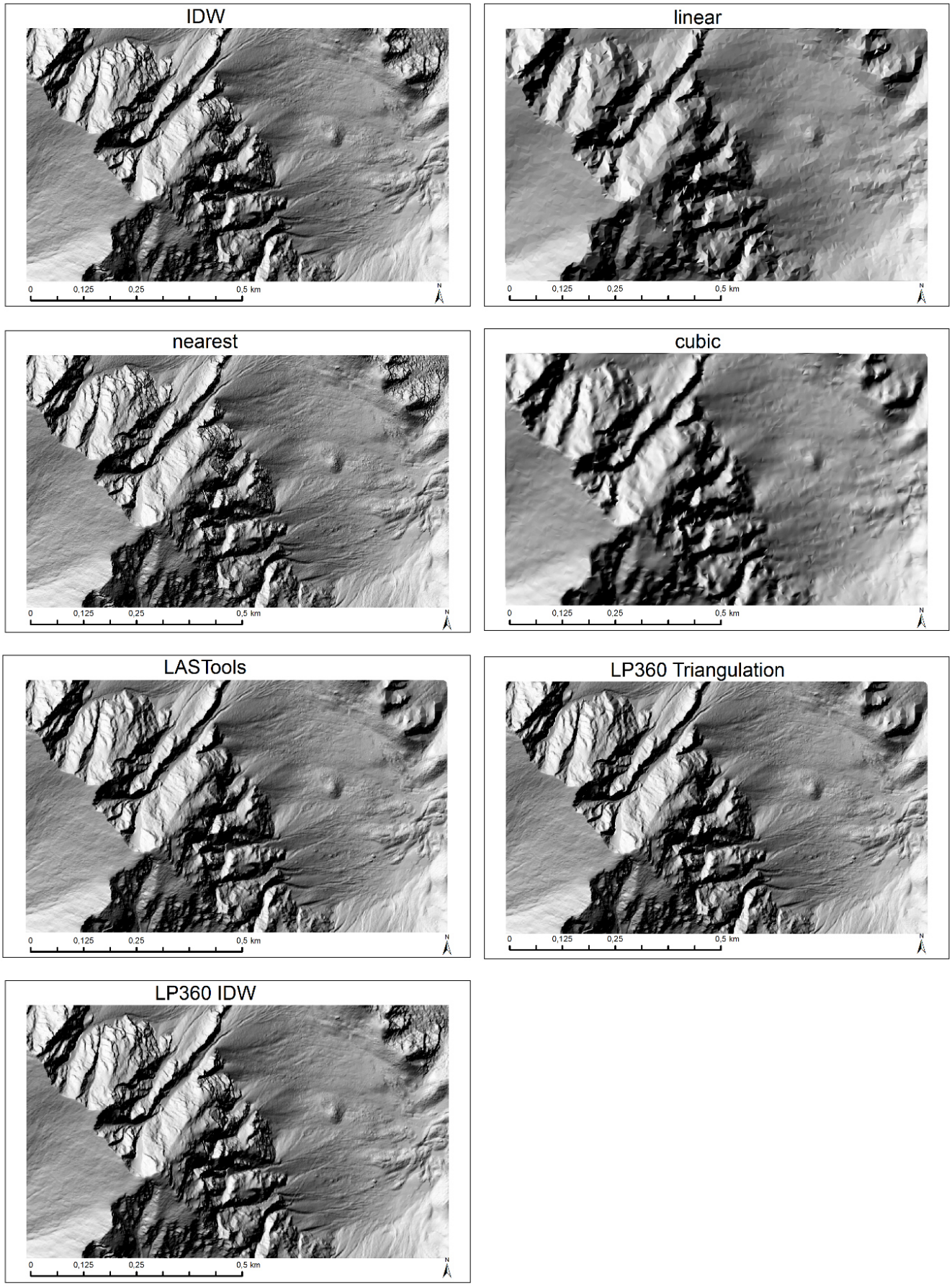

Abbildung 20: Ergebnisse der Interpolationen der LIDAR Daten für das Untersuchungsgebiet 1 (eigene Darstellung)

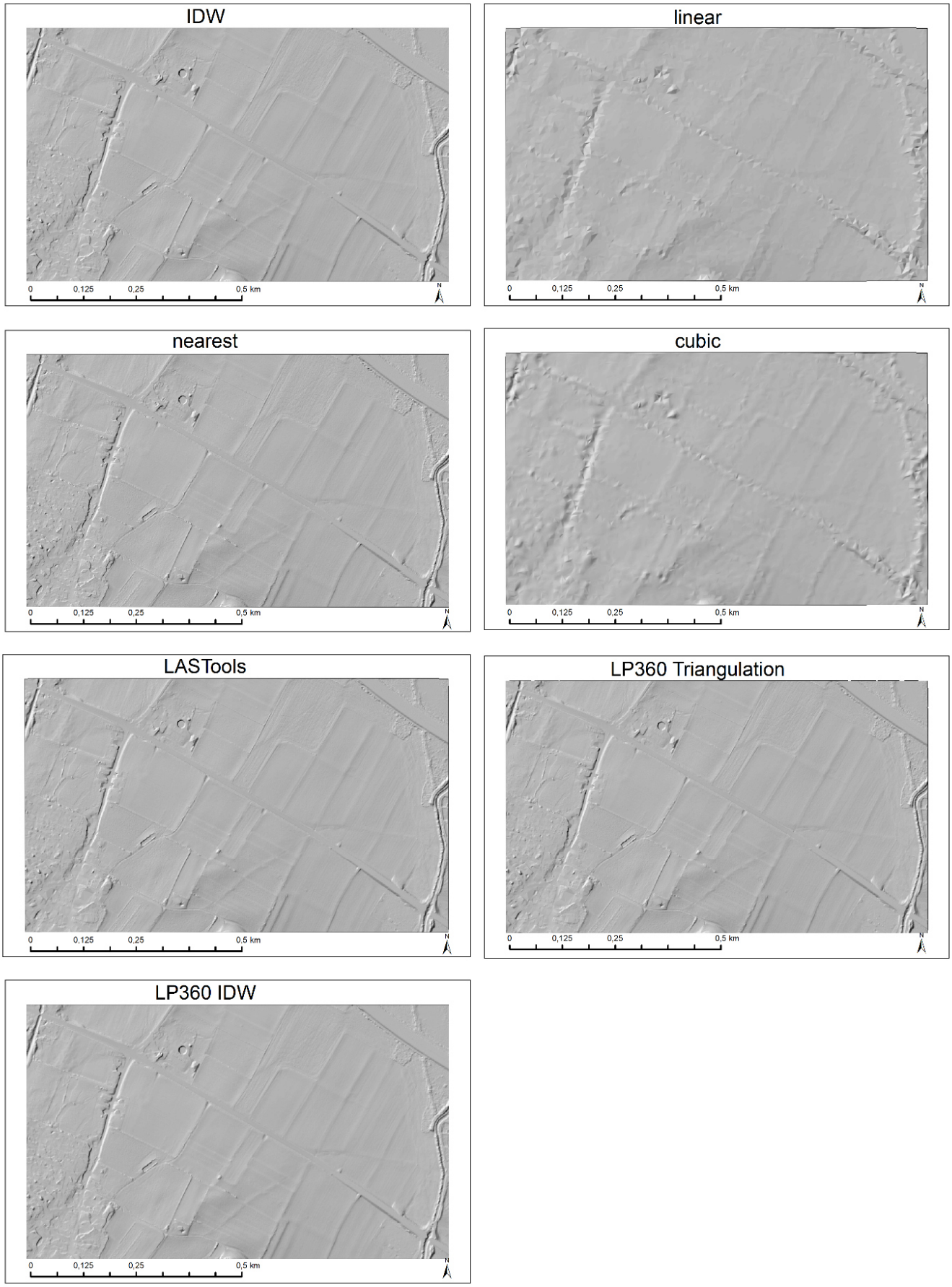

Abbildung 21: Ergebnisse der Interpolationen der LIDAR Daten für das Untersuchungsgebiet 2 (eigene Darstellung)

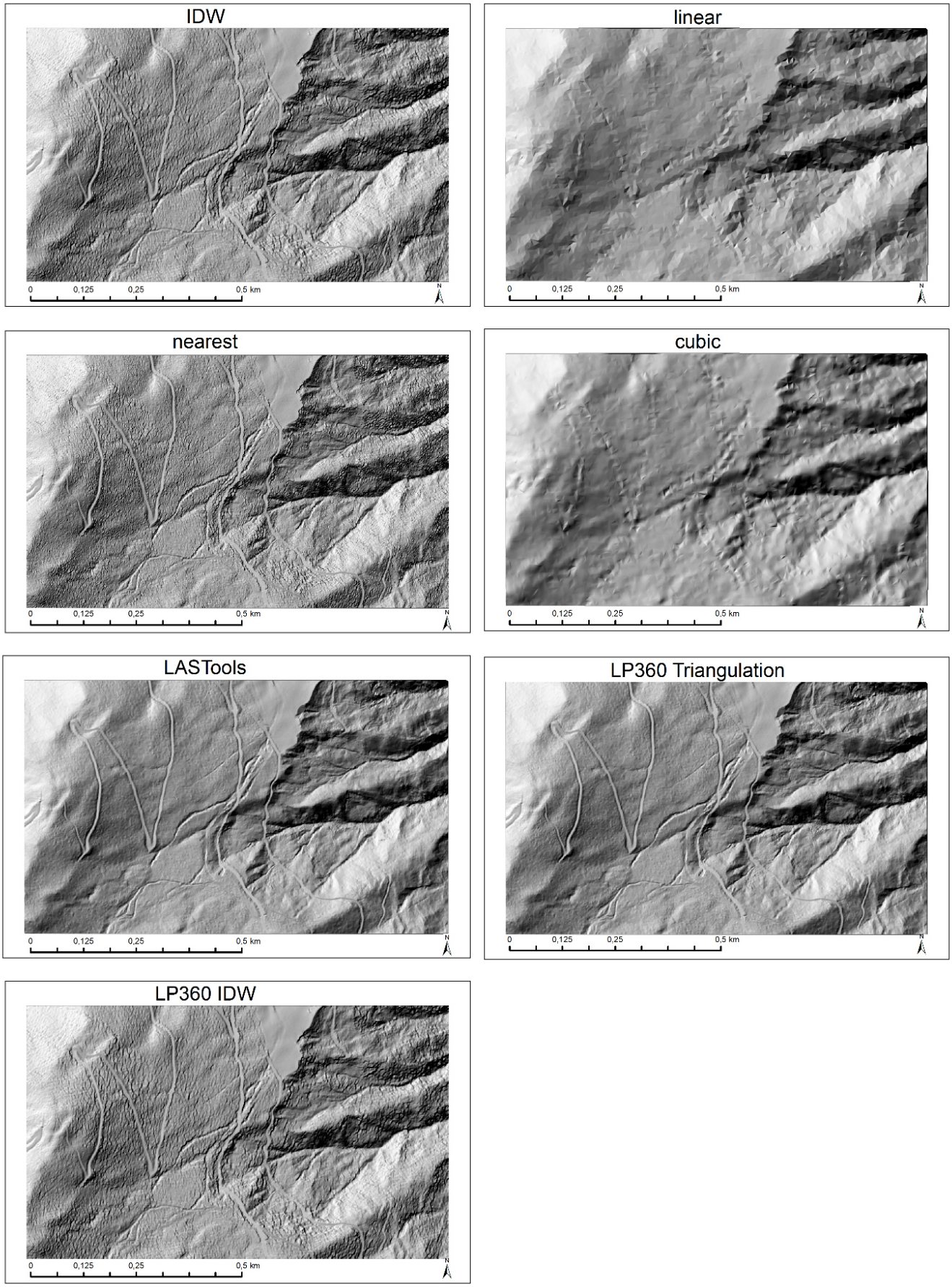

Abbildung 22: Ergebnisse der Interpolationen der LIDAR Daten für das Untersuchungsgebiet 3 (eigene Darstellung)

Ergebnisse für die DTM Validierung nach der Interpolation werden in Tabelle 10 präsentiert. Dabei wurde für die programmierte IDW Methode mit mehreren Parametereinstellungen ein Raster berechnet, um zu sehen, wie sich die Parameteränderung auf das Ergebnis auswirkt. Die Zahl neben der IDW Methode in Tabelle 10 bedeutet, wie viele Nachbarn für die Interpolation gesucht werden sollen. Für den Suchradius wurde bei allen die Standardeinstellung „15“ beibehalten. So wurde die programmierte IDW Methode mit der Nachbaranzahl „12“, „6“ und „20“ getestet. Weiterhin wurde für die Variante mit „12“ Nachbarn auch getestet, wie sich die Veränderung der Potenzzahl auf „3“ auf das Ergebnis auswirkt.

Für alle Interpolationsmethoden liegen die mittleren Fehler im unteren Zentimeterbereich. Der RMS Fehler reicht insgesamt von 0,04 m bis 1,4 m. Auch die mittleren absoluten Fehlerwerte der einzelnen Methoden sind sehr gleichmäßig. Alle drei Fehlerwerte variieren bei allen Werkzeugen mit der Reliefenergie. Auch in dieser Masterarbeit wird der Einfluss des Geländes auf die Interpolation sehr deutlich. Während das Untersuchungsgebiet 2 als ebene Fläche und mit den meisten Punkten bei allen Methoden die geringsten Fehlerwerte und auch die geringsten Bandbreiten zeigt, sind die angegebenen Fehlerwerte im Untersuchungsgebiet 1, welches die größte Reliefenergie besitzt, für alle Methoden am höchsten.

Im Vergleich zwischen den, im Rahmen der Masterarbeit, programmierten OpenSource Interpolationsmethoden (IDW und Multivariaten Methoden) besitzt die IDW Methode die geringsten Fehlerwerte. Dabei zeigen verschiedene Nachbarschaftsparameter für den IDW kaum nenneswerte Unterschiede in den Fehlerwerten. Auch die Nearest Neighbour Methode von Python zeigt ähnlich gute Ergebnisse. Die schlechtesten Ergebnisse liefern die lineare und die kubische Methode. Besonders die Maximalwerte der Fehlerwerte sind bei diesen beiden Methoden am höchsten. Für das UG 1 besitzt die OpenSource IDW Methode einen RMSE von 0,530 m, die lineare Methode einen RMSE von 1,477 m, die Nearest Neighbour Methode einen RMSE von 0,546 m und die kubische Methode einen RMSE von 1,278 m. Im Vergleich dazu sind die RMSE Werte für das UG 2 für den IDW bei 0,080 m, für die lineare Methode bei 0,196 m, für die Nearest Neighbour Methode bei 0,082 m und für die kubische Methode bei 0,186 m. Obwohl das Untersuchungsgebiet 3 durch die Waldbedeckung am wenigsten Punkte aufweist und auch große topographische Variabilität zeigt, die aber nicht ganz so stark wie beim Untersuchungsgebiet 1 ist, sind die Fehlerwerte in diesem Gebiet nur am

zweitschlechtesten, mit RMS Fehlern von 0,331 m für die IDW Methode, 0,655 m für die lineare Methode, 0,343 m für die nearest Methode und 0,631 m für die kubische Methode.
Im Vergleich des Opensource Tools zu den kommerziellen Tools LASTools und LP360 sind die Fehlerwerte etwas schlechter. Zwar werden die Fehlerwerte bei den kommerziellen Tools auch umso schlechter, je stärker die Reliefenergie wird, aber diese Tools zeigen gering bessere Werte. Zum Beispiel hat das LASTools beim Untersuchungsgebiet 1 einen RMSE von 0,166 m und die Trinagulationsmethode von LP360 einen RMSE von 0,167 m. Für das Untersuchungsgebiet 2 besitzen die kommerzielle Methoden einen RMSE von 0,042 m bis 0,048 m und für das UG 3 für LASTools und Triangulation 0,114 m und LP360 IDW 0,205 m.
Die Unterschiede zwischen den Fehlerwerten zu den OpenSource Tools werden mit steigender Reliefenergie größer. LASTools und LP360 besitzen jedoch bei allen drei Untersuchungsgebieten eine ähnlich hohe Bandbreite der maximalen Fehlerwerte, wie die OpenSource IDW und Nearest Neighbour Methode. Der Vergleich der RMS Fehler zwischen den einzelnen Methoden wird in Abbildung 23 und Abbildung 24 noch einmal visuell verdeutlicht. Weiterhin wird auch der Unterschied zwischen der OpenSource IDW Methode und der LP360 IDW Methode erkennbar. Der Unterschied könnte dabei durch die unterschiedliche Parametrisierung entstanden sein. Wie bereits in Kapitel 7.5.2 erläutert, wird beim programmierten OpenSource IDW nur die Anzahl der Nachbarn gesucht, die vom Benutzer angegeben werden. Auch wenn der Radius groß gewählt ist und noch mehr Nachbarn gefunden würden, wird nur die Nachbaranzahl berücksichtigt, die eingegeben wurde. Beim IDW vom LP360 wird die Minimum Anzahl der Nachbarn eingegeben, das heißt also, dass abhängig vom Radius noch mehr Nachbarn berücksichtigt werden.
Der Unterschied der mittleren absoluten Fehlern (MAE) zwischen den Tools in den einzelnen Untersuchungsgebieten wird in Abbildung 25 und Abbildung 26 dargestellt. Das OpenSource IDW zeigt ähnlich gute Ergebnisse im UG2 bezüglich der mittleren absoluten Fehler wie die kommerziellen Tools. Allerdings sind die mittleren absoluten Fehler für UG1 und UG2 für das OpenSource IDW Tool deutlich höher als für die kommerziellen Tools. Abbildung 27 und Abbildung 28 zeigen die Ergebnisse für den mittleren Fehler von allen Tools in den Untersuchungsgebieten.

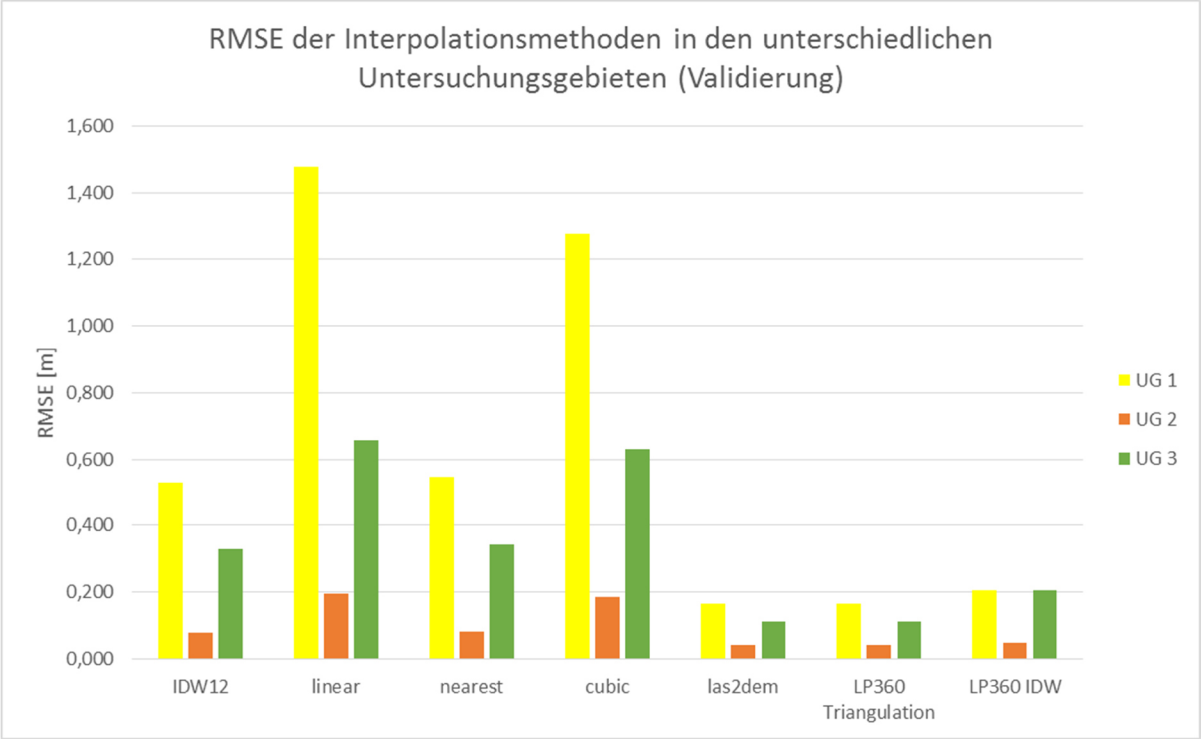

Abbildung 23: RMSE Fehler der Interpolationsmethoden in den unterschiedlichen Untersuchungsgebieten aus der Validierung (eigene Darstellung)

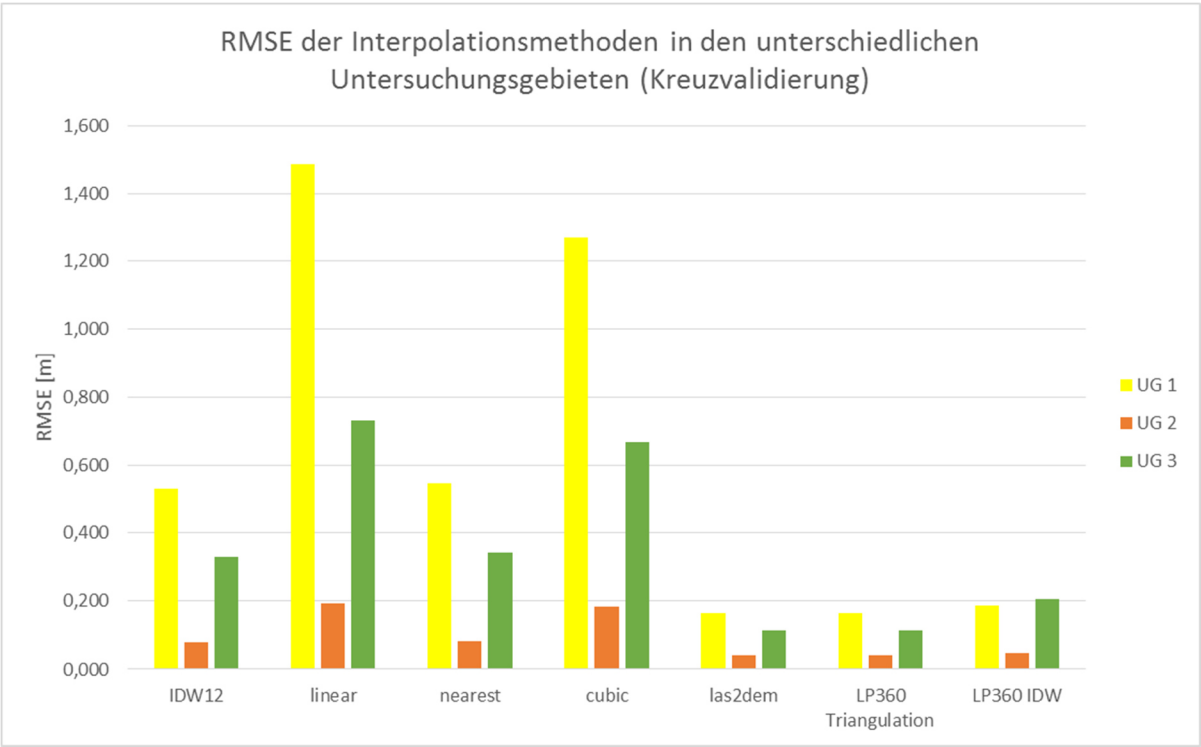

Abbildung 24: RMSE Fehler der Interpolationsmethoden in den unterschiedlichen Untersuchungsgebieten aus der Kreuzvalidierung (eigene Darstellung)

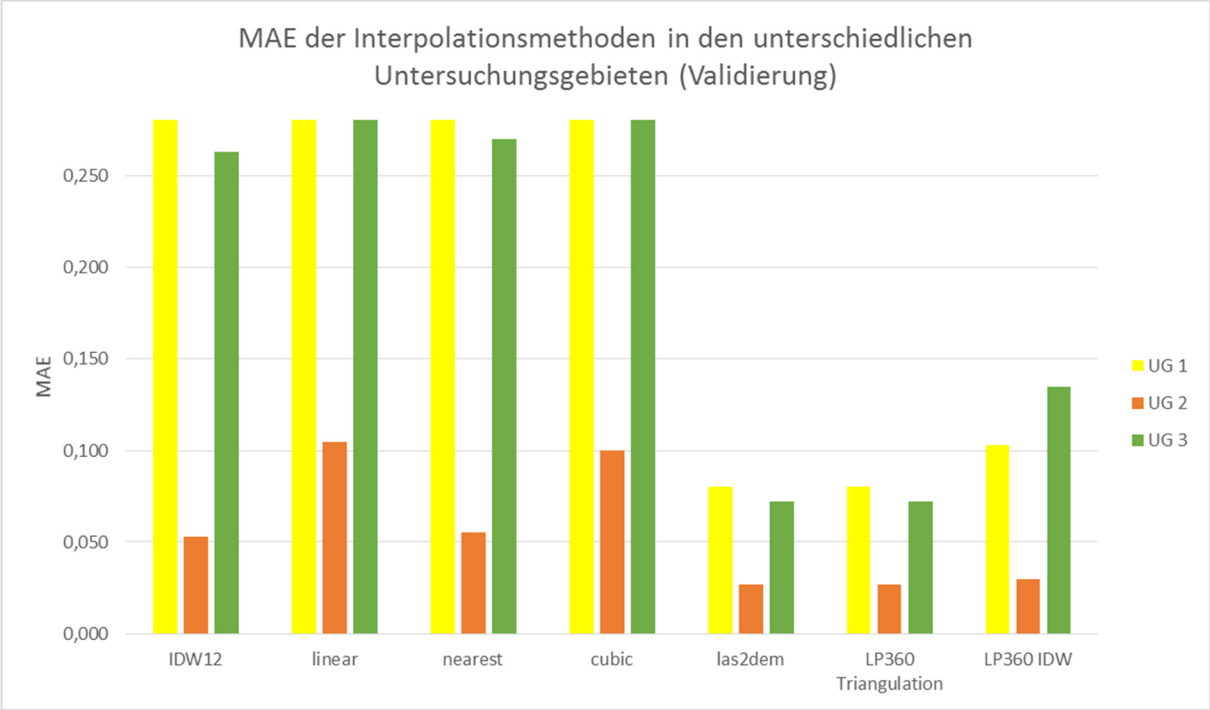

Abbildung 25: MA Fehler der Interpolationsmethoden in den unterschiedlichen Untersuchungsgebieten aus der Validierung (eigene Darstellung)

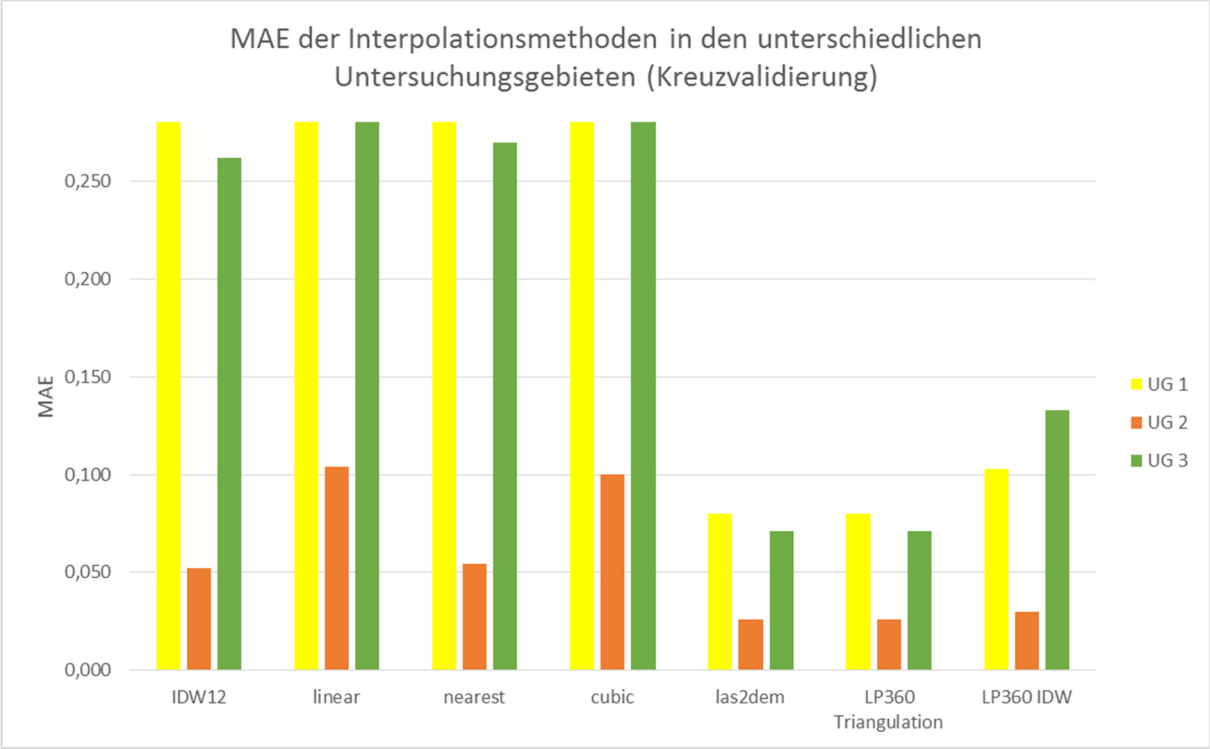

Abbildung 26: MA Fehler der Interpolationsmethoden in den unterschiedlichen Untersuchungsgebieten aus der Kreuzvalidierung (eigene Darstellung)

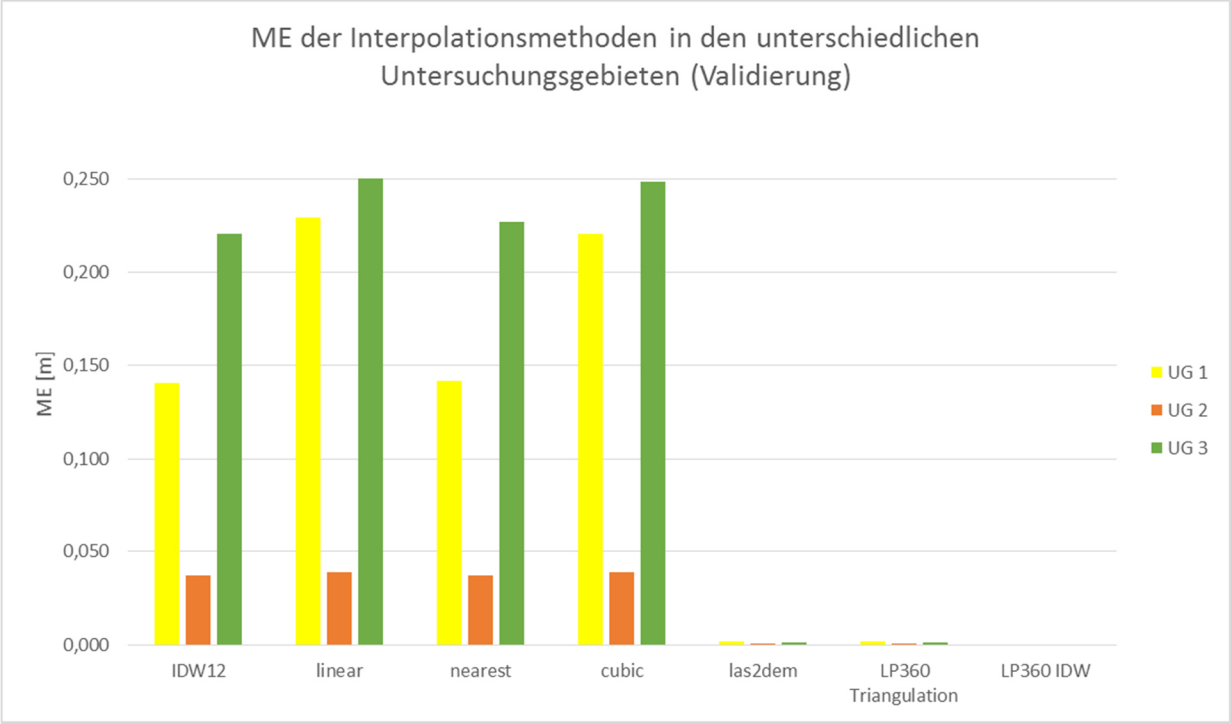

Abbildung 27: Mittlere Fehler der Interpolationsmethoden in den unterschiedlichen Untersuchungsgebieten aus der Validierung (eigene Darstellung)

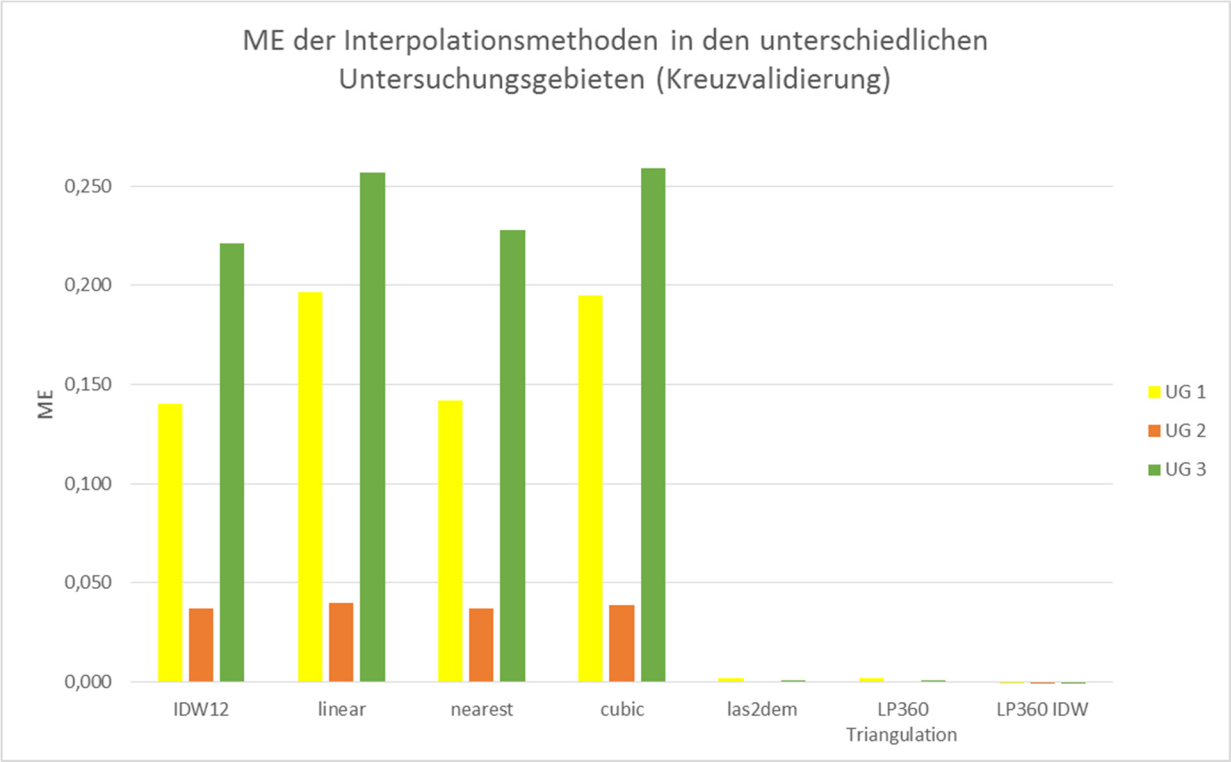

Abbildung 28: Mittlere Fehler der Interpolationsmethoden in den unterschiedlichen Untersuchungsgebieten aus der Kreuzvalidierung (eigene Darstellung)

Tabelle 10: Ergebnisse der einfachen Validierung und der 10-fach-Kreuzvalidierung durch verschiedene Tools (eigene Darstellung)

			Mean Error (m)	Min error (m)	Max error (m)	Range (m)	RMSE (m)	Mean absolut Error (m)	Mean Error (m)	Min error (m)	Max error (m)	Range (m)	RMSE (m)	Mean absolut Error (m)
UG 1	IDW	IDW12	0,141	-9,929	19,430	29,358	0,530	0,411	0,140	-10,346	19,741	30,087	0,532	0,411
		IDW12 Power 3	0,141	-9,755	20,348	30,103	0,530	0,411	0,141	-10,394	19,553	29,947	0,532	0,411
		IDW6	0,141	-9,552	20,934	30,486	0,530	0,411	0,141	-10,120	20,575	30,696	0,533	0,411
		IDW20	0,140	-10,537	18,856	29,393	0,530	0,411	0,140	-11,063	18,833	29,895	0,532	0,411
	Multivariat	linear	0,229	-31,690	30,764	62,454	1,477	0,835	0,196	-27,057	28,815	55,872	1,487	0,835
		nearest	0,142	-9,375	23,114	32,490	0,546	0,413	0,142	-10,146	24,879	35,025	0,549	0,413
		cubic	0,221	-28,869	26,514	55,383	1,278	0,736	0,194	-27,649	28,368	56,017	1,271	0,736
	LASTOOL	las2dem	0,002	-12,673	18,104	30,777	0,166	0,080	0,002	-12,298	14,137	26,436	0,166	0,080
	LP360	LP360 Triangulation	0,002	-13,082	17,422	30,504	0,167	0,080	0,002	-12,493	13,981	26,474	0,166	0,080
		LP360 IDW	-0,002	-15,233	17,882	33,115	0,207	0,103	-1,842	-10,832	17,623	26,634	0,187	0,103
UG 2	IDW	IDW12	0,037	-1,579	2,481	4,060	0,079	0,053	0,037	-1,693	2,245	3,938	0,079	0,052
		IDW12 Power 3	0,037	-1,589	2,526	4,115	0,080	0,053	0,037	-1,584	2,287	3,871	0,080	0,052
		IDW6	0,037	-1,567	2,573	4,141	0,080	0,053	0,037	-1,561	2,273	3,833	0,080	0,052
		IDW20	0,037	-1,584	2,366	3,950	0,079	0,053	0,037	-1,606	2,212	3,818	0,080	0,051
	Multivariat	linear	0,039	-8,990	2,731	11,721	0,196	0,105	0,040	-8,963	2,783	11,745	0,192	0,104
		nearest	0,037	-1,671	2,893	4,564	0,082	0,055	0,037	-1,669	2,410	4,079	0,083	0,054
		cubic	0,039	-8,070	2,904	10,974	0,186	0,100	0,039	-8,014	2,773	10,787	0,183	0,100
	LASTOOL	las2dem	0,000	1,058	-1,361	2,420	0,042	0,027	0,000	-1,188	1,274	2,462	0,042	0,026
	LP360	LP360 Triangulation	0,000	-2,519	1,058	3,577	0,042	0,027	0,000	-1,315	1,287	2,602	0,042	0,026
		LP360 IDW	0,000	-1,556	1,056	2,612	0,048	0,030	0,000	-1,391	1,510	2,901	0,048	0,030
UG 3	IDW	IDW12	0,221	-3,669	3,215	6,884	0,331	0,263	0,221	-3,721	3,239	6,960	0,330	0,262
		IDW12 Power 3	0,223	-3,546	3,121	6,667	0,329	0,263	0,224	-3,543	3,159	6,701	0,328	0,263
		IDW6	0,223	-3,521	3,107	6,628	0,329	0,263	0,224	-3,571	3,164	6,735	0,329	0,263
		IDW20	0,219	-3,849	3,087	6,936	0,334	0,260	0,219	-3,909	3,273	7,182	0,333	0,259
	Multivariat	linear	0,250	-14,097	11,150	25,247	0,655	0,449	0,257	-22,761	10,418	33,179	0,730	0,454
		nearest	0,227	-3,230	3,870	7,100	0,343	0,270	0,228	-3,490	3,611	7,101	0,343	0,270
		cubic	0,249	-14,412	12,727	27,139	0,631	0,424	0,259	-16,960	15,447	32,407	0,669	0,426
	LASTOOL	las2dem	0,002	-2,944	1,669	4,613	0,114	0,072	0,001	-2,870	1,992	4,862	0,113	0,071
	LP360	LP360 Triangulation	0,001	-2,943	1,669	4,613	0,114	0,072	0,001	-2,804	1,975	4,779	0,113	0,071
		LP360 IDW	-0,001	-3,833	2,540	6,373	0,205	0,135	-0,001	-4,121	2,902	7,023	0,205	0,133

Abbildung 29 bis Abbildung 33 zeigen Vergleiche der Werkzeuge in den Untersuchungsgebieten als Profildarstellung. Zum Vergleich der Werkzeuge wurde ein zufälliges Profil im Gelände gewählt. In den Abbildungen werden die Ergebnisse aus den Trainingsgebieten mit allen aufgenommenen ALS Bodenpunkten entlang des Profils verglichen.

Abbildung 29 bis Abbildung 31 zeigen die Ergebnisse der Interpolationswerkzeuge im Vergleich zu den ALS Punkten für das UG 1. Alle Werkzeuge zeigen deutlich Defizite bei starken Höhenunterschieden, wobei in diesem Profil das LASTools und das LP360 nur etwas mehr von den ALS Punkten abweichen als das OpenSource IDW Tool. Bei geringeren Höhenunterschieden zeigen jedoch alle Tools ähnlich geringe Abweichungen zu den ALS Punkten. Der Verlauf aller Werkzeuge ist sehr ähnlich. Der OpenSource IDW und der Nearest Neighbour Algorithmus zeigen dabei sehr angenäherte Verläufe zueinander (Abbildung 29), sowie die Ergebnisse von LASTools und LP360 (Abbildung 30 und Abbildung 31).

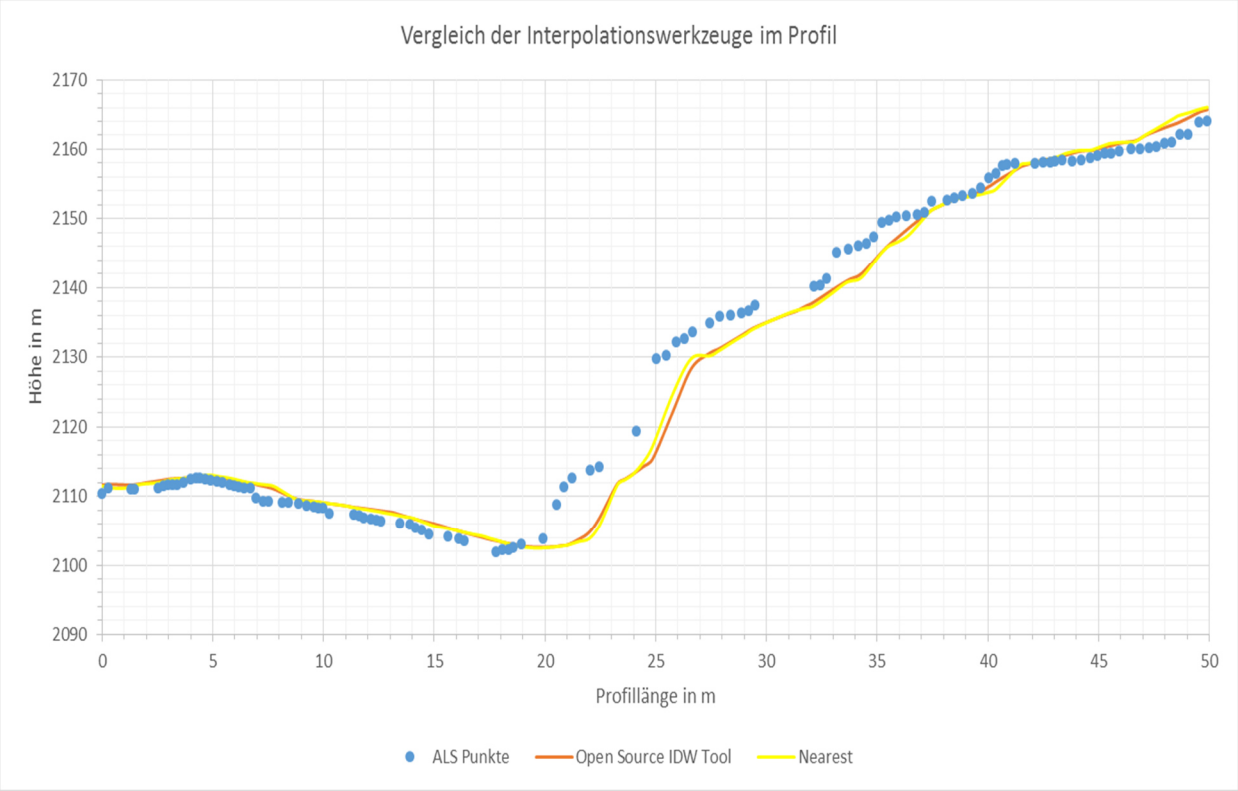

Abbildung 29: Vergleich der Ergebnisse für das Untersuchungsgebiet 1 in einem zufällig ausgewählten Profil zwischen dem programmierten IDW und Nearest Neighbour (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

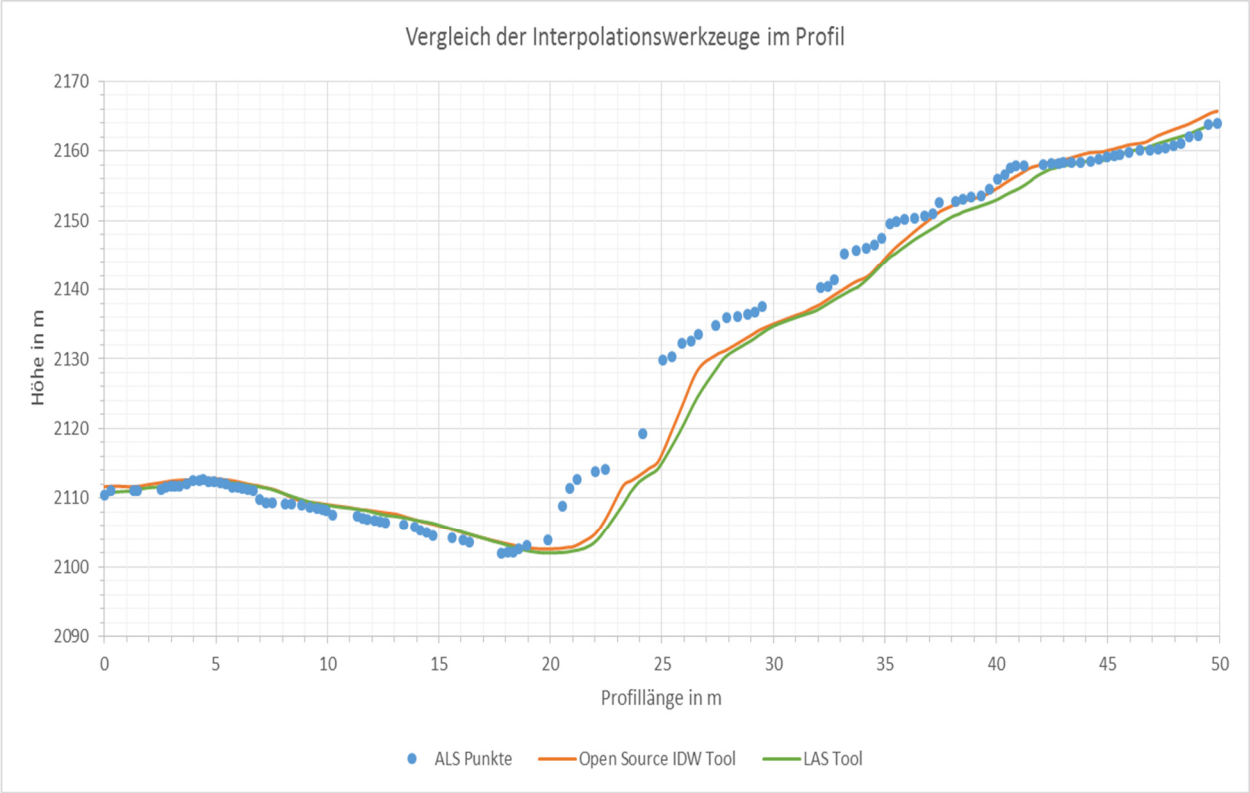

Abbildung 30: Vergleich der Ergebnisse für das Untersuchungsgebiet 1 in einem zufällig ausgewählten Profil zwischen dem programmierten IDW und LASTools (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

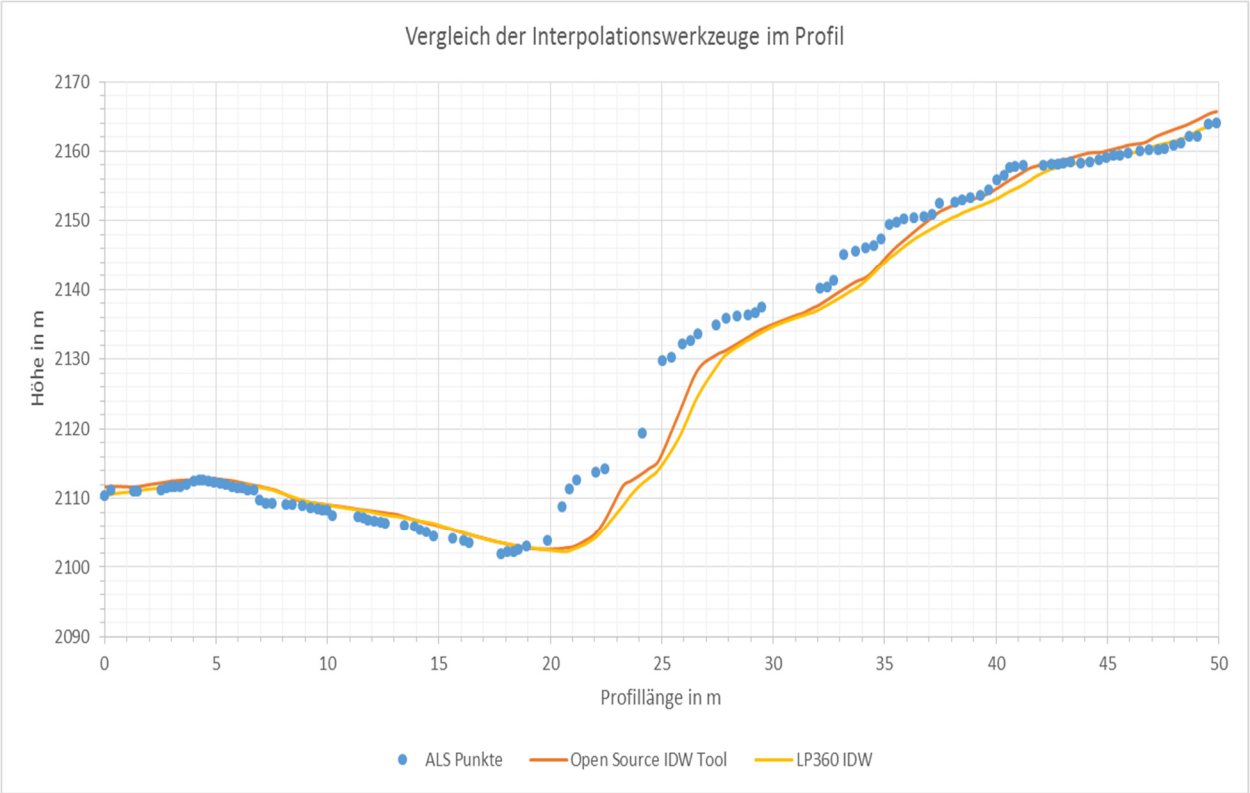

Abbildung 31: Vergleich der Ergebnisse für das Untersuchungsgebiet 1 in einen zufällig ausgewähltem Profil zwischen dem programmierten IDW und LP360 IDW (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

Abbildung 32 und Abbildung 33 zeigen die Ergebnisse der Werkzeuge in einem Profil für die Untersuchungsgebiete 2 und 3. Für das flache Untersuchungsgebiet 2 gibt es kaum Unterschiede im Verlauf zwischen den Werkzeugen. Im flachen Gebiet zeigen die Werkzeuge sehr gute Ergebnisse im Vergleich zu den ALS Daten. Aber auch in diesem Profil wird deutlich, wie sehr bei Veränderungen im Gelände die Interpolationen von den ALS Daten abweichen. Dadurch werden selbst auch im flachen Gebiet Veränderungen im Gelände, wie hier ein Graben im Profil, nicht exakt wiedergegeben. In Abbildung 33 ist der Unterschied im Profil durch die Höhenänderung der Interpolationen zu den ALS Punkten ebenfalls sehr deutlich. Die Geländeprofile für das Untersuchungsgebiet 3 werden von allen Tools am schlechtesten von allen drei Beispielprofilen dargestellt. Grund dafür kann die geringe Bodenpunktdichte durch die Waldabdeckung sein.

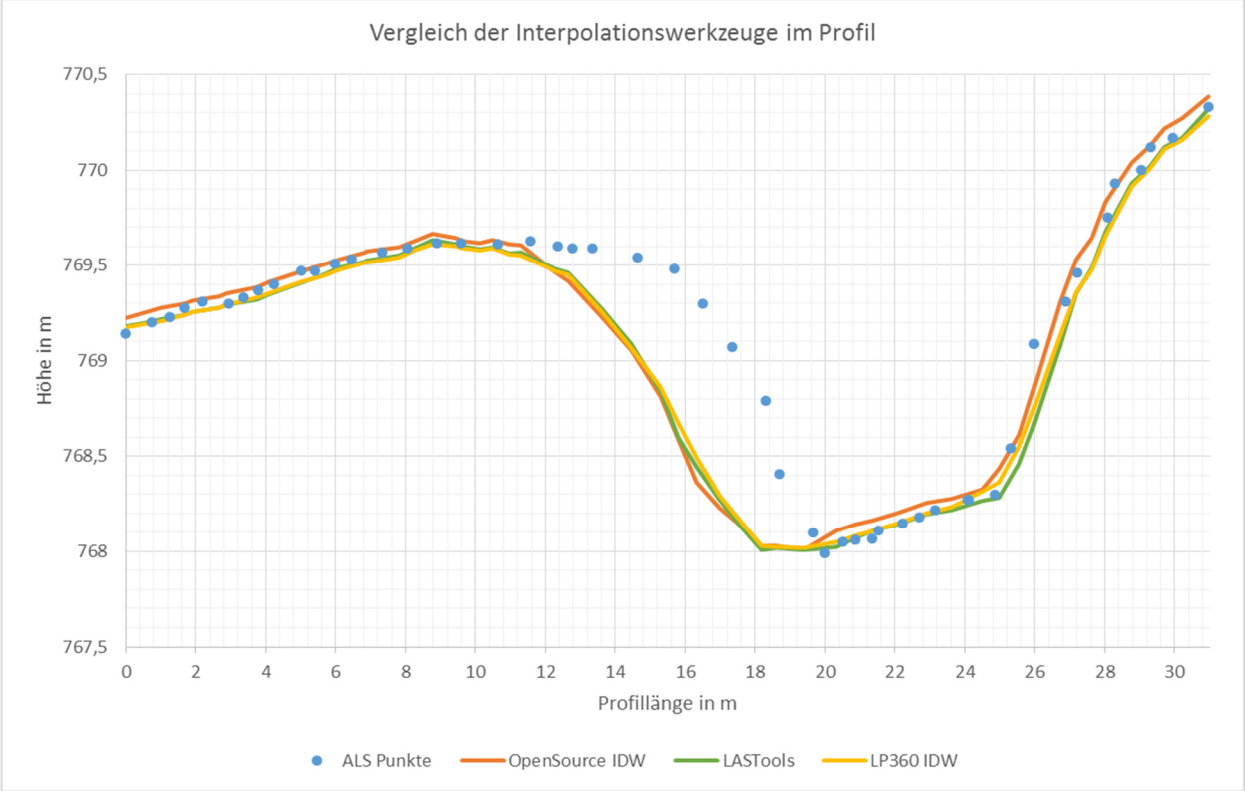

Abbildung 32: Vergleich der Ergebnisse für das Untersuchungsgebiet 2 in einem zufällig ausgewählten Profil zwischen den programmierten IDW, LASTools und LP360 IDW (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

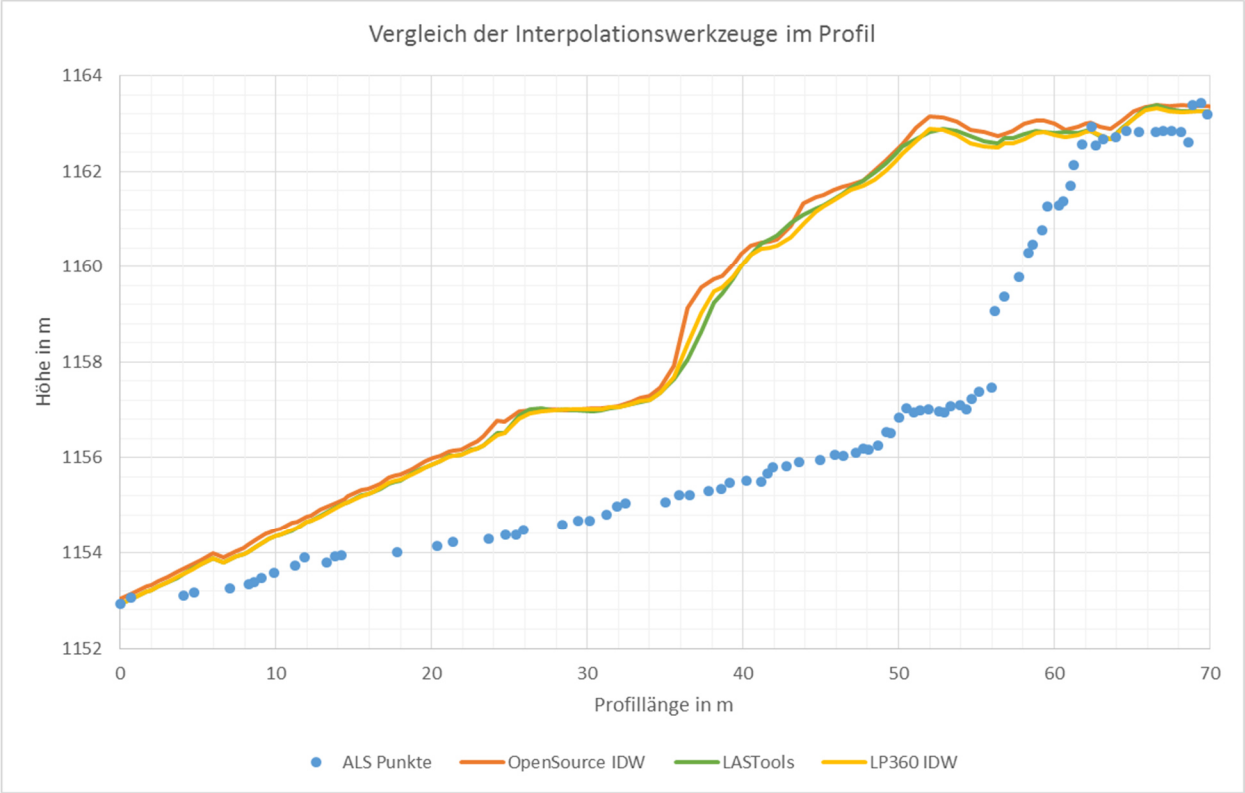

Abbildung 33: Vergleich der Ergebnisse für das Untersuchungsgebiet 3 in einem zufällig ausgewählten Profil zwischen den programmierten IDW, LASTools und LP360 IDW (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

Um zu klären, wo die Unterschiede zwischen den einzelnen Tools liegen, wurden die Ergebnisse in Abbildung 34 bis Abbildung 38 analysiert. Dabei wurden die Differenzen aus den Ergebnissen der Interpolationen mit dem OpenSource IDW, LASTools und LP360 IDW gebildet. Positive Werte zeigen, wo der OpenSource IDW die Geländehöhen höher als die anderen Tools geschätzt hat. Negative Werte zeigen, wo dieser das Gelände niedriger als die beiden lizenzpflichtigen Tools berechnet hat. Die Differenzbilder des jeweiligen Untersuchungsgebietes zeigen keine Unterschiede zueinander. Stattdessen zeigen diese die Gemeinsamkeiten, dass die Unterschiede zwischen den Tools an den selben Stellen im Gelände auftreten. Die mittleren Abbildungen zwischen den Differenzbildern sollen verdeutlichen, an welchen Stellen die größten Abweichungen vorkommen. Aus diesen Abbildungen wird ersichtlich, dass die größten Differenzen dort vorkommen, wo es kleinere Veränderungen im Gelände gibt, wie im UG2 und auch steile Neigungen, wie in UG1 und UG2. Das Differenzmuster entspricht sehr dem Neigungsmuster. Weiterhin wird auch aus diesen mittleren Abbildungen erkennbar, dass die größten Abweichungen zwischen den Tools bei geringer LIDAR Punktdichte auftreten. Abbildung 34 zeigt die Differenzmodelle der Ergebnisse

für das Untersuchungsgebiet 1. In diesem Gebiet zeigen die Tools teilweise extrem hohe Unterschiede von bis zu 46 m bzw. 60 m. Wie aus der mittleren Abbildung erkennbar, treten diese Differenzen bei extrem komplexem Gelände auf, wo Hangneigungen teilweise bis zu 90° betragen. Weiterhin wird aus den mittleren Bildern erkennbar, dass an den Stellen, wo diese Unterschiede auftreten, es keine bis kaum ALS Bodenpunktabdeckung im Gelände gibt. Durch das Fehlen der gemessenen Punkte und durch starke Neigungen und komplexe Gelände berechnen die Tools das Gelände sehr unterschiedlich , wodurch diese hohen Differenzen auftauchen. Um diese hohen Unterschiede besser analysieren zu können, wurden in beiden Auswahlfenstern in Abbildung 34 ein weiteres Mal zwei Profile gelegt. Ergebnisse sind in Abbildung 35 und Abbildung 36 dargestellt. Erkennbar ist, dass dort abschnittsweise keine ALS Daten vorhanden sind und sich das Gelände sehr stark ändert. Dabei verlaufen alle Geländehöhen der Tools bei sehr hoher ALS Dichte sehr ähnlich. Alle Tools haben jedoch Probleme, wenn es zu einem extremen Geländesprung kommt, wie einer senkrechten oder fast senkrechten Wand. In diesem Fall rechnet das LASTools fast am schlechtesten, denn es schätzt eine viel geringere Neigung ein und flacht die Steigung sehr ab. Das OpenSource IDW und das LP360 IDW verhalten sich fast ähnlich und können eine sehr starke Neigung berechnen, die jedoch nicht ganz auf die ALS Bodenpunkte passt und im Gelände versetzt berechnet wird. Weiterhin ist es sehr schwer einzuschätzen, wie das Gelände bei fehlenden ALS Bodenpunkten tatsächlich verläuft. Deshalb kann keine Aussage getroffen werden, welches Tool sich dem Gelände am besten annähert.

Abbildung 37 zeigt die Differenzmodelle für das Untersuchungsgebiet 2. Hier erweist sich, dass die Tools im flachen Gebiet das Gelände sehr ähnlich berechnen. Besonders ebene Flächen zeigen nur sehr geringe Abweichungen bis zu maximal 0,5 m Unterschiede zwischen den Tools. Allerdings kommt es zu Unterschieden bis zu 3 m, wo es entweder zu kleinen und plötzlichen Geländeänderungen kommt, wie zum Beispiel Gräben, oder wo es keine ALS Bodenpunkte gibt, wo zum Beispiel Objekte, wie Häuser, stehen. Die Flächen mit Objektpunkten und ohne Bodenpunkte werden also auch ganz unterschiedlich von den Tools interpoliert.

Abbildung 38 stellt die Ergebnisse der Tools im Untersuchungsgebiet 3 gegenüber. Dabei gibt es zwischen dem OpenSource IDW größere Unterschiede mit LASTools als mit LP360. Durch die starke Waldabdeckung gibt es hier auch wieder mehrere Bereiche, wo ALS Bodenpunkte fehlen bzw. eine geringe Dichte haben und weiters ein rauhes Gelände vorherrscht. Besonders an diesen Stellen weichen die Tools voneinander am meisten ab.

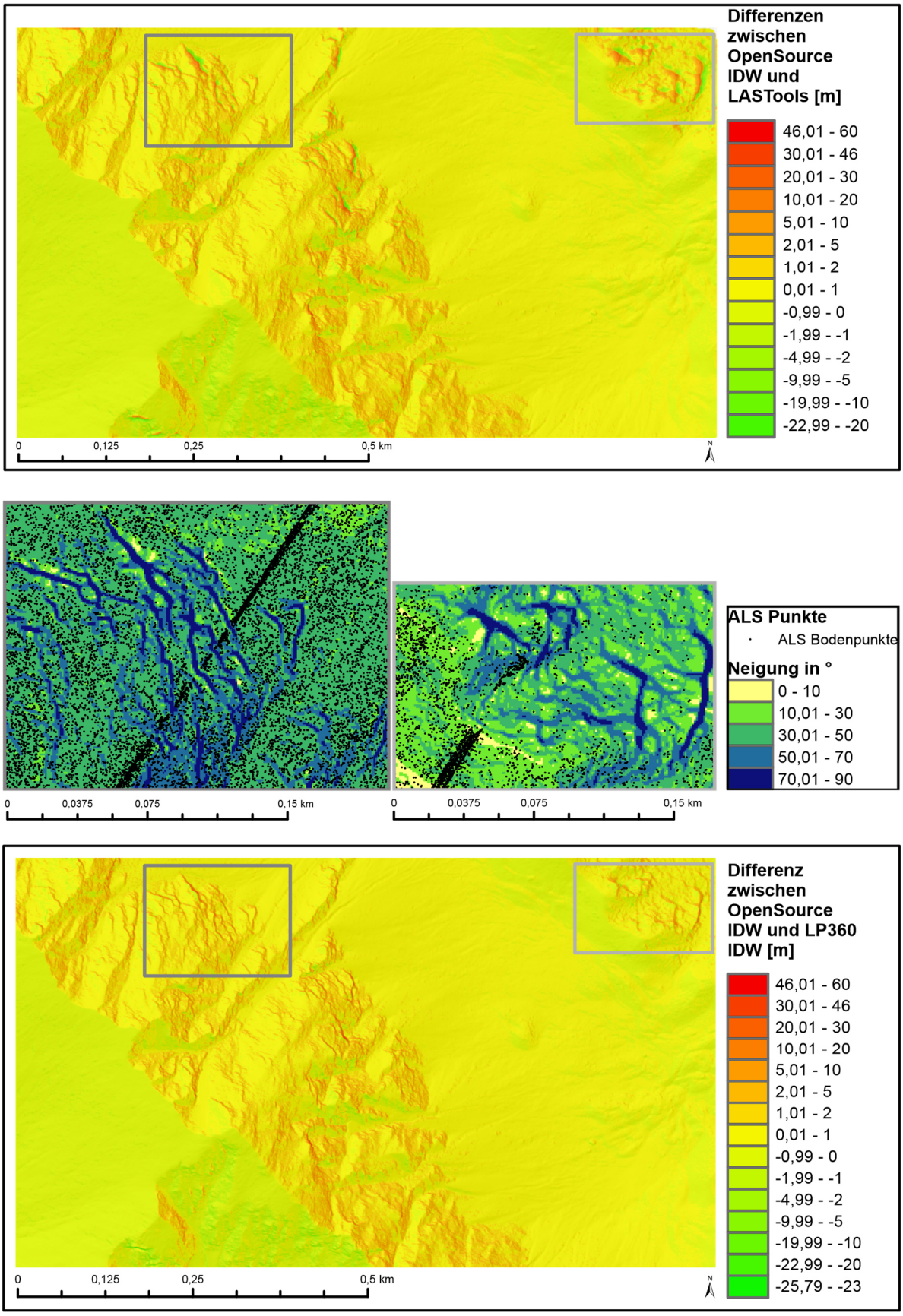

Abbildung 34: Differenzmodell aus dem OpenSource IDW und LASTools (oben) und dem OpenSource IDW und LP360 IDW (unten) für das Untersuchungsgebiet 1. In der Mitte: zwei ausgewählte Bereiche mit starken Unterschieden zwischen den Tools mit Neigung des Geländes und mit ausgedünnten ALS Punkten (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

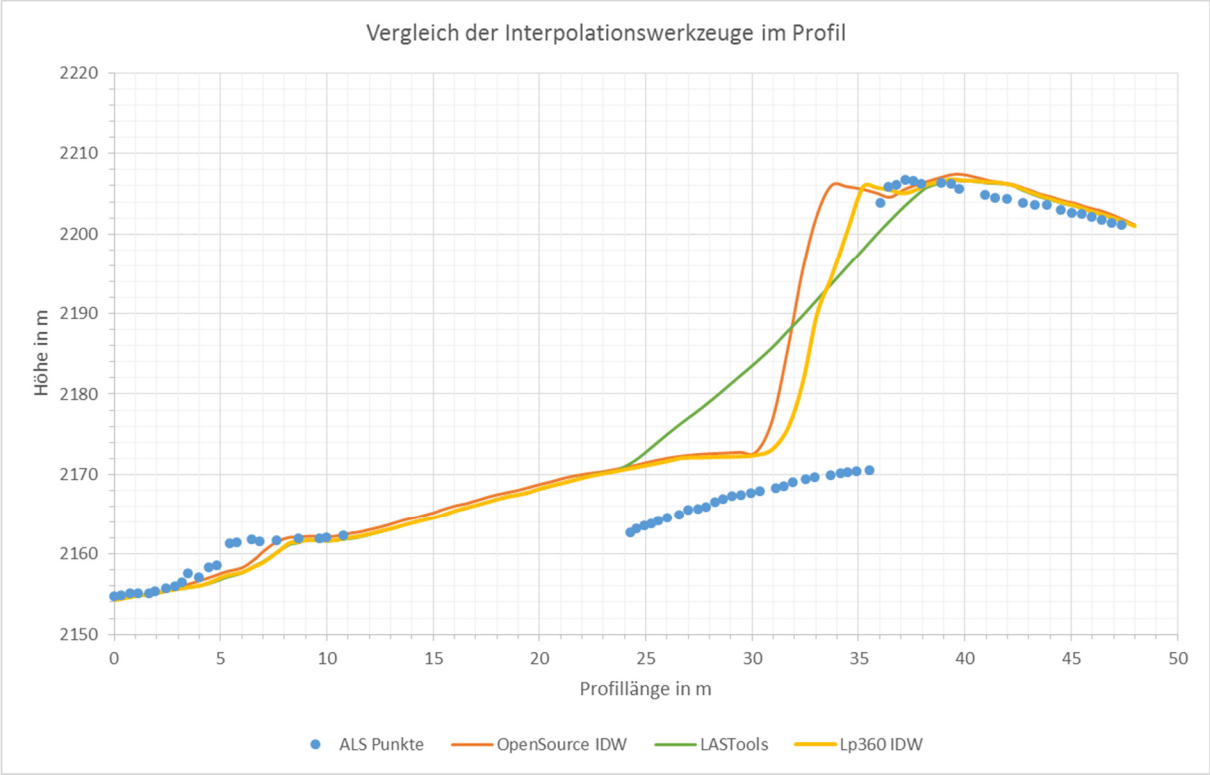

Abbildung 35: Vergleich der Ergebnisse für Untersuchungsgebiet 1 zwischen den programmierten IDW, LASTools und LP360 IDW in einem ausgewähltem Profil mit starken Unterschieden der Ergebnisse 1 (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

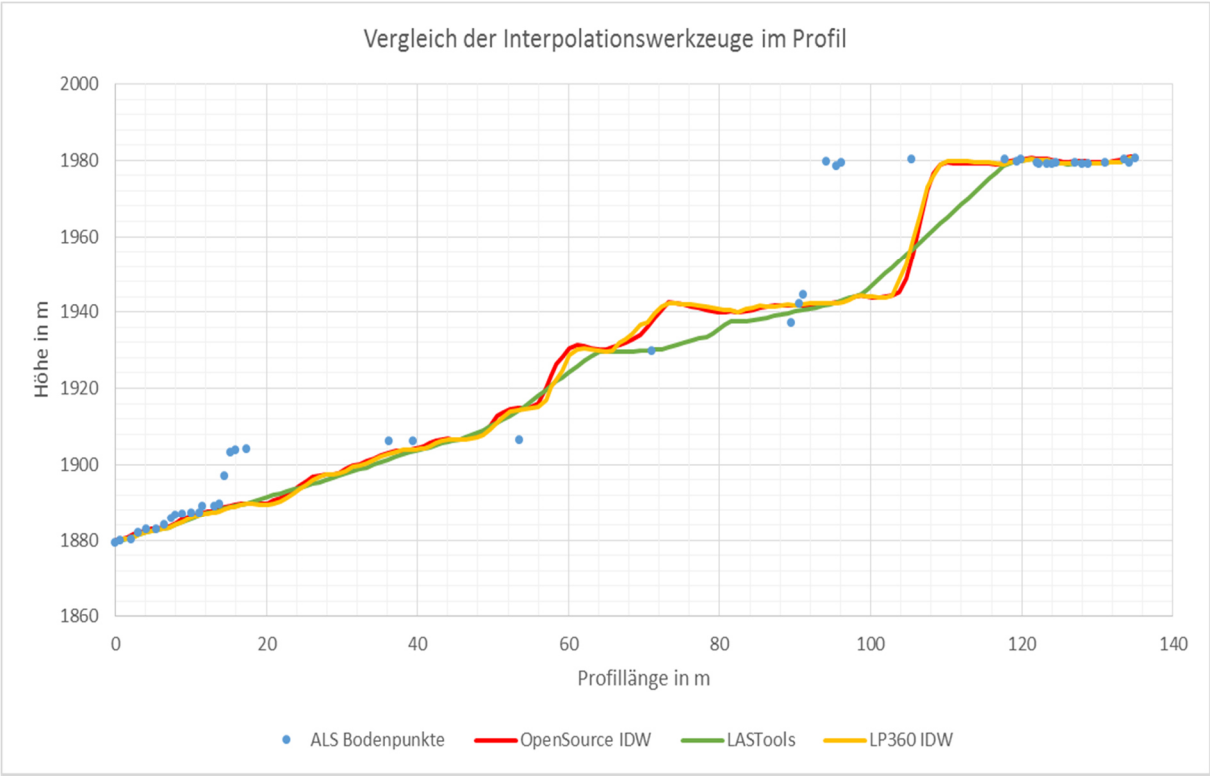

Abbildung 36: Vergleich der Ergebnisse für Untersuchungsgebiet 1 zwischen den programmierten IDW, LASTools und LP360 IDW in einem ausgewähltem Profil mit starken Unterschieden der Ergebnisse 2 (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

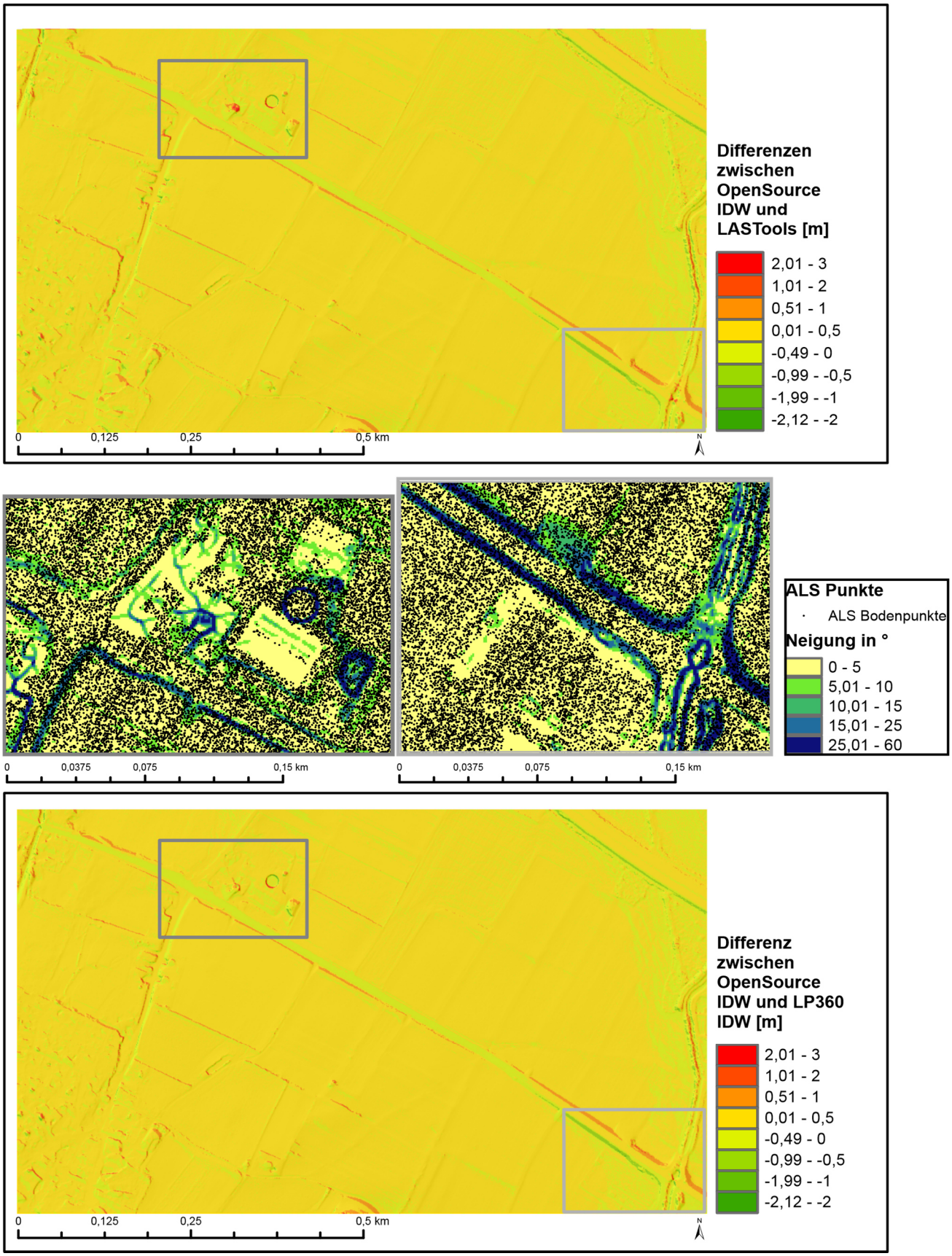

Abbildung 37: Differenzmodell aus dem OpenSource IDW und LASTools (oben) und dem OpenSource IDW und LP360 IDW (unten) für das Untersuchungsgebiet 2. In der Mitte: zwei ausgewählte Bereiche mit starken Unterschieden zwischen den Tools mit Neigung des Geländes und mit ausgedünnten ALS Punkten (Daten: ALS Daten des Landes Steiermark, eigene Darstellung)

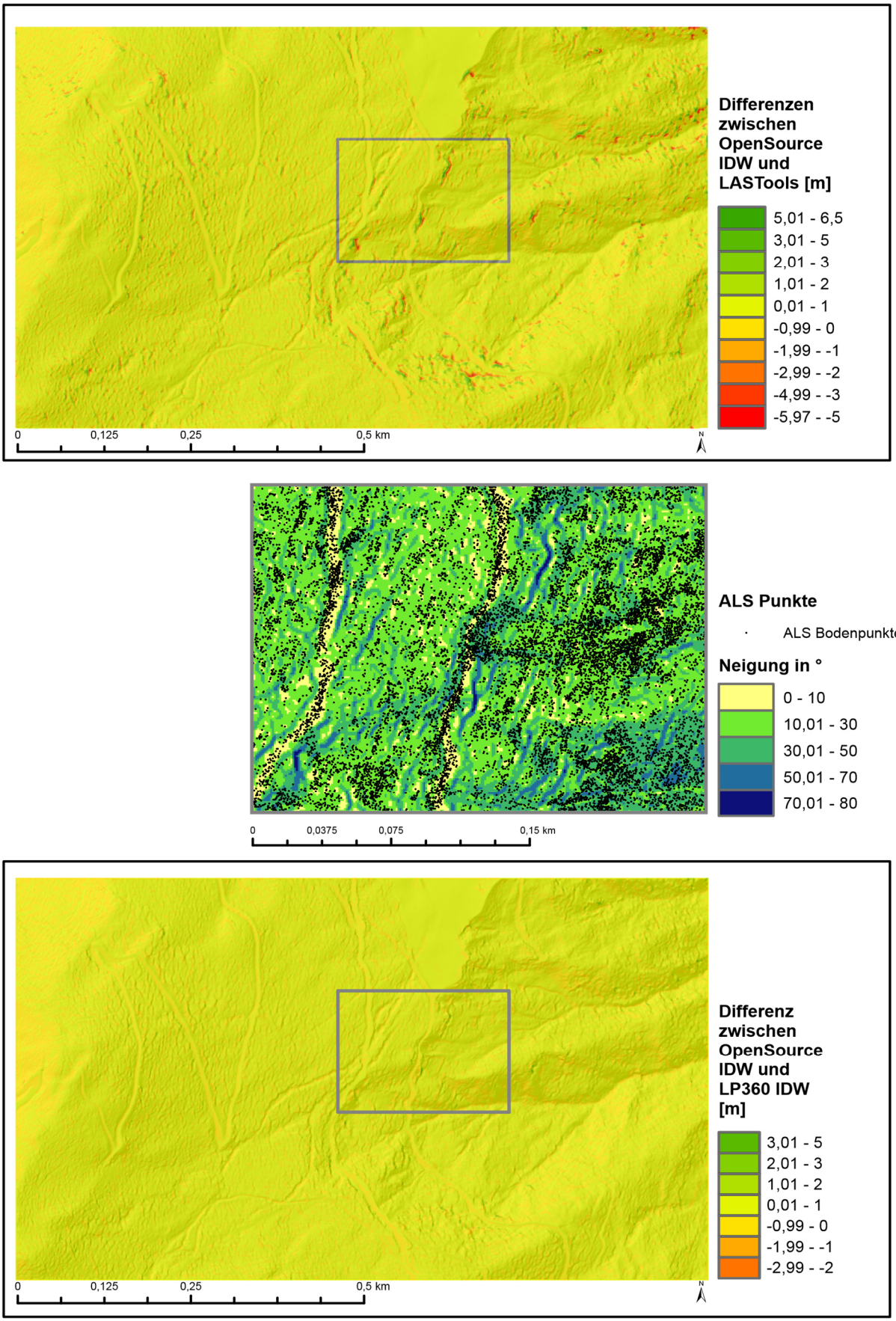

Abbildung 38: Differenzmodell aus dem OpenSource IDW und LASTools (oben) und dem OpenSource IDW und LP360 IDW (unten) für das Untersuchungsgebiet 3. In der Mitte: zwei ausgewählte Bereiche mit starken Unterschieden zwischen den Tools mit Neigung des Geländes und mit ausgedünnten ALS Punkten (Daten: ALS Daten des Landes Steiermark, eigene Darstellung).

10 Fallbeispiel Murenabgang in Sankt Lorenzen im Paltental

Eine Beispielanwendung von Interpolationswerkzeugen ist die Evaluierung der Geländeänderung nach einer Mure, wovon auch Sankt Lorenzen betroffen war. Am 21.07.2012 kam es zu einem Murenabgang, der den Ortskern von Sankt Lorenzen infolge heftigen flächendeckenden Starkregens verschüttete. Das Paltental war in diesem Sommer nach heftigen Unwettern schon mehrfach von Murenereignissen betroffen gewesen.
Das Land Steiermark besitzt bereits bestehende Geländemodelle von 2011 und 2012, die von einer externen Firma aus ALS Daten generiert wurden. Diese bereits bestehenden Modelle sollen mit den digitalen Geländemodellen, die aus den OpenSorce Tool mit der IDW Methode berechnet wurden, verglichen werden. Abbildung 39 zeigt den Überblick über die Auswahl des Ausschnittes für den Modellvergleich und die Mure im Ortskern von Sankt Lorenzen.

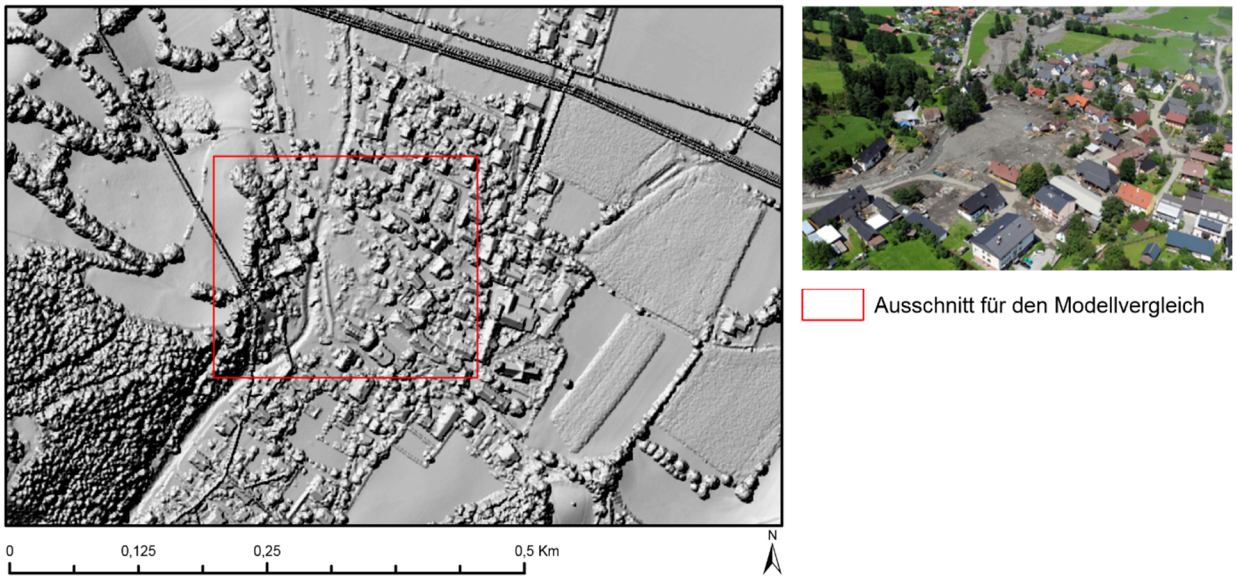

Abbildung 39: Links: Ort Sankt Lorenzen im Paltental als DSM mit Wahl des Ausschnittes zum Modellvergleich (Daten: DSM des Landes Steiermark). Rechts: Bild des Murenereignisses in Sankt Lorenzen im Paltental (DIE PRESSE, 2012)

Abbildung 40 präsentiert die Ergebnisse aus den Differenzbildungen der Jahre 2011 und 2012 beider Modelle. Das bereits bestehende Geländemodell und das Geländemodell, das aus der OpenSource IDW Methode generiert wurde, stimmen optisch in ihren Ergebnissen weitgehend überein, wobei das bereits bestehende Geländemodell im Gegensatz zum anderen auf den Berghängen noch einen Massenzuwachs berechnet hat. Jedoch stimmen beide Modelle im flachen Ortsgebiet im Großteil in den Klassen überein. Für eine genauere Quantifizierung der Ergebnisse wurden zusätzlich die ALS Bodenpunkte von 2011 und 2012

auf identische Punkte untersucht, um daraus ebenfalls einen punktuellen Differenzwert zum Vergleich auf die Genauigkeit ziehen zu können. An diesen ALS Differenzpunkten wurden diejenigen Punkte heraus gesucht, wo beide Modelle im Massenzuwachs über 0,2 m voneinander abweichen, um zu verdeutlichen, wo die Unterschiede beider Modelle liegen. Dazu sind in Tabelle 10 die genauen Ergebnisse der Differenzen aus den Jahren 2011 und 2012 beider Modelle sowie die ALS Differenzen an diesen Punkt ausgewertet. (Hinweis: An den Berghängen, wo es optische Unterschiede gibt, gibt es keine identischen ALS Punkte von 2011 und 2012. Daher wurden diese Stellen in der weiteren Auswertung nicht berücksichtigt.) Die Abweichung der Ergebnisse der Differenzmodelle zueinander und zu den tatsächlich gemessenen ALS Differenzen liegen im Zentimeterbereich. Die Mittelwerte der Abweichungen bei den Modelldifferenzen an diesen sechzehn Punkten zu den ALS Differenzen sind jeweils identisch mit 0,31 m. Der RMSE für die Abweichung des IDW Differenzmodells an diesen Punkten zu den ALS Differenzen beträgt 0,21 m, wohingegen der RMSE für die Abweichung der bereits bestehenden Geländemodelldifferenzen zu den ALS Differenzen 0,34 m beträgt. Im Mittel weichen die beiden Modelle an diesen sechzehn Punkten 0,32 m voneinander ab.

Zusätzlich wurden an insgesamt 800 Punkten in diesem Ausschnitt die Differenzwerte beider Geländemodelle zu den Differenzwerten der ALS Bodenpunkte evaluiert. Es ergeben sich 800 Vergleichspunkte, weil an diesen Punkten die ALS Punkte von 2011 und 2012 identisch sind. Ergebnisse dieser Evaluierung befinden sich in Tabelle 12. Die Abweichung der Modelle zu den ALS Differenzpunkten liegt auch in diesen 800 Punkten im Zentimeterbereich. Wobei auch hier in beiden Modellen die Abweichungen zu den ALS Differenzen im Mittel identisch sind und nur 0,001 m betragen. Der RMSE für die bereits bestehenden Modelle beträgt 0,09 m, wohingegen der RMSE für die IDW Methode um 0,03 m besser ist.

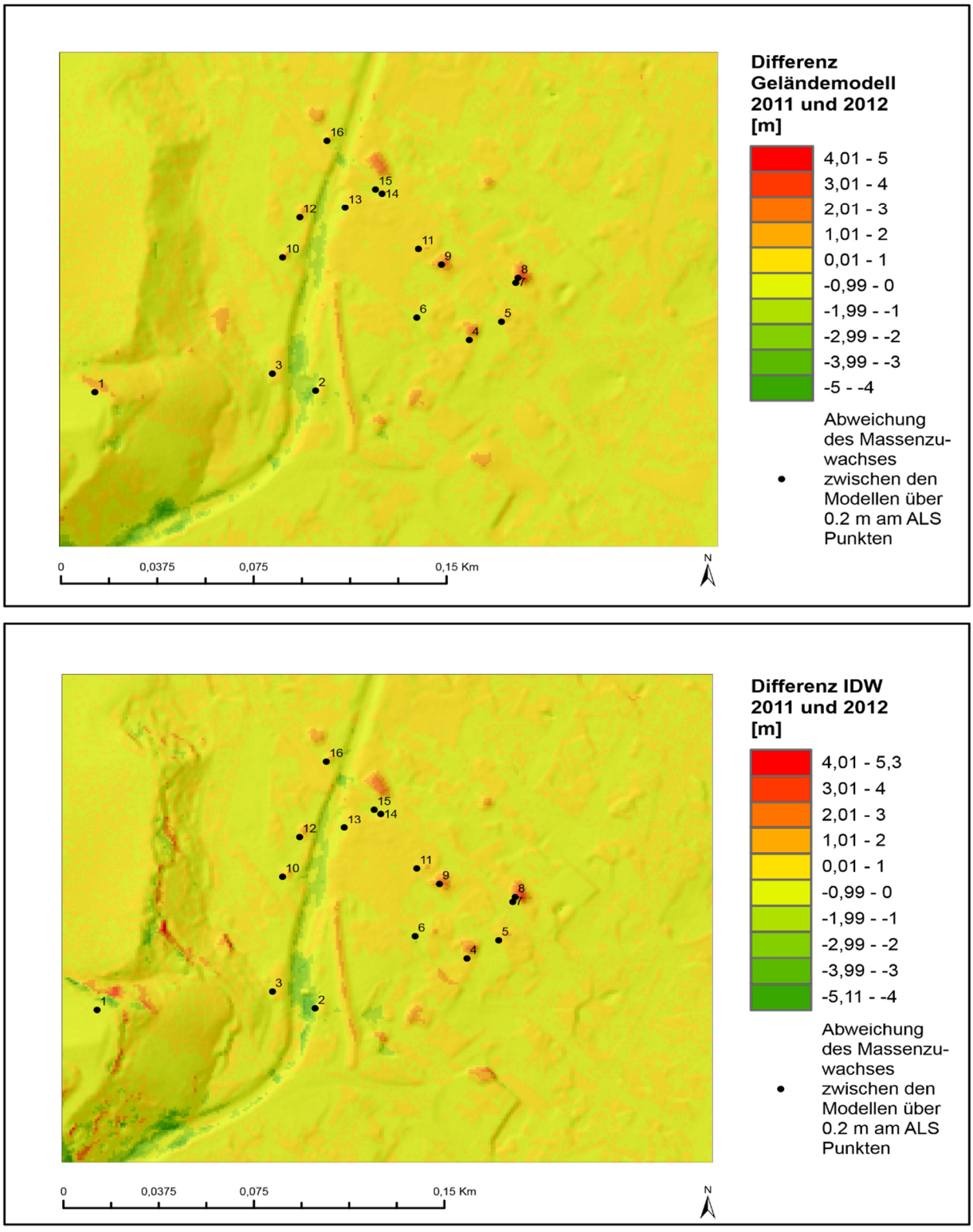

Abbildung 40: Differenzmodelle von 2011 und 2012 vom bereits bestehendem Geländemodell (oben) und aus der OpenSource IDW Methode generiertem DTM (unten) (Daten des Landes Steiermark, eigene Darstellung). Zusätzlich nummerierte ALS Punkte, an denen der Massenzuwach beider Modelle über 0,2 m voneinander abweichen (siehe Tabelle 11)

Tabelle 11: Genaue Ergebnisse der Modelldifferenzen und ALS Differenzen von 2011 und 2012 im Vergleich (Vgl. mit Abbildung 40, eigene Darstellung)

ID	Differenz Gelände-modell [m]	Differenz IDW Modell [m]	Differenz ALS [m]	Abweichung Differenz Geländemodell und ALS Differenz [m]	Abweichung Differenz IDW Modell und ALS Differenz [m]	Abweichung Differenz Geländemodell und Differenz IDW Modell [m]
1	0,16	-0,06	-0,02	0,18	-0,04	0,22
2	-0,63	-1,38	-0,65	0,02	-0,73	0,75
3	0,56	0,35	0,11	0,45	0,24	0,21
4	0,47	0,24	0,33	0,14	-0,09	0,23
5	0,16	-0,09	-0,11	0,27	0,02	0,25
6	-0,29	-0,50	-0,45	0,16	-0,05	0,21
7	0,27	0,01	-0,11	0,38	0,12	0,26
8	2,46	1,75	1,87	0,59	-0,12	0,71
9	2,00	1,79	1,87	0,13	-0,08	0,20
10	0,33	-0,07	-0,25	0,58	0,18	0,40
11	0,63	0,41	0,38	0,25	0,03	0,22
12	0,40	0,14	0,19	0,21	-0,05	0,26
13	-0,50	-0,79	-0,96	0,46	0,17	0,28
14	-0,30	-0,61	-0,65	0,35	0,04	0,31
15	-0,26	-0,59	-0,68	0,42	0,09	0,33
16	0,44	0,16	0,10	0,34	0,06	0,28
			Mittelwert	0,31	0,31	0,32
			RMSE	0,347	0,212	

Tabelle 12: Evaluierung der Höhenänderungen von 2011 auf 2012 in Sankt Lorenzen im Paltental von einem bereits bestehenden Geländemodell mit einem Geländemodell generiert aus dem OpenSource Tool mit der IDW Methode (eigene Darstellung)

	Abweichung der Modelldifferenz zu ALS Differenz [m]	Abweichung der IDW Differenz zu ALS Differenz [m]
Mittelwert	0,001	-0,001
Max	0,751	0,733
Min	-0,589	-0,347
RMSE	0,091	0,060

11 Fazit

Zur Generierung von DTMs aus LIDAR Daten gibt es keine universelle Interpolationsmethode, die am besten für diese Daten geeignet wäre. Das liegt vor allem an der Unterschiedlichkeit jeder Geländeoberfläche, der Datendichte, der gewünschten Auflösung, sowie auch für den gewünschten Zweck, für den die Daten bestimmt sind. In der Literatur gibt es unterschiedliche Meinungen über die IDW Methode. Diese liefere schlechtere Ergebnisse als Kriging, vor allem wenn die Datendichte sehr gering ist. Jedoch gibt es kaum Unterschiede zu Kriging, wenn die Datendichte sehr hoch ist. Ein weiterer Vorteil der IDW Methode ist, dass sie ein sehr einfacher und schneller Interpolator ist. Da in dieser Masterarbeit keine Reduktion der LIDAR Daten vorgenommen wurde und im allgemeinen die Daten eine sehr hohe Dichte aufweisen, wurde die IDW Methode als geeigneste Methode zur Toolprogrammierung ausgewählt.

Für Python stehen zur Verarbeitung von LIDAR Daten viele OpenSource Bibliotheken zur Verfügung. So können mit der Bibliothek *Laspy* LIDAR Daten schnell und einfach eingelesen und sogar bei Bedarf modifiziert und in eine neue Datei gespeichert werden. Zur Interpolation der Daten stehen Bibliotheken, wie NumPy und SciPy, zur Verfügung. Bei den bereits zur Verfügung stehenden Interpolationsmethoden in Python hat sich nur die Nearest Neighbour Methode bewährt. Andere Methoden konnten die LIDAR Daten entweder nicht verarbeiten oder lieferten keine zufrieden stellenden Ergebnisse. Der IDW lässt sich ohne Probleme mit Hilfe der LIDAR Daten programmieren und damit ein Tool zur Geländeinterpolation aus LAS Daten entwickeln.

Im Vergleich zu den lizenzpflichtigen Tools, wie LASTools und LP360, hat vor allem die IDW Methode des programmierten OpenSource Tools ganz gute Ergebnisse zeigen können. Im flachen Gebiet ist der RMSE bei der OpenSource IDW Methode um nur ca. 0,03 m höher als der der anderen Tools. Im komplexeren Gelände ist der RMSE um 0,217 m bis 0,363 m höher als bei den lizenzpflichtigen Tools. In zufällig ausgewählten Profilen verlaufen alle interpolierten Geländeflächen aller Tools sehr ähnlich.

Auch im Vergleich zu den bereits bestehenden Modellen konnte die programmierte IDW Methode ebenfalls sehr gute Ergebnisse liefern. Im Mittel waren beide Methoden gleich gut im Vergleich zu den ALS Daten und der RMSE Fehler der IDW Methode ist ca. um 0,1 m besser als beim bereits bestehenden Geländemodell.

Der Unterschied zwischen allen Tools liegt vor allem bei kleinen bis großen plötzlichen Geländeänderungen, von kleinen Gräben bis hin zu großen Höhendifferenzen mit Steilhängen. In komplexen Gebieten mit hoher Reliefenergie sind auch oft in diesen Bereichen entweder nur wenige oder keine ALS Bodenpunkte vorhanden, wodurch das Gelände von den Tools sehr unterschiedlich berechnet wird. Bei sehr steilen Hängen konnten die IDW Methoden, sowohl das programmierte IDW Tool als auch die LP360 IDW Methode, im Gegensatz zum LASTools, einen solchen Hang besser interpolieren.

Weiters gibt es neben den bereits erwähnten kleinen Geländeänderungen auch bei Flächen mit ALS freien Bodenpunkten Unterschiede, zum Beispiel dort, wo eigentlich Häuser stehen. Die Flächen der Häuser werden von allen Tools sehr unterschiedlich interpoliert. Da dort jedoch die Bodenpunkte fehlen, kann keine Aussage getroffen werden, welches Tool diese Bereiche am besten annähert.

Vorteil des programmierten Tools ist, dass es kostenlos ist. Das Tool kann unabhängig von Lizenzen ausgeführt werden und daher ist es auch möglich, die generierten Rasterdaten in weiteren OpenSource Programmen, wie zum Beispiel QGIS, weiter zu verarbeiten. Es ist weiters möglich, das Tool in die ArcGIS Toolbox zu integrieren, wo es sehr benutzerfreundlich ist. Je nach Bedarf können unterschiedliche Methoden gewählt werden und eine unterschiedliche Parametrisierung getroffen werden. Besonders die programmierte IDW Methode liefert sehr gute Ergebnisse und ist eine sehr schnelle und zuverlässige Methode. Die freie IDW Methode ist im Gesamten nur wenige Zentimeter ungenauer als LASTools und LP360, aber ihr berechnetes Gelände verläuft in der Profilansicht genauso gut und sie kann sogar Steilhänge besser als LASTools interpolieren. Sie ist daher eine gute Alternative zu lizenzpflichtigen Methoden. Ein weiterer großer Vorteil besteht darin, dass für die Benutzung der ArcGIS Tools kein vertieftes Wissen über LIDAR Daten benötigt wird.

12 Anhang

```
"""*****************************************************************
=====Python Skript für die Generierung von *.LAS Dateien zu einem Raster======
Autor: Julia Hauczinger
Python Version: 2.7

Input: Las Datei
Output: Raster in Ascii Format

Das Tool generiert aus LAS Daten ein Digitales Geländemodell (DTM) mit Hilfe
der Interpolationsmethoden:
- IDW (inverse distance weighting)
- Multivariate Interpolationsmethoden:
    - Nearest-neighbour Interpolation
    - stückweise lineare Interpolation
    - stückweise kubische Interpolation

Die Filterung der Punkte in Boden- und Wasserpunkte erfolgt im Tool.
Eine vorherige Filterung ist deshalb nicht nötig.

Folgende kostenlose Python Packages müssen vorher installiert sein:
- laspy: http://laspy.readthedocs.org/en/latest/
- numpy: http://www.numpy.org/
- scipy: http://www.scipy.org/
*****************************************************************"""

#**********************Packages importieren*************************
# Importiere Numpy Package
import numpy as np
from numpy import inf               #für Inf von tree.query
#Importiere Laspy Package
from laspy.file import File         #Zum Lesen des .LAS Files
#Importiere Scipy Package
from scipy.spatial import cKDTree   #Zum Erstellen des cKDTree
#Importe weitere zusätliche Module
import time
start_time = time.time()
import math                         #für math.ceil
```

```
#**********************IDW Interpolation*****************************
def idw(coords, values, grid_x, grid_y, power, k, distance_upper_bound):
    #Füge einen Pseudowert für „values“ hinten an (Grund ist bei tree.query)
    values = np.hstack((values, [0]))

    #Abflachen der Meshgrids und „values“ zu einem 1D Array
    xi, yi = grid_x.flatten(), grid_y.flatten()
    gridshape = grid_x.shape
    del grid_x, grid_y
    values = values.flatten()

    #Erstelle cKDTree Objekt von den Sample Points
    tree = cKDTree(coords)
    del coords
    #Erstelle ein leeres Grid, in das später die IDW Werte herein kommen
    valuesGrid = np.array([])
    #Erstelle ein regelmäßige Koordinatenarray-Grid (interp)
    interp = np.vstack((xi,yi)).T
    del xi, yi

    #Teile das regelmäßige Grid des Untersuchungebietes in Untergebiete
    #Anzahl der Tiles ist abhängig von der Größe des UG und k
    #Bei zu großen Array kommt es zum Memory Array
    #9 000 000 wurde ausprobiert, bis zur welchen Größe es zu keinem Memory
    #Error kommt
    points_in_tiles = 9000000/k
    split = math.ceil(interp.shape[0] / float(points_in_tiles))
    split_array = np.array_split(interp, split, axis = 0)
    #print(len(split_array))

    eps=0            #Gibt die ungefähren nächsten Nachbarn wieder
    p=2              #Minkowski p-norm: 2 ist die normale euklidisch Distanz

    del interp

    for arr in split_array:
        distance, inds = tree.query(arr, k, eps, p, distance_upper_bound)
        distance[distance == inf] = np.nan
        #IDW Formel
        w = 1.0 / distance**power
        idw = np.nansum(w * values[inds], axis=1) /np.nansum(w, axis=1)
        #Lösche überflüssige Variablen, um RAM freizugeben
```

```
        del distance, inds, w
        #IDW Ergebnisse in ein Grid schreiben
        valuesGrid = np.hstack((valuesGrid,idw))
        del idw
    del values
    valuesGrid = np.reshape(valuesGrid, (gridshape))
    #Ersetze alle NAN mit -9999
    valuesGrid[np.isnan(valuesGrid)] = -9999

    return valuesGrid

#***********************Öffnen eines LAS Files**************************
#Checke, ob eingegebener Pfad existiert
while True:
    pfad = raw_input("Pfad zum Ordner: \n")
    if  os.path.exists(pfad):
        break
    else:
        print "Pfad existiert nicht!"
#Checke, ob eingegebener Dateiname existiert
while True:
    name = raw_input("Name der Eingabe-LAS-Datei: \n")
    name = "/" + name + ".las"
    if os.path.isfile(pfad + name):
        break
    else:
        print "File existiert nicht!"
#Öffne Las File
inFile = File(pfad + name, mode = "r")

#*************************Punkte im LAS File lesenlesen**************
#X, Y, Z und Klasse aus *LAS lesen und in ein Array speichern
points = np.array([inFile.x, inFile.y, inFile.z,
inFile.raw_classification]).T
#***************Boden- und Wasserpunkte Filtern*****************
#Boolean Array erstellen
filter_array = np.any([points[:, 3] == 2, points[:, 3] == 9],axis=0)

#Mit Boolean Array das original Array maskieren
points = points[filter_array]
```

```
#*************************Interpolation vorbereiten****************
#Checke, ob Eingabe für die Zellengröße eine Zahl ist
while True:
    try:
        cellsize = raw_input(
        "Die Zellengröße im zu erstellenden Ausgabe-Raster: \n")
        cellsize = cellsize.replace(",", ".")
        cellsize = float(cellsize)
        break
    except ValueError as e:
        print("Error message: ", e)
        print "Error. Keine Zahl!"

#xyz min und max
min = inFile.header.min
max = inFile.header.max

# Anzahl der Spalten
cols = (int(max[0] - min[0]) / cellsize) + 1
# Anzahl der Zeilen
rows = (int(max[1] - min[1]) / cellsize) + 1

#Generiere ein regelmäßiges Grid zur Interpolation der Daten
grid_x = np.linspace(min[0], max[0], cols)
grid_y = np.linspace(max[1], min[1], rows)
grid_x, grid_y = np.meshgrid(grid_x, grid_y)

coords = points[:,0:2]
values = points[:,2]

del points #Lösche überflüssige Variablen

#***********************Wahl einer Interpolationmethode************
while True:
    interpolationmethode = raw_input(
    "Die Interpolationsmethode, die zum Erstellen"
    " des Rasters verwendet wird:\n"
    " Wähle: 'idw' oder 'multivariat'\n"
    "     idw - Verwendet inverse Entfernungsgewichtung"
    " (Inverse Distance Weighted)-Interpolation"
    "\n     multivariat - Verwendet multivariate Interpolationsmethoden\n")
```

```
#***********IDW*******************************
    if interpolationmethode == 'idw':
        #Potenzzahl bei der IDW Formel
        while True:
            try:
                power = raw_input(
                "Exponent der Entfernung: \n Je höher der Exponent,desto"
                " weniger Einfluss haben entferntere Punkte."
                "\n Gültige Eingabe ist jede reelle Zahl größer 0"
                "\n Die besten Ergebnisse liefern Werte zwischen 0.5 und 3"
                "\n Standartwert: 2\n")
                power = power.replace(",", ".")
                power = float(power)
                break
            except ValueError as e:
                print("Error message: ", e)
                print "Error. Keine Zahl!"
     #k: Maximale Zahl der nächsten Nachbarn, die wieder gegeben werden sollen
        while True:
            try:
                k = raw_input("Die Zahl der nächsten Nachbarn,"
                " die wieder gegeben werden sollen. z.B. 12: \n")
                k = int(k)
                break
            except ValueError as e:
                print("Error message: ", e)
                print "Error. Keine Zahl!"
#distance_upper_bound: Wiedergabe der Nachbarn innerhalb einer bestimmten
#Distanz
        while True:
            try:
                distance_upper_bound = raw_input("Suchradius z.B. 15:"
                                                 "\n Gibt nur Nachbarn"
                                                 " innerhalb dieser Distanz
"
                                                 "wieder.\n")
                distance_upper_bound = int(distance_upper_bound)
                break
            except ValueError as e:
                print("Error message: ", e)
                print "Error. Keine Zahl!"
```

```
        #Starte Zeitmessung
        start_time = time.time()
        #Rufe IDW Funktion auf
        grid = idw(coords, values, grid_x, grid_y, power, k,
                   distance_upper_bound)
        print("Raster erfolgreich berechnet.")
        print("\n{}{}".format("Laufzeit der Methode in Sekunden: ",
            round((time.time() - start_time), 0)))
        break

#*********Multivariate Interpolation***********
    elif interpolationmethode == 'multivariat':
        while True:
            method = raw_input("Multivariate Interpolationsmethoden:\n"
                               " Wähle: 'nearest', 'linear' oder 'cubic'\n"
                               "      nearest      -      Nearest-neighbour
                                Interpolation\n"
                               "     linear     -     stückweise     lineare
                                Interpolation\n"
                               "     cubic     -     stückweise     kubische
                                Interpolation\n")
            if method == 'linear':
                #Starte Zeitmessung
                start_time = time.time()
                grid = griddata(coords, values, (grid_x, grid_y), method,
                fill_value=-9999)
                break
            elif method =='nearest':
                #Starte Zeitmessung
                start_time = time.time()
                grid = griddata(coords, values, (grid_x, grid_y),
                method, fill_value=-9999)
                break
            elif method == 'cubic':
                #Starte Zeitmessung
                start_time = time.time()
                grid = griddata(coords, values, (grid_x, grid_y),
                method, fill_value=-9999)
                break
            else:
                print('Keine gültige Eingabe')
```

```
        #Lösche überflüssige Variablen
        del coords, values, grid_x, grid_y
        print("Raster erfolgreich berechnet.")
        print("\n{}{}".format("Laufzeit der Methode in Sekunden: ",
        round((time.time() - start_time), 0)))
        break

    else:
        print('Keine gültige Methode')

#************************Schreibe Ergebnisse in ein Outputfile********
# No data Werte für das Output DTM
NODATA = -9999

# Output ASCII DEM
outname = raw_input("Name der Outputdatei: \n")
target = outname + ".asc"

# Erstelle ein ASCII DTM Kopfzeile
header = "ncols         %s\n" % cols
header += "nrows         %s\n" % rows
header += "xllcorner     %s\n" % min[0]
header += "yllcorner     %s\n" % min[1]
header += "cellsize      %s\n" % cellsize
header += "NODATA_value      %s\n" % NODATA

# Öffne die Outputdatei, füge den Header an und speichere das Array
with open(target, "wb") as output:
  output.write(header)
  # %1.2f stellt sicher, dass die Zahlen im Array Floats sind
  # mit zwei Dezimalstellen
  np.savetxt(output, grid, fmt="%1.2f")

#************************Files schließen*****************************
inFile.close()
output.close()
```

13 Bibliographie

13.1 Monographien

ALBERTZ, J., & WIGGENHAGEN, M. (2009). *Taschenbuch zur Photogrammetrie und Fernerkundung*. Wichmann. Heidelberg. 334 S.

ALBERTZ, J. (2009). *Einführung in die Fernerkundung: Grundlagen der Interpretation von Luft- und Satellitenbildern*. Wiss. Buchges. Darmstadt. 249 S.

BONHAM-CARTER, G. (1994). *Geographic information systems for geoscientists: modelling with GIS* (No. 13). Pergamon. Oxford. 389 S.

BURROUGH, P. A., MCDONNELL, R., (1998). Principles of geographical information systems (Vol. 333). Oxford Univ. Press. Oxford. 333 S.

HÄRDER, T., RAHM, E. (2001): Datenbanksysteme: Konzepte und Techniken der Implementierung. Springer-Verlag berlin – Heidelberg. 582 S.

JENSEN, J. R. (2009). Remote Sensing of the Environment: An Earth Resource Perspective 2/e. Pearson Education India. Upper Saddle River. 592 S.

KLEIN, B. (2013): Einführung in Python 3: in einer Woche programmieren lernen. Hanser Verlag. München. 427 S.LAWHEAD, J. (2013). Learning Geospatial Analysis with Python. Packt Publishing Ltd. 364 S.

LI, Z., ZHU, C., & GOLD, C. (2005). *Digital terrain modeling: principles and methodology*. CRC press. Boca Raton. 340 S.

WANG, F. (2010). *Quantitative methods and applications in GIS*. CRC Press. Boca Raton. 265 S..

13.2 Wissenschaftliche Artikel

ABRAMOV, O., & MCEWEN, A. (2004). Technical note: An evaluation of interpolation methods for Mars Orbiter Laser Altimeter (MOLA) data. *International Journal of Remote Sensing*, *25*(3), 669-676.

ALI, T. A. (2004). On the selection of an interpolation method for creating a terrain model (TM) from LIDAR data. In *Proceedings of the American Congress on Surveying and Mapping (ACSM) Conference*. 1-18. AXELSSON, P. (1999). Processing of laser scanner data—algorithms and applications. ISPRS Journal of Photogrammetry and Remote Sensing, 54(2), 138-147.

AXELSSON, P. (2000). DEM generation from laser scanner data using adaptive TIN models. International Archives of Photogrammetry and Remote Sensing, 33(B4/1; PART 4), 111-118.

BATER, C. W., & COOPS, N. C. (2009). Evaluating error associated with lidar-derived DEM interpolation. *Computers & Geosciences*, *35*(2), 289-300.

BENTLEY, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, *18*(9), 509-517.

BROVELLI, M. A., CANNATA, M., & LONGONI, U. M. (2002). Managing and processing LIDAR data within GRASS. In Proceedings of the GRASS Users Conference (Vol. 29). 1-29.

BROVELLI, M. A., CANNATA, M., & LONGONI, U. M. (2004). LIDAR data filtering and DTM interpolation within GRASS. *Transactions in GIS*, *8*(2), 155-174.

CHAPLOT, V., DARBOUX, F., BOURENNANE, H., LEGUÉDOIS, S., SILVERA, N., & PHACHOMPHON, K. (2006). Accuracy of interpolation techniques for the derivation of digital elevation models in relation to landform types and data density. *Geomorphology*, *77*(1), 126-141.

CHILDS, C. (2004). Interpolating surfaces in ArcGIS spatial analyst. *ArcUser, July-September*, 32-35.

GARNERO, G., & GODONE, D. (2013). Comparisons Between Different Interpolation Techniques. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *1*(3), 139-144.

GUO, Q., LI, W., YU, H., & ALVAREZ, O. (2010). Effects of topographic variability and LIDAR sampling density on several DEM interpolation methods. Photogrammetric Engineering & Remote Sensing, 76(6), 701-712.

KRAUS, K., & PFEIFER, N. (2001). Advanced DTM generation from LIDAR data. International Archives Of Photogrammetry Remote Sensing And Spatial Information Sciences, 34(3/W4), 23-30.

LAM, N. S. N. (1983). Spatial interpolation methods: a review. The American Cartographer, 10(2), 129-150.

LIU, X. (2008). Airborne LiDAR for DEM generation: some critical issues. *Progress in Physical Geography*, *32*(1), 31-49.

LIU, X., ZHANG, Z., & PETERSON, J. (2009, October). Evaluation of the performance of DEM interpolation algorithms for LiDAR data. In Proceedings of the 2009 Surveying and

Spatial Sciences Institute Biennial International Conference: Spatial Diversity (SSC 2009). Surveying and Spatial Sciences Institute. 771-779.

LLOYD, C. D., & ATKINSON, P. M. (2006). Deriving ground surface digital elevation models from LiDAR data with geostatistics. *International Journal of Geographical Information Science*, *20*(05), 535-563.

MANEEWONGVATANA, S., & MOUNT, D. M. (1999, December). It's okay to be skinny, if your friends are fat. In *Center for Geometric Computing 4th Annual Workshop on Computational Geometry* (Vol. 2). 1-8.

MENG, X., CURRIT, N., & ZHAO, K. (2010). Ground filtering algorithms for airborne LiDAR data: A review of critical issues. *Remote Sensing*, *2*(3), 833-860.

MITAS, L., & MITASOVA, H. (1999). Spatial interpolation. *Geographical information systems: principles, techniques, management and applications*, *1*, 481-492.

PFEIFER, N. (2003). Oberflächenmodelle aus Laserdaten. *Österreichische Zeitschrift für Vermessung and Geoinformation*, *4*, 243-252.

RAMIREZ, J. R. (2006). A new approach to relief representation. *Surveying and Land Information Science*, *66*(1), 19-25.

REUTEBUCH, S. E., MCGAUGHEY, R. J., ANDERSEN, H. E., & CARSON, W. W. (2003). Accuracy of a high-resolution lidar terrain model under a conifer forest canopy. *Canadian Journal of Remote Sensing*, *29*(5), 527-535.

SITHOLE, G. (2001). Filtering of laser altimetry data using a slope adaptive filter. International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences, 34(3/W4), 203-210.

SU, J., & BORK, E. (2006). Influence of vegetation, slope, and lidar sampling angle on DEM accuracy. *Photogrammetric Engineering & Remote Sensing*, *72*(11), 1265-1274.

VOSSELMAN, G. (2000). Slope based filtering of laser altimetry data. In International Archives of Photogrammetry and Remote Sensing, Vol. XXXIII, B3. Amsterdam, Netherlands. 935–942

WEHR, A., & LOHR, U. (1999). Airborne laser scanning—an introduction and overview. *ISPRS Journal of Photogrammetry and Remote Sensing*, *54*(2), 68-82.

13.3 Internetquellen

AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING (2005): LAS Specification Version 1.1 March 07, 2005. http://www.asprs.org/a/society/committees/standards/asprs_las_format_v11.pdf (letzter Zugriff: 11/2014).

AMERICAN SOCIETY FOR PHOTOGRAMMETRY & REMOTE SENSING (2012): Divisions and Committees. LASer (LAS) File Format Exchange Activities. What is the LAS Format? http://www.asprs.org/Committee-General/LASer-LAS-File-Format-Exchange-Activities.html (letzter Zugriff: 02/2015).

BUTLER, H., LOSKOT M., ET AL. (2014): libLAS - LAS 1.0/1.1/1.2 ASPRS LiDAR data translation toolset. http://www.liblas.org/ (letzter Zugriff: 02/2015).

BROWN, G., BUTLER, H. (2012): Laspy: Documentation. http://pythonhosted.org/laspy/ (letzter Zugriff: 02/2015).

DIE PRESSE (2012): Naturkatastrophen: Ein Land in der "roten Zone". http://immobilien.diepresse.com/home/oesterreich/1270908/Naturkatastrophen_Ein-Land-in-der-roten-Zone (letzter Zugriff: 03/2015).

ESRI (2014): Desktop Help 10.0 - Was sind LIDAR-Daten? http://resources.arcgis.com/de/help/main/10.1/index.html#/na/015w0000004100000/ (letzter Zugriff: 11/2014).

LAND STEIERMARK (2010): Airborne Laserscanning für große Teile der Steiermark im Laufen. http://www.gis.steiermark.at/cms/beitrag/11696643/803916/ (letzter Zugriff: 02/2015).

MAGISTRAT DER STADT WIEN (2014): Technologie - Airborne Laserscanning (ALS). https://www.wien.gv.at/stadtentwicklung/stadtvermessung/geodaten/als/technologie.html (letzter Zugriff: 02/2015).

QCOHERENT (2015): LP360. http://www.qcoherent.com/. (letzter Zugriff: 02/2015).

RAPIDLASSO GMBH (2014): LASToolss. http://rapidlasso.com/LASToolss/ (letzter Zugriff: 02/2015).

SCIPY.ORG (2014): scipy.spatial.cKDTree. http://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.cKDTree.html (letzter Zugriff: 02/2015).