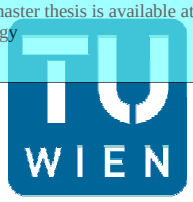

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

Studienrichtung Vermessung und Katasterwesen

DIPLOMARBEIT

Baummodellierung mit Hilfe von terrestrischem Laserscanning durch Zeichnen auf erstellten Plattkarten

ausgeführt am

Institut für Photogrammetrie und Fernerkundung (E122)
der Technischen Universität Wien

durch

Andreas Grafl

Ardaggerstraße 118
3300 Amstetten

Begutachter:

Univ.Prof. Dipl.-Ing. Dr.techn. Norbert Pfeifer

Betreuer:

Univ.Ass. Dipl.-Ing. Dr.techn. Markus Hollaus

Dipl.-Ing. Lothar Eysn

Wien, im Oktober 2012 ……..

Danksagung

Ich möchte mich sehr bei Univ. Ass. Dipl.-Ing. Dr. techn. Markus Hollaus, Dipl.-Ing. Lothar Eysn sowie Univ. Prof. Dipl.-Ing. Dr. techn. Norbert Pfeifer für die hervorragende Betreuung in fachlicher und persönlicher Hinsicht bedanken.

Zu besonderem Dank bin ich auch Dipl.-Ing. Dr. techn. Camillo Ressl für seine Beratung und rasche Hilfe in einigen Bereichen verpflichtet.

Kurzfassung

Im Rahmen des Projekts „3DVegLab“ der Europäischen Raumfahrtbehörde (ESA) wird für mehrere, ausgewählte Waldgebiete ein Linienmodell für die darin befindlichen Bäume benötigt. Die Linien des Modells sollen zusammen mit den dazugehörigen Durchmessern die Stämme und Äste exakt in ihrer Geometrie repräsentieren. Basis für die Bestimmung des Modells ist terrestrisches Laserscanning (TLS), das schon seit geraumer Zeit für die Aufnahme räumlicher Objekte in Verwendung ist.

In dieser Arbeit werden das Grundprinzip und die Möglichkeiten von TLS erklärt, sowie die Aufnahme durch TLS an Waldgebieten erläutert. Anschließend werden einige aktuelle automatisierte Methoden zur Baummodellierung präsentiert, und zwar nach Buksch [6], Pfeifer et al. [7], Bremer et al. [9] und Schilling et al. [8]. Daraufhin werden die Festlegung, die Entwicklung und die Umsetzung einer neuen, manuellen Methode und deren Anwendung in den festgelegten Testgebieten dargebracht. Ergebnisse der Methodenentwicklung und der Anwendung werden am Ende diskutiert.

Ziel dieser Arbeit ist, eine Methode zu finden, welche alle bisherigen Methoden der Baummodellierung an Vollständigkeit und geometrischer Korrektheit übertrifft. An Stelle der bisherigen, automatisierten Modellierungsmethoden, die sich noch als sehr mangelhaft erweisen, soll die Baumstruktur durch manuelles Nachzeichnen auf den Intensitätsplattkarten der TLS-Aufnahmen bewerkstelligt werden. Die Strukturen werden somit in den lokalen Standpunkten erstellt und anschließend ins Landeskoordinatensystem transformiert. Die Methode wird in der Praxis an einem Waldgebiet in Lägeren (Schweiz) und in Tharandt bei Dresden (Deutschland) durchgeführt und ihre Ergebnisse bezüglich Korrektheit demonstriert.

Abstract

For the project "3DVegLab" the European Space Agency (ESA) needs a line model that depicts the trees in multiple, selected forest areas. The lines of the model should exactly represent the geometry of trunks and branches, together with their corresponding diameters. Terrestrial laser scanning (TLS), which is in use for quite some time now, will be used to determine the model.

This thesis will explain the basic principle and the potential of TLS and offer an insight in TLS measurements of forest areas. Afterwards some current automated methods for tree modelling are presented, namely the ones introduced by Buksch [6], Pfeifer et al. [7], Bremer et al. [9] and Schilling et al. [8]. The main part of this thesis is the offering of a definition, and the development and implementation of a new, manual method and its application in the specified test areas. The results of method development and application will be discussed at the end.

The aim of this thesis is to find a method that surpasses all previous methods of tree modelling both in completeness and correctness of geometry. Instead of the previous, automated modelling methods that turned out to be very poor, the tree structure can be constructed by manually tracing the intensity image of equirectangular projection maps of TLS scans. This way the structures are created in the local viewpoints, and then transformed into the national coordinate system. The method is used in the forest area of Lägeren (Switzerland) and in Tharandt (Germany) and its results are evaluated on correctness.

Inhaltsverzeichnis

1 Motivation

Im Gegensatz zu klassischen, terrestrischen Vermessungssystemen hat TLS in den letzten Jahren stetig an Bedeutung zugenommen. Es wird immer öfter als Alternative zur Photogrammetrie verwendet. Insbesondere in der Aufnahme der Geometrie von Gebäuden und deren Fassaden, stellt sich TLS als kostengünstigere und schnellere Messmethode heraus als eine stereoskopische Aufnahme durch Photoaufnahmen. Zusätzlich ermöglicht diese Methode eine stärkere Automatisierung von Messabläufen und Modellauswertungen. Terrestrisches Laserscanning bestimmt ein Objekt durch ein dichtes Raster an 3-dimensionalen Messpunkten, somit ist eine Anwendung auf komplexe, unförmige Objekte möglich, welche deutlich mehr Punkte zur geometrischen Repräsentation benötigen, als die einfachen Drahtmodelle von Gebäuden. Interessante und häufige Objekte, welche komplex und unförmig sind, sind Bäume. Die Bestimmung der Baumgeometrie ist schon länger im Blickfeld der Forschung, die zu diesem Thema einige Publikationen hervorgebracht hat. Dabei fallen die Schwerpunkte auf Genauigkeit und Vollständigkeit der zu bestimmenden Bäume, sowie auch wichtige Parameter dieser Objekte (Höhe, Stammdurchmesser, Baumkronenparameter).

Im Jahr 2010 wurde von der Europäischen Raumfahrtbehörde (ESA) das Projekt „3DVegLab“ bewilligt, in dem die Bestimmung der Baumgeometrie einen wichtigen Abschnitt darstellt. So heißt es im veranschlagten Proposal:

„Erdbeobachtung spielte eine immer wichtigere Rolle bei der Beurteilung der räumlichen Ausdehnung und des Reichtums der terrestrischen Ökosysteme und optische Sensoren (wie die, die von der ESA betrieben werden) sind nachweislich ein Schlüsselquelle für Daten zur Ableitung von globalen Geoinformationsprodukten in der Wald- und Vegetationsdynamik. Ein wichtiger Aspekt der Vegetation ist die Baumkronenstruktur, die z.B. in der Ökosystem- oder Biotopbegutachtung sich auf Fernerkundungsergebnisse wie Höhe, Biomasse, Dichte und Albedo auswirkt. Allerdings kann in der passiven, optischen Fernerkundung die Struktur auch negativen Einfluss haben, wie die effektive Entkopplung des EO-Signals von radiometrischen Eigenschaften der Vegetation. Somit kann die Ableitung durch die Blattbiochemie erheblich beeinträchtigt werden, wenn die Wirkung der Baumstruktur nicht bekannt ist. Diese Einschränkungen können durch das Verständnis und die Nutzung des

Signals überwunden werden (z.B. durch Strahlungstransfermodellierung), insbesondere in Verbindung mit Mehrwinkelbeobachtungen, und die Einbindung neuer Methoden und Werkzeuge, vor allem die direkte Bestimmung der Vegetationsstruktur durch Terrestrisches Laserscanning (TLS) und Airborne Laserscanning (ALS).“ (eigene Übersetzung aus dem Englischen)

2 Zielsetzung

Im Rahmen dieses Projekts ist die Modellierung der Baumstruktur in speziellen, ausgewählten Gebieten von stehenden einzelnen Bäumen notwendig. Die Aufnahme der Bäume soll hier mit TLS erfolgen, da nur hier sich die genaue Struktur der Bäume in aufnahmetechnischer Hinsicht extrahieren lässt. Ein generiertes Modell der Bäume soll gewissen Anforderungen entsprechen. Es soll ein Linienmodell (auch Drahtmodell) der Aststruktur sein, dessen Linien durch die Achsen der einzelnen Astsegmente führen. Dabei sollen die Linien an den Endpunkten miteinander verbunden sein und auch Information über den Durchmesser des repräsentierten Astes beinhalten. Somit kann ein komplettes Zylindermodell erstellt werden. Die Position der Linien innerhalb der Punktwolke soll auf wenige Zentimeter genau sein. Ziel dieser Diplomarbeit ist, vorhandene Methoden zu untersuchen, wie aus georeferenzierten Punktwolken aus der Aufnahme der Laserscanner-Standpunkte entsprechende Linienmodelle der Bäume erstellt werden kann, sowie eine endgültige, geeignete Methode zu finden, die die Zielsetzungen des Projekts erfüllen kann. Mit dieser Methode soll das gesuchte Modell der Bäume in den gewählten Gebieten erzeugt werden.

3 Grundlagen des Terrestrischen Laserscannings

3.1 Grundprinzip

Mit einem Laserscanner kann eine Objektoberfläche abgetastet werden. Mit Hilfe eines stark gebündelten Laserstrahls wird aus der Laufzeit eines Impulses von der Aussendung bis zum Empfang die Entfernung zwischen dem Laserscanner und der jeweiligen Objektstelle, an der der Laserstrahl diffus reflektiert wird, gemessen ([1], S. 471). Die Messung erfolgt an Hand einer Photodiode am Sendegerät, welche das zurückkommende Lichtsignal empfängt ([2], S. 3). Aus der Laufzeit, multipliziert mit der Gruppengeschwindigkeit des Lichts, die etwa um 0,03 % geringer als die Vakuum-Lichtgeschwindigkeit ist, ergibt sich die doppelte Entfernung zwischen Sender und Objekt ([1], S. 471). Der terrestrische Laserscanner wird, im Gegensatz zum flugzeuggetragenen Laserscanner, während des Messvorgangs in seiner Lage nicht bewegt. Er ist stationär und benötigt daher Ablenkmechanismen an zwei Achsenrichtungen ([1], S. 471).

Laserlicht kann entweder kontinuierlich oder in kurzen Impulsen ausgesendet werden. Im Fall eines „Continuous Wave“ Laserscanners kann die Entfernung zu einem Zielobjekt über den Vergleich der Phasenlage der ausgesendeten und der empfangenen, intensitätsmodulierten Strahlung gemessen werden, im Fall eines „Pulsed Lasers“ über die Zeit, die der Lichtimpuls benötigt, um vom Scanner zum Objekt und zurück zu gelangen ([2], S. 3).

3.1.1 Distanzmessung mittels Impulsverfahren

Viele terrestrische Laserscanninggeräte basieren auf dem Impulsverfahren. Dabei wird die Messrate durch einen Pulsgenerator vorgegeben und die Laufzeit eines oder mehrerer, rückgestreuter Impulse gemessen, wenn es zu Mehrfachreflexionen auf der Erdoberfläche kommt ([2], S. 3). Abbildung 1 illustriert, wie ein ausgesendeter Laserimpuls (zur Vereinfachung als Rechteckimpuls dargestellt) durch die Interaktion mit einer Oberfläche in seiner Form verändert werden kann, wenn er auf räumlich ausgedehnte Streuobjekte trifft ([2], S. 4).

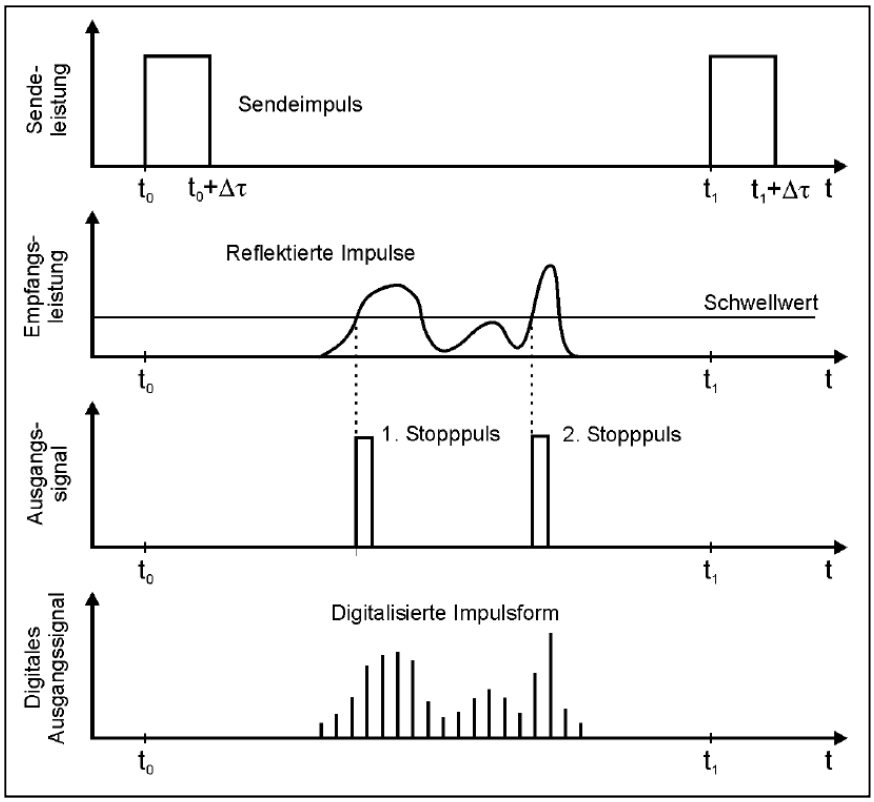

Abbildung 1 : Ausgesendeter und empfangener Impuls beim Laserscanning. Kommerzielle Laserscanningsysteme erfassen einen oder mehrere Stopppulse. Experimentelle Systeme digitalisieren die gesamte Impulsform des empfangenen Signals ([2], S. 5)

Für die Bestimmung der Entfernung einzelner Objekte werden verschiedene Verfahren eingesetzt, die in Abhängigkeit von Form und Anordnung der Streukörper unterschiedliche Ergebnisse liefern können. Abbildung 1 zeigt ein einfaches Schwellwertverfahren, welches auf ansteigende Impulsflanken anspricht. Andere Verfahren können zum Beispiel darauf abzielen, den Impulsschwerpunkt zu bestimmen. ([2], S. 6) Beim Impulsschwerpunkt-verfahren wird das Echo im Normalfall einem späteren Zeitpunkt (größere Entfernung) als vergleichsweise beim Grenzwertverfahren zugeordnet. Ein weiteres alternatives Verfahren beruht auf der Bestimmung der lokalen Maxima über die zeitliche Differenzierung des Echosignals und die Bestimmung der Nulldurchgänge (zero-crossing-Verfahren) ([2], S. 6).

3.1.2 Distanzmessung mittels Phasenvergleichsverfahren

Beim Continuous-Wave-Verfahren wird kontinuierlich elektromagnetische Energie abgestrahlt. Auch bei dieser Technik kommt das Signal um Δt verzögert an der Photodiode an. Es werden aber andere physikalische Merkmale als die Laufzeit ausgewertet. Das Continuous-Wave-Verfahren erlaubt eine höhere Messgeschwindigkeit als ein Impulslaufzeitverfahren, da bei ihm für jeden Impuls eine bestimmte Latentzeit nötig ist, um Folgeimpulse voneinander unterscheiden zu können ([3], S. 35).

Anstatt der Pulsmodulation tritt beim Phasendifferenzverfahren eine sinusförmige Amplitudenmodulation $s_s(t)$ mit der Messfrequenz ν_m auf ([3], S. 36). Dabei ist A_s die Amplitude und t die Zeit.

$$s_s(t) = A_s \cdot \sin(2\pi\nu_m t) \qquad \text{(Gl. 1)}$$

Das empfangene Signal $s_e(t)$ kommt um $\Delta t = 2\frac{r}{c}$ verzögert an und weist somit eine Phasendifferenz $\Delta\phi$ zwischen Sende- und Empfangssignal auf ([3], S. 37).

$$s_e(t) = A_e \cdot \sin(2\pi\nu_m t + \Delta\varphi) \qquad \text{(Gl. 2)}$$

Die Laufzeit wird aus der Phasenverschiebung $\Delta\varphi$ bestimmt (Abbildung 2) ([3], S. 37).

$$r = \frac{\Delta\varphi}{4\pi\nu_m} c = \frac{\lambda_m}{4\pi}\Delta\varphi \qquad \text{(Gl. 3)}$$

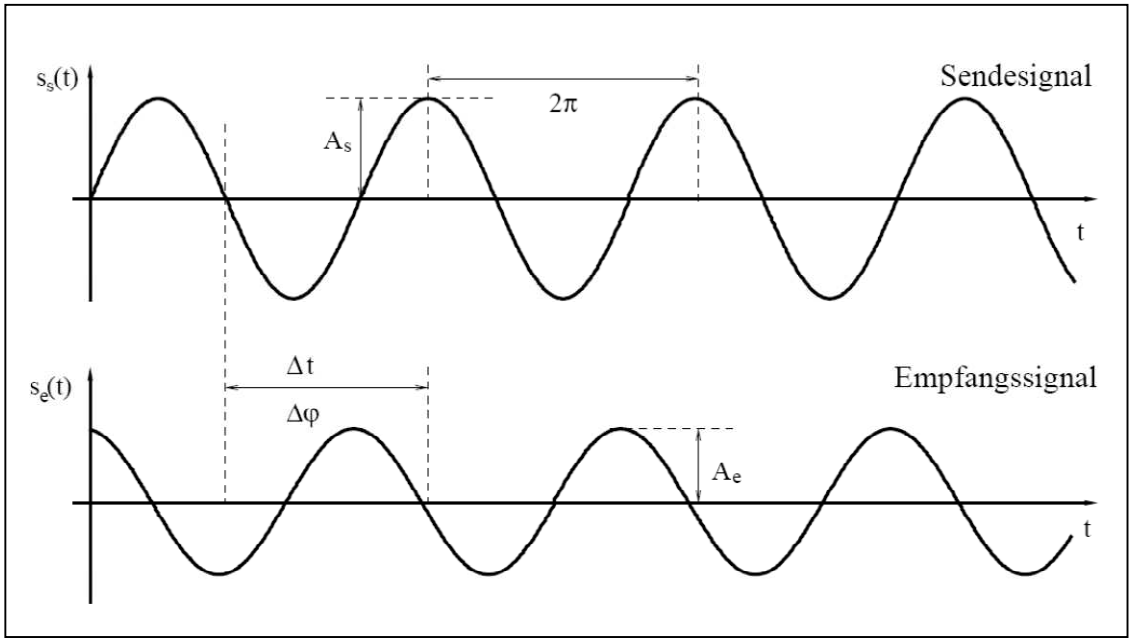

Abbildung 2 : Verlauf des Sende- und Empfangsvorgangs beim Phasendifferenzverfahren ([3], S. 36)

3.1.3 Laserscanner-Ablenkeinheit

Das Impulsverfahren und das Phasenvergleichsverfahren sind beide Distanzmessverfahren, die in modernen Tachymetern ihre Anwendung finden. Der Übergang von der einzelnen Entfernungsmessung zur flächenhaften und bildgebenden Entfernungsmessung erfordert bei der Benutzung eines Laser-Entfernungsmessers eine effektive Abtast- oder Ablenkeinheit, auch Scanner genannt. Der Scanner muss insbesondere hinsichtlich der Arbeitsgeschwindigkeit optimiert werden. Die Geschwindigkeit der Entfernungsmessung

kann bei Duldung von Genauigkeitsverlusten deutlich erhöht werden. Die Geschwindigkeit der Bewegungsabläufe einer Ablenkeinheit ist aber begrenzt, da hier Teile mechanisch zu bewegen und zu beschleunigen sind. Eine Abtasteinheit mit hoher Dynamik ist ein entscheidender Bestandteil eines Laserscanners. Eine zweite wichtige Aufgabe der Abtastung ist die Realisierung diskreter Abtastfolgen mit möglichst geringen Abtastinkrementen und großen Überdeckungen. Durch die Größe eines Abtastinkrements wird indirekt über verschiedene Abtastsysteme die räumliche Auflösung z.B. bezüglich des Azimuts und der Elevation festgelegt. Die Auflösung der lasergestützten Entfernungsmessung und die zweidimensionale Auflösung des Abtastsystems bilden zusammen die dreidimensionale Gesamtauflösung eines Laserscanners. Im Allgemeinen ist die räumliche Auflösung eines Laserscanners bezüglich eines kartesischen Koordinatensystems nicht homogen, da die Abtasteinheiten überwiegend Rotationsbewegungen ausführen ([3], S. 50-51).

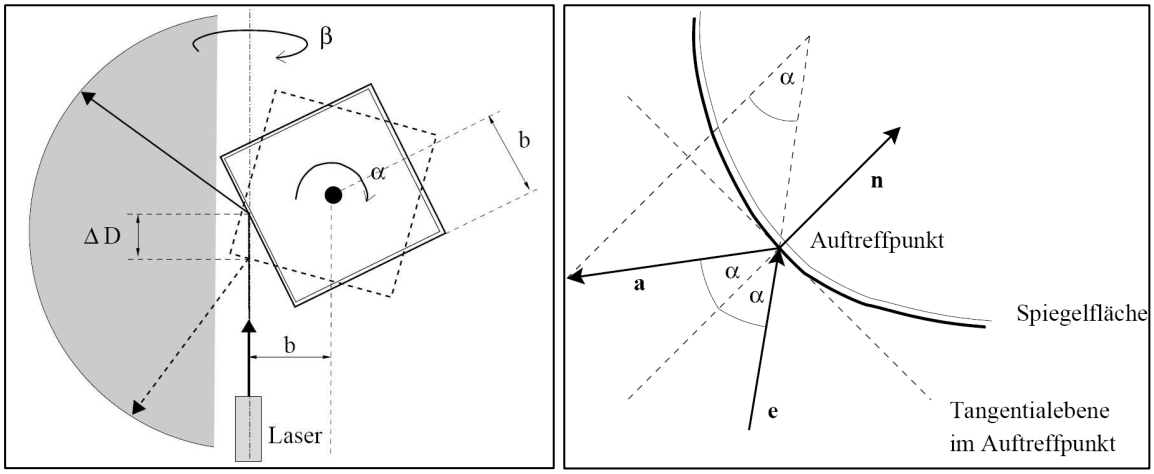

Abbildung 3 : links: Abtastung mit einem kontinuierlich rotierenden Spiegelpolygon ([3], S. 55)
rechts: Reflexion des Lichtes bei Planspiegeln ([3], S. 55)

Für die Ablenkung des Laserstrahls werden verschiedene Systeme verwendet. Hohe Abtastgeschwindigkeiten werden mit Spiegelpolygonen erreicht (Abbildung 3, links). Die kontinuierliche Drehbewegung erfordert nur geringe Motorleistungen. Die Abtastung erfolgt durch die gleichförmige Drehbewegung. Der Rücksprung entlang der Zeile (β-Profil, Abbildung 3, links) wird beim Übergang zur nächsten Spiegelfacette ausgeführt. Die Spiegelflächen sind in der Kantennähe geschwärzt, um eine eindeutige Trennung der Abtastzeilen zu erreichen. Da die Drehachse nicht in der spiegelnden Facette liegen kann (Abstand b, siehe Abbildung 3), tritt während der Abtastung eine Veränderung der inneren

Länge des Lichtweges ein. Die Streckendifferenz ΔD ist abhängig vom Drehwinkel α und ist an die Rohmessung anzubringen. Durch ein Spiegelpolygon mit vier Facetten kann theoretisch ein Gesichtsfeld von 0 bis 180° abgetastet werden. ([3], S. 51)

Ein Abtastsystem lässt sich auch durch Spiegel oder Prismen realisieren (Abbildung 3, rechts). Eine einfache und effektive Methode ist die Ablenkung durch einen Planspiegel der senkrecht in den Strahlengang des Lasers platziert und über einen Schrittmotor gedreht wird. Zur Beschreibung der Ablenkung eines Lichtstrahls an einem Planspiegel, reicht es aus, das einfache Reflexionsgesetz der Wellenoptik anzunehmen. Danach ist der Einfallswinkel bezüglich der Spiegelnormalen gleich dem Ausfallswinkel (Abbildung 3, rechts). Einfallender und reflektierter Lichtstrahl liegen immer in einer Ebene. Ist eine Spiegelfläche als Ebene mit dem Normalvektor $\vec{n}$ und der Richtung des einfallenden Lichtstrahls durch den Richtungsvektor $\vec{e}$ gegeben, so gilt für den Richtungsvektor $\vec{a}$ des reflektierenden Lichtstrahls:

$$\vec{a} = \vec{e} - 2 \cdot (\vec{e} \cdot \vec{n}) \cdot \vec{n} \qquad \text{(Gl. 4)}$$

Da die Vektoren $\vec{a}$, $\vec{n}$ und $\vec{e}$ in einer Ebene liegen, kann mit einem Spiegel ein Lichtstrahl nur in dieser Ebene abgelenkt werden. Ein rotierender Planspiegel lenkt einen Lichtstrahl so ab, dass in alle Richtungen (ausgehend vom Auftreffpunkt) innerhalb der durch Einfallsrichtung und Rotationsachse gebildeten Ebene abgetastet werden kann. Mit einem einzigen, rotierenden Planspiegel kann nur ein Profil- bzw. Linienscanner realisiert werden. Eine Auslenkung des Planspiegels um den Wert $\Delta\alpha$ bewirkt eine Ablenkung um den Winkel $2\Delta\alpha$. Also ist bereits mit einer 180°-Drehung eines einseitig verspiegelten Planspiegels der gesamte Richtungsbereich von 360° abgetastet. Der Planspiegel kann in einer kontinuierlichen Drehbewegung gehalten werden und muss nicht nach einer 180°-Drehung auf die Startposition zurückgedreht werden. Denn die dabei notwendigen Beschleunigungsvorgänge verlangsamen den Abtastvorgang deutlich, wobei die Masse eines Planspiegels auf Grund der hohen Laserbündelung klein ist im Vergleich zu der Masse eines Theodolitunterbaus. Die Ablenkung mittels optischer Systeme führt zu hohen Abtastgeschwindigkeiten. Eine kontinuierliche Drehbewegung, ohne Zwischenstopps an diskreten Messstellen, erlaubt die höchsten Abtastgeschwindigkeiten. Die Abtastung erfolgt dann durch das Abstimmen der Frequenzen der Planspiegeldrehfrequenz mit der

Abgrifffrequenz für die Entfernungsmessung. Der zu einer Entfernungsmessung gehörende Ablenkwinkel wird dann eine Funktion der Zeit. ([3], S. 53)

3.1.4 Vergleich aktueller Laserscanner

Die unterschiedlichen Systeme in der Distanzmessung (siehe Kapitel 3.1.1 und 3.1.2) und der Ablenkeinheit (siehe Kapitel 3.1.3) finden in aktuellen, am Markt befindlichen Terrestrischen Laserscannern Verwendung. Dabei ist es nicht einfach, das richtige Gerät für den entsprechenden Bedarf auszuwählen. „Scheint es doch für jede Anwendung einen speziellen Scanner zu geben, der sich am besten eignet, andersherum jedoch keinen Scanner, der für alle Anwendungen einsetzbar scheint ([4], S.1).“ Aktuelle Laserscannermodelle wurden von Lindstaedt [4] hinsichtlich ihrer Anwendung und Genauigkeit untersucht. Die Modelle waren Riegl VZ-400, Leica C10 (beide mit Impulsverfahren), Faro Photon 120 und IMAGER 5006i von Zoller + Fröhlich (beide Phasenvergleichsverfahren) und sind in Abbildung 4 in ihren Eigenschaften und Leistungsdaten dargestellt, wobei immer die höchsten Leistungsgrade verglichen wurden. Beim Vergleich fällt vor allem die hohe Messgeschwindigkeit der Phasenvergleichsscanner im Gegensatz zu den Impulsscannern auf. Die Impulsscanner überzeugen aber mit deutlich höherer Reichweite und mit etwas höhere Genauigkeit in der Distanzmessung. Für die jeweilige Anwendungsmöglichkeit muss daher der passende Laserscanner gefunden werden.

Scanner/Kriterium	Leica C10	Faro Photon 120	Riegl VZ-400	Z+F IMAGER 5006i
Messverfahren	Impuls	Phase	Impuls	Phase
Gesichtsfeld [°]	360 x 270	360 x 320	360 x 100	360 x 310
Scandistanz [m]	300	120	600	< 79
Scangeschwindigkeit	50.000pts/s	976.000pts/s	125.000pts/s	500.000px/s
Winkelauflös. H/V [°]	0,0023	0,009	0,0005	0,0018
Laserspotgröße	4,5mm/50m	11,3mm/50m	16mm/50m	13mm/50m
Wellenlänge [nm]	532	785	Nahes Infrarot	658
3D Punktgenauigkeit	6mm	keine Angabe	keine Angabe	10mm/50m
Distanzgenauigkeit	4mm bis 50m	2mm/25m	5mm/100m	6mm/50m
Kamera	integriert	Aufsatz	Aufsatz	Aufsatz
Neigungssensor	ja	ja	ja	ja

Abbildung 4 : Technische Spezifikationen der untersuchten Terrestrischen Laserscanner ([4], S. 2)

3.2 Messablauf

Der Ablauf eines Laserscanning-Messprojekts ist in mehrere Arbeitsschritte unterteilt, wobei nicht alle Schritte stets durchgeführt werden, wie z.B. eine vorhergehende Kalibrierung im Labor. Nach Pfeifer et al. [5], S. 2 sind das die Schritte:

1. Kalibrierung im Labor (unabhängig vom Scannen eines bestimmten Objekts)
2. Festlegung der erforderlichen Auflösung (am Objekt), der Genauigkeit (des Modells) und des Modelltyps
3. Wahl der Standpunkte
4. Scannen, also Datenerfassung vor Ort
5. Feldkontrolle und/oder Kontrolle der Datenerfassung im Nachhinein
6. Projektbegleitende Kalibrierung
7. Relative Orientierung
8. Absolute Orientierung
9. Modellerstellung
10. Qualitätskontrolle

Der 4. Schritt beinhaltet das Messen, also die Gewinnung von Beobachtungen. Vorhergehende Schritte beinhalten Vorbearbeitungen und nachhergehende Schritte Nachbearbeitungen. Dabei ist in allen Schritten eine gewisse Automatisierung möglich, aber nach wie vor ist eine Planung notwendig. Die aufzunehmenden Objekte werden oft von mehreren Standpunkten aus aufgenommen, was zu einer Überlappung bzw. Ergänzung der Punktwolke führt. Die von einem Standpunkt aus erfassten Daten bezeichnet man beim terrestrischen Laserscanning oft als Scan ([1], S. 471). Eine Planung ist vor allem bei der Bestimmung der Anzahl der notwendigen Standpunkte wichtig (Schritt 3), sowie die an die jeweilige Aufgabe angepasste Aufnahmeentfernung. Je nach Aufgabe sollen die aufzunehmenden Objekte von verschiedenen Seiten gemessen werden. Dabei ist auch die erforderliche Auflösung festzusetzen. Nach der Aufnahme der Scans sind diese zu einander relativ zu orientieren (Schritt 7). Beim TLS versteht man unter Orientierung die Bestimmung von Transformationsparametern der Scans zu einem übergeordneten Koordinatensystem. In der Regel genügen die 6 Parameter zu einer räumlichen Kongruenztransformation. Die Scans

sind an Hand einer Helmerttransformation zu orientieren, wobei die Parameter durch die Messung von Verknüpfungspunkten und Passpunkten in der Nachbearbeitung berechnet werden. Diese Punkte werden schon vor der Messung durch Markierungen und Messmarken, welche von möglichst vielen Standpunkten aus gesehen werden sollen, im Aufnahmegebiet untergebracht. Einige Messmarken werden zur absoluten Orientierung als Passpunkte festgesetzt. Diese dienen zur Orientierung in einem übergeordneten, in der Regel genordeten Koordinatensystem, wie einem Landeskoordinatensystem. Diesen Vorgang nennt man auch Georeferenzierung. Die georeferenzierte Gesamtpunktwolke von allen aufgenommenen Scans ist Basis zur Erstellung eines einfacheren, mathematischen Modells der aufgenommenen Objekte. Dieser Vorgang wird ausschließlich im Postprocessing angewendet. Trotzdem hat die vorhergehende Planung der Messung einen großen Einfluss auf alle Möglichkeiten zur Erstellung eines Modells. Dies ist in jeder Planung zu berücksichtigen. Es gibt viele Möglichkeiten zur Erstellung eines Modells und doch laufen alle auf das gleiche Ziel hinaus: Es soll so viel wie möglich an Information aus den Messungen der Punktwolke für das Ziel des Projekts extrahiert werden. Auf Grund der in den letzten Jahren stark angewachsenen Datenmengen, welche das Laserscanning erzeugt, ist Automatisierung in der Modellerstellung das derzeit aktivste Forschungsgebiet. Viele Arbeitsstunden könnten dadurch eingespart werden. Trotzdem ist die Automation in vielen Anwendungen noch nicht zufriedenstellend.

4 Testgebiete und Daten

4.1 Aufnahmegebiete

Es werden zwei verschiedene Aufnahmegebiete für die Erstellung der Baumstrukturen verwendet. Diese Aufnahmegebiete wurden im Projekt „3D-VegLab“ ausgewählt.

4.1.1 Lägeren, Schweiz

Ungefähre Lage des Aufnahmegebiets in geographischen Koordinaten:

47°28‘41‘‘ Nord 8°21‘51‘‘ Ost

Abbildung 5 : zwei Waldgebiete von der Größe 30 x 30 m bei Lägeren, Schweiz (aus GoogleMaps)

Hier wurden zwei kleinere Waldgebiete von einer Größe von jeweils 30 x 30 m mit TLS aufgenommen (Abbildung 5). Die Gebiete wurden nach unterschiedlicher Vegetationsart ausgewählt. So befindet sich im östlichen der zwei Gebiete ein Mischwald mit Laubbäumen wie auch Nadelbäumen. Das westliche Gebiet wurde vorrangig auf Grund von überwiegend vorhandenem Laubwald ausgewählt. Es gibt unterschiedliche Baumarten mit unterschiedlicher Stammdicke. In beiden Gebieten ist mit einer größeren Menge Unterholz zu rechnen.

4.1.2 Tharandt bei Dresden, Deutschland

Ungefähre Lage des Aufnahmegebiets in geographischen Koordinaten:

50°57‘49‘‘ Nord 13°34‘01‘‘ Ost

Abbildung 6 : Waldgebiet von der Größe 60 x 100 m bei Tharandt, Deutschland (aus GoogleMaps)

Ein größeres Waldgebiet von 60 x 100 m wurde mit terrestrischem Laserscanning aufgenommen (Abbildung 6). Das Gebiet wurde auf Grund des überwiegenden Anteils an Nadelwald ausgewählt. Dieser Nadelwald ist als Fichtenmonokultur erkennbar und weist in überwiegend regelmäßigen Abständen von einander stehende Bäume ähnlicher Stammdicke auf. Unterholz ist hier nur gering vorhanden.

4.2 Messgerät

Für die Aufnahmen wurde der Laserscanner „Z+F IMAGER® 5006h“ von der Firma Zoller und Fröhlich (Z+F) verwendet.

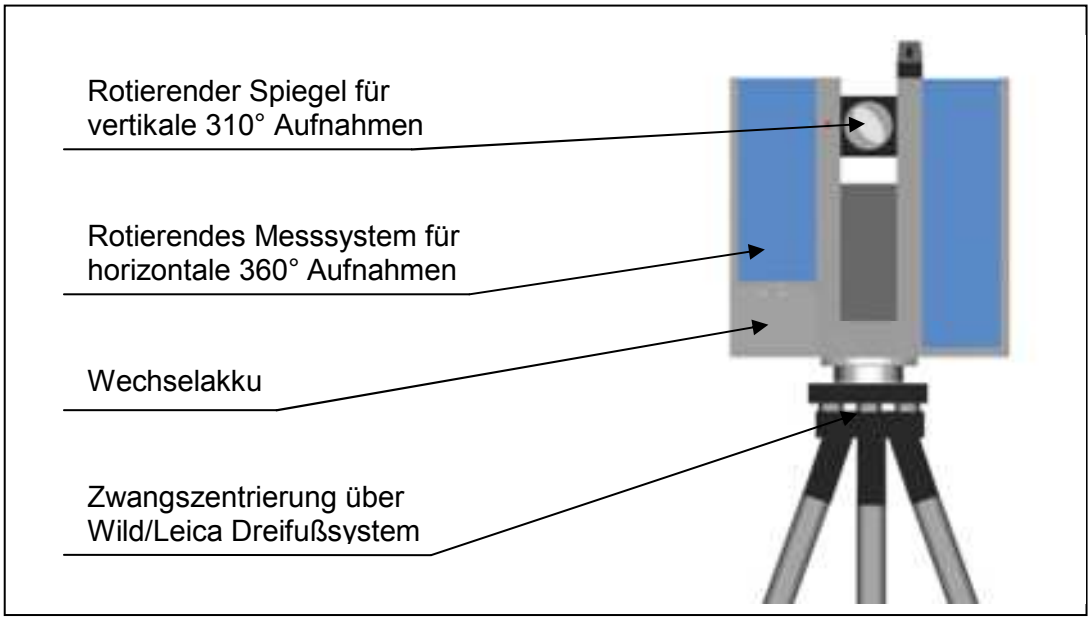

Abbildung 7 : Einfache Darstellung des Laserscanners Z+F IMAGER® 5006h [14]

Der Scanner verwendet das sogenannte „Phasendifferenzverfahren" (wie in Kapitel 3.1.2). Es wird bei der Aufnahme Entfernungswert (Entfernung zum Objekt) und Intensitätswert (Reflektivität des Objekts) gleichzeitig gemessen ("pixelweise Korrespondenz") [13]. Der Laserscanner hat eine Messreichweite von 79 m. Es wird eine Genauigkeit im Millimeterbereich erreicht. Dabei ist die Ablenkeinheit für die vertikale und horizontale Genauigkeit und das Lasermesssystem für die Genauigkeit der Distanz verantwortlich. Die Auflösung des aufgenommenen Punktrasters kann beliebig nach Bedarf eingestellt werden.

Technische Daten des Z+F IMAGER® 5006h [14]

Lasermesssystem	
Eindeutigkeitsbereich:	79 m
Min. Messentfernung:	0,4 m
Auflösung, Entfernung:	0,1 mm
Linearitätsfehler bis 50m:	≤ 1mm
Optischer Sender	
Laser:	im sichtbaren Bereich
Strahldurchmesser (Distanz 1m):	3 mm kreisrund
Ablenkeinheit	
System vertikal/horizontal:	rotierender Spiegel/drehender
Messkopf Sichtfeld vertikal/horizontal:	310°/360°
Auflösung vertikal/horizontal:	0,0018°/0,0018°
Genauigkeit vertikal/horizontal:	0,007° rms/0,007°rms
Auflösung	
Auflösungsstufen:	Pixel/360°(vertikal, horizontal)
„preview":	1250
„middle":	5000
„high":	10000
„super high":	20.000
„ultra high":	40.000

4.3 Datenaufnahme

Die Datenaufnahme erfolgte pro Aufnahmegebiet in mehreren Schritten und entsprechen als praktische Teile den Schritten 3 und 4 nach Pfeifer et al. [5] in Kapitel 3.2:

1. Aufbau
2. Koordinatenbestimmung von Hilfspunkten

3. Messung der Zieltafeln und Verknüpfungskugeln
4. Messungen mit dem Laserscanner
5. Abbau

Der Aufbau geschah nach einem vor Ort erstellten Aufnahmeplan. Die Bäume im Aufnahmegebiet sollten von allen Seiten gemessen werden. Bei Betrachtung des Gebietes wurde entschieden, wo der Scanner mit dem Stativ aufgestellt wird. Diese Punkte wurden durch Markierungen in Form von kleinen, rosaroten Signalfähnchen bestimmt. Zur Verknüpfung der einzelnen Scans wurden nummerierte, ca. 20 cm dicke Styroporkugeln (Abbildung 8) so aufgestellt, dass sie möglichst von allen Aufnahmestandpunkten aus sichtbar waren. Zusätzlich wurden noch Zieltafeln (Abbildung 8) verwendet, die zur Georeferenzierung, also zur absoluten Positionierung der Scans im Landeskoordinatensystem im Aufnahmegebiet herangenommen werden. Zusätzlich wurden noch am Boden Hilfspunkte für den Anschluss an das Landeskoordinatensystem festgelegt, von denen aus alle Zieltafeln und möglichst alle Verknüpfungskugeln zu sehen sind. Für die bessere Sichtbarkeit wurde abgestorbenes Unterholz, welches sich im Aufnahmegebiet befand, wenn möglich, weitestgehend entfernt. Für die Aufnahme der Bäume wurde mit einer Aufnahmeentfernung von bis höchstens 60 m gerechnet. Dementsprechend mussten die Positionen der Aufnahmestandpunkte gewählt werden, um eine entsprechende Dichte und Überlagerung der lokalen Scans zu gewährleisten. Die Zieltafeln dienten in der späteren Georeferenzierung (siehe Kapitel 4.4) als Passpunkte, um den Bezug zum übergeordneten Landeskoordinatensystem herzustellen. Diese wurden mit einer Totalstation von den Hilfspunkten aus polar eingemessen. Mit der Totalstation wurde auch ein abgeschlossener Polygonzug von diesen Hilfspunkten zu weiter entfernten, amtlichen Vermessungspunkten gemessen, um einen Bezug zum Landeskoordinatensystem zu erhalten. Die Verknüpfungspunkte wurden als Kontrolle für die Georeferenzierung ebenfalls mit der Totalstation polar eingemessen. Dann folgten als nächster Schritt die Messungen mit dem Laserscanner. Der Scanner wurde direkt über das erste Fähnchen aufgestellt. Sobald die Messung eines Scans abgeschlossen war, wurde das jeweilige Signalfähnchen entfernt und der Scanner konnte über dem nächsten Fähnchen aufgestellt werden. Insgesamt wurden für das Aufnahmegebiet in Lägeren 24 Standpunkte (13 im westlichen, 11 im östlichen Teil) und für das Aufnahmegebiet in Tharandt 33 Standpunkte für den Terrestrischen Laserscanner

benötigt. Alle Aufnahmen wurden mit der höchsten Auflösungsstufe durchgeführt (Stufe „super-high“). Dabei dauerte der reine Messvorgang pro Scan ca. 6 min. Es fielen pro Scan ca. 170 Millionen Punkte an. Als letzten Schritt wurden alle verwendeten Geräte und Utensilien abgebaut.

Der gesamte Aufnahmevorgang erforderte bei Lägeren, sowie bei Tharandt mehrere Tage. Dabei spielte auch das Wetter eine Rolle. Es sollte für den Messvorgang möglichst windstill sein. Dies war nicht immer der Fall, vor allem im Messgebiet bei Tharandt. Das wird sich auf die Lage der Objekte während der Messung auswirken und eine geometrisch korrekte Darstellung der Objekte in der Punktwolke erschweren. An einem Arbeitstag war mit einer durchschnittlichen Messleistung von 15 Standpunkten zu rechnen.

4.4 Georeferenzierung

Die Georeferenzierung erfolgt nach der Datenaufnahme im Büro. Dabei werden als Erstes die Polygonzüge und darauf die Koordinaten der Hilfspunkte und somit der Passpunkte im Landeskoordinatensystem berechnet. Für die aufgenommenen Scans in Lägeren wird das Schweizer Landeskoordinatensystem verwendet und für Tharandt das Deutsche Gauß-Krüger-Koordinatensystem. Die Daten der Scans liegen nach der Entnahme aus dem Scanner im firmeninternen zfs-Dateiformat vor. Für die Georeferenzierung der aufgenommenen Scans muss das firmenzugehörige Programm z+f-LaserControl verwendet werden. Dieses Programm ermöglicht dem Nutzer, die Daten abzubilden und zu analysieren. Der einzelne Scan eines Standpunktes wird als Plattkarte abgebildet, also als 2-dimensionales Bild, dessen Bildachsen Horizontal- und Vertikalwinkel des Aufnahmerasters bilden. Die Plattkarte bildet das lokale Polarkoordinatensystem des Scans. Jedes Pixel der Plattkarte repräsentiert einen gemessenen Punkt des Scans, wobei die Farbinformation die vom Scanner gemessene Entfernung oder Intensität sein kann. Unter der Intensität versteht man die Menge der Strahlungsenergie, die vom Scanner aufgenommen wird. Diese ist stark von der Oberflächenrauigkeit und der Entfernung des Objekts und dem Auftreffwinkel des Laserstrahls abhängig. Üblicherweise wird die Plattkarte als Intensitätsbilds dargestellt und den Pixeln werden Grauwerte zugewiesen. Die Intensität lässt das Bild wie eine Photographie aussehen, deren aufgenommene Objekte vom Standpunkt aus beleuchtet werden (ähnlich einer Blitzlichtaufnahme) (Abbildung 8). Objekte, wie die Äste der Bäume sind dadurch gut erkennbar.

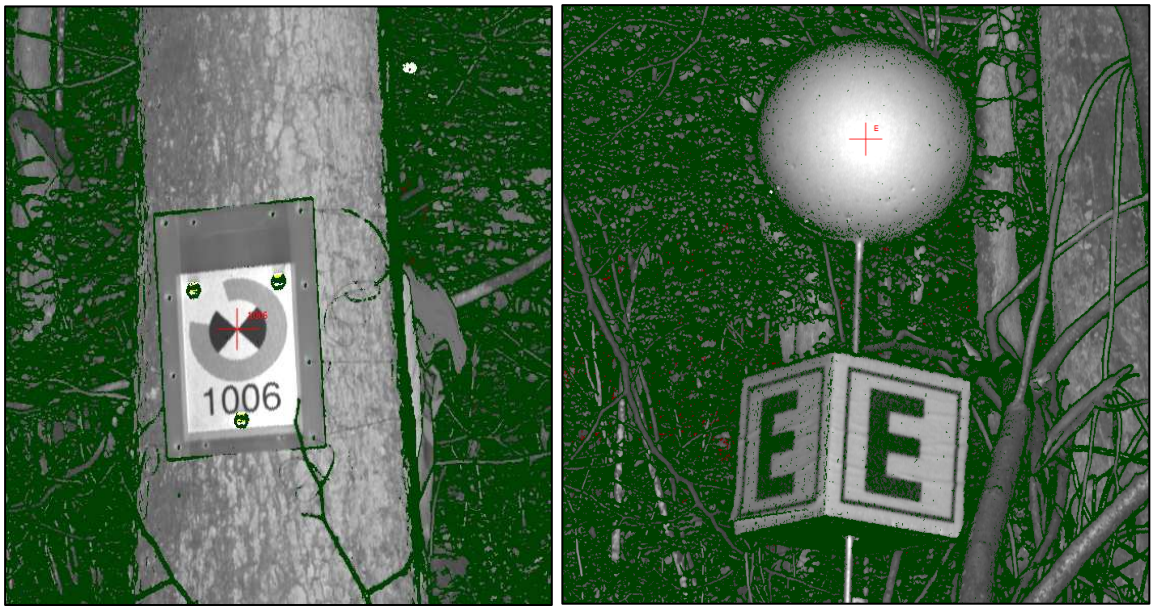

Abbildung 8 : Zieltafeln und Verknüpfungskugeln im Intensitätsbild des Scans

Im Intensitätsbild sind die Zieltafeln und auch Verknüpfungskugeln sichtbar und können damit im lokalen Koordinatensystem gemessen werden (Abbildung 8). Durch eine Helmerttransformation werden nun die Transformationsdaten des Scans zum Landeskoordinatensystem in Form einer räumlichen Ähnlichkeitstransformation berechnet. Die Transformation ist eigentlich eine räumliche Kongruenztransformation, da der Maßstab auf 1 gesetzt wird. Dies wird bei jedem Scan durchgeführt. Im Rahmen einer Gesamtausgleichung werden diese dann nochmals neu berechnet. Die Transformationsdaten bestehen aus einer orthogonalen Rotationsmatrix (3 Parameter) und einem Translationsvektor (3 Parameter).

$$x_I = R \cdot x_{LK} + t$$

$$\begin{pmatrix} x_{LK} \\ y_{LK} \\ z_{LK} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{12} & r_{22} & r_{23} \\ r_{13} & r_{23} & r_{33} \end{pmatrix} \cdot \begin{pmatrix} x_I \\ y_I \\ z_I \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \qquad \text{(Gl. 5)}$$

r_{xx} ..... Rotationselemente t_x .....Translationselemente

Dabei muss davor das Polarkoordinatensystem der Plattkarte in ein lokales, kartesisches Koordinatensystem umgewandelt werden:

$$\begin{pmatrix} x_I \\ y_I \\ z_I \end{pmatrix} = D_I \cdot \begin{pmatrix} \sin(H_I) \cdot \sin(V_I) \\ \cos(H_I) \cdot \sin(V_I) \\ \cos(V_I) \end{pmatrix}$$

D_I .....Entfernung
H_I ..... Horizontalwinkel
V_I .....Vertikalwinkel

(Gl. 6)

5 Stand der Technik

Die Modellierung der Struktur von Bäumen aus einer TLS-Punktwolke ist eine Thematik, zu der schon seit einigen Jahren eine Reihe von Publikationen erschienen ist. Einige Publikationen bieten verschiedene Wege, die Modellierung zu automatisieren. Prinzipiell werden alle Methoden in mehreren Schritten durchgeführt. Diese können im Vergleich verschiedene Ansätze, aber auch Gemeinsamkeiten beinhalten. Insofern ist eine gegenseitige Beeinflussung wichtig zur Findung einer bestmöglichen Methode. Alle Methoden basieren auf der Bearbeitung der bereits georeferenzierten Punktwolke. Ein terrestrischer Laserscanner erzeugt in mehreren Standpunkten Punkte von der Oberfläche der Bäume. Alle Punkte des im Raum befindlichen Baumes aus diesen Standpunkten sollen auch gleichzeitig verarbeitet werden, auch wenn es durch Windeinflüsse zu starken Veränderungen der Objekte kommen kann. Basis einiger Überlegungen ist die Unterteilung des Raums, in dem sich die Punktwolke befindet, in genormte Zellen. Diese Zellen werden Voxel genannt. Voxel sind das 3-dimensionale Pendant zu den in der 2-dimensionalen Bildverarbeitung bezeichneten Pixeln. Sie haben alle zueinander die gleiche Ausdehnung, welche je nach Bedarf gewählt werden kann. Sie sind damit homogen. Die Ausdehnung kann aber in unterschiedlichen Raumrichtungen unterschiedlich sein. Sie müssen also nicht isotrop sein. So finden sich gleich 3 interessante Methoden, welche Voxel zur Bestimmung eines mathematischen Modells verwenden. So verwendet Buksch [6] in seinem sogenannten SkelTre-Algorithmus Voxel zur Einteilung der Punktwolke in Punktgruppen, um aus diesen lokale Schwerpunkte zu berechnen. Diese Schwerpunkte werden miteinander verbunden, wobei deutlich mehr Verbindungen erstellt werden, als für eine Struktur notwendig. Daher werden diese auf wenige Verbindungen reduziert, bis nur ein einfaches Linienmodell zurückbleibt. Auch Pfeifer et al. [7] verwenden für eine Methode, welche die Schaffung eines Modells durch Zylinderanpassung der Punktwolke zum Ziel hat, Voxel in der Verarbeitung der Messpunkte. Dabei ist der wichtigste Schritt weniger die Anpassung durch Zylinder als die Segmentierung der Punktwolke. Das bedeutet, dass die aufgenommenen Bäume in ihre Astsegmente aufgeteilt, also segmentiert werden. Erst diese einzelnen Segmente können für eine Zylinderanpassung verwendet werden, wenn automatisiert werden soll.

Schilling et al. [8] verwenden die Voxel-Einteilung des Raumes um die Punktwolke in horizontale Schichten einzuteilen, die wie 2-dimensionale Bilder verarbeitet werden können.

Diese Bilder können mit den gängigen 2D-Bildverarbeitungsalgorithmen bearbeitet werden. So entstehen auf den Bildern zwangsläufig kreis- bzw. ellipsenförmige Strukturen, wie sie bei horizontalen Schnitten durch Stämme und Äste der Bäume zu erwarten sind. Diese können mit Hilfe der Kreis-Hough-Transformation detektiert werden und somit der Kreismittelpunkt ermittelt werden. Aus diesen kann die Achslinienstruktur der Äste bestimmt werden.

Ein weiterer wichtiger Ansatz in bisherigen Methoden ist die bereits erwähnte Segmentierung der Punktwolke. Ein Segment stellt jene Teilmenge der Wolke dar, welche einem einfacheren, geometrischen Objekt zugeordnet werden kann. Dabei handelt es sich in diesem Fall um Abschnitte der Äste, welche keine Gabelung aufweisen. Diese Segmentierung ist Basis zur Bestimmung eines Modells in der Methode von Bremer et al. [9]. Die Segmentierung wird hier mit Hilfe des Oberflächenwachstumsverfahrens (Surface Growing Algorithm) [10] durchgeführt und kann entscheidend beeinflusst werden bezüglich der Strenge ihrer Bestimmung. Aus den Segmenten wird hier mit Hilfe der Eigenvektorberechnung die geometrische Form abgeleitet, welche die Achse und auch Ausdehnung der Astsegmente festlegt. Gleichzeitig können dadurch auch Blätter gefiltert werden, da diese eine andere, in der Regel flachere Form aufweisen. Die Segmentierung wird oft auch mit Hilfe der Voxel-Einteilung durchgeführt, so wie in der bereits erwähnten Methode von Pfeifer et al. [7].

5.1 Methode nach Buksch

Buksch [6] präsentiert eine automatisierte Methode zur Extrahierung der Struktur von Bäumen aus einer georeferenzierten Punktwolke, welche mit Terrestrischem Laserscanning aufgenommen worden ist. Er nennt sie den SkelTre-Algorithmus ([6], S. 37-52).

Diese Methode verläuft in 3 Schritten([6], S. 38):

1. Erstellung von Octree-Graphen aus einer Octree-Organisation (Voxeleinteilung)
2. Reduktion des Octree-Graphen zu einem Sklelett-Graphen
3. Einbettung des Skelett-Graphen in die Punktwolke

Im ersten Schritt wird der Raum, in der sich die georeferenzierte Punktwolke befindet, in würfelförmige Zellen eingeteilt. Diese Unterteilung erfolgt durch Bildung von „Octrees“ („Achterbaum“). Dieser Octree entsteht im 3-dimensionalen Raum durch Unterteilung eines

Würfels in 8 weitere Würfel, die bei der Halbierung der Seitenkanten des Würfels entstehen. Diese Würfelzellen entsprechen den bereits erwähnten Voxeln. Der Raum wird in $2^n \times 2^n \times 2^n$ Zellen aufgeteilt, wobei n die Unterteilungstiefe ist. Die Unterteilungstiefe ist ein Parameter, der frei gewählt werden kann, aber Einfluss auf das Ergebnis hat. Es hängt stark von der Auflösung der Laserscanning-Aufnahme ab.

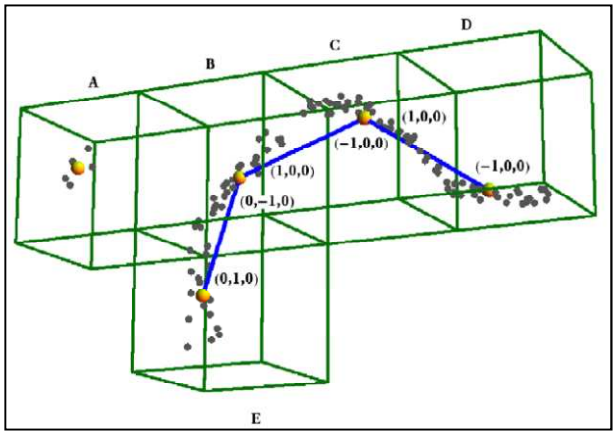

Abbildung 9 : Erstellung eines Octree-Graphen [6]

Für jede Zelle wird der Schwerpunkt aller in der jeweiligen Zelle befindlichen Punkte berechnet. Zellen, welche keine Punkte beinhalten, werden ausgelassen. Benachbarte Schwerpunkte werden schließlich miteinander verbunden und bilden damit die Knotenpunkte des Octree-Graphen. Es sollen aber nicht alle benachbarten Schwerpunkte miteinander verbunden werden, da Ausreißer, Fehlaufnahmen und Lücken nicht durch Linien repräsentiert werden sollen. Daher wird ein Robustheitskriterium ([6], S. 40) definiert, welches diese unerwünschten Verbindungen verhindert.

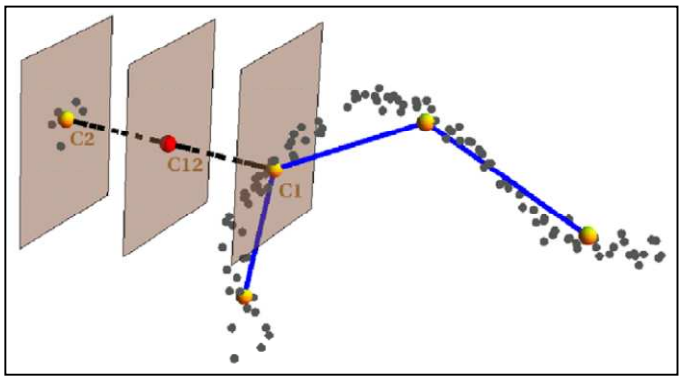

Abbildung 10 : Robustheitskriterium für das Verbinden von Schwerpunkten [6]

Dieses Robustheitskriterium stellt auf mathematisch einfachem Wege fest, ob sich zwischen zwei Schwerpunkten Messpunkte gleichmäßig verbreitet vorfinden. Es besteht die Annahme, dass diese einen Ast bilden.

Die mathematische Definition ist durch Abbildung 10 beschrieben: Mögen C_1 und C_2 die beiden zu verbindenden Schwerpunkte sein und C_{12} der Halbierungspunkt der Verbindungslinie, so sind P_1, P_2 und P_{12} die zur Verbindungslinie orthonormalen Ebenen durch diese Punkte. Darauf wird der Median der quadrierten Normalabstände der Punkte von der Ebene berechnet. Dabei ist d_1 der Median für die Punkte in der Zelle von C_1, d_2 für die Punkte in der Zelle von C_2 und d_{12} für die Punkte in beiden Zellen zusammen.

Erfüllen nun die 3 Medianwerte das Kriterium

$$\frac{1}{16} d_{12} \leq \min(d_1, d_2), \qquad \text{(Gl. 7)}$$

so wird eine Verbindung zwischen den 2 Schwerpunkten hergestellt.

Der zweite Schritt im SkelTre-Algorithmus ist die Reduktion des erstellten Octree-Graphen zu dem gesuchten Skelett-Graphen ([6], S. 41-50). Die Verbindungen der Knotenpunkte im Octree-Graphen bilden auf Grund der kleinen Größe der Octree-Zellen ein gitterförmiges Linienkonstrukt, welches sich durch die Punktwolke der aufgenommenen Äste und Stämme der Bäume zieht. Gesucht ist aber ein Linienmodell, welches für jeden Ast und jeden Stamm genau eine Linie entlang der Achse besitzt. Im Octree-Graphen wird diese Mittellinie durch annähernd parallel verlaufende Linien repräsentiert, dessen Knotenpunkte oft miteinander verbunden sind. Diese Linien sollen auf diese Mittellinie vereinheitlicht, also reduziert werden. Dies geschieht ohne Verwendung der Punktwolke allein auf Grund festgesetzter Regeln zur Zusammenlegung von Knotenpunkten. Allerdings werden den Knotenpunkten Gewichte zugewiesen, welche der Anzahl der Punkte, die sich in der jeweiligen Zelle befinden, entsprechen.

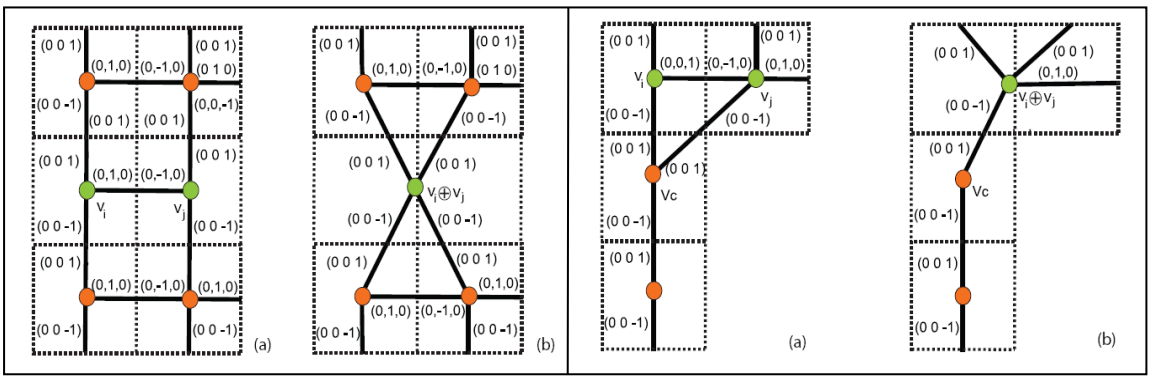

Abbildung 11 : Vereinigung von Knotenpunkten zur Reduktion zum Skelett-Graphen (Koordinaten V_i, V_j)
→ (a) Auswahl potentieller Knotenpunkte (grün) (b) Reduktion der Knotenpunkte (grün) [6]

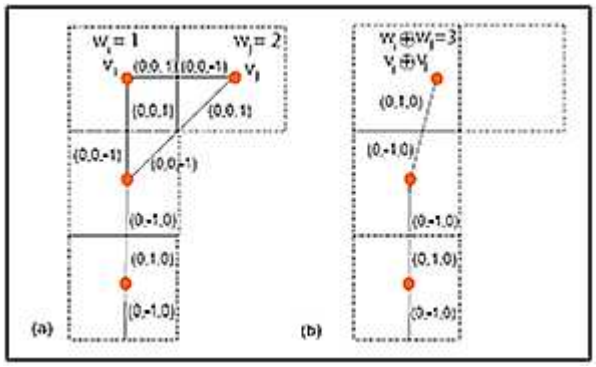

Abbildung 12 :

Vereinigung von Knotenpunkte (Koordinaten V_i, V_j) zur Reduktion zum Skelettgraphen bei Berücksichtigung der Gewichte (hier: W_i, W_j) [6]

So sind die Anzahl und die Richtung der Kanten, welche in einen Knotenpunkt münden, so wie die Höhe der Gewichte für die Vereinigung bzw. Zusammenlegung von Knotenpunkten entscheidend. Die durch die Vereinigung der Knotenpunkte erstellten neuen Knotenpunkte befinden sich mittig zwischen den Knotenpunkte (Abbildung 11). Die Berechnung der Koordinate wird durch die Gewichte stark beeinflusst (Abbildung 12):

$$\left(v_i \otimes v_j\right) = \frac{\omega_i \cdot v_i + \omega_j \cdot v_j}{\omega_i + \omega_j} \qquad \text{(Gl. 8)} \qquad \text{([6], S. 50)}$$

Der neue, vereinigte Knotenpunkt erhält als Gewicht die Summe der Gewichte der beiden vorangegangen Knotenpunkte.

Der dritte Schritt des SkelTre-Algorithmus beinhaltet die Einbettung des reduzierten Skelettgraphen in die Punktwolke. Der Skelett-Graph repräsentiert die gesuchte Linienstruktur nur annähernd. Durch unterschiedliche Gewichte der Schwerpunkte bei der Reduktion, kommt es zu Abweichungen der Linien vom Ast- bzw. Stammzentrum. Dies wird durch unregelmäßige Punkdichten an den Asträndern verursacht ([6], S. 50).

Es muss daher die Punktwolke dazu verwendet werden, die Linien in die richtige Position zu bringen. Gleichzeitig wird sie dazu verwendet, den Durchmesser des jeweiligen Astes bzw. Stammes zu bestimmen, den die jeweiligen Linien repräsentieren. ([6], S. 70-81)

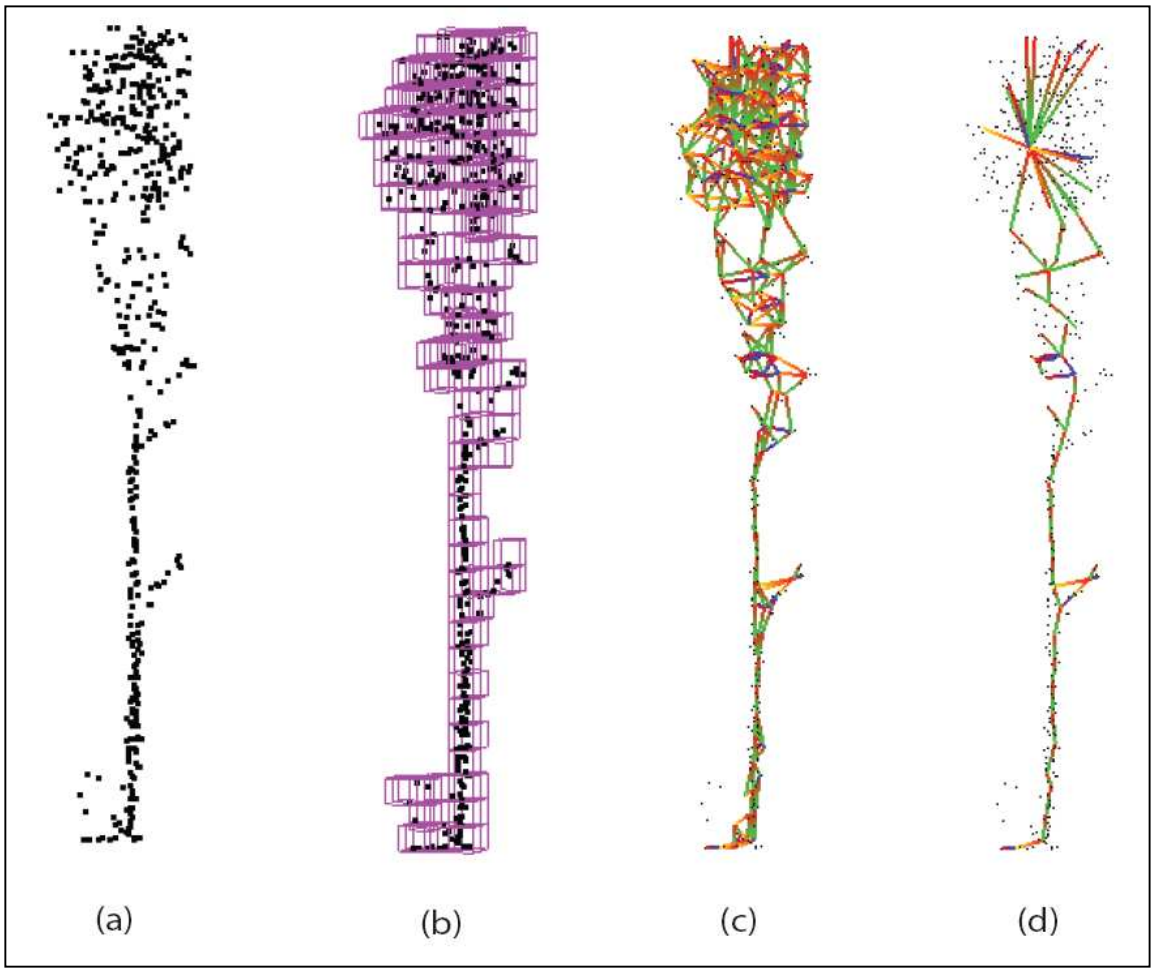

Abbildung 13 : (a) Punktwolke des Baumes, (b) Oktree-Organisation (Voxeleinteilung), (c) extrahierter Oktreegraph, (d) reduzierter Skelettgraph [6]

Der Ablauf des SkelTre-Algorthmus ist bildlich in Abbildung 13 dargestellt. Vor allem bei einer nur geringen Anzahl an Punkten sind die Fehlleistungen dieser Methode der Automatisierung gut zu sehen. Abbildung 13 (c) zeigt viele fehlerhafte Linien des Octree-Graphen. Dies wird noch verstärkt in der Skelettgraphen-Darstellung in Abbildung 13 (d). Hier sind in der Baumkrone praktisch keine echten Strukturen mehr erkennbar, auch wenn diese durch Begutachtung mit freiem Auge kaum erkennbar sind. Nur am Stamm tauchen Abweichungen von der Struktur auf.

5.2 Methode nach Pfeifer et al.

Die Methode von Pfeifer et al. [7] erzeugt Baummodelle in Form von Zylinderanpassung. Die durch das Laserscanning erzeugten Punkte sind an Bäumen nur an der Astoberfläche zu finden und bilden dadurch röhrenförmige Gebilde. Bei der Annahme, dass die Äste rund sind und in ihren Querschnitten Kreisscheiben bilden, können diese annähernd durch eine Aneinanderreihung von Kreiszylindern repräsentiert werden. Ein Kreiszylinder ist ein geometrisches Objekt, das durch die Parameter Achspunkt P und Raumvektor a, um die Drehachse zu beschreiben, und den Kreisradius r definiert wird. Q_i sind die Punkte auf der Zylinderfläche.

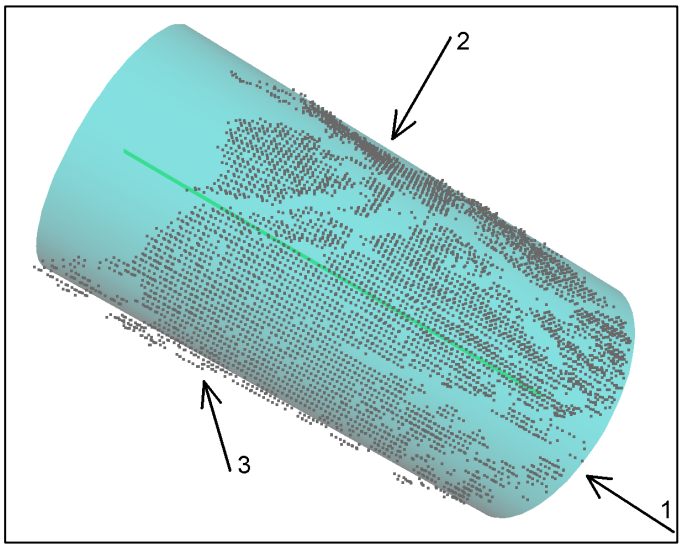

Abbildung 14 : Anpassung eines Zylinders an die Punktwolke [7]

$$\left\| \left(P - Q_i \right) \times a \right\| - r = 0 \qquad \text{(Gl. 9)}$$

Die Anpassung des Zylinders (Abbildung 14) an die Punktwolke erfolgt mittels Kleinste-Quadrate-Ausgleichung. Die Residuen v_i sind die Abstände der Punkte von der Zylinderfläche.

$$v_i = \left\| \left(P - Q_i \right) \times a \right\| - r \qquad \text{(Gl. 10)}$$

Da bei der Ausgleichung eine Linearisierung durchgeführt wird, sind Näherungswerte für die Zylinderparameter erforderlich. Diese können dadurch gefunden werden, indem ein bereits angepasster Zylinder um eine gewisse Strecke entlang seiner Achse verschoben wird. Diese Strecke kann frei gewählt werden, wird aber vom Durchmesser des Astes abhängig sein. Bei Betrachtung der Baumstruktur sind Äste mit kleinem Durchmesser kurvenreicher als Äste mit großem Durchmesser. Diese Abhängigkeit betrifft auch die Länge des Zylinders. Somit kann ein Ast durch eine Aneinanderreihung von Drehzylindern automatisch erfasst werden. Dabei bleiben zwei Probleme aufrecht. Erstens kann die Aneinanderreihung mittels

Zylinderverschiebung nur entlang eines Astes durchgeführt werden. Verzweigungen werden ignoriert und müssen getrennt voneinander als eigene Äste behandelt werden. Dabei sollen auch nur die Punkte des Astes zur Zylinderanpassung verwendet werden. Zweitens sind die Näherungswerte des ersten Zylinders eines Astsegmentes erforderlich, für deren Bestimmung ein automatisches Verfahren anzustreben ist. Daher wird die Punktwolke in die einzelnen Äste aufgeteilt bzw. segmentiert. Zusätzlich ist eine Skeletonisierung, also die Schaffung eines Linienmodells, hilfreich. Die Linien eines solchen Modells können den Drehzylindern die verlangten, genäherten Parameter automatisch beschaffen. Die Segmentierung und Skeletonisierung wird mit Hilfe einer Voxel-Aufteilung durchgeführt, wie in Gorte et al. [11] vorgeschlagen. Allerdings werden keine Schwerpunkte berechnet, sondern die Zellen werden nur danach unterschieden, ob sie Punkte beinhalten oder nicht. Zwangsläufig werden sich, abhängig von der Wahl der Voxelgröße, Lücken im Voxelraum befinden, welche sich in erster Linie im Stamm und den größeren Ästen befinden, da diese in der Punktwolke hohl sind. Da ein Skelettgraph extrahiert werden soll, müssen diese Hohlräume mittels Nachbarschaftsalgorithmen gefüllt werden. Die Reduktion der Voxeln auf den Skelettgraphen wird anhand der Methode von Palágyi et al. [12] durchgeführt. Dabei werden die Äste und Stämme an der Oberfläche „geschält“, also die äußeren Voxeln entfernt, bis nur mehr eine einfach verbundene Voxelreihe übrigbleibt, bei der jedes Voxel nur höchstens vier Nachbarvoxel haben kann. Durch Verbindung durch Kanten wird daraus ein Skelettgraph gebildet. Den Kanten werden Gewichte zugewiesen, welche vom Abstand zum untersten Stammvoxel abhängig sind (Abbildung 15).

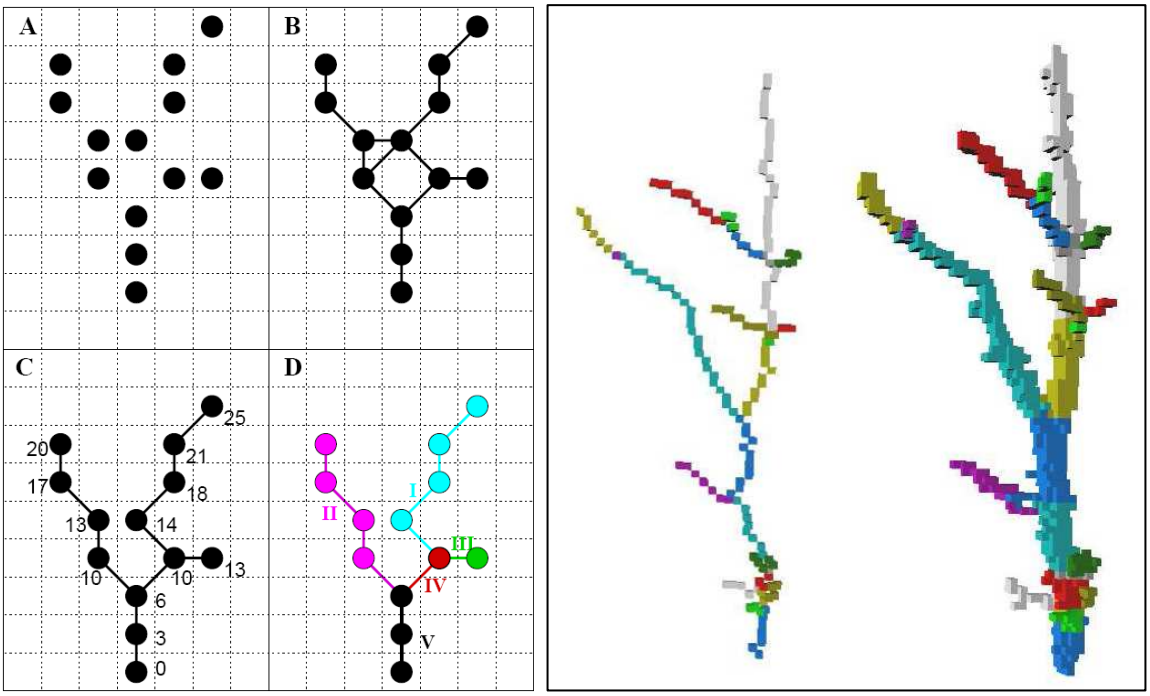

Abbildung 15 : **2D-Illustriation der Segmentierung [11]**
A Voxelskelett
B Skelettgraph
C Suche nach kürzesten Wegen
D Segmente

Segmentierung
inks: segmentierter Skelettgraph
rechts: segmentierter Voxelraum

Dieser Skelettgraph kann nun einer Segmentierung unterzogen werden. Dafür eignet sich der von Paláyi et al. [12] vorgeschlagene Dijkstra-Algorithmus. Dieser Algorithmus sucht den kürzesten Weg in einem gewichteten Graphensystem. In dieser Methode wird vom Stamm aus immer der kürzeste Weg zur Spitze eines jeden Astes des Baumes gesucht und diese registriert. Aus den Überschneidungen der gefundenen Wege können die einzelnen Äste segmentiert und gekennzeichnet werden. Nach diesen Skelettsegmenten wird die Punktwolke segmentiert (Abbildung 15). Somit können die Segmente getrennt voneinander für die Zylinderanpassung verwendet werden, wobei für die Anfangszylinder Näherungswerte vorhanden sind. Somit ist eine automatische Bestimmung des Zylindermodells möglich. Das Ergebnis ist im besten Fall ein Zylindermodell, dessen Elemente den gesamten Baum und dessen Äste repräsentiert. Das Ergebnis eines Testlaufs ist in Abbildung 16 zu sehen.

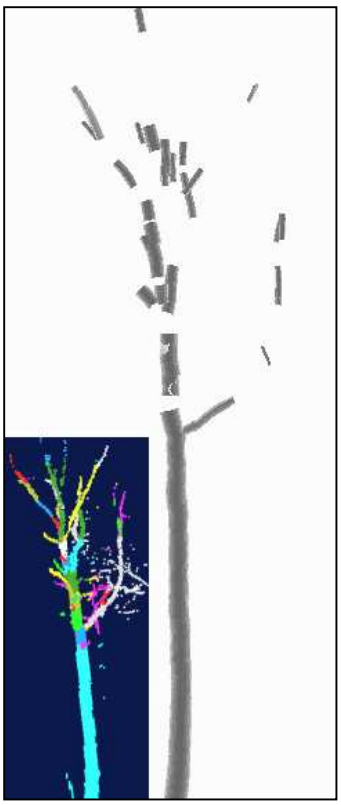

Abbildung 16 : links unten: automatisierte Segmentierung
großes Bild: Zylinderdarstellung des Baumes
(Pfeifer et al. 2004 [7])

Die Abbildung 16 stellt einen Baum dar, der mit vier Scans von allen Seiten aufgenommen wurde, und besteht in Folge aus 2,4 Millionen Punkten. Die Methode der Zylinderanpassung dürfte trotz der vielen Punkte massive Probleme bezüglich der Vollständigkeit aufweisen. So sind die vielen, in der Punktwolke klar erkennbaren Äste oft gar nicht durch Zylinder dargestellt. Dies gilt vor allem für die kleineren Äste. Größere, dicke Äste werden erkannt, haben aber auch trotzdem viele Lücken. Einzig der Stamm wird gut durch diese Methode dargestellt. Offensichtlich hängt die Funktion stark von der Punktdichte ab. Die Vollständigkeit der Oberflächen der Äste in der Punktwolke spielt eine starke Rolle. Ob daher das „Füllen“ der Äste gelingt, ist eine weitere Frage. Angesichts dieser Probleme könnte eine Anwendung dieser Methode an mit Blättern vollbesetzten Laubbäumen nicht genügend sein. Bei Nadelbäumen wird abgesehen vom Stamm bei einer Zylinderanpassung wenig übrigbleiben.

5.3 Methode nach Bremer et al.

Bremer et al. [9] präsentieren einen weiteren Ansatz zur Erstellung einer Linienstruktur von Bäumen aus einer Punktwolke. Dieser erfolgt in mehreren Schritten, die an Hand einer Darstellung des Arbeitsflusses beschrieben sind (Abbildung 18) ([9], S. 37).

Als erster Schritt erfolgt eine Vorstrukturierung der Punktwolke in einzelne Segmente. Die Segmente sind Teilmengen der Punktwolke, welche einfachere geometrische Formen am Aufnahmeobjekt bilden. Diese werden mittels eines Oberflächenwachstumsverfahrens (Surface Growing Algorithm) bestimmt [10]. Dieses Verfahren sucht Punktemengen, deren Punkte untereinander ein vordefiniertes Homogenitätskriterium erfüllen, z.B. Zugehörigkeit zur selben Ebene oder Zylinderfläche).

Im zweiten Schritt wird für jedes Segment ein Tensor berechnet, der die Punktkoordinaten eines jeden Punktes des Segments auf dessen Schwerpunkt reduziert. Der Tensor wird aus der Multiplikation $A^T A$ berechnet.

$$A = \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} & z_1 - \bar{z} \\ \vdots & \vdots & \vdots \\ x_k & y_k & z_k \end{bmatrix}$$

$\bar{x}, \bar{y}, \bar{z}$ ..... Koordinaten des Schwerpunkts
x_1, y_1, z_1 ..... Punkte
k ..... Anzahl der Punkte

(Gl. 11)

Das jeweilige Segment hat durch die mathematische Definition als Tensor die Form eines Quaders. Durch Berechnung der Eigenvektoren kann die Form des Quaders charakterisiert werden. Diese Charakterisierung ermöglicht die Erkennung und damit die Filterung der Äste und Stämme des Baumes (Abbildung 17). So werden lange Quader mit nahezu quadratischem Querschnitt definitionsgemäß die Segmente eines Astes abbilden, während flache, breite Quader Blätter abbilden ([9], S. 37). Für jedes dieser Astsegmente kann nun die Achsenlinie aus dem jeweils größten Eigenvektor bestimmt werden.

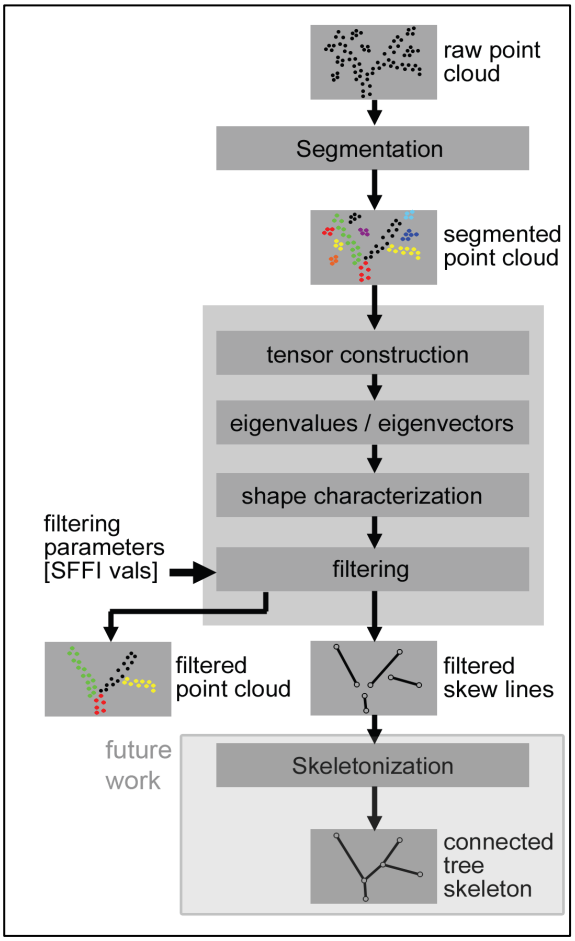

Abbildung 18 : Arbeitsfluss zur Er-stellung der Linienstruktur ([9], S. 37)

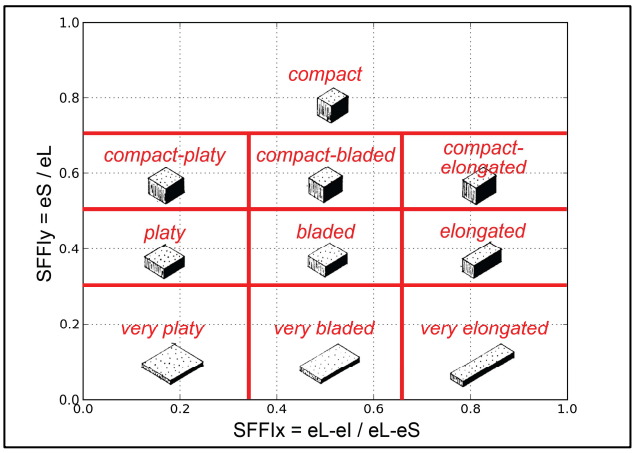

Abbildung 17 : Charakterisierung der Segmente ([8], S. 38)

Die Länge der Linie entspricht der Länge der längsten Quaderseite. Die so entstehenden Linien werden schließlich miteinander verbunden, um eine geschlossene Linienstruktur des Baumes zu erhalten ([9], S. 37). Abbildung 19 stellt einen Baum als segmentierte Punktwolke und als fertiges Linienmodell dar, sowohl mit Blättern als auch ohne Blätter. Die Punktwolke

stellt den Baum nahezu komplett dar ohne besondere Abschattungen, die in der Praxis oft vorkommen. Die Abbildung zeigt beispielhaft die Fähigkeit der Filterung von Blattwerk. Die Stärke der Filterung ist je nach Parameter (Abbildung 17) veränderbar, um ein gewünschtes Ergebnis zu erzielen. Das Ergebnis gibt eine sehr umfangreiche und möglicherweise fast vollständige Darstellung des Baumes als Linienmodell wieder. Selbst sehr kleine Äste werden dargestellt. Über genaue Positionen der kleinen Äste wäre eine genauere Analyse notwendig. Diese Methode dürfte sehr effizient mit der vorhandenen Information umgehen. Diese könnte bei Waldgebieten nützlich sein, um möglichst viel von den Baumkronen zu bekommen. Allerdings stellt diese Methode nur Wahrscheinlichkeiten in den Raum, welche Linien des Modells tatsächliche Äste repräsentieren. Insofern ist die Segmentierung entscheidend für die semantische Richtigkeit der Linien im Modell.

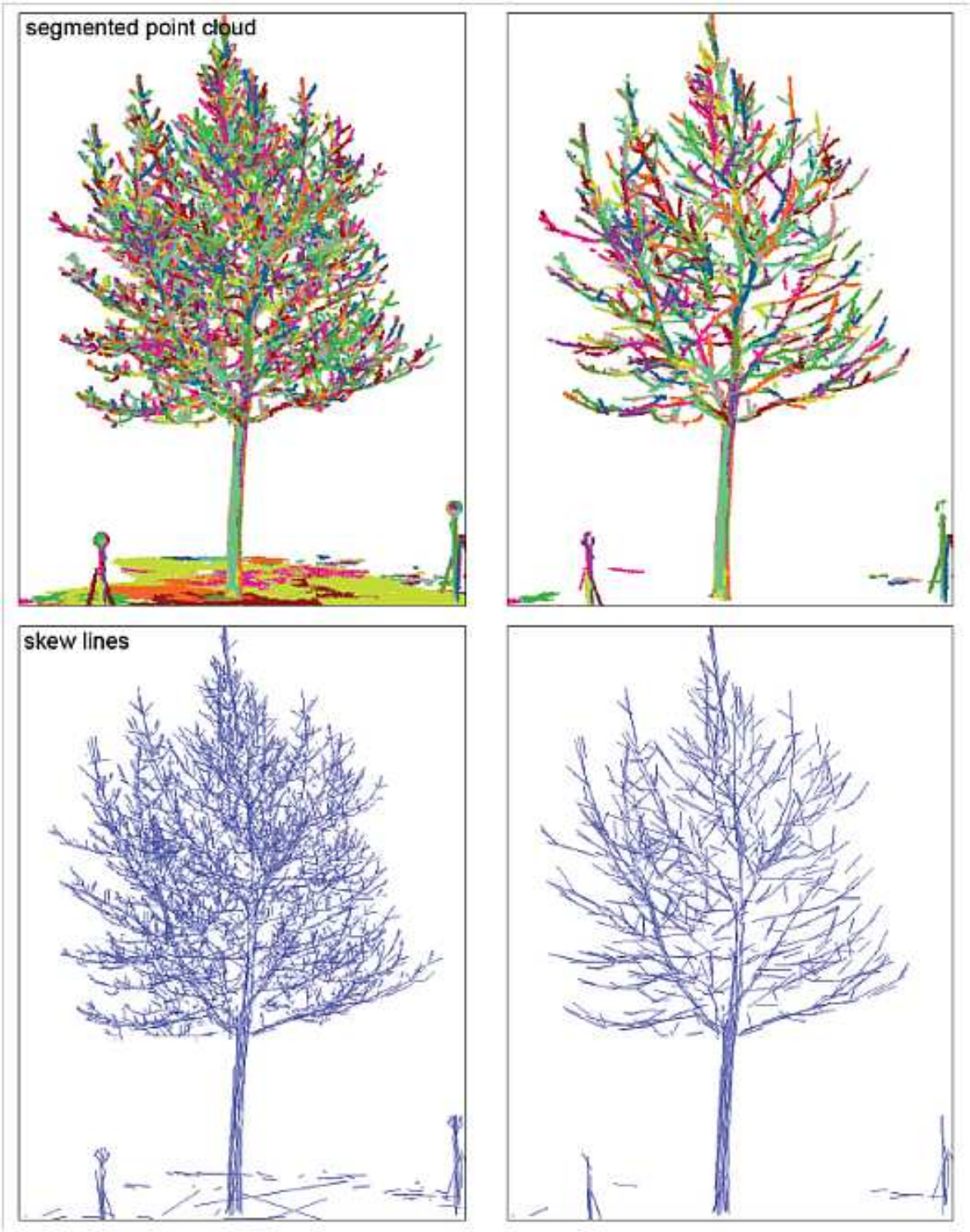

Abbildung 19 : **Darstellung eines Baumes mit Blättern (links) und ohne Blätter (rechts) als segmentierte Punktwolke (oben) und als fertiges Linienmodell (unten) [9]**

5.4 Methode nach Schilling et al.

Bei der in Schilling et al. [8] beschriebenen Methode wird der Raum, in der sich die Punktewolke befindet, wie in der Methode von Pfeifer et al. [7] (siehe Kapitel 5.2) in gleichförmige Voxel eingeteilt.

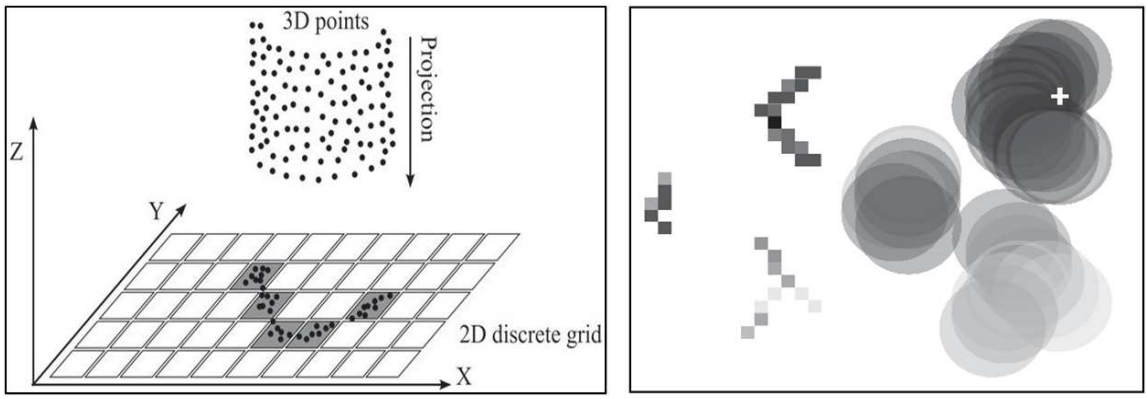

Abbildung 20 : links: Projektion der Punkte auf das 2-dimensionale Bild [8]
rechts: Kreisscheiben mit Punkt höchster Dichte als Kreismittelpunkt [8]

Dabei werden die Höhenschichten der Voxel je nach festgelegter Dicke zu einem 2-dimensionalen Bild zusammengefasst. Alle Punkte in der jeweiligen Höhenschicht werden auf das jeweilige 2-dimensionale Bild projiziert (Abbildung 20, links). Jedes dieser Bilder entspricht einem horizontalen Schnitt durch die Punktwolke. Bei der Aufnahme eines Waldstücks oder eines einzelnen Baumes werden sich auf den Bildern zwangsläufig Kreise und Ellipsen befinden, die den senkrecht wachsenden Stämmen und Ästen zuzuweisen sind. Mit Hilfe der Hough-Transformation lassen sich in den 2-dimensionalen Pixelbildern Kreise grob detektieren. Jedem Pixel eines detektierten Kreises wird nun eine gewichtete Kreisscheibe zugewiesen, dessen Gewicht von der Anzahl der projizierten Punkte im Pixel abhängt (Abbildung 20, rechts). Die Kreisscheibe hat den gleichen Radius wie der detektierte Kreis. So entsteht eine Fläche von überlagerten Kreisscheiben, dessen Punkt

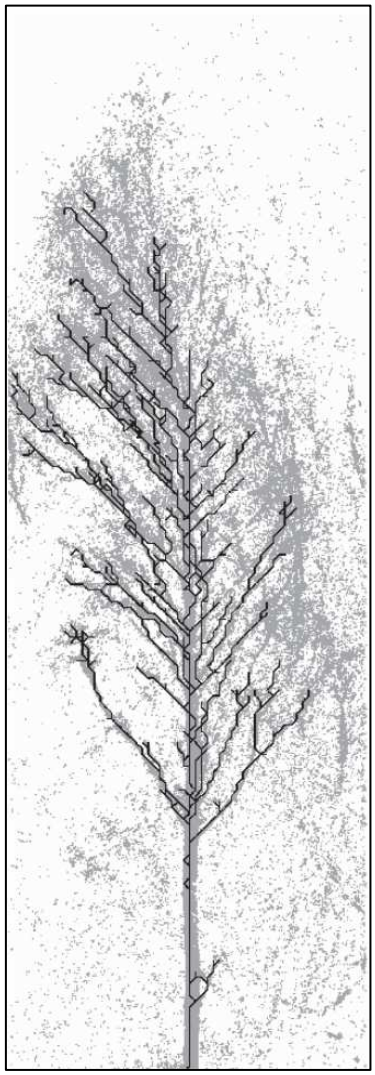

Abbildung 21 : Generiertes Linienmodell mit Punktwolke ([8], S. 6)

höchster Dichte der gesuchte Kreismittelpunkt des detektierten Kreises ist. Durch das Verbinden dieser Kreismittelpunkte zwischen den übereinanderliegenden Höhenschichten können die Achsen der durchschnittenen Äste ermittelt werden. Diese bilden zusammen das Linienmodell des Baumes (Abbildung 21). Durch die lose Verbindung dieser Kreismittelpunkte entstehen unerwünschte Verbindungen zwischen Ästen, wo real keine Äste sind, sowie doppelte Verbindungen. Mittels entsprechender Such- und Sortieralgorithmen, wie der sogenannte „Depth-First-Search“-Algorithmus ([8], S. 5) werden richtige Verbindungen detektiert und falsche Verbindungen entfernt.

6 Methode

6.1 Anforderungen an die Baummodellierung

Eine wichtige Aufgabe im Rahmen des Projekts „3DVegLab“ ist, eine schnelle, geeignete Methode zu finden, ein Linienmodell aus der Punktwolke eines jeden Standpunktes zu extrahieren. Eine wichtige Überlegung ist, ob dieses Linienmodell manuell gezeichnet oder automatisiert bestimmt werden soll. In Kapitel 5 wurden bereits aktuelle Methoden aus anderen Publikationen präsentiert, die in erster Linie eine automatisierte Bestimmung der Baumstruktur als Ziel hatten.

Dabei sind vier wichtige Eigenschaften des Modells zu erzielen:

- Vollständigkeit
- Geometrische Richtigkeit
- Genauigkeit
- Zeitdauer

Wichtig ist die Anforderung bezüglich der Zeitdauer. Manuelle Modellierung benötigt im Gegensatz zur Automation um ein Vielfaches mehr Zeit. Diese ist im Rahmen des Forschungsprojekts „3DVegLab“ vorhanden, da nur einige, wenige Waldgebiete von geringer Ausdehnung für die Modellierung ausgewählt worden sind. Damit können die Vorteile menschlicher Interpretationsgabe voll ausgespielt werden. Die Vollständigkeit des Modells ist eine Eigenschaft, bei der automatisierte Methoden gravierende Nachteile gegenüber einer manuellen Modellierung haben. Bäume sind Objekte, die sehr unterschiedliche Größenverhältnisse in ihrer Struktur aufweisen. So soll ein wenige Millimeter dickes Ästchen genauso bestimmt werden wie der mehrere Dezimeter dicke Stamm. Dies stellt die zur Automation benötigte Diskretisierung vor Probleme. Für rechnerische Zwecke ist eine einheitliche, homogene Größe eines Voxels wünschenswert. Doch erfordert das Funktionieren von Algorithmen eine angepasste Größe an das jeweilige Objekt. Um die Hohlräume mittels Nachbarschaftsalgorithmen zu füllen (Kapitel 5.2), sind für Ästchen deutlich kleinere Voxel notwendig, als für den Stamm, für den viel größere Voxel notwendig sind. Dies ist vor allem dort schwierig, wo beide Objekte nah beieinander sind. Kleine Ästchen werden oft nur durch eine Reihe Punkte gemessen, zusätzlich treten Lücken auf.

Eine Interpretation als Zylinder ist dabei nicht möglich. Lücken können hingegen durch Einbindung der Intensität überbrückbar werden. Es ist bisher keine Methode zur Baummodellierung bekannt, die Intensitäten in die Automation einbindet. Dies gilt zumindest für terrestrische Laserscanner mit Phasenvergleichsverfahren. Denn es darf nicht vergessen werden, dass Intensitäten vom Scannerstandpunkt abhängen und in einer georeferenzierten Punktwolke, die aus mehreren Standpunktdaten zusammengesetzt ist, Punkte desselben Astes in ihren Intensitäten unterschiedlich sein können. Fraglich ist auch, ob Automation auch geometrisch richtige Objekte erzeugt. Sind die erzeugten Linien an der richtigen Position und repräsentieren sie den jeweiligen Ast? Die unterschiedliche Größe der Äste kann die Linien in der Struktur in andere Richtungen leiten (Kapitel 5.1, Abbildung 13). Dazu kommt, dass Automation in bisherigen Methoden keine entsprechende Gewichtung an Objekten durchführt. Die geometrische Richtigkeit von Stämmen ist gewiss wichtiger als von kleinen aus dem Stamm herausragenden Ästchen. Dies kann durch visuelle Interpretation berücksichtigt werden. Ein schwerwiegendes Problem, das bei der Baummodellierung allgemein auftaucht, ist der Einfluss des Windes bei der TLS-Messung. Der Wind bringt besonders die obersten Äste der Baumkrone zum Schwanken und dies wirkt sich auf die richtigen, geometrischen Positionen der Messpunkte aus. Es entstehen wellenförmige Körper in der Punktwolke, aus welchen die bisherigen automatisierten Methoden keine Struktur erkennen (Abbildung 22). Durch visuelle Interpretationen ist es besser möglich, aus diesen wellenförmigen Gebilden Äste zu modellieren.

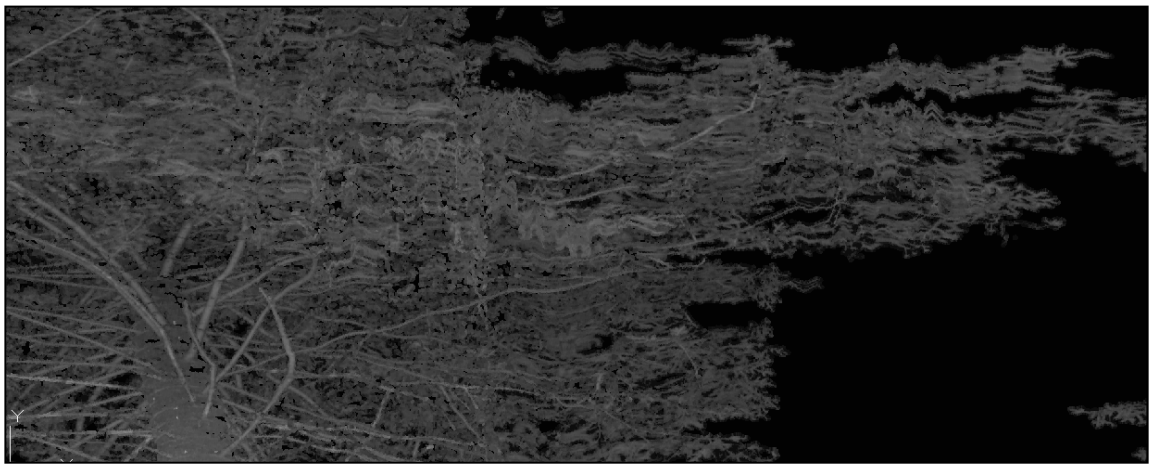

Abbildung 22 : Intensitätsbild einer Laserscanningaufnahme zeigt wellenförmige Gebilde, beeinflusst durch den Wind

6.2 Grundsätzliche Überlegungen zur Methodenentwicklung

Die Modellierung hat das Ziel, die Baumstruktur näherungsweise als Linienmodell zu bestimmen, sowie Durchmesser der Äste zu errechnen und dem Modell zuzuweisen. Bei der manuellen Bestimmung der Linienstruktur gibt es dazu zwei Möglichkeiten:

1. 3D-Konstruktion direkt in der georeferenzierten Punktwolke

Die gesamte Punktwolke aller verfügbaren Scans wird 3-dimensional dargestellt. Alle Linien werden direkt durch die röhrenförmigen Punktmengen, welche die Rinde der Baumstämme und Äste darstellen, gezeichnet. Dies hat den Vorteil, dass alle aufgenommenen Punkte des Aufnahmegebiets für die Konstruktion zur Verfügung stehen. Trotzdem ist das Zeichnen im 3-dimensionalen Raum schwierig, weil äußerst unhandlich. Zusätzlich wird das durch die Vielzahl an Ästen erschwert. Dazu kommt, dass für die Darstellung der Punktwolke in höchster Auflösung hohe Rechenleistung und großer Arbeitsspeicher erforderlich ist. Zusätzlich ist diese Darstellung in einem Zeichenprogramm notwendig, wo das Editieren der Linien möglich ist.

2. Konstruktion auf einer Plattkarte eines jeden Scans

Die aufgenommenen Punkte eines Scans werden zu Polarkoordinaten transformiert, welche in einem lokalen, kartesischen Koordinatensystem dargestellt werden. Eine Koordinatenrichtung bildet die Entfernung, eine den Horizontalwinkel und die dritte den Vertikalwinkel ab. Dadurch sind die Objekte abgeplattet. Die Objekte müssen hier nicht aus 3-dimensionalen Punkten bestehen, sondern können auch als Bilddatei in einem komprimierten Format dargestellt werden, welches die Entfernungsinformation in Form von Grauwerten besitzt. Diese Darstellung wird als Plattkarte bezeichnet. Diese Plattkarten können auch statt der Entfernungsinformation die Intensität darstellen. Es handelt sich hier um eine 2-dimensionale Abbildung mit Entfernungs- oder Intensitätsinformation. In einer solchen Darstellung kann definitiv besser und schneller ein Modell konstruiert werden. Dies ermöglicht die Verwendung einer einfachen Bilddatei als Datengrundlage an Stelle von 3-dimensionalen Punkten, was Rechen- und Speicheraufwand förmlich verschwinden lässt.

Die Vorteile der zweiten Möglichkeit überwiegen hier eindeutig. 3-dimensionales Zeichnen ist gerade bei einem so komplexen und unförmigen Gebilde, wie einem Baum, schwierig und zu zeitaufwendig. Die Darstellung von 170 Millionen Punkten eines einzigen Scans ist auf Grund des hohen Bedarfs an Rechenleistung auch im Jahr 2012 mit aktuellen PC-Systemen schwer zu bewerkstelligen. Da ist der Vorteil, die Linienstruktur direkt im Landeskoordinatensystem 3-dimensional zu zeichnen und dabei alle Scans gleichzeitig zu berücksichtigen, nicht groß genug. Das Zeichnen auf einem Entfernungs- oder Intensitätsbild ist hier wesentlich einfacher, auch wenn das genaue geometrische Einpassen einer Bilddatei in das lokale Koordinatensystem, das Abgreifen der Entfernungsinformation beim Liniensetzen, das Transformieren der Strukturen in das Landeskoordinatensystem und die Berücksichtigung mehrere Scans einen großen Programmieraufwand nach sich zieht.

6.3 Bestehende Software

Zur Lösung der Aufgabenstellung ist es möglich, verschiedene Arten an Software zu wählen. Dabei ist zu beachten, die Bearbeitungsschritte möglichst einfach und unkompliziert durchzuführen. Danach sollen die Programme gewählt werden. Für die Georeferenzierung ist ein Programm zur Datenvisualisierung notwendig. Für die Erstellung der Linienstruktur ist ein Zeichenprogramm (CAD-System, „computer-aided design“ → „rechnerunterstütztes Konstruieren“) zu wählen, mit dem Linien 3-dimensional gezeichnet und per Objektfang verbunden werden können. Die Linien sollen dabei unabhängig voneinander markiert und verarbeitet werden. Es soll möglich sein, jeder Linie unabhängig voneinander eine Eigenschaft zuzuweisen, um den zusätzlich gemessenen Durchmesser des jeweiligen Astes bzw. Stammes zu speichern. Diese Eigenschaft muss daher eine Zahl im Single-Datentyp sein. Das Programm soll eine Programmierumgebung besitzen, um Punkte und Linien zwischen dem lokalen Koordinatensystem und dem übergeordneten Landeskoordinatensystem hin und her zu transformieren. Zusätzlich sind noch Programme für verschiedenste mathematische Berechnungen von Nutzen.

z+f-LaserControl zur Georeferenzierung

Für die Bearbeitung der Scans im firmeneigenen zfs-Format ist das zum Scanner von Zoller+Fröhlich angebotene Verarbeitungsprogramm z+f-LaserControl notwendig. Damit ist die Abbildung und Analyse der Scans möglich. In Kapitel 4.4 wurde das

Programm für die Georeferenzierung verwendet und liefert damit für jeden Scan die zugehörigen Transformationsdaten. Das Programm bietet aber keine Funktionen zum Zeichnen bzw. Konstruieren der Linienstrukturen an.

AutoCAD als CAD-System

Das Programm AutoCAD von der Firma Autodesk hat die entsprechenden Voraussetzungen und Vorteile für das Zeichnen der Baumstrukturen:

- Darstellung von Punkten und Linien als voneinander unabhängige Objekte
- umfassende und erprobte Zeichenfunktionen (Objektfang, Zoom-, Panfunktionen)
- 3-dimensionale Zeichnungsdarstellung
- Import und geometrische Bearbeitung von Pixelbildern wie die erstellten Plattkarten (Intensitätsbild, Entfernungsbild) in Position und Ausdehnung
- Programmierumgebung (Visual Basic for Applications) zur automatischen Markierung, Transformation und Zeichnung von Objekten sowie Import und Export von Daten aus und in Textdateien
- bekannte und weit verbreitete Dateiformate für das Zeichendokument (*.dwg)
- Möglichkeit, den gesamten Modellierungsaufwand in einer einzigen Programmumgebung zu bewerkstelligen

OPALS

OPALS steht für “Orientation and Processing of Airborne Laser Scanning data”. Es ist ein modulares Softwaresystem, welches sich vor allem auf automatische Prozessierung großer Datenmengen konzentriert, welche sich im Airborne Laserscanning (ALS) zwangsläufig ergeben. Trotz dieses Fokus auf ALS kann für gewisse Zwecke das Softwaresystem auch für die Punktwolken von TLS verwendet werden [15]. Das System ist ein reines Kommandozeilenprogramm und wird daher durch keine graphische Benutzeroberfläche gesteuert. Die Prozessierung kann durch Scripts durchgeführt werden, die ganze Befehlsreihen enthalten können. Dabei können einzelne Module (Softwarekomponenten) aufgerufen werden, die bestimmte Verarbeitungsschritte der Daten durchführen. Insbesondere die Schaffung von farbkodierten Rasterkarten ist eine für dieses Projekt brauchbare Funktion.

Matlab

Matlab steht für **Mat**rix **Lab**oratory und ist eine kommerzielle Software der Firma The MathWorks. Das Programm dient zur numerischen Auswertung von mathematischen Aufgabenstellungen an Hand von Matrix-Implementierungen. Es ist ein mächtiges Instrument, um große Datenmengen zu verarbeiten und bietet eine mächtige Programmierumgebung zur Bearbeitung von Textdateien und Bilddateien in allen möglichen Formaten. Für Berechnungen in der Baummodellierung kann das Programm hilfreich sein. Es bietet aber keine Module zum manuellen Zeichnen von graphischen Objekten.

6.4 Methodenentwicklung

Um das Ergebnis in Form eines Linienmodells für die aufgenommenen Bäume zu bekommen, ist ein geeigneter Arbeitsfluss festzusetzen. Dabei wird eine entsprechende Modellierungsstrategie angewendet.

6.4.1 Modellierungsstrategie

Die Strategie für den Modellierungsablauf erfolgt in zwei Schritten (Abbildung 23). Im ersten Schritt werden die Aufnahmen in z+f-LaserControl für die Konstruktion bearbeitet. Auf Grund der hohen Auflösung von TLS und der damit verbundenen großen Datenmenge pro Standpunkt ist eine Konstruktion mit Hilfe eines von z+f-LaserControl exportierten Intensitätsbildes der abgeplatteten Punktwolke im TIFF-Format (Tagged Image File Format) notwendig. Wie in Kapitel 4.4 bereits erwähnt, sind die Bäume und ihre Struktur im Intensitätsbild gut erkennbar. Auf diesem Intensitätsbild, welches direkt in ein lokales Standpunktkoordinatensystem eingebunden ist, soll die Struktur in AutoCAD gezeichnet werden. Zusätzlich soll noch ein Entfernungsbild geschaffen werden. Der Entfernungsinformation werden Grauwerte zugewiesen und dadurch eine Bilddatei im TIFF-Format mit der gleichen Ausdehnung (Höhe, Breite) wie das Intensitätsbild erstellt. Beim Setzen eines Linienpunktes auf das Bild soll nun der Grauwert des Pixels, auf dem dieser Punkt gesetzt wird, abgegriffen und dieser in die Entfernungsinformation umgerechnet werden. Dabei kommt es zu einem Problem in der Datenstruktur der Bilddatei: Um Entfernungsinformation in Millimeter-Genauigkeit im Intervall von 0 bis 60 m (festgelegte Aufnahmeentfernung im Projekt → Kapitel 4.3) Grauwerten zuzuweisen, ist

16 Bit-Information notwendig (65536 verschiedene Grauwerte), um eine Entfernungsdifferenz zwischen benachbarten Grauwerten von unter 1 mm zu erreichen. Das Programm z+f-LaserControl kann aber nur Grauwertbilder in 8 Bit-Information exportieren (256 Grauwerte). Dadurch wäre die Entfernungsdifferenz bei ca. 10 cm. Abgesehen davon ist die genaue mathematische Zuweisung der Grauwerte den Entfernungswerten in diesem Programm nicht einsichtig.

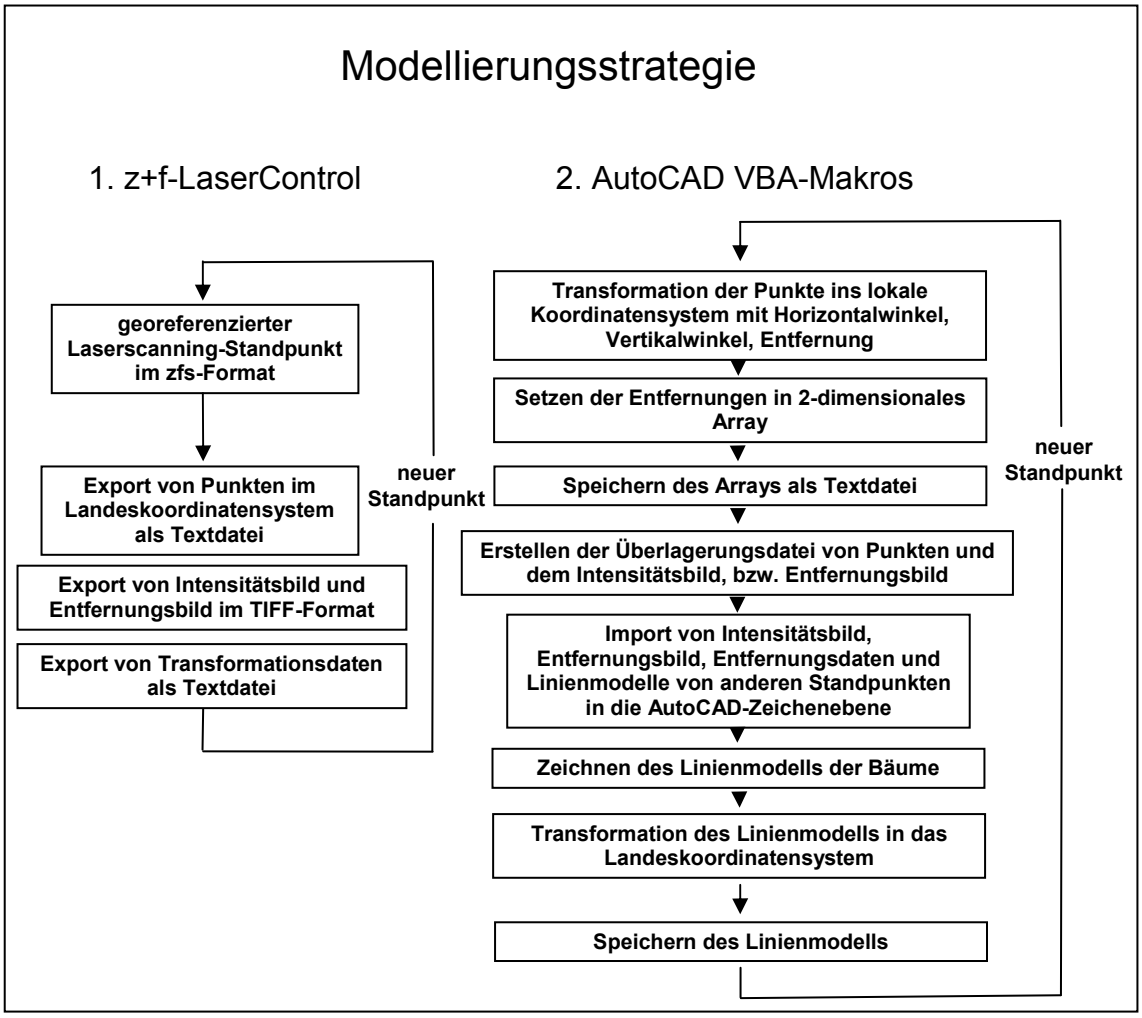

Abbildung 23 : Schematische Darstellung der Modellierungsstrategie

Eine bessere Variante ist, die Entfernungsinformation der Punkte in ein 2-dimensionales Array von Zahlen abzuspeichern, welches direkt in mathematischer Beziehung mit den Plattkarten steht. Ein Entfernungsbild mit 8 Bit-Grauwertinformation kann trotzdem bei der Konstruktion hilfreich sein, aber nicht als Datenquelle für genaue Entfernungen. Da z+f-LaserControl die Punkte nur im Landeskoordinatensystem als weiterverarbeitbare Textdatei exportieren kann, muss eine Transformation (wie in Kapitel 4.4) ins abgeplattete lokale Standpunktkoordinatensystem mit Hilfe der Programmierumgebung VBA

durchgeführt werden. Somit werden mit z+f-LaserControl für jeden aufgenommenen Scan (Standpunkt) insgesamt vier Dateien erstellt:

- Punktkoordinaten der Punkte in einer Textdatei (ca. 3-4 GB)
- Intensitätsbild im Tiff-Format
- Entfernungsbild im Tiff-Format
- Transformationsdaten in einer Textdatei

Im zweiten Schritt (Abbildung 23) werden die Punkte im Landeskoordinatensystem zuerst in AutoCAD in das lokale, abgeplattete Koordinatensystem transformiert. Die Entfernungsinformation der Punkte wird nun in ein 2-dimensionales Array abgespeichert, welches dieselbe Ausdehnung und Auflösung wie der gemessene TLS-Punktraster besitzt. Um sich während der Konstruktion das rechenintensive Lesen und Transformieren der Punkte im Landeskoordinatensystem zu ersparen, werden die Entfernungen in einer eigenen Textdatei abgespeichert. Diese kann bei Bedarf mit deutlich weniger Zeitaufwand gelesen werden, da sie geringeren Speicherbedarf hat. Die Lage der Punkte in der AutoCAD-Zeichenebene müssen mit den Pixeln der Plattkarten übereinstimmen, also überlagert werden (Abbildung 24). Dies wird durch die entsprechende Position und Skalierung der Plattkarten in der Zeichenebene gewährleistet. Die Parameter für die Position und Skalierung der Plattkarten sollen in eine eigene Textdatei gespeichert werden, der sogenannten Überlagerungsdatei. Diese Überlagerungsdatei muss je nach Ausdehnung der Plattkarte für jeden einzelnen Scan erstellt werden.

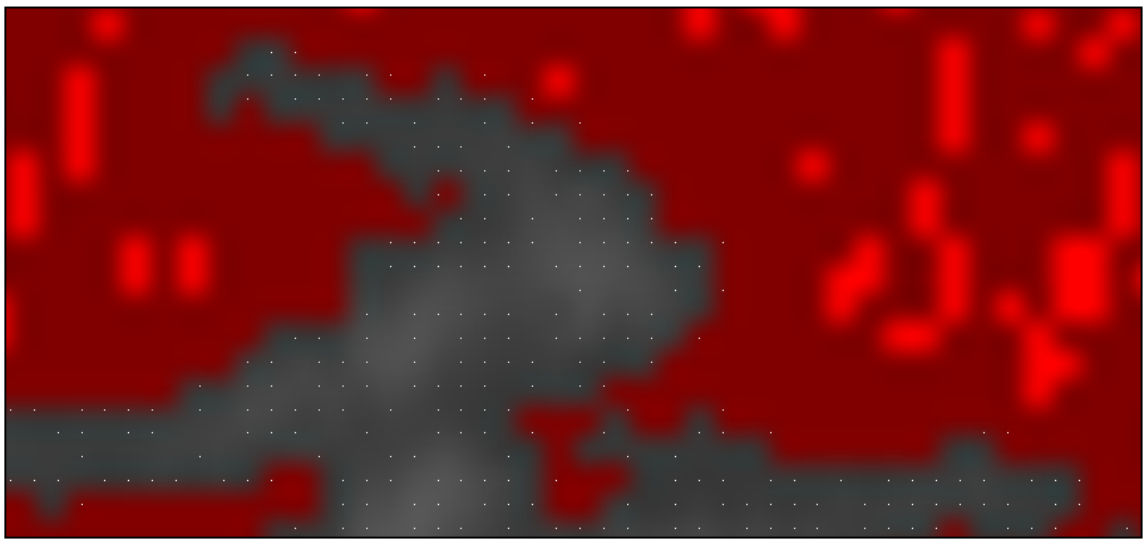

Abbildung 24 : Überlagerung der Punkte im Array (weiß) mit dem Intensitätsbild

Beim Setzen eines Linienpunktes auf das Intensitätsbild wird über die Koordinaten und einige Überlagerungsparameter sofort auf die entsprechende Stelle im Array zugegriffen und dem 3-dimensionalen Punkt eine Entfernung zugewiesen. Diese Entfernung bildet die z-Koordinate des Punktes. Somit werden die Linien im Raum bestimmt und die Struktur kann gezeichnet werden.

Die fertig konstruierte Baumstruktur soll anschließend in das Landeskoordinatensystem transformiert, im *.dwg-Format abgespeichert und kann für die Linienkonstruktion des nächsten Standpunkts importiert werden.

6.4.2 Erweiterte Modellierungsstrategie

Die Modellierungsstrategie kann aber auch durch gewisse Anwendungen verbessert bzw. erweitert werden. So soll im ersten Schritt (Abbildung 26) das Programm z+f-LaserControl nur zum Export der Transformationsdaten und der Punktkoordinaten im lokalen, kartesischen Koordinatensystem in einer Textdatei verwendet werden. Diese kartesischen Koordinaten werden im zweiten Schritt mit Hilfe von Matlab in Polarkoordinaten umgerechnet. Diesen Polarkoordinaten werden noch Intensitätswerte und berechnete Horizontalentfernungen hinzugefügt. Zusätzlich wird mit Matlab eine Textdatei geschrieben, welche Entfernungsdaten in exakt derselben Ausdehnung wie das 2-dimensionale Array zeilenweise beinhaltet.

Mit Matlab werden daher folgende Dateien erstellt:

- Lokale Polarkoordinaten mit Intensitätswerten und Horizontalentfernungen in Textdatei
- Entfernungsdaten in Textdatei

Anschließend werden als dritten Schritt mit dem Programm OPALS aus den Intensitätswerten der Polarkoordinatendatei ein Intensitätsbild und aus den Horizontalentfernungen ein farbiges Entfernungsbild erstellt (Abbildung 25).

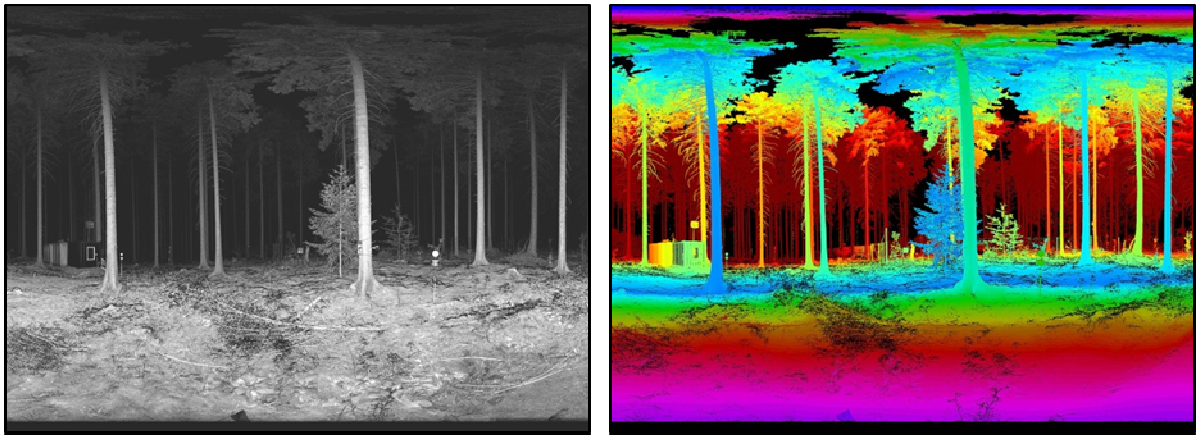

Abbildung 25 : links: Intensitätsbild in Grauwertdarstellung
rechts: Entfernungsbild, Farben geben Horizontalentfernung wieder

Es ist nun die gewünschte, einheitliche Ausdehnung vorhanden, womit die Überlagerung mit dem Entfernungsarray vereinfacht wird. Es ist nur eine einzige Überlagerungsdatei für alle Standpunkte notwendig. Außerdem wird pro Standpunkt nur mehr eine Plattkarte erstellt, die den ganzen 360°-Rundumblick abdeckt und sogar noch mit einem zusätzlichen Überlappungsbereich ausgestattet werden kann. Das Entfernungsbild wird zusätzlich entsprechend eingefärbt und zwar auch nach der Horizontalentfernung, was die visuelle Orientierung beim Zeichnen erleichtert.

Mit OPALS werden daher folgende Dateien erstellt.

- Intensitätsbild im Tiff-Format
- Entfernungsbild im Tiff-Format

Der vierte Schritt (Abbildung 26) wird in AutoCAD mit VBA durchgeführt. Dabei werden das Intensitätsbild, das Entfernungsbild und die Entfernungsdatendatei importiert und die Baumstrukturen in derselben Art und Weise erstellt wie im zweiten Schritt der einfachen Modellierungsstrategie.

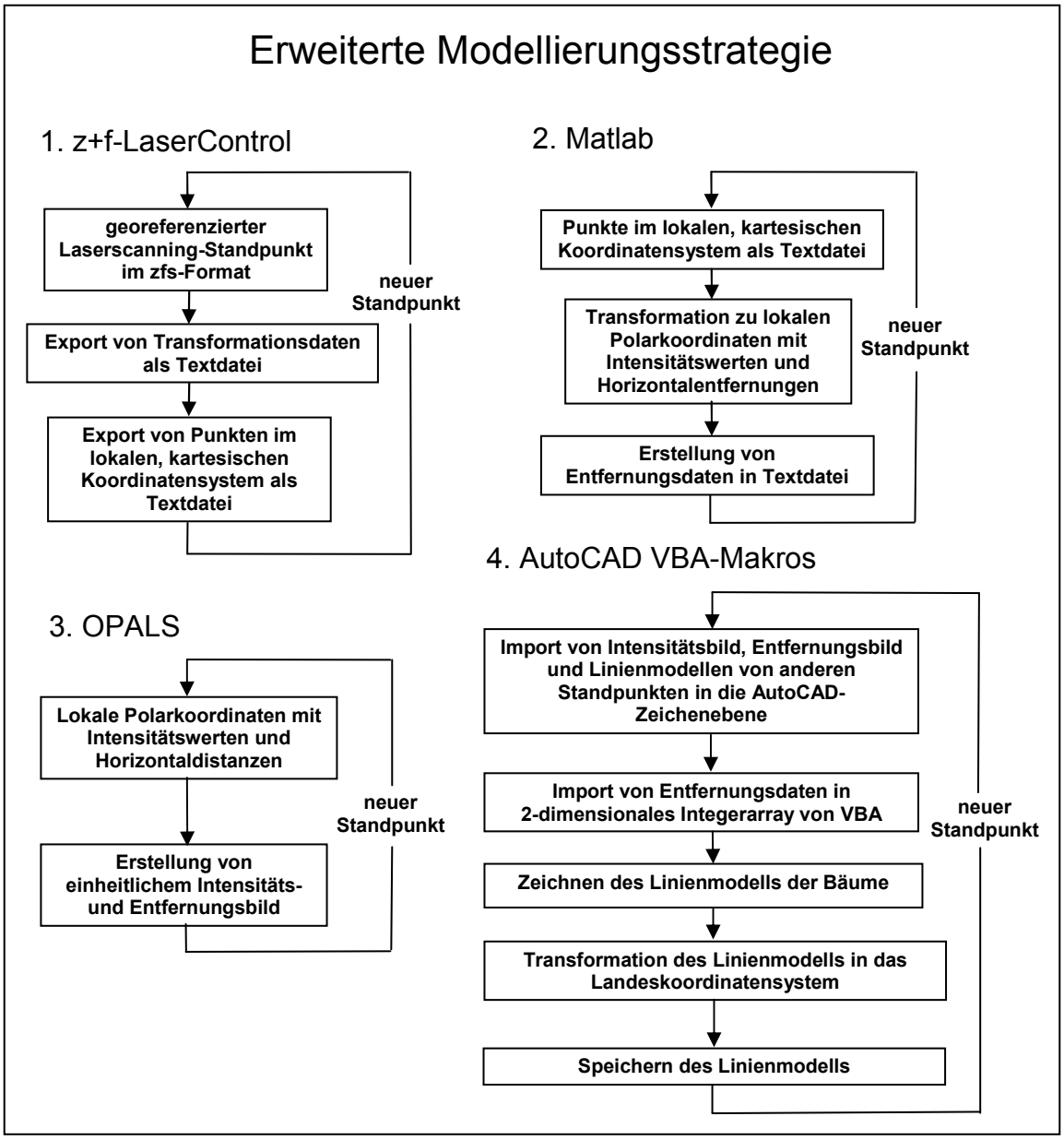

Abbildung 26 : Schematische Darstellung der erweiterten Modellierungsstrategie

6.4.3 Folgerung aus der Strategienentwicklung

Die Erstellung von Intensitäts- und Entfernungsbildern exakt gleichgroßer Ausdehnung mittels OPALS hat den großen Vorteil, dass die Parameter zum Import der Bilder in AutoCAD immer gleich sind und daher keine eigene Datei mit Importinformationen für jeden Standpunkt notwendig ist. Es ist daher kein eigenes Makro zur Bestimmung der individuellen Überlagerung zu erstellen. Dieses Programm müsste die Importinformation durch idente Punkte ermitteln, was einen komplizierten Programmieraufwand bedeuten würde. Dann müssten zusätzlich beide getrennten Teile der Bilder extra bestimmt werden. Nachteilig ist, dass zusätzlich ein drittes Programm in den Arbeitsfluss miteinbezogen wird und daher eine zusätzliche Beschäftigung mit diesem Programm notwendig ist.

Trotzdem ist die Vereinfachung, die die Erstellung der Bilder mit OPALS mit sich bringt, eindeutig von Vorteil. Das Zeichnen mit dem farbigen Entfernungsbild als Orientierung wird einfacher und eine bessere Übereinstimmung der Objekte mit der Punktwolke ist vorhanden. Deshalb wird für die softwaretechnische Umsetzung (Kapitel 6.5) und deren Anwendung (Kapitel 6.6) die erweiterte Modellierungsstrategie angewendet.

6.5 Softwaretechnische Umsetzung

Für die Umsetzung diverser Berechnungen wird Matlab verwendet. Dafür werden zwei Programme verwendet, sogenannte Scripts, die in der in Matlab implementierten Scriptsprache geschrieben werden. Auch in OPALS wird ein Script in einer eigenen Programmiersprache geschrieben.

AutoCAD besitzt zwei Programmiersprachen zur Erstellung von Programmen zur Bearbeitung der Zeichnungen: List Processing (LISP) und Visual Basic for Applications (VBA). Das Erstellen eines Programms zur Baummodellierung ist in AutoCAD durch Makros in der Programmiersprache VBA möglich. Makros sind kleine Programme, die vollen Zugriff auf alle Zeichenelemente und deren Bearbeitungsmöglichkeiten besitzen. Damit sind automatische Zeichen- und Bearbeitungsschritte möglich. LISP wird als ältere Programmiersprache ebenfalls verwendet, aber nur um die VBA-Makros mit Hilfe von einfachen LISP-Programmen über die Befehlszeile des AutoCAD-Programms aufzurufen. Die von AutoCAD jährlich erschienenen Versionen können alle für die Aufgaben verwendet werden. Wichtig ist dabei das Zusatzmodul zur Programmierung in VBA. Dieses Zusatzmodul ist seit der Version AutoCAD 2010 nicht mehr standardmäßig beim Programmpaket dabei, kann aber kostenlos von der Website der Firma Autodesk bezogen werden. Ein Wechsel zwischen zwei unterschiedlichen VBA-Versionen sollte vermieden werden. Es kann zu Problemen bei der Kompatibilität mit den Objektdatenbanken kommen.

Für jeden Arbeitsschritt wird ein eigenes Makro verwendet, sowohl für das Einrichten zur Modellierung im jeweiligen Standpunkt (Makro „Standpunkt“), zum Zeichnen des Linienmodells (Makro „Linienzeichnen“), sowie zur Transformation des gezeichneten Linienmodells (Makro „Transformation). Diese drei Makros bilden das Grundgerüst für mathematische Prozesse in der Modellierung.

6.5.1 Matlab Scripts

Diese zwei Scripts bilden jeweils einen Berechnungsabschnitt ab, wie im zweiten Schritt in Abbildung 26.

Im ersten Maplab-Script “Transformation ins lokale Polarkoordinatensystem” (siehe Anhang) werden die lokalen, kartesischen Koordinaten, welche aus z+f-LaserControl exportiert wurden, aus einer Textdatei eingelesen. Danach werden die Koordinaten in Polarkoordinaten umgerechnet, analog der (Gl. 12):

$$\begin{pmatrix} D_l \\ V_l \\ H_l \end{pmatrix} = \begin{pmatrix} \sqrt{x_l \cdot y_l \cdot z_l} \\ \arcsin\left(\frac{z_l}{D_l}\right) \\ \arcsin\left(\frac{y_l}{D_l \cdot \cos(V_l)}\right) \end{pmatrix}$$

D_l ..... Entfernung
H_l ..... Horizontalwinkel (Gl. 12)
V_l .....Vertikalwinkel vom Horizont

Diese werden in eine Textdatei in 3 Spalten geschrieben. Zusätzlich werden in einer 4. Spalte die Horizontalentfernungen jedes einzelnen Punktes aufgelistet (Gl. 13):

$$D_{horizontal} = \cos(V_l) \cdot D_l$$

D_l ..... Entfernung
$D_{horizontal}$ ..... Horizontalentfernung
V_l ..... Vertikalwinkel vom Horizont (Gl. 13)

An die Daten wird ein 90°-Überlappungsbereich angehängt. Das heißt den Polarkoordinaten des ersten Quadranten wird der Horizontalwinkel um 360° vergrößert und die Koordinaten in der Textdatei dem Dateiende hinzugefügt.

Im zweiten Matlab-Script “Erstellung der Entfernungsdatei” (siehe Anhang) wird mit Matlab eine Textdatei geschrieben, welche Entfernungsdaten in exakt derselben Ausdehnung wie bei jedem anderen Standpunkt zeilenweise beinhaltet. Die in der Datei (Abbildung 27) enthaltenen Zahlen $D_{Textdatei}$ sind die Entfernungswerte eines jeden Pixels im jeweiligen Intensitätsbild. Alle Werte werden zu ganzen Zahlen im Bereich zwischen -32000 und +32000 transformiert:

$$D_{Textdatei} = \text{Rundung}\left(D_l\right) \cdot 1000 - 32000 \qquad \text{(Gl. 14)}$$

Sie können daher beim Import als Integerdatentyp gespeichert werden. Das hat den großen Vorteil, dass ein Speicherbedarf von nur 2 Byte pro Wert vorliegt, statt der 4 Byte eines Singledatentyps. Das ist notwendig, da der zur Verfügung gestellte Speicher in AutoCAD begrenzt ist. Die Genauigkeit nimmt durch diese Komprimierung in den 2 Byte-Bereich ab, ist aber für unseren Bedarf noch immer genau genug. Bei einer für uns typischerweise höchsten Aufnahmeentfernung von 60 m liegt die Auflösung der Entfernungswerte noch bei 1 mm.

```
-5708
-5712
-5708
-5708
-5708
-5724
-32000
-32000
-5604
-5604
-5604
-5604
-5604
-32000
-32000
```

Abbildung 27 : Entfernungen für jedes Pixel des Arrays in einer Textdatei

6.5.2 OPALS-Script

Mit dem Programm OPALS wird ein Script “Erstellung der Intensitäts- und Entfernungsbilder” (siehe Anhang) erstellt, das aus den Entfernungswerten der Polarkoordinatendatei, die mit dem Matlab-Script “Transformation ins lokale Polarkoordinatensystem” erstellt wurden, ein Intensitätsbild und ein farbiges Entfernungsbild erzeugt (Abbildung 25). Für das farbige Entfernungsbild werden die Horizontalentfernungen verwendet, da sie bei der Konstruktion mehr Vorstellungskraft schaffen sollen.

6.5.3 Makro „Standpunkt“

Dieses Makro bildet den Hauptanteil an Funktionen im vierten Schritt der erweiterten Modellierungsstrategie (Kapitel 6.4.2) ab. Das betrifft in erster Linie den Import von Plattkarten und Entfernungsdaten in die Zeichenebene von AutoCAD.

Im ersten Schritt ist die von Matlab geschriebene Entfernungsdatendatei im txt-Format zu öffnen (Abbildung 27). Die Werte werden zeilenweise eingelesen und in ein 2-dimensionales

Integerarray gespeichert, welches dieselbe Ausdehnung hat wie die verwendete Plattkarte. Das heißt, jedes Pixel bekommt genau einen Integerwert.

Der zweite Schritt ist das Öffnen von Intensitäts- und Entfernungsbild, die mit OPALS erzeugt wurden. Die Geometrie der auf dem Intensitätsbild dargestellten Objekte (Bäume) muss genau mit dem lokalen, abgeplatteten Koordinatensystem übereinstimmen bzw. überlagert sein. Daher muss Intensitäts- und Entfernungsbild mit festgesetzter Position und Skalierung in AutoCAD importiert werden. Dazu sind die Überlagerungsinformationen notwendig, die in eine Textdatei abgespeichert sind. Diese Datei wird zusätzlich zu den Bildern importiert. Durch die einheitliche Erstellung der Plattkarten mit OPALS haben sie in jedem Standpunkt dieselbe Auflösung und Größe. Somit kann immer dieselbe, einheitliche Überlagerungsdatei verwendet werden (Abbildung 28). Die Datei enthält die Koordinaten des linken unteren Eckpunktes als Einfügepunkt. Die Rotation kann vernachlässigt werden und wird auf 0 gesetzt. Dann werden die Höhe und die Breite des Bildes festgesetzt, wodurch in beiden Richtungen eine Dehnung stattfindet. Dadurch ist eine Skalierung nicht notwendig und wird daher auf 1 gesetzt.

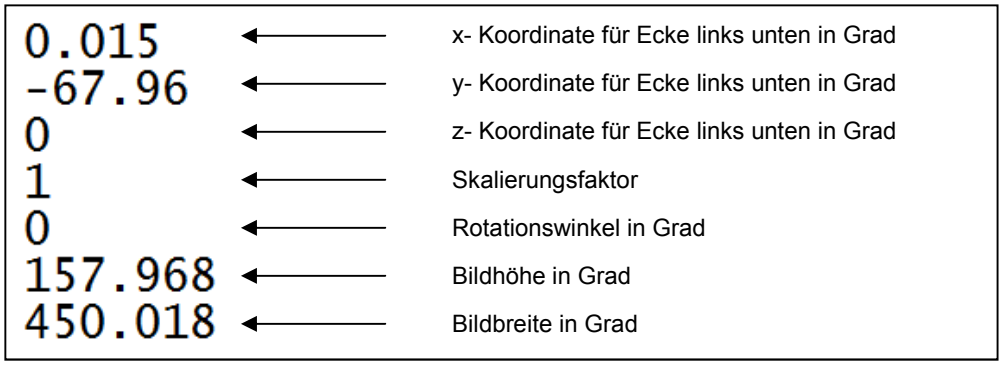

Abbildung 28 : Überlagerungsinformationen in einer Textdatei, wie in AutoCAD erforderlich

Der dritte Schritt ermöglicht das Öffnen von bereits erstellten Baumstrukturen, falls diese existieren. Die Baumstrukturen sind im übergeordneten Landeskoordinatensystem dargestellt und im AutoCAD-Dateityp *.dwg abgespeichert. Es können mehrere dwg-Dateien mit darin befindlichen Baumstrukturen importiert werden. Für den Import sind die Transformationsdaten (Abbildung 29) zwischen dem jeweiligen Standpunkt und dem Landeskoordinatensystem notwendig. Diese werden als txt-Datei geöffnet. Sie besteht aus einer 4x4-Matrix mit einer 3x3-Rotationsmatrix, einem Translationsvektor (in m) und einem Maßstabsfaktor, der auf 1 gesetzt ist.

Rotation			Translation
-0.1070324435	0.9942547771	0.0012223683	33399222.8910534870
-0.9942550942	-0.1070334650	0.0008031371	5646660.4183985749
0.0009293572	-0.0011293842	0.9999989304	383.3510947546
0.0000000000	0.0000000000	0.0000000000	1.0000000000
			Maßstab

Abbildung 29 : Inhalt der Transformationsdatei – 4x4-Matrix mit Rotation und Translation und Maßstabsfaktor

Die Linien in der AutoCAD-Zeichnung werden Linie für Linie transformiert, indem Anfangs- und Endpunkt der Linie transformiert werden. Die Transformation ist die Kongruenztransformation vom Landeskoordinatensystem ins lokale System also in umgekehrter Richtung wie bei den Gleichungen (Gl. 5), (Gl. 6) in Kapitel 4.4:

$$x_l = R^{-1} \cdot (x_{LK} - t)$$

$$\begin{pmatrix} x_l \\ y_l \\ z_l \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}^{-1} \cdot \left(\begin{pmatrix} x_{LK} \\ y_{LK} \\ z_{LK} \end{pmatrix} - \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \right) \qquad \text{(Gl. 15)}$$

r_{xx} .....Rotationselemente V_l ..... Translationselemente

Dann werden die lokalen, kartesischen Koordinaten in das Polarkoordinatensystem der Plattkarte umgewandelt:

$$\begin{pmatrix} D_l \\ V_l \\ H_l \end{pmatrix} = \begin{pmatrix} \sqrt{x_l \cdot y_l \cdot z_l} \\ \arcsin\left(\frac{z_l}{D_l}\right) \\ \arcsin\left(\frac{y_l}{D_l \cdot \cos(V_l)}\right) \end{pmatrix}$$

D_l ..... Entfernung
H_l ..... Horizontalwinkel (Gl. 16)
V_l .....Vertikalwinkel vom Horizont

Dieser dritte Schritt ist für das Zeichnen der Strukturen nicht unbedingt erforderlich, aber notwendig, um mehrere Standpunkte für ein und dieselben Objekte zu verwenden.

6.5.4 Makro „Linienzeichnen“

Zum Zeichnen der Linien wird nicht die von AutoCAD standardmäßig zur Verfügung gestellte Linienkonstruktion verwendet, sondern ein zum Zeichnen programmiertes Makro. Da auf einer 2-dimensionalen Plattkarte gezeichnet wird, ist eine automatische Entfernungsbestimmung notwendig.

Der genaue Vorgang der Linienkonstruktion ist in Abbildung 30 dargestellt. Das Makro fordert als ersten Schritt das Setzen des Anfangspunkts A der Linie, welcher in die Mitte des Stammes bzw. Astes gesetzt wird. Danach ist das Setzen von Randpunkten, nämlich des linken bzw. des rechten Punktes notwendig, welche beide die äußeren Ränder des Stammes bzw. Astes festsetzen sollen. Daraufhin werden der Endpunkt B der Linie sowie ebenfalls dessen zusätzlicher linker und rechter Randpunkt gesetzt.

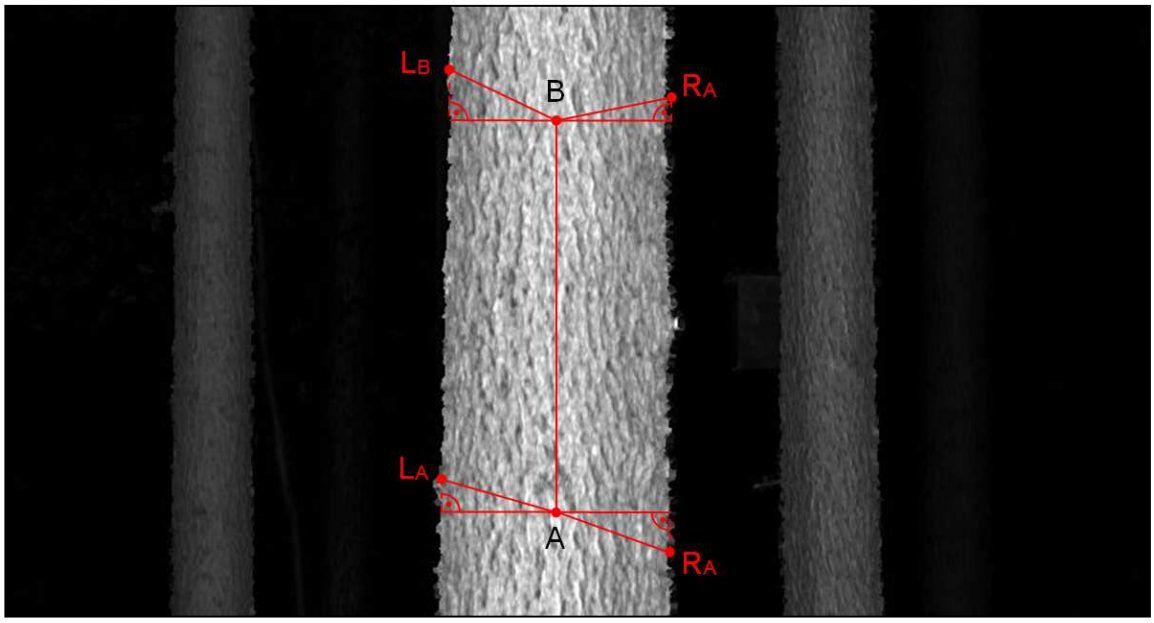

Abbildung 30 : Bestimmung der Linie und deren Randpunkte in AutoCAD

Wenn der Anfangspunkt per Objektfang als Endpunkt einer anderen Linie bestimmt wird, entfällt die Bestimmung der Randpunkte des Anfangspunktes automatisch.

Für die Berechnung wird als Erstes für den Anfangspunkt und den Endpunkt die Entfernung bestimmt. Durch das Setzen des Punktes auf dem im lokalen Koordinatensystem positionierten Intensitätsbild können die Lagekoordinaten bestimmt werden. Diese stehen in direkter mathematischer Beziehung zu dem im Hintergrund gespeichertem Array, wodurch dem Punkt genau der Wert zugewiesen werden kann, der an der Stelle im Array gespeichert ist, der dem Punkt am nächsten ist (Abbildung 31). Die 8 im rasterförmigen Array umliegenden Werte werden zur Bestimmung zusätzlich verwendet. Es werden von den insgesamt 9 Werten alle Entfernungswerte, welche ungleich 0 sind (Menge Q), für die Bestimmung der Entfernung e herangezogen, indem der Median berechnet wird:

$$D = \text{Median}(Q) \qquad \text{(Gl. 17)}$$

Der Median hat den großen Vorteil einer robusten Bestimmung der tatsächlichen Entfernung des Punktes. Ausreißer unter den bis zu 9 Entfernungswerten wirken sich nicht aus, solange sie nicht die Hälfte von ihnen einnehmen.

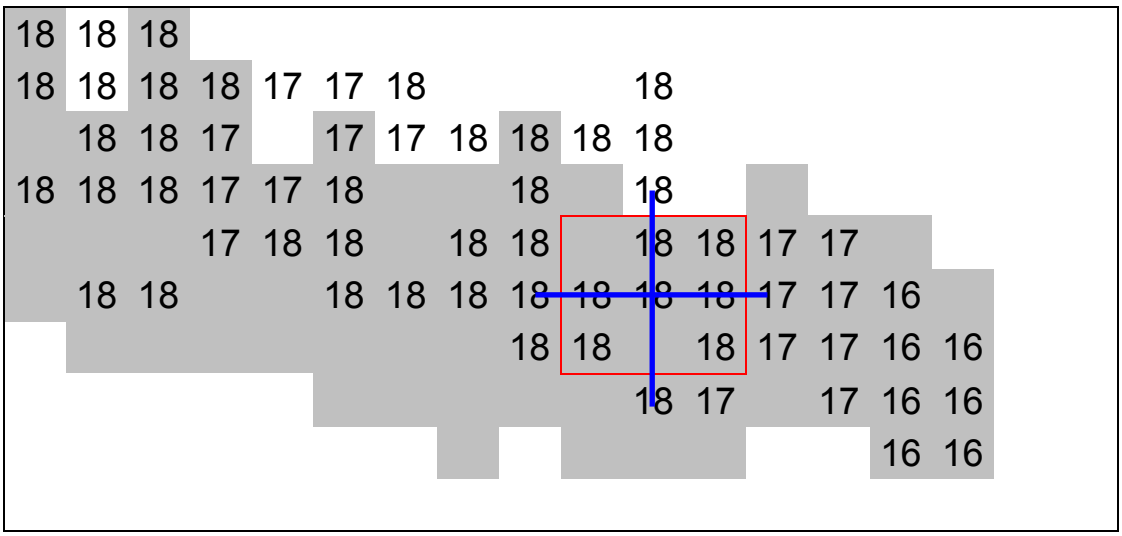

Abbildung 31 : Im Entfernungsarray werden die 9 Werte um den ausgewählten Punkt zur Bestimmung der Entfernung verwendet (Menge Q).

Die gesetzte Linie soll aber durch die Achse des Stammes bzw. Astes verlaufen. Daher wird aus dem linken und dem rechten Randpunkt der entsprechende Abstand zur Achse berechnet, welcher der Entfernung hinzuaddiert wird (siehe Abbildung 32). Außerdem soll noch der Durchmesser des Stammes bzw. Astes bestimmt werden.

Dafür wird folgendermaßen vorgegangen: Der Startpunkt A und der Endpunkt B wird in das lokale kartesische Koordinatensystem transformiert:

$$\begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} = D_A \cdot \begin{pmatrix} \sin(H_A)\cdot\sin(V_A) \\ \cos(H_A)\cdot\sin(V_A) \\ \cos(V_A) \end{pmatrix} \quad \begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = D_B \cdot \begin{pmatrix} \sin(H_B)\cdot\sin(V_B) \\ \cos(H_B)\cdot\sin(V_B) \\ \cos(V_B) \end{pmatrix} \qquad \text{(Gl. 18)}$$

Dies passiert auch mit den beiden Randpunkten L_A und R_A:

$$\begin{pmatrix} X_{L_A} \\ Y_{L_A} \\ Z_{L_A} \end{pmatrix} = \begin{pmatrix} \sin(H_{L_A})\cdot\sin(V_{L_A}) \\ \cos(H_{L_A})\cdot\sin(V_{L_A}) \\ \cos(V_{L_A}) \end{pmatrix} \quad \begin{pmatrix} X_{R_A} \\ Y_{R_A} \\ Z_{R_A} \end{pmatrix} = \begin{pmatrix} \sin(H_{R_A})\cdot\sin(V_{R_A}) \\ \cos(H_{R_A})\cdot\sin(V_{R_A}) \\ \cos(V_{R_A}) \end{pmatrix} \qquad \text{(Gl. 19)}$$

Der Richtungsvektor $\overrightarrow{AB}$ wird berechnet:

$$\overrightarrow{AB} = \begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} - \begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} \qquad \text{(Gl. 20)}$$

Mit Hilfe des Vektors $\overrightarrow{AB}$ werden die Normalvektoren der Tangentialebenen τ_L und τ_R, welche durch S und L bzw. R gelegt sind, berechnet. Der Raumwinkel der beiden Ebenen bildet schließlich den Öffnungswinkel ε_A bzw. ε_B am Stamm bzw. Ast:

$$\varepsilon_A = \angle\left(\tau_{L_A}, \tau_{R_A}\right) = \angle\left(\vec{n}_{L_A}, \vec{n}_{R_A}\right) = \angle\left(\overrightarrow{SL}_A \times \overrightarrow{AB}, \overrightarrow{SR}_A \times \overrightarrow{AB}\right) \qquad \text{(Gl. 21)}$$

$$\varepsilon_B = \angle\left(\tau_{L_B}, \tau_{R_B}\right) = \angle\left(\vec{n}_{L_B}, \vec{n}_{R_B}\right) = \angle\left(\overrightarrow{SL}_B \times \overrightarrow{AB}, \overrightarrow{SR}_B \times \overrightarrow{AB}\right)$$

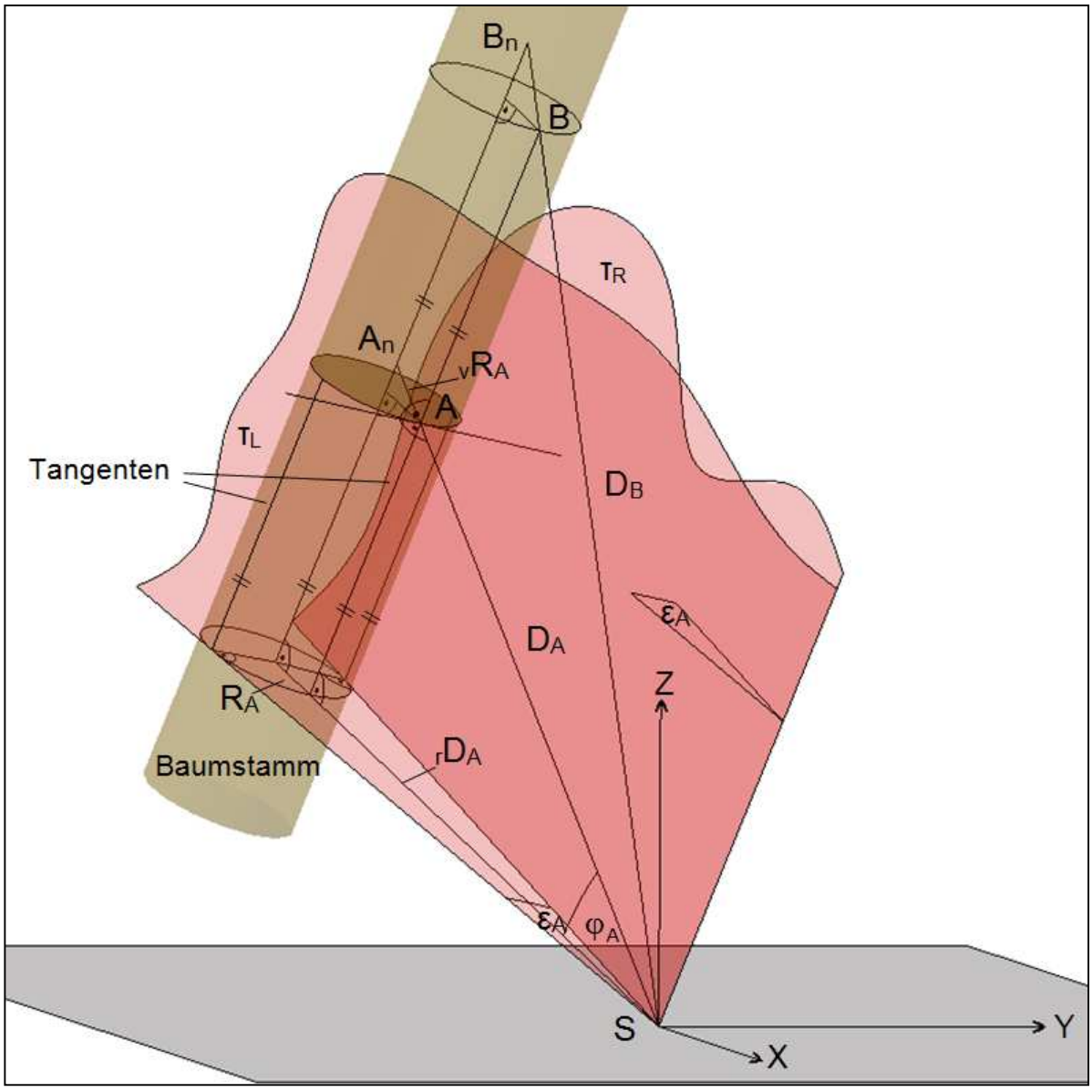

Abbildung 32 : Bestimmung der Verlängerung aus dem Entfernungsabgriff, AutoCAD

Da der Abstand in der Regel sehr klein (< 1,5 Bogensekunden) ist, kann der Öffnungswinkel durch die Bogensehne angenähert werden. Das ist wichtig, da eine genaue Winkelbestimmung eine unnötig hohe numerische Rechengenauigkeit erforderlich machen würde. Dies gilt z.B. bei der Berechnung des Cosinus von einem sehr kleinen Winkel, wie es hier der Fall wäre. Nun wird der Winkel φ_A bzw. φ_B berechnet:

$$\varphi_A = \left| 90° - \angle\left(\overrightarrow{SA}, \overrightarrow{AB}\right)\right| \qquad \varphi_B = \left| 90° - \angle\left(\overrightarrow{SB}, \overrightarrow{AB}\right)\right| \quad \text{(Gl. 22)}$$

Der Abstand E vom Aufnahmezentrum zur Astvorderseite wird mit Hilfe des Cosinus des Vertikalwinkels in die Normalebene zu $\overrightarrow{AB}$ projiziert, also reduziert (rD_A) (Abbildung 32):

$$rD_A = D_A \cdot \cos(\varphi_A) \qquad rD_B = D_B \cdot \cos(\varphi_B) \qquad \text{(Gl. 23)}$$

Denn nur in der Normalebene kann der Stammdurchschnitt als Kreis angenommen werden. Somit kann der Radius (R_A) des Stammes mit folgender Formel berechnet werden:

$$R_A = \frac{rD_A}{\frac{1}{\sin\left(\frac{\varepsilon_A}{2}\right)} - 1} \qquad R_B = \frac{rD_B}{\frac{1}{\sin\left(\frac{\varepsilon_B}{2}\right)} - 1} \qquad \text{(Gl. 24)}$$

Der Radius wird mit Hilfe des Cosinus des Vertikalwinkels wieder in die Schnittebene projiziert, somit verlängert (vR_A) (Abbildung 32):

$$vR_A = \frac{R_A}{\cos(\varphi_A)} \qquad vR_B = \frac{R_B}{\cos(\varphi_B)} \qquad \text{(Gl. 25)}$$

Dieser verlängerte Radius wird der Entfernung des Anfangspunktes und des Endpunktes hinzuaddiert, damit beide Punkte im Zentrum des Stammes liegen.

$$D_A = D_A + vR_A \qquad D_B = D_B + vR_B \qquad \text{(Gl. 26)}$$

Wenn der Abstand der beiden Punkte D_A und D_B größer als 50 cm ist, dann wird das Makro „Schnittpunktdarstellung“ aufgerufen und die Berechnung noch einmal durchgeführt (siehe Makro „Schnittdarstellung“.

$$\left| D_A - D_B \right| > 50\,\text{cm} \qquad \text{(Gl. 27)}$$

Zusätzlich bekommen wir den Durchmesser des Stammes aus dem Radius R_A. Dieser wird dem AutoCAD-Linienobjekt als zusätzliche Eigenschaft angehängt. Dafür wird die Eigenschaft „Linientypfaktor“ verwendet, da sie eine rationale Zahl beinhalten kann. Für die Bestimmung des Durchmessers werden aber nur die Randpunkte des Endpunktes verwendet, da sie beim Konstruieren immer gemessen werden müssen. Das ergibt sich aus der Konstruktionsmethode: Der Anfangspunkt ist in der Regel der bereits gezeichnete und in der Achse des Astes gelegene Endpunkt der davor konstruierten Linie. Dieser wird per AutoCAD-Objektfang bestimmt.

6.5.5 Makro „Transformation“

Dieses Makro dient zur Transformation der gezeichneten Strukturen, also aller Linien im lokalen Plattkartensystem, ins Landeskoordinatensystem. Dies erfolgt durch die Kongruenztransformation wie in den Gleichungen (Gl. 5) und (Gl. 6). Dabei müssen die Koordinaten des Polarkoordinatensystem der Plattkarte in ein lokales, kartesisches Koordinatensystem umgewandelt werden

$$\begin{pmatrix} x_l \\ y_l \\ z_l \end{pmatrix} = D_l \cdot \begin{pmatrix} \sin(H_l) \cdot \sin(V_l) \\ \cos(H_l) \cdot \sin(V_l) \\ \cos(V_l) \end{pmatrix} \qquad \text{(Gl. 28)}$$

D_l .....Entfernung
H_l ..... Horizontalwinkel
V_l .....Vertikalwinkel

Danach transformiert:

$$x_I = R \cdot x_{LK} + t$$

$$\begin{pmatrix} x_{LK} \\ y_{LK} \\ z_{LK} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{12} & r_{22} & r_{23} \\ r_{13} & r_{23} & r_{33} \end{pmatrix} \cdot \begin{pmatrix} x_I \\ y_I \\ z_I \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \qquad \text{(Gl. 29)}$$

r_{xx} ..... Rotationselemente t_x ..... Translationselemente

6.6 Modellierung in den Testgebieten

6.6.1 Grundsätzlicher Ablauf

Die Auswertung erfolgt nach der erweiterten Modellierungsstrategie in Kapitel 6.4.2. Dafür ist ein Personal Computer mit entsprechend hoher Leistung erforderlich. In diesem Fall wird ein Prozessor AMD A4-3400 APU mit 2,70 GHz verwendet, dazu ein Arbeitsspeicher (RAM) von 8,00 GB. Es sind für das Laden der Daten mindestens 2 GB RAM notwendig. Damit ist der Arbeitsspeicher ausreichend. Die Auswertung beginnt mit dem Import des ersten Standpunktes eines der drei ausgesuchten Gebiete (2x Lägeren, 1x Tharandt) durch das Makro „Standpunkt“ (siehe Kapitel 6.5.3). Das Lesen und Importieren der Entfernungsdaten aus der gespeicherten Textdatei erfordert eine Zeitdauer von ca. 5 Minuten. Das Laden und Importieren der Plattkarten erfordert hingegen keine nennenswerte Zeitdauer. Damit ist die Plattkarte in der AutoCAD-Zeichenebene dargestellt. Mit Hilfe der Makros „Linienzeichnen“, „Schnittdarstellung“ und „Entfernungsbild“ (Kapitel 6.5) können nun alle sichtbaren Stämme und Äste der dargestellten Bäume konstruiert werden. In diesem Standpunkt konstruierte Linien sind auf einem eigenen Layer und allgemein rot dargestellt. Linien, deren Entfernungsdifferenz der Endpunkte 50 cm übersteigen, sind grün dargestellt (Abbildung 33). Somit können fehlerhafte Abgriffe detektiert werden, das aber nicht eindeutig ist, da bei größerem Vertikalwinkel diese große Differenz normal sein kann. 50 cm Differenz bilden einen guten Kompromiss in der Fehlerdetektierung.

Abbildung 33 : gezeichnete Strukturen im ersten Standpunkt

Die gezeichneten Baumstrukturen werden anschließend mit dem Makro „Transformation“ in das Landeskoordinatensystem der Schweiz (Lägeren) und Deutschlands (Tharandt) transformiert (Abbildung 34).

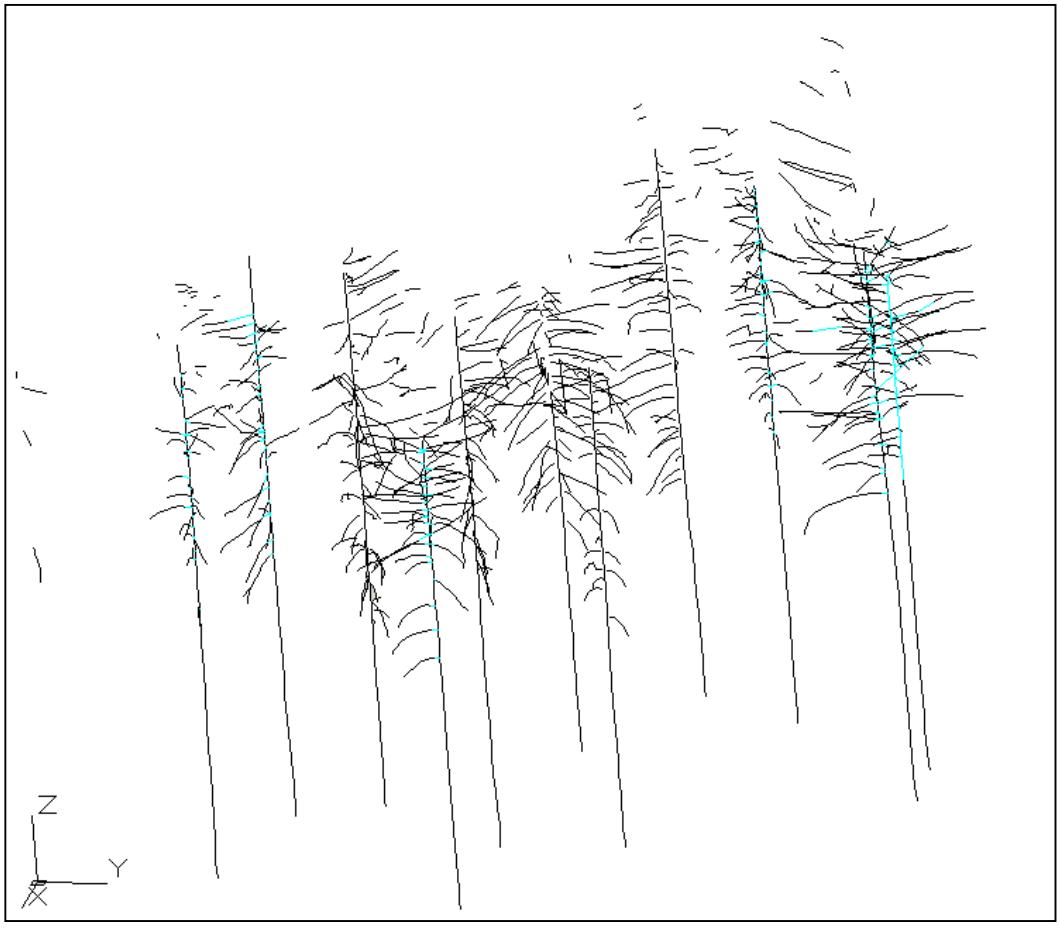

Abbildung 34 : Struktur des ersten Standpunkts im Landeskoordinatensystem

Der zweite Standpunkt wird nun ebenfalls mit dem Makro „Standpunkt“ importiert. Zusätzlich zu den Entfernungsdaten und den Plattkarten werden aber auch die bereits

konstruierten Linien des ersten Standpunktes importiert und ins lokale Koordinatensystem des zweiten Standpunktes transformiert. Diese sind auf einem eigenen Layer blau dargestellt und stellen nahezu exakt einige Äste und Stämme auf der Plattkarte da. Erwartungsgemäß sind hier Fehler und Abweichungen vorhanden, welche im ersten Standpunkt nicht zu sehen waren und hier leicht auffallen. Diese werden nun korrigiert. Äste, die noch nicht gezeichnet worden sind, werden wieder rot bzw. grün gezeichnet (Abbildung 35).

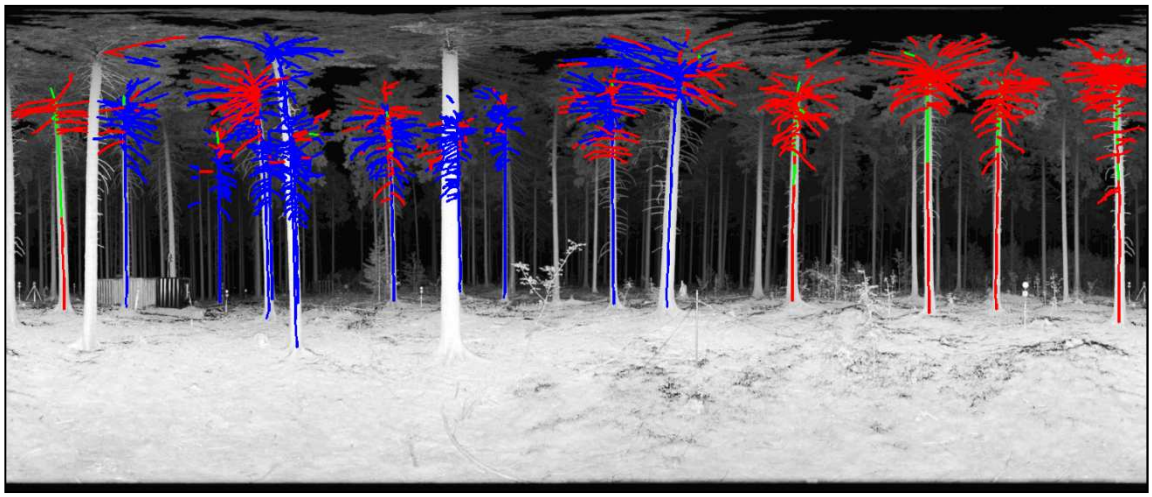

Abbildung 35 : gezeichnete Strukturen nach dem zweiten Standpunkt

Sobald alle sichtbaren Äste und Stämme gezeichnet sind, werden alle Linien, einschließlich der Linien des ersten Standpunktes ins Landeskoordinatensystem transformiert. Für alle restlichen Standpunkte des Aufnahmegebiets wird auf dieselbe Art vorgegangen. Alle AutoCAD-Zeichnungen im Konstruktionsprozess werden stets abgespeichert. Das heißt, für jede Zeichnung (im lokalen Polarkoordinatensystem) und für jede transformierte Struktur (im Landeskoordinatensystem) wird pro Standpunkt eine Datei im AutoCAD-eigenen Format *.dwg angelegt. Das ist wichtig, um die lokale Linienkonstruktion nachvollziehen zu können.

6.6.2 Zeichentechnik

Auch wenn durch das Makro „Linienzeichnen“ die Linie automatisch entlang der Astachse gesetzt werden kann, so ist bei der Konstruktion eines ganzen Baumes mit Biegungen, Verengungen, Vergabelungen und Abzweigungen von Ästen zu rechnen. Dafür ist schon während des Zeichnens eine visuelle Interpretation erforderlich. Räumliches Vorstellungsvermögen ist wichtig, um zu wissen, wo ein Ast anfängt und wo er endet. Schließlich ist für Abzweigungen und Astgabelungen ein Linienendpunkt erforderlich.

Zusätzlich ist wichtig, zu schätzen, wie kurz die Linien sein müssen, um eine Astbiegung zu repräsentieren.

Wichtigste Interpretation beim Zeichnen ist die Identifizierung von Ästen. Wie bereits in Kapitel 6.2 erwähnt, wird als vorrangige Plattkarte ein Intensitätsbild verwendet. Objekte sind hier wie auf einer Photographie sichtbar, aber mit einer deutlich höheren Schärfe, was viele Details sichtbar macht. Trotzdem gibt es im Intensitätsbild Schwankungen in der Rückstrahlintensität, die den Verlauf eines Astes unkenntlich machen kann. Dies ist vor allem bei dünnen Ästen (Durchmesser von unter 3 cm) der Fall. Zusätzlich tauchen beträchtliche Lücken auf, die überbrückt werden müssen. Oft sind es andere Äste, doch vor allem Blatt- und Nadelwerk, das große Lücken verursacht. Diesbezüglich hilft auch das für jeden Standpunkt mitimportierte, farbige Entfernungsbild, das bei der Identifizierung der Äste helfen kann, auch wenn die Farbkodierung nur Entfernungsunterschiede von mehr als 10 cm sichtbar macht. Trotzdem sind ein kleiner Teil der Äste von Fehlern in der Entfernung und von falschen Verbindungen betroffen. Überkreuzungen von Ästen führen zu Verwirrung und zu Entfernungssprüngen und sind dann erst im nächsten Standpunkt sichtbar (Abbildung 36).

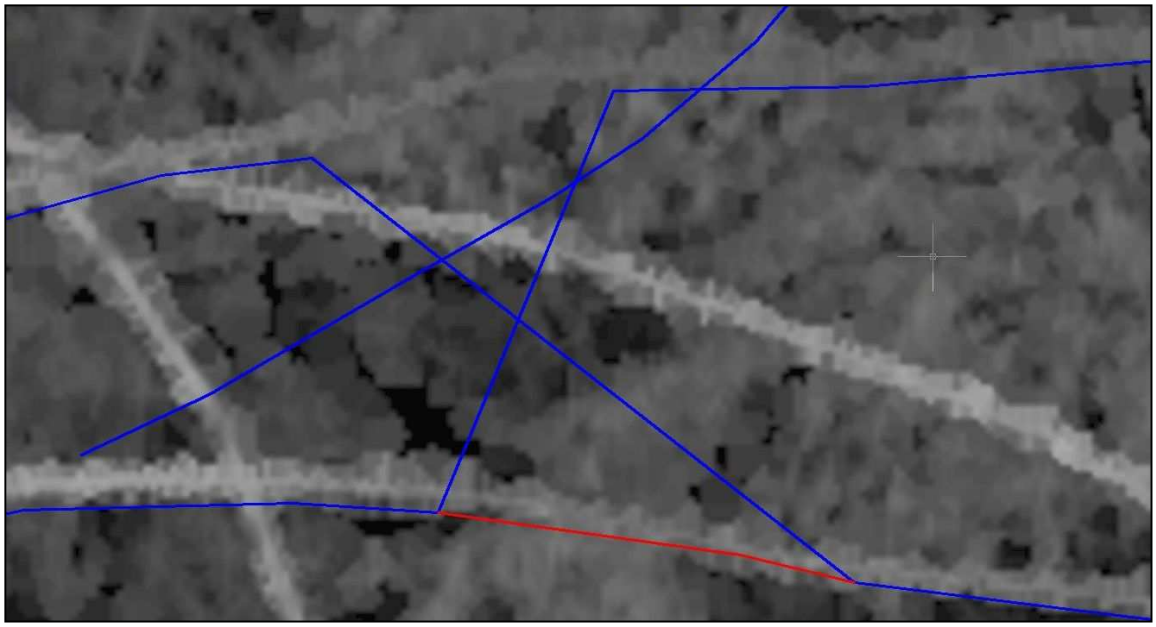

Abbildung 36 : Sprünge zwischen Ästen sind beim Konstruieren möglich und müssen von anderen Standpunkten aus ausgebessert werden

In den anderen Standpunkten sind Fehler eindeutiger zu beobachten und können somit beseitigt werden. Linien von anderen Standpunkten (Abbildung 36, blau) werden hier einfach neu gezeichnet oder verschoben. Im stark verzerrten oberen Bereich der Plattkarte wird die Konstruktion generell vermieden. Hier entstehen sonst große Abweichungen. Um

trotzdem von den Baumkronen möglichst viel zu rekonstruieren, werden diese vor allem auf Plattkarten weiter entfernter Laserscanning-Standpunkte gezeichnet, da sie dann geringer verzerrt sind. Problematisch ist auch das Erscheinungsbild von Nadelbaumästen. Sie sind in der Regel dicht mit Nadeln bewachsen und oft nur durch visuelle Interpretation zu detektieren (Abbildung 37).

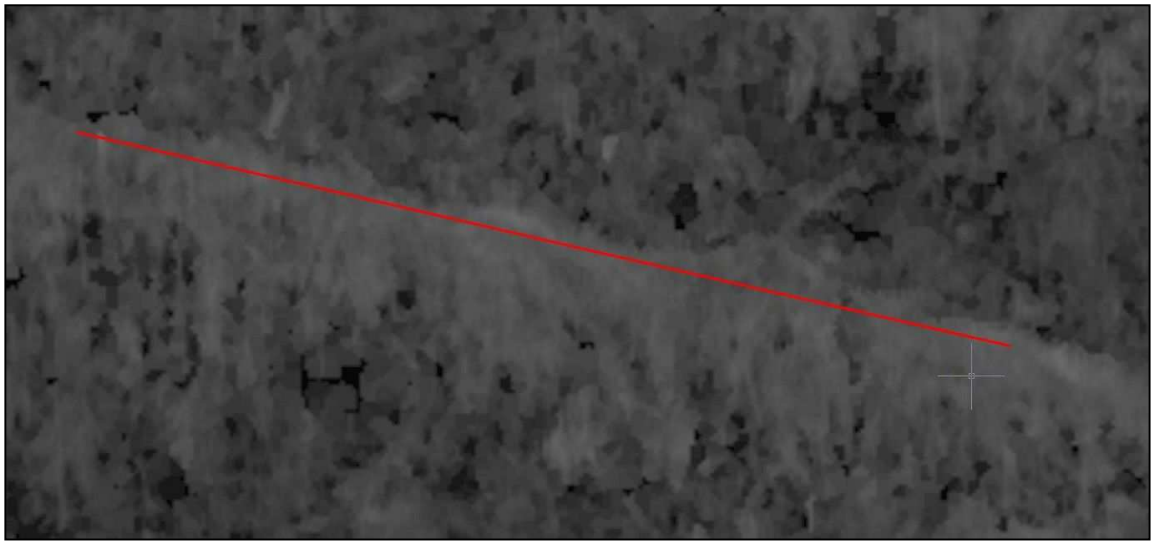

Abbildung 37 : Nadelbaumäste sind durch ihre Nadeln auf dem Intensitätsbild schwer erkennbar.

Ein wichtiger Punkt ist die entsprechende Gewichtung der Äste. Dicke Äste sind geometrisch von größerem Interesse als dünnere Äste. Das betrifft vor allem auch den Übergang vom Baumstamm zu seinen Ästen. Es ist wichtiger, den Stamm vorrangig mit höherer Genauigkeit zu modellieren. Wenn der Stamm jedes Mal nur bis zum nächsten Ast gezeichnet wird, bekommt der Stamm, trotz seiner in der Regel geradlinigen Erscheinung, eine unförmige, zackige Struktur. Deshalb werden die Äste, die zum Stamm führen, beim Zeichnen nur „angehängt“ (Abbildung 38). Anschließend werden die Stammlinien an den Anhängepunkten geteilt. Dies wird durch die mächtigen Konstruktionswerkzeuge in AutoCAD bewerkstelligt.

Die meisten Linien sind mit anderen Linien per AutoCAD-Objektfang verbunden. Trotzdem bleiben diese Linien mit ihren benachbarten Linien nach den eigentlich unabhängigen Einzeltransformationen verbunden. Dies gilt nicht für „angehängte“ Linien, welche zwischen Anfangs- und Endpunkt verbunden sind (Abbildung 38). Da entstehen durch die Transformation unvermeidbare Abweichungen. Daher sollten „angehängte“ Linien vermieden und dieses Problem noch vor der Transformation durch Linienteilung behoben werden.

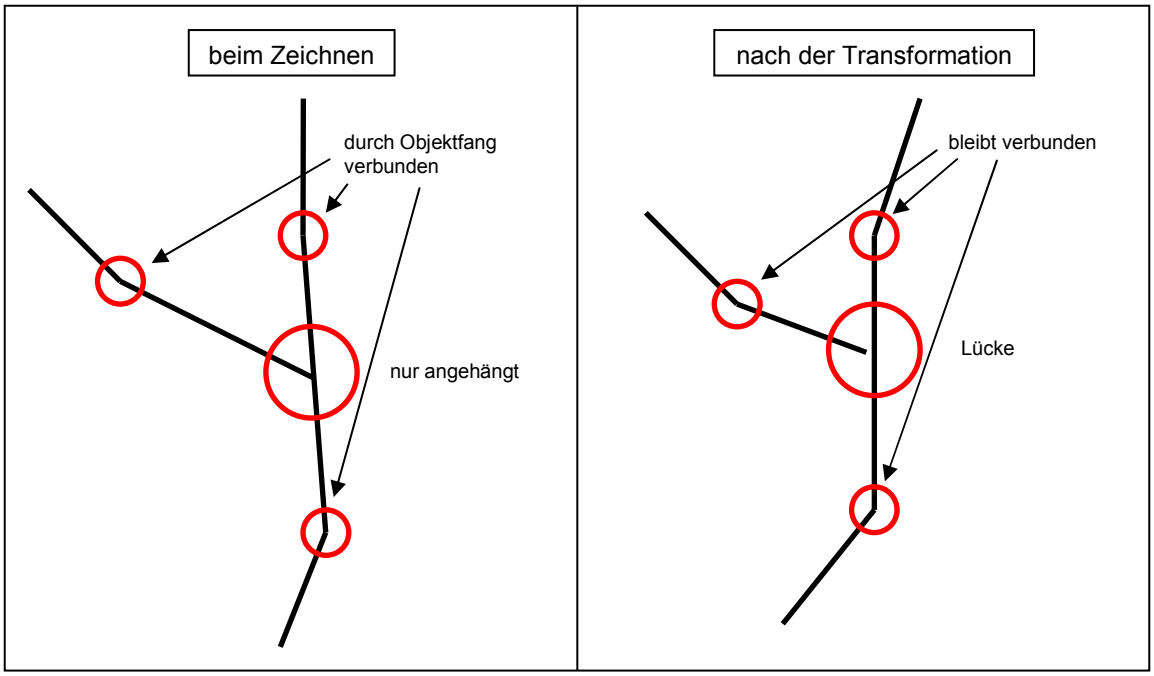

Abbildung 38 : Angehängte Linien bei Transformation

An den Baumstrukturen müssen trotz der ständigen Kontrolle durch andere TLS-Standpunkte viele Ausbesserungen durchgeführt werden. So tauchen Ausreißer in Form von Zacken auf, die durch Missinterpretationen der Aststruktur im Intensitäts- und Entfernungsbild entstehen. Diese Zacken werden durch freie Schätzung und Interpretation mit einer neuen Linie verkürzt und dieser Linie der Durchmesser der gelöschten Linien zugewiesen. Fliegende Äste, die bei der Konstruktion gezeichnet werden und trotz verschiedenster Standpunkte nicht mit dem Stamm oder größeren Ästen verbunden sind, werden mit diesen verbunden, solange es möglich ist. Wenn das nicht möglich ist, werden sie dennoch nicht gelöscht. Trotz nicht vorhandener Plattkarte und sonstiger Referenzdaten ist auch eine eingeschränkte 3-dimensionale Nachbearbeitung an den Strukturen im übergeordneten Landeskoordinatensystem möglich. Denn auch hier sind unnatürliche Ausreißer in Form von Zacken zu sehen. Diese werden beseitigt, indem die Linien um die Zacken verkürzt werden. Dabei muss aber die Durchmessereigenschaft für die neu geschaffenen Linien kopiert werden.

6.6.3 Überprüfung der Strukturentransformationen

In Abbildung 39 ist der vorgesehene Konstruktionsablauf dargestellt, wie er in Kapitel 6.6.1 beschrieben wird. Dabei werden nach jedem Zeichnen auf der Plattkarte des Standpunktes die gesamten Linien ins Landeskoordinatensystem transformiert und diese in den nächsten Standpunkt rücktransformiert. Am Ende wird die exportierte Struktur des letzten,

verwendeten Standpunktes 3-dimensional nachbearbeitet und ausgebessert. Diese ist schließlich das Endergebnis der gesuchten Baumstruktur. Bei diesem Konstruktionsablauf werden z.B. im Testgebiet Tharandt 4.1.2 die Linien des 1. Standpunktes bis zum Endergebnis 66-mal transformiert.

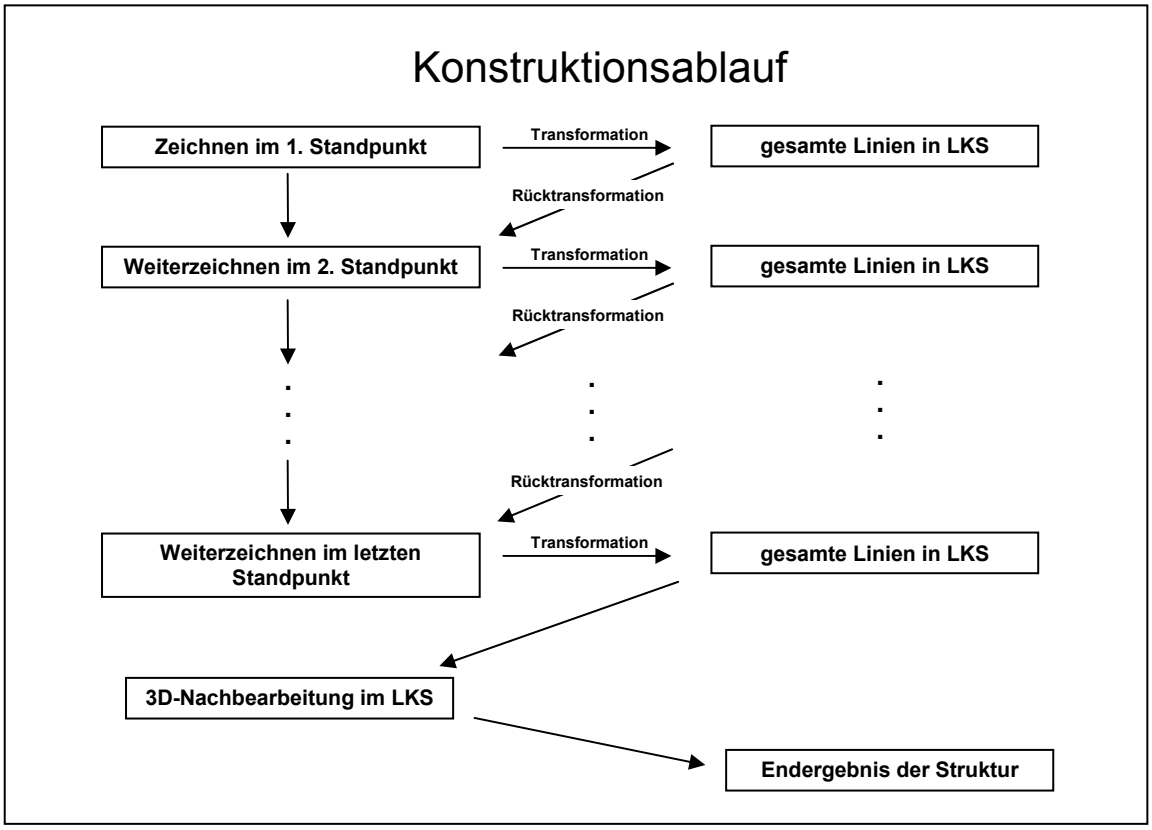

Abbildung 39 : Konstruktionsablauf durch Strukturentransformationen

Wichtig für das Endergebnis ist die richtige Position der Äste im Landeskoordinatensystem. Dabei könnte es durch die Transformation zu einer Veränderung der 3-dimensionalen Lage auf Grund der Grenzen in der numerischen Rechengenauigkeit kommen. In diesem Fall ist vor allem die Verschiebung in der Lage zu beachten. Wenn die Transformation fehlerhaft ist, so wird sich dieser Fehler mit jeder Transformation stärker auswirken. Es ist daher interessant diese Auswirkungen näher zu überprüfen. Aus diesem Grund wird ein Überprüfungsablauf (Abbildung 40) festgelegt. In diesem werden von jedem Standpunkt nur die neu hinzugefügten Linien ins Landeskoordinatensystem transformiert. Insgesamt wird dadurch jede Linie nur einmal transformiert, wodurch es nur zu geringer Fehlerfortpflanzung kommt.

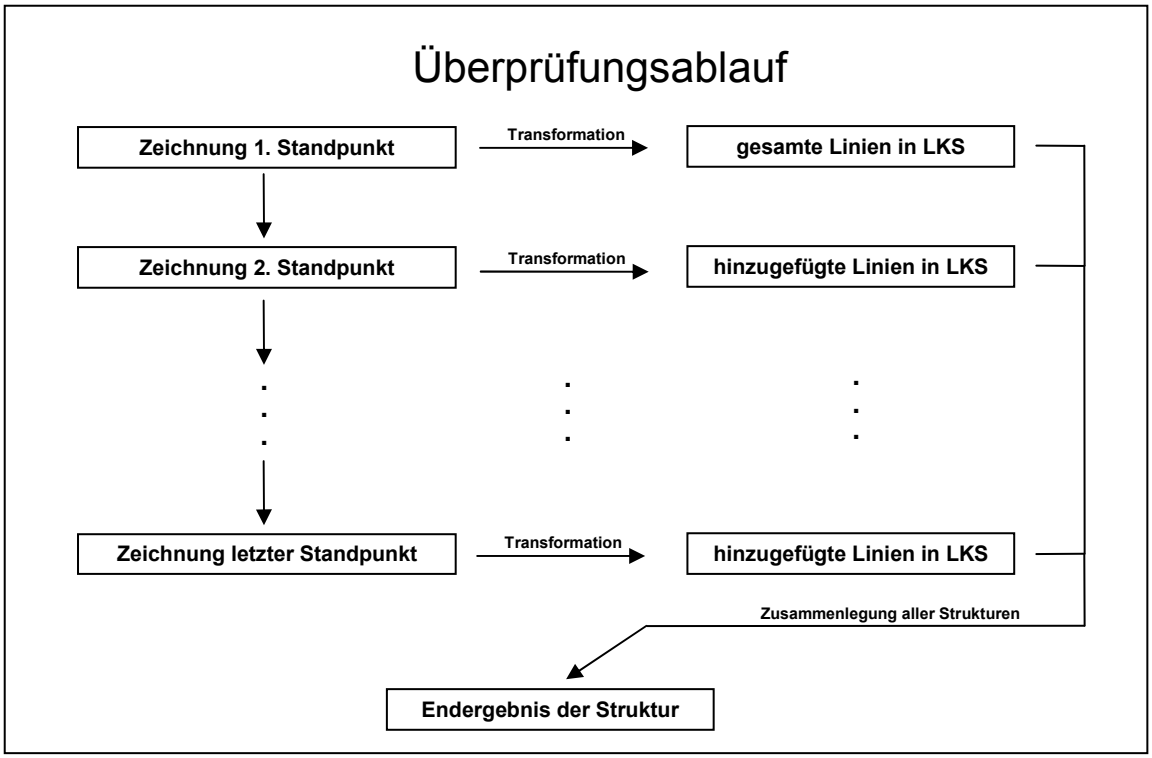

Abbildung 40 : Überprüfungsablauf

Die durch die Transformationen mögliche Fehlerfortpflanzung auf Grund der numerischen Rechenungenauigkeit ist beim Vergleich des Konstruktionsablaufs und des Überprüfungsablaufs nicht sichtbar (Abbildung 41). Die Linien stimmen exakt überein. Es findet praktisch keine Fehlerfortpflanzung statt.

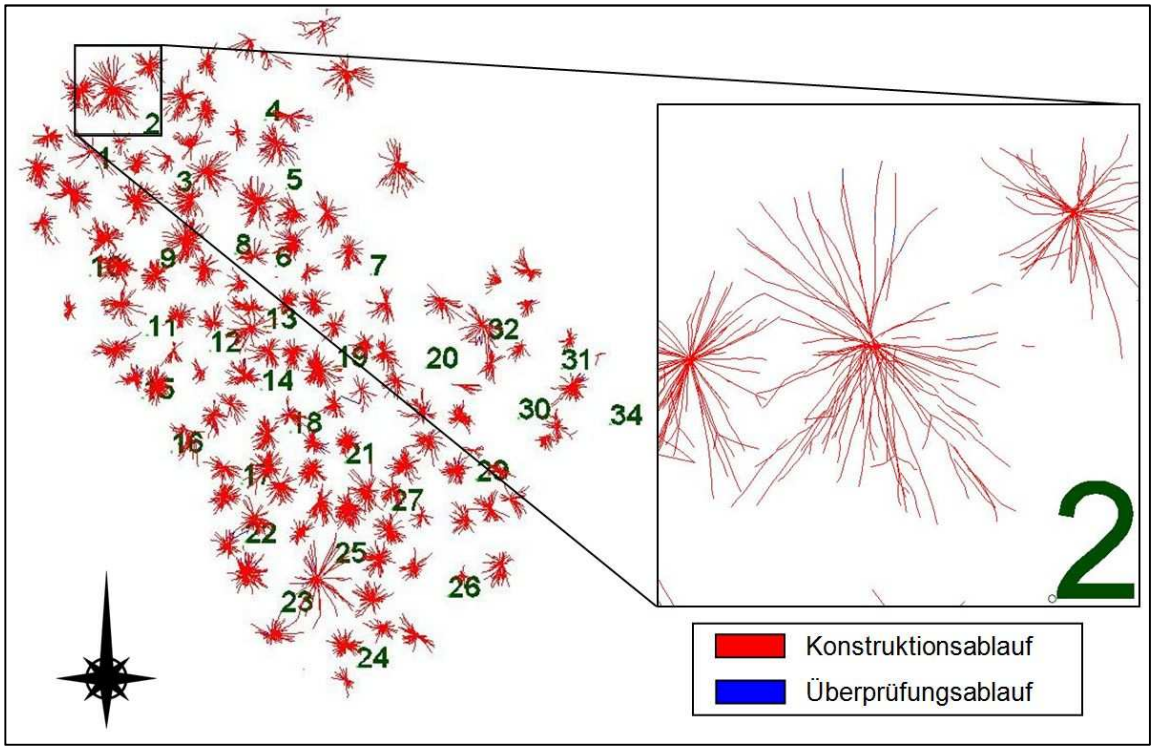

Abbildung 41 : Vergleich des Konstruktionsablaufs (Tharandt) mit dem Überprüfungsablauf

7 Ergebnisse und Diskussion

7.1 Ergebnis der Softwaretechnischen Umsetzung

Das Ergebnis der Sofwareentwicklung sind 7 Makros in AutoCAD (Abbildung 42), bestehend aus 1200 Zeilen Programmcode in VBA (siehe Anhang), sowie 2 Scripts in Matlab und 1 Script in OPALS, die der Vorbereitung zur eigentlichen Modellierung in AutoCAD dienen. Für das Schreiben dieser Programme war ein Zeitaufwand von mehreren Monaten notwendig, wobei die Entwicklung parallel zur Auswertung erfolgte. Viele Programmbefehle flossen erst durch Ideen aus der praktischen Baummodellierung in den Testgebieten in die Makros ein. So wurde das Makro „Schnittdarstellung“ und die Idee der Querschnittdarstellung der Punkte durch die fehlerhafte Liniensetzung inspiriert.

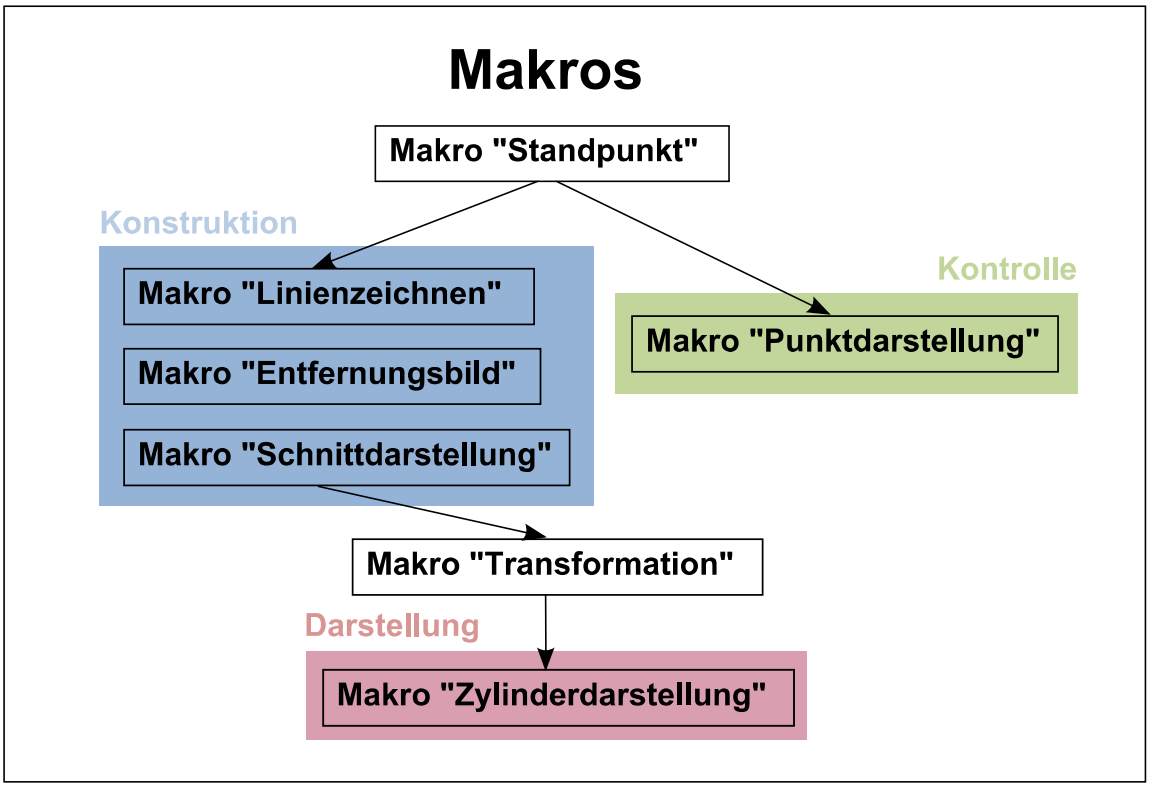

Abbildung 42 : Übersicht über die erstellten 7 VBA-Makros und ihre Verwendung

Makro „Standpunkt“

Das Makro bildet den Hauptanteil an Funktionen im dritten Schritt der erweiterten Modellierungsstrategie (Kapitel 6.4.2) ab und ist in Kapitel 6.5.3 bereits hinreichend erklärt worden. Es startet ein Formblatt, mit dem alle notwendigen Dateien und des jeweiligen Laserscanning-Standpunktes in AutoCAD importiert werden. Das Formblatt enthält 3 Schritte, welche optional mit Kontrollkästchen ausgewählt werden können. Das Makro

„Standpunkt“ ist Voraussetzung für die Makros „Punktdarstellung“, „Entfernungsbild“, „Linienzeichnen“, „Schnittdarstellung“ und „Transformation“.

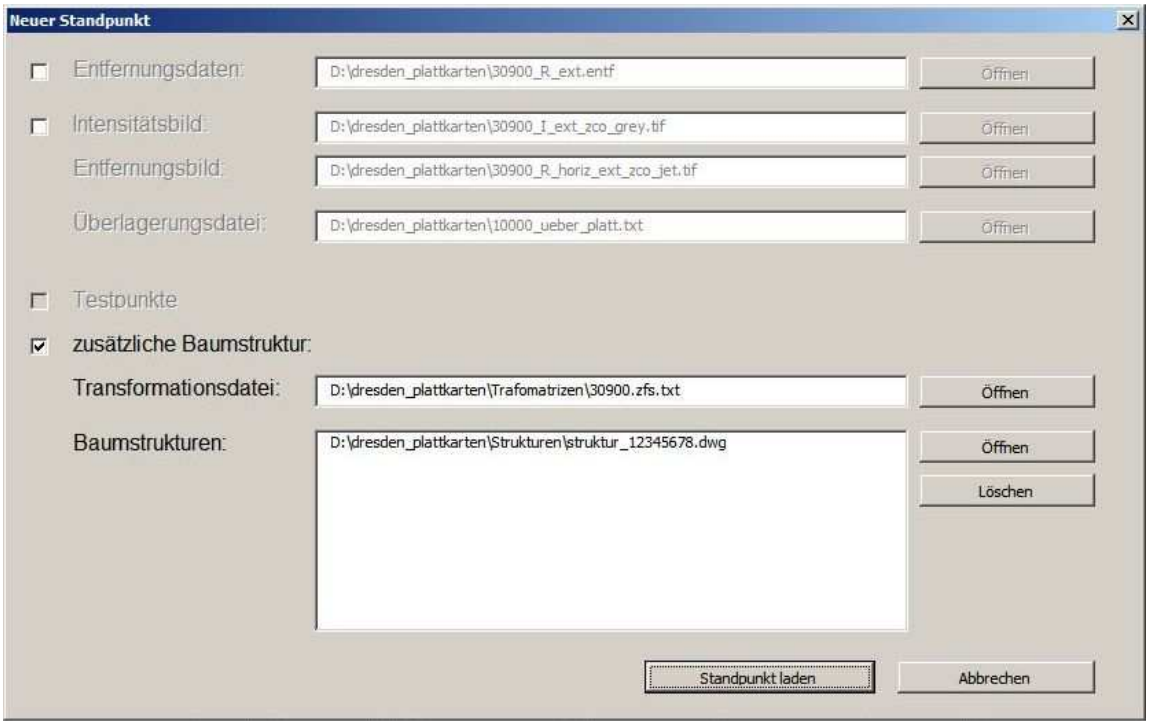

Abbildung 43 : Formblatt des Makros „Standpunkt“, AutoCAD

Makro „Transformation“

Dieses Makro dient zur Transformation der gezeichneten Strukturen, also aller Linien im lokalen Plattkartensystem, ins Landeskoordinatensystem. Es wurde als wichtiger Teil der Auswertung bereits in Kapitel 6.5.5 ausführlich erklärt. Es öffnet dazu ein kleines Formblatt. Dabei muss die Transformationsdatei des jeweiligen Standpunktes geöffnet, sowie ein Name des Layers für die Struktur gewählt werden. Nach Bestätigung wird eine neue AutoCAD-Zeichenebene gestartet, in welcher die transformierte Struktur im angegebenen Layer dargestellt wird. Die Zeichenebene kann nun als dwg-Datei gespeichert werden und im nächsten Standpunkt reimportiert werden.

Abbildung 44 : Formblatt des Makros „Transformation“, AutoCAD

Makro „Linienzeichnen“

Das Makro „Linienzeichnen“ ist Basis der entwickelten Methode zur Baummodellierung in Kapitel 6.4 und ist daher in Kapitel 6.5.4 hinreichend erklärt worden. Für dieses Programm ist keinerlei Dialogfenster notwendig. Für die Anwendung dieses Makros ist das Makro „Standpunkt“ Voraussetzung.

Makro „Schnittdarstellung“

Dieses Makro ist ein wichtiges Hilfsmittel bei der Linienkonstruktion. Beim Setzen eines Endpunktes der Linie, kommt es vor, dass der Stamm bzw. Ast durch andere Objekte (z.B. Blätter, Zweige) teilweise verdeckt ist. Dadurch wird die abgegriffene Entfernung verfälscht. Um das zu vermeiden, ist eine manuelle Korrektur der Entfernung notwendig. So wird bei einem zu starken Entfernungsunterschied zwischen dem Anfangspunkt und dem Endpunkt der gesetzten Linie (prinzipiell frei wählbar, praktisch sind 50 cm üblich) beim Ausführen des Makros „Linienzeichnen“ dieses Makro ausgeführt. Es führt zu einer automatischen Ansichtsänderung in AutoCAD, in der Punkte des Entfernungsarrays, welche zwischen den Randpunkten liegen, in einer Schnittebene dargestellt werden (Abbildung 45). Diese bilden die Objekte einschließlich des Stammes ab. So kann die Entfernung des Punktes anhand der Punktverteilung neu ausgewählt werden. Nach Auswahl wird die alte Zeichenansicht wieder angezeigt und die Konstruktion kann fortgesetzt werden.

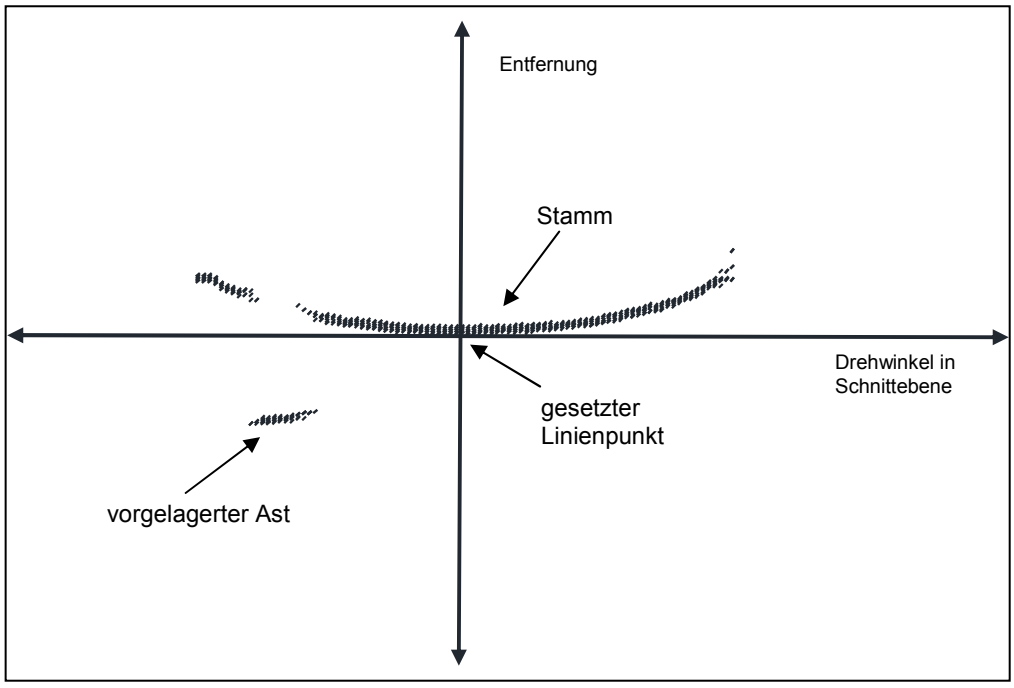

Abbildung 45 : Schnittebenendarstellung der Punkte des Entfernungsarrays

Makro „Punktdarstellung“

Dieses Makro dient zur Kontrolle der richtigen Position der Plattkarten in der AutoCAD Zeichnung. Es ermöglicht an einem ausgewählten Punkt der Karte Punkte des Entfernungsarrays darzustellen (Abbildung 46). Punkte mit der Entfernung 0 werden nicht angezeigt. Damit ist die richtige Überlagerung mit den Plattkarten erkennbar. Es ist auch erkennbar, ob das Entfernungsarray überhaupt mit den Plattkarten übereinstimmt und damit falsch importiert worden ist.

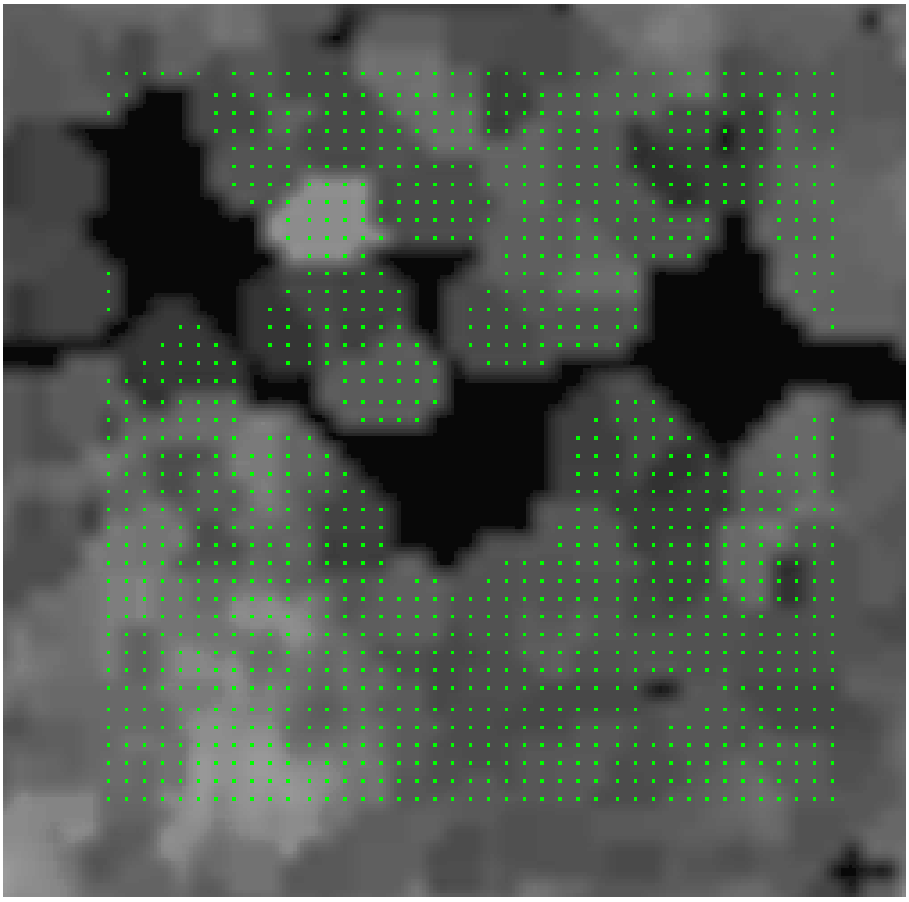

Abbildung 46 : Punkte des Rasters werden in einem ausgewählten Bereich dargestellt

Makro „Entfernungsbild“

Standardmäßig wird auf dem Intensitätsbild gezeichnet. Zum Ein- und Ausblenden des Entfernungsbildes kann dieses Makro gestartet werden. Das funktioniert, da das Intensitätsbild sowie auch das Entfernungsbild auf ihren eigenen AutoCAD-Layer liegen.

Makro „Zylinderdarstellung“

Diese Makro erstellt aus bestehenden 3-dimensionalen Linien die dazugehörigen Zylinder. Der Zylinderdurchmesser wird aus der Eigenschaft „Linientypfaktor“ der jeweiligen Linienobjekte genommen. Somit ist eine realistisch anmutende Baumdarstellung mit Körper möglich, welches mit AutoCAD beliebig gerendert werden kann.

7.2 Ergebnis der Modellierung in den Testgebieten

Ergebnis der Anwendung der Methode in den Testgebieten sind vollständig konstruierte Strukturen der Bäume in den beiden Gebieten in Lägeren, Schweiz und im deutlich größeren Gebiet in Tharandt bei Dresden, Deutschland.

Mit Hilfe des Makros „Zylinderdarstellung“ werden für die Linien mit deren Durchmessereigenschaft Zylinder automatisch konstruiert und auf einem eigenen Layer untergebracht. AutoCAD hat die Möglichkeit diese zu rendern und so eine körperhafte Darstellung der Strukturen zu erstellen (Abbildung 47).

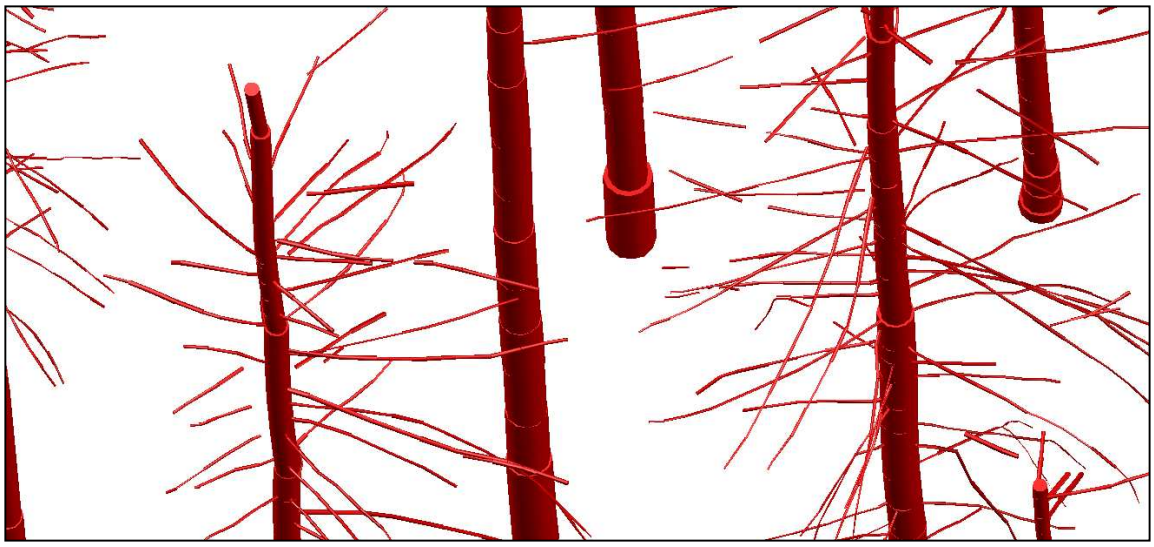

Abbildung 47 : Zylinderdarstellung der Baumstrukturen (gerendert), AutoCAD

Prinzipiell ist schon bei den Plattkarten festzustellen, dass dem Laserstrahl des Scanners vor allem im Baumkronenbereich viele Hindernisse im Weg stehen. Das Blattwerk und die Nadeln verhindern, dass Äste vollständig ohne Lücken zu sehen sind und aufgenommen werden können. Damit sind kaum Punkte auf den Ästen der oberen Schichten der Baumkrone vorhanden. Damit sind die Grenzen jeglicher Modellierungsmethoden, ob automatisch oder manuell, erreicht. Wo keine Punkte vorhanden sind, kann kein Ast

rekonstruiert werden. Allerdings können Äste lückenweise erfasst werden, wobei vom selben Ast die Oberflächenpunkte sich auf mehrere Scans verteilen kann. Dadurch kann ein Ast von jeder einzelnen Plattkarte aus kaum erfasst werden, aber in der georeferenzierten Gesamtpunktwolke als Punktröhre erkennbar sein, weil er sich aus den Punkten mehrerer Scans zusammensetzt. Das ist einer der wenigen größeren Nachteile der Modellierung anhand von Plattkarten. Nichtsdestotrotz reichen viele detektierte Äste bis hoch in die Baumkronen hinauf, das betrifft vor allem die Laubbäume. Deren Äste sind oft deutlich klarer in der Struktur zu sehen, als die der Nadelbäume. Die Spitzen der Nadelbäume sind oft nur von weiteren, entfernten Standpunkten zu sehen. Die stark benadelten Äste sind oft in ihrer Struktur nur zu erraten.

Das Ergebnis der Baummodellierung lässt sich graphisch auch gut mit einem generierten Digitalen Geländemodell (DGM) und einem Digitalen Oberflächenmodell (DOM) kombinieren. In Abbildung 48 ist das DOM grün dargestellt und repräsentiert die Oberfläche der Baumkronen. Das DGM ist grau dargestellt und bildet den lokalen Waldboden ab. Die Stämme schließen gut an die Oberflächendarstellung des DGM an. Die Äste reichen teilweise bis zur Fläche des DOM hinauf, wobei es da von Baum zu Baum Unterschiede gibt.

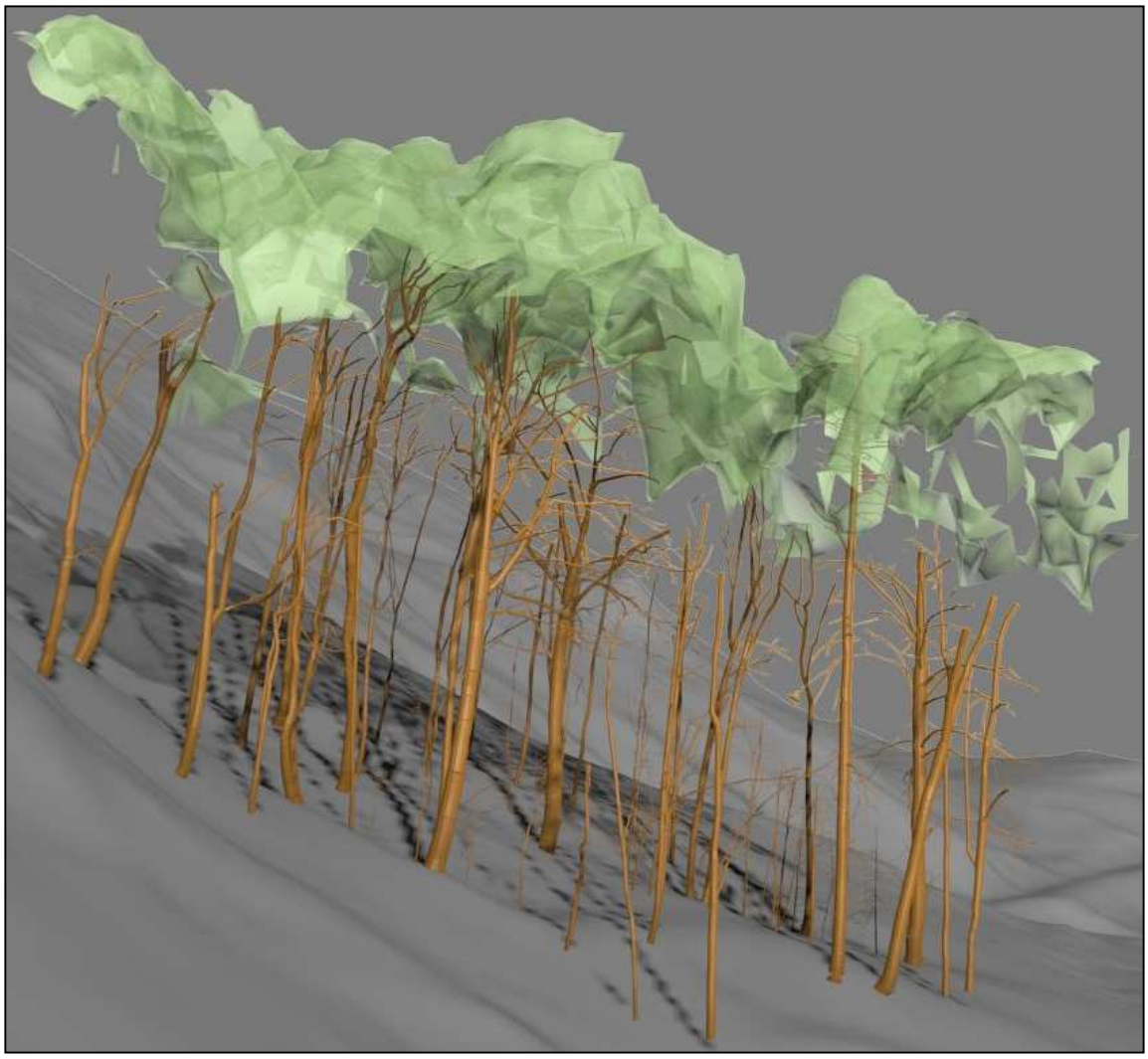

Abbildung 48 : Zylindermodell des Ergebnisses mit DGM (grau) und DOM (grün) – Lägeren, westliches Gebiet

8 Zusammenfassung und Ausblick

Für das Projekt „3DVegLab“ wurden Waldgebiete in Lägeren (Schweiz) und Tharandt bei Dresden (Deutschland) mit Hilfe des terrestrischen Laserscanners Z+F IMAGER® 5006h aufgenommen. Dafür wurde eine genauere Übersicht über die Messverfahren und aktuellen Messgeräte geschaffen. Schließlich wurde ein Aufnahmeplan geschaffen, der an die entsprechenden Zwecke angepasst wurde. An Hand dessen wurden im Rahmen von mehrtägigen Messkampagnen die Gebiete aufgenommen. Nach einer Georeferenzierung wurden die Punktwolken geschaffen. Aus den Punktwolken der Waldgebiete soll nun ein Linienmodell für die darin befindlichen Bäume geschaffen werden. Die Linien des Modells sollen zusammen mit den dazugehörigen Durchmessern die Stämme und Äste exakt in ihrer Geometrie repräsentieren. Dazu wurden einige in Publikationen präsentierte Methoden der Modellierung analysiert, welche alle eine umfangreiche Automatisierung aufweisen. Für das Projekt wurde aber eine Methode entwickelt, implementiert und in der Praxis an den aufgenommenen Punktwolken angewandt, die sich durch bessere Vollständigkeit und höherer geometrischer Korrektheit auszeichnet. Die Methode basiert auf manuelles Nachzeichnen der Baumstrukturen auf den Intensitätsplattkarten der Aufnahmestandpunkte. Das Zeichnen wurde in AutoCAD mit eigens für die Anwendung programmierten Makros bewerkstelligt und das Ergebnis im jeweiligen Landeskoordinatensystem dargestellt.

In der in Kapitel 6.4 festgelegten und in Kapitel 6.6 durchgeführten Plattkartenmethode zur Baummodellierung werden die Strukturen der Bäume manuell mit Hilfe eines Zeichenprogramms wie AutoCAD erstellt. Wie in Kapitel 6 erwähnt, ist im Projekt „3DVegLab“ die Zeitdauer zur Modellierung nicht bedeutend. Es werden nur relativ kleine Testgebiete modelliert. Für großflächigere Gebiete ist die Plattkartenmethode in dieser Form zu aufwendig, langsam und kostspielig. Eine Automatisierung steht damit im Raum. Es wäre möglich, diesen Vorgang der Modellierung teilweise oder gänzlich durch automatische Linienziehung zu ersetzen. Automatische Berechnungen werden in der Methode definitionsgemäß nicht durchgeführt. Denn findet zwar ein automatischer Entfernungsabgriff aus dem Array statt, doch der Linienpunkt wird manuell an Hand von Interpretation ausgewählt. Auch der Durchmesser wird aus der manuellen Auswahl der Randpunkte abgeleitet.

Anstatt einer Vollautomatisierung wie in den bereits vorgestellten Methoden in Kapitel 0, die gewisse Nachteile in der Auswertung besitzen, könnte die in dieser Arbeit durchgeführte Methode schrittweise teilautomatisiert werden. Zum Beispiel wäre die Durchmesserbestimmung ohne die Setzung von Randpunkten eine hilfreiche Automatisierung. Für die Bestimmung könnten die Punkte verwendet werden, die beim Makro „Schnittdarstellung“ dargestellt werden. Für die Detektion des gesuchten Punktes in der Schnittebene könnte die Kreis-Hough-Transformation benutzt werden. Trotzdem ist Skepsis angebracht. Schließlich sind kleine Äste im Scan oft nur wenige Aufnahmepunkte breit. Eine solche Bestimmung ist dann mit Sicherheit fehlerbehaftet.

Es wird für eine konsequente Automatisierung auch nichts anderes übrig bleiben, Entfernungsabgriff und Durchmesserbestimmung eines Linienpunktes von vorangegangen/benachbarten Linienpunkten und deren Entfernungen und Durchmesser beeinflussen zu lassen. Das heißt, bei den Punkten aus der Schnittebene sollen für die Detektion (Kreis-Hough-Transformation) nur die Punktmenge verwendet werden, die eine ähnliche Entfernung aufweisen, wie vorangegangene/benachbarte Linienpunkte. So könnten gravierende Fehldetektionen verhindert werden.

Die nächste wichtige Teilautomatisierung währe die automatische Linienziehung entlang der Äste und Stämme. Eine solche Automatisierung müsste die Fläche aus Pixeln, welche den Ast und den Stamm auf dem Intensitätsbild und auch dem Entfernungsbild abbilden, erkennen und durch diese Linien ziehen. Allerdings müsste hier voller Zugriff auf die Pixelinformation des Intensitätsbildes vorhanden sein, was AutoCAD nicht gewährleistet. Es müsste sonst auf die Intensität genauso zugegriffen werden, wie auf das Entfernungsarray. Das könnte ineffizient sein. Somit könnte, da das Zeichnen durch die Automatisierung mehr oder weniger obsolet wird, die gesamte Modellierung nicht mehr in AutoCAD sondern in einem schnelleren, mächtigeren Rechenprogramm durchgeführt werden. Das Programm Matlab (Matrix Laboratory) wäre dafür geeignet, Bilder zu lesen, Bildverarbeitung durchzuführen und ein Linienmodell, zumindest als Zahlenstruktur in einer Textdatei zu erstellen.

Digitale Bildverarbeitung von 2-dimensionalen Plattkarten wäre einfacher zu implementieren als die 3-dimensionalen Algorithmen mit Voxeleinteilung in den Methoden in Kapitel 0 und wäre in der Lage die Intensitätswerte der Messpunkte für die automatische

Modellierung zu benützen. Für das Ausbessern von Fehlern in der Automatisierung könnte die manuelle Methode in Kapitel 6 noch immer verwendet werden. Allerdings sollten die Anzahl der Fehler und die Arbeitsbelastung im Gegensatz zum gänzlich manuellen Zeichnen deutlich geringer sein, ansonsten ist die automatische Linienziehung nicht rentabel und unnötig.

Literaturverzeichnis

[1] Kraus, K., 2004: *Photogrammetrie Band 1 – Geometrische Informationen aus Photographien und Laserscanneraufnahmen*, 7. Auflage, Walter-de-Gruyter-Verlag, S. 471

[2] Wagner, W., Ullrich A., Briese, C., 2003: *Der Laserstrahl und seine Interaktion mit der Erdoberfläche*, Osterreichische Zeitschrift für Vermessung and Geoinformation, VGI 4/2003, S. 3 – 6

[3] Kern, F., 2003: *Automatisierte Modellierung von Bauwerksgeometrien aus 3D-Laserscanner-Daten*, Dissertation. Technische Universität Braunschweig, S. 35 – 55

[4] Lindstaedt, M., Graeger, T., Mechelke, K., Kersten, T., 2010: *Terrestrische Laserscanner im Prüfstand – Geometrische Genauigkeitsuntersuchungen aktueller terrestrischer Laserscanner*, HafenCity Universität Hamburg, VDE Verlag GmbH, S. 1 – 2

[5] Pfeifer, N., Haring, A., Briese, C., 2007: *Automatische Auswertung im terrestrischen Laserscanning*, Talk: Seminar der DVW - Gesellschaft für Geodäsie, Geoinformation und Landmanagement, Fulda (invited); 2007-12-05 - 2007-12-06; in: "Terrestrisches Laserscanning (TLS 2007) - Ein Messverfahren erobert den Raum", Wißner-Verlag, 53 (2007), ISSN: 0940-4260, S. 2

[6] Buksch, A., 2011: *Revealing the skeleton from imperfect point clouds*, PhD Thesis at Delft University of Technology, Dr.Hut, Munich. ISBN 978-3-86853-877-9, S. 37 - 81

[7] Pfeifer, N., Gorte, B., Winterhalder, D., 2004: *Automatic reconstruction of single trees from terrestrial laser scanner data*, Proceedings of the XXth ISPRS Congress: Geo-Imagery Bridging Continents, 12–23 July, Istanbul, Turkey, pp. 114–119

[8] Schilling, A., Schmidt, A., Maas, H. G., 2012: *Tree Topology Representation from TLS Point Clouds Using Depth-First Search in Voxel Space*, Photogrammetric Engineering and Remote Sensing, Vol. 78, No. 4, pp. 383-392

[9] Bremer, M., Jochem, A., Rutzinger, M., 2012, *Comparison of branch extraction for deciduous single trees in leaf-on and leaf-off conditions – an eigenvector based approach for terrestrial laser scanning point clouds*, EARSeL eProceedings, Vol. 11, No. 1, 33-43,

[10] Brenner, C., 2008: *Interpretation terrestrischer Scandaten*, DVW-Seminar Terrestrisches. Terrestrisches-Laser-Scanning (TLS2007) – Ein Messverfahren erobert den Raum, Beiträge zum 74. DVW-Seminar am 5.-6.12.2007 in Fulda, Schriftenreihe Band 53, pp. 59-80

[11] Gorte, B, Pfeifer, N., 2004: *Structuring laser-scanned trees using 3d mathematical morphology*, Proceedings of the XXth ISPRS Congress: Geo-Imagery Bridging Continents, 12–23 July, Istanbul,Turkey, pp. 929–933

[12] Palágyi, K., Tschirren, Juerg., Sonka, M., 2003: Quantitative analysis of intrathoracic airway trees: methods and validation, LNCS 2732, Springer, 2003, 222-233

[13] *Z+F IMAGER® 5006h*, http://www.zf-laser.com/d_messprinzip.html, besucht am 11.5.2012

[14] *Z+F IMAGER® 5006h*, http://www.zf-laser.com/IMAGER_5006EX_brosch%C3%BCre_D.kompr.pdf, besucht am 11.5.2012

[15] Mandlburger, G., Otepka, J., Karel, W., Wöhrer, B., Wagner, W., Pfeifer, N., 2010: *OPALS (Orientation and Processing of Airborne Laser Scanning data) – Konzept und Anwendungsbeispiele einer wissenschaftlichen Laserscanning Software*, DGPF Tagungsband 19/2010 – Dreiländertagung OVG, DGPF und SGPF

Anhang

Hier sind alle wichtigen Codezeilen für Transformationen in Matlab und OPALS und für die VBA-Makros enthalten.

Matlab-Script “Transformation ins lokale Polarkoordinatensystem”

Das Matlab-Script wurde freundlicher Weise von Dipl.-Ing. Lothar Eysn zur Verfügung gestellt.

```
mainFolder = 'D:\ALS_projects\TLS_3dVeglab\Plattkarten\Laegeren\SOCS_int02\';
cd(mainFolder)
Files = dir('*.asc');
Filesmitext = {Files.name};
Filenamen = strrep(Filesmitext,'.xyz.asc','');

for k=1:length(Filenamen);
    test = Filenamen(:,k)
    fid = fopen([mainFolder Filenamen{k} '.xyz.asc'],'rt'); % 'rt' or 'r' does not matter, as format
convertion takes much longer
    fod = fopen([mainFolder Filenamen{k} '.plr'],'wt');
    inFormat = '%f%f%f%f'; % no digit-specification needed - just the type
    outFormat = '%12.6f %12.6f %8.4f %8.4f %15.0f\n';
    maxChunkRows = 1000000;
    cellArray{1} = 'dummy text';
    readRows = 0;
    tic
    while ~isempty(cellArray{1})
        cellArray = readTextChunkFormated(fid, inFormat, maxChunkRows);
        xyz = cell2mat(cellArray); % convert to matrix
        % do something with the data
        plr = cartesian2spherical(xyz(:,1:3));
        %geodetic azimuth
        plr(:,1) = 90-plr(:,1);
        k=find(plr(:,1)<0);
        plr(k,1) = plr(k,1) + 360;
        if ~isempty(xyz)
            fprintf(fod, outFormat, [plr(:,1) 180-plr(:,2) plr(:,3) sind(plr(:,2)).*plr(:,3) xyz(:,4)]');
            % Hz und V-Winkel mit negativem Vorzeichen verspeichert -->
            % Geodetic Azimuth von Y-Achse (Statt mathem. Azimuth von X-Achse), Nadir distance (statt
Poldistanz)
            % Format: Hz_winkel V-Winkel schrägDist. horizDist. Intensität
            % seitenrichtige Ansicht
        end
        readRows = readRows + size(xyz,1);
    end
    disp(['Processed ' num2str(readRows) ' rows.'])
    toc
    fclose(fod);
    fclose(fid);
end
```

Matlab-Script “Erstellung der Entfernungsdatei”

Das Matlab-Script wurde freundlicher Weise von Dipl.-Ing. Lothar Eysn zur Verfügung gestellt.

```
mainFolder = 'D:\PHOTO\11_3DVeglab\Plattkarten\Laegeren\SOCS_int02\';
cd(mainFolder)
Files = dir('*_R_ext.tif');
Filesmitext = {Files.name};
Filenamen = strrep(Filesmitext,'.tif','');

for k=1:length(Filenamen);
    T=imread([mainFolder Filenamen{k} '.tif']); %Read Rangeimage -> Nodatavalues are set to zero "0"!
    fod = fopen([mainFolder Filenamen{k} '.entf'],'wt'); %Open file for writing

    tic
    udT=flipud(T); %Matrix is fliped upside down -> Values in the outfile should start from the lower left
corner of the image
    udTtrans=udT'; % The Values of the Matrix are grabbed line-wise (horizontal pixels in image) -> each
value is written in a new single line in the outfile...therefore transposing helps to build this
arrangement
    Tvect=udTtrans(:); %Building a Vector -> One Range value in each line
    outFormat = '%i\n'; % outformat will be left aligned integers!
    if ~isempty(Tvect)
```

```
        fprintf(fod, outFormat, [round((Tvect(:,1)*1000)-32000)]');
        % writing values to the outfile: a linear transfomation of the
        % values is applied!! formula is "value*1000-32000" -> results must
        % be integers!! Nodata values become -32000...
    end
    toc

    fclose(fod);
end
```

OPALS-Script “Erstellung der Intensitäts- und Entfernungsbilder”

Das OPALS-Script wurde freundlicher Weise von Dipl.-Ing. Lothar Eysn zur Verfügung gestellt.

```
@echo off

REM ## Projektverzeichnis - Bitte ohne " angeben

set workingdir=D:\PHOTO\11_3DVeglab\Plattkarten\Laegeren\SOCS_int02\

if %workingdir:~0,+1%==^" (
        SET laufwerk=%workingdir:~1,+2%
        GOTO LAUFWCHECK
 ) ELSE (
        SET laufwerk=%workingdir:~0,+2%
        GOTO LAUFWCHECK
 )
:LAUFWCHECK
if %workingdir:~1,+1%==: (GOTO OK) ELSE (GOTO ERROR1)
:ERROR1
Echo .
Echo ERROR1:
Echo Der angegebene Pfad ist kein gueltiger, lokaler Pfad wie z.B.: c:\test.
Echo Netzwerkpfade wie etwa \\fs1\test sind nicht erlaubt. Workaround: Netzlaufwerk verbinden!
:OK

cd %workingdir%
%laufwerk%

REM setLocal EnableDelayedExpansion



cd %workingdir%
if exist list.txt del list.txt
ls -1 | grep .plr | sed "s/\.plr//g" > list.txt


for /f "tokens=* delims= " %%a in (list.txt) do (

:: Falls nötig ODM mit den aus Matlab generierten Polarkoordinatenfile erzeugen
 if not exist %workingdir%%%a.odm.dat opalsImport -inf %workingdir%%%a.plr -iformat
%workingdir%format_xyz_rhoriz_I.xml

 if not exist %workingdir%%%a_R_horiz.tif (


 REM Berechnung der Entfernungskarte mit HORIZONTALDISTANZEN!! (Ausmaße der Karte vorgegeben)
  opalsGrid -inf %workingdir%%%a.odm -grid 0.018 -interpolation nearestNeighbour -outfile
%workingdir%%%a_R_horiz.tif -attribute _Horizdist -limit "center (0.0 22.05 360.0 180.0)"

 REM Visualisierung der horizdist. Entfernungskarte mit der Matlab Jet Palette (Bereich von 0 - 40+ Meter)
  opalszcolor -inf %workingdir%%%a_R_horiz.tif -scale 0,25 -pal C:\Users\le\OPALS\addons\pal\jetTLSPal.xml
-outf %workingdir%%%a_R_horiz_zco_jet.tif
  del %workingdir%%%a_R_horiz_zco_jetTLSLegend.svg

 REM horiz R Karte beschneiden
  opalsalgebra -inf %workingdir%%%a_R_horiz.tif -outf %workingdir%%%a_R_horiz_cut090.tif -formula "return
z[0]" -limit "center(0.0,22.05,90.0,180.0)" -gridsize 0.018 -resampling nearestNeighbour
  opalsalgebra -inf %workingdir%%%a_R_horiz.tif -outf %workingdir%%%a_R_horiz_cut0360.tif -formula "return
z[0]" -limit "center(0.0,22.05,360.0,180.0)" -gridsize 0.018 -resampling nearestNeighbour
  gdal_translate -a_ullr 359.991 180.009 450.009 22.041 %workingdir%%%a_R_horiz_cut090.tif
%workingdir%%%a_R_horiz_cut090trans.tif
  opalsalgebra -inf %workingdir%%%a_R_horiz_cut090trans.tif -inf %workingdir%%%a_R_horiz_cut0360.tif -outf
%workingdir%%%a_R_horiz_ext.tif -formula "return z[0] if (z[1] == None) else z[1]" -limit
"center(0.0,22.05,450.0,180.0)" -noData 0 -gridsize 0.018

  del %workingdir%%%a_R_horiz_cut090.tif
  del %workingdir%%%a_R_horiz_cut090.tfw
  del %workingdir%%%a_R_horiz_cut090.tif.aux.xml
  del %workingdir%%%a_R_horiz_cut0360.tif
  del %workingdir%%%a_R_horiz_cut0360.tfw
  del %workingdir%%%a_R_horiz_cut0360.tif.aux.xml
  del %workingdir%%%a_R_horiz_cut090trans.tif
  REM del %workingdir%%%a_R_horiz_cut090trans.tfw
```

```
  REM del %workingdir%%%a_R_horiz_cut090trans.tif.aux.xml
  del %workingdir%%%a_R_horiz_ext.tfw
  del %workingdir%%%a_R_horiz_ext.tif.aux.xml

 REM Visualisierung der Entfernungskarte mit der Matlab Jet Palette (Bereich von 0 - 40+ Meter)
  opalszcolor -inf %workingdir%%%a_R_horiz_ext.tif -scale 0,25 -pal
C:\Users\le\OPALS\addons\pal\jetTLSPal.xml -outf %workingdir%%%a_R_horiz_ext_zco_jet.tif -pixelsize 0.018
  del %workingdir%%%a_R_horiz_ext_zco_jetTLSLegend.svg



REM Berechnung der Entfernungskarte (Ausmaße der Karte vorgegeben)
 opalsGrid -inf %workingdir%%%a.odm -grid 0.018 -interpolation nearestNeighbour -outfile
%workingdir%%%a_R.tif -limit "center (0.0 22.05 360.0 180.0)"

REM Visualisierung der Entfernungskarte mit der Matlab Jet Palette (Bereich von 0 - 40+ Meter)
 opalszcolor -inf %workingdir%%%a_R.tif -scale 0,25 -pal C:\Users\le\OPALS\addons\pal\jetTLSPal.xml  -outf
%workingdir%%%a_R_zco_jet.tif
 del %workingdir%%%a_R_zco_jetTLSLegend.svg

REM R Karte beschneiden
 opalsalgebra -inf %workingdir%%%a_R.tif -outf %workingdir%%%a_R_cut090.tif -formula "return z[0]" -limit
"center(0.0,22.05,90.0,180.0)" -gridsize 0.018 -resampling nearestNeighbour
 opalsalgebra -inf %workingdir%%%a_R.tif -outf %workingdir%%%a_R_cut0360.tif -formula "return z[0]" -limit
"center(0.0,22.05,360.0,180.0)" -gridsize 0.018 -resampling nearestNeighbour
 gdal_translate -a_ullr 359.991 180.009 450.009 22.041 %workingdir%%%a_R_cut090.tif
%workingdir%%%a_R_cut090trans.tif
 opalsalgebra -inf %workingdir%%%a_R_cut090trans.tif -inf %workingdir%%%a_R_cut0360.tif -outf
%workingdir%%%a_R_ext.tif -formula "return z[0] if (z[1] == None) else z[1]" -limit
"center(0.0,22.05,450.0,180.0)" -noData 0 -gridsize 0.018

 del %workingdir%%%a_R_cut090.tif
 del %workingdir%%%a_R_cut090.tfw
 del %workingdir%%%a_R_cut090.tif.aux.xml
 del %workingdir%%%a_R_cut0360.tif
 del %workingdir%%%a_R_cut0360.tfw
 del %workingdir%%%a_R_cut0360.tif.aux.xml
 del %workingdir%%%a_R_cut090trans.tif
 REM del %workingdir%%%a_R_cut090trans.tfw
 REM del %workingdir%%%a_R_cut090trans.tif.aux.xml
 del %workingdir%%%a_R_ext.tfw
 del %workingdir%%%a_R_ext.tif.aux.xml

REM Visualisierung der Entfernungskarte mit der Matlab Jet Palette (Bereich von 0 - 40+ Meter)
 opalszcolor -inf %workingdir%%%a_R_ext.tif -scale 0,25 -pal C:\Users\le\OPALS\addons\pal\jetTLSPal.xml -
outf %workingdir%%%a_R_ext_zco_jet.tif -pixelsize 0.018
 del %workingdir%%%a_R_ext_zco_jetTLSLegend.svg


REM Berechnung der Intensitätskarte (Ausmaße der Karte vorgegeben)
 opalsGrid -inf %workingdir%%%a.odm -grid 0.018 -interpolation nearestNeighbour -outfile
%workingdir%%%a_I.tif -attribute _Intens -limit "center (0.0 22.05 360.0 180.0)"

REM Visualisierung der Intensitätskarte mit der Grauwert Palette (Bereich von 20000 - 600000)
 opalszcolor -inf %workingdir%%%a_I.tif -scale 20000,600000 -pal C:\Users\le\OPALS\addons\pal\greyPal.xml -
outf %workingdir%%%a_I_zco_grey.tif
 del %workingdir%%%a_I_zco_greyLegend.svg


REM I Karte beschneiden
 opalsalgebra -inf %workingdir%%%a_I.tif -outf %workingdir%%%a_I_cut090.tif -formula "return z[0]" -limit
"center(0.0,22.05,90.0,180.0)" -gridsize 0.018 -resampling nearestNeighbour
 opalsalgebra -inf %workingdir%%%a_I.tif -outf %workingdir%%%a_I_cut0360.tif -formula "return z[0]" -limit
"center(0.0,22.05,360.0,180.0)" -gridsize 0.018 -resampling nearestNeighbour
 gdal_translate -a_ullr 359.991 180.009 450.009 22.041 %workingdir%%%a_I_cut090.tif
%workingdir%%%a_I_cut090trans.tif
 opalsalgebra -inf %workingdir%%%a_I_cut090trans.tif -inf %workingdir%%%a_I_cut0360.tif -outf
%workingdir%%%a_I_ext.tif -formula "return z[0] if (z[1] == None) else z[1]" -limit
"center(0.0,22.05,450.0,180.0)" -noData 0 -gridsize 0.018

 del %workingdir%%%a_I_cut090.tif
 del %workingdir%%%a_I_cut090.tfw
 del %workingdir%%%a_I_cut090.tif.aux.xml
 del %workingdir%%%a_I_cut0360.tif
 del %workingdir%%%a_I_cut0360.tfw
 del %workingdir%%%a_I_cut0360.tif.aux.xml
 del %workingdir%%%a_I_cut090trans.tif
 REM del %workingdir%%%a_I_cut090trans.tfw
 REM del %workingdir%%%a_I_cut090trans.tif.aux.xml
 del %workingdir%%%a_I_ext.tfw
 del %workingdir%%%a_I_ext.tif.aux.xml


REM Visualisierung der Intensitätskarte mit der Grauwert Palette (Bereich von 20000 - 600000)
 opalszcolor -inf %workingdir%%%a_I_ext.tif -scale 20000,600000 -pal
C:\Users\le\OPALS\addons\pal\greyPal.xml -outf %workingdir%%%a_I_ext_zco_grey.tif -pixelsize 0.018
 del %workingdir%%%a_I_ext_zco_greyLegend.svg
)
)
```

Makro „Standpunkt“

Formblattcode

```
Option Explicit

Private Sub CheckBox1_Click()
    If CheckBox1.Value = True Then
       Label9.Enabled = True
    End If
    If CheckBox1.Value = False Then
       Label9.Enabled = False
    End If
End Sub

Private Sub CheckBox2_Change()
    If CheckBox2.Value = True Then
       Label1.Enabled = True
       TextBox1.Enabled = True
       CommandButton1.Enabled = True
       CheckBox1.Enabled = True
       'Label9.Enabled = True
    End If
    If CheckBox2.Value = False Then
       Label1.Enabled = False
       TextBox1.Enabled = False
       CommandButton1.Enabled = False
       CheckBox1.Enabled = False
       'Label9.Enabled = False
    End If
End Sub

Private Sub CheckBox3_Change()
    If CheckBox3.Value = True Then
       Label2.Enabled = True
       TextBox2.Enabled = True
       CommandButton2.Enabled = True
       Label4.Enabled = True
       TextBox4.Enabled = True
       CommandButton4.Enabled = True
       Label13.Enabled = True
       TextBox11.Enabled = True
       CommandButton12.Enabled = True
    End If
    If CheckBox3.Value = False Then
       Label2.Enabled = False
       TextBox2.Enabled = False
       CommandButton2.Enabled = False
       Label4.Enabled = False
       TextBox4.Enabled = False
       CommandButton4.Enabled = False
       Label13.Enabled = False
       TextBox11.Enabled = False
       CommandButton12.Enabled = False
    End If
End Sub

Private Sub CheckBox8_Change()
    If CheckBox8.Value = True Then
       Label8.Enabled = True
       Label11.Enabled = True
       Label12.Enabled = True
       TextBox10.Enabled = True
       CommandButton10.Enabled = True
       CommandButton9.Enabled = True
       CommandButton11.Enabled = True
       ListBox1.Enabled = True
    End If
    If CheckBox8.Value = False Then
       Label8.Enabled = False
       Label11.Enabled = False
       Label12.Enabled = False
       TextBox10.Enabled = False
       CommandButton10.Enabled = False
       CommandButton9.Enabled = False
       CommandButton11.Enabled = False
       ListBox1.Enabled = False
    End If
End Sub

Private Sub CommandButton1_Click()
    Dim Filter As String, FileName As String
    Dim Flags As Long

    Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY Or _
```

```
            OFN_PATHMUSTEXIST

    Filter = "Entfernungsdaten (*.entf)" & Chr$(0) & "*.entf*" & Chr$(0) & _
              "Alle Dateien (*.*)" & Chr$(0) & _
              "*.*" & Chr$(0) & Chr$(0)

    FileName = ShowOpen(Filter, Flags)

    TextBox1.Value = FileName
End Sub

Private Sub CommandButton10_Click()
    Dim Filter As String, FileName As String
    Dim Flags As Long

    Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY Or _
            OFN_PATHMUSTEXIST

    Filter = "Transformationsdatei(*.txt)" & Chr$(0) & "*.txt*" & Chr$(0) & _
              "Alle Dateien (*.*)" & Chr$(0) & _
              "*.*" & Chr$(0) & Chr$(0)

    FileName = ShowOpen(Filter, Flags)

    TextBox10.Value = FileName
End Sub

Private Sub CommandButton11_Click()
    If ListBox1.ListCount > 0 And ListBox1.Value = ListBox1 Then
        ListBox1.RemoveItem (ListBox1.ListIndex)
    End If
End Sub

Private Sub CommandButton12_Click()
    Dim Filter As String, FileName As String
    Dim Flags As Long

    Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY Or _
            OFN_PATHMUSTEXIST

    Filter = "Entfernungsbild (*.tif)" & Chr$(0) & "*.tif*" & Chr$(0) & _
              "Alle Dateien (*.*)" & Chr$(0) & _
              "*.*" & Chr$(0) & Chr$(0)

    FileName = ShowOpen(Filter, Flags)

    TextBox11.Value = FileName

End Sub

Private Sub CommandButton2_Click()
    Dim Filter As String, FileName As String
    Dim Flags As Long

    Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY Or _
            OFN_PATHMUSTEXIST

    Filter = "Intensitätsbild (*.tif)" & Chr$(0) & "*.tif*" & Chr$(0) & _
              "Alle Dateien (*.*)" & Chr$(0) & _
              "*.*" & Chr$(0) & Chr$(0)

    FileName = ShowOpen(Filter, Flags)

    TextBox2.Value = FileName
End Sub

Private Sub CommandButton4_Click()
    Dim Filter As String, FileName As String
    Dim Flags As Long

    Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY Or _
            OFN_PATHMUSTEXIST

    Filter = "Überlagerungsdatei (*.txt)" & Chr$(0) & "*.txt*" & Chr$(0) & _
              "Alle Dateien (*.*)" & Chr$(0) & _
              "*.*" & Chr$(0) & Chr$(0)

    FileName = ShowOpen(Filter, Flags)

    TextBox4.Value = FileName
End Sub

Private Sub CommandButton5_Click()

    Call bildinfoschnell
    Standpunkt.Hide

End Sub
```

```
Private Sub CommandButton6_Click()

    Standpunkt.Hide

End Sub


Private Sub CommandButton9_Click()
    Dim Filter As String, FileName As String
    Dim Flags As Long

    Flags = OFN_FILEMUSTEXIST Or OFN_HIDEREADONLY Or _
            OFN_PATHMUSTEXIST

    Filter = "Baumstruktur (*.dwg)" & Chr$(0) & "*.dwg*" & Chr$(0) & _
              "Alle Dateien (*.*)" & Chr$(0) & _
              "*.*" & Chr$(0) & Chr$(0)

    FileName = ShowOpen(Filter, Flags)

    ListBox1.AddItem (FileName)
End Sub

Private Sub UserForm_Click()

End Sub
```

Ladecode

```
Sub bildinfoschnell()

'On Error GoTo fehler

Dim inspunkta(0 To 2) As Double, skala As Double, richta As Double, hoehea As Double, weitea As Double
Dim Wlddatei As String, Wlddateinr As Integer
Dim layername As String
Dim objlayer As AcadLayer

'ThisDrawing.Application.ActiveDocument = ThisDrawing

Set zeichenebene = ThisDrawing.Application.ActiveDocument

Dim a As String
Dim q, o As Integer

a = Standpunkt.TextBox1.Value

For q = 1 To Len(a)
   If Mid(a, q, 1) = "\" Then
        o = q
    End If
    If Mid(a, q, 1) = "." Then
        layername = Mid(a, o + 1, q - 1 - o)
    End If
Next q
'MsgBox dat

If Standpunkt.CheckBox3.Value = True Then

    'Intensitätsbild und Entfernungsbild

    Set objlayer = zeichenebene.Layers.Add("Intensitaetsbild")
    zeichenebene.ActiveLayer = zeichenebene.Layers("Intensitaetsbild")
    'objlayer.color = acRed

    Wlddatei = Standpunkt.TextBox4.Value
    Wlddateinr = FreeFile()

    Open Wlddatei For Input As Wlddateinr

    Line Input #Wlddateinr, p
    inspunkta(0) = Val(p)

    Line Input #Wlddateinr, p
    inspunkta(1) = Val(p)

    Line Input #Wlddateinr, p
    inspunkta(2) = Val(p)

    Line Input #Wlddateinr, p
    skala = Val(p)

    Line Input #Wlddateinr, p
    richta = Val(p)

    Line Input #Wlddateinr, p
    hoehea = Val(p)
```

```
    Line Input #Wlddateinr, p
    weitea = Val(p)

    Close Wlddateinr

    Set objrasterimgi = ActiveDocument.ModelSpace.AddRaster(Standpunkt.TextBox2.Value, inspunkta, 1,
richta)
        objrasterimgi.ImageHeight = hoehea
        objrasterimgi.ImageWidth = weitea
        'objrasterimgi.ScaleFactor = skala
        objrasterimgi.Update

    Set objlayer = zeichenebene.Layers.Add("Entfernungsbild")
    zeichenebene.ActiveLayer = zeichenebene.Layers("Entfernungsbild")
    'objlayer.color = acRed

    Set objrasterimge = ActiveDocument.ModelSpace.AddRaster(Standpunkt.TextBox11.Value, inspunkta, 1,
richta)
        objrasterimge.ImageHeight = hoehea
        objrasterimge.ImageWidth = weitea
        'objrasterimge.ScaleFactor = skala
        objrasterimge.Update

    zeichenebene.Layers.Add("Intensitaetsbild").Lock = True
    zeichenebene.Layers.Add("Entfernungsbild").Lock = True
    zeichenebene.Layers.Add("Entfernungsbild").LayerOn = False
End If

'Punkteimport

Dim punkt(0 To 2) As Double
Dim punktdatei As String, punktdateinr As Integer
Dim s As String, r As String
Dim z As Long
Dim n As Double

Dim i As Long

If Standpunkt.CheckBox2.Value = True Then

    Set objlayer = zeichenebene.Layers.Add(layername)
    zeichenebene.ActiveLayer = zeichenebene.Layers(layername)


    punktdatei = Standpunkt.TextBox1.Value
    punktdateinr = FreeFile()

    Open punktdatei For Input As punktdateinr

    z = 1

    'Entfernungsinformation in Matrix schreiben
    For j = 1 To 8776
        For g = 1 To 25001

            'If Not EOF(punktdateinr) Then
                Line Input #punktdateinr, s

                entfdaten(g, j) = Val(Mid(s, 1, 6))

                If Standpunkt.CheckBox1.Value = True Then
                    If (z > 225000000 And z < 225300000) Or (z > 110700000 And z < 111000000) Then
                        punkt(0) = g * 0.018 + 0
                        punkt(1) = j * 0.018 - 67.95

                        'If entfdaten(g, j) = 0 Then
                        '    n = CDbl(entfdaten(g, j))
                        'Else
                            n = (CDbl(entfdaten(g, j)) + 32000) / 1000
                        'End If

                        punkt(2) = n

                        If punkt(2) > 0.2 Then

                            zeichenebene.ModelSpace.AddPoint (punkt)

                        End If
                    End If
                End If
            'End If
            z = z + 1
        Next g
    Next j

    Close punktdateinr

    Set objlayer = zeichenebene.Layers.Add("Baumstruktur")
    objlayer.color = acRed
```

```
    zeichenebene.ActiveLayer = zeichenebene.Layers("Baumstruktur")
End If

'Set objlayer = zeichenebene.Layers.Add("Baumstruktur")
'objlayer.color = acRed
'zeichenebene.ActiveLayer = zeichenebene.Layers("Baumstruktur")

ZoomExtents

'zusätzliche Baumstruktur

Dim dwgName As String

Dim oAcadLine As AcadLine
Dim oEntity As AcadEntity
Dim Point As Variant
Dim objzeich As AcadObject
Dim u As Integer
Dim lin As AcadLine

If Standpunkt.CheckBox8.Value = True Then

    For u = 0 To Standpunkt.ListBox1.ListCount - 1

        dwgName = Standpunkt.ListBox1.List(u)

        If Dir(dwgName) <> "" Then
                Set kopierebene = ThisDrawing.Application.Documents.Open(dwgName)
        Else
                MsgBox "Die Datei " & dwgName & " existiert nicht."
        End If

        'Set strukturenebene = ThisDrawing.Application.Documents.Add

        Dim datname As String
        Dim v, w As Integer

            For v = 1 To Len(dwgName)
                If Mid(dwgName, v, 1) = "\" Then
                    w = v
                End If
                If Mid(dwgName, v, 1) = "." Then
                    datname = Mid(dwgName, w + 1, v - 1 - w)
                End If
            Next v

        'Set objlayer = strukturenebene.Layers.Add(datname)

        'strukturenebene.ActiveLayer = strukturenebene.Layers(datname)

        'For Each objzeich In kopierebene.ModelSpace

        '    If objzeich.ObjectName = "AcDbLine" Then

        '        Set oAcadLine = objzeich
        '        With oAcadLine

        '        Set lin = strukturenebene.ModelSpace.AddLine(.StartPoint, .EndPoint)

        '        lin.Thickness = oAcadLine.Thickness
        '        End With

        '    End If
        'Next

        ZoomExtents

        'transformierte zusätzliche Baumstruktur

        Dim inspunkt(0 To 2) As Double, kpunkt(0 To 2) As Double, epunkt(0 To 2) As Double
        Dim apunkt As Variant, bpunkt As Variant
        Dim rotation(0 To 2, 0 To 2) As Double, translat(0 To 2) As Double
        Dim transdatei As String, transdateinr As Integer

        transdatei = Standpunkt.TextBox10.Value
        transdateinr = FreeFile()

        Open transdatei For Input As transdateinr

        Line Input #transdateinr, r
        rotation(0, 0) = Val(Mid(r, 1, 13))
        rotation(0, 1) = Val(Mid(r, 15, 13))
        rotation(0, 2) = Val(Mid(r, 29, 13))
        translat(0) = Val(Mid(r, 43, 16))

        Line Input #transdateinr, r
        rotation(1, 0) = Val(Mid(r, 1, 13))
        rotation(1, 1) = Val(Mid(r, 15, 13))
        rotation(1, 2) = Val(Mid(r, 29, 13))
```

```
        translat(1) = Val(Mid(r, 43, 16))

        Line Input #transdateinr, r
        rotation(2, 0) = Val(Mid(r, 1, 13))
        rotation(2, 1) = Val(Mid(r, 15, 13))
        rotation(2, 2) = Val(Mid(r, 29, 13))
        translat(2) = Val(Mid(r, 43, 16))

        Close transdateinr

        Set objlayer = zeichenebene.Layers.Add(datname)

        zeichenebene.Layers(datname).color = acBlue

        zeichenebene.ActiveLayer = zeichenebene.Layers(datname)

            For Each objzeich In kopierebene.ModelSpace

                If objzeich.ObjectName = "AcDbLine" Then

                    Set oAcadLine = objzeich

                    'apunkt

                    inspunkt(0) = oAcadLine.StartPoint(0)
                    inspunkt(1) = oAcadLine.StartPoint(1)
                    inspunkt(2) = oAcadLine.StartPoint(2)

                    kpunkt(0) = inspunkt(0) - translat(0)
                    kpunkt(1) = inspunkt(1) - translat(1)
                    kpunkt(2) = inspunkt(2) - translat(2)

                    epunkt(0) = rotation(0, 0) * kpunkt(0) + rotation(1, 0) * kpunkt(1) + rotation(2, 0) *
kpunkt(2)
                    epunkt(1) = rotation(0, 1) * kpunkt(0) + rotation(1, 1) * kpunkt(1) + rotation(2, 1) *
kpunkt(2)
                    epunkt(2) = rotation(0, 2) * kpunkt(0) + rotation(1, 2) * kpunkt(1) + rotation(2, 2) *
kpunkt(2)

                    punkt(2) = (epunkt(0) ^ 2 + epunkt(1) ^ 2 + epunkt(2) ^ 2) ^ (1 / 2)

                    If punkt(2) > 0.5 Then

                        On Error Resume Next

                        punkt(1) = Sgn((epunkt(2) / punkt(2))) * Atn(((epunkt(2) / punkt(2)) ^ 2 / (1 -
(epunkt(2) / punkt(2)) ^ 2)) ^ (1 / 2))

                        If epunkt(0) > 0 And epunkt(1) > 0 Then

                           punkt(0) = 90 - 360 + Abs(180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2))))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 /
Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2)) - 360)

                        End If

                        If epunkt(0) <= 0 And epunkt(1) >= 0 Then

                           punkt(0) = 90 + 180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) * ((epunkt(1) /
punkt(2))))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2)) + 180

                        End If

                        If epunkt(0) < 0 And epunkt(1) < 0 Then

                           punkt(0) = 90 + 180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) * ((epunkt(1) /
punkt(2))))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2)) + 180

                        End If

                        If epunkt(0) >= 0 And epunkt(1) <= 0 Then

                           punkt(0) = 90 + Abs(180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) * ((epunkt(1) /
punkt(2))))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2)))

                        End If

                        punkt(1) = 180 / (4 * Atn(1)) * punkt(1)

                        'If punkt(2) > 60 Then

                        'punkt(2) = 0

                        'End If

                        apunkt = CVar(punkt)
```

```
                    End If

                    'bpunkt

                    inspunkt(0) = oAcadLine.EndPoint(0)
                    inspunkt(1) = oAcadLine.EndPoint(1)
                    inspunkt(2) = oAcadLine.EndPoint(2)

                    kpunkt(0) = inspunkt(0) - translat(0)
                    kpunkt(1) = inspunkt(1) - translat(1)
                    kpunkt(2) = inspunkt(2) - translat(2)

                    epunkt(0) = rotation(0, 0) * kpunkt(0) + rotation(1, 0) * kpunkt(1) + rotation(2, 0) *
kpunkt(2)
                    epunkt(1) = rotation(0, 1) * kpunkt(0) + rotation(1, 1) * kpunkt(1) + rotation(2, 1) *
kpunkt(2)
                    epunkt(2) = rotation(0, 2) * kpunkt(0) + rotation(1, 2) * kpunkt(1) + rotation(2, 2) *
kpunkt(2)

                    punkt(2) = (epunkt(0) ^ 2 + epunkt(1) ^ 2 + epunkt(2) ^ 2) ^ (1 / 2)

                    If punkt(2) > 0.5 Then

                        On Error Resume Next

                        punkt(1) = Sgn((epunkt(2) / punkt(2))) * Atn(((epunkt(2) / punkt(2)) ^ 2 / (1 -
(epunkt(2) / punkt(2)) ^ 2)) ^ (1 / 2))

                        If epunkt(0) > 0 And epunkt(1) > 0 Then

                           punkt(0) = 90 - 180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) * ((epunkt(1) /
punkt(2)))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2))

                        End If

                        If epunkt(0) <= 0 And epunkt(1) >= 0 Then

                           punkt(0) = 270 + 180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) * ((epunkt(1) /
punkt(2)))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2))

                        End If

                        If epunkt(0) < 0 And epunkt(1) < 0 Then

                           punkt(0) = 270 + 180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) * ((epunkt(1) /
punkt(2)))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2))

                        End If

                        If epunkt(0) >= 0 And epunkt(1) <= 0 Then

                           punkt(0) = 90 - 180 / (4 * Atn(1)) * Sgn((1 / Cos(punkt(1)) * ((epunkt(1) /
punkt(2)))) * Atn(((1 / Cos(punkt(1)) * ((epunkt(1) / punkt(2)))) ^ 2 / (1 - (1 / Cos(punkt(1)) *
((epunkt(1) / punkt(2)))) ^ 2)) ^ (1 / 2))

                        End If

                        punkt(1) = 180 / (4 * Atn(1)) * punkt(1)

                        'If punkt(2) > 60 Then

                        'punkt(2) = 0

                        'End If

                        bpunkt = CVar(punkt)

                    End If

                    If (bpunkt(0) - apunkt(0)) > 330 Then
                       apunkt(0) = apunkt(0) + 360
                    End If
                    If (apunkt(0) - bpunkt(0)) > 330 Then
                       bpunkt(0) = bpunkt(0) + 360
                    End If


                    zeichenebene.ActiveLayer = zeichenebene.Layers(datname)

                    Set lin = zeichenebene.ModelSpace.AddLine(apunkt, bpunkt)

                    lin.LinetypeScale = oAcadLine.LinetypeScale
                    lin.Lineweight = acLnWt040

                End If
```

```
            Next
        zeichenebene.Layers(datname).color = acBlue
        zeichenebene.Layers(datname).Lock = True

    Next u

    kopierebene.Close

End If

zeichenebene.Activate

zeichenebene.ActiveLayer = zeichenebene.Layers("Baumstruktur")
zeichenebene.Layers("Baumstruktur").Lineweight = acLnWt040

'objrasterimgi.select

'ThisDrawing.SendCommand "_.draworder" & vbCr & "_p" & vbCr & vbCr & "_back" & vbCr

'objrsterimge.Select

'ThisDrawing.SendCommand "_.draworder" & vbCr & "_p" & vbCr & vbCr & "_back" & vbCr


'Exit Sub

'fehler:

'MsgBox "Fehler! Abbruch!"

End Sub
```

Makro „Punktdarstellung“

```
Sub punktdarstellung()

Dim varpunkt As Variant
Dim n As Double, i As Integer, j As Integer, dx As Integer, dy As Integer
Dim punkt(0 To 2) As Double

varpunkt = ThisDrawing.Utility.GetPoint(, "Punktdarstellung")

Set objlayer = ThisDrawing.Layers.Add("Testpunkte")
objlayer.color = acGreen
ThisDrawing.ActiveLayer = ThisDrawing.Layers("Testpunkte")

dx = Round((varpunkt(0) - 0) / 0.018)
dy = Round((varpunkt(1) + 67.95) / 0.018)

For i = -20 To 20
    For j = -20 To 20

        punkt(0) = (dx + i) * 0.018 + 0
        punkt(1) = (dy + j) * 0.018 - 67.95

            n = (CDbl(entfdaten(dx + i, dy + j)) + 32000) / 1000

        punkt(2) = n

        If punkt(2) > 0.2 Then

            ThisDrawing.ModelSpace.AddPoint (punkt)

        End If

    Next j
Next i

ThisDrawing.ActiveLayer = ThisDrawing.Layers("Baumstruktur")


End Sub
```

Makro „Entfernungsbild“

```
Sub entfer()
    If ThisDrawing.Layers("entfernungsbild").LayerOn = False Then
       ThisDrawing.Layers("entfernungsbild").LayerOn = True
    Else
        ThisDrawing.Layers("entfernungsbild").LayerOn = False
    End If


End Sub
```

Makro „Linienzeichnen“

```
Sub liniezeichnen()

On Error GoTo fehlerroutine

Dim objlayer As AcadLayer
Dim svarpunkt, evarpunkt, varpunkt, l1varpunkt, r1varpunkt, l2varpunkt, r2varpunkt As Variant
Dim nl1varpunkt(0 To 2) As Double, nr1varpunkt(0 To 2) As Double, nl2varpunkt(0 To 2) As Double,
nr2varpunkt(0 To 2) As Double
Dim sfang, efang As Boolean
Dim p As Double, c As Double, zylzeichdm As Double
Dim equadrat(1 To 9) As Double
Dim i, j, k, h, e, m As Integer
Dim tmp, median1, median2 As Double
Dim aktlinie As AcadLine
Dim objZylinder As Acad3DSolid
Dim lpud(2) As Double

Set objZylinder = ThisDrawing.ModelSpace.AddCylinder(lpud, 5, 7)

Do

'Startpunkt
varpunkt = ThisDrawing.Utility.GetPoint(, "Startpunkt:")

If varpunkt(2) > 0.01 Then

svarpunkt = varpunkt

sfang = True

Else

    k = 1

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) - 1, Round((varpunkt(1) + 67.95) / 0.018) - 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) - 1, Round((varpunkt(1) + 67.95) / 0.018))
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) - 1, Round((varpunkt(1) + 67.95) / 0.018) + 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018), Round((varpunkt(1) + 67.95) / 0.018) - 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018), Round((varpunkt(1) + 67.95) / 0.018))
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018), Round((varpunkt(1) + 67.95) / 0.018) + 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) + 1, Round((varpunkt(1) + 67.95) / 0.018) - 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
```

```
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) + 1, Round((varpunkt(1) + 67.95) / 0.018))
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) + 1, Round((varpunkt(1) + 67.95) / 0.018) + 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    k = k - 1


    If k = 0 Then

    median1 = 0
    Else

    'Sortierung
        For i = 2 To k

            tmp = equadrat(i)
            j = i - 1

            Do While (j >= 1) And (tmp < equadrat(j))

                equadrat(j + 1) = equadrat(j)
                j = j - 1
                If j < 1 Then
                   Exit Do
                End If
            Loop

            equadrat(j + 1) = tmp

        Next i

    'Medianberechnung

        If (k Mod 2) = 0 Then
           median1 = (equadrat(k \ 2) + equadrat((k \ 2) + 1)) / 2
        Else
           median1 = equadrat((k \ 2) + 1)
        End If

    End If

    ThisDrawing.SetVariable "osmode", 0
    l1varpunkt = ThisDrawing.Utility.GetPoint(, "Linkspunktabgriff")
    r1varpunkt = ThisDrawing.Utility.GetPoint(, "Rechtspunktabgriff")
    ThisDrawing.SetVariable "osmode", 1

    p = ((l1varpunkt(0) - r1varpunkt(0)) ^ 2 + (l1varpunkt(1) - r1varpunkt(1)) ^ 2) ^ (1 / 2)

    If p < 0.0001 Then

    MsgBox "Distanz zu klein!"

    Else

    sfang = False

    svarpunkt = varpunkt
    svarpunkt(2) = median1

    End If

End If

'Endpunkt
varpunkt = ThisDrawing.Utility.GetPoint(, "Endpunkt:")

If varpunkt(2) > 0.01 Then

    evarpunkt = varpunkt

    ThisDrawing.SetVariable "osmode", 0
    l2varpunkt = ThisDrawing.Utility.GetPoint(, "Linkspunktabgriff")
    r2varpunkt = ThisDrawing.Utility.GetPoint(, "Rechtspunktabgriff")
```

```
    ThisDrawing.SetVariable "osmode", 1

    p = ((l2varpunkt(0) - r2varpunkt(0)) ^ 2 + (l2varpunkt(1) - r2varpunkt(1)) ^ 2) ^ (1 / 2)

    If p < 0.0001 Then

    MsgBox "Distanz zu klein!"

    Else

    efang = True

    End If

Else

    k = 1

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) - 1, Round((varpunkt(1) + 67.95) / 0.018) - 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) - 1, Round((varpunkt(1) + 67.95) / 0.018))
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) - 1, Round((varpunkt(1) + 67.95) / 0.018) + 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018), Round((varpunkt(1) + 67.95) / 0.018) - 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018), Round((varpunkt(1) + 67.95) / 0.018))
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018), Round((varpunkt(1) + 67.95) / 0.018) + 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) + 1, Round((varpunkt(1) + 67.95) / 0.018) - 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) + 1, Round((varpunkt(1) + 67.95) / 0.018))
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If

    e = entfdaten(Round((varpunkt(0) - 0) / 0.018) + 1, Round((varpunkt(1) + 67.95) / 0.018) + 1)
    If e = -32000 Then
       equadrat(k) = e
    Else
       equadrat(k) = (CDbl(e) + 32000) / 1000
       k = k + 1
    End If
```

```
    k = k - 1

    If k = 0 Then

    median2 = 0

    Else

    'Sortierung

        For i = 2 To k

            tmp = equadrat(i)
            j = i - 1

            Do While (j >= 1) And (tmp < equadrat(j))

                equadrat(j + 1) = equadrat(j)
                j = j - 1
                If j < 1 Then
                   Exit Do
                End If
            Loop

            equadrat(j + 1) = tmp

        Next i

    'Medianberechnung

        If (k Mod 2) = 0 Then
           median2 = (equadrat(k \ 2) + equadrat((k \ 2) + 1)) / 2
        Else
           median2 = equadrat((k \ 2) + 1)
        End If

    End If

    ThisDrawing.SetVariable "osmode", 0
    l2varpunkt = ThisDrawing.Utility.GetPoint(, "Linkspunktabgriff")
    r2varpunkt = ThisDrawing.Utility.GetPoint(, "Rechtspunktabgriff")
    ThisDrawing.SetVariable "osmode", 1

    p = ((l2varpunkt(0) - r2varpunkt(0)) ^ 2 + (l2varpunkt(1) - r2varpunkt(1)) ^ 2) ^ (1 / 2)

    If p < 0.0001 Then

    MsgBox "Distanz zu klein!"

    Else

    efang = False

    evarpunkt = varpunkt
    evarpunkt(2) = median2

    End If

End If


Dim l1n(0 To 2), r1n(0 To 2), l2n(0 To 2), r2n(0 To 2) As Double
Dim l1d, r1d, l2d, r2d, d1, d2 As Double
Dim nsvarpunkt(0 To 2) As Double, nevarpunkt(0 To 2) As Double, nsevarpunkt(0 To 2) As Double,
l1taunvektor(0 To 2) As Double, r1taunvektor(0 To 2) As Double
Dim l2taunvektor(0 To 2) As Double, r2taunvektor(0 To 2) As Double
'Normaleinheitsvektor von Vektor zwischen Start- und Endpunkt

nsvarpunkt(0) = svarpunkt(2) * Cos(svarpunkt(0) / 180 * 4 * Atn(1)) * Cos(svarpunkt(1) / 180 * 4 * Atn(1))
nsvarpunkt(1) = svarpunkt(2) * Sin(svarpunkt(0) / 180 * 4 * Atn(1)) * Cos(svarpunkt(1) / 180 * 4 * Atn(1))
nsvarpunkt(2) = svarpunkt(2) * Sin(svarpunkt(1) / 180 * 4 * Atn(1))

nevarpunkt(0) = evarpunkt(2) * Cos(evarpunkt(0) / 180 * 4 * Atn(1)) * Cos(evarpunkt(1) / 180 * 4 * Atn(1))
nevarpunkt(1) = evarpunkt(2) * Sin(evarpunkt(0) / 180 * 4 * Atn(1)) * Cos(evarpunkt(1) / 180 * 4 * Atn(1))
nevarpunkt(2) = evarpunkt(2) * Sin(evarpunkt(1) / 180 * 4 * Atn(1))

nsevarpunkt(0) = nevarpunkt(0) - nsvarpunkt(0)
nsevarpunkt(1) = nevarpunkt(1) - nsvarpunkt(1)
nsevarpunkt(2) = nevarpunkt(2) - nsvarpunkt(2)

If sfang = True Then

    'svarpunkt = svarpunkt
Else

    nl1varpunkt(0) = Cos(l1varpunkt(0) / 180 * 4 * Atn(1)) * Cos(l1varpunkt(1) / 180 * 4 * Atn(1))
    nl1varpunkt(1) = Sin(l1varpunkt(0) / 180 * 4 * Atn(1)) * Cos(l1varpunkt(1) / 180 * 4 * Atn(1))
    nl1varpunkt(2) = Sin(l1varpunkt(1) / 180 * 4 * Atn(1))
```

```
    nr1varpunkt(0) = Cos(r1varpunkt(0) / 180 * 4 * Atn(1)) * Cos(r1varpunkt(1) / 180 * 4 * Atn(1))
    nr1varpunkt(1) = Sin(r1varpunkt(0) / 180 * 4 * Atn(1)) * Cos(r1varpunkt(1) / 180 * 4 * Atn(1))
    nr1varpunkt(2) = Sin(r1varpunkt(1) / 180 * 4 * Atn(1))

    l1taunvektor(0) = nl1varpunkt(1) * nsevarpunkt(2) - nl1varpunkt(2) * nsevarpunkt(1)
    l1taunvektor(1) = nl1varpunkt(2) * nsevarpunkt(0) - nl1varpunkt(0) * nsevarpunkt(2)
    l1taunvektor(2) = nl1varpunkt(0) * nsevarpunkt(1) - nl1varpunkt(1) * nsevarpunkt(0)

    r1taunvektor(0) = nr1varpunkt(1) * nsevarpunkt(2) - nr1varpunkt(2) * nsevarpunkt(1)
    r1taunvektor(1) = nr1varpunkt(2) * nsevarpunkt(0) - nr1varpunkt(0) * nsevarpunkt(2)
    r1taunvektor(2) = nr1varpunkt(0) * nsevarpunkt(1) - nr1varpunkt(1) * nsevarpunkt(0)

    snwinkel = (l1taunvektor(0) * r1taunvektor(0) + l1taunvektor(1) * r1taunvektor(1) + l1taunvektor(2) *
r1taunvektor(2)) / (l1taunvektor(0) ^ 2 + l1taunvektor(1) ^ 2 + l1taunvektor(2) ^ 2) ^ (1 / 2) /
(r1taunvektor(0) ^ 2 + r1taunvektor(1) ^ 2 + r1taunvektor(2) ^ 2) ^ (1 / 2)

    sepsilon = Atn(1) * 2 - Sgn(snwinkel) * Atn((snwinkel ^ 2 / (1 - snwinkel ^ 2)) ^ (1 / 2))

    'sabwinkel = Abs(Atn(1) * 2 - (nsvarpunkt(0) * nsevarpunkt(0) + nsvarpunkt(1) * nsevarpunkt(1) +
nsvarpunkt(2) * nsevarpunkt(2)) / (nsvarpunkt(0) ^ 2 + nsvarpunkt(1) ^ 2 + nsvarpunkt(2) ^ 2) ^ (1 / 2) /
(nsevarpunkt(0) ^ 2 + nsevarpunkt(1) ^ 2 + nsevarpunkt(2) ^ 2) ^ (1 / 2))
    sabwinkel = (nsvarpunkt(0) * nsevarpunkt(0) + nsvarpunkt(1) * nsevarpunkt(1) + nsvarpunkt(2) *
nsevarpunkt(2)) / (nsvarpunkt(0) ^ 2 + nsvarpunkt(1) ^ 2 + nsvarpunkt(2) ^ 2) ^ (1 / 2) / (nsevarpunkt(0) ^
2 + nsevarpunkt(1) ^ 2 + nsevarpunkt(2) ^ 2) ^ (1 / 2)

    sabphi = Abs(Sgn(sabwinkel) * Atn((sabwinkel ^ 2 / (1 - sabwinkel ^ 2)) ^ (1 / 2)))

    median1r = median1 * Cos(sabphi)
    crn = median1r / (1 / Sin(sepsilon / 2) - 1)
    cr = crn / Cos(svarpunkt(1) * Atn(1) * 4 / 180)

    svarpunkt(2) = svarpunkt(2) + cr

    MsgBox cr

End If

If efang = True Then

    nl2varpunkt(0) = Cos(l2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(l2varpunkt(1) / 180 * 4 * Atn(1))
    nl2varpunkt(1) = Sin(l2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(l2varpunkt(1) / 180 * 4 * Atn(1))
    nl2varpunkt(2) = Sin(l2varpunkt(1) / 180 * 4 * Atn(1))

    nr2varpunkt(0) = Cos(r2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(r2varpunkt(1) / 180 * 4 * Atn(1))
    nr2varpunkt(1) = Sin(r2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(r2varpunkt(1) / 180 * 4 * Atn(1))
    nr2varpunkt(2) = Sin(r2varpunkt(1) / 180 * 4 * Atn(1))

    l2taunvektor(0) = nl2varpunkt(1) * nsevarpunkt(2) - nl2varpunkt(2) * nsevarpunkt(1)
    l2taunvektor(1) = nl2varpunkt(2) * nsevarpunkt(0) - nl2varpunkt(0) * nsevarpunkt(2)
    l2taunvektor(2) = nl2varpunkt(0) * nsevarpunkt(1) - nl2varpunkt(1) * nsevarpunkt(0)

    r2taunvektor(0) = nr2varpunkt(1) * nsevarpunkt(2) - nr2varpunkt(2) * nsevarpunkt(1)
    r2taunvektor(1) = nr2varpunkt(2) * nsevarpunkt(0) - nr2varpunkt(0) * nsevarpunkt(2)
    r2taunvektor(2) = nr2varpunkt(0) * nsevarpunkt(1) - nr2varpunkt(1) * nsevarpunkt(0)

    enwinkel = (l2taunvektor(0) * r2taunvektor(0) + l2taunvektor(1) * r2taunvektor(1) + l2taunvektor(2) *
r2taunvektor(2)) / (l2taunvektor(0) ^ 2 + l2taunvektor(1) ^ 2 + l2taunvektor(2) ^ 2) ^ (1 / 2) /
(r2taunvektor(0) ^ 2 + r2taunvektor(1) ^ 2 + r2taunvektor(2) ^ 2) ^ (1 / 2)

    eepsilon = Atn(1) * 2 - Sgn(enwinkel) * Atn((enwinkel ^ 2 / (1 - enwinkel ^ 2)) ^ (1 / 2))

    sbawinkel = (nevarpunkt(0) * nsevarpunkt(0) + nevarpunkt(1) * nsevarpunkt(1) + nevarpunkt(2) *
nsevarpunkt(2)) / (nevarpunkt(0) ^ 2 + nevarpunkt(1) ^ 2 + nevarpunkt(2) ^ 2) ^ (1 / 2) / (nsevarpunkt(0) ^
2 + nsevarpunkt(1) ^ 2 + nsevarpunkt(2) ^ 2) ^ (1 / 2)

    sbaphi = Abs(Sgn(sbawinkel) * Atn((sbawinkel ^ 2 / (1 - sbawinkel ^ 2)) ^ (1 / 2)))

    zylzeichdm = eepsilon / (Atn(1) * 4) * 180

    evar = evarpunkt(2) * Cos(sbaphi)

    crn = evar / (1 / Sin(eepsilon / 2) - 1)
    crn = (evar - crn) / (1 / Sin(eepsilon / 2) - 1)

    cr = crn / Cos(sbaphi)

    evarpunkt = evarpunkt

    MsgBox cr

Else

    nl2varpunkt(0) = Cos(l2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(l2varpunkt(1) / 180 * 4 * Atn(1))
    nl2varpunkt(1) = Sin(l2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(l2varpunkt(1) / 180 * 4 * Atn(1))
    nl2varpunkt(2) = Sin(l2varpunkt(1) / 180 * 4 * Atn(1))

    nr2varpunkt(0) = Cos(r2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(r2varpunkt(1) / 180 * 4 * Atn(1))
    nr2varpunkt(1) = Sin(r2varpunkt(0) / 180 * 4 * Atn(1)) * Cos(r2varpunkt(1) / 180 * 4 * Atn(1))
```

```
    nr2varpunkt(2) = Sin(r2varpunkt(1) / 180 * 4 * Atn(1))

    l2taunvektor(0) = nl2varpunkt(1) * nsevarpunkt(2) - nl2varpunkt(2) * nsevarpunkt(1)
    l2taunvektor(1) = nl2varpunkt(2) * nsevarpunkt(0) - nl2varpunkt(0) * nsevarpunkt(2)
    l2taunvektor(2) = nl2varpunkt(0) * nsevarpunkt(1) - nl2varpunkt(1) * nsevarpunkt(0)

    r2taunvektor(0) = nr2varpunkt(1) * nsevarpunkt(2) - nr2varpunkt(2) * nsevarpunkt(1)
    r2taunvektor(1) = nr2varpunkt(2) * nsevarpunkt(0) - nr2varpunkt(0) * nsevarpunkt(2)
    r2taunvektor(2) = nr2varpunkt(0) * nsevarpunkt(1) - nr2varpunkt(1) * nsevarpunkt(0)

    enwinkel = (l2taunvektor(0) * r2taunvektor(0) + l2taunvektor(1) * r2taunvektor(1) + l2taunvektor(2) *
r2taunvektor(2)) / (l2taunvektor(0) ^ 2 + l2taunvektor(1) ^ 2 + l2taunvektor(2) ^ 2) ^ (1 / 2) /
(r2taunvektor(0) ^ 2 + r2taunvektor(1) ^ 2 + r2taunvektor(2) ^ 2) ^ (1 / 2)

    eepsilon = Atn(1) * 2 - Sgn(enwinkel) * Atn((enwinkel ^ 2 / (1 - enwinkel ^ 2)) ^ (1 / 2))

    sbawinkel = (nevarpunkt(0) * nsevarpunkt(0) + nevarpunkt(1) * nsevarpunkt(1) + nevarpunkt(2) *
nsevarpunkt(2)) / (nevarpunkt(0) ^ 2 + nevarpunkt(1) ^ 2 + nevarpunkt(2) ^ 2) ^ (1 / 2) / (nsevarpunkt(0) ^
2 + nsevarpunkt(1) ^ 2 + nsevarpunkt(2) ^ 2) ^ (1 / 2)

    sbaphi = Abs(Sgn(sbawinkel) * Atn((sbawinkel ^ 2 / (1 - sbawinkel ^ 2)) ^ (1 / 2)))

    zylzeichdm = eepsilon / (Atn(1) * 4) * 180


    evar = evarpunkt(2) * Cos(sbaphi)

    crn = evar / (1 / Sin(eepsilon / 2) - 1)
    crn = (evar - crn) / (1 / Sin(eepsilon / 2) - 1)

    cr = crn / Cos(sbaphi)


    If Abs(svarpunkt(2) - (median2 + cr)) > 0.5 Then

    median2 = punktschnitt(CDbl(varpunkt(0)), CDbl(varpunkt(1)), median2, CDbl(l2varpunkt(0)),
CDbl(l2varpunkt(1)), CDbl(r2varpunkt(0)), CDbl(r2varpunkt(1)))

    End If

    median2r = median2 * Cos(sbaphi)
    crn = median2r / (1 / Sin(eepsilon / 2) - 1)
    cr = crn / Cos(sbaphi)

    evarpunkt(2) = median2 + cr

    MsgBox cr
End If

ThisDrawing.ActiveLayer = ThisDrawing.Layers("Baumstruktur")

Set aktlinie = ThisDrawing.ModelSpace.AddLine(svarpunkt, evarpunkt)

aktlinie.LinetypeScale = crn * 200

If svarpunkt(2) = 0 Or evarpunkt(2) = 0 Or Abs(svarpunkt(2) - evarpunkt(2)) > 0.5 Then

aktlinie.color = acGreen

End If

'Zylinder zeichnen

Dim apunkt(0 To 2) As Double, bpunkt(0 To 2) As Double, mpunkt(0 To 2) As Double, rbasis(0 To 2) As Double
Dim vektor(0 To 2) As Double


Dim dblZentrum(0 To 2) As Double
Dim dblRadius As Double
Dim dblHoehe As Double
Dim lage As Double, hoch As Double

objZylinder.Delete

    apunkt(0) = aktlinie.StartPoint(0)
    apunkt(1) = aktlinie.StartPoint(1)
    apunkt(2) = 0 'aktlinie.StartPoint(2)

    bpunkt(0) = aktlinie.EndPoint(0)
    bpunkt(1) = aktlinie.EndPoint(1)
    bpunkt(2) = 0 'aktlinie.EndPoint(2)

    mpunkt(0) = (apunkt(0) + bpunkt(0)) / 2
    mpunkt(1) = (apunkt(1) + bpunkt(1)) / 2
    mpunkt(2) = (apunkt(2) + bpunkt(2)) / 2

    'dblRadius = aktlinie.LinetypeScale / 200
```

```
    dblHoehe = ((apunkt(0) - bpunkt(0)) ^ 2 + (apunkt(1) - bpunkt(1)) ^ 2 + (apunkt(2) - bpunkt(2)) ^ 2) ^
(1 / 2)

    Set objZylinder = ThisDrawing.ModelSpace.AddCylinder(mpunkt, zylzeichdm / 2, dblHoehe)

    rbasis(0) = (apunkt(1) - bpunkt(1)) + mpunkt(0)
    rbasis(1) = -(apunkt(0) - bpunkt(0)) + mpunkt(1)
    rbasis(2) = 0 + mpunkt(2)

    vektor(0) = (bpunkt(0) - apunkt(0))
    vektor(1) = (bpunkt(1) - apunkt(1))
    vektor(2) = (bpunkt(2) - apunkt(2))

    lage = (vektor(0) ^ 2 + vektor(1) ^ 2) ^ (1 / 2)
    hoch = vektor(2)

    'winkel = Atn(((lage / dblHoehe) ^ 2 / (1 - (lage / dblHoehe) ^ 2)) ^ (1 / 2)) * Sgn(bpunkt(2) -
apunkt(2))

    If Abs(hoch) > 0.01 Then

    winkel = Atn(lage / hoch) '* Sgn(bpunkt(2) - apunkt(2))

    Else

    winkel = Atn(1) * 2 - Atn(hoch / lage) '* Sgn(bpunkt(2) - apunkt(2))

    End If

    Call objZylinder.Rotate3D(mpunkt, rbasis, winkel)


'----------------------------

Loop Until m = 1

Exit Sub

fehlerroutine:

MsgBox "Fehler! Abbruch!"

ThisDrawing.SetVariable "osmode", 1
objZylinder.Delete

End Sub
```

Makro „Schnittdarstellung“

```
Public Function punktschnitt(abgpunkt1 As Double, abgpunkt2 As Double, abgpunkt3 As Double, aschnittpunkt1
As Double, aschnittpunkt2 As Double, bschnittpunkt1 As Double, bschnittpunkt2 As Double) As Double

On Error GoTo fehlerfunc

Dim n As Double, i As Integer, j As Integer, dx As Integer, dy As Integer
Dim npunkt(0 To 2) As Double, apunkt(0 To 2) As Double, bpunkt(0 To 2) As Double, punkt(0 To 2) As Double,
vpunkt(0 To 2) As Double, zpunkt(0 To 2) As Double, abgpunkt(0 To 2) As Double, abgpunktnord(0 To 2) As
Double, abgpunktost(0 To 2) As Double
Dim laenge As Double, entf As Double
Dim min(0 To 2) As Double, max(0 To 2) As Double
Dim minv(0 To 2) As Double, maxv(0 To 2) As Double
Dim abgpunktobjnord, abgpunktobjost As AcadXline

Set objlayer = ThisDrawing.Layers.Add("Schnittpunkte")
'objlayer.color = acGreen
ThisDrawing.ActiveLayer = ThisDrawing.Layers("Schnittpunkte")

ThisDrawing.SetVariable "osmode", 0

'aschnittpunkt = ThisDrawing.Utility.GetPoint(, "Startpunkt:")
'bschnittpunkt = ThisDrawing.Utility.GetPoint(, "Endpunktpunkt:")

Dim doppelweite As Double
Dim doppelhoehe As Double
Dim doppelfaktor As Double
Dim varzentrum As Variant
Dim varbildschirm As Variant

' aktuelles Zoomfenster bestimmen

With ThisDrawing

varzentrum = .GetVariable("VIEWCTR")
varbildschirm = .GetVariable("SCREENSIZE")
doppelhoehe = .GetVariable("VIEWSIZE")
doppelfaktor = varbildschirm(0) / varbildschirm(1)
doppelweite = doppelhoehe * doppelfaktor
```

```
End With

minv(0) = Round(varzentrum(0) - (doppelweite / 2), 4)
minv(1) = Round(varzentrum(1) - (doppelhoehe / 2), 4)
maxv(0) = Round(varzentrum(0) + (doppelweite / 2), 4)
maxv(1) = Round(varzentrum(1) + (doppelhoehe / 2), 4)
'----------

apunkt(0) = (aschnittpunkt1 - 0) / 0.018
apunkt(1) = (aschnittpunkt2 + 67.95) / 0.018
apunkt(2) = 0

bpunkt(0) = (bschnittpunkt1 - 0) / 0.018
bpunkt(1) = (bschnittpunkt2 + 67.95) / 0.018
bpunkt(2) = 0

laenge = ((bpunkt(0) - apunkt(0)) ^ 2 + (bpunkt(1) - apunkt(1)) ^ 2 + (bpunkt(2) - apunkt(2)) ^ 2) ^ (1 /
2)

npunkt(0) = (bpunkt(0) - apunkt(0)) / laenge
npunkt(1) = (bpunkt(1) - apunkt(1)) / laenge
npunkt(2) = (bpunkt(2) - apunkt(2)) / laenge

' Abgriffspunkt zeichnen

abgpunkt(0) = (((abgpunkt1 - 0) / 0.018 - apunkt(0)) * npunkt(0) + ((abgpunkt2 + 67.95) / 0.018 -
apunkt(1)) * npunkt(1)) * 0.018 + 0
abgpunkt(1) = abgpunkt3
abgpunkt(2) = 0

abgpunktost(0) = abgpunkt(0) + 10
abgpunktost(1) = abgpunkt(1)
abgpunktost(2) = 0

abgpunktnord(0) = abgpunkt(0)
abgpunktnord(1) = abgpunkt(1) + 10
abgpunktnord(2) = 0

Set abgpunktobjost = ThisDrawing.ModelSpace.AddXline(abgpunkt, abgpunktost)
Set abgpunktobjnord = ThisDrawing.ModelSpace.AddXline(abgpunkt, abgpunktnord)
'---------------

min(0) = 0
min(1) = 0
max(0) = 0
max(1) = 0

For i = 0 To Round(laenge)
    For j = -2 To 2

        punkt(0) = Round(apunkt(0) + i * npunkt(0) + j * npunkt(1)) * 0.018 + 0
        punkt(1) = Round(apunkt(1) + i * npunkt(1) + j * npunkt(0)) * 0.018 - 67.95

        If entfdaten(Round(apunkt(0) + i * npunkt(0) + j * npunkt(1)), Round(apunkt(1) + i * npunkt(1) + j
* npunkt(0))) = 0 Then
            n = CDbl(entfdaten(Round(apunkt(0) + i * npunkt(0) + j * npunkt(1)), Round(apunkt(1) + i *
npunkt(1) + j * npunkt(0))))
        Else
            n = (CDbl(entfdaten(Round(apunkt(0) + i * npunkt(0) + j * npunkt(1)), Round(apunkt(1) + i *
npunkt(1) + j * npunkt(0)))) + 32000) / 1000
        End If

        punkt(2) = n

        If punkt(2) > 0.2 Then

            zpunkt(0) = i * 0.018 + 0
            zpunkt(1) = punkt(2)
            zpunkt(2) = 0

            ThisDrawing.ModelSpace.AddPoint (zpunkt)

            If zpunkt(0) < minx Then
               min(0) = zpunkt(0)
            End If
            If zpunkt(0) > maxx Then
               max(0) = zpunkt(0)
            End If
            If zpunkt(1) < miny Then
               min(1) = zpunkt(1)
            End If
            If zpunkt(1) > maxy Then
               max(1) = zpunkt(1)
            End If

        End If

    Next j
Next i
```

```
ThisDrawing.ActiveLayer = ThisDrawing.Layers("Schnittpunkte")

ThisDrawing.Layers("Intensitaetsbild").LayerOn = False
ThisDrawing.Layers("Entfernungsbild").LayerOn = False
ThisDrawing.Layers("Baumstruktur").LayerOn = False
'ThisDrawing.la

ZoomWindow min, max

entfabgriff = ThisDrawing.Utility.GetPoint(, "Entfernung bestimmen:")

punktschnitt = CDbl(entfabgriff(1))

Dim ss As AcadSelectionSet
Dim FilterType(0) As Integer
Dim FilterData(0) As Variant

On Error Resume Next

Set ss = ThisDrawing.SelectionSets.Add("aschnitt")
If Err.Number <> 0 Then
  Set ss = ThisDrawing.SelectionSets.Item("aschnitt")
End If

FilterType(0) = 8
FilterData(0) = "Schnittpunkte"

ss.Select acSelectionSetAll, , , FilterType, FilterData

'MsgBox "The selectionset contains:" & ss.Count & " entities."

ss.Erase

ss.Clear

'ThisDrawing.Application.ZoomPrevious
ZoomWindow minv, maxv

ThisDrawing.SetVariable "osmode", 1

ThisDrawing.Layers("Intensitaetsbild").LayerOn = True
'ThisDrawing.Layers("Entfernungsbild").LayerOn = True
ThisDrawing.Layers("Baumstruktur").LayerOn = True

Exit Function

fehlerfunc:

MsgBox "Fehler! Abbruch!"

End Function
```

Makro „Transformation“

```
Sub linientrans()

    Dim objlayer As AcadLayer

    Dim inspunkt(0 To 2), kpunkt(0 To 2), epunkt(0 To 2), spunkt(0 To 2), wpunkt(0 To 2) As Double
    Dim apunkt, bpunkt As Variant
    Dim rotation(0 To 2, 0 To 2) As Double, translat(0 To 2) As Double
    Dim transdatei As String, transdateinr As Integer
    Dim r As String

    Dim oAcadLine As AcadLine
    Dim oEntity As AcadEntity
    Dim Point As Variant
    Dim objzeich As AcadObject
    Dim lin As AcadLine

    Set zeichenebene = ThisDrawing.Application.ActiveDocument

    Set strukturenebene = ThisDrawing.Application.Documents.Add

    Set objlayer = strukturenebene.Layers.Add(Transformation.TextBox7.Value)

    strukturenebene.ActiveLayer = strukturenebene.Layers(Transformation.TextBox7.Value)


    transdatei = Transformation.TextBox2.Value
    transdateinr = FreeFile()

    Open transdatei For Input As transdateinr

    Line Input #transdateinr, r
    rotation(0, 0) = Val(Mid(r, 1, 13))
    rotation(0, 1) = Val(Mid(r, 15, 13))
```

```
    rotation(0, 2) = Val(Mid(r, 29, 13))
    translat(0) = Val(Mid(r, 43, 16))

    Line Input #transdateinr, r
    rotation(1, 0) = Val(Mid(r, 1, 13))
    rotation(1, 1) = Val(Mid(r, 15, 13))
    rotation(1, 2) = Val(Mid(r, 29, 13))
    translat(1) = Val(Mid(r, 43, 16))

    Line Input #transdateinr, r
    rotation(2, 0) = Val(Mid(r, 1, 13))
    rotation(2, 1) = Val(Mid(r, 15, 13))
    rotation(2, 2) = Val(Mid(r, 29, 13))
    translat(2) = Val(Mid(r, 43, 16))

    'MsgBox translat(0)

    Close transdateinr

    'Transformation der Linien

    For Each objzeich In zeichenebene.ModelSpace

    'If objzeich.ObjectName = "AcDbLine" And objzeich.Layer = zeichenebene.ActiveLayer.Name Then
    If objzeich.ObjectName = "AcDbLine" Then
        Set oAcadLine = objzeich
        With oAcadLine

        inspunkt(0) = .StartPoint(0)
        inspunkt(1) = .StartPoint(1)
        inspunkt(2) = .StartPoint(2)

        If inspunkt(2) = 0 Then

        inspunkt(2) = 0.2

        End If

        If (inspunkt(0) >= 0) And (inspunkt(0) < 90) Then

            spunkt(0) = 90 - inspunkt(0)

            epunkt(0) = inspunkt(2) * Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        If (inspunkt(0) >= 90) And (inspunkt(0) < 180) Then

            spunkt(0) = 90 - inspunkt(0)

            epunkt(0) = inspunkt(2) * Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        If (inspunkt(0) >= 270) And (inspunkt(0) < 360) Then

            spunkt(0) = inspunkt(0) - 270

            epunkt(0) = inspunkt(2) * -Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        If (inspunkt(0) >= 180) And (inspunkt(0) < 270) Then

            spunkt(0) = inspunkt(0) - 270

            epunkt(0) = inspunkt(2) * -Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        kpunkt(0) = rotation(0, 0) * epunkt(0) + rotation(0, 1) * epunkt(1) + rotation(0, 2) * epunkt(2)
        kpunkt(1) = rotation(1, 0) * epunkt(0) + rotation(1, 1) * epunkt(1) + rotation(1, 2) * epunkt(2)
        kpunkt(2) = rotation(2, 0) * epunkt(0) + rotation(2, 1) * epunkt(1) + rotation(2, 2) * epunkt(2)

        wpunkt(0) = kpunkt(0) + translat(0)
        wpunkt(1) = kpunkt(1) + translat(1)
        wpunkt(2) = kpunkt(2) + translat(2)
```

```
        apunkt = CVar(wpunkt)

        inspunkt(0) = .EndPoint(0)
        inspunkt(1) = .EndPoint(1)
        inspunkt(2) = .EndPoint(2)

        If inspunkt(2) = 0 Then

        inspunkt(2) = 0.2

        End If


        If (inspunkt(0) >= 0) And (inspunkt(0) < 90) Then

            spunkt(0) = 90 - inspunkt(0)

            epunkt(0) = inspunkt(2) * Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        If (inspunkt(0) >= 90) And (inspunkt(0) < 180) Then

            spunkt(0) = 90 - inspunkt(0)

            epunkt(0) = inspunkt(2) * Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        If (inspunkt(0) >= 270) And (inspunkt(0) < 360) Then

            spunkt(0) = inspunkt(0) - 270

            epunkt(0) = inspunkt(2) * -Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        If (inspunkt(0) >= 180) And (inspunkt(0) < 270) Then

            spunkt(0) = inspunkt(0) - 270

            epunkt(0) = inspunkt(2) * -Cos(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(1) = inspunkt(2) * Sin(spunkt(0) / 180 * 4 * Atn(1)) * Cos(inspunkt(1) / 180 * 4 *
Atn(1))
            epunkt(2) = inspunkt(2) * Sin(inspunkt(1) / 180 * 4 * Atn(1))
        End If

        kpunkt(0) = rotation(0, 0) * epunkt(0) + rotation(0, 1) * epunkt(1) + rotation(0, 2) * epunkt(2)
        kpunkt(1) = rotation(1, 0) * epunkt(0) + rotation(1, 1) * epunkt(1) + rotation(1, 2) * epunkt(2)
        kpunkt(2) = rotation(2, 0) * epunkt(0) + rotation(2, 1) * epunkt(1) + rotation(2, 2) * epunkt(2)

        wpunkt(0) = kpunkt(0) + translat(0)
        wpunkt(1) = kpunkt(1) + translat(1)
        wpunkt(2) = kpunkt(2) + translat(2)

        bpunkt = CVar(wpunkt)

        End With

        strukturenebene.ActiveLayer = strukturenebene.Layers(Transformation.TextBox7.Value)

        Set lin = strukturenebene.ModelSpace.AddLine(apunkt, bpunkt)

        lin.LinetypeScale = oAcadLine.LinetypeScale
        lin.Lineweight = acLnWt040

        'zeichenebene.ActiveLayer = zeichenebene.Layers("baumstruktur")

    End If

    Next

    Set stpunkt = ThisDrawing.ModelSpace.AddCircle(translat, 0.1)
    stpunkt.color = acGreen

    ZoomExtents
End Sub
```

Makro „Zylinderdarstellung“

```
Sub zylinder()

On Error Resume Next

    Dim objlayer As AcadLayer

    Dim apunkt(0 To 2) As Double, bpunkt(0 To 2) As Double, mpunkt(0 To 2) As Double, rbasis(0 To 2) As
Double
    Dim vektor(0 To 2) As Double
    Dim oAcadLine As AcadLine
    Dim oEntity As AcadEntity
    Dim Point As Variant
    Dim objzeich As AcadObject
    Dim lin As AcadLine

    Dim objZylinder As Acad3DSolid
    Dim dblZentrum(0 To 2) As Double
    Dim dblRadius As Double
    Dim dblHoehe As Double
    Dim lage As Double, hoch As Double

    Set objlayer = ThisDrawing.Layers.Add("Zylinderdarstellung")

    ThisDrawing.Layers("Zylinderdarstellung").color = acRed

    ThisDrawing.ActiveLayer = ThisDrawing.Layers("Zylinderdarstellung")

    For Each objzeich In ThisDrawing.ModelSpace

    If objzeich.ObjectName = "AcDbLine" Then 'And objzeich.Layer = zeichenebene.ActiveLayer.Name Then

        Set oAcadLine = objzeich

        apunkt(0) = oAcadLine.StartPoint(0)
        apunkt(1) = oAcadLine.StartPoint(1)
        apunkt(2) = oAcadLine.StartPoint(2)

        bpunkt(0) = oAcadLine.EndPoint(0)
        bpunkt(1) = oAcadLine.EndPoint(1)
        bpunkt(2) = oAcadLine.EndPoint(2)

        mpunkt(0) = (apunkt(0) + bpunkt(0)) / 2
        mpunkt(1) = (apunkt(1) + bpunkt(1)) / 2
        mpunkt(2) = (apunkt(2) + bpunkt(2)) / 2

        dblRadius = objzeich.LinetypeScale / 200

        dblHoehe = ((apunkt(0) - bpunkt(0)) ^ 2 + (apunkt(1) - bpunkt(1)) ^ 2 + (apunkt(2) - bpunkt(2)) ^
2) ^ (1 / 2)

        Set objZylinder = ThisDrawing.ModelSpace.AddCylinder(mpunkt, dblRadius, dblHoehe)

        rbasis(0) = (apunkt(1) - bpunkt(1)) + mpunkt(0)
        rbasis(1) = -(apunkt(0) - bpunkt(0)) + mpunkt(1)
        rbasis(2) = 0 + mpunkt(2)

        vektor(0) = (bpunkt(0) - apunkt(0))
        vektor(1) = (bpunkt(1) - apunkt(1))
        vektor(2) = (bpunkt(2) - apunkt(2))

        lage = (vektor(0) ^ 2 + vektor(1) ^ 2) ^ (1 / 2)

        hoch = vektor(2)

        If Abs(hoch) > 0.01 Then

        winkel = Atn(lage / hoch) '* Sgn(bpunkt(2) - apunkt(2))

        Else

        winkel = Atn(1) * 2 - Atn(hoch / lage) '* Sgn(bpunkt(2) - apunkt(2))

        End If

        Call objZylinder.Rotate3D(mpunkt, rbasis, winkel)


    End If

    Next

    ZoomExtents

End Sub
```

LEBENSLAUF

PERSÖNLICHE DATEN

• Name	Grafl Andreas
• Anschrift	Ardaggerstraße 118, 3300 Amstetten
• Geburtsdatum	13.08.1985
• Geburtsort	Amstetten
• Staatsbürgerschaft	Österreich
• E-Mail Adresse	andreas.grafl@gmx.at
• Familienstand	ledig
• Eltern	Hermann Grafl Hauptschullehrer Walpurga Grafl Volksschullehrerin
• Geschwister	Wolfgang Grafl Dipl. Ing. Fachrichtung Raumplanung Angestellter Eva Grafl Angestellte

AUSBILDUNG

• 1992 – 1996	Volksschule Elsa Brändström Amstetten
• 1996 – 2004	Bundesgymnasium Amstetten
• Juni 2004	Reifeprüfung/Matura
• 2004 – 2005	Präsenzdienst Bundesheer
• Oktober 2005	Inskription an der Technischen Universität Wien Bachelorstudium Geodäsie und Geoinformatik
• Oktober 2009	Abschluss Bachelor of Science
• November 2009	Inskription an der Technischen Universität Wien Masterstudium Vermessung und Kataster
• November 2012	Abschluss Master of Science/Dipl. Ing.