Kern, F.: Vorschlag für ein binäres, offenes TLS-Austauschformat. In: Riedel, B. & Schäfer, M. & Niemeier, W. (Red.): Terrestrisches Laserscanning – TLS 2010 mit TLS-Challenge. Beiträge zum 101. DVW-Seminar am 6. und 7. Dezember 2010 in Fulda, DVW-Schriftenreihe Band 64, Wißner, Augsburg, 2010, S. 27-42

Vorschlag für ein binäres, offenes TLS-Austauschformat

Prof. Dr.-Ing. Fredie Kern
Fachhochschule Mainz – i3mainz
Lucy-Hillebrand-Straße 2
55 128 Mainz
kern@fh-mainz.de

1 Einführung

Die Anwendungsgebiete im Bereich des terrestrischen Laserscannings (TLS) sind sehr weit gefächert. Daher werden innerhalb eines Projektes häufig verschiedene Softwareprodukte, die jeweils nur für bestimmte Aufgaben spezialisiert sind, in einer Prozesskette eingesetzt. Damit besteht ein unmittelbarer Bedarf, TLS-Daten möglichst verlustfrei, schnell und bequem zwischen verschiedenen Programmen auszutauschen. Lösungen, die auf Textdateien (ASCII) beruhen, sind hierfür aufgrund des immensen Speicherbedarfs nicht praxisgerecht und leiden zudem unter kleinen aber störenden Kompatibilitätsproblemen. Auch die Archivierung von TLS-Daten - ein wichtiger Aspekt bei geisteswissenschaftlichen Projekten [1] - erfordert ein einheitliches Datenformat, das hersteller- und systemunabhängig ist sowie vollständig dokumentiert und frei von Patentrechten u.ä. ist. Abhilfe kann hier nur ein Datenformat schaffen, das auf breiten Konsens der Hard- und Softwarehersteller sowie der Anwender fußt.

1.1 Struktur von TLS-Daten

Ein TLS-System führt in einer Vielzahl von Raumrichtungen innerhalb seines Gesichtsfelds, oder in Ausschnitten davon, reflektorlose Streckenmessungen D_i aus. Durch die zwei Ablenkungen (α und β) des Laserstrahls im Abtaster werden diese Raumrichtungen, vergleichbar der Ausrichtung des Fernrohres eines Tachymeters, realisiert. Das Ablenken geschieht - grob betrachtet - gestuft mit den festen Winkelinkrementen $\Delta\alpha$ und $\Delta\beta$. Ein Scanvorgang erzeugt somit eine

Punktwolke, die bezüglich der Ablenkrichtung α und β rasterförmig angeordnet werden kann (Abb. 1). Dabei vernachlässigt man, dass bei vielen TLS-Systemen die Ausrichtung des Laserstrahls durch eine kontinuierliche Bewegung erreicht wird, bei der zu diskreten Zeitpunkten die Raumstrecken gemessen werden (Abb. 1).

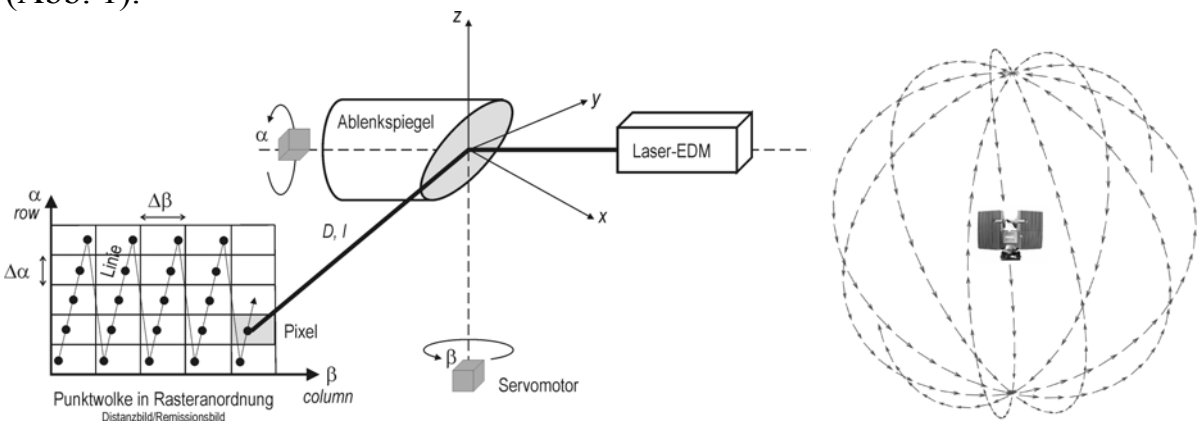
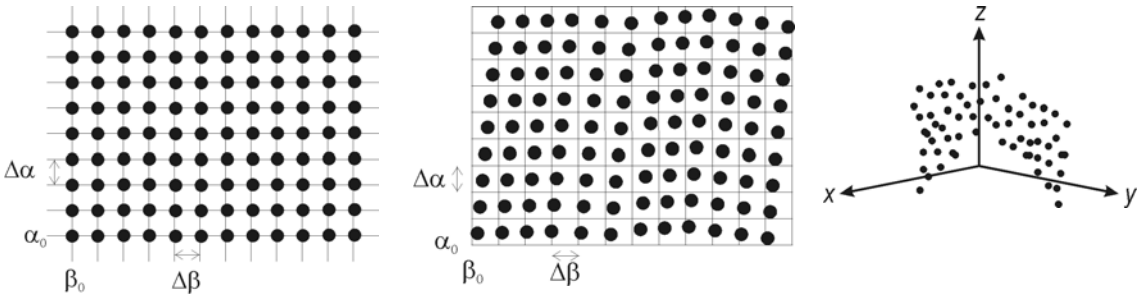


Abb. 1: Strukturelle Entstehung und Aufbau einer Einzelpunktwolke (links) und räumliche Darstellung der kontinuierlichen Abtastbewegung eines TLS mit $\Delta\alpha$ = 20gon und $\Delta\beta$=10gon über die volle Hemisphäre.

Die gemessen Ablenkwinkel α und β entsprechen somit nur annähernd den Mittelpunkten der Rasterzellen. Die β-Linien verlaufen schräg zur α-Achse und in dieser Darstellung parallel zueinander. Üblicherweise werden die Messungen α_i, β_i und D_i in kartesischen Koordinaten x_i, y_i ,z_i im lokalen TLS-Sensor-System gespeichert, wobei der Ursprung durch den Schnittpunkt der Rotationsachsen für die Ablenkungen α und β gegeben ist. Gemäß der rasterförmigen Abtastung werden die Messungen (*Pixel*) sequentiell - der „Bewegung" des Laserstrahls gemäß - Linie für Linie gespeichert; es ergibt sich eine Messreihenfolge, die für die Auswertung von Bedeutung sein kann. Eine von einem Standpunkt aus erfasste Punktwolke kann daher als Distanzbild mit *N*-vielen Spalten (*column*) und *M*-vielen Zeilen (*row*) aufgefasst werden, wobei anstelle des Grauwertes die *x*-, *y*-, *z*-Koordinaten und der Remissionswert *I* gespeichert werden.

Vorausgesetzt die gemessenen polaren Elemente α, β und *D* würden von einem TLS-System in strenger Rasteranordnung, ob nun mit oder ohne schräg stehenden Linien, erfasst und es wären keinerlei Kalibrierwerte oder Reduktionen anzubringen, so könnten die Distanzen (und weitere Werte wie z.B. der Remissionswert) ohne die explizite Angabe der Raumrichtung in der Struktur einer Bildmatrix gespeichert werden (Abb. 2 Typ A). Dies entspricht der Speicherung eines Raster-DGMs mit einem in der Praxis ärgerlichen Unterschied, dass die

Reihenfolge, in der die Pixel vom TLS aufgenommen und gespeichert wird, spaltenweise erfolgt und die Reihenfolge der Bildpixel zeilenweise geschieht; was einer Transponierung der Rastermatrix gleich kommt. Das Tagged-Image-File-Format (TIFF, [2]) ist ein anerkanntes Datenformat für den professionelle Austausch von Rasterdaten; nicht zuletzt, weil mehrkanalige Informationen gespeichert und Farbtiefen von 8-Bit-unsigned-int, 16-Bit-unsigned-int, float oder double möglich sind. Weiterhin erlaubt es auch die Aufnahme von Meta-Informationen incl. Angaben zur Georeferenzierung (GeoTIFF, [3]).



Typ A	Typ B	Typ C
strenge Rasteranordnung	Pseudo-Rasteranordnung	unstrukturierte Ansammlung
z.B.: GIS-Rasterdaten	z.B.: PTX, PTG, OSF	z.B.: PTS, OBJ, PLY, WRL

Abb. 2: Punktwolken bestehend aus dreidimensionalen Koordinaten (eventuell erweitert um weitere Attribute wie Remissionswert und RGB-Farbwert) können beim TLS in drei Typen auftreten.

Durch die Notwendigkeit interne Kalibrierungen an den originären Messungen vornehmen zu müssen und durch den oben geschilderten kontinuierlichen Messprozess selbst, ist eine streng gitterförmige Anordnung nicht garantiert. Da die Abweichung zur Rasterzellenmitte in der Regel gering sind, die von einem Standpunkt aufgenommen Punktwolke nur 2,5-D hat und die Messreihenfolge dokumentiert werden soll, ist die Speicherung in Pseudo-Rasteranordnung zweckmäßig und äußerst sinnvoll (Abb. 2 Typ B). Dabei wird eine TLS-Punktwolke als Matrix gespeichert wobei jedes Matrixelement ein Koordinaten-Tripel enthält und die Zeilen- und Spaltenindizes durch die Soll-Abtaststellen definiert sind. Beispiele für das Abspeichern einer Typ-B-Punktwolke sind das PTX- und PTG-Format. Aber auch die nativen (proprietären) Datenformate, wie z.B. das ZFS, verwenden die Matrixvorstellung. Nachteilig an der Speicherung in einer Rasteranordnung ist, dass zum Teil erhebliche Matrixbereiche unbelegt bleiben, weil in diese Richtung kein Messpunkt bestimmt werden konnte z.B. weil die Entfernung zum Objekt außerhalb des Messbereiches liegt (Himmel, weit entfernter Objekthintergrund). Sehr vorteilhaft an einer Rasterstruktur ist die impli-

zite Dokumentation des Messablaufes und der leicht realisierbare indizierte Zugriff auf Teile der Punktwolkenmatrix.

Für unstrukturierte Punktwolken, bei denen entweder die Zuordnung zu einer Abtastrasterzelle, im Rahmen eines Auswerteschrittes, z.B. nach der Georeferenzierung und Zusammenführung mehrerer Punktwolken, verloren gegangen ist oder die sich nicht mehr verebnen lassen; also echte 3D-Punktwolken sind, sind andere Arten der Abspeicherung erforderlich. Für diese Typ-C-Punktwolken sind einfache ASCII Koordinatenlisten (*xyz*-Datei) oder um einen Intensitätswert und Farbwerte erweiterte ASCII-Listen (PTS-Format) im Gebrauch.

1.2 Wertebereiche, Quantisierung und Datentypen

Beschränkt man sich auf den Entwurf eines Datenformates zur Speicherung, Archivierung und zum Austausch von Punktwolken, welche im Rahmen des Terrestrischen Laserscannings erfasst werden, so können hilfreiche Annahmen über den Wertebereich der Messungen getroffen werden (Tab. 1). So kann z.B. der Entfernungsmessbereich auf etwa zehn Kilometer beschränkt werden und die Messgenauigkeit ist deutlich schlechter als ein Zehntel Millimeter. Die Tab. 1 stellt für die möglichen Messinformationen eines TLS die typischen Wertebereiche, erforderlichen Auflösungen (Quantisierung) und dem daraus resultierenden Speicherbedarf zusammen.

Tab. 1: Angaben zu TLS-Informationen

Information	**Formelzeichen**	**typische Wertebereich je Element u. Einheit**	**Quantisierung**	**Datentyp für Elementgruppe**	**Speicherbedarf**
Koordinaten	*(x,y,z)* *(X,Y,Z)*	[-10.000,10.000] m [-1.000.000,1.000.000] m	0,1mm 0,001mm	float[3] o. long[3] double[3]	12 Byte 30 Byte
Remission/ Intensität	*i* *I* *P*	[0,255] 8-Bit [0,4095] 16-Bit [0,∞(	1 Grauwert 1 Grauwert 1 dB	unsigned char unsigned int float	1 Byte 2 Byte 4 Byte
Farbwert	*(r,g,b)* *R,G,B*	[0,255] 8-Bit [0,4095] 16-Bit	1 Grauwert 1 Grauwert	unsigned char[3] unsigned int[3]	3 Byte 6 Byte
Raumstrecke	*d* *D*	[0,10.000] m [0,1.000.000] m	0,1 mm 0,001 mm	float double	4 Byte 10 Byte
Ablenkung α	*a*	[0,400] gon	0,1 mgon	float	4 Byte
Ablenkung β	*b*	[0,400] gon	0,1 mgon	float	4 Byte

Für eine Speicherung einer 12,5 Mio. Punkte umfassenden Punktwolke (*2.500* x *5.000* Pixel) mit den Informationen UTM-Koordinaten (*X,Y,Z*), Remissionswert

P und Farbwerte (*r,g,b*) im PTG-Format sind nach diesen Angaben $12{,}5 \cdot 10^6 \cdot (30+4+3)$ Byte = ca. *440* MByte Speicherplatz erforderlich. Bei einer (mit vertretbaren Verlusten behafteten) Speicherung der wesentlichen Messgrößen *D* und *I* als Rastermatrix (Typ-A, *d, P* je Pixel) ergebe sich lediglich eine Dateigröße von $12{,}5 \cdot 10^6 \cdot (4+4)$ Byte = ca. *95* MByte. Diese Angaben berücksichtigen noch keine Komprimierung, wie sie bei allen Bildformaten üblich sind. Mögliche Komprimierungsverfahren, deren Komplexität gering ist und für die Standards sowie Open-Source-Implementierungen existieren, wären eine Lauflängenkodierung (RLE) oder eine Lempel-Ziv-Welch-Komprimierung (LZW).

1.3 Anforderungen an ein TLS-Austauschformat

Folgende Minimal-Anforderungen sind an ein TLS-Austauschformat aus Anwendersicht zu stellen:

- vollständig dokumentiert
- hersteller-, produkt- und betriebssystemunabhängig
- Metainformationen incl. Georeferenzierung
- Speicherung der rohen Messwerte (incl. Herstellerkalibrierung) einer Einzelpunktwolke (Typ-B) unter Beibehaltung der Erfassungsreihenfolge
- optionale Speicherung einer Typ-C-Punktwolke
- Speicherung der *X,Y,Z*-Koordinaten und der Remissionswerte (optional) und der *RGB*-Werte (optional) in verschiedenen Quantisierungen
- möglichst einfach zu implementieren und zu handhaben
- Ressourcenschonend und performant
- leicht erweiterbar im Hinblick auf zukünftige Geräteentwicklungen oder eigenen Ergänzungen
- Implementierungen als Open Source

Neben der möglichst breiten Unterstützung eines TLS-Austauschformates bei den Softwaresystemen zur TLS-Auswertung spielen auch Performancegesichtspunkte hinsichtlich der immer noch ansteigenden Datenmengen eine gewichtige Rolle in der Praxis. Schon jeder Anwender hat mehrere Stunden oder Tage mit den Export und Import verbracht. Man fragt sich in diesen Momenten, was daran so schwer und zeitaufwändig ist *XYZ*-Koordinaten von A nach B zu kopieren. Dem Autor fiel diesbezüglich z.B. der PTG-Export aus Cyclone 7.0.3 mit einer Konvertierungszeit von ca. 30 Minuten und einer zwischenzeitlichen Vervierfachung der Datenmenge auf der Festplatte negativ auf.

1.4 TLS-Austauschformate im Auswerteprozess

Die im Laufe einer TLS-Messung und Auswertung entstehenden Datenformate lassen sich wie folgt grob kategorisieren, wobei häufig fließende Übergänge festzustellen sind (Abb. 3).

1. Speicherung der Punktwolke (**originäre Messungen**) in Messreihenfolge mit zusätzlichen hardwarespezifischen Attributen; Berücksichtigung interner Kalibrierungsparameter und Filterungen sowie Quantisierung (Analog-Digital-Wandlung)
   Vertreter dieser Kategorie sind z.B.: Faro FLS, Leica Cyclone IMG/PCE, Zoller&Fröhlich ZFS, Riegl 3DD.
   Für Entwickler von herstellerunabhängiger Software werden geeignete Software Development Kits (SDKs) kostenpflichtig lizenziert; z.B. [8].
2. Speicherung der Punktwolke (**rohe Messdaten**) in Messreihenfolge (Pseudoraster); Parameter zur Georeferenzierung sind enthalten; Daten sind für die Weiterverarbeitung mit Fremdsoftware geeignet.
   Vertreter dieser Kategorie sind z.B.: Cyclone PTG Point Cloud Format, Open Scan Format OSF, E57
   Diese Datenformate sind in der Regel offen dokumentiert.
3. Speicherung eines **3D-Modells**, das durch die Auswertung aus den Punktwolken abgeleitet wurde; echtes 3D-Modell mit rekonstruierten Geometrien (Punkte, Linien, Flächen) und der Topologie.
   Vertreter dieser Kategorie sind z.B: VRML/WRL, Google Earth KML

Für den Anwender interessant – insb. unter dem Archivierungsaspekt – sind die Formate der Kategorie 2, soweit ihre Spezifikationen offen dokumentiert sind. Die Formate der Kategorie 1 sind hierfür explizit, trotz ihrer weiten Verbreitung und der Verfügbarkeit von SDKs, nicht geeignet. Sie sind nicht offen und unterliegen zu sehr der Willkür des Herstellers.

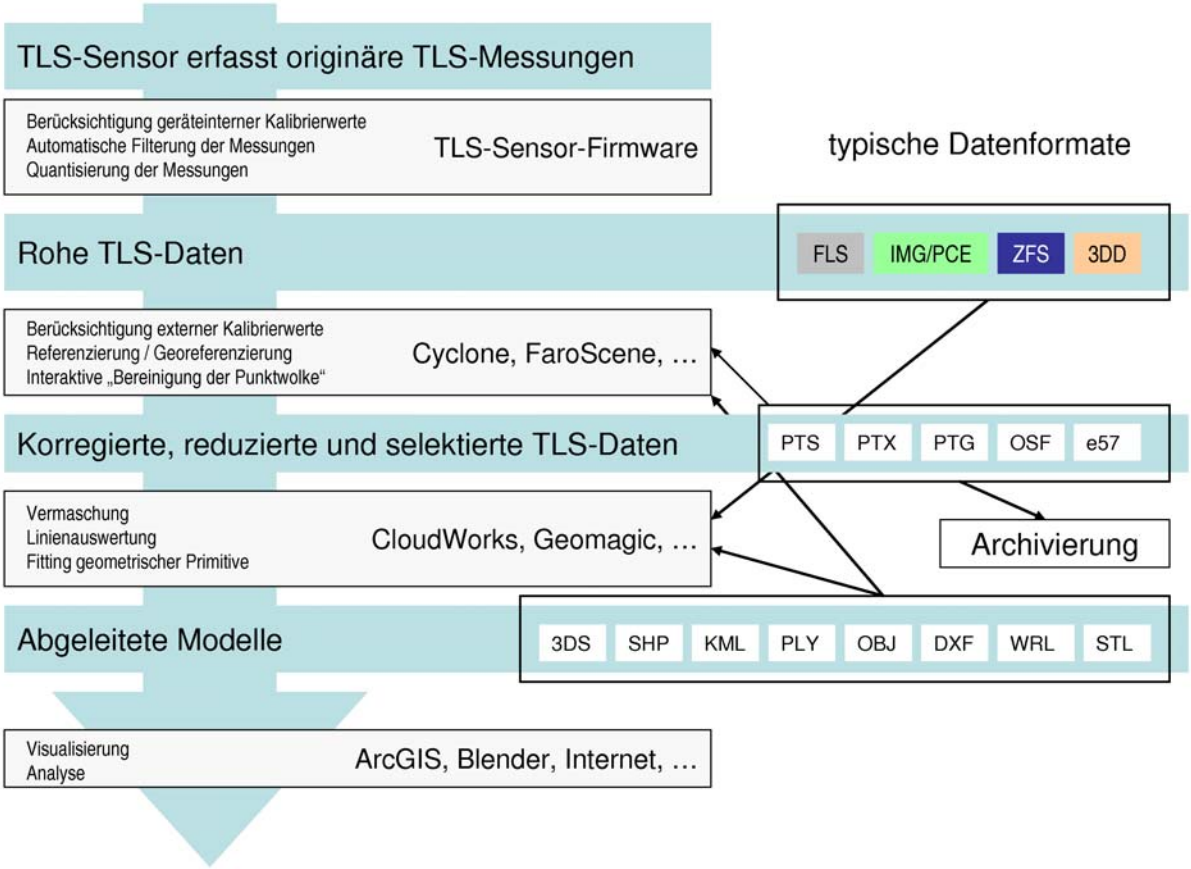


Abb. 3: Die Rollen der Datenformate im TLS-Erfassungs-, Auswerte- und Archivierungsprozess. Einige Formate erlauben den „mißbräulichen“ Einsatz als Punktwolken-Container (z.B. DXF, VRML)

1.5 Vorhandene TLS-Datenformate

Die Speicherung von Punktwolken erfordert in ihren einfachsten Formen nur eine sehr simple Datenstruktur. Bereits eine zeilenweise Ausgabe der *x, y, z*-Werte als ASCII-Datei erfüllt wesentliche Anforderungen zum Datenaustausch zwischen zwei Programmen. Da die Datenstruktur so einfach ist, verwundert es nicht, dass so viele verschiedene Datenformate von Hardware- und Softwareherstellern erfunden und bereitgehalten werden. Auch hinsichtlich der binären Speicherung existiert eine schier endlose Liste an möglichen Formaten; vielfach aus dem Bereich der Computergrafik (Wavefront OBJ, PLY, WRL, 3D-Studio-Max, …), dem konstruktiven Maschinenbau (DXF, STL) und dem GIS-Bereich (KML, Shapefile).

Eine Diskussion der speziellen TLS-Formate (PTS, PTX, PTB, i3mainz-TLS-Format, ASPRS-LAS [4]) findet sich in [5]. An dieser Stelle soll noch ergänzend auf das Cyclone PTG Point Cloud Format [6] und den in der Konzeption befindlichen Standard „ASTM E57 3D file format“ kurz E57 [7] hingewiesen werden. Das E57 wird demnächst als Standard der American Society of Testing and Materials (ASTM) vom Subcommittee E57.04 „Data Interoperability“ des Committee E57 „3D Imaging Systems“ unter Mitwirkung der marktführenden TLS-Hersteller erscheinen. Derzeit existiert bereits – bzw. nur – eine Referenzimplementierung [8], der man aber schwer entnehmen kann welche Funktionalitäten und Speichertechniken dieses Format hat bzw. einmal haben wird. Interessant ist hier die kombinierte Speicherung im XML-Format und in binärer Form.

Tab. 2 verdeutlicht die Leistungsfähigkeit dieser neuen Formate im Vergleich mit den häufig in Anwendung befindlichen Formaten PTS und PTG sowie dem in diesem Aufsatz vorgestellten Open Scan Format (OSF).

Tab. 2:Eigenschaften verschiedener TLS-Datenformate

Format	**Metadaten**	**Georeferenzierung**	**Typen**	**Encoding**	**Kompression**	**Anz. Punktwolken**	**Informationen (Notation gemäß Tab. 1)**
PTS	nein	nein	C	ASCII	nein	∞	*XYZ[I][RGB]*
PTX	nein	4x4-Matrix	B	ASCII	nein	∞	*XYZIrgb*
PTG	binär	4x4-Matrix	B	binary	nein	1	*(xyz|XYZ)[I][rgb]*
LAS	binär	GeoTIFF	C	binary	nein	1	*xyzi[rgb][GPS-Zeit][…]*
OSF	XML	4x4-Matrix	B u. C	binary	nein	1	*xyz[i][rgb]*
E57	XML	ja	B? u. C	binary, XML	ja?	1?	*(xyz|XYZ)[i|I][rgb|RGB]*

2 Open Scan Format - OSF

2.1 Allgemeines

In den nachfolgenden Abschnitten wird die Format-Spezifikation Version 1.04 [10] erläutert, wie sie erstmalig im Rahmen der Sitzung des Offenen Forums Terrestrisches Laserscanning (http://www.laserscanning.org) auf den Oldenburger 3D-Tagen Anfang 2010 vorgestellt wurde.

Unter Punktwolke wird beim OSF-Format eine Menge von Messpunkten verstanden. Jeder Messpunkt wird dabei durch seine dreidimensionalen, kartesischen Koordinaten bezüglich eines gerätebezogene Sensorkoordinatensystem räumlich-geometrisch beschrieben. Ursprung des Sensorkoordinatensystems ist der Kreuzungspunkt der Rotationsachsen des TLS-Abtasters. Das Sensorkoordinatensystem ist ein rechtshändisches System (*x*-Achse = „Rechtswert“, *y*-Achse

= „Hochwert“, *z*-Achse = „Höhe“). Jeder Messpunkt ist von diesem Ursprung aus durch polare Messung (reflektorlose Distanzmessung bezüglich einer durch den Abtaster eingestellten Raumrichtung) entstanden. Die Punktwolke kann bezüglich der Raumrichtungen strukturiert bzw. sortiert sein gleichbedeutend mit der rasterförmige Anordnung der Messpunkt innerhalb der Punktwolke (Typ-B).

Zusätzliche Messelemente je Messpunkt können sein:

- ein Remissionswert als Maß für die zurückgestreute und am TLS empfangenen Energie des Laserlichtes (Intensität, Reflektanz).
- ein Farbwert, der aus einem zusätzlichen Sensor (Fotokamera) stammt oder durch Berechnung/Verarbeitung hervorgegangen ist. Mit dem Farbwert kann die Punktwolke z.B. zur Visualisierung „photorealistisch“ eingefärbt werden.

Mit dem OSF-Format kann nur eine einzelne Punktwolke mit seinen Metainformationen je Datei gespeichert werden. Bei den abgespeicherten Daten handelt es sich in der Regel um rohe Messwerte. Ob, welche und wie Korrekturen und Reduktionen angebracht sind, wird im OSF-Format nicht dokumentiert.

2.2 OSF-Struktur

Das OSF-Format setzt sich aus zwei Teilen zusammen. Zum einen ist dies der XML-Teil mit den Metainformationen (ASCII), der sich am Anfang der Datei befindet. Daran schließt sich der binäre Teil mit den eigentlichen Messdaten an. Der Anfang des binären Teils wird durch das Zeichen Ctrl-Z (`0x1A`) markiert.

2.2.1 Teil Metainformationen (XML-Teil)

Der Teil mit den Metainformationen stellt ein valides XML-Dokument dar. Seine Struktur ist durch eine Document-Type-Definition (`osf.dtd`) bzw. ein XML-Schema (`osf.xsd`) festgelegt (Abb. 4). Es existieren zwei Sektionen in diesem XML-Dokument. Die erste Sektion beinhaltet das Element `<metadata>` und das zweite das Element `<pointcloud>`, in dem die technischen Parameter über Umfang, Struktur und Georeferenzierung der Daten im zweiten Teil der Datei abgelegt sind.

Wichtige, aber zwingend erforderliche, Metainformationen werden im Element `<metadata>` zusammengefasst. Innerhalb dieses Elementes sind bislang folgende in der Tab. 3 aufgeführte Elemente vorgesehen. Weitere Elemente können durch das flexible XML-Konzept beliebig hinzugefügt werden. Solange diese

aber nicht in einer Formatbeschreibung erläutert sind, können sie nicht exakt vom importierenden Programm interpretiert werden.

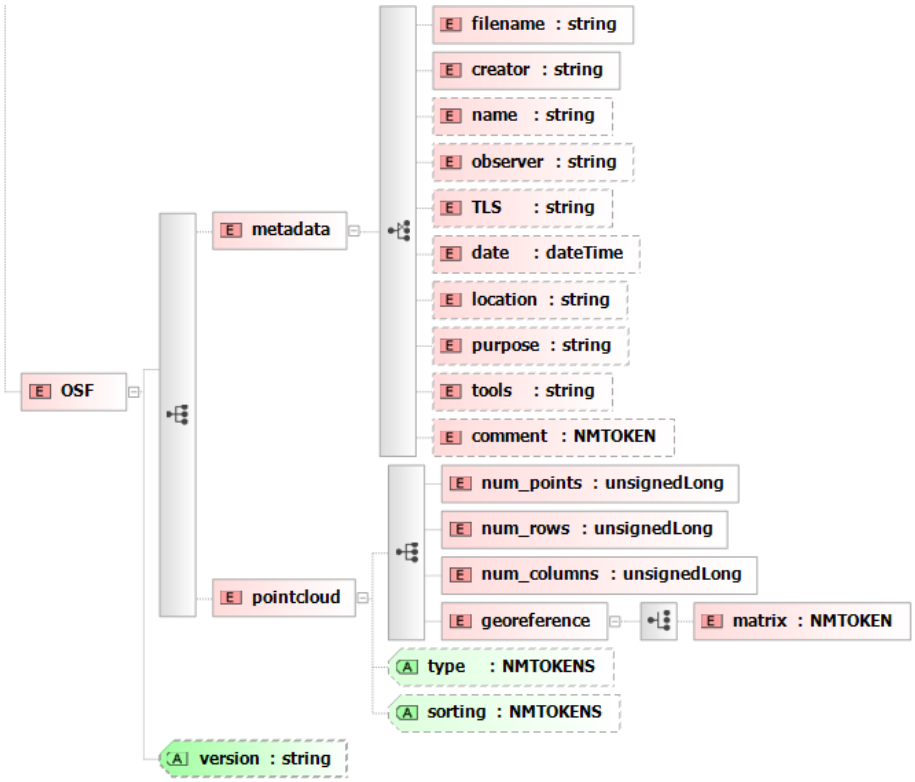


Abb. 4: XML-Schema (osf.xsd) für das OSF-Format Version 1.0.

Tabelle 3: OSF-Metainformationen

Element-name	Bedeutung und Inhalte
`<filename>`	Name der Datei im Orignalformat
`<creator>`	Information über das Programm, welches diese Datei erstellt hat.
`<name>`	Name dessen, was gespeichert ist? Z.B.: Standpunktnummer
`<TLS>`	Mit welchem Sensor sind die Daten erfasst worden oder durch welches Programm sind sie verarbeitet/erzeugt worden? Z.B.: TLS-Typ, Seriennummer, Firmware-Version etc.
`<observer>`	Wer hat gescannt oder ausgewertet? Z.B.: Name des Beobachters/Auswerters oder der Programmname
`<date>`	Wann sind die Daten erfasst oder verarbeitet worden?
`<location>`	Wo sind die Daten erfasst worden? Z.B.: Angabe einer Adresse.
`<purpose>`	Warum sind die Daten erfasst bzw. verarbeitet worden?
`<tool>`	Welche Werkzeuge/Algorithmen sind verwendet worden?
`<comment>`	Andere unstrukturierte Metainformationen oder Kommentare.

Das Element <pointcloud> hat verschiedene Attribute anhand deren man die binär gespeicherten Daten innerhalb des zweiten Teils herauslesen kann. So bestimmt das Attribut sorting, ob die Daten in einer Rasterstruktur oder in einer freien Sortierung gespeichert sind und das Attribut type spezifiziert welcher Datenumfang pro Messpunkt vorliegt. Damit ist es möglich sowohl Messungen in ihrer originären Anordnung (Typ-B) als auch völlig unstrukturierte Punktwolken (Typ-C), wie sie häufig nach Verarbeitungsprozessen entstehen, auszutauschen. Auch wird damit ein Weg eröffnet zu unterscheiden, ob

a) nur die Geometriedaten (type="xyz"),
b) Geometriedaten und ein Remissionswert (type="xyzI"),
c) Geometriedaten und RGB-Werte (type="xyzRGB") oder
d) Geometriedaten, ein Intensitätswert und RGB-Werte (type="xyzIRGB")

je Messpunkt abgespeichert sind. Durch die Variabilität, nur die Datenmengen abzuspeichern, die auch wirklich von Interesse sind, kann Speicherplatz gezielt eingespart werden.

An Unterelementen für <pointcloud> sind die Gesamtanzahl der Messpunkte <num_points>, die Anzahl an Zeilen <num_rows> und Spalten <num_columns> sowie eine Transformationsmatrix für die Georeferenzierung <georeference> vorgesehen. Die Georeferenzierung wird durch eine 4x4-Transformationsmatrix für homogene Koordinaten definiert, die wie folgt aufgebaut ist:

$$\underset{4\times 4}{\mathbf{T}} = \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & t_X \\ r_{2,1} & r_{2,2} & r_{2,3} & t_Y \\ r_{3,1} & r_{3,2} & r_{3,3} & t_Z \\ 0 & 0 & 0 & s \end{bmatrix}$$

und beinhaltet folgende Parameter:

$r_{i,j}$: Rotationsmatrix inkl. Spiegelung und Scherungen
t_i : Translationsvektor
s : Skalierungsfaktor

Um die im zweiten Dateiteil gespeicherte Punktwolke vom lokalen Sensorkoordinatensystem ins übergeordnete System zu transformieren ist die Formel 1 zu benutzen.

$$\mathbf{x}_{global} = \begin{bmatrix} X_{global} \\ Y_{global} \\ Z_{global} \end{bmatrix} = \begin{bmatrix} x_{global}/s \\ x_{global}/s \\ x_{global}/s \end{bmatrix} = \frac{1}{s}\begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_{local} \\ y_{local} \\ z_{local} \end{bmatrix} + \begin{bmatrix} t_X \\ t_Y \\ t_Z \end{bmatrix} \quad (1)$$

2.2.2 Teil Punktwolke (binäre Teil)

Die Länge des Teils Metainformationen ist momentan auf 2048 Byte beschränkt. Das Ende wird durch das Zeichen `0x1A` (Ctrl-Z) an der Dateiposition 2047 angezeigt. Danach folgen die binär abgelegten Daten der Punktwolke. Wenn eine rasterförmige Anordnung (`sorting="graticule"`) und der Datentyp `type="xyzI"` eingestellt sind, sind `<num_rows>` mal `<num_columns>` Datensätze gespeichert, mit folgender der Datenstrukur (C-Syntax):

```
// float : IEEE-Format 4Byte = 4*8 =32Bit  8 signifikante Stellen
// short : 2-byte-integer

struct xyzI {
  float x[3];         // cartesian 3D coordinates   3*4 = 12 byte
  unsigned short I;   // intensity  16 bit                2 byte
}
```

2.3 OSF-Beispiel

Der Vorteil einer XML-basierten Speicherung der Metainformation wird an folgendem Blick in eine OSF-Datei mittels ASCII-Editor sichtbar. Ohne spezielle Hilfsmittel lässt sich das Was, Wie, Womit, Wozu etc. bezüglich dieser Daten herauslesen.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/css" href="osf.css"?>
<!DOCTYPE TLS_IO SYSTEM "osf.dtd">
<OSF version="1.0">
  <metadata>
    <filename>mini.ptx</filename>
    <creator>PBC-Processor v1.0 (c) Fast-Software Ltd.</creator>
    <name>Test</name>
    <TLS>(unknown) Serial-No: 900267-007</TLS>
    <observer>Kaluschke, Alfred</observer>
    <date>2008-09-22T12:13:47</date>
    <location>Mainz, Holzstraße</location>
    <purpose>testing TLS-IO</purpose>
    <tool>export from i3mainzscene, no changes</tool>
    <comment>ptx_bpc.exe</comment>
  </metadata>
  <pointcloud type="xyzIRGB" sorting="graticule">
    <num_points>64</num_points>
    <num_rows>8</num_rows>
    <num_columns>8</num_columns>
    <georeference>
      <matrix>
        1.000000 0.000000 0.000000 0.000000
        0.000000 1.000000 0.000000 0.000000
```

```
        0.000000 0.000000 1.000000 0.000000
        0.000000 0.000000 0.000000 1.000000
      </matrix>
    </georeference>
  </pointcloud>
</OSF>
…
← pointcloud data will following here at file position 2048
```

2.4 Mögliche Erweiterungen

Mit der vorgelegten OSF-Spezifikation sind noch nicht alle wünschenswerten Merkmale eines universellen Austauschformates erreicht. In der nächsten Entwicklungsstufe sollten die Metadaten zur Georeferenzierung auch die Angaben zum gewählten Bezugssystem enthalten. Wünschenswert wäre hier eine Codierung nach dem GeoTIFF-Standard. Auch sollten Anstrengungen unternommen werden, die Pixel-Daten unter Ausnutzung der besondern Rasterstruktur der TLS-Daten zu komprimieren. Interessant sind hier insbesondere Wavelet-basierte Ansätze, wie sie bereits beim JPEG2000-Standard mit beeindruckenden Kompressionsraten im Einsatz sind. Darüber hinaus wären statistische Angaben, wie z.B. die Ausdehnung der Punktwolke (Bounding-Box), als beschreibende Elemente im OSF sowie ein „Vorschaubild“ hilfreich im schnellen Handling.

3 OSF-Implementierung

Mit der Programmbibliothek TLS_IO, bzw. einer damit erstellbaren Dynamic Link Library (DLL), liegt bereits eine erste Implementierung des OSF-Formates als Open Source für Microsoft-Windows-Betriebssysteme vor, inkl. Dokumentation und Beispielen [11]. Mit TLS_IO ist es bislang nur möglich, Punktwolken in Rasteranordnung mit dem Datentyp `xyzIRGB` zu speichern. Als Beispielanwendungen sind im TLS_IO-Paket Konverter für das PTX-Format enthalten. So lässt sich z.B. mit dem Programm `ptx_ctls.exe` eine PTX-Datei nach OSF übertragen und dabei mit Metainformationen versehen. Der Aufruf kann dabei wie folgt aussehen:

```
ptx_ctls.exe turm.ptx turm.osf –D30.09.2007 -NTurm -SFARO -OKern -PTestPrj
```

4 Schluss

Bei der Konzeption des OSF-Formates wurde auf die speziellen Gegebenheiten beim terrestrischen Laserscanning eingegangen (z.B. begrenztes Messvolumen, begrenzte Auflösung, große Datenmengen). Es deckt die Haupteinsatzbereiche der Praxis ab ohne auf jeden Spezialfall einzugehen. Es stellt einen Kompromiss dar aus Speicher- und Verarbeitungseffizienz und sollte überall leicht zu implementieren sein. Es ist Dank der verwendeten XML-Technologie leicht erweiterbar und so auch zukünftigen Anforderungen gewachsen.

Eine gute Alternative zum OSF-Format ist das PTG-Format. Es hat zwei Vorteile gegenüber dem OSF. Zum einen ist es bereits in einem der führenden TLS-Auswertepakete für den Im- und Export implementiert. Der zweite Vorteil liegt darin, dass für nicht belegte Rasterzelle kein nennenswerter Speicherplatz verbraucht wird. Über ein Belegungsfeld je Linie wird angezeigt, welche Pixel in der Zeile Messungen enthalten. Der Overhead für dieses Belegungsfeld ist nur gering; für eine vollbesetzte Matrix mit 2.500 mal 5.000 Pixel werden zusätzlich 1,5 MByte benötigt. Nachteilig am PTG-Format sind die eingeschränkten Möglichkeiten zur Speicherung zusätzlicher Metainformationen. Hier ist die Idee des XML-Header beim OSF und des E57 wesentlich flexibler und für den Anwender leichter zu editieren.

Da das PTG-Format eine Vielzahl der Anforderungen für ein offenes Austauschformat erfüllt, hat der Autor zur Unterstützung einer stärken Verbreiterung eine PTG-DLL unter GPL-Lizenz erstellt [12].

Literatur

[1] Kern, F. & Bruhn, K.-Ch. (2010): *Terrestrisches Laserscanning – Eine Quellenkritik. In: Von Handaufmass bis High-Tech III – 3D in der historischen Bauforschung.* Verlag Philipp von Zabern, Mainz – im Druck

[2] Adobe Developers Association (Hrsg.) (1992): *TIFF Revision 6.0 – Final – June 3, 1992*

[3] *GeoTIFF*: http://www.remotesensing.org/geotiff/spec/geotiffhome.html (Zugriff: 20.10.2010)

[4] ASPRS-LAS (2009): *ASPRS Lidar Exchange Format (LAS) Specification Version 1.3 – R10* - Approved by ASPRS Board 14/07/2009. www.asprs.org/society/committees/standards/lidar_exchange_format.html (Zugriff: 20.10.2010)

[5] Kern, F. & und Pospiš, M. & Prüm, O. (2009): *Das Datenaustauschformat Binary Pointcloud (BPC) für TLS-Punktwolken*. In: Luhmann, T. & Müller, Ch. (Hrsg.): Photogrammetrie, Laserscanning, Optische 3D-Messtechnik. Heidelberg, Wichmann, S. 20-30

[6] Leica Geosystems (2008): *Cyclone PTG Point Cloud Format Specification Version 1.0*. Leica Geosystems HDS LLC.

[7] ASTM International Committee E57 on 3D Imaging Systems (2010): *ASTM standard E2761* http://www.astm.org/DATABASE.CART/WORKITEMS/WK27860.htm (Zugriff: 16.10.2010)

[8] Ackley, K. (2010): l*ibE57: Software Tools for Managing E57 files*. http://www.libe57.org. (Zugriff: 16.10.2010)

[9] Zoller & Fröhlich (2010): *Z+F LaserControl SDK*. Version 7.6.0.

[10] Kern, F. & Pospiš, M. (2010): *Open Scan Format OSF Version 1.0 Documentation*. 1.04 draft. 29.03.2010 http://www.xdesy.d [Zugriff: 16.10.2010]

[11] Kern, F. (2010): *OSF-IO Documentation*. 1.01 draft. 29.03.2010 http://www.xdesy.de (Zugriff: 16.10.2010)

[12] Kern, F. (2010): *Open Source DLL für das Cyclone Point Cloud Format PTG 1.0*. www.xdesy.de (Zugriff: 16.10.2010)