Abstract

Inhaltsverzeichnis

1 Verwendung und Umfeld des JHelioviewers

Wie der Name bereits verrät ist JHelioviewer eine Applikation, die zur Analyse von Sonnendaten verwendet wird. Es wird international zur Sonnenforschung eingesetzt und wird von der FHNW zusammen mit der ESA entwickelt. Momentan ist eine neue Version des JHelioviewer in Entwicklung, welche die Sonne im dreidimensionalen Raum darstellt. Ein Feature von JHelioviewer ist die Magnetfeldlinien darzustellen und zu animieren, die Abbildung 1 zeigt die Visualisierung. Es wird zwischen drei Feldlinien Unterschieden: Linien, die auf der

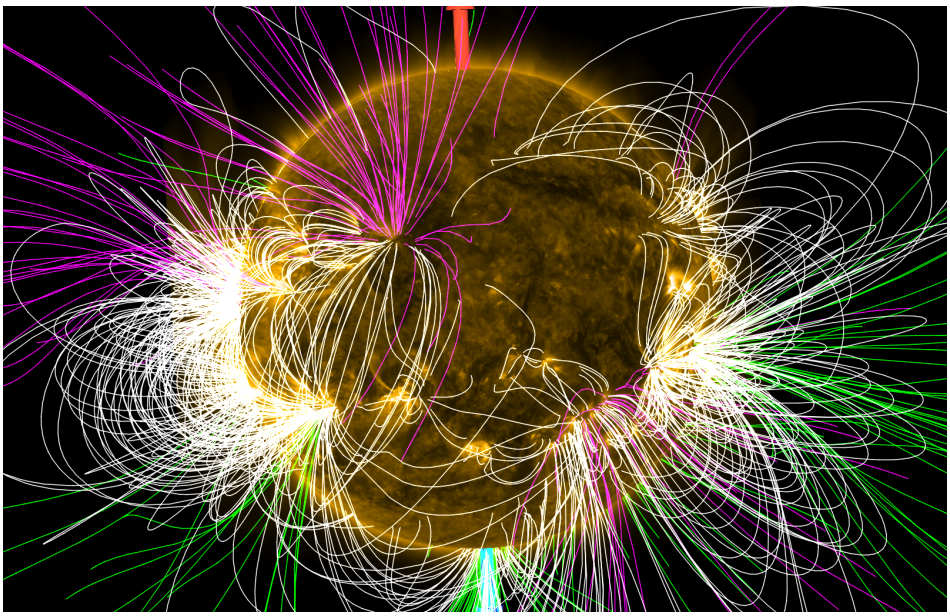

Abbildung 1: Visualisierung der Feldlinien im JHelioviewer

Sonne starten und wieder auf der Sonne landen, auf der Sonne starten und ins Weltall führen oder vom Weltall auf der Sonne landen. Die weissen Feldlinien repräsentieren ”Sonne zu Sonne´´, die Grünen ”Sonne zu Weltall´´ und die Violetten ”Weltall zu Sonne´´. Die Feldlinien sind, allgemein Betrachtet, eine grosse Menge an Punkten, welche ein Server bereitstellt. Die Abbildung 2 visualisiert den Datenfluss. In regelmässigen

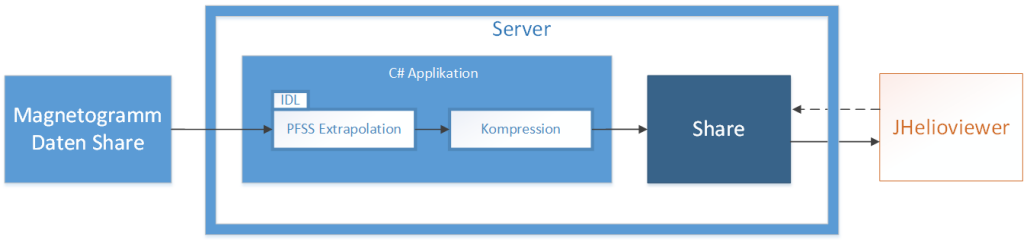

Abbildung 2: Aufbau und Datenfluss des Servers

Abständen sucht der Server nach neuen Oberflächen-Magnetogramm-Daten der Satelliten. Alle sechs Stunden wird die Oberfläche der Sonne neu gemessen.Daraus werden mittels Potential Field Source Surface (PFSS) Extrapolation die Feldlinien zu diesem Zeitpunkt errechnet. Danach führt der Server eine verlustlose Kompression durch und stellt die Daten auf einem öffentlichen Share dem JHelioviewer zur Verfügung. Der JHelioviewer lädt dann zur Laufzeit die Feldlinien, die er benötigt.

Jede halbe Stunde schiesst ein Satellit ein Bild der Sonne. Wenn der JHelioviewer also einen Zeitraum animieren soll, existiert für jede halbe Stunde ein Bild der Sonne, eine Aufnahme des Magnetfelds existiert aber nur alle sechs Stunden eine Simulation. Bei einer Animation gibt es also nur für jedes zwölfte Bild neue Feldlinien.

Durch den grossen Zeitabstand und durch die Eigenschaften der PFSS Extrapolation unterscheiden sich die Feldlinien stark von Simulation zu Simulation. Das führt zu einem schlagartigen Wechsel in der Animation. In der Zukunft soll dieser schlagartige Wechsel behoben werden, indem beispielsweise die PFSS Extrapolation in kürzeren Intervallen berechnet wird, oder durch Interpolation von einer Feldlinien-Simulation zur Anderen.

1.1 Rohdatenmenge der Feldlinien

Die PFSS Extrapolation produziert für eine Aufnahme etwa 15 MiBytes an Daten. Wenn man annimmt, dass für jedes zwölfte Bild eine Aufnahme existiert und der JHelioviewer bei 60 Bilder in der Sekunde animiert, muss eine Aufnahme in etwa 0.2 Sekunden heruntergeladen und angezeigt werden. Wenn man noch beachtet, dass der Server vermutlich nicht lokal, sodern nur über eine Internetverbindung erreichbar ist, ist eine Animation mit dieser Datenmenge nicht möglich. Deshalb wurde bereits die Datenmenge verkleinert auf etwa 1.5 MiBytes pro Aufnahme. Aber auch das ist noch zu gross. Der JHelioviewer müsste eine konstante Geschwindigkeit um die 60 MiBit haben. Ziel dieser Arbeit ist deshalb eine Kompression zu entwickeln, welche eine flüssige Animation erlaubt.

1.2 Komprimierung mittels verlustloser und verlustbehafteten Ansätzen

Verlustlos:Kein Datenverlust, aber nur begrenzte komprimierung (shannons source code theorem?) Datenmenge bereits im Vorfeld mit verlustbehafteten und verlustlosen Ansätzen verringert. mehr verlustbehaftet aber in guter Qualität

2 State of the Art

2.1 3d Mesh Kompression

In der Computergrafik müssen alle Gegenstände und Akteure in der virtuellen Welt modelliert werden. So wird ein Mensch eine Menge von Dreiecken, das Mesh, modelliert. Jeder Punkt eines Dreieck kann zusätzliche Informationen gespeichert haben wie Texturkoordinaten, Farbe oder Oberflächen-Normalen. Je nach Anwendungsfall kann ein Modell aus ein paar hundert oder mehreren hunderttausend Dreiecken bestehen. Um diese Datenmenge zu verkleinern versuchen Formate wie OpenCTM [2] die Datenmenge verlustfrei oder verlustbehaftet zu komprimieren.

wie funktioniert es

wie es verwendet werden könnte

2.2 PointCloud Kompression

Industrie Lasersampling Bild pointcloud Grosse Punktmenge, welche im 3d Raum komprimiert werden soll. verlustfrei/ Verlustbehaftet Je nach Implementation können zu jedem Punkt zusatzinfos gespeichert werden wie Farbe/Normalen etc, darüber könnte die Information, zu welcher Linie ein Punkt gehört, gespeichert werden.

2.3 Signal Approximation

Messtechnik von Medizin bis Fotografie, überall wo man ein Signal über Zeit hat Versucht ein Signal durch eine Folge von Funktionen zu approximieren.

Gute Approximation kommt mit wenigen Funktionen aus. Verlustbehaftete Kompression, indem man mit einer begrenzten Anzahl

fourier, wavelet Compressed sensing, Spline Approximation

2.4 Entropie Kodierung

Verlustfreie Komprimierung basierend auf Shannons coding theorem

Arithmeic, Huffman Dictionary Type: LZ77, Byte paoir encoding, RLE

3 Eigenschaften und Kompression der Feldlinien

Jede Datei beinhaltet 1200 Feldlinien und insgesamt etwa 60'000 Punkte. Die PFSS Extrapolation rechnet alle Punkte im sphärischen Koordinatensystem, mit Längengrad ϕ,Breitengrad θ und dem Radius. Der Wertebereich des Radius geht von 0 bis 4. Die Einheit ist der Sonnenradius, heisst $R = 1 \Rightarrow 695'800km$. Die Winkel ϕ und θ sind in Radians angegeben, 0 bis 2π respektive bis π für θ. Das sphärische Koordinatensystem hat den Effekt, dass Punkte genauer dargestellt werden, wenn sie näher an der Sonnenoberfläche sind.

Umrechnung von sphärisch auf euler Fits format [**?**]

3.1 Ist-Komprimierung

Der JHelioviewer bietet an, die Feldlinien zu einem gegebenen Zeitpunkt darzustellen. Damit der Benutzer eine vernünftige Zeit auf die Feldlinien wartet, wurde bereits im Vorfeld eine Kompression implementiert. Zuerst werden die Daten im sphärischen Koordinatensystem auf dem Server quantisiert und mit GZip verlustfrei komprimiert. Der JHelioviewer Dekomprimiert die Daten, transformiert sie in das Eulerische Koordinatensystem um. Die Punktmenge währe für schwächere Grafikkarten zu gross, weshalb der JHelioviewer eine weitere Quantisierung durchführt.

Quantisierung und Dateiformat auf dem Server
Zuerst werden die Kanäle R,ϕ und θ Kanäle zu shorts diskretisiert:

1. R: 4 = 2^{15}.
2. ϕ: 2π = 2^{15}
3. θ: 2π = 2^{15}

θ Wertebereich geht aber nur von 0 bis π, die letzten Bits werden gar nicht verwendet. Die Kanäle R und ϕ haben das Problem, dass der Wert 2^{15} einen Signed Integer Overflow verursacht und auf -2^{15} zu liegen kommt. R scheint den maximalen Wert nie zu erreichen. Wenn aber eine Feldlinie durch den Längengrad Nullpunkt geht, springt der Kanal von $2^{15} - 1$ auf -2^{15} und dann auf 0.

Subsampling, jeder vierte Punkt 0 Löschen.

Format: zuerst Konstanten, alle Radien

Quantisierung des JHelioviewers
Clientseitig wieder ein subsampling und umrechnung in eulerische xyz

3.2 Lösung 0, Clientseitiges Subsampling auf dem Server ausführen

Subsampling 5Grad auf dem Server Minimale Lösung, ziel ist es diese Lösungen zu schlagen.

3.3 Lösung 1, Diskrete Kosinus Transformation

DCT, da alles nahe an harmonischen Halbwellen

subsampling? koordinatentransformation –¿ kein wrap around, Diskretisierung? Cosinus-Transformation

DCT 2 idct ist dct3

Quantisierung

speicherung für Entropie encoding, alles was ähnlich ist zusammen.

encoding-¿ rar

4 Qualitätsmessung der Kompression

Um die Datenmenge der Feldlinien zu verringern werden verlustbehaftete Kompressionsverfahren angewendet. Trotz des Dateverlustes sollen die dekomprimierten Linien möglichst ihren Originalen ähneln. Kleine Abweichungen werden von den Sonnenforschern toleriert. Ihnen ist wichtig, dass vorallem die Form der Kurve möglichst erhalten bleibt. Grosse Abweichungen, die selten eintreten, können aber das Aussehen einer Kurve grundlegend verändern.

Zusammen mit den Sonnenforschern wurden zwei Fehlermasse bestimmt: Der absolute maximale Fehler und die Standardabweichung von der komprimierten Linie zum Original. formel der Standardabweichung, grosse Abweichungen werden stärker gewichtet.
Der absolute maximale Fehler wird noch als Absicherung gemessen. In den meisten Fällen wird die Kompression mit der tieferen Standardabweichung auch den kleineren maximalen Fehler haben. Da aber die Messung über ein paar hunderttausend Punkte durchgeführt wird, ist das Gegenteil denkbar. Eine gute Kompression muss es einen minimalen absoluten Fehler haben und eine minimale Standardabweichung bei minimalem Platzbedarf.

Eine feste Grenze für die Genauigkeit ist leider nicht objektiv festzulegen. Auch wenn eine Grenze gefunden wird, kann diese sich in der Zeit verändern. Im Fall der Feldlinien ist die Internetverbindung der Flaschenhals. Es kann sein, dass in Zukunft mehr Präzision bei mehr Platzbedarf verlangt wird. Deshalb werden die Verfahren, wenn möglich, mit unterschiedlichen Qualitätsstufen getestet und verglichen. Die Abbildung 3 zeigt ein

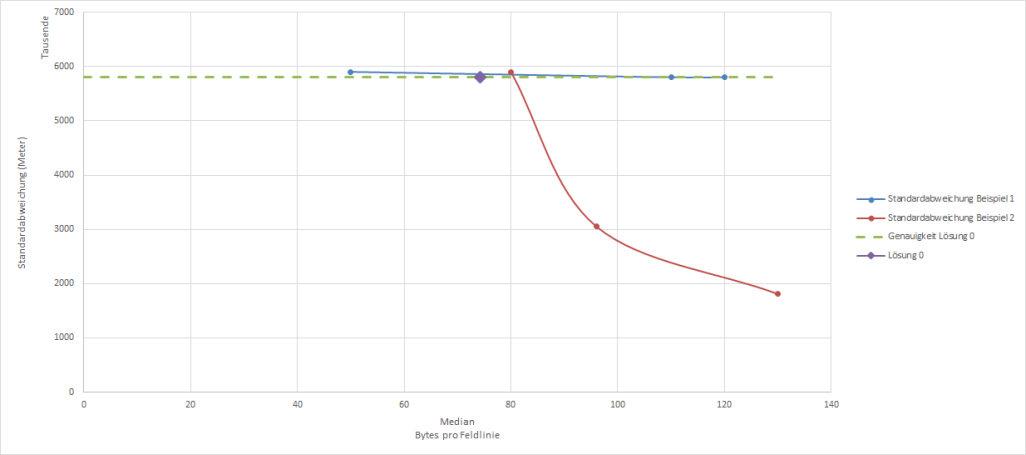

Abbildung 3: Beispiel eines Vergleichsdiagramms. Auf der X-Achse ist die durchschnittliche Anzahl Bytes pro Feldlinie dargestellt. Die Y-Achse zeigt die Standardabweichung in Meter an. Die rote und blaue Kurven sind Beispieldaten

Vergleichsiagramm, welches im Abschnitt 6 verwendet wird. Es werden die Standardabweichungen von zwei Beispielansätzen mit der Lösung 0 verglichen. Der Violette Punkt stellt die Lösung 0 dar, der grüne Strich ist eine hilfslinie und zeigt die Genauigkeit der Lösung 0 an. Damit eine Kompression besser ist, muss sie links und unterhalb des Violetten Punktes hingelangen.

4.1 Auswahl und Erhebung der Testdaten

Die Testdaten sollen zu einem alle Randfälle abdecken, als auch durchschnittliche Fälle enthalten. Aus diesem Grund wurden insgesamt zehn Datensätze ausgewählt: Vier Datensätze mit hoher Sonnenaktivität, zwei mit wenig und vier zufällig. Für die vier Datensätzen mit hoher Aktivität wurde in den Jahren 2014 und 2013 nach

den grössten Solare Flares gesucht. Für die Datensätze mit wenig Aktivität wurde das Gegenteil gemacht, nach Zeiträumen mit möglichst kleinen Solar Flares gesucht.
Die feldlinien werden aber nur alle sechs Stunden berechnet und Solar Flares sind sehr spontane Ereignisse. Auch eine grosse Flare kann während den sechs Stunden angefangen und wieder aufgehört haben. Für die grossen Solar Flares wurde deshalb beachtet, dass die Datensätze vor dem Ereignis verwendet wurden. Grosse Solar Flares entladen das Feld, vor dem Ereignis ist das Magnetfeld komplexer.

Wie im Abschnitt 3.1 beschrieben, führt der IDL-Code schon eine Quantisierung durch. Diese wurde entfern und deshalb wurde der IDL Code angepasst. Die Quantisierung wurde für die Testdaten entfernt. Die Punkte werden nun mit 32 Bit Floating-Point Genauigkeit gespeichert.

4.2 Ablauf des Tests und Messung des Fehlers

Float daten werden geladen. Daten kopiert und Kompression/Dekompression durchgeführt für alle Testdaten. Die kopierten Daten wissen aber noch, welches ihr Originalpunkt ist. Zwei Mengen, Originalpunkte O, dekomprimierte Punkte D. Es gibt immer gleich viele oder mehr Originalpunkte wie dekomprimierte Punkte. Die Fehlerberechnung muss der allgemeine Fall und die Randbehandlung unterschieden werden.

4.2.1 Allgemeiner Fall

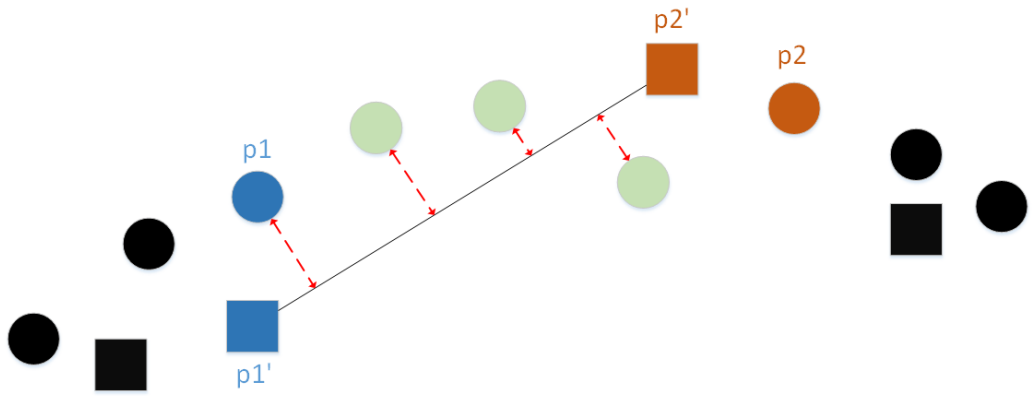

Abbildung 4: Darstellung der Fehlerberechnung. Die Punkte sind die Originaldaten, die Quadrate sind die Punkte nach der Kompression.

Die Berechnung ist Dargestellt im Diagramm 4. Für jeden Punkt $p1'$ aus D, nehme $p1'$ und den folgenden Punkt und $p2'$. Ziehe eine Strecke s durch $p1'$ und $p2'$. Suche von $p1'$ den Originalpunkt $p1$ aus O und rechne den Abstand aus zur Strecke s. Führe das für alle folgenden Originalpunkte durch, bis $p2$ erreicht wurde. Der Abstand s zu $p2$ wird nicht mehr berechnet.

Abstandsberechnung eines Punktes zu einer Strecke
Gegeben: Stecke s mit Eckpunkten A und B und Punkt P.
Gesucht: Kürzeste Distanz zwischen s und P

Zuerst wird überprüft, ob eine Senkrechte durch P überhaupt auf der Strecke s zu liegen kommt. Das ist der Fall, wenn die Strecke AP auf die Strecke s projizierbar ist:

$$t = \frac{\vec{AB} * \vec{AP}}{|\vec{AB}|^2}$$
$$0 \leq t \leq 1$$

Wenn das nicht möglich ist, wird der kürzere Distanz von P zu einem der Eckpunkte genommen. Falls aber eine Senkrechte auf s zu liegen kommt, muss jetzt die Länge der Senkrechte berechnet werden. Aus der vorhgehenden Berechnung könnte man den Fusspunkt auf s berechnen und dadurch die Distanz, oder man kann die Distanz direkt über das Kreuzprodukt berechnen.

$$distance = \frac{|\vec{BA} \times \vec{BP}|}{|\vec{BP}|}$$

4.2.2 Randbehandlung

Es ist möglich, dass die originalen Endpunkte durch eine Quantisierung verworfen wurden. Das bedeutet, wenn man den Fehler für den allgemeinen Fall berechnet, am Anfang und am Ende Originalpunkte existieren, für die nie eine Distanz berechnet wurde. Deshalb müssen die Ränder speziell behandelt werden.
Anfangsrand
endrand

4.2.3 Berechnung der Standardabweichung

5 Implementation

5.1 Komprimierung auf dem Server

C# architektur des servers, nicht viel. neuer Ordner auf Share

5.2 Asynchrone Dekomprimierung im JHelioviewer

alter zustand?

neuer Zustand asynchrone architektur Klassendiagramm preload (precache)

6 Resultate

6.1 Lösung 0, Clientseitiges Subsampling auf dem Server ausführen

5grad

6.1.1 Artefakte

6.2 Lösung 1, Diskrete Kosinus Transformation

Die Lösung 0 erreicht bereits tiefe Kompressionen.

6.2.1 Proof of Concept

Es gibt unterschiedlichste Wege eine Diskrete Kosinus Transformation zu implementieren. Im Rahmen von diesem Projekt ist es leider nicht möglich alle Wege zu erforschen. Es wird deshalb zuerst mit einem vereinfachten Test ein möglichst vielversprechender Ansatz gesucht, welcher dann im zweiten Schritt komplett ausgearbeitet wird. Mit dem vereinfachten Test sollen folgende Fragen beantwortet werden:

- Welches Koordinatensystem soll verwendet werden?
- Was für einen Einfluss hat eine Verschiebung der Koordinatenachsen?
- Was für einen Einfluss hat die Transformation der Ableitung?

Dazu wird nur eine einzige Simulation der Feldlinien verwendet, welche bereits Quantisiert ist. Gemessen wird die Standardabweichung des Radius, Phi und Theta Kanals respektive der X,Y,Z Kanäle. Für den Test wird jeweils eine einfache Quantisierung durchgeführt: Ab dem 128 Parameter wird alles auf 0 gesetzt.

Einfluss der Korrdinatenverschiebung
Der JPEG Standard benutzt ebenfalls eine Diskrete Kosinus Transformation. Vor der Transformation wird aber der Wertebereich um Null zentriert: Wenn die Eingabepixel einen Wertebereich von 0 bis 255 haben, wird dieser -128 bis 127 verschoben [**?**]. Diese Verschiebung soll den ersten Parameter, den DC Koeffizienten [**?**] dämpfen.

Im Abschnitt 3.1 wurde bereits der Wertebereich von den Kanälen Radius, Phi und Theta beschrieben. Es ist nochmals darauf hinzuweisen, dass bei der Ist-Quantisierung der Höchstwert auf 2^{15} fällt, was beim Signed 16 Bit Integer zu einem Overflow führt, sprich zum Wert -2^{15}. Wenn also eine Feldlinie den Längengrad Nullpunkt überschreitet, wird Phi zuerst den Wert $2^{15}-1$ annehmen, dann auf -2^{15} springen und wieder zurück auf 0. Diese Sprünge können sich negativ auswirken. Solche Sprünge können nur mit hochfrequente Anteilen dargestellt werden und sind dadurch schlechter zu komprimieren. Durch eine Verschiebung würde der Phi-Kanal zwar immernoch sprünge beinhalten, wenn der Nullpunkt überschritten wird. Jedoch nur noch von $2^{14}-1$ auf -2^{14}. In der Abbildung 5 ist deutlich zu sehen, dass der DCT Koeffizient im Vergleich gedämpft wurde. Das bedeutet, dass bei einer späteren Quantisierung der DCT Koeffizient mit weniger Präzision abgespeichert werden kann, ohne einen hohen Fehler zu verursachen.
Die Tabelle 1 zeigt deutlich, dass dank der Koordinatenverschiebung der Phi-Kanal etwa vier Mal genauer dargestellt werden kann. Vermutlich ist es den verkleinerten Sprüngen zu verdanken, wenn eine Kurve den Nullpunkt überschreitet.

Einfluss der Ableitung

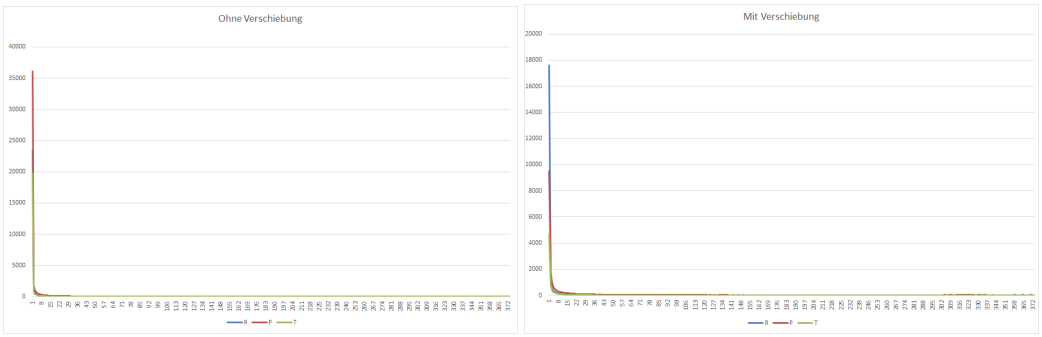

Abbildung 5: Absoluter Median des Resultat der Kosinustransformation. Links ist das Resultat der sphärischen Koordinaten. Rechts das Resultat der um den Nullpunkt zentrierten Koordinaten.

Kanäle	Std. Abw. Normal	Std. Abw. zentrierten Koordinaten
Radius	124.397	124.316
Phi	1236.350	256.565
Theta	83.723	83.527

Tabelle 1: Vergleich der Standardabweichung pro Kanal. Die Werte sind im diskretisierten sphärischen Koordinatensystem

Warum Ableitung: kleinere parameter, Resultat: Dämpfung der Parameter, aber Keine bessere Genauigkeit.

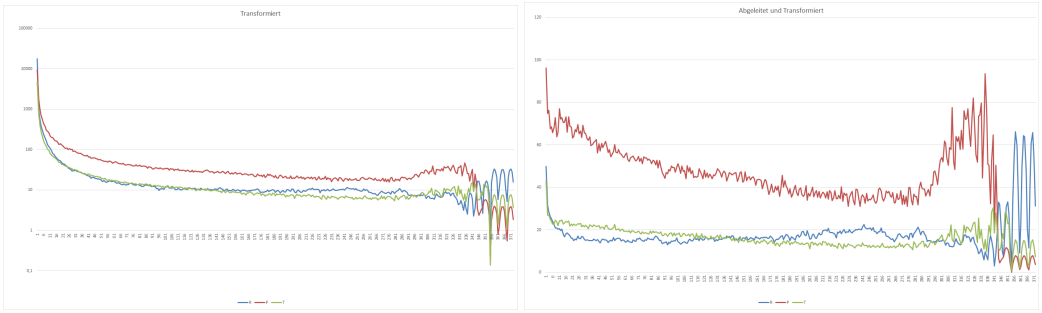

Abbildung 6: Absoluter Median des Resultat der Kosinustransformation. Links ist das Resultat der um den Nullpunkt zentrierten Koordinaten auf einer logarithmischen Skala. Rechts das Resultat der Ableitung auf einer linearen Skala.

Kanäle	Std. Abw. zentrierten Koordinaten	Std. Abw. der Ableitung
Radius	124.316	124.135
Phi	256.565	256.084
Theta	83.527	83.723

Tabelle 2: Vergleich der Standardabweichung pro Kanal. Die Werte sind im diskretisierten sphärischen Koordinatensystem

Aber sachverhalt repeat:

repeat wirkt sich positiv auf die Genauigkeit aus. Es besteht die Möglichkeit die DCT zu approximieren mit anderen Funktionen?

Auswahl des Koordinatensystems

Kanäle	Std. Abw. der Ableitung	Std. Abw. bei Parameterwiederholung
Radius	124.135	94.879
Phi	256.0845	228.332
Theta	83.723	67.667

Tabelle 3: Vergleich der Standardabweichung pro Kanal. Die Werte sind im diskretisierten sphärischen Koordinatensystem

Warum Sphärisch: Kleinere Koeffizienten. Warum Euklidisch: Kein Wrap-around. Messung: Nach der Dekompresssion rücktransformation auf euklidische Koordinaten und vergleichen.

Euklidisch Quantisierung auf shorts, ergebnis: Tabelle

Kanäle	Std. Abw. sphärische Koordinaten	Std. Abw. euklidische Koordinaten
X (Meter)	43551184	27087946
Y (Meter)	19656055	17154731
Z (Meter)	40825926	26242294

Tabelle 4: Vergleich der Standardabweichung pro Kanal.

6.2.2 Implementation

Residuals + Speicherung Subsampling da Punktmenge zu gross Erster Versuch speicherung:

Fehler mit Diskretisierung

Kann das Client Subsapmling verwendet werden 5grad währe schön, viel kleinere Punktmenge, aber nicht konstante Abstände

6.2.3 Artefakte

2D Feldlinie zeigen, normal und komprimiert

7 Diskussion

8 Fazit

Literatur

[1] Nobody Jr. My article, 2006.

[2] Marcus Geelnard. Openctm the open compressed triangle mesh file format, November 2014.

Abbildungsverzeichnis

Tabellenverzeichnis

9 Anhang

subsectionInstallationsanleitung

10 Ehrlichkeitserklärung