HAW HAMBURG

Eduard Strehlau
Matrikel-Nr. 2511393

Raytracing impliziter Flächen mit Distanzfunktionen

WP Computergrafik (Prof. Jenke)

Ausarbeitung Referat

Department Informatik

1 Einleitung

Raytracing bietet im Vergleich zu Rasterung die Möglichkeit viele in der Natur vorkommende Phänomene abzubilden.

Dabei setzt klassisches Raytracing vor allem auf Szenen aus heterogenen Objekten zu denen jeweils analytische Strahl Intersektionsfunktionen bekannt sind.

Spheretracing hingegen, erstmals von Hart et al. [HSK89] eingeführt um komplexe Geometrische Strukturen wie etwa Fraktale zu rendern, baut auf einer homogenen Szenenbeschreibung durch eine Distanzfunktion auf.

Eine solche Szenenbeschreibung erlaubt es Constructive Solid Geometry, im folgenden CSG, sowie weitere interessante Modellierungsfunktionen durch simple mathematische Funktionen ohne Fallunterscheidungen auszudrücken [Har96].

Der von Hart vorgeschlagene Algorithmus [Har96] ist leicht umzusetzen, er ruft Iterativ die Distanzfunktion auf, zeigt aber dennoch, da die Distanzfunktion gleichzeitig Informationen liefert welche man klassischerweise nur durch Raum/Objekt Optimierungsstrukturen bekommt, eine effiziente Laufzeit im Bezug zur geometrischen Komplexität der Szene. Die Ausführungskosten der Distanzfunktion sind hier in den meisten Fällen Bottleneck, nicht Komplexität der impliziten Flächen die sie beschreibt.

Diese Eigenschaften lassen eine effiziente und simple Implementierung auf heutigen GPUs zu und ermöglichen die Darstellung von komplexeren Szenen in Echtzeit, sofern diese durch eine vergleichsweise günstige Distanzfunktion beschrieben werden können. Im letzten Jahrzehnt ist Spheretracing so vor allem durch die Demoszene [Qui08] popularisiert worden und hat eine neue Welle an Forschungsarbeiten eingeleitet.

2 Grundlagen

Spheretracing basiert auf der Annahme, dass wir eine Signed Distance Funktion, im folgenden SDF, für unsere zu renderende Szene besitzen.

Die SDF nimmt als Parameter einen Punkt im Raum und gibt uns als Rückgabewert die Distanz zur nächsten Oberfläche, positiv im Falle dass wir außerhalb eines Körpers sind, negative falls wir uns in einem Körper befinden.

Eine SDF beschreibt also implizit die Oberfläche einer oder mehrere Körper bei denen $SDF(p) = 0$ ist.

SDFs für primitive Objekte wurden von Hart et. al. [Har96] abgeleitet, auch wurden einige Operationen wie Vereinigung zweier SDF definiert. Wir greifen hier vor allem auf den Funktionskatalog von [Quia] zu da diese bereits in GLSL, unserer Implementationssprache, umgesetzt sind.

Der große Unterschied von SDFs, im Vergleich zu einer Menge von analytischen Körpern, zur Modellierung einer Szene, ist, dass wir Operationen auf den SDFs einheitlich definiert haben. So lässt sich z.B. einmal eine Smooth Minimum [Har96] Operation ableiten und diese automatisch auf alle Primitiven und deren Kombinationen anwenden. Genauso erlauben uns Domänentransformationen den Raum selbst zu verändern und so mit geringem Mehraufwand die dargestellte Szene um massive Komplexität zu erweitern. Zum Beispiel erlaubt uns der Modulo Operator anstelle ein Objekt nur einmal in einer SDF darzustellen, dieses Objekt unendlich oft zu zeigen [Quia].

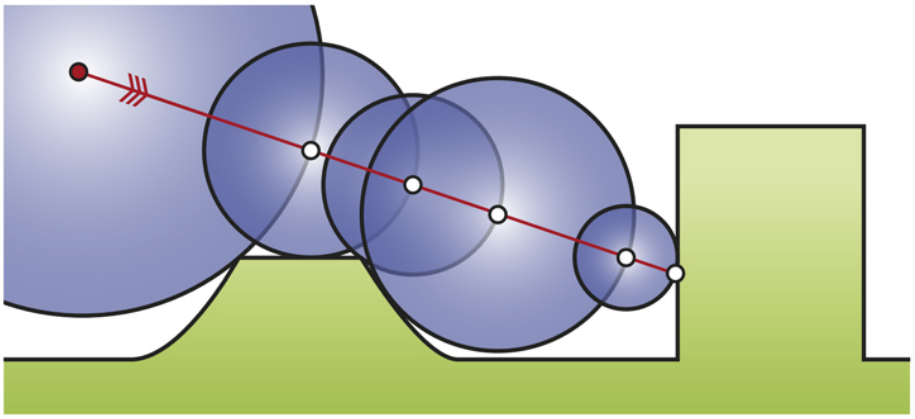

Abbildung 1: Eine Visualisierung von Spheretracing. Die genommenen Schritte entsprechenden den weißen Kreisen. Die SDF an den Schritten entspricht dem Radius der Kreise, aus [Kei+14]

Es lässt sich leicht erkennen wie man mit Hilfe einer SDF einen Raytracer umsetzten kann. Eine SDF gibt per Definition an wie lange wir entlang eines Strahls reisen können, ohne auf Geometrie zu stoßen. Wir müssen also lediglich für jeden Ursprungsstrahl die SDF aufrufen, diesen Wert entlang des Strahls reisen und das ganze wiederholen bis die SDF hinreichend nahe an 0 ist, wir uns also hinreichend nah an einer Oberfläche befinden. Diese leicht intuitiv zu verstehende Eigenschaft wird von Hart et. al. [Har96] mit Rigor belegt.

Die Anzahl der benötigten Schritte pro Stahl verhält sich hier abhängig zu der Nähe des Strahls von der Geometrie, enges Vorbeifliegen an Flächen benötigt viele Schritte. Eine Ebene an der der Strahl parallel, in geringer Distanz vorbei fliegt, ist hier der degenerative Fall, welcher sich nicht von klassischen Raymarching mit einem fixen Offset unterscheidet. Von [BV18] sind für diese Fälle Optimierungsmaßnahmen vorgeschlagen, welche aber für die Ausarbeitung nicht umgesetzt sind.

Eine Besonderheit die sich durch einen so implementierten Raytracer ergibt, ist, dass weiche Schatten einfach performant umzusetzen sind [Quid]. Wir kennen an jedem Iterationsschritt unseres Raytracers die Distanz zum nächsten Objekt sowie die Distanz entlang des Strahls, auch wissen wir das wir zwischen den Schritten nicht näher an einem Objekt gewesen sein können als bei den Schritten. Aus diesen Informationen lässt sich die stärke einer Penumbra berechnen. Um weiche Schatten umzusetzen müssen wir also lediglich aus diesen Informationen bei jedem Schritt eine Penumbrastärke berechnen und das Minimum, also die dunkelste aller dieser möglichen Penumbra, protokollieren.

Dabei gilt zu beachten, dass SDF Funktionen eine Untermenge von Signed Distance Bounds, im folgenden SDB, sind. SDBs geben für einen Punkt nicht die exakte Distanz zu einer Oberfläche an, sondern unterschätzen diese stets. Für Punkte die auf der Oberfläche liegen evaluieren diese auch zu 0 wie eine SDF.

Weiche Schatten können wenn SDBs anstelle von SDFs eingesetzt werden, nicht korrekt realisiert werden. Je nachdem wie weit die SDBs von den SDFs abweichen stellt dies in der Praxis aber kein Problem dar.

Raytracing im Allgemeinen ist ein sehr gut parallelisierbares Problem, alle Ursprungsstrahlen sind unabhängig von einander, der Algorithmus kann also parallel für alle ausgeführt werden.

3 Anforderungen und Konzept

Aus den vorgegebenen Anforderungen:

- Primärstrahlen (vom Augpunkt durch die Bildebene in Szene)
- Objektschnitt
- Reflektierte Strahlen
- Beleuchtungsrechnung
- Schattenstrahlen

Kommen unter Anbetracht der besonderen Eigenschaften eines Spheretracers noch folgende selbst gestellte Anforderungen hinzu:

- Rendering in Echtzeit $>$ 30Hz.
- Weiche Schatten.
- Darstellung hoch detaillierte Geometrie welche sich nicht leicht analytisch beschreiben lässt (z.B. Fraktale).
- Raum Deformierungseffekte.

4 Umsetzung

Die Repository der Umsetzung findet sich unter `https://git.haw-hamburg.de/acs552/spheretracer/`

Es wurde mehrfach demonstriert, dass auch die Darstellung komplexerer Szenen über Spheretracing als ein Fragment-Shader Programm in Echtzeit möglich ist [Quie].

Da Raytracing allgemein äußerst gut parallelisierbar ist und Spheretracing speziell aufgrund der geringen Datenabhängigkeiten und wenig Kontrollfluss, ist anzunehmen, dass diese Implementierungen nah an dem theoretischen Maximum der Hardware in reinen FLOPs liegen. Damit würden sich sich die Geschwindigkeit einer CPU gegenüber einer GPU Implementierung vermutlich in einer Größenordnung bewegen [Sun+20].

Daher wurde, um den Echtzeitanforderungen gerecht zu werden, die Implementation Hauptsächlich als ein GLSL Fragment-Shader Programm vollzogen.

4.1 Architektur

Der Raytracing Algorithmus ist als das Fragment-Shader Programm, sphereTracer.frag umgesetzt, zu finden unter:

`src/main/resources/referat`

Das JMonkey Projekt aus der Vorgabe wird als interaktive Shell für die Argumente des Shader Programmes genutzt, dabei ist der Einstiegspunkt in der wpcg.referat.App Klasse.

Hierbei können manche der Argumente durch Slider eingestellt werden z.B. Lichtintensität, andere werden an interaktive JMonkey Kontrollen wie die FlyBy Kamera gebunden.

Für jede Frame werden die gesammelten Argument der JMonkey Shell als Uniforme für den Fragment-Shader festgelegt und der Fragment-Shader wird aufgerufen.

4.2 Szenenbeschreibung

Unsere Szenenbeschreibung muss per Definition als eine SDF erfolgen. Um eine finale Szenen SDF zu bauen, werden SDFs von primitiven Objekten mit Operationen, welche die SDF Eigenschaften erhalten, verknüpft.

Für jede der primitiven SDF und Operationen wurde ein GLSL Funktion geschrieben. Eine Szene ist repräsentiert als eine GLSL Funktion, welche diese Funktionen nutzt.

Zwischen dem Callgraph einer so beschriebenen Szene und einem Baum welcher als innere Knoten Operationen und als Blätter primitive SDF besitzt, existiert eine Bijektion.

Die Beschreibung einer Szene ließe sich also auch problemlos durch ein explizites Datenmodell vollführen. Dieses erlaubt eine Szenenbaum in einem CAD ähnlichen Ansatz zu bearbeiten, besonders die Tatsache das sich über SDF Operationen CSG ausdrücken lässt [Har96] macht diesen Ansatz attraktiv.

Nach der Bearbeitung des Szenenbaum kann dieser durch die Bijektion einfach in GLSL Code überführt werden, jeder Node wird durch einen Funktionsaufruf ersetzt, welcher die Nodes darunter als Argumente erhält.

Aufgrund von JMonkey Limitationen lassen sich Shader leider nicht dynamisch neu kompilieren, daher wurde hier nur eine implizite Szenenbeschreibung ausschließlich als GLSL Programmcode gewählt.

SDFs geben nur implizit Auskunft darüber wo im Raum sich Körper befinden, eine Unterscheidung zwischen Körpern kann über eine SDF nicht trivial erfolgen. Wir wollen in unserer Szene aber bestimmten Körpern bestimme Materialeigenschaften geben, auch wollen wir Materialeigenschaften entlang von Körpern variieren.

Die benötigten Information sind zur Konstruktionszeit der finalen SDF bekannt, diskrete Körper lassen sich z.B. durch einen Teilbaum des Szenenbaums beschreiben. Wir können also für den gleichen Szenenbaum eine alternative Algebra definieren, welche bei Anwendung nicht eine SDF zurück gibt, sondern eine Funktion welche die MaterialId zurück gibt sofern sie mit einem Oberflächenpunkt aufgerufen wird.

Da wir nicht dem expliziten Modellierungsansatz nachgehen ist dieses in unserer Implementierung ad hoc gelöst.

Die finale SDF gibt zusätzlich zu der Distanz noch eine MaterialId zurück, diese wird entweder variabel zu Koordinaten gesetzt oder bei der Union zweier SDFs auf eine fixe Id pro SDF.

4.3 LOD Operator

Spheretracer eignen sich besonders gut um hoch detaillierte Geometrie, welche sich durch simple Regeln beschreiben lässt, darzustellen.

In unserer Beispielszene haben wir einen Menger-Schwamm [Quic] mit 10 Iterationen, also etwa 3^{10} Einkerbungen entlang eines Streifen. Wollen wir diesen Würfel über die volle vertikale Auflösung von 1080 Pixel darstellen ergeben sich an sehr detaillierten Stellen in etwa 3000 Wechsel der Oberflächeneigenschaft pro Pixel.

Es zeigen sich also erhebliche Aliasing Probleme welche sich durch klassische Techniken wie Multisampling oder Mip Mapping nicht leicht lösen lassen.

Für das Antialasing von durch einer SDF beschriebenen Geometrie wurde hier ein neuartiger LOD Operator entwickelt, um hochfrequente Oberflächenmerkmale in Relation zur Oberflächendistanz zu filtern.

SDFs bieten die Möglichkeit zur uniformen Skalierung des Raumes [Har96], dabei wird nicht von einem Ursprung heraus skaliert sondern Distanz in alle Richtungen wird verlängert bzw. verkürzt. Das hat im ersten Fall die Folge das aus allen Punkten Spheren werden, für den 2. Fall ist eine intuitive Interpretation schwieriger.

Verlängern wir alle Distanzen um einen Wert, werden geometrische Details entlang der Oberfläche die näher als dieser Wert liegen geschluckt.

Wir sind also in der Lage hochfrequente Oberflächendetails zu entfernen, wir wollen aber einen Körper mit maximalen Detailgrad rendern und Details nur entfernen sofern diese eine höhere Frequenz haben als die Samplingrate des Raytracers.

Die Samplingrate des Raytracers nimmt mit steigender Distanz ab, wir müssen also in der Lage sein den von der SDF beschriebenen Raum in Abhängigkeit von der Kameradistanz uniform zu skalieren.

Bei einem naiven Ansatz, $\text{lod}_{naiv}(\mathbf{x}) = f(\mathbf{x}) - p\|\mathbf{x} - \mathbf{o}\|$, gehen die SDF Eigenschaften verloren.

Das liegt daran, dass sich die Distanzen im Raum ja in Abhängigkeit von p ändert. Der Raum wird bei Reise entlang eines Strahls kleiner, ferne Objekte rücken näher an uns heran. Wenn wir also die SDF an einem Punkt P entlang des Strahls evaluieren, kriegen wir die nächste Distanz für einen Raum in dem die Distanzen Konstant sind. Schreiten wir dann diese Distanz voran und evaluieren bei p+t ist der Raum möglicherweise so weit geschrumpft das wir jetzt mitten in einem Körper sind und nicht davor wie durch eine SDF garantiert.

Um diese Problematik zu lösen, müssen wir die maximale Rate mit der sich der Raum pro Einheit ändern kann ermitteln. Dann können wir durch diese Rate teilen und stellen sicher das die SDF immer unterschätzt.

Es folgt eine rigorose Definition des LOD Operators, so wie ein Beweis das die Anwendung auf einer SDF in einer SDB resultiert. Hierbei ist aus Zeit- und Schwierigkeitsgründen nicht belegt dass dies auch bei Anwendung auf einer SDB gilt, lässt sich aber vermuten:

Gegeben seien:

- eine SDF $f : \mathbb{R}^3 \to \mathbb{R}$
- der View Ursprung $\mathbf{o} \in \mathbb{R}^3$

- Ein LOD Faktor $p \in \mathbb{R}, p > 0$

Wir definieren den LOD Operator $\text{lod} : \mathbb{R}^3 \to \mathbb{R}$ als:

$$\text{lod}(\mathbf{x}) = \frac{1}{1+p}(f(\mathbf{x}) - p\|\mathbf{x} - \mathbf{o}\|) \tag{1}$$

Lemma 4.1. *Per SDF Definition [BVG19] gilt: Ist $f : \mathbb{R}^3 \to \mathbb{R}$ ein SDF gilt für alle $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$*

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq \|\mathbf{x} - \mathbf{y}\|$$

Lemma 4.2. *Für eine SDF $f : \mathbb{R}^3 \to \mathbb{R}$ gilt für alle $q \in \mathbb{R}$ $\mathbf{n}, \mathbf{x} \in \mathbb{R}^3$ $\|\mathbf{n}\| = 1$*

$$|f(\mathbf{x}) - f(\mathbf{x} + q\mathbf{n})| \leq |q|$$

Beweis. Es gilt für alle $q \in \mathbb{R}$ $\mathbf{n}, \mathbf{x} \in \mathbb{R}^3$ $\|\mathbf{n}\| = 1$

$$\begin{aligned} |f(\mathbf{x}) - f(\mathbf{x} + q\mathbf{n})| \leq & \|\mathbf{x} - \mathbf{x} + q\mathbf{n}\| \text{ durch 4.1} \\ \|\mathbf{x} - \mathbf{x} + q\mathbf{n}\| &= \|q\mathbf{n}\| \\ &= |q|\|\mathbf{n}\| \\ &= |q| \end{aligned}$$

□

Lemma 4.3. *[Har96] $f : \mathbb{R}^3 \to \mathbb{R}$ ist ein SDB wenn für alle alle $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$ gilt:*

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq \|\mathbf{x} - \mathbf{y}\|$$

Lemma 4.4. *Für alle $p, q \in \mathbb{R}$ $\mathbf{n}, \mathbf{x}, \mathbf{o} \in \mathbb{R}^3$ $\|\mathbf{n}\| = 1$ $p > 0$ gilt*

$$|p(\|\mathbf{x} - \mathbf{o} + q\mathbf{n}\| - \|\mathbf{x} - \mathbf{o}\|)| \leq p|q|$$

Beweis. Da $p > 0$

$$\begin{aligned} |p(\|\mathbf{x} - \mathbf{o} + q\mathbf{n}\| - \|\mathbf{x} - \mathbf{o}\|)| &= p(|\|\mathbf{x} - \mathbf{o} + q\mathbf{n}\| - \|\mathbf{x} - \mathbf{o}\||) \\ |\|\mathbf{x} - \mathbf{o} + q\mathbf{n}\| - \|\mathbf{x} - \mathbf{o}\|| \leq |q| &\implies 4.4 \end{aligned}$$

Per umgekehrter Dreiecksungleichung

$$\begin{aligned} |\|\mathbf{x} - \mathbf{o} + q\mathbf{n}\| - \|\mathbf{x} - \mathbf{o}\|| \leq \|\mathbf{x} - \mathbf{o} + q\mathbf{n} - (\mathbf{x} - \mathbf{o})\| \\ \|\mathbf{x} - \mathbf{o} + q\mathbf{n} - (\mathbf{x} - \mathbf{o})\| &= \|q\mathbf{n}\| \\ &= |q| \end{aligned}$$

□

Theorem 4.5. $\text{lod} : \mathbb{R}^3 \to \mathbb{R}$ *ist ein SDB*

Beweis. Nehmen wir eine simplere LOD Definition $g(\mathbf{x}) = f(\mathbf{x}) - p\|\mathbf{x} - \mathbf{o}\|$
Dann gilt laut Dreiecksungleichung:

$$|f(\mathbf{x}) - f(\mathbf{x}+q\mathbf{n}) + p\|\mathbf{x}-\mathbf{o}+q\mathbf{n}\| - p\|\mathbf{x}-\mathbf{o}\|| \leq |f(\mathbf{x}) - f(\mathbf{x}+q\mathbf{n})| + |p(\|\mathbf{x}-\mathbf{o}+q\mathbf{n}\| - \|\mathbf{x}-\mathbf{o}\|)|$$

Aufgrund von 4.2 wissen wir das:

$$|f(\mathbf{x}) - f(\mathbf{x}+q\mathbf{n})| + |p(\|\mathbf{x}-\mathbf{o}+q\mathbf{n}\| - \|\mathbf{x}-\mathbf{o}\|)| \leq |q| + |p(\|\mathbf{x}-\mathbf{o}+q\mathbf{n}\| - \|\mathbf{x}-\mathbf{o}\|)|$$

Per 4.4:

$$|f(\mathbf{x}) - f(\mathbf{x} + q\mathbf{n})| + |p(\|\mathbf{x} - \mathbf{o} + q\mathbf{n}\| - \|\mathbf{x} - \mathbf{o}\|)| \leq |q| + p|q| = (1 + p)|q|$$

Also:

$$|\frac{1}{1+p}(f(\mathbf{x}) - p\|\mathbf{x} - \mathbf{o}\|) - \frac{1}{1+p}(f(\mathbf{x} + q\mathbf{n}) - p\|\mathbf{x} - \mathbf{o} + q\mathbf{n}\|| \leq |q|$$

$$|\mathrm{lod}(\mathbf{x}) - \mathrm{lod}(\mathbf{x} + q\mathbf{n})| \leq |q|$$

Per 4.3 handelt es sich um eine SDB. □

5 Evaluation

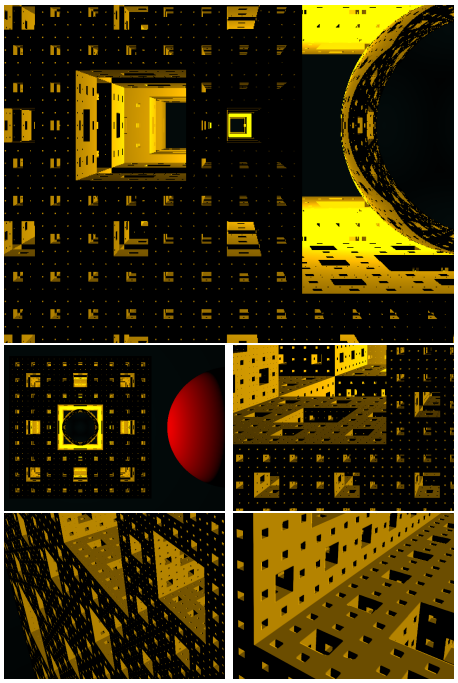

Abbildung 2: Mit LOD

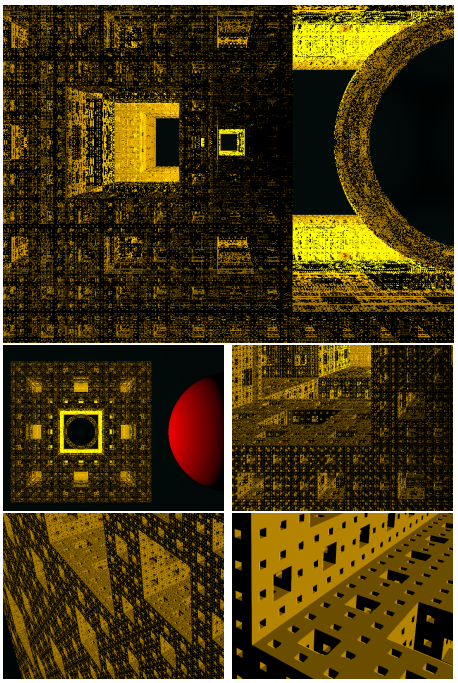

Abbildung 3: Ohne LOD

Wir haben alle Anforderungen umgesetzt, dabei sind Vor- und Nachteile von Spheretracern und dem gewählten Ansatz sichtbar geworden.

Der simple Raytracing Algorithmus, sowie die einfache Szenenbeschreibung ohne Datenstrukturen, haben sehr gut auf die restriktive GLSL Umgebung gepasst. Was sicher einer der Gründe ist warum diese Art des Raytracing in der Demoszene auf Resonanz stößt.

Die Implementierung von Effekten welche Rekursion benötigen, z.B. Spiegelungen, haben sich schwieriger gestaltet. GLSL erlaubt hier keinerlei Rekursion, rekursive Algorithmen müssen also in eine endrekursive Form umgewandelt und durch Schleifen ersetzt werden. Die manuelle Umformung von Algorithmen welche sich auf einen Callstack stützen ist nicht trivial und der Hauptgrund warum Refraktion, welche 2 Strahlen pro Refraktion benötigt, nicht umgesetzt wurde.

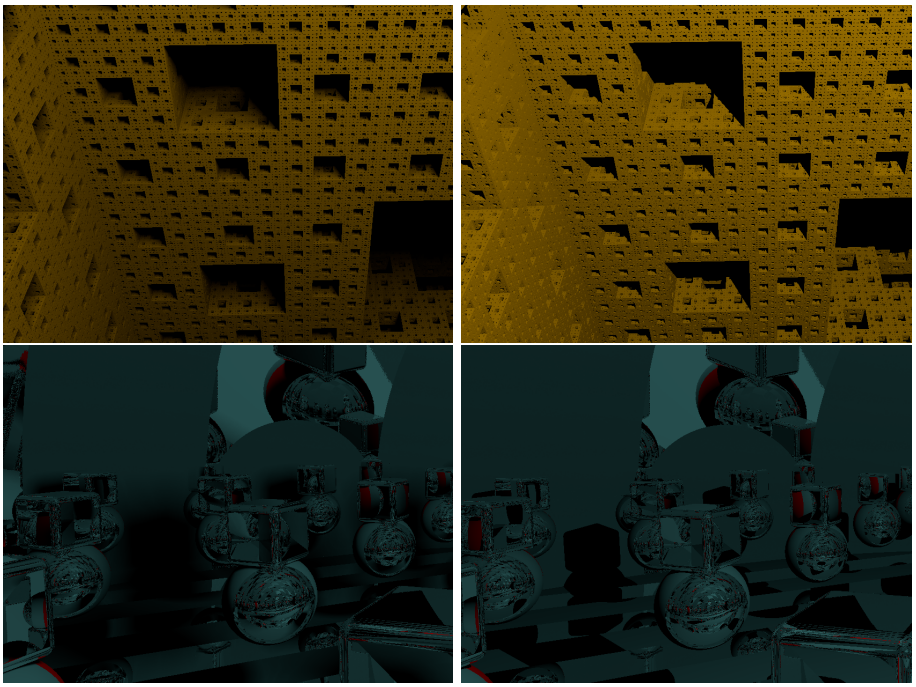

Abbildung 4: Mit Softshadows

Abbildung 5: Ohne Softshadows

Ein weiterer Aspekt warum Refraktion schwieriger durch Spheretracing umsetzbar ist, stützt sich auf den SDF Operationen. Nur wenige erhalten auch innerhalb von Körpern korrekte SDFs [Quib], Spheretracing innerhalb von Objekten lässt sich also nur in Spezialfällen anwenden.

Besonderer Fokus wurde darauf gelegt Aliasingprobleme von Geometrie mit einer sehr hohen Frequenz an Oberflächendetails zu lösen. Hierbei haben uns die SDF Eigenschaften erlaubt, eine neuartige Lösung zu entwickeln, welche visuell versprechende Ergebnisse erzielt. Aus Zeitgründen wurde im Umfang der Ausarbeitung nicht weiter geforscht aber ein Lageplan erstellt um diese Technik weiter zu entwickeln. Ziel hierbei ist es einen allgemein auf SDBs Anwendbaren LOD Operator zu erhalten, welcher den Oberflächendetailgrad nur soweit reduziert wie nötig ist um eine weitestgehend Aliasingfreie Darstellung zu ermöglichen. Dafür fehlen noch:

- Beweis das der LOD Operator auch auf SDB angewandt werden kann
- Erforschung von Effekten uniformer Raumskalierung auf Oberflächeneigenschaften, Verbindungsherstellung zum Bereich der Signalverarbeitung
- Ausarbeitung eines LOD Operators welcher mit einem quadratischen Detailabfall arbeitet

6 Schluss

In dieser Ausarbeitung wurde das Konzept und die besonderen Eigenschaften eines Spheretracers vorgestellt. Aus diesen Eigenschaften wurde eine neuartige

Methode zu LOD entwickelt. Eine prototypische Implementierung, welche diese Methode umsetzt und so detaillierte Geometrie mit einem geringen Maß an Aliasing in Echtzeit darstellen kann wurde gemacht.

Ein Großteil der Probleme die hier durch den Einsatz von GLSL zusammen mit JMonkey aufgekommen sind (rigide Szenenbeschreibung, kein nativer Callstack) lassen sich durch partielle Evaluation und Codegeneriereung lösen.

6.1 Ausblick

Eine realistische Anwendung über Technologiedemonstrationen hinaus werden vor allem im Bereich der VR gesehen. Dort kann sich die der Tausch von Szenenkomplexität zu realistischer Darstellung bezahlt machen. Hier muss noch erforscht werden welche Optimierungen bei stereoskopischen Rendering möglich sind. Da dabei mehrere Strahlen parallel verlaufen sollte die Nutzung eines einzigen Strahls mit einer pessimistischeren SDB bis zu einer gewissen Oberflächen Distanz möglich sein. Ebenso sollten sich in dem VR Anwendungsfall Fixed Foveated Rendering zu einer größeren Ersparnis als bei Rasterung führen, die Kosten pro Szene sind rein proportional zu der Auflösung und es gibt keine fixen Kosten von Geometrietransformationen wie bei Rasterung.

Literatur

[BV18] Csaba Bálint und Gábor Valasek. "Accelerating Sphere Tracing". In: *EG 2018 - Short Papers* (2018), 4 pages. ISSN: 1017-4656. DOI: 10.2312/EGS.20181037. URL: https://diglib.eg.org/handle/10.2312/egs20181037 (besucht am 01. 03. 2021).

[BVG19] Csaba Bálint, Gábor Valasek und Lajos Gergó. "Operations on Signed Distance Functions". In: Acta Cybernetica. Bd. 24. 6. März 2019. DOI: 10.14232/actacyb.24.1.2019.3.

[Har96] John C. Hart. "Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces". In: *The Visual Computer* 12.10 (18. Dez. 1996), S. 527–545. ISSN: 01782789. DOI: 10.1007/s003710050084. URL: http://link.springer.com/10.1007/s003710050084 (besucht am 01. 03. 2021).

[HSK89] J. C. Hart, D. J. Sandin und L. H. Kauffman. "Ray Tracing Deterministic 3-D Fractals". In: *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '89. New York, NY, USA: Association for Computing Machinery, 1. Juli 1989, S. 289–296. ISBN: 978-0-89791-312-6. DOI: 10.1145/74333.74363. URL: https://doi.org/10.1145/74333.74363 (besucht am 17. 03. 2021).

[Kei+14] Benjamin Keinert u. a. "Enhanced Sphere Tracing". In: *Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference* (2014), 8 pages. DOI: 10.2312/STAG.20141233. URL: http://diglib.eg.org/handle/10.2312/stag.20141233.001-008 (besucht am 01. 03. 2021).

[Quia] Iñigo Quilez. *Inigo Quilez Distance Functions*. URL: https://www.iquilezles.org/www/articles/distfunctions/distfunctions.htm (besucht am 01. 03. 2021).

[Quib] Iñigo Quilez. *Inigo Quilez Interior SDFs*. URL: https://www.iquilezles.org/www/articles/interiordistance/interiordistance.htm (besucht am 19. 03. 2021).

[Quic] Iñigo Quilez. *Inigo Quilez Menger Fractal*. URL: https://www.iquilezles.org/www/articles/menger/menger.htm (besucht am 17. 03. 2021).

[Quid] Iñigo Quilez. *Inigo Quilez Soft Shadows in Raymarched SDFs*. URL: https://www.iquilezles.org/www/articles/rmshadows/rmshadows.htm (besucht am 17. 03. 2021).

[Quie] Iñigo Quilez. *Iq - Shadertoy BETA*. URL: https://www.shadertoy.com/user/iq/sort=popular (besucht am 18. 03. 2021).

[Qui08] Iñigo Quilez. "Rendering Worlds With Two Triangles". NVScene 08. 22. Aug. 2008. URL: https://www.iquilezles.org/www/material/nvscene2008/rwwtt.pdf (besucht am 17. 03. 2021).

[Sun+20] Yifan Sun u. a. *Summarizing CPU and GPU Design Trends with Product Data*. 13. Juli 2020. arXiv: 1911.11313 [cs]. URL: http://arxiv.org/abs/1911.11313 (besucht am 18. 03. 2021).