HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Herr
Daniel Pollok

Photogrammetrie in Computerspielen – Welche Herausforderungen beinhaltet die Technologie für Spielentwickler und wie können diese bewältigt werden?

2017

Fakultät für Angewandte Computer- und Biowissenschaften

BACHELORARBEIT

Photogrammetrie in Computerspielen – Welche Herausforderungen beinhaltet die Technologie für Spielentwickler und wie können diese bewältigt werden?

Autor/in:
Herr Daniel Pollok

Studiengang:
Medieninformatik und Interaktives Entertainment

Seminargruppe:
MI13w2-B

Erstprüfer:
Prof. Alexander Marbach

Zweitprüfer:
Dipl. Ing. Sieglinde Klimant

Einreichung:
Mittweida, 06.02.2017

Faculty Applied Computer Sciences & Biosciences

BACHELOR THESIS

Photogrammetry in computer games – Which challenges does the technology hold for game developers and how can they be overcome?

author:
Mr. Daniel Pollok

course of studies:
Media Informatics and Interactive Entertainment

seminar group:
MI13w2-B

first examiner:
Prof. Alexander Marbach

second examiner:
Dipl. Ing. Sieglinde Klimant

submission:
Mittweida, 06.02.2017

Bibliografische Angaben

Pollok, Daniel

Photogrammetrie in Computerspielen – Welche Herausforderungen beinhaltet die Technik für Spielentwickler und wie können diese bewältigt werden?

Photogrammetry in computer games – Which challenges does the technology hold for game developers and how can they be overcome?

47 Seiten, Hochschule Mittweida, University of Applied Sciences,
Fakultät für Angewandte Computer- und Biowissenschaften, Bachelorarbeit, 2017

Abstract

Die vorliegende Arbeit befasst sich mit den Herausforderungen, die bei der Verwendung von Photogrammetrie für Computerspiele entstehen. Ziel der Arbeit ist es, die Lösungsmöglichkeiten für diese Probleme zu ergründen und ihre Vor- und Nachteile zu beleuchten. Die Erkenntnisse werden am Ende der Arbeit in einem Leitfaden zusammengefasst, der den richtigen Umgang mit der Technik beschreibt.

Inhaltsverzeichnis

Abkürzungsverzeichnis

et al.	et alii
etc.	et cetera
Hrsg.	Herausgeber
PBR	Physically Based Rendering
HDR	High Dynamic Range
LOD	Level of Detail
SDK	Software Development Kit
Prefab	Prefabrication

Abbildungsverzeichnis

1 Einleitung

1.1 Hinführung zur Thematik

Wie Arnold Gehlen bereits treffend bemerkte, Fortschritt ist der Übergang von Situationen, deren Nachteile man schon kennt, zu Situationen, deren Nachteile man noch nicht kennt. Dasselbe gilt auch für die Grafikentwicklung in Computerspielen. Stetig werden neue Technologien erforscht um bessere Grafiken zu produzieren und jede Technik bringt neue Herausforderungen mit sich.

Als Anfang des 21. Jahrhunderts die Hardware begann, deutlich leistungsfähiger zu werden, wuchs auch der Bedarf an Techniken zur Herstellung realistischer Assets und Charaktere in Computerspielen. Spielentwickler fingen an, mit 3D-Scanning-Verfahren auf Grundlage von Fotos Spielcharaktere zu erstellen.

Dies markierte den Anfang der Verwendung von 3D-Scans zur Herstellung von Grafiken und führte später zur Verwendung von Photogrammetrie in Computerspielen. Eine Technik, die zuvor vorrangig in der Kartographie verwendet und Wurzeln in Mathematik, Optik und Geometrie besitzt (Foster et al. 2014, S.14).

Bereits bei diesen ersten Versuchen, die Technologie in die Spielentwicklung einzubinden, offenbarten sich schnell die neuen Probleme, die diese Technik mit sich bringt. Seitdem wurden 3D-Scanning und Photogrammetrie immer häufiger bei Computerspielen aufgrund der realistischen Resultate verwendet. Obwohl sich die Verfahren dabei stark weiterentwickelt haben und Entwickler heutzutage über deutlich leistungsfähigere Hardware als damals verfügen, sind viele Probleme bei der Verwendung von Photogrammetrie bestehen geblieben sowie neue hinzugekommen.

1.2 Fragestellung

Die Technik der Photogrammetrie existiert zwar schon seit Jahrhunderten und die Theorie hinter ihr wurde dementsprechend ausführlich erforscht und dokumentiert, die Fragen der praktischen Anwendung für Computerspiele beantworten diese Forschungen aber nicht. Sie stellen lediglich die Berechnungen für jene Programme bereit, die Spielentwickler nutzen um aus Fotografien hochdetaillierte 3D-Modelle zu extrahieren. Wie mit diesen Modellen im Zusammenhang mit der Spielentwicklung umgegangen werden muss, ist aber eine ganz andere Frage.

Die praktische Anwendung in einem Spiel mit Echtzeit-Rendering unterscheidet sich stark von der Verwendung von Photogrammetrie in Forschungsbereichen. Die Technik muss nicht nur sauber durchgeführt, sondern sinnvoll und effizient in den bestehenden Ablauf der Spielentwicklung eingebunden werden. Alle Schritte müssen auf die Technik abgestimmt werden und das Spiel muss weiterhin flüssig laufen.

Die ersten Photogrammetrie-Versuche scheitern deswegen bei den meisten Entwicklern schnell, wenn vorher keine Erfahrung mit der Technik bestand. Die vorliegende Arbeit befasst sich daher mit den Herausforderungen speziell für die Verwendung von Photogrammetrie in Computerspielen. Die entstehenden Probleme werden dargestellt und Lösungsmöglichkeiten mit ihren Vor- und Nachteilen abgewogen. Schlussendlich werden die Ergebnisse der Untersuchungen in einem Leitfaden zusammengefasst, der darstellt, wie Photogrammetrie für ein Computerspiel verwendet werden sollte.

1.3 Recherche

Die verwendeten Informationen stammen aus Fachliteratur und Fachzeitschriften sowie Veröffentlichungen von Experten zu den Themen Photogrammetrie, Fotografie, Spielentwicklung und 3D-Scanning. Weiterhin wurde von Erfahrungsberichten von Spielentwicklern Gebrauch gemacht, die mit der Technik bereits erstaunliche Ergebnisse erzielen konnten, auch wenn diese häufig nicht ins Detail gehen. Es wurde daher zusätzlich versucht, Spielentwickler und 3D-Artists direkt zu befragen, um detailliertere Informationen und persönliche Meinungen von Profis auf dem Gebiet zu erhalten. Da es jedoch nicht garantiert ist, dass solche Anfragen an Spielentwickler beantwortet werden, sind diese nicht zu einem essentiellen Teil der Arbeit geworden. Die Antworten, die von Spielentwicklern gegeben wurden, waren jedoch sehr aufschlussreich und haben die Arbeit maßgeblich beeinflusst.

2 Theorie

2.1 Wissenschaftlicher Hintergrund

2.1.1 Definition

Photogrammetrie ist definiert als Wissenschaft, genaue, verlässliche Messungen und dreidimensionale Koordinaten eines Objektes aus zweidimensionalen Bilder zu extrahieren. Die Technologie funktioniert auf Grundlage von Triangulation, einer mathematischen Berechnung zur Bestimmung der unbekannten Position eines Punktes innerhalb eines Dreiecks im dreidimensionalen Raum. Durch mehrere Bilder (wenigstens 2 oder mehr) mit überlappenden Bildpunkten können Strahlen von der Kamera zum Objekt berechnet werden. Mit den unterschiedlichen Winkeln dieser Strahlen kann die Position der Kamera in Relation zum Objekt berechnet werden. Bei der Photogrammetrie wird dieses Prinzip auf mehrere Punkte gleichzeitig angewendet. Ein Limit, wie viele Punkte zur selben Zeit berechnet werden können, besteht zumindest theoretisch nicht (Slama et al. 1980).

2.1.2 Anwendungsgebiete

Die Verwendung von Photogrammetrie hat eine lange Geschichte mit Wurzeln in der Mathematik, Optik und Geometrie. Die erste breite Anwendung der Technik wurde in der Anfertigung hochdetaillierter Kartographie gefunden. Seitdem sind viele weitere Verwendungsmöglichkeiten hinzugekommen.

Eine aktuelle Liste an Anwendungsgebieten besteht aus:

- Erfassung und Kartierung von Geoinformationen
- Dokumentation
- Denkmalpflege und Architektur
- Luft-, Land- und Unterwasserarchäologie
- Überwachungen von Verformungen der Erdoberfläche und Bauwerken

- Bauingenieurswesen
- Automobil-, Luftfahrt- und Schiffsfahrtindustrie
- Zahnmedizin, Orthopädie und Biomechanik
- Forensik (Redweik 2013, S. 133-183)

Die Anwendung der Technik für Computergrafiken, speziell für Computerspiele und visuelle Effekte in Filmen, ist mit Anfang des 21. Jahrhunderts zu dieser Liste hinzugekommen und wird ausführlicher in dieser Arbeit behandelt.

2.1.3 Arten

Die Technologie lässt sich grundsätzlich in zwei Arten unterscheiden:

- Luftbildphotogrammetrie (engl.: arial photogrammetry) und
- Nahbereichsphotogrammetrie (engl.: close-range photogrammetry).

Bei der Luftbildphotogrammetrie ist die Kamera an einem Luftfahrzeug wie etwa einer Drohne montiert und wird meist senkrecht zum Boden ausgerichtet. Es werden mehrere überlappende Fotos des Bodens genommen, während sich das Luftfahrzeug entlang der Flugbahn bewegt. Diese Fotos werden in einem Stereo-Plotter verarbeitet (ein Instrument, mit dem ein Operator zwei Fotos gleichzeitig in einer Stereobildansicht sehen kann). Diese Fotos werden auch bei der automatischen Bearbeitung von Digital Elevation Model (DEM) verwendet.

Bei der Nahbereichsphotogrammetrie befindet sich die Kamera nahe am Motiv und wird typischerweise in der Hand gehalten oder auf einem Stativ befestigt. Die Montierung auf einem Fahrzeug ist aber ebenso möglich. Normalerweise ist diese Art von Photogrammetrie nicht topographisch – das heißt, die Ausgabe sind nicht topografische Produkte wie Geländemodelle oder topografische Karten, sondern Zeichnungen, 3D-Modelle, Messungen und Point Clouds.

Eine Point Cloud ist eine Ansammlung von unverarbeiteten Eingangsinformationen bei der Photogrammetrie oder des Laserscannings eines Objektes. Die Informationen werden als einzelne Punkte mit einem Set aus 3D-Koordniaten, einer Farbe und einer Normale dargestellt (Huang et al. 2009).

2.2 Verwendung in Computerspielen

2.2.1 Beweggründe

Game Artists sind heutzutage in der Lage, mit Hilfe moderner 3D-Software und ihres umfangreichen Wissens vom 3D-Modeling, Texturing und Rendering beeindruckende Assets für Computerspiele zu produzieren. Auf den ersten Blick wirken diese Objekte überzeugend und geben dem Spieler kurz den Eindruck einer realistischen Spielwelt.

Bei genauerem Betrachten zerbricht diese Welt jedoch schnell. Das Gehirn des Spielers beginnt die Objekte genau zu analysieren und mit der bekannten Realität zu vergleichen. Der Spieler merkt, wenn auch manchmal nur unterbewusst, dass etwas nicht stimmt und die Illusion zerbricht. Er bemerkt die gekachelten Texturen, die fehlende Willkür und Detailtiefe der Formen und Oberflächen. Selbst die begabtesten 3D-Artists sind meist nicht in der Lage die Natur so zu imitieren, das der Spieler sie ohne zu fragen als Realität akzeptiert.

Das hat jedoch nicht mit dem Stand der Technologie zu tun, die vorhandenen Softwarepakete sind leistungsfähig genug, um diese nötigen Details darzustellen. Abgesehen von der immensen Zeit, die nötig wäre und der dadurch entstehenden Kosten, liegt das Problem vielmehr bei den Game Artists selbst und ihrer fehlenden gedanklichen Kapazität.

Die Natur ist einfach zu komplex, sie ist unberechenbar und ergibt gleichzeitig immer Sinn. Risse und Abschürfungen an einem Felsen etwa sind durch Jahre der Witterung entstanden und nicht durch vorsichtiges Denken und Planen von Spielentwicklern künstlich hinzugefügt. Durch jahrelanges Beobachten dieser Details kann das menschliche Gehirn sehr schnell zwischen Realität und Imitat unterscheiden (Poznanski 2014).

Photogrammetrie hingegen bietet immense Möglichkeiten realistische Spielwelten zu erstellen, deren visuelle Qualität nicht von dem Geschick eines 3D-Artists abhängig sind. Der Realismus der Natur wird nicht länger versucht von Menschenhand nachzuahmen, sondern wird quasi direkt in die Spielwelt übertragen.

2.2.2 Jüngste Beispiele

Betrachtet man Computerspiele der letzten Jahre so stechen drei Spiele heraus, die umfangreichen Gebrauch von Photogrammetrie in ihrer Entwicklung gemacht haben:

„The Vanishing of Ethan Carter“ von 2014, entwickelt von The Astronauts, „Star Wars Battlefront“ von EA Dice aus dem Jahr 2015 und dem noch zu erscheinenden „Get Even“ von Entwickler Farm 51. Diese Entwickler haben Grafik zu ihrem Hauptverkaufsargument gemacht und sind dafür entsprechend bekannt geworden.

„The Vanishing of Ethan Carter” ist ein gutes Beispiel dafür, dass Photogrammetrie nicht zwangsläufig mit komplettem Fotorealismus einhergehen muss. Das Ziel der Entwickler war hauptsächlich eine glaubwürdige Spielwelt zu erschaffen, die sich für den Spieler echt anfühlt, ohne genau der Realität zu gleichen. Viele Assets wie Felsen, Bäume und ganze Häuser wurden abfotografiert und rekonstruiert. Genauso befinden sich aber auch traditionell erstellte Assets im Spiel, sowie stilisierte Beleuchtung und Postproduktion. So erschuf das Team von The Astronauts weniger eine Abbildung der Realität als vielmehr eine eigene Welt mit einem speziellen visuellen Stil, der die Handlung der Story unterstützt (Poznanski 2014).

In dem im Jahr 2015 erschienenen „Star Wars: Battlefront“ wurde dagegen versucht wirklich jedes Detail einzufangen und den größtmöglichen Realismus zu erschaffen. Dazu besuchten die Entwickler zu einem großen Teil die tatsächlichen Drehorte des Filmes und nahmen abertausende Fotos zur späteren Rekonstruktion auf. Für jedes Level im Spiel wurde ein durch Luftbildphotogrammetrie erstelltes Terrain und ein durch Nahbildphotogrammetrie erzeugtes Levelkit erzeugt. Ebenso wurden viele der Assets aus Fotos von originalen Filmrequisiten erstellt wie Helme, Kleidung und Waffen. Dasselbe gilt für Aufnahmen der Darsteller für die Charaktermodelle. Das macht „Star Wars: Battlefront zu einem Computerspiel mit wirklich allumfassenden Einsatz von Photogrammetrie (Electronic Arts Inc. 2015).

Über den Einsatz der Technik in „Get Even“, dass zurzeit noch von Farm 51 entwickelt wird, können noch keine wirklich genauen Angaben gemacht werden. Tech-Demos und Trailer lassen jedoch erahnen, dass ebenso wie bei EA kompletter Fotorealismus für die Level angestrebt wird. Laut Entwickler Farm 51 werden dabei große Mengen an photogrammetrischen Daten von Kameras und Scannern benutzt. Ob das Endergebnis den Trailern gerecht wird, bleibt aber noch abzuwarten (Reuther 2014).

2.2.3 Ausrüstung und Software

Für die erfolgreiche Durchführung eines Photogrammetrie-Projektes werden generell nur eine Kamera und eine Photogrammetrie-Software benötigt. Es besteht eine Vielzahl an optionalen Gegenständen, die das Aufnehmen erleichtern wie Stative, Fernsteuerungen, Drohnen und viele mehr. Die Kamera und Software sind aber die einzigen zwei essentiellen Bestandteile zum erfolgreichen Durchführen des Prozesses.

Die Fotos können mit allen Arten von Kameras geschossen werden, von modernen Handykameras bis zu High-End-Digitalkameras und allen Modellen dazwischen. Die Qualität der Kamera überträgt sich jedoch direkt auf die Qualität der Fotos und damit auch auf das spätere 3D-Modell. Jede Kamera mit einer Auflösung unter 5 Megapixeln sollte deshalb vermieden werden. Die konstantesten Resultate bietet ein Kameraobjektiv mit fester Brennweite, wie ein 50 mm Objektiv. Fischaugen- und andere Weitwinkelobjektive sind aufgrund der starken Verzerrung des Fotomotivs weniger geeignet. Nicht jede Photogrammetrie-Software ist in der Lage mit dieser Verzerrung zu arbeiten (Sketchfab Inc. 2015).

Auf Softwareseite verwenden die meisten Spielentwickler zum Berechnen der Modelle Agisoft Photoscan oder Autodesk 123D Catch. Es existiert jedoch eine Vielzahl ähnlicher Software wie PhotoModeler, Visual SFM, Photosynth und andere, jede davon mit ihren Vor- und Nachteilen. Ein genauer Vergleich wurde von den Designern und Entwicklern von Arc-Team durchgeführt, die Recherche ist frei verfügbar (Moraes 2016).

123D Catch ist eine kostenlose Software mit einer sehr benutzerfreundlichen Oberfläche und wird aufgrund dessen von vielen Hobbyentwicklern favorisiert. Die Fotos werden auf Autodesk-Servern verarbeitet, ein leistungsfähiger Rechner ist nicht zwingend notwendig. Nach Abschluss der Berechnungen wird dem Nutzer das fertige Modell zugeschickt. Es gibt daher nicht viele Konfigurationsmöglichkeiten während des Prozesses, was 123D Catch für die meisten professionellen Spielentwickler unattraktiv macht (Autodesk Inc. 2017).

Agisoft Photoscan ist das Programm, dass am meisten Verwendung in der Entwicklung von AAA-Spielen findet. Die Software verfügt über sehr ausführliche Konfigurationsmöglichkeiten während des Prozesses und ermöglicht so optimale Ergebnisse für professionelle Entwickler. Die Resultate sind jedoch sehr stark abhängig von der Leistung der CPU, Grafikkarte und vom Arbeitsspeicher des jeweiligen Rechners. Speziell der Bedarf an benötigtem Arbeitsspeicher für die Berechnung einer großen Anzahl hochauflösender Bilder ist immens. Bereits die Konstruktion eines Modells in hoher Qualität aus 500 Fotos benötigt 60-180 GB RAM (Agisoft LLC 2014).

2.2.4 Prozessablauf

Wie von Heaven formuliert, „Photogrammetrie funktioniert, indem Hunderte Fotografien eines Objekts oder einer Szene aus mehreren Winkeln aufgenommen und in einem 3D-Modell kombiniert werden. Ein Algorithmus findet dann die gemeinsamen Punkte zwischen den Fotos zur Erstellung einer Point Cloud des Ziels. Schließlich können die Vertices der Point Cloud in einem Mesh zusammengefügt und so ein äußerst detailliertes polygonales Modell erstellt werden.“ (Heaven 2014, S.18)

Dieser Prozess, mit Photogrammetrie Assets für Computerspiele zu erzeugen, startet mit dem Aufnehmen von Fotos unter möglichst optimalen Aufnahmebedingungen. Welche Bedingungen dabei erfüllt sein sollten wird in Kapitel 3.1 behandelt. Das zu erstellende Objekt muss von jedem möglichen Blickwinkel abfotografiert werden, um für die Software genug Informationen für ein lückenloses Modell zu garantieren.

Die Anzahl der Fotos hängt dabei stark von der Größe und Komplexität des Objektes ab. Generell gilt bei diesem Schritt die Faustregel, immer mehr Fotos zu machen als notwendig sind. Eine größere Anzahl an Fotos verlängert dabei zwar die Rechenzeit, garantiert aber ein lückenfreies Modell ohne Artefakte.

Für die Aufnahme aus verschiedenen Blickwinkeln sollte ein Foto im Abstand von 10-15 Grad auf der horizontalen und vertikalen Achse geschossen werden. Zwischen den Fotos sollte zudem eine Überlappung von 50-60% bestehen. Dieses Überlappen von Informationen zwischen 2 oder mehreren Fotos ist eine der wichtigsten Grundlagen auf denen die Photogrammetrie-Software die 3D-Modelle erzeugt (Sketchfab Inc. 2015).

Sind alle nötigen Fotos aufgenommen, können sie in die Software importiert werden. Die folgenden Schritte werden speziell für die Verwendung von Agisoft Photoscan beleuchtet, da dieses Programm zum Industriestandard geworden ist. Dementsprechend basieren die folgenden Abschnitte auf dem Agisoft Photoscan Manual (Agisoft LLC 2016). Es lohnt sich an diesem Punkt die Bilder genau zu sortieren, um die Rechenzeit zu optimieren und die Qualität des Modells nicht mit ungenauen Informationen aus unbrauchbaren Fotos zu verschlechtern.

Um noch akkuratere Resultate zu bekommen, besteht die Möglichkeit das jeweilige Objekt zu maskieren, sodass die Software außerhalb des Objektbereiches keine Informationen bezieht. Für diesen Vorgang besitzt Photoscan ein eigenes Masking-Tool, die Fotos können aber auch außerhalb in Programmen wie Adobe Photoshop ausmaskiert werden.

Anschließend sucht die Software nach gemeinsamen Bildpunkten in den importierten Fotos. Gleichzeitig wird die Ausrichtung und Position der Kamera relativ zum fotografierten Objekt berechnet. Das Ergebnis ist eine spärliche Point Cloud mit den jeweiligen Kamerapositionen, von denen die Fotos aufgenommen wurden.

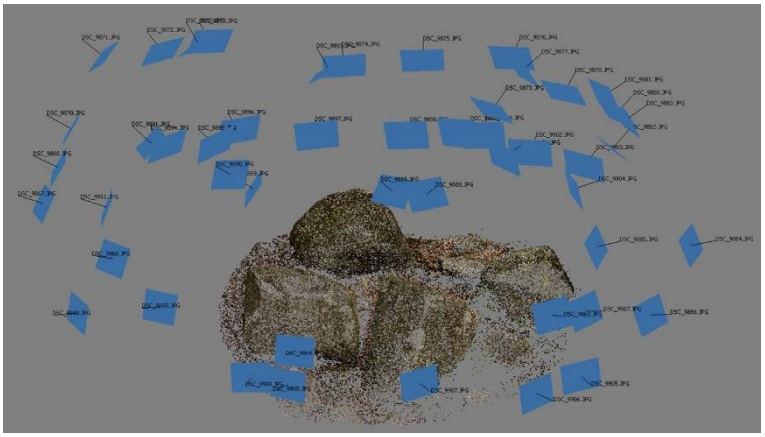

Abbildung 1: Spärliche Point Cloud (Kamerapositionen in blau) (Poznanski 2014)

Auf Grundlage der Kameras in der Szene berechnet das Programm Tiefeninformationen für jede Kamera und vereint diese in einer einzelnen dichten Point Cloud bestehend aus Millionen von Bildpunkten. Die genaue Auflösung und Qualität kann zuvor exakt konfiguriert werden. Diese Einstellungen beeinflussen maßgeblich die Zeit, die zur Berechnung benötigt wird.

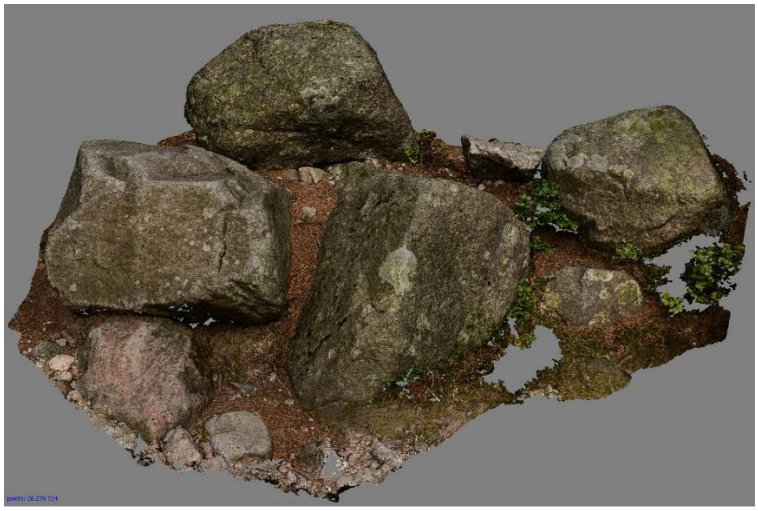

Abbildung 2: Dichte Point Cloud (Poznanski 2014)

Selbst bei gut ausmaskierten Fotos entstehen häufig überschüssige Punkte in der Point Cloud, die nicht zum gewünschten Objekt gehören. Diese können anschließend manuell gelöscht werden, damit sich im späteren Modell keine Artefakte bilden. Aus der so optimierten Point Cloud erzeugt die Software ein HighPoly-Modell. Dafür kann zwischen zwei Algorithmen gewählt werden: Height Field und Arbitrary.

Height Field eignet sich am besten für ebene Flächen wie Terrains und Reliefs. Es sollte speziell für Luftbildphotogrammetrie (engl. arial photogrammetry) ausgewählt werden, da es weniger Arbeitsspeicher benötigt und die Verarbeitung größerer Datensätze ermöglicht.

Arbitrary hingegen eignet sich theoretisch für alle Arten von Objekten. Es sollte für geschlossene Objekte wie Statuen, Gebäude, Bäume oder Steine verwendet werden. Durch dieses breite Spektrum an Objekten verbraucht der Algorithmus aber auch deutlich mehr Arbeitsspeicher.

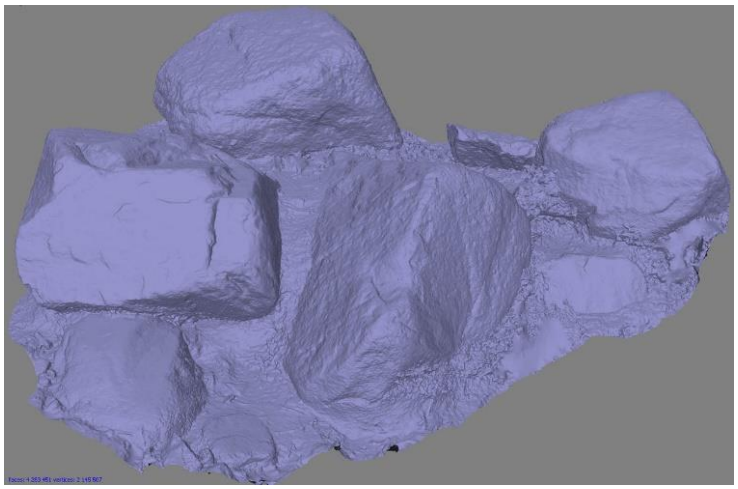

Abbildung 3: HighPoly-Modell (Poznanski 2014)

Das erzeugte Modell kann innerhalb der Software nun noch weiterbearbeitet werden. So kann etwa die Anzahl der Polygone reduziert, abgetrennte Teile des Modells entfernt und entstandene Löcher geschlossen werden.

Im nächsten Schritt wird die Textur für das Modell generiert, wofür die Software mehrere Möglichkeiten des Texture Mappings bereitstellt. Aufgrund der hohen Informationsdichte muss die Auflösung der Textur entsprechend hoch eingestellt werden, um alle Details einfangen zu können. Später kann die Größe der Textur angepasst werden, sollte die hohe Auflösung Probleme verursachen.

Durch die Art und Weise wie Photoscan die Textur aus einzelnen Bildpunkten erstellt, sind die entstehenden UV-Koordinaten allerdings äußerst ineffizient und müssen später in jedem Fall optimiert werden. Das Modell mit Textur kann nun aus Photoscan im gewünschten Format exportiert werden (Agisoft LLC 2016).

3 Probleme und Lösungsmöglichkeiten

3.1 Generelle Einschränkungen

Gute Fotos sind absolut essentiell für ein erfolgreiches Photogrammetrie-Projekt. Sollte der Algorithmus scheitern, liegt der Fehler nahezu immer an zu schlechten und für die Software verwirrenden Aufnahmen. Im Idealfall werden nur scharfe, saubere und gleichmäßig ausgeleuchtete Bilder in das Programm importiert. Das Objekt wurde von allen Seiten und Winkeln abfotografiert. Der Hintergrund ist möglich unkompliziert oder sogar komplett gleichmäßig, wie eine Wand oder ein Green Screen. Dies sind die optimalen Voraussetzungen für einen geeigneten Fotosatz.

Jedoch gibt es eine Vielzahl an Bedingungen bei der Fotoaufnahme, die den Photogrammetrie-Prozess später scheitern lassen. Diese stellen große Schwierigkeiten für jeden Spielentwickler, der die Technik verwendet dar und werden deshalb in diesem Kapitel ausführlich beleuchtet.

Unscharfe und verschwommene Bilder

Verschwommene Aufnahmen verwirren die Software und resultieren in Artefakten und inakkuraten Stellen im Modell. Ursache für solche Bilder sind meistens der fehlende Kamerafocus oder zu viel Bewegung in der Szene. Der Focus der Kamera kann leicht justiert werden und ist einer der einfachsten Wege, bessere Modelle zu erhalten.

Objekte, die sich bewegen, stellen ein größeres Problem dar. Besonders beim Fotografieren von Personen für Charaktermodelle wird dies schnell deutlich. Bereits minimale Bewegungen, wie z.B. ein Lächeln reichen aus, um die relative Position der Gesichtsmerkmale komplett zu verändern. Wenn die Software diese Bilder später vergleicht, resultieren diese widersprüchlichen Informationen in inkorrekten oder unscharfen Modellen **(**Blizard 2014).

Sind verschwommene Fotos das Resultat einer unruhigen Hand, kann dies mit optionalem Zubehör behoben werden. Ein gutes Stativ verhindert verwackelte Aufnahmen und gehört für erfahrene Entwickler zur Standardausrüstung. Zusätzlich kann eine Fernsteuerung helfen, jeglichen unnötigen Kontakt mit der Kamera zu eliminieren (Sketchfab Inc. 2015).

Tiefenschärfe (Depth of Field) (Malison 2013)

Die Tiefenschärfe stellt ein ähnliches Problem dar wie der fehlende Kamerafocus. Die Teile des Objektes, die zu nah oder weit weg von der Kamera sind verschwimmen und nur der mittlere Bereich des Fotos ist wirklich scharf. Der entscheidende Faktor um dies zu umgehen ist die Blendenzahl der Kamera. Sie bestimmt wie groß oder klein die Öffnung der Linse beim Fotografieren ist.

Um die Tiefenschärfe und damit den scharfen Bereich des Fotos zu vergrößern sollte mit einer großen Blendenzahl fotografiert werden, welche die Öffnung der Linse sehr klein hält. Dabei ist zu beachten, dass durch die geringe Menge an einfallendem Licht die Linse länger geöffnet bleiben muss. Eine Blendenzahl zwischen f/8 und f/16 ist dabei ein guter Richtwert.

Auch bei diesem Problem kann ein Stativ sehr hilfreich sein. Es erlaubt generell längere Belichtungszeiten ohne Bewegung der Kamera.

JPEG Kompression

Die Kompression bei Bildern im JPEG-Formate reicht aus, um falsche Punkte für das Tracking der Software zu verursachen. Ab einer höheren Kompressionsrate von 10 und mehr ist die Kompression nahezu verlustfrei, jedoch eignen sich komplett kompressionsfrei Bildformate deutlich besser (Li et al. 2002, S. 847-853). Spieleentwickler sind daher dazu übergegangen fast ausschließlich Bilder im RAW-Format für Photogrammmetrie-Projekte zu verwenden. Die rohen Bilddaten können sehr fein angepasst werden und später in positive Bildformate wie TIFFs konvertiert werden.

HDR Aufnahmen

Ein umstrittenes Thema in der Photogrammetrie ist die Verwendung von HDR-Fotos zur Rekonstruktion eines Modells. High Dynamic Range (HDR) Fotografie ist eine Technik, Bilder zu erstellen, bei der sowohl in den hellsten als auch in den dunkelsten Bereichen des Fotos sehr viele Details zu erkennen sind. Dafür werden mehrere Fotos mit unterschiedlicher Belichtung in einem HDR-Programm in einem Bild kombiniert (Lantz 2007).

Diese Eigenschaften lassen vermuten, dass sich solche Bilder perfekt für Photogrammetrie eignen. Schließlich bringt die Software die besten Ergebnisse, wenn mehr Details in einem Foto zu erkennen sind. Ebenso wird der negative Effekt von suboptimalen Lichtbedingungen reduziert (Joyce 2015). Aus diesen Gründen werden HDR-Aufnahmen häufig für diese Zwecke verwendet oder wenigstens mit ihnen experimentiert.

Die Technik besitzt jedoch durchaus Nachteile, die bedacht werden sollten. Viele Photogrammetrie-Programme benötigen für die Rekonstruktion die korrekten EXIF-Daten der Bilder. EXIF ist ein Industriestandard, der Informationen über das Foto (Aufnahmezeit und Belichtung) und über die Kamera (Modell, Bauart, Linse etc.) in einem Bildformat speichert. Durch das Kombinieren von mehreren Bildern haben viele HDR-Programme Schwierigkeiten, die korrekten EXIF-Daten zu liefern, was bei der Photogrammetrie später zu Problemen führen kann.

Die Kombination von Bildern hat zusätzlich oft einen negativen Effekt auf die Schärfe des finalen Bildes, was - wie bereits aufgezeigt - eines der größten Probleme für Photogrammetrie-Programme darstellt. Alle Fotos müssen in exakt derselben Position aufgenommen werden, um sie später verlustfrei kombinieren zu können. Um das zu erreichen, ist die Verwendung eines Stativs unabdingbar. Die HDR-Ergebnisse wirken jedoch trotzdem häufig nicht so scharf wie die einzelnen Fotos.

Vom Erstellen von HDR-Bildern mit Kameras, die einen HDR-Modus besitzen sollte komplett abgesehen werden. In diesem Modus wird das HDR-Bild von der Kamera, aus üblicherweise drei Aufnahmen, zusammengestellt. Dabei wird jedoch nicht die Belichtung, sondern der ISO-Wert der Kamera verändert. Bilder, die mit hohem ISO-Wert aufgenommen werden, besitzen ein hohes Maß an visuellem Rauschen (Noise), welches sich später deutlich sichtbar im Modell niederschlägt (Malison 2013).

Die Verwendung von HDR-Bildern hängt so häufig von der Situation ab. Ist sich der Nutzer der Risiken bewusst, so kann sich das Experimentieren mit der Technik durchaus lohnen.

Transparente und reflektierende Objekte

Photogrammetrie-Software ist hauptsächlich für matte, lichtundurchlässige Objekte beabsichtigt, andere Materialien bereiten daher Probleme. Die Muster, nach denen die Software auf den Objekten sucht, bewegen sich bei transparenten und reflektierenden Objekten nicht zusammen mit ihrer Oberfläche. Bei Glas zum Beispiel kann im Bild sowohl Vorderseite des Objektes als auch die Innenseite, sowie was dahinterliegt gesehen werden. Sind Reflektionen im Spiel kann die Software zudem nicht zwischen einem Highlight von der Lichtquelle und einem weißen Punkt auf dem Objekt unterscheiden. Diese widersprüchlichen Informationen lassen die Rekonstruktion der Software schnell scheitern **(**Blizard 2014).

Um die Stärke der Reflektionen zu reduzieren, kann ein Polarisationsfilter für die Kamera benutzt werden. Für metallische Oberflächen sind diese Filter ungeeignet, für Gegenstände wie etwa ein Bild mit einem Glasrahmen kann er jedoch helfen, die hellen Punkte der Reflektion zu vermindern (Tirosh 2012).

Lässt sich das Problem so nicht beheben, muss auf Umgehungsstrategien zurückgegriffen werden. Der einzige Weg ein stark reflektierendes oder transparentes Objekt durch Photogrammetrie zu rekonstruieren ist die Oberfläche mit einer Beschichtung zu präparieren, die von der Software gescannt werden kann. Es gibt Beschichtungssprays aus feinkörnigem Pulver, das auf den meisten Oberflächen haftet und leicht wieder entfernt werden kann. Sie geben dem Objekt eine matte, gleichmäßige Oberfläche.

Die Gleichmäßigkeit kann aufgebrochen werden, indem zusätzlich Farbe auf das Objekt gespritzt wird. So hat die Software genügend Unregelmäßigkeiten für das Tracking. Ist die Aufnahme und die Rekonstruktion des Modells abgeschlossen, muss das Material mit den richtigen Shadern wiederhergestellt werden (Busby 2016).

Gleichmäßige Oberflächen

Wenn Objekte zu gleichmäßig sind hat die Software keine markanten Punkte, die verfolgt werden können. Der Algorithmus sucht nach Verschiebungen von bekannten Merkmalen in mehreren Fotos, fehlen diese, da alles gleicht aussieht, scheitert er. Die Rekonstruktion von blanken Wänden zum Beispiel bringen daher häufig schlechte Resultate. Das Hinzufügen von eindeutigen Punkten durch Sticker oder Ähnlichem kann helfen, diese erscheinen dann aber auch im finalen Modell und müssen manuell entfernt werden **(**Blizard 2014).

Repetitive Merkmale

Zu viele sich wiederholende Elemente stellen eine ähnliche Herausforderung dar. Wenn die relative Bewegung eines gewissen Musters von der Software verfolgt wird, kann sie auch schnell auf eine andere Stelle mit demselben Muster überspringen. Der Sprung wird von der Software als Bewegung von Bild zu Bild wahrgenommen und nicht von einem Merkmal das identisch an anderen Stelle des Objektes vorhanden ist. Beispiele dafür sind ebenmäßige Holzzäune, Gitterstäbe oder Honigwaben **(**Blizard 2014).

Dünne Objekte

Aufgrund der Art und Weise, wie die Software ein polygonales Modell aus einer Point Cloud erzeugt, sind Objekte bestehend aus hauchdünnen Elementen wie Haare sehr schwer zu generieren. Die Software platziert für diese Objekte zu wenige Punkte um eine zusammenhängende Form aus ihnen bilden zu können **(**Blizard 2014).

Überlappung

Zusätzlich kommt es gerade bei Objekten mit dünnen Elementen häufig vor, dass sich diese stark überlappen, wie zum Beispiel die Blätter eines Baumes. In so einem Fall werden Teile des Objektes in vielen Fotos mehr oder weniger verdeckt, wodurch lückenhafte Informationen entstehen **(**Blizard 2014). Dieser Effekt wird noch verstärkt, wenn sich die überlappenden Elemente sehr stark ähneln. Das Ergebnis ist meistens eine löchrige Point Cloud, in der die einzelnen Teile zu einem nicht identifizierbaren Klumpen verschmelzen. Die einzige Lösung besteht darin, die sich überlappenden Elemente einzeln zu fotografieren und später zusammenzuführen.

Wechselnde Lichtverhältnisse

Gute Belichtung ist essentiell für optimale Fotos. Im Idealfall werden die Bilder in einem Studio aufgenommen, wo sich Lichtverhältnisse anpassen lassen und konstant bleiben. Dieser Vorteil ist jedoch nicht immer gegeben. Für Naturaufnahmen und Landschaftselemente muss unter freiem Himmel fotografiert werden, wo sich das Licht innerhalb von Sekunden gänzlich verändern kann. Eine Veränderung der Schatten kann so von der Software als falsche Bewegung interpretiert werden. Die Zeit der Aufnahme sollte demnach auf Tage mit stabilen Wetterbedingungen abgepasst werden (Foster et al. 2014).

Tiefe Schatten

Der Algorithmus der Software funktioniert am besten ohne direkte Lichtquelle. Direkte Lichtquellen wie die Sonne verursachen tiefe Schatten, die von der Software aufgrund der wenigen Informationen schlecht verfolgt werden können. In manchen Fällen können an diesen Stellen sogar Löcher im Modell auftreten.

Die besten Bedingungen für das Fotografieren bietet daher ein komplett bewölkter Himmel oder indirekte Studiobeleuchtung. Es entstehen keine tiefen Schatten auf dem Objekt, es ist gleichmäßig ausgeleuchtet und die Lichtquelle ist konstant (Sachs 2016).

Über- und Unterbelichtung

Auch die Stärke des Lichtes sollte beim Fotografieren beachtet werden. Ist zu wenig Licht vorhanden um alle Formen des Objektes sichtbar zu machen, führt dies zu einem starken Informationsverlust. Speziell ambiente Beleuchtung, wenn zu stark gedimmt, neigt leicht dazu, die Objekte nicht ausreichend auszuleuchten. Überbelichtung ist jedoch genauso schlecht. Zu starkes Licht verursacht ausgebrannte Fotos, in denen viele Details verloren gehen. Wenn direkt in eine starke Lichtquelle fotografiert wird können zu dem ungewünschte Lens Flares entstehen. Das richtige Mittelmaß zu finden ist hier essentiell (Malison 2013).

3.2 Baked Lighting

Werden 3D-Modelle auf Grundlage von Fotos rekonstruiert, lässt es sich nicht vermeiden die in den Bildern bestehenden Lichtinformationen auf das Modell zu übertragen, es besitzt sogenanntes Baked Lighting. Das Resultat ist eine Textur, die nur in genau der Lichtsituation funktioniert, in der die Fotos ursprünglich geschossen wurden. Werden diese Texturen nicht nachträglich bearbeitet, sodass sie in jeder Lichtsituation funktionieren, sind sie für Computerspiele komplett unbrauchbar. Um dieses Problem zu lösen haben Entwickler verschiedene Workflows entwickelt, die Lichtinformationen komplett aus den Texturen zu entfernen.

Wie bereits in Kapitel 3.1 erwähnt, bereiten reflektierende Eigenschaften von Objekten große Probleme bei der Photogrammetrie. Jedoch besitzt so gut wie jedes Objekt einen gewissen Reflexionsgrad, auch wenn er häufig nur schwach sichtbar ist. Bringt der besagte Polarisationsfilter nicht die gewünschten Resultate oder wurden die Fotos ohne Filter aufgenommen, müssen die Highlights händig aus der Textur entfernt werden.

Dafür kann die Textur in einem Bildbearbeitungsprogramm, vorzugsweise Adobe Photoshop justiert werden. Durch Maskieren der Highlight-Bereiche und dem Anpassen der Gammawerte sowie der Belichtung (Exposure), können bereits gute Resultate erzielt werden. Starke Highlights können zudem durch vorsichtiges Verwenden des Kopierstempels überdeckt werden. Dieser Prozess ist eine der simpelsten Möglichkeiten Reflektionen zu entfernen und wird häufig von professionellen Scanning-Studios wie Ten 24 verwendet.

Abbildung 4: Erfolgreiche Highlight-Entfernung (Azzam 2016)

Die Schatten auf dem fotografierten Objekt können auf ähnliche Weise in Photoshop behoben werden. Es existieren aber auch Wege dies effektiver und akkurater zu erreichen.

Eine Möglichkeit ist die Verwendung eines „HDR/Grey-Chrome Ball Setups“, welches von Epic Games in einem Artikel genauer beschrieben wurde (Oh 2015). Die Grundidee dieser Methode ist, dasselbe Lichtszenario wie zum Zeitpunkt der Fotoaufnahme in einem 3D-Programm nachzustellen. Dazu wird ein Gerüst mit einer darauf befestigten mattgrauen Kugel, einer Chromkugel, sowie einer Farbpalette zur späteren Farbkalibrierung, zusammen mit dem Objekt fotografiert. Zusätzlich wird ein HDR-Bild der Umgebung zum Zeitpunkt der Fotoaufnahme genommen.

In der 3D-Software wird unter Verwendung des HDR-Bildes die Szene zunächst beleuchtet. Die herrschende Lichtintensität wird mit Hilfe der grauen Kugel und die Illumination durch den Himmel und Wolken in der Szene mit der Chromkugel angepasst.

Abbildung 5: HDR/Grey-Chrome Ball Setup (Oh 2015)

Damit die Beleuchtung auf dem Modell wirklich akkurat ist, sollte es auf Lebensgröße skaliert werden. Dafür sollte das Objekt gemessen oder Fotos genommen werden, welche für die Größe später referenziert werden können. Das Gestell der Kugeln kann ebenso als Referenz benutzt werden. Um die Lichtinformation korrekt zu baken sollte das Mesh des Zielobjektes auf 20% Grau gestellt werden, da dies ungefähr dem Wert der Graukugel entspricht.

Als nächstes können die Lichtinformationen vom Mesh in der Szene gebaked werden. Dabei entsteht eine invertierte Version des Lichtszenarios, auch Baked Lighting Map genannt. Diese wird mit der Originaltextur zusammengesetzt, um eine Textur ohne Lichtinformation zu erhalten. Die zu verwendende Operation für das Compositing ist:

Originaltextur/ („Baked Lighting Information“ x 5) = Textur ohne Lichtinformation

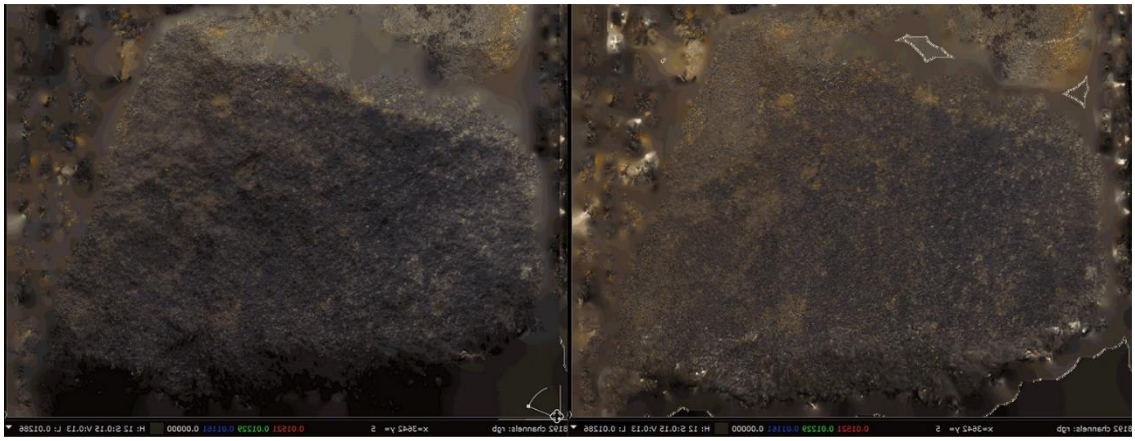

Abbildung 6: Felsen-Textur mit und ohne Lichtinformation (Antoine 2015)

Diese Technik wurde so in der UE4 Open World Demo von Epic Games verwendet und stellt eine der effektivsten und akkuratesten Methoden dar Lichtinformationen von Photogrammetrie-Modellen zu entfernen (Antoine 2015).

Eine weitere Lösung für dieses Problem ist der Photoshop-Workflow, welcher bei der Entwicklung von Star Wars Battlefront von EA Dice verwendet wurde. Dieser verlangt im Vergleich nicht so viel Know-how und erzielt trotzdem gute Resultate.

Nach erfolgreicher Rekonstruktion des Objektes wird die Textur in Photoshop mit dem Camera Raw-Plugin geöffnet. Die Textur sollte in 16-bit geladen werden, da dies ein umso größeres Farbspektrum ermöglicht. Hat die Textur ursprünglich nur 8-bit ist dies natürlich nicht möglich. Das Plug-In besitzt einen separaten Regler zur Schattenentfernung, welcher in vielen Fällen bereits gute Resultate bringt. Zusätzlich können weitere Eigenschaften wie Belichtung und Farbtemperatur angepasst werden (Brown et al. 2016).

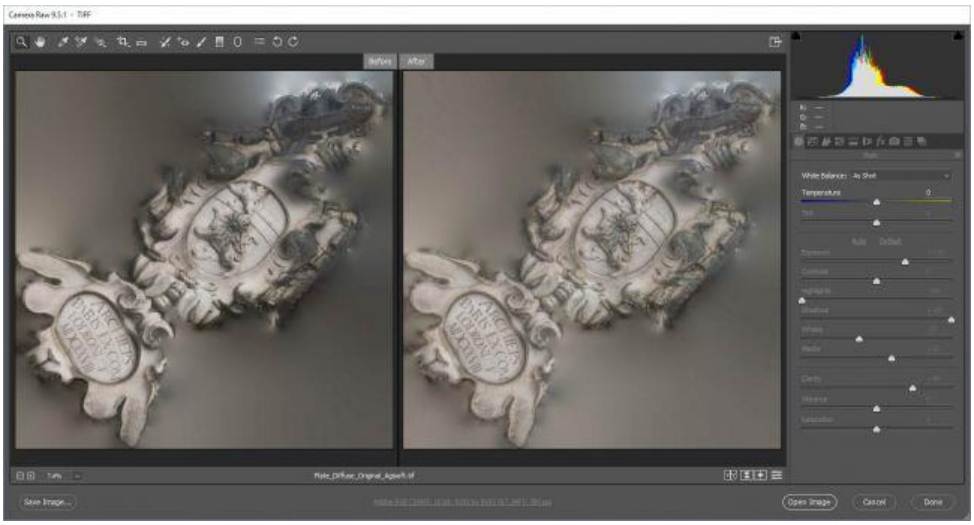

Abbildung 7: Photoshops Camera Raw Plugin (Azzam 2016)

Wurden mit dem Plug-In die meisten Schatten entfernt, wird die Object Space Normal Map des Objektes in Photoshop importiert, um die Ambient-Schatten zu beseitigen. Die Normal Map enthält die Ausrichtung der Polygone, welche in den RGB-Kanälen eingebettet sind. Um diese Informationen zu extrahieren müssen die Kanäle getrennt werden.

Dazu wird eine Schwarz-Weiß-Anpassungsebene benötigt und das jeweilige Preset (Rot-, Grün- und Blaufilter) für die drei Kanäle ausgewählt. Für jedes der drei aus den Presets entstandenen Bildern wird eine Levels-Anpassungsebene erstellt und das Kanalbild in die Maske eingefügt. Zusätzlich wird eine weitere Levels-Anpassungsebene mit der Ambient Occlusion Map als Maske benötigt. Durch manuelles Anpassen der Schieberegler können nun die Ambient-Schatten entfernt werden (Azzam 2016).

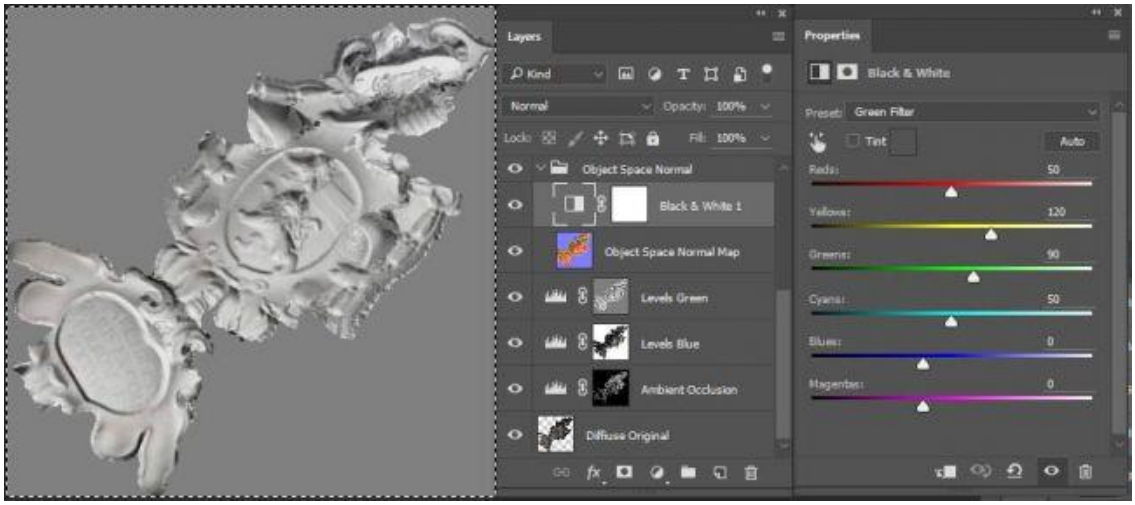

Abbildung 8: Kanaltrennung in Photoshop (Azzam 2016)

Am Beispiel von Star Wars: Battlefront kann man sehen, dass dieser Prozess durchaus sehr gute Resultate hervorbringen kann. Im Gegensatz zu dem von Epic Games entwickeltem Workflow, der semiautomatisiert abläuft wird jedoch eine Menge manuelle Einstellung benötigt. Deswegen sind die Resultate sehr stark von der Erfahrung des Benutzers abhängig.

3.3 Optimierung

3.3.1 Mesh

Ein essentieller Schritt für Photogrammetrie-Modelle, die in Computerspielen verwendet werden sollen, ist das Optimieren des Meshes. Wie in 2.4 beschrieben, steht am Ende des Rekonstruktionsprozesses ein Modell bestehend aus Millionen von Polygonen und einer hochdetaillierten Textur mit ineffizienten UV-Koordinaten. In einem Spiel mit Echtzeit-Rendering würde eine größere Anzahl solcher Modelle die Performance aufgrund der riesigen Datenmenge extrem negativ beeinflussen. Mit der nötigen Optimierung steht und fällt somit die Verwendung der Modelle. Die Datenmenge der Assets muss stark verringert werden, gleichzeitig sollen so wenig wie möglich Details an den Modellen verloren gehen.

Inzwischen besitzt nahezu jedes 3D-Programm Tools zum Retopologisieren und Optimieren eines Meshes. Der Standardworkflow besteht darin das High-Poly-Modell in die Software der Wahl zu laden, dort manuell eine Low-Poly-Variante zu erstellen und die verloren gegangenen Details mit verschiedenen Texture Maps wie Normal und Displacement Maps zu übertragen. Die vorhandenen Tools in den verschiedenen Programmen wie 3Ds Max, Maya, Zbrush und anderen ähneln sich dabei sehr stark. Eine objektive Bewertung, welche Software die besten Möglichkeiten bereitstellt, Meshes manuell zu optimieren lässt sich so nicht wirklich treffen. Der entscheidende Faktor ist der gekonnte Umgang mit den vorhandenen Tools. Somit sind die Ergebnisse stark vom 3D-Artist abhängig und nicht wirklich von der verwendeten Software.

Diese manuelle Optimierung, speziell von komplexen Photogrammetrie-Modellen, ist jedoch eine komplizierte Aufgabe für jeden 3D-Artist, die zudem viel Zeit kostet. Die Entwicklung geht daher in Richtung automatisierter Verfahren, mit denen sich das Problem schneller und besser lösen lässt.

Solche automatisierten Verfahren sind entweder bereits in den erwähnten 3D-Modeling-Programmen integriert oder in einer separaten Software vorhanden. Als Beispiel kann unter anderem ZRemesher in ZBrush, ProOptimizer in 3Ds Max oder die freie zugängliche Software Meshlab genannt werden. Das Problem, welches die meisten dieser automatisierten Programme haben ist, dass die Kontrolle über das Endergebnis sehr beschränkt ist.

In vielen dieser Programme kann vom Benutzer nur die gewünschte Polygonzahl eingestellt werden und die Software optimiert das Modell auf Grundlage des Algorithmus. Ist das Ergebnis nicht zufriedenstellend, kann der Prozess höchstens erneut mit mehr zur Verfügung stehenden Polygonen durchgeführt werden. Ausnahmen existieren jedoch trotzdem. In der jüngsten Version von ZBrush 4R7 zum Beispiel können bereits vor dem Optimierungsprozess, zusätzlich zum gewünschten Polycount, Richtlinien für den Edgeflow angelegt werden.

Wenn solche automatisierten Verfahren von Entwicklern verwendet werden, so dienen die Ergebnisse häufig nur als Grundlage und werden manuell weiterbearbeitet. So wird Zeit gespart und nicht komplett die Kontrolle über die Low-Poly-Modelle aufgegeben. Bei der Optimierung der Assets in Star Wars: Battlefront zum Beispiel, wurden die Objekte in Meshlab retopologisiert und wenn nötig in Topogun weiterbearbeitet. Die Texturprojektion wurde anschließend in ZBrush durchgeführt (Brown et al. 2016).

Eine Alternative zu diesen komplett automatisierten Verfahren bieten interaktive Softwarelösungen, wie das 2015 vorgestellte Instant Meshes. Der verwendete Prozess erlaubt mehr Kontrolle und Anpassung, während das Modell optimiert wird. Dazu wird von dem Modell zuerst ein Orientationsfeld berechnet, welches die Platzierung der Polygone leitet. In der zweiten Phase wird das Positionsfeld optimiert, dass Polygone des Output-Meshes gleichmäßig verteilt. Vertices werden dann von dem Positionsfeld extrahiert um das Mesh einzugrenzen. Der Edgeflow des neu entstandenen Meshes kann durch Hinzufügen von Pinselstrichen und anderen Tools interaktiv gesteuert und geändert werden. (Wenzel et al. 2015)

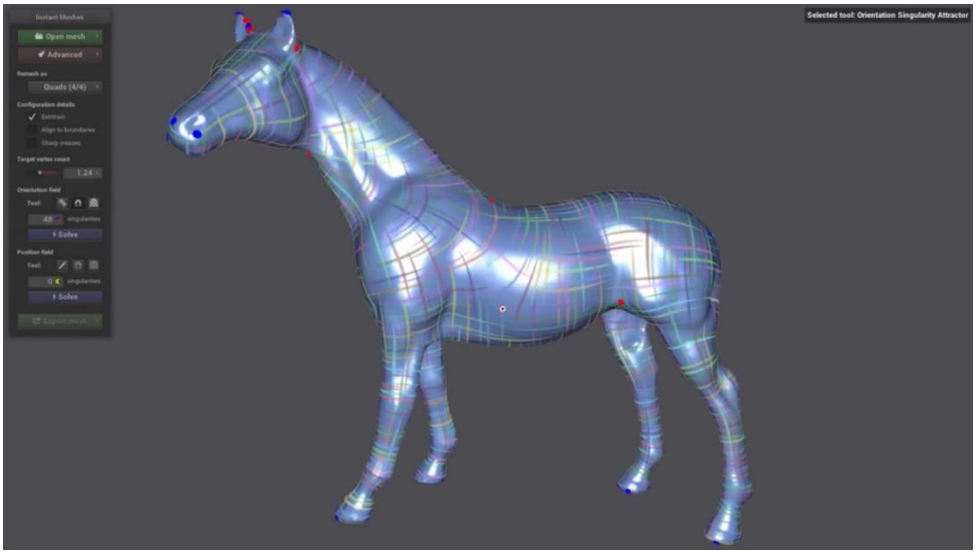

Abbildung 9: Orientationsfeld in Instant Meshes (Sorkine–Hornung 2015)

Auch wenn diese Software nicht perfekt ist, so ist der Ansatz, den automatisierten Optimierungsprozess interaktiv steuern zu können, eine interessante Entwicklung mit vielen Vorteilen. Speziell die Tatsache, dass sich die Topologie des Modells bei der Bearbeitung ohne lange Rechenzeit in Echtzeit aktualisiert ist ein Feature, welches in anderen Programmen so noch nicht vorhanden war.

Die Ergebnisse von automatisierten Optimierungsverfahren, sowohl mit als auch ohne interaktiven Bestandteilen, unterscheiden sich in ihrer Qualität deutlich. Eine genauere Recherche zu diesem Thema kann in der Bachelorarbeit von Florian Rüger mit dem Titel „Verfahren zur Aufbereitung von 3D-Modellen für die Verwendung in Echtzeitanwendungen“ gefunden werden (Rüger 2016).

Für Entwickler, deren Optimierungsprozess lediglich aus dem Reduzieren der Polygonanzahl eines Assets besteht, sind die vorgestellten Programme häufig ausreichend. Sollen jedoch alle Aspekte eines Modells optimiert werden, reichen diese nicht mehr aus. Die professionellste Lösung, eine umfassende Modell-Optimierung zu erreichen, stellt derzeit die Software Simplygon dar.

Die Firma wurde im Jahr 2006 gegründet und hat sich einzig auf 3D-Optimierung spezialisiert. Ihre Software wird von einem Großteil der heutigen Topentwickler verwendet und kam so in Spielen wie „The Witcher 3“, „Forza Horizon“ und „Hellblade“ zum Einsatz.

Das Programm besitzt einen hocheffektiven Reduzierungsalgorithmus, welcher nach Wichtigkeit der Merkmale eines Modells die Polygonzahl reduziert. Neben rein geometrischen Merkmalen wird ebenso der Einfluss der Optimierung auf UV-Koordinaten, Tangenten, Vertex-Normalen, Vertex-Color und das Skinning beachtet. Der Nutzer kann so abwiegen, welche Priorität die einzelnen Features bei der Polygonreduktion erhalten.

Zudem kann von dem Programm automatisch ein LOD-Kette für das Modell erzeugt werden, welche vom Nutzer direkt im Viewport betrachtet werden kann. LODs (Level of Detail) sind unterschiedliche Detailstufen eines Modells, die je nach Entfernung der Kamera zum Objekt ineinander überführt werden.

Für die Erstellung der LODs bestehen mehrere Möglichkeiten:

- Mesh-LODs optimieren die Anzahl von Polygonen in einem Asset,
- Material-LODs reduziert Zugriffe durch die Kombination mehrerer Materialien zu einem,
- Proxy-LODs reduzieren die Zugriffe und die Polygonanazahl durch Zusammenführung mehrerer Objekte und
- Bone-LODs reduzieren die Skelettkomplexität, welche viele Animations-CPU-Zyklen sparen.

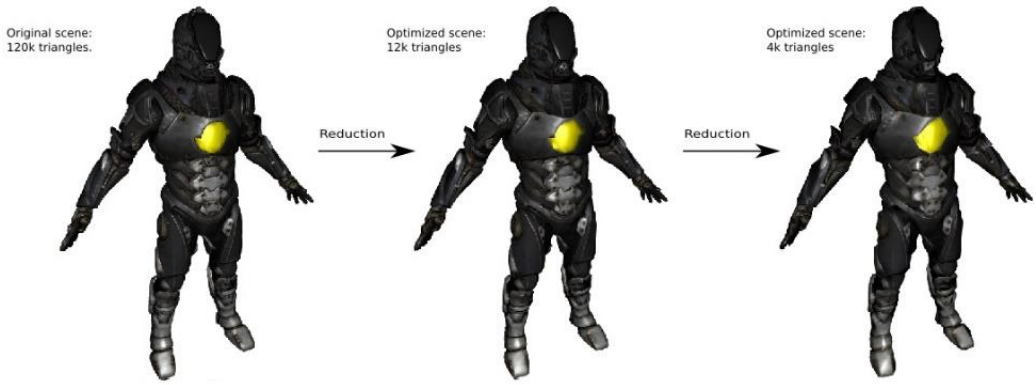

Abbildung 10: Reduktionsschritte in Simplygon (Simplygon 2017)

Mit diesen und vielen weiteren nützlichen Tools und Eigenschaften erleichtert die Software die Optimierung maßgeblich. So können nicht nur alle Schritte in einem einzigen kompakten Programm durchgeführt werden, Simplygon besitzt weiterhin Plugins in gängigen 3D-Programmen und ist mittlerweile fest in Game-Engines wie Unity und der Unreal Engine integriert (Simplygon 2017).

3.3.2 Texturbudget

Um den hohen Detailgrad der Objekte bei der Photogrammetrie einzufangen, ist es nötig die Texturen mit einer entsprechend hohen Auflösung aus der Point Cloud zu erzeugen. Bei einer Größe von 4096x4096 Pixeln entstehen meist schon sehr gute Resultate, jedoch ist durchaus genug Detail vorhanden auch größere Texturen im Bereich von 8 oder auch 16K zu extrahieren. Texturen dieser Größe sehen zwar umwerfend aus, bringen aber natürlich auch Nachteile mit sich, wenn sie in einem Spiel verwendet werden sollen.

Auch wenn sich die Speichermenge der Texturen in einem Spiel durch Fortschritt der Technologie immer weiter vergrößert, so ist sie heutzutage bei weitem noch nicht unendlich. Das Texturbudget für ein Spiel hängt neben der angestrebten Gesamtgröße auf dem Datenträger zu einem Großteil vom Speicher der Grafikkarte ab. Wird das Budget überschritten, resultiert dies im Abfallen der Framerate während die Texturen geladen werden.

Ein weiterer Faktor, der bei der Verwendung von solchen Riesentexturen berücksichtigt werden muss, ist die Texel Density des Modells. Monitore mit einer Auflösung von 1080p sind mittlerweile zum Standard geworden. Für Gamer die noch bessere Grafik anstreben existieren auch 4K-Monitore und Grafikkarten, die diese unterstützen. Jedoch selbst wenn auf 4K-Monitoren gespielt wird, besitzen die Photogrammetrie-Assets mit 4K- oder 8K-Texturen zum Großteil der Spielzeit eine viel höhere Pixeldichte, als der Monitor darstellen kann. Erst wenn der Spieler dem Asset so nah kommt, dass es nahezu den gesamten Bildschirm einnimmt, würde sich die Größe der Textur lohnen.

Eine gute Veranschaulichung dieses Problems bietet ein Screenshot aus der Entwicklung von SW: Battlefront. In dem Bild repräsentieren die weißen Kästchen eine 512 x 512-Textur und die roten Kästchen eine 32 x 32-Textur. Somit befinden sich 25 Textur-Pixel für jeden Bildschirm-Pixel in dem Bild (Brown et al. 2016).

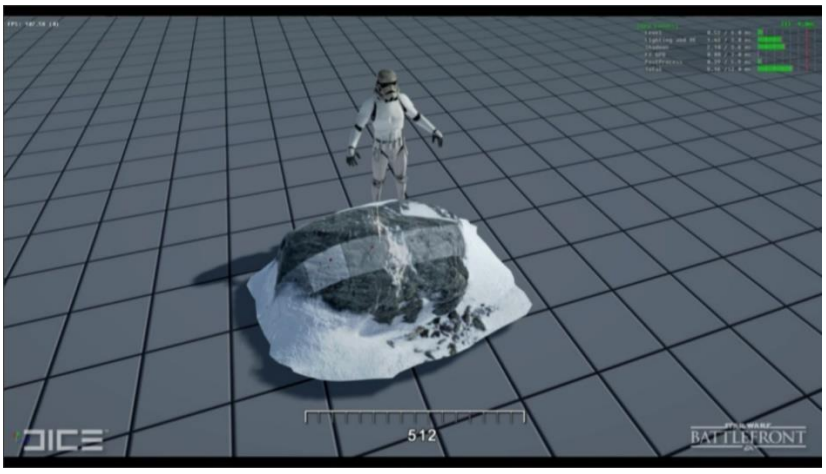

Abbildung 11: Texel Density-Test in SW: Battlefront (Brown et al. 2016)

Eine naheliegende Lösung dieses Problems wäre, die Auflösung der Texturen nach der Photogrammetrie-Rekonstruktion zu verringern. Damit werden zwar viele Details aus den Fotos zerstört doch sie können problemlos verwendet werden.

Für den entstehenden Informationsverlust können Detail Maps für die Assets erstellt werden. Detail Maps erlauben es, eine sekundäre Textur über das Objekt zu legen, zum Beispiel eine zweite Normal Map mit feineren Details. So hat das Asset auf kurze Entfernung ein hohes Maß an Detail, das verringert wird, wenn sich der Spieler entfernt. Ohne dieses Vorgehen müsste eine einzige Textur mit großer Auflösung sowohl für kurze als auch weite Entfernung zum Objekt verwendet werden.

Tatsächlich ist es häufig überraschend, mit wie viel weniger Auflösung damit im direkten Vergleich nahezu kein Unterschied sichtbar ist. Als Quelle für die Detail Maps können für diese Technik ebenso die photogrammetrischen Daten aus der Software verwendet werden. Somit wird das Detail aus den Fotos weniger zerstört als vielmehr mit anderen Mitteln wiederhergestellt.

Abbildung 12: Vergleich Felsen mit 4K und 1K Texturen (inkl. Detail Maps) (Brown et al. 2016)

Es sollte außerdem beachtet werden, welche Stellen eines Modells tatsächlich eine hohe Auflösung benötigen und welche nicht. Hohe Gebäude oder Bäume zum Beispiel können meist nur in den Abschnitten auf Augenhöhe genauer vom Spieler betrachtet werden. Somit werden auch nur in diesem Bereich Texturen mit guter Qualität benötigt. Ein fließender Übergang von photogrammetrischen Texturabschnitten hin zu gekachelten Texturen an entfernteren Stellen kann sich in solchen Fällen durchaus lohnen. Die benötigte Datenmenge für fotorealistische Texturen sollte nie an Stellen verschwendet werden, an denen das Detail nicht gesehen wird.

Für weitere Entfernungen müssen zwangsläufig Mipmaps von den Texturen erstellt werden. Eine Mipmap ist eine Sequenz, die mehrere Versionen der Originaltextur beinhaltet, welche stufenweise in ihrer Auflösung verkleinert wurden (Heckbert 1983). Diese Technik existiert schon seit Jahren, ist aber immer noch der gängigste Weg die Texturauflösung auf weiter Entfernung zur Kamera anzupassen bzw. zu verringern.

Ein zusätzlicher Weg, das Texturbudget zu optimieren, ist die Kompression der Texturen zu verbessern. Indem die Texturen komprimiert werden, kann die Menge an Speicher die ein Bild zum Laden benötigt deutlich verringert werden. Texturkompression ist ein komplexes Thema, dessen Verständnis ein gutes Programmierwissen voraussetzt. Die Details der verschiedenen Methoden würden den Umfang dieser Arbeit bei weitem sprengen, deshalb werden sie nur oberflächlich behandelt. Die objektive Bewertung von Texturkompressions-Methoden ist ähnlich schwierig, da neben der komprimierten Größe die verschiedenen negativen Auswirkungen auf die Qualität der Textur betrachtet werden müssen. Zusätzlich funktionieren die Methoden nicht gleich gut für jede Texturart.

DTX war jahrelang das Kompressionsverfahren, das von Spielentwicklern vorrangig verwendet wurde. Es besitzt in den neusten Versionen eine Kompressionsrate von 4:1, wird von einem Großteil der vorhandenen Grafikkarten unterstützt und die Ergebnisse besitzen eine solide Qualität (Renambot et al. 2007). Die Methode ist jedoch deutlich verlustbehaftet, was speziell bei der Verwendung für Normal Maps sichtbar wird.

Im Laufe der Zeit wurden daher immer fortgeschrittener Formate entwickelt wie PVRTC, ETC und ASTC, welche unterschiedliche Vor- und Nachteile besitzen. Die Entwickler von NVIDIA haben die gängigsten Formate bereits 2015 in einem Test verglichen. Das im Jahr 2012 eingeführte ASTC-Format erzielte, sowohl was die komprimierte Größe als auch die Qualität betrifft, die besten Ergebnisse (Nvidia Corporation 2015). Welches Kompressionsverfahren aber wirklich von Entwicklern verwendet wird unterschiedet sich trotzdem häufig von Fall zu Fall.

Ein weiteres Problem, mit dem sich Entwickler in diesem Zusammenhang auseinandersetzen müssen, ist die Art und Weise wie die Texturen geladen werden. Das klassische Vorgehen, alle Texturen am Anfang des Levels zu laden ist längst veraltet und wurde mit Texture Streaming Systemen ersetzt. „Textur Streaming ist der Prozess des kontinuierlichen Ladens von Texturdaten im Hintergrund, während eine Anwendung oder ein Spiel läuft. Der Streaming-Prozess wird vor dem Endbenutzer verborgen, es existieren keine Ladebildschirme. Ein Textur-Streaming-System ermöglicht die Verwendung von mehr Texturen in einer scheinbar kontinuierlichen 3D-Umgebung als die Menge, die in den Videospeicher passen würde." (Granite 2016)

Doch auch bei dieser Technik gibt es Unterschiede. Die erwähnten Mipmaps können durch Mipmap Streaming Systeme selbstständig geladen werden, je nachdem, wie weit sie von der Kamera entfernt sind. Dies entlastet den Speicher, indem nur die benötigten Mipmaps tatsächlich geladen werden, nicht die unnötigen Objekte im Hintergrund. Diese Systeme eignen sich für größere Auflösungen jedoch nicht besonders gut. Durch eine höhere Texturauflösung müssen auch größere Mipmaps geladen werden, was meistens eine deutliche Verzögerung nach sich zieht. Speziell wenn sich die Kamera nah an einem Objekt befindet und damit auf einmal eine 4 oder 8K Textur geladen werden muss, entsteht schnell ungewolltes Texture Popping (plötzlicher Wechsel von Texturen).

Die zurzeit fortschrittlichste Variante sind Tile-based Texture Streaming Systeme. Diese besitzen ebenfalls den Vorteil, dass nur die wirklich sichtbaren Texturen in den Videospeicher geladen werden. Sie nutzen dafür zusätzlich Virtual Texturing, eine Technologie, bei der die Texturen in kleine Einzelteile zerlegt werden, z.B. einzelne 128x128 Texturen. Durch dieses Vorgehen erlaubt die Technik die Verarbeitung von extrem hochauflösenden Texturen, welche anderenfalls nicht in den Videospeicher geladen werden könnten. Der Videospeicher ist üblicherweise auf 16K beschränkt, mit Virtual Texturing können theoretisch Texturen bis zu 256 K geladen werden.

Gleichzeitig wird durch das Vorgehen, immer nur einzelne Kacheln zu laden der Anspruch an den Videospeicher komplett von der verwendeten Texturauflösung getrennt. Die Verwendung von 2K oder 4K Texturen in einer Szene verbrauchen so dasselbe Maß an Speicher (Granite 2016).

Die vorgestellten Methoden bieten viele Möglichkeiten die schiere Datenmenge zu begrenzen, die photogrammetrische Texturen besitzen, sowie eine Überlastung des Videospeichers beim Laden zu verhindern. Die Programmierung und Implementierung von effektiven Kompressionsverfahren und Texture Streaming-Systemen stellt jedoch wiederum eine große Herausforderung dar.

Größere Spielunternehmen mit genug Ressourcen entwickeln häufig ihre eigenen Lösungen für diese Probleme. Sie besitzen sowohl eine Menge hochqualifizierter Programmierer als auch die Erfahrungen aus früheren Projekten, deren Techniken immer weiterentwickelt werden. Kleinere Unternehmen besitzen diese Vorteile meist nicht und müssen sich daher auf bereits vorhandene Lösungen verlassen.

Aufgrund dessen haben sich auch in diesem Bereich Firmen gegründet, die speziell Lösungen für die Optimierung des Texturbudgets entwickeln. Ein gutes Beispiel dafür ist die Firma Graphine, die mit der Granite SDK ein Toolkit geschaffen hat, welches von vielen Firmen für Photogrammetrie-Projekte genutzt wird. So findet es laut Entwickler Farm 51 Verwendung im noch zu erscheinenden „Get Even“, welches sich stark auf riesige photogrammetrische Texturen stützt (Granite 2014 [1]).

Das Software Development Kit ist spezialisiert auf optimale Texturkompression und effektives Tile-based Texturstreaming. Es erreicht Kompressionsraten von 60% und mehr, reduziert sowohl benötigten Speicher als auch Ladezeiten der Texturen um bis zu 75% und macht es so möglich eine deutlich größere Menge hochauflösender Texturen zu verwenden.

Das System funktioniert so gut, dass es mittlerweile ähnlich wie Simplygon in der Unity und Unreal Engine integriert ist. Für kommerzielle Verwendung wird ebenso eine Lizenz benötigt, welche in mehreren Varianten erworben werden kann. Die Anschaffung eines solchen Development Kits ist sehr kostenintensiv. Vor dem Hintergrund der Entwicklungskosten ist dieser Preis jedoch letztendlich gerechtfertigt. Gerade für kleinere Teams, die Photogrammetrie ernsthaft für ihre Spiele nutzen wollen, sind solche SDKs eine gute Alternative.

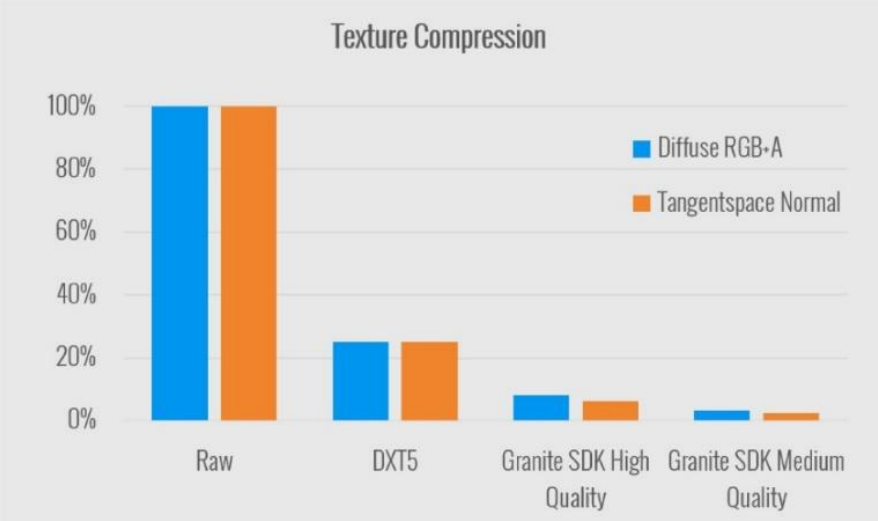

Abbildung 13: Granite SDK Kompression im Vergleich (Graphine 2014 [2])

3.4 Qualitätsunterschiede

Wenn Assets in einem Computerspiel mit Photogrammetrie erzeugt werden, setzen diese Objekte automatisch den Maßstab für die visuelle Qualität des Spiels. Werden ausschließlich solche Assets benutzt, entsteht kein großes Problem. Die Grafik des Spiels ist konsistent und sieht gut aus. Wie in Punkt 3.1 besprochen gibt es jedoch eine Vielzahl von Objekten, die Probleme bei der Photogrammetrie verursachen und nicht jedes dieser Probleme kann gelöst werden.

Aus diesem Grund lässt es sich kaum vermeiden auch Assets, die mit traditionellem Modeling und Texturing erstellt wurden, in einem Spiel zu verwenden. Besonders bei anorganischen Objekte mit harter Oberfläche ist dies häufig der Fall. Doch genau in diesem Punkt entstehen Schwierigkeiten. Im direkten Vergleich werden die Qualitätsunterschiede zu schnell sichtbar und traditionell erstellte Assets sehen um ein vielfaches schlechter aus als in einem Spiel, in dem ausschließlich herkömmliches Modeling verwendet wurde.

Teilweise kann dieses Problem durch mehr Zeit und Aufwand für die traditionellen Modelle behoben werden. Ein deutlicher Unterschied bleibt aber meist trotzdem bestehen. Es ist jedoch möglich die gesamte visuelle Qualität durch realistisches Rendering zu steigern und den Unterschied weniger offensichtlich zu gestalten.

Ein großer Schritt in Richtung wirklich realistischem Rendering ist die Entwicklung und Implementierung von Physically Based Rendering (kurz PBR). PBR ist erneut ein sehr umfangreiches Thema, sowohl was Theorie als auch Umsetzung angeht. Die folgenden Abschnitte sind daher nur eine Zusammenfassung der Technologie und wie deren Verwendung die visuelle Qualität aller Assets und den allgemeinen Realismus in einem Spiel erhöht.

Physically Based Rendering stützt sich auf tatsächliche physikalische Prinzipien um die Interaktion von 3D-Modellen mit Licht akkurater darzustellen und so die Realität zu imitieren. Es unterscheidet sich dabei deutlich von nicht-fotorealistischem Rendering, das häufig in Spielen mit künstlerischem Stil verwendet wird und interaktivem Rendering, bei dem die Performance die höchste Priorität besitzt. Da es sich bei PBR aber vielmehr um ein Konzept als um strikte Vorgaben handelt kann es bei der Implementierung Unterschiede geben (Pharr et al. 2010).

Aus der Sicht von Game Artists beinhaltet PBR vorrangig zwei Dinge: weiter entwickelte Shader und eine neue Art und Weise Texturen zu erstellen. Die Shader haben erweiterte Inputs welche deutlich feinere Abstufungen bei der Materialerstellung erlauben. Sie beruhen auf physikalischen Eigenschaften wie:

- Reflektion (wieviel Prozent des einfallenden Lichts wird reflektiert, reicht von diffus bis zu spiegelnd oder metallisch),
- Lichtdurchlässigkeit (Licht kann das Objekt teilweise durchqueren),
- Transparenz (Licht kann das Objekt komplett durchqueren),
- Energiekonservation (Objekte können nicht mehr Licht reflektieren als sie aufnehmen),
- Mikrooberfläche (wie rau oder glatt ist die Oberfläche auf mikroskopischen Level) und
- Fresnel (Reflektionen abhängig vom Winkel zwischen Sehstrahlen und Objektoberfläche) (Russell 2015)

Durch Berücksichtigung dieser Eigenschaften besitzt nahezu jedes Material ein gewissen Grad an Reflektion und Fresnel, auch wenn nicht sofort sichtbar. Der Unterschied wird erst deutlich, wenn sich die Umgebung um das Objekt herum verändert. Die komplexen Materialien erlauben es, das Assets Farben aus der Umgebung aufnehmen, welche sich in der Textur wiederspiegeln. Das entstehende Bild wirkt sehr viel natürlicher und einheitlich.

Abbildung 14: Asset mit Physically Based Rendering (Pettit 2015)

Die Texturen müssen für diese Shader entsprechend angepasst werden. Für die traditionelle Erstellung von Texturen wurden üblicherweise eine Diffuse Map, eine Normal Map und eine Specular Map gewählt. Es bestanden noch weitere Möglichkeiten, wie Ambient Occlusion oder Cavity Maps, häufig wurden diese Informationen aber bereits mit der Diffuse Map gebakt um Speicherplatz zu sparen oder weil die Shader diese Texturen nicht unterstützten. In diesem Punkt unterscheidet sich die Texturerstellung mit einem PBR-Workflow am meisten. Durch die akkurate Beleuchtung in der Szene müssen keine Lichtinformationen in die Textur eingearbeitet werden. Cavity und Ambient Occlusion werden aus der Diffuse Map entfernt und separat erstellt. Die Diffuse Map (für PBR auch häufig Albedo Map genannt) wird so auf einen akkuraten Farbwert ohne Baked Lighting aufgehellt. Die reflektierenden Eigenschaften werden in eine Specular Map für Highlights und eine Gloss Map für die Oberflächenvariationen aufgeteilt (Wilson 2015).

Für Gegenstände aus Metall können Spec und Gloss Map auch durch eine Metallness Map ersetzt werden. Für diesen Workflow muss wiederum die Diffuse Map angepasst werden, da Metalle genau genommen keine diffusen Eigenschaften besitzen. Sowohl der Metallness als auch der Specular Workflow haben Vor- und Nachteile. Häufig unterstützen PBR-Shader aber nur eine dieser beiden Möglichkeiten.

PBR ist mittlerweile in den meisten Game-Engines implementiert. So muss häufig nur der neue Workflow adaptiert werden, um das Rendering zu ermöglichen. Das Erlernen von neuen Techniken kostet immer eine gewisse Zeit. Die Vorteile, die PBR bringt sind jedoch essentiell, vor allem bei Photogrammetrie-Projekten. Neben der höheren Qualität ist einer der größten Vorteile die Einheitlichkeit des Renderings. Durch das Vorhandensein konsistenter Basis-Materialien gibt es weniger Unterschiede bei der Materialerstellung, wenn mehrere Artists am Projekt mitwirken. Zudem machen es die neuen Shader möglich, dass Modelle in wirklich jeder Lichtsituation gleich gut aussehen. Auch traditionell erstellte Assets sehen so durch die konsistente und akkurate Beleuchtung realistisch aus, sodass Unterschiede zu photogrammetrischen Assets nicht so stark hervorstechen.

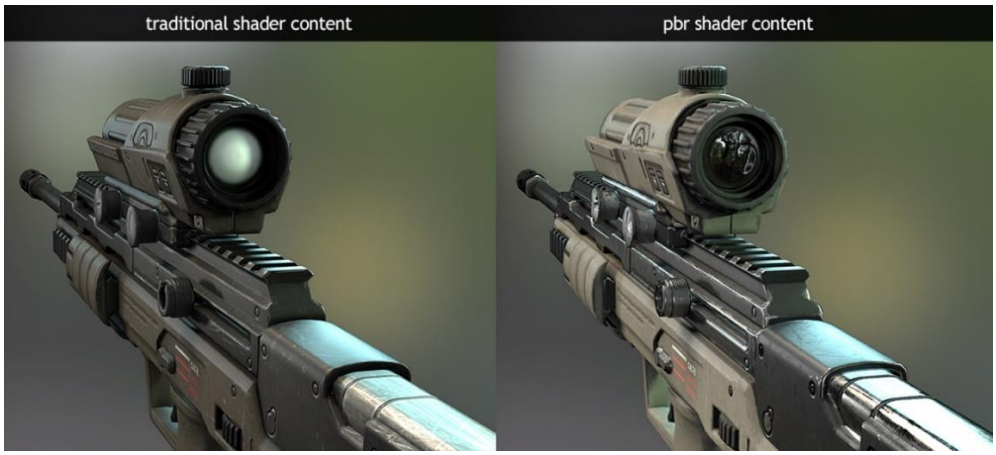

Abbildung 15: Traditionelle und PBR-Shader im Vergleich (Wilson 2015)

Es sollte trotzdem beachtet werden, dass Photogrammetrie in einem Computerspiel niemals halbherzig verwendet werden sollte. Entscheiden sich Entwickler für die Technik, so sollte dies möglichst früh in der Produktion geschehen und nicht nachträglich wenn alle anderen Stricke gerissen sind. Der fotorealistische Effekt, der erzielt werden soll, funktioniert schließlich nur, wenn alle grafischen Aspekte auf demselben Level sind. Photogrammetrie birgt eine gewisse Verpflichtung an die Grafik und es sollten alle Aspekte berücksichtig werden bevor diese eingegangen wird (Willson 2017).

Traditionelles Modeling und Texturing sollten dementsprechend nur als letzter Ausweg gewählt werden. Selbst in diesen Fällen empfiehlt es sich, stark von Fotos und gescannten Objekten Gebrauch zu machen. Gescannte Objekte können immer noch als exzellente Referenz oder Basis verwendet werden, selbst wenn nur einzelne Aspekte von ihnen in dem zu erstellenden Asset auftauchen sollen. Möglicherweise reicht bereits die Textur oder einzelne Abschnitte der Geometrie eines Scans um einem anderen Modell einen fotorealistischen Charakter zu verleihen. Bei der Verwendung von Photogrammetrie muss nicht jedes fotografierte Objekt 1:1 im Computer rekonstruiert werden. Durch kreatives Wiederverwenden von den gescannten Daten können komplett neue Assets erschaffen werden, die so in der Realität nicht existieren aber trotzdem echt aussehen.

Die Texturen sind dabei um eine Vielfaches wichtiger als die Geometrie des Modells. (Long 2010, S.1) Selbst ein sehr simples, manuell erstelltes Modell kann kombiniert mit einer Fototextur bereits ausreichen, um den Spieler zu täuschen. Das ist in dem Zusammenhang auch einer der wichtigsten Gedanken für eine konsistente Grafik. Den Spieler interessiert nicht, mit welcher Technik gearbeitet wurde oder mit welchen Tricks Fehler kaschiert werden, solange die Illusion des Spiels nicht gebrochen wird. Dieser Effekt nennt sich Immersion oder auch Präsenz. Der Spieler nimmt die Spielwelt als Wirklichkeit wahr ohne sie zu hinterfragen (Madigan 2010).

3.5 Leveldesign

Leveldesign ist definiert als Prozess die räumlichen Gegebenheiten eines Spiels zu gestalten und festzulegen. Es ist eng verwoben mit dem Game Design, das die Inhalte und Regeln eines Spiels beinhaltet (Kremers 2009, S. 12).

Gutes Leveldesign beruht auf mehreren Prinzipien, die darauf ausgerichtet sind, was ein Spieler in einem Spiel sucht und was er vom Spiel erwartet. Es gibt sehr unterschiedliche Auslegungen welche Regeln tatsächlich beachtet werden müssen, um ein gutes Level zu erstellen. Die meisten davon lassen sich jedoch einem der folgenden fünf Prinzipien zuordnen:

- **Unterhaltung:** Das Spiel ist herausfordernd, überraschend, abwechslungsreich, nicht frustrierend, etc.
- **Storytelling:** Alles im Spiel erzählt eine Geschichte
- **Selbsterklärende Bedienung:** Das Spiel ist nicht auf Worte angewiesen um etwas zu erklären, sagt dem Spieler was er zu tun hat aber nicht wie
- **Immersion:** Der Spieler akzeptiert die virtuelle Welt als Realität
- **Effizienz:** Die Ressourcen des Spiels werden sinnvoll und effizient verwendet

Jeder dieser Punkte bedarf einer ausführlicheren Erklärung. Für den Zweck dieser Arbeit sind aber nur die beiden letzten Punkte wichtig. Die gesteigerte Immersion ist einer der Hauptgründe für die Verwendung von Photogrammetrie, wie bereits in Punkt 2.2.1 erklärt wurde. Der Realismus, der mit der Technik möglich wird, lässt die Illusion der virtuellen Welt nicht so leicht zerbrechen, zumindest auf grafischer Seite. Wird der Prozess fachgerecht durchgeführt, entstehen in diesem Punkt auch keine Probleme.

In Zusammenhang mit der Effizienz des Leveldesigns kann Photogrammetrie jedoch Schwierigkeiten verursachen. Die gängige Vorgehensweise in der heutigen Spielproduktion ist die Verwendung von modularen Bestandteilen (auch Prefabs genannt) zum Gestalten der Levels. Durch modulares Level Design ist es möglich geworden, nicht jedes kleinste Teil eines Levels einzeln modellieren und texturieren zu müssen. Die Bestandteile des Levels werden stattdessen in einem frühen Stadium der Entwicklung mit vorgefertigter Geometrie ausgeblockt, die später als Basis für die finalen Assets dient. Diese simplen geometrischen Bestandteile besitzen bereits die richtige Größe, sodass sie sich auf einem Gitterraster lückenlos zu einem kompletten Level wie Legosteine zusammenfügen lassen (Perry 2002).

Dieses Vorgehen hat mehrere Vorteile, unter anderem:

- Die Iteration für die Levels geht sehr viel schneller von statten
- Die Level können problemlos jederzeit verändert werden
- Es muss nur ein Bruchteil der Assets erstellt werden, die sich immer wiederverwenden lassen
- Leveldesign und Asseterstellung kann zeitlich unabhängig voneinander behandelt werden
- Speicher wird entlastet durch das Verwenden von mehreren Instanzen desselben Objektes
- Modellierer müssen sich nicht um den Maßstab ihrer Modelle kümmern
- Die Level sehen konsistent aus und haben einen einheitlichen Stil

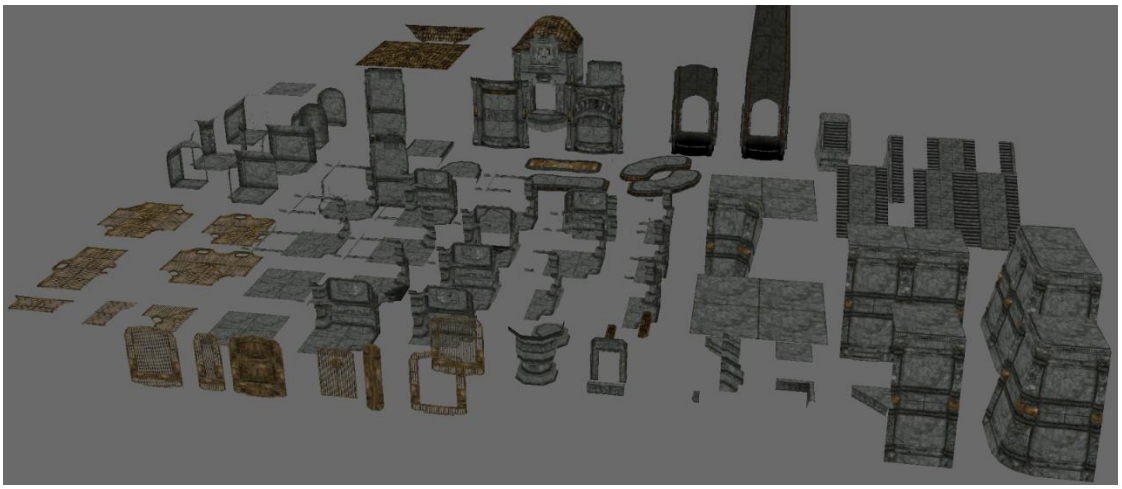

Abbildung 16: Modulares Level Design Kit für TES: Skyrim (Burgess 2013)

Die häufige Wiederverwendung von Assets, die damit einhergeht, hat aber ebenso Nachteile. Durch die riesige visuelle Vielfalt, die der Spieler von der Natur gewohnt ist, kann eine zu offensichtliche Wiederholung von Elementen die Immersion des Spiels schnell zerstören. Im Fall von photogrammetrischen Assets ist dieses Problem besonders prägnant. Assets, die mit der Technik erzeugt werden, besitzen ein so hohes Maß an einzigartigen Details, dass bereits eine geringe Wiederverwendung häufig sofort auffällt. Spielentwickler müssen sich somit genau überlegen, welches Maß an Modularität und Fotorealismus sie für ihre Level wählen.

Soll die Modularität für ein Spiel trotz Photogrammetrie beibehalten werden, dann muss dies auf eine Art und Weise geschehen, die diese Art des Level Design unterstützt. Assets eines typischen Level Design Kits können in drei Kategorien eingeteilt werden:

- Helden-Assets: einzigartig, werden in kleiner Anzahl im Level eingesetzt
- Nutzbarer Kern: generisch, werden für Großteil des Levels verwendet, z.B. Wände, Säulen, Gänge
- Varianten: Alternative Versionen des Nutzbaren Kerns je nach Bedarf

Photogrammetrie eignet sich am besten für Helden-Assets, da die fotorealistische Einzigartigkeit deren Zweck unterstützt. Das Problem ist, das ohne Bearbeitung der gescannten Daten jedes Objekt zu einem Helden-Asset wird. Es muss somit genau darauf geachtet werden, welches fotografierte Objekt später welchen Platz im Spiel haben soll. Der Großteil des Level Kits muss trotzdem generisch bleiben um nicht repetitiv zu wirken. Für Kern-Assets sollte demnach darauf geachtet werden, dass die fotografierten Objekte sowohl was Textur als auch Geometrie angeht keine auffallenden Merkmale besitzen.

Zudem sollte Photogrammetrie für modulare Kits seltener zur kompletten Reproduktion von realen Objekten benutzt werden. Die photogrammetrischen Daten sollten stattdessen vielmehr als Unterstützung für das Modeling dienen. Das Erstellen von Varianten sollte auch eine höhere Priorität erhalten.

Der Workflow zur Verwendung von photogrammetrischen Daten für modulare Assets könnte wie folgt aussehen. Zuerst wird ein Lowpoly-Modell erstellt, entweder durch Reduktion des photogrammetrischen Highpoly-Modells oder durch manuelles Modellieren mit Referenzieren des Scans. Danach wird das Lowpoly-Modell generisch texturiert, z.B. mit einer unscharfen Textur aus dem Scan. Aus den gescannten Daten werden individuelle Texturen für Farbe, Rost, Dreck, Witterung etc. erstellt. Die einzelnen Texture Maps können dann mit Masken in das generische Lowpoly-Modell eingearbeitet werden um mehrere Varianten des Assets zu erstellen. Um den realistischen Charakter zu bewahren, sollte dabei eng mit der Referenz des Scans gearbeitet werden. So können einfach mehrere Instanzen des Assets erstellt werden, die alle auf demselben fotografierten Objekt basieren (Willson 2017). Bereits eine überschaubare Anzahl solcher Varianten kann ausreichen um den repetitiven Charakter aufzubrechen.

Entwickler wie Bethesda machen für ihre Spiele extensiven Gebrauch von modularen Level Design Kits. Dies muss aber nicht immer der Fall sein. Speziell bei Produktionen mit großem Budget und Fokus auf realistischer Grafik kann der Einsatz von modularen Level Kits, je nach der gewünschten visuellen Einzigartigkeit des Levels, durchaus variieren. Die Wiederverwendung gewisser Kernassets wie Zäune, Bäume, Mauern etc. macht nahezu immer Sinn, da diese mehr zum Auffüllen des Levels dienen, als alles andere. Soll aber für ein Spiel eine große Anzahl an komplett einzigartigen Levelabschnitten produziert werden, kann es teilweise günstiger sein, den gesamten Abschnitt als größeres statisches Mesh zu modellieren und später zu zerlegen als dafür ein separates Level Kit zu erstellen.

Theoretisch besteht sogar die Möglichkeit, ein Spiel komplett ohne modulare Bauweise zu erstellen. Sofern in der echten Welt vorhanden, wäre es möglich mit Photogrammetrie Orte aus der echten Welt 1:1 ins Spiel zu übertragen und als Level zu nutzen.

Dieses Vorgehen würde das Level Design jedoch stark einschränken. Die realen Schauplätze würden den Großteil des Level Layouts bestimmen und der Level Designer müsste um die vorhandenen Gegebenheiten herum arbeiten. Die Freiheit und Flexibilität beim Erstellen der Levels würde somit größtenteils verloren gehen. Ganz zu schweigen von den Datenmengen, die dabei entstehen würden.

3.6 Alternativen

Photogrammetrie ist längst nicht die einzige Möglichkeit, reale Objekte zu scannen und ins Spiel zu bringen. Es gibt eine Vielzahl anderer Techniken, die gute Resultate ermöglichen und sich in manchen Fällen sogar besser eignen. Dieses Kapitel soll einen groben Überblick über diese Alternativen geben.

3D-Scanner werden sehr häufig als Alternative oder unterstützend zur Photogrammetrie verwendet. Diese Scanner existieren in einer Vielzahl verschiedener Ausführungen. Sie werden üblicherweise nach ihrer Reichweite und der verwendeten Scan-Technologie unterteilt. Die Technologie der meisten 3D-Scanner basiert entweder auf Lasern oder auf Strukturlicht (auch projiziertes Licht genannt).

Laserbasierte 3D-Scanner arbeiten - wie der Name bereits vermuten lässt - mit Laserstrahlen, die auf das Zielobjekt geschossen werden und die Zeit aufnehmen, welche die Reflektion der Strahlen braucht, um zurück zum Sensor zu gelangen. Durch die bekannte Distanz des Sensors zur Laserquelle können akkurate Messungen basierend auf dem Reflexionswinkel des Lasers erstellt werden (Kleinkemper 2014).

Einige Scanner nehmen bei diesem Prozess auch Farbinformationen auf, diese sind für die Spielentwicklung um einiges interessanter. Aus den gescannten Daten wird genau wie bei der Photogrammetrie eine Point Cloud erzeugt, die dann in ein nutzbares 3D-Modell umgewandelt werden kann.

3D-Scanner, die mit Lasern arbeiten, haben folgende Vorteile: (Leonova 2014)

- Weniger Fehler durch mögliche Automatisierung
- Hohe Genauigkeit
- Hohe Reichweite möglich

Strukturlicht-Scanner arbeiten hingegen mit einer Projektionsquelle und einem oder mehreren Sensoren. Auf das Objekt wird zuerst ein Muster projiziert (Streifen, Blöcke etc.). Die Sensoren beobachten die Verformung und Grenzen der Muster und bestimmen so die dreidimensionale Form des Objektes. Es wird dabei dieselbe Triangulation zwischen Lichtquelle und Sensor verwendet wie bei laserbasierten Scannern (Engineering & Manufacturing Services Inc. 2015).

Strukturlicht-3D-Scanner besitzen folgende Vorteile: (Engineering & Manufacturing Services Inc. 2015)

- Sehr schnelle Scanzeiten (bis zu 2 Sekunden)
- Hohe Auflösung und Genauigkeit
- Sicher für das menschliche Auge

Bezogen auf die Reichweite werden 3D-Scanner eingeteilt in Kurzstrecken- und Mittel bis Langstrecken-Scanner.

Kurzstrecken-Scanner existieren sowohl mit Laser als auch Strukturlicht und können häufig ohne Gestell mit einer Hand bedient werden. Sie sind dementsprechend sehr leicht, schnell und vielseitig.

Ein Beispiel für ein hochqualitatives Modell mit Strukturlicht ist der Artec Eva 3D Handscanner. Der Scanner wird um das Objekt geführt während es mit hoher Geschwindigkeit beleuchtet und gescannt wird. Die Bilder werden währenddessen in Echtzeit mit der Software zusammengeführt. Die Daten können in der Software sowohl automatisch als auch manuell bearbeitet werden, falls mehr Kontrolle verlangt wird. Die entstehenden 3D-Modelle sind sehr genau, hochauflösend und besitzen eine qualitativ hochwertige Textur. Durch die mögliche Konfiguration können sowohl glänzende als auch reflektierende Objekte gescannt werden. Die Scanner eignen sich besonders für sehr kleine Objekte, mittelgroße Objekte sind aber auch möglich. (Artec Europe 2017)

Eine vergleichbare Variante, die statt Strukturlicht Laser verwendet, ist der Sense 3D-Scanner von 3D Systems.

Abbildung 17: Artec Eva 3D-Handscanner (Artec Europe 2016)

Mittelstrecken-Scanner können sehr nützlich sein, da sie sowohl kleine als auch mittelgroße Objekte gut abdecken. Ein gutes Beispiel sind David Laser Scanner. David Laser Scanning ist eine kostengünstige Eigenbau-Alternative, die ohne spezialisierte Hardware auskommt. Die Technologie wurde kürzlich von HP erworben und vermarktet. Die David 3D-Scanner bestehen aus einer hochqualitativen HDMI Kamera und einem Videoprojektor auf einem Stativ. Diese Bestandteile werden mit dem restlichen Zubehör mit einem Rechner verbunden, auf dem sich die zugehörige Software befindet.

Die Scan Technologie basiert ebenfalls auf Strukturlicht. Mit dem Videoprojektor wird ein schwarz-weißes Muster auf das Objekt projiziert. Diese Informationen werden mit der Kamera aufgenommen und in Echtzeit von der Software verarbeitet. In einer erweiterten Version kann die Kameraaufnahme auch in Stereo mit zwei Kameras geschehen. Mit dem richtigen Computer können Objekte bereits innerhalb von 2 Sekunden gescannt werden und von der Software in 3D-Modelle mit über 2 Millionen Vertices umgewandelt werden (HP Development Company,L.P. 2017).

Abbildung 18: David SLS-3 3D-Scanner (HP Development Company, L.P. 2017)

Mittel- und Langstreckenscanner verwenden ausschließlich Laser, da Strukturlicht nicht die nötige Reichweite besitzt. Diese Scanner nehmen Millionen von Punkten auf, indem sie sich um 360 Grad drehen, während Spiegel die Laser in Richtung des Objektes leiten. Sie lassen sich in Phasenverschiebungs-Scanner und pulsbasierte Scanner unterteilen. Phasenverschiebungs-Scanner eignen sich dabei auch für mittellange Strecken bzw. mittelgroße Objekte wie Kraftfahrzeuge.

Die Vorteile dieser Scanner sind: (Engineering & Manufacturing Services Inc. 2015)

- Aufnahme riesiger Datenmengen
- Aufnahmebereich bis zu 1000 Meter
- Sehr hohe Auflösung und Genauigkeit

Obwohl solche Langstrecken-Scanner sehr gute Ergebnisse ermöglichen und theoretisch für Computerspiele genutzt werden können, sind sie für industrielle Zwecke doch um einiges gebräuchlicher. Sollen wirkliche große Objekte wie Gebäude oder ganze Landschaften gescannt werden, wird für Computerspiele häufiger ein Mittelstrecken-Scanner an einer Drohne befestigt, um das Gebiet zu scannen. Gründe dafür sind vor allem die geringeren Kosten und die größere Flexibilität.

Erwähnenswert als weitere Alternative, abgesehen von den verschiedenen Arten der professionellen 3D-Scanner, ist im Zusammenhang der Spielentwicklung noch das Scannen mit einer Microsoft Kinect.

Mit dem Sensor der Kinect können ebenso Objekte gescannt und in 3D-Modelle verwandelt werden. Die Auflösung ist jedoch in der derzeitigen Version Kinect V2 sehr beschränkt und kommt den Ergebnissen der anderen Möglichkeiten nicht wirklich nahe. Microsoft hat zudem bekanntgegeben das zurzeit keine Spiele mit Kinect-Unterstützung entwickelt werden (Plante 2016). Es ist somit ungewiss, ob in Zukunft weitere Versionen der Kinect mit besseren Scan-Eigenschaften erscheinen.

4 Leitfaden

Planung des Levels

Für den erfolgreichen Einsatz von Photogrammetrie in einem Computerspiel muss die Planung an erster Stelle stehen. Die Level sollten bereits ordentlich ausgeblockt sein und es sollte genau feststehen, welche Assets aufgenommen werden müssen. Die folgenden Fragen müssen zu diesem Zeitpunkt beantwortet werden:

- Welche Objekte werden als Helden Asset verwendet?
- Welche Objekte werden als nutzbarer Kern verwendet?
- Welche und wie viele Varianten müssen erstellt werden?
- Welche Objekte eigenen sich nicht für Photogrammetrie und wie wird mit diesen umgegangen?
- Welche Schauplätze müssen für welche Objekte besucht werden?

Diese Einteilung vereinfacht die Wahl der späteren Fotomotive deutlich. So wird verhindert, dass für Helden-Assets Fotos von generischen Objekten genommen werden und für Assets des nutzbaren Kerns nicht komplett einzigartige Objekte fotografiert werden. Es können genug Referenzbilder für Varianten genommen und bereits Maßnahmen für ungeeignete Objekte getroffen werden. Die Wahl des Ortes ist besonders wichtig bei eindeutig erkennbaren geografischen Merkmalen. Sollten sich Objekte nicht für die Technik eignen, müssen Alternativen wie 3D-Scanner verwendet werden oder die Objekte mit Referenzieren des Scans oder Fotos manuell modelliert werden.

Optimale Fotos

Die Einhaltung folgender Punkte bei der Aufnahme garantiert die besten Resultate:

- Kamera mit über 5 Megapixeln, Objektiv mit fester Brennweite (z.B. 50mm) auf Stativ mit Fernsteuerung
- Blendenzahl vergrößern, ISO-Wert verkleinern, kurze Belichtungszeit
- Format: RAW,16-Bit
- Scharfe Fotos im Abstand von 10-15 Grad auf horizontalen und vertikalen Achse, Überlappung der Fotos: 50-60%

- Objekte mit Transparenz, Reflektionen, gleichmäßigen Oberflächen, repetitiven Merkmalen, dünnen oder überlappenden Elementen vermeiden oder präparieren

- Konstante, indirekte Beleuchtung ohne tiefe Schatten (Wetterverhältnisse abpassen oder optimale Beleuchtung im Studio sicherstellen)

Workflow Asseterstellung

1. Exportieren des Highpoly-Modells und der Textur aus Photogrammetrie-Software (Auflösung so hoch wie möglich)

2. Retopology/Erstellung des Lowpoly-Modells

3. Sauberes UV-Layout (so wenig wie möglich Verzerrung für spätere Detail Maps)

4. Texture Maps auf Basis der Photogrammetrie-Textur generieren (AO, Cavity, Heightmap, Specular, Tangent Normal, Object Normal)

5. Baked Lighting aus Diffuse Map entfernen, Texturen dem gewählten PBR-Workflow anpassen

Helden-Assets können mit diesem Workflow ohne weitere Bearbeitung erstellt werden. Für den Nutzbaren Kern sollte bereits in den ersten beiden Schritten auf einzigartige Merkmale geachtet werden, die besser manuell entfernt werden. Texturmerkmale können durch Stichproben von anderen Stellen des Modells ersetzt werden und Geometriemerkmale werden bei der Retopologisierung angepasst. Für die Erstellung von Varianten kann die finale Diffuse Map in Photoshop in separate Texturen aufgeteilt werden, die mit Masken in die Modelle eingearbeitet werden. Alle anderen Texturen müssen auf dieser Grundlage angepasst oder neu generiert werden. Alternativ können die gescannten Daten auch bloß als Vorlage genutzt werden und die Varianten manuell erstellt werden.

Um sicherzustellen, dass das Spiel trotz photogrammetrischer Assets später flüssig läuft, sollten weitere Maßnahmen getroffen werden. Für die erstellten Modelle bedeutet das:

- Auflösung der Texturen entsprechend der Texel Density verringern

- Detail Maps erstellen

- mindestens 3 LODs für jedes Asset erstellen

Engine Import

Die Modelle und Texturen können schließlich in die Engine importiert und zu Prefabs des Level Kits zusammengestellt werden. Nach dem Import sind dafür folgende Schritte zu absolvieren:

- PBR-Shader erstellen
- Shader mit Texturen ausstatten, je nach gewähltem Workflow (Specular oder Metallness)
- Detail Texturing hinzufügen und Eigenschaften festlegen (Größe, Distanz, Intensität etc.)
- LOD-Kette etablieren

Das Blockout des Levels kann nun mit den Prefabs ersetzt werden. Dabei sollten die Gegenstände immer wieder zusammen mit der Szene ausbalanciert werden. Wenn sich alle Elemente im Spiel befinden kann objektiv abgeschätzt werden, ob zu viele wiederholende Elemente im Level eingesetzt wurden und ob die PBR-Shader die richtigen Einstellungen besitzen, um ein einheitliches Bild zu erzeugen.

Sollte sich herausstellen, dass trotz der getroffenen Maßnahmen zu viel Wiederholung im Level herrscht, so müssen die Assets weiterbearbeitet werden. Mögliche Lösungen können sein:

- Filter über Texturen legen (z.B. Farbton verschieben)
- Mehrere Shader für verschiedenen Varianten erstellen
- Weitere Texturen mit Masken einarbeiten (Textur Blending)

Diese Schritte werden solange durchgeführt bis keine sichtbare Repetition mehr im Level herrscht.

Tabellarische Zusammenfassung

Problem	Lösungsmöglichkeiten	Nachteile/Hindernisse
Unscharfe Bilder	Bewegung in der Szene vermeiden Stativ und Fernsteuerung verwenden	keine
Tiefenschärfe	Blendenzahl vergrößern Stativ verwenden	keine
JPEG Kompression	Bilder in RAW-Format	keine
Transparente und reflektierende Objekte	Polarisationsfilter verwenden Beschichtung für matte Oberflächen	Für starke Reflektionen ungeeignet Beschichtung eignet sich nicht für jedes Objekt
Repetitive Merkmale	keine	keine
Gleichmäßige Oberflächen	Merkmale schaffen z.B. durch Aufkleber	Hinzugefügte Merkmale erscheinen im finalen Modell
Dünne Objekte	keine	keine
Überlappung	Elemente einzeln fotografieren und rekonstruieren	keine
Wechselnde Lichtverhältnisse	Wetterbedingungen genau abpassen Studioaufnahme	Wetter lässt sich nicht beeinflussen, ist nie 100% konstant nicht für alle Objekte geeignet
Tiefe Schatten	Bewölkter Himmel Indirekte Studiobeleuchtung Sekundäre Lichtquellen (Lightbox)	keine
Über- und Unterbelichtung	Stärke der Lichtquelle genau justieren	Natürliche Lichtverhältnisse lassen sich nicht anpassen
Baked Lighting	Highlightentfernung in Photoshop HDR/Grey-Chrome Ball Setup Camera RAW Plugin in Photoshop Kanaltrennung mit Normal Map	vom Können des Benutzers abhängig keine Effizienz kann variieren vom Können des Benutzers abhängig
Meshoptimierung	Manuelle Optimierung Automatische Optimierung mit eventueller Nachbearbeitung	vom Können des Benutzers abhängig von Effizienz des Algorithmus abhängig
Optimierung Texturebudget	Auflösung der Texturen verringern Verwendung von Detail Maps Kompression maximieren	keine
Qualitätsunterschiede	Physically Based Rendering Kaschierung durch Fototexturen/ Wiederverwendung gescanter Daten	keine
Leveldesign	Detailmaß an Art des Assets orientieren Zerlegung und Kombination der gescanten Daten	keine

5 Diskussion

Die Untersuchungen haben gezeigt, dass bei der Verwendung von Photogrammetrie für Computerspiele eine Vielzahl von Aspekten beachtet werden muss. Teilweise überschneiden sich diese mit den anderen Verwendungsmöglichkeiten der Technik, für Computerspiele kommen jedoch auch viele individuelle Herausforderungen hinzu.

Die rein fotografischen Probleme sind so recht universell und nicht spezifisch für diese eine Verwendungsart. Es müssen bei der Verwendung für Computerspiele dieselben Regeln für hochwertige Fotos eingehalten werden wie zum Beispiel bei der Verwendung für Denkmalspflege. Für diesen Bereich waren dementsprechend genügend Ressourcen vorhanden, um die Herausforderungen sowie deren Lösungen zu ergründen. Auch wenn einige Aspekte, wie zum Beispiel die vorhandene Lichtsituation durch ihre Unberechenbarkeit nicht gänzlich gelöst werden konnten, stellen die Fotografien keine größeren Probleme für die Verwendung der Technik dar.

Größere Probleme machen die fotografierten Objekte selbst. Den Materialeigenschaften muss nach wie vor große Aufmerksamkeit geschenkt werden. Wie sich gezeigt hat, wurden von Spielentwicklern bereits einige Verfahren entwickelt, mit denen sich ein Teil der problematischen Objekte dennoch rekonstruieren lassen. Es existieren jedoch genauso viele Fälle, in denen das noch nicht erreicht werden kann. Um diese Gegenstände zu bewältigen, ist die Nachbearbeitung im Computer wohl der vielversprechendste Ansatz. Einige spezielle Präparationsmethoden für die Gegenstände können zwar ebenso funktionieren, diese sind aber äußerst situationsabhängig. Das Experimentieren mit weiteren Präparationsverfahren, die sich besser und in mehreren Situationen eignen, könnte sich daher lohnen. Zusätzlich könnten die vorhandenen Algorithmen der Photogrammetrie-Programme weiter untersucht und verbessert werden.

Die Probleme bei der tatsächlichen Asseterstellung entstehen vorrangig durch die großen Datenmengen, die bei der Technik entstehen. Die Optimierung des gesamten Assets, sowohl was Mesh, Textur und Material angeht, ist somit ein sehr wichtiger Punkt. Da aber bereits vor der Verwendung der Technik stets versucht wurde, die Datengröße eines Spiels so klein wie möglich zu halten, sind Optimierungsmöglichkeiten schon recht fortgeschritten. Es ist trotzdem weiterhin schwierig, ein komplettes Spiel ausschließlich mit photogrammetrischen Assets zu erstellen, ohne ständig ans Limit zu stoßen. An Optimierungsalgorithmen sollte deswegen trotzdem noch weiter gearbeitet werden, um das optimale Verhältnis zwischen Qualität und Datengröße zu finden. Zudem sollte Automatisierung weiter ein großes Ziel sein. Für die Optimierung wird in dieser Richtung schon viel entwickelt. Jedoch kann gerade bei photogrammetrischen Assets noch deutlich mehr automatisiert werden.

Das Erstellen eines Assets mit Photogrammetrie ist ein recht mühseliger Prozess. Die Bilder müssen manuell importiert, sortiert und ausmaskiert werden. Auch nachdem das Asset von der Photogrammetrie-Software generiert wurde müssen noch die Lichtinformationen aus der Textur entfernt und möglicherweise fehlerhafte Stellen im Modell repariert werden. Das alles könnte komplett automatisiert werden, da es sich hier nur um Fleißarbeit handelt und kein kreativer Input gefordert wird. Die Software könnte zum Beispiel automatisch fehlerhafte Bilder aussortieren, ein Algorithmus könnte das primäre Objekt im Foto bereits grob ausmaskieren und das gesamte Entfernen der Lichtinformationen könnte ebenso komplett automatisch geschehen. Das wäre nicht nur effizienter, es würde auch die Last von den Artists abnehmen. Durch diese vielen monotonen Arbeitsschritte ist Photogrammetrie keine beliebte Aufgabe und wird sogar von manchen Artists wegen der fehlenden Kreativität abgelehnt. Eine bessere Automatisierung würde bei diesen Problemen schnell Abhilfe schaffen.

Das Identifizieren der Probleme für die tatsächliche Spielentwicklung war recht schwierig. Zu diesem Thema gibt es sehr wenig Ressourcen und wirkliches Wissen dazu besitzen nur Spielentwickler, die viel mit der Technik arbeiten. Trotzdem konnte durch genaue Untersuchung der Phasen der Computerspielentwicklung und einzelne Antworten von Entwicklern wichtige Erkenntnisse gewonnen werden. Bereits in dem frühen Stadium des Level-Layouts muss die Technik berücksichtigt und genau geplant werden, welche Objekte wie im Spiel erscheinen sollen. So können bereits früh in der Produktion die größten Probleme eliminiert werden, die mit photogrammetrischen Assets einhergehen: die fotorealistische Einzigartigkeit und die schnelle Wiederholung, die dadurch eintritt.

Um dieses Problem weiterhin zu umgehen ist es wichtig, die Photogrammetrie bei Computerspielen nicht zur naturgetreuen Reproduktion realer Objekte zu nutzen. Vielmehr sollten die Vorteile des klassischen Workflows für Assets weiterhin erhalten werden, indem die gescannten Daten als Unterstützung für Modeling und Texturing genutzt werden. Durch kreative Bearbeitung und Wiederverwendung von gescannten Texturen und Geometrie kann Fotorealismus hergestellt werden, der nicht durch zu starke Wiederholung in sich zerfällt. Weiterhin können so photogrammetrische Daten auch für Assets verwendet werden, die sich eigentlich wenig für die Technik eignen. So stechen diese Assets später im Gesamtbild nicht zu stark heraus.

Es könnten weitere Erkenntnisse durch einen besseren Einblick in die Spielentwicklung von größeren Firmen gewonnen werden. Eine ausführliche Befragung von 3D-Artists und anderen Experten wäre dafür optimal. Leider war solch ein Befragung für diese Arbeit nur teilweise erfolgreich.

6 Fazit/Zukunftsausblick

Das Ziel der vorliegenden Arbeit war es, die Probleme der Photogrammetrie für Computerspiele aufzuzeigen und eine Vorgehensweise zu entwickeln, mit der auch unerfahrenen Spielentwickler mit der Technik arbeiten können. Dazu wurden Recherchen über den genauen Ablauf der Technik sowie über die einzelnen Schritte der Spielentwicklung vorgenommen. Es hat sich gezeigt, dass Probleme in nahezu jedem Schritt des Prozesses bestehen, vom richtigen Präparieren und Fotografieren der Objekte über die richtige Art und Weise mit den Daten geeignete Assets zu erstellen bis hin zu der tatsächlichen Verwendung von photogrammetrischen Assets in einem Computerspiel.

Die Probleme wurden ausführlich beleuchtet und die möglichen Lösungen unter dem derzeitigen Stand der Technik abgewogen. Mit Erfahrungsberichten und Befragungen von Entwicklern und Experten zu dem Thema konnte ein Leitfaden entwickelt werden, unter dessen Zuhilfenahme die größten Fehler bei der Verwendung der Technik vermieden werden können.

Obwohl Photogrammetrie für Computerspiele ein großes Potenzial besitzt, ist die Technik selbst mit viel Erfahrung deutlich problembehaftet. Bei der derzeitigen Nachfrage nach fotorealistischer Grafik ist es trotzdem wahrscheinlich, dass die Technik in Zukunft vom Großteil der AAA-Entwickler verwendet wird. Eine breitere Verwendung der Technik würde es wiederum ermöglichen, dass mehr Entwickler mit ihr experimentieren und Lösungsansätze entwickeln. So könnte das Verwenden von Photogrammetrie zukünftig zugänglicher werden, vorausgesetzt neue Entwicklungen werden ausreichend publiziert.

Es könnte ebenso möglich sein, dass die Photogrammetrie in Zukunft durch andere Techniken abgelöst wird, die nicht so viele Probleme verursachen. Doch selbst in diesem Fall ist es wahrscheinlich, das Fotos weiterhin als wichtige Quelle für fotorealistische Grafiken dienen.

Literaturverzeichnis

Agisoft LLC (Hrsg.): PhotoScan Memory Requirements, 19.09.2014, http://www.agisoft.com/pdf/tips_and_tricks/PhotoScan_Memory_Requirements.pdf (Zugriff am 03.02.2017).

Agisoft LLC (Hrsg.): Agisoft PhotoScan User Manual, 24.08.2016, http://www.agisoft.com/pdf/photoscan-pro_1_2_en.pdf (Zugriff am 03.02.2017).

Antoine, Francois: Creating the Open World Kite Real-Time Demo in Unreal Engine 4, 12.03.2015, https://www.youtube.com/watch?v=clakekAHQx0 (Zugriff am 03.02.2017).

Artec Europe: Artec Eva, 2017, https://www.artec3d.com/de/3d-scanner/artec-eva (Zugriff am 03.02.2017).

Autodesk Inc. (Hrsg.): Learn how to use 123d Catch, 2017, http://www.123dapp.com/howto/catch (Zugriff am 03.02.2017).

Azzam, Joseph: The Workflows of Creating Game Ready Textures and Assets using Photogrammetry, 24.08.2016, http://www.gamasutra.com/blogs/JosephAzzam/20160824/278719/The_Workflows_of_Creating_Game_Ready_Textures_and_Assets_using_Photogrammetry.php?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed%3A+GamasutraNews+%28Gamasutra+News%29 (Zugriff am 03.02.2017).

Blizard, Brandon: The Art of Photogrammetry: How To Take Your Photos, 19.02.2014, http://www.tested.com/art/makers/460142-art-photogrammetry-how-take-your-photos/ (Zugriff am 03.02.2017).

Brown, Kenneth / Hamilton, Andrew: Star Wars: Battlefront and the Art of Photogrammetry, 12.08.2016, https://www.youtube.com/watch?v=U_WaqCBp9zo (Zugriff am 03.02.2017).

Burgess, Joel: Skyrim's Modular Level Design - GDC 2013 Transcript, 13.04.2013, http://blog.joelburgess.com/2013/04/skyrims-modular-level-design-gdc-2013.html (Zugriff am 03.02.2017).

Busby, James: 3D Scanning Reflective Objects with Photogrammetry, 02.11.2016, http://www.3dscanstore.com/index.php?route=journal2%2Fblog%2Fpost&journal_blog_post_id=19 (Zugriff am 03.02.2017).

Electronic Arts Inc. (Hrsg.): How we used Photogrammetry to Capture Every Last Detail for Star WarsTM BattlefrontTM, 19.05.2015, http://starwars.ea.com/starwars/battlefront/news/how-we-used-photogrammetry (Zugriff am 03.02.2017).

Engineering & Manufacturing Services Inc. (Hrsg.): Types of 3D Scanners and 3D Scanning Technologies, 02.09.2015, https://www.ems-usa.com/tech-papers/3D%20Scanning%20Technologies%20.pdf (Zugriff am 03.02.2017).

Foster, Shaun / Halbstein, David: Integrating 3D Modeling, Photogrammetry and Design, London 2014.

Graphine Software (Hrsg.): Texture Streaming, 14.01.2016, http://graphinesoftware.com/our-technology/texture-streaming (Zugriff am 03.02.2017).

Graphine Software (Hrsg.): The Farm 51 uses Granite SDK to Create Truly Next Gen Graphics, 12.09.2014, http://graphinesoftware.com/blog/2014-09-12-farm-51-uses-granite-sdk-create-truly-next-gen-graphics (Zugriff am 03.02.2017). [1]

Graphine Software (Hrsg.): Granite SDK Version 2.0: Texture Streaming Middleware, 28.05.2014, http://graphinesoftware.com/ckfinder/userfiles/files/Graphine-Granite-Technical-Description.pdf (Zugriff am 03.02.2017). [2]

Heaven, Douglas: Picture perfect pixels, in: New Scientist 223 Issue 2977(2014), S. 18.

Heckbert, Paul: Texture Mapping Polygons in Perspective, 28.04.1983, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.53.158&rep=rep1&type=pdf (Zugriff am 03.02.2017).

HP Development Company, L.P.: Overview, 2017, http://www8.hp.com/us/en/campaign/3Dscanner/overview.html (Zugriff am 03.02.2017).

Huang, Hui et al.: Consolidation of unorganized point clouds for surface reconstruction, Vancouver 2009.

Joyce, Dan: High Dynamic Range Photography/Photogrammetry – Part 1, 12.11.2015, https://digitalheritagerecording.wordpress.com/2015/11/12/high-dynamic-range-photography-photogrammetry-part-1/ (Zugriff am 03.02.2017).

Kleinkemper, Larry: The Basics of 3D Laser Scanning, 02.04.2014, http://lanmarservices.com/2014/04/02/the-basics-of-3d-laser-scanning/ (Zugriff am 03.02.2017).

Kremers, Rudolph: Level Design: Concept, Theory, and Practice, London 2009.

Lantz, Chris: High Dynamic Range Photography, in: Visual Communication Journal, 2007, http://gceaonline.org/wp-content/uploads/2014/08/2007springvcj.pdf#page=9 (Zugriff am 03.02.2017).

Leonova, Margarita: Laser Scanning vs. Photogrammetry, 07.11.2014, http://lanmar-services.com/2014/11/07/laser-scanning-vs-photogrammetry/ (Zugriff am 03.02.2017).

Li, Zhilin et al.: Effects of JPEG Compression on the Accuracy of Photogrammetric Point Determination, in: Photogrammetric Engineering & Remote Sensing Vol. 68 No. 8 vom August 2002, S. 847-853.

Mallison, Heinrich: Photogrammetry tutorial 2: picture taking, general remarks, 16.11.2013, https://dinosaurpalaeo.wordpress.com/2013/11/16/photogrammetry-tutorial-2-picture-taking-general-remarks/ (Zugriff am 03.02.2017).

Madigan, Jamie: The Psychology of Immersion in Video Games, 27.07.2010, http://www.psychologyofgames.com/2010/07/the-psychology-of-immersion-in-video-games/ (Zugriff vom 03.02.2017)

Moraes, Cicero: Comparing 7 photogrammetry systems. Which is the best one?, 07.12.2016, http://arc-team-open-research.blogspot.de/2016/12/comparing-7-photogrammetry-systems.html (Zugriff am 03.02.2017).

Nvidia Corporation (Hrsg.): Using ASTC Texture Compression for Game Assets, 20.02.2015, https://developer.nvidia.com/astc-texture-compression-for-game-assets#_Toc398571907 (Zugriff am 03.02.2017).

Oh, Min: Imperfection for Perfection Part 2: Photo Reconstruction/Delighting, 03.12.2015, https://www.unrealengine.com/blog/imperfection-for-perfection-part-2 (Zugriff am 03.02.2017).

Perry, Lee: Modular Level and Component Design, November 2002, https://docs.unrealengine.com/udk/Three/rsrc/Three/ModularLevelDesign/ModularLevelDesign.pdf (Zugriff am 03.02.2017).

Pharr, Matt / Humphreys, Greg: Physically Based Rendering: From Theory to Implementation, Zweite Edition, Burlington 2010.

Pettit, Nick: The Beginner's Guide to Physically Based Rendering in Unity, 17.11.2015, http://blog.teamtreehouse.com/beginners-guide-physically-based-rendering-unity (Zugriff am 03.02.2017).

Plante, Chris: Microsoft's Shannon Loftis on E3 leaks, the status of Kinect, and the future of controllers, 15.06.2016, http://www.theverge.com/2016/6/15/11940202/Project-scoropio-xbox-one-leaks-kinect-microsoft-interview-shannon-loftis (Zugriff am 03.02.2017).

Poznanski, Andrzej: Visual Revolution of the Vanishing of Ethan Carter, 25.03.2014, http://www.theastronauts.com/2014/03/visual-revolution-vanishing-ethan-carter (Zugriff am 03.02.2017).

Quan, Long: Image-Based Modeling, New York 2010.

Redweik, Paula: Photogrammetry. In: Xu, Guochang (ed.) Sciences of Geodesy – II: Innovations and Future Developments, S. 133-183, Heidelberg 2013.

Renambot, Luc et al.: Realtime Compression for High-Resolution Content, 21.07.2007, https://www.evl.uic.edu/documents/ag2007-renambot.pdf (Zugriff am 03.02.2017).

Reuther, Phillipp: Get Even: info blowout with exclusive interview - 3D scanning in games, in: PC Games Hardware vom 17.02.2014, http://www.pcgameshardware.de/Get-Even-Spiel-14308/News/Get-Even-exclusive-interview-3D-scanning-in-games-1109867/ (Zugriff am 03.02.2017).

Rüger, Florian: Verfahren zur Aufbereitung von 3D-Modellen für die Verwendung in Echtzeitanwendungen, Mittweida 2016.

Russell, Jeff: Basic Theory of Physically Based Rendering, 1. November 2015, https://www.marmoset.co/posts/basic-theory-of-physically-based-rendering/ (Zugriff am 03.02.2017).

Sachs, Seori: Tutorial: Lighting in Photogrammetry, 30.09.2016, https://blog.sketchfab.com/lighting-in-photogrammetry/ (Zugriff am 03.02.2017).

Simplygon: Simplygon 8.0 Feature List, 2017, https://www.simplygon.com/knowledgebase/documentation/simplygon-sdk-81/simplygon-81-feature-list/processors/ (Zugriff am 03.02.2017).

Sketchfab Inc. (Hrsg.): How to set up a successful photogrammetry project, 18.06.2015, https://blog.sketchfab.com/how-to-set-up-a-successful-photogrammetry-project/ (Zugriff am 03.02.2017).

Slama, Chester et al.: Manual of photogrammetry, Virginia 1980.

Sorkine–Hornung, Olga: Instant Field Aligned Meshes (SIGGRAPH ASIA 2015), 25.09.2015, https://www.youtube.com/watch?v=U6wtw6W4x3I (Zugriff am 03.02.2017).

Tirosh, Udi: Getting Started With Cross Polarized Light, 05.04.2012, http://www.diy-photography.net/getting-started-with-cross-polarized-light/ (Zugriff am 03.02.2017).

Wenzel, Jakob et al.: Instant Field-Aligned Meshes, 25.09.2015, http://igl.ethz.ch/projects/instant-meshes/instant-meshes-SA-2015-jakob-et-al.pdf (Zugriff am 03.02.2017).

Willson, Jeffrey Ian: schriftliche Befragung am 01.01.2017.

Wilson, Joe: PBR Texture Conversion, 01.09.2015, https://www.marmoset.co/posts/pbr-texture-conversion/ (Zugriff am 03.02.2017).

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe. Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Ort, Datum Vorname Nachname