HAW HAMBURG

Quantitative und qualitative technische Analyse von Tools zur Materialerstellung für die Unreal Engine 5

Bachelor-Thesis

zur Erlangung des akademischen Grades B.Sc.

im Studiengang Medientechnik

Nicole Carina Steidle, [redacted]

Erstprüfer: Prof. Ralf Hebecker

Zweitprüfer: Benjamin Assig

Hamburg, 10.08.2022

Hochschule für Angewandte Wissenschaften Hamburg

Fakultät Design, Medien und Information

Department Medientechnik

Zusammenfassung

Die folgende Thesis beschäftigt sich mit einem Vergleich der Software-Pakete *Adobe Substance 3D Suite* und *Quixel* zur Materialerstellung für *Unreal Engine 5*-Spiele. Im Verlauf der Arbeit werden Grundbegriffe der Materialerstellung erläutert und die Programme vorgestellt. Der praktische Teil beinhaltet die Erstellung einer Szene in Unreal und ihre Texturierung unter Zuhilfenahme beider Softwares. Anschließend erfolgt die Analyse von Funktionen und Workflow beider Tools exemplarisch anhand einzelner Beispiele, an denen ihre Unterschiede deutlich werden. Die darauffolgende Evaluation zeigt die Stärken, Schwächen und Anwendungsbereiche von Substance und Quixel auf.

Abstract

The following thesis deals with the comparison of the tools Adobe Substance 3D Suite and Quixel for material creation for Unreal Engine 5 games. The course of this thesis consists of the explanation of material creation and the introduction of the programs. The practical part includes the creation of the scene in Unreal and its texturing using both software packages. This is followed by an analysis of functions and workflows in an exemplary way, by using individual examples to illustrate their differences. The subsequent evaluation demonstrates the strengths, weaknesses and possible areas of application for Substance and Quixel.

Inhaltsverzeichnis

1. Einleitung 5
2. Grundlagen der Materialerstellung 7
   1. Materialien für Games 7
      Draw Calls 8
      Poly Count 8
      Texturen 9
      Texel-Dichte 10
      Shader 10
   2. Materialtypen 12
      Nahtlose Texturen 12
      UV-gebundene Texturen 13
      Decal-Materialien 14
3. Softwaregrundlagen 15
   1. Unreal Engine 5 15
      Texture Asset Editor 16
      Material Editor 16
      Blend Mode und Shading Model 17
      Profiling Tools 19
   2. Adobe Substance 3D Suite 21
      Sampler, Designer und Painter 22
   3. Quixel 24
      Mixer 25
   4. Softwareanforderungen 25
4. Praxistest 27
   1. Planung und Setup der Szene 27
      Materialien 28
      Beleuchtung 29
      Setup der Applikationen 29
   2. Library-Inhalte 30
   3. Material Presets in Unreal 32
      Substance-Presets 32
      Quixel-Presets 34
   4. Verwendung von Materialien direkt aus der Library 36
      Beet 36
      Landscape 38
   5. UV-gebundene Materialien 39
      Setup 39
      Hinzufügen der Materialien 40
      Bearbeitung 41
      Zusammenfassung 44
   6. Erstellung der finalen Szene 45
      Mauerstein 45

Fliesen 47
Foliage 48
Vorhänge 49
7. Performance 51
Shader Complexity 51
stat Unit 52
stat RHI 53
5. Auswertung 53
1. Optischer Eindruck 54
2. Libraries 55
3. Funktionen 56
4. Workflow und Integration in Unreal Engine 5 57
5. Einarbeitungszeit und Arbeitsaufwand 57
6. Performance 58
7. Fazit 59
Literaturverzeichnis 62
Abbildungsverzeichnis 63
Software 64
Asset-Verzeichnis 64
Eigenständigkeitserklärung 65

1. Einleitung

Ein eigenes Videospiel zu entwickeln ist der Traum vieler. Allein 2021 wurden über 10.500 neue Titel auf der größten Gaming-Plattform „Steam“ veröffentlicht (Statista, 2022). Doch insbesondere für kleinere Produktionen mit geringerem Budget kann die Spieleentwicklung eine große Herausforderung darstellen. Einerseits wachsen die Anforderungen an Grafik und Gameplay stetig, wodurch der Produktionsaufwand steigt. Andererseits sind kostenpflichtige Tools, die zeitliche und technische Erleichterungen versprechen, oft nicht finanzierbar.

Die grafische Qualität eines Spieles ist sein Aushängeschild und trägt zur Sichtbarkeit und einer Kaufentscheidung bei. Für die Grafik sind ein stimmiger Gesamteindruck und auf das Spiel zugeschnittene Materialien wichtig. Je individualisierter die Anpassungen sind, umso aufwändiger und teurer kann die Produktion durch benötigte kostenpflichtige Hilfsmittel werden.
In großen Studios hat sich seit längerem die Adobe Substance 3D Suite als Industriestandard für High-End-Materialien etabliert. Mit ihr lassen sich qualitativ hochwertige und detaillierte Grafiken für Spiele erstellen. Dazu gehört eine Asset-Library mit tausenden verfügbaren Materialien. Die Suite ist als kostenpflichtiges Abonnement verfügbar.
Mit dem offiziellen Release der Unreal Engine 5 probiert Epic Games eine Alternative zur Substance Suite einzuführen. Diese nennt sich Quixel und enthält eine Bibliothek sowie eine Texturierungssoftware. In Kombination mit Unreal ist Quixel vollständig kostenlos und soll ebenfalls qualitative Materialien erstellen können.

Das Ziel dieser Bachelorarbeit ist ein praktischer Vergleich von Funktionsweise, Nutzerfreundlichkeit, Effizienz, Stärken und Schwächen der Programme Substance und Quixel. Die Softwares sollen auf ihre Funktionalität zur Erstellung von Materialien für Spiele geprüft werden. Dafür wird eine Szene für ein Computerspiel erstellt und unter Verwendung beider Tools vollständig texturiert.
Als Szene habe ich einen mediterranen Innenhof gewählt, da mich die Umsetzbarkeit dieses südländischen Ambientes für ein eigenes Projekt interessierte. Daher enthält sie konkrete Elemente des künstlerischen Spiel-Konzeptes. Zudem birgt dieses Setting das Potenzial, viele unterschiedliche Materialien zu beleuchten. Materialien werden sowohl direkt aus der Substance 3D Asset Library und der Megascans Library verwendet als auch in den Texturierungsprogrammen auf Meshes angepasst.
Für die Erstellung der Gebäude werden selbst-modellierte Assets und zusätzliche 3D-Meshes aus dem Asset-Paket „Middle Eastern Courtyard“ von Joakim Stigsson sowie der Megascans Library verwendet. Die untexturierte Szene wird anschließend dupliziert, um für beide Softwares die gleichen Voraussetzungen zu schaffen.

Der Texturierungsprozess sowie die fertigen Szenen werden unter den folgenden Aspekten analysiert:

- Optischer Gesamteindruck
- Inhalte der Bibliotheken und Applikationen
- Workflow und Integration in Unreal Engine 5
- Einarbeitungszeit und Arbeitsaufwand
- Performance der Materialien in der Engine

Als Ergebnis soll eine differenzierte Evaluation Aufschluss darüber geben, für welche Projektanforderungen Substance oder Quixel besser geeignet ist. Damit einher geht die Fragestellung, ob Quixel für Indie-Produktionen mit kleinen Teams und überschaubarem Budget eine ernsthafte Alternative zu Substance darstellen kann, ohne dass die geringen finanziellen Mittel sichtbar oder im Prozess bemerkbar werden.

Die Arbeitet richtet sich an alle Interessierten, die Möglichkeiten suchen, Projekte zu realisieren. Das beinhaltet Artists, die einen Blick auf die Arbeitsweise der verschiedenen Softwares werfen möchten. Genauso richtet sie sich an Game-Developer, die unter Zuhilfenahme dieses Projektes die Realisierbarkeit ihres eigenen Spieles evaluieren möchten. Grundlegende Kenntnisse der Materialerstellung sowie des PBR-Workflows sind von Vorteil, da theoretische Hintergründe nicht im Detail erklärt werden.

2. Grundlagen der Materialerstellung

Materialien sind diejenigen Assets, die auf der Oberfläche des 3D-Meshes für deren individuelles Aussehen verantwortlich sind. Durch den seit langem etablierten Standard *„Physically Based Rendering“* – kurz PBR – setzen sich diese Materialien aus verschiedenen physikalischen Eigenschaften zusammen. Dadurch wird eine fast lebensnahe Darstellung ermöglicht. Lebensnah meint dabei nicht zwangsweise fotorealistisch, da dieses Beleuchtungsmodell genauso für stilisierte Materialien angewendet wird (McKenney, 2019). Vielmehr geht es dabei um die Reaktion von Oberflächen auf Licht, was beispielsweise Rauheit, metallische Reflexionen oder Hoch-Tief-Strukturen betrifft, sodass Materialien als natürlich wirkend wahrgenommen werden und auf Lichteinflüsse physikalisch korrekt reagieren (Russel, 2022) (Abbildung 1).

Abbildung 1: Auf Lichteinfall reagierendes PBR-Material.

1. Materialien für Games

Konkrete Anforderungen an Materialien lassen sich nicht für alle Spiele generell zusammenfassen. Vor jeder Produktion müssen verschiedene Aspekte beleuchtet werden, um eine individuelle Balance zwischen angestrebter Grafik und möglichst guter Performance zu erzielen. Einerseits sollte die Grafik dem Anspruch des Spiels gerecht werden, andererseits müssen Speicherkapazität und maximale Rechenleistung der Zielplattform beachtet werden. Für

bestmögliche Grafik sollten alle performanceintensiven Faktoren möglichst begrenz werden. Dazu zählen beispielsweise *Draw Calls*, *Poly Count*, die Größe der Texturen oder manche Umsetzungen von Materialien in der Engine.

Draw Calls

Unter einem *Draw Call* versteht man alles, was innerhalb eines Frames auf dem Bildschirm gerendert werden muss. Dazu zählen 3D-Meshes, Materialien und Shader, die in diesem einzelnen Frame dargestellt werden (Epic Games, 2022). Eine hohe Anzahl dieser Objekte – und somit viele *Draw Calls* – können die Performance stark beeinflussen.
Die Optimierung von *Draw Calls* kann folgendermaßen erfolgen:

- Durch die Reduktion von Einzelmeshes in einer Szene reduziert sich auch die Anzahl der *Draw Calls.*
- Durch die Reduktion von Einzelmaterialien können ebenfalls *Draw Calls* eingespart werden. Anstatt viele verschiedene Materialien auf ein Asset zu projizieren, sollten diese vorab zusammengefügt und als einzelnes Material in die Engine eingefügt werden.
- Ein modulares Bauteil, das mehrfach verwendet wird, kann instanziiert werden. Unabhängig davon, wie viele identische Bauteile platziert werden, werden die *Draw Calls* dadurch auf die eines einzelnen Bauteiles reduziert. Mit dieser Technik lässt sich beispielsweise auch Gras darstellen, ohne dass die Performance leidet.

Poly Count

Unter dem *Poly Count* eines Meshes versteht man die Anzahl einzelner Dreiecke, die für seine Darstellung benötigt werden. Ist der *Poly Count* sehr hoch, führt das ebenfalls zu Performanceeinbußen. Um dies zu umgehen, kann das Oberflächenverhalten eines hochdetaillierten Meshes auf eines mit weit niedrigerem *Poly Count* „gebacken" werden. Die aus dem *Baking* enthaltene *Tangent Space Normal Map* kann den *Poly Count* deutlich reduzieren.

Eine Ausnahme hiervon bildet eine mit Unreal Engine 5 neu implementierte Rendertechnik, die große Mengen Vertices ohne Performanceeinbußen rendert. Dafür werden aus den regulär genutzten *„Static Meshes"* sogenannte *„Nanite Meshes"*, die von der Engine analysiert und in Polygon-Gruppen unterteilt werden. Diese werden je nach Wahrnehmbarkeit aus Perspektive der Kamera nicht oder nur teilweise gerendert. Das neue System kann insbesondere bei der Entwicklung von Spielen auf stärkeren Plattformen eine deutliche Zeitersparnis bei Mesh-Optimierung und *Normal Map*-Baking bewirken (Epic Games, 2022). Wird jedoch zusätzlich für schwächere Plattformen entwickelt, muss wie bisher optimiert werden und Nanite kann lediglich als High-End-Option in den Grafikeinstellungen genutzt werden.

Texturen

Der PBR-Workflow arbeitet mit verschiedenen physikalischen Oberflächeneigenschaften, die entweder als Vektoren oder als 2D-Texturen dargestellt werden (VisCircle, 2021). Solche Texturen erhält man beim Gestalten von Materialien in Substance oder Quixel (Abbildung 2).

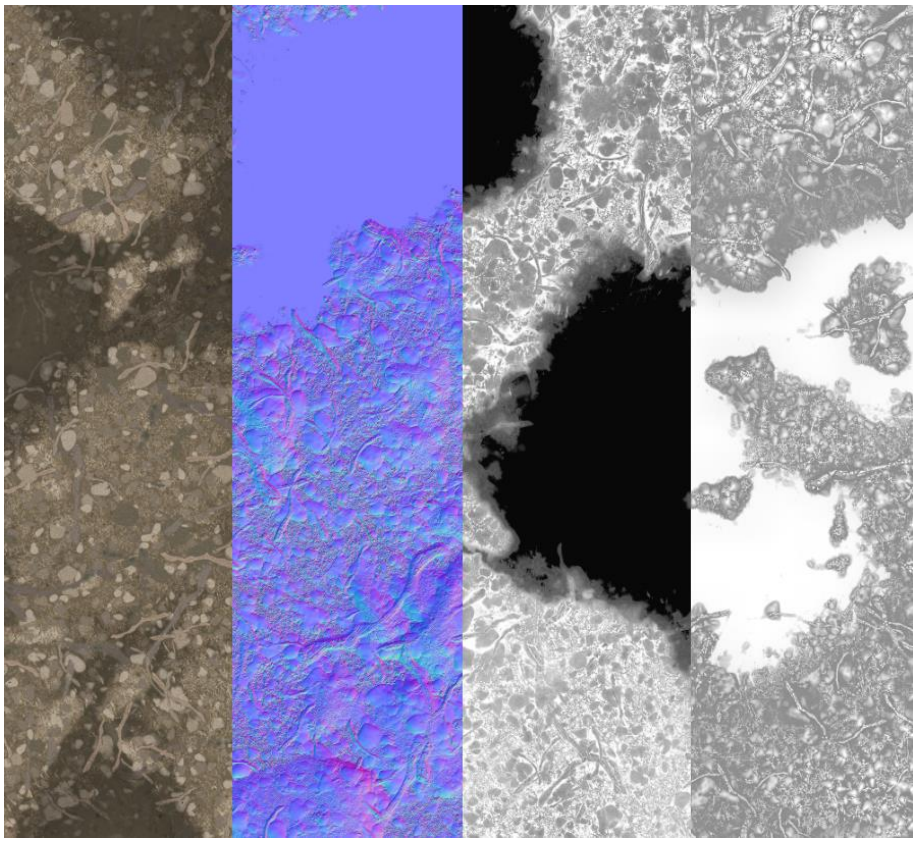

Abbildung 2: *Texture Maps*. Von links Base Color Map, Normal Map, *Roughness Map* und *Ambient Occlusion Map*.

Jede dieser Texturen wird beim Rendern des Materials im Video-RAM verarbeitet. Je größer die Auflösung ist, umso höher ist der Rechenaufwand. Sind Texturen zu groß für den VRAM, muss die Engine diese kleiner skalieren, was zu einem unkontrollierbaren und ungewollten Qualitätsverlust der Grafik führt (Epic Games, 2022).

Für die Wahl der richtigen Texturgröße sind der generelle Grafikstil und die Relevanz des jeweiligen texturierten Objektes für den Spielverlauf zu beachten. Inhaltlich relevante oder nah gezeigte Assets werden in höherer Auflösung gerendert, als weniger wichtige Objekte im Hintergrund. Auch große Farbflächen kommen ohne Informationsverlust mit einer geringeren Anzahl Pixel aus als detaillierte fotorealistische Grafiken.

Generell sollten Texturen nur verwendet werden, wenn sie zwingend erforderlich sind. Bei einfarbigen Flächen sollte statt der *Base Color Map* ein RGB-Vektor benutzt werden. Anstelle einer *Roughness Map* mit nur einem Farbwert kann ein Zahlenvektor eingesetzt werden.

Texel-Dichte

Die Texturgröße eines Assets muss relativ zur Texturgröße umliegender Objekte gewählt werden. Als Maßeinheit dafür dient die Texel-Dichte, die das absolute Verhältnis einer Texturgröße in Pixeln zur virtuellen Geometrieeinheit der Engine – Zentimeter – ausdrückt. Eine Textur mit einem Seitenverhältnis von 512 x 512 px hat auf einer Fläche von 1 x 1 m eine Texel-Dichte von 5,12 px/cm. Projiziert man auf ein benachbartes, gleich großes Objekt eine Textur mit 1024 x 1024 px, erhält man eine Texel-Dichte von 10,24 px/cm. Daraus erfolgt ein inkonsequentes Gesamtbild und das Asset mit der geringeren Texel-Dichte wirkt unscharf. Daher ist es wichtig, diese Texel-Dichte für alle Texturen einer Szene anzupassen.

Für verschiedene Arten von Spielen empfehlen sich unterschiedliche Texel-Dichten (Iezzi, 2016):

- Spiele in Vogelperspektive benötigen keine hohe Texel-Dichte, da die Kamera nie besonders nah an Objekte herankommt.
- *Third Person*-Spiele legen ihren Fokus auf den zentral dargestellten gespielten Charakter. Dieser und wichtige nahe Assets im spielbaren Bereich werden deshalb höher aufgelöst werden.
- *First Person*-Spiele sind am höchsten aufgelöst, da die Kamera durch die Ego-Perspektive sehr nah an viele Objekte herankommt. Besonders großformatige Elemente wie die Hände des gespielten Charakters benötigen eine besonders hohe Texel-Dichte (Abbildung 3).

Abbildung 3: Die Hände nehmen einen Großteil des Bildschirms ein und benötigen daher eine hohe Texel-Dichte (Electronic Arts, 2022).

Shader

Als Shader werden Programme bezeichnet, die Materialien rendern. Je nach Shader können mittels mathematischer Operationen unterschiedliche visuelle Effekte erreicht werden. Um die Performance nicht zu beeinträchtigen, sollten rechenintensive Shader möglichst sparsam eingesetzt werden. Zu diesen anspruchsvollen Rechenoperationen gehören beispielsweise Sinusse oder Potenzen, wohingegen Additionen, Subtraktionen oder Multiplikationen schneller berechnet werden (Arm, 2022).

Folgende weitere Elemente beeinflussen die Komplexität eines Shaders und somit seinen Rechenaufwand:

- **Beleuchtung**: Ein vom Licht unbeeinflusster Shader benötigt weniger Rechenaufwand als einer, für den ein vollständiges Beleuchtungsmodell berechnet wird. Daher werden für *Mobile Games* oftmals sogenannte *Unlit Shader* ohne zusätzliche Lichtquelle verwendet, um eine gute Performance zu erreichen (Arm, 2022). Dadurch, dass *Unlit* anstelle von *Base Color* nur den Input „*Emissive*" zulässt, sind die Objekte trotz eigentlicher Dunkelheit sichtbar, da sie selbst leuchten.

- **Materialien mit Alpha-Werten**: Transparenz in Materialien wird mit Alpha-Werten zwischen 0 und 1 angegeben. Dabei wird zwischen den drei Materialtypen opak, transparent und transluzent unterschieden. Opake Materialien sind die am häufigsten verwendeten und mit einem Alphawert von 1 vollständig sichtbar. Des Weiteren gibt es Materialien mit Transparenz. Diese enthalten Pixel mit einem Alphawert von 0, wodurch diese Stellen des Materials unsichtbar dargestellt werden. Materialien mit Werten größer 0 und kleiner 1 sind transluzent, also sichtbar, aber durchscheinend (Epic Games, 2022) (Abbildung 4).

  Abbildung 4: Alphawerte von 0 bis 1 mit Visualisierung des entsprechend transparenten, transluzenten und opaken Effekts (Epic Games, 2022).

  Opake und transparente Materialien werden von vorne nach hinten gerendert. Das bedeutet, dass die der Kamera nächsten Punkte zuerst gerendert werden. Entsprechend kann kein Pixel, nachdem es gerendert wurde, von einem dahinterliegenden Mesh oder Material beeinflusst werden.
  Bei transluzenten Materialien ist dies umgekehrt – sie werden von hinten nach vorne gerendert. Hier färben weiter vorne liegende Pixel auf die zuerst gerenderten ab, ein durchscheinender Effekt entsteht. Hierfür müssen Pixel mit deutlich höherem Rechenaufwand mehrfach gerendert („overdrawn") werden. Daher muss bei der Planung einer Szene mit Einsatz von Transluzenz die Leistungsfähigkeit der Plattform beachtet werden.

2. Materialtypen

Man unterscheidet zwischen verschiedenen Materialtypen, die sich im Aufbau in der Engine, in der Vorbereitung und Herstellung oder in der Nutzung unterscheiden.

Nahtlose Texturen

Eine nahtlose Textur sollte durch einfaches Kopieren rapportierbar sein, sich also ohne sichtbare Streifen zu einer größeren Fläche zusammenfügen lassen (Povey, 2022). Das bietet den Vorteil, dass mit geringem Aufwand und einer verhältnismäßig kleinen Textur deutlich größere Assets gestaltet werden können. Dieses Prinzip eignet sich für alle Materialien, bei denen durch die Erweiterung keine ungewollten offensichtlichen Wiederholungen im Muster erkennbar sind. Bei starken Helligkeitskontrasten oder auffälligen Mustern muss andernfalls nachgebessert werden und die Materialien können nicht ohne Überarbeitung auf einem Objekt platziert werden.
In Abbildung 5 sind Fliesen zu sehen, die durch ihre ohnehin geometrische Form prädestiniert für erweiterbare Texturen sind. Durch den geringen Farbkontrast des Staubes zum hellen Teil der Fliesen und seiner feinen Struktur fallen Wiederholungen erst bei genauerem Hinsehen auf. Anders als bei den anderen Materialien: Im Holzmuster werden Wiederholungen als dunkle horizontale Streifen sichtbar und auf der Steinwand als helle beziehungsweise dunkle Punkte.

Abbildung 5: Texturen mit unsichtbaren (links) und vertikalen (Mitte) beziehungsweise beidseitig (rechts) sichtbaren Duplikationen.

Die genannten Beispiele sind in alle Richtungen duplizierbar. Genauso gibt es Texturen, die ausschließlich entlang einer Achse erweiterbar sind. Diese Art von Textur eignet sich für Architekturelemente wie Stuck oder Details wie Bordüren. Ein weiteres Anwendungsbeispiel sind Zäune, die bei einem geringen Detaillevel aufgrund höherer Entfernung zur Kamera nur auf einer Fläche mit transparenten Zwischenräumen dargestellt werden (VisCircle, 2021).

Ein Spezialfall nahtlos erweiterbarer Texturen sind *Landscape Materials*, die auf ein teils enorm großes Gelände projiziert und mit anderen Materialien überblendet und verwischt werden. Auf dieser Größe ist es oftmals nicht möglich, dass Nähte automatisch vollständig

unsichtbar sind. Daher müssen diese Materialien gewöhnlich in der Engine speziell verblendet werden, damit sich sichtbare Streifen auflösen. Das ist beispielsweise durch Farbvariationen oder einer Skalierung der Textur relativ zu ihrer Entfernung zur Kamera möglich (Abbildung 6).

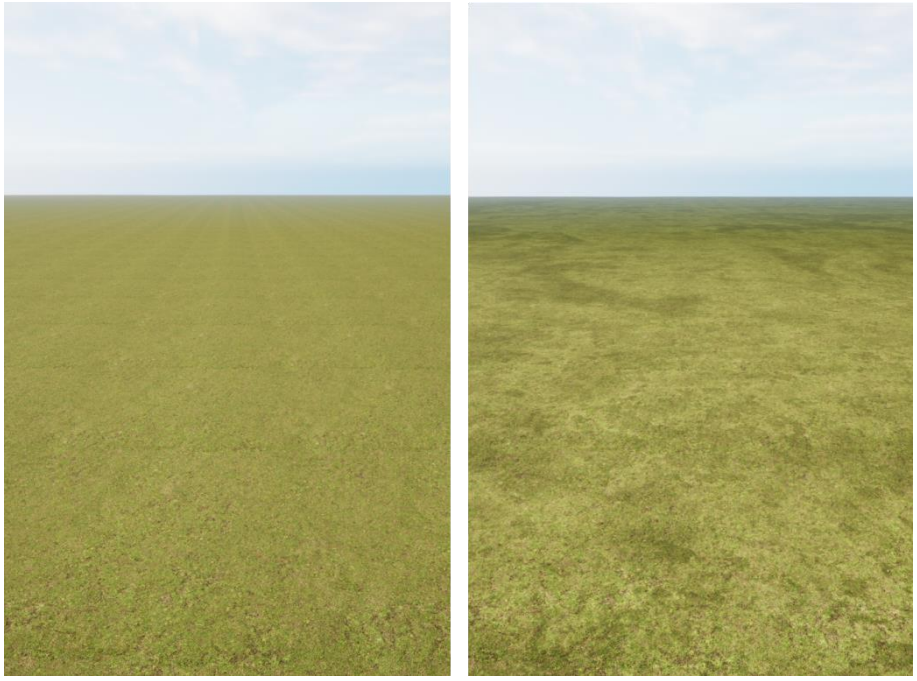

Abbildung 6: Links eine Landschaft mit sichtbaren Kopierstreifen. Rechts nach der Verdeckung der Kopierstreifen mit einer Farbvariations-Textur.

UV-gebundene Texturen

Bei vielen Assets genügt es nicht, ein Material auf die Oberfläche zu projizieren. Stattdessen müssen Anpassungen wie beispielsweise Kantenabnutzung, Verfärbungen oder Kratzer hinzugefügt werden. Ebenfalls ist es möglich, dass ein Objekt aus mehreren Materialien besteht. Materialien sollten zum Einsparen von *Draw Calls* soweit es die jeweilige Rendertechnik zulässt idealerweise in möglichst wenigen Texturen gespeichert werden. Je individueller und aufwändiger eine Spielumgebung aussehen soll, umso mehr Zeit muss in die Einzelanpassung von Materialien direkt auf Assets investiert werden.

Abbildung 7: Beispiel für ein UV-gebundenes Material, das speziell für dieses Asset hergestellt wurde.

Abbildung 7 zeigt ein Asset, dessen Oberfläche aus mindestens 4 Materialien gestaltet wurde. Grundsätzlich wäre es möglich, diese erst direkt in der Engine einzufügen. Das wäre jedoch sowohl für die Performance, als auch für den Speicher enorm ineffizient. Daher werden die Materialien vorab extern kombiniert.

In Abbildung 8 wurden die Farbwerte und Materialeigenschaften wie Oberflächenrauheit zu einem Material verschmolzen. So wird in der Engine nur eine weitere Materialinstanz benötigt.

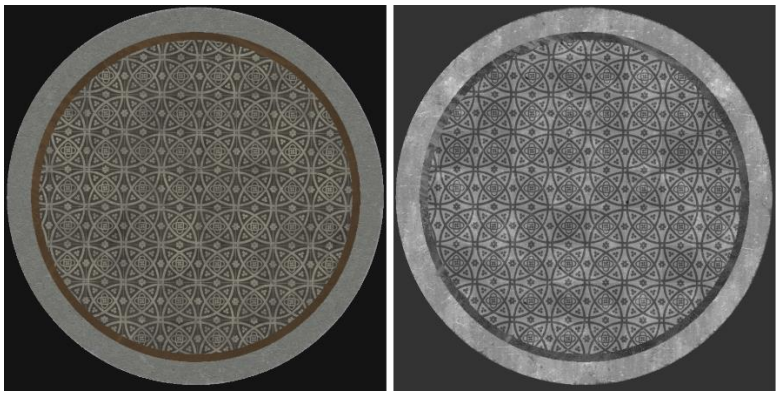

Abbildung 8: *Base Color* und *Roughness Map* der Tischplatte. Die Oberflächeneigenschaften der verschiedenen Materialien wurden auf ein einziges Material kombiniert.

Decal-Materialien

Diese sogenannten „Aufkleber"-Materialien tragen ihren Namen zurecht, denn sie können in der Engine direkt auf Geometrieobjekten platziert werden (World of Level Design, 2016). Durch diese Materialtypen kann eine Szene durch zusätzliche Details aufgewertet werden, ohne dass dafür zusätzliche Meshes benötigt werden. Decal-Materialien haben alle benötigten Informationen wie Farbwerte, *Normal Map* oder *Roughness* in die Texturen gebacken, ebenso wie die Alphawerte für die Darstellung der Umrisse des jeweiligen Objektes. Auf diese Art können sie direkt an der gewünschten Stelle platziert sowie problemlos skaliert und rotiert werden. Dadurch erhöhen sie die grafische Komplexität der Szene deutlich. Da einzelne Texturen nicht extern bearbeitet und als neue Instanzen importiert werden müssen, können Artists immens Zeit einsparen und bleiben hinsichtlich etwaiger Änderungen weitaus flexibler.

Decals können Beschädigungen an Wänden, Abdrücke von Autoreifen im Staub oder gesprühte Graffitis sein (Abbildung 9). Selbst Fenster, Türen oder vertikale Bepflanzung können auf diese Weise optisch hinzugefügt werden.

Abbildung 9: Decal-Materialien in Form von Schmierern und Rissen an der Hausfassade sowie als Reifenspur auf dem Boden (Sleegers, 2022).

3. Softwaregrundlagen

Das Praxisprojekt wird in der Unreal Engine 5 erstellt. Die Gestaltung der Materialien erfolgt in den Softwares Adobe Substance 3D Suite und Quixel. In diesem Kapitel werden ein Gesamtüberblick über die Programme und ihre relevanten Schlüsselfunktionen dargelegt.

1. Unreal Engine 5

Die Unreal Engine 5 ist eine der führenden Spiele-Engines auf dem Markt. Ihre mittlerweile fünfte Generation hatte am 5. April 2022 ihr offizielles Release. Hinter ihr steht das amerikanische Softwareunternehmen Epic Games, Inc, das die Engine im Jahr 1998 erstmals veröffentlichte. Sie ist neben Unity 3D eine der stärksten 3D-Spiele-Engines, mit der viele bekannte Spieletitel umgesetzt wurden (Abbildung 10 und 11).

Abbildung 10: Screenshot aus Life is Strange: True Color, entwickelt mit Unreal Engine 4 (Square Enix, 2021).

Abbildung 11: Screenshot aus Fortnite (Epic Games, Inc., Lucasfilm Ldt., 2022)

Seit 2015 ist der Download der Engine kostenlos (Steinlechner, 2015), wodurch sie zunehmend interessanter für kleinere Indie-Produktionen und Projekte mit kleinem Budget wurde. Sogar veröffentlichte Spiele bleiben bis zu einer Einnahmegrenze von 1 Million US-Dollar kostenfrei, erst darüber werden 5 % der Quartalseinnahmen an Epic Games fällig. Dadurch eignet sich Unreal Engine gut zum Lernen und Ausprobieren.

Texture Asset Editor

Unreal Engine arbeitet mit dem Metallic-PBR-Workflow, bei dem sich einzelne Materialien unter anderem aus *Base Color*, *Roughness* und *Metallic*-Texturen oder Vektoren zusammensetzen. Diese Texturen werden über den Import-Button oder per Drag & Drop in die Engine importiert. Im *Texture Asset Editor* können Eigenschaften wie Texturgröße und *Compression Settings* eingesehen werden (Abbildung 12). Letztere sollten insbesondere beim Import von *Channel Packed Maps* überprüft und gegebenenfalls manuell zu „Mask (no sRGB)“ geändert werden.

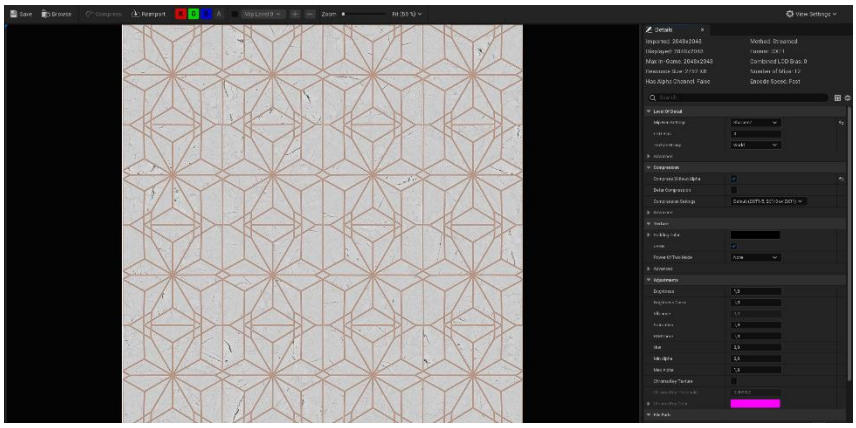

Abbildung 12: User-Interface des Texture Asset Editors mit Vorschau der Textur und Übersicht über Eigenschaften und Import-Einstellungen.

Material Editor

Das Zusammenfügen eines Materials erfolgt nach Erstellen einer neuen Materialdatei im *Material Editor* (Abbildung 13). Dies ist eine Oberfläche zur grafischen Programmierung mit Nodes, sogenannter *Material Expressions* und *Material Functions* (Epic Games, Inc.).

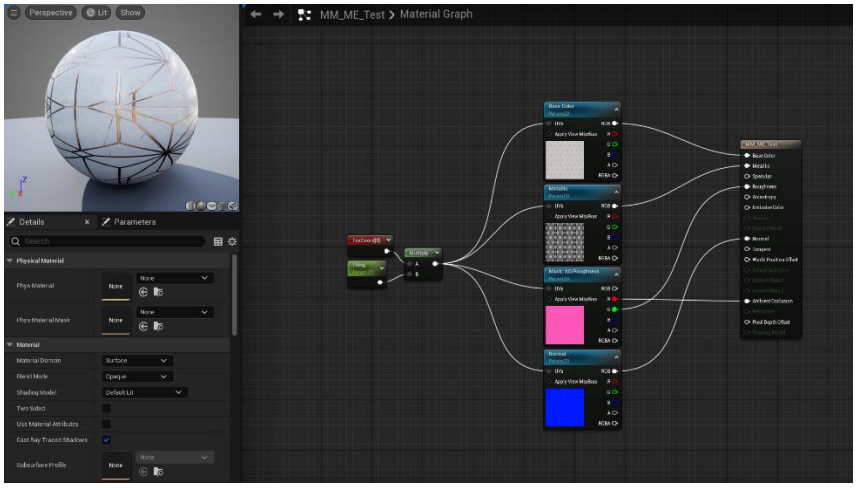

Abbildung 13: User-Interface des Material Editors. Links oben sieht man die Vorschau des Materials, das rechts mit Nodes zusammengefügt wurde.

Material Expressions sind die Grundbausteine des Editors und enthalten Code in HLSL, der für directX entwickelten Programmiersprache *High Level Shading Language*. Somit werden Shader und Materialien, welche die Engine anschließend ausführt, für 3D-Artists unsichtbar programmiert.
Material Functions sind editierbare Dateien, die verschiedene Nodes enthalten und deren Funktion daher bereits über die der mathematischen Basisfunktionen hinausgehen. Sie können selbst erstellt und der Bibliothek hinzugefügt werden, sodass sie innerhalb des Projektes in jedem Material nutzbar sind (Abbildung 14). Solche Funktionen können beispielsweise die Skalierung von Texturen oder das Animieren dieser unter Einbeziehung von Wind sein. Der Vorteil dabei ist, dass für das Implementieren im finalen Material lediglich ein einzelnes Node erforderlich ist – was schneller und übersichtlicher ist. Zudem verringert sich der Aufwand beim Verändern und Erweitern von Funktionen oder der Fehlersuche, da dies nicht für jedes Material separat erfolgen muss, sondern innerhalb der *Material Function* gelöst werden kann (Epic Games, 2022).

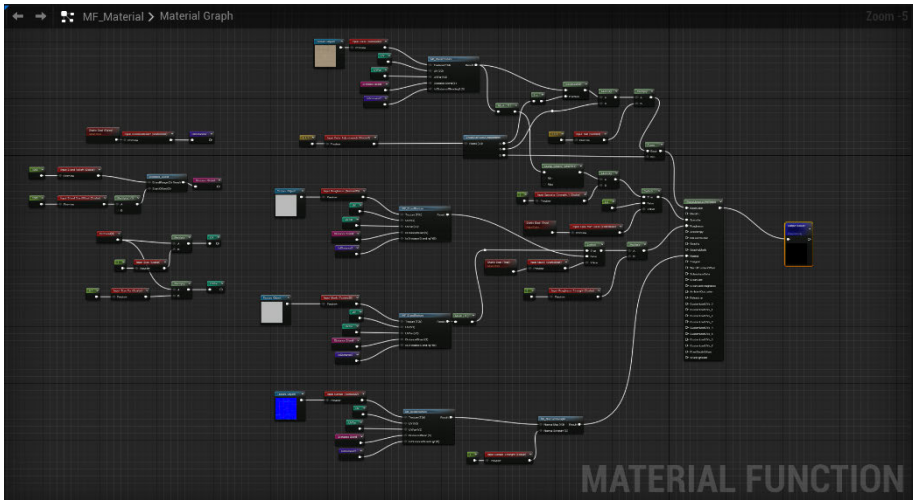

Abbildung 14: Beispiel einer *Material Function* für ein Material, in dem mehrere Texturen miteinander verblendet werden sollen. Diese Funktion kann pro Textur-Set als einzelnes Node im finalen Material eingefügt werden.

Blend Mode und Shading Model

Die Unreal Engine verfügt über verschiedene Rendertechnik-Templates für Materialien: *Blend Mode* und *Shading Model*.
Die unterschiedlichen *Blend Modes* kontrollieren, wie ein Material in Bezug auf seinen Hintergrund gerendert wird (Epic Games, 2022).
Die für diese Arbeit relevanten *Blend Modes* sind:

- „*Opaque*“ als Standardeinstellung für Wände, Böden oder Möbel.
- „*Masked*“ für Materialien mit Alpha-Channel. Der Alphawert kann nur maskiert und nicht gesteuert werden.
- „*Translucent*“ schaltet den Input für Alphawerte zwischen 0 und 1 frei.

Alle weiteren in Unreal Engine verfügbaren *Blend Modes* eignen sich überwiegend zur Gestaltung von VFX-Materialien und werden in dieser Thesis nicht behandelt.

Shading Models bestimmen, auf welche Art einfallendes Licht vom Material reflektiert wird (Epic Games, 2022). Unreal Engine bietet für verschiedene Anwendungszwecke eine ganze

Liste unterschiedlicher Reflexionsmodelle an, beispielsweise speziell für Wasseroberflächen oder Haut und Haare von Charakteren. In dieser Arbeit werden die folgenden Shading-Modelle beleuchtet:

- „*Default Lit*“ ist die Standard-Shadereinstellung und nutzt direktes und indirektes Licht sowie Reflexionen. Daher wird sie für die meisten regulären PBR-Materialien genutzt.
- „*Unlit*“ eignet sich für vollständig emittierende Materialien. PBR-Oberflächenbeschaffenheiten wie *Roughness* oder *Normal Maps* werden für diese Art von Shader nicht verwendet. Stattdessen erwecken diese Materialien den Eindruck, sie würden leuchten, ohne Licht auszustrahlen oder Schatten zu erzeugen.
- „*Two Sided Foliage*“ ist ein Shader, der für ein natürlicheres Aussehen von Pflanzen sorgt, indem er ein Durchscheinen von Licht durch die Materialoberfläche erlaubt. Dadurch wirken beispielsweise Blätter von Bäumen nicht platt oder inkorrekt beleuchtet und schattiert (Abbildung 15).

Abbildung 15: Blume mit den Shader-Modellen „*Default Lit*" (links) und „*Two Sided Foliage*" (rechts) (Epic Games, 2022).

Profiling Tools

Zur Performance-Analyse und -Optimierung werden sogenannte „*Profiling Tools*“ eingesetzt, mit denen die Rechenlast verschiedener Assets ausgeschlüsselt werden kann. So können beispielsweise GPU-intensive Assets erkannt und gegebenenfalls verbessert werden. Durch eine solche „Kostenübersicht“ kann eine bessere Balance zwischen möglichst hochqualitativer Grafik und deren Frames pro Sekunde erreicht werden.

Shader Complexity

Über View Mode gelangt man in den Modus „*Shader Complexity*“, in dem die gerenderte Szene unter Berücksichtigung der Anzahl an Shaderinstruktionen einzelner Materialien farbcodiert dargestellt wird (Abbildung) (Epic Games, 2022). Das Farbspektrum von grün bis rot bedeutet einen geringen bis hohen Rechenbedarf, während pink und weiß sehr performanceintensive Assets kennzeichnen (Abbildung 16 und 17).

Abbildung 16: *Shader Complexity View Mode* einer Szene. Verschiedene Materialen werden dort farbcodiert dargestellt (Epic Games, 2022).

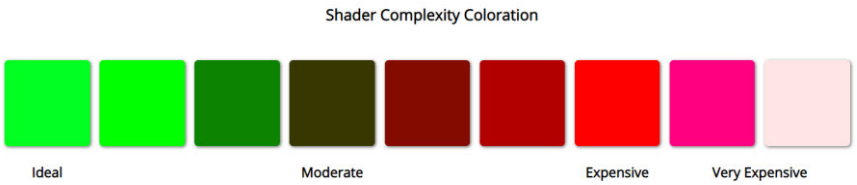

Abbildung 17: Farbwerte des *Shader Complexity View Modes* und ihre Bedeutung für die Performanceintensität einzelner Materialien (Epic Games, 2022).

Zur farblichen Darstellung einzelner Assets wird nur die absolute Anzahl der Shaderinstruktionen berücksichtigt und nicht der individuelle Rechenaufwand unterschiedlicher Shader. Daher können Materialien unter Umständen komplexer sein als in diesem *View Mode* dargestellt. Dennoch stellt das Tool eine nützliche Abschätzung der Materialkomplexität dar, durch die ein Überblick über die Performanceintensität einzelner Assets gewonnen werden kann.

Statistik-Befehle

Über die Konsole können durch verschiedene Befehle Live-Statistiken zur sichtbaren Szene eingeholt werden. Dabei sind die folgenden Befehle zur Performanceanalyse von Materialien relevant:

- **stat Unit**
  Dieser Befehl zeigt an, wie lange ein einzelnes Frame zum Rendern braucht. Dabei schlüsselt es den Wert in verschiedene Faktoren auf. Unter anderem kann dort angezeigt werden, wie viele Millisekunden die Renderzeit in der Grafikkarte beträgt (Epic Games, 2022). Je komplexer die verwendeten Materialien – und andere Elemente, die in der GPU gerendert werden, wie Animationen oder Partikeleffekte -, umso höher wird dieser Wert.

- **stat RHI**
  Mit diesem Befehl lassen sich die VRAM-Auslastung und die Anzahl der aktuell dargestellten Dreiecke sowie der *Draw Calls* anzeigen.

2. Adobe Substance 3D Suite

Anfang 2019 wurde das 2003 gegründete französische Softwareunternehmen Allegorithmic von Adobe übernommen und mit ihm alle Teile des bisherigen Substance-Softwarepakets: *Sampler* (zuvor *Alchemist*), *Designer*, *Painter* und die zugehörige Bibliothek *Substance 3D Assets* (zuvor *Source*). Diese Programme zum Erstellen und Gestalten von Texturen und Materialien haben sich zu einem Industriestandard in der Film- und Gamingbranche entwickelt. Nach eigenen Angaben nutzten 2018 mehr als 85% aller großen Spiele-Studios Substance (McDermott, Adobe, 2018). Viele namhaften Spiele wurden mit Unterstützung dieser Programme entwickelt, wie beispielsweise „Cyberpunk 2077“, die Neuauflage von „Final Fantasy VII“, „Deathloop“, „Elden Ring“ oder „Far Cry 6“. Auch Film-Blockbuster wie „Blade Runner 2049“ oder „Pacific Rim 2: Uprising“ wurden mithilfe der Substance Suite texturiert (Adobe, 2022) (Abbildung 18 und 19).

Abbildung 18: Screenshot aus Final Fantasy VII (Square Enix, 2020).

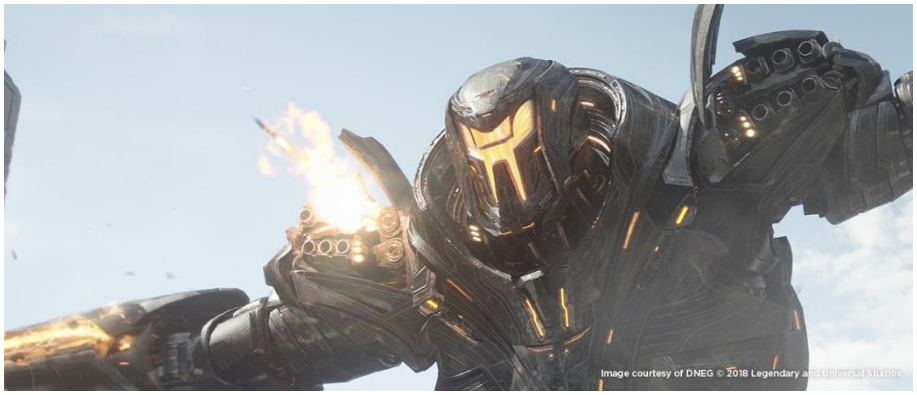

Abbildung 19: Hero-Asset aus Pacific Rim (DNEG, Legendary Entertainment, Universal Studios, 2018).

Wie bei Adobe üblich sind die Applikationen als Abonnements erhältlich:

Substance 3D Texturing	Substance 3D Collection
- Für Einzelanwender (19.03€ pro Monat) - 30 monatliche Library-Materialien - Enthält die Applikationen Painter, Sampler und Designer	- Für Einzelanwender (46,59€ pro Monat) und Teams (95,19€ pro Monat) - 50 bzw. 100 monatliche Library-Elemente (Materialien und 3D Assets) - Enthält die Applikationen Painter, Sampler, Designer und Stager

Sampler, Designer und Painter

Die in der Substance 3D Suite enthaltenen Applikationen arbeiten alle mit einem non-destruktiven Layer-Stacking-System und den eigenen Dateiformaten „SBS" und „SBSAR". SBS-Dateien sind Bearbeitungsdateien, die den Aufbau des Materials beinhalten. SBSAR-Dateien enthalten Texturen und Parameter-Informationen, die in die Unreal Engine importiert werden können.
Im Folgenden werden die drei Programme des Substance 3D Texturing Pakets näher erläutert:

- **Sampler**: Dieses Programm ist für die Generierung nahtloser Materialien aus Fotos und Grafiken geeignet. Der Prozess wird durch den Einsatz einer KI unterstützt, die automatisch Licht- und Schatteninformationen in *Normal-* und *Height-Maps* extrahiert. Die dadurch erstellten Oberflächen können mit Filtern und Effekten kombiniert und anschließend als SBS-, SBSAR- oder PNG-Dateien exportiert werden.

- **Designer**: Die Arbeitsweise in Substance Designer ähnelt der des Material-Editor in der Unreal Engine, denn auch dieser Workflow ist node-basiert (Abbildung 20).

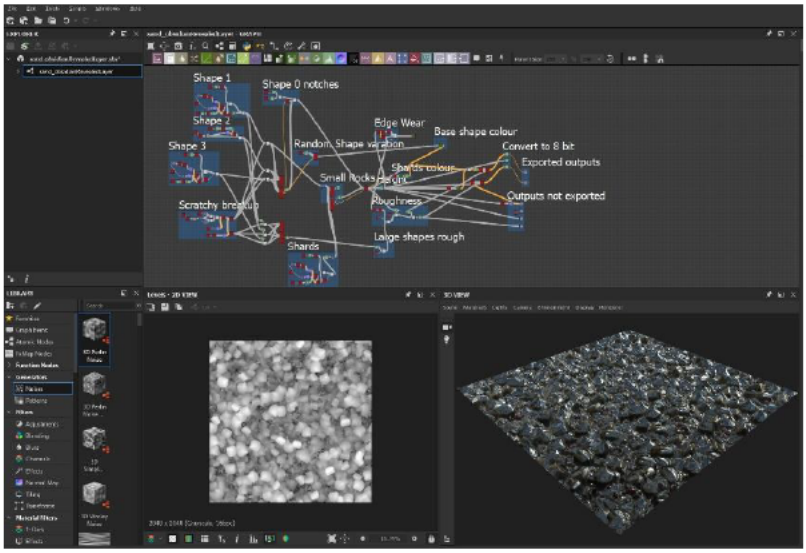

Abbildung 20: Screenshot aus Substance Designer. Zu sehen ist ein mit Nodes erstelltes Obsidian-Material für Gear 5 (The Coalition, 2019).

Aufgrund der umfangreichen Node-Bibliothek eignet sich Designer sowohl zur Erstellung vollständig prozeduraler Texturen, als auch zur Bearbeitung bestehender Materialien. Geeignete Nodes des *Material Graphs* können als Parameter gekennzeichnet werden. Diese werden in den SBSAR-Dateien als veränderliche Variablen gespeichert.

- **Painter**: Diese Applikation eignet sich vor allem zur Erstellung und Bearbeitung UV-gebundener Texturen (Abbildung 21).

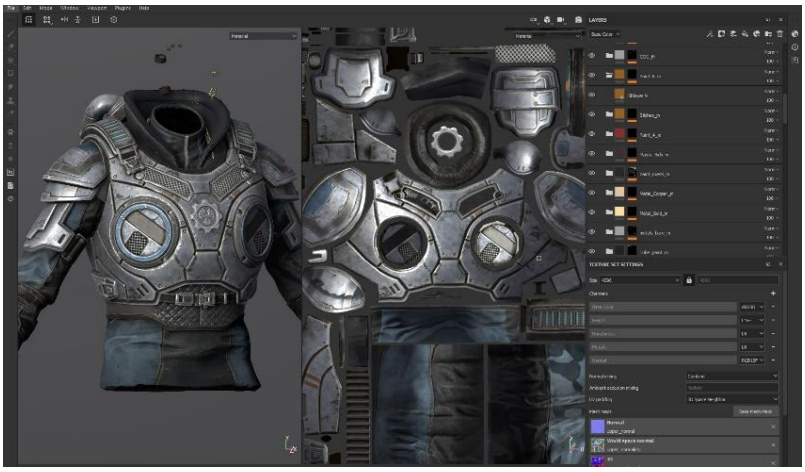

Abbildung 21: Texturierung einer Rüstung für Gear 5 in Substance Painter (The Coalition, 2019).

Ein wichtiges Feature von Painter ist die *Baking*-Funktion, die Texturen wie beispielsweise *Normal Maps* aus einem Mesh erstellt. Für die Gestaltung der Materialien werden Filter, Generatoren, *Smart Materials* und Pinsel benutzt. Der Export erfolgt als Texturen im PNG-Format. Dafür gibt es Presets, die bei Bedarf manuell angepasst werden können. Ein Unreal Engine-Preset ermöglicht den Export einer *Channel Packed Map*.

3. Quixel

2019 übernahm Epic Games das 2011 gegründete Unternehmen Quixel und dessen Software-Paket: *Megascans Library*, *Mixer* und *Quixel Bridge*. Aus Letzterer wurde ein Plugin entwickelt, das die Bibliothek direkt mit Unreal Engine 5 verbindet. Die Nutzung aller Materialien der Library, der Bearbeitungssoftware Mixer und des Bridge-Plugins ist in Kombination mit der Unreal Engine kostenlos.
Materialien aus Quixel tauchen in Produktionen namhafter Unternehmen wie Disney, Square Enix und Electronic Arts auf (Abbildung 22 und 23).

Abbildung 22: Screenshot aus "Shadow of the Tomb Raider" (Square Enix, 2021).

Abbildung 23: Ausschnitt aus "Der König der Löwen" (The Walt Disney Company, 2019).

Mixer

Die Software Quixel Mixer ist ein Texturierungsprogramm für Assets und daher hauptsächlich für UV-gebundene Materialien geeignet. Zusätzlich ist die Bearbeitung von Library-Assets auf einer *Plane* möglich.
Zur Gestaltung von Assets stehen insgesamt 18 Masken und Filter zur Verfügung, die in einem non-destruktiven *Layer Stacking*-System miteinander verblendet werden. Der Export fertiger Materialien erfolgt als PNG-Dateien. Presets für *Channel Packed Maps* oder den Export eines Alpha-Kanals gibt es nicht. Diese können selbst erstellt und gespeichert werden (Abbildung 24).

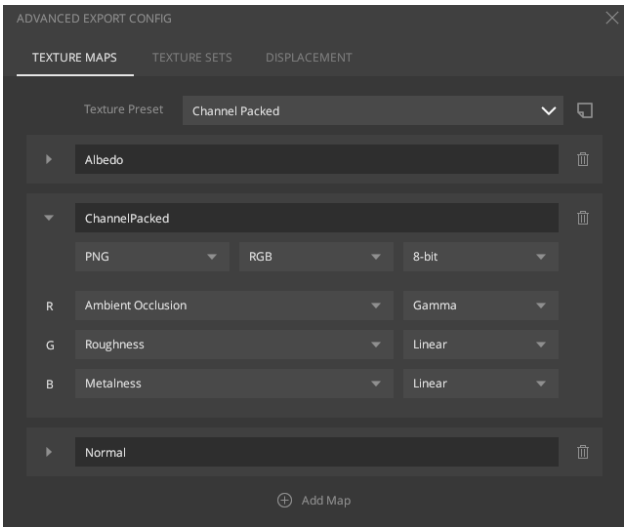

Abbildung 24: Manuell erstelltes *Channel Packed*-Preset in Quixel Mixer.

4. Softwareanforderungen

Die Wahl der richtigen Software hängt von den Anforderungen des jeweiligen Projektes ab. Insbesondere bei einem kleinen Budget ist ein Vergleich der Alternativen unerlässlich. Wichtige Vergleichsaspekte sind technische Features der Applikationen, ebenso relevant ist die Usability für 3D-Artists. Daraus ergeben sich folgende Fragestellungen, die im Laufe des Vergleiches geklärt werden müssen:

- **Können beide Softwares mit einem äquivalenten Ergebnis aufwarten?**
  Die texturierten Assets sollten im subjektiven, optischen Qualitätseindruck und ihrer Performance miteinander vergleichbar sein.
- **Können alle angeforderten Materialien erstellt werden? Welche Unterschiede in den Möglichkeiten der Applikationen gibt es?**
  Für einen Vergleich zweier Applikationen müssen vorab die konkreten Anforderungen des Projektes geklärt werden. Anschließend kann ermittelt werden, welche Funktionen für die benötigten Materialien erforderlich sind und unter welchen Einschränkungen diese in den Programmen verfügbar sind.
- **Wie groß ist der Zeitaufwand?**
  Der Preis einer Software setzt sich aus verschiedenen Faktoren zusammen, die bestimmen, wie wirtschaftlich ihre Nutzung für ein Projekt ist. Ein wichtiger Aspekt

sind Einarbeitungszeit und Komplexität in der Bedienung. Ein wichtiges ökonomisches und psychologisches Argument ist auch, ob das Team schon Erfahrung mit einer Anwendung hat und ob sich der Wechsel auf ein anderes System lohnt.

- **Wie problemlos ist die Integration in Unreal Engine 5?**
  Der Import der Materialien sollte möglichst reibungslos funktionieren. Außerdem sind Anpassungsmöglichkeiten in der Engine vorteilhaft.

4. Praxistest

Ziel dieser Arbeit ist es, die vorgestellten Softwares auf ihre Funktionalität zur Erstellung von Materialien für Spiele zu prüfen. Um einen realistischen Vergleich zu ermöglichen, müssen zuerst für beide Applikationen gleiche Grundvoraussetzungen in der Engine geschaffen werden. Dazu wird ein Unreal Engine-Projekt mit Master-Materialien, *Landscape* und thematisch passender Beleuchtung erstellt. Dann wird eine Szene für ein Computerspiel erstellt und dupliziert, damit lediglich die Materialien aus Quixel und Substance eingefügt werden können. Diese Szene wird vorab so konzeptioniert, sodass sie möglichst viele unterschiedliche, für das Spiel relevante Materialien enthält.

Im Folgenden werden die einzelnen Schritte des Setups sowie die angeforderten Materialien erläutert. Im Anschluss werden Inhalte der Libraries sowie Workflows zur Materialerstellung unter Nutzung der beiden Tools verglichen sowie ein Performancetest durchgeführt. Anhand einiger Beispiele werden konkrete Unterschiede in den Funktionen von Substance und Quixel hervorgehoben.

1. Planung und Setup der Szene

Für die Szene wird ein mediterraner Innenhof gewählt, dessen Baustil arabische Elemente aufweist (Abbildung 25).

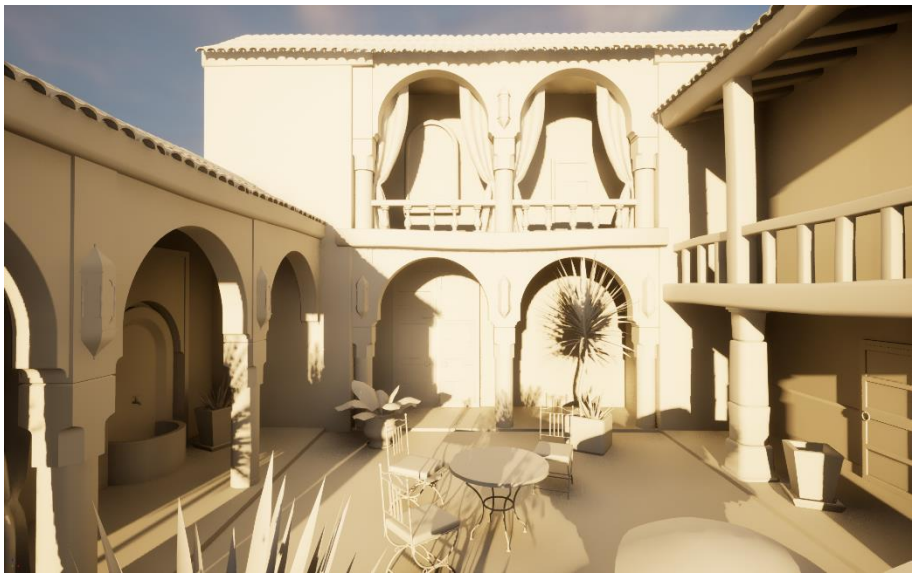

Abbildung 25: Blocking der Szene.

Der Boden wurde als *Landscape* eingefügt, sodass Materialien darauf verblendet werden können. Die verwendeten 3D-Assets wurden selbst modelliert oder stammen aus dem Asset-Paket „*Middle Eastern Courtyard*“ von Joakim Stigsson und der Megascans Library. Für alle Objekte wurde die Texel-Dichte auf 512 px/m angepasst.

Materialien

Die gewählten Materialien sollen zum Baustil des Gebäudes passen. Für einen umfassenden Vergleich von Substance und Quixel sollen möglichst viele unterschiedliche Materialtypen verwendet werden. Die Konzeption des Innenhofes sieht folgende Materialien vor:

- **Mauerstein**: Für die Wände des mittleren Gebäudes sollen auf der gesamten Fläche überstrichene Mauersteine verwendet werden.
- **Steinblöcke und Natursteinmauer**: Die Säulenkonstruktionen des rechten Gebäudes werden aus rauem grauem Stein bestehen. Diese sollen leichte Beschädigungen und poröse Oberflächen aufweisen. Der untere Teil der rechten Fassade besteht aus Naturstein.
- **Marmor**: Glatter Marmor soll den Sockel der Hufeisenbögen des linken und mittleren Gebäudes bilden. Auch Teile des Brunnens sollen aus diesem Material bestehen.
- **Mosaik und Fliesen:** Oberhalb der Marmorsockel soll ein farblich passendes Mosaik am Übergang zum oberen Teil des Bogens entstehen. Für Gehwege und Beet-Umrandung werden Fliesen gewählt.
- **Holz**: Die Türen, Deckenkonstruktionen und Balken sollen aus dunklem Holz mit wetterfester Lackierung bestehen. Zusätzlich sollen die Kassettenelemente der Türen mit Schnitzereien verziert werden.
- **Metall**: Die Tisch- und Stuhlbeine, sowie Türgriffe und der Wasserhahn bestehen aus schwarzlackiertem Metall.
- **Terracotta/Keramik:** Die Pflanzgefäße werden mit einem Terracotta-Material texturiert.
- **Erde und Sand:** Für das Beet wird trockene rötliche Erde verwendet. Dies gilt auch für die Landscape, auf der sie mit Sand verblendet werden soll.
- **Textilien:** Die Sitzkissen der Stühle sollen aus gewebtem Stoff mit Streifenmuster bestehen. Für die Vorhänge soll ein feiner Baumwollwebstoff mit Lochstickerei oder Spitze verwendet werden.
- **Putz:** Die Wände der seitlichen Gebäude werden verputzt und gestrichen.
- **Foliage / Pflanzen:** Für die Palmen sollen Atlanten aus den Libraries verwendet und als Foliage-Material appliziert werden.

Zur Umsetzung werden folgende Master Materialien erstellt, die später fürs Implementieren der Texturen beider Softwares verwendet werden können:

- *M_Base_SingleMaps* ist ein Master-Material für alle Standardmaterialien, die einzelne *Texture Maps* für *Roughness*, *Metallic* und *Ambient Occlusion* enthalten. Diese Maps können auch auf Vektorwerte umgestellt werden.
- *M_Base_Mask* ist das Master-Material für *Default Lit*-Materialien mit *Channel Packed Map* anstelle einzelner Texturen.
- *M_Emissive* bildet das Master-Material für einfache, selbstleuchtende *Unlit*-Materialien.

- *M_Translucent* rendert eingestellte Texturen mit dem *Blend Mode „Translucent*“ und enthält eine Einstellung zur beidseitigen Darstellung der Texturen auf einer Fläche. Dadurch ist auch die Rückseite sichtbar.
- *M_Foliage* dient der Implementierung von Materialien für Pflanzen in der Engine. Durch den *Blend Mode „Masked*“ werden lediglich weiß maskierte Teile gerendert. Das Shading-Modell „*Two Sided Foliage*“ optimiert dies für durchscheinende Blätter und Blüten. Auch dieses Material wird beidseitig dargestellt.
- *M_Landscape* ist das Master-Material für den erstellten Boden. Mit diesem können bis zu drei verschiedene Materialien im *Landscape Paint Mode* miteinander verblendet werden.

Beleuchtung

Die Szene wird durch ein einfaches Lumen-Setup beleuchtet (Abbildung 26). Dafür werden ein *Sky Light* und ein *Directional Light* verwendet. Zudem kommen eine *Sky Atmosphere* für die Färbung des Himmels und *VolumetricCloud* als *Visual Effect* mit animiertem Wolkenmaterial zum Einsatz. Dadurch lässt sich einerseits optisch eine realistischere Lichtsituation schaffen, andererseits können Materialien, die Himmel und Wolken reflektieren, die Renderzeit beeinflussen.

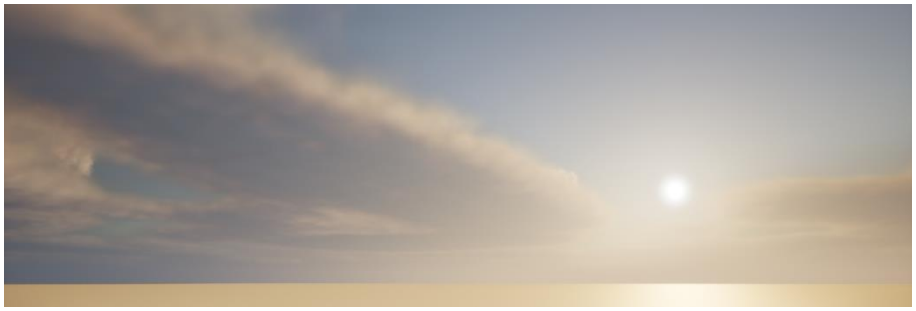

Abbildung 26: Licht-Setup der Szene.

Setup der Applikationen

Das Plugin für Substance wird über den Unreal Engine-Marketplace heruntergeladen und für die genutzte Engine-Version installiert. Nach der Aktivierung in den Plugin-Optionen stehen Library und *Substance Presets* zur Verfügung.

Das Quixel Bridge-Plugin ist in Unreal Engine automatisch installiert und aktiviert. Es kann direkt auf die Library und die Presets zugegriffen werden.

2. Library-Inhalte

Für diese Bachelorarbeit wurde die Einzelentwicklerlizenz „Substance 3D Texturing“ abgeschlossen. Daher erfolgt der Vergleich zwischen den dort enthaltenen und den Megascans-Assets.

Elementarer Bestandteil der Applikationen sind die zugehörigen Bibliotheken. Auch wenn die Quixel Megascans-Library mit 17.000 Assets rein quantitativ größer als die 11.000 Elemente große Substance 3D Asset Library ist, sind beide sehr umfangreich.
Zur Erleichterung der Materialsuche arbeiten beide Bibliotheken mit Tags. Diese sind intern für jedes einzelne Objekt vergeben und enthalten Informationen über den Zustand, Anwendungsbereich, Farbe oder Art des Assets. Das System funktioniert bei beiden sehr gut, denn alle für die Szene gesuchten Materialien konnten schnell und problemlos gefunden werden. Die Assets sind hochauflösend bis zu 8 K erhältlich.

Die Bibliotheken teilen sich eine sehr große Schnittmenge an natürlichen Materialien wie Stein, Holz, Gras, Erde oder Sand. Auch typische Baumaterialien wie Fliesen, Mauersteine, Keramik, verputzte Oberflächen oder Holzfurnier sind enthalten. Unterschiedliche Metallarten, Textilien oder Decal-Materialien findet man ebenfalls.

Trotz der Gemeinsamkeiten unterscheiden sich die Bibliotheken von Substance und Quixel in ihren Angeboten:

Die **Substance 3D Assets Library** enthält sowohl gescannte als auch prozedural erstellte Materialien. Dieser prozedurale Aufbau ermöglicht eine enorme Flexibilität in der parametrischen Anpassung der Materialien. Das macht sich auch im Materialdesign bemerkbar: Komplexe Muster, Stoffe mit anpassbaren Perlenstickereien oder kreativ gemusterte Verpackungen mit Papieroberfläche werden dadurch ermöglicht (Abbildung 27). Das macht die Substance Bibliothek sehr bunt, ohne farblich festgelegt zu sein. Zusätzlich gibt es eine Auswahl an transluzenten und selbstleuchtenden Materialien.

Abbildung 27: Detaillierte, prozedural erstellte Muster-Materialien in der Substance 3D Asset Library.

Die Quixel Library **Megascans** enthält ausschließlich gescannte Materialien. Dabei liegt ihr Schwerpunkt auf Natur und verlassenen Bauwerken. Das macht sich sowohl in den Oberflächenmaterialien als auch in den Atlanten bemerkbar. Diese sind in höchstem Detailgrad und enormer Anzahl vorhanden (Abbildung 28).

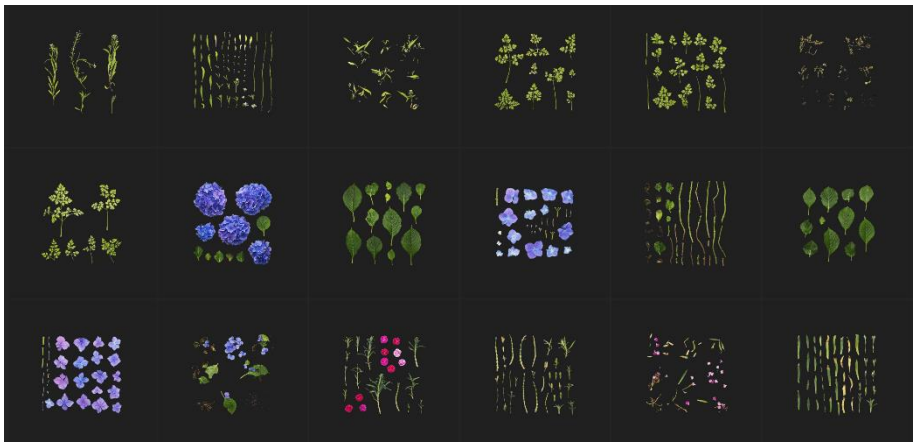

Abbildung 28: Detaillierte Scans von Pflanzen in der Quixel Megascans Library.

Zusätzlich gibt es vollständig texturierte 3D-Assets, die per Drag & Drop in Unreal Engine genutzt werden können. Diese sind in verschiedenen *Level of Detail*-Stufen vorhanden und eignen sich durch die schnelle Implementierung in die Engine sowohl zum Level Blocking als auch als Game Assets. Die 3D-Asset Kategorie beinhaltet große Gebirge, Gebäude, Pflanzen, Möbel und kleinere Szenendekorationen.

3. Material Presets in Unreal

Zur Verwendung von Assets direkt aus der Library werden mit den Plugins *Master Material Presets* installiert. Da dieser Grundaufbau der Materialien in der Engine für die *Shader Complexity*-Farbe verantwortlich ist, kann eine Einschätzung zur Performanceintensität bereits jetzt getroffen werden.

Substance-Presets

Substance stellt zum Import seiner Materialien drei Unreal-Templates zur Verfügung, die im „Substance in UE5 Content"-Ordner gespeichert sind. Aus diesen werden beim Import von Substance-Materialien im SBSAR-Dateiformat Materialinstanzen erstellt. Dabei kann man im Importfenster frei zwischen „*Refraction_Template*", „*Standard_Template*" und „*TriPlanar_Template*" wählen.

- „**Standard_Template**" enthält Platzhalter-Inputs für alle gängigen Texturen sowie Parameter zur Materialskalierung. Im *Shader Complexity View Mode* wird ein Material, das dieses Preset nutzt, hellgrün dargestellt (Abbildung 29).

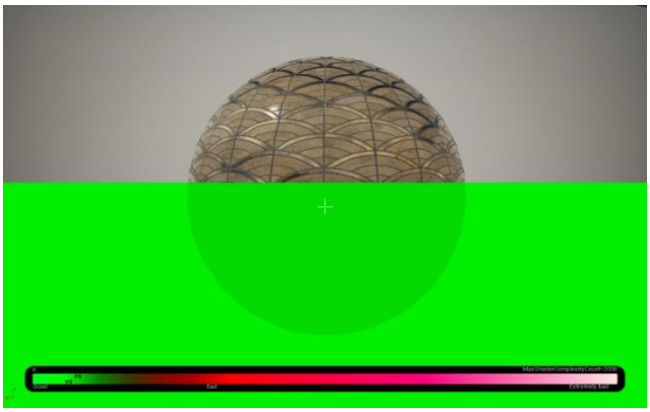

Abbildung 29: *Shader Complexity View Mode* des Material-Presets für Standardoberflächen.

- „**TriPlanar_Template**" kommt zum Einsatz, wenn in der Vorbereitung keine UV erstellt wurde. Das Material wird nur auf Grundlage des Meshes passend positioniert. Zur idealen Platzierung sind im Master Material mehr Einstellungsmöglichkeiten eingerichtet als im „Standard_Template". Daher ist die Farbe im *Shader Complexity View Mode* dunkler (Abbildung 30).

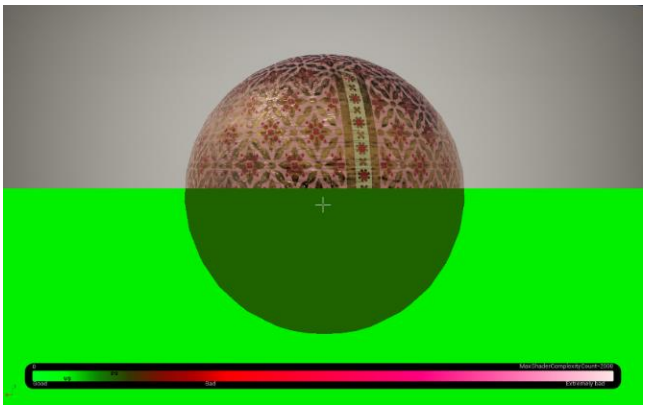

Abbildung 30: *Shader Complexity View Mode* für„TriPlanar_Template"-Materialien.

- „**Refraction_Template**“ ist speziell für Glasmaterialien. Dieses arbeitet mit einem Transluzenz-Shader und einer Fresnel-Funktion, weshalb es zu den sehr rechenintensiven Materialien zählt (Abbildung 31).

Abbildung 31: *Shader Complexity View Mode* für „Refraction_Template"-Materialien.

Die Einstellungsmöglichkeiten in den Unreal-Materialinstanzen beschränken sich auf Positions- und Skalierungsparameter. Da das Material für die „TriPlanar“-Projektion komplexer aufgebaut ist, kann es detaillierter an das Mesh angepasst werden.
Zusätzlich zur Instanz für den Grundaufbau des Materials in der Engine wird mit jeder SBSAR-Datei eine sogenannte „*Substance Graph Instance*“ importiert. Diese lädt ein eigenes User-Interface für Materialeinstellungen (Abbildung 32). Darin hat man Zugriff auf alle in Substance Designer freigegebenen Parameter.

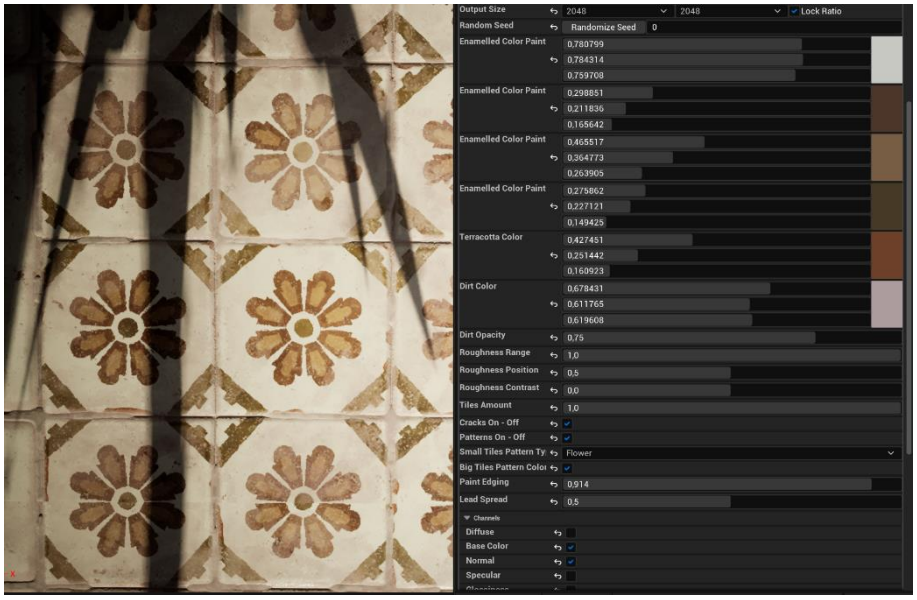

Abbildung 32: Fliesenmaterial mit zugehörigem User-Interface der „*Substance Graph Instance*".

Quixel-Presets

Mit dem ersten Import eines Megascans-Assets wird im *Content Browser* der neue Ordner „*MSPresets*“ für Quixel-Templates erstellt. Beim Einfügen eines Objektes wird automatisch eine Instanz aus dem dafür vorgesehenen Master-Material erstellt. Per Default sind neun Presets installiert:

- „**MS_Surface_Material**“ ist das Standard-Master-Material für die meisten Oberflächen. Im *Shader Complexity View Mode* erscheint es grün (Abbildung 33).

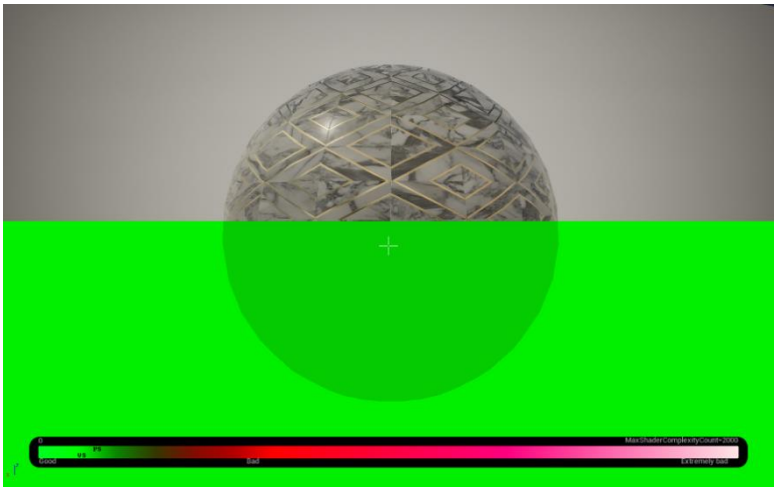

Abbildung 33: *Shader Complexity View Mode* des Material-Preset für "Surface"-Materialien.

- „**MS_Decal_Material**“ ist das Template für Decals, die im Unreal Engine-Viewport per Drag & Drop skaliert werden können. Diese sind im *Shader Complexity View Mode* vollständig hellgrün (Abbildung 34).

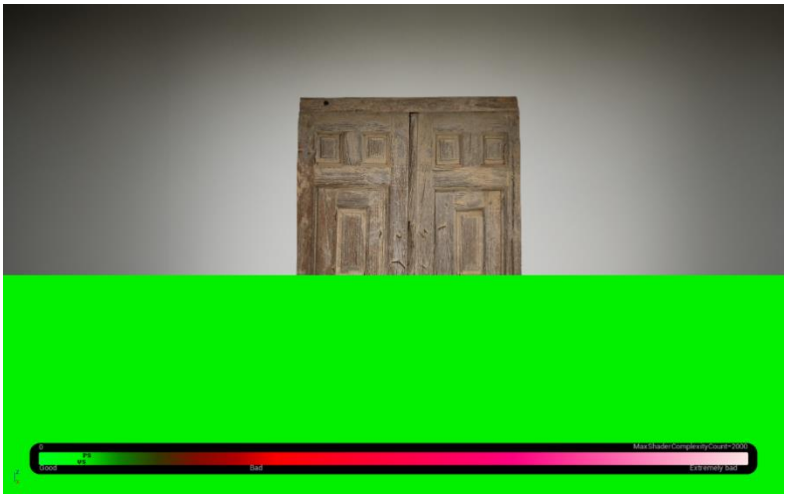

Abbildung 34: Decal-Material im *Shader Complexity View Mode.*

- „**MS_Foliage_Material**“ wird bei importierten 3D-Pflanzen verwendet. In diesem lassen sich Parameter wie Windintensität für bewegte Materialien steuern (Abbildung 35).

Abbildung 35: Einstellungsmöglichkeiten für bewegte Materialien in der Materialinstanz des Foliage-Presets.

Trotz Einstellungsmöglichkeiten für Wind-Animationen ist das Template für 3D-Pflanzen aus der Megascans-Bibliothek im *Shader Complexity View Mode* grün (Abbildung 36).

Abbildung 36: Shader Complexity von Megascans 3D-Assets mit windbewegtem Material.

- Weiterhin gibt es Templates für besondere Oberflächenbeschaffenheiten, die mit speziellen Shader-Einstellungen wie „*Cloth*“ oder „*Subsurface*“ arbeiten. Diese werden in dieser Bachelorarbeit nicht verwendet.

Für Quixel-Materialien sind alle Einstellungsmöglichkeiten im Master-Material und damit in der Unreal-Materialinstanz enthalten. Dort sind Positions-, Skalierungs- und Farbanpassungseinstellungen für Texturen freigegeben. Außerdem kann auf *Roughness*-, *Specularity*-, *Normal*- und *Metallic*-Werte zugegriffen werden.

4. Verwendung von Materialien direkt aus der Library

Für den mediterranen Innenhof sollen Materialien direkt aus beiden Bibliotheken importiert und verwendet werden. Dort werden sie mit den dafür vorgesehenen Einstellungsparametern der Templates angepasst.

Beet

Das Beet im Zentrum des Innenhofs besteht aus zwei Materialien, die direkt importiert werden: terracottafarbene Fliesen und rotbraune Erde mit Steinchen. Die Mesh-Elemente sind in Abbildung 37 farblich markiert.

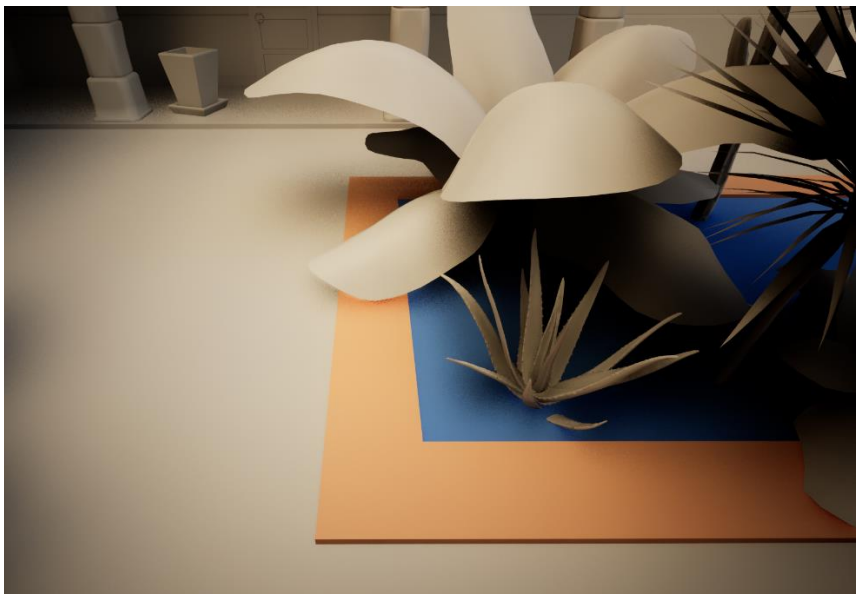

Abbildung 37: Das Beet vor der Texturierung. Zur besseren Sichtbarkeit sind die Mesh-Teile markiert.

Als erster Schritt werden aus beiden Libraries Materialien herausgesucht:

- Für die Substance-Szene wird das Fliesen-Material „Sand_Ceramic_Tiles“ als SBSAR-Datei heruntergeladen und per Drag & Drop in den Content-Browser der Engine geladen. Dort erstellt das Substance-Plugin nach Einstellung des Import-Templates Unreal-Materialinstanz und *Substance Material* Graph. Außerdem erhält man Zugriff auf die PNG-Dateien der Texturen.
- Für die Quixel-Szene wird das Material „Square_Floor_Tiles“ heruntergeladen. Über einen Pfeil in der Bridge kann es nach der Auswahl der richtigen Texturgröße direkt in die Engine gesendet werden. Dabei wird ein neuer Ordner erstellt, indem Materialinstanz und Texturen als PNG-Dateien enthalten sind.

Bei der Anpassung des Fliesenmaterials ist es wichtig, die Fugen bündig an den Kanten des Meshes zu platzieren. Andernfalls erhält man unnatürliche Legemuster. Daher ist eine gute Kontrolle über die Position und die Größe der Textur erforderlich.

- Das Substance Material lässt sich über seine Unreal-Instanz entlang beider Achsen unabhängig verschieben und skalieren. In der *Substance Graph Instance* können

weitere Anpassungen vorgenommen werden. Diese beinhalten beispielsweise Parameter für Farben, Muster, Oberflächenprofil und Beschädigung der Kanten (Abbildung 38). Außerdem wird über dieses Interface die Texturgröße so eingestellt, dass sie zur angestrebten Texel-Dichte passt.

Abbildung 38: Substance Fliesen-Material. Links mit aktivierter Beschädigung der Kanten, rechts ohne.

- Das Quixel-Material kann ebenfalls in der Unreal-Materialinstanz über beide Achsen unabhängig skaliert und verschoben werden. Die Farbanpassung für das gesamte Material erfolgt ebenfalls dort. Auch *Roughness* und *Normal Intensity* können in der Instanz verändert werden.

Für die Erde wurden das Substance-Material „Rubble_Covered_Soil“ und das Megascans-Material „Brick_Debri_On_Ground“ verwendet. Diese wurden ebenfalls importiert und in den Unreal-Instanzen platziert. Eine Anpassung der Materialgröße proportional zur Umgebung war nötig, ebenso wie eine Rotation, um Wiederholungen im Steinchen-Muster durch geschicktes Platzieren verschwinden zu lassen.

Abbildung 39: Fertig texturiertes Beet-Asset. Links die Materialien aus der Quixel -Szene, rechts aus der Substance-Szene.

Die Texturierung direkt aus den Libraries war sowohl für Substance als auch für Quixel problemlos möglich. Durch die zusätzliche *Substance Graph Instance* ermöglicht Substance eine vollständige Kontrolle über alle freigegebenen Parameter, wodurch die Anpassung sehr präzise erfolgen kann. Wird bei Quixel mehr Anpassungsspielraum benötigt, als die Unreal-Materialinstanz bieten kann, muss auf die Bearbeitungssoftware Mixer zurückgegriffen werden.

Landscape

Für Landscape-Materialien gibt es keine Presets, daher muss das Master Material selbst erstellt werden. In dessen Materialinstanz können drei verschiedene Textur-Sets hinzugefügt werden. Diese werden anschließend im *Landscape Paint Mode* auf dem Landscape-Material miteinander verblendet. Die von den Plugins erstellten Unreal Instanzen werden nicht benötigt.

- **Substance:**
  Das Substance-Plugin importiert alle Texturen einzeln. Man erhält für jedes Material eine *Base Color*, *Ambient Occlusion*, *Metallic*, *Normal* und *Roughness*-Textur. Die benötigten Texturen können innerhalb der Engine unabhängig von den zugehörigen Instanzen verwendet werden. Daher lassen sie sich per Drag & Drop in das Landscape-Material ziehen. Die Parameter der *Substance Graph Instance* funktionieren auch, wenn sie außerhalb des Unreal-Templates von Substance genutzt werden.
- **Quixel:**
  Die Texturen aus Megascans sind in der Reihenfolge *Ambient Occlusion*, *Roughness* und *Metallic Channel Packed*. Somit erhält man zusammen mit der *Base Color* und *Normal Map* insgesamt drei Texturen. Die *Channel Packed* Textur ist im *Texture Editor* bereits als „Masks (no sRGB) " formatiert.

Aus beiden Bibliotheken wurden folgende Materialien verarbeitet:

- Trockene Erde mit Steinchen oder Stöckchen
- Trockene Erde mit sichtbaren Rissen
- Sand mit leichter Hoch-Tief-Struktur

Abbildung 40: Verblendetes Landscape Material mit Substance-Materialien.

Abbildung 41: Verblendetes Landscape Material mit Quixel-Materialien.

Auch bei Landscape-Materialien ist die gute Kontrolle über die Parameter in der *Substance Graph Instance* von Vorteil. Alle Anpassungen, die an Quixel-Texturen vorgenommen werden müssen, benötigen zusätzliche Parameter im Master Material oder eine Änderung in Mixer.

5. UV-gebundene Materialien

Die Gestaltung von UV-gebundenen Materialien erfolgt in Substance Painter und Quixel Mixer. Einzelne wichtige Funktionen werden am Beispiel des Brunnens in beiden Applikationen demonstriert. Dieser besteht aus einem Mesh und soll mit vier verschiedenen Materialien texturiert werden (Abbildung 42). Der Wasserhahn wird dasselbe Material wie die Metall-Elemente der Stühle und Türklinken erhalten, daher muss er nicht bearbeitet werden.

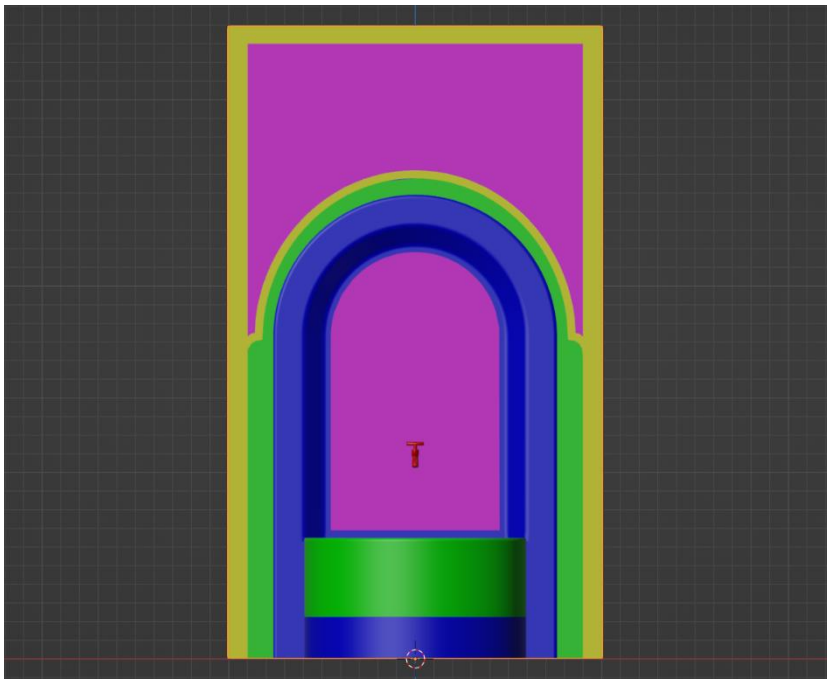

Abbildung 42: Die Platzierung der Materialien wurde mit *Vertex Colors* markiert.

Setup

In **Substance Painter** wird ein neues Projekt gestartet und das Mesh geladen. Anschließend werden *Normal* und *ID Map* gebacken. Das funktioniert auch ohne ein zusätzliches *High Poly Mesh* und ist wichtig, damit Filter und Generatoren die Kanten richtig erkennen. Dann kann damit begonnen werden, Materialien auf das Objekt zu applizieren.

Für **Quixel Mixer** wird ein neuer Mix angelegt und das Mesh geladen. Da kein *Baking* möglich ist, gibt es mehrere Optionen, um die einzelnen Farbflächen zu trennen:

- Maskierung mit *Paint Layern*
- Import des Meshes mit verschiedenen Materialien anstelle von *Vertex Colors*
- Erstellen einer ID Map außerhalb von Mixer und deren Import in die Bearbeitungssoftware

Da externe Softwares nicht Teil des Vergleichs zwischen Substance und Quixel sind und eine Maskierung von Hand einen enormen Mehraufwand bedeutet, werden mehrere Materialien angelegt. Das Mesh enthält nun 4 Materialien, die nach der Bearbeitung als separate Texturen exportiert werden. Damit ist auch dieses Mesh bereit zur Texturierung.

Hinzufügen der Materialien

In **Substance Painter** können Materialien per Drag & Drop auf das Mesh oder den *Layer Stack* gezogen werden. Da eine *ID Mask* benutzt wird, muss für jedes Material einzeln eine Auswahl getroffen werden, welcher *Vertex Color* beeinflusst wird. Das funktioniert durch das Hinzufügen einer *„mask with color selection“* (Abbildung 43). Damit kann die Farbe direkt aus der *ID Mask* ausgewählt werden.

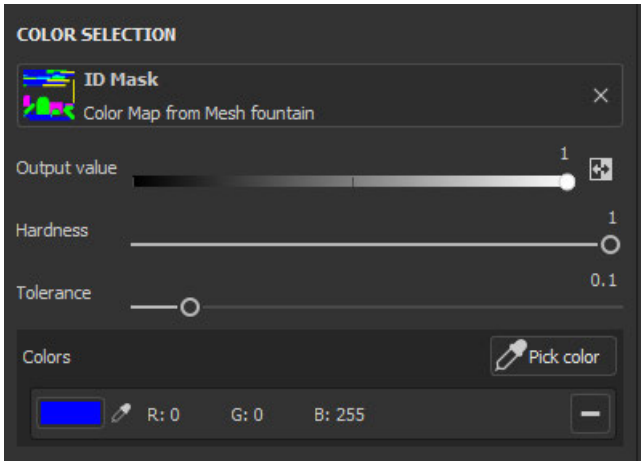

Abbildung 43: "*Add mask with color selection*"-Filter zur Auswahl eines *Vertex Colors*.

Diese Auswahl funktioniert auch für gruppierte Layer. Werden mehrere Materialien, Filter und Generatoren gestapelt, können sie dadurch gemeinsam einem *Vertex Color* hinzugefügt werden.

In **Quixel Mixer** wird mit mehreren Einzelmaterialien gearbeitet, die in der 2D-Ansicht als einzelne UVs angezeigt werden (Abbildung 44).

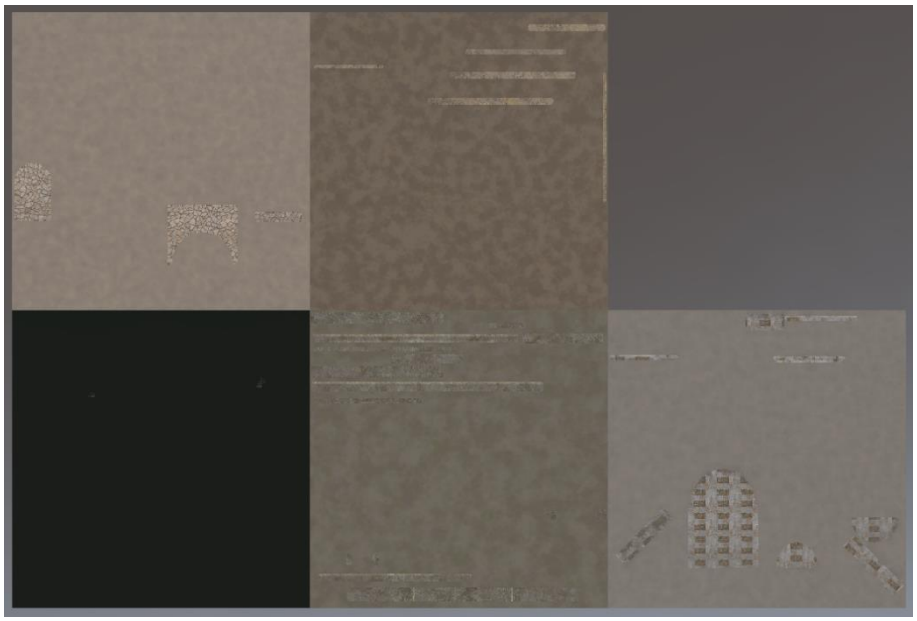

Abbildung 44: Aufteilung des Meshes in je eine UV pro Material.

Im *Layer Stack* werden Ordner angelegt, denen die jeweils betroffene UV zugewiesen wird. Alle Materialien und Filter, die nun per Drag & Drop einem der Ordner hinzugefügt werden, beeinflussen nur das eingestellte Material. Für die Realisierung von Details wie Verschmutzungen kann ein weiterer Ordner angelegt werden, der alle UVs gleichzeitig betrifft (Abbildung 45).

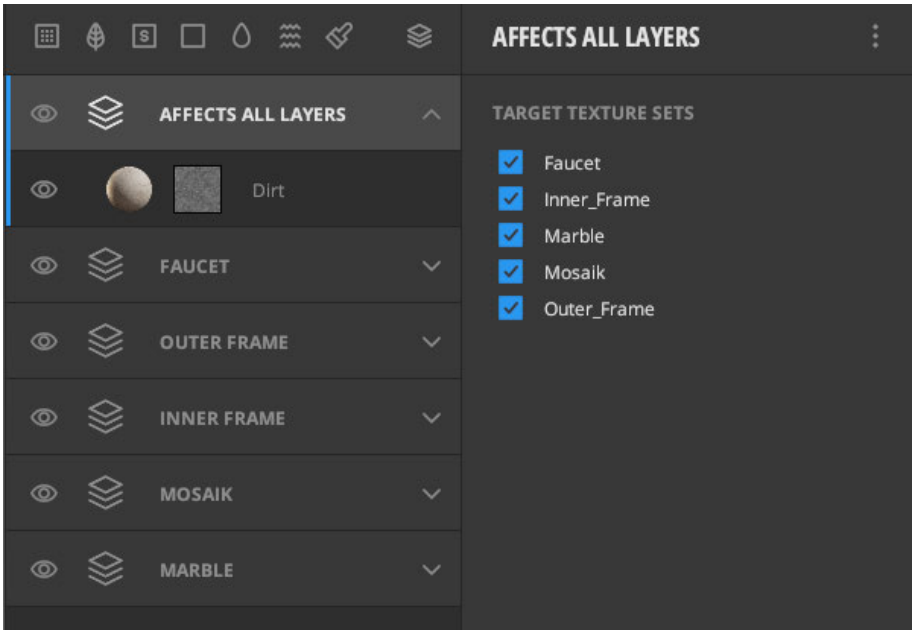

Abbildung 45: Quixel-Material-Ordner mit Zuweisung der beeinflussten UVs.

Bearbeitung

Die Gestaltung der Materialien auf dem Mesh erfolgt in beiden Softwares als non-destruktives *Layer-Stacking*. Dennoch gibt es Unterschiede in der Herangehensweise und Umsetzung. Dies wird im Detail anhand des inneren Brunnen-Rahmens gezeigt.

In beiden Applikationen wurde ein Marmormaterial zur Bearbeitung gewählt. Dieses wurde auf die gewünschte Stelle appliziert. In den weiteren Schritten wurden Kantenbeschädigung, Staub und Schmutz hinzugefügt. Dabei sollten manche Effekte nur das einzelne Material betreffen, andere sollen sich über das gesamte Mesh erstrecken. Ein weiteres Detail ist ein dezentes Moos-Material, das an manchen Stellen mit dem Marmor verblendet werden soll.

Zur besseren Sichtbarkeit der Effekte wurden diese in ihrer Maskierung dargestellt.

Abbildung 46 zeigt die Bearbeitung mit **Substance Painter**:

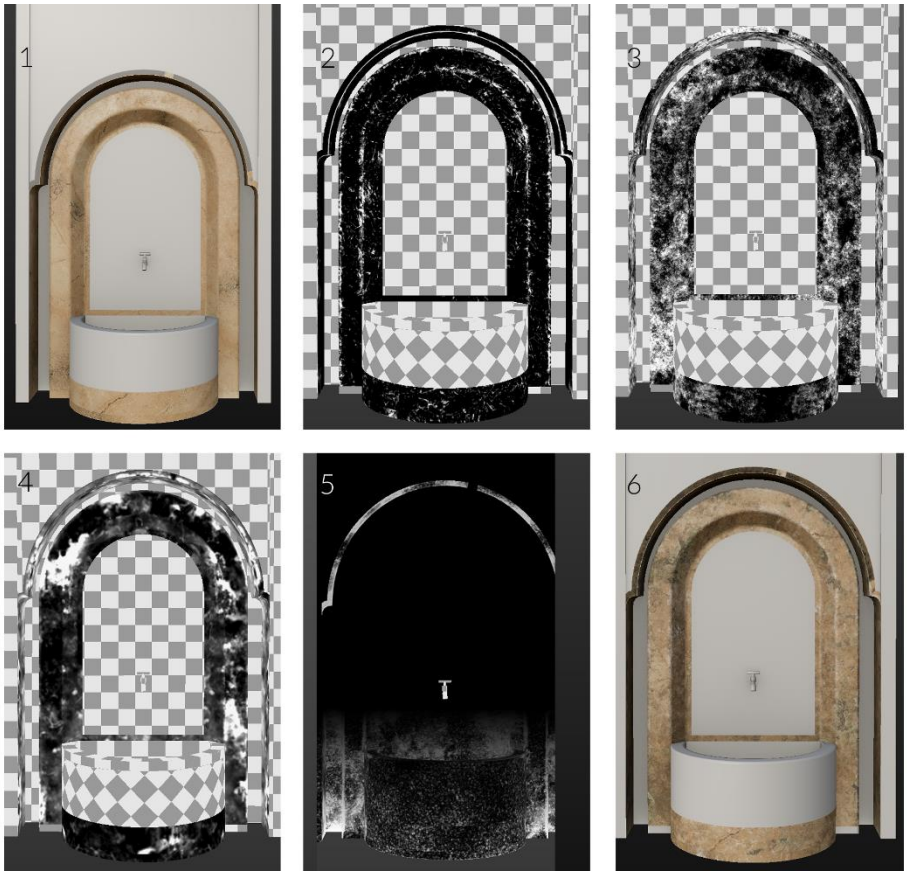

Abbildung 46: Schritt-für-Schritt-Bearbeitung der Marmorumrandung des Brunnens mit Substance Painter.

- **Bild 1:** Ursprungsmaterial – ein glatter, cremefarbener Marmor mit leichten Verfärbungen.
- **Bild 2:** Auf einer neuen Ebene wurde ein „Metal Edge Wear“ Generator hinzugefügt. Dieser konnte über Schieberegler in der Intensität und der Stärke des Grunge-Effekts angepasst werden.
- **Bild 3:** Dann wurde ein mittelbrauner Farb-Layer hinzugefügt und maskiert. Dafür wurde eine „Dirty Dust“ Maske verwendet, um auf der gesamten Fläche unregelmäßige, dunkle Verfärbungen zu erzeugen. Diese Maske konnte in ihrer Intensität, der Position und Schärfe angepasst werden.
- **Bild 4:** Ein zweites Material – „moss splotches on concrete“ – wurde hinzugefügt. Das sollte nur an einzelnen Stellen dargestellt werden. Dazu wurde eine „Sand Dust“-Maske mit leichtem *Blur* benutzt. Um die Flächen aufzulockern, wurde eine „Dust Stained“-Maske addiert.
- **Bild 5:** Dieser Schritt zeigt einen Schmutz-Layer mit Gradienten. Hierfür wurde ein Farb-Layer hinzugefügt und mit der Textur „Dirt 1“ maskiert. Für einen stärkeren Effekt an den seitlichen Kanten wurde anschließend eine „Dirt Ground“-Maske addiert. Zum Schluss wurden beide mit einem Positionsfilter multipliziert, sodass die Effekte nur im unteren Bereich des Brunnens auftreten. Damit der Effekt alle *Color IDs* des Brunnens beeinflusst, wurde er ganz oben im *Layer Stack* platziert und hat keine Zuweisung einer Vektor Farbe erhalten.
- **Bild 6:** Das fertig bearbeitete Marmormaterial.

Die Bearbeitung erfolgte auch in **Quixel Mixer** (Abbildung 47):

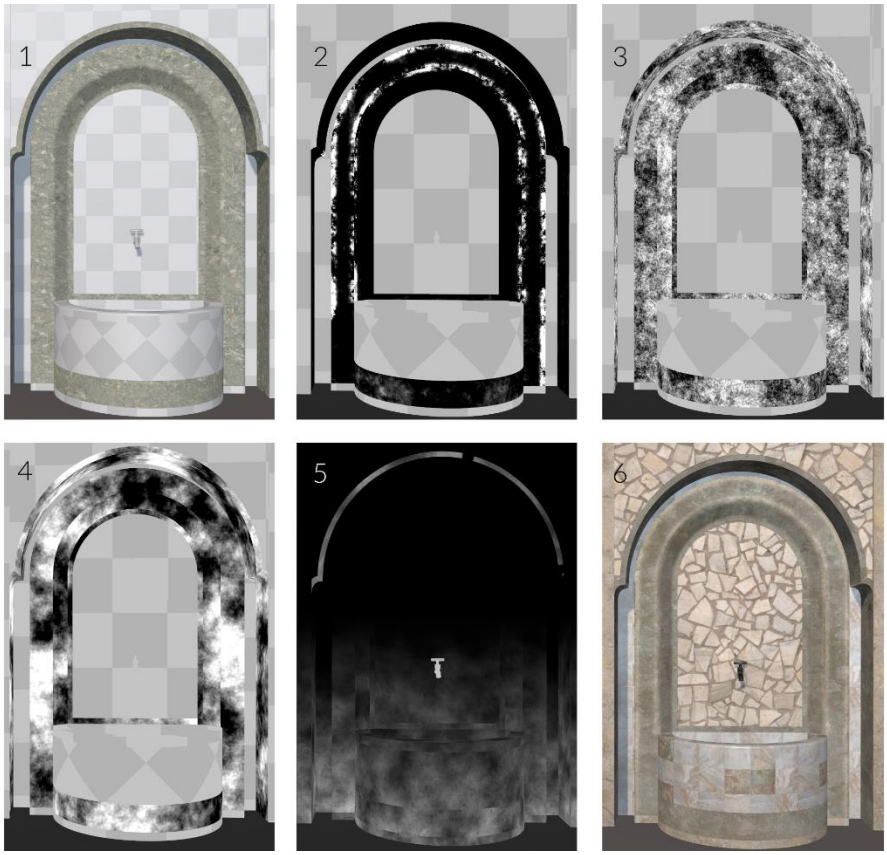

Abbildung 47: Schritt-für-Schritt-Bearbeitung der Marmorumrandung des Brunnens mit Quixel Mixer.

- **Bild 1:** Das Ausgangsmaterial ist ein grau-grünes Marmormaterial mit Farbvariationen.
- **Bild 2**: Auf einer neuen Ebene wurde ein *Curvature*-Filter angepasst und mit einer angepassten *Perlin Noise* multipliziert, um einen durchbrochenen Grunge-Effekt für die Kantenabnutzung zu generieren.
- **Bild 3:** Ein mittelbrauner Farb-Layer wurde mit einer angepassten *Perlin Noise* maskiert, um unregelmäßige Farbabweichungen und schmutzige Stellen zu erzeugen. Der geringe Schärfegrad der Rauschtextur lässt auch graue Flächen zu, welche leicht eingefärbte Farbflächen ergeben.
- **Bild 4:** Die sichtbare Maske wurde dazu benutzt, ein „Mossy Rock" Material auf das bestehende Marmor-Material zu applizieren. Die Albedo-Werte wurden automatisch an die der darunter liegenden Ebene angepasst. Dadurch sieht man den Effekt in einer subtilen Farbabweichung. Die Verringerung des „Wrap to Underlying"-Wertes konnte eine leichte Erhebung des Materials über alle darunterliegenden Ebenen erreichen.
- **Bild 5:** Alle UVs wurden einem neuen Ordner im *Layer Stack* hinzugefügt. Dadurch konnte ein „*Position Gradient*"-Effekt über alle Materialien gelegt werden. Um ein unregelmäßiges Erscheinungsbild zu erzeugen, wurde dieser mit einer Rauschtextur multipliziert. Im Anschluss wurde die Textur „Mud Stain" für eine Wasserspritzer-Struktur hinzugefügt.
- **Bild 6:** Das fertig bearbeitete Marmormaterial.

Zusammenfassung

Die Bearbeitung beider Brunnen war unter Einhaltung aller Vorgaben möglich. Dennoch wird spätestens beim Export deutlich, dass die Nutzung von *ID Masks* bei mehreren Materialien hilfreich ist:

- Aus Substance werden drei Texturen exportiert: *Base Color*, *Channel Packed* und *Normal Map*. Diese können in eine Materialinstanz in Unreal importiert und auf das Mesh gelegt werden.
- Aus Quixel werden insgesamt 12 Texturen exportiert. Für jedes Material werden einzeln *Base Color*, *Channel Packed* und *Normal Map* erstellt. Daher müssen in Unreal vier Materialinstanzen erstellt und einzeln auf die entsprechenden Teile des Meshes projiziert werden.

Der Prozess selbst unterschied sich insbesondere in der Verfügbarkeit von Presets. Für die Bearbeitung in Substance steht eine Vielzahl von Generatoren und Filtern zur Verfügung. Diese können komfortabel über Schieberegler angepasst und justiert werden (Abbildung 48). Dabei übernimmt die Software selbst die Erstellung des Effektes durch Rauschen und Texturen. Im Gegensatz dazu müssen diese Schritte in Quixel manuell vollzogen werden. Im *Mask Stack* werden Texturen und Rauschen verblendet, angepasst und gefiltert, um die gewünschten Maskierungen und Effekte zu erhalten.

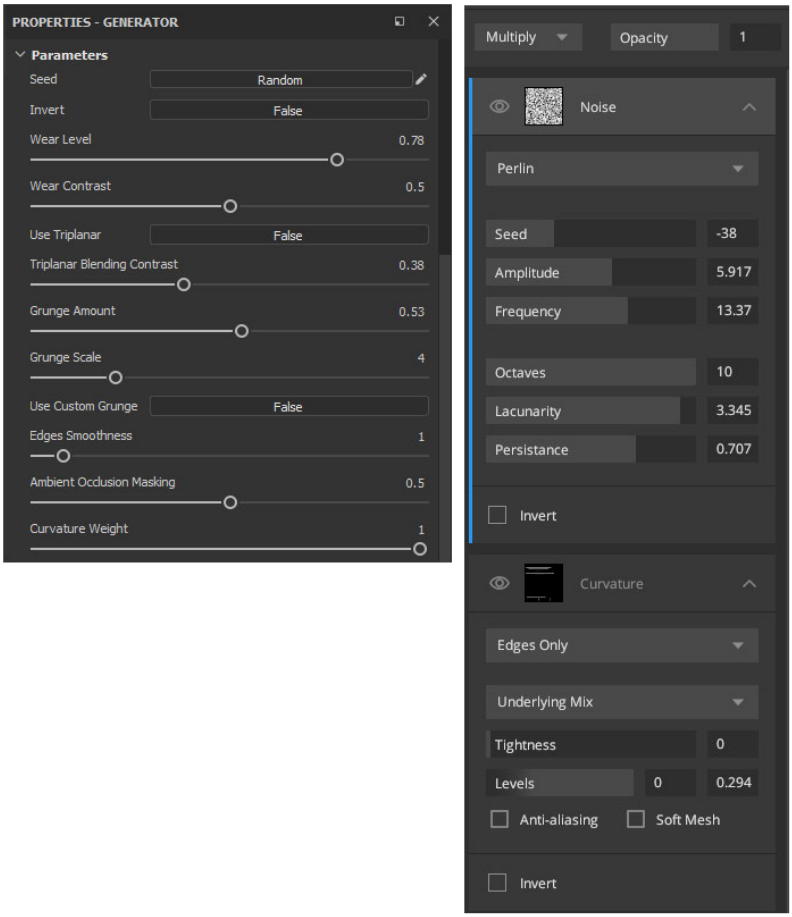

Abbildung 48: Links sieht man die Parameter des Substance "Edge Wear"-Generators. Rechts ist ein in Quixel nachgebauter Filter.

6. Erstellung der finalen Szene

Für die finale Szene sollen alle Assets mit Materialien versehen werden. In diesem Schritt werden diese Bearbeitungskriterien überprüft:

- Flexibilität bei der Materialerstellung
- Zeitaufwand für die vollständige Texturierung
- Anpassungsmöglichkeiten nach dem Import in die Engine

Im Folgenden wird die Herangehensweise für die einzelnen Materialien näher betrachtet.

Mauerstein

Die Wände des mittleren Gebäudes bestehen aus acht einzelnen Meshes in zwei verschiedenen Größen (Abbildung 49). Um eine gleichmäßige Texel-Dichte zu gewährleisten, haben diese unterschiedlich große UVs. Das darauf applizierte Material soll eine mit Weiß überstrichene Mauerstein-Struktur aufweisen. Da sich die Meshes in der Ecke treffen, ist es wichtig, dass beide Seiten sowohl im Muster als auch in der Texel-Dichte zueinander passen. Außerdem sollen kleine Beschädigungen vorhanden sein.

Abbildung 49: Die Wände sind aus vier schmalen Wandelementen (orange) und vier Eckelementen (blau) zusammengesetzt.

- **Substance:** In der 3D-Asset-Library ist ein Material enthalten, das die optischen Anforderungen erfüllt. Im „Painted_Brick_Wall“-Material lassen sich die Farbe und der Grad der Beschädigung einstellen. Der Import erfolgt als SBSAR-Datei direkt aus der Library.
  Da zwei verschieden große UVs verwendet werden, muss auch die Texturgröße angepasst werden. Durch das Duplizieren der „Substance Instance Factory“-Datei erhält man zwei funktionsfähige und anpassbare Instanzen desselben Materials. In den *Substance Graph Instances* kann dadurch eine voneinander unabhängige Texturgröße gewählt werden.
  Die Anpassung der Größe erfolgt nun über die Parameter der Unreal Materialinstanz. Nachdem alle Meshes mit Materialien versehen wurden, kann eine Beschädigungsintensität eingestellt werden. Diese soll einerseits das Muster

unterbrechen, aber andererseits keine deutlichen Wiederholungen erkennen lassen.

Abbildung 50: Mauerstein-Ecke mit Beschädigungen in der finalen Substance-Szene.

- **Quixel:** Für Quixel wird das Material in Mixer angepasst. Dafür wird das erste Mesh geladen und Steingröße, Farbe, Filter und Beschädigungen werden darauf angepasst. Anschließend können die Texturen als *Base Color*, *Channel Packed* und *Normal Map* exportiert werden. Die erstellten Layer werden in einen neuen Ordner kopiert und ausgeblendet, sodass alle Einstellungen für das erste Mesh erhalten bleiben.
  In den „Model Settings“ wird das zweite Mesh geladen. Da die bearbeiteten Material-Layer kopiert wurden, sind alle Größenverhältnisse noch vorhanden – die Texturen können direkt exportiert werden.
  In Unreal werden für die Texturen zwei Instanzen des „Base_Mask“-Master-Materials erstellt und auf die Meshes projiziert.

Abbildung 51: Mauerstein-Ecke mit Beschädigungen in der finalen Quixel-Szene.

Fliesen

Die Fliesen des mittleren Gebäudes sollen aus hellem Marmor mit Metallic-Elementen bestehen.

- **Substance**:
  Ein Marmor-Material mit Metall-Inlets ist in der Bibliothek nicht verfügbar. Daher wird ein prozedurales Marmor-Material modifiziert. Anstelle der SBSAR- wird die SBS-Datei aus der 3D-Asset-Library geladen. Diese erlaubt den Zugriff auf den Substance-Graphen in Designer. Im Node-Editor wird durch eine Maske vor dem Metallic-Output der Bereich markiert, der metallisch dargestellt werden soll (Abbildung 52). Anschließend wird das Material als SBSAR-Datei exportiert.

Abbildung 52: Links sieht man das prozedurale Marmor-Material, rechts das in Substance Designer veränderte Metallic-Material.

- **Quixel:**
  Megascans enthält mehrere Marmor-Materialien mit Metallic-Elementen. Da diese zu glatt für den Einsatzzweck sind, müssen Anpassungen in Mixer gemacht werden. Auf der Default-Plane kann das Material nahtlos geladen werden (Abbildung 53). Dann können Sand-Layer mit hoher *Roughness* hinzugefügt und gefiltert werden.
  Der Export erfolgt auch hier als einzelne Texturen. Dadurch werden keine automatischen Presets mehr geladen, stattdessen wird für den Import eine eigene Materialinstanz benötigt.

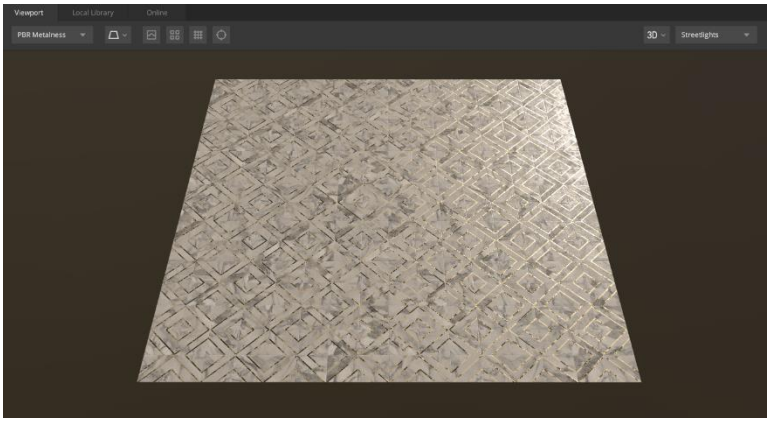

Abbildung 53: Bearbeitung von Library-Materialien in Quixel Mixer.

Foliage

Das Mesh einer Palme soll mit Atlanten aus den Libraries versehen werden. Dies erfolgt in Substance Painter und Quixel Mixer.

Substance: In der 3D-Assets-Library gibt es vier geeignete Atlanten. Da die UV der Palme in sechs Inseln geteilt wurde, werden zwei davon doppelt verwendet. Nach dem Download wird manuell ein *Opacity*-Layer hinzugefügt, um später die Maske exportieren zu können.
Das Anpassen der Atlanten erfolgt im 3D-/2D-View-Mode. Die einzelnen Blätter lassen sich mit der Maus auf der UV-Plane bewegen. Parallel kann das Ergebnis auf dem Mesh gesehen werden (Abbildung 54). Zum Schluss werden überlappende Teile maskiert und die Texturen exportiert.

Abbildung 54: Screenshot des 3D-/2D-View-Modes in Substance Painter.

- **Quixel**:
  Für jede der sechs UV-Inseln wird ein anderes Atlanten-Blatt und ein geeigneter Stiel heruntergeladen. Zur Anpassung wechselt man in den 2D-Modus – das Mesh kann man parallel nicht sehen. Erst nach dem Hinzufügen eines Material-Layers sieht man die UVs im 2D-Modus. Dann können die Atlanten platziert werden (Abbildung 55).

Abbildung 55: Links sieht man den 2D-View-Mode mit zusätzlichem Material-Layer, rechts ohne.

Mixer verfügt über kein Export-Preset für Masken. Dafür müssen alle Masken-Layer manuell in einem gesonderten Export-Channel gespeichert werden. Dieser wird anschließend einer anderen Textur – *Base Color* oder *Channel Packed Mask* – als Alpha-Kanal hinzugefügt.

Vorhänge

Teil des Konzeptes ist es, einen filigranen Stoff mit Spitzen- oder Lochstick-Elementen zu erstellen. Dieses Material wird in Unreal mit dem transluzenten *Material Blend Mode* gerendert.

- **Substance:** Ein geeignetes Material ist in der Library vorhanden und muss in Painter angepasst werden. Dies funktioniert problemlos. Weitere Layer übernehmen die Transparenz des Loch-Musters automatisch, sodass schnell ein Vorhang gestaltet werden kann (Abbildung 56). Nach dem Export als Texturen mit *Channel Packed Mask* kann das Material aus dem „Translucent"-Master-Material erstellt werden.

Abbildung 56: Der Vorhang mit transluzenter Darstellung in Substance Painter.

- **Quixel:** Die Megascans-Library enthält keine transluzenten oder transparenten Materialien abseits von Atlanten. Ein geeignetes Material muss daher selbst in Mixer erstellt werden. Dabei ist problematisch, dass in der Bearbeitungssoftware weder Transparenz noch Transluzenz angezeigt werden. Das Material wird nahezu ausschließlich im *Mask View* erstellt. In der beleuchteten PBR-Ansicht sieht man nur die Struktur des Stoffes (Abbildung 57).

Abbildung 57: Das erstellte Vorhang-Material in Mixer. Links die *Mask*-Layer, rechts das beleuchtete Mesh mit Material.

Erst nach dem Import der Texturen in die transluzente Materialinstanz in Unreal Engine wird das Muster sichtbar (Abbildung 58).

Abbildung 58: Quixel-Vorhang mit Transluzenz in der Engine.

7. Performance

Nach der Fertigstellung der Szenen können die verschiedenen *Profiling Tools* zur Analyse der Performance verwendet werden. Um gleiche Bedingungen in beiden Szenen zu schaffen, wird ein *Camera Actor* an einer festen Position installiert, von der aus eine Totale des mediterranen Innenhofs erfasst werden kann. Ein *Blueprint* wird dafür verwendet, das Level aus Perspektive der Kamera zu starten.

Shader Complexity

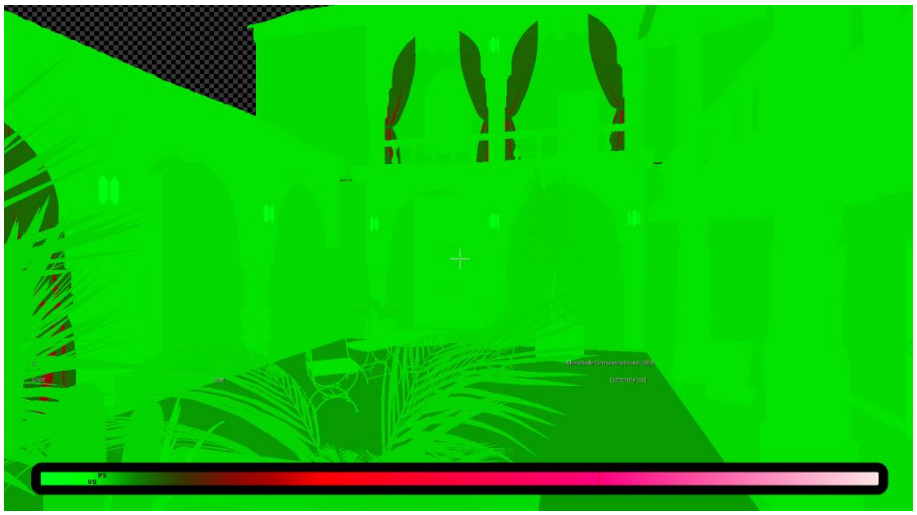

Abbildung 59: *Shader Complexity* der Substance-Szene.

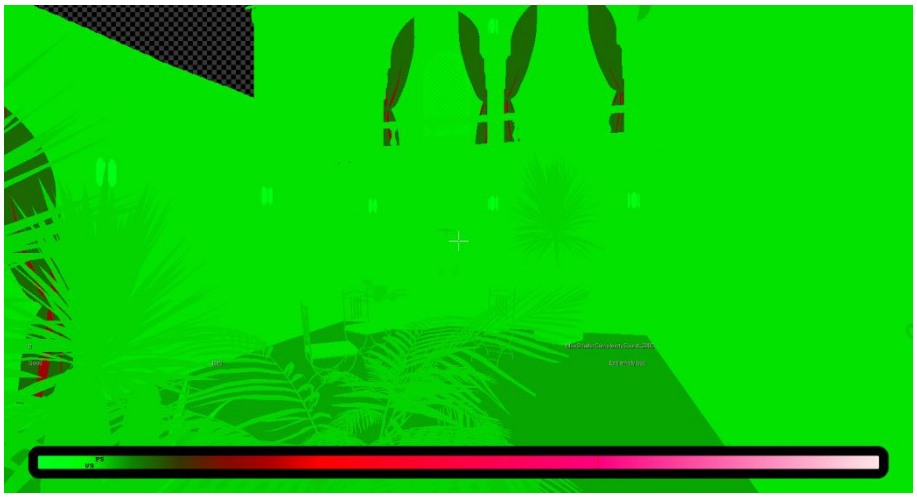

Abbildung 60: *Shader Complexity* der Quixel-Szene.

Im *Shader Complexity View Mode* sind sich die Szenen sehr ähnlich. Die Assets, die in der Substance Szene (Abbildung 59) mit SBSAR-Materialien texturiert wurden, erscheinen leicht dunkler, als die von mir erstellen Materialinstanzen. Abgesehen von Teilen des transluzenten Materials sind alle Materialien grün und damit im guten bis idealen Bereich. Die roten Elemente sind Mesh-Überlappungen, an denen mehrere Schichten transluzenten Materials gerendert werden.

stat Unit

Abbildung 61: stat Unit-Statistik der Substance-Szene.

Abbildung 62: stat Unit-Statistik der Quixel-Szene.

Die für diesen Vergleich wichtigen Werte sind „*Draw*“ und „*GPU*“:

- Die GPU-Werte sind mit 8,66ms bei Quixel und 8,7ms bei Substance sehr ähnlich. Das bedeutet, dass die Szenen eine annäherungsweise gleich lange Renderzeit in der Grafikkarte benötigen.
- Auch die Draw-Werte sind mit 4,92 ms bei Quixel und 4,72 ms bei Substance ähnlich. Der etwas höhere Wert für Quixel kommt durch die größere Menge verwendeter Materialien zustande. Für ein PC-Spiel ist eine Abweichung in dieser Größenordnung vernachlässigbar.

stat RHI

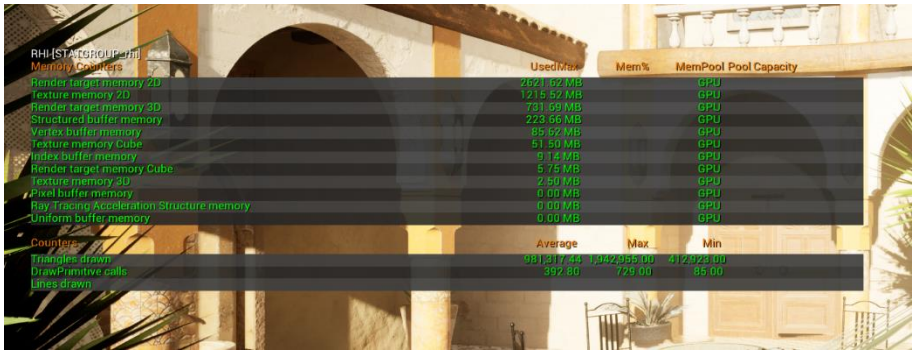

Abbildung 63: stat RHI-Statistik der Substance-Szene.

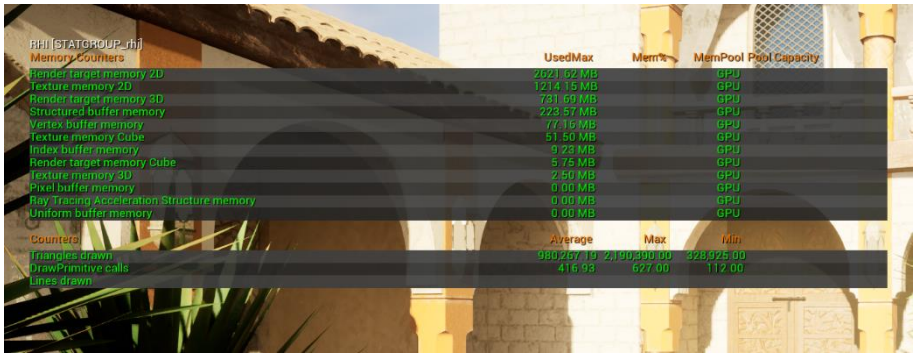

Abbildung 64: stat RHI-Statistik der Quixel-Szene.

In den Abbildungen 63 und 64 sieht man die „stat RHI“-Statistiken der beiden Szenen:

Die Werte für „Texture memory 2D“ geben den benötigten Speicher der verwendeten Texturen an und unterscheiden sich bei den erstellten Szenen kaum.
Bei den *Draw Calls* schneidet Quixel wie bereits bei den *Draws* in der „stat Unit“-Statistik leicht schlechter ab.

5. Auswertung

Das Ziel dieser Arbeit ist ein praktischer Vergleich von Funktionsweise, Nutzerfreundlichkeit und Effizienz der untersuchten Programme Substance und Quixel. Dafür wurde mit beiden Tools ein mediterraner Innenhof vollständig texturiert. Alle für die Szene angeforderten Materialien konnten ohne Einschränkungen sowohl mit Substance als auch mit Quixel erstellt werden. In diesem Kapitel wird evaluiert, wo die Stärken und Schwächen der Applikationen im Entwicklungsprozess lagen.

1. Optischer Eindruck

Bei der Entwicklung eines Spieles ist die Gestaltung einer glaubhaften Umgebung essentiell. Damit diese glaubhaft wirkt, ist ein stimmiges Gesamtbild wichtig. Das erreicht man zum einen über technische Parameter wie eine konstante Texel-Dichte, zum anderen durch stilistisch zueinanderpassende Materialien. Die Anforderung an die Szene „mediterraner Innenhof“ war es, eine Spielumgebung mit mediterranem Flair und leichten Abnutzungspuren zu erschaffen. Dabei entstanden folgende Szenen:

Abbildung 65: "Mediterraner Innenhof" mit Substance-Materialien.

Abbildung 66: "Mediterraner Innenhof" mit Quixel-Materialien.

Beide Szenen sind sehr ähnlich geworden und erfüllen die Kriterien. Die Texel-Dichte konnte über alle Materialien hinweg auf konstanten 512 px/m eingehalten werden. Stilistisch geeignete Materialien aus den Libraries konnten in den zugehörigen Applikationen angepasst werden, um ein stimmiges Bild zu erzielen. Auch der erzielte Detailgrad einzelner Materialien ist vergleichbar, wenngleich er in Quixel Mixer mit mehr Aufwand verbunden war.

Die Erstellung dieser konkreten Szene erfolgte nach dem Konzept eines eigenen Projektes, für das die Funktionalität der Softwares und Libraries verglichen werden sollte. Von meinem

Standpunkt als 3D-Artist dieses Projektes aus, konnten beide Szenen mit einem vergleichbaren optischen Gesamteindruck punkten. Die dafür gesetzten Anforderungen konnten erfüllt und eine überzeugende mediterrane Atmosphäre geschaffen werden.

2. Libraries

Die Bibliotheken sind beide sehr umfangreich. Das Potenzial der Substance 3D Asset Library lässt sich jedoch nur sehr gezielt und wohlüberlegt nutzen. Für die erstellte Szene wurden insgesamt 63 Elemente heruntergeladen. Davon verwendet wurde gerade einmal die Hälfte, da sich erst bei der Texturierung gezeigt hat, ob das Asset tatsächlich passt. Die Megascans Library hingegen bietet den Vorteil, dass alle Assets verwendet und auch zum Ausprobieren heruntergeladen werden können.
Aufgrund der unterschiedlichen Materialien in den Bibliotheken, unterscheidet sich das visuelle Ergebnis in manchen Bereichen. Ein Beispiel hierfür sind die unterschiedlichen Fliesen oder Mosaiken der beiden Szenen (Abbildung 67 und 68). Nachdem die Anforderungen kein identisches Ergebnis verlangten, sondern die Texturierung mit verfügbaren Library-Assets, ist dies für keins der beiden Software-Pakete ein Minuspunkt.

Abbildung 67: Detailaufnahme des Brunnens der Substance-Szene.

Abbildung 68: Detailaufnahme des Brunnens der Quixel-Szene.

In den Abbildungen 69 und 70 sieht man die Ergebnisse des Eingangsbereichs der beiden Szenen. Die Meshes der Türen sind bis auf die Kassettenelemente flach und sollten mit Schnitzereien verziert werden. Dafür wurden passende Elemente der Libraries an den geeigneten Stellen maskiert. Aufgrund der Unterschiede in den Bibliotheken fallen die geschnitzten Muster unterschiedlich aus, erfüllen aber beide die Erwartungen:

Abbildung 69: Detailaufnahme des Eingangsbereichs der Substance-Szene.

Abbildung 70: Detailaufnahme des Eingangsbereichs der Quixel-Szene.

3. Funktionen

Die Adobe Substance 3D Suite und Quixel unterscheiden sich auf den ersten Blick direkt in der Anzahl der enthaltenen Applikationen. Substance besteht aus drei Programmen und der Bibliothek, Quixel hingegen aus der Software Mixer und der Megascans Library. Von den generellen Funktionen ist Mixer vergleichbar mit Substance Painter, für Designer und Sampler gibt es kein Quixel-Äquivalent.
Quixels Hauptaugenmerk liegt auf der Erstellung UV-gebundener Texturen, es kann jedoch aufgrund der fehlenden *Baking*-Funktion nicht alle Anforderungen erfüllen. Substance enthält diese Funktion, sowie zusätzliche Features zum Erstellen eigener Materialien und der

Bearbeitung nahtloser Texturen. Dadurch kann das gewünschte Material optisch exakt gestaltet werden. Bei der Nutzung von Quixel ist man dagegen überwiegend auf die Inhalte der Megascans Library angewiesen.
Beispielsweise erwies sich die Erstellung der Vorhänge in Quixel als suboptimal. Während in Substance sogar Transluzenz dargestellt wurde, war in Quixel viel Phantasie nötig um sich das Ergebnis nur anhand einer Maske vorstellen zu können. Daher waren mehrere Reimporte in die Engine nötig, um die Texturen anzupassen.

4. Workflow und Integration in Unreal Engine 5

Alle Materialien in Unreal Engine benötigen Master-Materialien. Für Quixel sind Material-Presets für alle Library Assets verfügbar, sodass diese per Drag & Drop direkt genutzt werden können. Das ist insbesondere bei den Pflanzen- und Environment-3D-Assets praktisch, da sie für ein schnelles *Blocking* der Szene oder das Gestalten des Hintergrunds genutzt werden können. Nachteil bei diesem System ist, dass sich die Assets und Materialien aufgrund fehlender Einstellungsmöglichkeiten ohne große Anpassungen in die Szene integrieren müssen.
Dieses Problem hat Substance durch die direkte Implementierung der Designer-Material-Graphen in Unreal gelöst. Dadurch bleiben die importierten SBSAR-Dateien flexibel und bieten auch in der Engine viele Anpassungsmöglichkeiten. Zudem ist es möglich, eigene SBSAR-Dateien mit ihren Parametern per Drag & Drop einzufügen. Die dafür vorhandenen Master-Material-Presets sind nicht so umfangreich. Materialien wie beispielsweise *Decals* können erst nach Erstellung eines eigenen Master-Materials verwendet werden.
Die Anpassung von UV-gebundenen Assets erfolgt in Mixer und Painter. Während Export aus den Bearbeitungssoftwares und Import in die Engine gleich funktionieren, unterscheidet sich der Workflow innerhalb der Tools. Substance Painter ist ein sehr umfangreiches und dadurch komplexes Programm, das – wenn man es beherrscht – nahezu keine Grenzen in der Gestaltung von Assets kennt. Die Software bietet viele Spezialfunktionen, Generatoren und Filter, die im Vergleichsprogramm Mixer aus Basisfunktionen wie Rauschtexturen manuell erstellt werden müssen.

5. Einarbeitungszeit und Arbeitsaufwand

Die hohe Komplexität von Substance spiegelt sich in umfangreichen Bedienelementen im User-Interface wider. Um die Möglichkeiten der Software voll auszuschöpfen, bedarf es längerer Einarbeitungszeit. Hat man erst einen Überblick über die Funktionen, können die spezialisierten Filter und Generatoren die Materialgestaltung beschleunigen.
Das User-Interface und die Funktionen von Quixel Mixer sind in wenigen Minuten vollständig erklärt. Das Prinzip der wenigen Bedienelemente wird schnell klar, sodass sie in der Anwendung intuitiv erscheinen und schnell Ergebnisse erzielt werden können. Bei längerer Nutzung zeigt sich jedoch, dass die Erstellung von Filtern aus einzelnen Rausch-Layern langwierig ist, da jeder Effekt von Grund auf neu aufgebaut werden muss.
Der Arbeitsaufwand für die Integration der Materialien in die Unreal Engine hängt davon ab, wie viele Texturen exportiert werden. Am Beispiel des Brunnens war die benötigte Zeit zum

Importieren der Mixer-Materialien viermal so hoch, wie bei den Substance-Texturen. Hätte man vorab auch eine *ID Mask* für Mixer erstellt, wäre der Importaufwand geringer, aber zusätzliche Zeit für die Erstellung der Maske nötig gewesen.
Umgekehrt verhielt es sich beim Erstellen des Foliage-Materials der Palme. In Quixel wurde ein zusätzliches Material benötigt, um die UVs sichtbar zu machen. Die Platzierung der Atlanten war nur mit Schiebereglern möglich und jeder Layer musste einzeln dem *Mask*-Export hinzugefügt werden. Trotzdem war für die ganzen Arbeitsschritte weniger Zeit nötig, als bei Substance. Grund dafür war die zusätzlich benötigte Einarbeitungszeit in *Opacity*-Export-Einstellungen und die richtige Maskierung überlappender Meshes in Painter.

Auch wenn ich vor dieser Thesis bereits mit Substance gearbeitet hatte und mir Quixel vollkommen neu war, hat die Texturierung beider Szenen mit jeweils ca. 50 Stunden ungefähr gleich viel Zeit beansprucht. Zeitliche Vor- und Nachteile wurden in dieser Tabelle zusammengefasst:

	Substance	Quixel
+	*Color ID Maps* ermöglichten Textur-Einsparungen, daher mussten weniger Materialinstanzen erstellt werden. Durch die Praktikabilität der SBSAR-Dateien wurden zusätzlich 11 Materialien direkt aus der Library verwendet, sodass nur 34 weitere Instanzen erstellt werden mussten.	Eine Einarbeitung war mithilfe eines kurzen Tutorial-Videos sehr einsteigerfreundlich. Die Bedienung in Mixer folgt immer demselben Muster, wodurch sie sehr intuitiv und schnell erfolgen kann.
-	Die Einarbeitungszeit ist sehr hoch, insbesondere wenn man alle Tools beherrschen will. Zudem mussten neue Arbeitsschritte und einzelne Funktionen nachgeschlagen werden.	Für Quixel mussten 55 Materialinstanzen erstellt werden, aufgrund geringer Anpassungsmöglichkeiten wurden nur 3 weitere Materialien direkt aus der Library verwendet. Dadurch war der Export- und Importaufwand hoch.

6. Performance

In der Performance-Analyse hat sich gezeigt, dass die beiden Szenen keine großen Unterschiede in Renderzeit und Speicher haben. Zwar sind die *Draw Calls* der Quixel-Szene leicht erhöht, für ein Computerspiel ist diese Abweichung jedoch vernachlässigbar. Dennoch sieht man daran, dass mehr Einzelmaterialien für die Texturierung benötigt wurden.
Für die Entwicklung auf CPU- und GPU-schwächeren Plattformen wie etwa Android könnte sich dieser Unterschied bereits bemerkbar machen, sodass an dieser Stelle optimiert werden müsste.

7. Fazit

Abschließend lässt sich zusammenfassen, dass eine Texturierung beider Szenen unter Berücksichtigung aller Kriterien erfolgreich war. Aus den Erkenntnissen der vorangegangenen Auswertung wurde die folgende Tabelle erstellt:

	Substance 3D Suite	Quixel
Optischer Eindruck der Szene	++	++
Library		
Inhalt	++	++
Verfügbarkeit der Assets	+	++
Funktionen der Applikationen		
Erstellung UV-gebundener Materialien	++	+
Erstellung eigener Materialien	++	-
Bearbeitung von nahtlosen Materialien	++	0
Integration in Unreal Engine 5		
Presets	+	++
Anpassung importierter Library-Materialien	++	0
Anpassung UV-gebundener Materialien	+	+
Einarbeitungszeit	-	++
Effektive Arbeitszeit	+	0
Performance	++	+

Was die Softwares und die Flexibilität der Materialien in Unreal Engine 5 betrifft, ist Substance nicht umsonst Industriestandard. Mit dem Abonnement erhält man umfangreiche Profi-Tools zur Materialerstellung, welche die Arbeit mit etwas Übung sehr erleichtern. Ich persönlich finde es bedauerlich, dass es den Library-Zugriff ausschließlich im Abo gibt und Materialien nicht auch einzeln oder in Sets erworben werden können. Dadurch könnte die Materialerstellung ein kreativer Prozess des Ausprobierens sein, ohne durch Download-Beschränkungen ausgebremst zu werden. Zudem hat die Library bei verschiedenen Materialtypen, wie beispielsweise Atlanten, noch Potenzial nach oben.
Einsteigerfreundlich zeigt sich die Substance 3D Suite nicht. Durch die enorme Komplexität der Tools und der Bedienelemente ist die Einarbeitungszeit relativ lang. Beherrscht man die benötigten Funktionen, können Materialien in kürzester Zeit bearbeitet werden. Insbesondere die Integration der SBSAR-Parameter in Unreal Engine erspart viel Arbeit.

Mit Quixel Mixer erhält man ein solides Tool, das – wenn auch über Umwege – alle Herausforderungen meistern konnte. Die angeschlossene Library ist umfangreich und voll nutzbar. Was fehlt, sind emittierende und transluzente Materialien – es bleibt zu hoffen, dass an dieser Stelle noch nachgerüstet wird. Unter *MS_Presets* findet sich bereits ein (noch) leerer „*Glass*"-Ordner.

Auch Mixer hat Verbesserungspotenzial. Insbesondere die fehlende *Color ID*- und *Normal Map-Baking*-Funktion ist für viele Projekte substanziell. Obwohl für dieses Projekt keine externen *Baking*-Tools verwendet wurden, konnte Quixel überraschenderweise gute Ergebnisse in der Performance erzielen.
Durch die Übersichtlichkeit der Funktionen und Einfachheit des User-Interfaces ergibt sich eine sehr intuitive, gleichförmige Arbeitsweise. Deshalb eignet sich Mixer sehr gut für Einsteiger und für neue Herausforderungen.
Quixel kann somit durchaus als kostenlose Alternative zu Substance in Betracht gezogen werden. Damit hat Epic Games einen Workflow zur Erstellung qualitativer Materialien auch für alle Indie-Produktionen ermöglicht, die ihre Ideen ohne großes Budget umsetzen müssen.

Für das erstellte Projekt und damit für mich persönlich hat sich diese Software-Analyse gelohnt. Sie hat gezeigt, dass mit beiden Programmen ein sehr gutes Ergebnis erzielt werden kann – aber auch, dass Quixel Mixer wichtige Funktionen bislang fehlen.

Besonders gut eignet sich Mixer für:

- Prototyping
- Einsteiger-Projekte
- Qualitative Low-Budget Produktionen
- Als Zusatz-Software zu Substance für schnelle und unkomplizierte, manuelle Material-Anpassungen

Sobald die Erstellung eigener Materialien erforderlich wird, ist die Substance 3D Suite bislang unverzichtbar. Auch bei einer großen Anzahl von Assets, die ähnliche Anpassungen benötigen, wie beispielsweise „Edge Wear"-Effekte, beschleunigt Substance die Produktion enorm, sodass über ein Abonnement nachgedacht werden sollte.

Diese Arbeit hat durch ihr konkretes Projekt lediglich eine Analyse von themenspezifischen Materialien erreicht. Daher können Abweichungen bei anderen Projekt-Anforderungen und Environment-Settings entstehen.
Die Performanceanalysen der Master-Materialien und der Überblick über die Grundfunktionen der Softwares sind allgemeingültig und auf andere Produktionen übertragbar. Deshalb können auf dieser Arbeit aufbauend weitere Library-Vergleiche für andere Projekte erfolgen.

Bei der Entwicklung des Spiels, aus dem das Konzept des „mediterranen Innenhofs" stammt, werde ich auf beide Tools zurückgreifen. Das kombiniert die Flexibilität der SBSAR-Dateien in der Unreal Engine, den prozeduralen Workflow des Designers und die *Baking*-Funktionen mit der intuitiven Bedienung von Mixer und dem unbegrenzten Library-Zugang. Dadurch können die Stärken aller Tools gemeinsam zu einem dynamischen Material-Workflow zusammengefügt und die Schwächen umgangen werden.
Bei zukünftigen Herausforderungen wird sich zeigen, ob meine aus diesem Vergleich entstandenen Präferenzen Bestand haben oder ob sich neue ergeben. Zum einen hat Quixel Mi-

xer immer noch ein „Beta“ in der Versionsnummer, weshalb mit zusätzlichen Funktionen gerechnet werden darf. Zum anderen sind die Grenzen von Substance mit dieser Analyse noch lange nicht erreicht und ich entdecke immer noch neue Features in diesem Software-Paket.

Literaturverzeichnis

Alle Links wurden zuletzt am 07.08.2022 abgerufen.

Adobe (2022). *Substance 3D Assets*. Von Adobe Substance 3D: https://substance3d.adobe.com/assets abgerufen

Adobe (2022). *Substance 3D Designer*. Von Adobe: https://www.adobe.com/de/products/substance3d-designer.html abgerufen

Adobe (2022). *Substance 3D Painter*. Von Adobe: https://www.adobe.com/de/products/substance3d-painter.html abgerufen

Adobe (2022). *Substance 3D Sampler*. Von Adobe: https://www.adobe.com/de/products/substance3d-sampler.html abgerufen

Adobe (2022). *Substance 3D-Abos vergleichen*. Von Adobe: https://www.adobe.com/de/products/substance3d/compare-plans.html abgerufen

Adobe (2022). *Substance in Unreal Engine*. Von Adobe Substance 3D: https://substance3d.adobe.com/plugins/substance-in-unreal-engine abgerufen

Arm, Ltd. (1995-2022). *Comparing unlit shaders to lit shaders*. Von armDeveloper: https://developer.arm.com/documentation/102471/0100/Comparing-unlit-shaders-to-lit-shaders abgerufen

Arm, Ltd. (1995-2022). *Profile and visualize shader complexity*. Von armDeveloper: https://developer.arm.com/documentation/102676/0100/Profile-and-visualize-shader-complexity abgerufen

Bergsman Lind, T. (2019). *Quixel joins forces with Epic Games*. Von Quixel: https://quixel.com/blog/2019/11/12/quixel-joins-forces-with-epic-games abgerufen

Clement, J. (2022). *Number of Games released on Steam*. Von Statista: https://www.statista.com/statistics/552623/number-games-released-steam/ abgerufen

Epic Games Inc. (2004-2022). *Guidelines for Optimizing Rendering for Real-Time*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/guidelines-for-optimizing-rendering-for-real-time-in-unreal-engine/ abgerufen

Epic Games Inc. (2004-2022). *Texture Streaming Overview*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/texture-streaming-overview-for-unreal-engine/ abgerufen

Epic Games, Inc. (2004-2022). *Frequently Asked Questions (FAQs)*. Von Unreal Engine: https://www.unrealengine.com/en-US/faq abgerufen

Epic Games, Inc. (2004-2022). *Instanced materials in Unreal Engine*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/instanced-materials-in-unreal-engine/ abgerufen

Epic Games, Inc. (2004-2022). *Material Blend Modes in Unreal Engine*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/material-blend-modes-in-unreal-engine/ abgerufen

Epic Games, Inc. (2004-2022). *Material Functions*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/material-functions-in-unreal-engine/ abgerufen

Epic Games, Inc. (2004-2022). *Nanite Virtualized Geometry*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/nanite-virtualized-geometry-in-unreal-engine/ abgerufen

Epic Games, Inc. (2004-2022). *Placing Material Expressions and Functions*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/placing-material-expressions-and-functions-in-unreal-engine/ abgerufen

Epic Games, Inc. (2004-2022). *Shading Models in Unreal Engine*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/shading-models-in-unreal-engine/ abgerufen

Epic Games, Inc. (2004-2022). *Stat Commands*. Von Unreal Engine Documentation: https://docs.unrealengine.com/4.26/en-US/TestingAndOptimization/PerformanceAndProfiling/StatCommands/ abgerufen

Epic Games, Inc. (2004-2022). *Using Transparency in Unreal Engine*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/using-transparency-in-unreal-engine-materials/ abgerufen

Epic Games, Inc. (2022). *Viewport Modes in Unreal Engine*. Von Unreal Engine Documentation: https://docs.unrealengine.com/5.0/en-US/viewport-modes-in-unreal-engine/ abgerufen

Iezzi, L. (2016). *Texel Density: All you need to know.* Von https://leonano.gumroad.com/l/texeldensity abgerufen

McDermott, W. (2018). *The PBR Guide*. Von Adobe Substance 3D: https://substance3d.adobe.com/tutorials/courses/the-pbr-guide-part-1 abgerufen

McKenney, E. (2019). *Creating Stylized Art in a PBR World*. Von medium: https://medium.com/velan-studios/tip-of-the-brush-creating-stylized-art-in-a-pbr-world-b803b91c082f abgerufen

Povey, D. (2022). *What is a Seamless Texture (Tileable Texture)?* Von PlusSpec: https://plusspec.com/seamless-texture-tileable-texture/ abgerufen

Quixel (2016). *Megascan's an indispensable resource for the Jungle Book*. Von Quixel: https://quixel.com/blog/2016/8/23/megascans-an-indispensable-resource-for-the-jungle-book abgerufen

Quixel (2022). *Bridge*. Von Quixel: https://quixel.com/bridge abgerufen

Quixel (2022). *Megascans*. Von Quixel: https://quixel.com/megascans/home/ abgerufen

Russel, J. (2022). *Basic Theory of Physically-Based Rendering*. Von Marmoset: https://marmoset.co/posts/basic-theory-of-physically-based-rendering/ abgerufen

Sawall, A. (23. Januar 2019). *Adobe kauft 3D Software Substance*. Von Golem: https://www.golem.de/news/allegorithmic-adobe-kauft-3d-software-substance-1901-138936.html#cmts abgerufen
Steinlechner, P. (02. März 2015). *Unreal Engine mit Quellcode jetzt kostenlos*. Von Golem: https://www.golem.de/news/epic-games-unreal-engine-mit-quellcode-jetzt-kostenlos-1503-112700.html abgerufen
Ta'eed, C. (24. Januar 2008). *How to Turn a Texture Into a Seamlessly Tiled Background*. Von design.tutsplus.com: https://design.tutsplus.com/articles/how-to-turn-a-texture-into-a-seamlessly-tiled-background--psd-1565 abgerufen
Umar-Khitab, E. (15. August 2019). *The Lion King returns with the power of Megascans*. Von Quixel: https://quixel.com/blog/2019/8/15/the-lion-king-returns-with-the-power-of-megascans abgerufen
VisCircle GmbH. (2021). *Einsteigerguide: Die Grundlagen des Texture Mappings*. Von Viscircle: https://viscircle.de/einsteigerguide-die-grundlagen-des-texture-mappings/ abgerufen
World of Level Design LLC. (2016). *UE4: How to Insert, Apply and Use Decals Inside Your Level*. Von World of Level Design: https://www.worldofleveldesign.com/categories/ue4/ue4-decals02-insert-use-decals-inside-level.php abgerufen

Abbildungsverzeichnis

Sofern nicht anders angegeben, stammen alle Abbildungen von der Autorin.
Alle Links wurden zuletzt am 07.08.2022 aufgerufen.

Electronic Arts (2022). *Die Hände nehmen einen Großteil des Bildschirms ein und benötigen daher eine hohe Texel-Dichte.* Respawn Entertainment, Electronic Arts. URL: https://www.ea.com/de-de/games/apex-legends/media#screenshots
Epic Games, Inc. (2004-2022). *Alphawerte von 0 bis 1 mit Visualisierung des entsprechend transparenten, transluzenten und opaken Effekts*. URL: https://docs.unrealengine.com/5.0/Images/designing-visuals-rendering-and-graphics/materials/tutorials/transparency/Trans_Opacity_Settings.webp
Sleegers, E. (2022). *Decal-Materialien in Form von Schmierern und Rissen an der Hausfassade sowie als Reifenspur auf dem Boden*. URL: https://cdnb.artstation.com/p/marketplace/presentation_assets/001/722/181/large/file.jpg
Square Enix, Co., Ltd. (2021). *Screenshot aus Life is Strange: True Color, entwickelt mit Unreal Engine 4*. URL: https://fyre.cdn.sewest.net/siren/60464abe28d636001263c667/12_alex_record_store_1080-6k7-u7rAyQ.jpg?width=2500&quality=80
Epic Games, Inc., Lucasfilm Ldt., (2022). *Screenshot aus Fortnite*. URL: https://www.epicgames.com/fortnite/de/news/fortnite-battle-royale-v21-10-update-darth-vaders-lightsaber-super-level-styles-looking-for-party-and-more
Epic Games, Inc., (2004-2022). *Blume mit den Shader-Modellen „Default Lit" (links) und „Two Sided Foliage" (rechts)*. URL: https://docs.unrealengine.com/5.0/en-US/shading-models-in-unreal-engine/
Epic Games, Inc., (2004-2022). *Shader Complexity View Mode einer Szene. Verschiedene Materialien werden dort farbcodiert dargestellt*. URL: https://docs.unrealengine.com/5.0/Images/building-virtual-worlds/level-editor/editor-viewports/viewport-modes/11-viewmode-shader-perf.png
Epic Games, Inc., (2004-2022). *Farbwerte des Shader Complexity View Modes und ihre Bedeutung für die Performanceintensität einzelner Materialien*. URL: https://docs.unrealengine.com/5.0/en-US/viewport-modes-in-unreal-engine/
Square Enix, Co., Ltd. (2020). *Screenshot aus Final Fantasy VII*. URL: https://substance3d.adobe.com/magazine/wp-content/uploads/2020/07/image3-1.png?x74734
DNEG, Legendary Entertainment, Universal Studios (2018). *Hero-Asset aus Pacific Rim*. URL: https://store.substance3d.com/sites/default/files/Marketing9%20%281%29.jpg
The Coalition (2019). *Screenshot aus Substance Designer. Zu sehen ist ein mit Nodes erstelltes Obsidian-Material für Gear 5*. URL: https://substance3d.adobe.com/magazine/wp-content/uploads/2019/12/obsidian_designer.jpg?x74734
The Coalition (2019). *Texturierung einer Rüstung für Gear 5 in Substance Painter*. URL: https://substance3d.adobe.com/magazine/wp-content/uploads/2019/12/kait_painter.jpg?x74734
Square Enix, Co., Ltd. (2021). *Screenshot aus „Shadow of the Tomb Raider"*. URL: https://hydra.cdn.sqexeu.com/assets/SHADOW-OF-TOMB-RAIDER_ENTITIES/0de6b65f9710dbc8c913f133b71c5e36.jpg
The Walt Disney Company (2019). *Ausschnitt aus „Der König der Löwen"*. URL: https://blog.megascans.se/wp-content/uploads/2019/08/TLK-ONLINE-USE_017_CW_0890_comp_v0699_REC709.1079.jpg

Software

Unreal Engine 5.0.2 bis 5.0.3
© 1998-2022 Epic Games, Inc.
https://www.unrealengine.com/en-US

Quixel Mixer 2021.1.3 - Beta
© 2020-2021, Quixel
https://quixel.com/mixer

Adobe Substance 3D Suite
© 2010-2022, Adobe
https://www.adobe.com/de/products/substance3d/3d-augmented-reality.html

Blender 3.2.0
GNU General Public License v3.0
https://www.blender.org

Asset-Verzeichnis

Folgende 3D-Meshes aus dem „Middle Eastern Courtyard“-Set von Joakim Stigsson wurden benutzt.
URL: https://www.unrealengine.com/marketplace/en-US/product/middle-eastern-courtyard

SM_Arch_01.fbx
SM_Floor_02_A.fbx
SM_Floor_Flowerpot_01_A.fbx
SM_Fountain_01.fbx
SM_Pillar_01_A.fbx
SM_RoofTile_01_A.fbx
SM_Curtain_01_A.fbx
SM_Door_01_Left.fbx
SM_Door_01_Right.fbx
SM_Door_01_Small.fbx
SM_Pot_01_B.fbx
SM_WallLamp_01.fbx
MI_Table_01.fbx
SM_Chair_A.fbx

Folgende 3D-Meshes aus der Quixel Megascans Library wurden benutzt:

S_Spineless_Yucca_wfypfebha_Var9_lod4.fbx
URL: https://quixel.com/megascans/home?search=spineless&search=yucca&assetId=wfypfebha
S_Tropical_Palm_rkqjj_Var1_lod5.fbx
URL: https://quixel.com/megascans/home?search=palm&assetId=rkqjj
S_Aloe_Vera_ve4tbbrha_Var12_lod2.fbx
URL: https://quixel.com/megascans/home?search=aloe&search=vera&assetId=ve4tbbrha

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Bachelor-Thesis mit dem Titel:

Quantitative und qualitative technische Analyse von Tools zur Materialerstellung für die Unreal Engine 5

selbständig und nur mit den angegebenen Hilfsmitteln verfasst habe.
Alle Passagen, die ich wörtlich aus der Literatur oder aus anderen Quellen wie z. B. Internetseiten übernommen habe, habe ich deutlich als Zitat mit Angabe der Quelle kenntlich gemacht.

.....
Datum, Unterschrift