Prototypendokumentation

Inhalt

[Einleitung 3](#_Toc66644551)

[Programmstruktur 3](#_Toc66644552)

[Milestones and Deadlines 4](#_Toc66644553)

[Varianten 4](#_Toc66644554)

[Initialer Stand der Arbeit 4](#_Toc66644555)

[Research & Related Work 4](#_Toc66644556)

[Streaming 4](#_Toc66644557)

[Visualisierungen 4](#_Toc66644558)

[Woche 1 (15.11 – 22.11) 5](#_Toc66644559)

[Research & Related Work 5](#_Toc66644560)

[Streaming 5](#_Toc66644561)

[Visualisierungen 5](#_Toc66644562)

[Woche 2 (23.11 – 29.11) 5](#_Toc66644563)

[Research & Related Work 5](#_Toc66644564)

[Streaming 5](#_Toc66644565)

[Visualisierungen 5](#_Toc66644566)

[Woche 3 (30.11 – 06.12) 6](#_Toc66644567)

[Research & Related Work 6](#_Toc66644568)

[Streaming 6](#_Toc66644569)

[Visualisierungen 6](#_Toc66644570)

[Woche 4 (07.12 – 13.12) 6](#_Toc66644571)

[Research & Related Work 6](#_Toc66644572)

[Streaming 6](#_Toc66644573)

[Visualisierungen 6](#_Toc66644574)

[Woche 5 (14.12 – 20.12) 6](#_Toc66644575)

[Research & Related Work 6](#_Toc66644576)

[Streaming 7](#_Toc66644577)

[Visualisierungen 7](#_Toc66644578)

[Woche 6 (21.12 – 23.12) 7](#_Toc66644579)

[Research & Related Work 7](#_Toc66644580)

[Streaming 7](#_Toc66644581)

[Visualisierungen 7](#_Toc66644582)

[Woche 7,8,9 (28.12 - 10.01) 7](#_Toc66644583)

[Research & Related Work 7](#_Toc66644584)

[Streaming 7](#_Toc66644585)

[Visualisierung 1 (Mesh Generation) 8](#_Toc66644586)

[Visualisierung 3 (Displacement Shader) 8](#_Toc66644587)

[Woche 10 (11.01 - 17.01) 8](#_Toc66644588)

[Research & Related Work 8](#_Toc66644589)

[Streaming 8](#_Toc66644590)

[Visualisierung 3 (Displacement Shader) 8](#_Toc66644591)

[Visualisierung 4 (Volumetrischer Shader) 9](#_Toc66644592)

[Woche 11 (18.01 - 24.01) 9](#_Toc66644593)

[Research & Related Work 9](#_Toc66644594)

[Streaming 9](#_Toc66644595)

[Visualisierung 3 (Displacement Shader) 9](#_Toc66644596)

[Woche 12 (25.02 - 31.02) 9](#_Toc66644597)

[Research & Related Work 9](#_Toc66644598)

[Streaming 9](#_Toc66644599)

[Visualisierung 2 (PointCloud) 10](#_Toc66644600)

[Visualisierung 3 (Displacement Shader) 10](#_Toc66644601)

[Visualisierung 4 (Volumetric Shader) 10](#_Toc66644602)

[Woche 13 (01.02 – 07.02) 11](#_Toc66644603)

[Research & Related Work 11](#_Toc66644604)

[Streaming 11](#_Toc66644605)

[Visualisierung 1 11](#_Toc66644606)

[Visualisierung 2 (PointCloud) 11](#_Toc66644607)

[Visualisierung 3 11](#_Toc66644608)

[Woche 14,15,16,17,18 (08.02 – 15.03) 11](#_Toc66644609)

[Research & Related Work 11](#_Toc66644610)

[Streaming 11](#_Toc66644611)

[Visualisierungen 11](#_Toc66644612)

[Visualisierung 1 (Mesh Generation) 11](#_Toc66644613)

[Visualisierung 2 (Point Cloud) 12](#_Toc66644614)

[Visualisierung 3 (Displacement Shader) 12](#_Toc66644615)

[Visualisierung 4 (Volumetrischer Shader) 12](#_Toc66644616)

# Einleitung

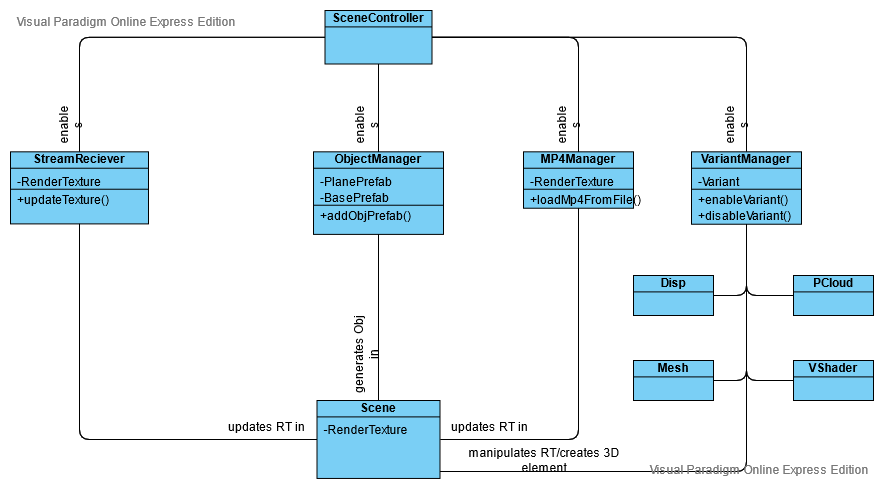
In den Abschnitten der einzelnen Arbeitswochen wird jeweils der Fortschritt zur vorherigen Woche beschrieben.

Dieses Dokument dient als Dokumentation der Prototypen Entwicklung für das Projekt: stereoskopische Echtzeitdarstellung von Personen. Dieses Projekt, in Zusammenhang mit einer schriftlichen Ausarbeitung im Rahmen einer Bachelorarbeit, hat den Fokus eine Anwendung für den Anwendungsbereich Office zu entwickeln. Diese Anwendung beschäftigt sich mit der Umwandlung von Daten gewonnen durch einen RGBD-Bilddaten Stream in, für eine HoloLense2 nutzbare, 3D-Objekte. Für die Entwicklungsumgebung werden wir Unity wählen. Als Ziel setzt sich die Implementation folgender notwendiger Teilbereiche:

* Verwaltung eines Datenstreams von einer Azure Kinect in eine Unity Umgebung
* Umwandlung der Daten des Streams in verwendbare 3D-Objecte
* Implementation von Manipulationvarianten
  + Displacement Shader
  + Mesh Generierung
  + Volumetrischer Shader
  + Punktewolke
* Erfolgreicher Deploy zu Endausgabegerät (HoloLense2)

Unity Version 2020.1.3

# Programmstruktur



# Milestones and Deadlines



## Varianten

Erwartungen für die Fertigstellung der Implementation/Analyse der Varianten

Fertigstellung von erstem Prototyp: 07.1

Fertigstellung von zweitem Prototyp: 07.1

Fertigstellung von drittem Prototyp: 14.1

Fertigstellung von viertem Prototyp: 21.1

Deadline für Fertigstellung von Varianten: 31.1

# Initialer Stand der Arbeit

## Research & Related Work

Arbeit startet mit ursprünglich vorgeschlagenen Ressourcen bezüglich der 4 Varianten, die im Abstract vorgeschlagen wurden. Ein Großteil der bis dahin gesammelte Materialien wurde aufgrund der Fokussierung auf „Officework“ ungeeignet gemacht und deshalb verworfen.

## Streaming

Azur Kinect läuft nur in einer Linux Umgebung. Primär wird diese von einem Nuc bereitgestellt. Eine bereits vorhandene GStreamer Architektur mixt den Farbstream und den Tiefenstream zusammen und schickt ihn erfolgreich übers Netzwerk. Derzeit ist eine Verzögerung in der Übertragung von ~2-3 Sekunden vorhanden. Farbstream ist in etwa 2/3 des Bildes. Tiefenstream besitzt Colorizer der in Tiefenstufen das Bild farbig einfärbt.

VLC-Mediaplayer zeigt derzeit gute Möglichkeiten und hat bereits bewiesen, den Stream empfangen zu können. Aufgrund von Vielseitigkeit von VLC und der Verfügbarkeit von VLC als Unity Plugin, wurde beschlossen Versuche zu unternehmen das Plugin für eine Einbindung des Streams in Unity zu benutzen. (Das Plugin ist kostenpflichtig, hat aber eine kostenlose Version mit einem Wasserzeichen)

MP4Loader.cs hält als Prototyp des MP4Manager und beinhaltet vorläufig den Code.

## Visualisierungen

-nc-

# Woche 1 (15.11 – 22.11)

## Research & Related Work

Unity verwandelt Videodateien intern in ein custom Format um. Eingehende Videostreams unterstützen H264 Encoding. Dies wird ebenfalls vom Kincetstream benutzt.

Render textures sind Texturen, zu denen man rendern kann. Für unsere Fälle ist dies

## Streaming

Integration von VLC Plugin funktioniert. Alternativen von UDP/TCP Clients zeigen keinen Erfolg (Obwohl man argumentieren könnte, dass dies dem Verständnis von Video stream Protokollen hilft, war die Unternehmung von vornherein ungünstig gewählt, da es sich um einen RTSP-Stream handelt.). VLC Player in Unity erstellt, verwaltet eine externe Textur, die den Stream entgegennimmt und erzeugt eine neue Textur, die benutzt wird um Daten anzuzeigen. Dies ist suboptimal und muss geändert werden, falls wir Rendertexturen verwenden wollen.

## Visualisierungen

-nc-

# Woche 2 (23.11 – 29.11)

## Research & Related Work

## Streaming

Implementation von VLC\_StreamReciever als Prototype des StreamRecievers. Erstellt aus der BasicPlayer Klasse des VLC Plugins.

Es gab einen Fehler in der MP4Loader der verhinderte, dass die vorbereitete Rendertextur benutzt wird. Gelöst durch Änderung in des Target Mediums des Unity Videoplayers zu Rendertextur und Angabe selbiger.

Experimente mit dem lesen der Pixeldaten haben eine temporäre Klasse erzeugt: TexReader.

Diese Klasse implementiert eine CPU basierende Lösung für das lesen und manipulieren der Rendertextur. Zu diesem Zweck liest die Klasse den Tiefenanteil der Textur und färbt Farbanteil basierend auf willkürlichem Kriterium. Gewähltes Kriterium: Tiefenwert Blauanteil > 5. Gewählt da Person in optimaler Position im Blaubereich des Tiefenbildes.

Probleme mit vorangegangener Lösung. Die Tatsache, dass Farb- und Tiefenbild nicht die selbe Auflösung haben, führt ohne geeignete Transformationsmatrixen zu einem Heiligenschein-Effekt um die Person. Weiterhin ist die Lösung bereits überfordert bei großen Texturen, dies wird bei der limitierten Hardware der HoloLense zu Problemen führen.

Erste Schritte mit der Hololense. Schwierigkeiten das Programm zu deployen.

## Visualisierungen

-nc-

# Woche 3 (30.11 – 06.12)

## Research & Related Work

Erste Schritte mit Shader unter Leitung von Robert Schmitz

## Streaming

Lösung der Probleme von letzter Woche. Kinect besitzt Einstellung „rectify“. Diese Einstellung passt Größe und Verzerrung des Tiefenbildes dem Farbbild an. Dies resultiert in einer 1:1 Pixel Relation von Tiefen und Farbbild. Zusätzlich wurde entschieden den Colorizer der die Farbeinteilung des Tiefenbilds verursacht auszuschalten und ein Graustufenbild zu übertragen. Das löst die Willkürlichkeit der Farbwahl.

Zusätzlich wurde TexReader durch die GPU basierte Lösung eines CutoutShaders abgelößt. Resultiert in stark verbessertem Verhalten bei großen Texturen.

Weitere Fortschritte mit der HoloLense. Deploy funktioniert jetzt und erlaubt Laufen der Programme. Wie zu erwarten ist die CPU basierte Lösung des TexReaders zu langsam und ineffizient, um auf der HoloLense verwendet zu werden. Allerdings obwohl Cutoutshader Variante eine bessere Framerate aufweist scheint es Probleme mit der Darstellung des Streams zu geben. Ein vorher sichtbarer Würfel wird unsichtbar sobald die Rendertextur die den Stream anzeigt auf den Würfel angewant wird. Die Textur bleibt schwarz und wird daher in einer HoloLense Umgebung unsichtbar. Der Grund ist unbekannt. Könnte an einem Nicht-laden des Streams liegen.

## Visualisierungen

-nc-

# Woche 4 (07.12 – 13.12)

## Research & Related Work

Fokus auf dem Ansammeln von zitierbaren Quellen für schriftliche Ausarbeitung

## Streaming

Rückschlag mit VLC Media Player. Es stellt sich heraus, dass das Betriebssystem der Hololense explizit nicht von VLC unterstützt wird. Dies verursacht das Nicht-Empfangen des Streams auf der HoloLense. Als Alternative wird AV Pro vorgeschlagen.

## Visualisierungen

-nc-

# Woche 5 (14.12 – 20.12)

## Research & Related Work

Erste Schritte mit volumetrischen Shadern unter der Leitung von Robert Schmitz

Fokus auf dem Ansammeln von zitierbaren Quellen für schriftliche Ausarbeitung

Schreiben der Arbeit und das Sammeln der Quellen kommt nur schleppend voran, aufgrund falschem Suchverhalten und ungenügender Notizen zu bereits gelesenen Quellen.

## Streaming

Sammlung von Testdaten mit verschiedenen EInstellungen bezüglich der Kinect. Es stellt sich heraus dass die Kinect wenn sie auf einem weißen Untergrund steht, signifikante Einbüßungen macht bezüglich erfolgreich gelesenen Tiefendaten. Vermutlich durch das Reflektieren der Strahlen an der weißen Oberfläche.

## Visualisierungen

-nc-

# Woche 6 (21.12 – 23.12)

## Research & Related Work

Fokus auf dem Ansammeln von zitierbaren Quellen für schriftliche Ausarbeitung

Ungünstiges Schreibtempo. Verglichen mit erwartetem Tempo sollte mehr als doppelt soviel Material erzeugt werden wie bislang vorliegt.

Sammlung von Quellen geht gut voran. Ich befürchte allerdings dass zu viel Faltung vorliegt und ich mehr Quellen zu den eigentlichen Varianten benötige als ich zu diesem Zeitpunkt benutze.

## Streaming

-nc-

## Visualisierungen

-nc-

# Woche 7,8,9 (28.12 - 10.01)

## Research & Related Work

Umstrukturierung des schriftlichen Anteils von “Materials and Methods” zu “Related Work”. Dies beinhaltet erste Paragraphe zu den Themen um Virtual Reality, Avatar Kreation und Nutzen von Virtueller Realität in einer professionellen Arbeitsumgebung.

Weiter Fortschritte in Research zu Anwendung der Varianten in verwanden professionellen Umgebungen (Militär, Soziale Umgebungen, Medizin).

Prototype Developement Teil der Arbeit beinhält nun mehr Teile bezüglich der genutzten Software und Hardware Umgebung.

## Streaming

Komplette Neu-Implementierung von sämtliche Hintergrundstrukturen des Programms. “Prototype ” Strukturen wurden ersetzt durch Struktur näher an der angedachten Programmstruktur. Dies beinhaltet die Implementation der, von dem SceneController gesteuerten, Layerstruktur der nötigen Hintergrundprogramme. Fürs erste, um spätere Additionen von Interfaceoptionen zu erleichtern, wurde sämtliche Steuerung über Editoroptionen der SceneController Klasse geregelt.

## Visualisierung 1 (Mesh Generation)

Prototype Phase der Mesh-Generierung-Variante. Implementation von Mesh Generierung durch das Erstellen von individuellen Vertices, basierend auf Tiefendaten der RenderTextur. Durch Zusammenfassung dieser Vertices zu Dreiecken und Neuberechnung der Normalen wird eine sichtbare Mesh erstellt.

Für die Prototype Phase wurde die Generation auf 250\*250 Pixel beschränkt, um Verzögerungen bei der Bilderstellung vorzubeugen. Hier zeigen sich bereits erste klare Bilder, dass Mesh Generierung aufgrund seiner hohen Rechenzeit möglicherweise nicht geeignet ist, um Echtzeitdarstellungen zu erzeugen. Parallele Berechnung von Blöcken der Textur, wäre eine Option die nötige Rechengeschwindigkeit zu erreichen. Dies muss noch erforscht werden.

## Visualisierung 3 (Displacement Shader)

Prototypen Phase des Displacement Shader Ansatzes. Weiterentwicklung des Cutout Shaders Ansatzes, mit Displacement Elementen erzeugt einen Shader der die gesamte Textur anhand ihrer Tiefenwerte Verschieben kann. Der Shader implementiert nun auch die Optionen den Abschnitt des Tiefenwert variable zu halten, sowie eine Anpassung des Alpha-Wertes der Fragmente basierend auf Distanz zu jenem Abschnittspunkt. Diese Variablen werden im Moment durch Slider im Editor angepasst.

Aufgrund der niedrigen Vertex Anzahl der primitive Plane in Unity, erhalten wir im Moment keine ausreichend detaillierte Vertiefung durch den Shader. Für dieses Problem sind zwei Lösungsansätze bekannt. Auf der einen Seite kann über Tesselation im Graphik-Prozess die Vertices aufgeteilt werden, oder alternativ kann ein neues PlaneObject erstellt/importiert werden das bereits über eine ausreichende Anzahl an Vertices verfügt. Fürs erste wurden Versuche gestartet Tesselation zu benutzen.

# Woche 10 (11.01 - 17.01)

## Research & Related Work

Research in OpenGL Shader Technologie spezifisch zu Transformation von Koordinaten Systemen

## Streaming

Probleme durch das Senden von Nachrichten während des OnValidate() Schrittes in der SceneController Klasse wurden behoben.

Implementierung einer Custom Plane Creation - Methode in der ObjectManager Klasse.

## Visualisierung 3 (Displacement Shader)

Es wird angenommen, dass durch Verschiebung entlang der Normalen der Vertices, der gewünschte Verschiebungseffekt erzielt wird. In bisheriger Implementierung wurde allerdings aufgrund von Verschiebungen in ungewünschte Richtungen eine Richtung angegeben. Dies resultiert möglicherweise von der Verschiebung während des falschen Tranformationsschritts der Bildschirmpixel. Verschiebung wird daher nun in Worldspace getätigt, anstatt Clipspace.

Weiterhin Tesselation wurde zu Gunsten besser steuerbarer Objekt Kreation ausgetauscht. Der ObjectManager erstellt nun zu Beginn des Displacement Prozesses eine benutzerdefinierte Plane mit einer feineren Meshstruktur. Dieser Schritt ist definitiv von Erfolg gekrönt. Die Erstellung des PlaneObject nimmt kaum Zeit ein und erhöht die Displacement Qualität ungemein.

## Visualisierung 4 (Volumetrischer Shader)

Erste Tests zu Implementierung des volumetrischen Ansatzes.

# Woche 11 (18.01 - 24.01)

## Research & Related Work

## Streaming

## Visualisierung 3 (Displacement Shader)

/\* Unzureichend genau

Implantation von Filtern um die Stördatenanzeil im Tiefenbild zu verringern. Dies sollte mir erlauben die Displacement Faktoren zu erhöhen und so meinen Wertebreich aus Range 0.001 zu 0.02 in einen besser zu skalieren. Dies sollte in einem angemssen verschobenen Vertec resultieren.

Weiterhin wurden Experimente mit smoothing Filtern betrieben ebenfalls im Sinne die Tiefenwert an den Rändern und in der Mitte der Person stabil zu halten. Ich war anfänglich skeptisch ob der zusätzliche Ballast eines Tiefenbuffers in den Berechnungen ausreichend Ergebnisse erzielt.

Der Tiefenbuffer hat sich allerdings als nicht hilfreich erwiesen. Die Menga an Störungen und Dauer derer hat als zu gravierend erwiesen als dass einfaches smoothen ausreichend wäre.

# Woche 12 (25.02 - 31.02)

## Research & Related Work

Arbeit an der schriftlichen Seite des Projekts muss vorraungetrieben werden. Mir wurde nahegelegt Aufbau bezüglich der Sektion von related Work noch einmal zu überdenken. Dementsprechend wird nun der related Work Anteil ebenfalls die 4 Varianten beinhalten während der Inplementationsteil stärker auf die Implementation fokussiert sein wird. Allgemein wurden erste Implementationsspezifikationen eingetragen, auch wenn ich erwarte, dass dies noch geändert werden.

## Streaming

Ergebnisse letzter Woche im Displacement Shader haben gezeigt, dass keine ausreichende Verschiebung der Vertices stattfindet. Versuche mit höheren Displacement Faktoren resultiert in großen Zacken, die nicht erwünscht sind. Untersuchungen der aufgezeichneten Testdateien zeigen, dass Graustufen (vor allem zwischen Gesichtszügen) nicht voneinander abzugrenzen sind, was in flachen Gesichtern und Oberkörpern resultiert.  
  
Untersuchungen des Graustufenbildes bevor Komprimierung durch GStreamer plugin in H“64 vormat zeigt noch Unterschiede zwischen den Graustufen. Als Lösung wurde vorgeschlagen vom Graustufenbild wieder auf den Colorizer zu wechseln. Der Colorizer der anfänglich aufgrund von Manipulation des Farbwertes durch einen Table komplizierte Farbabstufung hervorruft, scheint einfach die Farbwerte auf einen Bereich von 1-256 zu skalieren und per Linksshift als RGB Werte zu erzeugen.

Mit Hilfe wurde dementsprechend ein einfacher Codeblock geschrieben, der die Tiefendaten von ihren Farbwerten wieder zurück in eine linear wachsende Abbildung ihrerselbst verwandeln kann.

Dies erlaubt mir nun, ohne das GStreamer Plugin zu manipulieren, wieder Zugriff auf den dort eingebauten Nearcut/Farcut derin der Graustufen Variante nicht enthalten ist. Wie sich herausstellt skalliert das Plugin seinen Werteberech zwischen diesen Abschnitten. Dies erlaubt nun für wesentlich feinere Abstufungen zwischen den Tiefenwerten und macht Gesichter und Körperabbildungen wieder möglich.

Fürs erste da, mein primäres Ziel Darstellungen des gesamten Körpers sind, habe ich mich auf einen Nc/Fc von 300/2000 entschieden. Dies scheint einer Person ausreichend Platz zu geben. Andere Einstellungen mit geringerem Wertebereich (nur Gesicht?) könnten interessant sein.

Streaming an sich gewinnt nun, nach dem Rückschlag mit VLC, wieder an Relevanz. Ich bin allerdings nicht nicht dazu gekommen die Testversion des neuen AvPro Plugins zu verifizieren.

## Visualisierung 2 (PointCloud)

Will require additional Research and Codereview of Pcx.

## Visualisierung 3 (Displacement Shader)

Der Displacement Shader hat durch den Wechsel zum farbigen Tiefenbild enorm an Darstellungsqualität gewonnen. Durch den Wechsel wurden auch einige Probleme mit meiner Implementation des Abblendbereiches, der den Alphawert der Pixel basierend auf Distanz manipuliert, klar. Formel wurde und besitzt nun einen festen Bereich mittig zwischen Nc/Fc das vollen Alphawert besitzt und abnehmendes Alpha außerhalb.

Das Tiefenbild scheint allerdings immer noch Halo-Effekte hervorzurufen, die besonders nahe an den Nearcut Plane herausstechen. Mir wurde nahegelegt einen Errosionsfilter zu benutzen, um das Tiefenbild dementsprechend zu säubern.

Das Tiefenbild allgemein ist durch den eingebauten Nearcut auf Gstreamer Seite wesentlich weniger „noisy“. Methoden die letzte Woche entwickelt wurden, um diese Störungen entgegenzuwirken sind fast unnötig geworden.

Anstelle werden weiter Optionen benötigt das Bild stabil und dennoch flexibel zu halten. Vielleicht sogar Geometrie Shader Anteil der scharfe Kanten an den Rändern der Person abschwächt.

## Visualisierung 4 (Volumetric Shader)

Weiter Fortschritte mit dem volumetrischen Ansatz. Durch neu gewonnene Genauigkeit wurden grundlegende Schritte gemacht. Der Shader funktioniert bei frontaler Ansicht, da nur ein einziges Backface betrachtet wird. Bei Ansicht von der Seite allerdings stellt sich ein Akkordeon Effekt ein, dessen Ursprung sich findet in der Abschnittsfunktion die Tiefe basierend auf z-Wert des Tracers misst.

Weiterhin existieren runde Formen auf der Außenseite des Würfels, die ebenfalls durch den Errosionsfilter behoben werden könnten.

Ich erwarte eine Selektion der Vertices die Tracer benutzen könnte Erfolge zeigen. Dies allerdings muss noch gezeigt werden.

# Woche 13 (01.02 – 07.02)

## Research & Related Work

## Streaming

## Visualisierung 1

## Visualisierung 2 (PointCloud)

Gängige Strukturen bei PointClouds bestehen oft, wenn nicht aus speziellen PointCloud Dateien (zB. .ply), aus einer Kombination aus positions-Buffer und Farb-Buffer welche dann per Graphics.DrawProcedural als Point Struktur direkt von der Graphikkarte gemalt werden. Versuche die jetzige Textur Struktur in diese Form durch pure CPU-Leistung zu überführen, führen zu niedrigen FPS Zahlen wie sie auch bei mesh Generierung zu beobachten sind. Alternativen bestehen in ComputeShadern die die nötigen Daten Strukturen schneller erstellen könnten und besser mit den große mengen an Vektor und Textur daten umgehen können, zusätzlich zu ihrer parallelen Bearbeitung dieser. Unglücklicherweise sind wir ebenfalls nicht in Besitz einer Mesh-Generierungs Methode, um die dortigen Vertices einfach in Punkte umzuwandeln. Weiteroptionen bestehen in der Implementation eines Geometrie Shaders der Dreiecke oder Quads erstellt die als Punkte der Punktewolke dienen.

Der ComputeAnsatz stellt sich momentan noch als schwierig heraus. Lesen der Texturen sowie einfacher Zugriff auf UV-Coordinaten oder Positionen ist erschwert. Vielleicht ist ein Hybrid Ansatz eines ComputeBuffers innerhalb eines Vertex/Fragment Shaders eine bessere Option.

## Visualisierung 3

# Woche 14,15,16,17,18 (08.02 – 15.03)

Die letzten Wochen des Projekts sind angebrochen und wir müssen das Project zu einem baldigen Ende bringen um mehr Fokus auf die nebenläufige Ausarbeitung setzen. Dies ist ein Bericht, der gegen Ende der letzten Woche erstellt wurde und als Zusammenfassung dienen soll

## Research & Related Work

Ende der wissenschaftlichen Arbeit. Über die letzten Wochen wurde verstärkt Fokus auf die Ausarbeitung gelegt. Viele Bereiche die bislang nicht bearbeitet wurden, wurden fertiggestellt nicht ohne Hilfe der mit zugewiesenen Betreuer.

## Streaming

Die Entwicklung an den Streaming Funktionalitäten wurde frühzeitig beendet. Die AVPro Test-Version hatte keinen Erfolg auf den Stream auf der HoloLens zuzugreifen. Endstand ist demnach eine funktionierende Verbindung zwischen der Unity Umgebung mit VLC Plugin und dem RTSP-Stream

## Visualisierungen

Die Arbeit an den Visualisierungen wurde beendet. Alle Visualisierungen sind in der Lage auf der HoloLens dargestellt zu werden und implementieren Single Pass Instanced Rendering.

## Visualisierung 1 (Mesh Generation)

Die Arbeit an Mesh Create wurde beendet. Es wurde beschlossen, dass Visualisierung 3 in seiner Idee verwandt genug ist um zusammen mit verbesserter Performanz Visualisierung 1 als überholt zu klassifizieren. Visualisierung 1 ist daher mit seiner letzten Implementation die auf Tastendruck einen Teil der Rendertextur erzeugt als fertig eingestuft worden.

## Visualisierung 2 (Point Cloud)

Die Arbeit an Point Cloud wurde beendet. Am Ende wurden überdacht Ansätze im Bereich Compute Shader und Draw.Procidural. Diese Ansätze wurden zugunsten einem Ansatz der mehr an dem Deplacement Shader liegt verworfen. Der Ansatz erzeugt mit Hilfe eines Geometrie Shaders Punkte Haufen. Dies hat den Vorteil, dass die Einschränkung durch Unity bezüglich der Vertex Anzahl einer Sub-Mesh übergangen werden kann. Eine Variante die Dreiecke erzeugt wurde ebenfalls implementiert wurde aber kurz vor Ende durch die Punkte Haufen ausgetauscht, um sich weiter von der Displacmeent Variante abzugrenzen.

## Visualisierung 3 (Displacement Shader)

Die Arbeit am Displacement Shader wurde beendet. Erosion und Dilatation Filtermethoden haben die Methode stark verbessert zusammen mit den Tiefenblending und starker Performanz hat sich Visualisierung 3 als die beste Methode herausgestellt.

## Visualisierung 4 (Volumetrischer Shader)

Die Arbeit am Volumetrische Shader wurde beendet. Qualität des Shaders wurde stark verbessert auch wenn Performanz darunter leiden musste.

# Fazit

Am Ende wurden alle Visualisierungen zu einem Ende gebracht und erfolgreich zur HoloLens deployed. Streaming zur Unity Umgebung wurde hergestellt. Ziele wie sie beschrieben sind wurden erfüllt.

Milestones und Deadlines wurden nicht eingehalten und haben immer länger gedauert als geplant. Optimierungsfortschritt der einzelnen Visualisierungen ist nicht besonders befriedigend.

Funktionalitäten der HoloLens wie das Hand Menü oder Ziehen der Hologramme wurden nicht fertiggestellt.

Streaming zur HoloLens ist ebenfalls nicht mehr Teil der Implementierung.