

Onderzoek naar optimalisatie van 3D-modellen voor AR-applicaties

Door bestaande optimalisatie technieken toe te passen voor 3D-modellen

Door **Roberto Gemin**

Studentnummer: **137723**

Afstudeerbegeleider: **Tim Roossen**

Afstudeerbedrijf: **Twinsense**

Bedrijfsleider: **Albert Hoekstra**

Bedrijfsbegeleider: **Robin Kuiper**

Voorwoord

Voor u ligt de scriptie ‘Onderzoek naar optimalisatie van 3D-modellen voor AR-applicaties’ . Deze scriptie is geschreven in het kader van mijn afstuderen aan de opleiding Creative Media and Game Technologies van de Saxion Hogeschool in Enschede en in opdracht van stagebedrijf Twinsense.

Als eerste wil ik mijn astudeercoach Tim Roosen bedanken, voor zijn feedback en advies over het schrijven van deze scriptie.

Verder wil ik de bedrijfscoach Robin Kuiper bedanken voor de tijd die hij vrij heeft gemaakt voor gesprekken. Deze gesprekken hebben geleid naar een duidelijke definitie van mijn probleemstelling.

Verder wil mijn projectgroep bedanken voor de dynamische werkomgeving en de leerzame gesprekken die we met elkaar hebben gedeeld.

Als laatst wil ik mijn vrienden en familie bedanken voor de motivatie om mijn honderd procent te geven aan dit onderzoek.

INHOUDSOPGAVE

1. Introductie	8
2. De opdrachtgever	11
3. Betrokken partijen	15
4. Aanleiding van de probleemstelling	16
5. Probleemstelling	18
6. Achtergrondinformatie	20
7. Theoretisch kader	24
8. Onderzoeksvragen	30
9. Methodieken	32
10. Resultaten	46
11. Conclusie	50
12. Discussie	53
13. Aanbevelingen	54
Bijlage 1. Optimalisatie SketchUp 3D-model	59
Bijlage 2. Model informatie	61
Bijlage 3. Onderzoeksproduct	65
Bijlage 4. Screenshots	67
Bijlage 5. Adviesrapport	81

Begrippenlijst

API -	Application programming interface, een communicatie protocol.
CPU -	Central Processing Unit, een processor.
FPS -	Frames per seconde.
GPU -	Graphics Processing Unit, voert renderingen taken uit.
Draw Call -	Render opdracht naar de grafische API.
OpenGL -	OpenGL, een grafisch API, die communiceert met de GPU.
Unity3D -	Een game engine.
Setpass Calls -	Geeft de opdracht aan de GPU welke setting gebruikt moet worden voor de volgende mesh die gerenderd wordt.
Render state -	Verandering van de status van de state machine van OpenGL.
Mocap suit -	Een pak met sensoren dat bewegingen registreert.

1. Introductie

Augmented Reality(AR) is het samenvoeging van twee werelden, virtuele wereld en de echt wereld. De virtuele wordt op een display bekeken. Er zijn verschillende apparaten die AR gebruiken smart glasses, smartphones en tablets (zie figuur 1).



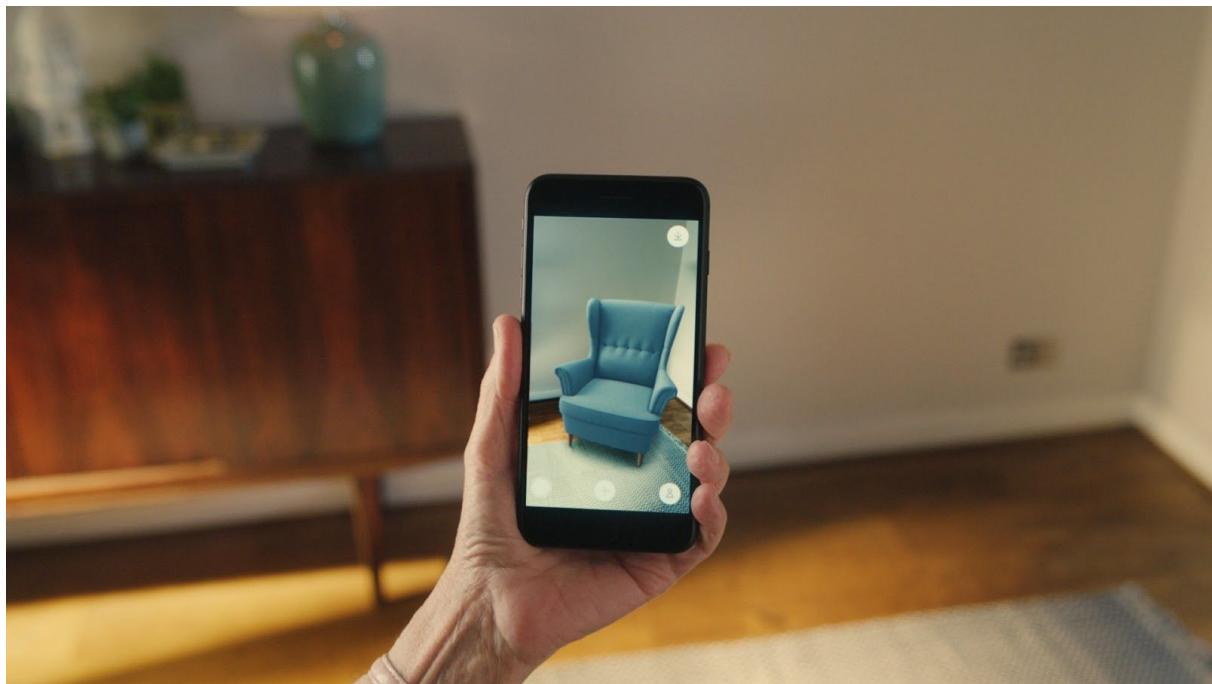
Figuur 1: Links boven een tablet, links onder smartphone ,rechtsboven Microsoft Hololens en rechts onder de Meta2.

AR heeft een ontwikkeling doorgemaakt in het aantal gebruikers en de AR-apps die gedownload worden; In 2018 waren er 0,5 miljard meer smartphone gebruikers in vergelijking met het vorige jaar. Het totale aantal AR smartphone gebruikers in 2018 was 1,1 miljard (Superdata research z.d.). Merel(2018) maakt een schatting dat er binnen de komende vijf jaar dat er 1.3 miljard smartphones apps worden geïnstalleerd.

De toename van AR-smartphone gebruikers en downloads is een gevolg van de volgende ontwikkeling:

- De ontwikkeling van efficiënte hardware zoals CPU en GPU van smartphones.
- Ontwikkeling van toolkits om AR-applicatie te ontwikkelen.
- Beschikbaarheid van distributiekanaal van AR-applicaties.

Door vernieuwingen van de GPU kunnen complexe 3D-modellen gerenderd worden. Vervolgens door de toename van CPU rekenkracht kunnen functionaliteiten gebruikt worden om bijvoorbeeld op de oppervlakte van een omgeving een 3D-model te plaatsen (zie figuur 2).



Figuur 2: Projecteren van een 3D-model in een omgeving.

Bedrijven zoals Google, Apple (Horwitz, 2018) en Vuforia hebben bepaalde toolkits ontwikkeld. Deze toolkits geven de mogelijkheid om AR-functionaliteiten toe te voegen aan smartphone applicaties (Unity Technologies,z.d.-d)

Door de groei van het aantal AR-gebruikers in de komende jaren en samenhangende groei van het aantal apps dat wordt geïnstalleerd wil ik specialiseren in het ontwikkelen en optimaliseren van AR-applicaties. Vandaar mijn keuze voor het communicatiebureau Twinsense dat AR-applicaties ontwikkelt.

Dit afstudeeronderzoek is uitgevoerd in opdracht van het communicatiebureau Twinsense. Tijdens het afstuderen bood ik ook ondersteuning in een andere projectopdracht. De opdracht is uitgevoerd binnen het project *Tabletop RPG with VR and AR* voor Twinsense. De projectgroep bestaat uit Saxion studenten en één student van de Universiteit van Twente. Dit onderzoek belicht een specifieke probleemstelling van de ontwikkelaars en designers van Twinsense.

De opbouw van het scriptie gaat als volgt. In het volgende hoofdstuk wordt het bedrijf Twinsense geïntroduceerd en de aanleiding van projectopdracht.

In hoofdstuk 3 wordt het projectgroep voorgesteld die opdracht wordt uitgevoerd.

In hoofdstuk 4 wordt de aanleiding van de probleemstelling introduceert. In hoofdstuk 5 wordt de probleemstelling gedefinieerd.

In hoofdstuk 6 wordt de achterinformatie informatie beschreven van het productie proces van de ontwikkelaarse en die designers van Twinsense.

In hoofdstuk 7 wordt informatie gegeven over het probleemstelling en hoe het opgelost kan worden.

In hoofdstuk 8 wordt de hoofd en subvragen beschreven.

In hoofdstuk 9 worden de methodieken beschreven hoe het onderzoek uitgevoerd moet worden.

In hoofdstuk 10 worden de resultaten gepresenteerd.

In hoofdstuk 12 wordt de conclusie beschreven.

In hoofdstuk 13 wordt de aanbeveling geschreven.

2. De opdrachtgever

Twinsense is begonnen in 2005 als een communicatiebureau. De doelstelling van Twinsense is om het imago van bedrijven te versterken, wat moet resulteren in klantenwerving.

Dit gebeurt door:

- Social Media Strategieën
- Corporate Identity en huisstijlen
- Bedrijfsvideo's

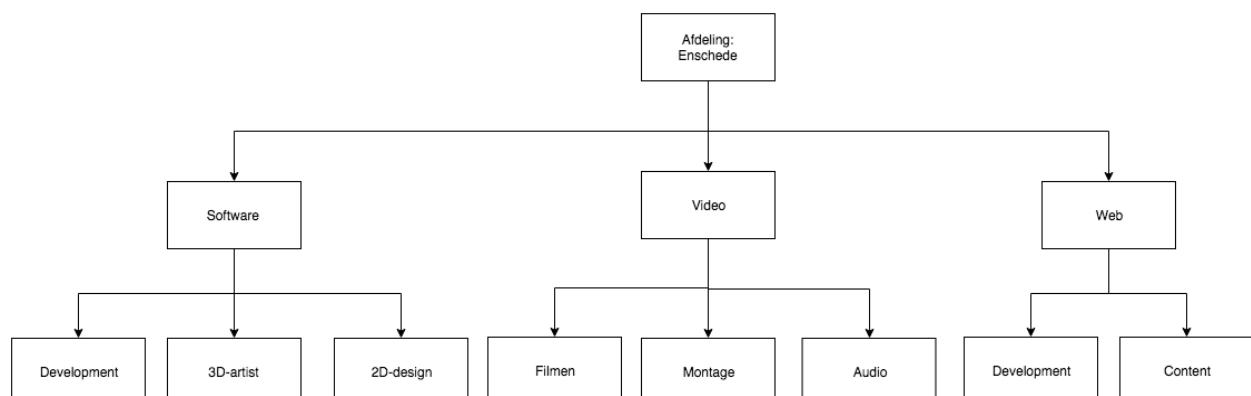
In 2016 is Twinsense begonnen met een uitbreiding. Er kwam een dochteronderneming bij voor het ontwikkelen van VR, AR en 360 video. Twinsense zag hier potentieel in, omdat verschillende klanten behoefte hadden aan training en promotionele applicaties. Voor dit onderzoek wordt de focus gelegd op de dochteronderneming genaamd Twinsense360.

2.1. Afdelingen bij Twinsense en Twinsense360

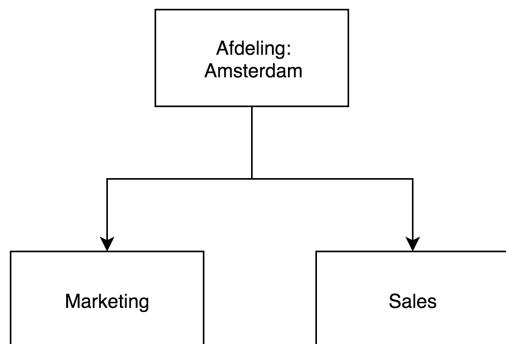
Twinsense en Twinsense360 zijn opgedeeld in twee gezamenlijke vestigingen, één bevindt zich in Enschede en de ander in Amsterdam.

In Enschede worden de videocontent, webpagina's en applicaties ontwikkeld (zie figuur 1).

In Amsterdam worden de marketing campagnes ontwikkeld en bevinden zich de sales (zie figuur 2).



Figuur 1: Afdeling Enschede



Figuur 2: Afdeling Amsterdam

2.2. Twinsense360 klanten

De klanten van Twinsense komen voornamelijk uit de technische en/of productie sector. Een typisch project is de visualisatie van 3D-modellen (aan de hand van AR en VR) ter presentatie op beurzen. De aangeleverde 3D-modellen worden ontwikkeld door de klanten van Twinsense. Door de ontwikkelaars wordt het 3D-model geïmplementeerd in een AR of VR applicatie.

2.3. De AR en VR producten voor klanten

Twinsense ontwikkelt AR-applicaties en VR-applicaties in de game engine Unity3D. In Unity3D kan Twinsense VR- en AR-applicaties ontwikkelen met behulp van externe software.

Voor AR-applicaties worden de volgende externe software gebruikt: Vuforia, AR-core en AR-kit.

Voor VR-applicaties wordt SteamVR gebruikt.

Het komt ook voor dat Twinsense zelf de apparaten ontwikkelt, denk aan het gebruik van een Arduino apparaat die via seriële poorten informatie verstuur naar een applicatie

2.4. Medewerkers binnen Twinsense360

De medewerkers zijn gespecialiseerd in het ontwikkelen van producten voor specifieke technische vraagstellingen van klanten. De medewerkers hebben specifieke kennis die ze hebben opgedaan vanuit zelfstudie, via een opleiding of via het uitvoeren van opdrachten voor klanten. Op dit moment werken er in Enschede op de afdeling AR en VR twee vaste medewerkers, namelijk voor het programmeren van applicaties Robin Kuiper en het modelleren van 3D-modellen John Keane. Verder werken er afwisselende hoeveelheid stagiaires.

De werkzaamheden van Robin Kuiper zijn:

- Het ontwikkelen van AR, VR of 360 applicaties in Unity3D.
- Het begeleiden van afstudeerders.

De werkzaamheden van John Keane zijn:

- Het optimaliseren van aangeleverde 3D-modellen van klanten.
- Het ontwikkelen van 3D-modellen.
- Het testen van 3D-modellen in Unity3D.

Twinsense360 zoekt voornamelijk stagiaires van HBO of WO niveau.

De stagiaires kunnen binnen Twinsense360 bij de afdeling programmeurs of 3D-design stage lopen.

Bij de afdeling voor programmeurs is het van belang dat de stagiair minimaal één jaar ervaring heeft in het ontwikkelen van applicaties binnen Unity3D.

Voor de 3D-designers is het van belang om kennis te hebben van Unity3D, Photoshop en een 3D-modelleer applicatie.

2.5. Vragen buiten de specialisatie

Onderstaande vraagstellingen kwamen binnen van klanten van Twinsense met betrekking tot het presenteren van content. Door een tekort aan tijd als ook relevante kennis, kon Twinsense deze vraagstellingen tot nu toe niet behandelen. Niettemin vond Twinsense het belangrijk dat deze vraagstellingen onderzocht zou worden. Hiervoor hebben ze meerdere sub-opdrachten ontwikkeld voor een projectgroep van studenten van Saxion en de Universiteit Twente.

De vraagstellingen van klanten zijn als volgt beschreven:

- Kan de Mocap Suit in Virtual Reality (VR) applicatie gezamenlijk werken met de HTC Vive?
- Kan de HTC Vive gecombineerd worden met een face tracking camera?
- Kan er data verzonden worden tussen een AR en VR applicatie?
- Kunnen meerdere AR-applicatie met elkaar communiceren?
- Is het mogelijk om externe modellen in een VR scène te plaatsen?
- Is het mogelijk om externe modellen in een AR scène te plaatsen?
- Is het mogelijk om verschillende levels te genereren van opgeslagen data van een AR applicatie.
- Is audio communicatie mogelijk tussen AR-applicaties onderling.
- Is audio communicatie mogelijk tussen een AR-applicatie en een VR applicatie.

Aan de hand van bovenstaande vraagstellingen heeft Twinsense de volgende sub-opdrachten beschreven:

- Data-overdracht tussen de Mocap suit en de VR-applicatie.
- Registreren van gezichtsuitdrukking met de HTC Vive via een face tracking camera.
- Data-overdracht via wifi tussen meerdere AR-applicaties en één VR-applicatie.
- Plaatsen van externe 3D-modellen in een VR-applicatie.
- Plaatsen van externe 3D-modellen in een AR-applicatie.
- Multiplayer in een AR applicatie.
- VoIP tussen AR-applicaties onderling.

2.6. Vraagstelling van dit onderzoek

De vraagstelling van dit onderzoek is ontstaan tijdens het uitvoeren van de sub-opdracht: ‘Plaatsen van externe 3D-modellen in een VR-applicatie’. De externe 3D-modellen die gebruikt worden zijn 3D-modellen van het internet en van designer Adi Leka van de projectgroep *Tabletop RPG with VR and AR*.

De overige sub-opdrachten worden niet onderzocht in deze scriptie maar worden wel ondersteund door de auteur van dit onderzoek.

2.7. De aannemers van de opdracht

Om de specifieke sub-opdrachten te onderzoeken werd een projectgroep van studenten van Saxion en Universiteit Twente samengesteld. Deze groep bestond uit studenten die afstuderen, een minor of een specialisatie volgden. Om de verschillende vraagstukken op te lossen had Twinsense een spelconcept ontwikkeld.

Het spelconcept had meerdere doelen:

- Uitdagender maken voor studenten.
- Studenten kunnen hun eigen ideeën geven aan de eind-applicatie.
- Het spelconcept kan worden gebruikt voor sociale media uitingen, met het presenteren van nieuwe mogelijkheden van AR en VR applicaties.

2.8. Het spelconcept

Aan het spelconcept is een thema toegevoegd. Het thema van het spel is gebaseerd op het bordspel Dungeon and Dragons (DnD). De vormgeving van DnD is geïnspireerd op de klederdracht van de middeleeuwen met toevoeging van fantasy elementen (Zie figuur 3).



Figuur 3: “De vormgeving van DnD” (2018)

DnD is een bordspel dat gespeeld wordt met meer dan twee spelers. Één speler in het spel is de dungeon master (DM). De DM creëert een campagne, een speurtocht, dat gespeeld wordt door de rest

van de spelers. Het spelconcept leent zich aan de fantasy elementen en het principe van de dungeon master. Het spel wordt gespeeld met één HTC VR bril en met meerdere smartphones. De DM gebruikt de HTC Vive en creëert daarmee een wereld. De spelers gebruiken smartphones om door de wereld te navigeren en campagnes uit te voeren. De DM begeleidt spelers via visuele hints.

3. Betrokken partijen

De ontwikkelaars van Twinsense hebben een vaste werkmethode voor het ontwikkelen van AR en VR applicaties. Twinsense wil een creatieve inbreng hebben van een externe partij, vandaar de betrokkenheid van de studenten. Deze studenten zijn onder andere afkomstig van de HBO-opleidingen Creative Media and Game Technologies, ICT en Media, Technische Informatica, maar ook van de WO-opleiding Creative Technology.

3.1. Projectgroep

De projectgroep heeft de naam *Tabletop RPG with VR and AR*.

De projectgroep bestaat uit zestien leden, die worden opgedeeld in drie afdelingen (zie figuur 3):

- **De ontwikkelaars** zijn verantwoordelijk voor het programmeren van de AR-applicatie en VR-applicatie.
- **De 3D artiesten** zijn verantwoordelijk voor het visuele aspect.
- **De game designer** is verantwoordelijk voor de game logica en interactie design.

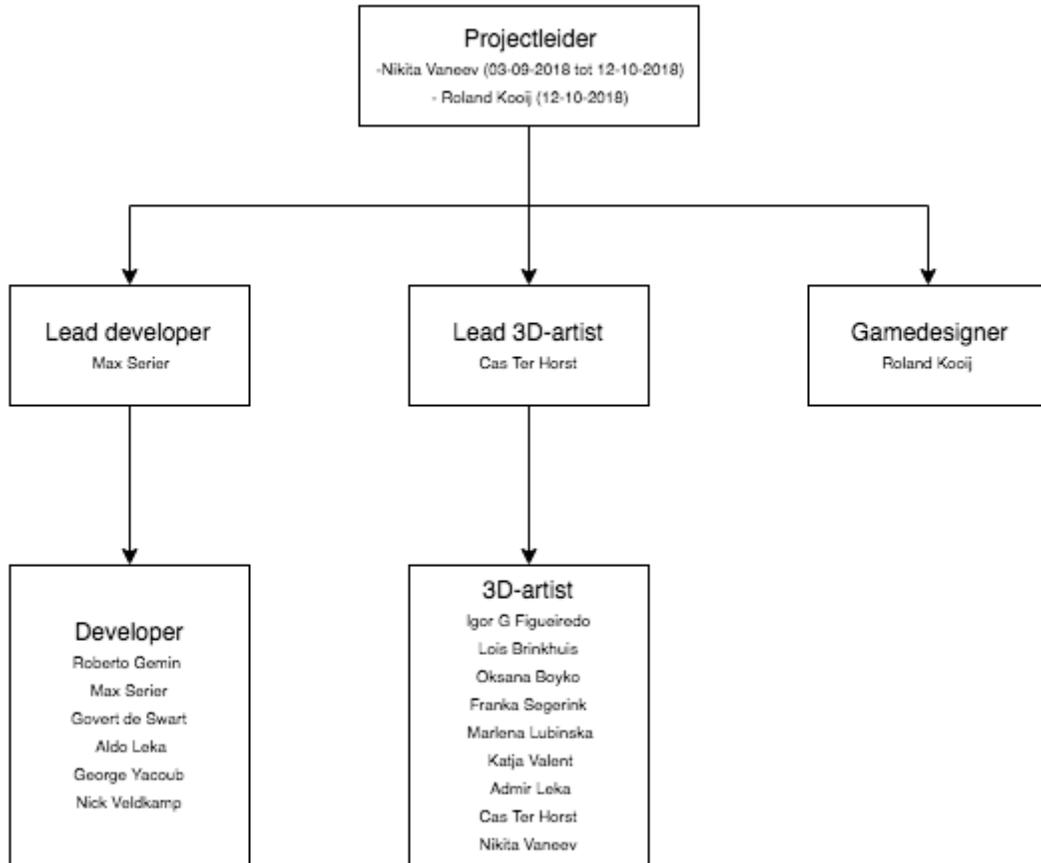
In de afdeling ontwikkelaars bevindt de onderzoeker (Roberto Gemin) van dit onderzoek.

De volgende werkzaamheden zijn uitgevoerd binnen het project door de onderzoeker:

- Ondersteuning bieden bij programmeeropdrachten van de ontwikkelaars.
- Advies geven op code architectuur.
- Ontwikkelen van prototypes.

Elke afdeling heeft een hoofdverantwoordelijke die communiceert met de projectleider.

De projectleider zorgt voor de communicatie met het bedrijf.



Figuur 3: Project hiërarchie

4. Aanleiding van de probleemstelling

Tijdens de ontwikkelingsfase van de opdracht die door Twinsense is voorgeschreven, heeft de auteur Roberto Gemin een probleemstelling geformuleerd.

De eerst taak van de ontwikkelaars van de projectgroep was het ontwikkelen van prototypes. De prototypes werden ontwikkeld volgens de opdrachtbeschrijving die Twinsense had samengesteld voor de projectgroep.

De volgende opdracht: "Plaatsen van externe 3D-modellen in een AR-applicatie." werd door de onderzoeker uitgevoerd. In de volgende paragraaf wordt de achtergrondinformatie beschreven van het ontwikkelingsproces van het AR-prototype.

4.1. Plaatsen van externe 3D-modellen in een AR-applicatie

Bij het ontwikkelen van het AR-prototype gebruikte de onderzoeker in de beginfase alleen externe 3D-modellen afkomstig van websites waarvan ze gratis te downloaden zijn.

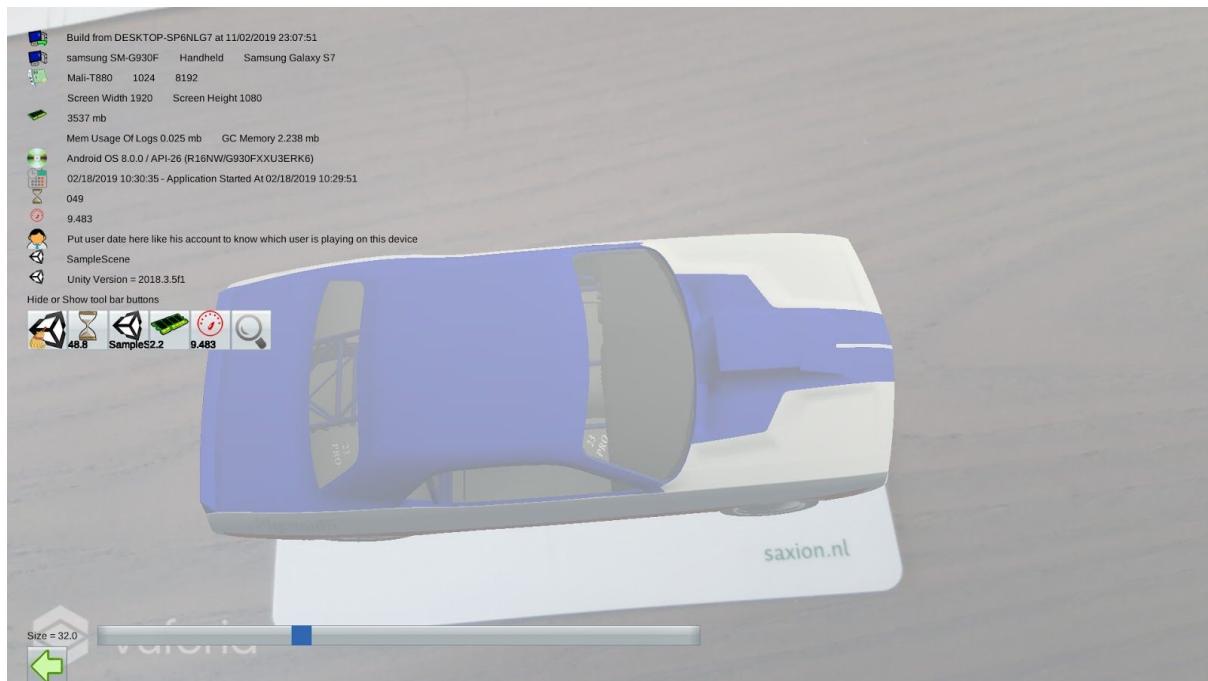
De reden daarvoor was dat het zelf ontwikkelen nog niet in de planning zat van de designers. Pas later in het project ontwikkelden de designers van de projectgroep eigen modellen.

De 3D-modellen werden geïmplementeerd in een applicatie die ontwikkeld is in Unity3D. In de applicatie werd een plugin van Vuforia gebruikt. Vuforia zorgt voor AR-functionaliteiten in de applicatie. De AR-applicatie is ontwikkeld voor smartphones.

4.1.1. Eerste testfase met een extern 3D-model

Tijdens het testen van de AR-applicatie was er een vertraging in de applicatie. Om te analyseren wat de oorzaak van het probleem was, werd de Log Viewer gebruikt (Dreammakersgroup, z.d.). Log Viewer is een softwarepakket dat toegevoegd wordt in een Unity3D project.

De Log Viewer laat de snelheid van de applicatie zien. De snelheid van de applicatie wordt in frames per seconde (fps) weergegeven. Uit de analyse kwam naar voren dat na het plaatsen van het 3D-model de framerate onder de 27 fps lag (zie figuur 4). Dit gaf een stotterend beeld. Een optimaal werkende AR-applicatie functioneert op 27 tot 30 fps.

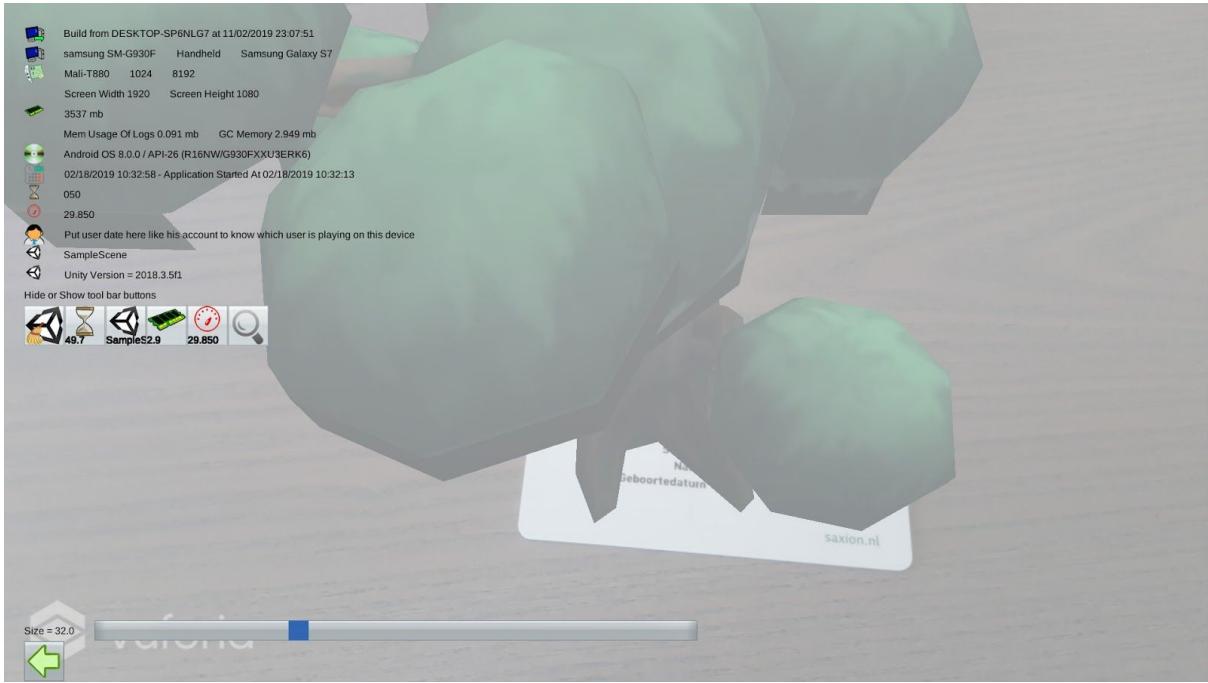


Figuur 4: SketchUp model rond de 9fps.

4.1.2. Tweede testfase met handmatig 3D-model

Bij de tweede testfase werd een 3D-model gebruikt die deze keer wel handmatig door een designer van het designteam was ontwikkeld. De 3D-model is ontwikkeld in de 3D-applicatie Maya.

Het model heeft een eenvoudige geometrie met één textuur. Na de test gaf het AR-prototype aan dat er geen vertraging plaatsvond in de applicatie. De Log Viewer (zie afbeelding 5) van de framerate zat rond de 30 fps.



Figuur 5: Maya 3D-model rond de 30 fps.

4.1.3. Conclusie van de eerste twee testfases

Een van de verschillen tussen de twee testfases was het 3D-model. De Log Viewer gaf bij de tweede testfase met een eigen ontwikkeld 3D-model de gewenste waarde. Een ander verschil is dat alle materialen een kleurwaarde bevatten zonder textuur.

Bij het vergelijken van beide 3D-modellen in de profiler van Unity3D op de desktop werd er bij het eerste 3D-model (zie figuur 6) een hoger batch aantal weergegeven in vergelijking met het tweede 3D-model (zie figuur 7). Een batch is een samenvoeging van twee opdrachten om een 3D-model te renderen. De eerste opdracht is de specificatie hoe het 3D-model gerenderd moet worden. De tweede opdracht is informatie over de eigenschappen van het 3D-model, dit wordt stapsgewijs verstuurd naar de GPU.



Figuur 6: Model met 9408 batches.



Figuur 7: Model met 3 batches.

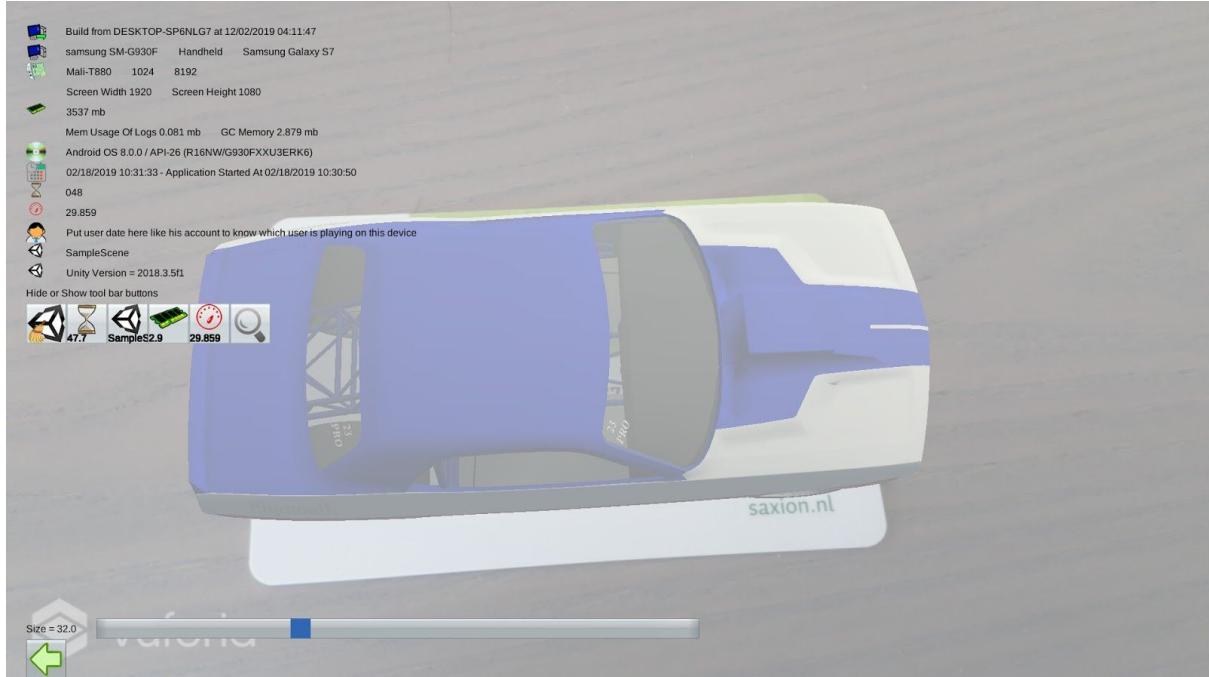
4.2. Aanleiding probleemstelling

Na het uitvoeren van de testfasen kwam een probleemstelling tot stand die geen hoge prioriteit had in het project van *Tabletop RPG with VR and AR*, maar wel voor Twinsense: een 3D-model kan performance problemen veroorzaken in een AR-applicatie.

Er is een gesprek gevoerd met Twinsense waarin de resultaten van de testfase werden besproken met de bedrijfsbegeleider, tevens ontwikkelaar. De bedrijfsbegeleider heeft voorgesteld om het 3D-model

met performance opnieuw te gebruiken en te optimaliseren (Zie bijlage 1: Optimalisatie SketchUp 3D-model).

Bij het testen van het geoptimaliseerde 3D-model functioneert de applicatie op de gewenste 30 fps (zie figuur 8).



Figuur 8: SketchUp model rond de 30fps.

Het resultaat van de optimalisatie van het 3D-model werd voorgesteld aan Twinsense.

Uit het gesprek over het resultaat blijkt dat de ontwikkelaars van Twinsense een soortgelijk probleem hebben met aangeleverde 3D-modellen.

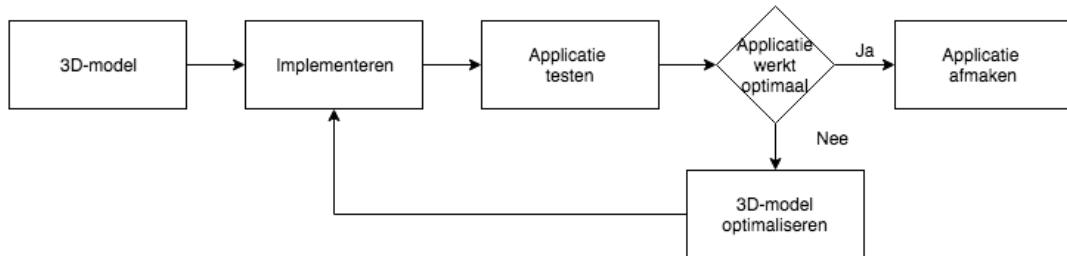
Twinsense werkt met externe modellen die zijn aangeleverd door klanten. De klanten van Twinsense gebruiken diverse modelleer-applicaties. Implementatie van door klanten aangeleverde 3D-modellen veroorzaken vaak stotteringen in de AR-applicaties van Twinsense.

5. Probleemstelling

In het proces van optimalisatie van een extern 3D-model is er een nieuwe probleemstelling ontdekt. Twinsense heeft veel klanten in de bouw en techniek. Voor deze bedrijven ontwikkelen zij AR-applicaties voor demonstratie en promotionele doeleinden.

De bouwbedrijven werken veelal met modelleer-programma's. Elk bouwkundig modelleerprogramma heeft een andere datastructuur in het exporteren van 3D-modellen.

De aangeleverde 3D-modellen werken niet optimaal in een AR applicatie. Twinsense heeft geen programma of methode dat 3D-modellen exact analyseert en specifiek aan kan geven welke technieken nodig zijn voor optimalisatie. Wel werken de ontwikkelaars van Twinsense met één analyse programma (Unity3D) die aangeeft waar performance problemen zijn. Zie hiervoor figuur 9. De workflow van de ontwikkelaars van Twinsense kan gekarakteriseerd worden door trial-en-error wat veel tijd kost. Dat zouden ze graag geoptimaliseerd zien worden.



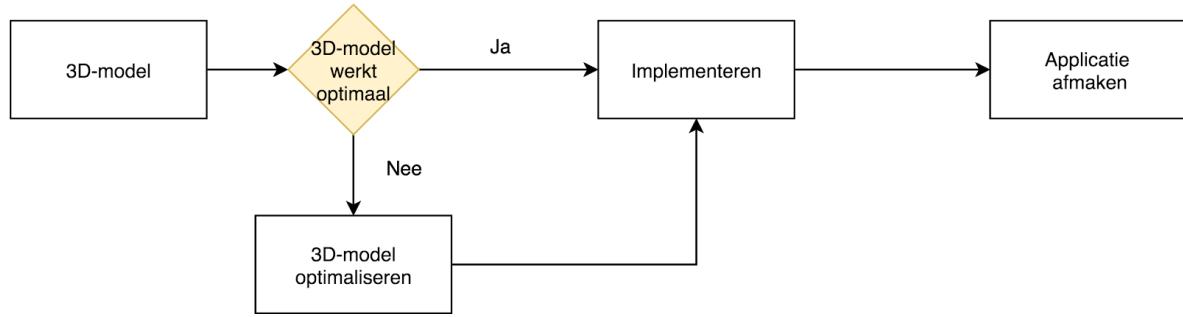
Figuur 9: Workflow voor AR-ontwikkelaars bij Twinsense voor het implementeren van een 3D-model.

5.1. Product voor afstudeeropdracht

Voor mijn afstudeeropdracht is een prototype ontwikkeld dat de 3D-modellen analyseert. Het prototype is bedoeld voor de ontwikkelaars van Twinsense die ondersteuning biedt voor de workflow (Zie bijlage 4: Beroepsproduct).

Het prototype is toegevoegd in Unity3D. De ontwikkelaars van Twinsense werken vanuit één programma. Het prototype analyseert binnen Unity3D de 3D-modellen en controleert de data. Uit de data van het 3D-model wordt gekeken naar oorzaken die performance problemen veroorzaken in AR-applicaties.

Het doel is dat een tussenstap wordt toegevoegd aan de standaard workflow van figuur 9 die het 3D-model analyseert voordat het in de applicatie geïmplementeerd wordt (zie figuur 10).



Figuur 10: Workflow voor AR-ontwikkelaars bij Twinsense voor het implementeren en optimaliseren van CAD-modellen.

5.2. Doelstelling onderzoeksopdracht

Het doel van dit onderzoek is dat de ontwikkelaars een beter beeld zullen krijgen van de 3D-modellen die door klanten van technische en productie sector worden aangeleverd. Zij zullen sneller tot inzicht komen welke methodes en technieken moeten worden toegepast om een model te optimaliseren. Dit zal zowel tijd als budget besparen op den duur.

6. Achtergrondinformatie

Om de kernbegrippen van de probleemstelling die in het vorige hoofdstuk zijn beschreven te beantwoorden is het van belang om de productieprocessen van het ontwikkelen van een AR-applicatie te analyseren.

Bij het ontwikkelen van AR-applicaties voor smartphones, worden de volgende optimalisatietechnieken toegepast in de productie pipeline:

- Geen gebruik van normal maps, reflection maps of emissions maps.
- Unity3D Standard Shader wordt gebruikt.
- De 3D-modellen hebben geen game physics.
- De 3D-modellen hebben geen animatie.

6.1 Werkproces van de 3D-designers en ontwikkelaars

De designer bij Twinsense ontvangen de 3D-model van klanten.

De aangeleverde 3D-modellen worden geconverteerd in een bestandsformaat dat de 3D-designers kunnen gebruiken. De bestandsformaat kan obj, fbx of colleda zijn.

7. Theoretisch kader

In dit hoofdstuk wordt een overzicht gegeven over de werking van de door Twinsense gebruikte software (Unity3D), de type modellen, het renderproces en de verschillende oorzaken van stottering. Aan het eind wordt het theoretische kader van mogelijke oplossingen aangeleverd.

7.1. Type modellen

De 3D-modellen die door de bedrijven van Twinsense worden geleverd zijn zelf ontwikkelde of geconverteerde mesh-modellen. De geconverteerde mesh-modellen zijn NURBS of point cloud modellen.

7.1.1 Mesh-model

De geometrie van de Mesh-modellen bestaat uit de volgende componenten: vertices , lines en faces. Een vertices bevat de volgende informatie: ruimtelijke driedimensionale coördinaten, kleurinformatie, normals en UV coördinaten. De UV beschrijft hoe een tweedimensionale textuur op een driedimensionaal object toegevoegd wordt.

Een edge is een verbinding tussen twee vertices met een elkaar (Blender, z.d.-b).

Een face is een driehoekige connectie tussen drie edges (Thomas, 2018).



Figuur 11: Mesh-model.

7.1.2 NURBS-model

In CAD-applicaties worden voornamelijk NURBS modellen ontwikkeld.

NURBS zijn 3D-modellen die gecreëerd worden uit wiskundige formules om nauwkeurige vormen te creëren. NURBS-modellen bestaan uit Points of Control vertices (CV) (Autodesk, 2017). De point en de CV bepalen de coördinaten van de oppervlakte (Vectorworks, z.d.).

7.1.3 Point-cloud model

Point-cloud zijn gegenereerde modellen die door een reeks foto's wordt gegenereerd.

De foto's worden rondom het voorwerp of landschap gefotografeerd. Met behulp van externe software worden de afbeeldingen geconstrueerd in een point cloud model (Stachniss, 2015).

7.2. Unity3D

De ontwikkelaars van Twinsense gebruiken Unity3D, omdat Unity3D de mogelijkheid geeft om op verschillende platformen applicaties te ontwikkelen. Met een externe APK van Vuforia kan Unity3D gebruik maken van AR-functionaliteiten (Unity Technologie, zd.-d).

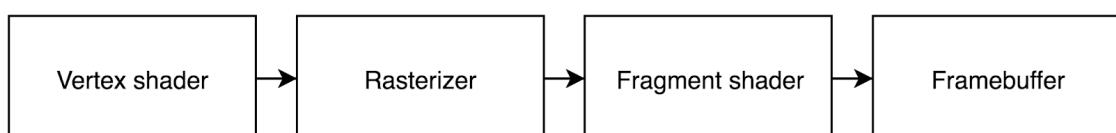
Een 3D-model (of een gameObject) in Unity3D heeft twee componenten nodig om gerenderd te worden in de applicatie.

Het eerste component is de meshfilter, dit bevat de geometrie informatie. De mesh informatie bestaat uit vertices wordt verstuurd naar de vertices Shader. Het tweede component is het Mesh Render component. De Mesh Render bevat informatie hoe het 3D-model gerenderd moet worden. Deze verstuurt informatie naar de Fragment Shader om het model in te kleuren.

7.3. Rendering Pipeline

Het proces om een 3D-model binnen Unity3D in een afbeelding op het scherm te transformeren, wordt een Rendering Pipeline genoemd (Unity Technologies, z.d.-b).

OpenGL ES is het framework voor de communicatie van (Android) AR-applicatie naar GPU. De GPU voert het rendering proces uit (Messaoudi, Simon, & Ksentini, 2015). OpenGL ES is een ‘state machine’ die één functie per proces state kan uitvoeren. OpenGL ES heeft vier belangrijke processen om een 3D-model te renderen (Touch, 2016) welke in figuur 12 weergegeven zijn.



Figuur 12: OpenGL ES processen.

De vertices Shader leest de vertices data van het 3D-model en de coördinaten van de camera van Unity3D. De vertices informatie wordt getransformeerd in geometrie. De geometrie wordt in een digitale 3D-ruimte geprojecteerd (Stanford, 2010).

De Rasterizer transformeert de geometrie-informatie in fragmenten. Fragmenten zijn 2D-pixels met ruimtelijke informatie, die opgeslagen wordt. Vanuit het perspectief van de camera wordt de geometrie geregistreerd en opgesplitst in pixels. In elke pixel wordt de afstand tussen de geometrie en de camera opgeslagen (Touch, 2016).

De Fragment Shader verzorgt het inkleuren van de 2D-pixels op het scherm (Khronos, z.d.). De framebuffer faciliteert het opslaan van 2D-pixels op een gereserveerde ruimte van het geheugen van de videokaart. Wanneer het proces afgerond is worden de pixels weggeschreven naar het scherm (O'Conor, 2017).

7.4.1. Draw call

Een draw call is een pakket met opdrachten dat bestemd is voor de state machine van OpenGL ES.

De opdrachten bevatten specificaties van de rendering van een 3D-model. De functie van een draw call is om de toestand van de state machine te veranderen of het verwerken van specifieke informatie op de videokaart (Stanford, 2010).

7.4.2. Batching

De specificatie van de draw call bevat de informatie over een enkel 3D-model in de Unity3D scène met materiaal en textuur-informatie. Wanneer er zich 3D-modellen in de scène bevinden met hetzelfde materiaal en textuur informatie of kleuren informatie, worden de 3D-modellen in één draw call samengevoegd. Combineren van meerdere modellen met dezelfde informatie heet ‘batching’(Simonov, 2017).

7.4.3. SetPass call

Voor 3D-modellen die niet dezelfde geometrie, materialen, texturen of kleuren hebben, wordt een SetPass call uitgevoerd. Een SetPass call is een opdracht om de instellingen aan te passen voor het volgende te renderen 3D model. Bij een SetPass call wordt een extra opdracht verstuurd naar de state machine (Stanford, 2010).

7.5. Oorzaken van performance problemen

In de handleiding pagina van Unity3D worden verschillende oorzaken van performance problemen gepresenteerd die in de volgende paragrafen worden uitgelegd.

7.5.1. Hoog aantal batch calls

Een hoog aantal batch calls kan worden veroorzaakt doordat een 3D-model uit meerdere sub-3D-modellen bestaat. Elke (sub)3D-model heeft unieke eigenschappen die apart gerenderd moet worden (Unity Technologies, zd -a).

7.5.2. Hoog aantal SetPass calls

Een probleem voor de GPU kan zijn dat er teveel SetPass calls worden verstuurd. Een 3D-model kan meerdere (verschillende) materialen hebben. Denk hierbij bijvoorbeeld aan een model van een auto, waar de wielen en de carrosserie andere texturen hebben. Of een 3D-model kan uit meerdere sub-3D-modellen bestaan welke ieder ten minste een andere materiaal bevat. De materialen kunnen kleur of een textuur hebben. De materiaal informatie wordt geprojecteerd op het 3D-model (Simonov, 2017).

7.5.3. Een hoog aantal vertices

Vertices shader verwerkt aangeleverde vertices data in geometry informatie. Bij een hoog aantal vertices duurt de verwerking langer om een 3D-modellen te ontwikkelen(Unity Technologies, zd -a).

7.5.4. Textuur Formaat

Textuur data is de afbeelding die op het 3D-model wordt geprojecteerd. De textuur data bestaat uit tweedimensionale kleuren data. Ze worden verstuurd via het RAM geheugen. Hoe groter de afbeeldingsresolutie hoe groter het textuur formaat (Unity Technologies, zd -a).

7.6. Oplossing voor de performance problemen

In de vorige paragraaf zijn mogelijk oorzaken beschreven van performance problemen. In deze paragraaf worden de oplossingen beschreven van de specifieke probleemstelling.

De oplossingen zijn tot stand gekomen uit het bestuderen van artikelen die de performance probleemstelling beschrijven. Door de achterliggende oorzaken te analyseren van de performance problemen zijn er twee werkwijzen tot stand gekomen om een oplossing te vinden.

De eerste werkwijze is om naar artikelen te zoeken die een oplossingen voor de performance problemen beschrijven. De tweede werkwijze is het experimenteren met zelf ontwikkelde technieken die de oorzaak van de performance problemen oplossen.

7.6.1. Oplossing voor het hoog aantal batch calls

Om het hoog aantal batch calls te verminderen zijn er twee manieren: Dynamic Batching en verschillende 3D-modellen samenvoegen.

Dynamic batching zorgt dat alle geometrie in een keer verstuurd wordt naar de gpu.

De 3D-modellen moet aan de volgende voorwaarden voldoen(Unity3D, zd -a):

- Geen negatieve schalling van het 3D-model.
- De 3D-model moet gekoppeld zijn aan dezelfde material.

Voor het samenvoegen van verschillende 3D-modellen, kan worden uitgevoerd in 3D-modelleer applicatie(All3DP,2018).

7.6.2. Oplossing voor het hoog aantal SetPass call

Elk 3D-model moet hetzelfde materiaal hebben met dezelfde kleurinformatie of textuur.

Met de textuur kan een textuur altas worden gebruikt.(Fsdevconf, 2013).

7.6.3. Oplossing voor de hoog aantal vertices

Het verwijderen van de geometrie van het 3D-model die niet gezien wordt of geen toegevoegde waarde is (Daukintis, 2017).

Het reduceren van de vertices met behoud van de geometrie of het reduceren van het aantal vertices met kans op vervorming van de geometrie (Blender, z.d.-a).

De geometrie van de 3D-modellen opnieuw modelleren (Retopology) met de werkmethode van low poly modeling (Duffy, 2018).

8. Onderzoeksvragen

In het vorige hoofdstuk zijn de oorzaken van de performance problemen van AR-applicaties beschreven. De volgende variabelen veroorzaken performance problemen.

- Het aantal verschillende materialen toegevoegd aan modellen.
- Het aantal sub 3D-modellen in een 3D-model.
- Het aantal vertices op een 3D-model.
- Het formaat van de textuur.

Om Twinsense advies te geven hoe de probleemstelling opgelost kan worden. Worden de onderwerpen die in het theoretische kader beschreven zijn, gebruikt in de hoofdvraag. De hoofdvraag wordt in de volgende paragraaf beschreven.

8.1. Hoofdvraag

De hoofdvraag wordt als volgt beschreven.

Welke bestaande optimalisatietechnieken kunnen performance problemen verminderen van aangeleverde 3D-modellen voor AR-applicaties?

De hoofdvraag wordt opgedeeld in de volgende deelvragen. In de volgende paragraaf worden de deelvraag beschreven.

8.1.1. Deelvraag 1

Welke variabelen van een 3D-model kunnen leiden tot performance problemen voor AR-applicaties?

8.1.2. Deelvraag 2

Welke bestaande methodes kunnen de performance problemen verbeteren van aangeleverde 3D-modellen?

8.1.3. Deelvraag 3

Wat zijn de visuele consequenties van het toepassen van optimalisatie methodes van 3D-modellen?

9. Methodieken

Onderzocht zijn twee CAD-modellen en twee Point-cloud modellen die geconverteerd zijn in 3D-modellen. De 3D-modellen zijn geleverd door klanten van Twinsense of gedownload van openbare pagina's (zie Bijlage 2: Model informatie).

Deze modellen zijn vooraf geanalyseerd met betrekking tot de fps. Zoals in het theoretisch kader is uitgelegd, correleert fps direct met performance problemen waar de ideale fps waarde tussen de 27 en 30 ligt.

In de tweede stap worden de vier modellen verder geanalyseerd en aangepast om antwoorden op de drie deelvragen op te kunnen leveren.

Getest worden de 3D-modellen op een Samsung S7 met de volgende hardware:

- GPU Mali-T880 MP12.
- Processor Exynos 8890.
- De schermgrootte is 1400 x 2560 pixels.

9.1. Deelvraag 1

Om deelvraag 1 te beantwoorden: ‘*Welke variabelen van een 3D-model kunnen leiden tot performance problemen?*’ wordt een analyse uitgevoerd op de vier 3D-modellen die laat zien hoe hoog de waarden zijn per variabele die mogelijk tot performance problemen kan leiden.

Deze variabelen zijn: (sub)3D-modellen, materialen, textuur formaat, aantal vertices .

9.1.1. Aantal sub-3D-modellen

Of een 3D-model een sub-model bevat en zo ja hoeveel wordt gemeten via de Unity3D functie ‘this.gameObject.GetComponentsInChildren<Meshfilter>()’. Deze functie laat zien of een 3D-model een mesh filter heeft. Het aantal mesh filters vertelt het aantal sub-3D-modellen. Die hiervoor gebruikte code kan teruggevonden worden in bijlage 3.

9.1.2. Aantal materialen

Via de Unity3D componenten RenderFilter worden alle materialen van alle (sub) 3D-modellen geteld.

9.1.3. Unieke materialen

De namen van de materialen komen uit de componenten RenderFilter en worden in list object geplaatst en gefilterd op unieke namen.

9.1.4. Aantal vertices

Via de MeshFilter componenten worden alle vertices waarden bij elkaar geteld.

9.1.5. Aantal texturen

Via de mesh render wordt er gekeken naar het formaat van de textuur die is toegevoegd aan een model. Niet ieder model material hoeft een textuur te hebben.

9.2. Deelvraag 2

Om het antwoord te vinden op deelvraag 2: 'Welke bestaande methodes kunnen de performance problemen verbeteren van aangeleverde 3D-modellen?' is een analyse uitgevoerd op de componenten van één 3D-model. De 3D-modellen zijn de 3D-modellen van deelvraag 1. De 3D-modellen van deelvraag 1 zijn geduplicateerd en vervolgens geoptimaliseerd aan de variabelen die problemen geven.

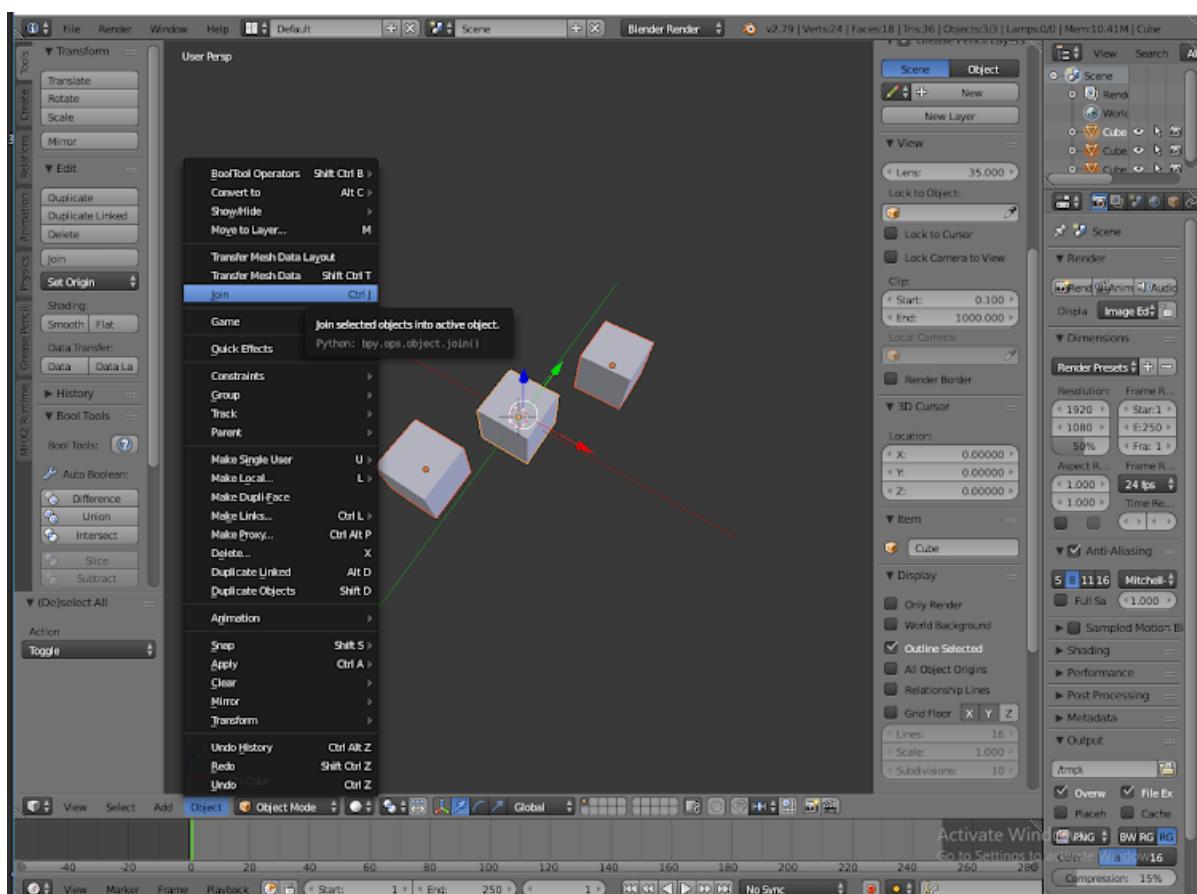
9.2.1. Methodes om het aantal (sub)3D-modellen te minderen

De volgende methodes om de (sub)3D-modellen te verminderen kunnen gebruikt worden:

- Het samenvoegen van (sub)3D-modellen in een 3D-model.
- Het verwijderen van (sub) 3D-modellen.

9.2.1.1. Samenvoegen van (sub) 3D-modellen

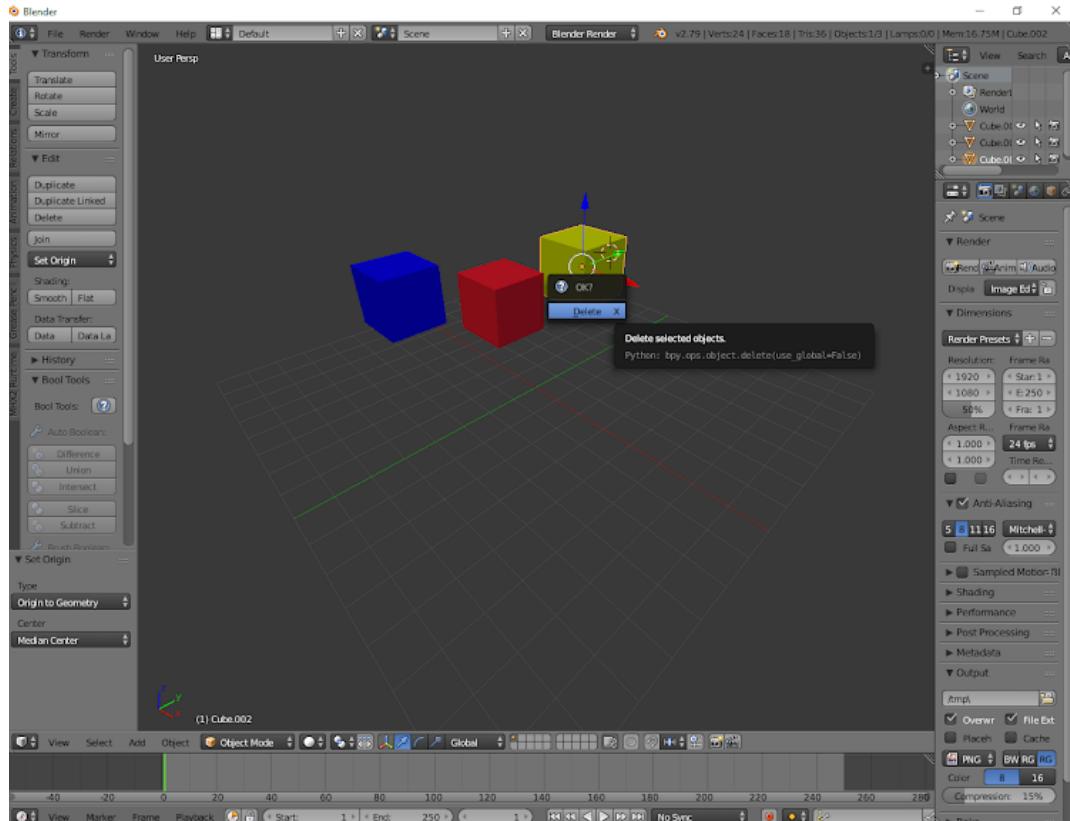
De 3D-modellen worden in Blender geïmporteerd, daarna worden ze geselecteerd. Met de Join functie worden de 3D-modellen samengevoegd (Zie figuur 13).



Figuur 13: Join functie in Blender.

9.2.1.1. Verwijderen van een (sub) 3D-model

Importeer het 3D-model en selecteer het 3D-model. Kies de interactie modus. Selecteer de (sub) 3D-model die verwijderd moeten worden (zie figuur 14).



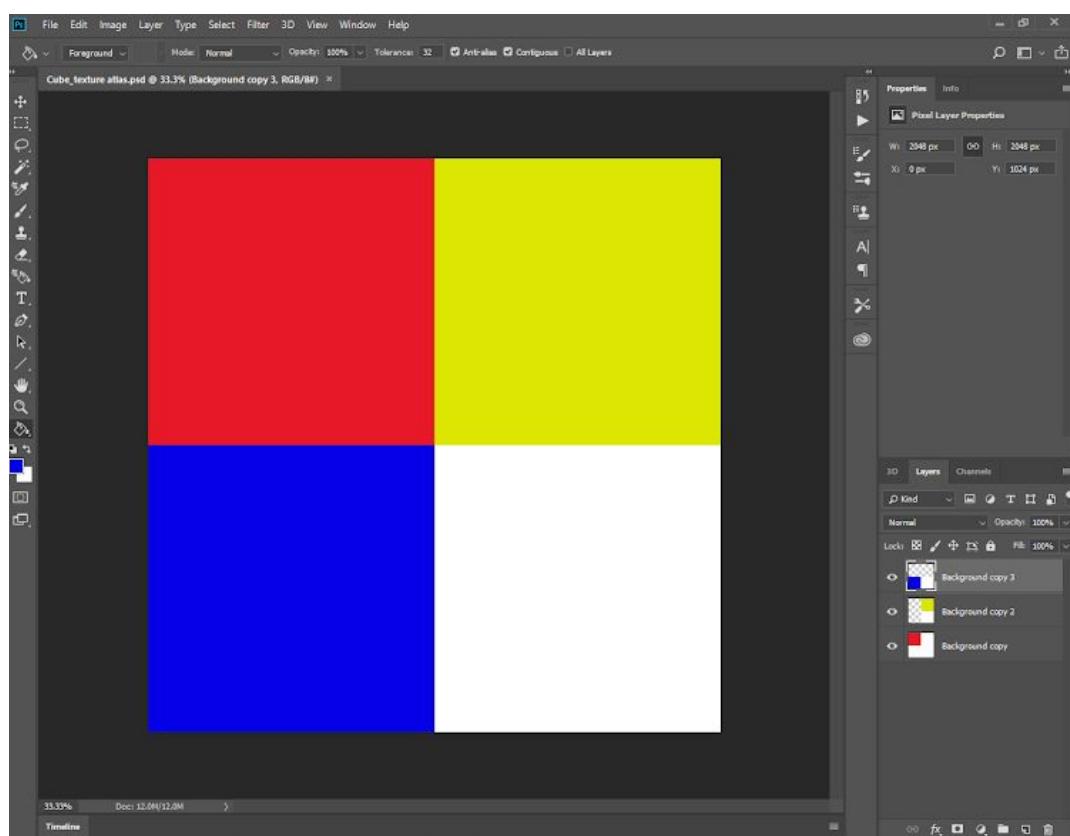
Figuur 14: Verwijderen van een (sub)3D-model

9.2.2. Methode voor verminderen van het aantal materialen

Om het aantal materialen te minderen moet ook het aantal (sub) 3D-modellen verminderd worden. De materialen zijn gekoppeld aan de (sub) 3D-modellen.

9.2.2.1. Het samenvoegen van (sub)3D-modellen en één texture atlas

Importeer de 3D-modellen in Blender. Er wordt een 2048 bij 2048 pixel canvas gecreëerd in Photoshop. Maak van elk materiaal dat zich in Blender bevindt een vierkant in Photoshop. Verklein de formaten van alle vierkanten, zodat alle vierkanten in de canvas passen(zie figuur 16).



Figuur 16: Material kleuren in Photoshop.

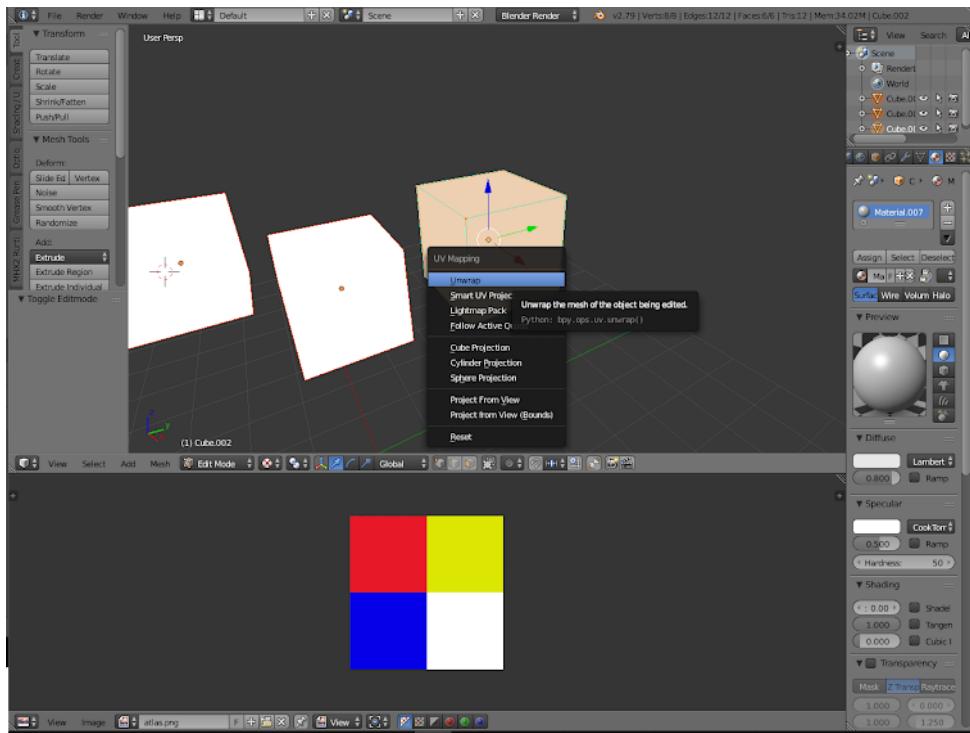
Sla het Photoshop canvas op als een afbeelding.

Importeer de in Photoshop opgeslagen afbeelding in Blender.

Selecteert de geïmporteerde afbeelding als UV-canvas.

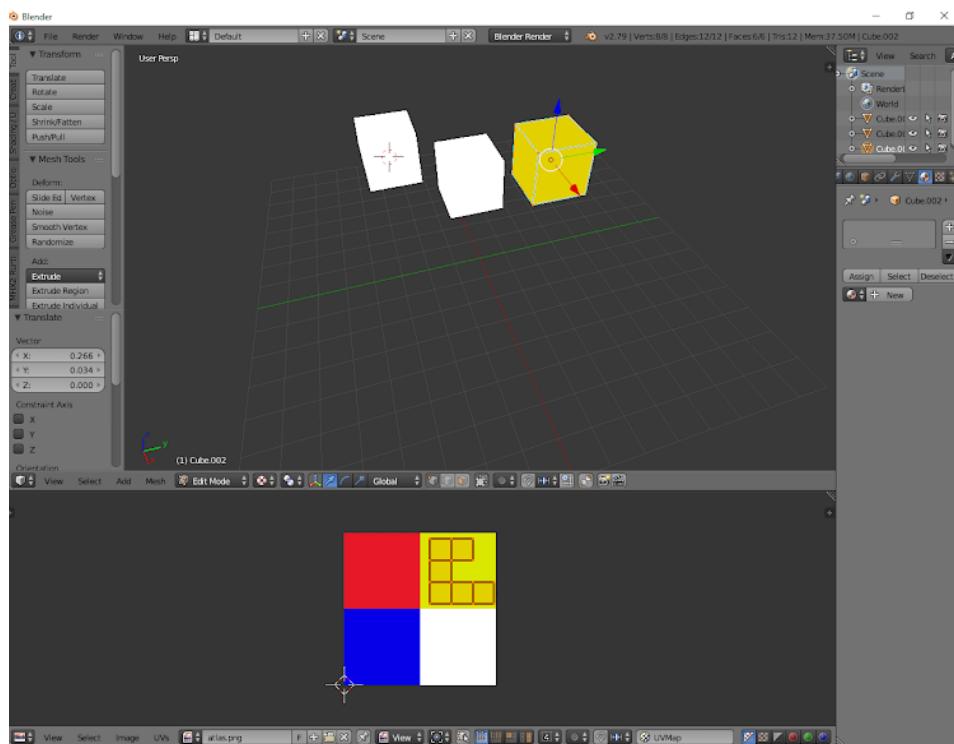
Selecteer het 3D-model en ga naar de edit modus.

Selecteer de geometrie en gebruik de functie Unwrap(zie figuur 17).



Figuur 17: UV-Mapping functie.

Verplaats de UV-map in het vak waar de materiaal kleur bevindt(Zie figuur 18).



Figuur 18: Verplaatsing van de UV-map.

9.2.3 Methode tot verminderen van het aantal vertices

Het verminderen van het aantal vertices en kan op de volgende manieren:

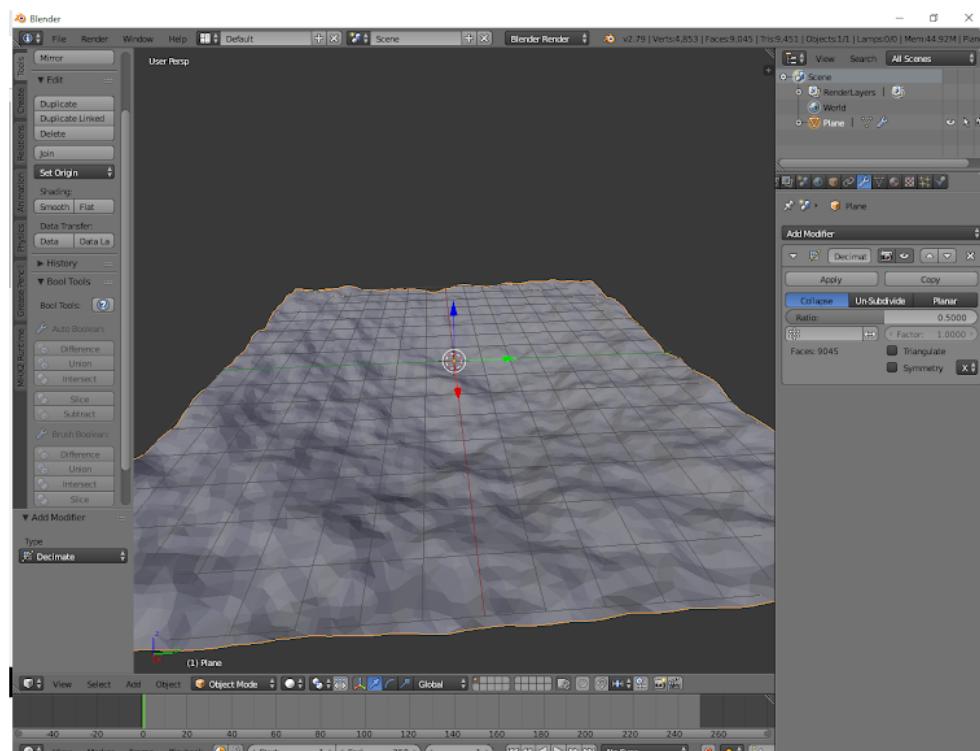
- Het decimeren van het aantal vertices
- Het verwijderen van (sub) 3D-modellen of een aantal vertices .

9.2.3.1. Decimeren van het aantal vertices

Met het uittrekken van het aantal vertices van een 3D-model wordt dit aantal met een percentage gehalveerd. Deze functie behoudt de UV-informatie en de materiaal informatie.

Importeer het 3D-model in Blender. Selecteer de modifier optie in Blender en selecteer de decimate functie(zie figuur 19).

Selecteer een percentage om het aantal vertices van het 3D-model te decimeren.



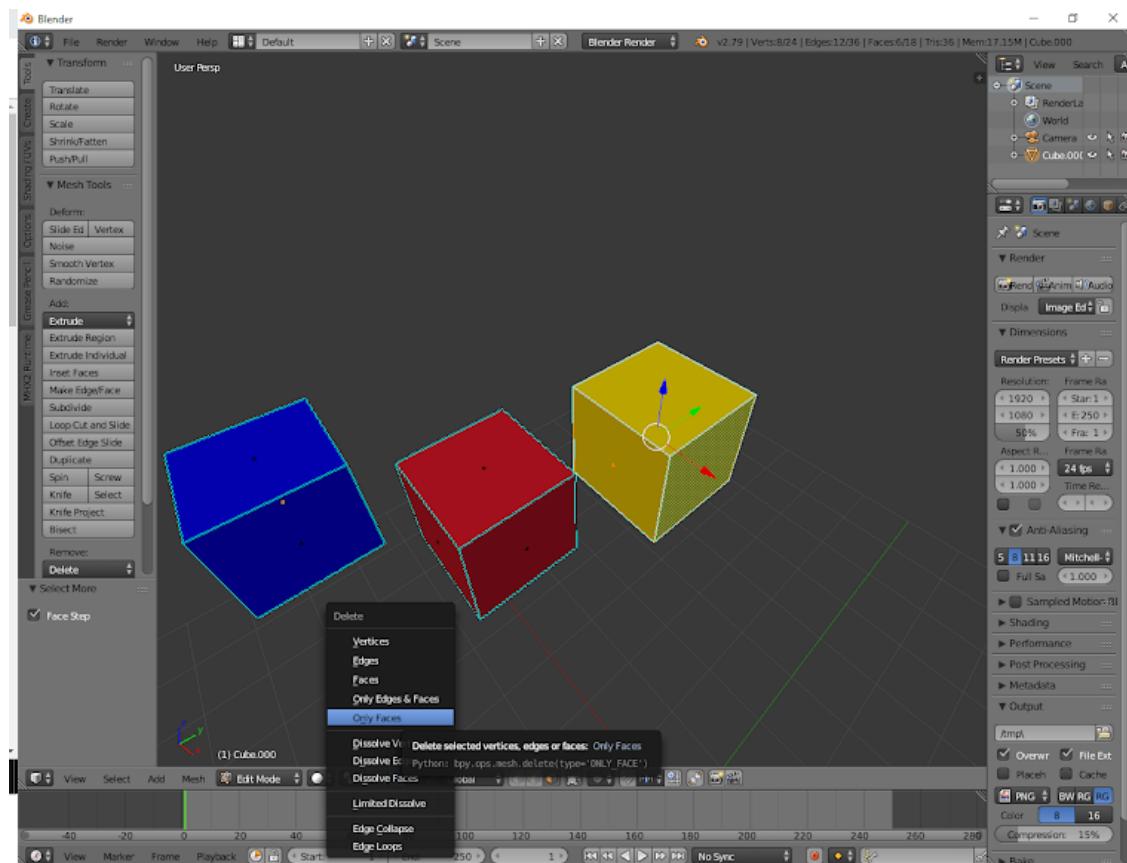
Figuur 19: Decimeren van een aantal vertices .

9.2.3.2. Verwijderen van een (sub)3D-model

Bij het verwijderen van (sub)3D-modellen wordt het totaal aantal vertices vermindert.

Importeer de 3D-model in Blender.

Selecteert de face van de (sub) 3D-model die verwijderd wordt in de edit modus(zie figuur 20).



Figuur 20: Verwijderen van een (sub) 3D-model.

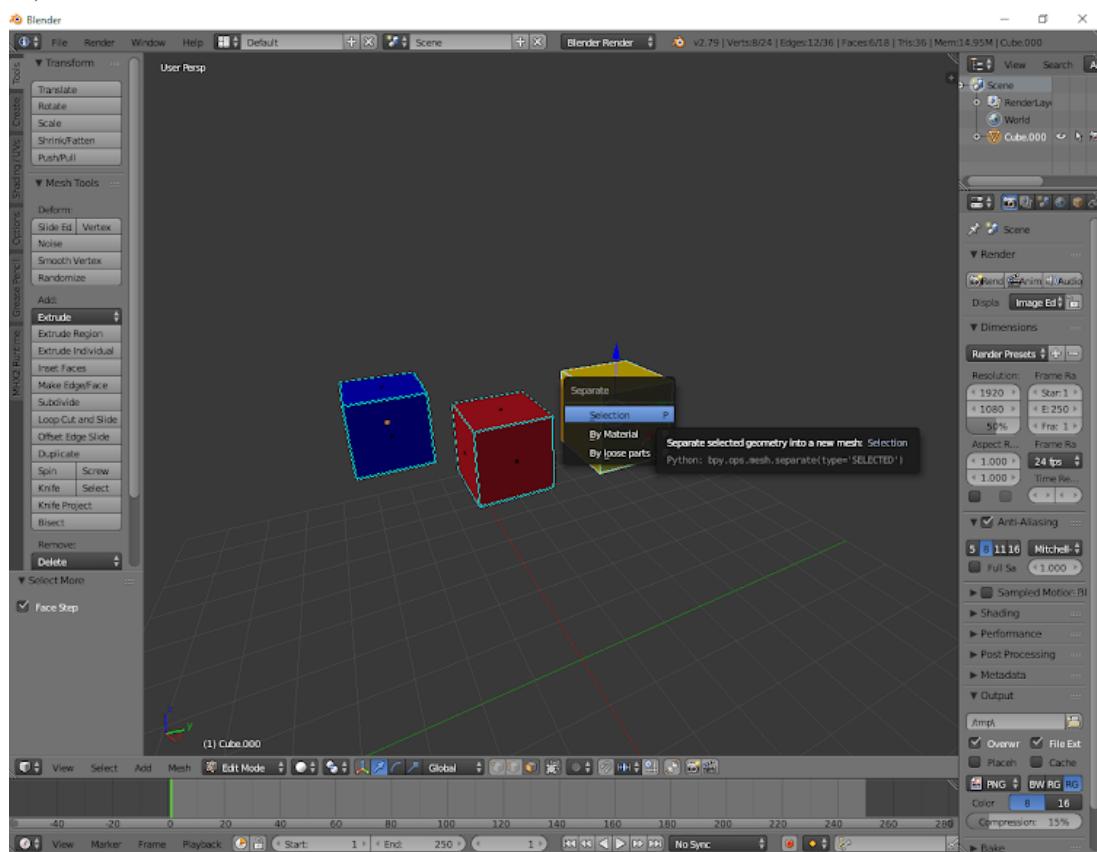
9.2.4. Creëren van (sub) 3D-modellen

De volgende methode kan gebruikt worden om een (sub) 3D-modellen te maken:

- Splitsen van (sub) 3D-model.
- Splitsen geometry.

9.2.4.1. Selecteren van aparte geometry binnen een 3D-model

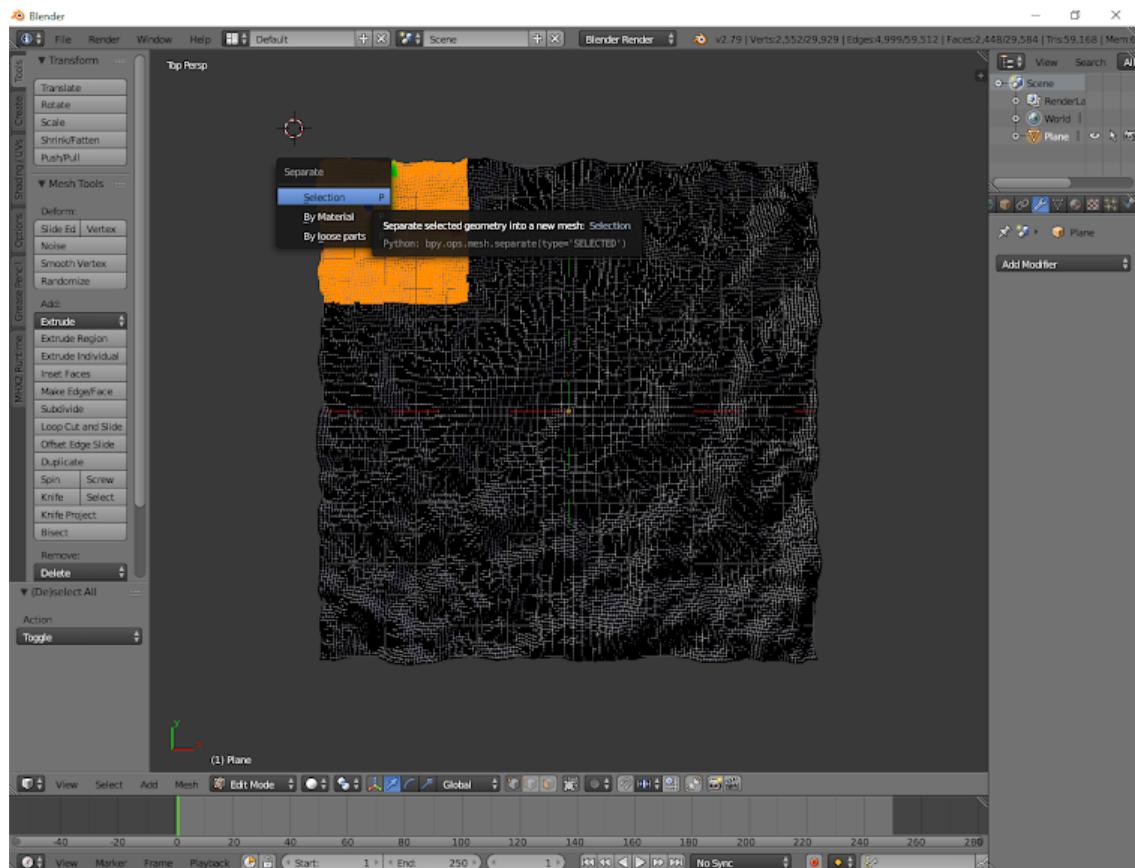
Om het 3D-model op te splitsen in (sub) 3D-modellen worden de geometry geselecteerd die gegroepeerd zijn aan elkaar. Druk op de ‘P’ om de geometry te splitsen. Deze methode werkt alleen op 3D-modellen die meerdere (sub) 3D-modellen hebben en niet met elkaar verbonden zijn (Zie figuur 21).



Figuur 21: Separate functie binnen Blender.

9.2.4.1. Splitsen van een 3D-model

Om een 3D-model te splitsen van geselecteerde faces. Deze techniek werkt snel efficiënt



Figuur 22: Separate functie binnen Blender.

9.2.5. Verkleinen van de textuur resolutie

De volgende twee methodes kunnen gebruikt worden om de textuur resolutie aan te passen.

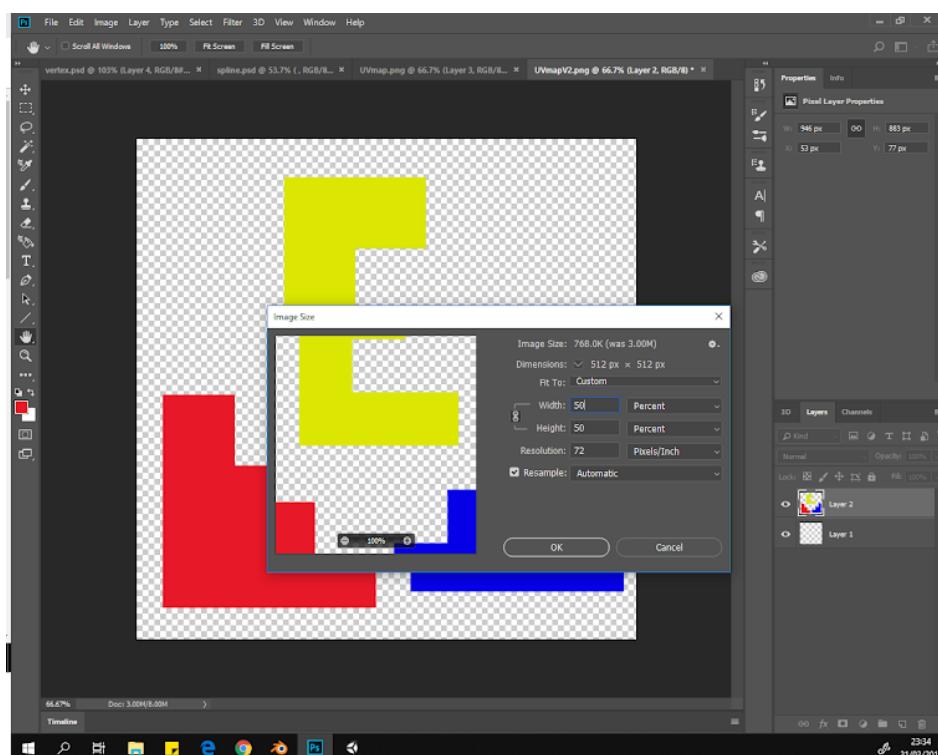
- De textuur resolutie kan verkleint worden in Photoshop.
- Quality setting binnen Unity3D.

9.2.5.1. Photoshop

In deze methode wordt de resolutie van de afbeeldings aangepast.

Selecteer de afbeelding in Photoshop.

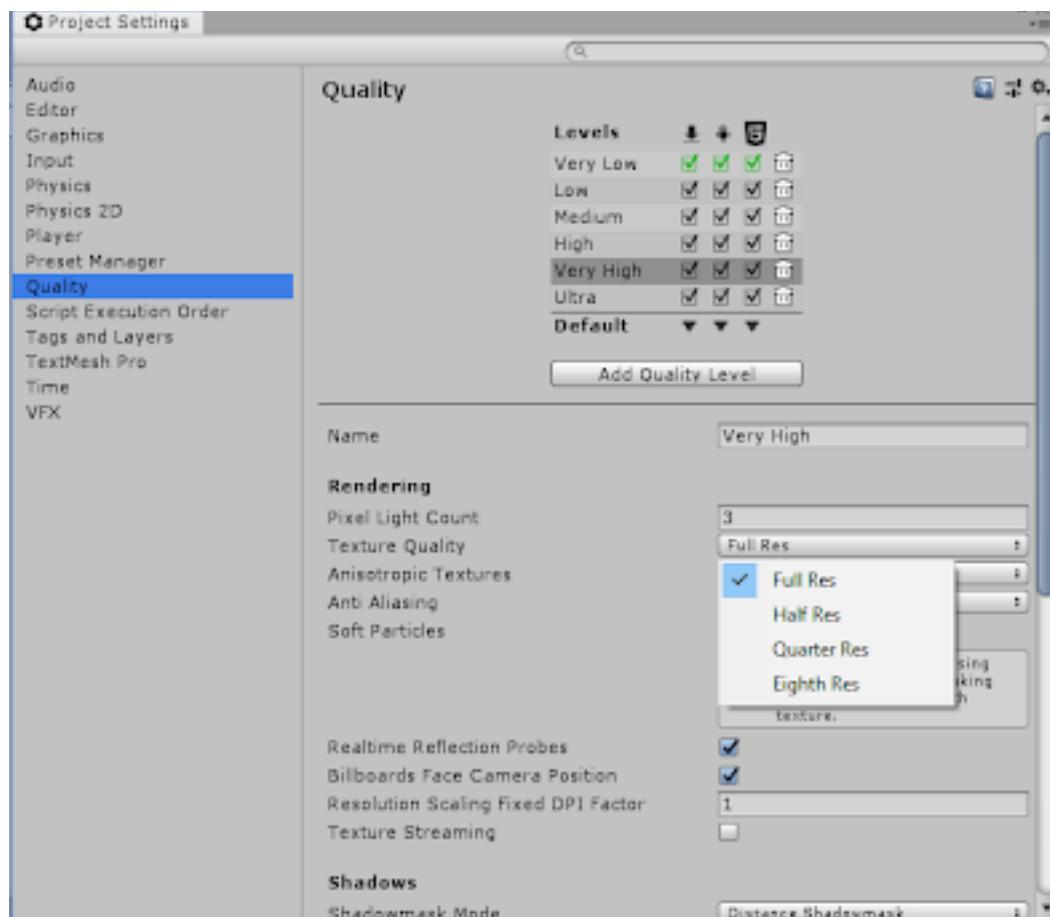
Ga naar image en verklein de resolutie met een percentage.



Figuur 23: Separate functie binnen Blender.

9.2.5.2. Quality Setting aanpassen

Met deze methode wordt in Unity3D de textuur resolutie in de Quality Setting aangepast(Zie figuur 24).



Figuur 24: Quality Setting in Unity3D.

9.3. Deelvraag 3

Deelvraag 3 is *'Wat zijn de visuele consequenties van het toepassen van optimalisatie methodes van 3D-modellen?'*. Zijn er ongewenste consequenties met betrekking tot de kwaliteit van een 3D-model na het toepassen van de optimalisatie technieken? De kwaliteit van een 3D-model wordt naast de performance gedefinieerd als kleurovereenkomst, textuur resolutie en geometry resolutie.

9.3.1. Verschil in kleur

Om het kleurverschil te zien, worden screenshots van de applicatie naast elkaar geplaatst.



Figuur 25: Vergelijking in kleur.

9.3.2. Zichtbaarheid van (sub) 3D-modellen

Zijn alle (sub) 3D-modellen zichtbaar die in het originele 3D-model ook zichtbaar zijn.



Figuur 27: Zichtbaarheid van de 3D-modellen..

9.3.3. Verschil in textuur resolutie

Als een 3D-model een textuur heeft en de resolutie wordt aangepast, dan wordt er gekeken naar het verschil met het originele textuur.



Figuur 28: Zichtbaarheid van de 3D-modellen..

9.3.4. Verschil in geometrie resolutie

Als het aantal vertices van een 3D-model verminderd, wordt er gekeken naar resolutie van de geometry.



Figuur 28: Zichtbaarheid van de 3D-modellen..

10. Resultaten

Het doorvoeren van de verschillende analyses heeft relevante resultaten opgeleverd. Deze zijn vervolgens per deelvraag opgedeeld. In bijlage 4 bevinden de afbeeldingen van de resultaten.

10.1. Resultaten van deelvraag 1

‘Welke variabelen van een 3D-model kunnen leiden tot performance problemen?’

Naam 3D-model	Aantal(sub) 3D-modellen	Aantal materialen	Aantal unieke materialen	Aantal vertices	Aantal afbeeldingen	Fps
3D-Model_1	1.429	1.537	30	209.136	0	25
3D-Model_2	12	12	12	1.014.652	12	18
3D-Model_3	136	136	34	1.236.448	0	25
3D-Model_4	1	1	1	1.404.863	1	18

Tabel 1: Eigenschappen originele 3D-modellen.

10.2. Resultaten van deelvraag 2

In deze paragraaf wordt *deelvraag 2: Welke bestaande methodes kunnen de performance problemen verbeteren van aangeleverde 3D-modellen?* beantwoord.

Naam 3D-model	Optimalisatie techniek	Aantal(sub) 3D-modellen	Aantal materialen	Aantal unieke materialen	Aantal vertices	Aantal afbeeldingen	Fps
3D-Model_1_Atlas	Texture atlas	1	1	1	209.136	1	28
3D-Model_1_Del	Verwijderen van (sub) 3D-modellen	737	940	31	58.475	0	27

Tabel 2: Optimalisatie van 3D-Model_1

Naam 3D-model	Optimalisatie techniek	Aantal(sub) 3D-modellen	Aantal materialen	Aantal unieke materialen	Aantal vertices	Aantal afbeeldingen	Fps
3D-Model_21_Dec_50	Verminderen van vertices (50%)	12	12	12	523.337	12	27
3D-model_2_Del	Verwijderen van (sub) 3D-modellen	3	3	3	537.249	3	27

Tabel 3: Optimalisatie van 3D-Model_2

Naam 3D-model	Optimalisatie techniek	Aantal(sub) 3D-modellen	Aantal materialen	Aantal unieke materialen	Aantal vertices	Aantal afbeeldingen	Fps
3D-Model_3_Del	Verwijderen van (sub) 3D-modellen	44	44	11	339.868	0	29

Tabel 4: Optimalisatie van 3D-Model_3

Naam 3D-model	Optimalisatie techniek	Aantal(sub) 3D-modellen	Aantal materialen	Aantal unieke materialen	Aantal vertices	Aantal afbeeldingen	Fps
3D-Model_4_Dec_25	Verminderen van vertices (25%)	1	1	1	338.657	1	29
3D-Model_en_4_Dec_Div	Verminderen van vertices (50%) en verdelen	9	9	1	694.680	1	29

Tabel 5: Optimalisatie van 3D-Model_4

10.3. Resultaten van deelvraag 3

Antwoord op deelvraag 3: ‘Wat zijn de visuele consequenties van het toepassen van optimalisatie methodes van 3D-modellen?’

10.3.1. Resultaat van Model_1

Naam 3D-model	Gelijknis in kleur	Zijn alle (sub) 3D-modellen zichtbaar	Is er verschil in textuur resolutie	Is er verschil in geometrie resolutie
3D-model_1_Atlas	Nee	Ja	Nvt	Nvt
3D-model_1_Del	Ja	Nee	Nvt	Nvt

Tabel 6: Vergelijking van model_1

10.3.2. Resultaat van Model_2

Naam 3D-model	Gelijknis in kleur	Zijn alle (sub) 3D-modellen zichtbaar	Is er verschil in textuur resolutie	Is er verschil in geometrie resolutie
3D-model_2_Del	Ja	Nee	Nvt	Nee
3D-model_2_Dec_50	Ja	Ja	Nvt	Nee

Tabel 7: Vergelijking van model_2

10.3.3. Resultaat van Model_3

Naam 3D-model	Gelijkenis in kleur	Zijn alle (sub) 3D-modellen zichtbaar	Is er verschil in textuur resolutie	Is er verschil in geometrie resolutie
3D-model_3_Del	Nee	Ja	Nvt	Nee

Tabel 8: Vergelijking van model_3

10.3.4. Resultaat van Model_4

Naam 3D-model	Gelijkenis in kleur	Zijn alle (sub) 3D-modellen zichtbaar	Is er verschil in textuur resolutie	Is er verschil in geometrie resolutie
3D-model_4_Dec_25	Ja	Nvt	Nvt	Ja
3D-model_4_Dec_50_Div	Ja	Nvt	Nvt	Nee

Tabel 9: Vergelijking van model_4

11. Conclusie

In dit onderzoek is gezocht naar een antwoord op de hoofdvraag: ‘Welke bestaande optimalisatietechnieken kunnen performance problemen verminderen van aangeleverde 3D-modellen voor AR-applicaties?’.

De hoofdvraag kan alleen beantwoord worden wanneer de volgende deelvragen zijn beantwoord:

- Welke variabelen van een 3D-model kunnen leiden tot performance problemen voor AR-applicaties?
- Welke bestaande methodes kunnen de performance problemen verbeteren van aangeleverde 3D-modellen?
- Wat zijn de visuele consequenties van het toepassen van optimalisatie methode op 3D-modellen?

11.1. Deelvraag 1

Om deelvraag 1 te beantwoorden: ”*Welke variabelen van een 3D-model kunnen leiden tot performance problemen voor AR-applicaties*” .

Uit de vier 3D-modellen die onderzocht werden, zijn er verschillende variabelen die de performance problemen veroorzaken. Voor 3D-model_1 zijn het de variabelen materialen en (sub) 3D-modellen, die leiden tot performance problemen. Daarentegen voor 3D-Model_4 leiden de variabele vertices tot een performance problemen.

Uit de resultaten is duidelijk te zien dat de waarden van de variabele 3D-modellen en materialen zich boven de duizend moeten bevinden en voor de variabele vertices boven een miljoen.

11.2. Deelvraag 2

Om deelvraag 2 te beantwoorden: ”*Welke bestaande methodes kunnen de performance problemen verbeteren van aangeleverde 3D-modellen*?”.

De bestaande optimalisatie methoden kunnen alleen worden toegepast, aan de variabelen die de performance problemen veroorzaken. De volgende variabelen kunnen performance problemen veroorzaken:

- De variabelen: (sub) 3D-modellen en materialen.
- De variabele vertices.
- De variabelen: (sub) 3D-modellen, materialen en vertices.

11.2.1. De variabelen (sub) 3D-modellen en materialen

Voor de optimalisatie van de variabelen (sub) 3D-modellen en materialen, kunnen de volgende bestaande optimalisatie methodes toegepast worden:

- Creëren van textur atlas.
- Verwijderen van een (sub) 3D-model.

Bij 3D-model_1 kunnen bij beide optimalisatie methodes toegepast worden. Bij 3D-model_1_Dec is ook te zien dat de aantal vertices verminderd zijn.

11.2.2. De variabele vertices

Voor de optimalisatie van de vertices, kunnen de volgende bestaande optimalisatie methodes toegepast worden:

- Opsplitsen en decimeren.
- Decimeren.

Voor een 3D-model met alleen een waarden van variabele vertices dat een performance probleem veroorzaakt.

Dan is de optimalisatie methodes en decimeren en opsplitsen.

11.2.3 De variabelen (sub) 3D-modellen, materialen en vertices

Voor de optimalisatie van de vertices, kunnen de volgende bestaande optimalisatie methodes toegepast worden:

- Verwijderen van (sub) 3D-modellen.
- Decimeren en het creëren van een textuur altas.

11.3. Deelvraag 3

Wat zijn de visuele consequenties van het toepassen van optimalisatie methodes van 3D-modellen?

In de vorige resultaten zijn er verschillende optimalisatie methodes per model toegepast.

De volgende optimalisaties zijn toegepast.

- Textuur atlas.
- Verwijderen van (sub) 3D-modellen.
- Decimeren.
- Decimeren en opsplitsen.

11.3.1. Textuur atlas

Bij de het toepassen van de textuur atlas voor de optimalisatie methode is er een kleurverschil te zien bij alle 3D-modellen die textuur atlas gebruiken.

Bij het implementeren van textuur atlas is er geen transparantie meer.

11.3.2. Verwijderen van (sub) 3D-modellen

Bij het verwijderen van de (sub) 3D-modellen hoeven er niet altijd visuele consequentie te zijn.

Bijvoorbeeld 3D-model 1_Del is het zichtbaar dat de (sub) 3D-modellen zijn verwijderd.

Voor de 3D-model_3_Del is dit niet zichtbaar, omdat de (sub) 3D-modellen zich binnen het 3D-model bevinden.

11.3.3. Decimeren

Niet bij elk 3D-model die gedecimeerd wordt, is een geometrie resolutie verschil te zien.

11.3.3. Decimeren en opsplitsen

Bij 3D-model_4_Dec_50_Div dat gedecimeerd wordt, is een geometrie resolutie verschil te zien.

11.3. Hoofdvraag

Welke bestaande optimalisatietechnieken kunnen performance problemen verminderen van aangeleverde 3D-modellen voor AR-applicaties?.

De volgende optimalisatie technieken kunnen de performance problemen verbeteren:

- Textuur atlas.
- Verwijderen van (sub) 3D-modellen.
- Decimeren.
- Opsplitsen en decimeren.

Elke optimalisatietechniek is gespecialiseerd in het oplossen van een of meerdere variabelen die een performance probleem veroorzaken. Alle optimalisatietechnieken kunnen een visuele consequenties met zich mee brengen. De optimalisatie techniek met het verwijderen van (sub) 3D-modellen hoeft niet in bijzondere gevallen een visuele consequentie met zich mee te brengen.

12. Discussie

Door een beperking van hardware en tijd konden de volgende punten niet onderzocht worden:

- Occlusion curling
- Optimalisatie applicaties.

12.1. Occlusion curling

Bij lezen van de variabelen van de 3D-model wordt de informatie rechtstreeks gelezen. Bij het inzoomen van een 3D-model wordt er maar een gedeelte gerenderd. Door een functie binnen Unity3D die oculus culling heet. Occlusion culling zorgt er voor dat er alleen gedeelten gerenderd worden die door de camera gezien worden.

12.2. De applicatie voor optimalisatie

De optimalisatietechnieken zijn technieken die gebruikt wordt in Blender en Photoshop. Optimalisatie technieken in de 3D-model Blender kunnen anders werken dan een andere 3D-applicatie.

De handelingen van de optimalisatie technieken werken anders en kunnen een andere resultaat geven. Bijvoorbeeld in het decimeren van het aantal vertices en.

13. Aanbevelingen

Voor de designers en ontwikkelaars van Twinsense wordt het volgende aanbevolen bij het optimaliseren van aangeleverde 3D-modellen.

Bespreken wat de voorwaardes zijn hoe het 3D-model gepresenteerd moet worden in AR. De voorwaardes van de aangeleverde 3D-modellen zijn voortgekomen uit de visuele consequentie van de optimalisatietechniek.

De voorwaardes kunnen het volgende zijn:

- Moet de kleuren exact overeenkomen met het 3D-model
- Moet de geometry resolutie van de 3D-model exact overeenkomen met de aangeleverde 3D-model.
- Moet alle (sub) 3D-modellen overeenkomen met het aangeleverde model.

Daarna is het belangrijk voor de ontwikkelaars om uit te zoeken welke variabelen de oorzaak zijn van de performance problemen. De variabelen worden weergegeven in het prototype dat ontwikkeld is voor dit onderzoek.

De variabelen kunnen de volgende probleemstelling zijn:

- Aantal vertices van een enkel 3D-model.
- Aantal (sub) 3D-modellen en het aantal materialen.
- Aantal (sub) 3D-modellen, het aantal materialen en het aantal vertices .

13.1. Optimalisatie voor een hoog aantal vertices

Aanbevolen wordt om het 3D-model om de percentage van de vertices te verdunnen.

Na het verdunnen van het 3D-model wordt het aangeraden om een 3D-model op splitten in (sub) 3D-modellen.

Bij deze optimalisatie methode kan het voorkomen dat de resolutie niet overeenkomt met de aangeleverde 3D-model.

13.2. Optimalisatie voor een aantal 3D-modellen en materialen

Het wordt aanbevolen bij een aantal (sub) 3D-modellen samen te voegen en textuur atlas te maken van de kleuren van de materialen.

Een tweede methode is het aantal (sub) 3D-modellen te verwijderen. Deze methode kan voor zorgen dat er minder modellen in de scene zijn.

13.3. Optimalisatie voor een aantal 3D-modellen, materialen en vertices

Het wordt aanbevolen bij een aantal (sub) 3D-modellen samen te voegen en textuur atlas te maken van de kleuren van de materialen. Daarna het 3D-model decimeren.

Een tweede methode is het aantal (sub) 3D-modellen te verwijderen. Van de overgebleven 3D-modellen samenvoegen en een textuur atlas maken.

Bronnenlijst

[De vormgeving van DnD] [Foto]. (2018, 8 december). Geraadpleegd op 20 februari 2019, van <https://www.want.nl/dungeons-and-dragons-spelen/>

3D Hubs. (z.d.). 3D Modeling CAD Software. Geraadpleegd op 14 maart 2019, van <https://www.3dhubs.com/knowledge-base/3d-modeling-cad-software>

All3DP. (2018, 14 december). Blender: How to Merge Objects – Simply Explained | All3DP. Geraadpleegd op 9 maart 2019, van <https://all3dp.com/2/blender-how-to-merge-objects/>

Autodesk. (2017, 15 juni). NURBS Models: Objects and Sub-Objects. Geraadpleegd op 4 april 2019, van <https://knowledge.autodesk.com/support/3ds-max/learn-explore/caas/CloudHelp/cloudhelp/2017/ENU/3DSMax/files/GUID-7F3EB6C9-AF07-4FFC-B570-3D1AB9852AA5-htm.html>

Blender. (z.d.-a). Decimate Modifier — Blender Manual. Geraadpleegd op 9 maart 2019, van <https://docs.blender.org/manual/en/latest/modeling/modifiers/generate/decimate.html>

Blender. (z.d.-b). Structure — Blender Manual. Geraadpleegd op 15 maart 2019, van <https://docs.blender.org/manual/en/latest/modeling/meshes/structure.html>

Dreammakersgroup. (z.d.). Log Viewer - Asset Store. Geraadpleegd op 16 februari 2019, van <https://assetstore.unity.com/packages/tools/log-viewer-12047>

Duffy, B. (2018, 9 juli). Retopologizing VR and Game Assets Using Blender*. Geraadpleegd op 14 maart 2019, van <https://software.intel.com/en-us/articles/retopologizing-vr-and-game-assets-using-blender>

Fsdevconf. (2013, 17 november). FSDevConf: What's a drawcall? [Video]. Geraadpleegd op 9 oktober 2018, van <https://www.youtube.com/watch?v=CdWYdc9SVIc>

Horwitz, J. (2018, 23 oktober). SuperData: Oculus Quest will mainstream VR in 2019, but AR will lead by 2021. Geraadpleegd op 12 december 2018, van <https://venturebeat.com/2018/10/19/superdata-oculus-quest-will-mainstream-vr-in-2019-but-ar-will-lead-by-2021/>

Khronos. (z.d.). vertices Shader - OpenGL Wiki. Geraadpleegd op 20 november 2018, van https://www.khronos.org/opengl/wiki/vertices_Shader

Messaoudi, F., Simon, G., & Ksentini, A. (2015). Dissecting games engines: The case of Unity3D. 2015 International Workshop on Network and Systems Support for Games (NetGames), .
<https://doi.org/10.1109/netgames.2015.7382990>

O'Conor, K. (2017, 3 maart). GPU Performance for Game Artists. Geraadpleegd op 6 december 2018, van <http://fragmentbuffer.com/gpu-performance-for-game-artists/>

Stanford. (2010, 27 september). 19. OpenGL ES. Geraadpleegd op 12 november 2018, van https://www.youtube.com/watch?v=_WcMe4Yj0NM

Simonov, V. (2017, 3 april). How to see why your draw calls are not batched in 5.6 – Unity Blog. Geraadpleegd op 14 januari 2019, van <https://blogs.unity3d.com/2017/04/03/how-to-see-why-your-draw-calls-are-not-batched-in-5-6/>

SPACE10. (2018, 12 mei). Assembling IKEA's new AR app, without a manual [Foto]. Geraadpleegd op 1 april 2019, van <https://medium.com/space10/assembling-ikeas-new-ar-app-without-a-manual-c74c09d0488d>

Stachniss, C. (2015, 9 juli). Photogrammetry I - 01 - Introduction (2015) [Video]. Geraadpleegd op 16 maart 2019, van https://www.youtube.com/watch?v=_mOG_lpPnpY

SuperData Research. (z.d.). SuperData Research | Games data and market research » Virtual Reality Market and Consumers. Geraadpleegd op 13 december 2018, van <https://www.superdataresearch.com/market-data/virtual-reality-industry-report/>

Thomas, A. (2018, 30 maart). CAD vs. Modeling: Which 3D Software to Choose? Geraadpleegd op 14 maart 2019, van <https://www.shapeways.com/blog/archives/27653-cad-vs-modeling-which-3d-software-to-choose.html>

Touch, A. (2016, 28 juni). Unite Europe 2016 - A Crash Course to Writing Custom Unity Shaders! [Video]. Geraadpleegd op 12 februari 2019, van <https://www.youtube.com/watch?v=3penhrrKCYg>

Unity Technologies. (z.d.-a). Unity - Manual: Optimizing graphics performance. Geraadpleegd op september 26, 2018, van <https://docs.unity3d.com/Manual/OptimizingGraphicsPerformance.html>

Unity Technologies. (z.d.-b). Optimizing graphics rendering in Unity games - Unity. Geraadpleegd op 2 december 2018, van <https://unity3d.com/learn/tutorials/temas/performance-optimization/optimizing-graphics-rendering-unity-games?playlist=44069>

Unity Technologies. (z.d.-c.). Unity - Manual: Draw call batching. Geraadpleegd op 20 februari 2019, van <https://docs.unity3d.com/Manual/DrawCallBatching.html>

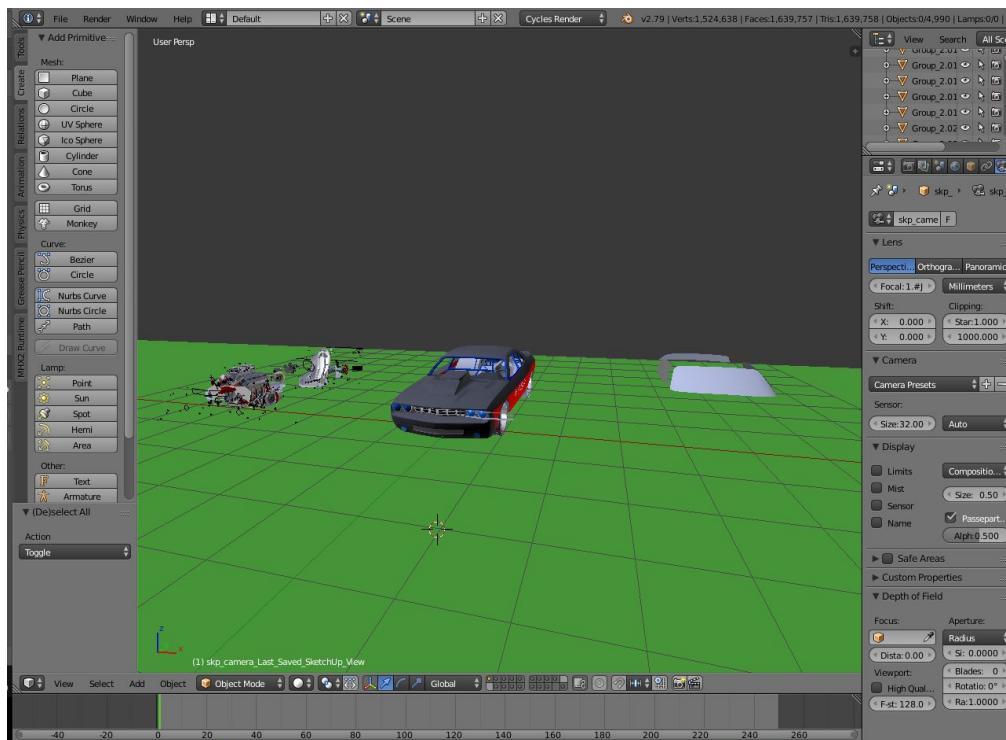
Unity Technologies. (z.d.-d). Partners - Vuforia - Unity. Geraadpleegd op 31 maart 2019, van <https://unity3d.com/partners/vuforia>

Vectorworks. (z.d.). Creating NURBS Curves. Geraadpleegd op 4 april 2019, van http://app-help.vectorworks.net/2018/eng/VW2018_Guide/Shapes2/Creating_NURBS_Curves.htm

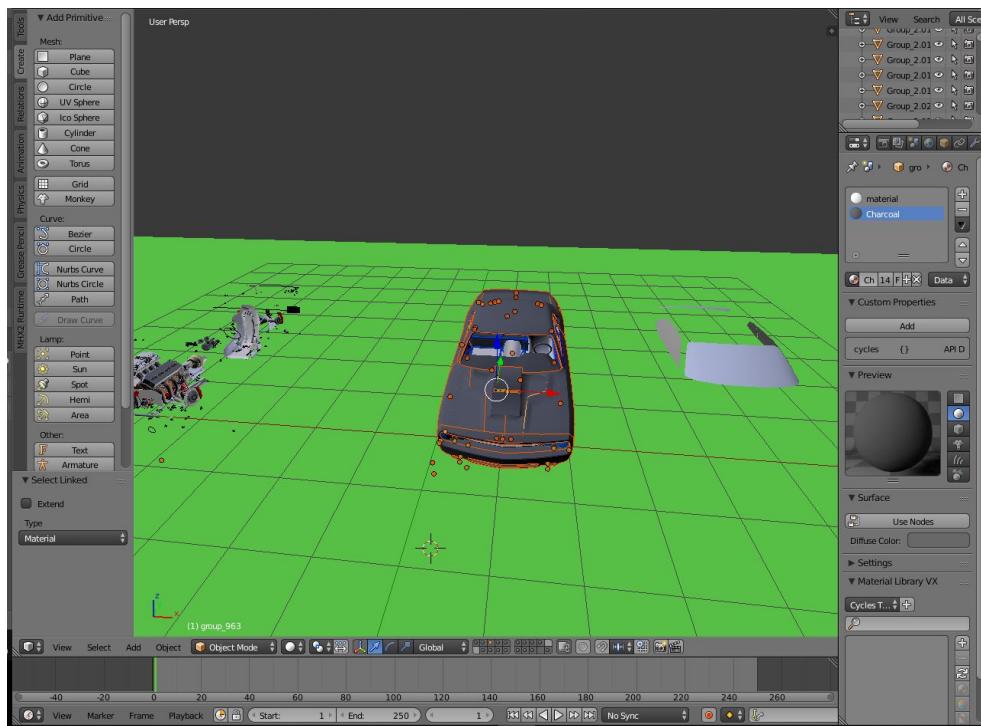
Bijlage 1. Optimalisatie SketchUp 3D-model

De zijn de volgende optimalisatie stappen:

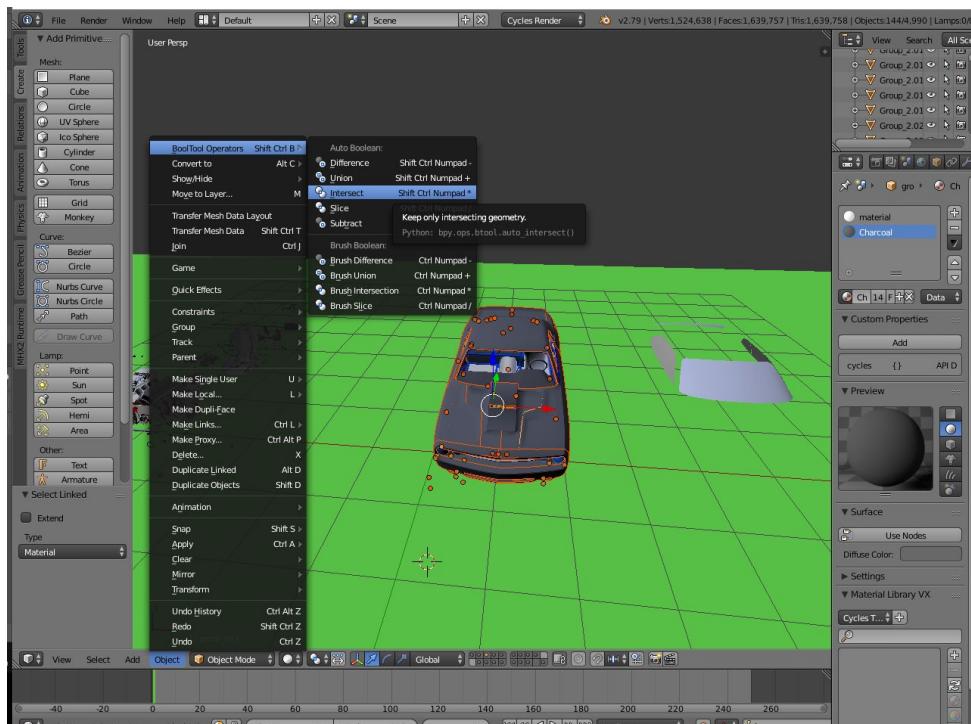
- De sub 3D-modellen wordt opgesplitst in zichtbaar en onzichtbaar geometry.
- De onzichtbare geometry wordt verwijderd(figuur B1.1).
- Geometry met dezelfde materialen kleurwaarden worden samengevoegd(figuur B1.1 tot B1.3)



Figuur B1.1 :Verwijderen van onzichtbare 3D-modellen.



Figuur B1.2: Selecteren de 3D-modellen met dezelfde materiaal.

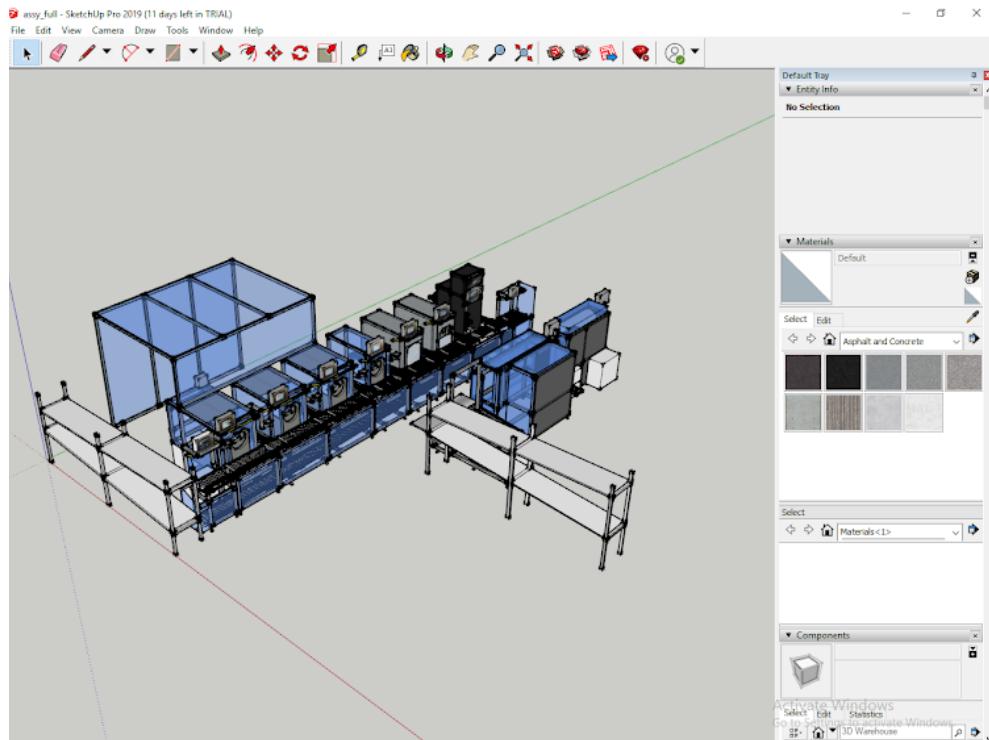


Figuur B1.3: Samenvoegen van 3D-modellen met dezelfde materiaal.

Bijlage 2. Model informatie

2.1. 3D-Model_1

Dit is een CAD-model uit de applicatie SketchUP(zie figuur B2.1).



Figuur B2.1:3D-Model _1

Link:

<https://3dwarehouse.sketchup.com/model/4a796316-2ffb-4d1e-a551-0808a0f67008/Production-line>

2.2. 3D-Model_2

Dit is een gescand 3D-Model (zie figuur B1.2).



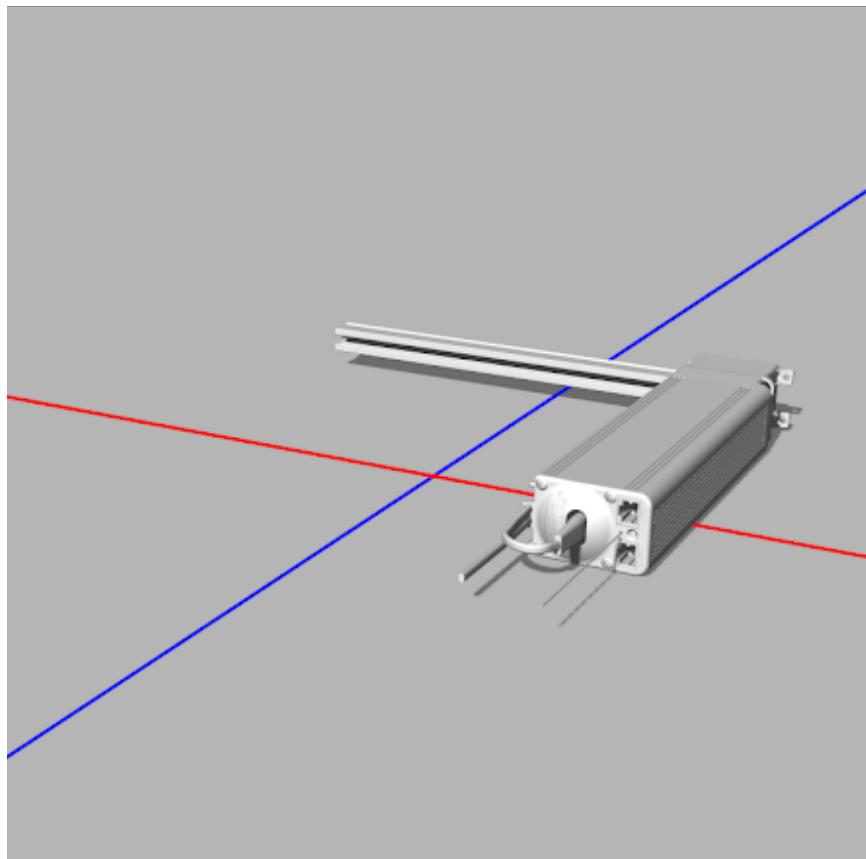
Figuur B2.2:3D-Model_2

Link:<https://sketchfab.com/3d-models/enisala-fortress-romania-afa25d862c734383aacb2774a5f5d927>

2.3. 3D-Model_3

Dit is een CAD-model(zie figuur B2.3) dat is aangeleverd door een klant van Twinsense.
De voorwaarde van hoe het 3D-Model in AR gepresenteerd moet worden zijn als volgt:

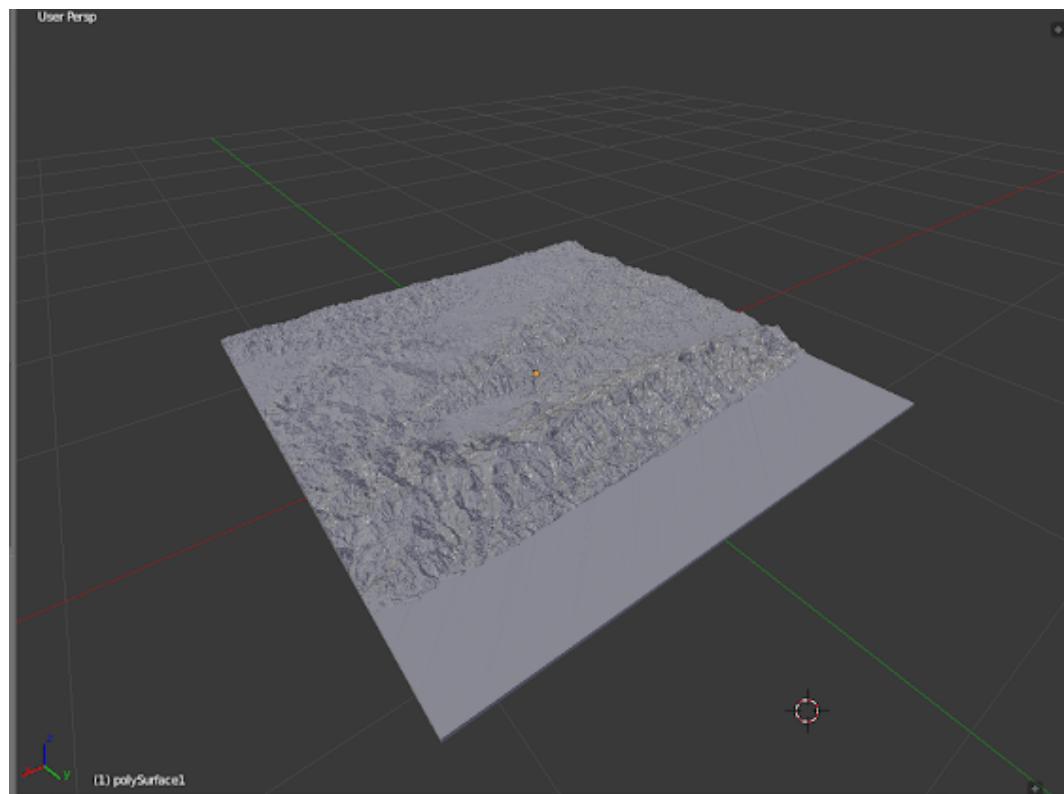
- De 3D-modellen moeten vier maal in scène bevinden.
- De 3D-modellen moeten individueel van elkaar bewegen.



Figuur B2.3:3D-Model_3

2.4. 3D-Model_4

Dit was een point-cloud model dat converteerde naar een mesh-model (zie figuur B1.2).
Dit 3D-Model is aangeleverd door een klant van Twinsense.



Figuur B2.3:3D-Model_4

Bijlage 3. Onderzoeksproduct

In de Unity3D code wordt de 3D-model geanalyseerd naar het volgende:

- Aantal geometry objecten
- Aantal shaders die in geometry bevinden.
- Aantal materialen van shaders die in geometry bevinden
- Afbeeldingen die verbonden zijn aan de materialen

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System.Linq;
public class childCount : MonoBehaviour
{
    public Renderer[] childRender;
    public MeshFilter[] childMesh;
    public Transform[] childObjects;
    public string verticesCount;
    public string textureCount;
    public string materialCount;
    public string materialUniqueCount;
    public List<string> materialNames;
    public List<string> uniqueMaterialNames;

    // Start is called before the first frame update
    void Start()
    {
        childRender = gameObject.GetComponentsInChildren<Renderer>();
        childMesh = gameObject.GetComponentsInChildren<MeshFilter>();
        getMaterial();
        verticesCount = getverticesCount();
    }

    public void setActive()
    {
        for (int i = 0; i < childRender.Length; i++)
        {
            if (childRender[i] != null)
            {
                childRender[i].enabled = true;
            }
        }
    }

    public void setNotActive()
    {
        for (int i = 0; i < childRender.Length; i++)
        {
            childRender[i].enabled = false;
        }
    }

    public string getverticesCount()
    {
        int tempCount = 0;
        for (int i = 0; i < childMesh.Length; i++)
        {
            if (childMesh[i] != null)
            {
                tempCount = tempCount + childMesh[i].mesh.verticesCount;
            }
        }
    }
}
```

```
        }

    }

    return tempCount.ToString("#,#").Replace(',', ' ');
}

public void getMaterial()
{
    int tempCount = 0;
    for (int i = 0; i < childRender.Length; i++)
    {

        int intMaterials = childRender[i].materials.Length;

        for (int x = 0; x < intMaterials; x++)
        {
            materialNames.Add(childRender[i].materials[x].name);
            if (childRender[i].materials[x].mainTexture != null)
            {
                tempCount++;
            }
        }
    }

    materialCount = materialNames.Count.ToString("#,#").Replace(',', ' ');
    materialUniqueCount = materialNames.Distinct().ToList().Count.ToString("#,#").Replace(',', ' ');
    if (tempCount == 0)
    {
        textureCount = 0.ToString();
    }
    else
    {
        textureCount = tempCount.ToString("#,#").Replace(',', ' ');
    }
}

}

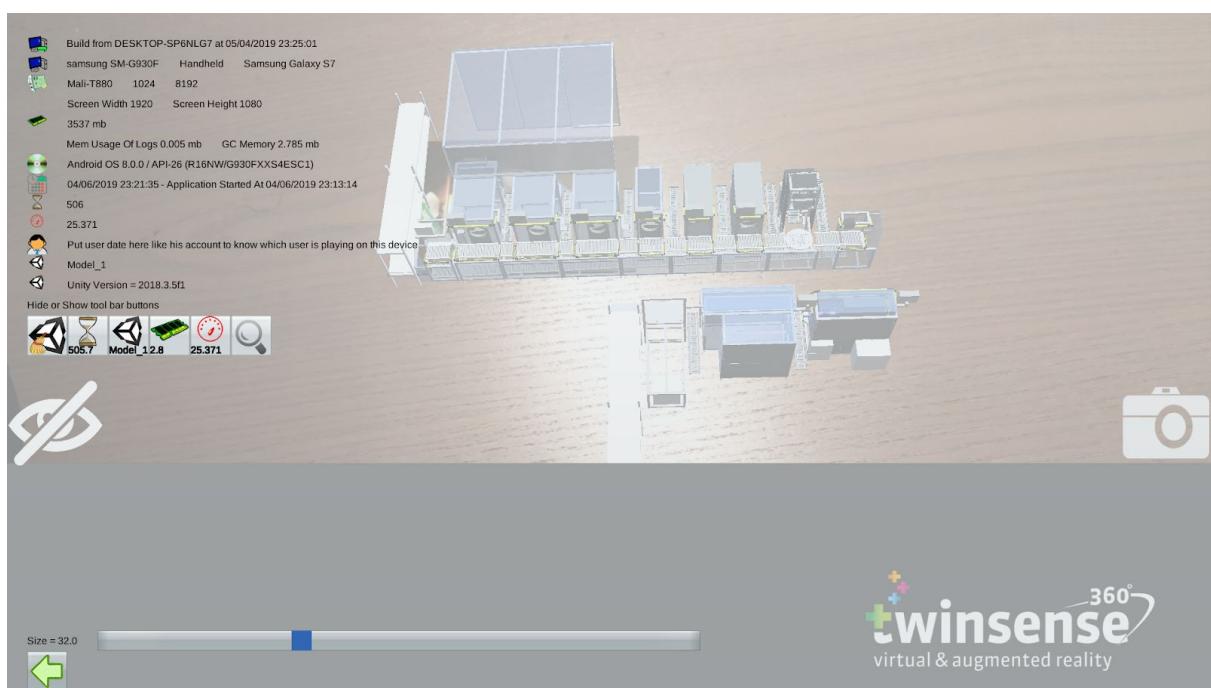
}
```

Figure 2.1: Registeren van mesh informatie

Bijlage 4. Screenshots

4.1. 3D-modellen voor deelvraag 1

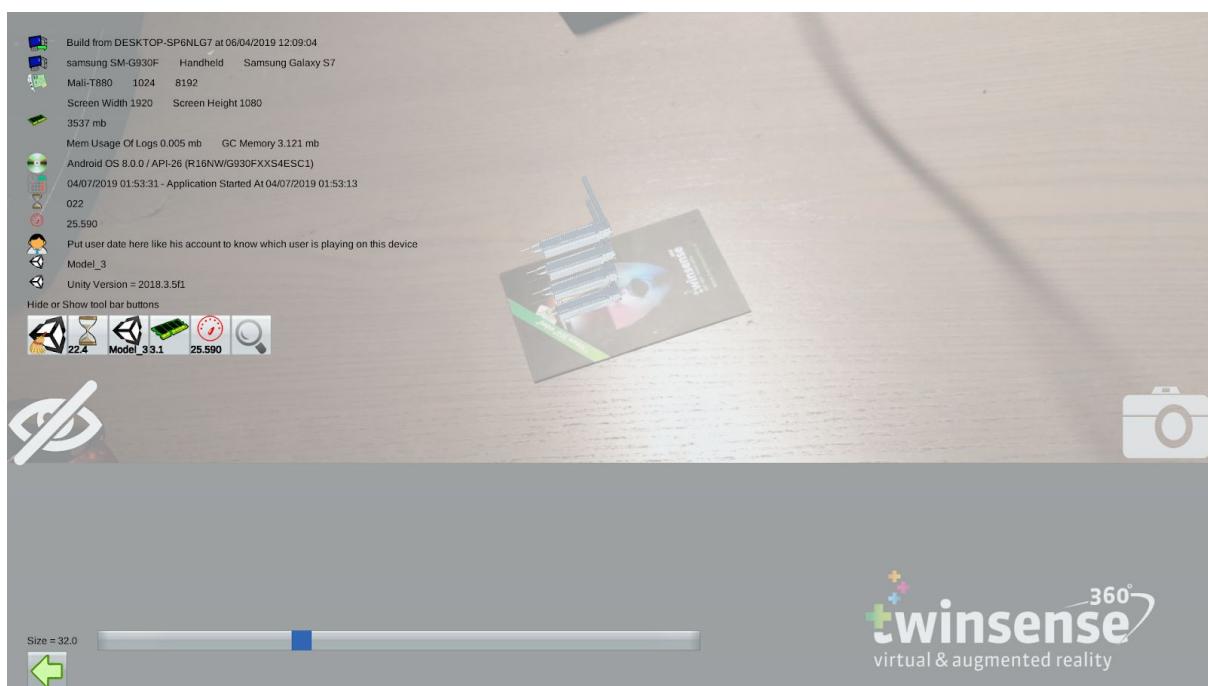
3D-Model_1



3D-Model_2



3D-Model_3

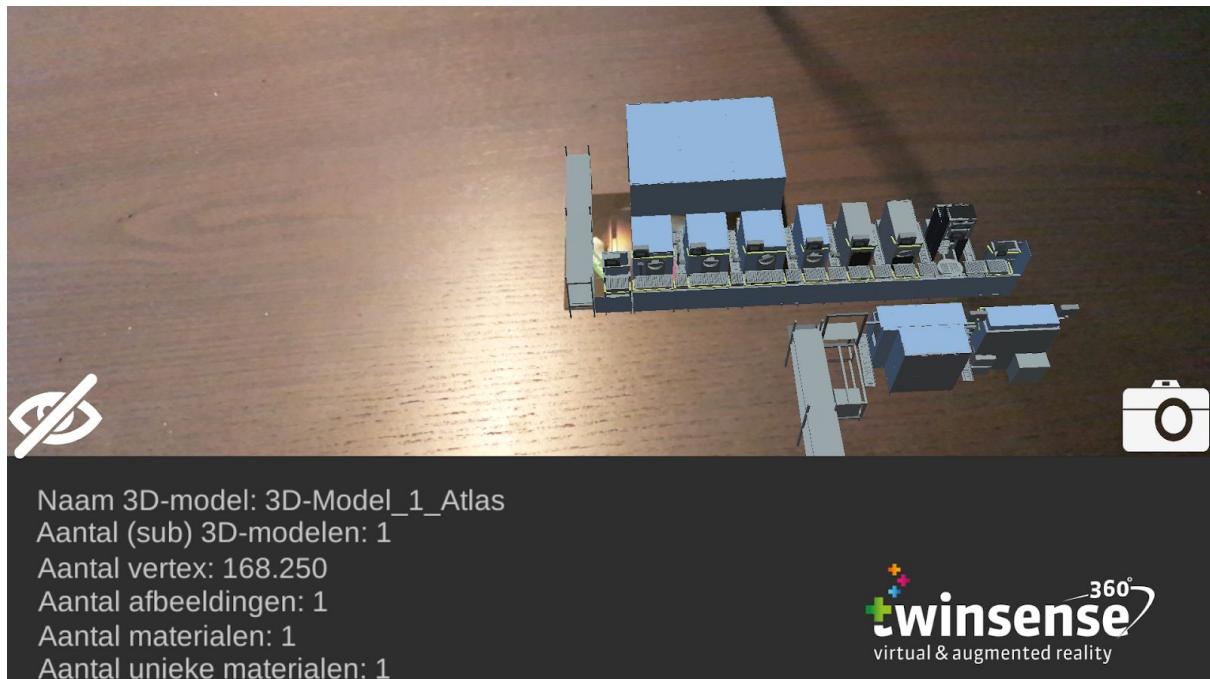


3D-Model_4

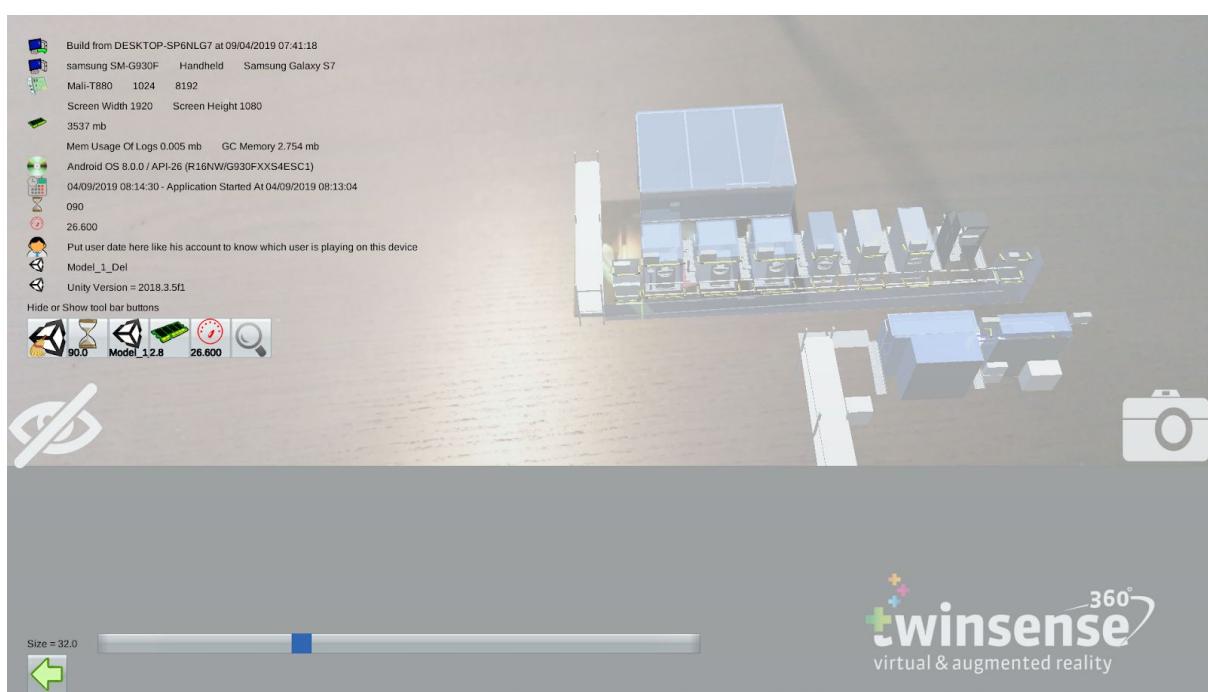
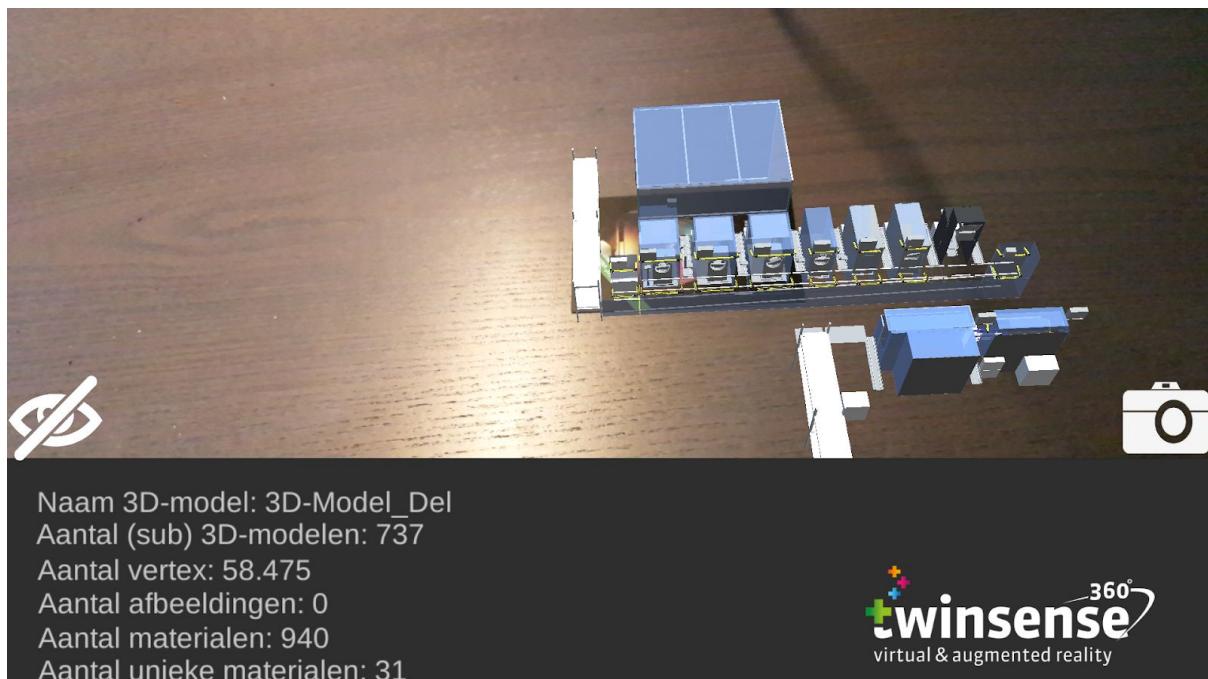


4.2. 3D-modellen voor deelvraag 1

3D-Model_1_Atlas



3D-Model_1_Del



3D-Model_2_Dec_50



Naam 3D-model: Model_2_Dec_50

Aantal (sub) 3D-modellen: 12

Aantal vertex: 523.337

Aantal afbeeldingen: 12

Aantal materialen: 12

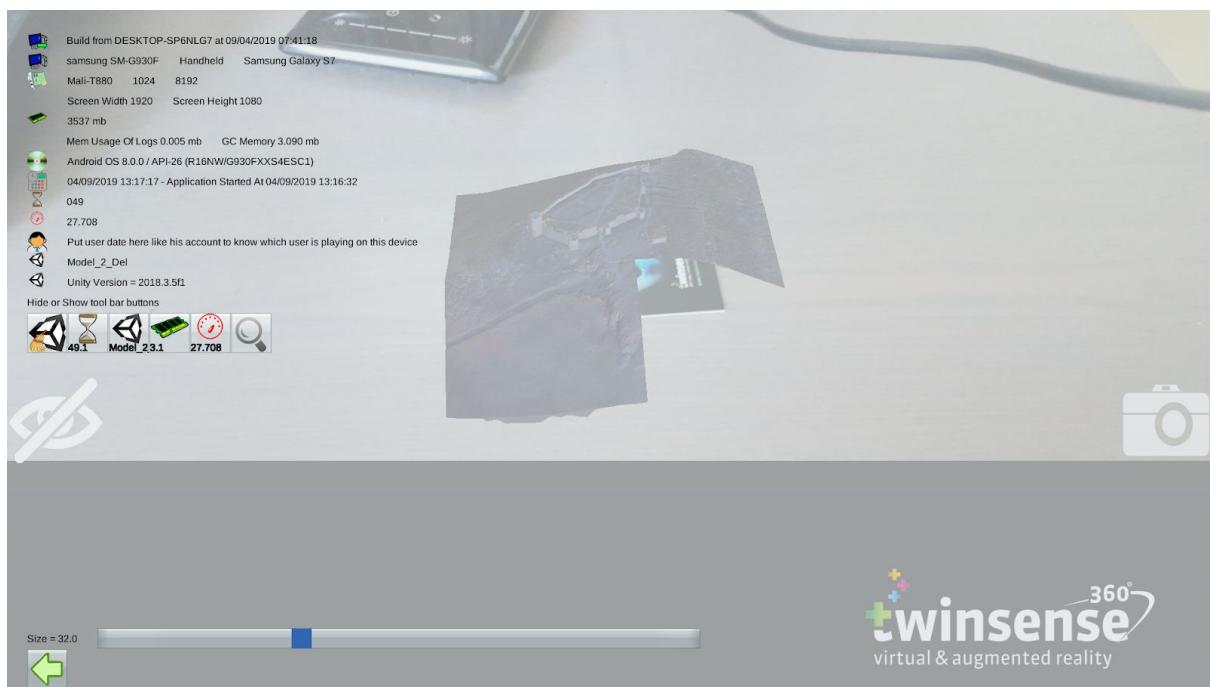
Aantal unieke materialen: 12



3D-Model_2_Del



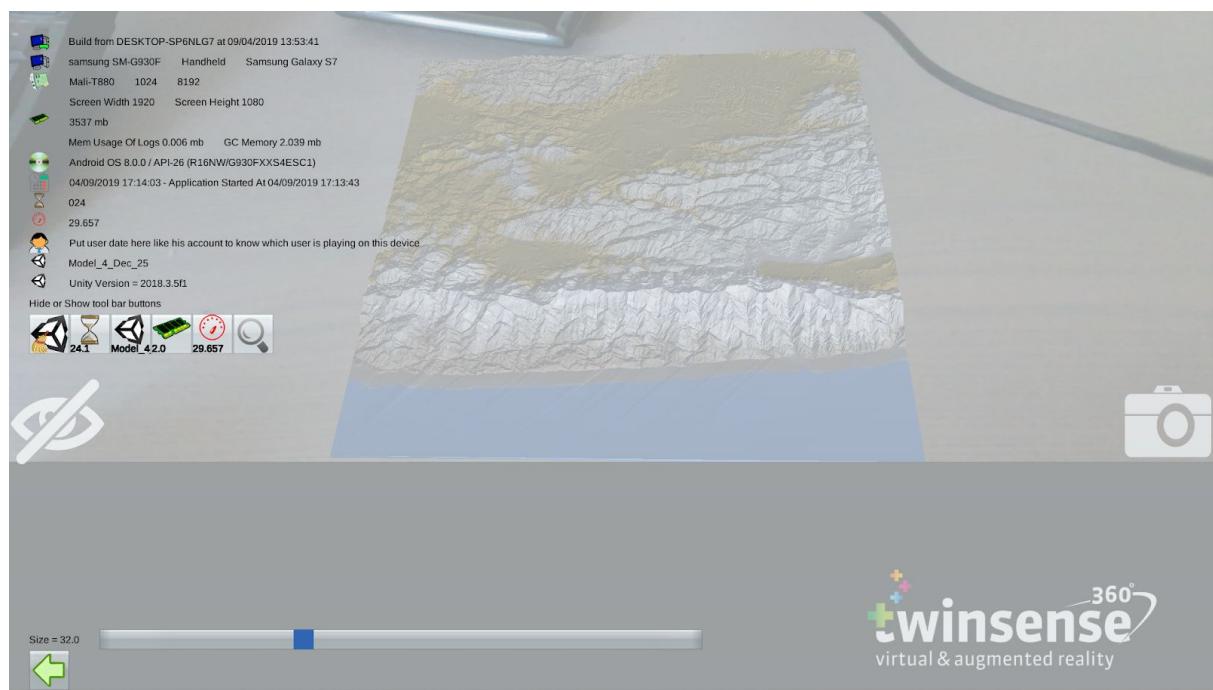
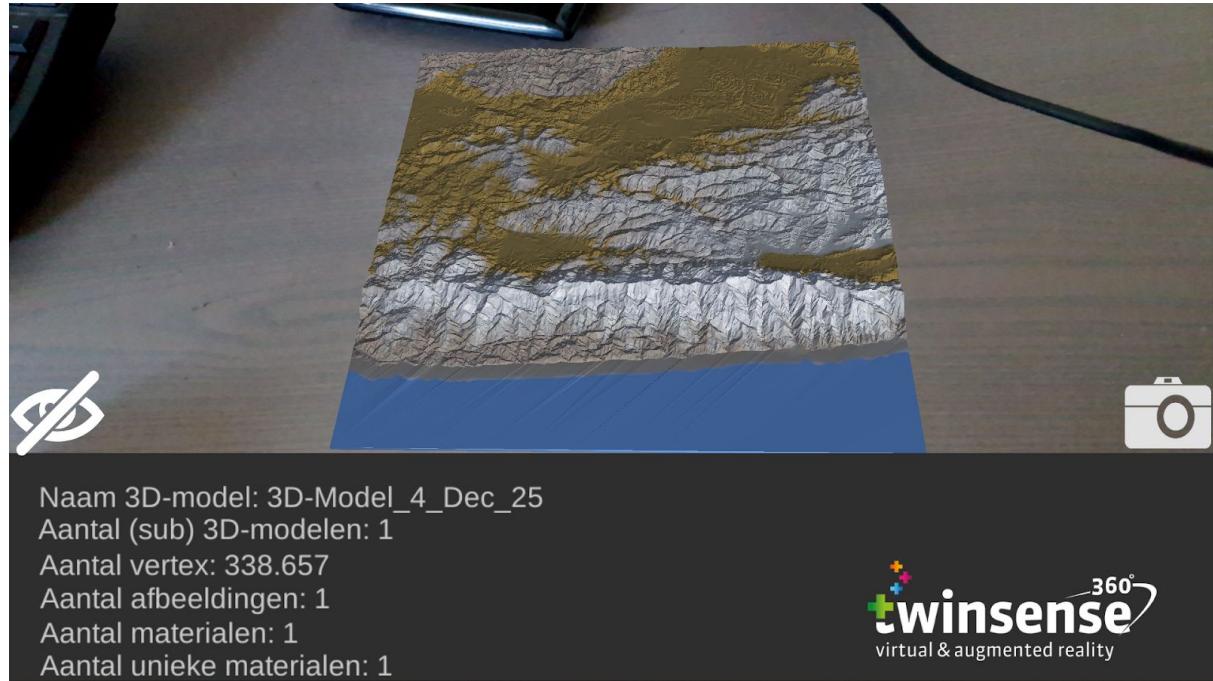
Naam 3D-model: Model_2_Del
Aantal (sub) 3D-modellen: 3
Aantal vertex: 537.249
Aantal afbeeldingen: 3
Aantal materialen: 3
Aantal unieke materialen: 3



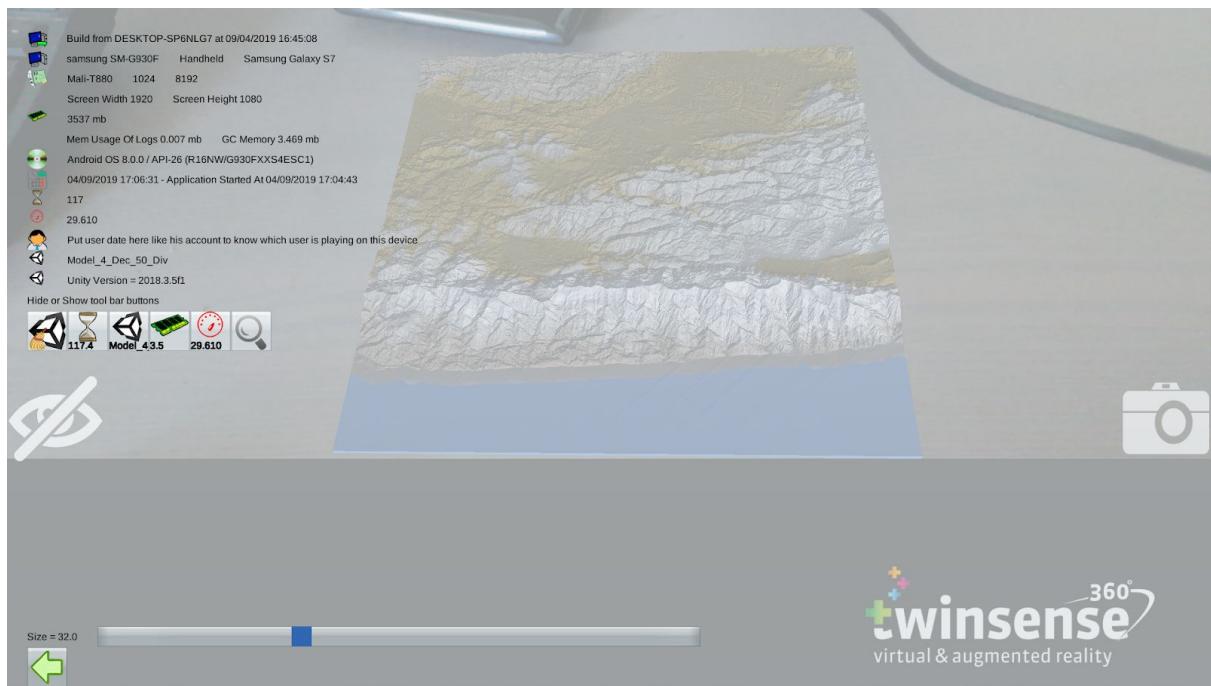
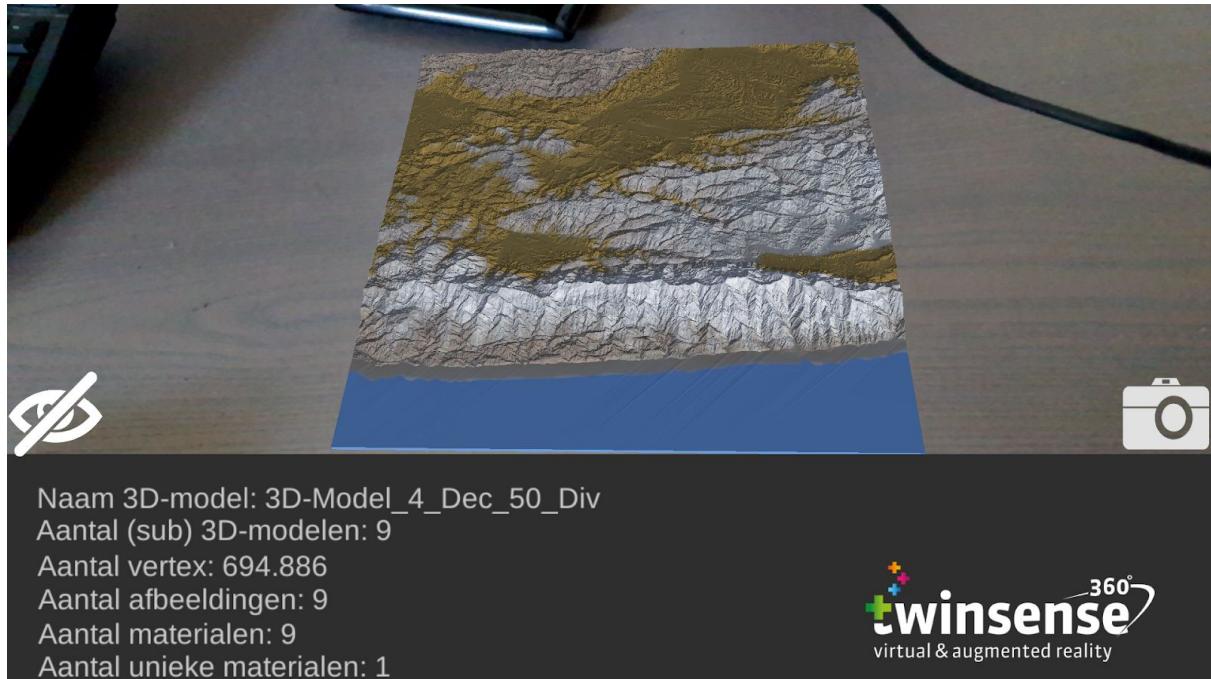
3D-Model_3_Del



3D-Model_4_Dec_25



3D-Model_4_Dec_50_Div



4.3.3D-modellen van deelvraag 2

3D-Model_1 en 3D-Model_1_Atlas



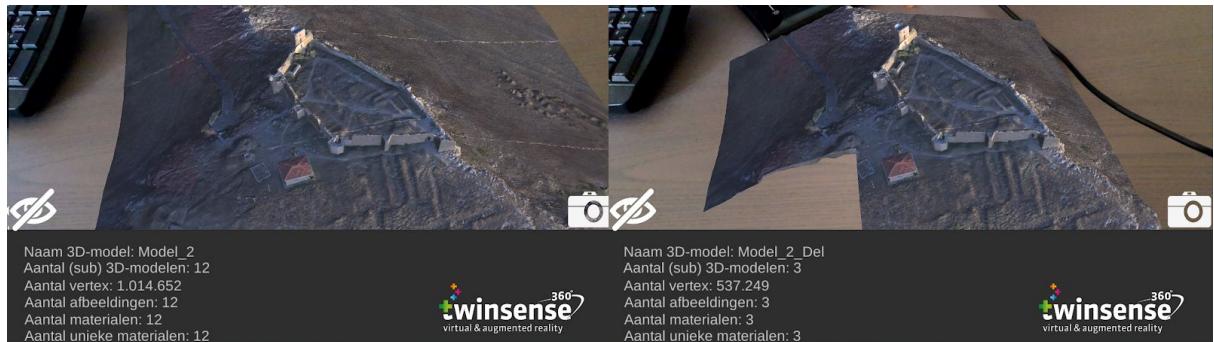
3D-Model_1 en 3D-Model_1_Del



3D-Model_2 en 3D-Model_2_Dec_50



3D-Model_2 en 3D-Model_2_Del



3D-Model_3 en 3D-Model_3_Del



3D-Model_4 en 3D-Model_4_Dec_25



3D-Model_4 en 3D-Model_4_Dec_50_Div



Bijlage 5. Adviesrapport

INHOUDSOPGAVE

Bijlage 5. Adviesrapport	78
5.1. Introductie	83
5.1.1. Belangrijke links voor bestanden	83

5.1. Introductie

De beroepsproduct is bedoeld voor de designers en ontwikkelaars van Twinsense.

In deze hoofdstuk worden de benodigdheden beschreven wat nodig is voor het project.

De software dat gebruikt wordt voor deze prototype is Unity3d 2018.2.1 en 3ds Max 2019.

De modules Android en Vuforia zijn al geïnstalleerd in Unity3D.

5.1.1. Belangrijke links voor bestanden

Prefab van de tool.

<https://drive.google.com/file/d/1ZK8ao4-lbCHpkqQM7cvGnuWrjB72zaO8/view?usp=sharing>

De Vuforia database is geïmplementeerd in de Unity3D project.

Vuforia database: https://drive.google.com/open?id=1QOfM_1nQ4VTQ-BYd-mDgpE7hGT3YgXcc

User \ID voor vuforia:

ASrZlDv////AAABmbe9Yq7crUkvtW18rlsKh3c5DODeUwD+G/IeyJvQhFYz7jSzbFESXCpT6Se5zNBx8c9128jf6lXZIwbNbqz0Q7hYHaei9/7aE7URQIWaDpYJDsFFZ71YP0grUCD5l8azw8d9wzBx1ieGJUgFCyhHXrEzK91rmgC9+yOuLdXRVXL68EUSOe6ncC2AassZ5rOvBwdxaxCgBKf6WDgeLJ6JmmEYH3qnihGwD9LOsMLPnFzFrrqSh0IBC0i1fecHsK+r5rNEMJZiATS1Poz2nZBmUb3i9lz7TGMcQ44rRKp4MPtNoXKIgSvGB7pF8kzoM0AGfSkIWEzHVUIVM5e5pV7LTKIge6qCe1SQumcbz6wtq/gc

Link voor Native Gallery

<https://assetstore.unity.com/packages/tools/integration/native-gallery-for-android-ios-112630>

De twee voorbeeld 3D-modellen bevinden via deze link.

Model 1: https://drive.google.com/open?id=1sxIpclhP_3VnmzstGVcC8wU29d8vn6Ou

Model 2:

https://drive.google.com/drive/folders/11CLNet2BRJAaSI9Uw_xXkXSD1G30kw2h?usp=sharing

Tutorial video

https://www.youtube.com/channel/UCQ3_npdsXUKD14Gz0yVhazA