ingenieur
wissenschaften
htw saar

Hochschule für
Technik und Wirtschaft
des Saarlandes
University of
Applied Sciences

EmRoLab

Dokumentation der Projektarbeit modularer VR-Fahrzeugkonfigurator

Max Schmidt, Florian Speicher, Tim Wall

Saarbrücken, den 17. Juli 2025

Inhaltsverzeichnis

1 Einleitung

Im Rahmen unserer Projektarbeit entwickeln wir einen modular aufgebauten Fahrzeugkonfigurator für Virtual Reality, der auf der Meta Quest 3 lauffähig ist. Ziel des Projekts ist es, eine benutzerfreundliche Anwendung zu realisieren, die es ermöglicht, Fahrzeuge in einer virtuellen Umgebung individuell zu konfigurieren.

1.1 Motivation

Die Automobilbranche befindet sich in einem tiefgreifenden Wandel, geprägt durch Digitalisierung, Elektromobilität und veränderte Kundenbedürfnisse. Insbesondere im Vertrieb gewinnen digitale Lösungen zunehmend an Bedeutung, um potenziellen Käufern ein interaktives und personalisiertes Produkterlebnis bieten zu können. Klassische 2D-Konfiguratoren auf Webseiten stoßen dabei zunehmend an ihre Grenzen, was Erlebnis und Nutzerbindung betrifft.
Virtual Reality eröffnet neue Möglichkeiten, Produkte emotional und räumlich erlebbar zu machen. Durch den Einsatz von VR-Headsets wie die Meta Quest 3 lassen sich Fahrzeuge in Echtzeit visualisieren, konfigurieren und in einer natürlichen, als auch ansprechenden Umgebung betrachten und das alles unabhängig vom physischen Standort des Nutzers. Besonders für jüngere Zielgruppen oder technikaffine Kunden kann ein VR-Fahrzeugkonfigurator ein innovatives und attraktives Tool darstellen.
Die Motivation hinter dieser Projektarbeit ist es, das Potenzial dieser Technologien im Bereich Produktkonfiguration zu erforschen und einen funktionalen Prototypen zu entwickeln, der technische Machbarkeit, Benutzerfreundlichkeit und visuelle Qualität vereint. Dabei stehen praxisnahe Anwendung, moderne Interaktion sowie Performance und Skalierbarkeit im Fokus.

1.2 Zielsetzung der Projektarbeit

Der Fahrzeugkonfigurator wird mit der Gameengine Unity speziell für die Meta Quest 3 umgesetzt und soll auf dieser Plattform stabil und performant betrieben werden können. Durch die gezielt implementierte Modularität des Systems wird sichergestellt, dass sich zukünftige Erweiterungen, in Form zusätzlicher Fahrzeugmodelle, sowohl Zwei- als auch Vierräder und weitere Features mit geringem Aufwand integrieren lassen. Dadurch bleibt die Modellpalette flexibel und kann bei Bedarf dynamisch erweitert werden.

2 Theoretische Grundlagen

In diesem Kapitel werden die wichtigsten technischen und konzeptionellen Grundlagen vorgestellt, die für die Entwicklung des Projekts relevant sind. Ziel ist es, ein fundiertes Verständnis der eingesetzten Technologien sowie der interaktiven Prinzipien zu vermitteln.

2.1 Virtual Reality und Meta Quest 3

Virtual Reality (VR) bezeichnet eine computergenerierte, interaktive Umgebung, die Nutzer über ein VR-Headset erleben können. Die **Meta Quest 3** ist ein solches VR-Headset, das ohne externe Rechner oder Sensoren funktioniert. Mit **Inside-Out-Tracking**, **Hand- und Controller-Tracking**, sowie der Unterstützung für **Passthrough-AR** eröffnet sie neue Möglichkeiten für fortschrittliche Interaktionen. Für Entwickler bietet die Meta Quest Plattform Tools zur Optimierung von Performance, Rendering und Usability in VR-Anwendungen.

2.2 3D-Modellierung und Interaktionsdesign

Die **3D-Modellierung** bildet die Grundlage für die visuelle Darstellung des konfigurierbaren Fahrzeugs. Modelle müssen sowohl ästhetisch ansprechend als auch performant sein. Typische Werkzeuge sind **Blender** oder **externe CAD-Systeme** mit anschließender Optimierung für **Echtzeitdarstellung**. In unserem Projekt ist dies jedoch durch die Verwendung von vorgefertigten Modellen nicht von großer Bedeutung. Im **Interaktionsdesign** steht die Benutzerfreundlichkeit im Vordergrund. In VR bedeutet dies intuitive Steuerung, klare visuelle Rückmeldungen der Ergebnisse und eine ergonomische Gestaltung der Benutzeroberflächen, um anstrengende Körperhaltungen möglichst zu meiden.

2.3 Fahrzeugkonfiguration im digitalen Raum

Die digitale Fahrzeugkonfiguration hat sich von 2D-Webanwendungen hin zu modernen 3D-Erlebnissen entwickelt. In VR eröffnen sich neue Möglichkeiten, wie das räumliche Erleben von Modellen, dynamisches Wechseln von Fahrzeugteilen (z.B. Felgen, Farben, Ausstattungen) und die direkte Interaktion durch Hand- oder Controllerbewegungen. Für eine erfolgreiche Umsetzung sind eine **modulare Architektur** und eine **flexible Logik** zur dynamischen Kombination von Komponenten essenziell.

3 Projektbeschreibung

Diese Projektarbeit befasst sich mit der Entwicklung eines dynamischen Fahrzeugkonfigurators in Virtual Reality für die Meta Quest 3. Ziel ist es, Nutzern die Möglichkeit zu geben, ein Fahrzeug in einer immersiven Umgebung individuell zusammenzustellen und zu erleben. Desweiteren soll das Projekt an die Systeme des Testfeldes der HTW angeschlossen werden können, um die Produktion des konfigurierten Fahrzeugmodells zu gestalten.

3.1 Projektidee und Anforderungen

Die Grundidee war es, klassische Fahrzeugkonfiguratoren, wie sie häufig auf Webseiten zu finden sind, in eine räumliche und interaktive VR-Erfahrung zu überführen. Im Mittelpunkt stehen die intuitive Bedienung, sowie erweiterte Funktionen wie Augmented Reality (AR) und eine animierte Explosionsdarstellung des Fahrzeugs. Die Applikation sollte vollständig auf der Meta Quest 3 lauffähig sein und sowohl technisch stabil als auch performant umgesetzt werden.

3.2 Zielgruppe und Nutzungsszenarien

Die Zielgruppe umfasst technikaffine Nutzer, Kunden/Besucher der HTW sowie Interessierte im Bereich VR/AR-Technologien. Denkbare Nutzungsszenarien reichen bis zum Einsatz auf Messen und in Showrooms. Der Konfigurator kann zudem als Grundlage für weitere Forschung im Bereich virtualisierter Produktpräsentationen dienen.

3.3 Funktionale und nicht-funktionale Anforderungen

Zu den funktionalen Anforderungen zählen:

- Auswahl zwischen zwei Modellen: Erlbach und einem Fahrrad.
- Konfiguration von Teilen wie Felgen, Grill und weiteren Komponenten.
- Farbauswahl verschiedener Teile
- Darstellung des Modells in 3D mit frei drehbarer Ansicht.
- Aktivierbare Explosionsdarstellung über Animation.
- Umschaltfunktion in eine AR-Ansicht.
- Export der Konfiguration in Form eines JSON.
- Zum Kontext passende VR-Umgebung.
- Grabben der UI Menüelemente.
- Auswahl der Teile durch Klicken auf das Model.

Nicht-funktionale Anforderungen umfassten vor allem:

- Eine flüssige Nutzererfahrung auf der Meta Quest 3.
- Intuitive Bedienung mithilfe des Meta Interaction SDK.
- Stabile Performance durch effiziente Modell- und Szenengestaltung.

4 Konzeption und Planung

Vor Beginn der Implementierung wurde eine fundierte Konzeption der Anwendung im Team besprochen. Ziel war es, eine modulare, erweiterbare und benutzerfreundliche VR-Anwendung zu entwickeln, die sich speziell für die Meta Quest 3 eignet. Neben technischen Aspekten flossen auch gestalterische und nutzerzentrierte Überlegungen in die Planung ein.

4.1 Technologiestack und Tools

Als Entwicklungsumgebung wurde die Unity Engine gewählt, da sie eine breite Unterstützung für XR-Anwendungen bietet und direkt mit dem OpenXR-Framework kompatibel ist. Für die Umsetzung der VR-spezifischen Interaktionen wurde das **Meta Interaction SDK** verwendet, das bereits optimierte Interaktionsbausteine (Building Blocks) für gängige VR-Aktionen mitliefert.
Weitere zentrale Technologien und Tools:

- **C#** als Hauptprogrammiersprache zur Steuerung der Logik.
- **OpenXR**: Plattformentsprechende API für XR-Geräte, kompatibel mit der Meta Quest 3.
- **Blender**: Für Modelloptimierung und Bearbeitung von 3D-Assets.
- **Git**: Versionskontrolle und kollaboratives Arbeiten.

4.2 Systemarchitektur

Die Anwendung wurde modular aufgebaut, um Erweiterbarkeit und Wartbarkeit zu gewährleisten. Die grundlegende Struktur besteht aus zwei Hauptdarstellungen:

1. **Modellauswahl**: Nutzer wählen zwischen Erlbach und einem Fahrrad. Hier werden nach dem Hinzufügen auch direkt weitere Modelle angezeigt.
2. **Konfigurationsansicht**: Darstellung des gewählten Modells, inklusive Konfiguration, Rotation, AR-Modus und Explosionsdarstellung.

Die Konfigurationslogik basiert auf Ports und ChildModels für austauschbare Komponenten (z. B. Felgen, Grill). Interaktionen werden über UI-Elemente (Buttons, Panels) sowie über Hand- und Controller-Gesten gesteuert.

4.3 Designentscheidungen

Bei der Gestaltung der Benutzeroberfläche und Interaktionen wurde besonderer Wert auf folgende Aspekte gelegt:

- **Intuitive Bedienung**: Klare Menüführung und direkte Rückmeldung bei Eingaben über ersichtliche Ergebnisse der Aktion.

- **Komponentenbasierter Aufbau**: Ermöglicht zukünftige Erweiterung um weitere Modelle oder Konfigurationsteile.
- **Mobile-Optimierung**: Reduzierung der Modellkomplexität zur Sicherstellung stabiler Performance auf der Meta Quest 3.
- **Visuelle Klarheit**: Reduktion auf wesentliche Elemente zur besseren Orientierung in der VR-Umgebung.

4.4 Projektmanagement

Die Umsetzung des Projekts erfolgte in etwa zu den im Zeitplan angegebenen Zeiträumen. Jedoch mussten wir einige Punkte schon vor dem Zwischenmeeting verändern, da wir uns bei der Umsetzung der Buttons und der Handinteraktionen etwas übernommen haben. Diese Punkte wurden dann in der Zeit nach dem Zwischenmeeting verbessert. Außerdem wurde im späteren Verlauf der Zeitplan nochmal ein wenig angepasst, da wir schon früher zum Testen der Grundfeatures kamen und weitere Features und Code-Optimierungen implementieren wollten. Somit umfasst das Testen nicht mehr so viel Zeit am Ende. Da wir schon schneller als geplant mit den fehlenden Komponenten vom Zwischenmeeting fertig geworden sind, konnten wir die von uns als nützlich betrachteten weiteren Features wie die Grab-Interaktion auch implementieren.
Das Projekt wurde anhand eines Zeitplans in folgende Phasen unterteilt:

KW	Beschreibung
KW17	**Recherche und Einarbeitung:** Grundlagen zu VR, Unity und Meta Interaction SDK.
KW18-19	**Ausarbeitung der Projektstruktur:** Festlegung der Systemarchitektur, Designentscheidungen und erste Prototypen.
KW20-22	**Entwicklung zur Änderung von Teilemodellen:** Implementierung der Logik zum Austausch von Komponenten.
KW23	**Entwicklung von Komponenteninfos:** UI-Elemente zur Anzeige von Informationen zu den einzelnen Teilen.
KW24	**Export der Konfigurationen:** Implementierung der Exportfunktion für die Konfigurationen in Form von JSON.

Fortsetzung auf nächster Seite

KW	Beschreibung
KW25-26	**Fehlende Komponenten:** Implementierung der fehlenden Teile, wie AR-Modus und verbesserte Explosionsanimation etc.
KW27-28	**Erweiterte Features:** Implementierung von zusätzlichen Features wie Grab-Interaktion, Hintergrundwelt, Klick auf Teile im Modell etc.
KW29	**Testphase und Bugfixing:** Überprüfung der Funktionalität und Performance.
KW29	**Dokumentation und Pufferphase:** Zusammenfassung der Ergebnisse, Erstellung der Dokumentation, Abschlusspräsentation und Pufferzeit für unvorhergesehene Probleme.

5 Implementierung

Die Umsetzung des dynamischen Fahrzeugkonfigurators erfolgte mithilfe der Unity-Engine in Kombination mit OpenXR und der Meta Interaction SDK. Dabei lag der Fokus auf einer modularen Architektur, die sowohl eine klare Benutzerführung als auch die Erweiterbarkeit um neue Modelle und Komponenten ermöglicht.

5.1 Umsetzung in Unity

Die Anwendung wurde mit der Unity Engine entwickelt, die sich durch ihre gute Unterstützung für VR und ihre Kompatibilität mit der Meta Quest 3 auszeichnet. Zur Integration von VR-Funktionalitäten wurde OpenXR als Schnittstelle verwendet. Für Interaktionen kamen die Meta Interaction SDK Building Blocks zum Einsatz, mit denen standardisierte VR-Elemente wie Button-Interaktionen, Grabbing und Hand- sowie Controllereingabe umgesetzt werden konnten.

5.2 VR-Interaktion und Benutzersteuerung

Die Benutzeroberfläche gliedert sich in zwei Hauptbereiche: die Modellauswahl und die Konfigurationsansicht. In der ersten Ansicht kann über Buttons zwischen dem Erlbach und einem Fahrrad gewählt werden. Nach der Auswahl wird das jeweilige Modell in die Szene geladen.
In der Konfigurationsansicht kann das Modell mit Hilfe der Joysticks der Controller oder an einem Slider gedreht und betrachtet werden. Die Drehung ist außerdem mit den Thumbsticks des Controllers möglich. Einzelne Fahrzeugteile wie Felgen oder der Grill lassen sich durch Auswahlbuttons im dafür vorgesehenen Menü dynamisch verändern. Die Steuerung wurde so gestaltet, dass sie für VR-Einsteiger intuitiv verständlich ist, aber trotzdem alle essentiellen Features beinhaltet.

5.3 Dynamische Fahrzeugkonfiguration

Die Konfiguration basiert auf der Auswahl von Child-Modellen innerhalb eines im Base-Models definierten Ports. Beim Auswählen einer Komponente wird das entsprechende 3D-Objekt geladen und an der vorgesehenen Position instanziert. Durch diese modulare Struktur lassen sich neue Teile einfach ergänzen. Die Auswahl verschiedener Ausführungen für ein Teil erfolgt durch das ersetzten des an dem Port liegenden ChildModels durch das gewählte Teil.

5.4 Integration von AR-Ansicht und Explosionsdarstellung

Ein zentrales Feature ist die Umschaltung in einen AR-Modus, in dem das Modell mithilfe des Passthrough-Modus der Meta Quest 3 in die reale Umgebung projiziert wird. Dadurch kann das konfigurierte Fahrzeug realitätsnah im Raum betrachtet werden.
Zusätzlich wurde eine Explosionsanimation implementiert, bei der sich alle Bauteile des Modells von der Basis lösen und sich radial voneinander entfernen. Diese Funktion wurde über animierte Positionsoffsets realisiert und dient dazu, die einzelnen Komponenten des

Fahrzeugs verständlich darzustellen, beispielsweise zu Demonstrations- oder Lernzwecken oder auch um Teile im Inneren sehen und konfigurieren zu können.

5.5 Features

Die Anwendung bietet eine Reihe von Funktionen, die sowohl auf Benutzerfreundlichkeit als auch auf technologische Machbarkeit in einer mobilen VR-Umgebung ausgelegt sind. Im Folgenden werden die zentralen Features zusammengefasst:

- **Modularer Aufbau:** Die Anwendung ist so konzipiert, dass neue Fahrzeugmodelle und Konfigurationsoptionen einfach hinzugefügt werden können. Dies ermöglicht eine flexible Erweiterung der Modellpalette.
- **Hand- und Controller-Interaktionen:** Die Benutzer können das Fahrzeugmodell mit Hilfe der Joysticks der Controller oder an einem Slider drehen, Teile auswählen und konfigurieren. Dies verbessert das Nutzererlebnis in der VR-Umgebung und erleichtert unerfahrenden Nutzern die Bedienung.
- **Modellauswahl:** Zu Beginn der Anwendung können Nutzer zwischen den Fahrzeugmodellen wählen. Die Auswahl erfolgt über einfach zugängliche Buttons beim Start der Szene.
- **3D-Modellansicht mit Rotation:** Nach der Auswahl wird das Modell in der interaktiven Szene dargestellt. Mithilfe von Controllern oder Handeingaben kann das Modell gedreht und aus verschiedenen Perspektiven betrachtet werden.
- **Komponenten-Konfiguration:** Einzelne Fahrzeugteile, wie z. B. Felgen, Grill oder weitere Komponenten lassen sich per UI-Button individuell austauschen. Die Änderungen werden in Echtzeit am Modell sichtbar.
- **Farbauswahl:** Nutzer können die Farbe einzelner Teile des Fahrzeugs anpassen, um eine personalisierte Konfiguration zu erstellen.
- **Explosionsanimation:** Durch einen zusätzlichen Button lässt sich eine animierte Explosionszeichnung auslösen, bei der sich die einzelnen Komponenten vom Fahrzeug lösen und räumlich voneinander entfernen. Dies bietet eine anschauliche Darstellung des Modellaufbaus.
- **AR-Ansicht (Passthrough):** Die Anwendung unterstützt den Wechsel in einen Augmented-Reality-Modus, in dem das Fahrzeugmodell über die Passthrough-Funktion der Meta Quest 3 in der realen Umgebung angezeigt wird. Dadurch entsteht ein hybrides Erlebnis aus physischer und digitaler Welt.
- **Export der Konfiguration:** Nutzer können ihre Fahrzeugkonfigurationen in Form eines JSON-Dokuments exportieren. Dies ermöglicht eine spätere Wiederverwendung oder Weitergabe der Konfiguration.
- **Grab-Interaktion:** Die Benutzeroberfläche ermöglicht es, UI-Elemente durch Greifen und Ziehen zu interagieren.

- **Interaktion mit Modellteilen:** Nutzer können direkt auf Teile des Modells klicken, um diese auszuwählen und zu konfigurieren. Dies ermöglicht eine intuitive und direkte Interaktion mit dem Fahrzeug.
- **Hintergrundwelt:** Die Anwendung bietet eine ansprechende virtuelle Umgebung, die das Fahrzeugkonfigurator-Erlebnis unterstützt und eine realistische Präsentation der Modelle ermöglicht. Wir haben passend zum Fahrzeugkonfigurator dafür als Umgebung ein Modell des Red Bull Ring Spielberg Österreich gewählt.
- **Materialien und Texturen:** Die Modelle sind mit realistischen Materialien und Texturen ausgestattet, die eine ansprechende visuelle Darstellung in der VR-Umgebung gewährleisten.
- **Optimierte Performance für Standalone-VR:** Alle Modelle und Materialien wurden für die Verwendung auf der Meta Quest 3 optimiert, um eine stabile Framerate und kurze Ladezeiten zu gewährleisten.

Diese Funktionen stellen die wesentlichen Interaktions- und Darstellungsmöglichkeiten des Prototyps dar und bilden die Grundlage für mögliche zukünftige Erweiterungen der Anwendung.

5.6 Hinzufügen von neuen Modellen

Um neue Modelle in die Anwendung zu integrieren, wurde ein flexibles System entwickelt, das es ermöglicht, jede Art von Modellen einfach hinzuzufügen.

- Das Modell muss im passendem Format im Assets/Resources Ordner abgelegt werden (z.B. .fbx, .obj)
- Das Material muss im passendem Format im Assets/Resources Ordner abgelegt werden
- Ein Eintrag des jeweiligen Typen (BaseModel, ChildModel, Ansammlung an ChildModel's) muss in der ModelList getätigt werden
  - **BaseModel:** Bildet das Grundlegende Modell, was hinzugefügt werden muss
  - **ChildModel:** Bilden alle Teile des Modells. Hier kann zwischen wählbaren und festen Modellen unterschieden werden. Es kann zusätzlich eine Explosionsrichtung und die wählbaren Farben hinterlegt werden, sowie die Position des Teils angepasst werden. Hierfür muss jedoch der Modellport und die Id Richtig zu dem Teil gemappt sein.
  - **Ansammlung an ChildModels:** Eine Ansammlung an Childmodels erlaubt es aus einer Modelldatei mehrere ChildModels raus zu laden.

Diese Struktur erlaubt es, neue Modelle ohne tiefgreifende Änderungen am Code hinzuzufügen.

6 Technische Dokumentation

Das Unity Projekt basiert auf verschiedenen C# Klassen, welche den Ablauf Regeln. Dabei unterscheiden wir zwischen 3 Typen.

6.1 Informationsmodell

Die Klassen des Informationsmodell befinden sich in `Assets/Scripts/Model` und beschreiben die theoretische Ausführung eines Modelles. Sie tragen die Informationen über die einzelheiten der Modelle, führen selbst jedoch nichts aus.

- **Model**
  Das Model ist die Grundlage für das Base- und ChildModel. Es umfässt Namen (Human-Readable), ID, Mesh (Form), Material (Oberfläche), Position, Rotation, Skalierung, Mögliche Farbauswahlen. Die Möglichkeit des Passthrough-Models existiert auch, welches für Hilfsobjekte gedacht ist, und Informationen wie die Farbe an seine Kinder weiterleitet. Auch werden Ports definiert.

- **Port**
  Ein Port ist eine Schnittstelle, welche von einem Modell definiert wird und passenden ChildModels erlaubt, angebunden zu werden. Er trägt eine nicht-unique PortID, eine DefaultID (eine Modell-ID welche Standardmäßig an diesem Port geladen wird), eine Chooseable-Flag (ob der Port in der Auswahl erscheint), Positionsinformationen, einen Human-Readable Name der im Menü erscheint (sofern Chooseable), sowie eine Explode-Direction für das Explosions-Feature.

- **BaseModel**
  Ein BaseModel ist der Startpunkt einer Konfiguration. Es ist im Hauptmenü auswählbar. Die Farbe des BaseModels ist in der aktuellen Struktur nicht änderbar und es ist empfohlen, ein leeres oder festes unänderbares Objekt als BaseModel auszuwählen, um maximale Konfigurierbarkeit zu gewährleisten.

- **ChildModel**
  Ein ChildModel ist ein an einen Port angebundenes Modell. Der ParentPort ist der Name des Ports an den dieses Modell angebunden werden darf. Das ChildModel selbst kann auch Ports definieren, wodurch eine Rekursive Baumstruktur möglich ist.

6.2 Real-Modelle

Die Real-Modelle befinden sich in `Assets/Scripts/Model` und beschreiben die konkreten MonoBehaviours welche dem Nutzer angezeigt werden. Sie bilden das Informationsmodell somit ab.

- **ModelBehaviour**
  Ein ModelBehaviour verwaltet konkret die Visualisierung eines Ports sowie seiner Kinder. Es ist einem konkreten Port zugewiesen (Ausnahme BaseModelSelector) und zeigt das aktuell ausgewählte Modell des Ports an. Der Logikablauf des Modellwechsels ist in diesem implementiert.

- **BaseModelBehaviour**
  Die Implementierung des ModelBehaviour's für das BaseModel. Sie enthält keine Logik über die des ModelBehaviour's und existiert nur zu Identifikationszwecken

- **ChildModelBehaviour**
  Die Implementierung des ModelBehaviour's für das ChildModel. Sie enthält neben dem ModelBehaviour Informationen die nur ChildModels tragen. Dies umfässt Port sowie Parent.

6.3 Manager

Die Manager verwalten verschiedene Aspekte des Programmablaufes.

- **ModelManager**
  Der ModelManager befindet sich in `Assets/Scripts/Model` und verwaltet das Wissen über alle Informationsmodelle und deren (theoretische) Zusammenhänge. Nur Modelle die dem ModelManager bekannt sind können genutzt werden. Er kennt und verwaltet das aktuelle BaseModel sowie dessen BaseModelBehaviour.

- **StateManager**
  Der StateManager befindet sich in `Assets/Scripts/Managing` und verwaltet alle IResettable's. Dies ist ein Interface welches alle bei Modelwechsel zu resettenden Elemente anspricht. Es wird konkret von dem "Return to Menu"Knopf genutzt.

6.4 Transformer

Die Transformer verwalten das Interaktionsverhalten der angezeigten Konfiguration.

- **Exploder**
  Der Exploder verwaltet den Explosionszustand der Konfiguration und spielt bei Explode oder Unexplode eine Animation ab.

- **Rotator**
  Der Rotator verwaltet das Drehverhalten sowie die Eingabe über die Meta-Quest 3 Joysticks. Dabei können beide Joysticks genutzt werden, um die Drehgeschwindigkeit anzupassen. Wir haben uns entschieden, nur Rotation um die y-Achse zu erlauben, da sonst durch das Controllerdesign des Meta Quest 3 Plus Controller keine saubere Rotation möglich ist (keine Einrastpunkte).

6.5 UI

Das UI basiert auf vielen Einzelteilen welche zusammen eine angenehme Nutzererfahrung bieten. Das UI lässt sich in drei Teile einteilen. Generell ist zu erwähnen das die Objekte in Unity genauso heißen wie die Skripts, d.h. wenn von Objekt A gesprochen wird ist auch das Skript A gemeint.

- **BaseModelSelector**
  Der BaseModelSelector ruft das zugehörige Skript auf und erstellt aufgrund diesem

dynamsich für jedes BaseModel in der Modellist einen Button. Er ist somit das Startmenü in welchem die einzelen Modelle ausgewählt werden können. Nach Auswahl eines Models wird er auf inaktiv gestellt. ChildModelSelector und die Canvas der Buttons werden dann auf aktiv gestellt.

- **ChildModelSelector**
  Der ChildModelSelector ist zuständig für die eigentliche Konfiguration. Im untergeordnet ist der PortSelector, der ChildModelSelector und der ColorSelector. Der PortSelector zeigt Optionen für alle Ports an die als Choosable in der ModelList markiert sind, also die Auswahl welches Teil des Autos man gerne ändern möchte. Der ChildModelSelector ist zuständig um die richtigen Auswahlmöglichkeiten für den jeweiligen Port graphisch darzustellen. Durch das Auswählen eines neuen ChildModels wird der ColorSelector aktiv. Durch diesen lassen sich die Farben der Modelle anpassen. Nach dieser Auswahl wird das ChildModel in der gewünschten Farbe im Model ersetzt.

- **CanvasButtons** Diese Canvas hat untergeordnet den ExplosionsButton, den ReturnButton, den ExportButton , den PassthroughButton und den RotatorSlider. Der ExplsosiosnButton ist zuständig für die Explosionsansicht des Modelles. Die einzelen Explosionsrichtigungen werden hierbei in der ModelList bei der Definitions der Ports den einzelnen ChildModels zugeordnet. Der ReturnButton führt zurück ins Hauptmenü. Der ExportButton ist zuständig für das Erstellen einer Json Datei der aktuellen Konfigauration. Der PassthroughButton schaltet zwischen AR und VR Modus um, hierbei wird das Skript AR Toggle Button verwendet, welches das vorgefertigte Passthrough Skirpt von Meta verwendet. Der RotatorSlider ist für die Drehung des Models zuständig. Hierbei kennt er Werte von 0 bis 360, sodass die der Drehwinkel genau eingestellt werden kann. Es handelt sich hierbei um einen unendlichen Slider, wenn am Ende angelangt springt er wieder auf 0 und dreht das Model weiter.

7 Test und Evaluation

Die Qualität und Funktionalität der Anwendung wurden im Rahmen interner End-to-End Tests überprüft. Ziel war es, die Stabilität, Usability und Performance auf der Meta Quest 3 sicherzustellen. Dabei kamen sowohl manuelle Testszenarien als auch explorative Tests durch das Team selbst zum Einsatz. Auch das Ausprobieren durch projektfremde Personen gab einen wertvollen Einblick wie User mit wenig technischer Erfahrung sich in unserer App zurechtfanden, wodurch wir wertvolle Tipps zur UI Gestaltung erhalten haben.

7.1 Testmethodik

Zur Evaluierung der Anwendung wurden verschiedene Testszenarien definiert, die typische Nutzerinteraktionen abbilden: Modellauswahl, Rotation, Komponentenkonfiguration, Aktivierung der Explosionsanimation und Wechsel in den AR-Modus. Dabei wurde besonderes Augenmerk auf die Reaktionsgeschwindigkeit der Benutzeroberfläche sowie auf das Verhalten bei mehrfacher Interaktion gelegt.
Die Tests wurden sowohl im Unity Editor als auch direkt auf der Meta Quest 3 als apk Build durchgeführt, um das Verhalten unter realen Bedingungen zu prüfen.

7.2 Usability und Benutzererlebnis

Die Anwendung wurde darauf ausgelegt, auch von VR-unerfahrenen Nutzenden intuitiv bedient werden zu können. Die klare Struktur der Benutzeroberfläche und der Einsatz von Standard-Interaktionen aus dem Meta SDK erleichterten die Orientierung in der VR-Umgebung. Die Möglichkeit zur freien Rotation und die direkte Rückmeldung bei Konfigurationsänderungen trugen zu einem positiven Benutzererlebnis bei.
Verbesserungspotenzial zeigte sich insbesondere in der Feinabstimmung der Steuerung, beispielsweise bei der Genauigkeit der Handeingaben, sowie bei der visuellen Darstellung der aktiven Auswahl.

7.3 Leistung und technische Stabilität

Die Performance der Anwendung wurde auf der Meta Quest 3 als zufriedenstellend bewertet. Die durchschnittliche Bildrate blieb im angestrebten Bereich, auch bei Nutzung der AR-Funktionalität und der Explosionsanimation. Um dies zu erreichen, wurden die verwendeten 3D-Modelle im Vorfeld optimiert indem wir nicht benötigte Komponenten entfernt haben.

7.4 Optimierungen

Basierend auf den Testergebnissen wurden gezielte Optimierungen vorgenommen:

- Vereinfachung der Materialstrukturen zur besseren Rendering-Performance.
- Verbesserung der Interaktions-Hitboxen zur präziseren Auswahl.

In zukünftigen Testreihen könnten gezielt Nutzerstudien mit Fokusgruppen durchgeführt werden, um fundierteres Feedback zur Usability und Wirkung des VR-Erlebnisses zu erhalten. Hier wäre besonders die Nutzergruppe wichtig, die auch später die App nutzen soll bzw. diese im EmRoLab zu zeigen bekommt.

8 Ergebnisse und Diskussion

Im Rahmen der Projektarbeit konnte ein funktionaler Prototyp eines dynamischen Fahrzeugkonfigurators für die Meta Quest 3 erfolgreich umgesetzt werden. Die Anwendung ermöglicht es, zwischen Modellen zu wählen und diese anschließend interaktiv in Virtual Reality zu konfigurieren. Darüber hinaus wurden zusätzliche Funktionen wie eine Explosionsanimation und ein AR-Modus integriert, die die Anwendung sowohl funktional als auch visuell erweitern.

8.1 Erfüllung der Projektziele

Die definierten Projektziele wurden vollständig erreicht. Der Konfigurator ist lauffähig auf der Meta Quest 3, bietet eine intuitive Steuerung und deckt alle grundlegenden Interaktionen ab, die zur Konfiguration eines Fahrzeugs notwendig sind. Die Kombination aus Auswahlbuttons, 3D-Darstellung, Interaktion und AR-Integration wurde erfolgreich umgesetzt.
Die Explosionsanimation bietet einen besonderen Mehrwert, da sie das Modell auf anschauliche Weise in seine Einzelteile zerlegt und somit einen zusätzlichen Lerneffekt oder Präsentationswert mit sich bringt.

8.2 Lernerfahrungen

Die Arbeit an dem Projekt ermöglichte einen tiefen Einblick in die Entwicklung von VR-Anwendungen. Insbesondere der Umgang mit XR-spezifischen Frameworks, die Optimierung von 3D-Modellen für Mobile-VR und die Gestaltung benutzerfreundlicher Interaktionen konnten in der Praxis vertieft werden. Die Komplexität des Zusammenspiels zwischen Performance, Usability und technischer Umsetzung war eine zentrale Herausforderung, aber auch eine wertvolle Lernchance.

9 Fazit und Ausblick

9.1 Zukünftige Erweiterungen

Für eine Weiterentwicklung des Konfigurators bieten sich verschiedene Ansatzpunkte:

- Erweiterung um weitere Fahrzeugmodelle und Konfigurationsoptionen.
- Integration realistischerer Materialien, Texturen und Beleuchtung zur Erhöhung der visuellen Qualität.
- Verbesserung der AR-Erfahrung durch gezieltes Platzieren von Modellen im Raum und Tracking-Anpassungen.
- Einbindung von Nutzerfeedback und systematischer Usability-Tests zur weiteren Optimierung der Benutzerführung.
- Speichern und Laden von Konfigurationen für spätere Betrachtung.

9.2 Abschließende Bewertung

Das Projekt hat gezeigt, dass Fahrzeugkonfiguration in Virtual Reality technisch realisierbar und nutzerfreundlich umsetzbar ist. Der entwickelte Prototyp stellt eine solide Grundlage dar, auf der zukünftige VR-Produktkonfiguratoren aufbauen können, sowohl im akademischen als auch im industriellen Kontext.