

VIRTUELLE REALITÄT IM KABINENSYSTEMENTWURF: ENTWICKLUNG EINER METHODIK ZUR INTEGRATION, OPTIMIERUNG UND VERNETZUNG VON KABINENMODELLEN FÜR DIE DARSTELLUNG IN UNREAL ENGINE

L. Fuszenecker, M. Fuchs, J. Biedermann, B. Nagel
Deutsches Zentrum für Luft- und Raumfahrt e.V., Institut für Systemarchitekturen in der Luftfahrt, Hein-Saß-Weg 22, 21129 Hamburg, Deutschland

Zusammenfassung

Die Luftfahrtindustrie steht vor einem Wandel, der das Potenzial birgt, das Verkehrsflugzeug nachhaltig zu verändern. Alternative Antriebstechnologien, die zur Erreichung der Klimaziele notwendig sind, sowie veränderte Passagiererwartungen in Bezug auf Nachhaltigkeit und Komfort treiben diese Entwicklung voran. Darüber hinaus erfordern disruptive Konzepte und neue Technologien für die Kabine innovative Methoden und Lösungen, um Effizienzsteigerungen im Entwurfsprozess zu erzielen und den Flugverkehr zukunftsfähig zu gestalten.

Eine zentrale Herausforderung ist der Ausbau der Digitalisierung des Kabinensystementwurfs im Sinne des Digitalen Fadens. Dafür müssen neue Methodiken und Modelle entwickelt werden, die eine end-to-end Vernetzung ermöglichen. Dadurch sollen frühzeitig Subsysteme und Komponenten in den Flugzeugentwurf eingebunden werden. Eine weitere Herausforderung liegt in der Interoperabilität, um Ergebnisse aus verschiedenen Fachbereichen zu verknüpfen und ganzheitliche Bewertungen neuer Technologien oder Konfigurationen zu ermöglichen. Die virtuelle Realität (VR) wird dabei als unterstützende Entwicklungsplattform eingesetzt. Mit Hilfe von VR ist es möglich, Wechselwirkungen und Synergien zwischen Systemkomponenten zu identifizieren und Konzepte frühzeitig zu bewerten. Durch immersive Analysen und das Vernetzen von Experten kann die Entwicklung beschleunigt werden. VR kann dabei weit über die geometrische Systemauslegung hinausgehen. Sie bietet das Potenzial, Simulationsergebnisse aus Analysen direkt mit der geometrischen Systemdarstellung zu verknüpfen. Die gewonnenen Erkenntnisse können so auch fachfremden Experten durch die immersive Visualisierung leicht zugänglich gemacht werden. Zudem eröffnet sich die Möglichkeit, in einem frühen Entwicklungsstadium Probandenstudien in VR durchzuführen. Die Integration von VR bringt jedoch Herausforderungen mit sich, da große Datenmengen und komplexe 3D-Modelle nicht direkt nutzbar sind. Deshalb sind optimierte 3D-Modelle, eine Reduktion der Modellgröße und automatisierte reproduzierbare Übertragungsverfahren zwischen den einzelnen Tools in der Prozesskette erforderlich, um eine effiziente Visualisierung zu ermöglichen.

In dieser Arbeit wird eine Methodik entwickelt, die auf der einen Seite die parametrische und modulare Auslegung von Kabinensystemen weiterentwickelt und auf der anderen Seite mit einer Visualisierung in VR über die Unreal Engine verbindet. Der Ansatz erweitert dabei bestehende automatisierte Auslegungsprozesse für Kabinensysteme am DLR. Anschließend wird eine Schnittstelle zur Unreal Engine geschaffen, die im letzten Schritt des Workflows eine Visualisierung und Interaktion der Kabine mit ihren Systemen in VR ermöglicht. So wird generiertes Wissen immersiv nutzbar gemacht und für Experten sowie Probandenstudien bereitgestellt. Ein zentraler Aspekt dieser Arbeit ist die entwickelte Schnittstelle sowie die optimierte 3D-Modellgröße, die die Interoperabilität zwischen verschiedenen Disziplinen sicherstellt, den Datenaustausch vereinfacht und die generierten Modelle effizient handhabbar gestaltet. Die Reduktion der 3D-Modellgröße ist dabei entscheidend, da große Datenmengen für VR-Anwendungen nachteilig sein können und eine optimierte Modellbasis für die Visualisierung von Vorteil ist. Durch eine Reduzierung der Datengröße besteht die Möglichkeit, die Fidelität der 3D-Objekte für die VR-Anwendung zu erhöhen, was einen Einfluss auf die Realitätsnähe sowie die Immersion der VR-Darstellung hat. Die beschriebene Methodik wird am Anwendungsfall eines disruptiven Kabinenkonzeptes demonstriert.

1. EINLEITUNG UND MOTIVATION

Die Luftfahrtindustrie steht durch alternative Antriebskonzepte, das Erreichen von Klimazielen und steigender Nachhaltigkeitsanforderungen aufgrund sich ändernder Passagierbedürfnisse vor neuen Herausforderungen. Disruptive und nachhaltige Kabinenkonzepte rücken aus den genannten Gründen weiter in den Vordergrund. Um solche disruptiven und nachhaltigen Kabinenkonzepte frühzeitig im Entwicklungsprozess bewerten zu können, müssen nicht nur heterogene Toolketten, sondern auch die dahinterstehenden Fachdisziplinen gekoppelt werden um eine digitale Durchgängigkeit zu schaffen. Durch diese

heterogene Toollandschaft kann es zu Medienbrüchen und unterschiedlichen Fidelitätsgraden der Modelle je nach Fachdisziplin kommen. Um die Virtuelle Realität als unterstützende Entwicklungsplattform einzusetzen muss eine hohe Fidelität der VR-Umgebung sichergestellt werden um die nötige Immersion herzustellen, zum Beispiel durch Objekte mit Texturen [1]. Das Problem bei einer hohen Fidelität sind die immer größer werdenden Modelle, die wiederum einen Einfluss auf die Performance der VR-Szene haben [2]. Die entwickelte Methodik soll diese Probleme adressieren, indem eine automatisierte Schnittstelle zur Unreal Engine geschaffen wird, die gleichzeitig die Datengröße der Modelle reduziert, um so die digitale Durchgängigkeit von den Vorentwurfsdaten bis zur Dar-

stellung in VR zu ermöglichen.

VR entwickelt sich von einer reinen Visualisierungstechnologie zu einer kollaborativen Entwicklungsplattform. Jenseits der bloßen Darstellung von Geometrie ermöglicht sie die unmittelbare, räumliche Erfahrung von Kabinenkonzepten. Varianten lassen sich schneller vergleichen, Abstände und Sichtbeziehungen intuitiv prüfen, und Analysen können direkt an sichtbaren Objekten diskutiert werden. Damit dieser Mehrwert entsteht, genügt eine reine „Visualisierung“ nicht. Es ist eine Methodik erforderlich, die parametrisch erzeugte Modelle samt Systembezügen konsistent und performant in VR überführt und dabei Identifikatoren, Konfigurationsparameter und Analysewerte bewahrt. Erst dann kann VR als unterstützende Entwicklungsplattform im Sinne von Co-Design eingesetzt werden, sodass verschiedene Fachexperten kollaborativ in einer gemeinsamen Umgebung zusammenarbeiten können.

Die vorliegende Arbeit entwickelt eine solche Methodik und wendet sie an einem disruptiven Kabinen-Konzept aus dem DISKUS-Projekt an. Das Ziel des Verbundvorhaben DISKUS (Disruptive Kabine für ein umweltfreundliches Lufttransportsystem) ist eine deutliche Reduktion der Klimawirkung der Passagierkabine durch Gewichtsreduktion. Diese Reduktion soll durch eine radikale Abkehr von heutigen Auslegungsstandards erreicht werden. Darüber hinaus werden kreislauffähigere Materialien angewendet und Methoden für die weitere Verkürzung von Entwurfsprozessen erforscht. Ausgangspunkt sind bestehende, etablierte, parametrisierte Entwurfsalgorithmen und Workflows im DLR-Umfeld. Darauf aufbauend wird ein methodischer Ansatz ausgearbeitet, der die Modellgenerierung, die Geometrieaufbereitung in Blender und die interaktive Darstellung in der Unreal Engine so verbindet, dass das als Fallstudie verwendete DISKUS-Kabinenkonzept vollständig und nachvollziehbar in VR erlebbar wird (siehe Abbildung 1).

Dafür wird eine wissensbasierte Methode entwickelt, die eine für das Konzept angepasste Belüftungsarchitektur erzeugt und im Datenmodell verankert. Die Methode beschreibt unter anderem die Aufteilung der Zuluftleitungen, die Führung entlang struktureller Bereiche sowie die Kopplung an relevante Kabinenzonen. Durch die wissensbasierten Methoden und den parametrischen Ansatz können die Systementscheidungen beeinflusst werden. Dadurch lassen sich verschiedenste Systemarchitekturvarianten der Belüftung erzeugen, in VR inspizieren und hinsichtlich ihrer Bauraumbezogenen Auswirkungen diskutieren.

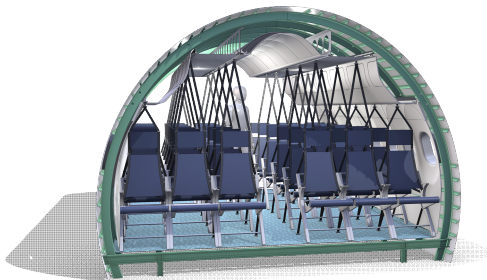


BILD 1. DISKUS-Kabinenkonzept (Sitze von CompriseTec, Traversenansatz von Diehl Aviation)

Die in dieser Arbeit entwickelte Methodik adressiert drei zusammenhängende Problemfelder. Erstens die Integration der heterogenen Toolkette, dabei wird die Parametrische Auslegung in MATLAB, die Aufbereitung in Blender und der Import in Unreal zu einem durchgängigen Ablauf verbunden. Objekt-IDs und Beziehungen zwischen Komponenten bleiben über die verschiedenen Tools hinweg erhalten. Das zweite Problem ist die Optimierung der Modelle für die VR-Darstellung. Hierzu gehören die konsequente Nutzung von Instanzen statt duplizierter Geometrie, eindeutige Namen und Ordnungsregeln, die Reduktion der Polygonzahl sowie praktikable Materialstrategien für 3D Kabinenteile, damit die Darstellung konsistent bleibt und die angestrebten Bildraten für VR erreicht werden. Das dritte Problem ist die Kopplung von Geometrie und Systeminformation, die entwickelte Belüftungsarchitektur dient als Beispiel, wie wissensbasierte Regeln in eine Parametrik übersetzt und in der VR-Szene überprüfbar gemacht werden können.

Die Zielsetzung reiht sich in die strategische Ausrichtung des DLR [3] und der Luftfahrtstrategie der Bundesregierung [4] ein. Beide setzen auf klimaverträgliche Transformation, kürzere Entwicklungszyklen und den systematischen Einsatz digitaler, modellbasierter Methoden. Im DLR-Kontext steht eine durchgängige und nachvollziehbare Prozesskette im Mittelpunkt, von der parametrischen Erzeugung bis zur bewertbaren Darstellung. Die nationale Strategie betont zudem vernetzte Zusammenarbeit und virtuelle Testumgebungen, um Risiken früh zu erkennen und Ressourcen zu schonen.

Die digitale Durchgängigkeit senkt manuellen Integrationsaufwand. Die Aufbereitung der 3D Modelle zielt auf stabile Bildraten in VR ab, ohne die Lesbarkeit der Systeme zu verlieren. Die Kopplung von Parametern und Analysewerten kann die VR-Szene zu einem Arbeitsraum für disziplinübergreifende Reviews machen, in dem sich Entscheidungen auf eine gemeinsame, anschauliche Basis stellen lassen. Das DISKUS-Anwendungsbeispiel zeigt, dass die Methodik gerade dort wirken kann, wo disruptive Konzepte und neue Architekturen entstehen und früh validiert werden müssen.

Daraus leiten sich zwei Forschungsfragen ab:

- 1) Wie lässt sich eine interoperable Pipeline gestalten, die parametrische Kabinenmodelle mitsamt Systemlogik konsistent und performant in VR darstellt.
- 2) Welchen quantifizierbaren Nutzen liefern Methoden wie Instanziierung und gezielte Modellreduktion gegenüber naiven Duplikaten, gemessen etwa an Bildrate, Speicherbedarf, Importzeit und Draw-Calls.

Die folgenden Kapitel beschreiben den Stand der Technik, das methodische Vorgehen und die Umsetzung der DISKUS-Konzeptkabine durch die entwickelte Methodik. Abschließend werden Grenzen und nächste Schritte skizziert, etwa die weitere Anbindung zusätzlicher Simulationsdomänen oder notwendige Funktionalität innerhalb der VR-Anwendung.

2. STAND DER TECHNIK

Das folgende Kapitel gibt einen Überblick über den Stand der Technik. Dabei werden bestehende Methoden zur parametrischen Systemauslegung und zur VR-gestützten Bewertung von Kabinenkonzepten zusammengefasst und in den aktuellen Forschungskontext eingeordnet.

Parametrische Entwurfswerkzeuge und Methoden zur Kabinensystemauslegung

Parametrische Entwurfswerkzeuge bilden im frühen Flugzeugentwurf die Grundlage für Variantenbildung und Schnittstellen ermöglichen Datenkonsistenz sowie Tool-Interoperabilität. Das Common Parametric Aircraft Configuration Schema (CPACS) dient dabei als, XML-basiertes Austauschschema, um Konfigurationen des Lufttransportsystems semantisch und hierarchisch zu beschreiben und zwischen Disziplin-Tools auszutauschen [5][6]. Mit CPACS v3.5 rücken systemspezifische Design-Parameter in den Fokus, sodass Systemarchitektur-Entscheidungen früher in den Gesamtentwurf rückgekoppelt werden können [6]. FUGA (Fuselage Geometry Assembler) ist ein wissensbasiertes Entwurfswerkzeug welches am DLR entwickelt wurde. Mit Hilfe von FUGA lassen sich aus Vorentwurfsdaten geometrische Modelle von Flugzeugrumpfen und der dazugehörigen Kabinen erzeugen, die für weitergehende Analysen verwendet werden können [7]. Weitere von Universitäten, der Industrie oder Forschungsinstituten entwickelte Tools sind zum Beispiel das Vehicle Sketch-Pad der NASA [8], PADlab von der TU Berlin [9], GeneSys der TUHH [10] oder das open source boarding-Simulationstool PAXelerate von Bauhaus Luftfahrt [11].

In der Literatur finden sich methodische Ansätze, mit denen Kabinensystemarchitekturen auf Basis von Vorentwurfsdaten konzipiert und variiert werden können. Fuchs et al. integrieren hierfür eine modellbasierte Systementwicklung mit Methoden des Model-based Systems Engineerings (MBSE) und eine funktionale Systemarchitektur in den konzeptionellen Kabinensystementwurf, inklusive Anbindung an eine virtuelle Designplattform zur frühzeitigen Bewertung der Kabinensystemkonfigurationen [12-15]. In der Arbeit von Bielsky et al. wird eine Methode vorgestellt, mit der das elektrische Versorgungssystem eines Flugzeugs parametrisch ausgelegt und unter Einsatz des GeneSys-Frameworks modelliert wird [16]. Diese Ansätze zeigen, dass Systemauslegung bereits im Vorentwurf sinnvoll mit Geometriedaten verknüpft werden kann, gleichzeitig zeigen sie den Bedarf eines durchgängigen Parametertausch zwischen den Modellen bis hin zur interaktiven, performanten Visualisierung auf.

Visualisierung und Geometrieaufbereitung

Zur automatisierten Geometrieerzeugung aus CPACS heraus liegen Methoden vor, die eine Kabinen 3D-Geometrie für immersive Anwendungen generieren. Walther et al. zeigen eine Methode, die aus Vorentwurfsdaten detaillierte Rumpf- und Kabinengeometrie ableitet, um die Bewertung in VR bereits früh zu ermöglichen [17][18]. Parallel dazu existieren Toolketten (z. B. PAXelerate bei Bauhaus Luftfahrt), die CPACS-Dateien in automatisierte 3D-Kabinengenerierung überführen und so Renderings bzw. VR-Voransichten im Vorentwurf unter-

stützen [19][20]. Dennoch bleibt es hier nur bei VR-Voransichten ohne Funktionalität innerhalb der Szene.

Virtual Reality im Kabinendesign

Virtual Reality wird als Technologie eingesetzt, um Kabinenlayouts, Ergonomie und Passagiererlebnisse frühzeitig zu evaluieren. Studien belegen, dass VR nicht nur Präsentationszwecken dient, sondern Iterationen beschleunigt und Nutzende in frühe Phasen einbindet. Durch das zusammenarbeiten verschiedener Stakeholder kann in frühen Entwicklungsphasen kosteneffektiv das Design in VR erlebt und untersucht werden. So können frühzeitig Fehler entdeckt und Komforterfahrungen gesammelt werden, bevor ein reales Mockup gefertigt wird [21][22]. Dabei können sowohl visuelle als auch interaktive Bewertungen adressiert werden, so können geometrische Anforderungen überprüft oder zusammenhängende Komponenten dargestellt werden [23].

Performance und Instanziierung in VR-Entwicklungsumgebungen (Unreal)

Für große, wiederholte Objektmengen gilt Instanziierung als Grundpfeiler der Performance. In der Entwicklungsumgebung Unreal Engine bündeln Instanced Static Meshes (ISM) Draw-Calls und reduzieren den UObject-Overhead. Hierarchical ISM (HISM) ergänzt dieses Vorgehen weiter durch das Clustern von Instanzen [24]. Praxisberichte verweisen zugleich auf Grenzen z. B. eine sehr hohe Anzahl an Instanzen und Aufbaukosten beim Hinzufügen von Instanzen, sodass disziplinierte Asset- und Material-Regeln die notwendige Basis bilden [25][26].

Für statische oder teil-statische Anteile bleibt gebackene Beleuchtung relevant, vorausgesetzt sind saubere Lightmap-UVs und angemessene Lightmap-Auflösungen. Die offiziellen Unreal-Richtlinien adressieren typische Fehlerbilder wie Overlapping UVs [27]. Ergänzend fassen XR-Best-Practices und Hersteller-Guides (Meta) praxistaugliche Performance-Regeln zusammen, etwa zum Umgang mit Transparenzen, Schatten und stabilen Frametimes in VR [28][29].

2.1. Einordnung in den Stand der Technik

Der Stand der Technik zeigt, dass bereits wesentliche Grundlagen für den digitalen Kabinensystementwurf existieren. Mit CPACS steht ein etabliertes, parametrisches Austauschformat zur Verfügung, das Flugzeugkonfigurationen konsistent beschreibt und insbesondere im Vorentwurf eine gemeinsame Datengrundlage für unterschiedliche Disziplinen bietet. Darauf aufbauend wurden in verschiedenen Forschungsarbeiten Methoden entwickelt, die erste Kabinensysteme beschreiben und aus denen sich funktionale Architekturen ableiten lassen können. Ebenso existieren Verfahren, mit denen aus diesen abstrakten Daten 3D-Geometrien abgeleitet werden, die wiederum in VR-Umgebungen zur Bewertung von Raumgefühl und Layouts herangezogen werden können. Gleichzeitig sind die Methodiken oft limitiert und müssen erweitert werden. CPACS beinhaltet zwar Informationen zur Rumpfstruktur und Kabinenobjekten, wie Spante oder Sitze. Es fehlen aber noch Methoden um Kabinensysteme standardisierte Auszulegen und in CPACS abzulegen. Systemauslegungen werden oft auf Grundlage von CPACS Vorentwurfsda-

ten in MATLAB o.ä. Werkzeugen dargestellt und enden meist auf einer funktionalen Ebene, es fehlt die automatische Anbindung an eine performante, immersive Visualisierung. Umgekehrt ermöglichen VR-Anwendungen heute bereits eine realistische Darstellung sowie Nutzerbewertungen. Sie basieren jedoch oft auf statischen Geometriemodellen ohne Anbindung an parametrische Entwurfsdaten und ohne die Möglichkeit, Varianten methodisch nachzuvollziehen. Auch die Frage der technischen Robustheit ist bisher nur unzureichend beantwortet. Viele Ansätze erfordern manuelle Zwischenschritte, etwa beim Transfer von Geometriedaten, bei der Aufbereitung von Modellen oder beim Einrichten von VR-Szenen. Dadurch entstehen Inkonsistenzen, die Variantenvergleiche erschweren und die Reproduzierbarkeit limitieren.

Ausgehend von diesen identifizierten Grenzen zeigt sich, dass eine durchgängige, automatisierte Prozesskette, die parametrische Entwurfsdaten, funktionale Systemauslegung, Geometrieaufbereitung und VR-Integration durchgehend digital verbindet, fehlt. Erst dadurch können disruptive Kabinenkonzepte modelliert und in VR erlebbar gemacht werden.

3. METHODISCHES VORGEHEN

Ziel der hier entwickelten Methodik ist es, eine durchgängige automatisierte Prozesskette vom parametrischen Systementwurf bis hin zur performanten Darstellung in der Unreal Engine bereitzustellen (Abbildung 2). Damit soll eine methodische Grundlage geschaffen werden, die den gesamten Ablauf von der Erzeugung der Entwurfsdaten über die Geometrieaufbereitung bis zur immersiven VR-Visualisierung konsistent miteinander verbindet. Im Vordergrund stehen dabei drei Aspekte: die Rückverfolgbarkeit von Informationen über alle beteiligten Tools hinweg, indem IDs (und Metadaten) konsequent mitgeführt werden; die technische Robustheit, indem manuelle Zwischenschritte vermieden und reproduzierbare Abläufe etabliert werden; sowie die VR-Tauglichkeit, die bereits für vertiefende Analysen neuartiger disruptiver Kabinenkonzepte gewährleistet sein muss.

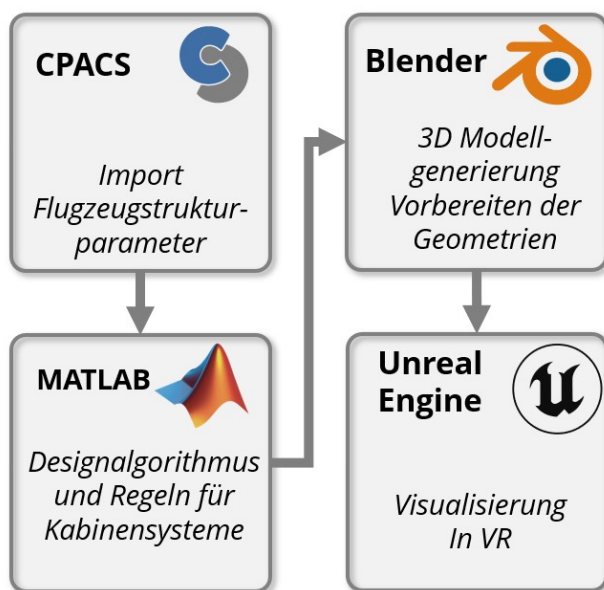


BILD 2. Automatisierte Prozesskette

3.1. Parametrische Grundlage

Die Datenbasis bildet CPACS als Schnittstellenformat und single-source-of-truth aus dem Flugzeugvorentwurf. Es beinhaltet eine parametrische Beschreibung von Flugzeugkonfigurationen und ermöglicht den Austausch von Informationen für das Flugzeug. Für den hier verfolgten Anwendungsfall sind insbesondere die in CPACS hinterlegten Kabinenobjekte sowie Rumpfinformationen wie die Spanten von Bedeutung, da sie die Grundlage für die weitere Systemauslegung in MATLAB bilden.

Kabinen- und Rumpfbjekte sind in CPACS strukturiert abgelegt und mit geometrischen Abmessungen hinterlegt. Für die Auslegung der Kabinensysteme werden daher in MATLAB weitere Parameter für die Systeme definiert und mit den aus CPACS übernommenen Positions- und Transformationsinformationen kombiniert. Auf dieser Basis erzeugt der Entwurfsalgorithmus Bounding-Box-Repräsentationen, die als robuste Platzhalter für die weitere Systemauslegung dienen. Diese Bounding-Boxen ermöglichen Bauraumabschätzungen, Kollisionsprüfungen und bilden stabile Ankerpunkte für die systematische Positionierung neuer Kabinensysteme.

Aktuell sind in CPACS keine Informationen zu Kabinensystemen wie zum Beispiel Belüftung, Elektrik oder Abwasser/Wasser hinterlegt. Als methodischer Bezugspunkt dient die in der Literatur beschriebene Vorgehensweise [12]. Das hier entwickelte Verfahren knüpft daran an, erweitert es jedoch, um auch disruptive Kabinenkonzepte wie die DISKUS-Konzeptkabine abzubilden. Aufgrund der veränderten Bauraum- und Verkleidungskonfiguration des zu untersuchenden disruptiven Kabinenkonzepts wird eine neue parametrische Belüftungsarchitektur entwickelt, die die Zuluftaufteilung, die Führung entlang struktureller Bereiche und die Zonenanbindung beschreibt. Auf diese Weise lassen sich Varianten derselben Grundkonfiguration konsistent erzeugen und vergleichen.

Alle in MATLAB vom Entwurfsalgorithmus gesetzten Objekte und deren Informationen (Typ, ID, Transform, etc.) werden in einer XML strukturiert abgelegt. Dieses Vorgehen erhält Semantik und Hierarchie, macht die Informationen maschinenlesbar und etabliert ein Austauschformat im Sinne von CPACS für nachfolgende Werkzeuge.

3.2. Geometrieaufbau in Blender: Platzierung, Konsolidierung und Instanziierung

Ein Python-Skript startet Blender im Batch-Modus, lädt die generierte XML-Datei ein, liest pro Eintrag Objekttyp und Transform und platziert das passende 3D-Modell aus einer FBX-Bibliothek mit bereits hinterlegten Grundmaterialien.

Zunächst werden die platzierten Modelle als Hard-Copies angelegt. Anschließend folgt die Instanziierung gleicher Objekte, um die genannte Herausforderung der Performance zu überwinden. Das Verfahren basiert auf einer topologischen Gruppierung. Objekte mit gleicher Anzahl an Eckpunkten, Kanten und Polygonen werden als Kandidaten einer Objektfamilie zusammengefasst. Innerhalb dieser Gruppen erfolgt ein toleranzgestützter Vergleich der normalisierten Geometrie, ergänzt durch eine Prüfung auf gespiegelte Varianten entlang der Hauptachsen (X, Y, Z).

So lassen sich Objekte sicher als identisch klassifizieren, auch wenn sie im Raum unterschiedlich platziert oder gespiegelt sind.

Erkannte Duplikate werden durch Instanzen eines gemeinsamen Mesh-Assets ersetzt. Die Welt-Transforms wie Position, Rotation und Skalierung bleiben dabei erhalten. Originale und Instanzen werden in Collections (Sammlungen) strukturiert, wobei je Original eine eigene Collection entsteht. Diese Organisation ist nicht kosmetisch, sondern steuert die Exportlogik. Pro Collection wird eine FBX-Szene erzeugt. Das ist entscheidend, damit die nachfolgende Importkette in Unreal die Hierarchie, die Material-Slots und vor allem das Verhältnis „ein Original, viele Platzierungen“ explizit wiederfinden kann.

Idealerweise sind die in der FBX-Bibliothek hinterlegten Modelle bereits so aufbereitet, dass ihre Normalen konsistent sind. Dadurch wird vermieden, dass später in Unreal ein Recompute durchgeführt werden muss. Diese Qualitätssicherung in der Bibliothek stellt sicher, dass die Meshes bereits beim Import korrekt interpretiert werden und nachträgliche Arbeitsschritte einspart.

3.3. Unreal-Integration, Post-Processing und Beleuchtung

Die in Blender vorbereiteten Collections werden automatisiert in ein Unreal-Template-Projekt importiert. Zentrale Rolle spielt hierbei die *FBXSceneImportFactory* [30], da nur sie die exportierte Szenen-Struktur vollständig auswertet und semantisch korrekt in Unreal überführt. Auf diese Weise entsteht pro Collection eine explizite Aufteilung in ein *Static-Mesh-Asset* für das jeweilige Original, eine *Blueprint-Class* (*FbxScene_Collection_**), die sämtliche Instanz-Platzierungen enthält, sowie die zugehörigen *Material-Assets* und die *FBX Scene Import Data* als Metadatenobjekt. Blueprints [31] sind in Unreal Engine ein visuelles Skriptsystem, das es erlaubt, Logiken, Abläufe und Interaktionen ohne klassischen Code zu implementieren. Sie stellen eine grafische Oberfläche bereit, in der Funktionen und Variablen über Knoten („Nodes“) miteinander verbunden werden. Auf diese Weise können komplexe Verhaltensweisen wie Objektinteraktionen, Animationen oder Benutzerinteraktionen direkt in der Engine abgebildet werden. Blueprints dienen damit als Brücke zwischen der reinen Geometriedarstellung und der funktionalen Ebene einer Anwendung. Entscheidend ist, dass Unreal die in Blender angelegten Instanzen erkennt und als solche abbildet. Anstatt redundante Mesh-Kopien zu erzeugen, wird ein einziges Original-Mesh als *Static-Mesh-Asset* importiert und mehrfach referenziert. Damit bleibt die Instanz-Semantik konsistent erhalten und bildet die Grundlage für eine effiziente Umsetzung mit *Instanced Static Meshes* (ISM). Die semantische Durchgängigkeit ist für die spätere Performance in VR maßgeblich.

Instanced Static Meshes (ISM) und *Hierarchical Instanced Static Meshes* (HISM) sind spezielle Komponenten in der Unreal Engine, die auf Effizienz beim Rendern abzielen. Beide erlauben es, ein einziges Mesh-Asset vielfach zu platzieren, ohne dass für jede Platzierung ein eigenes Objekt mit voller Datenlast entsteht. Bei ISM werden alle Instanzen eines Objekts in einem kompakten *Render-Call* verarbeitet, wodurch die Zahl der *Draw-Calls* deutlich sinkt. HISM erweitert dieses Prinzip um eine hierarchische

Struktur: Instanzen werden zusätzlich in räumlichen Clustern organisiert, sodass die Engine abhängig vom Sichtfeld und der Distanz nur die tatsächlich relevanten Cluster rendert. In der Praxis bedeutet dies, dass große Mengen wiederkehrender Objekte, etwa Sitze, Paneele oder Spannten in der Kabine, mit minimalem Rechenaufwand dargestellt werden können. Für VR, wo eine stabile *Frametime* entscheidend ist, stellt diese Technik einen zentralen Performancehebel dar.

Unmittelbar nach dem Import greift ein *Post-Processing-Skript* (Nachbearbeitungs-Skript), das keine Inhalte verändert, sondern gezielt Editor-Eigenschaften setzt. Dazu gehört insbesondere die Aktivierung von *Two-Sided-Material*. Sie ist notwendig, da die Kabine in vielen Bereichen aus einlagigen Schalenmodellen besteht, die andernfalls von einer Seite durchsichtig erscheinen würden (*Backface-Culling*). Ein weiterer Grund liegt in der Qualität der Ausgangsmodelle aus Blender. Einzelne Bereiche zeigten dort falsch orientierte Normalen, die im Import zu optisch fehlerhaften, „unsichtbaren“ Flächen führten. Durch die *Two-Sided-Einstellung* wird sichergestellt, dass auch diese Geometrien in der VR-Szene korrekt sichtbar bleiben. Ergänzend wird der *Reversed Index Buffer* aktiviert, um die Schattierung auf *Two-Sided-Geometrie* zu stabilisieren. *Remove Degenerates* wird bewusst deaktiviert, damit sehr schmale Dreiecke an offenen Kanten nicht entfernt werden.

Für die Behandlung der Normalen gilt aktuell folgende Regel. Die beim Export aus Blender mitgegebenen Normalen werden beim Import in Unreal übernommen. Damit bleibt die in Blender kontrollierte Geometrie- und Shading-Definition erhalten. Alternativ könnte nach dem Import ein konsistentes *Recompute* (*MikkTSpace*) angewandt werden, das sämtliche Werte einheitlich neu berechnet. Dieser Schritt ist technisch möglich, stellt aber keinen Standard dar. Im Idealfall sind die exportierten FBX-Modelle bereits so vorbereitet, dass ein *Recompute* in Unreal nicht erforderlich ist.

Darüber hinaus wird zur besseren Iteration eine reproduzierbare Beleuchtung aus drei rechteckigen Lichtquellen automatisiert erzeugt. Zwei Lichtquellen werden dabei auf Höhe der PSU (*Passenger Service Units*) in der Kabine positioniert und in Richtung Kabinenboden ausgerichtet; eine dritte befindet sich mittig leicht unterhalb der Deckenverkleidung. Große Quell-Flächen sorgen für gleichmäßige Ausleuchtung, und deaktivierte Schatten verhindern harte Kanten und Rauschmuster. Vorhandene Beleuchtungen mit identischem Präfix werden vorab entfernt, sodass die Szene stets in einem definierten Ausgangszustand vorliegt. Für den VR-Betrieb werden die Fülllichter, wo möglich, auf *Statisch* oder *Stationär* gesetzt.

3.4. Performance und Reproduzierbarkeit

Die für VR essenziellen Performancegewinne werden im Wesentlichen vor dem Rendern in der Engine erreicht. An erster Stelle steht die konsequente Instanziierung statt reiner Duplikate, die den Einsatz von *Instanced Static Meshes* (ISM) ermöglicht und die Zahl der *Draw-Calls* sowie den *UObject-Verwaltungsaufwand* (Unreal Objekt) deutlich reduziert. Ergänzend sorgt eine geregelte Materialstrategie dafür, dass Shaderkosten niedrig bleiben, bevorzugt werden *opaque* oder *masked* Materialien ein-

gesetzt. Two-Sided sollte idealerweise nur dort eingesetzt werden, wo es funktional unvermeidbar ist. Auf diese Weise wird verhindert, dass die Vielzahl dünnwandiger Schalenmodelle in der Kabine zu unnötiger Renderlast führt.

Ebenso wie die Performance ist die Reproduzierbarkeit der gesamten Prozesskette wichtig. Alle Stufen, von der XML-Erzeugung im MATLAB-Entwurfsalgorithmus über den Aufbau und die Instanziierung in Blender bis hin zum Szenen-Import, der Nachbearbeitung und Beleuchtung in Unreal, sind vollständig automatisiert. Dadurch entstehen deterministische Abläufe, bei denen Benennungen, Ordnerstrukturen und IDs konsistent bleiben. Varianten der Kabine oder der Belüftungsarchitektur können so systematisch erzeugt, belastbar miteinander verglichen und nahtlos in den Systementwurf zurückgespiegelt werden. Auf diese Weise wird nicht nur die technische Robustheit sichergestellt, sondern auch die Nachvollziehbarkeit wissenschaftlicher Ergebnisse gewährleistet.

3.5. Fallbeispiel: DISKUS-Kabine

Als Fallbeispiel zur Anwendung der vorgestellten Methodik dient das DISKUS-Kabinenkonzept wie in Abbildung 1 dargestellt. Grundlage für den Flugzeugentwurf bilden die Parameter aus einer CPACS-Datei (Spanten, Ausgänge, Sitze). Auf dieser Basis platziert der Entwurfsalgorithmus die Kabinenelemente automatisch und bildet sie, wie beschrieben, in MATLAB über Bounding-Boxes ab. Um ein so disruptives Konzept abbilden zu können, wurde der Entwurfsalgorithmus im Rahmen der Arbeit entsprechend angepasst.

Das Ziel des DISKUS-Kabinenkonzept ist es die Massen durch ein neuartiges Sitzkonzept und neue gedachte Systemarchitekturen wie zum Beispiel die Belüftung, zu reduzieren. Dabei sollen neue Materialien zum Einsatz kommen die auch recyclefähig sind um die Nachhaltigkeit zu erhöhen. Die DISKUS-Kabine verfolgt dabei den Gedanken, die Flugzeugkabine „*einfach mal*“ neu zu denken. Im Falle des DISKUS-Kabinenkonzept ergibt sich dadurch ein anderes Layout als in klassischen Single-Aisle-Konfigurationen (vgl. Abbildung 2):

- Anstelle der üblichen Kombination aus Seitenwand-, Licht- und Dado-Paneel wird ein Seitenwand- und Dado-Paneel ohne Lichtpaneel verbaut.
- Durch das spezielle Sitzkonzept entfällt eine klassische Gepäckablage („Overhead Stowage Compartment“, OHSC). Stattdessen ist lediglich eine kleine Gepäckablage ohne integrierte Passagier-Serviceeinheit („Passenger Service Unit“, PSU) vorgesehen.
- Das Sitzkonzept erfordert zudem Montagepunkte im Decken-Bereich. Zu diesem Zweck werden im oberen Decken-Bereich Traversen installiert, die als Befestigungspunkte für die Sitze dienen. Pro Gangseite sind jeweils zwei dieser Traversen vorgesehen.
- Zwischen den Traversen einer Seite wird ein Paneel eingebaut, während im Bereich über dem Gang zwischen den inneren Traversen das Decken-Paneel montiert wird.

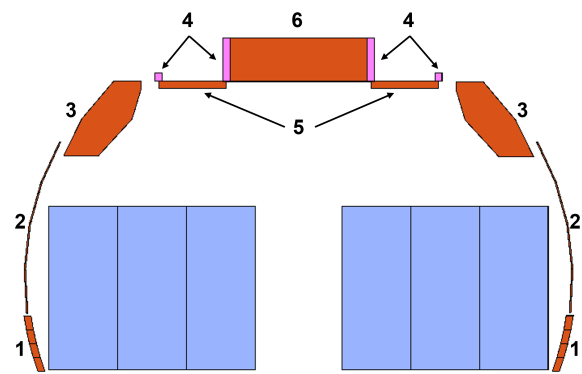


BILD 3. MATLAB Plot angepasstes Layout, zu sehen sind: 1 Dado-Paneel, 2 Seitenwand-Paneel, 3 Gepäckablage, 4 Traversen, 5 Paneel, 6 Decken-Paneel

Darauf aufbauend werden die Bounding-Box-Repräsentationen der Kabinenobjekte generiert (Abbildung 3 und 8). Diese bilden die Grundlage für Bauraumabschätzungen und die systematische Platzierung weiterer Komponenten. Durch diese Abweichungen verändert sich gegenüber einem klassischen Single-Aisle-Layout das gesamte Bauraumverhalten innerhalb der Kabine. Dies wirkt sich insbesondere auf die Platzierung der Systeme und deren Auslegung aus. Exemplarisch wird daher die Belüftungsarchitektur betrachtet. Auf Basis der geänderten Bauraumannahmen erfolgt eine Anpassung der gesamten Architektur. Klassische Belüftungskonzepten für Kurz- und Mittelstreckenflugzeuge, verteilen die Luft von der Mixereinheit („Mixer Unit“, MU) unter dem Kabinenboden entlang des Rumpfes und anschließend durch Zuluftleitungen nach oben.

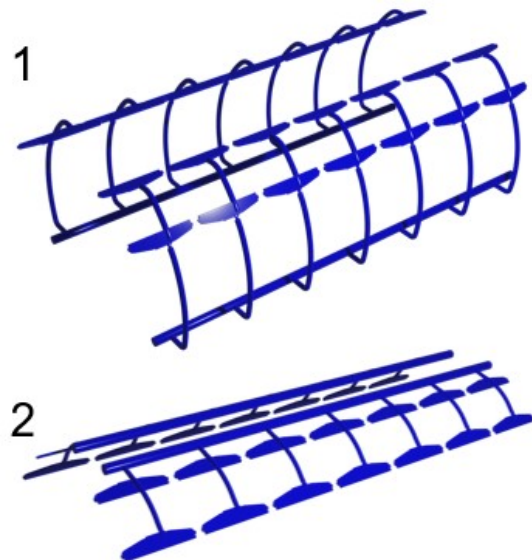


BILD 4. 1 Klassische Belüftungsarchitektur
2 DISKUS Belüftungsarchitektur

Im Gegensatz dazu wird für das DISKUS-Kabinenkonzept eine zentrale Belüftungsarchitektur aus dem oberen Bereich der Kabine betrachtet. In Abbildung 4 sind die beiden Varianten der Belüftungsarchitektur schematisch zu erkennen. Abbildung 5 zeigt die Belüftungsarchitektur für das DISKUS-Kabinenkonzept in MATLAB repräsentiert durch die Bounding-Boxen.

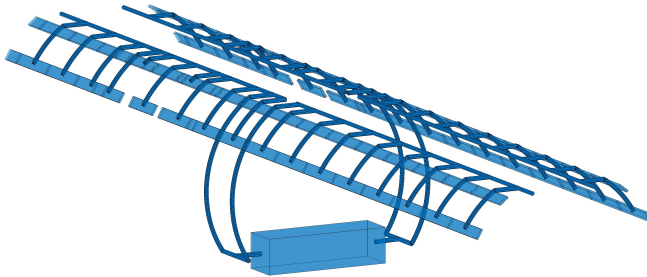


BILD 5. MATLAB Plot: DISKUS Belüftungsarchitektur

Die dem Bauraum angepasste Architektur befördert die Luft direkt von der Mixereinheit hinter der Verkleidung durch mehrere Zuluftleitungen in den oberen Deckenbereich; in diesem Beispiel wird die Luft dann in dem durch das Konzept entstandenen neuen Bauraum über Verteilerkanäle längs des Rumpfes verteilt. Von diesen Verteilerkanälen gehen Falleitungen ab, die die Luft zu den jeweiligen Auslässen transportieren wie in Abbildung 5, 6 und 8 zu erkennen ist. Die in MATLAB erzeugte XML dient anschließend als Input für die Geometrieaufbereitung in Blender. Dort werden die entsprechenden FBX-Modelle aus der 3D Modell Bibliothek platziert und nach der Instanziierung konsolidiert. Abbildung 6 zeigt die resultierende Kabinendarstellung, in der gleiche Objekte bereits durch Instanzen ersetzt wurden, um die spätere Engine-Performance vorzubereiten.

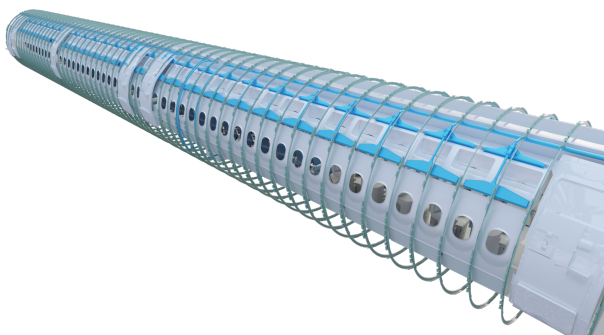


BILD 6. Blender Rendering der erzeugten Kabine mit der neuen Belüftungsarchitektur

Die nach Collections sortierten und exportierten FBX-Szenen werden schließlich automatisiert in Unreal importiert. Vor dem Import wird ein Template Projekt geöffnet in der grundlegenden Funktionalität für die VR-Szene bereitgestellt wird. Die importierten FBX-Szenen erscheinen dort als konsistente Mesh-Assets und Blueprint-Instanziierungen (Abbildung 7). In dieser Darstellung ist erkennbar, dass die ursprüngliche Struktur aus Blender (Collection, Hauptobjekt, Material und Instanzen) erhalten bleibt und für VR-Anwendungen reproduzierbar verfügbar ist.

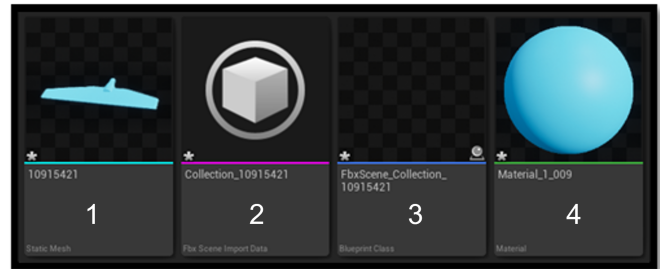


BILD 7. FBX Import Hierarchie für ein Objekt in Unreal: 1 Static Mesh Asset, 2 Fbx Scene Import Data, 3 Blueprint Class, 4 Material Asset

Das Post-Processing-Skript setzt die projektspezifischen Editor-Eigenschaften und erzeugt die definierte Beleuchtung. Das Ergebnis ist eine aus dem parametrischen Entwurf abgeleitete Szene, die konsistent in VR dargestellt werden kann (Abbildung 9).

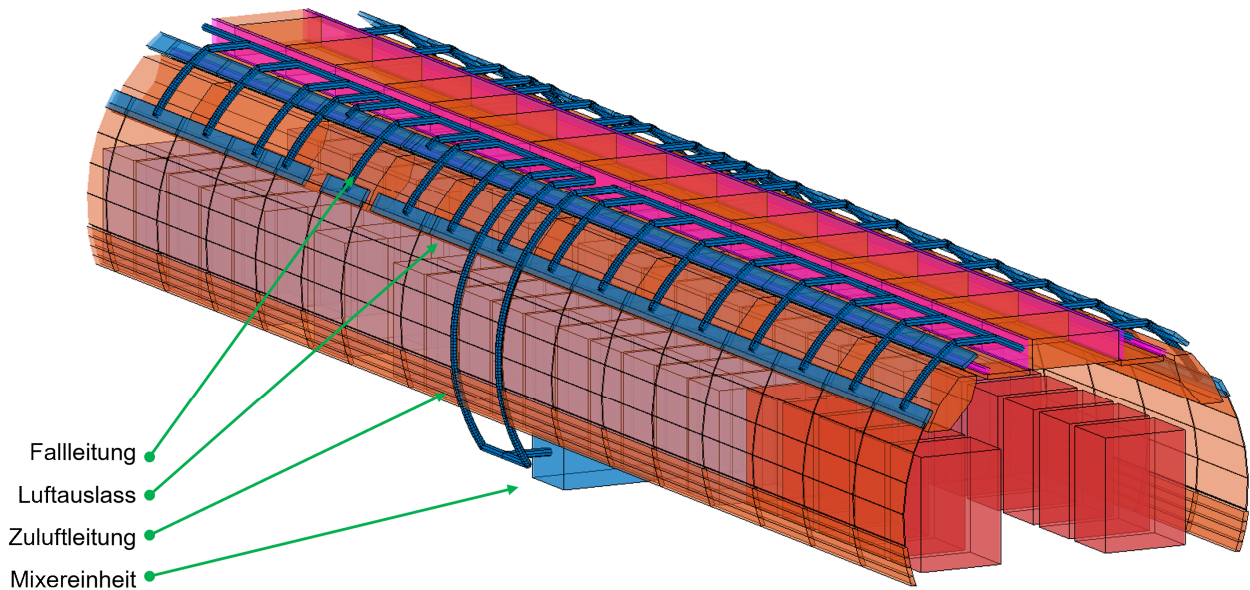


BILD 8. MATLAB Plot, Repräsentation des DISKUS-Kabinenkonzept mit Bounding-Boxen

Ergebnisse der Analyse

Die vorgestellte Methodik konnte am Beispiel des DISKUS-Kabinenkonzept demonstriert werden. Im Folgenden werden die zentralen Ergebnisse hinsichtlich Performance, Modellaufbereitung und Prozesszeiten diskutiert. Ein Resultat betrifft die Performance in der VR-Umgebung. Im vorliegenden Szenenstand wurde in Unreal eine Bildrate von etwa 50–55 Frames pro Sekunde erreicht. Dieser Wert liegt grundsätzlich im Bereich der VR-Tauglichkeit, erreicht jedoch noch nicht die Stabilität, die wünschenswert wäre, da die erzielten FPS noch nicht auf einem Wert sind bei dem Motion-Sickness verringert wird [2]. Gleichzeitig verdeutlicht das Ergebnis, dass die Pipeline in ihrer jetzigen Form bereits funktional einsetzbar ist, während Optimierungspotenziale klar erkennbar bleiben. Hierzu zählen insbesondere Verbesserungen in der Materialverarbeitung, die Einführung von Level-of-Detail-Strategien sowie der Einsatz moderner Engine-Technologien wie Nanite oder Streaming, die für große Szenen die Frametimes weiter stabilisieren können.

Besonders hervorzuheben ist die erzielte Reduktion der Modellgröße. Durch die instanzienbasierte Aufbereitung konnte die Größe der Blender-Szene von ursprünglich ca. 330 MB auf etwa 13 MB verringert werden, eine Verringerung um den Faktor ~25. Diese deutliche Reduktion zeigt den Mehrwert der Instanziierung. Nur durch die Eliminierung redundanter Mesh-Kopien und die konsistente Nutzung gemeinsamer Original-Assets wurde die VR-Anwendung in dieser Form möglich. Damit ist die Instanziierung für die praktische Anwendung immersiver Modelle ein notwendiger Schritt. Die Handhabung detaillierter Modelle wird durch die Reduktion der Datengröße durch Instanziierung wesentlich verbessert.

Darüber hinaus wurden die Prozesszeiten entlang der Pipeline gemessen wie in der Tabelle 1 aufgetragen. Der MATLAB-Entwurfsalgorithmus benötigte rund 41,9 Sekunden für die Erzeugung der Bounding-Box-Repräsentationen. Das Laden und Platzieren der Objekte in Blender stellte mit 156,5 Sekunden den größten Zeiteanteil dar, gefolgt vom Instanziierungsschritt mit 16,8 Sekunden und dem FBX-Export mit 7,5 Sekunden. In Unreal lagen die Zeiten beim Import der FBX-Szenen bei 41 Sekunden, für das Post-Processing bei 4,5 Sekunden und für den Aufbau der Beleuchtung bei 68 Sekunden. Das Erzeugen der rechteckigen Lichtquellen erfolgte vernachlässigbar schnell. Nicht berücksichtigt ist die Startzeit der Unreal Engine mit Template-Projekt, die je nach System zwischen 30 und 60 Sekunden betragen kann. Insgesamt ergibt sich ein reproduzierbarer, vollständiger Ablauf in wenigen Minuten, was die Praxistauglichkeit des Ansatzes unterstreicht.

TAB 1 Laufzeiten der einzelnen Prozessschritte

Prozessschritt	Zeit [s]
MATLAB Auslegung	41,9
Blender Platzierung	156,5
Blender Instanziierung	16,8
Blender Export	7,5
Unreal Import	41,0
Unreal Post-Processing	4,5
Unreal Beleuchtung	~0
Unreal Lichtaufbauen	68,0
Gesamt (ohne Startzeit)	336,2

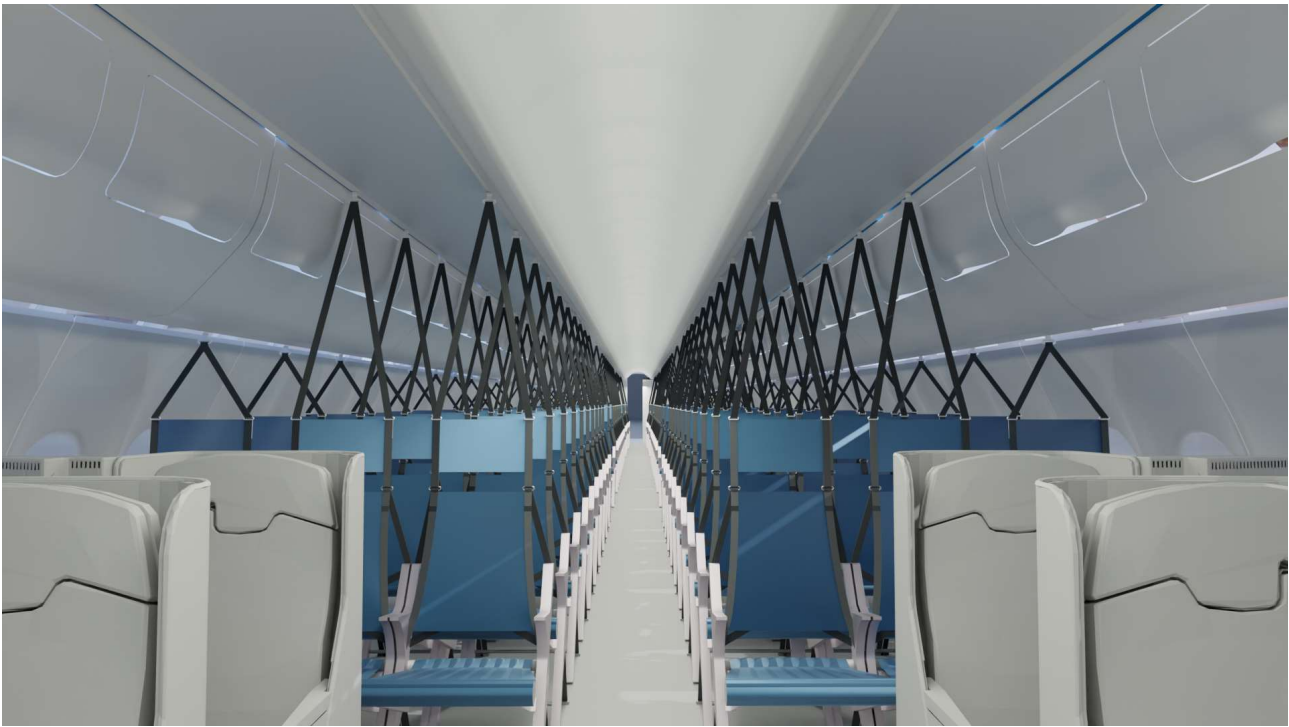


BILD 9. DISKUS-Kabinenkonzept in Unreal Engine

Ein weiterer Aspekt betrifft die Qualität und Robustheit der Ausgangsmodelle. In Blender zeigte sich, dass einzelne Objekte fehlerhafte Normalen aufwiesen. Diese wurden in Unreal durch die Aktivierung von Two-Sided-Materialien kompensiert, sodass Flächen auch bei falsch orientierten Normalen korrekt dargestellt werden. Gleichzeitig wird damit deutlich, dass die Qualität der Bibliotheksmodelle ein entscheidender Faktor ist. Ziel muss es sein, Normalen und Materialzuweisungen bereits im Exportzustand konsistent bereitzustellen, sodass Behelfslösungen wie Two-Sided möglichst vermieden werden können.

Die Diskussion zeigt insgesamt, dass die Pipeline in ihrer jetzigen Form einen erfolgreichen Proof of Concept darstellt. Sie ermöglicht eine konsistente, durchgängige Prozesskette vom parametrischen Systementwurf bis zur immersiven Darstellung in VR und ist nicht auf die DISKUS-Fallstudie beschränkt, sondern grundsätzlich auf andere Kabinenkonfigurationen übertragbar. Gleichzeitig verdeutlichen die Ergebnisse, dass Optimierungsbedarf in mehreren Bereichen besteht. Zu optimierende Teilaspekte wären unter anderem die Qualitätssteigerung der Ausgangsmodelle, die Beschleunigung des Blender-Ladeprozesses und die Verbesserung der VR-Performance durch Engine-seitige Mechanismen.

4. ZUSAMMENFASSUNG UND AUSBLICK

Die Arbeit zeigt eine durchgängige, automatisierte Prozesskette vom parametrischen Systementwurf bis zur Darstellung in der virtuellen Realität mit Unreal Engine. Der Kern der Methodik liegt in der Verbindung von parametrischen Entwurfsdaten mit einer konsistenten Geometrieauflbereitung und einer automatisierten Integration in

Unreal, die eine VR-taugliche Darstellung gewährleistet. Durch den Einsatz von Bounding-Box-Repräsentationen, Instanziierung und Post-Processing-Regeln konnte ein methodischer Ansatz etabliert werden, der sowohl technische Robustheit als auch eine für VR-Anwendungen hinreichende Performance sicherstellt.

Die Umsetzung des methodischen Ansatzes wurde an der DISKUS-Konzeptkabine exemplarisch demonstriert. Dort konnte gezeigt werden, dass auch nicht-standardisierte Kabinenlayouts mit spezifischen Anforderungen, wie einer angepassten Belüftungsarchitektur, konsistent in der Prozesskette abgebildet werden. Der prototypische Szenenstand erreichte in der VR-Arbeitsumgebung eine Bildrate von ~50–55 FPS und eine instanziierungsbedingte Modellgrößenreduktion der FBX-Szenen in Blender um den Faktor ~25.

Für die Zukunft eröffnen sich mehrere Entwicklungspfade. Auf Ebene der Systemauslegung bietet sich die Integration zusätzlicher Kabinensysteme an, sodass der Entwurfsalgorithmus in MATLAB weitere Systeme abbilden kann. Ein langfristiges Ziel am Institut Systemarchitektur in der Luftfahrt ist es zudem, Kabinensysteme direkt in FUGA auszuliegen und in CPACS zu schreiben um auf den Zwischenschritt MATLAB für die weitere Systemauslegung zu verzichten. Daraus ergibt sich ein klarer Entwicklungspfad. Die Pipeline müsste so erweitert werden, dass sie Systeme unmittelbar aus CPACS einlesen und anhand dieser Informationen die entsprechenden Objekte in Blender platzieren kann. Das bislang eingesetzte CPACS-nahe XML aus MATLAB würde damit entfallen, und eine direkte Anbindung an CPACS wäre möglich. Ein weiterer Entwicklungspfad betrifft die Robustheit und Nachvollziehbarkeit der gesamten Pipeline. Schon in Blender können systematisch Kennzahlen der erzeugten Szenen erfasst und in einer Log-Datei abgelegt werden.

Dazu gehört beispielsweise die Anzahl der platzierten Objekte pro Szene, die verwendeten Materialien oder die Größe der exportierten FBX-Dateien. Eine solche automatisierte Protokollierung schafft Transparenz, erleichtert die Analyse von Prozessschritten und erhöht die Nachvollziehbarkeit der Ergebnisse. Auch die Einbindung externer Simulationsdaten, beispielsweise aus Strömungs- oder Akustikberechnungen, eröffnet die Möglichkeit, physikalische Effekte direkt in VR nachvollziehbar zu machen.

Auf der Engine-Ebene bestehen Optimierungspotenziale durch fortgeschrittene Technologien wie Nanite, LOD-Systeme, Streaming-Mechanismen und verbesserte Beleuchtungspfade, die eine noch stabilere Darstellung großer Kabinenmodelle in VR ermöglichen. Ergänzend sollten Materialpfade konsolidiert und Mesh-Bibliotheken qualitativ weiter abgesichert werden, damit Normalen und UV-Layouts bereits im Exportzustand konsistent vorliegen.

Ein weiterer Aspekt ist die Weiterentwicklung der VR-Anwendung selbst. Damit sie nicht auf die reine Betrachtung von 3D-Geometrien beschränkt bleibt, müssen Interaktions- und Co-Design-Funktionalitäten eingeführt werden. Dazu gehören Selektions-, Ausblend- und Farbänderungsoptionen auf Objektebene, das Hervorheben kompletter Systemfamilien sowie das Visualisieren von Verbindungen zwischen Komponenten. Eine weitere Perspektive eröffnet die Einbindung kollaborativer Multi-User-Szenarien, bei denen mehrere Nutzer gleichzeitig in derselben Kabine arbeiten, Varianten bewerten oder Konfigurationen diskutieren können.

Darüber hinaus eröffnet die entwickelte Methodik neue Anwendungsmöglichkeiten für nutzerzentrierte Studien. Da Varianten der Kabine konsistent und reproduzierbar in VR verfügbar sind, können Probanden frühzeitig mit den Konzepten interagieren. Auf diese Weise könnten sich beispielsweise Komfortbewertungen, Sichtbarkeitsanalysen oder Interaktionsstudien mit Crew und Passagieren systematisch durchführen lassen. Solche Nutzerstudien tragen dazu bei, Entwurfsentscheidungen auf eine breitere empirische Basis zu stellen und die Akzeptanz disruptiver Kabinenkonzepte frühzeitig abzusichern.

Um diese Funktionalitäten zu erreichen, muss allerdings auch das Template-Projekt in Unreal grundlegend erweitert werden. Hierfür sind vorgefertigte Blueprints erforderlich, die die gewünschten Interaktionsmöglichkeiten nativ bereitstellen und eine flexible, wiederverwendbare Basis für verschiedene Anwendungsfälle schaffen. Gegenwärtig wird ein Template-Projekt als Ausgangspunkt genutzt, in das die Assets importiert werden. Damit dieses Template nicht verändert wird, empfiehlt sich eine Kopierstrategie. Vor jedem Import könnte das Template automatisiert dupliziert werden, sodass jede Pipeline-Ausführung auf einer unveränderten Ausgangsbasis startet. Das garantiert Konsistenz, verhindert ungewollte Effekte und erleichtert die saubere Archivierung von Ergebnissen.

Kontaktadresse:

lukas.fuszenecker@dlr.de

Acknowledgment

Die Arbeit entstand durch eine Förderung des Bundesministeriums für Wirtschaft und Klimaschutz im Rahmen des LuFo 6.3 Projekt DISKUS. Wir danken unseren Forschungspartnern CompriseTec und Diehl Aviation, die diese Untersuchung ermöglicht haben.

Referenzen

- [1] **Selzer, M. N. et al. (2023)** – A Methodology for Generating Virtual Reality Immersion Metrics based on System Variables. *Journal of Computer Science & Technology, Volume 23*
<https://doi.org/10.24215/16666038.23.e08>
- [2] **Geris, A. et al. (2024)**. Balancing performance and comfort in virtual reality: A study of FPS, latency, and batch values.
<https://doi.org/10.1002/spe.3356>
- [3] **DLR – Deutsches Zentrum für Luft- und Raumfahrt e.V. (2025)**. Auf dem Weg zu einer emissionsfreien Luftfahrt – Luftfahrtstrategie des DLR zum European Green Deal. *Programmstrategie Luftfahrt*. Abgerufen am 09.2025 von: <https://.../auf-dem-weg-zu-einer-emissionsfreien-luftfahrt.pdf>
- [4] **BMWi – Bundesministerium für Wirtschaft und Energie (2014)**. Die Luftfahrtstrategie der Bundesregierung. Abgerufen am 09.2025 von: <https://.../luftfahrtstrategie-der-bundesregierung.pdf>
- [5] **Alder, M., et al. (2020)**. Recent Advances in Establishing a Common Language for Aircraft Design with CPACS. *Aerospace Europe Conference 2020*.
https://elib.dlr.de/134341/1/AEC2020_174.pdf
- [6] **Burschik, T., et al. (2023)**. Introduction of a System Definition in the Common Parametric Aircraft Configuration Schema (CPACS). *Aerospace*, 12(5), 373.
<https://doi.org/10.3390/aerospace12050373>
- [7] **Walther, J.-N. et al. (2022)**. Extensible aircraft fuselage model generation for a multidisciplinary, multi-fidelity context. *33rd Congress of the International Council of the Aeronautical Sciences*.
<https://elib.dlr.de/189459/>
- [8] **Hahn, A. (2010)**. Vehicle Sketch Pad: A Parametric Geometry Modeler for Conceptual Aircraft Design. *48th AIAA Aerospace Sciences Meeting, Orlando, Florida. Document ID: 2010000304*
- [9] **Technische Universität Berlin (2025)**. PADLab – Preliminary Aircraft Design Lab. Abgerufen am 09.2025 von: <https://www.tu.berlin/luftbau/forschung/projekte/abgeschlossene-projekte/padlab>

- [10] **Jünemann, M.; Thielecke, F.; et al. (2018).** Methodology for Design and Evaluation of More Electric Aircraft Systems Architectures within the AVACON Project. *Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V*
<https://doi.org/10.25967/480197>
- [11] **Schmidt, M., & Engelmann, M. (2016).** PAXelerate – An Open Source Passenger Flow Simulation Framework for Advanced Aircraft Cabin Layouts. *54th AIAA Aerospace Sciences Meeting, San Diego, USA*.
<https://doi.org/10.2514/6.2016-1284>
- [12] **Fuchs, M.; Beckert, F et al. (2020).** Virtuelle Realität im digitalen Designprozess von Flugzeugkabinensystemen. *Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V.*
<https://doi.org/10.25967/530007>
- [13] **Fuchs, M.; Ghanjaoui, Y.; Abulawi, J. et al. (2022).** Enhancement of the virtual design platform for modeling a functional system architecture. *CEAS Aeronautical Journal*, 13, 1101–1117.
<https://doi.org/10.1007/s13272-022-00608-z>
- [14] **Fuchs, M., et al. (2022).** Virtual reconfiguration and assessment of aircraft cabins using model-based systems engineering. In: *33RD Congress of the International Council of the Aeronautical Sciences Stockholm, Sweden*
- [15] **Ghanjaoui, Y.; Fuchs, M.; Biedermann, J. et al. (2023).** Model-based design and multidisciplinary optimization of complex system architectures in the aircraft cabin. *CEAS Aeronautical Journal*, 14, 895–911
<https://doi.org/10.1007/s13272-023-00683-w>
- [16] **Bielsky, T.; Jünemann, M.; Thielecke, F. et al. (2020).** Parametric modeling of the aircraft electrical supply system for overall conceptual systems design. *Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V*
<https://doi.org/10.25967/530143>
- [17] **Walther, J.-N., et al. (2022).** Automatic cabin virtualization based on preliminary aircraft design data. *CEAS Aeronautical Journal*, 13, 403–418.
<https://doi.org/10.1007/s13272-021-00568-w>
- [18] **Walther, J.-N., et al. (2021).** High-Fidelity Digital Cabin Mock-Up based on Preliminary Aircraft Design Data. *AIAA Aviation 2021 Forum*.
<https://doi.org/10.2514/6.2021-2775>
- [19] **Engelmann, M.; Drust, D.; Hornung, M. et al. (2020).** Automated 3D cabin generation with PAXelerate and Blender using the CPACS file format. *Deutsche Gesellschaft für Luft- und Raumfahrt - Lilienthal-Oberth e.V..*
<https://doi.org/10.25967/530014>
- [20] **Bauhaus Luftfahrt (2025).** PAXelerate (GitHub Repository). Abgerufen am 09.2025 von: <https://github.com/BauhausLuftfahrt/PAXelerate>
- [21] **Moerland-Masic, I., et al. (2022).** Application of VR technology in the aircraft cabin design process. *CEAS Aeronautical Journal*, 13, 127–136.
<https://doi.org/10.1007/s13272-021-00559-x>
- [22] **De Crescenzo, F., Bagassi, S., Asfaux, S. et al. (2019).** Human-centred VR evaluation of cabin interiors for business jet aircraft in virtual reality. *International Journal on Interactive Design and Manufacturing*, 13, 761–772.
<https://doi.org/10.1007/s12008-019-00565-8>
- [23] **Fuchs, M., et al. (2021).** Experience of Conceptual Designs and System Interactions for the Aircraft Cabin in VR. *AIAA AVIATION 2021 Forum*.
<https://doi.org/10.2514/6.2021-2773>
- [24] **Epic Games.** Instanced Static Mesh Component in Unreal Engine. Abgerufen am 09.2025 von: <https://../instanced-static-mesh-component-in-unreal-engine>
- [25] **Hogan, J., et al. (2022).** Analyzing Performance Issues of Virtual Reality Applications. *arXiv preprint*. <https://doi.org/10.48550/arXiv.2211.02013>
- [26] **Epic Games.** VR Performance Testing in Unreal Engine. Abgerufen am 09.2025 von: <https://../vr-performance-testing-in-unreal-engine>
- [27] **Epic Games.** Unwrapping/Generating Lightmap UVs. Abgerufen am 09.2025 von: <https://../unwrapping-uvs-for-lightmaps>
- [28] **Epic Games.** XR Best Practices. Abgerufen am 09.2025 von: <https://../xr-best-practices-in-unreal-engine>
- [29] **Meta.** VR Performance Guidelines. Abgerufen am 09.2025 von: <https://developers.meta.com/performance-guidelines>
- [30] **Epic Games (2025).** FbxSceneImportFactory – Unreal Python 5.6 Documentation. Abgerufen am 09.2025 von: <https://../python-api/>
- [31] **Epic Games.** Introduction to Blueprints Visual Scripting in Unreal Engine 5.6. Abgerufen am 09.2025 von: <https://../introduction-to-blueprints-visual-scripting-in-unreal-engine>