Game Design Document

Der Hindenburg Flug

Frankfurt-Lakehurst

Lea Stegk (264100, MIM2)

Ron Metzger (265018, MIM1)

Marcel Bauer (263068, DIM2)

Erstellt für die Master Veranstaltungen Game Production und Game Design

Game concept

Introduction

Hindenburg VR ist ein exploratives VR Spiel, das in Unity entsteht und in dem der Absturz des berühmten Zeppelins verhindert werden muss.

Description

Du spielst ein Mitglied der Crew an Bord der LZ-129 Hindenburg. Mitten auf eurem Flug nach Nordamerika erreicht dich ein Funkspruch von außen - eine Bombe wurde auf dem Luftschiff platziert und droht zu explodieren falls du dem Anrufer nicht Folge leistest und ihm den Schatz bringst, der sich auf dem Schiff befindet.

Löse Rätsel und belausche Passagiere an Bord, um sie zu finden und die Katastrophe aufzuhalten! Ein in Originalgröße modellierter Zeppelin wartet darauf, von dir in First Person Virtual Reality erkundet zu werden.

Key features

- Volles Virtual Reality Erlebnis
- Object Physics
- Realitätsnaher Aufbau der Hindenburg
- Rätsel

Genre

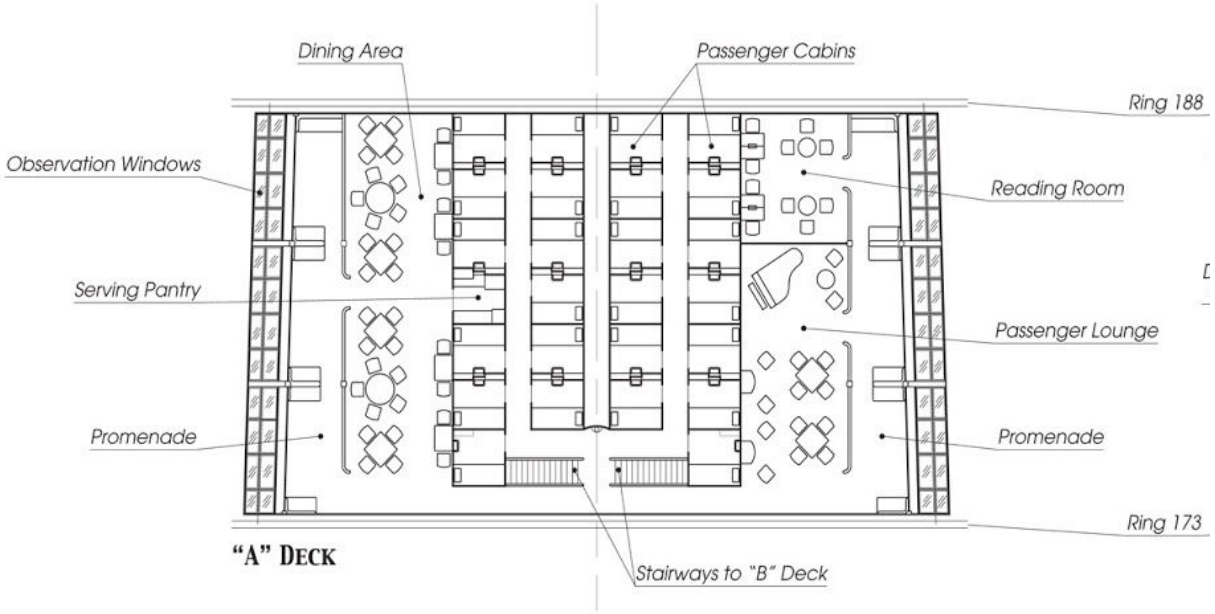
Hindenburg VR ist ein Singleplayer Puzzle-Adventure, das in der First Person Perspektive stattfindet.

Platform

Hindenburg VR wird primär für die Oculus Rift S entwickelt mit dem Ziel, das Spiel am Ende möglichst kompatibel zu anderen VR Geräten zu halten.

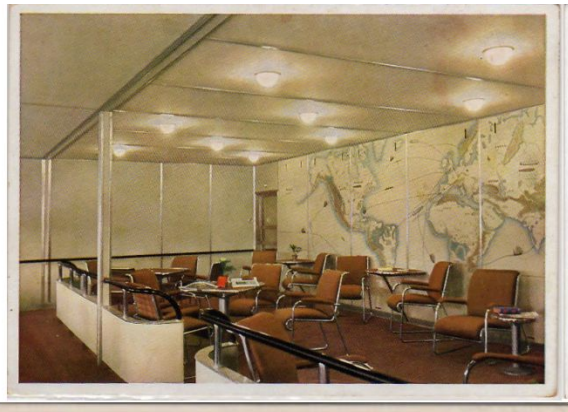
References

Deck A

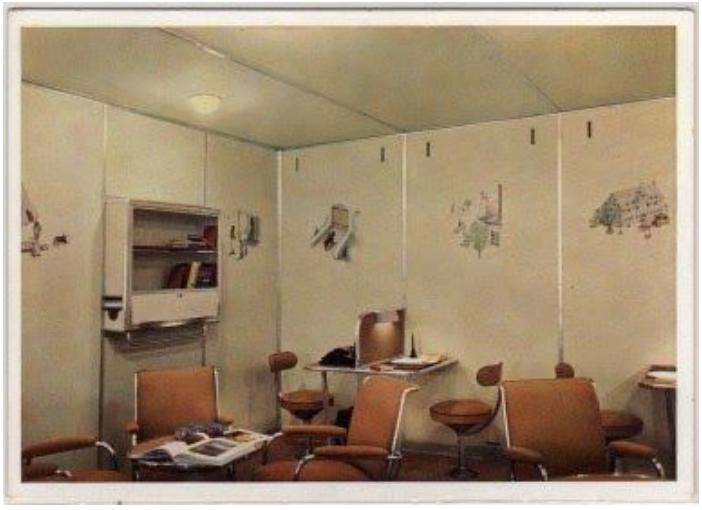


Dining Room

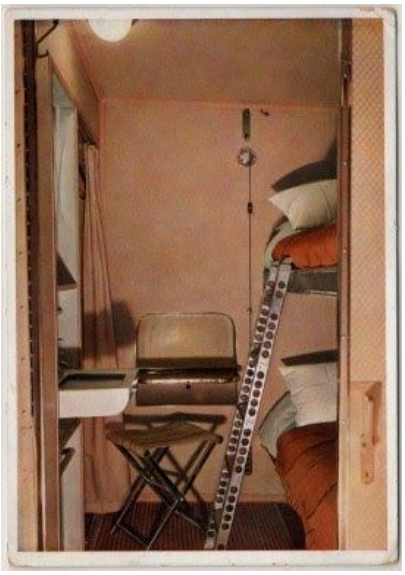
Lounge



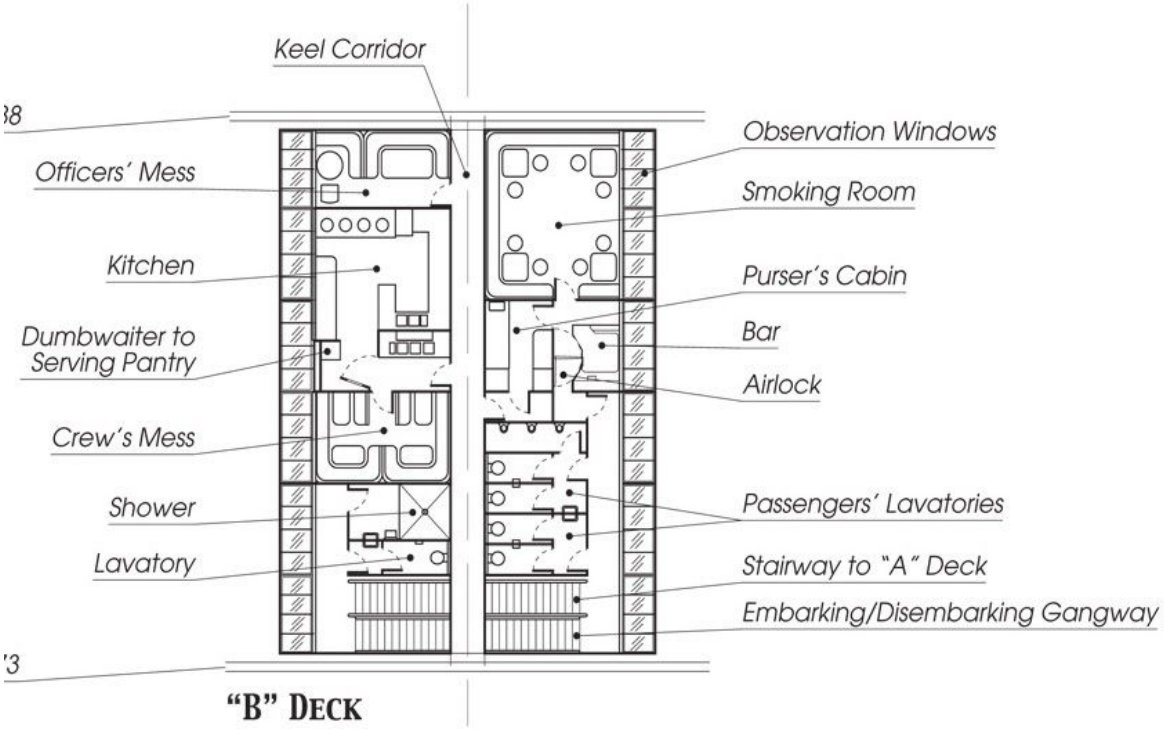
Writing room



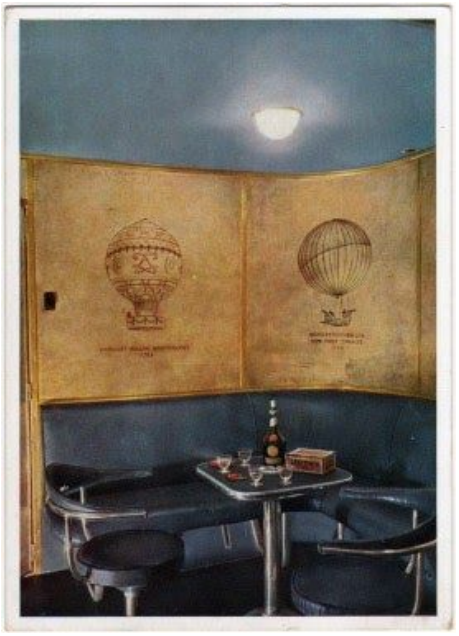
Kabine

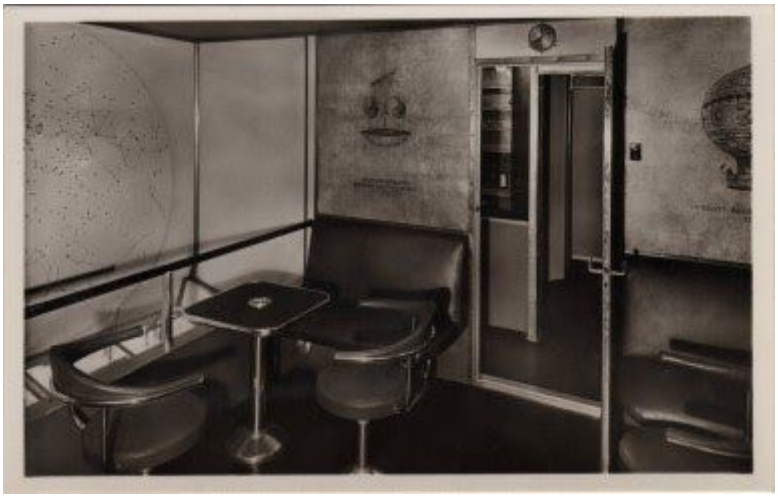


Deck B



Smoking Room





Bar

Funktionale Spezifikation

Game Mechanics

Core Game Play

Bei dem Spiel handelt es sich um ein Rätsel-Game im Bereich VR für die Plattform Oculus Rift S. Der Spieler steuert seinen Charakter in der Ego-Perspektive hierbei durch verschiedene Räume, die sich auf unterschiedlichen Stockwerken befinden. Ziel des Spiels ist es alle Rätsel zu lösen. Diese löst der Spieler, indem er verschiedene Gegenstände suchen muss. Dabei kann der Spieler mit fast allen Assets im Game interagieren um die Vorteile des VR-Systems ausnutzen zu können. Verschiedene Assets wie beispielsweise Teller können hierbei auch zu Bruch gehen. Der Spieler kann verschiedene Gegenstände in die Hand nehmen, genauer betrachten und teilweise auch anderweitig mit ihnen interagieren, beispielsweise indem er zwei Gegenstände ineinander steckt.

Game Flow

Der Spieler nimmt zunächst einen Anruf über Funk entgegen. Dafür wählt er das Funkgerät aus und bewegt es zu seinem Ohr. Anschließend bewegt sich der Spieler durch die verschiedenen Räume, mit dem Ziel einen Schatz zu finden, den der Terrorist am Funkgerät einfordert.
Der Spieler stellt fest, dass ein Raum verschlossen ist. Er sucht nach einer Lösung und findet Dietriche, mit denen er die Tür öffnen kann.

Der Spieler betritt den Raum und findet eine Schatztruhe vor, die zweifach verriegelt ist: durch ein Zahlenschloss und ein Loch, das mit einem Gegenstand gefüllt werden muss. Der Spieler erkundet die Räume und findet ein Rad, das sich an einer Tür anbringen lässt um diese zu öffnen.

Dort findet er ein Bild, dessen Rückseite sich abnehmen lässt, was einen 4-stelligen Code enthüllt. In einem weiteren Raum entdeckt er ein Murmelspiel, das er löst. Die

Murmel bekommt er in sein Inventar und kann nun mit ihr und dem Code die Truhe öffnen.
In der Truhe findet er den Schatz, den er nun aus dem Fenster werfen kann. Zuerst muss er allerdings noch einen Fallschirm für die Diamanten bauen.
Dafür holt sich der Spieler eine Saite aus dem Flügel und Stoff aus von der Bar. Eine Nietenzange, die er in seiner Kabine findet, lässt ihn Löcher in das Tuch schneiden, wodurch die Saiten gezogen werden. Daran kann er nun die Diamanten anhängen und aus dem Fenster werfen.

Zuletzt muss er allerdings, da die Bombe über einen Timer explodieren wird, in das Innere der Hindenburg und die Bombe finden. Nach dem finden muss er mit den Anweisungen des Täters die Bombe entschärfen.

Charaktere

Das Spiel besteht aus zwei Hauptcharakteren: dem Spieler selbst, der in Ego-Perspektive durch das Luftschiff gesteuert wird und dem Erpresser, den man nur über Funk hören kann. Zusätzlich gibt es NPCs, die sich in ihren Kabinen wegen der bombendrohung eingeschlossen haben und die man nur dort tuscheln hört, wenn man an den Kabinen vorbeiläuft. Teilweise bekommt der Spieler nützliche Infos wie den Fundort der Dietriche dort mit, wenn er eine Weile stehen bleibt und lauscht.

Game Play Elemente

Der Spieler kann mit fast allen Assets im Spiel interagieren. Teilweise tragen diese Interaktionen zum Fortschritt des Spielers bei, teilweise tun sie dies nicht. Beispielsweise kann der Spieler Teller von einem Tisch hochheben, betrachten und sogar zerschmettern, was ihn jedoch in der Story nicht weiterbringt. Die Elemente, die ihn in der Story weiterbringen sind das Türrad, ein Bilderrahmen, ein Murmelspiel, ein Zahlenschloss und der Beutel, in den die Dokumente zum Schluss gesteckt werden. Diese können nicht nur aufgehoben und betrachtet, sondern auch ineinander gesteckt werden (zB. kann das Türrad in die Tür eingefügt werden)

Game Physik und Statistiken

Die Physik des Spiels ist realistisch gehalten. Der Spieler kollidiert beim Laufen mit verschiedenen Gegenständen wie beispielsweise Tischen und Stühlen. Wenn die

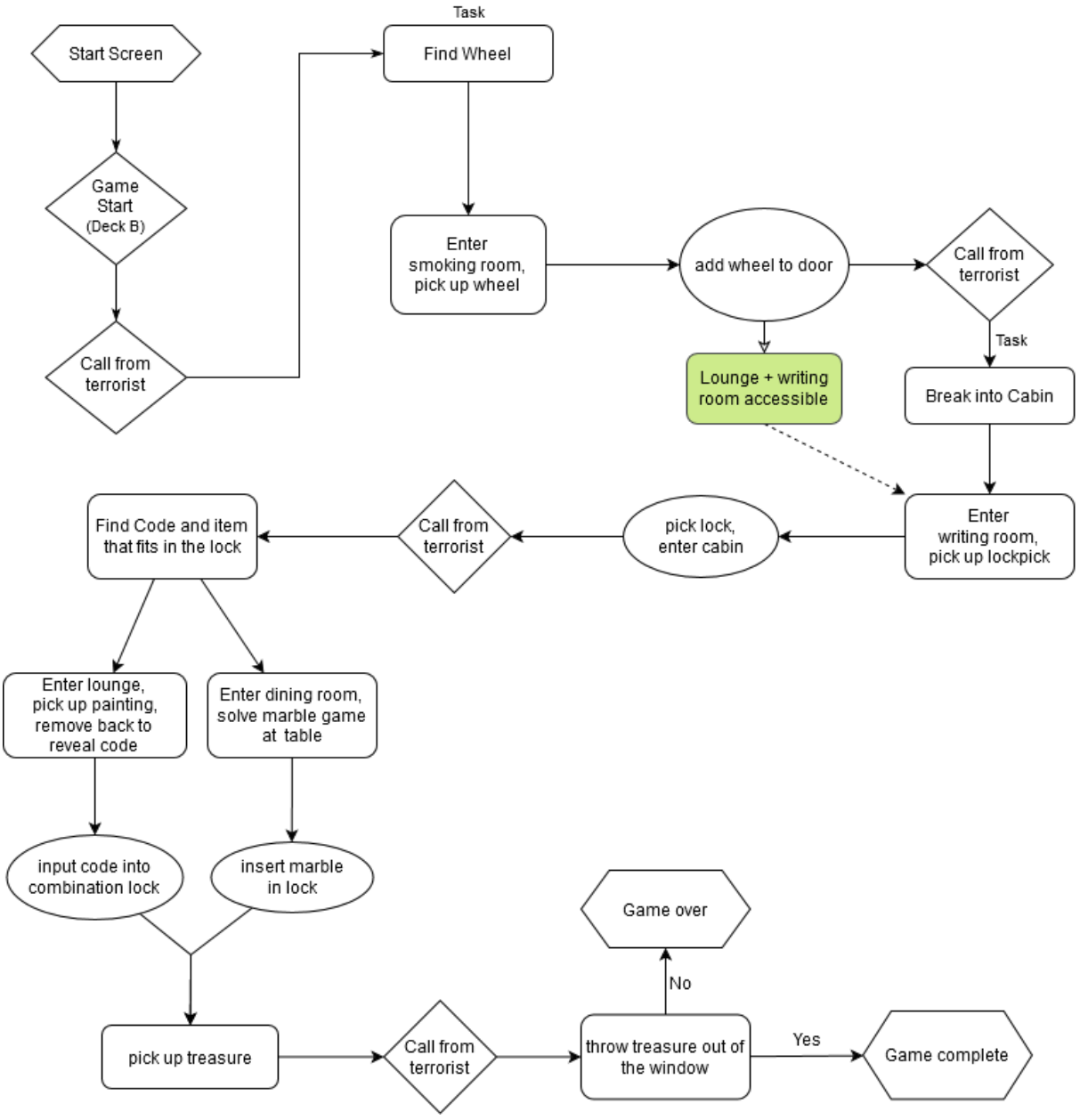
Teller, die sich auf den Tischen befinden über den Rand bewegt werden, fallen sie herunter und gehen zu Bruch. Die Dimensionen des Raumschiffes sind an die Größe einer durchschnittlichen realen Person angepasst, ebenso die Laufgeschwindigkeit des Spielers.

User Interface

Das User Interface wird in die Spielewelt in Form eines Gürtels mit zwei Gürteltaschen auf natürlich wirkende Art und Weise eingebaut und dient als Inventar. In jeder Tasche kann jeweils ein Gegenstand verstaut werden, in jede Hand kann ebenfalls ein Objekt genommen werden. Somit kann der Spieler maximal vier Gegenstände mit sich herumtragen.

Auf die Anrufe des Terroristen kann der Spieler Antworten geben, die er aus einer Liste mit Optionen auswählt.

Flowchart & Level Diagram



Hexagon: Game Screen, Raute: Ereignisse, Rechteck: Möglichkeiten des Spielers

Mockups

Start:

Gewonnen:

Verloren:

GUI Objects

Das GUI beschränkt sich, da es sich um ein VR-Spiel handelt auf den Gürtel, mit dem Inventar und einem Interface, das erscheint, wenn man dem Terroristen mit dem Funkgerät mit verschiedenen Antwortmöglichkeiten antwortet.

Art

Cutscenes oder Videos sind in VR-Spielen nicht nötig. Kunst findet sich nur als Gemälde wieder, das zugleich ein Schlüsselobjekt zur Lösung des Zahlencodes dient. Auch ein Fernseher mit abspielbaren Video Elementen macht an dem Ort, in der das Spiel stattfindet, keinen Sinn.

Overall Goals

Die großen Ziele des Game Designs sind:

1. Ein möglichst realistisches Gefühl zu erzeugen, sich tatsächlich auf einem Luftschiff zu befinden.
2. Den Spieler in die Zeit der Hindenburg zu versetzen.
3. Spannung und Spaß durch die Rätsel aufzubauen.

3D Art & Animation

Animationen finden bei der Interaktion mit der Umgebung in VR statt. Teller, Gläser und Flaschen zerbrechen wenn man sie auf den Boden wirft, Türen und die Schatztruhe öffnen sich, die Murmel bewegt sich in dem Murmelspiel, Tische und Stühle lassen sich umwerfen und Gegenstände lassen sich miteinander kombinieren.

Graphics Engine

Im Spiel wird 3D-Polygon-Rendering benutzt. Es findet in einer 3D-VR-Umgebung statt, die in Unity entstanden ist. Es werden keine 2D Sprites genutzt.

Artist Instructions

Die Vorgaben an das Design der Assets waren hauptsächlich die möglichst detailgetreue Umsetzung nach der Vorlage des Inventars der realen Hindenburg und die entsprechende Skalierung, damit die Größe der Räume und deren Einrichtung realistisch in VR erscheint.

Sound and Music

Overall Goals

Es gibt (außer im Start-/Endbildschirm) keine Hintergrundmusik. Stattdessen wird die Atmosphäre durch Umgebungsgeräusche unterstrichen - das Knarzen von Holzdielen, brummende Motoren und laute Klänge, wenn metallische Komponenten einander berühren. Generell ist die Soundkulisse im Passagierbereich ruhiger, im

Rest des Luftschiffes lauter und mechanischer, da sich dort die ganze Mechanik befindet, um den Zeppelin in der Luft zu halten.

Sound FX

GUI: Button klicken, Fenster pop-up, Auswahl/Benutzung Objekt

Charaktere: Voice acting, Funksprüche, Schritte, Getuschel

Game Play Elemente: Tür öffnen, Gegenstand aufheben, Benachrichtigung, Umgebungsgeräusche, piepende Bombe/Shut off sound, Zusammenstecken von Objekten

Terrain: Metallische Stoßgeräusche, Knacken, brummende Motoren

Event Jingles: Erfolg, Fehlschlag, Game over, erfolgreich abgeschlossenes Spiel, Entdeckung

Screen: Title screen (historisch)
Spielende: Game over / erfolgreich beendet (Flammen/fröhliche Melodie)

Situationen: Normal (Umgebungsgeräusche, knarzender Boden, zerbrechliche Gegenstände gehen kaputt)
Gefahr (Sirenen, Piepende Bombe)

Story

Das Spiel startet an Bord des Luftschiffes. Nachdem der Spieler in den ersten Raum kommt, kann er sich dort umsehen.
Nachdem Geräusche ertönen, die das Abheben des Zeppelins signalisieren, rauscht das Funkgerät des Spielers. Er nimmt den Anruf an und es ertönt eine verzerrte Stimme, die ihm mitteilt, dass er eine Bombe auf der Hindenburg versteckt hat und dem Spieler aufträgt, ein Schatz zu finden.
Falls er sich weigert, droht er damit, das Luftschiff explodieren zu lassen! Der Spieler bekommt ein Zeitlimit von einer Stunde, in der er seine Aufgabe erfüllen soll.

Der Anrufer legt auf und der Spieler kann nun den Rest des Zeppelins erkunden. Gelegentlich muss er Rätsel lösen um an bestimmten Stellen weiterzukommen, so fehlt beispielsweise an einer Tür ein Rad, um sie zu öffnen; dieses muss erst gefunden und dann mit der Tür kombiniert werden, ehe der Weg frei wird.

Nachdem der Spieler ein Rätsel löst, meldet sich der Anrufer erneut und fordert, das nächste zu lösen und am Ende den Schatz in einen Beutel zu stecken und aus einem Fenster zu werfen. Weigert sich der Spieler dies zu tun, so gibt der Täter nur ungefähre Informationen über die scharfe Bombe. Ein Countdown beginnt, und der Spieler kann sich nun nur retten, indem er die Bombe findet und unschädlich macht.

Explodiert sie, so erscheint der Game Over Screen. Bei erfolgreicher Beendigung des Spiels (Abwurf Schatz) sieht der Spieler einen anderen Bildschirm, auf dem “Gewonnen!” steht und ihm gratuliert wird, sowie die Nachricht, dass das Spiel nun vorbei ist.

Level Requirements

Asset Revelation Schedule

Im Spiel gibt es zwei verschiedene Arten von Assets: Assets, die der Spieler nutzt, um die Story durchzuspielen und Assets, die zur Ablenkung und Dekoration existieren.

Assets, die den Spieler weiterbringen sind: Das Funkgerät, die Dietriche, das Türrad, das Murmelspiel und die Murmel selbst, die Schatztruhe, das Bild mit der abnehmbaren Rückseite und der Schatz.

Assets, die zur Ablenkung und Dekoration existieren sind: Tische, Stühle, Teller, Vasen, Gläser, Flaschen, Sessel, Klavier, Betten, Waschbecken, Aschenbecher und Leitern.

Technical specifications

Game Mechanics

Wenn die .exe auf einer Windows Plattform oder die .apk App auf einer eigenständigen VR Brille wie Oculus Quest gestartet wird, startet die Verarbeitung der Tracking Ergebnisse für das Spiel, während die ersten Daten wie 3D Objekte geladen werden. Anschließend gelangt man in den ersten Screen, der den Start des Spiels ermöglicht. Über das Treffen der Hitbox mit dem Fingern der Controller wird die gewählte Option anschließend ausgeführt. Das wäre entweder das starten des Spiels oder anzeigen eines zweiten Screens, welche Optionen der Spieler ändern wolle. Beim starten des Spiels wird anschließend der Control Loop dauerhaft ausgeführt. Darin wird Position und Rotation des VR Headsets und der Controller für verlinkte GameObjects abgefragt und eingefügt. Zudem werden Daten für Tasten ebenso dauerhaft berechnet und anhand von Methoden im Control Loop genutzt. Zudem werden Kollisionen abgefragt und, falls Kollisionen entstehen, nach Layern oder Objekten gesucht und weitere Methoden gestartet.

Platform and OS

Die Zielplattform ist Windows 10, ein möglicher Port für die Android VR Brille “Oculus Quest” ist allerdings auch möglich. Die minimalen Anforderungen sind

Grafikkarte: GTX 980 o. ä.
CPU: Intel Core i5-7500 / Ryzen 5 1600
Arbeitsspeicher: 8 GB Ram
Es wird ein 64 Bit Prozessor und Betriebssystem vorausgesetzt.

Die Publikation wird über eine downloadbare ZIP Datei stattfinden, die nur entpackt werden muss.

External Code

Für die Programmierung des VR Titels wird die Unity Engine mit dem XR Toolkit verwendet. Dafür werden der XR Manager sowie dazugehörige XR Presets importiert. Diese dienen für eine schnelle Erstellung von Standard-Scripts wie für das Greifen von Objekten, Teleportieren etc.

Es wurde sonst kein externer Codes verwendet, da zudem Open XR, das als Standard für alle VR Brillen für kommende Jahren bieten soll, erst im Mai 2020 für Unity als Beta veröffentlicht wurde und daher viele Codes für z. B. das Oculus oder SteamVR Toolkit nicht damit kompatibel sind.

Code Objects

Es mussten keine neuen Code Objects programmiert werden. Alle nötigen Code Objects sind Bestandteil von Unity oder dem Open XR Manager.

Control Loop

Während dem Spiel werden folgende Loops berechnet:

1. Tracking der Kamera sowie der Hände anhand der VR Brille mit dazugehörigen Controller.
2. Collisions und Physik von Objekten
3. Ob greifbare Elemente mit der Hitbox der Hände kollidieren
4. Die Geschwindigkeit von Objekten anhand Physik.
5. Verschiedene Update() und LateUpdate() Methoden, die die korrekte Position, Rotation oder sonstiges Parameter überprüfen um gelöste Rätsel zu erkennen.
6. Status der Inputs der Tasten der Controller

Dazu verwenden wir

1. XR Rig, Locomotion System, Teleportation Provider, Locomotion Controller, Continuous Movement und Snap Turn Provider.
   Diese sind zusammenfassend für die Bewegungen im Spiel zuständig und sind Teil des XR Locomotion Rigs. Damit werden Bewegungen und Location der VR Brille sowie der Controller gemessen. Der Teleportation Provider bietet in unserer abgewandelten Form einen abgerundeten Pfad, dessen Ende ein blau leuchtenden Zylinder darstellt, der dem Spieler kommuniziert, dass er dort hin teleportiert. Der abgerundete Pfad wird rot, wenn ein Objekt die Teleportation verhindert. Der Snap Turn Provider bietet die Möglichkeit mithilfe des rechten Sticks den Charakter um 90 Grad zu drehen, womit Drehungen auch ohne physikalische Drehungen vom Spieler möglich sind. Dies ist z. B. wichtig, damit der Spieler sich nicht in seinen Kabeln durch häufiges Drehen einwickelt.

2. Objekte besitzen das XR Grab Interactable Script, welches Informationen liefert über die Aktionen, die während dem Greifen genutzt werden. Im Falle einer Pistole kann darin gespeichert werden, dass bei gegriffener Waffe und drücken des Triggers ein Schuss geschossen werden soll. Diese Funktion wird in unserem Spiel nach aktuellem Stand allerdings nicht genutzt werden.

   Desweiteren wird dort auch wiedergegeben, ob das Objekt gedreht oder auch von der Position bewegt werden darf. Dazu sind zudem noch Einstellungen für Position/Rotation Smoothing sowie Movement Type, womit eine direkte Bewegung (Kinematik) oder eine physikalische Bewegung (Velocity Tracking) ausgesucht werden kann. Eine physikalische Bewegung hat den Vorteil, dass das gegriffene Objekt von anderen Objekten blockiert werden kann. Der Nachteil ist allerdings, dass das Objekt dadurch ständig ein wenig wackelt.

   Bei direkten Bewegungen werden Objekte allerdings nicht von Objekten aufgehalten, wodurch massives Clipping entstehen kann. Dafür sind allerdings Bewegungen des Objekts korrekt und wackeln nicht.
   Im Falle des Murmelspiels, das als Rätsel eingebaut wurde, wurde zunächst versucht eine Lösung mit der physikalischen Bewegung zu finden. Dort war allerdings das Problem, dass das dauerhafte Wackeln eher hinderlich war. Daher haben wir uns entschieden das Murmelspiel in seiner Position statisch zu lassen, es dann über Kinematik rotieren zu lassen, wodurch kein Wackeln entsteht. Generell wurden, soweit nicht anders nötig, Objekte immer mit Velocity Tracking belegt.

3. Ein Shatter Script, welches die Geschwindigkeit des Rigid Body analysiert und bei Kontakt mit anderen Objekten einen Vergleich mit der Zerbrechlichkeit des Objekts misst, womit entschieden wird, ob das Objekt zerbrechen soll oder nicht.

4. Ein Script, das erkennt, welches Objekt in der Hitbox der Hände ist und welche Tasten gedrückt werden. Das XR Toolkit bietet keine Möglichkeit die gegriffenen Objekte zu erkennen oder zu merken. Daher prüft eine Update Methode, welches Objekt aktuell mit der Hand kollidiert. Dies ist unter anderem für verschiedene Rätsel wichtig, wo erkannt werden muss, was gerade von welcher Hand gegriffen wird und ob es gegriffen wird.

5. Sonstige Scripts, die Kollisionen abfragen, Rotationen berechnen, Sounds abspielen und Game Objekte verändern, die zu einem Großteil für Rätsel wichtig sind.

Game Object Data

Objekte wurden zu sogenannten Prefabs zusammengefügt. Diese Prefabs sind Voreinstellungen und Verlinkungen zu Objekten mit eingesetzten Komponenten und Werten. Dies erleichtert die neue Erstellung von Objekten während der Runtime und man spart an sogenannte Calls, da sie auch die Geometrie teilen und dadurch nicht mehrfach geladen werden.

Data Flow

In vielen Methoden werden GameObjects und Floats public über das UI der UnityEngine verlinkt oder eingesetzt. Dadurch erspart man sich Code und kann, falls nötig, gezielt eine Instanz eines GameObjects auswählen. Dies bietet viele Möglichkeiten, insbesondere durch die Möglichkeit Kollisionen von Hitboxen zweier (oder mehr) Objekten miteinander zu erkennen. Durch die enge Verbindung mit der UnityEngine kann man auch außerhalb des Codes im RunTime noch schnell Daten anpassen, die anschließend ohne Probleme ausgeführt werden.

Game Physics

Die greifbaren und bewegbaren Objekte werden durch den Rigidbody im UnityEditor jeden Frame neu physikalisch berechnet. Zudem ist die Weitergabe von Geschwindigkeiten in bestimmte Richtungen gegeben. Dadurch können gegriffene Objekte auch geworfen werden. Die Masse der digitalen Objekte wird stets mit der Masse vergleichbarer echten Objekte angegeben, wodurch der Spieler beim werfen auch ein Gefühl des Gewichts bekommt.

User Interface

Die Implementierung der UI geschieht über die Spielwelt. Damit ist gemeint, dass sämtliche UIs, die während dem Gameplay erscheinen, nicht einem allgemein Overlay folgen, sondern an Objekte verlinkt sind. Damit möchten wir das UI interaktiver gestalten, wodurch es ebenso sich mehr in die Spielwelt integriert. Ein Beispiel ist hierfür ein Inventar:
Anstatt eines üblichen Inventars, wie man sie aus Spielen kennt, die für einen PC Bildschirm erstellt werden, soll das Menü interaktiv genutzt werden. So ist kein Tastendruck vonnöten, welches eine Liste der Gegenstände o. ä. öffnet, sondern ein, innerhalb der Spielwelt, physikalisches Inventar platziert werden. In diesem Beispiel realisiert über einen Gürtel, der zwei Taschen aufweist. Um Gegenstände in

das Inventar zu packen, muss das gegriffene Objekt zur Gürteltasche geführt werden. Daraufhin öffnet sich die Gürteltasche, beim loslassen des Objekts wird dann das Objekt in die Tasche, und somit das Inventar, eingelegt. Dadurch ist das Inventar intuitiv und insbesondere für Menschen mit wenig Videospielerfahrung leicht zu verstehen. Das Entnehmen des Gegenstands geschieht schließlich wieder über das Greifen in/an die Tasche, wodurch das Objekt wieder erscheint.

Game Shell

In dem Spiel gibt es vier Screens.

1. Das Hauptmenü, in dem man das Spiel starten und Optionen verändern kann.
2. Die Spielwelt, die ohne sonstige Screens gezeigt wird.
3. Ein GameOver Screen, der bei zu starker Widerhandlung der Aufforderungen des Terrorists erscheint.
4. Ein Screen für den Abschluss des Spiels, der am Ende des Spiels erscheint.

Aufgrund der Spielwelt und der Art des Spiels sind keine weitere Screens wie für ein Inventar nötig.

Art

Graphics Engine

Als Grafikengines wird die für Unity native Direct3D11 Grafikengine genutzt. Auf Geräten, bei denen diese Grafikengine nicht vorhanden oder genutzt werden kann, wird automatisch auf Vulkan umgestellt. Allerdings bietet Direct3D11 einen großen Support und läuft auch auf der Oculus Quest, die keinen Computer benötigt um VR Anwendungen und Spiele abzuspielen.

Sound and Music

Sound und Musik wird über eine eigens erstellte Sound Library verwaltet, von der alle Objekte Gebrauch machen, die einen Sound abspielen müssen. Generell sieht unser Spiel keine typische Hintergrundmusik vor, die während dem Spielen läuft. Vereinzelt wird nur Musik durch das Funkgerät oder Lautsprecher im Luftschiff abgespielt.

Für Fußschritte wird ein Sound eines Schrittes verwendet, der im Pitch, Lautstärke und Geschwindigkeit bei jeder Ausführung verändert wird, um Abwechslung bei der Wiedergabe zu bieten. Das abspielen eines Schrittgeräusch wird zudem daran gemessen, auf welchem Untergrund sich der Charakter befindet. Dies wird durch Collider bestimmt, die nur beim CollisionEnter/Exit die Audiospur der Schrittgeräusche ändern, um Leistung zu sparen.

Geräusche, die interaktiv geändert werden, wie z. B. das Rätsel um die Murmel im Labyrinth, werden anhand ihrer Geschwindigkeit laufend aktualisiert und verändert. So ändert sich die Lautstärke und der Pitch der rollenden Murmel abhängig von der Geschwindigkeit, die die Murmel rollt. Sobald die Murmel eine geringe Geschwindigkeit erreicht hat und gegen das Labyrinth prallt, wird einmalig ein aufprallender Ton abgespielt, bis die Kugel wieder genug Geschwindigkeit gewinnt.

Sound Engineering Instructions

Die Sounds des Spiels sind alle relativ schlicht und normal gehalten. Es sollte auf eine normale, realistische Sound Umgebung geachtet werden, die nicht durch z. B. Dramaturgie stark verschärft wird.