Major Project

„Entwicklung eines modularen Systems für Gegner in Videospielen, vorgestellt in einem First Person Shooter Prototyp“

Modulnummer:	*6FSC1PD102*
Modulname:	*Research and Professional Development in Creative Media*
Abgabedatum:	*23.08.2024*
Abschluss:	*Bachelor of Science (Hons.) Games Programming*
Semester:	*März 2024*
Name:	*Kevin Catlett*
Campus:	*Frankfurt*
Land:	*Deutschland*
Wortanzahl:	*7418*

Selbstständigkeitserklärung:
Hiermit bestätige ich, dass ich die vorliegende Arbeit eigenständig erarbeitet habe und alle Hilfsmittel sowie Inhalte von Dritten vollständig angegeben wurden. Hierunter fällt zum Beispiel die korrekte Belegarbeit für die direkte oder sinngemäße Übernahme von Texten, Bild-, Audio- und Videomaterial sowie Code etc. Weiterhin die klare Kennzeichnung von Inhalten, die von anderen Personen oder technischen Hilfsmitteln wie z.B. einer künstlichen Intelligenz erstellt wurden.

Bad Schwalbach, 22.08.2024 — Kevin Catlett

Ort, Datum — Unterschrift Student/in

Rechtevereinbarung:
Hiermit räume ich, dem SAE Institut das nicht exklusive jedoch zeitlich und örtlich unbeschränkte Recht ein, die vorliegende Arbeit zum Zweck der Ausbildung, sowie der Darstellung von Ausbildungsinhalten, zu speichern und für Personen des SAE Instituts zugänglich zu machen.

Bad Schwalbach, 22.08.2024 — Kevin Catlett

Ort, Datum — Unterschrift Student/in

Abstract

Hintergrund: Die Videospielindustrie fordert fortlaufende Innovation. Modularität in der Entwicklung von Spielen ist ein wichtiges Thema und dessen Verwendung sollte wo möglich evaluiert und eingearbeitet werden. Bei der Erstellung von künstlicher Intelligenz ermöglicht es dynamische Spielerfahrungen sowie einen zeitsparenden Workflow.

Durchführung: Diese Arbeit präsentiert ein System zur modularen Erstellung von künstlicher Intelligenz in Videospielen, entwickelt mit Blueprints und unter Verwendung des Behavior Tree Frameworks der Unreal Engine. Zu Programmbeginn wird ein Gegner zufällig aus möglichen Körperteilvarianten zusammengesetzt, wodurch sich das Verhalten sowie visuelle Elemente dynamisch zur Laufzeit definiert und angepasst werden.

Ergebnis: Ein First Person Shooter Prototyp mit sechszehn möglichen Gegnerkombinationen demonstriert die Funktionalität des Systems und bietet eine herausfordernde Spielerfahrung.

Schlüsselwörter: künstliche Intelligenz, Modularität, Zufälligkeit, Behavior Tree, Unreal Engine

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BT……...Behavior Tree

KI……Künstliche Intelligenz

UE……..Unreal Engine

1 Einleitung

1.1 Hinführung zur kreativen Idee / Motivation

Die Videospielbranche benötigt zunehmend komplexere KI, die modular und leicht anpassbar sind. Ansprechende Gegner, meistens entwickelt unter Verwendung von Frameworks wie Finite State Machine, Utility AI, Behavior Tree oder Goal Oriented Action Planning, sind entscheidend für innovative Spielwelten. Sie sollen dynamisch auf den Spieler reagieren und ihr Verhalt durch äußere Einflüsse im Verlauf des Spiels ändern. Programmierer im Bereich der KI für Videospiele müssen daher Systeme entwickeln, die hohe Qualitätsstandards erfüllen und spezifische Projektanforderungen abdecken.

Modularität in der Programmierung bietet Vorteile wie Wiederverwendbarkeit, verbesserte Wartbarkeit und Übersichtlichkeit. Dies ist auch bei der KI-Entwicklung von Bedeutung, da verschiedene Komponenten unabhängig entwickelt, getestet und optimiert werden können. Solche Systeme ermöglichen eine schnellere Reaktion auf neue Anforderungen und ermöglichen komplexere KI.

Meine bisherigen Erfahrungen in der Entwicklung von Computerspielen hat mein großes Interesse an der Erstellung von KI geweckt. Durch meine Leidenschaft für Videospiele mit anspruchsvollen Bosskämpfen sehe ich die Möglichkeit meine Begeisterung in mein Studium einzubringen und kreative Ansätze zur Gestaltung von Gegnern in Videospielen zu entwickeln.

1.2 Kreative Idee

Mein Major Projekt zielt darauf ab, ein modulares System zur Erstellung von künstlicher Intelligenz zu entwickeln, um eine dynamische sowie vielseitige Spielerfahrung zu gewährleisten und einen effizienten Workflow zu ermöglichen. Durch die zufällige Kombination von vordefinierten Körperteilvarianten entstehen mehrere Gegner mit unterschiedlichen Verhaltensweisen, Mechaniken sowie visuellen Elementen, die anhand von Bosskampf Gameplay in einem First Person Shooter Prototyp demonstriert werden.

1.3 Zielsetzung

1.3.1 Inhaltliche Ziele

Must have

- Verhaltensweisen werden durch Körperteilvarianten definiert
- Generierung von Gegner durch zufällige Zusammensetzung von Körperteilvarianten
- Aufteilung und Festlegung der Modularität in Körperbereiche
- Integration eines First Person Shooter Controllers mit Laufen, Rennen und Schießen
- Implementierung von Interaktionsmöglichkeiten zwischen Gegner und Spieler
- Gameplay in Form eines Bosskampfes

Should have

- Visualisierung der Generierung von künstlicher Intelligenz und ihrer Möglichkeiten durch ein Graphical User Interface
- Lebenspunkte für Gegner und Spieler
- Implementierung von Sieg- und Niederlagebedingungen

Could have

- Gegner weist zusätzliche, Körperteilvariant unabhängige, Verhaltensweisen auf
- Optionen für die benutzerdefinierte Zusammenstellung von Gegner
- Ein 3D-Ansichtsmodus zum Betrachten der generierten künstlichen Intelligenz

1.3.2 Quantitative Ziele

Must have

- Generierung von sechs Gegnerkombinationen
- Drei Körperteilvarianten pro Körperbereich
- Zwei modulare Körperbereiche
- Ein Level im Prototyp

Should have

- Generierung von zwölf Gegnerkombinationen
- Vier Körperteilvarianten pro Körperbereich
- Drei modulare Körperbereiche

Could have

- Generierung von zwanzig Gegnerkombinationen
- Fünf Körperteilvarianten pro Körperbereich
- Vier modulare Körperbereiche
- Zwei Level im Prototyp

1.3.3 Qualitative Ziele

Must have

- Leicht verständlicher Workflow für die Implementierung weiterer Körperbereiche und Körperteilvarianten
- Fehlerfreies Ausüben der Verhaltensweisen
- Klar erkennbare Verhaltensweisen
- Fehlerfreie Bedienung des First Person Shooter Controllers
- Leicht erkennbares Leveldesign
- Leicht erkennbarer Gameplay loop durch ein Graphical User Interface mit Lebensanzeigen für Gegner und Spieler
- Gut verständliche Projektstruktur und Blueprints Programmierung

Should have

- Visuell unterscheidbare Körperteilvarianten
- Intuitive Bewegungsmechaniken durch den First Person Shooter Controller

Could have

- Ansprechende Körperteilvariant spezifische auditive Effekte
- Ansprechende Körperteilvariant spezifische visuelle Effekte
- Musikalische Untermalung des Gameplays

1.4 Verortung in der Industrie

Modulare KI-Systeme sind zwar äußerst flexibel und ermöglichen die Wiederverwendung und Anpassung von KI-Verhaltensweisen über verschiedene Entitäten hinweg, sie bringen jedoch auch eine zusätzliche Komplexität mit sich. Für kleine Projekte oder Spiele mit sehr ähnlichen KI-Verhaltensweisen kann dieser zusätzliche Aufwand nicht gerechtfertigt sein. In solchen Szenarien kann es effizienter und schneller sein, die KI für jede Entität spezifisch zu gestalten, insbesondere wenn die Variationen in den Verhaltensweisen minimal sind.

Dennoch gibt es Beispiele von Videospielen, wie SPORE, indem modulare KI Verwendung findet. Hier werden Kreaturen durch das Anstecken von Körperteilen zusammengesetzt, wodurch ihre Verhaltensweisen und Eigenschaften, wie zum Beispiel deren bevorzugte Nahrungsquelle, beeinflusst wird. Auch Armored Core VI: Fires of Rubicon zeigt, dass modulare Systeme ihren Nutzen haben. Hier wählt der Spieler die Körperteile seines Mechs aus, was die Fähigkeiten, Waffen und Bewegungsmöglichkeiten verändert.

Modulare KI ist besonders vorteilhaft für Open-World-, Echtzeit-Strategie- und Simulationsspiele, da sie dynamische und anpassungsfähigere Verhaltensweisen ermöglicht. Viele Spiele kombinieren deterministische und nicht-deterministische Ansätze. Dieser hybride Ansatz hilft Entwicklern, ein Gleichgewicht zwischen Vorhersehbarkeit und Anpassungsfähigkeit zu finden und ein abwechslungsreiches Spielerlebnis zu schaffen.

Die Entwicklung eines modularen KI-Systems in der UE könnte der Videospielbranche von Nutzen sein und anderen Entwicklern das Erstellen von KI vereinfachen. Obwohl ein paar ähnliche Systeme in Projekten bereits verwendet werden, ist dies nach wie vor eine Nische deren Erfolg und Implementierung stark von den Projektanforderungen abhängt.

1.5 Zielgruppen

Entwickler: Das Projekt richtet sich an Entwickler, die nach einer innovativen, leicht erweiterbaren Lösung für modulare KI suchen. Es bietet eine effiziente Methode zur Gestaltung dynamischer Verhaltensweisen und kann das Spielerlebnis von der Konkurrenz abheben.

Spieler: Spieler, die sich für Videospieltechnologien interessieren, können hier neue Herangehensweisen kennenlernen und ihr Verständnis für KI vertiefen. Fans von herausfordernden Gegnern und Action-Spielen haben die Möglichkeit, ihre Fähigkeiten praktisch zu testen.

Portfolio: Das Projekt stärkt mein Portfolio durch die Demonstration meines Könnens in der Entwicklung komplexer, modularer KI. Es zeigt potenziellen Arbeitgebern meine Kompetenzen und erhöht meine Attraktivität als Mitarbeiter.

2 Kontext

2.1 Künstliche Intelligenz

2.1.1 Definition von künstlicher Intelligenz

Die Natur der Intelligenz ist umstritten, und es ist noch ungelöst, inwiefern maschinelle Intelligenz der menschlichen ähnelt. Intelligenz umfasst die Problemlösung, die Informationsverarbeitung, das Lernen sowie die Entscheidungsfindung. Bei KI geht es darum, Maschinen zu entwickeln, die diese Fertigkeiten besitzen. Das Argument, dass Maschinen aufgrund fehlenden Bewusstseins oder Emotionen nicht dieselbe Intelligenz wie Menschen erreichen können, bleibt weiterhin bestehen (Görz and Nebel, 2014; Kaplan, 2017).

KI lässt sich in vier Typen unterteilen: reaktive, begrenzte Gedächtniskapazität, Theory of Mind und selbstbewusste Systeme (Hintze, 2016; Mitchell, 2020; Dorr, 2022). In Videospielen ist KI meist reaktiv, reagiert auf Eingaben und liefert vorhersehbare Ergebnisse, ohne zu lernen oder Erinnerungen zu haben (Champandard, 2004; Weber, n.d.). Beispiele für reaktive KI sind Nicht-Spieler-Charaktere, sowie die dynamische Anpassung der Spielwelt wie bei „Left 4 Dead“ (Development of Left 4 Dead by Mike Booth GDC Lecture, 2023) und taktische Systeme wie in „Starcraft“ (Uriarte and Ontañón, 2015).

Reaktive KI durch Nicht-Spieler-Charaktere ist entscheidend für die Umsetzung von dynamischen und immersiven Spielerlebnissen. Diese können in Echtzeit auf Spieleraktionen reagieren, etwa indem sie fliehen, Schutz suchen oder angreifen, je nach dem Verhalten oder der Position des Spielers. (Champandard, 2004; Millington, 2020).

Diese KI interagiert auch mit der Spielwelt, indem sie Türen öffnet oder Leitern benutzt und sich an Umweltveränderungen wie Wetter oder Tageszeit anpasst. Solche Verhaltensweisen tragen erheblich zur Tiefe und Komplexität moderner Videospiele bei (Millington, 2020).

2.1.2 Bewältigung zunehmender Komplexität durch KI-Frameworks

In den frühen Jahren der Videospielindustrie gab es keine standardisierten KI-Algorithmen. Die Aliens in „Space Invaders“ (1978) und die Geister in „Pac-Man“ (1980) nutzten einfache, heuristische Ansätze. Mit der zunehmenden Komplexität von 3D-Spielen wie „Doom“ (1993) wurden fortschrittlichere KI-Techniken notwendig. Entwickler begannen, eigene Frameworks und Bibliotheken zu erstellen, inspiriert von etablierten Methoden aus anderen Industrien (Kartsios, 2022; Perron et al., 2022). Ab den 2000er Jahren stieg die Nutzung von KI-Frameworks enorm, sodass auch etablierte Entwicklungsanwendungen wie die UE Algorithmen, wie das Behavior Tree Framework, einführte (Newton and Feng, 2016).

Die meist verwendeten Frameworks ermöglichen es, komplexe Entscheidungslogiken in modulare, wiederverwendbare Teile zu zerlegen, was die Entwicklung beschleunigt, und die Wartung vereinfacht (Buckland, 2005; Safadi, Fonteneau and Ernst, 2015).

2.1.3 Gegenüberstellung bekannter KI-Frameworks

Zu den häufigsten KI-Frameworks zählen Finite State Machines, Utility AI, Goal Oriented Action Planning und Behavior Trees. Je nach Anwendungsfall können diese Frameworks auch kombiniert werden, um verschiedene Aspekte eines KI-Systems zu steuern (Bourg and Seemann, 2004; Buckland, 2005).

Die Auswahl des passenden Frameworks hängt von mehreren Faktoren ab. Zunächst muss das spezifische Problem, das die KI lösen soll, identifiziert werden. Die Art und Komplexität des Problems bestimmen meist, welches Framework am besten geeignet ist (Millington, 2020).

Vor- und Nachteile der KI-Frameworks

- Finite State Machine

Die Implementierung von Finite State Machines ist einfach und bietet eine klare Struktur. Sie eignet sich gut für einfache Verhaltensweisen. Bei komplexeren Verhaltensmustern oder in Spielen mit vielen Interaktionsmöglichkeiten kann es jedoch schnell unübersichtlich werden und an Grenzen stoßen. Da Zustände und Übergänge vorab definiert sind, kann es schwierig sein, auf dynamische Spieleraktionen und Umgebungsveränderungen angemessen zu reagieren (Bourg and Seemann, 2004; Jagdale, 2021).

- Utility AI

Utility AI bewertet Aktionen basierend auf eingestellten Faktoren und errechnet eine Gewichtung, was zu flexibleren und realistischeren Verhaltensweisen führt. Sie ist gut für komplexe Szenarien geeignet, wo Finite State Machines möglicherweise nicht ausreichen. Allerdings kann die Implementierung zeitaufwendig sein, und der Berechnungsaufwand für die Nutzenfunktionen kann in komplexen Szenarien die Leistung beeinträchtigen (Rabin, 2020; Winding Road Ahead: Designing Utility AI with Curvature, 2021).

- Goal Oriented Action Planning

GOAP ermöglicht flexible Reaktionen auf sich ändernde Bedingungen und fördert klare Zielsetzungen, da Ziele nach Wichtigkeit priorisiert werden. Durch gezielte Planung können effiziente Lösungen für komplexe Probleme gefunden werden. Die Implementierung erfordert jedoch ein tiefes Verständnis der Planungsalgorithmen und eine sorgfältige Definition der Aktionen und Zustände, was zu höherem Entwicklungs- und Wartungsaufwand führen kann.

Zudem kann die Skalierbarkeit eine Herausforderung darstellen, da die Anzahl der Zustände und Aktionen in komplexen Umgebungen schnell wachsen kann (Jacopin, n.d.).

- Behavior Tree

BTs bieten eine modulare Struktur, die Wiederverwendung und Anpassung an sich ändernde Situationen ermöglicht. Die klare Hierarchie und visuelle Darstellung erleichtert die Fehlersuche. Die Planung erfordert jedoch Erfahrung, und komplexe Bäume können zu hoher Berechnungslast führen (Colledanchise and Ögren, 2018; Champandard and Dunstan, n.d.).

Übersicht

Framework	Vorteile	Nachteile
Finite State Machine (FSM)	• Einfache Implementierung • Klare Struktur • Gut für einfache Verhaltensweisen	• Schnell unübersichtlich bei komplexen Verhaltensmustern • Schwer an dynamische Änderungen anzupassen
Utility AI	• Flexible und realistische Verhaltensweisen • Gut für komplexe Szenarien	• Zeitaufwendige Implementierung • Hoher Berechnungsaufwand in komplexen Szenarien
Goal Oriented Action Planning (GOAP)	• Flexible Reaktionen auf sich ändernde Bedingungen • Klare Zielsetzungen und effiziente Lösungen	• Hoher Entwicklungs- und Wartungsaufwand • Skalierbarkeit kann eine Herausforderung darstellen
Behavior Tree (BT)	• Modulare Struktur • Wiederverwendbarkeit • Erleichtert die Fehlersuche durch visuelle Darstellung	• Erfordert Erfahrung bei der Planung • Hohe Berechnungslast bei komplexen Bäumen

Tabelle 1: Vor- und Nachteile von KI-Frameworks

2.1.4 Das Behavior Tree Framework in der Unreal Engine

In der UE bilden mehrere Komponenten die grundlegenden Bausteine einer Künstlichen Intelligenz: die Charakter- und AIController-Komponenten sowie das Blackboard und das BT Asset. Die Charakterkomponente steuert die Bewegung, Animation, Physik und Navigation des Charakters, während der AIController die Entscheidungen der KI basierend auf dem BT umsetzt und Sinneswahrnehmungen integriert. Ein weiteres zentrales Element ist das Navigation Mesh System, das der KI ermöglicht, sich in der Spielwelt zu bewegen, indem Navigationsbereiche definiert und zur Laufzeit dynamisch angepasst werden können (Newton and Feng, 2016).

Das BT Asset definiert den Verhaltensbaum der KI und wird im Editor von UE erstellt. Das Blackboard dient dabei als zentrale Datenbank, die Informationen zwischen dem Verhaltensbaum und den aktuellen Aufgaben austauscht. Der Verhaltensbaum beginnt mit einem Wurzelknoten, der bestimmt, welcher Ast des Baums basierend auf den gegebenen Bedingungen und dem aktuellen Spielzustand als nächstes ausgeführt wird (Anon., 2024a).

Innerhalb des Verhaltensbaums organisieren zusammengesetzte Knoten den Ablauf und bestimmen die Ausführung von Unterzweigen, wie beispielsweise durch Sequenzen, Selektoren oder Parallelknoten. Dekoratorknoten überwachen und beeinflussen das Verhalten, während Aufgabenknoten spezifische Aktionen basierend auf vorherigen Entscheidungen ausführen. Serviceknoten hingegen führen regelmäßig Überprüfungen oder Aktualisierungen im Blackboard durch. Das gesamte System ist eventgetrieben, um unnötige Rechenarbeit pro Frame zu vermeiden und reagiert passiv auf eintretende Ereignisse (Anon., 2024a).

UE ermöglicht die Erstellung von BTs sowohl visuell über Blueprints als auch programmatisch in C++. Die Engine stellt eine Vielzahl von Knoten bereit, die verschiedene Verhaltensweisen wie Zielanvisierung, Animationen oder das Auslösen von Ereignissen steuern. Im Vergleich zu anderen Implementierungen verwendet UE einfache Parallelknoten, die nur dann Kindknoten verarbeiten, wenn ein Hauptpfad in dem Moment ebenfalls durchlaufen wird (Anon., 2024a).

Ein spezieller Knoten namens „Run Behavior Dynamic“ erlaubt es, zur Laufzeit Teilbäume (Subtrees) zu starten oder zu laden, ohne den gesamten BT neu initialisieren zu müssen. Dies erhöht die Modularität und Anpassungsfähigkeit der KI. Zudem können sogenannte Injektion Tags definiert und Knoten zugewiesen werden, um diese per Code abzuändern (Anon., 2024a).

2.2 Gegner in Videospielen

2.2.1 Geschichte des Gegner Designs

Bereits in den frühen textbasierten Spielen wie „Zork“ (1977) begegneten Spieler Gegner. Diese waren oft Teil von Rätseln, jedoch eher geschichtenbezogen als interaktiv. Mit „Space Invaders“ (1978) traten erstmals Gegner auf, die aktiv auf die Spieler zukamen und abgeschossen werden mussten. Diese Gegner folgten festen Bewegungsmustern und wurden zunehmend herausfordernder (Kartsios, 2022).

„Pac-Man“ (1980) brachte Gegner mit unterschiedlichen Verhaltensmustern und Bewegungsstrategien, die eine taktische Herangehensweise erforderten. Sie jagten den Spieler, blockierten den Weg oder bewegten sich zufällig. Diese Differenzierung war ein Fortschritt im Gegner-Design und beeinflusste spätere Spiele wie „Super Mario Bros“ (1985) (DeNero and Klein, 2010).

In den 1990er Jahren, mit „Super Metroid“ (1994) und „Sonic the Hedgehog“ (1991), reagierten Gegner auf Spieleraktionen, was die Interaktivität erweiterte (Kartsios, 2022). Seit den 2000er Jahren liegt der Fokus auf Realismus und dynamischen Interaktionen. „Doom“ (2016) zeigt diese Weiterentwicklung mit 76 verschiedenen Gegnertypen und einem komplexen KI-System für realistisches Gameplay (The AI of DOOM (2016) | AI and Games #30, 2018; The AI of DOOM (1993) | AI and Games #66, 2022)

2.2.2 Die Rolle der Gegner

Gegner stellen für Spieler Herausforderungen dar, indem sie Hindernisse bieten, die durch Kampf, Geschicklichkeit oder Strategie überwunden werden müssen. Sie sind oft dazu da, dem Spieler die Spielmechaniken und -fähigkeiten beizubringen. Früh im Spiel auftauchende Gegner sind meist einfacher, um das Lernen der Grundlagen zu unterstützen. Mehrere Gegnertypen gleichzeitig erhöhen die Schwierigkeit und fördern durch ihre Vielfalt in Angriffsmustern, taktische Komplexität und Koordination neue Drucksituationen, was den Spielspaß steigert (Kartsios, 2022).

Das Design von Gegnern ist häufig eng mit der Geschichte und der Spielwelt verknüpft. Ihr Aussehen, Sound und Animationen reflektieren oft die Geschichte und die Welt des Spiels. Gegner spielen eine zentrale Rolle in der Handlung und tragen zur Entwicklung der Story bei. Sie fungieren auch als Belohnungssystem, indem Spieler für das Besiegen von Gegnern Belohnungen wie Erfahrungspunkte, Gegenstände oder neue Fähigkeiten erhalten (Agriogianis, 2018).

2.2.3 Dynamische Gegner und deren Modularität

Eine flexible und anpassungsfähige KI-Architektur ermöglicht es Gegnern, ihr Verhalten basierend auf verschiedenen Faktoren zu ändern, wodurch das Spielerlebnis abwechslungsreicher wird. Bei der modularen Implementierung werden Verhaltensweisen in separaten Modulen kodiert, die unabhängig entwickelt und kombiniert werden können.

Jedes Modul steuert einen bestimmten Aspekt wie Angriff, Verteidigung oder Flucht und kann je nach Spielbedingungen aktiviert oder deaktiviert werden. Diese Struktur erlaubt Entwicklern, die KI flexibel anzupassen und zu erweitern. Neue Module können hinzugefügt werden, ohne das gesamte System zu verändern, was die Entwicklung und Wartung der KI erleichtert (Buckland, 2005; Spronck et al., 2006; Millington, 2020).

2.3 Referenzanalyse - Armored Core VI: Fires of Rubicon

„Armored Core VI: Fires of Rubicon“ ist ein von FromSoftware entwickeltes und von Bandai Namco Entertainment veröffentlichtes 3D-Actionspiel aus dem Jahr 2023. Es bietet herausfordernde Gameplay-Elemente und eine düstere Atmosphäre. Spieler können aus dreihundertsechs Teilen wählen, um zwölf Ausrüstungsslots ihres Mechs anzupassen, wodurch ein hohes Wiederspielwert entsteht (Armored Core 6 Interview with Yamamura & Ogura, Featuring Armored Core Legacy, 2023).

Abbildung 1: verfügbare Ausrüstungsslots im Assembly in Armored Core VI, Fires of Rubicon

In den Spielmodi von „Armored Core VI: Fires of Rubicon“ trifft man auf Gegner, die aus verschiedenen Komponenten bestehen. Die große Menge an Feinden und die Tatsache, dass diese Komponenten nach dem Besiegen des Gegners für die Anpassung des Spielercharakters genutzt werden können, deuten darauf hin, dass die KI möglicherweise durch ein modulares System unterstützt wird.

Gegner haben grundlegende Mechaniken wie Lebenspunkte und Verhaltensweisen wie Angriffe, deren Ausführung durch spezifische Komponenten ermöglicht und definiert wird. Diese vorgestellte Modularität inspiriert mich, meine eigene KI zu entwickeln und bietet Flexibilität und Anpassungsfähigkeit in der KI-Gestaltung.

Aus diesem Spiel ziehe ich ebenfalls Inspiration, um zu entscheiden, welche Körperteile und -varianten in meinem eigenen Projekt vorhanden sein werden. Ein Beispiel ist die Vielzahl an Möglichkeiten bei der Auswahl der Beine. Es gibt vierzehn verschiedene Beinteile, die in vier unterschiedliche Beinarten kategorisiert werden: zweibeinig, mit umgekehrtem Gelenk, vierbeinig und Panzer. Der Typ definiert dabei die Bewegungsmechanik, wobei das Verhalten der KI erheblich verändert wird.

Abbildung 2: die vier Bein Typen in Armored Core VI, Fires of Rubicon

Die Tiefe der Anpassungsmöglichkeiten stellt sicher, dass Spieler ihren Mech nicht nur visuell, sondern auch funktional an ihre bevorzugte Spielweise anpassen können. Die modulare Struktur ermöglicht es, dass jede Entscheidung hinsichtlich der Ausrüstung des Mechs unmittelbare Auswirkungen auf das Gameplay hat. Diese sorgfältige Balance zwischen Anpassung und Strategie fördert nicht nur die kreative Freiheit der Spieler, sondern stellt auch sicher, dass jeder Mech einzigartig ist und auf spezifische Herausforderungen im Spiel zugeschnitten werden kann. Dies inspiriert mich dazu, ähnliche Mechaniken zu entwickeln, die eine vergleichbare Tiefe und Vielseitigkeit bieten, jedoch durch das Design der Gegner ermöglicht wird.

3 Methodik

3.1 Evaluation, Arbeitsphasen und Meilensteine

In der Methodik wird die geplante Herangehensweise zur Realisierung des Medienprojekts beschrieben. Die Umsetzung erfolgt in mehreren aufeinanderfolgenden Arbeitsphasen, wobei der Abschluss einer Phase gleichzeitig als Meilenstein fungiert.

3.1.1 Übersicht über die Arbeitsphasen

Die Arbeitsphasen lauten wie folgt:

- Arbeitsphase 1: *Pre-Production*
- Arbeitsphase 2: *Entwicklung der KI-Architektur*
- Arbeitsphase 3: *Prototyp Implementierung*
- Arbeitsphase 4: *Polishing*

Je nach Schwerpunkt der Arbeitsphase arbeite ich mit Supervisoren, die bei diesem Thema bereits eine große Menge an Erfahrung haben und mir qualitatives Feedback sowie Unterstützung geben können.

Die Supervisoren lauten wie folgt:

Christian Schaal, Geschäftsführer von Terovania UG und Projektleiter von Pentaquin: Deeds of Twilight, überwacht die Arbeitsphasen eins, drei und vier.

Ralf Schilderoth, Artificial Intelligence Programmer von Pentaquin: Deeds of Twilight, überwacht die Arbeitsphasen zwei, drei und vier.

3.1.2 Evaluationsmethoden

In jedem Abschnitt meines Arbeitsprozesses verwende ich speziell ausgewählte Evaluationstechniken, um sicherzustellen, dass ich kontinuierlich Fortschritt mache und meine Ergebnisse verbessere.

Die Evaluationsmethoden lauten wie folgt:

Selbsttests (verwendet in den Arbeitsphasen eins, zwei und drei)

Bei den Selbsttests mache ich es mir zur Aufgabe die wichtigen Bewertungskriterien vorab zu planen und diese selbstkritisch zu evaluieren. Die Evaluation erfolgt noch während dem Zeitrahmen der Phase, um frühzeitig reagieren zu können und nötige Iterationsprozesse einzuleiten.

Geschlossener Test (verwendet in der Arbeitsphase drei)

Die geschlossenen Tests erfolgen durch Freunde und andere Studierende in einem festen Zeitrahmen.

Um die Überprüfung von und Feedbackeinholung zu bestimmten Bewertungskriterien zu gewährleisten, kommuniziere ich persönlich mit den Testern und schreib wichtige Kenntnisse auf, wobei meine Fragen bei der Planung bereits definiert werden.

Öffentlicher Test (verwendet in der Arbeitsphase vier)

Die Testphase erfolgt über die Vertriebsplattform Itch.io. Tester werden über Instagram, Reddit, über die SAE sowie durch Freunde und Bekannte angeworben. Es wird auf einen Fragebogen hingewiesen, um spezifische Kriterien zu überprüfen.

Feedbackrunden

Diese Methode erfolgt über die SAE in zwei bis drei wöchentlichen Abständen über den gesamten Zeitrahmen des Projekts. Um qualitatives Feedback zu erhalten, bereite ich das Projekt so vor, dass Fragen gestellt und neu erarbeitetes vorgestellt werden kann. Rückmeldungen werden notiert und in die Iteration der momentanen Arbeitsphase integriert.

3.1.3 Arbeitsphasenplanung, Meilensteine und Evaluation

Arbeitsphase 1: Pre-Production

Vorhaben

In der ersten Arbeitsphase liegt der Fokus bei der umfassenden Recherche und der Entwicklung von Dokumenten, Diagrammen und Ablaufplänen. Ein Design Dokument, sowie ein Programmablaufplan, dienen als Grundlage für die spätere Implementierung. Die erstellten Ressourcen beeinflussen maßgeblich die Umsetzung des praktischen Projekts.

In den letzten Schritten der Arbeitsphase werden Third-Party-Assets ausgewählt, ein UE Projekt vorbereitet, Assets importiert und das Projekt mit Versionsverwaltungssoftware verknüpft.

Meilenstein

Der Meilenstein besteht darin, einen Plan ausgebaut zu haben, der leicht verständlich ist und als Leitpfaden fungiert.

Evaluation

Die Evaluation erfolgt als *Selbstevaluation* sowie durch einen Supervisor und geschieht nach der Recherche und der Erstellung der Dokumente. Hierbei wird die Qualität und die Ganzheit der Dokumente nochmals überprüft und überarbeitet.

Arbeitsphase 2: Entwicklung der KI-Architektur

Vorhaben

Diese Arbeitsphase konzentriert sich auf die Umsetzung der KI innerhalb der UE. Es gilt ein System zu entwickeln, das zufällig ausgewählte Körperteile zusammensetzt, um einen Gegner zu generieren. Hierbei wird stets auf die in Kapitel 1.3 erstellten Ziele geachtet und mehrere Körperteilvarianten pro Körperbereich erstellt. Die visuellen Eigenschaften der KI werden, durch die in der Pre-Production ausgewählten Third-Party-Assets realisiert.

Meilenstein

Der Meilenstein besteht darin, eine modulare Grundlage für die Generierung der KI bestehen zu haben. Das System ist komplett, Bugs sind behoben und es kann nun in einem Projekt angewendet werden.

Evaluation

Die Evaluation erfolgt nach der Implementierung der Features, um ausreichendes Material zum Testen zu haben.

Zu evaluierende Aspekte:

- Ist das System leicht erweiterbar und modular?
- Sind die Körperbereiche sinnhaft ausgewählt?
- Sind die Körperteilvarianten sinnhaft ausgewählt?
- Funktioniert das System Fehlerfrei?

Arbeitsphase 3: Prototyp Implementierung

Vorhaben

Die Arbeitsphase beinhaltet die Implementierung eines First Person Shooter Prototyps und die Einbringung der KI als Gegner, wobei Sieg- und Niederlagebedingungen anhand von Lebenspunkten und dem Besiegen der anderen Partei definiert werden. Hierbei wird die Spielersteuerung und -reaktion auf die KI, das Level und der Spielloop entwickelt. Um weiterhin den Arbeitsfokus auf die KI legen zu können, werden für diese Features überwiegend Third-Party-Assets angewandt.

Meilenstein

Der Meilenstein besteht darin, ein funktionsfähiges Spiel-Loop zu implementieren. Im Projekt soll das erstellte modulare System klar zum Vorschein kommen.

Evaluation

Die Evaluation erfolgt nach der Implementierung des Spiel-Loops. Dabei werden ein *Selbsttest* und ein *geschlossener Test* angewandt.

Zu evaluierende Aspekte:

- Ist die Steuerung des Spielers flüssig und intuitiv?
- Sind die Interaktionen mit dem Gegner Intuitiv und Spaßbringend?
- Werden die Mechaniken klar kommuniziert?
- Sind die verwendeten Körperteilvarianten visuell unterscheidbar?

Arbeitsphase 4: Polishing

Vorhaben

Die Arbeit in dieser Phase beinhaltet die Implementation von visuellen sowie auditiven Effekten und Eigenschaften, wobei diese durch Third-Party-Assets realisiert werden. Hierbei wird besonders darauf geachtet, ob die verschiedenen Verhaltensweisen der Gegner dem Spieler klar kommuniziert werden.

Ebenfalls erfolgt Bugfixing und die Überarbeitung und das Aufräumen des Codes und des Content Ordners in UE.

Meilenstein

Der Meilenstein besteht darin, ein höheres Qualitätsniveau, sowohl in Bezug auf die visuelle und auditive Präsentation als auch auf die technische Umsetzung, zu erreichen. Das Projekt soll Abgabebereit sein.

Evaluation

Die Evaluation erfolgt nach der Implementation der Effekten. Dabei wird ein *öffentlicher Test* über Itch.io angewandt. Es soll die Nutzbarkeit der neuen Features bewerten.

Zu evaluierende Aspekte:

- Sind die visuellen sowie auditiven Effekte ansprechend und tragen Sie zur Lesbarkeit der Mechaniken bei?
- Wird das Projekt Fehlerfrei ausgeführt?
- Sind weitere Änderungen sinnvoll?

3.2 Workflow Techniken

3.2.1 Iterationsprozess

Die Bearbeitung und Qualitätssicherung erfolgt durch *Iteratives Prototyping*, ein Prozess, bei dem ein Prototyp erstellt und getestet wird. Das Feedback aus den Evaluationsphasen wird genutzt, um den Prototypen zu überarbeiten und zu verbessern. Dieser Zyklus aus Testen, Feedback und Verbesserung wird so lange wiederholt, bis ein zufriedenstellendes Produkt entstanden ist. Durch die erforderte Flexibilität dieses Prozesses, kann auf Feedback reagiert und Anpassungen am Prototyp frühzeitig vorgenommen werden.

1. Erstellen	**Der Prototyp wird erstellt, basierend auf den ersten Anforderungen und Spezifikationen.**
2. Testen	Der Prototyp wird getestet, typischerweise durch Benutzer und/oder Stakeholder.
3. Analysieren	Die Ergebnisse der Tests werden analysiert und es wird Feedback gesammelt.
4. Überarbeiten	Es werden Änderungen am Prototyp vorgenommen, basierend auf dem Feedback und den Erkenntnissen aus der Testphase.

Abbildung 3: Iterative Prototyping Modell

3.2.2 Auswahl der Game Engine und des KI-Frameworks

Die Benutzerfreundlichkeit und die vorhandenen Funktionalitäten des BT Frameworks der UE eignen sich sehr gut für die Voraussetzungen des Projekts. Vor allem erlaubt das KI-Framework die dynamische Zusammensetzung eines BTs, was viele der Ziele dieses Major Projects ermöglicht. Die Vorteile eines BTs in Bezug zu anderen Algorithmen, wobei eine große Bandbreite an Verhaltensweisen überschaubar und modular zu implementieren sind, möchte ich hierbei zur Geltung kommen lassen (siehe Kapitel 2.1.3 und 2.1.4).

3.2.3 Hauptsächliche Verwendung von Blueprints

Blueprints in UE bieten viele Vorteile, die die Entwicklung effizienter, zugänglicher und besonders iterierbar machen, vor allem durch die visuelle Schnittstelle. Da Blueprints keine tiefgehenden Programmierkenntnisse erfordern, sind sie besonders nützlich für die Zusammenarbeit mit und Verwendung von Designern und anderen Nicht-Programmierern. Die Art des Projekts, wobei ein System anhand eines Prototypen getestet und vorgestellt wird, deckt sich mit den möglichen Verwendungsgründen von Blueprints (Sewell, 2015; Romero and Sewell, 2022).

Aus diesen Gründen habe ich mich entschieden das Projekt überwiegend in Blueprints zu entwickeln. Da ich das fertige System voraussichtlich auch anderen Spieleentwicklern mit unterschiedlichen Kenntnissen zu Verfügung stellen möchte, können so Designer und Nicht-programmierer einen besseren Einblick in die Funktionsweise des Systems erhalten und es einfacher anpassen oder erweitern.

Ebenso übersteigen meine Kenntnisse im Umgang mit Blueprints die Programmierung in UE C++. Um meine Stärken für das Major Project zu nutzen und das BT System innerhalb der Spieleengine verwenden zu können, ist diese Entscheidung von großem Vorteil.

3.2.4 Aufbau des Behavior Trees

Implementierung von Subtrees

Die KI erhält einen Parent-BT das auf andere Subtrees verweist (siehe Kapitel 2.1.4). Diese Subtrees beinhalten die Verhaltensweisen der Körperteilvarianten. Durch Injektion-Tags werden diese per Laufzeit den Run Behavior Dynamic Knoten zugewiesen.

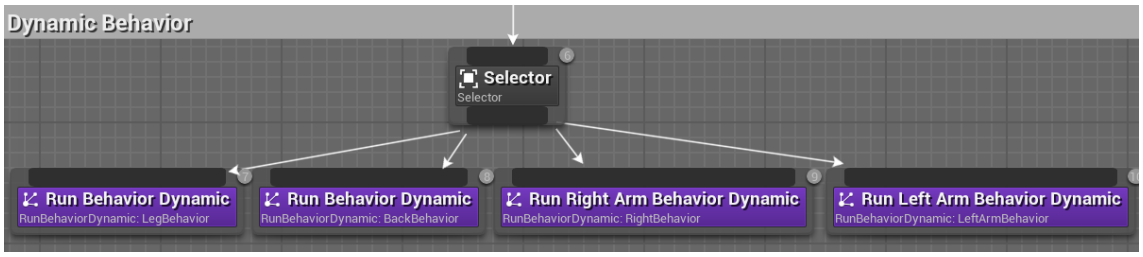

Abbildung 4: Verwendung von Subtrees innerhalb vom Behavior Tree

Implementierung eines Parent-BTs

Es wäre möglich, auf einen Parent-BT zu verzichten und stattdessen die einzelnen Subtrees unabhängig voneinander aufzurufen. Dies hätte die Modularität des Systems weiter erhöht, da eine zusätzliche Schnittstelle vermieden werden würde. Allerdings hätte dies die Möglichkeit weitere, nicht Modulspezifische, Verhaltensweisen zu implementieren erschwert. Nach sorgfältiger Überlegung habe ich mich daher entschieden, den Haupt-BT einzubauen. Dadurch erhält das System mehr Flexibilität und erweitert die mögliche Anwendungsweise. Dies erleichtert ebenso die theoretische Anpassung des Systems für andere Projekte.

Verwendung eines globalen Blackboards

Das Parent-BT sowie alle Subtrees kommunizieren durch eine Blackboard-Instanz, wodurch Informationen zwischen Ihnen geteilt werden können und sie so sinngemäß auf Veränderungen reagieren. Dadurch wird die Notwendigkeit eliminiert, Daten zwischen den Bäumen manuell zu übertragen oder auf andere Weise zu synchronisieren, was zu einer verbesserten Organisation und Effizienz führt. Ein weiterer Vorteil ist die erleichterte Lesbarkeit und Wartbarkeit des Systems, da alle relevanten Daten zentral in einem Blackboard gespeichert werden.

3.2.5 Aufbau der KI

Die wichtigsten Komponenten der KI sind ein benutzerdefiniertes UStruct das die relevanten Daten der Körperteilvarianten beinhaltet, Körperbereich Actors und der hauptsächliche AI Character.

Definieren von Körperteilvarianten durch ein UStruct

Durch das Erstellen eines benutzerdefinierten UStructs können Elemente in einem Datencontainer referenziert werden, die bei der Generierung der KI zum Verändern von deren Eigenschaften genutzt werden. Mehrere solcher UStructs können so in einem Array vorhanden sein, die jeweils ein Körperteilvariant definieren.

Folgende Variablen sind voraussichtlich im UStruct zu bestimmen, um die Erstellung von individualisierten Körperteilvarianten zu ermöglichen:

- Name und Beschreibung vom Körperteilvariant (zur Identifizierung)
- Offset vom Mesh (für die korrekte Positionierung des Objekts)
- Skeletal Mesh vom Körperteilvariant (für die korrekte Darstellung)
- Subtree vom Körperteilvariant (für die Injektion des Subtrees in den Haupt-BT)
- Die minimale Reichweite in der das Verhalten vom Körperteilvariant aktiv sein soll

Mit diesen Variablen kann das Aussehen sowie das Verhalten der KI zusammengesetzt werden und es ermöglicht eine effiziente Erstellung neuer Körperteilvarianten.

Definieren von Körperbereiche und Zuweisung von einem zufällig ausgewählten Körperteilvariant

Die modularen Körperbereiche unterliegen alle denselben Regeln: Sie definieren ein Ort am Körper der KI, dass verschiedene Ihrer Eigenschaften verändern können soll. Somit ist es möglich ein Körperbereich Actor zu erstellen, dass für alle Orte am Körper, die diese Logik verwenden sollen, die gleiche ist. Hierbei kann die Komplexität der hinzugefügten Subtrees ebenfalls variieren und so stark individualisiert werden.

Der erstellte Körperbereich Actor erhält zusätzlich eine Skeletal Mesh Komponente, dass per Laufzeit verändert werden kann.

Folgende Variablen sind voraussichtlich im Körperbereich Actor zu definieren, um die Generierung zu ermöglichen:

- Ein Array mit allen möglichen Körperteilvarianten für diesen Körperbereich (für die zufällige Auswahl von einem Körperteilvariant)
- Der ausgewählte Körperteilvariant (für die Anwendung der UStruct Daten an den Körperbereich Actor bei der Generierung)

Da dieser Körperbereich Actor alle nötigen Daten erhält, um ein Körperteilvariant per Laufzeit auszuwählen, kann dieser einen wichtigen Teil der Generierung selbstständig durchführen:

- Beim Starten des Spiels wird ein Körperteilvariant aus dem Array zufällig ausgewählt.
- Die gewünschte Position des Körperbereich Actors wird durch den Offset errechnet und dieser wird dorthin gesetzt.
- Das Skeletal Mesh wird mit dem gewünschten Asset ersetzt.

Weitere Generierung der KI

Wenn das Spiel startet, führt der AI Character folgende Schritte bei jedem Körperbereich durch:

- Der definierte BT vom zufällig ausgewählten Körperteilvariant wird in das Haupt-BT durch Injektion eingesetzt.
- Die minimale Distanz vom Körperteilvariant zum Spieler, um aktiv zu sein, wird in dem Blackboard eingesetzt.

3.2.6 Workflow Zusammenfassung

Die Struktur der KI ermöglicht es, einen grundlegenden Workflow für die Erstellung von rudimentären weiteren Körperbereichen und Körperteilvarianten zu definieren.

Körperbereich:

1. Körperbereich Actor zu dem AI Character hinzufügen
2. Dessen Array mit Körperteilvariant UStructs füllen
3. Den Haupt-Blueprint durch einen neuen Run Behavior Dynamic Knoten und einen Injektion Tag erweitern

Körperteilvariant:

1. Ein Subtree und, falls nötig, Tasks erstellen
2. Skeletal Mesh Asset importieren
3. Das Array im dazugehörigen Körperbereich Actor erweitern und das neue UStruct mit Daten füllen

Je nach Komplexität der neuen Erweiterung kann es jedoch sein, dass weitere Anpassungen nötig sind.

4 Durchführung

4.1 Pre-Production

4.1.1 Dokumente

Design Dokument

Während dem Erstellen des Design Dokuments machte ich mir Gedanken über das allgemeine Konzept, die verwendeten Tools, die vorhandenen Mechaniken, die verwendeten Assets, die Zielgruppe sowie die Projektplanung. [Einsehbar im Digital Appendix: MajorProjectDesignDokument].

Programmablaufplan

Der erstellte Programmablaufplan diente als allgemeiner Leitfaden und half dabei, eine erste Struktur zu schaffen sowie die verschiedenen Schritte visuell darzustellen, die das Projekt per Laufzeit durchlaufen soll. Er konzentriert sich auf den Gesamtverlauf des Prototyps und nicht speziell auf die Entwicklung der künstlichen Intelligenz. [Einsehbar im Digital Appendix: Programmablaufplan].

Planung der veränderbaren Körperbereiche und Körperteilvarianten

Beim Definieren der möglichen zufälligen Zusammensetzungen bestand der erste Schritt darin, Inspiration für mögliche Körperbereiche und Körperteilvarianten zu bekommen. Durch die Planung und Auseinandersetzung mit ähnlichen Systemen (siehe Kapitel 2.3) konnte ich sicherstellen, dass die verschiedenen Körperbereiche und Körperteilvarianten gute Synergien hervorrufen.

Abbildung 5: Planung der Körperbereiche und Körperteilvarianten

4.1.2 Third-Party-Assets

Gegner Asset

Nach sorgfältiger Recherche entschied ich mich für das Asset „Mech Constructor: Spiders and Tanks“ (Anon., 2024f), das im Unity Asset Store verfügbar war und zusätzlich in anderen Spiele Engines verwendet werden durfte.

Abbildung 6: Beispiele möglicher Zusammensetzungen im Gegner Asset Pack

Das Asset Pack bietet eine Vielzahl von Vorteilen, die es zu einer idealen Wahl für mein Projekt machte. Es enthält eine umfangreiche Sammlung von modularen Komponenten, die speziell für Roboter und Mechs entwickelt wurden. Dazu gehören verschiedene Waffen, Beine und andere essenzielle Teile, die flexibel zusammengefügt werden können.

Abbildung 7: Beispiele modularer Komponenten im Gegner Asset Pack

Es bietet zudem eine Reihe von Animationen, die den Bewegungsabläufen und Aktionen des Gegners Leben einhauchen. Dies sparte mir erheblich Zeit und Aufwand und stellte sicher, dass die visuellen Eigenschaften des Projekts den Zielen entsprechen.

Spieler Asset

Für die Spielersteuerung entschied ich mich, den First Person Shooter Template von UE zu verwenden. Diese Entscheidung wurde getroffen, um den Fokus der Entwicklung auf die KI zu legen und gleichzeitig eine solide Grundlage für das Spielerlebnis zu ermöglichen. Auf dessen Implementierung gehe ich in Kapitel 4.3.1 weiter ein.

Environment Asset

Das Leveldesign erfolgte durch das „Modular SciFi Season 1 Start Bundle“ (Anon., 2024g), welches im Unreal Marketplace kostenlos zu Verfügung stand.

VFX Assets

Bei der Suche nach VFX fiel ich die Entscheidung das Asset „Realistic Starter VFX Pack Vol 2“ (Anon., 2024h), welches im Unreal Marketplace kostenlos zu Verfügung stand, zu verwenden.

Sound Assets

Bei der Suche nach Sound Effekten und Musik fiel ich die Entscheidung die Webseite „WeLoveIndies“ (Anon., 2024i) zu verwenden.

4.1.3 Evaluation der Arbeitsphase Pre-Production

Das Erarbeiten des Design Dokuments und des Programmablaufplans erwies sich als eine gute Stütze.

Die ursprüngliche Planung der Körperbereiche und Körperteilvarianten erwies sich als nicht ausreichend. Durch die Auseinandersetzung mit Armored Core VI: Fires Of Rubicon sah ich, dass Körperbereiche bestimmte Typen unterlegen sind, um balancing sicherzustellen. Die Differenzierung zwischen leichten und schweren Geschossen beispielsweise erlaubt eine ausbalancierte Zusammensetzung von Körperteilvarianten.

Die Recherche und Auswahl der Third-Party-Assets hatte sich als sinnvoll erwiesen, um die nächsten Schritte des Projekts effizient zu planen.

4.1.4 Iteration der Arbeitsphase Pre-Production

Nach der Evaluation folgte eine weitere Iterationsphase, worin die Planung der Körperbereiche und Körperteilvarianten den neuen Erkenntnissen angepasst wurde.

4.1.5 Vorbereitung von Dateien

Nach dem Beenden der Iteration und Evaluation erstellte ich ein UE V4.2.1 Projekt, importierte alle ausgewählten Assets und verband das Projekt mit Github.

4.2 Entwicklung der KI-Architektur

4.2.1 Körperteilvariant UStruct

Innerhalb dieser Arbeitsphase erstellte ich zunächst den UStruct zum Definieren von Körperteilvarianten, so wie in Kapitel 3.2.5 beschrieben.

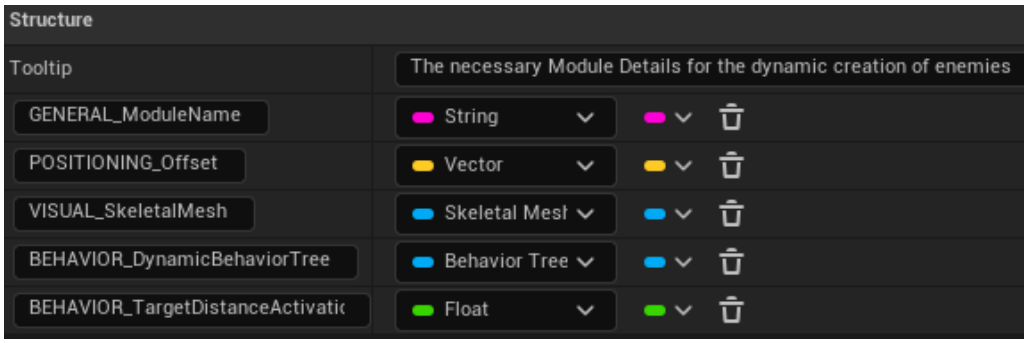

Abbildung 8: Das erstellte UStruct in der Unreal Engine

4.2.2 Körperbereich Actor

Die Grundstruktur der Generierung im Körperbereich Actor konnte ich vorab bereits durch Blueprints erstellen.

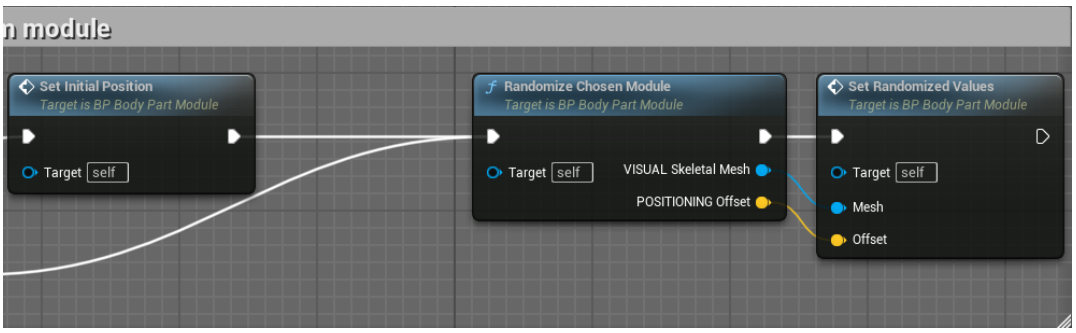

Abbildung 9: Die oberflächliche Logik im Körperbereich Actor

Wenn das Spiel beginnt, wird die initiale Position für die spätere Addition des Offsets gespeichert und ein Körperteilvariant zufällig ausgewählt sowie deren Daten verwendet. Ebenfalls besteht die Möglichkeit ein Event aufzurufen, um den Körperbereich erneut einzustellen.

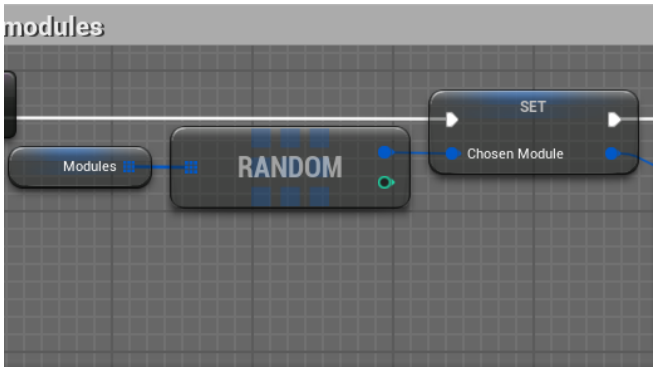

Abbildung 10: Die zufällige Auswahl von einem Körperteilvariant

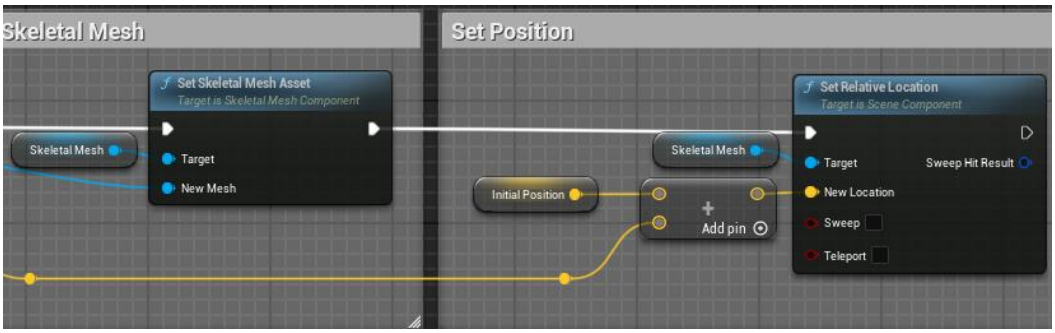

Abbildung 11: Das Einstellen von Werten basierend auf dem zufällig ausgewählten Körperteilvariant

Das Skeletal Mesh des Körperbereichs wird abgeändert und die Position der Komponenten abgeändert.

4.2.3 AI Character

Der AI Character erhielt pro Arm einen Körperbereich Actor. Das Vorgehen, erst einmal nur die Arme als Körperbereich Actor zu implementieren, war bewusst gewählt, um mein Vorhaben inkrementell testen zu können. Ebenso wurde ein BT Asset und ein dazugehöriges Blackboard erstellt. Ein AI Controller Blueprint sorgte für die Initialisierung des BTs.

Abbildung 12: Der AI Character nach dem Hinzufügen der nötigen Komponenten

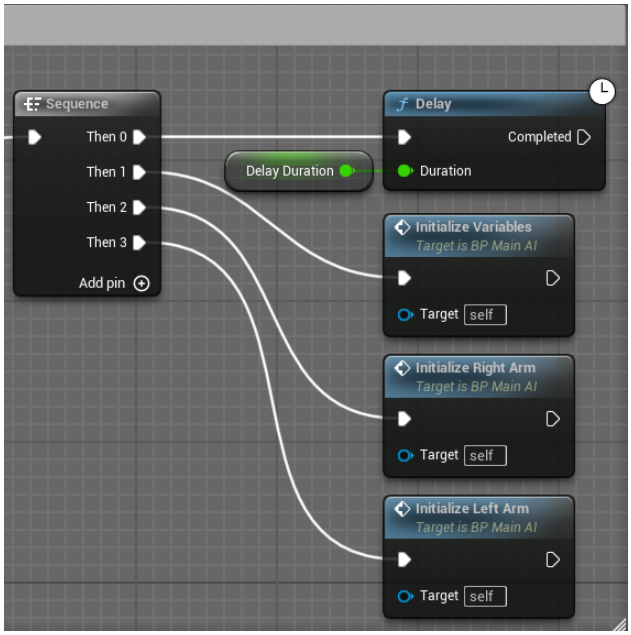

Abbildung 13: Initialisierung von Blackboard und Behavior Tree Variablen

Eine kurze Wartezeit wurde eingebaut, um den Körperbereichen bei der Auswahl des zufälligen Körperteilvariants Vorrang zu gewährleisten. Danach wurden weitere Einstellungen basierend auf dem ausgewählten Körperteilvariant durchgeführt.

Für die spätere Verwendung und den leichteren Zugang wurde das verwendete Blackboard sowie BT als gesonderte Variablen gespeichert.

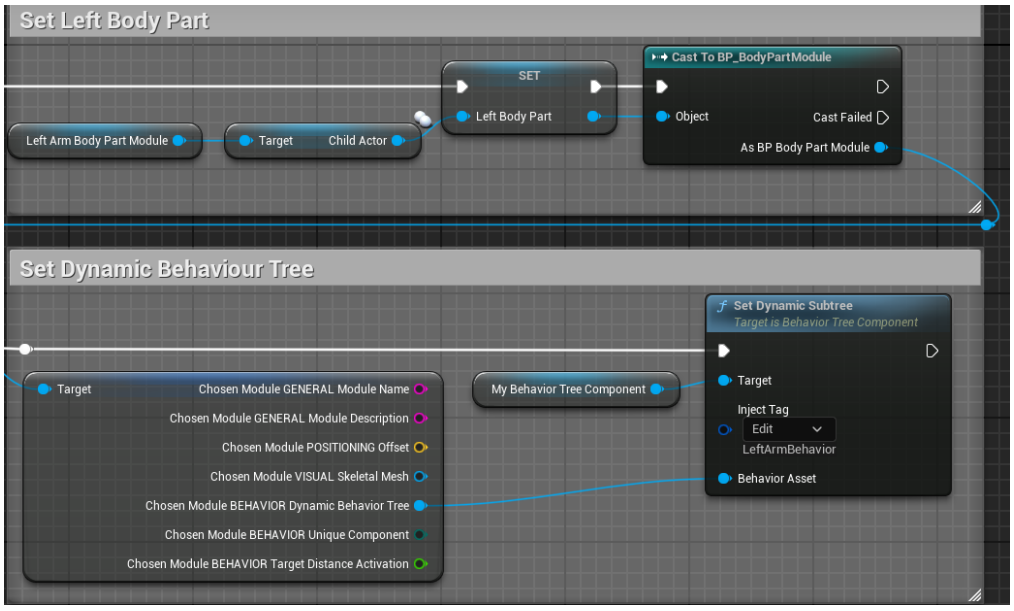

Abbildung 14: Weitere Initialisierung eines Körperbereichs innerhalb vom AI Character 1/2

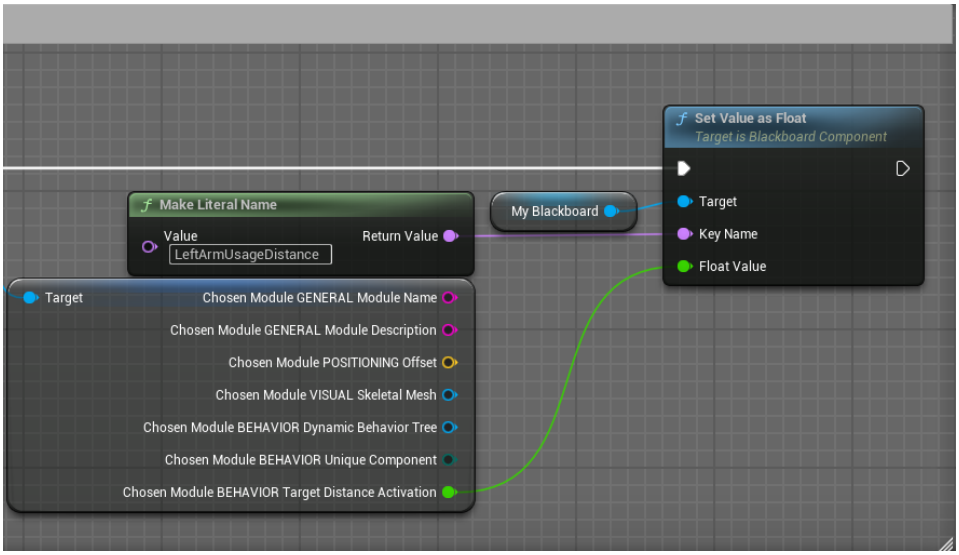

Abbildung 15: Weitere Initialisierung eines Körperbereichs innerhalb vom AI Character 2/2

Für alle Körperbereiche wurden die Körperteilvarianten verwendet, um den dazugehörigen Subtree innerhalb des Haupt-BTs per Laufzeit zu bestimmen.

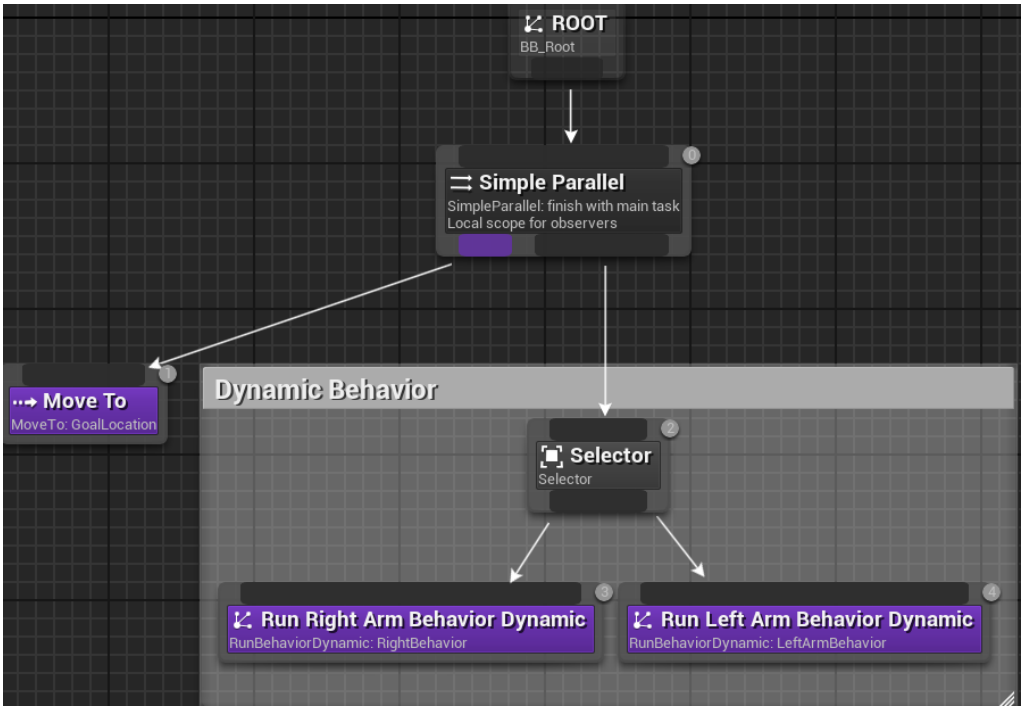

Abbildung 16: Aufbau des Behavior Trees

Um die bisherige Funktionalität testen zu können, erstellte ich ein BT das durch einen Simple Parallel Knoten ermöglicht, dass sich die KI dauerhaft bewegt und gleichzeitig die Subtrees der Arme verwendet.

4.2.4 Körperteilvarianten

Anhand der Planung der Körperteilvarianten entwarf ich zunächst zwei Varianten pro Arm.

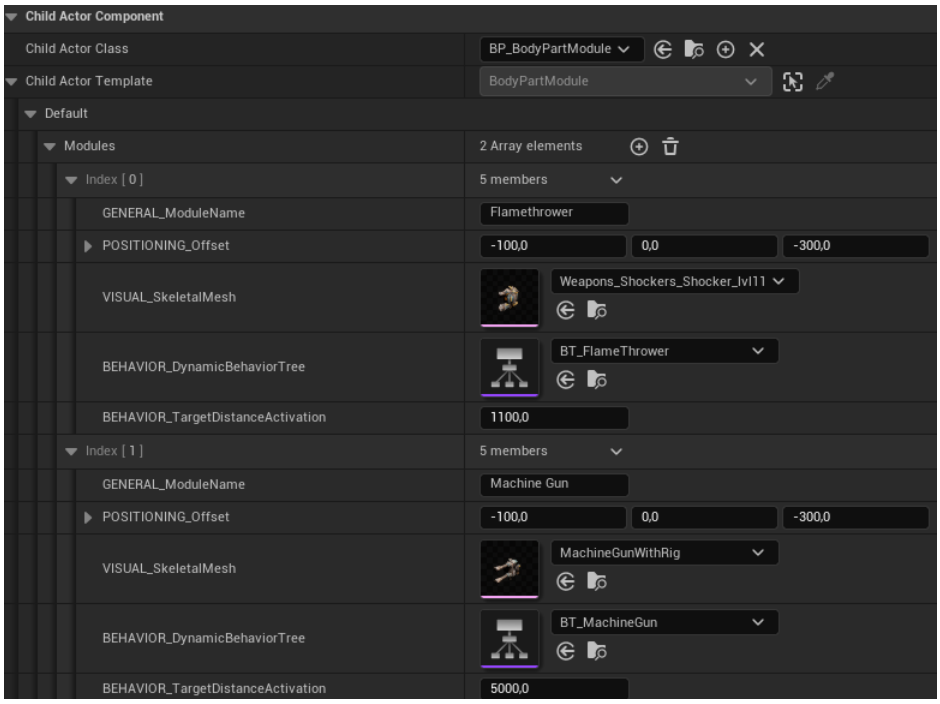

Abbildung 17: Gesetzte Variablen der Körperteilvarianten des linken Arms

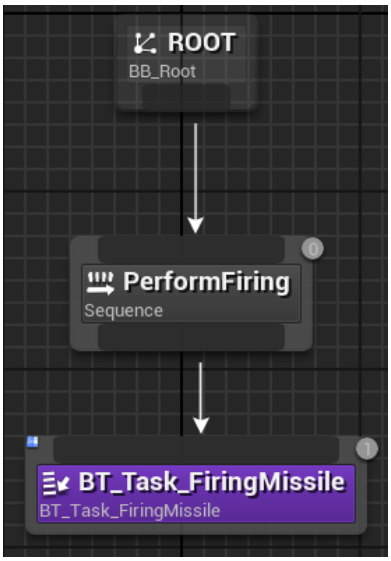

Abbildung 18: Beispiel eines Subtrees anhand des Raketenarms

Die zunächst simplen Subtrees unterstützten die Modularität des Systems, da hierbei die Möglichkeit gegeben ist, weitere Individualisierung der Verhaltensweisen durchzuführen.

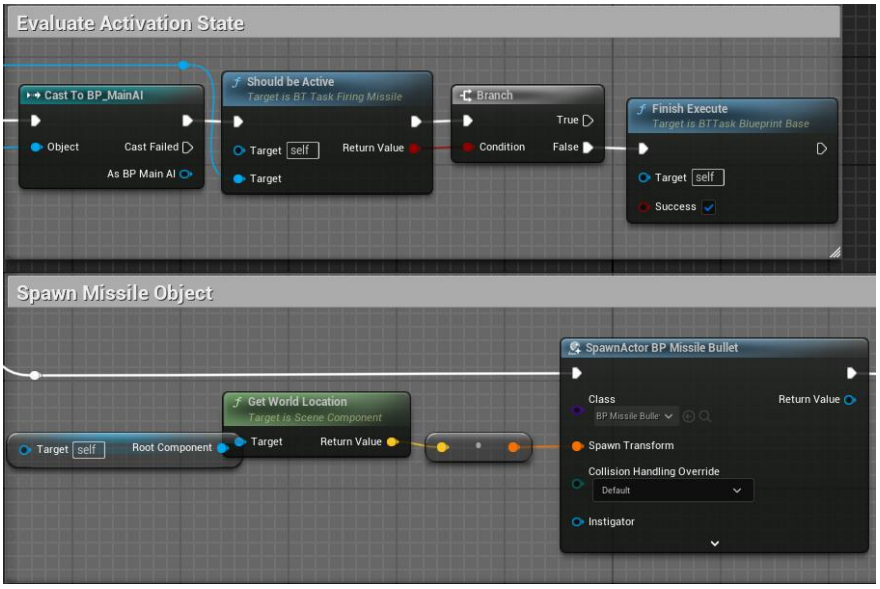

Abbildung 19: Beispiel eines Tasks anhand des Raketenarms zum Abfeuern von Raketen

4.2.5 Evaluation der Arbeitsphase Entwicklung der KI-Architektur

Die Entwicklung der KI-Architektur zeigte sich, durch die vorausgegangene Planung, gut umsetzbar. Die Entscheidung, auf inkrementelle Weise Körperbereiche zu implementieren, minimierte Risiken und ermöglichte eine flexible Anpassung des Systems.

Die zu evaluierende Aspekte wurden wie folgt bewertet:

- Ist das System leicht erweiterbar?

Um die leichte Erweiterbarkeit zu bewerten, folgten weitere Iterationsphasen, wobei neue Körperbereiche und Körperteilvarianten erstellt wurden. Es zeigte sich jedoch bereits ein gut durchdachter Workflow zur Implementierung neuer Körperbereiche und Körperteilvarianten, dass durch die, eher simplen, ausgewählten Erweiterungen der KI unterstützt wurde.

- Sind die Körperbereiche sinnhaft?

Die Körperbereiche wurden als sinnhaft bewertet. Sie dienten alle einem spezifischen, individuellen Zweck. Für die genaue Bewertung sollten die fehlenden Körperbereiche in der nächsten Iterationsphase implementiert werden.

- Sind die Körperteilvarianten sinnhaft?

Die Körperteilvarianten wurden als generell sinnhaft bewertet. Der Raketen- sowie Maschinengewehrarm erfüllte jedoch ähnliche Verhaltensweisen. Für die genaue Bewertung sollten die fehlenden Körperteilvarianten in der nächsten Iterationsphase implementiert werden und eine weitere Evaluation erfolgen.

- Funktioniert das System fehlerfrei?

Die Implementierung zeigte zwei Probleme, die in der Iteration zu lösen galt:

Die Verwendung eines Simple Parallel Nodes im BT hatte zur Folge, dass, wenn mehrere Run Behavior Dynamic Knoten hiermit verbunden sind, diese nicht gleichzeitig ausgeführt wurden.

Die UStruct Datenstruktur erwies sich als nicht funktionsfähig für das Projekt, da sich die vorab definierten Variablen bei erneutem Öffnen des Projekts wieder leerten.

Insgesamt erfüllte die Umsetzung der Arbeitsphase meine Erwartungen. Einige Aspekte hatte ich, wie oben beschrieben, nicht bedacht. Die frühzeitige Evaluation ermöglichte es mir diese zu bemerken und in der Iterationsphase einzupflegen.

4.2.6 Iteration der Arbeitsphase Entwicklung der KI-Architektur

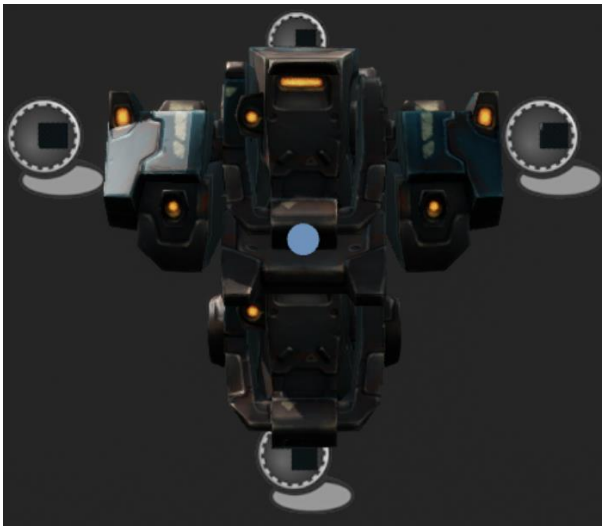

Abbildung 20: Beendete Implementation der vier Körperbereiche in dem AI Character

In der Iterationsphase wurde das System durch zwei weitere Körperbereiche und jeweils zwei Körperteilvarianten erweitert, so wie die Must Planung der Komponenten im Pre-Production definiert wurde, wodurch insgesamt sechszehn Kombinationen möglich waren.

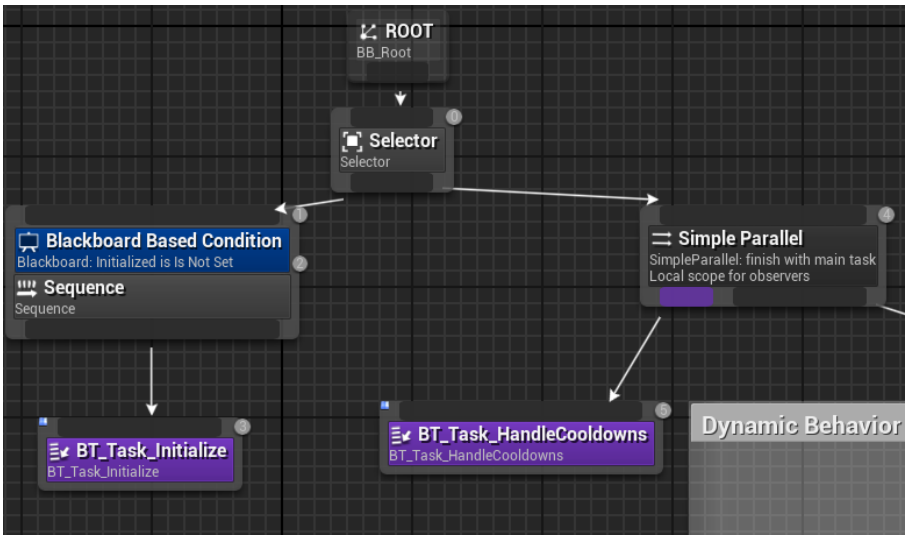

Abbildung 21: Neu hinzugefügte Knoten für die Bearbeitung von Abklingzeiten

Das Einfügen von Abklingzeiten ermöglichte das Durchlaufen mehrerer Run Behavior Dynamic Knoten innerhalb des Haupt-BTs. Es benötigte die Erweiterung des BTs durch Condition Knoten und einer Initialisierung.

Durch die weitere Auseinandersetzung mit UStructs habe ich von einem Mitstudierenden erfahren, dass diese seit einiger Zeit in UE innerhalb des Editors nicht wie vorgesehen funktionieren. Bei der Recherche nach einem Ersatz stieß ich auf die UE spezielle Datenstruktur Data Asset.

Data Assets sind strukturierte Datenspeicher, die es Entwicklern ermöglichen, wiederverwendbare und organisierte Daten in benutzerdefinierten Klassen zu speichern. Diese Assets können in Blueprints und anderen Teilen des Projekts verwendet werden, um Konfigurationsdaten effizient zu verwalten (Anon., 2024c). Der Umstieg hierauf war leicht und das Problem war gelöst.

Der geplante Workflow um neue Körperbereiche und Körperteilvarianten hinzuzufügen zeigte sich während der Iterationsphase als leicht durchführbar.

4.3 Prototyp Implementierung

4.3.1 First-Person-Controller

Das First Person Shooter Template brachte viele Mechaniken bereits mit sich, darunter laufen, springen und schießen. Ich beschleunigte die Bewegung des Controllers, erhöhte die Sprungreichweite und veränderte die Schussart zu Vollautomatik.

4.3.2 Spielloop

Die KI sowie der Spieler bekamen Lebenspunkte, die bei Kollision mit den jeweiligen Schüssen des anderen abgezogen worden sind. Wenn einer der Lebenspunkte auf null ging, wurde ein Sieg- beziehungsweise Niederlagefenster angezeigt und das Level startete neu. Um dieses Vorgehen zu visualisieren und dem Spieler den Game Loop zu verdeutlichen, implementierte ich Widgets mit Lebensbalken.

4.3.3 Generierungsmenü

Um innerhalb des Prototyps die zufällige Erstellung der Gegner zu visualisieren, implementierte ich ein Menü, das bei Spielstart angezeigt wird und worin der Spieler die Möglichkeit hat den Gegner neu generieren zu lassen. Ein Widget mit Textfeldern wird pro Ladevorgang aktualisiert, um Informationen über die aktuell ausgewählten Körperteilvarianten zu zeigen.

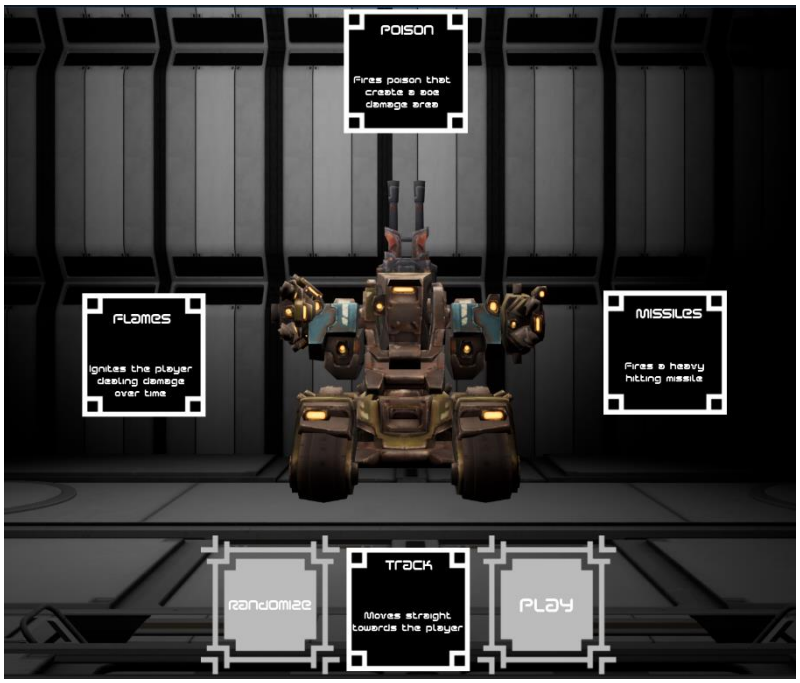

Abbildung 22: Menü zur Visualisierung der zufälligen Gegnergenerierung

4.3.4 Evaluation der Arbeitsphase Prototyp Implementierung

Von den acht Personen, die am geschlossenen Test teilnahmen, sprach ich mit vier Testern.

Die zu evaluierende Aspekte des Selbsttests und des geschlossenen Tests wurden wie folgt bewertet:

- Ist die Steuerung des Spielers flüssig und intuitiv?

Der Spieler bewegt sich in Kontrast zu der KI sehr langsam, wodurch sich die Steuerung nicht flüssig und schleichend anfühlt. Die Steuerungselemente sind intuitiv und leicht verständlich.

- Sind die Interaktionen mit dem Gegner intuitiv und spaßbringend?

Je nach Zusammensetzung der KI ist die Schwierigkeit des Kampfes sehr schwankend. Wenn der Gegner das Maschinengewehr und den Raketenwerfer hat, ist der Kampf sehr herausfordernd. Wiederum wenn der Gegner den Flammenwerfer und den Granatenwerfer hat, ist der Kampf sehr leicht. Ein Balancing der Waffen könnte dies verbessern. Die Interaktionen mit dem Gegner sind jedoch generell spaßbringend.

- Werden die Mechaniken klar kommuniziert?

Auch ohne visuellen sowie auditiven Effekten ist zu erkennen, welche Mechaniken gerade geschehen. Trotzdem kann es schwierig sein, beispielsweise Kugeln zu sehen. Das Implementieren von Lichteffekten könnte dies bereits deutlich verbessern.

- Sind die verwendeten Körperteilvarianten unterscheidbar?

Die Körperteilvarianten sind visuell klar unterscheidbar. Die implementierten Animationen helfen hierbei.

4.3.5 Iteration der Arbeitsphase Prototyp Implementierung

Bei der Iteration passte ich die Geschwindigkeit und die Steuerbarkeit des Spielers an, wodurch die Schnelligkeit des Gameplays sich deutlich veränderte und an Spielen wie Doom Eternal oder Quake erinnerte.

Um das Gleichgewicht bei der Schwierigkeit der verschiedenen Körperteilvarianten zu verbessern, veränderte ich die Schadenspunkte die verursacht werden, verringerte ebenfalls die Geschwindigkeit des Gegners und die Zielgenauigkeit des Granatenwerfers.

Ich entschied mich die weitere Verbesserung der Lesbarkeit der Mechaniken in der Arbeitsphase Polishing durchzuführen, da ich hier VFX sowie SFX implementierte.

4.4 Polishing

4.4.1 SFX

Die Implementierung, die Anpassung und der Feinschliff der Soundeffekte umfasst die Geräusche der Gegner, einschließlich spezifischer Sounds für verschiedene modulare Körperteilvarianten, um das Verhalten und die Aktionen der KI besser zu unterstreichen. Zudem wurde die Hintergrundmusik implementiert und angepasst, um die Stimmung des Spiels zu verstärken und nahtlos in den Spielfluss zu integrieren. Ein weiterer wichtiger Aspekt war das Soundbalancing, bei dem die Lautstärken feinjustiert wurden, um eine ausgewogene Audioerfahrung zu gewährleisten, ohne dass ein Element die anderen übertönt.

4.4.2 VFX

Effekte wurden implementiert und die Intensität angepasst um die Stärke und Häufigkeit der visuellen Effekte, insbesondere bei den Aktionen der Gegner, so zu steuern, dass die visuelle Klarheit gewährleistet bleibt und der Spielfluss nicht gestört wird.

Parallel dazu fand eine Performance-Optimierung statt, bei der die VFX so verbessert wurden, dass sie die Spielperformance nicht beeinträchtigen. Dies umfasste die Reduzierung unnötiger Partikeleffekte.

4.4.3 Bugfixing

Während der Identifikation und Dokumentation wurden alle während des Tests entdeckten Bugs in einem Bug-Tracking-System erfasst und entsprechend ihrer Priorität geordnet. Dieser Schritt war entscheidend, um sicherzustellen, dass die kritischen Fehler, die das Gameplay direkt beeinträchtigen, zuerst behoben werden konnten.

Im Anschluss daran folgte die Fehlerbehebung, bei der die identifizierten Probleme korrigiert wurden, beginnend mit den schwerwiegenden Spielfehlern und endend mit kleineren kosmetischen Problemen. Um die Stabilität des Spiels zu gewährleisten, wurden nach der Fehlerbehebung Selbsttests durchgeführt. Diese Tests dienten dazu, sicherzustellen, dass die vorgenommenen Korrekturen keine neuen Probleme verursachten und dass das Spiel insgesamt stabil läuft.

4.4.4 Evaluation der Arbeitsphase Polishing

Von drei der fünf Tester erhielt ich wertvolles Feedback.

Die zu evaluierende Aspekte des Selbsttests und des öffentlichen Tests wurden wie folgt bewertet:

- Sind die visuellen sowie auditiven Effekte ansprechend und tragen Sie zur Lesbarkeit der Mechaniken bei

Die Effekte sind passend und den unterschiedlichen Mechaniken angepasst und stark individualisiert. Der Schild könnte ebenso ein SFX erhalten, um auch beim nicht Betrachten des Gegners zu wissen, dass dieser aktiv ist.

- Wird das Projekt Fehlerfrei ausgeführt

Mehrere Fehler wurden weitergegeben und in der Iterationsphase bearbeitet. Trotz aktivem Schild kann der Spieler den Gegner verletzten, sollte dieser zu nah am Gegner sein. Je nach Bewegung des Spielers schießt der Gegner die Granaten in die umgekehrte Richtung. Sollten beide Lebenspunkte gleichzeitig auf null fallen, löst dies die Sieg- und Niederlage Fenster gleichzeitig aus.

- Sind weitere Änderungen sinnvoll

Weiteres Balancing wäre sinnvoll. Je nach Beinart ist der Gegner deutlich unterschiedlich schwer zu bezwingen.

4.4.5 Iteration der Arbeitsphase Polishing

Ich habe das Projekt um weitere Soundeffekte erweitert, um die Atmosphäre und das Spielerlebnis zu verbessern. Diese neuen Audioelemente sorgen für eine intensivere und immersivere Klangkulisse.

Zusätzlich habe ich eine Reihe von Bugs behoben, die während der Entwicklung aufgetreten sind. Diese Fehlerbehebungen tragen zur Stabilität und Zuverlässigkeit des Spiels bei.

Darüber hinaus habe ich mich intensiv mit dem Balancing beschäftigt, um sicherzustellen, dass das Spiel fair und herausfordernd bleibt. Dieser Prozess war wichtig, um ein ausgewogenes Gameplay zu gewährleisten, das sowohl Anfänger als auch erfahrene Spieler anspricht.

5 Ergebnisse, Evaluation und Ausblick

5.1 Ergebnisse

Die Ergebnisse des Major Projekts beinhalten ein System und ein sich daraus gebildeter Workflow, um effizient in UE modulare KI zu erstellen, die das BT Framework verwenden.

Durch das Definieren von Körperbereichen, die dem System unterliegen, können individuelle Verhaltensweisen, Mechaniken, Animationen sowie Effekte und Audio für diese implementiert werden, die über ein BT, Subtrees und einem globalen Blackboard miteinander kommunizieren und synergetisch zusammenarbeiten.

Die ermöglichte Modularität wird innerhalb des Projekts durch zufälliges Auswählen von Körperteilvarianten vorgestellt. Insgesamt sind vier Körperbereiche mit jeweils zwei möglichen Varianten vorhanden. Zählt man alle Kombinationen zusammen können sechszehn Gegnerkombinationen generiert werden.

Ein First Person Shooter Prototyp zeigt, dass das modulare KI-System in der Spieleentwicklung einsetzbar ist und zu einer qualitativen Umsetzung führt. Der Nutzer kann gegen die Gegner kämpfen, Ihnen schaden und ebenfalls selbst verletzt werden. Auch wenn der Prototyp ein Bosskampf widerspiegelt, ist das System auch für rudimentäre Gegner geeignet.

5.2 Zielerreichung

5.2.1 Ergebnisse der inhaltlichen Ziele

Priorität	Ziel	Einordnung	Ergebnis ■ Erreicht ■ Teilweise erreicht ■ Nicht erreicht
Must have	*Verhaltensweisen werden durch Körperteilvarianten definiert*	Alle Verhaltensweisen werden durch Subtrees ausgeführt, welche innerhalb der Körperteilvariant spezifischen Data Assets referenziert und bei Spielstart in den BT der KI durch Injection Tags dynamisch hinzugefügt werden. *Somit beinhalten die Körperteilvarianten deren Verhaltensweisen und definieren sie.*	
Must have	*Generierung von Gegnern durch zufällige Zusammensetzung von Körperteilvarianten*	Bei Spielstart werden Elemente eines Körperbereichs abgeändert, sodass ein zufälliger Körperteilvariant widergespiegelt wird. Das Generierungssystem verwendet dafür ein Array von Körperteilvariant Data Assets, wovon eins ausgewählt wird. *Das bedeutet der Gegner wird durch die zufällige Zusammensetzung von Körperteilvarianten generiert.*	
Must have	*Aufteilung und Festlegung der modularen Komponenten in Körperbereiche*	Körperteilbereiche definieren die modular veränderbaren Komponenten der KI, welche sich durch Körperteilbereich Actors zusammensetzen, die innerhalb des Generierungssystems angesprochen werden. *Darum werden die modularen Komponenten der KI durch Körperteilbereiche festgelegt.*	
Must have	*Integration eines First Person Shooter Controllers mit Laufen, Rennen und Schießen*	Das First-Person-Shooter-Template der Unreal Engine bot bereits Mechaniken wie Laufen und Schießen. Die Laufgeschwindigkeit und die Schussart wurden auf Vollautomatik angepasst, jedoch wurde keine Rennmechanik integriert, da der Fokus aufgrund von Scope-Anpassungen auf der KI lag. *Nicht alle gewünschten Bewegungsmechaniken des Controllers sind existent, jedoch die meisten.*	

Must have	*Implementierung von Interaktionsmöglichkeiten zwischen Gegner und Spieler*	Gegner und Spieler interagieren miteinander, wobei der Fokus auf dem gegenseitigen Abschießen liegt. Der Gegner erkennt die Position des Spielers und berücksichtigt diese bei seiner Bewegungsplanung. Diese Mechanik kann der Spieler strategisch nutzen, um den Gegner in Fallen wie Gift oder Granaten zu locken. *Verschiedene Interaktionsmöglichkeiten schaffen eine interessante Dynamik zwischen beiden Parteien, da beide versuchen den anderen zu besiegen und dabei diese Mechaniken gezielt einsetzen.*	
Must have	*Gameplay in Form eines Bosskampfes*	Das Setting, der Lebensbalken des Gegners und das herausfordernde Gameplay fordern den Spieler zu strategischem Handeln heraus und vermitteln das Gefühl eines Bosskampfes. Die Musik und die hohen Lebens- und Schadenspunkte des Feindes verstärken zusätzlich die intensive Atmosphäre. *Die Kombination dieser Elemente sorgt für eine packende Spielerfahrung, in der der Spieler durch geschicktes Vorgehen den Boss besiegen muss.*	
Should have	*Visualisierung der Generierung von künstlicher Intelligenz und ihrer Möglichkeiten durch ein Graphical User Interface*	Beim Spielstart öffnet sich vorerst ein Menü, worin der Spieler die Generierung des Gegners per Knopfdruck auslösen kann. *Die visuelle Veränderung der vorgestellten KI und die Beschreibungen der zusammengesetzten Körperteilvarianten verdeutlichen das System.*	
Should have	*Lebenspunkte für Gegner und Spieler*	Gegner und Spieler verfügen über Lebenspunkte, wobei die KI deutlich mehr aushält. *Die Implementation von Lebenspunkten sowie dessen Visualisierung durch Lebensbalken verdeutlicht das vorgesehene Gameplay.*	

Should have	*Implementierung von Sieg- und Niederlagebedingungen*	Sieg- und Niederlage-Menüs, die nach dem Verlust der Lebenspunkte einer Partei erscheinen, definieren die Bedingungen für das Spielende. *Für jede mögliche Endbedingung des Spielloops wurde eine entsprechende Mechanik implementiert.*	
Could have	*Gegner weist zusätzliche Körperteilvariant unabhängige Verhaltensweisen auf*	Alle Verhaltensweisen des Gegners werden durch spezifische Körperteilvarianten festgelegt. *Zusätzliche Verhaltensweisen wurden nicht weiter implementiert, da sie bei Tests zu Verwirrung führten. Der Spieler erwartet durch das Generierungsmenü alle möglichen Verhaltensweisen zu kennen, was durch unerwartete Variationen beeinträchtigt werden würde.*	
Could have	*Optionen für die benutzerdefinierte Zusammenstellung des Gegners*	Im Generierungsmenü kann der Spieler per Knopfdruck die KI zufällig zusammensetzen lassen, wobei durch mehrfaches Verwenden dieser Option die gewünschte Gegnerkombination erzielt werden kann. *Zwar wurde die Möglichkeit zur zufälligen Zusammenstellung des Gegners implementiert, jedoch fehlt eine direkte Anpassungsoption, was die Benutzerfreundlichkeit einschränkt.*	
Could have	*Ein 3D-Ansichtsmodus zum Betrachten der generierten künstlichen Intelligenz*	Im Generierungsmenü hat der Spieler die Möglichkeit, den Gegner in einer 3D-Ansicht zu sehen. Diese Funktion erlaubt eine visuelle Überprüfung der generierten KI, bietet jedoch derzeit nur eine grundlegende Darstellung ohne zusätzliche Betrachtungsoptionen. *Die Implementierung des 3D-Ansichtsmodus ist theoretisch erreicht, da der Spieler die Möglichkeit hat, den Gegner visuell zu begutachten. Dennoch könnte die Funktionalität durch detailliertere Betrachtungsoptionen und eine umfassendere Darstellung der künstlichen Intelligenz weiter verbessert werden.*	

Tabelle 2: Zielerreichung der inhaltlichen Ziele

5.2.2 Fazit - Inhaltliche Ziele

Von den zwölf inhaltlichen Zielen wurden acht erfolgreich erreicht, drei teilweise und eins wurde nicht umgesetzt.

Die Gegner werden durch zufällige Körperteilvarianten generiert, jedoch fehlt eine gezielte Anpassungsoption für den Spieler, was die Benutzerfreundlichkeit einschränkt. Das Generierungsmenü ermöglicht eine grundlegende visuelle Überprüfung der KI, könnte aber durch detailliertere Betrachtungsoptionen verbessert werden. Diese teilweise Zielverwirklichung ist auf eine Umstrukturierung der Projektprioritäten und Zeitmanagement-Ressourcen zurückzuführen.

Die Integration des First Person Shooter Controllers bietet die wesentlichen Mechaniken wie Laufen und Schießen, jedoch wurde das Rennen aufgrund der Priorisierung der KI nicht integriert.

Außerdem wurden keine zusätzlichen, körperteilvariantenunabhängigen Verhaltensweisen implementiert, da diese bei Tests zu Verwirrung führten. Auch die benutzerdefinierte Zusammenstellung von Gegnern bietet nur eine zufällige Auswahl und nicht die Möglichkeit einer gezielten Anpassung, was die Benutzerfreundlichkeit einschränkt.

5.2.3 Ergebnisse der quantitativen Ziele

Priorität	Ziel	Einordnung	Ergebnis ■ Erreicht ■ Teilweise erreicht ■ Nicht erreicht
Must have	*Generierung von sechs Gegnerkombinationen*	Das Generierungssystem umfasst vier Körperteilbereiche, von denen jeder zwei mögliche Körperteilvarianten bietet. *Insgesamt ergeben sich dadurch sechzehn verschiedene Kombinationen.*	
Must have	*Drei Körperteilvarianten pro Körperbereich*	Die verschiedenen Gegner werden durch zwei Körperteilvarianten pro Körperbereich ermöglicht. *Dies führt zu einer Vielzahl von möglichen Gegnerkombinationen, jedoch keine hohe Vielseitigkeit bei den individuellen Körperteilbereichen.*	
Must have	*Zwei* modulare *Körperbereiche*	Die KI besteht aus mehreren Körperteilbereichen die bei Spielstart zufällige Elemente erhalten. Dazu gehört ein Beinbereich, zwei Armbereiche sowie ein Rückenbereich. *Das heißt der Gegner setzt sich aus vier Körperteilbereiche zusammen.*	
Must have	*Ein Level im Prototyp*	Das Projekt umfasst ein Level, das als Bosskampf-Arena dient. *Dieses Level ist speziell für überschaubare Bosskämpfe konzipiert und ermöglicht so die erforderlichen Ziele des Level Designs.*	
Should have	*Generierung von zwölf Gegnerkombinationen*	Siehe Must have Ziel „Generierung von sechs verschiedenen Gegnern“.	
Should have	*Vier Körperteilvarianten pro Körperbereich*	Siehe Must have Ziel „Drei Körperteilvarianten pro Körperbereich“.	
Should have	*Drei* modulare *Körperbereiche*	Siehe Must have Ziel „Zwei Körperbereiche“.	

Could have	*Generierung von zwanzig Gegnerkombinationen*	Siehe Must have Ziel „Generierung von sechs verschiedenen Gegnern“. *Durch die Entscheidung, die Feinjustierung der bestehenden Mechaniken zu priorisieren, wurden keine zusätzlichen Körperbereichs- und Körperteilvarianten implementiert. Deshalb blieb die Anzahl der möglichen Gegnerkombinationen auf sechzehn begrenzt.*	
Could have	*Fünf Körperteilvarianten pro Körperbereich*	Siehe Must have Ziel „Drei Körperteilvarianten pro Körperbereich“.	
Could have	*Vier* modulare *Körperbereiche*	Siehe Must have Ziel „Zwei Körperbereiche“.	
Could have	*Zwei Level im Prototyp*	Siehe Must have Ziel „Ein Level im Prototyp“. *Durch die mangelnde Notwendigkeit weitere Level zu entwickeln, wurde dieses Ziel nicht weiterverfolgt.*	

Tabelle 3: Zielerreichung der quantitativen Ziele

5.2.4 Fazit – Quantitative Ziele

Von den elf quantitativen Zielen wurden sechs vollständig erreicht, fünf wurden nicht umgesetzt, und keines wurde nur teilweise erfüllt.

Die Generierung der Gegner mit zwei Körperteilvarianten pro Bereich ermöglicht insgesamt sechzehn verschiedene Kombinationen. Obwohl die Option für drei Körperteilvarianten pro Bereich erwogen wurde, wurden aus Ressourcen- und Prioritätsgründen keine zusätzlichen Varianten implementiert, was die Vielfalt der Gegner einschränkt.

Die KI besteht aus vier Körperteilbereichen, die zur Erstellung des Gegners verwendet werden. Obwohl auch die Integration von fünf Körperteilvarianten und vier Körperbereichen geplant war, wurde aufgrund der Priorisierung der Feinjustierung bestehender Mechaniken die Anzahl auf vier Körperteilbereiche und zwei Varianten pro Bereich begrenzt.

Das Projekt umfasst ein Level, das als Bosskampf-Arena dient. Die Entscheidung, keine weiteren Level zu erstellen, wurde getroffen, da ein einzelnes, gut gestaltetes Level den Anforderungen des Prototyps vollständig gerecht wird.

5.2.5 Ergebnisse der qualitativen Ziele

Priorität	Ziel	Einordnung	Ergebnis ■ Erreicht ■ Teilweise erreicht ■ Nicht erreicht
Must have	*Leicht verständlicher Workflow für die Implementierung weiterer Körperbereiche und Körperteilvarianten*	Der Workflow ist grundsätzlich verständlich, jedoch etwas unflexibel. Das Hinzufügen eines neuen Körperbereichs erfordert umfangreiche Anpassungen, während die Implementierung neuer Körperteilvarianten einfacher ist, da lediglich ein Subtree und ein Data Asset erstellt und in das KI-Array integriert werden müssen. *Während das Hinzufügen neuer Körperteilvarianten relativ einfach ist, erfordert die Integration zusätzlicher Körperbereiche umfassende Anpassungen, was den Prozess insgesamt weniger dynamisch und überschaubar gestaltet.*	
Must have	*Fehlerfreies Ausüben der Verhaltensweisen*	Die KI führt alle Verhaltensweisen wie gewünscht aus. Es gibt minimale Unstimmigkeiten, die jedoch kaum wahrnehmbar sind und das fehlerfreie Verhalten insgesamt nicht beeinträchtigen. *Die KI setzt alle vorgesehenen Verhaltensweisen korrekt um.*	
Must have	*Klar erkennbare Verhaltensweisen*	Die Verhaltensweisen haben individuelle Effekte wie UI-Elemente, Audio, VFX, Animationen und unterschiedliche visuelle Gestaltungen *Diese Elemente machen die verschiedenen Verhaltensweisen klar unterscheidbar und sichtbar.*	
Must have	*Fehlerfreie Bedienung des First Person Shooter Controllers*	Der Spieler kann fehlerfrei schießen, springen und laufen, und die Reduzierung der Lebenspunkte funktioniert ebenfalls einwandfrei. *Die Bedienung des Controllers ist insgesamt fehlerfrei, und alle Funktionen arbeiten wie vorgesehen.*	

Must have	*Leicht erkennbares Leveldesign*	Die Übersichtlichkeit des einfachen, quadratischen Levels, das ein typisches Arena-Gefühl vermittelt, ist durch die gute Beleuchtung gewährleistet. *Das Leveldesign ist klar erkennbar und ermöglicht eine einfache Orientierung und Navigation innerhalb der Arena.*	
Must have	*Leicht erkennbarer Gameplay loop durch ein Graphical User Interface mit Lebensanzeigen für Gegner und Spieler*	Zwei Lebensbalken zeigen die aktuellen Lebenspunkte der Spieler und Gegner an. Veränderungen werden durch kurzes Aufflackern der Balken deutlich gemacht, um Treffer visuell anzuzeigen. Dies sorgt für einen klar erkennbaren Gameplay-Loop. Da der Spieler schnell sterben kann, wird ihm beim Stillstehen und Unklarheit schnell bewusst, dass das Spiel endet, wenn eine Partei besiegt wird. *Diese Elemente gewährleisten einen leicht erkennbaren Gameplay Loop.*	
Must have	*Gut verständliche Projektstruktur und Blueprints Programmierung*	Die Projektstruktur ist klar unterteilt in AI, Blueprints, Assets und andere Bereiche, und alle Komponenten sind gut und korrekt benannt. Die Blueprints-Programmierung nutzt Vererbung, Event Dispatchers und ist gut kommentiert. Einige Refactoring-Maßnahmen könnten jedoch durchgeführt werden um Redundanz und doppelten Code zu minimieren. *Das Projekt bietet eine gut strukturierte und nachvollziehbare Organisation sowie eine solide Blueprints-Programmierung, wobei kleinere Anpassungen durchgeführt werden könnten.*	
Should have	*Visuell unterscheidbare Körperteilvarianten*	Die Körperteilvarianten verfügen über unterschiedliche Meshes und viele sind zusätzlich mit spezifischen Animationen und Effekten ausgestattet, die bei ihren Aktionen sichtbar werden. *Diese hohe Individualisierung ermöglicht eine einfache visuelle Unterscheidung der Varianten.*	

Should have	*Intuitive Bewegungsmechaniken durch den First Person Shooter Controller*	Die Bewegungsmechaniken sind grundsätzlich intuitiv, jedoch werden sie dem Spieler nicht aktiv vermittelt, etwa durch ein Pop-up oder ein Tutorial. Da es sich um einen Prototyp handelt, wurde dies jedoch nicht als Priorität betrachtet. *Die Bewegungsmechaniken sind intuitiv, jedoch fehlt eine aktive Einführung für den Spieler, was aufgrund des Prototyp-Status nicht priorisiert wurde.*	
Could have	*Ansprechende Körperteilvariant spezifische auditive Effekte*	Jedes Körperteilvariant besitzt spezifische Audioeffekte, die bei den jeweiligen Aktionen hörbar werden. *Die Feinjustierung der Lautstärke und die Bearbeitung der Sounds tragen zusätzlich zur Qualität der akustischen Darstellung bei.*	
Could have	*Ansprechende Körperteilvariant spezifische visuelle Effekte*	Jede Körperteilvariante verfügt über spezifische visuelle Effekte, die während der entsprechenden Aktionen sichtbar werden. *Die Feinjustierung der Effekte, wie Anpassungen in Geschwindigkeit und Partikelmenge, trägt zusätzlich zur Verbesserung der visuellen Qualität bei.*	
Could have	*Musikalische Untermalung des Gameplays*	Ein Musikstück beginnt beim Spielstart und läuft in einer Schleife bis zum Ende des Spiels. Es wurde ausgewählt, um Spannung zu erzeugen und die Atmosphäre eines Bosskampfes zu unterstreichen. Das Hinzufügen von Musik trägt zur Gesamtqualität des Prototyps bei.	

Tabelle 4: Zielerreichung der qualitativen Ziele

5.2.6 Fazit – Qualitative Ziele

Von den zwölf qualitativen Zielen wurden neun vollständig erreicht und drei teilweise umgesetzt.

Die Implementierung eines leicht verständlichen Workflows für Körperbereiche zeigt grundlegende Verständlichkeit, jedoch fehlt es an Flexibilität, insbesondere bei der Integration neuer Körperbereiche, was den Prozess umständlicher macht. Zusätzliche Refactoring-Maßnahmen bei der Blueprints-Programmierung hätten die Redundanz reduziert.

Die Verhaltensweisen der KI funktionieren größtenteils fehlerfrei, auch wenn minimale Unstimmigkeiten vorhanden sind, die die Funktionalität nicht signifikant beeinträchtigen.

Die klaren, visuell unterscheidbaren Körperteilvarianten sowie die intuitive Bedienung des First Person Shooter Controllers tragen zur Qualität des Prototyps bei, auch wenn die Einführung der Bewegungsmechaniken für den Spieler nicht aktiv unterstützt wird.

Diese Ziele wurden aus Ressourcengründen oder aufgrund von Priorisierungen in der Entwicklungsphase alle nur teilweise erreicht. Die Feinjustierung von Audio- und visuellen Effekten sowie die musikalische Untermalung tragen zur Gesamtqualität bei, zeigen aber auch, dass der Fokus auf den Kernmechaniken lag, während einige erweiterte Features nur teilweise umgesetzt wurden.

4.3 Evaluation

Rückblickend lässt sich sagen, dass das hauptsächliche Ziel, ein modulares System für die Erstellung von KI zu entwickeln, erreicht wurde. Das entwickelte Programm erfüllt die wesentlichen Anforderungen und zeigt in den verschiedenen Tests eine gute Leistung. Besonders hilfreich war die Möglichkeit, verschiedene Mechaniken pro Körperbereich zu implementieren, was es dem System erlaubt, sich flexibel an unterschiedliche Aufgaben anzupassen.

Die Implementierung ist intuitiv gestaltet und die Komponenten sind so strukturiert, dass auch Nutzer ohne tiefgehendes technisches Wissen gut damit zurechtkommen. Dadurch ist das System nicht nur für Experten interessant, sondern auch für eine breitere Zielgruppe.

Beim Zusammenspiel mehrerer modularer Verhaltensweisen kann das Endprodukt an seine Grenzen stoßen, wobei das Balancing das hauptsächliche Problem ist und weitere Optimierung notwendig wäre, um dies gezielter anzupassen.

Ein zusätzliches Problem ergibt sich aus manchen hartcodierten Eigenschaften des Systems. Diese fest einprogrammierten Funktionen schränken die Flexibilität ein, das System an neue oder unerwartete Anforderungen anzupassen. Wenn sich Anforderungen ändern oder neue Erkenntnisse gewonnen werden, können aufwendige Neuprogrammierungen erforderlich sein, was die Anpassung zeitintensiv und fehleranfällig machen kann. Besonders problematisch wird dies, wenn das System in neue Anwendungsbereiche vordringen soll, die ursprünglich nicht berücksichtigt wurden. Ein flexiblerer, datengetriebener Ansatz könnte hier Abhilfe schaffen und die Wartung und Weiterentwicklung des Systems erleichtern.

Trotz dieser Herausforderungen war das Projekt insgesamt erfolgreich. Die entwickelten Lösungen bieten eine solide Grundlage und zeigen viel Potenzial. Die Kombination aus einer stabilen Grundstruktur und der Möglichkeit das System kontinuierlich zu erweitern und zu verbessern, macht es gut gerüstet für den Einsatz bei anderen Projekten. Die festgestellten Schwächen bieten darüber hinaus wertvolle Hinweise für zukünftige Arbeiten, die dazu beitragen können, das System weiter zu optimieren.

4.4 Ausblick

Das entwickelte modulare KI-System bietet zahlreiche Möglichkeiten für zukünftige Entwicklungen und Kooperationen.

Die Veröffentlichung des Projekts auf Plattformen wie GitHub ist ein wichtiger nächster Schritt. Eine solche Veröffentlichung würde nicht nur die Sichtbarkeit des Projekts erhöhen, sondern auch weiteres Feedback aus der Entwicklergemeinschaft ermöglichen, das zur Verbesserung des Systems genutzt werden kann.

Das System bietet eine solide Basis für zahlreiche Erweiterungen. In zukünftigen Projekten könnten weitere Körperbereiche entwickelt werden, um die Vielfalt und Komplexität der KI zu erhöhen.

Die Modularität des Systems könnte theoretisch weiterhin verbessert werden, um eine allumfassende Implementierung einer KI zu gewährleisten, ohne auf Komponenten außerhalb des Systems zugreifen zu müssen. Solche Erweiterungen würden die Relevanz und Anwendbarkeit des Systems in modernen Videospielen weiter steigern. Hierbei kommt es auf die Limitierungen und Voraussetzungen des Projekts an in dem das System angewandt wird.

Quellenverzeichnis

1. Agriogianis, T., 2018. The Roles, Mechanics, and Evolution of Boss Battles in Video Games. *Undergraduate Honors College Theses 2016-*. [online] Available at: <https://digitalcommons.liu.edu/post_honors_theses/45>.

2. Anon. 2023. *Legs*. [online] Armored Core 6 Wiki. Available at: <https://armoredcore6.wiki.fextralife.com/Legs> [Accessed 19 August 2024].

3. Anon. 2024a. *Behavior Trees*. [online] Available at: <https://dev.epicgames.com/documentation/en-us/unreal-engine/behavior-trees-in-unreal-engine> [Accessed 15 July 2024].

4. Anon. 2024b. *Building the AI of F.E.A.R. with Goal Oriented Action Planning*. [online] Available at: <https://www.gamedeveloper.com/design/building-the-ai-of-f-e-a-r-with-goal-oriented-action-planning> [Accessed 19 August 2024].

5. Anon. 2024c. *Data Assets in Unreal Engine | Unreal Engine 5.4 Documentation | Epic Developer Community*. [online] Epic Games Developer. Available at: <https://dev.epicgames.com/documentation/en-us/unreal-engine/data-assets-in-unreal-engine> [Accessed 22 August 2024].

6. Anon. 2024d. *GOOD SKY in Blueprints - UE Marketplace*. [online] Unreal Engine. Available at: <https://www.unrealengine.com/marketplace/en-US/product/good-sky> [Accessed 15 August 2024].

7. Anon. 2024e. *Iteratives Prototyping: Design & Strategien*. [online] StudySmarter. Available at: <https://www.studysmarter.de/studium/bwl/unternehmensgruendung/iteratives-prototyping/> [Accessed 14 July 2024].

8. Anon. 2024g. *Mech Constructor: Spiders and Tanks | 3D Robots | Unity Asset Store*. [online] Available at: <https://assetstore.unity.com/packages/3d/characters/robots/mech-constructor-spiders-and-tanks-54074> [Accessed 15 August 2024].

9. Anon. 2024h. *Modular AI Prototype - Bachelor Project by Kevin Catlett Games*. [online] itch.io. Available at: <https://kevincatlettgames.itch.io/modular-ai-prototype-major-project> [Accessed 20 August 2024].

10. Anon. 2024i. *Modular SciFi Season 1 Starter Bundle in Environments - UE Marketplace*. [online] Unreal Engine. Available at: <https://www.unrealengine.com/marketplace/en-US/product/modular-scifi-season-1-starter-bundle> [Accessed 15 August 2024].

11. Anon. 2024j. *Realistic Starter VFX Pack Vol 2 in Visual Effects - UE Marketplace*. [online] Unreal Engine. Available at: <https://www.unrealengine.com/marketplace/en-US/product/realistic-starter-vfx-pack-vol> [Accessed 15 August 2024].

12. Anon. 2024k. *Sound and Music for Games*. [online] We Love Indies. Available at: <https://www.weloveindies.com/en> [Accessed 15 August 2024].

13. Anon. n.d. *document.pdf*. Available at: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=4fc701793a7efe4b59c58b071e302e0c1eb87616#page=86> [Accessed 15 July 2024a].

14. Anon. n.d. *GameAIPro_Chapter09_An_Introduction_to_Utility_Theory.pdf*. Available at: <https://www.gameaipro.com/GameAIPro/GameAIPro_Chapter09_An_Introduction_to_Utility_Theory.pdf> [Accessed 19 August 2024b].

15. Anon. n.d. *historyOfZork.pdf*. Available at: <https://samizdat.co/shelf/documents/2004/05.27-historyOfZork/historyOfZork.pdf> [Accessed 15 July 2024c].

16. *Armored Core 6 Interview with Yamamura & Ogura, Featuring Armored Core Legacy*. 2023. Directed by FightinCowboy. Available at: <https://www.youtube.com/watch?v=0ng-qIRThCI> [Accessed 19 August 2024].

17. Booth, M., n.d. The AI Systems of Left 4 Dead.

18. Bourg, D.M. and Seemann, G., 2004. *AI for game developers: creating intelligent behavior in games*. 1. ed ed. Creating intelligent behavior in games. Beijing Köln: O'Reilly.

19. Buckland, M., 2005. *Programming game AI by example*. Plano, Texas: Wordware Pub.

20. Champandard, A.J., 2004. *AI game development: synthetic creatures with learning and reactive behaviors*. Indianapolis, Ind: New Riders.

21. Champandard, A.J. and Dunstan, P., n.d. The Behavior Tree Starter Kit.

22. Colledanchise, M. and Ögren, P., 2018. *Behavior Trees in Robotics and AI: An Introduction*. [online] https://doi.org/10.1201/9780429489105.

23. DaGraça, M., n.d. Practical Game AI Programming.

24. DeNero, J. and Klein, D., 2010. Teaching Introductory Artificial Intelligence with Pac-Man. *Proceedings of the AAAI Conference on Artificial Intelligence*, 24(3), pp.1885–1889. https://doi.org/10.1609/aaai.v24i3.18829.

25. *Development of Left 4 Dead by Mike Booth GDC Lecture*. 2023. Directed by Mr Morton. Available at: <https://www.youtube.com/watch?v=PJNQl3K58CQ> [Accessed 19 August 2024].

26. Dorr, L., 2022. Types of Artificial Intelligence, Explained. [online] 56. Available at: <https://www.dentalproductsreport.com/view/types-of-artificial-intelligence-explained> [Accessed 11 July 2024].

27. *Goal-Oriented Action Planning Part 1*. 2022. Directed by Holistic3D. Available at: <https://www.youtube.com/watch?v=tdBWk2OVCWc> [Accessed 19 August 2024].

28. Görz, G. and Nebel, B., 2014. *Künstliche Intelligenz*. unveränd. Repr. e. älteren Ausg.; Orig.-Ausg ed. Fischer Taschenbuch. Frankfurt am Main: Fischer Taschenbuch.

29. Hintze, A., 2016. *Understanding the four types of AI, from reactive robots to self-aware beings*. [online] The Conversation. Available at: <http://theconversation.com/understanding-the-four-types-of-ai-from-reactive-robots-to-self-aware-beings-67616> [Accessed 11 July 2024].

30. Jacopin, É., n.d. Optimizing Practical Planning for Game AI.

31. Jagdale, D., 2021. Finite State Machine in Game Development. pp.384–390. https://doi.org/10.48175/IJARSCT-2062.

32. Kaplan, J., 2017. *Künstliche Intelligenz: Eine Einführung*. [online] MITP-Verlags GmbH & Co. KG. Available at: <https://books.google.de/books?hl=en&lr=&id=mUIzDwAAQBAJ&oi=fnd&pg=PT2&dq=k%C3%BCnstliche+intelligenz+&ots=Ceq-A65a9O&sig=KsO6Hq6wnRFvWsYCot782i_jkhw&redir_esc=y#v=onepage&q=k%C3%BCnstliche%20intelligenz&f=false>.

33. Kartsios, G., 2022. *Das ABC der Videospiele*. 3. Auflage ed. Oldenburg Hamburg: Lappan Verlag.

34. Magerko, B., Laird, J.E., Assanie, M., Kerfoot, A. and Stokes, D., n.d. AI Characters and Directors for Interactive Computer Games.

35. Millington, I., 2020. *AI for games*. Third edition, first issued in paperback ed. Boca Raton, Fla. London New York: CRC Press, Taylor & Francis Group.

36. Mitchell, M., 2020. *Artificial intelligence: a guide for thinking humans*. Published in paperback ed. A Pelican book. London: Pelican, an imprint of Penguin Books.

37. Newton, P.L. and Feng, J., 2016. *Unreal Engine 4 AI Programming Essentials*. Packt Publishing Ltd.

38. Perron, B., Boudreau, K., Wolf, M.J.P. and Arsenault, D., 2022. *Fifty Key Video Games*. Taylor & Francis.

39. Rabin, S. ed., 2020. *Game AI Pro 360: guide to architecture*. Boca Raton London New York: CRC Press, Taylor & Francis Group.

40. Romero, M. and Sewell, B., 2022. *Blueprints Visual Scripting for Unreal Engine 5: Unleash the true power of Blueprints to create impressive games and applications in UE5*. Packt Publishing Ltd.

41. Safadi, F., Fonteneau, R. and Ernst, D., 2015. Artificial Intelligence in Video Games: Towards a Unified Framework. *International Journal of Computer Games Technology*, 2015(1), p.271296. https://doi.org/10.1155/2015/271296.

42. Sewell, B., 2015. *Blueprints Visual Scripting for Unreal Engine*. Packt Publishing Ltd.

43. Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I. and Postma, E., 2006. Adaptive game AI with dynamic scripting. *Machine Learning*, 63(3), pp.217–248. https://doi.org/10.1007/s10994-006-6205-6.

44. *The AI of DOOM (1993) | AI and Games #66*. 2022. Directed by AI and Games. Available at: <https://www.youtube.com/watch?v=owAxZPE9a0E> [Accessed 15 July 2024].

45. *The AI of DOOM (2016) | AI and Games #30*. 2018. Directed by AI and Games. Available at: <https://www.youtube.com/watch?v=RcOdtwioEfI> [Accessed 15 July 2024].

46. *The Director AI of Left 4 Dead | AI and Games #07*. 2014. Directed by AI and Games. Available at: <https://www.youtube.com/watch?v=WbHMxo11HcU> [Accessed 11 July 2024].

47. Uriarte, A. and Ontañón, S., 2015. A Benchmark for StarCraft Intelligent Agents. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 11(2), pp.22–28. https://doi.org/10.1609/aiide.v11i2.12810.

48. *Utility AI In Unity - Part 1 - Introduction*. 2021. Directed by TooLoo. Available at: <https://www.youtube.com/watch?v=ejKrvhusU1I> [Accessed 19 August 2024].

49. Weber, B., n.d. Reactive Planning for Micromanagement in RTS Games.

50. *Which AI Behavior Framework Should You Use? | AI Series 46*. 2023. Directed by LlamAcademy. Available at: <https://www.youtube.com/watch?v=CZvfuNfdc1M> [Accessed 19 August 2024].

51. *Winding Road Ahead: Designing Utility AI with Curvature*. 2021. Directed by GDC. Available at: <https://www.youtube.com/watch?v=TCf1GdRrerw> [Accessed 19 August 2024].

52. Zia, H., 2023. *How To Save loadout In Armored Core 6*. [online] SegmentNext. Available at: <https://segmentnext.com/armored-core-6-loadout/> [Accessed 19 August 2024].