



**HOCHSCHULE
MITTWEIDA**
University of Applied Sciences

Fakultät Angewandte Computer- und Biowissenschaften

Professur Medieninformatik

Bachelorarbeit

Untersuchung des Einflusses von visuellen Distraktoren auf die
Hand-Augen-Koordination bei virtuellem Präzisionstraining

Lars Goldammer

Mittweida, den 18. Mai 2020

Erstprüfer: Prof. Dr. rer. nat. Marc Ritter

Zweitprüfer: Manuel Heinzig M.Sc.

Goldammer, Lars

Untersuchung des Einflusses von visuellen Distraktoren auf die Hand-Augen-Koordination bei virtuellem Präzisionstraining

Bachelorarbeit, Fakultät Angewandte Computer- und Biowissenschaften

Hochschule Mittweida — University of Applied Sciences, Mai 2020

Name: Goldammer, Lars

Studiengang: Medieninformatik und Interaktives Entertainment

Seminargruppe: MI15w2-B

English Title: Investigation of the influence of visual distractors on hand-to-eye-coordination in virtual precision training

Inhaltsverzeichnis

Abbildungsverzeichnis	V
1 Einführung und Motivation	1
1.1 Zielstellung und Aufgabe	3
2 Theoretische Grundlagen	5
2.1 Distraktoren und deren Nutzen	5
2.1.1 Visuelle Distraktoren in der Spieleumgebung	8
2.2 Farbwahrnehmung des menschlichen Auges	9
2.3 Design	12
2.3.1 Das User Interface - UI	12
2.3.2 3D-Objekte	13
2.3.3 2D-Elemente	14
2.4 Unity	14
2.4.1 Unity Oberfläche	15
2.4.2 Szene	17
2.4.3 Graphic User Interface (GUI)	17
2.4.4 Prefab	17
2.4.5 FPS-Controller	17
2.4.6 Collider und Rigidbody	18
2.4.7 Skybox	18
3 Konzeptionierung des Präzisionstests	21
3.1 Grundlagen des Leveldesigns in 3D	21
3.1.1 Map-Gestaltung	23
3.2 Aufbau des Präzisionstests	25

3.2.1	Szenenbau	25
3.2.2	Menü	26
3.2.3	Test-Instruktionen	27
3.3	Levelgestaltung	27
3.3.1	Roam - Roaming Map	28
3.3.2	Normal Map - Test-Map1	30
3.3.3	Dis-Map1 - Distraktor-Map1	31
3.3.4	Dis-Map2 - Distraktor-Map2	32
3.4	Klassenbetrachtung	34
3.4.1	gun.cs	34
3.4.2	Timer.cs und startColider.cs	37
3.4.3	Target.cs	39
4	Evaluation	41
4.1	Durchführung und Auswertung des Tests	41
4.1.1	Testbeschreibung und Ablauf	41
4.2	Einzeltestdurchführung und Ergebnisanalyse	43
4.2.1	Teilnehmer 1 - professioneller CoD-Spieler	43
4.2.2	Teilnehmer 2 - erfahrener Taktik- <i>Shooter</i> -R6 Spieler	46
4.2.3	Teilnehmer 3 - erfahrener CS:S Spieler	48
4.2.4	Teilnehmer 4 – <i>Large-Scale Shooter</i> Spieler	50
4.2.5	Teilnehmer 5 - Entwickler und <i>Casual</i> Spieler	52
4.3	Gesamtauswertung des Präzisionstests	55
4.3.1	Virtuelles Präzisionstraining - Datenanalyse	55
4.3.2	Auswertung und Diskussion des Fragebogens	61
4.3.3	Bewertung der Grundstruktur - Leveldesign	63
5	Schlussfolgerungen und Ausblick	65
6	Zusammenfassung	67
	Literaturverzeichnis	I
A	Zusätzliche Abbildungen	A1

B Importierte Assets und AddOns	A7
C Script - Dokumentation	A9
C.1 Klassen	A9

Abbildungsverzeichnis

2.1	Text- und Farbdistraktoren zum Hauptmotiv - Die Linke	7
2.2	Modischer Distraktor: Ingame Avatar Outfit	7
2.3	Sichtbares Licht. Modifiziert nach [But10].	9
2.4	Farbwahrnehmung der Sinneszellen der Netzhaut des Auges [SHHH19].	10
2.5	Oberfläche Unity Engine.	16
2.6	<i>Skybox</i> und Szene Ohne <i>Collider</i>	19
3.1	Aussehen der <i>Roam</i> Map in <i>First Person</i>	29
3.2	Aussehen der ersten Testmap in <i>First Person</i>	30
3.3	Aussehen der ersten Distraktor-Map in <i>First Person</i>	31
3.4	Aussehen der zweiten Distraktor-Map in <i>First Person</i>	33
4.1	Effizienter Distraktor in Dis-Map2	45
4.2	Zieldarstellung Dis-Map2 (t2).	54
4.3	Test-Durchlauf 1 – Summe der Durchlaufzeiten aller Maps	58
4.4	Durchlaufdaten Teilnehmer	59
4.5	Vergleich der Durchlaufzeitenanteile je Map	60
4.6	Laufwege der Testteilnehmer (T) im Präzisionstest	61
A.1	Test zur Prüfung der Wirkung innerer und äußerer Distraktoren. . . .	A2
A.2	Rot-Grün-Farbsehschwäche: Distraktorenvergleich 1	A3
A.3	Rot-Grün-Farbsehschwäche: Distraktorenvergleich 2	A4
A.4	<i>Normal</i> Map mit Darstellung der Zielpositionen.	A5

1. Einführung und Motivation

Die Grafik von Videospielen entwickelt sich in der Spieleindustrie rasant. Jährlich gelangen neue Grafikkarten, Grafikprozessoren (*Graphics Processing Unit, GPU*) oder auch integrierte *OnBoard* Grafikprozessoren mit höherer Leistung auf den Markt, die in digitalen Endgeräten, wie Computern, *Tablets*, Spielekonsolen, *Smartphones*, *Smartwatches* oder VR-Brillen für *Virtual Reality* usw. eingebaut werden. VR-Brillen erhalten bereits eigene *GPUs* und zentrale Recheneinheiten (*Central Processing Unit, CPU*) und die visuellen Möglichkeiten von Videospielen werden weiter optimiert. Die weitere Zunahme der Rechenleistung der Grafikplattformen erlaubt die Herstellung von Konsolen und Displays mit immer besseren visuellen Möglichkeiten für den Endnutzer.

Auch *Mobile Games* bekommen eine qualitativ immer bessere Grafik, obwohl diese Spiele oft nur für den Zeitvertreib unterwegs auf dem *Smartphone* entwickelt werden. Bei der Betrachtung von Computerspielen, zumindest in den größeren Spielegenres, wie z.B. Action, Adventure, Strategie oder Simulation, erscheint es, als ob eine immer bessere Grafik für den Markterfolg eines Spiels bestimmend ist.

Doch ist bessere Grafik auch immer gleichzusetzen mit besserem *Gameplay*? Oder sind die aktuellen Spiele schon so sehr übersättigt mit Spezialeffekten und grafischen Elementen, sodass vielleicht die Spieler von dieser visuellen Vielfalt und den hochauflösenden Texturen abgelenkt und teilweise überfordert werden? Sind die Spieler möglicherweise aufgrund zu vieler Falsch-Alternativen bzw. *Multiple Choice*-Aufgaben in ihrem Spielerlebnis sogar negativ beeinflusst oder beeinträchtigt? In nahezu allen modernen Spielen, insbesondere im FPS (*First Person Shooter*) Genre, werden solche Ablenkungen (Distraktionen) immer mehr zu einem Trend, sei es durch Graffitis, Outfits oder Waffenskins.

Dazu zählt unter anderem auch, dass Modegegenstände per *Ingame*-Transaktion in Spielen angeboten und erworben werden können. Dabei stellt sich die Frage, ob durch ein neues Outfit für einen *Ingame*-Avatar auch gewisse taktische Vor- oder

Nachteile entstehen. Untersuchungen zu den Auswirkungen solcher Accessoires auf das Gameplay in kompetitiven Spielen könnten interessante Antworten zur deren potentieller Ablenkungsfähigkeit liefern.

Visuelle Distraktoren haben die Aufgabe der bewussten Ablenkung, insbesondere in der Werbebranche - dort werden sie als „Störer“ bezeichnet. Diese grafischen Elemente werden bewusst eingesetzt, um die Aufmerksamkeit des Betrachters gezielt so zu lenken, dass z.B. das beworbene Produkt ein positives Image erhält.

In kompetitiven Spielen sind störende Ablenkungen von Seiten des Konsumenten nicht immer erwünscht, da sich der Spieler in der Regel auf das Spiel konzentrieren möchte. Wenn visuelle Distraktoren erst durch ungünstige oder falsche Platzierungen von grafischen Elementen oder das Implementieren von auffälligen Outfits im Spiel entstehen kann es gegebenenfalls sogar zu unfairen Spielbedingungen kommen. Der Spieler wird also unabsichtlich von der eigentlichen Spieldesignidee des Entwicklers mit Ablenkungen konfrontiert.

Umgekehrt gibt es jedoch auch Spiele, die genau auf solchen Faktoren aufbauen, wie z.B. der Taktik Shooter *“Tom Clancys Rainbow Six:Siege“*. Dieser besitzt eine große Anzahl zusätzlich zerstörbarer Elemente in seinen Levels, wodurch z.B. Löcher in Wänden möglich werden oder auch bestimmte Objekte teilweise nach Beschuss nicht mehr existieren. Das führt oft dazu, dass Spielteams prophylaktisch derartige Distraktoren durch *Prefire* bekämpfen, um potentielle Versteckmöglichkeiten eines Gegners zu beseitigen. Dieses aber mit dem Nachteil, die eigene Position preiszugeben.

Nicht zuletzt werden bestimmte Personengruppen, z.B. mit Farbensehschwäche, teilweise unbewusst durch Distraktoren benachteiligt, da sie die visuelle Umgebung anders wahrnehmen. Dies wird in sehr vielen Spielen, noch nicht ausreichend in Betracht gezogen.

Mit dieser Arbeit sollen die Auswirkungen von zusätzlich gesetzten optischen Reizen, in Form von visuellen Distraktoren, in einer Test-Spieleumgebung untersucht werden. Dabei werden die generelle Wirkung und der Einfluss auf Präzision und Koordination der Spieler betrachtet. Im Rahmen einer von FPS-Mechaniken gesteuerten Test-Anwendung erfolgt dazu eine unterschiedliche Belastung der Spieler in drei Stufen.

1.1. Zielstellung und Aufgabe

Erstes Ziel der Arbeit ist es, eine Anwendung mit der Unity Engine zu erstellen, welche grundlegende Mechaniken einer *First Person* Zielübung beinhaltet.

Hierzu soll eine spielbare Karte mit Präzisionszielen entwickelt werden, welche mit gezieltem Klicken zerstört werden können. Dabei sollen neben einer „realistisch“ aussehenden Map, zwei weitere Varianten erstellt werden, die in unterschiedlichen Abstufungen zusätzliche visuelle Elemente und Distraktoren beinhalten. Um die Wirksamkeit der Distraktoren prüfbar zu gestalten, sollen möglichst effiziente und starke Distraktoren im Vergleich zu einer Kontrolle ohne Distraktoren eingesetzt werden. Dadurch wird die Aussagefähigkeit der zu ermittelnden Daten unterstützt. Die Distraktoren sollen für potentielle Probanden des Testspiels den Durchlauf der Levels sowie das Auffinden und Treffen von Zielen erschweren.

Diese Karten werden von Grund auf gemodelt und die Funktionalität der Anwendung wird über die Skriptsprache C# implementiert.

Das zweite Ziel besteht darin, ein Spieleexperiment zu entwickeln, welches basierend auf der gestalteten FPS-Anwendung eine Nachweismöglichkeit für die Wirksamkeit visueller Distraktoren auf Spielerteilnehmer hat. Dazu sollen Probanden mit Vorkenntnissen im FPS-*Gaming* für das Experiment rekrutiert werden. Die Teilnehmer werden die verschieden gestalteten Karten nach der Regie des Entwicklers durchlaufen und danach einen passend zur Aufgabenstellung entwickelten Fragebogen beantworten. Ebenfalls wird die Zeit gestoppt, welche diese Spieler für einen Durchlauf der verschiedenen Levels benötigen. Es wird weiterhin betrachtet, inwieweit sich durch wiederholtes Durchlaufen der Karten die Zeiten der Probanden verändern. Der zu entwickelnde Test soll zeigen, ob die Spieler abgelenkt werden und ob bei den Probanden durch wiederholtes Training eine Verbesserung der Spielleistung möglich ist, obwohl visuelle Distraktoren unterschiedlichster Art und Vielfältigkeit den Spieler behindern bzw. ablenken. Damit soll ermittelt werden, ob Testspieler in einem bestimmten und wenn ja, welchem Zeitraum in der Lage sind, bekannte Distraktoren auszublenden.

Mit dem gewählten experimentellen Ansatz kann eventuell am Rande auch ein kleiner bestätigender Baustein für die Hypothese erbracht werden, dass das Spielen von Videospielen eher positive, leistungssteigernde Effekte bezüglich des Wahrnehmungsvermögens und der Motorikfähigkeiten von Spielern haben kann, wie in einer

umfassenden und sehr gut abgesicherten Studie von Green und Bavelier im Jahr 2003 [GB03] gezeigt. Dort konnte nachgewiesen werden, dass die visuellen Fähigkeiten bei Gewohnheitsspielern gegenüber Nicht-Spielern verbessert sind, aber auch Nicht-Videospiel-Spieler durch das Spieltraining diese Fähigkeiten verbessern.

Im Rahmen der vorliegenden Arbeit werden in einem minimalistischen Ansatz die Testpersonen hinsichtlich der Trainierbarkeit ihrer visuellen und koordinativen Fähigkeiten geprüft. Die Grundidee fokussiert jedoch ganz spezifisch nur auf den Einfluss ablenkender Elemente im Spieledesign.

2. Theoretische Grundlagen

Dieses Kapitel gibt eine theoretische Einführung in die Funktionsweise von Distraktoren unter Berücksichtigung der Farbwahrnehmung des menschlichen Auges. Weiterhin werden der *Workflow* sowie die verwendeten *Tools* zur Umsetzung des Projektes kurz vorgestellt.

2.1. Distraktoren und deren Nutzen

In unserem alltäglichen Leben sind wir als Menschen zahlreichen Umweltreizen ausgesetzt, die wir mit unseren Sinnesorganen sensorisch erfassen. Dies geschieht beispielsweise auf visueller Basis, über akustische Signale oder auch über Gerüche und haptische Reize. Dabei wird die visuelle Wahrnehmung durch das Auge bei vielen Menschen als einer der dominantesten sensorische Reize bezeichnet [MRC⁺18]. Der menschliche Organismus nimmt sämtliche Reize auf und unser Gehirn filtert und verarbeitet sie permanent. Dabei werden die Reize nach der Stärke des Eindrucks sortiert [Bos12], das heißt, ein stärkerer Reiz wird möglichst priorisiert und zieht somit oft mehr Aufmerksamkeit auf sich. Diese stärkeren Reize, welche andere Wahrnehmungen „überschreiben“ nennt man „*distractors*“ oder auch zu Deutsch „Ablenker“ oder Distraktoren. Distraktoren sind per Definition vorgegebene Falsch-Alternativen in einer Mehrfachwahl- bzw. *Multiple Choice*-Aufgabe [Wen00]. Der Begriff Distraktor entstammt primär der psychologischen Fachbegriffswelt.

Diese Wahrnehmungsart des Menschen kann manipuliert, bzw. geleitet werden, durch konkretes Einsetzen verschieden starker Reize in mehrerlei Hinsicht [KL10]. Je mehr Distraktoren z.B. vorhanden sind, desto länger dauert die *Target*-Suche. Mit ausreichend Übung nimmt die Suchzeit ab und die Genauigkeit in der Suche zu. Von Bedeutung für die Suchgeschwindigkeit ist dabei auch, ob eine Suchumgebung im wiederholten Test grundsätzlich gleich, ähnlich oder immer wieder verändert ist. Entsprechend stellen sich bei Probanden im Wiederholungstest Suchautomatismen

und kontrolliertes Suchverhalten ein. Ergebnisse grundlegender Experimente dazu, basierend auf der Identifizierung einfacher Zahlen-Buchstaben-Folgen in Suchkarten, wurden bereits 1977 vorgestellt [SS77]. Sie zeigten unter anderem, dass ähnliche bzw. stark unterschiedliche Distraktoren erheblichen Einfluss auf die *Target*-Suche hatten.

Distraktoren sind in vielerlei Hinsicht nützlich und werden daher auch heute intensiv erforscht und eingesetzt. Dabei spielt neben ihrem Einsatz in der realen Welt insbesondere ihre allgegenwärtige Nutzung in multimedialen Umgebungen eine zunehmende Rolle. In letzterer sind vor allem akustische und visuelle Distraktoren wichtig.

So werden z.B. in der Werbeindustrie Distraktoren vor allem bei visueller und akustischer Werbung als sogenannte „Störer“ verwendet (siehe Kapitel 1). Hierbei werden zumeist mit positiven Emotionen besetzte Distraktoren eingesetzt, wie niedliche Tiere, schöne Menschen, großartige Landschaften, spannende Musik usw. [MSC⁺07]. In allen Fällen könnte das eigentlich beworbene Produkt aber z.B. aus dem Bereich Auto, Technik, Haushaltsmittel usw. stammen. Die Audio-Ablenkung erfolgt dabei z.B. durch ein *Jingle*. Ein anschauliches Beispiel aus der visuellen realen Welt hingegen ist die Plakatwerbung. Hier werden oft grelle oder kontrastreiche Farben oder geometrische 2D-Elemente über das Hauptmotiv einer Werbeanzeige gelegt, welches dadurch so deplatziert wirkt, dass der Betrachter es anschauen muss (Ablenkung), aber auch hinblicken will (Neugier, natürlicher Entdeckerdrang des Menschen) [Ans06]. Es zieht damit die Aufmerksamkeit des Betrachters länger auf sich. Insbesondere bei Wahlplakaten ist dies ein typisches Gestaltungselement (Abbildung 2.1). In Spielen können plötzlich auftauchende *Ingame* Avatare mit alternativen modischen Outfits unmittelbar zum Distraktor werden und damit sogar Einfluss auf das taktische Verhalten der Spielteilnehmer nehmen (Abbildung 2.2).

Zusammengefasst macht dies deutlich, dass Distraktoren neben einfacher Ablenkung oft dafür verwendet werden, die Wahrnehmung und Aufmerksamkeit durch die Ablenkung indirekt auf ein Hauptmotiv zu lenken und dessen Eindruck dadurch sogar noch zu verstärken.



Abbildung 2.1.: Text- und Farbdistraktoren zum Hauptmotiv - Die Linke.¹



Abbildung 2.2.: Modischer Distraktor: Ingame Avatar Outfit (links) in *Rainbow Six: Siege*.²

¹.<https://www.dielinke-sachsen.de/wahlen/landtagswahl-2019/> 14 Mai 17:48 Uhr

²<https://i.redd.it/n9gghyv6gbc31.jpg> 15 Mai 23:46 Uhr

2.1.1. Visuelle Distraktoren in der Spieleumgebung

Distraktoren werden bewusst in der Spieleindustrie genutzt, um Konsumenten durch das Spiel zu leiten und deren Wahrnehmung und Koordinationsfähigkeit herauszufordern. Sie dienen der Erhöhung der Spannung und binden Spieler generell länger an ein Spiel. Dabei findet man die Distraktoren insbesondere bei dreidimensionalen Umsetzungen von Levels in Einzelspielerkampagnen. Hier dienen sie indirekt der Navigation und somit der Leitung des Spielers durch die Karte, um entsprechenden Fortschritt und eine Orientierung im Verlauf des Spieles zu erzielen. Das kann beispielsweise eine beleuchtete Seitengasse sein, eine angelehnte Tür in greller Farbe oder auch ganz unkompliziert ein Schild mit einem Pfeil. Ebenso werden auch wichtige *Items* in der virtuellen Umgebung hervorgehoben, um für den Betrachter noch offensichtlicher zu sein [GDG98]. In einer sonst relativ klar und durchdacht gestalteten Umgebung zieht so eine Markierung über Helligkeitsunterschiede, veränderte Farben oder Muster etc. nachweislich die Aufmerksamkeit des Spielers auf sich. Diese weist darauf hin, dass hier ein wichtiges *Item* zu finden ist oder der Spieler hier eine Interaktionsmöglichkeit haben könnte. Ein weiterer zu berücksichtigender Aspekt in Bezug auf die Erkennbarkeit und Wirkungsweise von Distraktoren ist, dass die Spieler entsprechend auf diese Distraktoren hintrainiert werden können, was letztlich zu einer anderen Wahrnehmung dieser im Vergleich zu nicht bewusst genutzten Distraktoren führen kann.

Dies zeigt, dass, auch wenn Distraktoren grundsätzlich als eine Form der Ablenkung funktionieren, deren bewusste Anwendung oft sehr hilfreich für den Spieler ist. An dieser Stelle gibt es daher auch eine zumindest gedankliche Schnittmenge mit den sogenannten Attraktoren in Spielen, die das Spielerverhalten ebenso manipulieren. Sie dienen dem Spieler jedoch sehr direkt als Zielorientierung bzw. Unterstützung in der Wegfindung und führen ihn unausweichlich zum Levelende, ohne dass der Spieler selbst diese gerichtete Einschränkung im Spiel bemerkt [Joi18]. Natürlich werden Distraktoren in der Spieleumgebung auch genutzt, um den Spieler zunächst bewusst fehlzuleiten oder, wie auch bei Distraktoren in der Werbung (siehe Abschnitt 2.1), von einem anderen Event abzulenken. So findet man bei Spielen oder anderen visuell aufbereiteten Medien, wie unter anderem beim Horror Genre, sehr häufig Distraktoren zur Ablenkung vom tatsächlichen Geschehen. Der Entwickler versucht absichtlich die Aufmerksamkeit des Betrachters auf etwas, an sich Irrelevantes zu

leiten, um vom Hauptmotiv abzulenken, damit ein gewünschter Überraschungseffekt zustande kommt, z.B. der *Jump Scare*.

In der vorliegenden Arbeit liegt der Fokus auf der grundlegenden Prüfung, ob visuelle Distraktoren direkt im Spiel den Ablauf stören und damit aktiv Einfluss auf das Spielergebnis haben. Es gibt einige Studien, die basierend auf der Beobachtung von Videospiel-Spielern in einer Nichtspiel-Testumgebung versuchen, auf verschiedene Weise, die Ursachen für deren verbesserte Aufmerksamkeit durch antrainiertes und dadurch automatisiertes Ausblenden und Filtern irrelevanter Informationen bzw. Distraktoren aufzuklären. Immer wieder werden dafür fundierte Probandentests entwickelt, wie etwa die Arbeit von Bavelier und Mitarbeitern [BAMF12] zum Nachweis aktiver Gehirnareale bei Videospiel-Spielern gegenüber Nichtvideospiel-Spielern zeigt. Jene verwendeten einen Test, in dem unter Einwirkung bewegter äußerer bzw. zentraler Distraktoren bestimmte Formen von anderen Formen zeitabhängig herausgefiltert werden müssen (Abbildung A.1).

2.2. Farbwahrnehmung des menschlichen Auges

Wie zuvor gezeigt, werden visuelle Distraktoren, die zielgerichtet zum Einsatz kommen, oft mit klaren und sehr präsenten Farben gestaltet. Ein Grund dafür ist, dass der Mensch über seine Augen bestimmte Farben besser wahrnehmen kann als andere. Warum dies so ist, wird im Folgenden kurz physikalisch und physiologisch dargestellt.

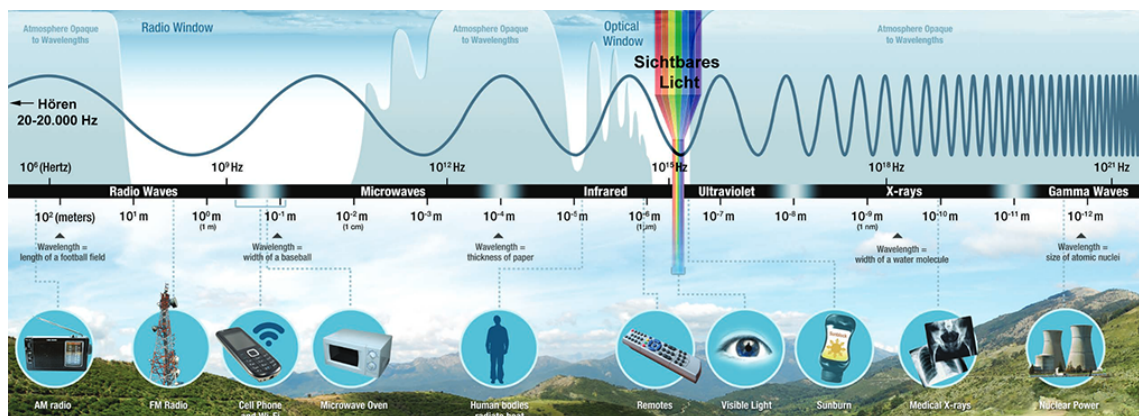


Abbildung 2.3.: Sichtbares Licht. Modifiziert nach [But10].

Physikalisch betrachtet, besteht das für den Menschen sichtbare Licht aus elektromagnetischen Wellen (Abbildung 2.3). Nur der sehr kleine Wellenlängenbereich von etwa 380nm (blau/violett) - 700nm (rot) zählt dazu. Längerwelliges Infrarotlicht kann der Mensch im Gegensatz zu anderen Lebewesen (z.B. Königsboa, *Boa constrictor*) nur als Wärmestrahlung über die Haut empfinden. Kürzer welliges UV-Licht – sichtbar z.B. für Honigbienen (*Apis mellifera*) – kann die Haut des Menschen, insbesondere aber dessen Augen, hingegen schon nach kurzer Einwirkdauer schädigen [SHHH19].

In der Summe addieren sich die für den Menschen sichtbaren Bereiche des elektromagnetischen Spektrums zu weißem Licht. Die Hauptbestandteile oder auch Hauptfarben des weißen Lichtes sind die Farben Rot, Grün und Blau sowie deren Nuancen (Abbildung 2.4). Diese Farbzusammensetzung spiegelt sich anteilig auch im computerorientierten additiven Farbraum des RGB-Systems und anderen Farbsystemen wider. Ebenfalls sind es diese drei Farben, die der Mensch stärker als andere Untertöne und Mischfarben wahrnehmen kann. Ursächlich ist dafür die Evolution des Menschen, seine damit verbundene vorrangig tageslichtabhängige Aktivität und die daraus resultierende Physiologie des menschlichen Auges verantwortlich. Dringt Licht in das Auge, fällt es auf die Netzhaut (Retina), welche unter anderem aus

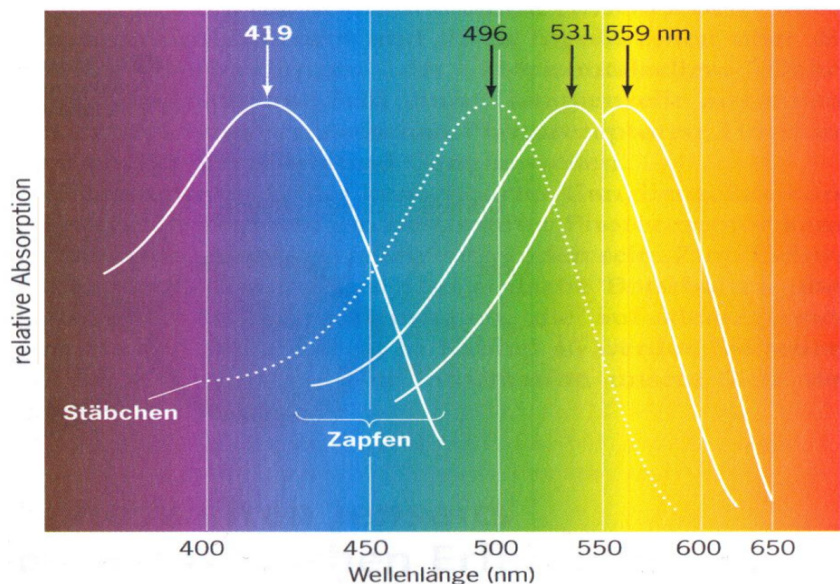


Abbildung 2.4.: Farbwahrnehmung der Sinneszellen der Netzhaut des Auges [SHHH19].

vier verschiedenen lichtempfindlichen fotorezeptorhaltigen Sinneszellen besteht. Dazu zählen lichtempfindliche Stäbchenzellen (ca. 120 Mio), die schon bei sehr geringen Lichtintensitäten Lichtreize aufnehmen und an das Gehirn weiterleiten, und die für das Dämmerungssehen und die Unterscheidung von Grautönen verantwortlich sind. Für das Farbsehen entscheidend sind drei weitere unterschiedlich fotosensible Rezeptorzelltypen, die Zapfen (ca. 6-7 Mio) [GRH02].

S-Zapfen

Die S-Zapfen (Anteil ca. 12%) sind zuständig für die Wahrnehmung der kurzwelligen Frequenz des sichtbaren Spektrums für den Menschen, also das blaue Licht im Bereich von etwa 380-500nm mit einem Absorptionsmaximum bei ca. 419nm.

M-Zapfen

M-Zapfen (variabel) sind in der Lage, Licht mittlerer Wellenlängen wahrzunehmen, etwa im Bereich von 450-630nm, was im sichtbaren Spektrum etwa dem Bereich zwischen der Farbe Blau und Orange entspricht. Das Absorptionsmaximum liegt bei ca. 531nm.

L-Zapfen

L-Zapfen (variabel) sind fotorezeptorisch in der Lage, Licht von langen Wellen im Bereich von 500-700nm mit einem Absorptionsmaximum bei ca. 559nm wahrzunehmen, was dem Gehirn die Darstellung des Farbbildes Rot ermöglicht [SHHH19].

Aus den detektierten Lichtreizen unterschiedlicher Wellenlängen in den drei verschiedenen Zapfenzelltypen des Auges setzt das Gehirn additiv ein korrektes Farbbild zusammen. Dies aber nicht bei allen Menschen. Acht Prozent der männlichen Bevölkerung in Europa haben beispielsweise weniger M- und/oder L-Zapfen und daraus resultierend eine Rot-Grün-Farbsehschwäche [WKL05]¹. Da der Autor der

¹<https://de.wikipedia.org/wiki/Rot-Grün-Sehschwäche>, Seitenrecherche 18.April 2020, 22:48 Uhr.

vorliegenden Arbeit zu dieser Gruppe dazuzählt, möchte er auf die Berücksichtigung dieser Faktoren bei der Entwicklung neuer farbbasierter Distraktoren verweisen c.

In Abbildung 2.4 fällt auf, dass sich die Wahrnehmungsmöglichkeiten der Zapfen überlappen, sie teilen sich ähnliche Bereiche der Wellenlängen, was die Farbwahrnehmung zwischen den drei Hauptfarben erst ermöglicht. Die Wellenlängenmaxima der Zapfen selbst entsprechen nicht Rot, Grün oder Blau, aber in der additiven Mischung bzw. Verarbeitung von Lichtreizen im Gehirn, die den Flächen unter den Absorptionskurven entsprechen, werden diese drei Farben besonders gut wahrgenommen [SHHH19].

Daraus lässt sich für das in dieser Arbeit zu erstellende *Gamedesign* die Hypothese ableiten, dass Objekte oder Texturgestaltungen in der spezifischen Farbgruppierung Rot-Grün-Blau am effizientesten ablenken. Deshalb bedienen sich die im Rahmen des Projektes verwendeten Distraktoren, bzw. die ablenkenden Faktoren hauptsächlich dieser drei Farben.

2.3. Design

Für die Untersuchung von Distraktoren in einer lauffähigen, für die Zielgruppe möglichst intuitiven Anwendung, ist das Design eines Spielprojektes ein wichtiger Faktor. Das Design eines Spiels basiert auf einigen grundlegenden Parametern. Wichtig ist bei jeglichen technischen Anwendungen ein solides *User Interface (UI)*, welches die relevanten Informationen für die Spieler beinhaltet, damit diese sich auf gewisse Situationen einstellen können. Hinzu kommt beim Design der logische Aufbau einer Anwendung, ganz wesentlich dabei sind klare Strukturen [Dah06]. Ein hochwertiges Design verbessert die Qualität der Anwendung und hilft dabei, die Kommunikation mit dem Nutzer zu verbessern.

2.3.1. Das User Interface - UI

Das *UI* ist die Benutzeroberfläche, über die ein Mensch mit einer Maschine in Verbindung tritt. Gleichzeitig ist sie eine Schnittstelle für die Kontrolle der Maschine durch den Nutzer. Das *UI* ist damit wesentlicher Bestandteil der Kommunikation des Bedienenden mit einer Software, sowie Werkzeugen, einer Maschine und anderen

Geräten. Es hilft den Menschen dabei, jene zu verwenden und Informationsrückmeldungen zu verzeichnen [Ch11].

Bei Spielen wird das *UI* oft auch als *GUI* oder Graphisches *User Interface* bezeichnet. Das *UI* sollte immer ein Hauptmenü oder ähnliches besitzen, zu dem man jederzeit und sehr leicht während des Spielens gelangen kann. Dies sollte ganz einfach über eine integrierte Pausenfunktion oder auch durch ein permanentes *In-Game* Menü gewährleistet sein. Ebenfalls gilt es, wichtige Informationen, die den Erfolg des Spiels nachverfolgbar machen, z.B. erreichte oder noch vorhandene Ressourcen über das *UI* darzustellen. Essentiell ist auch der einfache *UI*-Zugang zu den Zielstellungen des Spiels, um nie den Fokus des Spiels zu verlieren, z.B. nach intensiven Abschnitten.

Das *UI* sollte zudem immer möglichst intuitiv und mit sehr wenig Einarbeitungszeit schnell, zumindest in seinen wichtigsten Funktionen, zu erfassen sein. Das kann man erreichen durch spezifische farbliche Gestaltung, Interaktion oder Manipulation.

2.3.2. 3D-Objekte

Dreidimensionale Objekte werden mit Hilfe von Flächen (*Polygones*), Punkten (*Vertices*) und Kanten (*Edges*) erstellt. Diese Bestandteile lassen sich mit 3D-Modeling Programmen manipulieren, modifizieren und ausbauen.

Bei der Gestaltung von 3D-Objekten sollte darauf geachtet werden, dass die Polygone möglichst immer vier *Vertices* und *Edges* haben. In wenigen Fällen können es auch mehr sein, sofern es die Topologie nicht zu sehr beeinflusst. In der Spieleentwicklung wird auf die trianguläre Topologie geachtet, da 3D-Modelle beim Import von Spiele *Engines* trianguliert werden. Die Topologie bezieht sich auf die Gleichverteilung von Polygonen auf einem Modell, sie ist besonders wichtig bei animierbaren Objekten, sowie bei der strukturellen Texturierung.

Hinsichtlich der technischen Erstellung von 3D-Modellen, werden vorwiegend Programme wie *Autodesk Maya*, *Autodesk 3ds Max* sowie *Cinema 4D* verwendet. Ebenfalls für die 3D-Modellierung zu beachten sind *Open Source*-Programme, wie zum Beispiel *Blender*.

In dieser Arbeit wurden die 3D-Modelle im Stil des *Box Modeling* erstellt. *Box Modeling* ist einer der grundlegenden *Workflows* des *Modeling*. Beim *Workflow* der

Box-Modellierung, erstellen sich die 3D-Generalisten, ein einfaches Polygonmodell, welches sie fortlaufend bearbeiten, ergänzen und verschiedene Polygonmodelle zusammenfügen. Dies wird durch verschiedenste Funktionen in den gewählten 3D-Programmen erreicht [Nea11].

2.3.3. 2D-Elemente

In vielen 3D-Projekten werden 2D-Elemente, meist nur für das Texturieren und Strukturieren genutzt oder bei der Darstellung von viel verwendeten Vegetations-*Assets*, wie zum Beispiel Gras.

2D-Elemente bringen unter anderem die Farbe und visuelle Gestaltung zu 3D-Objekten. Damit werden Strukturen und Aussehen durch *Heightmaps*, *Normal Maps* und Texturen, auf die 3D-Poly-Objekte gebracht. Sie sind ebenfalls sehr effizient darin, Wiesen und Blätter in einem Spiel oder einer Szene zu ergänzen.

Die Anwendung von 2D-Elementen bei der Erstellung von Vegetation, erfolgt vor allem, um die Leistungsbelastung von Endgeräten gering zu halten und einen kontinuierlichen Datenfluss während eines Spiels zu sichern. Durch 2D-Elemente können Blätter oder Gräser dennoch voluminös, bewegt, farblich verschieden oder auch schattiert dargestellt werden. Bei Rendersequenzen werden teilweise schon 3D-Elemente für die Vegetation genutzt, jedoch bei einer interaktiven dreidimensionalen Anwendung, wie es Spielelevels sind, sind 2D-Elemente noch weit verbreitet.

Die *Assets* des in dieser Arbeit erstellten Projektes wurden mit dem 3D-Modellierungsprogramm *Autodesk Maya 2018* erstellt.

2.4. Unity

Die Unity Engine ist eine *Game Engine*, die auf das Entwickeln von 2D- und 3D-Anwendungen ausgelegt ist, mittlerweile aber auch für *VR*-Anwendungen (*Virtual Reality*) und *AR*-Anwendungen (*Augmented Reality*) zum Einsatz kommt. Mit Hilfe von verschiedenen Tools unterstützt Sie die Entwickler beim Erstellen von Spielen, oder Filmanimationen und ist auch für die Bereiche Entwicklung und Produktion in der Industrie oder auch im Bauwesen essentiell geworden. Sie besitzt Werkzeuge

für Partikelsysteme, Terraingestaltung sowie grundlegende Animationsmöglichkeiten. All jenes kann noch mit der unterstützten Skript-Sprache C# ergänzt und ausgestaltet werden.

Die Unity Engine wird auf globaler Ebene verwendet. Sie ist vor allem für Einsteiger in der Spieleentwicklung eine beliebte Software, da sie bereits in ihrer freien Version eine gute Anzahl an Arbeitsmöglichkeiten anbietet. Diese sind übersichtlich dargestellt und recht einfach zu nutzen. Etwas komplexer zu bedienen, aber mit weitaus komfortableren und umfangreicheren Arbeitsmöglichkeiten ausgestattet, ist die lizenz- und kostenpflichtige Pro-Version.

Die in der *Engine* entwickelten Apps, können für fast alle Betriebssysteme, Endgeräte und Medien exportiert werden, sofern dies beim Anlegen des Projektes beachtet wird. Das bedeutet, eine Anwendung kann für Mobile Endgeräte, PCs/Mac, Konsolen und wie bereits erwähnt VR- und AR-Geräte zur Verfügung gestellt werden.

Des Weiteren gibt es den *Unity Asset Store*, welcher von der *Community* und Unity selbst gespeist wird. Hier findet man als Anwender eine Vielzahl hilfreicher Dinge, wie auch die in diesem Projekt verwendeten *Standard-Assets*¹. Hinzukommen viele hilfreiche *Shader* und *Tools* zur Gestaltung von virtuellen Landschaften.

2.4.1. Unity Oberfläche

Wie auch viele andere Bearbeitungsprogramme, sei es für Bilder, 3D-Modelle oder einfache Textlayouts, hat auch Unity mehrere Fenster, welche der Nutzer für sich individuell in verschiedene Tabs oder Einzelfenster anlegen kann. Das Standard-Layout für Unity beinhaltet im Hauptbereich das Szenenfenster, in welchem alles gebaut wird, sowie Tabs mit der Möglichkeit zum *Assets Store* und zum *Game Preview* zu gelangen. Es kann auch durch das Drücken des *Play Buttons*, der *Game Preview*-Modus direkt aktiviert werden, ohne zuvor die Tabs zu ändern.

In der *Unity Free* Version ist links in der Standardansicht, die Hierarchie der Szene zu finden. Es ist eine Art Baum, in dem die einzelnen Objekte in ihrer Gruppierung, und nach dem *Parent-Child*-Prinzip angeordnet sind. Jene ist hilfreich beim

¹<https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2017-3-32351>

2. THEORETISCHE GRUNDLAGEN

Gestalten von etwas komplexeren Levels, um die einzeln eingearbeiteten Objekte im dreidimensionalen Raum wieder zu finden.

Auf der rechten Seite der Ansicht sind in der Regel die Konsole, der *Inspector* und oft auch das *Lighting* gelistet. Sobald man ein *Terrain Object* hinzufügt, findet man dort auch noch das *Terrain Tab* mit dem sich vieles an der Gelände-*Plain* anpassen und manipulieren lässt.

Unter dem Hierarchiebaum und der Szenen-Darstellung, finden die *User* bei der voreingestellten Ansicht das Projektverzeichnis. Hier sind alle Dateien zusammengefasst, die implementiert werden, wie im Standard-*Explorer* von Windows oder dem *Finder* bei MacOS. In dem Projektverzeichnis hat man auf den Ordner Zugriff in dem das Unity-Projekt gespeichert ist. Hier werden alle importierten und erstellten *Assets*, Skripts, *Prefabs* und ähnliche Dateien abgelegt, für die Darstellung, Funktionalität und das Erstellen einer Anwendung benötigt werden. Die Abbildung 2.5 stellt eine grobe Übersicht Unity Oberfläche dar.

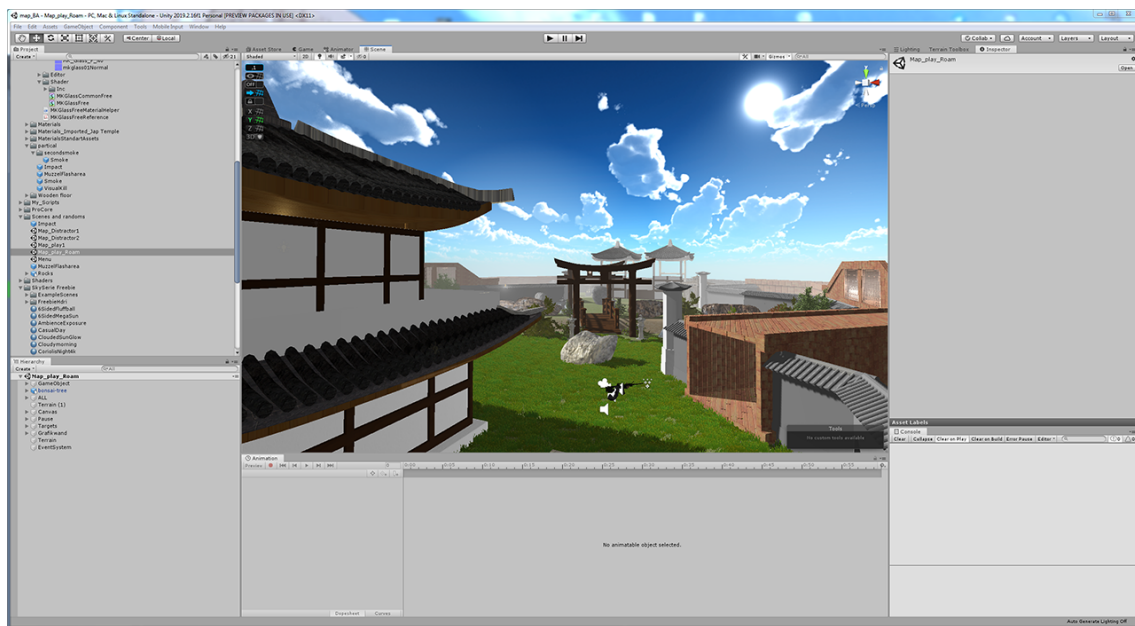


Abbildung 2.5.: Oberfläche Unity Engine.

2.4.2. Szene

Die Szene der Unity Engine beinhaltet den gesamten Inhalt einer Umgebung, von UI und 3D-Elementen zu Partikeleffekten, Belichtung, Animation und teilweise auch Audioquellen. Sofern der User eine Szene erstellt hat, muss diese in den *BuildSettings* eingefügt werden. Ist dies geschehen, kann die Szene mit der Funktion *SceneManager.LoadScene()*; aufgerufen werden. Oft legen sich die Entwickler hierfür ein *Script* an, welches *SceneManger.cs* oder ähnlich benannt wird. Dort werden dann alle *LoadScene()*; Funktionen implementiert [Tec].

2.4.3. Graphic User Interface (GUI)

Es wurde bereits im Abschnitt zuvor auf die *UI* eingegangen. Das *GUI* beinhaltet nur die graphischen Elemente, die in einer Anwendung implementiert werden, um die *UI* für den User intuitiver zu gestalten. Unity hat hierfür eine eigenes *Submenu*, welches grundlegende Funktionen für die *UI*-Gestaltung und Anwendung beinhaltet. Es ermöglicht dem Nutzer von Unity, 2D-Elemente zu erstellen, die als *GameObject* behandelt werden und somit in der Szene zu sehen sind. Grundlage für eine *GUI* ist immer ein *Canvas*, auf dem der Entwickler ergänzend mit *Buttons*, Textfelder oder Interaktionsboxen eine solide *UI* für den Spieler erstellen kann [Tec].

2.4.4. Prefab

Prefabs, das Kürzel für ‚*prefabricated*‘, erklärt sich mit dem Wort selbst. Hierbei handelt es sich um Objekte, *Scripts* oder Materialien, welche bereits zuvor erstellt und nur in die Engine importiert wurden. Jene können in der Szene weiterhin bearbeitet werden, allerdings gilt das importierte Objekt als Vorlage, welche immer wieder in eine Szene unverändert vom Original hineingeladen werden kann [Tec].

2.4.5. FPS-Controller

Der *FPS-Controller*, ist ein Spieleobjekt, welches von dem Anwender gesteuert werden kann. Es steht für *First Person Shooters Controller*. Allerdings kann es auch für nicht *Shooter* orientierte Spiele verwendet werden. Es handelt sich dabei nur

um eine kontrollierbare Kamera, welche mit einem *Rigidbody*, *Collidern*, Aktionen, *Sounds* und Steuerungsmöglichkeiten ausgestattet ist. Der FPS-Controller gibt dem User die Möglichkeit, die Kamera entweder langsamer oder schneller (Gehen und Sprinten) oder vertikal bewegen (Ducken und Springen) zu lassen.

Die Funktionalität des *Controllers* lässt sich gut erweitern, so kann der Entwickler jenen z.B. mit Fliegen, Rutschen und ähnlichen Bewegungsdarstellungen der Kamera ergänzen [Tec].

2.4.6. Collider und Rigidbody

Die *Collider* unterteilen sich in *2D-* und *3D-Collider*. Wie der Name vermuten lässt, handelt es sich hierbei um etwas das „*collided*“ also kollidiert. Es gibt den Spieleobjekten einen Kollisionsbereich, dieser funktioniert als Barriere, damit sich verschiedene Spielobjekte in der Live-Anwendung zum Beispiel nicht überlappen können. *Collider* sind essenziell für eine in der Engine erstellte Anwendung. Gibt es keine *Collider* auf dem *FPS-Controller* oder der Umgebung, so fällt der Anwender beispielsweise einfach durch die erstellte Szene in eine leere Umgebung (siehe Abbildung 2.6 oben).

Bei interagierbaren oder steuerbaren Objekten wird oft ein *Rigidbody* genutzt. Der *Rigidbody* ist ein Attribut, welches es ermöglicht, die Unity Engine *Physics* zu verwenden. Objekte mit dem *Rigidbody* Attribut lassen sich also von den Unity *Physics* beeinflussen, das heißt Gravitation, Kraft und ähnliches aus der Mechanik wirken sich auf diese Objekte aus [Tec].

2.4.7. Skybox

Die *Skybox* umhüllt die gesamte Szene in der Unity Engine, mit deren Hilfe die Impression geschaffen werden soll, dass am Horizont eine komplexe Szenerie existiert. Sie hat ebenfalls Einfluss auf die Belichtung in der Szene, sofern vom Anwender erwünscht. Bei Skybox-Texturen handelt es sich oft um ein hochauflösendes HDR Bild (*High Dynamic Range*), welches in verschiedenen Belichtungen aufgenommen wurde. Es dient zur Darstellung des Levels hinter der eigentlichen Geometrie.

Zur Veranschaulichung dazu Abbildung 2.6 unten [Tec].

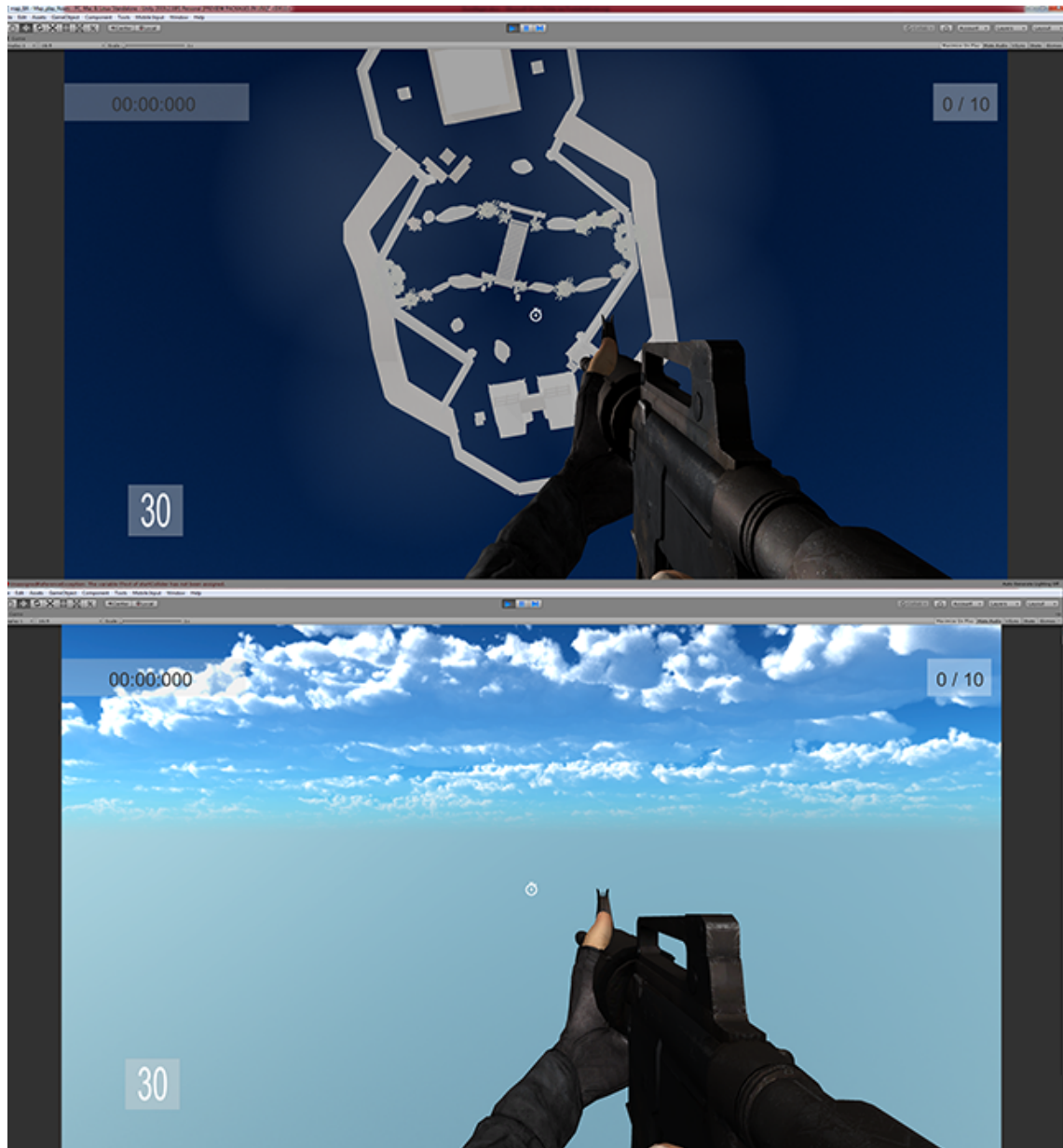


Abbildung 2.6.: Szene ohne *Collider* (Oben). *Skybox* einer Szene (Unten).

3. Konzeptionierung des Präzisionstests

Dieses Kapitel beschäftigt sich mit der Konzeption und dem grundsätzlichen Aufbau des Projektes, berücksichtigt die Implementierung essentieller Funktionen in den Präzisionstest und wird ebenfalls auf spezifische Grundlagen des Leveldesigns eingehen, die für die Bearbeitung der Aufgabenstellung eine Rolle spielen.

Für sämtliche Beschreibungen, die fachspezifische Begriffe enthalten und die innerhalb dieses Kapitels nicht mehr näher erläutert werden, wird auf das vorhergehende Kapitel verwiesen, in dem wissenschaftliche und informative Fakten zu Distraktoren, aber auch zur technischen Seite des Projektes, inklusive des Designs und der Unity dargelegt wurden. Es gibt dort den Fachbegriffen entsprechend zugeordnete Unterkapitel, die ähnlich einem Glossar, allerdings in logischer und nicht alphabetischer Reihenfolge angeordnet sind. Dies kann dazu beitragen, das Verständnis für die Inhalte in diesem Kapitel zu verbessern.

Wie generell in der Spieleentwicklung üblich, galt es auch für diesen Versuchsablauf, *Assets* für einen Level zu erstellen. Der Autor dieser Arbeit, hat sich hierfür eine *Asset*-Liste erdacht, welche in *Autodesk Maya 2018* umgesetzt wurde. Ausgewählt wurde dieses Programm, auf Basis vorhandener Praxiserfahrungen des Modellierers.

3.1. Grundlagen des Leveldesigns in 3D

Das Thema des Experiments, die Wirkung visueller Distraktoren auf Spieler zu prüfen, bezieht sich auf eine von vielen in der Spieleentwicklung vorherrschenden Fragestellungen, die für jedes konkrete Spiel neu beantwortet werden müssen und die die Entscheidung für das Design eines Spieles mitbestimmen. Es ist deshalb, je nach Zielrichtung und Zielgruppe eines Spieles nützlich, viele Vorinformationen zur Wirkungsweise verschiedener Distraktoren zu generieren. Genau dazu trägt der hier

vorgestellte Test bei. Der Test ist spielerisch aufgebaut und bedient sich dementsprechend einiger *Basics* des *Gamedesigns*. Da es sich aber nicht direkt um eine story- oder multiplayerbasierende Spielstruktur handelt, ist hierfür vorwiegend nur der Teil des Level-Designs von Bedeutung.

Die Erstellung eines 3D-Levels ist komplex. Deshalb gibt es spezialisierte Fachgebiete, die sich nur mit dem Design von Levels beschäftigen. Dabei muss die Bezeichnung Leveldesigner auf mehrere Fachgebiete, wie Programmierung, 3D-Art, Layoutdesign u.a., ausgeweitet werden, welche die Levels in der Regel gemeinsam produzieren. Neben der „handwerklichen Arbeit“, und zu lösenden technischen Problemen bei der Gestaltung von Levels, gilt es jedoch eine Frage von weit größerer Bedeutung bei der Levelgestaltung zu berücksichtigen und zu beantworten: „*Is it fun to play?*“ [Moo16].

Die Antwort auf diese Frage ist sehr komplex und sie beinhaltet einige Details mehr als nur den Spiele-Spaßfaktor. Dafür bezieht sich der Autor dieser Arbeit auf die 10 Prinzipien des Leveldesigns nach Dan Taylor. Diese hat Taylor in seiner Präsentation auf der *Game Developer Conference (GDC)* im Jahr 2013 vorgestellt. Dan Taylor war Level-Designer und *Lead Designer* in mehreren großen Unternehmen, zum Beispiel bei SquareEnix / Edios Montreal, wo er unter anderem an Spielen wie „*Deus Ex Mankind Divided*“ und „*Shadow of the Tombraider*“ gearbeitet hat.

Die 10 Prinzipien des Leveldesigns nach Dan Taylor:¹

- *Make it fun to navigate*
- *Visual language*
- *Tells what, but not how*
- *Constantly teaches the player*
- *Make it surprising*
- *Make it risky but rewarding*
- *Empowers the player* (nur im Einzelspielermodus)
- *Is efficient*
- *Is creating emotions* (nur im Einzelspielermodus)
- *Mechanic driven*

¹Ten Principles for Good Level Design

<https://www.gdcvault.com/play/1017803/Ten-Principles-for-Good-Level> [Tay13]

Auch wenn viele der Spiele, an denen er mitgewirkte, Story basiert sind, gelten grundlegende Inhalte der von ihm vorgestellten Prinzipien allgemein für das Level Design in jedem Genre.

In dem im Rahmen dieser Arbeit erstellten Hand-Augen-Koordinationstest sind nur die **Navigation**, die **Visuelle Sprache**, die **Effizienz** und die von der **Spielmechanik beeinflussten Faktoren** bedeutend. Es handelt sich bei der entwickelten Anwendung um ein Präzisionstraining von Probanden im Bezug zu ablenkenden Elementen, wofür eine Testumgebung notwendig ist. Diese soll aber nicht komplett abstrakt, sondern spielähnlich sein, um den Probanden ein Spielgefühl zu geben und deren Interesse am Leveldurchlauf aufrecht zu erhalten. Das Experiment ist somit spielerisch aufgebaut, es geht im Test jedoch nicht grundsätzlich um ein Spiel oder Spiellevels. Emotionale Faktoren spielen im Test daher nur eine untergeordnete Rolle.

3.1.1. Map-Gestaltung

Im Folgenden werden kurz die für den Präzisionstest relevanten Prinzipien des Leveldesigns (siehe Abschnitt 3.1) vorgestellt.

Bei der **Navigation** spielen mehrere Faktoren eine Rolle. Dazu gehören vor allem Möglichkeiten, die die Spieler haben, um sich fortzubewegen, zum Beispiel Springen, Laufen, Gehen, Ducken oder Rutschen. Ebenfalls dabei zu beachten ist, ob ein Level andere Fortbewegungsmittel beinhaltet und was diese können. So benötigt man mehr Fläche, wenn zum Beispiel Fahrzeuge in einem Level zu nutzen sind.

Es ist extrem wichtig, dass die Spieler beim Navigieren innerhalb von Levels nicht frustriert werden, weil diese zu verschachtelt sind, oder sich der Spieler langweilt, weil ein Level zu eintönig wird und wenige Erforschungsmöglichkeiten bietet. Dies kann vor allem durch Hilfsmittel wie eine Übersichtskarte, Weghinweise und ähnliches verhindert werden. Als Negativbeispiel kann hier das Level „Kathedrale des heiligen Blutes“ des Spiels „*Code Vein*“ angeführt werden, welches nur sehr wenige Navigationsinformationen enthält und unübersichtlich ist.²

²https://youtu.be/gf5Ke_CuJXs?t=1130 ab 18:50 Game Two Bericht zu *Code Vein* Kirche des Blutes

Zu bedenken ist ebenso der Einfluss von Gegner- oder Zielplatzierungen im Verlauf einer Map. Dabei gilt es bei einer *Down Phase*, also einer Phase in der längere Zeit nichts Bedeutendes passiert, entsprechend Gegner, visuelle Reize, wie Panoramas, oder Quest-Inhalte zu implementieren. Gleiches gilt nach einer actionreichen Phase, um den Spieler Ausruhen zu lassen, bevor das nächste große Event aktiviert wird.

Die **Visuelle Sprache** in einem Level hat direkten Einfluss auf die Navigation, sowie den Lerneffekt bei den Spielern. Inhalt dieses Prinzips ist es, den Spieler mit Hilfe von visuellen Reizen durch einen Level zu dirigieren. Dies wird beispielsweise mit optischen Akzenten, speziellen Attraktoren [Joi18] und Distraktoren [GB03] erreicht. So gibt es oft auffällig gestaltete Elemente, welche die generelle Aufmerksamkeit der Spieler erregen und ihn möglichst soweit manipulieren sollen, dass er diese strategisch ansteuert und damit bis zu einem entsprechenden Ort hingeleitet wird. So ein Attraktor kann z.B. die *Finish Line* eines Levels sein. Der visuelle Einsatz von Distraktoren hingegen ist entscheidend für die Spielspannung. Er fordert das intuitive, taktische Verhalten des Spielers heraus. Kurzzeitige, teilweise kleinste visuelle Manipulationen des Spielers mitten im laufenden Level können für den Ausgang einer konkreten Spielesituation oder in der Summe betrachtet, z.B. für die benötigte Zeit oder Präzision in einem Level entscheidend sein.

Visuelle Sprache heißt auch, insgesamt eine Orientierung für den Spieler zu geben. Hierzu gehört eine klare Farbstruktur, um etwa Gegner und Verbündete zu identifizieren. Insbesondere bei Multiplayerspielen nimmt jenes eine bedeutende Rolle ein. Dieses Prinzip wird auch verwendet, um klare Seitenstrukturen in einem Spielelevel zu erschaffen, z.B. durch die Lichtgestaltung der Basis oder der *Spawn*-Bereiche von Verbündeten in der jeweiligen Teamfarbe. Somit gibt das Level den Spielenden eine klare Aufteilung vor.

Das Prinzip ein Level **Effizient** zu gestalten, besagt, dass ein Entwickler ein Spiel so entwickeln muss, dass es möglichst wenig Ressourcen in der Nutzung (z.B. *Memory* oder *GPU*-Speicher etc.) verbraucht, bei bestmöglicher Funktion und gutem Design. Ebenso spielt die Produktionszeit für ein Spiel eine wesentliche Rolle. Sie muss möglichst kurz sein, was z.B. durch intelligentes modulares Design erreicht wird. Strukturen, Bauten oder *Assets* werden möglichst in leicht abgewandelter Form vielfach wiederverwendet.

Der Fokus der Testanwendung liegt vorrangig auf der Wiederverwertbarkeit der *Assets* und Grundgerüste. Wiederverwendete Objekte oder Wegstrukturen helfen

dabei, den Spieler zu lehren, wie gewisse Dinge funktionieren und aufgebaut sind. Das gilt unter anderem bei dem erneuten Auftreten von ähnlichen Gegnertypen, Gebäuden oder Bewegungsabläufen. Ein Kartenabschnitt, der gleichartig gestaltet ist, wie ein zuvor Durchlaufener, hilft dem Spieler dabei, das *Movement* zu optimieren und besser auf Gefahren oder Ziele zu reagieren und hat damit direkten Einfluss auf die Schwierigkeit eines Levels.

Der letzte für die zu erstellende Testumgebung relevante Punkt aus den Prinzipien ist, dass eine Karte **an die Spielmechaniken angepasst** sein soll. Dieses Prinzip bezieht sich auf die Funktionen, die dem Spieler zur Verfügung stehen. Und dieses Prinzip kann alle anderen abändern. Sollte der Spieler zum Beispiel eine Taschenlampe haben oder variationsreiche Bewegungsmöglichkeiten besitzen, sind diese Teil der Spielmechaniken, welche direkt Einfluss auf den Levelaufbau nehmen. Besitzt ein Charakter in einem Spiel z.B. *Wall Jumps* so kann ein Level deutlich mehr in der Vertikalen aufgebaut werden, als wenn dies nicht der Fall wäre. Wenn Spieler sich Ducken oder Rutschen können, so ist es möglich, eine Map mit zahlreichen niedrig liegenden Elementen für Seitenwege, Deckung oder ähnlichem auszustatten.

Neben diesen Funktionen spielen auch andere, nicht für die Fortbewegung relevante Mechaniken eine recht große Rolle. Hierzu gehören Faktoren wie Munition, Elemente, welche die visuelle Wahrnehmung verändern können oder der Rückstoß von Waffen, um nur einige zu nennen. All jene haben im Gesamtbild Einfluss auf die Levelgestaltung, damit ein Spiel den Spielern so angepasst, wie möglich, präsentiert werden kann.

3.2. Aufbau des Präzisionstests

3.2.1. Szenenbau

Damit mit der Konzeptionierung einer Szene begonnen werden kann, muss zunächst klargestellt werden, auf welchen Plattformen die Anwendung lauffähig sein soll. Durch das Aufteilen der Anwendung in verschiedene Szenen wird verhindert, dass es zu ausgiebigen Performance-Problemen kommt, da nur der Inhalt der jeweiligen Szene von dem Programm geladen wird. Der Entwickler dieser Anwendung entschied sich für den PC und das Betriebssystem Windows als Plattform.

Für die Navigation sollte ein klar strukturiertes Hauptmenü den Einstieg in die Anwendung präsentieren, von dem der Nutzer ohne Komplikation auf weitere Szenen zugreifen kann.

3.2.2. Menü

Das Hauptmenü dient als eine Art *Hub*, welcher es dem Spieler ermöglicht, durch das Programm zu navigieren, hierzu kommt noch ein Untermenü, das Pausenmenü, welches durch das Drücken der Escape-Taste innerhalb der Levels aufgerufen wird. Um Verwirrung zu vermeiden, ist es von Vorteil intuitiv gestaltete Unterpunkte sowie Bedienungsmöglichkeiten in den Menüs zu nutzen. Hierbei bieten sich die herkömmlichen *Buttons* an, da jene den meisten Benutzern durch andere Anwendungen bereits bekannt sind.

Die *Buttons* benötigen jedoch eine klare Kommunikation für den Anwender, dazu gehören visuelle *Feedbacks*, wie der *Hover*-Effekt und ein Klick-Effekt, sowie eine klare Beschreibung, welche darauf hindeutet, wohin diese *Buttons* führen oder was sie für eine Funktion haben.

Die ***Buttons*** sollten über das einfache Berühren und Klicken, ihrer Funktion nachgehen können. Das Menü beinhaltet die Unterpunkte: ***Start***, ***Roam***, ***Dis-Map1***, ***Dis-Map2*** und ***Quit***. Start steht gleichzeitig für die sogenannte Test-Map oder unveränderte *Normal Map*.

Beim Pausenmenü gilt, was grundlegende Funktionalität betrifft, dasselbe wie beim Hauptmenü, nur ist hierbei zu beachten, dass es sich um ein Menü handelt, was die Anwendung pausiert. Daher sind Stoppfunktionen für die laufende Zeit und das Einfrieren von Audioquellen, sowie der Bewegungsfunktion unabdinglich bei der Gestaltung. Die im Pausenmenü zu findenden Unterpunkte sind: ***Restart***, ***Menü*** und ***Quit***.

Die *Start*, *Roam*, *Dis-Map1* und *Dis-Map2 Buttons* rufen verschiedene spielbare Level auf. Ein im Untermenü zu findender *Button Restart*, setzt das Level zurück, in dem sich ein Nutzer gerade befindet. Somit lassen sich in einer kürzeren Zeitspanne Optimierungsläufe erarbeiten. Mit dem *Button Menu* kommt der Spieler wieder zurück in die Startszene, in der sich das Hauptmenü befindet. Mit *Quit* lässt sich die Applikation in beiden Menüs schließen.

3.2.3. Test-Instruktionen

In den Instruktionen werden der Ablauf des Tests und wichtige Funktionen sowie Details knapp erläutert. Die Beschreibung befindet sich im entsprechenden Ordner der Anwendung. Zur Instruktion dazugehörig ist eine minimalistisch gehaltene Orientierungsmappe des Testkurses.

3.3. Levelgestaltung

Im folgenden Abschnitt wird die Grundstruktur der in der entwickelten Anwendung vorherrschenden Maps beschrieben, sowie auf die konkrete Gestaltung jener eingegangen.

Die in der Anwendung erstellten Maps, richten sich in ihrem Aufbau nach *Tutorial Levels* von bekannten *First Person Shootern*, als Beispiel, die Map Killhouse aus „*Call of Duty 4: Modern Warfare*“ (2007). Die Idee eines *Tutorial Levels* ist es, dem Spieler in einer möglichst kompakt gestalteten Umgebung die Mechaniken des Spieles beizubringen. In dem hier erstellten Projekt beruhen diese auf Fortbewegung, Mausbewegungen, sowie der Schuss-, Schaden-, und Munitionsmechanik. Die Dimension des Grundgestells der Karten ist dementsprechend klein gehalten, und leicht zu überschauen. Mit einer 360 Grad Rotation kann der Spieler somit oft schon große Teile oder je nach Position fast die gesamte Map erfassen.

Im vorliegenden Projekt ist die Grundstruktur der Map spiegelsymmetrisch gehalten, sämtliche *Assets* finden sich in den Karten der unterschiedlichen Levels wieder. Dies soll das schnelle Einprägen des Layouts der Karte unterstützen. Damit wird der Grundgedanke der Arbeit unterstützt, dass es sich bei der Anwendung um Präzisionszielübungen handelt, welche nur durch visuelle Faktoren beeinflusst werden sollen. Die hier vorkommenden Maps sind somit gleich gehalten und nur die Textur variiert, *Shader* kommen teilweise zum Einsatz und einige Ziele wurden zwischen den Maps in ihrer Anordnung minimal deplatziert.

Die Maps bestehen aus zwei Seitengängen, welche identisch aufgebaut sind. Sie haben eine leichte Erhebung in der Mitte, mit Fenstern, die auf den zentralen Bereich gerichtet sind, wo unter anderem eine Brücke und zwei Wasserscrein-Torbögen aufzufinden sind. Des Weiteren gibt es zwei Gebäude, die es ermöglichen, auf eine andere

Höhenlage zu gelangen, von der aus, die Karte überblickt werden kann. Umgeben ist das Level von Mauern, die die Spielfläche arenaartig abgrenzen.

Der vertikale Raum der Levels ist so gering wie möglich gehalten, weil dies die Karten sonst unnötig verkompliziert und das Platzieren der zu zerstörenden Ziele auf zu vielen unterschiedlichen Niveaus zu konfus geworden wäre. Damit soll erreicht werden, dass die Zielsuche nicht zu komplex und frustrierend für die Probanden wird und die Durchläufe erfolgreich beendet werden können.

Das Design der Level orientiert sich an einem Japanischen Garten. So haben die Gebäude entsprechend ostasiatisch wirkende Architektureigenschaften, wie das geschwungene *yosemune* (Walmdach) oder das *hogyo* (Pyramidendach).¹ Ergänzt wurde dieses Design mit, für die westliche Welt stereotypisch erscheinenden japanischen Baustrukturen, wie die Wasserscrein-Torbögen und die für Japanische Gärten oft typischen Granitlaternen.

Abbildung A.4. im Anhang gibt einen groben Überblick über die gestaltete Testumgebung. Im Beispiel die *Top-On*-Darstellung der *Normal Map*. Die Ansicht gilt als Orientierung für die Zielpositionen.

3.3.1. Roam - Roaming Map

Diese frei begehbare Karte ist im Hauptmenü der Anwendung über das *Button Roam* zu erreichen. Sie stellt die Übungs-Map dar.

In diesem Level galt es für den Spieler zu lernen, dass es begrenzte Ressourcen gibt, der Schuss ohne Flugzeit direkt dort auftrifft, wo der Zielpunkt des Fadenkreuzes ist, und dass man als Spieler sprinten sowie springen kann.

Damit dies erreicht wird und der Spieler sich die Basisfunktionalitäten ohne Ablenkung aneignen kann, gibt es in der *Roam Map* keinen aktivierbaren *Timer* oder vorgegebene Laufwege. Als Orientierungshilfe findet der Nutzer bereits auf dieser Map entsprechend Ziele, die zerstörbar sind. In den beiliegenden Instruktionen, wird klargestellt, dass diese Ziele in ähnlichen Positionen platziert wurden, wie in den darauffolgenden Testkarten. Ebenfalls können sich die Spieler hier gut darauf einrichten, dass für jedes Ziel zwei Treffer benötigt werden und nicht jedes gesichtete

¹um 03:57 08.05.2020 https://de.wikipedia.org/wiki/Japanische_Tempelarchitektur

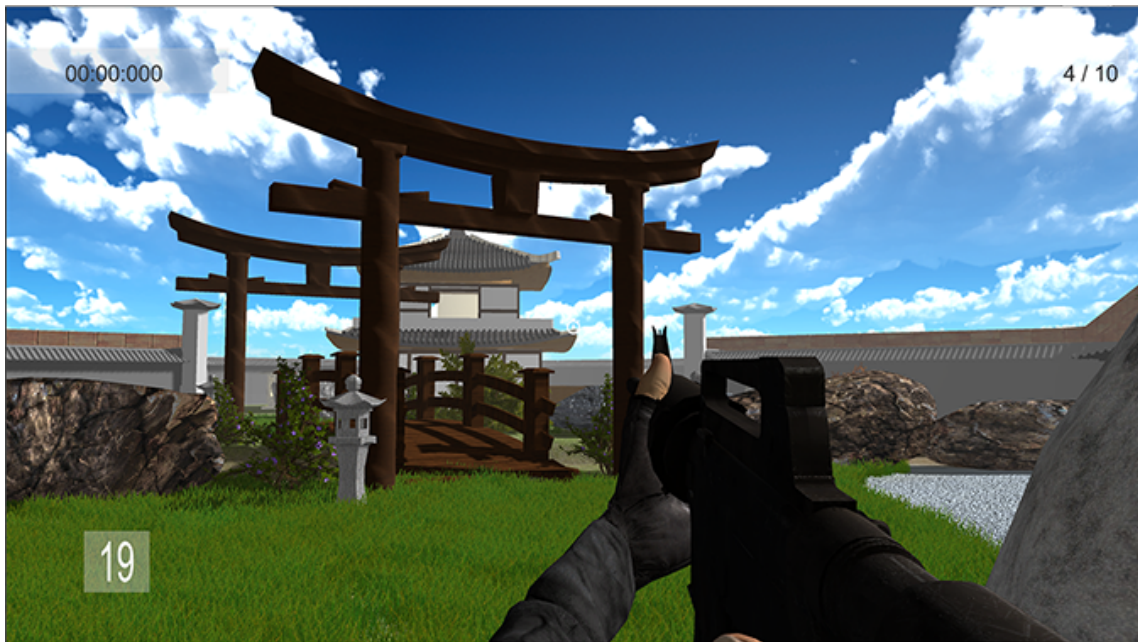


Abbildung 3.1.: Aussehen der *Roam Map* in *First Person*.

Ziel aus jeder beliebigen Position heraus getroffen werden kann. Der Spieler lernt damit auch, dass er je Ziel nur einen Fehltreffer haben kann.

Die *Roam Map* wurde mit einer recht hellen und lebhaften Belichtung ausgestattet, um die Strukturen und den Aufbau der Map so klar wie möglich zu gestalten und eine Distraction durch *Assets* zu vermeiden. Unterstützt wird die positive Grundstimmung der Übungskarte durch eine Schön-Wetter-*Skybox*, welche die Szene mit viel Sonnenlicht bei heiterem Himmel unterlegt (Abbildung 3.1).

Die auf den Objekten verwendeten Texturen sind an die reale Welt angelehnt, so hat man Holzstrukturen oder Glasbausteine, Gebäude mit Putz sowie entsprechende Bodenstrukturen, wie Stein- Holz-, Sand- oder Wiesenboden. Die *Terrain Plain* des gesamten Levels basiert dabei auf einer Grasstruktur.

Der Grundidee der *Roaming Map* soll es sein, dass die Spieler sich auf dieser Karte nicht überfordert fühlen und sich in einer entspannten Situation die Mechaniken der Anwendung aneignen und sich mit den *Physics* des *FPS-Controllers* vertraut machen können. Die Funktionalitäten und Informationen sind so ausgelegt, dass sie schnell für die Anwender zu erfassen sein sollten, sofern sich diese im FPS-Genre etwas auskennen.

3.3.2. Normal Map - Test-Map1

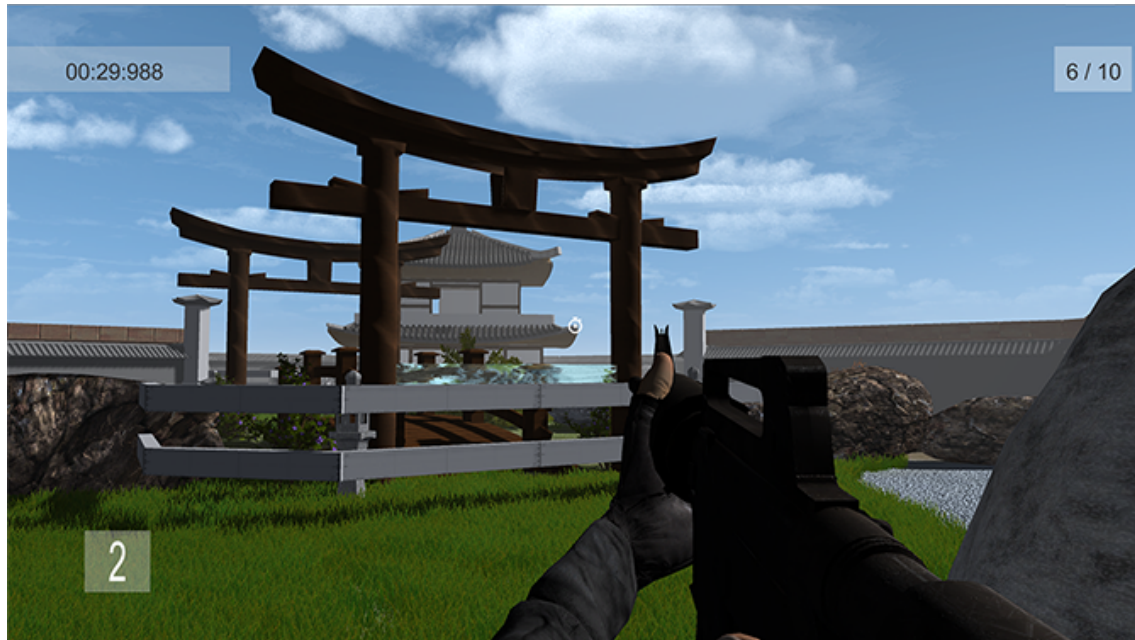


Abbildung 3.2.: Aussehen der ersten Testmap in *First Person*.

Die erste Map des Experiments ist über den **Start-Button** im Hauptmenü zu erreichen.

Dieses Level, ist das erste, in dem sich die Spieler beim Umgang mit *First Person Shooter* Funktionalitäten in allgemeiner Präzision beweisen müssen. In jener Karte, wurden Laufwege im Groben vorgegeben und es wurde über eine virtuelle, gläsern wirkende Startbarriere eine Möglichkeit zur Aktivierung eines *Timers* implementiert.

In dieser Karte soll der Proband beim Durchlaufen der *Start Barrier* alle Ziele auf dem Weg bis zum Erreichen und Durchlaufen der *Finish Barrier* zerstören. Die Map unterscheidet sich von der *Roam Map* nur sehr geringfügig. Neben der offensichtlichen Einschränkung der Laufwegfreiheit, durch verschiedene *Collider* Objekte, ist auch das *Lighting* der Map etwas kontrastärmer und weniger gesättigt (Abbildung 3.2). Die etwas andere Beleuchtung, ist aber durchaus realistisch gehalten und unterstützt den Probanden dabei, in einer normal aussehenden Umgebung oder auch Kontroll-Map-Umgebung, die Ziele zu finden und abzuschießen. Das violett/blau *highlighting* der Ziele ist hierbei besser zu erkennen, als in den anderen Testkursen.

An den Objektdarstellungen wurde nichts geändert, weiterhin sind Strukturen und Texturen, welche an die reale Welt angelehnt sind, in Verwendung.

Die Zielplatzierung ist identisch zu der *Roam* Map und gibt den Spielern somit einen Wissensvorteil, sollten diese die *Roaming* Map entsprechend lange einstudiert haben.

3.3.3. Dis-Map1 - Distraktor-Map1

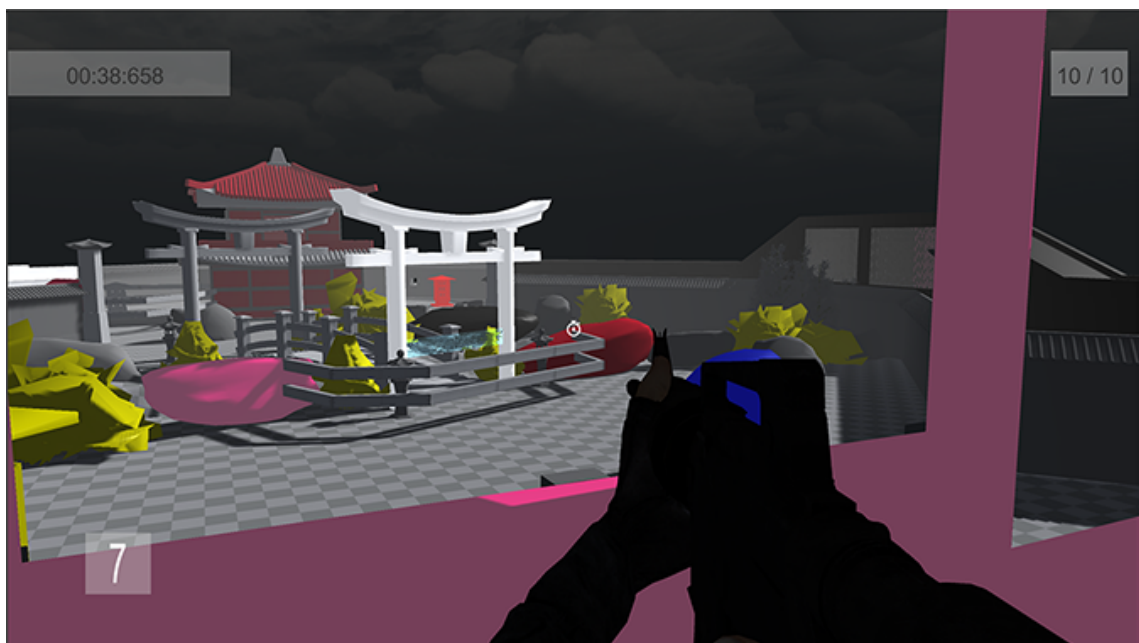


Abbildung 3.3.: Aussehen der ersten Distraktor-Map in *First Person*.

Zu erreichen ist dieser Kurs durch das Drücken des **Dis-Map1-Buttons** im Hauptmenü.

In dieser Karte werden die ersten extremen Abänderungen eingeführt, um die Funktionalität von visuellen Distraktoren besser nachvollziehen zu können. Objekte auf dieser Map sind vor allem in starken, sehr präsenten Farben gestaltet, die sich vor allem der Hauptfarben des menschlichen Sehens Rot, Grün und Blau bedienen. Der Boden auf dem sich der Spieler bewegt, ist in einem matten Grau gehalten, um den ablenkenden Objekten mehr Präsenz zu verschaffen. Dies wird durch die dunklere *Skybox* nochmals verstärkt (Abbildung 3.3).

Die Laufwege, die die Spieler auf der Karte absolvieren müssen, sind identisch zu der ersten normalen Map. Es gibt nur minimale Verschiebungen in den Positionen einzelner Ziele, um Trainingseffekte aus den Übungsläufen und dem ersten Testdurchlauf der *Normal Map* zu reduzieren.

Durch die weniger reflektierenden Oberflächen, ist das *Highlighting* der Ziele in diesem Level nicht optimal wahrzunehmen, was das Orten jener etwas schwerer gestalten soll.

Des Weiteren wurde die Belichtung erneut verringert und essenzielle Strukturbeleuchtungen aus den Gängen entfernt. Dadurch wird die Aufmerksamkeit auf die Ziele in den Gängen zwar stark erhöht, es sind die einzigen farblichen Erscheinungen dort, jedoch sollte die fehlende Strukturierung der Gänge auch zu einer gewissen Ablenkung und Desorientierung führen.

Ziel dieser Karte ist es, die Spieler mit Hilfe der plötzlich hinter den Gängen auftauchenden und auffallenden, farbigen Objekte in ihrer Lauf- und Zielgeschwindigkeit sowie Zielgenauigkeit zu beeinflussen und sie letztendlich von der sofortigen Anpeilung bereits durch das Üben und den ersten Durchlauf abgespeicherter und somit erwarteter Zielorte abzulenken. Es wird angenommen, dass dies am Ende dazu führt, dass die Probanden den Kurs in einer langsameren Zeit absolvieren.

3.3.4. Dis-Map2 - Distraktor-Map2

Die letzte Map des Experiments ist über den **Dis-Map2-Button** im Hauptmenü zu erreichen.

In dieser Map der Anwendung galt es, über die Farbgestaltung eine Überlastung der Spieler mit visuellen Informationen zu erzielen und die Erfassung, der für das Level essenziellen Ziel-Elemente, möglichst deutlich zu erschweren (Abbildung 3.4). Das Level ist deshalb gefüllt mit unterschiedlichen, chaotischen Farbtexturen, welche in verschiedensten Kontrasten zueinanderstehen. Es bedient sich ebenfalls vorwiegend der drei Hauptfarben Rot, Grün und Blau und verschiedener Helligkeitsvarianten davon. Des Weiteren wurden hier transparente *Shader* eingefügt, welche zu unvorhersehbaren Effekten führen, wie zum Beispiel der kompletten Transparenz von Objekten, sofern diese aus einem falschen Winkel betrachtet werden.

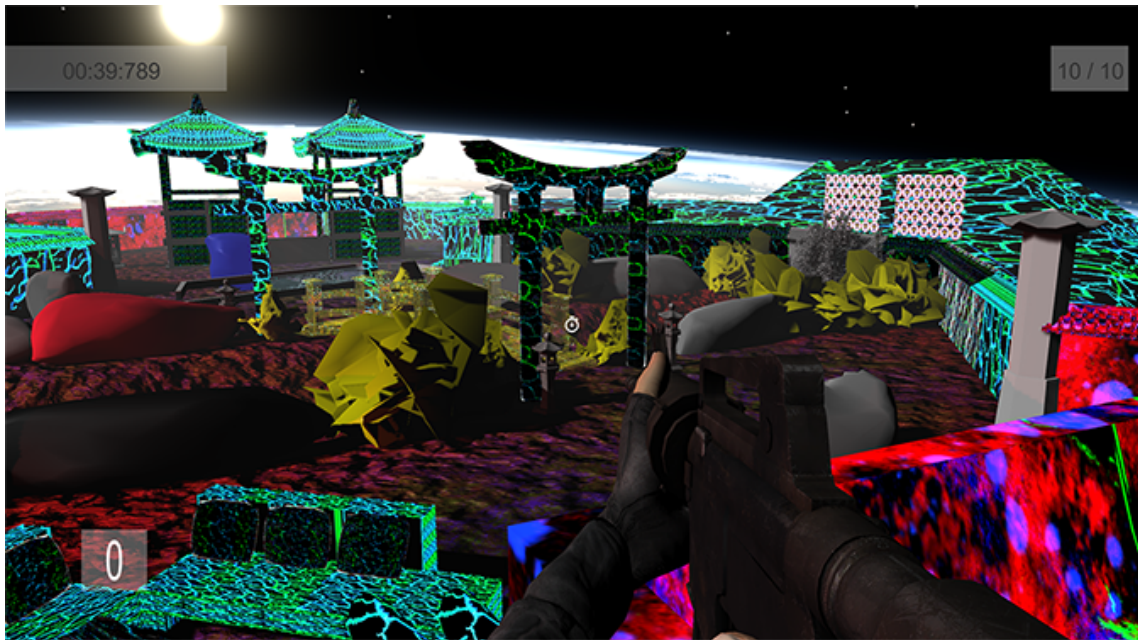


Abbildung 3.4.: Aussehen der zweiten Distraktor-Map in *First Person*.

Die Idee hinter dem Chaos in der Farbgestaltung dieser Map ist, dass die Probanden es bereits durch die rein farbliche Komplexität der Umgebung schwerer haben sollen, die Ziele in der Umgebung auszumachen und somit noch mehr Zeit benötigen müssten, als etwa in Dis-Map1 für einen Treffer. Es wird erwartet, dass die Augen der Spieler von einem starken farblichen Distraktor zum nächsten wandern, bevor sie sich letztendlich auf das tatsächliche Ziel konzentrieren werden. Die Textur der Ziele wurde bei verschiedenen Umgebungsobjekten verwendet, um zusätzliche Verwirrung zu schaffen, weil die Probanden zum Zeitpunkt des Durchlaufens dieser Map bereits das Design der Zielscheiben und auch deren potentielle Position gut kennen und abgespeichert haben. Um Trainingseffekte möglichst auszuschließen, die Ziele aber für die Spieler grundsätzlich auffindbar zu lassen, wurden die Ziele in diesem Level ergänzend teilweise neu platziert. Dies war auch aufgrund des Einfügens der speziellen *Shader* und der allgemeinen farblichen Vielfalt notwendig. Insgesamt sollen die antrainierten Zielbewegungen hierdurch behindert und neue Reize geschaffen werden.

Jegliche verwendeten Texturen, die in diesem Präzisionskurs eingebaut wurden, sind völlig unreal und übertrieben. Die *Skybox*, welche den Orbit der Erde darstellt, verstärkt den Effekt der Absurdität dieser visuellen Levelgestaltung nochmals.

3.4. Klassenbetrachtung

In diesem Kapitel wird kurz auf die für die Funktion des Tests wichtigsten Klassen eingegangen, die vom Entwickler selbst erstellt wurden. Dazu gehören, das *Gun Script*, die *Target Scripts* und der *Timer*. Die meisten *Scripts* wurden in *Microsoft Visual Studio 2019* erstellt. Einige weitere Skripte sind aus dem *Standard Asset Pack* von *Unity Essentials* importiert worden. Einige weniger relevante Skripte für die Funktion der Anwendung sind in diesem Kapitel nicht detailliert aufgeführt. Es handelt sich dabei um kleinere Skripte, welche eher als „*Quality of Life*“ oder UI-Funktionen eingefügt wurden. Alle selbst erstellten, in der Anwendung verwendeten Skripte sind vollständig im Anhang zu finden.

3.4.1. gun.cs

In dieser Arbeit wurde eine Anwendung auf Basis eines *Shooter* Spieles entwickelt, dementsprechend ist ein *Script*, welches es ermöglicht zu schießen, essenziell für die Durchführung. Im *Gun Script* ist dies zu finden, neben einigen kleineren Zusätzen, die Sound und Animation beinhalten, um dem Spieler ein gewisses Gefühl von Kraft zu suggerieren. Das Skript beinhaltet alle Informationen, die dem Gewehr in dieser Anwendung zugeteilt wurden, dazu zählen die Munition, der Schaden, die Feuerrate und die Reichweite. Ebenfalls sind Faktoren, wie die *Impact Force* in der Klasse zu finden. Dies wird aber nicht weiter betrachtet, da es in der Anwendung keine Situation gibt, in der sie von Relevanz wäre. Kurz zusammengefasst, kann man sagen, *Impact Force* nimmt Einfluss auf eine *Rigidbody* und kann daher als Stoppkraft der *Ingame* Waffe bezeichnet werden.

Folgendes Skript beinhaltet die Variablen für die Klasse:

```
1 public class gun : MonoBehaviour {
2
3     public float damage = 10f;
4     public float range = 100f;
5     public float impactForce = 20f;
6     public float fireRate = 15f;
7
8     public float maxAmmo = 30;
9     public float currentAmmo;
10    public bool IsEmpty = false;
11
12    public Transform ShotText;
13
14    private float nextTimeToFire = 0f;
15
16    public Camera fpsCam;
17    public ParticleSystem muzzleFlash;
18    public GameObject impactEffect;
19
20
21    public AudioSource GunShotSound;
22    public AudioSource GunClipping;
```

Methoden

Die erste *default* Methode, welche in *MS Visual Studios* in Kombination mit Unity, als *Start()* bezeichnet wird, beinhaltet in diesem *Script* nur notwendige Informationen, welche am Anfang einer Szene abgerufen werden müssen. Bei der *gun.cs* gilt dies nur für den Sound und die Munitionsmenge.

```
1 void Start()
2 {
3     GunShotSound = GetComponent<AudioSource>();
4
5     currentAmmo = maxAmmo;
6 }
```

Im nächsten Schritt wurde wiederum zunächst auf eine *default* Methode zurückgegriffen, nämlich auf *Update()*. In diese wurden dann Funktionen implementiert,

welche wiederholt abgerufen werden müssen. Dementsprechend ist hier die Aktivierung eines Schusses per Mausklick, welche in *Unity* als „Fire1“ vordefiniert ist, zu finden. Ebenfalls wird hier auch die Menge der restlichen Munition überprüft, um bei einem leeren Magazin den *Weapon Clip Sound* abzuspielen und die Schadensfunktion auszuscha­len.

```
1      void Update()
2      {
3          if (Input.GetButtonDown("Fire1") && Time.time >=
              nextTimeToFire && IsEmpty == false)
4          {
5              nextTimeToFire = Time.time + 1f / fireRate;
6              Shoot();
7          }
8          else if(Input.GetButtonDown("Fire1") && IsEmpty == true)
9          {
10             nextTimeToFire = Time.time + 1f / fireRate;
11             Empty();
12          }
13
14          if (currentAmmo == 0)
15          {
16              IsEmpty = true;
17          }
18      }
```

Die letzte Methode *Void Shoot()* stellt das Herzstück des Skriptes dar. Hier kommt die eigentliche Schussfunktion zu Stande und es werden alle zusätzlichen Effekte mit jener kontrolliert sowie die Munitionsreduzierung durchgeführt.

Für die entwickelte Anwendung wurde festgelegt, dass die direkte Schussfunktion mit einem *RayCast* ausgeführt wird, das bedeutet allgemein beschrieben, dass ein Vektor-Ray von der Mitte der zu betrachtenden Kamera (in diesem Falle die *FPS-Controller* Kamera) gerade nach vorne „herausgeschossen“ wird.

Die Waffe wird in diesem Fall zu einer *Hitscan*, bei der ein Schuss ohne Flugzeit auf ein anvisiertes und beschossenes Ziel trifft. Der Treffer wird dann dahingehend überprüft, ob das getroffene *GameObject* ein *Target* im Spiel darstellt. Ist ein *Target* getroffen worden, wird geprüft, ob es Lebenspunkte besitzt und entsprechend des Schadens werden Lebenspunkte – hier 10 Punkte je Treffer – abgezogen.

```
1
2     void Shoot(){
3         muzzleFlash.Play();
4         GunShotSound.Play();
5
6         currentAmmo--;
7         ShotText.GetComponent<Text>().text = currentAmmo.ToString
            ();
8
9         RaycastHit hit;
10        if (Physics.Raycast(fpsCam.transform.position, fpsCam.
            transform.forward, out hit, range)){                Debug
            .Log(hit.transform.name);
11
12            Target target = hit.transform.GetComponent<Target>();
13            if (target != null){
14                target.TakeDamage(damage);
15            }
16            if (hit.rigidbody != null){
17                hit.rigidbody.AddForce(-hit.normal * impactForce);
18            }
19            GameObject ImpactGO = Instantiate(impactEffect, hit.
            point, Quaternion.LookRotation(hit.normal));
20            Destroy(ImpactGO, 1f);
21        }
22    }
```

3.4.2. Timer.cs und startColider.cs

Die *Timer()* Klasse ist, wie bereits der Name zeigt, eine Klasse, die genutzt wurde, um die Zeit in den Levels zu erfassen. Hierbei sind die wichtigsten Attribute, das Starten und Stoppen des *Timers* sowie in welcher Form die Zeit erfasst werden soll. Um die Aktivierung der Stoppuhr zu kontrollieren wurde in die Anwendung entsprechend ein *Boolean TimerOn* implementiert, welcher überprüft, ob der *Timer* aktiviert wurde oder ob er sich im Ruhezustand befindet. Jener wird vor jedem Start der Szene auf *False* gesetzt und garantiert somit, dass der Timer nicht schon läuft bevor die entsprechenden *Trigger* aktiviert wurden.

Um die Zeit abzulesen, wurde *deltaTime* verwendet, hierbei handelt es sich um eines der *Defaults*, welche in Kombination mit *Unity* vorgegeben sind. Kurz erklärt, liest *deltaTime* die Sekunden ab, welche seit dem letzten *Frame* vergangen sind.³

Um die Zeit in *Void Update()* darzustellen, wurden in dieser Klasse die entsprechenden Zeiteinheiten Minute, Sekunde und Millisekunden mit einem *String* auf das *UI* übermittelt.

Das wiederum funktioniert nur in Kombination mit einer Klasse, die die *Trigger* bestimmt und kontrolliert. Dies wird durch *startCollider()* erreicht. Mit Hilfe dieses Skriptes, werden die beiden Barrieren *Start* und *Finish* in der Anwendung geprüft. Es wird die *Timer* Klasse zum Start der Szene von diesem Skript aufgerufen, anschließend wird betrachtet, ob ein *Rigidbody* mit der Barriere kollidiert, hier als *box1* und *box2* bezeichnet, und entsprechend startet bzw. stoppt somit der *Timer*. Beide Klassen sind vollständig im Anhang dargestellt.

³<https://docs.unity3d.com/ScriptReference/Time-deltaTime.html>

3.4.3. Target.cs

Zuletzt wird noch auf das kleine *Script Targets* eingegangen. Diese Klasse beinhaltet nicht viel, ihre Inhalte sind aber essenziell für die Anwendung. Hierbei werden lediglich die Lebenspunkte der Ziele definiert und die Effekte aktiviert, welche beim Zerstören der Ziele zustande kommen. Derartige Effekte sind für das *Gameplay*-Gefühl und dessen *Flow* wichtig, da es den Spielern eine präsenste Rückmeldung beim Zerstören der Ziele und gleichzeitig damit eine gewisse Befriedigung über das Erreichte gibt. Die Methode *TakeDamage()* ist zuständig dafür, dass ein Ziel, nachdem es getroffen wurde, Schaden angerechnet bekommt und das *Target* mit der Methode *Die()* zerstört wird, wenn keine Lebenspunkte mehr übrig sind.

In der Methode *Die()* werden alle darin implementierten Effekte abgespielt und dargestellt, die bei der Zerstörung möglich sind und das Ziel wird aus der Szene entfernt. Ebenfalls wird hier das eliminierte Ziel zur *Score*-Anzeige in der UI hinzugefügt.

```
1 public class Target : MonoBehaviour {
2
3     public ParticleSystem VisualKill;
4     public AudioSource targetDestroyed;
5     public float health = 20f;
6
7     public void TakeDamage(float amount)
8     {
9         health -= amount;
10        if (health <= 0f)
11        {
12            Die();
13        }
14    }
15    void Die()
16    {
17        Instantiate(VisualKill, transform.position, Quaternion.
            identity);
18
19        Destroy(gameObject);
20        scoreScript.scoreValue += 1;
21        targetDestroyed.Play();
22    }
23 }
```

4. Evaluation

4.1. Durchführung und Auswertung des Tests

4.1.1. Testbeschreibung und Ablauf

In dieser Arbeit galt es, den möglichen Einfluss von Distraktoren, bzw. ablenkenden Elementen auf die Hand-Augen-Koordination von Probanden in einem virtuellen Präzisionstraining zu testen, welches grundsätzlich dem Aufbau von *FPS Games* nachempfunden wurde. Wie in Kapitel 3.3. detailliert dargestellt, wurde hierfür eine Standard-*Shooter*-Szene entwickelt, die einem kurzen Spielelevel gleich kommt. Dazu wurden eine *Roaming* Map und darauf basierend drei zueinander variierende Test-Maps angefertigt. Die Test-Maps entsprechen in ihrem grundlegenden Aufbau einander und dienen als Testkurs. Jede dieser Test-Maps wurde mit verschiedenen Distraktoren ausgestattet, die sich in ihrer Funktion teilweise unterscheiden. Insgesamt müssen während eines Durchlaufs 10 Ziele identifiziert und zerstört werden. Alle Levels sind auf der kürzesten Strecke in etwa 27.000s ohne Zielbeseitigung zu durchlaufen. Die Zeit entspricht einem Durchschnittswert, der vom Entwickler nach 10-maligem Durchlaufen des Levels ermittelt wurde. Die Probanden durchlaufen alle Maps. Wesentliche physikalische Parameter, wie die benötigte Zeit, Abschüsse, Munitionsverbrauch etc., aber vor allem auch subjektive Spieleindrücke, wie die gefühlten Schwierigkeiten bei der Orientierung auf die Spielaufgabe, bei zunehmender Komplexität der Maps, werden protokolliert.

Der Anwendung beigelegt sind Test-Instruktionen, welche für die Probanden Anweisungen und Hilfestellungen für die Durchführung des Tests beinhalten.

Anfangs sollen die Testenden, nach dem Start der Anwendung, die *Roaming* Map unter dem Menüpunkt **Roam** aufrufen, um sich mit den Mechaniken und dem Grundlayout der verschiedenen Levels auseinanderzusetzen. Die Zeit dafür ist unbegrenzt. Ist dies geschehen und die Probanden sind der Überzeugung, dass sie ein

gutes Grundverständnis für die Bedienung des *FPS-Controllers* und der Anwendung entwickelt haben, folgt der erste Testkurs. In der **Normal Map** gilt es für die Probanden, die Ziele möglichst schnell und optimiert in einer teilweise vorgegebenen Reihenfolge zu eliminieren und die Zielbarriere zu durchlaufen. Die endgültige Zeit ist zu dokumentieren. Gleiches gilt für die darauffolgenden Maps, **Dis-Map1** und **Dis-Map2**. Es ist vorgegeben, dass zunächst jede der Maps ein einziges Mal durchlaufen wird, bevor Optimierungsläufe gestartet werden dürfen. Anschließend sollen die Teilnehmer einzelne Maps mehrmals durchlaufen, um ihre Zeiten zu optimieren. Dabei gilt es, ein Level mindestens 2-3-mal erneut zu spielen, um bei Bedarf den Durchlauf von Karten mit einem gefühlten bzw. messbarem höherem Schwierigkeitsgrad zu perfektionieren. Danach soll die beste Zeit der Optimierungsläufe in die Auswertung eingehen. Das Üben soll dazu beitragen, zu vermeiden, dass die Testteilnehmer aufgrund vorhandener potentieller Distraktoren in einer der Maps die Orientierung beim Durchlaufen verlieren bzw. diese zu schnell verlieren. Wichtig für die Auswertung ist, dass die Spieler alle Testdurchläufe erfolgreich beenden, um möglichst vollständig Rückschlüsse auf die Ablenkungen der Spieler bei der Zielsuche, -erfassung, Treffergenauigkeit etc. ziehen zu können.

Die Durchführung der Tests soll von den Probanden auf deren eigenen Computern durchgeführt werden. Während der Durchführung stehen der Entwickler und der jeweilige Testläufer über eine private *VOIP*-Verbindung, für den Fall, dass der jeweilige Spieler Fragen zum Spielablauf hat oder technische Probleme aufgetreten. Einer der Probanden bekommt diese zusätzliche Unterstützung während der Testläufe nicht, sondern ist ab Testbeginn auf sich allein gestellt.

Am Ende der Testsession beantworten alle Probanden einen kurzen Fragebogen des Entwicklers, dessen Antworten in die subjektive Auswertung des Tests einfließen. Einige dieser Fragen waren bereits der Instruktion beigelegt, während anderweitiges Feedback und Fragen in der *Livesession* stattfanden.

Der Test soll von 5 Probanden durchgeführt werden. Die Teilnehmer werden nach Spielstil und Spielerfahrung beurteilt.

Bei der Durchführung wurden abseits der schriftlich vermerkten Fragen noch weitere aus Sicht des Entwicklers interessante Faktoren betrachtet.

Allgemein:

- Wie lange hält sich der Proband auf der Roam Map auf - Beschränkung auf lang, mittel und kurz?
- Welches der Ziele ergab die meisten Probleme während der Übungsphase?
- Für welches Level benötigten die Probanden die meiste Zeit, welches wurde am schnellsten absolviert?
- Gab es Unterschiede bei den Laufweg-Entscheidungen?
- Was waren die schwersten Zielscheiben in den Testläufen?

4.2. Einzeltestdurchführung und Ergebnisanalyse

Es wurden 5 männliche Probanden (Alter 20-30) inklusive des Entwicklers getestet. Für diese erfolgte eine subjektive Einstufung im Hinblick auf den Spielstil und die Spielerfahrung. Unter den Probanden befanden sich ein professioneller Spieler von *Call of Duty 4: Modern Warfare* (CoD), sowie ein erfahrener Spieler in dem Taktik *Shooter Rainbow Six: Siege* (R6). Ein weiterer Spieler hatte viel Erfahrung mit *Counterstrike: Source* (CS:S). Der vierte Proband ist aktiver Spieler von *Large- Scale Shooters*-Spielen wie *Battlefield* (BF), bevorzugt darin jedoch grundsätzlich eine passive Spielweise. Final ist der Entwickler selbst unter den Testteilnehmern mit Vorerfahrung als *Casual*-Spieler in einer Reihe verschiedener *Shooter* Genres.

4.2.1. Teilnehmer 1 - professioneller CoD-Spieler

Professionell bezieht sich in diesem Kontext auf eine immens hohe Anzahl an Spielstunden und Teilnahme an verschiedenen kleinen ESL-Turnieren *Electronic Sports League*.

Der erste Probandentest wurde am 5. April 2020, um 16:00 Uhr für 2 Stunden mit dem CoD-Spieler durchgeführt. Jener Teilnehmer hat sich am längsten, insgesamt ca. 45min, auf der *Roaming* Map aufgehalten, um sich die Zielpositionen einzuprägen und auch entsprechend mit den Mechaniken sowie den *Physics* der Anwendung vertraut zu machen. Die meisten Schwierigkeiten bereiteten dem Teilnehmer zunächst die vertikal außerhalb der Spielersicht platzierten Ziele, welche ohne Vorwissen nicht spontan oder intuitiv gefunden werden können. Durch Blick auf die beigelegte minimalistisch gehaltene Map-Hilfe, hat der Teilnehmende diese jedoch schnell finden können und entsprechend in seinen Ablauf eingearbeitet.

Normal Map

Teilnehmer 1 hat die erste Map, sehr gut mit einer Zeit von 35.500s absolviert. Diese recht schnelle Zeit war aufgrund der sehr langen Vorbereitungszeit im Übungslevel zu erwarten. Die gilt umso mehr, als dass die *Normal* Map keinerlei zusätzlich platzierte Distraktoren beinhaltet und die Positionierung der Ziele im Vergleich zur *Roaming* Karte identisch ist. Daher konnte hier schon ein gewisses *Muscle Memory* (Gewöhnung der Armmuskulatur an die Bewegungsabläufe) einsetzen, was es dem Probanden ermöglichte, die Ziele sehr schnell und intuitiv zu orten, mit natürlichen Bewegungen zu „*Pre-Aim*“ und diese dann zu zerstören. Die Umgebung des Levels ist, wie im vorherigen Kapitel beschrieben „Realistisch“ gehalten und ist in dieser Gestaltungsart in vielen Spielen vergleichbar aufzufinden. Daher traten bei dem üblicherweise in realitätsnahen Maps agierenden Probanden während des Leveldurchlaufs keinerlei Probleme auf.

Dis-Map 1

Teilnehmer 1 hatte es schwerer, die erste Distraktor-Karte zu durchlaufen. Eine Ursache dafür war seine Desorientierung in den Gängen aufgrund der fehlenden Eckenbeleuchtung. Ebenfalls beeinflusste ihn die abweichende Platzierung eines Zieles (t4), wodurch es sich schwieriger gestaltete, dieses mit der bereits eingeübten Bewegung zu zerstören. Trotz dieser kleineren Besonderheiten im Vergleich zur *Normal* Map konnte Teilnehmer 1 eine gute Zeit von 41.266s für den Leveldurchlauf erreichen. Der Testende schien beim Zielen keinerlei Einschränkungen durch die auffällig gefärbten Objekte in seiner Umgebung verspürt zu haben und konnte sich ohne weiteres gut auf die Ziele fokussieren, nach Aussage des Probanden sogar besser als in der *Normal* Map.

Dis-Map 2

Hier hatte der Proband größere Probleme das Level zu beenden, Durchlauf 1 war daher nicht erfolgreich. So konnte der Teilnehmer durch die gewählte Laufroute das Ziel t5 nicht ausmachen, was durch die in dieser Map verwendeten *Shader* zustande kam, welche dafür sorgen, dass die Ziele vor transparenten Objekten ebenfalls transparent erscheinen. Außerdem ließ der Teilnehmer sich von den missdeutenden Zielmotiven auf Szenenobjekten ablenken, was jene zu den effizientesten Distraktoren machte (Abbildung 4.1). Des Weiteren sorgten die überladenen Farb-, Formen- und Musterreize in der Map bei dem Teilnehmer für Probleme, weiter entfernte Ziele zu zerstören, da jene teilweise durch die Umgebungsreize schwer auszumachen waren.

Anzumerken ist, dass der Proband sich entschloss, die Dis-Map2 als Optimierungslauf zu absolvieren. Bei den folgenden zusätzlichen Läufen lernte der Teilnehmer schnell die

Positionierung der einzelnen Zielscheiben und konnte auch beinahe unsichtbar erscheinende bzw. mit den visuellen Distraktoren nahezu verschmelzende Ziele (t5, t6, t10) fast problemlos zerstören. Ebenfalls bediente er sich einer *Power Position*, welches ein *Exploit* im Spiellevel ist, der es zulässt, vor dem Starten des *Timers* bereits 3 Ziele zu eliminieren. Aus Neugier und Fairness hat der Ersteller der Levels jene Position danach nicht gefixt, damit alle weiteren Probanden dieselben Möglichkeiten bekommen und die Testergebnisse in dieser Hinsicht entsprechend weniger verfälscht werden.

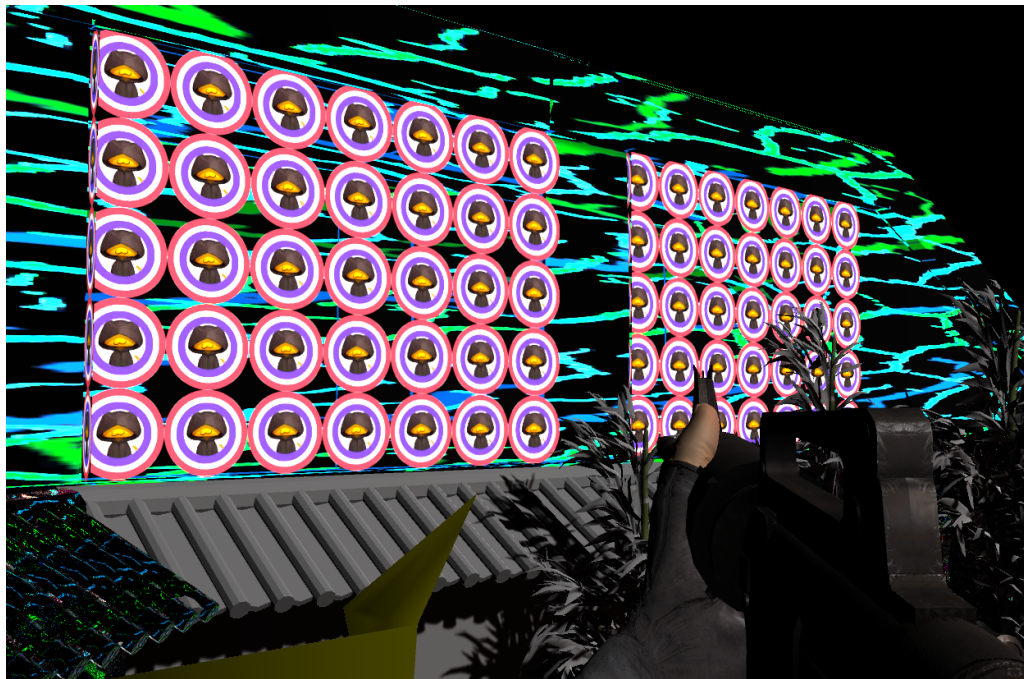


Abbildung 4.1.: Effizienter Distraktor in Dis-Map2

Nach einigen weiteren Versuchen war bereits eine immense Steigerung der Leistung zu erkennen. Durch das Antrainieren der Muskelbewegung und der Optimierung von *Ingame-Movement*, erreichte der Teilnehmer eine sehr schnelle Zeit von 29.500s für das Durchlaufen der Dis-Map2. Bereits der erste Wiederholungslauf im Training lag mit 34,148s mehr als 1,3s unter dem nahezu perfekten Erstversuch des Teilnehmers 1 in der *Normal Map* - dem eigentlich einfachsten Kurs. Die erfolgreiche Durchführung der Testdurchläufe lässt sich in gewisser Weise auf die großen Erfahrungen des Teilnehmers im *Action Shooter* Genre und damit einher kommenden Anpassungsfähigkeiten beschreiben. Ebenfalls sorgt dieses Vorwissen auch für eine durchaus bessere Reaktion gegenüber nahplatzierten Zielen, da es in *Action Shootern* oft nur verwinkelte Maps gibt, die nicht viele weitläufige Abschnitte in ihren Levels besitzen.

4.2.2. Teilnehmer 2 - erfahrener Taktik-Shooter-R6 Spieler

Dieser Test wurde im Rahmen einer Liveübertragung zum Entwickler durchgeführt und abgeschlossen.

Der zweite Probandentest wurde auf Wunsch des Probanden, in der Nacht vom 8. zum 9. April 2020 von 02:00-03:00 Uhr durchgeführt. Der Spieler hat sich verhältnismäßig kurz auf der *Roaming* Map aufgehalten, etwas mehr als 5min. Er hat die Ziele abgelaufen und sich entsprechend mit der Steuerung, der Anzahl verfügbarer Lebenspunkte und den Zielpositionen sowie deren Anzahl vertraut gemacht. Teilnehmer 2 hatte ebenfalls zunächst Probleme die vertikal platzierten Zielscheiben auf Anhieb zu finden und bediente sich kurz der beigelegten Orientierungshilfe.

Normal Map

Teilnehmer 2 hat es im ersten Versuch des normalen Anfangslevels nicht geschafft, alle Ziele zu finden und zu zerstören, ebenfalls ist dem Spieler hierbei die Munition ausgegangen, weshalb er keine weiteren Ziele eliminieren konnte, die sich noch vor ihm befanden. Der Versuch wurde in diesem Falle nicht gewertet. Weil der Erstversuch des Durchlaufens des Kurses misslang, bewegte sich der Proband im zweiten Testdurchlauf deutlich langsamer und aufmerksamer durch die Karte, und schaffte es somit die Zielbarriere nach Abschuss aller *Targets*, mit einer etwas langsameren Zeit von 52.400s zu durchlaufen. Da es in dieser Map keine bewusst platzierten Distraktoren gab, konnte nicht herausgefunden werden, ob bereits die *Normal* Map für den Teilnehmer Distraktoren unbekannter Art enthielt, die für das Ergebnis mitverantwortlich gewesen sein könnten.

Eher lässt sich vermuten, dass Teilnehmer 2 am Testbeginn zu wenig Zeit auf der Übungskarte verbracht hatte. Auch andere äußere Faktoren, wie z.B. die späte Uhrzeit können das Ergebnis beeinflusst haben.

Dis-Map 1

Die zweite Map hat Teilnehmer 2 im ersten Versuch erfolgreich durchlaufen, wenn auch hier mit einer recht moderaten Zeit von 1:05.281min. Diese kam zustande, da die Person mehrfach kurz die räumliche Orientierung in der Map verlor und sich in den Gängen um 180 gedreht hatte, was letztendlich auf die absichtlich fehlende Beleuchtung zurückgeführt werden kann. Hinzu kam, dass nach Aussage des Teilnehmers 2 die grellen und sehr präsenten Farben verschiedener Objekte auch einen spürbaren Einfluss auf die Navigation innerhalb der Karte hatten, da die Aufmerksamkeit kurzfristig auf diese Dinge geleitet wurde und der Bewegungsablauf dadurch weniger flüssig war. Trotz dieser minimalen Ablenkungen und Irritationen kam es jedoch nicht zu größeren Komplikationen beim Zielen und Treffen der *Targets*.

Dis-Map 2

Beim letzten der drei Testkurse, hatte der Proband diesen im ersten Durchlauf, nicht geschafft. Jenes war aber nicht auf seine eigene Zielleistung zurückzuführen, sondern auf die Anwendung *Transparent Shader*. Auch wenn nach Aussage des Probanden der Lauf bezüglich Steuerung und Präzision der beste war, spiegelte sich das nicht in einer Bestzeit wieder. Der Fehllauf basierte letztlich auf seiner Laufwegentscheidung durch das Level.

Beim allgemeinen Zweitdurchlauf aller Maps, war bei Teilnehmer 2 die beste Karte die Dis-Map2. Dies kam nach Aussage des Probanden durch die allgemeine Gewöhnung an die Steuerung und die Sensibilität des *FPS-Controllers* zustande. Die Ergebnisverbesserung war auf den Lerneffekt des Körpers (*Muscle Memory*, *Pre-Aim*) und daraus folgend schnellere Bewegungsabläufe in der Map zurückzuführen. Ab dem fünften Durchlauf erschloss sich diesem Spieler ebenfalls die *Power Position* von der aus er 3 Ziele bereits vor dem Start eliminieren konnte. Das führte dazu, dass der Proband mit Hilfe des *Exploits* eine Zeit von 35.782 schaffte, was zu einer massiven Verbesserung gegenüber seinem ersten erfolgreichen Lauf in der *Normal Map* von beinahe 17s führte.

Die im Vergleich zu anderen Testteilnehmern langsameren Level-Durchlaufzeiten ergaben sich vermutlich durch die sehr viel kürzere Aufenthaltszeit des Testteilnehmers in der *Roaming Map*. Auch erschien die Vorgehensweise jener Person, deutlich mehr durch die Taktik-*Shooter* Erfahrungen in R6 beeinflusst worden zu sein, da man in jenen Spielen oft eine längere *Down Phase* hat, als in anderen *Action Shooters*. Auch die Ablenkung scheint bei Taktik-*Shooter* Spielern besser zu wirken, da ihre Aufmerksamkeit auf die allgemeine Levelumgebung etwas stärker antrainiert ist. Daraus folgt, dass diese Spieler hinter jeder kleinen Änderung der Umgebung Gefahren vermuten oder Vorteile suchen, weshalb hier die offensichtlichen Distraktoren mehr Wirkung zeigen.

Dass dies bei der Dis-Map2 nicht zu stark erschien bei Teilnehmer 2, liegt nach Aussage des Teilnehmers selbst möglicherweise daran, dass er bewusst versucht hat, Nebenreize zu filtern und sich auf die tatsächlichen Ziele zu konzentrieren. In verschiedenen Studien konnte gezeigt werden, dass menschliche Sinnesorgane Reize lernbar filtern können [GB03],[BAMF12],[GDG98]. Das könnte Teilnehmer 2 geholfen haben, die Augen in den Testdurchläufen zu trainieren, damit die Wahrnehmung zu verbessern und dadurch die Ziele besser zu identifizieren. Diese Fähigkeiten sind bei jedem Menschen natürlich unterschiedlich stark ausgeprägt und somit schwer erfassbar. Hinzu kommt auch, dass mit jedem weiteren Durchlauf die Bewegung durch die räumlich gleichen Maps leichter fiel.

4.2.3. Teilnehmer 3 - erfahrener CS:S Spieler

Der Test mit Teilnehmer 3 wurde am 9. April 2020 um 23:00 Uhr durchgeführt, dabei befanden sich Teilnehmer und Entwickler in einem *VOIP*-Telefonat. Die gesamte Dauer des Tests betrug in diesem Falle etwa eine Stunde. Der Spieler verbrachte etwa 20 Minuten auf der *Roaming* Karte, um sich entsprechend mit dem Spiel vertraut zu machen. Auch bei Teilnehmer 3 bestätigte sich erneut, dass das Auffinden der vertikalen Ziele schwierig ist. Dennoch schaffte er es als erster Teilnehmer, diese ohne weitere Hilfe durch den Entwickler oder der beigelegten Orientierungs-Map zu finden und zu treffen.

Normal Map

Der Teilnehmer schaffte es nicht, beim Durchlaufen des ersten Kurses alle Ziele zu zerstören, da ihm seine Munition ausging. Hierzu führte die etwas überengagierte Herangehensweise des Spielers, mit der er die schnellste Durchlaufzeit erreichen wollte. Nach eigener Aussage und dem Mithören des Testers konnte geschlussfolgert werden, dass er viel zu unruhig versuchte, die Ziele zu treffen. Interessant war es aber zu betrachten, dass sich der Laufweg jener Person durch die Map von den Laufwegen der anderen Teilnehmer unterschied. Der Spieler entschied sich bewusst für einen etwas längeren Laufweg, um einige der Ziele besser treffen zu können. Im zweiten Durchlauf gelang es Teilnehmer 3, alle Ziele zu treffen und den ersten Kurs in einer Zeit von 41.511s zu absolvieren. Ebenfalls erschloss sich dem Spieler sofort die *Power Position* aus der heraus er ein Ziel vor dem eigentlichen Testbeginn eliminierte. Jedoch entschied sich der Spieler danach, diese Levelvereinfachung nicht in den weiteren Durchläufen zu nutzen, um realistischere Testergebnisse für die komplette Zielevernichtung in einem Durchlauf zu erzeugen.

Dis-Map 1

Das Level bereitete auch diesem Teilnehmer Probleme, weniger jedoch beim Zielen, als bei der Navigation, was erneut durch die fehlende Beleuchtung in den Seitengängen zustande kam. Der Proband war der Überzeugung, dass die Ziele auf dieser Map am besten zu sehen waren und damit entsprechend gut anzuvisieren und zu treffen. Demnach schaffte der Spieler es hier, die Karte beim ersten Durchlauf in einer Zeit von 46.244s zu absolvieren. Hinzu kommt auch, dass jener sich nach eigener Aussage hier bereits besser an die Steuerung gewöhnt hatte. Die farblichen Ablenkungen haben nach Aussage des Probanden auf der ersten Distraktoren-Karte nicht zu Irritationen geführt.

Dis-Map 2

Bei dieser Map kam es, zu dem Problem, dass eines der Ziele (t5) nicht aufzufinden war. Dies beruhte auf der Unaufmerksamkeit des Teilnehmers, welcher eine der vertikalen Zielpositionen bei seinem Durchlauf komplett ignoriert hatte.

Jenes war darauf zurückzuführen, dass nach Aussage des Teilnehmers, dieser von der Karte anfangs überfordert war, da sich zu viele Informationen (bzw. Distraktoren – Anmerkung des Entwicklers) gleichzeitig auf dem Bildschirm befanden. Der Teilnehmer war deshalb irritiert, abgelenkt und unaufmerksamer beim Absolvieren des Kurses. Nach Hinweis des Entwicklers, dass jenes Ziel, welches der Spieler vergaß zu zerstören, in der Map eingebaut ist und von seinem Laufweg aus eigentlich gut und trotz der Transparenzfilter auszumachen sei, schaffte es der Teilnehmer, die Map zu absolvieren. Dies allerdings in einer etwas langsameren Zeit von 51.460s. Die langsamere Zeit beruhte einerseits auf den zuvor gemachten Angaben des Probanden zur Map, aber zwangsläufig auch darauf, dass er diesmal den Durchlauf sehr konzentriert und bewusst und weniger intuitiv durchgeführt hatte, um das Ziel in jedem Fall zu entdecken und zu treffen.

Der dritte Teilnehmer, entschloss sich alle Maps erneut und mehrfach zu wiederholen, um seine anfänglich schlechteren Zeiten zu verbessern. Dabei hielt sich der Spieler jedoch am längsten auf der Distraktor-Map 2 auf und am kürzesten auf der Distraktor-Map 1. In allen Levels war eine deutlichere Verbesserung der Zeiten zu erkennen. Die finalen schnellsten Zeiten waren bei der *Normal* Map 37.942s (Optimierungslauf), auf der Dis-Map1 41.330s und auf der Dis-Map2 37.878s. Interessant war, dass auch hier der Proband letztendlich die schnellste Zeit auf der eigentlich konfusesten Map mit den meisten Distraktoren erzielte. Der Teilnehmer erwähnte dazu, dass die Platzierung der Ziele im dritten Zeitkurs als am einfachsten empfunden wurde.

Bezüglich seiner Durchläufe insgesamt, fasste der Spieler zusammen, dass ihm die *Normal* Map am angenehmsten erschien, und dass die zweite Karte (Dis-Map1) wegen der nicht vorhandenen Beleuchtung in den Gängen am unangenehmsten zu durchlaufen war. Die Ressourcenknappheit in Form der geringen Munitionsmenge beeinflusste seine Vorgehensweise immens und mit Abstand auch mehr als die visuellen Elemente. So sorgte diese beim Sprinten durch die Maps für sein zunächst ineffizienteres Abschießen von Zielen, weshalb er nach eigener Aussage dazu überging, im Gehen und im Stehen Ziele anzuvisieren und zu zerstören. Die Frage, ob die Beleuchtung der Level irgendeinen Einfluss auf seine Präzision oder Zielfindung hatte, beantwortete er so, dass die Belichtung der Karten ihn in keiner Weise beim Zielen unterstützt oder benachteiligt hätten, sondern nur in der Navigation geringe Beeinträchtigungen bei zu wenig Beleuchtung zu spüren waren.

Das Gesamtergebnis des Teilnehmers 3 war insofern unerwartet, da im Spiel CS:S, wie auch im Test, nicht die Möglichkeit besteht, die Ansicht der Zielvorrichtung zu wechseln, sondern in beiden nur mit Hilfe des Fadenkreuzes gezielt werden kann. Die Erwartung des Entwicklers war deshalb, dass der Spieler aufgrund seiner Vorkenntnisse die Testdurchläufe deutlich schneller absolvieren würde, als im Ergebnis zu sehen. Was jedoch Einfluss auf die Ergebnisse des Probanden gehabt haben kann, ist, dass in der Anwendung, in welcher

der Test durchgeführt wurde, nur eine Einzelschussfunktion vorhanden ist und der Spieler, nach eigener Aussage bevorzugt Schnellfeuerwaffen in CS:S nutzt. Damit setzt er im Spiel auf hohe Laufgeschwindigkeit und Streuzielen bzw. -treffen bei hohem Munitionsverbrauch – ein klarer Nachteil im Test. Hinzu kommt, dass in dem Spiel CS:S keine der vom Teilnehmer bevorzugten Maschinenpistolen neben ihrer Primärfunktion Dauerfeuer auch eine halbautomatische Schussfunktion für Einzelfeuer anbietet.

Dies war auch ein Nachteil gegenüber den anderen Teilnehmern, die basierend auf anderen Spielen und Spielweisen innerhalb des *Shooter* Genres bereits mehr Erfahrung mit der Einzelschussfunktion hatten.

4.2.4. Teilnehmer 4 – Large-Scale Shooter Spieler

Teilnehmer 4 hat den Test selbstständig ohne Supervision des Entwicklers, am 11. April 2020 in den Abendstunden durchgeführt. Nach Angabe des Probanden, benötigte dieser etwa eine Stunde, um den Test komplett abzuschließen. Er verbrachte etwa 15 Minuten auf der *Roaming* Map, um sich die einzelnen Zielpositionen einzuprägen. Der Spieler hatte keine Probleme beim Finden der Zielscheiben.

Leider hat Teilnehmer 4 die Instruktionen teilweise missinterpretiert, was dazu führte, dass die Zeiten für die Durchläufe mit Ausnahme des letzten nicht bis in den tausendstel Sekundenbereich angegeben wurden. Der Spieler vergaß bei seinen Durchläufen die *Finish*-Barriere zu durchlaufen, weshalb er nur mit Hilfe des Pausenmenüs die Zeiten gestoppt hatte, nachdem er alle Ziele eliminiert hatte. Hierdurch fehlte ein kurzes Stück Laufweg bis zum eigentlichen Ziel, welcher noch etwas Zeit in Anspruch genommen hätte. Dementsprechend wurde basierend auf der Abschätzung der ungefähren Position des Spielers am letzten vernichteten Ziel, die für das Ablaufen des restlichen Weges bis hin zur *Finish*-Barriere benötigte Zeit gemessen, was etwa 3 Sekunden ergab. Deshalb wurden die Testzeiten dieses Teilnehmers um 3 Sekunden erhöht, aber im tausendstel Bereich nicht betrachtet und auf ,000 gerundet dargestellt. Seine finalen Optimierungsläufe führte der Spieler im *VOIP*-Chat mit dem Entwickler am selbigen Tag nach einer kleinen Einübungsphase ab 22:00 Uhr aus.

Normal Map

Absolviert wurde die *Normal* Map von dem *Large-Scale Shooter* Spieler, ohne Probleme mit einer Zeit von 45.000s. Er merkte an, dass er seine Erstversuche auf jeder Map, entsprechend sicher gestaltet hat. Der Teilnehmer versuchte möglichst wenig zu Sprinten, um keines der Ziele während des Durchlaufs zu übersehen und wurde auch etwas langsamer,

wenn die Ziele in seinen Sichtbereich gelangten. Weiterhin hat der Spieler im Stehen geschossen, um bei bestem Blick auf das jeweilige Ziel sicher zu stellen, dass das Ziel auch zerstört wird. Dementsprechend hatte der Teilnehmer 4 keinerlei Probleme bei den Erstdurchläufen mit der Munition und Genauigkeit.

Dis-Map 1

Nach Aussage des Teilnehmers hatte dieser auf der Map, mit offensichtlichen Distraktoren Probleme. Diese lenkten nicht beim Zielen ab, erschwerten aber die korrekte Navigation durch die Karte. Insbesondere die Gänge, mit der fehlenden Beleuchtung, sorgten für Unbehagen, weshalb er hier besonders vorsichtig vorging. Gegensätzlich dazu war die Dis-Map1 bei ihm im Erstversuch die beste Map mit einer Zeit von 42.000s. Nach der persönlichen Meinung des Teilnehmers basiert die schnellere Durchlaufzeit teilweise auf „glücklichen“ Treffern. So identifizierte Teilnehmer 4 nach der Desorientierung im schwarzen Gang Ziel t7 mehr zufällig und traf dieses auch ebenso, nachdem er an jenem schon fast vorbeigelaufen war.

Dis-Map 2

Dieses Level schaffte Teilnehmer 4 im Erstversuch zu absolvieren. Ursache dafür war der von ihm gewählte bzw. bevorzugte Laufweg. Dieser entsprach dem vom Entwickler als optimal vorgesehenen Laufweg. So haben z.B. die Transparenz-*Shader* für Teilnehmer 4 die Darstellung der Ziele nicht erschwert, da er jene immer im richtigen Winkel betrachtet hat. Somit konnte der Spieler seine gewohnte Art des Durchlaufens der Map kontinuierlich durchführen und schaffte es, die Dis-Map2 mit einer soliden Zeit von 45.000s im Erstversuch zu durchlaufen.

Teilnehmer 4 entschied sich, alle Maps mehrfach zu durchlaufen, um bessere Zeiten zu erreichen und bevorzugte hierbei kein Level wesentlich mehr. Er durchlief alle Maps noch dreimal. Schon beim ersten Optimierungsversuch schaffte dieser, sich auf allen Karten um 3-6 Sekunden zu verbessern. Ebenfalls begann der Teilnehmer seine zuvor etwas passivere Spielweise ein wenig abzuändern, um die Kurse schneller zu absolvieren. Final waren seine Zeiten knapp 7 Sekunden besser, als bei seinen ersten Durchläufen. Zurückzuführen ist jenes auch darauf, dass der Spieler deutlich besser wurde, bei der Steuerung des *FPS-Controllers* und sicherer bezüglich der Zielpositionierung war.

Bei der Beantwortung der Fragen, gab Teilnehmer 4 an, dass er die Distraktor Map 2 bezüglich des Anvisierens der Ziele am besten fand. Hierzu meinte der Spieler, dass die fehlende Beleuchtung der Ziele, aber auch deren deutlich verringerte Sättigung und Helligkeit, dafür gesorgt haben, dass sie sich von der ansonsten grellen und chaotischen Farbgestaltung des Levels deutlich sichtbar unterschieden und abhoben. Dis-Map1 war der

für ihn unangenehmste Kurs, da er durch die Desorientierung in den Gängen das Gefühl bekam, weniger Einfluss und Kontrolle über den *FPS-Controller* zu haben und es dadurch als schwieriger empfand, durch die Gänge zu navigieren.

Des Weiteren erwähnte er auf die Frage des Einflusses der Munitionsknappheit, dass ihn jenes nicht wirklich beeinflusst habe in seinem Spielstil. Er sei dennoch etwas vorsichtiger beim Zielen gewesen, was für ihn aber nicht untypisch erschien, da sich jener Spieler sowieso in den meisten Spielen recht passiv verhält und sich entsprechend mehr Zeit nimmt beim Zielen. Auch haben ihn die visuellen Elemente der Distraktoren-Karten nicht weiter beeinflusst.

Auf die letzte Frage zur Beleuchtung der Zielscheiben und der allgemeinen Lichtgestaltung der Level entgegnete Teilnehmer 4, dass es an einigen Stellen hilfreich war, die Ziele auszumachen, aufgrund ihrer Hervorhebung durch Eigenlicht. Ansonsten hatte der Faktor Licht an sich keinen großen Einfluss auf sein Vorgehen, mit Ausnahme der fehlenden Gangbeleuchtung in Dis-Map1.

Grundlegend kann man zum Probanden Nummer 4 sagen, dass die visuellen distraktori-schen Elemente keine gravierenden Auswirkungen auf seine Spielweise hatten. Dies könnte damit erklärt werden, dass er als Spieler verschiedener *Large-Scale Shooter Games* darin geübt ist, die in Maps vorhandenen vielfältigen Effekte sehr gut zu filtern. Die möglichst gleichzeitig zu erfassenden Informationen auf den oft sehr großen Maps, wie sie zum Beispiel in "Battlefield" zu finden sind, sowie deren Interaktionsvielfalt, mit z.B. Fahrzeugen in der Horizontalen oder Helikoptern und Jets in der Vertikalen, stellen vermutlich ein exzellentes Training im Hinblick auf die Ausblendung von Distraktoren und die allgemeine Situation auf der Map dar. Die Konzentration des Spielers bleibt fokussiert auf kleinere bzw. konkrete und akut wichtigere Aufgaben innerhalb der Level.

Ergänzend hinzuzufügen ist, dass auch die bekannte passive Herangehensweise des Teilnehmers in verschiedenen Spielesituationen und die damit verbundene Geduld, sicherlich einen wirksamen Faktor beim Fokussieren auf das Wesentliche im Spielelevel darstellte.

4.2.5. Teilnehmer 5 - Entwickler und Casual Spieler

Der Ersteller des Tests entschloss sich jenen auch selbst durchzuführen. Damit geht ein Proband mehr in diese initialen Betrachtungen zum Einfluss von visuellen Distraktoren in die Testauswertung ein. Dies hilft in Anbetracht der kleinen Gesamtanzahl an Testteilnehmern, die Datenlage etwas zu verbessern. Gleichzeitig fließen aber auch Daten ein, die nur bedingt mit denen der anderen Teilnehmer vergleichbar sind. Der Entwickler ist zwangsläufig gegenüber anderen Teilnehmern im Vorteil, da er sämtliche Fallstricke der

einzelnen Maps kennt. Trotzdem, erscheint die Erhebung der Daten nützlich, da Teilnehmer 5 zu dem Teil der Bevölkerung gehört, der eine Rot-Grün-Farbsehschwäche ererbt hat und somit auch in dieser Hinsicht interessante Daten erhoben werden könnten.

Als Entwickler der Anwendung hat Teilnehmer 5 entsprechend mehrere Stunden auf der *Roaming* Map verbracht, vor allem für das *Play Testing* und Konstruieren. Ebenfalls ist ihm die Platzierung aller Zielscheiben von Grund auf bekannt. Der Selbsttest wurde am 6. April 2020 um 17:00–18:30 Uhr offiziell durchgeführt. Während der Testphase verbrachte der Entwickler wegen der korrekten Einhaltung der Testvorgaben etwa 5 Minuten auf der *Roaming* Map. Dem Spieler war die *Power Position* bekannt, er beschloss jedoch, sich dieser bei den Erstdurchläufen nicht zu bedienen. Obwohl es sich bei den Durchläufen der Testkarten zwangsläufig nicht um Erstversuche handelt, werden die dabei erhobenen Daten in die Ergebnisbetrachtung als solche einbezogen.

Normal Map

Die erste Map wurde von dem Teilnehmer 5 ohne Probleme in der Orientierung oder Zielfindung innerhalb von 37.746s absolviert. Die Zeit war dennoch mehr als 2s langsamer, als die des Teilnehmers 1 mit 35.500s. Die Gründe dafür, dass der Entwickler schlechter abschnitt als Teilnehmer 1, könnten in den schlechteren Zielfähigkeiten des Entwicklers oder auch an der Laufwegentscheidung – Nutzung des potentiell optimalen Weges – gelegen haben. Hinzu kommt, dass ein *Casual* Spieler kein professioneller *Gamer* ist. Im Vergleich zu allen anderen Testteilnehmern ist die Zeit von Teilnehmer 5 in den Erstdurchläufen deutlich besser gewesen. Auffinden, Anpeilung und Abschuss der Ziele wurden hauptsächlich im Laufen und per *Pre-Aiming* durchgeführt. Bei zwei Zielen musste der Teilnehmer jedoch stoppen, da er diese verfehlte und sich dementsprechend für einen erneuten Schuss repositionieren musste.

Dis-Map 1

Die Zeit von 34.678s, die der Teilnehmer in diesem Level beim Erstversuch erreichte, kann man als den beinahe optimalen Erstversuch bezeichnen, bei dem kein *Exploit* genutzt wurde. Die Distraktoren auf der Karte haben den Spieler in keiner Weise beeinträchtigt, da durch das Vorwissen über deren Platzierung und das *Play Testing* der Fokus auf die Ziele zu ausgeprägt war. Die minimalistische Gestaltung der Distraktor-Karte half dem Spieler zudem entsprechende Distraktoren auszublenden. Die fehlende Beleuchtung in den Gängen beeinflusste den Teilnehmer 5 nicht. Er hatte die Umgebung der Karten oft genug durchlaufen, deshalb war ihm die Navigation in den Gängen bekannt und auch die *FPS-Controller*-Steuerung erfolgte bereits weitgehend über *Muscle Memory*. Trotzdem war nach Aussage des Teilnehmers auch ein gewisser Anteil Glück dabei.

Dis-Map 2

Jene Map bereitete dem Teilnehmer trotz Vorwissen die meisten Schwierigkeiten. Diese bestanden dabei nicht in der Ortung der Zielscheiben oder der Navigation, sondern in der Beeinträchtigung durch die vielen farbigen Distraktoren und dem Ausmachen der eigentlich bekannten Targets. Deshalb hat Teilnehmer 5 im Erstversuch eine moderate Zeit von 39.313s benötigt.

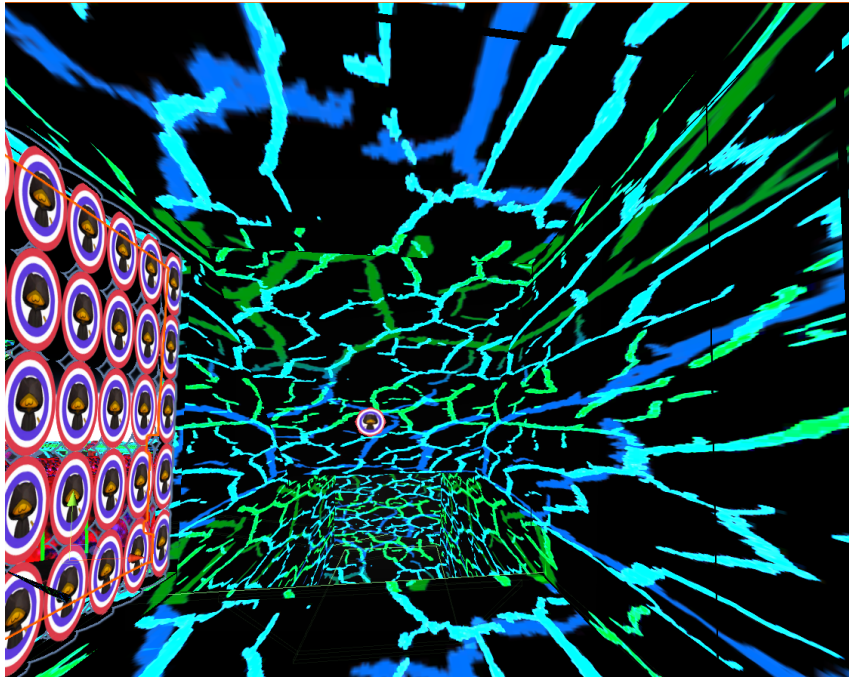


Abbildung 4.2.: Zieldarstellung Dis-Map2 (t2).

Teilnehmer 5 empfand das Chaos von Farben, sowie die sehr schwache Darstellung der Ziele auf der mehrfarbigen und mit vielfältigen abstrakten Formen sowie surrealen Hintergründen versehenen Map, als sehr problematisch bei der Zielauffindung. Besonders schwierig empfand Teilnehmer 5 das Auffinden von Zielen und das Zielen selbst in Gängen, sowie auch das Zielen aus weiter Entfernung, da sich die Ziele für ihn nicht sofort vom Hintergrund abhoben (siehe Abbildung 4.2). Die blau-weiß-rote Färbung der Ziele schien an einigen Stellen mit dem Hintergrund zu verschmelzen. Inwieweit die Farbsehschwäche des Probanden hineinspielt, kann hier nur vermutet werden.

Für den Optimierungslauf beschloss Teilnehmer 5, die Dis-Map2 zu durchlaufen. Trotz der Schwierigkeiten im Erstversuch gelang es dem Teilnehmer seine Zeit von 39.313s auf 35.695s zu verbessern. Anschließend machte der Spieler noch einen Durchlauf mit Hilfe der *Power Position* und verbesserte sich in seinem letzten Lauf auf eine Zeit von 33.708s.

Insgesamt war Teilnehmer 5 damit nach Teilnehmer 1 zweitschnellster Proband. Hauptursache für die etwas langsameren Zeiten von Teilnehmer 5 im Vergleich zu Teilnehmer 1, ist aus Sicht des Entwicklers, dass Teilnehmer 5 sich mehr Zeit für das Zielen nehmen musste. Grundsätzlich zielt er etwas langsamer, konnte aber auch gleichzeitig Ziele, insbesondere auf der Dis-Map2 möglicherweise nicht so schnell ausmachen, wie andere Teilnehmer. Interessant in diesem Zusammenhang ist dazu noch einmal die Aussage von Proband 4, für den es sogar leichter war, die *Targets* auf der Dis-Map2 zu orten. Da die verwendeten farblichen Distraktoren unter anderem auf die drei RGB-Grundfarben setzen, kann hier zumindest vermutet werden, dass die Rot-Grün-Farbsehschwäche die Zielgeschwindigkeit von Teilnehmer 5 mit beeinflusst haben könnte.

4.3. Gesamtauswertung des Präzisionstests

4.3.1. Virtuelles Präzisionstraining - Datenanalyse

Der folgende Abschnitt fasst die Ergebnisse des Tests und der detaillierten Evaluation zusammen und versucht, basierend auf einem Vergleich der Einzeldaten, allgemeine Schlussfolgerungen für das gewählte Testsystem und die eingesetzten Distraktoren sowie deren Wirkung zu ziehen.

Wesentlich für den Testerfolg war es, dass die Testteilnehmer ähnliche Voraussetzungen für einen FPS-Test mitbringen und somit ähnliche, vergleichbare Testergebnisse erwarten ließen. Dies konnte mit den fünf teilnehmenden Probanden nahezu gewährleistet werden, wenngleich die insgesamt noch zu kleine Anzahl der Teilnehmer nur vorläufige Ergebnisse liefern kann. Die Spieler gehörten zur Altersgruppe 20-30 Jahre, waren alle männlich und mit verschiedenen Spielen aus dem *Shooter* Genre und *3D-First-Person*-Anwendungen vertraut. Die Steuerung, die bei den meisten PC- oder Mac-Spielen mit den WASD-Tasten funktioniert sowie die Kamerakontrolle mit der Maus war ihnen intuitiv geläufig und antrainiert.

Die gute Zusammensetzung der Teilnehmergruppe wird daran erkennenbar, dass einer Person, ohne Vorerfahrung im *Shooters* Genre hatte, ebenfalls ermöglicht wurde, den Präzisionstest durchzuführen. Diese Person benötigte nach ausreichender Übungszeit in der *Roaming* Map mehr als 6min für nur einen Durchlauf in der *Normal* Map und wurde danach vom weiteren Test ausgeschlossen. Alle betrachteten Durchlaufzeiten der Testteilnehmer bewegten sich im Vergleich dazu in einem Bereich von 29,5s bis ca. 1:05min (Tabelle 1). Vorerfahrungen im *FPS-Gaming* sind demnach ein essenzielles Kriterium für die Testteilnahme unter den gewählten Bedingungen.

Die Tests wurden innerhalb des Zeitraums vom 5. April bis zum 11. April 2020 mit den einzelnen Probanden durchgeführt. Es war vor Testbeginn allen Teilnehmern erklärt worden, wie der Test absolviert werden sollte und inwiefern die einzelnen Maps gestaltet wurden. Jeder Proband erhielt eine Testanleitung und ein Fragenprotokoll. Die Teilnehmer 1, 2 und 3 erhielten während des Tests *VOIP*-Unterstützung durch den Entwickler. Proband 4 führte den Test alleine durch, um die Plausibilität der Testanleitung zu testen. Proband 5 war der Entwickler selbst. Wie die Evaluation zeigt (Abschnitt 4.2.4), ist es für derartige Tests sinnvoll, für ausnahmslos alle Testteilnehmer eine Supervision bereitzuhalten.

Da alle Testteilnehmer aktive Spieler von *FPS-Games* sind, wurde erwartet, dass sich die Durchlaufzeiten zwischen den drei Test-Maps, insbesondere bei den Optimierungsläufen nur wenig unterscheiden werden. Unterschiede wurden trotzdem erwartet, weil nicht alle Spieler die gleichen koordinativen Fähigkeiten vorweisen können und sich spieltechnisch in verschiedenen *Shooter* Genres bewegen.

Ein leicht zu erfassender Aufbau der Testkurse sollte diese Unterschiede möglichst ausgleichen. Was nicht vorhergesehen oder beeinflusst werden konnte, war, wie die Spieler auf die Distraktoren reagieren. Deshalb sollten die Zeiten der einzelnen Teilnehmer innerhalb der Verbesserung ihrer Läufe betrachtet werden. Von diesem Prinzip musste für die Auswertung der Durchlaufzeiten teilweise abgewichen werden, da nicht alle Probanden die Testlevels im Erstdurchlauf erfolgreich abschlossen. Für die Auswertung von Zeiten für die Durchläufe wurden deshalb die Zweiddurchläufe mitberücksichtigt, sofern sie den ersten erfolgreichen Durchlauf des Probanden darstellten. Die restlichen Zeiten der Zweiddurchläufe sind in Tabelle 1 trotzdem mit gelistet. Sie wurden für die allgemeine Datenauswertung aber nicht weiter berücksichtigt.

Teilnehmer 1 fand während des Tests eine *Power Position* mit der maximal drei Ziele vor dem Start getroffen werden können. Um Chancengleichheit zu bewahren zwischen den Teilnehmern, wurde dieses Feature nicht entfernt, aber den anderen Probanden auch nicht mitgeteilt. Da die *Power Position* nicht von allen Probanden gefunden wurde, von anderen aber insbesondere bei den Optimierungsläufen zur Anwendung kam, muss geschlussfolgert werden, dass dies für die Testauswertung suboptimal gewesen ist, vor allem in Bezug auf die zeitliche Betrachtung der Durchläufe. Dies müsste in zukünftigen Tests ausgeschlossen werden. Tabelle 1 stellt eine Übersicht der wesentlichen Testergebnisse aller Teilnehmer dar. Sie folgt in ihrer Chronologie der Reihenfolge der Teilnehmer am Test und der der Test-Maps und vergleicht ebenso die relevanten Antworten der Teilnehmer auf die Testfragen allgemein. Ausgewählte Daten der Tabelle werden im Folgenden näher diskutiert.

4.3. GESAMTAUSWERTUNG DES PRÄZISIONSTESTS

Auswertung virtuelles Präzisionstraining					
Probanden Spielertyp	T1 cod4:mw	T2 R6:S	T3 CS:S	T4* LSS bsp BF	T5** Casual FSP (Entwickler)
Roaming					
Aufenthalt ca.	45:00:000	05:00:000	20:00:000	15:00:000	n.d.
Normal Map					
Durchlauf 1	00:35:500	negativ	negativ	00:45:000	00:37:746
Zweitdurchlauf	n/a	00:52:400	00:41:511 (PP1)	00:39:000	00:36:556
Weitere Durchläufe	n/a	00:35:782 (PP3)	00:37:942	00:38:301	00:35:468
Distraktorwirkung?	nein	nein	nein	nein	nein
Dis-Map1					
Durchlauf 1	00:41:266	01:05:281	00:46:244	00:42:000	00:34:678
Zweitdurchlauf	n/a	00:48:149	00:42:795	00:39:000	n/a
Weitere Durchläufe	n/a	n/a	00:41:330	00:37:549	00:39:113
Distraktorwirkung?	ja (Navigation)	ja (Navigation)	ja (Navigation)	ja (Navigation)	nein
Dis-Map2					
Durchlauf 1	negativ	negativ	negativ	00:45:000	00:39:313
Zweitdurchlauf	00:34:148 (PP3)	00:55:517	00:51:460	00:41:000	00:35:695
Weitere Durchläufe	00:29:500 (PP3)	00:43:583	00:37:878	00:38:433	00:33:708 (PP3)
Distraktorwirkung?	ja (Zielen)	ja (Navigation)	ja (Navigation, Zielen)	nein	ja (Zielen)
Generelle Testinformationen					
Gewählter Laufweg für alle Maps	links h2	links h2	rechts h2	h2 Treppe	h2 Treppe
Beliebteste Map	Normal Map	Normal Map	Normal Map	Dis-Map2	Dis-Map1
Unbeliebteste Map	Dis-Map1	Dis-Map1	Dis-Map1	Dis-Map1	Dis-Map2
Schwer zu findende Ziele					
Generell	7, 8 vertikal	7, 8 vertikal	nein	nein	nein
Map-spezifisch	5 auf Dis-Map2	5 auf Dis-Map2	5 auf Dis-Map2	nein	nein
Munitionsverbrauch beim OL	gering	mittel	viel	gering	viel

Zeitangabe in Minuten:Sekunden:Millisekunden

T = Teilnehmer; * Teilnehmer führte den Test allein durch; ** Teilnehmer hat Rot-Grün-Sehschwäche

negativ = Durchlauf nicht erfolgreich, Testteilnehmer hat nicht alle Ziele vernichtet

Lauf blau markiert = Optimierungslauf (OL); h2 = Haus 2;

PPn= PowerPosition angewendet auf n Ziele (maximal 3)

Munitionsverbrauch beim OL: gering = 20-23; mittel = 24-26; viel = 27-30

Tabelle 1 zeigt zunächst, dass die Zeit, welche die Testteilnehmer in der *Roaming* Map verbracht haben, sehr unterschiedlich ist. Dies hatte erheblichen Einfluss auf die Testergebnisse, insbesondere für den im Test wichtigsten Durchlauf 1 (Abb. 4.3). Werden die Zeiten für die ersten erfolgreichen Durchläufe in allen drei Test-Maps verglichen, ergibt sich für den Teilnehmer 1, der sich am längsten in der Übungskarte aufgehalten hat, die beste Gesamtzeit. Teilnehmer 2 hingegen hielt sich in der *Roaming* Map im Vergleich dazu nur sehr kurz auf. Entsprechend war die Summe seiner Durchlaufzeiten mit fast 1min Differenz zu Teilnehmer 1 die schlechteste aller Teilnehmer (Abb. 4.3). Die Zeiten der anderen Teilnehmer weichen nicht so stark voneinander ab, wenngleich sich trotzdem deutlich zeigt, dass Teilnehmer 3 und 4 mit mittleren Aufenthaltszeiten in der *Roaming* Map auch schlechter im Test abschnitten, als Teilnehmer 1 und der Entwickler (T5). Eine Standardisierung der *Roaming*-Zeit von ca. 20-30min für alle Teilnehmer hätte vermutlich zu einer weiteren Annäherung der Durchlaufzeiten geführt.

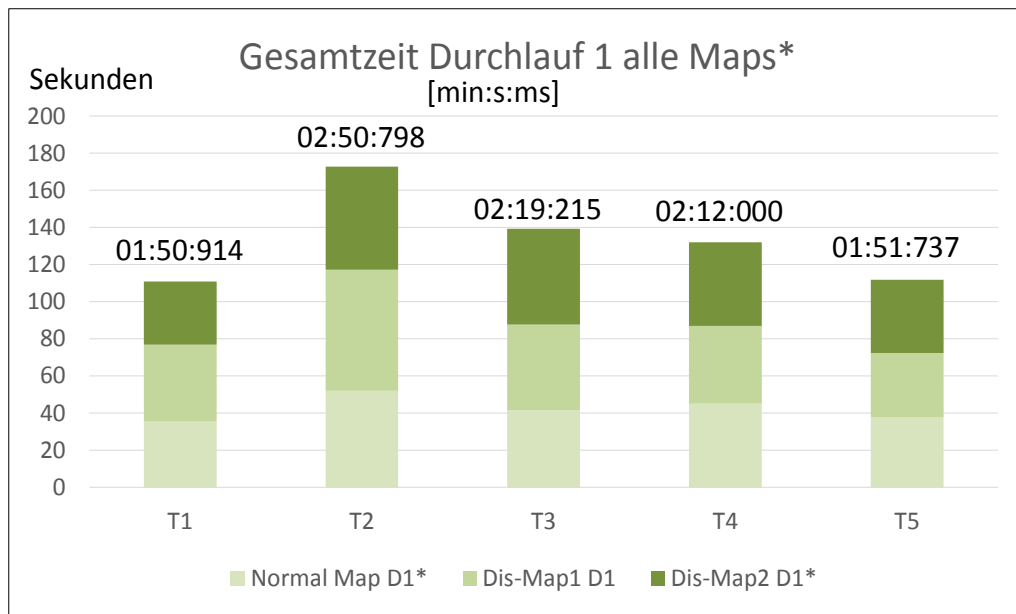


Abbildung 4.3.: Test-Durchlauf 1 – Summe der Durchlaufzeiten aller Maps. *Teilweise Daten von D2.

Da es im Erstdurchlauf aller Karten fünf Fehlversuche gab, rutschten an diesen Stellen die Zweitdurchlaufergebnisse an die entsprechenden Positionen. Eine alleinige Zweitdurchlaufauswertung aller Probanden wäre alternativ auch nicht ohne Abstriche möglich gewesen, weil der Zweitdurchlauf aller Maps nicht Pflicht war und die Probanden 1 und 5 diese auch nicht immer durchgeführt haben. Trotz dieser Abstriche zeigen die vorliegenden Durchlaufdaten in Abb. 4.4, dass alle Probanden aufgrund der Präzisionsübung mehr Zeit für die Testläufe benötigten, als für den Kontrolllauf ohne Zielübung benötigt wurde.

Gleichzeitig wird ersichtlich, dass Teilnehmer 1 in seinem Optimierungslauf sehr dicht an die Kontrollzeit herankam. Aus Abb. 4.4 wird ebenso deutlich, dass außer bei Teilnehmer 4, eine der beiden oder beide Distraktoren-Maps zur Erhöhung der Testzeiten im Vergleich zur *Normal Map* führten.

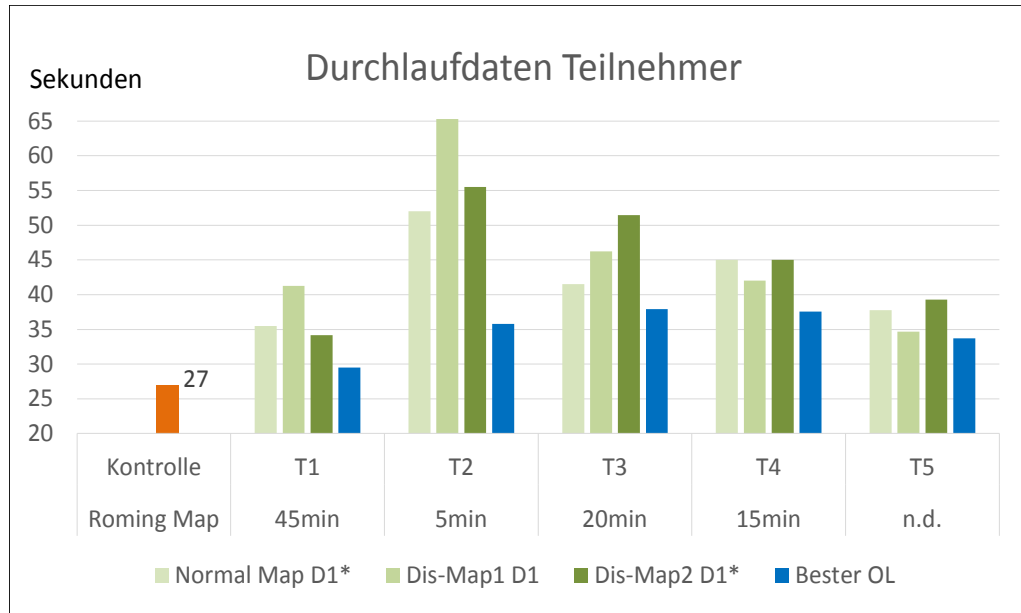


Abbildung 4.4.: Durchlaufdaten Teilnehmer: Vergleich der Zeiten in Durchlauf 1 und Optimierungslauf. *Teilweise Daten von D2.

Weiterhin kann aus Tabelle 1 und Abb. 4.4 abgeleitet werden, dass alle Teilnehmer nach Abschluss der Test-Durchläufe und nach mehrfachem weiterem Üben in ihren finalen Optimierungsläufen zum Teil erhebliche zeitliche Verbesserungen erreichten. Diese erzielten die Probanden unabhängig von der Art der dafür ausgewählten Map. Daraus lässt sich trotz geringer Datenlage ableiten, dass es direkte Trainingseffekte in Bezug auf die Hand-Augen-Koordination bei allen Teilnehmern gab. Deutlich wird auch, dass geübte FPS-Spieler teilweise bereits nach nur kurzem Üben eine Testsituation, wie hier vorgegeben, intuitiv durchlaufen aufgrund vieler Vorerfahrungen in entsprechenden Spielen. Die Distraktoren haben dennoch gewirkt, wenn auch nur für eine kurze Zeit. Für die Erstellung von neuen Anwendungen kann dies bedeuten, dass es wichtig ist, eine ausgewogene Balance zwischen einer Vielzahl unterschiedlichster Distraktoren zu finden.

Ein nützlicher Wert für zukünftige Tests zum Nachweis des Einflusses der visuellen Distraktoren ergibt sich auch aus der ganzheitlichen Betrachtung der zeitlichen Anteile aller Probanden in den jeweiligen Distraktor-Karten (Abb. 4.5). Trotz verschiedenster Empfindungen der Probanden innerhalb einer bzw. für oder gegen eine der Maps und auch trotz

weiterer Beeinflussungen der Ergebnisse durch Laufweg, Munition, Ziel- oder Navigationsprobleme etc. der Testteilnehmer zeigt der Vergleich der Durchlaufzeitenanteile bezogen auf die betrachteten Zeiten des 1. Durchlaufs (Tabelle 1), dass sich die Testteilnehmer mit jeweils 34% der gemittelten Zeiträume in Dis-Map1 und Dis-Map2 länger als in der Normal Map (32% der Zeit) aufhielten (Abb.4.5).

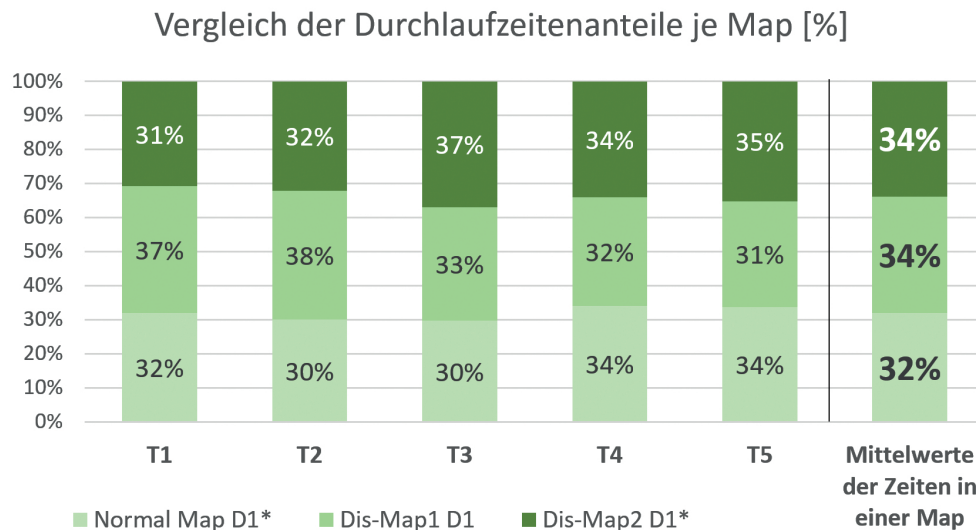


Abbildung 4.5.: Vergleich der Durchlaufzeitenanteile je Map: Prozentualer Vergleich der benötigten Zeiten der Teilnehmer für den ersten Durchlauf und Mittelwerte der verbrachten Zeit in einer Map. *Teilweise Daten von D2.

Auch wenn diese Zeitdifferenz relativ gering erscheint, lassen sich daraus zwei Dinge schließen. Erstens, die Distraktoren hatten einen nachweislichen Effekt auf die Wahrnehmung und Hand-Augen-Koordination der Teilnehmer, auch wenn dieser in Zeit ausgedrückt, gering ist, und die Einzelwerte der Probanden dazu schwanken. Dabei ist jedoch zu berücksichtigen, dass die Distraktor-Karten aufgrund identischer Kartengrundstruktur und dadurch möglicher schneller Trainingseffekte der FPS-erfahrenen Probanden eigentlich leichter bzw. schneller hätten durchlaufen werden können, als die jeweils davor durchlaufene Map. Und zweitens, der Grundgedanke der Arbeit, Einflüsse der visuellen Distraktoren aufzuzeigen, war mit dem erstellten Testsystem und den ausgewählten Probanden erfolgreich. Trotzdem bleiben Wiederholungen dieses Tests für eine valide Aussage zwingend.

4.3.2. Auswertung und Diskussion des Fragebogens

Die Antworten zu den an die Probanden gestellten Fragen sind in Tabelle 1 (Abschnitt 4.3.1) bereits zusammengefasst dargestellt. Die folgende Analyse bewertet einige ausgewählte Antworten genauer.

Aus Tabelle 1 geht hervor, dass die Teilnehmer unterschiedliche Laufwege für die Testläufe genommen haben. Diese sind in Abbildung 4.6. skizziert. Die gewählten Laufwege hatten sich nach dem Üben in der *Roaming*-Karte für die Probanden als optimal erwiesen und wurden dann für die Testläufe grundsätzlich fixiert.

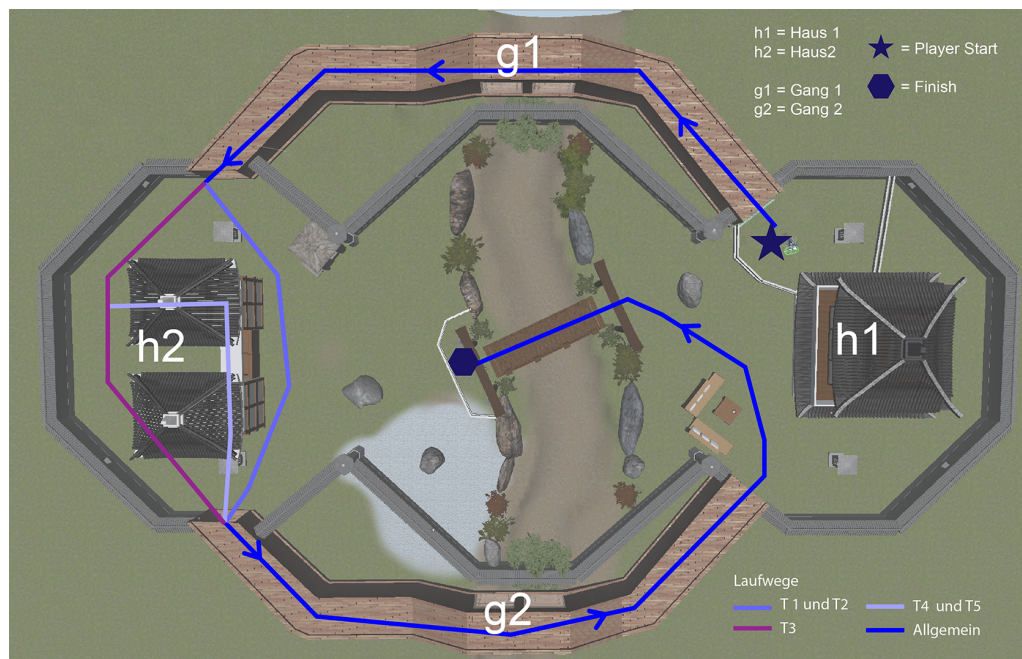


Abbildung 4.6.: Laufwege der Testteilnehmer (T) im Präzisionstest

Wie aus Abb. 4.6. hervorgeht, wichen die Laufwege nach dem Austritt aus dem 1. Gang ab. Haus Nr.2 wurde von dort aus gesehen für die Zielfindung links (T1, T2) oder rechts (T3) umgangen oder die Treppe im Haus (T4, T5) benutzt. Über alle drei Wege gelangen den Probanden erfolgreiche Durchläufe, wenngleich der Laufweg links und rechts am Haus Nr. 2 vorbei die Zielfindung (*Target t5*) in Dis-Map2 zumindest erschwert. Die erreichten Durchlaufzeiten insgesamt lassen sich jedoch nicht mit den verschiedenen Laufwegen in eine sinnvolle Beziehung setzen.

Die Frage nach der beliebtesten bzw. unbeliebtesten Map ergab, dass die Probanden T1, T2, und T3 die *Normal* Map am angenehmsten empfanden. Dieselben Probanden bewer-

teten die Dis-Map1 gemeinsam mit Proband T4 als am unangenehmsten zu durchlaufen. Die Navigation aller vier Probanden war hier durch die fehlende Beleuchtung behindert, dies störte aber niemanden bei der Zielerfassung. Das ist im vorliegenden Probandenpanel ein relativ starkes Votum für realitätsnahe Leveldarstellungen (= angenehm bzw. beliebt) ohne zusätzliche visuelle Distraktoren und ein noch stärkeres Votum gegen ein Leveldesign mit starken Helligkeits- und RGB-Distraktoren (=unangenehm bzw. unbeliebt). Im Zusammenhang mit der Hell-Dunkel und RGB-Wahrnehmung des menschlichen Auges sind die Einschätzungen der Probanden T4 und T5 interessant. T5 bevorzugt als einziger Teilnehmer Dis-Map1 und kam am wenigsten von allen Teilnehmern mit Dis-Map2 zurecht. Teilnehmer T4 hingegen bevorzugte Dis-Map2 als einziger Proband. Es kann vermutet werden, dass die Rot-Grün-Sehschwäche von Proband T5 den Teilnehmer Maps mit klareren Inhalten bzw. weniger Details grundsätzlich bevorzugen lässt. Und dies unabhängig von den starken RGB-Farbdistraktoren in der Dis-Map1. Immerhin erreichte T5 in Dis-Map1 die mit Abstand schnellste Durchlaufzeit (6-30s schneller als die anderen Teilnehmer und die insgesamt zweitschnellste Durchlaufzeit im Test). Teilnehmer 4 hat im Gegensatz dazu möglicherweise sehr gute Fähigkeiten, geringe Helligkeitsunterschiede zu erkennen, so dass die Zieleerkennung in Dis-Map2 für ihn leicht war und die Distraktorwirkung sehr gering (siehe dazu auch Abschnitt 4.2.4).

Befragt wurden die Probanden auch nach schwer zu findenden Zielen. Um Hinweise auf einen möglichen Distraktoreinfluss zu erhalten, war diese Frage unterteilt in generelle und levelspezifische Schwierigkeiten bei der Ziellokalisation. Die meisten Ziele konnten von allen Probanden während der Durchläufe jedoch gefunden und abgearbeitet werden. Für Teilnehmer 1 und 2 bereiteten die Ziele t7 und das vertikale *Target* t8 Schwierigkeiten bei der Lokalisation während des Durch“rennens“ der verschiedenen Maps. Den Teilnehmern T1, T2 und T3 bereitete die schnelle Identifizierung von *Target* t5 auf Dis-Map2 Schwierigkeiten. Wie oben erwähnt, wurden von diesen Teilnehmern abweichend von dem vom Entwickler vorgesehenen optimalen Laufweg andere Laufwege gewählt, weshalb die Ziele nur seitlich zu sehen waren oder auf diesen Laufwegpositionen von *Transparent Shadern* teilweise getarnt wurden. Werden die Aussagen zur Distraktorwirkung in Dis-Map2 hierzu in Beziehung gesetzt, zeigt sich, dass die Probanden T1 und T3 auch Probleme beim Zielen hatten. Die mit farblichen Distraktoren überladene Dis-Map2 könnte diese Teilnehmer demnach tatsächlich irritiert bzw. abgelenkt haben.

Betrachtet wurde zuletzt indirekt auch der Munitionsverbrauch der Teilnehmer. Indirekt deshalb, weil der Entwickler davon ausging, dass das Vorwissen der Probanden aus FPS-Anwendungen und die für den Test programmierte Munition von 30 Schuss bei 10 Zielen und 2 Treffern für den Abschuss eines Zieles gut miteinander korrelieren. Dies war insgesamt in den Durchläufen bei allen Teilnehmern der Fall. Der Munitionsverbrauch war

jedoch sehr variabel, wurde aber dennoch nicht zahlentechnisch näher erfasst. Lediglich für den jeweils besten Optimierungslauf der Teilnehmer fand ein Munitionsvergleich statt. Die Idee dahinter war, zu erfahren, ob die Teilnehmer nach hinreichender Übung mit einem geringen Munitionsverbrauch bei Bestzeit auskommen. Dies war nicht der Fall (siehe Tabelle 1). Eine Korrelation zu Distraktoren konnte ebenfalls nicht bestimmt werden.

4.3.3. Bewertung der Grundstruktur - Leveldesign

Für die Entwicklung des Testsystems zur Untersuchung visueller Distraktoren und deren Einfluss auf Probanden war es unter anderem bereits im Vorfeld der Entwicklung wichtig, Testanwendung und Testteilnehmerfeld gedanklich sinnvoll aufeinander abzustimmen. Im Ergebnis dieser Analyse wurde die Testumgebung von vorn herein für FPS-Spieler entwickelt und der Versuch unternommen, Ausgewogenheit zwischen theoretisch möglicher schneller Fortbewegung der Spieler in den Testlevels und Spannungseffekten durch plötzlich auftauchende *Assets*, Distraktoren und Zielen zu erreichen. Wie die Ergebnisse in den Abschnitten 4.2. und 4.3.2. zeigen, wurde die Testumgebung grundsätzlich positiv von den FPS-Probanden angenommen und sie erwies sich sowohl als unabhängige „Spiele“-Anwendung, aber vor allem auch als Testsystem geeignet.

Basierend auf den vorliegenden initialen Testergebnissen und den individuellen Einschätzungen der befragten Probanden, kann jetzt eine Nachjustierung in den Levels erfolgen, um den Test vor einem breiteren Einsatz mit mehr Teilnehmern potentiell aussagekräftiger zu gestalten.

Für den hier vorgestellten Test waren die wichtigsten Unterschiede zwischen den Levels die folgenden:

Als *Recap*:

- Die **Normal Map** hat keine bewusst gesetzten Distraktoren und sollte somit keinerlei Einfluss auf die Spieler beim Zielen haben.
- Die **Dis-Map1** ist minimalistisch gehalten und besitzt offensichtliche Distraktoren (Rot-Grün-Blau-Hauptfarben des Sehens) sowie eine andere Beleuchtung, was den Spieler beeinflussen kann und soll.
- Die **Dis-Map2** ist visuell (basierend auf RGB-Farben) übersättigt und enthält einige irritierende Elemente, wie ganze Flächen mit Zielscheibenmuster oder *Transparent Shader*.

Sowohl an der hier skizzierten Struktur der Ausgangskarte als auch an der Gestaltung der darauf aufbauenden drei Levels gibt es viele Verbesserungsmöglichkeiten. Unter anderem betrifft dies, die Position von Zielen, Distraktoren und vorhandenen *Assets* oder auch die Steuerung des *FPS-Controllers*. Dabei ist unter anderem zu berücksichtigen, wie die Test-Map von den Probanden genutzt wurde bzw. zukünftig genutzt werden soll. Dennoch muss im Hinblick auf eine Testvereinheitlichung überlegt werden, ob die Grundstruktur der Levels an dieser Stelle durch weitere *Collider* angepasst werden sollte, damit nur ein möglicher Laufweg übrig bleibt vom Start bis zum Ziel der Anwendung, oder ob die Laufwege variabel bleiben sollten. Anders herum zeigen die schnellen Laufzeiten der Probanden in den Optimierungsläufen, dass das Testsystem nur für wenige Durchläufe aussagekräftig bleibt bevor die Augen-Hand-Koordination durch Trainingseffekte verbessert wird. Abhilfe könnten hier mehr implementierte Ziele, die eventuell lokal bleiben, aber dennoch beweglich sind, Distraktoren, die auch als plötzliche *Assets* oder *Sprays* auftauchen oder die beweglich sind (siehe dazu auch Abb. A.1) und ein längerer Laufweg schaffen. Gerade letzteres ist basierend auf dem Grunddesign der Level noch möglich, z.B. durch die zwingende Einbeziehung der oberen Bereiche der beiden Gebäude in den Test. Gleichzeitig würde dies helfen, die Fähigkeiten der Probanden auszugleichen, die sie aus ihrer Aktivität in verschiedenen *Shooter* Genres mitbringen (siehe dazu auch Abschnitt 4.2).

5. Schlussfolgerungen und Ausblick

Mit den vorliegenden Ergebnissen konnte trotz nichtrepräsentativer Datenlage gezeigt werden, dass der entwickelte FPS-Präzisionstest zum Nachweis der Wirkungsweise visueller Distraktoren auf geeignete Testpersonen grundsätzlich funktioniert. Die entwickelte Anwendung erwies sich mit wiederverwendeten Elementen und *Assets* in der Grundstruktur der Karte als robust und geeignet für den Test. Die darauf basierenden Level mit drei unterschiedlichen Schwierigkeitsgraden bezüglich der Hand-Augen-Koordination führten insgesamt zum erwarteten Ergebnis und bestätigten die aufgestellte Hypothese, dass visuelle Distraktoren erfolgreich eine stärkere Ablenkung ermöglichen und weiteres Training ausnahmslos zur verbesserten Ausblendung der Distraktoren durch getestete Personen führt. Am wichtigsten für bessere und vor allem abgesicherte Testergebnisse wäre eine Erhöhung der Anzahl der teilnehmenden Probanden sowie deren spezifische Vorauswahl. Dann könnte über Gruppenvergleiche (jeweils 5-10 Probanden) zwischen z.B. professionellen, *casual* und Nicht-FPS-Spielern der Test bezüglich der Teilnehmer verlässlich gemacht werden. Die Einbeziehung einer weiteren Gruppe von FPS-Spieler mit Rot-Grün-Farbesehschwäche wäre sinnvoll.

Das entwickelte Testsystem hat an verschiedenen Stellen Verbesserungspotentiale aufgezeigt. So bedarf die Gesamtstruktur der Anwendung weiterer Optimierung. Einige strukturelle Elemente der vorhandenen Karte sind in Bezug auf die Spielbarkeit der Levels besser zu positionieren, darunter die *Collider*, *Shader* oder auch die Ziele. Ebenso muss das *Movement* des Controllers verbessert werden. Teilweise variable Laufwege sollten erhalten bleiben, eine zwingende (*Collider* basierte) Nutzung der unterschiedlichen Höhenlagen unter Einbeziehung der Gebäude und eine Erhöhung der Anzahl der Ziele auf z.B. 15-20 bietet Spielraum für weitere visuelle Distraktoren. Dies könnte die trainingsbedingt immer kürzer werdenden Durchlaufzeiten v.a. für professionelle FPS-Spieler etwas erhöhen. Ob die drei RGB-Grundfarben als Distraktor-Hauptfarben besonders gut als visuelle Ablenker wirken, ließ sich mit dem vorgestellten Test nicht ermitteln. Die Testteilnehmer befanden im Testlevel Dis-Map1 die bereits vorhandenen und hier nur in Rot-Grün-Blau eingefärbten *Assets* als interessant, aber nur minimal ablenkend, die fehlende Gangbeleuchtung wurde als störender beschrieben.

Im Level Dis-Map2 ergab v.a. die Kombination aus sehr vielen Farb- und Strukturänderungen den gewünschten Distraktoren-Effekt. Ein modifizierter Test angelehnt an die Ideen von Bavelier et al., 2012 [BAMF12] mit beweglichen, äußeren und inneren RGB-Distraktoren oder wechselnden Zielpositionen (siehe dazu Abb. A.1), eingebaut in eine FPS-Umgebung wie die hier vorliegende, könnte dazu neue detailliertere Daten liefern. Letztendlich ist auch zu überdenken, ob wechselnde Testumgebungen bzw. neu strukturierte Levels in den Durchläufen, Trainingseffekte in der Wahrnehmung und Koordination der Teilnehmer hinauszögern könnten. Umgekehrt sollte die volle Konzentration der Spieler herausgefordert werden, wodurch sich die Wirksamkeit plötzlich auftauchender und vorher nicht vorhandener visueller Distraktoren (z.B. *Sprays*) erhöhen würde. Abschließend wäre es spannend, ein Testsystem, wie das hier vorgestellte, an die neuesten Endgeräte mit Darstellungsmöglichkeiten für *Virtual Reality* anzupassen und zu testen.

6. Zusammenfassung

Die Idee dieser Arbeit war es, zu prüfen, ob in einer Gamingwelt voller neuer Technologien, verbesserter Grafik und immer mehr visuellen Distraktoren die Hand-Augen-Koordination von Spielern beeinflusst wird. Und wenn ja, wie und wie sehr? Umgekehrt, sind Spieler in der Lage, vorhandene Ablenkungen durch wiederholtes Training zu kompensieren und die Spielleistung zu verbessern? Grundgedanke dafür ist, dass Distraktoren – sogenannte Ablenker – Spieler von der eigentlichen Spielaufgabe, etwa einem taktischen oder strategischen Ziel (z.B. Levelerfolg) kurzzeitig abbringen oder die Spielspannung erhöhen, aber auch als störend empfunden werden. Dabei ist zu berücksichtigen, dass viele Distraktoren ohne Absicht in Spiele eingebaut werden, ganz einfach aufgrund der Gesamtkonstruktion einer Anwendung, teilweise aber ganz gezielt zur Manipulation der Spieler positioniert werden. Weiterhin muss berücksichtigt werden, dass jeder Spieler individuell auf bestimmte Reize oder Distraktoren in Spielen reagiert. Es ist auch wichtig zu beachten, dass ein Spiel trotz implementierter störender Elemente immer so aufgebaut wird, dass die von Michal Moore gestellte Frage „*Is it fun to play*“ [Moo16] mit ja beantwortet werden kann.

Für die Beantwortung dieser Fragestellungen wurde in dieser Arbeit eine Anwendung mit der Unity Engine erstellt, die grundlegende Mechaniken einer FPS-Zielübung beinhaltet. Diese hat drei Testlevels - *Normal Map*, *Distraktor Map 1* und *Distraktor Map 2*, in die absichtlich keine, wenige oder viele zusätzliche visuelle Distraktoren eingebaut wurden. Die ablenkenden Elemente waren unterschiedlicher Natur, orientierten sich aber insbesondere auf die RGB-Farb Wahrnehmung und die Hell-Dunkel-Wahrnehmung des Menschen. Die Distraktoren waren gezielt zur Ablenkung von Probanden in einem virtuellen Präzisions-training gedacht, basierend auf 10 zu findenden und zu treffenden *Targets*. In jenem durchliefen 5 Probanden mit ähnlichen Vorkenntnissen im *FPS-Gaming* mehrfach die erstellte Anwendung, so dass aussagekräftige Daten zur Distraktoren-Thematik erhoben werden konnten. Unterstützt wurden die Tests durch einen zusätzlich entwickelten Fragebogen, um die individuellen Eindrücke der Spieler in die Auswertung mit einbeziehen zu können. Der Test ist nicht repräsentativ, zeigte aber die folgenden Kernergebnisse. Visuelle Distraktoren hatten generell Auswirkungen auf die Hand-Augen-Koordination der Probanden. Dies wird durch die individuellen Testergebnisse belegt und auch von den Antworten der Teilnehmer auf die Fragen unterstützt.

Insgesamt betrachtet, wurde von allen Probanden zusammen für die Durchläufe der Distraktor-Maps mehr Zeit benötigt (jeweils 34% der Zeit), als in der Normal Map mit „natürlicher“ Umgebung (32% der Zeit). Drei Teilnehmer scheiterten in der Distraktor Map 2 im ersten Durchlauf. Außerdem benötigten drei der Probanden mehr Zeit für ihren ersten erfolgreichen Durchlauf in Distraktor Map 2, als in Distraktor Map 1. Vier Probanden brauchten mindestens für eine Distraktor-Map mehr Zeit, verglichen mit der Normal Map. Vier der Teilnehmer hatten nach eigener Aussage in den Distraktor-Maps Navigationsprobleme; drei trafen die Ziele schlechter, zwei davon benötigten deshalb viel Munition.

Die drei verschieden gewählten Laufwege der Probanden hatten keinen relevanten Einfluss auf die Testzeiten erfolgreicher Durchläufe im angewandten Testsystem. Nach Abschluss der Testdurchläufe 1 und 2 von den Teilnehmern durchgeführte beliebige weitere Übungsläufe mit mindestens einem abschließenden Optimierungslauf in einem beliebigen Level ergaben eine Reduktion der mittleren Durchlaufzeit von ca. 7s oder 17%. Dies war schneller, als die ersten erfolgreichen Durchläufe in der Normal Map. Zusammenfassend wurde damit gezeigt, dass die visuellen Distraktoren zunächst deutliche Effekte auf die Hand-Augen-Koordination hatten. Training verbesserte Letztere jedoch schnell und es erfolgte eine aktive Ausblendung der Distraktoren durch die Teilnehmer. Die Effektivität des Testsystems insgesamt wurde mit einer FPS-unerfahrenen Person indirekt positiv evaluiert. Diese benötigte für die *Normal* Map ca. 6min, während die Testteilnehmer über alle Durchläufe hinweg betrachtet, Zeiten von jeweils 29,5s bis ca. 01:05min benötigten. Basierend auf den vorliegenden Daten wird eine Erweiterung des Tests auf eine größere Probandenanzahl mit selektiverer Vorauswahl in einer weiter zu optimierenden Testumgebung vorgeschlagen.

Literaturverzeichnis

- [Ans06] Ulrich Ansorge: *Die Rolle von Absichten bei der automatischen Verarbeitung visuell-räumlicher Reizinformation*, *Psychologische Rundschau*, Bd. 57(1):S. 2–12, 2006.
- [BAMF12] Daphne Bavelier, Rebecca L Achtman, Merry Mani und Julia Föcker: *Neural bases of selective attention in action video game players*, *Vision research*, Bd. 61:S. 132–143, 2012.
- [Bos12] Terry RJ Bossomaier: *Introduction to the senses: from biology to computer science*, Cambridge University Press, 2012.
- [But10] Ginger Butcher: *Tour of the electromagnetic spectrum*, National Aeronautics and Space Administration, 2010.
- [Chl11] Paul Chlebek: *Praxis der User Interface-Entwicklung: Informationsstrukturen, Designpatterns, Vorgehensmuster*, Springer-Verlag, 2011.
- [Dah06] Markus Dahm: *Grundlagen der Mensch-computer-interaktion*, Pearson Studium München, 2006.
- [GB03] C Shawn Green und Daphne Bavelier: *Action video game modifies visual selective attention*, *Nature*, Bd. 423(6939):S. 534–537, 2003.
- [GDG98] Massimo Gangitano, Elena Daprati und M Gentilucci: *Visual distractors differentially interfere with the reaching and grasping components of prehension movements*, *Experimental brain research*, Bd. 122(4):S. 441–452, 1998.
- [GRH02] E Bruce Goldstein, Manfred Ritter und Gabriele Herbst: *Wahrnehmungspsychologie*, Bd. 2, Spektrum Akademischer Verlag Heidelberg, 2002.
- [Joi18] Simon Joisek: *Entwicklung und Implementierung einer 3D-Welt zur Analyse der Manipulation von Verhaltensmustern des Nutzers mittels fixierter, sowie durch Pseudozufall platzierter Attraktoren*, Hochschule Mittweida, University of Applied Sciences, BSc-Arbeit, oct 2018.

- [KL10] Thomas Kilian und Sascha Langner: *Blicke geschickt lenken: Beeinflussen mit Hilfe der Wahrnehmungsprozesse*, in *Online-Kommunikation*, S. 207–216, Springer, 2010.
- [Moo16] Michael Moore: *Basics of game design*, CRC Press, ISBN 1439867763, 2016.
- [MRC⁺18] Asifa Majid, Seán G Roberts, Ludy Cilissen, Karen Emmorey, Brenda Nicodemus, Lucinda O’grady, Bencie Woll, Barbara LeLan, Hilário De Sousa, Brian L Cansler *et al.*: *Differential coding of perception in the world’s languages*, *Proceedings of the National Academy of Sciences*, Bd. 115(45):S. 11369–11376, 2018.
- [MSC⁺07] Steven B Most, Stephen D Smith, Amy B Cooter, Bethany N Levy und David H Zald: *The naked truth: Positive, arousing distractors impair rapid target perception*, *Cognition and Emotion*, Bd. 21(5):S. 964–981, 2007.
- [Nea11] Bürger Neal: *Einfuehrung in Autodesk Maya 2012*, 2011.
URL [Einführung in Autodesk Maya 2012](#)
- [SHHH19] David Sadava, David M Hillis, H Craig Heller und Sally D Hacker: *Purves Biologie*, Springer-Verlag, 2019.
- [SS77] Richard M Shiffrin und Walter Schneider: *Controlled and automatic human information processing: II. Perceptual learning, automatic attending and a general theory.*, *Psychological review*, Bd. 84(2):S. 127, 1977.
- [Tay13] Dan Taylor: *Ten Principles of Good Level Design Part 1*, sep 2013.
URL https://www.gamasutra.com/blogs/DanTaylor/20130929/196791/Ten_Principles_of_Good_Level_Design_Part_1.php
- [Tec] Unity Technologies: *Unity User Manual*, Unity, unity user manual (2019.3) Aufl., abgerufen am 07.05.2020.
URL <https://docs.unity3d.com/Manual/index.html>
- [Wen00] Gerd Wenninger: *Lexikon der Psychologie: in fünf Bänden*, Spektrum, Akad. Verlag, 2000.
- [WKL05] Bernd Wissinger, Susanne Kohl und Molekulargenetisches Labor: *Genetische Ursachen der Farbenblindheit*, *BIOspektrum*, Bd. 11(1):S. 29–33, 2005.

Anhang

A. Zusätzliche Abbildungen

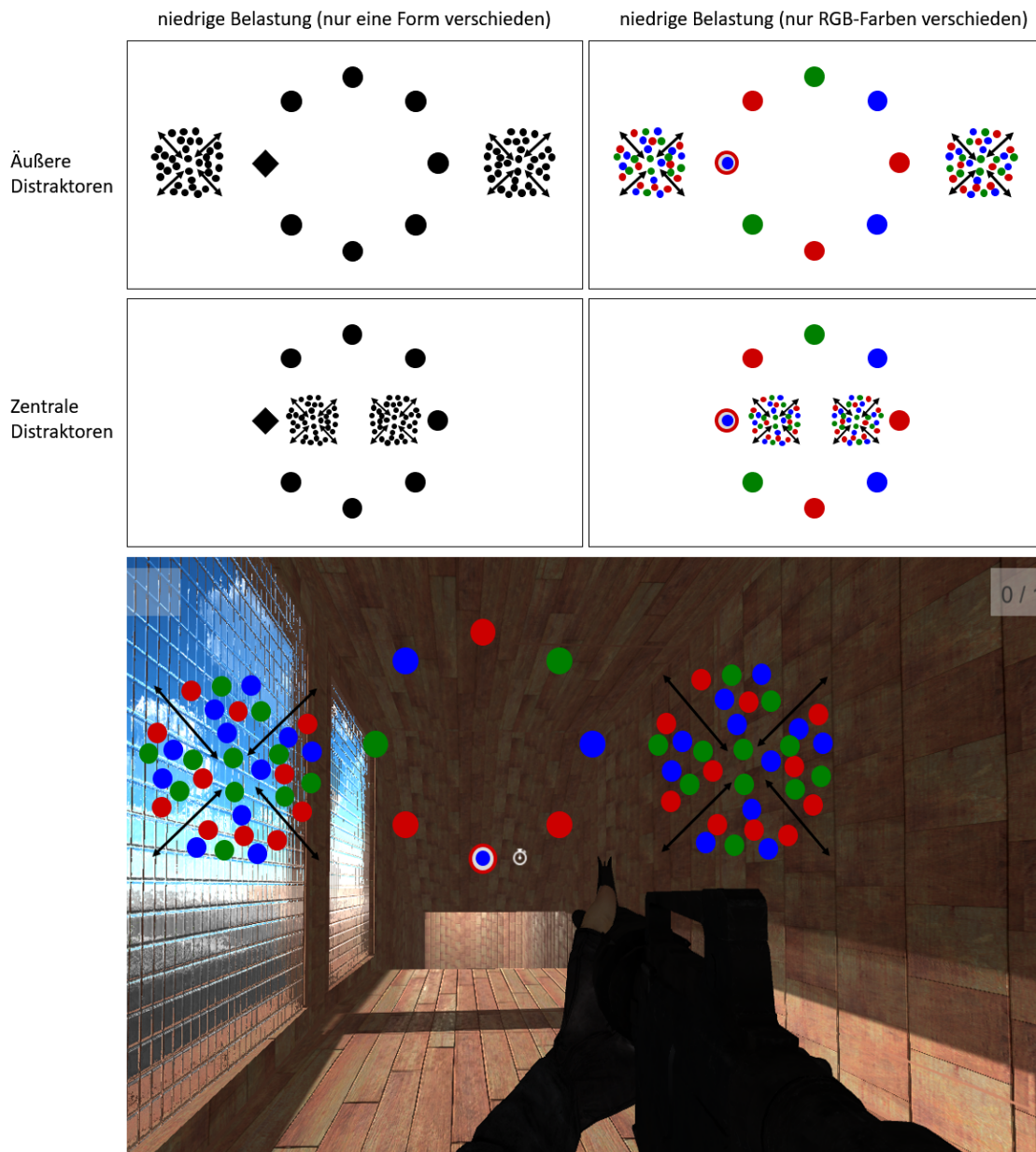


Abbildung A.1.: Test zur Prüfung der Wirkung innerer und äußerer Distraktoren. Ausschnitt einer Abbildung zur niedrigen Belastung von Probanden mit Distraktoren [BAMF12] (Oben Links). Vorschlag für einen daran angelehnten Test zur Prüfung RGB-basierter Distraktoren im Modellexperiment (Oben Rechts). Einbindung derartiger Distraktorelemente in die erstellte FPS-Testanwendung (Unten).



Abbildung A.2.: Rot-Grün-Farbschwäche: Distraktorenvergleich 1. Betrachtung derselben Szene in *Normal Map*, *Dis-Map1* und *Dis-Map2*. *Normal Map*: *Asset*- Distraktoren (Links); *Dis-Map1*: *Asset*-RGB-Distraktoren (Mitte). *Dis-Map2*: *Asset*-RGB- und *Textur*-Distraktoren (Rechts); Bildschirmeneinstellungen Standard FPS (Oben); Unterstützung von Rot-Grün-Farbschwäche (Unten) (Testdisplay: UHD4K IPS Monitor LG).

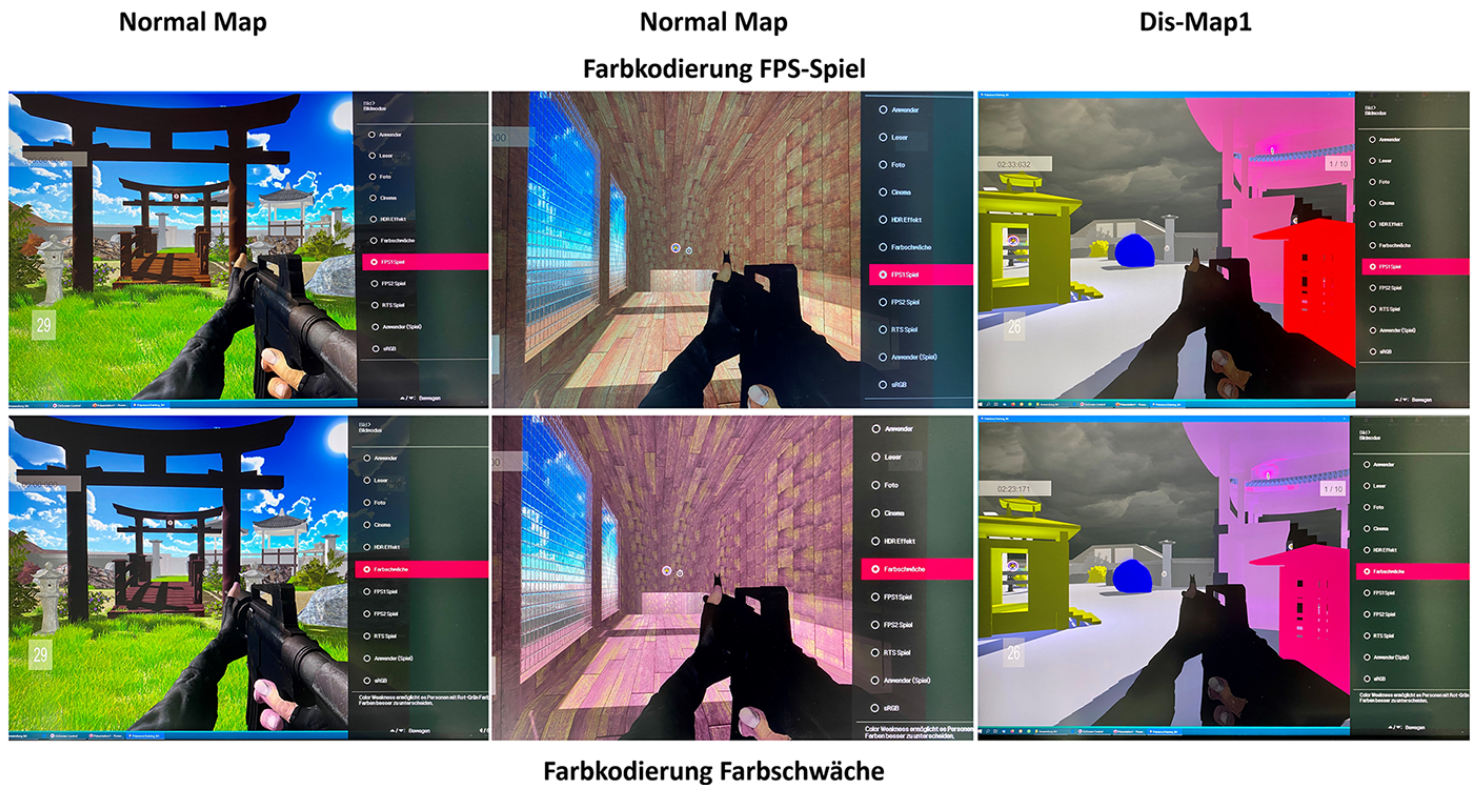


Abbildung A.3.: Rot-Grün-Farbsehschwäche: Distraktorenvergleich 2. Illustration unterschiedlicher Distraktoren. *Normal Map*: Asset-Distraktoren basierend auf Map-Gestaltung (Links); *Normal Map*: wenige Distraktoren - hier Glasfront (Mitte); *Dis-Map1*: starken RGB-farbenen Distraktoren (Rechts). Bildschirmeinstellungen Standard-FPS (Oben); Unterstützung von Rot-Grün-Farbsehschwäche (Unten) (Testdisplay: UHD4K IPS Monitor LG).

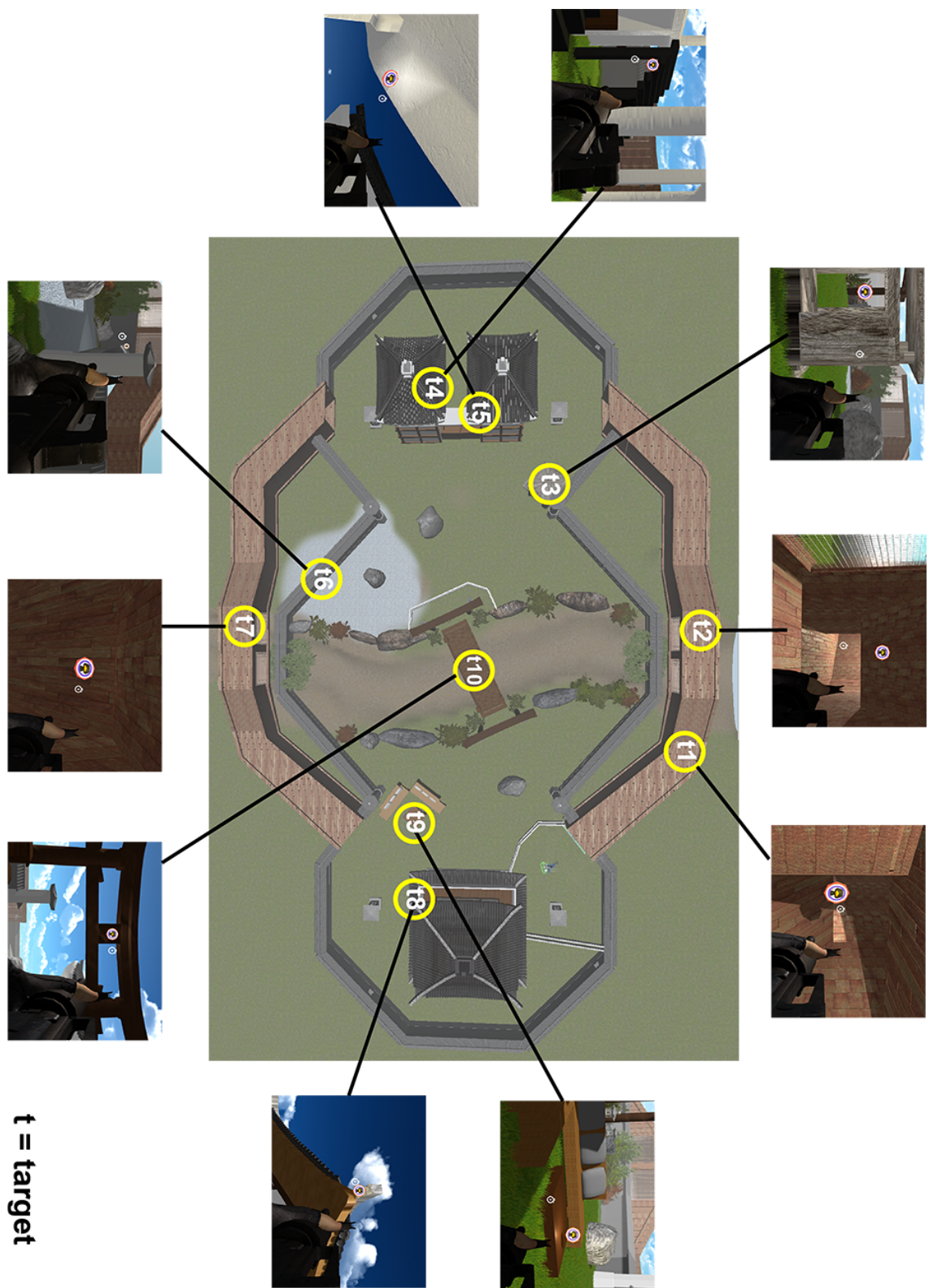


Abbildung A.4.: *Normal Map* mit Darstellung der Zielpositionen.

B. Importierte Assets und AddOns

- Unity Standard Assets <https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2017-3-32351>
- MK-Glas Free
<https://assetstore.unity.com/packages/vfx/shaders/mk-glass-free-100712>
- MK-Glow Free
<https://assetstore.unity.com/packages/vfx/shaders/fullscreen-camera-effects/mk-glow-lite-155643>
(Ordner noch auf Rechner gefunden ist mittlerweile nicht mehr Free)
- Probuilder AddOn
<https://unity3d.com/de/unity/features/worldbuilding/probuilder>
- Skybox Series Free
<https://assetstore.unity.com/packages/2d/textures-materials/sky/skybox-series-free-103633>
- Tom's Terrain Tools
<https://assetstore.unity.com/packages/tools/terrain/tom-s-terrain-tools-527>
- Clksoner Bonsai
<https://www.turbosquid.com/de/3d-models/free-3ds-model-potted-bonsai-tree-photorealistic/731271>
- Modern Weapons Unity
<https://assetstore.unity.com/packages/3d/props/guns/modern-weapons-pack-14233>
- Bamboo
<https://www.turbosquid.com/3d-models/3d-tree-nature-forest-1421571>

- Soundeffekte
<http://soundbible.com/1804-M4A1-Single.html>
<https://opengameart.org/content/library-of-game-sounds>
CS:GO SoundLibrary
- Yughues Free Wooden Floor Materials
<https://assetstore.unity.com/packages/2d/textures-materials/wood/yughues-free-wooden-floor-materials-13213>

C. Script - Dokumentation

C.1. Klassen

gun.cs

```
1
2 using UnityEngine;
3 using UnityEngine.UI;
4 using TMPro;
5
6
7 public class gun : MonoBehaviour {
8
9     public float damage = 10f;
10    public float range = 100f;
11    public float impactForce = 20f;
12    public float fireRate = 15f;
13
14    public float maxAmmo = 30;
15    public float currentAmmo;
16    public bool IsEmpty = false;
17
18    public Transform ShotText;
19
20    private float nextTimeToFire = 0f;
21
22    public Camera fpsCam;
23    public ParticleSystem muzzleFlash;
24    public GameObject impactEffect;
25
26
27    public AudioSource GunShotSound;
28    public AudioSource GunClipping;
29
30    void Start()
```

```
31     {
32         GunShotSound = GetComponent<AudioSource>();
33
34         currentAmmo = maxAmmo;
35
36     }
37     // Update is called once per frame
38     void Update()
39     {
40
41         if (Input.GetButtonDown("Fire1") && Time.time >=
            nextTimeToFire && IsEmpty == false)
42         {
43             nextTimeToFire = Time.time + 1f / fireRate;
44             Shoot();
45         }
46         else if(Input.GetButtonDown("Fire1")&& IsEmpty == true)
47         {
48             nextTimeToFire = Time.time + 1f / fireRate;
49             Empty();
50         }
51         if (currentAmmo == 0)
52         {
53             IsEmpty = true;
54
55         }
56     }
57     void Empty()
58     {
59         Debug.Log("something calls me");
60         GunClipping.Play();
61         return;
62     }
63     void Shoot()
64     {
65         muzzleFlash.Play();
66         GunShotSound.Play();
67
68         currentAmmo--;
69         ShotText.GetComponent<Text>().text = currentAmmo.ToString
            ();
70
71         RaycastHit hit;
```



```
72         if (Physics.Raycast(fpsCam.transform.position, fpsCam.  
transform.forward, out hit, range))  
73     {  
74         Debug.Log(hit.transform.name);  
75  
76         Target target = hit.transform.GetComponent<Target>();  
77         if (target != null)  
78         {  
79             target.TakeDamage(damage);  
80         }  
81         if (hit.rigidbody != null)  
82         {  
83             hit.rigidbody.AddForce(-hit.normal * impactForce)  
84             ;  
85         }  
86         GameObject ImpactGO = Instantiate(impactEffect, hit.  
point, Quaternion.LookRotation(hit.normal));  
87         Destroy(ImpactGO, 1f);  
88     }  
89 }
```

Target.cs

```
1 using System.Collections;  
2 using System.Collections.Generic;  
3 using UnityEngine;  
4  
5 public class Target : MonoBehaviour {  
6  
7     public ParticleSystem VisualKill;  
8     public AudioSource targetDestroyed;  
9     public float health = 20f;  
10  
11     public void TakeDamage(float amount)  
12     {  
13         health -= amount;  
14         if (health <= 0f)  
15         {  
16             Die();  
17         }  
18     }  
19     void Die()
```



```
20     {
21         Instantiate(VisualKill, transform.position, Quaternion.
            identity);
22
23         Destroy(gameObject);
24         scoreScript.scoreValue += 1;
25         targetDestroyed.Play();
26     }
27 }
```

Timer.cs

```
1 public class Timer : MonoBehaviour
2 {
3     private static int targetAmount;
4     public Text TimerText;
5     public bool TimerOn;
6     public float theTime;
7     public float speed = 1;
8
9     void Start()
10    {
11        TimerOn = false;
12
13        TimerText = GetComponent<Text>();
14    }
15
16    void Update()
17    {
18        if (TimerOn == true) {
19
20            theTime += Time.deltaTime * speed;
21            string minutes = Mathf.Floor((theTime % 3600) / 60).
                ToString("00");
22            string seconds = Mathf.Floor((theTime % 60)*1).
                ToString("00");
23            string milliseconds = Mathf.Floor((theTime * 1000 %
                1000)).ToString("000");
24
25            TimerText.text = minutes + ":" + seconds + ":" +
                milliseconds;
26        }
27    }
```

startCollider.cs

```
1
2 using System.Collections;
3 using System.Collections.Generic;
4 using UnityEngine;
5
6 public class startCollider : MonoBehaviour
7 {
8     public GameObject tText;
9     Timer StopUhr;
10
11     void Start()
12     {
13         StopUhr = tText.GetComponent<Timer>();
14
15     }
16
17     void OnTriggerEnter(Collider other)
18     {
19         Debug.Log("collided");
20         if (other.gameObject.name.Equals("box1"))
21         {
22             Debug.Log("Box1");
23             StopUhr.TimerOn = true;
24         }
25         else if (other.gameObject.name.Equals("box2"))
26         {
27             Debug.Log("Box2");
28             StopUhr.TimerOn = false;
29         }
30     }
31 }
```

MainMenu.cs

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5
6 public class MainMenu : MonoBehaviour
7 {
8     public void PlayGame()
9     {
10         SceneManager.LoadScene("Map_play1");
11     }
12     public void PlayRoam()
13     {
14         SceneManager.LoadScene("Map_play_Roam");
15     }
16     public void PlayDisOne()
17     {
18         SceneManager.LoadScene("Map_Distractor1");
19     }
20     public void PlayDisTwo()
21     {
22         SceneManager.LoadScene("Map_Distractor2");
23     }
24
25     public void QuitGame()
26     {
27         Debug.Log("Quitting game...");
28         Application.Quit(); ;
29     }
30
31 }
```

scoreScript.cs

```
1
2 using System.Collections;
3 using System.Collections.Generic;
4 using UnityEngine;
5 using UnityEngine.UI;
6
7 public class scoreScript : MonoBehaviour
8 {
9
10     public static int scoreValue = 0;
11     Text score;
12
13     void Start()
14     {
15         scoreValue = 0;
16         score = GetComponent<Text>();
17     }
18
19     // Update is called once per frame
20     void Update()
21     {
22         score.text = scoreValue + " / 10";
23     }
24 }
```

Crouch.cs

```
1 using UnityEngine;
2
3 public class Crouch : MonoBehaviour {
4     private CharacterController m_CharacterController;
5     private bool m_Crouch = false;
6     private float m_originalHeight;
7     [SerializeField] private float m_crouchHeight = 0.8f;
8     public KeyCode crouchKey = KeyCode.LeftControl;
9
10
11     void Start()
12     {
13         m_CharacterController = GetComponent<CharacterController>();
14         m_originalHeight = m_CharacterController.height;
15     }
16
17     void Update()
18     {
19         if (Input.GetKeyDown(crouchKey))
20         {
21             m_Crouch = !m_Crouch;
22             CheckCrouch();
23         }
24     }
25
26     void CheckCrouch()
27     {
28         if (m_Crouch == true)
29         {
30             m_CharacterController.height = m_crouchHeight;
31         }
32         else
33         {
34             m_CharacterController.height = m_originalHeight;
35         }
36     }
37 }
38 }
```

Selbstständigkeitserklärung

Hiermit erkläre ich, daß ich die vorliegende Arbeit selbstständig angefertigt, nicht anderweitig zu Prüfungszwecken vorgelegt und keine anderen als die angegebenen Hilfsmittel verwendet habe. Sämtliche wissentlich verwendete Textausschnitte, Zitate oder Inhalte anderer Verfasser wurden ausdrücklich als solche gekennzeichnet.

Mittweida, den 18. Mai 2020

Lars Goldammer