



**MEDIADESIGN Hochschule
für Design und Informatik**

**Bachelorarbeit im
Fachbereich Gamedesign**

Trigger-based Scripting: Narrative und spielerführende Wirkung

vorgelegt von:

Richard Winterstetter

Prüfer und Betreuer: Csongor Baranyai
 Prof. Dr.-Ing. Axel Hoppe

Beginn der Arbeit: 10. Juni 2011
Ende der Arbeit: 29. Juli 2011

Richard Winterstetter

Zusammenfassung

Winterstetter, Richard:

Trigger-based Scripting – Narrative und spielerführende Wirkung. 2011.

München, MEDIADESIGN Hochschule für Design und Informatik, Fachbereich Gamedesign, Bachelorarbeit.

Diese Arbeit behandelt das Thema der narrativen und spielerführenden Wirkung von trigger-based Scripting und ist in drei Teile gegliedert. Zunächst wird ein Überblick über die Technik des trigger-based Scriptings an sich gegeben. Im zweiten Teil wird der Fokus auf die narrative und spielerführende Wirkung dieser Technik gelegt und diese Wirkung im anschließenden dritten Teil dieser Arbeit auf ihren praktischen Anwendungsbereich hin überprüft.

Richard Winterstetter

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne unerlaubte fremde Hilfe angefertigt und andere als die angegebenen Quellen und Hilfsmittel nicht benutzt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Stellen sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keinem anderen Prüfungsaamt vorgelegt und auch nicht veröffentlicht.

Richard Winterstetter

Inhalt

1	Einleitung	6
2	Funktion.....	7
2.1	Scripting.....	7
2.1.1	Anfänge	7
2.1.2	Entwicklung.....	7
2.1.3	Definition.....	9
2.2	Funktionsweise	10
2.2.1	Trigger: Funktionsweise	10
2.2.2	Trigger – Event Verknüpfung	10
2.2.3	Arten von Triggern.....	14
2.2.4	Probleme und Lösungen	16
3	Wirkung von Scriptings.....	19
3.1	Theoretische Wirkungsbereiche	19
3.1.1	Atmosphäre.....	19
3.1.2	Immersion.....	19
3.1.3	Spielerführung	20
3.1.4	Narration.....	24
3.2	Spielerführende Wirkung.....	25
3.2.1	Checkpoints	26
3.2.2	Trigger-based Cutscenes	27
3.2.3	Trigger-based Attract Faktoren	27
3.2.4	Versperren des Weges	28
3.3	Narrative Wirkung	29
3.3.1	Trigger-based Cutscenes	29
3.3.2	Trigger-based Voice Over	30
3.3.3	Kontextsensitive Spielfigur Animationen	31
3.3.4	Trigger-based Attract Faktoren	31
3.4	Kombinierte Wirkung	32
3.4.1	Logik und Konsistenz.....	32
3.4.2	Rollercoaster Effect	33
3.4.3	Suspension of Disbelief.....	34
4	Analyse der Fallbeispiele	35
4.1	Bewertungskriterien.....	35
4.2	TINK – Ein Studentenprojekt [Prototyp Level 1]	35
4.2.1	Eckdaten	35
4.2.2	Analyse	35

**Trigger-based Scripting:
Narrative und spielerführende Wirkung**
Bachelorarbeit im Fachbereich Gamedesign

Richard Winterstetter

4.2.3	Auswertung.....	40
4.3	Batman Arkham Asylum – [Scarecrow Sequence 1: The Morgue]	41
4.3.1	Eckdaten	41
4.3.2	Analyse	41
4.3.3	Auswertung.....	50
5	Resümee	52
6	Ausblick.....	53
	Literaturverzeichnis	54
	Abbildungsverzeichnis.....	56
	Verzeichnis: Erwähnte Software	58

Richard Winterstetter

1 Einleitung

Diese Arbeit befasst sich mit trigger-based Scripting und dessen narrativer und spielerführender Wirkung in Spielen welche starken Gebrauch dieser Technik machen.

Der Begriff trigger-based Scripting bezeichnet dabei die, an einen Auslöser gebundene, Ausführung eines festgelegten Vorganges.

Der Sinn und Zweck dieser Arbeit soll sein, den Vorgang des Scriptings klar darzustellen, dessen Wirkung und Potential zu analysieren um anschließend zu erörtern, ob ein Studentenprojekt Scripts auf einem ähnlichen Niveau anzuwenden weiß, wie ein AAA Spiel.

Dabei wird zunächst die theoretische Funktionsweise anhand von theoretischen, realen und virtuellen Beispielen erörtert. Anschließend wird der Fokus auf die verschiedenen Wirkungsbereiche von trigger-based Scripting, im Speziellen auf das Vermitteln von narrativer und spielerführender Information, gelegt. Abschließend werden die aufgestellten Thesen anhand der praktischen Analyse zweier Spielabschnitte überprüft und ausgewertet. Hierbei wird zum einen das Studentenprojekt TINK [Mim11], zum anderen das vielgepriesene AAA Spiel Batman Arkham Asylum [Roc09] betrachtet.

Richard Winterstetter

2 Funktion

2.1 Scripting

2.1.1 Anfänge

In den frühen Tagen der Videospiele waren die Entwickler sehr stark an Hardware Limitationen gebunden. Der geringe Speicher – das Anfang der Neunziger veröffentlichte Super Nintendo Entertainment System verfügte beispielsweise nur über einen 16-Bit Prozessor [Nin11] – welcher den Entwicklern zur Verfügung stand, galt es optimal auszunutzen. Schnell wurden Wege gesucht, den aktiv zu berechnenden Levelabschnitt so gering wie möglich zu halten, ohne dabei den Spieler alle zwei Minuten mithilfe eines Ladebildschirms aus dem Spielgeschehen zu reißen.

Um dies zu vermeiden entsandt schließlich die Urform des trigger-based Scriptings. Genau platziert wurde es benutzt um Teile des Levels außerhalb des Sichtfelds des Spielers aus- und einzuladen. Türen schlossen sich hinter dem Spieler und versperrten den Blick auf den gerade passierten Bereich, welcher darauf hin aus dem Speicher gelöscht wurde. Level Designer fanden immer verwinkeltere und verschachtelte Wege ihre Levels so komplex und dennoch performant wie möglich zu bauen. Dies war mit einer der Gründe, der um die Jahrtausendwende zu einem wahren Boom an unübersichtlicher Levelarchitektur, besonders im Bereich der First-Person Shooter – führte.



Abbildung 1: Turok 2: Seeds of Evil [Igu98] zählt bis heute zu den Spielen mit den komplexesten und verwinkeltesten Leveln

2.1.2 Entwicklung

Mit zunehmender Hardware Power – spätestens mit dem Release von Sonys PlayStation 2 im Jahr 2000 [Son11] und ihrem 128-Bit Prozessor – wurde das trigger-based Scripting immer unwichtiger für performantes Level Design. Es wurden neue Techniken

Richard Winterstetter

entwickelt, die den nun größeren, zur Verfügung stehenden Speicher besser ausnutzten. Zunehmend wandelten sich Scriptings von Performanz-Tools zu kosmetischen Vorgängen. Mithilfe von Scriptings wurden Illusionen einer zusammenhängenden Welt geschaffen, komplexere Verhaltensmuster von NPC (Non Playable Characters) erstellt und weiterhin Todeszonen und Checkpoints gesetzt. Man begann zwar, trigger-based Scripting zu Zwecken der rudimentären Spielerführung zu nutzen – die Eingrenzung eines Spielers in einen kleineren Levelbereich, bis gewisse Konditionen erfüllt wurden – bis jedoch narrative Inhalte primär durch trigger-based Scripting transportiert wurden, sollte noch einiges an Zeit verstreichen.

Die bevorzugte Vorgehensweise waren Cutscenes (sowohl Ingame als auch Render Cutscenes). Ein etabliertes Mittel um narrative Inhalte zu vermitteln. Dass der Spieler durch besagte Zwischensequenzen aus dem Spielfluss gerissen wurde störte zu dieser Zeit die Wenigsten.

Erst als durch zunehmend opulentere Grafik, erhöhte Qualitätsansprüche der Spieler sowie steigende Konkurrenz Cutscenes immer aufwändiger und letztendlich auch teurer wurden, begannen Spiele-Entwickler damit immer mehr ihrer Geschichten mithilfe von Scripting zu erzählen. Schnell merkten auch die Spieler, dass mithilfe dieser Technik mehr „Spiel im Spiel“ zu finden war und man sich zunehmend vom Medium Film entfernte.

Gleichzeitig erkannte man aber auch die Problematik, welche mit der narrativen Nutzung von Scriptings verbunden war. Die Zwischensequenz war, wie bereits erwähnt, aus dem Medium Film entliehen, von daher war das gesellschaftliche Bewusstsein, die Stilmittel treffend, bekannt, akzeptiert und verständlich. Die Kniffe der Filmemacher und die Wirkung der verwendeten Bildsprache waren ausgiebig dokumentiert. Mit narrativen Scripting formte man jedoch ein neues Medium, dessen Grenzen und dessen Wirkung und Verständlichkeit auf den Konsumenten es auszutesten galt.

Viele Entwickler stützen sich bis heute teilweise immer noch auf Cutscenes und vermitteln lediglich sekundäre narrative Inhalte mithilfe von Scripting. Es gibt nur wenige mutige Studios, welche es wagten und wagen ganze Spiele frei von Cutscenes zu entwickeln und ihre primären narrativen Inhalte via Scripting vermitteln.



Abbildung 2: Half Life 2 Episode One [Val06] – Die Half Life Reihe von Entwickler Valve erzählt eine komplexe Geschichte ohne die Hilfe von Cutscenes.

Richard Winterstetter

Der enorme Erfolg mancher dieser Titel lässt Entwickler jedoch zunehmend darüber nachdenken es ihren Konkurrenten gleich zu tun und sich ebenfalls endgültig von der Cutscene zu verabschieden.

So oder so, der „Spiel im Spiel“ Anteil steigt kontinuierlich an und auch das Bewusstsein beim Spieler ist geweckt worden. Mehr und mehr erkennen Spieler innovative Ansätze im Bereich der Spielerführung und Narration in Spielen, und das trigger-based Scripting ist ganz Vorne mit dabei.

2.1.3 Definition

Doch wie genau definiert sich nun Scripting, genauer gesagt trigger-based Scripting?

Script

A computer script is a list of commands that are executed by a certain program or scripting engine.

[Tec11]

Frei übersetzt ins Deutsche sind Scripts somit Befehlsabfolgen die von bestimmten Programmen abgearbeitet werden.

Oder genauer gesagt:

„[...] wird häufig auch Scripting genannt. In der Regel dienen dazu sogenannte Scriptsprachen, die sich im Syntax an Programmiersprachen anlehnen und von einem speziellen Authoringtool (der „Engine“) interpretiert werden.“

[Gru07]

In einfacheren Worten: Scripting ist die Ausführung eines Programms innerhalb eines Programms.

Bezieht man diese Aussage nun auf Computerspiele kümmern sich Scripts um beinahe jeden komplexeren Rechenvorgang wie z.B. KI Wegfindung, Kameraverhalten etc.

Scripts sind von Programmierern gern genutzte Tools um Rechenvorgänge, die mehr als einmal benötigt werden automatisiert ablaufen zu lassen.

Was ist nun aber trigger-based Scripting?

trig·ger

1.
 - a. *The lever pressed by the finger to discharge a firearm.*
 - b. *A similar device used to release or activate a mechanism.*
 2. *An event that precipitates other events*
- [...]

[Far11]

Ins Deutsche übersetzt, handelt es sich bei Triggern um Auslöser.

Wenn man nun beide Begriffe zusammenführt handelt es sich bei trigger-based Scrip-

ting um an einen Auslöser gebundenen Ablauf von Rechenvorgängen.

2.2 Funktionsweise

2.2.1 Trigger: Funktionsweise

Bei einem Trigger handelt es sich wie bereits erörtert um einen Auslöser. Dies lässt sich im Fall der Programmierung jedoch noch spezifizieren:

“A script trigger is defined as:

A mechanism that causes a specified script to run when a particular event occurs.”

[Fil11]

Aus dieser Definition ergibt sich also, dass ein Trigger:

1. an ein spezifisches Script gebunden ist.
2. spezifische Konditionen besitzt, welche erfüllt sein müssen um ihn auszulösen.

Die Verknüpfung des Triggers mit genau definierten Auslöse-Konditionen ist hierbei das hervorzuhebende Element. Dies ermöglicht es zielgenau Scriptings auszulösen. Besonders bei Computerspielen ist dies wichtig, da mit dem Element des Spielers an sich immer ein gewisser unberechenbarer Faktor involviert ist.

2.2.2 Trigger – Event Verknüpfung

Wie bereits erörtert ergibt sich aus der vorherigen Definition des Triggers auch, dass dieser an ein spezifisches Script gebunden ist. Dies nennt man auch „Trigger – Event Verknüpfung“. Event ist hierbei nur ein anderer Name für Script.

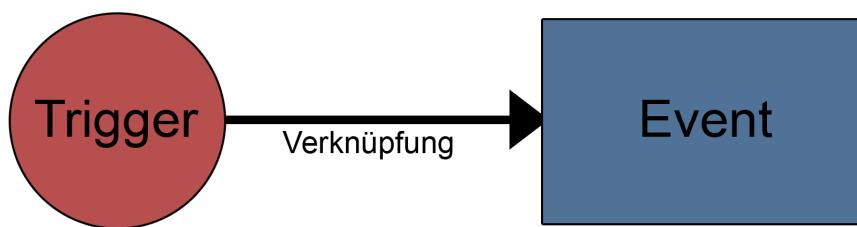


Abbildung 3: Trigger-Event Verknüpfung – schematische Darstellung

Dabei gilt es zu beachten, dass diese Verknüpfung spezifisch Trigger und Script verbin-

det. Es können auch mehrere Trigger mit einem Event/Script verknüpft sein, oder im Umkehrschluss auch mehrere Events/Scripts mit einem Trigger. Dennoch sind die Verknüpfungen zwischen Triggern und Events klar definiert und individuell.

2.2.2.1 Praktische Umsetzung mit Hilfe von UDK

Unreal Development Kit

Das Unreal Development Kit (kurz UDK) [Epi09] wird von Epic Games bereitgestellt. Dabei handelt es sich um eine Reihe von Tools, die den Umgang mit der Unreal Engine ermöglichen und erleichtern. Dieses regelmäßig mit Updates aktualisiertes Tool-Set ist dasselbe Handwerkszeug mit dem die Entwickler von Epic Games selbst arbeiten.

Die nicht kommerzielle Nutzung von UDK ist dabei kostenfrei und für jedermann zugänglich [Epi11].

Neben dem Level Editor und dem Unreal Engine internen Animationstool „Matinée“ ist auch das Scripting-Tool „Kismet“ Teil des UDK.

Kismet soll im Folgenden benutzt werden, um theoretisch die Funktion von trigger-based Scripting zu erörtern und das Verständnis der im vorherigen Teil dieser Arbeit hergeleiteten Definition zu fördern.

Kismet

Kismet bietet im Gegensatz zu anderen Scripting Tools einen sehr praxisorientierten Zugang und erlaubt es dem Benutzer komplexe an Trigger gebundene Scripts mithilfe eines Baukasten-Prinzips zu erstellen.

Erwähnenswert hierbei ist, dass durch besagtes Baukasten-Prinzip keinerlei eigenständiges Programmieren von Nöten ist. Scriptings können somit auch von Benutzern erstellt werden, die über keinerlei Programmiererfahrung verfügen.

Hierzu bedient sich Kismet einer eindeutigen Visualisierung. Trigger, Events und Verknüpfungen werden mit Hilfe eines manipulierbaren Schaubildes dargestellt was die Verständlichkeit fördert.

Trotzdem ermöglicht es das Baukasten-Prinzip Programmieren neue Elemente hinzuzufügen um Kismet um die Faktoren zu erweitern, welche für das eigene Projekt notwendig sind und nicht standardmäßig in Kismet integriert wurden.

Der modulare Aufbau, die verständliche Visualisierung und die generelle Benutzerfreundlichkeit von Kismet machen es zum optimalen Tool um trigger-based Scripting zu erörtern.

Fallbeispiel: automatisierte Schiebetür

Im Folgenden soll ein Beispiel für die Wirkungsweise von trigger-based Scripting aus der Realität gezeigt werden und anschließend mithilfe von Kismet in die virtuelle Realität übertragen werden:

Selbstöffnende Schiebetüren sind jedermann bekannt. Sie begegnen einem häufig im

Richard Winterstetter

täglichen Leben. Man nähert sich ihnen und sie öffnen sich. Mittlerweile wird dieser Öffnungs-Mechanismus durch Bewegungsmelder oder Lichtschranken ausgelöst. Um jedoch besser als Analogie zu dem späteren Kismet-Setup geeignet zu sein, wird das Fallbeispiel anhand von selbstöffnenden Schiebetüren der vorherigen Generation durchgeführt. Das Öffnen dieser Art von Schiebetür wurde durch einen im Boden oder unter einer bereitliegenden Fußmatte angebrachten Drucksensor ausgelöst.

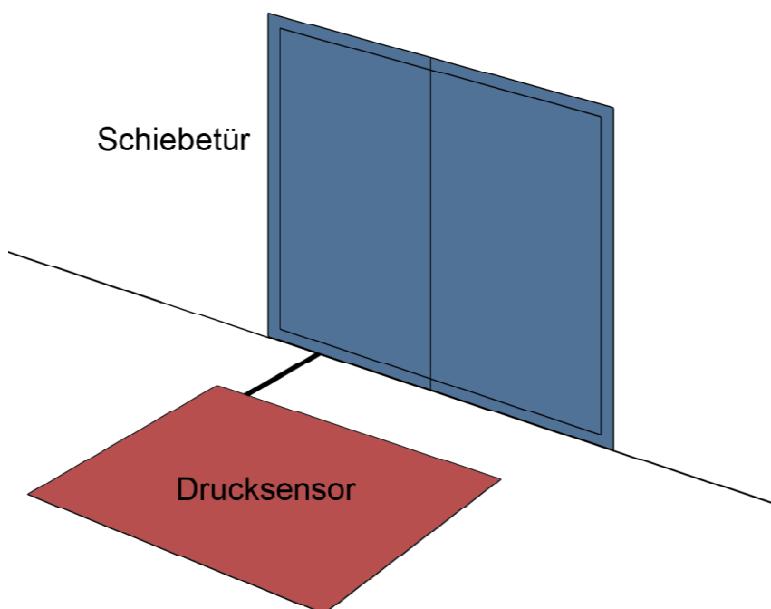


Abbildung 4: Automatisierte Schiebetür – schematische Darstellung

Wenn besagter Drucksensor aktiviert wird, sendet er ein Signal an den Öffnungsmechanismus, welcher darauf hin in Aktion tritt und die Tür öffnet.

Der Drucksensor dient somit als „Trigger“ und das Öffnen der Tür als „Event“. Da der Drucksensor mit dem Öffnen der Tür in Verbindung steht kann man von einer „Verknüpfung“ sprechen. Alle Voraussetzungen für eine Trigger-Event Verknüpfung sind vorhanden.

Visualisierung in Kismet

Im Folgenden soll das vorherig beschriebene Konzept einer automatisierten Schiebetür, welche sich automatisch öffnet, wenn sich der Spieler nähert, in UDK mithilfe von Kismet umgesetzt werden. So soll ein praktisches Beispiel der Trigger-Event Verknüpfung erläutert werden.

Zunächst wird in UDK mithilfe von einfacher Geometrie sowie einigen Static Meshes und Materials die Szene aufgebaut.

Richard Winterstetter

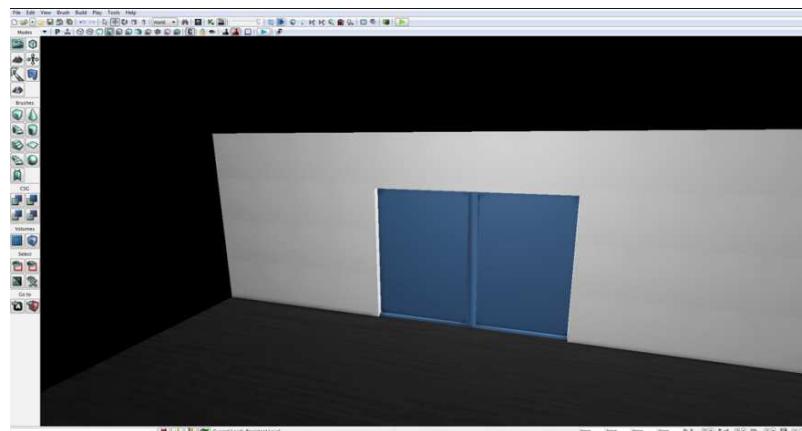


Abbildung 5: UDK Editor – Aufsetzen der Szene

Anschließend gilt es die Szene für die Interaktion mit dem Animationstool Matinée und dem Scripting Tool Kismet vorzubereiten.

Um die Türen später mit Hilfe von Matinée animieren zu können muss man dafür sorgen, dass es sich bei den zu animierenden Objekten um „Interp. Actor“ Objekte handelt.

Außerdem gilt es einen Trigger Actor zu setzen und dessen Trigger-Range (siehe 2.2.4.3) festzulegen.

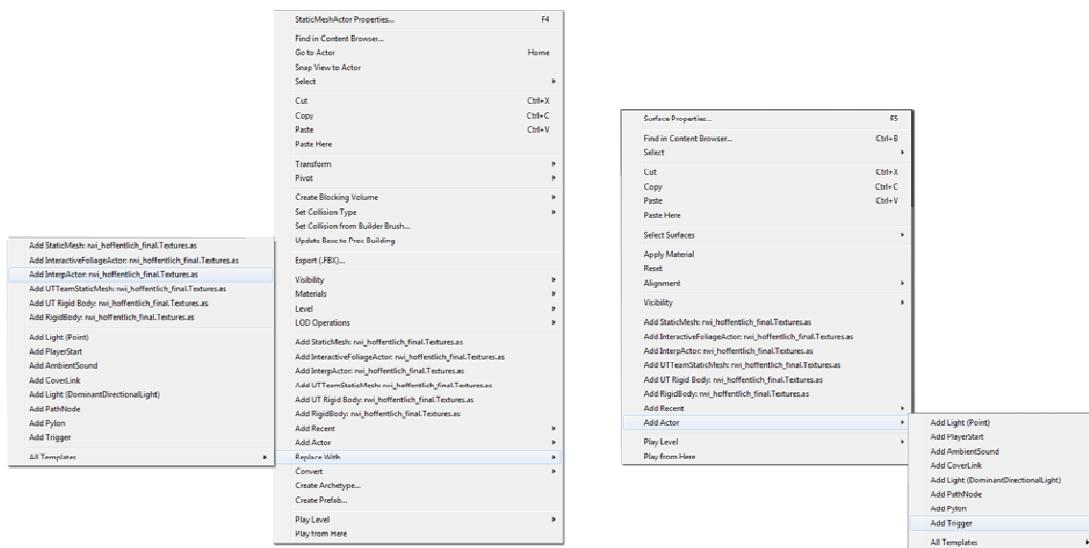


Abbildung 6: UDK Editor– Links: Festlegen der Objekteigenschaften für die Interaktion mit Matinée

Rechts: Hinzufügen eines Trigger-Actors

Öffnet man nun Kismet gilt es zunächst ein neues Event zu erstellen, welches den gesetzten Trigger benutzt. Hierbei kann man auch die Art des Triggers festlegen. In diesem Fall handelt es sich um ein „Touch“ Event – der Trigger wird aktiviert, sobald er

Richard Winterstetter

vom Spieler „berührt“ wird.

Anschließend wird die Matinée Animationsdatei erstellt, welche die Animation des „Öffnens“ der Tür beinhaltet.

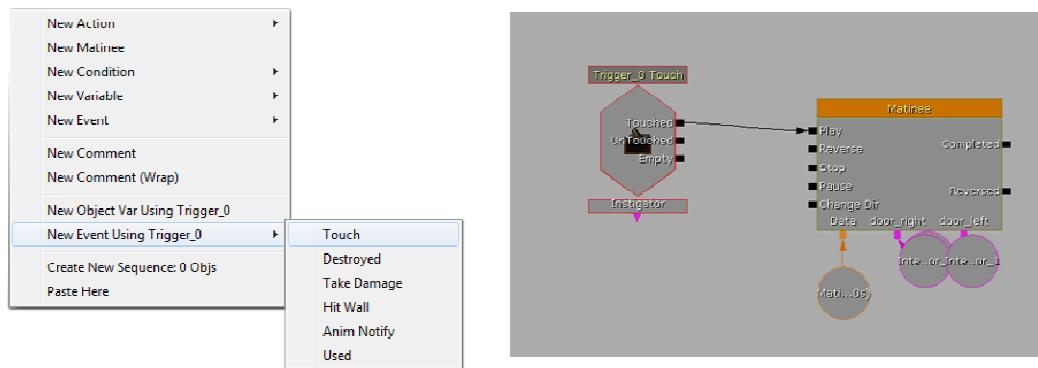


Abbildung 7: UDK Kismet – Links: Spezifizierung des Triggers

Rechts: Finale schematische Darstellung in Kismet

Abschließend verknüpft man den Trigger mit der Animationsdatei. Somit ist das Setup in UDK abgeschlossen. Startet man nun das Ganze im „Spiel“, so funktioniert die virtuelle Tür genauso wie ihr reales Vorbild und öffnet sich sobald sich der Spieler ihr nähert.

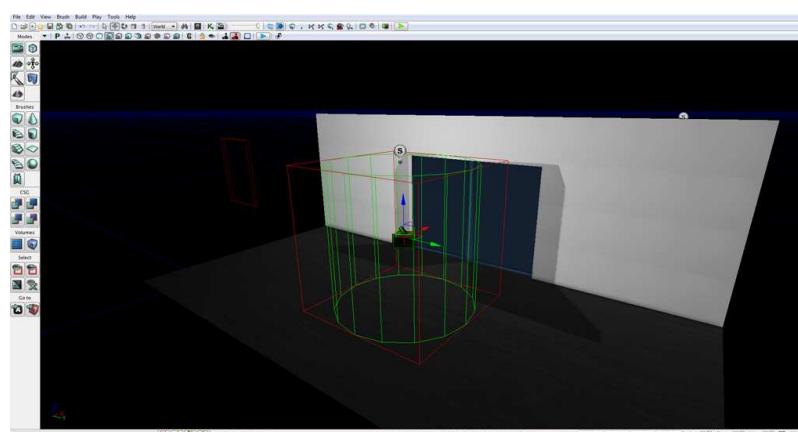


Abbildung 8: UDK Editor: Finales Setup

2.2.3 Arten von Triggern

Im Fall der automatisierten Schiebetür handelt es sich beim Trigger um eine Art Druckschalter. In Videospielen können Trigger jedoch die unterschiedlichsten Formen an-

Richard Winterstetter

nehmen. Der im Beispiel beschriebene „Druckschalter“ fällt unter die Kategorie der durch den Kontakt mit dem Spieler ausgelösten Trigger – oder „Touch-Events“ wie sie in Kismet bezeichnet werden.

Abseits der „Touch-Events“ kann aber auch das Besiegen von Gegnern, das Leeren eines Magazins, das Interagieren mit einem bestimmten Objekt oder NPC als Trigger dienen.

Jede Aktion oder Reaktion, die ausgelesen werden kann, kann auch als Trigger verwendet werden.

Ebenso kann man allen Triggern die verschiedensten Attribute zuweisen. So ist es beispielsweise möglich, das Eintreten des verknüpften Events durch einen Timer zu verzögern.

Das wichtigste Attribut eines Triggers ist jedoch wie oft er ausgelöst werden kann. hierbei wird unterschieden zwischen einmalig auslösenden Triggern und wiederholt auslösenden Triggern.

2.2.3.1 Einmalig auslösende Trigger

Einmalig auslösende Trigger lassen sich, wie es der Name bereits andeutet, nur einmal auslösen. Folglich tritt auch das verknüpfte Event nur einmal auf.

Bei der Verwendung von einmal auslösenden Triggern ist zu beachten, dass die eintretenden verknüpften Events nicht durch denselben Trigger rückgängig gemacht werden können, sofern nicht ein weiterer Trigger bereitsteht, welcher dem mit dem ersten Trigger verknüpften Event entgegenwirkt.

Bezogen auf grundlegende Spielmechaniken findet man einmalig auslösende Trigger – und damit verbundene Events – meistens bei Checkpoints. Diese werden ausgelöst sobald die entsprechenden Bedingungen erfüllt werden und halten den aktuellen Stand des Spiels fest um im Falle des Scheiterns im weiteren Spiel diesen wieder herzustellen.

Ein weiteres Verwendungsfeld von einmalig auslösenden Triggern ist das „Schließen von Türen hinter dem Spieler“. Damit ist der Vorgang gemeint, der den Spieler ab einem gewissen Punkt die Rückkehr in bereits passierte Levelabschnitte verwehrt. Dies kann buchstäblich eine Tür sein, die sich hinter dem Spieler schließt, oder aber auch ein einstürzender Minenschacht etc. Wichtig dabei ist nur, dass dieses Event nicht umkehrbar ist, da der Trigger einmal ausgelöst wird und fortan nicht mehr aktiv ist.

Im späteren Teil dieser Arbeit werden einmalig auslösende Trigger vermehrt im Bereich der narrativen Wirkung auftauchen, da sie meistens für das Vermitteln von erzählerischen Inhalten benutzt werden.

2.2.3.2 Wiederholt auslösende Trigger

Wiederholt auslösende Trigger werden – wie es der Name bereits andeutet – mehrfach ausgelöst. Dabei unterscheidet man zwischen Triggern, welche spezifisch oft auslösen – also über eine definierte Anzahl von Auslösevorgängen verfügen – und Triggern, die unendlich oft auslösen.

Richard Winterstetter

Bei Events welche mit wiederholt auslösenden Triggern verknüpft sind ist oft zu beobachten, dass das eingetretene Event ebenfalls wiederholbar ist. Oft sind Ausgangs- und Endpunkt gleich.

Die Verwendung von wiederholt auslösenden Triggern fällt meistens in eines der folgenden zwei Gebiete:

Umsetzung von Spielwelt-Regeln

Die „Umsetzung von Spielwelt-Regeln“ beschreibt allgemeingültige Abläufe und Gesetze innerhalb der Spielwelt.

Wiederholt auslösende Trigger werden hier beispielsweise bei „Todeszonen“ – Räumlich definierte Bereiche im Spiel, welche den Spieler bei Kontakt zurück zum letzten Checkpoint teleportieren – verwendet.

Auch rudimentäre Grundelemente der Spiellogik, wie das Öffnen von Türen fallen in den Verwendungsbereich von wiederholt auslösenden Triggern.

Spielerhilfe

Ein weiterer Bereich um wiederholt auslösende Trigger zu verwenden ist im Bereich der „Spielerhilfe“.

Um Frust und Stocken beim Spieler zu vermeiden empfiehlt es sich, den Spieler regelmäßig mit Hilfestellung und Tipps zu versorgen. Dies kann durchaus an einen wiederholt auslösenden Trigger gekoppelt werden, beispielsweise einen NPC, welcher in Abständen Hilfestellung zu einem zu bewältigenden Rätsel bietet.

Wiederholt auslösende Trigger werden im späteren Verlauf dieser Arbeit vermehrt im Bereich der spielerführenden Wirkung auftauchen.

2.2.4 Probleme und Lösungen

Bei der Verwendung von trigger-based Scripting gilt es, wie in jedem anderen Bereich auch, gewisse Faktoren im Auge zu behalten, um Probleme zu vermeiden oder zumindest zu minimieren.

2.2.4.1 Gameplay- und perspektivspezifische Faktoren

Wichtigster Faktor, welcher sich auf die Verwendung von trigger-based Scripting auswirkt ist das gewählte Gameplay sowie die gewählte Kamera-Ansicht.

Das Gameplay bestimmt den Fokus des Spielers sowie das Spieltempo. Ergo richten sich auch die Scriptings nach dem Gameplay.

Ist das Spieltempo beispielsweise sehr hoch, so müssen die Trigger vorausschauend gesetzt werden, da der Spieler sich unter Umständen wenn das verknüpfte Event eintritt bereits räumlich vom Trigger entfernt hat und dies möglicherweise gar nicht mehr wahrnimmt. Dies kann entweder durch Timing (siehe 2.2.4.2) oder das Anpassen der Trigger Range (siehe 2.2.4.3) ausgeglichen werden.

Ebenfalls zu beachten sind perspektivspezifische Faktoren. In einem FPS (First Person

Richard Winterstetter

Shooter) welches aus der Ich-Perspektive gespielt wird ist das Sichtfeld des Spielers auf die Umgebung direkt vor sich fokussiert wohingegen eine „Über die Schulter“ Perspektive es dem Spieler ermöglicht auch das Geschehen um seine Spielfigur herum wahrzunehmen.

Hinzu kommt, dass sich Spieler in einem FPS in der Regel schneller bewegen als in einem 3rd Person Action Adventure, da die selbe Laufgeschwindigkeit aus der Ich-Perspektive wesentlich langsamer wirkt, als aus der „Über die Schulter“ Perspektive.

All diese Faktoren gilt es zu beachten. Nicht wenige davon lassen sich durch „Timing“ beim Scripten, sowie durch anpassen der „Trigger Range“ beheben.

2.2.4.2 Timing

tim·ing

[...] the skill of doing something at exactly the right time [...]

[Lan03]

Im Falle von trigger-based Scripting bezeichnet Timing die Latenz zwischen dem Auslösen des Triggers und dem Eintreten des verknüpften Events. Diese Latenz lässt sich durch das Einsetzen von Timern steuern.

Versieht man den Trigger mit einem Timer, so startet dieser nach dem Auslösen des Triggers. Sobald die eingestellte Zeit des Timers abgelaufen ist, wird das verknüpfte Event gestartet. Man kann den Timer somit als zwischengeschaltetes Element bezeichnen.

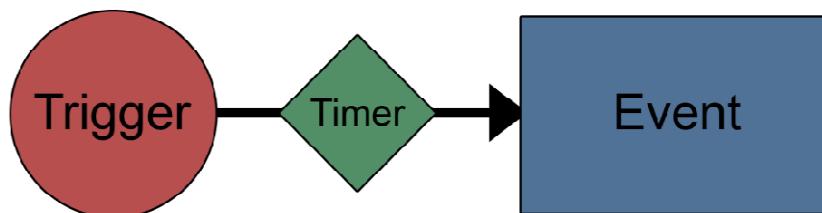


Abbildung 9: Trigger-Event Verknüpfung samt Trigger – schematische Darstellung

Das Hinzufügen eines Timers kann mehrere Vorteile bieten. Zum einen verschleiert es für den Spieler den Zusammenhang zwischen Auslöser und Event und wirkt somit dem Eintreten des „Rollercoaster-Effect“ entgegen (siehe 3.4.2). Andererseits kann es bei komplexen Scripting Systemen dem Ersteller mehr Kontrolle über die Abfolge der einzelnen Scripts ermöglichen.

Richard Winterstetter

2.2.4.3 Trigger Range

range

[...] the distance within which something can be seen or heard [...]

[Lan03]

Als Trigger Range bezeichnet man die Reichweite eines Triggers, beziehungsweise den Bereich in dem der Trigger aktiv ist. Folglich ist die Trigger Range primär wichtig bei „Touch-Event“ Triggern.

Grundsätzlich sollte gewehrleistet sein, dass Spieler im Laufe ihrer eigenen Spielerfahrung auch die Trigger auslösen, welche vom Entwickler gesetzt wurden. Dies lässt sich zu einem gewissen Teil durch Spielerführung, Kommunikationsdesign und Level Design erreichen, essentielle Trigger, wie beispielsweise Checkpoints oder Todeszonen jedoch gilt es so zu setzen, dass sie vom Spieler nicht umgangen werden können!

Hierbei ist zu beachten, dass man sich – abgesehen von 2D Spielen – in der Regel im dreidimensionalen Raum befindet. Das heißt anders als im Anfangs erwähnten Beispiel der automatisierten Schiebetür, reicht es nicht eine „Druckplatte“ als Trigger zu setzen.

Hat die Spielfigur beispielsweise die Fähigkeit zu Springen, kann vom Entwickler nicht garantiert werden, dass der Spieler genau an der Stelle, an der sich der Trigger befindet, auch den Boden berührt. Deshalb werden Touch-Event Trigger in der Regel als Volumen gesetzt, das heißt sie verfügen neben Längen- sowie Breiteninformation ebenfalls über Höheninformation, welche die räumliche Position des Triggers klar definiert.

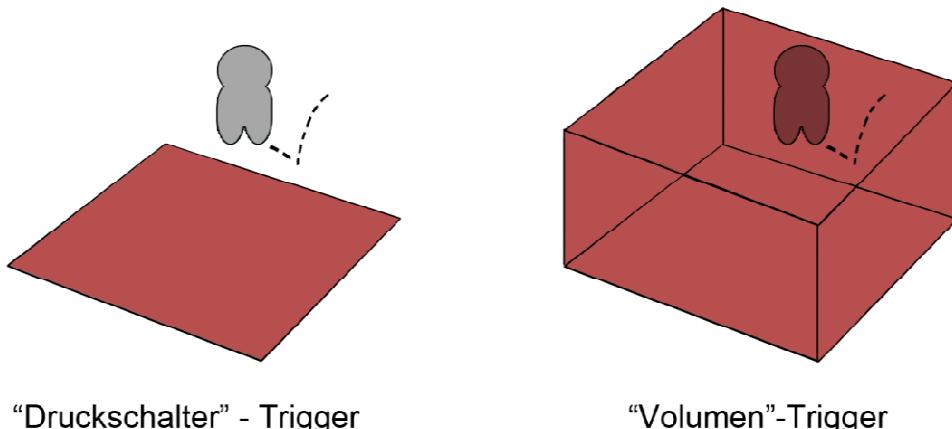


Abbildung 10: „Druckschalter“ und „Volumen“-Trigger – schematische Darstellung

Richard Winterstetter

3 Wirkung von Scriptings

"It is absolutely vital that we start to build a vocabulary that allows us to examine [...] how games evoke emotional-intellectual responses in players. " – Warren Spector

[Sal10]

Nun, da geklärt ist wie trigger-based Scripting funktioniert und was es zu beachten gilt, sollen im Folgenden ausführlich die möglichen Wirkungsweisen von Scriptings erörtert werden um die theoretische Grundlage für die Analyse der Fallbeispiele im späteren Teil dieser Arbeit zu bilden.

Dabei wird nach einer kurzen allgemeinen Betrachtung der verschiedenen möglichen Wirkungsbereiche ein Fokus auf die spielerführende sowie narrative Wirkung von Scriptings gelegt.

3.1 Theoretische Wirkungsbereiche

3.1.1 Atmosphäre

"Atmosphere is what I call those touches of detail that make [...] a game's world real and absorbing. [...] As a communication method it is more subtle than the normal clues and information used to further the game and plot. " – David Perry

[Per09]

Wie bereits im obigen Zitat von David Perry beschrieben steht Atmosphäre für die Menge an Detail, die dem Spieler ein Bild der erschaffenen Spielwelt zeichnet. Dies lässt sich dabei nicht auf einzelne Teile der Produktion beschränken, sondern ergibt sich vielmehr aus dem Zusammenspiel aller Faktoren. Angefangen vom Artstil über die erzählte Geschichte bis hin zu Level- sowie Sounddesign. Das harmonische Zusammenspiel all dieser Teilbereiche erschafft die Atmosphäre eines Spiels.

Dabei ist – wie ebenfalls von Perry erwähnt – Atmosphäre trotz des Fakts, dass sie mit jedem Element des Spiels in Verbindung steht – stets subtil und nie Kernthema des Spiels. Sie legt die Grundstimmung des Spieles fest und birgt die Grundlage für die Immersion des Spielers.

3.1.2 Immersion

im·merse

- [...] 1. [...] to plunge or dip into liquid
 2. [...] to involve deeply

[...]

[Far11]

Immersion ist, in Verbindung mit dem Medium Videospiel, der Zustand des Eintauchens in eine künstlich geschaffene Welt verbunden mit einer verminderten Wahrnehmung der eigenen Person.

Richard Winterstetter

Dabei gibt es je nach Spiel, Spieldauer und Persönlichkeit des Spielers laut Richard Bartle [Bar04] vier unterschiedliche Stufen des Immersionsgefühls:

- **Player:**
Die Spielfigur ist lediglich ein Mittel um mit der Spielwelt zu interagieren.
- **Avatar:**
Die Spielfigur ist der Repräsentant des Spielers. Es erfolgt ein gewisser Grad von Identifizierung zwischen Spieler und Spielfigur. Der Spieler spricht von seiner Spielfigur in der dritten Person.
- **Character:**
Die Spielfigur ist nicht länger Repräsentant, sondern Repräsentation. Der Grad der Identifizierung steigt. Spieler und Spielfigur verschmelzen zunehmend. Der Spieler spricht von der Spielfigur in der ersten Person.
- **Persona:**
Jegliche Unterscheidung zwischen Spieler und Spielfigur verschwindet. Der Spieler wird Teil der Spielwelt.

Hierbei ist zu beachten, dass der Übergang zwischen diesen vier Stufen fließend und in beide Richtungen während des Spiels erfolgen kann.

Der endgültige Grad der Immersion hängt zwar zu einem gewissen Teil vom Rezipienten ab, kann aber bis zu einem gewissen Grad vom Entwickler beeinflusst werden. Neben der Atmosphäre des Spiels (siehe 3.1.1) kann man den Immersionsfaktor auch durch gezielt eingesetzte Spielerführung erhöhen.

3.1.3 Spielerführung

„Spielerführung ist das was Spiele von jeglicher als Anstrengung empfundenen Arbeit unterscheidet, denn jemand hat sich die Mühe gemacht die Abarbeitung von Aufgaben so zu verpacken, dass sie als Unterhaltung empfunden wird.“ – Jens Bolanz

[Bol09]

Spielerführung ist also mehr als die bloße „Führung des Spielers“. Sie funktioniert dann am besten wenn sie am wenigsten präsent ist. Im Grunde handelt es sich bei Spielerführung um eine Reihe von bewusster sowie unbewusster Manipulationsversuche seitens der Entwickler.

Das die Spielerführung funktioniert ist wichtig um die Immersion (siehe 3.1.2) aufrecht zu erhalten. Kommt beim Spieler Frust auf, da er sich im Level nicht zu Recht findet ist es auch mit der Immersion vorbei. Hier muss die Spielerführung dem Rezipienten den rechten Weg aufzeigen.

Dabei gilt es laut Martin Nerurkar [Ner11] zwischen folgenden Arten der Spielernavigation zu unterscheiden:

Richard Winterstetter

- **Attract**

Der Spieler soll zu einem gewissen Punkt hingezogen werden.

- **Identify**

Dem Spieler soll dabei geholfen werden, bestimmte Areale klar identifizieren zu können.

- **Guides**

Der Spieler soll mit ausreichenden „Hilfssystemen“ unterstützt werden. Bei diesen Systemen kann es sich sowohl um Landkarten oder Richtungsschilder handeln, als auch um NPC Charaktere, die um wiederholte Hilfestellung bemüht sind.

Um diese Arten der Spielernavigation zu gewährleisten bedienen sich Entwickler verschiedenster Möglichkeiten:

Points of Interest:

Points of Interest (kurz PoIs) sind spezielle Objekte oder auch Arrangements, die den Blick des Spielers einfangen und somit seine Aufmerksamkeit auf sich lenken. PoIs können auch als feste Referenzpunkte innerhalb der Level Geometrie dienen und dem Spieler somit als Orientierungshilfe das „Zurechtfinden“ im Level erleichtern.

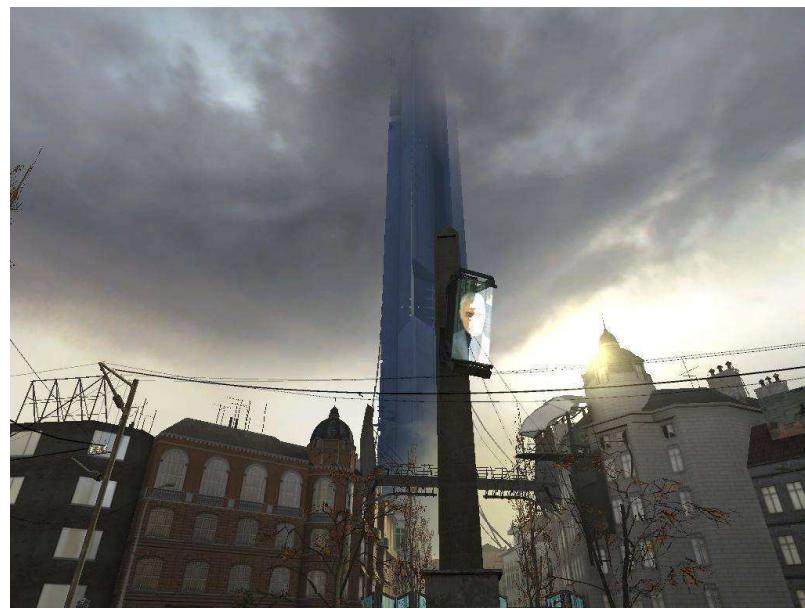


Abbildung 11: Half Life 2 [Val04] – Die Citadel fungiert immer wieder als Referenzpunkt um dem Spieler ein Gefühl dafür zu geben, wo er sich gerade in der Spielwelt befindet.

Lighting:

Lighting bezeichnet das Aus- und Beleuchten eines Levels. Hierbei können gezielt gesetzte Lichtpunkte die Aufmerksamkeit des Spielers auf sich lenken, wenn sie bei-

Richard Winterstetter

spielsweise Spielmechanik relevante Objekte an-, oder an Abzweigungen den „richtigen“ Weg beleuchten.



Abbildung 12: Dead Space [Vis08] nutzt unter Anderem geschicktes Lighting um den Spieler durch die dunklen Level zu führen.

Farbe:

Ähnlich wie mit Lighting lassen sich auch gezielt gesetzte Farbpunkte benutzt um die Aufmerksamkeit des Spielers auf sich zu ziehen. Ebenso kann gezielte Farbwahl dabei helfen den Wiedererkennungswert bestimmter Areale im Level zu erhöhen, bzw. den Weg zu gewissen Arealen zu markieren.



Abbildung 13: Team Fortress 2 [Val07] – mehrere blaue Assets weisen den Weg zur Basis des blauen Teams.

Form:

Auch gezielte Formwahl beziehungsweise das Arrangement verschiedenerer Assets kann den Blick des Spielers auf sich lenken. Ähnlich wie in der Kunst lässt sich im Level Design durch geschicktes Arrangement die Wahrnehmung des Rezipienten fokussie-

ren.

Bewegung:

Bewegungen von Objekten stechen in einer ansonsten statischen Welt hervor, weil sie dem Spieler einen gewissen Grad von Interaktion versprechen. Deswegen wird Bewegung immer die Aufmerksamkeit des Spielers auf sich ziehen.

Sound:

Sounds können ebenfalls zur Spielerführung genutzt werden, wirken in der Regel jedoch noch eine Spur unbewusster als die optischen Hinweise.

Ein Beispiel: In Tale of Tales Adaption des Rotkäppchen Märchens „The Path“ [Tal09] ertönt schnelles Herzklopfen wenn sich der Spieler vom eigentlichen Weg entfernt. Dieses Herzklopfen wird lauter umso mehr der Abstand zum „rechten“ Weg wächst.



Abbildung 14: The Path [Tal09] – Je weiter sich der Spieler vom Weg entfernt, desto lauter schlägt das Herz der Spielfigur.

Eine elegante und subtile Anwendung von trigger-based Scripting, welche der Spielerführung dient.

Haptisches Feedback:

Die Integration von Force Feedback – gezieht einsetzbare Vibration – in die Peripherie von Spielkonsolen erlaubt es Entwicklern diese ebenfalls für Zwecke der Spielerführung zu benutzen.

Ein Beispiel: In Team Bondis fünfziger Jahre Krimi „L.A. Noire“ [TBo11] gilt es in regelmäßigen Abständen Tatorte nach Hinweisen und Indizien abzusuchen. Dabei wird der Spieler auf mögliche Beweise hingewiesen indem der Controller stärker vibriert je näher sich die Spielfigur am zu untersuchenden Gegenstand befindet.

Ähnlich wie Sounds wirken gezielt eingesetzte Controllervibrationen eher unbewusst auf den Spieler und sollten deswegen bevorzugt in Kombination mit anderen, eindeutigeren spielerführenden Mittel kombiniert werden.

Generell werden meist mehrere Möglichkeiten der Spielerführung kombiniert, da dies

Richard Winterstetter

sicherstellt, dass die Navigation des Spielers erfolgreich verläuft.

3.1.4 Narration

nar·ra·tion

1. *The act or process of narrating*
2. *A narrated account or story; narrative*
- [...]

[Far11]

Narration bezeichnet das Erzählen von Geschichten – im Falle dieser Arbeit, im Medium Videospiel. Anders als in passiv rezipierten Medien gilt es jedoch die Narration im interaktiven Medium anzupassen, da nicht garantiert werden kann, dass jeder Rezipient die selbe Erfahrung erlebt und dieselben Informationen vermittelt bekommt. Im Film ist der vermittelte narrative Inhalt für jeden Zuschauer gleich. In einem Buch schreibt der Autor beispielsweise „*Er ging nach rechts!*“ und es ist somit ein für jedermann gleichermaßen anerkanntes Faktum.

Im Spiel jedoch muss man sich fragen „*Geht er nach rechts?*“ und vielmehr „*Was, wenn er nach links geht?*“

„**Interaktive Storys bedeuten Mehraufwand:** Eben gerade diese Symbiose ist es auch, welche die Gestaltung interaktiver Geschichten von linearen Geschichten unterscheidet. Die Handlung muss für den interaktiven Gebrauch angepasst werden.“

[Gru07]

Narration in Videospielen beinhaltet Elemente von Verhaltensforschung. Man muss für jede Aktion des Spielers gewappnet sein und ihr entgegenwirken.

Die Vermittlung von narrativen Inhalten erfolgt im Videospiel mithilfe mehrerer Techniken:

Passives Storytelling

Passives Storytelling (Cutscenes, Splashscreens etc.) nimmt dem Spieler die Kontrolle über das Geschehen. Wie in anderen Medien (z.B.: Film) stellt man damit sicher, dass jeder Spieler dasselbe Maß an Information zur Hintergrundgeschichte erhält. Dabei entfernt man sich jedoch vom eigentlichen Charakter des Spiels, als interaktives Narrationsmedium.

Aktives Storytelling

Aktives Storytelling lässt dem Spieler die Kontrolle über die Spielfigur und vermittelt narrative Inhalte innerhalb der Spielstruktur. Dennoch ist der Grad der vermittelten narrativen Information für alle Spieler gleich.

Interaktives Storytelling

Interaktives Storytelling macht den Spieler selbst zum Erzähler. Es nimmt die vom Spieler ausgeführten Handlungen und wandelt sie in kontextsensitive, narrative Inhalte

Richard Winterstetter

um. Somit variieren die vermittelten narrativen Informationen von Spieler zu Spieler.

Environmental Storytelling

Environmental Storytelling vermittelt narrative Inhalte durch das Design und die Anmutung der Spielwelt. Es gibt dem Spieler einen Grund die Welt zu erforschen und tiefer in die Atmosphäre des Spiels einzutauchen. Die vermittelten narrativen Informationen variieren je nach Spieler.



Abbildung 15: Parasite Eve II [Squ00] – Der umgestürzte Tisch und die Verwüstung im Restaurant deuten an, dass hier ein Kampf stattgefunden hat.

Impliziertes Storytelling

Impliziertes Storytelling ist eine direkte Konsequenz aus Atmosphäre (siehe 3.1.1) und Immersion (siehe 3.1.2). Diese Art des narrativen Inhaltes entsteht ausschließlich in den Köpfen der Spieler. Sie wird kombiniert aus den gegebenen narrativen Informationen und impliziert Zusammenhänge, Stimmungen etc. Somit variiert die Art des narrativen Inhalts von Spieler zu Spieler.

Wie auch schon bei Spielerführung so führt auch bei Narration eine Kombination der verschiedenen Ebenen zum Erfolg.

3.2 Spielerführende Wirkung

Nachdem nun die allgemeinen Wirkungsbereiche übergreifend erklärt wurden, gilt es nun den Fokus auf das trigger-based Scripting zu richten.

Im Folgenden soll ausführlich die spielerführende Wirkung von Trigger-Event Verknüpfungen, ihre Funktions-, Wirkungs- und Anwendungsweisen erklärt werden.

Wie bereits erwähnt (siehe 2.1.3) steht dem Level Designer ein weites Spektrum an Tools zur Verfügung um den Spieler durch das Level zu navigieren. Im Gegensatz zum

Richard Winterstetter

Einsatz von Farben und Formen, welche beim Rezipienten eher auf unbewusste Weise wirken, so ist Spielerführung mithilfe von Scripting meist für den Spieler bewusster zu identifizieren.

3.2.1 Checkpoints

Checkpoints verfügen über eine rudimentär spielerführende Wirkung. Allerdings muss man hier unterscheiden zwischen Checkpoints die nur einmal auslösen – also über einen einmalig auslösenden Trigger verfügen – und Checkpoints, welche wiederholt ausgelöst werden können.

Checkpoints mit einfach auslösenden Triggern findet man meistens in Spielen mit streng linearen Leveln. Hier ist der spielerführende Charakter gering bis nicht vorhanden. Lediglich die Richtung, in der die Spielfigur nach dem Bildschirmtod vom Checkpoint ausgerichtet wird kann zu Zwecken der Spielernavigation genutzt werden.

Anders verhält es sich mit mehrfach auslösablen Checkpoints. Diese verfügen über mehrfach auslösende Trigger und können somit in die Routenplanung des Spielers integriert werden. Besonders wenn Checkpoints von weitem klar als Checkpoint erkennbar sind, werden sie zum zentralen Element der Spielerführung.



Abbildung 16: Super Metroid [Nin94] – Die in den komplexen Levels verteilten Checkpoints werden von Spielern oft aktiv in die Routenplanung mit einbezogen.

Sobald ein Spieler „gelernt“ hat, dass das es sich bei dem gewählten Asset um einen Checkpoint handelt wird er bewusst und unbewusst beginnen den Checkpoint in die Routenplanung durch das Level zu integrieren.

So wird er beispielsweise vor schweren Abschnitten gerne einen Umweg in Kauf neh-

men um den Checkpoint erneut zu triggern, um den aktuellen Zustand und alle derzeitigen Werte der Spielfigur abzusichern, für den Fall, dass im kommenden Abschnitt das Game Over wartet.

Aber auch die bewusste Meidung des Checkpoints erfüllt einen spielerführenden Zweck. So kann ein sichtbar in der Distanz gesetzter Checkpoint Spieler dazu anregen, vorerst den Levelabschnitt gründlich nach Bonusobjekten und Ähnlichem zu durchsuchen und erst anschließend den Checkpoint zu triggern.

3.2.2 Trigger-based Cutscenes

Ein weiteres Beispiel für die spielerführende Wirkung von trigger-based Scripting sind durch Trigger ausgelöste Cutscenes.

Bei einer Cutscene – oder zu Deutsch: Zwischensequenz – handelt es sich um eine, vom Spielgeschehen losgelöste, durch den Spieler nicht beeinflussbare Filmsequenz.

Da trigger-based Cutscenes auch im späteren Punkt auf ihre narrative Wirkung (siehe 3.3.1) überprüft werden, wird hier ein Schwerpunkt auf getriggerte Kamerafahrten gelegt.

Seit es die technischen Limitationen erlauben In-Engine Cutscenes – in Echtzeit berechnete Zwischensequenzen – zu erstellen sind getriggerte Kamerafahrten ein althergebrachtes Mittel dem Spieler zu zeigen wo sich sein nächstes Ziel befindet. Der Spieler betätigt beispielsweise einen Schalter der als Trigger für eine gescriptete Kamerafahrt dient, welche dem Spieler den nächsten Weg weist.

Viele Videospiele bedienen sich dieser Mechanik unter anderem auch das im späteren Teil dieser Arbeit thematisierte Studentenprojekt TINK (siehe 4.2).

3.2.3 Trigger-based Attract Faktoren

Eine elegante Möglichkeit die Aufmerksamkeit des Spielers auf sich zu ziehen und seinem Fortschritt eine gezielte Richtung zu geben, ist die Verwendung von an Trigger gebundene Attract Faktoren.

at·tract

I. to make someone interested in something [...]

[Lan03]

Wie bereits beschrieben ist laut Martin Nerurkar „Attract“ – das bewusste Anziehen des Spielers – eines der Grundprinzipien der Spielerführung.

Bei getriggerten Attract Faktoren handelt es sich folglich um durch Trigger ausgelöste Scripts, welche dem Zweck dienen die Aufmerksamkeit des Spielers auf sich zu ziehen und ihn so anzulocken. Dabei kann es sich um Geschehnisse im Level handeln – beispielsweise Zerstörungsanimationen oder Interaktion mit NPCs – aber auch um subtile Dinge wie Sounds und Voice-Over, welche die Richtung weisen.

Das implizierte Versprechen von weiterführender Interaktivität wird den Spieler dazu veranlassen sich in die Richtung der Attract Faktoren zu begeben.

Richard Winterstetter

3.2.4 Versperren des Weges

“You shall not pass! “ – Gandalf Greyhame

[Tol05]

Ganz zu Beginn dieser Arbeit wurde ein Beispiel aus den Anfängen des Scripting erwähnt. Der Spieler betritt durch eine Tür den nächsten Levelabschnitt, hinter ihm schließt sich die Tür, sodass der alte Levelabschnitt dahinter aus dem Speicher gelöscht werden kann. In diesem Fall war der Einsatz von trigger-based Scripting zwar eher zur Förderung der Performanz gedacht, erfüllt jedoch auch grundlegende Funktionen von Spielerführung. Dadurch, dass dem Spieler das Zurückkehren in den alten Levelteil verwehrt wird, bleibt ihm nur der Weg nach „Vorne“ um das Level zu beenden. Im Grunde genommen wirkt das Schließen dieser Tür wie ein Guide.

Im Laufe der Zeit wurden viele Türen mithilfe von trigger-based Scripting geschlossen um dem Spieler die grobe Richtung zu weisen. Dabei gilt es zu beachten, dass diese Türen nicht immer Türen sein müssen und dass sie sich auch nicht immer hinter dem Spieler befinden müssen, denn prinzipiell handelt es sich hierbei nur um das mutwillige Versperren eines Weges durch die Entwickler.

Dies kann aus mehreren Gründen geschehen. Zum einen – wie im obigen Beispiel – um alte Levelteile auszuladen, zum anderen kann man aber auch eine einfache Levelstruktur komplexer erscheinen lassen, in dem man dem Spieler die Illusion vermittelt, es gäbe mehrere Routen, nur um ihm diese durch das „Versperren des Weges“ zu rauben.

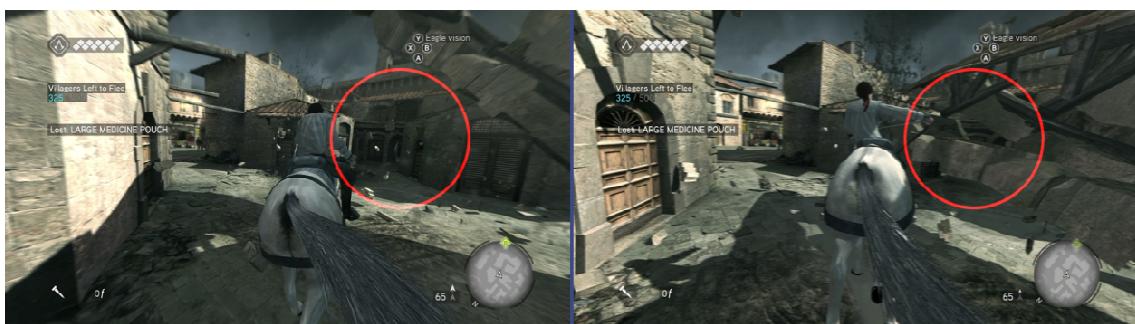


Abbildung 17: Assassin's Creed Brotherhood [Ubi10] – Links: Eine vermeintliche Weggabelung. Rechts: Der „falsche“ Weg wird versperrt.

Bei diesem „Versperren des Weges“ handelt es sich in der Regel um ein durch einen einfach auslösenden „Touch“ Trigger ausgelöstes Scripting. Man platziert den Trigger so, dass der Spieler sicher und eindeutig den zu versperrenden Punkt passiert hat - beziehungsweise sich nicht an derselben Stelle befindet - und aktiviert das Event, welches den Weg versperrt. Bei diesem Event kann es sich um eine zufallende Türe handeln, aber auch um auffälligere Aktionen, welche kaum oder nicht im Rahmen der Spiellogik rückgängig gemacht werden können - beispielsweise Steinschläge. Dem Spieler muss klar und unmissverständlich vermittelt werden, dass diese Route permanent und unumkehrbar blockiert ist und bleibt.

Richard Winterstetter

3.3 Narrative Wirkung

Die zahlreichen Möglichkeiten trigger-based Scripting in die Spielerführung eines Levels zu integrieren wurden behandelt. Nun gilt es die vielfältigen Funktionen von trigger-based Scripting und deren Wirkung im Bereich der Narration zu klären.

Hierbei werden mehrere Punkte die bereits unter „Spielerführende Wirkung“ (siehe 3.2) besprochen wurden erneut aufgegriffen, da sie ebenfalls über eine gewisse narrative Wirkung verfügen, welche es zu erläutern gilt.

3.3.1 Trigger-based Cutscenes

Durch Trigger ausgelöste Zwischensequenzen wurden bereits im Hinblick auf ihre spielerführende Wirkung untersucht (siehe 3.2.2). Trigger-based Cutscenes haben ihren Ursprung jedoch im Transport narrativer Information.

Bei der Verwendung von Zwischensequenzen bedienen sich Entwickler dem umfangreichen, erörterten und klar dokumentierten Vokabular des Mediums Film. Somit lässt sich sicherstellen, dass der Rezipient den wiedergegebenen Inhalt wahrnimmt und versteht. Zwar wird dem Spieler die Kontrolle über das Geschehen genommen, jedoch erkauft man dadurch eine sichere Art und Weise den narrativen Inhalt zu vermitteln.

Trotz, oder gerade wegen ihres Ursprungs in einem anderen Medium ist die Zwischensequenz eines der dienstältesten Mittel zum Transport narrativer Inhalte im Medium Videospiel.

Zu Beginn fielen besagte Cutscenes minimalistisch aus. Im 1989 veröffentlichten Super Mario Land [Nin89] für Nintendos Game Boy beispielsweise wird nach jedem besieгten Endgegner (Trigger) eine Sequenz abgespielt, in der sich die zu rettende Prinzessin als Illusion entpuppt und die Spielfigur Mario – und somit auch den Spieler – antreibt das nächste Level in Angriff zu nehmen.

Mit steigender Komplexität der erzählten Geschichten und den wachsenden grafischen Darstellungsmöglichkeiten formte sich die Technik der FMV – Full Motion Video – Sequenzen. Dabei handelt es sich um vorgerenderte Filmsequenzen, die nicht in Echtzeit berechnet und unabhängig vom Spielgeschehen separat eingeladen werden müssen. Die Art von FMV Cutscenes reicht dabei von real gefilmten Sequenzen mit echten Schauspielern – beispielsweise im Intro von Resident Evil [Cap96] – über aufwendig am Computer gerenderten Zwischensequenzen, welche die Final Fantasy Reihe [Squ87] berühmt gemacht hat.



Abbildung 18: Final Fantasy 7 [Squ97] –Wie für die Final Fantasy Reihe üblich wird die Geschichte in regelmäßigen Abständen via FMV Sequenzen eindrucksvoll weitergesponnen.

Da die Produktionskosten für FMV Sequenzen jedoch sehr hoch sind wurden diese bald von In-Engine Cutscenes abgelöst. Diese Zwischensequenzen werden in Echtzeit berechnet und erlauben es den Entwicklern Cutscenes mehr in den Ablauf des Spiels zu integrieren, da – anders als bei FMV Sequenzen – nicht eine komplett neue Datei geladen werden muss, sondern die Zwischensequenz in das Spielgeschehen integriert werden kann.

Dank dieser gesteigerten Flexibilität können trigger-based In-Engine Cutscenes innerhalb des Levels gesetzt werden um unmittelbar zum Spielgeschehen passenden narrativen Inhalt zu vermitteln.

3.3.2 Trigger-based Voice Over

Voice-over

The voice of an unseen narrator, or of an onscreen character not seen speaking [...]]

[Far11]

Anders als bei Cutscenes handelt es sich bei einem trigger-based Voice Over um ein narratives Mittel, welches dem Spieler nicht die Kontrolle über das Spielgeschehen entzieht.

Anders als gesprochene Dialoge oder Erzählerstimmen in Zwischensequenzen, welche festgelegten Ereignissen und einem festen Drehbuch folgen, ist das trigger-based Voice Over ein interaktives Tool. Das Drehbuch – genauer genommen, der genaue Ablauf – wird in diesem Fall vom Spieler geschrieben. Das wiedergegebene Voice Over muss zum gerade erlebten Spielgeschehen passen.

Aus diesem Grund werden die Trigger für die einzelnen Voice Over so im Level ver-

Richard Winterstetter

teilt, dass die getätigte Aussage zum gerade eintretenden Geschehen passt. Oftmals wird ein und derselbe Trigger für ein Event und das dazu passende Voice Over benutzt.

Vom narrativen Standpunkt aus lässt sich mit einem Voice Over neben primären erzählerischen Inhalt vor allem sekundäre Information an den Spieler weitergeben. Die Charakterisierung der Spielfigur und anderer NPCs lässt sich mit Hilfe dieser Technik wesentlich genauer und feiner ausarbeiten, was wiederum direkt in gesteigerter Immersion gipfelt, da es dem Spieler erlaubt sich mehr mit seiner Spielfigur und der Welt zu identifizieren.

3.3.3 Kontextsensitive Spielfigur Animationen

Ähnlich wie mit trigger-based Voice Over Events verhält es sich mit kontextsensitiven, getriggerten Spielfigur Animationen.

Hierbei wird von der jeweiligen Spielsituation oder Position der Spielfigur, kontextabhängig eine gewisse Animation getriggert, welche in gewisser Weise narrative Informationen in sich trägt. Hierbei lässt sich jedoch kein primärer Inhalt vermitteln, sondern lediglich sekundäre Narration, welche die Atmosphäre des Levels oder den Charakter der Spielfigur genauer beschreibt.

Wenn beispielsweise die Spielfigur während sie an einer Hitzequelle vorbei gesteuert wird das Gesicht mit der Hand vor der Hitze abschirmt, wirkt sie auf den Spieler unbewusst menschlicher.

Durch kleinere kontextsensitive Animationen erreicht man beim Spieler unter anderem einen gewissen immersiven Effekt und lenkt von der Tatsache ab, dass die Spielfigur nur auf die Eingabe des Spielers reagiert.

Spiele wie Uncharted 2: Among Thieves [Nau09] arbeiten neben vielen anderen Techniken viel mit kontextsensitiven Animationen und verschleiern somit fast perfekt die Abfolge der spielrelevanten, spielerkontrollierten Animationen.

3.3.4 Trigger-based Attract Faktoren

Die spielerführende Wirkung von Attract-Faktoren wurde bereits erörtert. Jedoch sind trigger-based Attract Faktoren mit das wichtigste Tool zur Vermittlung primären narrativen Inhaltes wenn man sich dazu entschließt auf Cutscenes und ähnliche, dem Spieler die Kontrolle entziehende Mittel zu verzichten.

In Half Life 2 [Val04] wird beispielsweise der komplette Fortschritt der erzählten Geschichte mithilfe von getriggerten Monologen von NPCs vermittelt. In regelmäßigen Abständen trifft man auf eine Person, welche den stummen Spielercharakter über die aktuellen Geschehnisse aufklärt. Ab dem später erschienenen Half Life 2: Episode 1 [Val06] wird dem Spieler mit Alyx ein computergesteuerter Charakter zur Seite gestellt um diese NPC Monologe besser ins Spielgeschehen zu integrieren.

Richard Winterstetter



Abbildung 19: Half Life 2: Episode 1 – Mit der Einführung des NPC Charakters Alyx wurden die trigger-based NPC Monologe besser in den Spielfluss integriert.

Des Weiteren kann man auch NPCs untereinander interagieren lassen um narrativen Inhalt zu vermitteln. Der Nachteil dieser Methode ist jedoch, dass nicht sicher gestellt werden kann, dass der Spieler diese Unterhaltungen auch bemerkt. Da die Spielfigur hierbei nicht Teil der Konversation ist kann es durchaus sein, dass sich der Spieler während des Verlaufes der Unterhaltung langweilt und sich von ihr entfernt. Zwar kann dem entgegen gewirkt werden indem man den Spieler in seiner Beweglichkeit eingrenzt oder ihm gar gänzlich die Kontrolle nimmt, will man jedoch dem Spieler die komplette Handlungsfreiheit lassen, so sollte man keinen primären narrativen Inhalt durch diese Methode vermitteln. Darüber hinaus bleiben Spielfluss und Immersion unangetastet.

Aber auch abseits des Einsatzes von NPCs lässt sich via Nutzung von trigger-based Attract Faktoren narrativer Inhalt, Atmosphäre und Immersion generieren indem man ähnlich wie im Beispiel von Assassin's Creed Brotherhood [Ubi10] das Level an sich zur Interaktion benutzt. Viele sekundäre narrative Informationen kann man durch trigger-based Environment Events vermitteln. So kann man beispielsweise das Level wesentlich größer wirken lassen, als es wirklich ist, oder dem Spieler das Gefühl geben Teil einer epischen Schlacht zu sein.

3.4 Kombinierte Wirkung

3.4.1 Logik und Konsistenz

Beim Konstruieren von trigger-based Scripts gilt es die Verknüpfung zwischen Trigger und Thematik des Scripts zu beachten. Bei sichtbar als solche erkennbaren Triggern wird der Spieler ein logisch und konsistent zur Visualisierung des Triggers einhergehendes Event erwarten. Wenn beispielsweise das Auslösen eines Schalters dafür sorgt, dass die Spielfigur die Farbe ändert – und dies im Vorfeld nicht im Rahmen des Settings

Richard Winterstetter

erklärt wurde – widerspricht das der Erwartungshaltung des Spielers und wird für Verwirrung sorgen. Der Trigger sollte in seiner auslösenden Funktion also meist logisch zum verknüpften Event stehen.

Eine rein konsistent und logisch konstruierte Abfolge von trigger-based Scripts kann beim Spieler jedoch ebenfalls zu Langeweile führen, da sie Gefahr läuft offensichtlich und vorhersehbar zu werden.

Die Aufgabe des Entwicklers ist es somit die Balance zwischen Logik und Spielspaß in Anbetracht des Spielsettings zu halten.

3.4.2 Rollercoaster Effect

“A rollercoaster is an experience that is entirely defined by some to be optimal. From the time you’re in the line, you go in the back of the rollercoaster and through the tunnel and everything is defined. We knew while you were waiting how to make the stress grow, how to make you feel something, get you scared, make you feel better, et cetera. This rollercoaster is being conceived by someone to optimize the experience. “— David Cage

[Cag08]

Je etablierter trigger-based Scripting als Transportmittel für kosmetische, narrative und spielerführende Information wird desto mehr festzustellen, dass es sich immer öfter in Spielen bei dem eigentlichen Level „nur“ noch um einen ausgeschmückten, verwickelten Schlauch handelt. Geschickt platzierte Level Geometrie und Assets sowie komplexe Scripting Systeme werden dabei benutzt um den Spieler von diesem Fakt abzulenken und ihn „durch das Level zu ziehen“.



Abbildung 20: Uncharted 2: Among Thieves führt den Spieler durch komplexes und umfangreiches Scripting mithilfe von schlauchartige Level.

Spiele wie die Uncharted [Nau09] oder Call of Duty [Inf03] Serie kreieren mit dieser Methode zwar cineastische Spielerfahrungen, stoßen aber bei immer mehr Spielern dabei auf Widerstand, da das Gefühl das eigentliche Geschehen aktiv zu beeinflussen sinkt. Dies kann man als „Rollercoaster – Effekt“ bezeichnen.

Richard Winterstetter

Wie bei einer Achterbahn steht der Ablauf solcher Level bereits im Vorhinein fest. Die Aktionen des Spielers haben wenig bis keinen Einfluss und der einzige spielerische Inhalt des Levels ist der, den Endpunkt zu erreichen.

Um den „Rollercoaster-Effekt“ zu vermeiden sollte man beim Setzen von Scriptings immer darauf achten, spielerische Freiheit zu bieten oder die Events an sich subtiler zu gestalten.

3.4.3 Suspension of Disbelief

“We know this is a lot of nonsense, but let's forget about that for a couple of hours and allow ourselves to have some fun”. – Peter Jackson

[Fit09]

Der aus der Filmbranche stammende Begriff „Suspension of Disbelief“ bezeichnet ein unausgesprochenes Abkommen zwischen den Medienschaffenden und den Rezipienten. Beide Parteien wissen, dass das geschaffene Werk reine Fiktion ist, trotzdem stellen die Rezipienten diesen Fakt beiseite und nehmen das Erlebte für gegeben. Diese Annahme erlaubt es ihnen, das Medium als Unterhaltung zu genießen.

Obwohl dieses Theorem aus der Filmbranche stammt trifft es auf Videospiele ebenfalls zu. Das Gezeigte in Spielen ist nicht nur reine Fiktion, es bedient sich, anders als das Medium Film, eines virtuell geschaffenen Vokabulars. Umso wichtiger ist es für Entwickler, dass trotz ihrer Fiktion und Virtualität das Geschaffene so glaubwürdig und konsistent wie möglich präsentiert wird um den besprochenen „Suspension of Disbelief“ Faktor erreichen zu können.

Ein großer Teil dieses Aufwandes bildet dabei das Zusammenspiel von spielerführender und narrativer Wirkung des Scriptings um unabhängig von Design die Illusion einer interagierenden Welt zu schaffen, in welcher die Spielfigur einen konsistenten Teil darstellt.

Richard Winterstetter

4 Analyse der Fallbeispiele

4.1 Bewertungskriterien

Die folgenden Analysen werden auf Grundlage der in den vorherigen Teilen dieser Arbeit ermittelten Kriterien und Abgrenzungen getätig. Dabei wird zuerst wertungsfrei der gewählte Abschnitt Script für Script analysiert und im Anschluss ausgewertet.

Diese Auswertung erfolgt aufgrund mehrerer Kriterien:

Anzahl

War die Anzahl der im analysierten Spielabschnitt verwendeten Scripts gering oder hoch?

Differenziertheit

Wurden unterschiedliche Ansätze gewählt oder nur mit wenigen Methoden gearbeitet?

Spiefluss

Wird dem Spieler die Kontrolle entnommen? Überschatten die Scripts das eigentliche Gameplay?

Zusammenspiel

Wie wirkt die narrative und spielerführende Wirkung der Scripts im Zusammenspiel auf den Spieler? Wurde auf die Auswirkungen von „Rollercoaster Effect“ und „Suspension of Disbelief“ geachtet oder wurden diese im Idealfall sogar integriert? Sind Trigger und Events im gewählten Setting logisch und konsistent?

4.2 TINK – Ein Studentenprojekt [Prototyp Level 1]

4.2.1 Eckdaten

TINK [Mim11] ist ein 3rd Person Action-Adventure, welches von elf Studenten der Mediadesign Hochschule München unter dem Namen MiMiMi Productions entwickelt wurde. Das Gameplay kombiniert dabei geschickt vom Trendsport Parcour inspirierte Jump and Run Einlagen und ein hohes Spieltempo mit komplexen, auf Farben basierenden Rätseln nach dem Stein, Schere, Papier Prinzip.

Der Stil der Welt und der Charaktere wurde dabei von Pappmaché-Basteleien und Pinatas inspiriert und ergibt zusammen mit den Farbrätseln das Setting der „Farbstadt“.

Der Spieler übernimmt die Kontrolle über Tink, einen Bewohner der Welt, der die „Farbstadt“ mithilfe seiner „Farbkräfte“ gegen das „Weiß“ und dessen Handlanger verteidigt.

4.2.2 Analyse

4.2.2.1 Kontext:

Bei dem zu analysierenden Abschnitt handelt es sich um das erste aus drei Leveln des im Laufe der Projektarbeit entstandenen ersten Prototypen von TINK. Im Laufe dieses

Richard Winterstetter

in der Farbstadt angesiedelten Levels soll dem Spieler die rudimentäre Bewegungssteuerung beigebracht werden.

4.2.2.2 Establishing Cutscene

Sobald das Level fertig geladen ist wird dadurch eine gescriptete Cutscene getriggert, welche dem Spieler die erste Hälfte des Levels zeigt.

Innerhalb der Cutscene wird der zu bewältigende Weg gezeigt. Außerdem fliegt – ebenfalls gescriptet – eine Gruppe von Bienen durch das Bild.

Diese einführende, trigger-based Cutscene erfüllt insofern einen spielerführenden Zweck, da sie dem Spieler bereits vor Spielbeginn einen Eindruck des Levels vermittelt und ihm den vor ihm liegenden Weg zeigt.

Die gescripteten Bienen, welche durch das Bild fliegen fallen in die Kategorie der trigger-based Attract Faktoren. Der Bienenflug wird wie die Cutscene selbst durch das Starten des Levels getriggert und startet nach dem Ablauf eines Timers. Der Blick des Spielers soll eingefangen werden und es soll der Eindruck einer lebendigen Welt vermittelt werden.



Abbildung 21: TINK – Establishing Cutscene samt gescripteten Bienenflug

4.2.2.3 Tutorialschaf #1

Nachdem der Spieler die Kontrolle erlangt hat erwartet ihn nach wenigen Metern das erste Tutorialschaf.

Bei den Tutorialschafen handelt es sich um getriggerte Attract Faktoren, welche den Blick des Spielers einfangen sollen.

Diese von Pinatas inspirierten Schafe wurden von den Level Designern verwendet um Spielmechaniken via Sprechblasen zu vermitteln. Ausgelöst wird das Erscheinen der Sprechblasen durch einen Touch-Trigger.



Abbildung 22: TINK – Tutorialschaf #1

In diesem Fall befindet sich der Trigger innerhalb des Tunnels, den der Spieler durchqueren muss. Somit wird sichergestellt, dass der Trigger auch ausgelöst wird und dem Spieler die Information zugetragen wird. In Falle dieses Schafes handelt es sich um die nötige Anweisung den „Turnmodus“ zu aktivieren, der es dem Spieler erlaubt automatisch über Abgründe hinweg- und auf entsprechende Objekten hinaufzuspringen.

4.2.2.4 Ein schwimmendes Fass

Nach dem „Übungsbereich“ der im Anschluss an das erste Tutorialschaf aufgebaut wurde wird mit dem Sprung auf den ersten Balkon über dem Farbfluss via Touch-Trigger ein Event getriggert. In diesem Event treibt ein Fass den Farbfluss entlang.

Dieser getriggerte Attract-Faktor soll zum einem dem Spieler die grobe Richtung weisen. Auf narrativer Ebene soll dieses Scripting dem Aufbau von Atmosphäre dienen und das Level weniger steril und spielerbezogen wirken lassen.



Abbildung 23: TINK – Ein schwimmendes Fass

Richard Winterstetter

4.2.2.5 Megasprung Cutscene

Nach einigen Sprüngen triggert der Spieler eine Cutscene welche eine Sprungdistanz zeigt, die nicht durch den normalen Einsatz des „Turnmodus“ bewältigt werden kann. Hierfür muss der Spieler den Megasprung einsetzen.

Diese trigger-based Cutscene wurde gesetzt um den Spielfluss zu bremsen, da sonst viele Spieler unbeabsichtigt in den Fluss stürzen würden und somit negatives Spieler-feedback entstehen würde.

Der Trigger für die Cutscene dient gleichzeitig als Auslöser für den ersten Checkpoint in diesem Level.

4.2.2.6 Tutorialschaf #2

Nach Ablauf der „Megasprung“ Cutscene befindet sich das Tutorialschaf #2 gleich im Sichtfeld des Spielers und die Spielfigur im Touch-Trigger, sodass die Textblase sofort sichtbar ist.



Abbildung 24: TINK – Tutorialschaf #2

In der Textblase wird dem Spieler die nötige Information zu teil, die benötigt wird um einen Megasprung auszuführen und die Plattform auf der anderen Seite des Farbflusses zu erreichen.

4.2.2.7 Tutorialschaf #3

Nach Bewältigung des „Megasprungs“ und einigen weiteren Sprungeinlagen gelangt der Spieler zum dritten Tutorialschaf in diesem Level. Der Touch-Trigger dient gleichzeitig als Auslöser für den zweiten Checkpoint.

Richard Winterstetter



Abbildung 25: TINK – Tutorialschaf #3

Hier wird dem Spieler die Information vermittelt, dass nur bei deaktiviertem „Turnmodus“ gewährleistet ist, dass die Spielfigur nicht über Kanten in das Game Over stürzt.

4.2.2.8 Libellenflug

Nach einiger Zeit gelangt der Spieler in einen eher offenen, waldartigen Levelabschnitt. Auf dem primären Weg fliegt getriggert durch einen Touch-Trigger und geregelt durch einen Timer in regelmäßigen Abständen eine Libelle entlang.



Abbildung 26: TINK – Die Libelle weist den Weg zum finalen Levelabschnitt

Auf narrativer Ebene handelt es sich hierbei, wie schon bei dem treibenden Fass um einen Atmosphäre erzeugenden Attract-Faktor. Dadurch, dass die Libelle jedoch in Richtung des finalen Levelabschnittes fliegt besteht auch eine gewisse spielerführende Wirkung.

4.2.2.9 Tutorialschaf #4

Das vierte und letzte Tutorialschaf in diesem Level begegnet dem Spieler direkt vor

Richard Winterstetter

dem Levelausgang. Der vermittelte Inhalt ist bei diesem Schaf narrativer Natur, da es den Spieler zur Eile bewegen soll.



Abbildung 27: TINK – Tutorialschaf #4

Trigger und Event sind so platziert, dass sie der Spieler nach Bewältigung der letzten Sprungpassage in Level 1 automatisch im Blickfeld hat.

Begibt sich der Spieler in den Durchgang neben dem vierten Tutorialschaf wird der Levelwechsel getriggert und das erste Level ist beendet.

4.2.3 Auswertung

Die folgende Auswertung des analysierten Spielabschnittes wird zunächst rein in Anbe tracht der Faktoren Anzahl, Differenziertheit, Spielfluss und Zusammenspiel betrachtet. Im weiteren Verlauf wird das Ergebnis in Relation mit externen Faktoren wie Entwicklungszeitraum und verwendeter Engine gewertet.

Anzahl

Die Anzahl der verwendeten Scripts ist in Relation zur Länge und zum Umfang des Abschnittes gering.

Differenziertheit

Bei allen Auslösern handelt es sich um Touch-Trigger. Selten werden Timer verwendet und bei den verknüpften Events handelt es sich größtenteils um Attract-Faktoren und Cutscenes.

Spielfluss

Der Spielfluss ist durchgehend hoch. Die getriggerte Zwischensequenz, welche den ersten Megasprung einführt ist bewusst gewählt und hat trotz ihrer unterbrechenden Wirkung weniger Einfluss auf den Spielfluss als ein – ohne Einsatz dieser Cutscene – sehr wahrscheinlicher Bildschirmtod.

Zusammenspiel

Auf spielerführender Ebene funktionieren die gewählten Scripts. Dem Spieler werden Levelziele und mögliche Routen klar vermittelt ohne das Gefühl zu vermitteln, dass sein Agieren keinerlei Einfluss auf das Spielgefühl hat. Aus narrativer Sicht bietet der gewählte Abschnitt im Bereich der trigger-based Scripts wenig Inhalt. Die meiste narrative Information wird durch Environmental Storytelling vermittelt und nicht durch Scripts.

Fazit

Objektiv betrachtet befindet sich das trigger-based Scripting im analysierten Abschnitt von TINK im Vergleich zu anderen Spielen im unteren Mittelfeld. Die Art der Scripts und Trigger sind arm an Abwechslung und es ist klar erkennbar wie gewisse Events ausgelöst werden können.

Dies ist jedoch auf den Zustand zurückzuführen, dass der Prototyp innerhalb von drei Wochen von einem sehr kleinen Team entwickelt wurde. Alle eingesetzten Arten von trigger-based Scripting mussten als Tools von Programmierern geschrieben werden um von den Level Designern verwendet werden zu können, da in Unity [Uni11] kein benutzerfreundliches Scripting Tool wie Kismet zur Verfügung steht, welches es den Designern erlaubt ohne Programmierkenntnisse eigenständig Scripts zu erstellen.

4.3 Batman Arkham Asylum – [Scarecrow Sequence 1: The Morgue]

4.3.1 Eckdaten

Bei Batman Arkham Asylum [Roc09] handelt es sich um ein 2009 erschienenes Action Adventure von Entwickler Rocksteady. Die Geschichte des Spiels basiert dabei auf der erfolgreichen Graphic Novel von Grant Morrison [Mor09], in welcher Batmans Erzfeind, der Joker, die Kontrolle über die Nervenheilanstalt Arkham an sich reißt und Batman dazu zwingt sich seinen tiefsten Ängsten zu stellen.

Das Spiel wurde sowohl von Presse als auch von den Spielern hoch gelobt – nicht zuletzt wegen der innovativen Scarecrow-Sequenzen – und gewann mehrere „Spiel des Jahres“ Auszeichnungen. Es ist für Xbox 360, PlayStation 3 und PC erhältlich. Für Ende 2011 ist ein Nachfolger „Batman: Arkham City“ geplant.

4.3.2 Analyse

4.3.2.1 Kontext:

Nachdem der Joker die Anstalt unter seine Kontrolle gebracht und Commissioner Gordon entführt hat macht sich Batman – respektive der Spieler – auf die Suche nach besagtem Comissioner. In dem Moment, in dem der Spieler den Aufzug, welcher ihn zur Leichenhalle bringen soll betritt, wird Batman dem Gas des Schurken „Scarecrow“ ausgesetzt, welches die Urängste der Spielfigur freisetzt und ihn halluzinieren lässt.

4.3.2.2 Der Aufzug

Der Touch-Trigger welcher vom Spieler ausgelöst wird, sobald er den Aufzug betritt

Richard Winterstetter

lässt mehrere Events aus.

Die Aufzugtüren schließen sich. Der Bildschirm erwacht zum Leben und zeigt eine Aufnahme von Joker. Die zur Aufnahme passenden Voice-Over werden abgespielt. Des Weiteren starten mit dem ausgelösten Trigger zwei Timer.



Abbildung 28: Batman Arkham Asylum – Der Aufzug

Sobald der erste Timer abgelaufen ist, setzt sich der Aufzug in Bewegung. Das Ablauen des zweiten Timers lässt Gas in den Aufzug strömen und startet eine „Husten“-Animation der Spielfigur. Der Timer dieses Events ist dabei auf die Voice-Over des Jokers abgestimmt, sodass das Einströmen des Gases gleichzeitig mit den Worten „[...] to face your fears. All of them!“ auftritt.

All diese Scripts dienen der Vermittlung narrativer Information. Die Atmosphäre wird durch geschicktes Spiel mit dem psychologischen Grundmotiv Klaustrophobie und impliziertem Storytelling – in diesem Fall der Vorahnung und Erwartung des Spielers – erhöht.

4.3.2.3 Cutscene #1: Dr Crane

Mit dem Ende der Aufzugsfahrt wird eine Cutscene getriggert, welche den Spieler mit der neuen Umgebung – dem Keller der Anstalt – vertraut macht, sowie narrative Inhalte vermittelt. Die Auswirkung des Gases wird anhand von intoxinierten Insassen verdeutlicht und Dr Cranes monströses Alter Ego, Scarecrow wird anhand eines markanten Silhouettenschattens eingeführt. Darüber hinaus wird verdeutlicht, dass der direkte Weg in den Kellerbereich versperrt ist.

Die spielerführende Wirkung dieser Zwischensequenz betrifft einzig das Einführen der neuen Umgebung wohingegen auf narrativer Ebene neben den bereits erwähnten Informationen ebenfalls durch Präsentation und Schnitt eine beklemmende Atmosphäre aufgebaut wird.

Das Ende der Cutscene geht fließend in die Spielansicht über und dient gleichzeitig als Trigger sowohl für neue Voice-Over des Jokers via Lautsprecher als auch für den ersten Checkpoint in diesem Abschnitt.

Richard Winterstetter

4.3.2.4 Cutscene #2: Gordon

Nachdem sich der Spieler nach der Eröffnungssequenz Zugang zum versperrten Kellerbereich verschafft hat, wird via Touch-Trigger eine weitere Zwischensequenz getriggert, welche den hilflosen Commissioner zeigt, der von einer unbekannten Kraft aus dem Bild gezerrt wird.



Abbildung 29: Batman Arkham Asylum – Commissioner Gordon

Der Trigger für dieses Event dient gleichzeitig als zweiter Checkpoint in diesem Abschnitt.

4.3.2.5 Cutscene #3: Zu spät

Der Spieler muss nachdem ein Lüftungsschacht durchquert wurde einer Blutspur folgen, welche bei dem leblosen Körper von Commissioner Gordon endet. Hier wird via Touch-Trigger eine weitere Cutscene ausgelöst.

Diese Cutscene ist größtenteils narrativer Natur und vermittelt neben dem tragischen Ende von Gordon ebenfalls erste Hinweise auf die Auswirkung des Gases auf Batman, respektive die Spielfigur (glühende Augen, wirr flackerndes Licht). Einziges spielerführendes Element dieser Sequenz ist das erneute Auftauchen des Silhouettenschattens von Scarecrow, welcher dem Spieler die weitere Fortschrittsrichtung weist.

Richard Winterstetter



Abbildung 30: Batman Arkham Asylum – Commissioner Gordon ist tot.

4.3.2.6 Der Weg zur Leichenhalle

Das Ende der Sequenz triggert ein Voice-Over von Batman, welcher versucht Oracle, die Tochter des Commissioners – via Funk zu erreichen, sowie die passende Animation.



Abbildung 31: Batman Arkham Asylum – Funkanimation

Während der Spieler den Gang entlang schreitet wird ihm durch flackernde Lichter, am Boden entlanglaufende Kakerlaken und eine leicht schräge Kameraposition die Auswirkungen des Gases immer deutlicher vor Augen geführt.

Im ersten Drittel des Ganges wird via Touch-Trigger ein Event ausgelöst, welches die Fliesen von den Wänden des Korridors herabfallen lässt. Dies dient ebenfalls der Vermittlung der Auswirkungen des Gases.

Richard Winterstetter



Abbildung 32: Batman Arkham Asylum – Fliesen fallen von den Wänden

Im zweiten Drittel des Ganges wird ebenfalls via Touch Trigger ein blitzartiges erscheinen des Silhouettenschattens von Scarecrow ausgelöst. Neben der offensichtlichen narrativen Wirkung ist dies rudimentär spielerführend, da es die Aufmerksamkeit auf das Ende des Ganges lenkt.



Abbildung 33: Batman Arkham Asylum – Der Silhouettenschatten wird zunehmend zum zentralen spielerführenden Element.

Bei der Mehrheit der Scripts, welche während der Durchquerung des Korridors in Aktion treten handelt es sich um trigger-based Attract-Scripts, welche den Blick des Spielers einfangen sollen. Die narrative Wirkung steht in diesem Abschnitt klar im Vordergrund.

4.3.2.7 Die Leichenhalle #1

Am Ende des Korridors betritt der Spieler die Leichenhalle. Die Interaktion mit der Tür dient hierbei als Trigger für mehrere Scripts. Zum einen wird ein Checkpoint ausgelöst. Zum anderen werden diverse Attract-Scripts aktiviert welche Atmosphäre und Immersion fördern sollen. Bei diesen Attract Scripts handelt es sich beispielsweise um wie von

Richard Winterstetter

Geisterhand auf und zu schlagende Leichenkühlzellen.

Auf technischer Ebene wird neben dem Checkpoint durch das Betreten der Leichenhalle auch ein Ladevorgang ausgelöst, welcher den alten Levelteil aus dem Speicher entfernt und gegen einen Neuen ersetzt.

Außergewöhnlich ist jedoch, dass jede vom Spieler getätigte Aktion innerhalb der Leichenhalle als Trigger dient, welche das Abspielen einer Sounddatei verändert.

Betritt der Spieler den Raum und tätigt keinerlei Tasteneingabe, so wird lediglich alle zehn bis zwanzig Sekunden sehr leise eine Sounddatei abgespielt welche den Spieler auffordert – „Get out!“ – den Raum zu verlassen.

Mit jeder Tasteneingabe werden jedoch Häufigkeit und Lautstärke der Wiedergabe der Sounddatei erhöht. Mangels anderer Ausgänge bleibt dem Spieler nichts anderes übrig, als die Leichenhalle durch den Eingang wieder zu verlassen, durch den sie betreten wurde.

Neben der offensichtlichen spieler führenden Wirkung sorgt dieses wiederholt abgespielte Soundfile ebenfalls für eine Steigerung von Atmosphäre und Immersion.

4.3.2.8 Die Leichenhalle #2

Verlässt der Spieler die Leichenhalle findet er sich zu seiner Überraschung erneut in derselben Leichenhalle wieder. Nur, dass dieses Mal auf den Seziertischen in der Mitte drei Leichensäcke liegen.



Abbildung 34: Batman Arkham Asylum – Drei Leichensäcke in der Leichenhalle

Durch das Betreten der Leichenhalle wird darüber Hinaus via Touch-Trigger ein Checkpoint ausgelöst.

Nähert sich der Spieler den Leichensäcken wird via Touch-Trigger ein Animationsevent ausgelöst. Einer der Leichensäcke beginnt zu zucken. Dem Spieler wird somit impliziert, dass sich in den Säcken lebende Menschen befinden.

Im Folgenden öffnet der Spieler nacheinander alle drei Säcke. Jedes Öffnen triggert

Richard Winterstetter

dabei eine Cutscene. Im ersten Leichensack befindet sich die Leiche von Batmans – respektive Bruce Waynes – Vater. Der zweite Sack fordert Batmans Mutter zu Tage. Der dritte Leichensack beinhaltet Scarecrow, welcher Batman attackiert und zu Boden gehen lässt.

Hierbei ist zu erwähnen, dass die Reihenfolge der Cutscenes immer gleich bleibt. Egal in welcher Abfolge der Spieler die Leichensäcke öffnet. Das heißt, jeder Trigger der Leichensackevents frägt ab, ob und wenn ja wie viele der anderen Trigger bereits ausgelöst wurden.

Diese Events beinhalten eine große narrative Wirkung, da sie gezielt auf der bereits durch das eingeführte geometrische Paradoxon – gespiegelte Leichenhalle - verursachten verwirrenden und bedrückenden Atmosphäre aufbauen.

Die spielerführende Wirkung wird hier bewusst außen vorgelassen, da das komplette Setup dieses Segments darauf ausgelegt ist, den Spieler zu verwirren.

4.3.2.9 Showdown mit Scarecrow

Die letzte Leichensack Cutscene lässt Batman zu Boden gehen. Wenn er sich wieder aufrichtet sind die Leichensäcke verschwunden und die Leichenhalle dient als Ausgangspunkt für ein fantastisches Gebilde.



Abbildung 35: Batman Arkham Asylum – Die Leichenhalle bricht auf und wandelt sich in eine offene Geometrie.

Somit dient die dritte Leichensack Sequenz als Trigger für mehrere Events. Zum einen werden die Leichensäcke aus dem Speicher geladen, zum Anderen werden Teile des Levels ausgetauscht.

Sobald der Spieler sich der offenen Struktur nähert wird via Touch-Trigger eine weitere Zwischensequenz ausgelöst. In dieser Zwischensequenz wird neben dem Levelziel – einem Flutlichtscheinwerfer – ebenfalls die monströse Manifestation von Scarecrow eingeführt.

Richard Winterstetter



Abbildung 36: Batman Arkham Asylum – Dem Spieler wird klar vermittelt, dass es Scarecrow's Blick auszuweichen gilt.

Das Ende der Cutscene triggert einen weiteren Checkpoint.

Im folgenden Abschnitt gilt es den Flutlichtscheinwerfer zu erreichen und dabei dengleißenden Blick von Scarecrow zu vermeiden.

Nach wenigen Metern wird via Touch-Trigger eine Mauer eingerissen, welche vermeintlichen Schutz vor besagtem Blick bot.



Abbildung 37: Batman Arkham Asylum – Die Mauer bricht weg und erschwert dem Spieler das Vorankommen.

Auf narrativer Ebene trägt dieses Event zur Atmosphäre bei und setzt den Spieler weiterhin unter Druck, da es gilt das Spielverhalten an neue Umstände anzupassen.

Nach einigen Schritten versperrt eine Steinwand den Weg, welche es zu sprengen gilt. Das Auftragen des Sprengstoffs löst hierbei ein Event aus, bei dem das Spielermodell für einen Sekundenbruchteil durch das Modell von Scarecrow ausgetauscht wird.

Richard Winterstetter



Abbildung 38: Batman Arkham Asylum – Psychologische Spiele der Entwickler mit dem Spieler.

Das Zünden des Sprengstoffes triggert darauf hin ein Animationsevent, welches den sich in der Mitte des Levels befindenden Scarecrow auf das Sprengen der Wand aufmerksam macht.



Abbildung 39: Batman Arkham Asylum – Immersion durch Reaktion auf das Spielverhalten

Während das Austauschen des Spielermodells lediglich der Förderung der Atmosphäre dient und das beim Spieler etablierte Gefühl der Unruhe anspricht, zeigt die Reaktion auf das Sprengen der Wand dem Spieler, dass er Teil einer logisch agierenden Welt ist, da auf seine Aktionen reagiert wird. Dies fördert die Immersion.

Richard Winterstetter



Abbildung 40: Batman Arkham Asylum– Das Levelziel ist erreicht.

Erreicht der Spieler schließlich den Flutlichtscheinwerfer triggert die Interaktion mit selbigem eine weitere Zwischensequenz. In dieser Sequenz richtet Batman den Lichtkegel des Scheinwerfers auf Scarecrow, der Bildschirm wird in strahlendes Licht getaucht und nachdem das Licht abebbt befindet sich die Spielfigur wieder in der Leichenhalle. Der Einfluss des Gases ist abgeklungen.

4.3.3 Auswertung

Die folgende Auswertung des analysierten Spielabschnittes wird zunächst rein in Anbe tracht der Faktoren Anzahl, Differenziertheit, Spielfluss und Zusammenspiel betrachtet. Im weiteren Verlauf wird das Ergebnis in Relation mit externen Faktoren wie Entwicklungszeitraum und verwendete Engine gewertet.

Anzahl

Die Anzahl der verwendeten Scripts ist in Relation zur Länge und zum Umfang des Abschnittes hoch

Differenziertheit

Es wird mit verschiedensten Trigger Typen und Arten von Events gearbeitet. Einige Ansätze – Leichensack-Zwischensequenzen – sind innovativ und spielen geschickt mit der Wahrnehmung des Spielers.

Spielfluss

Der Spielfluss ist zu Beginn des gewählten Abschnitts dank des Einsatzes mehrerer Zwischensequenzen gestört. Im Gegensatz zur späteren Hälfte verlassen sich die Entwickler zu Beginn zu sehr auf Cutscenes um narrative Information zu vermitteln und das Gameplay wird auf Laufwege zwischen Sequenzen reduziert. Im späteren Verlauf wird dieses Manko jedoch behoben und der Spielfluss wird erhöht.

Zusammenspiel

Die Ausrichtung der Spielfigur nach manchen Sequenzen könnte verbessert werden, da

Richard Winterstetter

der Spieler zunächst in die falsche Richtung geschickt wird. Abgesehen davon funktioniert die Spielerführung einwandfrei.

Auf narrativer Ebene wird viel Information vermittelt. Neben primärem Inhalt, der zu Beginn größtenteils mit Hilfe von Zwischensequenzen erzählt wird, transportieren die Entwickler sekundäre Narration auf mehreren Ebenen und spielen mit psychologischen Grundmotiven sowie mit der durch dichte Atmosphäre aufgebauten Spielermotivation. Dem Spieler wird glaubhaft vermittelt, dass die Spielfigur unter dem Einfluss des Gases leidet und halluziniert.

Das Zusammenspiel von narrativer sowie spielerführender Wirkung ist superb und vermeidet trotz auftretendem Rollercoaster Effect negatives Feedback seitens des Spielers.

Fazit

Objektiv betrachtet befindet sich das trigger-based Scripting im analysierten Abschnitt von Batman Arkham Asylum im Vergleich zu anderen Spielen im auf hohem Niveau. Die Art der Scripts und Trigger sind abwechslungsreich und greifen ineinander. Trotz des anfänglich starken Einsatzes von Zwischensequenzen arbeiten die Entwickler mit innovativen und ausführlichen Methoden um narrative Inhalte zu vermitteln ohne den Spieler vom Geschehen auszuschließen und schaffen so ein Spielerlebnis, das zum jetzigen Zeitpunkt einzigartig ist. Mehrfach werden bekannte Motive und Mechaniken bewusst umgekehrt. Die daraus resultierenden Brüche in Logik und Konsistenz werden vom Spieler jedoch aufgrund des gewählten Settings und dem Prinzip von „Suspension of Disbelief“ nicht abgestoßen, sondern fördern das Immersionsgefühl zusätzlich.

Da das Spiel mithilfe der Unreal Engine realisiert wurde liegt die Vermutung nahe, dass die Entwickler einen Großteil der trigger-based Scripts mithilfe von Kismet umgesetzt haben.

Richard Winterstetter

5 Resümee

Im Folgenden sollen die Ergebnisse dieser Arbeit noch einmal allgemein festgehalten werden.

Trigger-based Scripting begann als eleganter Weg Performanzprobleme zu kaschieren und wandelte sich im Laufe der Zeit zu einem elementaren Träger von Spielerführung und Narration. Richtig angewandt ist es essentiell für die Interaktion mit dem Spieler ohne auf etabliertes Vokabular aus dem Medium Film zurückgreifen zu müssen.

Stellt man das eigentliche Gameplay jedoch zu Gunsten des Scriptings in den Hintergrund entsteht beim Spieler leicht das Gefühl ohne eigne Teilnahme durch festgelegte Events gezogen zu werden. Darüber hinaus ist exzessives Scripting weniger für non-lineare Spiele geeignet, da das Auslösen von Triggern in einer freien Spielwelt nur bedingt gewährleistet werden kann. In linearen Spielen entfaltet trigger-based Scripting jedoch sein volles Potenzial.

Scripting-Tools wie Kismet erlauben es dabei zunehmend, programmier-unerfahrenen Entwicklern exzessives und komplexes Scripting zu erstellen.

Mit Hilfe von trigger-based Scripting lassen sich spielerführende und narrative Informationen geschickt und umfangreich vermitteln um einen anhaltenden und konsistenten Dialog mit dem Spieler zu führen.

Nichts desto trotz ist Scripting ein aufwändiges Feld, weswegen kleinere Entwickler-teams es nur begrenzt und im Rahmen ihrer Möglichkeiten nutzen können. Etablierte Studios jedoch legen zunehmend ihren Fokus auf die ausführliche Nutzung von trigger-based Scripting zur Vermittlung von spielerführender und narrativer Information.

Richard Winterstetter

6 Ausblick

Was also ist trigger-based Scripting?

Zum Einen: Die Zukunft!

Programme wie Kismet erlauben es Level Designern ohne jegliche Programmiererfahrung komplexe Scripts zu erstellen. Das Anfertigen dieser aufwändigen Komplexe erfordert immer mehr Zeit. Deswegen gehen immer mehr Entwickler dazu über spezielle Script Designer einzustellen, welche ausschließlich für das Erstellen und Platzieren von Scripts zuständig sind und sich speziell in diesem Bereich fortbilden und einbringen können.

Es können neue Wege ausgelotet werden wie und wo man trigger-based Scripting benutzt. Die virtuellen Welten werden zunehmend spielerbezogen und intensiver, und bieten dem Spieler somit ein Spielerlebnis weit immersiver und dichter als noch vor ein paar Jahren.

Die Branche kratzt gerade erst an den möglichen Einsatzarten von trigger-based Scripting. Spiele wie Batman Arkham Asylum rechtfertigen den Mut Neues auszuprobieren mit finanziellem Erfolg und zunehmend wenden sich Entwickler von Cutscenes ab und dem narrativen Einsatz von Scripting zu. Bei vielen kommenden Spielen erkennt man klar den Trend zu immer komplexeren Abläufen von Scripts.

Aber es ist nicht alles Gold was glänzt!

Die steigende Popularität von trigger-based Scripting führt zu einem Anstieg von linearen Spielen wie den Uncharted oder Call of Duty Reihen. Diese Spiele, perfekt durchgescriptet wie sie sind, verzichten auf grundlegende Elemente von Spielen wie beispielsweise Exploration. Diese werden einer dichteren und spektakuläreren Atmosphäre geopfert.

So essentiell und bahnbrechend Scripting für das interaktive Vermitteln von Narration ist, die exzessive Nutzung birgt immer die Gefahr, dass eigentlich interaktive Element des Videospiels, das Spiel, in den Hintergrund zu drängen und eine Reihe von perfekt inszenierten aber spielerisch flachen Schießbuden und Achterbahnen hervorzubringen.

Wie schon bei so vielen in dieser Arbeit besprochenen Aspekten des Scriptings gilt es auch beim Scripting selbst gekonnt eine Balance zu halten.

Trigger-based Scripting mag ein Alleinstellungsmerkmal von Videospielen sein, welches auf keinem vordefinierten Vokabular aufbaut und das es auszuloten gilt um das Medium voranzutreiben. So sehr man als Entwickler den Spieler auch führen kann und will, umso wichtiger ist es nicht zu vergessen, dass das was das Medium Videospiel von den anderen abhebt, die Interaktivität, die Freiheit in eine virtuelle Welt einzutauchen und diese zu erforschen ist!

Richard Winterstetter

Literaturverzeichnis

- [Bar04] Richard A. Bartle. „*Designing Virtual Worlds*“. New Riders, Indianapolis, 2004. ISBN-10: 9780131018167
- [Bol09] Jens Bolanz. „*Spielerführung in dem 3D-Action Adventure Phobos*“. Bachelorarbeit im Fachbereich Gamedesign, Mediadesign Hochschule München, 2009.
- [Cag08] David Cage. „*Dreaming of a new day: David Cage's Heavy Rain*“. Online-Präsentation, Gamasutra.com, 2008.
http://www.gamasutra.com/view/feature/3744/dreaming_of_a_new_day_heavy_.php?page=1
- [Epi11] Epic Games Inc. „*Unreal Development Kit Licensing*“. Online-Präsentation, 2011. <http://www.udk.com/licensing>
- [Far11] Farlex Inc. „*The free dictionary*“. Online-Präsentation, 2011.
<http://www.thefreedictionary.com>
- [Fil11] FileMaker Inc. „*Understanding and using Script Triggers*“. Online Präsentation, 2011.
http://help.filemaker.com/app/answers/detail/a_id/7465/~/undundundu-and-using-script-triggers
- [Fit09] Lisa Fitzpatrick. „*The Art of Avatar*“. Abrams, New York, 2009. ISBN: 978-0-8109-8286-4
- [Gru07] Sebastian Grünwald. „*Methoden interaktiven Storytellings*“. Eigenverleger, Winhöring, 2007. ISBN: 978-3-00-021978-8
- [Lan03] Langenscheidt. „*Longman – Dictionary of Contemporary English*“. Pearson Education Limited, Harlow, 7. Auflage, 2003. ISBN: 3-526-50820-8
- [Mor09] Grant Morrison. „*Arkham Asylum: Ein düsteres Haus in einer finsternen Welt*“. Panini Comics, Nettetal-Kaldenkirchen, 2009.
- [Ner11] Martin Nerurkar. „*Experiencing Environments*“. Online-Präsentation, 2011.
<http://www.gamearch.com/2011/04/16/experiencing-environments-pdf-download/>
- [Nin11] Nintendo of Europe GmbH. „*Super Nintendo – Technische Details*“. Online Präsentation, 2011.
http://www.nintendo.de/NOE/de_DE/systems/technische_details_1111.html
- [Per09] David Perry, Rusel DeMaria. „*David Perry on Gamedesign – A Brainstorming Toolbox*“. Course Technology, Boston, 1. Auflage, 2009. ISBN-10: 1-58450-668-7

**Trigger-based Scripting:
Narrative und spielerführende Wirkung**
Bachelorarbeit im Fachbereich Gamedesign

Richard Winterstetter

- [Sal10] Katie Salen, Eric Zimmerman. „*Rules of Play*“. Massachusetts Institute of Technology, 6. Auflage, 2010. ISBN-10: 9780262240451
- [Son11] Sony Computer Entertainment Inc. „Corporate History“. Online-Präsentation, 2011.
http://www.scei.co.jp/corporate/history_e.html
- [Tec11] TechTerms.com. „*Script*“. Online-Präsentation, 2011.
<http://www.techterms.com/definition/script>
- [Tol05] J.R.R. Tolkien. „*Lord of the Rings: The fellowship of the Ring*“. Harpercollins UK, Auflage: 50th anniversary, 2005. ISBN-10: 0007203543

Abbildungsverzeichnis

Abbildung 1: Turok 2: Seeds of Evil [Igu98] zählt bis heute zu den Spielen mit den komplexesten und verwinkeltesten Leveln.....	7
Abbildung 2: Half Life 2 Episode One [Val06] – Die Half Life Reihe von Entwickler Valve erzählt eine komplexe Geschichte ohne die Hilfe von Cutscenes.....	8
Abbildung 3: Trigger-Event Verknüpfung – schematische Darstellung.....	10
Abbildung 4: Automatisierte Schiebetür – schematische Darstellung	12
Abbildung 5: UDK Editor – Aufsetzen der Szene	13
Abbildung 6: UDK Editor– Links: Festlegen der Objekteigenschaften für die Interaktion mit Matinée	13
Rechts: Hinzufügen eines Trigger-Actors	13
Abbildung 7: UDK Kismet – Links: Spezifizierung des Triggers	14
Rechts: Finale schematische Darstellung in Kismet.....	14
Abbildung 8: UDK Editor: Finales Setup	14
Abbildung 9: Trigger-Event Verknüpfung samt Trigger – schematische Darstellung .	17
Abbildung 10: „Druckschalter“ und „Volumen“-Trigger – schematische Darstellung	18
Abbildung 11: Half Life 2 [Val04] – Die Citadel fungiert immer wieder als Referenzpunkt um dem Spieler ein Gefühl dafür zu geben, wo er sich gerade in der Spielwelt befindet.	21
Abbildung 12: Dead Space [Vis08] nutzt unter Anderem geschicktes Lighting um den Spieler durch die dunklen Level zu führen.	22
Abbildung 13: Team Fortress 2 [Val07] – mehrere blaue Assets weisen den Weg zur Basis des blauen Teams.	22
Abbildung 14: The Path [Tal09] – Je weiter sich der Spieler vom Weg entfernt, desto lauter schlägt das Herz der Spielfigur.....	23
Abbildung 15: Parasite Eve II [Squ00] – Der umgestürzte Tisch und die Verwüstung im Restaurant deuten an, dass hier ein Kampf stattgefunden hat.	25
Abbildung 16: Super Metroid [Nin94] –Die in den komplexen Levels verteilten Checkpoints werden von Spielern oft aktiv in die Routenplanung mit einbezogen.	26
Abbildung 17: Assassin’s Creed Brotherhood [Ubi10] – Links: Eine vermeintliche Weggabelung. Rechts: Der „falsche“ Weg wird versperrt.	28
Abbildung 18: Final Fantasy 7 [Squ97] –Wie für die Final Fantasy Reihe üblich wird die Geschichte in regelmäßigen Abständen via FMV Sequenzen eindrucksvoll weitergesponnen.	30
Abbildung 19: Half Life 2: Episode 1 – Mit der Einführung des NPC Charakters Alyx wurden die trigger-based NPC Monologe besser in den Spielfluss integriert.	32
Abbildung 20: Uncharted 2: Among Thieves führt den Spieler durch komplexes und umfangreiches Scripting mithilfe von schlauchartige Level.	33
Abbildung 21: TINK – Establishing Cutscene samt gescripteten Bienenflug	36
Abbildung 22: TINK – Tutorialschaf #1	37
Abbildung 23: TINK – Ein schwimmendes Fass	37
Abbildung 24: TINK – Tutorialschaf #2	38
Abbildung 25: TINK – Tutorialschaf #3	39
Abbildung 26: TINK – Die Libelle weist den Weg zum finalen Levelabschnitt	39
Abbildung 27: TINK – Tutorialschaf #4	40

Richard Winterstetter

Abbildung 28: Batman Arkham Asylum – Der Aufzug.....	42
Abbildung 29: Batman Arkham Asylum – Commissioner Gordon	43
Abbildung 30: Batman Arkham Asylum – Commissioner Gordon ist tot.....	44
Abbildung 31: Batman Arkham Asylum – Funkanimation	44
Abbildung 32: Batman Arkham Asylum – Fliesen fallen von den Wänden	45
Abbildung 33: Batman Arkham Asylum – Der Silhouettenschatten wird zunehmend zum zentralen spielerführenden Element.....	45
Abbildung 34: Batman Arkham Asylum – Drei Leichensäcke in der Leichenhalle	46
Abbildung 35: Batman Arkham Asylum – Die Leichenhalle bricht auf und wandelt sich in eine offene Geometrie.....	47
Abbildung 36: Batman Arkham Asylum – Dem Spieler wird klar vermittelt, dass es Scarecrow's Blick auszuweichen gilt.....	48
Abbildung 37: Batman Arkham Asylum – Die Mauer bricht weg und erschwert dem Spieler das Vorankommen.....	48
Abbildung 38: Batman Arkham Asylum – Psychologische Spiele der Entwickler mit dem Spieler.....	49
Abbildung 39: Batman Arkham Asylum – Immersion durch Reaktion auf das Spielverhalten	49
Abbildung 40: Batman Arkham Asylum– Das Levelziel ist erreicht.....	50

Richard Winterstetter

Verzeichnis: Erwähnte Software

- [Cap96] Capcom. „*Resident Evil*“. Capcom/Virgin Interactive, 1996.
- [Epi09] Epic Games. „*Unreal Development Kit*“. Epic Games, 2009.
- [Igu98] Iguana Entertainment. „*Turok 2: Seeds of Evil*“. Acclaim Entertainment, 1998.
- [Inf03] Infinity Ward. „*Call of Duty*“. Activision, 2003.
- [Mim11] Mimimi Productions Gbr. „*TINK*“. Mediadesign Hochschule München, 2011.
- [Nau09] Naughty Dog. „*Uncharted 2: Among Thieves*“. Sony Computer Entertainment, 2009.
- [Nin89] Nintendo, Gunpei Yokoi. „*Super Mario Land*“. Nintendo, 1989.
- [Nin94] Nintendo R&D1. „*Super Metroid*“. Nintendo, 1994.
- [Roc09] Rocksteady Studios. „*Batman Arkham Asylum*“. Eidos Interactive / Square-Enix, 2009.
- [Squ87] Squaresoft. „*Final Fantasy*“. Squaresoft, 1987.
- [Squ97] Squaresoft. „*Final Fantasy VII*“. Squaresoft, 1997.
- [Squ00] Squaresoft. „*Parasite Eve II*“. Squaresoft, 2000.
- [Tal09] Tale of Tales. „*The Path*“. Tale of Tales, 2009.
- [TBo11] Team Bondi. „*L.A. Noire*“. Rockstar Games, 2011.
- [Ubi10] Ubisoft Montreal. „*Assassin's Creed Brotherhood*“. Ubisoft, 2010.
- [Uni11] Unity Technologies. „*Unity Engine 3*“. Unity Technologies, 2011.
- [Val04] Valve Corporation. „*Half Life 2*“. Vivendi Universal, 2004.
- [Val06] Valve Corporation. „*Half Life 2 Episode One*“. Electronic Arts, 2006.
- [Val07] Valve Corporation. „*Team Fortress 2*“. Electronic Arts, 2007.
- [Vis08] Visceral Games. „*Dead Space*“. Electronic Arts, 2008.