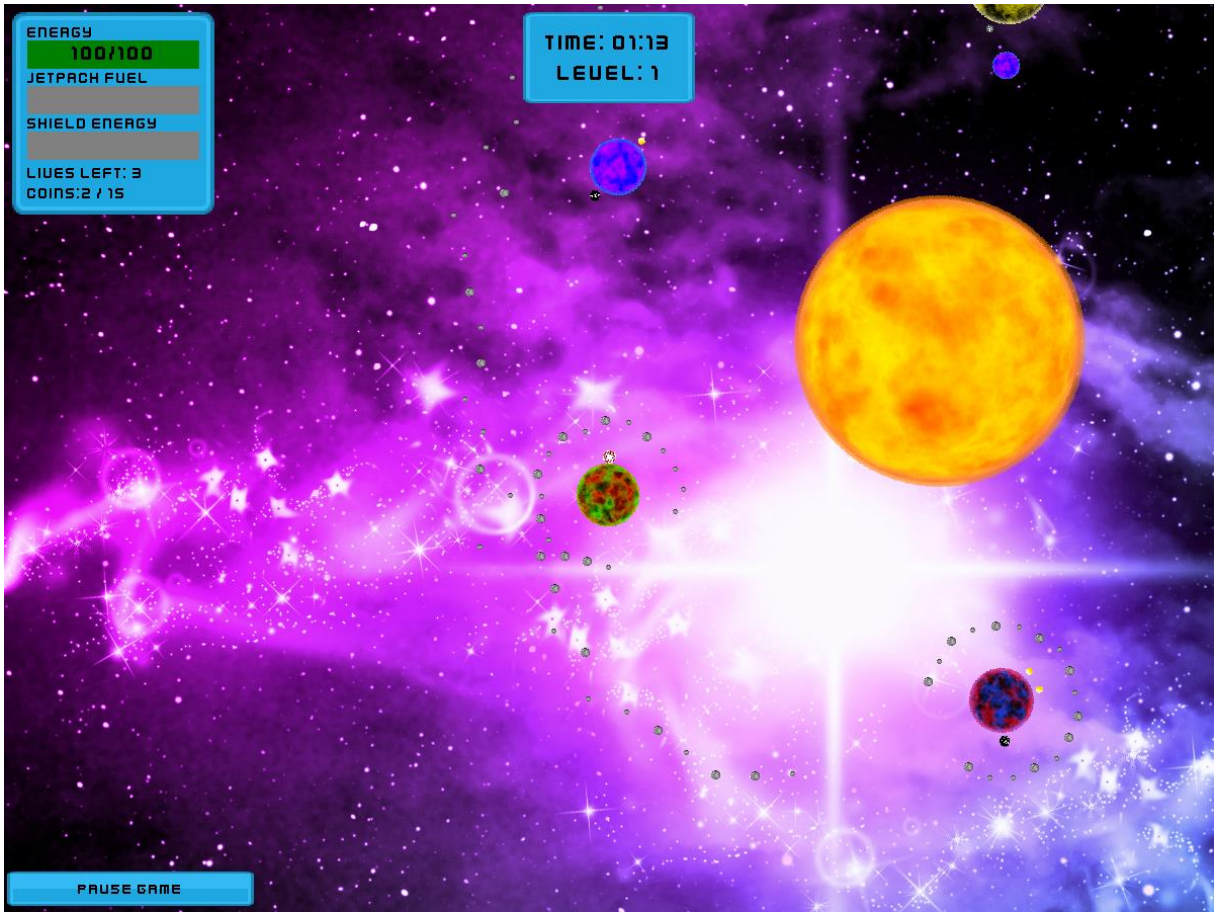
Gymnasium Kirchenfeld
Abteilung für Mathematik und Naturwissenschaften

Gravity

Entwerfen und Programmieren einer eigenen Spielidee



Verfasser:
Michael Marti

Betreut durch:
Stefan Rothe

Bern, November 2014

Inhaltsverzeichnis

1 Einleitung

Computerspiele sind längst nicht mehr Produkte für ein beschränktes Zielpublikum, sondern haben sich dank ihrem wirtschaftlichen Erfolg als Massenmedium behaupten können. Die Umsetzung der Spiele erfolgt in Teams von teilweise mehr als 100 Programmierern und Designern; die Entwicklung der Computerspiele kann mehrere Jahre dauern. Da der Kauf und Austausch von digitalen Produkten sich grosser Beliebtheit erfreut, man denke an den App Store und Google Play, eröffnet sich auch ein neuer Markt für Entwickler von Videospielen. Dank dieser neu geschaffenen Basis, sind Ein-Mann-Projekte in der kommerziellen Videospielentwicklung wieder möglich, genauso wie zu Beginn der Videospielära. Jeder kann seine Idee umsetzen und dank Einrichtungen wie eben zum Beispiel Google Play dazu noch ein grosses Publikum erreichen.

Im Rahmen meiner Maturaarbeit ist das Videospiel „Gravity“ entstanden. „Gravity“ ist ein Ein-Mann-Projekt. Ich habe den Weg von der Idee bis zum fertigen Produkt verfolgt. Gesagt sei, dass ich keine kommerziellen Absichten verfolge, zudem wurde das Spiel nicht für Smartphones entwickelt, sondern für einen Computer. Eine allfällige Portierung für mobile Geräte mit Touchscreen-Steuerung wäre jedoch möglich.

Die Entwicklung eines Videospiels verläuft in mehreren Phasen. Am Anfang eines Projektes steht ein Konzept. Die Hauptmechanik in meinem Spiel beruht auf den Newtonschen Gesetzen der Gravitation. Im Spiel werden Sonnensysteme mit mehreren Planeten simuliert. Die Aufgabe des Spielers ist es, sich zwischen den Planeten durch Springen fortzubewegen und so alle im Sonnensystem verteilten Münzen zu sammeln und damit den nächsten Level erreichen zu können. Bei der Entwicklung musste immer wieder abgewogen werden, wie realitätsnah eine Simulation sein darf, ohne den rechnerischen Aufwand zu gross werden zu lassen. Zusätzlich darf die Spielbarkeit und der Spielspass nie auf Kosten des Realismus zu stark vernachlässigt werden. Wichtige Design-Entscheidungen, wie der Grafikstil und die Level-Generierung, mussten getroffen werden. Nach der Realisierung des Projektes wurde das Spiel von Testpersonen gespielt. Zusätzlich erhielten die Testpersonen frei zugängliche Spiele mit ähnlichem Spielprinzip, um diese mit Gravity vergleichen zu können. Nur durch Rezensionen erhält ein Entwickler aussagekräftige Korrekturvorschläge, die der Verbesserung des Spiels dienen können. Die Testergebnisse sind analysiert und in meiner Arbeit zusätzlich verarbeitet worden.

Von der Konzeption bis zur abschliessenden Analyse und Bewertung durch Probanden, sind in meiner Maturaarbeit alle Phasen der Entwicklung meines Spiels „Gravity“ dokumentiert worden.

2 Theoretische Grundlagen

2.1 Newtonsche Mechanik

Sir Isaac Newton machte sich, wie bereits viele Wissenschaftler vor ihm, Gedanken über die Ursache von Bewegungen. Newton kam zum Schluss, dass Körper sich ohne Einwirkung einer Kraft stets gradlinig und gleichförmig durch den Raum bewegen. Anders ausgedrückt: Wirkt keine Kraft auf einen Körper, so bewegt sich dieser mit konstanter Geschwindigkeit in eine Richtung. Wirken aber eine oder mehrere Kräfte auf den Körper, so wird dieser beschleunigt. Unter einer Beschleunigung versteht man in der Physik nicht nur das Schnellerwerden eines Körpers, sondern auch eine Abbremsung oder eine Änderung in der Bewegungsrichtung. Die Beschleunigung eines Körpers beschrieb Newton mit folgender Formel:

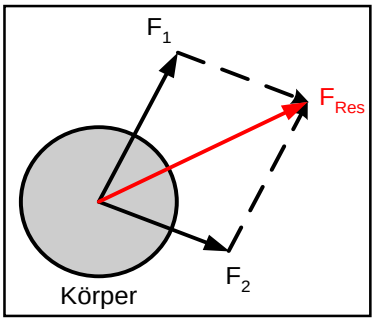


Abbildung 1 Beispiel einer Vektoraddition

$$F_{Res} = m \cdot a$$ [1]

Unter F_{Res} versteht man die resultierende Kraft, also die Summe aller auf den Körper wirkenden Kräfte. Kräfte sind vektorielle Grössen. Das heisst, Kräfte haben nicht nur einen Betrag, sondern auch eine Richtung. Beim Addieren muss die Richtung zusätzlich mit einberechnet werden. Eine grafische Lösung dieses Problems ist in Abbildung 1 ersichtlich. Das Prinzip der Vektoraddition funktioniert auch in umgekehrter Richtung. Dieser Zusammenhang wird mit Hilfe der Abbildung 2 erläutert. Eine Kraft, in diesem Beispiel *F*, kann auch in zwei Teilkomponenten aufgespalten werden, nämlich F_1 und F_2. Dieser Aspekt vereinfacht das Zusammenzählen vieler einzelner Kraftvektoren. Wenn man jede Kraft in zwei Kraftkomponenten aufspaltet, deren Richtung bekannt ist, kann man deren Beträge addieren, ohne auf die Richtung Rücksicht nehmen zu müssen. Im zweidimensionalen Raum eignen sich der „vertikale“ und der „horizontale“ Einheitsvektor. Jede Kraft wird in ihre x- und y-Komponente aufgespalten.

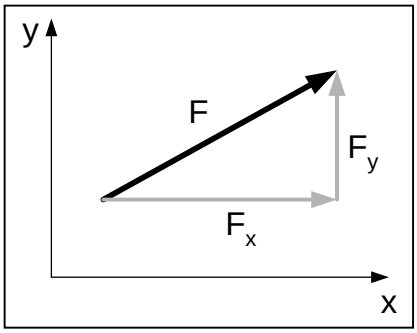


Abbildung 2 Aufspaltung eines Vektors in Teilkomponenten

2.2 Gravitationskraft

Massen ziehen sich an. Zu dieser Erkenntnis soll Isaac Newton gekommen sein, als ihm ein Apfel vor die Füsse fiel. Die Erde übt immerwährend eine Kraft auf andere Massen aus. Diese sorgt dafür, dass Gegenstände auf der Erde auf den Boden fallen. Mit seiner Theorie konnte Newton auch die Bewegungen von Himmelskörpern voraussagen und berechnen. Auch wenn Newtons Gravitationstheorie durch Einsteins Allgemeine Relativitätstheorie präzisiert wurde, reicht ihre Genauigkeit aus, um ein Sonnensystem für ein Spiel zu simulieren.

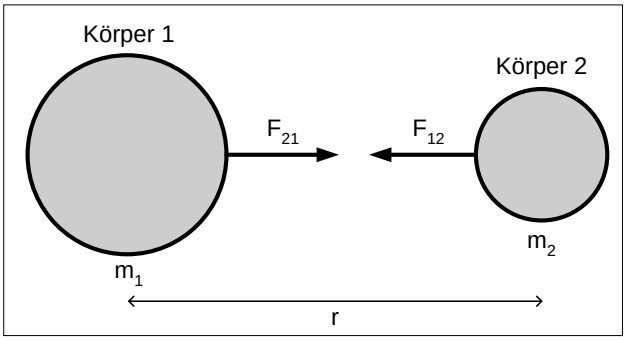


Abbildung 3 Schematische Darstellung der Gravitationskraft

[1] (DMK, DPK, 2011) S. 157

Die Formel lautet wie folgt:

$$|F_G| = \mathrm{G} \cdot \frac{\mathrm{m}_1 \cdot \mathrm{m}_2}{\mathrm{r}^2} = |F_{12}| = |F_{21}| \ ^{2}$$

Die Formel beschreibt, welche Kraft ein Körper 1 mit der Masse m_1 auf den Körper 2 mit der Masse m_2 im Abstand *r* ausübt. Aus der Formel geht hervor, dass die Gravitationskraft umso grösser ist, je massenreicher die Körper sind. Die Kraft nimmt umgekehrt-proportional zum Quadrat des Massenmittelpunktabstandes ab. Die Gravitationskonstante G sorgt dafür, dass das Resultat der SI-Einheit für Kraft, nämlich Newton entspricht. Die Gravitationskraft ist eine Wechselwirkungskraft: Körper 1 wirkt auf Körper 2 die gleiche Kraft aus, wie Körper 2 auf Körper 1. Das Resultat ergibt jedoch nur den Betrag der Kraft. Die Gravitationskraft wirkt entlang der Geraden durch die Massenmittelpunkte der Körper. Im Gegensatz zu den anderen fundamentalen Wechselwirkungen wirkt die Gravitation ausschliesslich anziehend.

2.3 Abbildungsgleichungen in der Ebene

Das Programm simuliert Sonnensysteme in einem statischen, kartesischen Koordinatensystem. In einem kartesischen Koordinatensystem werden Punkte durch ihre spezifischen *x*- und *y*-Werte beschrieben. Punkte in diesem Koordinatensystem werden manipuliert, um so das Bildschirmkoordinatensystem zu schaffen. Das Programm stellt nur Punkte im Bildschirmkoordinatenfeld dar, das somit als Spielausschnitt angesehen werden kann. Das heisst, dass die Berechnung der Bewegung von Himmelskörpern und deren Darstellung in Gravity zwei getrennte Schritte sind und in anderen Koordinatensystemen stattfinden.

Die Koordinaten der Spielfigur werden so manipuliert, dass die Spielfigur inmitten des Spielausschnittes fixiert wird. Alle anderen Himmelskörper werden um den gleichen Vektor verschoben wie die Spielfigur. Als erste Abbildungsgleichung ergibt sich die Translation, also eine Verschiebung des Punktes P mit den Koordinaten $(x_P|y_P)$ um den Vektor $\vec{v}$. Resultat ist der Punkt *P'* (Abbildung 4).

$$x_{P'} = x_P + x_{\vec{v}} \ ^{3}$$

$$y_{P'} = y_P + y_{\vec{v}}$$

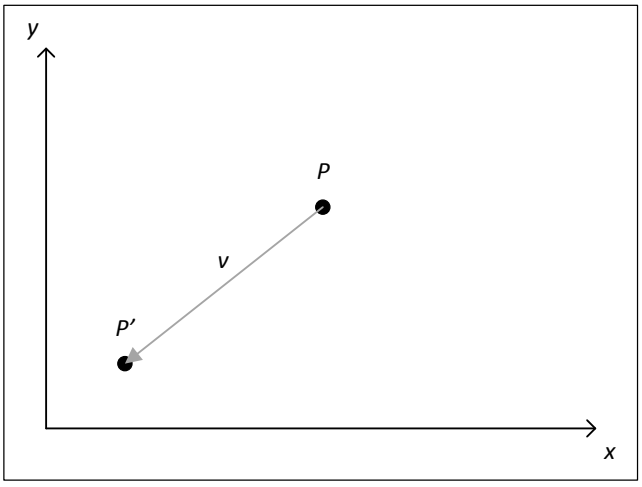


Abbildung 4 Translation um den Vektor $\vec{v}$.

[2] (DMK, DPK, 2011): S. 161

[3] (DMK, DPK, 2011): S. 112

Weiter soll es möglich sein, den Spielausschnitt zu skalieren. Dazu wird eine zentrische Streckung durch den Ursprung durchgeführt. Soll der Punkt *P* um dem Faktor *k* zentrisch gestreckt werden, ergeben sich für die Koordinaten des Punktes *P'* folgende Gleichungen (Abbildung 5):

$$x_{P'} = k \cdot x_P \quad [4]$$

$$y_{P'} = k \cdot y_P$$

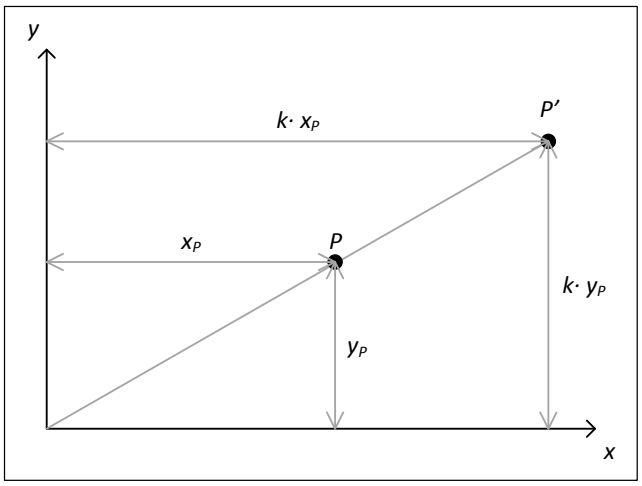


Abbildung 5 Zentrische Streckung um den Faktor *k*.

Zuletzt verlangt das Spieldesign, dass alle Himmelskörper um einen beliebigen Punkt im Koordinatensystem rotiert werden können. Ausgang der Rotation sind immer die Koordinaten der Spielfigur Punkt C mit den Koordinaten: $(x_C|y_C)$. Der Punkt *P* soll um den Winkel α gedreht werden, um den Punkt *P'* zu erhalten (Abbildung 6).

$$x'_P = x_C + \cos\alpha \cdot (x_P - x_C) - \sin\alpha \cdot (y_P - y_C) \quad [5]$$

$$y'_P = y_C + \sin\alpha \cdot (x_P - x_C) + \cos\alpha \cdot (y_P - y_C)$$

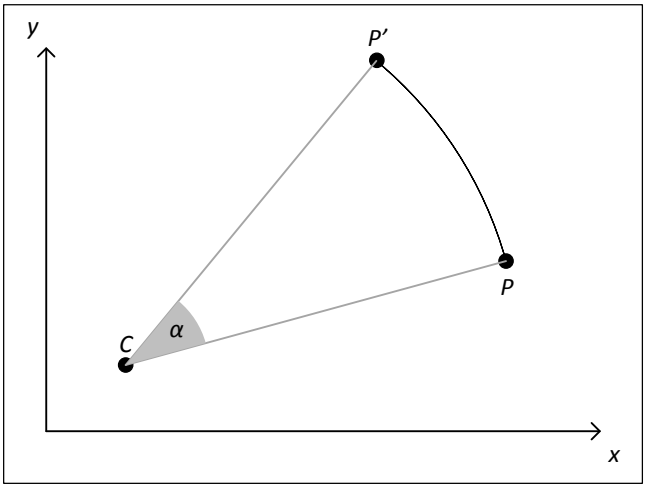


Abbildung 6 Rotation des Punktes *P* um den Punkt *C* um den Winkel α.

Das Programm führt alle diese Abbildungsgleichungen für jeden dargestellten Himmelskörper durch. Die Reihenfolge spielt hierbei keine Rolle.

[4] (DMK, DPK, 2011): S. 112

[5] Siehe: I Beweis der Rotationsfunktion um einen beliebigen Punkt

3 Die grundlegende Spielidee

3.1 Spielidee

Der Spieler findet sich auf einem Planeten in einem Sonnensystem wieder, indem eine gewisse Anzahl Planeten um die Sonne kreisen. Des Spielers Hauptziel ist es, die überall auf den Planeten verteilten Münzen einzusammeln. Erst wenn alle Münzen eines Levels eingesammelt worden sind, gilt der Level als bestanden.

Der Spieler kann auf dem Planeten umhergehen und von ihm abspringen, um in den interplanetaren Raum zu gelangen. Doch der Weltraum ist keinesfalls harmlos – im Gegenteil – um die Aufgabe zu erschweren, kann der Spieler durch verschiedene Umstände von seinem Ziel abgebracht werden. Der Spieler hat pro Versuch nur eine bestimmte Anzahl Leben, die ihm durch Fallenstellungen wie Asteroidenschläge oder durch die Strahlung und Hitze der Sonne genommen werden.

Es soll jedoch nicht nur Hindernisse geben, sondern auch Hilfen, die das gezielte Reisen zwischen Planeten vereinfachen. So gibt es Bonusgegenstände, wie zum Beispiel ein Raketenantrieb, der eine gezielte Beschleunigung im Weltraum ermöglicht. Wurde der Level geschafft, errechnet das Spiel eine Punktzahl, die die Fertigkeiten des Spielers widerspiegeln soll. Diese Punktzahl soll den Spieler motivieren, weiterzuspielen und sich zu verbessern.

3.2 Hindernisse

Ein Spiel ist nur interessant, wenn eine Balance zwischen Scheitern und Erfolg besteht. Weder zu einfache Spiele, die den Spieler schnell langweilen, noch schwierige, die nur zu Frusterlebnissen führen, können überzeugen. Es genügt also nicht, dass der Spieler nur von Planet zu Planet hüpfen kann. Wichtig ist, dass er auch noch andere Spielelemente mit einberechnen muss, um zum Ziel zu gelangen.

Um einen Planeten herum und auch zwischen zwei Planetenbahnen kann ein Asteroidengürtel generiert werden (Abbildung 7). Sobald die Spielfigur durch den Asteroidengürtel fliegt, verliert sie an Leben. Als Erleichterung hat jeder Asteroidengürtel eine Lücke, die frei von Asteroiden ist. Ziel ist es, einen Absprung zeitlich so zu planen, dass die Spielfigur möglichst keinen Schaden nimmt und dabei trotzdem in die gewünschte Richtung abspringt.

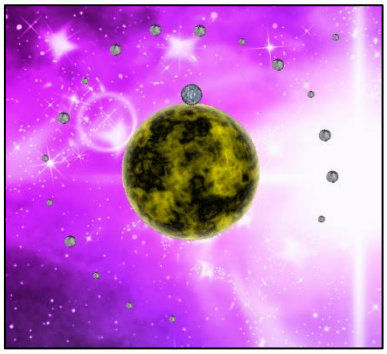


Abbildung 7 Der gezeigte Planet ist von einem Asteroidengürtel umgeben

Gerät die Spielfigur in Sonnennähe, verliert diese Lebensenergie. Fliegt sie sogar in die Sonne, stirbt sie sofort. Um ewiges Kreisen in einem Orbit zu vermeiden, ist die Zeit, in der sich die Spielfigur im Weltraum ohne Zwischenlandung bewegen kann, limitiert. Wird das Zeitlimit überschritten, verliert sie ein Leben. Mit dieser Absicherung kann vermieden werden, dass der Spieler für eine längere Zeit überhaupt keinen Einfluss mehr auf das Spielgeschehen hat.

Ein eingebauter Dämpfer teilt einen Planeten in zwei Hälften. Läuft die Spielfigur in den Dämpfer hinein, wird sie davon abprallen. Eine ganze Umrundung des Planeten wird so verunmöglicht. Dies hat somit zwei Konsequenzen zur Folge. Zum einen kann der Spieler nicht mehr beliebig Anlauf holen, um einen Absprung zu wagen. Zum anderen kann man die andere Seite eines Planeten nur noch über Umwege erreichen.

3.3 Bonusgegenstände

Dem Spieler soll der Weg zum Ziel mit Bonusgegenständen erleichtert werden. Bonusgegenstände haben teilweise einen positiven Einfluss auf die Steuerbarkeit der Spielfigur. Ein solcher Bonusgegenstand ist zum Beispiel der Supersprung, der die Sprungkraft verdoppelt. Wegen der hohen Sprungbeschleunigung verringert sich die Wirkung der Gravitationskraft auf die Bewegungsrichtung der Spielfigur zunächst. Jedoch birgt der verstärkte Sprung auch Gefahren. Wird das Ziel verfehlt, kann die Spielfigur das Sonnensystem verlassen, was ihr nach Ablauf des Zeitlimits ein Leben kosten wird.

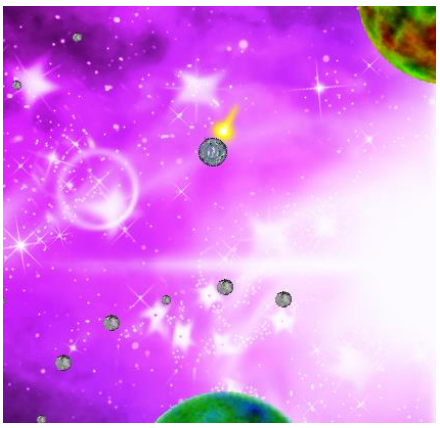


Abbildung 8 Die Spielfigur bewegt sich mit dem Raketentriebwerk durch den Weltraum.

Mit dem Raketenrucksack kann der Spieler auch während dem Aufenthalt im Weltraum aktiv auf die Bewegungsrichtung der Spielfigur Einfluss nehmen (Abbildung 8). Jedoch reicht die Schubkraft des Raketenrucksackes nicht immer aus, die Wirkung der Gravitationskraft in alle Richtungen zu kompensieren.

Der Schild bietet zusätzlichen Schutz vor Asteroiden und der Sonnenstrahlung. Doch auch dieser Schild vermag es nicht, die Spielfigur stets vor Gefahren zu schützen. Seine Energie ist limitiert und wird beim Gebrauch geschmälert. Gesundheitspakete, die auf den Planeten verteilt sind, heilen den bereits erlittenen Schaden.

3.4 Darstellung des Spieles

Die einzelnen Positionen werden so angepasst, dass sich die Spielfigur immer in der Mitte des Programmausschnittes befindet. Der erste Schritt besteht also darin, die ganze Level-Ebene so zu verschieben, dass die Position der Spielfigur fixiert scheint. Anschliessend wird diese Ebene so rotiert, dass die Spielfigur sich immer auf der oberen Planetenhälfte des auf dem Bildschirm dargestellten Spielausschnittes befindet. Abhängig von dem gewünschten Zoom-Faktor werden die Koordinaten noch massstäblich verändert.

4 Physikalische Näherungen

4.1 Näherungen bei der Berechnung der Gravitationskraft

Um eine wirklich realistische Gravitationssimulation zu programmieren, müsste man per Definition die Anziehungskraft zwischen allen Objekten, die Masse besitzen, berechnen. Somit nimmt der Rechenaufwand stark zu, wenn man viele Objekte simulieren will. Da zwischen jeder möglichen Kombination von zwei Objekten die daraus entstehende Gravitationskraft berechnet werden muss, ergibt sich für *n* Objekte folgende Anzahl Rechenschritte S: $S = \frac{1}{2}(n^2 - n)$. Doch ist dieser Aufwand gar nicht nötig, wie die Realität zeigt. Die Erde und alle anderen Planeten besitzen eine stabile Umlaufbahn. Zudem bewegen sich Planeten auf einer nahezu kreisrunden Bahn. Somit lässt sich die Ausgangssituation idealisieren: Die Planeten haben ein nahezu kreisrundes Orbit und beeinflussen sich gegenseitig in ihrer Umlaufbahn während einer kurzen Zeitspanne kaum. Als letztes muss noch betrachtet werden, dass ein Planet sich nicht um den Sonnenmittelpunkt, sondern um den gemeinsamen Schwerpunkt dreht. Ist ein Objekt jedoch viel massereicher als das andere, lässt sich vereinfacht sagen, dass das leichtere Objekt sich um den Masseschwerpunkt des massereicheren Objektes bewegt. Mit all diesen Vereinfachungen wird es möglich, die Situation neu zu formulieren. Die Kraft, die ein Objekt auf einer Kreisbahn hält, ist die Zentripetalkraft.

$$F_z = m \cdot \frac{v^2}{r}$$ [6]

Die Gravitationskraft funktioniert in dieser Situation als Zentripetalkraft. Es gilt die Gleichung: $F_G = F_Z$ (Abbildung 9). Da der Radius der Kreisbahn bekannt ist, muss nur noch berechnet werden, mit welcher konstanten Geschwindigkeit sich das Objekt um sein Zentrum bewegt. Löst man die Gleichung nach *v* auf erhält man:

$$G \cdot \frac{m_{Zentrum} \cdot m}{r^2} = \frac{m \cdot v^2}{r} \Leftrightarrow v = \sqrt{\frac{m_{Zentrum} \cdot G}{r}}$$

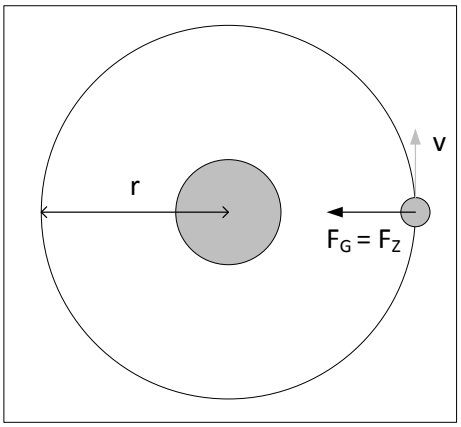


Abbildung 9 Grafische Darstellung der Situation

Die Geschwindigkeit ist jedoch für die Berechnung der Position eine umständliche Grösse. Mit der Winkelgeschwindigkeit zu rechnen macht in diesem Fall mehr Sinn, denn mit Hilfe der Trigonometrie lässt sich die Position des Planeten viel einfacher bestimmen. Eine volle Umdrehung entspricht: $r \cdot 2\pi$. Teilt man dies durch die Geschwindigkeit, erhält man die Winkelgeschwindigkeit. Diese gibt an, wie gross der Winkel ist, den das Objekt pro Sekunde überstreicht.

Aus diesen Überlegungen geht zudem hervor, dass die Umlaufgeschwindigkeit nur von der Distanz zum Zentrum und dessen Masse abhängt. Dies gilt jedoch nur, wenn die Masse des Zentrums ungleich grösser als jene des umkreisenden Objektes ist.
Der Spieler wird natürlich von dieser Vereinfachung ausgeschlossen. Die Spielfigur ist das einzige Objekt, bei der eine vollständige Berechnung erfolgt. Somit kann auch der Rechenaufwand in Grenzen gehalten werden. Die Gravitationskonstante *G* hat in der Realität einen Wert von: $G = 6{,}6743 \cdot 10^{-11} \left[\frac{m^3}{kg \cdot s^2}\right]$ [7]. Dieser Wert wird jedoch nicht zwanghaft in eine Gravitationssimulation übernommen, sondern kann angepasst werden, schliesslich simuliert das Spiel nur die Abhängigkeiten der Gravitation von Masse und Abstand von Körpern nach dem Gesetz von Newton.

[6] (DMK, DPK, 2011): S. 159

[7] (DMK, DPK, 2011): Umschlag hinten, gerundet

4.2 Distanzen

Es ist unmöglich, die unvorstellbaren Distanzen zwischen Planeten und anderen Himmelskörpern einzuhalten. Wären Distanzen massstabsgetreu, würde das Spiel grösstenteils im völlig leeren Raum stattfinden. Damit wäre gezieltes Abspringen unmöglich, da man kein Ziel vor Augen hätte. Auch eine dynamische Kameraführung, die das Spielgeschehen für den Spieler darstellt, ist realitätsgetreu schwer umsetzbar. Die Radien von Planeten, Sonne und Spielfigur entziehen sich allen Realitätsansprüchen. Um ein Spiel zu gestalten, müssen die Grössenordnungen von Objekten vernachlässigt werden.

4.3 Orbitale

Eine weitere Massnahme zur dichteren Gestaltung der Spielumgebung ist das Platzieren von mehreren massereichen Himmelskörpern in derselben Umlaufbahn (Abbildung 10). In der Realität ist dies nicht möglich. Ein Planet hat nicht nur eine bestimmte Mindestmasse, sondern muss auch sein Orbit „säubern“. Das heisst, dass in einer Umlaufbahn nur ein dominanter Planet kreist, der alle kleineren Planeten aus dieser Bahn werfen würde.

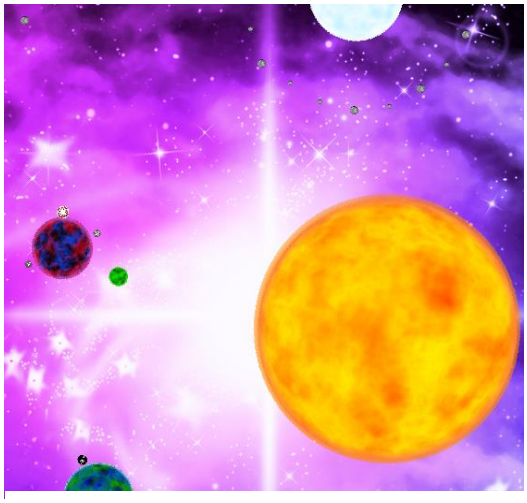


Abbildung 10 Ein Orbital ist von mehreren Planeten besetzt.

4.4 Massen

Die Sonne ist mit Abstand das massereichste Objekt eines Sonnensystems. So hat der Jupiter rund hundertfach weniger Masse als die Sonne[8]. In Gravity ist dieser Faktor eine Grössenordnung kleiner. Ein durchschnittlicher Planet ist nur rund zwanzigmal weniger massig als die Sonne. Mit der Wahl solch hoher Planetenmassen wäre es in der Realität ausgeschlossen, dass der gegenseitige Einfluss der Gravitationskraft zwischen Planeten vernachlässigbar ist. Trotzdem wurde Gravity nach diesem Prinzip umgesetzt.

[8] Nach (DMK, DPK, 2011): S. 212, S. 213

5 Problemstellungen während dem Entwicklungsprozess

5.1 Terraforming

Bis jetzt wurden alle Inhalte vorgestellt, die es in das fertige Spiel geschafft haben. Es gibt aber auch Ideen, die wegen ihrer Komplexität nicht in das Spiel miteingebunden werden konnten. Zu diesen Ideen gehört beim Spiel Gravity jene des Terraformen. Die Grundidee dahinter: Planeten sollten nicht nur schlicht rund sein, sondern aus mehreren Hügeln aufgebaut werden. Dies würde den Planeten eine Beerenform geben. Auf der Abbildung 11 ist eine Projektion eines Planeten zu sehen, der aus vier zusätzlichen Kreisen aufgebaut ist. Weil die Hügel sich weiter vom Mittelpunkt des Planten befinden würden, wäre auch das Gravitationsfeld schwächer. Für den Spieler wäre es theoretisch einfacher abzuspringen. Die Entwicklung einer solchen Mechanik bot jedoch Probleme.

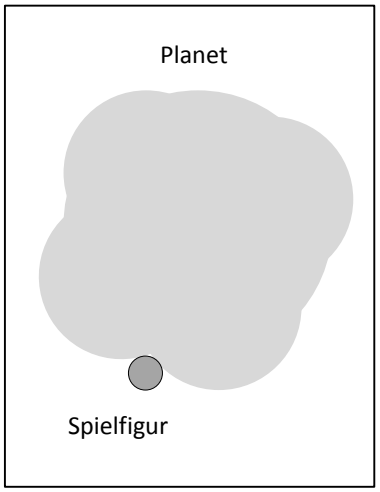


Abbildung 11 Schema eines Terraforming-Planeten

Zunächst soll erklärt werden, wie sich die Spielfigur überhaupt auf einem modifizierten Planeten bewegen würde. Während dem Laufen würde sie bei jedem Rechnungsschritt überprüfen, wie viele Kreise sie berührt, von denen der Planeten aufgebaut wird. Berührt sie nur einen, hat dies keinen Einfluss auf die weitere Bewegung. Sind es zwei Kreise (Abbildung 11), wechselt die Spielfigur den Kreis, auf dem sie sich rundherum bewegt. Zudem muss sie noch etwas verschoben werden, damit sie im nächsten Iterationsschritt nicht wieder die zwei selben Kreise berühren würde. Wäre das der Fall, wäre die Spielfigur gefangen, weil sie den Berührungspunkt nicht verlassen könnte. Anstatt auf dem nächsten Kreis weiterzulaufen, würde sie dauernd zwischen zwei Kreisen hin- und herwechseln. Diese Lösung des Problems zeigte bei anfänglichen Versuchen keine speziellen Schwächen. Erst als der Lösungsansatz mit sich bewegenden Planeten getestet wurde, zeigten sich erhebliche Mängel. Das eben genannte Problem, das mit der Verschiebung der Spielfigur gelöst werden sollte, wurde manifest. Die Spielfigur steckte in Tälern zwischen zwei Kreisen fest und wechselte zwischen jenen hin und her. Mit einer Weiterentwicklung der Kollisionsabfrage hätte das Problem gelöst werden können. Da zu diesem Zeitpunkt jedoch schon viel Zeit in das Terraformen investiert wurde, musste abgeschätzt werden, wie hoch der spielerische Nutzen dieser Funktion wäre. Wie auch in der Realität wird das Gravitationsfeld eines Planeten wegen geologischen Erhebungen irrelevant schwächer. Es wurde entschieden, das Terraformen komplett aus dem Spiel zu entfernen, auch wenn damit viel Arbeit verloren ging. Als positiver Nebeneffekt wird somit auch die Berechnung der Bewegung auf Planeten einfacher, da die Kollisionsabfrage wegfällt.

5.2 Darstellen von Grafiken

Um das Spiel optisch möglichst ansprechend zu gestalten, werden vorgefertigte, statische Grafiken verwendet. Deren Darstellung kann jedoch zu Problemen führen. Da Java Grafiken sehr ineffizient darstellt, kann es sein, dass die vorgeschriebene Berechnungszeit überschritten wird. Das Spiel Gravity läuft auf einer Simulationsfrequenz von 35 Hertz. Dauert die Berechnung eines Iterationsschrittes mehr als eine $1/35$ Sekunde, beginnt das Spiel zu stocken. Dies muss auf jeden Fall vermieden werden.

Um die Berechnungszeit nicht zu hoch werden zu lassen, musste eine Bildschirmauflösung gefunden werden, die das Spiel noch unterstützen kann. Die Darstellung im heutzutage üblichen HD-Format ist nicht möglich. Die Auflösung des Spiels beträgt deswegen nur 1200 mal 900 Pixel. Dies ist eine erste, grundlegende Massnahme, die Berechnungszeit möglichst zu minimieren.

Um im Spiel immer die Übersicht bewahren zu können, ist ein variabler Massstab bei der Darstellung möglich. Anfänglich sollte dieser Vergrösserungs- oder Verkleinerungsfaktor dynamisch über die Distanz des der Spielfigur am nächsten liegenden Planeten errechnet werden. Dynamisches Vergrössern und Verkleinern der Ansicht übersteigt jedoch den maximalen Aufwand eines Iterationsschrittes bei weitem. Da ein dynamischer Massstab nicht möglich ist, wurden für das Spiel drei fixe Zoomstufen gewählt. Jedes Objekt, also Planeten und Monde, generieren bei der Erstellung vier unterschiedlich grosse Grafiken aus einem Ursprungsbild. Der Zoomfaktor bestimmt nun, welches dieser zuvor erzeugten Bilder dargestellt werden soll. Damit bleibt auch beim Wechseln des Massstabs eine Berechnungszeit einer neuen Grafik aus. Mit dieser Methode läuft das Spiel zwar flüssig, jedoch setzt diese neue Grenzen. Die drei für jeden Planeten generierten Bilder beanspruchen einen grossen Teil des Arbeitsspeichers des Computers. Konkret hat das zur Folge, dass ab einer bestimmten Anzahl von Objekten der Arbeitsspeicher überfüllt ist, was den Abbruch des Spieles zur Folge hat. Mit aufsteigendem Level werden immer mehr Planeten generiert. Diese Lösung schafft automatisch ein Höchstlevel, bei dem die höchste Anzahl an Planeten generiert wird, die der Arbeitsspeicher abdecken kann.

Das Problem des Überfüllens des Arbeitsspeichers könnte durch eine Abänderung der Skalierung umgangen werden. Anstatt bei der Initialisierung des Spiels für jedes Objekt bereits ein zuvor skaliertes Bild zu generieren, würde nur beim Wechseln der Zoomstufen aus dem Originalbild die passende Grafik mit der korrekten Grösse berechnet und gespeichert werden. Eine weitere Massnahme zur Einsparung der Berechnungszeit ist das gezielte Zeichnen von jenen Objekten, die auch wirklich dargestellt werden müssen. Nur Objekte, die einen Ausschnitt des Bildschirmes bedecken, müssen auch gezeichnet werden. Auf dem Bildschirm haben im Normalfall nur etwa fünf Objekte Platz. Die Anzahl hängt natürlich stark von dem verwendeten Darstellungsmassstab ab. Bedenkt man, dass ein Level aus mehr als fünfzig Planeten bestehen kann, ist die Reduktion auf durchschnittlich fünf darzustellende Objekte eine grosse Einsparung von Rechenressourcen.

Durch die gezielte Darstellung von Objekten können diese, trotz zusätzlicher Berechnungszeit, rotiert werden. Ohne eine Rotation aller Himmelskörper würde keine Illusion des Gehens erzeugt. Die Illusion ist wichtig, um die Geschwindigkeit und Position der Spielfigur auf dem Planeten abschätzen zu können.

5.3 Ziel des Spieles

Bei jedem Spiel arbeitet man auf ein Ziel hin. Sei es das Erreichen eines bestimmten Ortes im Level oder das Finden von benötigten Gegenständen. Zudem gibt es Spiele, bei denen man in einem immer gleichbleibenden Spielablauf eine möglichst hohe Punktzahl erreichen muss. Bei Gravity war noch bei Beginn der Realisierungsphase nicht klar, wie das Ziel eines Levels definiert sein sollte. Es gab zwei unterschiedliche Ansätze. Der erste Ansatz würde das Spielprinzip so gestalten, dass die Spielfigur sich vom Mittelpunkt des Sonnensystems wegbewegen muss. Der Spieler hätte dazu nicht unbegrenzt viel Zeit. Die Sonne wäre am Ende ihres Lebenszyklus‘. Sie würde sich zu einem Roten Riesen aufblähen und sogar so gross werden, dass nach und nach Umlaufbahnen von Planeten von ihr überdeckt werden würden. Der Abbruch des Spiels erfolgt, wenn die Spielfigur auf dem Weg zum äussern Rand des Sonnensystems von der Sonne eingeholt worden ist.

Der zweite Ansatz ist wesentlich einfacher umzusetzen. Der Level gilt dann als geschafft, wenn entweder eine gewisse Anzahl an Objekten eingesammelt worden ist oder die Spielfigur einen bestimmten Planeten erreicht hat. In Anlehnung an bekannte Vertreter des „Jump and Run"-Genres müssen bei Gravity Münzen eingesammelt werden, um einen höheren Level zu erreichen.

5.4 Level-Generierung

Genauso wichtig wie die Mechanik des Spiels ist auch deren Umsetzung in ein ansprechendes Leveldesign. Bei Leveldesigns gibt es grundsätzlich zwei verschiedene Ansätze. Entweder sind alle Elemente und Objekte eines Levels bereits vordefiniert oder der Level wird zufällig generiert. Die Vorzüge der vordefinierten Levels sind, dass diese keine Fehler aufweisen. Durch Fehler bei einer Zufallsgenerierung kann es dazu kommen, dass sich zwei Planeten in ihrer Umlaufbahn kreuzen würden, weil der Abstand zwischen Bahnen nicht ausreichend gross ist. Zudem bieten vordefinierte Levels die Möglichkeit, bestimmte Stellen bewusst einzubauen, die den Spieler mehr fordern. Ein ganz essentieller Nachteil der designten Levels ist der Aufwand, diese zu gestalten und anschliessend so abzuspeichern, dass die Levels vom Programm geladen werden können.

Für die Levelgenerierung sorgt bei Gravity ein Algorithmus, der nur mit einer natürlichen Zahl zwischen eins und zehn einen funktionsfähigen Level generieren kann. Der Spieler beginnt mit Level 1 und muss sich bis auf Level 10 hochspielen. Für jeden Level wird ein neues Sonnensystem generiert. Auch zuvor abgeschlossene Levels werden beim wiederspielen neu berechnet und gleichen dem damaligen Level nicht exakt. Die Zahl des Levels bestimmt dabei Faktoren wie die Anzahl der Bahnen um die Sonne sowie auch die Anzahl der darin befindlichen Planeten. Weiter passt der Algorithmus Konstanten an. Je höher die Levelzahl, desto schwieriger wird der Level. Der erlittene Schaden durch Asteroiden und Sonnenstrahlung nimmt zu, Dämpfer und Asteroidengürtel werden immer häufiger erzeugt und gleichzeitig werden Bonusgegenstände immer seltener.

6 Beurteilung durch Probanden

6.1 Die Probanden

Nach der Fertigstellung der ersten Version des Spieles, der sogenannten Beta, sollte nun ein erstes Fazit gezogen werden. Das Spiel soll von unabhängigen Testern beurteilt werden. Somit können kleine Fehler, in der Fachsprache auch Bugs genannt, gefunden werden. Allenfalls kann man auch Spielmechaniken anpassen und so den Spielspass erhöhen.
Die Probandengruppe bestand aus zehn Probespielern, die sich in zwei Gruppen unterscheiden lassen: Den Gelegenheitsspielern und die „Vollzeitspielern. Der Gelegenheitsspieler spielt selten und dabei hauptsächlich auf einem Smartphone. Der Vollzeitspieler ist dagegen bereit, Zeit und Geld ins Spielen zu investieren. Das macht ihn im Umgang mit Videospielen am geübtesten. Er hat auch die Möglichkeit, Parallelen zu anderen Spielen zu ziehen und kann sie besser miteinander vergleichen. Damit auch Gelegenheitsspieler Gravity vergleichen können, mussten alle Probanden zwei Referenzspiele testen, die ein ähnliches Spielprinzip aufweisen. Diese werden in den zwei folgenden Unterkapiteln kurz vorgestellt.

6.2 Kurzportrait „Escape the Red Giant“

Die Sonne eines Sonnensystems wird zum Roten Riesen und bläht sich dabei auf. Die Spielfigur kann grosse, runde Asteroiden als Sprungbretter benutzen, um möglichst schnell von der Sonne weg zu gelangen. Wie bei Gravity kann der Spieler um Asteroiden herum rennen, um für den Absprung Anlauf zu holen. Der Spieler sammelt im Verlauf des Spieles nur Punkte. Diese erhält man immer dann, wenn die Spielfigur auf einem Asteroiden landet. Je länger der Sprung, desto mehr Punkte erhält der Spieler. Asteroiden können durch die Wucht der Spielfigur zerstört werden. Durch das Zerstören eines Asteroiden verliert die Spielfigur an Geschwindigkeit, jedoch erhöht dies den Punktemultiplikator. Beim Springen können Tricks ausgeübt werden, die den Punktemultiplikator zusätzlich erhöhen. Das Spielprinzip ist sehr simpel. Das Hauptziel besteht darin, einen Highscore zu erreichen. Mehrere Levels gibt es nicht.

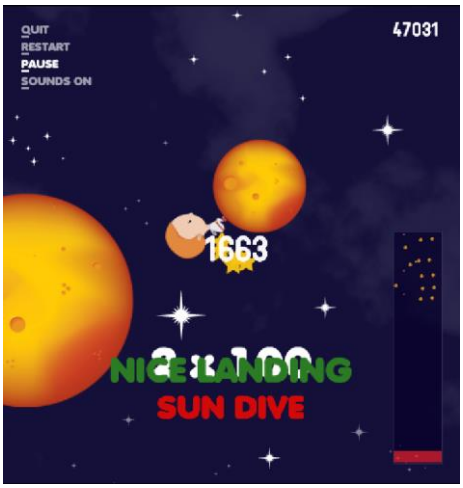


Abbildung 12 Screenshot aus dem Gameplay von „Escape the Red Giant“. (ooPixel, 2009)

6.3 Kurzportrait „Planet Hopper“

Die Spielfigur findet sich auf einem Planeten wieder. Ziel des Spiels ist es, die überall verteilten Planeten mit Hilfe von Flaggen einzunehmen und damit Geld zu verdienen. Dazu kann der Spieler auf dem Planeten rennen, um Anlauf zu nehmen. Sobald der Spieler im Weltraum ist, kann er zur zusätzlichen Bewegungskontrolle einen Raketenantrieb zünden. Dieser Raketenantrieb ermöglicht ihm gezielte Bewegungen im Weltall. Mit dem eingenommenen Geld kann man die Ausrüstung reparieren oder sogar verbessern. Tut man dies nicht, lebt man gefährlich. Ist die Ausrüstung zu sehr abgenutzt, kann ein Aufenthalt im Weltraum oder das harte Auftreffen auf einem Planeten mit dem Tod enden. Mit dem ersten Tod ist auch das Spiel vorbei. Auch hier gibt es nicht mehrere Levels. Auch eine Sonne sucht man vergeblich. Vielleicht mit ein Grund, dass sich die Planeten nicht relativ zu einander bewegen.

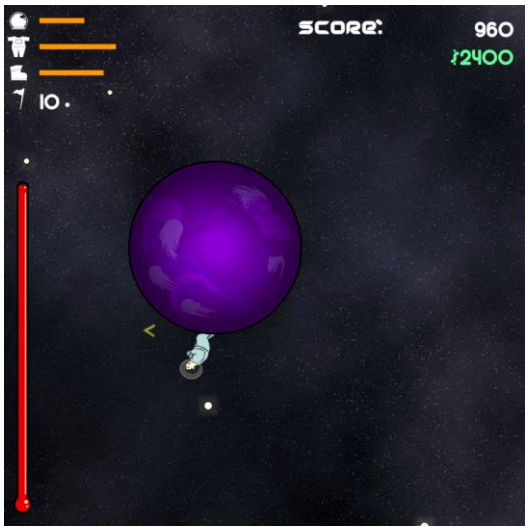


Abbildung 13 Screenshot aus dem Gameplay von „Planet Hopper“. (beast games, unbekannt)

6.4 Fazit zur Spielmechanik

Die Probanden mussten bewerten, wie spielbar Gravity für sie ist. Der wichtigste Einstieg in das Spiel ist das interaktive Tutorial, welches die Spieler über die Steuerung und das Ziel des Spieles aufklärt. Alle Probanden fanden das Tutorial so informativ, dass sie Gravity spielen konnten. Es fehlte jedoch eine genaue Aufklärung über die erhältlichen Bonusgegenstände. Entweder wurden diese während dem Probespielen gar nicht erst bemerkt oder man konnte ihre Funktion nicht abschätzen. Als Verbesserung wurde die Wirkung aller Bonusgegenstände verstärkt und im Tutorial der fertigen Version deren Funktionsweise kurz erklärt. Auch die Übersicht über das Spielgeschehen war für den Grossteil der Probanden gegeben. Es stellte sich heraus, dass nur die zwei geringsten der anfänglichen vier Zoomstufen verwendet wurden. Für die fertige Version wurden die drei verbliebenen Zoomstufen so abgestimmt, dass sie etwa den zwei vorherigen meistgenutzten Zoomstufen entsprachen. Bemängelt wurde, dass das Anlauf holen keine Wirkung auf den Absprung zeigte. Zudem war es nicht möglich, entgegengesetzt der Laufrichtung des Planeten abzuspringen. Dies war nicht möglich, weil beim Absprung die Relativgeschwindigkeit des Planeten zur Sonne zur Bewegungsgeschwindigkeit der Spielfigur addiert wurde. Springt die Spielfigur entgegengesetzt der Laufrichtung, bremst es sie mangels Sprungkraft so stark aus, dass sie den Planeten nicht verlassen konnte. Dank diesen Rückmeldungen wurde das ganze Absprungverhalten überarbeitet. Der Grossteil aller Testversuche fand auf dem gleichen Computer statt. Wichtig war nun zu kontrollieren, ob das Spiel auch auf anderen Geräten fehlerfrei funktioniert. Es stellte sich heraus, dass auf anderen Geräten das Spiel abstürzt, wenn man das Spiel aus dem Pausenmodus wieder startet. Mit diesem Problem wurde darauf aufmerksam gemacht, dass das Spiel auf mehreren Geräten ausprobiert werden muss, um sicher zu gehen, dass Gravity flüssig läuft.

6.5 Fazit zur Spielidee

Die Frage, wie sehr sich die Referenzspiele und Gravity vom Prinzip her ähneln, wurde von den Probanden damit beantwortet, dass man bei allen Spielen zwischen Himmelskörpern hin und her springen muss. Für die Probanden fühlten sich die Spiele abgesehen von diesem Aspekt verschieden an. Die Kombination von bekannten „Jump and Run“-Elementen verbunden mit der Dynamik eines Sonnensystem sei neu und innovativ. Die Einbindung einer Gravitationssimulation erfordert neues Denken und muss zuerst erlernt werden. Dass Gravity in verschiedene Levels gegliedert wird, deren Schwierigkeitsgrad stetig zunimmt, ist als Ansporn aufgenommen worden. Eine reine Highscore-Jagd ist dem Grossteil der Probanden zu simpel und senkt den Wiederspielwert.

Grundsätzlich fällt das Fazit zur Spielidee sehr positiv aus. Folgende Verbesserungsmöglichkeiten wurden angeregt: Grösster Kritikpunkt ist das komplette Verzichten auf Musik und Soundeffekte. Für ein abgerundetes Spielerlebnis brauche es auch akustische Untermalungen. Zudem fehlt einigen Probanden die Spieltiefe. Funktionen in Planet Hopper, die das Verbessern der Ausrüstung erlauben, würden den Widerspielwert erhöhen. Die Spielfigur, die einfach durch einen Ball dargestellt wird, sei zu charakterlos. Die Referenzspiele bieten eine Figur, mit der man sich besser identifizieren kann. Auch die animierten Bewegungen dieser Spielfiguren lobten die Probanden. Kritisiert wurde zudem der Schwierigkeitsgrad von Gravity. Der erste Level sei bereits zu schwer und sorge für Frust.

6.6 Fazit zur Testserie

Dieser Test hat gezeigt, wie wichtig es ist, ein Produkt einer ausgewählten Gruppe zu präsentieren, bevor dass das Produkt finalisiert wird. Nur so können Überlegungs- und Umsetzungsfehler gefunden werden, die während der Entwicklung nicht aufgefallen sind, weil die nötige Distanz zum Projekt gefehlt hat. Mit all den genannten Kritikpunkten ist ein gezieltes Verbessern möglich. Im Fall von Gravity ganz wichtig die Erkenntnis der mangelnden Kompatibilität des Spiels auf allen Computern.

7 Produktreflexion

Nach einem halben Jahr Planung und Realisierung soll eine Bilanz gezogen werden.

Das Spiel, das in dieser Zeit entstanden ist, entspricht grösstenteils dem, was ich mir zu Beginn des Projektes vorgestellt habe. Alle wichtigen Spielelemente wurden integriert, während dem Entwicklungsprozess wurden auch einige verworfen. Ich lernte in meinem ersten grösseren Informatikprojekt zu planen und Schulwissen kreativ umzusetzen. Jedoch verläuft selten ein erster Versuch fehlerfrei. Schon bei der Planung machte sich bemerkbar, dass ich eigentlich, entgegen meiner damaligen Meinung, nicht wirklich wusste, was ich tat. So erhoffte ich von mir selbst, dass ich das Projekt gut über das Schuljahr verteilen würde. Jedoch zeigte sich auch da, dass Vorstellung und Realität nicht immer dasselbe sind. So waren die Ferien doch viel stärker von der Maturaarbeit geprägt als zunächst erwartet. Auch muss ich im Nachhinein meine Effizienz stark bemängeln. Doch nur Übung macht schlussendlich den Meister. Ich habe viel darüber gelernt, wie man Spielsituationen mathematisch umsetzen kann. Solche Spielsituationen können Kollisionsabfragen, Bewegungsmechaniken oder Darstellungsaspekte sein.

Der prägendste Teil des Arbeitsprozesses waren die Schwankungen der Gefühlslage. Stets wechselte sie zwischen Euphorie und Frustration. Euphorische Momente erlebte ich dann, wenn Spielelemente endlich fehlerfrei implementiert wurden. Die dann freudige Stimmung wurde meist durch ein neues oder wiederkehrendes Problem zunichte gemacht. Die Euphorie wich der Frustration. Zusammengefasst: ein Informatikprojekt braucht viel Geduld und starke Nerven. Der schönste Moment kommt dann, wenn das Produkt zum ersten Mal Spielern vorgestellt werden kann. Das positive Feedback, das ich erhalte habe, macht alle Strapazen auf dem Weg dorthin wieder wett.

Die Motivation, ein Videospiel zu entwerfen, hat zwei Hintergründe: Erstens bin ich selbst jahrelanger passionierter Videospieler, was zum zweiten Punkt führt. Spielentwickler zu werden ist ein bereits länger bestehender Berufswunsch meinerseits. Mit dieser Arbeit wollte ich für mich selbst herausfinden, wie sehr mich Spielentwicklung auch wirklich anspricht. Zu diesem Punkt sei nach hinzuzufügen, dass dieser Berufswunsch während der Entwicklung sich doch von Spieleentwickler wegbewegt hat.

8 Danksagungen

Ich danke:

Linus Bucher,
Jonas Chastonay,
Susanna Continelli,
Dinah Marti,
Carmen Meier
Laura Sasdi,
Dan Studer,
Aron Szakàcs,
Jacob Werner
und Peter Werner für das ausgiebige Testen meines Spiels sowie für das positive und konstruktive Feedback.

Auch danke ich meinem Betreuer Stefan Rothe, der immer wenn ich Fragen zu Spielelementen oder Mechaniken hatte, mir stets gleich eine oder mehrere Ideen geben konnte.

Spezieller Dank richtet sich an meine Eltern, die sich die Zeit genommen haben, meine Arbeit sorgfältig durchzulesen und Verbesserungen anzuregen.

9 Quellenangaben

9.1 Literaturverzeichnis

DMK, DPK. (2011). *Formeln, Tabelle, Begriffe* (3 Ausg.). Zürich, Zürich, Schweiz: orell füssli.

9.2 Spielreferenzen

beast games. (unbekannt). Planet Hopper. *Planet Hopper*. Von http://www.arcadebomb.com/play/planet_hopper.html abgerufen

ooPixel. (07. April 2009). Escape the Red Giant. http://www.kongregate.com. Von http://www.kongregate.com/games/oopixel/escape-the-red-giant abgerufen

9.3 Abbildungen

I. Herleitung der Rotationsabbildung um einen beliebigen Punkt

Eine mathematische Matrix besteht aus einer tabellenartigen Anordnung von Elementen. Wie eine Tabelle besteht eine Matrix aus einer beliebigen Anzahl an Zeilen und Spalten. Einträge in einer Matrix nennt man Elemente. Dazu zählen unter anderem Zahlenmengen, wie die der reellen Zahlen. Matrizen besitzen eigene Rechenregeln. Die weitaus wichtigste Operation ist die Multiplikation von zwei Matrizen. Prinzipiell erfolgt eine Multiplikation immer nach dem gleichen Schema: Man multipliziert die Zeilen der rechten Matrix mit den Spalten der linken Matrix. Dabei multipliziert man das erste Element der Zeile mit dem ersten Element der ersten Spalte, addiert anschliessend die Produkte der zweiten Elemente der ersten Zeile und Spalte. Dies wird fortgeführt bis zum n-ten Element. So erhält man das Element der ersten Spalte und der ersten Zeile der Produktmatrix. Man erhält alle Elemente der Produktmatrix, indem man jede Zeile der ersten Matrix mit jeder Spalte der rechten Matrix auf diese Weise „multipliziert“. Der Platz des neuen Elementes ergibt sich aus der Nummer der jeweiligen Zeile und Spalte.

Aus dieser Definition geht hervor, dass die Multiplikation von Matrizen nicht kommutativ ist. Sind A und B Matrizen heisst dies: $A \cdot B \neq B \cdot A$.
Als Beispiel einer Multiplikation sei jene zweier 2×2-Matrizen gezeigt:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} a \cdot e + b \cdot g & a \cdot f + b \cdot h \\ c \cdot e + d \cdot g & c \cdot f + d \cdot h \end{pmatrix}$$

Die Rotationsmatrix um einen beliebigen Punkt *C* $\left(x_C \middle| y_C\right)$ ist die Multiplikation von drei speziellen Kongruenzabbildungen. Mit der ersten Translation werden alle Punkte um den Vektor $\vec{v}$ verschoben. Der Vektor ist so gewählt, dass das Drehzentrum *C* sich nach der Translation im Ursprung $(0|0)$befindet. Diese Matrix hat folgende Form:

$$\begin{pmatrix} 1 & 0 & -x_C \\ 0 & 1 & -y_C \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}$$

Anschliessend erfolgt eine Rotation um den Winkel *α* um die *z*-Achse. Im zweidimensionalen Raum, der sich auf die *xy*-Ebene beschränkt, kommt dies einer Rotation um den Ursprung gleich.

$$\begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} x'' \\ y'' \\ 1 \end{pmatrix}$$

Zum Schluss erfolgt eine Translation aller Punkte um den Vektor $-\vec{v}$.

$$\begin{pmatrix} 1 & 0 & x_C \\ 0 & 1 & y_C \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x'' \\ y'' \\ 1 \end{pmatrix} = \begin{pmatrix} x''' \\ y''' \\ 1 \end{pmatrix}$$

Multipliziert man diese drei Abbildungen, erhält man die folgende vollständige Matrix:

$$\begin{pmatrix} 1 & 0 & x_C \\ 0 & 1 & y_C \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & -x_C \\ 0 & 1 & -y_C \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha & \sin\alpha \cdot y_C + x_C(1-\cos\alpha) \\ \sin\alpha & \cos\alpha & -\sin\alpha \cdot x_C + y_C(1-\cos\alpha) \\ 0 & 0 & 1 \end{pmatrix}$$

Multipliziert man nun diese Matrix mit dem Koordinatenvektor eines beliebigen Punktes *P* $(x_P|y_P)$, ergibt sich die Funktion für die neuen Koordinaten des Punktes *P'*:

$$\begin{pmatrix} \cos\alpha & -\sin\alpha & \sin\alpha \cdot y_C + x_C(1-\cos\alpha) \\ \sin\alpha & \cos\alpha & -\sin\alpha \cdot x_C + y_C(1-\cos\alpha) \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_P \\ y_P \\ 1 \end{pmatrix} = \begin{pmatrix} x_P{}' \\ y_P{}' \\ 1 \end{pmatrix}$$

Daraus lassen sich folgende Funktionen schliessen:

$$x'_P(x_P, y_P, x_C, y_C) = x_C + \cos\alpha \cdot (x_P - x_C) - \sin\alpha \cdot (y_P - y_C)$$
$$y'_P(x_P, y_P, x_C, y_C) = y_C + \sin\alpha \cdot (x_P - x_C) + \cos\alpha \cdot (y_P - y_C)$$

II. Eigenständigkeitserklärung

Ich bestätige hiermit, dass ich die Arbeit „Gravity – Entwerfen und Programmieren einer eigenen Spielidee“ selbstständig verfasst habe und keine nicht angegebenen Hilfsmittel benutzt habe.

Weiter bestätige ich, dass alle nicht von mir erzeugten Spielgrafiken und Spielklangeffekte, die in Gravity benutzt werden entweder unter der Lizenz CC0 (Public Domain Dedication) oder CC BY 3.0 (creative commons; Attribution 3.0 unported) stehen. Bei allen Grafiken unter der CC BY 3.0 Lizenz wurden die Erschaffer ersichtlich sowie Manipulationen durch den Entwickler (Michael Marti) angegeben.

Ort und Datum	Unterschrift
_____	_____