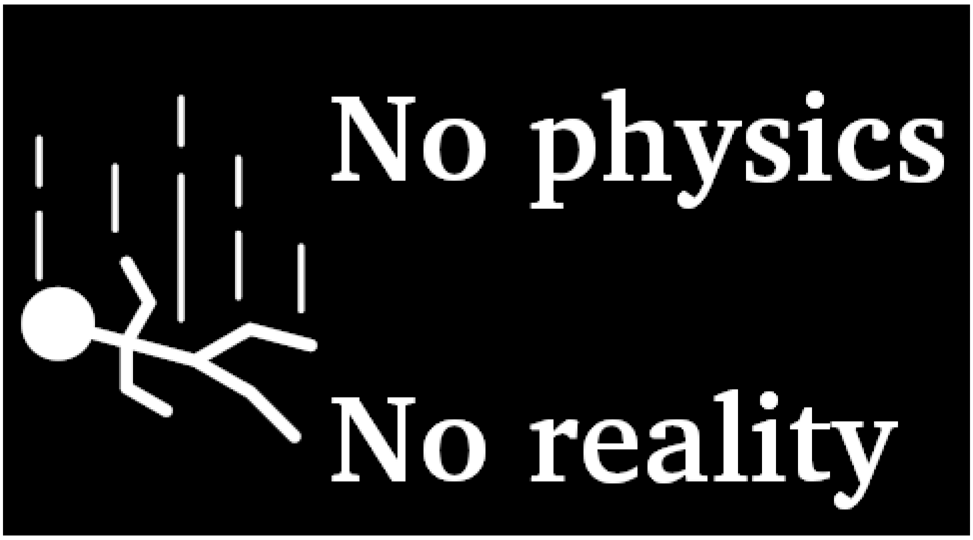
Städtisches Gymnasium Rheinbach
Qualifikationsphase 2
2020

Besondere Lernleistung im Leistungskurs

Informatik



Physik-Engine

Entwicklung einer Physik-Engine, die einen Sprung eines Fallschirmspringers simulieren soll. Dies beinhaltet die Berechnung eines Objekts im freien Fall, mit Luftwiderstand und den Fall aus großer Höhe mit wachsender Erdbeschleunigung mit und ohne Luftwiderstand. Dabei soll die Anpassung sämtlicher Parameter möglich sein. Zur Datenoptimierung wird ein Modell-Experiment durchgeführt und die Daten werden durch Parameterfitting in der Physik-Engine simuliert.

Leitender Lehrer: Herr Faßbender
Verfasser: Samuel Schneider

Inhaltsverzeichnis

Einleitung

Schon seit Beginn der Raumfahrt benutzten Wissenschaftler physikalische Simulationen, um die genaue Flugbahn von Raketen und Flugkörpern zu berechnen. Diese sind so exakt, dass man die Position der Raketen auf den Meter genau approximieren kann. Diese Genauigkeit ist aber auch notwendig, wenn man Flugkörper in der Atmosphäre steuern möchte. Somit sind Simulationen im 21. Jahrhundert ein großes Thema im Bereich der Physik.

Denn durch Simulationen ist es möglich eine Situation oder Verhalten eines Objekts genauestens in der Zukunft zu approximieren. Besonders in komplexen Systemen wie im Weltall oder auf Planeten wirken viele Kräfte auf das Zielobjekt und müssen berechnet werden. Eine Simulation ist die optimale Möglichkeit, die Auswirkungen unterschiedlicher Kräfte effizient darzustellen.

Aber physikalische Simulationen sind schon lange nicht mehr nur in der Wissenschaft vorhanden. Mithilfe von Physik-Engines besitzt jeder Computer eine eigene physikalische Simulationmöglichkeit, die heutzutage effektiv benutzt wird. Denn unter anderem sind Computerspiele und Anwendungen mit Schwerkraft und anderen physikalischen Effekten umsetzbar.

Im Verlauf dieser Arbeit wird die Methodik erläutert, die es ermöglicht, einen Fall mit Luftwiderstand mittels einer Physik-Engine zu simulieren. Diese wurde im Rahmen der Arbeit mit der Programmiersprache Java implementiert und getestet. Dazu wurde ein Versuch durchgeführt, bei dem der Fall eines Kegels ausgewertet wurde. Mittels Videoanalyse konnten Daten errechnet und Rückschlüsse auf die Beschleunigung und Position des Kegels getroffen werden. Durch Parameterfitting können die Daten des Versuchs in der Physik-Engine simuliert und die Genauigkeit der Verfahren überprüft werden. Dafür wurden die Ergebnisse aus dem Versuch und aus der Physik-Engine verglichen.

Definition und Anwendung

Eine Physik-Engine ist eine Computersoftware und wird für die Simulation von physikalischen Gesetzen genutzt. Dazu gehören unter anderem die Gesetze für statische und dynamische Körper, Fall-, Rotations-, Impuls- und Fließgesetze. Dabei findet eine Physik-Engine Gebrauch in Computerspielen und hochpräzisen Anwendungen, die für physikalische Simulationen von Wissenschaftlern genutzt werden.

In Computerspielen werden die physikalischen Gesetze in Echtzeit simuliert und können somit nicht exakt berechnet werden, wie in Anwendungen für Forschungszwecke. Trotzdem wirken die Simulationen in Computerspielen realistisch und weisen nur kleine Abweichungen in den Werten vor. Außerdem geht es in Echtzeit-Programmen nicht darum, exakte Berechnungen anzustellen, sondern effizient eine physikalische Welt zu simulieren. Denn je akkurater die Berechnung, desto größer wird der Rechenaufwand.[1]

Physik-Spiele

In einigen Computerspielen werden physikalische Prozesse simuliert, um die Spielerfahrung realistisch und interessant zu gestalten. Dabei interagiert der Spieler mit dem Spiel und bemerkt, inwiefern sich Elemente nach physikalischen Gesetzen bewegen. Außerdem können durch physikalische Eigenschaften Herausforderungen für den Nutzer gestaltet werden, sodass der Spieler zum Beispiel mit der Schwerkraft, Flugbahnen, Hebelwirkungen und unterschiedlichen Fluiden und Gasen experimentieren muss, um einen Fortschritt zu erreichen.[2]

Hochpräzise Anwendungen

Steht Exaktheit im Vordergrund, können Physik-Engines hochkomplizierte physikalische Bewegungen und Effekte simulieren. Für genaue Ergebnisse, die kaum von der Realität abweichen sollen, ist ein enormer Rechenaufwand nötig. Beispiele für Simulationssoftware sind SimulationX oder Wolfram SystemModeler.[3]

Physik-Engines in Spiel-Engines

Um physikalische Eigenschaften wie Schwerkraft in Spielen zu simulieren, benutzen Spieleentwickler Spiel-Engines, die vollständige Physik-Engines enthalten. Die größten Spiel-

[1] (7), Physik-Engine
[2] (7), Physik-Engine
[3] (8), Simulationssoftware

Engines zurzeit sind Unity und UnrealEngine. Beide dieser Spiel-Engines verwenden die Physik-Engine PhysX. PhysX ist eine open source Physik-Engine des Unternehmens Nvidia Corporation. Dabei verlagert PhysX die Berechnung physikalischer Effekte in Computerspielen und Simulationssoftware auf Grafikkarten der GeForce-Serie des Herstellers. Dadurch wird der Prozessor entlastet, weil die aufwendigen Berechnungen auf der Grafikkarte berechnet werden können. Somit verfügt der Prozessor über mehr Kapazität, um die Ablauf- und Darstellungsgeschwindigkeit zu verbessern.[4] [5] [6]

Physic Processing Unit (PPU)

Ein Physikbeschleuniger (eng. Physic Processing Unit kurz PPU) ist ein Koprozessor zur eigenständigen Berechnung vorrangig physikalischer Effekte. Dabei soll eine PPU dazu dienen, den Hauptprozessor, die Central Processing Unit (CPU), zu entlasten. Vorgestellt wurde die erste PPU „PhysX-Chip“ 2005 von der Firma Ageia. Dieser unterstützte die Simulation von Kleidung, Haaren, Festkörpern sowie Flüssigkeiten, konnte aber auch für Kollisionserkennung und -berechnung verwendet werden. 2008 wurde Ageia von Nvidia Corporation aufgekauft. Seitdem verwenden alle Grafikkarten ab der 8000er-Serie des Unternehmens die integrierte Physik-Engine PhysX.

Jedoch konnte sich die PPU aufgrund der hohen Anschaffungskosten nicht durchsetzen, sodass mittlerweile alle Physik-Engines auf der GPU (Graphics Processing Unit) oder nur auf der CPU basieren. Eine Weiterentwicklung wurde durch das Unternehmen Nvidia Corporation nicht vorgesehen, da die Physik-Engine auf den Grafikkarten die PPU ersetzt.[7]

Dementsprechend entwickeln sich GPU-Prozessoren nach dem Prinzip GPGPU (General Purpose Computation on Graphics Processing Unit) zu Prozessoren, die für Berechnungen über ihren ursprünglichen Aufgabenbereich gedacht sind. Denn durch Multithreading auf einer GPU können enorme Geschwindigkeitssteigerung erzielt werden und beispielsweise physikalische Effekte berechnet werden.[8]

[4] (6), PhysX
[5] (17), UnrealEngine
[6] (18), Unity
[7] (9), PPU
[8] (10), GPGPU

Vektorielle Größen

Unter vektoriellen Größen versteht man eine physikalische Größe, die einen Richtungscharakter hat. Solche Größen sind durch einen Betrag als auch durch eine Richtung gekennzeichnet. Beispiele für vektorielle Größen sind Geschwindigkeit, Beschleunigung und Kräfte. Dabei sind vektorielle Größen wie geometrische Vektoren zu Verwenden. Vektoren sind besonders nützlich und übersichtlich in Raumbeispielen, um zum Beispiel Bewegungen darzustellen. [9]

Aufgrund der Kompatibilität und Darstellung wurde in der Physik-Engine mit Vektoren gerechnet. Diese veranschaulichen die X und Y-Größe der Beschleunigung und Geschwindigkeit des Fallschirmspringers und werden in folgende Darstellung mittels eines Pfeils veranschaulicht.

Verfahren

In der Physik wird das Eulersche Polygonzugverfahren benutzt, um näherungsweise physikalische Bewegungen zu beschreiben. Dazu zählen zum Beispiel die Beschleunigung, Geschwindigkeit und die folgende Position. Diese Anwendung in der Physik wird auch Methode der kleinen Schritte genannt.

Eulersche Polygonzugverfahren

Das Eulersche Polygonzugverfahren oder auch explizites Euler-Verfahren ist ein Einschrittverfahren, um Differentialgleichungen mit Anfangswertproblem numerisch zu lösen.[10]

Gegeben sei die Differentialgleichung $y'(t) = f(x, y)$ und die Bedingung $y(t_0) \; = \; y_0$.
Dabei ist y_0 der Anfangswert der Differentialgleichung y(t) bei $t = 0$. Betrachtet wird die Differentialgleichung in einem bestimmten Intervall I [a, b].

[9] (11), Vektoren
[10] (13), Explizites Euler Verfahren

Um die Differentialgleichung y(t) näherungsweise zu bestimmen, wählt man eine bestimmte Schrittweite h mit $h > 0$. So ergeben sich in dem Intervall I eine Anzahl n an Punkten, um die Gleichung zu skizzieren.

$$n = \frac{b - a}{h}$$

Für die Punkte ergeben sich die Zeitpunkte t_k.

$$t_k = t_0 + k \cdot h, k = 0, 1, 2, \ldots$$

Der Startpunkt um die Differentialgleichung y(t) annähernd zu beschreiben ist y_0. Um die folgenden Punkte zu bestimmen, muss der Wert der Punkte zu den Zeitpunkten t_{k+1} berechnet werden. Dazu wird bei kleiner Schrittweite näherungsweise eine konstante Steigung angenommen.[11] So ergibt sich:

$$y_{k+1} = y_k + h \cdot f(t_n, y_n), k = 0, 1, 2, \ldots$$

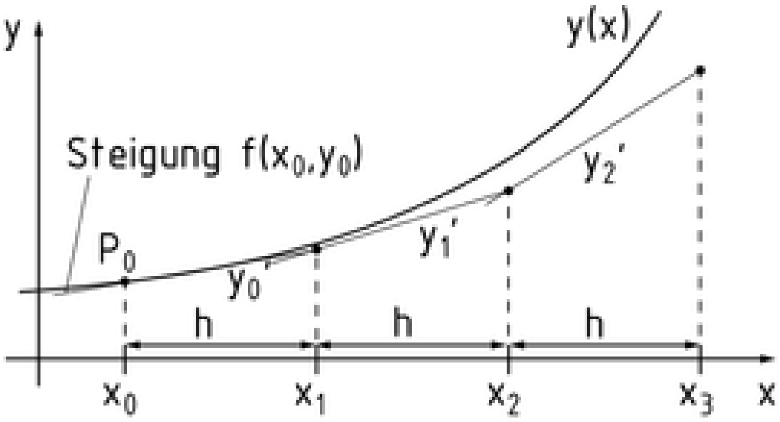


Bild 1: Skizze zur Veranschaulichung des Verfahrens

[11] (19), Eulersche Polygonzugverfahren

Methode der kleinen Schritte

Die Anwendung des Eulerschen Polygonzugverfahren wird in der Physik durch die Methode der kleinen Schritte umgesetzt. Diese wird ebenfalls benutzt, um in der Physik-Engine annähernd die Geschwindigkeit und Position zu bestimmen. Denn bei schwereren mathematischen Berechnungen wie bei einer inkonstanten Beschleunigung, ist es effizienter die Geschwindigkeit und Position zu approximieren.[12]

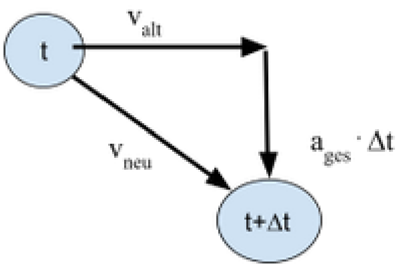


Abbildung 2: Berechnung der neuen Position

Dafür nimmt man an, dass in einem kurzen Zeitintervall Δt die Beschleunigung und Geschwindigkeit konstant ist.

Die Geschwindigkeit v_{neu} wird berechnet, indem man die Geschwindigkeit aus dem vorherigen Schritt v_{alt} mit dem Produkt der Beschleunigung a_{ges} und dem Zeitschritt Δt addiert.

$$v_{neu} = v_{alt} + a_{ges} \cdot \triangle t$$

Die nächste Position s_{neu} ergibt sich aus der Differenz der letzten Position s_{alt} und der mit Δt skalierten Geschwindigkeit v_{neu}.

$$s_{neu} = s_{alt} - v_{neu} \cdot \triangle t$$

Dabei funktioniert diese Methode exakter, je kleiner die Schrittweise Δt ist. Jedoch bedeutet es ebenfalls, dass mehr Rechenleistung nötig ist, um die Methodik anzuwenden.

---

[12] (1), Methode der kleinen Schritte

Optimierung des Verfahrens

Die Methode der kleinen Schritte ist die einfachste und ungenauste Möglichkeit der Einschrittverfahren, um die Geschwindigkeit und Position eines simulierten Objektes zu approximieren. Ein weitaus genaueres Verfahren ist das Runge-Kutta Verfahren.[13]

Physik

Fall mit Luftwiderstand

Befindet sich ein Körper in einem homogenen Feld unter Vernachlässigung der Luftreibung und der Zunahme der Gravitationskraft, fällt dieser Körper senkrecht mit der konstanten Beschleunigung g. Der Wert entspricht der Erdbeschleunigung und liegt in Deutschland bei ungefähr $9.81\ \frac{m}{s^2}$. Damit beschreibt der Fall eine gleichmäßig beschleunigte Bewegung. Das würde bedeuten, dass sich der Körper konstant beschleunigt und an 9.81 m/s pro Sekunde an Geschwindigkeit zunimmt.[14]

Jedoch trifft ein Körper in der Atmosphäre der Erde auf Luft und der Luftwiderstand wirkt auf ihn. Wenn sich ein Körper relativ zur Luft bewegt, wirkt die Luftwiderstandskraft F_{LR} auf den Körper. Dabei wirkt die Luftwiderstandskraft entgegengesetzt der Fallrichtung und somit gegen die Gravitationskraft F_G, die den Körper in Richtung Erde bewegt. Die Luftwiderstandskraft nimmt jedoch bei steigender Geschwindigkeit quadratisch zu, weshalb ein Körper eine annähernd maximale Geschwindigkeit erreichen kann.[15]

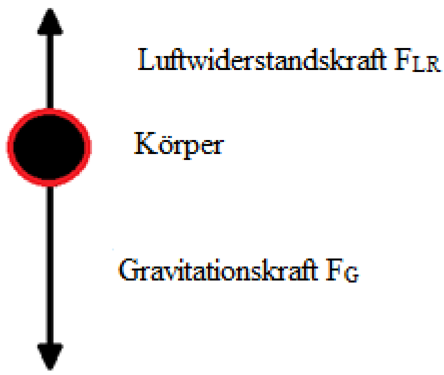


Abbildung 3: Wirkung der Kräfte auf den Körper

---
[13] (12), RK4
[14] (2), Freier Fall
[15] (3), Luftreibung

Strömungswiderstandskoeffizient c_W

Der Strömungswiderstandskoeffizient oder Luftwiderstandsbeiwert ist ein Parameter für den Strömungswiderstand in einer verformbaren Substanz. Denn in einem gasförmigen und flüssigen Medium erfährt jeder Körper ein Strömungswiderstand, der entgegengesetzt der Bewegungsrichtung wirkt.
Dabei handelt es sich bei der Atmosphäre der Erde um Luft - eine verformbare Substanz. Deswegen wirkt ein Strömungswiderstand auf den Körper in der Atmosphäre der Erde, da dieser die Luft bei einem Fall mit Luftwiderstand verdrängt. Der Strömungswiderstandskoeffizient lässt sich mit der Querschnittfläche A des Körpers und der Dichte der Atmosphäre ρ berechnen.[16]

Der Betrag der Luftwiderstandskraft verändert sich dadurch in Abhängigkeit von der Geschwindigkeit v, der Querschnittsfläche A, dem Strömungswiderstandskoeffizienten c_W und der Dichte der Atmosphäre ρ. Somit lässt sich die geschwindigkeitsabhängige Luftwiderstandskraft $F_{LR}(v)$ durch die folgende Gleichung beschreiben:

$$F_{LR}(v) = \frac{1}{2} \cdot c_W \cdot \rho \cdot A \cdot v^2$$

Dichte der Atmosphäre

Da sich die Dichte der Luft mit dem Abstand zur Erde verändert, kann mit der barometrische Höhenformel die Dichte der Luft in einer bestimmten Höhe h berechnet werden. Die Darstellung wird jedoch so vereinfacht, dass überall in der Atmosphäre dieselbe Temperatur herrscht. Ebenfalls führt die Einführung der Skalenhöhe h_s zur Vereinfachung der Höhenformel. Dabei ist die Skalenhöhe ein natürliches Maß für die Höhe der Atmosphäre und den Druckverlauf in ihr. So ergibt sich die folgende Gleichung für die Dichte der Luft $\rho(h)$ in der Höhe h.[17] [18]

$$\rho(h) = \rho_M \cdot e^{-\frac{h}{h_s}}$$

[16] (4), Cw-Wert
[17] (14), Barometrische Höhenformel
[18] (15), Skalenhöhe

Bei einer Temperatur von 15°C ergibt sich für die Skalenhöhe ein Wert von ungefähr 8400m. Dadurch ergibt sich:

$$\Rightarrow \rho(h) = \rho_M \cdot e^{-\frac{h}{8400m}}$$

Die Dichte der Luft auf Meereshöhe ρ_M ergibt sich aus dem Luftdruck p_0, der Gaskonstante R und der Temperatur in Kelvin. Für den Luftdruck auf Meereshöhe liegt der Literaturwert p_0 = 101325 Pa vor.[19]

$$\rho_M = \frac{p_0}{R \cdot T}$$

Die Temperatur in Kelvin ergibt sich aus der Temperatur in Celsius addiert mit 273.15 K.

$$T = 15 + 273.15\,K = 288.15\,K$$

Mit $R = 287.05\ \frac{J}{kg \cdot K}$, T = 288.15 K und p_0 = 101325 Pa:

$$\Rightarrow \rho_M = \frac{101325\,Pa}{287.05\ \frac{J}{kg \cdot K} \cdot 288.15\,K}$$

$$\Leftrightarrow \rho_M \approx 1.225\ \frac{kg}{m^3}$$

So ergibt sich für die höhenabhängige Dichte der Luft $\rho(h)$ bei 15°C:

$$\rho(h) = 1.225\ \frac{kg}{m^3} \cdot e^{-\frac{h}{8400m}}$$

Gravitationskraft und Fallbeschleunigung

Befindet sich ein Körper in der Erdatmosphäre wirkt die Gravitationskraft F_G auf ihn. Die Gravitationskraft zwischen zwei punktförmigen Massen kann mit dem Gravitationsgesetz von Isaac Newton bestimmt werden. Das Gravitationsgesetz beschreibt die Kräfte zwischen zwei

[19] (20), Luftdichte

Körpern 1 und 2 mit den Massen m_1 und m_2, deren Schwerpunkte sich in einem Abstand r voneinander befinden.[20]

$$F_G = G \cdot \frac{m_1 \cdot m_2}{r^2}$$

Dabei ergibt sich der Abstand r aus dem Radius der Erde und dem Abstand zwischen dem Körper und der Erdoberfläche.

Die Gravitationskonstante G beträgt dabei:

$$G = 6{,}674 \cdot 10^{-11} \frac{m^3}{kg \cdot s^2}$$

Unter Vernachlässigung des Luftwiderstands kann aus der berechneten Kraft F_G die Fallbeschleunigung a_g bestimmt werden:

$$F_G = m \cdot \ a \quad |: m$$

$$\Leftrightarrow a_g = \frac{F_g}{m}$$

Fallbeschleunigung mit Luftwiderstand

Mittels der Luftwiderstandskraft F_{LR} und der Gravitationskraft F_G kann nun die eigentliche Kraft F_{ges}, die auf den Körper im Fall mit Luftwiderstand wirkt, berechnet werden.[21] So ergibt sich:

$$F_{ges} = F_G - F_{LR}$$

$$\Rightarrow F_{ges} = m \cdot a_g - \frac{1}{2} \cdot c_w \cdot \rho(h) \cdot A \cdot v^2$$

[20] (5), Gravitationskraft
[21] (3), Luftreibung

Gleichzeitig lässt sich die Beschleunigung des fallenden Körpers mit Luftwiderstand berechnen. So ergibt sich:

$$F_{ges} = m \cdot a_g - \frac{1}{2} \cdot c_w \cdot \rho(h) \cdot A \cdot v^2$$

$$\Rightarrow m \cdot a = m \cdot a_g - \frac{1}{2} \cdot c_w \cdot \rho(h) \cdot A \cdot v^2 \quad |: m$$

$$\Leftrightarrow a_{ges} = a_g - \frac{c_w \cdot \rho(h) \cdot A \cdot v^2}{2 \cdot m}$$

Geschwindigkeit und Position nach dem Zeitschritt Δt

Durch die Beschleunigung a_{ges} unter Berücksichtigung der steigenden Gravitationskraft, Dichte der Luft und dem Luftwiderstand kann die Geschwindigkeit und Position des Körpers nach dem Zeitschritt Δt mithilfe der Methode der kleinen Schritte berechnet werden.

So gilt annähernd für die Geschwindigkeit und Position eines Körpers:

$$v_{neu} = v_{alt} + a_{ges} \cdot \triangle t$$

$$s_{neu} = s_{alt} - v_{neu} \cdot \triangle t$$

Beispielsrechnung

Im Folgenden wird die nächste Position des Fallschirmspringers zu dem Zeitpunkt des Absprungs mit einer Geschwindigkeit von $v = 0\ \frac{m}{s}$ aus einer Höhe von 1000m ausgerechnet. Der Fallschirmspringer hat eine Masse von 80kg und der Fallschirm verfügt über eine Querschnittsfläche von $40\ m^2$ und einen Strömungswiderstandskoeffizient aus der Literatur von 1.33.[22]

[22] (4), Literaturwerte für Cw-Wert

Für die Dichte der Luft in der Höhe von 1000m ergibt sich:

$$\rho(h) = 1.225 \frac{kg}{m^3} \cdot e^{-\frac{h}{8400m}}$$

$$\Rightarrow \rho(1000) = 1.225 \frac{kg}{m^3} \cdot e^{-\frac{1000m}{8400m}}$$

$$\Leftrightarrow \rho(1000) \approx 1.0875 \frac{kg}{m^3}$$

Für die Gravitationskraft F_G mit $G = 6{,}674 \cdot 10^{-11} \frac{m^3}{kg \cdot s^2}$, Masse der Erde $m_1 = 5.972 \cdot 10^{24}$, Masse des Fallschirmspringers $m_2 = 80kg$ und der Höhe $r = 1000m$ ergibt sich:

$$F_G = G \cdot \frac{m_1 \cdot m_2}{r^2}$$

$$\Rightarrow F_G = 6{,}674 \cdot 10^{-11} \frac{m^3}{kg \cdot s^2} \cdot \frac{5.972 \cdot 10^{24} \cdot 80kg}{(6371000 + 1000)^2}$$

$$\Leftrightarrow F_G \approx 785.3160\ N$$

Umgerechnet beträgt die Fallbeschleunigung ohne Luftwiderstand:

$$a_g = \frac{785.316\ N}{80\ kg} \approx 9.8165\ \frac{m}{s^2}$$

Für die Fallbeschleunigung mit Luftwiderstand ergibt sich:

$$a_{ges} = a_g - \frac{c_w \cdot \rho(h) \cdot A \cdot v^2}{2 \cdot m}$$

$$a_{ges} = 9.8165\ \frac{m}{s^2} - \frac{1.33 \cdot 1.0875 \frac{kg}{m^3} \cdot 40\ m^2 \cdot (0 \frac{m}{s})^2}{2 \cdot 80\ kg}$$

$$a_{ges} = 9.8165\ \frac{m}{s^2}$$

Anhand der Ergebnisse erkennt man, dass die Fallbeschleunigung mit Luftwiderstand der Fallbeschleunigung ohne Luftwiderstand entspricht, weil sich der Fallschirmspringer noch nicht bewegt und die Geschwindigkeit $0 \frac{m}{s}$ beträgt.

Die resultierende Geschwindigkeit:

$$v_{neu} = v_{alt} + a_{ges} \cdot \triangle t$$
$$\Rightarrow v_{neu} = 0\frac{m}{s} + 9.8165\ \frac{m}{s^2} \cdot \frac{30}{1000}\ s$$
$$\Leftrightarrow v_{neu} \approx 0.2945\frac{m}{s}$$

Die resultierende Position:

$$s_{neu} = s_{alt} - v_{neu} \cdot \triangle t$$
$$\Rightarrow s_{neu} = 1000 - 0.2945\frac{m}{s} \cdot \frac{30}{1000}\ s$$
$$\Leftrightarrow s_{neu} \approx 999.9912\ \mathrm{m}$$

Somit befindet sich der Fallschirmspringer nach der ersten Approximation auf einer Höhe von 999.9912 m.

Optimierung

Rechenleistung

Um eine realistische Umgebung in der Physik-Engine zu schaffen, nimmt die Gravitationskraft und somit auch die Fallbeschleunigung zu, desto näher sich der Fallschirmspringer der Erde befindet. Somit muss die Gravitationskraft F_G bei jeder neuen Position eines fallenden Körpers in der Physik-Engine neu berechnet werden.
Da es sich jedoch um aufwendige Berechnungen handelt, ist ein möglicher vereinfachender Ansatz, nur bei jeder zehnten Position des Fallschirmspringers die Gravitationskraft zu berechnen. Denn die Werte der Gravitationskraft ändern sich bei kleinen Distanzen nur geringfügig und die Werte weichen deshalb insgesamt nur minimal ab.

Leistungsunabhängige Berechnung – Delta Time

In der Spieleprogrammierung und in der Computergrafik ist es essenziell, dass der Ablauf der Spielschleife hardwareunabhängig ist, da sonst die Positionen der Elemente in Spielen und Berechnungen zu anderen Zeitpunkten je nach Kapazität der Computer ablaufen. Das Konzept Delta Time ermöglicht eine hardwareunabhängige Geschwindigkeit von Abläufen in einer

Spielschleife. Dabei handelt es sich bei Delta Time um die Zeitdifferenz zwischen dem vorhergehenden Bild und dem aktuell gerenderten Bild.[23]

Würde man die implementierte Physik-Engine in einer Spiel-Engine realisieren, muss das Konzept Delta Time notwendig umgesetzt werden, da es sonst zu hardwareabhängigen Unterschieden kommt.

Implementation in Java

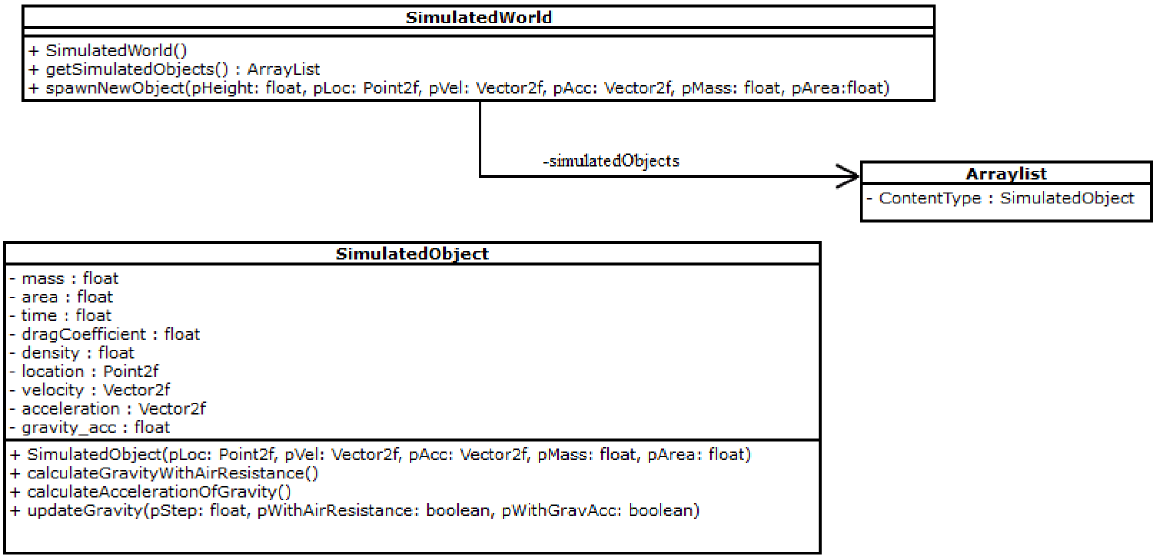


Abbildung 4: Auszug aus dem Implementationsdiagrams

Um eine Physik-Engine mit Computerberechnungen zu modellieren, wurde ein Projekt mit der objekt-orientierten Programmiersprache Java entworfen. Dabei wurde die Physik-Engine so realisiert, dass mehrere Objekte der Klasse SimulatedObject mit unterschiedlichen Parametern erzeugt werden können. Dabei ist mit einem Objekt eine Instanz der Klasse SimulatedObject gemeint. Im Sachzusammenhang würde man von einem Fallschirmspringer sprechen. Dabei können mehrere Fallschirmspringer mit z.B. verschiedenen Massen und Flächen berechnet und visualisiert werden.

Dahingegen sind die Parameter und Einstellungen der Welt für alle, in der Welt simulierten Objekte, gleich. Die SimulatedWorld entspricht dabei der Erde, die als Simulationsraum genutzt wird. Möchte man die Werte der Welt ändern, muss die Simulation neu gestartet werden.

[23] (16), Delta Time

Dabei werden die Instanzen der Klasse SimulatedObject in einer Liste simulatedObjects verwaltet und sind somit aus der Klasse SimulatedWorld erreichbar. Die Klasse SimulatedWorld besitzt mit der Methode *spawnNewObject* ebenfalls die Funktion neue Objekte zu erzeugen. Dazu werden jedoch alle relevanten Parameter wie Höhe, Anfangsgeschwindigkeit, Anfangsbeschleunigung, Masse und Querschnittsfläche benötigt, um ein SimulatedObject zu erzeugen. Diese Werte werden in der GUI angegeben und können beliebig verändert werden.

Bei dem simulierten Fall mit Luftwiderstand wird in jeder Instanz der Klasse SimulatedObject die individuelle Gravitationskraft bzw. Gravitationsbeschleunigung, Dichte der Luft, Geschwindigkeit und nächste Position berechnet. Dazu wird die Methode der kleinen Schritte genutzt.

Anwendung des Eulerschen Polygonzugverfahren

```
public void updateGravity(float pStep, boolean pWithAirResistance, boolean pWithGravAcc){
    if(location_onEarth.y > 0) {

        if(pWithGravAcc) calculateAccelerationOfGravity();
        else gravity_acc = 9.81f;

        if (pWithAirResistance) calculateGravityWithAirResitance();

        time += pStep;

        Vector2f lAcc = new Vector2f(acceleration);
        lAcc.scale(pStep);
        velocity.add(lAcc);

        Vector2f lVel = new Vector2f(velocity);
        lVel.scale(pStep);

        location.sub(lVel);

    }
}
```

Abbildung 5: Methode für die Berechnung der Bewegung

Mithilfe der Methode *updateGravity* kann die nächste Position des Fallschirmspringers in der Physik-Engine approximiert werden. Jedoch wird für die Berechnung vorausgesetzt, dass der Fallschirmspringer sich über der Erde befindet. In der realisierten Physik-Engine kann durch Parameteranpassung eingestellt werden, ob eine steigende Gravitationskonstante (pWithAirGravAcc) und Luftwiderstand (pWtihAirResistance) gelten soll.

Die Schrittweite pStep bestimmt die Genauigkeit, mit der die Berechnung durchgeführt werden soll. Je kleiner die Schrittgröße ist, desto genauer ist auch die Bewegung des Fallschirmspringers. Dementsprechend ist aber die Rechenleistung höher.

Die hauptsächliche Berechnung beginnt, sobald man die momentane Geschwindigkeit mithilfe der momentanen Beschleunigung berechnen möchte. Realisiert wird diese Berechnung in der Physik-Engine, indem man den Beschleunigungsvektor acceleration mit der Schrittgröße Δt skaliert und zu dem Geschwindigkeitsvektor velocity addiert. Damit die ursprüngliche Beschleunigung vor der Skalierung nicht verschwindet, wird anstelle des Vektors acceleration ein lokales Vektorobjekt lAcc erzeugt, in dem die momentane Beschleunigung gespeichert wird.

```
Vector2f lAcc = new Vector2f(acceleration);
lAcc.scale(pStep);
velocity.add(lAcc);
```

Abbildung 6: Berechnung der Geschwindigkeit

Um die nächste Position des Fallschirmspringers zu berechnen, wird von der aktuellen Position der Geschwindigkeitsvektor abgezogen, da der Fallschirmspringer sich auf die Erde zubewegt und sich die Höhe zwischen dem Fallschirmspringer und der Erde verringert. Dafür muss der Geschwindigkeitsvektor aber zuerst mit der Schrittgröße skaliert werden und ebenfalls wie die Beschleunigung lokal gespeichert werden.

```
Vector2f lVel = new Vector2f(velocity);
lVel.scale(pStep);
location.sub(lVel);
```

Abbildung 7: Berechnung der Position

Berechnung der Beschleunigung mit Luftwiderstand

```
private void calculateGravityWithAirResistance(){
    acceleration.y = gravity_acc -
            (drag_coefficient * density * area *
            (float) Math.pow(velocity.y, 2)) / (2 * mass);
}
```

Abbildung 8: Methode für die Berechnung der Beschleunigung mit Luftwiderstand

Um die Bewegung des Fallschirmspringers mit Luftwiderstand zu simulieren, wird die Beschleunigung benötigt, die aus der Differenz der Luftwiderstandskraft und Gravitationskraft geschlossen werden kann. Dafür benötigt man die aktuelle Gravitationsbeschleunigung, den Strömungswiderstandskoeffizient, die Dichte der Luft, die angreifbare Fläche, die Geschwindigkeit und die Masse des Fallschirmspringers. Die Variablen sind lokal in der Klasse SimulatedObject gespeichert.

Berechnung der aktuellen Gravitationskraft

```
private void calculateAccelerationOfGravity(){
    gravity_acc = (float) ((6.67f * Math.pow(10, -11) *
            5.98f * Math.pow(10, 24)) / Math.pow(6.38f * Math.pow(10,6)
            + location_onEarth.y, 2));

    acceleration.y = gravity_acc;
}
```

Abbildung 9: Methode für die Berechnung der steigenden Gravitationskraft

Die aktuelle Gravitationsbeschleunigung kann mit dem Radius bzw. der Position über der Erde und dem Erdradius bestimmt werden. Wird eine konstante Erdbeschleunigung angenommen, wird der Wert auf 9.81 $\frac{m}{s^2}$ gesetzt.

Java-Bibliothek: Vecmath-1.5.1:

In der Physik-Engine wurde die Geschwindigkeit und Beschleunigung mit vektoriellen Größen und die Position mit einem Punkt dargestellt.
Um diese ebenfalls in Java zu integrieren und zu kombinieren, wird die Bibliothek Vecmath Version 1.5.1 verwendet. Diese stellt Vektoren, Matrizen, Punkte und andere mathematische Werkzeuge zur Verfügung. Für die Physik-Engine sind jedoch nur die Klassen Vector2f und Point2f von Bedeutung. Denn mithilfe dieser Klassen können Vektoren erstellt und miteinander verrechnet werden.

In der Physik-Engine werden zwei Vektoren addiert, um die Methode der kleinen Schritte umzusetzen. Dazu wird die Methode *add* der Klasse Vector2f benutzt. Diese addiert zwei Vektoren, indem die X und Y-Werte der zwei Vektoren einzeln addiert werden.

```java
public final void add(Tuple2f var1) {
    this.x += var1.x;
    this.y += var1.y;
}
```

Abbildung 10: Addition zweier Vektoren

Damit der Geschwindigkeitsvektor und Beschleunigungsvektor skaliert werden kann, wird die Methode *scale* der Klasse Vector2f benutzt. Diese skaliert den X und Y-Wert eines Vektors mit einer angegebenen Variabel des Typs float.

```java
public final void scale(float var1) {
    this.x *= var1;
    this.y *= var1;
}
```

Abbildung 11: Skalierung eines Vektors

Erklärung der GUI

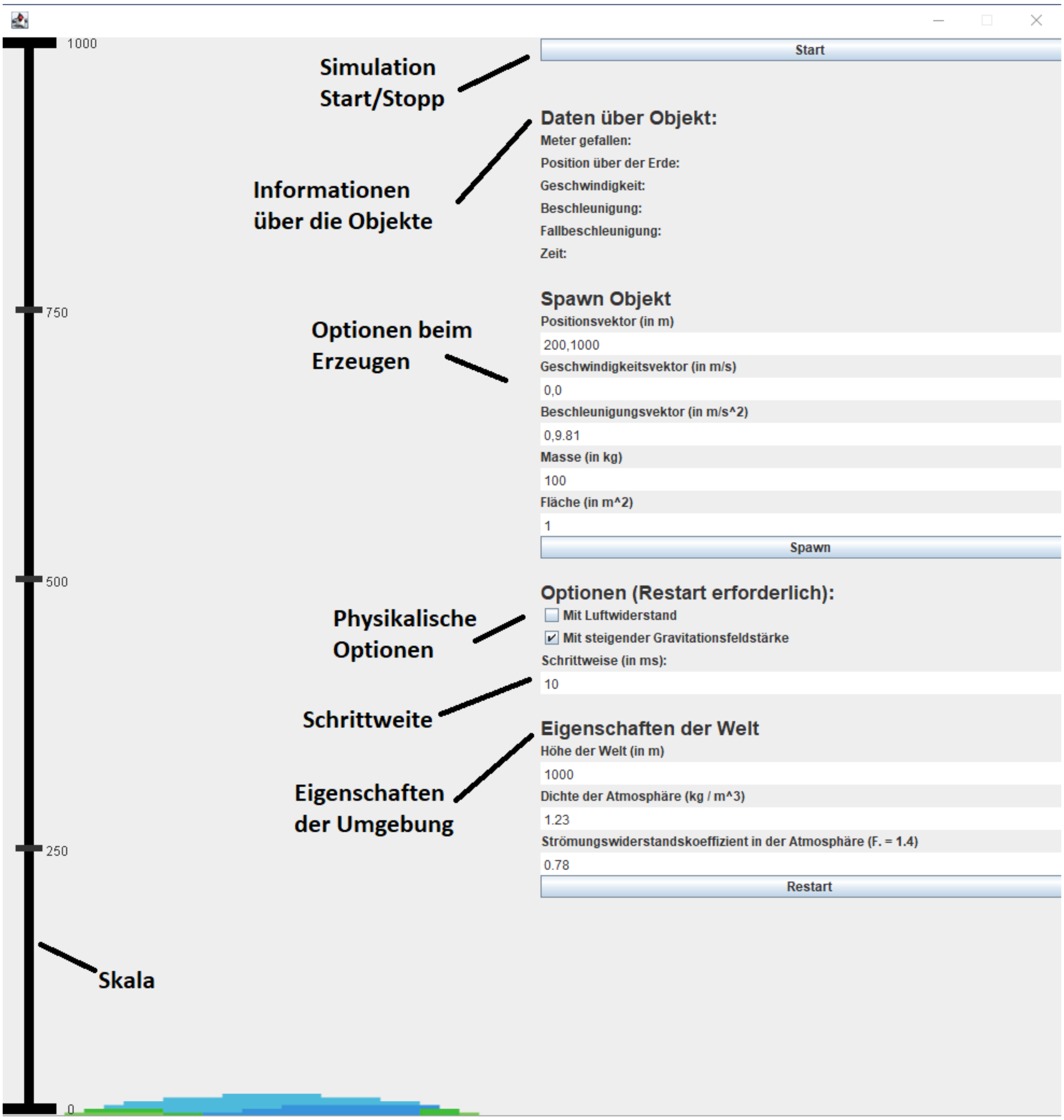


Abbildung 12: Screenshot der GUI

Um den Fall des Fallschirmspringers visuell darzustellen, wurde im Rahmen der Facharbeit eine GUI entworfen. Dabei lässt sich die GUI in zwei Bereiche unterteilen. Auf der linken Seite wird der Sprung des Fallschirmspringers bzw. der Fall eines Objektes visualisiert. Dazu gibt es eine verstellbare Skala und einen Boden zur Orientierung. Die Höhe der Simulation kann dabei variiert werden und die Skala passt sich an die Veränderung an.

Auf der rechten Seite der GUI, befinden sich die genauen Daten des Fallschirmspringers. Dort kann der Nutzer die detaillierte Position, Geschwindigkeit, Beschleunigung, die Fallbeschleunigung und die verstrichene Zeit des simulierten Objekts ablesen. Außerdem gibt es die Möglichkeit, die Simulation zu starten und zu stoppen. Zusätzlich sind Parametereinstellungen für die Atmosphäre und für den Fallschirmspringer verfügbar. Dazu zählen die Höhe der Welt, Dichte der Atmosphäre und der Strömungswiderstandskoeffizient in der Atmosphäre und für den Fallschirmspringer die Position, Geschwindigkeit, Beschleunigung, Masse und Widerstandsfläche. Zusätzlich kann man für die Simulation einstellen, ob man den Fallschirmsprung mit Luftwiderstand und steigender Gravitationsfeldstärke testen möchte.

Sämtliche Daten einer Simulation können ebenfalls in einem Textdokument mit dem Dateiformat CSV (Comma-Separated Values) gespeichert werden. Dadurch können die Daten des Objektes deutlich unterschieden und in anderen Programmen weiterverwendet werden.

Versuchsprotokoll

Thema

Um die simulierten Werte der Physik-Engine zu vergleichen, wird ein Versuch durchgeführt. Dazu wird der Fall eines Kreiskegels betrachtet und ausgewertet. Dabei werden mehrere Messungen mit unterschiedlichen Massen und Flächen durchgeführt.

Dieselben Messungen werden in der Physik-Engine simuliert und verglichen. Durch Parameterfitting, dem Anpassen unbekannter Parameter bis zum optimalen Ergebnis, wird ermöglicht, dass Rückschlüsse auf den Strömungswiderstandskoeffizienten getroffen werden können. Zusätzlich werden die Aspekte der Schrittweite und des Luftwiderstands in der Physik-Engine betrachtet.

Ebenfalls ermöglichen die Ergebnisse des Vergleichs, dass eine Aussage über die Funktionalität der Physik-Engine getroffen werden kann.

Hypothese

Durch die Kreiskegelform des Versuchsobjekts wird das Objekt durch den Luftwiderstand stark verlangsamt, weshalb es eine annähernd maximale Geschwindigkeit erreicht. Jedoch könnte aufgrund der geringen Fallhöhe diese noch nicht erreicht werden.

Ebenfalls wird sich bei einer größeren Masse die Geschwindigkeit erhöhen, da dadurch die massenabhängige Gravitationskraft größer und die Luftwiderstandskraft geringer wird.

Bei einer größeren Querschnittsfläche wird sich die Geschwindigkeit verringern, weil die Luftwiderstandskraft größer wird.

Dagegen wird die Gravitationskraft nur bei einer höheren Masse größer, jedoch nicht bei einer größeren Fläche. Die Auswirkung der Fläche hat keine Auswirkung auf die Gravitationskraft.
Zusätzlich wird vermutet, dass die Schrittweite des Verfahrens einen großen Einfluss auf die Werte der Simulation haben, da diese die Genauigkeit der Simulation bestimmen.
Grundlegend wird angenommen, dass die Werte der Messreihe und der Physik-Engine mit einigen geringen Abweichungen übereinstimmen. Das würde bedeuten, dass die oben beschriebene Methodik annähernd die Werte der Messreihe bzw. den Fall mit Luftwiderstand simuliert.

Verwendete Materialien

- Fotokarton, 1 m^2
- Schere und Klebstoff
- Messstock mit einer Länge von 4m
- Videoaufnahmegerät (Full-HD, 24 FPS)

Versuchsaufbau und Durchführung

Bei dem Versuch wurde Fotokarton benutzt, um den Kreiskegel herzustellen. Dazu schneidet man einen kreisförmigen Bereich aus dem Fotokarton aus. Aus diesem Kreis wird ein Viertel herausgeschnitten und der Rest zu einem Kegel geformt. Schließlich wird der Fotokarton mit Kleber zusammengeklebt.

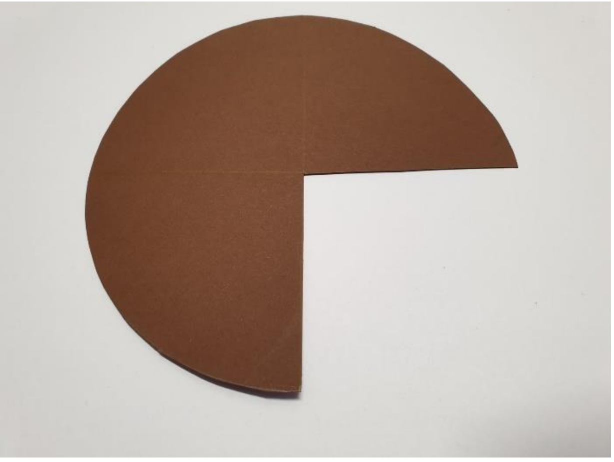


Abbildung 13: Kegelkreisfläche

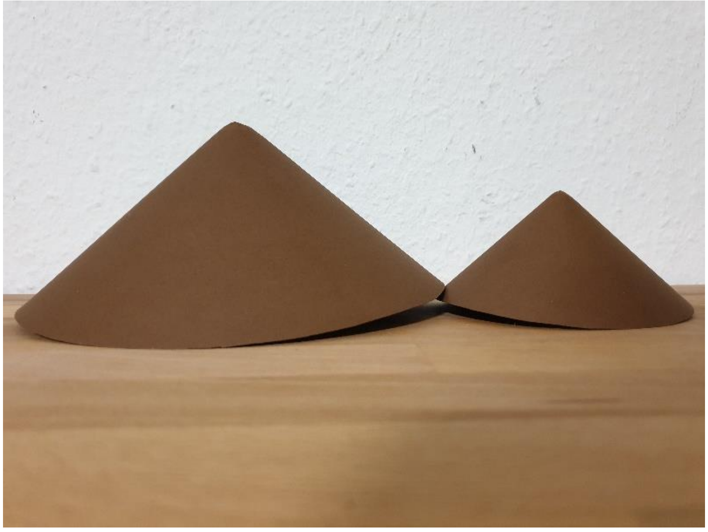


Abbildung 14: Kreiskegel 1 (links) und Kreiskegel 2 (rechts)

Für den Fall des Kreiskegels wird der Messstock mit einer Länge von 4m befestigt, um die fallende Strecke des Kegels zu messen. Das Videoaufnahmegerät wird in einiger Distanz aufgestellt, sodass die gesamte Fallstrecke sichtbar wird. Nun wird der Kreiskegel mit der Spitze nach unten an den obersten Punkt des Messstockes gehalten und fallen gelassen. Der Fall wird mit dem Videoaufnahmegerät in einer Qualität von Full-HD mit 24 FPS aufgenommen.

Um die Auswirkung der Masse festzustellen, wird Kegel 1 mit einer zusätzlichen Masse von 10 Gramm beschwert. Für die Messreihe wurden folgende Werte gemessen bzw. gewogen.

Tabellarische Übersicht

	Kegel 1	Kegel 1 mit Zusatzgewicht	Kegel 2
Radius	12.5 cm	12.5 cm	7.5 cm
Masse	0.008 kg	0.018 kg	0.004 kg

Damit die Werte aus der Physik-Engine belegbar sind, wird dieselbe Messreihe in der Physik-Engine simuliert. Dafür werden die Werte aus der später berechneten Übersichtstabelle verwendet.

Beobachtung

Während der Durchführung des Versuchs ist festzustellen, dass sich die Kegel unterschiedlich schnell bewegen, da die benötigten Zeiten für die Strecke voneinander abweichen.

Dabei fällt auf, dass der schwerste Kegel (Kegel 1 mit Zusatzgewicht) am schnellsten wird und somit auch am wenigsten Zeit benötigt, um die 4 Meter Strecke zurückzulegen. Ebenfalls fällt der Kegel 1 mit zusätzlicher Masse schneller als der Kegel 1 ohne Zusatzgewicht.

Zusätzlich wird deutlich, dass der Kegel 2 schneller fällt als Kegel 1, aber nicht schneller als Kegel 1 mit der zusätzlichen Masse.

Auswertung

Dadurch, dass der Versuch mit einem Videoaufnahmegerät aufgenommen wurde, kann mittels Videoanalyse Rückschlüsse auf die Position der Kegel und deshalb auch auf deren Geschwindigkeit geschlossen werden. Sämtliche Videos wurden mit dem Programm Viana.NET ausgewertet und Diagramme mit der Software Excel erstellt.

Werte der Kreiskegel

Die Fläche der Kreiskegel ergibt sich durch die Berechnung der Fläche des ausgeschnittenen Kreises. Dabei ist zu beachten, dass nur noch dreiviertel der Fläche des Kreises für den Kreiskegel benutzt werden.
Für die Fläche eines Kreises gilt:

$$A = \pi \cdot r^2$$

So ergibt sich für die Fläche des Kreiskegels:

$$A = \frac{3}{4} \cdot \pi \cdot r^2$$

Kegel 1 mit einem Radius von 12.5 cm:

$$\Rightarrow A = \frac{3}{4} \cdot \pi \cdot (12.5\ cm)^2 \approx 368.1554\ cm^2 = 0.03681554\ m^2$$

Kegel 2 mit einem Radius von 7.5 cm:

$$\Rightarrow A = \frac{3}{4} \cdot \pi \cdot (7.5\ cm)^2 \approx 132.5359\ cm^2 = 0.01325359\ m^2$$

Tabellarische Übersicht

	Kegel 1	Kegel 1 mit Besch.	Kegel 2
Radius	12.5 cm	12.5 cm	7.5 cm
Masse	0.008 kg	0.018 kg	0.004 kg
Fläche	0.0368 m^2	0.0368 m^2	0.0132 m^2

Im Vergleich zu den Werten aus dem Versuch, werden die Kegel ebenfalls in der Physik-Engine simuliert. Dazu wird der Strömungswiderstandskoeffizient benötigt, damit der Fall mit Luftwiderstand simuliert werden kann. Dieser wird durch Paramterfitting ermittelt und ungefähr an die Werte aus dem Versuch angepasst. Jedoch unterscheiden sich die Kegel minimal durch ihre Form, weshalb sich unterschiedliche Strömungswiderstandskoeffizienten für Kegel 1 und Kegel 2 ergeben.

Dadurch werden folgende Parameter in der Physik-Engine verwendet:

	Kegel 1	Kegel 1 mit Besch.	Kegel 2
Strömungswiderstandskoeffizient	0.34	0.34	0.31
Masse	0.008 kg	0.018 kg	0.004 kg
Fläche	0.0368 m^2	0.0368 m^2	0.0132 m^2

Durch die Funktion, die es ermöglich die simulierten Daten aus der Physik-Engine weiterzuverarbeiten, können Diagramme mit der Software Excel erstellt werden. Ebenfalls können die Werte aus der Videoanalyse genutzt werden. So ergeben sich folgende Darstellungen aus dem Versuch und aus der Simulation.

Versuch mit Kegel 1

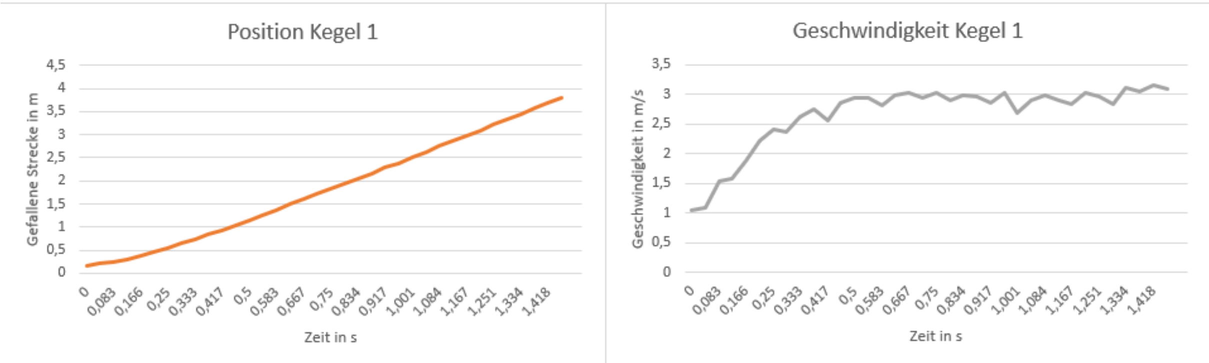


Abbildung 15: Position und Geschwindigkeit, Kegel 1

Simulation in der Physik-Engine mit Werten von Kegel 1

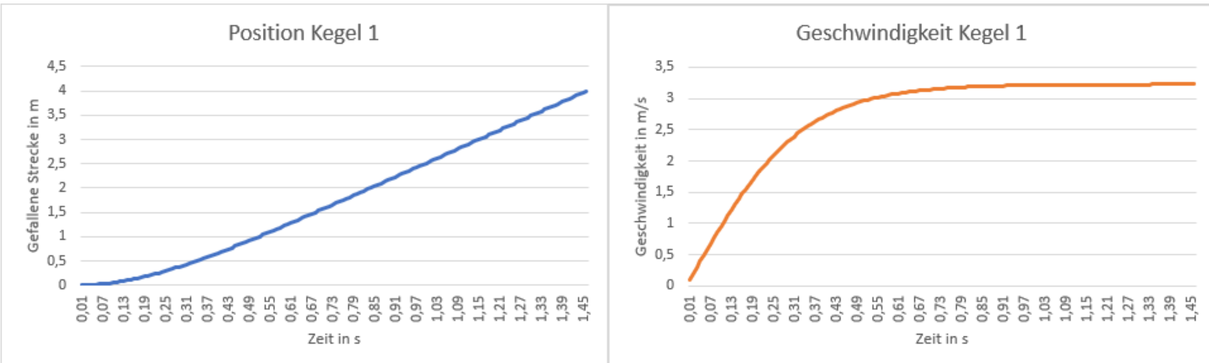


Abbildung 16: Position und Geschwindigkeit aus der Physik-Engine, Kegel 1

Versuch mit Kegel 1 mit Zusatzgewicht

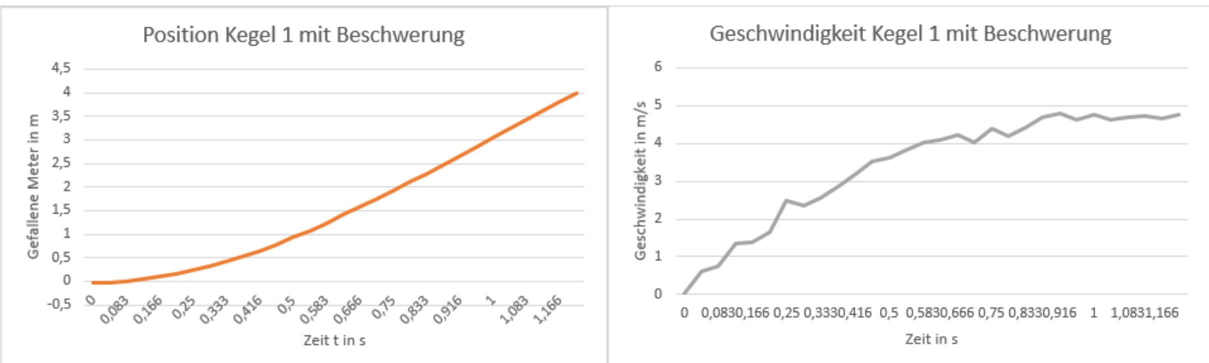


Abbildung 17: Position und Geschwindigkeit, Kegel 1 mit Zusatzgewicht

Simulation in der Physik-Engine mit Werten von Kegel 1 mit Zusatzgewicht

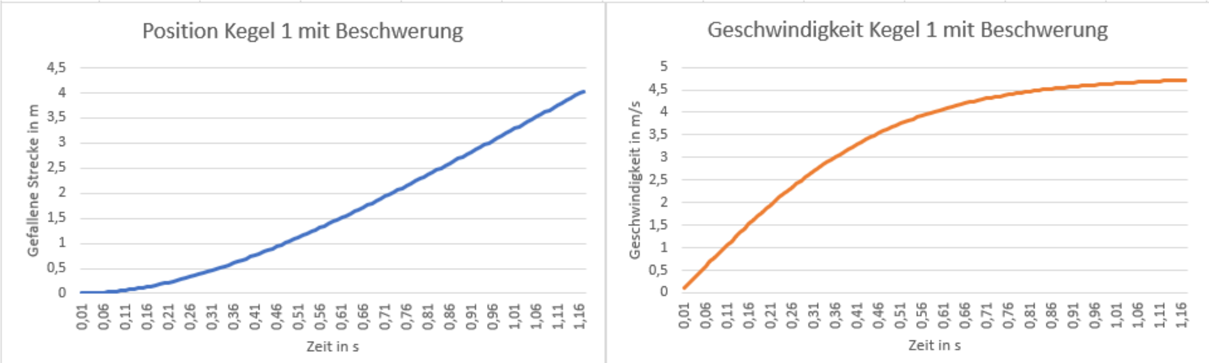


Abbildung 18: Position und Geschwindigkeit aus der Physik-Engine, Kegel 1 mit Zusatzgewicht

Versuch mit Kegel 2

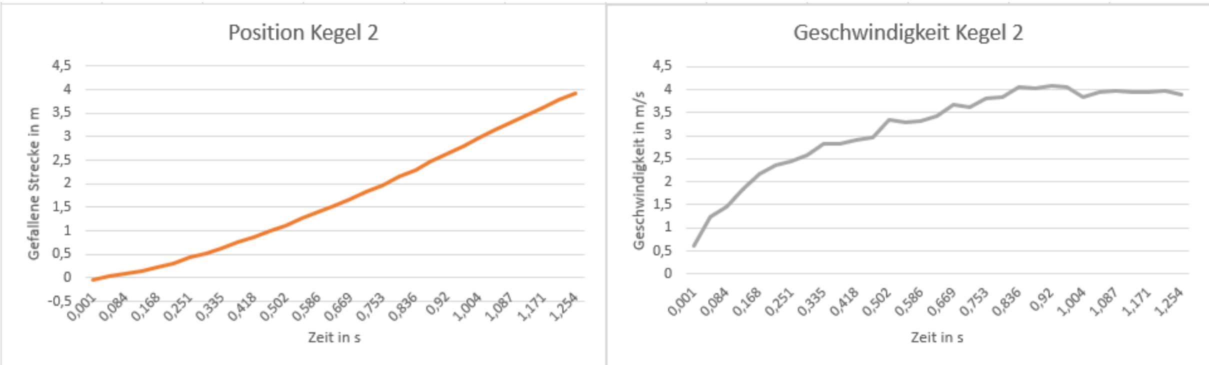


Abbildung 19: Position und Geschwindigkeit, Kegel 2

Simulation in der Physik-Engine mit Werten von Kegel 2

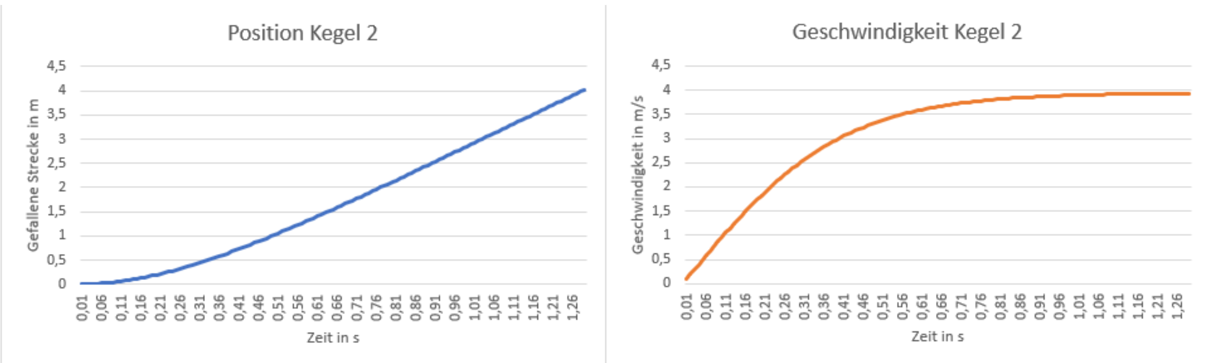


Abbildung 20: Position und Geschwindigkeit aus der Physik-Engine, Kegel 2

Auswertung des Versuchs und der Ergebnisse der Simulation

Anhand der Ergebnisse wird deutlich, dass die Werte aus der Physik-Engine leicht von denen aus dem Versuch abweichen. Hauptursache ist die leicht ungenaue Videoanalyse mit besonders Schwankungen bei der Geschwindigkeit. Betrachtet man jedoch die Position und Geschwindigkeit auf längere Zeit, nähern sich die Werte stark an.

Somit wird deutlich, dass die Simulation realistische Werte für den Versuch bietet. Das bedeutet, dass die beschriebene Methodik erfolgreich umgesetzt und implementiert wurde.

Zur Vereinfachung wird bei den folgenden Diagrammen nur mit den Werten aus der Physik-Engine gerechnet, da dort die Zeitschritte gleichmäßig und die Werte somit vergleichbar sind.

Aufgrund der schwankenden Geschwindigkeit, ergeben die Werte der Videoanalyse keine realistischen Werte für die Beschleunigung, da die Beschleunigung zwischen positiven und negativen Werten schwankt. Jedoch kann durch eine logarithmische Regression eine Funktion für die Geschwindigkeit aufgestellt und Rückschlüsse auf die Beschleunigung getroffen werden.

Vergleich Position und Geschwindigkeit

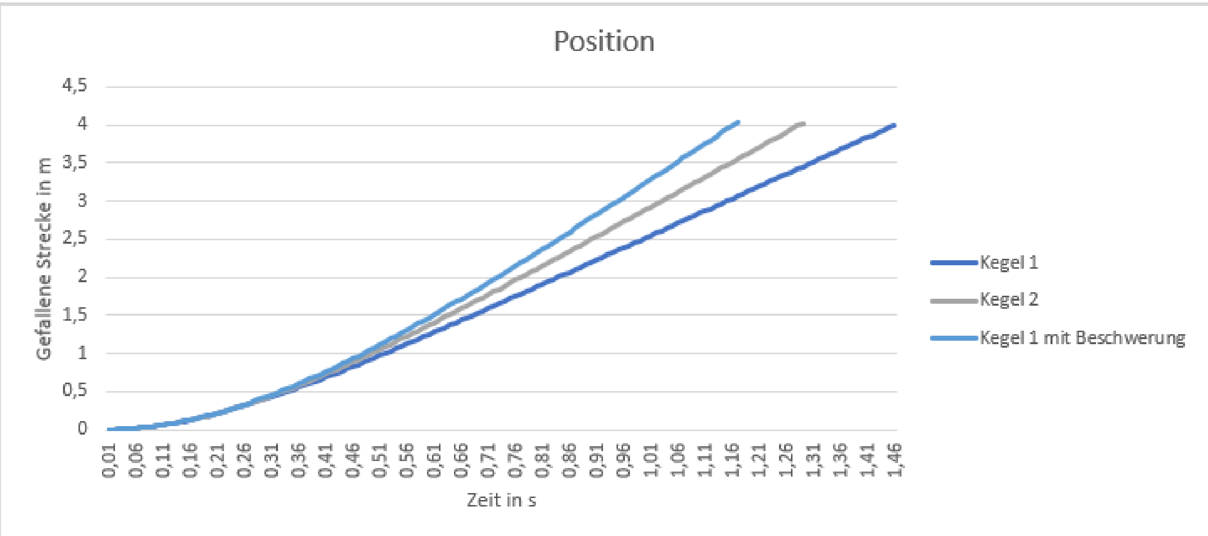


Abbildung 21: Vergleich der Position aus der Physik-Engine

Vergleicht man die simulierten Positionen der Kegel aus der Physik-Engine, fällt auf, dass sämtliche Objekte zu Beginn der Simulation kaum Abweichungen voneinander aufweisen. Erst ungefähr nach 400 Millisekunden ergeben sich visuelle Unterschiede, die im Laufe der Simulation zunehmen. Die unterschiedlichen Ergebnisse werden durch die verschiedenen Geschwindigkeiten erreicht. So erreicht Kegel 1 die Strecke von 4 Metern innerhalb von 1,46

Sekunden. Dagegen benötigt der Kegel 1 mit Zusatzgewicht nur 1,17 Sekunden. Kegel 2 mit der geringeren Fläche benötigt 1,29 Sekunden.

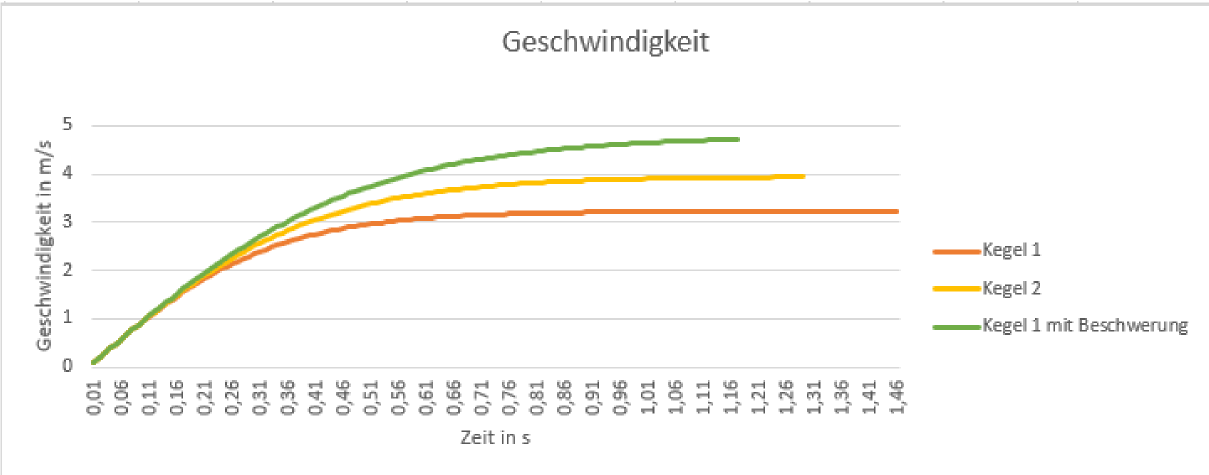


Abbildung 22: Vergleich der Geschwindigkeiten aus der Physik-Engine

Beim Betrachten der unterschiedlichen Geschwindigkeiten der simulierten Kegel wird deutlich, dass die Geschwindigkeiten nicht linear steigen, sondern sich einen bestimmten Wert annähern. Das lässt sich durch die Luftwiderstandskraft erklären, da diese mit der Geschwindigkeit quadratisch zunimmt. Ebenfalls die Parameter Fläche und Masse der unterschiedlichen Kegel haben einen Einfluss auf die Simulation.

So lässt sich feststellen, dass Kegel 1 mit Zusatzgewicht eine höhere Maximalgeschwindigkeit erreicht als ohne eine Beschwerung. Denn die Geschwindigkeit bei Kegel 1 erreicht ihr Maximum bei ungefähr $v = 3.1\frac{m}{s}$. Kegel 1 mit Zusatzgewicht dagegen erreicht das Maximum bei ungefähr $v = 4.7\frac{m}{s}$.

Zusätzlich wird deutlich, dass die Fläche zwischen Kegel 1 und Kegel 2 einen Unterschied ergibt, denn obwohl Kegel 1 vier Gramm schwerer ist als Kegel 2, erreicht Kegel 2 eine höhere Maximalgeschwindigkeit. Diese liegt bei ungefähr $v = 3.9\frac{m}{s}$. Daraus lässt sich schließen, dass die Fläche eine Auswirkung auf die Geschwindigkeit hat. Erklären lässt sich das durch den Strömungswiderstandskoeffizienten, da sich dieser aufgrund der verschiedenen Flächen und Winkel der Kegel unterscheidet. Da dieser bei Kegel 1 bei 3.4 liegt, ist die Luftwiderstandskraft höher als bei dem Strömungswiderstandskoeffizienten von 3.1 bei Kegel 2.

Zusätzlich spielt ebenfalls die Fläche bei der Luftwiderstandskraft eine Rolle, weshalb die Luftwiderstandskraft aufgrund der höheren Fläche des ersten Kegels größer ist als bei Kegel 2.

Theoretisch wäre es möglich die größere Fläche des ersten Kegels durch eine höhere Masse so auszugleichen, dass die selbe Endgeschwindigkeit wie für Kegel 2 erreicht wird. Dafür wird derselbe Strömungswiderstandskoeffizient benötigt, der jedoch in dem durchgeführten Versuch abweicht. Geht man jedoch von demselben Strömungswiderstandskoeffizient aus, kann ein doppelter Radius und somit vierfache Querschnittsfläche durch eine vierfache Masse ausgeglichen werden. So sollten bei einem theoretischen Versuch dieselben Endgeschwindigkeiten erreicht werden.

Auswirkung des Luftwiderstands

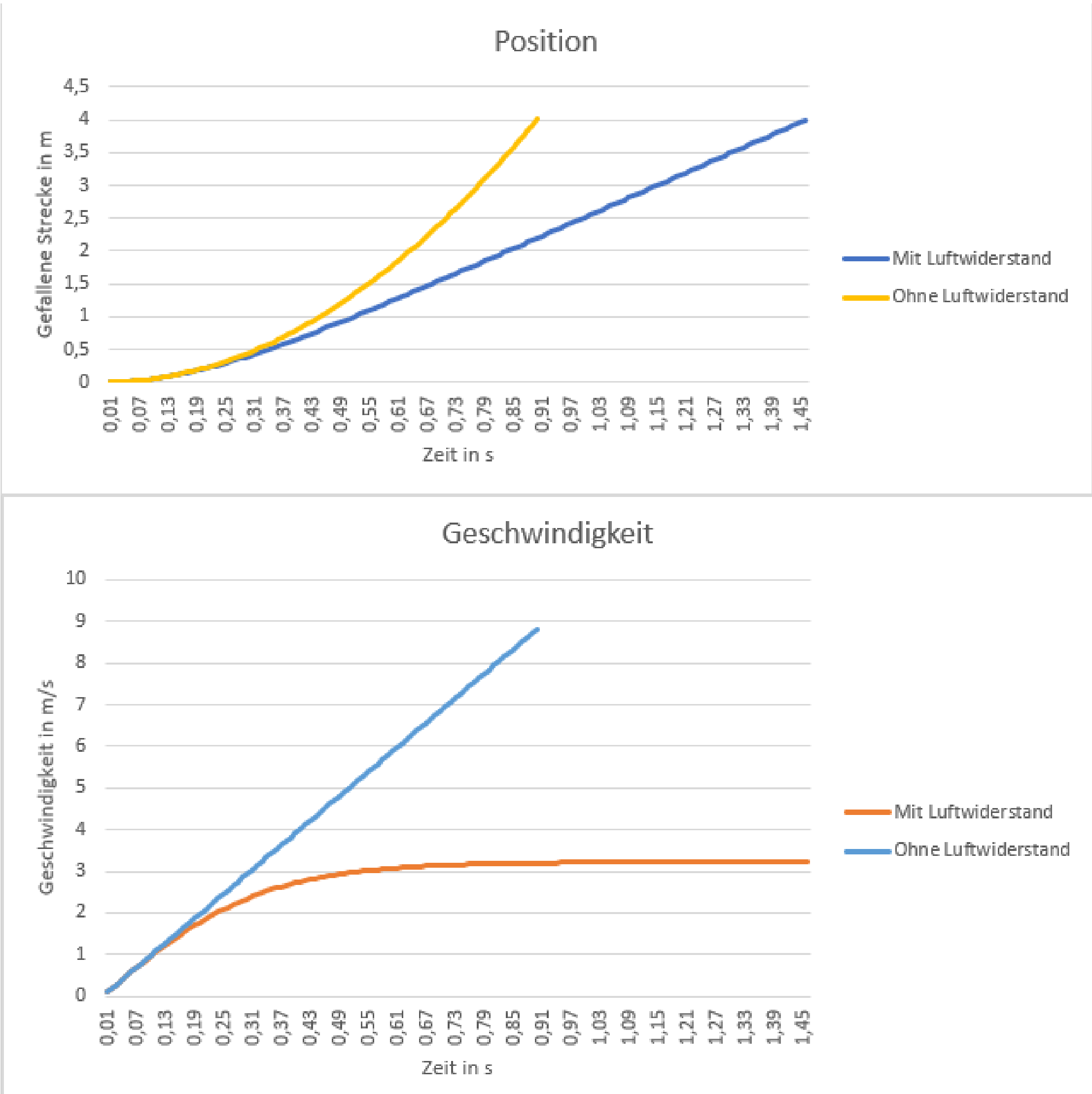
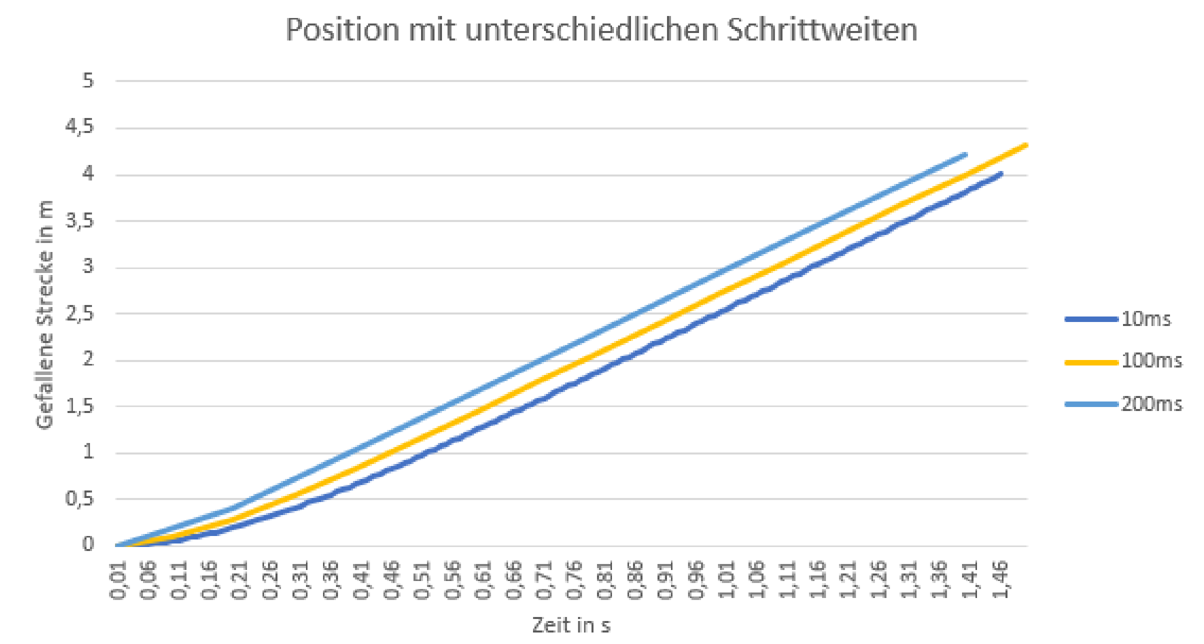


Abbildung 23: Position und Geschwindigkeit mit und ohne Luftwiderstand

Betrachtet man die Ergebnisse aus der Physik-Engine mit und ohne Luftwiderstand, treten massive Abweichungen auf. Denn schon nach nur 0.9 Sekunden hat der simulierte Kegel 1 ohne Luftwiderstand die Strecke von 4 m zurückgelegt. Der Kegel mit Luftwiderstand erreicht dieselbe Distanz erst nach 1,46 Sekunden. Das entspricht einer Differenz von 0.56 Sekunden. Dabei ist ebenfalls zu beobachten, dass die gefallene Strecke bei einem Fall ohne Luftwiderstand exponentiell zunimmt und entfernt sich somit immer weiter von den Werten mit Luftwiderstand. Dabei beschreiben die Werte mit Luftwiderstand eine annähernd lineare Zunahme ab 0.6 Sekunden. Diese Werte ergeben sich durch die maximale Geschwindigkeit, die der Kegel erreicht hat. Durch die steigende Luftwiderstandskraft, die bei steigender Geschwindigkeit quadratisch zunimmt, wird die Geschwindigkeit nicht höher und die zurückgelegte Strecke nimmt linear zu. Ohne Luftwiderstand befindet sich der Kegel jedoch in einem freien Fall, bei dem außer der Schwerkraft keine weiteren Kräfte wirken. Ebenfalls bleibt die Beschleunigung bei geringen Höhen konstant und die Geschwindigkeit nimmt deswegen linear zu.

Auswertung unterschiedlicher Schrittweiten



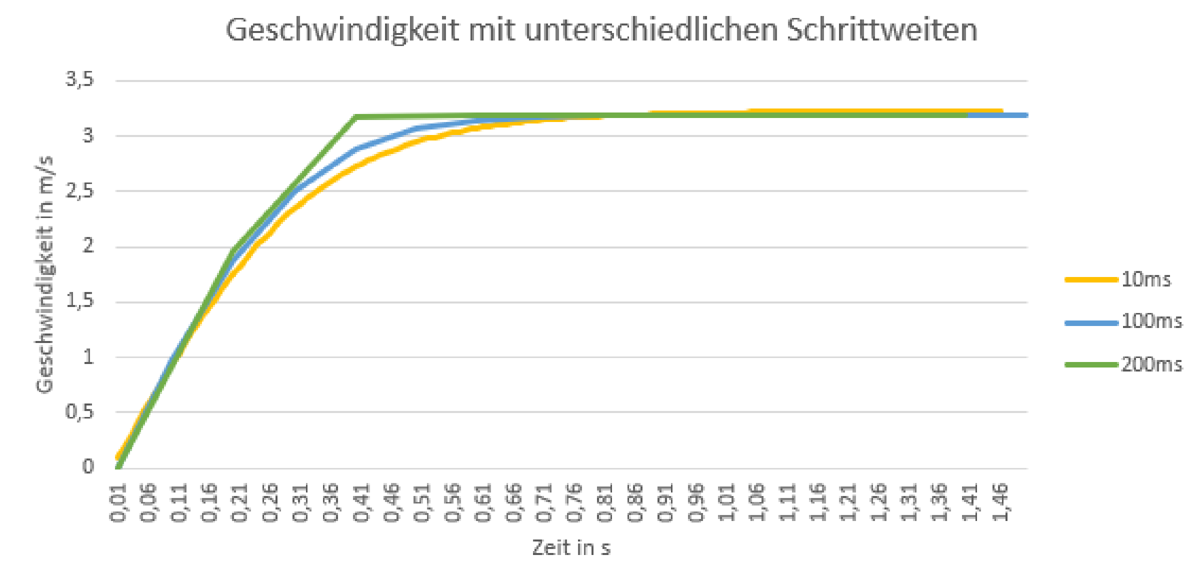


Abbildung 24: Position und Geschwindigkeit bei unterschiedlichen Schrittweisen

Simuliert man denselben Versuch mit den Werten von Kegel 1 ergeben sich unterschiedliche Werte je nach Schrittweite. Dabei sind die Werte genauer, je geringer die Schrittweite ist.

Betrachtet man die Position der simulierten Kegel, fällt schon zu Beginn der Simulation auf, dass sich die Werte stark unterscheiden. Dabei weicht die Position mit einer Schrittweite von 200ms stärker ab als die simulierte Position mit einer Schrittweite von 100ms. Die Abweichung der Position ergibt sich jedoch besonders schon zu Beginn der Simulation und bleibt beinahe parallel im Verlaufe der Simulation.

Betrachtet man die Geschwindigkeit, ergeben sich große Abweichungen besonders zwischen dem Zeitpunkt 0,21 Sekunden und 0,66 Sekunden. Dahingegen sind zu Beginn der Simulation und zu Ende der Simulation die Werte der Geschwindigkeit beinahe gleich. Die maximale Differenz der Geschwindigkeit zwischen der Simulation mit einer Schrittweite von 10ms und einer Schrittweite von 200ms beträgt $0.46\frac{m}{s}$ zu dem Zeitpunkt t = 0.46 Sekunden.

Trotzdem wird deutlich, dass sich die Geschwindigkeit bei einer Schrittweite von 10ms eher den realistischen Werten annähert, da die Werte beinahe eine Kurve beschreiben. Im Gegensatz dazu weichen die Werte aus den anderen Simulationen stark von der optimalen Kurve ab.

Fazit

In der vorliegenden Lernleistung wurde die Methodik erläutert, die für die Implementation einer Physik-Engine notwendig ist. Durch die Programmiersprache Java konnte die Physik-Engine umgesetzt und implementiert werden. Dazu wurde eine benutzerfreundliche GUI entwickelt, sodass die Physik-Engine leicht zu kontrollieren ist.

Damit die Werte aus der Physik-Engine und die Korrektheit der Algorithmen überprüft wird, wurde ein experimenteller Versuch durchgeführt. Dieser beinhaltet den Fall eines Kegels im Fall mit Luftwiderstand. Durch Auswertung mittels Videoanalyse konnten Rückschlüsse auf die Position und Geschwindigkeit gezogen werden. Dadurch lassen sich die Werte aus der Physik-Engine mit denen aus dem Versuch vergleichen und wurden in Diagrammen dargestellt.

So ergab sich aus dem Vergleich, dass die Parameter bzw. Werte Fläche, Masse und der Strömungswiderstandskoeffizient eine Auswirkung auf die Position des Kegels ausüben. So wird der Fall des Kegels durch eine größere Fläche gebremst und durch eine höhere Masse beschleunigt. Ein höherer Strömungswiderstandskoeffizient würde den Kegel ebenfalls bremsen.

Erklären lassen sich diese Beobachtungen durch die Luftwiderstandskraft, die unter anderem durch die Querschnittsfläche, Masse und den Strömungswiderstandskoeffizient geprägt ist. Je nachdem wie die Werte gewählt werden, fällt die Größe der Kraft aus und beeinflusst die Beschleunigung des fallenden Objekts. Denn ohne Luftwiderstand würde der Kegel eine gleichmäßig beschleunigte Bewegung beschreiben und die Geschwindigkeit würde linear zunehmen. Tatsächlich wird der Kegel jedoch durch die Luftwiderstandskraft, die entgegengesetzt der Fallrichtung wirkt, gebremst. Dabei nimmt die Luftwiderstandskraft mit steigender Geschwindigkeit quadratisch zu.

Ebenfalls wurde die Auswirkung der Schrittweite in der Methode der kleinen Schritte getestet. So ergibt sich eine genauere Darstellung, je geringer die Schrittweite Δt gewählt wird. Außerdem fällt auf, dass sich die Werte bei einer Schrittweite von 10ms an die realen Werte aus dem Versuch annähern, wenn man von den Schwankungen des Versuchs absieht.

Zusammenfassend ergibt sich, dass die Simulation in der Physik-Engine realistische Werte für den Versuch im Fall mit Luftwiderstand bietet. Das bedeutet, dass die beschriebene Methodik, besonders die Methode der kleinen Schritte und die Berechnung sämtlicher physikalischer

Parameter funktioniert und eine effektive Approximation bietet. Dadurch wurde die Physik-Engine erfolgreich entwickelt und ermöglicht Simulationen für den Fall mit Luftwiderstand.

Literaturverzeichnis

Internetseiten

Joachim Herz Stiftung, LEIFIphysik:

(1) Google-Suche zu „Methode der kleinen Schritte“
https://www.leifiphysik.de/uebergreifend/allgemeines-und-hilfsmittel/ausblick/methode-der-kleinen-schritte
Zugriff am 22.03.2021

(2) Google-Suche zu „Freier Fall“
https://www.leifiphysik.de/mechanik/freier-fall-senkrechter-wurf/grundwissen/freier-fall
Zugriff am 22.03.2021

(3) Google-Suche zu „Fall mit Luftwiderstand“
https://www.leifiphysik.de/mechanik/reibung-und-fortbewegung/grundwissen/luftreibung
Zugriff am 22.03.2021

(4) Google-Suche zu „Cw-Wert“
https://www.leifiphysik.de/mechanik/stroemungslehre/grundwissen/stroemungswiderstand-und-crmw-wert
Zugriff am 22.03.2021

(5) Google-Suche zu „Gravitationskraft“
https://www.leifiphysik.de/mechanik/gravitationsgesetz-und-feld/grundwissen/gravitationskraft
Zugriff am 22.03.2021

Wikimedia Deutschland – Gesellschaft zur Förderung Freien Wissens e. V.:

(6) Google-Suche zu „PhysX“
https://de.wikipedia.org/wiki/PhysX
Zugriff am 22.03.2021

(7) Google-Suche zu „Physik-Engine“
https://de.wikipedia.org/wiki/Physik-Engine
Zugriff am 22.03.2021

(8) Google-Suche zu „Simulationssoftware“
https://de.wikipedia.org/wiki/Kategorie:Simulationssoftware
Zugriff am 22.03.2021

(9) Google-Suche zu „PPU“
https://en.wikipedia.org/wiki/Physics_processing_unit
Zugriff am 22.03.2021

(10) Google-Suche zu „Usage PPU“
https://de.wikipedia.org/wiki/General_Purpose_Computation_on_Graphics_Processing_Unit
Zugriff am 22.03.2021

(11) Google-Suche zu „Vektoren“
https://de.wikipedia.org/wiki/Vektorielle_Gr%C3%B6%C3%9Fe
Zugriff am 22.03.2021

(12) Google-Suche zu „RK4“
https://de.wikipedia.org/wiki/Klassisches_Runge-Kutta-Verfahren
Zugriff am 22.03.2021

(13) Google-Suche zu „Explizit Euler“
https://de.wikipedia.org/wiki/Explizites_Euler-Verfahren
Zugriff am 22.03.2021

(14) Google-Suche zu „Dichte der Luft berechnen“
https://de.wikipedia.org/wiki/Barometrische_H%C3%B6henformel
Zugriff am 22.03.2021

(15) Google-Suche zu „Skalenhöhe“
https://de.wikipedia.org/wiki/Skalenh%C3%B6he
Zugriff am 22.03.2021

(16) Google-Suche zu „Delta Time“
https://de.wikipedia.org/wiki/Delta_Time
Zugriff am 22.03.2021

Epic Games, Inc.:

(17) Google-Suche zu „UnrealEngine PhysX“
https://docs.unrealengine.com/en-US/InteractiveExperiences/Physics/index.html
Zugriff am 22.03.2021

Unity Technologies:

(18) Google-Suche zu „Unity PhysX“
https://docs.unity3d.com/Manual/UpgradeGuide5-Physics.html
Zugriff am 22.03.2021

Technische Universität Hamburg-Harburg TUHH:

(19) Google-Suche zu „Eulersche Polygonzugverfahren“
https://www.mat.tuhh.de/lehre/material/num_sim/kap2.pdf
Zugriff am 22.03.2021

LUMITOS AG:

(20) Google-Suche zu „Luftdichte“
https://www.chemie.de/lexikon/Luftdichte.html
Zugriff am 22.03.2021

Bildverzeichnis

Item Industrietechnik GmbH:

(1) Google-Suche zu „Eulersche Polygonzugverfahren“
https://glossar.item24.com/glossarindex/artikel/item/euler-verfahren.html
Zugriff am 22.03.2021

Versicherung der selbständigen Erarbeitung

Ich versichere, dass ich die vorliegende Arbeit einschließlich evtl. beigefügter Zeichnungen, Kartenskizzen, Darstellungen u. ä. m. selbstständig angefertigt und keine anderen als die angegebenen Hilfsmittel benutzt habe. Alle Stellen, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen sind, habe ich in jedem Fall unter genauer Angabe der Quelle deutlich als Entlehnung kenntlich gemacht.

Rheinbach, den 24.03.2021

Unterschrift