UNIVERSITÄT
KOBLENZ · LANDAU

Fachbereich 4: Informatik

Simulation von Rauch

Bachelorarbeit

zur Erlangung des Grades Bachelor of Science (B.Sc.)
im Studiengang Computervisualistik

vorgelegt von
Sebastian Gaida

Erstgutachter:	Prof. Dr.-Ing. Stefan Müller (Institut für Computervisualistik, AG Computergraphik)
Zweitgutachter:	Bastian Krayer MSc. (Institut für Computervisualistik, AG Computervisualistik)

Koblenz, im September 2019

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

	Ja	Nein
Mit der Einstellung der Arbeit in die Bibliothek bin ich einverstanden.	☐	☐

.....

(Ort, Datum) (Unterschrift)

Inhaltsverzeichnis

Abstract

In dieser Arbeit wird auf die realistische Simulation von Rauch eingegangen. Dabei bezieht sich die Arbeit hauptsächlich auf die Simulationen von Müller et al.[MCG03] und Ren et al.[RYY+16]. Die Simulation wurde mittels C++, der Open Graphics Libary (OpenGL) und Compute-Shadern erstellt. Hierbei wurde das Smoothed Particle Hydrodynamics (SPH) Verfahren genutzt und die Möglichkeiten zur Beschleunigung auf der Graphics Processing Unit (GPU) untersucht.

This paper deals with the realistic simulation of smoke. The work refers mainly to the simulations of Müller et al.[MCG03] and Ren et al.[RYY+16]. The simulation was created using C++, the Open Graphics Libary and compute shaders. Here the Smoothed Particle Hydrodynamics method was used and the possibilities to accelerate it on the GPU were investigated.

1 Vorwort

Vor dem Beginn der vorliegenden Bachelorarbeit möchte ich mich zunächst bei einigen Personen bedanken die mich während der Arbeit unterstützt haben.

Zunächst einmal bedanke ich mich bei Prof. Dr.-Ing. Stefan Müller und Bastian Krayer MSc. für die großartige Betreuung meiner Arbeit.
Außerdem möchte ich mich bei Pascal Bendler bedanken, der mich tatkräftig beim debugging unterstützt hat.
Ein großes Dankeschön geht auch an den Freund, der mich immer wieder dazu motiviert hat weiter zu arbeiten und nach alternativen Möglichkeiten zu suchen.

2 Einleitung

Das Ziel dieser Arbeit ist es eine möglichst physikalisch korrekte Rauchsimulation zu implementieren. Dazu nutzen wir, dass sich Rauch wie ein Fluid verhält [Sta03], dabei wird die Simulation der physikalischen Basis von Fluiden angenähert. Hierbei wird in dieser Implementation ein Partikelsystem zur Berechnung der physikalischen Eigenschaften genutzt. Echtzeitanwendungen wie die Unity-Engine bieten eine Partikelsimulation an, jedoch beschränkt sich diese lediglich auf das Ausstoßen von Partikeln. Dabei können Partikelinteraktionen, sowie Verhaltensmuster nicht bearbeitet werden.

Die Graphics Processing Unit eignet sich besonders gut zum berechnen parallelisierbarer Rechenoperationen, da sie im Vergleich zur Central Processing Unit (CPU), die nur wenige Kerne besitzt, über tausend Kerne verfügt, die zwar nicht so leistungsfähig sind wie die der CPU, aber dennoch einen signifikante Steigerung der Leistung bieten.

In der Arbeit wird auch darauf eingegangen den genannten Aufwand zu minimieren, dazu wurden zwei Verfahren zur Beschleunigung des Partikelsystems, auf der GPU, implementiert und gegenübergestellt.

Für die Implementation wurde OpenGL genutzt, welches das programmieren auf der GPU deutlich vereinfacht und seit der Version 4.3 auch das verarbeiten von Daten mit Hilfe von Compute-Shadern unterstützt. Für den schnellstmöglichen Zugriff auf diese Daten wird Speicherplatz, in Form von Shader Storage Buffer Object (SSBO), auf der GPU angelegt. Dabei sollte auch auf eine effiziente Nutzung des limitierten Speicherplatzes geachtet werden.

3 State of the Art

Die Simulation von einem Systemen, zur Darstellung von Fluiden, ist ein jahrelange Herausforderung für die Computergrafik. Dabei treten immer wieder die gleichen Problemstellungen auf. Zum einen soll die Simulation physikalisch korrekt sein, um eine für den Beobachter ein möglichst schönes, sowie nachvollziehbarer Ergebnis zu bieten. Schon kleinstes Fehlverhalten können die Immersion zerstören. Andererseits soll das System auch in Echtzeit berechnet werden und dabei auf mögliche Interaktionen reagieren können. Für eine möglichst effiziente Berechnung werden verschiedene Beschleunigungsverfahren verwendet, die ein gutes Ressourcenmanagement in Form der Laufzeit sowie Speicherplatz erfordern.

Die physikalische Grundlage, die Navier-Stokes-Gleichungen, basiert dabei auf den Gleichungen die von Claude Louis Marie Henri Navier und George Gabriel Stokes im 19. Jahrhundert aufgestellt wurden [Wik19]. Diese Gleichungen beschreiben die physikalischen Eigenschaften von Fluiden und werden für die Simulation dieser angewendet. Diese Formeln werden je nach Fluid noch angepasst um speziellere Eigenschaften darzustellen.
Diese Simulation wird meist in Form eines rasterbasierenden Verfahren oder eines Partikelsystems implementiert.

3.1 Vektorfeldverfahren

Beim Vektorfeldverfahren wird die Umgebung in gleichgroße Voxels unterteilt, auch bekannt als Voxelgrid oder eulersches Grid. Bei diesem Vektorfeldverfahren betrachtet man die Partikel nicht direkt sondern einen Masse die in Form des Voxels generalisiert wird. Dabei werden Parameter wie Dichte, Druck und Geschwindigkeit in dem jeweiligen Voxel gespeichert. Die Berechnungen lassen sich in Advektion, Druck, Diffusion und Beschleunigung unterteilen. Die Advektion beschreibt dabei den Strömungstransport, das Übertragen der Bewegungskraft auf ein anliegendes Objekt. Druck wiederum beschreibt die Übertragung von Kräften an benachbarte Partikel, wodurch bei einem zu hohen Druck eine Kraft vom Zentrum weg entsteht und wiederum bei einem Unterdruck eine Kraft zum Zentrum hin. Die Diffusion beschreibt die Viskosität des Fluids. Je nach Anpassung der breitet sich das Fluid stark aus wie zum Beispiel Wasser oder weniger stark wie Lava aus. Bei Beschleunigung handelt es sich um externe Kräfte die auf das Fluid einwirken, dies ist vergleichbar mit der Schwerkraft oder einer Windgeschwindigkeit. Zur Beschleunigung zählt man bei den Vektorfeldern aber auch die Wirbelstärke, die bei Rauch die typischen Turbulenzen verursacht und damit einen signifikanten Einfluss auf die Erscheinung hat.
Wegen der physikalisch präziseren Ergebnisse eignet sich diese Verfahren besonders für Strömungsimulationen in Innenräumen [Pes09], da man Kraft dem System zuführt, diese Kraft wird daraufhin eingefärbt und spiegelt dabei das Fluid wieder.
Bei dieser Methode stellt das Lösen der Gleichungen und die Visualisierung der Ergebnisse die größte Schwierigkeit da. Die Visualisierung erweist sich als Hindernis, da in jedem Voxel Kräfte vorhanden sind. Dabei unterscheidet man in dem Grid unter einem gefärbten Teil und einem nicht sichtbaren Teil der meist Luft repräsentiert 1. Zur Darstellung des Fluids wird meist Volumerendering genutzt. Außerdem ist, wegen der Grid-Architektur des Verfahrens, der Rechenaufwand hoch und lässt sich nur schwer verbessern.

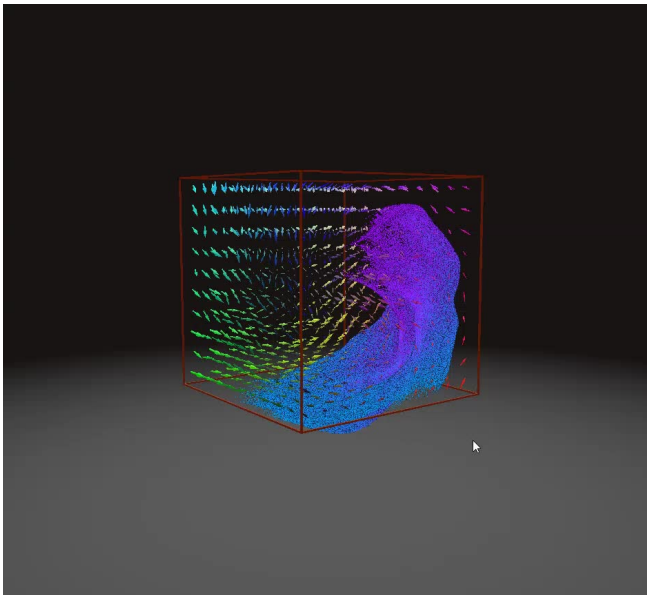

Abbildung 1: Fluidsimulation in Form des Vektorfeldverfahren

3.2 Partikelsystem

Bei dem Verfahren einer Partikelsimulation werden die Partikel einzeln betrachtet, dies bezeichnet man auch lagrangiansches Verfahren. Diese speichern Parameter wie Position und Geschwindigkeit selber ab. Die Berechnungen beschränken sich dabei auf die Dichte, Druck, Viskosität, Auftrieb und Wirbelstärke. Bei den Berechnungen werden die Nachbarpartikel mit einbezogen. Die Dichte beschreibt dabei wie viele Nachbarpartikel Einfluss auf dieses bestimmte Partikel haben und wird als Gewichtung für die Berechnung der Kräfte verwendet. Das typische Verhalten des Fluides, wird aber durch das Zusammenspiel der Kräfte Drucke und Viskosität erzeugt. Dabei sorgt der Druck dafür, dass Partikel sich voneinander wegbewegen und die Viskosität wirkt dem entgegen und führt das Anziehen von Partikel hervor. Der Auftrieb wiederum lässt sich über die Temperatur des Rauches bestimmen, welche je nach Dichte steigt oder sinkt. Zum anderen lässt sich aber die Wirbelstärke nicht so einfach berechnen und stellen somit ein Problem in der Forschung dar. Für die Berechnungen werden die Nachbarpartikel benötigt, welche aber nicht für jeden Partikel bekannt sind und es entsteht ein großer Aufwand, wenn man aus Einfachheit alle Partikel mit einbezieht. Hierbei entstehen viele Möglichkeiten das System zu beschleunigen.
Der Ansatz eines Partikelsystems eignet sich hervorragen zum einbinden in eine Echtzeitanwendung, wie ein Computerspiel oder einer Engine, da man mit der Partikelanzahl die Performance beeinflussen kann. Beim Reduzieren der Partikel sollte eine Anpassung der Parameter erfolgen, da dies

sonst einen signifikanten Einfluss auf das Verhalten des Fluids hat.
Die größten Schwierigkeiten bei einer Rauchsimulation in Form eines Partikelsystems entstehen durch Beschleunigung der Nachbarschaftssuche, sowie den Auftrieb und die Wirbelstärke.

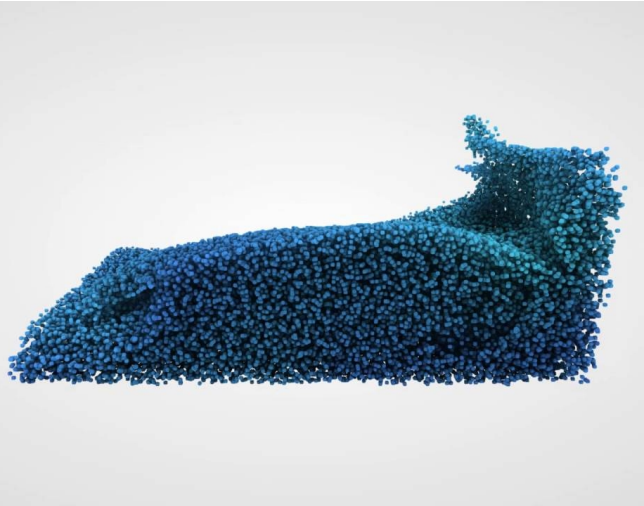

Abbildung 2: Partikelsimulation von Wasser

4 Rauchsimulation

Zur Simulation von Rauch wurde ein Partikelsystem implementiert, wessen physikalische Grundlage auf dem SPH Verfahren basiert, welches Müller [MCG03] 2003 zur Simulation von Fluiden genutzt hat. Dabei handelt es sich um eine Abwandelung der Navier-Stokes-Gleichungen für die Berechnung der Dichte, Druckes und Viskosität. Diese wurden zur Verwendung in einem SPH angepasst. Der Auftrieb, sowie die Temperaturberechung, stammen aus einem Paper von Ren [RYY+16]. Die größten Probleme bei der Implementation bereitete aber die Wirbelstärke, Ren und Pfaff [PTG12] boten eine Formel zur Berechnung dar, welche aber nicht das gewünschte Ergebnis lieferte.

```
1  for all particle i do
2      get neighbor
3      calculate density
4  end for
5  for all particle i do
6      calculate pressure
7  end for
8
9  for all particle i do
10     get neighbor
11     calculare normal
12 end for
13 for all particle i do
14     calculate vorticity
15 end for
16
17 for all particle i do
18     get neighbor
19     calculate pressure force
20     calculate viscosity force
21     calculate vorticity force
22     calculate temperature
23 end for
24 for all particle i do
25     calculate temperature cooldown
26     calculate buoyancy force
27     update velocity
28 end for
29
30 for all particle i do
31     apply velocity on position
32 end for
```

Abbildung 3: Updateschleife der Physik

Symbol	Bedeutung	Format
m_i	Masse des Partikel i	float
r_i	Position des Partikel i	vec3
r_{ij}	Abstandsvektor von $r_i - r_j$	vec3
v_i	Geschwindigkeitsvektor des Partikel i	vec3
W_{ij}	Gewichtungsfunktion, kurz für $W(r_i - r_j)$	float
∇W_{ij}	Gradienten-Gewichtungsfunktion	vec3
$\nabla^2 W_{ij}$	Laplace-Gewichtungsfunktion	float
ρ_i	Dichte des Partikel i	float
ρ_0	Ruhedichte im Allgemeinen	float
k	Steifheit des Fluids	float
p_i	Druck des Partikel i	float
$f_i^{pressure}$	Druckkraft des Partikel i	vec3
μ	Viskosität des Fluids	float
ν_i	Viskosität des Partikel i	float
$f_i^{viscosity}$	Viskositätskraft des Partikel i	vec3
T_i	Temperatur des Partikel i	float
D_r	Zeit zum halbieren der Temperatur	float
b	Up-Vektor	vec3
D_c	Wärmeleitfähigkeitsfunktion des Fluids	float
c	Wärmeleitfähigkeit	float
C_b	Auftriebs-Koeffizient	float
$a_{b,i}$	Auftriebsbeschleunigung	vec3
n_i	Normale des Partikel i	vec3
C_N	Nutzer definierter Schwellenwert	float
y	Zahl die nahezu 0 ist	float
$f_i^{buoyancy}$	Auftriebskraft des Partikel i	vec3
β	Nutzer definierter Wert	float
ω_i	Wirbelstärke des Partikel i	vec3
f_i^{vortex}	Wirbelstärkenkraft des Partikel i	vec3
g	Gravitationskraft	vec3
δt	Zeit seit letzter Iteration	float
δ	veränderte Wert im Abstand von δt	
h	Radius	float

Abbildung 4: Bedeutung aller Symbole der Berechnungen

OpenGL Open Graphics Libary

SPH Smoothed Particle Hydrodynamics

GPU Graphics Processing Unit

CPU Central Processing Unit

SSBO Shader Storage Buffer Object

Abbildungsverzeichnis

Literatur

[MCG03] MÜLLER, Matthias ; CHARYPAR, David ; GROSS, Markus: Particle-based fluid simulation for interactive applications. In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* Eurographics Association, 2003, S. 154–159

[Pes09] PESCHEL, Franz: *Simulation und Visualisierung von Fluiden in einer Echtzeitanwendung mit Hilfe der GPU*. `http://aleph1.uni-koblenz.de/F?func=find-b&find_code=wrd&request=franz+peschel`. Version: 2009. – [Online; Stand 28. August 2019]

[PTG12] PFAFF, Tobias ; THUEREY, Nils ; GROSS, Markus: Lagrangian vortex sheets for animating fluids. In: *ACM Transactions on Graphics (TOG)* 31 (2012), Nr. 4, S. 112

[RYY+16] REN, Bo ; YAN, Xiao ; YANG, Tao ; LI, Chen-feng ; LIN, Ming C. ; HU, Shi-min: Fast SPH simulation for gaseous fluids. In: *The Visual Computer* 32 (2016), Nr. 4, S. 523–534

[Sta03] STAM, Jos: Real-time fluid dynamics for games. In: *Proceedings of the game developer conference* Bd. 18, 2003, S. 25

[Wik19] WIKIPEDIA: *Navier-Stokes-Gleichungen — Wikipedia, Die freie Enzyklopädie*. `https://de.wikipedia.org/w/index.php?title=Navier-Stokes-Gleichungen&oldid=191399294`. Version: 2019. – [Online; Stand 28. August 2019]