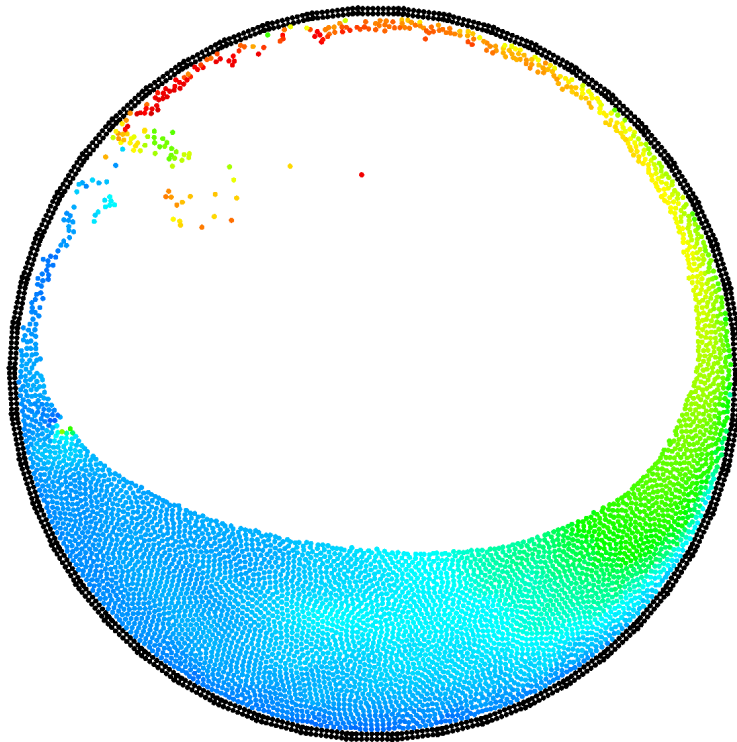


Einfacher SPH Flüssigkeitssimulator mit Beschleunigter Nachbarschaftssuche

Thierry Meiers

Bachelor Projekt



Albert-Ludwigs-Universität Freiburg
Technische Fakultät
Graphische Datenverarbeitung

17. August 2024

Inhaltsverzeichnis

1	Einführung	1
2	Navier-Stokes-Gleichungen	1
2.1	Impulsgleichung	2
2.2	Kontinuitätsgleichung	3
2.3	Nicht komprimierbare Fluide	3
3	Smoothed Particle Hydrodynamics	3
3.1	Kernel Funktion	4
3.2	Kernel Ableitung	5
3.3	Kontinuitätsgleichung	6
3.4	Impulsgleichung	6
3.5	Berechnung der Partikeleigenschaften	7
4	Simulationsschritt	7
5	Nachbarschaftssuche	8
5.1	Einfügen von Partikeln	9
5.2	Updaten der Partikel Positionen	9
5.3	Ermitteln von Nachbarn	9
6	Implementierung	10
6.1	Tests	10
6.2	Tools	11
6.3	Parallelisierung des Simulationsschrittes	12
7	Analysen	12
7.1	Säulenexperiment	12
7.2	Nachbarschaftssuche	15
7.3	Prozessverarbeitung	15
8	Schlussfolgerung	16
9	Bibliographie	17

1 Einführung

Fluid Simulationen haben sich zu einem unverzichtbaren Werkzeug in zahlreichen Bereichen unserer Gesellschaft entwickelt. Sie ermöglichen das präzise Simulieren von Flüssigkeiten und Gasen in verschiedenen Umgebungen, wodurch das Verhalten dieser Stoffe und ihre Interaktionen mit der Umgebung untersucht werden können. Aufgrund ihrer Vielseitigkeit kommen Fluid Simulationen in einer großen Bandbreite von Anwendungsgebieten zum Einsatz, darunter die Optimierung von Fahrzeugen und Gebäuden, Wettervorhersagen, medizinische Forschung, die Erzeugung realistischer Effekte in Filmen und Spielen, industrielle Prozesse und Energieerzeugung. Diese Simulationen tragen wesentlich dazu bei, Designs zu verbessern, die Effizienz zu steigern und komplexe Strömungen besser zu verstehen.

Die Allgegenwärtigkeit und Bedeutung dieser Technologie in verschiedenen Branchen wecken das Interesse an dem Thema. Insbesondere die Verbindung von Physik und Informatik weckt zusätzlich persönliche Interessen und ist eine Motivation zur Arbeit an diesem Projekt. Das Hauptziel ist die Implementierung einer Fluid Simulation mit der Smoothed Particle Hydrodynamics (SPH) Methode, um ein tieferes Verständnis für die Komponenten und Mechanismen von SPH zu entwickeln. Um die Performance zu verbessern wird zusätzlich eine beschleunigte Nachbarschaftssuche implementiert. Dazu werden zu beginn alle Komponenten von SPH implementiert und auf Korrektheit getestet. Daraufhin werden die einzelnen Komponenten zusammengefügt, woraus sich der Simulationsschritt ergibt. Durch Analysen wird schließlich das Verhalten des Fluiden für verschiedene Einstellungsparameter studiert und interpretiert. Dadurch können die Parameter von SPH richtig gesetzt werden um die Stabilität der Simulation zu optimieren.

In diesem Bericht folgt zuerst eine Motivation. In Abschnitt 2 werden die Navier-Stokes-Gleichungen vorgestellt. Sie bilden die Grundlage aller Methoden zur Simulation von Fluiden und Gasen. In Abschnitt 3 folgt eine Übersicht aller Komponenten der SPH-Methode, die zur Implementierung der Simulation verwendet wurden. Des weiteren folgt in Abschnitt 4 eine Übersicht über den verwendeten Simulationsschritt. Daraufhin wird in Kapitel 5 die Funktion der beschleunigten Nachbarschaftssuche erläutert. In Kapitel 6 werden einige erwähnenswerte Implementierungen erläutert. Schlussendlich wird in Kapitel 7 die Analysen des Simulation vorgestellt.

2 Navier-Stokes-Gleichungen

Die Navier-Stokes-Gleichungen sind fundamentale Gleichungen in der Fluidmechanik, die das Verhalten von strömenden Flüssigkeiten und Gasen beschreiben. Sie bilden ein System von nichtlinearen partiellen Differentialgleichungen zweiter

Ordnung. Diese basieren auf den Prinzipien der Erhaltung von Masse, Impuls und Energie und modellieren die Bewegung des Fluids, indem sie die Einflüsse von Druck, Viskosität und externen Kräften auf das Strömungsverhalten berücksichtigen.

In ihrer klassischen Form sind sowohl die Impulsgleichung (1) als auch die Kontinuitätsgleichung (8) spezifisch für Newtonsche Fluide formuliert. Newtonsche Fluide zeichnen sich durch ein lineares viskoses Fließverhalten aus, bei dem die Viskosität unabhängig von der Schergeschwindigkeit ist.

2.1 Impulsgleichung

$$\rho(\nabla\vec{v} + (\vec{v} \cdot \vec{\nabla})\vec{v}) = -\vec{\nabla}p + \eta\vec{\nabla}^2\vec{v} + (\lambda + \eta)\vec{\nabla}(\vec{\nabla} \cdot \vec{v}) + \rho\vec{f} \quad (1)$$

Gleichung (1) beschreibt die Impulserhaltung für komprimierbare Fluide. Diese entspricht dem zweiten Newtonschen Gesetz und kann dementsprechend hergeleitet werden:

$$\begin{aligned} \vec{F} &= \Delta m \cdot a = \Delta m \cdot \nabla\vec{v} \\ \vec{F} &= \rho \cdot \Delta V \cdot \nabla\vec{v} \\ \vec{F} &= \rho \cdot \Delta V (\nabla\vec{v} + (\vec{v} \cdot \vec{\nabla})\vec{v}) \end{aligned} \quad (2)$$

Die Masse eines Objektes kann durch $m = \rho \cdot V$ bestimmt werden. Die Geschwindigkeitsänderung ergibt sich aus der lokalen zeitlichen Beschleunigung $\nabla\vec{v}$ und der konvektiven Beschleunigung $(\vec{v} \cdot \vec{\nabla})\vec{v}$, also aus der Änderung der Geschwindigkeit aufgrund der Bewegung der Flüssigkeit selbst.

Wir benötigen nun die Kräfte \vec{F} . Diese setzen sich aus verschiedenen Kräften zusammen, welche auf einen Fluid wirken. Die Druckkraft (3), die Reibungskraft (4), die Kompressionskraft (5) und die Kräfte, die von außen wirken (6).

$$\vec{F}_{Druck} = -\vec{\nabla}p \cdot \Delta V \quad (3)$$

$$\vec{F}_{Reib} = \eta\vec{\nabla}^2\vec{v} \cdot \Delta V \quad (4)$$

$$\vec{F}_{Kompress} = (\lambda + \eta)\vec{\nabla}(\vec{\nabla} \cdot \vec{v}) \cdot \Delta V \quad (5)$$

$$\vec{F}_{ext} = \rho\vec{f} \cdot \Delta V \quad (6)$$

$$\vec{F} = \vec{F}_{Druck} + \vec{F}_{Reib} + \vec{F}_{Kompress} + \vec{F}_{ext} \quad (7)$$

Wir setzen (7) in die Gleichung (2) ein und erhalten somit:

$$-\vec{\nabla} p \cdot \Delta V + \eta \vec{\nabla}^2 \vec{v} \cdot \Delta V + (\lambda + \eta) \vec{\nabla}(\vec{\nabla} \cdot \vec{v}) \cdot \Delta V + \rho \vec{f} \cdot \Delta V = \rho \cdot \Delta V (\nabla \vec{v} + (\vec{v} \cdot \vec{\nabla}) \vec{v})$$

Durch das Dividieren beider Seiten mit ΔV erhält man schließlich die Gleichung (1).

$$-\vec{\nabla} p + \eta \vec{\nabla}^2 \vec{v} + (\lambda + \eta) \vec{\nabla}(\vec{\nabla} \cdot \vec{v}) + \rho \vec{f} = \rho (\nabla \vec{v} + (\vec{v} \cdot \vec{\nabla}) \vec{v})$$

2.2 Kontinuitätsgleichung

Des Weiteren folgt die Kontinuitätsgleichung (8), welche die Erhaltung der Masse innerhalb eines Fluides beschreibt.

$$\Delta \rho + \vec{\nabla} \cdot (\rho \vec{v}) = 0 \quad (8)$$

Sie stellt sicher, dass die zeitliche Änderung der Dichte $\Delta \rho$ und die Divergenz des Massenflusses $\vec{\nabla} \cdot (\rho \vec{v})$ sich so ausgleichen, dass die Gesamtmasse konstant bleibt. In einem stationären Strömungsfeld ($\Delta \rho = 0$) besagt die Gleichung, dass der Massenzufluss und -abfluss ausgeglichen sein müssen.

2.3 Nicht komprimierbare Fluide

Für nicht komprimierbare Fluide vereinfachen sich die Impulsgleichung (1) und die Kontinuitätsgleichung (8),

$$\rho (\nabla \vec{v} + (\vec{v} \cdot \vec{\nabla}) \vec{v}) = -\vec{\nabla} p + \eta \vec{\nabla}^2 \vec{v} + \rho \vec{f} \quad (9)$$

da $(\lambda + \eta) \vec{\nabla}(\vec{\nabla} \cdot \vec{v}) = 0$ für nicht komprimierbare Fluide und,

$$\vec{\nabla} \vec{v} = 0 \quad (10)$$

da $\Delta \rho = 0$ und $\vec{\nabla} \rho = 0$

3 Smoothed Particle Hydrodynamics

Smoothed Particle Hydrodynamics (SPH) ist eine numerische Methode zur Approximation der Navier-Stokes-Gleichungen. Im Verlauf des Kapitels werden die einzelnen Bestandteile von SPH erläutert und die Verbindung von SPH und den Navier-Stokes-Gleichungen beleuchtet.

Das Konzept von SPH lässt sich allgemein als eine Methode zur Diskretisierung von räumlichen Feldgrößen und räumlichen Differentialoperatoren verstehen. Die

zentrale Idee von SPH ist es, Partikel einzusetzen, die Proben von Feldgrößen enthalten und eine Kernel Funktion zu nutzen, um kontinuierliche Felder zu approximieren. Ein Partikel besitzt Eigenschaften wie die Masse m_i , die Position x_i , die Geschwindigkeit v_i und den Druck p_i .

3.1 Kernel Funktion

Die Kernel Funktion $W_{ij} = W(x_i - x_j, h)$ ist eine Funktion zur Glättung der Räumlichen Diskretisierung, wobei h als die Glättungslänge des Kernels bezeichnet wird. Diese steuert, wie stark der Wert an Position x_j Einfluss auf den Wert an Position x_i nimmt. Dabei soll die Kernel Funktion zweifach ableitbar sein und folgende Eigenschaften besitzen: $\forall x_i, x_j \in \mathbb{R}^d, h \in \mathbb{R}^+ \rightarrow \int_{\mathbb{R}^d}$

$$W(x_i - x'_j, h) dx'_j = 1 \quad (11)$$

$$\lim_{h' \rightarrow 0} W(x_i - x_j, h') = \delta(r) \quad (12)$$

$$W(x_i - x_j, h) \geq 0 \quad (13)$$

$$W(x_i - x_j, h) = W(-(x_i - x_j), h) \quad (14)$$

$$W(x_i - x_j, h) = 0 : |x_i - x_j| \geq h \quad (15)$$

wobei $\delta(x)$ die Delta-Verteilung ist. In diesem Projekt wird die Cubic Spline Kernel Funktion verwendet, wie in [TODO] beschrieben:

$$W_{ij} = W(x_j - x_i, h) = \alpha \begin{cases} (2 - \frac{\|x_j - x_i\|}{h})^3 - 4(1 - \frac{\|x_j - x_i\|}{h})^3 & 0 \leq \frac{\|x_j - x_i\|}{h} \leq 1 \\ (2 - \frac{\|x_j - x_i\|}{h})^3 & 1 \leq \frac{\|x_j - x_i\|}{h} \leq 2 \\ 0 & \frac{\|x_j - x_i\|}{h} \geq 2 \end{cases}$$

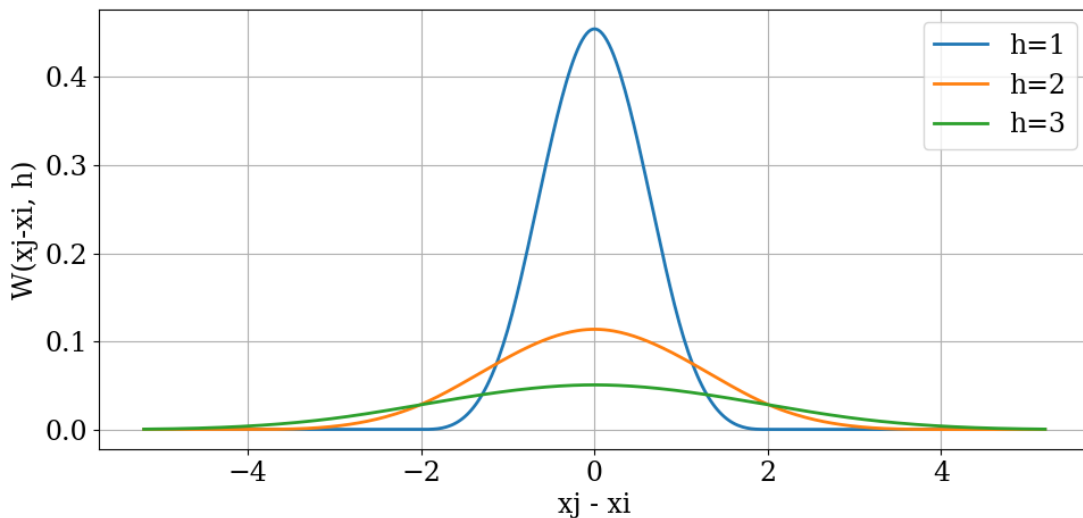


Abbildung 1: Plot der Kernel Funktion für Verschiedene h

Dabei ist α ein Normalisierungsfaktor, der in verschiedenen Dimensionen wie folgt definiert ist: 1D: $\alpha = \frac{1}{6h}$, 2D: $\alpha = \frac{5}{14\pi h^2}$, 3D: $\alpha = \frac{1}{4\pi h^3}$

3.2 Kernel Ableitung

Die Ableitung der Kernel Funktion W_{ij} spielt eine entscheidende Rolle bei der Berechnung von Gradienten in der SPH. Sie wird insbesondere zur Berechnung der Druck- und Geschwindigkeitsgradienten verwendet. Die Ableitung der Kernel Funktion $\nabla_i W_{ij}$ bezüglich der Position des Partikels i ist wie folgt definiert:

$$\nabla W_{ij} = \nabla W(x_j - x_i, h) = \alpha \frac{x_j - x_i}{\|x_j - x_i\| h} \begin{cases} -3(2 - \frac{\|x_j - x_i\|}{h})^2 + 12(1 - \frac{\|x_j - x_i\|}{h})^2 & 0 \leq \frac{\|x_j - x_i\|}{h} < 1 \\ -3(2 - \frac{\|x_j - x_i\|}{h})^2 & 1 \leq \frac{\|x_j - x_i\|}{h} < 2 \\ 0 & \frac{\|x_j - x_i\|}{h} \geq 2 \end{cases}$$

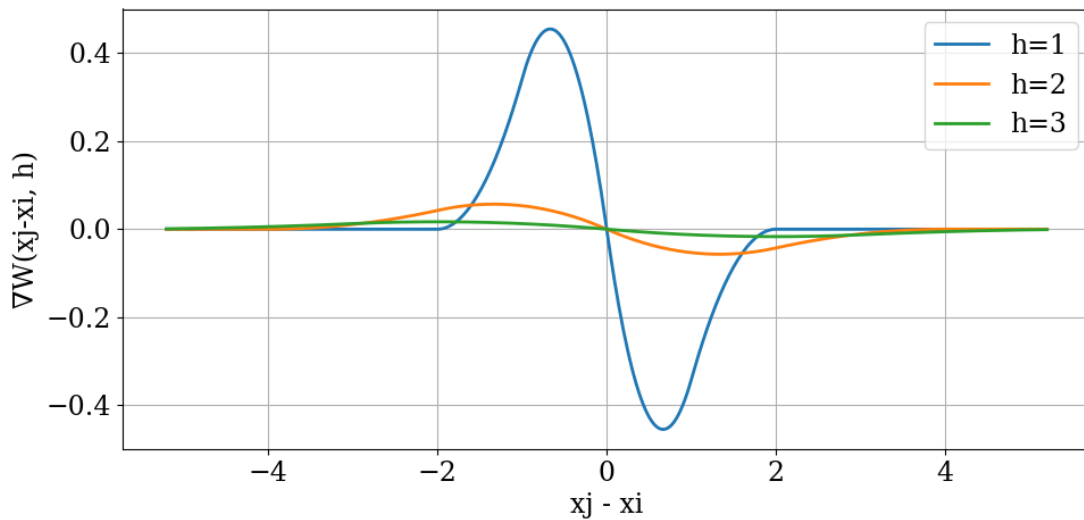


Abbildung 2: Plot der Kernel Ableitung Funktion für verschiedene h

Die Kernel Ableitung ist entscheidend, weil sie in mehreren wichtigen Gleichungen verwendet wird, wie z.B. bei der Berechnung der Kontinuitätsgleichung, der Impulsgleichung, insbesondere bei der Berechnung der Druck- und Viskositätsbeschleunigung.

3.3 Kontinuitätsgleichung

Die Kontinuitätsgleichung beschreibt die Erhaltung der Masse und wird in der SPH-Methode wie folgt diskretisiert:

$$\frac{d\rho_i}{dt} = \sum_j m_j (v_i - v_j) \cdot \nabla_i W_{ij}$$

Dabei ist ρ_i die Dichte des Partikels i , m_j die Masse des Partikels j , v_i und v_j die Geschwindigkeiten der Partikel i und j , und $\nabla_i W_{ij}$ das Gradientenfeld der Kernel Funktion bezüglich des Partikels i .

3.4 Impulsgleichung

Die Impulsgleichung beschreibt die Erhaltung des Impulses und wird in der SPH-Methode wie folgt diskretisiert:

$$\frac{dv_i}{dt} = - \sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla_i W_{ij} + 2\nu \sum_j \frac{m_j}{\rho_j} \frac{(v_i - v_j) \times (x_i - x_j)}{(x_i - x_j) \times (x_i - x_j) + 0.01 \cdot h^2} \nabla_i W_{ij} + f_i$$

3.4.1 Druckbeschleunigung

Die Beschleunigung aufgrund des Drucks wird durch die folgende Gleichung beschrieben:

$$a_{druck} = - \sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla_i W_{ij} \quad (16)$$

Dabei sind p_i und p_j die Drücke der Partikel i und j , ρ_i und ρ_j die Dichten der Partikel i und j , und $\nabla_i W_{ij}$ der Gradient der Kernel Funktion bezüglich des Partikels i .

3.4.2 Viskositätsbeschleunigung

Die Beschleunigung aufgrund der Viskosität wird durch die folgende Gleichung beschrieben:

$$a_{viskosität} = 2\nu \sum_j \frac{m_j}{\rho_j} \frac{(v_i - v_j) \cdot (x_i - x_j)}{(x_i - x_j) \cdot (x_i - x_j) + 0.01 \cdot h^2} \nabla_i W_{ij} \quad (17)$$

Hier ist ν die kinematische Viskosität, v_i und v_j sind die Geschwindigkeiten der Partikel i und j , x_i und x_j sind die Positionen der Partikel i und j , und $\nabla_i W_{ij}$ ist der Gradient der Kernel Funktion bezüglich des Partikels i .

3.4.3 Externe Kräfte

Die externen Kräfte, die auf ein Partikel wirken, werden durch f_i dargestellt. Dies kann beispielsweise die Gravitation oder andere externe Einflüsse umfassen:

$$a_{extern} = f_i \quad (18)$$

3.5 Berechnung der Partikeleigenschaften

3.5.1 Dichteberechnung

Die Dichte jedes Partikels i wird durch die Summe der Massen der benachbarten Partikel j gewichtet durch die Kernel Funktion berechnet:

$$\rho_i = \sum_j m_j W_{ij} \quad (19)$$

3.5.2 Druckberechnung

Der Druck p_i wird häufig durch eine Zustandsgleichung berechnet, z.B.:

$$p_i = k \left(\frac{\rho_i}{\rho_0} - 1 \right) \quad (20)$$

wobei k die Steifigkeit Konstante ist und ρ_0 die Ruhe-Dichte.

4 Simulationsschritt

Der Simulationsschritt ist in der SPH Simulation von Zentraler Bedeutung. Dort werden alle nötigen Beschleunigungen ermittelt, die zum simulieren eines Fluides mithilfe von Partikeln, nötig sind. In diesem Projekt wurde die Methode in 3 Schleifen unterteilt wobei die erste als einzige alle Partikel berücksichtigt. In den Restlichen Schleifen werden nur Partikel welche das Fluid bilden beachtet und Partikel zum Simulieren der Grenze Ignoriert. Eigenschaften wie die Partikelgröße und die Ruhedichte der Flüssigkeit werden vor dem Starten der Simulation festgelegt. Aus diesen wird eine Masse für die Partikel ermittelt, welche im Laufe der Simulation konstant bleibt. Eigenschaften wie die Position, Geschwindigkeit, lokale Dichte und lokaler Druck werden im Partikel gespeichert. Die Nachbarn der einzelnen Partikel als auch alle berechneten Beschleunigungen werden in Hashtabellen gespeichert damit diese über alle schleifen zugänglich sind.

Algorithm 1 Simulationsschritt

```
1: Parallel for each particle in particles
2:   FindNeighbors()
3:   ComputeLocalDensity() (19)
4:   ComputeLocalPressure() (20)

5: Parallel for each particle in noBoundaryParticles
6:   GetViscosityAcceleration() (17)
7:   GetPressureAcceleration() (16)

8: For each particle in noBoundaryparticles
9:   acceleration = viscosityAcceleration + pressureAcceleration + gravitation
10:  particle.Velocity+ = timeSteps · acceleration
11:  particle.Position+ = timeSteps · particle.Velocity
```

In der ersten Schleife werden die lokalen Eigenschaften ermittelt. Dazu müssen die Nachbarpartikel in einem Radius $2 \cdot \text{Partikelgröße}$ um die Partikel ermittelt werden. Mithilfe der Nachbarn wird die lokale Dichte und der lokale Druck jedes Partikels bestimmt. Diese werden für die Berechnungen der nächsten Schleife benötigt. In dieser werden für alle Partikel die das Fluid bilden, die Viskosität- und Druckbeschleunigungen berechnet. Anhand dieser Beschleunigungen werden die Positionen der Partikel in der letzten schleife aktualisiert.

5 Nachbarschaftssuche

Die Nachbarschaftssuche ist das Fundament der Simulation. Diese erlaubt das Ermitteln der Nachbarpartikel in einem festen Radius um eine Position zum Berechnen lokaler Werte. Ein naiver Ansatz wäre, für alle Partikel den Abstand zur Position zu ermitteln und zu prüfen, ob sich diese im Radius befinden. Diese Methode weist jedoch eine quadratische Laufzeitkomplexität auf, da für jedes Partikel der Abstand zu allen anderen Partikeln berechnet werden muss. Dies führt zu einer Laufzeit von $O(n^2)$, wobei n die Anzahl der Partikel ist. Bei einer großen Partikelanzahl wird dieser Ansatz daher ineffizient und rechenintensiv.

Für das Projekt wurde dementsprechend das Räumliche Hashing verwendet. Dabei handelt es sich um eine Datenstruktur die den Raum in einzelne Blöcke unterteilt. Beim ermitteln von Nachbarpartikel müssen dadurch nicht mehr alle Partikel geprüft werden sondern nur diese, die sich in dem selbem Block oder Nachbarblock befinden. Dies führt zu einer effizienteren und weniger rechenintensiven Methode zum Bestimmen der Nachbarn.


5.1 Einfügen von Partikeln

Um ein Partikel in die Datenstruktur einzufügen wird als erstes ein Hashwert bezüglich der Position errechnet. Dazu wird der Positionsvektor durch die Blockgröße geteilt und das Ergebnis abgerundet.

Algorithm 2 Hash Funktion

Require: Vector $\mathbf{v} = (v_x, v_y)$, scalar CellSize

Ensure: Hashed vector $\mathbf{h} = (h_x, h_y)$

- 1: $\mathbf{h} \leftarrow \frac{\mathbf{v}}{\text{CellSize}}$
 - 2: $\mathbf{h} \leftarrow \lfloor \mathbf{h} \rfloor$  \mathbf{h}
-

Schließlich wird der Partikel in eine Liste innerhalb der Hashtabelle eingefügt. Das Löschen eines Partikels aus der Datenstruktur funktioniert analog.

5.2 Updaten der Partikel Positionen

Wurden die Positionen der Partikel im Simulationsschritt aktualisiert, müssen ebenfalls die Einträge in der Hashtabelle Aktualisiert werden. Dazu werden die Partikel vor dem Aktualisieren der Position aus der Datenstruktur entfernt und nach der Aktualisierung wieder eingefügt.


5.3 Ermitteln von Nachbarn

Das ermitteln von Nachbar Partikeln ist das Fundament des Simulationsschrittes. Zu erst werden die Blöcke, welche Teil des Radiuses sind und somit durchsucht werden müssen, ermittelt. Des Weiteren werden die Partikel in den Blöcke überprüft, ob sich diese in dem Radius um die Position befinden.

Algorithm 3 Partikel im Radius

Require: Position \mathbf{p} , Radius r

Ensure: Liste der Partikel im Radius: `partikelImRadius`

- 1: $startX \leftarrow \lfloor (\mathbf{p}_x - r) / \text{CellSize} \rfloor$
 - 2: $endX \leftarrow \lceil (\mathbf{p}_x + r) / \text{CellSize} \rceil$
 - 3: $startY \leftarrow \lfloor (\mathbf{p}_y - r) / \text{CellSize} \rfloor$
 - 4: $endY \leftarrow \lceil (\mathbf{p}_y + r) / \text{CellSize} \rceil$
 - 5: **for** ($x \leftarrow startX$ **to** $endX$) **do**
 - 6: **for** ($y \leftarrow startY$ **to** $endY$) **do**
 - 7: **if** ((x, y) existiert in Hashtabelle) **then**
 - 8: **for all** (Partikel in Hashtabelle[hash]) **do**
 - 9: **if** ($\text{Distance}(\text{Partikel.Position}, \mathbf{p}) \leq r$) **then**
 - 10: `partikelImRadius.add(q)`
-  `partikelImRadius`
-

Dies erlaubt es die Anzahl an Partikeln welche überprüft werden müssen zu beschränken und die Laufzeit zu verringern.

6 Implementierung

Die Implementierung ist zentraler Bestandteil des Projektes. Implementiert wurde in C# mit Hilfe des Frameworks Monogame. In diesem Kapitel wird die Implementierung von Tests, essenziell für die Korrektheit der Simulation, und die Implementierung Spezieller Hilfsmittel vorgestellt.

6.1 Tests

Das Testen der einzelnen Komponenten des Simulationsschrittes ist essenziell für die Funktionalität und Korrektheit der Simulation. Sie helfen dabei offensichtliche Probleme in der Simulation zu isolieren und zu korrigieren. Allerdings sind ebenfalls nicht offensichtliche Fehler in der Simulation nicht auszuschließen, weshalb die Tests ebenfalls zusätzliche Sicherheit garantieren. Diese werden beim Starten der Simulation gestartet. Dabei wird ein ideales Gitter der Größe $10 \cdot 10$ mit Partikeln erzeugt. „Ideale“, dass alle Partikel den gleichen Abstand h zueinander haben. Auf diesen Partikeln werden schließlich die Tests angewendet. Schlägt ein Test fehl, erscheint eine Nachricht in roter Schrift auf dem Bildschirm.

Das Fundament aller Berechnungen ist die Nachbarschaftssuche. Ausschließlich bei diesem Test wird ein kleineres ideales Gitter mit $5 \cdot 5$ Partikeln erzeugt. TODO LONG Um zu gewährleisten dass alle Partikel, die sich im Such-Radius um den mittleren Partikel befinden, gefunden werden, werden die von der Räumlichen Datenstruktur gefunden Partikeln in einer Liste gesammelt. Die Anzahl der Partikel in der Liste muss nach dem Suchen 13 ergeben.

Für die folgenden Tests werden für alle Partikel aus dem größerem Gitter die Nachbarn gesucht und die Komponenten getestet. Begonnen wird mit der Kernel Funktion. Eine falsche Gewichtung der Nachbarn in den Berechnungen führt zu falschen Kräften und Bewegungen der Teilchen. Um die Funktionalität der Kernel Funktion zu gewährleisten werden folgende vier Tests auf diese angewandt:

$$\sum_j W_{ij} = \frac{1}{h^2}$$

$$W(1,0) = \alpha((2-0)^3 - 4(1-0)^3) = \alpha(8-4) = 4\alpha$$

$$W(1,1) = \alpha((2-1)^3) = \alpha$$

$$W(h,2) = 0$$

Weiterhin wird die Funktionalität der Kernel Ableitung getestet. Diese ist ein wesentlicher Bestandteil der Gleichungen zum Berechnen der wirkenden Kräfte auf die Partikel. Auf dieser Komponente werden folgende zehn Tests angewandt:

$$\begin{aligned}
\sum_j \Delta W_{ij} &= (0, 0) \\
\Delta W(h, 0) &= (0, 0) \\
\Delta W(h, (0, 0) - (\pm h, 0)) &= \left(\frac{3\alpha}{h}, 0\right) \\
\Delta W(h, (0, 0) - (0, \pm h)) &= \left(0, \frac{3\alpha}{h}\right) \\
\Delta W(h, (0, 0) - (h, h)) &= \Delta W(h, (0, 0) - (-h, -h)) = \left(-\frac{9\sqrt{2}-12}{h}\alpha, -\frac{9\sqrt{2}-12}{h}\alpha\right) \\
\Delta W(h, (0, 0) - (h, -h)) &= \Delta W(h, (0, 0) - (-h, h)) = \left(-\frac{9\sqrt{2}-12}{h}\alpha, \frac{9\sqrt{2}-12}{h}\alpha\right)
\end{aligned}$$

Ebenfalls werden die Funktionen zum Berechnen lokaler Werte getestet. Da die Partikel sich beim Test in einem idealem Gitter befinden, ergibt die Berechnung der lokalen Dichte die Ruhedichte ρ_0 .

$$\rho_i = \sum_j m_j W_{i,j} = m \sum_j W_{i,j} = \frac{m}{h^2} = \rho_0$$

Daraus folgt ebenfalls dass die Funktion des lokalen Drucks 0 ergibt:

$$p_i = k\left(\frac{\rho_0}{\rho_0} - 1\right) = k(1 - 1) = 0$$

6.2 Tools

Die Simulation verfügt über einige implementierte Tools zur Steuerung der Simulation. Durch das Drücken der Taste öffnet sich eine Liste von Aktionen zum Steuern der Simulation. Einige der wichtigsten Aktionen werden in diesem Unterkapitel vorgestellt.

Die Simulation verfügt über mehrere verschiedene „Szenarios“. Ein „Szenarios“ bedeutet hier, eine Box oder eine andere geometrische Form, welche eine Barriere für die zu simulierende Flüssigkeit bildet. Diese können mit der Taste gewechselt werden. Durch das Wechseln der Szene oder Drücken der Taste werden alle Partikel der Szene gelöscht.

Ebenfalls können zu Zwecken der Simulation Partikel mit Hilfe der Maus in der jeweiligen Szene platziert werden. Dazu wird mit der Taste die Form einer Box, eines Kreises oder eines einzelnes Partikels ausgewählt. Im Falle der Box und des Kreises können die Höhe und Breite mit den Tasten , und , TODO verändert werden. Platziert werden die Partikel durch ein Linksklick.

Im Sinne der Analyse können durch das Drücken der Taste Daten gesammelt

werden und durch das Drücken von **F1** gespeichert werden. Zusätzlich wird durch drücken der Taste **F12** ein Screenshot aufgenommen. Gespeichert werden alle Dateien im Dokumentenordner des Benutzers.

6.3 Parallelisierung des Simulationsschrittes

Die ersten beiden Schleifen des Simulationsschrittes (siehe Kapitel 4) wurden mit Hilfe der Funktion `Parallel.ForEach()` implementiert. Diese Funktion ermöglicht es, die einzelnen Prozesse auf verschiedene Threads der CPU aufzuteilen. Dabei wartet die Funktion automatisch, bis alle Threads ihre Aufgaben abgeschlossen haben, bevor die Verarbeitung im sequentiellen Code fortgesetzt wird. Da beide Schleifen auf Hashtabellen zugreifen werden diese Zugriffe gesperrt um Wettlaufsituation zu verhindern.

7 Analysen

Zuletzt werden die gesammelten Daten von Experimenten visualisiert und vorgestellt. Die Experimente befassen sich mit Schwerpunkten der Simulation. Das erste Experiment beobachtet die Auswirkung der Einstellungsparameter auf die Stabilität um das Verhalten des Fluiden. Dazu wird eine Wassersäule simuliert und das Verhalten der Dichte aller Partikel zu analysieren. Da der Druck innerhalb eines Fluiden Abhängig der Gravitation, Fluid Höhe und Fluid Dichte ist, reicht es hier eine schmale Säule für das Experiment zu nutzen. Zunächst wird der Einfluss der beschleunigten Nachbarschaftssuche auf die Performance der Simulation betrachtet. Hierfür wird die Zeit um einen Simulationsschritt zu durchlaufen (Simulationsschrittzeit) beobachtet. Dabei wird die Anzahl an Partikeln kontinuierlich erhöht und in Relation zur Simulationsschrittzeit gesetzt. Dieses Vorgehen wird auf beide Methoden der Nachbarschaftssuche angewandt. Analog dazu wird ebenfalls betrachtet, wie sich die Performance verhält für eine sequentielle- und parallele Prozessverarbeitung.

7.1 Säulenexperiment

In dem durchgeführten Säulenexperiment wurden die Auswirkungen der Steifigkeit k , der Viskosität ν und des Zeitschritts Δt auf die lokalen Dichten aller Partikel untersucht. Hierbei wurde in jedem Simulationsschritt der Mittelwert der lokalen Dichten aller Partikel ermittelt und im Verhältnis zu den Simulationsschritten bzw. der Zeit dargestellt. Die Dichten werden in den Grafiken als relativer Dichtefehler angezeigt, der die prozentuale Abweichung von der Ruhedichte angibt. Der relative Dichtefehler dient als Maß für die numerische Kompressibilität des Fluids; ein optimaler Wert wäre ein relativer Dichtefehler von Null. In der Praxis ist dies jedoch

schwer zu erreichen, insbesondere bei numerischen Simulationen. Ziel ist es daher, ein Verständnis für die Beziehung zwischen den Parametern und dem Verhalten des Fluids zu entwickeln.

Der wichtigste Parameter in diesem Zusammenhang ist der Steifigkeitsparameter k . Dieser skaliert die Berechnung der lokalen Dichte (19) und hat einen direkten Einfluss auf die Druckbeschleunigungen (16) sowie die Kompressibilität des Fluids. In Abbildung 3 wurde das Säulenexperiment mit verschiedenen Werten für k simuliert. Um die Visualisierung anschaulicher zu gestalten, wurde für dieses Experiment eine hohe Viskosität gewählt, wodurch Schwankungen des Dichtefehlers minimiert werden, was zu einem konstanten Wert führt. Auffällig ist, dass die lokale Dichte konstant bleibt und sich der Dichtefehler mit steigendem k verringert. Die lokale Dichte konvergiert dabei in Richtung der Ruhedichte. Daraus lässt sich schließen, dass eine höhere Steifigkeit k die numerische Kompressibilität des Fluids reduziert und somit die Simulation stabiler und präziser wird.

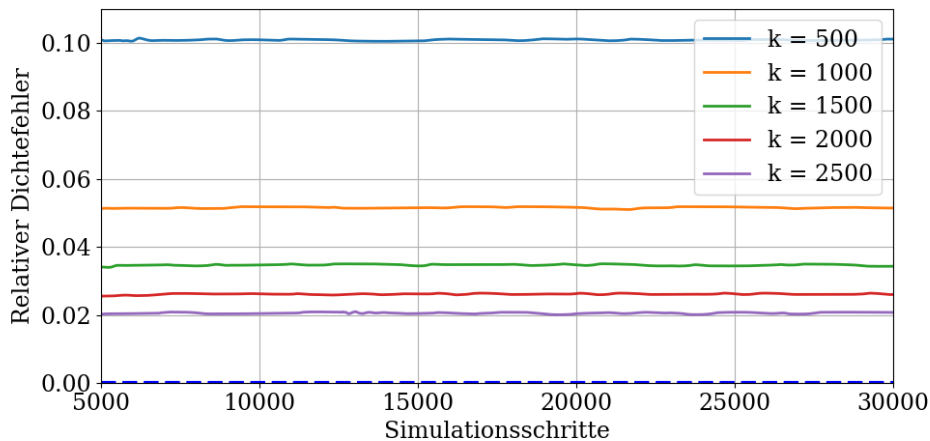


Abbildung 3: Säulenexperiment mit Verschiedener Steifigkeit k

Im zweiten Säulenexperiment wurde die Viskosität ν variiert. Diese beeinflusst, wie stark ein Partikel von seinen Nachbarn gedämpft wird. Abbildung 4 zeigt, wie die Viskosität die Schwankungen der lokalen Dichte dämpft. Dies wird durch eine Verringerung der Amplitude der Schwankungen deutlich. Je höher die Viskosität, desto stärker dämpfen sich die Partikel gegenseitig, was zu einer schnelleren Anpassung des Fluids an Gleichgewichtszuständen führt. Es ist jedoch wichtig zu beachten, dass eine zu hohe Viskosität das physikalische Verhalten des Fluids verfälschen könnte, indem das Fluid zu zähflüssig erscheint.

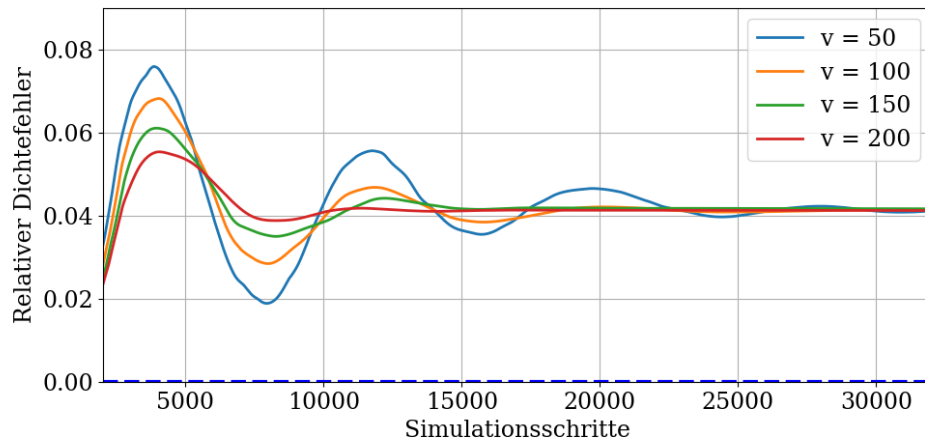


Abbildung 4: Säulenexperiment mit Verschiedener Viskosität v

Der letzte untersuchte Parameter ist der Zeitschritt Δt . Dieser skaliert die auf die Partikel wirkende Beschleunigung und damit auch die Bewegung der Partikel. In Abbildung 7.1 ist zu erkennen, dass kleinere Zeitschritte Δt zu einer Vergrößerung der Amplitude und einer Verkleinerung der Frequenz der Dichteschwankungen führen. Die verringerte Frequenz ist darauf zurückzuführen, dass die Bewegungsdistanz der Partikel bei kleineren Zeitschritten innerhalb eines Simulationsschritts reduziert wird. Die größere Amplitude rührt daher, dass bei kleineren Δt auch die Beschleunigungen geringer ausfallen, was dazu führt, dass die Partikel mehrere Simulationsschritte benötigen, bis die Beschleunigungen ihre volle Wirkung entfalten. Es sollte jedoch betont werden, dass ein zu großer Zeitschritt die Simulation instabil machen kann, da wichtige physikalische Prozesse möglicherweise nicht richtig aufgelöst werden. Andererseits können zu kleine Zeitschritte die Rechenzeit erheblich verlängern, ohne einen proportionalen Nutzen in der Genauigkeit zu bringen.

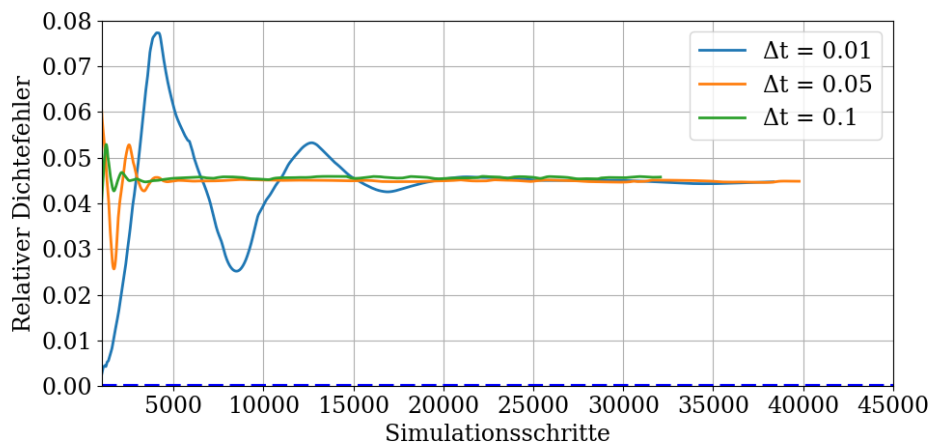


Abbildung 5: Säulenexperiment mit Verschiedenen Zeitschritten Δt

7.2 Nachbarschaftssuche

Wie in Kapitel 5 erläutert, ist die Nachbarschaftssuche das Fundament. Alle Berechnungen für die Simulation von Fluiden benötigen die Nachbarpartikel. Daher werden diese wie in Kapitel 6 beschrieben, für alle Partikel in jedem Simulationsschritt ermittelt. Dabei ist eine effiziente Nachbarschaftssuche von großem Vorteil. Eine schnellere Suche ermöglicht eine weniger rechenintensive Simulation bei einer höheren Anzahl an Partikeln. In dieser Analyse wird geprüft, ob die Benutzung der Räumlichen Datenstruktur die Performance der Simulation, im Vergleich zur Quadratischen suche, positiv beeinflusst wird. Dazu werden während der Simulation kontinuierlich neue Partikel erzeugt. Dabei wird die Simulationsschrittzeit gemessen und in Verhältnis zur Anzahl an Partikeln dargestellt.

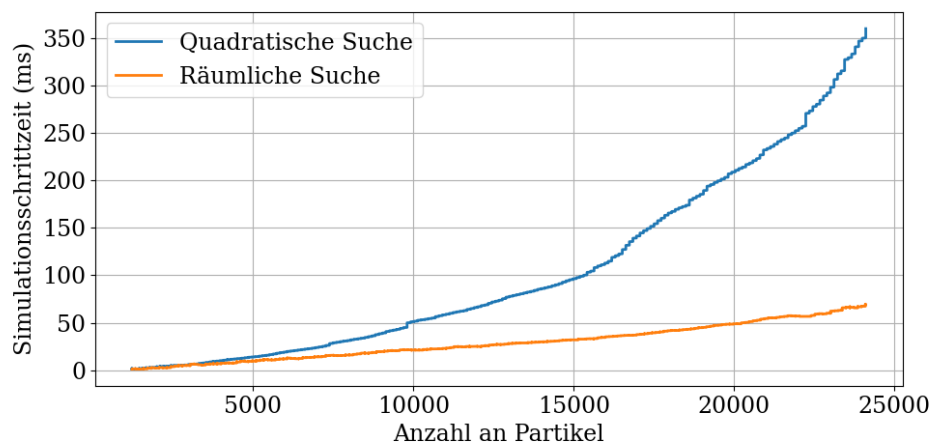


Abbildung 6: Nachbarschaftssuche mit Paralleler Prozessverarbeitung

Wie erwartet steigt sich die Simulationsschrittzeit für die Quadratische Suche, bei erhöhen der Partikel Anzahl, Exponentiell, wobei sich die Simulationsschrittzeit bei der Räumlichen Datenstruktur linear erhöht. Die Räumliche Datenstruktur erlaubt es eine eine erhebliche höhere Anzahl an Partikeln simulieren zu können bei linearem Wachstum des Rechenaufwandes.

7.3 Prozessverarbeitung

Die Parallelisierung des Simulationsschrittes (siehe Kapitel 4) trägt ebenfalls zu einer Verringerung der Simulationsschrittzeit bei. Bei dieser Analyse wird wie zuvor, die Anzahl an Partikeln kontinuierlich erhöht und die Simulationsschrittzeit gemessen. Dies wird dann mithilfe eines sequentiellen Simulationsschrittes und mit einem parallelen Simulationsschritt simuliert.

Die Ergebnisse (siehe Abbildung 7) zeigen eine klare Verkürzung der Simulationsschrittzeit bei einer parallelen Prozessverarbeitung. Bei diesen Daten wird die

Simulationsschrittzeit bei paralleler Verarbeitung halbiert. Dabei gilt jedoch zu beachten, dass dieser Trend je nach CPU abweichen kann.

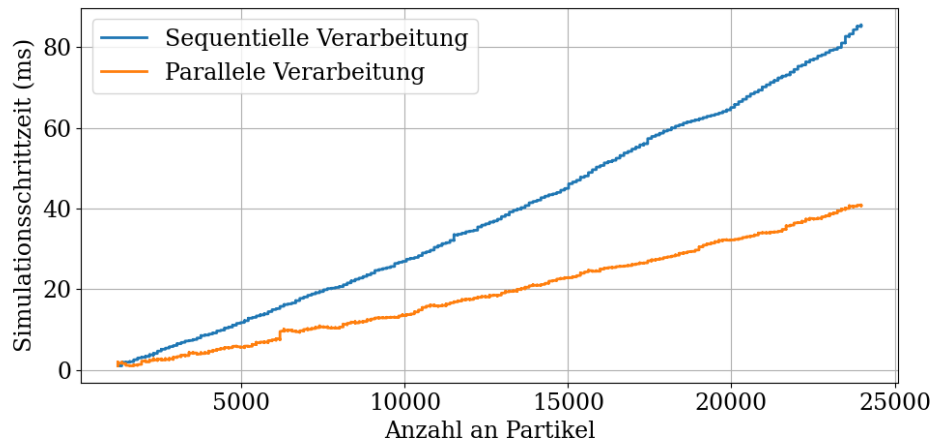


Abbildung 7: Prozessverarbeitung mit beschleunigter Nachbarschaftssuche

8 Schlussfolgerung

Das Ziel dieses Projekts war, eine grundlegende SPH-Simulation (Smoothed Particle Hydrodynamics) zu implementieren, ein vertieftes Verständnis für die SPH-Methode zu entwickeln und die Auswirkungen der Einstellungsparameter auf das Fluidverhalten zu analysieren. SPH erwies sich als eine leistungsfähige und zugleich einfache Methode zur Simulation von Fluiden. Im Rahmen des Projekts wurde ein einfacher SPH-Simulator erfolgreich implementiert. Durch die Integration einer beschleunigten Nachbarschaftssuche und paralleler Verarbeitung konnte eine stabilere und flüssigere Simulation mit einer größeren Anzahl von Partikeln erreicht werden.

Das Projekt trug maßgeblich dazu bei, ein tieferes Verständnis für die zugrunde liegenden Prozesse in SPH zu entwickeln und zu erkennen, wie komplexe Strömungen realistisch simuliert werden können. Die Analyse und Interpretation der experimentellen Ergebnisse ermöglichte es, die Einflüsse der Parameter auf das Fluidverhalten besser zu verstehen und deren Wechselwirkungen zu identifizieren. Dadurch konnte eine stabile Simulation durch gezielte Anpassung der Parameter gewährleistet werden.

Zur Veranschaulichung der Effektivität der optimierten Parameter wurde eine Simulation durchgeführt. Abbildung 8 zeigt, dass durch wählten optimaler Parameter eine signifikante Verbesserung der Simulation erreicht wurde. Die Flüssigkeit verhält sich stabil und realistisch, ohne unerwünschte Artefakte oder Instabilitäten. Die Anpassungen führten zu einer präziseren und natürlicheren Darstellung der Fluidodynamik,.

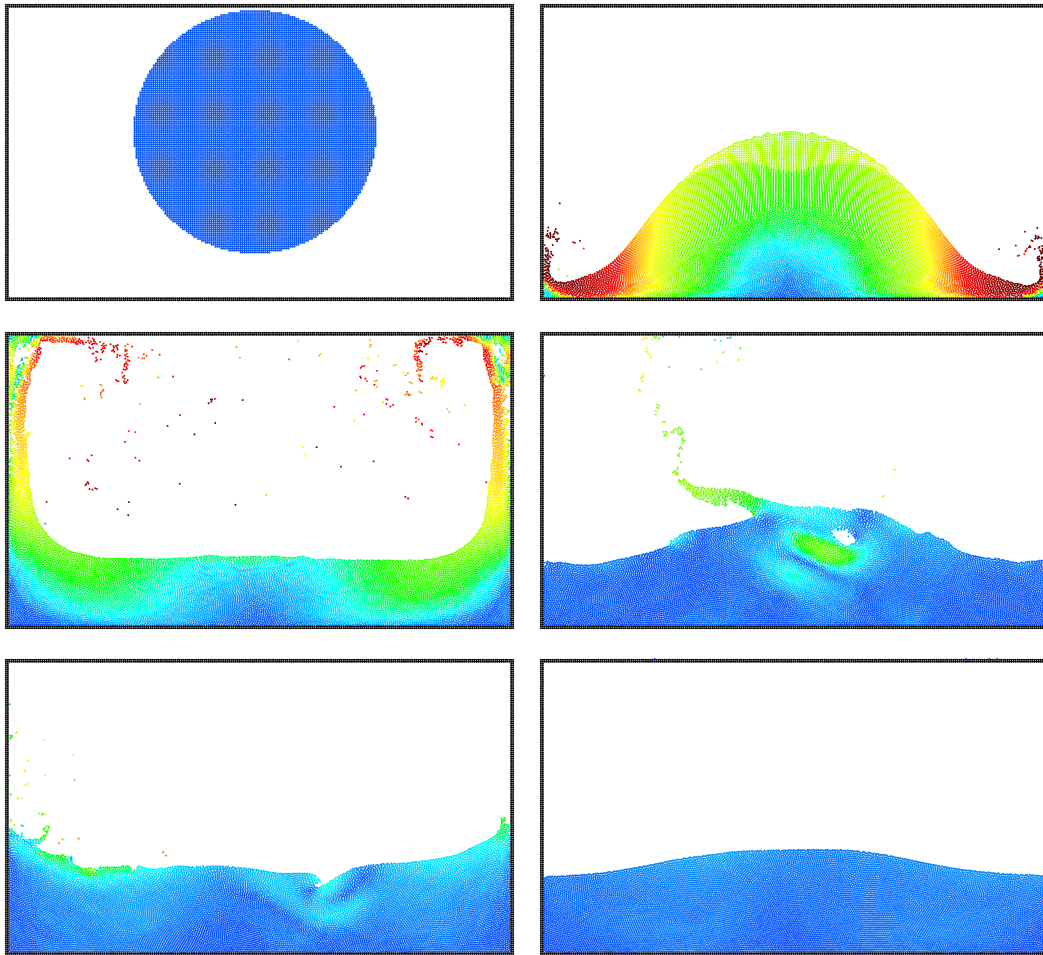


Abbildung 8:

Dennoch zeigte sich ein Nachteil dieser Implementierung: Die Parameter müssen szenenabhängig angepasst werden, was einen erheblichen zeitlichen Aufwand erfordert. Zukünftige Arbeiten könnten darauf abzielen, einen Drucklöser zu implementieren, um die Fluidodynamik weiter zu verbessern. Die Einführung eines Drucklösers könnte helfen, den Druck in den Flüssigkeitsbereichen besser zu regulieren und somit die Stabilität der Simulation zu erhöhen.

9 Bibliographie