Institut für Visualisierung und Interaktive Systeme
Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit Nr. 9

Oberflächenrekonstruktion für partikelbasierte Fluidsimulationen

Alexander Tivonenko

Studiengang:	Informatik
Prüfer:	Prof. Dr. Daniel Weiskopf
Betreuer:	Dipl.-Inf. Markus Huber
begonnen am:	21. Mai 2012
beendet am:	20. November 2012
CR-Klassifikation:	I.3.5, I.3.7

Kurzfassung

Man kommt in der Computergrafik immer wieder mit komplexen Simulationen in Berührung. Ohne eine realistische Simulation von Wasser oder anderen Fluiden wären viele 3D-Cartoons oder Spezialeffekte in großen Filmproduktionen kaum möglich. Viele der bis heute verwendeten Methoden basieren auf dem Eulerschen gitterbasierenden Ansatz oder dem partikelbasierten Ansatz von Lagrange. Die Smoothed Particle Hydrodynamics (SPH)-Methode ist ein Lagrange-Ansatz, der aus der Astronomie stammt und heutzutage oft Anwendung in Fluidsimulationen findet. Auf der SPH-Simulation basieren die beiden, in dieser Arbeit vorgestellten Methoden zur Rekonstruktion der Oberfläche von Fluiden.
Die erste Methode, von Müller et al., berechnet aus den Attributen der Partikel aus der SPH-Simulation ein Skalarfeld und verwendet eine Isofläche im Skalarfeld als Oberfläche. Die zweite Methode, von Yu und Turk, ist neuer und optimiert den Ansatz von Müller, indem anisotrope Kerne die umliegenden Partikel in die Berechnung des Skalarfeldes miteinbeziehen. Das soll dazu dienen, die Oberfläche glatter zu gestalten. Beide Verfahren werden in dieser Arbeit verglichen und auf ihre optischen Eigenschaften untersucht. Desweiteren werden Leistungsoptimierungen angewandt und Benchmarkergebnisse analysiert.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

1.1 Motivation

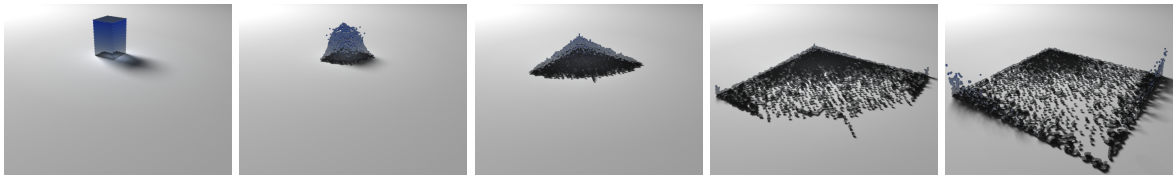

Abbildung 1.1: Mit der Methode von Müller et al. berechnete Dam Break-Simulation.

In vielen Bereichen, wie Industrie oder Medienbranche werden heute Fluidsimulationen und ihre Darstellung benötigt. Es gibt viele Arbeiten zu diesem Themenfeld und dementsprechend gibt es viele verschiedene Ausführungen. Im Gegensatz zu gitterbasierenden Eulerschen Ansätzen, baut Smoothed Particle Hydrodynamics (SPH) auf dem partikelbasierten Ansatz von Lagrange auf. SPH stammt ursprünglich aus der Astronomie [Luc77], wo asymmetrische Gebilde, wie z.B. Weltraumnebel damit simuliert wurden, wurde jedoch von Gingold und Monaghan [GM77][Mon92] in der Fluidsimulation verwendet. Die beiden in dieser Arbeit vorgestellten Methoden [MCG03][YT10] berechnen aus den Attributen der Partikel aus der SPH-Simulation ein Skalarfeld, in dem eine Isofläche für die Oberfläche verwendet wird.
Das neuere der beiden Verfahren optimiert die optischen Ergebnisse mit der gewichtetten Hauptkomponentenanalyse und der Singulärwertzerlegung und liefert letztlich eine glattere Oberfläche. Wieviel Leistung diese Optimierung kostet, inwiefern die neuere Methode optisch bessere Ergebnisse liefert und wie genau sie funktionieren, wird in den folgenden Kapiteln geklärt.

1.2 Ziele dieser Arbeit

In dieser Arbeit werden zwei Verfahren zur Oberflächenrekonstruktion für partikelbasierte Fluidsimulationen implementiert. Beide Verfahren werden von Grund auf erklärt und verglichen. Die optischen Ergebnisse werden gegenübergestellt und der Rechenaufwand beider Verfahren wird mit Messungen analysiert.

Gliederung

Zu Beginn dieser Bachelorarbeit werden verwandte Arbeiten aufgezählt und somit ein Überblick über existierende SPH-basierte Verfahren zur Oberflächenrekonstruktion geschaffen. Im nächsten Kapitel werden nötige Grundlagen erklärt, damit im darauffolgenden Kapitel das nötige grundlegende Wissen gegeben ist, um die beiden vorgestellten Rekonstruktionsmethoden nachvollziehen zu können. Danach wird mehr über die Implementierung des begleitenden Programmes erzählt, bevor im nächsten Kapitel die Ergebnisse der Arbeit verglichen und ausgewertet werden. Zuletzt wird die Arbeit zusammengefasst und ein Ausblick über weitere, mögliche Forschungsbereiche gegeben.

Kapitel 2 – Verwandte Arbeiten: Verwandte oder weiterführende Arbeiten.

Kapitel 3 – Grundlagen: Grundlagen, die für das Verständnis benötigt werden.

Kapitel 4 – Rekonstruktionsmethoden: Vorstellung und Erläuterung der beiden Methoden.

Kapitel 5 – Implementierung: Das implementierte Programm mitsamt Optimierungen.

Kapitel 6 – Vergleiche: Vergleich und Auswertung der Ergebnisse dieser Arbeit.

Kapitel 7 – Zusammenfassung und Ausblick Fazit dieser Arbeit und kurzer Ausblick.

2 Verwandte Arbeiten

Um den Einstieg in die Thematik zu erleichtern, werden in dieser Arbeit zu Beginn weitere verwandte Arbeiten vorgestellt, die mit der SPH-basierten Fluidsimulation in der Computergrafik zu tun haben. Nachdem ein gewisser Überblick über die Entwicklungen der letzten Jahrzehnte geschaffen wurde werden einige weiterführende Arbeiten sowie anders ansetzende Methoden erwähnt.

2.1 Partikelbasierte Fluidsimulation mit der SPH Methode

1977 wurde SPH in der Astrophysik vorgestellt [Luc77] und diente zum berechnen asymmetrischer Phänomene im Weltall. Gingold und Monaghan verwendeten SPH im selben Jahr für partikelbasierte Fluidsimulation [GM77] und legten damit den Grundstein für unter anderem diese Arbeit. Es war eines der ersten nicht gitterbasierten Verfahren.

Monaghan erläuterte daraufhin die einzelnen Aspekte des SPH-Verfahrens [Mon92] und zeigte damit, dass das Verfahren nicht immer die genauesten Ergebnisse liefert, deren Fehler aber im Toleranzbereich liegt und das Verfahren schneller ist als andere vergleichbare Verfahren.

Später stellten Desbrun und Gascuel die geglätteten Partikel vor [DG96], die die bisherige SPH-Methode erweiterten. Ein Partikel steht in dieser Methode stellvertretend für den ihn umgebenden Abschnitt eines deformierbaren Objektes, wie z.B. eines Fluids. Die in dieser Arbeit vorgestellte Methode dient zur Simulation deformierbarer Objekte.

Müller et al. [MCG03] verwendeten ein SPH-basierendes Verfahren für interaktive Anwendungen und arbeiteten vor allem die Viskosität und Oberflächenspannung von Fluiden heraus. Zudem wandten sie äußere Kräfte, wie Schwerkraft auf die Partikel direkt an. Somit konnten realistische Einflüsse auf das Verhalten simuliert werden. Die erste, in dieser Bachelorarbeit verwendete Oberflächenrekonstruktion stammt ebenfalls aus der Arbeit von Müller et al.

Clavet et al. stellten 2005 ein SPH-basierendes Verfahren für die Berechnung viskoelastischer, also zähflüssiger, Fluide vor [CBP05]. Das wird mit einer doppelten Dichteberechnung ermöglicht.

Becker und Teschner beschäftigten sich mit der Kompressibilität von Fluiden [BT07]. Ihre Weakly Compressible SPH verhindert unerwünschtes Verhalten, wie Verwirbelungen durch leichte Kompressibilität.

Solenthaler und Pajarola entwickelten ein weiteres Verfahren [SP09], das die Simulation von inkompressiblen Fluiden mit einem reinen, einfacheren Lagrange-basiertem Ansatz ermöglicht. Ihr Ansatz verspricht geringeren Rechenaufwand pro Berechnungsschritt, als Verfahren, die Inkompressibilität durch Lösen der Poissongleichung erreichen. Das erlaubt größere Zeitintervalle für die Berechungsschritte, ohne längere Rechenzeiten in Kauf zu nehmen.

Da alle diese Algorithmen trotz Optimierungen relativ lange brauchen, um eine große Anzahl an Partikeln zu simulieren, gab es auch eine Reihe von Arbeiten, die die SPH-Simulation auf Grafikkarten ausgelagert haben. Da Grafikkarten hohes Paralellisierungspotenzial haben, lassen sich vor allem Simulation mit sehr vielen Partikeln sehr gut auf ihnen durchführen, solange der Grafikspeicher für die Partikel ausreicht. 2005 implementierten Kolb et al. die später vorgestellte SPH-Methode von Müller et al. [MCG03] auf einer Grafikkarte [KC05]. Diese Arbeit diente als Proof-of-Concept, also ein Beweis des Machbaren.

Harada et al. erreichten 2007 auf einer einzelnen GPU eine bis zu 28-fache Beschleunigung der SPH gegenüber der CPU-Berechnung auf einer Grafikkarte [HKK07].

2010 veröffentlichten Hérault et al. eine Arbeit, bei der sie SPH auf einer GPU über die CUDA-Schnittstelle berechnen ließen [HBD10]. Das benutzte Verfahren skaliert je nach Rechenschritt verschieden aber dennoch deutlich gut. Die Anzahl der einzelnen Prozessoren der Grafikeinheit beschleunigt die Rechenzeit proportional.

Rustico et al. beschäftigten sich mit der Arbeit mehrerer GPU-Einheiten im Verbund, unter anderem am Beispiel der SPH-Berechnung [RBG+12]. Sie erreichten mit vier Grafikkarten weitere, erhebliche Beschleunigung gegenüber einer.

2.2 Verfahren mit impliziter Oberfläche

Blinn stellte 1982 mit Blobby Surface ein Verfahren vor [Bli82], das den in dieser Arbeit gezeigten Verfahren ähnelt, jedoch nicht auf der SPH-Methode basiert. Er definierte impliziten Skalarfelder durch Partikel und ihre Radien. Diese konnten zu Isoflächen berechnet werden. Lagen so mehrere Objekte nebeneinander, verschmolzen sie. Die Blobby Surfaces funktionierten sehr gut für Darstellungen des Atomteilchenmodells, weil einzelne Punkte anfangen zu verschmelzen, wenn sie sich einander nähern, ganz so, wie man es sich bei Elektronenwolken von Atomen vorstellt. Jedoch hatte das Verfahren die Nachteile, immer nur runde Formen zu erzeugen und zu Beulen anzuwachsen, sobald sich mehrere Objekte überschnitten.

Zhu und Bridson knüpfen mit ihrer Arbeit [ZB05], bei der Sand wie ein Fluid simuliert wird, am Verfahren von Blinn an und optimieren es, indem sie gewichtete Mittelwerte aus Positionen und Radien anwendeten bevor sie mithilfe eines Gitters die Isofläche berechneten. Dadurch wirkt das Ergebnis nicht mehr so beulenhaft sondern schmiegt sich an die bestehenden Partikel an und wirkt realistischer.

2007 stellen Adams et al. eine weitere Verbesserung vor [APKG07]. Dieses Verfahren reduziert, mit Hinblick auf die Rechenzeit, die Partikeldichte in Bereichen in denen der Detailgrad nicht von Nöten ist, wie z.B. tief innerhalb des Fluids. Dazu werden Partikel-zu-Oberfläche-Distanzen berechnet und in jedem Frame neuberechnet. Die finale Rekonstruktion funktioniert nach der Methode von Zhu und Bridson, generiert aber glatte Oberflächen für feste und adaptive Partikelradien.

3 Grundlagen

Um die später vorgestellten Methoden besser verstehen zu können, werden in diesem Kapitel einige verwendete Grundlagen erläutert.
Zuerst wird die Funktion der SPH-Methode näher erklärt. SPH bietet die Basis für diese Arbeit, da mit SPH das Fluid simuliert wird, aus dessen Partikelattributen die in dieser Arbeit vorgestellten Methoden die Oberfläche rekonstruieren.
Danach folgt die Hauptkomponentenanalyse und ihre modifizierte Variante, die gewichtete Hauptkomponentenanalyse. Dieses Verfahren ermöglicht es, in den Berechnungen Partikel, die weiter entfernt sind und in der Berechnung des Skalarfeldes weniger Einfluss haben sollten als sich näher befindende Partikel, geringer zu gewichten.
Die letzte, näher erklärte Grundlage ist die Singulärwertzerlegung, die für die zweite vorgestellte Methode nötig ist. Sie ermöglicht den anisotropen Glättungskern, ist also das Werkzeug, mithilfe dessen die Partikel sich anhand ihrer Nachbarpartikel ausrichten und so eine glattere Oberfläche erzeugen.

3.1 Smoothed Particle Hydrodynamics

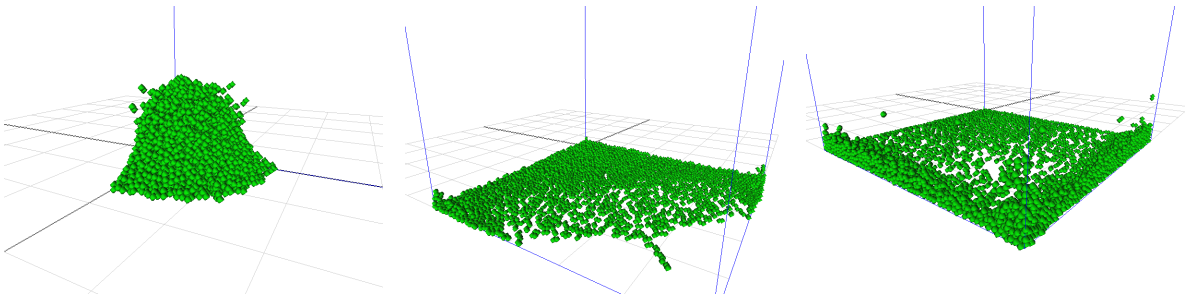

Abbildung 3.1: Alle Bilder aus dem Simulationsprogramm. Sie veranschaulichen die SPH-Simulation.

Smoothed Particle Hydrodynamics (dt. *geglättete Teilchen-Hydrodynamik*) ist, wie eingangs erwähnt, ein empirisches Verfahren, das für die Berechnung nicht achsensymmetrischer Phänomene der Astrophysik entwickelt wurde [Luc77] [GM77].

SPH ist ein auf einer Lagrange-Interpolation aufbauendes, partikelbasierendes Verfahren, also ein Verfahren, das im Gegensatz zu gitterbasierten Verfahren, bewegliche Teilchen besitzt. Positionen und wirkende Kräfte dieser Teilchen werden durch Differentialgleichungen

bestimmt. Diese basieren auf Gleichungen, die aus physikalischen Gesetzen resultieren, wie etwa der Berechnung für Impuls oder Energie. Somit finden sich Kräfte zwischen einzelnen Partikeln in der Gesamtleichung wieder und dieser Wirkungsgrad kann intuitiv in der Formel angepasst werden. Wegen dieser intuitiven Herangehensweise ist SPH in vielen Bereichen der Physik vertreten. Obwohl es präzisere Verfahren gibt, rechtfertigen die Unterschiede der Ergebnisse den Mehraufwand anderer Verfahren oft nicht.

Mit der Dirac-Funktion $\delta(\mathbf{x})$, die für $lim_{a\to 0}$ immer höher wird, deren Integral aber immer 1 ist, beschreibt eine Integralinterpolation ein kontinuiertliches Skalarfeld:

$$f(\mathbf{x}') = \int f(\mathbf{x}')\delta(\mathbf{x}-\mathbf{x}')d\mathbf{x} \tag{3.1}$$

Die Dirac-Funktion wird durch einen Glättungskern W ersetzt:

$$f(\mathbf{x}) = \int f(\mathbf{x}')W(\mathbf{x}-\mathbf{x}',h)\mathbf{d}\mathbf{x}'. \tag{3.2}$$

Der Glättungskern verfügt über folgende Eigenschaften:

$$\int W(\mathbf{x}-\mathbf{x}',h)\mathbf{d}\mathbf{x}' = 1 \tag{3.3}$$

und

$$\lim_{h\to 0} W(\mathbf{x}-\mathbf{x}',h) = \delta(\mathbf{x}-\mathbf{x}'), \tag{3.4}$$

wobei sich die Kerninterpolation auf einzelne Integrale bezieht [Nat61].

Hat man unendlich viele Partikel von unendlich kleiner Größe, also $\lim_{h\to 0}$, so gleicht die Summe der Kerne dem Integral. In der numerischen Integralinterpolation hat man nur endlich viele Partikel, nähert aber das Ergebnis mithilfe einer Summe an

$$f(\mathbf{x}) = \sum_j m_j \frac{f_j}{\rho_j} W(\mathbf{x}-\mathbf{x}_j,h), \tag{3.5}$$

wobei der Summenindex j sich auf Werte des Partikels j bezieht und die Summe somit alle Partikel durchläuft. Partikel j hat also Masse m_j, Position $\mathbf{x}_j$, Dichte ρ_j und Geschwindigkeit v_j. Der Wert einer Größe A an der Stelle $\mathbf{x}_j$ wird als A_j ausgedrückt.

In der Ursprünglichen Berechnung [GM77] wurde ein Gaußkern verwendet. Eindimensional:

$$W(x,h) = \frac{1}{h\sqrt{\pi}} e^{-(x^2/h^2)}, \tag{3.6}$$

Dieser Gaußkern imitiert eine Deltafunktion mit $\lim_{h\to 0}$. In den Rechnungen dieser Arbeit werden andere, an die Aufgaben angepasste Kerne verwendet.

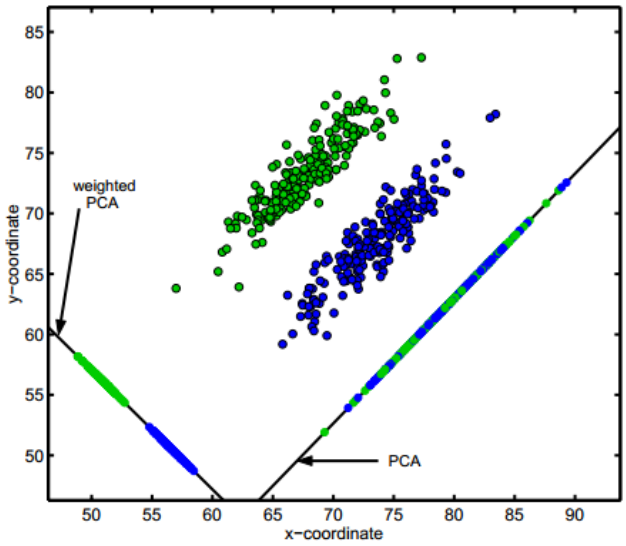

Abbildung 3.2: Gewichtete Hauptkomponentenanalyse. Man erkennt deutlich, dass die PCA zwei Cluster nicht als solche erkennt, wärhend die WPCA diese Cluster durch die distanzabhängige Gewichtung trennt [KC03].

3.2 Gewichtete Hauptkomponentenanalyse (WPCA)

Beim zweiten, später vorgestellten Verfahren will man erreichen, dass die Partikel sich umliegenden Partikeln angleichen, damit das Ergebnis der Oberflächenrekonstruktion glatter wirkt. Das soll ein anisotroper Glättungskern ermöglichen. Um ihn zu berechnen, werden Abstandsvektoren zwischen einem Partikel i und allen anderen Partikeln $j, j = 1, ..., n$ gewichtet in einer Matrix $\mathbf{A}$ verrechnet. Für diesen Schritt wird die gewichtete Hauptkomponentenanalyse (*engl:* weighted principal component analysis, WPCA) benutzt. Dabei handelt es sich um ein Verfahren, das große Datensätze veranschaulichend zusammenfassen kann. Die gewichtete Version der Hauptkomponentenanalyse verhindert, dass Ausreißer das Ergebnis verfälschen. Dabei sind in dieser Arbeit die Datensätze die Positionen aller Partikel und ihre Abstandsvektoren untereinander.

Sei $\mathbf{x}$ mit den Elementen $\mathbf{x}_j, j = 1, ..., n$ ein stochastischer Vektor mit der Wahrscheinlichkeitsverteilung $P(\mathbf{x})$ und sei $\{\mathbf{x}^\alpha | \alpha = 1, ..., m\}$ ein Ausschnitt aus $P(\mathbf{x})$.
Sei $\mathbf{X}$ nun eine Datenmatrix mit den Elementen $\mathbf{X}_{j,\alpha} = x_j^\alpha$. Die (ungewichtete) Hauptkomponentenanalyse (PCA) basiert auf den ersten beiden empirischen Momenten der Datenmatrix: Der Vektor des Mittels

$$\langle \mathbf{x} \rangle \equiv \frac{1}{m} \sum_{\alpha=1}^{m} \mathbf{x}^\alpha \tag{3.7}$$

und der empirischen Kovarianzmatrix

$$\mathbf{C} \equiv \frac{1}{m} \sum_{\alpha=1}^{m} (\mathbf{x}^{\alpha} - \langle \mathbf{x} \rangle)(\mathbf{x}^{\alpha} - \langle \mathbf{x} \rangle)^{T}, \tag{3.8}$$

die sich auch durch entfernen des Mittels der Daten in Matrizenschreibweise ausdrücken lässt

$$\mathbf{C} \equiv \frac{1}{m} \mathbf{X}\mathbf{X}^{T}, \tag{3.9}$$

Bei der Hauptkomponentanalyse gilt es die Eigenvektoren mit den größten Eigenvektoren der Kovarianzmatrix zu finden und die Daten in diese Richtung zu projezieren.

Wie Koren und Carmel zeigten [KC03] berechnet PCA die p-dimensionale Projektion, die die Summe aller quadrierten paarweisen Distanzen der Elemente maximiert:

$$\sum_{i<j} (\mathrm{dist}_{ij}^{p})^2, \tag{3.10}$$

Diese PCA-Variante führt dazu, dass wir ungewichtete quadrierte Distanzen aufsummieren. In manchen Anwendungen ist es jedoch erwünscht, dass die Distanzen gewichtet werden. In der partikelbasierten Simulation heißt das, dass einzelne, abgelegene Partikel weniger auf die gesamte Menge der Partikel wirken als Partikel, die sich unmittelbar nebeneinander, also in großen Ansammlungen, befinden (zur Veranschaulichung, siehe Abb. 3.2). Darum schlagen Koren und Carmel die gewichtete Hauptkomponentenanalyse vor, bei der die maximierte Summe gewichtete Distanzen beinhaltet:

$$\sum_{i<j} d_{ij} (\mathrm{dist}_{ij}^{p})^2, \tag{3.11}$$

wobei d_{ij} kleiner wird, je größer die Distanz der Elemente ist, also:

$$d_{ij} = \frac{1}{\mathrm{dist}_{ij}}. \tag{3.12}$$

Diese Gewichtung führt dazu, dass Ausreißer oder Messfehler weniger wirken und das Ergebnis nicht wesentlich verändern oder gar verfälschen.

Nach Anwendung der WPCA erhält man die Kovarianzmatrix $\mathbf{C}$. Sie beinhaltet für jeden Partikel i Informationen, in welchen Richtungen die größten Vorkommen naheliegender Partkiel sind und bildet die Basis des anisotropen Glättungskerns.

3.3 Singulärwertzerlegung (SVD)

Auf die aus der WPCA resultierende Kovarianzmatrix wird im nächsten Schritt die Singulärwertzerlegung angewendet. Bei der SVD wird eine Matrix $\mathbf{C}$ in ein Produkt aus drei Matrizen zerlegt. Eine dieser Matrizen ist eine Diagonalmatrix, auf deren Diagonalen sich

die Eigenwerte von $\mathbf{C}$ befinden. Diese Eigenwerte lassen sich anpassen, um das Ergebnis zu beeinflussen, bevor man das Matrizenprodukt berechnet und so eine nach Bedürfnissen modifizierte Version, also $\tilde{\mathbf{C}}$ erhält.

Um die Singulärwertzerlegung (*engl:* singular value decomposition) [MHW04] besser zu verstehen, sollte man einige Definitionen kennen:

Singulärwerte einer Matrix $\mathbf{C}$ sind die Quadratwurzeln der Eigenwerte von $\mathbf{C}^T\mathbf{C}$.

- Die Kondition einer Matrix ist der Faktor zwischen dem größten und kleinsten Singulärwert.
- Eine schlecht konditionierte Matrix ist eine Matrix mit zu großer Kondition. Ab wann die Kondition zu schlechter Konditionierung führt, hängt von der Genauigkeit des verwendeten Rechenverfahrens bzw. Rechners ab.
- Eine Matrix ist singulär, wenn ihre Kondition unendlich ist. Die Determinante einer solchen Matrix ist 0.
- Der Rang einer Matrix ist die Dimension der linearen Hülle. Er hängt von der Anzahl nicht-singulärer Werte der Matrix ab, also der Anzahl der linear unabhängiger Zeilenvektoren der Matrix.

Als Singulärwertzerlegung bezeichnet man die Aufteilung einer realen $n \times m$ Matrix $\mathbf{C}$ mit $n \geq m$ in ein Produkt aus 3 Matrizen:

$$\mathbf{C} = \mathbf{U}\mathbf{\Sigma}\mathbf{V} \tag{3.13}$$

$\mathbf{U}$ ist eine $n \times m$ Matrix mit orthogonalen Spaltenvektoren. Für sie gilt also, dass $\mathbf{U}^T\mathbf{U} = \mathbf{I}$, wobei $\mathbf{I}$ die Identitätsmatrix ist. $\mathbf{V}$ ist eine orthonormale $m \times m$ Matrix, also gilt $\mathbf{V}^T\mathbf{V} = \mathbf{I}$. Σ ist eine $m \times m$ Diagonalmatrix mit Werten, die positiv oder gleich 0 sind und Singulärwerte heißen.

$\mathbf{C}$ lässt sich in folgende positiv definite symmetrische Matrizen zerlegen:

$$\mathbf{C}\mathbf{C}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^T \tag{3.14}$$

$$\mathbf{C}^T\mathbf{C} = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V} \tag{3.15}$$

Wenn $\mathbf{C}$ eine $m \times m$ Matrix ist, wird schnell deutlich, dass

$$\mathbf{C}\mathbf{C}^T = \mathbf{C}^T\mathbf{C} = \mathbf{R}\mathbf{\Sigma}^2\mathbf{R}^T \tag{3.16}$$

Mit diesen Gleichungen lassen sich Eigenvektoren und Eigenwerte von $\mathbf{C}^T\mathbf{C}$ als Spaltenvektoren von $\mathbf{R}$ bzw. Werte auf der Hauptdiagonalen von $\mathbf{\Sigma}$ identifizieren.

4 Rekonstruktionsmethoden

Die im vorherigen Kapitel erklärten Grundlagen werden hier angewandt. Es wird näher beleuchtet, welche Methoden zur Rekonstruktion von Oberflächen aus partikelbasierten Fluidsimulation in dieser Bachelorarbeit benutzt werden, was sie bewirken sollen und wie sie funktionieren. Dabei wird davon ausgegangen, dass eine SPH-Simulation bereits zugrunde liegt und die Partikel mit ihren physikalischen Werten, wie Masse und Dichte, zur Verfügung stehen. Nach der Simulation setzen die Oberflächenrekonstruktionen ein.

4.1 Rekonstruktion der Fluidoberfläche mit dem Massendichtefeld

Abbildung 4.1: Szene 2, die Methode von Müller et al. (links) mit dem Wasser-Material und (rechts) mit dem matten, grauen Material. Man erkennt die ausgebeulte Oberfläche im Mittleren Bereich.

4.1.1 Grundlegende Idee

Die Methode von Müller et al. [MCG03] basiert darauf, mit einer Funktion aus den Partikeln, die man nach der SPH-Methode erhalten hat, ein Skalarfeld zu erzeugen. Dieses wird anhand der Massendichte der Partikel errechnet. Danach definiert man eine Schranke, ab welcher Dichte man nicht mehr innerhalb des Fluids ist. Eine Isofläche entlang dieser Schranke entspricht somit der Oberfläche des Fluids und kann mit Point Splatting, Marching Cubes oder anderen Verfahren bestimmt werden.

4.1.2 Funktion und Berechnung der Methode

Für die Rekonstruktion braucht man eine Funktion, die für einen beliebigen Punkt **x** über alle Partikel j iteriert wird. Diese berechnet das sogenannte geglättete Dichtefeld, das die Massendichte im gesuchten Punkt angibt.

$$c_S(\mathbf{r}) = \sum_j m_j \frac{1}{\rho_j} W(\mathbf{r} - \mathbf{r}_j, h), \tag{4.1}$$

wobei $W(\mathbf{r}, h)$ der Glättungskern ist, der bestimmt, wie die Funktion für verschiedene Distanzwerte reagiert. In der Implementierung wurde der Poky6-Kern [MCG03] verwendet:

$$\text{Poly6-Kern} : W_{\text{poly6}}(\mathbf{r}, h) = \frac{315}{64\pi h^9} = \begin{cases} (h^2 - r^2)^3 & \text{wenn} \quad 0 \leq r \leq h \\ 0 & \text{sonst,} \end{cases} \tag{4.2}$$

Die Funktion des Glättungskerns wurde schon in den Grundlagen erläutert. Er ermöglicht es, den Wirkungsgrad der einzelnen Partikel einzugrenzen, indem nähere Partikel höher, weiter entfernte Partikel weniger gewichtet werden. Das hat den Vorteil, dass unerwünschte Effekte über große Distanzen hin nicht mehr auftreten. Außerdem kann man gerade in der Computergrafik viel Performance einsparen, da der Kern nur über eine bestimmte Distanz hinweg wirkt, also nur eine begrenzte Anzahl an Nachbarpartikeln betrachtet werden muss.

Nachdem die Formel für die Berechnung des Glättungskerns in die Berechnung des Dichtefeldes eingesetzt wird, summiert man für jeden gewählten Punkt **x** die durch den Kern gewichtete Massendichte von allen relevanten Partikeln. Die so entstehenden Werte entsprechen einzelnen Punkten des Skalarfeldes und mit ihnen lässt sich die Isofläche berechnen, die der gesuchten Fluidoberfläche entspricht.

4.2 Oberflächenrekonstruktion mit einem anisotropen Kern

4.2.1 Grundlegende Idee

Die Methode von Müller et al. führt unweigerlich zu einer welligen, gar ausgebeulten Oberfläche. Der isotrope Glättungskern führt zur Beulenbildung an der Oberfläche, da jedes Partikel einen festen Radius hat. Außerdem sind Stellen an denen die Partikeldichte dünner wird oft von mehr Beulen und Löchern gesäumt, eben weil die Partikel nach der Rekonstruktion von Müller et al. ihren bestimmten Radius einnehmen. Diese Schwächen wollten Yu und Turk mit ihrer Methode [YT10] ausgleichen.

Zum einen positionieren sie jedes Partikel für die Rekonstruktion neu. Die Partikel richten sich nach ihren Nachbarpartikeln. Somit ordnen sich Partikel automatisch gleichmäßiger an.

Abbildung 4.2: Szene 2, die Methode von Yu und Turk. (links) mit dem Wasser-Material und (rechts) mit dem matten, grauen Material, um die Formen besser zu erkennen.

Zum anderen entwickelten sie einen anisotropen Glättungskern (siehe Abb. 4.3), der einzelne Partikel in Richtung anderer Partikel streckt. In Randbereichen des Fluids, wo es keine Nachbarpartikel gibt, werden die einzelnen Partikel in Tangentialrichtung, also an der Kante entlang, gestreckt und in Normalenrichtung gestaucht. Das Ergebnis führt dazu, dass die Isoflächen einzelner Partikel im Skalarfeld Ellipsoide erzeugen, statt Kugeln, wie beim ersten vorgestellten Verfahren.

4.2.2 Funktion und Berechnung der Methode

Der erste Schritt ist die Neuberechnung der Positionen der Partikel. Dabei ist zu beachten, dass die neuen Positionen nur der Oberflächenrekonstruktion dienen und nicht die Positionen der Partikel für die SPH-Methode verändern, da sonst die Simulation beeinflusst wird.

Die neuen Positionen der Partikel $\mathbf{x}_i$ werden wie folgt berechnet:

$$\bar{\mathbf{x}}_i = (1-\lambda) + \lambda \sum_j w_{ij}\mathbf{x}_j / \sum_j w_{ij}, \tag{4.3}$$

wobei λ eine Konstante mit $0 < \lambda < 1$ ist und in dieser Arbeit als $\lambda = 0.9$ festgesetzt wird. w ist eine passende Gewichtungsfunktion. Ich verwende, die weiter unten stehende Gewichtungsfunktion (siehe Gleichung 4.6) auch an dieser Stelle. Dieser Schritt ist eine Art dreidimensionale Laplace'sche Glättung und führt unweigerlich zu leicht reduziertem Volumen, da alle Partikel zueinander und somit zur Mitte des Fluids angezogen werden.

Als nächstes berechnet man die Kovarianzmatrizen, also die Matrix mit den anisotropischen Informationen über naheliegende Partikel. Dafür verwendet man für jeden Partikel i die gewichtete Hauptkomponentenanalyse (WPCA), (siehe Kapitel 3.2). Die Kovariantmatrix

$\mathbf{C}_i$ wird benötigt, um zu ermitteln, in welchen Richtungen von Partikel i die meisten Nachbarpartikel vorkommen. Der aus der Isofläche des Partikels entstandene Ellipsoid soll sich letztendlich in diese Richtungen auswölben.

$\mathbf{C}_i$ wird wie folgt definiert:

$$\mathbf{C}_i = \sum_j w_{ij}(\mathbf{x}_j - \mathbf{x}_i^w)(\mathbf{x}_j - \mathbf{x}_i^w)^T / \sum_j w_{ij}, \tag{4.4}$$

wobei $\mathbf{x}$ das gewichtete Mittel ist, das auch in Gleichung 5.5 verwendet wird. Es ist wie folgt definiert:

$$x_i^w = \sum_j w_{ij}\mathbf{x}_j / \sum_j w_{ij}. \tag{4.5}$$

Die hier verwendete Funktion w_{ij} ist eine isotropische Gewichtungsfunktion über alle Partikel j, bezüglich des Partikels i:

$$w_{ij} = \begin{cases} 1 - (\left\|\mathbf{x}_i - \mathbf{x}_j\right\| / r_i)^3 & \text{wenn} \quad \left\|\mathbf{x}_i - \mathbf{x}_j\right\| < R_i, \\ 0 & \text{sonst.} \end{cases} \tag{4.6}$$

Durch den Wirkungsradius r_i, der hier auf $2h_i$, also den doppelten Radius des Partikels i gesetzt wurde, wird gewährleistet, dass einerseits nur Nachbarpartikel in unmittelbarer Nähe gewichtet werden, andererseits genug Partikel zusammenkommen, um eine sinnvolle Ausbeute anisotropischer Informationen zu gewährleisten.

Damit man die Kovarianzmatrix im Glättungskern verwenden kann, wird die Singulärwertzerlegung (siehe Kapitel 3.3) angewendet. Dabei werden zwei Rotationsmatrizen von der eigentlichen Diagonalmatrix, die die Dehnung bzw. Stauchung verursacht getrennt

$$\mathbf{C} = \mathbf{R}\mathbf{\Sigma}\mathbf{R}^T \tag{4.7}$$

$$\mathbf{\Sigma} = \text{diag}(\sigma_1, ..., \sigma_d), \tag{4.8}$$

wobei $\mathbf{R}$ eine oben erwähnte Rotationsmatrix mit Hauptachsen als Spaltenvektoren ist und $\mathbf{\Sigma}$ eine Diagonalmatrix mit Eigenwerten $\sigma_1 \geq ... \geq \sigma_d$ auf der Hauptdiagonalen ist. Um dieses Größenverhältnis der einzelnen Eigenwerte zu gewährleisten, wird geprüft, ob $\sigma_1 \geq k_r\sigma_d$, wobei $k_r > 1$ eine positive Konstante ist. Wie in der Arbeit von Yu und Turk ist hier $k_r = 4$. Ist die obige Bedingung erfüllt, so ist gewährleistet, dass die größte Varianz in einer Hauptachse wesentlich kleiner ausfällt als die kleinste Varianz einer anderen Hauptachse.
Wenn die Partikeldichte gering ausfällt, verliert das Ergebnis die sphärische Form. Deshalb wird die Anzahl der Nachbarpartikel mitgezählt. Sollte ein Partikel zu wenig Nachbarpartikel haben, in diesem Fall weniger als $N_\epsilon = 25$, wird anstatt der Diagonalmatrix aus Eigenwerten $\tilde{\mathbf{\Sigma}} = k_n\mathbf{I}$ gesetzt, wobei $k_n = 0.5$ und I die Identitätsmatrix ist, also eine Diagonalmatrix mit 0.5 auf der Hauptdiagonalen gefüllt. Außerdem wird die Matrix $\mathbf{C}$ im Falle einer gut gefüllten Nachbarschaft mit $k_s = 1400$ multipliziert, einem Faktor, der $\|k_s\mathbf{C} \approx 1\|$ erfüllt. Damit wird der Raum innerhalb des Fluids besser ausgefüllt.

Nach all diesen Modifikationen erhält man folgende Matrix

$$\tilde{\mathbf{C}} = \mathbf{R}\tilde{\mathbf{\Sigma}}\mathbf{R}^T \tag{4.9}$$

$$\tilde{\mathbf{\Sigma}} = \begin{cases} k_s \text{diag}(\sigma_1, \tilde{\sigma}_2, ..., \tilde{\sigma}_d) & \text{wenn} \quad N > N_\epsilon, \\ k_n \mathbf{I} & \text{sonst.} \end{cases} \tag{4.10}$$

wobei $\sigma_k = \max(\sigma_k, \sigma_1 / k_r)$, und N die Anzahl der Nachbarpartikel ist.

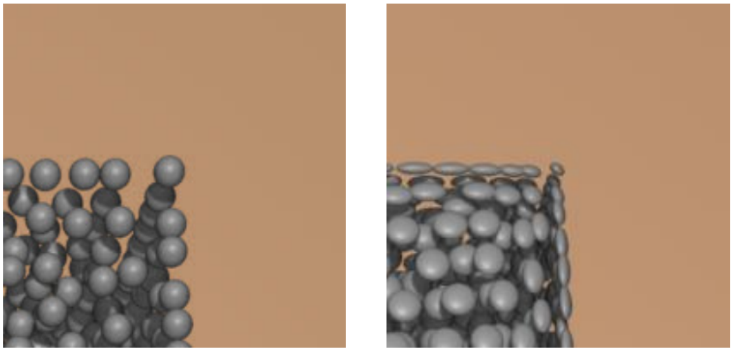

Abbildung 4.3: Vergleich zwischen isotropem und anisotropem Kern. Man sieht, dass die einzelnen Partikelrepräsentationen sich rechts mehr an die Form des Würfels anschmiegen [YT10].

Um im Glättungskern aus der Differenz der Raumkoordinaten eines Gitterpunktes und eines momentan betrachteten Partikels Rückschlüsse auf den Abstand zur Hülle des Ellipsoiden ziehen zu können, muss $\tilde{\mathbf{C}}_i$ invertiert und mit $\frac{1}{h_i}$ skaliert werden. Das Ergebnis wird als $\mathbf{G}_i$ bezeichnet:

$$\mathbf{G}_i = \frac{1}{h_i}\mathbf{R}\tilde{\mathbf{\Sigma}}^{-1}\mathbf{R}^T \tag{4.11}$$

Die symmetrische Matrix $\mathbf{G}_i$ kann nun mit dem Abstandsvektor zwischen einem Gitterpunkt der Rekonstruktionsmethode und dem momentan betrachteten Partikel multipliziert werden. Der Abstandsvektor wird abhängig von seiner Richtung im selben Maße verzerrt, wie der Ellipsoid. Somit kann die Länge des Ergebnisvektors nun mit einer gewünschten Wirkungsdistanz verglichen oder verrechnet werden, um zu erfahren, ob dieses Partikel nahe genug ist, um auf das Dichtefeld des zu Anfang definierten Punktes zu wirken. Für diesen Schritt gibt es einen Glättungskern, der sich vom Kern der ersten Methode unterscheidet. Yu und Turk stützen sich auf einige Paper von Becker und Teschner [BT07], der Kern wird aber auch in [Bec09] ausführlich erklärt:

$$W(\mathbf{x}', h) = W(q) = \begin{cases} \frac{1}{4\pi h^3}[(2-q)^3 - 4(1-q)^3] & \text{wenn} \quad 0 \leq q < 1 \\ \frac{1}{4\pi h^3}(2-q)^3 & \text{wenn} \quad 1 \leq q < 2 \\ 0 & \text{wenn} \quad q \geq 2, \end{cases} \tag{4.12}$$

wobei $\mathbf{x}'$ eine Position ist, h die Kernlänge und $q = \frac{|\mathbf{x}'|}{h}$. Im Verfahren von Yu und Turk ist also $\mathbf{x}' = \mathbf{x} - \bar{\mathbf{x}}_i$, $h = \|\mathbf{G}\mathbf{x}'\|$ und $q = \frac{|\mathbf{x}-\bar{\mathbf{x}}_i|}{\|\mathbf{G}\mathbf{x}'\|}$.

Zusammengefasst ergibt sich das neue geglättete Feld mit anisotropem Kern

$$\phi_{new}(\mathbf{x}) = \sum_j \frac{m_j}{\rho_j} W(\mathbf{x} - \bar{\mathbf{x}}_j, \left\|\mathbf{G}_j(\mathbf{x} - \bar{\mathbf{x}}_i)\right\|). \tag{4.13}$$

Mit den Gleichungen 4.1 und 4.13 erhält man also jeweils eine Funktion, die für einen beliebigen Punkt $\mathbf{x}$ im Skalarfeld einen Massendichte-wert berechnet. Um eine möglichst genaue Isofläche zu erhalten, braucht man nun ein Verfahren, das genug solcher Punkte $\mathbf{x}_i$ liefert.

5 Implementierung

Nachdem alle Partikelpositionen mit dem SPH-Verfahren berechnet wurden und mit einer Rekonstruktionsmethode eine Funktion zur Berechnung des Skalarfeldes aufgestellt wurde, muss die Oberfläche berechnet werden. In der Arbeit von Müller et al. werden dafür zwei Methoden vorgeschlagen: Point Splatting [MSHC99][Wes91] und Marching Cubes [LC87]. Es wären noch weitere Möglichkeiten denkbar, eine Isofläche aus dem Skalarfeld zu berechnen. In dieser Arbeit wird jedoch nur auf den in der Arbeit implementierten Marching Cubes-Algorithmus näher eingegangen.

5.1 Marching Cubes

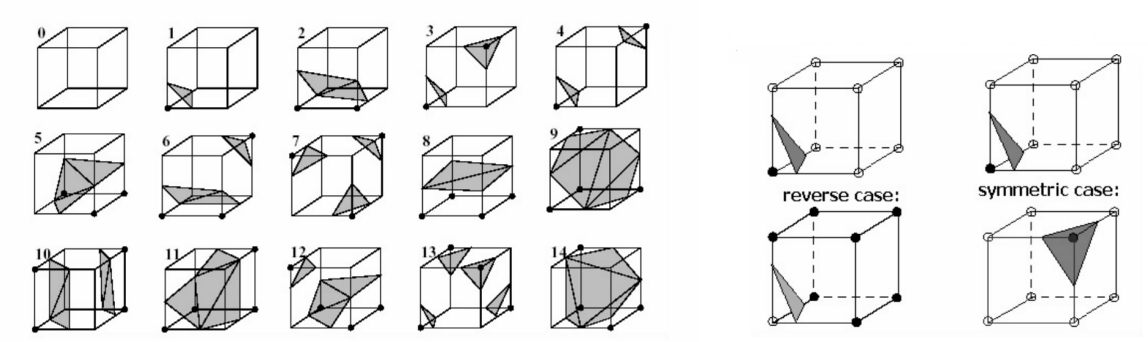

Abbildung 5.1: Die 256 Fälle lassen sich in (links) 15 Untergruppen aufteilen. Die anderen Fälle ergeben sich durch (rechts) Drehungen und Spiegelungen. [LC87].

Es wird ein dreidimensionales Gitter über den gesamten zu prüfenden Raum gespannt. Die einzelnen Knoten des Gitters werden in die Berechnung des Skalarfeldes eingesetzt. Das gewährleistet eine regelmäßige Abtastung der Isofläche. Somit erhält man für jeden Gitterpunkt v_{klm} die Distanzen zu allen Partikeln $\mathbf{x}_j$:

$$c_S(\mathbf{v}_{klm}) = \sum_j m_j \frac{1}{\rho_j} W(\mathbf{v}_{klm} - \mathbf{x}_j, h). \tag{5.1}$$

Danach wird eine Schranke definiert, in meinem Fall $t = 0.4$. Überschreitet der Dichtewert in einem Knoten die Schranke, ist der Knoten innerhalb des Fluides. Ist der Dichtewert darunter,

liegt der Knoten außerhalb. Zwischen Knotenpunkte, die innerhalb des Fluides liegen und ihren direkten Nachbarn, die außerhalb liegen muss also die Oberfläche verlaufen.

Nun werden je acht Knotenpunkte, die sozusagen einen Würfel umfassen, betrachtet, also für jeden Knoten v_{lmn} werden zusätzlich noch $v_{(l+1)m,n}$, $v_{l(m+1)n}$, $v_{(l+1)(m+1)n}$, $v_{lm(n+1)}$, $v_{(l+1)m(n+1)}$, $v_{l(m+1)(n+1)}$ und $v_{(l+1)(m+1)(n+1)}$ betrachtet.
Es gibt 256 verschiedene Konstellationen (siehe Abb. 5.1), welche dieser Knoten innerhalb und welche außerhalb des Fluids liegen. Jeder Konstellation wurde ein bestimmtes Oberflächenfragment zugeordnet. Somit ist es möglich, wenn man alle Knoten und ihre Nachbarn innerhalb des Würfels prüft eine abgeschlossene Oberfläche zu berechnen und darzustellen.

5.2 Optimierungen

Es gibt einige relevante Beschleunigungen, die die Rechenzeit deutlich verringern können.

Zu Beginn der Marching Cubes-Berechnung ließe sich Rechenzeit einsparen. Der Algorithmus berechnet in einem bestimmten dreidimensionalen Gitter Distanzen zu allen Partikeln. Wenn man von vornherein die Maxima und Minima der Partikelpositionen errechnet, lässt sich ein kleineres dreidimensionales Gitter mit weniger Zellen anlegen, das weniger Berechnungen benötigt.

Außerdem kann man, bevor man mit einer Rekonstruktionsmethode begonnen hat, ein mal über alle Partikel iterieren und sie räumlich in dreidimensionale Gitterzellen einteilen. Für jeden Marching Cubes-Knoten müsste man also nicht mehr alle Partikel anschauen, sondern nur die Partikel in den Zellen rund um diesen Knoten prüfen. Die Komplexität der Berechnung wird dadurch gesenkt.

6 Vergleiche

Dieses Kapitel beschäftigt sich mit den Ergebnissen der Arbeit. Bilder der Ergebnisse und Algorithmen beider Verfahren werden verglichen und durch Benchmarks ergänzt.

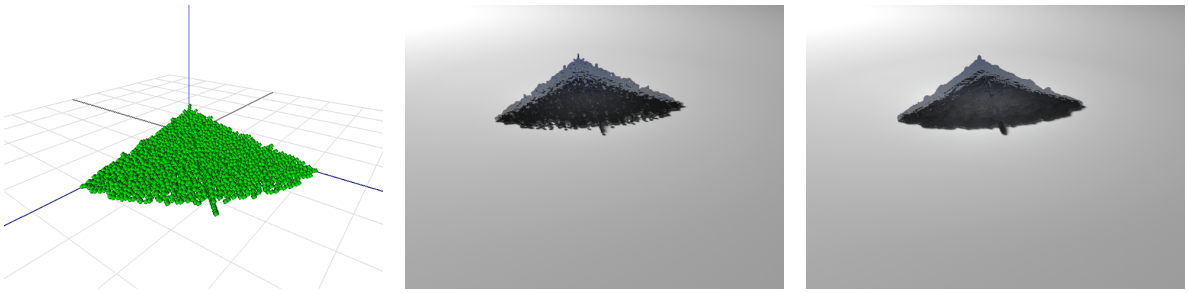

Abbildung 6.1: Szene 3, (links) ein Screenshot aus der Simulation, (mittig) die Methode von Müller et al. und (rechts) die Methode von Yu und Turk. Mit Wasser-Material gerendert.

6.1 Qualität

Vergleicht man die Ergebnisse aus Abb. 6.1 sieht man, dass die neuere Methode generell eine glattere Oberfläche erzeugt. Außerdem sieht man sehr gut, dass die Neupositionierung und die Streckung der anisotropen Kerne in Richtung der Masse besonders in Bereichen mit niedriger Partikeldichte gut funktioniert. Wie in dieser Darstellung vorne zu erkennen ist, wird eine klarere Kante erzeugt, die aussieht, als wäre sie von der Oberflächenspannung des Fluids erzeugt.

In Abb. 6.2 ist ein schwer zu rekonstruierendes Szenario mit weiträumig sehr niedriger Partikeldichte zu sehen. Die Methode von Yu und Turk verbindet viele Partikel zu gemeinsamen Gebilden. Das sieht man ganz vorne, wo einzelne voneinander getrennte Partikel zu schlauchartigen Fragmenten verbunden werden, sowie an mehreren Stellen in der Mitte der Szene.
Dabei ist zu beachten, dass in meinen Ergebnissen die Formen nicht mehr gestimmt haben, wenn beide Verfahren mit derselben Schranke für die Isofläche berechnet wurden. Die erst Methode blieb bei der Schranke $t = 0,4$, während die zweite Methode eine niedrigere, $t = 0,1$ für ansehnliche Ergebnisse benötigte. Dennoch sind die Ergebnisse durchaus Aussagekräftig.

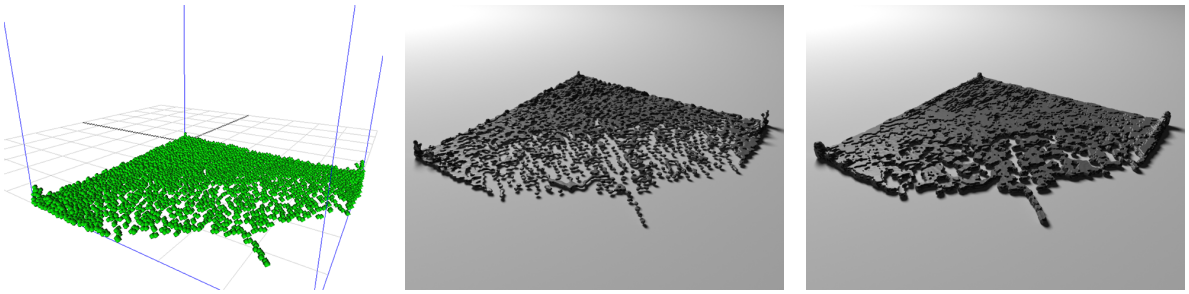

Abbildung 6.2: Szene 4, (links) ein Screenshot aus dem Simulationsprogramm, (mittig) die Methode von Müller et al. und (rechts) die Methode von Yu und Turk, um die Flächen besser zu erkennen, in mattem Grau gerendert.

6.2 Rechenaufwand

Ohne Optimierungen muss beim ersten Verfahren jeder Marching Cubes-Gitterpunkt Distanzen zu allen Partikeln berechnen. Das braucht viel Rechenzeit, die sich einsparen ließe. Mit der im vorherigen Kapitel vorgeschlagenen Optimierung, bei der die Partikel räumlich eingeordnet werden, muss jede Gitterzelle nur noch eine sehr kleine Zahl an naheliegenden Partikeln prüfen, die als konstant angesehen werden kann. Das reduziert die Programmlaufzeit wesentlich.

Das zweite Verfahren kann man in mehrere aufwändige Arbeitsschritte aufteilen: Die Neupositionierung der Partikel, die gewichtete Hauptkomponentenanalyse und die Isoflächenberechnung. Die Neupositionierung berechnet die Distanzen aller Partikel zueinander, die gewichtete Hauptkomponentenanalyse gewichtet im nicht optimierten Fall ebenfalls alle Distanzen. Die Isoflächenberechnung berechnet, wie beim ersten Verfahren, für jeden Gitterpunkt Distanzen zu allen Pertikeln.
Für alle wichtigen Rechenschritte wäre es eine Hilfreiche Optimierung, die Partikelnachbarschaften zu kennen. Eine Aufteilung in dreidimensionale Gitterzellen ist also auch hier sehr nützlich. Die Gewichtungsfunktion (siehe Gleichung 4.6) berücksichtigt ohnehin nur die Partikel, deren Distanz kleiner ist als der doppelte Partikelradius. Da diese Gewichtungsfunktion bei der Neupositionierung und bei der Hauptkomponentenanalyse verwendet wird und die Isoflächenberechnung eine noch kleinere Distanz vorsieht, läuft man nicht Gefahr, zu wenige Partikel zu berücksichtigen, wenn man den Radius für die Bestimmung Nachbarpartikel passend gewählt hat.

Das Marching Cubes Verfahren ist für die Brechnung der Laufzeit vernachlässigbar, da je nach Implementierung nur eine bestimmte konstante Anzahl an Rechenschritten notwendig ist, um den richtigen der 256 Fälle zu bestimmen und die Vertices und Polygone zu erzeugen oder zu speichern.

6.3 Benchmarks

Es wurde ein Phenom II X4 955 mit einem Takt von 3,78GHz für die Rechnungen verwendet. Die Methode von Müller et al. wurde unoptimiert gerechnet, da Caching der Nachbarpartikel in meinem Programm zu Fehlern führte. Die Methode von Yu und Turk prüft zu Beginn für jeden Partikel, welche der anderen Partikel in seiner potenziellen Reichweite sind. Nur diese werden von da an für die Neupositionierung sowie Gewichtung gewertet. Somit ist der neuere Algorithmus zwar auch nicht optimal aber mehr optimiert, als der ältere. Die

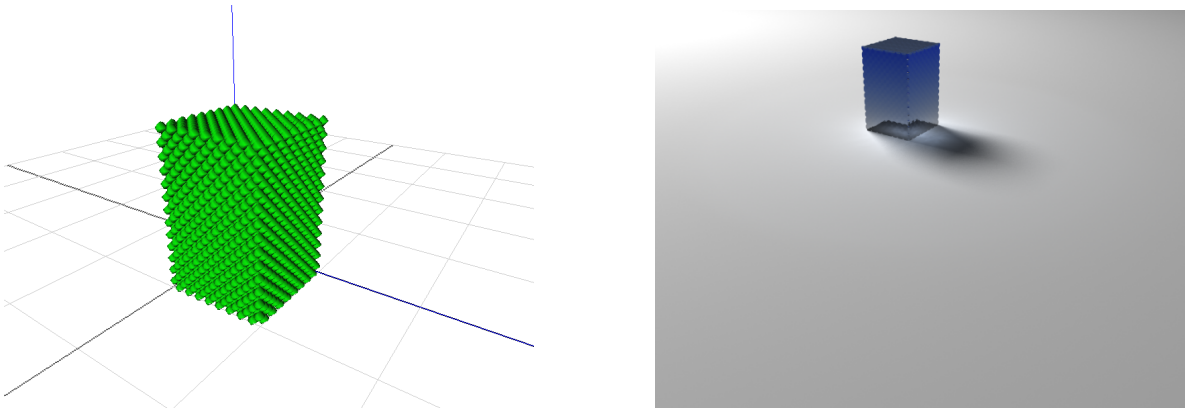

Abbildung 6.3: Szene 1, (links) ein Screenshot aus dem Simulationsprogramm und (rechts) die Methode von Müller et al. Gut zu sehen, der kompakte Quader, zu Beginn der Simulation.

gerenderte Szene enthält 3757 Partikel. Gemessen wurde nur die Zeit, die von Anbeginn der Rekonstruktion, bis zum Ende des Marchung Cubes-Algorithmus benötigt wurde. Weitere Ladezeiten, die danach beim Übertragen der Vertexpositionen und zeichnen der Polygone entstehen, wurden nicht mehr gemessen, da sie nicht relevant für die eigentliche Oberflächenrekonstruktion sind.

Nummer der Szene (Frame)	Methode 1	Methode 2
1 (1)	50	131
2 (145)	195	182
3 (294)	195	275
4 (527)	314	360
5 (665)	585	605
6 (839)	678	654

Alle Messergebnisse in Sekunden.

Die erste Szene ist kompakt (siehe Abb. 6.3), da die Partikel zu einem Quader geformt sind und das Gitter des Marching Cubes-Algorithmus nicht viele unnötige Zellen auf naheliegende Partikel prüfen muss. Dadurch ist die Zeit zu Beginn wesentlich kürzer als bei späteren Szenen. Die Gesamthöhe der Partikel nimmt nicht so schnell ab, wie deren Gesamtfläche durch von oben nachrückende Partikel zunimmt. Es sind also bei späteren Frames viel mehr Gitterzellen zu berechnen, die die Laufzeit in die Höhe treiben.

Man sieht auch, dass beim letzten Frame die neue Methode durch das Cachen der Nachbarpartikel schneller ist als die ältere. Das ist ein gutes Beispiel für den Sinn und Erfolg von Optimierungen.

7 Zusammenfassung und Ausblick

7.1 Fazit

Zusammenfassend lässt sich sagen, dass die Methode von Müller et al. sich schneller berechnen lässt. Das ließ sich auch durch die Messungen meistens bestätigen, obwohl die Methode von Yu und Turk ein wenig beschleunigt wurde. Betrachtet man die Algorithmen und die einzelnen Rechenschritte, so ist klar, dass die erste Methode das größere Potenzial hat, schneller zu sein, da komplexe Rechnungen mit Matrizen ausbleiben.

Optisch liefert die Methode von Müller et al. ein ausgebeultes Fluid, das seine Schwächen deutlich an den Rändern und an Stellen mit dünner Partikeldichte zeigt. Die Methode von Yu und Turk glättet das Ergebnis und erschafft damit, wie im in Kapitel 6 beschrieben, Strukturen, die realistischer wirken.

7.2 Ausblick

Wie eingangs erwähnt (siehe Kapitel 2), ist es möglich diese Berechnungen an modernere Rechnerarchitekturen anzupassen. Man kann Berechnungen der SPH und Marching Cubes multithreaded auf alle Prozessorkerne und ihre Threads aufteilen oder gar auf der GPU rechnen, da GPUs noch mehr Parallelisierung bieten. Die Verkürzung der Rechenzeit würde immens ausfallen.
Auch denkbar wäre es, eine große Simulation auf mehreren Rechnern zu parallelisieren. Je nach Größe des Projektes kann das den Rechenvorgang erneut beschelunigen.

Interessant für die Simulation von Fluiden sind noch viele weitere Aspekte, wie Oberflächenspannung oder Kollision mit Objekten, die wiederum verschiedene Eigenschaften haben und sich vielleicht von dem Fluid verformen oder bewegen lassen. Andere physikalische Eigenschaften, wie die temperaturabhängige Ausdehnung oder sogar verschiedene Aggregatzustände wären ebenfalls interessant.

Literaturverzeichnis

[APKG07] B. Adams, M. Pauly, R. Keiser, L. J. Guibas. Adaptively sampled particle fluids. *ACM Trans. Graph.*, 26(3):48, 2007. (Zitiert auf Seite 11)

[Bec09] M. Becker. *Particle-based animation.* Dissertation, University of Freiburg, 2009. (Zitiert auf Seite 23)

[Bli82] J. F. Blinn. A Generalization of Algebraic Surface Drawing. *ACM Trans. Graph.*, 1(3):235–256, 1982. doi:10.1145/357306.357310. (Zitiert auf Seite 10)

[BT07] M. Becker, M. Teschner. Weakly compressible SPH for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '07, S. 209–217. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007. (Zitiert auf den Seiten 9 und 23)

[CBP05] S. Clavet, P. Beaudoin, P. Poulin. Particle-based Viscoelastic Fluid Simulation. In *Symposium on Computer Animation 2005*, S. 219–228. 2005. (Zitiert auf Seite 9)

[DG96] M. Desbrun, M.-P. Gascuel. Smoothed Particles: A new paradigm for animating highly deformable bodies. In *Computer Animation and Simulation '96 (Proceedings of EG Workshop on Animation and Simulation*, S. 61–76. Springer-Verlag, 1996. (Zitiert auf Seite 9)

[GM77] R. A. Gingold, J. J. Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 181:375–389, 1977. (Zitiert auf den Seiten 7, 9, 13 und 14)

[HBD10] A. Hérault, G. Bilotta, R. A. Daltrymple. SPH on GPU with CUDA. *Journal of Hydraulic Research*, 2010. doi:10.3826/jhr.2010.0005. (Zitiert auf Seite 10)

[HKK07] T. Harada, S. Koshizuka, Y. Kawaguchi. Sliced data structure for particle-based simulations on GPUs. In *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, GRAPHITE '07, S. 55–62. ACM, New York, NY, USA, 2007. doi:10.1145/1321261.1321271. (Zitiert auf Seite 10)

[KC03] Y. Koren, L. Carmel. Visualization of Labeled Data Using Linear Transformations. In *Proc. IEEE Information Visualization (InfoVis'03), IEEE*, S. 121–128. 2003. (Zitiert auf den Seiten 6, 15 und 16)

[KC05] A. Kolb, N. Cuntz. Dynamic particle coupling for gpu-based fluid simulation. In *In Proc. of the 18th Symposium on Simulation Technique*, S. 722–727. 2005. (Zitiert auf Seite 10)

[LC87] W. E. Lorensen, H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '87, S. 163–169. ACM, New York, NY, USA, 1987. doi:10.1145/37401.37422. (Zitiert auf den Seiten 6 und 25)

[Luc77] L. B. Lucy. A Numerical Approach to Testing the Fission Hypothesis. *Astronomical Journal*, 82(12):1013–1024, 1977. (Zitiert auf den Seiten 7, 9 und 13)

[MCG03] M. Müller, D. Charypar, M. Gross. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, S. 154–159. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2003. (Zitiert auf den Seiten 7, 9, 10, 19 und 20)

[MHW04] R. E. Madsen, L. K. Hansen, O. Winther. Singular Value Decomposition and Principal Component Analysis. Technischer Bericht, 2004. (Zitiert auf Seite 17)

[Mon92] J. J. Monaghan. Smoothed Particle Hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 30(1):543–574, 1992. doi:10.1146/annurev.aa.30.090192.002551. (Zitiert auf den Seiten 7 und 9)

[MSHC99] K. Müeller, N. Shareef, J. Huang, R. Crawfis. High-Quality Splatting on Rectilinear Grids with Efficient Culling of Occluded Voxels. *IEEE Transactions on Visualization and Computer Graphics*, 5:116–134, 1999. (Zitiert auf Seite 25)

[Nat61] I. P. Natanson. *Theory of functions of a real variable (Teoria functsiy veshchestvennoy peremennoy)*. F. Ungar, New York,, rev. ed. Auflage, 1961. (Zitiert auf Seite 14)

[RBG+12] E. Rustico, G. Bilotta, G. Gallo, A. Herault, C. Del Negro. Smoothed Particle Hydrodynamics Simulations on Multi-GPU Systems. *16th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP 2008)*, 0:384–391, 2012. (Zitiert auf Seite 10)

[SP09] B. Solenthaler, R. Pajarola. Predictive-corrective incompressible SPH. *ACM Trans. Graph.*, 28(3):40:1–40:6, 2009. doi:10.1145/1531326.1531346. (Zitiert auf Seite 10)

[Wes91] L. A. Westover. SPLATTING: A Parallel, Feed-Forward Volume Rendering Algorithm. Technischer Bericht, Chapel Hill, NC, USA, 1991. (Zitiert auf Seite 25)

[YT10] J. Yu, G. Turk. Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, S. 217–225. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2010. (Zitiert auf den Seiten 6, 7, 20 und 23)

[ZB05] Y. Zhu, R. Bridson. Animating sand as a fluid. *ACM Trans. Graph.*, 24(3):965–972, 2005. doi:10.1145/1073204.1073298. (Zitiert auf Seite 10)

Erklärung

Hiermit versichere ich, diese Arbeit selbständig verfasst und nur die angegebenen Quellen benutzt zu haben.

(Alexander Tivonenko)