

Seminarfacharbeit der Oberstufe  
Schuljahre 2018 bis 2020

# **Bestimmung der Widerstandsbeiwerte definierter Körper mittels 3D-Simulation umströmender Gase**

Fachbetreuer: Herr Brenner  
Seminarfachbetreuer: Frau Bangsow-Bösa

Schüler: Adrian Kühn (12b)  
Frank Long (12a)  
Paul Alexander Marschall (12a)

Datum: 20.12.2019

# Inhaltsverzeichnis

<b>1 Mathematisch-physikalische Betrachtung von Fluiden</b>	<b>4</b>
1.1 Die physikalischen Grundlagen der Flüssigkeitsdynamik . . . . .	4
1.2 Zum Problem der linearen Näherung . . . . .	5
<b>2 Mikroskopische Herangehensweise an Strömungssimulationen</b>	<b>6</b>
2.1 Beweggründe und Grundidee . . . . .	6
2.2 Umsetzung eines Testprogramms . . . . .	6
2.3 Auswertung dieser Simulationsmethode . . . . .	7
<b>3 Umsetzung des Computerprogramms zur Simulation strömender Fluide</b>	<b>7</b>
3.1 Vorgehen bei der Programmierung . . . . .	7
3.1.1 Einführung der Lattice-Boltzmann-Methode . . . . .	7
3.1.2 Umsetzung des Simulationsverfahrens im 2D-Raum . . . . .	8
3.1.3 Umsetzung des Simulationsverfahrens im 3D-Raum . . . . .	8
3.1.4 Optimierung der Strömungssimulation durch Kollisionstherme . . . . .	9
3.2 Virtuelle Körper im Programm . . . . .	11
3.2.1 Vorbetrachtung zum Umgang mit Körpern im Programm . . . . .	11
3.2.2 Importieren von Körpern und Umgang mit Dateiformaten . . . . .	12
3.2.3 Umwandlung geometrischer Körper in boolsche Körper . . . . .	12
3.2.4 Direkter Import boolscher Körper . . . . .	13
3.2.5 Graphische Darstellung von Körpern . . . . .	13
3.3 Grundstruktur des Simulationsprogramms . . . . .	13
<b>4 Anwendung unseres Simulationsprogramms</b>	<b>14</b>
4.1 Die Verbindung zwischen Theorie und Realität . . . . .	14
4.2 Optimierung der Cw-Wert-Berechnung . . . . .	15
4.3 Ergebnisse der Cw-Wert-Berechnung . . . . .	16
4.3.1 Ergebnisse für einfache Körper . . . . .	16
4.3.2 Ergebnisse des Lamborghinis und eingescannter Körper . . . . .	17
4.4 Fehlerbetrachtung für die von uns entwickelte Simulationsmethode . . . . .	17
4.5 Einordnung der Ergebnisse . . . . .	18
<b>5 Reflexion unserer Arbeit</b>	<b>19</b>
<b>6 Anhang</b>	<b>20</b>

## **Einleitung**

Jede Form der Mobilität auf unserem Planeten hängt von ihrem Grad an Effizienz ab. So haben wir die aerodynamischen Eigenschaften von Pinguinen und Robben auf unsere Autos, Züge und Flugzeuge übertragen. Doch wir müssen auch das Prinzip hinter der Bewegung eines Festkörpers durch ein Fluid verstehen, um unsere Technik weiterzuentwickeln. Maßgeblich dafür ist die Betrachtung des Strömungswiderstandskoeffizienten, auch Cw-Wert genannt. Untersuchungen dieses Maßes an Windschlüpfigkeit eines Körpers werden derzeit in Windkanälen durchgeführt. Doch diese Anlagen sind teuer und die Instandhaltung erweist sich als aufwendig. So haben wir uns die Frage gestellt, ob diese Untersuchungen alternativ auch mit Hilfe entsprechender Software durchgeführt werden können.

In unserer Seminarfacharbeit wollen wir ein Programm erstellen, das es ermöglicht, mittels eines virtuellen Windkanals Strömungen um Körper zu analysieren. Dafür müssen wir uns mit den grundlegenden physikalischen und mathematischen Modellen der Fluidodynamik auseinandersetzen. Anschließend sehen wir eine Übertragung dieser Modelle auf eine numerisch umsetzbare Simulationsmethode vor. In dieser Methode gilt es, die Spezifikationen unseres virtuellen Windkanals sowie analysierbare Größen festzulegen. Abschließend können wir unsere Simulation an Normkörpern eichen.

Mit unserem Programm soll es möglich sein, dreidimensionale Objekte zu importieren. Dafür werden die physikalischen Eigenschaften des umströmenden Fluides festgelegt und in einen entsprechenden Simulationsraum eingefügt. Im Anschluss daran wird eine Simulation durchgeführt, bei der das Fluid um den Körper strömt. Hierbei werden als Kernstück unserer Arbeit spezifische Daten gesammelt, anhand derer wir im späteren Verlauf auf den Cw-Wert des Körpers schließen können. Zuletzt wollen wir eine möglichst umfangreiche Visualisierung der Strömung umsetzen und die gesammelten Daten veranschaulichen.

Wir selbst haben dieses Thema gewählt, weil wir uns sehr für Strömungsprobleme, speziell in der Technik interessieren. Mit unserem Programm soll es dem Hobbybastler oder Schüler im Unterricht möglich sein, Objekte hinsichtlich ihrer aerodynamischen Eigenschaften zu untersuchen.

An dieser Stelle möchten wir uns bei unserem Fachbetreuer Herrn Brenner für seine tatkräftige Unterstützung bezüglich des Verständnisses physikalischer und mathematischer Modelle bedanken. Weiteren Dank möchten wir dem Schülerforschungszentrum und damit Herrn Paulig und Herrn Dr. Wagner aussprechen, die uns die Arbeit an einem Windkanal ermöglicht und uns einen 3D-Scanner zur Verfügung gestellt haben. Zudem danken wir unserer Projektbetreuerin Frau Bangsow-Bösa, bezüglich ihrer Unterstürzung zur Ausarbeitung des schriftlichen Teils unserer Arbeit. Des Weiteren möchten wir unseren Familien für die moralische Unterstützung danken sowie allen anderen, die uns beim Bewältigen unserer Aufgaben unterstützt haben.

# 1 Mathematisch-physikalische Betrachtung von Fluiden

## 1.1 Die physikalischen Grundlagen der Fluidodynamik

Ziel unserer Arbeit ist es, den Strömungswiderstandskoeffizienten von Körpern simulativ zu ermitteln. Im Zuge dessen beschäftigen wir uns in diesem Kapitel zunächst mit den physikalischen Grundlagen der Fluidodynamik. Dementsprechend betrachten wir zunächst das makroskopische Modell der Fluidodynamik, welches auf den Navier-Stokes-Gleichungen basiert. Dieses beschreibt die Dynamik von Volumenelementen im Fluid. Die Gleichungen gehen jedoch über den Schulstoff weit hinaus, weshalb wir uns mit den grundlegenden Elementen der Gleichungen beschäftigt haben. So werden, für uns zunächst unbekannte, Operatoren und Operanden verwendet, die an den Gleichungen (1) bis (5) nachvollzogen werden können.[35, 36]

$$\nabla \cdot \vec{v} = 0 \quad (1)$$

$$\rho \left( \frac{d \vec{v}}{dt} \right) = -\nabla p + \mu \nabla^2 \vec{v} + \rho F \quad (2)$$

$$\rho \left( \frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + w \frac{\partial u}{\partial z} \right) = \rho g_x - \frac{\partial p}{\partial x} + \mu \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) \quad (3)$$

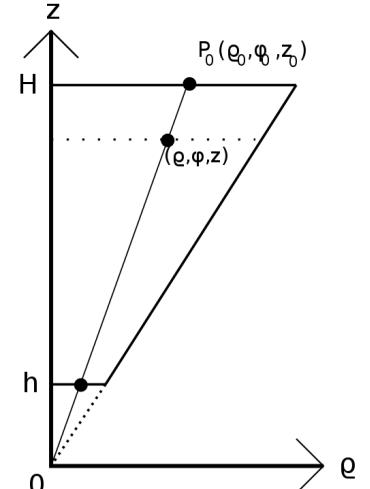
$$\rho \left( \frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + w \frac{\partial v}{\partial z} \right) = \rho g_y - \frac{\partial p}{\partial y} + \mu \left( \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} \right) \quad (4)$$

$$\rho \left( \frac{\partial w}{\partial t} + u \frac{\partial w}{\partial x} + v \frac{\partial w}{\partial y} + w \frac{\partial w}{\partial z} \right) = \rho g_z - \frac{\partial p}{\partial z} + \mu \left( \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2} \right) \quad (5)$$

Die Gleichungen beschreiben das Verhalten von inkompressiblen Fluiden. Die Gleichung (1) stellt die Massenerhaltung dar. Die Gleichung (2), deren drei Komponenten in den Gleichungen (3) bis (5) separat erneut ausgeschrieben stehen, stellt die Impulserhaltung, sprich das zweite Newton'sche Axiom, im Fluid dar. In beiden Gleichungen kommt unter anderem der Operator Nabla ( $\nabla$ ) vor, der als Vektor betrachtet werden kann. Dieser erzeugt je nach Anwendung auf eine Größe ein Vektorfeld oder ein Skalarfeld. Der Operator  $\nabla$  steht in der Gleichung (1) vor der physikalischen Größe Geschwindigkeit. Diese ist ebenfalls ein Vektor, weshalb das Skalarprodukt beider Größen ein Skalarfeld, die sogenannte Divergenz, erzeugt. Diese beschreibt das Vorkommen von Senken und Quellen im Vektorfeld der Geschwindigkeit. Ist sie gleich null, kommen beide Phänomene nicht vor und wir können, wie erwähnt, von Massenerhaltung ausgehen. In der zweiten Gleichung steht  $\nabla$  vor der Größe Druck. Der Druck in einem Fluid ist eine skalare Größe. Bei der Anwendung von  $\nabla$  ergibt sich demnach ein Vektorfeld, der sogenannte Gradient. Dieser zeigt in die Richtung der größten Druckänderung und stellt damit die Richtung der wirkenden Kraft dar. Die Änderung des Impulses wird durch alle einwirkenden Kräfte beschrieben, die durch die rechte Seite der Gleichung dargestellt werden. Um die Navier-Stokes-Gleichungen anwenden zu können, benötigt man Anfangsbedingungen. Eine solche Anwendung wollen wir im Folgenden an einem einfachen Beispiel nachvollziehen. Dabei fiel die Wahl auf einen durchflossenen Trichter (Vergleich Abbildung 1).

Da Turbulenzen numerisch schwer zu erfassen sind, wird von einer laminaren Strömung ausgegangen. Damit die Geschwindigkeit an einzelnen Stellen des Strömungsfeldes zeitunabhängig ist, ist die Strömung weiterhin stationär. Wir nehmen zudem an, dass die Flüssigkeit ideal ist, womit sie auch die Voraussetzung der Inkompressibilität der Navier-Stokes-Gleichungen erfüllt.

Ziel ist es nun, eine Geschwindigkeitsgleichung für Volumenelemente des Fluids aufzustellen.  $P_0$  ist der Startpunkt eines Volumenelementes, gesucht ist die Geschwindigkeit auf einem bestimmten Punkt von der Stromlinie. Dabei spannt der Kegel von  $H$  bis  $h$ . Für das mathematische Modell nehmen wir an, dass die Geschwindigkeit in der Ebene  $H$  gleich 0 ist und dass die Volumenelemente in einem Strahl sich zum Fokuspunkt  $0^a$  hinbewegen. Wir gehen davon aus, dass die Geschwindigkeit auf dieser Bahn linear zunimmt<sup>b</sup>. Eine weitere Annahme ist, dass die Geschwindigkeit innerhalb einer Ebene gleich ist. Wir können also die Torricelli-Formel  $v = \sqrt{2g(H-z)}$  für den Betrag der Geschwindigkeit verwenden.



<sup>a</sup>Der Koordinatenursprung

<sup>b</sup>Die entsprechenden Bahngleichungen können Sie im Anhang auf Seite 22 vorfinden.

Abbildung 1: Modell der Bahn eines Volumenelementes in einer Trichterströmung [61]

Neben der Gravitation wird das Fluid lediglich von den Trichterwänden beeinflusst. Wir nehmen Gleichung (6) für den ortsabhängigen Geschwindigkeitsvektor an.

$$\vec{v} = \frac{\sqrt{2g(H-z)}}{\sqrt{\rho^2 + z^2}} \begin{pmatrix} -\rho \\ 0 \\ -z \end{pmatrix} \quad (6)$$

Diese Gleichung müssen wir im Folgenden überprüfen. Dafür setzen wir sie in Gleichung (1), die wir an das Zylindrische Koordinatensystem anpassen [20], ein.

$$\frac{\rho}{z} = \frac{\rho_0}{z_0} \quad (7)$$

$$\varphi = \varphi_0 \quad (8)$$

$$\nabla \cdot v = \frac{1}{\rho} \frac{d}{d\rho} (\rho v_\rho) + \frac{1}{\rho} \frac{d}{d\varphi} (v_\varphi) + \frac{d}{dz} v_z \quad (9)$$

Es ergibt sich der in Gleichung (10) nachzuvollziehende Ausdruck.

$$\nabla \cdot v = \frac{1}{\rho} \frac{d}{d\rho} \left( \rho \frac{\sqrt{2g(H-z)}(-\rho)}{\sqrt{\rho^2+z^2}} \right) + \frac{d}{dz} \left( \frac{\sqrt{2g(H-z)}(-z)}{\sqrt{\rho^2+z^2}} \right) = -\frac{\rho(4H-5z)}{\sqrt{(\rho^2+z^2)2g(H-z)}} \neq 0 \quad (10)$$

Dieser stellt eine Ungleichung dar, weshalb wir schlussfolgern können, dass Gleichung (6) Gleichung (1) nicht erfüllt. So ist die Divergenz nicht für alle Punkte im Trichter gleich null.

In einem alternativen Verfahren könnte man mittels computergestützten Näherungsmethoden für Gleichung (2) einen alternativen Ausdruck für Gleichung (6) ermitteln. Eine exakte Lösung von Gleichung (2), angewendet auf die Trichterströmung, ist uns derzeit nicht bekannt.

Zusammenfassend kann geschlussfolgert werden, dass die angenommene Gleichung für die Geschwindigkeit  $v$  falsch ist.

## 1.2 Zum Problem der linearen Näherung

Im Kapitel 1.1 haben wir computergestützte Näherungsmethoden, die auf die Navier-Stokes-Gleichungen angewendet werden können, erwähnt. Eine dieser Methoden ist die lineare Näherung, die wir hier nun etwas genauer betrachten wollen.

Dabei ist die allgemeine Form einer solchen Näherung das explizite Euler-Verfahren. Dieses ist eine einfache Methode zum numerischen Lösen einer Differentialgleichung. In ihr wird in vorher festgelegten Abständen der Zuwachs des Graphen berechnet, welcher im Anschluss den sich anschmiegenden Graphen vervollständigt.<sup>1</sup>

Das numerische Verfahren muss angewandt werden, da für zahlreiche Differentialgleichungen eine explizite Lösungs-darstellung nicht möglich ist.

Im folgenden Beispiel ist der y-Wert einer Funktion als Anfangswert vorhanden. Um nachfolgende y-Werte ermitteln zu können, kann die Gleichung (11) verwendet werden:

$$f(x + \Delta x) = f(x) + f'(x)\Delta x + r(x, \Delta x) \quad (11)$$

In 3D-Koordinatensystemen, mit denen in unserer Simulation gearbeitet wird, verändern sich hingegen drei Werte, wodurch Gleichung (11) durch Gleichung (12) ersetzt wird.

$$f(x_0 + \Delta x, y_0 + \Delta y) \approx f(x_0, y_0) + \frac{\partial f}{\partial y} \Delta y + \frac{\partial f}{\partial x} \Delta x \quad (12)$$

Den Anstieg der Koordinatenebenen rechnet man mit Hilfe der jeweiligen partiellen Ableitung aus. Man erhält einen 3D-Graphen.

Das  $\approx$  Zeichen in der Gleichung (12) kommt durch die Vernachlässigung von  $r(x, \Delta x)$ . So ist  $r$  (Vergleich Abbildung 2) der Abstand von der Ableitung zur eigentlichen Funktion, weswegen sich der anschmiegende Graph immer weiter von dem ursprünglichen Graphen entfernt. Jedoch wird  $r$  immer kleiner, je kleiner  $\Delta x$  ist. Daher kann man  $r$  beim numerischen Lösen von Differentialgleichungen außer Acht lassen, weil man hier mit minimalen  $\Delta x$ -Werten rechnet.

Um bestimmen zu können, ob ein Graph steiler oder flacher wird, benötigt man die 2. Ableitung seiner Funktion. Schließlich erhält man aus Gleichung (11) Gleichung (13). Den zusätzlichen Term  $r$  nähern wir durch Verwendung des Mittelwertsatzes.

$$f(x_0 + \Delta x) \approx f(x_0) + f'(x_0)\Delta x + \frac{f''(x_0)}{2}\Delta x^2 \quad (13)$$

Als Beispiel für eine solche Rechnung betrachten wir nun eine Funktion  $f(x) = \ln(x)$ . Da die Ableitung von  $\ln$  schon bekannt ist, erhält man für  $y$  die Gleichung (14).

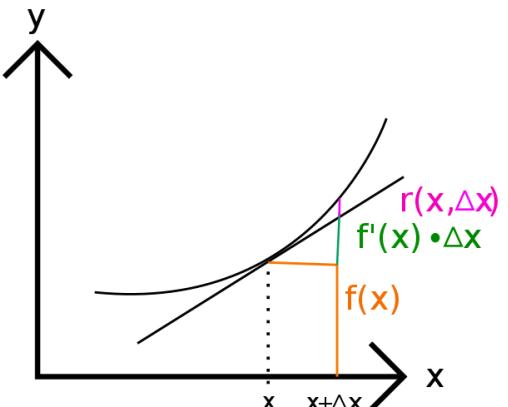


Abbildung 2: Näherung einer Funktion mit Hilfe ihrer Ableitung [61]

<sup>1</sup>Anhang S.22 Abbildung 16 : Explizites Euler-Verfahren

$$y \approx \ln(x) + \frac{1}{x}\Delta x + \left(-\frac{1}{x^2}\right)\Delta x^2 \quad (14)$$

Wir legen  $\Delta x$  auf 0.01 fest und betrachten die Funktion sowie ihre Näherung im Intervall von 5 bis 5.07. Die Punkte der Näherung berechnen wir nun numerisch mit Hilfe eines Computers. Jetzt können wir die Diskrepanz zwischen den berechneten Punkten ermitteln. Diese beläuft sich für den  $x$ -Wert von 5.07 auf 0.02.

## 2 Mikroskopische Herangehensweise an Strömungssimulationen

### 2.1 Beweggründe und Grundidee

Im Kapitel 1.1 haben wir festgestellt, dass eine Simulation auf Grundlage makroskopischer Modelle kaum umsetzbar ist. Ausweichend versuchen wir in diesem Kapitel eine Simulation auf Grundlage der Gesetze der Thermodynamik umzusetzen. Wir wechseln also die Perspektive von der makroskopischen zur mikroskopischen Betrachtung von Fluiden.

Die Grundidee ist es, die Bestandteile eines Fluides, Atome oder Moleküle, zu simulieren. Gemeint ist eine Anzahl  $N$  an möglichst günstigen Bestandteilen zu bestimmen und ihnen anhand von Naturkonstanten einen Simulationsraum zu geben. Der einzelne Bestandteil ist nun im Sinne der Informatik ein Objekt, dem wir Attribute und Methoden zuweisen können. Ebenso ist der zu untersuchende Körper ein Objekt, jedoch anderer Klasse. Mit Hilfe ihrer Methoden können wir versuchen, diese Objekte möglichst natürlich im Simulationsraum interagieren zu lassen. Anhand von bestimmten, auftretenden Events lässt sich im Anschluss auf die Windschläufigkeit des Testkörpers schließen.

### 2.2 Umsetzung eines Testprogramms

Als Fluid haben wir zunächst den Stoff Helium ausgewählt. Helium ist ein Edelgas, weshalb es als reiner Stoff keine Moleküle ausbildet. Die Bestandteile des Fluids können daher geometrisch als Kugeln, Heliumatome, betrachtet werden. Die Temperatur unseres Simulationsraumes legen wir selbst auf 0°C, beziehungsweise 273,15K, fest. Jetzt können wir mittels der kinetischen Gastheorie und Naturkonstanten alle Parameter unseres Simulationsraumes festlegen. Sie sind in Tabelle 1 zu finden.

Tabelle 1: Festgelegte und ermittelte Parameter unseres Simulationsraumes [2]

Größe	Variable	Wert	Festlegung / Formel / Naturkonstante
Temperatur	$T$	273,15K	Festlegung
Atomanzahl	$N$	z.B. 100	Festlegung, variabel
Atomradius	$r_{Atom}$	$3,1 \cdot 10^{-11} m$	Naturkonstante
Atomvolumen	$V_{Atom}$	$1,2 \cdot 10^{-31} m^3$	$V_{Atom} = \frac{4}{3} \cdot \pi \cdot r_{Atom}^3$
Atomare Masseneinheit	$u$	$1,660539040 \cdot 10^{-27} kg$	Naturkonstante
Atommasse	$m_{Atom}$	$6,64647 \cdot 10^{-27} kg$	$4,0026 \cdot u$
Bolzmann-Konstante	$K$	$1,38065 \cdot 10^{-23} J \cdot K^{-1}$	Naturkonstante
Dichte von Helium (Normalsdruck)	$\rho$	$0,1785 kg \cdot m^{-3}$	Naturkonstante, für $T = 273,15K$
Ø kinetische Energie eines Atoms	$E_{kinetisch}$	$5,65687 \cdot 10^{-21} J$	$E_{kinetisch} = \frac{3}{2} \cdot K \cdot T$
Ø Geschwindigkeit eines Atoms	$v_{\varnothing}$	$1304,69 m \cdot s^{-1}$	$v_{\varnothing} = \sqrt{2 \cdot E_{kinetisch} \cdot m_{Atom}^{-1}}$

Damit ist jedoch noch nicht der Simulationsraum als Raum bestimmt. Wir wählen ihn hier zunächst als Würfel, dessen Kantenlänge  $s$  mittels obiger Werte ermittelt werden kann. Gesucht ist demnach:  $s(N)$  (Vergleich Gleichung (15)).

$$s(N) = V(N)^{\frac{1}{3}} = (N \cdot V_{Atom} \cdot C)^{\frac{1}{3}} \quad (15)$$

Das Volumen des Simulationsraumes entspricht nicht der Summe der Atomvolumen. Es besteht jedoch ein fester Zusammenhang zwischen beiden Größen, der hier mittels  $C$  ausgeglichen wird.  $C$  kann über die Dichte von Helium bestimmt werden. Es ergibt sich folgender Ausdruck (16):

$$C = \frac{m_{Atom}}{\rho \cdot V_{Atom}} = 310303,45 \quad (16)$$

Jetzt kann eine Anzahl  $N$  an Atomen definiert werden. Sie erhalten innerhalb des Simulationsraumes eine zufällige Startposition sowie eine zufällige Anfangsgeschwindigkeit. Der Betrag der Anfangsgeschwindigkeit kann dabei von der berechneten durchschnittlichen Geschwindigkeit der Atome abweichen. Grundlegend dafür ist die Energieverteilung nach Boltzmann. Die Atome verfügen zunächst nur über eine Methode, die ihre Position anhand eines Zeitschrittes und ihrer Geschwindigkeit aktualisiert. Zudem prallen sie an den Grenzen des Simulationsraumes vollkommen elastisch ab. Wir lassen keine Kräfte auf die Atome einwirken und sie führen auch keine Stöße untereinander aus. Dieses Programm testen wir in erster Linie auf seine Eignung bezüglich seiner Laufzeit.<sup>2</sup>

<sup>2</sup>Eine Visualisierung finden Sie im Anhang: Seite 23, Abbildung 17

## 2.3 Auswertung dieser Simulationsmethode

Ohne Krafteinwirkungen oder komplexere Aktualisierungsmethoden der Atome, beziehungsweise Bestandteile, läuft die Simulation bis zu einer Atomanzahl von 5000 flüssig. Mit dieser Atomanzahl befinden wir uns im Bereich einer Kantenlänge des Simulationsraumes von  $10^{-8} m$ . Zudem haben wir bisher die Interaktion der Teilchen untereinander außer Betracht gelassen. Stöße von Kugeln in einem dreidimensionalen Raum zu berechnen, stellt kein Problem dar. Letztendlich müssen wir dem System jedoch eine Uhr geben, die es in bestimmten Zeitschritten aktualisiert. Interaktionen von mehr als zwei Objekten innerhalb eines Zeitschrittes zu verfolgen, würde die Rechenaufwendigkeit des Programms weiter erhöhen. Die mikroskopische Herangehensweise eignet sich damit nicht zur Strömungssimulation.

# 3 Umsetzung des Computerprogramms zur Simulation strömender Fluide

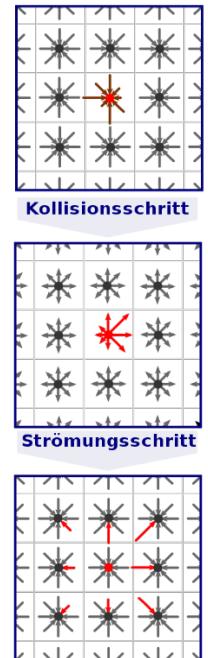
## 3.1 Vorgehen bei der Programmierung

### 3.1.1 Einführung der Lattice-Boltzmann-Methode

Sowohl die makroskopische als auch die mikroskopische Betrachtung der Fluide konnten nicht zu einer zufriedenstellenden Fluidsimulation führen. Dementsprechend bedienen wir uns nun der mesoskopischen Betrachtungsebene, welche zwischen den anderen agiert. Hierbei verwenden wir Simulationsmethoden der numerischen Strömungsmechanik, welche die Komplexität der strömungsmechanischen Probleme, basierend auf den Navier-Stokes-Gleichungen, approximativ und numerisch lösen. Anstatt einzelne Teilchen zu simulieren, werden nun näherungsweise Rechengitter verwendet, was in Abbildung 3 veranschaulicht wird. Die gebräuchlichsten Simulationsmethoden in der numerischen Strömungsmechanik zur Simulation von strömenden Fluiden sind die Finite-Differenzen-Methode (FDM) und die Finite-Volumen-Methode (FVM). Wir haben diese Methoden jedoch nicht direkt angewendet und sind stattdessen zur Lattice-Boltzmann-Methode (LBM) gewechselt. Die Lattice-Boltzmann-Methode war aus unserer Sicht anschaulicher, effizienter und leichter in einer Programmiersprache realisierbar. Aufgrund der internen Struktur, welche einen geringen Speicher- und Rechenbedarf je Zelle einschließt, eignet sich das Verfahren unter anderem zur Berechnung von Strömungen in komplexen Geometrien.

Die Lattice-Boltzmann-Methode wurde Ende der 1980er Jahre [13] entwickelt und hat seine theoretische Basis in der statistischen Physik. Wie oben schon erwähnt, wird der Simulationsraum oder die Simulationsebene dabei durch ein Gitter diskretisiert. Die Wechselwirkung der mikroskopischen Teilchen wird durch die Boltzmann-Gleichung beschrieben. (17)[13]

$$\left( \frac{\partial}{\partial t} + \vec{v} \cdot \nabla_{\vec{x}} + \frac{\vec{F}}{m} \cdot \nabla_{\vec{v}} \right) f(\vec{x}, \vec{v}, t) = \frac{\partial f}{\partial t} \Big|_{Stoß}$$



(17) Abbildung 3:  
Teilschritte der  
Lattice-Boltzmann-  
Methode [56]

Die linke Seite der Gleichung stellt die Verteilungsdichte eines Fluides als totale Zeitableitung dar. Diese Verteilungsdichte wird durch das Kollisionsintegral auf der rechten Seite der Gleichung beschrieben.

Zuerst gehen wir auf das Lattice-Boltzmann-Verfahren im zweidimensionalen Raum, also in der Ebene, ein. Wie oben schon erwähnt, wird die Ebene durch ein Gitter in viele gleichgroße Quadrate, sogenannte Zellen eingeteilt. Jeder Zelle werden acht Richtungspfeile und ein Nullpfeil zugeordnet. Die Richtungspfeile stellen dar, wie wahrscheinlich die Geschwindigkeit der Teilchen, die einer Zelle zugeordnet sind, in die Richtung des Pfeiles an der jeweiligen Zelle auftritt. Ein Fluidpartikel kann pro Zeitschritt an gleicher Stelle bleiben oder sich in die jeweils angrenzenden Zellen des quadratischen Gitters bewegen.

Der Algorithmus lässt sich in zwei Teilschritte, den Strömungsschritt und den Kollisionsschritt, einteilen. Im Kollisionsschritt kollidieren die Pfeile von den Nachbarzellen in der zu berechnenden Zelle. Je nachdem, welche Simulation angestrebt wird, können Kollisionstherme mit Gleichgewichtsfunktionen zu den Pfeilen addiert werden, welche von der Viskosität des Fluids abhängen.

Diese Kollisionstheoreme haben wir jedoch im ersten Schritt unserer Programmierung noch nicht angewendet und durch selbst entwickelte Funktionen ersetzt, um das Programm möglichst einfach zu halten.

Im Strömungsschritt werden alle acht Richtungspfeile gemäß ihrer Richtung zum nächsten Gitterpunkt weitergeleitet. Die Nullpfeile werden nicht verändert, sodass die Teilchen des Nullpfeils in der Zelle verbleiben. Die so verschobenen Pfeile bilden wieder die Ausgangssituation für den nächsten Kollisionsschritt.

### 3.1.2 Umsetzung des Simulationsverfahrens im 2D-Raum

Das hergeleitete Simulationsverfahren haben wir mit der Programmiersprache Python umgesetzt. Dieses Programm basiert auf einer endlichen Schleife mit den Funktionen *Update\_Fluids*, *Reibung*, *Eigenschwingung*, *Show\_Fluids* und *Hindernis*.

In der ersten Funktion *UpdateFluids* werden die Pfeile in Bewegungsrichtung zu den nächsten Zellen verschoben und in *Show\_Fluids* in Abhängigkeit von der Pfeildichte der jeweiligen Zelle auf der Darstellungsfläche grau dargestellt. In der Funktion *Reibung* werden die Pfeilgrößen, der Realität entsprechend, um eine Konstante reduziert und eine Zelle zurückgesetzt, damit sich das Fluid nicht unendlich weit im Raum bewegt. Bei der Funktion *Eigenschwingung* wird die Ausbreitung der Teilchen, unabhängig von *Update\_Fluids* simuliert, indem jeder Pfeil einer Zelle auf die Pfeile mit derselben Pfeilrichtung der umliegenden Zellen aufgeteilt wird. Die Reflektion der Teilchen am Hindernis wird durch die Funktion *Hindernis* bearbeitet. Dabei wird das Hindernis durch die Randzellen des Hindernisses dargestellt. Diese sind in der Abbildung 4 rot gefärbt.

Jeder Hinderniszelle wird ein Reflexionswinkel zwischen  $0^\circ$  und  $360^\circ$  zugeordnet, welcher die Reflexion des Fluids an dieser Zelle beschreibt. Falls eine Hinderniszelle mehrere, verschiedene Reflexionswinkel haben soll, kann jede Quadratseite zusätzlich als Reflexionsseite festgelegt werden, an der das Fluid im  $90^\circ$  Winkel reflektiert wird. Weiterhin gilt, dass das Hindernis durchgehend mit Hinderniszellen abgebildet sein muss, sodass bei einer Schrägenicht die Ecken die Verbindung zur nächsten Hinderniszelle sind, sondern die Kanten einer zwischengeschobenen Hindernisszelle, in Abbildung 4 blau gefärbt. Damit wird jede Schräge durch eine Treppenform dargestellt und außerdem wird garantiert, dass durch die Funktion *Update\_Fluids* keine Teilchen durch die Schrägenicht in das Hindernisinnere gelangen können. Jedoch kann das Fluid bei manchen Reflexionswinkeln trotzdem in das Hindernisinnere hinein reflektiert werden. Ist dies der Fall, bewegt sich das Fluid durch die Zellen des Hindernisinneren hindurch, in Abbildung 4 grün gefärbt, und wird am Hindernisende ohne Reflektion wieder nach außen in die Ebene gelassen.

Weiterhin lassen sich im Programm die Parameter Viskosität, Strömungswiderstand, Ausbreitungsgeschwindigkeit und Größe der Pfeilrichtungen festlegen.

Um die Teilchenbewegung besser nachvollziehen zu können, haben wir in Abbildung 4 links (600x600) eine Teilchenwand, welche 400 Zellen hoch und 2 Zellen breit ist, erzeugt.

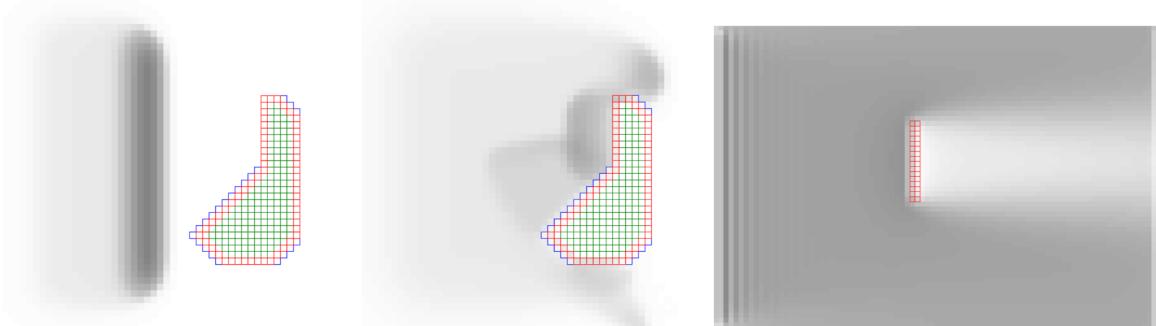


Abbildung 4: Grafische Ausgabe unserer Strömungssimulation im zweidimensionalen Raum mit verschiedenen Fluidquellen [61]

In der Realität ist bei einer Strömungssimulation jedoch der ganze Raum mit strömenden Teilchen gefüllt. Diese Simulation ist in Abbildung 4 rechts (900x600) mit einer einfachen Wand als Hindernis abgebildet. Dabei nimmt die Fluidquelle den ganzen Raum ein.

### 3.1.3 Umsetzung des Simulationsverfahrens im 3D-Raum

Im zweidimensionalen Modell wurden 8 Richtungspfeile verwendet. Im dreidimensionalen Raum arbeiten wir hingegen mit 26 Richtungspfeilen. Die Programmstruktur ähnelt dem 2D-Modell.

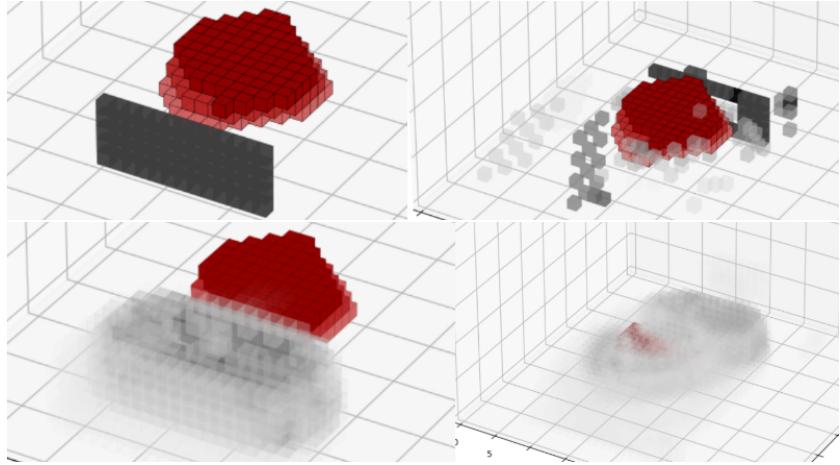
Nur die Winkeleingabe der einzelnen Hinderniszellen unterscheidet sich. Für jede Hinderniszelle muss einmal ein Reflexionswinkel hinterlegt werden, welcher die Reflektion in der zweidimensionalen Ebene bestimmt und ein weiterer Reflexionswinkel, der die Reflexion senkrecht zur zweidimensionalen Ebene festlegt. Dadurch lässt sich jedes Hindernis eindeutig darstellen.

Für die Darstellung wird nun anstelle von Tkinter das Modul Poly3DCollection von Matplotlib verwendet.

Die Abbildung 5 oben zeigt die Reflexion an einem beliebigen Körper ohne die Funktionen *Eigenschwingung* und *Reibung*. Bezieht man die Reibung und Eigenschwingung mit ein, ergibt sich das Resultat in Abbildung 5 unten.

Abbildung 5: Die grafische Ausgabe unserer Strömungssimulation im dreidimensionalen Raum mit verschiedenen Strömungsfunktionen [61].

Leider mussten wir uns am Ende dieser beiden Programmumsetzungen eingestehen, dass diese zu langsam und nicht für Darstellungen in Echtzeit geeignet sind. Jeder Berechnungsschritt benötigte im 3D-Raum 2 Sekunden bei der oben genannten Auflösung.



Um damit eine flüssige Simulation darstellen zu können, hätte man von jedem Berechnungsschritt ein Bild machen und dieses dann zu einem stark beschleunigten Video zusammenschneiden müssen. Neben dem Zeitproblem ergaben sich noch andere Fehler in der Simulation, welche nicht zu vernachlässigen waren. Die Fluidansammlungen vor den Hindernissen sind mit jedem Zeitschritt immer größer geworden und haben sich nur geringfügig auf ein stabiles Maß eingependelt. Außerdem sollte bei einem inkompressiblem Gas die Dichte in jeder Zelle gleich sein. Auch diese Voraussetzung war bei unseren Simulationen nicht erfüllt.

### 3.1.4 Optimierung der Strömungssimulation durch Kollisionstherme

Unser erster Versuch, die Kollisionstherme durch die vereinfachten Funktionen *Reibung* und *Eigenschwingung* zu ersetzen, war abschließend nicht erfolgreich und wir mussten uns genauer mit der Boltzmann-Gleichung auseinandersetzen.

Zuerst haben wir jedoch den Strömungsschritt in unserem Programm optimiert und von großen Listen auf Array Systeme von Numpy umgestellt, um den Strömungsschritt effizienter zu gestalten. Außerdem haben wir die sehr zeitintensive Reflexionswinkelberechnung gekürzt. Jetzt werden die Teilchen in Richtung ihrer ursprünglichen Position reflektiert.

Die größte Veränderung war letztendlich jedoch die Verwendung der Kollisionstherme und der damit einhergehenden Gleichgewichtsfunktion im Kollisionsschritt, welche die Funktionen *Reibung* und *Eigenschwingung* effizienter und realistischer ersetzen. Diese ermöglichen, dass jede Zelle die gleiche Dichte an Teilchen hat. Dadurch wird eine Strömungsanimation auf Basis der Teilchengeschwindigkeiten möglich. Diese sollen im Folgenden näher erläutert werden.

Die Lattice-Boltzmann-Methode basiert auf einer Diskretisierung der BGK-Gleichung<sup>3</sup>, welche eine Vereinfachung der Boltzmann-Gleichung ist. Gelöst wird die Gleichung (18): [16]

$$N_i(t+1, x + c_i) = (1 - \omega)N_i(t, x) + \omega N_{ie}(t, x) \quad (18)$$

Daraus ergibt sich die Boltzman-Maxwell Verteilung (19): [14, 16]

$$f(v)dv = \left(\frac{m}{2\pi kT}\right)^{\frac{3}{2}} 4\pi v^2 e^{-\frac{mv^2}{2kT}} dv \quad (19)$$

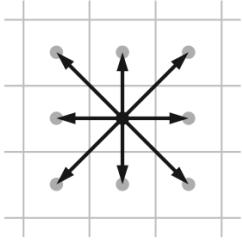
Aus der Taylor-Entwicklung der Maxwell Verteilung erhält man die für die Lattice-Boltzmann-Methode benötigte Gleichgewichtsverteilung (20): [16]

$$D(\vec{v}) \longrightarrow \omega_i \left[ 1 + \frac{3\vec{e}_i \cdot \vec{u}}{c} + \frac{9}{2} \left( \frac{\vec{e}_i \cdot \vec{u}}{c} \right)^2 - \frac{3}{2} \frac{|\vec{u}|^2}{c^2} \right] \quad (20)$$

Wobei  $c=1$  ist und  $\omega_i$  für jeden Richtungspfeil einen unterschiedlichen Wert aufweist. Die unterschiedlichen Werte für  $\omega_i$  resultieren daraus, dass die Richtungspfeile, welche horizontal oder senkrecht verlaufen, einen kleineren Weg als die schräg verlaufenden Pfeile pro Strömungsschritt zurücklegen. Pro Berechnungsschritt würden sich also manche Pfeile (Vergleich Abbildung 6) schneller bewegen. Da das der Realität jedoch nicht entspricht, müssen für alle Pfeile die folgenden Gewichtungen eingeführt werden (Vergleich Formeln (21) bis (32)): [16]

---

<sup>3</sup>Bhatnagar–Gross–Krook - Gleichung



$$\begin{aligned} \vec{e}_0 &= 0 & (21) & \vec{e}_5 = (1, 1) & (26) & \omega_0 = \frac{4}{9} & (30) \\ \vec{e}_1 &= (1, 0) & (22) & \vec{e}_6 = (-1, 1) & (27) & \omega_1 = \omega_2 = \omega_3 = \omega_4 = \frac{1}{9} & (31) \\ \vec{e}_2 &= (0, 1) & (23) & \vec{e}_7 = (-1, -1) & (28) & \omega_5 = \omega_6 = \omega_7 = \omega_8 = \frac{1}{36} & (32) \\ \vec{e}_3 &= (-1, 0) & (24) & \vec{e}_8 = (1, -1) & (29) \\ \vec{e}_4 &= (0, -1) & (25) \end{aligned}$$

Abbildung 6: Darstellung der Pfeilrichtungen mit zugehörigen Gewichtungen [57]

Letztendlich basiert die Gleichgewichtsfunktion auf den Gesetzen der Fluiddynamik und gibt das Verhalten der strömenden Teilchen eines idealen, inkompressiblen Gases an. Vereinfacht kann man dementsprechend sagen, dass die Gleichgewichtsfunktion im Kollisionsschritt die Aufgabe hat, mehrere Zellen mit unterschiedlichen Teilchengeschwindigkeiten bei gleich großen Dichten, auszugleichen. Um abschließend die endgültige, in unserem Programm verwendete Gleichgewichtsfunktion zu erhalten, muss die Summe (Dichte)  $\rho$  der Geschwindigkeiten einer Zelle auf die jeweilige Gleichgewichtsfunktion multipliziert werden. Damit erhalten wir den Ausdruck (33): [16]

$$n_i^{eq} = \rho \omega_i \left[ 1 + 3 \vec{e} \cdot \vec{u} + \frac{9}{2} (\vec{e} \cdot \vec{u})^2 - \frac{3}{2} |\vec{u}|^2 \right] \quad (33)$$

Um nun aus der Gleichgewichtsfunktion die neuen Geschwindigkeiten jeder Zelle zu erhalten, muss für jede Geschwindigkeit einer Zelle der folgende Term angewendet werden (34): [16]

$$n_i^{new} = n_i^{old} + \omega (n_i^{eq} + n_i^{old}) \quad (34)$$

Wobei hier  $\omega$  in der Theorie einen Wert zwischen 0 und 2 haben kann und im Wesentlichen die Viskosität des Fluids widerspiegelt. So berechnet sich die Viskosität  $\nu$  wie folgt (35): [16]

$$\nu = \frac{1}{3} \left( \frac{1}{\omega} - \frac{1}{2} \right) \quad (35)$$

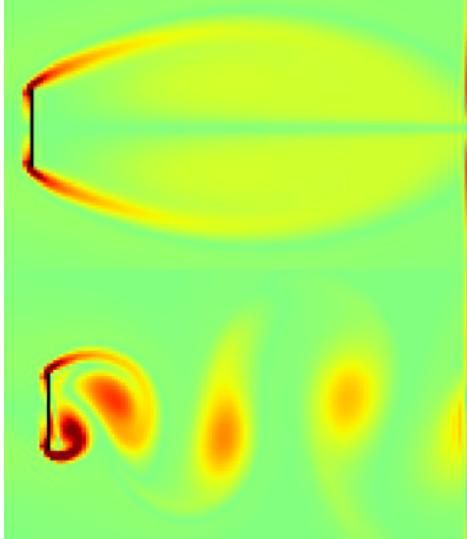


Abbildung 7: Aktualisierte zweidimensionale Strömungssimulation für laminare und turbulente Strömung [61]

In Abhängigkeit von der Fließgeschwindigkeit  $u_0$  des Fluids treten jedoch schon bei Viskositätswerten von über 1.96 und unter 0.25 Instabilitäten der Simulationen auf.

Indem wir den Kollisionsschritt sowie den Strömungsschritt optimiert haben, konnten wir unser Programm im zweidimensionalen Bereich ca. um den Faktor 100 beschleunigen. In Abbildung 7 sind 2 Simulationen von einer einfachen Wand als Hindernis mit dem neuen Programm zu sehen.

Deutlich erkennbar sind die Wirbelschleppen in der unteren Simulation in Abbildung 7 hinter dem Hindernis. Sie sind ein wichtiges Merkmal einer realitätsgtreuen Simulation. Denn in jedem Windkanal treten unregelmäßige Wirbelschleppen auf. Dies bestätigt auch die Korrektheit der zugrunde liegenden fluiddynamischen Modelle und der Gleichgewichtsfunktion. Denn mit dem alten 2D-Programm war es uns nicht möglich, Wirbelschleppen hinter dem Hindernis zu erzeugen. Trotzdem lassen sich die Wirbelschleppen nur unter bestimmten Parametereinstellungen erzeugen. Demnach muss die Viskosität im Verhältnis zur Geschwindigkeit sehr klein sein und damit der Faktor  $w$  sehr groß. Dieser Zusammenhang geht direkt aus der physikalischen Strömungslehre mit der Reynolds-Zahl hervor. Die Reynoldszahl ergibt sich aus (36): [32]

$$Re = \frac{\nu_m \cdot d}{\nu} \quad (36)$$

Je größer die Geschwindigkeit und je kleiner die Viskosität ist, desto größer wird die Reynoldszahl. Der Zusammenhang zwischen Viskosität und Geschwindigkeit ist in Abbildung 30 im Anhang dargestellt. Übersteigt die Reynoldszahl einen kritischen Wert, so verfällt die laminare Strömung in eine turbulente Strömung. Dabei ist in Experimenten die Bestimmung der kritischen Reynoldszahl auf  $2040 \pm 10$  gelungen. Eine Strömung ist laminar, wenn Verwirbelungen und andere Gleichgewichtsstörungen durch äußere Anregungen, wie zum Beispiel ein Hindernis, wieder zerfallen. Je kleiner die Reynoldszahl ist, desto schneller lösen sich die Verwirbelungen auf. Zerfallen die Verwirbelungen jedoch nicht mehr oder werden sogar noch stärker, handelt es sich um eine turbulente Strömung.

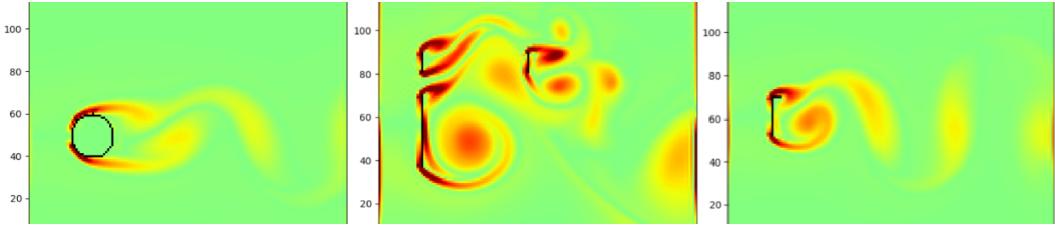


Abbildung 8: Darstellung turbulenter Strömung in unserem Simulationsprogramm [61]

In unseren Störungssimulationen können wir Wirbelschleppen erzeugen (Vergleich Abbildungen 7 und 8) und übersteigen damit die kritische Reynoldszahl. Eine äußere Anregung und Gleichgewichtsstörung benötigen wir jedoch trotzdem, um Verwirbelungen und letztendlich Wirbelschleppen zu erzeugen. Diese erzeugen wir, indem wir entweder asymmetrische Hindernisse oder Hinderniskombinationen erstellen, welche im mittleren und rechten Teil von Abbildung 8 zu sehen sind, oder die Hindernisreflektion einer Kante einer Zelle auslassen, was im linken Teil von Abbildung 8 dargestellt wird.

### Die Kollisionstherme im dreidimensionalem Raum

Natürlich haben wir die neuen Kollisions- und Strömungsschritte auch in den dreidimensionalen Bereich umgesetzt. Dabei mussten wir jedoch zuerst überprüfen, welche Anzahl an Pfeilen für unsere Simulationen am besten geeignet ist. Denn im dreidimensionalen Raum wird bei der Latice-Boltzmann-Methode zwischen 15-(D3Q15), 19-(D3Q19) und 27-(D3Q27) Pfeilen unterschieden. Wir probierten die letzten beiden Modelle und konnten keinen signifikanten Unterschied in den Simulationen feststellen. Also entschieden wir uns zu Gunsten der Laufzeit für das D3Q19 Modell. Hierbei muss jedoch beachtet werden, dass die Gewichtungen der einzelnen Pfeile für eine stabile Simulation nochmals geändert werden müssen. Die neuen Gewichtungen sind in Tabelle 2 gezeigt.

Model	$W_0$	$W_1$	$W_2$	$W_3$
D1Q3	2/3	1/6	0	0
D2Q9	4/9	1/9	1/36	0
D3Q15	2/9	1/9	0	1/72
D3Q19	1/3	1/18	1/36	0
D3Q25	1/3	1/36	0	0

Tabelle 2: Gewichtungsfaktoren der unterschiedlichen Simulationsmodelle der LBM-Methode [17]

Mit dem neuen Programm für 3D-Simulationen können wir sogar einen Geschwindigkeitsanstieg um den Faktor 500 im Vergleich zu dem alten Programm erreichen.

Ein Beispiel einer dreidimensionalen Strömungssimulation mit Wirbelschleppen können sie in den Abbildungen 9 und 10 nachvollziehen.

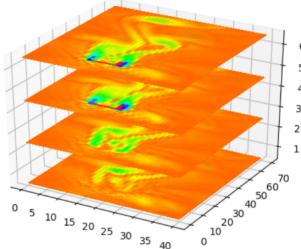


Abbildung 9: Dreidimensionale Strömungssimulation mit horizontalen Wirbelschleppen [61]

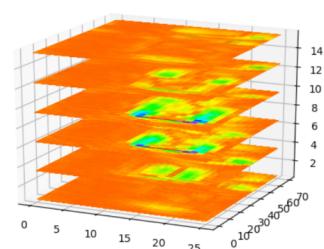


Abbildung 10: Dreidimensionale Strömungssimulation mit vertikalen Wirbelschleppen [61]

Zusammenfassend sind wir zufrieden mit den resultierenden Programmen und können uns damit dem Berechnen der Cw-Werte von Hindernissen widmen.

## 3.2 Virtuelle Körper im Programm

### 3.2.1 Vorbeachtung zum Umgang mit Körpern im Programm

Da wir mit Hilfe unseres Programms 3D Simulationen durchführen wollen, brauchen wir eine Möglichkeit mit Körpern umzugehen. Umgehen bedeutet in diesem Fall:

- (i) sie eindeutig zu definieren und zu speichern sowie
- (ii) an ihnen Berechnungen durchzuführen.

Da Punkt (i) nicht den Kernteil unserer Arbeit füllen soll, haben wir uns entschieden, bereits vorhandene Software zu nutzen, um Körper in digitaler Form zu erstellen.

### 3.2.2 Importieren von Körpern und Umgang mit Dateiformaten

In der Grafik oder in 3D-Bearbeitungsprogrammen werden Körper mittels der Polygon-Geometrie dargestellt. Ein Körper ist damit ein Polygon-Verbund und besteht aus ebenen N-Ecken. Kurven und nicht ebene Oberflächen, wie sie in der Freikörpergeometrie vorkommen, können nur approximiert werden. Diese Approximation ist ausreichend, da wir die geometrischen Körper später in boolsche Körper umwandeln. Sie werden in ein 3D Raster überführt, wobei jede Zelle des Raumes den boolschen Wert speichert, der aussagt, ob sie Teil des Körpers ist.

Geometrische Körper können wir mit Hilfe eines 3D-Bearbeitungsprogramms erstellen. Die entstehenden Daten zu den Polygonen des Polygon-Verbundes lassen sich im Anschluss exportieren. Dafür stehen viele verschiedene Dateiformate zur Verfügung. Um ein solches Dateiformat mit Python auslesen zu können, muss es den ASCII Code verwenden. Ein entsprechendes Format ist das *.obj*. Dieses ist offen und kann ohne rechtliche Einschränkungen genutzt werden.<sup>4</sup>

Ein Polygon besteht aus Kanten und Eckpunkten, die seine Fläche bilden. Die Kanten und die Fläche benennen wir und tragen sie in Listen ein, die die Daten zu allen Polygonen speichern. Die Eckpunkte interpretieren wir als Ortsvektoren. Diese speichern wir mittels des Numpy Moduls als Array. Wir ziehen diese Methode einfachen Tupeln vor, da wir so später auf bestimmte Rechenoperationen des Moduls zurückgreifen können.

Zum jetzigen Zeitpunkt sind wir in der Lage, den gesamten Körper zu manipulieren. Wichtig ist, zum Beispiel seine Ausrichtung im Raum zu verändern, um später Anströmungen aus bestimmten Richtungen vornehmen zu können. Haben wir alle nötigen Anpassungen vorgenommen, müssen wir den geometrischen Körper in einen boolschen Körper umwandeln. Auf diesen Schritt gehen wir im folgenden Kapitel ein.

### 3.2.3 Umwandlung geometrischer Körper in boolsche Körper

Bei der boolschen Definition<sup>5</sup> wird der Raum in diskrete Zellen eingeteilt, die die Information enthalten, ob sie Teil des Körpers sind. Mit der Einteilung des Raumes in Zellen verliert dieser jedoch auch sein Maß an Volumen und Ausdehnung. So ist die Zellengröße für diese Daten ausschlaggebend. Eine Zelle besitzt also gewissermaßen die Informationen ihrer Ausdehnung und Position im herkömmlichen, kartesischen Raum. Um ihr nun die ausschlaggebende Information zu geben, die sie als Teil des Körpers definiert, müssen wir überprüfen, ob die Zelle Teil des ursprünglichen Körpers ist. Das tun wir, indem wir ermitteln, ob ihr Mittelpunkt innerhalb des Körpervolumens liegt.

Somit benötigen wir eine Funktion, die berechnet, ob ein beliebiger Punkt innerhalb des kartesischen Koordinatensystems Element des Volumens eines Polygon-Verbundes ist.

Im zweidimensionalen Raum gibt es dafür die sogenannte Strahl-Methode.[24] Bei dieser Methode wird vom Testpunkt aus ein Strahl definiert. Tritt dieser Strahl durch das Kreuzen einer Kante eines Polygons in dieses Polygon ein, so muss er es durch Kreuzen einer weiteren Kante auch wieder verlassen. Bei konkaven Polygonen kann der Strahl mehrmals in den Körper eintreten und ihn somit auch mehrmals wieder verlassen. Anhand der Anzahl der Schnittpunkte des Strahls mit Kanten des Polygons kann ermittelt werden, ob der Testpunkt innerhalb oder außerhalb des Polygons liegt. Eine ungerade Anzahl an Schnittpunkten sagt aus, dass der Strahl im Polygon begonnen haben muss. Der Testpunkt liegt demnach innerhalb der Fläche des Polygons. Kein Schnittpunkt oder eine gerade Anzahl an Schnittpunkten besagt, dass der Testpunkt außerhalb des Polygons liegt.

In Abbildung 11 wird der Punkt H getestet. Der von ihm ausgehende Strahl kreuzt drei Kanten des Polygons: c, e und f. H liegt innerhalb der Fläche.

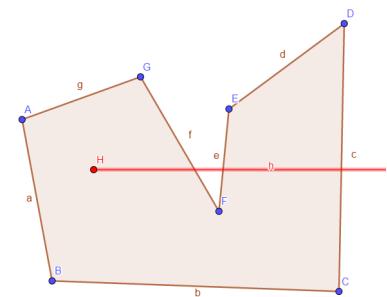


Abbildung 11: Darstellung der Strahl-Methode für den Testpunkt H [61]

Diese Methode haben wir auf den dreidimensionalen Raum übertragen. Dafür definieren wir den vom Testpunkt ausgehenden Strahl als Gerade. Die einzelnen Polygone des Verbundes ersetzen wir durch Ebenen. Wir ermitteln also die Anzahl der Schnittpunkte der Geraden mit den Ebenen. Zwischen einer Geraden und einer Ebene können folgende Lagebeziehungen auftreten:

1. Die Gerade und die Ebene sind echt parallel, es gibt einen Schnittpunkt im Unendlichen.
2. Die Gerade ist Teilmenge der Ebene, es gibt unendlich viele Schnittpunkte.
3. Die Gerade schneidet die Ebene in genau einem Punkt.

Im 1. Fall gibt es keinen definierbaren Schnittpunkt. In den Fällen 2. und 3. müssen wir jedoch mit der Methode für den zweidimensionalen Raum ermitteln, ob der Schnittpunkt innerhalb oder außerhalb des Polygons liegt, welches die Ebene darstellt. Für 2. wird dabei der Testpunkt selbst verwendet, während für 3. zunächst der Schnittpunkt aus

<sup>4</sup>Mehr Informationen zum Dateiformat *.obj* finden Sie im Anhang.

<sup>5</sup>Bilder von umgewandelten boolschen Körpern finden Sie auf Seite 24 im Anhang.

Gerade und Ebene berechnet werden muss. Dafür verwenden wir den Gauß-Algorithmus<sup>6</sup>. Wichtig ist auch, dass der Schnittpunkt in positiver Richtung der Gerade liegt, damit wir sie als Strahl ansehen können. Es gilt erneut, dass der Testpunkt bei gerader Schnittpunktzahl, oder fehlendem Schnittpunkt im Körper liegt und bei ungerader Schnittpunktzahl außerhalb.

Beim Verwenden dieser Methode, beziehungsweise Methoden, müssen wir darauf achten, dass an Eckpunkten Schnittpunkte mit mehreren Kanten oder Polygonen gleichzeitig auftreten können. Diese vielfachen Mehrungen müssen wir eliminieren. In diesen bestimmten Fällen treten zudem Rundungsfehler auf, was den Filterprozess erschwert, da es an einem Eckpunkt zu unterschiedlichen, sich in Kommastellen unterscheidenden, Schnittpunkten kommt.

Jetzt können wir die Mittelpunkte aller Zellen auf ihre Lage im Verhältnis zum Polygon-Verbund prüfen. Im Anschluss wird eine boolsche Matrix erstellt, die den geometrischen Körper ersetzt. Eine Zelle der Matrix spiegelt eine Zelle des vorherigen Raumes wieder. Am entstandenen boolschen Körper können wir nun Simulationen mit Hilfe der Lattice-Boltzmann Methode durchführen.

### 3.2.4 Direkter Import boolscher Körper

In 3D-Bearbeitungsprogrammen gibt es eine Funktion, die den erstellten Polygon-Verbund in alternative Formen umwandelt. Eine dieser Formen ist die Umwandlung zu einem Würfelgitter. Dieses Würfelgitter ist im Grunde ein neuer Polygon-Verbund. Wir können uns diesen besonderen Verbund jedoch zu Nutze machen. So stellen die Würfel bereits eine Aufteilung des Raumes in Zellen dar. Körper dieser Form können erneut als *.obj* exportiert werden und mittels einer alternativen Funktion ausgelesen werden. Diese erstellt keine Listen aus Kanten und Flächen sowie Ortsvektoren, sondern sofort die boolsche Matrix.

Dabei behandeln wir jeden Eckpunkt eines Würfels, aus denen der Körper besteht, als Mittelpunkt einer Zelle, da wir für sie bereits wissen, dass sie im Körper liegen. Nachteil dieser Methode ist, dass die durch das 3D-Bearbeitungsprogramm erstellte Würfelung um eine Würfelschicht in jede Richtung am Rand des Körpers ergänzt wird. Weiterhin kann so die Ausrichtung des Körpers im Raum nicht nachträglich verändert werden. Jede neue Anströmrichtung erfordert also den Import eines neu angepassten und veränderten Körpers.

### 3.2.5 Graphische Darstellung von Körpern

Für die graphische Oberfläche unseres Simulationsprogramms verwenden wir das Modul Tkinter. Dieses stellt eine Vielzahl von Widgets zur Verfügung und kann mit dem Modul Matplotlib verbunden werden. Letzteres dient der Visualisierung verschiedenster Daten mit Python und ist für wissenschaftliches Arbeiten konzipiert. Die mit Matplotlib erstellten Graphen lassen sich auf einem Tkinter Fenster einbinden.

Im Programm betrachten wir zwei verschiedene Arten von Körpern. Zum einen behandeln wir geometrische Körper, die aus Eckpunkten, Kanten und Flächen bestehen und zum anderen boolsche Körper. Die Körperarten benötigen unterschiedliche Darstellungen. Geometrische Körper visualisieren wir durch Polygon-Collections und ein Wireframe, also durch ihre Flächen und ihr Kantengitter. Die Flächen erhalten zudem einen Transparenzeffekt. Matplotlib stellt hierfür eine 3D-Projektion bereit.<sup>7</sup> Einen boolschen Körper kann man in 2D als ein Bild aus schwarzen und weißen Pixeln ansehen. In 3D sind die Pixel keine Quadrate mehr sondern Würfel. Matplotlib bietet hierfür den sogenannten Voxelplot. Um das Darstellungsvolumen zu minimieren, legen wir Ebenen in den Raum, die 2D Ausschnitte des Körpers als Bild darstellen.

## 3.3 Grundstruktur des Simulationsprogramms

Der Programmaufbau ist am Frontend, der graphischen Oberfläche, orientiert. Das Backend, sämtliche Berechnungs- und Simulationsfunktionen, sind demnach zu einem Objekt der graphischen Oberfläche zugehörig. Grundbaustein des Programms ist die Klasse *Programmoberfläche()*. Diese erbt aus der *Tk()* Klasse des von uns verwendeten *Tkinter* Moduls. Hier wird die graphische Oberfläche initialisiert und eine grundlegende Struktur gebildet. Diese beinhaltet neben dem *Notebook()*, das Untergrund für alle wichtigen Funktionen sein wird, auch die Objekte *Menu()*, *Treeview()* und *Progressbar()* aus *Tkinter*. Letztere dienen der Funktionalität zweitrangiger Programm-funktionen. Dem *Notebook* können nun *Frame()* Objekte hinzugefügt werden. Diese bilden den Untergrund für ein *Projekt()* Objekt, das aus dem *Frame()* erbt, um es zu initialisieren. Nun kommen wir zu den Kernfunktionen unseres Programms. Diese sind:

1. Die Eingabe eines Körpers.
2. Das Tätigen spezifischer Einstellungen.
3. Das Durchführen einer Strömungssimulation auf Basis der Einstellungen.
4. Das Auswerten des Projektes.

<sup>6</sup>Ein Strukturgramm des Gauß-Algorithmus kann im Anhang nachvollzogen werden.

<sup>7</sup>In einer vorherigen Version unseres Programms haben wir geometrische Körper auf eine zweidimensionale Ebene projiziert. Die entsprechende Umrechnung der Koordinaten finden Sie im Anhang.

Entsprechend obiger Erklärung orientieren sich alle vier Kernfunktionen an der graphischen Oberfläche. Im Umkehrschluss bedeutet dies, dass sie jeweils ein eigenes *Frame()* Objekt zur Verfügung gestellt bekommen. Sie alle werden demnach als Objekt umgesetzt, das vom *Frame()* erbt. So kann ihr graphisches Frontend initialisiert werden und gleichzeitig werden benötigte Funktionen als Methoden des Objektes umgesetzt, die das zugehörige Backend bilden. Da es für uns sinnvoll ist, dass die Funktionen zwei und drei für einen Körper mehrmals durchgeführt werden können, werden sie einem weiteren Objekt untergeordnet. Weiterhin verfügen wir über Objekte, die als Informations- oder Hilfsfenster dienen. Für den in einem Projekt bearbeiteten Körper wird zudem ein weiteres, eigenes Objekt angefertigt. Dieses speichert körperspezifische Attribute, wie die Koordinaten der Eckpunkte und Informationen über Kanten und Flächen. Zudem verfügt es über Methoden, die diese Attribute bearbeiten oder aus ihnen weitere Attribute ableiten können. Im Programm findet demnach ein stetiger Wechsel zwischen Methoden der graphischen Benutzeroberfläche und Methoden des Körperobjektes statt. Bei der Programmumsetzung legen wir also fest, zu welchem Objekt eine Funktion gehört. Bearbeitet sie beispielsweise die Eckpunkte des Körpers, so ist sie eine Methode des Körperobjektes und muss auch dort implementiert werden. Soll eine Funktion jedoch das Feedback auf die Betätigung eines Buttons auf der Programmoberfläche geben, so gehört sie zum jeweiligen Oberflächenobjekt. Das ist auch der Fall, wenn es der einzige Zweck dieser Methode ist, eine Funktion des Körperobjektes aufzurufen.

Der Benutzer gelangt somit zunächst auf die Programmoberfläche. Hier kann er ein neues Projekt starten und gelangt somit auf das Eingabefenster des Projektes. Dort kann er den Körper eingeben und weiter auf das Simulationsfenster gelangen. In diesem Fenster wird eine neue Simulation erstellt, bei der der Nutzer auf dem Einstellungsframe zunächst alle nötigen Einstellungen tätigt. Ist das erfolgt, kann er eine Simulation auf dem Simulationsframe durchführen. Jetzt kann der Nutzer weitere Simulationen mit jeweils neuen Einstellungen starten oder das Projekt auswerten. Im Auswertungsfenster werden schließlich alle, in den Simulationen gesammelten Daten, ausgewertet.

Weiterhin haben wir eine Datenbank erstellt, in der Projekte abgespeichert werden können<sup>8</sup>. Funktionen, die mathematischen Problemen gelten, haben wir in einem extra Dokument implementiert, da sie keine Methoden des Körpers sind und von mehreren Oberflächen aus aufgerufen werden müssen.

## 4 Anwendung unseres Simulationsprogramms

### 4.1 Die Verbindung zwischen Theorie und Realität

Wir können nun feststellen, eine realistische Strömungssimulation von Teilchen um Körper erzeugt zu haben. Doch auf das eigentliche Ziel der Seminarfacharbeit, die Berechnung der Cw-Werte von den umströmten Körpern, wurde bisher noch nicht genauer eingegangen. Allgemein ist der Cw-Wert ein variabler Wert in Abhängigkeit von der Reynoldszahl. Die Reynoldszahl ist dabei eine nach dem Physiker Oswald Reynold benannte dimensionslose Kennzahl und gibt das Verhältnis von Trägheits- und Zähigkeitskräften in einer Strömung an. (37)[13]

$$Re = \frac{\rho \cdot v \cdot d}{\mu} = \frac{v \cdot d}{\nu} \quad (37)$$

Die Abhängigkeit des Cw-Wertes von der Reynoldszahl wird in Abbildung 12 verdeutlicht.

Deutlich wird, dass der Cw-Wert mit der Reynoldszahl stark variieren kann. Er pendelt sich erst bei einer Reynoldszahl zwischen 1000 und 100.000 auf einen konstanten Wert ein.

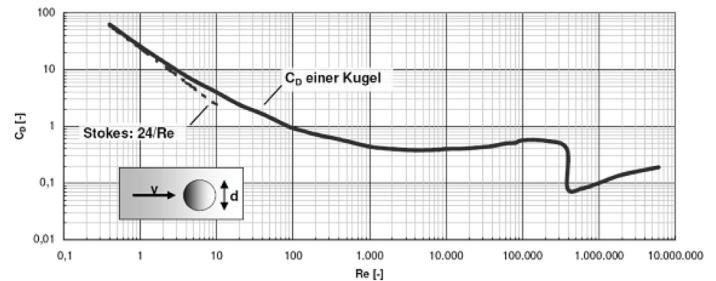


Abbildung 12: Cw-Wert einer Kugel in Abhängigkeit von der Reynoldszahl [62]

Dies ist der Bereich, indem heute die Cw-Werte von Körpern in Windkanälen berechnet werden.

Für diese Reynoldszahlen sind die Cw-Werte von gängigen Körpern bereits gemessen wurden und generell anerkannt. So hat die Kugel einen Cw-Wert von 0.47, der Würfel 1.05 und die Scheibe 1.1.

Um unsere Ergebnisse der späteren Cw-Wert-Berechnung einordnen zu können, müssen wir sicherstellen, dass unsere Simulationen mit Reynoldszahlen im genannten Bereich arbeiten.

Generell ist die Reynoldszahl durch die oben abgebildete Formel definiert und lässt sich in der Praxis ausrechnen. Jedoch haben wir in unserer Simulation keine Einheiten und können aus der im Programm varierbaren Viskosität und Geschwindigkeit nur Verhältnisse berechnen. Für die Bestimmung der verwendeten Reynoldszahlen müssen also andere Ansätze gewählt werden. Zum einen haben wir uns zu nutzen gemacht, dass laminare und turbulente Strömungen durch eine kritische Reynoldszahl unterschieden werden. Diese kritische Reynoldszahl variiert in Abhängigkeit vom Körper und Strömungsraum. Als Näherung lässt sich die kritische Reynoldszahl rund auf den Wert 2500 festlegen. Wir haben also aus dem Verhältnis von Viskosität und Geschwindigkeit im Programm eine theoretische Reynoldszahl berechnet, wobei die charakteristische Länge *d* vernachlässigt wird. Letztendlich konnten wir die

<sup>8</sup>Der Aufbau der Datenbank kann im Anhang nachvollzogen werden.

kritische Reynoldszahl auf die von uns berechnete theoretische Reynoldszahl von 18 bestimmen. Dementsprechend würden wir mit einer theoretischen Reynoldszahl von 18, welche in der Realität einer Reynoldszahl von 2500 entspricht, zwischen dem gewünschten Bereich von 1000 bis 100.000 liegen.

Für die Cw-Wert-Berechnung wäre dieses Verhältnis geeignet.

Um dieses Ergebnis zu überprüfen, haben wir den Cw-Wert einer Kugel in Abhängigkeit von mehreren theoretischen Reynoldszahlen ausgerechnet. Dabei ist der Graph in Abbildung 13 entstanden, welcher fast gleich zu den Graphen eines Würfels und eines Kegels<sup>a</sup> ist.

Vergleicht man das Diagramm aus Abbildung 13 mit dem Diagramm aus Abbildung 12, lässt sich der Abschnitt fast identisch im Bereich der reellen Reynoldsahl von 1000 bis 5000 einordnen. Damit entspricht im Diagramm (Abbildung 13) die theoretische Reynoldsahl von 20 wieder der reellen Reynoldsahl von 2500 und das Ergebnis von der kritischen Reynoldsahl ist bestätigt.

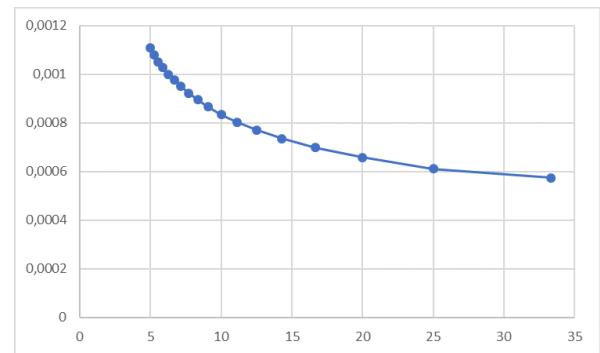


Abbildung 13: Cw-Wert einer Kugel in Abhängigkeit von der Reynoldsahl [61]

<sup>a</sup>Die entsprechenden Graphen finden Sie im Anhang auf Seite 24.

## 4.2 Optimierung der Cw-Wert-Berechnung

Oben ist bereits ein Diagramm mit Berechnungsergebnissen des Cw-Wertes dargestellt. Wie genau diese Berechnung funktioniert, wird im Folgenden erläutert. Der Cw-Wert ist allgemein definiert durch die Formel (38): [19]

$$c_w = \frac{F_w}{q \cdot A} = \frac{2 \cdot F_w}{\rho \cdot v^2 \cdot A} \quad (38)$$

Die Kraft  $F$ , welche auf den Körper in Strömungsrichtung wirkt, berechnen wir in unserer Simulation, indem wir alle Richtungspfeile von den umliegenden Gitterzellen des Körpers in Strömungsrichtung aufsummieren. Dabei nutzen wir das D3Q19 Lattice-Boltzmann-Modell. Die Dichte wird über die Summe aller Richtungspfeile jeder Zelle im gesamten Raum bestimmt und durch die Anzahl der Zellen des Raumes, abzüglich der des Körpers, geteilt. Weiterhin wird die Fläche über die Anzahl der direkt angestromten Würfel vom Körper ermittelt und die Geschwindigkeit ist im Programm einstellbar. Damit lassen sich alle benötigten Variablen für eine korrekte Cw-Wert-Berechnung mit der oben erwähnten Formel erschließen.

Anschließend kann man für jeden Körper den theoretischen Cw-Wert berechnen, dass Ergebnis mit den Werten aus der Realität vergleichen und die theoretischen Ergebnisse um einen allgemeinen Faktor  $\alpha$  anpassen, sodass sie den realen Cw-Werten möglichst gleich sind. Vorher muss jedoch kontrolliert werden, dass alle Cw-Wert-Berechnungen mit den verschiedenen Körpern unter den gleichen Bedingungen stattfinden. Denn nur so lässt sich ein allgemeiner Faktor  $\alpha$  festlegen, welcher für alle theoretischen Ergebnisse gilt. Um dies zu gewährleisten, muss die Reynoldsahl für jede Simulation gleich sein. Dies garantieren wir zum einen, indem wir das Verhältnis aus Viskosität und Geschwindigkeit im Programm konstant lassen. Zum anderen mussten wir aber auch während der Bestimmung der kritischen Reynoldsahl feststellen, dass die Reynoldsahl und damit der Cw-Wert unter anderem von der Skalierung und der Positionierung des Körpers im Raum abhängt. Dieser Zusammenhang lässt sich durch die Abhängigkeit der Reynoldsahl von der charakteristischen Länge  $d$  des Körpers erklären.

Denn je größer die Länge des Raumes in Strömungsrichtung ist, desto kleiner die kritische Reynoldsahl. Wobei sich die Reynoldsahl ab einer Raumlänge hinter dem Körper und in Strömungsrichtung, welche das sechsfache der dreidimensionalen Maximalausdehnung des Körpers entspricht, auf einen konstanten Wert eingependelt. Mit dem Faktor von 6 wurde die auch schon erwähnte kritische Reynoldsahl oben bestimmt. Der genannte Zusammenhang wird in Abbildung 14 noch einmal verdeutlicht. Die Maximalausdehnung ist dabei beschrieben durch den Abstand der minimalen und maximalen X, Y und Z-Koordinate zueinander.

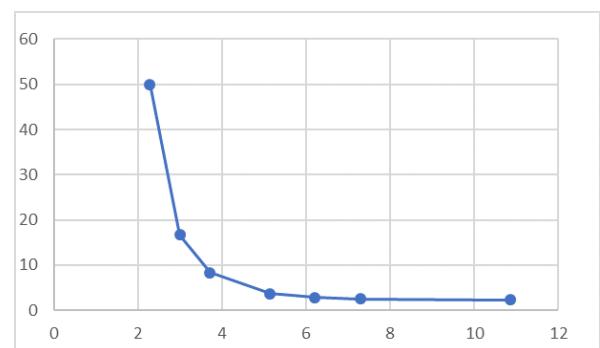


Abbildung 14: Abhängigkeit der Kritischen Reynoldsahl vom der Maximalausdehnung des Raumes [61]

Dementsprechend verwenden wir im Programm bei jedem Körper einen Faktor der Maximalausdehnung von 6, denn damit bleibt die Reynoldszahl auch bei unterschiedlichen Körperformen immer nahezu konstant.

Um die Cw-Wertberechnung weniger fehleranfällig zu machen, haben wir die Formel für den Cw-Wert auf  $c_W = \frac{F}{A}$  heruntergekürzt. Denn die einstellbare Reynoldszahl mit der Viskosität und Geschwindigkeit bleibt für jede Berechnung gleich. Auch für die Dichte haben wir durch Berechnungsergebnisse zwischen verschiedenen Körpern bewiesen, dass diese unabhängig vom Körper ist und nur durchschnittlich um 0.4% vom Mittelwert abweicht. Die geringe Abweichung lässt sich auf eine ungenaue Volumenberechnung der Körper zurückführen. Damit sind bis auf die Fläche  $A$  und die Kraft  $F$  alle Variablen der Cw-Wert Formel konstant und entfallen. Da in der Strömungssimulation mit höheren Reynoldszahlen Verwirbelungen auftreten, welche den Cw-Wert in Abhängigkeit vom Strömungsschritt annähernd periodisch zum Schwanken bringen, berechnen wir den resultierenden Cw-Wert aus dem Durchschnitt der Cw-Wert zwischen dem Strömungsschritt 1000 und 1400. Die Schwankungen sind in Abbildung 22 auf Seite 25 im Anhang abgebildet. Um zu bestimmen, welche Reynoldszahl die besten Cw-Wert Ergebnisse liefert, haben wir Berechnungen mit verschiedenen theoretischen Reynoldszahlen und zwölf einfachen Körpern, deren Cw-Werte sich in der Literatur finden, durchgeführt. Mit unserem Programm können wir theoretische Reynoldszahlen zwischen 5 und 50 abdecken, welche in der Realität Reynoldszahlen zwischen 700 und 7000 widerspiegeln. Dabei haben wir verschiedene Diagramme mit Cw-Werten erzeugt. Es hat sich herausgestellt, dass der allgemeine Faktor  $a$  zwischen dem theoretisch berechneten Cw-Wert und dem realen Cw-Wert eines Körpers nicht konstant ist, sondern eine lineare Funktion darstellt. (Vergleich Abbildung 15)

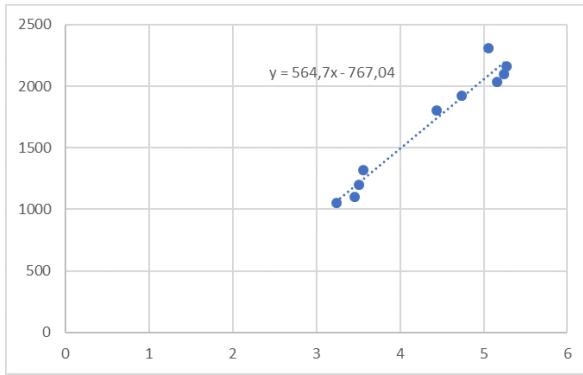


Abbildung 15: Faktor  $a$  in Abhängigkeit vom berechneten Cw-Wert [61]

Wir konnten die geringste Abweichung von dem Faktor  $a$  zur Geraden bei einer Würfelgenauigkeit von 14 mit einer theoretischen Reynoldszahl von 13 feststellen. Bei einer Würfelgenauigkeit von 20 befand sich das Minimum der Abweichung nach einer Berechnungsdauer von über 70 Stunden bei einer theoretischen Reynoldszahl von 9. Mit größeren Reynoldszahlen überschreiten wir die Grenze der kritischen Reynoldszahl 18 und die Verwirbelungen nehmen zu. Damit nehmen auch die Schwankungen der Cw-Werte zwischen den einzelnen Strömungsschritten zu und die Cw-Wert Ergebnisse werden ungenau. Bei kleineren theoretischen Reynoldszahlen treten ungenaue Cw-Werte auf, da man nicht mehr im Bereich der Reynoldszahl zwischen 1000 und 100.000 rechnet und damit die Cw-Wertergebnisse nicht mehr die bereits gemessenen Cw-Werte in der Literatur repräsentieren.

## 4.3 Ergebnisse der Cw-Wert-Berechnung

### 4.3.1 Ergebnisse für einfache Körper

Nachdem wir die Cw-Wert-Berechnung bestmöglich optimiert hatten, haben wir die Cw-Werte von 12 einfachen Körpern mit verschiedenen Würfelgenauigkeiten durchgerechnet. Dabei konnten wir mit einer Würfelgenauigkeit von 14 Würfeln eine durchschnittliche Abweichung von 4.4 % erzielen, was in Tabelle 3 noch einmal verdeutlicht wird. [16, 17, 12, 32]

Körper	berechneter Cw-Wert	realer Cw Wert
Kegel	0.344	0.340
Hohlkugel	0.411	0.380
Halbkugel	0.426	0.420
Kugel	0.443	0.470
Würfel (45 Grad gedreht)	0.773	0.800
Zylinder	0.905	0.910
Halbkugel entgegen	1.058	1.170
Würfel	1.108	1.050
Scheibe	1.149	1.100
Kegel entgegen	1.164	1.140

Tabelle 3: Berechnete Cw-Werte im Vergleich mit Werten aus der Literatur [61, 16, 17, 12, 32]

Bei einer Würfelgenauigkeit von 20 beträgt die durchschnittliche Abweichung 5.1%. Bei den Berechnungen wurden die einfachen Körper jeweils zentral von vorne angeströmt. Um die Abhängigkeit des Anströmwinkels vom Cw-Wert des Körpers darzustellen, haben wir im Anhang in den Bildern 23 und 24 eine Abbildung des Cw-Wertes von einem Würfel und einer Halbkugel in Abhängigkeit vom Anströmungswinkel erstellt. Auffallend ist, dass die theoretisch berechneten Cw-Werte bei extrem großen und kleinen Werten stark fehleranfällig sind. So hat real ein stromlinienförmiger Körper einen Cw-Wert von 0.04. Unser Programm berechnet dagegen einen Cw-Wert von 0.6, größer als der

Cw-Wert einer Kugel. Hier stößt die Simulation also an ihre Grenzen, welche in der Fehlerbetrachtung genauer erklärt werden. Eine ähnliche Grenze finden wir auch bei extrem großen Cw-Werten wie zum Beispiel dem einer, entgegen der Strömung ausgerichteten, halben Hohlkugel. Dieser beträgt normalerweise 1,4. Unsere Simulation berechnet hingegen einen Wert von 0,9.

#### 4.3.2 Ergebnisse des Lamborghinis und eingescannter Körper

Seit Beginn unserer Arbeit haben wir es uns zur Hauptaufgabe gemacht, den Cw-Wert eines Lamborghinis zu berechnen. Die *.obj* Datei des Lamborghinis haben wir kostenlos aus dem Internet bezogen. Mit Blender haben wir die Datei so bearbeitet, dass unser Programm sie würfeln kann. Leider konnten wir bei der Cw-Wert-Berechnung des Lamborghinis keine repräsentativen Ergebnisse im Vergleich zum realen Wert von 0.33 verzeichnen. Dies erklären wir damit, dass der Lamborghini eine kleine Anströmfläche im Vergleich zu der gesamten Oberfläche des Körpers hat. Darauf werden wir jedoch in der Fehlerbetrachtung noch einmal ausführlicher eingehen. Um unseren Lamborghini nicht gänzlich außer Acht zu lassen, haben wir weiterhin das von unserer Software berechnete Strömungsbild mit dem Strömungsbild eines Lamborghinis aus dem Windkanal verglichen. Die beiden Strömungsbilder sehen fast identisch aus und sprechen für die zugrunde liegende simulierte Strömung. Der Vergleich ist in Bild 27 im Anhang abgebildet. Damit wir auch noch weitere und vor allem komplexere Körper berechnen können, haben wir uns über das Schülerforschungszentrum in Erfurt einen mit Apple Tablets kompatiblen 3D-Scanner beschafft. Dieser scannt, die von uns gewünschten Körper ein und wandelt sie in kompatible *.obj* Dateien um. Die *.obj* Dateien können anschließend mit der Grafiksoftware Blender in Strömungsrichtung ausgerichtet werden und gegebenenfalls um fehlerhafte Teile korrigiert werden. Dabei sei angemerkt, dass der 3D-Scanner kaum Einschränkungen unterliegt. Es können sowohl große Objekte wie Schränke und Tische, als auch kleine Objekte, wie eine Federnmappe oder Computermaus eingescannt werden. Anschließend kann unser Programm den Cw-Wert berechnen. Jedoch steht dieses Mal kein realer Vergleichswert aus der Literatur zur Verfügung, denn wir haben unter anderem Körper wie einen Nussknacker oder einer Tasse verwendet. Diese wurden bisher nicht in einem Windkanal gemessen. Deswegen haben wir den Windkanal im Schülerforschungszentrum verwendet, um mit den eingescannten Körpern unsere eigenen Cw-Wertmessungen durchzuführen.

Um unsere Messungen später einordnen zu können, haben wir zuerst die im Windkanal verwendete Reynoldszahl bestimmt. Dabei konnten wir in Abhängigkeit der Größe des Körpers einen durchschnittlichen Bereich zwischen 20.000 und 50.000 ermitteln. Dieser liegt ebenfalls im Bereich der Reynoldszahlen mit einem konstanten Cw-Wert und damit können die Messungen vom Windkanal direkt mit unseren theoretischen Berechnungen verglichen werden. Die Ergebnisse haben wir in der Tabelle 4 im Anhang auf Seite 30 kurz zusammengefasst. Dort haben wir weiterhin unsere Messungen, die Funktionsweise des Windkanals und die Genauigkeit der Ergebnisse expliziter erläutert.

Für die komplexen Körper können wir eine durchschnittliche Abweichung zum gemessenen Cw-Wert von 11.4% festhalten. Wobei hier angemerkt sei, dass der Windkanal eine durchschnittliche Fehlerquote von 15% hat. Unsere Software arbeitet also, basierend auf den Abweichungen von einfachen Körpern mit 5%, genauer als der Windkanal. Wir können dementsprechend nur überprüfen, dass der von uns berechnete Cw-Wert grob im Bereich des vom Windkanal gemessenen Wertes liegt, plus minus 20% Abweichung. Dies ist bei jedem berechneten Cw-Wert erfüllt. Damit schlussfolgern wir, dass unsere Software auch für komplexe Körper geeignet ist, bei Beachtung der Einschränkungen in der Fehlerbetrachtung. Mit dem Windkanal ist es ebenfalls möglich, dass Strömungsbild um den Körper sichtbar zu machen. Dieses haben wir mit mehreren Kamerabildern für die Körper festgehalten. Anschließend haben wir die Bilder mit den Bildern unserer eigenen Simulation des jeweiligen Körpers verglichen. Die Strömungsbilderpaare eines Körpers sehen dabei annähernd identisch aus, was durch die Abbildungen 28 und 29 im Anhang verdeutlicht wird. Zusammen mit dem Strömungsbild des Lamborghinis unterstreicht dies die Genauigkeit und die Realitätsnähe der von uns entwickelten Simulation.

### 4.4 Fehlerbetrachtung für die von uns entwickelte Simulationsmethode

Wie in den Ergebnissen bei den einfachen Körpern bereits deutlich geworden ist, kommt unsere Cw-Wert-Berechnung bei extrem großen und kleinen Cw-Werten an ihre Grenzen. Jedoch sind auch grobe Fehler bei dem Cw-Wert des Lamborghinis aufgetreten. Wir haben die Fehlerursachen bei weiteren, selbst erstellten Körpern untersucht und konnten zwei signifikante Fehlerquellen ausmachen. Im Kapitel Optimierung Cw-Wert-Berechnung erwähnten wir bereits, dass die Reynoldszahl und damit der Cw-Wert von der Skalierung des Körpers im Strömungsraum abhängt. Ab einer Raumlänge, basierend auf dem sechsfachen der Maximalausdehnung des Körpers, hat sich die Reynoldszahl dabei auf ein konstantes Maß eingependelt. Bei Körpern, welche jedoch in einer der Dimensionsrichtungen senkrecht zur Strömungsrichtung sehr lang sind, scheint diese Näherung nicht mehr zu funktionieren. So hat eine quadratische Scheibe normalerweise einen Cw-Wert von 1.1. Dieser wird von unserer Software mit 1.15 berechnet. Ziehen wir die Scheibe jedoch senkrecht zur Strömungsrichtung Y, beispielsweise in X-Richtung, in die Länge, so haben wir schon bei einem Verhältnis von drei zwischen X-Länge und Z-Länge des Körpers einen doppelt so großen Cw-Wert. Dieser Zusammenhang wird noch einmal in Abbildung 26 im Anhang verdeutlicht. Die zweite Fehlerquelle geht aus dem Verhältnis zwischen Anströmfläche und gesamter Oberfläche des Körpers hervor. Bei der Ermittlung der auf den Körper wirkenden Kraft bestimmen wir standardmäßig die Differenz der fünf auf den Körper wirkenden Richtungspfeile

in X-Richtung. Dabei kann der zentrale Pfeil in X-Richtung nur auf die Anströmfläche wirken. Jedoch können die vier schrägen Pfeile in X-Richtung auf einer Körperfläche parallel zur Strömungsrichtung einwirken. Wird nun die Gesamtfläche des Körpers größer, aber die Anströmfläche bleibt konstant, können mehr schräge Pfeile auf den Körper einwirken. Damit wird die Endsumme der auf den Körper wirkenden Kraft größer, obwohl die Anströmfläche konstant bleibt. Letztendlich mündet dieser Prozess in einem vergrößerten Cw-Wert. Wie haben versucht, die Kraftberechnung nur durch die zentrale Kraft in X-Richtung durchzuführen, jedoch waren diese Ergebnisse noch weniger repräsentativ. Um diesen Zusammenhang darzustellen, haben wir eine Scheibe verwendet und diese in Y-Richtung kontinuierlich verlängert. Der Verlauf des Cw-Wertes ist in Abbildung 25 im Anhang grafisch dargestellt. Der Cw-Wert eines Würfels beträgt normalerweise 1.05. Verlängert man ihn um den Faktor 3.3 in Y-Richtung, sodass das Verhältnis von Anströmfläche zur Oberfläche 15 ist, beträgt der von unserem Programm berechnete Cw-Wert fast das Doppelte. Dadurch lässt sich letztendlich auch der von unserem Programm falsch berechnete Cw-Wert vom Lamborghini und dem Strömungskörper erklären. Der Lamborghini hat ein Verhältnis zwischen den Flächen von 16 und das Verhältnis zwischen X- und Z-Ausdehnung des Körpers beträgt 1.6. Damit das Programm jedoch den richtigen Cw-Wert berechnet, sollte, basierend auf den beiden Diagrammen, das Flächenverhältnis nicht größer als 11 und das Verhältnis zwischen X- und Z-Ausdehnung nicht größer als 1.8 sein.

## 4.5 Einordnung der Ergebnisse

Abschließend und zusammenfassend sind wir mit unseren Ergebnissen zufrieden. Mit unserem Programm haben wir eine realistische und anschauliche Strömungssimulation für Körper entwickelt, mit der das Strömungsverhalten eines Körpers benutzerfreundlich untersucht werden kann. Dabei steht unsere Animation der strömenden Fluide den Strömungsbildern im Windkanal nichts nach. Im Gegenteil, sie ist sogar noch anschaulicher und informativer, da sie mehreren Perspektiven und unterschiedliche Darstellungsformen umfasst. Im Windkanal ist die Strömungsdarstellung auf wenige farbige Strömungsstreifen, welche um den Körper laufen, beschränkt. Weiterhin können wir äußerst genaue Rechenergebnisse erzielen, welche im Bereich von Cw-Werten zwischen 0.25 und 1.2 eine Fehlerquote von nur 5% aufweisen. Außerhalb dieses Bereiches treten Ungenauigkeiten auf. Hier sollte die Software keine direkte Anwendung finden. Es müssen die genannten Bedingungen aus Kapitel 4.4 berücksichtigt werden. Gänzlich ersetzen kann unsere Software den Windkanal also noch nicht.

Weiterhin ist unsere Software praxistauglich. Mit einem 3D-Scanner kann man gewünschte Objekte einfach einscannen, mit Blender ausrichten und von unserer Software den Cw-Wert berechnen lassen. Existieren die Objekte nicht physisch, sondern nur auf dem Computer im *.obj* Format, können sie dennoch importiert werden. Letztendlich ist der Einsatz unserer Software um einiges effizienter als die Verwendung eines Windkanals. Zum einen ist unsere Software im genannten Bereich genauer, als der von uns getestete Windkanal im Schülerforschungszentrum, der zu den hochwertigsten Windkanälen gehört, die privat erhältlich sind. Zum anderen ist sie aber auch zeitsparender, denn das Einstellen und Berechnen des Cw-Wertes dauert maximal 30 Minuten. Dabei werden, basierend auf Erfahrungswerten, zum Einstellen des Körpers und Ausrichten mit Blender durchschnittlich 5 bis 10 Minuten benötigt. Für die Berechnung des Cw-Wertes sind weiterhin bei einer Würfelgenauigkeit von 14 rund 7 Minuten und bei einer Würfelgenauigkeit von 20 um die 20 Minuten Berechnungszeit einzuplanen. Einen Windkanal mit der richtigen Größe und der gewünschten Genauigkeit aufzusuchen kann dagegen Tage dauern und schnell Kosten im vierstelligen Bereich verursachen. Wir haben also eine Komplettlösung entwickelt, mit der jedermann einfach, zeitsparend und ohne größeren Aufwand den Cw-Wert eines Körpers bestimmen kann. Aufgrund der Flexibilität des 3D-Scanners, können dabei sowohl große als auch kleine Objekte berechnet werden, welche nicht in den Windkanal passen würden oder zu klein dafür sind. Damit ist unsere Komplettlösung für nahezu jedermann geeignet, vom einfachen Hobbybastler zu Hause bis hin zu dem Physiklehrer im Unterricht.

## 5 Reflexion unserer Arbeit

Ziel unserer Arbeit war es, eine Software zur Strömungssimulation von Fluiden um Körper zu entwickeln, welche die Cw-Wertberechnung von definierten Körpern ermöglicht. Dabei stellten wir uns die Frage, wie repräsentativ die theoretischen Ergebnisse sind und inwiefern es heutzutage noch nötig ist, für die Cw-Wertberechnung einen Windkanal zu verwenden.

Wir haben uns mit den Grundlagen der Fluidmechanik, speziell der Navier-Stokes-Gleichung, befasst und sind auf mathematische Modelle und ihre Komponenten sowie Operationen eingegangen. Auf Basis der Boltzmann-Gleichung und der Lattice-Boltzmann Methode haben wir ein Programm zur Strömungssimulation im zweidimensionalen und dreidimensionalen Raum erstellt. Dafür implementierten wir Methoden zum Einlesen und Würfeln von Körpern, die es uns ermöglichen mit offiziellen Dateiformaten zu arbeiten. Mit Hilfe dieses Programmkomplexes und einem umfangreichen GUI<sup>9</sup> lassen sich fluidmechanische Daten einer Strömung sammeln, anhand derer wir den Strömungswiderstandskoeffizienten eines Körpers nähern können. Zudem ist es möglich, die Strömung durch Diagramme und Animationen nachzuvollziehen. Basierend auf mehreren Körpern, von denen der Cw-Wert bereits gemessen wurde und anerkannt ist, haben wir die Cw-Wertberechnung optimiert und konnten letztendlich eine durchschnittliche Abweichung zu den realen Werten von rund 5% erzielen. Anschließend haben wir unsere Berechnungen auf eingescannte Körper angewendet und diese mit dem Windkanal verglichen. Wir konnten aufzeigen, dass unsere Simulation bessere Ergebnisse als der Windkanal erzielt, haben aber auch erläutert, wo unsere Simulation an ihre Grenzen stößt.

Zusammenfassend stellen wir fest, dass wir unsere gestellten Ziele erreicht haben und dass wir mit unseren Arbeitsergebnissen sehr zufrieden sind. Wir haben einen umfassenden Einblick in das Fachgebiet der Fluidodynamik erhalten und unsere Fähigkeiten im Bereich der Programmentwicklung vervielfacht. Das Arbeitsklima in unserer Gruppe war stets konstruktiv und motivierend. Unseren Zeitplan haben wir konsequent eingehalten und schließen unsere Seminarfacharbeit hiermit ab.

---

<sup>9</sup>Graphical-User-Interface

## 6 Anhang

### Bahn eines Volumenelementes für das Beispiel der Trichterströmung

Hier stellen wir die Bahngleichung eines Volumenelements für die Trichterströmung auf. Die Bahn des Elements verläuft parallel zum Geschwindigkeitsfeld. Somit ergibt sich folgender Zusammenhang für  $(\varrho_0, \varphi_0, z_0)$  im zylindrischen Koordinatensystem (39):

$$\begin{pmatrix} \varrho - \varrho_0 \\ \varphi - \varphi_0 \\ z - z_0 \end{pmatrix} = t \cdot \begin{pmatrix} \varrho_1 - \varrho_0 \\ 0 \\ z_1 - z_0 \end{pmatrix} \quad (39)$$

$(\varrho_0, \varphi_0, z_0)$  befindet sich hierbei am Boden des Trichters und  $(\varrho_1, \varphi_1, z_1)$  stellt den Ausgangspunkt der Bahn an der Spitze des Trichters dar. Für diese Größen gilt folgendes Verhältnis (40):

$$\frac{\varrho_1}{\varrho_0} = \frac{h}{H} \quad (40)$$

Damit die Bahn auf einer senkrechten Ebene verläuft muss zudem gelten ((41) und (42)):

$$\varphi_1 = \varphi_0 \quad (41)$$

$$\varphi = \varphi_0 \quad (42)$$

Schreibt man die erste Gleichung aus erhält man 6 Gleichungen, die die Bahn eines Volumenelementes durch den Punkt  $(\varrho_0, \varphi_0, z_0)$  beschreiben.

### Projektion dreidimensionaler Koordinaten in auf zwei Dimensionen

Die Projektion dreidimensionaler Körper auf eine zweidimensionale Fläche war Bestandteil unseres Programms, bevor wir uns dazu entschieden haben, *Matplotlib* als Darstellungsmodul zu verwenden. Für diesen Vorgang gibt es viele verschiedene Methoden, auf die wir hier nicht weiter eingehen werden. Für uns war es wichtig, dass die Projektion laufzeittechnisch effizient ist. Am einfachsten wäre es demnach, zum Beispiel alle Z-Koordinaten auf null zu setzen. Dies kann erreicht werden, indem man die Z-Koordinaten überschreibt. Diese Lösung ist jedoch nicht besonders elegant. Eine bessere Möglichkeit ist die Multiplikation einer Matrix mit dem Punkt. Diese Matrix kann über Variablen angepasst werden und somit auch zum automatischen Zoomen der projizierten Punkte verwendet werden. Indem man sie mit einem Faktor multipliziert, kann ihr Abstand voneinander verändert werden. Zudem ist die Projektion nicht nur allein auf die XY-Ebene beschränkt.

Um nun zum Beispiel einen dreidimensionalen Punkt auf der YZ-Ebene darzustellen, würde die Rechnung folgendermaßen aussehen (43):

$$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0 + Y + 0 \\ 0 + 0 + Z \end{pmatrix} = \begin{pmatrix} Y \\ Z \end{pmatrix} \quad (43)$$

## Aufbau von Wavefront 3D-Objektdateien

```

1 # Blender v2.79 (sub 0) OBJ File: ''
# www.blender.org
3 mtllib Cube_fine.mtl
o Cube_Cube.001
5 v -4.197928 -2.718778 5.815319
v -4.197928 7.281222 5.815319
7 v -4.197928 -2.718778 -4.184681
v -4.197928 7.281222 -4.184681
9 v 5.802072 -2.718778 5.815319
v 5.802072 7.281222 5.815319
11 v 5.802072 -2.718778 -4.184681
v 5.802072 7.281222 -4.184681
13 vn -1.0000 0.0000 0.0000
vn 0.0000 0.0000 -1.0000
15 vn 1.0000 0.0000 0.0000
vn 0.0000 0.0000 1.0000
17 vn 0.0000 -1.0000 0.0000
vn 0.0000 1.0000 0.0000
19 usemtl None
s off
21 f 1//1 2//1 4//1 3//1
f 3//2 4//2 8//2 7//2
23 f 7//3 8//3 6//3 5//3
f 5//4 6//4 2//4 1//4
25 f 3//5 7//5 5//5 1//5
f 8//6 4//6 2//6 6//6

```

Oben können sie das *.obj* Dokument eines Würfels sehen. Rechts sind die Listen zu den Daten des Polygon-Verbundes dargestellt, wie wir sie in unserem Programm verwenden. Die erste Liste enthält die einzelnen Eckpunkte des Körpers. Jeder Eckpunkt wird anhand eines Namens und seines Ortsvektor gespeichert. Den Ortsvektor speichern wir als Array-Objekt des Numpy Moduls. Die zweite Liste enthält die Kanten und die dritte Liste die Flächen des Körpers. Auch sie werden durch einen Namen gespeichert. Zudem beinhaltet sie eine Liste der Namen der Eckpunkte, die sie definieren.

```

[[ 'P0' , array([[ 0. ] ,
2 [ 0. ],
[10.]])],
4 [ 'P1' , array([[ 0. ] ,
[10.],
[10.]])],
6 [ 'P2' , array([[0. ] ,
8 [ 0.],
[0.]])],
10 [ 'P3' , array([[ 0. ] ,
[10.],
[ 0.]])],
12 [ 'P4' , array([[10. ] ,
14 [ 0.],
[10.]])],
16 [ 'P5' , array([[10. ] ,
18 [ 0.],
[10.]])],
20 [ 'P6' , array([[10. ] ,
22 [ 0.],
[ 0.]])],
24 [ 'P7' , array([[10. ] ,
26 [ 0.],
[ 0.]])]

[[ 'K0' , [ 'P2' , 'P0' ]],
28 [ 'K1' , [ 'P0' , 'P1' ]],
[ 'K2' , [ 'P1' , 'P3' ]],
[ 'K3' , [ 'P3' , 'P2' ]],
[ 'K4' , [ 'P6' , 'P2' ]],
30 [ 'K5' , [ 'P3' , 'P7' ]],
[ 'K6' , [ 'P7' , 'P6' ]],
32 [ 'K7' , [ 'P4' , 'P6' ]],
[ 'K8' , [ 'P7' , 'P5' ]],
34 [ 'K9' , [ 'P5' , 'P4' ]],
[ 'K10' , [ 'P0' , 'P4' ]],
36 [ 'K11' , [ 'P5' , 'P1' ]]

[[ 'F0' , [ 'P0' , 'P1' , 'P3' , 'P2' ]],
38 [ 'F1' , [ 'P2' , 'P3' , 'P7' , 'P6' ]],
[ 'F2' , [ 'P6' , 'P7' , 'P5' , 'P4' ]],
40 [ 'F3' , [ 'P4' , 'P5' , 'P1' , 'P0' ]],
[ 'F4' , [ 'P2' , 'P6' , 'P4' , 'P0' ]],
42 [ 'F5' , [ 'P7' , 'P3' , 'P1' , 'P5' ]]]

```

Wavefront 3D-Objektdateien werden dafür verwendet, geometrische Eigenschaften von Objekten zu speichern. Objekte können hierbei einzelne Körper aber auch Körperansammlungen sein. Gespeichert werden die Eckpunkte und die Flächen von Objekten. Zudem wird für jede Fläche eine Normale angegeben. Die Normale ist aus der Vektorrechnung bekannt und zeigt hier aus dem Objekt heraus. Die in der *.obj* Datei gespeicherten Informationen werden zeilenweise ausgelesen. Zeilen, die Daten zu Eckpunkten speichern sind mit dem Buchstaben *v* gekennzeichnet. Für einen Punkt wird der Ortsvektor in einem kartesischen Koordinatensystem angegeben. Normalen werden durch *vn* und die Flächen durch *f* markiert. Eine Fläche besteht aus mindestens 3 Punkten. Angegeben werden die Nummern der Punkte anhand ihrer Reihenfolge im Dokument. Zu jedem Punkt wird die Normale der Fläche angegeben, ebenfalls über ihre Nummer. Punkt- und Normalennummer sind durch einen doppelten Slash getrennt. Einzelne Angaben einer Zeile werden durch ein Leerzeichen getrennt. Im obigen Dokumentausschnitt sind die acht Eckpunkte eines Würfels der Kantenlänge 10 sowie die sechs Normalen und dazugehörigen Flächen gespeichert.

Es finden sich jedoch weitere Zeilen im Dokument, die für unsere Zwecke keine Rolle spielen. Die Zeilen eins und zwei beginnen mit einem Hashtag und sind somit Kommentare. Die vierte Zeile beginnt mit einem *o*. Dahinter findet sich der Name des gespeicherten Objektes. Dieser hat nichts mit dem Dateinamen zu tun. Er hat einen Nutzen, wenn mit mehreren Objekten gleichzeitig in einem 3D Bearbeitungsprogramm gearbeitet wird und dient somit der Benutzerfreundlichkeit. Hinter einem *g* befindet sich der Name einer Gruppierung, also Körperansammlung. Der Buchstabe *s*, im Dokument vor den Flächen zu finden, markiert die Glättung des Objektes. In diesem Fall ist sie deaktiviert. Sie spielt beispielsweise bei gekrümmten Körpern eine Rolle, wenn es darum geht, die durch Polygone approximierte Oberfläche zu 'glätten'. Zuletzt wird in der *.obj* Datei noch auf eine Materialbibliothek mit der Dateiendung *.mtl* verwiesen. Gekennzeichnet ist sie durch *mtllib*. Dahinter befindet sich der Pfad zur Datei. Im obigen Beispiel wird die vermerkte Datei jedoch durch die Zeile *usemtl None* als nicht benötigt gekennzeichnet. Weiterhin können im Dokument Texturen des Körpers, markiert durch den Zeilenanfang *vt*, angegeben werden.

Rechts im Dokumentausschnitt können sie die Übersetzung der Datei für unser Programm sehen. Dabei müssen wir beachten, dass wir beim Nummerieren der Punkte, Kanten und Flächen mit null starten. Zudem werden die Koor-

dinaten der Eckpunkte so angepasst, dass die kleinste Koordinate pro Dimension jeweils 0 ist. Dadurch kann das Rechnen mit rundungsanfälligen Gleitkommazahlen vermieden werden, was wiederum Rundungsfehlern durch das Programm vorbeugt. Zuletzt ist zu erkennen, dass die Eckpunkte einer Fläche bereits im *.obj* Dokument geordnet angegeben werden. Dadurch können wir die Kanten des Körpers bestimmen. Es ist lediglich darauf zu achten, Doppellungen zu vermeiden.

## Aufbau der Datenbank zum Speichern von Projekten unseres Simulationsprogramms

In Python lässt sich eine Datenbank mittels des Moduls *sqlite3* implementieren. Unsere Datenbank besteht aus zwei Haupttabellen und zugehörigen Nebentabellen. Die erste Haupttabelle heißt *IDs* und wird dafür verwendet, um die in der Datenbank verwendeten IDs zu speichern. Mit ihrer Hilfe können wir neue IDs automatisch erstellen und sichergestellt, dass es eine eindeutige Zuordnung zwischen den Tabellen gibt. Die Namen der Körper verwenden wir hierfür nicht, da an dieser Stelle Namensdopplungen auftreten können. In der zweiten Haupttabelle namens *Projekte* können nun im Programm bearbeitete Projekte gespeichert werden. Ein Eintrag besteht hierbei aus der erstellten Kennungs-ID, dem Namen des Projektes und des verwendeten Körpers sowie einem Pfad zur *.obj* Datei des Körpers. Die ID des Projektes dient weiterhin als Name der zum Projekt gehörigen Simulationstabelle. Hier werden alle Simulationen, welche im Zuge des Projektes durchgeführt wurden, abgespeichert. Ein Simulationseintrag erhält ebenfalls eine automatisch erstellte ID. Weiterhin sind der Simulationsname und die Simulationseinstellungen Bestandteil des Eintrags. Zuletzt fehlen die in der Simulation gesammelten Daten. Für sie wird eine Messwertetabelle mit der Simulations-ID als Namen angefertigt.

Da beim Öffnen eines Projektes die letzten Zustände aller Simulationen visualisiert werden sollen, müssen die zugehörigen Variablen ebenfalls gespeichert werden. Diese sind zum einen die Pfeildichte für jede Zelle und die Pfeilrichtungssumme pro Zelle für jede Dimension. Beim Speichern dieser Daten müssen wir beachten, dass es sich um eine 3D-Matrix handelt, die in eine zweidimensionale Tabelle eingetragen werden muss. Dank der *reshape()* Funktion des Numpy-Moduls, lässt sich diese reversibel in eine eindimensionale Matrix umwandeln. Diese kann jetzt in eine beliebig lange Spalte der Datenbanktabelle eingetragen werden. Die dafür angelegte Tabelle trägt analog zur Messwertetabelle die ID der zugehörigen Simulation als Namen, jedoch mit einem zusätzlichen Kürzel versehen.

## Grafiken und Visualisierungen

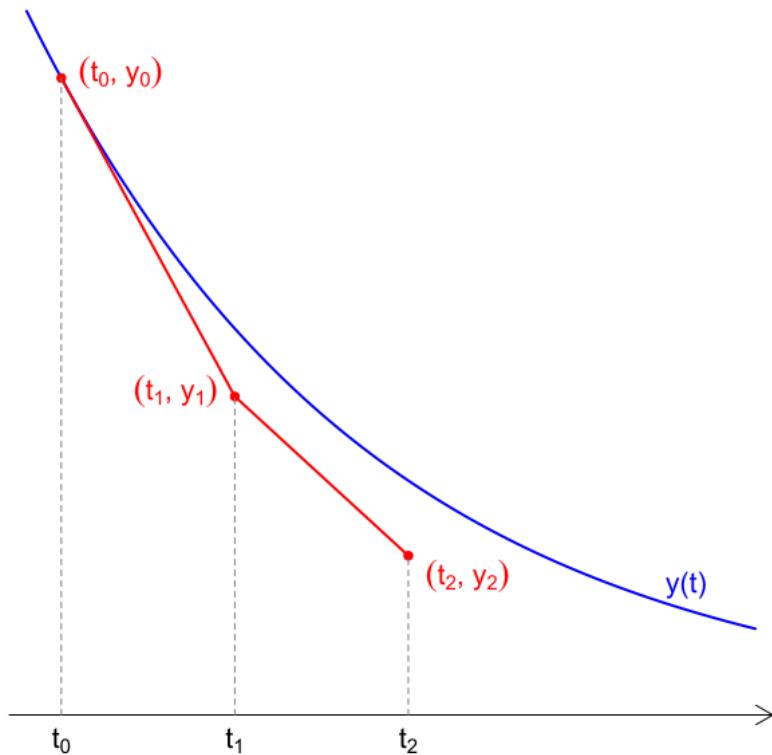


Abbildung 16: Modell des expliziten Euler-Verfahrens [26]

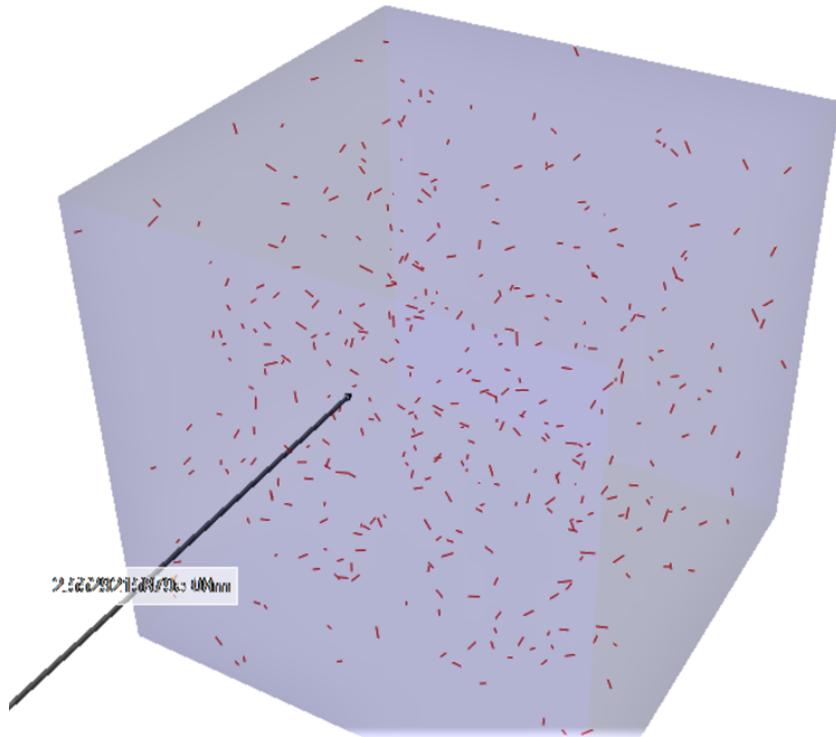


Abbildung 17: Darstellung der mikroskopischen Simulationsmethode [61]



Abbildung 18: Struktogramm zum Gaus Algorithmus [61]

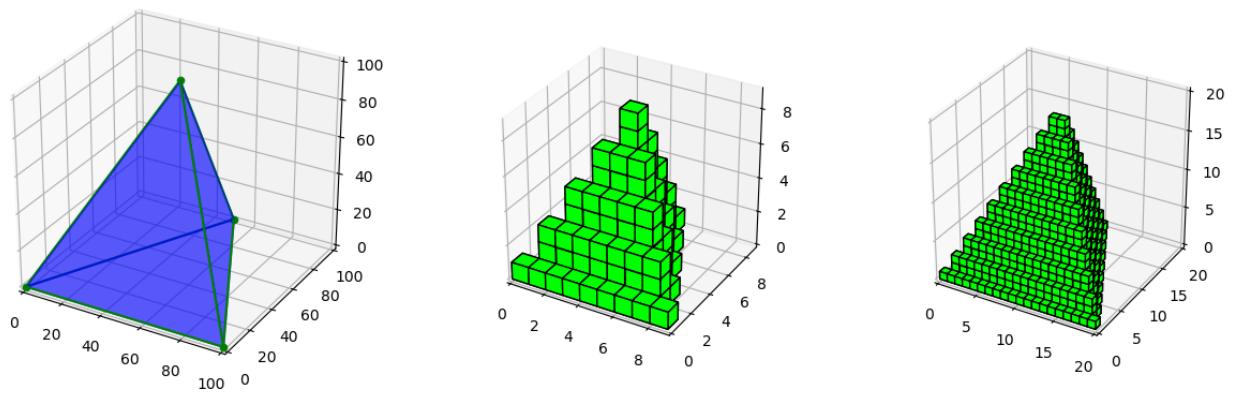


Abbildung 19: Umwandlung eines geometrischen Körpers in einen boolschen Körper mit unserem Programm [61]

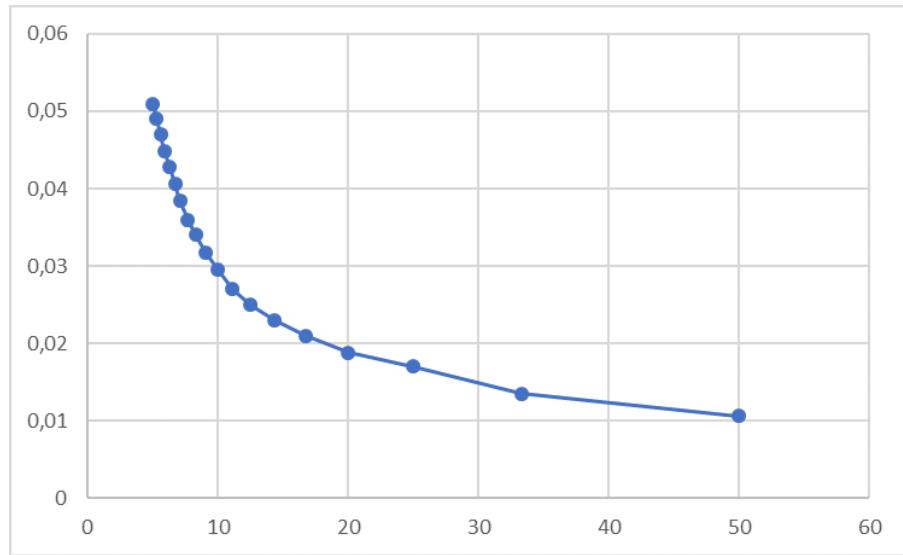


Abbildung 20: Ermittelter Cw-Wert eines Würfels in Abhängigkeit von der Reynoldszahl [61]

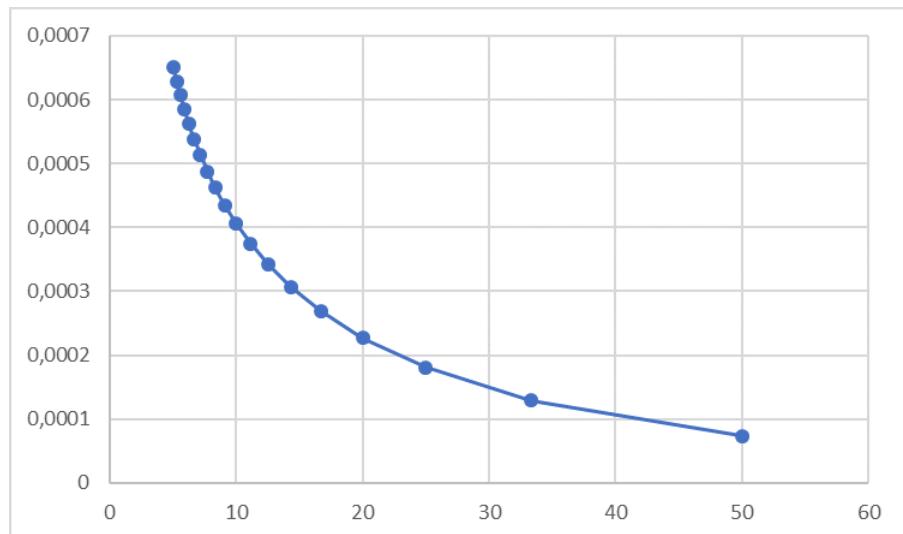


Abbildung 21: Ermittelter Cw-Wert eines Kegels in Abhängigkeit von der Reynoldszahl [61]

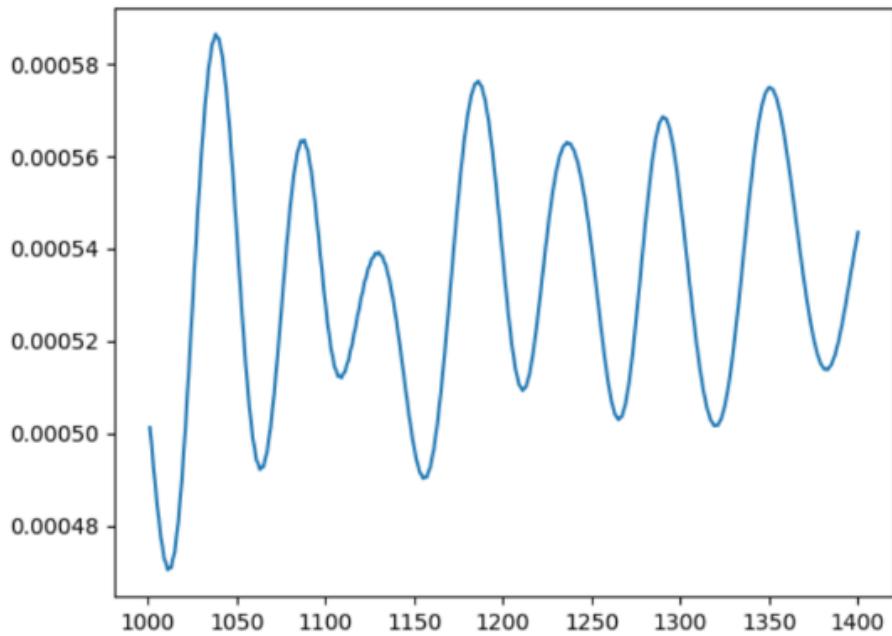


Abbildung 22: Schwankungen des  $C_w$ -Wertes in Abhängigkeit vom Simulationsschritt für eine umströmte Scheibe [61]

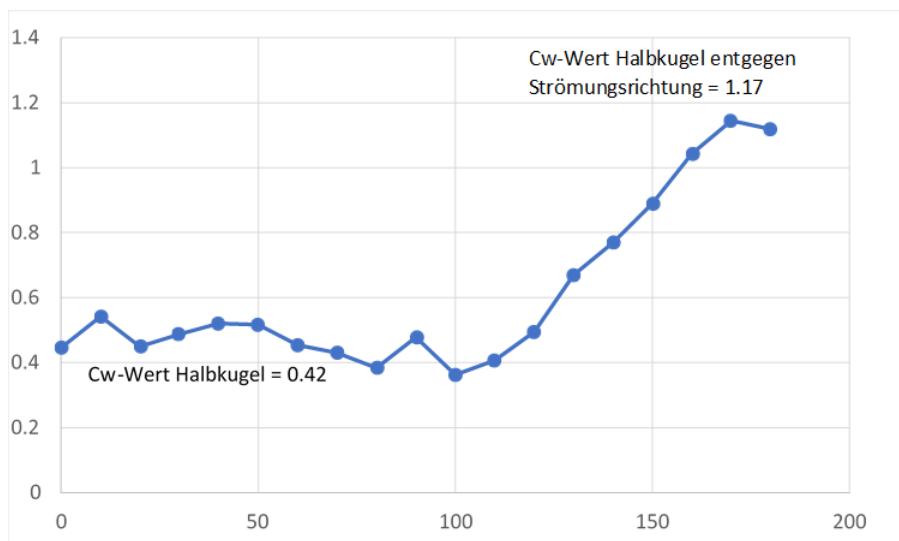


Abbildung 23:  $C_w$ -Wert einer hohlen Halbkugel in Abhängigkeit vom Anströmungswinkel [61]

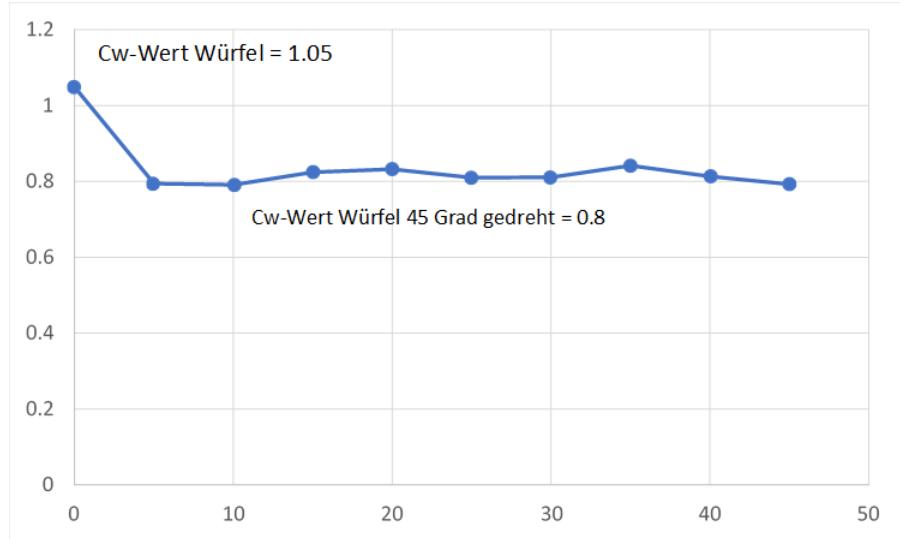


Abbildung 24:  $C_w$ -Wert eines Würfels in Abhängigkeit vom Anströmungswinkel [61]

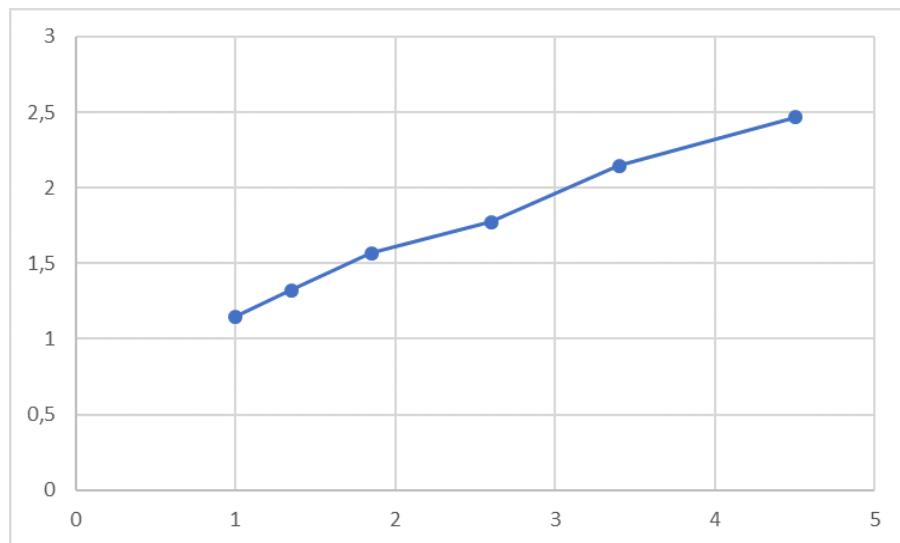


Abbildung 25:  $C_w$ -Wert in Abhängigkeit von der Anströmläche zur Oberfläche des Körpers [61]

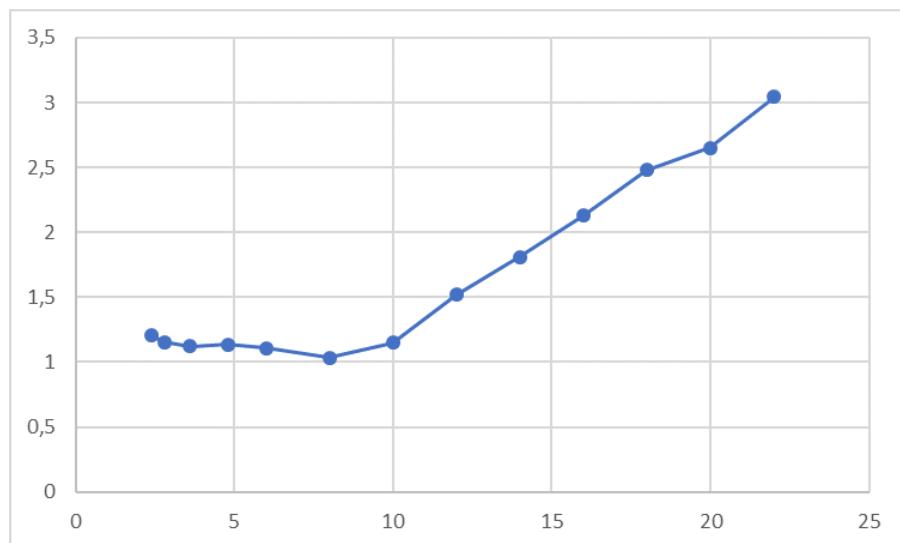


Abbildung 26:  $C_w$ -Wert in Abhängigkeit des Verhältnisses aus X-Ausdehnung zur Z-Ausdehnung des Körpers [61]

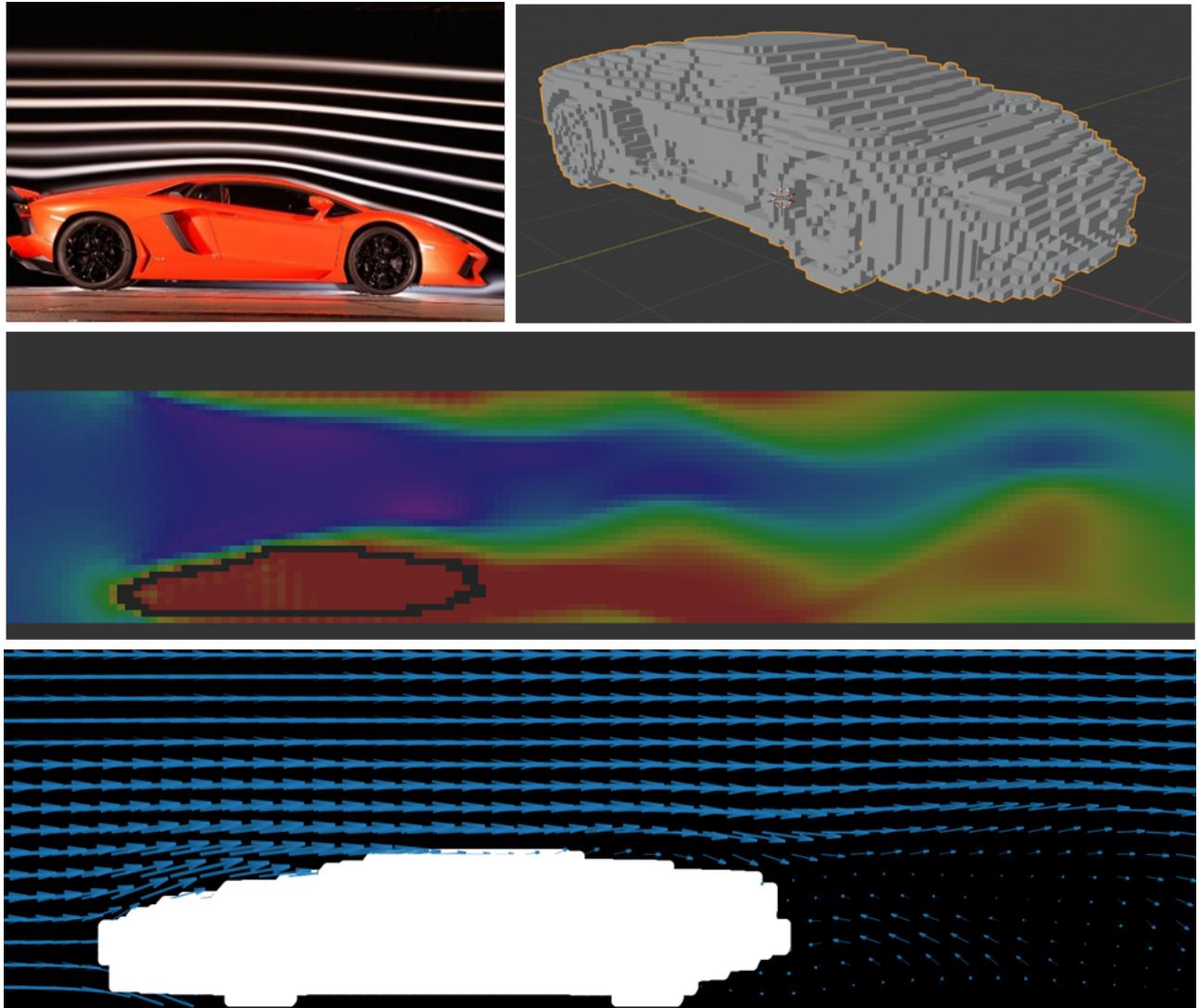


Abbildung 27: Gegenüberstellung unserer Strömungssimulation um einen Lamborghini mit der Strömungsvisualisierung in einem Windkanal [61, 60]

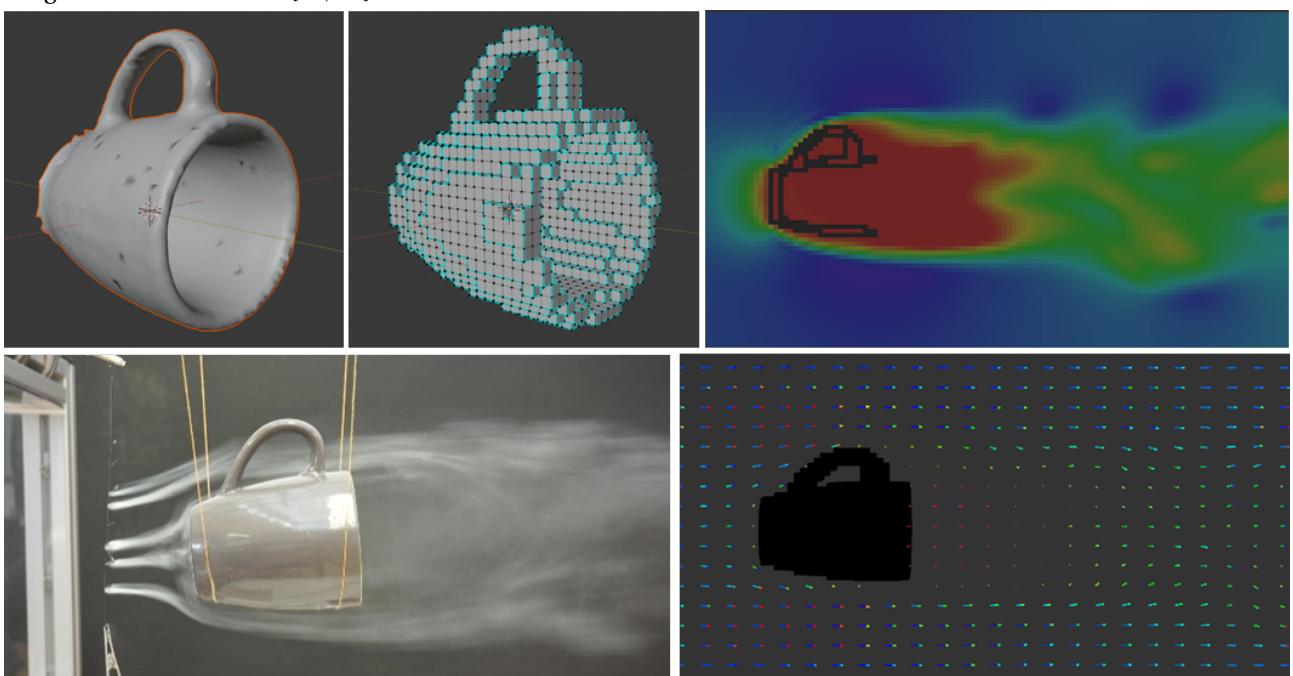


Abbildung 28: Gegenüberstellung unserer Strömungssimulation um eine Tasse mit der Strömungsvisualisierung in einem Windkanal [61]

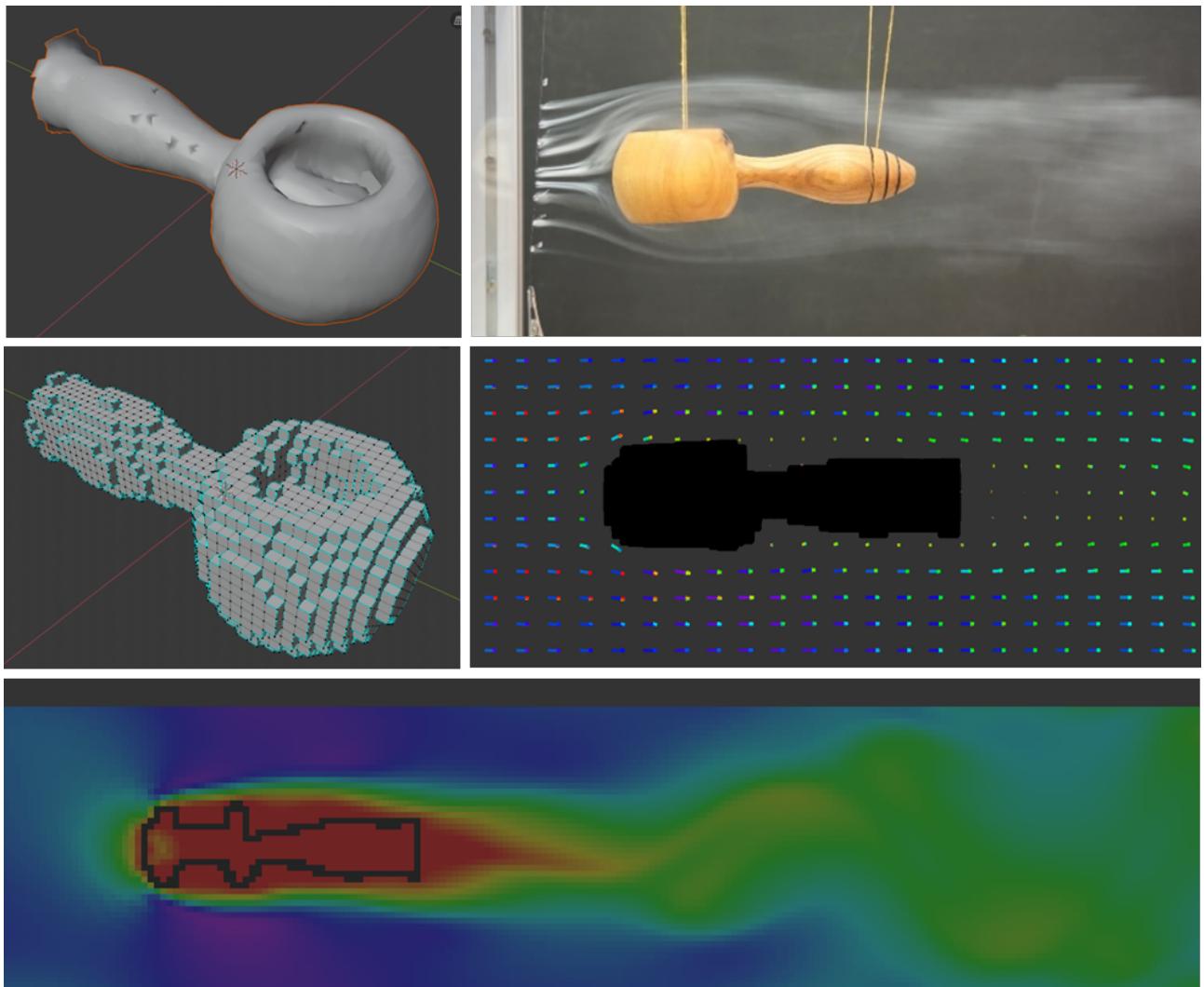


Abbildung 29: Gegenüberstellung unserer Strömungssimulation um einen Nussknacker mit der Strömungsvisualisierung in einem Windkanal [61]

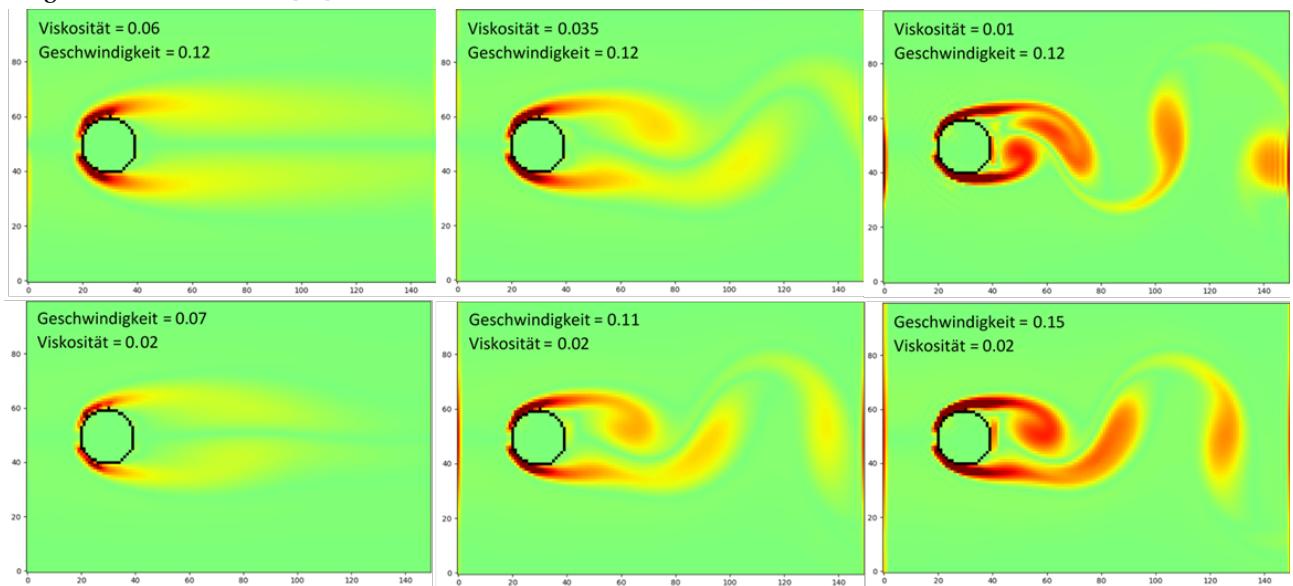


Abbildung 30: Abhängigkeit des Strömungsbildes von der Viskosität und Strömungsgeschwindigkeit [61]

## Einordnung der mesokopischen Betrachtungsweise

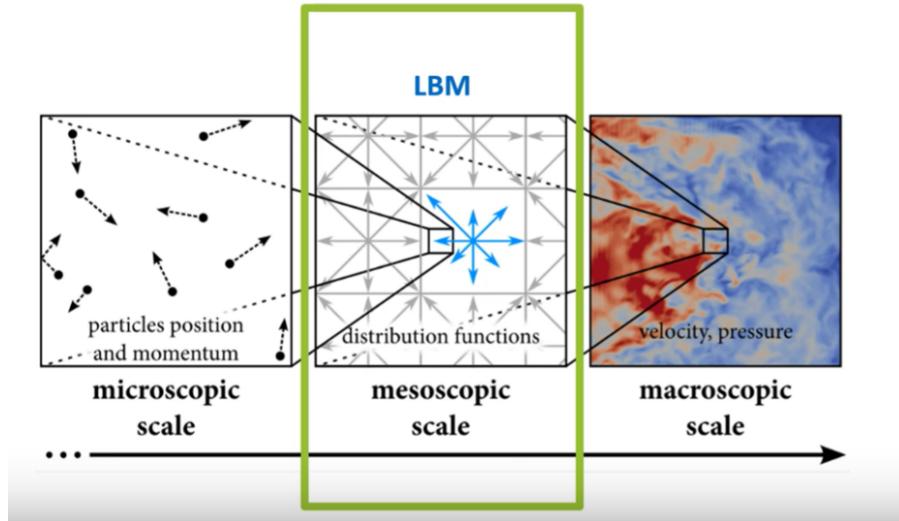


Abbildung 31: Einordnung der mesokopischen Betrachtungsebene zwischen der mikroskopischen und makroskopischen Betrachtungsebene [59]

Die makroskopische Betrachtungsebene wird oft mit dem verglichen, was mit bloßen Auge sichtbar ist. Die mikroskopische Betrachtungsebene befindet sich dagegen im Größenbereich vom Atomen und Molekülen, also im Bereich zwischen 1 bis 100 Nanometer. Die mesokopische Betrachtungsweise mit der verwendeten Lattice-Boltzmann-Methode stellt den Übergang zwischen beiden Ebenen dar.

## Die Messungen im Windkanal

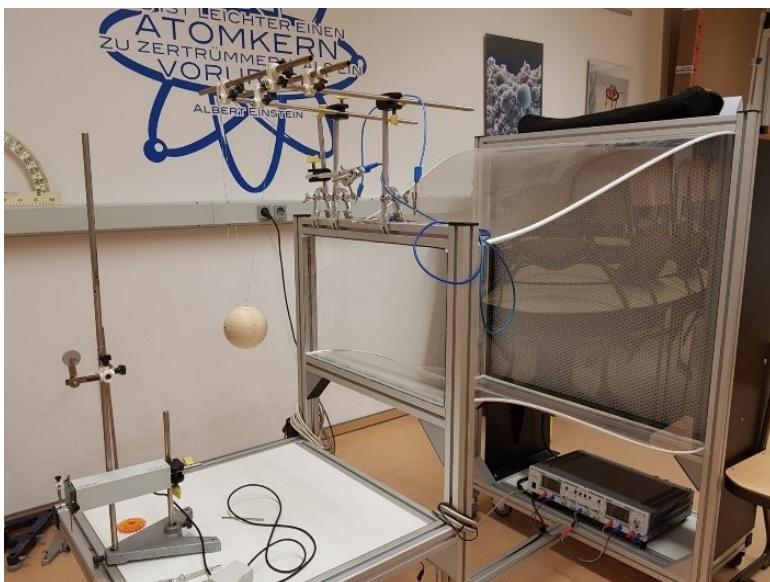


Abbildung 33: Datenblatt zum Windkanal im Schülerforschungszentrum [63]

Abbildung 32: Der Windkanal im Schülerforschungszentrum [61]

Um unsere Cw-Wert-Berechnungen von den eingescannten Körpern zu kontrollieren, haben wir die Cw-Werte dieser Körper mit dem Windkanal vom Schüler-Forschungszentrum gemessen. Dieser ist basierend auf seiner technischen Bauart ein offener Windkanal, auch Eiffelkanal genannt. Dabei wird die Luft über 25 Ventilatoren angesaugt, welche über das Bedienpult regelbar sind. Mit dem Bedienpult lässt sich die Steuerspannung der Ventilatoren und damit die Geschwindigkeit der Luftströmung im Windkanal einstellen. Die Maximalgeschwindigkeit beträgt dabei mit einer Steuerspannung von 2.5 V rund 11.5 m/s. Die Abhängigkeit der Geschwindigkeit von der Steuerspannung ist im Diagramm oben rechts dargestellt. Die Verwirbelungen der angesaugten Luft werden über einen Gleichrichter aus dünnen Röhren mit 12 mm Durchmesser und 50 mm Tiefe herausgefiltert. Letztendlich durchströmt die Luft eine Kontraktionsstrecke, wobei der Querschnitt der Strömung um 75% auf  $0.5 \cdot 0.3$  m verringert wird. Der Strahlausritt trifft letztendlich auf den Körper, dieser ist im Bild oben links eine Kugel.

Um die auf den Körper wirkende Kraft zu messen, wird dieser an mehreren Fäden an einer Aufhängungsvorrichtung über den Windkanal gehangen. Ein weiterer Faden befindet sich hinter dem Körper und wird über eine Rolle auf

einen Kraftmesser übertragen. Der Kraftmesser misst die Kraft, welche auf den Körper in Strömungsrichtung wirkt und überträgt die Daten per USB-Kabel auf einen Laptop. Dort wiederum kann man die Daten grafisch mit dem Programm Cassy-Lab 2 auswerten. Um die auf den Körper einwirkende Luftwiderstandskraft für die Cw-Wert-Messung zu bestimmen, wird der Körper einmal ohne Gegenströmung an den Kraftmesser gehangen und dann noch einmal mit Gegenströmung. Durch die Gegenströmung verringert sich die gemessene Kraft am Kraftsensor. Die Differenz beider gemessenen Kräfte ergibt die Luftwiderstandskraft. Wir haben die Luftwiderstandskräfte von den Körpern mit mehreren Geschwindigkeiten im Bereich zwischen 3.6 und 11.5 m/s gemessen. Letztendlich haben wir für die Cw-Wertberechnung die Messungen zwischen 6.4 und 8.8 m/s verwendet, da bei geringeren Geschwindigkeiten die Kraftmessung ungenau war und bei größeren Geschwindigkeiten der Körper zu stark umhergependelt ist. Um den Cw-Wert aus den Kraftmessungen für den jeweiligen Körper zu berechnen, fehlen neben der gemessenen Luftwiderstandskraft noch die Anströmfläche, die Geschwindigkeit und die Dichte der Luft. Die Geschwindigkeit lässt sich, wie oben abgebildet, aus der Steuerspannung bestimmen. Die Anströmfläche haben wir näherungsweise mit einem Messschieber ermittelt. Die Dichte der Luft haben wir bei einer Raumtemperatur von 20 Grad Celsius über die thermische Zustandsgleichung von idealen Gasen auf  $1.204 \text{ Kg/m}^3$  berechnet.

Um unsere Cw-Wert-Messungen auch einordnen zu können, haben wir die im Windkanal verwendete Reynoldszahl bestimmt. Diese wurden über die oben bereits erwähnte Formel von Viskosität, Geschwindigkeit, Dichte und der charakteristischen Länge ermittelt. Als charakteristische Länge wird dabei der Durchschnitt der Länge des Körpers senkrecht zur Strömungsrichtung verwendet. Die dynamische Viskosität von Luft beträgt  $18 \cdot 10^{-6} \text{ Pa} \cdot \text{s}$ . Alle Reynoldszahlen liegen im Bereich der Reynoldszahlen mit einem konstanten Cw-Wert, basierend auf dem Verlauf des Cw-Wertes in Abhängigkeit der Reynoldszahl von einer Kugel. Mit der Näherung, dass der Cw-Wertverlauf bei den anderen gemessenen Körpern gleich ist, können die Messungen vom Windkanal direkt mit unseren theoretischen Berechnungen verglichen werden. Die Ergebnisse haben wir in der Tabelle unten kurz zusammengefasst.

Basierend auf den ersten vier gemessenen Körpern in der Tabelle haben wir eine durchschnittliche Abweichung von dem Cw-Wert, welchen wir mit unserem Programm berechnet haben und dem Cw-Wert, welcher im Windkanal gemessen wurde, von 11.4%. Die letzten beiden Werte in der Tabelle haben größere Abweichungen und sind demnach nicht repräsentativ.

Tabelle 4: Berechnete Cw-Werte im Vergleich mit gemessenen Werten

Körper	berechneter Cw-Wert	gemessener Cw-Wert (Windkanal)	Reynoldszahl mit $\nu = 7.6 \frac{\text{m}}{\text{s}}$
Tasse	0.47	0.41	44424
Nussknacker	0.69	0.67	29871
Federnmappe	0.53	0.45	47743
Kugel	0.46	0.52	38297
Haargel-Tube	0.92	0.70	24511
Brotbüchse	1.11	0.50	37276

Der von unserer Software berechnete Cw-Werte dieser Körper befindet sich, basierend auf Erfahrungswerten mit ähnlichen Körpern, im richtigen Bereich. Die Messungen im Windkanal weichen jedoch stark davon ab, obwohl wir sie mehrmals wiederholt haben. Wir haben die beiden Werte aus dem Grund mit in die Tabelle integriert, da wir aufzeigen möchten, dass auch im Windkanal schnell Fehler entstehen können. Zum einen haben wir als Eichung des Windkanals eine Kugel verwendet. Dabei haben wir einen Cw-Wert von 0.52 gemessen. In der Realität beträgt der Cw-Wert der Kugel jedoch 0.47. Dies ist eine Abweichung von fast 10% und kann mit dem Standardfehler des Windkanals gleichgesetzt werden. Dieser Standardfehler hat sich auch bei anderen Körpern, wie zum Beispiel eine Scheibe, bewiesen. Jedoch treten daneben auch noch weitere, zufällige Fehler auf. So beginnen leichte Körper oder stark unsymmetrische Körper bei größeren Geschwindigkeiten schnell in der Aufhängung zu schwingen. Weiterhin verläuft die Kraftübertragung des Luftwiderstandes durch einen Faden auf den Kraftmesser nicht exakt senkrecht und die Körper hängen nur annähernd parallel zu Strömungsrichtung des Windkanals. Hängt zum Beispiel eine Scheibe nur leicht schräg zur Strömungsrichtung im Windkanal, so sinkt der Cw-Wert erheblich. Bei Betrachtung der Abweichung zwischen den gemessenen und berechneten Cw-Werten muss ebenfalls erwähnt werden, dass durch das Einscannen und Würfeln die Form der Körper im Programm nur näherungsweise der Form in der Realität entsprechen. Dies wird an den Abbildungen 27 bis 29 deutlich. Zusammenfassend können wir also für Messungen im Windkanal von einem durchschnittlichen Fehler zwischen 10 bis 15% ausgehen.

Neben der Cw-Wert-Messung konnten wir im Windkanal auch ein Strömungsbild um den Körper erzeugen. Dafür verwendeten wir einen Draht mit verdrehten Windungen, welcher zwischen dem Körper und dem Windkanal hängt. Auf diese Windungen tropften wir eine Glycerinlösung. Anschließend wurde der Draht an eine Spannungsquelleangeschlossen und soweit erhitzt, bis das Glycerin verdampft ist und ein Strömungsbild, basierend auf mehreren Streifen, ermöglicht hat. Dieser Prozess ist in Abbildungen 27, 28 und 29 erkennbar.

# Quellenverzeichnis

## Literaturquellen

- [1] ARNOLD, KARIN; DIETRICH, VOLKMAR; EBERLE ANDREAS; GRIMMER, ANDREAS; JENCKEL, ASTRID; GRIMMER, ANJA; KARHOS, MARIANNE; LABAHN, BETTINA; LÜTTGENS, UWE; MALZ, RALF; PETERS, JÖRN; SCHÄFER, STEFFEN; TEICHERT, BORIS; TISCHENDORF, KERSTIN; *Chemie Oberstufe*; Cornelsen Verlag; 2015 1. Auflage 4. Druck
- [2] KUCHLING, HORST; *Taschenbuch der Physik*; Fachbuchverlag Leipzig im im Carl Hanser Verlag; 2011 20.Auflage
- [3] MESCHEDE, DIETER; *Gerthsen Physik*; Springer-Verlag; 2002 21.Auflage

## Internetquellen

[3D Grafik]

- [4] Autoren Kollektiv um: 129.132.239.8; STL-Schnittstelle  
<https://de.wikipedia.org/wiki/STL-Schnittstelle>
- [5] Autoren Kollektiv um: Arilou; Dateiformat  
<https://de.wikipedia.org/wiki/Dateiformat>
- [6] Autoren Kollektiv um: Cavasin, Christian; Wavefront OBJ  
[https://de.wikipedia.org/wiki/Wavefront\\_OBJ](https://de.wikipedia.org/wiki/Wavefront_OBJ)
- [7] Autoren Kollektiv um: Coumans, Erwin; FBX  
<https://en.wikipedia.org/wiki/FBX>
- [8] Autoren Kollektiv um: Moskopp, Nils; Offenes Format  
[https://de.wikipedia.org/wiki/Offenes\\_Format](https://de.wikipedia.org/wiki/Offenes_Format)
- [9] Autoren Kollektiv um: Vierge, Marie; Rendern (Design)  
[https://de.wikipedia.org/wiki/Rendern\\_\(Design\)](https://de.wikipedia.org/wiki/Rendern_(Design))
- [10] Wavefront Technologies; .OBJ Dateierweiterung  
<https://www.reviversoft.com/de/file-extensions/obj>

[Lattice-Boltzmann]

- [11] Autoren Kollektiv um; Abas, Aizat; Lattice Boltzmann Model of 3D Multiphase Flow in Artery Bifurcation Aneurysm Problem  
<https://www.hindawi.com/journals/cmmm/2016/6143126/>
- [12] Crowl, Lindsay; The Lattice Boltzmann Method Computational Fluid Dynamics  
<https://www.math.utah.edu/~crowl/pres.pdf>
- [13] Autoren Kollektiv um: Debenben; Lattice-Boltzmann-Methode  
<https://de.wikipedia.org/wiki/Lattice-Boltzmann-Methode>
- [14] Fitzpatrick, Richard; The Maxwell distribution  
<http://farside.ph.utexas.edu/teaching/sm1/lectures/node72.html>
- [15] Autoren Kollektiv um: Harris, Stewart; Boltzmann equation  
[https://en.wikipedia.org/wiki/Boltzmann\\_equation](https://en.wikipedia.org/wiki/Boltzmann_equation)
- [16] Autoren Kollektiv um: Mandl, F; Maxwell-Boltzmann distribution  
[https://en.wikipedia.org/wiki/Maxwell\text{\textbackslash}T1\text{\textbackslash}text{\textendash}Boltzmann\\_distribution](https://en.wikipedia.org/wiki/Maxwell\text{\textbackslash}T1\text{\textbackslash}text{\textendash}Boltzmann_distribution)
- [17] Mattila, Keijo; Implementation Techniques for the Lattice Boltzmann Method  
<https://jyx.jyu.fi/bitstream/handle/123456789/24953/9789513939915.pdf?sequence=1>
- [18] Two Phase Flow and Heat Transfer; Lattice Boltzmann Method  
<https://www.youtube.com/watch?v=Cg-IRe19BEw>
- [19] Weber State University; Lattice-Boltzmann Fluid Dynamics  
<http://physics.weber.edu/schroeder/javacourse/LatticeBoltzmann.pdf>

[math. Grundlagen]

- [20] Autoren Kollektiv um: Alva2004; Divergenz mit Zylinder Koordinaten  
[https://de.wikipedia.org/wiki/Divergenz\\_eines\\_Vektorfeldes](https://de.wikipedia.org/wiki/Divergenz_eines_Vektorfeldes)
- [21] Arvo, James; Rotationsmatrix  
<https://en.wikipedia.org/wiki/Rotationsmatrix>
- [22] Autorenkollektiv um: Franzl aus tirol; Drehmatrix  
<https://de.wikipedia.org/wiki/Drehmatrix>
- [23] Gräber, Peter-Wolfgang; Vektoralgebra  
<https://tu-dresden.de/bu/umwelt/hydro/iak/ressourcen/dateien/systemanalyse/studium/folder-2009-01-29-lehre/systemanalyse/folder-2010-04-12-1173264546/wws-02.pdf?lang=de>
- [24] Autorenkollektiv um: Hagman; Punkt-in-Polygon-Test  
[https://de.wikipedia.org/wiki/Punkt-in-Polygon-Test\\_nach\\_Jordan](https://de.wikipedia.org/wiki/Punkt-in-Polygon-Test_nach_Jordan)
- [25] Universität Stuttgart; Differentialoperatoren  
[http://vhm.mathematik.uni-stuttgart.de/Vorlesungen/Vektoranalysis/Folien\\_Differentialoperatoren\\_in\\_Zylinderkoordinaten.pdf](http://vhm.mathematik.uni-stuttgart.de/Vorlesungen/Vektoranalysis/Folien_Differentialoperatoren_in_Zylinderkoordinaten.pdf)
- [26] Autoren Kollektiv um: HerrHartmuth; explizite Euler-Verfahren  
[https://de.wikipedia.org/wiki/Explizites\\_Euler-Verfahren](https://de.wikipedia.org/wiki/Explizites_Euler-Verfahren)
- [27] Weitz / HAW Hamburg; Gauß-Verfahren  
<https://www.youtube.com/watch?v=kfopPCDY1F0>

[phys. Grundlagen]

- [28] Autoren Kollektiv um: Alva2004; Navier-Stokes-Gleichungen  
<https://de.wikipedia.org/wiki/Navier-Stokes-Gleichungen>
- [29] Butkevich, Andrey; Schmiedel, Benjamin; Hydrodynamik - Euler- und Navier-Stokes-Gleichungen  
<https://www.thphys.uni-heidelberg.de/~mielke/Mechanik17-9d.pdf>
- [30] Autoren Kollektiv um: Bammel, Katja; Navier-Stokes-Gleichungen  
<https://www.spektrum.de/lexikon/physik/navier-stokes-gleichungen/10145>
- [31] Autoren Kollektiv um: Bauhofer, Prof. Dr. W.; Eulersche Gleichungen  
<https://www.spektrum.de/lexikon/physik/eulersche-gleichungen/4573>
- [32] Autoren Kollektiv um: Duesi; Reynolds-Zahl  
<https://de.wikipedia.org/wiki/Reynolds-Zahl>
- [33] Autoren Kollektiv um: Funkmich008; Torricelli Formel  
<https://de.wikipedia.org/wiki/Ausflussgeschwindigkeit>
- [34] Graber, Peter-Wolfgang; Torricelli Formel  
<https://tu-dresden.de/bu/umwelt/hydro/iak/ressourcen/dateien/systemanalyse/studium/folder-2009-01-29-lehre/systemanalyse/folder-2010-04-12-1173264546/wws-02.pdf?lang=de>
- [35] LearnMechE; Description and Derivation of the Navier-Stokes Equations  
<https://www.youtube.com/watch?v=NjoMoH51UZc&list=PLebuRGYfFXg28n878rvd3lDGtqJfYvABN&index=8&t=20s>
- [36] Numberphile; Navier-Stokes Equations  
<https://www.youtube.com/watch?v=ERBVFcvt13M&t=124s>
- [37] Autoren Kollektiv um: Svebert; ideale Flüssigkeit  
[https://de.wikipedia.org/wiki/Ideale\\_Fl%C3%BCssigkeit](https://de.wikipedia.org/wiki/Ideale_Fl%C3%BCssigkeit)
- [38] Uni Magdeburg; Körperumströmung - reibungsbehaftet  
<http://www.uni-magdeburg.de/isut/LSS/Lehre/Arbeitsheft/VII.pdf>

[Python]

- [39] Autorenkollektiv um: Holden, Steve; Python Einführung  
<https://wiki.python.org/moin/>

- [40] Hunter, John; Matplotlib Einführung  
<https://matplotlib.org/>
- [41] Klein, Bernd; Python Einführung  
<https://www.python-kurs.eu/>
- [42] Koonce, Grayson; Multithreading  
<https://graysonkoonce.com/waiting-until-a-thread-is-ready-in-python/>
- [43] Lundh, Fredrik; Python Einführung  
<http://effbot.org/>
- [44] NumPy developers; NumPy Einführung  
<http://www.numpy.org/>
- [45] Petri, Ulrich & Gutmann, Horst; Python Einführung  
<https://pyformat.info/>
- [46] SciPy developers; SciPy Einführung  
<https://docs.scipy.org/>
- [47] sentdex; Python GUI  
<https://www.youtube.com/user/sentdex/>
- [48] Seppke, Benjamin; Bildverarbeitung Python  
<https://kogs-www.informatik.uni-hamburg.de/~neumann/BV-WS-2010/Uebungen/bv-python-einfuehrung.pdf>
- [49] Shenk, Justin; 3D Würfel Darstellung  
<https://gist.github.com/JustinShenk/c407b02a4d5c19f89dfc87e9678dcc22>
- [50] Sherwood, Bruce; 3D-Animation Python  
<https://vpython.org/>
- [51] Stack Overflow developers; Stack Overflow  
<https://stackoverflow.com/>

[Texmaker]

- [52] Hammersley, John; Latex Templates  
<https://de.overleaf.com/latex/templates>
- [53] Safra; Latex Einführung  
<https://latex-kurs.blogspot.com/>
- [54] Stack Overflow developers; Latex Einführung  
<https://tex.stackexchange.com/>
- [55] Wipper, Joachim; Matrizen  
<https://mo.mathematik.uni-stuttgart.de/kurse/kurs44/seite26.html>

## Bildquellen

- [56] [https://upload.wikimedia.org/wikipedia/commons/c/cb/Lattice\\_boltzmann\\_3steps.svg](https://upload.wikimedia.org/wikipedia/commons/c/cb/Lattice_boltzmann_3steps.svg)
- [57] <http://physics.weber.edu/schroeder/javacourse/LatticeBoltzmann.pdf>
- [58] <https://jyx.jyu.fi/bitstream/handle/123456789/24953/9789513939915.pdf?sequence=1> S.68
- [59] <https://youtu.be/qhr3p0aShjg>
- [60] <https://img1.auto-motor-und-sport.de/Lamborghini-Aventador-LP-700-4-\\Seite-Windkanal-articleGalleryOverlay-9c21d659-615877.jpg>
- [61] Aus eigener Quelle
- [62] <https://upload.wikimedia.org/wikipedia/de/thumb/8/81/Kugel-Reynolds.png/500px-Kugel-Reynolds.png>
- [63] Send, Johanna / Dr.Wolfgang; Anleitung zum Windkanal; Großer Windkanal ANIPROP GWK 1; Seite 11

## **Eidesstattliche Erklärung**

Hiermit erklären wir, dass wir die vorliegende Arbeit selbständig und ohne fremde Hilfe angefertigt haben. Textpassagen, die wörtlich oder dem Sinn nach auf Arbeiten Dritter beruhen, haben wir als solche gekennzeichnet.

Unterschriften der Autoren:

Adrian Kühn : 

Frank Long : 

Paul Marschall : 