



Fachbereich Elektrotechnik, Maschinenbau
und Technikjournalismus (EMT)
Studiengang Maschinenbau (M. Eng.)

Master-Thesis

Untersuchung der turbulenten Umströmung verschiedener Dachformen eines freistehenden Hauses mit der Lattice-Boltzmann-Methode

Vorgelegt von:

Martin Leonhard Kliemank
Stöcken 11
53797 Lohmar
Tel. +49 1578 9033776
martin.kliemank@gmx.de
Matr.-Nr. 9024158

Erstprüfer: Prof. Dr. Dirk Reith
Zweitprüfer: Prof. Dr. Gerd Steinebach

Sankt Augustin, den 8. April 2021

Abstract

In this work, the effects of the roof pitch of isolated gable-roof houses on pressures impacting the roof surface as result of airflow orthogonal to the ridge are investigated by simulation with the lattice Boltzmann method using the KBC collision operator. The influence of roof overhangs on these pressures is also investigated.

The simulations results are compared to reference wind tunnel measurements for validation. The results show a good agreement on velocity, but only qualitative agreement on pressures with relative differences of up to 50 %. Next, three roof pitches of 20°, 35° and 50° simulated at a Reynolds number of 2×10^4 . The pressures in an area above the roof are investigated, as well as the airflow in a lengthwise slice through the middle of the domain. To investigate the influence of roof overhangs on the pressures the simulation is conducted twice, once with roof overhangs and once without.

It is found that there is a limited number of boundary conditions for the lattice Boltzmann method that perform well at high Reynolds numbers. The edges between boundary conditions are of special concern as a frequent source of instability.

The results of the simulation show that airflow around the roofs with 35° and 50° pitch behaves very similarly while the airflow around the roof with 20° pitch is distinct. It is concluded that between 20° and 35° a shift in the airflow pattern occurs, which leads to the steeper roofs being subject to lower wind loads in the investigated airflow. Since pressures among the steeper roofs grow with the roof pitch, an optimal pitch is suspected between 20° and 35°.

The comparison between the two simulations results shows that roof overhangs on gable-roofs can reduce the intensity and amount of fluctuations of the pressure on the roof, however, this effect decreases with increasing pitch and is barely noticeable at 50°.

Kurzfassung

In dieser Arbeit werden die Auswirkungen der Dachneigung von Satteldächern freistehender Einfamilienhäuser auf die Drücke, welche bei Anströmung orthogonal zum First auf die Dachflächen wirken, untersucht. Der Einfluss von Dachüberständen auf diese Drücke wird ebenfalls untersucht. Die Untersuchung geschieht durch Simulation mit der Lattice-Boltzmann-Methode mit dem KBC-Kollisionsoperator.

Die Ergebnisse der Simulation werden zur Validierung mit Windtunnel-Messwerten einer Referenz verglichen. Der Vergleich zeigt eine gute Übereinstimmung hinsichtlich der Strömungsgeschwindigkeit, aber nur qualitative Übereinstimmung der Drücke, welche relative Abweichungen von bis zu 50% aufweisen. Anschließend wird die Umströmung dreier Satteldächer mit Dachneigungen von 20°, 35° und 50° bei einer Reynolds-Zahl von 2×10^4 simuliert. Die auftretenden Drücke oberhalb des Daches sowie die Strömungsgeschwindigkeit in einem Längsschnitt der Simulationsdomäne werden untersucht. Um den Einfluss der Dachüberstände zu untersuchen, wird die Simulation einmal mit und einmal ohne Dachüberstände durchgeführt.

Es zeigt sich, dass es für die Lattice-Boltzmann-Methode nur eine begrenzte Auswahl an Randbedingungen gibt, die bei hohen Reynolds-Zahlen gute Ergebnisse liefern. Insbesondere die Kanten zwischen den Randbedingungen sind als Quelle von Instabilität problematisch.

Die Ergebnisse der Simulation zeigen, dass sich die Umströmung der Dächer mit Neigungswinkeln von 35° und 50° sehr ähnlich verhält, während die des Daches mit dem Neigungswinkel von 20° davon abweicht. Es wird entsprechend gefolgert, dass zwischen 20° und 35° eine bedeutende Veränderung in der Umströmung auftritt. Die steileren Dächer erfahren dabei bei der untersuchten Anströmung eine geringere dynamische Belastung. Da die auftretenden Drücke unter den steileren Dächern mit dem Neigungswinkel zunehmen, wird vermutet, dass zwischen 20° und 35° einen optimalen Neigungswinkel liegt.

Der Vergleich der Ergebnisse beider Simulationen deutet darauf hin, dass Dachüberstände bei Satteldächern die Intensität und Häufigkeit von Druckschwankungen an der Leeseite des Daches reduzieren können. Der beobachtete Effekt nimmt jedoch mit wachsender Dachneigung ab und ist bei 50° bereits kaum feststellbar.

Inhaltsverzeichnis

Abstract	i
Kurzfassung	iii
Abbildungsverzeichnis	III
Tabellenverzeichnis	V
Formelzeichenverzeichnis	VII
1 Einleitung	1
1.1 Motivation	1
1.2 Stand der Forschung	1
1.3 Aufbau der Arbeit	2
2 Die Lattice-Boltzmann-Methode	3
2.1 Makroskopische Gleichungen für das Fluidverhalten	3
2.2 Herleitung der Lattice-Boltzmann-Gleichung und ihre Momente	4
2.3 Zusammenhang von Realität und Simulation	6
2.4 Algorithmus und Implementierung	7
2.5 Kollisionsoperatoren	8
2.5.1 BGK	8
2.5.2 KBC	9
2.6 Anfangsbedingungen	9
2.7 Randbedingungen	10
2.8 Genauigkeit der Methode	12
2.9 Parametrierung der Simulation	13
2.9.1 Parameter der Lattice-Boltzmann-Methode	13
2.9.2 Performanz-Aspekte und ihre Einflussgrößen	14
2.9.3 Strategie zur Parameterwahl	15
3 Die LBM mit dem Löser Lettuce	17
3.1 Der Löser Lettuce	17
3.1.1 Aufbau des Lösen	17
3.1.2 Benutzung des Lösen	19
3.2 Erweiterungen des Lösen	21
3.2.1 Parallelisierung	21
3.2.1.1 Funktionsweise und Aufbau	22
3.2.1.2 Benutzung	25
3.2.1.3 Verifizierung	26
3.2.1.4 Benchmark	26
3.2.1.5 Ausblick	29
3.2.2 Randbedingungen	30
3.2.2.1 Null-Gradienten-Randbedingung	31
3.2.2.2 Non-Equilibrium-Extrapolation-Randbedingung	31
3.2.2.3 Probleme der Randbedingungen	31

4 Untersuchung der Umströmung des Hauses	33
4.1 Aufbau der Simulations-Umgebung	33
4.1.1 Allgemeine Konfiguration	33
4.1.2 Aufbau der Simulations-Domäne	33
4.1.3 Simulations-Parameter	35
4.2 Verifizierung der Simulationsumgebung	37
4.3 Untersuchung der Umströmung des Hauses	41
4.3.1 Einfluss der Dachneigung auf den Druck an der Dachoberfläche	41
4.3.2 Einfluss der Dachneigung auf die umgebende Strömung	42
4.3.3 Einfluss des Dachüberstands	45
4.4 Auswertung und Diskussion	51
5 Zusammenfassung und Ausblick	55
Literatur	57

Abbildungsverzeichnis

2.2-1	Beispiele für Geschwindigkeitsdiskretisierungen: der D2Q9- und D3Q27-Geschwindigkeitssatz	5
2.4-1	Visualisierung des Kollisions- und Strömungsschrittes der Lattice-Boltzmann-Methode	7
2.4-2	Übersicht über die zentralen Schritte des üblichen Algorithmus einer Implementierung der Lattice-Boltzmann-Methode	8
3.1-1	Logo des Lattice-Boltzmann-Lösers Lettuce	17
3.1.1-1	Vereinfachtes UML-Klassendiagramm des LBM-Lösers Lettuce	18
3.1.1-2	Vereinfachte Visualisierung des Ablaufs einer Simulation mit Lettuce	20
3.2.1.1-1	Visualisierung der Aufteilung der Simulationsdomäne auf einzelne Prozesse im Rahmen der Parallelisierung	22
3.2.1.1-2	Übersicht über der Erweiterungen des Lösers Lettuce für die Parallelisierung	23
3.2.1.3-1	Beispiel für ein Ergebnis der Verifizierung der Parallelisierung	27
3.2.1.4-1	Skalierung der Rechengeschwindigkeit der parallelisierten Simulation mit der Anzahl an ausführenden Grafikkarten	28
3.2.1.4-2	Übersicht der Rechengeschwindigkeit der parallelisierten Simulation auf verschiedenen Anzahlen an Prozessor-Threads	28
3.2.1.4-3	Skalierung der Simulationsdomänengröße mit der Anzahl an Grafikkarten	30
4.1.2-1	Übersicht über den Aufbau der Simulationsdomäne für die Untersuchung der Umströmung des Hauses	34
4.1.2-2	Übersicht über die untersuchten Häuser bzw. Dachformen	35
4.1.2-3	Visualisierung des Geschwindigkeitsprofils am Einlass der Simulationsdomäne	36
4.2-1	Abgleich der normierten Geschwindigkeit der Simulationsergebnisse mit den Windkanal-Messwerten der Referenz	38
4.2-2	Direkter Vergleich der Verteilung der Strömungsgeschwindigkeit mit den Simulationsergebnissen der Referenz [3]	38
4.2-3	Abgleich der normierten turbulenten kinetischen Energie der Simulationsergebnisse mit den Windkanal-Messwerten der Referenz	39
4.2-4	Abgleich des Druckkoeffizienten der Simulationsergebnisse mit den Windkanal-Messwerten der Referenz	39
4.2-5	Direkter Vergleich der Verteilung des Druckkoeffizienten mit den Simulationsergebnissen der Referenz [3]	40
4.3-1	Visualisierung der Linie und Flächen am Dach, auf welchen der Druckkoeffizient untersucht wird	41
4.3.1-1	Vergleich des Druckkoeffizienten am Dach bei den verschiedenen Neigungswinkeln	43
4.3.1-2	Vergleich der Verteilung des mittleren Druckkoeffizienten über die Dachfläche bei den untersuchten Dachneigungen	43
4.3.1-3	Vergleich der Verteilung der Standardabweichung des Druckkoeffizienten über die Dachfläche bei den untersuchten Dachneigungen	44
4.3.1-4	Vergleich der Verteilung der Maxima des Druckkoeffizienten über die Dachfläche bei den untersuchten Dachneigungen	44

4.3.1-5	Vergleich der Verteilung der Minima des Druckkoeffizienten über die Dachfläche bei den untersuchten Dachneigungen	45
4.3.2-1	Vergleich der Verteilung der Strömungsgeschwindigkeit der beobachteten Strömung bei den verschiedenen Dachneigungen	46
4.3.2-2	Darstellung eines Zeitschritts der betrachteten Strömung als Oberflächen gleichen Q's	46
4.3.3-1	Vergleich des zeitlich gemittelten Druckkoeffizienten der untersuchten Dächer mit und ohne Dachüberstand	48
4.3.3-2	Vergleich des Druckverhaltens beim Dach mit 20° Neigungswinkel mit und ohne Dachüberstand	48
4.3.3-3	Vergleich der Verteilung des Druckkoeffizienten über das Dach mit 20° Neigungswinkel mit und ohne Dachüberstand	49
4.3.3-4	Vergleich des Druckverhaltens beim Dach mit 35° Neigungswinkel mit und ohne Dachüberstand	50

Tabellenverzeichnis

3.2.1.1-1 Übersicht über die Funktionsweise der Implementierung des verteilten Strömungsschrittes	24
4.1.3-1 Übersicht über die grundlegenden Umrechnungsfaktoren zwischen physikalischen und Lattice-Einheiten bei den gewählten Parametern	37

Formelzeichenverzeichnis

α	Profilexponent des Windgeschwindigkeitsprofils
c_i	i-ter Geschwindigkeitsvektor eines Geschwindigkeitssatzes
C_p	Druckkoeffizient
c_s	Schallgeschwindigkeit
c_s^*	Schallgeschwindigkeit in Lattice-Einheiten
f_i	Verteilungsfunktion der Partikelpopulationen mit Geschwindigkeit c_i
f_i^{eq}	Gleichgewichtsverteilung der Partikelpopulationen mit Geschwindigkeit c_i
f_i^{neq}	Nichtgleichgewichtsanteil der Verteilungsfunktion f_i
\mathbf{j}	Impulsdichte
l	Seitenlänge des Hauses
l^*	Seitenlänge des Hauses in Lattice-Einheiten
l_0	charakteristische Länge
l_0^*	charakteristische Länge in Lattice-Einheiten
μ	dynamische Viskosität
μ^*	dynamische Viskosität in Lattice-Einheiten
Ma	Machzahl
ν	kinematische Viskosität
ν^*	kinematische Viskosität in Lattice-Einheiten
Ω_i	Kollisionsoperator in Richtung c_i
p	Druck
p^*	Druck in Lattice-Einheiten
Q	Q-Wert
Re	Reynolds-Zahl
ρ	Dichte
ρ^*	Dichte in Lattice-Einheiten
ρ_0	charakteristische Dichte
ρ_0^*	charakteristische Dichte in Lattice-Einheiten
Δt	Zeitschrittweite
Δt^*	Zeitschrittweite in Lattice-Einheiten
τ	Relaxationszeitkonstante
τ^*	Relaxationszeitkonstante in Lattice-Einheiten
\mathbf{u}	makroskopische Fluidgeschwindigkeit
\mathbf{u}^*	makroskopische Fluidgeschwindigkeit in Lattice-Einheiten
\mathbf{u}_0	makroskopische, charakteristische Fluidgeschwindigkeit
\mathbf{u}_0^*	makroskopische, charakteristische Fluidgeschwindigkeit in Lattice-Einheiten
\mathbf{u}_x	makroskopische Fluidgeschwindigkeit in x-Richtung
w_i	Gewicht zur Geschwindigkeit c_i
\mathbf{x}	Positionsvektor innerhalb der Simulationsdomäne
Δx	Gitterkonstante
Δx^*	Gitterkonstante in Lattice-Einheiten

1 Einleitung

In diesem Kapitel soll zunächst auf die Motivation für die Durchführung dieser Arbeit eingegangen werden. Anschließend wird der Stand der Forschung hinsichtlich der relevanten Themen vorgestellt und abschließend auf den weiteren Aufbau der Arbeit eingegangen.

1.1 Motivation

In Deutschland gibt es rund 15.9 Millionen Einfamilienhäuser, was einen Großteil der etwa 19.1 Millionen Wohngebäude darstellt [1]. Von diesen werden im Durchschnitt 0.9 Millionen pro Jahr durch Sturm und Hagel beschädigt [2]. Dementsprechend ist die Widerstandskraft von Häusern und insbesondere Einfamilienhäusern gegen den Einfluss von Wind ein bedeutendes Forschungsfeld. Wichtige Einflussparameter sind hier unter anderem die Form und Neigung des Daches [3, 4]. Das Ziel dieser Arbeit ist es daher, genauerer Einblick in die Zusammenhänge zwischen diesen Faktoren und auftretenden Drücken, die zur Schädigung führen können [4], zu gewinnen. Dabei wird der Fokus auf das Satteldach als eine sehr verbreitete Dachform gelegt. Als Grundlage der Untersuchung wird die Strömung um Häuser verschiedener Dachformen simuliert, wodurch umfassendere Informationen gewonnen werden können als im Windkanal. Hierzu wird die Lattice-Boltzmann-Methode verwendet, da sie das Untersuchen komplexer Geometrien erleichtert und sich auf Grund der niedrigen numerischen Dissipation gut für die Berechnung von Turbulenz eignet [5].

1.2 Stand der Forschung

Als wichtiger Aspekt des Lebensraums von Menschen, ist die Umströmung von Gebäuden eine aktive Forschungsrichtung, nicht nur aus der eingangs genannten Motivation. Untersucht werden dabei sowohl ganze Stadtteile, bspw. hinsichtlich Themen wie der Schadstoffverteilung [6], als auch einzelne Gebäude. Bei den Einfamilienhäusern, welche im Fokus dieser Arbeit stehen, ist ebenfalls schon eine Vielzahl von Aspekten untersucht worden, so etwa die Auswirkungen der Dachform auf die Strömungen in der Umgebung des Hauses [7] oder auf die Belastung benachbarter Häuser [8]. Auch zu den Auswirkungen der Dachform auf die Windlasten am selben Gebäude gibt es bereits verschiedene Untersuchungen, diese fokussieren sich dabei unter anderem auf pyramidenförmige [9] und flache Dächer [10]. Auch die stetige Umströmung von Satteldächern wurden in dieser Hinsicht bereits untersucht [11]. Der Einfluss der Dachneigung von Satteldächern auf ihre turbulente Umströmung wurde bisher vor allem hinsichtlich der Struktur der auftretenden Wirbel betrachtet [3, 12], und um die Qualität numerischer Modelle zu eruieren. Eine zeitlich feiner aufgelöste Untersuchung hinsichtlich der Drücke am Dach wurde bisher nicht durchgeführt, und ist daher der Fokus dieser Arbeit. Darüber hinaus soll der Einfluss von Dachüberständen am Satteldach untersucht werden, da es hierzu bisher wenige Arbeiten gibt.

Die Untersuchungen der Umströmung von Gebäuden haben historisch vor allem mit Windkanalexperimenten [3, 8, 12] und Simulatoren [13] stattgefunden, die das Strömungsverhalten zwar sehr realistisch wiedergeben können, aber eingeschränkte Auswertungsmöglichkeiten bieten. Immer häufiger werden daher auch numerische Methoden genutzt, die meist durch Windkanalexperimente oder Messdaten aus der Natur verifiziert werden [3, 6, 7]. Dabei wird vor allem eine Methode basierend auf den Reynolds-gemittelten Navier-Stokes-Gleichungen

(RANS) angewendet [3, 7], bei der die Strömung über die Zeit gemittelt wird und so nur die groben, statischen Strukturen der Turbulenz wiedergeben werden können [14]. Dagegen bietet eine Simulation mit der Lattice-Boltzmann-Methode, welche auch transiente Strukturen wiedergibt, das Potential detailliertere Daten zu gewinnen, bei denen auftretende Extrema besser abgebildet werden [5]. Die Anwendung der Lattice-Boltzmann-Methode in diesem Feld wurde bereits von einigen Arbeiten erprobt, so bspw. für die Simulation des Krankenhausregertransports durch die Luft in Krankenzimmern mit COVID-19-Patienten [15], für die Simulation der Windströmungen in der Umgebung eines Gebäudes [16] und auch für die Simulation der Windbewegung innerhalb größerer Gebiete [6, 17].

Während die turbulente Umströmung von Satteldächern verschiedener Steigung bereits mit RANS untersucht wurde, soll das in dieser Arbeit nun zeitlich höher aufgelöst mit der Lattice-Boltzmann-Methode geschehen. Dabei wird besonderer Fokus auf die auftretenden Drücke am Dach gelegt, da diese eine wichtige Grundlage für die Schädigung des Gebäudes durch Wind darstellen. Zusätzlich wird außerdem der Einfluss von Dachüberständen auf die Drücke untersucht, da es dazu bisher wenig Ergebnisse gibt.

Darüber hinaus wird in dieser Arbeit der KBC-Kollisionsoperator für implizite Large-Eddy-Simulation eingesetzt (vgl. Abschn. 2.5.2), der bereits seit einiger Zeit veröffentlicht ist und an Standardfällen validiert wurde, mit dem aber bisher noch viele Anwendungen simuliert wurden [18, 19, 20]. Dementsprechend sollen hier entsprechende Erfahrungen gesammelt und dokumentiert werden.

1.3 Aufbau der Arbeit

Im weiteren Verlauf der Arbeit werden zunächst die Grundlagen erläutert, die für das Verständnis der weiteren Arbeit erforderlich sind (vgl. Kap. 2), darunter die Herleitung der Lattice-Boltzmann-Methode, die Umsetzung des sich daraus ergebenden Algorithmus sowie einige seiner Parameter und Eigenschaften. Anschließend wird der verwendete Lattice-Boltzmann-Löser Lettuce vorgestellt und auf die im Rahmen der Arbeit geleisteten Beiträge dazu eingegangen (vgl. Kap. 3), darunter eine Erweiterung, die parallelisiertes Simulieren erlaubt sowie neue Randbedingungen. Danach wird in Kapitel 4 der Aufbau der durchgeführten Simulation erläutert, wobei auch auf den Aufbau der Simulationsdomäne und die gewählten Parameter eingegangen wird. Die so definierte Simulation wird dann in Abschnitt 4.2 gegen die Ergebnisse einer Referenz verifiziert, um ihre Aussagekraft zu evaluieren. In Abschnitt 4.3 werden dann die eigentlichen Ergebnisse der Untersuchung vorgestellt und untersucht, insbesondere hinsichtlich der Auswirkungen der Dachneigung und dem Einfluss von Dachüberständen. Es werden außerdem mögliche zukünftige Schritte für eine Verbesserung der Aussagekraft und Gewinn weiterer Ergebnisse vorgeschlagen. Abschließend werden die Ergebnisse der Arbeit in Kapitel 5 noch einmal zusammengefasst und ein mögliches weiteres Vorgehen umrissen.

2 Die Lattice-Boltzmann-Methode

In diesem Abschnitt werden die Grundlagen der Lattice-Boltzmann-Methode (LBM) erläutert, nach welcher die Simulationen in dieser Arbeit durchgeführt werden. Die LBM ist eine etablierte Methode zur Simulation von Fluiddynamik. Sie basiert auf der Boltzmann-Gleichung, die Teil der kinetischen Gastheorie ist. [5]

Einleitend wird kurz auf die Herleitung der Methode eingegangen, bevor der Algorithmus der Methode und seine Implementierung vorgestellt werden. Anschließend wird auf Details zu einigen Bestandteilen des Algorithmus, wie Kollisionsoperatoren und Randbedingungen eingegangen und die Errechnung von weiteren Größen aus den Simulationsergebnissen vorgestellt. Abschließend wird außerdem auf einige Aspekte zur Genauigkeit und Stabilität der Methode eingegangen. Die Erläuterungen basieren dabei auf [5].

2.1 Makroskopische Gleichungen für das Fluidverhalten

Bevor auf die Lattice-Boltzmann-Methode an sich eingegangen wird, sollen zunächst die makroskopischen Gleichungen, deren Lösungen sie approximiert, vorgestellt werden. Die sog. Navier-Stokes-Gleichungen bilden das makroskopische Verhalten von Fluiden als Kontinuum ab. Mit Hilfe ihrer Lösungen kann daher das Verhalten von Fluiden vorhergesagt bzw. simuliert werden. [5, 21]

Die Navier-Stokes-Gleichungen setzen sich dabei aus der sog. Kontinuitäts- und Impulsgleichung zusammen. Die Kontinuitätsgleichung drückt die Erhaltung der Masse aus [5, 21]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \quad (2.1-1)$$

Dabei bezeichnet ρ die Dichte des Fluids, t die Zeit und \mathbf{u} den Geschwindigkeitsvektor des Fluids. $\rho \mathbf{u} = \mathbf{j}$ wird dabei auch als Impulsdichte bezeichnet. Für inkompressible Fluide vereinfacht sich die Kontinuitätsgleichung entsprechend zu [21]

$$\nabla \cdot \mathbf{u} = 0. \quad (2.1-2)$$

Die nächsten wichtigen Gleichungen sind die eigentlichen Navier-Stokes-Gleichungen (je eine pro Raumrichtung), welche die Impulserhaltung ausdrücken. Die von der LBM approximierte, inkompressive Form lautet [5, 21]:

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right) = -\nabla p + \mu \Delta \mathbf{u} + \mathbf{F} \quad (2.1-3)$$

wobei Δ den Laplace-Operator bezeichnet, \mathbf{F} auf das Fluid einwirkende Kräfte und μ die dynamische Viskosität des Fluids.

Um dieses System von Gleichungen zu schließen wird noch eine weitere Gleichung benötigt. Dazu wird unter anderem die isotherme Zustandsgleichung für ideale Gase verwendet, die auch in der LBM Anwendung findet [5]:

$$p = \rho R T_0 = c_s^2 \rho \quad (2.1-4)$$

wobei c_s die Schallgeschwindigkeit und R die Gaskonstante bezeichnet.

Obwohl die Lattice-Boltzmann-Methode und auch ihre im nächsten Abschnitt angerissene Herleitung nicht direkt etwas mit diesem Gleichungssystem zu tun haben, kann mit Hilfe einer sog. Chapman-Enskog-Analyse [22] gezeigt werden, dass sie eben dieses Gleichungssystem löst und somit das Verhalten von inkompressiblen Fluiden simulieren kann [5].

Neben den Gleichungen zur Beschreibung des Fluidverhaltens, sind auch einige dimensionslose Kenngrößen von Bedeutung. Eine wichtige Größe ist dabei die Reynolds-Zahl

$$\text{Re} = \frac{u_0 l_0}{\nu}. \quad (2.1-5)$$

Sie ergibt sich aus einer charakteristischen Länge und Geschwindigkeit sowie der kinematischen Viskosität, die über $\nu = \mu/\rho$ auch mit der dynamischen Viskosität verknüpft ist. Die Reynolds-Zahl ist ein Maß für die Turbulenz einer Strömung. Bei hohen Reynolds-Zahlen wird die Strömung meist von Turbulenz dominiert. [5, 14]

Eine weitere wichtige Größe ist die sog. Machzahl, sie bezeichnet das Verhältnis einer Geschwindigkeit zu der Schallgeschwindigkeit des entsprechenden Fluids [5, 14]:

$$\text{Ma} = \frac{|\mathbf{u}|}{c_s} \quad (2.1-6)$$

Eine andere Größe zur Beschreibung von Fluidverhalten ist der sog. Q-Wert, der zur Erkennung von Wirbeln verwendet werden kann. Er ergibt sich aus der zweiten Invarianten des Geschwindigkeitsgradienten:

$$Q = -\frac{1}{2} \frac{\partial u_i}{\partial x_j} \frac{\partial u_j}{\partial x_i}. \quad (2.1-7)$$

Oberflächen mit konstantem Q-Wert werden häufig zur Visualisierung dynamischer Turbulenz eingesetzt [14].

2.2 Herleitung der Lattice-Boltzmann-Gleichung und ihre Momente

In diesem Abschnitt soll kurz auf die Herleitung der Lattice-Boltzmann-Gleichung, die den Kern der Methode bildet, und ihre relevanten Momente eingegangen werden. Die Erläuterungen orientieren sich dabei an [5].

Die Lattice-Boltzmann-Methode basiert ursprünglich auf der Boltzmann-Gleichung, die Teil der kinetischen Gastheorie ist. Die kinetische Gastheorie beschreibt das Verhalten von Gasen auf einer mesoskopischen Skala mithilfe der Partikel-Verteilungsfunktion $f(\mathbf{x}, \xi, t)$. Diese beschreibt, wie die Partikel des Fluids sich im Raum \mathbf{x} und im Geschwindigkeitsraum ξ , dem Raum möglicher Geschwindigkeiten, verteilen. Die Boltzmann-Gleichung beschreibt die Veränderung dieser Verteilungsfunktionen über die Zeit.

Um diese Gleichung numerisch zu lösen, werden ihre Eingangsgrößen diskretisiert. Die Zeit wird auf nacheinander folgende Zeitschritte Δt beschränkt und der Raum auf ein regelmäßiges Gitter von Gitterpunkten im Abstand Δx . Der Geschwindigkeitsraum wird ebenfalls diskretisiert, indem die möglichen Geschwindigkeiten auf eine begrenzte Menge von gewichteten Geschwindigkeitsvektoren c_i beschränkt werden, was einer Quadratur entspricht. Dabei gibt es verschiedene mögliche Kombinationen, wie den sog. D2Q9- oder D3Q27-Geschwindigkeitssatz, die in Abbildung 2.2-1 dargestellt sind. Bei der Bezeichnung $DdQq$ gibt d die Anzahl der Dimensionen und q die Anzahl der zulässigen Geschwindigkeitsvektoren an. Für die Simulation auf regelmäßigen Gittern ergeben sich die Geschwindigkeiten so, dass sie jeweils in einem Zeitschritt zu einem der benachbarten Gitterpunkt führen, oder einen Stillstand ausdrücken. Die Wahl des Geschwindigkeitsmodells beeinflusst dabei unter anderem Rechenaufwand und

Speicherbedarf sowie Genauigkeit und Stabilität der Simulation. Für die Simulationen im Rahmen dieser Arbeit wird der D3Q27-Geschwindigkeitssatz verwendet, da er zwar mehr Ressourcen benötigt, sich aber auf Grund seiner stärkeren Isotropie besser für die Simulation von Strömungen mit hoher Reynolds-Zahl eignet als kleinere 3D-Geschwindigkeitssätze [5].

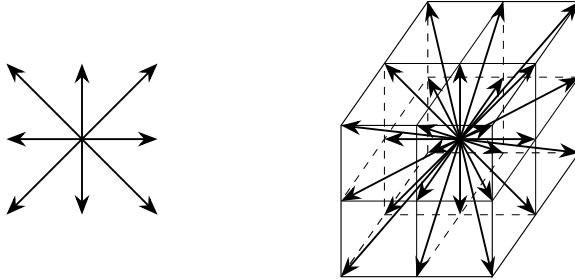


Abbildung 2.2-1: Beispiele für Geschwindigkeitsdiskretisierungen: der D2Q9- (links) und D3Q27-Geschwindigkeitssatz (rechts). Die gezeigten Geschwindigkeitsvektoren stellen die nach der Diskretisierung möglichen Geschwindigkeiten der Partikel-Populationen dar. Neben den eingezeichneten Vektoren gibt es zusätzlich einen Vektor der Länge 0 im Ursprung, welcher dem stationären Fall, also Populationen, die sich nicht bewegen, entspricht. Die weiteren Linien in der Darstellung des D3Q27-Geschwindigkeitssatzes dienen der Verdeutlichung der räumlichen Anordnung.

Durch diese Diskretisierung wird die Verteilungsfunktion zur Lattice-Verteilungsfunktion $f_i(\mathbf{x}, t)$, $i \in [0, q[$ die die Verteilung der Partikel mit dem zugehörigen Geschwindigkeitsvektor \mathbf{c}_i über das Gitter angibt. Sie ist das Kernelement der LBM. Die Boltzmann-Gleichung wird dabei zur sog. Lattice-Boltzmann-Gleichung, welche die Entwicklung dieser Lattice-Verteilungsfunktion über die diskretisierte Zeit beschreibt [5]:

$$f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t) + \Omega_i(\mathbf{x}, t). \quad (2.2-1)$$

Wie sich aus dieser Gleichung nun die Simulationsmethodik ergibt, wird im nächsten Abschnitt vorgestellt. Die Auswirkungen der Diskretisierungen auf die Genauigkeit der Methode werden in Abschnitt 2.8 näher beleuchtet.

Mit Hilfe der sog. Chapman-Enskog-Analyse [22] kann nun gezeigt werden, dass die Gleichung trotz der Diskretisierung sowohl die Navier-Stokes-Gleichungen als auch die fluiddynamische Kontinuitätsgleichung, also die Erhaltung der Masse, löst. Dies bestätigt, dass damit eine Simulation des Verhaltens von Fluiden möglich ist. [5]

Um schließlich Erkenntnisse aus den Simulationsergebnissen ziehen zu können, ist es natürlich auch erforderlich, aus den Verteilungsfunktionen makroskopische Größen zu ermitteln. Diese werden als Momente bezeichnet und nach ihrer Ordnung sortiert. Das sog. nullte Moment "nullter" Ordnung ist die lokale Dichte des Fluids an jedem Gitterpunkt. Sie kann als [5]:

$$\rho(\mathbf{x}, t) = \sum_i f_i(\mathbf{x}, t) \quad (2.2-2)$$

errechnet werden. Daraus kann dann auch der lokale Druck ermittelt werden. Dieser ergibt sich innerhalb der Simulation nach der isothermen Zustandsgleichung für Gase zu

$$p(\mathbf{x}, t) = p_0 + (\rho(\mathbf{x}, t) - \rho_0)c_s^2 \quad (2.2-3)$$

indem die Abweichung der Dichte von ihrem Mittelwert mit der Schallgeschwindigkeit skaliert wird [5].

Das erste Moment ist die lokale Impulsdichte des Fluids, sie kann als Summe der zulässigen Geschwindigkeitsvektoren, die mit den zugehörigen Verteilungsfunktionen gewichtet werden,

bestimmt werden [5]:

$$\mathbf{j}(\mathbf{x}, t) = \rho \mathbf{u}(\mathbf{x}, t) = \sum_i \mathbf{c}_i f_i(\mathbf{x}, t). \quad (2.2-4)$$

Teilt man diese nun durch die lokale Dichte, so ergibt sich zudem die lokale, makroskopische Strömungsgeschwindigkeit. Neben diesen beiden grundlegenden Momenten gibt es noch weitere höhere Momente, darunter den Spannungstensor als Moment 2. Ordnung.

Eine weitere wichtige Gleichung zur Lattice-Verteilungsfunktion ist die sog. Gleichgewichtsverteilung, die in den erläuterten Momenten mit der zugehörigen Verteilungsfunktion übereinstimmt. Sie beschreibt eine im Geschwindigkeitsraum isotropische Verteilung der Populationen um die makroskopische Geschwindigkeit \mathbf{u} . Die Verteilungen streben der Gleichgewichtsverteilung entgegen, da sich die Verteilung der Bewegungsrichtung der Partikel auf verschiedene Richtungen durch die Kollisionen nach und nach angleicht. Die Gleichgewichtsverteilung spielt daher in Lattice-Boltzmann-Kollisionsoperatoren (vgl. Abschn. 2.5) eine wichtige Rolle. Sie ist als [5, 23]:

$$f_i^{eq}(\mathbf{x}, t) = w_i \rho \left(1 + \frac{\mathbf{u} \cdot \mathbf{c}_i}{c_s^2} + \frac{(\mathbf{u} \cdot \mathbf{c}_i)^2}{2c_s^4} - \frac{\mathbf{u} \cdot \mathbf{u}}{2c_s^2} \right). \quad (2.2-5)$$

definiert. Dabei sind w_i die Gewichte, welche mit den einzelnen Geschwindigkeitsvektoren c_i verknüpft sind und c_s die Schallgeschwindigkeit. Um die Gleichgewichtsverteilung zu bestimmen sind dementsprechend lediglich \mathbf{u} und ρ erforderlich. Abweichungen der Verteilungsfunktion von der Gleichgewichtsverteilung

$$f_i^{neq} = f_i - f_i^{eq} \quad (2.2-6)$$

werden dabei auch als Nichtgleichgewichtsverteilung bezeichnet [5].

2.3 Zusammenhang von Realität und Simulation

Bei der LBM werden Größen innerhalb der Simulation meist in sog. Lattice-Einheiten (LU) angegeben. Dabei handelt es sich um dimensionslose Zahlen, die an die zuvor erläuterte Diskretisierung der LBM angepasst sind, so entspricht z. B. $\Delta t = 1$ in Lattice-Einheiten einem Zeitschritt der Simulation, welcher in physikalischen Einheiten bspw. $\Delta t = 0.01\text{s}$ entsprechen könnte. In diesem Abschnitt soll auf die Umrechnung der beiden Einheitensysteme eingegangen werden, wobei die Erklärung auf [5] basiert. Diese Umrechnung ist dabei wichtig, um die Aussage der Simulationsergebnisse auf reale Größen übertragen zu können. Es ist wichtig, dass beide Systeme klar getrennt bleiben, da sonst die Funktion der Simulation gestört wird. Im Folgenden werden die Lattice-Einheiten daher als *Einheit** markiert.

Die Entdimensionalisierung funktioniert, wie eine normale Einheitenumrechnung, mit Hilfe eines Proportionalitätsfaktors. Anstatt bspw. zwischen zwei Längeneinheiten umzurechnen:

$$l = 10 \text{ m} = 10l_0 = 10l_0' \frac{l_0}{l_0'} = 10 \text{ m} \frac{1 \text{ ft}}{0.3048 \text{ m}} \approx 32.18 \text{ ft} = 32.18l_0' \quad (2.3-1)$$

kann ebenso zwischen den Lattice-Größen und den physikalischen Einheiten (wie bspw. Meter) umgerechnet werden:

$$l = 10 \text{ m} = 10l_0 = 10l_0' \frac{1}{C_l} = 10 \text{ m} \frac{20}{1 \text{ m}} = 200 = 200l_0' = l^* \text{ mit } C_l = \frac{l_0}{l_0^*} \quad (2.3-2)$$

Alle bei der isothermen LBM relevanten Einheiten bestehen aus einer Kombination der

Einheiten von Masse, Zeit und Länge, daher kann mit drei unabhängigen Proportionalitätsfaktoren ein Umrechnungsfaktor für jede Einheit ermittelt werden. Da die Reynolds-Zahl, auf Grund der Ähnlichkeitstheorie, in der Simulation und außerhalb im Normalfall identisch ist, ergibt sich zudem, dass die Kombination mit einem Faktor von 1 stattfindet. Dementsprechend sind jeweils nur die passenden Exponenten der Faktoren zu finden:

$$C_q = C_1^{q_1} C_2^{q_2} C_3^{q_3} \quad (2.3-3)$$

In der LBM werden dabei aus praktischen Gründen häufig die Faktoren für Länge C_l , Geschwindigkeit C_u und Dichte C_ρ als grundlegende Faktoren verwendet. Der Faktor für die Umrechnung der Kraft ergibt sich dann bspw. als

$$C_F = [F] = \frac{\text{kgm}}{\text{s}^2} = \text{m}^2 \frac{\text{m}^2}{\text{s}^2} \frac{\text{kg}}{\text{m}^3} = C_l^2 C_U^2 C_\rho \quad (2.3-4)$$

Dass die jeweiligen Lattice-Bestandteile der Faktoren meist als 1 gewählt werden (vgl. Abschn. 2.9) erleichtert das Aufstellen der Umrechnungsfaktoren zusätzlich.

2.4 Algorithmus und Implementierung

Aus der in Abschnitt 2.2 vorgestellten Lattice-Boltzmann-Gleichung 2.2-1 ergeben sich direkt die beiden zentralen Schritte des Algorithmus der LBM: der Kollisions- und der Strömungsschritt, deren Auswirkungen in Abbildung 2.4-1 visualisiert sind.

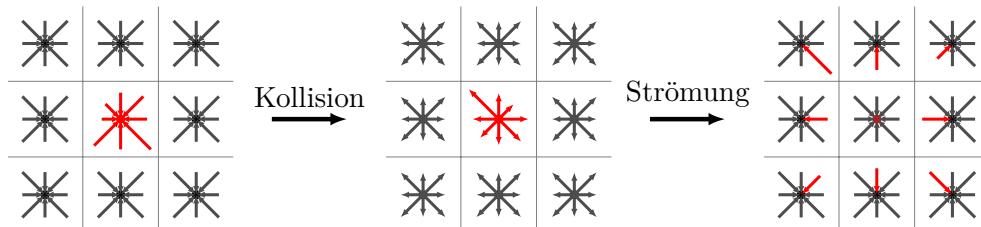


Abbildung 2.4-1: Visualisierung des Kollisions- und Strömungsschrittes der LBM mit D2Q9-Geschwindigkeitssatz. Die Länge der Pfeile stellt die Größe der Teilchenpopulation mit der Geschwindigkeit in dieser Richtung dar, der Fokus liegt auf dem rot eingefärbten Gitterpunkt in der Mitte. Im Kollisionsschritt werden die Populationen innerhalb des Gitterpunkts durch Kollisionen untereinander umverteilt, danach werden sie im Strömungsschritt entlang ihrer jeweiligen Geschwindigkeit einen Zeitschritt bewegt und landen so beim Nachbargitterpunkt. Darstellung ähnlich [24].

Der *Kollisionsschritt* beinhaltet die Anwendung des sog. Kollisionsoperators $\Omega_i(\mathbf{x}, t)$. Es handelt sich dabei um eine lokale Operation auf jedem Gitterpunkt, welche die Auswirkungen von Kollisionen zwischen den Partikeln des Fluids ausdrückt. Durch die Kollisionen ändern manche Partikel ihre Bewegungsrichtung und wechseln dadurch ihren zugeordneten Geschwindigkeitsvektor c_i und somit auch die Verteilung f_i . So werden durch den Kollisionsschritt die Partikelverteilungen an jedem Gitterpunkt etwas durchmischt. Nach dem Kollisionsschritt werden die Populationen als f_i^* bezeichnet. Im folgenden Abschnitt 2.5 wird der Kollisionsoperator näher erläutert. [5]

Der *Strömungsschritt* ergibt sich aus der geänderten Position von $f_i(\mathbf{x} + c_i \Delta t, t + \Delta t)$, wodurch ausdrückt wird, dass sich jede Population einen Zeitschritt lang mit der zugeordneten Geschwindigkeit bewegt hat. Da diese Geschwindigkeiten die Partikel in einem Zeitschritt stets zu einem benachbarten Gitterpunkt führen, wird der Strömungsschritt meist umgesetzt, indem alle Populationen zum entsprechenden Gitterpunkt kopiert werden. [5]

Neben diesen beiden zentralen Schritten beinhaltet der Algorithmus häufig einige weitere Zwischenschritte, die für den praktischen Einsatz erforderlich sind. Eine Auswahl möglicher

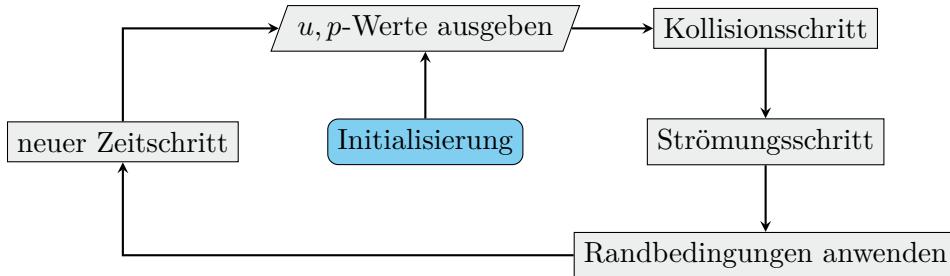


Abbildung 2.4-2: Übersicht über die zentralen Schritte des üblichen Algorithmus einer LBM-Implementierung, nach [5].

Zwischenschritte ist in Abbildung 2.4-2 dargestellt. Der erste dieser Schritte ist die Initialisierung, in welcher die Anfangswerte für die Simulation vorgegeben werden (vgl. Kap. 2.6). Zudem müssen in jedem Zeitschritt auch eventuelle Randbedingungen der Domäne an den entsprechenden Stellen angewendet werden (vgl. Kap. 2.7). Außerdem muss die Zeit vor jedem Schritt hochgezählt werden, um den Bezug von Zeit und Fluidverhalten aufrecht zu erhalten. Insgesamt kann mit einer Implementierung dieser Elemente eine Simulation nach der LBM durchgeführt werden, wobei für eine spätere Auswertung natürlich auch die Ausgabe der Ergebnisse nach jedem Zeitschritt erforderlich ist. In den folgenden Abschnitten werden einige dieser Bestandteile noch einmal genauer beleuchtet. Auf die Implementierung der LBM im Löser Lettuce, welcher in dieser Arbeit verwendet wird, wird in Abschnitt 3.1 eingegangen.

2.5 Kollisionsoperatoren

Im Kollisionsschritt wird die Interaktion der Fluid-Partikel untereinander über den sog. Kollisionsoperator simuliert. Dieser stellt die Auswirkungen elastischer Kollisionen zwischen den Fluid-Partikeln dar, durch welche die Verteilungen f_i an jedem Gitterpunkt etwas umverteilt werden. Dies kann man sich so vorstellen, dass durch die Kollisionen ein Teil der Partikel seine Bewegungsrichtung ändert. Dadurch besitzen diese Partikel nun einen anderen Geschwindigkeitsvektor c_i und gehören damit zu einer anderen Verteilung f_i . Da es sich um elastische Kollisionen handelt, muss der Operator Masse, Impuls und Energie erhalten. Es gibt verschiedene Kollisionsoperatoren mit unterschiedlichen numerischen Eigenschaften, so zum Beispiel den Bhatnagar-Gross-Krook-Kollisionsoperator (BGK) und den Karlin-Bösch-Chikatamarla-Kollisionsoperator (KBC). Als einfacherster und grundlegender Kollisionsoperator wird im Folgenden zunächst der BGK-Kollisionsoperator erklärt, bevor anschließend der komplexere KBC-Kollisionsoperator vorgestellt wird, welcher in den Simulationen in dieser Arbeit eingesetzt wird.

2.5.1 BGK

Der *BGK-Kollisionsoperator* ist definiert als [23]:

$$\Omega_i(f) = \frac{f_i - f_i^{eq}}{\tau} \Delta t. \quad (2.5.1-1)$$

Er beschreibt die Relaxation der Strömung mit einer Relaxationszeitkonstante $\tau = \frac{\nu}{c_s^2} + \frac{\Delta t}{2}$ entgegen der Gleichgewichtsverteilung 2.2-5. Durch seine Einfachheit und den geringen Rechenaufwand ist BGK ein häufig verwendeter Kollisionsoperator. Er bringt jedoch auch Nachteile bei der numerischen Stabilität und Genauigkeit unter bestimmten Umständen mit sich, weshalb er in einigen Gebieten von neueren Kollisionsoperatoren verdrängt wurde, die hier bessere Eigenschaften bieten. [5, 18, 25, 26]

2.5.2 KBC

Einer dieser neuen Kollisionsoperatoren ist der *KBC-Kollisionsoperator*, ein Multi-Relaxationszeit-Kollisionsoperator (MRT), der 2014 vorgestellt wurde und verbesserte Stabilität und Auflösung bei untaufgelösten Strömungen bietet [18, 19], was auch die Gründe für seine Anwendung in dieser Arbeit sind. Wie der Name schon sagt, zeichnen sich MRT-Kollisionsoperatoren dadurch aus, dass sie mehrere verschiedene Relaxationszeiten für Momente verschiedener Ordnung verwenden, was in mehr Parametern für die Stabilitätsoptimierung resultiert. Aufgrund der Komplexität des Operators und der zugrundeliegenden Konzepte wird an dieser Stelle nur eine grundlegende Erklärung gegeben, weitere Details können jedoch [18, 19] entnommen werden.

Wie bei MRT-Kollisionsoperatoren üblich, ist auch bei KBC der erste Schritt, die Werte der Verteilungsfunktion f mit Hilfe einer Matrixmultiplikation $\mathbf{m} = \mathbf{M}f$ mit einer Transformationsmatrix \mathbf{M} als \mathbf{m} in einen Momentenraum zu überführen. Anschließend können die Momente sortiert oder mit je einer eigenen Relaxationszeitkonstante gegen die Gleichgewichtsverteilung relaxiert werden. Anschließend werden sie mit Hilfe von \mathbf{M}^{-1} wieder in die Verteilungsfunktion zusammengefügt. KBC nutzt diese Möglichkeit, um die Verteilungsfunktion in drei Teile zu teilen:

$$f_i = k_i + s_i + h_i. \quad (2.5.2-1)$$

Der kinetische Anteil (k_i) beinhaltet die Momente "nullter" und erster Ordnung, also Dichte und Impulsdichte, der Scherteil (s_i) beinhaltet die Momente zweiter Ordnung, also des Spannungstensors, und der dritten Teil (h_i) beinhaltet die weiteren Momente höherer Ordnung, die in den anderen Anteilen nicht enthalten sind. Nun können diese Anteile mit unterschiedlichen Relaxationszeitkonstanten entgegen ihrer jeweiligen Gleichgewichtsverteilung relaxiert werden [18]:

$$\Omega_i(f) = -\beta(2(s_i - s_i^{eq}) + \gamma(h_i - h_i^{eq})). \quad (2.5.2-2)$$

k_i taucht hier nicht auf, da es mit Dichte und Impulsdichte lokal erhaltene Felder repräsentiert und daher unverändert bleibt. Da $\beta = \frac{1}{2\tau}$, wird der Scherteil hier s_i genauso behandelt, wie bei Verwendung des BGK-Operators die gesamte Population. Die Momente höherer Ordnung, die in h_i zusammengefasst sind, werden mit:

$$\gamma = \frac{1}{\beta} - (2 - \frac{1}{\beta}) \frac{\sum_i \frac{\Delta s_i - \Delta h_i}{f_i^{eq}}}{\sum_i \frac{\Delta h_i - \Delta h_i}{f_i^{eq}}} \quad (2.5.2-3)$$

relaxiert. Dabei handelt es sich um eine dynamische Relaxationszeit, welche durch Minimieren der Entropie an jedem Gitterpunkt in jedem Zeitschritt neu ermittelt wird. So optimiert der Operator die zusätzlichen Relaxationszeiten automatisch, wodurch dem Anwender umfangreiches manuelles Tuning erspart wird. [18]

2.6 Anfangsbedingungen

Wie in Abschnitt 2.4 erwähnt, beginnt eine Simulation mit der LBM mit der Initialisierung. Die Aufgabe der Initialisierung ist es, aus den gewünschten makroskopischen Anfangsbedingungen die Verteilungsfunktion zu errechnen. Der genaue Prozess, der für die Initialisierung verwendet wird, ist dabei vor allem wichtig, wenn sich die Strömung aus der Initialisierung ergibt, wie bei vordefinierten Turbulenzen. Bei langfristigen Simulationen können eventuelle unphysikalische Effekte der Initialisierung über die Dauer der Simulation abklingen, während die physikalische Lösung verbleibt. [5]

Der einfachste Ansatz für die Initialisierung ist die Gleichgewichtsverteilung nach Gleichung 2.2-5. Dazu wird die aus den gewünschten Werten für ρ und \mathbf{u} berechnete Gleichgewichtsverteilung f^{eq} als Anfangsverteilung verwendet. Dabei werden häufig die Werte $\rho = 1$ und $\mathbf{u} = 0$ gewählt. [5]

Neben diesen Ansätzen gibt es noch weitere komplexere Initialisierungen, auf diese wird hier jedoch nicht weiter eingegangen, da die geplanten Simulationen als langfristige Untersuchungen davon nur hinsichtlich der Dauer bis zum Erreichen des eingeschwungenen Zustandes profitieren. [5]

2.7 Randbedingungen

In diesem Abschnitt soll auf die Randbedingungen für die LBM eingegangen werden. Diese sind essentiell, um mit der Simulation realistische Anwendungen abilden zu können. Sie erlauben es bspw. die Interaktion des Fluids mit einem Körper zu ermöglichen, oder die Simulationsdomäne sinnvoll abzuschließen. Sie sind dabei bei der LBM von besonderer Bedeutung, da sie die Teilchen-Populationen, welche in die Domäne einströmen, vorgeben, die aus der Lattice-Boltzmann-Gleichung nicht ermittelt werden können. [5]

In der kontinuierlichen Flüssigkeitsdynamik werden drei Arten von Randbedingungen unterschieden [5, 14]:

1. Die *Dirichlet-Randbedingung* für die Strömungsgeschwindigkeit gibt die Geschwindigkeit des Fluids als $\mathbf{u}(x_b, t) = \mathbf{u}_w(\mathbf{x}_w, t)$ vor, d.h. die Geschwindigkeit des Fluids $\mathbf{u}(x_b, t)$ entspricht der Geschwindigkeit des Rands $\mathbf{u}_w(\mathbf{x}_w, t)$ an der selben Stelle. So lassen sich stillstehende oder bewegte Wände realisieren, die vom Fluid nicht durchdrungen werden. Dabei ist wichtig, dass es sich um eine sog. Haftbedingung handelt, die Strömung sich also auch orthogonal zum Körper nicht anders bewegt als der Körper und sich so eine Grenzschicht ausbilden kann.
2. Die *Neumann-Randbedingung* für die Strömungsgeschwindigkeit $\mathbf{n}\sigma(\mathbf{x}_b, t) = \mathbf{T}_b(\mathbf{x}_b, t)$ besagt, dass der Gesamtspannungstensor am Rand dem vorgegebenen Traktionstensor entspricht.
3. Die *Robin-Randbedingung*, welche eine Linearkombination der Dirichlet- und Neumann-Bedingung ist.

Die Lattice-Boltzmann-Gleichung bzw. LBM besitzt ähnliche Randbedingungen. Da sie aber die Entwicklung der Teilchen-Populationen beschreibt, reicht es nicht aus, makroskopische Größen von Interesse wie Druck und Geschwindigkeit vorzugeben. Die Randbedingungen müssen hier die Verteilungen f_i , welche von außerhalb in die Simulationsdomäne eintreten, vorgeben. Diese können mit der Lattice-Boltzmann-Gleichung nicht berechnet werden, sind aber erforderlich, damit die Simulation funktionieren kann [5]. Dementsprechend ist es die Aufgabe der Randbedingungen, aus den vorgesehenen makroskopischen Größen die einströmenden Populationen zu ermitteln. Da diese Umrechnung ist nicht eindeutig ist, gibt es für die LBM eine Vielzahl von Randbedingungen mit mehr oder weniger unterschiedlichen Eigenschaften. Ein weiterer Unterschied zu den kontinuierlichen Methoden ist, dass die Randbedingungen bei der LBM nicht nur eine Vorgabe für die Lösung einer Gleichung sind, sondern ein aktiver Teil des Simulationsalgorithmus werden, der in jedem Zeitschritt seinen Beitrag zur Annäherung an die Lösung leisten muss (vgl. Abschn. 2.4). [5]

Randbedingungen müssen alle Gitterpunkte betreffen, bei denen ein benachbarter Gitterpunkt außerhalb der Domäne liegt bzw. liegen würde. Hier ist es erforderlich, die entsprechenden Geschwindigkeiten c_i von diesen Gitterpunkten durch eine Randbedingung zu bestimmen, da diese sonst unbekannt bleiben. Die Randbedingungen sind dabei, obwohl sie nur einen kleinen Teil der Simulationsdomäne direkt betreffen, von großer Bedeutung für die Genauigkeit

des Ergebnisses. Eine falsche Wahl der Randbedingungen kann die Genauigkeit der Lösung in der gesamten Domäne verschlechtern oder zu Instabilität führen. Es wird daher empfohlen, Randbedingungen zu verwenden, deren Fehler-Ordnung höher ist als die der Methodik für die normalen Fluid-Gitterpunkte. [5]

Im Detail lassen sich mit den Randbedingungen eine Vielzahl von verschiedenen Situationen darstellen:

- Wiederholte Geometrien können mit Hilfe einer sog. *periodischen Randbedingung* dargestellt werden. Diese funktioniert, indem die Verteilungen, welche die Domäne an der einen Seite verlassen an der gegenüberliegenden, welche ebenfalls eine periodische Randbedingung sein muss, wieder einströmen. Dadurch verhält sich die Strömung als gäbe es in der entsprechenden Richtung noch eine identische Simulationsdomäne, die aber nicht simuliert werden muss. Es ist zu beachten, dass die Lösung für die Strömung dadurch nicht automatisch auch periodisch wird. Dementsprechend kann diese Randbedingung nur periodische Strukturen, oder Strukturen die kleiner als die Simulationsdomäne sind, darstellen. [5]
- Bereiche, in denen keine Veränderung an der Strömung mehr auftritt, können mit Hilfe einer *Zero-Gradient-Randbedingung* dargestellt werden. Die Randbedingung funktioniert, indem sie die einströmenden Populationen am benachbarten Gitterpunkt übernimmt.
- Festkörper im oder am Rand des Fluids
- Strömung des Fluids durch die Randbedingung mit einem vorgegebenen Druck
- Strömung des Fluids durch die Randbedingung mit einer vorgegebenen Geschwindigkeit

Die letzten drei Situationen können dabei mit einer Vielzahl verschiedener Randbedingungen dargestellt werden. Im Folgenden werden mit der Bounce-Back-Randbedingung und der Equilibrium-Randbedingung zwei Grundlegende Beispiele vorgestellt.

Bounce-Back-Randbedingung

Die Bounce-Back-Randbedingung ist eine sehr häufig verwendete Randbedingung. Ihre grundlegende Funktionsweise ist, dass Populationen, die auf die Randbedingung treffen dahin reflektiert werden, wo sie herkommen. Dazu gibt es zwei Vorgehensweisen:

- Die sog. *Fullway-Bounce-Back-Randbedingung* [27] arbeitet mit Populationen, die in einen Gitterpunkt außerhalb der Domäne geströmt sind. Diese werden hier invertiert, indem sie jeweils von f_i zum gegenüberliegenden $f_{\bar{i}}$ konvertiert werden. Im nächsten Strömungsschritt kehren sie dann wieder in die Domäne zurück.
- Alternativ gibt es die sog. *Halfway-Bounce-Back-Randbedingung* [28], welche die Populationen nach $f_{\bar{i}}(\mathbf{x}_b, t + \Delta t) = f_i^*(\mathbf{x}_b, t)$ sofort reflektiert, d.h. also anstatt Populationen zunächst aus der Domäne strömen zu lassen, wird der Strömungsschritt für die betreffenden Populationen durch das Invertieren ersetzt.

Die Halfway-Bounce-Back-Randbedingung besitzt dabei eine bessere zeitliche Genauigkeit, da die Populationen direkt reflektiert werden, und nicht zuerst einen Zeitschritt außerhalb der Domäne verbringen. Dadurch eignet sie sich besser für unstetige Strömungen. Da bei beiden Varianten keine Strömung entlang der Randbedingung auftreten kann und die einströmenden Populationen direkt vorgegeben werden, handelt es sich hierbei um eine Umsetzung der Dirichlet-Randbedingung. [5]

Während als Anwendungsfall zunächst nur die Darstellung von unbewegten Festkörpern naheliegend scheint, kann mit Hilfe eines entsprechenden Korrekturterms auch der Einfluss bewegter Körper auf die reflektierten Populationen dargestellt werden:

$$f_i(\mathbf{x}_b, t + \Delta t) = f_i^*(\mathbf{x}_b, t) - 2w_i\rho_w \frac{\mathbf{c}_i \cdot \mathbf{u}_w}{c_s^2} \quad (2.7-1)$$

dabei ist u_w die Geschwindigkeit der Wand und ρ_w die Dichte des Fluids vor der Wand. Neben unbewegten Körpern können hiermit auch offene Randbedingungen mit Strömungen vorgegebener Geschwindigkeit erzeugt werden. Mit einem entsprechenden anderen Korrekturterms kann in ähnlicher Weise eine offene Randbedingung erzeugt werden, welche den Druck des Fluids vorgibt. [5]

Die Bounce-Back-Randbedingung hat dabei den Vorteil guter numerischer Stabilität und garantiert bei der unbewegten Wand auch einen exakten Masseerhalt der reflektierten Populationen. Sie bietet jedoch nur eine Fehlerordnung von 1 bis 2, die von der Viskosität und anderen Simulationsparametern abhängt. Aufbauend auf der Bounce-Back-Randbedingung gibt es noch einige komplexere Randbedingungen, die bspw. höhere Fehlerordnungen bieten können. [5]

Equilibrium-Randbedingung

Die zweite grundlegende Randbedingung ist die sog. *Equilibrium-Randbedingung* [5, 29]. In ihrer einfachsten Form funktioniert sie, indem sie die Populationen der Gitterpunkte am Rand auf die Werte der Gleichgewichtsverteilung setzt, die aus den makroskopischen Größen errechnet werden:

$$f_i(\mathbf{x}_b, t) = f_i^{eq}(\rho_b, \mathbf{u}_b) \quad (2.7-2)$$

Die Equilibrium-Randbedingung besitzt dabei, außer unter speziellen Bedingungen, nur eine Genauigkeit 1. Ordnung. Im Gegenzug besitzt sie jedoch eine sehr gute Stabilität. Aufbauend auf der Equilibrium-Randbedingung gibt es bspw. die Non-Equilibrium-Extrapolation-Randbedingung [30], die den errechneten Werten der Gleichgewichtsverteilung die Nichtgleichgewichtsanteile der benachbarten Gitterpunkte hinzufügt. [5]

Einige dieser Möglichkeiten werden auch im Rahmen dieser Arbeit eingesetzt. Darauf wird im Abschnitt 3.2.2 näher eingegangen.

2.8 Genauigkeit der Methode

Durch die Diskretisierungen und weitere Einflüsse entstehen Abweichungen zwischen den Ergebnissen der LBM und den Lösungen der Navier-Stokes-Gleichungen. Die Herkunft und Ordnung dieser Fehler sollen in diesem Abschnitt kurz zusammengefasst werden. Die Ausführungen basieren dabei vor allem auf [5].

Der erste Fehler ist der sog. *Diskretisierungsfehler*, der durch die erläuterte Diskretisierung der Gleichungen zustande kommt. Die Änderung des Fehlers aus der Zeitdiskretisierung kann mit $\mathcal{O}(\Delta t^2)$ abgeschätzt werden. Das bedeutet, dass der Fehler sich bei konstantem Δx mit Δt^2 ändert. Der Fehler aus der räumlichen Diskretisierung wird ebenso mit $\mathcal{O}(\Delta x^2)$ geschätzt, dementsprechend ändert sich der Fehler, wenn $\nu\Delta t/\Delta x^2$ konstant ist, mit Δx^2 . Diese Fehler besitzen über diese pauschale Angabe hinaus jedoch auch eine Abhängigkeit von der Relaxationszeit τ und müssen für eine genaue Angabe entsprechend für die einzelnen Fehler und Kollisionsoperatoren hergeleitet werden.

Neben diesen Diskretisierungsfehlern wird die Genauigkeit der LBM zusätzlich durch den sog. *Modellierungsfehler* beeinträchtigt, welcher durch die Diskretisierung des Geschwindigkeitsraumes entstehen. Es handelt sich um einen Fehler der Ordnung $\mathcal{O}(u^3)$ in einem der

höheren Momente, wodurch die LBM mit Standard-Geschwindigkeitssätzen, wie D2Q9 oder D3Q27, keinen thermischen Energietransport ausdrücken kann und auf die Simulation der schwach-kompressiblen Navier-Stokes-Gleichungen limitiert ist.

Ein weiterer Fehler entsteht, da das Equilibrium, welches für Fluidodynamik-Simulationen mit der LBM eingesetzt wird (siehe Gleichung 2.2-5) die kompressiblen Navier-Stokes-Gleichungen wiedergibt. Da die LBM jedoch auf Grund der Begrenzung auf den schwach-kompressiblen Bereich durch den Modellierungsfehler meist für die Simulation der inkompressiblen Navier-Stokes-Gleichungen (vgl. Abschn. 2.1) eingesetzt wird, ergibt sich hier der sog. *Kompressibilitätsfehler* der Ordnung $\mathcal{O}(\text{Ma}^2)$. Dieser ist dabei unabhängig von dem verwendeten Gitter, sodass er auch bei beliebig feinem Gitter bzw. Δx bestehen bleibt.

Zusammenfassend lässt sich sagen, dass die LBM die Navier-Stokes-Gleichungen mit einer Genauigkeit 2. Ordnung in Raum und Zeit approximiert. Zusätzlich weist sie jedoch weitere Fehler $\mathcal{O}(u^3)$ und $\mathcal{O}(\text{Ma}^2)$ auf. Diese Fehler gelten so jedoch nur ohne einschränkende Rand- und Anfangsbedingungen, welche die Ordnung mancher Fehler reduzieren oder weitere Fehler hinzufügen können. Bei praktischen Simulationen fällt es durch diesen Einfluss häufig schwer, einen Fehler analytisch zu bestimmen.

2.9 Parametrierung der Simulation

Die LBM besitzt eine Vielzahl von Parametern, die es richtig zu wählen gilt, um ein Optimum aus Genauigkeit, Stabilität und Rechenaufwand zu erzielen. Im Folgenden werden zunächst die einzelnen Parameter vorgestellt. Anschließend wird darauf eingegangen, wie welche Parameter die genannten Performanz-Aspekte beeinflussen und schließlich eine Methode zur Auswahl der Parameter vorgestellt. Die Erläuterungen basieren dabei auf [5].

2.9.1 Parameter der Lattice-Boltzmann-Methode

Die wichtigsten Parameter der Simulation sind die physikalischen Parameter des zu simulierenden Fluids. So sollten die Reynolds-Zahl, die charakteristische Länge, die Viskosität, die charakteristische Dichte und der charakteristische Druck bekannt sein. Über die Parameter des Fluids hinaus wird die entsprechende LBM-Simulation jedoch von einer Vielzahl weiterer Parameter charakterisiert, von denen hier eine Auswahl vorgestellt werden soll:

- Die Reynolds-Zahl Re stellt auf Grund der Ähnlichkeitstheorie die Vergleichbarkeit der Ergebnisse in beiden Einheitensystemen sicher. Sie stellt zudem einen Zusammenhang zwischen einigen der folgenden Parameter da, wie in den folgenden Abschnitten erläutert wird.
- Die Gitterkonstante Δx bezeichnet den Abstand der Gitterpunkte in physikalischen Einheiten bzw. m. Die zugehörige Gitter-Größe Δx^* wird dabei meist auf 1 festgelegt.
- Die Zeitschrittweite Δt bezeichnet die physikalische Dauer eines Zeitschritts in s. Die zugehörige Gitter-Größe Δt^* wird auch hier meist auf 1 festgelegt. So vergehen bis zum n-ten Zeitschritt auch n-mal Δt .
- Die Relaxationszeit τ oder τ^* in s oder LU ist, wie näher im Abschnitt 2.5 beschrieben, ein wichtiger Parameter vieler Kollisionsoperatoren und beeinflusst damit einen sehr zentralen Schritt der LBM.
- Die Dichte ρ^* in LU wird meist auf ihre Schwankung um den mittleren Wert $\rho_0^* = 1$ reduziert. Dieser muss nicht mit einer physikalischen Referenzdichte übereinstimmen, da er nur bei der Simulation von Wärmeübertragung von Bedeutung ist.

- Der physikalische Druck p wird als $p = p_0 + p' = p_0 + p'^* C_p$ mit $p'^* = c_s^* \rho'^*$ aus der Dichte errechnet. Wie bei der Dichte ist auch hier bei der Standard-LBM ohne Wärmeübertragung nur die Variation um den frei gewählten Referenzdruck von Bedeutung.
- Die Gitter-Schallgeschwindigkeit c_s^* beträgt in der Standard-LBM $c_s^* = 1/\sqrt{3}$. Damit die Simulation quasi-inkompressibel bleibt, müssen alle anderen Geschwindigkeiten U^* deutlich kleiner (mind. < 20%) sein.
- Die Machzahl drückt das Verhältnis der charakteristischen Geschwindigkeit u_0 und der Schallgeschwindigkeit aus. Die charakteristische Geschwindigkeit ist dabei häufig die Geschwindigkeit der Anströmung. Dieser Zusammenhang gilt sowohl in Lattice- als auch in physikalischen Einheiten.
- Die kinematische Viskosität ν steht mit den anderen Parametern über $\nu = c_s^{*2}(\tau^* - \frac{1}{2})\frac{\Delta x^2}{\Delta t}$ im Verhältnis. So wird bei Vorgabe der Viskosität die Wahl von τ^* , Δx und Δt eingeschränkt. Sie ist dabei über die entsprechende Einheitenumrechnung auch mit ihrem Pendant in Lattice-Einheiten verknüpft.

2.9.2 Performanz-Aspekte und ihre Einflussgrößen

Die Wahl dieser Parameter wirkt sich auf die Stabilität und den Rechenaufwand der Simulation sowie auf die Genauigkeit ihrer Ergebnisse aus. In diesem Abschnitt soll die jeweilige Beeinflussung kurz vorgestellt werden, da ihre Kenntnis die Grundlage für eine sinnvolle Wahl der Parameter bildet.

Genauigkeit

Nachdem auf die Genauigkeit im Abschnitt 2.8 bereits näher eingegangen wurde, werden die hier relevanten Fehler und Parameter nur kurz zusammengefasst. Die Genauigkeit hängt vor allem von drei Fehlern ab:

- Dem räumlichen Diskretisierungsfehler, der mit Δx^2 skaliert
- Dem zeitlichen Diskretisierungsfehler, der mit Δt^2 skaliert
- Dem Kompressibilitätsfehler, der mit $\text{Ma}^2 \propto u_0^*{}^2$ skaliert und entsprechend bei gleicher physikalischer Geschwindigkeit mit $1/C_u^2 = C_t^2/C_l^2$ und somit, bei entsprechender Wahl der LU-Anteile der Faktoren als 1, mit $\Delta t^2/\Delta x^2$

Wie hier zu erkennen, hängen die verschiedenen Fehler im Endeffekt vor allem von der Wahl der Parameter Δx und Δt ab. Durch den Kompressibilitätsfehler reicht es dabei jedoch nicht, beide so klein wie möglich zu wählen, ihr Verhältnis spielt ebenfalls eine Rolle. Eine gängige Vorgehensweise ist die Nutzung der sog. diffusiven Skalierung, bei der man $\Delta t \propto \Delta x^2$ festlegt. So wird garantiert, dass der jeweils größte der Fehler mit Δx^2 skaliert.

Stabilität

Die Stabilität der LBM ist ein komplexes Thema, nicht zuletzt da ihre Abhängigkeit von den Simulationsparametern sich je nach Kollisionsoperator unterscheidet. Beim BGK-Kollisionsoperator gilt als hinreichende Stabilitätsbedingung, dass die Simulation stabil ist, solange $\tau^*/\Delta t > 1/2$ und $f_i^{eq} \geq 0$. Aus dieser und weiteren Bedingungen können dann mit verschiedenen Abhängigkeiten Richtwerte für die maximal zulässige makroskopische Geschwindigkeit bei der jeweiligen Relaxationszeit errechnet werden. Diese Kriterien berücksichtigen jedoch nur die Möglichkeit des Entstehens einer Instabilität in der offenen Domäne, häufig sind jedoch auch Randbedingungen Ursache von Instabilität. [5]

Bei MRT-Kollisionsoperatoren, wie dem in dieser Arbeit eingesetzten KBC-Kollisionsoperator, fällt es noch schwerer die Stabilität zu analysieren, da sie durch die zusätzlichen Relaxationszeiten mehr Freiheitsgrade besitzen [5]. Beim KBC-Kollisionsoperator sind die genauen Zusammenhänge bisher unbekannt, experimentelle Untersuchungen ergeben jedoch, dass der Operator eine bessere Stabilität besitzt als der BGK-Kollisionsoperator [18].

Effizienz

Der Rechenaufwand einer LBM-Simulation hängt vor allem von der Anzahl an erforderlichen Gitterpunkt-Updates ab. Diese ergibt sich als $N_s N_t$, also aus der Multiplikation der Anzahl an Zeitschritten und Gitterpunkten, da jeder Gitterpunkt pro Zeitschritt einmal geupdated werden muss. Dabei gilt $N_s \propto 1/\Delta x^d$, wobei d die Anzahl der Dimensionen bezeichnet und $N_t \propto 1/\Delta t$. Dementsprechend gilt für die benötigte Rechenzeit T und den Speicherbedarf M :

$$T \propto \frac{1}{\Delta x^d \Delta t} \quad M \propto \frac{1}{\Delta x^d} \quad (2.9.2-1)$$

Insgesamt sollten zum Senken der Rechenzeit Δx und Δt also so groß gewählt werden, wie möglich bzw. eine möglichst geringe Auflösung von Simulationsdomäne- und Dauer angestrebt werden.

2.9.3 Strategie zur Parameterwahl

Abschließend soll nun eine von [5] empfohlene Strategie zur Auswahl der Simulationsparameter, die einen guten Mittelweg aller drei Performanz-Charakteristiken verfolgt, vorgestellt werden. Wie bereits eingangs erwähnt, sollten die physikalischen Parameter wie Reynolds-Zahl Re , charakteristische Länge l_0 , Geschwindigkeit u_0 , Druck p_0 und Dichte ρ_0 und Viskosität ν für die zu untersuchenden Situation festgelegt worden sein.

Anschließend wird festgelegt, welche der dimensionslosen Kennzahlen in beiden Systemen gleich sein sollen. Außer bei sehr kleinen Reynolds-Zahlen (z. B. $Re < 10^{-2}$) muss die Reynolds-Zahl entsprechend der Ähnlichkeitstheorie in beiden Einheitensystemen gleich sein, um eine Vergleichbarkeit zu ermöglichen. Die Machzahl $Ma = \mathbf{u}_0^*/c_s^*$ braucht bei der Simulation inkompressibler Fluide, solange ihr Wert klein bleibt ($Ma < 0.3$), nicht mit der physikalischen Machzahl übereinzustimmen. Der Unterschied beider Machzahlen kann dann genutzt werden, um die Simulation zu beschleunigen, indem damit die Zeitschrittweite bzw. Δt vergrößert wird.

Basierend auf der Reynolds- und Machzahl können nun einige Parameter ins Verhältnis gesetzt werden:

$$Re = \frac{l_0^* \mathbf{u}_0^*}{\nu^*} = \frac{l_0^* \mathbf{u}_0^*}{c_s^{*2} (\tau^* - \frac{1}{2})} \Rightarrow \frac{Re}{Ma} = \frac{l_0}{c_s^{*2} (\tau^* - \frac{1}{2}) \Delta x} \quad (2.9.3-1)$$

wobei $l_0^* = l_0 / \Delta x$ eine charakteristische Längenskala (wie bspw. die Anzahl an Gitterpunkten entlang einer Koordinatenrichtung) bezeichnet. Dieses Verhältnis erlaubt es, bei gegebener Reynolds- und Machzahl τ und Δx auszubalancieren.

Um die weiteren Parameter sinnvoll zu wählen, empfiehlt [5] folgendes Vorgehen:

1. Dichte in LU ρ_0^* festlegen. Da sie außer ihrem Anteil an einigen Umrechnungsfaktoren keinen Einfluss hat, wird sie meist auf 1 gesetzt. Somit ergibt sich der Umrechnungsfaktor der Dichte zu $C_\rho = \rho_0 / \rho_0^* = \rho_0$
2. Gitterkonstante Δx so festlegen, dass die resultierende Größe der Simulationsdomäne die Kapazitäten des Computers fast vollständig ausgenutzt, um die bestmögliche Genauigkeit zu erreichen. Damit wird zugleich auch der Umrechnungsfaktor der Länge als

$$C_l = \Delta x / \Delta x^* = \Delta x \text{ festgelegt.}$$

3. Gitter-Machzahl im Rahmen der verschiedenen Verhältnisse sinnvoll festlegen, wie bereits erläutert kann sie bei der Simulation inkompressibler Fluide unabhängig von der physikalischen Machzahl gewählt werden. Um eine sinnvolle Genauigkeit zu erreichen sollte sie nicht höher als 0.3 gewählt werden, für bessere Genauigkeit wird 0.1 oder weniger empfohlen, wobei eine kleinere Machzahl auch den Zeitschritt verringert und so eine größere Anzahl an Zeitschritten und damit eine längere Simulationsdauer nötig macht. In Kombination mit der zuvor bekannten physikalischen charakteristischen Geschwindigkeit wird hier zugleich auch der Umrechnungsfaktor der Geschwindigkeit als $C_u = \frac{u_0}{u_0^*} = \frac{u_0}{Ma c_s^*}$ festgelegt.

4. Relaxationszeitkonstante τ^* über

$$\tau^* = \frac{1}{2} + \frac{l_0^* \mathbf{u}_0^*}{c_s^{*2} \Delta x^* \text{Re}} \quad (2.9.3-2)$$

aus der festgelegten charakteristischen Länge $l_0^* = l_0 / \Delta x$ und Geschwindigkeit $\mathbf{u}_0^* = Ma \cdot c_s^*$ ermitteln. Bei Nutzung des BGK-Kollisionsoperator liegt τ^* im Optimalfall nahe 1, aber nie unter 0.5.

5. Mit Hilfe der Verhältnisse in Gleichung 2.9.3-1 wird nun sichergestellt, dass in der offenen Domäne Stabilität gegeben ist. Falls τ^* zu klein ist kann Δx reduziert (mehr Rechenaufwand) oder Ma erhöht werden (schlechtere Genauigkeit).
6. die Dauer eines Zeitschritts kann schließlich als $\Delta t = \frac{C_l}{C_u} = \frac{l_0/l_0^*}{u_0^*/u_0^*}$ aus den anderen Parametern bestimmt werden.

So sind schließlich alle nötigen Parameter festgelegt, wobei so insgesamt die bestmögliche Genauigkeit, bei akzeptablem Rechenaufwand und ausreichender Stabilität erreicht werden sollte. Es gibt noch weitere Abfolgen, sinnvolle Simulationsparameter aus den physikalischen Parametern und Systembeschränkungen zu ermitteln, auf diese wird hier nicht näher eingegangen, sie sind aber in [5] zu finden. Soll nach Festlegung der Parameter noch einmal die Gitterkonstante Δx geändert werden, so sollte hier die diffusive Skalierung $\Delta t \propto \Delta x^2$ angewendet werden.

3 Die LBM mit dem Löser Lettuce

In diesem Kapitel soll die Implementierung der LBM im verwendeten Löser Lettuce vorgestellt werden. Dabei wird zunächst der Löser selbst vorgestellt, bevor anschließend auf Erweiterungen, welche im Rahmen der Arbeit hinzugefügt werden, eingegangen wird.

3.1 Der Löser Lettuce



Abbildung 3.1-1: Das Logo des LBM-Lösers Lettuce [31]

In dieser Arbeit wird der LBM-Löser *Lettuce* eingesetzt, welcher in diesem Abschnitt kurz vorgestellt werden soll. Der Fokus bei der Entwicklung des Lösers wird auf schnelles Prototyping und damit leichte Zugänglichkeit und Modularität gelegt, sodass der Nutzer sich eine Simulation schnell aus bereits implementierten Elementen zusammenstellen, aber auch leicht eigene hinzufügen kann. Lettuce ist dabei auf GitHub frei zugänglich [31]. Der Löser wurde und wird dabei bereits in einigen Arbeiten eingesetzt, somit ist seine generelle Funktion bereits validiert [32, 33, 34]. Im Folgenden wird kurz auf seinen Aufbau eingegangen und die Verwendung anhand eines Beispiels erläutert.

3.1.1 Aufbau des Lözers

Zunächst soll der Aufbau des Lözers erläutert werden. Dazu werden zunächst die verwendete Sprache und Bibliotheken vorgestellt und dann näher auf den Löser selbst eingegangen.

Um das leichte Prototyping zu ermöglichen, ist der Löser in der nutzerfreundlichen Sprache Python geschrieben, welche es erlaubt bereits mit wenig Code viel zu bewirken. Um dennoch die nötige Performanz zu erreichen, basieren viele der Rechenoperationen auf den Tensormathematik-Modulen der Bibliothek Pytorch [35, 36], die durch die integrierte CUDA-Bibliothek auch Hardware-Beschleunigung auf NVIDIA-Grafikkarten (GPU) ermöglicht. Da Pytorch primär für das Trainieren von neuronalen Netzen gedacht ist und so auch automatisches Differenzieren integriert, sind diese Funktionen ebenfalls leicht mit Lettuce kombinierbar. Für das Vorgeben von Anfangswerten und Masken wird zudem die Mathematik-Bibliothek Numpy verwendet, da die meisten Python-Nutzer mit dieser vertraut sein sollten.

Lettuce selbst besteht dabei aus einer Vielzahl von Klassen, die durch ihre Interaktion die Gesamtfunktion realisieren. Für viele der Klassen gibt es dabei ein festgelegtes Interface, sodass sie unabhängig von den anderen austauschbar sind, wodurch sich die Modularität des

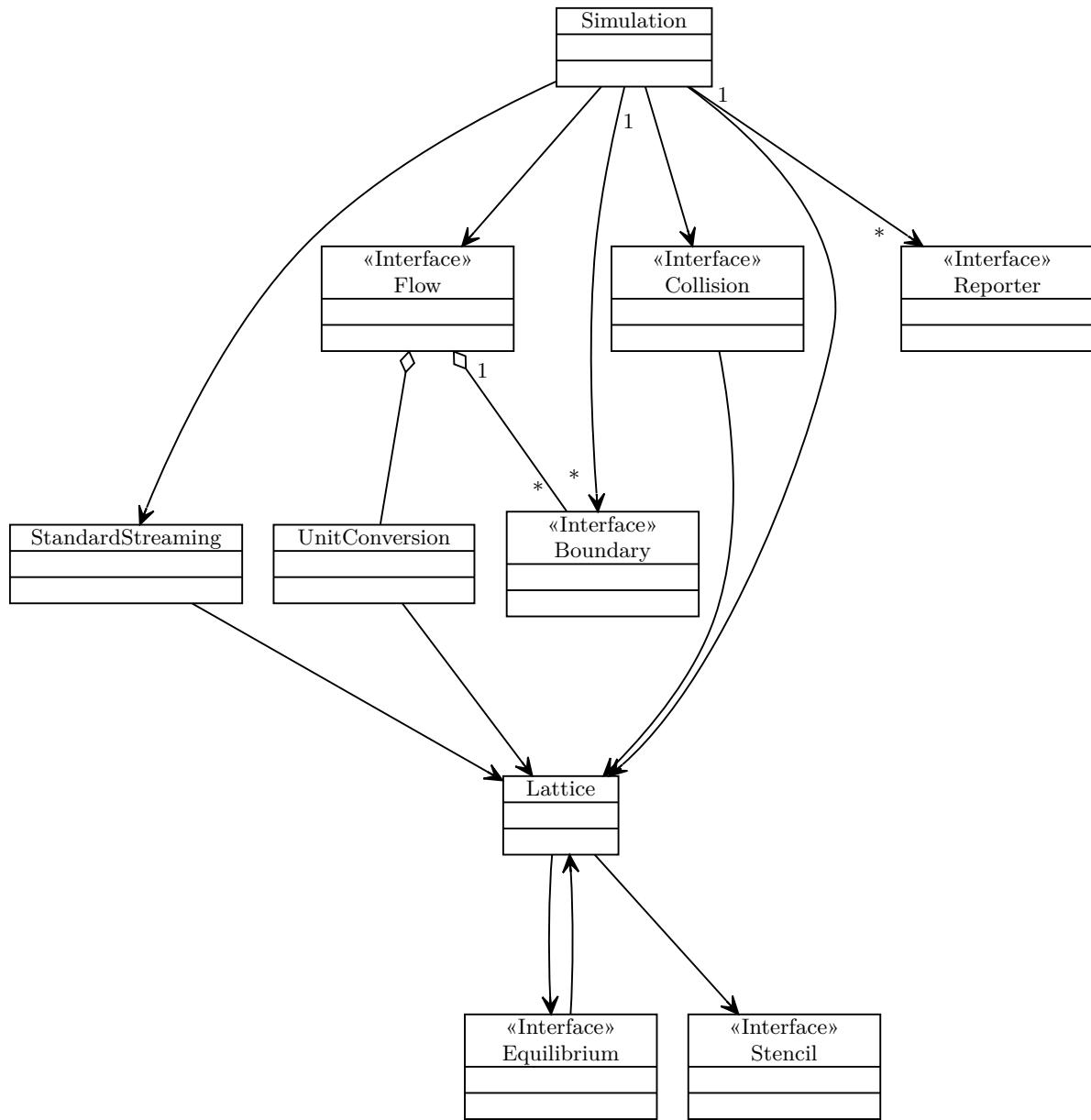


Abbildung 3.1.1-1: Vereinfachtes UML-Klassendiagramm des LBM-Lösers Lettuce als Übersicht seiner Bestandteile. Von jedem der gezeigten Interfaces erben mehrere verschiedene Klassen, welche diese Funktion erfüllen können. Die erbenden Klassen haben dabei die selben Assoziationen wie die vererbende Klasse. Alle ausgelassenen Multiplizitäten sind 1 zu 1.

Lösers ergibt. So können nicht nur die vorgefertigten Elemente ausgetauscht werden, sondern auch neue Elemente mit dem entsprechenden Interface integriert werden. Im Klassendiagramm in Abbildung 3.1.1-1 ist eine zusammengefasste Übersicht der Struktur dargestellt.

Dabei ist die Klasse **Simulation** die zentrale Komponente, in welcher der in Abschnitt 2.4 beschriebene Algorithmus der LBM realisiert ist. Seine hier implementierte Umsetzung ist in Abbildung 3.1.1-2 als Diagramm veranschaulicht. Während der Ausführung ruft **Simulation** die Objekte anderer Klassen auf, um die darin hinterlegten Informationen zu gewinnen oder darin implementierte Berechnungen und Funktionen einzusetzen. Dementsprechend werden Strömungs- und Kollisionsschritt von den Klassen **StandardStreaming** und Klassen, die dem **Collision**-Interface entsprechen, umgesetzt. Sie werden dazu in jedem Zeitschritt von **Simulation** aufgerufen und führen dann die jeweiligen Operationen aus. Für die Anwendung der Randbedingungen sind entsprechende Objekte der Klasse **Boundary** zuständig. Zusätzlich gibt es die verschiedenen **Reporter**, welche von der Simulation am Ende eines Zeitschritts aufgerufen werden, um verschiedene Ausgaben aus der Verteilungsfunktion zu foltern und auf die gewünschte Art auszugeben.

Über diese Kernelemente hinaus gibt es weitere Klassen, die den anderen Informationen oder grundlegendere Funktionen bereitstellen. **Lattice** speichert Informationen über die Prozessorart, also CPU oder GPU, auf welcher die Simulation durchgeführt werden soll, den Datentyp in dem das geschehen soll sowie die gewählte Geschwindigkeitsdiskretisierung. Zusätzlich führt **Lattice** die Berechnung von ρ , j und u aus der Verteilungsfunktion durch und stellt das **Equilibrium** bereit, welches die Gleichgewichtsverteilung berechnet. Eine Klasse, welche dem **Flow**-Interface entspricht, stellt jeweils alle Informationen bereit, die mit dem Aufbau der Strömung zu tun haben. Darunter die Koordinatenwerte aller Gitterpunkte, die Gesamtmaße der Simulationsdomäne und die Anfangswerte. Zusätzlich beinhaltet **Flow** auch die erwähnten Randbedingungen und die Klasse **UnitConversion**, in der die Umrechnung zwischen Lattice-Einheiten und physikalischen Einheiten für die relevanten Größen umgesetzt ist.

Welche Umsetzung der jeweiligen Klassen mit welchen Parametern verwendet werden soll, entscheidet der Nutzer dabei bei der Initialisierung, indem Objekte der entsprechenden Klassen an die passende Stelle übergeben werden. So kann bspw. auch ein selbst implementierter Kollisionsoperator als Klasse mit dem entsprechenden Interface implementiert und dann zusammen mit weiteren vorgefertigten Bestandteilen genutzt werden.

3.1.2 Benutzung des Lözers

Um Lettuce zu nutzen, muss zunächst entsprechend der Anweisungen im ReadMe [31] eine Python-Umgebung mit den benötigten Paketen aufgesetzt werden. Anschließend kann bereits mit Hilfe eines kurzen Skripts eine Simulation gestartet werden. Die dazu benötigten Schritte sollen an dieser Stelle an einem kurzen Beispiel erläutert werden:

```
import lettuce as lt
import torch

# ----- Set up simulation -----
lattice = lt.Lattice(stencil=lt.D3Q27, device=torch.device("cuda"), dtype=torch.
                     float32)
flow = lt.TaylorGreenVortex3D(resolution=100, reynolds_number=1600, mach_number=0
                               .1, lattice)
collision = lt.BGKCollision(lattice, tau=flow.units.relaxation_parameter_lu)
streaming = lt.StandardStreaming(lattice)
simulation = lt.Simulation(flow, lattice, collision, streaming)

# Add reporter to get outputs
reporter = lt.VTKReporter(lattice, flow, interval=25, filename_base="/home/user/
                           VTK/example")
```

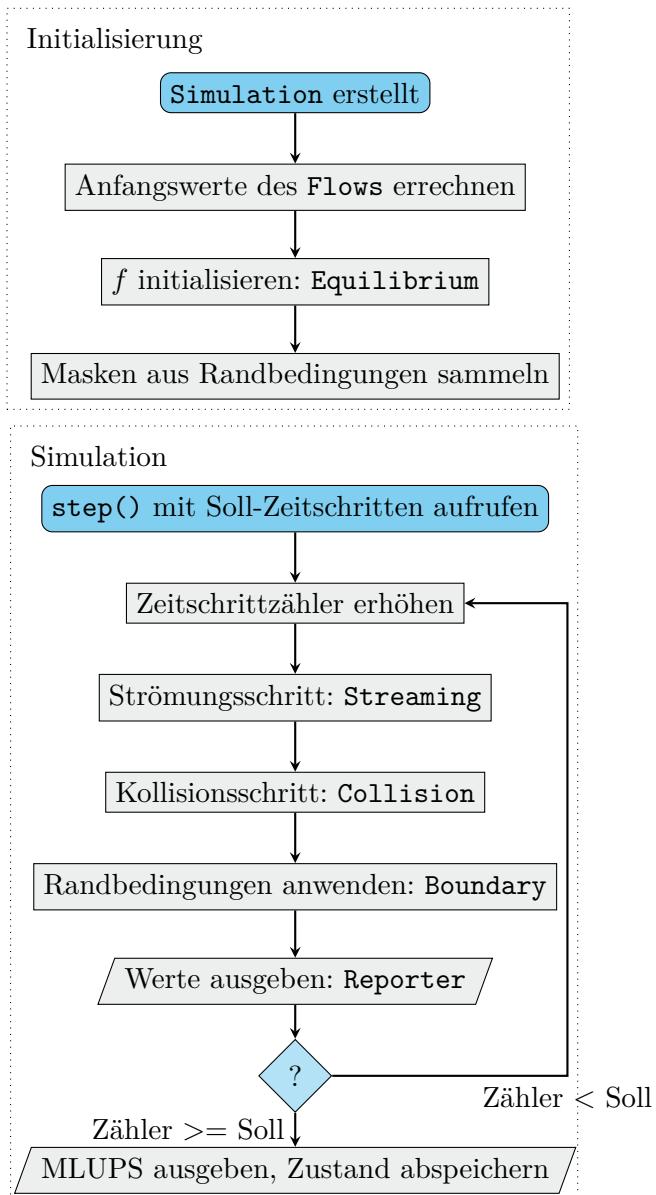


Abbildung 3.1.1-2: Vereinfachte Visualisierung des Ablaufs einer Simulation mit Lettuce als Flussdiagramm mit Nennung der jeweils zuständigen Klasse. Der Ablauf beider Teile ist dabei in der Klasse `Simulation` implementiert. MLUPS steht für "Million Lattice Updates per Second" und ist eine Metrik für die Rechengeschwindigkeit.

```

simulation.reporters.append(reporter)

# ----- Simulate until time = 10 (PU) -----
print("MLUPS: ", simulation.step(1000))

```

In diesem Beispiel wird eine dreidimensionale Tylor-Green-Vortexströmung von einer Gräfikarte auf einer Domäne von 100^3 Gitterpunkten für 1000 Zeitschritte simuliert, wobei die Druck- und Geschwindigkeitsfelder alle 25 Zeitschritte als vtk-Datei für die Visualisierung ausgegeben werden.

Dazu werden zunächst `Lattice` sowie `TaylorGreenVortex3D` vom Interface `Flow`, also die Klassen, welche als Informationsquelle dienen, mit den gewünschten Einstellungen instanziert. Anschließend folgen die funktionalen Bestandteile der Simulation, der Kollisionsoperator und das Streaming, welche ebenfalls aus den vorgefertigten Klassen gewählt werden. Schließlich wird mit diesen Bestandteilen die Klasse `Simulation` instanziert, wodurch der Initialisierungs-Vorgang durchlaufen wird. Dem Objekt der Klasse `Simulation` wird nun zusätzlich ein `Reporter` hinzugefügt, welcher hier bspw. die Druck- und Geschwindigkeitsfelder als Datei für die Visualisierung ausgibt. Schließlich wird die Simulation durchgeführt, dazu wird die Funktion `step()` der Simulation mit der gewünschten Anzahl an Zeitschritten aufgerufen, welche abschließend die Ausführungsgeschwindigkeit in Millionen Lattice-Updates pro Sekunde (MLUPS) zurückgibt.

3.2 Erweiterungen des Lösers

Um den geplanten Simulationsaufbau gut umsetzen zu können, sind einige Erweiterungen des Lösers Lettuce erforderlich, auf welche in diesem Abschnitt eingegangen wird. Zum einen wird eine Möglichkeit zur Parallelisierung der Simulation auf mehreren Prozessoren eingefügt, getestet und erprobt. Zum anderen werden Randbedingungen implementiert, welche für den Aufbau der Simulationsumgebung, die im nächsten Kapitel vorgestellt wird, benötigt werden. Der Quellcode des Lösers mit diesen Änderungen ist auf der beigelegten CD zu finden.

3.2.1 Parallelisierung

Die Parallelisierung einer Simulation bietet den Vorteil, dass Rechnungen, die gleichzeitig stattfinden können, auf mehrere Prozesse oder Geräte verteilt werden und so schneller abgearbeitet werden können. Auch ermöglicht die Verteilung der Daten ein Simulieren größerer Simulationsdomänen, da die Größe nicht mehr durch den Arbeitsspeicher (RAM) des einzelnen Geräts beschränkt wird, sondern durch den gesamten RAM der beteiligten Geräte. Für die Simulationen im Rahmen der Arbeit ist dabei vor allem die Möglichkeit einer größere Simulationsdomäne von Vorteil. Da ein Fehler der LBM proportional zu Δx^2 ist (vgl. Abschn. 2.8), ermöglicht eine größere Domäne auch eine höhere Genauigkeit und Aussagekraft der Ergebnisse. [5]

Die Lattice-Boltzmann-Methode eignet sich naturgemäß sehr gut für die Parallelisierung, da der rechenaufwändigste Schritt des Algorithmus, der Kollisionsschritt, in den meisten Fällen an jedem Gitterpunkt lokal abläuft. So kann die Domäne auf verschiedene Prozesse verteilt werden, die die Kollision in je einem Bereich durchführen ohne sich dabei austauschen zu müssen. Lediglich beim Strömungsschritt und für eventuelle Ausgaben ist es erforderlich, zwischen den Prozessen Daten auszutauschen, wobei auch hier relativ geringe Datenmengen anfallen. [5]

In diesem Abschnitt soll die Parallelisierung des Lösers Lettuce, welche im Rahmen der Arbeit implementiert wird, vorgestellt werden. Dabei werden zunächst der Aufbau der Erweiterung und ihre Funktionsweise erläutert. Anschließend wird die Implementierung durch den Abgleich mit der sequenziellen Implementierung verifiziert. Abschließend wird die Skalierung

des parallelisierten Lösers untersucht, um den Zuwachs an Geschwindigkeit und maximaler Domänengröße zu quantifizieren.

3.2.1.1 Funktionsweise und Aufbau

Wie bereits im vorherigen Abschnitt erläutert, ist eine Kernidee von Lettuce, dass alle Funktionselemente, wie Kollisionsoperatoren, Ausgaben und Ähnliches modular angelegt sind und leicht ausgetauscht werden können. Dementsprechend soll sich auch die Parallelisierung in diese Struktur einfügen und als zweite Option neben der sequenziellen Implementierung stehen und ebenso mit den anderen Elementen funktionieren, ohne Inkompatibilitäten zu erzeugen.

Die Funktionsweise der Parallelisierung wird wie in [5] vorgeschlagen umgesetzt, indem die Simulationsdomäne entlang der x-Richtung in Abschnitte eingeteilt wird, welche von jeweils einem Prozess bearbeitet werden. Diese Aufteilung ist in Abbildung 3.2.1.1-1 verdeutlicht. Sie bietet den Vorteil, dass jeder Prozess nur zwei Nachbarprozesse hat, mit denen er sich austauschen muss, was die Komplexität und die Anzahl erforderlicher Kommunikationsvorgänge reduziert. Allerdings bietet diese Aufteilung wenig Flexibilität und erfordert bei sehr schmalen Abschnitten pro Abschnitt mehr Datenaustausch als eine Aufteilung in bspw. Würfel gleichen Volumens.

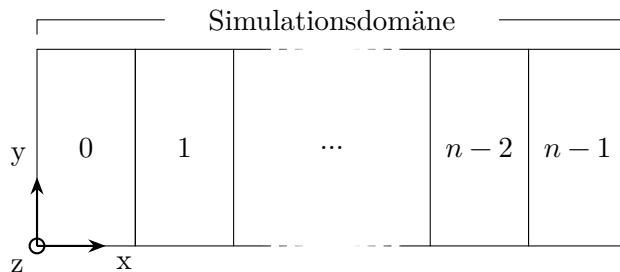


Abbildung 3.2.1.1-1: Visualisierung der Aufteilung der Simulationsdomäne auf die einzelnen Prozesse: entlang der x-Achse wird jedem Prozess ein Abschnitt der Domäne zugewiesen (hier dargestellt durch die zugehörige Nummer des Prozesses im Feld).

Die Bibliothek Pytorch, auf welcher Lettuce basiert, bietet verschiedene integrierte Backends für die Parallelisierung von Berechnungen an [36]. Da die Bibliothek jedoch vor allem für das Trainieren neuronaler Netze gedacht ist, bieten nur wenige davon die hier erforderliche Möglichkeit, Tensoren zwischen einzelnen Prozessen zu übertragen, anstatt zwischen allen zugleich [36]. Als Optionen verbleiben so nur das Backend "Gloo", welches die nötigen Funktionen auf CPUs unterstützt und MPI (Message Passing Interface), welches die Funktionen auf CPUs und GPUs unterstützt [36]. Während MPI sehr verbreitet ist, bietet Gloo den Vorteil einer sehr engen Integration in Pytorch, sodass es keiner externen Software bedarf um damit verteilte Programme auszuführen. Da sie sich aus Sicht der Implementierung nur in der Initialisierung unterscheiden, werden im Folgenden beide Backends in Lettuce integriert.

Wie die anderen Bestandteile des Lösers, wird auch die Parallelisierung mit Hilfe verschiedener neuer Klassen umgesetzt. Im Klassendiagramm in Abbildung 3.2.1.1-2 ist dargestellt, um welche Klassen es sich handelt und wie sie sich in die Struktur des Lösers einfügen. Im Folgenden soll ihre jeweilige Funktion und Funktionsweise kurz beleuchtet werden. Wie eingangs erläutert, ist eine Kommunikation der verschiedenen Prozesse vor allem im Rahmen des Strömungsschritts zwingend erforderlich. Dementsprechend wird eine neue Streaming-Klasse implementiert, welche neben der Advektion auch den Austausch mit den benachbarten Prozessen beinhaltet. Diese Klasse bildet das Kernelement der Parallelisierung. Zusätzlich wird mit **DistributedSimulation** eine modifizierte **Simulation**-Klasse erstellt, welche die Aufteilung der Domäne berücksichtigt. Für die Aufteilung von Tensoren auf die verschiedenen Prozesse wird zusätzlich die Klasse **Grid** erstellt. Diese ermöglicht es, Teile der Domäne dem jeweiligen

Prozessen zuzuordnen und stellt Informationen über die gesamte und lokale Domäne bereit.

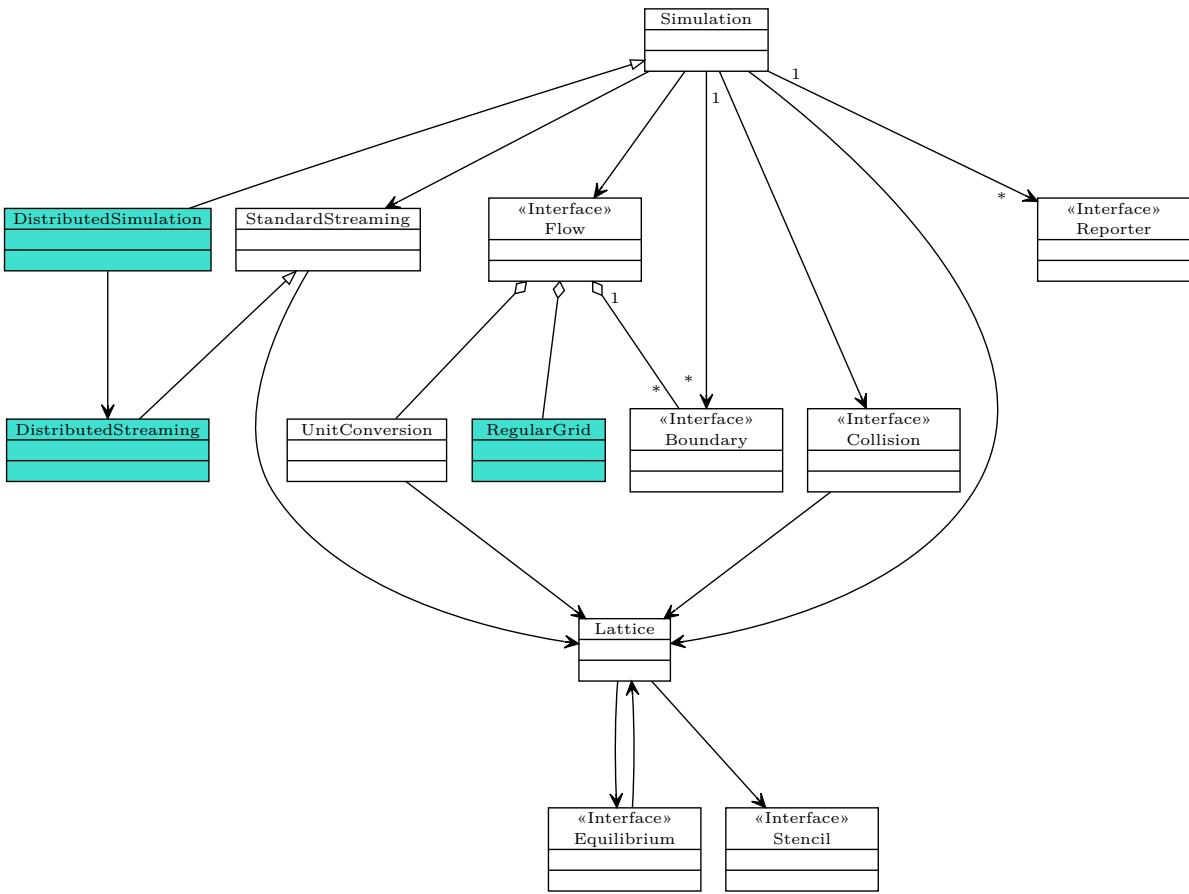


Abbildung 3.2.1.1-2: Vereinfachtes UML-Klassendiagramm des Lösers Lettuce, in dem die Erweiterungen für die Parallelisierung cyan hinterlegt sind, als Übersicht über die Erweiterungen für die Parallelisierung. Von jedem Interface erben mehrere Klassen, welche diese Funktion erfüllen können. Die erbenden Klassen haben dabei die selben Assoziationen wie die vererbende Klasse. Alle fehlenden Multiplizitäten sind 1 zu 1.

DistributedStreaming

Die neue Streaming-Klasse funktioniert nach dem in Tabelle 3.2.1.1-1 dargestellten Ablauf. Dementsprechend werden zunächst die Aus- und Eingaben an andere Prozesse vorbereitet und dann die asynchrone Übertragung mit den Prozessen, zu welchen die entsprechenden benachbarten Abschnitte gehören, angestoßen. Dies geschieht mit Hilfe der von Pytorch bereitgestellten Funktionen `isend` und `irecv`. Während die Übertragung aussteht oder durchgeführt wird, werden die Domäne und die gesammelte "No-Stream-Mask", welche angibt, in welchen Bereichen der Domäne kein Strömungsschritt durchgeführt werden soll, vorbereitet. Dazu wird auf jeder Seite eine y-z-Ebene (oder y-Reihe) von Nullen angehängt, um Platz für die Inputs zu schaffen. Nachdem das Empfangen der Inputs, also der Populationen, welche aus den benachbarten Bereichen in den aktuell betrachteten strömen, abgeschlossen ist, werden diese hier an den entsprechenden Stellen eingefügt. Dann sind alle Informationen, die zur Advektion innerhalb der Domäne des einzelnen Prozesses benötigt werden, vorhanden und die eigentliche Advektion wird durchgeführt. Abschließend wird gewartet, bis das Senden der Outputs, also der Populationen, welche aus dem betrachteten Bereich hinaus in einen benachbarten Strömen, ebenfalls abgeschlossen ist. Dann werden die Populationen f ohne die zuvor angehängten Gitterpunkte, in welchen sich nun nur noch die ausgestromten Outputs

befinden, zurückgegeben. Insgesamt kann der Strömungsschritt so stattfinden, als gäbe es innerhalb der Simulationsdomäne keine unterschiedlichen Bereiche.

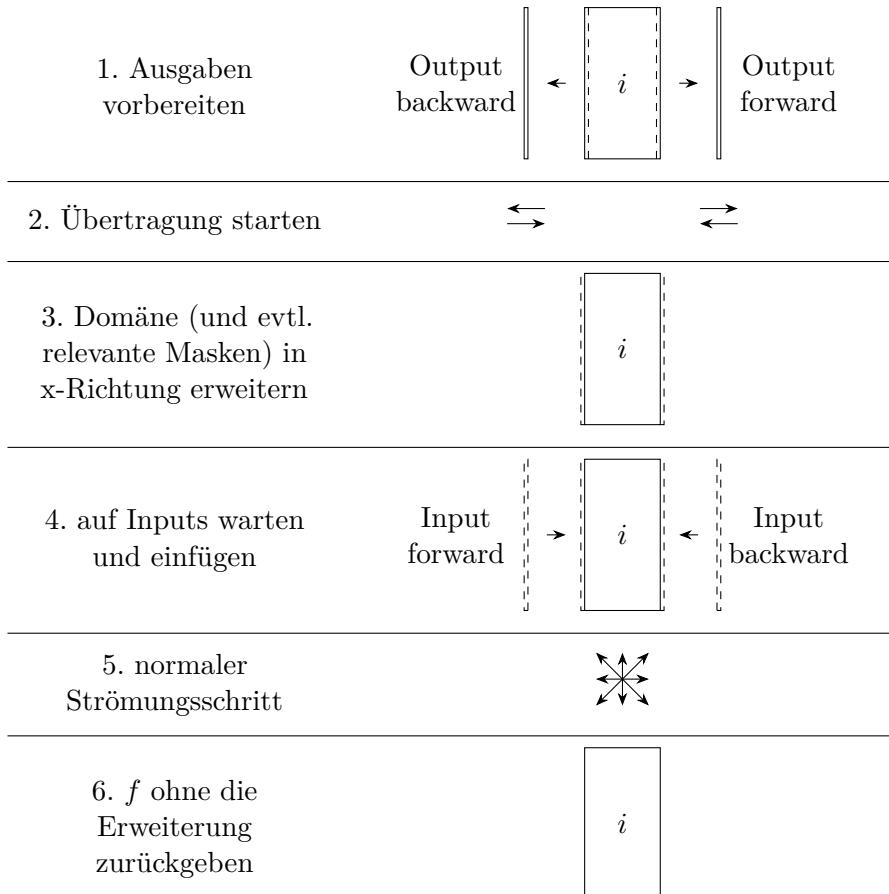


Tabelle 3.2.1.1-1: Übersicht über die Funktionsweise der Klasse `DistributedStreaming` bzw. der verteilten Advektion: links jeweils eine Beschreibung des Schritts, rechts die Darstellung, was mit dem Domänen-Bereich des i -ten Prozesses passiert.

DistributedSimulation

Die neue Klasse `DistributedSimulation` unterscheidet sich von der normalen `Simulation`-Klasse insofern, dass sie bei der Initialisierung und der Anwendung von Randbedingungen die Aufteilung der Simulationsdomäne berücksichtigt und entsprechend nur die für den jeweiligen Prozess relevanten Ausschnitte des Koordinatensystems und der Masken berücksichtigt. Zudem erlaubt es diese Klasse, dass beim Zwischenspeichern und Laden des Zustands der Simulation jeder Prozess eine eigene Datei erstellt und lädt. Da diese Änderungen relativ geringfügig sind, kann diese Klasse in einem nächsten Schritt potentiell wieder mit der ursprünglichen `Simulation`-Klasse vereinigt werden.

RegularGrid

Um die verteilte Domäne zu handhaben wird eine zusätzliche `Grid`-Klasse erstellt, welche die entsprechende Eigenschaft der Klassen des `Flow`-Interfaces ersetzt. Die neue Klasse generiert ebenfalls die Koordinatenwerte an den einzelnen Gitterpunkten, kann sich dabei aber auch auf den Bereich des zugehörigen Prozesses beschränken. Das selbe gilt für die Anzahl an Gitterpunkten in den einzelnen Koordinatenrichtungen. Zusätzlich bietet sie eine Funktion mit welcher ein globaler Tensor, der sich über die gesamte Simulationsdomäne erstreckt, auf den

Bereich eines bestimmten Prozesses beschränkt werden kann. Schließlich gibt es außerdem eine Funktion, mit welcher die einzelnen Stücke eines derart verteilten Tensor beim Prozess 0 gesammelt und wieder vereint werden können. Dies kann beispielsweise für Ausgaben verwendet werden, sofern der Tensor in den verfügbaren Arbeitsspeicher von Prozess 0 passt.

Startup-Routine

Neben diesen Erweiterungen wird zusätzlich eine Funktion bereitgestellt, welche das Starten der verteilten Simulation übernimmt. Sie wird mit der gewünschten Anzahl an Prozessen und einer weiteren Funktion, welche dann die weitere Erstellung und Durchführung der Simulation beinhaltet, aufgerufen. Dabei wird je nach gewähltem Backend eine entsprechende Startabfolge ausgeführt. Hierbei wird, sofern die Simulation auf mehreren Grafikkarten eines Computers durchgeführt werden soll, auch die Verteilung auf die verschiedenen Karten des Systems durchgeführt. Nach der Erstellung der Prozesse wird dann die beigegebene Funktion auf diesen ausgeführt, sodass die Simulation beginnt.

3.2.1.2 Benutzung

In diesem Abschnitt soll kurz auf die Nutzung der parallelisierten Simulation eingegangen werden. Zusätzlich zur eingangs beschriebenen Python-Umgebung wird für die Nutzung der parallelisierten Simulation auf GPUs noch das Paket OpenMPI, welches auf Conda-Forge erhältlich ist, benötigt. Dieses ermöglicht die Nutzung des MPI-Backends, wobei darauf zu achten ist, dass je nach verwendeten Komponenten verschiedene zusätzliche Software erforderlich sein kann. Um die Nutzung des Backends zu ermöglichen muss zusätzlich die Pytorch-Bibliothek mit den entsprechenden MPI-Bibliothek neu gebaut werden. Mit Hilfe der entsprechenden Umgebung kann dann, wie bei der normalen Nutzung von Lettuce, mit wenig Aufwand eine verteilte Simulation gestartet werden:

```
import lettuce as lt
import torch

def run(device, rank, size):
    # ----- set up simulation -----
    lattice = lt.Lattice(lt.D3Q27, device=device, dtype=torch.float32)
    flow = lt.TaylorGreenVortex3D(resolution=100, reynolds_number=1600,
                                   mach_number=0.1, lattice=lattice,
                                   rank=rank, size=size)
    collision = lt.BGKCollision(lattice, tau=flow.units.relaxation_parameter_lu)
    streaming = lt.DistributedStreaming(lattice, rank, size)
    simulation = lt.DistributedSimulation(flow, lattice, collision, streaming,
                                            rank, size)

    # Add reporter to get outputs
    reporter = lt.VTKReporter(lattice, flow, interval=25, filename_base="/home/
    user/VTK/example")
    simulation.reporters.append(reporter)

    # ----- Simulate -----
    print("MLUPS: ", simulation.step(1000))

# ----- set up processes -----
if __name__ == "__main__":
    size = 4
    device = torch.device("cuda")
    lt.distribute(run, size, device, "mpi")
```

Im Vergleich zum Beispiel für die allgemeine Nutzung von Lettuce in Abschnitt 3.1.2 befindet sich hier quasi der gesamte Inhalt des vorherigen Beispiels, also die Anweisungen für den

Aufbau und das Durchführen der Simulation, in einer Funktion. Diese wird an die Startup-Routine übergeben, welche zunächst die gewünschte Anzahl an Prozessen startet und auf die verfügbaren Geräte verteilt, bevor sie in allen Prozessen die Ausführung der Funktion anstößt. Der Code für die Simulation selber unterscheidet sich nur darin, dass nun an `Flow` die Parameter `rank` und `size`, also die Nummer des aufrufenden Prozesses und die Anzahl an erstellten Prozessen, übergeben werden. Mit diesen Angaben wird dann die Domäne auf die Prozesse aufgeteilt. Entsprechend werden hier natürlich auch Objekte der Klassen `DistributedSimulation` und `DistributedStreaming` verwendet. Mit einem solchen Skript kann dann bereits eine verteilte Simulation auf einem Cluster durchgeführt werden.

Dabei ist zu beachten, dass in Pytorch bereits OpenMP, eine Bibliothek zur lokalen Parallelisierung, integriert ist. Dadurch werden alle weiteren Threads, welche dem Prozess zur Verfügung stehen und nicht von den Prozessen der Parallelisierung verwendet werden, zur lokalen Parallelisierung auf dem jeweiligen Gerät genutzt.

3.2.1.3 Verifizierung

In diesem Abschnitt soll die korrekte Umsetzung der Parallelisierung verifiziert werden. Dazu werden Benchmark-Strömungen, von je einem einzelnen und mehreren Prozessen mit den selben Parametern simuliert und die jeweiligen Ergebnisse verglichen. Dabei zeigen ähnliche bzw. identische Werte, dass sich das Verhalten der Simulation durch die Parallelisierung nicht verändert hat.

Für einen leichten Abgleich der Ergebnisse von normaler und verteilter Simulation wird nach jedem Zeitschritt der verteilten Simulation auf Prozess 0 ein Zeitschritt der sequenziellen Simulation durchgeführt und sein Ergebnis mit dem entsprechenden Ergebnis des verteilten Zeitschritts verglichen. Nach einer größeren Anzahl an Zeitschritten, in denen sich etwaige Abweichungen aufsummieren können, werden die Werte von u und p an jedem Gitterpunkt abgeglichen. Dabei werden mit der zwei- und dreidimensionalen Taylor-Green-Vortexströmung (TGV2D bzw. TGV3D) als üblichen Benchmark-Strömungen sowie der zweidimensionalen Couette-Strömung, einer verbreiteten Standardströmung mit nicht-periodischen Randbedingungen, sowohl zwei- und dreidimensionale Fälle als auch Randbedingungen abgedeckt. Um auch das Zusammenspiel der verschiedenen Kommunikationsprotokolle, die von der Bibliothek eingesetzt werden, zu überprüfen, wird die Verifizierung sowohl für 10^5 Zeitschritte auf drei der NVIDIA V100 des Knotens `gpu4` als auch für je 10^4 Schritte auf 17 Threads eines Intel Xeon Gold 6130 eines `hpc3`-Knotens durchgeführt. Die ungeraden Zahlen sind dabei bewusst gewählt, um auch den Fall einer ungleichmäßigen Verteilung der Domäne auf die Prozesse zu erproben. Wie auch in Abbildung 3.2.1.3-1 zu erkennen, können dabei keine Differenzen zwischen beiden Simulationen festgestellt werden. Das bestätigt, dass sich durch die Parallelisierung der Simulation keine Änderung der berechneten Werte bzw. des Verhaltens ergibt und sie somit wie vorgesehen funktioniert.

3.2.1.4 Benchmark

Abschließend soll außerdem die Skalierung der Simulation quantifiziert werden, also untersucht werden, wie sich die Anzahl an Prozessoren auf die Geschwindigkeit der Simulation auswirkt. Zusätzlich wird außerdem untersucht, welche Domänen-Größe mit der jeweiligen Menge an RAM der verschiedenen Geräte-Kombinationen simuliert werden kann.

Rechengeschwindigkeit

Um die Skalierung der Rechengeschwindigkeit der Simulation zu untersuchen wird die Standard-Benchmark-Strömung TGV3D mit dem D3Q27-Geschwindigkeitssatz verwendet. Sie wird von jeder Gerätekombination auf einer Domäne von 256^3 mit einer Reynolds-Zahl von 10, einer Machzahl von 0.1 und einer Genauigkeit von 32 Bit simuliert. Dabei wird die Rechenzeit,

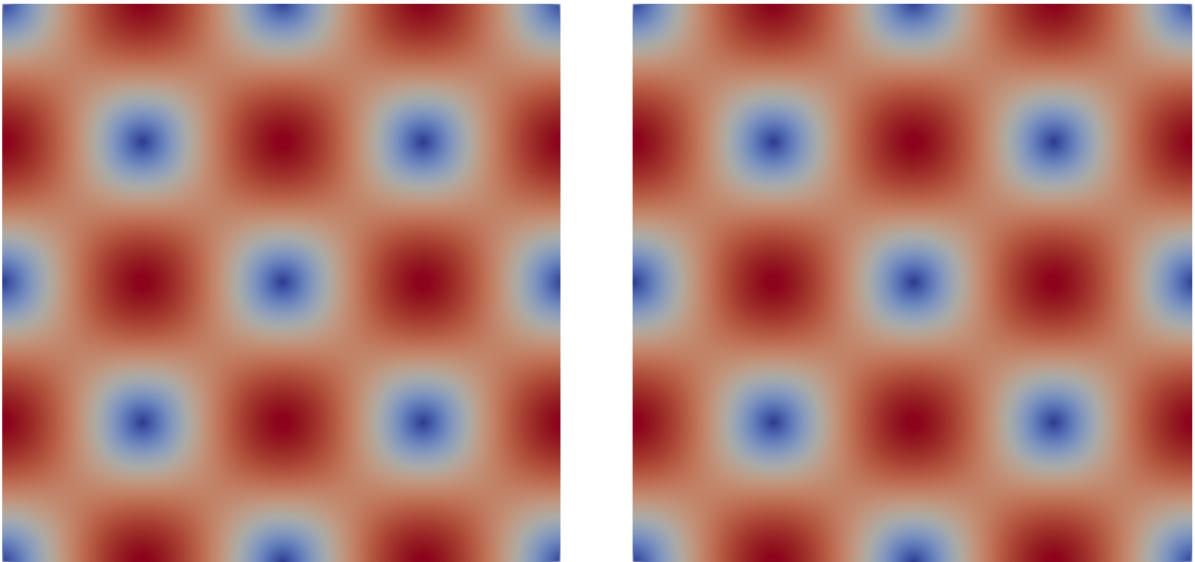


Abbildung 3.2.1.3-1: Beispiel für ein Ergebnis der Verifizierung der Parallelisierung auf GPUs: der Betrag der Geschwindigkeit der zweidimensionalen Taylor-Green-Vortexströmung mit $\text{Re} = 10$ und $\text{Ma} = 0.1$ nach 10^5 Zeitschritten, der sequenziell (links) oder verteilt (rechts) berechnet wird, unterscheidet sich nicht.

welche für einen Zeitschritt der Simulation in Anspruch genommen wird, aufgezeichnet. Es werden dabei ein bis vier NVIDIA V100 des gpu4-Knotens getestet sowie verschiedene Mengen an Knoten und Threads der Prozessoren aus dem Pool hpc3, welche je Knoten einen Intel Xeon Gold 6130 mit 64 Threads bieten. Nachdem ein erster Test keinen Geschwindigkeitsunterschied zwischen beiden Backends ergeben hatte, wird für das Benchmarking der CPUs statt MPI das leichter zu verwendende Gloo eingesetzt. Neben der neuen Parallelisierung wird dabei auch der Einfluss der bereits in Pytorch integrierten, lokalen Parallelisierung mit OpenMP untersucht.

In Abbildung 3.2.1.4-1 ist die mittlere Rechenzeit für einen Zeitschritt über der beteiligten Anzahl an Grafikkarten dargestellt. Hier zeigt sich eine proportionale Abnahme der Rechendauer, wird die Anzahl der Grafikkarten verdoppelt, so halbiert sich die benötigte Zeit. Die Rechenzeit von rund 0.15 s pro Zeitschritt, die eine einzelne GPU benötigt, kann durch die parallele Nutzung von 4 GPUs auf rund 0.04 s pro Zeitschritt reduziert werden.

Die Ergebnisse des Benchmarkings der Parallelisierung mit CPUs sind in Abbildung 3.2.1.4-2 zusammengefasst. Dazu ist die mittlere Rechenzeit für einen Zeitschritt über der beteiligten Anzahl an Gloo-Prozessen dargestellt. Verschiedene Anzahlen an Gloo-Prozessen pro Knoten, sowie das selbe mit zusätzlicher OpenMP-Parallelisierung sind dabei jeweils als separate Messreihen dargestellt. Hier zeigt sich bei Gloo-Prozessen, die auf nur einem Knoten liegen, ("1 Kn, -") zwischen zwei und 16 Prozessen ebenfalls ein proportionaler Geschwindigkeitszuwachs. Werden alle 64 Threads des Prozessors für Gloo-Prozesse genutzt wird die Bestzeit von 0.89 s Rechenzeit pro Zeitschritt erreicht. Dies ist geringfügig schneller als die Verwendung der gleichen Anzahl an Threads auf einem Knoten für eine Kombination aus Gloo- und OpenMP-Prozessen ("1 Kn, ✓"), die unabhängig von der Anzahl an Gloo-Prozessen eine ähnliche geringe Rechenzeit benötigen. Aus den Ergebnissen mit 2 bzw. 4 Gloo-Prozessen pro Knoten ("2/Kn, -" bzw. "4/Kn, -") zeigt sich jedoch, dass diese Steigerung nicht über mehrere Knoten hinweg gegeben ist. Sind mehrere Knoten involviert, so bleibt die Rechengeschwindigkeit beim Hinzufügen weiterer Prozesse auf anderen Knoten etwa gleich. Auch bei der vollständigen Nutzung zweier Knoten mit je 64 Gloo-Prozessen ergibt sich kein Zuwachs im Vergleich zur vollständigen Nutzung eines Knotens. Die Ergebnisse bei Verwendung von 2 bzw. 4 Gloo-Prozesse mit Unterstützung durch OpenMP ("2/Kn, ✓" bzw. "4/Kn, ✓")

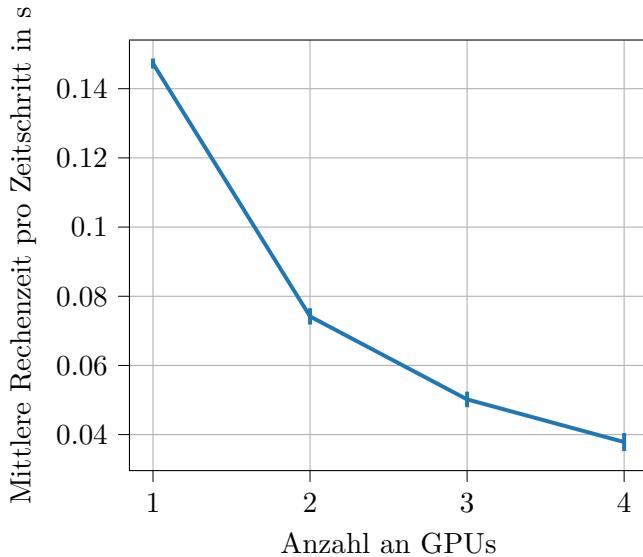


Abbildung 3.2.1.4-1: Durchschnittliche Rechenzeit der Simulation über der Anzahl beteiligter GPUs (NVIDIA V100 auf Knoten gpu4), auf welchen die Simulation parallelisiert ausgeführt wird. Es ist zu erkennen, dass die Rechenzeit mit der Anzahl an Grafikkarten proportional abnimmt. Die Simulation mit einer GPU wird mit der sequenziellen Implementierung durchgeführt, die anderen mit der verteilten. Die Fehlerbalken zeigen dabei die Standardabweichung an.

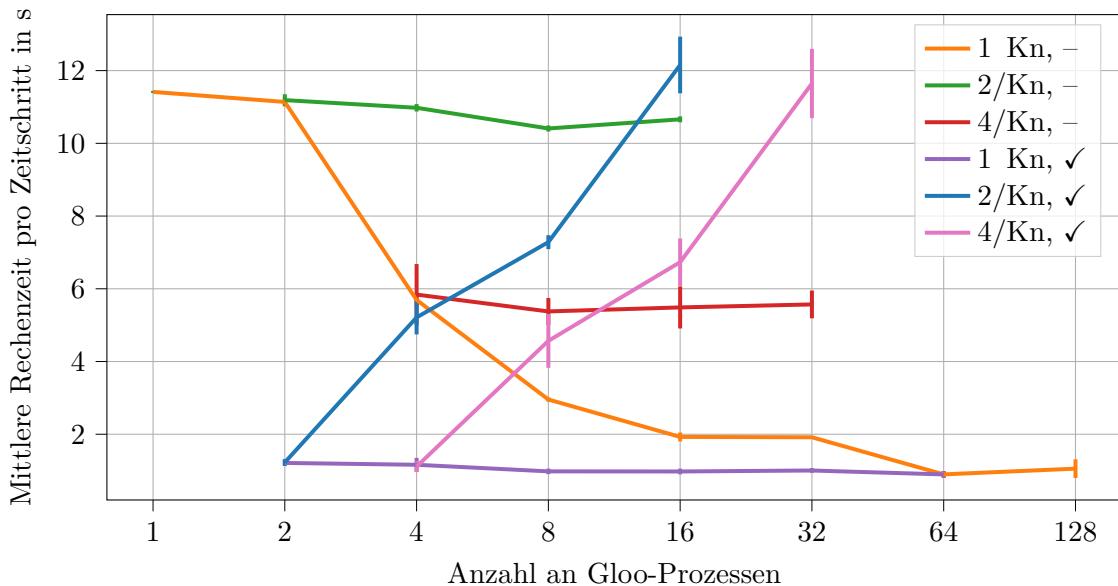


Abbildung 3.2.1.4-2: Zusammenfassung der durchschnittlichen Rechenzeit pro Zeitschritt der Simulation über der Anzahl beteiligter Gloo-Prozesse, auf welche die Simulation parallelisiert ist. Jeder Knoten besitzt dabei 64 Threads (Intel Xeon Gold 6130 auf Knoten hpc3). Die Simulation auf einem Gloo-Prozess wird mit der sequenziellen Implementierung durchgeführt, die anderen mit der parallelisierten. Die jeweiligen Linien ergeben sich bei verschiedener Kombination verschiedener Anzahlen von Gloo- und OpenMP-Prozessen. Der erste Teil der Legende gibt an, wie viele Gloo-Prozesse jedem der beteiligten Knoten zugewiesen sind (1 steht hierbei für alle Threads auf dem selben Knoten (bzw. auf zwei Knoten im Fall der 128 Prozesse)). Der zweite Teil der Legende gibt an, ob OpenMP verwendet wird, wobei bei Verwendung jeweils die restlichen Threads jedes Knotens für OpenMP-Prozesse genutzt werden. Die Fehlerbalken zeigen die Standardabweichung an.

zeigen erneut, dass die Nutzung aller Threads der Knoten zwar zunächst eine deutliche Verbesserung der Rechenleistung ermöglicht, diese bei Hinzufügen weiterer Knoten aber sogar abnimmt, wodurch die verteilte Simulation auf mehreren Knoten mit OpenMP schließlich sogar langsamer ist als die ohne. Bei der gleichen Anzahl an Gloo-Prozessen, ist der Prozess mit 4 Gloo-Prozessen je Knoten, genauso wie bei den Ergebnissen ohne OpenMP, schneller als der mit je 2. Vergleich man die Rechenzeiten bei der gleichen Anzahl an involvierten Knoten, also bspw. das Ergebnis von 2 Gloo-Prozessen pro Knoten mit OpenMP bei 8 Threads mit dem von 4 Gloo-Prozessen pro Knoten mit OpenMP bei 16 Threads, so sind die Rechenzeiten jedoch auch hier sehr ähnlich.

Insgesamt zeigen Ergebnisse des Benchmarkings mit CPUs, dass die Parallelisierung der Simulation auf mehrere Gloo-Prozesse eine Verringerung der Rechenzeit ermöglicht, diese übersteigt die Verringerung durch die bereits integrierte OpenMP-Parallelisierung jedoch kaum. Zudem lassen die Ergebnisse vermuten, dass die Kommunikation der Prozesse innerhalb eines Knotens deutlich schneller ist als die zu Prozessen in anderen Knoten, weshalb das Hinzufügen weiterer Prozesse auf anderen Knoten keinen Geschwindigkeitszuwachs ergibt. Dies könnte darauf hindeuten, dass die Gloo-Bibliothek durch suboptimal unterstützte Komponenten innerhalb des Clusters verlangsamt wird. Warum die Verwendung von OpenMP bei Kommunikation über mehrere Knoten hinweg zu einer Abnahme der Rechengeschwindigkeit führt, konnte im Rahmen der Arbeit nicht näher geklärt werden. Eine Einflussgröße könnten hier die lokalen Speicherzugriffe der einzelnen Prozesse sein. Bei den GPUs ist dagegen ein klarer Geschwindigkeitszuwachs zu erkennen. Da die GPUs in jedem Fall deutlich schneller sind als alle untersuchten Konfigurationen von CPUs, bzw. eine GPU bereits sechs mal schneller ist als das beste Ergebnis der CPUs, werden die weiteren Untersuchungen in dieser Arbeit, wie mit Lettuce bisher ohnehin üblich, auf GPUs durchgeführt.

Simulationsdomänengröße

Neben dem Geschwindigkeitszuwachs bedeuten mehr kombinierte Knoten und Geräte auch einen Zuwachs an verfügbarem Arbeitsspeicher. Um die jeweils mögliche Größe der Simulationsdomäne auszuloten, werden so lange neue Simulationen mit größerer Domäne aufgesetzt und fünf Zeitschritte durchgeführt, bis die Simulation wegen Speichermangel abgebrochen wird. Dabei wird erneut der TGV3D verwendet, da dieser eine übliche Benchmark-Strömung ist. Dabei ist zu beachten, dass die GPUs pro Gerät eine Menge von 16 GB RAM besitzen, während bei den CPUs jeweils 192 GB RAM zu einem Knoten mit bspw. 64 Threads gehören.

Beim Rechnen auf CPUs kann so bereits bei Verwendung nur eines Knotens eine Domänengröße von mehr als 730^3 mit rund 390 Millionen Gitterpunkten realisiert werden. In Abbildung 3.2.1.4-3 ist entsprechend die maximale Domänen-Größe, die mit den verschiedenen Anzahlen an GPUs simuliert werden kann, dargestellt. Man kann erkennen, dass die GPUs auf Grund des kleineren Arbeitsspeichers weniger große Domänen ermöglichen als die CPUs. Dennoch zeigt sich eine Erhöhung der maximalen Domänengröße durch die Parallelisierung. Durch das Hinzufügen einer zweiten und dritten GPU wird die mögliche Seitenlänge der würfelförmigen Domäne um 44 % erhöht, bzw. die Anzahl an Gitterpunkten verdreifacht. Beim Hinzufügen einer vierten GPU ist allerdings ein geringerer Zuwachs erkennbar. Dies könnte auf mögliche Ineffizienzen hinweisen und sollte vor der weiteren Verwendung des parallelisierten Lösers untersucht werden.

3.2.1.5 Ausblick

Insgesamt zeigen die vorangegangenen Abschnitte, dass die im Rahmen dieser Arbeit integrierte Parallelisierung bereits in einem nutzbaren Zustand ist. Durch den parallelen Einsatz von GPUs konnte die Rechenzeit der Beispieldomäne auf ein Viertel reduziert werden. Auf CPUs wurde ebenfalls eine Geschwindigkeitssteigerung erreicht, diese entsteht jedoch bisher nur wenn alle Prozesse auf dem selben Knoten liegen. Während bei den CPUs bereits der

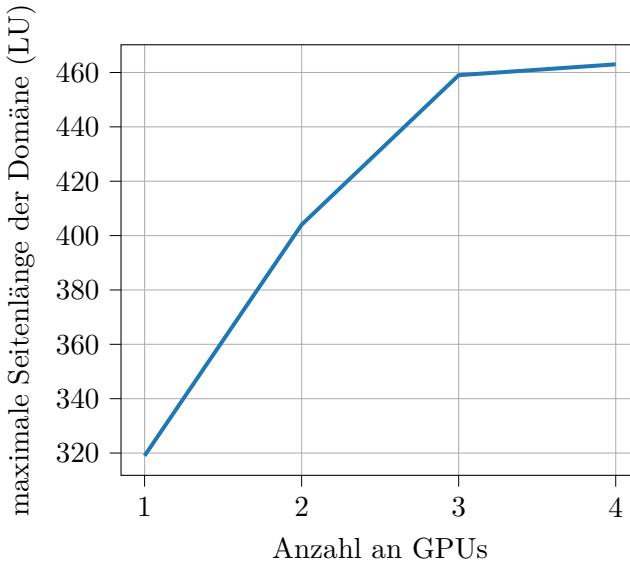


Abbildung 3.2.1.4-3: Darstellung der maximalen Seitenlänge einer würfelförmigen Simulationsdomäne in Gitterpunkten über der Anzahl beteiligter GPUs (V100 auf Knoten gpu4). Es ist zu erkennen, dass durch die Parallelisierung der Simulation auf mehr GPUs das Simulieren größerer Simulationsdomänen möglich wird. Die Simulation mit einer GPU wird mit der sequenziellen Simulation durchgeführt, die anderen mit der parallelisierten.

RAM eines Knotens sehr große Domänen ermöglicht, sind sie beim Simulieren mit GPUs stärker limitiert. Durch den Einsatz von vier statt einer GPU kann die mögliche Anzahl an Gitterpunkten jedoch zumindest auf rund 100 Millionen verdreifacht werden.

Auf Grund dieser Vorteile soll die Parallelisierung natürlich über diese Arbeit hinaus in Lettuce eingepflegt werden. Dazu sollte im Idealfall die Geschwindigkeit weiter optimiert werden. Hierbei könnte es hilfreich sein, auch mit der CPU-Parallelisierung auf das verbreitete MPI zu setzen und das zugehörige Environment hinsichtlich Hardware-Unterstützung zu vervollständigen. Dazu gibt es verschiedene Bibliotheken, die OpenMPI helfen, herstellerspezifische Kommunikationsprotokolle zu verwenden, die dem Programm mit größerem Aufwand hinzugefügt werden können. Zusätzlich besteht die Möglichkeit, die Funktionen der Klasse `DistributedSimulation` in die normale `Simulation`-Klasse zu integrieren, sodass der Wechsel von einer normalen Simulation zur parallelisierten weniger Aufwand für den Nutzer bedeutet. Zudem sollten Testfälle für die Parallelisierung hinzugefügt werden, um die korrekte Funktion der verteilten Elemente bei der weitergehenden Entwicklung sicherstellen zu können.

3.2.2 Randbedingungen

Da mit Lettuce bisher vor allem Testfälle, aber noch keine realistischen Anwendungen simuliert wurden, sind bisher nur sehr grundlegende Randbedingungen implementiert. Neben der Parallelisierung werden daher eine Anzahl an Randbedingungen implementiert, die im Rahmen der Arbeit benötigt werden. Dabei handelt es sich um die Zero-Gradient-Randbedingung [37] und die Non-Equilibrium-Extrapolation-Randbedingung [5, 30] mit vorgegebener Geschwindigkeit. Damit die Randbedingungen auch über die Arbeit hinaus in Lettuce verwendet werden können, fügen sie sich in das entsprechende Interface ein und werden unabhängig von der jeweiligen Position und Anzahl der Dimensionen verfasst. Der vollständige Code der Randbedingungen ist auf der beigelegten CD zu finden. Darüber hinaus werden die bereits implementierte Fullway-Bounce-Back-Randbedingung und die Equilibrium-Randbedingung mit Vorgabe des Drucks verwendet, welche bereits in Abschnitt 2.7 erläutert wurden. Im

Folgenden werden also zunächst die beiden implementierten Randbedingungen vorgestellt. Anschließend werden einige Schwierigkeiten, welche bei der Suche nach funktionierenden Randbedingungen aufgefallen sind, beleuchtet.

3.2.2.1 Null-Gradienten-Randbedingung

Die Null-Gradienten-Randbedingung [37] funktioniert, indem die Werte der einströmenden Verteilungen von denen des orthogonal zum Rand benachbarten Gitterpunkts an der Position x_{b-1} übernommen werden:

$$f_i(\mathbf{x}_b, t) = f_i(\mathbf{x}_{b-1}, t) \quad (3.2.2.1-1)$$

Durch das Übernehmen der Populationen verhält sich die Strömung, als wäre der Gradient der Verteilungsfunktion aus Richtung der Randbedingung null. Damit diese Randbedingung sinnvoll angewendet werden kann, ist es erforderlich, dass in der Nähe auch so bereits keine starken Gradienten auftreten.

3.2.2.2 Non-Equilibrium-Extrapolation-Randbedingung

Die Non-Equilibrium-Extrapolation-Randbedingung (NEEB) [30] wird von Hu et al. [38] als stabile Randbedingung für hohe Reynolds-Zahlen empfohlen. Zudem weist sie eine besitzt sie eine Genauigkeit 2. Ordnung auf und ist daher eine attraktive Option für die Einlassrandbedingung. [5]

Die NEEB basiert auf der in Abschnitt 2.7 vorgestellten Equilibrium-Randbedingung. Diese legt die Verteilungen am Rand der Domäne entsprechend den vorliegenden makroskopischen Größen auf die Equilibrium-Verteilung f^{eq} fest. Die NEEB erweitert diese Funktionsweise, indem sie zusätzlich den Nichtgleichgewichts-Anteil der Verteilungen an den Nachbarknoten ermittelt und hinzufügt [5, 30]:

$$f_i(\mathbf{x}_b, t) = f_i^{eq}(\rho_b, \mathbf{u}_b) + (f_i(\mathbf{x}_{b-1}, t) - f_i^{eq}(\rho_{b-1}, \mathbf{u}_{b-1})) \quad (3.2.2.2-1)$$

wobei x_b einen Gitterpunkt an dem die Randbedingung wirkt und x_{b-1} den orthogonal zum Rand benachbarten Gitterpunkt bezeichnet. Die zugehörigen makroskopischen Größen sind jeweils mit dem selben Index gekennzeichnet. Durch die Einbeziehung des Nichtgleichgewichts-Anteils der Populationen besitzt die NEEB den Vorteil einer höheren Genauigkeit im Vergleich zur Equilibrium-Randbedingung. [5]

Im Rahmen dieser Arbeit führt der Einsatz der NEEB zunächst zu Instabilitäten durch unphysikalische Druckwellen nach der Initialisierung der Simulation. Um diese zu reduzieren und zugleich auch andere unphysikalische Wellen besser dämpfen zu können, wird der Wert der Dichte, welcher den Equilibrium-Anteil mitbestimmt, wie von [39] vorgeschlagen, zusätzlich mit einem zeitlichen Tiefpassfilter erster Ordnung gefiltert:

$$\bar{\rho}(\mathbf{x}_i, t) = \frac{\rho(\mathbf{x}_i, t) + T_c \bar{\rho}(\mathbf{x}_i, t - \delta t)}{1 + T_c} \quad (3.2.2.2-2)$$

Die Filter-Zeitkonstante T_c wird nach entsprechenden Erprobungen auf 100 festgelegt.

3.2.2.3 Probleme der Randbedingungen

Über die hier vorgestellten Randbedingungen hinaus wurde auch eine Anzahl an weiteren Randbedingungen implementiert, mit denen jedoch keine stabile Simulation erreicht wurde, weshalb sie in der Simulation nicht verwendet werden. Darunter sind die Halfway-Bounce-Back-Randbedingung [5, 28], die Convective-Randbedingung [40], die Kinetic-Randbedingung

[41] (von [38] für hohe Reynolds-Zahlen empfohlen) und die besonders genaue Equilibrium-Extrapolation-Randbedingung [5, 42]. Die Instabilitäten treten dabei häufig an den Grenzen der verschiedenen Randbedingungen auf und scheinen durch hohe Reynolds-Zahlen begünstigt zu werden. Zusätzlich tritt bei vielen der Randbedingungen, die als Auslass in Erwägung gezogen werden, ein weiteres Problem auf. Passiert ein Wirbel die Randbedingung, kann es dazu kommen, dass in einem Teil des Wirbels Fluid in die Simulationsdomäne einströmt. Da viele der Randbedingungen, die den Druck vorgeben, die makroskopische Geschwindigkeit von den benachbarten Gitterpunkten extrapoliieren, kann sich hier eine Rückkopplungsschleife ausbilden. Am benachbarten Knoten ergibt sich eine Geschwindigkeit, die in die Simulationsdomäne hinein zeigt, die Randbedingung übernimmt die entsprechende Geschwindigkeit, sodass am Rand nun ebenfalls Fluid in die Domäne hineinströmt. Dieses einströmende Fluid strömt nun zu den benachbarten Gitterpunkten, von denen es die Randbedingung dann wieder übernimmt, sodass das Einströmen nicht abreißt, auch wenn der Wirbel die Domäne bereits verlassen hat. Dieser Effekt ist vor allem bei den größeren Wirbeln in zweidimensionalen Simulationen zu beobachten und wurde bei dreidimensionalen Simulationen, bei denen Wirbel meist in sehr kleine Strukturen zerfallen [5], nicht beobachtet.

Diese Effekte werden nicht näher untersucht, zeigen aber dass Randbedingungen eine wichtige Komponente einer Simulation mit der LBM darstellen und ihr Zusammenspiel insbesondere bei hohen Reynolds-Zahlen ein mögliches Forschungsfeld darstellt.

4 Untersuchung der Umströmung des Hauses

In diesem Kapitel soll auf die Durchführung und die Ergebnisse der Simulation der Umströmung der freistehenden Häuser mit Satteldach verschiedener Dachneigungen eingegangen werden. Dabei wird zunächst der Aufbau der Simulationsumgebung erläutert. Anschließend werden Ergebnisse der Simulation mit Referenzen abgeglichen, um die Qualität der Ergebnisse zu evaluieren. Abschließend werden die Ergebnisse vorgestellt und schließlich zusammengefasst und diskutiert.

4.1 Aufbau der Simulations-Umgebung

In diesem Abschnitt soll die genaue Konfiguration der Simulation erläutert werden. Dazu wird zunächst auf die allgemeinen Festlegungen eingegangen, bevor der Aufbau der Simulationsdomäne und die gewählten Parameter vorgestellt werden. Der Quellcode für die Simulationen kann auf der beigelegten CD eingesehen werden.

4.1.1 Allgemeine Konfiguration

Wie bereits in vorangegangenen Abschnitten erläutert, wird die Simulation, auf Grund der deutlich höheren Rechengeschwindigkeit pro Gerät, auf Grafikkarten, d.h. auf den NVIDIA V100 der gpu-Knoten des HBR5-Clusters durchgeführt. Die Simulation kann dabei leider nicht parallelisiert durchgeführt werden, da bei parallelisierter Simulation mit Konfiguration der Domäne für die Umströmung des Hauses nach einigen tausend Zeitschritten ein Fehler in einem der unterliegenden Software-Pakete auftritt, der zuvor noch nie aufgetreten war. Aus Zeitmangel konnte im Rahmen dieser Arbeit leider keine Fehlersuche mehr betrieben werden. Da der resultierende Fehler bei der Simulation hoher Reynolds-Zahlen üblich [43], mit 32 Bit-Variablen gerechnet. Dies bietet den Vorteil eines geringeren Speicherbedarfs und ermöglicht so eine größere Simulationsdomäne. Zugleich ermöglicht es auch eine höhere Rechengeschwindigkeit. Als Kollisionsoperator wird, wie bereits erwähnt, der KBC-Kollisionsoperator (vgl. Abschn. 2.5.2) verwendet, welcher bessere Stabilität bei unteraufgelösten Strömungen bietet als die Standardoption BGK. Es wird dabei mit dem D3Q27-Geschwindigkeitssatz gerechnet, um von dessen besserer Abbildung von Strömungen mit hohen Reynolds-Zahlen zu profitieren (vgl. Abschn. 2.2).

4.1.2 Aufbau der Simulations-Domäne

Die Domäne der Simulation ist aus zuvor vorgestellten Bestandteilen aufgebaut. In Abbildung 4.1.2-1 ist eine schematische Darstellung des räumlichen Aufbaus abgebildet. Im Folgenden werden die einzelnen Bestandteile, also die gewählten Randbedingungen, der Aufbau des Hauses und das Geschwindigkeitsprofil am Einlass der Domäne näher beleuchtet, bevor im nächsten Abschnitt auf die gewählten Parameter eingegangen wird.

Randbedingungen

An den jeweils nummerierten Seiten der Domäne werden die folgenden Randbedingungen verwendet:

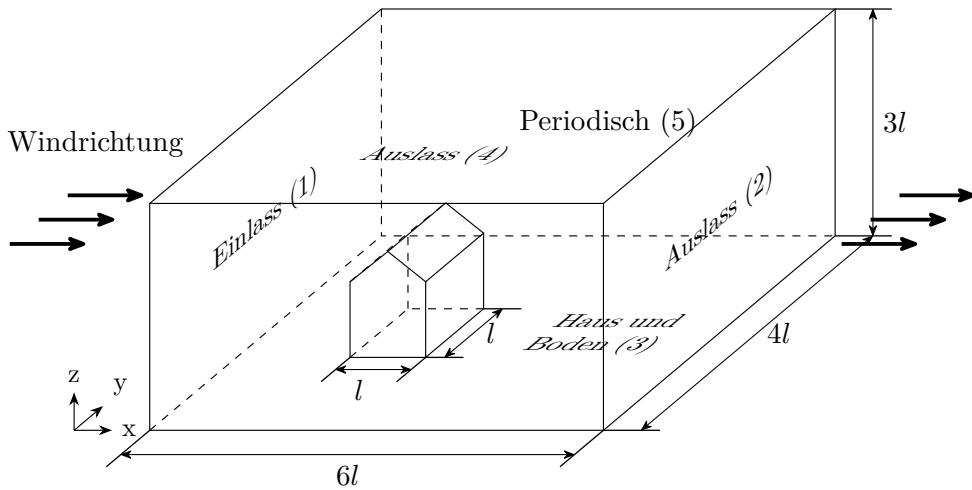


Abbildung 4.1.2-1: Übersicht über den Aufbau der Simulationsdomäne mit Koordinatenachsen; der äußere Quader stellt die Maße der gesamten Domäne als Vielfaches der Länge des Hauses l dar, während innen das Haus um den Punkt $(x = 2l, y = 2l, z = 0)$ aufgebaut ist. An den Seiten der Domäne sind die jeweiligen Funktionen der hier implementierten Randbedingung mit der Nummer der jeweiligen Erklärung eingetragen, die Seite gegenüber (5) ist dabei ebenfalls eine periodische Randbedingung.

1. Die *Non-Equilibrium-Extrapolation-Randbedingung* (vgl. Abschn. 3.2.2.2) bildet den Einlass der Domäne. Hier strömt die Luft mit vorgegebener Geschwindigkeit in die Domäne ein, dazu wird ein übliches exponentielles Windgeschwindigkeitsprofil eingesetzt, welches im folgenden Absatz erläutert wird.
2. Die *Equilibrium-Randbedingung* (vgl. Abschn. 2.7) wird als Auslass eingesetzt. Sie gibt einen Druck von $\rho^* = 1$ vor, der dem mittleren Druck in der Domäne entspricht und übernimmt die Geschwindigkeit des benachbarten Gitterpunkts, wodurch sie es den Populationen erlaubt die Domäne ungehindert zu verlassen. Sie wird hier vor allem eingesetzt, da sie gut mit den benachbarten Randbedingungen kompatibel ist.
3. Die *Fullway-Bounce-Back-Randbedingung* (vgl. Abschn. 2.7) bildet die festen Wände, also den Boden und das Haus als schlupffreie Randbedingung ab. Die genaue Konstruktion des Hauses wird im folgenden Absatz erläutert.
4. Die *Zero-Gradient-Randbedingung* (vgl. Abschnitt 3.2.2.1) bildet die Oberseite der Domäne. Sie sorgt hier ebenfalls für ein ungehindertes Ausströmen der Luft, indem sie die einströmenden Populationen vom benachbarten Gitterpunkt übernimmt.
5. *Periodische Randbedingungen* bilden die windparallelen Seiten der Domäne. Sie werden hier eingesetzt, da sie eine sehr gute Stabilität besitzen und sehr gut mit benachbarten Randbedingungen kompatibel sind. Dass die Strömung eigentlich nicht periodisch ist, stellt dabei kein Problem dar, da bei dieser Randbedingung nur großflächige Phänomene auftreten, die symmetrisch zur x-z-Ebene sind.

Die Wahl dieser Randbedingungen ist dabei, wie in Abschnitt 3.2.2.3 erläutert, durch fehlende Stabilität, insbesondere an Kanten zu anderen Randbedingungen, stark eingeschränkt. Dementsprechend konnten die Randbedingungen nicht hinsichtlich einer optimalen Fehlerordnung gewählt werden und es besteht noch Verbesserungspotential in dieser Hinsicht.

Aufbau des Hauses

Das Haus innerhalb der Domäne wird als Maske für die Bounce-Back-Randbedingung realisiert. Diese wird erzeugt, indem entsprechend der Längenmaße und dem Ursprung des Hau-

ses die zugehörigen Gitterpunkte ausgewählt werden. Um abgesehen von der Variation der Dachform möglichst einfache, vergleichbare Geometrien zu nutzen, besitzt das Haus eine quadratische Grundfläche, wie es auch in der Literatur häufig der Fall ist [3, 12]. Um den Einfluss des Geschwindigkeitsprofils auf die Strömung bei verschiedenen Neigungswinkeln möglichst einheitlich zu gestalten, wird die Firsthöhe des Hauses festgelegt und die Höhe des Untergeschosses nach Bedarf variiert. Das Haus zeichnet sich außerdem durch traufseitige Dachüberstände aus, die als Prismen ausgeführt sind. Um den Einfluss dieser Überstände zu untersuchen, wird die Simulation zusätzlich mit einem gleichen Haus ohne Überstände durchgeführt. Um ein breites Spektrum von Dachneigungen zu erfassen, werden in dieser Arbeit Dächer mit 20° , 35° und 50° Neigungswinkel untersucht. Die entsprechenden Häuser mit Überstand sind in Abbildung 4.1.2-2 dargestellt. Hier fällt auf, dass die Dachflächen nicht glatt, sondern stufig sind. Dies liegt daran, dass die eingesetzte Bounce-Back-Randbedingung keine Interpolation beherrscht und die Randbedingung daher parallel zum Gitter verlaufen muss. Die dadurch entstehende Struktur des Daches kann möglicherweise als Abbildung von Dachziegeln oder ähnlichen Rauigkeiten des Daches interpretiert werden.

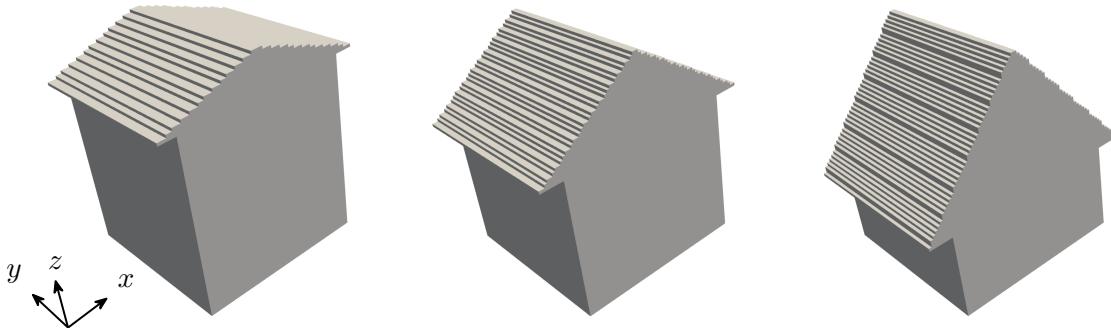


Abbildung 4.1.2-2: Übersicht der Häuser, welche im Rahmen der Druckuntersuchung umströmt werden. Sie unterscheiden sich in der Dachneigung, welche jeweils 20° , 35° und 50° beträgt (links nach rechts).

Windgeschwindigkeitsprofil

Um am Einlass eine realistische Windgeschwindigkeitsverteilung zu erreichen, wird ein exponentielles Geschwindigkeitsprofil verwendet, welches den in der Bautechnik üblichen Windprofilen entspricht [4, 44]:

$$u_x(z) = u_0 \left(\frac{z}{z_{Haus}} \right)^\alpha \quad (4.1.2-1)$$

Dabei wird für die finalen Simulationen des Profilexponent $\alpha = 0.25$ gewählt, was einem Wert in vorstädtischem Gebiet entspricht [44]. Als z_{Haus} wird die konstante Firsthöhe gewählt, damit der Einfluss des Geschwindigkeitsprofils auf die verschiedenen Dächer möglichst ähnlich ist. Insgesamt ergibt sich so das in Abbildung 4.1.2-3 dargestellte Profil. Die Anströmung geschieht dabei in dieser Arbeit nur von der hier dargestellten Seite, orthogonal zum Dachfirst. Bei der Initialisierung der Domäne wird die lokale Geschwindigkeit der gesamten Domäne nach diesem Profil initialisiert.

4.1.3 Simulations-Parameter

In diesem Abschnitt sollen die gewählten Parameter der Simulation vorgestellt werden. Dabei wird zunächst auf die physikalischen und dann auf die LBM-Parameter eingegangen.

Relevante physikalische Parameter der Luft sind hier die Luftdichte ρ und kinematische Viskosität ν , sowie die Strömungsgeschwindigkeit am Einlass in u_0 und die zugehörige Reynolds-

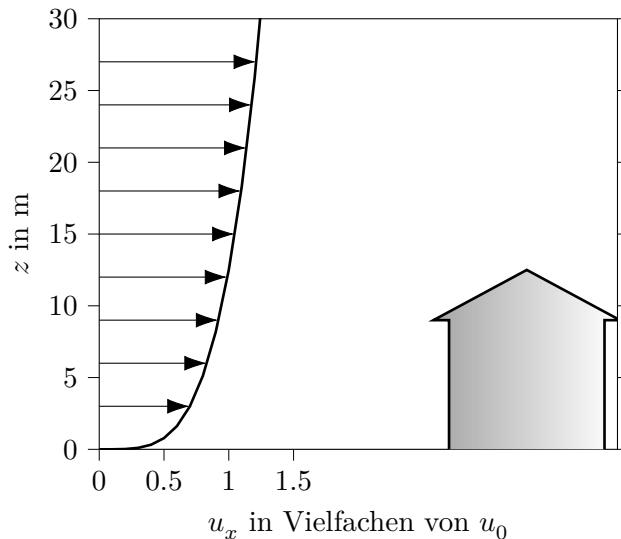


Abbildung 4.1.2-3: Visualisierung des Geschwindigkeitsprofils, welches am Einlass der Domäne angewendet wird. Die Größe von u_x ist dabei in Vielfachen der charakteristischen Geschwindigkeit u_0 , welche auf Höhe des Dachfirsts wirkt, angegeben. Um einen Eindruck der Höhe relativ zum Haus zu vermitteln, ist das Haus rechts im Bild dargestellt.

Zahl Re . Als Luftdichte wird $\rho = 1.225 \text{ kg/m}^3$ gewählt, was der Dichte bei 15°C und 1 atm entspricht. Passend hierzu ergibt sich eine kinematische Viskosität von $\nu = 14.85 \times 10^{-6} \text{ m}^2/\text{s}$. Um einen realistischen Sturm mit Potential zur Beschädigung von Gebäuden darzustellen, würden laut Beaufort-Skala Geschwindigkeiten von $u_0 > 20.8 \text{ m/s}$, bzw. eine Reynolds-Zahl von $\text{Re} > 14 \times 10^6$ benötigt. Bei Reynolds-Zahlen dieser Größenordnung sind mit der LBM stabile Simulationen jedoch kaum erreichbar [5]. In entsprechenden Erprobungen wurde die Stabilität stets von verwendeten Randbedingungen beschränkt. Basierend auf diesen Stabilitätsproben wird eine für die Untersuchung der Umströmung von Gebäuden übliche [7, 12] Reynolds-Zahl von $\text{Re} = 20000$ bzw. eine charakteristische Geschwindigkeit von $u_0 \approx 0.03 \text{ m/s}$ gewählt. Diese ist zwar deutlich langsamer, aber resultiert dennoch in einer turbulenten Umströmung, sodass gewonnene Erkenntnisse potentiell auf höhere Geschwindigkeiten übertragen werden können. Neben den Eigenschaften der Luft sind die Maße des Hauses und der Domäne von Bedeutung, welche mittels $l = 10 \text{ m}$ festgelegt werden. Dementsprechend betragen die Seitenlängen des Hauses 10 m , die Höhe des Dachfirsts 12.5 m und die Breite des Überstandes 1 m . Die Simulationsdomäne ist somit $60 \text{ m} \times 40 \text{ m} \times 30 \text{ m}$ groß.

Bei der Auswahl der LBM-Parameter wird wie in [5] vorgeschlagen vorgegangen (vgl. Abschn. 2.9). Dazu wird zunächst die mittlere Dichte, welche nur als Teil des Umrechnungsfaktors der Dichte $C_\rho = \frac{\rho_0}{\rho_0^*}$ relevant ist, als $\rho_0^* = 1$ festgelegt. Als nächstes wird die Größe der Simulationsdomäne als $6l \times 4l \times 3l$ festgelegt, wie bereits im vorangegangenen Abschnitt abgebildet. Dabei wird l^* , also die Seitenlänge des Hauses in LU, entsprechend des verfügbaren Arbeitsspeichers der Grafikkarten auf 60 festgelegt. Daraus ergibt sich die Gitterkonstante Δx als $1/6 \text{ m}$. Dementsprechend beträgt die Größe der Domäne $360 \text{ LU} \times 240 \text{ LU} \times 180 \text{ LU}$. Diese Maße werden gewählt, um sicherzustellen, dass genügend Abstand zwischen den Rändern der Domäne und den Turbulenzen am Haus vorhanden ist, sodass ihr Verhalten ungestört ist, und zugleich ein möglichst kleines Δx erreicht wird. Die Simulationsdomäne umfasst so rund 15.6 Millionen Gitterpunkte womit ein Zeitschritt auf einer NVIDIA V100 rund 0.3 s in Anspruch nimmt. Anschließend wird die Machzahl als 0.1 gewählt, um ein Mittel aus Stabilität und Genauigkeit zu erzielen. Die Relaxationszeit für die niedrigeren Momente ergibt sich dann als $\tau^* = \frac{1}{2} + \frac{\nu^*}{c_s^* \Delta x^*} \approx 0.50052$ bestimmt. Schließlich ergibt sich $\Delta t = C_t = C_l/C_u = \frac{\Delta x}{\Delta x^*} / \frac{u_0}{c_s^* \text{Ma}} = 0.324 \text{ s}$. Aus den gewählten Größen ergeben sich so auch

die nötigen Umrechnungsfaktoren zwischen physikalischen und Lattice-Einheiten, welche in Tabelle 4.1.3-1 zusammengefasst sind.

Tabelle 4.1.3-1: Übersicht über die grundlegenden Umrechnungsfaktoren zwischen physikalischen und Lattice-Einheiten bei den gewählten Parametern. Alle weiteren Umrechnungsfaktoren können als Kombination aus diesen ermittelt werden.

	Wert	Einheit
C_l	$l/l^* = 10 \text{ m} / 60 = 0.167 \text{ m}$	m
C_ρ	$\rho_0 / \rho_0^* = 1.225 \text{ kg/m}^3 / 1 = 1.225 \text{ kg/m}^3$	kg/m ³
C_u	$\mathbf{u}_0 / (c_s^* \text{Ma}) = 0.0297 \text{ m/s} / (0.1 / \sqrt{3}) = 0.514 \text{ m/s}$	m/s

4.2 Verifizierung der Simulationsumgebung

Um zu überprüfen, wie realistisch und verlässlich die Ergebnisse der beschriebenen Simulation sind, werden sie mit den Ergebnissen einer anderen Arbeit verglichen. Dazu werden die Ergebnisse von Tominaga et al. [3] herangezogen, da hier ebenfalls Satteldächer verschiedener Neigungswinkel untersucht werden und die Arbeit unter anderem das Ziel hat, Daten für die Validierung numerischer Modelle bereitzustellen. Um die Realitätsnähe der Simulationsergebnisse zu quantifizieren, werden sie im Folgenden zunächst an verschiedenen Stellen der Domäne mit den Windkanal-Messergebnissen der Referenz zu Strömungsgeschwindigkeit, Druck und kinetischer, turbulenter Energie verglichen. Um auch die weitere Verteilung von Druck und Strömungsgeschwindigkeit abzugleichen, werden anschließend entsprechende Längsschnitte der Domäne mit den Simulationsergebnissen der Referenz gegenübergestellt.

Um dafür eine Vergleichbarkeit der Ergebnisse zu schaffen, werden einige Parameter der Simulation innerhalb dieses Abschnitts an die der Referenz angepasst. Dementsprechend besitzen die für den Abgleich simulierten Häuser keine Dachüberstände und haben Neigungswinkel von 16.7°, 26.6° und 36.9°. Zwischen den verschiedenen Häusern ist dabei anstelle der Höhe des Firsts die Höhe des Erdgeschosses konstant, die sich dabei zu $z_{Haus} = l / 1.1 = 9.09 \text{ m}$ bzw. 54.5 LU ergibt. Dementsprechend wird auch das Geschwindigkeitsprofil der Anströmung angepasst, sodass die Referenzhöhe der Höhe des Erdgeschosses entspricht. Die Reynolds-Zahl liegt dabei in der Referenz bei 3.5×10^4 , beim diesem Wert kann jedoch keine stabile Simulation erreicht werden, dementsprechend wird in diesem Abschnitt weiterhin mit $Re = 2 \times 10^4$ gerechnet.

In der Referenz werden dabei über einen Beobachtungszeitraum gemittelte Werte betrachtet. Dementsprechend werden die Simulationsergebnisse aus einem Beobachtungszeitraum von 1×10^5 bis 4×10^5 Zeitschritten (rund 26 h in physikalischen Einheiten) für den Abgleich ebenfalls gemittelt. Die ersten 1×10^5 Zeitschritte werden verworfen, damit unphysikalische Effekte abklingen können und nur der eingeschwungene Zustand betrachtet wird.

In Abbildung 4.2-1 wird die normierte Strömungsgeschwindigkeit u_x/u_0 aus den Simulationsergebnissen an mehreren Punkten der mittleren x-z-Ebene der Simulationsdomäne mit den Windkanalmessungen von [3] verglichen. Hierbei ergibt sich eine mittlere Abweichung von rund 8 %, was geringer ist als die angegebenen 15 % der Simulationen der Referenz. Somit zeigt sich, dass die Strömungsgeschwindigkeit in der Simulationsdomäne vergleichsweise gut wiedergegeben wird. Zum Abgleich der auftretenden Wirbelstruktur bei den verschiedenen Dachneigungswinkeln werden in Abbildung 4.2-2 zusätzlich die Stromlinien in der betreffenden Ebene den Ergebnissen der Simulation der Referenz gegenübergestellt. Hier zeigt sich, dass die generelle Struktur der Strömung der hier erreichten Ergebnisse denen der Referenz sehr nahe kommt.

In Abbildung 4.2-3 wird der Vergleich mit den Windkanalmessungen für die normierte turbulente kinetische Energie k/u_0^2 durchgeführt, wobei k der halben Summe der Varianz

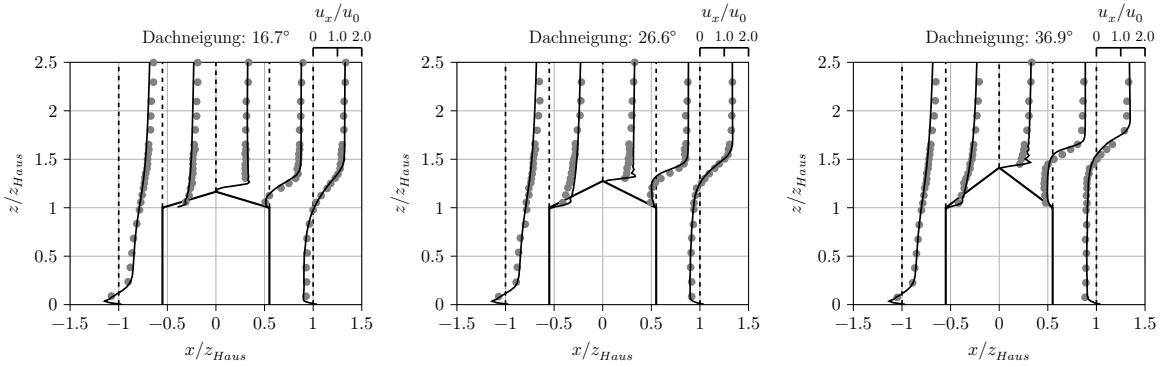


Abbildung 4.2-1: Vergleich der zeitlich gemittelten normierten Geschwindigkeit in x-Richtung u_x/u_0 der in dieser Arbeit erzeugten Simulationsergebnisse (Linie) mit den Windkanal-Messwerten der Referenz [3] (Punkte) an verschiedenen Stellen eines Längsschnitts der Domäne durch die Mitte des Hauses bei den verschiedenen Dachneigungen. Die normierte Geschwindigkeit wird entsprechend der Skala oben rechts an der x-Achse ab der jeweiligen gestrichelten Linie abgelesen. Die Anströmung verläuft von links nach rechts. Es ist eine gute Übereinstimmung der Simulationsergebnisse mit den Messergebnissen der Referenz erkennbar.

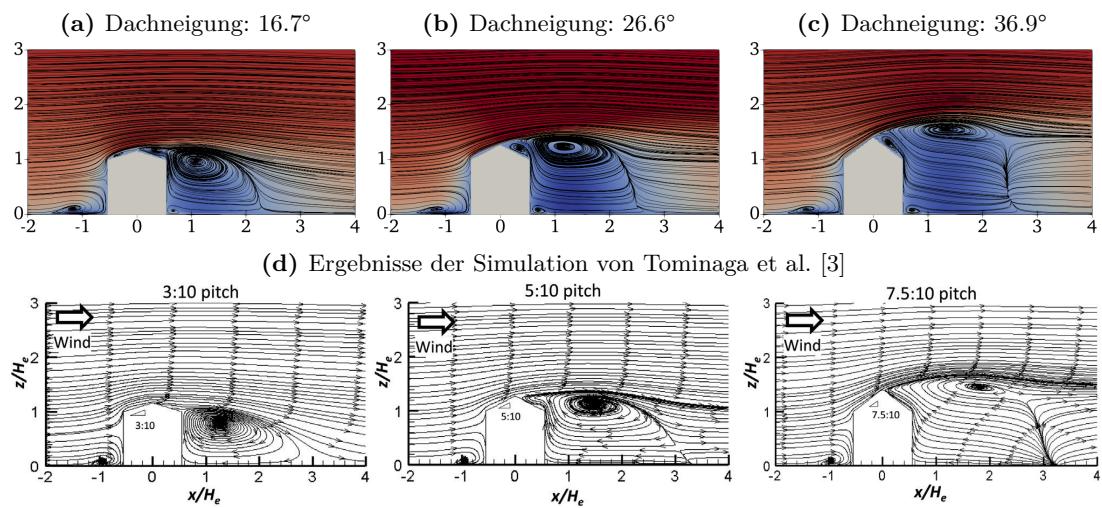


Abbildung 4.2-2: Oben: Darstellung der Stromlinien in einem zentralen Längsschnitt der Simulationsdomäne bei den verschiedenen Dachneigungen. Die Position in der Domäne ist in Vielfachen der charakteristischen Länge, also der Höhe des Hauses, bemäßt. Die Farbe ist ein Indikator für die lokale Geschwindigkeit, wobei rot schnellere und blau langsamere Bereiche markiert. Unten: Entsprechende Visualisierung der Simulationsergebnisse von Tominaga et al. aus [3] zum direkten Vergleich. Es zeigt sich, dass die Stromlinien beider Simulationen sehr gut übereinstimmen.

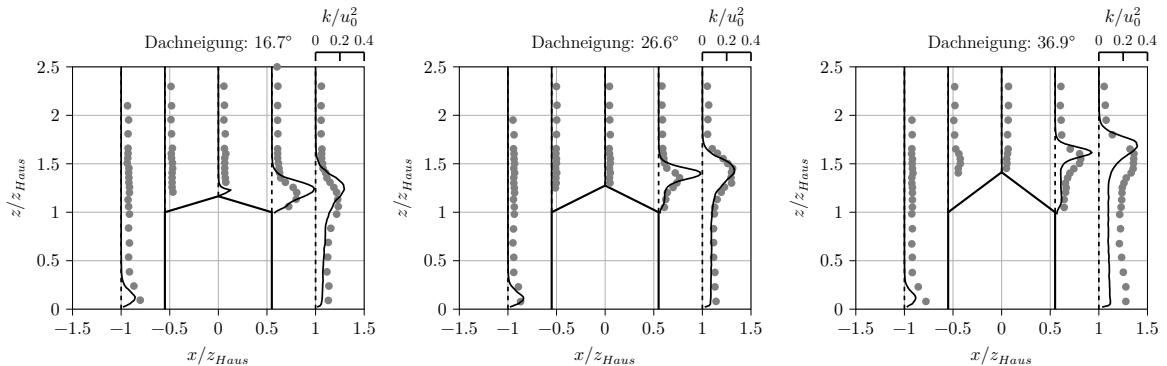


Abbildung 4.2-3: Vergleich der zeitlich gemittelten normierten turbulenten kinetischen Energie k/u_0^2 der in dieser Arbeit erzeugten Simulationsergebnisse (Linie) mit den Windkanal-Messwerten der Referenz [3] (Punkte) an verschiedenen Stellen eines Längsschnitts der Domäne durch die Mitte des Hauses bei verschiedenen Dachneigungen. Die normierte turbulente kinetische Energie wird entsprechend der Skala oben rechts an der x-Achse ab der jeweiligen gestrichelten Linie abgelesen. Die Anströmung verläuft von links nach rechts. Es große Abweichungen der Simulationsergebnisse von den Messergebnissen der Referenz erkennbar.

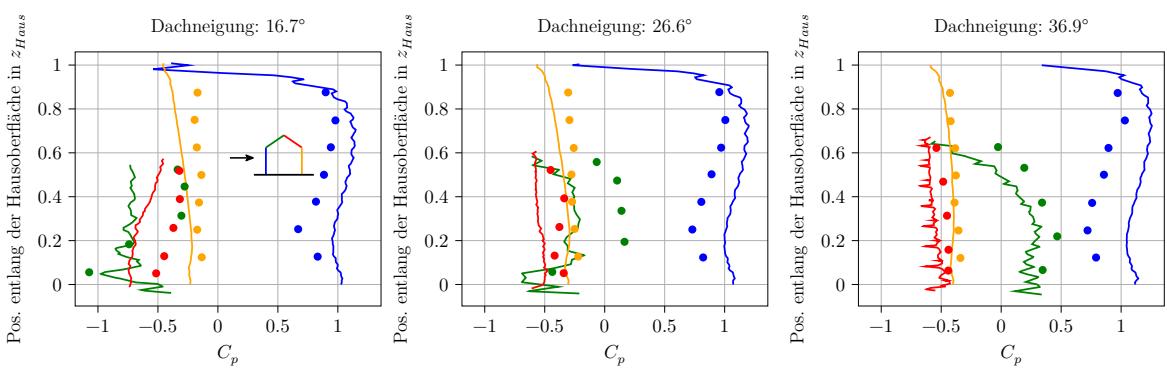


Abbildung 4.2-4: Vergleich des zeitlich gemittelten Druckkoeffizienten $C_p = p/(0.5\rho_0 u_0^2)$ entlang des Querschnitts in der Mitte des Hauses der in dieser Arbeit erzeugten Simulationsergebnisse (Linie) mit den Windkanal-Messwerten der Referenz [3] (Punkte) bei verschiedenen Dachneigungen. Die Zuordnung der Farben zu den Seiten des Querschnitts ist in der linken Abbildung dargestellt, der Pfeil zeigt die Windrichtung an. Es ist zu erkennen, dass die Simulationsergebnisse von den Werten der Messergebnisse der Referenz abweichen, aber jeweils einen ähnlichen Verlauf beschreiben. Es ist auch zu erkennen, dass die Abweichungen zwischen beiden mit wachsendem Dachneigungswinkel abnehmen.

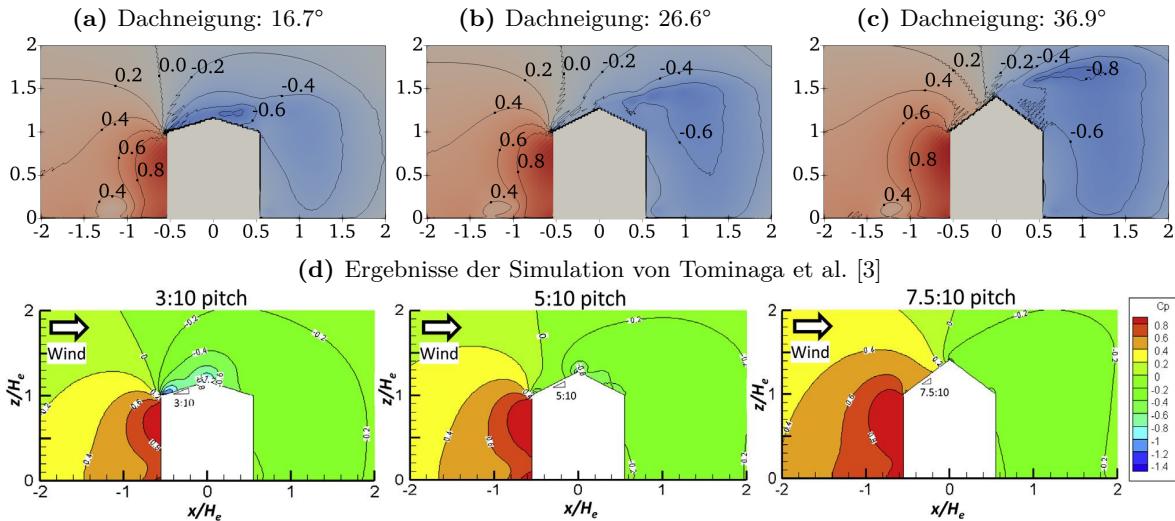


Abbildung 4.2-5: Oben: Darstellung des zeitlich gemittelten Druckkoeffizienten $C_p = (p - p_0)/(0.5\rho_0 u_0^2)$ der Simulationsergebnisse dieser Arbeit in einem zentralen Längsschnitt der Simulationsdomäne bei den verschiedenen Dachneigungen. Die Position in der Ebene ist in Vielfachen der charakteristischen Länge, also der Höhe des Hauses, bemäst. Die schwarzen Linien stellen Linien mit gleichem Druckkoeffizient dar, wobei die jeweiligen Werte in der Grafik eingetragen sind. Die Farbe indiziert ebenfalls den Druckkoeffizient, sie wechselt mit wachsendem Druckkoeffizient von blau zu rot. Unten: entsprechende Visualisierung der Simulationsergebnisse von Tominaga et al. aus [3] zum direkten Vergleich. Es zeigt sich, dass die Werte sich unterscheiden, die qualitative Form der Verteilung jedoch ähnlich ist.

aller drei Geschwindigkeitsanteile entspricht. Dabei sind größere Unterschiede erkennbar, die sich auch in einer mittleren relativen Abweichung von 80% äußern. Diese Abweichungen sind jedoch zu erwarten, da der verwendete Einlass von [3] bereits Turbulenz erzeugt, was der in dieser Arbeit verwendete Einlass nicht unterstützt. So entstehen Turbulenz und turbulente kinetische Energie erst durch die Interaktion mit dem Haus. Dies wird dadurch bestätigt, dass die gemittelte relative Abweichung an der 5. Messstelle, also nach Passieren des Hauses, statt bei 100 % nur noch bei etwa 50 % liegt.

In Abbildung 4.2-4 werden die Werte für den Druckkoeffizienten $C_p = (p - p_0)/(0.5\rho_0 u_0^2)$ entlang der Außenseite des mittigen Gebäudequerschnitts verglichen. Dabei ist zu beachten, dass auf Grund des stufigen Daches in den Simulationsergebnissen nicht direkt an der Oberfläche gemessen werden kann, sondern in 3 LU Abstand gemessen wird. Im Vergleich ist zu erkennen, dass die Entwicklung der Werte zwar qualitativ ähnlich ist, quantitativ aber vor allem auf den beiden flacheren Dächern größere Abweichungen aufweist. Dies zeigt sich auch in der mittleren relativen Abweichung von ca. 44 %, die bei dem flachen Dach jedoch 59 %, beim mittleren 44 % und beim steileren 28% beträgt. Es fällt außerdem auf, dass die Ergebnisse an der Luvseite des Daches mit einer relativen Abweichung von 92 % die größten Abweichungen aufweisen, lässt man sie außen vor, so beträgt die mittlere relative Abweichung nur noch 35%. Eine mögliche Ursache für diese Abweichungen kann die stufige Oberfläche des Daches darstellen. An den Kanten des Daches treten lokale Druck-Extrema auf, welche den Wert nahe der Oberfläche verfälschen und dadurch auch die Messung in etwa 30 cm Abstand erforderlich machen. Ein weiter Einflussfaktor könnte die eingesetzte Fullway-Bounce-Back-Randbedingung sein, die eine geringere zeitliche Genauigkeit besitzt als die Halfway-Alternative (vgl. Abschn. 2.7). Die zu geringe Reynolds-Zahl kann hier ebenfalls einen Einfluss haben. In Abbildung 4.2-5, wo der Druckkoeffizient über den gesamten Querschnitt der Simulationsdomäne im Vergleich zu den Simulationsergebnissen der Referenz dargestellt ist, zeigt sich, dass auch die Druckverteilung in größerer Entfernung vom Haus zwar qualitativ ähnlich ist, aber größere

Extremwerte aufweist, wobei auch hier die Unterschiede nimmt mit zunehmender Dachneigung abnehmen. Es fällt außerdem ein fehlender Druckabfall in der Nähe des Einlasses auf. Eine mögliche Ursache dafür kann der Tiefpassfilter der Dichte in der Einlassrandbedingung sein, der eventuell Druckdynamiken in der Nähe stört.

Insgesamt lässt sich sagen, dass die aufgebaute Simulation die turbulente Umströmung, insbesondere hinsichtlich der Position und Form von Wirbeln sowie der Strömungsgeschwindigkeit sehr gut wiedergibt. Der Druckkoeffizient wird dagegen weniger gut wiedergegeben, insbesondere bei flacheren Dachneigungen und auf der Luvseite des Daches. Qualitativ weisen die Ergebnisse jedoch trotz des großen Unterschieds der jeweiligen Reynolds-Zahlen große Ähnlichkeit mit den Messergebnissen der Referenzen auf. Dementsprechend wird der Fokus bei den weiteren Untersuchungen vor allem auf qualitative Beobachtungen gelegt, die dabei scheinbar auch das Potential besitzen, für Reynolds-Zahlen über der der simulierten Strömung zuzutreffen.

4.3 Untersuchung der Umströmung des Hauses

In diesem Abschnitt sollen die Ergebnisse der zuvor erläuterten Untersuchung vorgestellt werden. Dabei werden Daten aus einem Beobachtungszeitraum von 1×10^5 bis 4×10^5 Zeitschritten untersucht, in dem die Geschwindigkeit mit einer Periode von 500 Zeitschritten (160 s) und der Druck mit einer Periode von 125 Zeitschritten (40 s) aufgezeichnet werden. Als übliche Größe in der Bautechnik wird dabei der Druck erneut in Form des Druckkoeffizienten $C_p = (p - p_0)/(0.5\rho_0 u_0^2)$ betrachtet, welcher aus der Abweichung vom Referenzdruck ermittelt wird, aus der auch die Windlast auf das Gebäude resultiert [4]. Auf Grund der im vorangegangenen Abschnitt festgestellten Störungen des Drucks nahe den Stufen der Dächer, wird der Druck dabei jeweils in 3 LU bzw. 50cm Abstand vom Dach untersucht. In Abbildung 4.3-1 sind die entsprechende Mittellinie sowie Ebenen parallel zu den Dachflächen dargestellt, in welchen der Druckkoeffizient im Folgenden betrachtet wird. Dabei werden die Ergebnisse entlang der Mittellinie stets beginnend auf der Luvseite aufgetragen.

Im Folgenden werden zunächst die Ergebnisse hinsichtlich des Einflusses der Dachneigung auf den Druck an der Dachoberfläche vorgestellt, bevor kurz auf die Ergebnisse hinsichtlich ihres Einflusses auf die Umströmung der Häuser eingegangen wird. Abschließend werden die Ergebnisse hinsichtlich des Einflusses der Dachüberstände auf diesen Druck präsentiert.

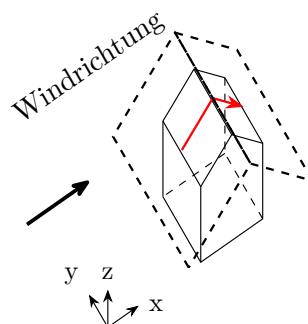


Abbildung 4.3-1: Visualisierung der Flächen und Mittellinie, auf welchen der Druckkoeffizient in der Untersuchung betrachtet wird, und ihrer Position relativ zum Haus.

4.3.1 Einfluss der Dachneigung auf den Druck an der Dachoberfläche

In Abbildung 4.3.1-1 ist der Verlauf des über den Beobachtungszeitraum zeitlich gemittelten Druckkoeffizienten, sowie seiner Extrema und Standardabweichung entlang der Mittellinie des Daches aufgetragen. In Abbildung 4.3.1-2, 4.3.1-3, 4.3.1-4 und 4.3.1-5 sind die selben

Größen in der entsprechenden Ebene über dem Dach dargestellt, um einen Überblick über die räumliche Verteilung der Drücke zu erlangen.

Bei Betrachtung des gemittelten Druckkoeffizienten zeigt sich, dass auf der Leeseite des Daches bei allen Neigungen Sog herrscht. Dieser ist bei 20° Neigungswinkel am stärksten und bei 35° am schwächsten, wobei hier der Unterschied zum steilsten Dach gering ist. Auf der Luvseite herrscht bei den Dächern mit 35° und 50° Neigungswinkel ein Überdruck, der beim steileren Dach deutlich stärker ist. Im Gegensatz dazu herrscht bei dem Dach mit 20° Neigungswinkel auch auf der Luvseite Sog. Auch ist hier an der Traufe ein Bereich mit stärkerem Sog zu erkennen, der möglicherweise aus einer Ablösung der Strömung in diesem Bereich resultiert. In der zugehörigen Darstellung der Verteilung des mittleren Druckkoeffizienten über die Dachfläche in Abbildung 4.3.1-2 zeigt sich, dass die Verteilung seitlich der Mittellinie bei den steileren Dächern weitgehend homogen ist, wobei der Druck auf der Luvseite nahe den Stirnseiten etwas nachlässt. Beim Dach mit 20° Neigungswinkel sind dagegen nahe den luvseitigen Stirnseiten größere Bereiche mit Überdruck zu erkennen, welchen auf der Leeseite entsprechende Bereiche mit stärkerem Sog gegenüberstehen.

Die in Abbildung 4.3.1-1 aufgetragene Standardabweichung des Druckkoeffizienten gibt Einblick in seine Schwankungen. Hier zeigt sich, dass der Druck beim Dach mit 20° Neigungswinkel deutlich stärker variiert als bei den anderen, wobei beim Dach mit 35° Neigungswinkel insgesamt die wenigsten Schwankungen auftreten. Bei allen Neigungswinkeln liegt dabei ein lokales Maximum im Bereich um den Dachfirst vor, während weitere an den luv- und leeseitigen Traufen auftreten. Dabei nimmt die Standardabweichung bei allen Dächern über die Dachlänge zu. Bei den steileren Dächern geschieht dies, abgesehen vom Maximum auf Höhe des Firsts, linear. Beim Dach mit 20° Neigungswinkel tritt dagegen am First eine sprunghafte Erhöhung der Standardabweichung auf. In der Darstellung des Drucks auf der Dachebene kann man hier erkennen, dass am Maximum in der Mitte des Daches ein etwa dreieckiger Bereich auf der Leeseite des Daches beginnt, der eine höhere Standardabweichung aufweist. Dies lässt vermuten, dass sich die entsprechenden Schwankungen von diesem Punkt aus ausbreiten.

Um Einblick in das Ausmaß dieser Schwankungen zu gewinnen, sind in Abbildung 4.3.1-1 zusätzlich die Extremwerte, die über den betrachteten Zeitraum aufgetreten sind, aufgetragen. Hier fällt auf, dass das Dach mit 20° Neigungswinkel auf der Leeseite sowohl die größten Maxima als auch die kleinsten Minima aufweist, was sich mit der hohen Standardabweichung in diesem Bereich deckt. Wie in Abbildung 4.3.1-4 und 4.3.1-5 zu erkennen, treten diese Extremwerte erneut in einem Bereich auf, der mittig am First beginnt und sich von dort über die Leeseite ausbreitet. Bei den beiden steileren Dächern fällt ein deutlich gleichmäßigeres Verhalten auf, ihre Extremwerte begrenzen ein schmales Druckspektrum und weisen nur geringe Variation zwischen verschiedenen Positionen der Leeseite auf. Auf der Luvseite zeigen beide Dächern ebenfalls ein sehr gleichmäßiges Verhalten, die Extremwerte weichen nur geringfügig von den Mittelwerten ab und zeigen einen sehr ähnlichen Verlauf entlang des Daches. Die zweidimensionale Darstellung über die Dachfläche zeigt dabei, dass diese Gleichmäßigkeit auch in den weiteren Bereichen der Dächer gegeben ist.

4.3.2 Einfluss der Dachneigung auf die umgebende Strömung

Neben den Untersuchungen zum Druck, werden beim Vergleich der Umströmung der Häuser mit den verschiedenen Dachneigungen auch einige Unterschiede der Anordnung der Wirbel in der Umgebung festgestellt, die in diesem Abschnitt kurz vorgestellt werden. In Abbildung 4.3.2-1 sind dazu die Stromlinien in einem vertikalen Längsschnitt parallel zur x-z-Ebene, welcher mittig durch das Haus verläuft, dargestellt. Die Stromlinien verdeutlichen dabei die lokale Bewegungsrichtung der Strömung und werden jeweils aus den x- und z-Komponenten der gemittelten Geschwindigkeit bestimmt.

Im Vergleich der Strömung bei den verschiedenen Dachneigungen fällt auf, dass die Wirbel stromabwärts des Hauses je nach Dachneigung verschiedene Positionen und Formen haben.

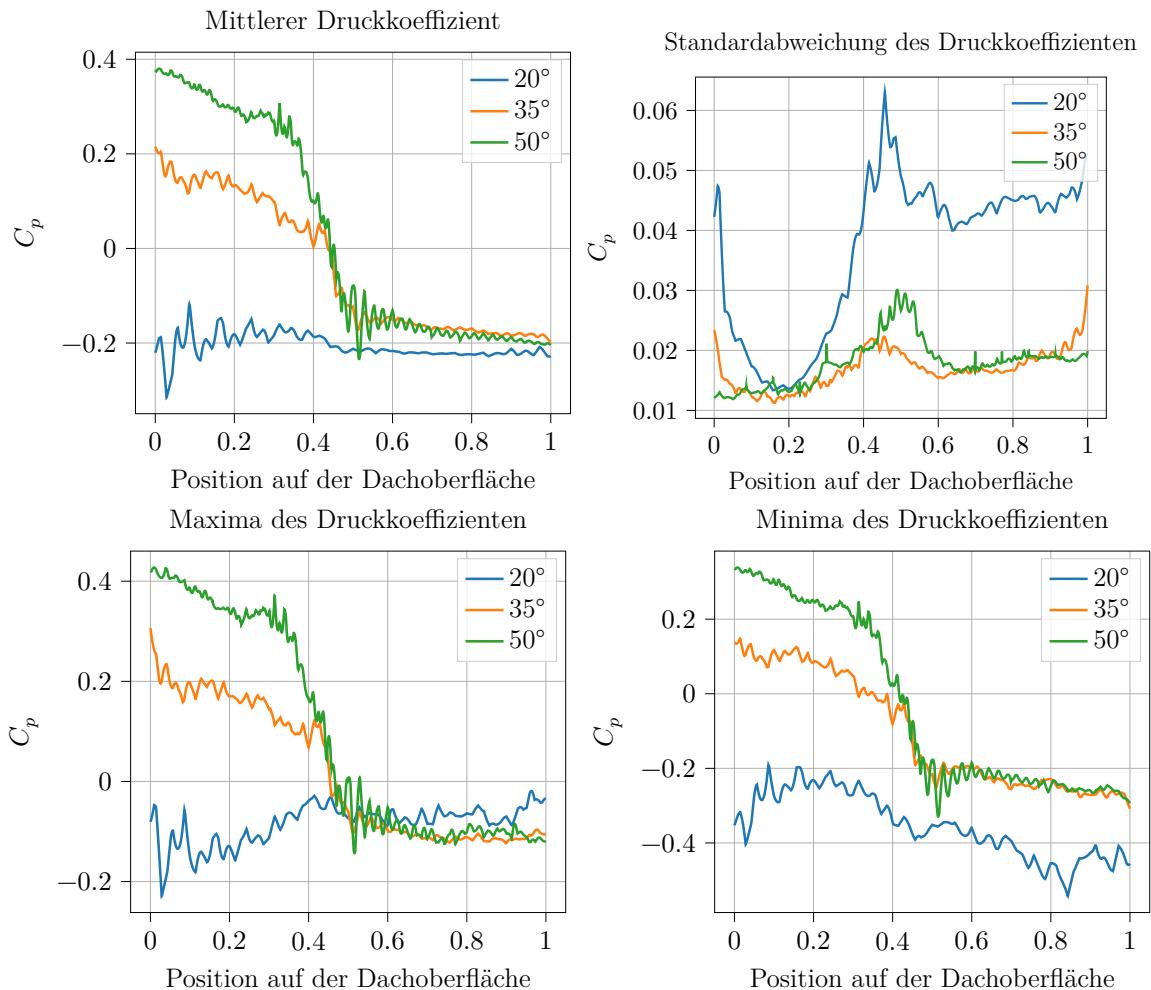


Abbildung 4.3.1-1: Vergleich des Mittelwerts, der Standardabweichung und der Extrema des Druckkoeffizienten über den Beobachtungszeitraum bei den drei untersuchten Neigungswinkeln. Die Werte sind entlang der Mittellinie des Daches aufgezeichnet, die Position ist dabei beginnend an der luvseitigen Traufe anteilig an der Gesamtlänge der Mittellinie des Daches abgegeben.

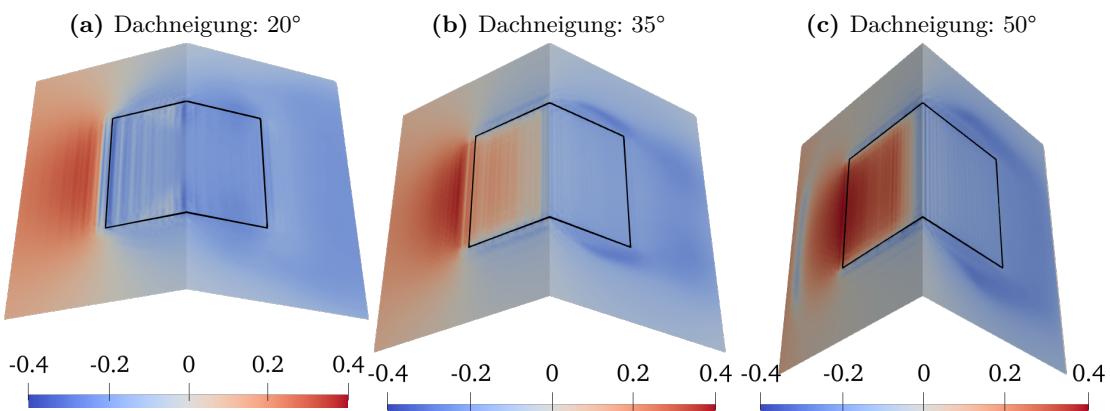


Abbildung 4.3.1-2: Vergleich der Verteilung des über den Beobachtungszeitraum gemittelten Druckkoeffizienten über die Dachfläche bei den verschiedenen Dachneigungen. Der schwarze Rand markiert die Umrisse des Dachs. Es ist zu erkennen, dass beim Dach mit 20° Neigungswinkel seitlich der Mittellinie eine inhomogene Verteilung des Drucks vorliegt, während sie bei den steileren Dächern homogener ist.

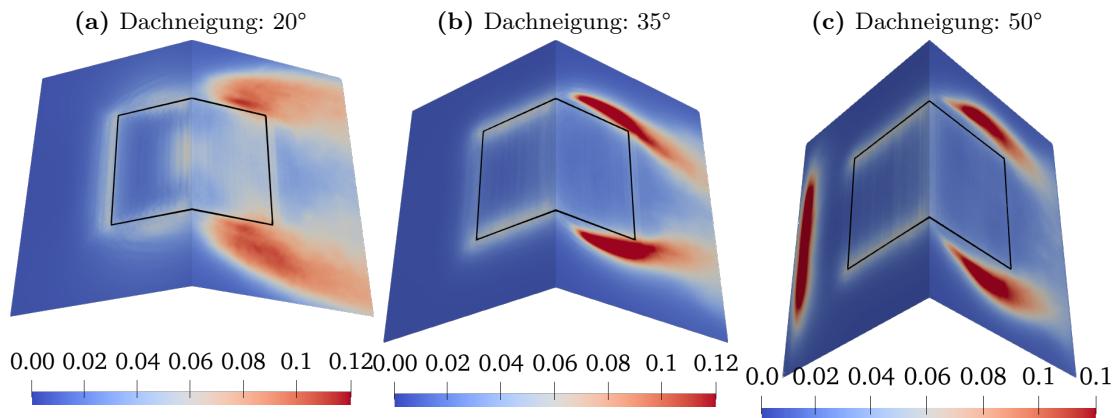


Abbildung 4.3.1-3: Vergleich der Verteilung der Standardabweichung des Druckkoeffizienten über den Beobachtungszeitraum über die Dachfläche bei den verschiedenen Dachneigungen. Der schwarze Rand markiert die Umrisse des Dachs. Beim Dach mit 20° Neigungswinkel ist mittig am First ein Maximum zu erkennen, an dem ein Bereich mit höherer Standardabweichung beginnt, der den Großteil der Leeseite umfasst. Bei den steileren Dächern ist eine homogenere Verteilung zu erkennen.

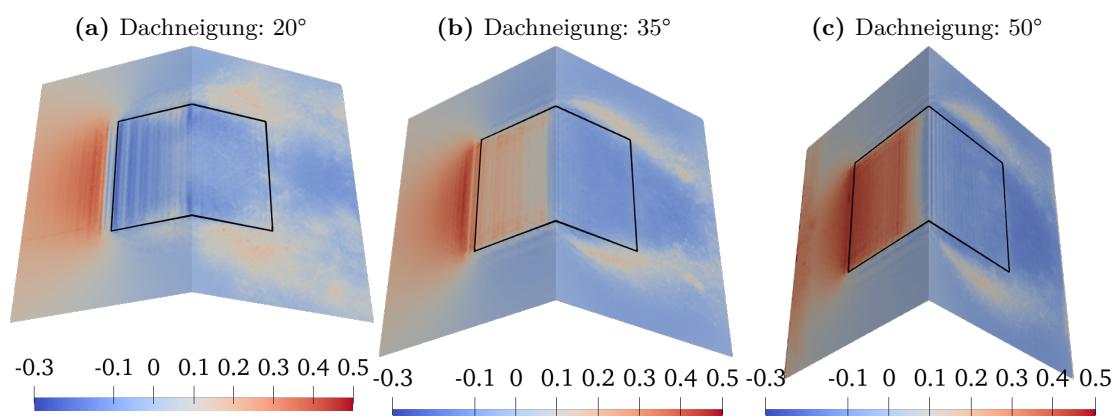


Abbildung 4.3.1-4: Vergleich der Verteilung der innerhalb des Beobachtungszeitraums aufgetretene Maximalwerte des Druckkoeffizienten über die Dachfläche bei den verschiedenen Dachneigungen. Der schwarze Rand markiert die Umrisse des Dachs. Es ist zu erkennen, dass die stärksten Druckmaxima der Leeseite beim Dach mit 20° Neigungswinkel auftreten und eine inhomogene Verteilung aufweisen, während die Verteilung bei den steileren Dächern homogener ist.

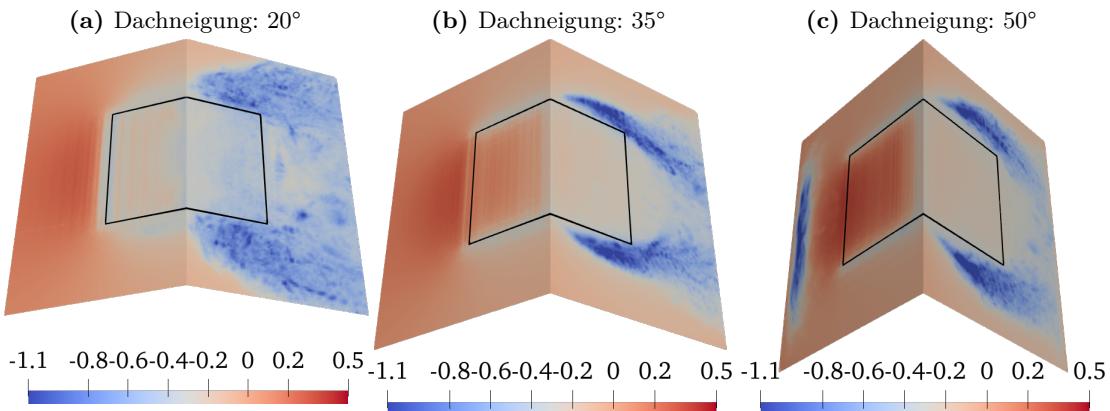


Abbildung 4.3.1-5: Vergleich der Verteilung der innerhalb des Beobachtungszeitraums aufgetretene Minimalwerte des Druckkoeffizienten über die Dachfläche bei den verschiedenen Dachneigungen. Der schwarze Rand markiert die Umrisslinie des Daches. Es ist zu erkennen, dass die stärksten Druckminima der Leeseite des Daches mit 20° Neigungswinkel auftreten und eine inhomogene Verteilung aufweisen, während die Verteilung bei den steileren Dächern homogener ist.

Das lässt darauf schließen, dass die Dachneigung einen großen Einfluss auf die Strömung stromabwärts des Hauses hat. Beim Dach mit 20° Neigungswinkel, liegt ein großer Nachlaufwirbel dicht hinter der leeseitigen Traufe. Diese Nähe kann hierbei eine mögliche Erklärung für die starken Schwankungen sein, die hier im vorangegangenen Abschnitt beobachtet wurden. Zwischen 20° und 35° ergibt sich nun eine starke Änderung der Anordnung der Wirbel. Das Zentrum des ersten Wirbels liegt bei der Umströmung des Daches mit 35° Neigungswinkel höher und ist weiter von der Traufe entfernt. Unter diesem Wirbel liegt eine Zone, in welcher die Strömung zunächst parallel zur Dachneigung entgegen der Windrichtung fließt und sich dann über dem Dach zu einem kleineren Wirbel staut. Die Strömung beim Dach mit 50° Neigung ist ähnlich, jedoch befindet sich der große Wirbel hier weiter stromabwärts, sodass der Bereich, in dem die Strömung zurückströmt, größer ist. Der Wirbel am Ende dieser Strömung liegt nun außerdem unterhalb des Dachüberstands.

Im Gegensatz zu diesen starken Unterschieden in der Strömung stromabwärts, sind stromaufwärts nur Änderungen in der Interaktion der Strömung mit dem Dach erkennbar. Beim Dach mit 20° Neigungswinkel löst die Strömung sich an der Traufe vom Dach ab und verläuft dann bis zum First in einem flachen Bogen oberhalb der Luvseite. Bei den beiden steileren Dächern scheint die Strömung dagegen direkt auf das Dach aufzutreffen und anschließend parallel zu verlaufen. Diese Unterschiede könnten ein Grund für die verschiedenen Druckverhältnisse sein, die hier im vorangegangenen Abschnitt beobachtet wurden. Die Ablösung der Strömung beim flachen Dach ist zudem eine Erklärung für den beobachteten Bereich mit stärkerem Sog nahe der luvseitigen Traufe.

Die Darstellung der Oberflächen gleichen Q-Werts in Abbildung 4.3.2-2 zeigt allerdings, dass diese großvolumigen Phänomene nur im Mittel auftreten. Hier ist zu erkennen, dass die Strömung tatsächlich aus einer Vielzahl kleiner Wirbel besteht, was auch zu der geringen räumlichen Ausdehnung der Extremwerte in Abbildung 4.3.1-4 und 4.3.1-5 passt.

4.3.3 Einfluss des Dachüberstands

Um den Einfluss des Dachüberstands zu untersuchen, wird die Simulation erneut ohne den Dachüberstand durchgeführt, um so einen Vergleich zwischen beiden Alternativen zu ermöglichen. Dabei ist zu beachten, dass für das Haus mit 20° Dachneigungswinkel auf Grund fehlender Stabilität der Simulation nur Ergebnisse aus dem Zeitraum zwischen 1×10^5 und 2.5×10^5 Zeitschritten genutzt werden können. Bei allen Darstellungen ist außerdem zu be-

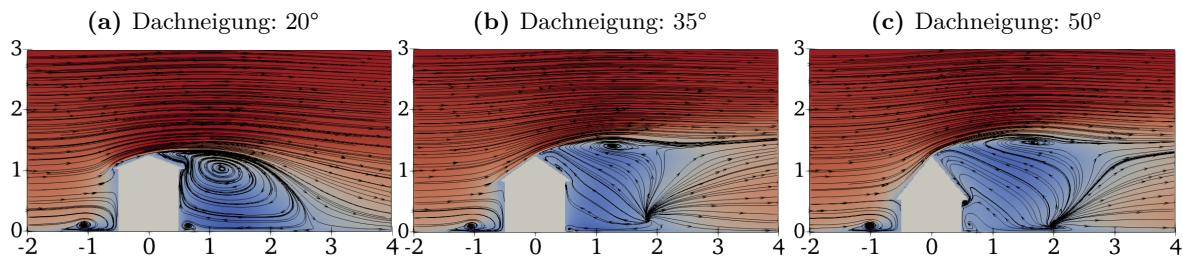


Abbildung 4.3.2-1: Vergleich der Verteilung der Strömungsgeschwindigkeit der beobachteten Strömung bei den Verschiedenen Dachneigungen. Darstellung der Stromlinien in einem zentralen Längsschnitt parallel zur x-z-Ebene der Simulationsdomäne bei den verschiedenen Dachneigungen. Die Koordinaten sind in Vielfachen der Länge des Hauses l angegeben. Die Farbe ist ein Indikator für die normierte lokale Geschwindigkeit, wobei rot schnellere und blau langsamere Bereiche markiert. Es ist eine starke Änderung der Wirbelanordnung zwischen 20° und 35° Neigungswinkel zu erkennen.

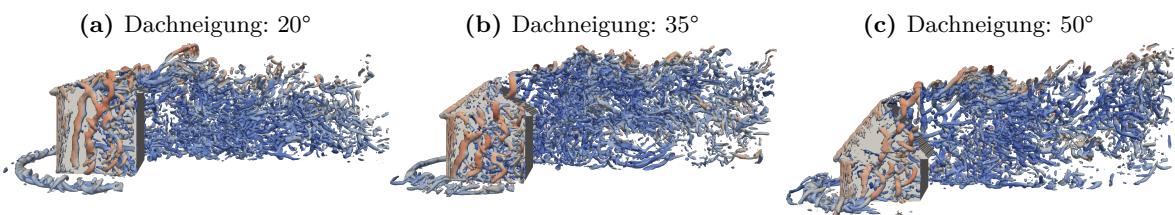


Abbildung 4.3.2-2: Darstellung eines Zeitschritts der betrachteten Strömung als Oberflächen gleichen Q's ($Q = 4 \times 10^{-6}$) zur Verdeutlichung der tatsächlichen Größe der Wirbelstrukturen. Für Einblick in das Innere der Turbulenz ist rechts des Hauses nur die hintere Hälfte der Strömung, ab einem Schnitt parallel zur x-z-Ebene, dargestellt. Die Farbe ist ein Indikator für die normierte lokale Geschwindigkeit, wobei rot schnellere und blau langsamere Bereiche markiert.

achten, dass das Dach ohne Überstand eine geringere Breite von Traufe zu Traufe besitzt.

In Abbildung 4.3.3-1 ist ein Vergleich des zeitlich gemittelten Druckkoeffizienten entlang der Mittellinie der Dachfläche aufgetragen. Der Vergleich der Änderungen, welche der Dachüberstand bei den jeweiligen Neigungswinkeln verursacht, lässt darauf schließen, dass sein Einfluss auf die Druckentwicklung mit wachsendem Dachneigungswinkel abnimmt. Während bei 20° Neigungswinkel noch große Änderungen zu erkennen sind, sind beide Verläufe bei 50° nahezu identisch. Im Weiteren werden dementsprechend die Auswirkungen bei 20° und 35° Neigungswinkel näher betrachtet.

Die Werte für einen weitergehenden Vergleich des Druckkoeffizienten mit und ohne Dachüberstände am Dach mit 20° Neigungswinkel sind in Abbildung 4.3.3-2 dargestellt. Hier ist zu erkennen, dass sowohl Mittelwert als auch Extrema auf der Luvseite des Daches mit Überstand näher am Referenzdruck liegen als beim Dach ohne Überstand. Demnach reduziert der Überstand hier also den Sog an der Luvseite. Während der Mittelwert an der Leeseite mit und ohne Überstand sehr ähnlich ist, ist hier eine Änderung der Standardabweichung zu erkennen. Beim Dach mit Überstand liegt eine deutlich geringere Standardabweichung vor als beim Dach ohne Überstand. Zugleich liegen die Extremwerte in diesem Bereich beim Dach mit Überstand auch näher am Mittelwert. Insgesamt lässt sich dies so interpretieren, dass der Überstand die Schwankungen auf der Leeseite reduziert und eine Art stabilisierende Wirkung besitzt.

Auf Grund der im vorangegangenen Abschnitt beim Dach mit 20° Neigungswinkel beobachteten Inhomogenitäten, wird hier zusätzlich die in Abbildung 4.3.3-3 zusammengefasste Auswirkung auf den Rest der Dachfläche verglichen. Hier zeigt sich, dass die festgestellten stärkeren Druckunterschiede, die ohne Überstand auftreten, sich auch auf den Großteil der Dachfläche seitlich der Mittellinie auswirken. Zudem ist zu erkennen, dass die zusätzlichen Schwankungen, die ohne den Überstand auftreten, sich ähnlich verhalten wie diejenigen beim Dach mit Überstand. In einem zentralen Bereich am First liegt eine besonders hohe Standardabweichung vor, die sich von dort über einen etwa dreieckigen Bereich auf der Leeseite des Daches zu verbreiten scheint. Dementsprechend sind in diesen Bereichen beim Dach ohne Überstand auch die stärkeren Extremwerte zu erkennen, welche der Dachüberstand sonst zu verringern scheint. Insgesamt scheint sich die räumliche Verteilung des Druckkoeffizienten über das Dach mit und ohne Dachüberstand also nur wenig zu unterscheiden.

Die Werte für die Untersuchung des Einflusses des Dachüberstands auf die Drücke am Dach mit 35° Neigungswinkel sind in Abbildung 4.3.3-4 zusammengefasst. Es fällt auf, dass hier sowohl der Mittelwert als auch beide Extrema beim Dach mit Überstand auf beiden Seiten des Daches etwas höher liegen. Dementsprechend wird durch den Überstand der Druck auf der Luvseite verstärkt und der Sog auf der Leeseite abgeschwächt. Ein Vergleich der Standardabweichungen zeigt erneut sehr ähnliche Werte auf der Luvseite und eine niedrigere Standardabweichung des Daches mit Überstand auf der Leeseite. Dies lässt darauf schließen, dass der Überstand bei diesem Neigungswinkel ebenfalls die Schwankungen an der Leeseite reduziert. Insgesamt fällt außerdem auf, dass die Unterschiede zwischen beiden Varianten hier wie erwartet geringer sind als beim Dach mit einem Neigungswinkel von 20° .

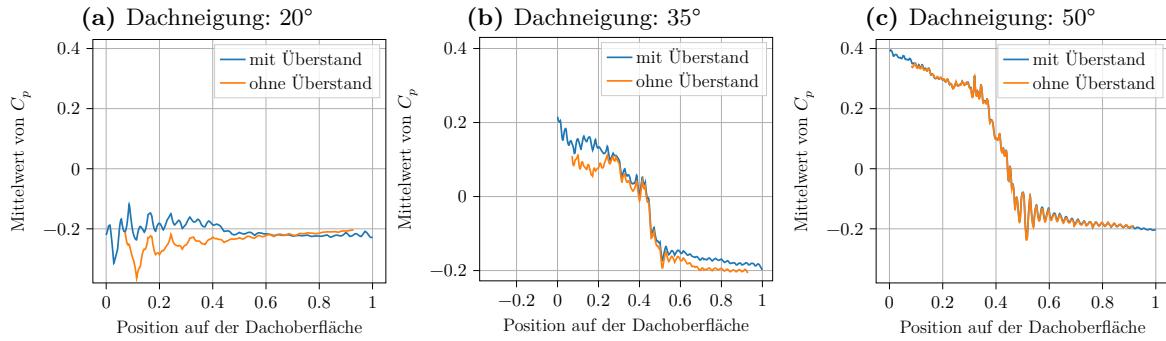


Abbildung 4.3.3-1: Vergleich des zeitlich gemittelten Druckkoeffizienten entlang der Mittellinie des Daches mit und ohne Dachüberständen bei den verschiedenen Dachneigungen, aufgetragen relativ zur Länge des Daches mit Überstand. Es ist zu erkennen, dass der Einfluss des Dachüberstandes mit wachsendem Neigungswinkel abnimmt.

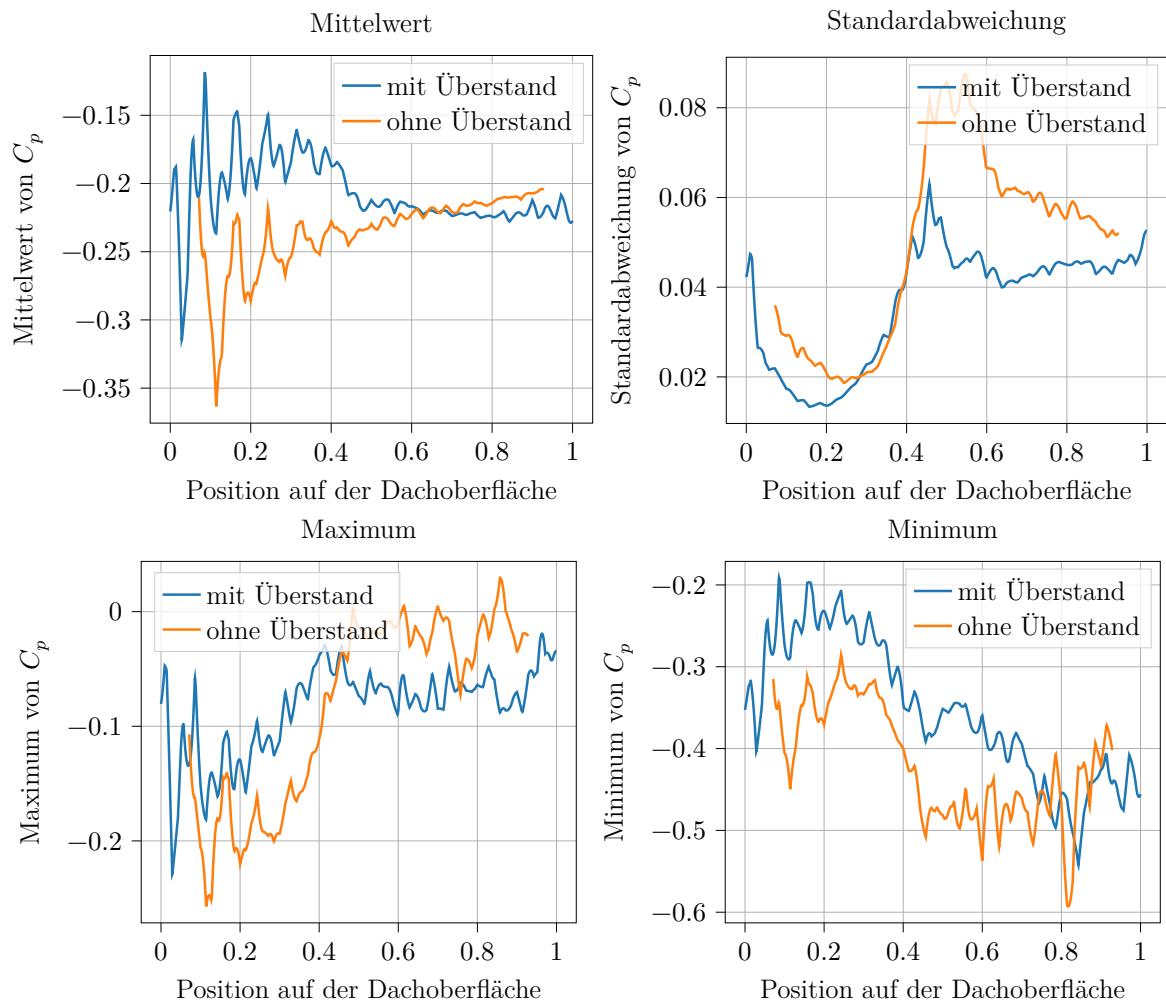


Abbildung 4.3.3-2: Vergleich des Mittelwerts, der Standardabweichung sowie der Extrema des Druckkoeffizienten beim Dach mit 20° Neigungswinkel mit und ohne Dachüberstand zur Verdeutlichung seines Einflusses auf das Druckverhalten. Es fällt auf, dass der Sog auf der Luvseite sowie die Standardabweichung und die Abweichung der Extremwerte vom Mittelwert auf der Leeseite beim Dach mit Dachüberständen geringer sind. Die Werte sind über die Länge der Mittellinie des Daches aufgetragen, wobei zu beachten ist, dass die absolute Länge des Daches ohne Überstand geringer ist.

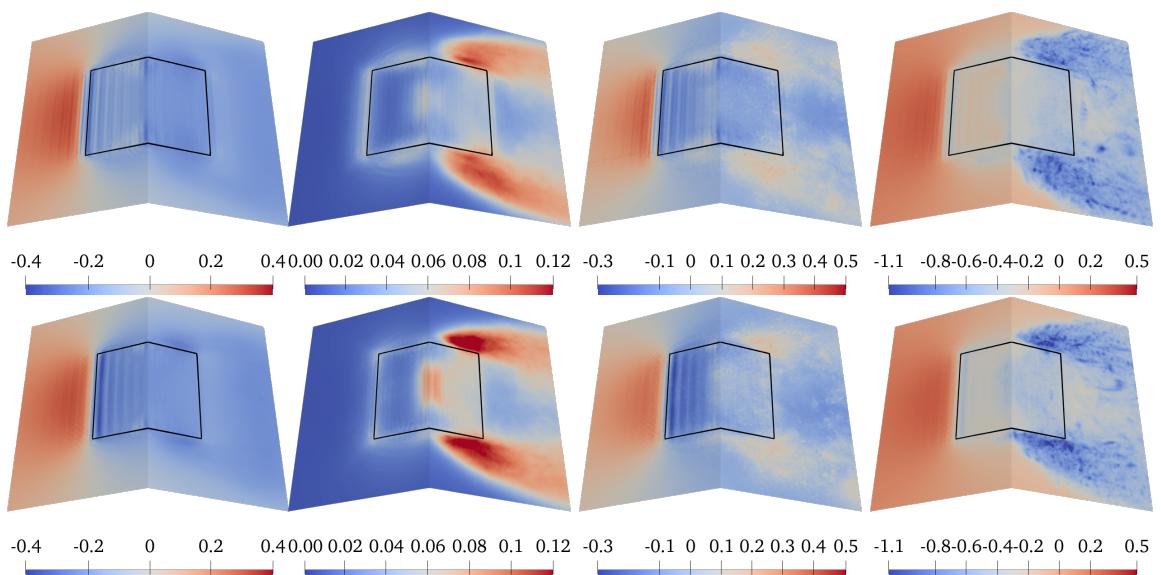


Abbildung 4.3.3-3: Vergleich des Druckkoeffizienten am Dach mit 20° Dachneigung mit (oben) und ohne (unten) Dachüberstand (links nach rechts: gemittelter Druckkoeffizient, Standardabweichung des Druckkoeffizienten, Maximum und Minimum des Druckkoeffizienten). Der schwarze Rand markiert die Umrisse des Dachs. Es fällt auf, dass im Verhalten des Drucks beim Dach ohne Dachüberstand die selben Inhomogenitäten vorliegen wie beim Verhalten mit Dachüberstand, jedoch mit größerer Amplitude.

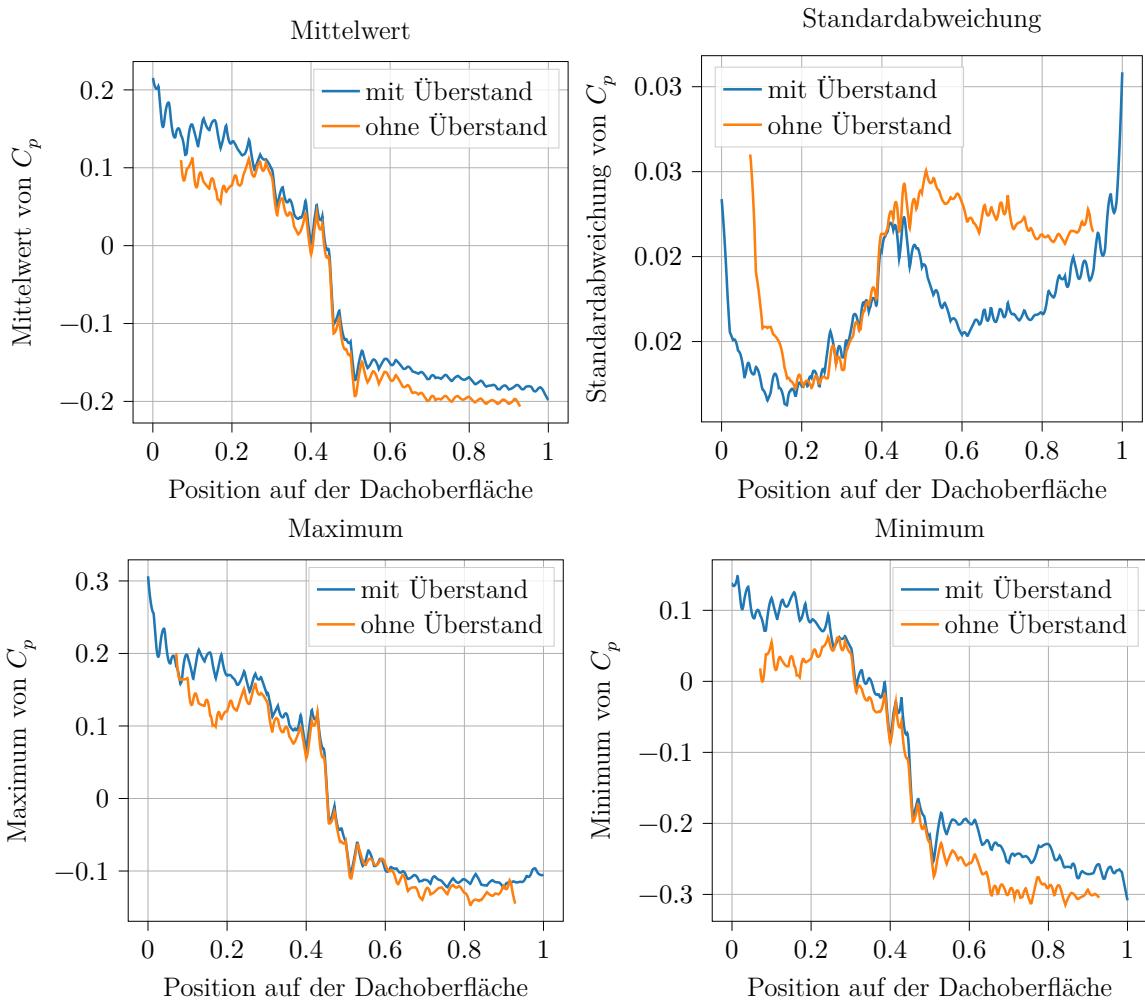


Abbildung 4.3.3-4: Vergleich des Mittelwerts, der Standardabweichung und der Extrema des Druckkoeffizienten beim Dach mit 35° Neigungswinkel mit und ohne Dachüberstand zur Verdeutlichung seines Einflusses auf das Druckverhalten. Die Werte sind über die Länge der Mittellinie des Daches aufgetragen, wobei zu beachten ist, dass die absolute Länge des Daches ohne Überstand geringer ist. Es fällt auf, dass der Druck beim Dach mit Dachüberstand auf einem Großteil der Länge höher ist und die Standardabweichung und die Abweichung der Extremwerte vom Mittelwert auf der Leeseite geringer sind.

4.4 Auswertung und Diskussion

Abschließend sollen die Ergebnisse zusammengefasst und, wo möglich, in den Kontext des Standes der Forschung eingeordnet werden. Darauf basierend werden dann entsprechende Schlüsse für die Wahl des Neigungswinkels von Satteldächern gezogen. Abschließend wird außerdem auf Schwächen der Untersuchung und mögliche Schritte um diese zu verbessern, eingegangen.

Auswertung der Untersuchungen zum Neigungswinkel

Bei der Untersuchung des Einflusses der Dachneigung auf den Druckkoeffizienten und die Strömungsverhältnisse werden verschiedene Aspekte beobachtet:

- Auf der Leeseite des Daches herrscht bei allen Neigungen Sog. Dieser ist beim Dach mit 20° Neigungswinkel am stärksten, bei den beiden anderen Dächern vergleichbar.
- Auf der Luvseite herrscht beim Dach mit 20° Neigungswinkel Sog, der nahe der Traufe besonders stark ist. Die Betrachtung der Strömung in diesem Bereich lässt vermuten, dass dies durch die Ablösung der Strömung an der Traufe und den folgenden bogenförmigen Verlauf oberhalb der Dachfläche verursacht wird.
- Bei 35° und 50° Dachneigung herrscht auf der Luvseite Überdruck, der mit dem Neigungswinkel zuzunehmen scheint. Dies scheint dadurch verursacht zu werden, dass die Strömung hier auf das Dach auftrifft und dann parallel dazu verläuft.
- Bei den Dächern mit 35° und 50° Neigungswinkel ist das Verhalten über den Großteil des Daches homogen und weist wenige Schwankungen auf. In der Nähe des Firsts treten dabei die meisten Schwankungen auf.
- Beim Dach mit 20° Neigungswinkel werden insbesondere auf der Leeseite relativ starke Schwankungen beobachtet, die sich von einer zentralen Region am First des Daches auszubreiten scheinen. Es weist dabei auch die größte Differenz der Extremwerte auf. Es wird vermutet, dass diese Schwankungen mit einem großen Nachlaufwirbel nahe der Leeseite des Daches zusammenhängen.
- Zwischen 20° und 35° Neigungswinkel tritt stromabwärts des Hauses eine größere Veränderung der Strömung auf als zwischen 35° und 50° . Neben einer Positionsveränderung eines großen Wirbels ist bei 35° Neigungswinkel ein Wirbel zu erkennen, der bei 20° nicht vorhanden ist. Vergleicht man die Strömungen bei 35° und 50° ist dagegen nur eine Verschiebung der Wirbel zu erkennen.

Insgesamt fällt hier auf, dass zwischen den Dächern mit 20° und 35° Neigungswinkel deutlich größere Unterschiede im Verhalten der Drücke und Strömungen zu beobachten sind als zwischen denen mit 35° und 50° . Dies lässt darauf schließen, dass in einem Bereich von Neigungswinkeln zwischen 20° und 35° grundlegende Veränderungen in der Umströmung des Daches auftreten, was auch Tominaga et al. im Fazit einer ähnlichen Untersuchung vermuten [3]. Dementsprechend wird vermutet, dass die Dächer mit Neigungswinkeln steiler bzw. flacher als dieser Bruch jeweils unter sich ähnliches Verhalten zeigen und die untersuchten Neigungen als Beispiele für die jeweilige Gruppe betrachtet werden können. Unter dieser Annahme zeigen die Ergebnisse, dass die Gruppe mit flacherem Neigungswinkel hinsichtlich der Windlast benachteiligt ist. Zum einen werden hier beide Seiten des Daches durch Sog belastet, zum anderen ist der Sog auf der Leeseite stärker und weist stärkere Schwankungen auf. Ozmen et al. gelangt in [12] zu einem ähnlichen Fazit bezüglich der Windlasten eines flachen Dachs. Folglich scheint die Gruppe mit steilerem Neigungswinkel hinsichtlich der Windlast bei der betrachteten Anströmung günstiger zu sein. Hier wird an der Leeseite ein geringerer

Sog festgestellt und es wird ein deutlich homogeneres Verhalten mit schwächeren Schwankungen beobachtet. Der Vergleich der Dächer mit 35° und 50° Neigungswinkel deutet dabei darauf hin, dass die Belastungen innerhalb dieser Gruppe mit dem Neigungswinkel zunehmen. Dementsprechend wird insgesamt vermutet, dass der flachste Neigungswinkel, bei welchem die Umströmung noch ähnlich der zweiten Gruppe ist, der optimale Dachneigungswinkel hinsichtlich der untersuchten Windlasten ist. Dieser wird dabei für die untersuchten Parameter zwischen 20° und 35° erwartet.

Auswertung der Untersuchungen zu Dachüberständen

Bei der Untersuchung der Auswirkungen der Dachüberstände auf die Drücke am Dach, werden folgende Beobachtungen gemacht:

- Am Dach mit Dachüberstand treten auf der Luvseite des Daches höhere Drücke auf, als am Dach ohne Dachüberstand.
- Das Dach mit Dachüberstand weist auf der Leeseite eine geringere Standardabweichung des Druckkoeffizienten auf und die entsprechenden Extremwerte liegen hier näher am Mittelwert.
- Die beobachteten Auswirkungen nehmen bei zunehmendem Dachneigungswinkel ab.

Insgesamt scheint ein Dachüberstand demnach unabhängig der Dachneigung eine stabilisierende Wirkung auf die Schwankungen an der Leeseite zu haben und den Druck auf der Luvseite zu erhöhen. Dementsprechend scheint ein Dachüberstand insbesondere für flachere Dächer, wie das untersuchte mit 20° Neigungswinkel, bei denen auf der Luvseite Sog herrscht, empfehlenswert. Auch steilere Dächer können jedoch potentiell von der beobachteten stabilisierenden Wirkung profitieren.

Beobachtungen hinsichtlich der Simulation selbst

Neben den Erkenntnissen hinsichtlich der Druckverhältnisse, zeigen die Ergebnisse jedoch auch weitere Schwachstellen der Simulation auf. Zum einen bestätigt die Untersuchung die in der Verifizierung festgestellten Probleme des stufigen Dachs. In der Nähe des Daches zeichnen sich die Stufen auch im Druck ab und sorgen so für lokale Druckunterschiede, die bei einem glatten Dach nicht auftreten würden. Obwohl in der Untersuchung Drücke in rund 50cm bzw. 3 LU Abstand vom Dach betrachtet werden, sind die Auswirkungen der Stufen noch zu erkennen, wie bspw. Abbildung 4.3.1-4 zeigt.

Zusätzlich sind an einigen Stellen in der Simulationsdomäne hochfrequente, wahrscheinlich unphysikalische Wellen zu erkennen. Sie sind bspw. in Abbildung 4.3.1-2 beim Haus mit 20° Dachneigungswinkel in dem Überdruck vor der Luvseite und beim Haus mit 50° Dachneigungswinkel auf der Leeseite des Firsts zu erkennen.

Insgesamt bestätigt der Vergleich mit der Literatur jedoch, dass die Ergebnisse zumindest qualitativ belastbar sind. Das zeigt auch, dass die LBM trotz der aufgetretenen Schwierigkeiten das Potential besitzt auch in diesem Bereich sinnvolle Ergebnisse zu liefern, die auch das Potential für neue Erkenntnisse besitzen, wie am Beispiel der aufgezeichneten lokalen Extrema zu erkennen, die bspw. mit RANS nicht ermittelt werden könnten.

Problematiken der Untersuchung und Lösungsansätze

Die Aussagekraft der vorangegangenen Ergebnisse wird dabei von verschiedenen Problematiken eingeschränkt, die auf Grund der begrenzten Zeit in Kauf genommen werden mussten, auf welche hier noch einmal eingegangen werden soll:

- Die Reynolds-Zahl, bei der die Simulation durchgeführt wurde, ist etwa drei bis fünf Größenordnungen von denen eines Sturms entfernt. Obwohl dennoch eine turbulente

Umströmung erzielt wird, stellt dies die Übertragbarkeit der Ergebnisse auf Wind, welcher tatsächlich Schädigungspotential besitzt, in Frage.

- Die Druckverhältnisse in nächster Nähe des Daches werden durch den stufigen Aufbau des Daches und eventuell auch durch die hier wirkende Fullway-Bounce-Back-Randbedingung, die eine schlechtere zeitliche Genauigkeit besitzt als andere Alternativen, gestört. Sinnvolle Werte können erst in einem Abstand von rund 3 LU bzw. 50 cm über dem Dach erfasst werden. Entwicklungen, die näher am Dach auftreten, können daher nicht erfasst oder berücksichtigt werden.
- Die Größe der untersuchten Simulationsdomäne ist im Vergleich zu anderen, die in ähnlichen Simulationen verwendet werden, relativ klein. Häufig werden hier Simulationsdomänen verwendet, die in jeder Richtung bis zu doppelt so weit ausgedehnt sind, um so eine Beeinflussung der Strömung durch die Randbedingungen zu verringern. Diese Problematik wird in der vorliegenden Arbeit möglicherweise dadurch abgeschwächt, dass für die Oberseite eine offene Randbedingung statt einer Wand verwendet wird, wie es in anderen Arbeiten zum Vergleich mit Windkanalergebnissen üblich ist (vgl. [7, 12]).
- Im Gegensatz zu natürlichen Luftströmungen in der betrachteten Höhe, weist die untersuchte Strömung vor dem Haus keine Turbulenzen auf [4].

Im Vergleich zu den Windkanal-Messergebnissen von [3] in Abschnitt 4.2, bei denen dennoch eine qualitative Übereinstimmung festgestellt wird, summieren sich diese Punkte zu der mittleren Abweichungen des Drucks an der Oberfläche des Hauses von bis zu 60 %, wobei bei den untersuchten Neigungswinkeln Abweichungen von 20% bei 50° Neigung bis zu 60 % bei 20° Neigung zu erwarten sind. Hiervon sind also in besonderem Maße die Ergebnisse hinsichtlich des flachen Daches betroffen.

Um hier eine Verbesserung dieser Problematiken zu erreichen und aussagekräftigere Ergebnisse zu erzielen, können verschiedene Ansätze verfolgt werden. Zum einen sollten der bei der abschließenden Simulation erstmalig aufgetretene Fehler der Parallelisierung behoben werden, um so eine größere Simulationsdomäne mit entsprechend besserer Genauigkeit oder höherer Reynolds-Zahl simulieren zu können. Eine Verbesserung der Geschwindigkeit der Parallelisierung bei Verteilung auf CPU-Knoten bietet ebenfalls das Potential sehr große Domänen realisieren zu können. Ähnliche Vorteile bietet hier auch die Möglichkeit einer Gitterverfeinerung, eine solche erlaubt es, kritische Bereiche selektiv höher aufzulösen.

Darüber hinaus stellen auch Randbedingungen mit höherer Fehlerordnung und Stabilität eine Möglichkeit zur Verbesserung der Ergebnisse dar. Eine mögliche Option einer besser geeigneten Randbedingung für Haus und Boden ist eine von Dorschner et al. publizierte Randbedingung, die auf der Grad-Approximation basiert [45]. Diese bietet neben höherer Genauigkeit auch die Möglichkeit die genaue Position des Randes zu interpolieren und kann so auch ein glattes Dach darstellen. Zudem ist sie bereits an Testfällen mit höheren Reynolds-Zahlen erprobt worden. Auf Grund ihres hohen Implementierungsaufwandes, konnte sie im Rahmen dieser Arbeit nicht umgesetzt werden, stellt aber eine interessante Option dar. Um die Abbildung des Drucks an der Dachoberfläche weiter zu verbessern bietet sich außerdem die Integration eines Wandmodells an. Diese bilden die Effekte der verschiedenen Grenzschichten an umströmten Oberflächen ab, die auf einem normalen Gitter kaum dargestellt werden können. Auch eine Einlassrandbedingung, die in der Lage ist, turbulente Strömung zu erzeugen, ist eine weitere Möglichkeit die Realitätsnähe der Simulation zu erhöhen.

5 Zusammenfassung und Ausblick

In dieser Arbeit wurde die turbulente Umströmung freistehender Einfamilienhäuser mit verschiedenen Dachformen und Dachneigungen mit der Lattice-Boltzmann-Methode untersucht. Dabei wurde besonderer Fokus auf die Drücke an der Dachfläche gelegt, da diese Potential besitzen, das Dach zu beschädigen. Zudem wurden Dachüberstände als Formvariante untersucht, da es hierzu bisher wenig Literatur gibt. In diesem Kapitel soll die Arbeit abschließend zusammengefasst werden und ein Ausblick auf ein mögliches weiteres Vorgehen gegeben werden.

Für die Untersuchung wurden dem verwendeten Löser Lettuce zunächst verschiedene Erweiterungen hinzugefügt, darunter eine Erweiterung auf parallelisiertes Rechnen, die auch validiert und hinsichtlich der Skalierung auf mehreren Prozessoren untersucht wurde. Dabei konnte insbesondere beim Rechnen auf Grafikkarten eine nahezu lineare Skalierung der Rechenleistung erreicht werden. Zusätzlich wurden einige benötigte Randbedingungen implementiert.

Anschließend wurde eine Simulationsumgebung für die Untersuchungen aufgebaut. Dabei wird der KBC-Kollisionsoperator verwendet. Die Anströmung erfolgt orthogonal zum Dachfirst mit einem üblichen Windgeschwindigkeitsprofil, wobei eine Reynolds-Zahl von 2×10^4 erreicht wird, was einer Strömungsgeschwindigkeit von rund 0.03 m/s entspricht.

Die Simulationsumgebung wurde anschließend anhand der Windkanal-Messergebnisse einer vorangegangenen Arbeit validiert. Dazu wurden in einer Simulation mit der selben Dachform vergleichbare Werte aufgezeichnet. Dabei zeigte sich, dass die Strömungsgeschwindigkeit mit einer mittleren relativen Abweichung von 9 % gut wiedergegeben wird. Im Gegensatz dazu liegt die mittlere relative Abweichung des Druckes je nach Dachneigungswinkel bei rund 20 bis 50%, wobei dieser Wert bei steileren Dächern niedriger ist. Hierbei wurde dennoch eine qualitative Übereinstimmung der Druckverläufe beobachtet.

Anschließend wurde die Umströmung dreier Satteldächer mit Dachneigungen von 20°, 35° und 50° simuliert und Druck- und Geschwindigkeitswerte aufgezeichnet. Die auftretenden Drücke oberhalb des Daches sowie die Strömungsgeschwindigkeit in einem Längsschnitt der Simulationsdomäne wurden untersucht. Um den Einfluss der Dachüberstände untersuchen zu können, wird die Simulation einmal mit und einmal ohne Dachüberstände durchgeführt.

Im Rahmen dieser Untersuchung wurden schließlich vor allem die Unterschiede des beobachteten Verhaltens der Umströmung des Hauses und der Drücke an der Dachfläche in Betracht gezogen, um die Auswirkung der verschiedenen Dachneigungen zu differenzieren. Dabei fällt auf, dass sich das beobachtete Verhalten bei den Dächern mit 35° und 50° Neigungswinkel ähnelt, während sich das beim Dach mit 20° Neigungswinkel beobachteten Verhalten von diesen stärker unterscheidet.

Bei den Dächern mit steilerem Neigungswinkel treten stromabwärts des Hauses zwei Wirbel auf, bei dem Haus mit flacherem Dach nur einer. Bei den Dächern mit steilerem Neigungswinkel trifft die Strömung auf der Luvseite auf und fließt daran entlang, beim Dach mit flacherem Winkel löst sich die Strömung an der Traufe der Luvseite ab und fließt in einem Bogen oberhalb der Dachfläche. Dementsprechend wirkt bei beiden steileren Dächern an der Luvseite Überdruck, während beim flacheren Dach Sog wirkt. Zusätzlich weist der Druck bei beiden steileren Dächern schwächere Schwankungen auf, die relativ gleichmäßig über die Dachfläche verteilt sind. Im Gegensatz dazu treten beim flacheren Dach vor allem auf der Leeseite Schwankungen auf, die deutlich stärker ausfallen.

Entsprechend dieser möglichen Gruppierung der Dächer, wird darauf geschlossen, dass

zwischen 20° und 35° Neigungswinkel grundlegende Veränderungen in der Umströmung des Daches auftreten und Dächer oberhalb und unterhalb jeweils ähnliches Verhalten zeigen. Dabei werden die steileren Dächer von der betrachteten Anströmung weniger stark belastet, da hier geringere statische und variable Drücke auftreten. Zugleich deuten die Ergebnisse darauf hin, dass die Belastung innerhalb dieser Gruppe mit dem Neigungswinkel zunimmt. Dementsprechend wird insgesamt vermutet, dass es einen optimalen Neigungswinkel gibt, bei welchem das Dach bei der betrachteten Anströmung die geringste Belastung erfährt, und dass dieser zwischen 20° und 35° liegt.

Die Ergebnisse der Untersuchung des Einflusses von Dachüberständen deuten darauf hin, dass Dachüberstände bei Satteldächern die Intensität und Häufigkeit der beobachteten Druckschwankungen an der Leeseite des Daches reduzieren können. Der Einfluss des Überstandes nimmt dabei jedoch mit der Dachneigung ab, sodass er bei 50° kaum noch feststellbar ist.

Während der Untersuchung zeigte sich gleichzeitig auch, dass es wenige Randbedingungen für die Lattice-Boltzmann-Methode gibt, die bei hohen Reynolds-Zahlen miteinander kompatibel sind und keine Instabilitäten verursachen. Insbesondere die Kanten, an denen Randbedingungen aneinanderstoßen, sind dabei sehr problematisch.

Die Aussagekraft der Ergebnisse wird dabei durch eine Anzahl an Problematiken eingeschränkt. So ist unsicher, inwieweit die Druckverhältnisse von der betrachteten Reynolds-Zahl auf Stürme mit deutlich höheren übertragen werden können. Zudem weist die Oberfläche des Daches auf Grund der gewählten Randbedingung Stufen auf, welche die Druckverhältnisse in der Nähe stören. Dadurch müssen alle Werte in einem Abstand von rund 50 cm zum Dach aufgenommen werden. Entwicklungen, die näher am Dach auftreten, können daher nicht erfasst oder berücksichtigt werden.

Die Ergebnisse der Untersuchung zeigen, dass es in diesem Bereich noch großes Forschungspotential gibt. Um die gewonnenen Erkenntnisse zu vertiefen sollten in folgenden Untersuchungen weitere Neigungswinkel zwischen 20° und 35° untersucht werden, um den vermuteten Zusammenhang zwischen Windlast und Neigungswinkel zu überprüfen. Zusätzlich sollten hier weitere Anströmungsszenarien betrachtet werden, um herauszufinden, welche Windrichtungen am kritischsten sind. Außerdem sollten die Auswirkungen weiterer Dachformen, wie bspw. Dachgauben, Schornsteine oder Solarpanelen auf die Belastung erkundet werden.

Bevor jedoch weitere Untersuchungen zu diesem Thema durchgeführt werden, sollten die Schwächen der Untersuchung angegangen werden, um aussagekräftigere Ergebnisse zu erzielen. Ein möglicher Schritt ist es dabei, den Fehler, welche die neuimplementierte Parallelisierung des Lösers in der Untersuchung unbenutzbar gemacht hat, zu beheben. Dann können größere Simulationsdomänen verwendet werden und so eine höhere Genauigkeit und/oder Reynolds-Zahl erreicht werden. Darüber hinaus stellen auch Randbedingungen mit höherer Fehlerordnung und Stabilität eine Möglichkeit zur Verbesserung der Ergebnisse dar. Eine mögliche Option einer besser geeigneten Randbedingung für Haus und Boden ist eine Randbedingung von Dorschner et al. [45] die sowohl interpolierte Verläufe der Randbedingung beherrscht als auch eine höhere Genauigkeit bietet. Darüber hinaus gibt es noch weitere Verbesserungsmöglichkeiten, die in Abschnitt 4.4 aufgeführt sind.

Literatur

Onlinequellen

- [2] Gesamtverband der Deutschen Versicherungswirtschaft e. V. *Serviceteil zum Naturgefahrenreport 2020*. 2020. URL: <https://www.gdv.de/resource/blob/63612/9bf0708f9a0017e98b878078894c7e52/naturgefahrenreport-2020---serviceteil-data.pdf> (besucht am 02.03.2021).
- [24] Debenben. *Schematische Darstellung eines Zeitschrittes in einem D2Q9-Modell*. URL: https://de.wikipedia.org/wiki/Lattice-Boltzmann-Methode%5C#/media/Datei:Lattice%5C_boltzmann%5C_3steps.svg (besucht am 21.02.2021).
- [31] Andreas Krämer u. a. *Lettuce Master Branch/ReadMe*. URL: <https://github.com/0lllom/lettuce> (besucht am 05.10.2020).
- [36] Adam Paszke u. a. *Pytorch Master Branch/ReadMe*. URL: <https://github.com/pytorch/pytorch> (besucht am 17.11.2020).

Gedruckte Quellen

- [1] Statistisches Bundesamt. *Bautätigkeit und Wohnungen Bestand an Wohnungen*. 2019.
- [3] Yoshihide Tominaga u. a. „Air flow around isolated gable-roof buildings with different roof pitches: Wind tunnel experiments and CFD simulations“. In: *Building and Environment* 84 (Jan. 2015), S. 204–213.
- [4] Peter Schmidt. „Windlasten“. In: *Lastannahmen - Einwirkungen auf Tragwerke: Grundlagen und Anwendung nach EC 1*. Wiesbaden: Springer Fachmedien Wiesbaden, 2019, S. 155–378.
- [5] Timm Krüger u. a. *The Lattice Boltzmann Method - Principles and Practice*. Berlin, Heidelberg: Springer, 2016. ISBN: 978-3-319-44649-3.
- [6] Stephan Lenz u. a. „Towards real-time simulation of turbulent air flow over a resolved urban canopy using the cumulant lattice Boltzmann method on a GPGPU“. In: *Journal of Wind Engineering and Industrial Aerodynamics* 189 (Juni 2019), S. 151–162.
- [7] Zhixiang Liu u. a. „An investigation on external airflow around low-rise building with various roof types: PIV measurements and LES simulations“. In: *Building and Environment* 169 (Feb. 2020).
- [8] Santiago Pindado, José Meseguer und Sebastián Franchini. „Influence of an upstream building on the wind-induced mean suction on the flat roof of a low-rise building“. In: *Journal of Wind Engineering and Industrial Aerodynamics* 99.8 (2011), S. 889–893.
- [9] Jagbir Singh und Amrit Roy. „Wind Pressure Coefficients on Pyramidal Roof of Square Plan Low Rise Double Storey Building“. In: (Apr. 2019).
- [10] Aly-Mousaad Aly und Joseph Bresowar. „Aerodynamic mitigation of wind-induced uplift forces on low-rise buildings: A comparative study“. In: *Journal of Building Engineering* 5 (2016), S. 267–276.

- [11] X.-H Zhou u. a. „Numerical simulation of 3D steady atmospheric flow around low-rise gable roof building“. In: *Gongcheng Lixue/Engineering Mechanics* 27 (März 2010), S. 19–29.
- [12] Y. Ozmen, E. Baydar und J.P.A.J. van Beeck. „Wind flow over the low-rise building models with gabled roofs having different pitch angles“. In: *Building and Environment* 95 (2016), S. 63–74.
- [13] Hui Hu u. a. „Characterization of the wind loads and flow fields around a gable-roof building model in tornado-like winds“. In: *Experiments in Fluids* 51 (2011). Discrete Simulation of Fluid Dynamics in Complex Systems, S. 835–851.
- [14] Takeo Kajishima und Kunihiko Taira. *Computational fluid dynamics: incompressible turbulent flows*. Springer International Publishing AG, 2017.
- [15] Cyril Crawford u. a. „Modeling of aerosol transmission of airborne pathogens in ICU rooms of COVID-19 patients with acute respiratory failure“. In: *medRxiv* (2020).
- [16] Mengtao Han, Ryozo Ooka und Hideki Kikumoto. „Validation of lattice Boltzmann method-based large-eddy simulation applied to wind flow around single 1:1:2 building model“. In: *Journal of Wind Engineering and Industrial Aerodynamics* 206 (2020), S. 104277.
- [17] Alain Schubiger, Sarah Barber und Henrik Nordborg. „Evaluation of the lattice Boltzmann method for wind modelling in complex terrain“. In: *Wind Energy Science* 5.4 (Nov. 2020), S. 1507–1519.
- [18] Fabian Bösch, Shyam S. Chikatamarla und Ilya V. Karlin. „Entropic multirelaxation lattice Boltzmann models for turbulent flows“. In: *Phys. Rev. E* 92 (4 Okt. 2015), S. 043309.
- [19] I. V. Karlin, F. Bösch und S. S. Chikatamarla. „Gibbs' principle for the lattice-kinetic theory of fluid dynamics“. In: *Phys. Rev. E* 90 (3 Sep. 2014), S. 031302.
- [20] Benedikt Dorschner, S. Chikatamarla und Iliya Karlin. „Transitional flows with the entropic lattice Boltzmann method“. In: *Journal of Fluid Mechanics* 824 (Aug. 2017), S. 388–412.
- [21] Joseph Spurk und Nuri Aksel. *Strömungslehre: Einführung in die Theorie der Strömungen*. 9., vollst. überarb. Auflage 2019. Berlin: Springer Berlin, 2019.
- [22] Sydney Chapman und Cowling T. G. *The Mathematical Theory of Non-uniform Gases*. 2. Ed. Cambridge University Press, 1952.
- [23] P. L. Bhatnagar, E. P. Gross und M. Krook. „A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems“. In: *Phys. Rev.* 94 (3 Mai 1954), S. 511–525.
- [25] Martin Geier u. a. „The cumulant lattice Boltzmann equation in three dimensions: Theory and validation“. In: *Computers & Mathematics with Applications* 70.4 (2015), S. 507–547.
- [26] Jonas Latt und Bastien Chopard. „Lattice Boltzmann method with regularized pre-collision distribution functions“. In: *Mathematics and Computers in Simulation* 72.2 (2006). Discrete Simulation of Fluid Dynamics in Complex Systems, S. 165–168.
- [27] Michael C. Sukop und Daniel T. Thorne. *Lattice Boltzmann Modeling*. Springer Berlin Heidelberg, 2006.
- [28] Anthony J. C. Ladd. „Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 1. Theoretical foundation“. In: *Journal of Fluid Mechanics* 271 (1994), S. 285–309.

- [29] Xiaoyi He u. a. „Analytic solutions of simple flows and analysis of nonslip boundary conditions for the lattice Boltzmann BGK model“. In: *Journal of Statistical Physics* 87.1-2 (Apr. 1997), S. 115–136.
- [30] Zheng Chu-Guang Shi Bao-Chang Guo Zhao-Li. „Non-equilibrium extrapolation method for velocity and pressure boundary conditions in the lattice Boltzmann method“. In: *Chinese Physics B* 11.4, 366 (2002), S. 366.
- [32] Dominik Wilde u. a. „Semi-Lagrangian lattice Boltzmann method for compressible flows“. In: *Phys. Rev. E* 101 (5 Mai 2020).
- [33] Alexander Hagg u. a. „Designing Air Flow with Surrogate-Assisted Phenotypic Niching“. In: *Parallel Problem Solving from Nature – PPSN XVI*. Cham: Springer International Publishing, 2020, S. 140–153.
- [34] Martin Leonhard Kliemank. „Masters Project 2: Examination of the Enstrophy of Two and Three-Dimensional Flow Around Objects with the Lattice Boltzmann Method“. Hochschule Bonn-Rhein-Sieg, Sep. 2020.
- [35] Adam Paszke u. a. „PyTorch: An Imperative Style, High-Performance Deep Learning Library“. In: *Advances in Neural Information Processing Systems 32*. Hrsg. von H. Wallach u. a. Curran Associates, Inc., 2019, S. 8024–8035.
- [37] Salvador Izquierdo, Paula Martínez-Lera und Norberto Fueyo. „Analysis of open boundary effects in unsteady lattice Boltzmann simulations“. In: *Computers & Mathematics with Applications* 58.5 (2009). Mesoscopic Methods in Engineering and Science, S. 914–921.
- [38] Kainan Hu u. a. „A comparative study of boundary conditions for lattice Boltzmann simulations of high Reynolds number flows“. In: *Computers & Fluids* 156 (2017). Ninth International Conference on Computational Fluid Dynamics (ICCFD9), S. 1–8.
- [39] Salvador Izquierdo. „Computational Gas Dynamics with the Lattice Boltzmann Method. Preconditioning and Boundary Conditions“. Diss. Okt. 2013.
- [40] Zhaoxia Yang. „Lattice Boltzmann outflow treatments: Convective conditions and others“. In: *Computers & Mathematics with Applications* 65.2 (2013). Special Issue on Mesoscopic Methods in Engineering and Science (ICMMES-2010, Edmonton, Canada), S. 160–171.
- [41] Santosh Ansumali und Iliya Karlin. „Kinetic boundary conditions in the lattice Boltzmann method“. In: *Physical review. E, Statistical, nonlinear, and soft matter physics* 66 (Sep. 2002).
- [42] Qisu Zou und Xiaoyi He. „On pressure and velocity boundary conditions for the lattice Boltzmann BGK model“. In: *Physics of Fluids* 9.6 (Juni 1997), S. 1591–1598.
- [43] Freddie D. Witherden. *On the Impact of Number Representation for High-Order ILES*. 2020.
- [44] *Nationaler Anhang – National festgelegte Parameter – Eurocode 1: Einwirkungen auf Tragwerke – Teil 1-4: Allgemeine Einwirkungen - Windlasten*. Norm. Dez. 2010.
- [45] B. Dorschner u. a. „Grad’s approximation for moving and stationary walls in entropic lattice Boltzmann simulations“. In: *Journal of Computational Physics* 295 (2015), S. 340–354.

Danksagung

Ein besonderer Dank gilt meinen Betreuer Dominik Wilde, der sich die Zeit genommen hat, mir neben seiner Arbeit an der Doktorarbeit wegweisend beiseite zu stehen. Ihm ist es zu verdanken, dass die Arbeit nicht dem Featurecreep erlegen ist! Auch für die vielen inspirierenden fachlichen und anderweitigen Gespräche danke ich sehr.

Auch den weiteren Mitgliedern des Lettuce-Teams danke ich für die fachlich wie persönlich bereichernde Zusammenarbeit.

Ich bedanke mich zudem auch bei Herrn Professor Reith für die Gelegenheit auf diesem Wege Einblick in die Wissenschaft zu erlangen und seine Mitwirkung an der Bewertung dieser Arbeit. Auch Herrn Professor Steinebach möchte ich für seine Mitwirkung an der Bewertung der Arbeit danken.

Die Simulationen wurden auf der Plattform für wissenschaftliches Rechnen an der Hochschule Bonn-Rhein-Sieg durchgeführt, die vom Bundesministerium für Bildung und Forschung sowie vom Ministerium für Kultur und Wissenschaft des Landes Nordrhein-Westfalen finanziert wird (Forschungsförderung 13FH156IN6).

Erklärung zur Master-Thesis

„Ich versichere hiermit, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.“

Mir ist bewusst, dass sich die Hochschule vorbehält, meine Arbeit auf plagierte Inhalte hin zu überprüfen und dass das Auffinden von plagierten Inhalten zur Nichtigkeit der Arbeit, zur Aberkennung des Abschlusses und zur Exmatrikulation führen können.“

Trondheim, den 8. April 2021

Ort, Datum



Unterschrift