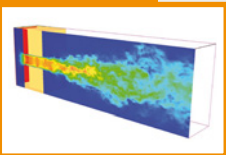
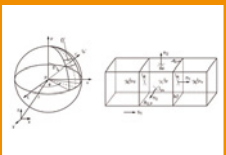


Nr. 5

Berichte des Instituts für Bauphysik

Hendrik C. Belaschk



Näherungsformeln für Spill Plumes in Atrien variabler Größe

CFD-Studien unter Verwendung
von GPU-Technologie

Näherungsformeln für Spill Plumes in Atrien variabler Größe

CFD-Studien unter Verwendung von
GPU-Technologie

von Hendrik C. Belaschk

FRAUNHOFER IRB VERLAG

Herausgeber

Univ.-Prof. Dr.-Ing. Nabil A. Fouad
Leibniz Universität Hannover
Institut für Bauphysik
Appelstraße 9A
30167 Hannover

Verfasser

Hendrik C. Belaschk

Alle Rechte vorbehalten

Dieses Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung in anderen als den gesetzlich zugelassenen Fällen bedarf deshalb der vorherigen schriftlichen Einwilligung des Herausgebers.

© 2012 by Univ.-Prof. Dr.-Ing. Nabil A. Fouad
Leibniz Universität Hannover
Institut für Bauphysik
ISBN 978-3-8167-8679-5

Fraunhofer IRB Verlag

Fraunhofer-Informationszentrum Raum und Bau IRB
Postfach 80 04 60, D-70504 Stuttgart
Telefon 0711 970-2500
Telefax 0711 970-2508
E-Mail irb@irb.fraunhofer.de
www.bauinformation.de

Berichte

des Instituts für Bauphysik der Leibniz Universität Hannover
Herausgeber:
Univ.-Prof. Dr.-Ing. Nabil A. Fouad;
Leibniz Universität Hannover – Institut für Bauphysik
Heft 5, April 2012

Fraunhofer IRB Verlag

Für Pamela

Kurzfassung

Horizontal verlaufende Rauchgasströme, die in ein Atrium eindringen und in diesem bis zur Decke aufsteigen, stellen ein bekanntes Phänomen bei Gebäudebränden dar (spill plumes). In der Planungsphase eines Gebäudes besteht deshalb häufig der Bedarf, die in diesem Fall erforderliche Entrauchung des Atriums auch ohne aufwendige CFD-Simulationen und Modellversuche beurteilen zu können. Dazu stehen dem Ingenieur eine Reihe von Methoden und Näherungsformeln zur Verfügung, mit denen die erforderliche Anzahl und Größe von natürlichen Rauchabzugsöffnungen bestimmt bzw. maschinelle Rauchabzüge dimensioniert werden können. Alle vereinfachten Methoden beschränken sich derzeit auf Atrien, die ein ausreichend großes Rauchgasreservoir aufweisen. Für kleine Atrien steht dem Ingenieur derzeit ausschließlich die BRE Spill-Plume-Methode zur Verfügung, die aufgrund ihrer Komplexität für den baupraktischen Einsatz nur bedingt geeignet ist. Im Rahmen dieser Arbeit werden Näherungsformeln entwickelt, die unabhängig von der Größe des Rauchgasreservoirs eine ausreichend genaue Abschätzung des Massenstroms im aufsteigenden Plume liefern.

Die neuen Näherungsformeln basieren im Wesentlichen auf einer detaillierten CFD-Studie. Infolge des hohen Bedarfs an Rechenleistung wird im Vorfeld untersucht, inwieweit die verwendete Simulationssoftware durch Einsatz von GPU-Prozessortechnologie beschleunigt werden kann. Dazu wird ein Großteil des Rechenkerns der Software FDS in das Programmiermodell C/CUDA portiert und parallelisiert. Darüber hinaus wird ein Multi-Wellenfrontalgorithmus für den Strahlungswärmetransport in gasförmigen Medien an die Erfordernisse der GPU-Technik angepasst und in die Software FDS integriert. Nach Abschluss der Parallelisierungsarbeit wird die modifizierte Software verifiziert und die erreichte Geschwindigkeitssteigerung ermittelt. Durch die vorgenommenen Änderungen an der Software wird eine Steigerung der Rechenleistung zwischen dem 10fachen (Algorithmen zum Strahlungswärmetransport) und dem 30fachen (Algorithmen zur Strömungsmechanik) erreicht.

Unter Verwendung der neuen Software wird anschließend eine CFD-Studie erstellt, deren FE-Modelle vorab über bereits bekannte Versuche validiert werden. Im Rahmen dieser Studie werden die Modelle im Hinblick auf die Atriengeometrie, die Öffnungsbreite des Brandraumes sowie die Wärmefreisetzungsrate variiert. Die entwickelten Näherungsformeln können unabhängig von der Atriengröße für einen Großteil der baupraktisch relevanten Anwendungsfälle verwendet werden und vereinfachen damit die bisherigen Rechenverfahren.

Schlagworte: Brandschutz, Ingenieurmethoden, Überlaufplumes

Abstract

A frequently encountered scenario in a building fire consists of a smoke layer underneath the ceiling within one fire compartment. If the compartment is open to an atrium, then the gases flow out immediately they reach the opening and rise as a thermal spill plume into a buoyant layer of gases formed beneath the atrium ceiling. During the planning phase of a building it is often required to estimate the amount of smoke which enters the smoke layer in order to calculate the required vent area for a smoke exhaust ventilation system. At the moment there are several simplified calculation methods available which are restricted only to large area reservoirs where the visible smoke layer depth is less than a certain height. In all other cases only the BRE spill plume method is applicable, a method which is rather complex and complicated to use. This work aims to provide a simplified formula to estimate the entrainment of air into a spill plume regardless of the size of the smoke gas reservoir.

The new formula is based essentially on an extensive CFD study. Due to the high computing performance required by the CFD program, research is carried out beforehand to determine the possibility of using GPU processor technology to speed up the computation. Therefore a large part of the source code of the Software used in the CFD study (Fire Dynamics Simulator) is ported to C/CUDA and parallelized as best as possible. Moreover, a multidimensional wavefront algorithm for radiation transport in gaseous media is integrated in the software. After the parallelization has been completed the new algorithms are verified and tested with regard to the performance increase. Based on sample calculations the new algorithms reduce the computing time by a factor of approximately 10 (radiation transport) resp. 30 (fluid mechanics) compared to the original software.

The CFD study carried out using the modified software is based on known test results performed on a physical scale model. After validation the CFD model is varied regarding the geometry of the atrium, the width of the opening in the fire compartment and the rate of heat release. Afterwards a new formula is developed to predict the mass flow rate of gases produced by a spill plume, which can be applied independently of the size of the smoke gas reservoir.

Keywords: fire protection engineering, spill plumes, GPU acceleration

Inhaltsverzeichnis

Kurzfassung.....	4
Abstract.....	5
Nomenklatur.....	10
Lateinische Formelzeichen.....	10
Griechische Formelzeichen.....	11
1 Zweck und Ziel der Arbeit.....	12
1.1 Einführung in die Problemstellung.....	12
1.2 Verwendung von Brand- und Rauchs simulationsrechnungen.....	15
1.3 Abgrenzung zu alternativen Fragestellungen.....	17
1.4 Zusammenfassung der Forschungsziele.....	19
1.5 Vorgehensweise.....	20
2 Stand des Wissens und der Technik.....	21
2.1 Einführung in die Modellbildung.....	21
2.2 Luftp einmischung in einen freien linienförmigen Brandherd.....	22
2.3 Die „BRE Spill-Plume-Methode“ für den Überlaufplume.....	23
2.4 Alternative Berechnungsansätze.....	25
2.4.1 Law [30] (1986).....	25
2.4.2 Thomas [12] (1987).....	26
2.4.3 Law [33] (1995).....	27
2.4.4 Poreh [11] (1998).....	28
2.4.5 Thomas et. al. [35] (1998).....	29
2.4.6 Harrison und Spearpoint [5] (2008).....	30
2.4.7 Ko et. al. [9] (2008).....	31
2.4.8 NFPA 92B [36] (2009).....	32
2.4.9 Harrison und Spearpoint [37] (2010).....	32
Teil A / Numerische Brand- und Rauchs imulation unter Verwendung von GPU-Prozessortechnologie.....	34
3 Einführung in die GPU-Technologie.....	35
3.1 Grundlagen der GPU-Technik.....	36
3.2 Hardware im GPGPU-Bereich.....	38
3.3 Überblick über das CUDA-Programmiermodell.....	40
4 Entwicklung einer GPU-Version für den Fire Dynamics Simulator (FDS).....	42
4.1 Die Software „Fire Dynamics Simulator (FDS)“.....	42
4.2 Paralleles Rechnen mit FDS (Version 5.4).....	43

4.3 Überblick über die neue GPU-Version.....	46
4.3.1 Beschreibung des Programm-Modells.....	47
4.3.2 Arithmetic Intensity Problem.....	49
4.3.3 Rechengenauigkeit	50
4.3.4 Ein- und Mehrnetzrechnungen.....	51
4.4 Programmtechnische Umsetzung.....	52
4.4.1 Parallelisierungsschema	52
4.4.2 Neuordnung von Speicherbereichen.....	53
4.4.3 Parallele Reduktion (parallel reduction).....	55
4.4.4 Dokumentation neuer Eingabeparameter.....	56
4.4.5 Versionsänderungen.....	56
4.5 Parallelisierung der Algorithmen zum Strahlungswärmetransport.....	57
4.5.1 Grundlagen des Wärmetransports über Strahlung.....	57
4.5.2 Numerische Lösung der Transportgleichung mit der Finite-Volumen- Methode (FV-Methode).....	59
4.5.3 Modellierung des Strahlungswärmetransports in FDS.....	61
4.5.4 Möglichkeiten zur Parallelisierung der Finite-Volumen-Methode.....	64
4.5.4.1 Angular Decomposition Parallelization (ADP-Verfahren).....	64
4.5.4.2 Spatial Domain Decomposition Parallelization (DDP-Verfahren).....	65
4.5.4.3 Der KBA-Algorithmus für orthogonale Netze.....	66
4.5.5 Anpassung des KBA-Verfahrens an GPU-Systeme.....	69
4.5.6 Implementierung des KBA-Verfahrens in FDS.....	72
4.5.7 Zusammenfassung.....	74
4.6 Entwicklung eines Profilers.....	74
5 Verifikation und Performance.....	75
5.1 Verifikation im Diagnosemodus (diagnostic mode).....	76
5.2 Beispiel 1: Isothermer Freistrahler.....	78
5.2.1 Theoretische und experimentelle Grundlagen.....	78
5.2.2 Zusammenfassung bisheriger Untersuchungen.....	79
5.2.3 Verifizierung der GPU-Version.....	80
5.2.4 Rechenleistung / Performance.....	83
5.2.5 Zusammenfassung.....	84
5.3 Beispiel 2: Strahlung in einer Box.....	84
5.3.1 Theoretische Grundlagen.....	85
5.3.2 Verifizierung der GPU-Version.....	85
5.3.3 Rechenleistung / Performance.....	87

Teil B / Simulation von Spill Plumes in Atrien.....88**6 Numerische Simulation von Spill Plumes in Atrien.....89**

6.1 Zusammenfassung der Versuche nach Harrison [10] und [4].....	89
6.1.1 Versuchsaufbau und Versuchsdurchführung.....	89
6.1.2 Ergebnisse aus den Versuchsreihen.....	92
6.1.3 Diskussion der Ergebnisse.....	92
6.2 Beschreibung des Computermodells.....	93
6.2.1 Anordnung von virtuellen Messfühlern.....	95
6.2.2 Vorlaufzeiten für die stationäre Problemstellung.....	96
6.3 Modellvalidierung: Strömung am Überlauftrand.....	98
6.3.1 Zusammenfassung der Berechnungsergebnisse.....	99
6.3.2 Diskussion der Ergebnisse	100
6.4 Modellvalidierung: Massenstrom im aufsteigenden Plume (ohne Atrium).....	100
6.4.1 Zusammenfassung der Berechnungsergebnisse.....	101
6.4.2 Diskussion der Ergebnisse.....	102
6.5 Modellvalidierung: Massenstrom im aufsteigenden Plume (mit Atrium).....	104
6.5.1 Zusammenfassung der Berechnungsergebnisse.....	105
6.5.2 Diskussion der Ergebnisse.....	105
6.6 Parameterstudie: Massenstrom im aufsteigenden Plume (ohne Atrium).....	107
6.6.1 Zusammenfassung der Berechnungsergebnisse.....	108
6.6.2 Entwicklung einer Näherungsformel für den freien Plume.....	108
6.7 Parameterstudie: Massenstrom im aufsteigenden Plume (mit Atrium).....	110
6.7.1 Zusammenfassung der Berechnungsergebnisse.....	111
6.7.2 Entwicklung einer Näherungsformel für die Masseneinmischung.....	113
6.7.3 Bewertung der bestehenden Näherungsformeln.....	115

7 Zusammenfassung und Ausblick.....116

7.1 Einsatz der GPU-Prozessortechnologie in der Brand- und Rauchs simulation.....	116
7.2 Entwicklung von Näherungsformeln für den Massenstrom im Überlaufplume innerhalb von Atrien.....	118

Anlagenteil A / Grundlagen paralleler Rechentechnik.....127

A.1 Flynnsche Klassifizierung für Parallelrechner.....	128
A.2 Parallelrechner mit gemeinsamen oder verteiltem Speicher.....	129
A.3 ECC-Fehlerschutz für den Hauptspeicher.....	131
A.4 Explizite und implizite Parallelisierung.....	132
A.5 Parallele Effizienz und Leistungssteigerung (SpeedUp).....	133
A.6 Die Programmiermodelle MPI und OpenMP.....	134

Anlagenteil B / Beispielrechnung zur BRE Spill-Plume-Methode.....	137
Anlagenteil C / Auszug aus dem Quelltext zum Multi-Wellenfrontalgorithmus.....	141
Anlagenteil D / Beispiele zur Verifizierung der GPU-Version.....	145
Anlagenteil E / Parameterstudie zu Spill Plumes in Atrien.....	152

Nomenklatur

Lateinische Formelzeichen

A_a	Grundfläche des Atriums [m^2]
A_p	Oberfläche eines Kontrollvolumens [m^2]
b	Balkenbreite [m]
C_m	Einmischungskonstante nach Gl. (3) [–]
c	Spezifische Wärmekapazität [$kJ/(kg \cdot K)$]
c_p	Spezifische Wärmekapazität der Umgebungsluft, $c_p \approx 1,01 \text{ kJ}/(kg \cdot K)$
D	Durchmesser an der Austrittsöffnung [m]
db, D_b	Höhe der rauchgasführenden Schicht im Brandraum, vgl. Abb. 6, S. 21 [m]
g	Erdbeschleunigung, $g \approx 9,81 \text{ m/s}^2$
H	Höhe des Brandraumes [m]
H_a	Höhe des Atriums [m]
H_w	Höhe der Brandraumöffnung (Höhe bis zum Unterzug) [m]
I_b	Strahlungsintensität des schwarzen Körpers [W]
I_p	Strahlungsintensität eines Kontrollvolumens [W]
I_λ	Spektrale Strahlungsintensität [W]
m	Mischzahl, vgl. Abs. 5.2.1 [–]
M, M'	Massenstrom im Plume über einem freien Brandherd [kg/s], [$kg/(s \cdot m)$]
$M_{1,2,3,4}$	Massenströme im Brandraum [kg/s]
M_5	Massenstrom im Rotationsbereich [kg/s]
M_a, M_a'	Massenstrom im Plume innerhalb des Atriums [kg/s], [$kg/(s \cdot m)$]
M_b, M_b'	Massenstrom am Überlauftrand [kg/s], [$kg/(s \cdot m)$]
M_{ends}	Massenstrom im Bereich der freien Enden des Plumes [kg/s]
M_p, M_p'	Massenstrom im Plume über einer freien Brandquelle [kg/s], [$kg/(s \cdot m)$]
Q, Q'	Konvektiver Wärmestrom [W], [W/m]
Q_{tot}	Wärmefreisetzungsrate des Brandherdes [W]
Re	Reynolds-Zahl [–]
t_v, T_v	Vorlaufzeiten zum Erreichen des stationären Zustandes [s]
T_o	Temperatur der Umgebungsluft, $T_o \approx 293 \text{ K}$
u	Geschwindigkeit in der Gasströmung [m/s]
V_p	Kontrollvolumen [m^3]
W	Breite der Öffnung im Brandraum [m]
W_e	Effektive Breite des Plumes am Überlauftrand (unchannelled flow) [m]
x_o	Kernlänge des Freistrahls [m]
z	Höhe des aufsteigenden Plumes im Atrium, vgl. Abb. 6, S. 21 [m]
z_{int}	Grenzschichthöhe zwischen Rauchgas und Umgebungsluft im Atrium [m]
z_{edy}	Wirbelgrenze nach Abb. 50, S. 104 [m]

z_o	Abstand zum Ursprung der virtuellen Feuerquelle, vgl. Abb. 6, S. 21 [m]
z_k	Höhe des aufsteigenden Plumes über einem freien Brandherd [m]

Griechische Formelzeichen

α	Einmischungskonstante nach Lee und Emmons, vgl. Abs. 2.2 [–]
α, β, γ	Experimentelle Konstanten nach Thomas, vgl. Abs. 2.4.5 [–]
θ_a	Temperatur der Rauchgase im Atrium [°C]
θ_b	Temperatur der Rauchgase am Überlaufrand [°C]
κ	Absorptionsgrad [–]
λ	Einmischungskonstante nach Lee und Emmons, vgl. Abs. 2.2 [–]
λ	Wärmeleitfähigkeit [W/(m·K)]
Ω^l	Kontrollwinkel [–]
ρ	Dichte des Rauchgases [kg/m³]
ρ_o	Dichte der Umgebungsluft, $\rho_o \approx 1,22 \text{ kg/m}^3$
σ	Stefan-Boltzmann-Konstante, $\sigma \approx 5,67 \cdot 10^{-8} \text{ W/(m}^2\text{K}^4\text{)}$
σ_s	Streuungskoeffizient [–]
Φ	Strahlungsaustauschfaktor (configuration factor) [–]
χ_r	Anteil der Strahlungsleistung an der Gesamtenergiefreisetzung [–]

1 Zweck und Ziel der Arbeit

1.1 Einführung in die Problemstellung

Die Häufigkeit von Brandkatastrophen zeigt die Notwendigkeit auf, bereits bei der Errichtung eines Gebäudes Maßnahmen vorzusehen, die die Gefahr der Brandentstehung reduzieren bzw. die die Folgeschäden im Brandfall begrenzen. Das allgemeine Ziel derartiger Maßnahmen ist neben dem Schutz von Eigentum (Sachwertschutz) vor allem der Schutz von Leben und Gesundheit (Personenschutz). Aus der Komplexität und Vielfalt möglicher Brandgeschehen resultieren dabei eine ganze Reihe von Anforderungen, die an die Bausubstanz und technische Ausrüstung eines Bauwerks zu stellen sind. So muss die Tragfähigkeit der Gebäudestruktur im Brandfall über einen definierten Zeitraum erhalten bleiben, um Evakuierungsmaßnahmen durchführen und Maßnahmen zur Brandbekämpfung einleiten zu können. Darüber hinaus muss sichergestellt sein, dass die Ausbreitung des Brandes im Gebäudeinneren begrenzt wird, Fluchtwege rauchgasfrei bleiben und der Brand nicht auf die Nachbarbebauung übergreifen kann. Um die Risiken im Brandfall zu reduzieren, besteht in der Ingenieurpraxis der Bedarf, bereits in der Planungsphase mögliche Brandverläufe vorherzusagen und deren Auswirkung auf Personen und Sachgüter zu beurteilen.

Diese Dissertation beschäftigt sich mit einem in der Praxis häufig auftretenden Spezialfall, dem Rauchgasmanagement in Atrien. Bei einem „Atrium“ im Sinne dieser Arbeit handelt es sich um einen großen Raum im Gebäudeinneren, der geschossübergreifend angeordnet und mit den angrenzenden Räumlichkeiten durch Umgänge oder Fenster verbunden ist. In der Praxis sind dabei sowohl offene als auch geschlossene Atrien vorzufinden (s. Abb. 1). Die Tatsache, dass derartige Räumlichkeiten als architektonisches Element in der Gegenwart häufig eingesetzt werden (z. B. in Form von Einkaufszentren, Empfangshallen, Flughäfen, Sportstätten etc.), darf nicht darüber hinweg täuschen, dass ein Atrium den fundamentalen Prinzipien des baulichen Brandschutzes widerspricht. So läuft die Anordnung eines geschossübergreifenden Raumes der angestrebten Trennung horizontaler Ebenen durch feuerbeständige Geschossdecken zuwider. Auch birgt ein Atrium durch seine Anbindung an angrenzende Räumlichkeiten die Gefahr, dass sich Rauchgase im Brandfall unkontrolliert im gesamten Gebäude ausbreiten. Eine detaillierte Kenntnis über die Brand- und Rauchentwicklung in Atrien ist für den Ingenieur daher von besonderer Bedeutung.

Da in den Verbrennungsprodukten chemische Verbindungen enthalten sind, die auf den menschlichen Organismus stark toxisch wirken (Kohlenmonoxid, Chlorwasserstoff, Cyanid etc.), kann die Inhalation von Rauchgas innerhalb weniger Minuten zum Tode führen. Tatsächlich ist die Rauchgasvergiftung mit über 70 % die häufigste Verletzungs- bzw. Todesursache bei Bränden [1], so dass ein Kontakt zwischen Rauchgas und Personen im

Brandfall zwingend vermieden werden muss. Mit dem Rauchgasmanagement in Atrien soll daher vor allem eine „kontrollierte Entrauchung“ des Atriums erreicht werden, bei der Personenschäden auszuschließen sind.

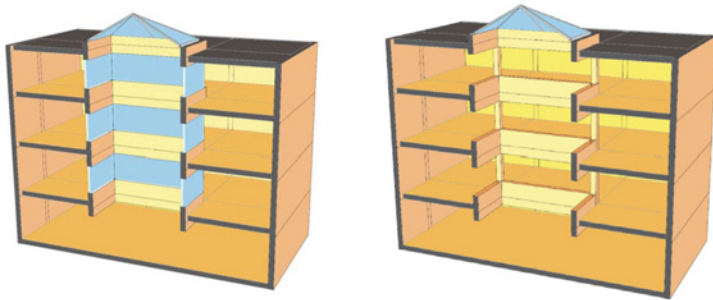


Abb. 1: Ausbildung eines Atriums in geschlossener (links) und offener Form (rechts), Auszug aus [2]

Ganz allgemein kann eine derartige „kontrollierte Entrauchung“ von Atrien auf unterschiedliche Art und Weise erfolgen. Während es bei größeren Atrien häufig ausreichend ist, die Anzahl der Entrauchungsöffnungen klein zu halten (Evakuierung des Gebäudes in der Zeit bis zum Befüllen des Atriums), ist vor allem bei kleineren Atrien die natürliche Entrauchung infolge thermischen Auftriebs nicht immer ausreichend. In diesem Fall können maschinelle Rauch- und Wärmeabzüge zum Einsatz kommen, die das Rauchgas im oberen Bereich des Atriums absaugen bzw. in die freie Umgebung ableiten. Soll das Befüllen des Atriums mit Rauchgas planmäßig zulässig sein, so ist zusätzlich sicherzustellen, dass keine Rauchgase in angrenzende Räume entweichen können (z. B. über ein geschlossenes Atrium, vgl. Abb. 1, links). Ebenso muss die Standsicherheit von trennenden Bauteilen auch bei hohen Temperaturen gewährleistet sein (z. B. Einsatz von Brandschutzverglasung). Letztendlich kann es vorkommen, dass eine ausreichende Entrauchung des Atriums prinzipiell nicht zu erreichen ist, so dass das Einstromen von Rauchgas in das Atrium im Vorfeld unterbunden werden muss (z. B. durch Absaugen der Rauchgase vor dem Eintritt in das Atrium).

Diese Arbeit beschäftigt sich ausschließlich mit der Entrauchung von Atrien über Rauch- und Wärmeabzüge, die im Bereich der Atriumdecke angeordnet sind und bei denen sich das Rauchgas aus angrenzenden Räumen in das Atrium „ergießt“. Dabei strömt Rauchgas aus dem Brandraum über einen Überlaufrand (spill edge) in das Atrium ein und steigt dort infolge thermischen Auftriebs auf (s. Abb. 2). Aufgrund der meist geringeren Brandlasten innerhalb des Atriums ist dieses Szenario gegenüber einem direkten Brand im Atrium für die Dimensionierung der Rauch- und Wärmeabzugsöffnungen häufig maßgebend. Durch den hohen Turbulenzgrad der Brandgase im Atrium mischt sich kontinuier-

lich Luft in das aufsteigende Rauchgas ein, was zur Abnahme der Rauchgastemperatur und Zunahme der Rauchgasmenge über die Höhe führt. Da die ausströmende Rauchgasmenge durch die Größe der Rauch- und Wärmeabzugsöffnungen begrenzt ist, bildet sich im oberen Teil des Atriums ein Rauchgasreservoir aus, das durch eine Grenzschicht von der rauchgasfreien Zone abgetrennt ist. Die Höhe dieser Grenzschicht ergibt sich aus dem Gleichgewicht zwischen aus- und nachströmendem Rauchgas und ist von der Wärmefreisetzungsrate des Brandherdes, den geometrischen Randbedingungen (Atrium, Brandraum etc.) sowie der Anzahl und Größe der Rauch- und Wärmeabzugsöffnungen abhängig.

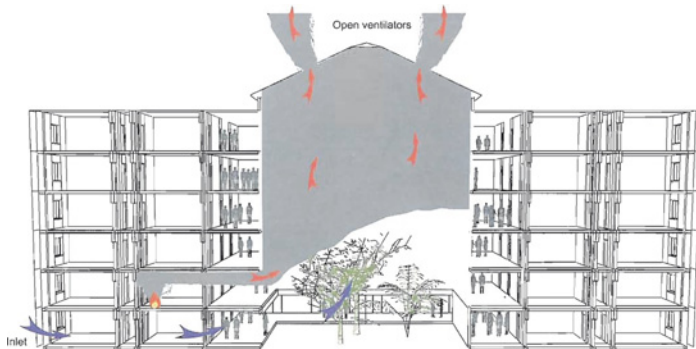


Abb. 2: Prinzipskizze zur Entrauchung eines Atrium über Rauch- und Wärmeabzüge, Auszug aus [2]

Bei dem oben beschriebenen Szenario besteht das Ziel vor allem darin, die Rauch- und Wärmeabzugsöffnungen derart auszubilden, dass unterhalb des Rauchgasreservoirs ein ausreichend großer rauchgasfreier Bereich vorhanden ist. Diese Forderung kann als erfüllt angesehen werden, wenn über den Evakuierungszeitraum hinweg das Eindringen von Rauchgas in die angrenzenden Räumlichkeiten ausgeschlossen werden kann. Darüber hinaus muss – sofern das Atrium zur Entfluchtung genutzt wird – zusätzlich eine ausreichend hohe rauchgasfreie Schicht im Bereich der Fluchtwege vorhanden sein. Ausgehend vom Fußboden des zu entfluchtenden Geschosses werden dabei in der Literatur bzw. den einschlägigen Normen rauchgasfreie Schichthöhen zwischen von 2,5 bis 3 m verlangt [3].

Um die maximal zulässige Höhe des Rauchgasreservoirs einzuhalten, muss durch den Ingenieur der zu erwartende Rauchgas-Massenstrom im aufsteigenden Plume quantifiziert werden. Darüber hinaus sind die Rauchgastemperaturen abzuschätzen, da sich bei zu geringer Rauchgastemperatur keine klare Trennung zwischen verrauchtem und rauchgasfreiem Bereich einstellt. In diesem Fall sind keine gesicherten Aussagen über die Höhe der rauchgasfreien Zone möglich und es besteht die Gefahr der vollständigen Ver-

rauchung des Atriums. Sowohl die Rauchgastemperatur als auch der Massenstrom können nach dem Stand der Technik über Modellversuche oder Simulationsrechnungen im Computer (CFD-Studien) für den Einzelfall recht genau ermittelt werden. Vor allem im Entwurfsstadium eines Gebäudes besteht jedoch der Bedarf, derartige Abschätzungen auch ohne größeren Aufwand treffen zu können (z.B. über eine Näherungsformel bzw. ein Handrechenverfahren). Mit dieser Arbeit werden die in diesem Zusammenhang bereits bestehenden vereinfachten Verfahren überprüft (vgl. Abs.2) und durch ein robusteres Verfahren ergänzt.

1.2 Verwendung von Brand- und Rauchs simulationsrechnungen

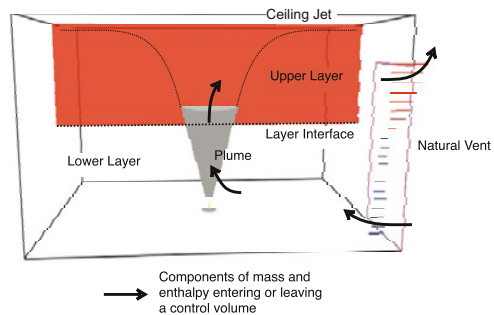
Wie im Abschnitt 1.1 erläutert, besteht das Ziel dieser Arbeit im Aufstellen einer vereinfachten Methodik zur Ermittlung des Massenstroms im Überlaufplume innerhalb von Atrien. Hierzu sind bereits früher experimentelle Versuche durchgeführt worden, auf die in dieser Arbeit zurückgegriffen wird (z. B. [4, 5]). Da alle Experimente spezifischen Rahmenbedingungen unterliegen (Öffnungsbreite im Brandraum, Atriengeometrie, Beschränkung der Wärmefreisetzungsrate etc.), weisen auch die daraus resultierenden vereinfachten Formeln entsprechende Beschränkungen auf (vgl. Abs. 2.4). In dieser Arbeit sollen die bekannten Näherungsformeln erweitert und generalisiert werden. Hierfür werden die verfügbaren Versuchsergebnisse in Computermodelle übertragen, deren Randbedingungen anschließend in der gewünschten Form abgeändert bzw. variiert werden.

Das Aufstellen einer derartigen Parameterstudie setzt ausreichend detaillierte Modelle des Brandgeschehens voraus, wofür – infolge der inhärent komplexen Struktur des Feuers selbst – ein ganz massiver Bedarf an Rechenleistung erforderlich wird. Neben der reinen Fluiddynamik (viskose, turbulente Mehrphasenströmung) müssen in der Simulation zusätzlich chemische Prozesse (Verbrennungsreaktionen in der Gasphase, Ausgasung von Feststoffen etc.) und Aspekte zum Wärme- und Stofftransport (Strahlungsübertragung in gasförmigen Medien, Wärmeleitung in Feststoffen, Stofftransport über Diffusion etc.) berücksichtigt werden. Erschwert wird die Berechnung der physikalischen und chemischen Vorgänge zusätzlich durch den hohen Grad ihrer Abhängigkeit untereinander sowie die unterschiedlichen Raum- und Zeitskalen, auf denen die Prozesse ablaufen.

Bei dem in der Ingenieurpraxis am häufigsten eingesetzten Rechenmodell handelt es sich um das „Zonenmodell“. In diesem Modell wird jeder am Brandgeschehen beteiligte Raum in zwei homogene Zonen unterteilt: eine Heißgasschicht (hot upper layer) und eine Kaltluftschicht (cooler lower layer). Im Brandraum (s. Abb.3) werden gegebenenfalls zusätzliche Zonen berücksichtigt (fire plume, ceiling jet). Während des Rechenlaufes werden in Abhängigkeit der physikalischen Randbedingungen (Gasströmungen durch Fenster und Türen, Wärmeaustausch über Strahlung, Konvektion und Verbrennungsprozesse) Masse- und Energiebilanzen zwischen den einzelnen Zonen aufgestellt. Da das

Zonenmodell nur für Brandversuche in einfachen Räumen validiert ist (compartment fires), ist ein Einsatz bei den hier betrachteten Atrien nicht möglich. Obwohl die Verwendung des Zonenmodells Vorteile hätte (einfache Handhabung, geringer Bedarf an Recheneleistung etc.), wird in dieser Arbeit daher auf ein aufwendigeres Modell, das „Feldmodell“ zurückgegriffen.

Abb. 3: Schemaskizze zum Zonenmodell: Aufteilung eines Brandraumes in unterschiedliche Zonen, Auszug aus [6]



Im Gegensatz zum Zonenmodell berücksichtigen Feldmodelle (CFD-Modelle) den kompletten Satz gekoppelter, nicht linearer partieller Differenzialgleichungen (Erhaltungsgleichungen für Energie, Masse und Impuls, Transportgleichungen für die Strahlung in gasförmigen Medien bzw. den Wärmetransport in Feststoffen, Gleichungen zur Beschreibung der chemischen Verbrennung usw.) mit einer wesentlich größeren Raum- und Ortsauflösung. Dadurch unterliegen diese Modelle nicht den Einschränkungen der Zonenmodelle und können auch komplexe Strömungsvorgänge in beliebigen Raumgeometrien darstellen. Die für eine numerische Lösung dieser Gleichungen in ihrer Gesamtheit erforderliche Rechenleistung übersteigt jedoch die Möglichkeiten heutiger Computertechnik um viele Zehnerpotenzen, so dass im Vorfeld Vereinfachungen vorgenommen werden müssen, die den Rechenbedarf reduzieren.

Das im Rahmen dieser Arbeit verwendete, vereinfachte Feldmodell ist in der Software „Fire Dynamics Simulator“ (FDS) [7] implementiert. Eine Kurzbeschreibung des Modells ist im Abschnitt 4.1 enthalten. Werden Brandsimulationsrechnungen für Atrien bzw. größere Räume durchgeführt, so stellt sich schnell heraus, dass der Rechenaufwand trotz aller in FDS vorgenommenen Vereinfachungen im Grenzbereich der verfügbaren Rechenleistung heutiger Computertechnik liegt. Um die geplante Parameterstudie in realistischer Rechenzeit bewältigen zu können, sollen daher in dieser Arbeit die bestehenden Algorithmen der Software FDS im Vorfeld umgeschrieben und an eine neue Rechentechnik, die GPU-Prozessortechnik (s. Erläuterungen im Abs. 3.1) angepasst werden. Wie die Anwendung in anderen wissenschaftlichen Gebieten zeigt, kann durch den Einsatz von

GPU-Technik teilweise eine ganz erhebliche Beschleunigung von Simulationsrechnungen erreicht werden, was sich positiv auf den erreichbaren Detailgrad der Simulation und folglich auf die Verlässlichkeit der Ergebnisse auswirkt.

Aufgrund des erheblichen Quellcodeumfanges (ca. 50.000 Zeilen) ist es nicht möglich, die Software FDS vollständig an die GPU-Technik anzupassen. Im Rahmen dieser Dissertation werden daher lediglich die für die spezifische Problemstellung erforderlichen Algorithmen neu formuliert. Zum einen wird dadurch die Rechenzeit für die zu erstellende Parameterstudie massiv reduziert und zum anderen wird die Voraussetzung für eine vollständige Anpassung der Software durch das Entwicklerteam von FDS geschaffen.

1.3 Abgrenzung zu alternativen Fragestellungen

Die Entwicklung einer Näherungsformel bzw. eines vereinfachten Verfahrens für die in Abschnitt 1.1 umrissene Fragestellung setzt u. a. die Wahl geeigneter Randbedingungen voraus, die die Gültigkeit des vereinfachten Verfahrens beschränken. So beschäftigt sich diese Arbeit ausschließlich mit dem Verhalten des Rauchgases hinter dem Überlaufrand und setzt damit voraus, dass bereits ein vollständig entwickelter, horizontaler Rauchgasstrom vorhanden ist. Für die Anwendung der neuen Formeln müssen damit der konvektive Wärmestrom sowie der Massenstrom der Rauchgasschicht unterhalb der Geschossdecke kurz vor dem Eintritt in das Atrium bekannt sein. Darüber hinaus müssen das Temperatur- bzw. Geschwindigkeitsprofil des horizontalen Rauchgasstroms in etwa der im Anlagenteil E auf den Seiten 153 bis 158 dargestellten Form entsprechen. Sofern der Rauchgasstrom durch einen Brandherd verursacht wird, der direkt an das Atrium angrenzt, können entsprechende Formeln zur Abschätzung des Wärme- und Massenstroms aus der Literatur entnommen werden (z. B. [8, 22]).

Ebenfalls vorausgesetzt werden in dieser Arbeit weitgehend parallel verlaufende Strömungslinien im horizontalen Rauchgasstrom vor dem Überlaufrand. Dies ist in der Praxis nicht immer der Fall. Sofern hinter der Austrittsöffnung beispielsweise ein Balkon bzw. Umgang angeordnet ist, muss sichergestellt sein, dass der Rauchgasstrom durch seitliche Wände oder Rauchschrägen bis zum Überlaufrand geführt wird (s. Abb. 4). Andernfalls kann es zu einer unkontrollierten Ausbreitung des Rauchgases unterhalb des Balkons und einer damit verbundenen erhöhten Luftermischung in diesem Bereich kommen. Eine Betrachtung dieses, als „unchannelled flow“ bezeichneten Falles findet sich beispielsweise in [9].

Des weiteren können mit der in dieser Arbeit verwendeten Software lediglich auftriebsgetriebene Strömungen mit geringen Geschwindigkeits- und Dichteunterschieden modelliert werden. Davon abweichende Sonderfälle (Flash-Over, Schockwellen infolge Explosionen etc.) werden in dieser Dissertation nicht berücksichtigt. In Übereinstimmung mit

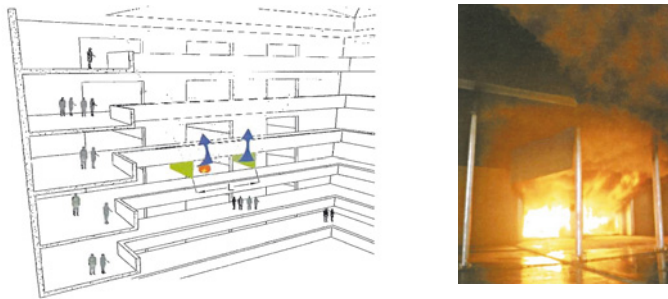


Abb. 4: Anordnung von Rauchschürzen seitlich des Überlaufplumes (channelled flow), Auszug aus [2]

den bestehenden vereinfachten Berechnungsformeln (z. B. „BRE Spill-Plume-Methode“, vgl. Abs. 2.3) werden in diesem Zusammenhang zusätzlich maximale Rauchgastemperaturen in Höhe von $\theta_{b,max} \leq 350^\circ\text{C}$ vorausgesetzt. Für die hier betrachtete Anfangsphase des Brandes (Zeit bis zum Abschluss der Gebäudeevakuierung) und üblichen Brandlasten ist diese Annahme gerechtfertigt. Für höhere Rauchgastemperaturen existiert nach Kenntnis des Autors derzeit kein vereinfachtes Verfahren, dass eine ausreichende Zuverlässigkeit aufweist.

Eine weitere Einschränkung der betrachteten Problemstellung besteht in der Ausbildung des aufsteigenden Plumes selbst. In der Praxis treten in Abhängigkeit der geometrischen Randbedingungen und der Wärmefreisetzungsrate des Brandherdes zwei mögliche Fälle auf. Überschreitet das Verhältnis von Trägheits- zu Auftriebskraft der Rauchgase am Überlauftrand einen spezifischen Grenzwert, so bildet sich ein freier bzw. zweiseitiger Plume aus (s. Abb. 5, links). Bei geringen Trägheits- und großen Auftriebskräften entsteht hingegen ein anliegender bzw. einseitiger Plume (s. Abb. 5, rechts). In dieser Arbeit wird ausschließlich auf den zweiseitigen, freien Plume eingegangen. Da die Lufteinmischung in einen einseitigen Plume geringer ausfällt, können die im Rahmen dieser Arbeit gewonnenen Erkenntnisse auf der sicheren Seite liegend auch für den einseitigen Fall verwendet werden.

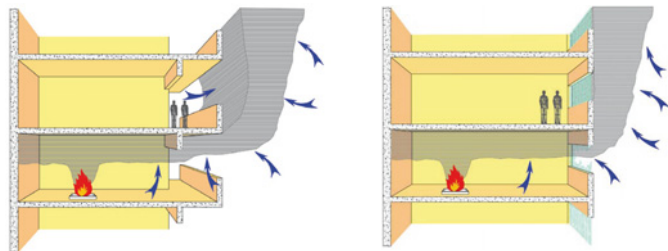


Abb. 5: Prinzipskizze zum freien (links) und anliegenden Plume (rechts), Auszug aus [2]

1.4 Zusammenfassung der Forschungsziele

Vor allem im Entwurfsstadium eines Gebäudes besteht der Bedarf, die Entrauchung von Atrien auch ohne aufwendige CFD-Simulationen und Modellversuche beurteilen zu können. Dazu stehen dem Ingenieur eine Reihe von Methoden und Näherungsformeln zur Verfügung, mit denen die erforderliche Anzahl und Größe von natürlichen Rauchabzugsöffnungen bestimmt bzw. maschinelle Rauchabzüge dimensioniert werden können. Alle vereinfachten Methoden beschränken sich derzeit auf Atrien, die ein ausreichend großes Rauchgasreservoir aufweisen (s. Abs. 2.4.1–2.4.9). Für kleine Atrien steht dem Ingenieur derzeit ausschließlich die BRE Spill-Plume-Methode zur Verfügung (s. Abs. 2.3 und Anlagenteil B), die aufgrund ihrer Komplexität für den baupraktischen Einsatz nur bedingt geeignet ist. Wünschenswert ist in diesem Zusammenhang eine Näherungsformel, die weitgehend unabhängig von der Atriengröße eine ausreichend genaue Abschätzung des Massenstroms im aufsteigenden Plume liefert. Unter diesem Gesichtspunkt soll mit dieser Arbeit die folgende Fragestellung geklärt werden:

- **Entwicklung einer Näherungsformel für den Massenstrom im Überlaufplume innerhalb eines Atriums variabler Größe**

Es soll eine Näherungsformel entwickelt werden, mit der die Lufteinmischung in einen aufsteigenden Überlaufplume auf der sicheren Seite liegend abgeschätzt werden kann. Die Näherungsformel soll dabei für alle Atrien gelten und möglichst robust ausfallen (unabhängig von der Grundrissform des Atriums, Berücksichtigung der Breite des Überlaufplumes etc.).

Für die Entwicklung der Näherungsformel ist die Kenntnis der Lufteinmischung in einer Vielzahl unterschiedlicher Fälle erforderlich (Variation der geometrischen Form von Atrium und Brandraum unter Berücksichtigung unterschiedlicher Wärmefreisetzungsraten). Im Rahmen dieser Arbeit sollen die hierzu erforderlichen Ergebnisse über eine detaillierte CFD-Studie ermittelt werden, deren Modell zuvor auf der Grundlage von experimentellen Untersuchungen [10, 4] validiert wird. Da für die geplanten Simulationsrechnungen eine ganz erhebliche Rechenleistung erforderlich ist, sollen im Vorfeld Möglichkeiten für den Einsatz der neuen GPU-Prozessortechnologie in der CFD-Simulation im Brandschutz untersucht werden:

- **Anpassung einer bestehenden CFD-Software an die GPU-Prozessortechnologie**

Es sollen wesentliche Algorithmen der CFD-Software „Fire Dynamics Simulator“ [7] umgeschrieben und an die massiv parallele GPU-Technik angepasst werden. Neben der grundsätzlichen Machbarkeit soll dabei vor allem die erreichbare Rechenbeschleunigung für unterschiedliche Problemstellungen aus den Bereichen Fluidmechanik, Wärmetransport durch Strahlung und Verbrennungsmodellierung ermittelt werden. Die im Rahmen dieser Arbeiten gewonnenen Erkenntnisse bzw. Problemunkte

sollen dargestellt und mit Blick auf das Aufwand-Nutzen-Verhältnis bewertet werden (Grundlage für die Entscheidung für oder gegen eine vollständige Anpassung der verwendeten Brandsimulationssoftware).

1.5 Vorgehensweise

Zunächst werden die bisher durchgeführten Untersuchungen zu den oben beschriebenen Überlaufplumes zusammengefasst (vgl. Abs. 2). Die Literaturrecherche beschränkt sich dabei auf die Zusammenfassung der bekannten Rechenmethoden zur Ermittlung der Lufteinmischung innerhalb von Atrien und verzichtet auf die ausführliche Darstellung der den Formeln zugrundeliegenden Versuchsergebnisse.

Im Abschnitt 3 werden die Grundlagen der GPU-Prozessortechnologie in kurzer Form dargestellt. Zum besseren Verständnis dieses Abschnitts enthält der Anlagenteil A darüber hinaus einen Kurzüberblick über einige Aspekte zur parallelen Rechentechnik im Allgemeinen.

Im Abschnitt 4 werden die vorgenommenen Änderungen am Quellcode der verwendeten Software FDS erläutert. Dabei wird neben der allgemeinen Vorgehensweise vor allem auf die Bereiche in der Software eingegangen, bei denen Änderungen im größeren Umfang erforderlich waren. In diesem Zusammenhang wird im Abschnitt 4 besonderes Augenmerk auf den Strahlungsübertragung in gasförmigen Medien gelegt (vgl. Abs. 4.5), da in diesem Bereich vom Autor vollständig neue Algorithmen entwickelt wurden.

Im Abschnitt 5 wird die neue Software anhand von einfachen Beispielen verifiziert. Hierdurch wird sichergestellt, dass die neue GPU-Version gegenüber der Basisversion identische Ergebnisse liefert. Ebenfalls wird in diesem Abschnitt die Steigerung der Rechenleistung ermittelt, die durch die Modifikation der Software erreicht wurde (Performance-Tests).

Der Abschnitt 6 beinhaltet die Beschreibung der durchgeführten Simulationsrechnungen zur Fragestellung des Massenstroms im Überlaufplume. Dazu erfolgt zu Beginn des Abschnitts eine Validierung der verwendeten Modelle über die in [10, 4] beschriebenen experimentellen Untersuchungen (vgl. Abs. 6.1–6.5). Im Anschluss werden zusätzliche Untersuchungen mit geänderten Randbedingungen durchgeführt (Variation der Atriengeometrie, Wärmefreisetzungsrate etc.), auf deren Grundlage neue Zusammenhänge für die Lufteinmischung in den Überlaufplume innerhalb von Atrien variabler Größe entwickelt werden (vgl. Abs. 6.6–6.7). Alle Ergebnisse werden im Abschnitt 7 zusammengefasst.

2 Stand des Wissens und der Technik

2.1 Einführung in die Modellbildung

In Abb. 6 ist das in der Literatur gängige Modell zur Rauchentwicklung beim Überlaufplume schematisch dargestellt. Hierbei bilden die Rauchgase des Brandherdes zunächst eine heiße Rauchgasschicht unter der Decke im Brandraum. Unter der Voraussetzung, dass kontinuierlich Frischluft nachströmen kann, entweichen die heißen Rauchgase anschließend über einen Überlauftrand (spill edge) in das Atrium und steigen infolge von Auftrieb bis zur Decke auf. Während des gesamten Vorgangs findet eine Einmischung von Luft in die Rauchgase statt, so dass die Menge des Rauchgases bei gleichzeitigem Absinken der Gastemperatur ansteigt. Für die Dimensionierung der Rauchabzüge in der Atriumdecke ist vor allem die Rauchgasmenge interessant, die in die Heißgasschicht des Atriums eingetragen wird (M_a).

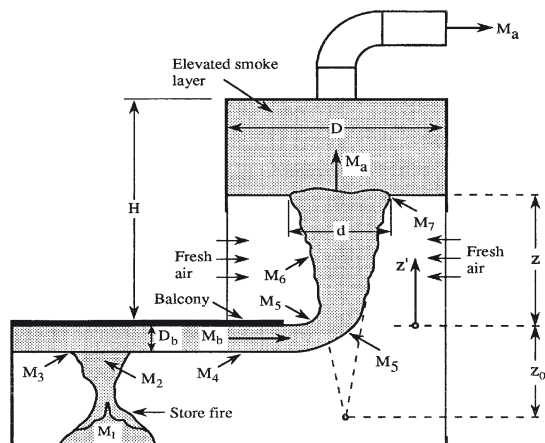


Abb. 6: Modell zur Rauchentwicklung im Atrium bei einem Brand im angrenzenden Raum (Skizze nach [11])

Wie aus experimentellen Versuchen bekannt ist, fällt die Lufteinmischung in den Zonen M_1 , M_3 und M_4 vergleichsweise gering aus, so dass nach [11] vereinfacht die folgende Beziehung angenommen werden kann:

$$M_a \sim M_2 + M_5 + M_6 \quad (1)$$

Thomas [12] und Hansell [8] weisen darüber hinaus darauf hin, dass die Lufteinmischung in der aufsteigenden Plume M_6 näherungsweise mit der Einmischung über einer linienförmigen virtuellen Feuerquelle gleichgesetzt werden kann. Der Ursprung des virtuellen Brand-

herdes befindet sich dabei im Abstand z_0 unter dem Überlauftrand (vgl. Abb. 6). Für einen derartigen „ungestörten Brandherd“ wurden bereits von Lee und Emmons in [13] experimentelle Untersuchungen durchgeführt (vgl. folgender Abschnitt).

2.2 Luftereinmischung in einen freien linienförmigen Brandherd

Bereits 1969 wurden von Lee und Emmons [13] Untersuchungen zur Luftereinmischung in den Plume eines freistehenden linienförmigen Brandherdes durchgeführt. Für den ungestörten, vom Brandherd entfernten Bereich des Plumes ($z_k > z_{k,min}$) geben Lee und Emmons aus theoretischen Überlegungen heraus den folgenden Zusammenhang an:

$$M_p' = 2^{5/6} \rho \alpha^{2/3} (1 + \lambda^2)^{1/6} \left(\frac{g Q'}{\rho c_p T_0} \right)^{1/3} z_k \quad (2)$$

Dabei ist Q' der aus dem Brandherd resultierende konvektive Wärmestrom bezogen auf die Brandherdlänge, c_p die spezifische Wärmekapazität der Luft, T_0 die Umgebungstemperatur, z_k die Höhe über dem Ursprung des Brandherdes und ρ die Rauchgasdichte. Die Werte α und λ stellen Konstanten dar, die Lee und Emmons basierend auf experimentellen Untersuchungen mit $\alpha = 0,16$ bzw. $\lambda = 0,9$ angeben. In späterer Literatur werden diese Konstanten vereinfacht in der Einmischungskonstanten C_m zusammengefasst:

$$C_m = 2^{5/6} \alpha^{2/3} (1 + \lambda^2)^{1/6} \quad (3)$$

Während in der Fachwelt über den physikalischen Zusammenhang zwischen Luftereinmischung und konvektivem Wärmestrom über einem linienförmigen Brandherd prinzipiell Einigkeit besteht (vgl. beispielsweise alternative Herleitung der Gleichung (2) in [11] oder [14]), können der Literatur für die experimentell bestimmten Einmischungskonstanten unterschiedliche Angaben entnommen werden. Yuan und Cox fassen in [15] die bis dato ermittelten Werte für die Einmischungskonstanten zusammen. In Tabelle 1 sind die Werte aus [15] erweitert um Angaben aus neuerer Literatur wiedergegeben.

Wie Poreh in [11] ausführt, ergeben sich die in der Tabelle 1 sichtbaren Abweichungen zwischen den Einmischungskonstanten vor allem aus der Art der Versuchsanordnungen (direktes Messen der Rauchgasmenge über eine Abzugshaube oder nachträgliche Ermittlung der Massen über gemessene Temperatur- und Geschwindigkeitsprofile). Eine weitere Vereinfachung der Gleichung (2) ist möglich, wenn ihre Anwendung auf baupraktisch übliche Randbedingungen begrenzt ($T_0 \approx 293 \text{ K}$, $g \approx 9,81 \text{ m/s}^2$, $c_p \approx 1,0 \text{ kJ/(kg·K)}$) und der Dichteunterschied zwischen dem Plume und der Umgebungsluft vernachlässigt wird ($\rho \approx \rho_0 \approx 1,2 \text{ kg/m}^3$):

$$M_p' = 0,364 C_m Q'^{1/3} z_k \quad (4)$$

Autor	C_m [–]	α [–]
Rouse et. al. (1952) [16]	0,58	0,162
Chen and Rodi (1975) [17] (Rouse's data)	0,48	0,126
Lee and Emmons [13]	0,58	0,160
Yokoi (1960) [19] and Zukoski (1980) [18] (Yokoi's data)	0,52	0,125
Kotsovinos (1975) [20]	0,66	0,200
Ramparian and Chanrasekara (1989) [21]	0,50	0,117
Yuan and Cox [15]	0,51	0,126
Poreh (1998) [11]	0,44	0,110
Grove and Quintiere (2002) [22]	0,65	0,189

Tabelle 1: Vorgeschlagene Werte für die Einmischungskonstanten C_m und α , Angaben basieren auf [15] und zusätzlicher Literatur

Für einen linienförmigen Plume der Länge W ergibt sich die Luftermischung durch Einsetzen der Beziehungen $M_p' = M_p / W$ und $Q' = Q / W$ in die Gleichung (4) wie folgt:

$$M_p = 0,364 C_m Q^{1/3} W^{2/3} z_k \quad (5)$$

2.3 Die „BRE Spill-Plume-Methode“ für den Überlaufplume

Erste Versuche zum Überlaufplume in Atrien wurden in den siebziger Jahren von Morgan und Marshall durchgeführt (vgl. [23, 24]). Auf der Grundlage von experimentellen Daten entwickelten die Autoren eine (vergleichsweise komplexe) Methodik zur Ermittlung der Luftermischung in einen aufsteigenden Überlaufplume, die in der Literatur unter dem Begriff „BRE Spill-Plume-Methode“ bekannt geworden ist. In [25] bzw. [26] stellen Morgan und Hansell zusätzliche Aspekte zum horizontalen Massenstrom im Brandraum sowie zur Vorgehensweise bei anliegenden Plumes (adhered plumes) dar. Die prinzipielle Anwendbarkeit der BRE-Methode wurde zwischenzeitlich durch weitere Experimente bestätigt (vgl. [27, 28]). Im Rahmen dieser Dissertation werden die notwendigen Berechnungen anhand des in [2] enthaltenen Beispiels nachvollzogen (s. Anlagenteil B) und die wesentlichen Aspekte der Methodik kurz zusammengefasst (vgl. folgenden Absätze). Für eine weitergehende Erläuterung des theoretischen Hintergrundes und Hinweise zur praktischen Anwendung wird auf [23, 8, 2] verwiesen.

Die BRE-Methode ist ausschließlich für Fälle gültig, in denen sich ein deckenunterseitiger, vollständig entwickelter turbulenter Rauchgasstrom in ein Atrium ergießt und dort infolge von Auftriebskräften aufsteigt. Voraussetzung für die Anwendung ist außerdem, dass die Stromlinien weitgehend parallel verlaufen bzw. dass der Rauchgasstrom durch seitliche Wände oder Rauchgasschürzen bis hin zum Überlauftrand geführt wird. Zusätzli-

che Untersuchungen in [29] legen dar, dass die Anwendung der BRE-Methode darüber hinaus auf eine maximale Rauchgastemperatur in Höhe von $\theta_{b,max} \approx 350^\circ\text{C}$ beschränkt ist. Eine Methodik zur Ermittlung der Luftermischung in Plumes mit höheren Rauchgastemperaturen ist derzeit nicht verfügbar [4].

Als Eingangswerte zur Berechnung der Luftermischung werden der konvektive Wärmestrom Q , der Massenstrom M_b , die Schichthöhe des Rauchgases d_b sowie die mittlere (oder alternativ die maximale) Rauchgastemperatur θ_b (bzw. $\theta_{b,max}$) am Überlauftrand benötigt. Mit Ausnahme des Wertepaares M_b/d_b ist es dabei ausreichend, lediglich über zwei der vorstehend aufgeführten Werte zu verfügen. Die übrigen Eingangswerte können über die im Anlagenteil B, Zeile 16 bis 19 wiedergegebenen Zusammenhänge überschlägig ermittelt werden. Der in den Formeln verwendete Wert C_d berücksichtigt die Änderung des Temperatur- bzw. Geschwindigkeitsprofils unter dem Austrittsrand im Falle eines vorhandenen Unterzuges. Während bei geringen Unterzughöhen ebenso wie bei fehlenden Unterzügen mit $C_d = 1,0$ gerechnet werden sollte, ist der Wert C_d im Falle eines „großen Unterzuges“ auf $C_d = 0,6$ abzumindern.

Im Anschluss an die Berechnung der Eingangswerte wird mit dem Zusammenhang in Zeile 22 (vgl. Anlagenteil B) der Massenstrom im aufsteigenden Plume in Höhe der Deckenunterkante $M_{a,z=0}$ bestimmt. Der von Morgan und Marshall in [23] ermittelte Wert für die Einmischungskonstante im Rotationsbereich ($\alpha' = 0,90$) wird von Morgan und Hansell in [26] nachträglich korrigiert ($\alpha' = 1,10$). Durch den höheren Wert soll evtl. auftretenden Anomalien in diesem Bereich Rechnung getragen werden.

Mit den Zusammenhängen aus den Zeilen 23 bis 46 kann anschließend der längenbezogene Massenstrom des aufsteigenden Plumes in der Höhe z bestimmt werden. Hierbei gehen die Autoren analog zu Lee und Emmons [13] von einer Gauss'schen Normalverteilung der horizontalen Temperatur- und Geschwindigkeitsprofile aus und verwenden die Beziehung (2) zur Bestimmung des virtuellen Brandherdursprungs z_0 . Bei freien Plumes greifen Morgan und Marshall für die Einmischungskonstante α dabei auf den von Lee und Emmons ermittelten Wert $\alpha = 0,16$ zurück. Bei anliegenden Plumes schlagen die Autoren eine Reduktion von α auf $\alpha = 0,077$ vor. In den Zeilen 47 bis 50 wird abschließend die Luftermischung in den gesamten Plume durch Multiplikation des längenbezogenen Wertes mit der Öffnungsbreite und – falls der Plume über freie Enden verfügt – durch Addition eines entsprechenden Zuschlags δM_r bestimmt.

Ein wesentlicher Punkt bei Anwendung der BRE-Methode besteht in der Unterscheidung zwischen Atrien mit großem und kleinem Rauchgasreservoir. Morgan und Marshall stellen fest, dass der Massenstrom in Atrien mit ausreichend großem Rauchgasreservoir prinzipiell geringer ausfällt, als dies bei kleinen Atrien der Fall ist. In ihrer Methodik nehmen die Autoren daher eine Korrektur des z -Wertes vor, sofern die Höhe der sichtbaren

Rauchgasschicht den Wert $0,67 \cdot A_a^{0,5}$ überschreitet (vgl. Anlagenteil B, Zeile 33). Kritik an diesem Konzept der „effective layer height“ wird in späterer Literatur von Law [30] angebracht (vgl. Abs. 2.4.1). Auch Harrison und Spearpoint weisen darauf hin, dass diese Korrektur des z-Werte als unzweckmäßig zu bezeichnen ist (vgl. Abs. 2.4.6).

2.4 Alternative Berechnungsansätze

Infolge der Komplexität der BRE-Methode und der damit einhergehenden Fehleranfälligkeit ist der baupraktische Einsatz dieser Methode nur bedingt möglich. In späterer Literatur wird daher nach vereinfachten Formeln gesucht, die einen möglichst großen Bereich praktischer Anwendungsfälle abdecken und gleichzeitig eine ausreichend genaue Abschätzung der gewünschten Werte liefern. Die wesentlichen Ansätze hierzu werden in den folgenden Abschnitten kurz zusammengefasst.

2.4.1 Law [30] (1986)

Law wertet in [30] die von Morgen und Marshall [23, 24] durchgeführten experimentellen Untersuchungen neu aus. Unter Berücksichtigung des von Yokoi [19] für linienförmige freie Brandherde gefundenen Zusammenhangs zwischen der Temperatur im aufsteigenden Plume und der Höhe z,

$$\theta_{a,max} = \frac{6,89}{z + z_0} \left(\frac{Q}{W} \right)^{2/3}, \quad (6)$$

gibt die Autorin zunächst Abschätzungen für den Abstand zur virtuellen Brandquelle z_0 an (vgl. Abb. 6, S. 21). Durch Einsetzen der von Morgan und Marshall experimentell gemessenen Maximaltemperaturen im aufsteigenden Plume $\theta_{max,z=31 \text{ cm}}$ bzw. $\theta_{max,z=40 \text{ cm}}$ ergibt sich der Abstand zum virtuellen Brandherd zu $z_0 \sim 2/3 H$, wobei H die Höhe des Brandraums darstellt. Unter der Annahme, dass die gemessenen Maximaltemperaturen im horizontalen Massenstrom direkt am Überlaufbrand in etwa der Temperatur in der Höhe $z = 0 \text{ cm}$ entsprechen, ($\theta_{b,max} \approx \theta_{max,z=0 \text{ cm}}$) ermittelt Law darüber hinaus den Wert z_0 alternativ zu $z_0 \approx 1/2 H$.

Weiterhin gibt Law einen Zusammenhang zwischen dem Massenstrom am Überlaufpunkt M_b und der Wärmefreisetzungsrate des Brandherdes Q bzw. der Öffnungsbreite W an:

$$M_b = 0,025 (QW^2)^{1/3} \quad (7)$$

Law geht dabei von der Annahme aus, dass die Massenströme M_b und $M_{a,z=0}$ annähernd gleich groß ausfallen und korreliert die experimentell gewonnen Werte zur Lufteinmischung mit den übrigen Randbedingungen. Unter Verwendung der bereits bekannten Beziehung für den Massenstrom im aufsteigenden Plume $M_a \sim Q^{1/3} \cdot W^{2/3} \cdot z$ gibt Law eine

Abschätzung des Massenstroms im aufsteigenden Plume an:

$$M_a = 0,34 (QW^2)^{1/3} (z + 0,15H) \quad (8)$$

In Gleichung (8) ist die Lufteinmischung in die freien Enden des Plumes (3D-Plume) dabei bereits berücksichtigt. In einem Brief an Law [31] weist Morgan auf die Tatsache hin, dass die Gleichung (8) ausschließlich für die geometrischen Randbedingungen der in [23, 24] beschriebenen Experimente gültig ist (Öffnungsbreite des Brandraumes etc.). Darüber hinaus gibt er zu bedenken, dass die mit der Gleichsetzung der Massenströme M_b und $M_{a,z=0}$ verbundene Unschärfe in die Korrelation von Law überproportional eingeht, da Law diesem speziellen Datenpunkt wesentlich größeres Gewicht gegenüber den versuchsbedingt ermittelten Datenpunkten zuweist. Unter Verzicht dieser, von Law getroffenen Annahme wertet Morgan die Daten erneut aus und schlägt einen allein auf der Grundlage von Versuchsergebnissen ermittelten Zusammenhang vor [31]:

$$M_a = 0,40 (QW^2)^{1/3} (z + 0,30H) \quad (9)$$

Neben der vereinfachten Gleichung (8) äußert Law in [30] und [32] ebenfalls Kritik an dem Konzept der „effective layer height“ in der BRE-Methode. Law verweist hier auf die große Abhängigkeit der Ergebnisse in Bezug auf die Höhe des Atriums. Die Tatsache, dass der Zweck der Korrektur vor allem in der Berücksichtigung der Temperaturverteilung unterhalb der Rauchgasschicht besteht, rechtfertigt nach Law nicht die erhebliche Abhängigkeit der Ergebnisse von einem Punkt weit oberhalb der Rauchgas-Grenzschicht.

2.4.2 Thomas [12] (1987)

Thomas wendet in [12] die von Lee und Emmons hergeleitete Beziehung (vgl. Gl. (2)) auf das in Abb. 6, Seite 21 dargestellte Modell an und substituiert die längenbezogenen Werte $M_a' = M_a/W$ und $Q' = Q/W$. Die Lufteinmischung in einen Überlaufplume der Breite W ergibt sich damit wie folgt:

$$M_a = 0,58 \rho \left(\frac{g W^2 Q}{\rho C_p T_0} \right)^{1/3} (z + z_0) \quad (10)$$

In [12], Tabelle 2 gibt Thomas darüber hinaus unterschiedliche Werte für den Abstand zur virtuellen Brandquelle z_0 an ($0,32 H \leq z_0 \leq 0,66 H$). Weiterhin weist er darauf hin, dass aus den bis dato verfügbaren experimentellen Ergebnissen nicht auf einen einheitlichen Wert für den Abstand zur virtuellen Brandquelle geschlossen werden kann und vermutet, dass dieser von der geometrischen Anordnung in den Versuchen abhängig ist. In diesem Zusammenhang sieht Thomas noch Forschungsbedarf.

Im Gegensatz zu der von Law angegebenen Gleichung (8) schließt die Gleichung (10) nicht die Luftermischung in die freien Enden des Plumes ein und ist damit nur für den zweidimensionalen Fall gültig (2D-Plume). Für den Fall eines freien Plumes (inkl. Luftermischung in die Enden) leitet Thomas – basierend auf den Überlegungen von Morgan und Marshall [24] – einen alternativen, vereinfachten Zusammenhang ab und erweitert die Gleichung (10) um einen Korrekturfaktor für die Luftermischung in die Enden:

$$M_a = 0,58 \rho \left(\frac{g W^2 Q}{\rho c_p T_0} \right)^{1/3} (z + z_0) \left(1 + \frac{0,22(z + 2z_0)}{W} \right)^{2/3} \quad (11)$$

Unter Annahme üblicher Randbedingungen für baupraktische Problemstellungen ($T_0 \approx 293 \text{ K}$, $g \approx 9,81 \text{ m/s}^2$, $c_p \approx 1,0 \text{ kJ/(kg·K)}$) und Vernachlässigung des Dichteunterschiedes zwischen dem Plume und der Umgebungsluft ($\rho \approx \rho_0 \approx 1,2 \text{ kg/m}^3$) kann die Gleichung (11) wie folgt vereinfacht werden:

$$M_a = 0,21 Q^{1/3} [W + 0,22(z + 2z_0)]^{2/3} (z + z_0) \quad (12)$$

Hierbei stellt der Term $W + 0,22(z + 2z_0)$ eine effektive Öffnungsbreite dar, die die zusätzliche Luftermischung in die freien Enden des Plumes berücksichtigt.

2.4.3 Law [33] (1995)

Law überprüft in [33] die in [30] ermittelte Näherungsformel (vgl. Abs. 2.4.1) auf der Grundlage von zusätzlichen experimentellen Untersuchungen, die von Hansell et. al. in [29] durchgeführt worden sind. Infolge der zwischenzeitlich erweiterten Basis an experimentellen Daten gibt Law eine überarbeitete Fassung der in [30] ermittelten Näherungsformeln wie folgt an:

$$M_b = 0,31 (QW^2)^{1/3} (z + 0,25H) \quad (13)$$

Auch stellt Law fest, dass die obige Gleichung ausschließlich für Plumes gültig ist, die durch seitliche Rauchschrägen unterhalb des Balkons begrenzt werden (channelled flow). Sofern diese nicht vorhanden sind, weist der horizontale Massenstrom auf Grundlage der Experimente von Hansell et. al. ein eher undefiniertes Verhalten auf. Bis weitere Untersuchungen vorliegen, schlägt Law vor, für einen derartigen Fall die Gleichung (13) dennoch anzuwenden und die Breite des Plumes am Überlauftrand W_e auf der Grundlage der Balkonbreite b wie folgt abzuschätzen:

$$W_e = W + b \quad (14)$$

2.4.4 Poreh [11] (1998)

Poreh entwickelt in [11] ein vereinfachtes Modell für die Luftermischung M_5 im gekrümmten Bereich (vgl. Abb. 6, S. 21), dass ohne die Kenntnis des Abstandes z_0 auskommt. Poreh geht davon aus, dass sich im gekrümmten Bereich der geringere Luftertrag auf der konkaven Seite in etwa mit dem erhöhten Luftertrag auf der konvexen Seite ausgleicht. In diesem Fall hätte der Überlaufplume im gekrümmten Bereich die gleiche Einmischungskonstante, wie im aufsteigenden Bereich und die zusätzliche Luftermischung in der gekrümmten Region $M_5 = M_{a,z=0} - M_b$ entspräche der Luftermischung eines geraden Plumes zwischen $z = -d_b$ und $z = 0$ cm. Der Überlaufplume kann dann mit einem ungestörten geraden Plume gleichgesetzt werden, der in Höhe $z = -d_b$ (untere Grenze der Rauchgasschicht im Brandraum) den Massenstrom M_b aufweist. In diesem Fall ergibt sich aus Gleichung (5):

$$M_b = 0,364 C_m Q^{1/3} W^{2/3} (d_b - z_0) \quad (15)$$

Durch Subtraktion der Gleichung (15) von Gleichung (5) kann der Abstand zum virtuellen Brandherd z_0 aus der Beziehung entfernt werden und man erhält das von Poreh vorgeschlagene vereinfachte Modell:

$$M_{a,2D} - M_b = 0,364 C_m Q^{1/3} W^{2/3} (z - d_b) \quad (16)$$

Auf der Grundlage von vier Testserien, die von Marshall und Harrison [34] durchgeführt wurden, ermittelt Poreh die Einmischungskonstante C_m zu $C_m \approx 0,44$. Poreh weist darauf hin, dass dieser Wert ausschließlich für zweidimensionale Plumes ohne Luftermischung in die Enden und Atrien mit ausreichender Größe gültig ist. Darüber hinaus können sich bei einem vergleichsweise großen Verhältnis von z/W abweichende Luftermischungen ergeben. Poreh sieht in diesem Zusammenhang den Bedarf einer systematischen Parameterstudie für die Einmischungskonstante C_m in Abhängigkeit der vorliegenden geometrischen Randbedingungen.

In [34] wurde eine fünfte Testserie dokumentiert, die eine vergleichsweise kleine Rauchabzugshaube verwendet. Die dort gemessene Luftermischung fällt höher aus und korreliert daher nicht mit der von Poreh ausgestellten Beziehung (16). Poreh führt dies zum einen auf Wirbel zurück, die sich im oberen Bereich der Rauchabzugshaube direkt unter dem Entlüftungsschacht bilden und verweist zum anderen auf eine nicht eindeutig ausgebildete Trennschicht zwischen Rauchgas und rauchgasfreier Schicht.

2.4.5 Thomas et. al. [35] (1998)

Auch Thomas et. al. werten die Ergebnisse der von Marshall und Harrison [34] durchgeführten Tests aus [35]. Dazu entwickeln die Autoren über eine Dimensionsanalyse zunächst eine allgemeine Form für die Lufteinmischung in den aufsteigenden Plume. Die Autoren geben den Zusammenhang zwischen den auf die Öffnungsbreite W bezogenen Größen $M'_a = M_a / W$, $M'_b = M_b / W$, und $Q' = Q / W$ wie folgt an:

$$\frac{M'_a}{Q'} = \alpha \frac{z}{Q^{1/3}} + \beta \frac{M'_b}{Q'} + \gamma \quad (17)$$

Der Parameter α entspricht dem Term „ $0,364 \cdot C_m$ “ in Gleichung (16) und stellt damit ein Maß für die Lufteinmischungsrate in den aufsteigenden Plume dar. Der Wert $\beta \cdot M'_b / Q' + \gamma$ entspricht der Menge an aufsteigendem Rauchgas in Höhe des Überlaufendes ($z = 0$ m). Nach statistischer Auswertung der vier Testreihen aus [34] ergibt sich nach Thomas et. al. die Lufteinmischung in den aufsteigenden Plume im Atrium zu:

$$\begin{aligned} M'_{a,2D} &= 0,16 z Q^{1/3} + 1,20 M'_b + 0,0027 Q' \quad \text{bzw.} \\ M_{a,2D} &= 0,16 z Q^{1/3} W^{2/3} + 1,20 M_b + 0,0027 Q \end{aligned} \quad (18)$$

Die Gleichung (18) entspricht im Wesentlichen dem bereits von Poreh angegebenen Zusammenhang für zweidimensionale Plumes ($C_m = \alpha / 0,364 \sim 0,44$), fällt jedoch einfacher aus, da die Lufteinmischung im Gegensatz zur Gleichung (16) nicht von der Höhe der rauchgasführenden Schicht am Überlaufend d_b abhängig ist. Auch Thomas et. al. weisen darauf hin, dass die Beziehung (18) ausschließlich für zweidimensionale Plumes in Atrien mit ausreichender Größe gültig ist. Für kleine Atrien schätzen die Autoren aus den Versuchsergebnissen der fünften Testreihe in [34] den Parameter α zu $\alpha \sim 0,25$ ($C_m = 0,69$) ab und merken an, dass dieser Wert infolge der großen Streuung bei den Versuchsergebnissen eine entsprechende Unschärfe aufweist. In diesem Zusammenhang sehen die Autoren eine umfangreiche CFD-Studie zur Vertiefung der Kenntnisse für kleine Atrien als wünschenswert an.

Abschließend geben Thomas et. al. eine Abschätzung für die zusätzliche Lufteinmischung in die freien Enden eines Plumes an (dreidimensionaler Plume), wobei sich die Autoren auf die von Hansell et. al. [29] durchgeführten Experimente beziehen. Unter der Annahme, dass das Verhältnis z/W nicht zu groß ausfällt, schlagen die Autoren die Verwendung des folgenden Zusammenhangs vor:

$$M'_{\text{ends}} = 0,09 z Q^{1/3} \quad (19)$$

2.4.6 Harrison und Spearpoint [5] (2008)

In [5] führen Harrison und Spearpoint zusätzliche Untersuchungen für zwei- und dreidimensionale Überlaufplumes durch. Die Autoren weisen darauf hin, dass die von Thomas et. al. [35] bzw. Poreh [11] verwendeten Werte für die Lufteinmischung M_a in Höhe des Überlaufplumes ($z = 0 \text{ cm}$) auf vergleichsweise wenig Messungen beruhen und führen in diesem Zusammenhang zusätzlich Versuche durch. Auf der Grundlage der neuen Ergebnisse sowie den bereits bestehenden Versuchen von Marshall und Harrison [34] bestimmen die Autoren die Parameter β und γ aus Gleichung (17) neu ($\beta = 1,34$ und $\gamma = 0$). Für zweidimensionale Plumes (ohne Lufteinmischung in die Enden) geben Harrison und Spearpoint den folgenden Zusammenhang an:

$$\begin{aligned} M_{a,2D}^I &= 0,16 Q^{1/3} z + 1,34 M_b^I \quad \text{bzw.} \\ M_{a,2D} &= 0,16 Q^{1/3} W^{2/3} z + 1,34 M_b \end{aligned} \quad (20)$$

Des Weiteren weisen Harrison und Spearpoint darauf hin, dass die bisher durchgeführten Versuche zu Plumes mit Einnischung in die freien Enden (3D-Plumes) unterschiedliche Einnischungskonstanten ergeben haben. Die Autoren legen dar, dass sich aus den von Harrison in [10] beschriebenen Testreihen auf eine Einnischungskonstante in Höhe von $\alpha \sim 0,22$ schließen lässt:

$$M_{a,3D}^I = 0,22 z Q^{1/3} + 1,92 M_b^I - 0,0042 Q^I \quad (21)$$

Demgegenüber ergibt die Auswertung der von Hansell et. al. in [29] durchgeführten Versuche eine Einnischungskonstante in Höhe von $\alpha \sim 0,34$:

$$M_{a,3D}^I = 0,34 z Q^{1/3} + 2,64 M_b^I - 0,0083 Q^I \quad (22)$$

Ursächlich für diese Abweichungen sind nach Harrison und Spearpoint die unterschiedlichen Öffnungsweiten W , die in den Versuchen verwendet wurden. Bei der in [10] vorhandenen breiteren Öffnung ($W = 0,60 \text{ m}$) fällt die zusätzliche Einnischung infolge der freien Enden in Bezug auf die Gesamteinnischung geringer aus als bei der in [29] verwendeten schmalen Öffnung ($W = 0,43 \text{ m}$). Um eine Aussage über die zusätzliche Lufteinmischung in die freien Enden des Plumes treffen zu können, führen Harrison und Spearpoint daher zusätzliche Versuche durch und bestimmen die Einnischungskonstante α in Abhängigkeit von der Öffnungsweite (s. Abb. 7).

Aus Abb. 7 ist ersichtlich, dass die Einnischung in die freien Enden des Plumes mit abnehmender Öffnungsweite ansteigt. Harrison und Spearpoint nehmen für die Einnischung – analog zur Einnischung im Innenbereich – eine Proportionalität in Bezug auf $Q^{1/3} \cdot z$ an:

$$M_{\text{ends}}^I = (\alpha_{3D} - \alpha_{2D}) Q^{1/3} z \quad (23)$$

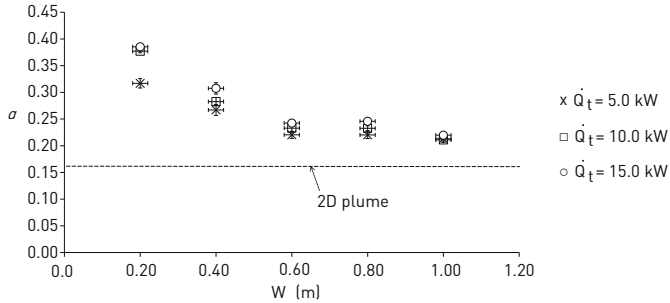


Abb. 7: Einmischungskonstante α in Abhängigkeit der Öffnungsweite W , Auszug aus [5]

Die Versuchsergebnisse zeigen, dass die zusätzliche Einmischung infolge der freien Enden vor allem von der Höhe der Rauchgasschicht im Brandraum d_b abhängig ist. Auf der Grundlage der Versuche bestimmen die Autoren die zusätzliche Einmischung zu

$$\alpha_{3D} - \alpha_{2D} = 0,25 (W/d_b)^{-2/3} \quad (24)$$

und geben für den Massenstrom im Überlaufplume mit freien Enden den folgenden Zusammenhang an:

$$\begin{aligned} M_{a,3D} &= 0,16 Q^{1/3} W^{2/3} z + 1,34 M_b + 0,25 d_b^{2/3} Q^{1/3} z \quad \text{bzw.} \\ M_{a,3D} &= \left(0,16 W^{2/3} + 0,25 d_b^{2/3} \right) Q^{1/3} z + 1,34 M_b \end{aligned} \quad (25)$$

2.4.7 Ko et.al. [9] (2008)

Ko et.al. beschäftigen sich in [9] vor allem mit der Generalisierung der Ansätze zur Ermittlung der Massenströme am Überlaufpunkt M_b sowie im Rotationsbereich M_5 (vgl. Abb. 6, S. 21). Die Autoren führen diesbezüglich eine CFD-Studie durch, in der u.a. auf die Anordnung seitlicher Rauchschürzen verzichtet (unchannelled flow) und ein optionaler Unterzug im Bereich der Brandraumöffnung angeordnet wird. Davon ausgehend, dass W_e die effektive Breite des Plumes am Überlauftrand (vgl. Gl. (14)) und H_w die Öffnungshöhe der Austrittsöffnung unterhalb des (optionalen) Unterzuges darstellen, ergibt sich der vertikale Massenstrom im Atrium in Höhe $z = 0$ cm wie folgt:

$$M_{a,z=0} = 3,00 \left[Q_c^{1/5} L_e^3 \left(H/H_w \right)^{K_c-1} \left(H_w/b \right) \right]^{-0,57} \left(H/L_e H_w \right)^{-2} M_b \quad (26)$$

Für die Ermittlung des in Gleichung (26) enthaltenen Faktors K_c wird an dieser Stelle auf die ausführliche Beschreibung in [9] verwiesen. Neben Gleichung (26) geben die Autoren zusätzlich einen Zusammenhang zwischen dem Massenstrom am Überlauffrand M_b sowie dem Massenstrom in der Brandraumöffnung M_w an (Massenstrom unter dem optionalen Unterzug):

$$M_b = 1,25 \left(H/H_w \right)^{-0,5} M_w \quad (27)$$

2.4.8 NFPA 92B [36] (2009)

Die in den USA gültige Norm zur Auslegung von Entrauchungssystemen in Atrien und großen Hallen [36] verwendet die folgende Gleichung zur Ermittlung der Rauchgasmenge im aufsteigenden Plume:

$$M_b = 0,36 (Q_{tot} W^2)^{1/3} (z + 0,25 H) \quad (28)$$

Wie im Abschnitt A.6.2.2.1 der Norm erläutert, basiert dieser Zusammenhang auf den Experimenten von Morgen und Marshall [23, 24] und der daraus von Law abgeleiteten Gleichung (13). Die Autoren der Norm gehen davon aus, dass der konvektive Wärmestrom am Austrittspunkt etwa 15 % unterhalb der gesamten Wärmefreisetzungsrate liegt ($Q \sim 0,85 \cdot Q_{tot}$) und erhalten daher gegenüber Gleichung (13) eine geringfügig höhere Einmischungskonstante. In [36] wird darauf verwiesen, dass die Gleichung (28) ausschließlich für Höhen unterhalb von 15 m ($z + H < 15 \text{ m}$) und Atrien mit ausreichender Größe gültig ist. Bei kleineren Atrien wird auf die Anwendung der BRE Spill-Plume-Methode (vgl. Abs. 2.3) bzw. einer detaillierten Modellstudie verwiesen, da sich in diesem Fall höhere Einmischungsraten ergeben können. Darüber hinaus ist der Wert z auf einen Maximalwert in Höhe von 80 % der Gesamthöhe des Atriums zu begrenzen ($z \leq 0,8 \cdot H_a$), um eine mögliche Wirbelbildung im oberen Bereich des Atriums zu berücksichtigen (minimum design smoke layer depth).

2.4.9 Harrison und Spearpoint [37] (2010)

Alle bisher beschriebenen Ansätze zur Ermittlung der Lufteinmischung in einen linienförmigen Überlaufplume sind ausschließlich bis zu einer maximalen Höhe z gültig, für die die Gleichungen experimentell validiert sind. In [37] beschreiben Harrison und Spearpoint einen theoretischen Ansatz zur Ermittlung dieser Maximalhöhe. Die Autoren gehen davon aus, dass sich der linienförmige Plume mit zunehmender Höhe in einen axialsymmetrischen Plume umwandelt und vergleichen die in [5] ermittelte Gleichung (25) mit dem für axialsymmetrische Plumes (punktförmiger Brandherd) gefundenen Zusammenhang von Zukoski [18]:

$$M_{a,3D} = 0,071 Q^{1/3} z^{5/3} \quad (29)$$

Aufgrund des in Gleichung (25) enthaltenen zusätzlichen Faktors W weist der linienförmige Brandherd gegenüber dem axialsymmetrischen Plume bei geringen Höhen eine höhere Luftmischung auf. Durch den überproportionalen Anstieg von z in Gleichung (25) steigt die Rauchmenge beim axialsymmetrischen Plume jedoch mit zunehmender Höhe schneller an, so dass die Graphen aus den Gleichungen (25) und (29) einen gemeinsamen Schnittpunkt bilden. Harrison und Spearpoint bestimmen die Höhe dieses Schnittpunktes durch Gleichsetzen der beiden Gleichungen zu:

$$z_{\text{trans}} = 3,4 (W^{2/3} + 1,56 d^{2/3})^{3/2} \quad (30)$$

Die Autoren schlagen vor, für die Rauchgasmenge unterhalb der Höhe z_{trans} die von ihnen ermittelte Gleichung (25) und für z -Werte oberhalb dieses Grenzwertes den von Zukoski aufgestellten Zusammenhang für punktförmige Brandquellen (vgl. Gl. (29)) zu verwenden.

Teil A

Numerische Brand- und Rauchsimulation unter Verwendung von GPU-Prozessortechnologie

3 Einführung in die GPU-Technologie

Grundlage für die Steigerung der Rechenleistung von modernen Computern stellt die Erhöhung der Transistoranzahl auf einem Computerchip dar. Entsprechend empirischen Beobachtungen verdoppelt sich die Anzahl der Transistoren pro Flächeneinheit alle 18 bis 24 Monate. Dieser Sachverhalt wurde erstmalig 1965 von Gordon Moore erkannt und ist seither unverändert gültig („Gesetz von Moore“) [38].

Bis vor wenigen Jahren ging die Erhöhung der Transistoranzahl auch mit einer Erhöhung der Taktrate (gemessen in MHz oder GHz) einher, so dass vor allem die Verarbeitungsgeschwindigkeit der Prozessoren als alleiniges Merkmal für deren Leistungsfähigkeit herangezogen wurde. Die steigende Anzahl von Transistoren auf einem Prozessorkern wurde hauptsächlich für architektonische Verbesserungen verwendet (Vergrößerung interner Cache-Speicher, Verwendung von Pipelineverarbeitung zur ressourcenoptimierten Ausführung von Befehlen etc.). Zwischenzeitlich werden die Entwickler von Prozessoren im Bereich der Verarbeitungsgeschwindigkeit jedoch immer häufiger von den Gesetzen der Physik ausgebremst. So nimmt beispielsweise die Geschwindigkeit, mit der Daten aus dem Hauptspeicher gelesen bzw. in diesen geschrieben werden nicht in dem Umfang zu, wie die Ausführungsgeschwindigkeit von arithmetischen Operationen. Während beispielsweise ein Intel-i486-Prozessor aus dem Jahr 1990 für einen Zugriff auf den Hauptspeicher noch sechs bis acht Maschinenzyklen benötigte, sind dies bei einem modernen Prozessor bereits mehr als 200 Zyklen [39]. Dieser Engpass beim Datentransfer vom und zum Hauptspeicher wird auch als „von-Neumannscher Flaschenhals“ bezeichnet. Ebenfalls erhöht sich mit dem Anstieg der Transistordichte auch die Wärmeentwicklung pro Flächeneinheit. Bei modernen Prozessoren ist daher eine Kühlung nur noch mit hohem Aufwand möglich.

Um auch in Zukunft eine Leistungssteigerung in der Prozesstechnik zu erreichen, besteht seit längerem die Tendenz, die Anzahl der für eine Berechnung verwendeten Prozessoren zu erhöhen und die Problemstellung verteilt auf mehreren separaten Prozessoren bzw. Prozessorkernen simultan zu lösen. Während bei diesem Ansatz zwar die Rechenzeit selbst nicht verkürzt wird, kann die Gesamtzeit für die Lösung der Problemstellung aufgrund der simultanen Bearbeitung dennoch erheblich reduziert werden. Rechnersysteme, die einen derartigen Ansatz verfolgen, werden als Parallelrechner bezeichnet. Einen Überblick über die Klassifizierung der Parallelrechner sowie grundlegende Aspekte, die für das Verständnis dieser Arbeit von Bedeutung sind, sind zusammenfassend im Anlagenteil A dargestellt.

3.1 Grundlagen der GPU-Technik

Für den Ingenieur bzw. wissenschaftlich-technischen Anwender, der große Datensätze parallel verarbeiten muss, bietet sich seit wenigen Jahren eine neue Möglichkeit, erforderliche Berechnungen wesentlich effizienter durchzuführen. Im Gegensatz zu den Herstellern von CPUs verfügen die Grafikkartenhersteller mit dem von Ihnen verbauten Graphics Processing Units (GPUs) über einen Prozessortyp, der insbesondere für rechenintensive wissenschaftlich-technische Aufgaben geeignet ist. Ausgangspunkt für die Entwicklung dieses spezialisierten Prozessortyps war die bei der Grafikkartengrafik häufig auftretende Problemstellung, viele voneinander unabhängige Datensätze verarbeiten zu müssen. So eignen sich die Hauptaufgaben der 3D-Computergrafik (Koordinatentransformationen, Ausschluss nicht sichtbarer grafischer Primitive, Bildrendering über Shader etc.) im besonderen Maße für eine Parallelisierung, da alle Koordinaten und Primitive voneinander unabhängig die gleichen Berechnungsschritte durchlaufen müssen.

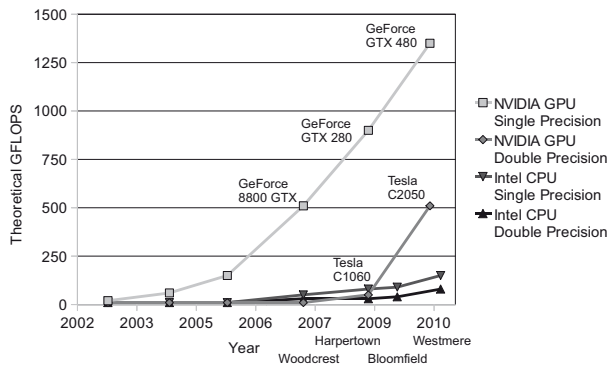


Abb. 8: Vergleich der Rechenleistung von handelsüblichen GPU-Prozessoren der Firma NVIDIA mit herkömmlichen CPUs, Angabe der Rechengeschwindigkeit in Fließkommaoperationen pro Sekunde (FLOPS) [40]

Ausgehend von der theoretisch maximal erreichbaren Rechengeschwindigkeit gemessen in Floating Point Operations per Second (FLOPS) weisen die GPU-Prozessoren gegenüber den CPUs derzeit einen erheblichen Geschwindigkeitsvorteil auf (vgl. Abb. 8). Hierbei ist jedoch zu berücksichtigen, dass der tatsächlich erreichbare Geschwindigkeitsvorteil neben der theoretischen Rechenleistung vor allem von der vorliegenden Problemstellung und deren Implementierung innerhalb der Software abhängt. Je komplexer ein Algorithmus und je größer der darin enthaltene sequenzielle (nicht parallele) Codeanteil ausfällt, desto geringer ist der Geschwindigkeitsvorsprung eines GPU-Prozessors gegenüber einer CPU (vgl. Anlagenteil A.5). Auch muss berücksichtigt werden, dass die GPU-Prozessoren ursprünglich für Berechnungen mit einfacher Genauigkeit ausgelegt waren und ihre volle

Rechenleistung daher vor allem in diesem Bereich zur Geltung kommt. Bei Verwendung der einfachen Genauigkeit (single precision) können gegenüber der doppelten Genauigkeit (double precision) nur sieben bis acht Dezimalstellen berücksichtigt werden, was eine hohe Stabilität des verwendeten Lösungsverfahrens voraussetzt. Nicht jeder bestehende Algorithmus kann mit einer derartigen Stabilität aufwarten und damit die volle Rechenleistung von GPU-Prozessoren ausnutzen.

Der erhebliche Geschwindigkeitsvorteil der GPU gegenüber einer CPU liegt vor allem im grundlegend abweichenden Prozessoraufbau begründet. Während bei einer CPU ein möglichst universeller Parallelismus in allen Bereichen des Prozessors (Arithmetik, Instruktionen, Speicherzugriffe) angestrebt wird, arbeitet eine GPU in großen Teilen nach dem SIMD-Prinzip (vgl. Anlagenteil A.1). Ein und dieselbe arithmetische Instruktion wird auf einen ganzen Block von Daten – soweit möglich – parallel angewendet. Zudem sind auf einer GPU Cache und Steuereinheiten (control units) ganz massiv zu Gunsten der Anzahl der Verarbeitungseinheiten (arithmetic logic units) reduziert (vgl. Abb. 9). Unter der Voraussetzung, dass auf jeden Datensatz die gleiche mathematische Instruktion angewendet wird, werden auf diese Weise gegenüber einer herkömmlichen CPU wesentlich mehr arithmetische Operationen pro Zeiteinheit erreicht. Letztendlich weisen die Grafikprozessoren bereits vom Grundsatz her eine größere Anzahl von Transistoren auf und sind den CPU-Prozessoren daher überlegen. So verfügt der GPU-Prozessor auf dem aktuellen Kartenmodell der Firma NVIDIA (GeForce GTX 580) über ca. 3.000 Mio. Transistoren während auf dem aktuellen Sechskern-Prozessor der Firma Intel (Intel Core i7-980X) lediglich ca. 1.170 Mio. Transistoren vorhanden sind¹.

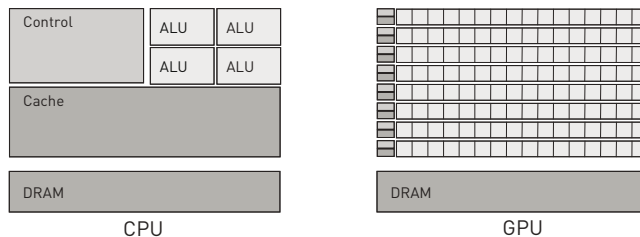


Abb. 9: Reduktion der Chipfläche für Steuereinheiten (control units) und Cache zugunsten von mehr arithmetischen Recheneinheiten (ALU) [40]

Neben der Anzahl der arithmetischen Operationen besteht eine der größten Schwierigkeiten bei der Erhöhung der Rechenleistung in der Bereitstellung der zu verarbeitenden Datensätze aus dem Hauptspeicher (vgl. Abb. 9, DRAM). Während das Rechenvermögen

¹ Der Unterschied zwischen CPU und GPU wird umso deutlicher, wenn berücksichtigt wird, dass ca. ein Drittel der Chipfläche einer CPU für den Cache benötigt wird und damit für arithmetische Operationen nicht zur Verfügung steht.

der Prozessoren in den letzten Jahren kontinuierlich angestiegen ist, konnte die Zugriffszeit auf den Hauptspeicher nur um durchschnittlich etwa 10 % gesteigert werden [41]. Die Folge sind Leerlaufzeiten innerhalb des Prozessorkerns, die insbesondere dann zutage treten, wenn große Datenmengen aus und in den Hauptspeicher bewegt werden müssen. Auch in diesem Bereich weisen die GPU-Prozessoren Vorteile auf. Das derzeit aktuelle Kartenmodell der Firma NVIDIA (GeForce GTX 580) ist auf eine Datentransferrate von und in den Hauptspeicher in Höhe von ca. 192,4 GB/s ausgelegt [42]. Die Datentransferraten aktueller CPU-Prozessoren vom und zum Hauptspeicher liegen dagegen lediglich bei maximal etwa 32 GB/s [43].

Aufgrund dieser für wissenschaftlich-technische Zwecke günstigen Eigenschaften der Grafikprozessoren hat sich in letzter Zeit mit dem General Purpose Computation on Graphics Processing Units (GPGPU) ein Entwicklungszweig herausgebildet, in dem mathematisch-technische Problemstellungen auf GPU-Prozessoren umgesetzt werden. Sinnvoll ist eine derartige Entwicklung im besonderen aufgrund der vergleichsweise günstigen Kosten, mit denen bei der Anschaffung von Systemen auf GPU-Basis zu rechnen ist. Bereits auf einer handelsüblichen 3D-Grafikkarte für wenige hundert Euro befinden sich mehrere hundert Prozessorkerne, die – sofern programmtechnisch korrekt angesteuert – eine Berechnung massiv beschleunigen können.

3.2 Hardware im GPGPU-Bereich

Als Marktführer im Bereich der GPGPU hat sich bis zum heutigen Tag die Firma NVIDIA etabliert. Obwohl die Firma Advanced Micro Devices (AMD) unter der Marke ATI® ebenfalls Hardware für den GPGPU-Bereich anbietet, wird diese in der Praxis kaum verwendet. Ursächlich hierfür ist neben dem weniger ausgereiften Programmiermodell (ATI® Stream) vor allem die fehlende Unterstützung für Berechnungen mit doppelter Genauigkeit (double precision, bin64) sowie der Mangel an Speicherschutzfunktionalität (vgl. Anlagenteil A.3). Im Rahmen dieser Arbeit wird daher ausschließlich Hardware der Firma NVIDIA verwendet.

Es gibt eine Vielzahl unterschiedlicher GPU-Systeme, die sich im Wesentlichen durch die verwendete Prozessorgeneration, die Ausbaustufe sowie die Prozessoranzahl unterscheiden. Da der Ursprung der GPGPU im Bereich der 3D-Computergrafik zu finden ist, können auch heute noch handelsübliche Grafikkarten – wie diese beispielsweise für den Computerspielemarkt unter der Bezeichnung „GeForce“ angeboten werden – zur Rechenbeschleunigung verwendet werden. Dem günstigen Anschaffungspreis für derartige Karten steht ein begrenzter Hauptspeicher (derzeit max. 1,5 GB beim Modell GeForce GTX 580) sowie die nicht vorhandene ECC-Funktion (vgl. Anlagenteil A.3) gegenüber. Die ein-

zelenen Karten unterscheiden sich vor allem in der Anzahl der GPU-Prozessoren auf der Karte, der Ausbaustufe (Anzahl der freigeschalteten CUDA-Kerne) sowie der Größe des Arbeitsspeichers. Eine Auswahl gängiger Kartenmodelle ist in Tabelle 2 wiedergegeben.

GPU Architecture		Gene- ration	Number of GPUs	Number of CUDA-Cores	DRAM [GB]	Bandwidth [GB/s]	ECC
GeForce	GTX 280	Tesla	1	240	1,0	141,7	–
	GTX 295	Tesla	2	2 x 240	2 x 0,9	2 x 111,9	–
	GTX 470	Fermi	1	448	1,25	133,9	–
	GTX 580	Fermi	1	512	1,5	192,4	–
Tesla	C1060	Tesla	1	240	4,0	102	–
	S1070	Tesla	4	4 x 240	16	4 x 102	–
	C2050/70	Fermi	1	448	3,0/6,0	144	x
	S2050/70	Fermi	4	4 x 448	12/24	4 x 144	x
		Kepler	Releasedate: Early 2012				

Tabelle 2: Eigenschaften von ausgewählten GPU-Karten der Firma NVIDIA

Insbesondere für professionelle Clustersysteme bieten sich die eigens für den Bereich der GPGPU entwickelten Karten mit der Bezeichnung „Tesla“ an. Diese sind bei Bedarf auch als 19“-Rack lieferbar und verfügen sowohl über eine ECC-Funktion als auch wesentlich mehr Arbeitsspeicher (s. Abb. 10).



Abb. 10: Tesla C2070 mit 515 GFlops (bin64) und 6 GB DRAM (links), Tesla S2070 im 19“-Format mit 2.060 GFlops (bin64) und 24 GB SRAM (rechts)

Neben der Geschwindigkeit und dem Arbeitsspeicher sind noch andere Punkte wesentlich bei der Wahl eines geeigneten Systems. So beschränkte sich die verfügbare Prozes-
sortechnik im GPU-Sektor über längere Zeit auf die einfache Genauigkeit, in der vier Byte
pro Gleitkommazahl verwendet werden (single precision, bin32). Aufgrund des Ur-
sprungs der Technologie im grafischen Bereich war die heute in wissenschaftlich-techni-

schen Anwendungen häufig verwendete doppelte Genauigkeit mit acht Byte pro Gleitkommazahl (double precision, bin64) prinzipiell nicht erforderlich. Zwischenzeitlich verfügen jedoch viele der erhältlichen Grafikkarten auch über die Möglichkeit, Gleitkomma-Operationen mit doppelter Genauigkeit durchzuführen. Die seit Mitte 2010 verfügbaren Karten der Fermi-Generation erfüllen darüber hinaus zusätzlich den Standard für arithmetische Berechnungen mit binären Gleitkommazahlen IEEE 754–2008 [44]. Die Fähigkeiten der einzelnen GPU-Karten werden für NVIDIA-Produkte durch die GPU-Spezifikation angegeben. Einen Überblick über die derzeit vorhandenen Spezifikationen ist in der Tabelle 3 wiedergegeben.

GPU Specification	Model	Features
sm_10	Tesla x870, Quadro FX 4600/5600, Quadro Plex 2100 Model S4/1000 Model IV	• ISA_1 / Basic features
sm_11	GeForce 9800/9600/8800, GT 130, GTS 150/250	• atomic memory operations on global memory
sm_12	GeForce 210, GeForce GT 220/240, Quadro FX 380LP	• atomic memory operations on shares memory • vote instructions
sm_13	GeForce GTX 2xx, Tesla x1060/x1070, Quadro FX 4800/5800, Quadro Plex 2200 D2	• double precision floating point support
sm_20	GeForce GTX 4xx/5xx, Tesla x2050/x2070, Quadro 4000/5000/6000	• support of FERMI processor technology • IEEE-754-2008 standard

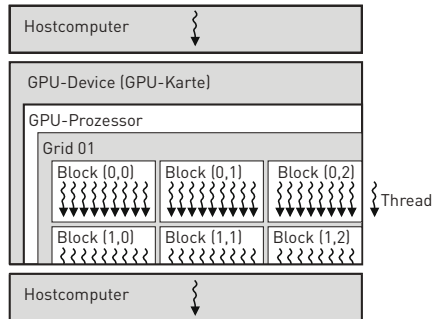
Tabelle 3: GPU-Spezifikationen für ausgewählte Kartenmodelle der Firma NVIDIA [40]

3.3 Überblick über das CUDA-Programmiermodell

Die zur Verwendung von GPU-Technik notwendige Funktionalität wird für NIVIDA-Karten durch das CUDA-Programmiermodell gewährleistet. Im Wesentlichen handelt es sich hierbei um eine Programmierschnittstelle, die die Steuerungsaufgaben für die massiv parallel arbeitenden GPU-Prozessoren beinhaltet. Das CUDA-Programmiermodell ermöglicht es unter anderem, Aufgaben auf die einzelnen Prozessorkerne zu verteilen, Daten zwischen dem GPU- und dem Hostsystem zu kopieren sowie parallel laufende Prozesse zu synchronisieren. In Abb. 11 sind grundlegende Aspekte zur Struktur eines CUDA-Programms schematisch skizziert. Nach dem Kopieren aller erforderlichen Daten aus dem Hauptspeicher des Hostcomputers in den Arbeitsspeicher der GPU-Karte(n) muss der sequenzielle Programmablauf für den GPU-Prozessor aufgespalten werden. Dazu werden die Anweisungen innerhalb von Programmschleifen in Blöcke aufgeteilt, die eine definierte Anzahl von Threads beinhalten. Ein Thread stellt dabei den kleinsten separat ausführbaren Prozess dar. Die Summe aller Threadblöcke, das so genannte „Grid“, ent-

spricht dabei jeweils einer Programmschleife im sequenziellen Quelltext. Die Aufteilung der Schleifen in Threadblöcke und Threads ist vom Programmierer für jede Schleife separat vorzunehmen.

Abb. 11: Schemaskizze zur Struktur eines CUDA-Programms



Alle Threads innerhalb eines Blocks werden durch den GPU-Prozessor im Rahmen der Programmausführung im SIMD-Prinzip (vgl. Anlagenteil A.2) parallel verarbeitet und können bei Bedarf untereinander synchronisiert werden. Die Blöcke selbst werden vor der Berechnung in eine Warteschlange (pipe) eingereiht und – je nach freier Kapazität – vom GPU-Prozessor nacheinander bzw. parallel verarbeitet. Somit ist weder eine Kommunikation zwischen den Blöcken möglich (keine Synchronisation von Threads aus unterschiedlichen Blöcken untereinander) noch kann der Softwareentwickler Einfluss auf die Reihenfolge der Blockverarbeitung nehmen. Diese fehlende Synchronisation auf Blockebene sorgt zwar für eine hohe Rechengeschwindigkeit, legt dem Programmierer jedoch Beschränkungen auf.

Hinsichtlich der Speicherverwaltung können alle Threads innerhalb eines Blocks auf einen gemeinsamen Speicherbereich zugreifen, der sich direkt auf dem Prozessorchip befindet (shared memory). Infolge der direkten Anbindung dieses Speicherbereichs an die arithmetischen Einheiten ist der Zugriff auf diesen gemeinsamen Speicher sehr schnell. Daten zwischen Threads unterschiedlicher Blöcke können dagegen nur über den Arbeitsspeicher der GPU-Karte (GPU-Device) ausgetauscht werden. Da dieser Speicherbereich entsprechend hohe Latenzzeiten aufweist (ca. 400 bis 800 Maschinenzyklen [40]), sollte das Lesen und Schreiben vom bzw. in den Hauptspeicher der GPU-Karte auf das zwingend erforderliche Maß begrenzt werden.

Die oben wiedergegebene Darstellung des CUDA-Programmiermodells macht den massiv parallelen Ansatz der GPU-Karten besonders deutlich. Um die bei Kopiervorgängen von und in den Hauptspeicher der GPU-Karte auftretenden Latenzzeiten zu „verstecken“, ist es erforderlich, mehrere Blöcke je Prozessorkern (Multiprozessor) in die Verarbeitungs-

schlange einzureihen. Wenn beispielsweise vier Blöcke je Multiprozessor angestrebt werden sollen und jeder Block 256 Threads enthält, so müssen einer GPU-Karte mit 16 Multiprozessoren (z.B. GeForce GTX 580) $4 \times 16 \times 256 = 13.384$ Threads gleichzeitig zur Verfügung gestellt werden. Parallelisierungsalgorithmen, die nur für wenige bzw. einige hundert Prozessoren ausgelegt sind, können in der GPU-Technik damit nicht effizient eingesetzt werden (vgl. APD-Methode im Abs. 4.5.4.1). Für weitere ausführliche Details zum CUDA-Programmiermodell wird auf [40] verwiesen.

4 Entwicklung einer GPU-Version für den Fire Dynamics Simulator (FDS)

Im Rahmen dieser Arbeit werden Teile der Rauch- und Brandsimulationssoftware „Fire Dynamics Simulator“ [7] umgeschrieben und an die GPU-Technologie angepasst. Hierdurch wird gegenüber der Basisversion eine Beschleunigung der Rechenleistung um den Faktor 20 bis 30 erreicht (vgl. Abs. 5). Insgesamt wurden dazu über 10.000 Programmzeilen umgeschrieben bzw. neu erstellt. Die neue Version (GPU-Version) der Software liegt dieser Arbeit auf CD-ROM bei.

4.1 Die Software „Fire Dynamics Simulator (FDS)“

Das Programm „Fire Dynamics Simulator (FDS)“ [7] ist eine CFD-Software für auftriebsgetriebene, thermisch expandierte Strömungen, die seitens der Entwickler explizit auf die Simulation von Rauch- und Brandentwicklung in Gebäuden ausgelegt wurde. Neben der Brand- und Rauchsimulation lassen sich jedoch auch allgemeine physikalische Prozesse simulieren (Wärme- und Stofftransport in turbulenten Strömungen, Strahlungswärmetransport in gasförmigen Medien, Wärmeleitung in Feststoffen, Verbrennung von Feststoffen (Pyrolyse), Ausbreitung von Flammen und Brandherden, Einfluss von Sprinkleranlagen und Rauchmeldern im Brandszenario etc.). Die Software wird seit Februar 2000 vom Building and Fire Research Laboratory (BFRL) des National Institute of Standards and Technology (NIST) entwickelt und steht unter <http://fire.nist.gov/fds> kostenfrei zum Download zur Verfügung.

Das strömungsmechanische Modell von FDS verwendet eine modifizierte Form der Navier-Stokes-Gleichungen, wie diese erstmalig von Rehm und Baum [45] für verschwindend kleine Mach-Zahlen hergeleitet wird. Dieser, theoretisch nur für inkompressible Strömungen gültige Ansatz kann ebenfalls in Strömungen verwendet werden, in denen die Fluidgeschwindigkeiten der beteiligten Gase gering ausfallen. Die zu erwartenden Fehler sind in diesem Fall vernachlässigbar klein [46]. Die Turbulenzmodellierung basiert in FDS auf der Large-Eddy-Simulationstechnik (LES) und verwendet zur Ermittlung der turbulenten Viskosität das Smagorinsky-Modell [47]. Ein ebenfalls in FDS implementiertes

Modell zur direkten numerischen Simulation (DNS) hat hinsichtlich der praktischen Verwendung keine Bedeutung, da die hierfür erforderliche Rechenleistung durch die derzeit verfügbare Computertechnik nicht erreicht werden kann. Die Diskretisierung der Erhaltungsgleichungen erfolgt über die Methode der Finite Differenzen unter Verwendung eines expliziten Prädiktor-Korrektor-Verfahrens zweiter Ordnung. Die Poissonsgleichung für den Druck wird jeweils im Prädiktor- und Korrektor-Schritt über einen direkten FFT-Löser (schnelle Fourier-Transformation) gelöst. Eine detaillierte Beschreibung des im Programm enthaltenen strömungsmechanischen Modells bzw. dessen numerische Umsetzung kann [48] entnommen werden.

Im Bereich der Verbrennungsmodellierung hat in FDS im Wesentlichen das implementierte Mischungsbruch-Modell praktische Relevanz. Dazu wird für jede Gaszelle zu jedem Zeitschritt der Massenanteil des unverbrannten Brennstoffs, des Sauerstoffs sowie evtl. vorhandener unvollständig verbrannter Reaktionsprodukte (z. B. Kohlenmonoxid) gespeichert. Die chemische Reaktion wird dann auf der Grundlage vorgegebener Schaltfunktionen in Abhängigkeit der Massenanteile und der Temperatur gesteuert. Mit diesem Modell lassen sich sowohl Brennstoff- als auch oxidator-gesteuerte Verbrennungsprozesse in der Gasphase abbilden, sofern die Modellgrenzen eingehalten werden (keine Stichflammen bzw. Explosionen, keine vorgemischte Verbrennung etc.). Das Ausgasen von Feststoffen während des Brandes wird in FDS über ein separates Pyrolyse-Modell berücksichtigt. Details zu den in FDS enthaltenen Mischungsbruch- bzw. Pyrolyse-Modellen können der technischen Dokumentation [48] entnommen werden.

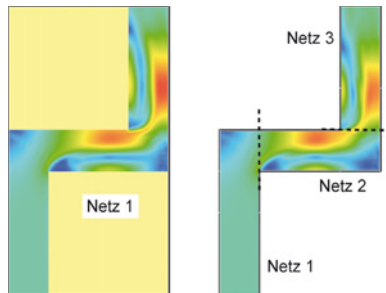
Zur Berücksichtigung des Strahlungswärmetransportes in gasförmigen Medien wird zu jedem Zeitschritt die Transportgleichung (31) für ein oder mehrere Strahlungsbänder separat gelöst. Zum Einsatz kommt hier ein vereinfachter, rechentechnisch effizienter Ansatz, die Finite-Volumen-Methode für den Wärmetransport über Strahlung. Für detaillierte Angaben zum verwendeten Transportmodell wird an dieser Stelle auf den Abschnitt 4.5 verwiesen. Alle genannten Modelle werden auf einem dreidimensionalen rechteckigen Gitter gelöst, welches unterschiedliche Gitterweiten in den einzelnen Rauchrichtungen aufweisen kann. Zusätzlich enthält die Software die Möglichkeit zur Gitterverfeinerung in bestimmten Raumbereichen (embedded meshes).

4.2 Paralleles Rechnen mit FDS (Version 5.4)

Derzeit existieren zwei unterschiedliche Ansätze, mit denen die Software FDS auf Parallelrechnern ausgeführt werden kann. Zum einen besteht die Möglichkeit, ein Lösungsgebiet in mehrere Teilnetze aufzuspalten und jedes einzelne Netz einem separaten Prozessor bzw. Prozessorkern zuzuweisen. In FDS wird dieser Ansatz des „verteilten Rechnens“ über das Message Passing Interface (MPI) realisiert (vgl. Anlagenteil A.6). Der Vorteil dieser Art der Parallelisierung liegt vor allem in der theoretisch kaum begrenzten Prozesso-

ranzahl bzw. Hauptspeichergröße begründet, da die Simulationsrechnung auf einem vernetzten Computercluster mit verteiltem Speicher (vgl. Anlagenteil A.2) ausgeführt werden kann. Als weiteren Vorteil ist die Reduktion der Rechenzeit bzw. des benötigten Hauptspeichers für nicht quaderförmige Netzstrukturen zu nennen. Sofern nur ein einziges Netz verwendet wird, müssen komplexere geometrische Formen durch entsprechende Festkörper modelliert werden (vgl. Abb. 12, links, gelbe Blöcke). Die dadurch resultierenden ungenutzten Bereiche im Hauptspeicher fallen bei der Arbeit mit mehreren Netzen wesentlich geringer aus, da die Form des Lösungsgebiets besser an die vorgegebene Geometrie angepasst werden kann (vgl. Abb. 12, rechts). Unabhängig von einer möglichen parallelen Bearbeitung kann die Verwendung mehrerer Netze daher auch dann eine sinnvolle Ergänzung darstellen, wenn nur ein Prozessor zur Verfügung steht.

Abb. 12: Modellierung einer Rohrströmung mit einem Netz (links) und drei Netzen (rechts), [49]



Hinsichtlich der Mehr-Netz-Routinen von FDS bestehen in der Fachliteratur derzeit Bedenken, da der diesbezüglich in FDS implementierte Algorithmus teilweise fehlerhafte Ergebnisse liefert. Als Kernproblem wird dazu in [49, 50] die Lösung der Poissongleichung für den hydrodynamischen Druck identifiziert. Da es sich hierbei um eine elliptische Differentialgleichung handelt, wirken sich lokale Störungen ohne Zeitverzögerung auf das gesamte Lösungsgebiet aus. In FDS erfolgt die Lösung dieser Gleichung in jedem Zeitschritt durch einen für quaderförmige Lösungsgebiete optimierten FFT-Löser [51], der sich für Ein-Netz-Probleme als äußerst robust und zuverlässig erwiesen hat. Bei der Verwendung von mehreren Netzen wird dieser FFT-Löser in FDS zunächst auf jedes Netz separat angewendet, was zu einer Aufspaltung der physikalischen Verbindung zwischen den Netzen führt. Dem entgegenwirken soll ein Verfahren zur Druckkorrektur (pressure correction), dass einen stetigen Volumenstrom an den Netzgrenzen sicherstellt. Wie in Abb. 13 an einem Beispiel gezeigt wird, werden mit diesem Korrekturverfahren jedoch in Abhängigkeit der Netzanzahl teilweise erhebliche Abweichungen innerhalb der Ergebnisse erzielt. In [49, 50] wird daher ein Lösungsverfahren vorgestellt, mit dem eine effizien-

ente parallele Bearbeitung der Differenzialgleichung zum hydrodynamischen Druck auch ohne Verwendung des FFT-Lösers erreicht werden könnte. Eine Einarbeitung dieses Ansatzes wird ggf. in zukünftige Versionen der Software FDS erfolgen.

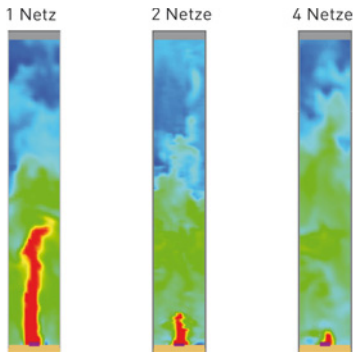


Abb. 13: Vergleich der Temperaturen für einen Brandherd in einem vertikalen Schacht mit einer Unterteilung in ein Netz (links), zwei Netze (mittig) und vier Netze (rechts), FDS Version 5.2, [49]

Unabhängig vom mangelhaften Drucklöser findet die mit der MPI-Version von FDS erreichbare Leistungssteigerung vor allem durch die maximale Geschwindigkeit und Bandbreite des Netzwerkes ihre Begrenzung, da pro Zeitschritt eine Vielzahl von Informationen zwischen den beteiligten Prozessoren ausgetauscht werden müssen. Um hier Abhilfe zu schaffen, gibt es Bestrebungen, eine zusätzliche Parallelisierung der Algorithmen innerhalb eines einzelnen Netzes für Shared-Memory-Systeme vorzunehmen. Insbesondere durch die in letzter Zeit immer häufiger eingesetzten Mehrkernprozessoren ergäbe sich hier für eine Vielzahl von Anwendern die Möglichkeit, eine Beschleunigung über parallele Rechentechnik auch bei Verwendung nur eines einzigen Netzes zu erreichen. In [52, 53] werden erste Ansätze für eine derartige Parallelisierung über die OpenMP-Schnittstelle (vgl. Anlagenteil A.6) des Quelltextes vorgestellt. Da die praktisch erreichbare Beschleunigung bei einer derartigen Parallelisierung erheblich von der Anzahl der verbleibenden sequenziellen Codeanteile abhängig ist (vgl. Anlagenteil A.5), müssen für eine OpenMP-Version von FDS möglichst alle vorhandenen Routinen parallelisiert werden. Ob der daraus resultierende Aufwand gerechtfertigt erscheint, hängt vor allem von der erreichbaren Leistungssteigerung ab. Zumindest bei Mehrkernprozessoren mit einer geringen Anzahl von Prozessoren (z. B. Quad-Core-Prozessoren) ist gegenüber der oben beschriebenen Mehr-Netz-Rechnung mit MPI nicht zwingend eine signifikante Leistungssteigerung zu erwarten, da in der MPI-Version für diesen Fall die Kommunikation zwischen den einzelnen Netzen über den sehr schnellen Bus zwischen CPU und Arbeitsspeicher abgewickelt wird.

Zusammenfassend lässt sich feststellen, dass durch die in FDS enthaltene Mehr-Netz-Implementierung mehr Flexibilität bei der Gestaltung der Geometrie erreicht und eine Rechenbeschleunigung durch Einsatz von Parallelrechnern ermöglicht werden soll („verteiltes Rechnen“ mit dem MPI-Modell). Tatsächlich sind die in FDS derzeit enthaltenen Mehr-Netz-Routinen noch nicht ausreichend ausgereift [49, 50]. Ein erster OpenMP-Ansatz zur Parallelisierung der Software wird in [52, 53] vorgestellt.

4.3 Überblick über die neue GPU-Version

Neben den bisher beschriebenen Ansätzen zur Parallelisierung von FDS sollen im Rahmen dieser Arbeit die Möglichkeiten des Einsatzes der GPU-Technik zur Leistungssteigerung der Software beurteilt werden. Zu diesem Zwecke wurde ca. ein Viertel des bestehenden Quelltextes umgeschrieben und an die GPU-Technologie angepasst. Ziel ist es vor allem aufzuzeigen, inwieweit Brandsimulationsrechnungen unter Verwendung der neuartigen, massiv parallel arbeitenden GPU-Technik effizienter erstellt werden können. Eine vollständige Anpassung des gesamten Rechenkerns kann im Rahmen dieser Arbeit nicht erfolgen, da der Funktionsumfang des Programms erheblich ist (ca. 40.000 Programmzeilen). Sofern der Aufwand-/Nutzen-Aspekt positiv ausfällt, bedarf die endgültige Integration der GPU-Technik in die Software FDS daher zusätzlicher Arbeit. In Tabelle 4

Modell	Parallelisiert Module	Nicht übernommene Module
Hydrodynamisches Modell (hydrodynamic model)	<ul style="list-style-type: none"> • Large Eddy Simulation (LES) in 3D • Ein-Netz-Berechnung • Homogene Netze • Karthesische Koordinaten • Konstanter Hintergrunddruck 	<ul style="list-style-type: none"> • Mehr-Netz-Berechnungen • Netzstreckungen (mesh stretching) • Zylinderkoordinaten • Mehrere Druckzonen • Bakroklone Druckkorrektur • Direct Numerical Simulation (DNS) • Dynamische Smagorinsky Konstante
Verbrennungsmodell (combustion model)	<ul style="list-style-type: none"> • Mixture Fraction Modell ohne CO-Produktion (single-step, instantaneous reaction) 	<ul style="list-style-type: none"> • Mixture Fraction Modell mit CO-Produktion • (two-step reaction, extinction) • Finite-Rate Combustion • Pyrolyse-Modell
Wärmeleitung in festen Stoffen (solid phase model)	<ul style="list-style-type: none"> • Eindimensionaler Wärmetransport in Feststoffen 	
Wärmestrahlung (thermal radiation)	<ul style="list-style-type: none"> • Strahlungstransportmodell für Gase mit konstantem Absorptionskoeffizienten („graues Gas“) 	<ul style="list-style-type: none"> • Modelle für sechs bzw. neun Spektralbänder
Zusatzmodule		<ul style="list-style-type: none"> • Rauchmelder • Evakuierung • Sprinkleranlagen

Tabelle 4: Aus der Basisversion (FDS Version 5.4.0) in die GPU-Version übernommene und parallelisierte Module (Stand: Juni 2011)

sind die in dieser Arbeit parallelisierten Bereiche den noch fehlenden Programmteilen gegenübergestellt. Der Quelltext der neuen GPU-Version ist dieser Dissertation auf CD-ROM beigelegt.

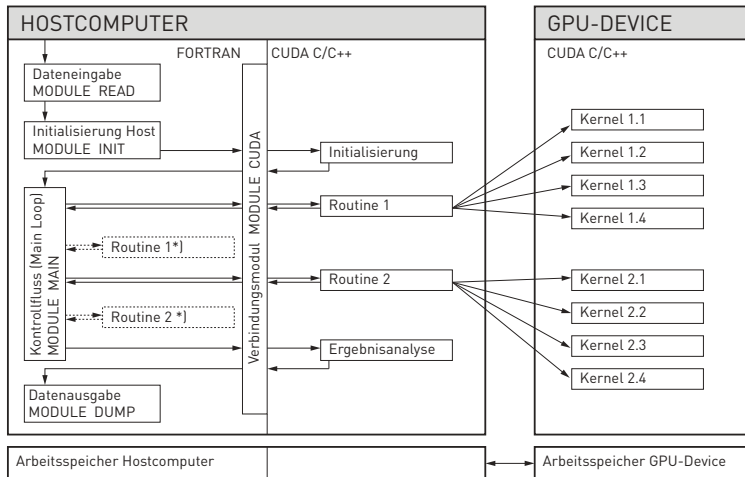
4.3.1 Beschreibung des Programm-Modells

Zu Beginn der Entwicklung der Originalsoftware FDS wurde seitens der Entwickler die Programmiersprache FORTRAN verwendet. Als Konsequenz liegen auch heute noch sämtliche Algorithmen der Software im FORTRAN-Quelltext vor (Version: FORTRAN 90). Da die betrachtete GPU-Technologie von Hause aus über eine auf C/C++ basierende Schnittstelle angesprochen wird, ergibt sich eine wesentliche Hürde, die die Anpassung der Software FDS an die GPU-Technologie zusätzlich erschwert. Zu Beginn dieser Arbeit war die Programmierung von GPU-Software ausschließlich über die auf C basierende Schnittstelle CUDA [40] möglich. Daher bestand zu einer Portierung des Rechenkerns von FORTRAN nach C/C++ keine Alternative. Zwischenzeitlich ist ebenfalls eine auf FORTRAN basierende Schnittstelle für die GPU-Technik der Firma Portland Group verfügbar [54], so dass auch die direkte Umsetzung der parallelen Version in FORTRAN denkbar wäre. Inwieweit die im Rahmen dieser Arbeit erreichte massive Steigerung der Rechenleistung ebenfalls mit dem FORTRAN-Compiler der Portland Group [54] zu erreichen ist und ob sich hier Vereinfachungen bei der Implementierung ergeben, wird im Rahmen dieser Arbeit nicht näher untersucht.

Im Hinblick auf die Erfordernis, Teile der Software nach C/C++ zu übertragen, wurde im Laufe der Bearbeitung das in Abb. 14 dargestellte Programm-Modell entwickelt. Demnach wird die Originalsoftware vom Grunde her komplett beibehalten und um ein Zusatzmodul (MODULE CUDA) ergänzt. Dieses Modul stellt die Verbindung zu den in C/C++ geschriebenen Funktionen dar und bearbeitet die Strukturen im Arbeitsspeicher derart, dass auf diese von C/C++ heraus zugegriffen werden kann. In einem zweiten Schritt wurde das in FORTRAN geschriebene Hauptprogramm um den Aufruf der entsprechenden C/C++-Routinen ergänzt. Die Anzahl dieser Aufrufe ist absichtlich minimiert worden, so dass der Anpassungsbedarf bei einer Versionsänderung von FDS gering ausfällt (vgl. Abs. 4.4.5).

Eine ganze Reihe von neuen Routinen beschäftigen sich zunächst mit der Initialisierung der GPU-Karte (GPU-Device). In den Routinen `cuc_init`, `cuc_initialize_mesh_variables`, `cuc_initialize_poisson_solver`, `cuc_initial_noise` etc. werden Arbeitsspeicher auf der GPU-Karte allokiert, die Anordnung der Variablen im Speicher modifiziert (vgl. Abs. 4.4.2) und bestehende Variableninhalte in die Grafikkarte übertragen. Im Rahmen dieser vorbereitenden Maßnahmen werden zudem Algorithmen zur Freigabe nicht mehr benötigter Speicherbereiche ergänzt, so dass überflüssige Speicherbereiche nach dem Kopiervorgang automatisch freigegeben werden. Durch diese Maßnahme können mehrere GPU-Karten gleichzeitig angesprochen werden, ohne über die entsprechende Menge an

Arbeitsspeicher im Hostcomputer zu verfügen (z. B. gleichzeitige Bearbeitung von Problemstellungen mit drei Karten vom Typ Tesla C2070 ($3 \times 6 \text{ GB} = 18 \text{ GB DRAM}$ mit nur 4 GB Arbeitsspeicher im Hostcomputer)).



*] Aufruf nur im Diagnosemodus

Abb. 14: Programm-Modell für die neue GPU-Version (Schemaskizze)

Im Anschluss an die Initialisierung wird anstelle jeder FORTRAN-Routine die neu entwickelte C/C++ Funktion aufgerufen, die ihrerseits ihre Arbeit an die Grafikkarte über sogenannte Kernel-Funktionen weitergibt. Jeder in der Abb. 14 bezeichnete CUDA-Kernel entspricht dabei einer einzelnen Schleife im Originalprogramm. Mit diesem Ansatz verbleibt der Haupt-Kontrollfluss im Hostcomputer (MODULE MAIN) während alle zeitintensiven arithmetischen Operationen und Speicherzugriffe auf die GPU-Karte ausgelagert werden. Während der gesamten Rechenphase sind lediglich einige wenige „Kontrollbyte“ zwischen dem Speicher der GPU-Karte und dem Hostcomputer zu kopieren (z. B. Kontrolle der Zeitschrittgröße für die CFL-Bedingung etc.), was sich positiv auf die erforderliche Rechenzeit auswirkt.

Für die abschließende Ergebnisausgabe werden die benötigten Werte erneut in den Arbeitsspeicher des Hauptcomputer kopiert. Um den Anforderungen an die GPU-Technik gerecht zu werden, wurde das entsprechende Modul (MODULE DUMP) in wesentlichen Punkten neu verfasst, so dass auch eine zeitschrittweise Ergebnisausgabe ohne massive Einschränkung der Rechenzeit infolge der Kopiervorgänge zwischen GPU-Karte und Hostcomputer erreicht wird (Reduktion erforderlicher Kopiervorgänge zur Maximierung der arithmetischen Intensität, vgl. folgendes Kapitel).

4.3.2 Arithmetic Intensity Problem

Bei einer Parallelisierung von bestehendem Quelltext besteht prinzipiell die Möglichkeit, lediglich die rechenintensiven Teile einer Software umzuschreiben. Sofern die damit verbundene Reduktion der parallelen Effizienz akzeptabel ist, reduziert dieses Vorgehen den Arbeitsaufwand ganz erheblich, da nur Teile der Software parallelisiert werden müssen. Eine derartige „Teilparallelisierung“ wirkt sich jedoch – unabhängig von den verbleibenden sequenziellen Codeanteilen – bei Verwendung von GPU-Technik ganz massiv leistungsmindernd aus. Wie bereits im Abschnitt 3 erwähnt, stellt die Kommunikation zwischen Prozessor und dem Hauptspeicher einen „Flaschenhals“ dar, den man durch entsprechende Maßnahmen zu reduzieren versucht (z. B. Aufnahme eines Cache-Speichers in die CPU etc.). Beim Einsatz von GPU-Technik sind zusätzlich zum Datentransfer zwischen dem Arbeitsspeicher auf der GPU-Karte und dem(n) GPU-Prozessor(en) die Kopiervorgänge zwischen dem Arbeitsspeicher der GPU-Karte und dem Hauptspeicher im Hostcomputer zu berücksichtigen. Das Kopieren von Daten zwischen dem Hostsystem sowie der GPU-Karte ist dabei um ein vielfaches zeitaufwendiger, als das Auslesen bzw. Schreiben von Daten aus dem bzw. in den Arbeitsspeicher der GPU-Karte. Aus dieser Tatsache folgt, dass für jeden Kopiervorgang zwischen der GPU-Karte und dem Hauptspeicher des Hostrechners eine gewisse „Mindestanzahl“ an Rechenoperationen innerhalb des GPU-Systems durchgeführt werden müssen, um überhaupt einen Geschwindigkeitsvorteil zu erreichen. Dieser, als „Arithmetic Intensity Problem“ bekannte Sachverhalt, hat allgemein weitreichende Folgen für die Implementierung von wissenschaftlich-technischer Software auf GPU-Basis und bildet die wesentliche Hürde bei der Anpassung bestehender Software an die GPU-Technik.

Bei näherer Betrachtung stellt sich auch bei der Software FDS heraus, dass mit einer Teilanpassung von Algorithmen an die GPU-Technik nicht die gewünschte Leistungssteigerung erreicht werden kann. Nach einigen diesbezüglich durchgeführten Versuchen ist festzustellen: Das Kopieren von Daten zwischen dem Arbeitsspeicher des GPU-Systems und dem Hauptspeicher im Hostsystems innerhalb eines Zeitschrittes muss, soweit irgend möglich, vermieden werden. Bei der im Rahmen dieser Arbeit durchgeführten Parallelisierung der Software FDS werden daher sämtliche Eingangsparameter zu Beginn der Simulationsrechnung in das GPU-System kopiert (vgl. Abb. 14, MODULE INIT) und erst für die Ergebnisausgabe wieder ausgelesen. Die in dieser Arbeit erreichte massive Steigerung der Rechenleistung liegt zum überwiegenden Teil in diesem Vorgehen begründet.

Wie groß der Zeitbedarf für Kopiervorgänge zwischen der GPU-Karte und dem Hostcomputer tatsächlich ausfällt, kann an dem Beispiel aus Abschnitt 6.4 demonstriert werden. In Abb. 15 ist der Zeitbedarf für die Berechnung der Luftermischung in einen Überlaufplume mit einer CPU im Vergleich zu einer GPU-Karte wiedergegeben. Es ist ersichtlich, dass die Module zur Lösung der physikalischen Problemstellungen DIVG, VELO, MASS,

PRES (Strömungsmechanik), RAD1 (Strahlung), FIRE (Verbrennung) und WALL (Wärmeleitung) ganz massiv beschleunigt werden können. Das Modul zur Ausgabe der Ergebnisse (DUMP) benötigt jedoch in der GPU-Version mehr Zeit, da zu den Schreibvorgängen auf die Festplatte Zeit für den Datentransfer von der GPU-Karte in den Hostcomputer aufzuwenden ist. Entsprechend der Voreinstellung in FDS werden in diesem Modul unabhängig von der Anzahl der Zeitschritte pauschal 1.000 Datensätze in den Hauptspeicher kopiert und von dort aus auf die Festplatte gesichert. Allein auf diese Ergebnisausgabe entfallen in dem Beispiel bei Verwendung der GPU-Version 70 % der gesamten Programmlaufzeit. Dieser Effekt tritt bei kleinen Modellen, kurzen Rechenlaufzeiten und detaillierten Ergebnisausgaben besonders zu Tage.

	CPU (s)	%		CPU (s)	%
MAIN	1523.59	100.00	MAIN	228.00	100.00
DIVG	191.64	12.58	DIVG	2.23	0.98
MASS	398.22	26.14	MASS	5.13	2.25
VELO	475.58	31.21	VELO	6.52	2.86
PRES	55.00	3.61	PRES	7.86	3.45
WALL	47.34	3.11	WALL	1.09	0.48
DUMP	58.52	3.84	DUMP	161.43	70.80
PART	0.00	0.00	PART	0.00	0.00
RADI	264.98	17.39	RADI	30.16	13.23
FIRE	20.73	1.36	FIRE	0.36	0.16
COMM	0.02	0.00	COMM	0.03	0.01
SubTot	1512.03	99.24	SubTot	214.81	94.21

Abb. 15: Laufzeitvergleich der Programm-Module zwischen der CPU-Version (links) und der GPU-Version (rechts), Beispiel aus Abschnitt 6.4

Bei Anwendung der GPU-Version ist es demnach zweckmäßig, die Datenausgabe auf das notwendige Maß zu beschränken. So kann beispielsweise die voreingestellte Ausgabe von 1.000 Datensätzen je Simulationsrechnung reduziert werden, um die Zeitschrittzahl zwischen zwei Datensatzausgaben zu erhöhen (Parameter: NFRAMES). Bei stationären Problemstellungen ist es darüber hinaus sinnvoll, auf die Ausgabe von Daten während der Anlaufphase (Vorwärmphase) vollständig zu verzichten (vgl. neuer Eingabeparameter GPU_START_DUMP im Abs. 4.4.4).

4.3.3 Rechengenauigkeit

Obwohl der überwiegende Teil der CFD-Software derzeit auf doppelter Genauigkeit basiert, ist grundsätzlich auch eine Implementierung von CFD-Verfahren mit einfacher Genauigkeit denkbar. In diesem Fall werden jedoch wesentlich höhere Anforderungen an die Stabilität der eingesetzten numerischen Verfahren gestellt. Ein Vorteil der einfachen Genauigkeit ist neben der höheren Ausführungsgeschwindigkeit vor allem die Halbierung des Speicherbedarfs für die entsprechende Problemstellung.

Insbesondere zu Beginn der Arbeiten an der Software FDS lag die Rechenleistung der verfügbaren Hardware bei Verwendung der einfachen Genauigkeit erheblich über den Beschleunigungen, die bei doppelter Genauigkeit erzielt werden konnten (ca. achtfache Rechenleistung). Daher wurde die neue Software von Beginn an derart modifiziert, dass ohne zusätzlichen Aufwand jeweils eine Version mit einfacher und eine mit doppelter Genauigkeit kompiliert werden kann. Mit Einführung der neuen FERMI-Kartengeneration (vgl. Abs. 3.2) ist der Geschwindigkeitsvorteil der einfachen gegenüber der doppelten Genauigkeit erheblich geschrumpft (ca. 1,5fache Rechenleistung), so dass für die Verwendung der einfachen Genauigkeit nur noch der verminderte Speicherbedarf spricht. In den im Abschnitt 5 verwendeten Beispielen werden die Ergebnisse für beide Genauigkeiten ermittelt und diskutiert. Weitergehende Untersuchungen, insbesondere zu den Ursachen der in FDS vorhandenen numerischen Instabilitäten bei Verwendung der einfachen Genauigkeit, sind nicht Gegenstand dieser Arbeit.

4.3.4 Ein- und Mehrnetzrechnungen

Wie bereits im Abschnitt 4.2 dargestellt wurde, verfügt die Software FDS über die Möglichkeit, Lösungsgebiete in mehrere quaderförmige Netze aufzuspalten. Obwohl die hierzu enthaltenen Algorithmen noch fehlerbehaftet sind, stellt dies derzeit die einzige (ausgereifte) Möglichkeit dar, Rechenläufe in FDS über parallele Datenverarbeitung zu beschleunigen. Der in dieser Arbeit verfolgte GPU-Ansatz basiert auf einer Parallelisierung von Schleifen und Verzweigungen innerhalb der einzelnen Algorithmen, so dass die erreichte Rechenbeschleunigung unabhängig von der Netzanzahl ist. Tatsächlich wurden die in FDS enthaltenen Algorithmen zur Verwendung von mehreren Netzen bis zu diesem Zeitpunkt nicht an die GPU-Technik angepasst, so dass mit der hier entwickelten GPU-Version bisher lediglich Einnetzrechnungen durchgeführt werden können. Da die gesamte Problemstellung vollständig in den Hauptspeicher der GPU-Karte kopiert werden muss, ergibt sich hinsichtlich der maximal zulässigen Problemgröße zur Zeit eine hardwareseitige Beschränkung. Bei Verwendung der GPU-Version darf die Gesamtgröße der Problemstellung eine Größe von 6 GB im Hauptspeicher nicht überschreiten (vgl. Tab. 2, Tesla C2070).

Eine Anpassung der Algorithmen für die Mehrnetzrechnung würde es ermöglichen, mit mehreren GPU-Prozessoren parallel zu rechnen. Der Einsatz von mehreren GPU-Prozessoren in einem System (z. B. Tab. 2, NVIDIA Tesla S2050/70) wäre damit ebenso möglich, wie der Einsatz von mehreren, untereinander vernetzten GPU-Karten. Die theoretisch unbegrenzte Größe der Problemstellung wird in diesem Fall lediglich durch die erforderliche Zeit für Kommunikation über das Netzwerk begrenzt, die mit steigender Netzanzahl überproportional anwächst. Im Rahmen dieser Arbeit wurde auf die Umsetzung der in

FDS enthaltenen Mehrnetzroutinen vor allem deshalb verzichtet, da die dort enthaltenen Algorithmen derzeit noch keine verlässlichen Ergebnisse liefern (vgl. hierzu Erläuterungen im Abs. 4.2).

4.4 Programmtechnische Umsetzung

Mit Blick auf die in dieser Arbeit verfolgte Zielstellung wird auf die ausführliche Erläuterung aller am Quellcode von FDS vorgenommenen Änderungen verzichtet. In diesem Abschnitt werden lediglich einige wesentliche Aspekte erläutert, die für das Verständnis von Bedeutung sind. Für weitergehende Details zu den vorgenommenen Änderungen wird auf die Kommentare im Quelltext verwiesen.

4.4.1 Parallelisierungsschema

Die vorgenommene Parallelisierung im eigentlichen Sinne besteht in der Aufspaltung der vorhandenen Programmschleifen und der Einbindung der Schleifeninhalte in Kernelroutinen (vgl. Abb. 14, S. 48). Während dieses Prozesses sind alle Schleifen der sequenziellen Algorithmen derart aufzuspalten, dass jeder einzelne Rechenkern der GPU-Karte Kenntnis über den von ihm zu verarbeitenden Datensatz hat. So erfolgt beispielsweise der sequenzielle Zugriff auf ein Lösungsgebiet der Dimensionen IBAR x JBAR x KBAR mittels Dreifachschleife (vgl. Abb. 17, S. 54) in der parallelisierten Form über eine Abfolge von Befehlssequenzen, mit der die Zellposition I, J, K aus dem Index desjenigen Blocks/Threads ermittelt werden, der dem jeweiligen Prozessorkern zum Zeitpunkt der Bearbeitung zugeordnet ist:

```
int bIY = blockIdx.y / (DIMK - 1);
int I = blockIdx.x * blockDim.x + TX;
int J = bIY * blockDim.y + TY;
int K = (blockIdx.y - bIY * (DIMK - 1));
if (I >= DIMI - 1 || J >= DIMJ - 1) return;
```

Innerhalb der programmtechnischen Umsetzung wurde für jeden Schleifentyp eine derartige Befehlssequenz entwickelt und als Makro in der Datei „fds_cuda_def.h“ definiert. Dabei sind die einzelnen Makros nicht nur von der sequenziellen Schleife an sich, sondern auch vom Zusammenhang der zu verarbeitenden Daten untereinander abhängig. So ist eine Schleife, innerhalb derer unabhängige Datensätze verarbeitet werden, effizienter und einfacher zu parallelisieren, als eine identische Schleife mit von einander abhängigen Daten. Bis auf einige wenige Bereiche (z. B. FFT-Löser etc.) verfügt die Software FDS ausschließlich über explizite Lösungsansätze, so dass die Parallelisierung ohne erhebliche Änderung an den bestehenden Algorithmen erfolgen kann. Eine Ausnahme hiervon bildet der Transportalgorithmus für die Wärmestrahlung, dessen Parallelisierung separat im Abschnitt 4.5 beschrieben wird.

4.4.2 Neuordnung von Speicherbereichen

Neben der reinen Aufspaltung der Programmschleifen wurden eine Reihe weiterer Änderungen an der Basissoftware vorgenommen. So hängt beispielsweise die effektive Speicherbandbreite (und damit die Geschwindigkeit der Datenübertragung) zwischen dem Prozessor und dem Arbeitsspeicher der Grafikkarte erheblich von der Art des Speicherzugriffes ab. Da die GPU-Kerne über keinen bzw. nur einen minimalen Cache verfügen (vgl. Abb. 9, S. 37), ist es von erheblicher Bedeutung, die Anordnung der Daten im Arbeitsspeicher sowie den Zugriff auf die Datenstrukturen sorgfältig zu planen und umzusetzen. Eine hohe Speicherbandbreite ist nur dann zu erreichen, wenn der Datentransfer zwischen Arbeitsspeicher und GPU in „geordneten Segmenten“ fester Größe erfolgt (coalesced access). Hierfür ist es erforderlich, dass nebeneinander liegende Prozesse (threads) jeweils auf aufeinanderfolgende Adressen im Speicher zugreifen. Sofern die Prozesse willkürlich bzw. unkoordiniert auf den Arbeitsspeicher zugreifen (uncoalesced access), wird der mögliche Datentransfer und damit die Rechengeschwindigkeit massiv reduziert. In Abb. 16 ist das Prinzip des geordneten Speicherzugriffs dargestellt. Weitere Einzelheiten zu den Voraussetzungen des geordneten Speicherzugriffs können der technischen Dokumentation [40] entnommen werden.

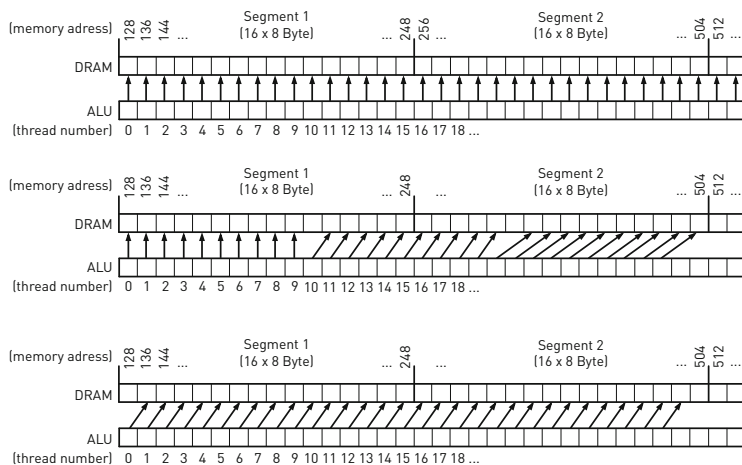


Abb. 16: Geordneter Speicherzugriff (coalesced access) in einer einzigen Speichertransaktion pro Segment (oben) und „willkürlicher“ Speicherzugriff (uncoalesced access) mit jeweils 16 Speichertransaktionen pro Segment (mittig und unten)

Im Rahmen der Portierung des Quelltextes von FDS nach C/CUDA wurde ein Großteil der vorhandenen Datensätze derart neu strukturiert, dass ein geordneter Speicherzugriff möglich ist. Von wesentlicher Bedeutung ist hierbei insbesondere die Anordnung der

Datenstrukturen für die Gaszellen, da diese den weitaus größten Teil der Speicher- und Rechenkapazitäten binden. Für ein Lösungsgebiet mit der Dimension $IBAR$, $JBAR$, $KBAR$ wird in FDS insgesamt Arbeitsspeicher für $N = (IBAR+2) \times (JBAR+2) \times (KBAR+2)$ Zellen benötigt, wobei die Randbedingungen des Lösungsgebietes in den jeweiligen Außenzellen (ghost cells) untergebracht sind (vgl. Abb. 17). Ein Großteil der rechenintensiven Teile des Quelltextes greift auf die Gaszellen über Dreifachschleifen mit der in Abb. 17 rechts dargestellten Form zu.

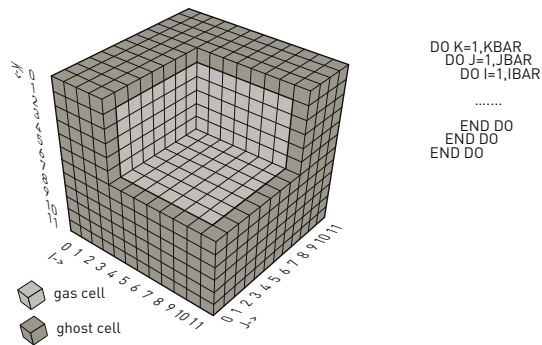


Abb. 17: Lösungsgebiet der Dimension $IBAR \times JBAR \times KBAR = 10 \times 10 \times 10$ Gaszellen (gas cells) mit zusätzlichen Randzellen (ghost cells), sequenzieller Zugriff auf alle Gaszellen über eine Dreifachschleife (rechts)

Um für ein derartiges Konstrukt einen geordneten Speicherzugriff zu gewährleisten, werden alle Daten in Segmenten von jeweils 16 Zellen angeordnet. Darüber hinaus wird sichergestellt, dass der Zugriff auf die Zellen zu jedem Zeitpunkt auch dann segmentweise erfolgt, wenn innerhalb einer Schleife nicht alle Zellen des Lösungsgebietes zu bearbeiten sind. Prozesse (threads), die ungültigen Speicherbereichen zugeordnet sind, werden dann vorübergehend abgeschaltet (idle threads). Für die in Abb. 17 dargestellte Schleife ist das Vorgehen schematisch in Abb. 18 wiedergegeben.

Obwohl bei der verwendeten Systematik nicht in jedem Fall alle Prozesse beschäftigt werden können, übersteigt die gewonnene Zeit durch den geordneten Speicherzugriff den Verlust infolge brach liegender Prozesse bei weitem. Für eine Netzbreite in Höhe von $IBAR = 16 \times N - 2$ ergeben sich keine ungenutzten Speicherbereiche. Andernfalls ist ein geringer Mehrbedarf an Arbeitsspeicher zu erwarten. Die Dimensionen $JBAR$ und $KBAR$ sind – unter Berücksichtigung der [48] dargestellten Empfehlungen für den FFT-Löser – frei wählbar.

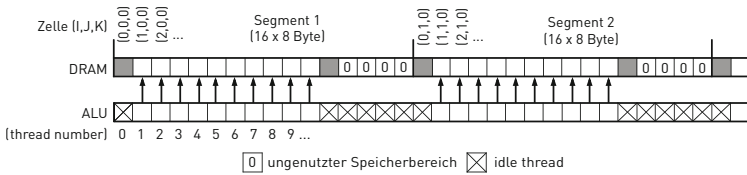


Abb. 18: Aufteilung des Arbeitsspeichers in Blockstrukturen und Zuordnung der einzelnen Zellen zu den Recheneinheiten (threads) für die in Abb. 17 dargestellte Struktur

4.4.3 Parallele Reduktion (parallel reduction)

In einigen Teilbereichen des Quelltextes von FDS werden Algorithmen ausgeführt, deren Parallelisierung eine spezielle Technik, die „Parallele Reduktion“ erfordert. Sofern beispielsweise der Maximalwert eines Feldes (arrays) der Größe „N“ zu ermitteln ist, kann dies in sequenzieller Form über eine einfache Schleife erfolgen:

```
double a = Array[0];
for (int i=1; i < N; i++) a = fmax (a, Array[i]);
```

Eine parallele Bearbeitung dieser Problemstellung unter Verwendung von GPU-Technik gestaltet sich ungleich schwieriger, da eine möglichst große Anzahl von Prozessen beschäftigt werden soll. Eine effiziente Möglichkeit besteht in der Zerlegung des Feldes in Teilbereiche, die jeweils einem Threadblock zugewiesen und anschließend gleichzeitig bearbeitet werden. Innerhalb eines Threadblocks erfolgt die Ermittlung des blockspezifischen Maximalwerts über den „shared memory“, einen chipinternen Speicherbereich, auf den alle Prozesse des jeweiligen Blocks gemeinsam zugreifen können (vgl. Abb. 19, links). Das gesamte Feld wird durch rekursives Aufrufen des Reduktionskernels solange reduziert, bis letztlich nur noch der Maximalwert des einzig verbliebenen Threadblocks zu ermitteln ist (vgl. Abb. 19, rechts, Schritt 2).

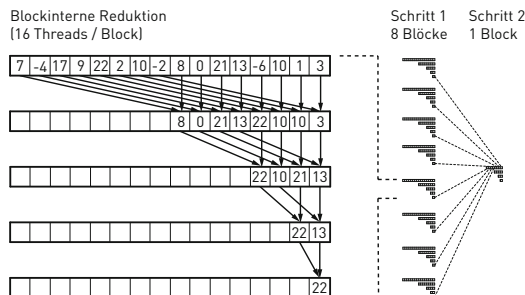


Abb. 19: Ermittlung des Maximalwertes innerhalb eines Blocks im Shared Memory (links) und rekursiver Aufruf des Reduktions-Kernels zur Ermittlung des Maximalwertes im gesamten Feld (rechts)

Für die Implementierung eines derartigen Algorithmus' in die neue GPU-Version von FDS wurden die in [55] enthaltenen Beispiele zur parallelen Reduktion an die Erfordernisse der jeweiligen Problemstellung angepasst. Der derzeitige Stand aller Routinen zur parallelen Reduktion findet sich in der Datei „reduction.cu“ auf der beiliegenden CD-ROM.

4.4.4 Dokumentation neuer Eingabeparameter

Im Rahmen der Weiterentwicklung der bestehenden Version von FDS wurde der Quellcode um zwei neue Eingabeparameter erweitert. Die in Tabelle 5 aufgeführten Parameter zur Steuerung ergänzen damit die in [56], Abschnitt 13 enthaltene tabellarische Zusammenfassung aller Eingabeparameter für die Basisversion. Über den neuen Parameter GPU_DEV ist die Auswahl einer bestimmten GPU-Prozessorkarte möglich, sofern im System mehrere GPU-Karten enthalten sind. Der Parameter GPU_START_DUMP ermöglicht es, die Ergebnisausgabe erst nach einer vorgegebenen Anzahl von Sekunden beginnen zu lassen. In Abschnitt 4.3.2 wird gezeigt, dass sich die Kopiervorgänge zwischen GPU-Prozessorkarte und dem Arbeitsspeicher negativ auf die Gesamtleistung auswirken. Da bei einigen Problemstellungen zu Beginn der Simulation keine Ergebnisse benötigt werden (z. B. Anlaufphase für ein stationäres Problem), kann die erforderliche Rechenzeit über diesen Parameter (teilweise erheblich) reduziert werden.

&CUDA (GPU-Parameter)		
GPU_DEV	Integer	Index der zu verwendenden GPU-Karte
GPU_START_DUMP	Real	Anzahl der Sekunden vor dem Beginn der Ergebnisausgabe

Tabelle 5: Zusätzliche Eingabeparameter für die GPU-Version (Ergänzung zu [56], Abschnitt 13)

4.4.5 Versionsänderungen

Die Software FDS unterliegt einer ständigen Fortentwicklung, so dass auch die im Rahmen dieser Arbeit erstellte GPU-Version immer neu angepasst werden muss. Die dieser Arbeit beigelegte Version bezieht sich auf die Software FDS Version 5.4. Für eine Anpassung an aktualisierte FDS-Versionen sind prinzipiell alle im FORTRAN-Quelltext mit [„!CUDA“] gekennzeichneten Funktionsaufrufe in die neue Version zu übertragen. Im Anschluss müssen die in FDS erfolgten Änderungen in die GPU-Version eingepflegt werden. Zur Überprüfung der korrekten Funktionsweise sollten abschließend Beispiele zur Verifikation gerechnet werden (vgl. Abs. 5).

4.5 Parallelisierung der Algorithmen zum Strahlungswärmetransport

Bereits in Abschnitt 4.4.1 wird darauf verwiesen, dass der in FDS enthaltene Transportalgorithmus nicht ohne erhebliche Änderungen parallelisiert werden kann. Ursächlich hierfür ist die Wellenfrontbewegung der Strahlungspartikel durch den Raum, die in der eingesetzten „Finite-Volumen-Methode (FV-Methode)“ eine maßgebliche Rolle spielt. In den folgenden Abschnitten werden Möglichkeiten zur Parallelisierung des Transportproblems in FDS aufgezeigt und die für die GPU-Version neu entwickelten Algorithmen näher erläutert.

4.5.1 Grundlagen des Wärmetransports über Strahlung

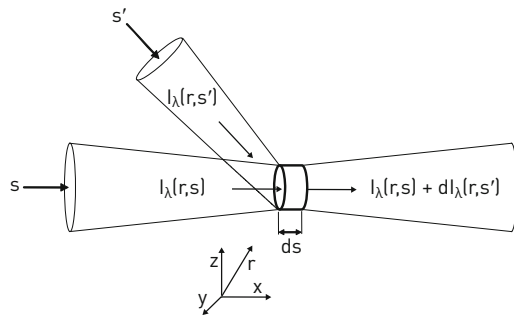
Versuche zeigen, dass bei üblichen Brandszenarien in Gebäudebränden etwa 20 bis 40 % der freigesetzten Energie direkt in Strahlung umgewandelt wird [57]. Durch diese Wärmestrahlung erhitzen sich auch weiter entfernte, außerhalb der heißen Brandgase befindlichen Objekte, was zum Ausgasen zusätzlicher brennbarer Stoffe führen kann (Pyrolyse). Sofern das Verhalten eines Brandes durch eine Simulationsrechnung verlässlich abgeschätzt werden soll, muss die eingesetzte Brandsimulationssoftware den Transport von Wärmestrahlung daher zwingend berücksichtigen. Bei diesem, als „Wärmetransport über Strahlung in gasförmigen Medien“ bezeichneten Phänomen, handelt es sich im Wesentlichen um den Transportmechanismus von Photonen in einem Gas, der durch die Interaktion (Absorption, Emission und Streuung) zwischen den Photonen und den Gasparkeln gekennzeichnet ist. Obwohl die dem Wärmetransport über Strahlung zugrunde liegenden Mechanismen zwischenzeitlich weitgehend bekannt sind und durch Differenzialgleichungen beschrieben werden können, stellt die Lösung dieser Gleichungen immer noch ein aufwendiges Unterfangen dar. Mit Ausnahme von einfachen Spezialfällen können derartige Problemstellungen daher ausschließlich auf numerischem Wege bearbeitet werden. Durch die Kopplung mit dem Strömungslöser und der Notwendigkeit, den Strahlungswärmetransport für jeden Zeitschritt erneut bestimmen zu müssen, stellt sich das Transportproblem von Wärmestrahlung im Bereich der Rauch- und Brandsimulation darüber hinaus als äußerst rechenintensiv heraus.

Die Basis für die Beschreibung des Strahlungswärmetransports in gasförmigen Medien ist die Transportgleichung. In dieser Gleichung wird die Strahlungsintensität I_λ in Abhängigkeit der Wellenlänge λ sowie der Raumrichtung s (spektrale, gerichtete Strahlungsintensität) bilanziert, wobei r als beliebige Position im gewählten Koordinatensystem aufzufassen ist:

$$s \cdot \nabla I_\lambda = - \left[\kappa(r, \lambda) + \sigma_s(r, \lambda) \right] I_\lambda(r, \lambda) + B(r, \lambda) + \frac{\sigma_s(r, \lambda)}{4\pi} \int_{4\pi} \phi(s, s') I_\lambda(r, s') ds' \quad (31)$$

Der Term auf der linken Seite repräsentiert die Änderung der Strahlungsintensität I_λ an der Koordinate r für die betrachtete Raumrichtung s (vgl. Abb. 20). Der erste Term auf der rechten Seite berücksichtigt die Abnahme der Intensität I_λ infolge Absorption (spektraler gerichteter Absorptionsgrad κ) sowie Streuung an Partikeln im Raum (spektraler gerichteter Streuungskoeffizient σ_s). Der zweite Term auf der rechten Seite beschreibt die Zunahme der Strahlungsintensität I_λ infolge Strahlungsemission von Gasteilchen (Schwarzkörper-Strahlung). Schließlich berücksichtigt der letzte Term die Zunahme der Strahlungsintensität infolge einfallender Streustrahlung aus allen möglichen Raumrichtungen s' , aufsummiert über die Kugeloberfläche 4π . Die Funktion Φ stellt dabei den richtungsabhängigen Grad der Einstreuung dar. (Für isotrope Einstreuung gilt: $\Phi = 1,0$). Für eine ausführliche Herleitung der o. g. Gleichung wird auf [58, 59, 60] verwiesen.

Abb. 20: Bilanzierung der spektralen gerichteten Strahlungsintensität I_λ über die Weglänge ds in Richtung s (Prinzipiskizze)



Die Gleichung (31) stellt eine Integrodifferenzialgleichung dar, die sich nicht zuletzt aufgrund ihrer Kopplung mit den übrigen Differenzialgleichungen aus der Strömungsmechanik bzw. der Verbrennung nur numerisch und mit hohem Aufwand lösen lässt. Aus Gründen der Recheneffizienz werden daher häufig Vereinfachungen vorgenommen, die eine Bearbeitung des Problems mit praktisch zu bewältigendem Aufwand ermöglichen. Oft wird dazu das gasförmige Medium als nicht-streuendes graues Gas betrachtet. Damit vereinfacht sich die Gleichung (31) zu

$$s \cdot \nabla I(r, s) = \kappa(r) [I_b(r) - I(r, s)], \quad (32)$$

wobei I_b die Schwarzkörper-Strahlung nach dem Stefan-Boltzmann-Gesetz darstellt:

$$I_b(r) = \sigma T(r)^4 / \pi \quad (33)$$

Obwohl diese Näherung in Bereichen mit dichtem Rauchgas infolge des gleichförmigen Absorptionsspektrums von Rauch durchaus gute Ergebnisse liefert, wird die Strahlungsabsorption in Medien mit geringer optischer Dichte durch die Annahme eines grauen

Gases überschätzt. Für diesen Fall bietet sich die Möglichkeit an, die Gleichung (32) für eine definierte Anzahl spektraler Bänder separat zu lösen:

$$s \cdot \nabla I_n(r, s) = \kappa_n(r) [I_{b,n}(r) - I_n(r, s)], n = 1..N \quad (34)$$

$$I_{b,n}(r) = F_n(\lambda_{\min}, \lambda_{\max}) \sigma T(r)^4 / \pi \quad (35)$$

Die Gesamtintensität in einer Strahlungsrichtung ergibt sich dann aus der Summe der Einzelintensitäten aller Strahlungsbänder:

$$I(r, s) = \sum_{n=1}^{n=N} I_n(r, s) \quad (36)$$

Die Vorfaktoren F_n sind von der maximalen bzw. minimalen Wellenlänge eines Strahlungsbandes abhängig und werden beispielsweise in [61] näher erläutert.

4.5.2 Numerische Lösung der Transportgleichung mit der Finite-Volumen-Methode (FV-Methode)

Um die Transportgleichung (31) numerisch zu lösen, wurden in den letzten 40 Jahren diverse Näherungsverfahren entwickelt. Während ein Teil dieser Methoden unter Einsatz von hohem Rechenaufwand überaus akkurate Ergebnisse liefert (z. B. Monte-Carlo-Methode), stellen andere Methoden angenäherte Ergebnisse mit vergleichsweise geringem Aufwand zur Verfügung (z. B. Methode der Spherical-Harmonics P_1). Die in dieser Arbeit betrachtete Software FDS (vgl. Abs. 4) berücksichtigt den Strahlungswärmetransport in gasförmigen Medien über einen vereinfachten und rechentechnisch effizienten Ansatz, der „Finite-Volumen-Methode für den Wärmetransport über Strahlung (FV-Methode)“. Dieses numerische Verfahren wurde erstmalig von Raithby und Chui in [62] für orthogonale Volumenelemente vorgestellt. In den weiteren Jahren wurde die Methode auf Volumenelemente mit regelmäßiger nicht-orthogonaler Geometrie (vgl. [63, 64]) und zylindrischer Form (vgl. [65, 66, 67, 68]) erweitert. Unstrukturierte Netze beliebiger Form werden in [69] und [70] behandelt. Einen guten Überblick über die FV-Methode bzw. eine ausführliche Diskussion findet sich darüber hinaus in [71, 72, 73].

Ausgangspunkt der FV-Methode ist die Integration der Transportgleichung über das gesamte Lösungsgebiet. Der Übersichtlichkeit halber wird hier von einem nicht streuenden, grauen Gas ausgegangen. Eine entsprechend verallgemeinerte Herleitung für streuende Gase ist in [71] enthalten. Aus Gleichung (32) ergibt sich damit:

$$\int_{\Omega} \int_V (s \cdot \nabla) I(r, s) dV d\Omega = \int_{\Omega} \int_V \kappa(r) [I_b(r) - I(r, s)] dV d\Omega \quad (37)$$

Hierbei steht dV für ein Volumenintegral über die räumlichen Koordinaten ($dV = dx \cdot dy \cdot dz$) und $d\Omega$ für ein Oberflächenintegral auf der Einheitskugel ($d\Omega = d\Phi \cdot d\theta$). Für die numerische Lösung der Gleichung (37) muss das Lösungsgebiet in diskrete Kontrollelemente (Kontrollvolumen V_P und Kontrollwinkel Ω^l) zerlegt werden (vgl. Abb. 21). Zweckmäßigerweise können für die räumliche Diskretisierung hier die Kontrollvolumen aus dem Strömungslöser verwendet werden (soweit vorhanden).

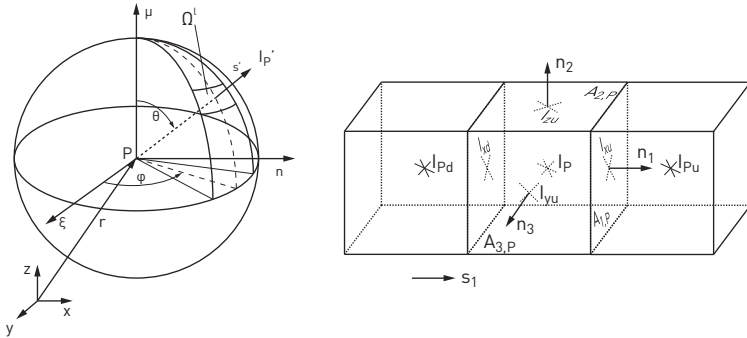


Abb. 21: Aufteilung des Lösungsgebiets in Kontrollwinkel (links) und Kontrollvolumen (rechts)

Wird angenommen, dass die Strahlungsintensität innerhalb eines Kontrollvolumens V_P und innerhalb eines Kontrollwinkels Ω^l konstant ist, so entfallen die Integrale auf der rechten Seite der Gleichung. Bei Anwendung des Divergenztheorems ergibt sich die Gleichung (37) zu:

$$\int_{\Omega^l} \int_{A_P} I(r, s) (s \cdot n) dA_P d\Omega = \kappa(r) [I_{b,p}^l(r) - I_P^l(r, s)] V_P \Omega^l \quad (38)$$

Hierbei stellt A_P die Oberfläche des Kontrollvolumens dar, wobei der Normalenvektor n an jedem Punkt senkrecht zur betrachteten Oberfläche steht. Im Rahmen der numerischen Lösung ist damit für jeden Kontrollwinkel eines Kontrollvolumens die gerichtete Strahlungsintensität I_P^l in Abhängigkeit der Intensitäten in den benachbarten Kontrollvolumen sowie der Absorption bzw. Emission im Kontrollvolumen selbst zu bestimmen. Die Gesamt-Strahlungsintensität in einem Kontrollvolumen ergibt sich dann aus der Summe der Einzelintensitäten über alle Richtungen bzw. Kontrollwinkel. Unter Annahme einer definierten Anzahl von Grenzflächen zwischen den einzelnen Kontrollvolumen kann die Gleichung (38) auch folgendermaßen geschrieben werden:

$$\sum_j A_{j,P} I_{j,P}^l \int_{\Omega^l} (s \cdot n) d\Omega = \kappa(r) [I_{b,p}^l(r) - I_P^l(r, s)] V_P \Omega^l \quad (39)$$

Dabei stellt j den Index der jeweiligen Grenzfläche dar. (vgl. Abb. 21, rechts: A_1 ; A_2 ; A_3). Das Integral auf der linken Seite der Gleichung kann auch als Projektion des Kontrollwinkels auf die betrachtete Grenzfläche angesehen werden und stellt damit ein Maß für den Anteil der Intensität I_p dar, die in Richtung n zum benachbarten Kontrollvolumen übertragen wird. Infolge der darin enthaltenen Geometrieabhängigkeit sind diese „Projektionen“ für jedes Kontrollvolumen separat zu ermitteln (in FDS vor Beginn des Rechenlaufes). Da die Kontrollwinkel auf eine allseitig geschlossene Fläche projiziert werden, ergibt sich für die FV-Methode – im Gegensatz zu anderen Methoden – der Vorteil der Energieerhaltung im Gesamtsystem. Sofern in einer Software ausschließlich orthogonale Kontrollvolumen betrachtet werden (wie dies in FDS der Fall ist), ergeben sich weitergehende Vereinfachungen für die Berechnung. Diese werden u. a. ausführlich in [74] erläutert.

Um die Gleichung (39) auf numerischem Wege lösen zu können, muss zusätzlich die räumliche Beziehung zwischen der im Zentrum und den auf den Grenzflächen des Kontrollvolumens liegenden Intensitäten diskretisiert werden (vgl. Abb. 21, rechts). Dabei stellen I_{xd}^1 , I_{yd}^1 , und I_{zd}^1 die aus Richtung des Kontrollwinkels s^1 kommenden (downstream) und I_{xu}^1 , I_{yu}^1 , und I_{zu}^1 die in Richtung des Kontrollwinkels verlaufenden (upstream) Grenzflächenintensitäten dar. Eine einfache Möglichkeit zur Diskretisierung besteht in der Anwendung eines gewichteten Finite-Differenzen-Schemas [75], wobei f der Wichtungsfaktor mit den Grenzen $0 \leq f \leq 1$ darstellt:

$$\begin{aligned} I_p^1 &= f I_{xd}^1 + (1-f) I_{xu}^1 \\ I_p^1 &= f I_{yd}^1 + (1-f) I_{yu}^1 \\ I_p^1 &= f I_{zd}^1 + (1-f) I_{zu}^1 \end{aligned} \quad (40)$$

Für den Spezialfall $f = 1,0$ erhält man das Upwind-Differencing-Schema erster Fehlerordnung, für $f = 0,5$ das Central-Differencing-Schema zweiter Fehlerordnung. Weitere Diskretisierungsverfahren für die FV-Methode werden beispielsweise in [62] und [72] diskutiert.

4.5.3 Modellierung des Strahlungswärmetransports in FDS

Da es für baupraktische Anwendungsfälle im Allgemeinen ausreichend ist, von einem gasförmigen Medium mit konstantem Absorptions- bzw. Emissionskoeffizienten auszugehen (Modellierung des dichten Rauchgases als „graues Gas“), wird in FDS standardmäßig die Gleichung (32) numerisch gelöst. Für Problemstellungen mit optisch dünnen Verbrennungsgasen, bei denen im Vergleich zum Wasser- und CO_2 -Gehalt wenig Rußpartikel vorliegen, führt die Annahme eines grauen Gases jedoch zu verfälschten Ergebnissen. Für diesen Fall kann der Anwender von FDS auf zwei zusätzliche Modelle mit einer definierten Anzahl von Spektralbändern zurückgreifen, bei denen die Gleichungen (34) bis (36) zur Ermittlung der Strahlungsintensitäten verwendet werden. Die mittleren Absorptionskoeffizienten für die einzelnen Spektralbänder werden in FDS generell vor

Beginn der Simulationsrechnung in Abhängigkeit von Temperatur und Stoffanteil nach [76] ermittelt und in einer Tabelle gespeichert. Zur Programmaufzeit wird dann auf die vorab gespeicherten Werte nach Bedarf zurückgegriffen.

Unabhängig von der Anzahl der verwendeten Spektralbänder wird in FDS die Strahlungsenergiefreisetzung in den Kontrollvolumen mit Verbrennungsprozessen gesondert betrachtet. Da die Raumzellen in Bezug auf die Verbrennungsskalen große Abmaße aufweisen, würde ohne besondere Behandlung die tatsächlich vorhandene, lokale Verbrennungstemperatur auf die jeweilige Raumzelle verschmiert und damit verringert ausgegeben. Da die Temperatur im Term der Strahlungsemission in der vierten Potenz eingeht, würde in diesem Fall die freiwerdende Strahlungsenergie beim Einsetzen der Zelltemperaturen in Gleichung (35) erheblich unterschätzt werden. Aus diesem Grund wird in FDS der Bereich der Flammenzone separat behandelt, d. h. die insgesamt in der Flammenzone emittierte Strahlungsintensität $\kappa \cdot I_b$ wird nach unten hin auf einen prozentualen Anteil ξ_r der Gesamtenergiefreisetzung Q_{tot} begrenzt:

$$\kappa I_b = \max\left(\frac{\xi_r Q_{\text{tot}}}{4\pi}, \frac{\kappa \sigma T^4}{4\pi}\right) \quad (41)$$

Die Festlegung der im Verbrennungsprozess freiwerdenden Strahlungsenergie obliegt dabei dem Anwender, d. h. der Ingenieur muss den prozentualen Anteil der freiwerdenden Strahlungsenergie zu Beginn der Simulationsrechnung vorgeben. Hieraus resultiert ein relativ hoher Unsicherheitsfaktor bei den Ergebnissen. Weitere Angaben zu dem in FDS verwendeten Strahlungsmodell – insbesondere zur Behandlung der Randbedingungen – können der technischen Dokumentation [48] entnommen werden.

Zur Lösung der Transportgleichung wird in FDS die im Abschnitt 4.5.2 beschriebene Finite-Volumen-Methode für den Strahlungswärmetransport verwendet. Für die räumliche Diskretisierung wird dabei auf ein explizites Upwind-Differencing-Schema erster Ordnung (vgl. Gl. (40)) zurückgegriffen, so dass eine Iteration lediglich im Falle von Strahlungsflexion an den Wänden erforderlich ist. Der Lösungsalgorithmus in FDS fällt durch das explizite Schema vergleichsweise schnell aus. Betrachtet man ein System mit einem einzigen Prozessor, so können die erforderlichen Schritte zur Lösung der Transportgleichung über die FV-Methode wie folgt zusammengefasst werden:

1. Vorbereiten der Ränder und Löschen des Iterationszählers
2. Berechnen der Intensitäten für jedes Kontrollvolumen, jeden Kontrollwinkel und jedes Spektralband separat wie folgt:
 - Ermitteln der einfallenden Strahlungsintensitäten I'_{xd} , I'_{yd} und I'_{zd} auf den drei Grenzflächen aus Richtung des Kontrollwinkels (vgl. Abb. 22)

- Ermitteln der Strahlungsintensität I_p im Inneren des Kontrollvolumens unter Berücksichtigung von Strahlungsquellen oder -senken (Absorption, Streuung, Emission)
 - Ermittlung der ausgehenden Strahlungsintensitäten I_{xu} , I_{yu} und I_{zu} auf den drei Grenzflächen in Richtung des Kontrollwinkels (vgl. Abb. 22)
3. Aktualisieren der Randbedingungen (z.B. Strahlungsabsorption und -reflexion an den Wänden) durch Aufsummieren der einfallenden Strahlung über alle Kontrollwinkel und Spektralbänder
 4. Wiederholen ab Schritt (2), sofern das Konvergenzkriterium nicht erreicht wurde²

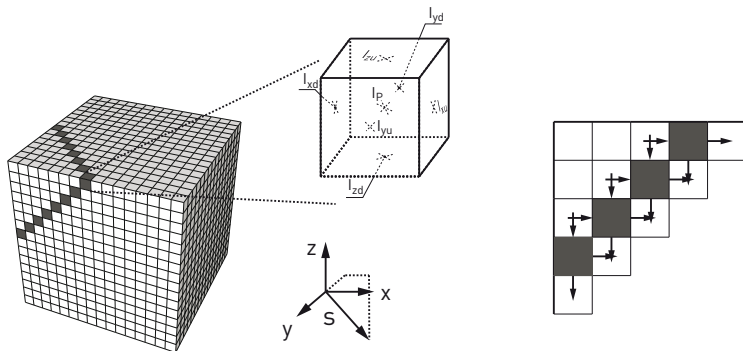


Abb. 22: Wellenfront bei der Ermittlung der Intensitäten für einen Kontrollwinkel (Richtung s_1) für ein orthogonales Gitter

Die Lösung der Transportgleichung erfolgt in FDS auf dem gleichen Gitter, auf dem auch die Differentialgleichungen zur Strömungsmechanik gelöst werden. Auch die Zeitschrittgröße wird aus dem strömungsmechanischen Programmteil ohne Änderung übernommen. Da die Zeitschritte hinsichtlich der Änderung des Strahlungszustandes im System damit tendenziell zu klein ausfallen, werden lediglich zu Beginn der Berechnung die Intensitäten für alle Kontrollwinkel ermittelt. In allen übrigen Zeitschritten wird aus Effizienzgründen nur ein Teil der Kontrollwinkel aktualisiert (standardmäßig in Version 5.4 ein Fünftel aller Kontrollwinkel je Zeitschritt). Dem dadurch gestiegenen Leistungszuwachs steht jedoch ein erhöhter Speicherbedarf gegenüber, da die Intensitäten in den Gaszellen und Wandoberflächen aus den vorangegangenen Zeitschritten zur Summenbildung vorgehalten werden müssen. Insbesondere bei der Verwendung von mehreren Strahlungsbändern kann sich dieser erhöhte Speicherbedarf als sehr störend auswirken.

² Die in Schritt (4) aufgeführte Iteration ist nur dann erforderlich, wenn die Reflexion von Strahlung an den Wänden nicht vernachlässigt werden kann. Zum derzeitigen Zeitpunkt wird in FDS keine Strahlungsreflexion berücksichtigt.

Für Mehrnetzrechnungen (vgl. Abs. 4.2) werden nach Beendigung jedes Zeitschrittes die Strahlungsintensitäten an den Randzellen der angrenzenden Netze ausgetauscht (vgl. „Kommunikationsphase“ im DDP-Verfahren in Abs. 4.5.4.2). Eine iterative Berechnung innerhalb eines Zeitschrittes erfolgt jedoch nicht, so dass sich bei vielen Netzen ein Strahlungsanstieg/-abfall in Abhängigkeit der strömungsmechanisch bedingten Zeitschrittweite erst allmählich im gesamten Lösungsgebiet fortpflanzt. Eine weitergehende Beschreibung der numerischen Lösungsalgorithmen der Transportgleichung findet sich unter anderem in der technischen Dokumentation [48].

4.5.4 Möglichkeiten zur Parallelisierung der Finite-Volumen-Methode

Die Parallelisierung der in FDS enthaltenen Transportalgorithmen kann über mehrere Ansätze erfolgen. Aufgrund des Charakters der GPU-Technik ist jedoch nicht mit jeder Methode die gleiche Effizienz zu erreichen, so dass vor der Entwicklung eines Algorithmus eine Vorauswahl zu treffen ist.

4.5.4.1 Angular Decomposition Parallelization (ADP-Verfahren)

Bei der in FDS enthaltenen FV-Methode müssen je Iterationsschritt $N \times L \times B$ Intensitäten ermittelt werden, wobei N die Anzahl der Kontrollvolumen, L die Anzahl der Kontrollwinkel und B die Anzahl der Spektralbänder darstellt. Da die Intensitäten unterschiedlicher Kontrollwinkel und Spektralbänder nicht voneinander abhängen, liegt es Nahe, jedem Prozessor/Prozess ein oder mehrere Kontrollwinkel/Spektralbänder zuzuweisen, für welche(n) er die erforderlichen Rechenschritte durchführt. Diese vergleichsweise einfache Möglichkeit der Parallelisierung wird als Angular Decomposition Parallelization (ADP) bezeichnet. Neben der einfachen Implementierung liegt der Vorteil dieser Methode vor allem in der hohen parallelen Effizienz (vgl. Anlagenteil A, Abs. A.5). Von Nachteil ist dem gegenüber, dass jeder beteiligte Prozessor zu jedem Zeitpunkt Zugriff auf das gesamte Lösungsgebiet haben muss, was bei Systemen mit verteiltem Speicher praktisch nicht zu realisieren ist. Ebenfalls von Nachteil ist, dass die Anzahl der maximal einsetzbaren Prozessoren auf die Anzahl der Kontrollwinkel begrenzt ist. Die Lösung der Transportgleichung mittels ADP-Methode wird für graue Gase und orthogonale Volumenelemente erstmalig von Gonçalves und Coelho in [77] und von Burns und Christon in [78] eingehend untersucht. Die in [77] für die Diskrete-Ordinaten-Methode vorgestellten Ergebnisse erweitern die Autoren in [79] auf die Finite-Volumen-Methode.

Insbesondere aufgrund der Begrenzung der Prozessoranzahl muss der Einsatz der ADP-Methode in der GPU-Technik als wenig effizient eingestuft werden. Wie im Abschnitt 3.3 erläutert, werden bei der massiv parallel arbeitenden GPU-Prozessortechnik zehntausende unabhängig voneinander arbeitende Prozesse (threads) benötigt, so dass die Kontrollwinkel- und Spektralbandanzahl nur selten als ausreichend erachtet werden kann.

Zusätzlich verlangt die GPU-Technik eine möglichst hohe arithmetische Intensität, d. h. auf jeden Datentransfer zwischen Hauptspeicher und GPU-Prozessor sollte eine möglichst große Anzahl von arithmetischen Operationen folgen. Diese Anforderung wird durch die ADP-Methode vom Grundsatz her nicht erfüllt.

4.5.4.2 Spatial Domain Decomposition Parallelization (DDP-Verfahren)

Alternativ zur ADP-Methode ist es denkbar, eine räumliche Parallelisierung der Berechnung durchzuführen. Bei dieser so genannten Spatial Domain Decomposition Parallelization (DDP) wird das Lösungsgebiet in Teilgebiete aufgespalten, wobei jedem Prozessor eines der Teilgebiete zugewiesen wird (vgl. Abb. 23). Im ersten Rechenschritt ermitteln alle beteiligten Prozessoren – ausgehend von den Rändern des Teilgebiets (vgl. Abb. 23, Kreismarkierung) – alle Intensitäten in Richtung des betrachteten Kontrollwinkels parallel. Im Anschluss werden die Ergebnisse im Randbereich (vgl. Abb. 23, Kreuzmarkierung) in einer „Kommunikationsphase“ in Richtung des Kontrollwinkels an die benachbarten Prozessoren weitergegeben und der nächstfolgende Rechenschritt gestartet. Für diesen iterativen Berechnungsvorgang sind zum einen die Startwerte vor dem ersten Rechenschritt für alle Teilgebieten geeignet abzuschätzen und zum anderen ist ein Grenzkriterium festzulegen, bei dem die Iteration abgebrochen wird. Mit allen übrigen Kontrollwinkeln wird entsprechend ihrer Richtung analog verfahren.

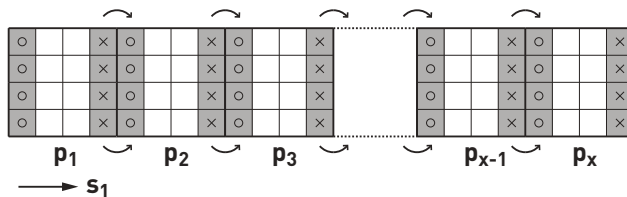


Abb. 23: Aufteilung des Lösungsgebietes in Teilgebiete (Prozessor p_1, p_2, \dots, p_x), Datenaustausch nach jedem Iterationsschritt für die Richtung des Kontrollwinkels s_1

Ein DDP-Algorithmus für die FV-Methode auf unstrukturierten Netzen wird in [80] vorgestellt. Zur Steigerung der Effizienz bzw. zur Verbesserung der Konvergenz entwickeln Ben-Zvi und Kribus in [81] und [82] ein kombiniertes ADP-DDP-Verfahren unter Verwendung des Pseudo-Time-Stepping. Während sich die vorstehend erwähnten Untersuchungen lediglich auf graue Gase mit einem Absorptions- bzw. Emissionsband beschränken, werden die vorhandenen Algorithmen in [83] und [84] auf Multi-Spektralband-Modelle erweitert. Krishnamoorthy gibt darüber hinaus in [85] eine kurze tabellarische Zusammenfassung der kürzlich unternommenen Anstrengungen zur Parallelisierung des Strahlungswärmetransportproblems mit dem DDP-Verfahren und anderen Methoden (Diskrete-Transfer-Methode, Monte-Carlo-Methode etc.). Die überwiegende Anzahl aller Methoden

zur Lösung der Transportgleichung für Wärmestrahlung basiert auf Entwicklungen aus der Kernphysik zum Problem des Neutronen- bzw. Teilchen-Transports (particle transport problem). Eine sehr ausführliche Beschreibung der bekannten Konzepte zur Parallelisierung dieses generalisierten Teilchen-Transport-Problems über die DDP-Methode findet sich daher auch in [86] und [87].

Als wesentlicher Vorteil der DDP-Methode gegenüber dem ADP-Verfahren ist vor allem der begrenzte Speicherbedarf je Prozessor zu nennen, da jeder beteiligte Prozessor während der Rechenphase nur Zugriff auf die Werte aus dem ihm zugewiesenen Teilgebiet haben muss. Darüber hinaus ist eine Aufteilung der Gesamtanzahl der Kontrollvolumen auf hunderte oder tausende Prozessoren viel eher möglich, als bei einer Parallelisierung über die meist begrenzte Anzahl der Kontrollwinkel. Auch die für die GPU-Technik wesentliche arithmetische Intensität (vgl. Abs. 4.3.2) ist gegenüber dem ADP-Verfahren bei der DDP-Methode wesentlich höher. Infolge vorstehend genannter Gründe ist das DDP-Verfahren für den Strahlungswärmetransport derzeit auch als Standardverfahren für parallele Berechnungen anzusehen.

Trotz aller Vorteile muss die DDP-Methode auf Systemen mit einer hohen Prozessoranzahl als wenig effizient eingestuft werden, da mit steigender Prozessoranzahl auch die Anzahl der erforderlichen Iterationsschritte zunimmt [78]. Insbesondere für die im Rahmen dieser Arbeit betrachteten GPU-Prozessoren, bei denen 10.000 und mehr unabhängige Prozesse gleichzeitig zu bearbeiten sind, ist dieser Effekt als äußerst störend anzusehen und kann die Effizienz des Verfahrens erheblich beeinflussen.

4.5.4.3 Der KBA-Algorithmus für orthogonale Netze

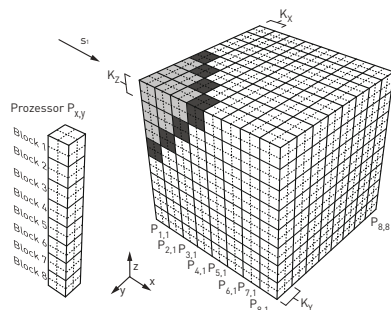
Aufgrund der oben beschriebenen Nachteile bei der Parallelisierung über das ADP- bzw. DDP-Verfahren auf massiv parallel arbeitenden Rechnern wird in dieser Arbeit auf ein alternatives Verfahren – das KBA-Verfahren für orthogonale Netze – zurückgegriffen, dessen Grundlagen im Folgenden kurz erläutert werden.

Für die Ermittlung der Intensitäten eines Kontrollvolumens muss die Strahlungsintensität in jeweils drei benachbarten Volumenelementen bekannt sein. Sofern auf die im Abschnitt 4.5.4.2 beschriebene iterative Berechnung verzichtet werden soll, kann daher im ersten Schritt für den betrachteten Kontrollwinkel lediglich eine einzige Raumzelle in der Ecke des Lösungsgebietes berechnet werden. Nur für diese Zelle sind zu Beginn alle drei Intensitäten I_{xu}^1 , I_{yu}^1 und I_{zu}^1 aus den Randbedingungen bekannt. Die Intensitäten aller übrigen Zellen hängen von den Intensitäten I_{xu}^1 , I_{yu}^1 und I_{zu}^1 dieser Raumzelle ab und können somit erst in den späteren Schritten bestimmt werden. Unter Berücksichtigung dieser Abhängigkeit ergibt sich für jeden Kontrollwinkel eine „Wellenfront“, die diagonal durch das Lösungsgebiet verläuft (vgl. Abb. 22).

Um eine einzelne Wellenfront mit mehreren Prozessoren bearbeiten zu können, muss ein Algorithmus die oben beschriebene Abhängigkeit der Raumzellen untereinander berücksichtigen. Koch, Baker und Alcouffe entwickelten in [88] ein Verfahren, das sich durch eine vergleichsweise hohe parallele Effizienz auszeichnet. Bei dem als KBA-Algorithmus bekannten Verfahren wird ein Lösungsgebiet mit $X \times Y \times Z$ Gitterzellen in $P_x \times P_y$ Bereiche aufgeteilt, wobei $P_x \times P_y$ die Anzahl der zur Verfügung stehenden Prozessoren ist. Jeder Bereich weist die Form eines Quaders mit $K_x \times K_y \times Z$ Gitterzellen auf ($K_x = X / P_x$, $K_y = Y / P_y$) und wird von je einem Prozessor $P_{x,y}$ bearbeitet. Zusätzlich wird jeder Bereich in Blöcke mit der Höhe K_z unterteilt, wobei jeder Block eine oder mehrere Zellebenen umfassen kann ($K_z \geq 1$). In Abb. 24 ist eine derartige Aufteilung für ein Lösungsgebiet mit 4.096 Gitterzellen ($X=Y=Z=16$), 64 Prozessoren ($P_x=P_y=8$) und die Blockhöhe $K_z=2$ exemplarisch dargestellt.

Das Lösungsgebiet wird nun für jeden Kontrollwinkel schrittweise separat bearbeitet. Während im ersten Schritt lediglich ein einzelner Prozessor mit einem Block beschäftigt ist (P_{11} –Block 1), werden im folgenden Schritt bereits drei Prozessoren eingesetzt (P_{11} –Block 2, P_{12} –Block 1, P_{21} –Block 1). In jedem weiteren Schritt wird die Anzahl der verwendeten Prozessoren sukzessive weiter erhöht. Gegen Ende der Wellenfront nimmt die Anzahl der beteiligten Prozessoren wieder ab, bis schließlich der letzte Block nur noch von einem einzigen Prozessor bearbeitet wird (P_{88} –Block 8). Obwohl an jedem Schritt eine unterschiedliche Anzahl von Prozessoren beteiligt ist, ist die benötigte Rechenzeit infolge der parallelen Bearbeitung für einen Schritt jeweils identisch, d.h. alle Blöcke innerhalb einer Diagonalebene werden parallel verarbeitet.

Abb. 24: Vierter Berechnungsschritt einer Wellenfront in einem orthogonalen Gitter mit $16^3 = 4.096$ Zellen unter Verwendung von $8 \times 8 = 64$ Prozessoren und acht Blöcken je Prozessor (dunkelgraue Zellen befinden sich aktuell in Berechnung, hellgraue Zellen wurden bereits bearbeitet)

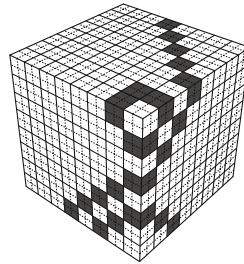


Sofern die Kommunikationszeiten zwischen den Prozessoren vernachlässigt werden, kann die parallele Effizienz dieses Algorithmus' wie folgt angegeben werden [88]:

$$PCE = Z / [Z + K_z (X/K_x + Y/K_y - 2)] \quad (42)$$

Für das Lösungsgebiet in Abb. 24 ergibt sich daraus beispielsweise ein Wert in Höhe von $PCE = 36,7\%$. Ursächlich für diesen geringen Wert ist vor allem die Tatsache, dass die Kapazität des Prozessors P_{11} brach liegt, sobald dieser seinen letzten Block (P_{11} –Block 8) bearbeitet hat. Auch alle übrigen Prozessoren werden nicht mehr verwendet, sobald diese ihre jeweils letzten Blöcke verarbeitet haben. Zweckmäßigerweise schlagen Koch, Baker und Alcouffe zur Erhöhung der parallelen Effizienz vor, mit den freiwerdenden Prozessoren einen weiteren Kontrollwinkel zu bearbeiten. Die freie Kapazität des Prozessors P_{11} wird in diesem Fall genutzt, indem dieser nach Verarbeitung von Block 8 umgehend eine zweite Wellenfront für einen neuen Kontrollwinkel startet. Auch die Prozessoren P_{12} und P_{21} können ohne Verzögerung ihre Arbeit an der zweiten Wellenfront aufnehmen, da die entsprechende „Vorarbeit“ durch den Prozessor P_{11} (P_{11} –Block 1) bereits geleistet wurde. Da die Kontrollwinkel auf alle Oktanten verteilt sind und damit die Wellenfronten an unterschiedlichen Ecken des Lösungsgebietes beginnen, muss dabei nicht von Nachteil sein. In Abb. 25 ist ein exemplarischer Zustand für das Beispielgebiet bei gleichzeitiger Verwendung mehrerer Wellenfronten dargestellt.

Abb. 25: Exemplarische Darstellung einer gleichzeitigen Bearbeitung des Lösungsgebietes über mehrere Wellenfronten entsprechend der freien Prozessorkapazitäten



Die parallele Recheneffizienz steigt durch diesen Ansatz erheblich. Nach [88] gilt

$$PCE = MZ / [M Z + K_z (X/K_x + Y/K_y - 2)], \quad (43)$$

wobei M die Anzahl der in der Berechnung verwendeten Kontrollwinkel darstellt. Sofern im betrachteten Beispiel die Strahlungsintensitäten für 100 Kontrollwinkel zu ermitteln sind, steigt die parallele Effizienz damit auf $PCE = 98,3\%$. Lediglich zu Beginn und zum Ende des gesamten Berechnungslaufes liegen Prozessorkapazitäten brach.

Die bei Anwendung des KBA-Algorithmus insgesamt benötigte Rechenzeit ist – unabhängig von der erreichten Recheneffizienz – in der Praxis auch von der geometrischen Form der einzelnen Bereiche und Blöcke abhängig. Hier fällt vor allem der Kommunikationsbedarf zwischen den Prozessoren ins Gewicht. Hoisie, Lubeck und Wasserman analysieren in ihren Arbeiten [89, 90, 91, 92] verschiedene Möglichkeiten zur Einteilung des Lösungsgebietes im Hinblick auf die benötigte Rechenzeit (computation time, t^{comp}) sowie die Zeit

für die Kommunikation zwischen den beteiligten Prozessoren (communication time, t^{comm}). Die Analyse basiert auf einer bestehenden Applikation zum Neutronen-Transport-Problem [93]. Während sich das Modell von Hoisie et. al. noch auf eine einzelne Wellenfront beschränkt, stellen Mathis, Amato und Adams in [94] ein erweitertes Modell vor, welches auch den Leistungsgewinn durch eine gleichzeitige Verwendung von mehreren Wellenfronten für unterschiedliche Kontrollwinkel einschließt. Die Möglichkeit einer Anwendung des KBA-Algorithmus auf unstrukturierten Netzen demonstrieren Plimpton und Hendrickson in [95] sowie Pautz in [96]. Hier ergeben sich gegenüber den orthogonalen Netzen zusätzliche Problemstellungen (z. B. Zirkelbezüge), die in [97] ausführlich diskutiert werden.

Der Wellenfrontalgorithmus nach Koch, Baker und Alcouffe stellt sich in der Praxis als sehr effizient heraus, wenn eine hohe Bandbreite bei der Kommunikation zwischen den beteiligten Prozessoren vorhanden ist. Auf Shared-Memory-Systemen, bei denen jeder Prozessor auf das gesamte Lösungsgebiet zugreifen kann, stellt der KBA-Algorithmus daher häufig die erste Wahl dar. Für die in dieser Arbeit durchgeführte Parallelisierung des Strahlungswärmetransportproblems wird der KBA-Algorithmus als Grundlage verwendet (vgl. folgende Abschnitte).

4.5.5 Anpassung des KBA-Verfahrens an GPU-Systeme

Für die im Rahmen dieser Arbeit betrachteten GPU-Systeme sind einige Besonderheiten zu beachten, die eine Anpassung des KBA-Verfahrens erforderlich werden lassen. So setzt der KBA-Algorithmus die Möglichkeit zur Synchronisation der einzelnen Prozessoren untereinander voraus (z. B. Blockieren einzelner Prozesse, bis alle erforderlichen Gitterzellen berechnet wurden). Eine derartige Synchronisation kann bei der GPU-Technik jedoch ausschließlich zwischen den Prozessen auf Threadebene innerhalb eines Threadblocks erfolgen (vgl. Erläuterungen im Abs. 3.3). Auf die Reihenfolge der Bearbeitung der einzelnen Threadblöcke an sich kann softwareseitig kein Einfluss genommen werden. Dennoch muss dafür gesorgt werden, dass eine möglichst große Anzahl von Threadblöcken für die gleichzeitige Bearbeitung zur Verfügung steht (bei heutiger Technologie mindestens einige Hundert). Erst wenn mehrere Threadblöcke je Prozessorkern (Multiprozessor) gleichzeitig verarbeitet werden, wirken sich die Latenzzeiten für den Zugriff auf den Arbeitsspeicher nicht mehr negativ auf die Berechnungsgeschwindigkeit aus.

Unter Berücksichtigung dieser Voraussetzungen bietet es sich an, das Lösungsgebiet entsprechend Abb. 24 zunächst in Threadblöcke mit einer definierten Zellenanzahl $K_x \times K_y \times K_z$ aufzuteilen (z. B. $K_x \times K_y \times K_z = 8 \times 8 \times 8$). Für alle jeweils in einer diagonalen Ebene liegenden Blöcke sind die Intensitäten aller Zellen rückwärtig der Wellenfront aus dem vorherigen Schritt bekannt, so dass alle Threadblöcke einer Diagonalebene zusammen in die Pipeline des GPU-Prozessors eingereiht werden können. Aufgrund der fehlenden Syn-

chronisation auf Blockebene kann mit der Berechnung der folgenden Diagonalebene erst nach Abschluss der vorausgegangenen Diagonale begonnen werden, so dass jeweils zu Beginn und Ende der Wellenfront nicht genügend Threadblöcke je Prozessorkern vorhanden sind. Dieser Nachteil wird „systembedingt“ in Kauf genommen. Die Anzahl der erforderlichen Diagonalen für einen vollständigen Durchlauf einer Wellenfront bestimmt sich für ein Lösungsgebiet mit $l \times b \times h$ Blöcken wie folgt:

$$S_{\max} = h + l + b - 2 \quad (44)$$

Die Anzahl der in der jeweiligen Diagonalen s ($1 \leq s \leq S_{\max}$) vorhandenen Threadblöcke kann wie folgt ermittelt werden:

$$D(s) = f(s) - f(s-l) - f(s-b) - f(s-h) + f(s-l-b) + f(s-l-h) + f(s-b-h) \quad (45)$$

mit $f(x) = \max(x; 0) \cdot (x+1)/2$

Im Gegensatz zur Blockebene ist innerhalb jedes einzelnen Threadblocks eine Synchronisation der Prozesse (threads) untereinander durchaus möglich, so dass innerhalb eines Blocks ein unmodifizierter KBA-Algorithmus zum Einsatz kommen kann (s. Abs. 4.5.4.3). Dabei wird jeder Block nach Abb. 24 einer Einzelzelle ($K_x \times K_y \times K_z = 1 \times 1 \times 1$) und somit einem separaten Prozess zugewiesen. Als Resultat werden somit je Diagonalebene $D(s)$ Blöcke mit jeweils einem blockinternen KBA-Algorithmus gleichzeitig verarbeitet.

In Abb. 26 ist die Anzahl der voneinander unabhängigen Blöcke entsprechend Gleichung (45) für ein Lösungsgebiet mit unterschiedlichen Blockgrößen aufgetragen (dreiecksförmige Markierungen). Jeder Block wird dabei durch einen separaten KBA-Algorithmus bearbeitet. Die in Abb. 26 grau-markierte Fläche stellt die maximale Anzahl der in allen KBA-Algorithmen parallel ablaufenden Prozesse dar. Die Einbrüche bei den maximal gleichzeitig laufenden Prozessen jeweils am Anfang und am Ende jeder Diagonalebene wirken sich negativ auf die parallele Effizienz des Algorithmus¹ aus. Ursache hierfür ist

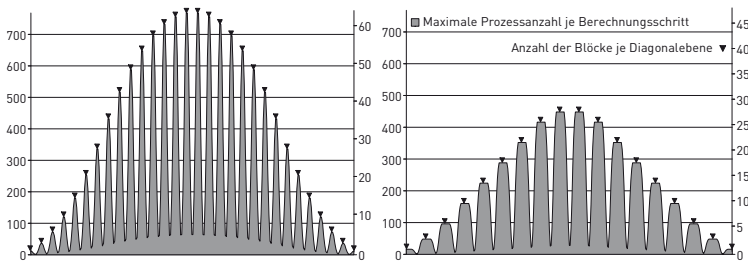
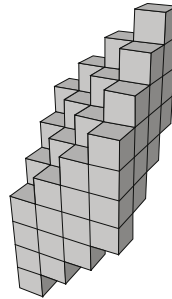


Abb. 26: Theoretisch mögliche Prozessanzahl je Berechnungsschritt für ein Gitter mit 65.536 Zellen, aufgeteilt in $8 \times 8 \times 16$ Blöcke und $4 \times 4 \times 4$ Zellen je Block (links) bzw. $8 \times 8 \times 4$ Blöcke und $4 \times 4 \times 16$ Zellen je Block (rechts)

der Umstand, dass infolge der fehlenden Synchronisation auf Blockebene erst nach vollständiger Bearbeitung einer Blockdiagonalen mit der nächsten Diagonalebene fortgefahren werden kann.

Eine Möglichkeit, die in Abb. 26 dargestellten „Einbrüche“ bei der Prozessorauslastung zu reduzieren, besteht in der Änderung der Blockform. Sofern die quaderförmigen Blöcke als Rhomboid ausgebildet werden (s. Abb. 27), reduzieren sich die Einbrüche für alle vollständig im Lösungsgebiet liegenden Blöcke auf Null. In diesem Fall sind für jede Zelle zum Zeitpunkt ihrer Verarbeitung die jeweiligen Intensitäten in den Nachbarzellen bekannt. Lediglich für die am Rande liegenden Blöcke können nicht alle Prozesse (threads) beschäftigt werden.

Abb. 27: Ausbildung eines Blocks als Rhomboid zur Erhöhung der blockinternen Prozessorauslastung (Darstellung einer möglichen Ausführungsvariante)



Dem Vorteil der blockinternen Prozessorauslastung bei rhomboidförmigen Blöcken steht ein wesentlicher Nachteil gegenüber. Während bei quaderförmigen Blöcken die schrittweise Vergrößerung der Blockanzahl je Diagonalebene nach Gleichung (45) erfolgt (Blockanzahl je Berechnungsschritt: 1, 3, 6, 10, ...), können bei rhomboidförmigen Blöcken je Berechnungsschritt weit weniger Blöcke je Diagonalebene verarbeitet werden (Blockanzahl je Berechnungsschritt: 1, 1, 1, 3, 3, 3, 6, 6, 6, ...). Die daraus resultierende höhere Anzahl von Berechnungsschritten wirkt sich negativ auf die Leistungsfähigkeit des Algorithmus' aus. Im Abschnitt 4.5.6 wird gezeigt, dass bei Verwendung der GPU-Technik insbesondere der schnelle Anstieg der verfügbaren Blockanzahl maßgebend für die parallele Effizienz ist, so dass der Ansatz mit rhomboidförmigen Blöcken hier nicht weiter verfolgt wird. Eine Änderung der Blockform kann jedoch für zukünftige Ausführungsvarianten der massiv parallelen Prozesstechnik durchaus zweckmäßig sein.

Alternativ zur Änderung der Blockform kann eine Effizienzsteigerung erreicht werden, wenn für das blockinterne KBA-Verfahren ein Multi-Wellenfrontalgorithmus entsprechend Abb. 25 eingesetzt wird. Dabei werden innerhalb jedes Threadblocks alle bzw. mehrere Kontrollwinkel gleichzeitig bearbeitet. Obwohl die systembedingten „Einbrüche“ bei den parallelen Prozessen dadurch nicht vollständig vermieden werden, wird deren Anzahl

infolge gleichzeitiger Bearbeitung mehrerer Kontrollwinkel wesentlich reduziert. Die Anzahl der Rechenschritte, die innerhalb eines Blocks zur Verarbeitung der Diagonalebenen erforderlich sind, werden für n gleichzeitig zu verarbeitende Kontrollwinkel wie folgt bestimmt:

$$S_{\max} = n \cdot h + l + b - 2 \quad (46)$$

Als nachteilig bei der gleichzeitigen Verarbeitung mehrerer Kontrollwinkel stellt sich der erhöhte Rechenaufwand je Gitterzelle heraus. Infolge der komplexeren Systematik an sich sowie der erforderlichen aufwendigeren Speicherverwaltung (temporäres Zwischenspeichern von winkelabhängigen Strahlungsintensitäten) fällt die Anzahl der arithmetischen Operationen und Speicherzugriffe je Zelle nahezu doppelt so hoch aus. Wie im folgenden Abschnitt gezeigt wird, ist bei Verwendung der GPU-Technik jedoch weniger die Anzahl der arithmetischen Operationen, sondern vielmehr die Bereitstellung ausreichender Datenmengen aus dem Arbeitsspeicher von Bedeutung, so dass sich die gleichzeitige Verarbeitung mehrerer Kontrollwinkel als zweckmäßig erweist.

4.5.6 Implementierung des KBA-Verfahrens in FDS

Im Rahmen dieser Arbeit wurde der in FDS enthaltene sequenzielle Algorithmus zur Lösung des Strahlungswärmetransportproblems durch das im Abschnitt 4.5.5 dargestellte, auf die GPU-Technik angepasste KBA-Verfahren ersetzt. Dabei wurde sowohl ein einfacher Wellenfrontalgorithmus als auch ein Multi-Wellenfrontalgorithmus entwickelt. Da bei letzterem Algorithmus mehrere Kontrollwinkel zugleich bearbeitet werden, waren zusätzlich zur Parallelisierung auch umfangreichere Änderungen bei der Speicherverwaltung erforderlich (Neuordnung temporärer Variablen etc.). Im Anlagenteil C ist der Kern der neu entwickelten Routine zum Multi-Wellenfrontalgorithmus wiedergegeben und stichpunktartig erläutert. Die Erläuterungen können sinngemäß auf den einfachen Wellenfrontalgorithmus (vgl. beigefügte CD-ROM) übertragen werden.

Die Verifizierung der neuen Algorithmen kann mit einem Ergebnisvergleich zwischen der Basissoftware und der GPU-Version durch das Einschalten des Diagnosemodus (vgl. Abs. 5.1) erfolgen. Da dadurch nicht alle möglichen Fehlerquellen auffindbar sind, wird in dieser Arbeit zusätzlich ein Beispiel gerechnet, für das die analytische Lösung bekannt ist (vgl. Abs. 5.3). Auf der Grundlage dieser Beispielrechnung können darüber hinaus Aussagen zur Leistungsfähigkeit der neuen Algorithmen getroffen werden.

In Abb. 28 (links) ist die Leistungssteigerung (SpeedUp) der neuen Wellenfrontalgorithmen bei Verwendung einer GPU-Karte vom Type GeForce GTX 580 gegenüber einer CPU vom Typ Intel Core i7 980X dargestellt. Es ist ersichtlich, dass die Leistungssteigerung bei kleineren Lösungsgebieten stark abfällt, da bei derartigen Lösungsgebieten keine ausreichende Anzahl unabhängiger Blöcke generiert werden kann. Sofern ein ausrei-

chend großes Lösungsgebiet vorliegt, wird bei dem Beispiel aus Abschnitt 5.3 mit einem einfachen Wellenfrontalgorithmus eine Leistungssteigerung um etwa Faktor sieben erreicht. Bei Verwendung des neu entwickelten Multi-Wellenfrontalgorithmus steigt dieser Wert auf etwa Faktor zehn an.

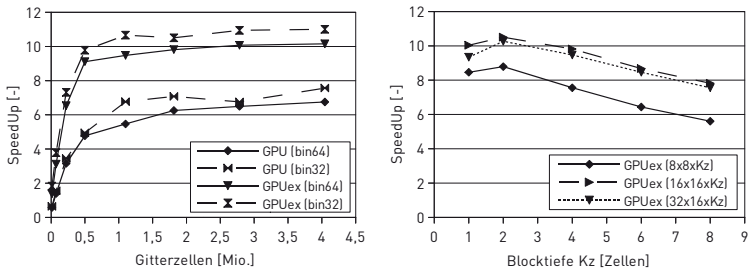


Abb. 28: Leistungssteigerung (SpeedUp) einer GeForce GTX 580-GPU gegenüber einer Intel Core i7 980X-CPU bei Einsatz der neuen Wellenfrontalgorithmen und gleichzeitiger Berechnung eines (GPU) bzw. mehrerer (GPUex) Kontrollwinkel, unterschiedlich große Lösungsgebiete (links) und Blockgrößen (rechts)

Beschleunigungsfaktoren in Höhe von 30 und mehr, wie diese im strömungsmechanischen Beispiel in Abschnitt 5.2 erreicht werden, sind für den Strahlungswärmetransport in gasförmigen Medien prinzipiell nicht zu erreichen. Ursache hierfür ist die Natur der physikalischen Problemstellung. Durch die Abhängigkeit der Zellen untereinander in Verbindung mit den Einschränkungen der GPU-Technik (fehlende Synchronisation auf Blockebene), fällt die parallele Effizienz gegenüber dem strömungsmechanischen Teil der Software geringer aus. Auch durch die Verwendung von Zahlen mit einfacher Genauigkeit (bin32) wird im Vergleich zur doppelten Genauigkeit (bin64) kaum eine weitere Leistungssteigerung erreicht. Durch die geringe arithmetische Intensität der Problemstellung wirkt sich die dadurch erreichte Beschleunigung der arithmetischen Operationen vergleichsweise gering auf die Gesamtrechnenzeit aus.

In Abb. 28 (rechts) ist die Leistungssteigerung des neuen Multi-Wellenfrontalgorithmus in Bezug auf unterschiedliche Blockgrößen $K_x \times K_y \times K_z$ dargestellt. Dabei fällt auf, dass ein Leistungsmaximum bei einer Blockhöhe von $K_z = 2$ (zwei Zellen Blockhöhe) erreicht wird. Dieser relativ kleine Wert lässt den Schluss zu, dass vor allem die Zeit bis zur Vollausslastung der GPU maßgeblich für die erreichte parallele Effizienz ist. Zu Beginn und zum Ende der Wellenfront können die Latenzzeiten für die Bereitstellung der Daten vom und zum Arbeitsspeicher nicht durch die gleichzeitige Bearbeitung einer ausreichend großen Blockanzahl kompensiert werden (vgl. Hüllkurven in Abb. 26, S. 70). Es ist daher von maßgebender Bedeutung, die Blöcke derart „flach“ zu wählen, dass diese Zeitspannen möglichst gering ausfallen. Nach dem Erreichen der Vollausslastung werden die theoretisch vorhandenen „Einbrüche“ in der Prozessanzahl durch den hardwareseitigen

Mechanismus der GPU-Karte wesentlich reduziert. Aus diesem Sachverhalt wird ebenfalls ersichtlich, warum sich die Anpassung der Blockform entsprechend Abschnitt 4.5.5 (rhomboidförmige Blöcke) negativ auf die Gesamtleistung auswirkt: In diesem Fall werden die An- und Auslaufzeiten bis zur Vollausslastung der GPU-Karte erheblich verlängert.

4.5.7 Zusammenfassung

In diesem Abschnitt wurden Möglichkeiten zur Parallelisierung der in FDS enthaltenen Algorithmen zum Strahlungswärmetransport in gasförmigen Medien vorgestellt. Im Anschluss wurde ein bereits bekannter Algorithmus ausgewählt (KBA-Verfahren), an die GPU-Technik angepasst und in zwei verschiedenen Versionen (einfacher und Multi-Wellenfrontalgorithmus) in die Software FDS integriert. Ein Leistungsvergleich auf der Grundlage des in Abschnitt 5.3 dargestellten Beispiels zeigt, dass der Multi-Wellenfrontalgorithmus in Verbindung mit relativ „flachen“ Blöcken ($K_x \times K_y \times K_z = 16 \times 16 \times 2$) im Hinblick auf die erreichbare parallele Effizienz die besten Ergebnisse liefert. Als Ursache hierfür wurde die Natur des physikalischen Strahlungswärmetransports in gasförmigen Medien identifiziert. Durch die Abhängigkeiten der Zellen untereinander tritt die reine Bearbeitungszeit für arithmetische Operationen in den Hintergrund, wohingegen die schnelle Bereitstellung einer ausreichenden Menge von parallelen Datensätzen maßgeblich wird.

4.6 Entwicklung eines Profilers

Während der Anpassung der Software FDS an die GPU-Technik war es notwendig über den Rechenbedarf bzw. die Ausführungsgeschwindigkeit der einzelnen Programmteile detailliert Kenntnis zu haben. In diesem Zusammenhang wurde eine speziell auf das Programm FDS zugeschnittene Profilersoftware entwickelt, mit der der vorliegende Quellcode automatisch modifiziert werden kann. Nach Änderung des Quelltextes durch die Profilersoftware können wahlweise Geschwindigkeit sowie Speicherbedarf der einzelnen Routinen zur Programmlaufzeit ausgegeben werden. Die Ergebnisse wurden zur Optimierung der Rechen- sowie Speichereffizienz der Software herangezogen. In der Profilersoftware sind zwei unterschiedliche Funktionsmodi wählbar:

- **Profilier-Modus: TIME**

Bei Anwahl des Profiler-Modus TIME wird der Quellcode von FDS nach Funktionsaufrufen vom Typ „CALL“ durchsucht. Diese werden im Anschluss derart modifiziert, dass während des Programmlaufes die Anzahl der Aufrufe sowie die benötigte Zeit für die Bearbeitung jeder Funktion aufgezeichnet wird. Die Ergebnisse werden zum Abschluss des Rechenlaufs in einer Datei ausgegeben (s. Abb. 29).

NR.	NAME	CALLS	TIME
5	CUDA::CUC_INIT_DEVICE	1	0.6090000000
67	DUMP::DUMP_SLCF	61	0.5450000000
80	DUMP::FLUSH_LOCAL_BUFFERS	55	0.4810000000
25	CUDA::CUC_DENSITY	115	0.4400000000
66	DUMP::DUMP_SMOKE3D	56	0.3730000000
78	DUMP::FLUSH_GLOBAL_BUFFERS	55	0.1750000000

Abb. 29: Auszug aus der Ergebnisdatei einer Zeitmessung (PROFILER_MODE TIME): Angabe von Nummer (NR.) und Name (NAME) der Funktion, der Anzahl der Aufrufe über den gesamten Rechenlauf (CALLS) sowie die hierfür benötigte Rechenzeit (TIME)

- **Profiler-Modus: MEMORY**

Neben der benötigten Zeit für die einzelnen Funktionsaufrufe ist der Speicherbedarf für die Variablen als maßgebend für die Effizienz der Software anzusehen. Bei Anwahl des Profiler-Modus MEMORY wird der Quellcode der Software FDS derart modifiziert, dass vor Beginn der tatsächlichen Berechnung die Größe des belegten Arbeitsspeichers aufgeteilt auf die einzelnen Variablen ausgegeben wird. Die Speicheraufteilung wird abschließend in einer Tabelle ausgegeben (s. Abb. 30).

NR.	NAME	MEMORY [MB]	%
32	DUMP::M%QQ	23,3050	5,1252
143	INIT::M%WALL	16,1280	3,5468
80	INIT:: M%RHO	9,3220	2,0501
81	INIT:: M%RHOS	9,3220	2,0501
82	INIT:: M%TMP	9,3220	2,0501
83	INIT:: M%FRHO	9,3220	2,0501

Abb. 30: Auszug aus den Ergebnissen einer Speicheranalyse (PROFILER_MODE MEMORY): Angaben zur Nummer (NR.) und Name (NAME) der Variable, des Speicherbedarfs (MEMORY) sowie des prozentualen Anteils am Gesamt-Speicherbedarf (%).

5 Verifikation und Performance

Bei der Portierung eines bestehenden Quelltextes von FORTRAN nach C/CUDA treten erfahrungsgemäß bereits ohne Änderung an den Algorithmen Fehler auf. Die in dieser Arbeit erfolgte Anpassung der Software an die GPU-Technik (Umordnung von Speicherbereichen, Parallelisierung von Schleifen, Entwicklung neuer Algorithmen etc.) beinhaltet darüber hinaus eine ganz erhebliche Anzahl zusätzlicher Fehlerquellen. In diesem Abschnitt werden daher Maßnahmen zusammengefasst, die die Stabilität der sequenziellen auf die parallele Version übertragen sollen. Zum einen wird hierfür ein „Diagnosemodus“ vorgestellt (vgl. Abs. 5.1), mit dem ein Teil möglicher Fehlerquellen bereits während der Portierung gefunden werden kann. Zusätzlich werden in den Abschnitten 5.2 und 5.3 Beispiele gerechnet, die zur Verifikation der neuen Programmteile dienen.

5.1 Verifikation im Diagnosemodus (diagnostic mode)

Ein wesentlicher Aspekt bei der Verifikation des portierten Quelltextes bzw. der neuen Algorithmen stellt der im Rahmen dieser Arbeit entwickelte Diagnosemodus (diagnostic mode) dar. Hierdurch lässt sich ein Großteil der auftretenden Fehler bereits während der Parallelisierung des Quelltextes finden und beseitigen. Dazu wird die Berechnung schrittweise zunächst mit den neuen parallelen Algorithmen und anschließend unter Verwendung des bestehenden sequenziellen Quelltextes durchgeführt (vgl. Abb. 14, S. 48). Anschließend werden eigens für den Diagnosemodus entwickelte Vergleichsroutinen aufgerufen, die beide Ergebnisse zellweise auf Übereinstimmung prüfen.

Als wesentliche Voraussetzung für den Einsatz des Diagnosemodus muss die Ausführung der neuen GPU-Algorithmen mittels Emulation auf der CPU erfolgen. Bei Verwendung des GPU-Prozessors treten infolge unterschiedlicher Ausführungsreihenfolge von arithmetischen Operationen Ergebnisabweichungen auf, die sich über die Zeit aufsummieren und einen Ergebnisvergleich unmöglich machen. Im Diagnosemodus wird daher ein in C/CUDA enthaltener Emulationsmodus verwendet, der die parallelen GPU-Algorithmen emuliert und auf der CPU sequenziell ausführt. Dadurch wird ein vollständiger Ergebnisvergleich in allen Zellen nach beliebig kleinen Berechnungsschritten (beispielsweise nach Abschluss einer Schleife) möglich. Dieser, direkte schleifenweise Vergleich der Rechenergebnisse kann in der Software durch Setzen des Wertes `GPU_DIAG_MODE` in den Dateien `CUDA.F90` (FORTRAN) sowie `fds_cuda_def.h` (C/CUDA) eingeschaltet werden. Gleichzeitig ist der Quelltext unter Verwendung des Emulationsmodus zu kompilieren (vgl. hierzu [98]).

Obwohl durch die Verwendung des Diagnosemodus ein großer Teil möglicher Fehlerquellen im neuen Quelltext ausgeschlossen wird, ist eine vollständige Übereinstimmung der Ergebnisse zwischen der GPU- und der Basisversion nicht zu erreichen. Wie die Untersuchungen zeigen, sind insbesondere die folgenden Punkte ursächlich für Ergebnisabweichungen, die über den Diagnosemodus nicht zu finden sind:

- **Chaotische fluktuierende Ergebnisse in Wand-Eckbereichen**

An vielen Stellen findet in der Software FDS ein Informationsaustausch zwischen den Gas- und Wandzellen statt. So werden beispielsweise in jedem Berechnungsschritt die Geschwindigkeitskomponenten in den oberflächennahen Gaszellen über eine (wählbare) Wandfunktion korrigiert, um die Grenzschichtströmung besser darzustellen. Zweckmäßigerweise erfolgen derartige Korrekturen mittels einer Schleife über alle vorhandenen Wandzellen. Im Falle einer sequenziellen, nicht parallelen Implementierung werden die Gaszellen in Abhängigkeit ihrer zugehörigen Wandzellen dabei nacheinander in der Reihenfolge ihres Auftretens im Hauptspeicher korrigiert (vgl. Abb. 31, links).

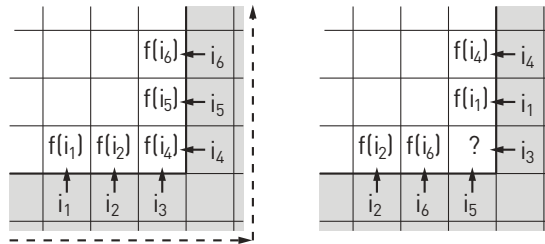


Abb. 31: Übernahme von Informationen aus den Wandzellen (grau) in die Gaszellen (weiß) mit einer Schleife über alle Wandzellen ($i_1 \dots i_6$), sequenzielle Bearbeitung der Schleife (links) und parallele Bearbeitung (rechts)

Zuvor ermittelte Werte für die Gaszellen in Eckbereichen werden in FDS in vielen Fällen ersatzlos mit nachfolgend berechneten Werten aus den benachbarten Wandzellen überschrieben. So wird beispielsweise der in der Eckzelle enthaltene Funktionswert $f(i_3)$ (vgl. Abb. 31, links) durch den zeitlich später ermittelten Funktionswert $f(i_4)$ überschrieben. Bei Verwendung der GPU-Version ist die Bearbeitungsreihenfolge der Wandzellen innerhalb der Schleife jedoch nicht festgelegt, da jede Wandzelle einem anderen Prozess zugewiesen sein kann. Das Ergebnis in der Eckzelle ist in diesem Fall davon abhängig, welcher der beiden Prozesse (i_3 oder i_5) zuerst seine Berechnung abgeschlossen hat (vgl. Abb. 31, rechts). Aus diesem Grunde werden bei Verwendung der GPU-Version auch dann abweichende Ergebnisse erzielt, wenn sowohl Eingabewerte als auch Hardware vollkommen identisch sind. Ein schrittweiser Ergebnisvergleich der Einzelzellen – wie im Diagnosemodus vorgesehen – ist infolge dieser Problematik für zur Verifikation der GPU-Version nicht ausreichend.

- **Fehlerhafte Implementierung paralleler Algorithmen**

Mit der im Diagnosemodus ablaufenden Emulation können Fehler, die allein infolge der physikalisch gleichzeitigen Bearbeitung von Programmsequenzen durch mehrere Prozesse entstehen, nicht aufgedeckt werden (z. B. Fehler infolge mangelhafter Synchronisation an kritischen Punkten). Ursache hierfür ist, dass im Emulationsmodus aus physikalischer Sicht nur der Hostcomputer die Algorithmen mit einem einzigen Prozessor (und damit sequenziell) bearbeitet.

- **Rundungsfehler**

Prinzipiell ist auf verschiedenen Computersystemen und insbesondere bei Verwendung verschiedener Compiler ein unterschiedliches Rundungsverhalten festzustellen. Sofern sich in einer entsprechenden Software derartige Rundungsfehler über die Zeit hin aufsummieren, ist nicht davon auszugehen, dass auf jeder Plattform bzw. mit jedem Compiler auch bei identischem Quellcode ein vollkommen identisches Ergeb-

nis erhalten wird. Für die entwickelte GPU-Version von FDS werden drei unterschiedliche Compiler eingesetzt (Microsoft Visual C++ 2005, Intel Fortran Compiler 10.0, CUDA Compiler Driver NVCC 2.0). Nähere Informationen zum Rundungsverhalten der einzelnen Compiler können der technischen Dokumentation [98, 99] entnommen werden. In Bezug auf die neu entwickelten GPU-Algorithmen ist daher zu erwarten, dass aufgrund des unterschiedlichen Rundungsverhaltens der Compiler die Ergebnisse zwischen der Basis- und der GPU-Version mit der Zeit auch dann auseinander divergieren, wenn alle Routinen identisch übernommen werden. Wenngleich diese Ergebnisunterschiede zu Beginn einer Berechnung nur gering ausfallen ($\Delta x < 10^{-10}$), wird infolge Fehlerfortpflanzung der direkte Ergebnisvergleich zu späteren Programmaufzeiten verhindert. Auch aus diesem Grunde ist eine separate Verifikation der Ergebnisse aus der GPU-Version über Beispielrechnungen zwingend erforderlich.

5.2 Beispiel 1: Isothermer Freistrah

Zur Überprüfung der vorgenommenen Änderungen im strömungsmechanischen Programmteil sowie der Abschätzung des Leistungszuwachses der GPU- gegenüber der Basisversion wird in diesem Abschnitt ein isothermer Niederdruck-Freistrah modelliert. Bei diesem strömungsmechanischen Grundlagenproblem handelt es sich um den Austritt eines Fluids aus einer runden, rechteckigen oder spaltförmigen Öffnung in ein Fluid gleicher Art im benachbarten Raum bzw. in die freie Umgebung. Baupraktische Fragestellungen hierzu betreffen neben dem Luftvolumenstrom durch die Öffnung vor allem Fragen hinsichtlich der Lufteinmischung im Bereich der Strahlgrenze (z.B. Einmischung nachströmender Frischluft).

5.2.1 Theoretische und experimentelle Grundlagen

Für turbulente Freistrahlen mit Reynolds-Zahlen größer als $Re > 8.500 - 10.000$ gibt das in Abb. 32 dargestellte Modell das in Versuchen beobachtete Verhalten gut wieder. Hinter der Austrittsebene kann dabei nach kurzer laminarer Anlaufstrecke eine Aufweitung des Strahls beobachtet werden, die im reibungsbedingten Mitreißen von Sekundärfluid aus der Umgebung begründet ist. Die Trennschicht zwischen Primär- und Sekundärfluid wird als „freie Strahlgrenze“ bezeichnet. Mit steigender Aufweitung des Freistrahls nimmt der im Strahl transportierte Massenstrom zu, während das Geschwindigkeitsprofil von den Rändern her abflacht. Der Anteil des Primärfluids mit unveränderter Austrittsgeschwindigkeit c_0 bildet einen kegelförmigen Strahlkern, der Bereich zwischen Strahlkern und Sekundärfluid die sogenannte Mischzone.

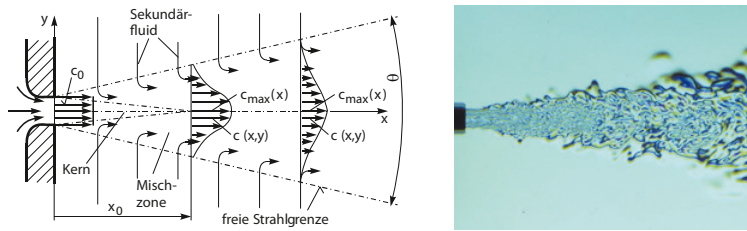


Abb. 32: Ausbreitung eines isothermen Freistrahls im Modell [101] (links) und im Versuch [100] (rechts)

Die Länge des Kerns nimmt bei einem runden Freistrahls einen Wert in Höhe von

$$x_0 = \frac{D}{m} \quad (47)$$

an, wobei D den Durchmesser der Austrittsöffnung und m die Mischzahl darstellen. Die Mischzahl (und damit die Kernlänge) hängt vor allem vom Turbulenzgrad des Strahls ab. Für Freistrahlen mit niedriger Turbulenz (abgerundete Einlauföffnungen) beträgt diese nach [101] etwa $m \sim 0,14$ bis $0,17$. Für eine quadratische Austrittsöffnung gleicher Querschnittsfläche und Austrittsgeschwindigkeit gelten die gleichen Gesetzmäßigkeiten, wie für den runden Freistrahls. Nach [101] kann die Kernlänge für einen Freistrahls mit einer quadratischen Austrittsöffnung der Breite h wie folgt bestimmt werden:

$$x_0 = \frac{h}{m} \quad \text{mit } m \sim 0,17 - 0,20 \quad (48)$$

Der Ausbreitungswinkel θ kann sowohl bei runden als auch rechteckigen Öffnungen mit $\theta \sim 24^\circ$ abgeschätzt werden. Für weitere physikalische Kenngrößen (Maximalgeschwindigkeit, Volumenstrom etc.) wird auf [101] verwiesen.

5.2.2 Zusammenfassung bisheriger Untersuchungen

Bereits in [102, 103] wurde versucht, einen Freistrahls mit der Software FDS nachzurechnen. Dazu wurde ein rechteckiges Lösungsgebiet ($2 \text{ m} \times 2 \text{ m} \times 12 \text{ m}$) modelliert, dass an fünf Seiten offene Randbedingungen aufweist. In der Mitte der sechsten Wand wurde ein Luftauslass ($0,4 \text{ m} \times 0,4 \text{ m}$) angeordnet, aus dem ein konstanter Luftstrom mit unterschiedlichen Geschwindigkeiten ($c_0 = 1, 2$ und 10 m/s) strömt. Im Anschluss wurden nach einer Anlaufphase (120 s) die Geschwindigkeiten in der Symmetrieachse über 30 s gemessen und gemittelt. Das aus diesen Werten erhaltene Geschwindigkeitsprofil wurde für die Beurteilung des Modells herangezogen. Der geometrische Aufbau des Modells und ein qualitatives Geschwindigkeitsbild sind zur Veranschaulichung der Situation in Abb. 33 skizziert.

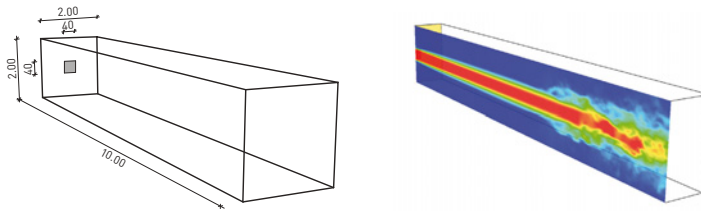


Abb. 33: Modellgeometrie A aus der Analyse [102] (links), Geschwindigkeitsprofil in der Symmetrieebene bei $t = 150$ s (Modell mit 2,5 cm Gitterabstand)

In [102] wurden unterschiedliche Modelle mit abweichenden Zellgrößen (20 cm, 10 cm, 5 cm und 2,5 cm) und veränderten Randbedingungen (Vorlaufzone, Geschwindigkeitsprofil an der Austrittsöffnung etc.) analysiert. Im Ergebnis konnten weder eine signifikante Strahlaufweitung noch eine quantitative Bestätigung der Gleichung (47) zur Kernlänge nachgewiesen werden. Infolge der langen Rechenlaufzeiten und der in diesem Zusammenhang fehlenden Rechenkapazität zum Zeitpunkt der Untersuchungen war es in [102] nicht möglich, weitere Untersuchungen mit kleineren Zellgrößen durchzuführen.

5.2.3 Verifizierung der GPU-Version

Zunächst wurden die Versuche aus [102] mit der im Rahmen dieser Arbeit neu erstellten GPU-Version der Software FDS nachvollzogen. Im Anlagenteil D.1 sind für drei unterschiedlichen Gitterweiten (10 cm, 5 cm und 2,5 cm) die qualitativen Strömungsbilder sowie das Geschwindigkeitsprofil in der Symmetrieebene dargestellt (vgl. Abb. 63–65, S. 146 f.). Es zeigt sich, dass mit der GPU-Version und der Basissoftware sowohl bei doppelter als auch bei einfacher Genauigkeit (bin64/bin32) nahezu identische Ergebnisse erzielt werden. Auch ist ersichtlich, dass sich bei Gitterweiten bis in Höhe von 2,5 cm tatsächlich kein turbulenter Freistrah entsprechend den Erwartungen aus den experimentellen Untersuchungen einstellt. Das Strömungsbild zeigt für alle Simulationsrechnungen einen weitgehend laminaren Strömungsfaden, der nur im Falle der 2,5 cm-Auflösung nach einer Strecke von ca. 10 m hinter der Austrittsöffnung abreißt und in einen turbulenten Zustand übergeht.

Unter Verwendung der massiv beschleunigten GPU-Version wurden in dieser Arbeit zusätzlich Simulationsrechnungen mit kleineren Gitterweiten durchgeführt (vgl. Abb. 66–68, S. 147 f.). Die Ergebnisse aller relevanten Berechnungsläufe sind in Abb. 34 zusammenfassend dargestellt. Es zeigt sich, dass die Kernlänge mit kleiner werdender Gitterweite zwar abnimmt, jedoch auch bei sehr feinen Gitterweiten immer noch erheblich über den experimentell ermittelten Werten liegt. Entsprechend Gleichung (47) wären unter Annahme einer Mischzahl in Höhe von $m = 0,17$ bis $0,20$ Werte im Bereich des Graphen „experimentelle Untersuchungen“ zu erwarten (vgl. Abb. 34).

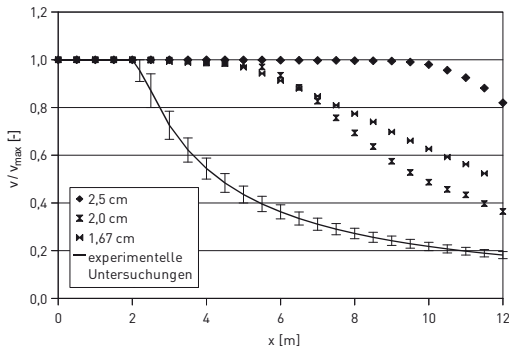


Abb. 34: Geschwindigkeitsverlauf in der Symmetrieachse gemittelt zwischen $t = 120$ s und 150 s für unterschiedliche Gitterweiten, Modellgeometrie A

Die Ursache der fehlenden Übereinstimmung liegt vor allem in der Art der Einströmöffnung begründet. Das einströmende Fluid weist bei Anordnung des gewählten Einlasses (Typ: VENT) eine ideale laminare Strömung auf, die in der Realität nicht auftritt. Wird anstelle der Einströmöffnung ein „Vorlaufrohr“ angeordnet, so ergibt sich neben einer Anfangsturbulenz auch ein realistischeres Geschwindigkeitsprofil an der Austrittsöffnung. Eine entsprechend modifizierte Modellgeometrie ist in der Abb.35 dargestellt. Neben dem realitätsnahen Geschwindigkeitsprofil an der Auslassöffnung weist der Strahl in diesem Modell auch bereits an der Auslassöffnung eine gewisse „Anfangsturbulenz“ auf. Diese ist bei dem Modell aus [102] (Anordnung eines Auslasses vom Typ „VENT“) nicht vorhanden³.

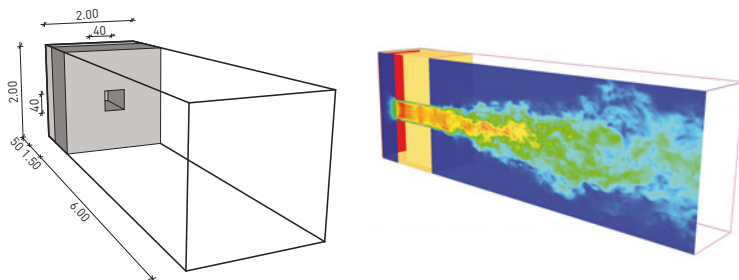


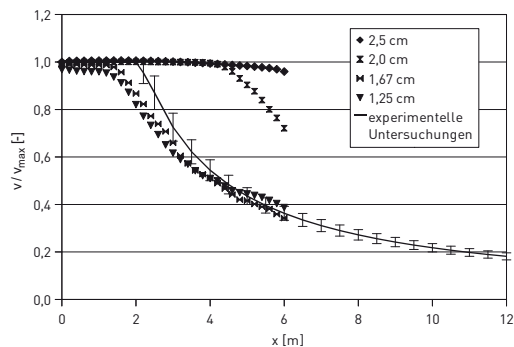
Abb. 35: Modellgeometrie B (links), Geschwindigkeitsprofil in der Symmetrieebene für einen exemplarisch ausgewählten Berechnungszeitpunkt (rechts)

³ Im Rahmen dieser Arbeit wurden unterschiedliche Modelle zum Freistrahls untersucht. Die genauesten Ergebnisse wurden mit dieser Modellvariante erzielt.

Auch in diesem Modell zeigt sich bei den mit groben Gitterweiten durchgeführten Vergleichsrechnungen eine sehr gute Übereinstimmung zwischen der CPU- und GPU-Version (vgl. Anlagenteil D.2, Abb. 69). Die ausschließlich mit der GPU-Version der Software erstellten Simulationsrechnungen auf feineren Gitterweiten sind in den Abbildungen 70 bis 72 (vgl. S. 149 f.) dargestellt. Zusätzlich enthält die Abb. 36 eine Zusammenfassung der Ergebnisse für die Modellgeometrie B.

Aus der qualitativen Darstellung des Geschwindigkeitsprofils in der Symmetrieebene (vgl. farbige Skizze in Abb. 72, S. 150) wird ersichtlich, dass bei ausreichend kleinen Gitterweiten die Einmischung des Sekundärfluids sowie die damit verbundene Strahlaufweitung – wie experimentell ermittelt – bereits kurz nach der Austrittsöffnung beginnt. Auch die über die Zeit gemittelten Kernlängen stimmen nun wesentlich besser mit den experimentellen Ergebnissen überein. Voraussetzung für diese Übereinstimmung ist jedoch, dass eine ausreichend kleine Gitterweite im Modell zur Verfügung steht. Für das hier betrachtete Beispiel (isothermer Freistrah mit $Re > 8.500\text{--}10.000$) müssen an der Austrittsöffnung mindestens zwölf Zellen je Kantenlänge vorhanden sein, um mit der Software FDS ein akzeptables Ergebnis zu erzielen (vgl. Abb. 36, Zellgröße $x \leq 1,67\text{ cm}$).

Abb. 36: Geschwindigkeitsverlauf in der Symmetriechse gemittelt zwischen $t = 120\text{ s}$ und 150 s für unterschiedliche Gitterweiten, Modellgeometrie B



Im Hinblick auf die verwendete Genauigkeit ist festzustellen, dass die Ergebnisse zwischen einfacher und doppelter Genauigkeit (bin64/bin32) lediglich bei den groben Gitterweiten übereinstimmen. Werden feinere Gitter verwendet, führt eine Berechnung mit einfacher Genauigkeit zu numerischen Instabilitäten. Die Ursache dieser Instabilitäten wurde im Rahmen dieser Arbeit nicht geklärt.

Die in den Simulationsrechnungen beobachtete Übereinstimmung der Ergebnisse zwischen CPU- und GPU-Version setzt den Einsatz von GPU-Karten mit der Spezifikation „sm_20“ voraus (vgl. Abs. 3.2). Im Anlagenteil D.2 in Abb. 73, S. 151 sind die Ergebnisse für eine Berechnung mit einer Karte vom Typ GeForce GTX 280 (GPU-Spezifikation

„sm_13“) wiedergegeben. Gegenüber der CPU-Version ergeben sich hier signifikante Abweichungen, deren Ursache mit hoher Wahrscheinlichkeit im abweichenden Rundungsverhalten der GPU-Karte zu finden ist. Erst mit der GPU-Spezifikation „sm_20“ wird der IEEE-754-Standard [44] vollständig erfüllt. Ebenso sind Fehler in der Treibersoftware bei den Karten der früheren Generation als Ursache für die abweichenden Ergebnisse nicht auszuschließen.

Die erstellten Simulationsrechnungen decken darüber hinaus ein inkonsistentes Verhalten der Basisversion von FDS für Elemente mit einer Wanddicke von Null auf. Für den Fall, dass das Vorlaufrohr mit Wandelementen der Wandstärke Null modelliert wird (welches entsprechend des Handbuches zulässig ist), ergibt sich ein komplett abweichendes Strömungsbild (s. Abb. 74, S. 151). Der hier vorliegende Fehler im Quellcode von FDS wurde im Rahmen dieser Arbeit nicht weiter verfolgt.

5.2.4 Rechenleistung / Performance

Um die Steigerung der Rechenleistung der neu entwickelten GPU-Version gegenüber der Basisversion beurteilen zu können, wurde die benötigte Rechenzeit für die Lösung eines exemplarisch ausgewählten Modells (Modellgeometrie A, Gitterweise $2,5 \times 2,5 \times 2,5$ cm, $t_{\max} = 150$ s) auf unterschiedlichen Rechnersystemen ermittelt. Neben den Rechenzeiten sind in der Tabelle 6 auch die Beschleunigungsfaktoren in Bezug auf den derzeit schnellsten, auf dem Markt verfügbaren CPU-Prozessor (Intel Core i7 980X) dargestellt.

Prozessorarchitektur		Rechenzeit	Beschleunigungsfaktor
CPU	AMD Phenom II X4 940, 1 core	37,14 h	0,4 x
	Intel Core i7 980X, 1 core	14,86 h	1,0 x
GPU (Fermi)	Fermi Tesla C2070, bin64 (ECC)	43,2 (54,0) min	20,6 x (16,5 x)
	Fermi Tesla C2070, bin32 (ECC)	29,8 (38,9) min	29,9 x (22,9 x)
	GTX 580, bin64	30,3 min	29,4 x
	GTX 580, bin32	19,5 min	45,7 x

Tabelle 6: Analyse der Rechenleistung für die Modellgeometrie A ($2,5 \times 2,5 \times 2,5$ cm), Werte in Klammern beziehen sich auf Simulationsrechnungen mit eingeschalteter ECC-Funktion

Es zeigt sich, dass die Berechnung für den strömungsmechanischen Teil der Software FDS durch die neu entwickelte GPU-Version etwa um den Faktor 30 (doppelte Präzision) bzw. um den Faktor 45 (einfache Präzision) beschleunigt werden kann.

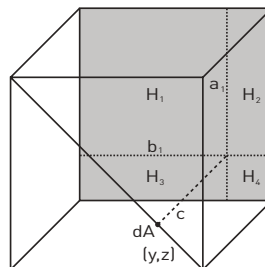
5.2.5 Zusammenfassung

Zusammenfassend ist festzustellen, dass die Ergebnisse aus der neu entwickelten GPU-Version und der Basisversion für den in diesem Abschnitt simulierten Niederdruck-Freisstrahl in hohem Maße übereinstimmen. Dabei ist es für größere Gitterweiten unerheblich, ob mit einfacher oder doppelter Genauigkeit gerechnet wird. Bei kleinen Zellgrößen treten bei Verwendung der einfachen Genauigkeit numerische Instabilitäten auf, deren Ursachen im Rahmen dieser Arbeit nicht geklärt wurden. Die Übereinstimmung beider Berechnungen setzt im Weiteren die Verwendung von GPU-Karten mit der Spezifikation „sm_2x“ voraus (vgl. Abs. 3.2). Erst bei dieser Kartengeneration wird der IEEE-754-Standard [44] vollständig erfüllt. Sofern für die Berechnung Karten mit der GPU-Spezifikation „sm_1x“ verwendet werden, sind zumindest in diesem Beispiel fehlerhafte Ergebnisse zu erwarten. Die Berechnung mit der GPU-Version (GPU-Prozessor: GeForce GTX 580) läuft in Bezug auf die Basisversion (Referenzprozessor: Intel Core i7 980x) in diesem Beispiel etwa um Faktor 30 (doppelte Genauigkeit) bzw. um Faktor 45 (einfache Genauigkeit) schneller ab.

5.3 Beispiel 2: Strahlung in einer Box

Zur Verifizierung des neuen Strahlungslösers (vgl. Kapitel 4.5) werden in diesem Abschnitt die Strahlungsaustauschfaktoren Φ (configuration factor) in einer würfelförmigen Box mit einer heißen und fünf kalten Innenwänden (Null Kelvin) ermittelt. Der Aufbau der Box entspricht im Wesentlichen dem Beispiel in [48] (s. Abb. 37).

Abb. 37: Geometrischer Aufbau der Beispielbox mit einer heißen (grau) und fünf kalten Innenwänden (transparent)



Da sich die oben beschriebene Problemstellung auf ein graues Gas ohne Einwirkung von Strahlungsquellen innerhalb des Lösungsgebietes (z.B. Flammenbildung) bezieht, können mit diesem Beispiel nicht alle in FDS enthaltenen Algorithmen zum Strahlungswärmetransport getestet werden. Die grundlegende Funktionsweise der in Kapitel 4.5 entwickelten Algorithmen kann jedoch in ausreichendem Maße überprüft werden.

5.3.1 Theoretische Grundlagen

Betrachtet werden im Folgenden ausschließlich die Flächenelemente dA auf der Wand, die der heißen Innenwand direkt gegenüber liegen. Der Faktor Φ_{Hi} für das Flächenelement dA , dass sich gegenüber der Ecke einer strahlenden Teilfläche H_i befindet, lässt sich auf analytischem Wege nach [61], Anhang C wie folgt ermitteln:

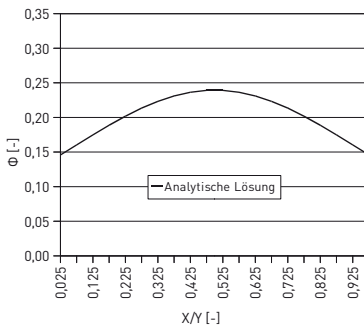
$$\Phi_{Hi} = \frac{1}{2\pi} \left[\frac{X_i}{\sqrt{1+X_i^2}} \arctan\left(\frac{Y_i}{\sqrt{1+X_i^2}}\right) + \frac{Y_i}{\sqrt{1+Y_i^2}} \arctan\left(\frac{X_i}{\sqrt{1+Y_i^2}}\right) \right] \quad (49)$$

mit $X_i = a_i/c$ und $Y_i = b_i/c$

Der Faktor Φ_{HdA} für das Flächenelement dA in Bezug auf die gesamte strahlende Innenwand berechnet sich aus der Summe der Einzelfaktoren wie folgt:

$$\Phi_{HdA} = \sum_{i=1}^4 \Phi_{Hi} \quad (50)$$

Für eine Box mit der Einheitskantenlänge ist die analytische Lösung auf der Diagonalen in Abb. 38 wiedergegeben.



(X/Y)	Φ_{HdA}	(X/Y)	Φ_{HdA}
0,025	0,1457	0,275	0,2135
0,075	0,1603	0,325	0,2233
0,125	0,1748	0,375	0,2311
0,175	0,1888	0,425	0,2364
0,225	0,2018	0,475	0,2391

Abb. 38: Analytisch ermittelte Faktoren Φ_{HdA} für einen Punkt auf der Diagonalen gegenüber der strahlenden Innenwand

5.3.2 Verifizierung der GPU-Version

Zur Verifizierung der Algorithmen wurden zwei Modelle mit unterschiedlicher Raumzellen- und Kontrollwinkelanzahl nachgerechnet. Die in Abb. 39 dargestellten Diagramme zeigen den Verlauf des Faktors Φ_{HdA} über die Diagonale für die Modelle mit 12.000 und 1,1 Mio. Gitterzellen und jeweils 50 bzw. 1.000 Kontrollwinkel.

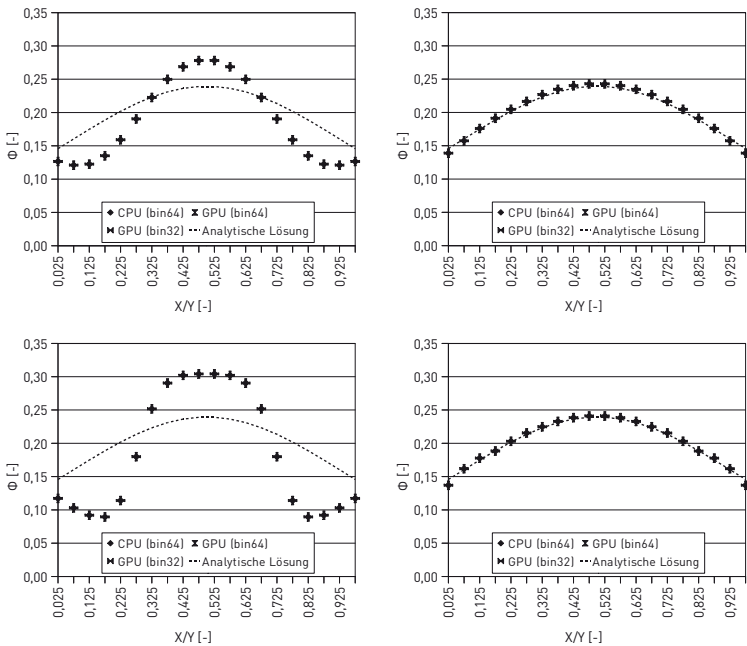


Abb. 39: Modell mit 12.000 Gitterzellen und 50 Kontrollwinkel (o. links) bzw. 1.000 Kontrollwinkel (o. rechts) sowie mit 1,1 Mio. Gitterzellen und 50 Kontrollwinkel (u. links) bzw. 1.000 Kontrollwinkel (u. rechts)

In jedem Diagramm sind neben der analytischen Lösung die Ergebnisse aus der Berechnung mit der Basisversion sowie der neuen GPU-Version dargestellt. Es zeigt sich, dass eine Übereinstimmung zwischen analytischer und numerischer Lösung bei einer hinreichenden Anzahl von Gitterzellen bzw. Kontrollwinkel erreicht wird. Darüber hinaus stimmen auch für Lösungsgebiete mit wenig Gitterzellen die Ergebnisse aus der neuen GPU-Version mit den Werten aus der Basisversion von FDS in allen Modellen überein. Aus den Ergebnissen wird ebenfalls ersichtlich, dass hier auch mit einfacher Genauigkeit gerechnet werden kann, ohne die Ergebnisse zu verfälschen.

5.3.3 Rechenleistung / Performance

Um die Steigerung der Rechenleistung der neu entwickelten GPU-Version gegenüber der Basisversion beurteilen zu können, wird die Leistungssteigerung (SpeedUp) analog zum Beispiel in Abschnitt 5.2.4 unter Verwendung unterschiedlicher Hardwaresysteme ermittelt (s. Tab. 7). Zusätzliche Angaben zur Rechenleistung sowie eine ausführliche Diskussion der Ergebnisse zur Leistung der neuen Wellenfrontalgorithmen für den Strahlungswärmetransport sind darüber hinaus im Abschnitt 4.5.6 enthalten.

Prozessorarchitektur		Rechenzeit	Beschleunigungsfaktor
CPU	AMD Phenom II X4 940, 1 core	33,70 s	0,4x
	Intel Core i7 980X, 1 core	13,45 s	1,0x
GPU (Fermi)	Fermi Tesla C2070, bin64 (ECC)	1,90 (2,26) s	7,1x (6,0x)
	Fermi Tesla C2070, bin32 (ECC)	1,75 (2,10) s	7,7x (6,4x)
	GTX 580, bin64	1,26 s	10,7x
	GTX 580, bin32	1,19 s	11,3x

Tabelle 7: Rechenleistung für ein Modell mit 126 x 120 x 120 Gitterzellen und 104 Kontrollwinkel, Werte in Klammern beziehen sich auf Simulationsrechnungen mit eingeschalteter ECC-Funktion

Teil B

Simulation von Spill Plumes in Atrien

6 Numerische Simulation von Spill Plumes in Atrien

In diesem Abschnitt wird eine Parameterstudie zur Lufteinmischung in einen aufsteigenden Überlaufplume unter Verwendung der neu entwickelten GPU-Version der Software FDS durchgeführt. Zu diesem Zweck wird ein Computermodell erstellt, welches anschließend auf der Grundlage von experimentellen Untersuchungen (vgl. Abs. 6.1–6.4) bzw. bekannten Näherungsformeln aus der Literatur (vgl. Abs. 6.5) validiert wird. Im Anschluss werden unterschiedliche Parameter variiert (Geometrie des Atriums, Öffnungsbreite des Brandraumes, Wärmefreisetzungsrate etc.), um das Verhalten des Plumes unter bestimmten Randbedingungen beurteilen und geeignete Näherungsformeln für das Szenario entwickeln zu können (vgl. Abs. 6.6 und 6.7).

6.1 Zusammenfassung der Versuche nach Harrison [10] und [4]

6.1.1 Versuchsaufbau und Versuchsdurchführung

Harrison verwendet in [10] ein Modell im Maßstab 1:10, welches aus zwei Teilen besteht (s. Abb. 40a). Der auf dem Fußboden stehende Brandraum (fire compartment) besteht aus hochtemperaturbeständigen keramischen Faserplatten (CFI-Platten, $d = 20 \text{ mm}$) auf deren Außenseite Stahlbleche ($d = 2 \text{ mm}$) montiert sind. Der Brandraum weist einseitig eine Öffnung auf, an deren Oberseite ein 30 cm breiter Balkon (balcony) bzw. an deren Seiten zwei Rauchschränke (channeling screens) aus 10 mm dicken CFI-Platten angebracht sind. Die Rauchschränke sorgen für eine homogene Ausbreitung der Rauchgasschicht und wurden im Modell bis zum Fußboden geführt. Der zweite Teilbereich des Experiments besteht aus einer über dem Boden angebrachten Rauchabzugshaube, die aus einem beplankten Stahlgerüst zusammengesetzt wurde (s. Abb. 40b). Die Seiten der Rauchabzugshaube wurden ebenfalls aus 10 mm dicken CFI-Platten gefertigt, wobei eine Seitenwand zur Einsicht wahlweise mit Plexiglas ausgetauscht werden kann. Auf der Oberseite der Rauchabzugshaube befindet sich eine maschinelle Entlüftung, deren Volumenstrom über die Drehzahl des Lüfters geregelt wird. Als Feuerquelle wurde eine Metallschale im Brandraum positioniert, in die eine kontrollierte Menge Flüssigbrennstoff (IMS) eingeleitet werden kann (s. Abb. 40c). Die Metallschale wurde zur Rückseite geneigt aufgestellt, so dass sich die Fläche des Brandherdes entsprechend dem Zufluss des Brennstoffes automatisch regelt. Für weitere Details zum Versuchsaufbau wird auf die Beschreibung in [4] verwiesen.

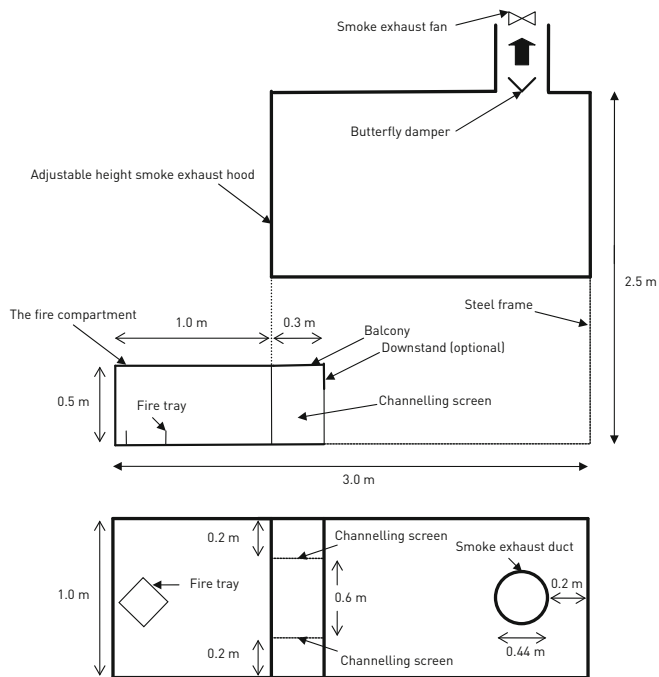


Abb. 40: a) Versuchsgeometrie (oben), b) Überblick über die gesamte Versuchsanordnung (rechts unten), c) Blick in den Brandraum (links unten)

Harrison führt insgesamt 64 Experimente mit unterschiedlichem Volumenstrom am maschinellen Rauchabzug durch, wobei die einzelnen Experimente in neun Versuchsreihen eingeordnet werden können. In den ersten drei Versuchsreihen wird lediglich die Wärmefreisetzungsrate des Brandherdes variiert. Für die übrigen Versuchsreihen wird im Bereich des Überlauffrandes ein Unterzug angeordnet (vgl. Abb. 40a, „downstand“), so dass der Effekt eines derartigen Bauteils auf die Strömung untersucht werden kann. Die Randbedingungen der neun Testgruppen sind in Tabelle 8 zusammengefasst.

Test	Wärmefreisetzungsrate	Unterzughöhe
1	6 kW	0 cm
2	9 kW	0 cm
3	12 kW	0 cm
4	6 kW	10 cm
5	9 kW	10 cm
6	12 kW	10 cm
7	6 kW	20 cm
8	9 kW	20 cm
9	12 kW	20 cm

Tabelle 8: Zusammenfassung der Versuchsreihen aus [4]

In jedem Experiment werden die Gastemperaturen innerhalb der Rauchabzugshaube über Temperaturfühler sowie unter dem Überlauffrand über einen bestimmten Zeitraum gemessen und gemittelt. Im Bereich des Überlauffrandes sind zusätzlich Staudruckrohre (Pitotrohre) angeordnet, mit denen ein Geschwindigkeitsprofil erstellt wird. Aus den Temperatur- und Geschwindigkeitsprofilen bestimmt Harrison den Massenstrom sowie den konvektiven Wärmestrom über eine EXCEL-Tabelle. Die Höhe der Rauchgasschicht im Atrium ermittelt Harrison vor allem über visuelle Beobachtungen. Zur Kontrolle berechnet der Autor zusätzlich eine „rechnerische Schichthöhe“ über das Temperaturprofil im Atrium nach Thomas [104]. Für detaillierte Angaben zu den Messfühlern sowie zur Ermittlung der relevanten Größen wird auf [4] verwiesen.

Bei Verwendung des durch Harrison untersuchten verkleinerten Modells (Maßstab 1:10) ist dafür Sorge zu tragen, dass sich analog zum tatsächlichen Brandgeschehen eine vollständig turbulente Strömung ausbilden kann. Damit werden Änderungen der Strömungsvorgänge infolge des veränderten Maßstabes ausgeschlossen. Nach [105] kann diese Voraussetzung als gegeben angenommen werden, wenn die Reynolds-Zahl die Bedingung $Re > 4.000$ erfüllt. In [4] werden für Reynolds-Zahlen der einzelnen Testfälle Werte zwischen $9.600 < Re < 25.800$ ermittelt, so dass die Anwendung des verkleinerten Modells

zulässig ist. Die tatsächlichen Werte für den Wärme-, Massen und Volumenstrom (Q , M , V), die Geschwindigkeit (u) bzw. die Zeitskala (ω) ergeben sich aus den im Modell gemessenen Größen über die folgenden Beziehungen:

$$Q \sim L^{5/2}; \quad M \sim L^{5/2}; \quad V \sim L^{5/2}; \quad u \sim L^{1/2}; \quad \omega \sim L^{1/2} \quad (54)$$

Die in [4] dargestellte Herleitung dieser Beziehungen basiert auf der Annahme einer vollständig turbulenten, auftriebsgetriebenen Strömung. Die in den einzelnen Versuchsreihen verwendeten Wärmefreisetzungsraten $Q_{1/10} = 6 \text{ kW}$, 9 kW bzw. 12 kW entsprechen damit in der Praxis (Maßstab 1:1) Brandherden mit $Q_{1/1} = 1,9 \text{ MW}$, $2,8 \text{ MW}$ bzw. $3,8 \text{ MW}$.

6.1.2 Ergebnisse aus den Versuchsreihen

Harrison ermittelt in [4] für jede der neun Versuchsreihen ein Temperatur- und Geschwindigkeitsprofil direkt unterhalb des Überlaufandes (vgl. Anlagenteil E, S. 153–158). Darüber hinaus gibt der Autor den zugehörigen konvektiven Wärmestrom $Q_{b,ex}$ (vgl. Tab. 9, S. 98), den Massenstrom der Rauchgase $M_{b,ex}$ (vgl. Tab. 10, S. 99) und die Schichthöhe $d_{b,ex}$ (vgl. Tab. 10) an. Die von Harrison ermittelte Lufteinmischung in den aufsteigenden Plume innerhalb der Rauchabzugshaube M_a ist in Abhängigkeit von der Höhe z im Anlagenteil E auf den Seiten 161 bis 163 dargestellt („Harrison (M_a)“). Je Versuchsreihe ermittelt Harrison vier bis sechs Datenpunkte, die den Zusammenhang zwischen der Höhe der Rauchgasschicht im Atrium und dem Massenstrom an der Rauchabzugsöffnung charakterisieren.

6.1.3 Diskussion der Ergebnisse

Ausgehend von der durch Thomas et. al. [35] vorgeschlagenen Gleichung (17) ist ein linearer Zusammenhang zwischen dem Term $z / Q^{1/3}$ und der Lufteinmischung M'_a/Q' zu erwarten. Werden alle im Rahmen der Versuche ermittelten Werte in ein Diagramm eingetragen, ergibt sich der Parameter α aus dem Anstieg der Regressionsgeraden (s. Abb. 41). Die Parameter β und γ können aus dem Schnittpunkt der Geraden mit der Y-Achse bestimmt werden.

Wie Abb. 41 zeigt, beeinflusst der Unterzug am Überlaufand die Ergebnisse nur dann, wenn dieser eine ausreichende Tiefe aufweist (Versuchsreihen 7, 8 und 9 mit 20 cm Unterzughöhe). In diesem Fall liegt die gemessene Lufteinmischung tendenziell etwas unter den übrigen Werten. Davon ausgehend, dass dieser Effekt von untergeordneter Bedeutung ist, erhält Harrison in [4] den folgenden Zusammenhang:

$$M'_a = 0,20 z Q^{1/3} + 1,50 M'_b + 0,0017 Q' \quad (52)$$

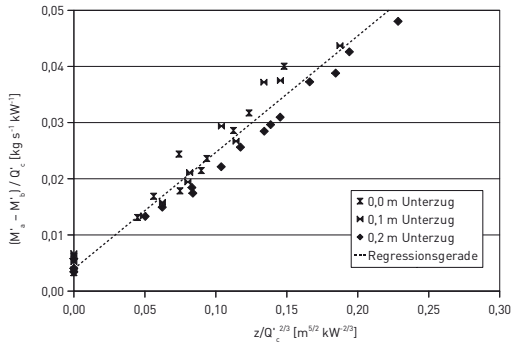


Abb. 41: Ermittlung der Einmischungskonstante α durch Bildung einer Regressionsgeraden, Versuchsergebnisse nach Harrison [4]

In [10] wertet der Autor seine Versuchsergebnisse erneut aus und berücksichtigt dabei ausschließlich die Versuchsreihen ohne oder mit kleinem Unterzug (Versuchsreihen 1 bis 6). Dadurch ergibt sich eine um 10 % höhere Einmischungskonstante $\alpha = 0,22$ (vgl. Gl. (21)). Die in Bezug auf die Angaben von Thomas (vgl. Gl. (18)) ermittelte größere Einmischungskonstante ($0,22 > 0,16$) resultiert vor allem aus der zusätzlichen Einmischung in die freien Enden des Plumes (3D-Plume). Thomas et al. betrachten in [35] einen zweidimensionalen Plume ohne Lufteinmischung in die freien Enden.

In diesem Zusammenhang ist noch zu erwähnen, dass Harrison für die Auswertung seiner Versuchsergebnisse nicht alle Messwerte heranzieht. So ergibt sich bei großen Höhen z eine ungleichmäßige Schichtung der Rauchgase in der Abzugshaube, die eine genaue Definition der Schichthöhe unmöglich werden lassen. Dieser, in [4] als „plugholing“ bezeichnete Effekt resultiert aus der Störung der gleichmäßigen Rauchgasschicht infolge von zu großem Volumenstrom im Rauchgasabzug.

6.2 Beschreibung des Computermodells

Auf der Grundlage der in [4] beschriebenen Randbedingungen wurden im Rahmen dieser Arbeit drei Computermodelle erstellt, die sich durch ihre Gitterweiten voneinander unterscheiden ($\Delta l = 2,0 / 1,25 / 1,0 \text{ cm}$). Neben den geometrischen Randbedingungen werden auch die stofflichen Kennwerte – soweit bekannt – in das Computermodell übernommen. Entsprechend den Vorgaben in [4] wird für die CFI-Platten die Wärmeleitfähigkeit mit $\lambda = 0,041 \text{ W/mK}$, die Wärmespeicherkapazität mit $c = 2,09 \text{ kJ/(kg} \cdot \text{K)}$, die Dichte mit $\rho = 229 \text{ kg/m}^3$ und ein Emissionskoeffizient in Höhe von $\alpha = 0,9$ angesetzt. Auf der Außenseite der CFI-Platten werden in Übereinstimmung mit [4] 2 mm dicke Stahlbleche ($\lambda = 45,8 \text{ W/mK}$, $c = 0,46 \text{ kJ/(kg} \cdot \text{K)}$, $\rho = 7850 \text{ kg/m}^3$, $\alpha = 0,45$) modelliert. Für den Fußboden wird vereinfacht ein adiabatischer Rand angesetzt. Diese Annahme hat keinen signifikanten Einfluss auf die Ergebnisse.

Die Entrauchung des Atriums bzw. der Rauchabzugshaube wird – abweichend vom Versuchsaufbau – über Entlüftungsöffnungen variabler Größe sichergestellt (vgl. Abb. 42). Diesbezüglich durchgeführte Vergleichsrechnungen ergaben, dass sich gegenüber einer modellierten maschinellen Entlüftung (vgl. Abb. 40 a, „smoke exhaust fan“) ein gleichmäßigeres Strömungsbild im Atrium einstellt und der in [4] beschriebene unerwünschte Effekt des „plugholing“ weitgehend vermieden wird.

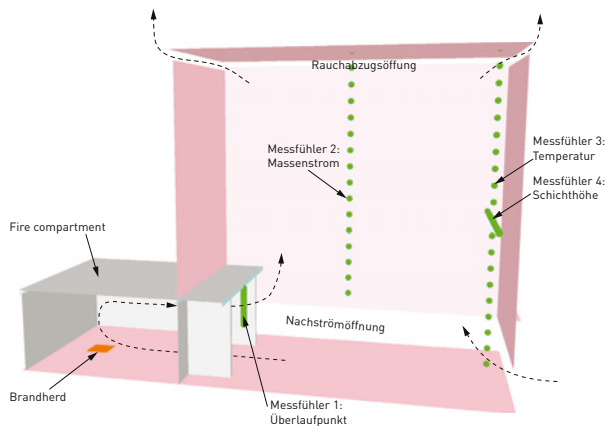


Abb. 42: Aufbau des Computermodells (Prinzipiskizze)

Als Brandherd wird ein Methangasbrenner mit konstanter Wärmefreisetzungsrate modelliert (Mischungsbruch-Modell ohne CO-Produktion, vgl. Abs. 4.1). Die Fläche des Brandherdes hat Einfluss auf die Flammenhöhe und damit letztendlich auch auf die Rauchgasströmung am Überlaufbrand. Da die Fläche des Brandherdes in [4] variabel gehalten ist (Zustrom des Flüssigbrennstoffes in eine geneigte Metallschale) und vom Autor auch nicht dokumentiert wird, werden im Computermodell Fläche und Wärmefreisetzungsrate des Brandherdes iterativ derart gewählt, dass die gewünschte Rauchgasströmung am Überlaufbrand modelliert werden kann (s. Validierung in Abs. 6.3).

Zur Beschleunigung der Simulationsrechnungen ist es denkbar, den Wärmetransport über Strahlung in der Berechnung zu vernachlässigen. In diesem Fall würde der vergleichsweise schlecht parallelisierbare Strahlungslöser nicht benötigt. Wie Vergleichsrechnungen zeigen, ergeben sich in diesem Fall jedoch nicht zu vernachlässigende Abweichungen in der Rauchgasströmung, so dass auf die Modellierung des Strahlungswärmetransports nicht verzichtet werden kann. Bei allen Modellrechnungen kommt daher das im Abschnitt 4.5 beschriebene erweiterte KBA-Verfahren zum Einsatz. Der

Anteil der bei der Verbrennung freiwerdenden Strahlungswärme ξ_r (vgl. Gl. (41)) kann für die hier betrachtete Fragestellung prinzipiell frei gewählt werden, solange der konvektive Wärmestrom am Überlaufrand dem experimentell gemessenem Wert entspricht (s. Validierung im Abs. 6.3). In dieser Arbeit wird modellübergreifend von einem pauschalen Anteil der freiwerdenden Strahlungsenergie in Höhe von $\xi_r = 10\%$ ausgegangen.

6.2.1 Anordnung von virtuellen Messfühlern

Das Computermodell verfügt über vier unterschiedliche Gruppen von virtuellen Messfühlern (vgl. Abb. 42). Mit der Messfühlergruppe 1 werden die Temperatur- und Geschwindigkeitsprofile an der Austrittsebene unterhalb des Überlaufrandes ermittelt. Dabei ist zu berücksichtigen, dass die Software FDS alle skalaren Größen in Zellmitte und alle übrigen Variablen an den Rändern der Kontrollvolumen speichert (staggered grid). Während damit die Geschwindigkeiten am Überlaufrand (spill edge) direkt auslesbar sind, müssen die Temperaturen aus zwei benachbarten Zellen separat berechnet und gemittelt werden. Neben den Temperatur- und Geschwindigkeitsprofilen werden darüber hinaus der Volumen-, Massen- und Wärmestrom von und in den Brandraum durch Summenbildung über die gesamte Öffnungsfläche berechnet.

Mit den Messfühlern der Gruppe 2 wird der Massenstrom innerhalb des Atriums als Summe aller Gitterzellen in einer horizontalen Ebene aufwärts (M_a^+) und abwärts (M_a^-) getrennt bestimmt. Da die gesuchte Massenstromdifferenz ($M_a = M_a^+ - M_a^-$) innerhalb des Atriums prinzipiell unabhängig von der Ebenenhöhe z ist, reicht hierfür theoretisch die Anordnung eines einzelnen Messfühlers in beliebiger Höhe aus. Im Rahmen der weitergehenden Untersuchungen werden jedoch auch Massenströme ohne Rauchabzugshaube (freier Plume) ermittelt (vgl. Abs. 6.4), so dass hier mehrere Messfühler mit variabler Höhe im Modell vorhanden sind.

Bei der Ermittlung der oben aufgeführten Massenstromdifferenz M_a wird der Massenstrom im Modell unabhängig von der Art der Gase bestimmt, d. h. es werden neben dem Rauchgas im Plume auch die Massenströme der Frischluft in die Berechnung mit einbezogen. Soll nur der Massenstrom im Rauchgas ermittelt werden, so muss sich die Summenbildung ausschließlich auf die Zellen beschränken, in denen tatsächlich Rauchgas vorhanden ist. Da eine derartige Funktion in der Software FDS nicht integriert ist, wurde ein entsprechender Quellcodeabschnitt ergänzt. Durch Angabe eines stoffspezifischen Grenzwertes in der Eingabedatei werden bei der Summenbildung lediglich die Zellen berücksichtigt, deren Massenanteil oberhalb dieses Schwellenwertes liegen (z. B. CO_2 -Gehalt $> 0,1\%$). Der Massenstrom der Rauchgase im aufsteigenden Plume lässt sich über diese neue Funktion durch Subtraktion der auf- und absteigenden Rauchgase ermitteln: ($M_p = M_p^+ - M_p^-$).

Mit den Messfühlergruppen 3 und 4 wird die Höhe der rauchgasführenden Schicht im Atrium bestimmt. Dazu wird ein Mittelwert aus zehn horizontal angeordneten Messfühlern gebildet, die ihrerseits die Höhe der Rauchgasschicht über eine programminterne Funktion berechnen (Messfühlergruppe 4). Dabei wird je Messfühler softwareseitig ein Temperaturprofil $T(z)$ zwischen dem Fußboden $z = 0$ und der Atriumdecke $z = H_a$ erstellt. Anschließend wird die Höhe der rauchgasfreien Schicht im Atrium wie folgt bestimmt:

$$z_{\text{int}} = \frac{T_1(I_1 I_2 - H^2)}{I_1 + I_2 T_1^2 - 2 T_1 H} \quad (53)$$

mit $I_1 = \int_0^H T(z) dz$, $I_2 = \int_0^H 1/T(z) dz$ und $T_1 = T(0 \text{ m})$

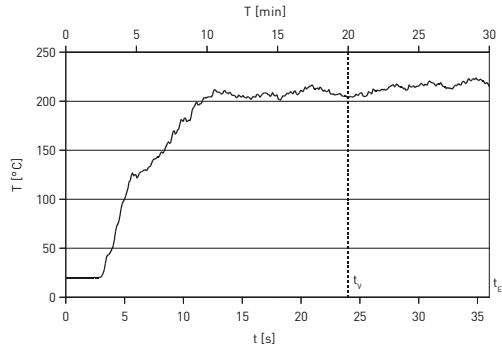
Weitere Angaben zum verwendeten Verfahren sind in [56] und [106] enthalten. Zur Kontrolle dieser rechnerischen Ermittlung der Schichthöhe wird das Temperaturprofil in der Rückwand des Atriums zusätzlich manuell aufgezeichnet (Messfühlergruppe 3).

6.2.2 Vorlaufzeiten für die stationäre Problemstellung

Die Messwerte aller Messfühler werden über eine vorgegebene Zeitspanne gemittelt ausgegeben. Dabei ist vor Beginn der Messung eine ausreichende Vorlaufzeit einzuplanen, in der sich der stationäre Zustand in der gewünschten Form ausbilden kann. Da mit größerer Vorlaufzeit auch die Rechenzeit ansteigt, ist es zweckmäßig, die benötigte Vorlaufzeit durch geeignete Maßnahmen zu minimieren.

Für die Messfühlergruppe 1 (Messwerte am Überlaufpunkt) ist vor allem die Temperatur im Brandraum maßgebend. Diese steigt infolge der Wärmespeicherkapazität der Umfassungsbauteile vergleichsweise langsam an, da sich die Wände erst in ausreichendem Maße erwärmen müssen. Insbesondere für die im Maßstab 1:1 erstellten Modelle werden damit 20 und mehr Minuten Vorlaufzeit erforderlich, was bei der Vielzahl der erforderlichen Rechenläufe rechentechnisch nicht zu leisten ist. Die Software FDS bietet jedoch für stationäre Problemstellungen mit der Variable TSF (TIME_SHRINK_FACTOR) die Möglichkeit, den Prozess des Aufheizens über eine programminterne Reduktion der Wärmespeicherkapazitäten abzukürzen. Dadurch entstehen zwei interne Zeitskalen: t_0 bis t_e für alle strömungsmechanisch relevanten Größen und T_0 bis T_e für den Wärmestrom innerhalb der Umfassungsbauteile. Wie durch Simulationsrechnungen am Modell ermittelt wurde, liefern die Messfühler der Gruppe 1 bei einer Vorlaufzeit von $t_v = 24 \text{ s}$ bzw. $T_v = 20 \text{ min}$ (TSF = 50) und einer anschließenden Messperiode von $t_e - t_v = 12 \text{ s}$ ausreichend genaue Ergebnisse (s. Abb. 43).

Abb. 43: Verlauf der Rauchgastemperatur kurz unter dem Überlauftrand für Versuchsreihe 1 (6 kW)



Auch für die Messfühler der Gruppe 2 ist diese Vorlaufzeit ausreichend, sofern der Massenstrom in einem freien Plume ohne Rauchabzugshaube gemessen wird (vgl. Abs. 6.4). Für die Ergebnisse der Messfühlergruppen 2 bis 4 (Massenstrom im aufsteigenden Plume, Temperaturprofil und Höhe der Rauchgasschicht im Atrium) wird eine Erhöhung der Vorlaufzeit immer dann erforderlich, wenn eine Rauchabzugshaube im Modell vorhanden ist. Dies folgt aus der Tatsache, dass sich zunächst eine Trennschicht zwischen Rauchgas- sowie rauchgasfreier Schicht ausbilden muss (zusätzlich benötigte Zeit für die Ansammlung von Rauchgas im oberen Teil des Atriums). Da sich die benötigte Vorlaufzeit bis zum Erreichen des stationären Zustandes nach dem Volumen der Rauchabzugshaube sowie der jeweiligen Wärmefreisetzungsrate im Brandraum richtet, wird diese in der Parameterstudie (vgl. Abs. 6.7) für jedes Modell separat angepasst. Für die Validierung der Versuche nach Harrison (s. Abb. 40, S. 90) ist eine Vorlaufzeit in Höhe von $t_v = 36$ s und eine anschließende Messperiode in Höhe von $t_e - t_v = 12$ s ausreichend (vgl. Abb. 44).

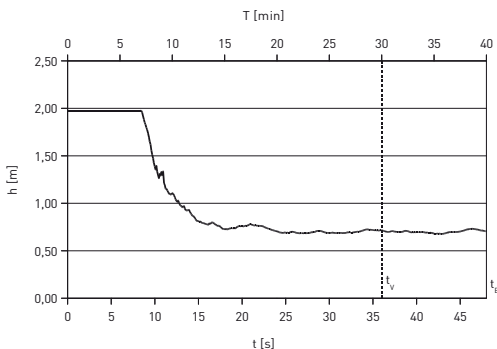


Abb. 44: Messfühlergruppe 4: Höhe der rauchgasfreien Schicht im Atrium für ein exemplarisch ausgewähltes Beispiel aus der Versuchsreihe 1 (6 kW)

6.3 Modellvalidierung: Strömung am Überlauftrand

Für die Parameterstudie zur Rauchgaseinmischung in den aufsteigenden Plume im Atrium ist zunächst der Überlaufplume an sich zu validieren. Hierbei ist sicherzustellen, dass der modellierte Strömungszustand am Überlauftrand mit dem experimentell ermittelten Zustand übereinstimmt. In dieser Arbeit wird daher für jede Versuchsreihe zunächst ein vereinfachtes Modell ohne Rauchabzugshaube erstellt.

Test	Versuch nach [4]		Computermodell		Abweichung
	$Q_{\text{tot,ex}}$ [kW]	$Q_{\text{b,ex}}$ [kW]	Q_{tot} [kW]	Q_{b} [kW]	$Q_{\text{b,ex}}/Q_{\text{b}}$ [%]
1	6,0	4,11	6,3	4,18	+1,7
2	9,0	6,34	10,0	6,27	-1,1
3	12,0	8,74	13,8	8,60	-1,6
4	6,0	4,33	5,3	4,19	-3,2
5	9,0	6,33	7,9	6,12	-3,3
6	12,0	9,1	11,3	8,84	-2,9
7	6,0	4,89	6,0	5,00	+2,2
8	9,0	7,40	9,0	7,43	+0,4
9	12,0	10,37	12,0	10,00	-3,6

Tabelle 9: Iterative Bestimmung der Wärmefreisetzungsrate Q_{tot} zur Minimierung der Abweichung zwischen experimentell ermitteltem, konvektivem Wärmestrom $Q_{\text{b,ex}}$ und dem Wärmestrom im Modell Q_{b}

Für die späteren Untersuchungen ist neben den Temperatur- und Geschwindigkeitsprofilen am Überlaufpunkt vor allem der konvektive Wärmestrom der Rauchgase unterhalb des Überlauftrandes Q_{b} maßgebend. Dieser ergibt sich aus der Wärmefreisetzungsrate Q_{tot} am Brandherd abzüglich der „Verluste“ infolge Wärmestrahlung und Wärmeleitung innerhalb des Brandraumes. Neben der geometrischen Form und den verwendeten Baustoffen ist der tatsächlich messbare konvektive Wärmestrom nicht unwesentlich abhängig von der Flammenhöhe bzw. dem Umfang des Brandherdes. Da diesbezügliche Werte in [4] nicht dokumentiert sind, ist es mit dem Computermodell prinzipiell nicht möglich den konvektiven Wärmestrom Q_{b} am Austrittspunkt in Bezug auf die Wärmefreisetzungsrate Q_{tot} zu validieren. Für das im Rahmen dieser Arbeit verfolgte Ziel (Abschätzung der Lufteinmischung in den Überlaufplume) ist diese Tatsache jedoch unerheblich, sofern die im Versuch gemessenen konvektiven Wärmeströme am Überlaufpunkt in den Modellrechnungen eingehalten werden. Zur Sicherstellung dieses Kriteriums wird die Wärmefreisetzungsrate Q_{tot} für jede untersuchte Gitterweite iterativ derart angepasst, dass der im Versuch gemessene konvektive Wärmestrom mit der Modellrechnung bis auf eine maximale Abweichung in Höhe von 5% übereinstimmt. Die Tabelle 9 zeigt die iterativ bestimmten Wärmeströme exemplarisch für eine Gitterweite in Höhe von $\Delta l = 1,00 \text{ cm}$.

6.3.1 Zusammenfassung der Berechnungsergebnisse

Unter Verwendung der in Tabelle 9 ermittelten Wärmefreisetzungsraten \dot{Q}_{tot} werden die Temperatur- und Geschwindigkeitsprofile am Überlaufpunkt für unterschiedliche Gitterweiten ($\Delta l = 2,00 / 1,25 / 1,00 \text{ cm}$) bestimmt. Die Abb. 45 zeigt hierzu den qualitativen Temperaturverlauf in der Symmetrieebene für zwei exemplarisch ausgewählte Simulationsrechnungen.

Test	Versuch nach [4]			Computermode		
	$M_{b,\text{ex}}$ [kg/s]	$\theta_{b,\text{max},\text{ex}}$ [°C]	$d_{b,\text{ex}}$ [m]	M_b [kg/s]	$\theta_{b,\text{max}}$ [°C]	d_b [m]
1	0,0332 \pm 0,0058	161,7 \pm 3,15	0,105	0,0289	171	0,100
2	0,0388 \pm 0,0130	210,2 \pm 6,58	0,100	0,0328	232	0,110
3	0,0408 \pm 0,0076	258,6 \pm 5,97	0,115	0,0365	271	0,130
4	0,0428 \pm 0,0049	121,8 \pm 2,02	0,125	0,0406	122	0,135
5	0,0483 \pm 0,0064	158,5 \pm 4,32	0,130	0,0454	164	0,145
6	0,0543 \pm 0,0067	200,6 \pm 2,15	0,140	0,0516	203	0,150
7	0,0584 \pm 0,0073	104,2 \pm 2,91	0,130	0,0540	113	0,135
8	0,0682 \pm 0,0083	132,6 \pm 5,13	0,140	0,0605	150	0,140
9	0,0749 \pm 0,0180	166,6 \pm 7,03	0,150	0,0677	180	0,150

Tabelle 10: Gegenüberstellung zwischen den experimentell ermittelten und den berechneten Werten am Überlauftrand ($\Delta l = 2,00 \text{ cm}$)

Die Temperatur- und Geschwindigkeitsprofile unterhalb des Überlauftrandes sind für jede Versuchsreihe separat im Anlagenteil E auf den Seiten 153 bis 158 dargestellt. Neben dem konvektiven Wärmestrom am Überlaufpunkt (vgl. Tab. 9) wurden darüber hinaus der Massenstrom M_b , die Maximaltemperatur $\theta_{b,\text{max}}$ sowie die Höhe der Rauchgasschicht d_b bestimmt. Eine Gegenüberstellung zwischen gemessenen und rechnerisch ermittelten Werten zeigt die Tabelle 10.

Alle in den Diagrammen und Tabellen wiedergegebenen Ergebnisse basieren auf Simulationsrechnungen unter Verwendung der doppelten Genauigkeit (bin64). Im Rahmen der Untersuchungen wurden ebenfalls Rechenläufe mit einer für die einfache Genauigkeit (bin32) kompilierten Version der Software erstellt. Wie bereits im Abschnitt 5.2 festgestellt, zeigten sich jedoch auch hier für Lösungsgebiete mit kleinen Gitterweiten ($\Delta l \leq 1,25 \text{ cm}$) numerische Instabilitäten, so dass mit dieser Programmversion keine aussagekräftigen Ergebnisse zu erzielen waren.

Die zur Modellvalidierung erstellten Computermodelle wurden – analog zu den experimentellen Versuchen – im Maßstab 1:10 erstellt. Da sich die Reynolds-Zahlen für alle Versuche im zulässigen Bereich bewegen (vgl. Abs. 6.1.1), ist bei Verwendung von Modellen

im Maßstab 1:1 keine Veränderung der Ergebnisse zu erwarten. Zur Kontrolle dieses Sachverhalts wurde im Rahmen der Validierung zusätzlich ein Modell im Maßstab 1:1 erstellt. Da die Simulationsrechnung an diesem Modell erwartungsgemäß keine abweichenden Ergebnisse ergibt (die Berücksichtigung der Beziehung (51) vorausgesetzt), wird an dieser Stelle auf die Darstellung der Ergebnisse verzichtet. Die Resultate aus der diesbezüglichen Simulationsrechnung können bei Bedarf der beiliegenden CD-ROM entnommen werden.

6.3.2 Diskussion der Ergebnisse

Wie die Diagramme im Anlagenteil E zeigen, stimmen die berechneten Werte mit den experimentell ermittelten Messgrößen weitgehend überein. Lediglich direkt unterhalb des Überlaufandes (wandnaher Bereich) weichen die gemessenen und berechneten Temperaturen bzw. Geschwindigkeiten voneinander ab. In diesem Bereich gibt die Computersimulation in einigen Versuchsreihen bis zu 15 % höhere Temperaturen aus, während das Geschwindigkeitsprofil direkt unterhalb der Decke abknickt. Erwartungsgemäß können Strömungen im wandnahen Bereich mit dem verwendeten Programm FDS nur näherungsweise abgebildet werden, da die turbulente Strömung in diesem Bereich programmintern über eine Wandfunktion vergleichsweise grob modelliert wird. Für die hier durchgeführten Untersuchungen zur Luftermischung in den aufsteigenden Plume sind die vorhandenen Abweichungen jedoch von untergeordneter Bedeutung, solange der Verlauf in den Temperatur- und Geschwindigkeitsprofilen in den übrigen Bereichen angenähert nachgebildet werden kann und der Massen- bzw. konvektive Wärmestrom übereinstimmen. Diese Voraussetzungen werden durch das Computermodell – unter Berücksichtigung der versuchsbedingten Messgenauigkeiten – erfüllt (vgl. Tab. 9 und 10).

6.4 Modellvalidierung: Massenstrom im aufsteigenden Plume (ohne Atrium)

Obwohl Harrison in [4] ausschließlich Experimente mit einer Rauchabzugshaube durchgeführt hat, soll in diesem Abschnitt zunächst die Luftermischung in den aufsteigenden Plume für das Modell ohne Rauchabzugshaube ermittelt werden. Im Gegensatz zum Experiment ist es im Computermodell möglich, den Massenstrom in unterschiedlichen Höhen direkt zu messen, so dass ein Rauchabzug (vgl. Abb. 40 a, S. 90, „smoke exhaust fan“) nicht zwingend erforderlich ist. Es kann davon ausgegangen werden, dass ein derartiges Modell in etwa die Situation in sehr großen Atrien widerspiegelt.

Für die Validierung der Luftermischung in den aufsteigenden Plume ohne Atrium wird das in Abschnitt 6.2 beschriebene Modell in vereinfachter Form (ohne Rauchabzugshaube) verwendet. Darüber hinaus werden in einer Variante die freien Enden des Plumes durch zusätzliche Wände gesperrt (s. Abb. 46, rechts), so dass ein zweidimensionaler

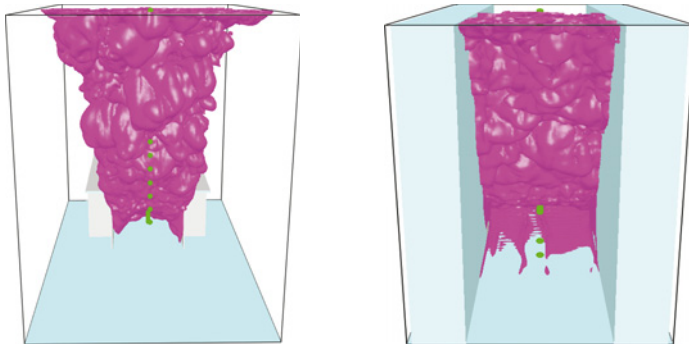


Abb. 46: Vereinfachtes Modell ohne Atrium: Aufsteigender Plume mit (links) und ohne (rechts) Einmischung in die Enden (Grenzfläche: CO₂-Gehalt = 0,01 %)

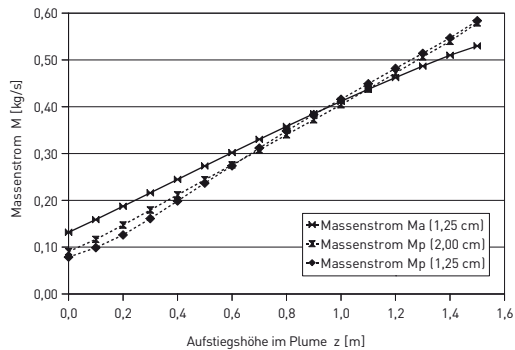
Plume ohne Luftereinmischung in die freien Enden entsprechend den Untersuchungen von Poreh (vgl. Abs. 2.4.4) und Thomas (vgl. Abs. 2.4.5) nachgebildet wird. In beiden Fällen ist lediglich ein einziger Rechenlauf je Wärmefreisetzungsrate erforderlich, da die Luftereinmischung unabhängig von der Größe der Rauchgasschicht im Atrium ist und alle Massenströme mit den Messfühlern der Messgruppe 2 zugleich ermittelt werden können. Die Simulationsrechnungen werden für Modelle mit Gitterweiten in Höhe von $\Delta l = 2,0 \text{ cm}$ und $1,25 \text{ cm}$ durchgeführt.

6.4.1 Zusammenfassung der Berechnungsergebnisse

Bei der Ermittlung der Luftereinmischung in den aufsteigenden Plume über die Messfühlergruppe 2 muss zwischen dem Massenstrom im gesamten Lösungsgebiet ($M_a = M_a^+ - M_a^-$) und dem Massenstrom der Rauchgase an sich ($M_p = M_p^+ - M_p^-$) unterschieden werden (vgl. Erläuterungen im Abs. 6.2.1). Während der Massenstrom M_a auch Bewegungen in der Frischluft außerhalb des Plumes beinhaltet, fließen in den Massenstrom M_p ausschließlich Zellen mit einem CO₂-Gehalt in Höhe von $m_{\text{CO}_2} \geq 0,01\%$ ein. Werden beide Massenströme in Abhängigkeit der Höhe z in ein Diagramm eingetragen, so wird ersichtlich, dass der Massenstrom M_a insbesondere bei geringen Höhen z größer ausfällt (s. Abb. 47). Die Ergebnisse der einzelnen Messfühler für alle Wärmefreisetzungsraten im vereinfachten Modell können der beiliegenden CD-ROM entnommen werden.

Der Anlagenteil E enthält darüber hinaus auf der Seite 159 die Ergebnisse aus den Versuchsreihen 1 bis 3 (ohne Unterzug am Überlauftrand) für den 3D-Plume mit Luftereinmischung in die freien Enden (vgl. Abb. 46, links). Die Ergebnisse für den zweidimensionalen Plume (vgl. Abb. 46, rechts) sind analog auf der Seite 160 dargestellt. Zusätzlich sind die Ergebnisse für den freien Plume in die Diagramme zur Luftereinmischung innerhalb von Atrien eingetragen (vgl. S. 161 ff., gestrichelte Linie).

Abb. 47: Massenströme M_a und M_p eines Plumes mit freien Enden, vereinfachtes Modell ohne Atrium ($W = 0,6 \text{ m}$, $Q_{\text{tot}} = 12 \text{ kW}$)



6.4.2 Diskussion der Ergebnisse

Der in Abb. 47 dargestellte Unterschied der Massenströme ist verständlich, wenn man berücksichtigt, dass die Heißgase im Brandraum auch Frischluft vor dem Brandraum erwärmen (Wärmetransport infolge Strahlung). Aufgrund dieser Erwärmung steigt Luft vor dem Brandraum auf und mischt sich – unabhängig vom Verhalten des Plumes – in die Rauchgasschicht des Atriums ein. Der im Experiment von Harrison innerhalb der Rauchabzugshaube gemessene Massenstrom beinhaltet daher nicht nur den Massenstrom im aufsteigenden Plume, sondern auch den Massenstrom aus aufsteigender Frischluft außerhalb des Plumes. Dieser Effekt wird in dieser Arbeit als Sekundäreinmischung bezeichnet (s. Abb. 50).

Für die Beurteilung der Lufteinmischung in einen aufsteigenden Plume ohne Atrium bzw. Rauchabzugshaube ist ausschließlich die Masseneinmischung M_p maßgebend, da für diesen Fall alle das Atrium betreffenden Effekte unberücksichtigt bleiben sollen. Wird die Masseneinmischung M_p aus der Computersimulation für die Versuchsreihen 1 bis 3 (Versuche ohne Unterzug) analog zu Abb. 41 ausgewertet, so erhält man für die Einmischungskonstante α einen Wert in Höhe von $\alpha = 0,218$ (Anstieg der Regressionsgeraden in Abb. 48). Bereits für den freien Plume (ohne Atrium) lässt sich mit dem ComputermodeLL damit eine gute Übereinstimmung mit der von Harrison versuchstechnisch ermittelten Einmischungskonstante erzielen (vgl. Gl. (21), $\alpha = 0,22$). Darüber hinaus ist ersichtlich, dass die Regressionsgerade die Y-Achse in Abb. 48 nur geringfügig oberhalb des Koordinatenursprungs schneidet. Diese Abweichung zu den experimentellen Ergebnissen (vgl. Abb. 41, S. 93) ist nachvollziehbar, da der Massenstrom im Experiment zusätzlich die Sekundäreinmischung berücksichtigt und damit insbesondere bei geringen z -Werten größer ausfällt.

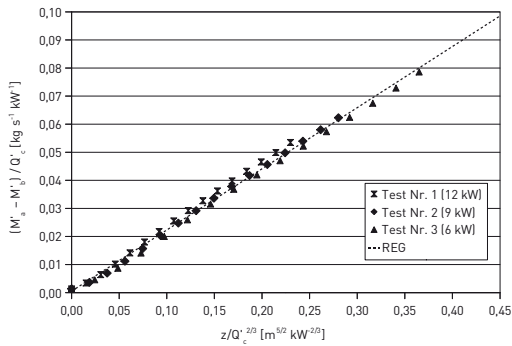
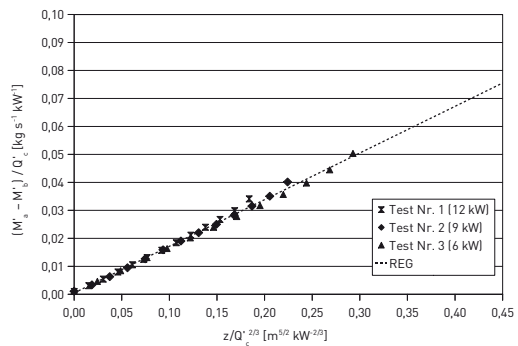


Abb. 48: Ermittlung der Einmischungskonstanten α für den aufsteigenden Plume ohne Atrium (Werte aus der Computersimulation für die Versuchsreihen 1 bis 3)

Die Auswertung der Masseneinmischung für das 2D-Modell lässt auf der Grundlage der von Marshall und Harrison [34] durchgeführten Experimente eine Einmischungskonstante in Höhe von $C_m = 0,44$ bzw. $\alpha = 0,364 \cdot C_m = 0,16$ erwarten (vgl. Abs. 2.4.4). Wie aus Abb. 49 ersichtlich wird, beträgt der mit der Computersimulation ermittelte Wert $\alpha = 0,167$ ($C_m = 0,455$), was in etwa den experimentellen Ergebnissen entspricht. Die Einmischungsrate in den aufsteigenden Plume kann damit sowohl für die experimentellen Untersuchungen nach Harrison [4] (3D-Plume) als auch für den von Marshall und Harrison [34] untersuchten zweidimensionalen Plume mit dem Computermodell gut nachvollzogen werden.

Abb. 49: Ermittlung der Einmischungskonstante α für einen 2D-Plume (Computersimulation)



6.5 Modellvalidierung: Massenstrom im aufsteigenden Plume (mit Atrium)

Die Validierung des Computermodells für des Massenstrom im Atrium anhand der Experimente von Harrison [4] gestaltet sich gegenüber den Simulationsrechnungen für den freien Plume (vgl. Abs. 6.4) ungleich aufwendiger. Während für die Modelle ohne Rauchabzugshaube die Massenströme in unterschiedlichen Höhen gleichzeitig ermittelt werden können (Messfühlergruppe 2), ist bei den Modellen mit Rauchabzugshaube für jeden Datenpunkt ein separater Rechenlauf durchzuführen. Dieser Aufwand ist erforderlich, da der Massenstrom im Atrium neben den geometrischen Randbedingungen bzw. der Wärmefreisetzungsrate auch von der Höhe der rauchgasfreien Schicht innerhalb des Atriums abhängig ist. Zur Validierung werden daher für jede Versuchsreihe acht Einzelmodelle erstellt, die sich nur durch die Größe der Rauchabzugsöffnungen voneinander unterscheiden (s. Abb. 42, S. 94). Für jedes Einzelmodell stellt sich nach einer entsprechenden Vorlaufzeit ein stationärer Zustand derart ein, dass sich der Massenstrom in den Rauchabzugsöffnungen und die nachströmende Frischluftmenge ausgleichen. Nach Erreichen des stationären Zustands wird die Höhe der rauchgasfreien Schicht z_{int} über die im Abschnitt 6.2.1 beschriebene Methode bestimmt. Abb. 50 zeigt die Ausbildung der rauchgasfreien Schicht in exemplarischer Form.

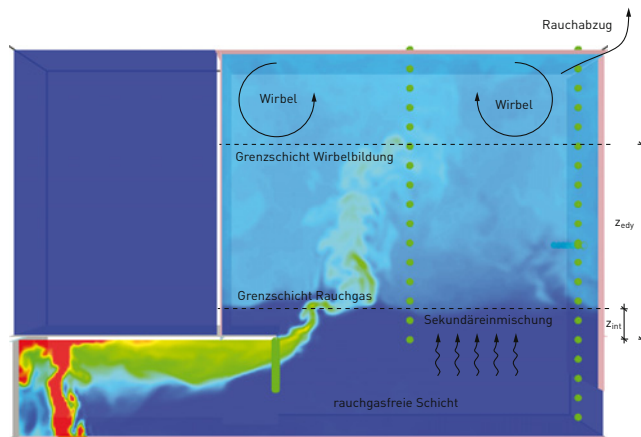


Abb. 50: Ausbildung der rauchgasfreien Schicht in der Höhe z_{int} oberhalb des Überlauffrandes nach Erreichen des stationären Zustands (Prinzipiskizze)

Für die Validierung der Massenströme innerhalb eines Atriums werden in dieser Arbeit Modelle mit einer Gitterweite in Höhe von $\Delta l = 2,0 \text{ cm}$ und $1,25 \text{ cm}$ verwendet. Für die von Harrison verwendete Rauchabzugshaube beinhaltet jedes Einzelmodell damit max.

3,5 Mio. Zellen. Die Rechenzeit für einen Datenpunkt beträgt unter Verwendung der entwickelten GPU-Version ca. 3,6 Stunden, so dass für die 54 Datenpunkte eine Gesamt-rechenzeit in Höhe von ca. 200 Stunden benötigt wird.

6.5.1 Zusammenfassung der Berechnungsergebnisse

In Abb. 51 ist die in der Computersimulation ermittelte Luftermischung im aufsteigenden Plume für die Versuchsreihe 1 (Gitterweiten $\Delta l = 2,0 \text{ cm}$ und $1,25 \text{ cm}$) dargestellt. Zusätzlich wurden in das Diagramm die von Harrison ermittelten Messwerte sowie die im Abschnitt 6.4 beschriebene Luftermischung in den freien Plume ohne Rauchabzugshaube eingetragen (gestrichelte Linie). Darüber hinaus enthält das Diagramm die Luftermischung, die sich unter Anwendung der BRE Spill-Plume-Methode für „kleine Atrien“ ergibt (vgl. Abs. 2.3). Der Anlagenteil E enthält auf den Seiten 161 bis 163 die Diagramme für alle übrigen von Harrison durchgeführten Versuchsreihen in analoger Form.

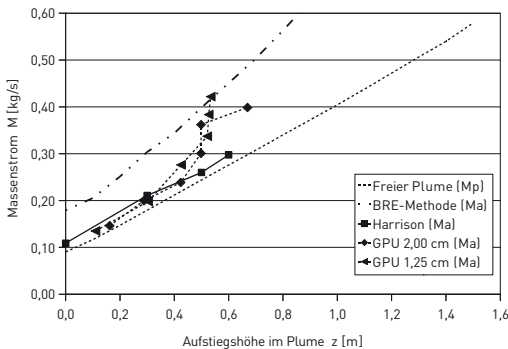


Abb. 51: Massenstrom im Atrium M_A in Abhängigkeit der Höhe der rauchgasfreien Schicht z (Versuchsreihe Nr. 1) und Massenstrom M_P im freien Plume

6.5.2 Diskussion der Ergebnisse

Relevant für die Validierung sind vor allem die Ergebnisse aus den Versuchsreihen 1 bis 3 (Versuche ohne Unterzug), da die Auswirkung von Unterzügen auf den Überlaufplume im Rahmen dieser Arbeit nicht untersucht werden soll. Werden die Ergebnisdigramme dieser Versuchsreihen betrachtet (vgl. S. 161), so wird ersichtlich, dass sich die Massenströme aus der Modellrechnung mit $\Delta l = 2,00 \text{ cm}$ Gitterweite nicht wesentlich von denen aus dem Modell mit feinerer Gitterweite ($\Delta l = 1,25 \text{ cm}$) unterscheiden. Darüber hinaus stimmen die im Rahmen der Simulationsrechnung ermittelten Massenströme für kleinere z -Werte mit den von Harrison gemessenen Werten überein. Die Zunahme der Massenströme liegt dabei etwa in der Größenordnung des freien Plumes (Anstieg der gestrichelten Linie), was zu der im Abschnitt 6.4.2 ermittelten Einmischungskonstante $\alpha = 0,22$ führt. Erwartungsgemäß liegen sowohl die berechneten als auch die gemessenen Mas-

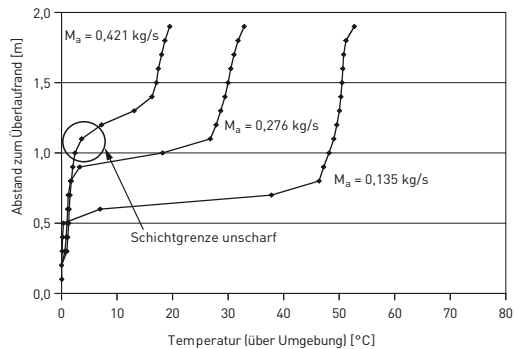
senströme M_a geringfügig über den für den freien Plume ermittelten Werten M_p . Ursächlich hierfür ist die Sekundäreinmischung, die den Massenstrom im Atrium gegenüber dem freien Plume höher ausfallen lässt.

Abweichend von den Messwerten ergibt sich für große Massenströme im Computermodell kaum eine weitere Vergrößerung der Schichthöhe z_{int} , was sich im überproportionalen Anstieg der berechneten Massenströme bei höheren z -Werten äußert. In diesem Bereich liegen die simulierten Massenströme signifikant über den von Harrison in [4] experimentell ermittelten Werten. Die Betrachtung der Videosequenzen aus der Simulationsrechnung lässt den Schluss zu, dass dieses Verhalten auf eine Wirbelbildung im oberen Bereich des Atriums zurückzuführen ist. Der Rauchgasanteil, der nicht durch die Entrauchungsöffnungen entweichen kann, wandert dabei über Wirbel nach unten und mischt sich erneut in den aufsteigenden Plume ein (s. Abb. 50). Dieser Effekt hat zur Folge, dass sich eine Vergrößerung der Rauchabzugsöffnungen ab einem bestimmten Wert nicht mehr in einer Vergrößerung der Schichthöhe z_{int} äußert, sondern lediglich zu einer Abnahme der Temperatur bzw. optischen Dichte im Rauchgas führt. In Abhängigkeit von Größe und Form des Atriums sowie der Wärmefreisetzungsrate kann dabei eine Wirbelgrenze z_{edy} ermittelt werden, oberhalb derer in jedem Fall Rauchgas auftritt.

Die Tatsache, dass die überproportionale Zunahme der Massenströme im Bereich der Wirbelgrenze in den von Harrison durchgeführten Versuchen nicht ermittelt werden konnte, liegt mit hoher Wahrscheinlichkeit in der Versuchsanordnung selbst begründet. Durch die Verringerung der optischen Rauchgasdichte wird es im Experiment schwer, die Höhe z_{int} aus visuellen Beobachtungen zu bestimmen. Zudem weist die Grenzschicht zwischen rauchgasführender und rauchgasfreier Schicht mit zunehmendem Massenstrom eine immer größer werdende Unschärfe auf, da die Temperaturdifferenz zwischen Rauchgas und Umgebungsluft abnimmt (vgl. Abb. 52). Tatsächlich wurden von Harrison in [4] 11 Versuche mit hohen z -Werten aussortiert, da hier infolge von Wirbelbildung keine klare Grenzschicht zwischen Rauchgas und rauchgasfreier Schicht erkennbar war (vgl. Abs. 6.1.3, letzter Absatz: „Plugholing-Effekt“).

Bereits Harrison weist in [10] und [4] darauf hin, dass die von ihm ermittelte Einmischungskonstante ($\alpha = 0,22$) in Bezug auf die BRE-Methode (vgl. Abs. 2.3) unerwartet gering ausfällt. Da es sich bei der von ihm verwendeten Rauchabzugshaube um ein „kleines Atrium“ handelt, wären bei Anwendung der BRE-Methode eine größere Luftereinmischungsrate bzw. größere Massenströme zu erwarten. Für diese Diskrepanz zwischen den hierbei gemessenen und den mit der BRE-Methode berechneten Werten findet Harrison keine Erklärung. Auch mit den im Rahmen dieser Arbeit erstellten Berechnungen im Anlagenteil B werden höhere Einmischungsraten ermittelt (vgl. Strich-Punkt-Linie in Abb. 51). Zur Ursache der Differenzen wird an dieser Stelle auf die Parameterstudie in Abschnitt 6.7 verwiesen.

Abb. 52: Temperaturprofile im Atrium für unterschiedliche Massenströme (Versuchsreihe 1)



Zusammenfassend ist festzustellen, dass mit den erstellten Computermodellen bei Gitterweiten in Höhe von $\Delta l \leq 2,0 \text{ cm}$ für den Massenstrom im Atrium eine gute Übereinstimmung zwischen den Versuchsergebnissen und den berechneten Werten erzielt wird. Voraussetzung dafür ist, dass die Wärmefreisetzungsrate im Brandraum derart gewählt wird, dass der konvektive Wärmestrom am Überlauftrand mit den experimentell gemessenen Werten übereinstimmt (vgl. Abs. 6.3) und dass die Grenzschichthöhe z_{int} einen durch Wirbelbildung charakterisierten Grenzwert z_{edy} nicht überschreitet.

6.6 Parameterstudie: Massenstrom im aufsteigenden Plume (ohne Atrium)

Obwohl sich diese Arbeit hauptsächlich mit dem Massenstrom innerhalb von Atrien beschäftigt, ist die Betrachtung eines Plumes in freier Umgebung von grundsätzlicher Bedeutung. Die in diesem Fall vorzufindende Luftermischung entspricht angenähert dem Massenstrom in großen Atrien (Vernachlässigung der Sekundäreinmischung) und stellt eine Untergrenze für Szenarien mit Atrien dar. Harrison und Spearpoint weisen in [5] darauf hin, dass die Einmischungskonstante α mit abnehmender Öffnungsweite W sukzessive zunimmt, da die zusätzliche Luftermischung über die freien Enden des Plumes immer stärker ins Gewicht fällt. Auf der Grundlage von experimentellen Untersuchungen geben die Autoren den Zusammenhang zwischen der Einmischung in einen Plume mit freien Enden mit $\alpha_{3D} = \alpha_{2D} + 0,25 \cdot (W/d_b)^{-2/3}$ an (vgl. Gl. (24)) und entwickeln daraus die Gleichung (25).

Im Rahmen der Validierung wurde bereits der Wert $\alpha_{2D} = 0,16$ für zweidimensionale Plumes sowie der Wert $\alpha_{3D} = 0,22$ für die von Harrison gewählte Öffnungsweite ($W = 0,6 \text{ m}$) nachvollzogen (vgl. Abs. 6.4). Zur Vervollständigung soll in diesem Abschnitt die Einmischungskonstante in Abhängigkeit der Öffnungsweite für einen freien Plume (ohne Atrium) bestimmt werden. Dazu werden für die Gitterweiten $\Delta l = 2,00 \text{ cm}$ und $1,25 \text{ cm}$ je 20 Modelle

erstellt, die sich durch die Öffnungsbreiten ($W = 0,2 / 0,3 / 0,4 / 0,6 / 0,8 \text{ m}$) und Wärmefreisetzungsraten ($Q_{\text{tot}} = 6 / 9 / 12 / 15 \text{ kW}$) voneinander unterscheiden. Alle übrigen Parameter entsprechen dem im Abschnitt 6.2 beschriebenen Computermodell.

6.6.1 Zusammenfassung der Berechnungsergebnisse

Für jedes der 20 Einzelmodelle kann die Einmischungskonstante α_{3D} durch die Ermittlung der Regressionsgeraden entsprechend Abb. 48 bzw. 49 bestimmt werden. Wird davon ausgegangen, dass die Einmischungskonstante für den zweidimensionalen Fall unabhängig von der Öffnungsbreite ist ($\alpha_{2D} \sim 0,16$, vgl. Abs. 6.4), so ergibt sich die Luftereinmischung in die freien Enden des Plumes als Differenz aus den beiden Einmischungskonstanten ($\alpha_{3D} - \alpha_{2D}$). Die Abb. 53 zeigt die im Rahmen dieser Arbeit ermittelten Werte $\alpha_{3D} - \alpha_{2D}$ in Abhängigkeit der Öffnungsbreite W und der Höhe der Rauchgasschicht am Überlaufrand d_b für alle 20 Einzelmodelle. Das Diagramm in Abb. 53 gibt dabei den bereits in Abb. 7 (S. 31) aus experimentellen Werten ermittelten Zusammenhang wieder, wobei dort die Luftereinmischung in den gesamten Plume α_{3D} betrachtet wird. Auf die Darstellung der Werte aus der Berechnung mit $\Delta l = 2,0 \text{ cm}$ Gitterweite wird an dieser Stelle verzichtet, da diese Simulationsrechnungen weitgehend gleichwertige Ergebnisse liefern.

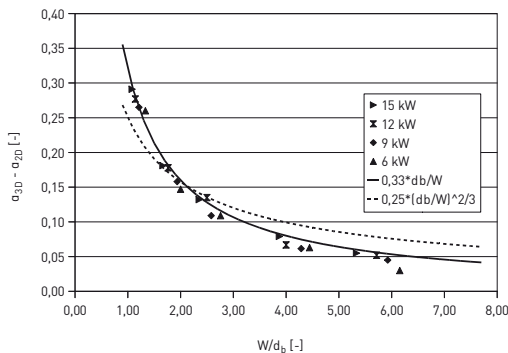


Abb. 53: Einmischungskonstanten in die freien Enden eines Plumes in freier Umgebung (ohne Atrium), $\Delta l = 1,25 \text{ cm}$

6.6.2 Entwicklung einer Näherungsformel für den freien Plume

Auf der Grundlage des Diagramms in Abb. 53 zeigt sich, dass die Einmischungskonstante indirekt proportional in Bezug auf die Öffnungsbreite W und direkt proportional in Bezug auf die Höhe der Rauchgasschicht im Brandraum d_b wächst bzw. abfällt. Der Zusammenhang zwischen der Einmischungskonstanten und den geometrischen Randbedingungen kann auf der Grundlage der Simulationsrechnung wie folgt abgeschätzt werden (vgl. durchgehende Linie in Abb. 53):

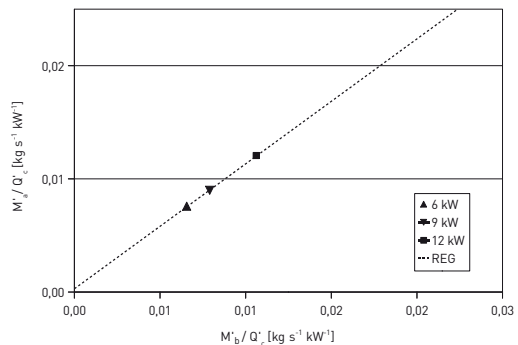
$$\alpha_{3D} = \alpha_{2D} + 0,33 d_b / W = 0,16 + 0,33 d_b / W \quad (54)$$

Dieser, über die Simulationsrechnung ermittelte Zusammenhang ist geringfügig abweichend von der durch Harrison und Spearpoint in [5] angegebenen Gleichung (24), die zum Vergleich in Abb. 53 ebenfalls eingetragen wurde (vgl. gestrichelte Linie). Es ist davon auszugehen, dass zumindest ein Teil der Abweichungen in der im Experiment erfassten Sekundäreinmischung zu finden ist. In den experimentellen Untersuchungen von Harrison und Spearpoint wurde durchweg der Massenstrom innerhalb von Rauchgasabzugshauben gemessen, so dass sich die zusätzliche Gasströmung infolge der Erwärmung von Frischluft unterhalb der Heißgasschicht auf die Messergebnisse auswirkt.

Die von Harrison und Spearpoint erfasste Sekundäreinmischung wirkt sich ebenfalls auf die im Experiment gemessene Einmischungsrate für den Krümmungsbereich des Plumes aus. Betrachtet man die Auswertung in Abb. 41 (S. 93), so kann die Massenstromzunahme zwischen dem horizontalen Rauchgasstrom am Überlaufrand M_b und dem aufsteigenden Plume in Höhe $z = 0 \text{ cm}$ ($M_{p,z=0}$) aus dem Schnittpunkt der Regressionsgeraden mit der y-Achse abgelesen werden. Die von Harrison und Spearpoint ermittelten Werte fallen dabei durchweg größer aus, als die im Rahmen der Simulationsrechnung (ohne Atrium) berechneten Werte (vgl. Abb. 48, S. 103). Auch dieser Umstand liegt in der Sekundäreinmischung begründet, d. h. die von Harrison und Spearpoint gemessenen Massenströme in Höhe $z = 0 \text{ cm}$ beinhalten einen Anteil zusätzlicher Frischluft, die sich außerhalb des Plumes in die Rauchgasschicht einmischt und stellen nicht ausschließlich die Luftereinmischung in den Krümmungsbereich des Plumes dar.

Im Modell von Thomas (vgl. Gl. (23)) wird die Masseneinmischung im Krümmungsbereich durch die Faktoren β und γ berücksichtigt. Harrison und Spearpoint bestimmen die Faktoren auf der Grundlage der ihnen zur Verfügung stehenden experimentellen Werte für den zweidimensionalen Plume zu $\beta = 1,34$ und $\gamma \sim 0$ (vgl. Abs. 2.4.6). Die Auswertung der Berechnungsergebnisse aus der Simulationsrechnung lässt auf einen kleineren Wert β schließen ($\beta = 1,10$ und $\gamma \sim 0$), da die Sekundäreinmischung in der Simulationsrechnung nicht berücksichtigt wird (vgl. Abb. 54).

Abb. 54: Zusammenhang zwischen M_b und $M_{p,z=0 \text{ cm}}$ in Abhängigkeit der Wärmefreisetzungsrate Q_{tot} , Ermittlung der Faktoren β und γ über die Regressionsgrade $y = \beta \cdot x + \gamma$



Wird Gleichung (54) in Gleichung (23) eingesetzt, so kann unter Berücksichtigung der neuen Faktoren β und γ der folgende Zusammenhang für den Massenstrom im aufsteigenden Plume angegeben werden:

$$\begin{aligned} M_{a,3D} &= 0,16 Q^{1/3} W^{2/3} z + 0,33 Q^{1/3} W^{-1/3} z + 1,10 M_b \quad \text{bzw.} \\ M_{a,3D} &= (0,16 + 0,33 d_b / W) Q^{1/3} W^{2/3} z + 1,10 M_b \end{aligned} \quad (55)$$

Dabei wird davon ausgegangen, dass die für den zweidimensionalen Fall gefundene Masseneinmischung in den Krümmungsbereich auch für den dreidimensionalen Fall gültig ist, d. h. dass die Luftermischung in die Enden des dreidimensionalen Plumes innerhalb des Krümmungsbereichs vernachlässigbar klein ausfällt. Die Gleichung (55) entspricht vom Wesen her der von Harrison und Spearpoint vorgeschlagenen Gleichung (25), berücksichtigt jedoch ausschließlich die direkte Masseneinmischung in den Plume und lässt zusätzliche Effekte (Sekundäreinmischung etc.) außen vor.

6.7 Parameterstudie: Massenstrom im aufsteigenden Plume (mit Atrium)

Ausgehend von der Gleichung (55) soll im Folgenden der Einfluss eines Atriums auf den Massenstrom im aufsteigenden Plume untersucht werden. Dabei ist zu erwarten, dass der Massenstrom im Atrium gegenüber dem freien Plume größer ausfällt, da innerhalb des Atriums zusätzliche Effekte eine Rolle spielen (Sekundäreinmischung, Wirbelbildung etc.). Im Rahmen dieser Arbeit wird zur Abschätzung dieser Effekte eine Parameterstudie erstellt, in der die Randbedingungen des im Abschnitt 6.2 beschriebenen Modells im Hinblick auf die Atriengröße, die Öffnungsbreite und die Wärmefreisetzungsrate variiert werden. Insgesamt werden vier Atrientypen mit unterschiedlichen Grundflächen (Atrium Typ A bis D) und Höhen ($H_a = 0,8 / 1,2 / 1,6 / 2,0$ m) modelliert. Für jedes Atrium werden dazu acht Simulationsrechnungen (S1 bis S8) durchgeführt, die sich durch die Größe der Rauchabzugsöffnungen A_{0ff} voneinander unterscheiden. In Tabelle 11 sind die untersuchten Atriengeometrien und die zugehörigen Vorlaufzeiten t_v (vgl. Abs. 6.2.2) zusammenfassend dargestellt. Zusätzlich ist die Rechenzeit eingetragen, die mit der neu entwickelten GPU-Version unter Verwendung einer handelsüblichen GPU-Karte (Tesla C2050/70) für die acht Simulationsrechnungen eines Atriums benötigt wurde.

Die in Tabelle 11 dargestellte Untersuchungsreihe wird analog zum Vorgehen in Abschnitt 6.6 für unterschiedliche Öffnungsbreiten ($W = 0,2 / 0,3 / 0,4 / 0,6 / 0,8$ m) und Wärmefreisetzungsraten ($Q_{tot} = 6 / 9 / 12$ kW) wiederholt. Insgesamt werden damit in der Parameterstudie 1.248 unterschiedliche Simulationsrechnungen durchgeführt. Die Gesamtrechenzeit beträgt unter Verwendung der neuen GPU-Version ca. 620 Stunden. Infolge der Vielzahl an erforderlichen Modellrechnungen wird die Parameterstudie ausschließlich an Modellen mit der Gitterweite $\Delta l = 2,0$ cm durchgeführt. Alle bisherigen Ergebnisse weisen darauf hin, dass mit kleineren Gitterweiten keine abweichenden Ergebnisse erzielt werden.

Atrium Typ A (2,20 m²)							Atrium Typ B (3,42 m²)					
	A _{off} [m²]	H _a [m]:	0,8	1,2	1,6	2,0	A _{off} [m²]	H _a [m]:	0,8	1,2	1,6	2,0
S1	0,0820	t _v [s]:	36	36	36	36	0,0731	t _v [s]:	36	72	72	72
S2	0,1408	t _v [s]:	36	36	36	36	0,1152	t _v [s]:	36	72	72	72
S3	0,1996	t _v [s]:	36	36	36	36	0,1776	t _v [s]:	36	72	72	72
S4	0,2940	t _v [s]:	36	36	36	36	0,2304	t _v [s]:	36	36	36	36
S5	0,3992	t _v [s]:	36	36	36	36	0,2928	t _v [s]:	36	36	36	36
S6	0,4936	t _v [s]:	36	36	36	36	0,4392	t _v [s]:	36	36	36	36
S7	0,5880	t _v [s]:	36	36	36	36	0,5856	t _v [s]:	36	36	36	36
S8	0,7056	t _v [s]:	36	36	36	36	0,7320	t _v [s]:	36	36	36	36
	T _{GPU} [h]:		~2,7	~4,5	~5,7	~7,3	T _{GPU} [h]:		~3,3	~5,7	~7,3	~8,6

Atrium Typ C (6,67 m²)							Atrium Typ D (16,24 m²)					
	A _{off} [m²]	H _a [m]:	0,8	1,2	1,6	2,0	A _{off} [m²]	H _a [m]:	0,8	1,2	1,6	2,0
S1	0,0548	t _v [s]:	48	96	96	96	0,0792	t _v [s]:	88	176	176	176
S2	0,0944	t _v [s]:	48	96	96	96	0,1584	t _v [s]:	88	176	176	176
S3	0,1568	t _v [s]:	48	96	96	96	0,2388	t _v [s]:	88	176	176	176
S4	0,2512	t _v [s]:	48	48	48	48	0,3192	t _v [s]:	88	88	88	88
S5	0,3608	t _v [s]:	48	48	48	48	0,3984	t _v [s]:	88	88	88	88
S6	0,5024	t _v [s]:	48	48	48	48	0,4776	t _v [s]:	88	88	88	88
S7	0,6120	t _v [s]:	48	48	48	48	0,6384	t _v [s]:	88	88	88	88
S8	0,7216	t _v [s]:	48	48	48	48	0,7196	t _v [s]:	88	88	88	88
	T _{GPU} [h]:		~6,7	~12	~15	~19	T _{GPU} [h]:		~16	~88	~104	~119

Tabelle 11: Geometrische Randbedingungen und Vorlaufzeiten für die Modelle der Parameterstudie zur Lufteinmischung in den Überlaufplume mit Atrium

6.7.1 Zusammenfassung der Berechnungsergebnisse

In Abb.55 sind die aus der Computersimulation ermittelten Massenströme innerhalb eines exemplarisch ausgewählten Atriums für unterschiedliche Wärmefreisetzungsraten aufgetragen. Darüber hinaus sind der Massenstrom im freien Plume entsprechend Abschnitt 6.6 (gestrichelte Linie) und die nach der BRE-Methode ermittelten Massenströme (Strich-Punkt-Linien) eingetragen.

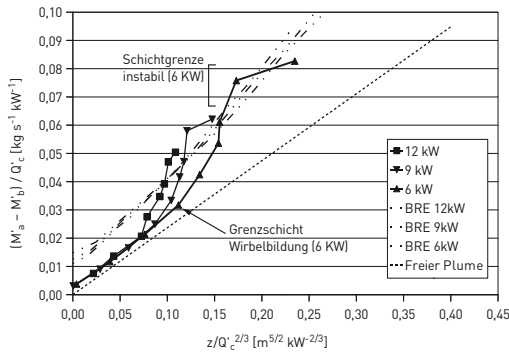
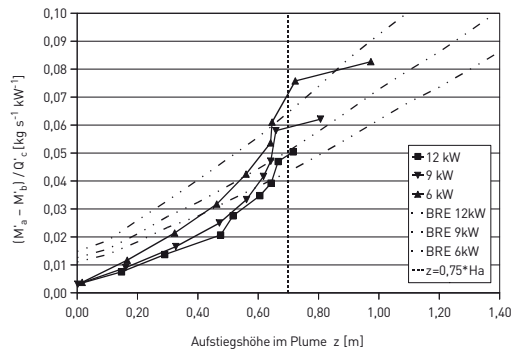


Abb. 55: Massenströme im Atrium Typ C ($H_a = 1,6$ m, $W = 0,6$ m) bei Wärmefreisetzungsraten in Höhe von $Q_{tot} = 6, 9$ und 12 kW

Wie bereits im Rahmen der Validierung festgestellt wurde (vgl. Abb. 51), steigt der Massenstrom in Bezug auf den Term $z/Q_c^{2/3}$ bis zur Wirbelgrenze linear und anschließend überproportional an. Eine weitere Steigerung des Massenstroms äußert sich im Bereich mit Wirbelbildung vermehrt in einer Reduktion der Rauchgastemperatur bzw. optischen Rauchgasdichte, was zunehmend zu einer unscharfen Schichtgrenze zwischen Frischluft und Rauchgas führt. Rauchgastemperatur und optische Dichte sinken dabei derart ab, dass die Bestimmung der Grenzschicht z_{int} ab einem bestimmten Massenstrom nicht mehr zuverlässig möglich ist und die ermittelten Schichtgrenzen eine große Streuung aufweisen (vgl. Abb. 55, „instabile Schichtgrenze“). Oberhalb des instabilen Bereichs steigt die Grenzschicht infolge der verschwindend geringen Temperaturdifferenz zwischen verrauchtem und rauchgasfreiem Bereich sprunghaft an.

Die Abb. 56 zeigt den Massenstrom im gleichen Atrium in Bezug auf die Koordinate z . Die Bestimmung der Messwerte erfolgte in der Computersimulation dabei mindestens bis zu einer Höhe z von 75 % der Atrienhöhe H_a . Diese Grenzhöhe ist in Abb. 56 durch die vertikal verlaufende Linie dargestellt.

Abb. 56: Massenströme im Atrium Typ C ($H_a = 1,6$ m, $W = 0,6$ m) bei Wärmefreisetzungsraten in Höhe von $Q_{tot} = 6, 9$ und 12 kW



Das oben beschriebene Verhalten wurde prinzipiell in allen Atrien festgestellt, wobei die unterschiedlichen Bereiche mit abnehmender Atriengröße sukzessive ineinander übergehen. Im Anlagenteil E sind auf den Seiten 164 bis 175 die ermittelten Massenströme für alle untersuchten Geometrien und Wärmefreisetzungsraten wiedergegeben.

6.7.2 Entwicklung einer Näherungsformel für die Masseneinmischung

Der Vergleich zwischen den Simulationsrechnungen und der BRE-Methode zeigt, dass der Zusammenhang aus der BRE-Methode im Wesentlichen als Einhüllende aller auftretenden Massenströme aufgefasst werden kann. Insbesondere bei kleinen Atrien (Atrium Typ A und B) ist dieser Effekt deutlich zu erkennen (vgl. Anlagenteil E, S. 164–170). Diese Tatsache ist verständlich, denn die BRE-Methode basiert neben den theoretischen Überlegungen zum freien Plume auf experimentellen Versuchsergebnissen, die für „kleine Atrien“ gewonnen wurden (vgl. Abs. 2.3). Die freien Parameter aus der Theorie wurden durch die Autoren derart gewählt, dass sämtliche, innerhalb eines Atriums auftretenden Effekte berücksichtigt sind. Dieses Vorgehen erklärt auch die von Harrison in [4] erwähnte Diskrepanz zwischen der von ihm ermittelten und der in der BRE-Methode verwendeten Einmischungskonstante (s. Abs. 6.5.2). Die letztere Einmischungskonstante fällt höher aus, da in ihr sämtliche zusätzlichen Effekte (Sekundäreinmischung, Wirbelbildung) bereits enthalten sind. Auf Grundlage dieser Erkenntnis ist es naheliegend, den im Abschnitt 6.6.2 für den freien Plume gewonnenen Zusammenhang derart anzupassen, dass die zusätzlichen Effekte innerhalb des Atriums pauschal berücksichtigt werden. In diesem Fall ist die Einmischungskonstante – analog zum Vorgehen in der BRE-Methode – als Einhüllende aller möglichen Massenströme aufzufassen, die sich in Atrien mit unterschiedlicher Geometrie einstellen können.

Da sich die Schichtgrenze im Bereich der Wirbelbildung indifferent verhält und damit klare Aussagen in Bezug auf das Verhältnis zwischen Schichtgrenze und Massenstrom bei großen z -Werten kaum noch möglich sind, ist es sinnvoll, eine Obergrenze für die Höhe der Rauchgasschicht zu definieren, die im Rahmen einer Bemessung nicht überschritten werden darf. In der amerikanischen Norm NFPA 92B [36] wird die minimale Höhe der Rauchgasschicht bei Anwendung der Gleichung (28) auf 20 % der Gesamthöhe des Atriums begrenzt, um der Wirbelbildung Rechnung zu tragen (vgl. Abs. 2.4.8). Damit ergibt sich für die Schichthöhe z_{int} ein maximal zulässiger Wert in Höhe von $h + z_{\text{int}} \leq 0,80 \cdot H_A$. Die Simulationsrechnungen zeigen, dass die in [36] angegebene Grenze für die Wirbelbildung insbesondere bei hohen Atrien nicht in jedem Fall ausreichend ist. Im Rahmen dieser Arbeit wird daher eine maximale Höhe der Schichtgrenze von $h + z_{\text{int}} \leq 0,75 \cdot H_A$ vorausgesetzt, d. h. das Rauchgasreservoir muss eine Mindesthöhe von 25 % der Atrienhöhe aufweisen. Die entsprechenden Grenzhöhen sind für ein exemplarisch ausgewähltes Atrium in Abb. 57 als vertikale Linien eingezeichnet. Der Anlagenteil E enthält diese Grenzlinien für alle übrigen Atrien in entsprechender Form.

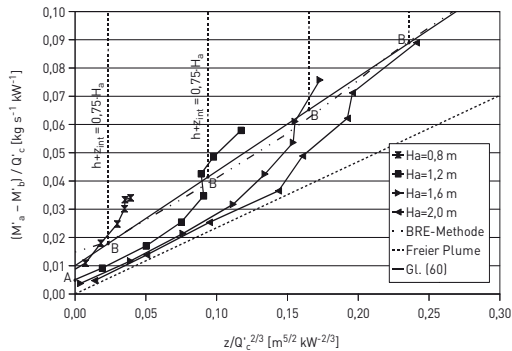


Abb. 57: Massenströme im Atrium Typ C ($H_a = 2,0\text{ m}$, $W = 0,6\text{ m}$) bei einer Wärme-freisetzungsrate in Höhe von $Q_{\text{tot}} = 6\text{ kW}$

Die Simulationsrechnungen zeigen, dass bei Verwendung des Ansatzes von Thomas (Gl. (17)) und den folgenden Parametern in allen untersuchten Atrien auf der sicheren Seite liegende Ergebnisse erhalten werden:

$$\alpha_{3D} = 0,26 + 0,33 d_b / W; \quad \beta = 1,10; \quad \gamma = 0,01 \quad (56)$$

Der Parameter β wurden dabei zunächst entsprechend dem Vorgehen in Abschnitt 6.6.2 beibehalten, die Parameter α und γ wurden entsprechend der Ergebnissen aus den Simulationsrechnungen passend gewählt, wobei ausschließlich Massenströme unterhalb der definierten Maximalgrenze ($h+z_{\text{int}} \leq 0,75 \cdot H_a$) berücksichtigt werden. Durch Einsetzen der Parameter (56) in die Gleichung (17) ergibt sich ein Zusammenhang, der vom Konzept her der Gleichung (55) ähnelt, jedoch als Einhüllende der maximalen Massenströme aller denkbaren Atriengrößen verstanden werden kann:

$$M_{a,3D} = \left(0,26 + 0,33 d_b / W \right) Q^{1/3} W^{2/3} z + 1,10 M_b + 0,01 \quad (57)$$

In Abb. 57 ist die über den Zusammenhang (57) gewonnene Masseneinmischung mit einer durchlaufenden Linie eingetragen (vgl. auch Diagramme im Anlagenteil E, „New Equation“). Die Simulationsrechnungen zeigen, dass mit der Gleichung (59) in nahezu allen untersuchten Atrien auf der sicheren Seite liegende Massenströme erhalten werden, sofern die Dicke der Rauchgasschicht mindestens 25 % der Atrienhöhe beträgt. Eine Ausnahme bilden große flache Atrien, bei denen die Wirbelbildung direkt oberhalb des Überlaufendes beginnt (bspw. S. 175, $H_a = 0,8\text{ m}$). In diesen, baupraktisch wenig relevanten Atrientypen werden mit der Näherungsformel (57) (ebenso der BRE-Methode) zu geringe Massenströme ermittelt.

Mit Ausnahme der oben erwähnten, großen flachen Atrien liegen alle ermittelten Schnittpunkte der Einmischungsgraphen mit der y-Achse unterhalb des Wertes $y_0 \leq 0,005$. Es ist daher ebenfalls denkbar, den Massenstrom durch eine Gerade zwischen den Punkten A und

B in Abb. 57 zu beschreiben. Auch in diesem Fall liegen die über die Schnittgeraden ermittelten Massenströme auf der sicheren Seite. Als Alternative zu Gleichung (57) kann daher auch der folgende Zusammenhang verwendet werden:

$$M_{a,3D} = \left(0,26 + 0,33 \frac{d_b}{W} + \frac{0,005}{0,75 H_a - h} \right) Q^{1/3} W^{2/3} z + 1,10 M_b + 0,005 \quad (58)$$

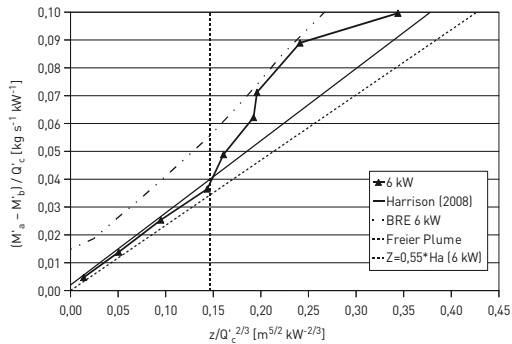
Die Gleichung (58) ergibt gegenüber der Gleichung (57) insbesondere bei geringen z -Werten günstigere Werte.

6.7.3 Bewertung der bestehenden Näherungsformeln

Auf der Grundlage der erstellten Parameterstudien ist es möglich, die bereits bestehenden Näherungsformeln neu zu bewerten. Voraussetzung für die Anwendung der im Abschnitt 2.4 beschriebenen vereinfachten Ansätze ist eine ausreichende Größe des Rauchgasreservoirs. In allen Literaturquellen wird darauf hingewiesen, dass die Zusammenhänge nur für Atrien mit ausreichend großem Rauchgasreservoir gelten, wobei der Begriff „groß“ entweder nicht näher spezifiziert oder in Anlehnung an die BRE-Methode definiert wird ($h + z_{int} < 0,67 \cdot A_a^{0,5}$, vgl. Abs. 2.3). Für das in Abb. 58 dargestellte Beispiel wäre die Anwendung der vereinfachten Näherungsformeln damit bis zu einer Höhe von $z_{int} \sim 0,67 \cdot (667 \text{ m}^2)^{0,5} - 5,0 \text{ m} = 12,3 \text{ m}$ möglich. Bei beispielhafter Betrachtung des von Harrison in [5] angegebenen Zusammenhangs ergibt sich der maximal zulässige Anwendungsbereich für das gewählte Beispiel aus dem Schnittpunkt zwischen der Gleichung (25) und den mit der Simulationsrechnung ermittelten Werten (vgl. Abb. 58, senkrechte Linie, $z_{int}/Q_c'^{2/3} \sim 0,147$). Unter Berücksichtigung der Zusammenhänge (52) beträgt der konvektive Wärmestrom $Q_c' = 4,96 \text{ kW} / 0,6 \cdot 10^{-2,5} / 10 = 261,4 \text{ kW/m}$ und die maximal zulässige Anwendungsgrenze damit $z_{int} = 0,147 \cdot 261,4^{2/3} = \underline{6,0 \text{ m} < 12,3 \text{ m}}$. Daraus lässt sich schlussfolgern, dass bei Verwendung der Gleichung (25) in dem gewählten Beispielatrium die Schichtgrenze ca. 55 % der Atrienhöhe nicht übersteigen darf. Andernfalls werden bei Anwendung der Näherungsformel (25) teilweise zu geringe Massenströme erhalten.

Die obige Beispielrechnung zeigt, dass die generelle Beschränkung der Anwendung auf große Atrien für den Anwender nicht ausreichend ist und dass die im Rahmen der BRE-Methode gefundene Abgrenzung von „großem und kleinem“ Rauchgasreservoir für die Näherungsformeln aus Abschnitt 2.4 keine Gültigkeit besitzt. Solange keine fortführenden Untersuchungen zur Gültigkeit der bekannten Näherungsformeln erfolgt sind, muss daher von einer Anwendung der Formeln bei Rauchgasschichthöhen über ca. 50 % der Atrienhöhe abgeraten werden.

Abb. 58: Massenströme im Atrium Typ C ($H_a = 2,0$ m, $W = 0,6$ m, $Q_c = 4,96$ kW) im Vergleich mit dem von Harrison in [5] gefundenen Zusammenhang



7 Zusammenfassung und Ausblick

Diese Arbeit beschäftigt sich mit der Entrauchung von Atrien über Rauch- und Wärmeabzüge, bei denen sich das Rauchgas aus angrenzenden Räumlichkeiten über einen Überlaufplume in das Atrium ergießt. Auf der Grundlage einer CFD-Studie werden für dieses Szenario vereinfachte Näherungsformeln entwickelt, mit der die Luftermischung in den aufsteigenden Plume und damit die zu erwartende Rauchgasmenge im Atrium auf der sicheren Seite liegend abgeschätzt werden kann. Im Gegensatz zu den bereits bekannten Zusammenhängen, deren Anwendung auf große Atrien bzw. große Rauchgasreservoirs beschränkt ist (vgl. Abs. 2.4), sind die neuen Formeln robuster und können für eine Vielzahl denkbarer Szenarien verwendet werden (Unabhängigkeit von der Grundrissform des Atriums, Berücksichtigung der Breite des Überlaufplumes etc.).

7.1 Einsatz der GPU-Prozessortechnologie in der Brand- und Rauchsimulation

Aufgrund der benötigten Rechenleistung für die CFD-Studie wird in dieser Arbeit zunächst untersucht, inwieweit die verwendete Software FDS [7] durch den Einsatz von GPU-Prozessortechnik beschleunigt werden kann. Dazu werden wesentliche Teile des in FDS enthaltenen Rechenkerns von FORTRAN nach C/CUDA portiert und an die massiv parallel arbeitende GPU-Technik angepasst (Parallelisierung bestehender und Entwicklung neuer Algorithmen). Ein Überblick über die parallelisierten Module enthält die Tabelle 4 (S. 46). Infolge des Zeitaufwandes für die Kopiervorgänge zwischen GPU-Karte und Hostcomputer ist eine Teilparallelisierung, welche ausschließlich die rechenintensiven Algorithmen beinhaltet, nicht zielführend (vgl. Abs. 4.3.2). In dieser Arbeit werden daher alle beteiligten Module (Strömungsmechanik, Verbrennung, Wärmetransport über Strahlung und Wärmeleitung) schleifenweise parallelisiert und in die GPU-Karte ausgelagert.

Die in FDS enthaltenen Algorithmen aus den Bereichen Strömungsmechanik, Wärmeleitung sowie Verbrennung lassen sich aufgrund ihres weitgehend expliziten Charakters vergleichsweise gut parallelisieren. Die Algorithmen zum Strahlungswärmetransport in gasförmigen Medien hingegen werden im Rahmen dieser Arbeit ausgetauscht, da die Natur des Wellenfrontalgorithmus⁴ in der verwendeten FV-Methode eine einfache Parallelisierung nicht zulässt. Dazu wird ein erweiterter Wellenfrontalgorithmus entwickelt, der besonders auf GPU-Systemen eine hohe Effizienz aufweist (vgl. Abs. 4.5). Zusätzlich werden im Rahmen der Parallelisierungsarbeiten in allen Modulen die Datenstrukturen neu geordnet (vgl. Abs. 4.4.2). Wie Untersuchungen zeigen, wird dadurch die Zugriffsgeschwindigkeit auf die zu verarbeitenden Datensätze erheblich beschleunigt.

Zur Verifikation der vorgenommenen Änderungen am Quelltext wird die neue GPU-Version darüber hinaus um einen „Diagnosemodus“ ergänzt, mit dem die neuen parallelen Algorithmen mittels Emulation auf der CPU ausgeführt werden können (vgl. Abs. 5.1). Durch einen anschließenden Ergebnisvergleich mit den Daten aus den bestehenden sequenziellen Algorithmen wird eine erhebliche Anzahl möglicher Fehlerquellen ausgeschlossen. Ebenfalls werden zur Verifikation zwei Beispiele gerechnet, für die analytische Lösungen bzw. Lösungen aus experimentellen Versuchen bekannt sind (vgl. Abs. 5.2 und 5.3).

Nach Abschluss der Parallelisierungsarbeit und Optimierung des neuen Quelltextes mittels einer dafür entwickelten Profilersoftware (vgl. Abs. 4.6), wird für den strömungsmechanischen Teil der Software eine Beschleunigung um etwa den Faktor⁴ 30 erreicht (vgl. Tab. 6, S. 83). Der neu entwickelte Multi-Wellenfrontalgorithmus für den Strahlungswärmetransport in gasförmigen Medien beschleunigt die bestehenden Algorithmen um etwa den Faktor⁴ 10 (vgl. Tab. 7, S. 87). Für praktische Anwendungsfälle (Kombination aus Strömungsmechanik, Strahlung und Verbrennung) sind Beschleunigungsfaktoren⁴ um den Faktor 20 als realistisch anzusehen.

Die vorgenommene Weiterentwicklung der Basisversion von FDS zeigt die Vorteile beim Einsatz der GPU-Prozessortechnik für die Brand- und Rauchsimulation auf. Dem vergleichsweise hohen Aufwand (Portierung des Quelltextes nach C/CUDA, schleifenweise Parallelisierung der Algorithmen, Neuordnung von Datenstrukturen) steht eine ganz erhebliche Beschleunigung der Rechenleistung gegenüber. Sofern eine Weiterführung der Arbeiten zweckmäßig erscheint (vollständige Anpassung aller Algorithmen an die GPU-Technik), wäre vorab zu klären, inwieweit mit dem zwischenzeitlich erschienenen FORTRAN-Compiler [54] eine gleichwertige Leistungssteigerung erreicht werden kann. In diesem Fall könnte auf die aufwendige Portierung des Quelltextes nach C/CUDA verzichtet werden.

⁴ Alle Beschleunigungsfaktoren stellen den Geschwindigkeitszuwachs bei Verwendung einer GPU-Karte vom Typ GeForce GTX 580 gegenüber einer CPU vom Typ Intel Core i7 980X (one core) dar.

Abschließend sei noch darauf verwiesen, dass die sequenziellen Algorithmen vor Weiterführung der Parallelisierung modifiziert werden sollten. In der derzeitigen Form sind die Ergebnisse nicht grundsätzlich unabhängig von der Bearbeitungsreihenfolge der Datensätze innerhalb der Programmschleifen (vgl. Erläuterungen im Abs. 5.1).

7.2 Entwicklung von Näherungsformeln für den Massenstrom im Überlaufplume innerhalb von Atrien

Unter Verwendung der neu erstellten GPU-Version der Software FDS wird in dieser Arbeit eine ausführliche CFD-Studie durchgeführt, mit der die Luftermischung in einen Überlaufplume innerhalb von Atrien analysiert wird. Dazu werden auf der Grundlage von Versuchsergebnissen (vgl. Abs. 6.1) Computermodelle erstellt, die einen Brandraum mit Austrittsöffnung sowie eine Rauchabzugshaube zum Auffangen der Rauchgase beinhalten (vgl. Abs. 6.2). Im Anschluss wird die Wärmefreisetzungsrate des Brandherdes iterativ derart angepasst, dass die Rauchgasströmung unter dem Überlauftrand in ihrer Charakteristik (konvektiver Wärmestrom, maximale Rauchgastemperatur, Temperatur- und Geschwindigkeitsprofile) mit den Versuchsergebnissen übereinstimmt (vgl. Abs. 6.3). Die Validierung des Massenstroms im aufsteigenden Plume erfolgt zum einen durch bekannte Zusammenhänge für den freien Plume (vgl. Abs. 6.4) und zum anderen durch die in den Versuchen gemessenen Massenströme innerhalb der Rauchabzugshaube (vgl. Abs. 6.5).

Nach Abschluss der Modellvalidierung wird zunächst eine Parameterstudie am freien Plume unter Vernachlässigung des Atriums durchgeführt (vgl. Abs. 6.6). Auf der Grundlage dieser CFD-Studie kann der Massenstrom im aufsteigenden Plume in Abhängigkeit der Öffnungsbreite W , der Höhe der Rauchgasschicht am Überlaufplume d_b , dem Massen- bzw. konvektiven Wärmestrom unter dem Überlauftrand M_b bzw. Q sowie der Höhe z wie folgt ermittelt werden:

$$M_{a,3D} = \left(0,16 + 0,33 d_b / W\right) Q^{1/3} W^{2/3} z + 1,10 M_b \quad (59)$$

Die Gleichung (59) entspricht vom Wesen her der von Harrison vorgeschlagenen Beziehung (25), berücksichtigt jedoch ausschließlich die direkte Masseneinmischung in den Plume und lässt zusätzliche Effekte (Sekundäreinmischung etc.) außen vor.

Mit einer weiteren Parameterstudie wird anschließend der Massenstrom im Überlaufplume innerhalb eines Atriums untersucht. Dazu wird das zuvor validierte Modell im Hinblick auf die Atriengeometrie, die Öffnungsbreite des Brandraumes sowie die Wärmefreisetzungsrate variiert (vgl. Abs. 6.7). Die Ergebnisse der CFD-Studie zeigen, dass der Massenstrom im aufsteigenden Plume unabhängig von der Atriengeometrie auf der sicheren Seite liegend wie folgt ermittelt werden kann:

$$M_{a,3D} = \left(0,26 + 0,33 d_b / W\right) Q^{1/3} W^{2/3} z + 1,10 M_b + 0,01 \quad (60)$$

Ein alternativer Zusammenhang, mit dem vor allem bei geringen Höhen z günstigere Massenströme erhalten werden, liefert der folgende Zusammenhang:

$$M_{a,3D} = \left(0,26 + 0,33 \frac{d_b}{W} + \frac{0,005}{0,75 H_a - h} \right) Q^{1/3} W^{2/3} z + 1,10 M_b + 0,005 \quad (61)$$

Die Anwendung der Gleichung (60) bzw. (61) setzt voraus, dass die rauchgasfreie Schicht nicht durch Wirbelbildung im oberen Bereich des Atriums beeinflusst wird. Wie die CFD-Studie zeigt, kann diese Forderung als erfüllt angesehen werden, sofern die Höhe des Rauchgasreservoirs im Atrium mindestens 25 % der Atrienhöhe beträgt. Darüber hinaus sollte bei großen flachen Atrien von der Anwendung der o. g. Näherungsformeln (wie auch aller übrigen Zusammenhängen aus den Abschnitten 2.3 und 2.4) Abstand genommen werden. Weitere Anwendungsgrenzen sind im Abschnitt 1.3 enthalten.

Die entwickelten Gleichungen (60) bzw. (61) sind für alle Atriengrößen gültig, liefern jedoch für große Atrien tendenziell zu hohe Werte. Die Anwendung der im Abschnitt 2.4 dargestellten alternativen Berechnungsansätze, deren Gültigkeit sich auf große Atrien und ausreichend große Rauchgasreservoirs beschränkt, kann daher auch weiterhin zweckmäßig sein. Voraussetzung hierfür ist jedoch, dass der Begriff „groß“ näher quantifiziert werden kann. Derzeit gibt lediglich die BRE-Methode ein Kriterium zur Abgrenzung zwischen „großen“ und „kleinen“ Rauchgasreservoirs an ($(h+z)_{\text{int}} < 0,67 \cdot A_a^{0,5}$). Wie diese Arbeit an einem Beispiel aufzeigt, kann die Verwendung dieses Abgrenzungskriteriums jedoch zu zu geringen Massenströmen führen (vgl. Abs. 6.7.3). Im Hinblick auf die Anwendungsgrenzen für die Formeln aus Abschnitt 2.4 (Abgrenzung zwischen großen und kleinen Atrien) sieht der Autor daher derzeit Forschungsbedarf.

Die Anwendung der Gleichungen (60) bzw. (61) wird in dieser Arbeit auf eine Mindesthöhe des Rauchgasreservoirs im Atrium in Höhe von 25 % der Atrienhöhe begrenzt. In derzeitig gültigen Norm [36] wird diese Grenze weniger restriktiv mit 20 % der Atrienhöhe angegeben (vgl. Abs. 2.4.8, $z \leq 0,8 \cdot H_a$). Unter Berücksichtigung der vorhandenen Unschärfe bei der Abgrenzung zwischen rauchgasführender und rauchgasfreier Schicht, kann eine derartige Annahme durchaus gerechtfertigt sein. Die Gleichungen (60) bzw. (61) sind in diesem Fall entsprechend anzupassen.

Literaturverzeichnis

- [1] Fachverband Lichtkuppel, Lichtband und RWA e.V. (FVLR): Brandschadensstatistik des Statistik-Service Berlin. www.fvlr.de, 2002
- [2] H. P. Morgan et al.: Design methodologies for smoke and heat exhaust ventilation. BRE Report 368, Building Research Establishment, UK, 1999
- [3] H. P. Morgan: A brief note on the history of 'acceptable clear heights' for smoke and heat exhaust ventilation systems in the UK. International Journal on Engineering Performed-Based Fire Codes, Vol. 5, Issue 2, pp. 43–44, 2003
- [4] R. Harrison: Smoke Control in Atrium Buildings: A Study of Thermal Spill Plume. Fire Engineering Research Report 04/1, University of Canterbury, Christchurch, New Zealand, 2004
- [5] R. Harrison, M. Spearpoint: Characterisation of Balcony Spill Plume Entrainment using Physical Scale Modelling. 9th International Symposium on Fire Safety Science, 21.–26. Sep., Karlsruhe, Germany, 2008
- [6] W. W. Jones, R. D. Peacock et. al.: CFAST – Consolidated Model of Fire Growth and Smoke Transport (Version 6) Technical Reference Guide. NIST Special Publications 1026, 2009
- [7] National Institute of Standards and Technology (NIST): Fire Dynamics Simulator and Smokeview (FDS-SMV). <http://fire.nist.gov/fds>, 2010
- [8] H. P. Morgan, G. O. Hansell: Design approaches for smoke control in atrium buildings. Report BR 258, Building Research Establishment, UK, 1994
- [9] Y. Ko, G. Hadjisophocleous, G. D. Lougheed: CFD study of the air entrainment of balcony spill plumes at the balcony edge. ASHRAE Transaction, Vol. 114, pp. 344–354, 2008
- [10] R. Harrison, M. Spearpoint: Entrainment of Air into a Balcony Spill Plume. Journal of Fire Protection Engineering, Vol. 16, Issue 3, pp. 211–245, 2006
- [11] M. Poreh et. al.: Entrainment by Two-Dimensional Spill Plumes. Fire Safety Journal, Vol. 30, pp. 1–19, 1998
- [12] P. H. Thomas: On the Upward Movement of Smoke and Related Shopping Mall Problems. Fire Safety Journal, Vol. 12, pp. 191–203, 1987
- [13] S.-L. Lee, H. W. Emmons: A study of natural convection above a line Fire. Journal of Fluid Mechanics, Vol. 11, Issue 3, pp. 353–368, 1961
- [14] U. Schneider: Ingenieurmethoden im Brandschutz. Werner Verlag, 2. Auflage, 2009
- [15] L. U. Yuan, G. Cox: An experiment study of some line fires. Fire Safety Journal, Vol. 27, pp. 123–139, 1996
- [16] H. Rouse, C. S. Yih, H. W. Humphries: Gravitational convection from a buoyancy source. Tellus, Vol. 4, pp. 201, 1952
- [17] C. J. Chen, A. Rodi: A review of experimental data of vertical turbulent buoyant jets. Report SFB 80/T/69, University of Karlsruhe, 1975

- [18] E. E. Zukoski, T. Kubota, B. Cetegen: Entrainment in fire plumes. *Fire Safety Journal*, Vol. 3, pp. 107–121, 1980
- [19] S. Yokoi: Study on the prevention of fire spread caused by hot upward current. *Building Research Report 34*, Japanese Ministry of Construction, 1960
- [20] N. E. Kotsovinos: A study of the entrainment and turbulence in a plane buoyant jet. W. M. Keck Laboratory of Hydraulics and Water Resources, Report No. KH-R-32, California Institute of Technology, Pasadena, CA, 1975
- [21] B. R. Ramaprian, M. S. Chandrasekhara: Measurements in vertical plane turbulent plumes. *Journal of Fluids Engineering*, Vol. 111, pp. 69–77, 1989
- [22] B. S. Grove, J. G. Quintiere: Calculating entrainment and flame height in fire plumes of axisymmetric and infinite line geometries. *Journal of Fire Protection Engineering*, Vol. 12, pp. 117–137, 2002
- [23] H. P. Morgan, N. R. Marshall: Smoke control measures in a covered two-story shopping mall having balconies as pedestrian walkways. *BRE Current Paper CP 11/79*, Building Research Establishment, UK, 1979
- [24] H. P. Morgan, N. R. Marshall: Smoke hazards in covered multi-level shopping malls: an experimentally-based theory for smoke production. *BRE Current Paper CP 48/75*, Building Research Establishment, UK, 1975
- [25] H. P. Morgan: The Horizontal Flow of Buoyant gases toward an opening. *Fire Safety Journal*, Vol. 11, pp. 193–200, 1986
- [26] H. P. Morgan, G. O. Hansell: Atrium buildings: calculating smoke flows in atria for smoke control design. *Fire Safety Journal*, Vol. 12, pp. 9–35, 1987
- [27] H. P. Morgan, C. Williams et. al.: BATC – Hot smoke ventilation test at Brussels Airport. 1st International Conference on Fire Safety of Large Enclosed Spaces, 25.–27. Sep., Lille, France, 1995
- [28] C. Williams: In situ acceptance testing of smoke ventilation systems using real fires at the European Parliament Building. *Proceedings Eurofire 98'*, Fire safety by design, engineering and management. Brussels, Belgium, 1998
- [29] G. O. Hansell, H. P. Morgan, N. R. Marshall: Smoke flow experiments in a model atrium. *BRE Occasional Paper OP55*, Building Research Establishment, UK, 1993
- [30] M. Law: A Note on Smoke Plumes from Fires in Multi-level Shopping Malls. *Fire Safety Journal*, Vol. 10, pp. 197–202, 1986
- [31] H. P. Morgan: Comments on „A Note on Smoke Plumes from Fires in Multi-level Shopping Malls". *Fire Safety Journal*, Vol. 12, pp. 83–84, 1987
- [32] M. Law: Reply to Comments on "A Note on Smoke Plumes from Fires in Multi-level Shopping Malls". *Fire Safety Journal*, Vol. 12, p. 85, 1987
- [33] M. Law: Measurements of balcony smoke flow. *Fire Safety Journal*, Vol. 24, pp. 189–195, 1995
- [34] N. R. Marshall, R. Harrison: Experimental studies of thermal spill plumes. *BRE Occasional Paper OP1*, Building Research Establishment, UK, 1996

- [35] P. H. Thomas, H. P. Morgan et. al.: The Spill Plume in Smoke Control Design. Fire Safety Journal, Vol. 30, pp. 21–46, 1998
- [36] National Fire Protection Association: NFPA 92B – Standard for Smoke Management Systems in Malls, Atria, and Large Spaces. Quincy, USA, 2009
- [37] R. Harrison, M. Spearpoint: A Simple Approximation to Predict the Transition from a Balcony Spill Plume to an Axisymmetric Plume. Journal of Fire Protection Engineering, Vol. 20, pp. 273–289, 2010
- [38] G. Moore: Cramming more components onto integrated circuits. Electronics, Vol. 3, pp. 114–117, 1965
- [39] T. Rauber, G. Rünger: Multicore – Parallele Programmierung. Springer Verlag, Berlin, Heidelberg, 1. Auflage, 2008
- [40] NVIDIA: CUDA Compute Unified Device Architecture – Programming Guide. Version 3.2, August 2010
- [41] N. R. Mahapatra, B. Venkatrao: The processor-memory bottleneck: problems and solutions. ACM Crossroads, Vol. 5, Issue 3, 1999
- [42] NVIDIA Corporation: NVIDIA GeForce GTX 580 GPU Datasheet. www.nvidia.com, 2010
- [43] Intel Corporation: Intel® Xeon® Processor 5600 Series Datasheet. Vol. 1+2, March 2010
- [44] ANSI/IEEE Std 754-2008: IEEE Standard for Binary Floating-Point Arithmetic. New York, 2008
- [45] R. G. Rehm, H. R. Baum: The Equations of Motion for Thermally Driven Buoyant Flows. Journal of Research, Vol. 83 (3), pp. 297–308, 1978
- [46] M. Münch: Kritische numerische Aspekte bei der Anwendung von Feldmodellen. Vortrag zur 55. vfdb-Jahresfachtagung 21. bis 25. Mai 2006 in Salzburg, 2006
- [47] J. Smagorinsky: General Circulation Experiments with the Primitive Equations. Monthly Weather Review, Vol. 91 (3), pp. 99–164, 1963
- [48] McGrattan, K. et. al.: Fire Dynamics Simulator (Version 5) Technical Reference Guide. NIST Special Publications 1018-5, 2009
- [49] S. Kilian, M. Münch: A new generalized domain decomposition strategy for the efficient parallel solution of the FDS-pressure equation – Part I: Theory, Concept and Implementation. Konrad-Zuse-Zentrum für Informationstechnik Berlin, ZIB-Report 09–19, 2009
- [50] S. Kilian, M. Münch: A new generalized domain decomposition strategy for the efficient parallel solution of the FDS-pressure equation – Part II: Verification and Validation. Konrad-Zuse-Zentrum für Informationstechnik Berlin, ZIB-Report 09–20, 2009
- [51] H. R. Schwarz, N. Köckler: Numerische Mathematik. Vieweg + Teubner Verlag, 7. Auflage, 2009

- [52] C. Rogsch: FDS und OpenMP. Tagungsband zum zweiten Treffen der FDS-Usergroup in Berlin, S. 174–186, hhp press, 2008
- [53] C. Rogsch: FDS und OpenMP – Erfahrungen bei der Entwicklung. Tagungsband zum dritten Treffen der FDS-Usergroup in Berlin, S. 206–214, hhp press, 2009
- [54] The Portland Group: CUDA Fortran – Programming Guide and Reference. Version 1.0, November 2009
- [55] NVIDIA: CUDA Compute Unified Device Architecture – Toolkit. Version 3.2, Januar 2011
- [56] McGrattan, K. et. al.: Fire Dynamics Simulator (Version 5) User's Guide. NIST Special Publications 1019–5,
- [57] B. Karlsson, J. Quintiere: Enclosure Fire Dynamics. CRC Press LLC, 2000
- [58] R. Siegel, J. R. Howell, J. Lohrengel: Wärmeübertragung durch Strahlung: Teil 3 – Strahlungsübergang in absorbierenden, emittierenden und streuenden Medien. Springer-Verlag Berlin, Heidelberg, 1993
- [59] K. G. T. Hollands: Thermal Radiation Fundamentals. Begell House Inc., New York, 2004
- [60] M. F. Modest: Radiative Heat Transfer. 2nd Edition, Elsevier Science, USA, 2003
- [61] R. Siegel, J. R. Howell: Thermal Radiation Heat Transfer. Taylor & Francis, New York, 4. Edition, 2002
- [62] G. D. Raithby, E. H. Chui: A Finite-Volume Method for Predicting Radiant Heat Transfer in Enclosures with Participating Media. Journal of Heat Transfer, Vol. 112, Issue 2, pp. 415–423, 1990
- [63] G. D. Raithby, E. H. Chui: Computation of Radiant Heat Transfer on a Nonorthogonal Mesh using the Finite Volume Method. Numerical Heat Transfer, Part B, Vol. 23, Issue 3, pp. 269–288, 1993
- [64] S. W. Baek, M. Y. Kim, J. S. Kim : Nonorthogonal Finite-Volume Solutions of Radiative Heat Transfer in a Three-dimensional Enclosure. Numerical Heat Transfer, Part B, Vol. 34, Issue 4, pp. 419–437, 1998
- [65] E. H. Chui, G. D. Raithby, P. M. J. Hughes: Prediction of Radiative Heat Transfer in Cylindrical Enclosures with the Finite Volume Method. Journal of Thermophysics and Heat Transfer, Vol. 6, No. 4, pp. 605–611, 1992
- [66] J. P. Moder, J. C. Chai et. al.: Nonaxisymmetric Radiative Transfer in Cylindrical Enclosures. Numerical Heat Transfer, Part B, Vol. 30, Issue 4, pp. 437–452, 1996
- [67] M. Y. Kim, S. W. Baek: Radiative Heat Transfer in a Body-fitted Axisymmetric Cylindrical Enclosure. Journal of Thermophysics and Heat Transfer, Vol. 12, No. 4, pp. 596–599,
- [68] M. Ben Salah, F. Askri, et. al.: Numerical Resolution of the Radiative Transfer Equation in a Cylindrical Enclosure with the Finite-Volume Method. International Journal of Heat and Mass Transfer, Vol. 47, pp. 225–242, 2004

- [69] J. Y. Murthy, S. R. Mathur: Finite Volume Method for Radiative Heat Transfer using Unstructured Meshes. *Journal of Thermophysics and Heat Transfer*, Vol. 12, No. 3, pp. 313–321, 1998
- [70] J. Y. Murthy, S. R. Mathur: Radiative Heat Transfer in Axisymmetric Geometries using an Unstructured Finite-Volume Method. *Numerical Heat Transfer, Part B*, Vol. 33, Issue 4, pp. 397–416, 1998
- [71] Modest, M. F.: *Radiative Heat Transfer*. 2nd Edition, Elsevier Science, USA, 2003
- [72] J. C. Chai, H. S. Lee, S. V. Patankar: Finite Volume Method for Radiation Heat Transfer. *Journal of Thermophysics and Heat Transfer*, Vol. 8, No. 3, pp. 419–425, 1994
- [73] G. D. Raithby: Discussion of the Finite-Volume Method for Radiation and its Application using 3D Unstructured Meshes. *Numerical Heat Transfer*, Vol. 35, No. 4, pp. 389–405, 1999
- [74] S. H. Kim, K. Y. Huh: Assessment of The Finite-Volume Method and The Discrete Ordinate Method for Radiative Heat Transfer In a Three-Dimensional Rectangular Enclosure. *Numerical Heat Transfer, Part B*, Vol. 35, Issue 1, pp. 85–112, 1999
- [75] W. A. Fiveland: Discrete-Ordinates Solutions of the Radiative Transport Equation for Rectangular Enclosures. *Journal of Heat Transfer*, Vol. 106, pp. 699–706, 1984
- [76] W. Grosshandler: RADCAL – A Narrow Band Model for Radiation Calculations in a Combustion Environment. NIST Technical Note TN 1402, National Institute of Standards and Technology, Gaithersburg, Maryland, 1993
- [77] J. Gonçalves, P. J. Coelho: Parallelization of the Discrete Ordinates Method. *Numerical Heat Transfer, Part B*, Vol. 32, Issue 2, pp. 151–173, 1997
- [78] S. P. Burns, M. A. Christon: Spatial Domain-Based Parallelism in Large-scale Participating-Media Radiative Transport Applications. *Numerical Heat Transfer, Part B*, Vol. 31, Issue 4, pp. 401–421, 1997
- [79] P. J. Coelho, J. Gonçalves: Parallelization of the finite volume method for radiation heat transfer. *International Journal of Numerical Methods for Heat & Fluid Flow*, Vol. 9, No. 4, pp. 388–404, 1999
- [80] J. Liu, H. M. Shang, Y. S. Chen: Parallel Simulation of Radiative Heat Transfer Using an Unstructured Finite-Volume Method. *Numerical Heat Transfer, Part B*, Vol. 36, pp. 115–137, 1999
- [81] A. Fiterman, R. Ben-Zvi, A. Kribus: DOTS – Pseudo-Time-Stepping Solution of the Discrete Ordinate Equations. *Numerical Heat Transfer, Part B*, Vol. 35, pp. 163–183, 1999
- [82] J. Tal, R. Ben-Zvi, A. Kribus: A High-Efficiency Parallel Solution of the Radiative Heat Transfer Equation. *Numerical Heat Transfer, Part B*, Vol. 44, pp. 295–308, 2003
- [83] S. Matsuyama, T. Sakai et. al.: Parallel Computation of Fully Coupled Hypersonic Radiating Flowfield Using Multiband Model. *Journal of Thermophysics and Heat Transfer*, Vol. 17, No. 1, pp. 21–28, 2003

- [84] G. Krishnamoorthy, R. Rawat, P. J. Smith: Parallel Computations of Nongray Radiative Heat Transfer. *Numerical Heat Transfer, Part B*, Vol. 48, pp. 191–211, 2005
- [85] G. Krishnamoorthy, R. Rawat, P. J. Smith: Parallel Computations of Radiative Heat Transfer using the Discrete Ordinates Method. *Numerical Heat Transfer, Part B*, Vol. 46, pp. 19–38, 2005
- [86] Y. Azmy, Y. Yousry: Multiprocessing for neutron diffusion and deterministic transport methods. *Progress in Nuclear Energy*, Vol. 31, Issue 3, pp. 317–368, 1997
- [87] J. W. Fischer, Y. Y. Azmy: Comparison via parallel performance models of angular and spatial domain decompositions for solving neutral particle transport problems. *Progress in Nuclear Energy*, Vol. 49, pp. 37–60, 2007
- [88] K. R. Koch, R. S. Baker, R. E. Alcouffe: Solution of the first-order form of the 3D discrete ordinates equation on a massively parallel processor. *Transactions of the American Nuclear Society*, Vol. 65, pp. 198–199, 1992
- [89] A. Hoisie, O. Lubeck, H. Wasserman: Performance analysis of wavefront algorithms on very-large scale distributed systems. *Workshop on wide area networks and high performance computing*, 1999
- [90] A. Hoisie, O. Lubeck, H. Wasserman: Scalability analysis of multidimensional wavefront algorithms on large-scale SMP clusters. *Frontiers of Massively Parallel Computation*, 1999
- [91] A. Hoisie, O. Lubeck et. al.: A General Predictive Performance Model for Wavefront Algorithms on Clusters of SMPs. *2000 International Conference on Parallel Processing (ICPP'00)*, 2000
- [92] A. Hoisie, O. Lubeck, H. Wasserman: Performance and Scalability Analysis of Teraflop-Scale Parallel Architectures Using Multidimensional Wavefront Applications. *The International Journal of High Performance Computing Applications*, Vol. 14, No. 4, pp. 330–346, 2000
- [93] Los Alamos National Laboratory: SWEEP3D – 3D Discrete Ordinates Neutron Transport. <http://www.ccs3.lanl.gov/PAL/software.shtml>, 1995
- [94] M. M. Mathis, N. M. Amato, M. L. Adams: A general performance model for parallel sweeps on orthogonal grids for particle transport calculations. *Proceedings of the 14th international conference on Supercomputing*, Santa Fe, New Mexico, 2000
- [95] S. J. Plimpton, B. Hendrickson et. al.: Parallel Algorithms for Radiation Transport on Unstructured Grids. *Proceedings of the 2000 ACM/IEEE conference on Supercomputing*, Dallas, Texas, United States, 2000
- [96] S. D. Pautz: An Algorithm for Parallel Sn Sweeps on Unstructured Meshes. *Nuclear Science and Engineering*, Vol. 140, pp. 111–136, 2002
- [97] S. J. Plimpton, B. Hendrickson, S. P. Burns: Parallel Sn Sweeps on Unstructured Grids: Algorithms for Prioritization, Grid Partitioning and Cycle Detection. *Nuclear Science and Engineering*, Vol. 150, pp. 267–283, 2005

- [98] NVIDIA: The CUDA Compiler Driver NVCC. Version 2.0, April 2006
- [99] Intel Corporation: Intel Fortran Compiler 10.0 – User and Reference Guides.
<http://software.intel.com>, 2007
- [100] LSTM: Versuch am Lehrstuhl für Strömungsmechanik der Ruhr-Universität Bochum.
<http://www.lstm.ruhr-uni-bochum.de>, 2004
- [101] W. Kümmel: Technische Strömungsmechanik: Theorie und Praxis. Vieweg+Teubner Verlag, 3. Auflage, 2007
- [102] F. Hausmann, F. Hourscht, B. Stock: Simulation – sicher mit dem CFD-Programm FDS. HLH Lüftung/Klima, Heizung/Sanitär, Gebäudetechnik, Jg. 58, Nr. 8, S. 47–52, 2007
- [103] B. Stock, K. Wallasch: Sensitivitätsuntersuchungen zur FDS-Validierung anhand einfacher strömungsmechanischer Standardfälle – Der isotherme Freistrah. Beitrag zum zweiten Treffen der FDS-Usergroup in Berlin, 2008
- [104] Thomas et. al.: Investigations into the flow of hot gases in roof venting. Fire Research Technical Paper No 7, London, The Stationary Office, 1963
- [105] B. S. Massey: Mechanics of fluids. Van Nostrand Reinhold Company, London, 1990
- [106] M. L. Janssens, H. C. Tran: Data Reduction of Room Tests for Zone Model Validation. Journal of Fire Science, Vol. 10, pp. 528–555, 1992
- [107] M. J. Flynn: Some Computer Organizations and their Effectiveness. IEEE Transactions on Computers, Vol. 21, Issue 9, pp. 948–960, 1972
- [108] T. Rauber, G. Rüniger: Parallele Programmierung. Springer Verlag, Berlin, Heidelberg, 2. Auflage, 2007
- [109] E. Normand: Single Event Upset at Ground Level. IEEE Transactions on Nuclear Science, Vol. 43, Issue 6, pp. 2742–2750, 1996
- [110] J. von Neumann: First draft of a report on the EDVAC. Contract No. W-670–ORD–4926. reproduced in H. H. Goldstine, The computer: From Pascal to von Neumann. Princeton, NJ: Princeton University Press, 1972
- [111] A. Aho, M. Lam, R. Sethi et. al.: Compilers: Principles, Techniques & Tools. Pearson-Addison Wesley, 2007
- [112] R. Allen, K. Kennedy: Optimizing Compilers for Modern Architectures. Morgan Kaufmann, 2002
- [113] G. Amdahl: Validity of the Single Processor Approach to Achieving Large-Scale Computer Capabilities. AFIPS Conference Proceedings, Vol. 30, pp. 483–485, 1967
- [114] P. S. Pacheco: Parallele Programmierung with MPI. Morgan Kaufmann, 1997
- [115] M. Snir, S. Otto, et. al.: MPI – The Complete Reference. MIT Press, Cambridge, MA, 1996
- [116] R. Chandra, L. Dagum et. al.: Parallel Programming in OpenMP. Morgan Kaufmann, 2001

Anlagenteil A

Grundlagen paralleler Rechentechnik

A.1 Flynnsche Klassifizierung für Parallelrechner

Eine gängige Einordnung der Parallelrechner kann nach der Flynnschen Klassifizierung [107] erfolgen, mit der die heute verfügbaren Rechner in vier Modellsysteme unterteilt werden. Bei dem ersten Modellsystem, dem SISD-Rechner (Single Instruction – Single Data), handelt es sich streng genommen nicht um einen Parallelrechner, da jede Verarbeitungseinheit (Prozessor) Zugriff auf genau einen Datenspeicher und einen Programmspeicher hat. Die Daten und Instruktionen werden nacheinander aus den entsprechenden Speicherbereichen geladen, verarbeitet und wieder gespeichert. Der SISD-Rechner stellt damit das klassische Modell eines sequenziellen, nicht parallelen Rechners dar (vgl. Abb. 59, links).

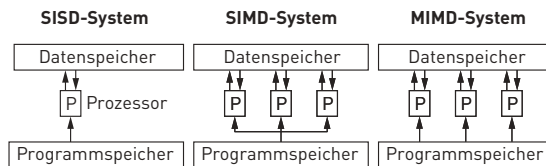


Abb. 59: Klassifikation von Parallelrechnern nach Flynn [107]

Das zweite Modellsystem, der SIMD-Rechner (Single Instruction – Multiple Data) besteht hingegen aus mehreren Verarbeitungseinheiten, die unterschiedliche Datensätze aus dem Speicher laden. Da ein SIMD-System lediglich über einen Programmspeicher verfügt, wird auf jeden Datensatz die gleiche Instruktion angewendet, bevor die Daten zurück gespeichert werden (vgl. Abb. 59, Mitte). Zu den SIMD-Rechnern gehören die sogenannten Vektorrechner, die seit vielen Jahren in wissenschaftlichen Rechenzentren im Einsatz sind. Programme, die auf SIMD-Systemen effizient laufen sollen, müssen „vektorisierbar“ sein, d.h. die Programmierweise muss eine möglichst unabhängige Berechnung einzelner Datensätze ermöglichen. Unter dem Aspekt sind auch klassische CFD-Programme nicht ohne Änderung der Programmstruktur auf eine SIMD-Architektur übertragbar. Zwar besitzen moderne PC-Prozessoren bereits SIMD-Erweiterungen, diese müssen jedoch speziell durch das Programm angesprochen werden und sind auch nur eingeschränkt nutzbar. Auch wenn reine SIMD-Systeme heute kaum noch eingesetzt werden, orientieren sich moderne, auf große Datenmengen spezialisierte Prozessorsysteme (z.B. Cell-Prozessoren, GPU-Prozessoren) in ihrem internen Aufbau an diesem Prinzip und setzen den SIMD-Ansatz mit teilweise großem Erfolg um.

Ein MIMD-Rechner (Multiple Instruction – Multiple Data) verfügt sowohl über mehrere Verarbeitungseinheiten als auch über mehrere lokale Programmspeicher, so dass Datensätze unabhängig voneinander geladen, verarbeitet und gespeichert werden können (vgl. Abb. 59, rechts). Die meisten derzeit verfügbaren Parallelrechner gehören zu dieser Rech-

nerklasse (z. B. Intel und AMD Multi-Core Prozessoren). Auch bei den MIMD-Rechnern können sequenzielle Programme nicht ohne (teilweise erhebliche) Änderungen in der Programmstruktur übernommen werden. Tatsächlich kann die Anpassung einer Software an einen MIMD-Rechner noch aufwendiger ausfallen, als dies bei SIMD-Systemen der Fall ist, da die einzelnen Prozessoren untereinander softwareseitig zusätzlich synchronisiert werden müssen. Der noch nicht erwähnte MISD-Rechner (Multiple Instruction – Single Data) hat keine praktische Relevanz. Detaillierte Ausführungen hierzu sowie eine vertiefende Betrachtung der übrigen Flynn'schen Modellsysteme sind in [108] enthalten.

A.2 Parallelrechner mit gemeinsamen oder verteiltem Speicher

Während die in Abschnitt A.1 erläuterte Flynn'sche Klassifizierung eher theoretischer Natur ist, weist die Klassifizierung von Parallelrechnern über die physikalische Anordnung des Arbeitsspeichers eine große praktische Bedeutung auf. Dabei werden im Wesentlichen zwei Klassen unterschieden. Rechnersysteme, bei denen die einzelnen Prozessoren auf einen gemeinsamen Arbeitsspeicher zugreifen können, werden als Multiprozessoren bzw. Shared-Memory-Systems (SMS) bezeichnet. Systeme, bei denen alle Prozessoren über einen eigenen Speicherbereich verfügen und bei denen die Prozessoren untereinander über ein Verbindungsnetzwerk kommunizieren, werden als Multicomputer bzw. Distributed-Memory-Systems (DMS) bezeichnet. In Abb. 60 sind beide Prinzipien vergleichend gegenübergestellt.

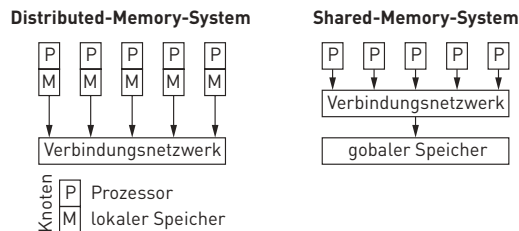


Abb. 60: Prinzipskizze zur Speicherorganisation bei Parallelrechnern

Multiprozessorsysteme (Shared-Memory-Systems) verfügen über eine hohe Geschwindigkeit beim Datenaustausch zwischen den Prozessoren, da jeder Prozessor Daten von und in den gemeinsamen Speicher schreiben kann. Insbesondere für Problemstellungen mit hohem Kommunikationsbedarf zwischen den beteiligten Prozessoren wirkt sich dies günstig auf die erforderliche Rechenzeit aus. Auch für den Programmierer bieten Multiprozessorsysteme Vorteile, da eine Verwaltung lediglich eines einzelnen Speicherbereichs leichter zu handhaben ist. Einzig und allein das gleichzeitige unkoordinierte

Schreiben auf ein und denselben Speicherbereich gilt es bei Shared-Memory-Systemen zu vermeiden. Die einzelnen Programmiermodelle (vgl. Abs. A.6) bieten hierfür unterschiedliche Ansätze zur Synchronisation.

Eine besondere Variante der Multiprozessorsysteme stellen die Symmetrischen Multiprozessoren (SMP) dar. Bei diesen Systemen sind alle Prozessortypen identisch und im Hinblick auf die Speicheranbindung gleichberechtigt ausgeführt. Im PC-Bereich sind für derartige Systeme Hauptplatinen mit zwei oder vier Prozessorsteckplätzen (Dual- oder Quad-Boards) verfügbar. Ebenfalls zu den SMP-Systemen gehören die in den letzten Jahren entwickelten Mehrkern- bzw. Multicore-Prozessoren. Bei diesen werden mehrere Prozessorkerne auf einem einzigen Chip untergebracht, was zu einem reduzierten Platzbedarf und günstigeren Produktionskosten führt. Nachteilig bei diesen Systemen ist vor allem, dass die Anbindung der einzelnen Prozessorkerne an den Hauptspeicher nur über einen einzigen Datenbus erfolgt, was zu einer Begrenzung der Rechenleistung des Gesamtsystems führen kann (s. Erläuterung zum „von-Neumannschen Flaschenhals“ im Abs. 3). Eine Kombination unterschiedlicher Shared-Memory-Systeme (z.B. Einsatz mehrerer Multicore-Prozessoren auf einer Hauptplatine) ist ebenfalls möglich.

Der Herstellungsaufwand für Multiprozessorsysteme steigt schnell mit der Anzahl der Prozessoren an, da die erforderliche Elektronik für den gemeinsamen Speicherzugriff und die effiziente Kommunikation zwischen den Prozessoren vergleichsweise kostenaufwendig ist. Shared-Memory-Systeme weisen aus diesem Grund selten mehr als 64 Prozessoren auf. Wesentlich mehr Prozessoren können eingesetzt werden, wenn auf den gemeinsamen Speicher verzichtet wird und stattdessen Systeme mit verteiltem Speicher eingesetzt werden.

Bei Systemen mit verteiltem Speicher (DMS) kann jeder Prozessor nur auf seinen eigenen Arbeitsspeicher zugreifen. Die Prozessoren kommunizieren untereinander über ein Netzwerk. Neben der theoretisch kaum begrenzten Prozessoranzahl liegt der Vorteil eines derartigen Systems vor allem in der einfachen Realisierung, da im Extremfall jeder Netzwerkknoten auch einem Desktoprechner entsprechen kann. Ein wesentlicher Nachteil besteht vor allem in der naturgemäß langen Zeit zur Nachrichtenübertragung innerhalb des Netzwerkes. Die Kommunikationszeiten im Netzwerk können die Kommunikation zwischen Prozessor und lokalem Hauptspeicher ohne weiteres um den Faktor 100 und mehr übersteigen.

Ein typisches Distributed-Memory-System stellt der PC-Computercluster dar. Beim PC-Computercluster handelt es sich um eine Ansammlung von Standard-PCs, die durch ein Netzwerk untereinander verbunden sind. Die einzelnen PCs im Cluster werden – entsprechend ihrer Aufgabe – als Serverknoten (Server Nodes) oder Rechenknoten (Compute Nodes) eingesetzt (s. Abb. 61). Während die Rechenknoten die eigentliche Rechenarbeit

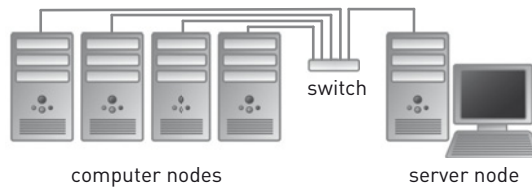


Abb. 61: Schematische Darstellung eines Computerclusters mit vier Rechen- und einem Serverknoten

erledigen, stellen die Serverknoten die zentrale Arbeitsplattform für die Verwaltung des Clusters dar (z.B. zur vollautomatischen Installation der Betriebssysteme auf den Rechenknoten etc.). Vom Prinzip her kann ein derartiger PC-Cluster in jedem Ingenieurbüro eingerichtet werden. Neben der Knotenanzahl sind beim Clusterbau vor allem Fragen nach den Prozessortypen, dem Arbeitsspeicher und der Netzwerkleistung zu klären. So muss bei kommunikationsintensiven Anwendungen eher auf ein gutes Netzwerk, bei speicher- und rechenintensiven Anwendungen eher auf leistungsfähige Rechenknoten Wert gelegt werden. Anleitungen und Hilfestellungen für den Aufbau eines PC-Computerclusters gibt es in der Literatur in großer Anzahl. Eine gute Übersicht über den Clusterbau ist beispielsweise in [6] enthalten.

A.3 ECC-Fehlerschutz für den Hauptspeicher

Bei der professionellen Verwendung von Parallelrechnern stellt sich immer auch die Frage nach deren Zuverlässigkeit. Fehler, die zu falschen Ergebnissen oder Systemabstürzen führen, können nie vollständig ausgeschlossen werden. Es ist jedoch für viele Anwendungen zweckmäßig, möglichst fehlertolerante Rechner einzusetzen. Eine typische Komponente derartiger Rechner stellt dabei der Schutz des Hauptspeichers vor Datenfehlern dar. Wie die Forschung zeigt, wird der überwiegende Teil von Datenfehlern in einem Speichermodul dabei von der kosmischen Hintergrundstrahlung und hier vor allem den Neutronen (Sekundärstrahlung) verursacht [109]. Diese Teilchen können direkt mit einzelnen Speicherzellen bzw. den Schaltkreisen zur Signalverarbeitung interagieren und einzelne Bits „umkehren“. Derartige Fehler werden auch als Soft Errors bezeichnet. Neben den zufälligen Soft Errors spielen auch Fehler eine Rolle, die durch physikalisches Rauschen bei der Signalübertragung oder Verunreinigungen im Siliziumchip verursacht werden. Darüber hinaus können vor allem im nicht professionellen Betrieb von Rechnersystemen Temperatur und Luftfeuchtigkeit erheblich schwanken, so dass die Elektronik in hohem Maße äußeren Störeinflüssen ausgesetzt ist.

Bei x86-Rechnern kommt für den Schutz des Arbeitsspeichers üblicherweise eine als ECC-Verfahren (Error-Correcting Code) benannte Technik zum Einsatz. Dabei wird im Hauptspeicher zu jeweils acht Byte ein redundantes, neuntes Byte gespeichert. Der

Inhalt dieses Bytes wird durch hardwareseitig implementierte Algorithmen aus den vorgegangenen acht Byte ermittelt und stellt eine Art „Kontrollbyte“ dar. Beim Lesezugriff prüft ein zweiter Algorithmus anhand der acht zusätzlichen Bits, ob sich die acht Byte Nutzdaten zwischenzeitlich verändert haben. Je nach verwendetem Algorithmus können in vielen Fällen Veränderungen nicht nur erkannt, sondern zeitgleich auch behoben werden. Das Betriebssystem erfährt in diesem Fall nichts von einem aufgetretenen Soft Error. Sofern Fehler größeren Ausmaßes vorliegen (z.B. das Kippen von mehreren Bits innerhalb der Nutzdaten) sollten ECC-Systeme diese Fehler zumindest erkennen, protokollieren und – sofern eine automatische Korrektur nicht möglich ist – die laufende Software zum Stillstand bringen.

Durch das redundante, neunte Byte benötigt ein ECC-Speichermodul (ECC-DIMM) gegenüber einem einfachen Speichermodul 12,5 % mehr Speicherplatz. ECC-DIMMs fallen daher etwas teurer aus. Darüber hinaus müssen im Speichercontroller geeignete Algorithmen implementiert werden, die für die Verwaltung der zusätzlichen Daten Sorge tragen. Da die Hersteller für diese Algorithmen häufig Lizenzen erwerben müssen, ergeben sich auch bei den Prozessoren zusätzliche Kosten.

Einzelne Soft Errors innerhalb des Arbeitsspeichers führen bei der Vielzahl von Datensätzen in einer numerischen Brand- und Rauchsimulation nicht immer zu signifikanten Ergebnisverfälschungen. Darüber hinaus treten Soft Errors vergleichsweise selten auf, so dass vor allem kürzere Berechnungsläufe häufig auch mit Systemen ohne ECC-Funktion fehlerfrei beendet werden. Demgegenüber stehen jedoch die sicherheitsrelevanten Aspekte beim Brandschutz. Da nicht ausgeschlossen werden kann, dass aus einer fehlerhaften Simulationsrechnung eine Gefahr für Leib und Leben resultiert, sollte ein höchstmögliches Maß an potentiellen Fehlerquellen ausgeschlossen und geschützter ECC-Speicher verwendet werden. Spätestens bei längeren Simulationsrechnungen wirkt sich die ECC-Funktion auch für den Anwender sinnvoll aus. Sofern Rechenläufe mehrere Tage oder Wochen dauern, kann ein Systemabsturz infolge eines Soft Errors kurz vor Ende der Berechnung sehr ärgerlich sein. Bei der Entscheidung für oder gegen ein System mit ECC-Funktion muss darüber hinaus berücksichtigt werden, dass allein die Verwendung von ECC-Speicher aus einem einfachen Desktop-PC noch kein zuverlässiges System macht. Das Fehlerisiko bei einem individuell zusammengestellten System ist ungleich größer als der Gewinn durch die ECC-Funktion.

A.4 Explizite und implizite Parallelisierung

Historisch bedingt basiert der überwiegende Teil der derzeit verfügbaren Software auf einer sequenziellen Programmstruktur, wie diese erstmalig von „von Neumann“ im Jahre 1945 vorgestellt wurde [110]. Dabei werden die Instruktionen vom Rechner schrittweise nacheinander bearbeitet (SISD-Architektur). Voraussetzung für die Verwendung von meh-

renen Prozessorkernen bzw. separaten Computern ist jedoch die Bereitstellung mehrerer paralleler Berechnungsströme. Eine vergleichsweise einfache Möglichkeit hierfür besteht in der Verwendung einer speziellen Compiler-Software, die voneinander unabhängige Arbeitsschritte erkennt und diese in parallele Anweisungen übersetzt. Der maßgebliche Vorteil dieser – auch als implizite Parallelisierung bezeichneten – Herangehensweise besteht vor allem in der Weiterverwendung der sequenziellen Algorithmen in bestehenden Software. Darüber hinaus ist der Quelltext einfach zu handhaben. Softwareentwickler müssen sich bei der impliziten Parallelisierung kaum Gedanken über die parallele Bearbeitung ihrer Algorithmen machen und können im bewährten Stil arbeiten.

Da viele Algorithmen jedoch eine vergleichsweise komplexe Struktur aufweisen, ist der Erfolg derartiger Parallel-Compiler im Hinblick auf die Recheneffizienz jedoch als sehr begrenzt einzustufen [111, 112]. Eine wesentlich höhere Effizienz lässt sich erreichen, wenn der Programmierer bereits bei der Entwicklung von Algorithmen Augenmerk auf die parallele Bearbeitung legt. Dazu muss er die Problemstellung vorab in mehr oder weniger stark gekoppelte, voneinander getrennte Teilprobleme zerlegen, die unabhängig voneinander bearbeitet werden können. Im Anschluss muss die Software um eine Methodik zur Bearbeitung der Kopplung zwischen den einzelnen Teilproblemen ergänzt werden. Diese, als explizite Parallelisierung bezeichnete Vorgehensweise kann in Abhängigkeit der Komplexität der Aufgabenstellung sowie des vorliegenden Programmiermodells sehr zeitaufwendig ausfallen.

A.5 Parallele Effizienz und Leistungssteigerung (SpeedUp)

Die maximale Beschleunigung, die bei der Verwendung eines parallelen Algorithmus gegenüber seinem sequenziellen Gegenstück erreicht werden kann, entspricht naturgemäß der Anzahl der für die Aufgabenstellung eingesetzten Prozessoren. Sofern für einen parallelen Algorithmus beispielsweise vier Prozessorkerne verfügbar sind, kann eine Leistungssteigerung gegenüber seinem sequenziellen Gegenstück um den Faktor vier nicht überschritten werden. Auf dieser Grundlage kann die parallele Effizienz wie folgt definiert werden:

$$\eta = \frac{T_s}{p T_p} \quad (62)$$

Dabei stellen T_s die für den sequenziellen und T_p die für den parallelen Algorithmus benötigten Ausführungszeiten und p die Anzahl der vorhandenen Prozessoren dar. Bei einer parallelen Effizienz in Höhe von $\eta = 1,0$ handelt es sich um einen theoretischen Wert, der in der Praxis nie erreicht wird, da durch die Parallelisierung immer ein zusätzlicher Zeitbedarf entsteht (Aufteilung der Arbeitsschritte auf die einzelnen Prozessoren, Kommunikation zwischen den Prozessoren etc.). Wird dieser systembedingte Zeitbedarf

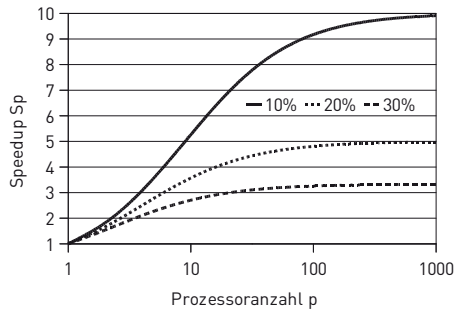
in der Betrachtung vernachlässigt, spricht man von der parallelen Recheneffizienz (parallel computational efficiency, PCE), die sich systemunabhängig ausschließlich aus den Eigenschaften des gewählten Algorithmus ergibt.

Neben den parallelen Algorithmen weisen Programme häufig Bereiche auf, die infolge von Abhängigkeiten der Daten untereinander zwingend sequenziell ausgeführt werden müssen (oder für die eine Anpassung an die parallele Rechentechnik zu aufwendig wäre). Sofern ein paralleles Programm derartige sequenzielle Bereiche enthält, kann die parallele Effizienz durch das Amdahlsche Gesetz quantitativ abgeschätzt werden [113]:

$$\eta = f + \frac{1-f}{p} \quad \text{mit} \quad f(0 \leq f \leq 1) \quad (63)$$

Bei der Abschätzung nach Gleichung (63) wird angenommen, dass die sequenziellen Anteile am Programm f mit höchstmöglicher Effektivität ausgeführt werden und sich die parallelen Anteile $(1 - f)$ „perfekt“ parallelisieren lassen (Vernachlässigung von Synchronisation und Kommunikation).

Abb. 62: Maximale Leistungssteigerung S_p (Speedup) in Abhängigkeit der Prozessoranzahl p für parallele Algorithmen mit unterschiedlichem Anteil sequenzieller Ausführung



Die Abb. 62 zeigt, dass der Anteil der sequenziellen Programmteile einen maßgeblichen Einfluss auf die mögliche Leistungssteigerung $S_p = 1 / \eta$ hat. So kann beispielsweise bei einem Programm mit sequenziellem Anteil in Höhe von 20 % unabhängig von der Anzahl der eingesetzten Prozessoren lediglich eine maximale Leistungssteigerung um den Faktor fünf erreicht werden.

A.6 Die Programmiermodelle MPI und OpenMP

Aufgrund der derzeit existierenden Vielzahl von Hardwarekomponenten gibt es eine große Anzahl unterschiedlicher Parallelrechner, für die die Programmierung auf eine ganz spezifische Art und Weise erfolgen müsste. Um eine Software nicht für jeden Rechartyp separat entwickeln bzw. anpassen zu müssen, bedient man sich so genannter Programmiermodelle, die alle wesentlichen Aufgaben der parallelen Programmierung beinhalten (Verteilung der Aufgaben auf die Prozessoren, Synchronisation etc.). Bei den Programmiermodellen handelt es sich im Wesentlichen um eine Vereinheitlichung von Befehlen oder Befehlssequenzen, die eine Umsetzung ein und desselben Programms auf Parallelrechnern mit unterschiedlichen Hardwarekomponenten erlaubt.

Für Parallelrechner mit verteiltem Arbeitsspeicher (vgl. Abs. A.2) ist das bekannteste Programmiermodell unter dem Begriff Message Passing Interface (MPI) bekannt. Ein MPI-Programm besteht dabei aus einer Anzahl von Prozessen, die ihrerseits Unterprogramme mit zugehörigen lokalen Daten ausführen. Der Informationsaustausch mit anderen Prozessen erfolgt über das explizite Verschicken von Nachrichten im Netzwerk. Jeder Prozess kann dabei entweder auf ein und demselben System oder einem Rechner im Netzwerkverbund ausgeführt werden. Obwohl vom Prinzip her jedes Unterprogramm vollkommen abweichende Instruktionen enthalten kann (MPMD, Multiple Program Multiple Data), führen im Regelfall alle Prozesse ein und dasselbe Unterprogramm mit verschiedenen lokalen Daten aus (SPMD, Single Program Multiple Data). Frei verfügbare MPI-Implementierungen sind z.B. MPICH2 vom Argonne National Lab und der Mississippi State University (<http://www.mcs.anl.gov/research/projects/mpich2>) sowie die von mehreren Universitäten und Unternehmen unterstützte Initiative Open MPI (<http://www.openmpi.org/>). Eine vollständige Beschreibung des MPI-Modells ist unter anderem in [114, 115] enthalten.

Für Parallelrechner mit gemeinsamem Arbeitsspeicher (vgl. Abs. A.2) muss ein Programmiermodell grundlegend andere Anforderungen erfüllen. Im Gegensatz zu einem Modell mit verteiltem Adressraum hat jede am Programm beteiligte ausführende Einheit Zugriff auf den gleichen Speicherbereich. In diesem Zusammenhang handelt es sich bei den ausführenden Einheiten nicht mehr um eigenständige Prozesse mit separatem Arbeitsspeicher sondern um Threads, die sich mit den jeweiligen anderen Threads des gleichen Prozesses einen Speicherbereich teilen. Im Unterschied zur MPI-Programmierung ist bei der Programmierung mit Threads keine Kommunikation über ein Netzwerkprotokoll notwendig. Stattdessen muss der Zugriff auf den gemeinsamen Speicher koordiniert werden, um zu verhindern, dass verschiedene Threads zum gleichen Zeitpunkt die gleiche Variable manipulieren. Diese Koordination zwischen den Threads wird auch als Synchronisation bezeichnet. Die Aufteilung der Instruktionen auf die Threads sowie die Thread-Synchronisation erfolgt im Allgemeinen durch besondere Sprachkonstrukte (vorgefertigte

Klassen und Methoden), die in einer Programmbibliothek zusammengefasst sind. Im wissenschaftlich-technischen Bereich stellt der OpenMP-Standard die am häufigsten verwendete Bibliothek zur Programmierung von Shared-Memory-Systemen dar. Er wurde von führenden Hard- und Softwareherstellern entwickelt, um eine einheitliche Programmierung für Parallelrechner mit gemeinsamen Speicher zu ermöglichen. Weiterführende Informationen zum OpenMP-Standard können [116] entnommen werden. Eine Kombination beider oben aufgeführten Programmiermodelle (z. B. Programmierung von Mehrkernprozessoren über den OpenMP-Standard innerhalb eines einzelnen Knotens auf einem MPI-Cluster) ist ebenfalls möglich.

Anlagenteil B

Beispielrechnung zur BRE Spill-Plume-Methode

Input values for the exsample scenario in [2], Annex D.1⁵

1. Convective heat flux leaving the room	Q_w [kW]	5000,000
2. Perimeter of fire	P [m]	12,000
3. Width of opening	W [m]	10,000
4. Height of the top of opening above floor	h [m]	5,000
5. Height of ceiling above floor	H [m]	15,000
6. Height of smoke base above spill edge	z [m]	3,600
7. Ambient temperature	θ_0 [°C]	15,000
8. Gas density at ambient temperature	ρ_0 [kgm ⁻³]	1,220
9. Specific heat at constant pressure of gas	c_p [kJkg ⁻¹ K ⁻¹]	1,010
10. Entrainment coefficient in 'Large-fire plume model'	C_e [kgs ⁻¹ m ^{-5/2}]	0,340
11. Effective coefficient of discharge for the opening	C_d [-]	1,000
12. Profile correction factor	k_M [-]	1,300
13. Profile correction factor	k_G [-]	0,950
14. Entrainment constant for air mixing into gases rotating around a horizontal edge	α' [-]	1,100
15. Entrainment constant for plume (0.077 and 0.16 for sigle-sided an double-sided edge)	α [-]	0,160

BRE procedure

16. Mass flow rate of gases approaching spill edge Gl. (5.7)	M_w [kg/s]	29,194
$M_w = \frac{C_e P W h^{3/2}}{\left[W^{2/3} + \frac{1}{C_d} \left(\frac{C_e P}{2} \right)^{2/3} \right]^{3/2}}$		
17. Mean layer temperature above ambient Gl. (E.1)	θ_w [°C]	169,575
$\theta_w = \frac{Q_w}{M_w c} \quad \theta_w = \frac{Q_w}{M_w c}$		
18. Maximum layer temperature above ambient Gl. (E.4)	θ_{cw} [°C]	232,050
$\theta_{cw} = \theta_w \frac{k_M}{k_G}$		
19. Depth of smoke layer approaching spill edge Gl. (E.3)	d_w [m]	1,140
$d_w = \left[\frac{3 M_w T_{cw}}{2 C_d^{3/2} k_M W \rho_0 (2 g \theta_{cw} T_0)^{1/2}} \right]^{2/3}$		
20. Layer's characteristic velocity Gl. (E.5)	v [ms ⁻¹]	3,709
$v = 0,96 \frac{C_d k_M}{k_G^{1/3}} \left[\frac{g Q_w T_{cw}}{c \rho_0 W T_0^2} \right]^{1/3}$		

⁵ Die in diesem Anhang verwendeten Formelzeichen entsprechend nicht der in dieser Arbeit verwendeten Konvention, sondern sind zur leichteren Nachvollziehbarkeit der Literaturquelle [2] entnommen.

21. Horizontal flux of vertical buoyant potential energy Gl. (E.8)	$B[m^{-1}]$	12,886
$B = \frac{\rho_0}{2} \frac{\theta_{cw}}{T_{cw}} g v d_w^2$		
22. Mass flux rising past the void edge Gl. (E.9)	$M_y[kg/s]$	72,480
$M_y = \frac{2}{3} \rho_0 W \alpha' (2 g \frac{\theta_c}{T_0})^{1/2} d_w^{3/2} + M_w$		
23. Upward mass flow rate per metre across the horizontal plate through the balcony	$A[kgs^{-1}m^{-1}]$	7,248
$A = M_y / W$		
24. Heat flux per unit width of gas flow	$Q_0[kWm^{-1}]$	500,000
$Q_0 = Q_w / W$		
25. Equivalent Gaussian Source Gl. (E.10) bis (E.14)	$\xi[m^2s^{-1}]$	4,146
$\xi = \left[A + \frac{Q_0}{T_0 c} \right] \frac{1}{\rho_0 \sqrt{\pi}}$		
26.	$[-]$ mit $\lambda = 0.9$	0,286
$\left[\frac{\theta}{T} \right]_G = \frac{Q_0 \sqrt{1 + \lambda^2}}{T_0 c \lambda \left[A + \frac{Q_0}{T_0 c} \right] \sqrt{\pi}}$		
27.	$\zeta [-]$	27,159
$\zeta = \frac{2B}{\rho_0 \left[\frac{1}{\sqrt{3}} - \left[\frac{\theta}{T} \right]_G \frac{\lambda}{\sqrt{1 + 3\lambda^2}} \right] \sqrt{\pi}}$		
28.	u_6	2,559
$u_6 = \sqrt{\frac{\zeta}{\xi}}$		
29.	b_6	1,620
$b_6 = \frac{\xi}{u_6}$		
30. Source Froude number Gl. (E.15) und (E.16)	F	0,452
$F = \left[\frac{2}{\pi} \right]^{1/4} \left[\frac{\alpha}{\lambda \left[\frac{\theta}{T} \right]_G} \right]^{1/2} \frac{u_6}{(g b_6)^{1/2}}$		
31.	v_6	1,079
$v_6 = \frac{1}{(1 - F^2)^{1/3}}$		
32.	$l_1(v_6)$	0,218
$v_6 \geq 1.549: l_1(v_6) = (v_6 - 0.75) / 0.9607$		
$v_6 \geq 1.242: l_1(v_6) = (v_6 - 0.843) / 0.8594$		
$v_6 \geq 1.059: l_1(v_6) = (v_6 - 0.9429) / 0.6243$		
$v_6 < 1.059: l_1(v_6) = (v_6 - 1.0) / 0.3714$		
33. Effective layer base for large area reservoir	$x[m]$	1,936
$x = (H - h) - 1,26 * (H - h - z)$		
34. Transformed height parameter Gl. (E.17) bis (E.22)	$x'[m]$	0,216
$x' = \frac{2}{\sqrt{\pi}} \alpha \frac{x}{b_6}$		
35.	$\Delta l_1(v)$	0,395
$\Delta l_1(v) = \frac{x'}{[F^2(1 - F^2)]^{1/3}}$		
36.	$l_1(v)$	0,614
$\Delta l_1(v) = \frac{x'}{[F^2(1 - F^2)]^{1/3}}$		

37.	$I_1(v) \geq 1.896: u'' = 1.0$ $I_1(v) \geq 1.786: u'' = 0.0908 I_1(v) + 0.821$ $I_1(v) < 0.786: u'' = I_1(v)^{0.35}$	u''	0,843
38.	$I_1(v) \geq 0.832: p'' = 0.9607 I_1(v) + 0.75$ $I_1(v) \geq 0.464: p'' = 0.8594 I_1(v) + 0.8429$ $I_1(v) \geq 0.186: p'' = 0.6243 I_1(v) + 0.9429$ $I_1(v) < 0.186: p'' = 0.3714 I_1(v) + 1.0$	p''	1,370
39.	$I_1(v) \geq 2.161: b'' = 0.938 I_1(v) + 0.820$ $I_1(v) \geq 1.296: b'' = 0.890 I_1(v) + 0.950$ $I_1(v) \geq 0.896: b'' = 0.810 I_1(v) + 1.070$ $I_1(v) \geq 0.650: b'' = 0.619 I_1(v) + 1.214$ $I_1(v) \geq 0.543: b'' = 0.330 I_1(v) + 1.414$ $I_1(v) \geq 0.421: b'' = 0.0627 I_1(v) + 1.55$ $I_1(v) \geq 0.348: b'' = -0.6 I_1(v) + 1.821$ $I_1(v) < 0.348: b'' = I_1(v)^{-0.4}$	b''	1,617
40.	$u' = u'' F^{1/3}$	u'	0,647
41.	$p' = \frac{1}{(1-F^2)^{1/3} p''}$	p'	0,788
42.	$b' = b'' [F^2(1-F^2)]^{1/3}$	b'	0,882
43.	Characteristic half-width of plume at height x $b = b' b_0$	$b[m]$	1,429
44.	Axial vertical velocity of gases at height x $u = \frac{u' u_0}{F}$	$u[m/s]$	3,664
45.	Mass flow per unit plume length at height x $m_r = \pi^{0.5} \rho_0 u b \left[1 - p' \left[\frac{\theta}{T} \right]_0 \frac{\lambda}{(1+\lambda^2)^{1/2}} \right]$	$m_r[kgs^{-1} m^{-1}]$	9,616
46.	Mass flow at height x $M_r' = m_r W$	$M_r'[kgs^{-1}]$	96,155
47.	$\bar{b} = \frac{(b_0 + b)}{2}$	\bar{b}	1,525
48.	$\bar{u} = \frac{(u_0 + u)}{2}$	\bar{u}	3,111
49.	$\delta M_r = 4 \bar{b} \bar{u} \alpha x \rho_0$	$\delta M_r[kgs^{-1}]$	7,171
50.	$M_r = m_r W + \delta M_r$	$M_r[kgs^{-1}]$	~105

Anlagenteil C

Auszug aus dem Quelltext zum Multi-Wellenfrontalgorithmus

```

--- CUDA-Routine für den Strahlungswärmetransport in gasförmigen Medien ---

extern "C" void cuc_radiation_fvm ()
{
--- Reinitialisierung des Arrays mit den Strahlungsintensitäten (IL) ---

    // Fill IL array
    func_kernel_set<<<FDS_ID_BIG(HMH->nILXYZ_Size)>>>(HMH->nILXYZ_Size, (REAL) \
        (BBFA * SIGMA / PI * G->TMPA4), DMH->ILX);
    CUT_CHECK_ERROR("Kernel func_kernel_set: execution failed");

--- CUDA-Kernel radi_kernel_bcilb_ex: Schleife über alle Wandzellen: Ermitteln der
    Strahlungsintensitäten in den angrenzenden Gaszellen ---

    // Boundary conditions: Intensities leaving the boundaries.
    radi_kernel_bcilb_ex<<<FDS_ID(HMH->NWC)>>>(HMH->DIMI, HMH->DIMIJ, HMH->NWC, \
        HMH->NDWC_Pitch, HMH->ILW_Pitch, NSTART, NEND, NSTEP, DMH->IJKW, \
        DMH->BOUNDARY_TYPE, DMH->ILW_IDX, BBFA, DMH->ILW, DMH->OUTRAD_W, \
        DMH->INRAD_W, DMH->E_WALL, G->DLN);
    CUT_CHECK_ERROR("Kernel radi_kernel_bcilb_ex: execution failed");

--- Ermitteln der Anzahl der zu verarbeitenden Kontrollwinkel ---

    // Get number of control angles to process
    int NCOUNT = 0; for (int N=NSTART; N >= NEND; N+=NSTEP) NCOUNT++;

--- Schleife über die Anzahl der Blockdiagonalen, vgl. Gl.(44) ---

    // Sweep through block diagonals
    int nBX = HMH->nWF_N_BX, nBY = HMH->nWF_N_BY, nBZ = HMH->nWF_N_BZ;
    int nSBlock = 0;
    for (int nD=1; nD <= HMH->nWF_N_DGL; nD++) {

--- Ermitteln der Blockanzahl für die jeweilige Diagonalebene nD, vgl. Gl.(45) ---

        // Number of blocks in this diagonal
        int nBlocks = nD*(nD+1)/2 - max(nD-nBX,0) * (nD-nBX+1)/2 - max(nD-nBY,0) * \
            (nD-nBY+1)/2 - max(nD-nBZ,0) * (nD-nBZ+1)/2 + max(nD-(nBX+nBY),0) * \
            (nD-(nBX+nBY)+1)/2 + max(nD-(nBX+nBZ),0) * (nD-(nBX+nBZ)+1)/2 \
            + max(nD-(nBY+nBZ),0) * (nD-(nBY+nBZ)+1)/2;

--- CUDA-Kernel radi_kernel_il_ex: Ermittlung der Intensitäten für alle Blöcke in
    einer Diagonalebene über einen KBA-Algorithmus je Block, s.unten ---

        // Compute IL in diagonal
        radi_kernel_il_ex<<<nBlocks, dim3(WF_TX, WF_TY)>>>(HMH->DIMI, HMH->DIMJ, \
            HMH->DIMJP, HMH->DIMK, HMH->DIMIJ, NSTART, NSTEP, NCOUNT, nSBlock, \
            MH->CELL_COUNT_Pitch, HMH->ILW_Pitch, DMH->WF_POSX, DMH->WF_POSY, \
            DMH->WF_POSZ, DMH->CELL_INDEX, DMH->SOLID, DMH->BOUNDARY_TYPE, \
            DMH->WALL_INDEX, DMH->ILW_IDX, DMH->DX, DMH->DY, DMH->DZ, G->DLN, G->DLX, \
            G->DLY, G->DLZ, G->RSA, DMH->ILW, EXTCOE, UIIEXT, DMH->ILX, DMH->ILY, \
            DMH->ILZ, DMH->INRAD_W, DMH->UIID);
        CUT_CHECK_ERROR("Kernel radi_kernel_il: execution failed");

        nSBlock += nBlocks;
    }
}

--- CUDA-Kernel radi_kernel_ifow (Schleife über alle Wandzellen): Ermittlung des
    Wärmestroms an den Wandoberflächen ---

    // Compute incoming flux on walls
    radi_kernel_ifow<<<FDS_ID(HMH->NWC)>>>(HMH->DIMI, HMH->DIMJ, HMH->DIMK, \
        HMH->DIMIJ, HMH->NWC, HMH->NDWC_Pitch, DMH->BOUNDARY_TYPE, DMH->IJKW, DS, \
        DMH->E_WALL, DMH->INRAD_W, DMH->QRADIN);
    CUT_CHECK_ERROR("Kernel radi_kernel_ifow: execution failed");
}

.
.
.

```

```

--- CUDA-Kernel radi_kernel_il_ex: Ermittlung der Intensitäten für alle Blöcke in
einer Diagonalebene über einen KBA-Algorithmus je Block: Der Kernel wird vom
System für jeden Block separat aufgerufen, wobei die Blocknummer in
blockIdx.x enthalten ist. ---

__global__
void radi_kernel_il_ex(int DIMI, int DIMJ, int DIMUp, int DIMK, int DIMIJ, \
int NSTART, int NSTEP, int NCOUNT, int nSBlock, int CELL_COUNT Pitch, \
int ILW Pitch, int *WF_POSX, int *WF_POSY, int *WF_POSZ, int *CELL_INDEX, \
int *SOLID, int *BOUNDARY_TYPE, int *WALL_INDEX, int *ILW_IDX, REAL *DX, \
REAL *DY, REAL *DZ, REAL *DLN, REAL *DLX, REAL *DLY, REAL *DLZ, REAL *RSA, \
REAL *ILW, REAL *EXTCOE, REAL *UIIEXT, REAL *ILX, REAL *ILY, REAL *ILZ, \
REAL *ENRAD_W, REAL *UIID)
{
    REAL ILXU, ILYU, ILZU, DXI, DYJ, DZK, VC, AX, AY, AZ, RAP, IL;
    int I, J, K, N, NT, IW, BT, ILW_IW; int3 STEP;
    bool bChanged;

--- Ermittlung der Startzelle (X,Y,Z) in Abhängigkeit der Blocknummer (die Arrays
WF_POSX, WF_POSY, WF_POSZ werden zu Programmstart initialisiert) ---

    // Compute start position for block (valid for all threads inside one block)
    int nB = nSBlock + blockIdx.x;
    __shared__ int3 WF_POS; WF_POS.x = WF_POSX[nB]; WF_POS.y = WF_POSY[nB];
    WF_POS.z = WF_POSZ[nB];

--- Schleife über die Anzahl der Diagonalebenen innerhalb des Blocks bei
NCOUNT gleichzeitig zu verarbeitenden Kontrollwinkeln, Gl.(46) ---

    // Sweep through all diagonals (d)
    for (int d=0; d < NCOUNT * WF_TZ + (WF_TX + WF_TY - 2); d++) {
        bChanged = false;

--- Ermittlung des zu bearbeitenden Kontrollwinkels aus der aktuellen Diagonale
(Multi-Wellenfrontalgorithmus) ---

        // Temporary Radiation Angle (0 ... G->NRA + G->ANGLE_INCREMENT - 1) /
        // G->ANGLE_INCREMENT - 1)
        NT = (d - TX - TY) / WF_TZ;

        // Resume only if thread in inside the block(s)
        if (NT >= 0 && NT < NCOUNT) {

            // Computes control angle to be processed
            N = NSTART + NT * NSTEP;

            // Specify direction values STEP for current block depended on N
            STEP.x = (DLX[N] < 0) ? -1 : 1; STEP.y = (DLY[N] < 0) ? -1 : 1;
            STEP.z = (DLZ[N] < 0) ? -1 : 1;

            // Compute cell position
            I = (1 - STEP.x)/2 * (DIMI-1) + STEP.x * (WF_POS.x + TX);
            J = (1 - STEP.y)/2 * (DIMJ-1) + STEP.y * (WF_POS.y + TY);
            K = (1 - STEP.z)/2 * (DIMK-1) + STEP.z * (WF_POS.z + (d - TX - TY) % WF_TZ);

            // Resume only for gas cells within the inner mesh
            if (I > 0 && I < DIMI-1 && J > 0 && J < DIMJ-1 && K > 0 && K < DIMK-1) {

                int IC = CELL_INDEX(I,J,K);

--- Ermittlung der Strahlungsintensität IL für die Gaszelle in Abhängigkeit der
benachbarten Gas- bzw. Wandzellen nach der FV-Methode ---

                if (!SOLID[IC]) {
                    bChanged = true;

                    // Compute IL for all non-solid cells
                    ILXU = ILX(J,K,NT);
                    ILYU = ILY(I,K,NT);
                    ILZU = ILZ(I,J,NT);

                    DXI = DX[I], DYJ = DY[J], DZK = DZ[K];

```

```

VC = DXI * DYJ * DZK;
AX =      DYJ * DZK * ABS(DLX[N]);
AY = DXI      * DZK * ABS(DLY[N]);
AZ = DXI * DYJ      * ABS(DLZ[N]);

if (IC != 0) {
  IW = WALL_INDEX(IC,-STEP.x);  BT = BOUNDARY_TYPE[IW];
  if (BT == SOLID_BOUNDARY) {AY *= (REAL) 0.5; AZ *= (REAL) 0.5;
    ILXU = ILW(ILW_IDX[IW-1], N);}
  IW = WALL_INDEX(IC,-STEP.y*2); BT = BOUNDARY_TYPE[IW];
  if (BT == SOLID_BOUNDARY) {AX *= (REAL) 0.5; AZ *= (REAL) 0.5;
    ILYU = ILW(ILW_IDX[IW-1], N);}
  IW = WALL_INDEX(IC,-STEP.z*3); BT = BOUNDARY_TYPE[IW];
  if (BT == SOLID_BOUNDARY) {AX *= (REAL) 0.5; AY *= (REAL) 0.5;
    ILZU = ILW(ILW_IDX[IW-1], N);}
}

RAP = (REAL) 1.0 / (AX + AY + AZ + EXTCOE(I,J,K) * VC * RSA[N]);
IL = FMAX(0., RAP * (AX*ILXU + AY*ILYU + AZ*ILZU + VC * RSA[N] * \
  UIEXT(I,J,K)));

--- Aktualisierung der Wand-Strahlungsintensitäten, sofern in Richtung des
Kontrollwinkels eine Wandzelle an die betrachtete Gaszelle angrenzt ---

// Compute incoming radiation in boundary cells
if (IC != 0) {
  IW = WALL_INDEX(IC,STEP.x);  BT = BOUNDARY_TYPE[IW];

  if (BT != NULL_BOUNDARY && BT != OPEN_BOUNDARY && \
    BT != POROUS_BOUNDARY) {
    ILW_IW = ILW_IDX[IW-1];
    // update incoming radiation
    INRAD_W[IW-1] += DLN(-STEP.x, N) * (ILW(ILW_IW, N) - IL);
    ILW(ILW_IW, N) = IL;
  }
  IW = WALL_INDEX(IC,STEP.y*2); BT = BOUNDARY_TYPE[IW];

  if (BT != NULL_BOUNDARY && BT != OPEN_BOUNDARY && \
    BT != POROUS_BOUNDARY) {
    ILW_IW = ILW_IDX[IW-1];
    // update incoming radiation
    INRAD_W[IW-1] += DLN(-STEP.y*2, N) * (ILW(ILW_IW, N) - IL);
    ILW(ILW_IW, N) = IL;
  }
  IW = WALL_INDEX(IC,STEP.z*3); BT = BOUNDARY_TYPE[IW];

  if (BT != NULL_BOUNDARY && BT != OPEN_BOUNDARY && \
    BT != POROUS_BOUNDARY) {
    ILW_IW = ILW_IDX[IW-1];
    // update incoming radiation
    INRAD_W[IW-1] += DLN(-STEP.z*3, N) * (ILW(ILW_IW, N) - IL);
    ILW(ILW_IW, N) = IL;
  }
}

--- Aufsummierungen der Strahlungsintensitäten über die Kontrollwinkel ---

// Calculate integrated intensity UIID (replaces kernel "radi_kernel_uuid")
UIID(I,J,K) += RSA[N] * IL;
}
}

--- Synchronisationspunkt: Ende einer Blockdiagonale ---

// Store new IL-values if bufil was new computed
__syncthreads(); if (bChanged) ILX(J,K,NT) = ILY(I,K,NT) = ILZ(I,J,NT) = IL;

// Wait for all threads
__syncthreads();
}
}

```

Anlagenteil D

Beispiele zur Verifizierung der GPU-Version

D.1 Isothermer Freistrahл ohne Vorströmung (Modellgeometrie A)

Geschwindigkeitsverlauf in der Symmetrieachse gemittelt zwischen $t = 120$ und 150 s nach Beginn der Simulation sowie das dazugehörige qualitative Geschwindigkeitsprofil für unterschiedliche Gitterweiten

Abb. 63: Isothermer Freistrahл ohne Vorströmung, Modell mit Gitterweite $\Delta l = 10$ cm

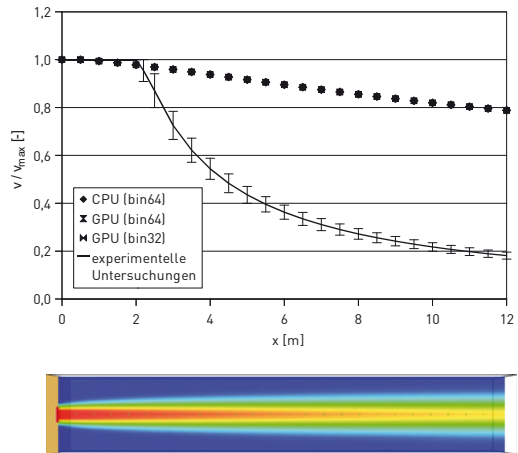


Abb. 64: Isothermer Freistrahл ohne Vorströmung, Modell mit Gitterweite $\Delta l = 5,00$ cm

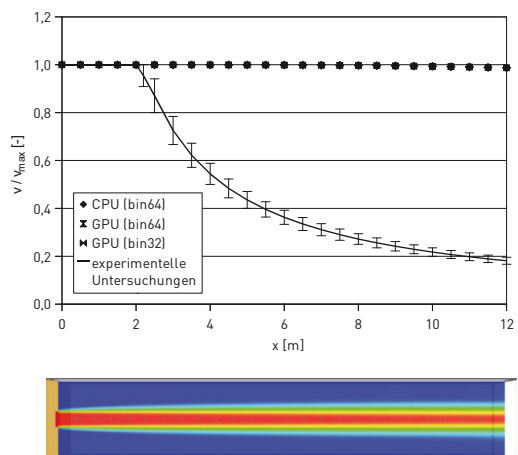


Abb. 65: Isothermer Freistrah
ohne Vorströmung, Modell mit
Gitterweite $\Delta l = 2,50$ cm

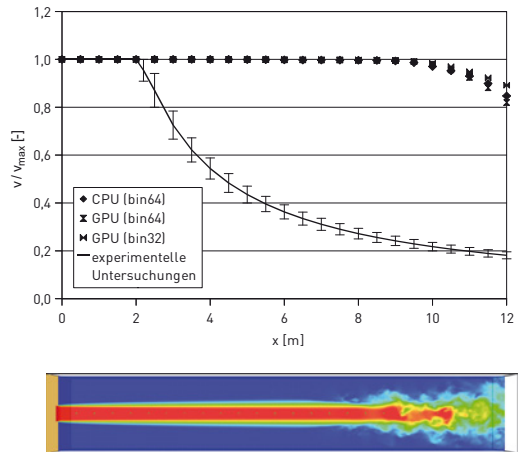


Abb. 66: Isothermer Freistrah
ohne Vorströmung, Modell mit
Gitterweite $\Delta l = 2,00$ cm

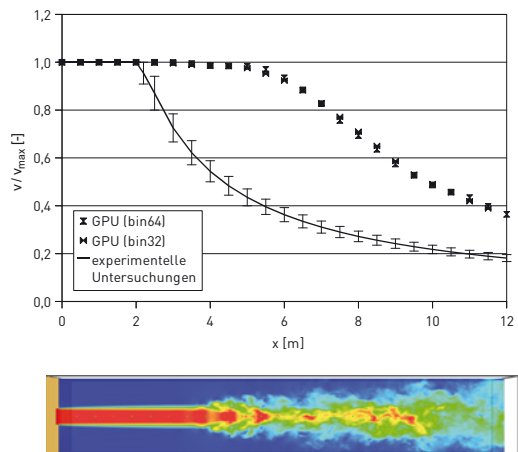


Abb. 67: Isothermer Freistrah
ohne Vorströmung, Modell mit
Gitterweite $\Delta l = 1,67$ cm

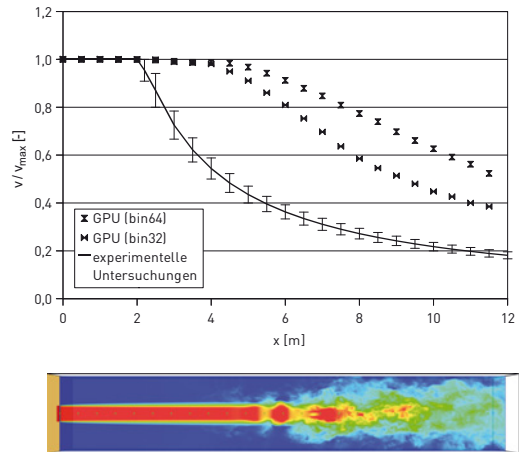
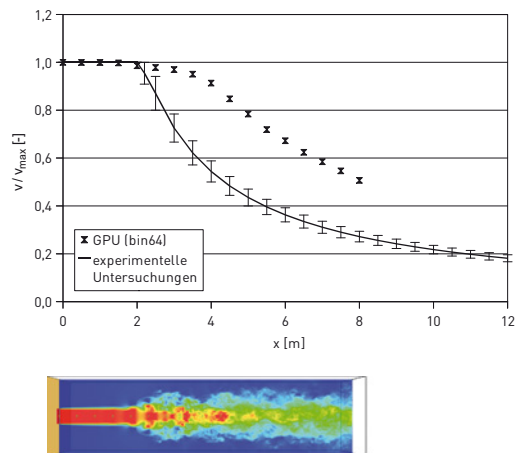


Abb. 68: Isothermer Freistrah
ohne Vorströmung, Modell mit
Gitterweite $\Delta l = 1,25$ cm



D.2 Isothermer Freistrah mit Vorströmung (Modellgeometrie B)

Geschwindigkeitsverlauf in der Symmetrieachse gemittelt zwischen $t = 120$ und 150 s nach Beginn der Simulation sowie das dazugehörige qualitative Geschwindigkeitsprofil für unterschiedliche Gitterweiten

Abb. 69: Isothermer Freistrah mit Vorströmung, Modell mit Gitterweite $\Delta l = 4,0$ cm

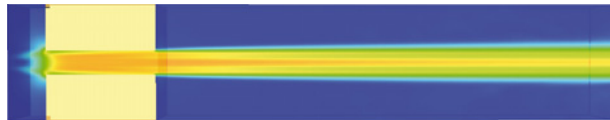
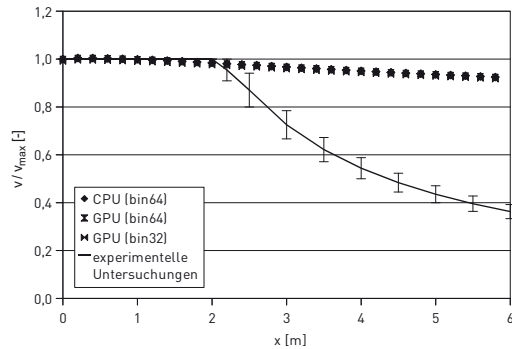


Abb. 70: Isothermer Freistrah mit Vorströmung, Modell mit Gitterweite $\Delta l = 2,0$ cm

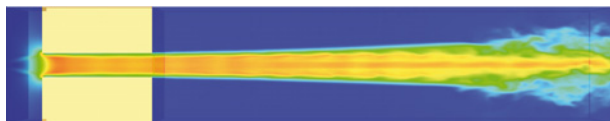
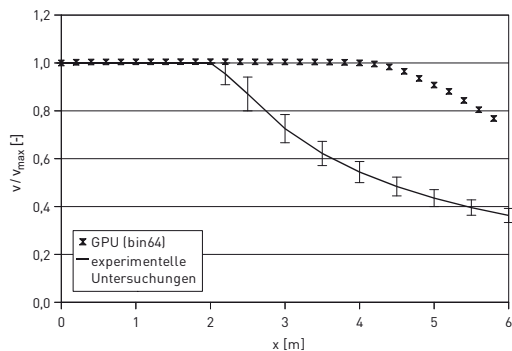


Abb. 71: Isothermer Freistrah
mit Vorströmung, Modell mit
Gitterweite $\Delta l = 1,67$ cm

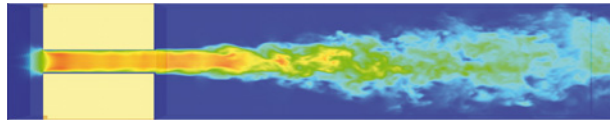
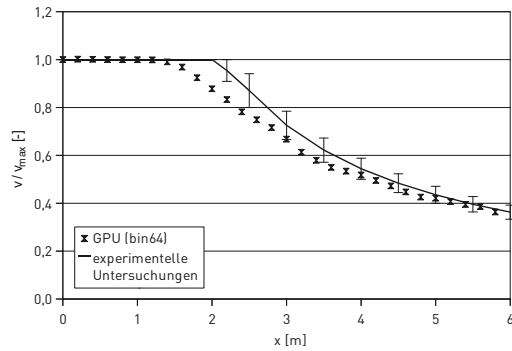


Abb. 72: Isothermer Freistrah
mit Vorströmung, Modell mit
Gitterweite $\Delta l = 1,25$ cm

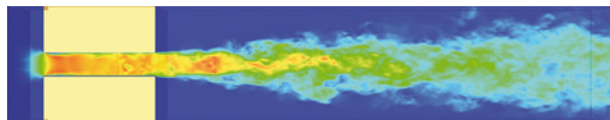
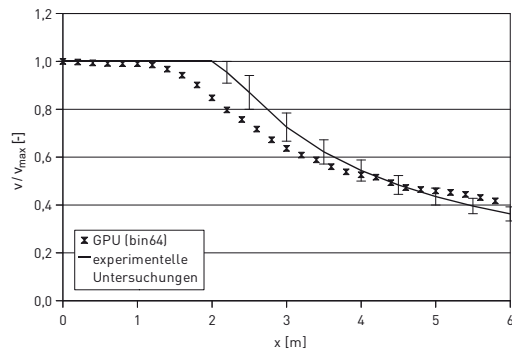


Abb. 73: Isothermer Freistrah
mit Vorströmung, Modell mit
Gitterweite $\Delta l = 2,00$ cm

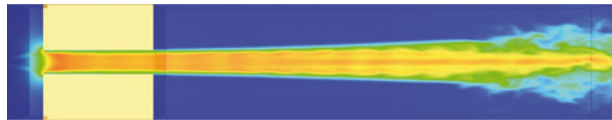
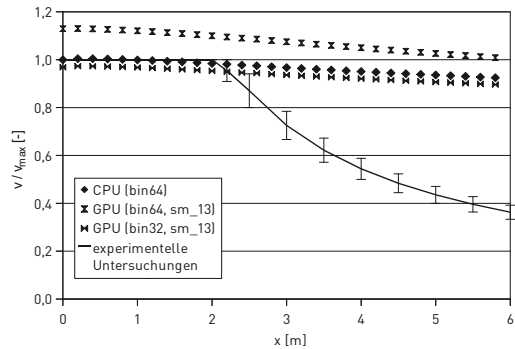
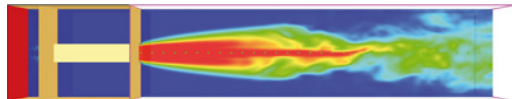
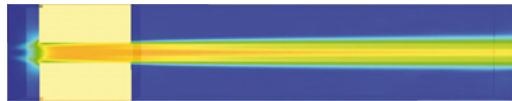
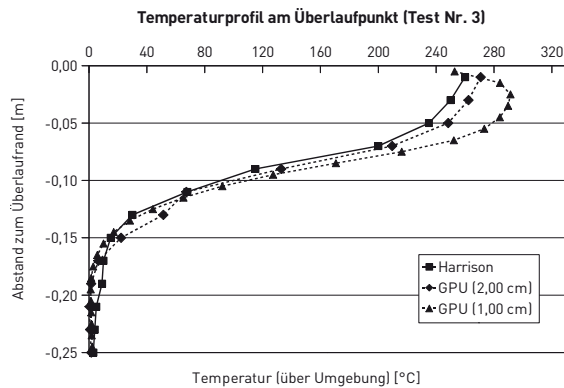
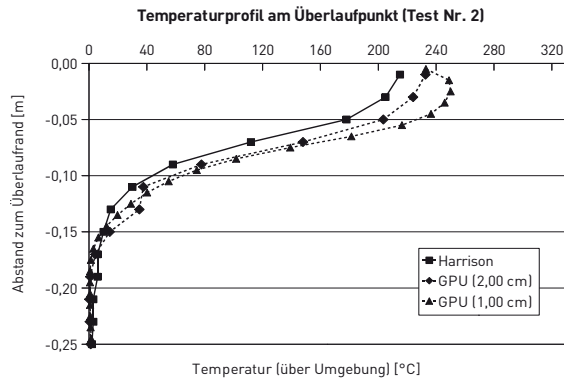
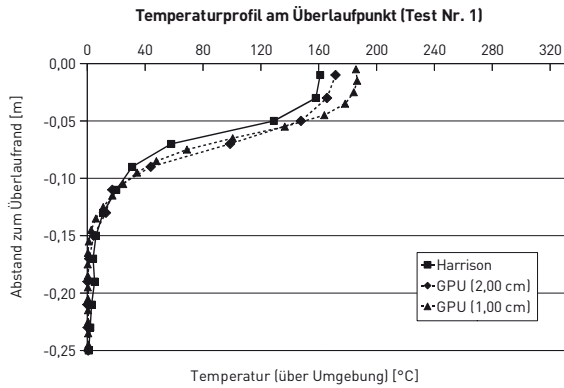


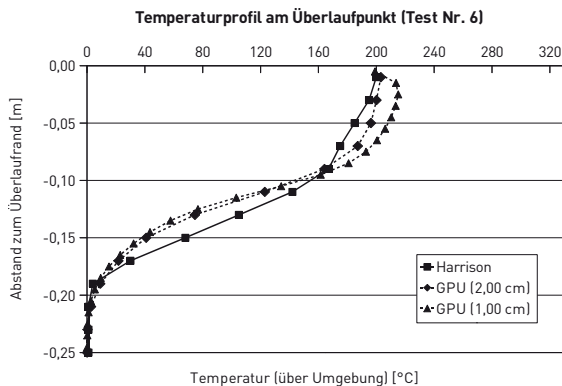
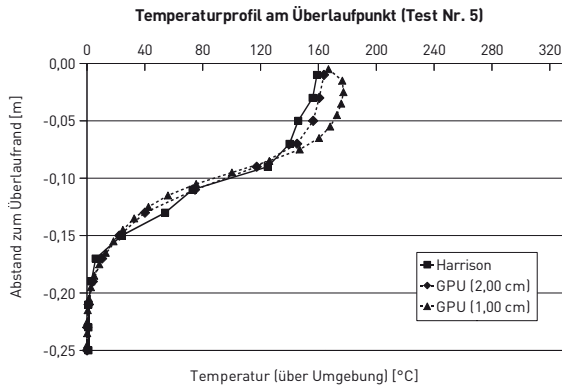
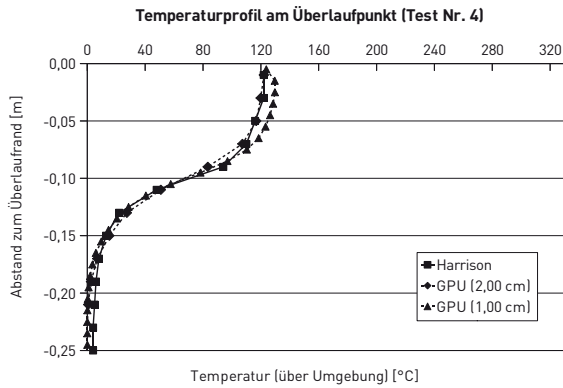
Abb. 74: Gegenüberstellung
des Strömungsbildes für die
Modellgeometrie B (oben) und
für die Modellierung mit Wan-
delementen der Dicke Null
($\Delta l = 4,0$ cm)

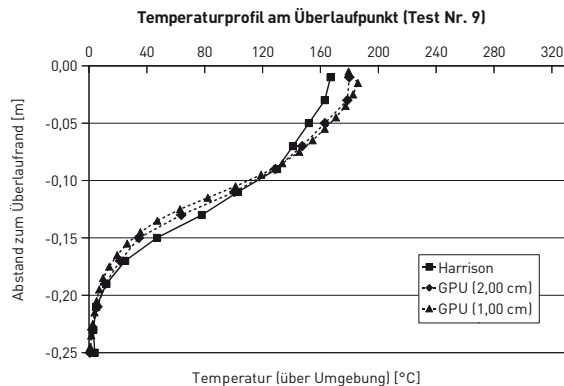
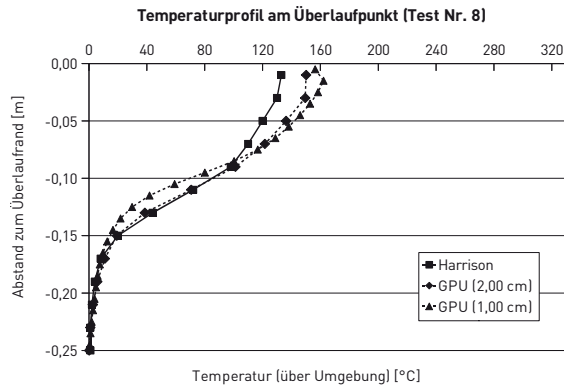
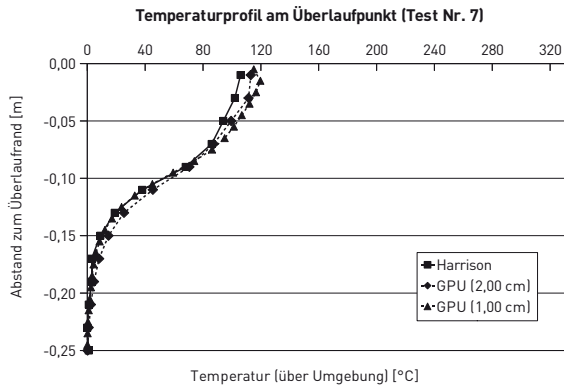


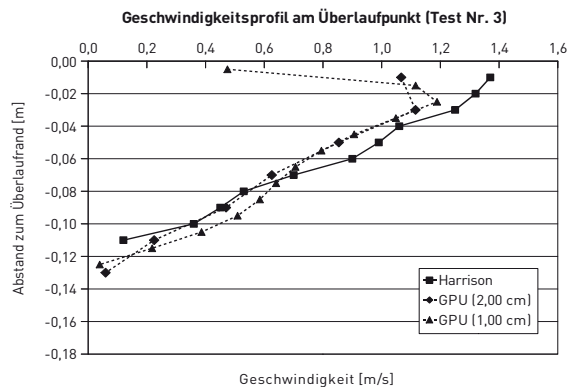
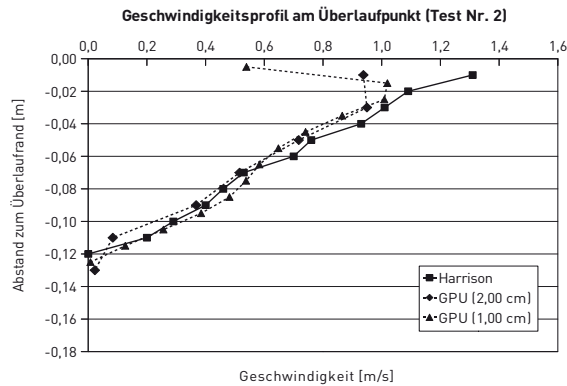
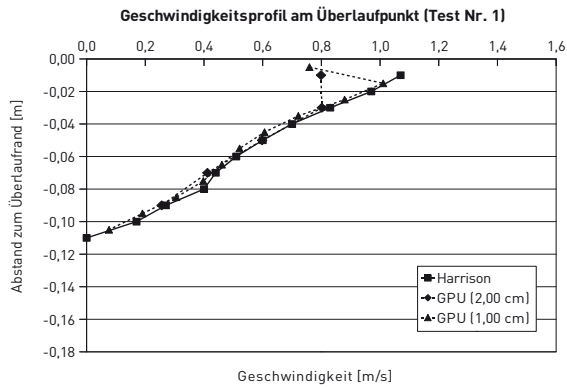
Anlagenteil E

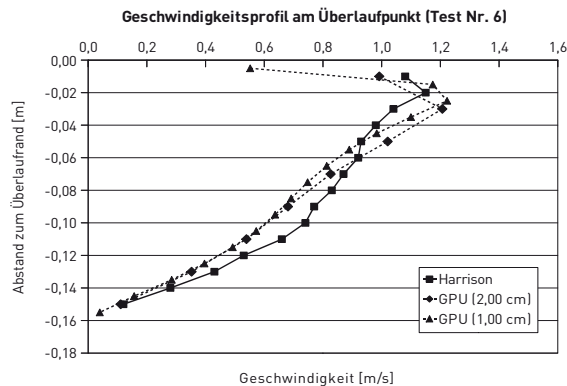
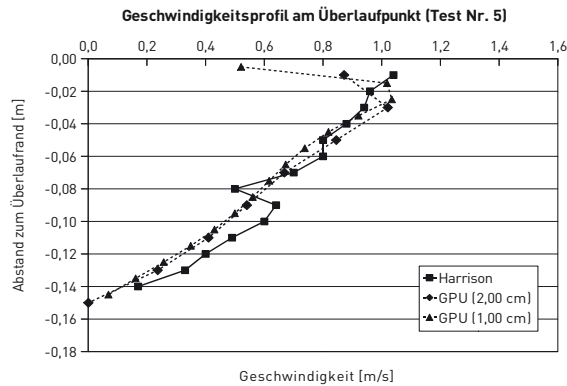
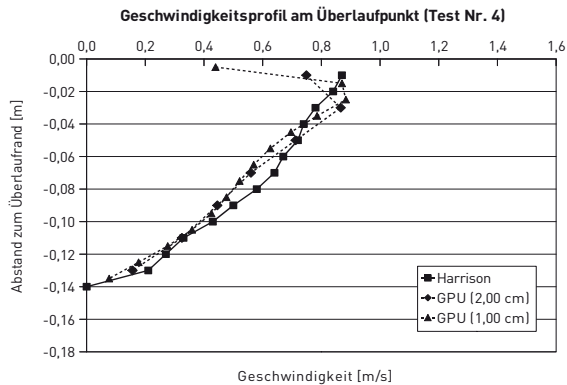
Parameterstudie zu Spill Plumes in Atrien

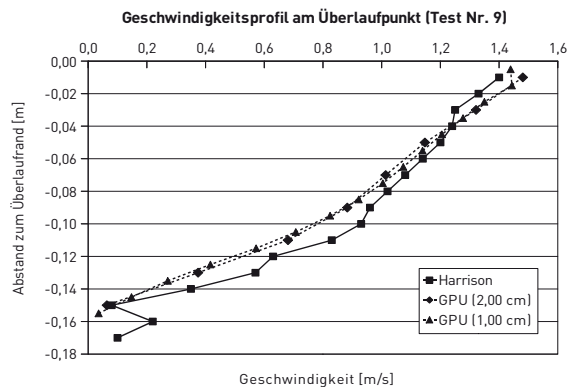
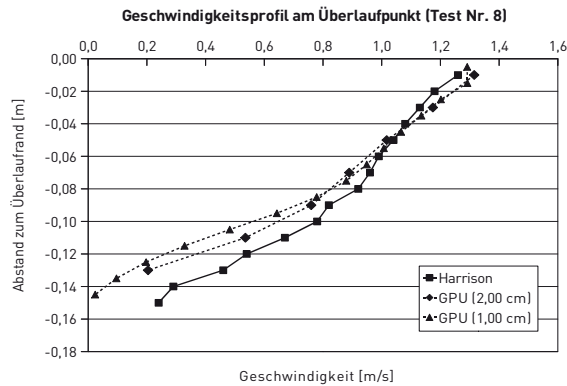
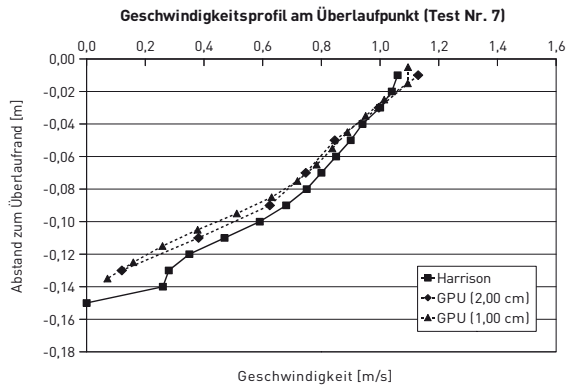


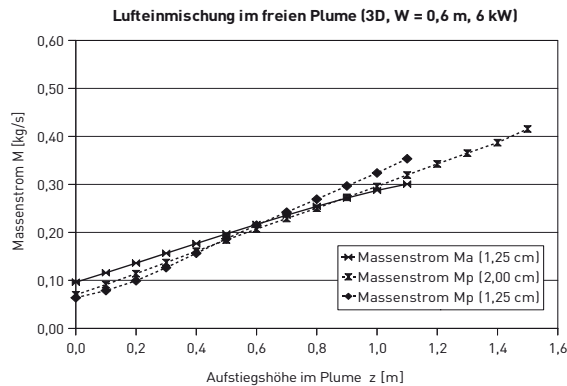
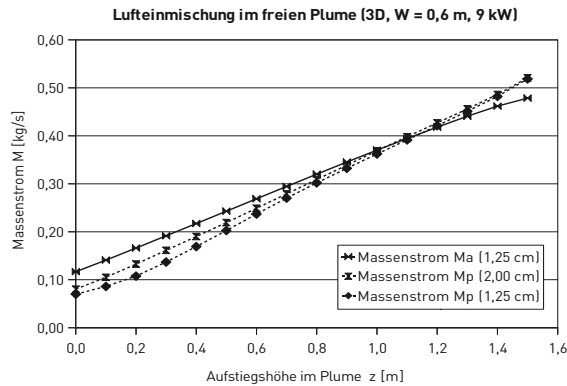
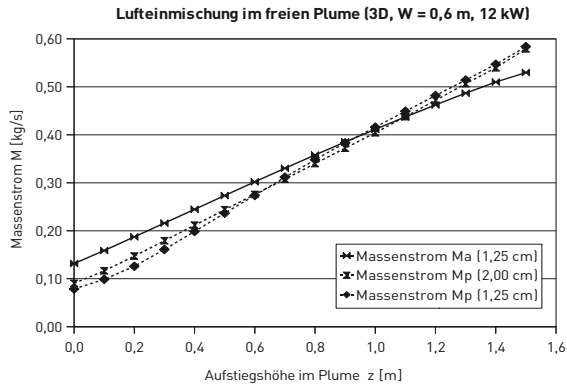


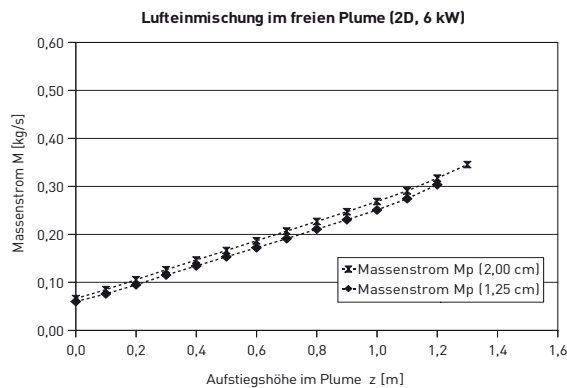
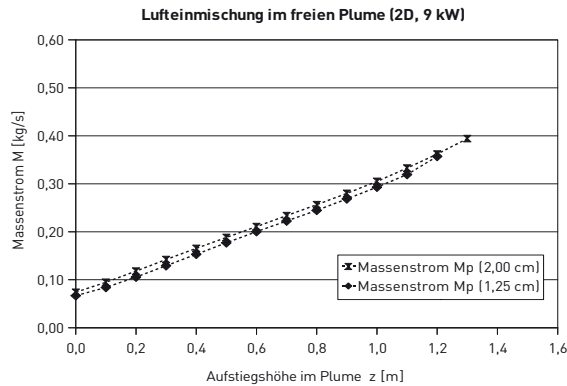
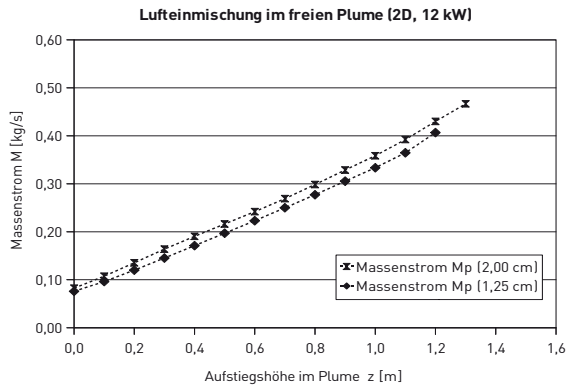


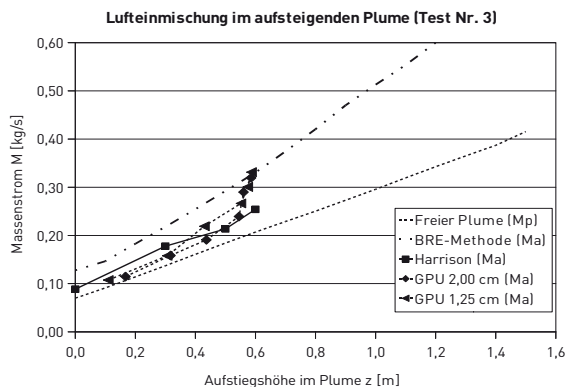
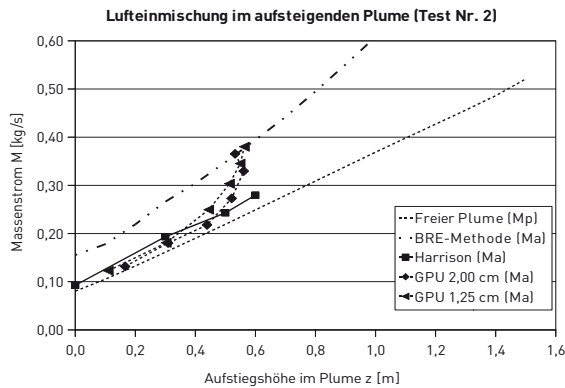
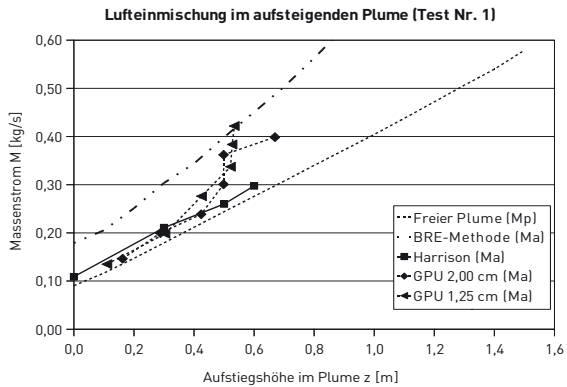


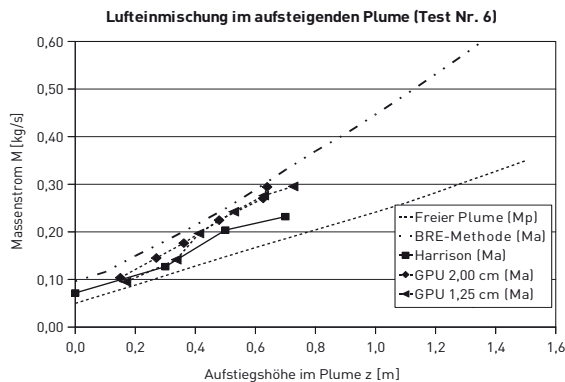
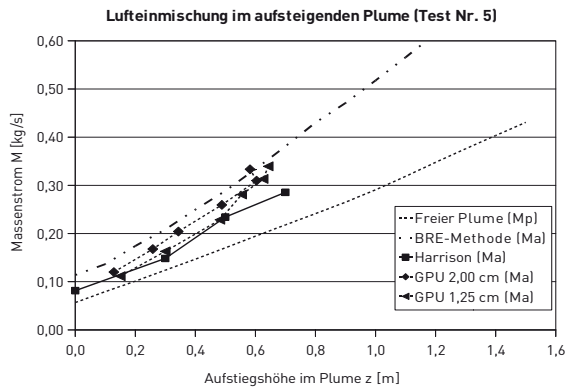
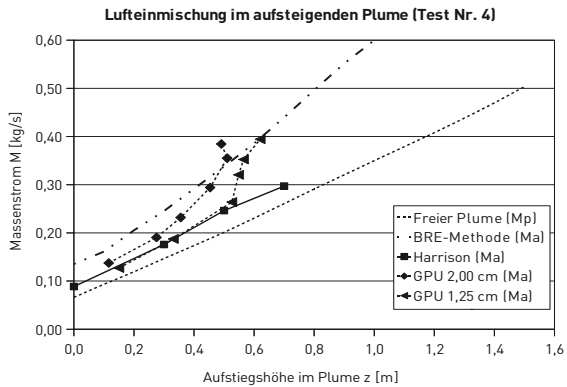


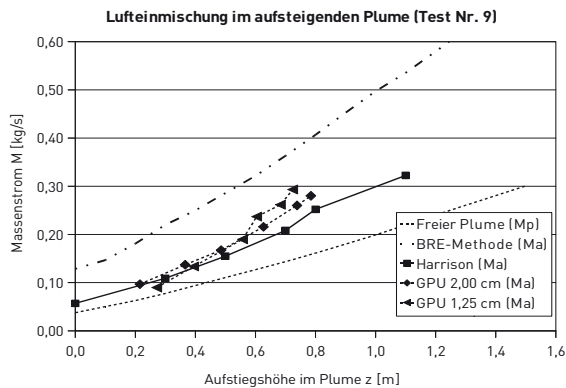
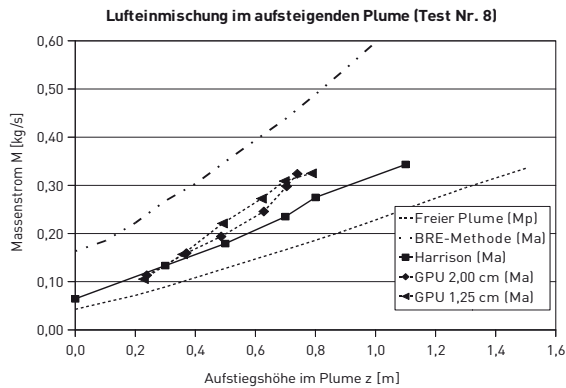
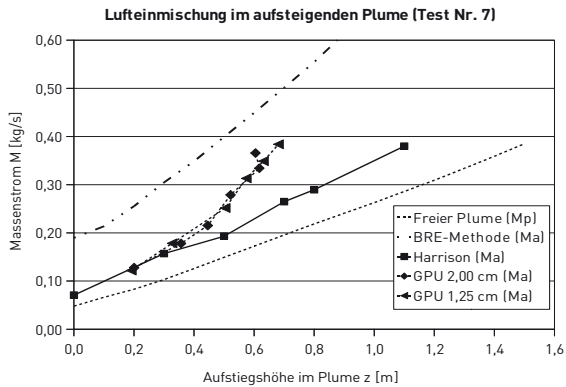


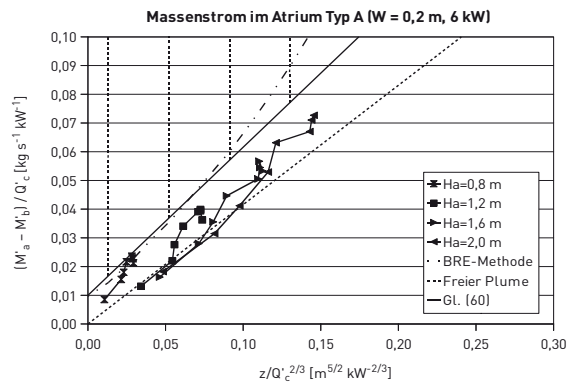
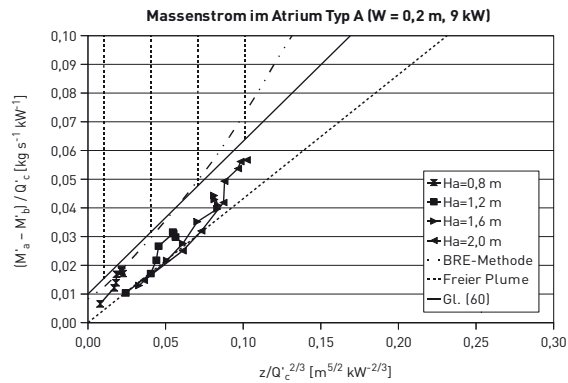
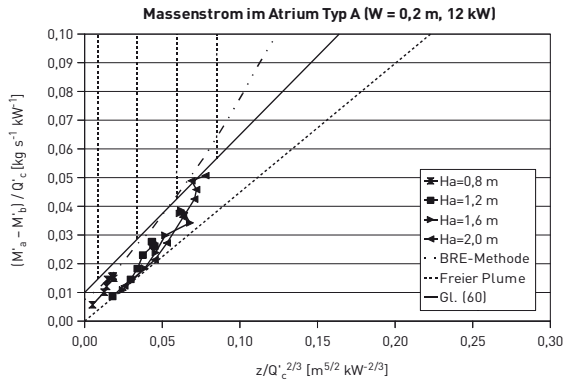


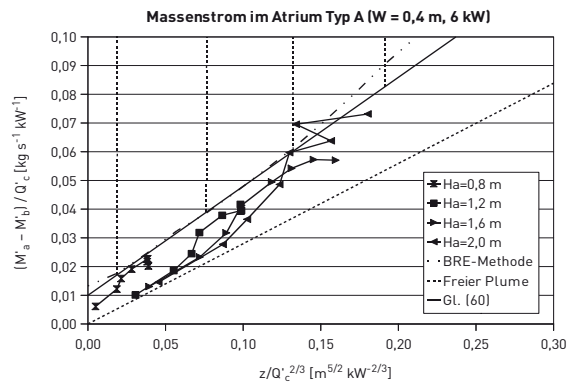
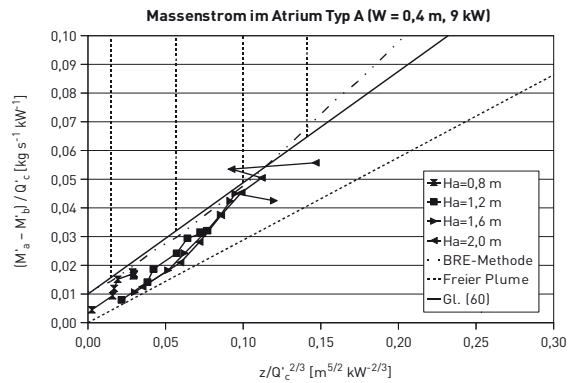
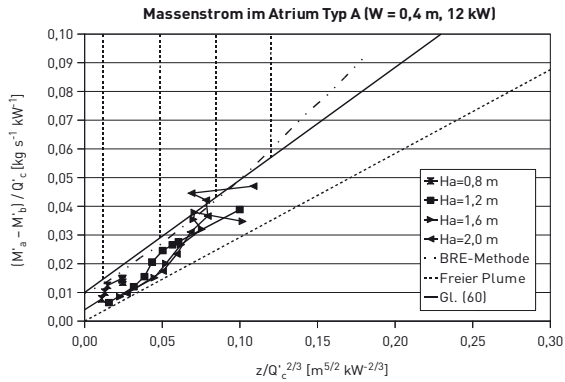


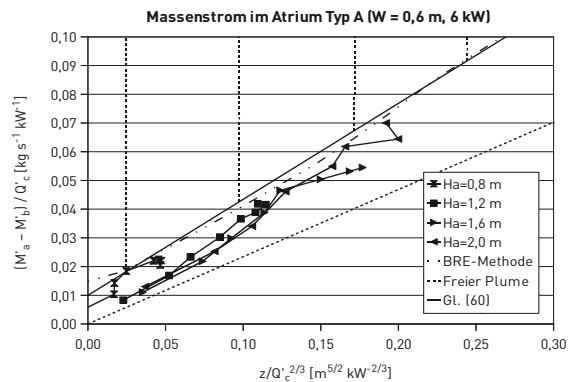
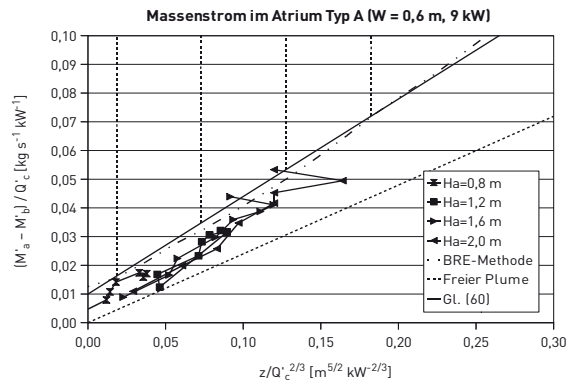
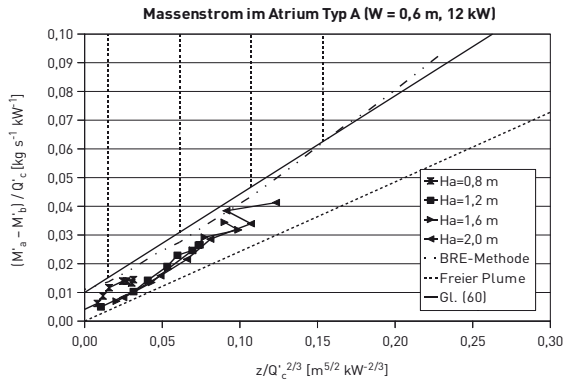


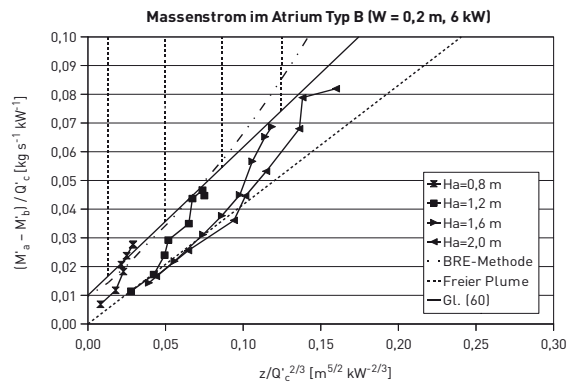
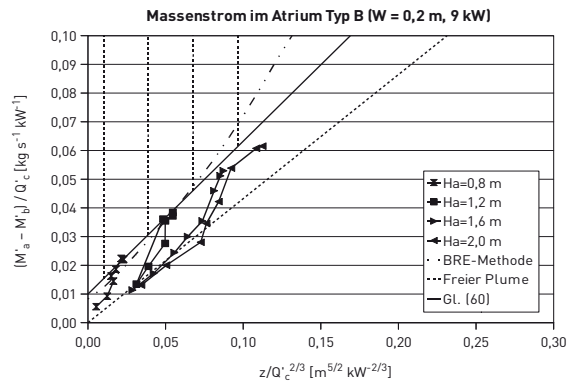
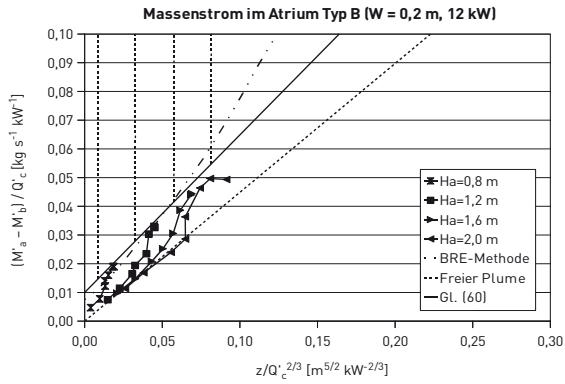


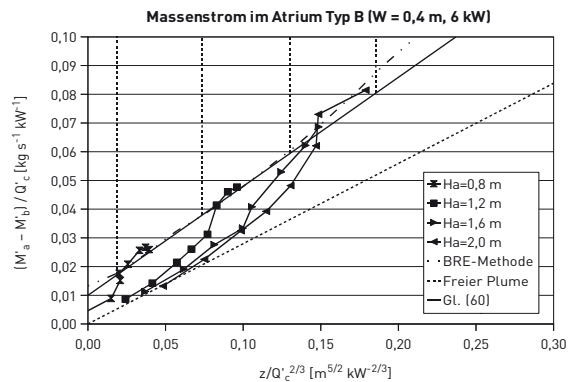
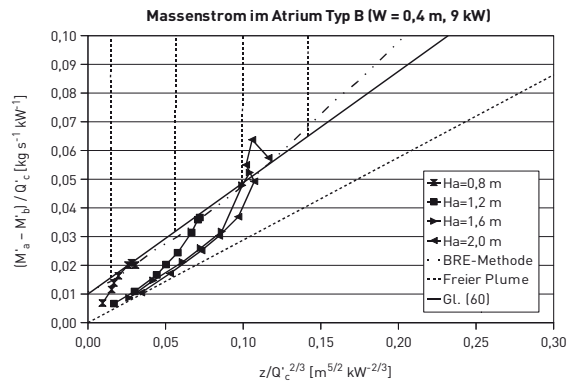
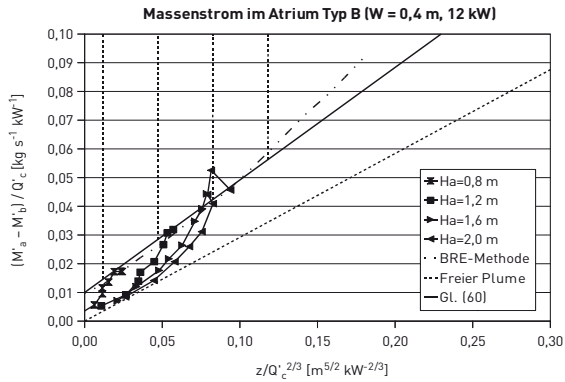


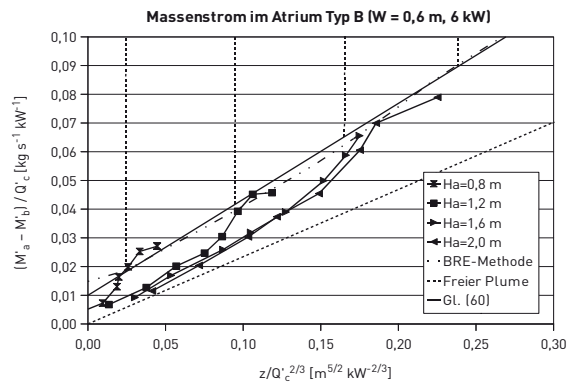
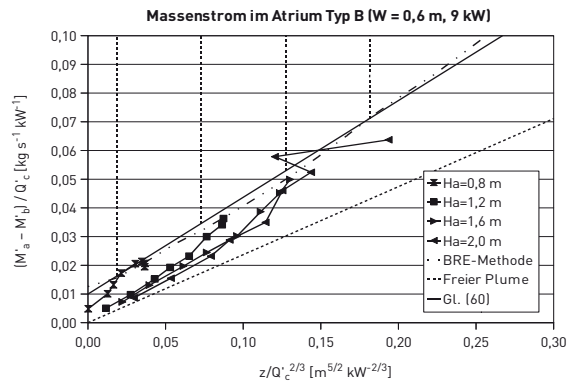
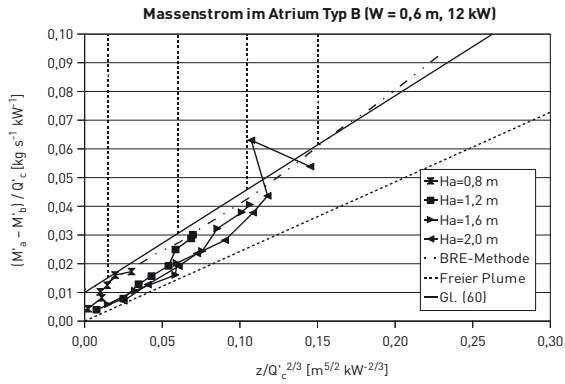


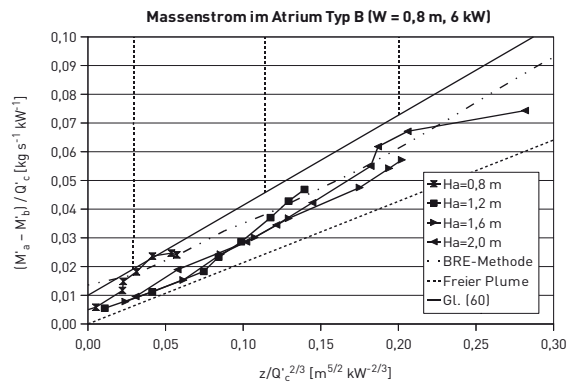
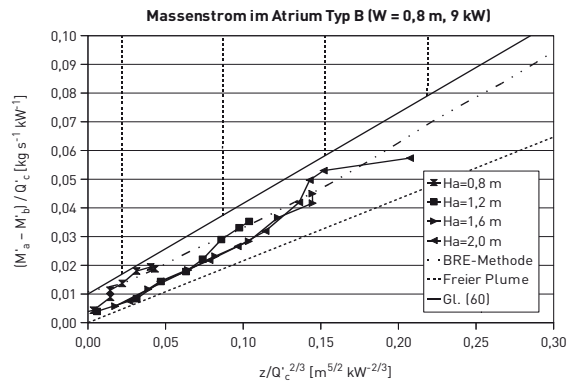
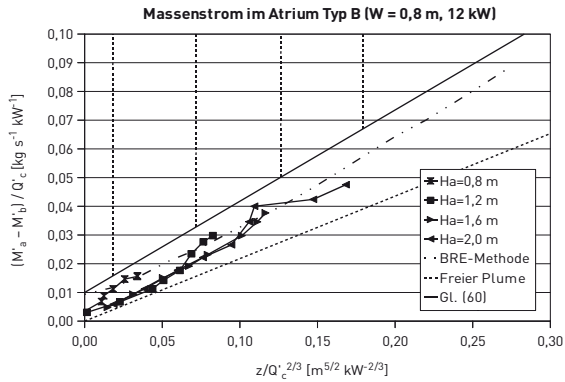


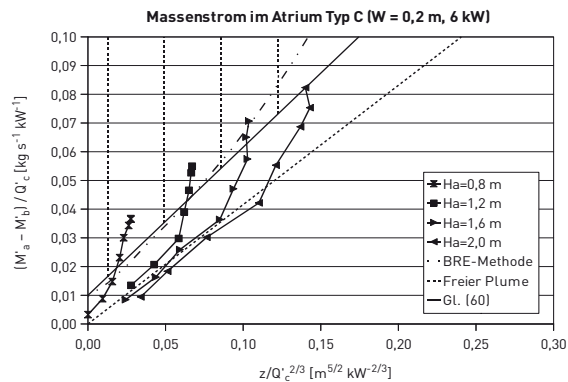
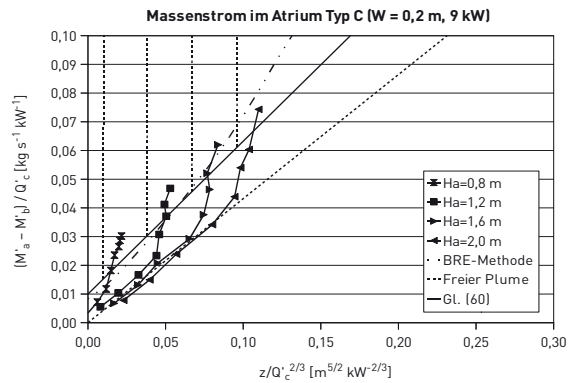
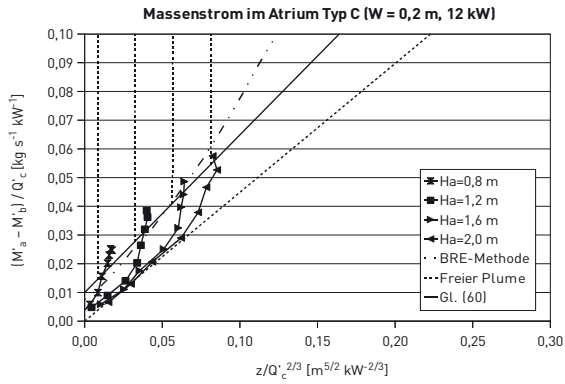


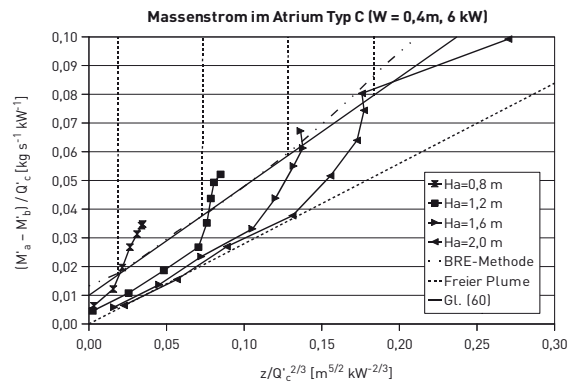
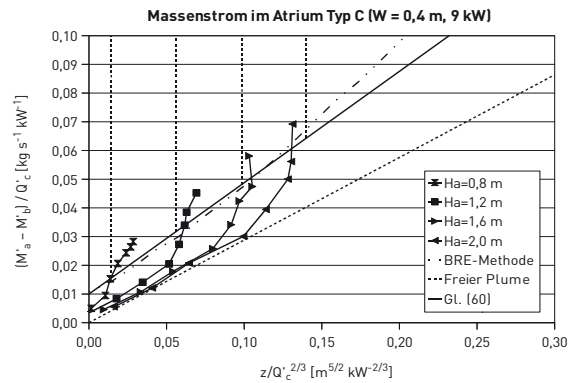
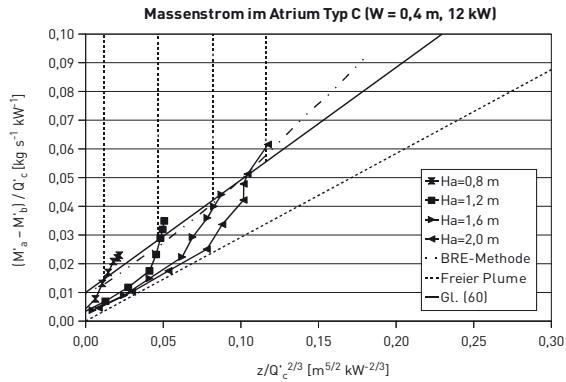


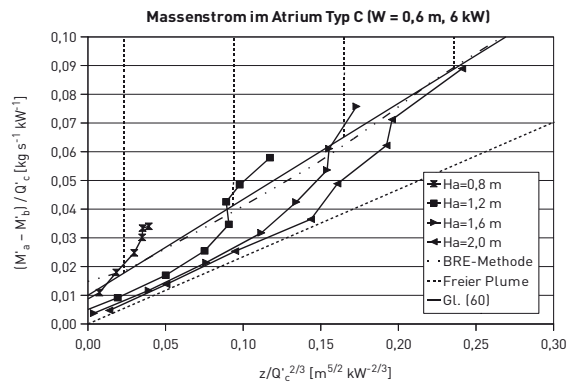
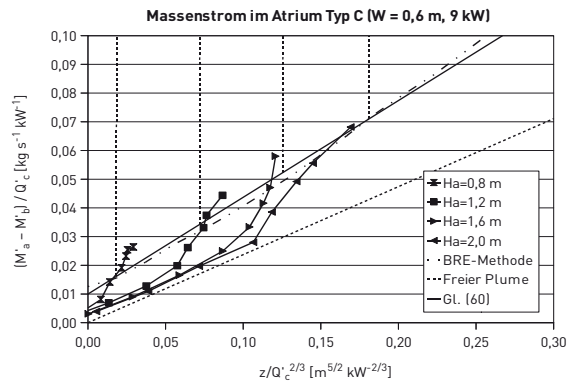
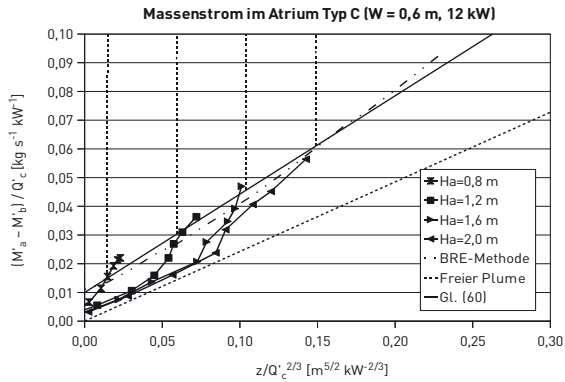


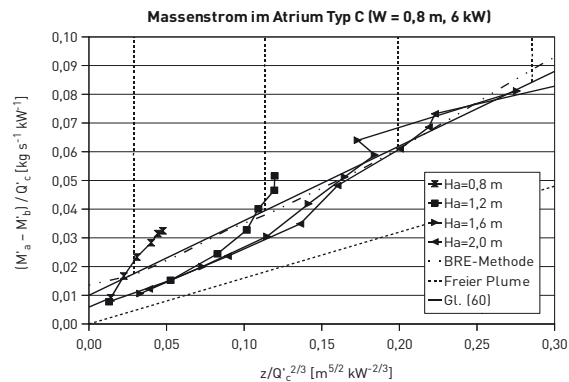
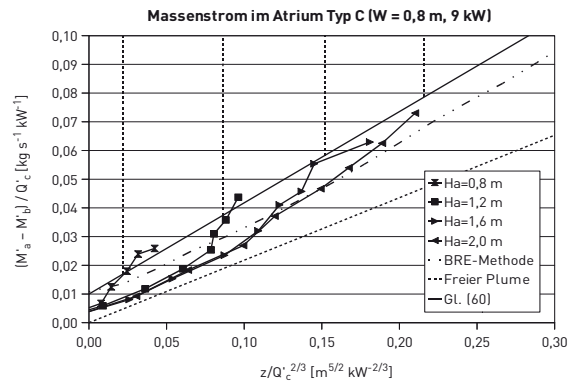
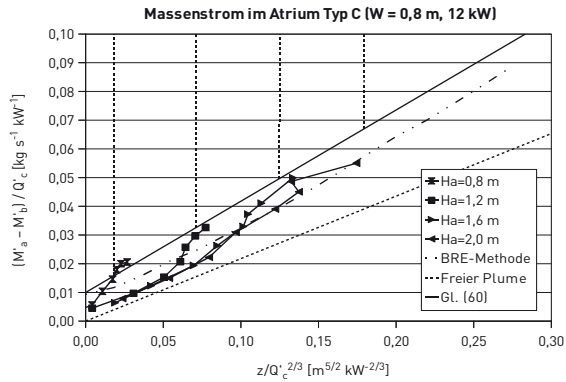


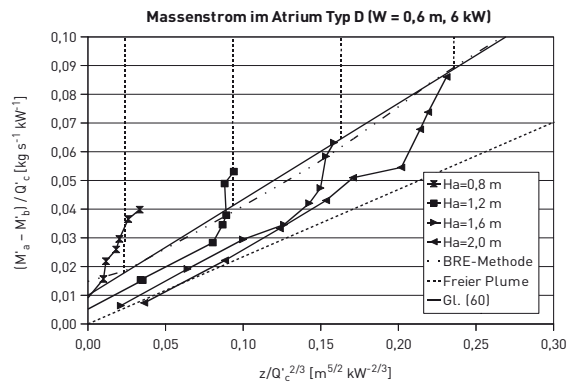
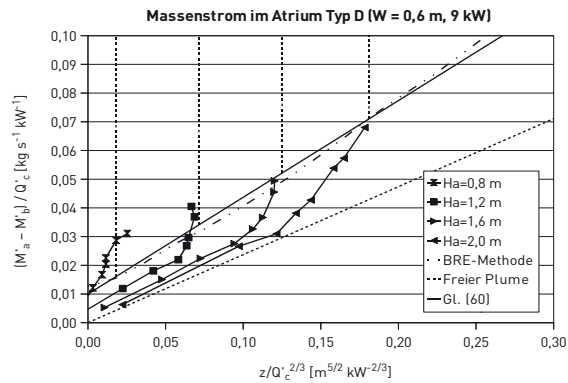
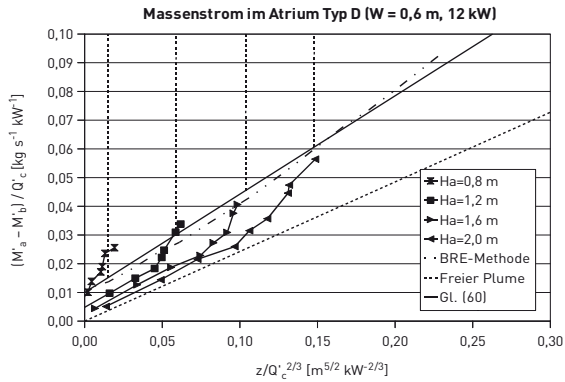












Näherungsformeln für Spill Plumes in Atrien variabler Größe

CFD-Studien unter Verwendung von GPU-Technologie

Horizontal verlaufende Rauchgasströme, die in ein Atrium eindringen und in diesem bis zur Decke aufsteigen, stellen ein bekanntes Phänomen bei Gebäudebränden dar (spill plumes). In der Planungsphase eines Gebäudes besteht deshalb häufig der Bedarf, die in diesem Fall erforderliche Entrauchung des Atriums auch ohne aufwendige CFD-Simulationen und Modellversuche beurteilen zu können. Dazu stehen dem Ingenieur eine Reihe von Methoden und Näherungsformeln zur Verfügung, mit denen die erforderliche Anzahl und Größe von natürlichen Rauchabzugsöffnungen bestimmt bzw. maschinelle Rauchabzüge dimensioniert werden können. Alle vereinfachten Methoden beschränken sich derzeit auf Atrien, die ein ausreichend großes Rauchgasreservoir aufweisen. Für kleine Atrien steht dem Ingenieur derzeit ausschließlich die BRE Spill-Plume-Methode zur Verfügung, die aufgrund ihrer Komplexität für den baupraktischen Einsatz nur bedingt geeignet ist. Im Rahmen dieser Arbeit werden Näherungsformeln entwickelt, die unabhängig von der Größe des Rauchgasreservoirs eine ausreichend genaue Abschätzung des Massenstroms im aufsteigenden Plume liefern.

Die neuen Näherungsformeln basieren im Wesentlichen auf einer detaillierten CFD-Studie. Infolge des hohen Bedarfs an Rechenleistung wird im Vorfeld untersucht, inwieweit die verwendete Simulationssoftware durch Einsatz von GPU-Prozessortechnologie beschleunigt werden kann. Dazu wird ein Großteil des Rechenkerns der Software FDS in das Programmiermodell C/CUDA portiert und parallelisiert. Darüber hinaus wird ein Multi-Wellenfrontalgorithmus für den Strahlungswärmetransport in gasförmigen Medien an die Erfordernisse der GPU-Technik angepasst und in die Software FDS integriert. Nach Abschluss der Parallelisierungsarbeit wird die modifizierte Software verifiziert und die erreichte Geschwindigkeitssteigerung ermittelt. Durch die vorgenommenen Änderungen an der Software wird eine Steigerung der Rechenleistung zwischen dem 10fachen (Algorithmen zum Strahlungswärmetransport) und dem 30fachen (Algorithmen zur Strömungsmechanik) erreicht. Unter Verwendung der neuen Software wird anschließend eine CFD-Studie erstellt, deren FE-Modelle vorab über bereits bekannte Versuche validiert werden. Im Rahmen dieser Studie werden die Modelle im Hinblick auf die Atriengeometrie, die Öffnungsbreite des Brandraumes sowie die Wärmefreisetzungsrate variiert. Die entwickelten Näherungsformeln können unabhängig von der Atriengröße für einen Großteil der baupraktisch relevanten Anwendungsfälle verwendet werden und vereinfachen damit die bisherigen Rechenverfahren.

ISBN 978-3-8167-8679-5



9 783816 786795