

# **Simulation der Stoff- und Wärmetransportvorgänge bei der Schnellpyrolyse von Lignocellulose im Doppelschneckenmischreaktor**

Zur Erlangung des akademischen Grades eines  
DOKTORS DER INGENIEURWISSENSCHAFTEN

von der KIT-Fakultät für Chemieingenieurwesen und Verfahrenstechnik des  
Karlsruher Instituts für Technologie (KIT)  
genehmigte

DISSERTATION

von  
Dipl.-Math. Robert Grandl  
aus Swaljawa (Ukraine)

Tag der mündlichen Prüfung: 05.09.2022  
Erstgutachter: Prof. Dr.-Ing. Jörg Sauer  
Zweitgutachter: Prof. Dr.-Ing. Hermann Nirschl



# Vorwort

Die Geschichte dieser Arbeit begann mit Konmayer, der 2009 mit seiner Dissertation „*Verfahrenstechnische Untersuchungen zur Schnellpyrolyse von Lignocellulose im Doppelschnecken-Mischreaktor*“ den Grundstein legte. Seine Arbeit mündete im Bau der PYTHON-Versuchsanlage. Einer Anlage Technikumsmaßstab und wichtigen Ergänzung der Stufe I der bioliq®-Pilotanlage, welche die Pyrolyse im Doppelschneckenmischreaktor nutzt um aus Lignocellulose einfach zu transportierende Pyrolyseprodukte zu gewinnen. Jedoch konnten im Rahmen dieser Dissertation nicht alle Fragen experimentell geklärt werden, so dass ein Projekt in Kooperation mit Prof. Weigang LIN vom Chinese Academy of Science/Institute of Process Engineering, mit einem Umfang von vier Doktoranden, angestrebt wurde. Diesem Projektantrag wurde nicht stattgegeben. Man entschloss sich jedoch diesen Fragen im kleinen Rahmen nachzugehen. Ich hatte die Ehre dieses Projekt ab Ende 2013 zu verfolgen.

Mit Abgabe dieser Arbeit endet eine sehr intensive Zeit. Sie war nicht immer angenehm, missen möchte ich sie dennoch nicht.

Rückblickend, muss ich gestehen, dass ohne das Mitwirken einzelner Personen diese Arbeit in der heutigen Form nicht möglich gewesen wäre. Daher möchte ich den Moment nutzen, um denen zu danken, die es verdienen.

Zunächst möchte ich mich bei meinem Professor: Jörg Sauer bedanken, der an mich glaubte und daran festhielt. Ich hoffe Sie können sich dies für kommende Generationen erhalten.

Doppelt Dank gebührt Prof. Dr. Nicolaus Dahmen für die fantastische Vorlesung Technische Chemie, die meine weitere Laufbahn prägte und für die Chance mich der Herausforderung dieser Promotion zu stellen.

In diesem Zuge möchte ich mich bei meinem Vorgesetzten: Dr. Axel Funke für seine stets offene Tür und die hervorragende Betreuung bedanken - unsere Diskussionen waren immer produktiv.

Auch möchte ich mich bei der “Bier um vier“-Runde rund um Julia Schuler, Marcus Breunig, David Steinbach, Roland Fritz und Dr. Klaus Raffelt, für den Moment der Erholung in stressigen Zeiten, bedanken.

Bei meiner Bürokollegin Sonja Habicht möchte ich mich für ihr offenes Ohr und für viel Verständnis für die Belange der Doktoranden und Herman Köhler, unserem liebevollem Paparazzo, der seine Kollegen selbst im Ruhestand nicht aus den Augen verliert, bedanken.

Danke auch an Birgit Rolli und Armin Lautenbach, die mich lange nach meinem Ausscheiden aus der Arbeitsgruppe, willkommen heißen haben.

Mein spezieller Dank gebührt unserem Technikumsleiter wider Wissen - Daniel Richter, der Mehrarbeit mit einer bewundernswerten Gleichmütigkeit durchführt. Ich hoffe ich kann mir diese Eigenschaft eines Tages zu eigen machen.

Ich danke auch meinen Studenten Gagan Sing Brar, Philipp Vetter, Larissa Ramme, Fabian Frey und Teresa Franzen für ihren Fleiß und die Mühe, mit der sie sich in die Arbeit eingebracht haben.

Jennifer Liebler danke ich für die Korrektur dieser Arbeit und die Mühe die es sie gekostet hat und meinem besten Freund - Houver Chabo der mir immer wieder zeigte: es gibt auch ein Leben außerhalb der Promotion.

Zum Schluss möchte ich mich bei meinen Elter bedanken, ohne eure Unterstützung während des Studiums und der Promotion wäre vieles schwieriger gewesen.

Nordheim, der 19. März 2021  
Robert Grandl

## Aufbau

Im Doppelschneckenmischreaktor, wie er zu Pyrolyse von Biomasse zum Einsatz kommt, wird granulare Biomasse mit granularem Wärmeträger in Kontakt gebracht. Die Biomasse reagiert beim Kontakt mit dem Wärmeträger und es entsteht ein gasförmiges Produkt. Ziel dieser Arbeit ist es den Stoff- und Wärmeübergang im Reaktor zu beschreiben. Dies kann für das granulare Edukt mittels Discrete Element Methode (DEM) geschehen. Da die bewegte Feststoffschüttung einen erheblichen Einfluss auf den Strömungsverlauf hat, ist es erforderlich neben der reinen Strömungssimulation (Computational Fluid Dynamic, CFD) auch die Schüttung zu berücksichtigen. Dies macht eine Kopplung aus CFD und DEM erforderlich.

Ein wesentlicher Teil dieser Arbeit besteht darin, die für die DEM-Simulation erforderliche Parameter für die Materialeigenschaften zu bestimmen. Im Fall der Biomasse – Weizenstroh eine herausfordernde Aufgabe. Dies geschah im Wesentlichen durch einen ständigen Abgleich von Simulationen mit Modellversuchen. Wo dies nicht möglich war, wurden fundierte Abschätzungen vorgenommen.

LIGGGHTS®, das verwendete DEM-Tool bietet die Möglichkeit den Wärmeübergang an Kontaktstellen mittels Wärmeleitung durch die Kontaktfläche zu simulieren. Es zeigte sich jedoch, dass der Wärmeübergang durch das Produktgas eine wesentliche Größe darstellt. Eine Anpassung des vorhandenen Modells erlaubte es diesen Effekt akkurater darzustellen. All dies sind erforderliche Vorarbeiten zur DEM-Simulation des Doppelschneckenmischreaktors die in einem eigenen Kapitel (Kapitel 2) zusammengefasst wurden (Abbildung 1.1).

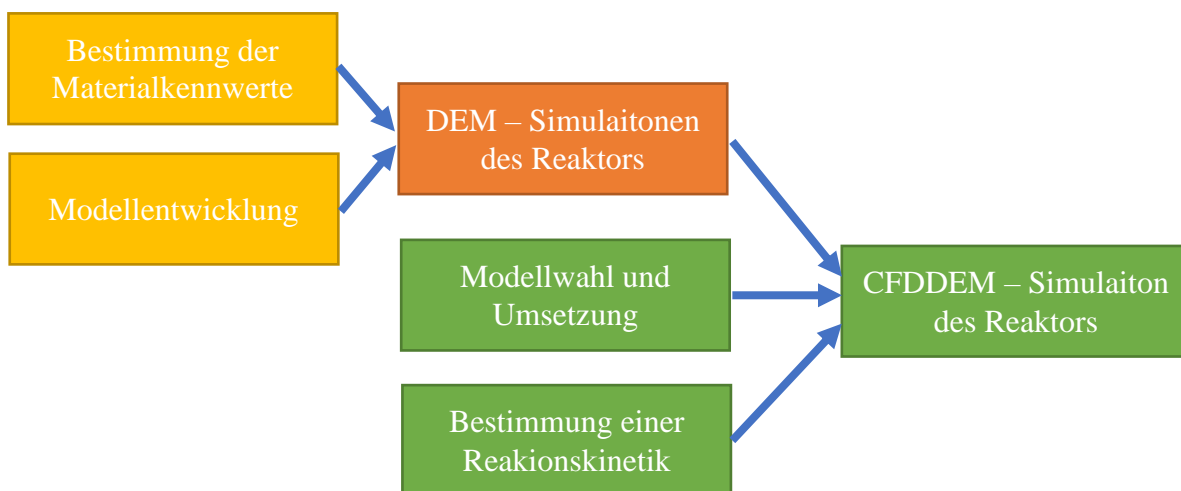


Abbildung 1.1: Aufbau der Dissertation. Gelb: Kapitel 2, Rot: Kapitel 3, Grün: Kapitel 4

Mittels DEM-Simulation wurde eine Charakterisierung des Doppelschneckenmischreaktors mit Fokus auf die Betriebsbedingungen, d.h. Rotationsgeschwindigkeit, Edukt/Wärmeträgerverhältnis und Zugabereihenfolge, durchgeführt. Der interessierte Leser findet unter Kapitel 3 das Ergebnis dieser Untersuchung.

In Kapitel 4 wurde all jenes zusammengefasst, das mit einer CFD-DEM – Simulation des Doppelschneckenmischreaktors in Verbindung steht. Dazu gehören, Informationen zum theoretischen Hintergrund, Theorie zum implementierten Bewegungsmodell (Darstellung der Schnecken), sowie eine kurze Exkursion in die Reaktionstechnik der Pyrolyse der Biomasse. Das Ergebnis dieser Mühsal sowie deren ausführliche Diskussion wurden im letzten Abschnitt dieses Kapitels untergebracht.

Die strenge Trennung zwischen DEM und CFD-DEM wurde gewählt um zu verdeutlichen, dass die DEM-Simulation an sich als eigenständiges Mittel zur Erforschung der Pyrolyse im Doppelschneckenmischreaktor Verwendung finden kann.

## Abstract

This work investigates the transport processes in the fast pyrolysis of biomass in the PYTHON – a process development unit plant at IKFT with a maximum capacity of  $20 \text{ kg h}^{-1}$  biomass. Kornmayer [1] performed first investigations utilizing this plant. In his dissertation, the pyrolysis in the twin-screw mixing reactor of the PYTHON was characterized using the black box method, i.e. the investigation of the incoming and outgoing mass and energy flows, without consideration of the internal processes. Following on from his work, this thesis considers the internal mass and energy transport processes inside the twin screw mixing reactor.

Due to the reaction conditions of  $500 \text{ }^\circ\text{C}$ , the reactivity of the products and the unknown influence of geometry on the particle behavior it is not possible to carry out complex measurements on the reactor during operation. For this reason, a numerical approach was chosen instead.

In a first step the mixing and transport behavior of the solid particles involved in the twin-screw mixing reactor were investigated, using simulations performed with the discrete element method. The simulation quality was optimized beforehand performing several investigations regarding the simulation parameters. First, the particle density, i.e. the material density including pore volume, the particle size distribution and the particle shape, were determined based on a characterization of wheat straw. Further parameter studies were performed to optimize the coefficients of friction and rolling friction. In addition, the model was adjusted to effectively approximate the heat transfer according to Schlünder [2].

Based on these preliminaries, studies were performed for the Froude number range of  $0.1 < Fr_W < 2.0$ . In this way, blockages were found in the lower rotational speed range ( $< 2 \text{ rpm}$ ). In the higher rotational speed range ( $> 3 \text{ rpm}$ ) a slight demixing of wheat straw and heat carrier could be observed. Furthermore, the residence time in this area is not sufficient to complete the heat transfer, so that a temperature difference between biomass and heat carrier particles is still observable at the particle outlet. In general, it could be observed that the higher the rotational speed, the higher the heat transfer coefficient, the lower the residence time, so that a value of  $2.5 \text{ rpm}$  represents a good compromise. All investigations of the current configuration revealed a disadvantageous back-mixing of the heat transfer medium in the biomass transport area. This effect could not be observed when the order in which the biomass/heat carrier are inserted was inverted. Previous observation regarding residence time, heat transfer and blocking could be transferred without restriction to the inverted insertion order. The maximum heat transfer coefficient is about twice as large as compared to the initial insertion order. However, at  $2 \text{ rpm}$ , it was found that the mixing effect of the reactor is too weak, so that a compact bed forms, which migrates through the reactor in cords and therefore worsens heat transfer.

Further studies regarding the biomass/heat carrier ratio confirm the assumptions, which could be expected. For example the maximum heat transfer coefficient increases as the proportion of biomass decreases. However, it is also evident that with decreasing biomass fraction, back-mixing becomes the dominant effect in heat transfer. When heat carrier moves against the flow direction, it loses thermal energy to the biomass particles, which decreases the temperature difference between biomass and heat carrier continuously.

The investigations discussed were undertaken without any consideration or knowledge of the chemical reactions or the fluid phase. In order to be able to consider both effects, a reaction model had to be implemented in to the DEM simulation and the flow simulation had to be accounted for. For this purpose, the incompressible flow treatment (cfDEM SolverPiso) was expanded to treat compressibility and a reaction model, i.e. the PaSR toolbox (**P**artially stirred reactor), had to be integrated. On behalf of the co-rotating, intermeshing screws, several methods to treat moving objects in flow simulation were tested. Only a single method - the Brinkmann-Penalty-Method, met the requirements for stability and accuracy and was integrated. In addition, a two-stage study of the chemical reaction of the pyrolysis of wheat straw was performed with the aim of determining a suitable reaction mechanism and its reaction parameters. For this purpose, methods of parameter

estimation were chosen to determine the reaction parameters based on a thermal gravimetric analysis and the molar masses of the pyrolysis products based on the residence time distribution in a semi-continuous batch pyrolysis reactor. The coefficient of determination was  $R^2 > 0.91$  in both cases.

Because of the numerical complexity, only one simulation was carried out at 2 rpm. In this process, a mass and energy loss of around 10 % could be observed when transferring data from DEM to CFD. This is due to an interaction with the Brinkmann-Penalty-Method.

Nevertheless, it could be observed that due to back-mixing heat carrier moves into the non-heated biomass transport area, which leads to a disadvantageous cooling of the heat carrier through the wall. Furthermore, a local offset of the water evaporation area and the reaction area could be determined. Therefore, the product composition can be influenced by modifying the outlet area.

While investigating the reaction behavior inside the reactor, it can be observed that the residence time is too short to completely decompose the biomass. Neither the thermally stable lignin nor the cellulose could be observed to react significantly. A sensitivity analysis of the reaction kinetics, in which reactions were considered at constant temperature and within a broad temperature range, confirms this observation.

Thermal gravimetric analysis have heating rates up to  $50 \text{ K min}^{-1}$ . In fast pyrolysis of biomass, heating rates above  $500 \text{ K s}^{-1}$  can be observed. Therefore a different reaction behavior between thermal gravimetric analysis and fast pyrolysis cannot be excluded. Nevertheless, the evaluation of the simulation indicates, that the residence time in the twin screw mixing reactor has to be increased to ensure a completed reaction of the biomass. Due to the narrow optimal operating conditions, this can only be achieved by lengthening the reactor.

## Zusammenfassung

Diese Arbeit betrachtet die Transportvorgänge bei der Pyrolyse von Biomasse in der PYTHON – einer Technikumsanlage mit einer maximalen Kapazität von  $20 \text{ kg h}^{-1}$  Biomasse. Erste Untersuchungen zu dieser Thematik wurden von Kornmayer [1] durchgeführt. Im Zuge seiner Dissertation erfolgte eine Charakterisierung der Pyrolyse im Doppelschneckenmischreaktor der PYTHON-Versuchsanlage anhand eines Blackbox-Verfahrens, d.h. der Betrachtung der ein- und ausgehenden Massen- und Energieströme, ohne nähere Betrachtung der inneren Vorgänge. Daran anknüpfend betrachtet diese Arbeit die Stoff- und Energietransportvorgänge im Doppelschneckenmischreaktor.

Aufgrund der Reaktionsbedingungen von  $500 \text{ }^\circ\text{C}$ , der Reaktivität der Produkte und der unbekannte Einfluss der Geometrie auf das Partikelverhalten, ist es nicht möglich komplexere Messungen am Reaktor im Betrieb durchzuführen. Aus diesem Grund wurde ein numerischer Ansatz gewählt.

In einem ersten Schritt wurde das Misch- und Transportverhalten der beteiligten Feststoffpartikel im Doppelschneckenmischreaktor mittels Diskrete-Elemente-Methode untersucht. Im Vorfeld wurden mehrere Schritte zur Optimierung der Simulationsqualität unternommen. Zunächst wurde im Rahmen einer Charakterisierung von Weizenstroh die Partikeldichte, d.h. die Materialdichte inklusive Porenvolumen, die Partikelgrößenverteilung und die Partikelform ermittelt. Weitere Parameterstudien hatten die Optimierung der Reib- und Rollreibungswerte zum Ziel. Darüber hinaus wurden Modelle angepasst, um den Wärmeübergang nach Schlünder [2] effektiv approximieren zu können.

Unter diesen Vorbetrachtungen wurden Studien für einen Werkzeug-Froude-Bereich von  $0.1 < Fr_w < 2.0$  durchgeführt. Dabei konnten Verblockungen im unteren Drehzahlbereich ( $< 2 \text{ rpm}$ ) beobachtet werden. In höheren Drehzahlbereich ( $> 3 \text{ rpm}$ ) konnte eine leichte Entmischung beobachtet werden. Des Weiteren ist die Verweilzeit in diesem Bereich nicht ausreichend, um den Wärmeübergang abzuschließen, sodass ein Wärmeübergang selbst am Reaktorende erkennbar war. Generell konnte beobachtet werden: je höher die Drehzahl, desto höher der Wärmeübergangskoeffizient und desto niedriger die Verweilzeit, sodass ein Wert von  $2.5 \text{ rpm}$  einen guten Kompromiss darstellt. Bei allen Untersuchungen des aktuellen Aufbaus konnte eine nachteilige Rückvermischung des Wärmeträgers in den Biomasse-Transportbereich festgestellt werden. Ein Effekt, der bei vertauschter Biomasse-/Wärmeträgerzugabereihenfolge, d.h. Biomasse zu Wärmeträger, nicht beobachtet werden konnte. Vorhergehende Erkenntnisse in Bezug auf Verweilzeit, Wärmeübergang und Verblockung konnten ohne Einschränkung auf die vertauschte Zugabereihenfolge übertragen werden. Jedoch zeigte sich bei  $2 \text{ rpm}$ , dass der Mischeffekt des Reaktors zu schwach ausfällt, sodass sich ein kompaktes Paket bildet, das in Schlieren durch den Reaktor wandert und den Wärmeübergang auf diese Weise hemmt.

Weitere Studien bzgl. des Biomasse-/Wärmeträgerverhältnisses bestätigen den zu erwartenden Sachverhalt. D.h. mit sinkendem Biomasseanteil steigt der maximale Wärmeübergangskoeffizient. Jedoch wird auch ersichtlich, dass mit sinkendem Biomasseanteil die Rückvermischung zum dominanten Effekt beim Wärmeübergang wird. D.h. Wärmeträger dringen bis in den Biomassezugabebereich und geben auf dem Weg dorthin ihre thermische Energie an die Biomasse ab, wodurch die Temperaturdifferenz zwischen Biomasse und Wärmeträger sinkt.

Die bisherigen Untersuchungen wurden unter Vernachlässigung der Reaktionstechnik oder der fluiden Phase unternommen. Um beide Effekte betrachten zu können, musste in die DEM-Simulation ein Reaktionsmodell implementiert und dies um die Strömungssimulation erweitert werden. Dazu wurde die inkompressible Strömungsbehandlung (cfdemSolverPiso) um die Kompressibilität erweitert und mit einem PaSR-Reaktionsmodell (**P**artially stirred reactor) kombiniert. Darüber hinaus wurden mehrere Methoden zur Darstellung der co-rotierenden, ineinander kämmenden Schnecken getestet. Nur ein einziges Verfahren - das Brinkmann-Strafverfahren, erfüllte die Anforderungen an Stabilität und Genauigkeit und wurde implementiert. Darüber hinaus wurde eine zweistufige Studie bzgl. der Reaktionstechnik bei der Pyrolyse von Weizenstroh durchgeführt, mit

dem Ziel ein geeignetes Reaktionsnetzwerk und die dazugehörigen Parameter zu bestimmen. Dazu wurden Methoden der Parameterschätzung gewählt, um die Reaktionsparameter anhand einer TG-Analyse und die molaren Massen der Pyrolyseprodukte anhand der Verweilzeitverteilung in einem semikontinuierlichen Bach-Pyrolysereaktor, zu bestimmen. Die Modellgüte war in beiden Fällen  $R^2 > 0.91$ .

Aufgrund des numerischen Aufwandes wurde nur eine Simulation bei  $2 \text{ rpm}$  durchgeführt. Dabei konnte ein Massen- und Energieverlust von rund 10 % beim Übertrag von DEM auf CFD beobachtet werden konnte. Dies ist auf eine Interaktion mit dem Brinkmann-Strafverfahren zurück zu führen. Dennoch konnte beobachtet werden, dass die Rückvermischung bis in den nicht-begleitbeheizten Biomasse-Transportbereich hineinragt und dort zu einer nachteiligen Abkühlung der Wärmeträger führt.

Des Weiteren konnte ein lokaler Versatz des Wasser-Verdampfungsbereichs und des Reaktionsbereichs festgestellt werden, sodass durch eine Modifikation des Auslassbereichs Einfluss auf die Produktqualität genommen werden kann.

Eine Betrachtung des Reaktionsverlaufes im Reaktor zeigt, dass die Verweilzeit nicht ausreicht, um Biomasse vollständig zu zersetzen. Weder beim thermisch stabilen Lignin, noch bei der Cellulose konnte eine wesentliche Zersetzung beobachtet werden. Eine Sensitivitätsanalyse der Reaktionskinetik, bei der Reaktionen mit konstanten Temperaturen und in einem breiten Temperaturbereich betrachtet wurden, bestätigt diese Aussage.

TG-Analysen weisen Heizraten von bis zu  $50 \text{ K min}^{-1}$  auf. Bei der Schnellpyrolyse von Biomasse können Heizraten von über  $500 \text{ K s}^{-1}$  beobachtet werden, sodass ein abweichendes Reaktionsverhalten durch den Übertrag von TGA auf die Schnellpyrolyse nicht auszuschließen ist. Jedoch weisen die aktuellen Daten darauf hin, dass eine Verlängerung der Verweilzeit im Reaktor erforderlich ist. Diese kann aufgrund der engen optimalen Betriebsbedingungen nur durch eine Verlängerung des Reaktors erzielt werden.





# Inhaltsverzeichnis

1	Einleitung	1
1.1	Motivation	1
1.2	Pyrolyse von Sekundärbiomasse	2
1.3	Schnelle Pyrolyse im Doppelschneckenmischreaktor	6
1.4	Aufgabenstellung	11
2	Discrete Element Methode	12
2.1	Grundlagen der DEM-Simulation	12
2.1.1	Integration	15
2.1.2	Nachbarsuche	16
2.2	Parameterbestimmung bei der DEM-Simulation	16
2.2.1	Bestimmung Materialkennwerte, Kennwerte für die Wärmeträger/Wärmeträger und Wärmeträger/Wand Interaktionen	17
2.3	Charakterisierung von Weizenstroh	33
2.3.1	Partikeleigenschaften	33
2.3.2	Untersuchung der Weizenstroh-Partikelform	33
2.3.3	DEM-Partikelmodell	39
2.3.4	Einfluss der DEM-Partikelform auf das Mischverhalten	41
2.4	Bestimmung der Wärmeträger/Biomasse Kennwerte	46
2.5	Einfluss der Biomasse/Biomasse Reibung und Rollreibung	49
2.6	Einfluss des Elastizitätsmoduls auf das Mischverhalten	50
2.7	DEM: Wärmeleitung und -übergang im Doppelschneckenmischreaktor	52
3	Mischverhalten im Doppelschneckenmischreaktor	65
3.1	Wärmeübergang des Doppelschneckenmischreaktors in Abhängigkeit der Rotationsgeschwindigkeit	65
3.2	Einfluss des Partikelverhältnisses auf den Wärmeübergang	69
3.3	Einfluss der Zugabereihenfolge	72
3.4	Diskussion	74
4	Computed Fluid Dynamics – Discrete Element Methods (CFD-DEM)	76
4.1	Grundlagen der Strömungssimulation beweglicher Objekte	76
4.1.1	Brinkman Strafverfahren in kompressiblen Strömungen	79
4.2	Grundlagen der CFDEM-Simulation	83
4.2.1	Wärmeübergang bei der CFDEM®-Simulation	84
4.3	Grundlagen der Simulation reaktiver Strömungen	85
4.4	Kinetik bei der Pyrolyse von Lignocellulose	87
4.4.1	Reaktionsmechanismus bei der Pyrolyse	87
4.4.2	Reaktionsparameterbestimmung bei der Pyrolyse von Weizenstroh	88

4.4.3	Ermittlung der molaren Massen bei der Schnellpyrolyse	90
4.5	Solver foamPy	97
4.5.1	DEM – Implementation	97
4.5.2	CFD – Implementation	99
4.5.3	Rahmenbedingung zur Simulation des Doppelschneckenmischreaktors	102
4.5.4	Stoff- und Energietransport im Doppelschneckenmischreaktor	106
4.6	Diskussion	113
5	Abschließender Rück- und Ausblick	114
	Appendix	118
	Gleichungs- und Symbolverzeichnis	216
	Tabellenverzeichnis	223
	Abbildungsverzeichnis	225
	Literaturverzeichnis	228

# 1 Einleitung

## 1.1 Motivation

Bei der Erforschung der Pyrolyse entwickeln sich Simulationen als potentes Mittel zur Charakterisierung der Reaktoren. Dabei werden unabhängig voneinander der Euler-Euler oder der Euler-Lagrange-Ansatz verfolgt. Der Euler-Euler-Ansatz stellt Feststoff, Flüssigkeit und Gas als Phasen in einem Kontinuum dar, die in einander übergehen können. Wechselwirkungen wie der Energieverlust durch Kollision oder Reibung werden mit speziellen statistischen Modellen berücksichtigt. Ein Musterbeispiel für den klassischen Euler-Ansatz ist die Strömungssimulation (Computational Fluid Dynamics) [3]. Dem gegenüber steht der Euler-Lagrange-Ansatz. Bei diesem wird die Gasphase durch den Euler-Ansatz, d.h. der Strömungssimulation, dargestellt und die Feststoffpartikel durch den Lagrange-Ansatz. Beim Lagrange-Ansatz werden die Partikel einzeln unter Verwendung Newtons zweitem Gesetz, d.h. „Kraft ist Gleich Masse mal Beschleunigung“, dargestellt. Die verwendeten Methoden reichen von Diskrete Element (DEM) [4] bis zu dessen Vorgänger den Diskrete Partikel (DPM) [5]. Im Vergleich zum Euler-Euler-Ansatz ist der Euler-Lagrange-Ansatz rechenintensiver, beschreibt jedoch insbesondere das Partikelverhalten genauer [6]. Der Fokus dieser Arbeit liegt daher auf dem Euler-Lagrange-Ansatz.

Zu den frühen Arbeiten in diesem Bereich gehörte Zhou et. al. 2003 [7] der den Wärmeübergang bei der Kohlevergasung in einer stationären Wirbelschicht untersuchte. Dies geschah aufgrund der erforderlichen Rechenleistung in einer 2D-Simulation mit rund 1480 Partikel. Dabei konnte Zhou zeigen, dass der Wärmeübergang durch Konvektion eine untergeordnete Rolle spielt.

Erst eine Weiterentwicklung der verwendeten Algorithmen erlaubte eine Betrachtung mit höheren Partikelzahlen. So gelang es Bruchmüller et. al. [8] 2012 den Stoff- und Energietransportes bei der Pyrolyse von Biomasse im Wirbelschichtreaktor zu untersuchen und die wichtige Einflussgröße: Strömungsgeschwindigkeit zu identifizieren. Darüber hinaus gelang es Bruchmüller eine Optimale Pyrolysetemperatur zur Verflüssigung von 750 K zu ermitteln. Dies wird u.a. von Scott et. al. [9] bestätigt.

Seit dem wurden Zahlreiche arbeiten die unter den Euler-Lagrange-Ansatz fallen zur Pyrolyse von Biomasse in Wirbelschichtreaktoren durchgeführt. Diese unterscheiden sich im Wesentlichen durch die verwendeten Modelle und durch die Reaktionskinetik. Jedoch sind zum Thema Pyrolyse von Lignocellulose im (Doppel-)Schneckenmischreaktors nur wenige arbeiten zu finden [6].

Ausnahmen sind Aramideh et. al. [10] der die Pyrolyse von Lignocellulose im Schneckenmischreaktor mittels Euler-Euler-Ansatz charakterisierte und Qi et. al. [11] der den Wärmeübergang bei der Pyrolyse von Biomasse im Doppelschneckenmischreaktor mittels reinem Lagrange-Ansatz betrachtete.

Bis dato sind dem Autor keine Untersuchungen zur Pyrolyse von Biomasse im Doppelschneckenmischreaktor mittels Euler-Euler oder Euler-Lagrange-Ansatz bekannt.

Dies ist zum einem auf die größere Präsenz von Wirbelschichtreaktoren, aber auch auf die Komplexität, ineinander kämmende, rotierende Körper mit engem Wandabstand in der Strömungssimulation zurückzuführen. Mehr dazu in Kapitel 4.1.

## 1.2 Pyrolyse von Sekundärbiomasse

Betrachten wir das Holzfeuer, so spaltet die Pyrolyse bei hohen Temperaturen die langkettigen organischen Verbindungen in Gas, Flüssigkeiten und Koks. Die Produkte verbrennen in einer Flammenreaktion und liefern die Energie, die bei der Pyrolyse erforderlich ist. So gesehen, ist die Pyrolyse die wichtigste und älteste thermochemische Reaktion, die uns seit der Steinzeit bekannt ist. Bewusst wurde die Pyrolyse seit dem Altertum zur Produktion von Holzkohle in Meilern verwendet [12]. Dabei wurde Holz auf einem Haufen geschichtet, abgedeckt und entzündet. Die Abdeckung sorgte für einen Luftmangel, der verhindert dass das Holz vollständig abbrennt. Die für die Pyrolyse notwendige Hitze wurde durch diese gehämmte Verbrennung erzeugt. Gesteuert wurde der Prozess über das Öffnen und Schließen von Löchern in der Abdeckung. Das Produkt – die Holzkohle – wies eine höhere Energiedichte als Holz auf und konnte aufgrund der höheren Temperaturen und der fehlenden Kondensate zur Eisenreduktion genutzt werden [13]. D.h. der gleiche Prozess, die schnelle Pyrolyse beim Holzfeuer und die langsame Pyrolyse bei der Holzkohleproduktion, brachten uns aus der Steinzeit in die Eisenzeit.

Neben der schnellen und langsamen Pyrolyse gibt es die mittelschnelle Pyrolyse. Diese stellt, wie auch der Name sagt, einen Mittelweg zwischen den beiden Pyrolysearten dar (Tabelle 1.1).

*Tabelle 1.1: Unterteilung der Pyrolyse nach Kaltschmitt et. al. [14]*

<b>Pyrolyse</b>	<b>Heizrate</b>	<b>Verweilzeit</b>
Schnelle	1000 K/s und mehr	< 4 s
Mittelschnelle	< 1000 K/s	10 – 30 s
Langsame	< 10 K/s	h – d

Die in Tabelle 1.1 genannten Definitionen sind nicht immer eindeutig. Viele Autoren unterscheiden die Pyrolyse in schnelle, ultraschnelle und langsame Pyrolyse, sodass derselbe Begriff – Schnellpyrolyse - für die mittlere nach Tabelle 1.1 und für ultraschnelle oder auch Flashpyrolyse stehen kann. Neben Gas und Koks entstehen bei der Pyrolyse auch flüssige Bestandteile. Diese wurden bei der Verkohlung aufgefangen und fanden als Klebe- und Dichtmittel Verwendung [15].

Technisch gewinnt die Pyrolyse im Zusammenhang mit dem Thema Recycling immer mehr an Bedeutung, stellt es eine Möglichkeit dar, Elastomere und Duroplaste stofflich zu recyceln [16]. Diese Fähigkeit, stabile organische Verbindungen zu spalten und in eine Form hoher Energiedichte zu bringen, wird auch beim BTL-Verfahren (Biomass to liquide) des bioliq®-Projektes genutzt [17].

Ziel ist es trockene Biomasse nahe der Bezugsquelle in eine leicht zu transportierende Form zu bringen, die in weiteren Prozessschritten zu Treibstoff und Plattformchemikalien umgewandelt werden kann. Technisch wird hierbei die schnelle Pyrolyse bevorzugt. Dies hat mehrere Vorteile. Zunächst ist bei gleicher Größe ein höherer Durchsatz möglich. Darüber hinaus werden bei der schnellen Pyrolyse vermehrt flüssige Produkte gebildet.

Das bioliq®-Verfahren ist ausgelegt auf die Verwendung sog. sekundärer Biomasse – pflanzliche Reste, die nicht in Konkurrenz zur Nahrungsmittelproduktion stehen. Dabei handelt es sich durchweg um sog. Lignocellulose, d.h. Biomasse die neben Asche und Restfeuchte, aus Cellulose (Abbildung 1.1), Hemicellulose (Abbildung 1.2) und Lignin (Abbildung 1.3) besteht.

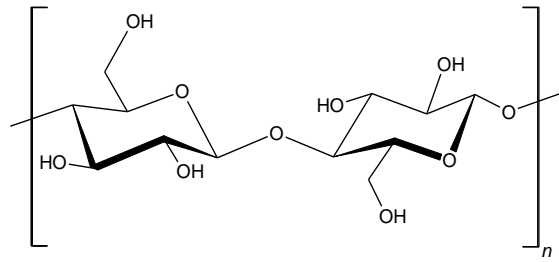


Abbildung 1.1: Strukturformel von Cellulose nach Suhas et. al. [18]

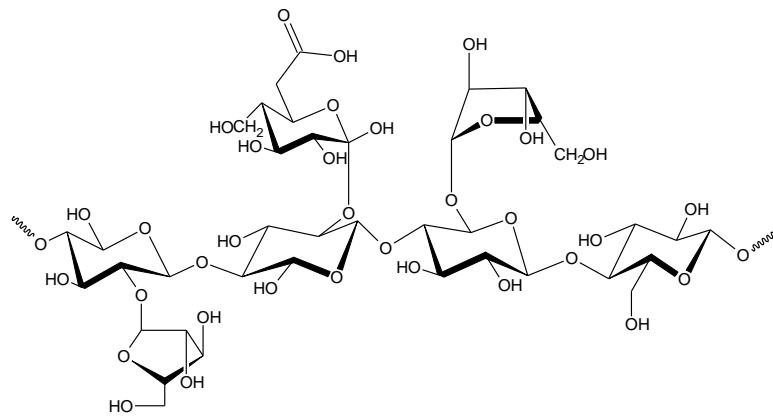
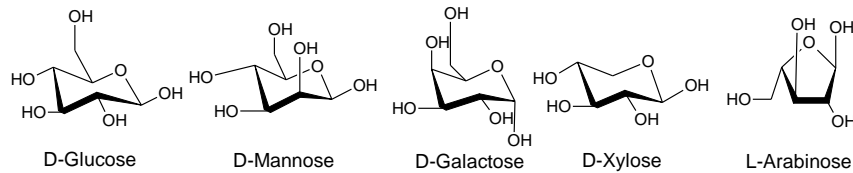


Abbildung 1.2: Mögliche Hemicellulosestruktur nach Zheng et. al. [19]

Cellulose ist ein Polymer des Monomers Cellobiose. Es ist der Hauptbestandteil pflanzlicher Zellwände. Durch deren parakristalline Regionen sorgt Cellulose in Fibrillen für die Zugfestigkeit [20]. Bei dem Polymer Lignin handelt es sich um ein Oligomer, das im Wesentlichen aus Cumaryl-, Coniferyl- und Synapylalkohol besteht. In der Pflanze ist Lignin für die Druckfestigkeit zuständig. D.h. in Holz ist Lignin in größeren Mengen an zu treffen als in Gräsern. Aufgrund der starken Verzweigung ist Lignin thermisch am stabilsten [21, 22].

Hemicellulose ist ein Sammelbegriff für die Polysaccharide in pflanzlichen Zellen. Die häufigsten Monomere sind Glucose, Mannose, Galactose, Xylose und Arabinose. Die amorphe Hemicellulose ist zusammen mit Cellulose und Lignin Stütz- und Gerüstsubstanz bei pflanzlichen Zellwänden [23].

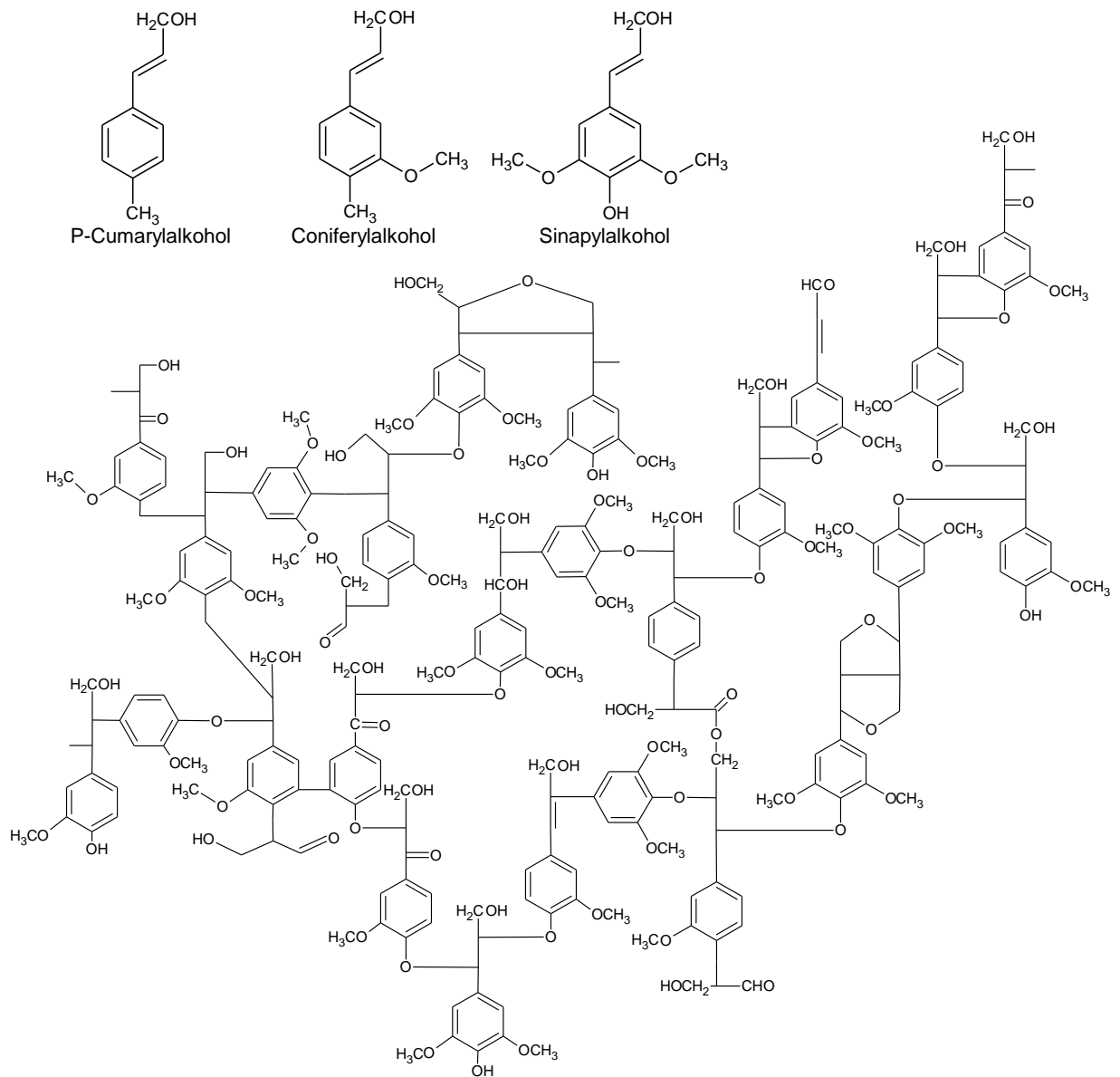


Abbildung 1.3: Mögliche Strukturformel von Lignin nach Watkins et. al. [12]

Aufgrund ihrer Zusammensetzung sind bei der Pyrolyse von die Biomasse mit steigender Temperatur mehrere Reaktionsbereiche zu beobachten. Bei 100 °C verdampft die restliche Feuchtigkeit, zwischen 220 °C - 315 °C findet die Zersetzung von Hemicellulose, zwischen 315 °C - 400 °C die Zersetzung von Cellulose und zwischen 160 °C - 900 °C die Zersetzung von Lignin statt [24]. Nicht nur die Primärreaktionen – die Zersetzung der Biomasse, sondern auch die Sekundärreaktionen – deren Folgereaktionen, weisen eine Temperaturabhängigkeit auf [25]. Bei höheren Temperaturen werden die Moleküle stärker gespalten, wodurch die Gasphase maximiert wird. Aus dem gleichen Grund ist der Feststoffgehalt bei niedrigeren Temperaturen am höchsten. Die Kondensatausbeute nimmt bei ca. 500 °C ihr Maximum an (Abbildung 1.4) [14].

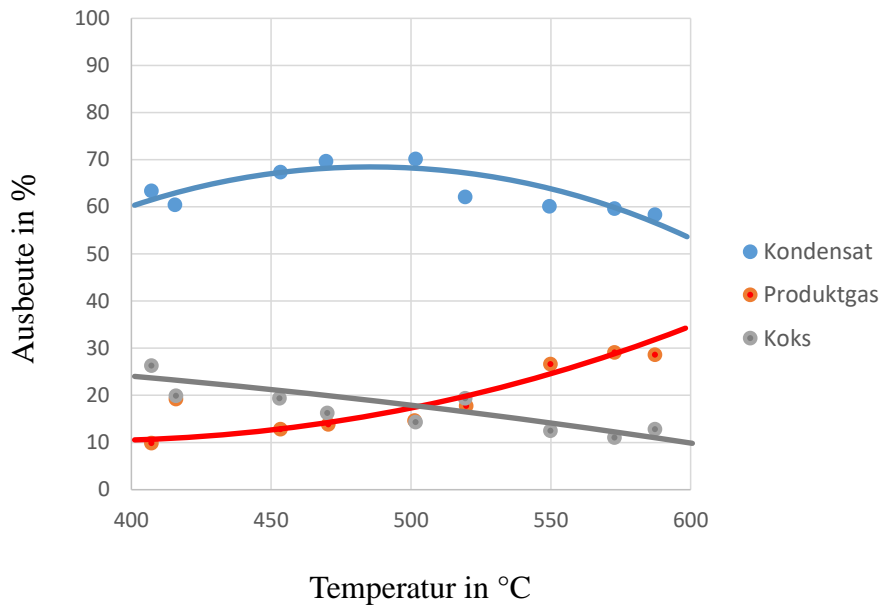


Abbildung 1.4: Produktausbeute in Abhängigkeit der Temperatur nach Bridgewater et. al.[26]

Unabhängig dem Reaktor müssen bei der Schnellpyrolyse die gleichen Prozessschritte durchlaufen werden (Abbildung 1.5). Um den Wärmeübergang bei der Pyrolyse zu maximieren, wird die Biomasse getrocknet und zerkleinert. Zur Maximierung der Produktausbeute kann die Biomasse bevor sie in den Reaktor gegeben wird inertisiert werden. Abhängig von Reaktortyp ist eine Feststoffabtrennung, z.B. mittels Zyklon, nach der Pyrolyse erforderlich. Das 500 °C heiße Produktgas muss nun, z.B. in einer Quenche, auf Raumtemperatur abgekühlt werden. Die verbliebenen Aerosole können mit Filtern aus der restlichen Gasphase abgetrennt werden.

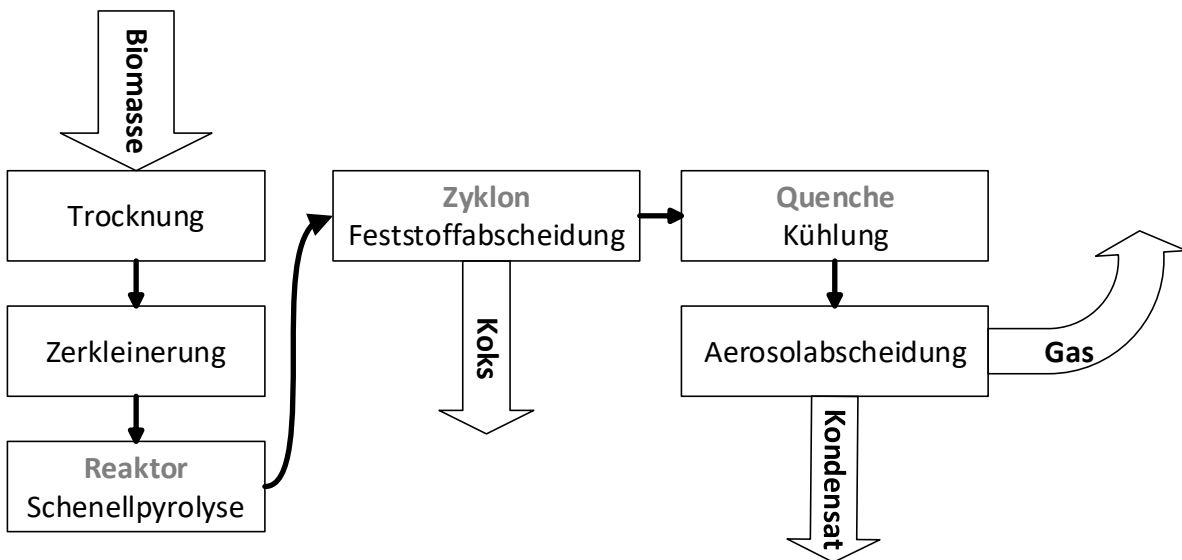


Abbildung 1.5: Allgemeine Prozessschritte bei der Schnellpyrolyse von Biomasse

Obwohl die Pyrolyse eine der ältesten chemischen Prozesse ist die von Menschen durchgeführt wird, stößt das Wissen um diesen Prozess an seine Grenzen. Große Fragen sind im Bereich der Reaktionskinetik weiterhin offen. Die Gründe hierfür sind zahlreich. Variierende



Biomassezusammensetzung, variierende Struktur des Lignin und der Hemicellulose, über 300 Produkte [27], das katalytische Verhalten der Asche [28] und die Kondensation des Pyrolysegases bei geringer Temperaturdifferenz sind nur einige Gründe, die eine Charakterisierung der Reaktionen erschweren. Ein weiterer ist die Betriebstemperatur von 500 °C, bei der die meisten Messinstrumente nicht eingesetzt werden können. Auf diese Thematik wird in Kapitel 4.4 ausführlich eingegangen.

### 1.3 Schnelle Pyrolyse im Doppelschneckenmischreaktor

Bei der Schnellpyrolyse wird das Edukt – die Biomasse, schnellstmöglich auf Reaktionstemperatur gebracht. Zur Vermeidung von Sekundärreaktionen werden die durch Pyrolyse entstandenen Produkte – Pyrolysegas und Koks – möglichst schnell getrennt und abgekühlt [26].

Im Laufe der Jahre wurden viele Systeme entwickelt, die versprechen, diese möglichst effizient zu bewerkstelligen (Tabelle 1.2). Diese Systeme/Reaktoren arbeiten nach Teils unterschiedlichen Prinzipien, die ihre eigenen Vor- und Nachteile mitbringen.

Eine Besonderheit in Tabelle 1.2 ist der **Mikrowellenreaktor**. Durch Mikrowellen werden Moleküle mit Dipolmoment zu einer Rotation angeregt. Die Wärme entsteht durch dielektrischen Verlust. Beim Mikrowellenreaktor wird das Wasser bis zur Verdampfung angeregt und der Dampf überhitzt. Dabei tritt der Dampf mit hoher Geschwindigkeit aus den Poren der Biomasse. Die dadurch verursachten Wirbel reißen die laminare Grenzschicht ab, wodurch der Wärmeübergang von Dampf auf Partikel begünstigt wird. Die exakte Temperaturregelung stellt dabei eine Herausforderung dar. Ein weiteres Problem ist die Durchdringungstiefe. Diese beträgt i.a. 1-2 cm, was das Upscaling erschwert [29]. Der Mikrowellenpyrolysereaktor stellt insbesondere im Labor eine interessante Möglichkeit dar. Die Forschung an diesen Reaktoren wird u.a. an der TU Wien/Österreich, U. Shandong/China und U. Minnesota/USA betrieben.

Beim **Ablativreaktor** (Abbildung 1.6) wird Biomasse gegen eine heiße rotierende Oberfläche gepresst. Dabei wird die Wärme von der Platte zur Biomasse übertragen. Durch die Rotation der Scheibe wird kontinuierlich Koks und Asche abgetragen, das Pyrolysegas wird abgeleitet. Eine Herausforderung bei der Verwendung des Ablativreaktors zur Pyrolyse von Biomasse ist eine konstante Temperatur auf der Scheibe zu erzielen. Industriell wurde der Ablativreaktor z.B. bei der Firma PyTec GmbH in Cuxhafen mit einer Kapazität von 250 kg/h eingesetzt [30, 31].

Den gängigsten Reaktor in der Liste stellt der stationäre **Wirbelschichtreaktor** dar. Dieser zeichnet sich durch seine simple Technik aus und trumps mit einem hohen Wärmeübergang, mit hoher Heizrate, guter Durchmischung und kurzen Gasverweilzeiten. Zur Reduzierung des Inertgasverbrauchs, wird das nach der Kondensation (bei 20 °C) anfallendem Gas wieder in den Reaktor geleitet. Dieses muss erneut auf Reaktionstemperatur von 500 °C gebracht werden und schlägt sich daher in der Wirtschaftlichkeit nieder [32]. Aufgrund der moderaten Gasgeschwindigkeit müssen Kokspartikel erst mechanisch beansprucht werden bevor sie klein genug sind, um mit dem Gas ausgetragen zu werden. Dies führt zu einer Akkumulation des Kokses und damit zur autokatalytischen Asche im Reaktor. In nennenswerten Umfang wird die Wirbelschichtanlage BioTherm™ bei der Firma Dynamotive/Kanada mit einem Durchsatz von 200 t pro Tag betrieben [33].

*Tabelle 1.2: Ausgewählte Reaktortypen zur Schnellpyrolyse von Biomasse  
nach Bridgwater et. al. [20, 23]*

	<b>Reaktortyp</b>	<b>Wesentliche Wärmeübertragung</b>	<b>Vorteile</b>	<b>Nachteile</b>
	Mikrowellenreaktor	Umwandlung elektromagnetischer Feldenergie	+ sehr hohe Heizrate	- Temperatur schwierig zu kontrollieren - max. Eindringtiefe 2 cm - schlechtes Upscaling
Direktkontakt	Ablativreaktor	Konduktion	+ auch große Partikel möglich + kompaktes Design + kein Inertgas erforderlich + kompaktes Design möglich	- mechanisch bewegte, beheizte Teile - keine konstante Wärmequelle
Gasfluidisiert	Stationäre Wirbelschicht	Konduktion, Konvektion	+ einfacher Aufbau und einfaches Upscaling + gute Durchmischung + hohe Heizrate möglich + gute Temperaturverteilung + kurze Produktverweilzeit	- nur kleine Partikel (< 2mm) - große Mengen Inertgas erforderlich - Energetisch ineffizient - Akkumulation von Koks (autokatalytisch) im Reaktor
	Zirkulierende Wirbelschicht	Konduktion, Konvektion	+ einfaches Upscaling + guter Wärmeübergang + gute Temperaturverteilung + gute Kondensatausbeute + große Partikel möglich (< 6 mm)	- kein Koks - starker Abrieb von Wärmeträgern - Akkumulation von autokatalytischer Asche im Kreislauf
	Flugstrom	Konduktion	+ einfache Technik + kurze Gasverweilzeit + einfaches Upscaling	- schlechter Wärmeübergang - zu kurze Feststoffverweilzeit (< 1 s) - Partikelgröße (< 1 mm) - Energetisch ineffizient - große Mengen Inertgas
Mechanische Fluidisierung	Drehkegelreaktor	Konduktion, Konvektion	+ gute Durchmischung + kurze Produktverweilzeit + gute Kondensatausbeute	- höherer Regelaufwand um die Teilprozesse abzustimmen - problematisches Upscaling
	Schneckenreaktor	Konduktion, Konvektion	+ kein Inertgas + robust bzgl. Partikelgröße + Handhabung + einfaches Upscaling	- höhere Anschaffungskosten - längere Verweilzeit (5-10s)

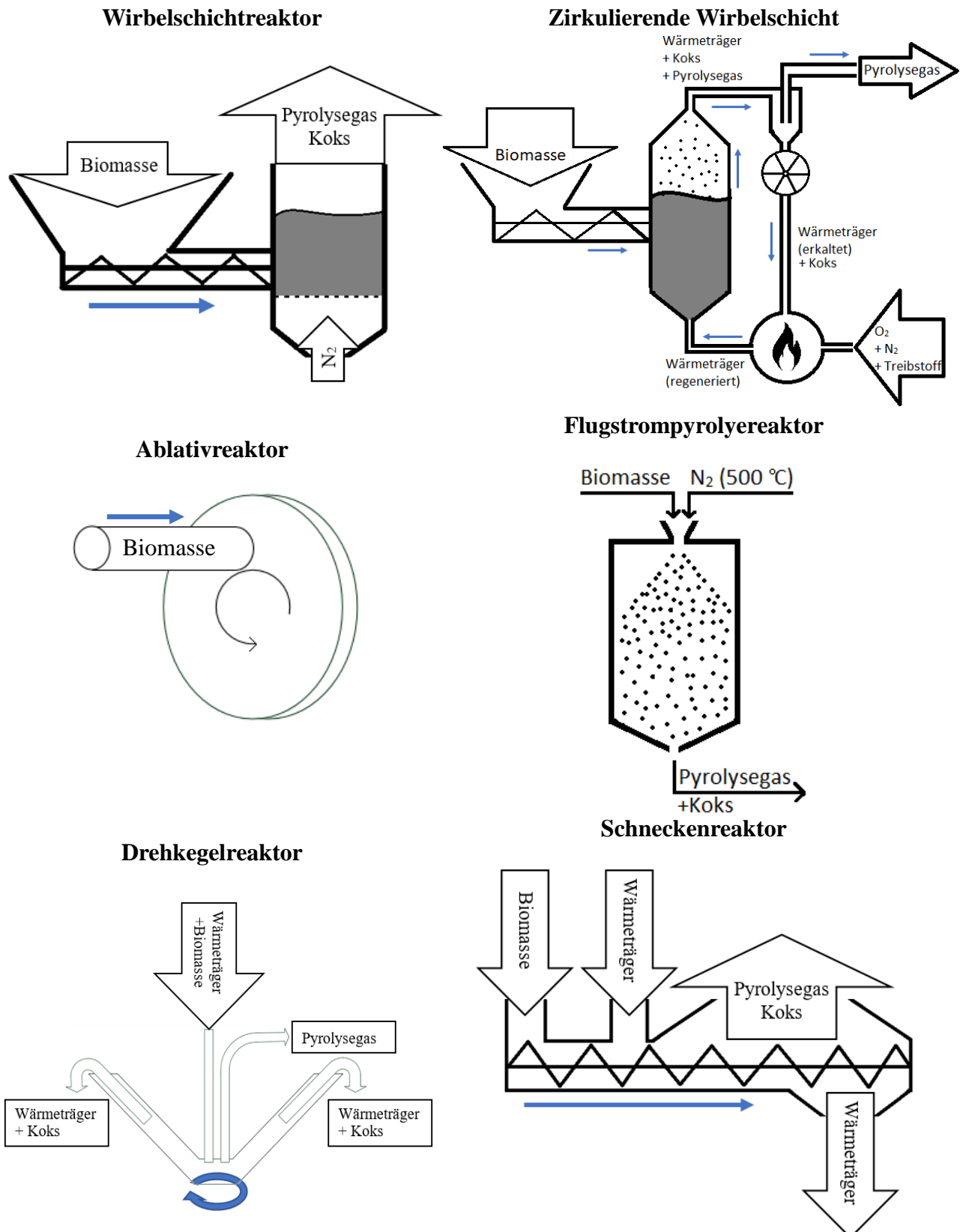


Abbildung 1.6: Schematischer Aufbau der Reaktortypen zur Schnellpyrolyse von Biomasse

Die **zirkulierende Wirbelschicht** stellt einen Ansatz dar, die Nachteile bei der Wirbelschicht zu umgehen. Thermisch regeneriert wird nicht das Fluidisierungsgas, sondern der Wärmeträger, der aufgrund einer höheren Gasgeschwindigkeit zusammen mit dem Koks aus dem Reaktor getragen

wird. Die Feststoffe Koks und Wärmeträger werden aus der Gasphase abgetrennt. Zur thermischen Regeneration des Wärmeträgers wird das Koks im Koks/Wärmeträgergemisch verbrannt, bevor der Wärmeträger wieder zurück in den Reaktor geleitet wird. Bei dieser Prozesskette wird jedoch die autokatalytische Asche im Prozesskreislauf akkumuliert. Die Firma Ensyn betreibt mehrere solcher Anlagen. Deren größte mit einer Kapazität von 70 t pro Tag steht in Wisconsin/USA und nutzt Holz um Raucharomen für die Lebensmittelindustrie zu produzieren [34].

Bei der **Flugstrompyrolyse** wird Biomasse und Wärmeträger in einem Rohrreaktor gegeben. Durch die hohen Gasgeschwindigkeiten werden diese direkt ausgetragen. Pyrolysegas und Koksverweilzeit sind hierbei nahezu identisch. Durch die hohen Volumenströme muss die Kondensation und die Aufheizung des zur Fluidisierung nötigen Gases entsprechend groß ausgelegt werden. Der Wärmeübergang erfolgt i.a. aus der Gasphase und ist vergleichsweise zu artverwandten Reaktorsystemen schlechter. Im größeren Maßstab wurden zwei Anlagen von den Firmen GTRI (50 kg/h, USA) und Egemin (200 kg/h, Belgien) gebaut. Beide sind heute u.a. wegen technischer Probleme und Wirtschaftlichkeit nicht mehr in Betrieb [29].

Beim **Drehkegelpyrolysereaktor** werden Biomasse und Sand kontinuierlich in einen drehenden Kegel gegeben. Bei Kontakt wird die Wärme vom Wärmeträger auf die Biomasse übertragen. Durch die Rotation im Konus wird das Partikelbett umgewälzt. Durch das dünne Partikelbett kann das Pyrolysegas zeitnah das Bett verlassen. Der Wärmeträger wird zusammen mit dem Koks auszentrifugiert. Thermisch regeneriert wird der Wärmeträger durch die Verbrennung des beigemischten Koks. Der Drehkegelreaktor zeichnet sich durch seine gute Durchmischung und damit verbundenen guten Wärmeübergang aus. Durch die mechanische Durchmischung ist kein Fluidisierungsgas erforderlich. Jedoch bilden die mechanisch bewegten Teile selbst eine Schwachstelle. Eine weitere Herausforderung ist die Abstimmung der Produktzugabe, Rotationsgeschwindigkeit, Verbrennung und des Sandkreislaufs aufeinander. Im industriellen Maßstab wird von der Firma Genting zwei Anlagen mit einer Kapazität von 50 t und 120 t pro Tag in Malaysia betrieben, erfolgreich weiterverkauft und zusätzliche Anlagen sind bereits im Bau [35]

Ein anderer Ansatz wird bei den **Schneckenreaktoren** verfolgt. Räumlich getrennt wird Wärmeträger und Biomasse zugegeben (Abbildung 1.6). Die Schnecke/-n übernehmen sowohl den Transport als auch die Vermischung. Durch das mechanische Mischen ist der Schneckenreaktor robust gegenüber Partikelverteilungen und größeren Partikeln. Darüber hinaus benötigt der Schneckenreaktor kein Fluidisierungsgas zum Betrieb. Koks wird zusammen mit dem Wärmeträger im Kreis gefahren, bis die Koks-Partikel durch mechanische Beanspruchung klein genug sind, um mit dem Pyrolysegas ausgetragen zu werden (Abbildung 1.7). Thermisch regeneriert werden kann der Wärmeträger durch elektrisches Aufheizen, durch die Verbrennung des Koks oder durch das heiße Liftgas, das den Wärmeträger umlaufen lässt – wie bei der bioliq der Fall.

Jedoch ist die mechanische Vermischung im Vergleich mit Wirbelschichtreaktoren häufig schlechter. Daher kommt, wie beim bioliq®-Verfahren, der Doppelschneckenmischreaktor zum Einsatz, der durch die zwei Schnecken ein besseres Mischverhalten gegenüber dem Schneckenmischer verspricht. Die Vielzahl an möglichen Parametern wie Drehgeschwindigkeit, Ganghöhe, Länge, Mischverhältnis Biomasse/Wärmeträger und Partikelmenge und Partikelgröße, macht den Reaktor tolerant gegenüber kleineren Abweichungen. Gleichzeitig erschweren die vielen Freiheitsgrade das Bestimmen der optimalen Betriebsparameter. Erforscht wurden diese am KIT an der bioliq®-Projektanlage mit einer Kapazität von 500 kg/h und der PYTHON (**P**yrolyseanlage zur **t**hermochemischen Umwandlung **o**rganischer Reststoffe) einer Technikumsanlage mit einer Kapazität von 10 kg/h (Abbildung 1.7). Die wesentlichen Unterschiede liegen in der Zugabereihenfolge und dem Wärmeträger. Bei der PYTHON besteht der Wärmeträger aus Stahlkugeln, bei der bioliq® aus Sand. Konrmyer beschäftigte sich im Rahmen seiner Dissertation [1] mit der Auslegung der PYTHON und führte in diesem Rahmen Grundlagenversuche durch.

Diese wurden im Blackbox-Ansatz durchgeführt. D.h. bis auf wenige Ausnahmen, konnten Vorgänge im Reaktor nur am Reaktoreinlass und –auslass beobachtet werden. Ursachen hierfür sind vielfältig. Fehlende Analysemethoden (bei 500 °C), komplexe chemische Reaktionen, Kondensation bei geringen Temperaturschwankungen, Staubbildung der Biomasse und mangelnde Einsicht in das Partikelbett – um einige zu nennen, erschweren die Charakterisierung des Reaktors.

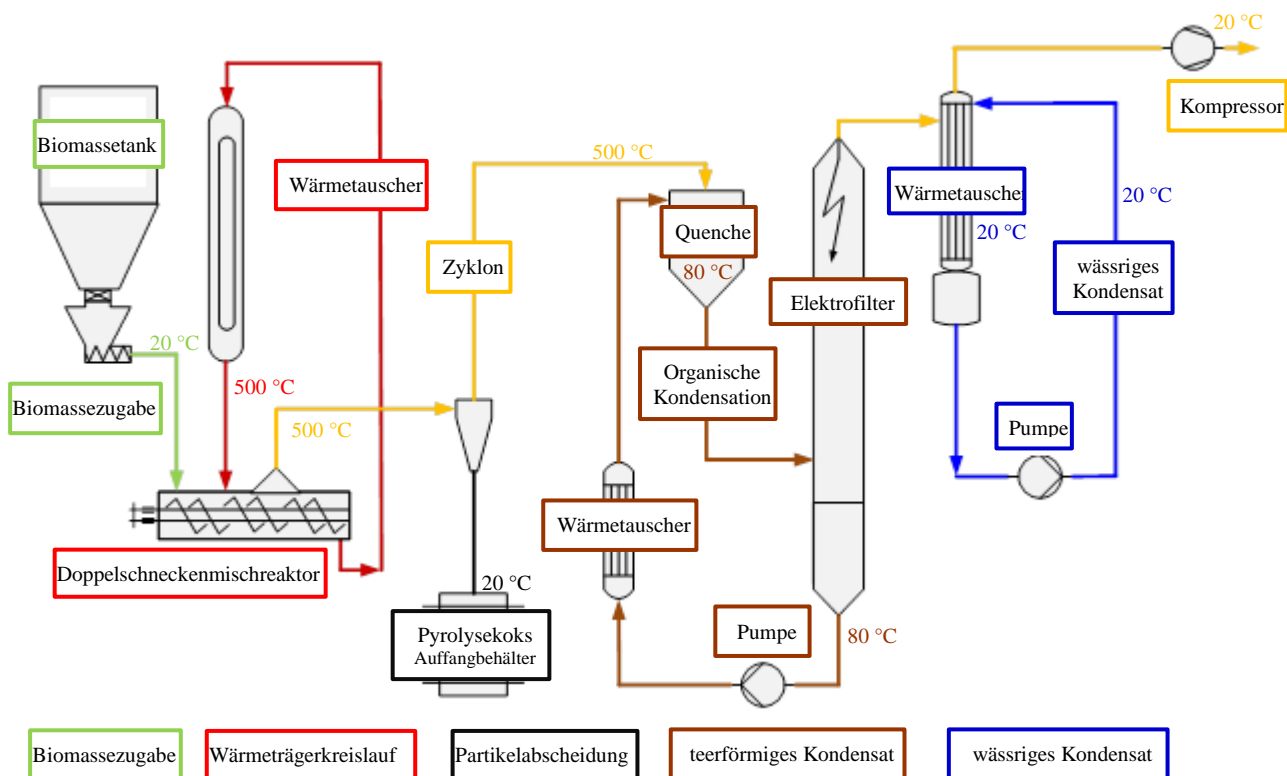


Abbildung 1.7: Schematischer Aufbau der PYTHON-Technikumsanlage nach Funke [36], modifiziert um Anlagenteile und Prozessströme besser dar zu stellen.

## 1.4 Aufgabenstellung

Eine Simulation hat im Vergleich zu einer experimentellen Charakterisierung den Vorteil, dass die Problemstellen unter idealisierten Bedingungen untersucht werden können. Hierfür erforderlich sind Simulationen mit einem hohen Maß an Realitätsnähe. Solche Simulationen stellen besondere Anforderungen an die zugrunde liegende Modelle, Simulationsparameter und Rechenzeit.

Die in dieser Arbeit durchgeführte Simulation der Pyrolyse von Weizenstroh im Doppelschneckenmischreaktor zur Betrachtung des Stoff- und Wärmeübergang umfasst die Aspekte:

- Granulat: 

Transport	}	→ Diskrete Element Methode (DEM)
Mischen		
Energieübertragung		
chemische Reaktion		
- Fluid: 

Mehrphasenströmung	}	→ numerische Strömungssimulation (CFD)
Reaktive Strömung		
Bewegte Objekte		

Das transportierte Partikelbett unterliegt einer zeitlichen Abhängigkeit und übt direkten Einfluss auf die Strömung aus. In diesem Fall empfiehlt es sich, einen Euler-Euler oder Euler-Lagrange-Ansatz zu verfolgen. Im Allgemeinen bildet eine Lagrange-Simulation das Partikelverhalten genauer ab als eine Euler-Simulation vermag. Daher ist Euler-Lagrange für diese Anwendung der bevorzugte Ansatz. Im Doppelschneckenmischreaktor spielt eine Fluidisierung des Partikelbettes durch die Strömung keine Rolle, so dass keine signifikante Abhängigkeit der Partikel von der umgebenden Strömung zu erwarten ist. In solch einem Fall, kann das Partikelverhalten mittels Lagrange-Simulation wiedergegeben werden. Jedoch ist es erforderlich für beide Methoden, Kennwerte zu bestimmen und Modelle zu optimieren.

Für den Lagrange-Ansatz wird das DEM-Tool LIGGGHTS®-Public - eine OpenSource-Programm, gewählt. Dies erfordert im Vorfeld die Charakterisierung des Weizenstrohs im Hinblick auf Form, Größenverteilung und Partikeldichte, d.h. die Skelettdichte inklusive Porenvolumen. Des Weiteren sind Reib- und Rollreibungswerte für Wärmeträger und Biomasse, sowie deren Interaktion zu bestimmen. Gebräuchliche Annahmen wie die Reduzierung des E-Moduls zur Erhöhung der Stabilität und Schrittweite sind zu verifizieren. Final ist das verfügbare Wärmeübergangsmodell, welches den Wärmeübergang als Leitung durch die Kontaktfläche darstellt, zu überprüfen.

Diese Vorbetrachtungen erlauben Studien betreffend der optimalen Rotationsgeschwindigkeit der Schnecken, der Zugabereihenfolge und des idealen Verhältnis von Biomasse und Wärmeträger, sowie der Lokalisierung von Problemstellungen wie, z.B. einer unerwünschten Rückvermischung.

Bei der Euler-Lagrange-Simulation des Doppelschneckenmischreaktors kommt das OpenSource-Programm CFDEM®-Public zum Einsatz. CFDEM® bietet die Möglichkeit LIGGGHTS® mit einer inkompressiblen Strömungssimulation, d.h. der CFD-Toolsammlung OpenFOAM®, zu koppeln. Die Betrachtung von chemischen Reaktionen in einem breiten Temperaturspektrum macht die Erweiterung um die Kompressibilität und die Implementierung eines Reaktionsmodells erforderlich. Darüber hinaus fehlen in der Strömungssimulation bislang geeignete Methoden zur Berücksichtigung der co-rotierenden, ineinander kämmenden Schnecken, so dass auch hier eine geeignete Methode gefunden und implementiert werden muss. Eine Parameterstudie bzgl. der Reaktionskinetik der Lignocellulose Weizenstroh schafft die Grundlage, welche eine Euler-Lagrange-Simulation des Doppelschneckenmischreaktors bei der Pyrolyse von Weizenstroh ermöglicht. Diese ist mit dem Ziel einer Charakterisierung des Doppelschneckenmischreaktors, der Reaktionsverteilung, des Wärmeüberganges und dem entstandenen Euler-Lagrange-Simulationstool durchzuführen.

## 2 Discrete Element Methode

Die Discrete Element Methode (DEM) bieten die Möglichkeit Feststoffbewegungen z.B. Steine, Sand usw. in einem Haufwerk zu simulieren [37]. Ihren Ursprung hat die DEM-Simulation in der Moleküldynamik (MD). Anfangs noch experimentell, wurden Molekülbewegungen in einem Kristallgitter oder auch Fluid unter Zuhilfenahme von Bällen erforscht [38].

Die Simulation der Moleküldynamik stellte lange Zeit, aufgrund des hohen Rechenaufwandes bei der Nachbarsuche, d.h. das Identifizieren sich berührender Partikel, eine Nische in der Wissenschaft dar. Erst durch die Entwicklung von Methoden zur effizienten Bestimmung von Nachbarn, der Optimierung von Algorithmen und dem technischen Fortschritt, wurde es möglich, mehrere Millionen Atome auf einem einzelnen Prozessorkern zu simulieren.

Aus der Moleküldynamik entwickelte sich schließlich die Discrete Element Methode (DEM), durch die Berücksichtigung von Reibung und Rollreibung, sowie durch die Einführung des Softsphere-Ansatzes, bei der eine Überlappung zweier Partikel mit einer steigenden Abstoßung beider einher geht [39].

### 2.1 Grundlagen der DEM-Simulation

Bei der Discrete Element Methode werden die Partikel als Punkte in einem zwei/drei-dimensionalen Raum mit vordefinierten Eigenschaften interpretiert, auf die Kräfte  $F$  einwirken. Diese setzen sich aus der Partikelmasse  $m$  und der Partikelgeschwindigkeit  $v$  bzw. deren Ableitung der Beschleunigung  $a$  zusammen (Gleichung 2.1). D.h., dass DEM-Simulationen dem Newtonschen Bewegungssatz folgen.

$$F = \frac{d(m \cdot v)}{dt} = m \frac{dv}{dt} = m \cdot a \quad (2.1)$$

Man unterscheidet zwischen langreichenden Kräften, wie Erdanziehung mit Gravitationskonstante  $g = 9.81 \text{ m} \cdot \text{s}^{-2}$  und kurzreichenden Kräften, worunter alle Partikel/Partikel-Interaktionen  $F_{i,j}^N$  fallen. Diese wirken ausschließlich, wenn der euklidische Abstand  $d_{i,j}$  der Schwerpunkte von Partikel  $i$  und  $j$  kleiner ist, als deren Radien  $r_i$  und  $r_j$  zusammen (Gleichung 2.2).

$$d_{i,j} < r_i + r_j \quad (2.2)$$

Da im Gegensatz zu gitterbasierten Methoden, die Position im Raum nicht bekannt ist und erst bestimmt werden muss, ist es erforderlich jeden Nachbar in jedem Zeitschritt neu zu bestimmen.

Nach dem Superpositionsprinzip werden die resultierenden weit- und kurzreichenden Kräfte addiert (Gleichung 2.3).

$$m_i \frac{dv_i}{dt} = \sum_{j=1}^n F_{i,j}^N + m_i g \quad (2.3)$$

Die Partikel/Partikel-Interaktionen werden nach Hertz [40] berechnet (Abbildung 2.1). Dabei werden die normal anliegende Kraft  $F_{i,j}^{H,N}$  und die tangential anliegende Kraft  $F_{i,j}^{H,T}$  unabhängig voneinander betrachtet (Gleichung 2.4).

---

Unter der Annahme einer konstanten Masse

$$F_{i,j}^H = F_{i,j}^{H,N} + F_{i,j}^{H,T} \quad (2.4)$$

Sowohl die normal anliegende als auch die tangential anliegende Kraft werden durch eine Kombination aus dämpfender Kraft mit Dämpfungskonstante  $\gamma$  und als federnde Kraft mit Federkonstante  $k$  interpretiert (Abbildung 2.1).

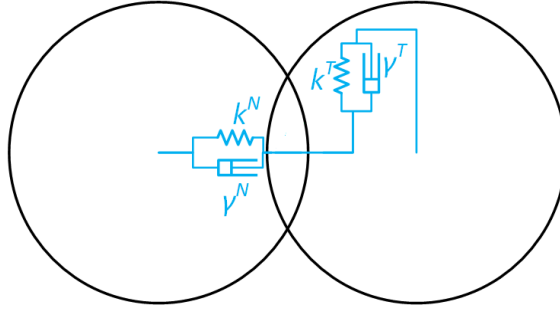


Abbildung 2.1: Kontaktmodell nach Hertz:

D.h. für die Partikelinteraktion nach Hertz gilt:

$$F_{i,j}^H = \underbrace{\underbrace{(k^N \cdot \delta_{i,j}^N - \gamma^N \cdot v_{i,j}^N)}_{=F_{i,j}^{H,N,F}} + \underbrace{\underbrace{(k^T \cdot \delta_{i,j}^T - \gamma^T \cdot v_{i,j}^T)}_{=F_{i,j}^{H,T,D}}}_{=F_{i,j}^{H,T}}}_{=F_{i,j}^{H,N}} \quad (2.5)$$

Während die normal anliegende Kraft  $F_{i,j}^{H,N}$  von der Überlappung  $\delta_{i,j}^N = d_{i,j} - r_i - r_j$  und von der relativen Normalgeschwindigkeit  $v_{i,j}^N$ , abhängt, ist die tangential anliegende Kraft  $F_{i,j}^{H,T}$  von der relativen tangential Geschwindigkeit  $v_{i,j}^T$  und von der tangentialen Überlappung  $\delta_{i,j}^T$  abhängig.

$$\delta_{i,j}^T = \int_{t_0}^t v_{i,j}^T dt$$

$$\delta_{i,j}^T = \delta_{i,j}^T_{n-1} + \left| v_{i,j}^T_{n-1} \cdot \frac{v_{i,j}^T_{n-1}}{\|v_{i,j}^T_{n-1}\|} \right| \cdot (t_n - t_{n-1}) \quad \text{mit } \delta_{i,j}^T_0 = 0 \quad (2.6)$$

Dabei wird die tangential Überlappung  $\delta_{i,j}^T$  iterativ durch Integration der relativen tangentialen Bewegungen über die Kontaktzeit bestimmt (Gleichung 2.6). Dabei gilt:  $F_{i,j}^{H,T} \leq COF \cdot F_{i,j}^{H,N}$ .  $COF$  (Coefficient of Friction) stellt den Reibungswert dar.

Sowohl die Federkonstanten  $k^N$  und  $k^T$  als auch die Dämpfungsfaktoren  $\gamma^N$  und  $\gamma^T$  können direkt aus den Materialeigenschaften bestimmt werden (Gleichungen 2.7).



$$\begin{aligned}
k^N &= \frac{4}{3} E_{i,j} \cdot \sqrt{r_{i,j} \cdot \delta_{i,j}^N} \\
\gamma^N &= -2 \cdot \sqrt{\frac{5}{6}} \cdot \beta \cdot \sqrt{S^N \cdot m_{i,j}} \geq 0 \\
k^T &= 8 \cdot G_{i,j} \cdot \sqrt{r_{i,j} \cdot \delta_{i,j}^N} \\
\gamma^T &= -2 \cdot \sqrt{\frac{5}{6}} \cdot \beta \cdot \sqrt{S^T \cdot m_{i,j}}
\end{aligned} \tag{2.7}$$

Dazu werden die mittlere Masse  $m_{i,j}$ , das mittlere E-Modul  $E_{i,j}$ , das mittlere Schubmodul  $G_{i,j}$  und der mittlere Radius  $r_{i,j}$  mit den Gleichungen 2.8-2.14 berechnet.

$$\beta = \frac{\ln COR}{\ln^2 COR + \pi^2} \tag{2.8}$$

$$S_{ij}^N = 2 \cdot E_{i,j} \cdot \sqrt{r_{i,j} \cdot \delta_{i,j}^N} \tag{2.9}$$

$$S_{ij}^T = 8 \cdot G_{i,j} \cdot \sqrt{r_{i,j} \cdot \delta_{i,j}^N} \tag{2.10}$$

$$\frac{1}{r_{i,j}} = \frac{1}{r_i} + \frac{1}{r_j} \tag{2.11}$$

$$\frac{1}{E_{i,j}} = \frac{(1 - \vartheta_i^2)}{E_i} + \frac{(1 - \vartheta_j^2)}{E_j} \tag{2.12}$$

$$\frac{1}{G_{i,j}} = \frac{2 \cdot (2 - \vartheta_i) \cdot (1 + \vartheta_i)}{E_i} + \frac{2 \cdot (2 - \vartheta_j) \cdot (1 + \vartheta_j)}{E_j} \tag{2.13}$$

$$\frac{1}{m_{i,j}} = \frac{1}{m_i} + \frac{1}{m_j} \tag{2.14}$$

Zur Berücksichtigung der Rollreibung wurde das Hertz Model um das Drehmodel  $M^H$  erweitert (Gleichungen 2.15-2.17). Das Drehmoment wird iterativ nach Ai et. al. [41], aus den vorhergehenden Zeitschritten berechnet:

$$M_{i,j;t+dt}^H = M_{i,j;t}^H + dM_{i,j;t}^H \tag{2.15}$$

$$dM_{i,j;t}^H = -k_{i,j}^{RF} \cdot d\theta_{i,j} \tag{2.16}$$

$$k_{i,j}^{RF} = k_{i,j}^T \cdot r_{i,j}^2 \tag{2.17}$$

Dabei stellt  $d\theta_{i,j}$  die relative Rotationsgeschwindigkeit am Kontaktpunkt dar. Die verwendete Federkonstante  $k_{i,j}^T$  (Gleichung 2.17) entspricht der Federkonstanten für tangentielle Reibung (Gleichung 2.7).

$$|M_{i,j;t+dt}^H| \leq CORF \cdot r_{i,j} \cdot F_{i,j}^N \tag{2.18}$$

Limitiert wird das Drehmoment durch die wirkende Normalkraft  $F_{i,j}^N$ , dem Äquivalenzradius  $R_{i,j}$  und dem  $CORF$  (Coefficient of Rolling Friction).

### 2.1.1 Integration

Die auf ein Partikel einwirkenden Kräfte werden addiert und integriert. Als Integrator dient meist eine explizite numerische Integrationsmethode, wie z.B. der Velocity-Verlet-Algorithmus [42] (Gleichung 2.19).

$$\begin{aligned}x_{t+dt} &= x_t + v_t \cdot dt + \frac{1}{2} \cdot a_t \cdot dt^2 + O(dt^4) \\v_{t+dt} &= v_t + \frac{1}{2} \cdot (a_t + a_{t+dt}) \cdot dt + O(dt^2)\end{aligned}\tag{2.19}$$

Dieser bestimmt die Position  $x$  mit einer lokalen Abweichung von  $O(dt^4)$  und die Geschwindigkeit  $v$  mit einer lokalen Abweichung von  $O(dt^2)$ . Die globale Abweichung ist somit  $O(dt^2)$ . Unter der Annahme eines konstanten Zeitschrittes  $dt$ , lässt sich der Algorithmus vereinfacht darstellen:

1. Bestimme die Geschwindigkeit zum Zeitschritt  $n + \frac{1}{2}$

$$v_{n+\frac{1}{2}} = v_n + \frac{1}{2} \cdot a_n \cdot dt\tag{2.20}$$

2. Bestimme Position zum Zeitschritt  $n + 1$

$$x_{n+1} = x_n + v_{n+\frac{1}{2}} \cdot dt\tag{2.21}$$

3. Bestimme  $F_{n+1}^H$  aus den Interaktionen in  $x_{n+1}$ . Dann ist:  $a_{n+1} = F_{n+1}^H/m$

4. Bestimme die finale Geschwindigkeit aus  $v$  zum Zeitschritt  $n + 1$

$$v_{n+1} = v_{n+\frac{1}{2}} + \frac{1}{2} \cdot a_{n+1} \cdot dt\tag{2.22}$$

Eine Abschätzung für den maximalen Zeitschritt  $dt$  liefert die Rayleigh-Zeit  $t_{Ray}$ . Diese gibt an, wie lange ein Impuls benötigt, um ein Partikel zu durchwandern.

$$t_{Ray_i} = \frac{\pi \cdot r_i}{0.8766 + 0.1631 \cdot \nu} \cdot \sqrt{\frac{\rho_i}{G_i}}\tag{2.23}$$

Die Rayleigh-Zeit ist im Wesentlichen von den Materialeigenschaften (Dichte  $\rho_i$ , Schubmodul  $G_i = \frac{E_i}{2 \cdot (1 + \nu_i)}$  und Poisson-Zahl  $\nu_i$ ) abhängig, aber auch von der Partikelform (Radius  $r_i$ ). Meist wird ein Zeitschritt  $dt \approx 0.2 \cdot \max(t_{Ray_i})$  gewählt [43]. Weitere Kriterien für Sonderfälle ist die Hertzzeit [44] und die Cundallzeit [45]. Zeitschrittkriterien können jedoch nur als Richtwert verwendet werden. So kann in einigen Fällen ein größerer Zeitschritt gewählt werden, während es in anderen erforderlich ist diesen zu verringern [46].

## 2.1.2 Nachbarsuche

Mit steigender Partikelzahl wächst der Aufwand bei der Nachbarsuche quadratisch. Daher konnten lange Zeit keine größeren Partikelzahlen simuliert werden. Dies änderte sich erst mit der Einführung eines imaginären strukturierten Gitters. Jedem Partikel wird zu jedem Zeitschritt eine Gitterzelle zugeordnet. Da die Struktur des Gitters unverändert bleibt, sind auch die Nachbarzellen einer beliebigen Zelle bekannt. D.h. die kontaktierenden Nachbarpartikel sind entweder in der eigenen oder in den benachbarten Gitterzellen zu finden. Daher ist eine Gitterzellbreite des Partikeldurchmessers zu empfehlen. Durch die Einführung dieses Zwischenschrittes wurde der Aufwand für die Nachbarsuche linear.

Heute wird eine Kombination aus dieser Gittersuche und einer von Verlet et. al. [47] der sog. Verlet-Liste angewandt. Dabei wird jedem Partikel eine Liste mit Partikeln zugeordnet. Für diese gilt:

$$d_{i,j} \leq r_i + r_j + c$$

$c$  ist der sog. skin Parameter. Dieser sorgt dafür, dass nicht nur in Kontakt stehende Partikel in der Liste enthalten sind, sondern auch räumlich benachbarte mit einem maximalen Abstand von  $c$ . Diese Liste wird durch die Gittersuche generiert und behält

$$N_{\text{Verlet}} = \frac{c}{2 \cdot v_{\text{max}} \cdot \Delta t}$$

ihre Gültigkeit bei [48].

## 2.2 Parameterbestimmung bei der DEM-Simulation

Eine DEM-Simulation eines binären Gemisches aus Weizenstroh und Stahlkugeln bzw. Wärmeträgern stellt aufgrund unbekannter Materialeigenschaften, aber auch durch die abweichende Form der Weizenstrohpartikel von der idealen Kugel, eine große Herausforderung dar. Um eine möglichst akkurate Aussage über die Mischeigenschaften im Doppelschneckenmischreaktor treffen zu können, ist es unabdinglich möglichst genaue Materialkennwerte zu generieren und die Weizenstrohform zu approximieren.

DEM-Simulationen in ihrer ursprünglichen Form bieten die Möglichkeit sphärische Partikel in großer Zahl zu simulieren. Zum Beginn dieser Arbeit bestanden drei Möglichkeiten, nicht-sphärische Partikel zu simulieren.

Partikel deren Form nur gering einer Kugel abweicht, können dennoch durch eine Kugel angenähert werden. Die resultierende Abweichung im Partikelverhalten kann durch eine Modifizierung der Werte für Reibung und Rollreibung dargestellt werden [49]. Diese Methode hat mehrere Vorteile

- die Algorithmen für sphärische Partikel sehr einfach zu implementieren
- der Rechenaufwand ist am geringsten
- die Stabilität ist am größten.

Daher ist sie das Mittel der Wahl bei kleineren Formabweichungen.

Eine weitere Methode zur Simulation nicht-sphärischer Partikel, ist die Multisphärensimulation [50]. Dabei wird die Form der Partikel durch eine Vielzahl von Sphären approximiert, deren Position in Relation zu den anderen unverändert bleibt. Da der Rechenaufwand proportional zur Anzahl der Sphären ist und durch den Multisphärenansatz mehr simuliert werden müssen, ist auch der Aufwand entsprechend höher. Des Weiteren werden kleinere Sphären zur Darstellung verwendet. D.h. die

erforderliche Schrittweite muss an den Sphärenradius angepasst werden (Gleichung 2.23) [51]. Daher wird diese Methode nur zur Darstellung einer moderaten Anzahl an Partikeln verwendet.

Die letzte Möglichkeit stellen die Superquadriken dar, d.h. Partikel deren Form durch mathematische Funktionen dargestellt wird [52]. Superquadriken sind artverwandt mit Superellipsoiden und können Würfel, Zylinder, Oktaeder, Kugeln und Torus darstellen.

Im Vergleich zu dem Multisphärenansatz muss nur ein Partikel simuliert werden. Da der Aufwand von Nachbar- und Kontaktsuchen bei nichtsphärischen Partikeln um ein Vielfaches höher ist, muss von Fall zu Fall entschieden werden, welche Methode die geeignetere darstellt [52].

Zu Beginn dieser Promotion (2013), war diese Methode Gegenstand der Forschung und erst in jüngster Zeit (2017/2018) verfügbar. Daher konnte diese Methode im Rahmen dieser Arbeit nicht berücksichtigt werden.

Im weiteren Verlauf dieser Arbeit werden sowohl Weizenstroh als auch der Wärmeträger durch Sphären dargestellt. Dieses Kapitel beschäftigt sich daher mit den verändernden Materialkennwerten und betrachtet den Einfluss der abweichenden Geometrie.

### **2.2.1 Bestimmung Materialkennwerte, Kennwerte für die Wärmeträger/Wärmeträger und Wärmeträger/Wand Interaktionen**

Zur Optimierung der Simulationszeit wird bei DEM-Simulationen häufig ein niedriges E-Modul gewählt. Konkret kann durch ein um den Faktor 100 reduzierter E-Modul die Simulationszeit um den Faktor 10 beschleunigen (Gleichung 2.13 und Gleichung 2.23). Dies hat direkten Einfluss auf die Reib- und Rollreibungswerte und macht deren Optimierung erforderlich. Dabei ist das generelle Vorgehen stets gleich. Ein Versuch wird mit den zu simulierenden Partikeln durchgeführt. Diese Versuche werden parallel dazu mit veränderten Reib- und Rollreibungswerten und identischen Rahmenbedingungen nachsimuliert. Die Parameter der Simulation mit der besten Übereinstimmung zum Versuchsergebnis entsprechen dem gesuchten Optimum [49]. Die allgemeinen Anforderungen an den Versuch sind:

- Kostengünstiger Aufbau
- Einfache Durchführung
- Einfache Auswertung
- Mit wenig Rechenaufwand nachsimuliert und ausgewertet

Anhand dieser Kriterien, kann eine Vielzahl von Versuchen zur Parameteroptimierung verwendet werden. Al-Hashemi et. al. [53] verwendeten zur Parameteroptimierung z.B. den Schüttwinkelversuch, Höhner et. al. [54] verwendeten den klassischen Reibversuch, bei der DCS Computing GmbH wurde der Ringshertest als Versuch verwendet [55] und Piotr. et. al. [56] optimierte anhand des Schüttwinkels in der rotierenden Trommel.

Aufgrund stark abweichender Materialeigenschaften von Stahl und Biomasse bzgl. Dichte, Reibung, Rollreibung und Wert für den elastischen Stoß, können viele der klassische Methoden, zur Bestimmung und Optimierung dieser Reibwerte nicht verwendet werden.

Eine Ausnahme stellt die rotierende Trommel dar, deren Inhalt durch Rotation der Trommel umgewälzt wird.

Da überwiegend Feststoffpartikel umgewälzt und vermischt werden, stellt sich bei einer konstanten Drehgeschwindigkeit der Trommel eine nahezu konstante Silhouette des Bettes ein.

Die generelle Form des Partikelbetts, durchwandert in Abhängigkeit der Materialeigenschaften und Rotationsgeschwindigkeit bis zu vier allgemeine Formen. Ist die Rotationsgeschwindigkeit niedrig, so stellt sich ein nahezu linearer Horizont ein. Bei steigender Rotationsgeschwindigkeit prägt sich ein Wirbel, im allgemeinen als Katarakt bezeichnet, in der Silhouette aus. Dieser prägt sich bei einer

Froudezahl von größer 1 aus. Bei hohen Drehgeschwindigkeiten, stellt sich ein kreisförmiger Horizont ein [57]. Siehe (Abbildung 2.2).

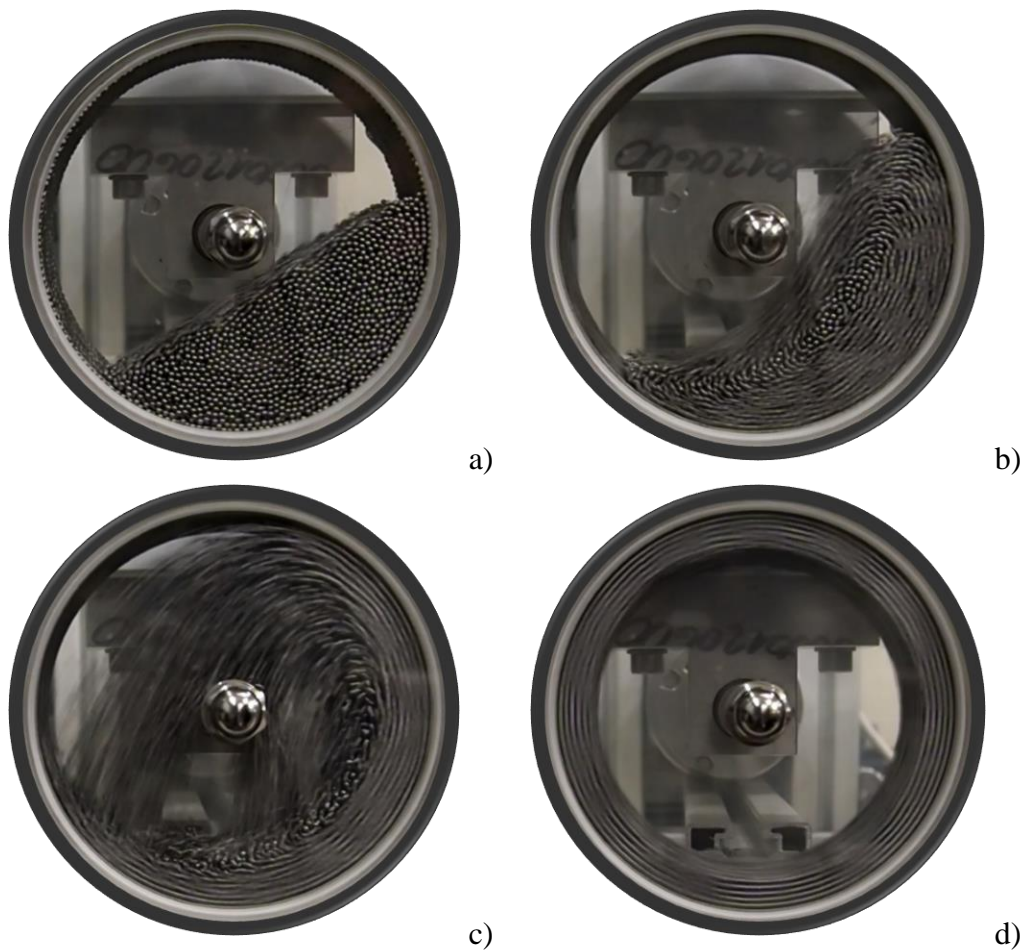


Abbildung 2.2: Zustände bei einer rotierenden Trommel in Abhängigkeit der Drehgeschwindigkeit. a) 10 rpm, b) 60 rpm, c) 120 rpm und d) 200 rpm

Dieses Verhalten lässt sich mit der dimensionslosen Froudezahl  $Fr$  (Gleichung 2.24), die das Verhältnis aus Zentrifugalkraft und Schwerkraft angibt, beschreiben [58].

$$Fr: \quad \begin{aligned} Fr_p &= \frac{\omega^2 \cdot R_p}{g} \\ Fr_w &= \frac{(2\pi \cdot n)^2 \cdot R_w}{g} \end{aligned} \quad (2.24)$$

Die Froudezahl lässt sich unterteilen in die Partikelfroudezahl  $Fr_p$  und die Werkzeugfroudezahl  $Fr_w$ . Die Partikelfroudezahl ist abhängig von der Winkelgeschwindigkeit  $\omega$ , dem Partikelabstand zur Rotationsachse  $R_p$  und der Erdbeschleunigung  $g$ . Die Werkzeug-Froudezahl  $Fr_w$  hängt von der Drehgeschwindigkeit  $n$  und dem Werkzeugradius  $R_w$  ab.

Für beide Froude-Zahlen gilt: Ist die Froudezahl  $Fr < 1$ , bilden die Partikel ein Bett, bei einer Froude-Zahl  $> 1$  lösen sich Partikel aus dem Bett (Kataraktentwicklung). Da bei  $Fr > 1$  eine Entmischung stattfindet, ist dieser Bereich i. A. unerwünscht.

Table 2.1: Froude-Zahlen der Betriebsbedingungen zu (Abbildung 2.2)

$rpm$	$Fr_w$
10	0.01
60	0.25
120	1.0
200	2.8

Auf eine geeignete Größe und Partikelzahl reduziert, lassen sich die Betriebsbedingungen bis 60  $rpm$  (Abbildung 2.2) zur Validierung und Anpassung von simulationsabhängigen Parametern mit einem großen Wertebereich nutzen [49]. Der im Rahmen dieser Arbeit verwendete Modellversuch basiert auf diesem System.

Eine Stahltrommel mit 125  $mm$  Durchmesser und 80  $mm$  Breite wird mit Stahlkugeln mit einem Durchmesser von 2  $mm$  gefüllt. Die Deckel der Trommel bestehen aus durchsichtigem Acrylglas. Auf diese Weise kann eine Kamera, die sich auf der Rotationsachse befindet, den Umriss des Bettes erfassen (Abbildung 2.3). Da sich die Stahlkugeln im Wesentlichen orthogonal zur Rotationsachse bewegen, ist der Anpressdruck der Stahlkugeln zur Acrylglaswand zu vernachlässigen.

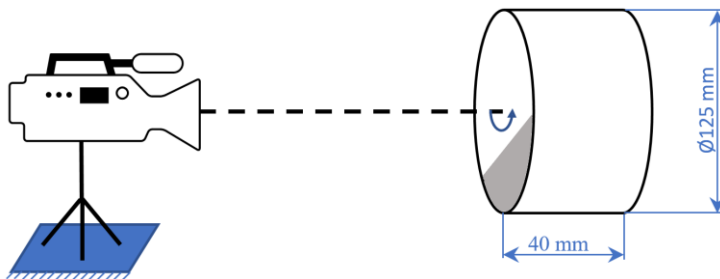


Abbildung 2.3: Versuchsaufbau des Drehtrommel-Modellversuches

Zur Auswertung verwendete Piotr et. al. [56] die Bestimmung des Schüttwinkels. Dies ist jedoch nur im niedrigen Drehzahlbereich möglich, da sich nur dort ein flaches Bett ausprägt (Abbildung 2.2, a). Um den Zustand im  $Fr = 1$  Bereich besser erfassen zu können, wird im Zuge dieser Arbeit die Auswertung dieses Versuches, durch den diskreten Abstand  $l_i$  des Bettes zur Rotationsachse erfolgen. Als Diskretisationswinkel  $\alpha_i$  wurde  $22.5^\circ$  gewählt (Abbildung 2.4).

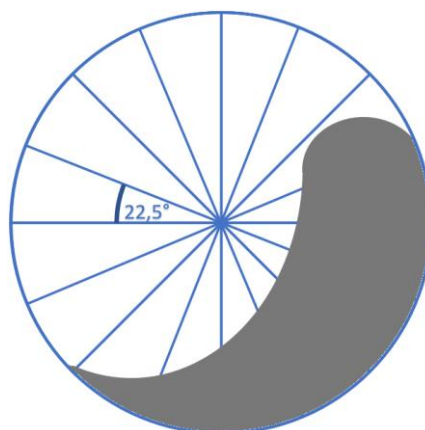


Abbildung 2.4: Auswertung des Drehtrommel-Modellversuches

Bei der DEM-Simulation dieses Versuches, wird die Position der Partikel in eine zwei dimensionalen Projektion überführt (Gleichung 2.25). Diese Projektion wird daraufhin in die Polarkoordinatendarstellung mit Winkel  $\varphi$  und Azimut  $R_p$  umgewandelt (Gleichung 2.26). Bestimmt

werden die kleinsten Azimute  $R_p$  (Gleichung 2.25-2.27), deren Winkel  $\varphi$  in einem Bereich  $\pm 2.5^\circ$  um  $\alpha_i$  liegt.

$$(x, y, z) \rightarrow (x, y) \quad (2.25)$$

$$(x, y) \rightarrow (R_p, \varphi) \quad (2.26)$$

$$\alpha_i - 2.5^\circ \leq \varphi_{ij} \leq \alpha_i + 2.5^\circ \quad (2.27)$$

Die Validierung der Modellparameter erfolgt über die Bestimmung der Abweichung (Gleichung 2.28).

$$Abw. = \sum_{i=1}^{16} \left| l_i - \frac{1}{10} \sum_{j=1}^{10} r_{ij} \right| \quad (2.28)$$

Stehen keine zehn Partikel zur Verfügung, so wird ein Mittelwert über die verfügbaren Werte gebildet. Steht kein Wert zur Verfügung, wird der Radius der Trommel verwendet.

### 2.2.1.1 Untersuchung der Wärmeträger – Wärmeträger Reibung und Rollreibung

Aufgrund von Interaktionen der beteiligten Materialien, stellt eine Bestimmung der Werte für die Reibung und Rollreibung eine Herausforderung dar.

Um die Anzahl der variierenden Parameter zu reduzieren, wurde zunächst der Einfluss der Wand-Wärmeträger Reibung und Rollreibung eliminiert, indem eine Kugellage auf der Wandoberfläche aufgeklebt wurde. Auf diese Weise ist es möglich, ausschließlich den Einfluss der Wärmeträger-Wärmeträgerparameter ohne Abhängigkeiten von weiteren Einflüssen zu beobachten.

Tabelle 2.1: Programmparameter der Wärmeträger-Wärmeträger Screeningversuche

<b>Partikeleigenschaften:</b>		<b>Programm:</b>	
E-Modul [Pa]	210 · 10 <sup>6</sup> vgl. [59]	3 s	mit 10 rpm
Poissonzahl	0.3 vgl. [60]	3 s	mit 25 rpm
COR	0.9 vgl. [61]	3 s	mit 60 rpm
COF	0.1 – 1.0		
CORF	0.001 – 0.5		
Dichte [kg/m <sup>3</sup> ]	7878	dt [s]	0.000006
Masse [kg]	0.63395		
Radius [m]	0.001		

Die in Tabelle 2.1 beschriebenen Parameter wurden in einer Reihe von Simulationen genutzt, die sich nur in ihrem Coefficient of Friction (COF - Reibung) und ihrem Coefficient of Rollingfriction (CORF - Rollreibung) unterscheiden. Die Definition des Coefficient of Friction (COF), Restitution (COR) und des Coefficient of Rollingfriction (CORF) entsprechen Kapitel 2.1. Dabei wurden Simulationen mit COF von 0.1 bis 1.0 in 0.1er Schritten und mit CORF von 0.001 bis 0.01 in 0.001er, von 0.01 bis 0.1 in 0.01er und von 0.1 bis 0.5 in 0.1er Schritten durchgeführt. Eine Besonderheit stellt in diesem Zusammenhang das E-Modul dar. Dieses wurde um den Faktor 10<sup>3</sup> gegenüber den Literaturwerten reduziert. Dies ist auf die Standardoperationsbedingungen von LIGGGHTS® zurückzuführen, die den Gültigkeitsbereich des E-Modul  $E$  auf  $5 \cdot 10^6 \text{ Pa} \leq E \leq 10^9 \text{ Pa}$  beschränken [62], aber auch

auf den signifikanten Geschwindigkeitsvorteil, der mit einer niedrigeren Schubspannung einher geht [63]. Im Rahmen dieser Arbeit wird an anderer Stelle ausführlich auf diese Thematik eingegangen.

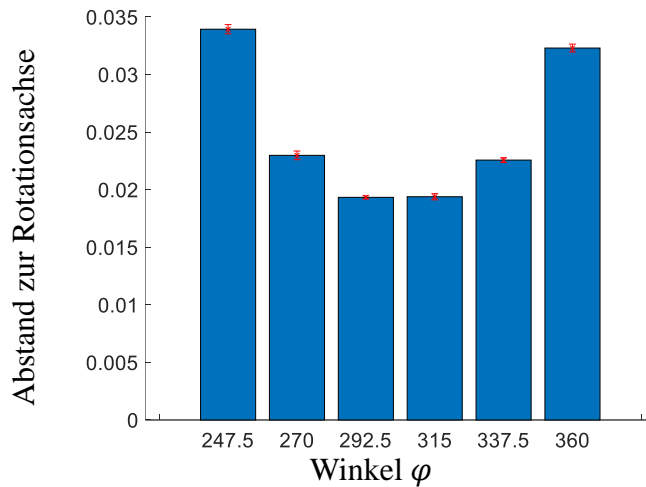


Abbildung 2.5: Qualität der Simulationsergebnisse.  
Abstand in blau, Varianz in rot

Zur Gewährleistung eines stationären Zustandes wurde eine Simulationszeit von 3 Sekunden gewählt. In Abbildung 2.5 wurde an den gewählten Messstellen, der Mittelwert und die Varianz des Abstands zur Rotationsachse an den gegebenen Messstellen aus den Zeitpunkten 1.8 s, 2.4 s und 3.0 s bestimmt.

Tabelle 2.2: Froude-Zahlen der Betriebsbedingungen

	$Fr_w$
<b>10 rpm</b>	0.01
<b>25 rpm</b>	0.04
<b>60 rpm</b>	0.25

Dies wurde für eine Rotationsgeschwindigkeit von 10 rpm mit den in Tabelle 2.3 festgelegten Werten für die Reibung und Rollreibung durchgeführt. Die maximal ermittelte Standardabweichung (Abbildung 2.5) beträgt 0.4 mm. Dies entspricht einem halben Kugelradius. Daher kann von einem stationären Zustand ausgegangen werden.



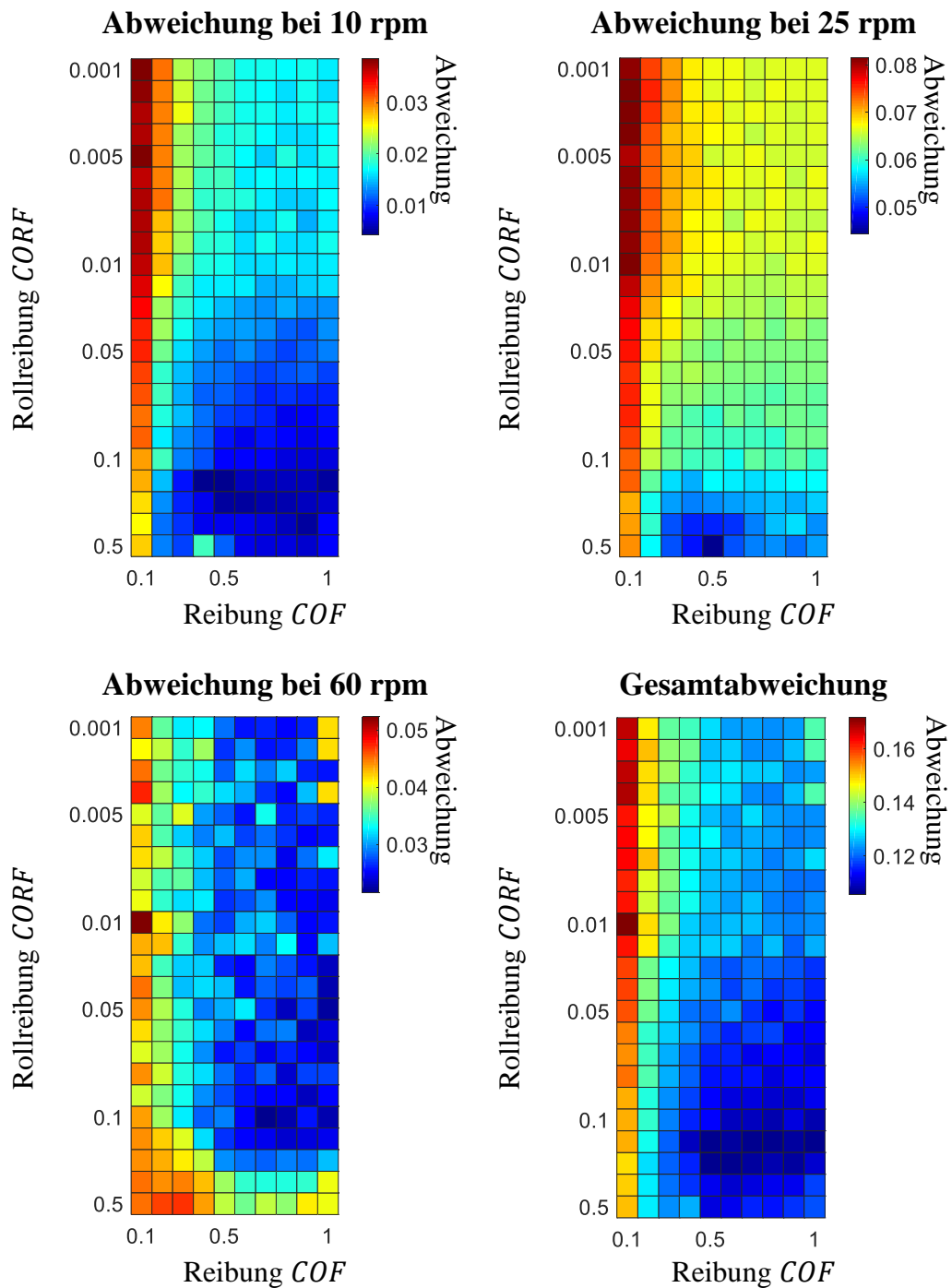


Abbildung 2.6: Abweichung (Gleichung 2.28) der Simulationen zu den Versuchswerten in Abhängigkeit zur Reibung und Rollreibung

Da der Einfluss der Reibung und Rollreibung auf den Fall  $Fr_w < 1$  beschränkt ist, wurden Simulationen mit einer Drehgeschwindigkeit von  $10 \text{ rpm}$ ,  $25 \text{ rpm}$  und  $60 \text{ rpm}$  durchgeführt (Tabelle 2.2). Ein Vergleich der Simulation mit den durchgeführten Experimenten (Abbildung 2.6) zeigt generelle Tendenzen. Eine differenzierte Betrachtung der Resultate weist jedoch leichte Divergenzen der Ergebnisse auf. Ein Optimum ist in jedem der Fälle im Bereich  $0.3 - 0.9$  für die Reibung und im Bereich  $0.2 - 0.5$  für die Rollreibung zu finden.

Während bei Simulationen mit  $10 \text{ rpm}$  und  $25 \text{ rpm}$  ein eindeutiger Trend und Gradienten zu erkennen sind, ist dies bei Simulationen mit  $60 \text{ rpm}$  nicht der Fall. Dies kann auf eine fortschreitende Fluidisierung des Bettes zurückgeführt werden.

Tabelle 2.3: Vergleich der Werte für Reibung und Rollreibung

	<b>E-Modul</b>	<b>Poisson-Zahl</b>	<b>Radius</b>	<b>COR</b>	<b>COF</b>	<b>CORF</b>
<b>Ermittelte Werte</b>	$210 \cdot 10^6 Pa$	0.3	1 mm	0.9	0.5	0.2
<b>Literatur [64-67]</b>	$210 \cdot 10^9 Pa$	0.3	n.n.	0.9	0.15	0.001
<b>Syed et. al. [68]</b>	$2.90 \cdot 10^9 Pa$	0.3	1.5 mm	n.a.	0.5	0.2
<b>Maione et. al. [69]</b>	$210 \cdot 10^9 Pa$	n.a.	12.5 mm	0.8	0.15	0.3
<b>Fatahi et. al. [59]</b>	$168 \cdot 10^6 Pa$	0.38	Var.	0.3	0.5	0.05

Verglichen mit Standardwerten für Stahl, weisen diese eine erhebliche Diskrepanz auf (Tabelle 2.3). Die Reibung und Rollreibung sind abhängig vom Material, aber auch von der Oberflächenbeschaffenheit und von weiteren Faktoren, wie Form und Verunreinigung der Oberfläche. Die hier ermittelten Werte berücksichtigen diese Faktoren.

Während die Literaturwerte stark abweichen, ermittelte Syed et. al. [68] mit seinen Scherversuchen identische Werte für die Stahlkugel/Stahlkugel-Interaktion. Eine größere Abweichung, zu den hier ermittelten Werten, bilden Maione et. al. [69] (Drehtrommel mit  $Fr_W < 1$ ) und Fatahi et. al. [59] (Knelson-Konzentrator). Jedoch haben sie eine Gemeinsamkeit: Alle hier vorgestellten Werte weisen teils größere Abweichungen zu den Literaturwerten auf.

### 2.2.1.2 Untersuchung der Wärmeträger – Wand Reibung und Rollreibung

Da Reibung und Rollreibung sowohl material- als auch oberflächenabhängig sind, ist es erforderlich den Einfluss der Wand – Stahlkugel Interaktion zu betrachten.

In einem weiteren Versuch mit der rotierenden Trommel wurde auf das Erzeugen einer "Partikelwand" verzichtet, sodass der Einfluss der Wand/Stahlkugel-Interaktion mitwirkt. Es wurden stationäre Zustände bei Drehgeschwindigkeiten von 10 rpm bis 120 rpm betrachtet und wie in (Gleichung 2.25) bis (Gleichung 2.28) beschrieben, ausgewertet. Diese Zustände wurden nach dem in (Tabelle 2.4) beschriebenen Programm angesteuert.

Tabelle 2.4: Parameter der Wand/Wärmeträger-Screeningversuche

<b>Partikeleigenschaften:</b>	<b>Wand/</b>		<b>Programm:</b>
	<b>Stahlkugeln</b>	<b>Stahlkugeln</b>	
E-Modul [Pa]	$210 \cdot 10^6$	$210 \cdot 10^6$	3 s mit 10 rpm
Poissonzahl	0.3	0.3	3 s mit 20 rpm
COR	0.9	0.9	3 s mit 45 rpm
COF	0.5	0.3 – 0.8	3 s mit 60 rpm
CORF	0.2	0.1 – 0.7	3 s mit 90 rpm
Dichte [ $kg/m^3$ ]	7878.17		3 s mit 120 rpm
Masse [kg]	0.76463		
Radius [m]	0.001		
dt [s]	0.000006		

Während der Durchführung der Versuche konnte ein Unterschied zu den Versuchen zur Untersuchung der Stahlkugel-Stahlkugel Interaktion festgestellt werden.

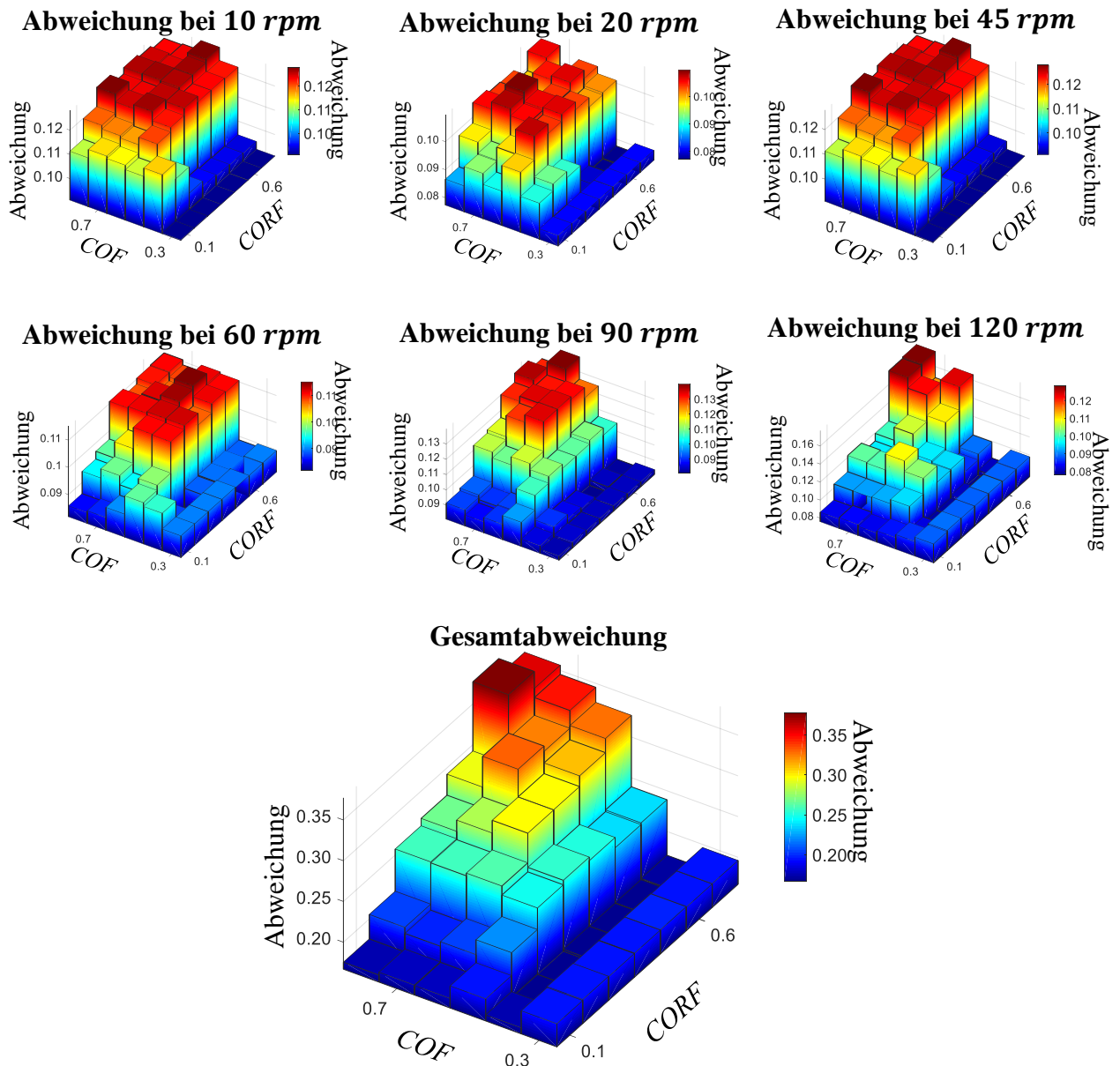


Abbildung 2.7: Abweichungen in Farbe und Höhe der Zustände bzgl. Rotationsgeschwindigkeit

Es konnten Betriebsbedingungen von 120 rpm eingestellt werden, ohne eine Kataraktentwicklung beobachten zu können. Dies steht im Widerspruch zu der in (Tabelle 2.2) ermittelten Werkzeug-Froudezahl. Die Ursache ist ein erheblicher Schlupf zwischen Trommelwand und Partikelbett. Aus diesem Grund muss für höhere Rotationsgeschwindigkeiten  $Fr_p \ll Fr_w$  gelten.

Ein Abgleich dieses Versuches (Abbildung 2.7) mit den Simulationen (Tabelle 2.4) ergibt kein eindeutiges Bild. Jedoch kann den Diagrammen entnommen werden, dass die beste Annäherung für den Wärmeträger/Wand-Kontakt mit einem  $COF < 0.3$  oder  $CORF < 0.2$  zu beobachten ist. Aufgrund der Auswertungsmethode (Gleichungen 2.25 - 2.28) lässt sich die Kataraktentwicklung nur unzureichend auswerten. Da diese aufgrund des Schlupfes zwischen Wand und Partikel eines der wichtigsten Kriterien darstellt, wurden zusätzlich die Partikel-Froudezahl der Betriebsbedingungen für ausgewählte Wand-Stahlkugel  $COF$  und  $CORF$  Paare bestimmt (Tabelle 2.5).

Anhand dessen können bereits einige Effekte identifiziert werden. Ist die Stahlkugel/Stahlkugel-Reibung größer als die Stahlkugel/Wand-Reibung rutscht das Bett an der Wand. Dadurch wird keine Kraft von der Trommel auf das Bett übertragen. Eine Durchmischung des Bettes bleibt dadurch aus.

Ist die Reibung zwischen Schüttung und Wand zu groß, prägt sich ein Katarakt bei geringen Drehgeschwindigkeiten aus (Abbildung 2.2, c, 120 rpm und Abbildung 2.8, 120 rpm). Beim Katarakt sind die Trägheitsmomente größer als die Erdanziehung, sodass die Stahlkugeln sich aus der Schüttung lösen und den Wirbel bilden. Damit ist die Kataraktausprägung mit der Frage verbunden: Wie viel kinetische Energie kann aus der rotierenden Trommel durch Wand/Stahlkugel und Stahlkugel/Stahlkugel-Reibung in die einzelnen Stahlkugeln übertragen werden.

Tabelle 2.5: Minima der Wand/Stahlkugeln für variierende Reibung und Rollreibung

<b>COF</b>	<b>0.5<sup>II</sup></b>	<b>0.6</b>	<b>0.4</b>	<b>0.4</b>	<b>0.4</b>	<b>0.4<sup>III</sup></b>
<b>CORF</b>	<b>0.3</b>	<b>0.4</b>	<b>0.2</b>	<b>0.3</b>	<b>0.5</b>	<b>0.6</b>
<b>10 rpm</b>	0.12	0.10	0.006	0.001	$3.1 \cdot 10^{-5}$	$2.5 \cdot 10^{-5}$
<b>20 rpm</b>	0.26	0.13	0.002	0.000	0.000	0.002
<b>45 rpm</b>	0.22	0.14	0.050	0.006	0.001	0.002
<b>60 rpm</b>	0.15	0.22	0.010	0.002	0.003	0.002
<b>90 rpm</b>	0.27	0.75	0.018	0.020	0.006	0.003
<b>120 rpm</b>	0.66	1.41	0.005	0.017	0.004	0.001

<sup>II</sup> Stahlkugel-Stahlkugel COF und CORF

<sup>III</sup> Sowohl globales als auch lokales Minimum

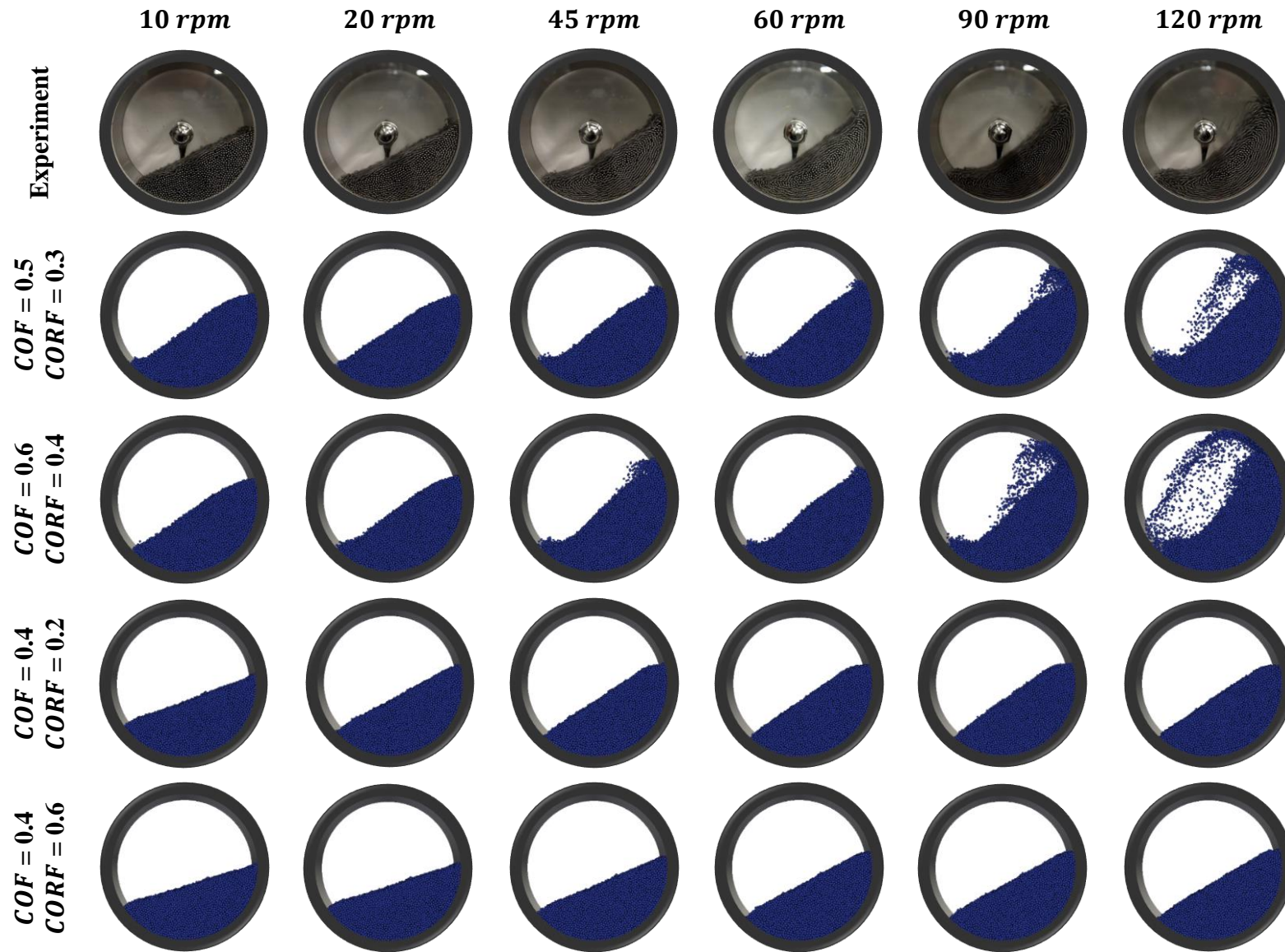


Abbildung 2.8: Ausgewählte Minima für Wand/Wärmeträger  $COF$  und  $CORF$

### 2.2.1.3 Untersuchung Wärmeträger-Wärmeträger/Wand Reibung und Rollreibung im ColdFlow-Modellreaktor – Reaktor

Bisher wurden die Wärmeträger/Wärmeträger bzw. Wärmeträger/Wand Reibung und Rollreibung anhand einer Parameterstudie mit einem Trommelmischer bestimmt. In diesem Zuge konnten sowohl generelle Tendenzen, als auch konkrete Werte bestimmt werden. Bestehend bleibt die Frage der Übertragbarkeit dieser Ergebnisse. Dieses Kapitel beschäftigt sich mit dieser Fragestellung.

Dazu wurden Versuche in einem Modellreaktor durchgeführt, der im Maßstab 1:1 dem Doppelschneckenmischreaktor der PHYTHON-Versuchsanlage entspricht (Abbildung 2.10).

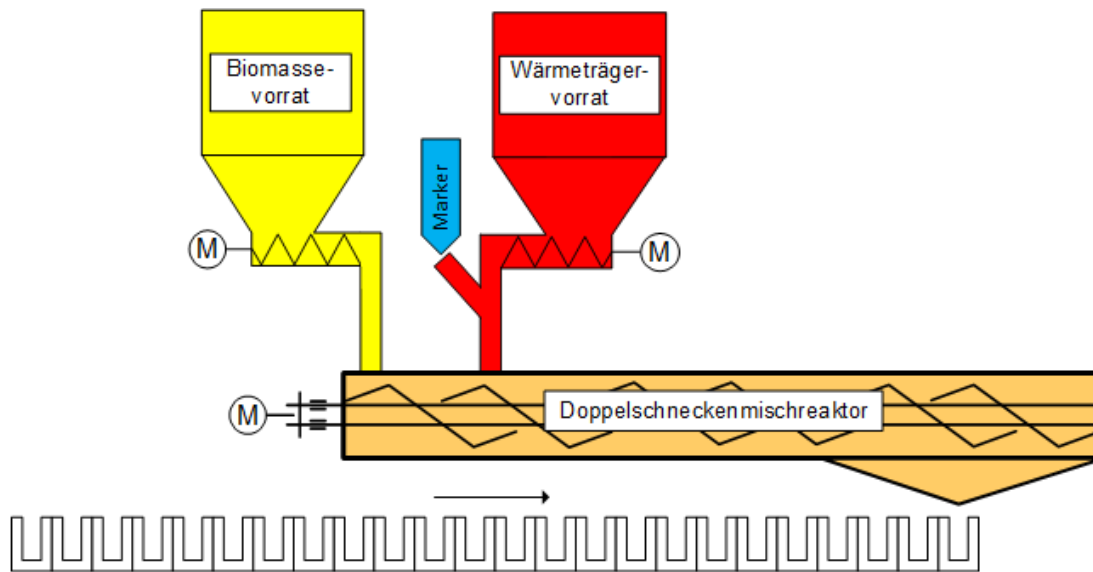


Abbildung 2.9: Versuchsaufbau des ColdFlow-Doppelschneckenmischreaktor zur Bestimmung der Verweilzeit

Biomasse und Wärmeträger befinden sich in zwei Einfülltrichtern. Deren kontinuierliche Dosierung erfolgt über Dosierschnecken, die abhängig von der Drehgeschwindigkeit, die gewünschte Fördermenge in den Reaktor geben. Die Fördermenge von Wärmeträger und Biomasse wurde dabei experimentell ermittelt.

Da Biomasse und Wärmeträger an unterschiedlichen Stellen zugegeben werden, unterteilt sich der Reaktor in Förderbereich und Misch- oder auch Reaktionsbereich. Der Förderbereich erstreckt sich zwischen dem Biomasseeinfallstutzen und dem Wärmeträgereinfallstutzen, an dem Wärmeträger auf die geförderte Biomasse fällt. In der PHYTHON-Versuchsanlage werden ab diesem Punkt die Biomasse und der Wärmeträger vermischt und es setzt die Pyrolyse ein. Der Mischbereich reicht beim ColdFlow-Modellreaktor und bei der PHYTHON-Versuchsanlage bis zur Öffnung am Reaktorboden. Ist dieser fehlerhaft ausgelegt, wird im Pyrolysebetrieb nichtreagierte Biomasse über diese Öffnung ausgeschieden oder verursacht zusätzliche Kosten.

Daher sind zur Auslegung des Reaktors und Beschreibung des Pyrolyseprozesses Kenntnisse über die Verweilzeit im Reaktions- oder auch Mischbereich erforderlich.

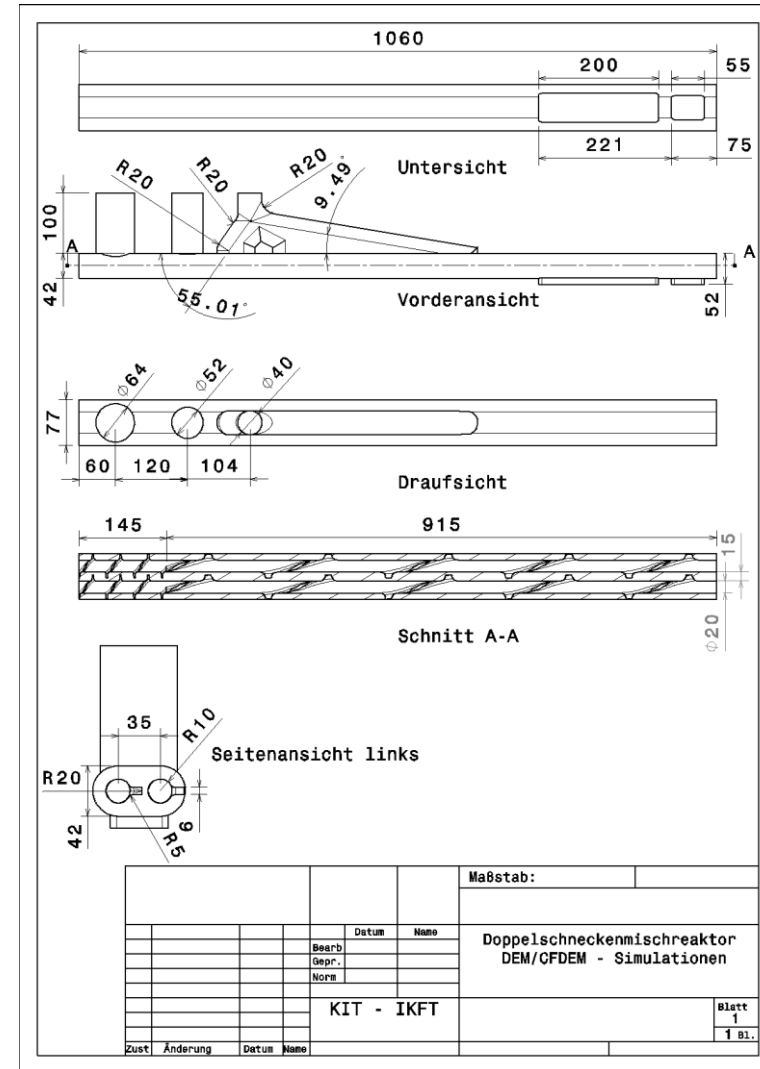
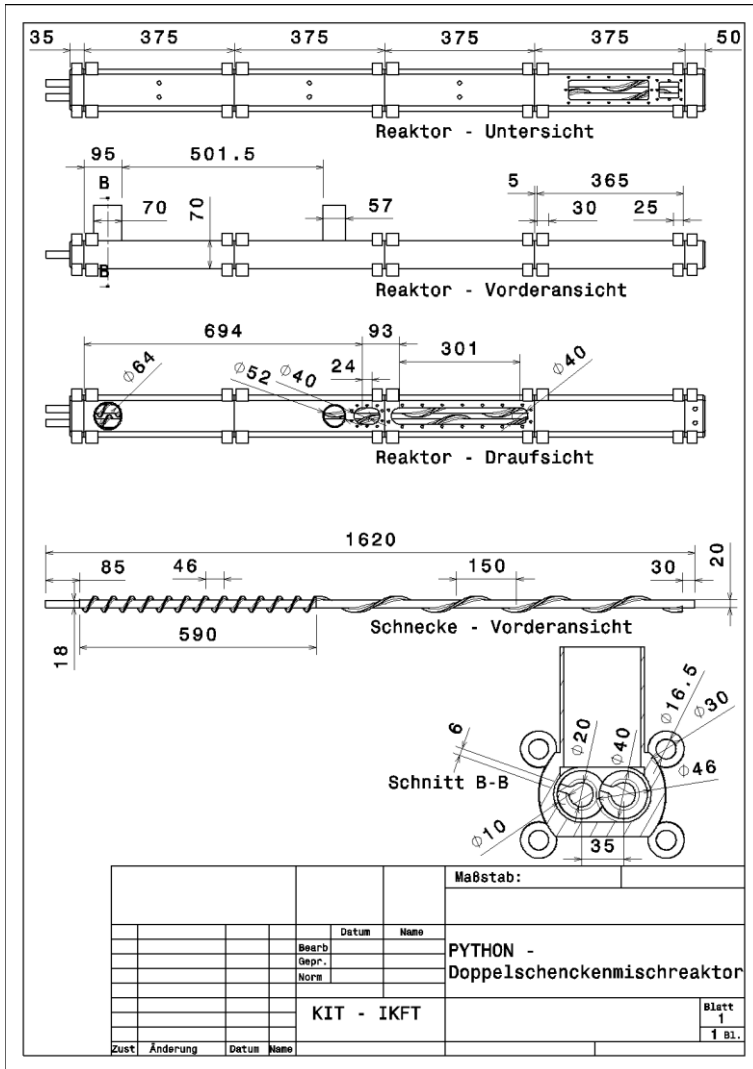


Abbildung 2.10: Skizze des Doppelschneckenmischreaktors. Links: PYTHON und Modell-Doppelschneckenmischreaktors, Rechts: modifizierte Geometrie, wie sie bei der DEM-Simulation zum Einsatz kommt.

Die Verweilzeit wurde in der vorliegenden Arbeit mit Hilfe der Puls-Methode bestimmt. D.h. zum Zeitpunkt  $t = 0$  wird ein Marker (markierte Wärmeträger) an der Wärmeträgeröffnung aufgegeben. Der unter dem Reaktor montierte Schlitten startet und fängt bei konstanter Geschwindigkeit während der Fahrt unter der Reaktoröffnung eine Probenmenge in jedem Schlittensegment auf (Abbildung 2.9). Da sowohl der Startpunkt als auch die Geschwindigkeit des Schlittens bekannt sind, kann jedem Schlittensegment eine Zeit  $t_i$  zugeordnet werden.

Die dazu gehörenden Markerkonzentrationen  $c(t_i)$  werden durch Auswertung der Auffangbehälter ermittelt.

Dies ermöglicht die Bestimmung der mittleren Verweilzeit  $\bar{\tau}$  nach (Gleichung 2.29). Des Weiteren wurde unter Bestimmung der Varianz  $\sigma^2$  (Gleichung 2.30) und der dimensionslosen Varianz  $\sigma_\theta^2$ , die Bodensteinzahl  $Bo$  ermittelt (Gleichung 2.31). Diese ermöglicht Rückschlüsse auf die axiale Dispersion.

$$\bar{\tau} = \frac{\int_0^\infty t \cdot c(t) dt}{\int_0^\infty c(t) dt} \approx \frac{\sum_{i=1}^n t_i \cdot c(t_i) \cdot \Delta t_i}{\sum_{j=1}^n c(t_j) \cdot \Delta t_j} \quad (2.29)$$

$$\sigma^2 = \frac{\int_0^\infty (t_i - \bar{\tau})^2 \cdot c(t) \cdot dt}{\int_0^\infty c(t) \cdot dt} \approx \frac{\sum_{i=1}^n (t_i - \bar{\tau})^2 \cdot c(t_i) \cdot \Delta t_i}{\sum_{j=1}^n c(t_j) \cdot \Delta t_j} \quad (2.30)$$

$$\sigma_\theta^2 = \frac{\sigma^2}{\bar{\tau}^2} = \frac{2}{Bo} + \frac{8}{Bo^2} \quad (2.31)$$

Die Verweilzeit wurde für einen Wärmeträgerfeed von  $920 \text{ kg} \cdot \text{h}^{-1}$  und einem Stahlkugeldurchmesser von 2 mm, für die Betriebszustände mit 1, 2, 3 und 4 Hz bestimmt (Tabelle 2.6). Da lediglich die Wärmeträger/Wärmeträger bzw. Wärmeträger/Wand-Interaktion betrachtet wird, wurde für diese Versuche auf Biomasse verzichtet.

Tabelle 2.6: Verweilzeitverteilung im Doppelschneckenmischreaktor nach Frey [70]

	<b>1 Hz</b>		<b>2 Hz</b>		<b>3 Hz</b>		<b>4 Hz</b>	
	$\bar{\tau}$	$Bo$	$\bar{\tau}$	$Bo$	$\bar{\tau}$	$Bo$	$\bar{\tau}$	$Bo$
	24.53 s	185.95	16.69 s	216.59	10.09 s	63.07	7.61 s	73.97
	23.82 s	181.24	14.76 s	219.15	10.14 s	106.48	7.43 s	44.73
	26.14 s	153.31	15.91 s	183.89	9.91 s	84.92	7.11 s	32.33
	22.98 s	102.59	16.46 s	153.46	-	-	-	-
<b>Mittelwert</b>	<b>24.37 s</b>	<b>155.77</b>	<b>15.96 s</b>	<b>193.27</b>	<b>10.05 s</b>	<b>84.82</b>	<b>7.38 s</b>	<b>50.34</b>
<b>Standardabw.</b>	<b>1.16 s</b>	<b>33.14</b>	<b>0.75 s</b>	<b>26.86</b>	<b>0.10 s</b>	<b>17.72</b>	<b>0.21 s</b>	<b>17.46</b>

Bei den Verweilzeiten nach Frey handelt es sich um Wiederholungen der Versuche von Kornmayer et. al. [1]. Der Fokus von Frey lag auf einer Minimierung der Fehlerquellen durch eine stärkere Automation beim Versuchsablauf. Ein Vergleich der Verweilzeiten ergibt eine sehr gute Übereinstimmung zwischen Frey und Kornmayer. Jedoch unterliegen die Bodensteinzahlen bei Frey sehr viel größeren Schwankungen, sodass die Verwendung dieser zur Validierung der Werte kritisch betrachtet werden muss.

Analog zu dem Versuch wurden DEM-Simulationen unter den gleichen Rahmenbedingungen und mit den Betriebsbedingungen 1 Hz, 2 Hz, 3 Hz und 4 Hz durchgeführt. (Tabelle 2.7). Verwendet wurde jedoch das modifizierte Reaktordesign (Abbildung 2.10 rechts).



Tabelle 2.7: Simulationsrahmenbedingungen des ColdFlow-Modellreaktor

**Partikeleigenschaften:**

	Stahlkugeln	Reaktorwand	Stahlkugeln/ Reaktorwand
E-Modul [Pa]	$210 \cdot 10^6$	$210 \cdot 10^6$	
Poissonzahl	0.3	0.3	
<i>COR</i>	0.9	0.9	0.1
<i>COF</i>	0.3-0.4 (-0.5)	0.2-0.5 (-0.6)	0.8
<i>CORF</i>	0.1-0.3	0.1-0.5	0.99
Dichte [ $kg/m^3$ ]	7878.4	-	
Radius [m]	$10^{-3}$	-	
Massenrate [kg/h]	920		

**Programm**

	<u>1 Hz</u>	<u>2 Hz</u>	<u>3 Hz</u>	<u>4 Hz</u>
Simulationszeit [s]	90	60	60	60
dt [s]	$5 \cdot 10^{-6}$	$5 \cdot 10^{-6}$	$5 \cdot 10^{-6}$	$5 \cdot 10^{-6}$
Auswertungszeit [s]	45	20	20	15

Bei den Simulationen wurden eine Reibung von 0.3 und 0.4 (bei 2 Hz Simulationen zusätzlich 0.5) und eine Rollreibung von 0.1-0.4 für die Stahlkugel/Stahlkugel-Interaktion verwendet. Dies stellt den verifizierten Bereich für die Stahlkugel/Stahlkugel-Interaktion dar (Kapitel 2.2.1.1).

Unter der Annahme, dass für rein sphärische Partikel  $COF, CORF > 0$  und  $CORF \leq COF$  gilt, schwanken die Werte für die Wand-Stahlkugelinteraktion in Abhängigkeit der Werte für die Stahlkugel/Stahlkugel-Interaktion. Dabei gilt:

$$(COF_{W-WT}, CORF_{W-WT}) = (COF_{WT-WT} \pm 0.1, CORF_{WT-WT} \pm 0.1)$$

Somit wurde für jedes Wertepaar der Stahlkugel/Stahlkugel-Interaktion ( $COF_{WT-WT}$  und  $CORF_{WT-WT}$ ), bis zu sechs Simulationen mit veränderten Werten für die Wand/Stahlkugel-Interaktion durchgeführt.

Der Auswertung (Tabelle 2.8) kann entnommen werden, dass die Wand/Stahlkugel Reibung einen signifikanten und die Rollreibung keinen erkennbaren Einfluss auf die Verweilzeit hat. Des Weiteren kann entnommen werden, dass die Stahlkugel-Stahlkugel Reibung und die Rollreibung einen erkennbaren Einfluss auf das Verweilzeitverhalten im Reaktor ausüben.

Tabelle 2.8: Bestimmung der Verweilzeit im DEM-Modellreaktor

COF		Stahlkugeln - Stahlkugeln																				
		0.3		0.3		0.3		0.4		0.4		0.5		0.5		0.5						
		CORF		0.1		0.2		0.3		0.1		0.2		0.3		0.1		0.2		0.3		
				$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		$\bar{t} \cdot s^{-1} Bo$		
Wand - Stahlkugeln	1 Hz	0.2	0.1	13.7	169.0	13.2	171.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
		0.2	0.2	13.9	184.2	13.4	156.5	12.8	168.0	-	-	-	-	-	-	-	-	-	-	-	-	-
		0.2	0.3	-	-	13.4	164.1	12.9	144.4	-	-	-	-	-	-	-	-	-	-	-	-	-
		0.2	0.4	-	-	-	-	13.0	185.3	-	-	-	-	-	-	-	-	-	-	-	-	-
		0.3	0.1	20.6	163.0	21.2	180.2	-	-	21.0	172.5	21.2	177.2	-	-	-	-	-	-	-	-	-
		0.3	0.2	22.0	229.4	21.5	217.8	22.2	199.7	21.8	227.9	22.6	188.3	21.8	247.2	-	-	-	-	-	-	-
		0.3	0.3	-	-	23.7	209.6	22.2	278.2	-	-	22.6	239.0	22.5	225.3	-	-	-	-	-	-	-
		0.3	0.4	-	-	-	-	22.4	261.1	-	-	-	-	23.7	217.5	-	-	-	-	-	-	-
		0.4	0.1	24.5	247.9	26.4	274.4	-	-	26.0	301.7	26.4	301.5	-	-	-	-	-	-	-	-	-
		0.4	0.2	28.0	330.2	28.7	337.5	28.6	319.8	-	-	28.6	331.8	28.5	305.6	-	-	-	-	-	-	-
	0.4	0.3	-	-	-	-	29.5	354.2	-	-	29.4	387.6	29.0	372.5	-	-	-	-	-	-	-	
	0.4	0.4	-	-	-	-	30.0	379.7	-	-	-	-	29.4	416.3	-	-	-	-	-	-	-	
	0.5	0.1	-	-	-	-	-	-	27.3	273.6	28.3	309.5	-	-	-	-	-	-	-	-	-	
	0.5	0.2	-	-	-	-	-	-	31.0	332.4	31.4	414.9	31.2	462.9	-	-	-	-	-	-	-	
	0.5	0.3	-	-	-	-	-	-	-	-	32.7	466.9	33.1	477.9	-	-	-	-	-	-	-	
	0.5	0.4	-	-	-	-	-	-	-	-	-	-	34.0	521.7	-	-	-	-	-	-	-	
	2 Hz	0.2	0.1	9.2	81.3	8.9	74.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
		0.2	0.2	9.3	81.4	9.2	80.2	9.0	78.6	-	-	-	-	-	-	-	-	-	-	-	-	-
		0.2	0.3	-	-	9.2	80.5	9.0	78.9	-	-	-	-	-	-	-	-	-	-	-	-	-
		0.2	0.4	-	-	-	-	8.9	82.3	-	-	-	-	-	-	-	-	-	-	-	-	-
0.3		0.1	11.5	122.8	11.3	126.2	-	-	11.5	117.6	11.3	128.4	-	-	-	-	-	-	-	-	-	
0.3		0.2	12.1	127.6	11.6	134.0	11.4	138.4	11.9	130.4	11.4	130.0	11.0	154.0	-	-	-	-	-	-	-	
0.3		0.3	-	-	12.0	128.1	11.6	137.2	-	-	11.6	131.3	11.2	138.7	-	-	-	-	-	-	-	
0.3		0.4	-	-	-	-	11.5	148.5	-	-	-	-	11.3	139.8	-	-	-	-	-	-	-	
0.4		0.1	12.7	131.3	12.6	132.9	-	-	12.6	135.5	12.5	142.1	-	-	12.7	140.7	12.4	146.5	-	-	-	
0.4		0.2	13.5	149.6	13.2	158.3	13.0	143.6	13.3	162.1	13.1	148.2	12.7	144.2	13.4	145.0	12.8	168.2	12.6	149.7	-	
0.4		0.3	-	-	13.5	152.8	13.4	141.1	-	-	13.3	151.6	12.9	152.7	-	-	13.1	151.0	12.7	142.9	-	
0.4		0.4	-	-	-	-	13.6	136.9	-	-	-	-	13.1	147.4	-	-	-	-	12.7	156.9	-	
0.5		0.1	-	-	-	-	-	-	13.0	146.0	12.9	147.4	-	-	13.1	140.3	13.0	140.5	-	-	-	
0.5		0.2	-	-	-	-	-	-	14.2	159.2	13.8	158.8	13.7	144.6	14.0	157.7	13.8	146.9	13.4	154.2	-	
0.5		0.3	-	-	-	-	-	-	-	-	14.2	166.5	14.1	163.1	-	-	-	-	13.8	158.9	-	
0.5		0.4	-	-	-	-	-	-	-	-	-	-	14.1	178.6	-	-	-	-	13.9	183.4	-	
0.6		0.2	-	-	-	-	-	-	-	-	-	-	-	-	14.7	161.7	14.5	166.6	14.2	161.8	-	
0.6		0.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	15.2	166.3	14.8	159.3	-	
0.6	0.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	15.3	159.3	-		
3 Hz	0.2	0.1	5.2	46.0	5.0	51.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-		
	0.2	0.2	5.5	46.9	5.3	50.1	5.2	50.6	-	-	-	-	-	-	-	-	-	-	-	-	-	
	0.2	0.3	-	-	5.5	56.9	5.3	49.5	-	-	-	-	-	-	-	-	-	-	-	-	-	
	0.2	0.4	-	-	-	-	5.4	54.8	-	-	-	-	-	-	-	-	-	-	-	-	-	
	0.3	0.1	8.5	80.1	8.7	79.6	-	-	8.6	81.9	8.5	82.3	-	-	-	-	-	-	-	-	-	
	0.3	0.2	9.3	92.4	9.2	88.0	9.1	87.1	9.2	86.2	9.2	87.7	8.9	89.4	-	-	-	-	-	-	-	
	0.3	0.3	-	-	9.3	102.5	9.3	95.8	-	-	9.2	91.6	9.1	93.5	-	-	-	-	-	-	-	
	0.3	0.4	-	-	-	-	9.5	93.0	-	-	-	-	9.2	88.5	-	-	-	-	-	-	-	
	0.4	0.1	9.3	97.5	9.7	104.6	-	-	9.3	106.1	9.6	102.0	-	-	-	-	-	-	-	-	-	
	0.4	0.2	10.5	111.2	10.5	118.5	10.6	120.4	10.5	115.9	10.3	114.1	-	-	-	-	-	-	-	-	-	
	0.4	0.3	-	-	10.9	137.7	10.9	119.6	-	-	10.8	128.4	10.5	123.8	-	-	-	-	-	-	-	
	0.4	0.4	-	-	-	-	11.0	123.1	-	-	-	-	10.7	128.4	-	-	-	-	-	-	-	
	0.5	0.1	-	-	-	-	-	-	9.6	106.6	9.8	112.7	-	-	-	-	-	-	-	-	-	
	0.5	0.2	-	-	-	-	-	-	10.9	123.7	10.9	122.3	10.9	124.9	-	-	-	-	-	-	-	
	0.5	0.3	-	-	-	-	-	-	-	-	11.4	142.1	11.2	136.1	-	-	-	-	-	-	-	
	0.5	0.4	-	-	-	-	-	-	-	-	-	-	11.5	128.5	-	-	-	-	-	-	-	
4 Hz	0.2	0.1	3.6	38.9	3.6	37.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-		
	0.2	0.2	3.8	38	3.6	40.8	3.6	38.3	-	-	-	-	-	-	-	-	-	-	-	-	-	
	0.2	0.3	-	-	3.7	38.1	3.7	37.7	-	-	-	-	-	-	-	-	-	-	-	-	-	
	0.2	0.4	-	-	-	-	3.6	34.8	-	-	-	-	-	-	-	-	-	-	-	-	-	
	0.3	0.1	5.8	64.2	6.3	72	-	-	6.1	66.8	6.5	79.9	-	-	-	-	-	-	-	-	-	
	0.3	0.2	7.3	81.8	7.5	83.8	7.6	82.2	7.4	83	7.4	83.3	7.4	86.9	-	-	-	-	-	-	-	
	0.3	0.3	-	-	7.7	82.6	7.8	82	-	-	7.7	89.6	7.7	84	-	-	-	-	-	-	-	
	0.3	0.4	-	-	-	-	7.8	87.1	-	-	-	-	7.7	90.8	-	-	-	-	-	-	-	
	0.4	0.1	7.2	85.1	7.7	90.5	-	-	7.4	89.3	7.8	86	-	-	-	-	-	-	-	-	-	
	0.4	0.2	8.6	98.2	8.7	99.8	8.8	102.1	8.5	106.8	8.6	104.4	8.7	97.9	-	-	-	-	-	-	-	
	0.4	0.3	-	-	9.2	117.1	9.3	111.1	-	-	9.2	108.8	8.9	115.5	-	-	-	-	-	-	-	
	0.4	0.4	-	-	-	-	9.3	117.4	-	-	-	-	9.3	112.1	-	-	-	-	-	-	-	
	0.5	0.1	-	-	-	-	-	-	7.8	93.2	8.1	99.6	-	-	-	-	-	-	-	-	-	
	0.5	0.2	-	-	-	-	-	-	9	123	9.2	113.6	9.1	115.5	-	-	-	-	-	-	-	
	0.5	0.3	-	-	-	-	-	-	-	-	9.9	130.9	9.8	126	-	-	-	-	-	-	-	
	0.5	0.4	-	-	-	-	-	-	-	-	-	-	10.1	140.5	-	-	-	-	-	-	-	

Simulationen mit 1, 3 und 4 Hz bieten eine gute Annäherung für eine Wand/Wärmeträger Reibung von 0.4 und eine Rollreibung von 0.1-0.4. Simulationen mit 2 Hz bieten erst ab einer Reibung von 0.6 eine gute Approximation der Verweilzeit. Eine mögliche Ursache kann z.B. ein nicht stationärer Zustand zum Beginn der Ermittlung der mittleren Verweilzeit sein. Kornmayer konnte bei Versuchen mit 2 Hz vergleichbare Verweilzeiten experimentell ermitteln. Jedoch belegen die Daten von Kornmayer und Frey eine erhebliche Streuungen in diesem Drehzahlbereich.

Da der Reaktor als Strömungsrohr dargestellt werden kann, wurde zusätzlich die hydrodynamische Verweilzeit  $\hat{\tau}$  bestimmt. Diese bildet sich aus dem gesamten Wärmeträgervolumen  $V_{WT}$  im Reaktor und den eingehenden Wärmeträgervolumenstrom  $\dot{V}_{WT}$ . Durch Kürzen der Wärmeträgerdichte  $\rho_{WT}$  und der Einzelpartikelmasse  $m_{St}$ , kann die hydrodynamische Verweilzeit aus der Anzahl der Wärmeträgerpartikel  $n_{WT}$  im Reaktor und dem Wärmeträgerpartikelstrom  $\dot{n}_{WT}$  bestimmt werden [71].

$$\hat{\tau} = \frac{V_{WT}}{\dot{V}_{WT}} = \frac{\frac{m_{WT}}{\rho_{WT}}}{\frac{\dot{m}_{WT}}{\rho_{WT}}} = \frac{n_{WT} \cdot m_{St}}{\dot{n}_{WT} \cdot m_{St}} = \frac{n_{WT}}{\dot{n}_{WT}} \quad (2.32)$$

Die hydrodynamische Verweilzeit und die mittlere Verweilzeit wurde für das Werte-Quartett ( $COF=0.4/CORF=0.1$  für Wärmeträger/Wärmeträger und  $COF=0.4/CORF=0.1$  für die Wärmeträger/Wand-Interaktionen) welches die gemessene mittlere Verweilzeit am besten beschreibt in Abbildung 2.11 gegenübergestellt.

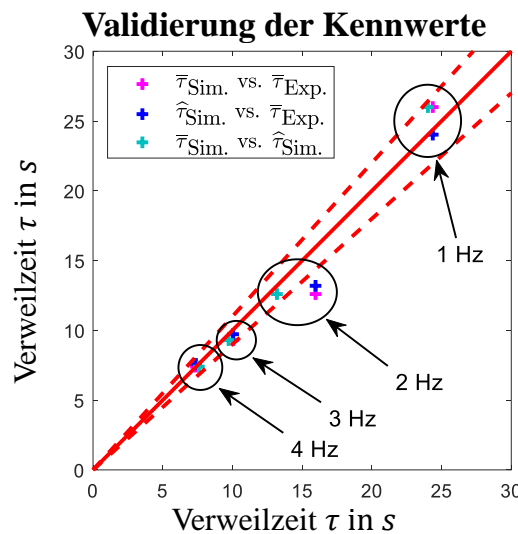


Abbildung 2.11: Vergleich der mittleren ( $\bar{\tau}$ ) und hydrodynamischen ( $\hat{\tau}$ ) Verweilzeit der Simulationen ( $\tau_{Sim.}$ ) mit den Experimentell ermittelten Werten ( $\tau_{Exp.}$ ).  
Jeweils erster Eintrag der Legende stellt den X-Wert dar.

Auch hier ist erkennbar, dass die Verweilzeiten bei 2 Hz voneinander abweichen. Darüber hinaus wird deutlich, dass die mittlere Verweilzeit und die hydrodynamische Verweilzeit bei den 2 Hz Simulationen weitestgehend übereinstimmen. Dies kann als Beleg gewertet werden, dass zum Zeitpunkt der Auswertung bei 2 Hz bereits ein stationärer Zustand erreicht ist und die Abweichung zwischen Versuch und Simulation auf die Streuung bei den Versuchen zurück zu führen ist.

## 2.3 Charakterisierung von Weizenstroh

Während Stahlkugeln für DEM-Simulationen eine günstige Form haben, verhält es sich bei Weizenstrohpartikeln anders. Aufgrund der Partikelform unbekannter und teils nicht konstanter Materialeigenschaften [72], stellt Weizenstroh in diesem Bereich eine Herausforderung dar. Daher mussten im Rahmen dieser Arbeit einige Annahmen getroffen werden, bevor eine Validierung der weizenstrohabhängigen Reibung und Rollreibung erfolgen kann.

### 2.3.1 Partikeleigenschaften

Weizenstroh wird als Ballen angeliefert. Für die weitere Verwendung ist es erforderlich, diese mittels Schredder grob zu zerkleinern ( $< 50 \text{ mm}$ ). Im Anschluss erfolgt eine Feinzerkleinerung ( $< 5 \text{ mm}$ ) mittels Schneidmühle.

*Tabelle 2.9: Anlagentypen der Zerkleinerungslinie*

Firma:	Neue Herbold Maschinen und Anlagenbau GmbH
Schredder-Typ:	HZR 1300
Schneidmühle-Typ:	LM 450-1000

Die verwendete Zerkleinerungslinie (Tabelle 2.9) ist Teil des bioliq®-Projektes. Deren Produkte werden sowohl im Rahmen des bioliq®-Projektes als auch in der PHYTON Versuchsanlage eingesetzt, sodass eine Charakterisierung der Weizenstrohpartikel für beide Versuchsanlagen anwendbar ist.

### 2.3.2 Untersuchung der Weizenstroh-Partikelform

Aufgrund der Biomassepartikelform (lange Stäbchen und Plättchen, Abbildung 2.12) lieferte eine klassische Fraktionierung durch Siebanalyse keine zuverlässigen Aussagen bzgl. der Form, Größe und Anzahl der Partikel. Eine manuelle Auswertung der Partikelform stellt wegen ihrer geringen Größe und der hohen Partikelzahl einen erheblichen Aufwand dar. Eine effektive Möglichkeit bietet die Bildverarbeitung, deren Methoden, die Objekterkennung und deren Auswertung ermöglichen [73].



*Abbildung 2.12: Auswahl der untersuchten Weizenstrohpartikel. Zweifach vergrößert.*

Dazu wurden  $7.922 \text{ g}$  Weizenstroh (etwa  $85 \text{ ml}$ ) auf eine durchsichtige, selbstklebende Folie aufgetragen. Dabei wurde darauf geachtet, dass sich die einzelnen Partikel nicht berühren. Zur

weiteren Verwendung wurde die partikelbeladene Folie auf Blätter der Größe A3 aufgeklebt und eingescannt.

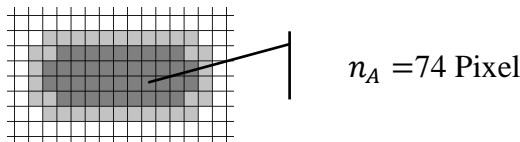
Da die Pixelbreite  $d_{\text{pix}}$  (600 dpi) bekannt ist, kann jedem Pixel eine lokale Position zugeordnet werden. So können durch die Anwendung der Bildverarbeitung die Partikel ausgewertet werden.

Bevor die Auswertung beginnen kann, müssen die Scans behandelt werden. Sodass jedes Partikel einzeln verfügbar ist. Dazu wurden die Schritte:

1. Lese Bild
2. Konvertiere Bild in S/W
3. Suche zusammenhängende Objekte

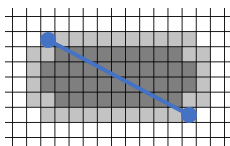
durchlaufen. Für ein solch identifiziertes Partikel wurde die Partikel Länge  $L$ , Breite  $B$  und Fläche  $A$  bestimmt. Dabei gilt:

1. Bestimme die Anzahl der Pixel  $n_A$  eines Partikels



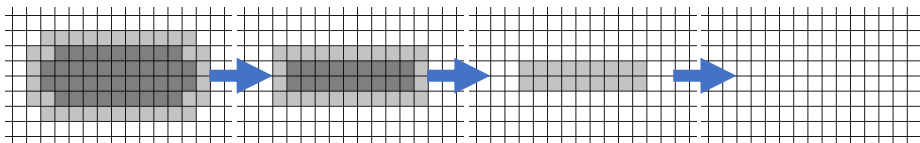
$$A = n_A \cdot A_{\text{pix}}$$

2. Bestimme den maximalen Abstand  $L$  zweier Pixel eines Partikels



$$L = \max_{\text{Pix}_1, \text{Pix}_2} \sqrt{(X_{\text{Pix}_1} - X_{\text{Pix}_2})^2 + (Y_{\text{Pix}_1} - Y_{\text{Pix}_2})^2} + d_{\text{pix}}$$

3. Reduziere den Rand eines Partikels um eine Pixelbreite. Bestimme die Anzahl  $n_B$  der erforderlichen Schritte.



$$B = 2 \cdot n_B \cdot d_{\text{pix}} - d_{n_B}$$

Bei der Berechnung der Partikelbreite, wird der Partikelrand soweit abgetragen, bis das Partikel nicht mehr existiert ist. Dabei muss im letzten Schritt darauf geachtet werden, ob der Partikelrand aus einer Lage oder aus zwei besteht. Da dies einen Unterschied in der resultierenden Breite darstellt, wurde der Korrekturterm  $d_{n_B}$  verwendet.

$$d_{n_B} = \begin{cases} d_{\text{pix}}, & \text{fallt } n_B - 1 \text{ nur eine Pixelreihe} \\ 0, & \text{sonst} \end{cases}$$

Eine Überlappung von zwei oder mehreren Partikeln kann nicht vollständig ausgeschlossen werden. Daher wurden die ermittelten Partikel Daten durch mehrere Bedingungen gefiltert.

Somit wurden alle Partikel, welche die Bedingungen:

$$\begin{aligned}L &> 3 \text{ cm} \\A &> L \cdot B \\U &> 2 \cdot L + 2 \cdot B\end{aligned}$$

Erfüllen, herausgefiltert und in den weiteren Untersuchungen nicht mehr berücksichtigt. Auf diese Weise konnten von 93612 erkannten Partikel, 413 Überlappungen ausgeschlossen werden. Diese Partikel konnten jedoch nicht in einer Neuberechnung der Masse berücksichtigt werden. Aufgrund der geringen Partikelgewichte und der geringen Anzahl, wird dieser Fehler durch die Pixel-Rasterung beim Scannen überlagert.

Aufgetragen in einem dreidimensionalen Histogramm (Abbildung 2.13) mit den Achsen Länge  $L$  und Breite  $B$  ist zu erkennen, dass die meisten Partikel sehr klein sind. Die gelegentlichen Ausreißer in der Monotonie sind auf die ganzzahlige Auswertung (Pixel) und die reelle Diskretisierung zurück zu führen.

### Formverteilung der Biomassepartikel

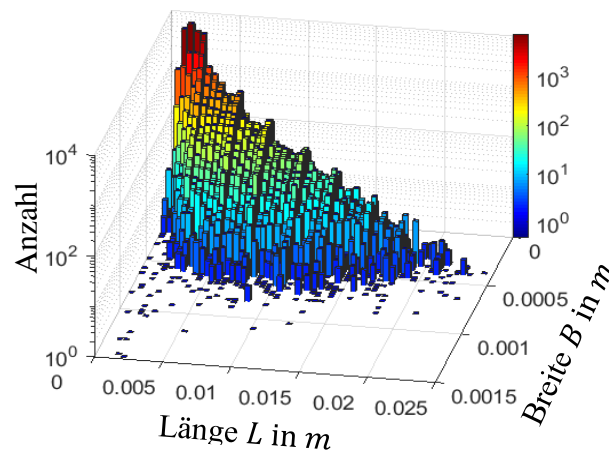


Abbildung 2.13: Histogramm in Abhängigkeit der Formparameter Länge und Breite

Getrennt betrachtet (Abbildung 2.14) ist zu erkennen, dass die Längen- und Breitenverteilung einer vergleichbaren Form- und Größenordnung entspricht. Jedoch ist die Spanne der Längenverteilung um den Faktor vier größer als die der Breitenverteilung. Das Längen/Breiten-Verhältnis hat einen Häufungspunkt um 2. Dabei ist zu erkennen, dass dieser überwiegend auf Staubpartikel zurückzuführen ist (Abbildung 2.13).

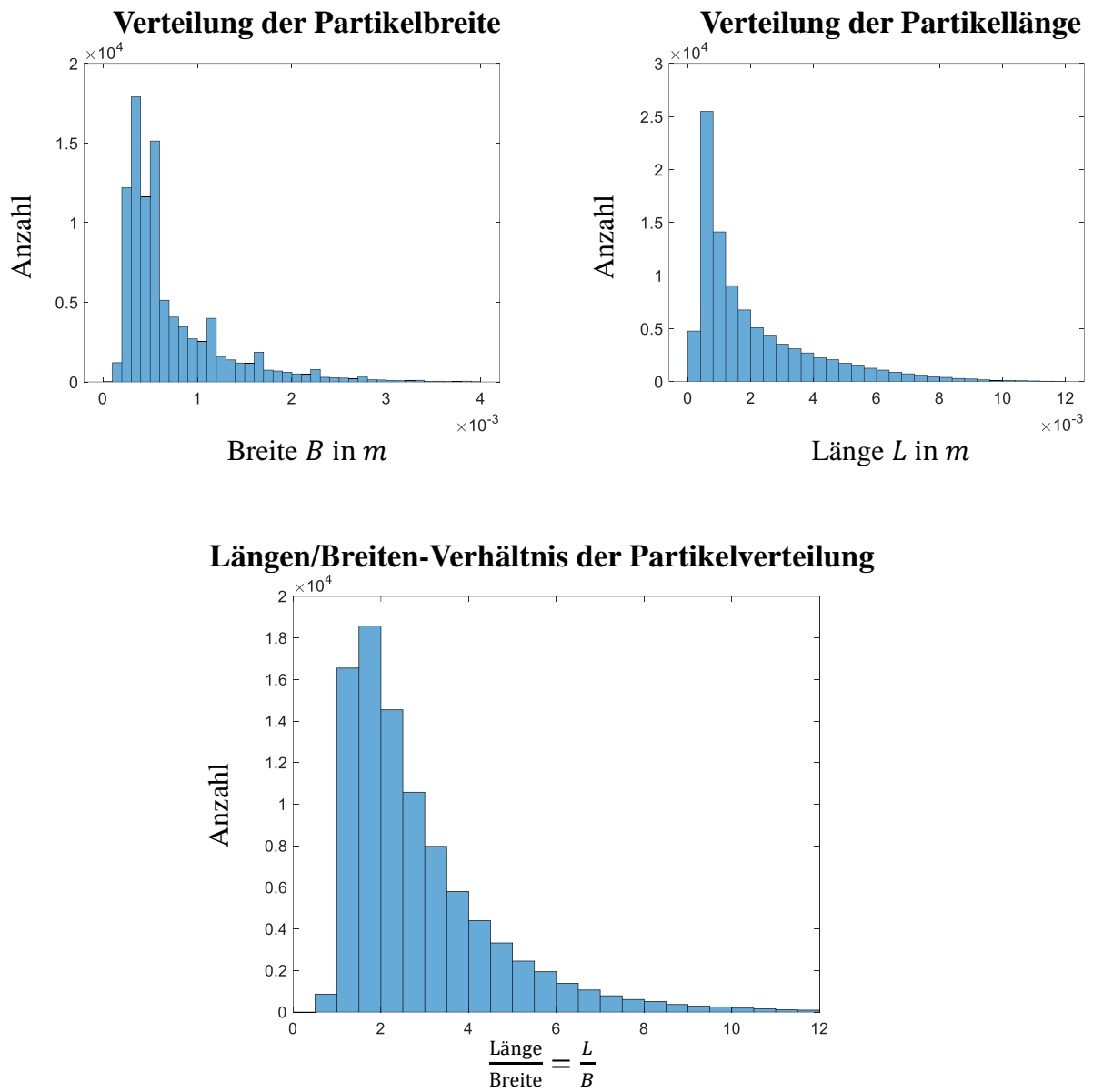


Abbildung 2.14: Partikelverteilung in Abhängigkeit der Partikelbreite (oben links), Partikellänge (oben rechts), und des Verhältnisses aus beiden (unten)

### 2.3.2.1 Bestimmung der Weizenstroh-Partikelhöhe

In einem ersten Schritt wurde die Höhe von 300 Biomassepartikel bestimmt. Dies erfolgte durch eine manuelle Messung, unter Zuhilfenahme eines digitalen Messschiebers mit zwei Nachkommastellen, für eine Genauigkeit von bis zu  $10^{-5} m$ .

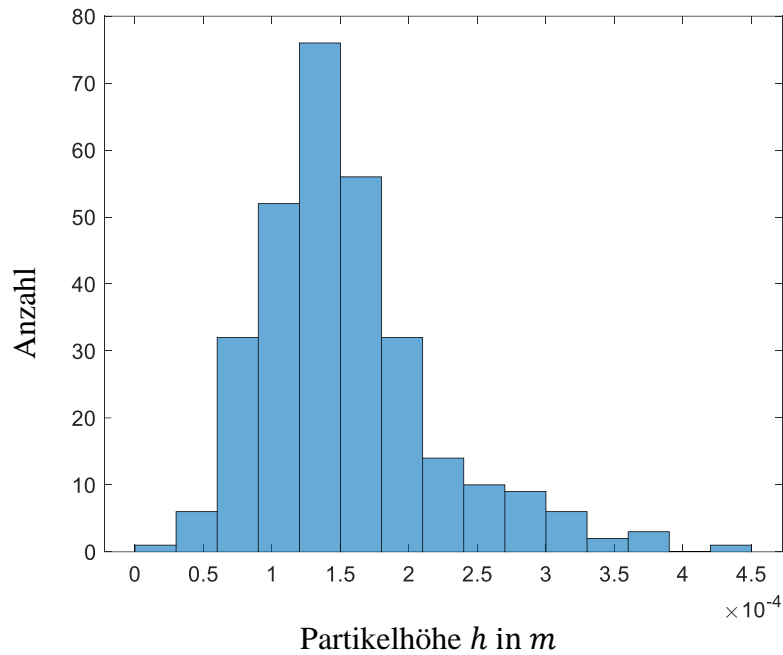


Abbildung 2.15: Histogramm der Partikelhöhenverteilung von Weizenstrohpartikel

Die Messwerte  $h_i$  wurden in einem Histogramm (Abbildung 2.15) aufgetragen. Dieses stellt eine Normalverteilung, mit Erwartungswert  $h$  (Gleichung 2.33) und Standardabweichung  $\sigma_h$  (Gleichung 2.34) dar.

$$h = \frac{1}{300} \sum_{i=1}^{300} h_i = 1.4995 \cdot 10^{-4} \text{ m} \approx 0.15 \text{ mm} \quad (2.33)$$

$$\sigma_h = \sqrt{\frac{1}{300 - 1} \sum_{i=1}^{300} (h_i - h)^2} = 6.563 \cdot 10^{-5} \text{ m} \approx 6.6 \cdot 10^{-2} \text{ mm} \quad (2.34)$$

Im Wesentlichen ist die Standardabweichung  $\sigma_h$  auf Spreu, einlagige Weizenstrohhalme und zweilagige Weizenstrohhalme zurück zu führen. Während das Spreu den Anfang der Höhenverteilung ausmacht, bilden die doppelwandigen Weizenstrohhalme das Ende der Höhenverteilung.

### 2.3.2.2 Vergleich der Partikelmodelle

Bei der DEM-Simulation wird die Partikelbewegung von sphärischen Partikeln bestimmt. Abweichende Formen können über die Partikeleigenschaften wie Reibung und Rollreibung kompensiert werden. Dies ist jedoch nur anwendbar, solange dieser Effekt die Partikelbewegung nicht dominiert. Wie (Abbildung 2.13) und (Abbildung 2.14) zeigen, sind die Partikel um den Faktor 2-12 länger. Unter Berücksichtigung der Dicke der Partikel führt die Form dazu, dass ein Rollen von Weizenstroh nahezu ausgeschlossen ist. Des Weiteren begünstigt die Partikelform, sowie die hohe Biomasse-Biomasse Reibung die Verklumpung, sodass geprüft werden muss, inwiefern Weizenstroh durch Sphären dargestellt werden kann. Dieses Kapitel betrachtet unterschiedliche Körpermodelle. Dabei werden die ermittelten Längen  $L$  und Breiten  $B$  zur Volumenberechnung dieser Körper genutzt. Dem gegenüber stehen Volumina, die nach der klassischen Säulengleichung, d.h.  $V = G \cdot h$  berechnet



werden. Bei der Auswertung der Partikelform, wurde neben der Partikellänge und Breite auch die Fläche  $A_i$  (Gleichung 2.35) bestimmt.

$$\begin{aligned} h_i &= \min(h, B_i) \\ V_i &= A_i \cdot h_i \end{aligned} \tag{2.35}$$

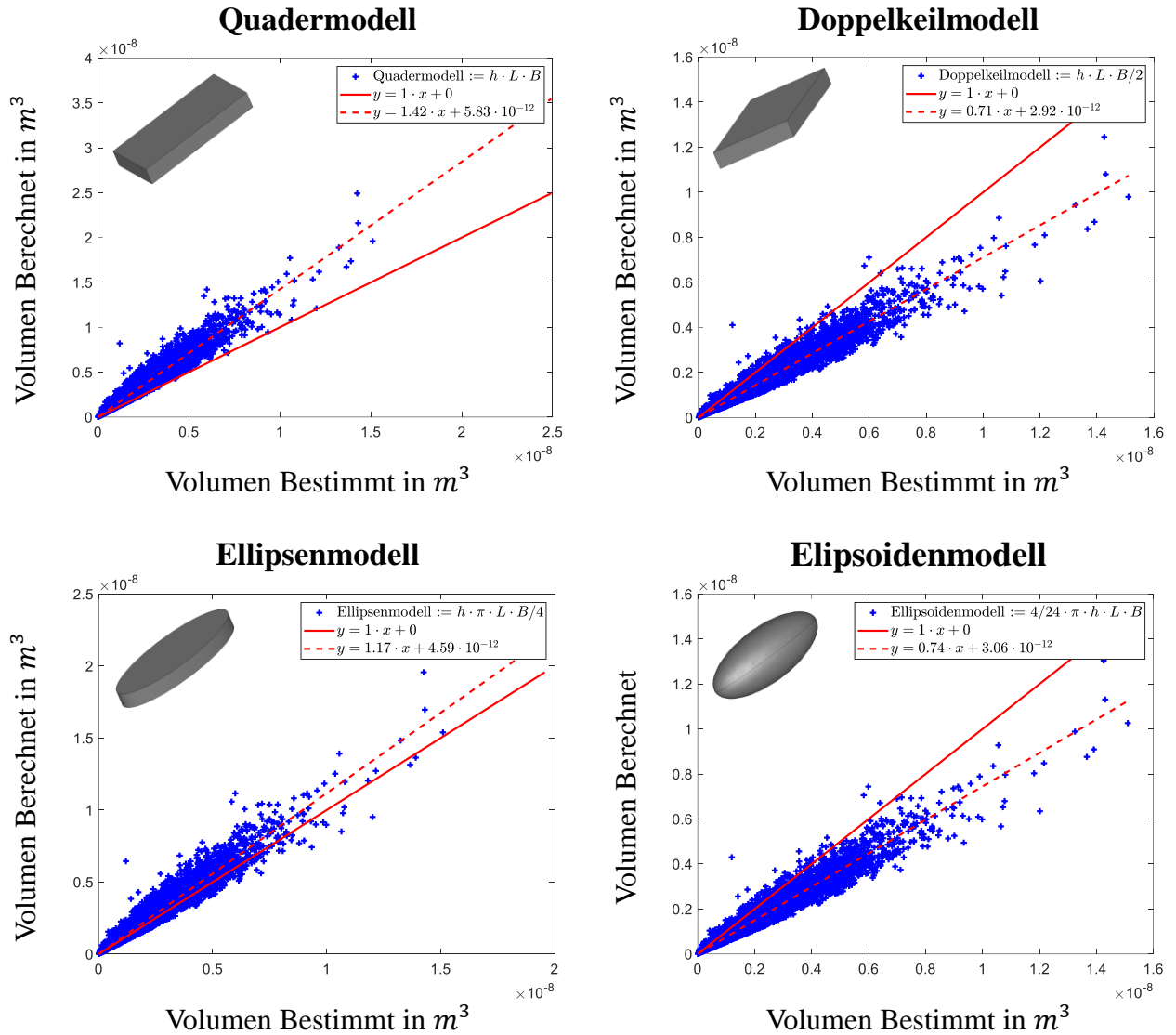


Abbildung 2.16: Vergleich der Soll- mit den Ist-Werten (Gleichung 2.35). Quadermodell (oben-links), Doppelkeilmmodell (oben-rechts), Ellipsenmodell (unten-links) und Ellipsoidenmodell (unten-rechts)

Tabelle 2.10: Regressionsgerade und Standardabweichung der Modelle

	Regressionsgerade $y(x) =$	Standardabweichung
Quadermodell	$1.42 \cdot x + 5.83 \cdot 10^{-12}$	$9.54 \cdot 10^{-11}$
Doppelkeilmmodell	$0.71 \cdot x + 2.92 \cdot 10^{-12}$	$4.77 \cdot 10^{-11}$
Ellipsenmodell	$1.17 \cdot x + 4.59 \cdot 10^{-12}$	$5.00 \cdot 10^{-11}$
Ellipsoidenmodell	$0.74 \cdot x + 3.06 \cdot 10^{-12}$	$7.50 \cdot 10^{-11}$

Ein Vergleich (Abbildung 2.16) dieser Modelle zeigt, dass bis auf einen konstanten Faktor (Tabelle 2.10) alle Modelle das Volumen sehr gut beschreiben. Da zur Berechnung der Volumina dieselbe Höhe  $h_i$  verwendet wurde, ist die Übereinstimmung auf die Grundflächen zurückzuführen.

Eine Ausnahme bildet das Ellipsoidenmodell. Es zeigt nur geringfügig bessere Resultate als das Doppelkeilmodell, jedoch folgt es nicht dem klassischen Säulenprinzip. Persönlichen Beobachtungen folgend, ist die Höhe eines Partikels weitestgehend konstant. Dennoch ist der Partikelrand aufgrund der Zerkleinerung ein wenig dünner. Sodass für das Ellipsoidenmodell ein minimal besseres Ergebnis zu erwarten ist.

Jedes der getesteten Modelle bildet mit dem Korrekturfaktor das Weizenstroh-Partikelvolumen ähnlich gut ab. Daher wurde für die weitere Untersuchung das Quadermodell gewählt. Dieses zeichnet sich durch die einfache Umsetzung aus. Der Korrekturfaktor fließt dabei in die Breite  $B$  ein.

### 2.3.2.3 Weizenstroh Partikeldichte

Die im vorangegangenen Kapitel bestimmten Partikelvolumina (Gleichung 2.35) können nun genutzt werden, um weitere wichtige Erkenntnisse zu erzielen. Ein für die Simulation wichtiger Wert ist die Dichte von Weizenstroh. Da Weizenstroh als Ballen transportiert und gelagert wird, beziehen sich die Literaturwerte auf die Schüttdichte des Weizenstrohs. Je nach Pressung schwanken diese im Bereich von  $90 - 150 \text{ kg} \cdot \text{m}^{-3}$  [74, 75]. Werte für die scheinbare Dichte, d.h. die Dichte eines Partikels inklusive der Meso-, Makro- und Mikroporen, können nur schwer bestimmt werden. Unter Verwendung der Partikelvolumina  $V_i$  (Gleichung 2.36) ist es möglich diese auf

$$\rho = \frac{m}{\sum V_i} = \frac{7.9 \text{ g}}{22.1 \text{ ml}} = 358.45 \frac{\text{kg}}{\text{m}^3} \approx 360 \frac{\text{kg}}{\text{m}^3} \quad (2.36)$$

festzusetzen.

### 2.3.3 DEM-Partikelmodell

Im bisherigen Verlauf wurde die Längen/Breiten-Verteilung diskutiert, sowie die Partikelhöhenverteilung ermittelt. Damit wurde die Partikelgeometrie der Biomasse angenähert. Des Weiteren wurde die Dichte eines Biomassepartikels ermittelt. Die ermittelte Verteilung ist zu umfangreich, um sie direkt in DEM-Simulationen verwenden zu können. Daher wurde eine maximale Anzahl von vier Partikeln festgelegt (Abbildung 2.17). Für das Histogramm (Abbildung 2.13) wurde das dazugehörige Partikelvolumen bestimmt. Dieses wurde neu diskretisiert. Die Intervalle wurden frei gewählt. Es wurde jedoch darauf geachtet, das Volumen eines jeden diskreten Elementes relevant zu halten.

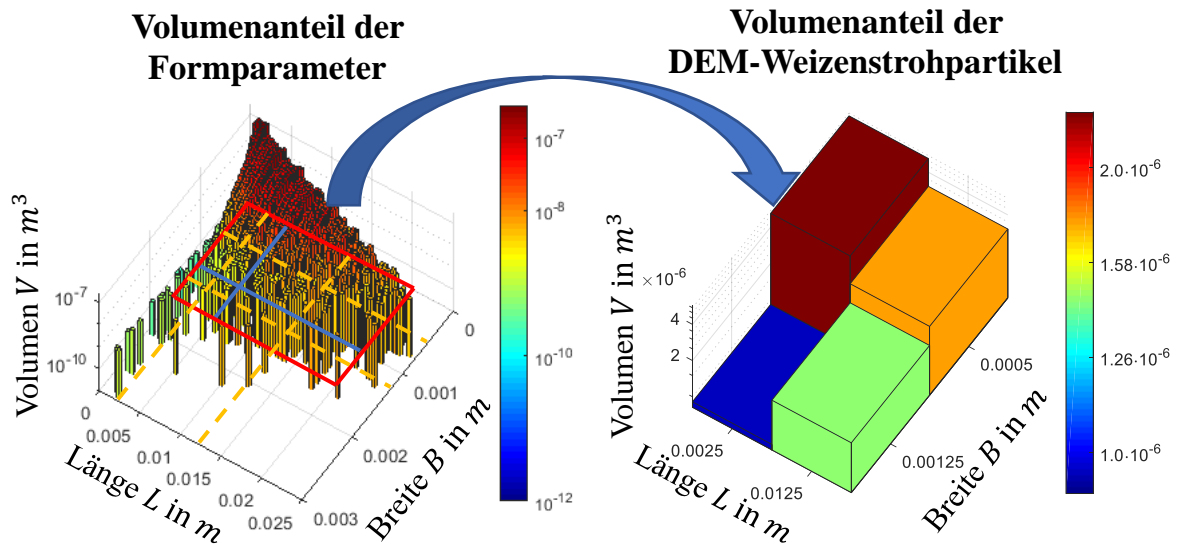


Abbildung 2.17: Diskretisierung der Partikelform

Mit der neuen Diskretisierung gehen neue Längen  $L$  und neue Breiten  $B$  einher. Diese entsprechen dem Mittelwert der ursprünglichen Diskretisierung. Die Gesamtvolumina der einzelnen diskretisierten Elemente wurden addiert. Die gewonnenen Informationen können genutzt werden, um Aussagen über Form und Verteilung der Partikel treffen zu können (Tabelle 2.11).

Tabelle 2.11: Partikelform und Gesamtvolumenanteil der Multisphäre-Weizenstrohpartikel

	Breite	Länge	Volumenanteil
<b>Partikel 1</b>	0.5 mm	2.5 mm	46.11 %
<b>Partikel 2</b>	0.5 mm	12.5 mm	27.06 %
<b>Partikel 3</b>	1.25 mm	2.5 mm	8.05 %
<b>Partikel 4</b>	1.25 mm	12.5 mm	18.78 %

Unter Verwendung des Multisphärenansatzes ist es möglich, die Partikelform beliebig genau anzupassen. Dies geht jedoch mit einem Mehraufwand, der proportional zu den verwendeten Kugeln steht, einher. Aufgrund der hohen Weizenstroh-Partikelzahl und der geringen Dicke (0.15 mm), der einzelnen Partikel, sind Grenzen gesetzt, wie groß die verwendeten Sphären zur Multisphäre-Darstellung sein dürfen. Dies führt zu einem erheblichen Mehraufwand im Sinne von Rechenzeit. Es muss daher geprüft werden, ob der Gewinn an Genauigkeit den Mehraufwand beim Einsatz des Multisphären-Ansatzes rechtfertigt. In einem ersten Schritt wurde die Anzahl der vier Biomassepartikel pro Gramm ermittelt. Da sowohl Form als auch der maximale Radius ( $h/2$ ) bekannt sind, kann die Anzahl der Sphären für den Multisphärenansatz bestimmt werden (Tabelle 2.12).

Tabelle 2.12: Abschätzung der erforderlichen Sphären für die Single- und Multisphäre-Auflösung für Biomassepartikel

	Partikel pro $g$ Biomasse	Sphären pro Partikel	Gesamtzahl der Sphären pro $g$ Biomasse
<b>Partikel 1</b>	8915	120	1098000
<b>Partikel 2</b>	1042	590	614780
<b>Partikel 3</b>	242	288	69696
<b>Partikel 4</b>	313	1416	443208
<b>Summe:</b>	10512		2197484

Bei einer Biomasse-Feedrate von  $10 \text{ kg/h}$  und einer Verweilzeit von bis zu  $16 \text{ s}$  sind mit dem Multisphere-Ansatz bis zu  $20\,000\,000$  Biomasse-Sphären zu simulieren. Dem gegenüber stehen knappe  $500\,000$  Biomasse-Sphären beim Singlesphäre-Ansatz. Eine Simulation des Reaktors mit einem Singlesphäre-Modell kann bis zu einem Monat auf  $16$  Kernen in Anspruch nehmen. Da die Parallelisierung unter dem Multisphären-Ansatz höheren Limitierungen unterliegt (höherer Datenaustausch durch Multisphären), wäre mit einem Ergebnis unter Verwendung des Multisphären Ansatzes im Rahmen dieser Dissertation nicht zu rechnen. Daher kann im weiteren Verlauf der Multisphären-Ansatz nur noch am Rande berücksichtigt werden.

Zur Partikelgrößenverteilung beim Singlesphäre-Ansatz wurden die in (Abbildung 2.17) diskretisierten Partikelvolumina  $V_p$ , unter Verwendung der Gleichung zur Berechnung des Kugelvolumens, (Gleichung 2.37) sowie der Partikelradius  $r_p$  ermittelt und in (Tabelle 2.13) eingetragen.

$$r_p = \left( \frac{3 \cdot V_p}{4\pi} \right)^{\frac{1}{3}} \quad (2.37)$$

Tabelle 2.13: Partikelradien und Gesamtvolumenanteil des Singlesphäre-Weizenstrohpartikels

	Breite	Länge	Radius $r_p$	Volumenanteil
<b>Partikel 1</b>	$0.5 \text{ mm}$	$2.5 \text{ mm}$	$\approx 0.32 \text{ mm}$	$45.9 \%$
<b>Partikel 2</b>	$0.5 \text{ mm}$	$12.5 \text{ mm}$	$\approx 0.54 \text{ mm}$	$27.17 \%$
<b>Partikel 3</b>	$1.25 \text{ mm}$	$2.5 \text{ mm}$	$\approx 0.43 \text{ mm}$	$8.07 \%$
<b>Partikel 4</b>	$1.25 \text{ mm}$	$12.5 \text{ mm}$	$\approx 0.73 \text{ mm}$	$18.85 \%$

Bis zu diesem Punkt wurde die Form der Biomassepartikel behandelt. Aus den erhobenen Daten wurde eine geeignete Partikelgrößenverteilung für Sphärische und nicht-Sphärische Partikel abgeleitet. Dabei wurde der Einfluss der Partikelform auf Materialparameter wie Reibung und Rollreibung außer Acht gelassen. Dies wird in kommenden Kapitel behandelt.

### 2.3.4 Einfluss der DEM-Partikelform auf das Mischverhalten

Aufgrund ihrer Kugelform sind Wärmeträger für DEM-Simulationen gut geeignet. Dies trifft jedoch nicht auf Weizenstrohpartikel zu, deren Form dem eines lang gezogenen Quader entspricht. Ein Maß zur Beschreibung der Formabweichung von Objekten zu einer Kugel, ist die Sphärizität  $\Psi$  (Gleichung 2.38). Eine perfekte Kugel hat eine Sphärizität von  $\Psi = 1$ . Mit zunehmender Formabweichung geht dieser Wert gegen  $0$  [76].

$$\Psi = \frac{\pi^{\frac{1}{3}}(6V_p)^{\frac{2}{3}}}{A_p} \quad (2.38)$$

Maione et. al. [69] beschäftigte sich mit dieser Problemstellung. Unter Verwendung von Stahlkugeln und Holzschnipseln wurden Singlesphären- und Multisphären-Simulationen in einem Trommelmischer durchgeführt. Die Kernaussage ist, dass bei einer Annäherung an eine Froudezahl von  $1$  das Mischverhalten von sphärischen und nicht-sphärischen Partikeln einander annähern. Bei abnehmender Froudezahl weicht das Mischverhalten voneinander ab. Diese Untersuchungen beschränkten sich jedoch auf Holzschnipsel einer Sphärizität von  $0.729$ . Diese wird im Rahmen dieser Arbeit weit unterschritten (Tabelle 2.14).

Tabelle 2.14: Spherizität der Singlesphären-Weizenstrohpartikel

	Breite	Länge	Volumenanteil	$h = 0.15 \text{ mm}$	
				Radius	Spherizität
<b>Partikel 1</b>	0.5 mm	2.5 mm	45.9 %	$\approx 0.32 \text{ mm}$	0.361
<b>Partikel 2</b>	0.5 mm	12.5 mm	8.07 %	$\approx 0.54 \text{ mm}$	0.222
<b>Partikel 3</b>	1.25 mm	2.5 mm	27.17 %	$\approx 0.43 \text{ mm}$	0.359
<b>Partikel 4</b>	1.25 mm	12.5 mm	18.85 %	$\approx 0.73 \text{ mm}$	0.225

Die für diese Arbeit erforderlichen Simulationen lassen sich nicht durch einen Multisphärenansatz umsetzen. Der enorm hohe Rechenaufwand, die inhärente Instabilität, sowie die fehlende Fähigkeit die Simulation bei bereits erreichtem Zustand neu zu starten, sind nur einige der Gründe, die dies verhindern. Da die Vermischung eine der Kernaussagen dieser Arbeit ist und die Partikelform maßgeblich dazu beiträgt, ist es sinnvoll den Einfluss der Partikelform zu bestimmen.

Jedoch mussten an dieser Stelle einige Abstriche bei den verwendeten Werten und dem Versuchsaufbau zugunsten der Stabilität und der Laufzeit unternommen werden. In einem ersten Schritt wurde die Partikelhöhe auf 0.3 mm gesetzt (Tabelle 2.14), wodurch weniger Sphären für ein Biomassepartikel benötigt werden. Werte für das E-Modul wurden um drei Größenordnungen reduziert (Tabelle 2.15). Dadurch konnte der Zeitschritt um eine Größenordnung erhöht werden. Schließlich wurde das Design der Trommel (Abbildung 2.3) modifiziert, indem der Durchmesser auf 60 mm und die Breite auf 20 mm reduziert wurden. Dadurch ist es möglich die Anzahl der Biomasse und Wärmeträgerpartikel um den Faktor 10 zu reduzieren (Tabelle 2.15).

Trotz all dieser Modifikationen, um die Schrittweite zu erhöhen und die Partikelzahl zu reduzieren, bleibt die Multisphärensimulation rechenintensiv. Zum Vergleich: Unter den genannten Bedingungen mussten für den Multisphärenansatz  $14.4 \cdot 10^6$  Partikel für dieses System simuliert werden, während für den Singlesphäre-Ansatz  $8.5 \cdot 10^4$  Partikel zu simulieren sind.

Tabelle 2.15: Aufbau der Multisphäre-DEM Simulation mit Drehtrommel

**Partikeleigenschaften:**

	Stahlkugeln	Weizenstroh	Weizenstroh/ Stahlkugeln
E-Modul [Pa]	$210 \cdot 10^6$	$5 \cdot 10^5$ [77]	
Poissonzahl	0.3	0.035 [78]	
COR	0.9	0.2 <sup>IV</sup>	0.6 <sup>V</sup>
COF	0.5	1.0	0.9
CORF	0.4	0.7	1.0
Dichte [ $kg/m^3$ ]	7878.4	360	
Radius [m]	$1 \cdot 10^{-3}$	(Tabelle 2.14)	

Drehtrommel:

Durchmesser [m]	$6 \cdot 10^{-2}$
Breite [m]	$2 \cdot 10^{-2}$

Program:

dt [s]	$1 \cdot 10^{-6}$ bzw. $5 \cdot 10^{-7}$
rpm [1/min]	10, 25, 45, 60, 75, 90, 120, 172, 240

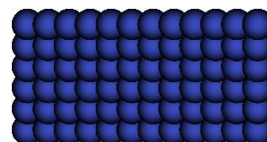
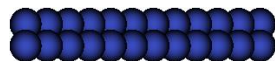
Programm:

Kreiere ein Biomassepartikel  
 Kreiere alle Wärmeträgerpartikel  
 Kreiere die restlichen Biomassepartikel  
 Simuliere mit vorgegebenem rpm

Simulationen unter diesen Rahmenbedingungen wurden sowohl für sphärische Biomassepartikel (Tabelle 2.14) als auch für multisphärische Biomassepartikel (Abbildung 2.18) identisch durchgeführt. Bei höheren Umdrehungen ( $> 90 \text{ rpm}$ ) ist eine Reduzierung der Schrittweite erforderlich geworden. Diese wurde sowohl für Multisphären-Simulationen als auch für Singlesphären-Simulationen gleichermaßen reduziert.

*Weizenstrohpartikel 1*

$0.5 \times 2.5 \times 0.3 \text{ mm}$   
 24 Sphären  
 45.9 %-Massenanteil



*Weizenstrohpartikel 3*

$1.25 \times 2.5 \times 0.3 \text{ mm}$   
 72 Sphären  
 27.17 %-Massenanteil



*Weizenstrohpartikel 2*

$0.5 \times 12.5 \times 0.3 \text{ mm}$   
 124 Sphären  
 8.07 %-Massenanteil

*Weizenstrohpartikel 4*

$1.25 \times 12.5 \times 0.3 \text{ mm}$   
 372 Sphären  
 18.85 %-Massenanteil

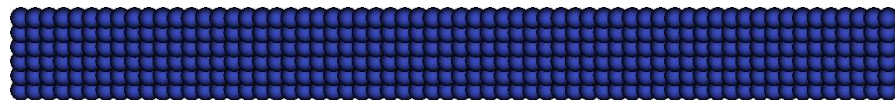


Abbildung 2.18: Multisphäre-Weizenstrohpartikel, eingesetzt bei der DEM-Simulation der Drehtrommel

Eine Auswertung der Multisphäre-Simulation erfolgt durch die Bestimmung der lokalen und globalen Mischgüte (Gleichung 2.39).  $\sigma_{\max}$  entspricht der höchsten lokal gemessenen Standardabweichung. Da in jedem Zeitschritt die Anzahl der Partikel bekannt ist, lässt sich der Erwartungswert  $E$  einfach

<sup>IV</sup> Aufgrund von fehlenden Werten, musste dieser Wert abgeschätzt werden

<sup>V</sup> Aufgrund von fehlenden Werten, musste dieser Wert abgeschätzt werden

bestimmen. So wird die Mischgüte  $MG$  lokal aus dem Erwartungswert  $E$  eines Zeitschrittes und der Zusammensetzung  $X_i$  in den diskreten Probenmengen bestimmt.

$$MG = 1 - \sqrt{\frac{\sigma^2}{\sigma_{\max}^2}} \quad (2.39)$$

mit  $\sigma^2 = \frac{1}{n} \sum_{i=1}^n (E - X_i)^2$

### Mischgüte in Abhängigkeit zur Partikelform

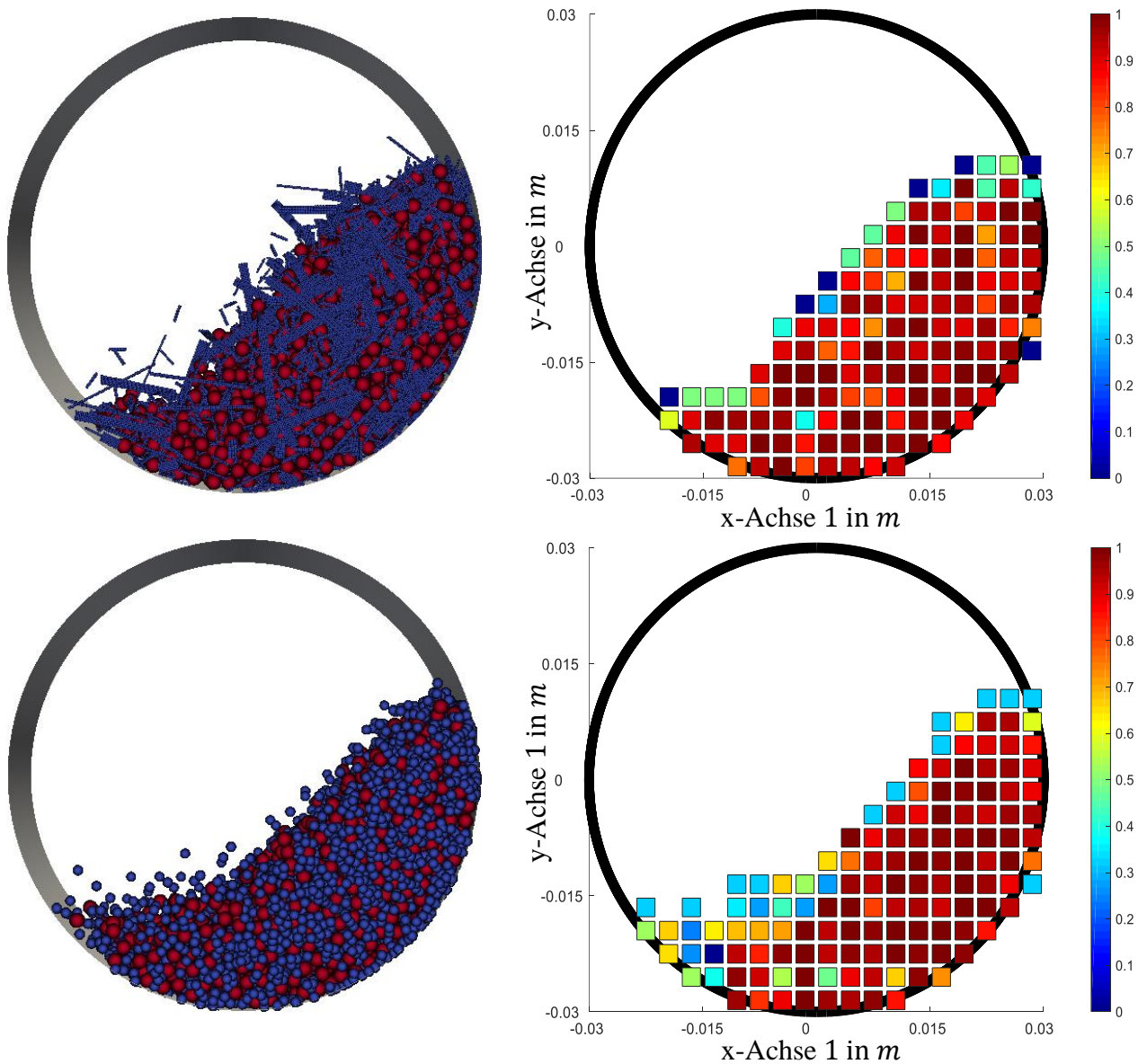


Abbildung 2.19: Single-/Multisphäresimulation einer Trommel mit 75 rpm, bei 13.6 s (links) und der dazugehörigen Mischgüte auf der x/y-Achse (rechts). Die Farbe der Auswertung gibt die lokale Mischgüte wieder.

Die Diskretisierung erfolgt durch  $3 \times 3 \text{ mm}$  Quadrate auf der x/y-Ebene. Ein weiteres Diskretisieren erfolgt entlang der z-Achse. Dadurch ist es möglich die lokale Mischgüte auf der x/y-Ebene (Abbildung 2.19) und die globale Mischgüte eines Zeitschrittes für die komplette Trommel zu bestimmen. Zur Auswertung wurde die Anzahl der Partikel bestimmt, deren Schwerpunkt sich im jeweiligen diskreten Element befindet.

Exemplarisch wurde der Fall für  $75 \text{ rpm}$  (bzw.  $Fr_W \approx 0.2$ ) betrachtet (Abbildung 2.19). Hier sind eine Reihe an Übereinstimmungen zu erkennen. Die Schüttwinkel der Multispheresimulation ( $43.4^\circ$ ) und der Singlespheresimulation ( $40.3^\circ$ ) stimmen weitestgehend überein. Auch die lokale Mischgüte auf der x/y-Ebene zeigt ein vergleichbares Verhalten. So ist innerhalb des Bettes eine sehr gute Durchmischung zu finden, während an den Rändern eine Entmischung zu beobachten ist, deren Intensität, abhängig von der Partikelform, schwankt. Betrachtet man die Auswertung der x/y-Ebene, so ist die Projektionsfläche der Multisphärens simulation größer ( $25.2 \text{ mm}^3$ ) als die Projektionsfläche der Singlesphärens simulation ( $23.7 \text{ mm}^3$ ). Dieser Flächenunterschied geht mit einer unterschiedlichen Schüttdichte einher. Aufgrund der großen Randfläche und der durch die Diskretisierung bedingten Abweichungen, ist diese Methode zur Bestimmung der Schüttdichte ungeeignet.

Daher wurde das Volumen einer realen Weizenstroh/Stahlkugelmischung im Massenverhältnis 1:100 experimentell in einem Messzylinder bestimmt. Aus dem Gewicht und Volumina konnte die Schüttdichte ermittelt werden. Eine DEM-Simulation mit sphärischen Partikeln unter identischen Rahmenbedingungen wurde identisch ausgewertet (Tabelle 2.16).

*Tabelle 2.16: Schüttdichte einer Weizenstroh/Stahlkugelmischung. Real vs. Simuliert*

	<b>Masse</b>	<b>Dichte</b>
<b>Wärmeträger</b>	300.006 g	7878.4 kg/m <sup>3</sup>
<b>Biomasse</b>	3.006 g	360 kg/m <sup>3</sup>
<b>Reales Gemisch</b>	303.012 g	3366.8 kg/m <sup>3</sup>
<b>Simuliertes Gemisch</b>	303.012 g	3963.6 kg/m <sup>3</sup>

Auch hier wird die Aussage bestätigt, wonach DEM-Simulationen mit sphärischen Partikeln eine höhere Schüttdichte aufweisen. Ursache ist die sperrige Biomasseform, welche Zwischenräume effektiv blockiert (Abbildung 2.19). Ein Effekt der mit Sphären nicht wiedergegeben werden kann. Im Vergleich zu dem realen Gemisch weisen Simulationen mit sphärischen Partikeln eine bis zu 15 % höhere Dichte auf und im Vergleich zu multisphärischen Simulationen eine 10 % höhere Schüttdichte. Jedoch ist zu berücksichtigen, dass bei den multisphärischen Simulationen die doppelte Partikeldicke gewählt wurde. Daher ist mit der ermittelten Dicke (Kapitel 2.3.2.3) die doppelte Partikelanzahl zu erwarten. Aus diesem Grund kann davon ausgegangen werden, dass sich die Schüttdichte der Simulationen mit Multisphären in einem vergleichbaren Bereich wie die des realen Gemischs befindet.

Zusätzlich wurde der Verlauf der Mischgüte in Abhängigkeit zur Froudezahl bestimmt (Abbildung 2.20).



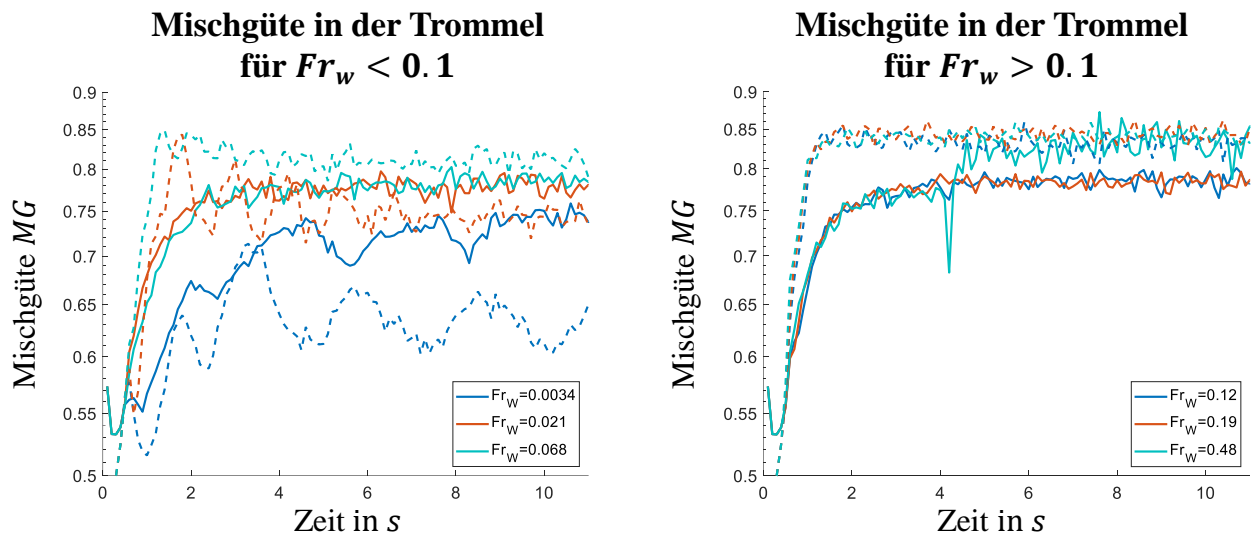


Abbildung 2.20: Verlauf der globalen Mischgüte in der rotierenden Trommel für Simulationen mit Multisphären (durchgezogene Linie) und Sphären (gestrichelte Linie)

Für sehr kleine Froudezahlen ( $Fr_w = 0.0034$ ) verläuft die Durchmischung durchgehend schleppend. Erst ab einer Froudezahl von  $Fr_w \geq 0.021$  ist dies nicht mehr der Fall.

Während die Mischgüte bei Simulationen mit Multisphären einen exponentiellen Verlauf annimmt, steigt die Mischgüte mit sphärischen Partikeln linear, bis der Konvergenzpunkt erreicht ist. Im Allgemeinen kann gesagt werden, dass die Mischgüte mit sphärischen Partikeln schneller konvergiert. Dennoch unterliegt die Mischgüte bei der Verwendung von Multisphären ab diesem Zeitpunkt keinen größeren Veränderungen.

Des Weiteren fällt auf, dass die Mischgüte für sphärische und multisphärische Partikel gegen unterschiedliche Werte konvergieren und sie gleichen sich erst ab einer Froudezahl von  $Fr_w = 0.48$  an.

## 2.4 Bestimmung der Wärmeträger/Biomasse Kennwerte

Da die Auswertung von Versuchen in einer rotierenden Trommel maßgeblich vom der Schüttdichte und -volumen abhängt, kann dieser Versuch, aufgrund der abweichenden Schüttdichten in Experimenten und in Simulation mit sphärischen Partikeln, nicht verwendet werden. Stattdessen wurden die erforderlichen Werte unter den Annahmen (Gleichung 2.40) und den soweit ermittelten Werten aus Kapitel 2.2.1 im ColdFlow-Modellreaktor bestimmt.

$$\begin{aligned}
 COF_{WT-WT} &< COF_{WT-BM} \leq COF_{BM-BM} = 0.9 \\
 CORF_{WT-WT} &< CORF_{WT-BM} \leq CORF_{BM-BM} = 1.0 \\
 COF, CORF &\geq 0.1
 \end{aligned}
 \tag{2.40}$$

Diese Versuche und deren Simulationen (Tabelle 2.17), bei denen Biomasse involviert ist, können nur mit großem Aufwand durchgeführt werden. Bei den Versuchen ist die Trennung von Biomasse und Wärmeträger eine langwierige Prozedur. Bei den Simulationen stellen die höhere Partikelzahl und der damit verbundene Aufwand ein Problem dar. Daher mussten Abstriche gemacht werden. Diese erstreckten sich in den Versuchen auf die Anzahl der Wiederholungen und bei den Simulationen auf die Laufzeit und in der Anzahl der simulierten Partikel. D.h. es wurden größere Wärmeträgersphären verwendet und mindestens drei Verweilzeiten simuliert.

Tabelle 2.17: Rahmenbedingungen der Simulationen zur Bestimmung der Wärmeträger/Biomasse Reibung und Rollreibung

**Partikeleigenschaften:**

	Stahlkugeln	Biomasse	Stahlkugeln/ Biomasse
E-Modul [Pa]	$210 \cdot 10^6$	$5 \cdot 10^6$	
Poissonzahl	0.3	0.035	
<i>COR</i>	0.9	0.2	0.6
<i>COF</i>	0.4/0.2	0.9	0.3-0.9
<i>CORF</i>	0.1-0.3	1.0	0.2-0.9
Dichte [ $kg/m^3$ ]	7878.4	360	
Radius [m]	$10^{-3}$	(Tabelle 2.14)	
Verteilung [% <sub>m</sub> ]	1	(Tabelle 2.14)	
Massenrate [ $kg/h$ ]	920	8.5	

**Programm**

	<u>2 Hz</u>
Simulationszeit [s]	24
dt [s]	$10^{-6}$

Experimentell wurden Verweilzeiten wie in Kapitel 2.2.1.3 über die Verteilungsfunktion, bestimmt und in Simulationen über die hydrodynamische Verweilzeit (Gleichung 2.41). Des Weiteren wurde die Auswertung auf Betriebsbedingungen von 2 Hz beschränkt.

$$\tau = \frac{V_{\text{Reaktor}}}{\dot{V}_{\text{ein}}} = \frac{m_{\text{Reaktor}}}{\dot{m}_{\text{ein}}} \quad (2.41)$$

Ein Vergleich der experimentell ermittelten Verweilzeit (Tabelle 2.18, 2 Hz) und der simulativ ermittelten Verweilzeit (Tabelle 2.19) ergibt eine erhebliche Diskrepanz und physikalisches Verhalten. Es konnte kein Wertpaar gefunden werden, sodass die Verweilzeit sich in einem vergleichbaren Bereich befindet. Darüber hinaus kommt es ab einem  $COF_{WT-BM} \geq 0.6$  zu einem Rückstau im Reaktor, der in Versuchen nicht reproduziert werden konnte.

Tabelle 2.18: Experimentelle Bestimmung der Verweilzeit im Doppelschneckenmischreaktor von Weizenstroh und Biomasse [70]

<b>Hz</b>	<b>Bo</b>	<b><math>\tau</math> in [s]</b>
<b>1</b>	162.08	21.98
<b>2</b>	68.23	10.16
<b>3</b>	19.55	7.18
<b>3</b>	31.94	6.56
<b>4</b>	28.57	5.87
<b>4</b>	47.06	4.88

*Tabelle 2.19: Validierung der Biomasse/Wärmeträger Kennwerte für eine 2 Hz Rotationsgeschwindigkeit der Schnecken. Bei den rot markierten Zeilen kam es zu einem Rückstau bei der Wärmeträgerzufuhr*

		<b>CORF</b>							
		<b>0.2</b>	<b>0.3</b>	<b>0.4</b>	<b>0.5</b>	<b>0.6</b>	<b>0.7</b>	<b>0.8</b>	<b>0.9</b>
<b>COF</b>	<b>0.5</b>	14.09 s	13.98 s	13.90 s	13.83 s	13.81 s	13.77 s	13.75 s	13.53 s
	<b>0.6</b>	14.19 s	14.08 s	13.91 s	13.97 s	13.78 s	13.76 s	13.73 s	13.54 s
	<b>0.7</b>	.	.	.	.	.	.	.	.
	<b>0.8</b>	.	.	.	.	.	.	.	.
	<b>0.9</b>	.	.	.	.	.	.	.	.

Ursachen hierfür können auf getroffene Annahmen zurückgeführt werden. Während die bisher diskutierten Annahmen begründet werden können, stellt der Einfluss der Partikelform weiterhin ein Problem dar.

Im vorangegangenen Kapitel konnte beobachtet werden, dass DEM-Simulationen mit sphärischen Partikeln eine höhere Dichte aufweisen, als Simulationen mit Multisphären. Dies deutet darauf hin, dass durch nicht sphärische Partikel die Struktur des Partikelbettes sehr viel stärker gestört wird, als es sphärische Partikel vermögen. Damit führen nicht-sphärische Partikel zu einer Reduzierung der Kontaktstellen, wodurch die Krafteinwirkung von Partikel zu Partikel behindert wird.

Daher kann die Biomasse/Wärmeträger und die Wärmeträger/Wärmeträger Reibung und Rollreibung nicht unabhängig voneinander bestimmt werden.

Die Anzahl der erforderlichen Simulationen um alle Parameter gleichzeitig zu betrachten macht dies jedoch unmöglich. Daher konnten im weiteren Verlauf die Werte lediglich abgeschätzt werden. Aus vorhergehenden Simulationen (Tabelle 2.19) kann entnommen werden, dass steigende Werte für den Wärmeträger/Biomasse  $COF$  und  $CORF$  zu einer Verlängerung der Verweilzeit führen. Da diese Werte nicht weiter reduziert werden können ohne die Annahme (Gleichung 2.40) zu verletzen, muss der Wert für die Wärmeträger/Wärmeträger  $COF$  und  $CORF$  reduziert werden. Betrachtet man nun (Tabelle 2.8), so wird deutlich, dass die in (Tabelle 2.18) bestimmten Verweilzeiten im getesteten Wertebereich überschritten werden. Daher muss für eine Simulation mit sphärischen Biomasepartikeln ein Wert  $COF_{WT/WT} < 0.3$  gewählt werden.

D.h. Im weiteren Verlauf wurde ein  $COF_{WT/WT} = 0.2$  und  $CORF_{WT/WT} = 0.1$  getestet.

Unter Berücksichtigung der Schwankungen in (Tabelle 2.18, 3 u. 4 Hz), ist die Annäherung (Tabelle 2.20) bei einer Wärmeträger/Biomasse  $COF$  von 0.3 am besten. Unklar ist hierbei jedoch der Einfluss der Rollreibung. Für den weiteren Verlauf wird ein  $CORF$  von 0.3 vorausgesetzt.

*Tabelle 2.20: Verweilzeiten in Abhängigkeit zur Biomasse/Wärmeträger  $COF$  und  $CORF$*

		<b>CORF</b>							
		<b>0.2</b>	<b>0.3</b>	<b>0.4</b>	<b>0.5</b>	<b>0.6</b>	<b>0.7</b>	<b>0.8</b>	<b>0.9</b>
<b>COF</b>	<b>0.3</b>	11.19 s	11.24 s	11.23 s	11.21 s	11.16 s	11.17 s	11.14 s	11.13 s
	<b>0.4</b>	12.04 s	12.13 s	12.17 s	12.17 s	12.12 s	12.12 s	12.10 s	12.07 s
	<b>0.5</b>	12.74 s	12.84 s	12.94 s	12.93 s	12.95 s	12.93 s	12.91 s	12.77 s
	<b>0.6</b>	13.37 s	13.48 s	13.51 s	13.56 s	13.55 s	13.59 s	13.59 s	13.54 s

Es ist zu erwarten, dass eine weitere Reduktion des  $COF_{WT-WT}$  niedrigere Verweilzeiten liefert. Jedoch zeigen die vorhandenen Wertepaare ein akzeptables Ergebnis, sodass auf eine weitere Investigation in Anbetracht der erforderlichen Zeit und Ressourcen, verzichtet wurde.

## 2.5 Einfluss der Biomasse/Biomasse Reibung und Rollreibung

Bei Versuchen ausschließlich mit Biomasse neigt Weizenstroh dazu, bedingt durch die Partikelform, größere Cluster zu bilden, sodass sich zu keinem Zeitpunkt ein stationärer Zustand einstellt. Versuche im ColdFlow-Modellreaktor können aufgrund der flachen Steigung im Misch- und Reaktionsbereich nicht ausschließlich mit Biomasse gefahren werden. Versuche, es dennoch umzusetzen, würden neue Schnecken erfordern. Da auch hier der (Stahl-)Wand/Biomasse Kontakt berücksichtigt werden muss, ist zuerst zu klären, ob und inwiefern der Biomasse/Biomasse Kontakt eine Rolle spielt. Dazu wurde eine bestehende stationäre Simulation (Kapitel 2.4) als Basis genommen, die Biomasse/Biomasse-Kennwerte geändert (Tabelle 2.21) und weitere 4 s Sekunden simuliert.

Tabelle 2.21: Materialkennwerte zur Bestimmung des Einflusses des Biomasse-Biomasse COF und CORF

	WT/WT	BM/BM	WT/BM
<b>E-Modul [Pa]</b>	$2.1 \cdot 10^9$	$5 \cdot 10^7$	
<b>Poisson</b>	0.3	0.035	
<b>COR</b>	0.9	0.2	0.3
<b>COF</b>	0.2	0.9 vs. 0.5	0.3
<b>CORF</b>	0.1	0.7 vs. 0.3	0.3

Eine Auswertung erfolgte durch eine Diskretisierung entlang der Förderrichtung. Für diese diskreten Teilmengen (Scheiben) wurde die Mischgüte nach (Gleichung 2.39) berechnet. Die erforderlichen Teilmengen wurden durch Diskretisierung in die verbliebenen Richtungen bestimmt. Dies ermöglicht die Beobachtung der Vermischung entlang der Förderrichtung und den direkten Vergleich der Vermischung innerhalb der Reaktorposition (Abbildung 2.21).

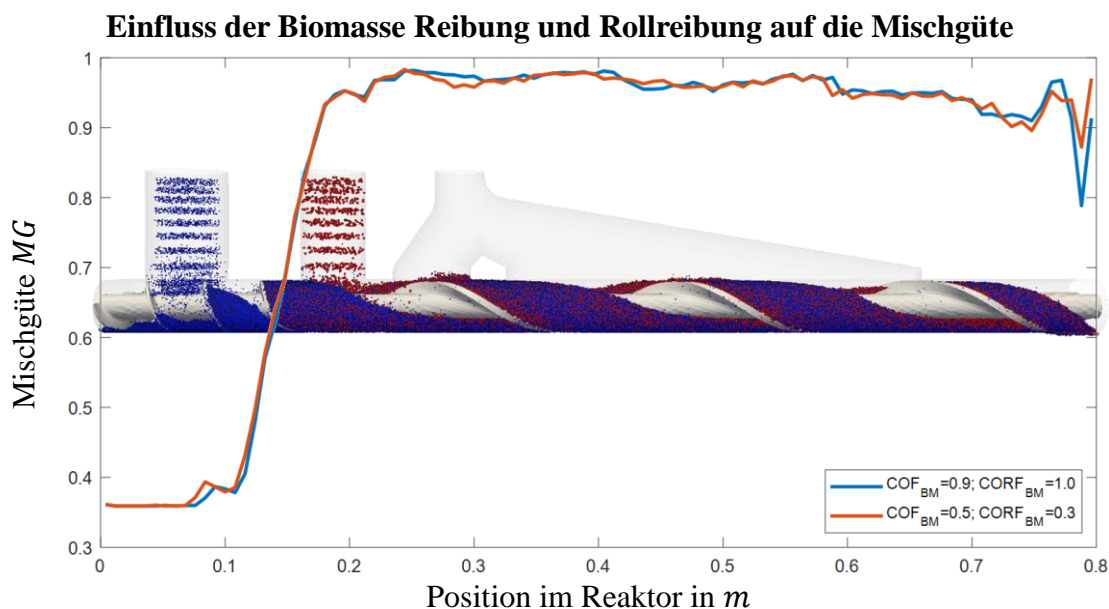


Abbildung 2.21: Einfluss der Biomasse Reibung auf die Mischgüte bezogen auf die Reaktorposition

Um die Vergleichbarkeit zu erhöhen, wurde darauf geachtet, dass die Positionen der Schnecken zum Zeitpunkt der Auswertung übereinstimmen. Die Abweichung der beiden Zustände bzgl. der Vermischung ist jedoch minimal und daher zu vernachlässigen. Erst im Bereich des Ausganges

können kleinere Veränderungen festgestellt werden. Diese können jedoch auch auf natürliche Schwankungen zurückgeführt werden.

Tabelle 2.22: Verweilzeit im Reaktor für die Biomasse-Biomasse  $COF$  und  $CORF$  Einstellungen.

$COF_{BM}/$ $CORF_{BM}$	$\tau_{WT}$ in s	$\tau_{BM}$ in s
0.9/1.0	11.24	17.67
0.5/0.3	11.19	17.68

Weiterhin wurde die hydrodynamische Verweilzeit für Wärmeträger und Biomasse für beide Einstellungen bestimmt (Tabelle 2.22). Wie auch schon in (Abbildung 2.21) zeigen die Änderungen der  $COF_{BM}$  und  $CORF_{BM}$  keine gravierenden Auswirkungen auf die Verweilzeiten. Anhand dessen kann geschlussfolgert werden, dass die Biomasse/Biomasse Reibung und Rollreibung keinen größeren Einfluss auf die Verweilzeit oder die Mischgüte ausübt.

## 2.6 Einfluss des Elastizitätsmoduls auf das Mischverhalten

Während Materialkennwerte wie die Poisson-Zahl und der  $COR$  aus der Literatur entnommen oder experimentell bestimmt werden, müssen die Parameter für Reibung und Rollreibung durch einen Abgleich von Experimenten und Simulation generiert werden. Dadurch können kleinere Abweichungen in Form und Materialoberfläche ausgeglichen werden. Eine Besonderheit stellte das E-Modul dar. Aufgrund des großen Einflusses auf das Schermodul (Gleichung 2.13) wird dieses in der Praxis sehr viel kleiner gewählt, sodass eine größere Überlappung zugelassen wird. Dies ermöglicht die Wahl eines größeren maximalen Zeitschrittes, welche die Rayleigh (Gleichung 2.23) Bedingung erfüllt. Dadurch kann die Rechenzeit beträchtlich reduziert werden. Im Allgemeinen wird der Einfluss des E-Moduls auf das Ergebnis der Simulation als vernachlässigbar betrachtet [55]. Jedoch existierten nach dem Wissen des Autors, zum Zeitpunkt der Niederschrift dieser Arbeit, keine Studien, die diese Aussage bestätigen. Daher wurde an dieser Stelle eine kleine Sensitivitätsstudie durchgeführt, die den Einfluss des Elastizitätsmoduls betrachtet.

Dazu wurde aus einem bestehenden Simulationszustand (Kapitel 2.4) heraus, 10 Sekunden mit einem niedrigeren E-Modul (Tabelle 2.23, blau) für Wärmeträger und Biomasse und einem Zeitschritt von  $dt = 10^{-5}$  s simuliert. Im Anschluss wurden für den erreichten Zustand die Werte für das E-Modul auf den Ursprungszustand gehoben (Tabelle 2.23, grün) und der Zeitschritt wieder auf  $dt = 10^{-6}$  s gesetzt (Tabelle 2.23). Unter diesen Rahmenbedingungen wurden weitere 4 s simuliert.

Tabelle 2.23: Materialkennwerte zur Bestimmung des Einflusses des E-Moduls und der Zeitschrittweite

	WT/WT	BM/BM	WT/BM
E-Modul [Pa]	$2.1 \cdot 10^6 /$ $2.1 \cdot 10^9$	$5 \cdot 10^5 /$ $5 \cdot 10^6$	
Poisson	0.3	0.035	
$COR$	0.9	0.2	0.3
$COF$	0.2	0.9/0.5	0.3
$CORF$	0.1	0.7/0.3	0.3

Dieses Simulation erreicht keinen stationären Zustand, jedoch lässt sich feststellen, ob und in welche Richtung Änderungen eintreten. Dazu wurde die Mischgüte  $MG$  entlang des Reaktors bestimmt

(Abbildung 2.22). Dies wurde sowohl für den Endzeitpunkt der Simulation mit kleinen E-Moduln und großem  $dt$  als auch im Sekundentakt für die Simulation mit erhöhten E-Moduln und kleinem  $dt$  bestimmt.

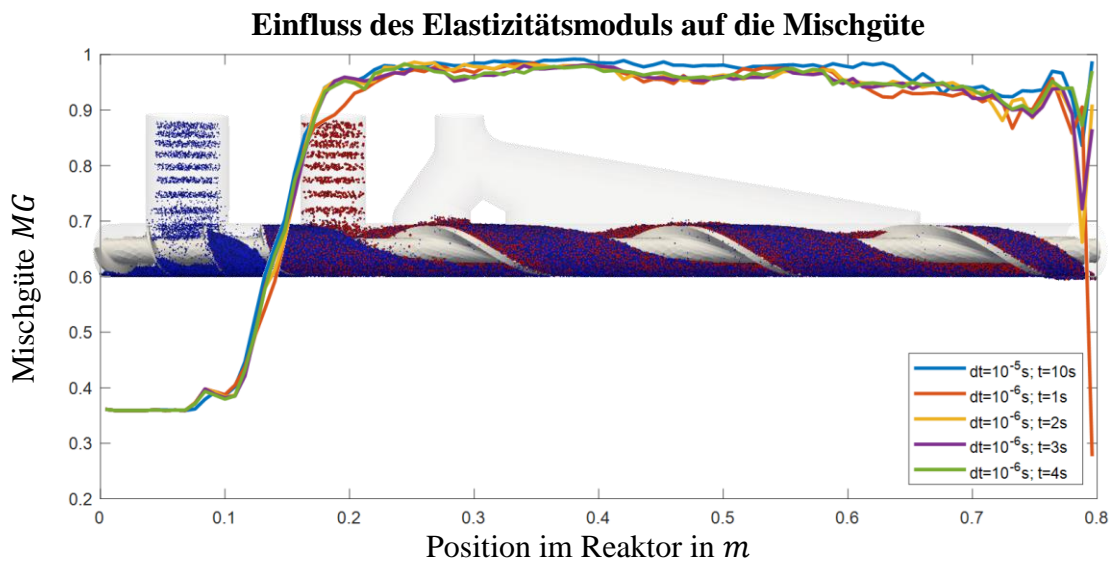


Abbildung 2.22: Einfluss des Elastizitätsmoduls auf die Mischgüte im Doppelschneckenmischreaktor

Ein Vergleich der Mischgüte bzgl. der Reaktorposition zeigt, dass es durchaus einen Unterschied zwischen den beiden Konfigurationen gibt. So ist die Mischgüte für die Simulation mit kleinem E-Moduln im Allgemeinen höher, als die Mischgüte für höhere E-Moduln. Die Auswertungen nach 1 und 2 s weisen noch kleinere Unterschiede gegenüber der Auswertung nach 4 s auf, während kaum ein Unterschied zwischen den 3 s und 4 s festzustellen ist.

Ein deutlicheres Bild liefert die Analyse der Verweilzeiten (Abbildung 2.23). Während die hydrodynamische Verweilzeit für den Wärmeträger und die Biomasse für kleine E-Moduln und großes  $dt$  monoton steigt, fällt diese für größere E-Moduln und kleineres  $dt$  monoton. Die maximale Veränderung der Verweilzeit ist unter einer Sekunde, jedoch kann an diesem Punkt gesagt werden, dass die gängige Praxis E-Moduln klein zu wählen, um einen Geschwindigkeitsvorteil zu erzielen, nur bedingt geeignet ist.

Sind Werte für  $COF$  und  $CORF$  für hohe E-Moduln validiert, kann in einem geschlossenen System, bei dem sich die Partikelzahl nicht verändert, Simulationen mit niedrigen E-Moduln und höherem  $dt$  beschleunigt werden. Eine oder mehrere finale Simulationen mit höherem E-Moduln und niedrigen  $dt$  pendeln die Mischgüte in den richtigen Wertebereich schnell ein.

Bei einem kontinuierlichen System ist jedoch bei gleichen Vorgehen mindestens eine Verweilzeit zu simulieren um den Wechsel der E-Moduln und deren Auswirkung auslaufen zu lassen.

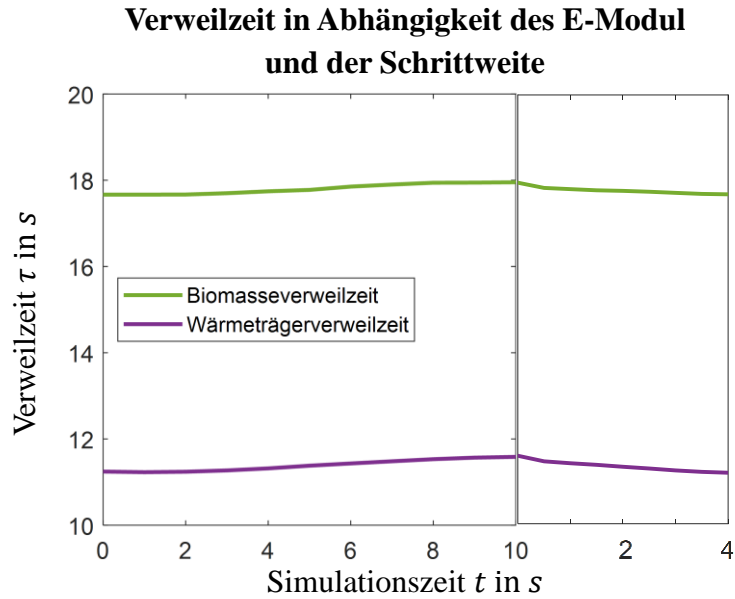
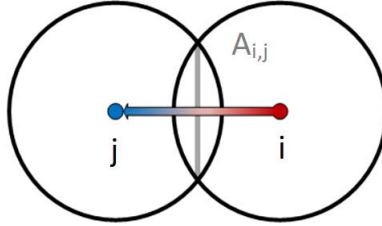


Abbildung 2.23: Verweilzeit in Abhängigkeit des E-Modul und der Schrittweite. Links: Kleine E-Moduln und großes dt. Rechts: Größere E-Moduln und kleines dt

## 2.7 DEM: Wärmeleitung und -übergang im Doppelschneckenmischreaktor

Neben der korrekten Darstellung des Mischverhaltens, ist eine richtige Darstellung des Wärmeüberganges von Wärmeträger zu Biomassepartikel zur Simulation der Pyrolyseprozesse im Doppelschneckenmischreaktor unerlässlich. LIGGGHTS® bietet die Möglichkeit den Wärmeübergang zweier Partikel zu berechnen (Gleichung 2.42, [62]). Dabei wird der Wärmeübergang von Partikel  $i$  nach Partikel  $j$  als Wärmeleitung durch die projizierte Kontaktfläche  $A_{i,j}$  (Gleichung 2.42, grau) und Partikelabstand  $d_{i,j}$  zwischen den Partikeln  $i$  und  $j$  betrachtet (Gleichung 2.42).

$$\begin{aligned}
 \dot{Q}_{i,j} &= \alpha_{i,j} \cdot (T_i - T_j) \\
 \alpha_{i,jB} &= 4 \cdot \frac{\lambda_i \cdot \lambda_j}{\lambda_i + \lambda_j} \cdot \sqrt{A_{i,j}} \\
 \dot{Q}_i &= m_i \cdot c_{P_i} \cdot \frac{dT_i}{dt} = \sum_{\text{kontakt}(i)} \dot{Q}_{i,j}
 \end{aligned}$$


(2.42)

mit

$$A_{i,j} = \frac{\pi \cdot (d_{i,j} - r_i - r_j) \cdot (d_{i,j} + r_i - r_j)^2 \cdot (d_{i,j} + r_i + r_j)}{4 \cdot d_{i,j}^2}$$

$T_i$  ist die Partikeltemperatur,  $\lambda_i$  ist die Wärmeleitfähigkeit,  $m_i$  die Partikelmasse,  $c_{P_i}$  die Partikelwärmekapazität und  $\dot{Q}_{i,j}$  der Wärmestrom von Partikel  $i$  nach  $j$ .

Batchelor et al. [79] führte dieses Modell ein um die Wärmeleitung durch die Kontaktfläche zweier Sphären zu beschreiben. Chaudhuri et al. [80] implementierte es schließlich in eine DEM-Simulation, welche heute als industrieller Standard gilt [81].

Durch die Abhängigkeit des Wärmeübergangs von der Überlappungsfläche ist eine Klassifizierung dieser erforderlich. Dazu wurde die Überlappungsfläche nach (Gleichung 2.42) in der Simulation einer rotierenden Trommel (Kapitel 2.2.1.2) mit den Parametern in (Tabelle 2.24) bestimmt und in einem Histogramm aufgetragen (Abbildung 2.24).

Tabelle 2.24: Simulationsparameter zur Bestimmung der Verteilung der Überlappungen

**Partikeleigenschaften:**

	Stahlkugeln bzw. Reaktorwand	Biomasse	Stahlkugeln/ Biomasse
E-Modul [Pa]	$210 \cdot 10^6$	$5 \cdot 10^6$	
Poissonzahl	0.3	0.035	
COR	0.9	0.2	
COF	0.2	0.9	0.3
CORF	0.1	0.7	0.3
Wärmeleitfähigkeit [W/(m · K)]	$40^{[82] \text{ VI}}$	$0.17^{[1]}$	
Wärmekapazität [J/(kg · K)]	$449^{[83]}$	$2000^{[1]}$	
Dichte [kg/m <sup>3</sup> ]	7878.4	360	
Radius [m]	$r_{WT} = 5 \cdot 10^{-4}$	$r_{BM_1} = 3.16 \cdot 10^{-4}$ $r_{BM_2} = 4.29 \cdot 10^{-4}$ $r_{BM_3} = 5.4 \cdot 10^{-4}$ $r_{BM_4} = 7.3 \cdot 10^{-4}$	
Massenstrom [kg/s]	$\dot{m}_{WT} = 5.71 \cdot 10^{-3}$	$\dot{m}_{BM_1} = 25.7 \cdot 10^{-4}$ $\dot{m}_{BM_2} = 8.64 \cdot 10^{-4}$ $\dot{m}_{BM_3} = 9.34 \cdot 10^{-4}$ $\dot{m}_{BM_4} = 13.4 \cdot 10^{-4}$	

**Programm**

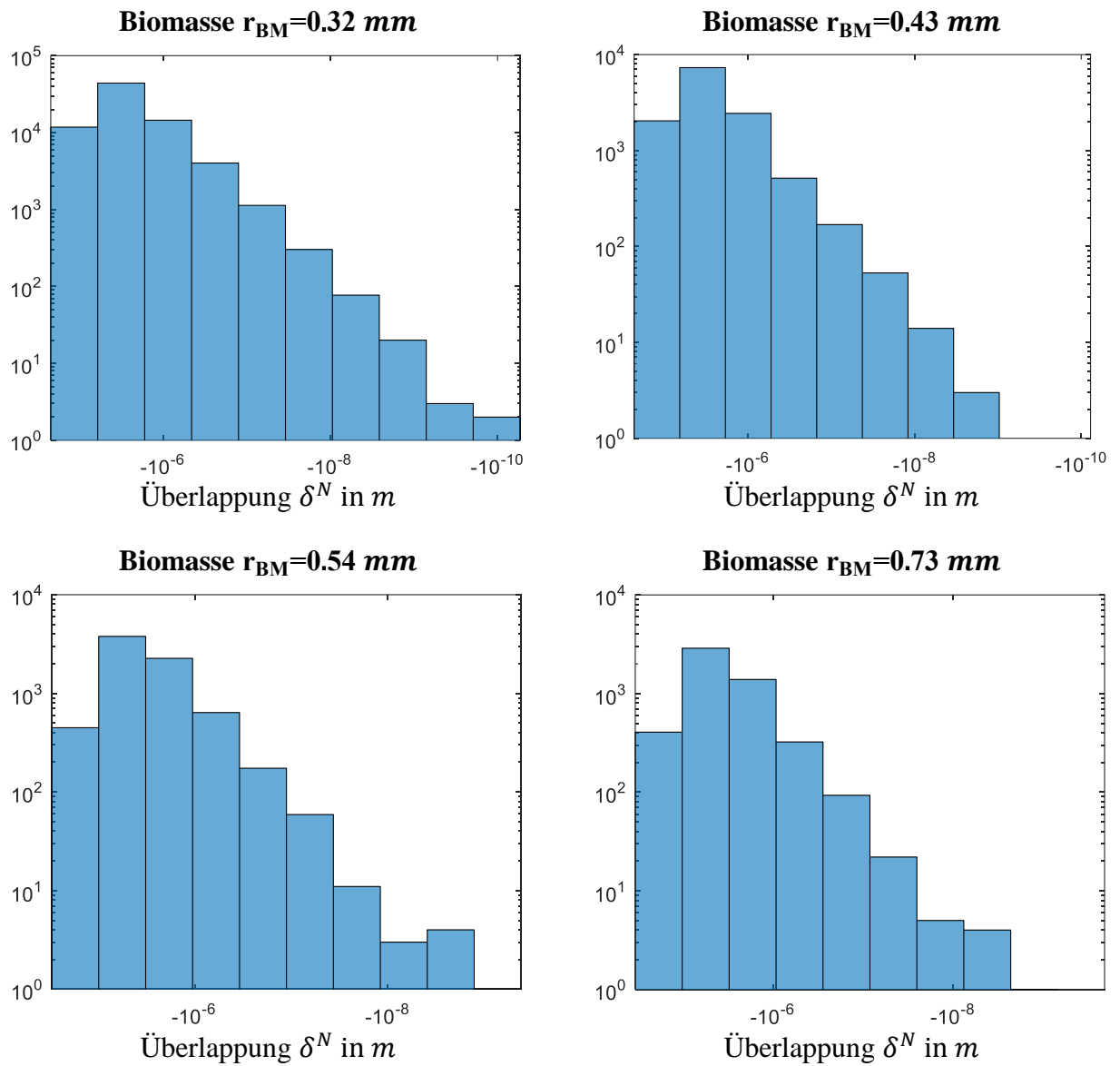
	<u>2 Hz</u>
Simulationszeit [s]	20
dt [s]	$2 \cdot 10^{-6}$

Die größte Temperaturdifferenz ist zwischen Biomasse und Wärmeträger zu erwarten. Daher lag das Hauptaugenmerk auf der Validierung des Biomasse/Wärmeträger-Wärmeübergang. Bei der Validierung des Biomasse/Biomasse bzw. Wärmeträger/Wärmeträger-Wärmeübergang wurde jedoch identisch vorgegangen.

<sup>VI</sup> Aufgrund des Chromanteil im Stahl gewählt



## Verteilung der Projektionsfläche bei der Überlappung von Biomasse und Wärmeträger



*Abbildung 2.24: Verteilung der Abstände an den Biomasse/Wärmeträger-Kontaktstellen*

Um eine Vergleichbarkeit mit anderen Modellen zu gewährleisten wurde als Skala die Überlappung  $\delta_{i,j}^N$  gewählt. Für den in (Abbildung 2.24) bestimmten Wertebereich wurde der Wärmeübergang nach (Gleichung 2.42) bestimmt (Abbildung 2.25).

## Temperaturverlauf beim Wärmeübergang nach Batchelor von Wärmeträgerpartikel nach Biomassepartikel

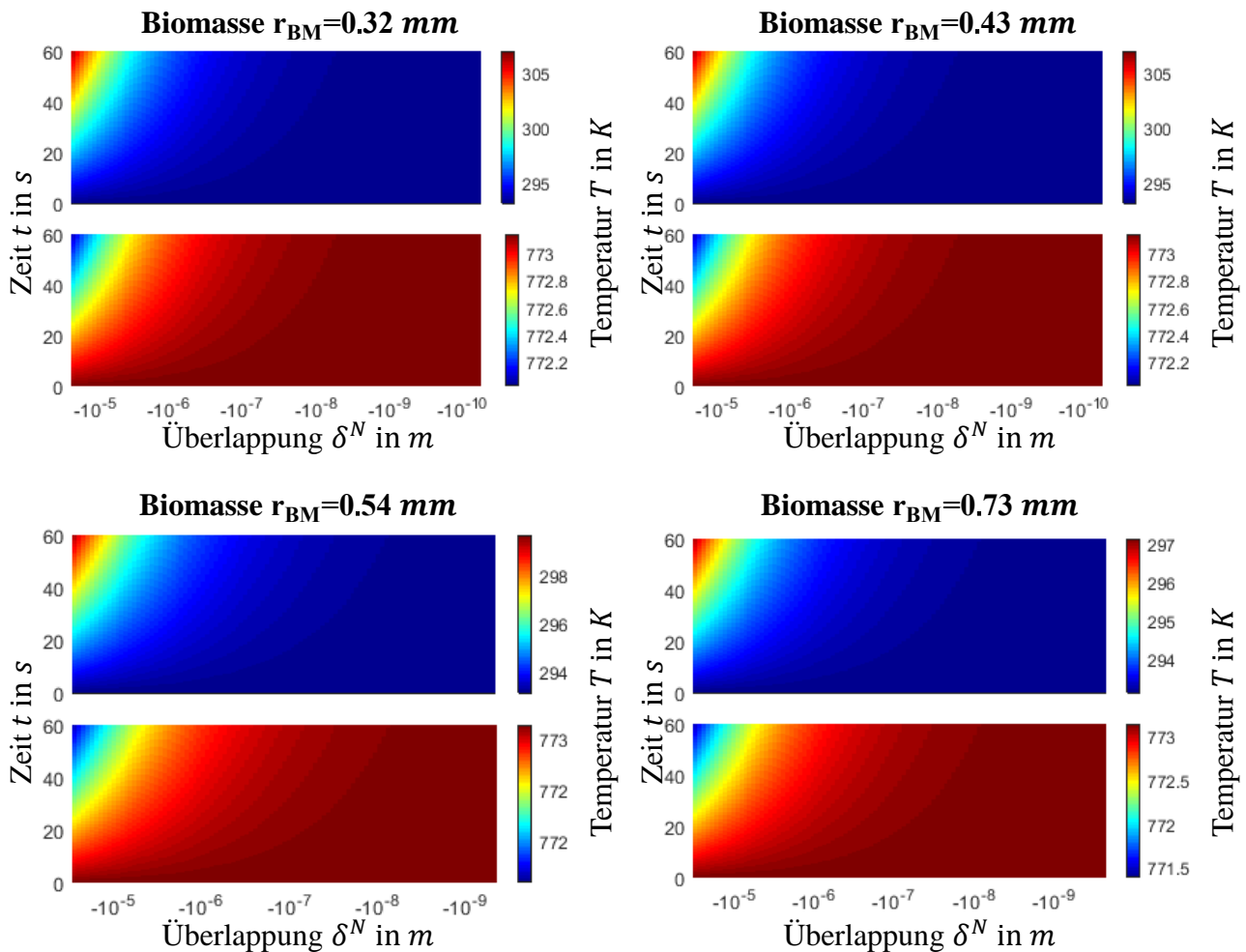


Abbildung 2.25: Temperaturverlauf beim Biomasse-Wärmeträgerkontakt, berechnet nach Batchelor. Wärmeträger (unten), Biomassepartikeln (oben)

Der Temperaturverlauf (Abbildung 2.25) der Biomasse ist, nach Batchelor und gegebenen Materialkennwerten, gering. Beobachtungen im Labor können dieses Verhalten nicht belegen. Versuche mit einem Batch-Pyrolysereaktor zeigen, dass ein Großteil der Reaktionen nach wenigen Sekunden abgelaufen sind [1]. Auswertungen der Pyrolyse am TG belegen, dass die endothermen Reaktionen u.a. von Lignin, in einem Temperaturbereich bis einschließlich 500 °C zu beobachten sind. Daher muss der tatsächliche Wärmeübergang um ein Vielfaches höher sein, als dieser nach Batchelor beschrieben wird.

Eine Methode, den Wärmeübergang von Wand zu einem Partikel darzustellen, beschrieb Schlünder et al. [2]. Dabei wird der Wärmeübergangskoeffizient  $\alpha_{i,jS}$  zwischen zwei parallelen Wänden durch eine Gasphase berechnet. Ein exakteres Bild ergibt sich durch ein Diskretisieren der Wandfläche  $\bar{A}$  in Wandteilflächen  $\bar{A}_k$ . Für diese Teilflächen muss gelten:  $\bar{A} = \sum \bar{A}_k$ . Für jede Wandteilfläche  $A_k$  muss der orthogonale Abstand  $s_k$  zur Partikeloberfläche bestimmt werden. Dieses Verfahren kann für den Fall eines Wärmeüberganges von einem Wärmeträger zu einem Biomassepartikel adaptiert werden (Gleichung 2.43 u. Abbildung 2.26).

$$\alpha_{i,j_S} = \sum \frac{\lambda_{N_2}}{\max(s_k, 0) + l} \cdot \bar{A}_k$$

$$\text{mit } l = 2 \frac{2 - \gamma}{\gamma} \sqrt{\frac{2\pi \cdot T_{N_2} \cdot R}{M_{N_2}}} \cdot \frac{\lambda_{N_2}}{p \cdot \left(2 \cdot c_{P_{N_2}} - \frac{R}{M_{N_2}}\right)}$$

$$\text{mit } \gamma = \frac{1}{10^{0.6 - \left(\frac{1000}{T_{N_2}} + 1\right)/2.8}} + 1$$

dann ist

$$\dot{Q}_{i,j} = \alpha_{i,j_S} \cdot (T_i - T_j)$$

$$R = 8.314 \frac{J}{K \cdot mol}$$

$$T_{N_2} = 500^\circ C$$

$$p = 1 \text{ bar}$$

$$c_{P_{N_2}}(T_{N_2}) = 1120 \frac{J}{K \cdot kg} \quad (2.43)$$

$$\lambda_{N_2}(T_{N_2}) = 0.045 \frac{W}{m \cdot K}$$

$$M_{N_2} = 28.96 \frac{g}{mol}$$

Beim Wärmeübergang nach Schlünder, wird der Wärmeübergang durch die Austauschfläche  $\bar{A}$  (Abbildung 2.26, rot und blau) und dem Partikelabstand  $s$  zwischen Partikel  $i$  und  $j$ , sowie deren lineare Grenzschichtdicke  $l$  betrachtet. Letzterer setzt sich zusammen aus den Rahmenbedingungen, den Stoffeigenschaften, und dem Akkomodationskoeffizient  $\gamma$ .

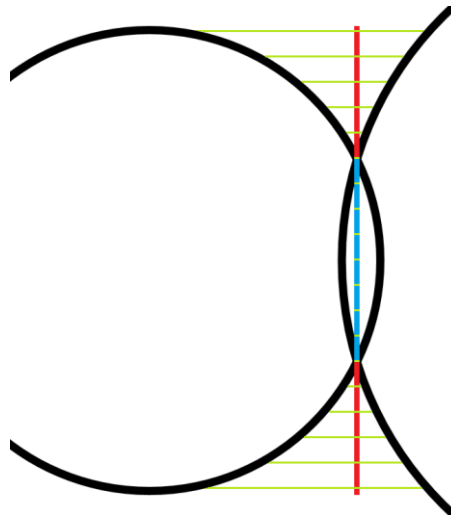


Abbildung 2.26: Abstandsbestimmung zwischen zwei Partikeln nach Schlünder. Die Projektionsfläche ist blau markiert, der Wärmeübergangsbereich rot mit grünen Abstandslinien  $s_k$ .

Da der Abstand  $s_k$  zwischen den Partikeln eines Flächenelementes aufgrund der Überlappung negativ sein kann, wurden in diesem Fall der Wert für dieses Flächenelement auf Null gesetzt (Abbildung 2.26). Zusätzlich muss für die Wandfläche zwischen zwei Partikel  $i$  und  $j$  gelten:

$$\bar{A}_{i,j} = \pi \cdot \min(r_i, r_j)^2 \quad (2.44)$$

Eine Berechnung des Wärmeüberganges nach Schlünder (Gleichung 2.43) hat den wesentlichen Vorteil, dass die Berechnung nicht auf eine Überlappung angewiesen ist (Abbildung 2.27). Somit kann der Wärmeübergang auch bei sich nichtberührenden Partikeln beobachtet werden.

## Temperaturverlauf beim Wärmeübergang nach Schlünder von Wärmeträgerpartikel nach Biomassepartikel

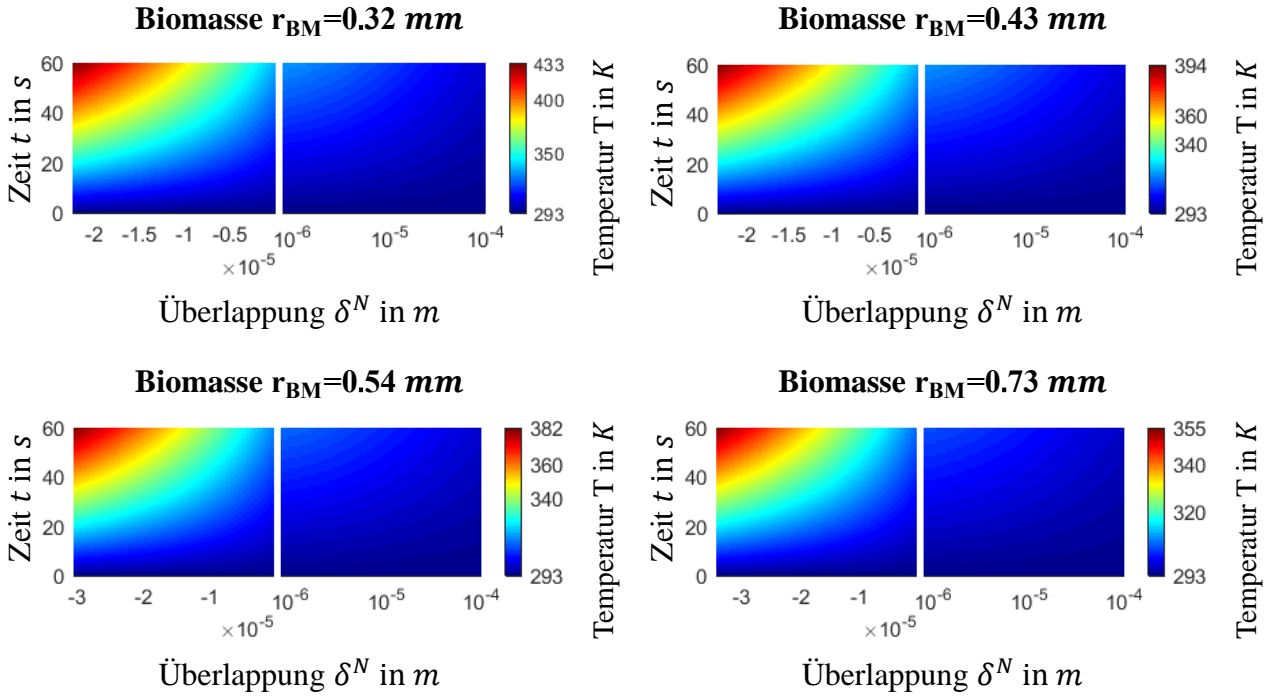


Abbildung 2.27: Temperaturverlauf beim Wärmeübergang nach Schlünder. Links der Fall der Überlappung der Partikel, rechts effektiver Abstand zwischen den Oberflächen

Im Vergleich mit dem Wärmeübergang nach Batchelor (Abbildung 2.25), weist der Wärmeübergang nach Schlünder eine bis zum Faktor 10 höhere Temperaturanstieg auf und selbst mit positivem Abstand kann eine höhere Temperatur beobachtet werden. Dieser Temperaturverlauf spiegelt Beobachtungen besser wider als es Batchelor vermag [84].

Eine naheliegende Schlussfolgerung, um beide Wärmeübergänge anzugleichen, ist die Anpassung des Wärmeleitkoeffizienten  $\lambda$ .

$$\alpha_{i,j_{B\sim R}} = 4 \cdot \frac{\lambda_i \cdot \lambda_j}{\lambda_i + \lambda_j} \cdot \sqrt{A_{i,j}} \quad (2.45)$$

Der Wärmeübergang nach Batchelor (Gleichung 2.45, [62]) ist proportional zum Radius der Projektionsfläche  $A_{i,j}$ , während der Wärmeübergang nach Schlünder proportional zur Fläche  $\bar{A}_{i,j}$  ist. Um die Vergleichbarkeit zu gewährleisten, wurde zusätzlich der Wärmeübergangskoeffizient so angepasst, dass er proportional zur Projektionsfläche  $A_{i,j}$  ist (Gleichung 2.46).

$$\alpha_{i,j_{B\sim A}} = 4 \cdot \frac{\lambda_i \cdot \lambda_j}{\lambda_i + \lambda_j} \cdot A_{i,j} \quad (2.46)$$

Da der mittlere Wärmeleitkoeffizient nach Batchelor für meine Material/Material-Interaktion konstant ist, weicht der Wärmeübergangskoeffizient  $\alpha_{i,j_{B\sim A}}$  um einen konstanten Faktor  $c_\alpha$  vom realen Wert ab. Man beachte, die unterschiedlichen Einheiten von  $\alpha_{i,j_{B\sim A}}$  und  $\alpha_{i,j_{B\sim R}}$ . Im weiteren Verlauf gehen wir stillschweigend davon aus, dass die charakteristische Weglänge im Korrekturfaktor  $c_\alpha$  enthalten ist.

## Vergleich der Wärmeübergangskoeffizienten

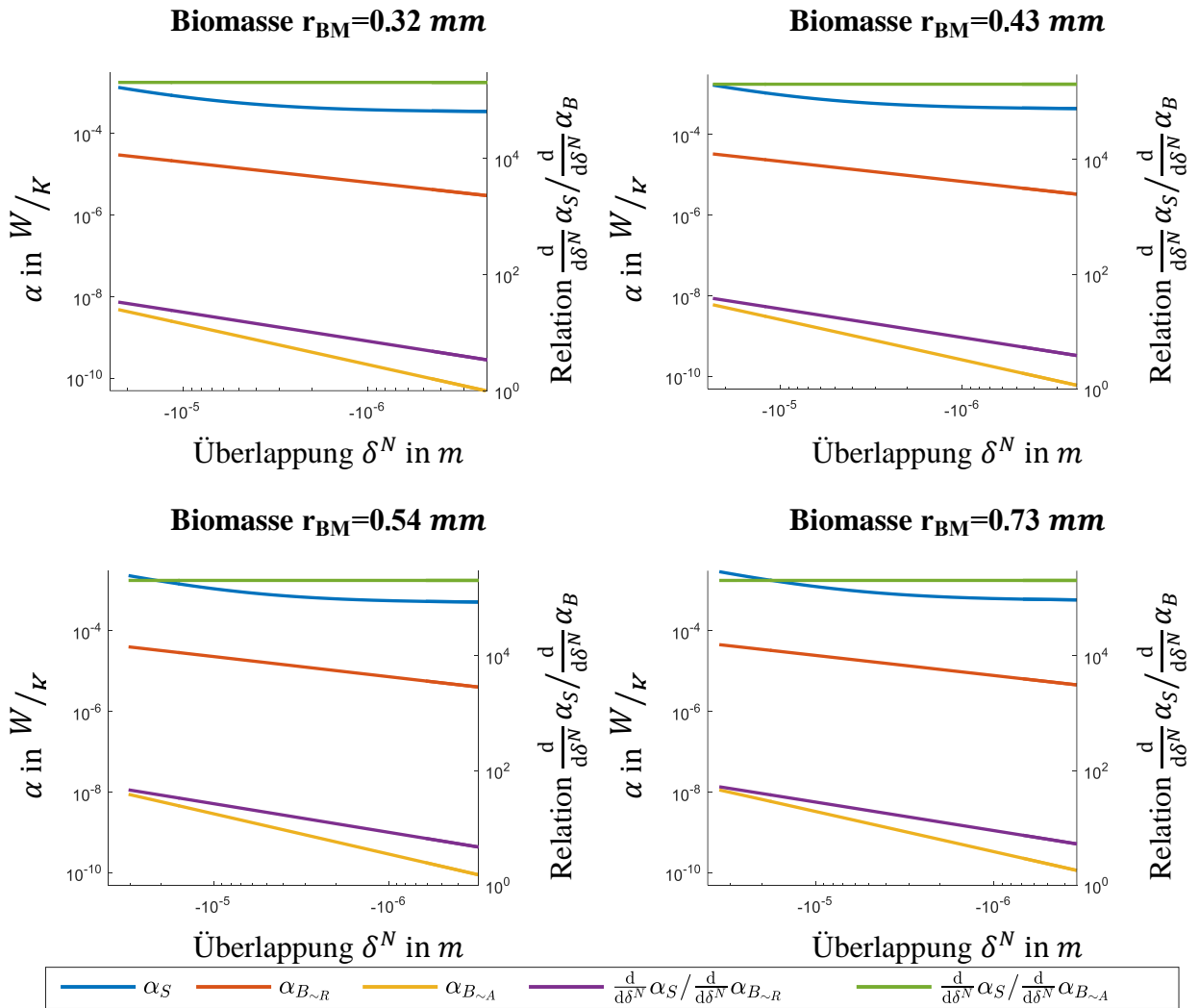


Abbildung 2.28: Vergleich der Wärmeübergangskoeffizienten.  
 Links: Verlauf des Wärmeübergangskoeffizienten entlang des Abstandes,  
 rechts: Vergleich der Wärmeübergangskoeffizienten Batchelor und Schlünder

Ein Vergleich der Wärmeübergangskoeffizienten nach Batchelor und Schlünder belegt die vermutete Abweichung des Wärmeüberganges (Abbildung 2.25 u. Abbildung 2.27). Ein Vergleich der Steigungen von  $\alpha_S$  und  $\alpha_{i,j_{B \sim A}}$  zeigt eine Konvergenz. Daher kann nach dem Hauptsatz der Differential- und Integralrechnung [85]  $\alpha_S$  durch  $\alpha_{i,j_{B \sim A}}$  rekonstruiert werden. D.h. es gilt:

$$\alpha_S \approx \alpha_{B_{\text{Mod}}} = \alpha_{B \sim A} \cdot c_\alpha + k_\alpha \quad (2.47)$$

mit

$$c_\alpha = \frac{\frac{d}{d\delta^N} \alpha_S}{\frac{d}{d\delta^N} \alpha_{B \sim A}} \quad (2.48)$$

$$k_\alpha = \alpha_S \quad \text{für } A_{i,j} = 0 \text{ m}^2$$

Die Koeffizienten  $c_\alpha$  und  $k_\alpha$  wurden für die Biomasse/Wärmeträger-Interaktionen bestimmt und der Mittelwert sowie die Standardabweichung gebildet (Tabelle 2.25).

Tabelle 2.25: Korrekturkoeffizienten für die Biomasse/Wärmeträger-Wärmeübergangskoeffizienten

	$c_\alpha$	$k_\alpha$
$r_{BM}=0.32 \text{ mm}$	$5.3 \cdot 10^4$	$3.30 \cdot 10^{-4}$
$r_{BM}=0.43 \text{ mm}$	$5.32 \cdot 10^4$	$4.27 \cdot 10^{-4}$
$r_{BM}=0.54 \text{ mm}$	$5.32 \cdot 10^4$	$4.96 \cdot 10^{-4}$
$r_{BM}=0.73 \text{ mm}$	$5.32 \cdot 10^4$	$5.58 \cdot 10^{-4}$
<b>Mittelwert</b>	<b><math>5.31 \cdot 10^4</math></b>	<b><math>4.52 \cdot 10^{-4}</math></b>
<b>Standardabw. %</b>	<b>0.2</b>	<b>18.6</b>

Unter Verwendung von (Gleichung 2.47) kann eine Annäherung erzielt werden, die sich nur geringfügig von dem nach Schlünder bestimmten Wärmeübergangskoeffizienten unterscheidet (Abbildung 2.29).

### Vergleich des Wärmeübergangskoeffizienten für den Biomasse/Wärmeträger-Kontakt

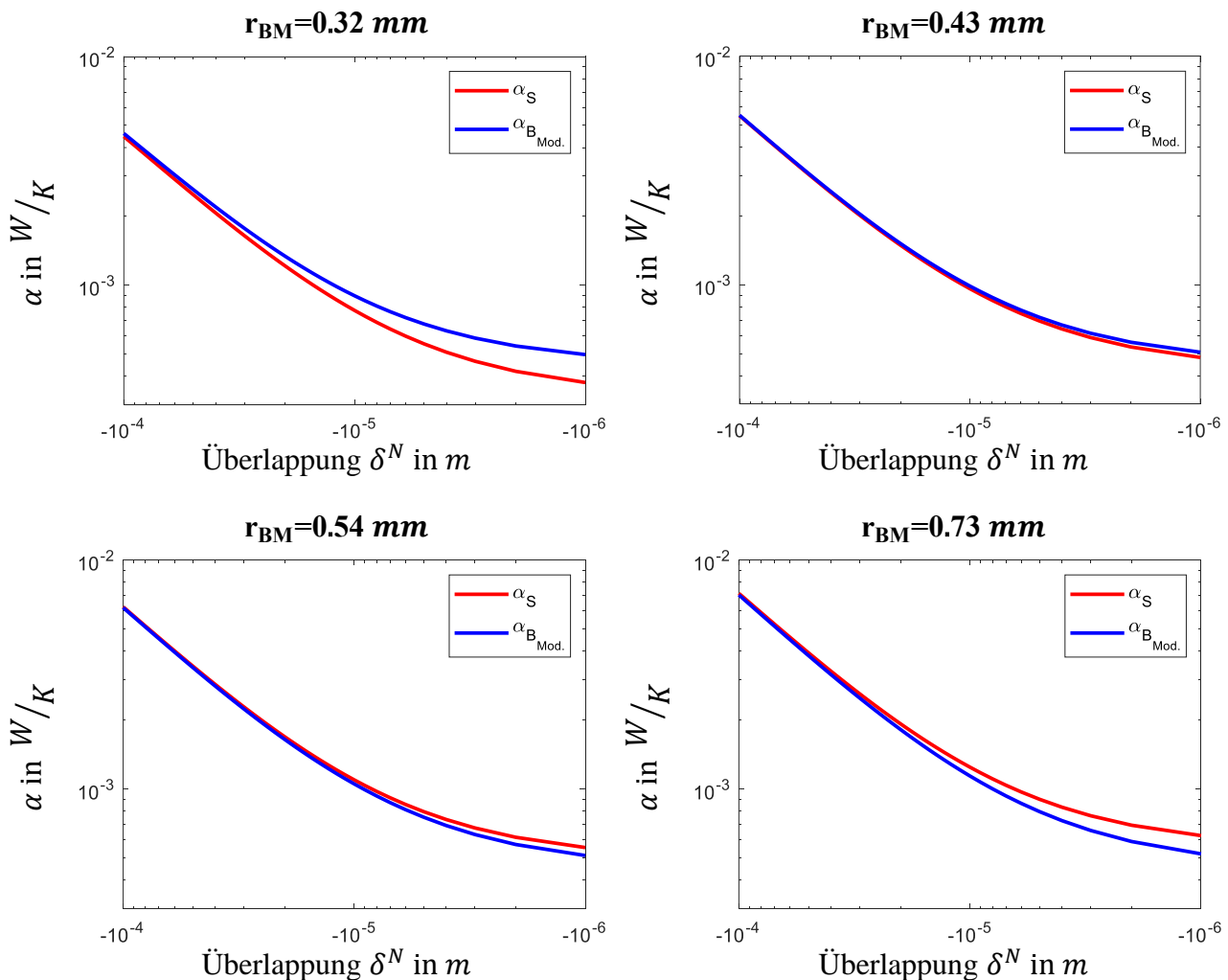


Abbildung 2.29: Vergleich der Wärmeübergangskoeffizienten für den Biomasse/Wärmeträger-Kontakt nach Schlünder und nach Batchelor

Allgemein ist die Übereinstimmung als gut zu bewerten. Der Verlauf stimmt weitestgehend überein, während Abweichungen durch den konstanten Term  $k_\alpha$  (Tabelle 2.25) zu erklären sind. Unabhängig davon wird der Wärmeübergang mit einer sehr guten Annäherung dargestellt (Abbildung 2.30), sodass augenscheinlich kein Unterschied zwischen dem Wärmeübergang nach Schlünder und dem modifizierten nach Batchelor, zu erkennen ist.

### Vergleich des Wärmeüberganges des modifizierten Batchelor-Modell und Schlünder

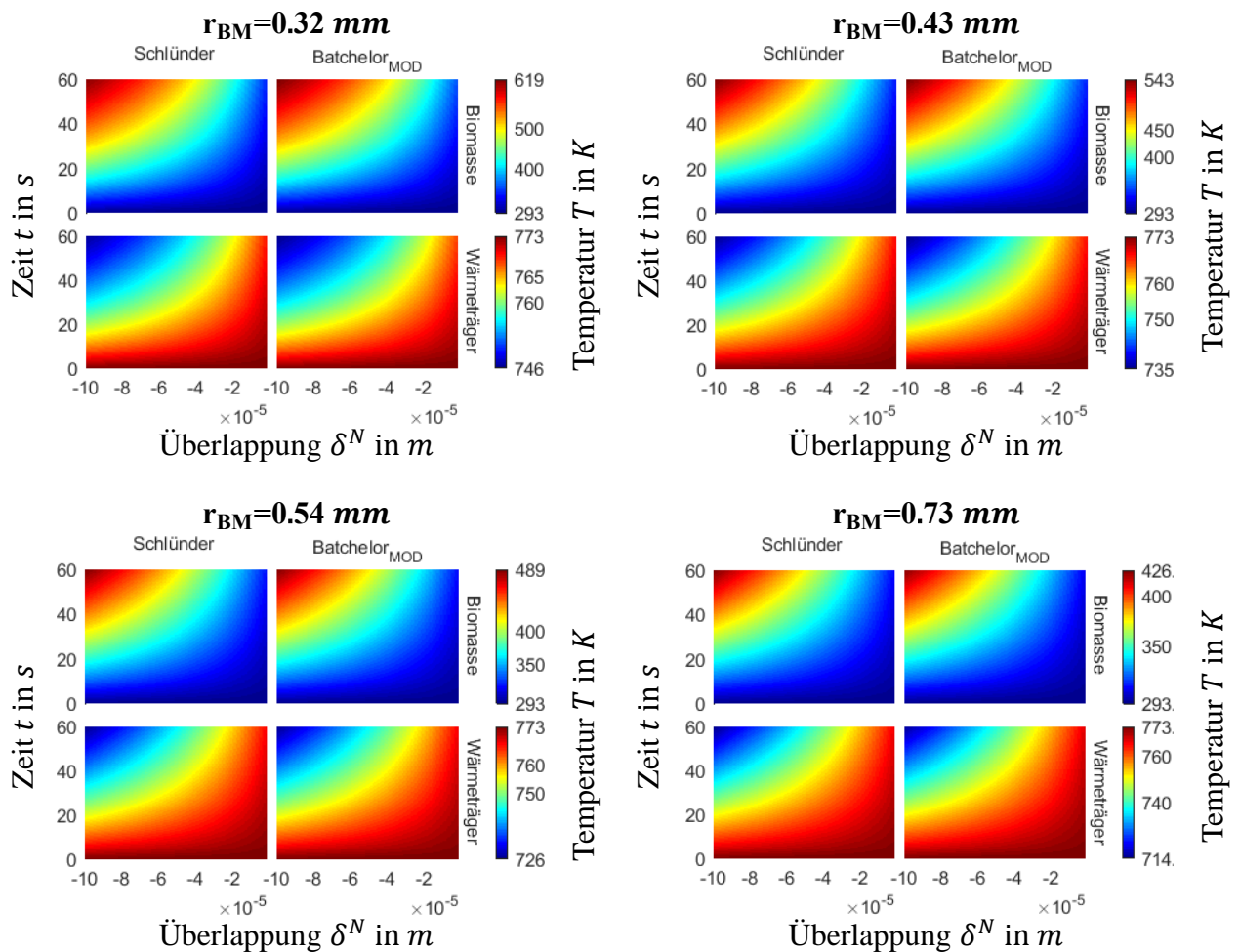


Abbildung 2.30: Direkter Vergleich des Wärmeüberganges. Biomassepartikel oben, Wärmeträger unten. Wärmeübergang nach Schlünder links, nach korrigierten Batchelor-Modell rechts

Analog zum Wärmeträger/Biomasse-Wärmeübergang wurde der Wärmeträger/Wärmeträger-Wärmeübergang betrachtet. Wie schon zuvor, wurde der Wärmeübergangskoeffizient  $\alpha_s$  nach (Gleichung 2.43) berechnet. Dem gegenüber steht der Wärmeübergangskoeffizient  $\alpha_{B_{MOD}}$ , der über die (Gleichung 2.46 u. Gleichung 2.47) berechnet wurde (Abbildung 2.31).

### Vergleich des Wärmeübergangskoeffizienten für den Wärmeträger/Wärmeträger-Kontakt

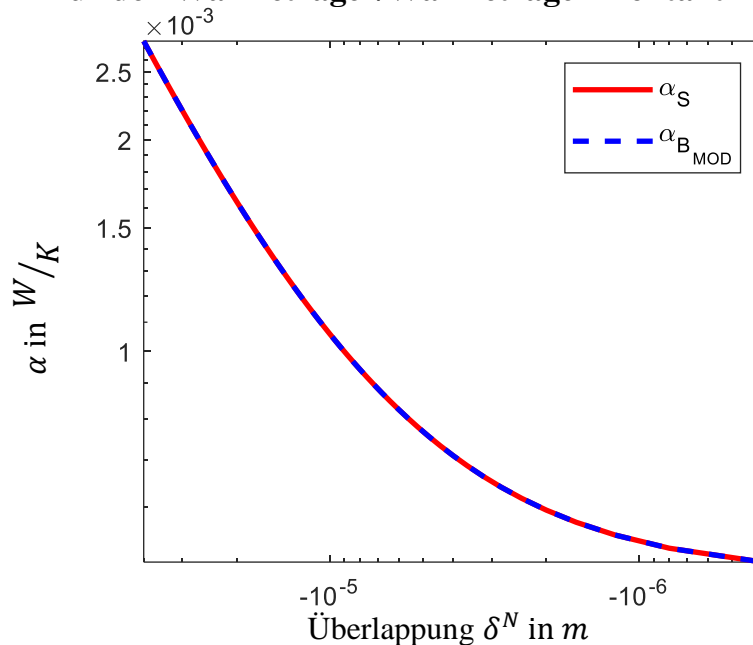


Abbildung 2.31: Vergleich der Wärmeübergangskoeffizienten für den Wärmeträger/Wärmeträger-Kontakt nach Schlünder und nach modifizieren Batchelor-Model

Da die Wärmeträger nur eine einheitliche Größe aufweisen, ist die Übereinstimmung der Wärmeübergangskoeffizienten, um ein Vielfaches besser, sodass die Korrekturkoeffizienten (Tabelle 2.26) eine gute Übereinstimmung liefern (Abbildung 2.31).

Tabelle 2.26: Korrekturkoeffizienten für den Wärmeträger/Wärmeträger-Wärmeübergangskoeffizienten

$c_\alpha$	$k_\alpha$
463.64	0.0004782

Dem gegenüber steht die Auswertung des Biomasse/Biomasse-Wärmeüberganges (Abbildung 2.32). Aufgrund der großen Vielfalt, der Interaktionen von vier Biomasse-Partikeltypen, kommt es zu einer größeren Abweichung. Es muss an dieser Stelle erwähnt werden, dass die Anzahl des Wärmeträgerpartikel 2-3 Mal der, der Biomassepartikel entspricht. Daher ist die Anzahl der Biomasse/Biomasse-Kontaktstellen um ein Vielfaches geringer. Umso geringer ist die Anzahl der Biomassekontaktstellen der einzelnen Partikeltypen, sodass zum Teil kein Kontakt der Partikeltypen zu finden war. Aufgrund von fehlenden Daten, wurde der Biomasse/Biomasse-Wärmeübergang für den Überlappungsbereich des Wärmeträger/Wärmeträger-Wärmeübergangs validiert. Zu erwarten ist jedoch, aufgrund des niedrigeren E-Moduls von Biomasse, eine stärkere Überlappung mit einem höheren Wärmeübergangskoeffizienten.



### Vergleich des Wärmeübergangskoeffizienten für den Biomasse/Biomasse-Kontakt

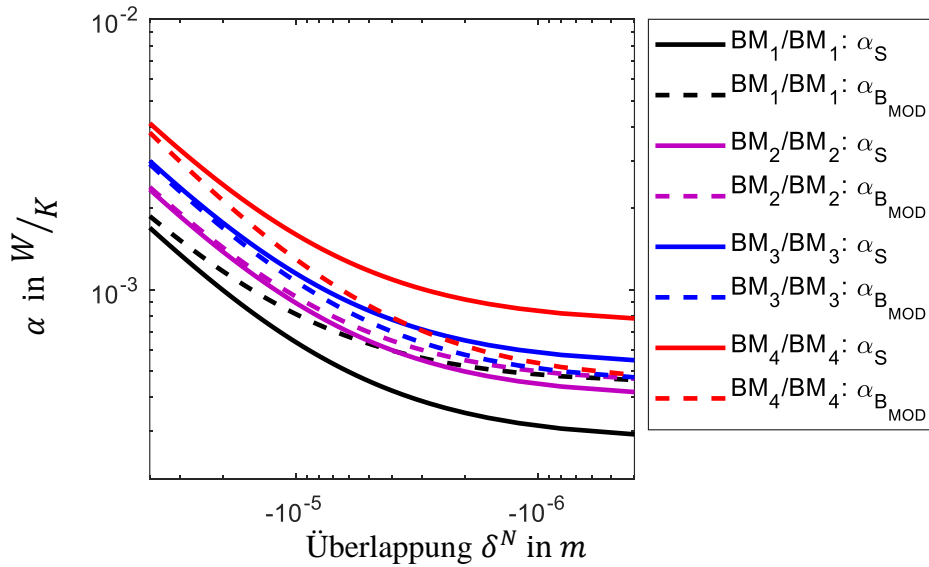


Abbildung 2.32: Vergleich der Wärmeübergangskoeffizienten für den Biomasse/Biomasse-Kontakt identischer Größe nach Schlünder und nach modifizierten Batchelor-Model

Der Übersicht wegen, wurde in (Abbildung 2.32) nur der Biomasse/Biomasse-Kontakt mit identischen Biomasse Radien der beteiligten Komponenten betrachtet. Es muss jedoch erwähnt werden, dass (Abbildung 2.29) Abweichungen proportional zur Differenz der Partikelradien sind. Wie in vorherigen Auswertungen, zeigt der konstante Multiplikator  $c_\alpha$  (Tabelle 2.27) nur geringe Abweichungen, wofür der konstante Term  $k_\alpha$  für Abweichungen aus (Abbildung 2.32) verantwortlich zu machen ist.

Tabelle 2.27: Korrekturkoeffizienten für die Biomasse-Biomasse-Wärmeübergangskoeffizienten.

	$c_\alpha$	$k_\alpha$
<b>BM1/BM1</b>	$108.6 \cdot 10^3$	$2.77 \cdot 10^{-4}$
<b>BM1/BM2</b>	$108.6 \cdot 10^3$	$3.14 \cdot 10^{-4}$
<b>BM1/BM3</b>	$108.9 \cdot 10^3$	$3.99 \cdot 10^{-4}$
<b>BM1/BM4</b>	$108.5 \cdot 10^3$	$3.39 \cdot 10^{-4}$
<b>BM2/BM2</b>	$109.0 \cdot 10^3$	$4.40 \cdot 10^{-4}$
<b>BM2/BM3</b>	$109.2 \cdot 10^3$	$5.23 \cdot 10^{-4}$
<b>BM2/BM4</b>	$108.4 \cdot 10^3$	$3.67 \cdot 10^{-4}$
<b>BM3/BM3</b>	$108.9 \cdot 10^3$	$4.88 \cdot 10^{-4}$
<b>BM3/BM4</b>	$109.2 \cdot 10^3$	$5.94 \cdot 10^{-4}$
<b>BM4/BM4</b>	$109.4 \cdot 10^3$	$7.49 \cdot 10^{-4}$
<b>Mittelwert</b>	$108.9 \cdot 10^3$	$4.49 \cdot 10^{-4}$
<b>Standardabw. %</b>	0.29	30.48

Ein wesentlicher Vorteil des modifizierten Batchelor Modelles ist der Rechenaufwand, der für die Berechnung erforderlich ist. Während die Berechnung nach Schlünder, abhängig von der Diskretisierung von  $\bar{A}_{i,j}$ , mehrere Minuten dauern kann, ist der Wärmeübergang nach dem modifizierten Batchelor in wenigen Schritten berechnet. Ein wesentlicher Nachteil, den die bisherigen Modelle haben, ist die Abhängigkeit von der Überlappung  $\delta^N$  und deren Anhängigkeit

beim verwendeten E-Modul. Da bei den DEM-Simulationen i.A. ein wesentlich kleineres E-Modul eingesetzt wird, ist die Überlappung stärker als bei der Verwendung von Literaturwerten. Eine Möglichkeit, dem entgegenzuwirken, bietet die Korrektur der Überlappung  $\delta_{i,j}^N$  (Gleichung 2.49).

$$\begin{aligned}\delta_{i,j\text{Org}}^N &= \delta_{i,j}^N \cdot \frac{\delta_{i,j\text{Org}}^N}{\delta_{i,j}^N} = \delta_{i,j}^N \cdot \frac{\left(\frac{3 \cdot F_{i,j}^N \cdot r_{i,j}}{4 \cdot E_{i,j\text{Org}}}\right)^{\frac{2}{3}}}{\left(\frac{3 \cdot F_{i,j}^N \cdot r_{i,j}}{4 \cdot E_{i,j}}\right)^{\frac{2}{3}}} \\ &= \delta_{i,j}^N \cdot \underbrace{\left(\frac{E_{i,j}}{E_{i,j\text{Org}}}\right)^{\frac{2}{3}}}_{<1}\end{aligned}\quad (2.49)$$

Dabei ist es von Nutzen, dass sich bei der Berechnung der normalen Überlappung nach Hertz [40] (Gleichung 2.50) alle Terme, bis auf die E-Moduln kürzen.

$$\delta_{i,j}^N = \left(\frac{3 \cdot F_{i,j}^N \cdot r_{i,j}}{4 \cdot E_{i,j}}\right)^{\frac{2}{3}} \quad (2.50)$$

Bei Verwendung von  $\delta_{i,j\text{Org}}^N$  in (Gleichung 2.42) kann ein Wärmeübergang berechnet werden, der nur von Literaturwerten für das E-Modul abhängt.

Die bisher diskutierten Modelle für den Wärmeübergang, gehen von einer homogenen Verteilung der Wärme im Partikel aus. Dies kann jedoch pauschal nicht für alle Partikel angenommen werden. Weizenstroh, bekannt für seine schlechte Wärmeleitung, wird daher häufig als Isolationsmaterial eingesetzt. Die ablative Pyrolyse arbeitet nach diesem Prinzip. Biomasse wird an eine heiße Oberfläche gepresst. Aufgrund der schlechten Wärmeleitung findet die Pyrolyse nur an der Kontaktstelle statt [86, 87]. Bei dieser Art der Pyrolyse kommen Partikel zum Einsatz, deren Größe sich erheblich zu den hier verwendeten Weizenstrohpartikeln unterscheidet. Dennoch wirft dies die Frage auf, ob der Wärmeübergang oder die Wärmeleitung dominieren. Die Biot-Zahl (Gleichung 2.51), formal gleich der Nußelt-Zahl gebildet, gibt das Verhältnis aus Wärmeleitung und Wärmeübergang in einem Feststoff wieder.

$$Bi = \frac{\alpha_{BMOD} \cdot d_p}{\lambda_p} \quad (2.51)$$

$\lambda_p$  entspricht den minimalen Wärmeleitfähigkeitskoeffizienten der beteiligten Partikel und  $d_p$  dessen charakteristischem Durchmesser. Für Biomasse ist dies gleich der Dicke eines Biomassepartikels ( $1.5 \cdot 10^{-4} \text{ m}$ , Kapitel 2.3.2) und für den Wärmeträger, dessen Durchmesser. Die Projektionsfläche  $\bar{A}$  entspricht der Wärmeübergangsfläche zur Berechnung des Wärmeübergangskoeffizienten nach Schlünder (Gleichung 2.44).

*Tabelle 2.28: Biot-Zahlen für die Partikel/Partikel-Interaktionen*

	<b>WT-WT</b>	<b>BM-BM</b>	<b>BM-WT</b>
$\alpha_{B_{Mod}}/\bar{A}$	$5.36 \cdot 10^{-4}$	$4.8 \cdot 10^{-4}$	$5.1 \cdot 10^{-4}$
$\lambda_P$	40	0.17	0.17
$d_P$	$5 \cdot 10^{-4}$	$1.5 \cdot 10^{-4}$	$1.5 \cdot 10^{-4}$
<b>Bi</b>	$1.0 \cdot 10^{-3}$	1.4	0.78

Zur Berechnung der Biot-Zahlen (Tabelle 2.28) wurde jeweils der maximale Wärmeübergangskoeffizient im validierten Bereich gewählt. Bei einer Biot-Zahl von 0.78 bewegen wir uns in einem Bereich, in dem sowohl der Wärmeübergang, als auch die Wärmeleitung im Partikel eine wichtige Rolle spielen. Jedoch ist bei einer Biot-Zahl kleiner eins anzunehmen, dass der Wärmeübergang den marginal wichtigeren Effekt darstellt. In der Tat ist zum jetzigen Stand der Technik die Berücksichtigung der Wärmeleitung im Partikel bei gegebener Partikelzahl zu rechenintensiv. Dies wird in weiteren Arbeiten thematisiert.

### 3 Mischverhalten im Doppelschneckenmischreaktor

Bisher beschäftigte sich diese Arbeit mit der Bestimmung, Abschätzung und Validierung von Kennwerten und Modellen die bei der DEM-Simulation erforderlich sind. Das Ziel ist durch die Wahl der Kennwerte physikalische Vorgänge mit höherer Genauigkeit zu simulieren und die dafür erforderliche Rechenzeit zu minimieren. Eine DEM-Simulation in Kombination mit dem erarbeiteten Know-How ermöglicht eine Betrachtung des Mischverhaltens und des Wärmeübergangs bei variierenden Betriebsparametern und Geometrien, zwei der wichtigsten Einflussgrößen bei der Pyrolyse von Biomasse. Der Fokus liegt hierbei auf den Partikel-Partikel-Wärmeübergang. Der Partikel-Wand-Wärmeübergang konnte aufgrund von sporadisch auftretenden und nicht reproduzierbaren Fehlern in LIGGGHTS® [88] nicht in Betracht gezogen werden, so dass der Fokus auf den Partikel-Partikel-Wärmeübergang liegt. Dies ermöglicht jedoch die Betrachtung des Partikel-Partikel-Wärmeübergangs ohne den Einfluss weiterer Effekte. Dieses Kapitel beschäftigt sich daher mit der Charakterisierung des Partikel-Partikel-Wärmeübergangs, des Mischverhaltens und der Lokalisierung der Problemstellen bei unterschiedlichen Betriebsbedingungen.

Bei der Verwendung von Schnecken erfolgt der Transport eines Partikelbettes durch Abrutschen entlang des Schneckenganges während deren Rotation. Somit hat die Schnecke durch ihre Ganghöhe direkten Einfluss auf das Verhältnis von Transport und Umwälzung. In vielen Fällen kann eine Modifikation der Schneckenganghöhe durch eine Anpassung der Rotationsgeschwindigkeit ausgeglichen werden. Jedoch hat die Umwälzung direkten Einfluss auf die Vermischung. Mit steigender Umwälzung wird die Vermischung besser. Ist diese zu groß, kommt es zu einer Entmischung. Bei rotierenden Werkzeugen wird häufig der Froude-Wert als Referenz herangezogen. Ist  $Fr \ll 1$ , führt eine steigende Umwälzung zu einer besseren Vermischung. Ist der Froude-Wert  $Fr \gg 1$  kann es aufgrund von Dichteunterschieden zu einer Entmischung kommen.

Eine Änderung der Ganghöhe kann bei  $Fr \ll 1$  durch eine Änderung der Einstellung im Hinblick auf Fördergeschwindigkeit oder Mischqualität approximiert werden. Daher wurde dies in dieser Arbeit nicht berücksichtigt. Diese Aussage gilt nicht uneingeschränkt und ist stark fehlerbehaftet, daher ist es im konkreten Fall anzuraten für neue Schneckengeometrien diese zu validieren.

Alle Simulationen wurden, sofern nicht anders angegeben, mit den Kennwerten aus Tabelle 2.24 durchgeführt.

#### 3.1 Wärmeübergang des Doppelschneckenmischreaktors in Abhängigkeit der Rotationsgeschwindigkeit

Bei der Charakterisierung und Optimierung des Doppelschneckenmischreaktors wurden mehrere Betriebszustände simuliert. Die Rotationsgeschwindigkeiten wurden so gewählt, dass ein Wertebereich zwischen 1 Hz ( $Fr \ll 1$ ) und 4 Hz ( $Fr \gg 1$ ) abgedeckt ist.

Dabei konnte ein Rückstau im unteren Drehzahlbereich beobachtet werden. Ein Effekt den auch Frey [70] in seiner Charakterisierung des Doppelschneckenmischreaktors bei Versuchen mit 1 Hz beobachten konnte.

Zur Verifizierung der Simulationen, wurde die Verweilzeit (Tabelle 3.1) aus der axialen Geschwindigkeitsverteilung (Abbildung 3.1) bestimmt und sie Frey gegenübergestellt. Dazu wurde der Reaktor im Mischbereich entlang der Rotationsachse diskretisiert. In jedem Segment  $i$  mit Segmentbreite  $dx_i$  wurde die mittlere Geschwindigkeit  $u_i$  aus der Partikelgeschwindigkeit  $u_{ij}$  ermittelt. Diese wurde aus der Position zweier Zeitschritte bestimmt. Daraus konnte die Segmentverweilzeit  $\tau_i$  und deren Varianz  $\sigma_i^2$  bestimmt werden. Über die Segment-Bodensteinzahl  $Bo_i$  konnte der Segment-Dispersionskoeffizient  $D_{ax_i}$  ermittelt werden. Der globale Dispersionskoeffizient ist der Mittelwert der Segment-Dispersionskoeffizienten (Gleichung 3.1).

$$\begin{aligned}
\tau &= \sum_{i=1} \tau_i \text{ mit } \tau_i = \frac{dx_i}{n_i} \cdot \sum_{j=1} \frac{1}{u_{i,j}} \\
\frac{\sigma_i^2}{\tau_i} &= \frac{2}{Bo_i} + \frac{8}{Bo_i^2} \text{ mit } \sigma_i^2 = \frac{1}{n_i} \cdot \sum_{j=1} \left( \tau_i - \frac{dx_i}{u_{i,j}} \right)^2 \\
D_{ax} &= \frac{1}{n} \cdot \sum_{i=1} D_{ax_i} \text{ mit } D_{ax_i} = \frac{u_i \cdot dx_i}{Bo_i} \text{ und } u_i = \frac{1}{n_i} \cdot \sum_{j=1} u_{i,j}
\end{aligned} \tag{3.1}$$

Tabelle 3.1: Vergleich der Verweilzeitverteilung Simulation vs. Versuch

	Simulation		Frey[70]	
	$\tau$ [s]	$D_{ax}$ [ $\frac{mm^2}{s}$ ]	$\tau$ [s]	$D_{ax}$ [ $\frac{mm^2}{s}$ ]
4.0 Hz	6.6	532.1	5.9	1050
3.5 Hz	7.1	455.9		
3.0 Hz	7.7	391.1	7.2	1534
2.5 Hz	8.6	322.7		
2.0 Hz	9.8	262.4	10.2	440

Dieser Ansatz erlaubt uns die Ergebnisse zu validieren, während er uns gleichzeitig wichtige Informationen zu Transportvorgängen liefert. Alle ermittelten Werte wurden noch einmal über eine Sekunde Simulationszeit gemittelt.

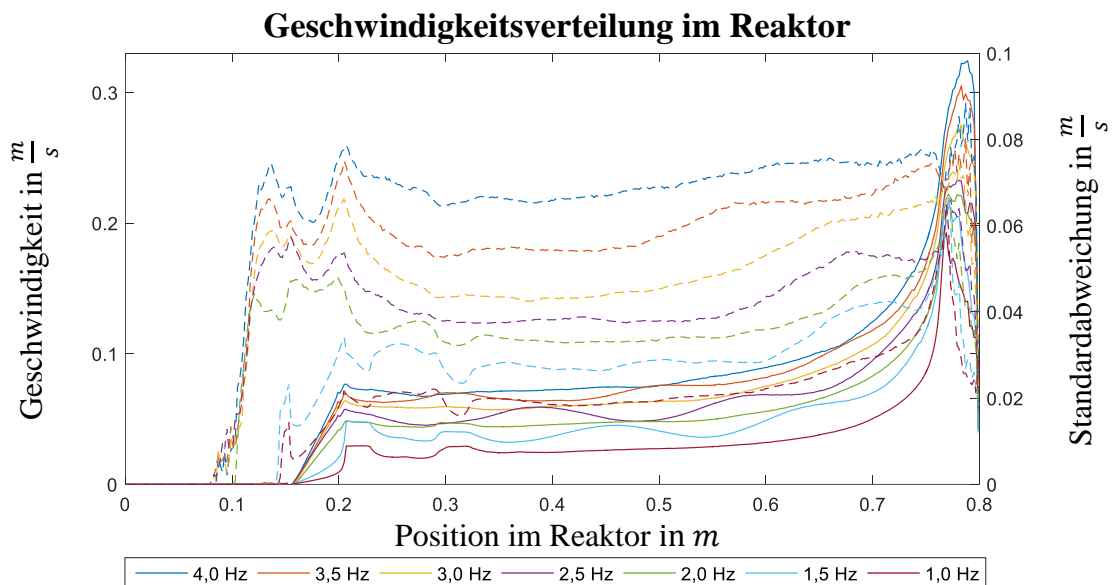


Abbildung 3.1: Wärmeträgergeschwindigkeitsverteilung im Doppelschneckenmischreaktor in Abhängigkeit der Rotationsgeschwindigkeit. Durchgezogene Linien entsprechen der mittleren Geschwindigkeit im Segment, die gestrichelten Linien deren Standardabweichung.

Betrachten wir die Geschwindigkeitsverteilung (Abbildung 3.1), kann im Allgemeinen eine weitestgehend konstante Geschwindigkeit im Reaktor beobachtet werden. Lediglich im Endbereich steigt die Geschwindigkeit sprunghaft an. Ein Abfallen der Geschwindigkeit bei 1 und 1.5 Hz ist Beleg für eine Verblockung an der Wärmeträgerzugabestelle und kann auch bei der Mischgüte (Abbildung 2.19) beobachtet werden.

## Mischgüte in Abhängigkeit der Rotationsgeschwindigkeit

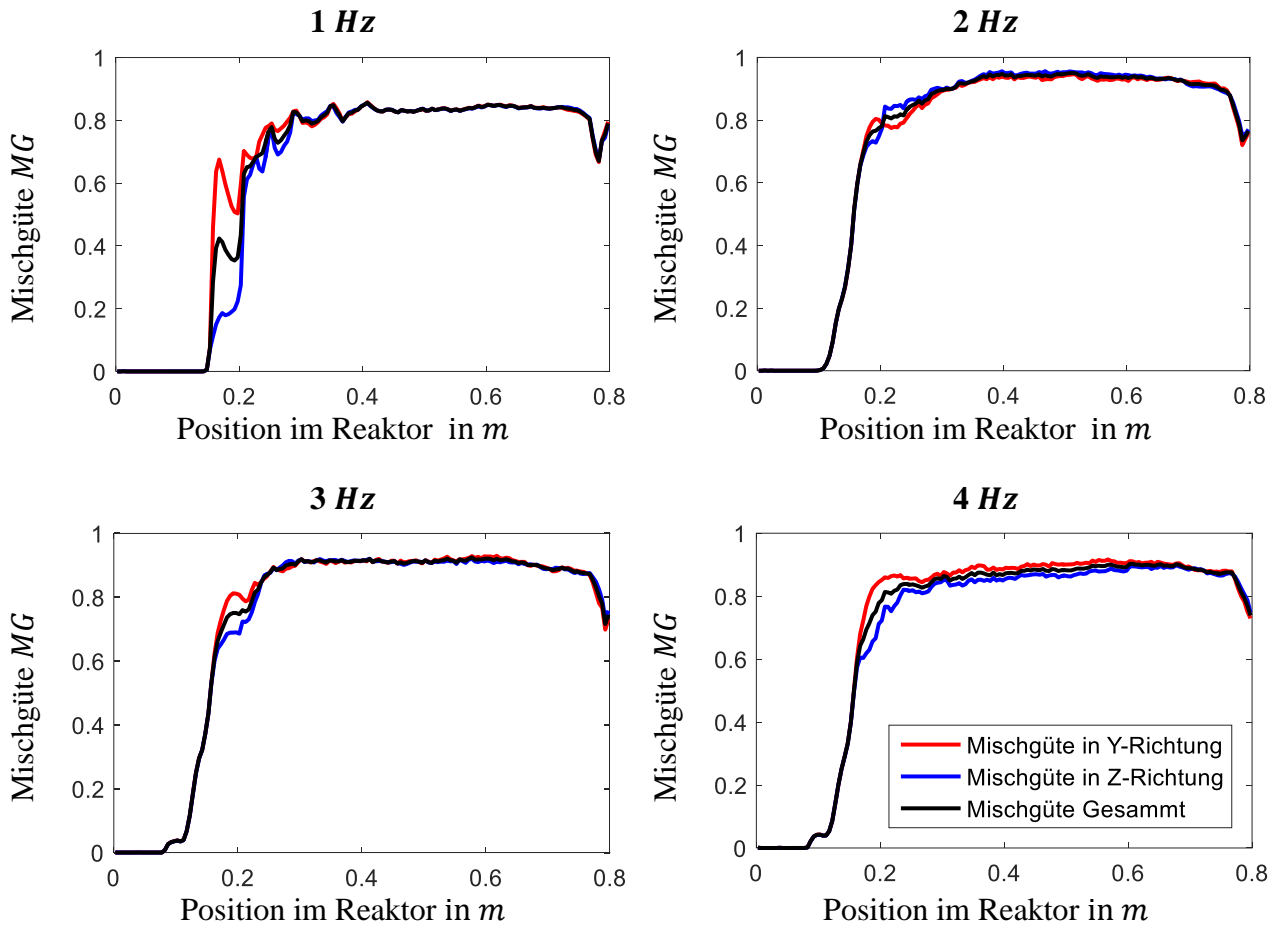


Abbildung 3.2: Verlauf der Mischgüte bezogen auf die Reaktorposition

Zur Untersuchung weiterer Vorgänge im Reaktor wurde die Mischgüte analog zu Abbildung 2.21 in Abbildung 3.2 bestimmt. Es wurde sowohl die horizontale Mischgüte als auch die vertikale Mischgüte berechnet. Sie unterscheiden sich anhand der Diskretisierung und bieten den Vorteil, dass sowohl horizontale als auch vertikale Misch- und Entmischvorgänge abgebildet werden können.

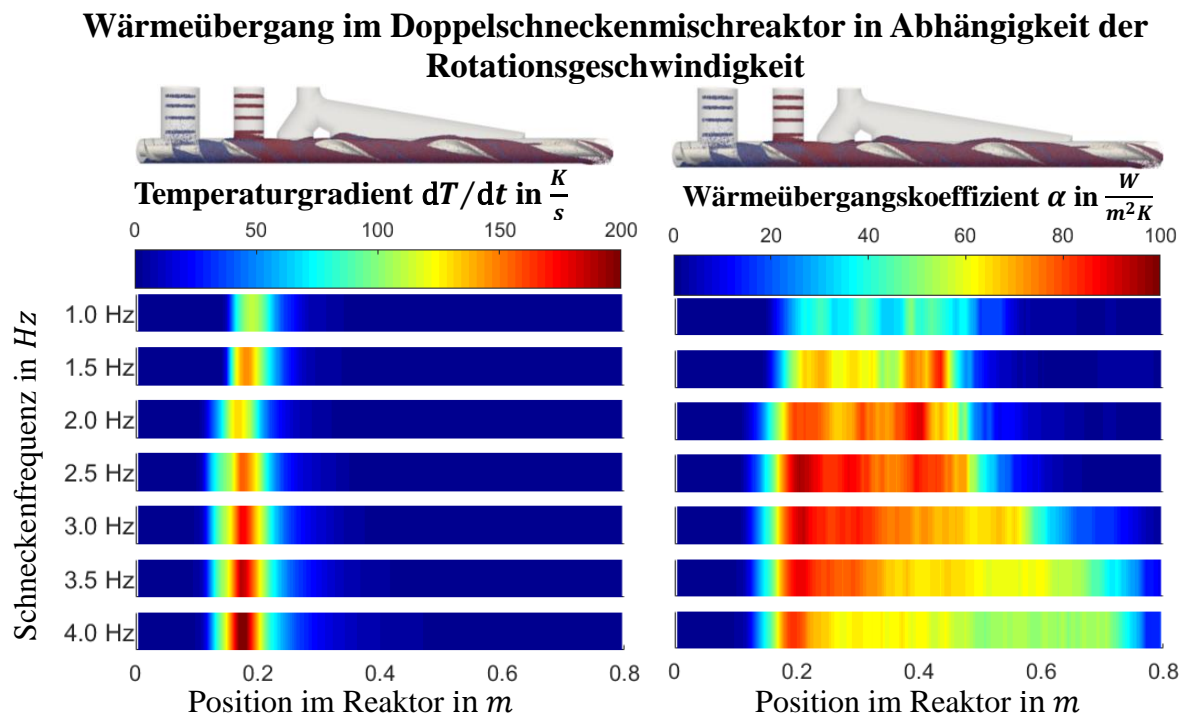
In diesem Fall konnte jedoch gezeigt werden, dass sowohl die horizontale als auch vertikale Vermischung sehr gut ist. Leichte Abweichungen sind nur im Wärmeträger-Zufuhr-Bereich zu erkennen. Die optimale Mischgüte wird im Bereich von 2-3 Hz erreicht. Eine Entmischung kann lediglich im Bereich des Auslasses beobachtet werden.

Zur weiteren Evaluierung wurde der mittlere Temperaturgradient der Biomasse  $dT/dt$  (aus zwei Zeitschritten) und der Wärmeübergangskoeffizient  $\alpha_i$  nach Gleichung 3.2 bestimmt. Die erforderlichen Werte für Biomasse- und Wärmeträgertemperatur ( $\overline{T_{BM_i}}$  u.  $\overline{T_{WT_i}}$ ) wurden für das Segment volumengemittelt berechnet. Aufgrund der Diskretisierung kommt es bei zu kleiner Temperaturdifferenz zwischen Biomasse und Wärmeträger zu unrealistischen Sprungstellen beim Wärmeübergangskoeffizienten daher wurde zur Stabilisierung die Nebenbedingung  $\overline{T_{WT_i}} - \overline{T_{BM_i}} \geq 0.1 \text{ K}$  gewählt.

$$c_p \cdot m \cdot \frac{dT}{dt} = \alpha \cdot A \cdot (T_{WT} - T_{BM})$$

$$\alpha = \frac{c_p \cdot m \cdot \frac{dT}{dt}}{A \cdot (T_{WT} - T_{BM})} \rightarrow \alpha_i = \frac{c_p \cdot \sum_j \left( m_{i_j} \cdot \frac{dT_{i_j}}{dt} \right)}{(\overline{T_{WT_i}} - \overline{T_{BM_i}}) \cdot \sum_j A_{i_j}} \quad (3.2)$$

Die Auswertung (Abbildung 3.3) zeigt einen kontinuierlichen Anstieg des Biomasse-Temperaturgradienten bei ansteigender Drehgeschwindigkeit. Jedoch geht der höhere mittlere Wärmeübergang mit einer größeren Streuung einher. D.h. ein Großteil der Biomassepartikel erlebt einen höheren Wärmeübergang, dafür gibt es zahlreiche Biomassepartikel, die erst spät die Solltemperatur erreichen. Dies führt dazu, dass der Wärmeübergang spät oder gar nicht abgeschlossen ist und wird durch die Auswertung des Wärmeübergangskoeffizienten bestätigt (Abbildung 3.3).



*Abbildung 3.3: Biomassetemperaturanstieg und Wärmeübergangskoeffizient im Doppelschneckenmischreaktor bei 1 – 4 Hz.*

Neben der Durchmischung ist der Wärmeträger/Biomasse-Wärmeübergang von der Anzahl der Kontaktstellen und dem Anpressdruck abhängig. Letzterer ist proportional zur Projektionsfläche der Partikelüberlappung. Sowohl die mittlere Anzahl der Kontaktstellen, als auch die mittlere Projektionsfläche wurden bestimmt (Abbildung 3.4). Dabei zeigt sich, dass mit steigender Fluidisierung, sowohl die Anzahl der Biomasse/Wärmeträgerkontaktstellen, als auch deren Kontaktfläche (Gleichung 2.42) abnimmt. Auffällig ist die geringe mittlere Anzahl von rund 2.2-2.4 Kontaktstellen pro Biomassepartikel.

Beim DEM-Wärmeübergang wird ein direkter Kontakt von Wärmeträger und Biomasse benötigt. Daher ist dieser Zustand des Partikelbettes nachteilig. Bei einer Partikelverteilung von zwei Teilen Wärmeträger und einem Teil Biomasse deutet dies entweder auf eine schlechte Durchmischung hin (was im Widerspruch zu Abbildung 3.2 steht) oder auf ein stark amorphes Partikelbett.

## Kontaktanalyse der Biomasse

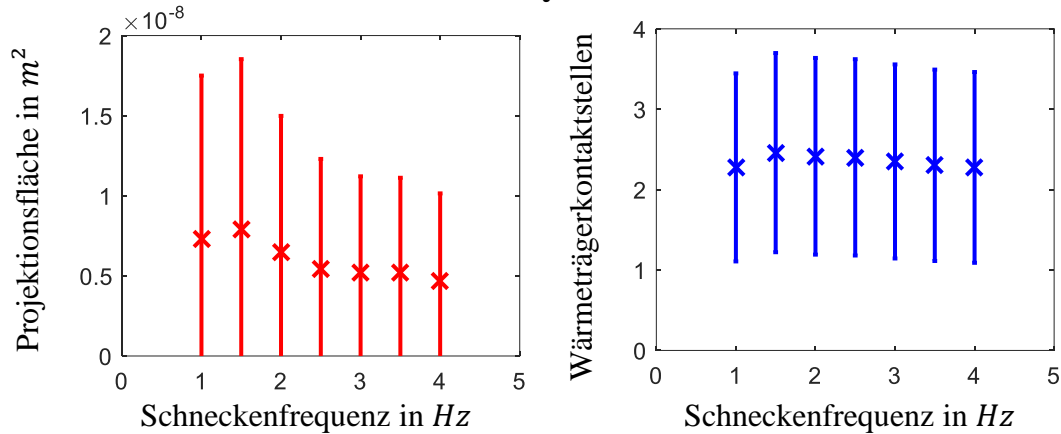


Abbildung 3.4: Mittlere Anzahl und mittlere Projektionsfläche der Biomassepartikel

Funke et. al. [89] untersuchten den Wärmeübergang bei der Pyrolyse von Biomasse im Doppelschneckenmischreaktor nach Schlünder für Partikel gleicher Größe. Dabei betrachtete er den Wärmeübergang durch die Gasphase auch von nicht in Kontakt stehenden Partikeln. Der ermittelte Wert schwankte in Abhängigkeit zur Position im Reaktor zwischen  $90$  und  $250 \text{ W m}^{-2} \text{ K}^{-1}$  und entspricht in etwa dem Doppelten des in Abbildung 3.3 zu beobachtenden Wertes. Wesentlicher Unterschied von Funke zu dieser Arbeit ist die Betrachtung des Partikel/Gas/Partikel-Wärmeüberganges und die Verwendung von Partikeln gleicher Größe.

Die Berechnung des Wärmeüberganges nach Schlünder benötigte in etwa einen Tag für einen Zeitschritt und kann aufgrund dieser Komplexität für Simulationen nicht in die Betrachtung gezogen werden. Jedoch führt die Verwendung von Partikeln gleicher Größe, zu einem dichterem Partikelbett mit mehr Kontaktstellen, als es ein Bett mit Partikeln unterschiedlicher Größe. Diese Thematik wird unter anderem im Verlauf des kommenden Kapitels behandelt.

### 3.2 Einfluss des Partikelverhältnisses auf den Wärmeübergang

Die gängige Meinung bzgl. des Wärmeübergangs ist, dass mit steigendem Wärmeträgerüberschuss der Wärmeübergang von Wärmeträger auf Biomasse verbessert wird. Dem gegenüber steht die Wirtschaftlichkeit, bei der mit größerem Wärmeträgerüberschuss die Investitions- und Betriebskosten steigen. Um dies zu untersuchen, wurden Simulationen mit unterschiedlicher Zusammensetzung, aber konstanten Gesamtvolumenstrom durchgeführt. Aufgrund des großen Spektrums an untersuchten Zusammensetzungen, wurde der Rechenaufwand, durch die Beschränkung der Biomassepartikel auf einen Durchmesser von  $1 \text{ mm}$ , d.h. dem Wärmeträgerdurchmesser, reduziert. Bis auf Zusammensetzung und Biomassepartikeldurchmesser sind die Simulationsparameter unverändert gegenüber Tabelle 2.17. Die Ergebnisse wurden in Abbildung 3.5 zusammengefasst.



## Wärmeübergang in Abhängigkeit der Volumenverteilung bei gleicher Partikelgröße

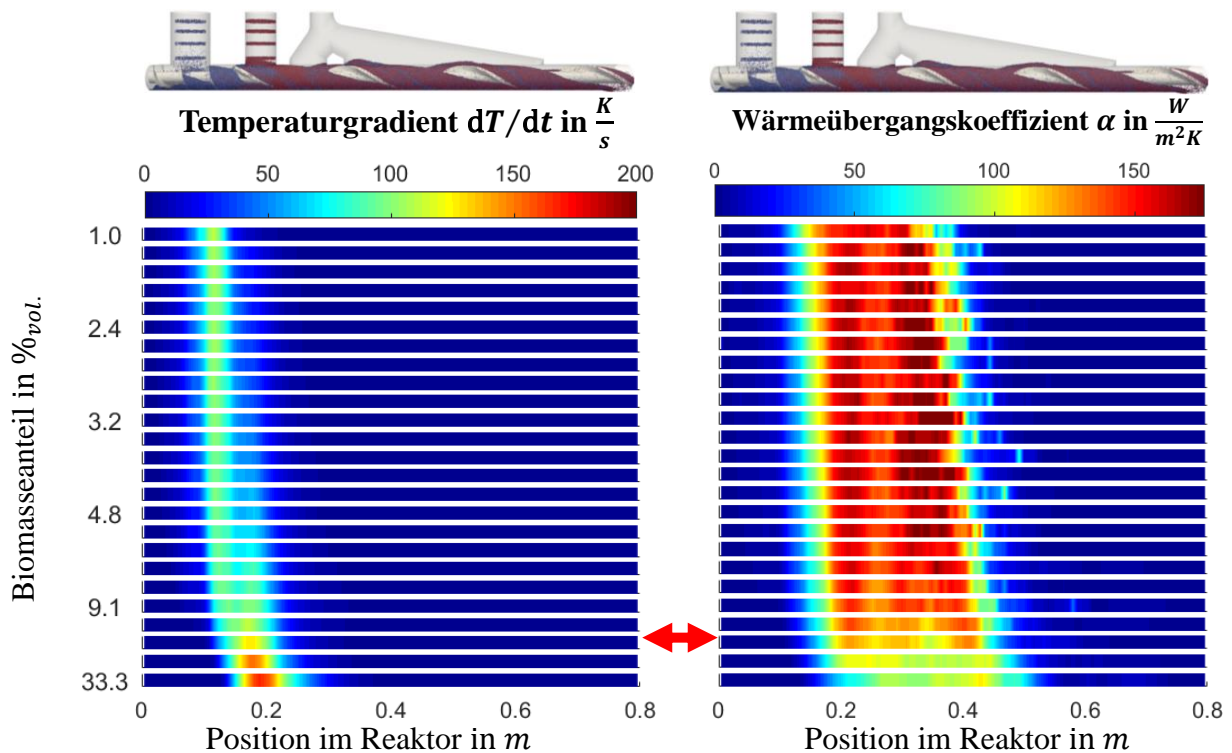


Abbildung 3.5: Mittlerer Temperaturgradient und Wärmeübergangskoeffizient bei konstantem Gesamtpartikelvolumenstrom. Biomasse- und Wärmeträgerdurchmesser  $d=1$  mm. Der rote Pfeil markiert die Standardeinstellung mit 15.4%<sub>vol.</sub> Analog zu Abbildung 3.3 berechnet.

Es muss erwähnt werden, dass alle Simulationen mit konstanten Anfangstemperaturen durchgeführt wurden. D.h. durch das verändern des Mischverhältnis, wird auch die Mischtemperatur verändert. Diese reicht von 728.8 K bei maximalem Biomassegehalt bis 772.2 K bei minimalem Biomasseanteil. Unabhängig von der Mischtemperatur konnte beobachtet werden, dass mit steigender Biomassekonzentration die Rückvermischung in den Biomassetransportbereich sinkt. Dies führt dazu, dass der Wärmeübergang oder auch die Aufheizrate der Biomassepartikel bei steigendem Wärmeträgeranteil sinkt. Die Rückvermischung des Wärmeträgers wird der dominierende Effekt beim Wärmeübergang. Ein Effekt der auch bei einer visuellen Gegenüberstellung (Abbildung 3.6) bestätigt wird.

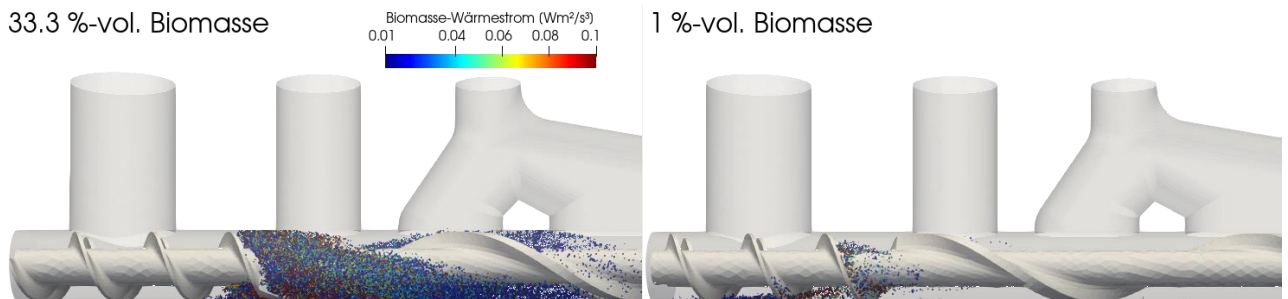


Abbildung 3.6: Gegenüberstellung: Biomassepartikel mit hohem Wärmestrom ( $\geq 0.01 Wm^2/s^3$ ). Links hoher Biomasseanteil, rechts niedriger Biomasseanteil. Wärmeträger ist ausgeblendet.

Erst bei sehr geringem Biomasseanteil beginnt die Biomasseheizrate zu steigen. Der zu beobachtende schmaler werdende Bereich in dem sich die Biomasse aufheizt, ist auf die langsamere Transportgeschwindigkeit, bedingt durch die geringere Ganghöhe (Abbildung 3.6 und Abbildung 3.9) im Biomasse-Transportbereich, zurückzuführen. Der Wärmeübergangskoeffizient bleibt über ein breites Spektrum ( $\leq 9.1\%_{vol}$  Biomasse) weitestgehend konstant. Erst bei höherem Biomasseanteil sinkt dieser merklich.

Diese Aussage wurde für Partikel gleicher Größe und unterschiedlicher Materialeigenschaften getroffen (Biomasse/Wärmeträger). Um den realen Fall abzudecken, wurden ausgewählte Simulationen mit der bekannten Biomassepartikelgrößenverteilung (Tabelle 2.14) durchgeführt.

### Wärmeübergang in Abhängigkeit der Volumenverteilung bei unterschiedlicher Partikelgröße

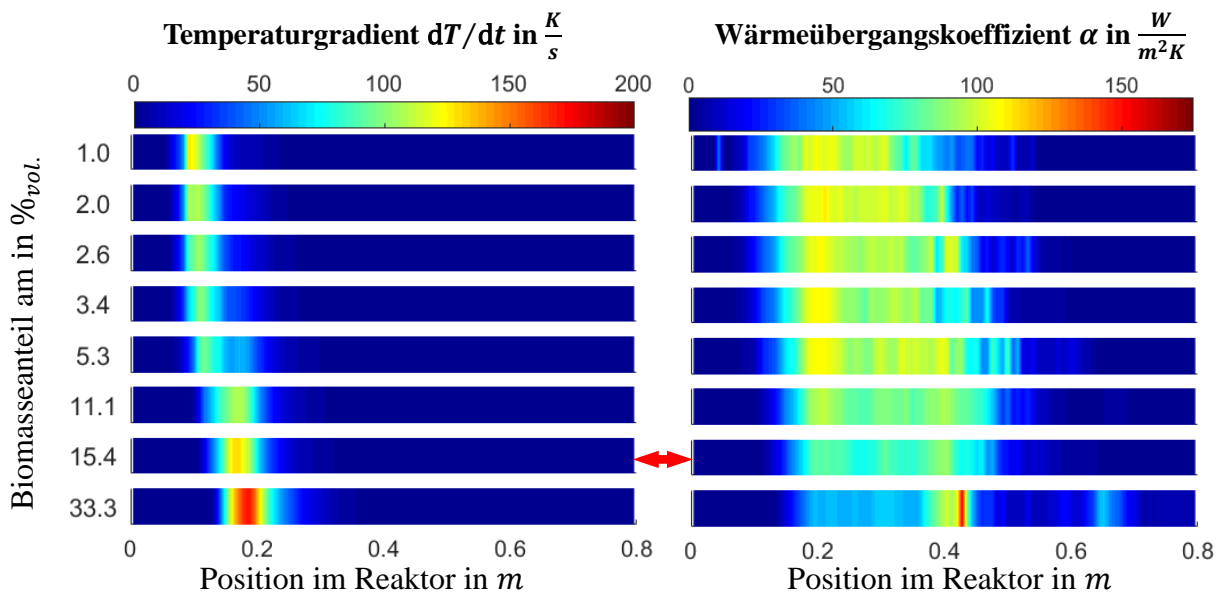


Abbildung 3.7: Wärmeübergang bei konstantem Gesamtpartikelvolumen.  
Biomassepartikeldurchmesser nach (Tabelle 18). Wärmeträgerdurchmesser  $d=1\text{ mm}$ .  
Analog zu Abbildung 3.3 berechnet.

Die Auswertung dieser Simulation mit der verwendeten Partikelgrößenverteilung (Abbildung 3.7) bestätigt weitestgehend die Beobachtungen aus Abbildung 3.5. Unterschiede sind jedoch bei den Heizraten im Bereich mit einer niedrigen Biomassekonzentration und dem rund Faktor zwei geringeren Wärmeübergangskoeffizienten, zu beobachten.

Funke et. al. [89] berechnete für den aktiven Bereich des Wärmeübergangs durch die Gasphase einen Wärmeübergangskoeffizienten von rund  $160\text{ W/m}^2\text{K}$ . Im gleichen Bereich, unter vergleichbaren Bedingungen und mit den in Kapitel 2.7 vorgenommenen Modifikationen des Wärmeübergangs, wird ein weitestgehend konstanten Wert von rund  $150\text{ W/m}^2\text{K}$  erreicht (Abbildung 3.5, roter Pfeil). Im Gleichen Bereich kann, unter den in Kapitel 2.3 gewählten Partikelgrößenverteilungen, ein Wärmeübergangskoeffizient von rund  $90 - 100\text{ W/m}^2\text{K}$  beobachten (Abbildung 3.7, roter Pfeil). Die gute Übereinstimmung bei Partikeln gleicher Größe, ist ein Indiz dass der direkte Partikelkontakt bei der Pyrolyse entscheidend ist. Die Diskrepanz bei unterschiedlicher Partikelgrößenverteilung kann auf den Einfluss der mittleren Anzahl der Kontaktstellen eines Biomassepartikels mit Wärmeträgerpartikeln (Abbildung 3.8) zurückgeführt werden. Mit durchschnittlich 3.2 Wärmeträgerkontaktstellen pro Biomasse, bei Partikeln gleicher Größe, ist dieser Wert mehr als 1/3 größer als 2.4 - der durchschnittlichen Anzahl der Wärmeträgerkontaktstellen pro Biomasse bei der gewählten Partikelgrößenverteilung.

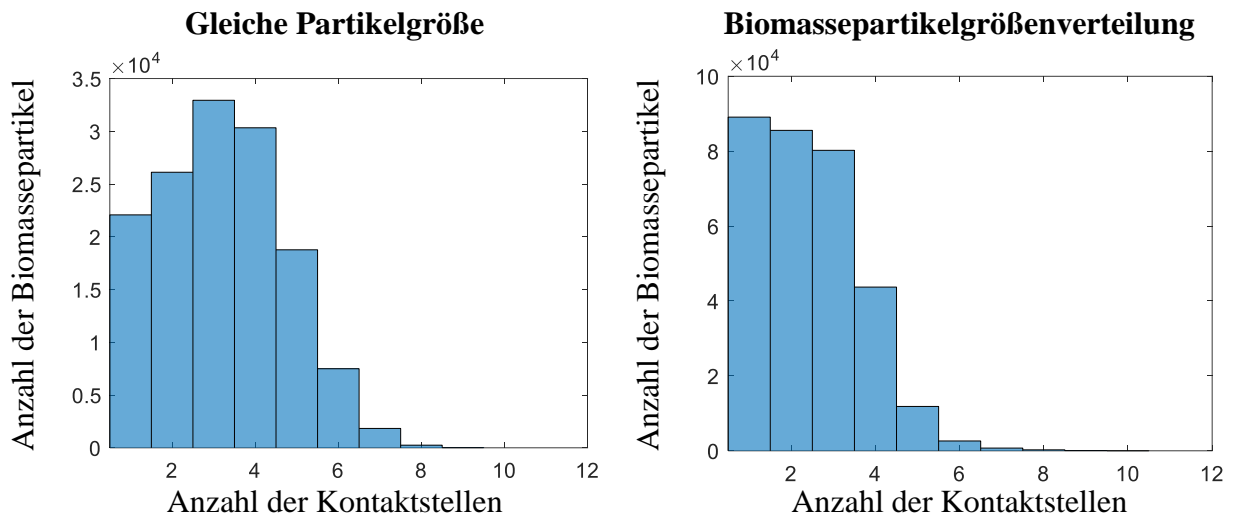


Abbildung 3.8: Verteilung der Biomasse/Wärmeträger-Kontaktstellen.  
 Links: Biomasse und Wärmeträgerpartikel gleicher Größe (Abbildung 3.5, roter Pfeil).  
 Rechts: mit Biomassepartikelgrößenverteilung (Abbildung 3.7, roter Pfeil).

### 3.3 Einfluss der Zugabereihenfolge

Eine häufig diskutierte Einflussgröße ist die Zugabereihenfolge von Biomasse und Wärmeträger. Bei der PYTHON-Technikumsanlage wird Wärmeträger zur Biomasse gegeben, bei der bioliq® Biomasse zum Wärmeträger. Da das Problem der Rückvermischung bei der PYTHON bekannt ist, wird vermutet, dass die bioliq® eine höhere Biomasse Aufheizrate vorzuweisen hat. Jedoch könnte dieser Unterschied auf den Wärmeträger - Sand, der bei der bioliq® verwendet wird und dem damit verbunden Unterschied in Partikelgröße und Dichte, zurückgeführt werden.

Um diesem Unterschied auf den Grund zu gehen, wurden DEM-Simulationen der PYTHON mit vertauschter Zugabereihenfolge von Biomasse und Wärmeträger durchgeführt. Dies machte eine Anpassung der Schneckenengeometrie erforderlich. Da zuerst Wärmeträger in großer Menge zugeführt wird, ist die Ganghöhe im Förderbereich zu niedrig, um diese Menge an Wärmeträger zu transportieren. D.h. dieses Design würde zu Verstopfungen führen. Daher wurde die Ganghöhe des Reaktions- und Mischbereiches auf die komplette Welle ausgeweitet, siehe Abbildung 3.9.



Abbildung 3.9: Schneckenengeometrien für die Zugabereihenfolge:  
 1. Biomasse 2. Wärmeträger (oben), 1. Wärmeträger 2. Biomasse (unten)

Die weiteren Simulationskennwerte und Parameter wurden beibehalten. D.h. Wärmeträger und Biomasse bleiben unverändert, sodass der Einfluss der Biomassezufuhr direkt verglichen werden kann.

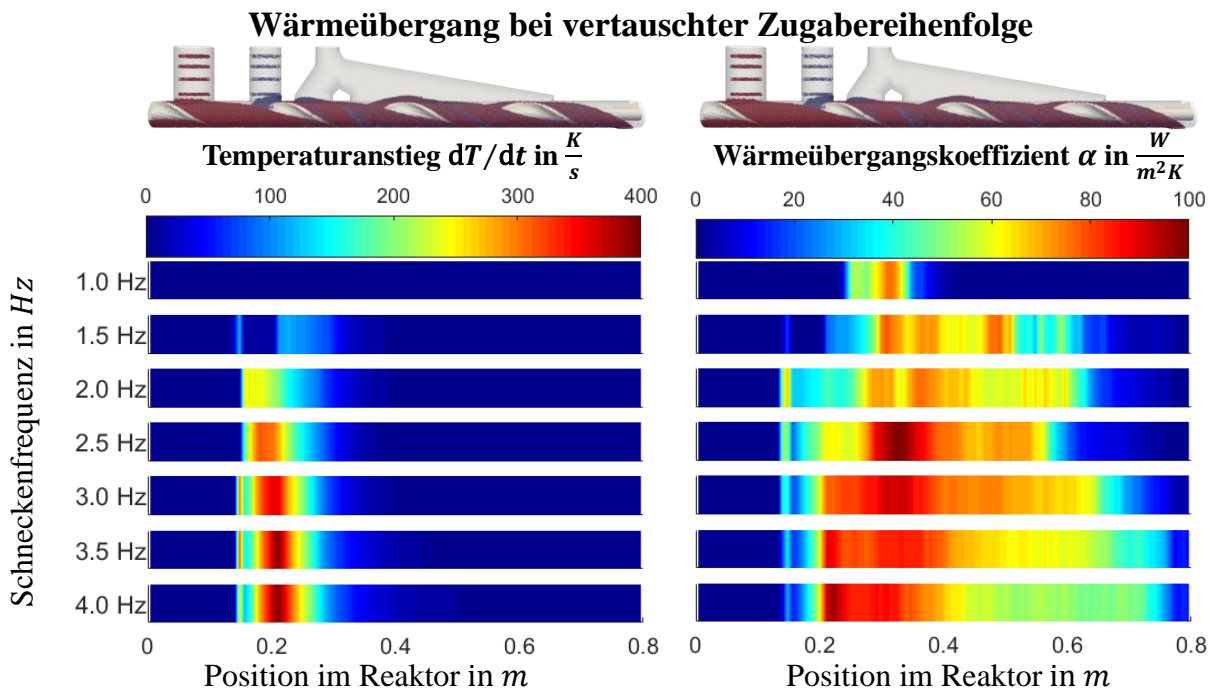


Abbildung 3.10: Wärmeübergang im Doppelschneckenmischreaktor bei vertauschter Zugabereihenfolge in Abhängigkeit zur Schneckengeschwindigkeit.  
 Links: Biomassetemperaturanstieg. Rechts: Biomassewärmübergangskoeffizient.  
 Analog zu Abbildung 3.3 berechnet.

Es zeigt sich, dass im unteren Drehzahlbereich (unterhalb 2 Hz) ein Rückstau der Biomasse am Einlass zu einem schlechten Wärmeübergang führt. Darüber hinaus ist ein durchweg hoher Biomassetemperaturanstieg mit minimaler Rückvermischung zu beobachten (Abbildung 3.10), der bis zu einem Faktor 2 größer ist, als in der ursprünglichen Konfiguration (Abbildung 3.3). Der ermittelte Wärmeübergangskoeffizient für 2.5 – 4 Hz ist jedoch mit dem der Originalkonfiguration vergleichbar. Bei 2 Hz ist ein leichter Rückstau, d.h. ein “schwimmendes“ Biomassebett, welches sich innerhalb einer Umdrehung bildet und auflöst, anhand der Verteilung des Wärmeübergangskoeffizienten zu erkennen (Abbildung 3.11).

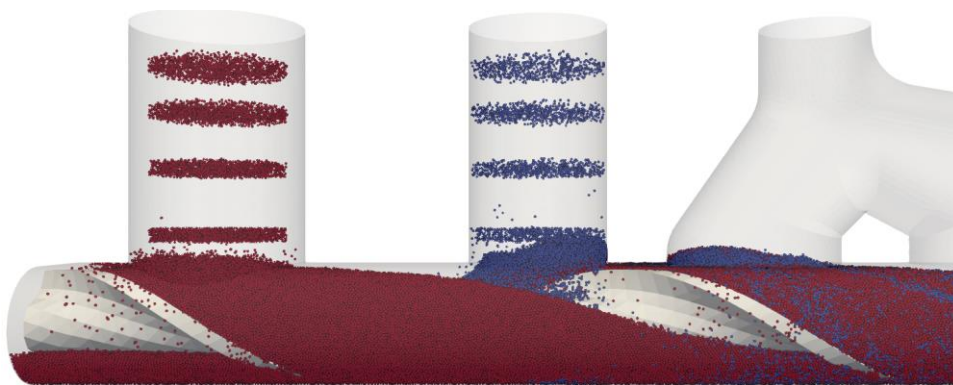


Abbildung 3.11: Leichter Rückstau bei 2 Hz und vertauschter Zugabereihenfolge.  
 Blau: Biomasse, rot: Wärmeträger

Diese Beobachtung wird von der Mischgüte (Abbildung 3.12) an der Biomassezugabestelle bei 2 Hz bestätigt. Bei 3 Hz ist kein Rückstau erkennbar. Bei 4 Hz ist eine eintretende Entmischung, wie auch schon bei der Originalkonfiguration (Abbildung 3.2), zu beobachten.

## Mischgüte bei vertauschter Zugabereihenfolge

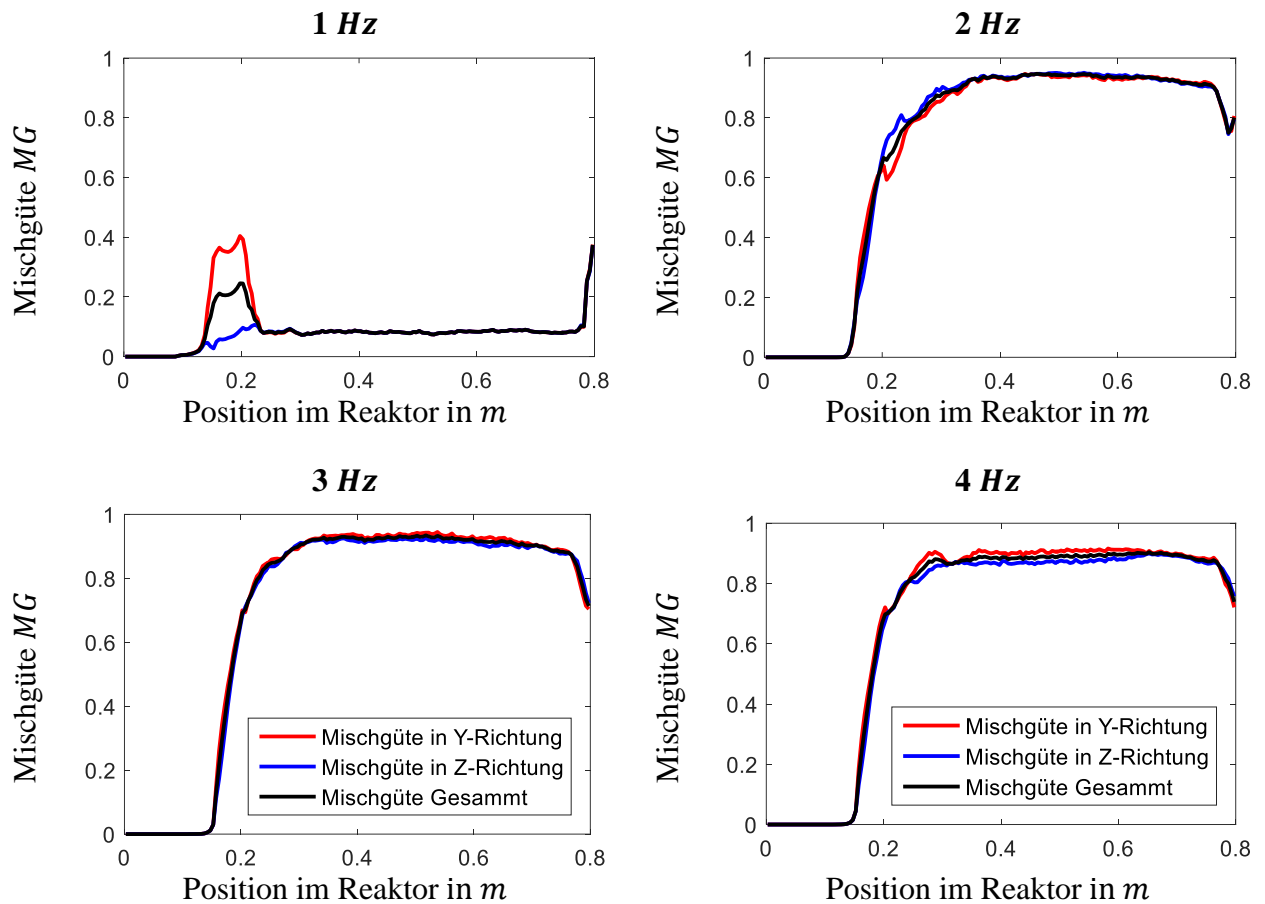


Abbildung 3.12: Ausgewählte Mischgüte bei vertauschter Zugabereihenfolge

### 3.4 Diskussion

Die Untersuchung der Rotationsgeschwindigkeit ergab einen optimalen Betriebsbereich zwischen 2 – 3 Hz. Im unteren Drehzahlbereich konnte in den Simulationen eine Verblockung reproduziert werden. Im oberen Drehzahlbereich konnte eine Entmischung festgestellt werden. Darüber hinaus konnte gezeigt werden, dass mit steigender Rotationsgeschwindigkeit, die Anzahl der Kontaktflächen und deren Größe sinkt. Ein deutliches Indiz dafür, dass der Anpressdruck sinkt. Bei den DEM-Simulationen konnte eine Rückvermischung des Wärmeträgers in den Transportbereich festgestellt werden, der den Wärmeübergang negativ beeinflusst. Ein Versuch diesem Problem zu begegnen ist das Vertauschen der Zugabereihenfolge. Simulationen wurden unter dieser Voraussetzung durchgeführt. Eine Rückvermischung konnte zwar beobachtet werden, stellt aber kein Problem dar (wenige Biomassepartikel wandern in einen Bereich mit hohem Wärmeträgerüberschuss). Jedoch ist das Rückstauprobem in diesem Fall akuter, sodass ein optimaler Betrieb nur bei 3 Hz möglich ist. Wie auch schon bei der ursprünglichen Zugabereihenfolge, entsteht unterhalb 2 Hz eine Verblockung, die den Betrieb unmöglich macht. Im Übergangsbereich um 2 Hz ist die Produktzugabe ähnlich groß wie Fähigkeit des Reaktors diesen einzumischen, so dass innerhalb einer Schneckenumdrehung der Aufbau und Vermsichung eines Biomassebettes auf dem Biomasse/Wärmeträgerbett zu beobachten ist. Da bei solch einem “schwimmenden“ Biomassebett nur die Außenseite direkten Kontakt zum Wärmeträger hat, ist diese Eigenschaft nachteilig. Wie auch schon bei der Originalkonfiguration, tritt bei 4 Hz eine Entmischung

(Abbildung 3.12) ein. Die beobachtete Heizrate der Biomassepartikel ist in etwa doppelt so hoch wie die der Originalkonfiguration, der Wärmeübergangskoeffizient bewegt sich in einem vergleichbaren Wertebereich.

Weitere Studien wurden im Hinblick auf das Biomasse/Wärmeträgerverhältnis durchgeführt. Dabei wurden Wärmeträger und Biomasse mit einem Durchmesser von 1 *mm* verwendet. Es konnte gezeigt werden, dass die Rückvermischung des Wärmeträgers einen dominierenden Effekt unterhalb 9.1 %<sub>vol</sub>-Biomasse darstellt. Ein Vergleich der Wärmeübergangskoeffizienten mit Funke et. al. [89] ergab eine gute Übereinstimmung. Da Funke auch den Wärmeübergang mit nicht in Kontakt stehender Partikel betrachtete, konnte daraus abgeleitet werden, dass der Partikelkontakt die dominierende Einflussgröße darstellt. Darüber hinaus zeigten Vergleichssimulationen, mit der in Kapitel 2.3 eingeführten Partikelgrößenverteilung, dass der Wärmeübergangskoeffizient stark von dieser abhängig ist. D.h. unter der verwendeten Verteilung sank der Wärmeübergangskoeffizient um ein Drittel.

## 4 Computed Fluid Dynamics – Discrete Element Methods (CFD-DEM)

Im Verlauf der bisherigen Arbeit wurde die Charakterisierung des Doppelschneckenmischreaktors mittels DEM-Simulation durchgeführt. Diese beschreibt das Partikelverhalten und das Mischverhalten der Biomasse und des Wärmeträgers. Dabei wird der Stoff- und Energietransport in die Gasphase vernachlässigt. Dieser stellt jedoch einen entscheidenden Einflussfaktor bei der Schnellpyrolyse dar. Sekundärreaktionen, die in der Gasphase ablaufen, nehmen unerwünschten Einfluss auf die Produktausbeute [90]. Daher geht mit einer Optimierung der Schnellpyrolyse auch eine Minimierung der Produktgasverweilzeit einher.

Da sowohl die Produktgasverweilzeit als auch die Bettverteilung in direktem Zusammenhang stehen, ist eine Kopplung erforderlich um beide Effekte in Relation zueinander darstellen zu können. Bei der Simulation der Partikel-Bewegungen wurde der Lagrange-Ansatz (d.h. DEM) gewählt. Für die Simulation der Gasphase stehen mehrere Verfahren zur Auswahl. So können bei der Strömungssimulation Navier-Stokes-Gleichungen, Boltzmann-Gleichungen oder ein weiterer Lagrange-Ansatz (Smoothed Particles Hydrodynamics, SPH) gewählt werden.

Jeder dieser Ansätze hat seinen eigenen Anwendungsbereich. So werden SPH z.B. in der Astrophysik oder aber auch bei der Simulation von Tsunamis eingesetzt. Bei den SPH handelt es sich um eine sehr empirische Methode, die häufig aufgrund ihrer Robustheit gewählt wird [91].

Lattice-Boltzmann-Methoden nutzen eine vereinfachte Teilchen-Mikrodynamik zur Strömungssimulation. Diese kann vielseitig eingesetzt werden. Aufgrund der guten Parallelisierbarkeit zeichnen sie sich durch ihre Geschwindigkeit aus. Jedoch wird bei deren Verwendung ein feines Gitter benötigt, das den Gewinn an Rechengeschwindigkeit aufzehrt [92].

Bei den Navier-Stokes-Gleichungen handelt es sich um ein allgemein gebräuchliches mathematisches Modell der Strömungssimulation. Diese setzen sich aus der Massenerhaltung (Gleichung 4.1), der Impulserhaltung (Gleichung 4.2) und der Energieerhaltung (Gleichung 4.3) zusammen [93].

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho u) = 0 \quad (4.1)$$

$$\frac{\partial}{\partial t} (\rho u) + \nabla \cdot (\rho u u) = \nabla \tilde{\tau} - \nabla p + \rho g \quad (4.2)$$

$$\frac{\partial}{\partial t} (\rho h_r) + \nabla \cdot (\rho u h_r) = -\nabla \cdot q'' + \frac{\partial p}{\partial t} + \nabla \cdot (\tilde{\tau} \cdot u) \quad (4.3)$$

Für inkompressible Medien, d.h.  $\nabla \cdot (\rho u) = 0$ , genügt es die Impulserhaltung (Gleichung 4.2) zu lösen. Diese berücksichtigt die Fluidichte  $\rho$ , die Strömungsgeschwindigkeit  $u$ , den Spannungstensor  $\tilde{\tau}$ , die Erdbeschleunigung  $g$  und den Druck  $p$ . Im Allgemeinen kann diese Methode auch für kompressible Fälle mit einer Machzahl  $Ma < 0.25$  angewandt werden.

Bei kompressiblen Problemen müssen zusätzlich die Massenerhaltung (Gleichung 4.1) und die Energieerhaltung (Gleichung 4.3) berücksichtigt werden. Letztere erfordert die zusätzliche Betrachtung der Enthalpie  $h_r$  und der Wärmestromdichte  $q''$ . [94]

### 4.1 Grundlagen der Strömungssimulation beweglicher Objekte

Die Simulation bewegter Objekte stellt in der gitterbasierten CFD-Simulation eine Herausforderung dar. In den vergangenen Jahren wurden mehrere Methoden entwickelt, um diese Herausforderung zu bewältigen.

Eine davon, das Meshmorphing (Abbildung 4.1, oben/links) bezeichnet eine Technik, bei der Bewegungen durch eine Verzerrung der Gitterstruktur dargestellt werden. In Kombination mit einer

Gitterverfeinerung um das Objekt, kann eine hohe Genauigkeit bei vergleichsweise geringem Rechenaufwand erzielt werden. Diese Methode wird z.B. beim Heben und Senken von Zylindern verwendet. Jedoch nimmt mit steigender Verzerrung die Gitterqualität ab, sodass Bewegungsabläufe wie Rotationen i.A. zu Gitterzellen mit negativen Volumen führen können. Dies hat einen erheblichen Einfluss auf die Stabilität und die Genauigkeit der Simulation [95].

Für einfache rotierende Geometrien wurde die Arbitrary Mesh Interface Methode (AMI) entwickelt (Abbildung 4.1, oben/rechts). Rotierende Objekte werden in einem rotationsymmetrischen Untergitter aus dem Hauptgitter getrennt. Dieses Untergitter kann als Ganzes unabhängig vom Hauptgitter rotiert werden ohne Verzerrungen im Gitter zu erzeugen. Ein Informationsaustausch erfolgt durch Interpolation der Kontaktstelle. Zur Erhöhung der Interpolationsgenauigkeit wird zusätzlich zum Objekt die Kontakt- und Interpolationsfläche verfeinert. Die Weiterentwicklung: Arbitrarily Coupled Mesh Interface (ACMI) enthält eine Notfall-Randbedingung, die in Kraft tritt, falls sich die Grenzflächen der Gitter nicht mehr berühren. In Kombination mit dem Mesh morphing können Rotationen, hebende und senkende Bewegungen umgesetzt werden. Jedoch sind komplexere Bewegungsabläufe ausgeschlossen [96].

Aus diesem Grund wurden zwei Techniken entwickelt, die in Konkurrenz zueinander stehen. Beim Overset Mesh (Abbildung 4.1, unten/links) werden Objekte durch ein zweites separates Gitter dargestellt, das den fluiden Randbereich des Objektes enthält. Um einen optimalen Austausch von Informationen zu gewährleisten, muss die fluide-Objektgrenzschicht mindestens drei bis fünfzehn Gitterzellen beinhalten. Als Faustregel gilt: je mehr desto besser. Des Weiteren ist es erforderlich, dass die Gitterzellen des fluiden Randbereiches und des Hauptgitters im Bereich der Überlappung annähernd gleich groß sind [97]. Da zwischen Schnecke und Reaktorwand ein minimaler Abstand  $1\text{ mm}$  zu finden ist, müsste die Gitterbreite auf rund  $0.1\text{ mm}$  gesenkt werden, um die erforderliche Überlappung zu gewährleisten. Neben der Verletzung, die für die CFD-DEM erforderlichen minimalen  $1\text{ mm}$  Gitterbreite, führt dies bei kubischem Wachstum zu einem Faktor 1000 größeren Rechenaufwand.

In Konkurrenz dazu stehen die Immersed Boundary Methods (Abbildung 4.1, unten/rechts) nach Peskin [98]. Dabei werden Objekte im Rechengitter identifiziert. Die korrekte Strömungsführung wird durch Krafttherme, die auf den Objektraum wirken und Randbedingungen herstellen, realisiert. Da eine Neuberechnung des Gitters nicht erforderlich ist, eignet es sich besonders für komplexe Objekte in Bewegung. Peskin legte mit seinem Ansatz den Grundstein. Heute werden Verfahren wie

- Immersed Interface Method [99],
- Penalty Method [100],
- Fictitious Domain Method [101] und
- Level-Set Based Cut-Cell Method [102]

zu den Immersed Boundary Methods gezählt [103].

Jedoch haben die für OpenFOAM® verfügbaren Implementationen alle das gleiche Problem. Sie sind nicht für die Verwendung in kompressibler Strömungssimulation geeignet. Häufig kommen dabei Verfahren wie der Cut-Cell Ansatz oder der Ghost-Cell Ansatz zum Einsatz. Beide stellen eine elegante Möglichkeit dar, Objekttrandbedingungen im Rechengitter darzustellen, haben jedoch Probleme die Massenerhaltung einzuhalten.

Abhilfe schaffen die Penalty Methoden, welche die Massenerhaltung am Rand nicht verletzen und einfach zu implementieren sind. Im Gegensatz zu den Cut-Cell-Ansatz erfolgt eine Korrektur nicht über die Objektgrenzfläche, sondern über die vom Objekt vollständig enthaltenen Gitterzellen (Abbildung 4.1, unten/rechts, blauer Bereich).



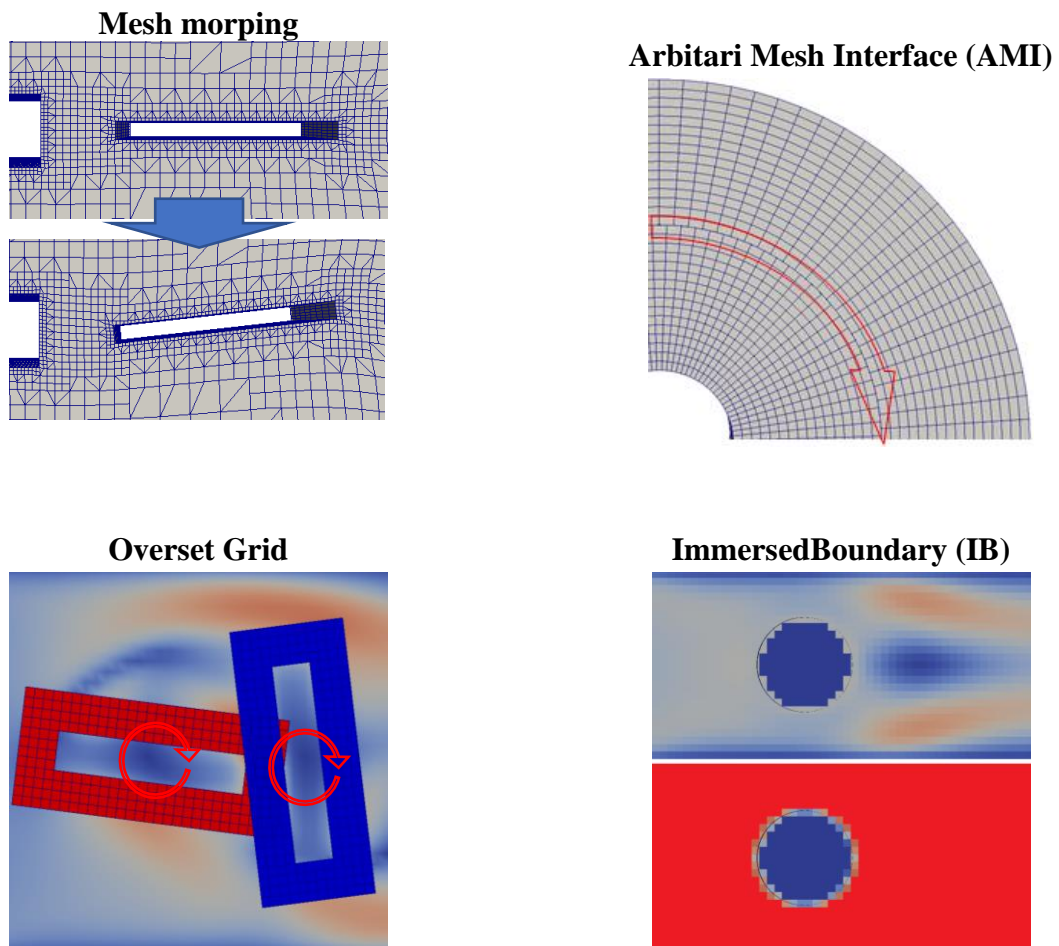


Abbildung 4.1: Bewegungsmodelle bei der CFD-Simulation

In den vergangenen Jahren hat sich insbesondere die Brinkman-Erweiterung (Gleichung 4.5, Brinkman et. al. [104, 105]) des Darcy-Gesetzes (Gleichung 4.4, Darcy et. al. [106]) zur Berechnung des Strömungswiderstandes als robustes und effizientes Verfahren erwiesen [107].

$$\nabla p = -\frac{\mu}{K} \alpha u \quad (4.4)$$

$$\nabla p = -\frac{\mu}{K} \alpha u + \mu \nabla^2 \alpha u \quad (4.5)$$

Für den inkompressiblen Fall formulierte Angot et. al. [107] die Impulsgleichung durch Berücksichtigung eines Strafterms (Gleichung 4.6)

$$\frac{\partial u}{\partial t} + u \cdot \nabla u - \nu \Delta u = -\nabla p - \frac{1}{\eta} \chi(x, t) (u - U_0) \quad (4.6)$$

Dabei ist  $U_0$  die Geschwindigkeit des Objektes,  $0 < \eta \ll 1$  die Gewichtung und  $\chi(x, t)$  die Diskretisierung des Objektes, d.h.

$$\chi(x, t) = \begin{cases} 1, & \text{falls } x \in \Omega_s \\ 0, & \text{sonst} \end{cases}$$

Sind mehrere Objekte zu betrachten, so gilt:

$$\Omega_s = \bigcup \Omega_{s_i}$$

Wobei  $\Omega_{s_i}$  die inneren Gitterzellen und  $\partial\Omega_{s_i}$  der Rand des diskretisierten Objektes  $i$  (Abbildung 4.2) darstellt.

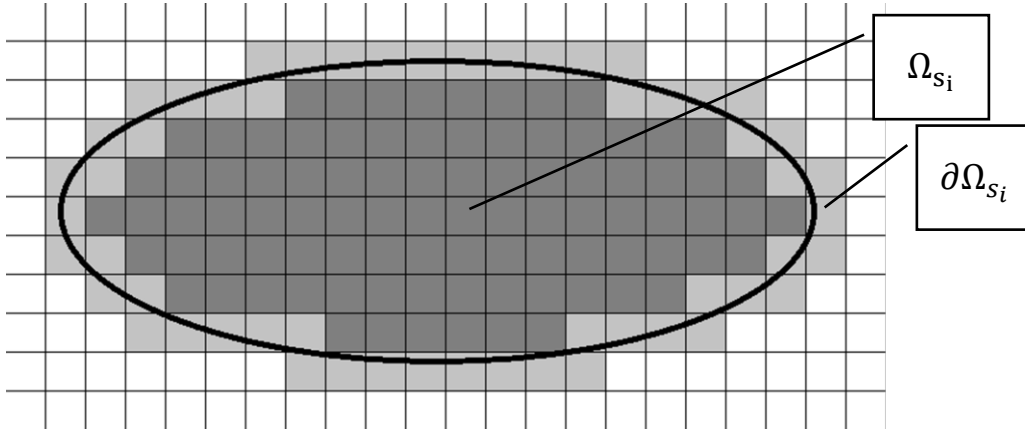


Abbildung 4.2: Diskretisierung eines Objektes beim Brinkman-Strafverfahren

Angot et. al. zeigt, dass für den inkompressiblen Fall mit steigender Gewichtung die Brinkman-Strafverfahren gegen klassische Methoden konvergieren.

#### 4.1.1 Brinkman Strafverfahren in kompressiblen Strömungen

Ein wesentlicher Vorteil des Brinkman-Strafverfahrens, gegenüber den oben diskutierten Methoden, ist die einfache Umsetzung für die kompressible Strömungssimulation, sein robustes Verhalten und die Kompatibilität mit der CFDEM-Simulationen.

Die Implementierung erfordert eine Erweiterung der Erhaltungsgleichung (Gleichung 4.1) in ihrer Form für poröse Medien (Gleichung 4.18). Die Impulserhaltung (Gleichung 4.6) kann unverändert übernommen werden.

Bei der Verwendung des Brinkmann-Strafverfahrens treten im Vergleich zu anderen Verfahren Abweichungen im Bereich der Objekt-oberfläche auf. So werden bei der Implementierung nach Angot et. al. [107] lediglich die im Objekt vollständig enthaltenen Gitterzellen mit einem Strafterm belegt. Dies führt in Kombination mit CFDEM-Simulationen, bei denen Partikel als poröses Medium betrachtet werden, zu Abweichungen im Randbereich, die auf nichtbeanspruchten Zellvolumen und fehlenden Widerstand zurückgeführt werden können. Unter Berücksichtigung dieser Problemstellung wurde die Objektdiskretisierung  $\chi(x, t)$  modifiziert, sodass

$$\chi(x, t) = \begin{cases} 1, & \text{falls } x \in \Omega_s \\ \varepsilon_p, & \text{falls } x \in \partial\Omega_s \\ 0, & \text{sonst} \end{cases}$$

gilt. Wobei  $\varepsilon_p$  der Anteil des Objekt- oder auch Feststoffvolumens in einer Zelle ist.

Brown-Dymkoski et. al. [108] formulierte die Zustandsgleichungen für das Brinkmann-Strafverfahren für den schwach turbulenten Fall (Gleichung 4.7). Dabei betrachtete er die fluiden Raum mit Porosität  $\varepsilon_\chi$ , welche sich aus der Objektdiskretisierung, d.h.  $\varepsilon_\chi = 1 - \chi(x, t) \cdot \varepsilon$ , ergibt.  $\varepsilon$

ist dabei der Steuerparameter. Für ihn gilt:  $0 \ll \epsilon < 1$ . D.h. je größer dieser gewählt wird, desto besser spiegelt die Porösität den realen Sachverhalt wieder.

$$\begin{aligned} \frac{\partial \varepsilon_\chi \rho}{\partial t} + \nabla \varepsilon_\chi \rho u &= 0 \\ \frac{\partial \rho u}{\partial t} + \nabla \cdot (\rho u u) &= -\nabla p + \nabla \cdot \tilde{\tau} + \frac{\chi}{\tilde{\eta}} (u - U_0) \\ \frac{\partial \rho h_r}{\partial t} + \nabla \cdot (\rho u h_r) &= -\nabla \cdot q'' + \frac{Dp}{Dt} + \nabla \cdot (\tilde{\tau} \cdot u) + \frac{\chi}{\tilde{\eta}_T} (h_r - h_{r0}) \end{aligned} \quad (4.7)$$

Sowohl die viskose Permeabilität  $\tilde{\eta}$  als auch die thermische Permeabilität  $\tilde{\eta}_T$  sind sehr klein gewählt. D.h.  $0 < \tilde{\eta}, \tilde{\eta}_T \ll 1$ . Da im Fall dieser Arbeit die Temperaturrandbedingung ein bewegliches Objekt vom Typ Neumann ist (d.h. zeroGradient), entfällt der Term  $\frac{\chi}{\tilde{\eta}_T} (h_r - h_{r0})$ .

Mit diesem Hintergrund und Überlegungen wurde im Rahmen dieser Arbeit ein Code geschrieben und an der Kármánschen-Wirbelstraße getestet. Bei dieser wird ein Zylinder umströmt. Dadurch bilden sich in einem Reynoldsbereich (Gleichung 4.8) von  $40 \leq Re \leq 200$  auf der Rückseite zwei gegenläufige Wirbel aus. In diesem Bereich ist die Wirbelablösung komplett laminar. Um schneller einen stationären Zustand zu erreichen, wurde ein Reynoldswert von rund 800 gewählt.

$$Re = \frac{\rho \cdot u \cdot D}{\eta} \quad (4.8)$$

Die Ablösefrequenz  $f$  dieser Wirbel kann unter Verwendung der Strouhal-Zahl  $Sr$  (Gleichung 4.9), des Objektdurchmessers  $D$  und der Strömungsgeschwindigkeit  $u$  berechnet werden. Da die Strouhal-Zahl im verwendeten Reynoldsbereich  $Sr \approx 0,21$  weitestgehend konstant ist, kann diese zur Validierung der Simulation herangezogen werden.

$$Sr = \frac{f \cdot D}{u} \quad (4.9)$$

Für den statischen Fall wurden Simulationen mit einer Gitterbreite von  $5 \text{ mm}$  durchgeführt. Dabei wurde für die kompressible Strömungssimulation, der Standardsolver rhoPisoFoam als Referenz verwendet. Für ihn wurde ein zylindrisches Loch mit Durchmesser  $D = 100 \text{ mm}$  eingebaut (Abbildung 4.3)

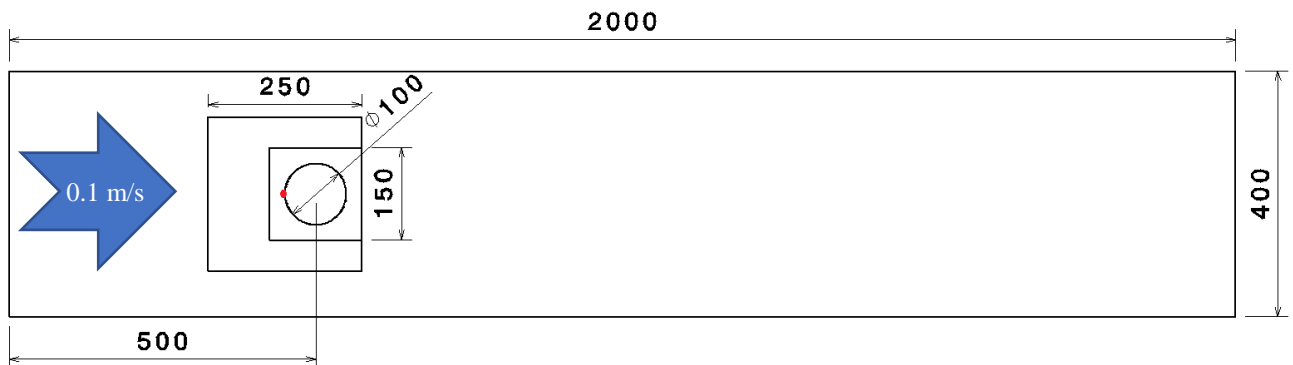


Abbildung 4.3: Geometrischer Aufbau des Kármánschen-Wirbelstraße Testfall

Dieser Aufbau dient als Grundlage aller weiteren Testsimulationen betreffend des Brinkman-Strafverfahrens. Während der Referenzfall geometrisch mittels blockMesh an den Zylinder angepasst

ist, wurde mit einer Ausnahme, ein homogenes Gitter für die Simulationen mittels rhoPenaltyFoam-Solvers gewählt. Die Randbedingungen (Tabelle 4.1) wurden dabei für alle Simulationen konstant gehalten.

Tabelle 4.1: Randbedingungen am Modellversuch: Karamansche Wirbelstraße (Abbildung 4.3)

Randbedingungen	$\mathbf{u}$ [m/s]	$p$ [kg/(m · s <sup>2</sup> )] <sup>VII</sup>
<b>Einlass</b>	fixedValue (0.1 0. 0.)	zeroGradient
<b>Seiten</b>	fixedValue (0. 0. 0.)	zeroGradient
<b>Auslass</b>	fixedValue (0. 0. 0.)	fixedValue 1000
<b>Zylinder</b> <sup>VIII</sup>	movingWallVelocity <sup>IX</sup>	zeroGradient

Für alle Simulationen wurden Kennwerte für Luft bei 20 °C verwendet. Die Simulationszeit einer jeden Simulation betrug mindestens 60 s.

Simuliert wurde der dynamische Fall (Zylinder in Bewegung), als auch der statische Fall (ruhender Zylinder). Für den statischen Fall wurden Simulationen mit dem OpenFoam-Standardsolver rhoPisoFoam, als auch mit dem eigenen rhoPenaltyFoam-Solver durchgeführt. Für den rhoPisoFoam-Solver wurde ein angepasstes Gitter mit Aussparung um den Zylinder verwendet. Verglichen wurde das Resultat mit Simulationen unter Verwendung des rhoPenaltyFoam-Solvers und identischen Gitters mit geschlossener Aussparung (rhoPenaltyFoam 1, Tabelle 4.2), homogenes Gitter (rhoPenaltyFoam 2, Tabelle 4.2) und einem homogenen Gitter mit anschließender Gitterverfeinerung um den Zylinder (Abbildung 4.3, kleines Quadrat) (rhoPenaltyFoam 3, Tabelle 4.2). Zur Auswertung wurde die Strouhal-Zahl bestimmt. Die dafür erforderliche Ablösefrequenz wurde mittels Fourier-Transformation aus Simulationsdaten ermittelt.

Tabelle 4.2: Vergleich der Simulationen. rhoPisoFoam mit zylindrischer Aussparung, rhoPenaltyFoam 1 mit geschlossener Aussparung, rhoPenaltyFoam 2 mit homogener Gittergröße, rhoPenaltyFoam 3 (wie rhoPenaltyFoam 2) mit verfeinertem Gitterbereich

	Str	Ablösefrequenz
<b>rhoPisoFoam</b>	0.202	0.404
<b>rhoPenaltyFoam 1</b>	0.208	0.417
<b>rhoPenaltyFoam 2</b>	0.194	0.388
<b>rhoPenaltyFoam 3</b>	0.204	0.408

Der Vergleich der Simulationen zeigt eine gute Übereinstimmung der Strouhal-Zahlen für den Gitter-angepassten Fall und für den verfeinerten Fall. Kleinere Abweichungen können für das homogene Gitter ohne Verfeinerung festgestellt werden. Jedoch legen die Simulationen 1 u. 3 den Schluss nahe, dass es sich hierbei um Abweichungen aufgrund des Randwertes handelt und diese damit der Diskretisierung zu Lasten gelegt werden kann.

Im dynamischen Fall rotiert der Zylinder mit 60 rpm um den roten Punkt (Abbildung 4.3). Als Standardmethode zur Simulation rotierender Objekte stehen zum Zeitpunkt der Niederschrift dieser Arbeit lediglich die AMI-Methoden zur Verfügung. Jedoch zeigte sich, dass mit gegebener

<sup>VII</sup> Im kompressiblen Fall. Für inkompressible Fälle wird der Druck durch die Dichte geteilt.

<sup>VIII</sup> Falls erforderlich

<sup>IX</sup> Relativgeschwindigkeit an Objektoberfläche

Diskretisierung die Interpolationsfehler die Simulation dominieren und damit ein Vergleich unmöglich macht. Daher wurde ein anderer Ansatz gewählt.

Eine gängige Annahme ist, dass Fluide in einer Strömungssimulation mit Strömungsgeschwindigkeiten unter dem 0.25-fachen der Schallgeschwindigkeit sich Verhalten wie ein inkompressibles Fluid [109]. Unter dieser Annahme konnte der Immersed-Boundary-Solver aus foam-extend-4.0 verwendet werden. Dieser wurde mit einem homogenen Gitter getestet und mit dem kompressiblen rhoPenaltyFoam-solver gegenübergestellt (Abbildung 4.4).

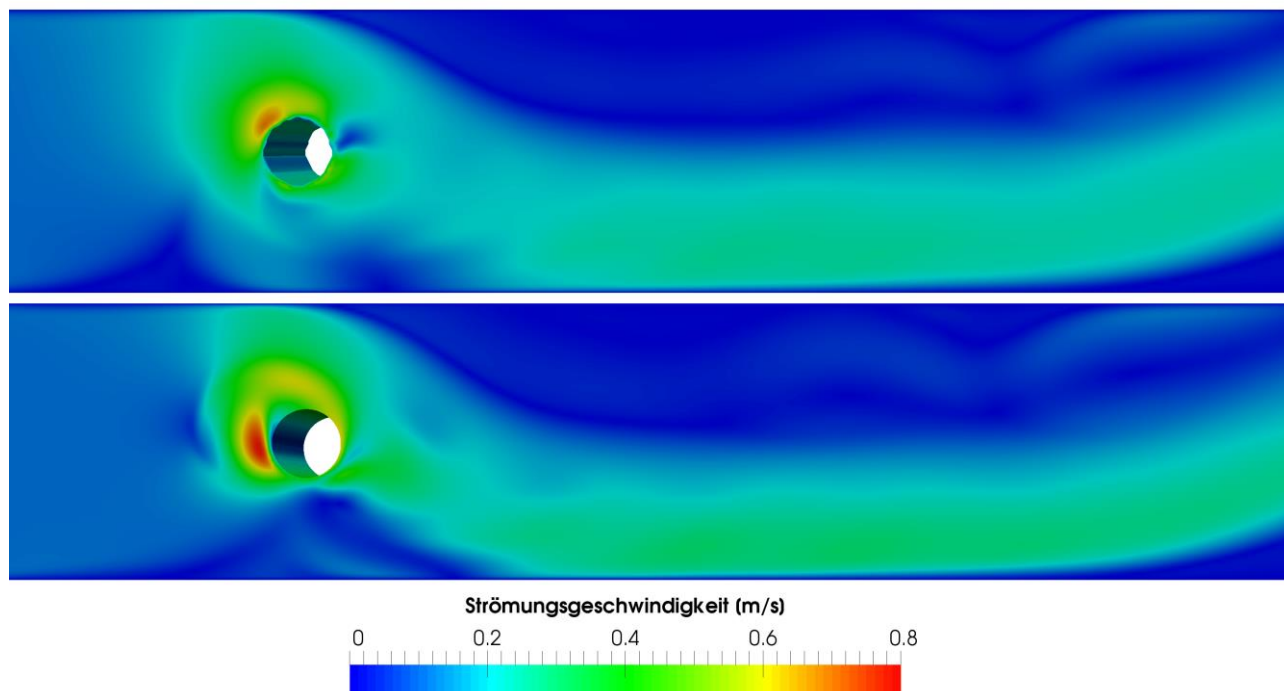


Abbildung 4.4: Vergleich des pimpleDyMibFoam-Solver (oben, FoamExtended-4.0) mit dem rhoPenaltySolver (unten)

Der Vergleich des Strömungsbildes (Abbildung 4.4) zeigt einen ähnlichen Strömungsverlauf. Jedoch gibt es Unterschiede bzgl. der Maximalgeschwindigkeit. Diese Beobachtung konnte anhand von Simulationsdaten (Tabelle 4.3) belegt werden.

Tabelle 4.3: Maximalgeschwindigkeiten ermittelt zwischen 50-60s an einem Messpunkt, 130 mm in x-Richtung von der Rotationsachse in Strömungsrichtung

	<b>min</b>	<b>max</b>
	<b>[m/s]</b>	<b>[m/s]</b>
<b>pimpleDyMibFoam</b>	0.18	0.34
<b>rhoPenaltyFoam 0</b>	0.21	0.46
<b>rhoPenaltyFoam 1</b>	0.18	0.50

Mit gegebener Drehzahl und Umfang, ist die Spitzengeschwindigkeit am Zylinder rund 0.63 m/s. Somit ist die Übereinstimmung mit dem zu erwartenden Ergebnis beim rhoPenaltyFoam-Solver am größten.

Die Abweichungen könnten auf unterschiedliche Annahmen (kompressibel/inkompressibel), auf unterschiedliche Verfeinerungsstufen oder auf größere Interpolationsabweichungen beim Ib-Solver zurückgeführt werden.

Jedoch kann an dieser Stelle festgehalten werden, dass der Brinkmann-Penalty-Algorithmus ein gutes Ergebnis liefert.

## 4.2 Grundlagen der CFDEM-Simulation

DEM-Simulationen bieten die Möglichkeit den Stoff- und den Energietransport von Schüttungen zu simulieren. Jedoch werden dabei Annahmen, wie fehlender Strömungswiderstand und konstante Eigenschaften des fluiden Hintergrundmediums, getroffen. Beides spielt bei der Simulation der Pyrolyse eine tragende Rolle. Aus diesem Grund wird häufig eine Kopplung aus DEM und CFD-Simulation verwendet. Diese beiden komplementären Methoden ergänzen einander und erlauben so eine Simulation der Schnellpyrolyse.

Bei der Strömungssimulation können Partikel über Immersed Boundaries (CFD-Zellgröße  $\ll$  DEM-Partikelgröße) oder als poröses Medium (CFD-Zellgröße  $\gg$  DEM-Partikelgröße) zur Umsetzung verwendet werden. Während Immersed Boundaries häufig zur Simulation von Einzelpartikeln verwendet werden, ist diese Methode bei größeren Partikelzahlen meist zu rechenaufwändig. Bei der Darstellung als poröses Medium werden die Partikel im CFD-Gitter lokalisiert und die Porosität  $\varepsilon_f$  in den Gitterzellen bestimmt [110]. In Kombination mit der Navier-Stokes-Gleichung für poröse Medien (Gleichungen 4.10) kann das Partikelbett als poröser Bereich im CFD-Gitter interpretiert werden.

$$\begin{aligned} \frac{\partial \varepsilon_f \rho}{\partial t} + \nabla \cdot (\varepsilon_f \rho u) &= 0 \\ \frac{\partial \varepsilon_f \rho u}{\partial t} + \nabla \cdot (\varepsilon_f \rho u u) &= -\varepsilon_f \nabla p - S_p + \nabla \cdot (\varepsilon_f \bar{\tau}) + \varepsilon_f \rho g \\ \frac{\partial \varepsilon_f h_r}{\partial t} + \nabla \cdot (\varepsilon_f \rho u h_r) &= -\nabla \cdot \varepsilon_f q'' + \frac{D \varepsilon_f p}{Dt} + \nabla \cdot (\varepsilon_f \bar{\tau} \cdot u) + \dot{q}_{PF} \end{aligned} \quad (4.10)$$

Ein Kräfteaustausch erfolgt durch den impliziten Quellterm  $S_p$  (Gleichung 4.11). Dieser wird aus dem Zellvolumen  $V_i$  über den darin enthaltenen Partikeln  $I_i$ , den Partikelkräften  $F_p$  und der Partikelgeschwindigkeit  $u_p$  berechnet.

$$S_{p,i} = \frac{1}{V_i} \cdot \sum_j^{I_i} \frac{V_j \beta}{\varepsilon_p} \cdot (u - u_p) \quad (4.11)$$

Dieser ist vom Kräfteaustauschkoeffizient  $\beta$  (Gleichungen 4.12) abhängig.

$$\begin{aligned} \beta &= \frac{18\mu\varepsilon_f^2\varepsilon_p}{d_p^2} \cdot \left( F_0(\varepsilon_p) + \frac{1}{2} \cdot F_3(\varepsilon_p) \cdot Re_p \right) \\ Re_p &= \frac{\varepsilon_f \rho_f |u - u_p| d_p}{\eta} \\ F_0(\varepsilon_p) &= \begin{cases} \frac{1 + 3\sqrt{\frac{\varepsilon_p}{2}} + \frac{135}{64}\varepsilon_p \ln(\varepsilon_p) + 16.14\varepsilon_p}{1 + 0.681\varepsilon_p - 8.48\varepsilon_p^2 + 8.16\varepsilon_p^3}, & \text{falls } \varepsilon_p < 0.4 \\ \frac{10\varepsilon_p}{\varepsilon_f^3}, & \text{falls } \varepsilon_p \geq 0.4 \end{cases} \\ F_3(\varepsilon_p) &= 0.0673 + 0.212\varepsilon_p + \frac{0.0232}{\varepsilon_f^5} \end{aligned} \quad (4.12)$$

Auf der Lagrange-Seite fließt der Kraftaustausch  $F_f$  (Gleichung 4.13) in die Kräftebilanz (Gleichung 2.3, Seite 12) mit ein.

$$F_f = \frac{V_p \beta}{\varepsilon_p} \cdot (u - u_p) \quad (4.13)$$

Der implizite Kräfte-Quellterm  $S_p$  geht mit dem nach Koch & Hill [111, 112] berechneten Strömungswiderstand von Fluid auf die Partikel einher. Koch, Hill & Ladd [113] zeigten die Gültigkeit dieses Modelles für einen moderaten Reynoldsbereich. Leitz et. al. [114, 115] untersuchten die in CFDEM® verfügbaren Modelle für die Gas-Fluid-Interaktion bei hohen Reynoldszahlen anhand eines Sprühvorganges. Er belegte, dass unter den gängigen Modellen, u.a. Ergun [116], Wen & Yu [117], Schiller & Naumann [118], Gidaspow [119], Beetstra et. al. [120, 121] und Di Felice [122], Koch & Hill den Versuch am genauesten beschreiben.

Bei der Simulation von Partikel-Fluid-Interaktionen spielt der Partikelantrieb  $F_A$  (Gleichung 4.14) eine entscheidende Rolle. Dieser fließt in die Kräfteerhaltung (Gleichung 2.3) der Lagrangesimulation mit ein.

$$F_A = g \cdot \rho_f \cdot V_p \quad (4.14)$$

#### 4.2.1 Wärmeübergang bei der CFDEM®-Simulation

Zur Simulation des Wärmeaustauschs von Fluid und Partikel bietet CFDEM-PUBLIC® die Möglichkeit diesen als Wärmeübergang an der Partikeloberfläche darzustellen (Gleichung 4.15).

$$\dot{q}_{PF} = \alpha_{PF} \cdot A_P \cdot (T_P - T_F) \quad (4.15)$$

Der Wärmeübergangskoeffizienten  $\alpha_{PF}$  wird dabei nach Li et. al. [123] berechnet (Gleichung 4.16).

$$Nu_s(Re) = \begin{cases} 2 + 0.6 \cdot r_p^{3.5} \cdot Re^{\frac{1}{2}} \cdot Pr^{\frac{1}{3}}, & \text{für } Re < 200 \\ 2 + 0.5 \cdot r_p^{3.5} \cdot Re^{\frac{1}{2}} \cdot Pr^{\frac{1}{3}} + 0.02 \cdot r_p^{3.5} \cdot Re^{0.8} \cdot Pr^{\frac{1}{3}}, & \text{für } 200 < Re < 1500 \\ 2 + 0.000045 \cdot r_p^{3.5} \cdot Re^{\frac{18}{10}}, & \text{für } Re > 1500 \end{cases}$$

$$\alpha_{PF} = \frac{\lambda_F \cdot Nu_{PF}(Re)}{2 \cdot r_p} \quad (4.16)$$

Dieser betrachtet den Wärmeübergang beim transport von Polymerpellets in einem pneumatischen Transportrohr bei einer Prandtl-Zahl von  $Pr = 0.713$  und erreichte eine gute Approximation der experimentell ermittelten Daten.

$$\dot{q}_{PF} = c_p \cdot m \cdot (T_{alt} - T_{neu}) \quad (4.17)$$

$\dot{q}_{PF}$  kann direkt als Quellterm in die Energieerhaltung (Gleichung 4.10) eingesetzt werden. Auf Lagrange-Seite wird die Partikeltemperatur unter Berücksichtigung der Beziehung der Gleichung 4.17 berechnet.

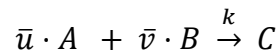
### 4.3 Grundlagen der Simulation reaktiver Strömungen

Aufgrund des modularen Aufbaus kann die Strömungssimulation, basierend auf den Navier-Stokes-Gleichungen durch deren Erweiterung, einfach auf weitere Probleme ausgedehnt werden. So auch auf chemisch-reaktive Strömungen.

Ein einfacher Transport von Gasen kann durch die Erweiterung der Navier-Stokes-Gleichung um die skalare Transportgleichung (Gleichung 4.18) dargestellt werden.

$$\frac{\partial \rho Y}{\partial t} + \underbrace{\nabla \cdot (\rho u Y)}_{\text{Konvektion}} - \underbrace{\alpha \Delta Y}_{\text{Diffusion}} = 0 \quad (4.18)$$

Damit kann der Transport einer Komponente  $Y$  mit Diffusionskoeffizienten  $\alpha$  in einem Fluid dargestellt werden. Um chemische Reaktionen in einem Fluid darstellen zu können, muss neben dem Transport (Gleichung 4.18), die Reaktion simuliert werden. Dazu müssen für die Reaktanten  $A$  und  $B$  eine Reaktionsgleichung



und das dazugehörige System an Differentialgleichungen (DGL, z.B. Gleichungen 4.19), unter Berücksichtigung der Reaktionsordnung  $n = \bar{u} + \bar{v}$  mit  $\bar{u} \geq 0$  und  $\bar{v} \geq 0$ , aufgestellt werden.

$$\begin{aligned} \frac{\partial c(A)}{\partial t} &= -k \cdot c(A)^{\bar{u}} \cdot c(B)^{\bar{v}} \\ \frac{\partial c(B)}{\partial t} &= -k \cdot c(A)^{\bar{u}} \cdot c(B)^{\bar{v}} \\ \frac{\partial c(C)}{\partial t} &= k \cdot c(A)^{\bar{u}} \cdot c(B)^{\bar{v}} \end{aligned} \quad (4.19)$$

Die erforderliche Reaktionsgeschwindigkeitskonstante  $k$  wird üblicherweise nach dem erweiterten Arrhenius berechnet (Gleichung 4.20).

$$k = A \cdot T^s \cdot \exp\left(-\frac{EA}{R \cdot T}\right) \quad (4.20)$$

Diese gibt die Reaktionsgeschwindigkeit in Abhängigkeit der Temperatur  $T$ , der Aktivierungsenergie  $EA$  und des Frequenzfaktors  $A$  an. Die Temperaturabhängigkeit des Frequenzfaktors wird durch  $T^s$  beschrieben, wobei  $s \geq 0$ . Die Parameter  $s$ ,  $A$ ,  $EA$ ,  $\bar{u}$  und  $\bar{v}$  sind Anpassungsparameter und  $R = 8.314 \text{ J}/(\text{K} \cdot \text{mol})$  die universelle Gaskonstante.

Um die Konzentrationsänderung in die Transportgleichung einzupflegen, ist es erforderlich diese anzupassen (Gleichung 4.21).

$$\frac{\partial \rho Y_i}{\partial t} + \underbrace{\nabla \cdot (\rho u Y_i)}_{\text{Konvektion}} - \underbrace{\alpha_i \Delta Y_i}_{\text{Diffusion}} = S_{Y_i} \quad (4.21)$$

Dies erfolgt über den Quell- und Senkterm  $S_{Y_i}$ , der den Konzentrationsgradienten zwischen den Zeitpunkten  $t_n$  und  $t_{n+1}$  darstellt. Äquivalent dazu wird die Änderung in der Reaktionsenthalpie

$$\dot{q}_{\text{reaction}} = - \sum \Delta h_{f,i}^0 \cdot \frac{\partial c(Y_i)}{\partial t}$$



in die Energieerhaltung (Gleichung 4.22) eingebunden.

$$\frac{\partial}{\partial t}(\rho h_r) + \nabla \cdot (\rho u h_r) = -\nabla \cdot q'' + \frac{Dp}{Dt} + \nabla \cdot (\tilde{\tau} \cdot u) + \dot{Q}_{source} \quad (4.22)$$

Dabei beschreibt  $q''$  die molekulare Wärmeleitung,  $\frac{Dp}{Dt}$  die reversible Temperaturschwankung, verursacht durch Druckschwankungen und  $\nabla \cdot (\tilde{\tau} \cdot u)$  ist der irreversible Temperaturanstieg, verursacht durch innere Reibung.  $\dot{Q}_{source}$  wird verwendet um weitere Modelle, wie z.B. Wärmestrahlung einzubinden. Da wir uns im weiteren Verlauf auf die chemischen Reaktionen konzentrieren, gilt:  $\dot{Q}_{source} = \dot{q}_{reaction}$ .

Aufgrund wechselnder Fluid-Zusammensetzungen müssen die thermodynamischen Zustandsgrößen für Wärmekapazität, Enthalpie und Entropie bestimmt werden. Dabei kann aus dem 2. Hauptsatz der Thermodynamik eine Relation aus Wärmekapazität  $c_p$ , Enthalpie  $h_r$  und Entropie  $S$  hergestellt werden (Gleichung 4.23).

$$c_p = \left( \frac{\partial h_r}{\partial t} \right)_{p=\text{const.}} \quad (4.23)$$

$$dS = \frac{dh_r}{T} \Big|_{p=\text{const.}} = \frac{c_p}{T} dT \Big|_{p=\text{const.}}$$

D.h. ist einer dieser Parameter bekannt, können die restlichen daraus berechnet werden. Das JANAF-Modell bietet die Möglichkeit, die Wärmekapazität über die Eingabe zweier Polynome 4. Ordnung mit Koeffizienten  $a_i$  zu approximieren. Diese beschreiben die temperaturabhängige Wärmekapazität in ineinander übergehenden Temperaturintervallen. Darüber hinaus könne Entropie und Enthalpie eines Stoffes mit wenig Mehraufwand in Abhängigkeit der Temperatur bestimmt werden (Gleichung 4.24). In der Basisversion dieses Ansatzes werden die molaren Zustandsgrößen bestimmt. Daher ist es erforderlich, diese mit der molaren Masse  $M$  zu dividieren.

$$c_p(T) = \frac{R}{M} \cdot \left( \sum_{i=1}^5 a_i \cdot T^{i-1} \right)$$

$$h_r(T) = \frac{R}{M} \cdot \left( a_6 + \sum_{i=0}^5 \frac{a_i \cdot T^i}{i} \right) \quad (4.24)$$

$$S(T) = \frac{R}{M} \cdot \left( a_1 \cdot \ln(T) + \sum_{i=1}^4 \frac{a_{i+1} \cdot T^i}{i} + a_7 \right)$$

Die thermodynamische Zustandsgröße für die Wärmekapazität wird für jede Zelle anteilig der Konzentrationsverteilung bestimmt (Gleichung 4.25).

$$c_p(T) = \sum c_{p,i}(T) \cdot \frac{c(Y_i)}{\sum c(Y_i)} \quad (4.25)$$

Für die Enthalpie und für die Entropie kann analog vorgegangen werden.[109, 124]

## 4.4 Kinetik bei der Pyrolyse von Lignocellulose

Das Ziel dieser Arbeit ist die Charakterisierung des Doppelschneckenmischreaktors. Bislang wurden die Punkte Feststofftransport, Vermischung und Wärmeübergang mittels DEM-Simulation behandelt. Jedoch ist für die Charakterisierung des Doppelschneckenmischreaktors auch eine Untersuchung der Reaktionsprozesse und deren Verteilung erforderlich. Dieses Unterkapitel betrachtet die dafür erforderlichen Grundlagen in Form von Reaktionsnetzwerk, Kinetik und deren Parameter sowie deren Ermittlung, bei der Schnellpyrolyse von Lignocellulose im Doppelschneckenmischreaktor.

In einem ersten Schritt wird ein geeigneter Reaktionsmechanismus gewählt und, anhand dessen Parameter für die Aktivierungsenergien und Frequenzfaktoren bestimmt. In einem weiteren Schritt erfolgt die Bestimmung der molaren Massen der Pyrolyseprodukte, die in den Strömungssimulationen eine wichtige Kenngröße darstellen.

### 4.4.1 Reaktionsmechanismus bei der Pyrolyse

Bei der Reaktionsbeschreibung der Pyrolyse gibt es zwei gegensätzliche Ansätze. Ranzi et. al. [125] versuchten die Pyrolyse in ihrer Gänze durch Elementarreaktionen darzustellen. Dieser Ansatz ist mittlerweile auf über 400 Komponenten (Debiagi et. al. [126]) angewachsen. Für eine Strömungssimulation, bei der die Transportgleichung für jede Komponente separat gelöst werden muss, stellt dies eine große Herausforderung hinsichtlich Rechenleistung und Rechenzeit dar.

Alternativ wird ein Ansatz verfolgt, der nur Produkt und die Edukte als Komponenten betrachtet. Eine Unterscheidung der Produkte wird dabei hinsichtlich des Aggregatzustandes getroffen (Abbildung 4.5).

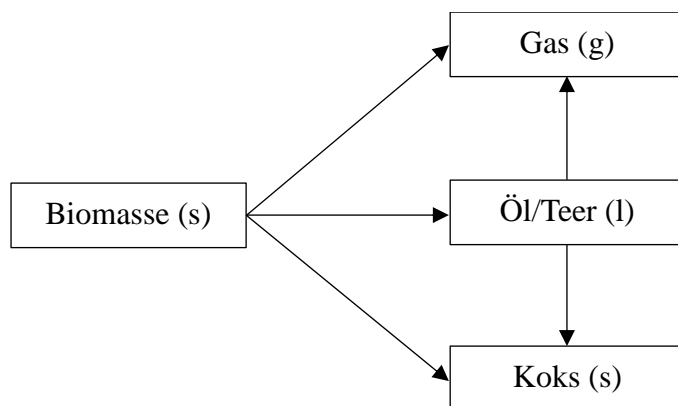
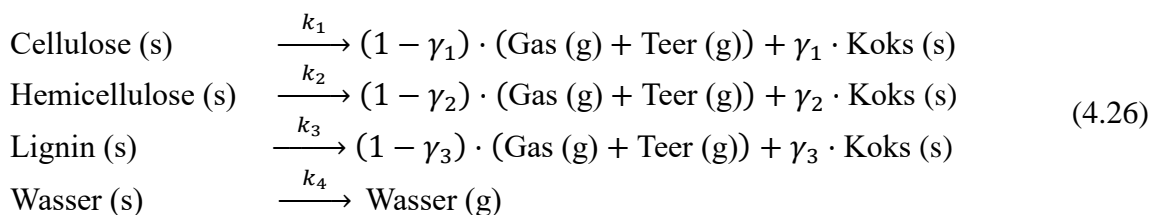


Abbildung 4.5: Reaktionsschema der Pyrolyse von Biomasse nach DiBlasi et. al. [127]

Variationen dieses Ansatzes gehen auf die unterschiedlichen Biomassezusammensetzungen ein. Aufgrund der unterschiedlichen Reaktionsregime ist auf diese Weise das Reaktionsverhalten besser approximierbar [128-130]. Darauf basierend erfolgte die Wahl der Reaktionsnetzwerkes (Gleichung 4.26). Zuzüglich zu den Biomassebestandteilen Lignin, Cellulose und Hemicellulose spielt die Restfeuchte, d.h. das in der Biomasse gebundene Wasser, im Reaktionsverhalten eine gewichtige Rolle und wurde daher als Reaktion erster Ordnung berücksichtigt.



Bei der Pyrolyse der Biomasse entsteht neben Koks auch Gas und Teer. Bei Raumtemperatur lassen sich Teer (l) und Gas (g) aufgrund ihres Aggregatzustandes unterscheiden. Jedoch sind beide Produkte im Reaktor bei 500 °C gasförmig und werden im Zuge dieser Arbeit zu dem Begriff Pyrolysegas zusammengefasst.

#### 4.4.2 Reaktionsparameterbestimmung bei der Pyrolyse von Weizenstroh

Die Pyrolysereaktionskinetik wird häufig unter Verwendung der Thermogravimetrie untersucht. Diese liefert auch bei weit über 500°C zuverlässige Onlinemessdaten des Feststoffgewichtes [128-130]. Für die Pyrolyse von Biomasse bedeutet dies, dass die Verdampfung von Wasser, der Zerfall von Hemicellulose, Cellulose und Lignin sowie indirekt die Bildung von Koks zu beobachten ist. Die Zerfallsreaktionen werden dabei durch Reaktionen 1. Ordnung in Abhängigkeit der Konzentrationen betrachtet (Gleichung 4.27).

$$\begin{aligned} \frac{dc_i}{dt} &= -k_i \cdot c_i \\ \frac{dc_{Koks}}{dt} &= \sum \gamma_i \cdot k_i \cdot c_i \\ \text{mit } k_i &= A_i \cdot \exp\left(\frac{-EA_i}{R \cdot T}\right) \end{aligned} \quad (4.27)$$

Die Ermittlung der Parameter erfolgt mittels Parameterschätzung. Dabei müssen die Werte für Aktivierungsenergie  $EA_i$ , Frequenzfaktor  $A_i$  und Verkohlungsrate  $\gamma_i$  für alle Biomassenbestandteile bestimmt werden. Trivialerweise gilt:  $\gamma_{Wasser} = 0$ . Jedoch unterscheiden sich die Biomassezusammensetzungen innerhalb der gleichen Biomasseart [14, 131-133]. Dies kann auf die variierende Boden- und Witterungsbeschaffenheit, aber auch auf die Analysemethoden zurückgeführt werden.

Tabelle 4.4: Startwerte der Parameterschätzung. LB:=lower Boundary, UB:=upper Boundary und X=Startwert

	Wasser			Cellulose			Hemicellulose			Lignin		
	LB	X	UB	LB	X	UB	LB	X	UB	LB	X	UB
$c$ [% <sub>m</sub> ]	0.06	0.083	0.12	0.3	0.389	0.6	0.1	0.164	0.5	0.2	0.308	0.5
$A$ [s <sup>-1</sup> ]	0	$9.6 \cdot 10^6$	$\infty$	0	$4.7 \cdot 10^7$	$\infty$	0	$4.5 \cdot 10^7$	$\infty$	0	$4.8 \cdot 10^4$	$\infty$
$EA$ [ $\frac{J}{mol}$ ]	$10^3$	$6.6 \cdot 10^4$	$10^7$	$10^3$	$10^5$	$10^7$	$10^3$	$1.4 \cdot 10^5$	$10^7$	$10^3$	$7.8 \cdot 10^4$	$10^7$
$\gamma$	0	0	0	0	0.001	1	0	0.5	1	0	0.5	1

Unter den Start- und Randbedingungen aus Tabelle 4.4 sowie dem verwendeten Levenberg-Marquardt-Algorithmus erreicht die Parameterschätzung ein Bestimmtheitsmaß von  $R^2 = 0.99879$  (Abbildung 4.6).

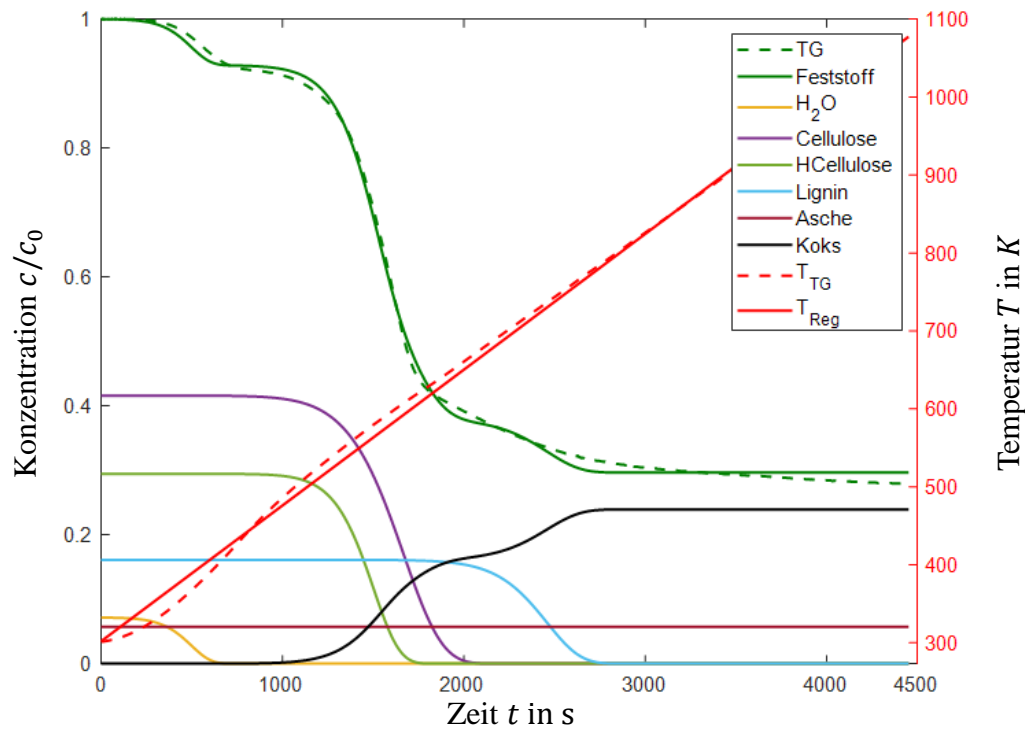


Abbildung 4.6: Thermogravimetrische Analyse von Biomasse von 0.2 g Weizenstroh, bei einer Heizrate von 10 K/min in einer NETZSCH STA 409 C/CD und deren Auswertung.

Ein Vergleich der gemessenen und berechneten Zusammensetzung zeigt insgesamt eine gute Übereinstimmung mit minimalen Schwankungen bei der Cellulose, Hemicellulose und Feuchtigkeit (Tabelle 4.5).

Tabelle 4.5: Vergleich der Weizenstrohzusammensetzung.

	Wasser	Cellulose	Hemicellulose	Lignin	Asche
Simulation	0.07	0.42	0.29	0.16	0.06
Gemessen	0.08	0.39	0.31	0.16	0.06

Größere Unterschiede können bei den Frequenzfaktoren und Aktivierungsenergien im Vergleich zur Literatur festgestellt werden (Tabelle 4.6). Diese Schwankungen können jedoch auch innerhalb der Literaturwerte wiedergefunden werden.

Die Verdunstung der Restfeuchte wurde als Reaktion erster Ordnung implementiert. Auch hier liefert der Arrheniusansatz, mit dem Wertepaar  $9.62 \cdot 10^6 \text{ s}^{-1}$  und  $67 \text{ kJ mol}^{-1}$  für Frequenzfaktor  $A$  und Aktivierungsenergie  $EA$ , eine gute Übereinstimmung.

Tabelle 4.6: Kinetikparameter der Weizenstrohpyrolyse im TG/DTG

	Cellulose		Hemicellulose		Lignin	
	$A_1$ $\left[\frac{1}{s}\right]$	$EA_1$ $\left[\frac{kJ}{mol}\right]$	$A_2$ $\left[\frac{1}{s}\right]$	$EA_2$ $\left[\frac{kJ}{mol}\right]$	$A_3$ $\left[\frac{1}{s}\right]$	$EA_3$ $\left[\frac{kJ}{mol}\right]$
<b>Ermittelt</b>	$4.67 \cdot 10^4$	79.3	$4.72 \cdot 10^7$	106	$4.47 \cdot 10^7$	139
<b>Chen et. Al. [133]</b>	$1.37 \cdot 10^{13}$	173	$1.74 \cdot 10^{12}$	148	$2.11 \cdot 10^{13}$	193
<b>Burhenne et. Al. [131]</b>	$2.69 \cdot 10^{17}$	236	$2.51 \cdot 10^6$	100	3.8	46
<b>Cai et. Al. [134]</b>	$6.31 \cdot 10^{13}$	204	$6.49 \cdot 10^{15}$	241	$5.012 \cdot 10^{12}$	176

Neben den Konzentrationen der Biomassebestandteile werden auch deren Verkohlungsraten, d.h. der Anteil einer Komponente, der in Koks umgewandelt wird, (Tabelle 4.7) bestimmt.

Tabelle 4.7: Verkohlungsrate  $\gamma_i$

	$\gamma_i$
<b>Cellulose</b>	0.25
<b>Hemicellulose</b>	0.20
<b>Lignin</b>	0.49

#### 4.4.3 Ermittlung der molaren Massen bei der Schnellpyrolyse

Die im vorigen Kapitel ermittelten kinetischen Parameter sind massebasiert. D.h. sie beschreiben die Gewichtsänderung bei der Schnellpyrolyse von Weizenstroh. Da Informationen über Mol-Änderungen fehlen, können die ermittelten Werte bei der Strömungssimulation, in welcher sie als Quellterme eingebunden werden, nicht verwendet werden. Zur Umrechnung werden die mittleren molaren Massen der Produkte benötigt.

Sowohl die Wahl der Biomasse, als auch die Betriebsweise, z.B. Temperatur und Verweilzeit, haben Einfluss auf die Produktzusammensetzung. Insofern ist zu erwarten, dass diese variieren. Da Feststoffe keinen gravierenden Einfluss auf das Gasvolumen und dadurch auf die Strömungsgeschwindigkeit ausüben, genügt es diesen groß zu wählen. Jedoch ist die Bestimmung der molaren Masse für die gasförmigen Produkte unerlässlich.

Kornmayer untersuchte im Zuge seiner Dissertation die volumetrische Produktbildung anhand eines Bach-Pyrolysereaktors [1]. Dabei verfolgte er die Pyrolyse anhand der Druckänderung im geschlossenen Reaktor. Jedoch werden Reaktionen höherer Ordnung mit steigenden Druck gefördert. Diese reduzieren die Teilchenzahl wodurch der Druck sinkt.

Im Zuge dieser Arbeit wurde Kornmayers Ansatz aufgegriffen. Dabei wird die Pyrolyse im semi-kontinuierlichen Bach-Pyrolysereaktor betrachtet. Der Feststoff stellt die nicht-kontinuierliche Phase dar. Das sich bildende Produktgas wird kontinuierlich mittels Stickstoff aus dem Reaktor gespült. Aufgrund der Produkteigenschaften, d.h. Temperatur, Kondensation und geringer Feststoffanteil, kann das Pyrolysegas nicht unmittelbar nach dem Reaktor analysiert werden. Im Zuge der Nachbehandlung wird zunächst der Feststoffanteil in einem Zyklon abgetrennt, bevor das Pyrolysegas unter Zugabe von Luft-Sauerstoff in einem Oxidationsreaktor vollständig in Kohlenstoffdioxid umgewandelt werden kann. Dies macht eine direkte Messung der Zusammensetzung unmöglich, jedoch erlaubt es das in CO<sub>2</sub> umgewandelte Pyrolysegas ohne Kondensation auf 120°C abzukühlen, sodass es online in einem Massenspektrometer gemessen werden kann (Abbildung 4.7).

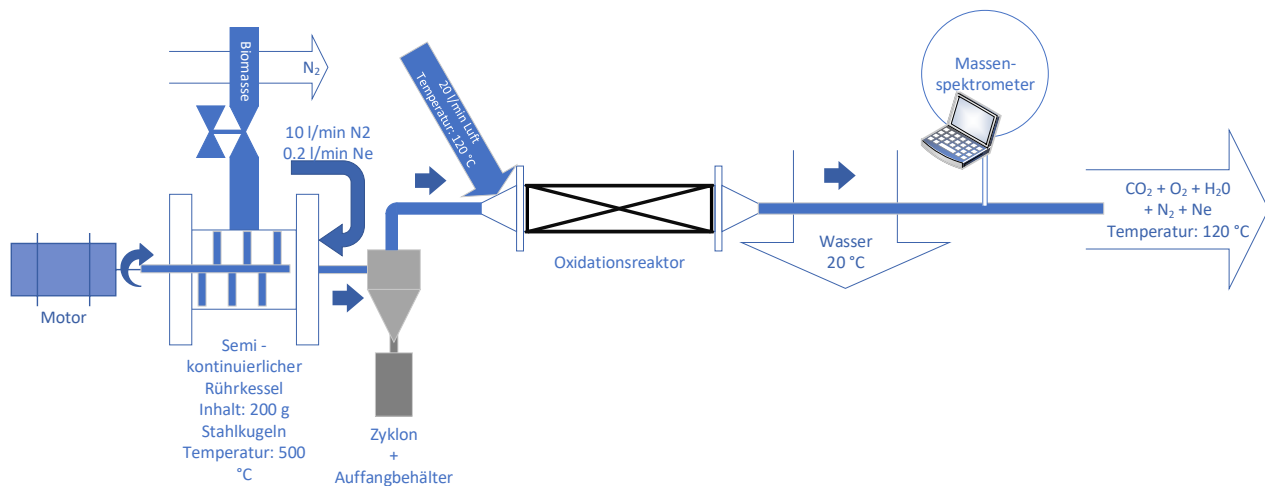


Abbildung 4.7: Versuchsaufbau zur semi-kontinuierlichen Pyrolyse von Biomasse

Mit Kenntnis der Transportprozesse im Oxidationsreaktor, dem  $\text{CO}_2$ -Konzentrationsverlauf und der Reaktionskinetik können über die Verweilzeitverteilung im semi-kontinuierlichen Bach-Pyrolyse-reaktor Rückschlüsse auf die mittlere molare Masse von Pyrolysegas gezogen werden.

#### 4.4.3.1 Transportprozesse im Oxidationsreaktor

Der Oxidationsreaktor ist ein begleitbeheizter Rohrreaktor, mit einem Innendurchmesser von 10 cm und einer Länge von 1 m, gefüllt mit dem Katalysator. Brar [135] untersuchte in diesem Zusammenhang mehrere Katalysatoren auf ihre Eignung zur vollständigen Oxidation von Pyrolysegas. Dabei zeichnete sich der VOC-Katalysator der Marke Haldor Topsøes CK-322 aus. Dieser ist ein CuMn-Oxid-Katalysator auf einem Aluminiumoxid-Raschigring. Durchgeführte GC-Analysen schlossen organische Rückstände in der Gasphase aus.

Der Transport von Fluiden im Rohrreaktor folgt im Allgemeinen der Konvektions-Diffusionsgleichung (Gleichung 4.28)

$$\frac{dc}{dt} = D_{ax} \cdot \nabla^2 c - u \cdot \nabla c \quad (4.28)$$

Unter Verwendung der Parameterschätzung kann dieses Model als Grundlage zur Berechnung des axialen Dispersionskoeffizienten verwendet werden. Zum Abgleich wurden Versuche (Tabelle 4.8) durchgeführt.

Tabelle 4.8: Rahmenbedingungen zur Charakterisierung der Transportprozesse

$\dot{V}_{Ne}(20^\circ\text{C})$	=	0,2	l/min
$\dot{V}_{N_2}(20^\circ\text{C})$	=	10	l/min
$\dot{V}_{Luft}(20^\circ\text{C})$	=	20	l/min
$T_{Reaktor}$	=	500	°C

Dabei wird der Oxidationsreaktor kontinuierlich mit Stickstoff (mit 500°C) und Luft (mit 120°C) gespült. Zu einem Zeitpunkt  $t = 0$  s wird der Marker Neon (mit 500°C) kontinuierlich zugegeben. Mit einem Massenspektrometer wird die Neon-Konzentration online gemessen (Abbildung 4.8). Die resultierende Verteilungsfunktion wird im Zuge der Parameterschätzung zum Abgleich verwendet.

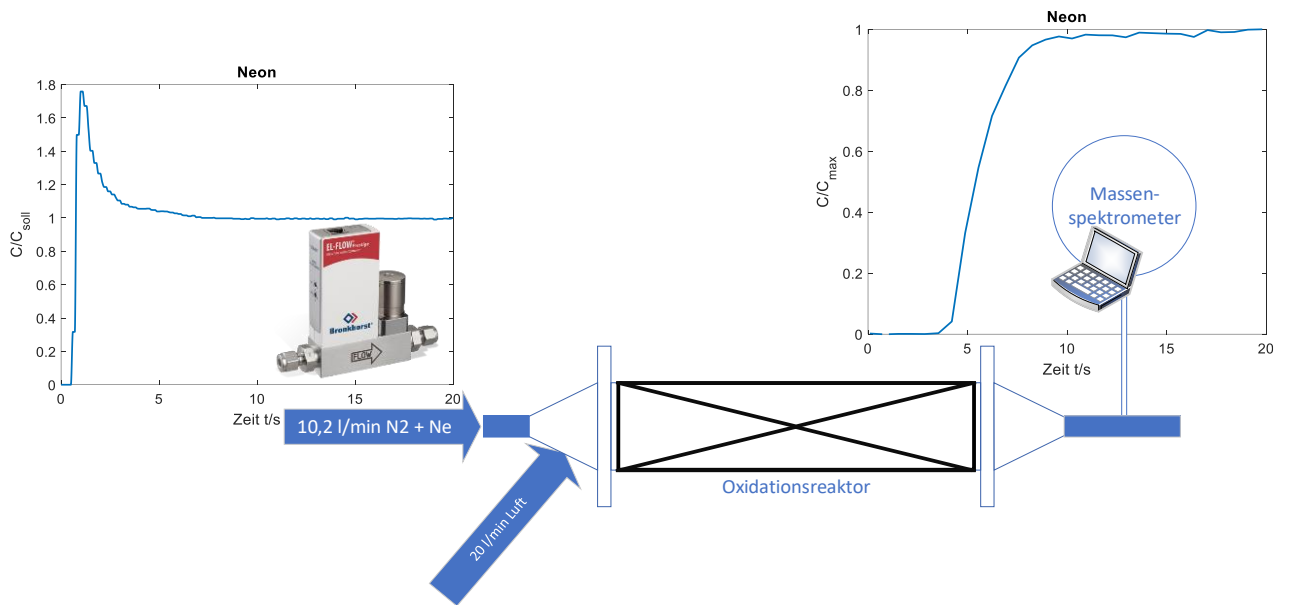


Abbildung 4.8: Versuchsaufbau zur Bestimmung der axialen Dispersion

Im Gegensatz zum idealen Reaktor, kann ein realer Reaktor Kurzschlussströmungen oder Totzonen aufweisen. Daher werden für reale Reaktoren häufig mehrere Transportgleichungen parallel oder auch seriell geschaltet, um die Qualität des Modells zu verbessern. Durch den liegenden Reaktor sind vertikal unterschiedliche Katalysatorkonzentrationen denkbar. Aus diesem Grund wurden mehrere Konvektions-Diffusionsgleichungen parallel geschaltet. Für jeden dieser Stränge muss der Anteil am Gesamtvolumenstrom, der axiale Dispersionskoeffizient und die Verweilzeit oder auch Strömungsgeschwindigkeit neu berechnet werden. D.h. für  $n$  Stränge muss die Parameterschätzung  $3n - 1$  Parameter bestimmen.

Unter diesem Sachverhalt wurden Parameter für bis zu drei Stränge bestimmt. Dabei zeigt sich, dass die Werte für das Zwei- und Dreistrangmodell im ersten und zweiten Strang nahezu übereinstimmen. Darüber hinaus ist das Bestimmtheitsmaß beim Zweistrangmodell am höchsten, sodass davon auszugehen ist, dass zwei Stränge zur Beschreibung genügen (Abbildung 4.10).

Tabelle 4.9: Modellgüte der parallelen axialen Dispersion

Stränge	1	2		3		
$R^2$	0.9941	0.9972		0.9966		
$\alpha_i$ [%]	100	29	71	28	71	1
$\tau$ [s]	5.5	7.0	5.2	6.9	5.2	4.4
$D_{ax}$ [ $m^2 \cdot s^{-1}$ ]	0.25	1.40	0.16	1.43	0.14	3.94

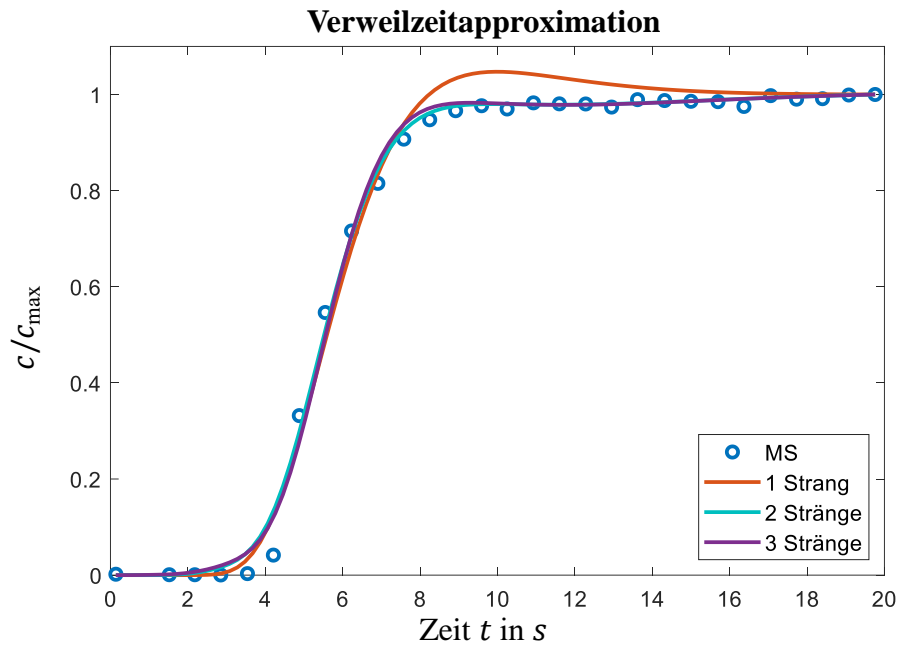


Abbildung 4.9: Verweilzeitverteilung des Dispersionsmodells in Abhängigkeit paralleler Stränge

#### 4.4.3.2 Ermittlung der molaren Massen bei der semi-kontinuierlichen Pyrolyse

In den vergangenen Kapiteln wurde das Reaktionsnetzwerk aufgestellt und die erforderlichen Parameter bestimmt. Darüber hinaus wurde ein Verweilzeitmodell aufgestellt und die erforderlichen Dispersionskoeffizienten bestimmt.

Bei Betrachtung der Primärreaktionen, d.h. die Zersetzung der Biomasse, fällt auf, dass sich die molaren Massen auf beiden Seiten kürzen (Gleichung 4.29).

$$\begin{aligned}
 \frac{dm_X}{dt} &= -k_X \cdot m_X \\
 \frac{dn_X \cdot M_X}{dt} &= -k_X \cdot n_X \cdot M_X \\
 \frac{dn_X}{dt} &= -k_X \cdot n_X
 \end{aligned}
 \tag{4.29}$$

Für die Pyrolyseprodukte kann jedoch gezeigt werden, dass diese von der molaren Produktmasse, als auch von den molaren Massen der Edukte abhängen. (Gleichung 4.30). Da Koks keinen signifikanten Einfluss auf die Änderung des Gasvolumens ausübt, kann dessen Wert für die molare Masse  $M_K$  auf  $10^{32} \frac{g}{mol}$  festgelegt werden.



$$\begin{aligned}
\frac{dm_K}{dt} &= \alpha_C \cdot k_C \cdot m_C + \alpha_H \cdot k_H \cdot m_H + \alpha_L \cdot k_L \cdot m_L \\
\frac{dn_K \cdot M_K}{dt} &= \alpha_C \cdot k_C \cdot n_C \cdot M_C + \alpha_H \cdot k_H \cdot n_H \cdot M_H + \alpha_L \cdot k_L \cdot n_L \cdot M_L \\
\frac{dn_K}{dt} &= \underbrace{\frac{\alpha_C \cdot M_C}{M_K}}_{=\psi_C} \cdot k_C \cdot n_C + \underbrace{\frac{\alpha_H \cdot M_H}{M_K}}_{=\psi_H} \cdot k_H \cdot n_H + \underbrace{\frac{\alpha_L \cdot M_L}{M_K}}_{=\psi_L} \cdot k_L \cdot n_L \\
\frac{dn_G}{dt} &= \underbrace{\frac{\beta_C \cdot M_C}{M_G}}_{=\omega_C} \cdot k_C \cdot n_C + \underbrace{\frac{\beta_H \cdot M_H}{M_G}}_{=\omega_H} \cdot k_H \cdot n_H + \underbrace{\frac{\beta_L \cdot M_L}{M_G}}_{=\omega_L} \cdot k_L \cdot n_L \\
&\text{mit } 1 = \alpha_X + \beta_X \quad \forall X
\end{aligned} \tag{4.30}$$

Jedoch erfordert die Berechnung der molaren Masse für das Pyrolysegas  $M_G$  die Kenntnis der molaren Massen von Lignin  $M_L$ , Cellulose  $M_C$  und Hemicellulose  $M_H$  (Gleichung 4.30). Diese wurden der Literatur (Tabelle 4.10) für deren Monomere entnommen, sodass nur die molare Masse von Pyrolysegas bestimmt werden muss.

Tabelle 4.10: Summenformel und Molmassen der Edukte bei der Pyrolyse von Biomasse

	Monomer	M [kg · mol <sup>-1</sup> ]	Quelle
<b>Cellulose</b>	$C_{12}H_{20}O_{10}$	324	[136]
<b>Hemicellulose (Xylan)</b>	$C_5H_8O_4$	134	[137]
<b>Lignin</b>	$C_{31}H_{34}O_{11}$	582	[138]

Innerhalb des Versuchsablaufs wird mit Stickstoff inertisierte Biomasse zum Zeitpunkt  $t_0$  in den Reaktor gegeben. Es reagiert und das entstehende Pyrolysegas wird aus dem Reaktor in den Zyklon gespült. Im Zyklon wird das Pyrolysegas vom restlichen Koks befreit und zusammen mit Luft-Sauerstoff in den Oxidationsreaktor geleitet. Das Pyrolysegas wird vollständig oxidiert und auf 120 °C abgekühlt. Zuletzt wird der  $CO_2$  und  $Ne$ -Konzentrationsverlauf im Massenspektrometer erfasst. Das Neon wird zusammen mit dem Stickstoff zum Spülen des semi-kontinuierlichen Pyrolysereaktors verwendet und dient als Marker. Dosierte wird das Neon mittels MFC (Mass Flow Controller) EL-FLOW Prestige der Marke Bronkhorst. Dieser ist bekannt für seinen geringen Fehler ( $\pm 0.5 \%_m$ ). Die bekannte Volumenstromgröße ermöglicht das Abschätzen des  $CO_2$ -Verlaufs anhand der MS-Messdaten.

Da die Versuchsdurchführung nicht computergesteuert erfolgt, ist der Zeitpunkt  $t_0$  unbekannt. Darüber hinaus ist die mittlere Verweilzeit durch den höheren Volumenstrom und die Strömungsgeschwindigkeit eine weitere Unbekannte. Zusammen mit den molaren Massen von Pyrolysegas ergeben sich drei Unbekannte, die mittels Parameterschätzung zu bestimmen sind. Dabei wird angenommen, dass die Umwandlung von Pyrolysegas in  $CO_2$  unmittelbar am Eintritt des Oxidationsreaktors erfolgt.

Eine Herausforderung bei der Beschreibung des kompletten Prozesses stellt die Temperaturabhängigkeit der Geschwindigkeitskonstante  $k$  dar. Idealerweise wird diese für jedes Biomassepartikel aus der DEM-Simulation entnommen. Jedoch kommt es bei der Parameterschätzung nach Newton zu mehreren Funktionsaufrufen und einer nicht absehbaren Anzahl an Iterationen, sodass eine große Anzahl nahezu identischer DEM-Simulationen durchgeführt werden müsste, um den Temperaturverlauf darstellen zu können. Dies hätte einen nicht tragbaren Rechenaufwand zufolge. Daher wurde der mittlere Temperaturverlauf durch ein Polynom 12.

Ordnung approximiert, welches im Zuge der Parameterschätzung verwendet wurde (Abbildung 4.10).

### Temperaturprofil der Biomassepartikel

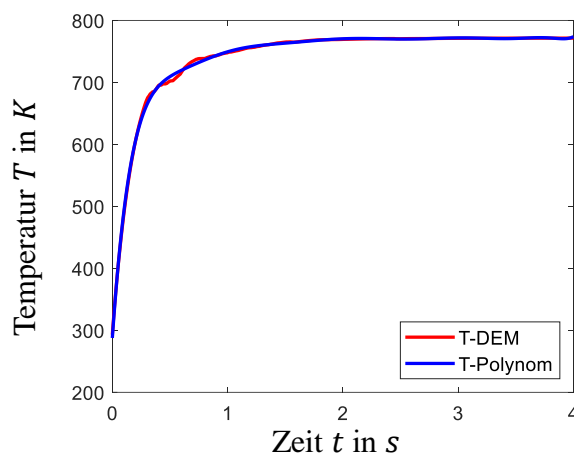


Abbildung 4.10: Temperaturprofil eines Biomassepartikels im semi-kontinuierlichen Pyrolysereaktor

Die dazugehörigen Rahmenbedingungen (Tabelle 4.11) sind identisch zur Versuchsdurchführung.

Tabelle 4.11: Rahmenbedingungen des Versuchsablaufs

<b><u>Wärmeträger</u></b>		<b><u>Biomasse</u></b>	
Masse	300 g	Masse	1 g
Durchmesser	2 mm	Durchmesser	Tabelle 2.13
Verteilung	1	Verteilung	
Temperatur	500 °C	Temperatur	20 °C
<b><u>Motor</u></b>			
Drehzahl	2 rpm		
<b><u>Einlass: Pyrolysereaktor</u></b>		<b><u>Luftstrom: Oxidationsreaktor</u></b>	
Temperatur	500 °C	Temperatur	120 °C
Stickstoffstrom (20°C)	10 l/min	Luftstrom (20°C)	20 l/min
Neonstrom (20°C)	0.2 l/min		

Mit dem nun vorhandenen Wissensstand wurde eine Parameterschätzung unternommen, zur Berechnung der mittleren molaren Masse von Pyrolysegas. Die verwendeten Startwerte sowie deren Grenzen wurden in Tabelle 4.12 zusammengetragen.

Tabelle 4.12: Startwerte, untere (*lb*) und obere (*ub*) Grenzwerte sowie Resultat der Parameterschätzung

Startparameter	<i>lb</i>	$\underline{x}$	<i>ub</i>	Resultat	$\underline{x}$
$M_G$ [ $g\ mol^{-1}$ ]	16	50	58	$M_G$ [ $g\ mol^{-1}$ ]	52.7
$\tau$ [s]	0	3	7	$\tau$ [s]	4.7
$t_0$ [s]	0	34.476	120	$t_0$ [s]	34.02

Das Resultat (Tabelle 4.12) beschreibt den Versuchsablauf mit einer Güte von  $R^2 = 0.994$  (Abbildung 4.11).

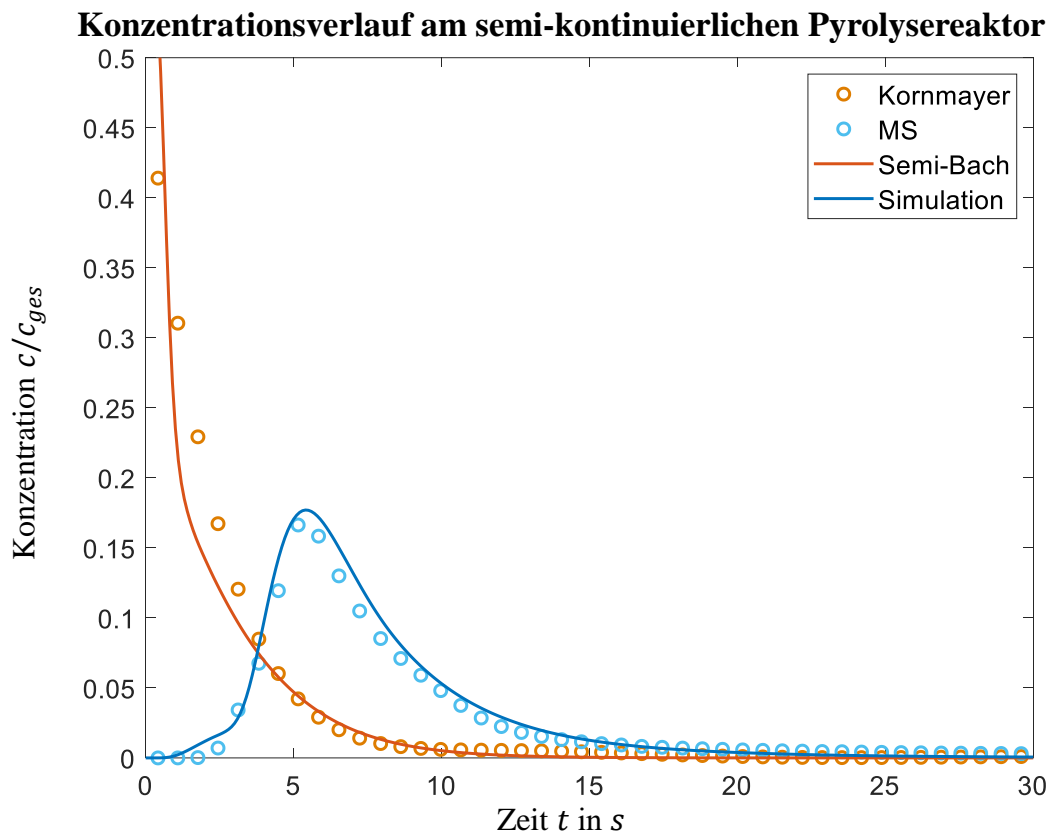


Abbildung 4.11: Konzentrationsverlauf am semi-kontinuierlichen Pyrolysereaktor. Gemessen vs. Berechnet. Blau: Konzentrationsverlauf am Oxidationsreaktor Ausgang. Rot: Pyrolysegas-Konversionsrate im Pyrolysereaktor

Unter Verwendung der ermittelten Werte und Reaktionsmechanismen wurde ein Reaktormodell nach Kornmayer aufgestellt und der Reaktionsverlauf ermittelt. Dabei wird eine Güte von  $R^2 = 0.9118$  erreicht (Abbildung 4.11).

## 4.5 Solver foamPy

Eine Simulation der Pyrolyse im Doppelschneckenmischreaktor unter Berücksichtigung der Pyrolysegases und mechanisch durchmischten Feststoffschüttung, sowie deren chemischen Reaktionen stellt eine große Herausforderung dar. Jedoch erlaubt eine Kombination der vorgestellten Methoden einen entscheidenden Schritt in diese Richtung (Abbildung 4.12).

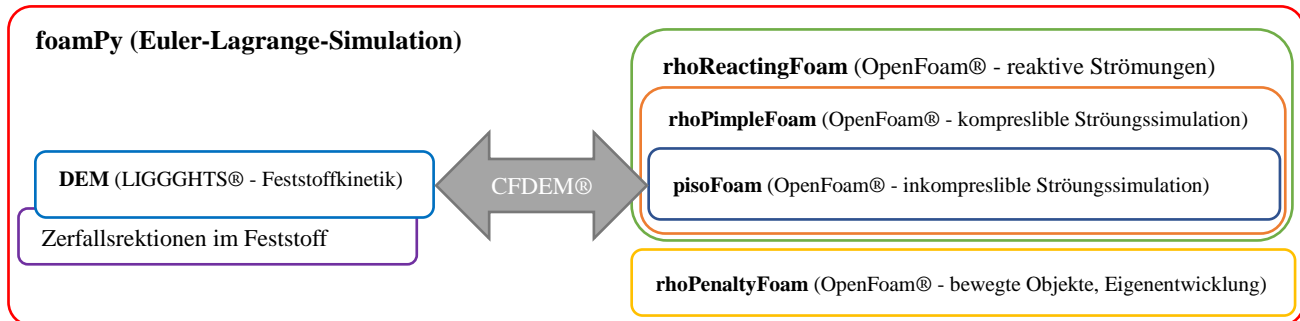


Abbildung 4.12: Hierarchischer Aufbau des foamPy-Solvers aus den Modulen.  
 Links: Lagrange-Modul LIGGGHTS® mit Erweiterung,  
 Rechts: Euler-Module piso-, rhoPimple-, rhoReactingFoam Module samt rhoPenaltyFoam-  
 Erweiterung

Dieses Kapitel befasst sich mit der genauen Implementation dieser Methoden. Das Resultat wird im weiteren Verlauf OpenFOAM für Pyrolyse oder auch foamPy genannt.

### 4.5.1 DEM – Implementation

Grundlagen der DEM-Simulation wurden bereits in Kapitel 2 vorgestellt. Dieses deckte weitestgehend alle Anforderungen zur Simulation der Partikelphase ab. Jedoch macht eine Betrachtung der chemischen Reaktionen, deren Implementierung in LIGGGHTS-Public® erforderlich. Kapitel 4.3 befasste sich mit der Reaktionskinetik bei der Pyrolyse von Weizenstroh. Das entwickelte Modell betrachtete Reaktionen erster Ordnung bei der thermischen Zersetzung von Weizenstroh in gasförmigen Pyrolyseprodukte. Da die Primärreaktion entscheidenden Einfluss auf die Biomassepartikelgröße ausübt, ist es erforderlich diese zu integrieren. Die entstandenen Pyrolyseprodukte werden als Quellterm in die CFD-Simulation übergeben und in der Transportgleichung der Komponenten implementiert. Mehr dazu im nachfolgenden Kapitel.

Im vorhergehenden Kapitel wurde sowohl die Reaktionskinetik als auch die massenbasierte prozentuale Zusammensetzung von Weizenstroh bestimmt. Der massenbezogene Anteil  $Y_{i,j}$  einer Reaktionskomponente  $j$  am Partikel  $i$  ist Zeitabhängig. Trivialerweise ist die Summe aller Anteile zu jedem Zeitschritt  $t_n$  eins (Gleichung 4.31). Damit setzt sich die Komponentenmasse  $m_{i,j}$  für jeden Zeitschritt  $t_n$  aus dem Massenanteil  $Y_{i,j}$  und der Partikelmasse  $m_i$  zusammen (Gleichung 4.32).

$$1 = \sum_j Y_{i,j}(t_n) \quad \text{mit } Y_{i,j}(t_n) \leq 1 \quad \forall i, n, j \in \mathbb{N} \quad (4.31)$$

$$m_{i,j}(t_n) = m_i(t_n) \cdot Y_{i,j}(t_n) \quad (4.32)$$

Bei der Zersetzung von Molekülen handelt es sich i. a. um Reaktionen erster Ordnung. Unter der Verwendung des Arrhenius-Ansatzes (Gleichung 4.27) kann der Massenanteil  $m_{i,j}(t_{n+1})$  des nächsten Zeitschrittes bestimmt werden (Gleichung 4.33).

$$m_{i,j}(t_{n+1}) = \int_{t_n}^{t_{n+1}} -k_j(T_i(t_n)) \cdot m_{i,j}(t_n) dh_n \quad \text{mit } h_n = t_{n+1} - t_n \quad (4.33)$$

Als numerischer Integrator der chemischen Reaktion dient ein Rosenbrockverfahren 4er Ordnung [139]. Dieses eignet sich zur Lösung steifer Systeme und besticht durch seine Präzision (Tabelle 4.13).

Tabelle 4.13: Abweichung eines Zeitschrittes zweier Integrationsalgorithmen zur analytischen Lösung

	Ordnung	Abweichung [%]		
<b>Rosenbrock</b>	4	$1.77 \cdot 10^{-7}$	$T =$	742 K
<b>Heun</b>	2	$1.22 \cdot 10^{-3}$	$A =$	$46.7 \cdot 10^3 \frac{1}{s}$
			$EA =$	$79.3 \frac{kJ}{mol}$
			$dt =$	0.001 s
			$k =$	$0.122 \frac{1}{s}$
			$c_0 =$	$1.168 \cdot 10^8 \text{ mol}$

Beim Übertrag der Informationen von DEM-Simulation nach CFD-Simulation wird der Konzentrationsgradient der Komponenten verwendet. Durch die Verwendung eines Geschlossenen Algorithmus der boost-Library [140] steht dieser jedoch nicht zur Verfügung und muss mittels Differential (Gleichung 4.34) berechnet werden.

$$\frac{\partial m_{i,j}}{\partial t}(t_{n+1}) = \frac{m_{i,j}(t_{n+1}) - m_{i,j}(t_n)}{t_{n+1} - t_n} \quad (4.34)$$

Durch die unterschiedliche Reaktion der Komponenten verändert sich die Zusammensetzung  $Y_{i,j}$ , das Partikelvolumen  $V_i$  und damit der Radius  $r_i$ . Diese Parameter müssen vor Beginn des nächsten Zeitschrittes neu berechnet werden (Gleichung 4.35 bis 4.37).

$$V_i(t_{n+1}) = \sum_j \rho_j \cdot m_{i,j}(t_{n+1}) \quad (4.35)$$

$$Y_{i,j}(t_{n+1}) = \frac{m_{i,j}(t_{n+1})}{\sum_j m_{i,j}(t_{n+1})} \quad (4.36)$$

$$r_i(t_{n+1}) = \sqrt[3]{\frac{3 \cdot V_i(t_{n+1})}{4 \cdot \pi}} \quad (4.37)$$

Durch die mechanische Beanspruchung während des Betriebes werden Koksschichten effektiv abgetragen, sodass die Wärmeleitung ungehindert bleibt. Durch die Gasbildung bei der Zersetzung wurde auf die Implementierung weiterer Transportprozesse verzichtet und das Shrinking Particle Model (Abbildung 4.13) zur Neuberechnung der Partikelform gewählt.



Abbildung 4.13: Shrinking Particle Modell

Bei der Berechnung wurden einige Annahmen getroffen. U. a. wurde die Reaktion als Homogenreaktion im Partikel interpretiert. Diese Annahme ist in Anbetracht einer Biot-Zahl von rund eins nur eingeschränkt gültig, aber aufgrund des erforderlichen numerischen Aufwandes wird sie hingenommen.

## 4.5.2 CFD – Implementation

Mit der Implementierung der Primärreaktionen in die DEM-Simulation ist ein erster wichtiger Schritt zur Simulation der Pyrolyse von Biomasse im Doppelschneckenmischreaktor genommen worden, jedoch erfordert deren weitere Behandlung eine Berücksichtigung in allen Teilen der kompressiblen Navier-Stokes-Gleichungen und der Transportgleichungen.

Darüber hinaus ist es erforderlich das im Vorfeld vorgestellte und getestete Brinkmann Strafverfahren einzubinden. Da sowohl CFD-DEM-Simulationen, als auch das Brinkmann-Strafverfahren mit der Porosität arbeiten, müssen diese vereinigt werden (Gleichung 4.38).

$$\varepsilon_G = \varepsilon_f \cup \varepsilon_\chi \quad (4.38)$$

Beim Brinkmann-Strafverfahren ist der poröse Bereich nur in der Massenerhaltung definiert (Gleichung 4.39). Daher genügt es den vereinigten porösen Bereich  $\varepsilon_G$  darauf zu beschränken. Aufgrund der Neumann-Randbedingung (d.h.  $\nabla T = 0$ ) entfällt bei der Energieerhaltung die Notwendigkeit die Randbedingung der Schneckenoberfläche zu behandeln.

$$\frac{\partial \varepsilon_G \rho}{\partial t} + \nabla \cdot (\varepsilon_G \rho u) = S_\rho \quad (4.39)$$

$$\frac{\partial \varepsilon_f \rho u}{\partial t} + \nabla \cdot (\varepsilon_f \rho u u) = -\varepsilon_f \nabla p - S_p + \nabla \cdot (\varepsilon_f \tilde{\tau}) + \varepsilon_f \rho g + \frac{\chi}{\tilde{\eta}} (u - U_0) + S_M \quad (4.40)$$

$$\frac{\partial \varepsilon_f \rho h_r}{\partial t} + \nabla \cdot (\varepsilon_f \rho u h_r) = -\nabla \cdot \varepsilon_f q'' + \frac{D \varepsilon_f p}{Dt} + \nabla \cdot (\varepsilon_f \tilde{\tau} \cdot u) + S_{h_{chem.}} + S_T + S_{h_s} \quad (4.41)$$

$$\frac{\partial \varepsilon_f \rho Y_k}{\partial t} + \nabla \cdot (\varepsilon_f \rho u Y_k) - \varepsilon_f D_k \Delta Y_k = S_{chem.} + S_{Y_k} + \frac{\chi}{\tilde{\eta}_{Y_k}} (\rho Y_k - 0) \quad (4.42)$$

Hardy et. al. [141] definierte das Brinkmann-Strafverfahren für niedrige Machzahlen ( $Ma < 0.03$ ) und reaktive Strömungen und belegte deren Kompatibilität. Aufgrund des Phasenübergang, d.h. Feststoff→Gas, erfordert die Kombination aus DEM und CFD-Simulation das jedoch einbinden der Massen- und Energieströme.

Anfangen mit der Massenerhaltung (Gleichung 4.39) wird der Quellterm  $S_\rho$  genutzt um jeder Zelle  $i$  die Massenströme der Reaktionskomponenten  $J_j$  der in der Zelle enthaltenen Partikel  $I_i$  zuzuordnen (Gleichung 4.43).

$$S_{\rho,i} = \frac{1}{V_i} \cdot \sum_j^{I_i} \sum_k^{J_j} \frac{\partial m_{j,k}}{\partial t} \quad (4.43)$$

Damit verbunden ist ein Anstieg der lokalen Strömungsgeschwindigkeit. Diese wird mit dem Quellterm  $S_M$  (Gleichung 4.44) in der Impulserhaltung (Gleichung 4.40) eingebunden.

$$S_{M,i} = \frac{1}{V_i} \cdot \sum_j^{I_i} \sum_k^{J_j} \frac{\partial m_{j,k}}{\partial t} \cdot u \quad (4.44)$$

Bei der Energieerhaltung (Gleichung 4.41) ist es erforderlich zwei Quellen einzubinden. Diese dienen der Berücksichtigung des Energieanstieg durch das Produktgas  $S_{h_s}$  (Gleichung 4.45) und der Reaktionsenthalpie  $S_{h_m}$  (Gleichung 4.46).

$$S_{h_s,i} = \frac{1}{V_i} \cdot \sum_j^{I_i} \sum_k^{J_j} c_{p_k} \cdot \frac{\partial m_{j,k}}{\partial t} \cdot T_j \quad (4.45)$$

$$S_{h_m,i} = \frac{1}{V_i} \cdot \sum_j^{I_i} \sum_k^{J_j} \frac{\partial m_{j,k}}{\partial t} \cdot \Delta_R h_k \quad (4.46)$$

Zuletzt ist es erforderlich, die Reaktionsprodukte  $S_{Y_k}$  (Gleichung 4.47) in die Transportgleichung (Gleichung 4.42) der Reaktanten, mit einzupflegen.

$$S_{Y_k,i} = \frac{1}{V_i} \cdot \sum_j^{I_i} \frac{\partial m_{j,k}}{\partial t} \quad (4.47)$$

Das Brinkmann-Strafverfahren berücksichtigt in seiner Grundform lediglich kompressible Strömungssimulationen. D.h. bei der Simulation mehrere Produkte würden die Komponenten in den Objektraum  $\chi$  vordringen. Um dies zu vermeiden, wurde der Strafterm:  $\chi/\tilde{\eta}_{Y_k} (\rho Y_k - 0)$  eingeführt. Dieser beschränkt die Produktkonzentration im Objektraum auf ein Minimum. Da die Transportgleichung auf alle Produkte außer dem Inertgas angewandt wird, bleibt der Objektraum mit Stickstoff gefüllt.

Eine Besonderheit an dem vorgestellten Algorithmus ist, dass eine Gitterzelle sowohl den Objektrandbereich als auch Biomassepartikel enthalten kann. Da dies zu unerwünschten Interaktionen und Unstetigkeit führen kann, müssen alle Quellterme auf dem Objektrand eliminiert werden. Dies führt zu einem Massen- und Energieverlust, der im weiteren Verlauf noch diskutiert wird.

Unter der Verwendung dieser Modifikationen können erste Simulationen gestartet werden. Jedoch enden alle Simulationen in einer Endlosschleife. Die Ursache ist der in CFDEM® verwendete Suchalgorithmus. Dieser Startet in einer beliebigen Gitterzelle und hangelt sich von Nachbarzelle zur Nachbarzelle an sein Ziel. Das Auswahl Kriterium ist der geringste euklidische Abstand zum Ziel. Ist die Verbindungsgerade aufgrund von Löchern oder Lücken nicht im Gitter, kann es zu einer unendlichen Rekursion und zum Einfrieren der Simulation kommen.

Eine schnell zu programmierende und effektive Lösung ist der Zwischenschritt über ein virtuelles strukturiertes Gitter, mit würfelförmigen Gitterzellen der Seitenlänge  $ds$ . D.h. um den fluiden Raum wird eine Quader gelegt. Dieser Quader wird in würfelförmige Gitterzellen der Seitenlänge  $ds$  unterteilt. Die Position eines jeden Würfels im virtuellem Gitter kann durch die Indizes  $i, j$  und  $k$  eindeutig beschrieben werden.

Im Anschluss werden allen Zellen der Strömungssimulation anhand ihrer Zellzentren durch ganzzahlige Division (Gleichung 4.48) in das virtuelle Gitter einsortiert.

$$\left\lfloor \frac{x}{ds} \right\rfloor = i, \quad \left\lfloor \frac{y}{ds} \right\rfloor = j, \quad \left\lfloor \frac{z}{ds} \right\rfloor = k \quad (4.48)$$

Eine Voraussetzung ist dass die Koordinaten der Zellzentren positiv sind. Dies kann durch eine Verschiebung der Zellzentren um das Minimum des virtuellen Gitters gewährleistet werden.

Ein Partikel mit Position  $(x, y, z)$  kann analog dem virtuellen Gitter zugeordnet werden. Die in der virtuellen Gitterzelle enthaltenen Zellen der Stömungssimulation können mit geringen Aufwand überprüft werden ob sie das Partikel enthalten. Ist dies nicht der Fall, kann der Inhalt der Nachbarzellen im virtuellen Gitter geprüft werden usw. (Abbildung 4.14).

Schritt 1					Schritt 2				
(i-2,j-2)	(i-1,j-2)	(i,j-2)	(i+1,j-2)	(i+2,j-2)	(i-2,j-2)	(i-1,j-2)	(i,j-2)	(i+1,j-2)	(i+2,j-2)
(i-2,j-1)	(i-1,j-1)	(i,j-1)	(i+1,j-1)	(i+2,j-1)	(i-2,j-1)	(i-1,j-1)	(i,j-1)	(i+1,j-1)	(i+2,j-1)
(i-2,j)	(i-1,j)	(i,j)	(i+1,j)	(i+2,j)	(i-2,j)	(i-1,j)	(i,j)	(i+1,j)	(i+2,j)
(i-2,j+1)	(i-1,j+1)	(i,j+1)	(i+1,j+1)	(i+2,j+1)	(i-2,j+1)	(i-1,j+1)	(i,j+1)	(i+1,j+1)	(i+2,j+1)
(i-2,j+2)	(i-1,j+2)	(i,j+2)	(i+1,j+2)	(i+2,j+2)	(i-2,j+2)	(i-1,j+2)	(i,j+2)	(i+1,j+2)	(i+2,j+2)

Abbildung 4.14: Schemata des verwendeten Suchalgorithmus

Da das Befüllen der virtuellen Matrix bei einem statischen Gitter der Strömungssimulation nur einmal erfolgt, ist dieser Algorithmus sowohl schnell als auch robust. Am effektivsten ist dieser Suchalgorithmus wenn die Zellgröße der Strömungssimulation der Gitterbreite des virtuellen Gitters entspricht.

Durch die Verwendung des neuen Suchalgorithmus, bieten sich neue Möglichkeiten an. So ist es z.B. ratsam aber nicht mehr nötig, dass der Simulationsraum der Strömungs- und Partikelsimulation übereinstimmen. Verlassen einzelne Partikel den Simulationsraum der Strömungssimulation werden sie lediglich bei der Strömungssimulation nicht mehr berücksichtigt.



### 4.5.3 Rahmenbedingung zur Simulation des Doppelschneckenmischreaktors

Die vergangenen Kapitel beschäftigten sich mit der Modellentwicklung und der Parameterbestimmung mit dem Ziel der Simulation der Pyrolyse von Weizenstroh im Doppelschneckenmischreaktor. Da die Reaktionsmodelle und Parameter über das gesamte Kapitel verteilt sind, wurde an dieser Stelle eine kompakte Übersicht (Tabelle 4.14-Tabelle 4.16) geschaffen und in diesem Zuge einzelne Lücken mit Literaturwerten geschlossen. Des Weiteren werden die Einstellungen und Besonderheiten des vorgestellten foamPy-Solvers behandelt.

Tabelle 4.14: Reaktionen der Biomasse

Reaktion	Reaktionsmechanismus
$k_1$	Cellulose $\rightarrow \alpha_1 \cdot \text{Koks} + (1 - \alpha_1) \cdot (\text{Gas} + \text{Pyrolysegas})$
$k_2$	Hemicellulose $\rightarrow \alpha_2 \cdot \text{Koks} + (1 - \alpha_3) \cdot (\text{Gas} + \text{Pyrolysegas})$
$k_3$	Lignin $\rightarrow \alpha_3 \cdot \text{Koks} + (1 - \alpha_3) \cdot (\text{Gas} + \text{Pyrolysegas})$
$k_4$	Wasser (s) $\rightarrow$ Wasser (g)
$k_5$	Pyrolysegas $\rightarrow$ Koks

Tabelle 4.15: Zusammensetzung und Stoffeigenschaften der Biomassepartikel

	Verteilung ( $t = 0$ )	Molare Masse $\frac{g}{mol}$	Wärme- kapazität $c_p = \left[ \frac{kJ}{kg \cdot K} \right]$	Wärme- leitfähigkeit $\lambda = \left[ \frac{W}{m \cdot K} \right]$
	%			
Cellulose	42	162	2.3 <sup>[142]</sup>	0.2426 <sup>[142]</sup>
Hemicellulose	29	164.8	2.3 <sup>[142]</sup>	0.2426 <sup>[142]</sup>
Lignin	16	192.3	2.3 <sup>[142]</sup>	0.2426 <sup>[142]</sup>
Pyrolysegas (Teer+Gas)	0	52.7	1.1 <sup>[127]</sup> -2.5 <sup>[142]</sup>	0.02577 <sup>[127, 143]</sup>
Wasser	7	18		
Koks	0	$1 \cdot 10^{32}$	1.1 <sup>[142]</sup>	0.1046 <sup>[127]</sup>
Stickstoff	0	28	1.091 <sup>[144]</sup>	0.0563 <sup>[144]</sup>

Tabelle 4.16: Reaktionskennwerte der Biomasse

Reaktionen	Frequenzfaktor $A=[1/s]$	Aktivierungsenergie $EA=[kJ/mol]$	Reaktionsenthalpie $\Delta h_s=[kJ/kg]$	$\alpha$
$k_1$	$4.67 \cdot 10^4$	79,3	-1545 <sup>[1]</sup>	0.25
$k_2$	$4.72 \cdot 10^7$	106	-1545 <sup>[1]</sup>	0.20
$k_3$	$4.47 \cdot 10^7$	139	-1545 <sup>[1]</sup>	0.49
$k_4$	$9.62 \cdot 10^6$	67	-2257	
$k_5$	$1 \cdot 10^6$ <sup>[127]</sup>	108 <sup>[127]</sup>	-42 <sup>[145]</sup>	

Bei der Simulation reaktiver Strömungen werden Stoffdaten, wie die Wärmekapazität häufig über das sog. JANAF-Modell, d.h. Polynome 5. Ordnung, generiert (Kapitel 4.3). Die erforderlichen Koeffizienten für Wasser und Stickstoff liegen in der OpenFOAM® Burcat-Datenbank vor. Im Bestreben größere Verfälschungen zu vermeiden und den Einfluss des Koks aus der Strömungssimulation heraus zu nehmen, wurde der Wert für Koks-Wärmekapazität auf nahe Null gesetzt und der Wert für Pyrolysegas konstant auf den Wert für Stickstoff bei Raumtemperatur.

Tabelle 4.17: Koeffizienten zur Berechnung der Wärmekapazitäten

	Wasser	N2	Pyrolyse- gas	Koks
$a_1$	2.67704	3	6.2417	0.1323
$a_2$	0.002973	0.001488	0	0
$a_3$	$-7.74 \cdot 10^{-7}$	$-5.68 \cdot 10^{-7}$	0	0
$a_4$	$9.44 \cdot 10^{-11}$	$1.01 \cdot 10^{-10}$	0	0
$a_5$	$-4.27 \cdot 10^{-15}$	$-6.75 \cdot 10^{-15}$	0	0
$a_6$	-29885.9	-922.798	-49586.2	-2066.2

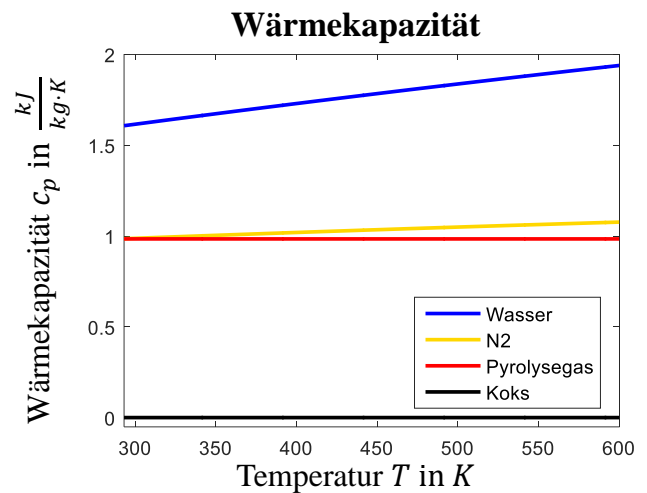


Abbildung 4.15: Verlauf der Wärmekapazitäten

Osama A. Marzouk [146] verglich die Burcat-JANAF-Daten mit weiteren Modellen und Stoffdaten und kam insgesamt zu einer guten Übereinstimmung.

Bei der Verwendung des foamPy-Solvers ist zu beachten, dass die Zeitschritte von Strömungs- und Partikelsimulationen unterschiedlich groß sein können. Jedoch sind die ausgetauschten Daten zwischen zwei Kopplungsintervallen konstant (Abbildung 4.16). Damit bewegt sich weder das Bett bei der Strömungssimulation, noch ändert sich der Kraftterm für die Partikel-Fluid Interaktion auf der Seite der Partikelsimulation.

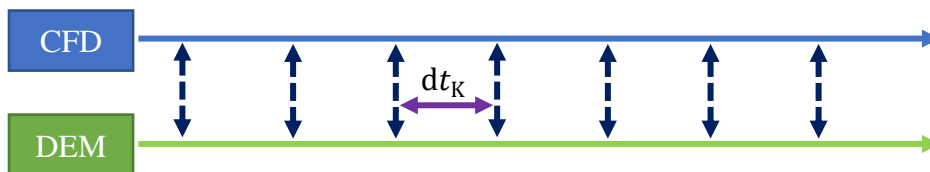


Abbildung 4.16: Datenaustausch zwischen CFD und DEM-Simulation mit Kopplungsintervall  $dt_K$

Bei der zu erwartenden Strömung wird das Bett nicht aufgewirbelt. Somit ist für hinreichend kleine Kopplungszeitschritte  $dt_K$  der zurückgelegte Weg minimal und damit der Kopplungsfehler auf Partikelseite vernachlässigbar. Anders sieht es bei der Strömungssimulation aus. Da die Position der Schnecken kontinuierlich aktualisiert wird, kommt es dabei zum Überschreiben der Quellterme. Dies hat unmittelbaren Einfluss auf die Massenerhaltung, Impulserhaltung, Energieerhaltung und die Transportgleichung. Dies ist nicht zu vermeiden, kann jedoch mit kleiner werdendem Kopplungszeitschritt gelindert werden.

Dies steht im Konflikt zur Rechenzeit - mit jeder Kopplung muss die Lokalisierung und Diskreditierung neu durchgeführt werden – und erfordert ein geeigneter Kompromiss. Betrachten wir eine Schneckenumdrehung (d.h. 0.5 s) und stellen die Anforderung, dass deutlich weniger als eine Zelle pro Zeitschritt (Gitterbreite 1 mm) durchschritten werden darf, so ergibt sich für  $dt_K \approx 0.001$  s.

Unter den genannten Voraussetzungen betrachtet Abbildung 4.17 diese Problematik des Massenverlusts. Dabei werden die in der Partikelsimulation ausgehenden Masseströme in Relation zur in der Strömungssimulation ankommenden Masseströme gesetzt. Diese Aussage kann auf die Energieströme ausgeweitet werden.

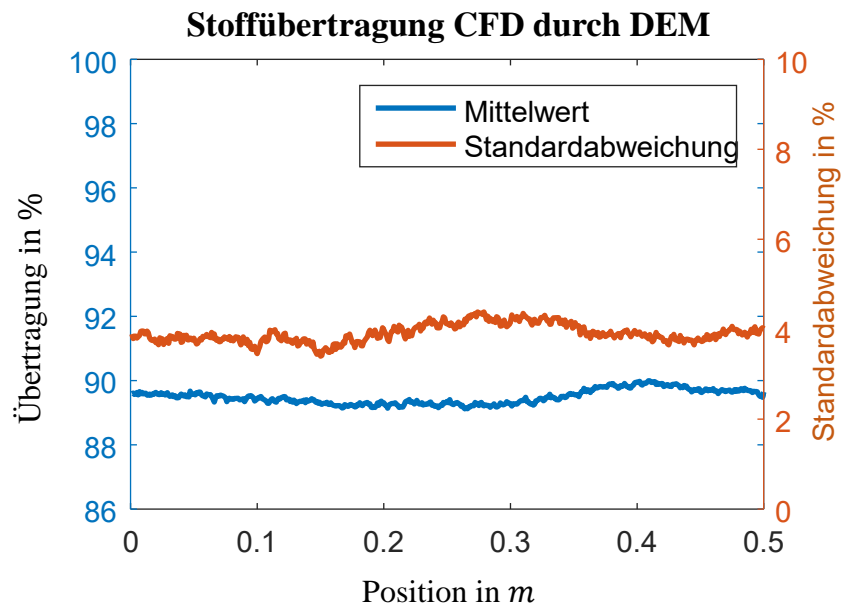


Abbildung 4.17: Datenübertragung der Reaktionskomponenten.  
 CFD ( $\sum_k S_{Y_k,i}$ , Gleichung 4.47) durch DEM ( $\sum_j \frac{\partial m_{i,j,n+1}}{\partial t}$ , Gleichung 4.34)

Eine durchschnittliche und vergleichsweise konstante Abweichung von 10.5 % ist über die Dauer einer Schneckenumdrehung zu beobachten.

Die Berechnung der chemischen Reaktionen während jedem Zeitschrittes, führt zu einer kontinuierlichen Reduzierung des Partikelradius. Es ist zu befürchten, dass dies einen negativen Einfluss auf das Partikelverhalten und die Rechenzeit ausübt. Daher wurde die Reaktionskinetik nur zum Zeitpunkt der Kopplung für das gesamte Kopplungsintervall berechnet. Für diesen Zeitraum ( $dt = 0.001 \text{ s}$ ) wird die Temperatur als konstant erachtet.

Eine Simulation wurde unter diesen Voraussetzungen und den in Tabelle 4.18 enthaltenen Rand- und Startbedingungen für die CFD-Simulation durchgeführt.

Tabelle 4.18: Randbedingungen bei der Strömungssimulation im Doppelschneckenmischreaktor

		$U$ [m/s]	$p$ [kg/(m · s <sup>2</sup> )]	$T$ [K]	$N_2$ [%]	$X$ [%]	$\varepsilon$ [%]
Inlet	Biomasse	$\dot{V} = 0.89 \text{ m}^3/\text{h}$	$\nabla p = 0$	$T = 293.15$	$N_2 = 100$	$X = 0$	$\varepsilon_f = 100$
	Wärmeträger	$\dot{V} = 0.036 \text{ m}^3/\text{h}$	$\nabla p = 0$	$T = 773.15$	$N_2 = 100$	$X = 0$	$\varepsilon_f = 100$
	Koks	$\dot{V} = 0.20 \text{ m}^3/\text{h}$	$\nabla p = 0$	$T = 773.15$	$N_2 = 100$	$X = 0$	$\varepsilon_f = 100$
Outlet	OUT	$\nabla U = 0$	$p = 0$	$\nabla T = 0$	$\nabla N_2 = 0$	$\nabla X = 0$	$\varepsilon_f = 100$
Wand	BMFörderbereich	$U = 0$	$\nabla p = 0$	$T = 293.15$	$\nabla N_2 = 0$	$\nabla X = 0$	$\varepsilon_f = 100$
	Reaktionsbereich	$U = 0$	$\nabla p = 0$	$T = 773.15$	$\nabla N_2 = 0$	$\nabla X = 0$	$\varepsilon_f = 100$
Startbe- dingung	BMFörderbereich	$U = 0$	$p = 0$	$T = 293.15$	$\nabla N_2 = 0$	$\nabla X = 0$	$\varepsilon_f = 100$
	Reaktionsbereich	$U = 0$	$p = 0$	$T = 773.15$	$\nabla N_2 = 0$	$\nabla X = 0$	$\varepsilon_f = 100$

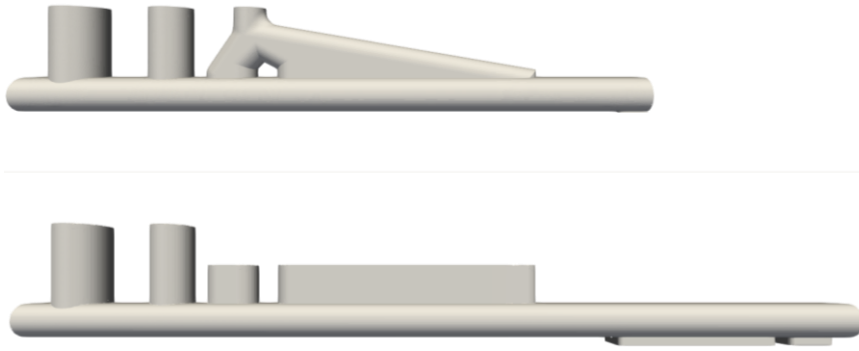


Abbildung 4.18: Verwendete Geometrien bei der CFDEM®-Simulation. Oben DEM, unten CFD.

Für die Partikelsimulation und die Strömungssimulation wurden leicht unterschiedliche Reaktor-geometrien verwendet. Beide Variationen dienen der Beschleunigung der Simulationen.

So wird bei der DEM-Simulation, wie auch schon in Kapitel 3, mit einer verkürzten Version gearbeitet. Dies erlaubt eine bessere Kugelverteilung auf die einzelnen Prozessoren.

Die Veränderungen am Gas-Auslass (Abbildung 4.18), bei der CFD-Simulation, erweitern eine Engstelle und senken so die Strömungsgeschwindigkeit. Da diese antiproportional zur Schrittweite ist, hat dies direkten Einfluss auf die Rechenzeit. Beide Modifikationen sind im weitesten Sinne unkritisch. In den hinteren Teil des Reaktors gelangen nur vereinzelt Wärmeträger und Biomassepartikel. Die absolute Majorität der Simulierten Partikel wird im Anfangsbereich des Wärmeträgerauslasses abgeschieden. Der Pyrolysegasauslass hat zwar einen Einfluss auf die Gas-Verweilzeit aber nicht auf die Reaktionsverteilung im Reaktor, sodass von keinem idealen Aufbau gesprochen werden kann. Dieser ergibt sich aus der unbekanntem Reaktionsverteilung.

Ein großes Problem bei der CFDEM® ist die Parallelisierung. Die Simulationsräume bei der DEM und bei der CFD Simulation sind unabhängig voneinander. Eine echte Synchronisation findet nicht statt. Stattdessen werden alle Partikel auf alle Prozessoren während der CFD-Simulation übertragen. Dies führt bei steigender Parallelisierung zu einem erheblichen Speicherbedarf, der bei Verteilungsmethoden wie "divided", bei der das Partikelvolumen auf bis zu 29 Subzellen verteilt wird, sich verschärft. Bei einer Zellgröße der Strömungssimulation von  $1\text{ mm}$ , und  $1.4 \cdot 10^6$  Partikeln, benötigt die Simulation, auf 80 Prozessoren und mit dem RAM schonenden "centre" Verteilungsalgorithmus, rund 820 GB RAM. Dieser Speicherbedarf kann auf den Clustern des KIT, d.h. *bwUniCluster I+II* und den *ForHLR I+II* nur von den "fetten"-Knoten erfüllt werden. Der *bwUniCluster II* benötigt zwischen 72 und 88 Stunden um eine Sekunde zu simulieren. Da die Maximalzeit für eine Rechnung auf dem "fetten"-Knoten auf 3 Tage, d.h. 72 Stunden beschränkt ist, wurde mit einem Auftrag lediglich 0.5 s simuliert (eine Schneckenumdrehung bei  $2\text{ Hz}$ ). Die finale Simulationszeit wird durch wiederholten Neustart der Simulation mit dem zuletzt erreichten Zustand erzielt.

#### 4.5.4 Stoff- und Energietransport im Doppelschneckenmischreaktor

Eine Simulation unter den im vorhergehenden Kapitel genannten Bedingungen wurde durchgeführt. Dabei wurden insgesamt 12.5 s simuliert. Bei 6 s wurden letzte kleine Änderungen vorgenommen. Damit wurde bei einem freien Volumen von rund 4 l und einem Produktstrom am Auslass von 2.1 l/s (Daten aus der Simulation entnommen) zwischen sechs und drei Gasverweilzeiten simuliert, um ausreichende und aussagekräftige Resultate erzielen zu können.

Bei der klassischen DEM-Simulation wird der Wärmeübergang zwischen zwei Partikeln durch die Wärmeleitung an der Kontaktfläche der Partikel dargestellt. In Kapitel 2.7 wurde gezeigt, dass dieser Wärmeübergang als zu konservativ betrachtet werden muss. Weiterhin wurde eine Methode beschrieben, wie der Wärmeübergang durch die Gasphase approximiert werden kann. Der mit dieser Methode berechnete Wärmeübergangskoeffizient (Kapitel 3.2) betrachtet nur den Wärmeübergang an Kontaktstellen und ist daher etwas niedriger als der von Funke et. al. [89] ermittelte Wert. Er zeigt jedoch insgesamt eine gute Übereinstimmung.

CFDDEM-Simulationen bieten, durch die Kopplung mit der Strömungssimulation, die Möglichkeit zusätzlich den konvektiven Wärmeübergang durch die Gasphase zu simulieren. Das dabei zum Einsatz kommende Modell wurde von Li. et. al. [123] für den pneumatischen Transport von Polymerpellets entwickelt. Dabei spielte der Partikelkontakt eine untergeordnete Rolle.

Im direkten Vergleich der Temperaturen zeigt sich, dass bei der DEM-Simulation mit modifizierten Bachelor-Modell die Temperatur schneller ansteigt und eine höhere Mischtemperatur erreicht wird (Abbildung 4.19). Ursachen können die Verdampfungsenthalpie von Wasser und die endotherme Reaktion von Cellulose, Hemicellulose und Lignin sein. Aus diesem Grund wurde darüber hinaus, der Wärmeübergangskoeffizient  $\alpha_{WT-BM_i}$  gebildet, welcher analog zu Abbildung 3.3 (Kapitel 3.1, Gleichung 3.2) für jedes Segment  $i$  eines axial diskretisierten Doppelschneckenmischreaktors bestimmt wurde.

Ein direkter Vergleich beider Wärmeübergangskoeffizienten (Abbildung 4.19) zeigt, dass der CFDDEM-Koeffizient mit dem Bachelor-Modell wie in der Literatur beschrieben gegenüber dem DEM-Koeffizient mit modifiziertem Bachelor-Modell um den Faktor 4 kleiner ist. Ab einer Reaktorposition von 0.6 m sind die Abweichungen auf den Wärmeübergang von Reaktorwand auf die Biomasse zurück zu führen. Da das  $\Delta T$ , d.h. der Temperaturunterschied zwischen Biomasse und Wärmeträger in diesem Bereich nahe Null ist, kommt es an dieser Stelle zu einer erheblichen Abweichung. Bei größerem  $\Delta T$  geht dieser Effekt unter.

Die Abweichungen im Wärmeübergang können auf die Modelle und deren Anwendungsbereich zurückgeführt werden. Li entwickelte sein Modell für den Wärmeübergang in einer pneumatischen Förderröhre, bei der der Wärmeübergang durch Wärmeleitung an Kontaktstellen eine untergeordnete Rolle spielt. Bachelor entwickelte sein Modell für ein dichtes Bett von Stahlkugel mit niedriger Biot-Zahl und freier Konvektion, sodass der Wärmeübergang durch die Kontaktstelle den dominierenden Einfluss bildet. Bei einer Kombination beider Methoden wird daher der Wärmeübergang z.B. an laminaren Grenzschichten um den Kontaktpunkt vernachlässigt. Dieser kann bei hoher Biot-Zahl, wie es beim Weizenstroh der Fall ist, signifikant ausfallen. Der modifizierte Bachelor-Wärmeübergang approximiert diesen Sachverhalt, weshalb zu vermuten ist, dass dieser den genaueren Wert wiedergibt.

## Wärmeübergang im Doppelschneckenmischreaktor CFDDEM vs. DEM

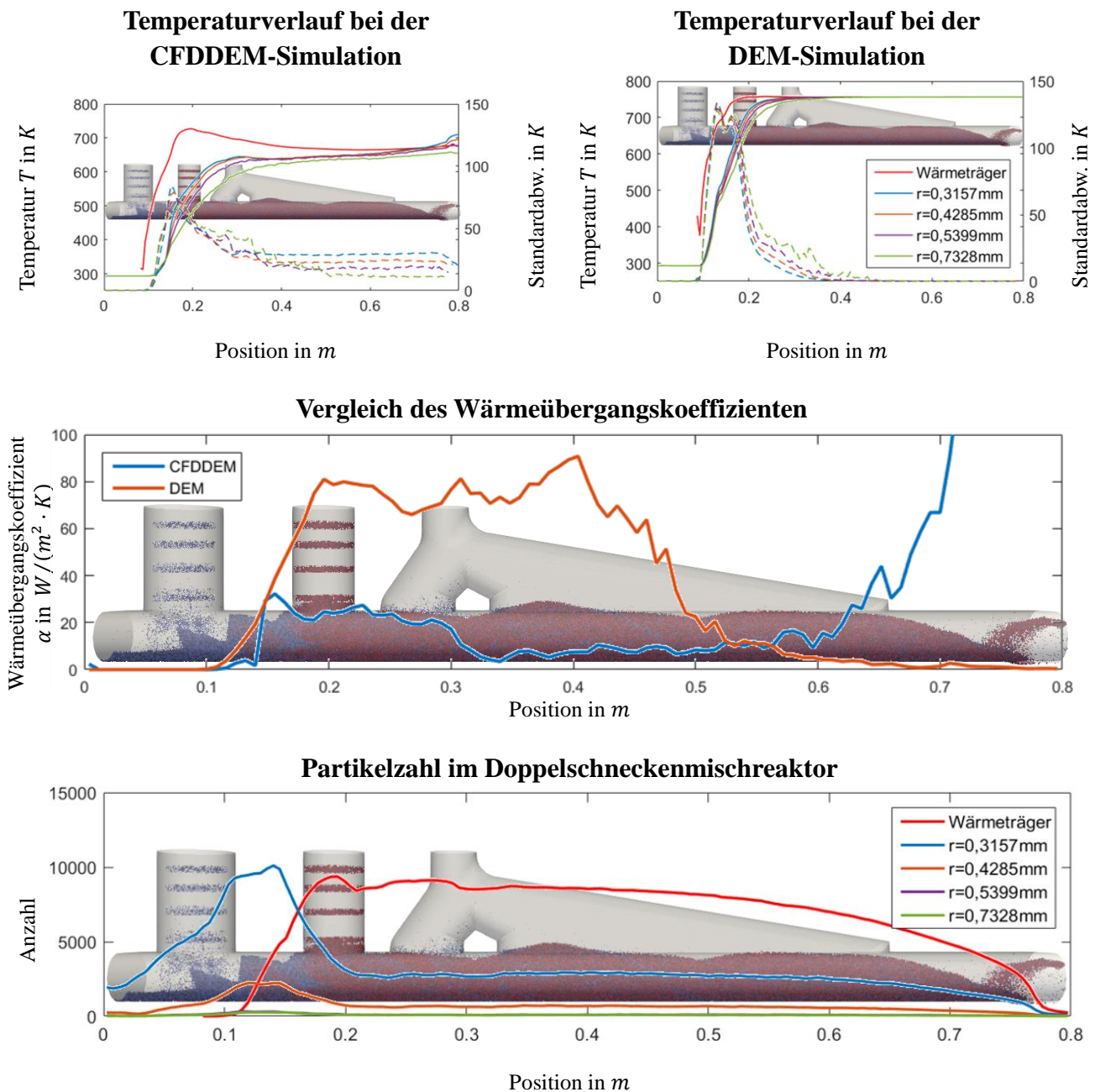


Abbildung 4.19: Wärmeübergang im Doppelschneckenmischreaktor. Gestrichelte Verläufe stellen die Standardabweichung des farblich korrespondierenden Verlaufs dar.

Unabhängig des Modells ist die Mischtemperatur zwischen Wärmeträger und Biomasse bei 0.6 m weitestgehend erreicht, sodass der Wand→Biomasse-Wärmeübergang dominiert und damit den Koeffizienten verfälscht. Durch die schwindende Partikelzahl am Reaktorende, wird dieser Effekt verstärkt. Der Wand→Biomasse-Wärmeübergang ist auch die Ursache für die Standardabweichung der Temperatur ab Reaktormitte.

Betrachten wir den Energiefluss im Doppelschneckenmischreaktor (Abbildung 4.20), so ist erkennbar, dass der Wärmeträger und die Reaktorwand, Wärmeleistung an das Gas abgeben, während die kalte Biomasse diese aufnimmt.

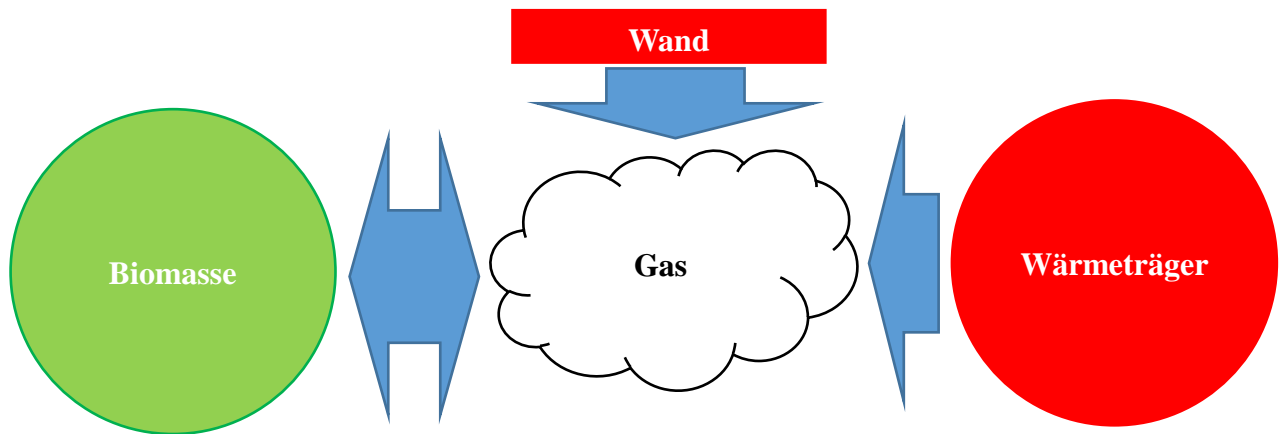


Abbildung 4.20: Allgemeiner Wärmeleistungsstrom durch die Gasphase bei der CDFDEM-Simulation der Schnellpyrolyse von Biomasse im Doppelschneckenmischreaktor

Durch die Pyrolyse der Biomasse wird das Gas um die Pyrolyseprodukte angereichert. Dabei geht die im Pyrolyseprodukt enthaltene Energie abzüglich der Enthalpie in die Gasphase über. Trivialerweise muss die Gastemperatur größer der Biomassetemperatur sein. Daher kann die Temperatur Gasphase als Indikator verwendet werden, um größere Fehler, Instabilitäten und Bilanzfehler zu erkennen.

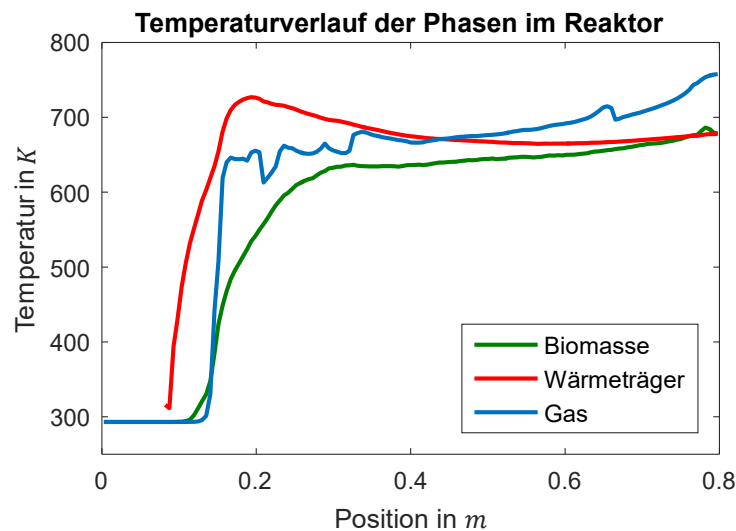


Abbildung 4.21: Temperaturverlauf von Biomasse, Wärmeträger und dem Hintergrundgas

Betrachten wir dessen Verlauf (Abbildung 4.21), so ist erkennbar, dass die Forderungen erfüllt sind. Auf ganzer Länge ist die Temperatur höher, als die Temperatur der Biomasse. Dennoch ist die beobachtete Mischtemperatur auffällig niedrig. Die Mischtemperatur  $T_{\text{Mix}}$  kann einfach abgeschätzt werden. Berücksichtigt man die Enthalpie  $\Delta h_{s_i}$  der Komponente  $i$  und deren Anteil in der Biomasse  $c_{BM_i}$  kann ein konservativer Wert für die Temperatur  $T_{\text{Reak.}}$  abgeschätzt werden (Gleichung 4.49).

$$T_{\text{Mix}} = \frac{c_{p_{\text{BM}}} \cdot \dot{m}_{\text{BM}} \cdot T_{0_{\text{BM}}} + c_{p_{\text{WT}}} \cdot \dot{m}_{\text{WT}} \cdot T_{0_{\text{WT}}}}{c_{p_{\text{BM}}} \cdot \dot{m}_{\text{BM}} + c_{p_{\text{WT}}} \cdot \dot{m}_{\text{WT}}} \approx 756 \text{ K}$$

$$T_{\text{Reak.}} = T_{\text{Mix}} + \frac{\dot{m}_{\text{BM}} \cdot \sum_i c_{BM_i} \cdot \Delta h_{s_i}}{c_{p_{\text{BM}}} \cdot \dot{m}_{\text{BM}} + c_{p_{\text{WT}}} \cdot \dot{m}_{\text{WT}}} \approx 732 \text{ K} \quad (4.49)$$

Die Biomassetemperatur  $T_{BM}$  sollte im stationären Zustand im Bereich  $T_{\text{Reak.}} < T_{BM} < T_{\text{Mix}}$  liegen. D.h. die stationäre Biomassetemperatur bei der CFDDEM-Simulation ist  $50 - 70 \text{ K}$  zu niedrig. Dies kann teilweise auf die Schneckenrandproblematik zurückgeführt werden. Jedoch ist dieser Effekt aufgrund des kleinen Temperaturgradienten zwischen Wärmeträger und Mischtemperatur zu schwach. Problematischer ist hierbei die Rückvermischung in dem Biomasseförderbereich. Abbildung 4.19 ist zu entnehmen, dass sich eine größere Zahl Wärmeträger in diesem Bereich befindet. Dadurch, dass die Wand-Randbedingung der Temperatur in diesem (Förder-) Bereich fest auf  $293.15 \text{ K}$  gestellt ist, bildet sich hier eine Wärmesenke, vom Partikelbett zur Reaktorwand, die dem Wärmeträger verstärkt thermische Energie entzieht. Der Wärmeübergang an der Reaktorwand wurde aufgrund technischer Schwierigkeiten bei der reinen DEM-Simulation nicht simuliert. Dies stellt somit eine wichtige Erkenntnis dar. Jedoch muss berücksichtigt werden, dass die Temperaturverteilung bei der PYTHON variiert. Wie beschrieben, wird der Reaktor im Reaktionsbereich begleitbeheizt und ist im Transportbereich nicht isoliert. Jedoch ist die Temperaturverteilung weniger sprunghaft wie in der CFDDEM-Simulation. D.h. dieser Effekt ist auch in der PYTHON vorhanden, aber nicht so stark ausgeprägt. Eine Temperaturerfassung der Reaktorinnenwand könnte auf die Randbedingung übertragen werden und somit genutzt werden, um diesen Effekt genauer zu beschreiben. Jedoch ist diese Temperaturerfassung mit großem Aufwand verbunden.

### Konzentrationsverlauf im Reaktor

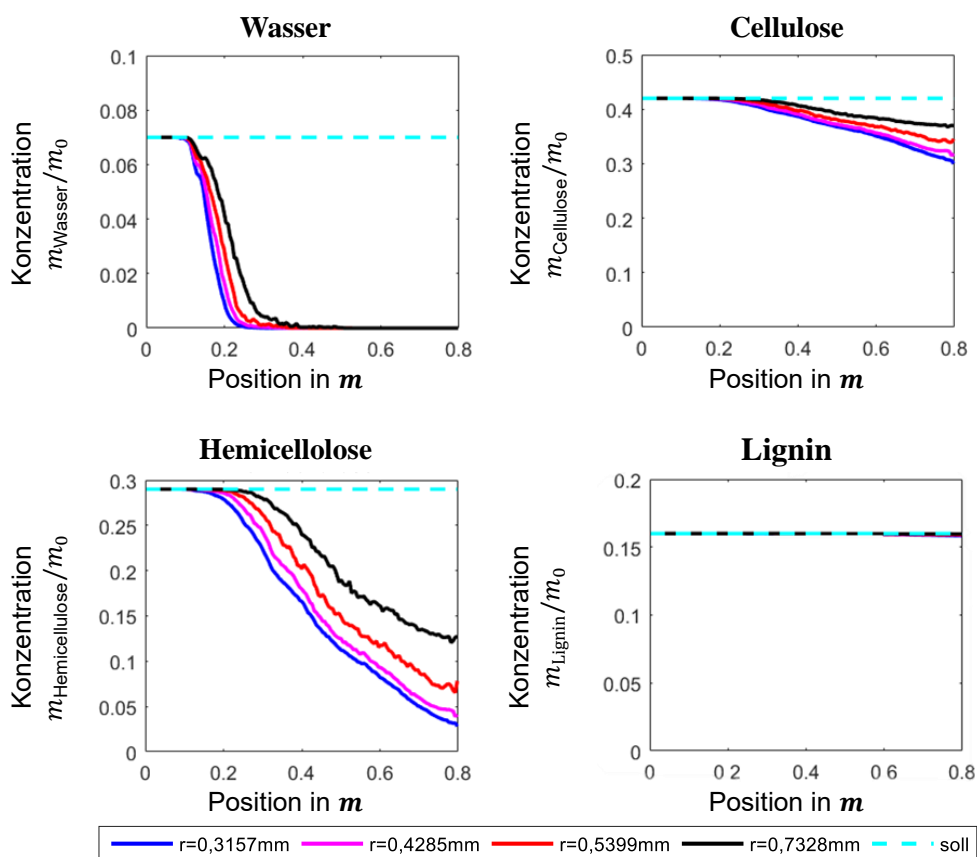


Abbildung 4.22: Größenabhängiger Konzentrationsverlauf der Biomasse im Reaktor

Die niedrige Temperatur hat natürlich erheblichen Einfluss auf den Reaktionsverlauf (Abbildung 4.22). Dieser ist nicht abgeschlossen. Berücksichtigen wir die Partikelform (Kapitel 2.3), so kommen bei einer Biomassepartikeldicke von  $0.15 \text{ mm}$ , Sphären mit dem kleinsten Radius am nächsten. Der Unterschied in der Form führt zu einer kleineren, effektiven Oberfläche und damit zu



einem schlechteren Wärmeübergang. D.h. Biomassepartikel mit einem Radius von  $0.3157\text{ mm}$  spiegeln das reale Verhalten am genauesten wieder. Dennoch ist die Reaktionsrate für Cellulose auffällig niedrig.

Ursachen könnten Programmierfehler sein oder die Übertragbarkeit der Reaktionsparameter von Experiment zum Reaktor.

Um Programmierfehler im Zusammenhang mit der Reaktionskinetik auszuschließen wurde der Doppelschneckenmischreaktor als Plug-Flow-Reaktor neuinterpretiert (Abbildung 4.23). Die verwendete Reaktionskinetik ist identisch zur gekoppelten Strömungs-/Partikelsimulation. Der Temperaturverlauf im Plug-Flow-Reaktor wird vorgegeben und entspricht dem volumengewichteten mittleren Temperaturverlauf der gekoppelten Simulation bei einer Verweilzeit von  $10.6\text{ s}$  (Frey [66]). Dabei wird ersichtlich, dass der Reaktionsverlauf, welcher nur vom vorgegebenen Temperaturverlauf und der Verweilzeit abhängig ist (Abbildung 4.23), weitestgehend mit der gekoppelten Strömungs-/Partikelsimulation (Abbildung 4.23) übereinstimmt. Zu Abweichungen kommt es aufgrund der Reaktionen und der damit verbundenen Volumenänderung der Biomassepartikel. Frey bestimmte die Verweilzeit im Reaktor anhand eines ColdFlow-Reaktors. Durch die Reduzierung des Gesamtvolumens steigt diese im Zweifel.

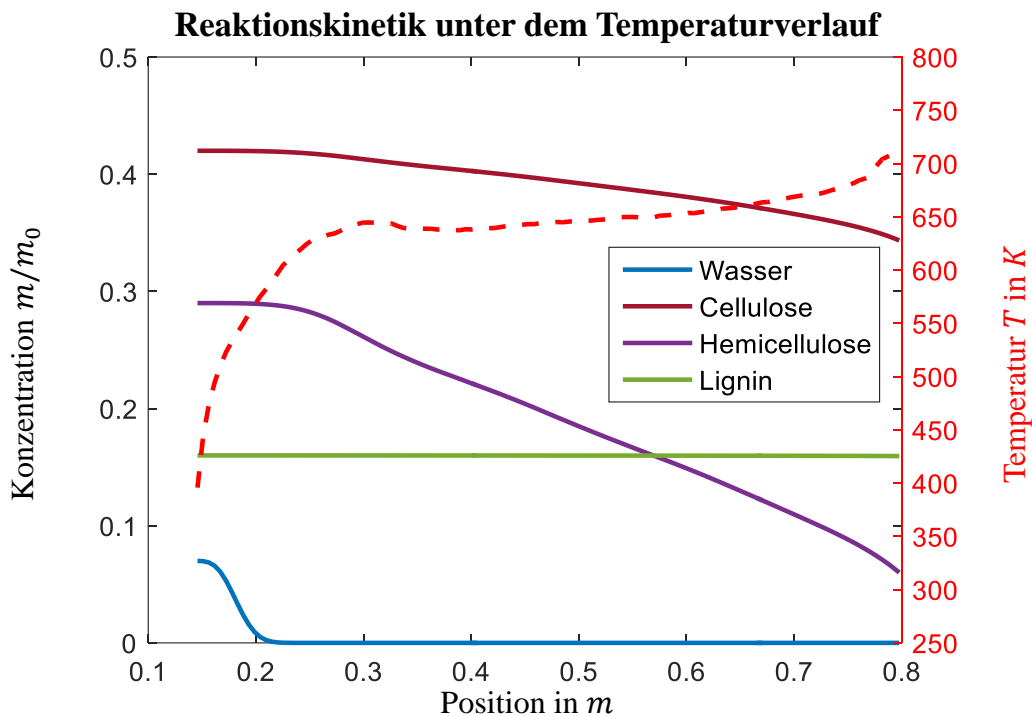


Abbildung 4.23: Konzentrationsverlauf unter dem Reaktortemperaturverlauf und einer Verweilzeit im Misch- und Reaktionsbereich von  $10.2\text{ s}$

Anca-Couce et. al. [147] setzten sich mit den in der Literatur zu findenden TG-Analysen von Lignocellulose bei der Pyrolyse und deren Kinetiken auseinander. Die im Rahmen dieser Dissertation ermittelten Werte für das kinetische Reaktionsmodell von Cellulose, bewegen sich am Rand des von Anca-Couce et. al. ermittelten Wertebereichs. Jedoch beinhalten die selbstermittelten Werte den katalytischen Effekt der Alkalimetalle in der Weizenstrohasche, sodass die Abweichung auch darauf zurückgeführt werden kann.

Abbildung 4.24 zeigt, dass Cellulose mit den ermittelten Werten, im Reaktor und verschiedenen Temperaturen, wie sie im Reaktor zu finden sind, träge reagiert.

### Temperaturabhängigkeit der Biomassereaktionen

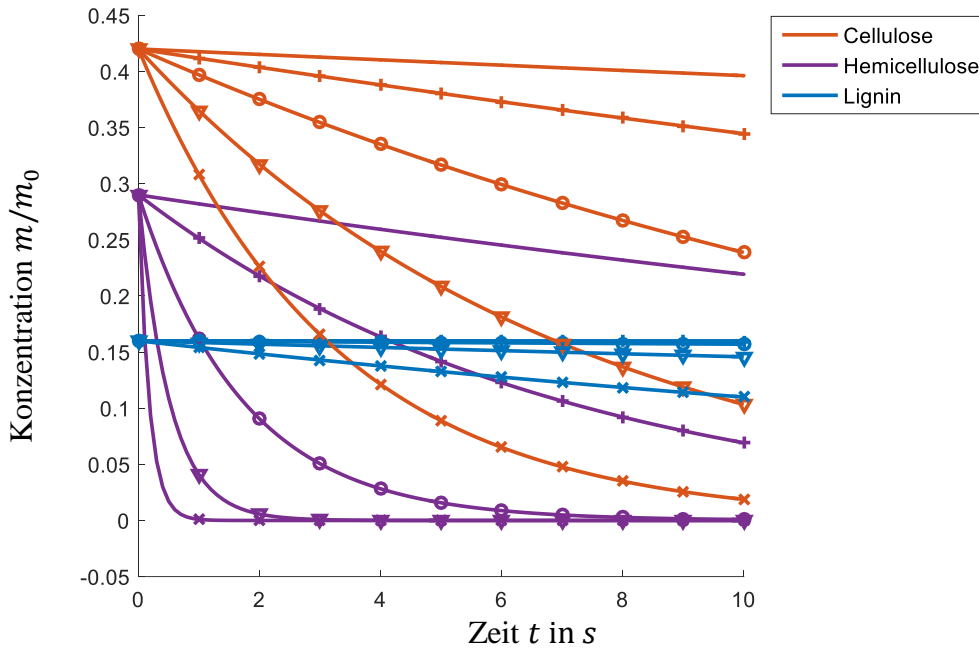


Abbildung 4.24: Reaktionsverlauf bei konstanten – 600 K, + 650 K, 0 700 K, ▽ 750 K und X 800 K

Die Cellulose-Reaktion ist im angegebenen Zeitraum selbst für höhere Temperaturen nicht abgeschlossen. Daher muss davon ausgegangen werden, dass die Parameter der Reaktionskinetik von Cellulose diese Reaktion nicht genau genug beschreiben.

### Pyrolyseproduktbildung im Doppelschneckenmischreaktor

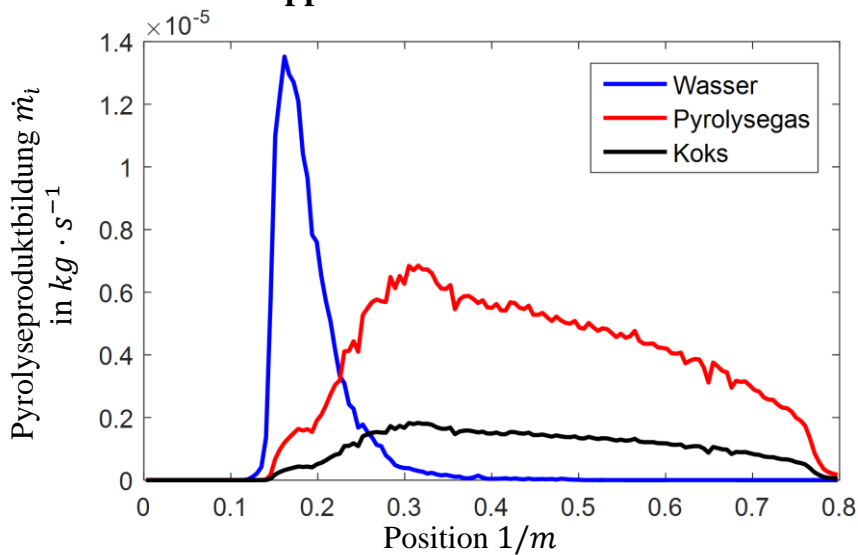


Abbildung 4.25: Produktentstehung im Doppelschneckenmischreaktor

Unabhängig der Kritiken an Mischtemperatur und an der Cellulose-Reaktion, können der Simulation wichtige Erkenntnisse entnommen werden. Betrachten wir den Quellterm (Abbildung 4.25), mit dem die einzelnen Produkte in die Differentialgleichungen eingepflegt werden und die Produktverteilung (Abbildung 4.26) lokal aufgelöst wird, so ist zu beobachten, dass Wasser an einer anderen Stelle im

Reaktor freigesetzt wird als Pyrolysegas und Koks. Dieses verlässt den Reaktor überwiegend an der ersten Öffnung des Gasturms (Abbildung 4.26).

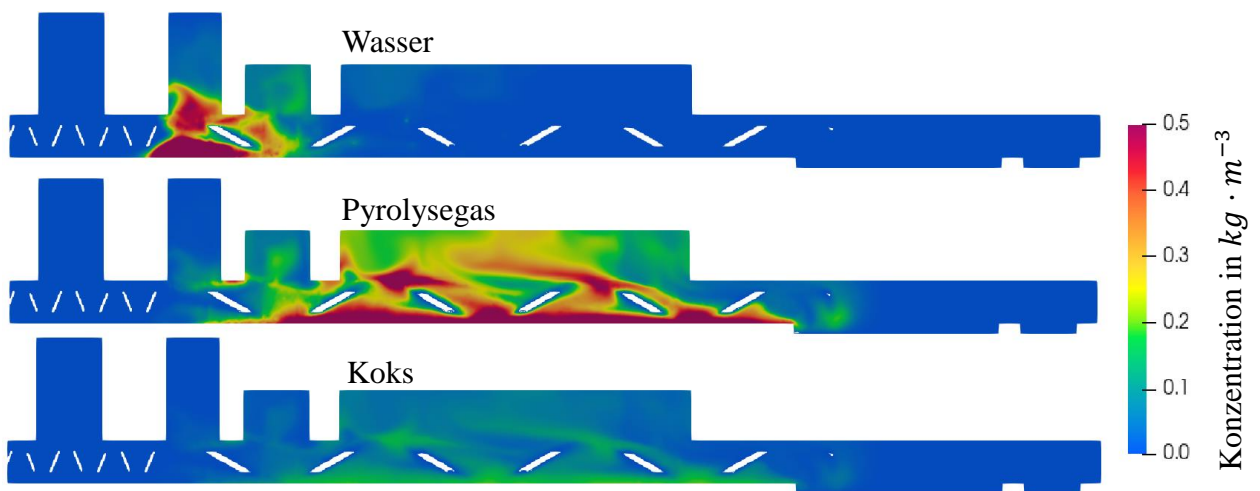


Abbildung 4.26: Konzentrationsverlauf im Reaktor. Schnitt durch Reaktormitte.

D.h. die Zusammensetzung des Produktes variiert abhängig von der Position im Reaktor. Dadurch ließe sich z.B. durch Einbauten am Gasturm Einfluss auf die Produktzusammensetzung nehmen.

## 4.6 Diskussion

Bei der Auswertung der gekoppelten Strömungs-/Partikelsimulation zeigt sich der erwartete Massen- und Energieverlust beim Übertrag von DEM nach CFD. Dabei handelt es sich um ein Systematisches Problem welches im Zusammenhang mit dem Brinkmann-Strafverfahren, zur Darstellung der Schneckenbewegungen, steht. Unter den aktuell verfügbaren und getesteten Methoden zur Darstellung von bewegten Objekten in der Strömungssimulation erwies sich das Brinkmann-Strafverfahren als das einzige welches sowohl dem geringen Abstand der Schnecken zur Reaktorwand, als auch die ineinander kämmenden Schnecken, stabil darstellen konnte.

Darüber hinaus konnte gezeigt werden, dass der Wärmeübergang durch die fluide Grenzschicht an der Partikel-Kontaktstelle eine tragende Rolle spielt. Jedoch wird dieser Effekt durch die aktuellen Modelle bei der CFDDEM-Simulation nicht wieder gegeben.

Unabhängig dessen, konnte ein räumlicher Versatz zwischen dem Wasser-Verdampfungsbereich und dem Reaktionsbereich, indem die Pyrolyse vonstattengeht festgestellt werden. D.h. durch Einbauten und einem weiteren Produktauslass kann Einfluss auf den Wassergehalt der Pyrolyseprodukte genommen werden.

Die zuvor im Zuge der DEM-Untersuchung des Doppelschneckenmischreaktors festgestellte Rückvermischung verursachte einen signifikanten Energieverlust. Dies ist darauf zurück zu führen, dass die Rückvermischung in den nicht-begleitbeheizten Biomassetransportbereich zurückreicht. Dieser hat in der CFDDEM-Simulation Raumtemperatur. Im Betrieb der PYTHON ist dies aufgrund der Wärmeleitung nicht der Fall. Jedoch ist auch hier diese Schwachstelle gegeben, wenn auch milder ausgeprägt.

Weiterhin konnte beobachtet werden dass die Pyrolyse am Wärmeträgerauslass nicht abgeschlossen ist. Eine Beobachtung, welche durch eine Sensitivitätsanalyse der Reaktionskinetik in Abhängigkeit der Temperatur bestätigt wird.

Die Reaktionskinetik wurde anhand von TG-Messdaten und einer Parameterschätzung bestimmt. Die Parameter liegen in dem von Anca-Couce et. al. [147] ermittelten Wertebereich für Reaktionskinetiken der Pyrolyse von Biomasse, welche anhand von TGA-Messungen bestimmt wurden. Darüber hinaus wurden diese Werte in weiteren Versuchen bestätigt. Unter der Annahme dass die Reaktionskinetik korrekt den Sachverhalt wiedergibt, ist es erforderlich die Verweilzeit im Doppelschneckenmischreaktor zu erhöhen. Alternativ ist es erforderlich die Reaktionskinetik zu überarbeiten. Ansatzpunkte könnten eine detailliertere Betrachtung des Lignins sein oder die Berücksichtigung einer Koks-Umformung, bei der flüchtigen organischen Verbindungen bei höheren Temperaturen als Nebenprodukt abgespalten werden. Darüber hinaus kann es von Interesse sein die Kinetik anhand einer anderen Methode zu ermitteln, da der maximale Temperaturgradient bei TGA-Messungen nicht an dem der Schnellpyrolyse heranreicht.

Unabhängig des Reaktionsnetzwerks und der Parameter liefert der Solver bereits jetzt wertvolle Erkenntnisse. Jedoch sind einige Modifikationen hinsichtlich des Bewegungsmodell und der Reaktionskinetik nötig.

## 5 Abschließender Rück- und Ausblick

Diese Arbeit betrachtet die Transportvorgänge bei der Pyrolyse von Biomasse in der PYTHON – einer Technikumsanlage mit einer maximalen Kapazität von  $20 \text{ kg h}^{-1}$  Biomasse. Erste Untersuchungen zu dieser Thematik wurden von Kornmayer durchgeführt. Im Zuge seiner Dissertation erfolgte eine Charakterisierung der Pyrolyse im Doppelschneckenmischreaktor anhand eines Blackbox-Verfahrens, d.h. der Betrachtung der ein- und ausgehenden Massen- und Energieströme, ohne nähere Betrachtung der inneren Vorgänge. Daran anknüpfend betrachtet diese Arbeit die Stoff- und Energietransportvorgänge im Doppelschneckenmischreaktor.

Aufgrund der Herausforderungen bei einer experimentellen Charakterisierung, wie die geringe Reaktionszeit, die Reaktivität der Produkte und die Komplexität der Geometrie und deren Einfluss auf das Mischverhalten, wurde ein numerischer Ansatz gewählt. Dabei kommt insbesondere ein Euler-Lagrange-Ansatz, d.h. die Diskrete Elemente Methode (DEM) gekoppelt mit einer Strömungssimulation (CFD), zum Tragen, der sowohl das Feststoffverhalten, als auch die Fluide-Phase gut beschreibt.

Eine CFD-Simulation erfordert Kenntnisse über die chemischen Reaktionen und Wissen über die thermophysikalischen Eigenschaften der beteiligten Produkte. Eine DEM-Simulation hingegen, erfordert die Kenntnis über die Materialeigenschaften der beteiligten Produkte. Beide Themen werden im Zuge dieser Arbeit ausführlich behandelt.

### DEM – Parameterbestimmung und –optimierung

Zunächst wurde eine Studie bzgl. der Form von Weizenstrohpartikeln und der Partikeldichte, d.h. der Skelettdichte inklusive Porenvolumen, durchgeführt. Unter der Verwendung einer Bildauswertung konnten Partikel nach Länge und Breite klassiert und in ein 3-Dimensionales Histogramm aufgetragen werden. Daraus wurden vier Partikelformen abgeleitet.

Werte für die Wärmeträger-Wärmeträger Reibung und Rollreibung wurden anhand einer Studie ermittelt. Hierbei wurde ein realer Versuch, d.h. eine rotierende Trommel, unter veränderten Reib- und Rollreibungswerten in Simulationen, mit dem Ziel der größten Übereinstimmung, nachgebildet.

Die Rechenzeit machte einige Annahmen bzgl. der Weizenstrohpartikelform erforderlich. Diese mussten im weiteren Verlauf durch Kugeln approximiert werden. Maione et. al. [69] zeigte, dass diese Annahme für eine Sphärizität von 0.73 und einer Werkzeug-Froude-Zahl nahe eins zutreffend ist. Diese Aussage konnte für Weizenstrohpartikel mit einer Sphärizität von 0.22 erweitert werden. Dabei wurden zwei rotierende Trommel Simulationen mit Weizenstrohpartikeln und Wärmeträger durchgeführt. Eine der Simulationen enthielt mittels Multisphärenansatz aufgelöste Weizenstrohpartikel, die andere kugelförmige Weizenstrohpartikel. Es konnte gezeigt werden, dass die Mischgüte und Verteilung weitestgehend übereinstimmen. Jedoch führt die Verwendung von sphärischen Partikeln zu einer Veränderung der Schüttdichte. Mit steigender Leistungskapazität von Hard- und Software könnte zukünftig eine weitere Betrachtung unter Berücksichtigung der Partikelform zu einem präziseren Ergebnis führen.

Eine der wichtigsten Aussagen bzgl. des Weizenstrohs konnte in einem ColdFlow-Modell des Doppelschneckenmischreaktor getroffen werden. Sobald Weizenstroh in das System gegeben wird, verändern sich die Wärmeträger-Wärmeträger-Reibwerte, sodass diese nicht unabhängig vom Weizenstroh ermittelt werden können. Eine Studie betreffend aller Reib- und Rollreibungswerte konnte aufgrund der erforderlichen Rechenzeit und Rechenleistung nicht durchgeführt werden. Jedoch gelang es geeignete Werte abzuschätzen. Dabei wurde die Verweilzeit im ColdFlow-Modellreaktor aus Experimenten mit jenen aus Simulationen verglichen.

Weitere Studien bzgl. des verwendeten Elastizitätsmoduls wurden durchgeführt. Dabei konnte gezeigt werden, dass sich das Mischverhalten im ColdFlow-Reaktor ändert, sodass Änderungen an Parametern und Einstellungen vorzugsweise nicht vorgenommen werden sollten, in jedem Fall aber eine Nachsimulation erfordern.

LIGGGHTS® verwendet zur Beschreibung des Wärmeübergangs zwischen zwei Partikeln ein Modell nach Batchelor et al. [79]. Dieses betrachtet den Wärmeübergang als Wärmeleitung durch die Kontaktfläche. Dieses wurde mit einem Modell nach Schlünder et. al. [2] verglichen, das den Wärmeübergang als Wärmeleitung durch die umgebene Gasphase betrachtet. Somit betrachtet Schlünder den Wärmeübergang an der Kontaktstelle und durch das umgebende Fluid des nahen Kontaktbereichs. Im direkten Vergleich ist der Wärmeübergangskoeffizient nach Schlünder höher als der nach Batchelor. Im weiteren Verlauf gelang es, die Berechnung des Wärmeübergangs nach Batchelor so anzupassen, dass er den Wärmeübergang nach Schlünder approximiert. Dies vereint die effiziente Berechnungsmethode nach Batchelor mit der genaueren, aber rechenintensiveren Berechnungsmethode nach Schlünder und ermöglicht wichtige Aussagen zum Wärmeübergang im Doppelschneckenmischreaktor mittels DEM-Simulationen.

### **DEM-Untersuchung des Doppelschneckenmischreaktors**

Eine Untersuchung betreffend der Drehzahl im Doppelschneckenmischreaktor wurde durchgeführt. Dabei wurde ein Werkzeug-Froude-Bereich von  $0.1 < Fr_w < 2.0$  abgedeckt. In diesem Zuge konnte eine Verblockung des Reaktors unterhalb von  $2 \text{ rpm}$  und eine einsetzende Entmischung oberhalb von  $3 \text{ rpm}$  beobachtet werden.

Darüber hinaus konnte, aufgrund der verkürzten Verweilzeit oberhalb von  $3 \text{ rpm}$ , am Feststoffaustrag eine Temperaturdifferenz zwischen Biomasse und Wärmeträger festgestellt werden, sodass der Wärmeübergang von Wärmeträger zu Biomasse nicht vollständig abgeschlossen wurde. Sein Optimum nahm der Wärmeübergang im Bereich von  $2.5 - 3 \text{ rpm}$  an. Ein bei allen Simulationen ohne Verblockung zu beobachtender Effekt war die Rückvermischung des Wärmeträgers in den Biomasetransportbereich. Dabei sinkt die Temperaturdifferenz aus Wärmeträger und Biomasse mit steigender Rückvermischung und führt zu einem gehemmten Wärmeübergang.

Aufgrund der beobachteten Rückvermischung wurde eine weitere Konfiguration getestet, bei der der Wärmeträger räumlich vor der Biomasse in den Reaktor gegeben wird, d.h. eine vertauschte Zugabereihenfolge. Aufgrund der veränderten Volumenströme ist es erforderlich die Schneckenengeometrie anzupassen. Dabei wurde die Ganghöhe des Misch- und Transportbereichs auf die komplette Schneckenlänge erweitert. Im Zuge dieser Studie wurde der gleiche Werkzeug-Froude-Bereich wie zuvor abgedeckt. Dabei konnte bis  $2.5 \text{ rpm}$  eine Verblockung des Reaktors beobachtet werden. Auch bei dieser Konfiguration konnte oberhalb von  $3 \text{ rpm}$  eine Temperaturdifferenz am Feststoffaustrag beobachtet werden, sodass nur ein enger optimaler Betriebsbereich um  $3 \text{ rpm}$  zu identifizieren ist. Des Weiteren wurde eine Rückvermischung der Biomasse in den Wärmeträgertransportbereich beobachtet. Dabei werden wenige Biomassepartikel in einen großen Wärmeträgerüberschuss gemischt. Dieser Effekt wirkt sich weniger nachteilig aus, sodass bis zu doppelt so hohe Wärmeübergangskoeffizienten zu beobachten sind. Jedoch ist selbst bei  $3 \text{ Hz}$  ein "schwimmendes" Biomassebett im Bereich der Zugabestelle zu beobachten, dass sich innerhalb einer Umdrehung bildet und eingemischt wird. Bei der Einmischung sind nachteilige Schlieren zu beobachten.

Eine weitere Studie wurde betreffend des Biomasse/Wärmeträger-Mischverhältnisses durchgeführt. Dabei wurde unter konstantem Gesamtvolumenstrom und bei  $2 \text{ rpm}$ , das Verhältnis aus Biomasse/Wärmeträger-Volumenstrom verändert. Im Rahmen dieser Studie konnte gezeigt werden, dass bei einem Biomassevolumenanteil von unter  $11 \%$  die Rückvermischung einen dominanten Effekt beim Wärmeübergang einnimmt. D.h. ein Großteil des Wärmeüberganges erfolgt im Biomassezugabebereich. Im Zuge dieser Studie konnte ein vergleichbarer Zustand, wie bei Funke et. al. [89] simuliert werden. Dabei stellte sich ein vergleichbarer Wärmeübergangskoeffizient ein, sodass der Einfluss von nicht in Kontakt stehender Partikel auf den Wärmeübergang vernachlässigt werden kann.

Aufgrund eines nicht-reproduzierbaren, sporadischen Programmfehlers in LIGGGHTS® im Zusammenhang mit dem Wand-Partikel-Wärmeübergang lag der Fokus in dieser Studien auf den

Partikel-Partikel-Wärmeübergang. Aufgrund seiner Natur entzog sich dieser Programmfehler einer Ermittlung. Jedoch ist zu vermuten, dass dieser im Zusammenhang mit der Kontaktflächenberechnung steht.

In diesem Zusammenhang bestehen kommende Schritte darin diesen Fehler zu beheben oder durch die Verwendung alternativer Simulationstools zu umgehen. Dies ermöglicht die Charakterisierung des Wand-Partikel-Wärmeübergangs. Dafür erforderlich ist die Wand-Innentemperatur, welche es zu bestimmen gilt.

Eine durchgeführte Untersuchung der Biot-Zahl von Biomassepartikeln belegte einen nicht zu vernachlässigenden Einfluss der intrapartikulären Wärmeleitung. Durch das Softwarepaket ParScale der TU Graz [148] ist die Möglichkeit gegeben, dies in die DEM-Simulationen zu implementieren. Mit fortschreitender Entwicklung könnte eine Implementierung von Interesse sein, jedoch ist die Anzahl der Partikel, numerisch bedingt und nach aktuellem Kenntnisstand, auf wenige hundert limitiert, sodass dies im Zuge dieser Arbeit nicht in Betracht gezogen werden konnte.

### **CFDDEM- Untersuchung des Doppelschneckenmischreaktors**

Bei der CFDDEM-Simulation des Doppelschneckenmischreaktors wurde das Softwarepaket CFDEM®-Public verwendet. Dieses ist für den Datenaustausch zwischen dem DEM-Tool LIGGGHTS®-Public und der inkompressiblen Strömungssimulation, d.h. OpenFoam verantwortlich. Da die Thermodynamik und die chemischen Reaktionen die kompressible Strömungssimulation erfordern, wurde das Simulationstool um diese Fähigkeit erweitert. Dabei wurde ein Reaktionsmodell für Homogenreaktionen in LIGGGHTS implementiert. Die Behandlung der Strömungssimulation erforderte die Erweiterung um die Kompressibilität und die Implementierung von Methoden zur Darstellung reaktiver Strömungen. In diesem Zuge wurde die PaSR-Toolbox (**P**artially stirred reactor) des rhoReactingFoam-Solvers implementiert.

Aus Stabilitätsgründen wurde es erforderlich ein alternatives Suchmodell, das DEM-Partikeln ihre Position im CFD-Gitter zuordnet, zu implementieren. Das neue Suchmodell ordnet über ganzzahlige Division CFD-Zellen einem virtuellen Gitter zu. Durch ganzzahlige Division der Partikelposition sind diese Zellen den Partikeln zuordenbar.

Eine Strömungssimulation des Doppelschneckenmischreaktors macht eine Behandlung der Schnecken erforderlich. Diese sind co-rotierend und ineinander kämmend. Unter den verfügbaren Algorithmen zur Behandlung von bewegten Objekten in der Strömungssimulation erwies sich das Brinkmann-Strafverfahren als das einzig einsetzbare Verfahren. Dieses wurde programmiert und zeigte sich am Modellversuch der Karmanschen Wirbelstraße als geeignet. Jedoch konnte im Zusammenhang mit der Partikelsimulation eine Instabilität im Objekt-Randbereich beobachtet werden, sodass es erforderlich wurde, den Quellterm der DEM-Simulation in diesen Bereichen zu eliminieren. Dies führte zu einem systematischen Fehler bei der Massen- und Energieübertragung von DEM zu CFD von rund 10%. Zum jetzigen Zeitpunkt ist das Brinkmann-Strafverfahren der einzige auf diese Problemstellung anwendbare Algorithmus, sodass der Massen- und Energieverlust hingenommen werden muss. Durch die Weiter- und Neuentwicklung von Algorithmen zur Objektdarstellung kann es in zukünftigen Arbeiten von Interesse sein, das Brinkmann-Strafverfahren zu ersetzen.

Die chemischen Reaktionen von Biomasse bei der Pyrolyse wurden als Zerfallsreaktionen erster Ordnung der Biomassekomponenten Cellulose, Hemicellulose, Lignin und Wasser interpretiert. Parameter wie Aktivierungsenergie, Frequenzfaktor und die Weizenstrohzusammensetzung wurden im Zuge einer TGA-Auswertung mittels Parameterschätzung bestimmt und liegen in dem von Angot. et. al. [107] ermittelten Wertebereich. Die molaren Massen der Pyrolyseprodukte wurden aus einem Versuch mit Weizenstroh in einem semi-kontinuierlichen Pyrolysereaktor über die Verweilzeit bestimmt. Dazu wurde das aus dem Reaktor austretende Gas zunächst von Kokspartikeln gefiltert und anschließend vollständig in Kohlenstoffdioxid umgewandelt. Der CO<sub>2</sub>-Konzentrationsverlauf wurde in einem Massenspektrometer online erfasst. Eine Auswertung erfolgte auch hier mittels Parameterschätzung. Beiden Parameterschätzungen weisen eine Modellgüte  $R^2 > 0.91$  auf.

Aufgrund des numerischen Aufwandes wurde nur eine Simulation bei 2 rpm durchgeführt.

CFDEM®-Public bietet die Möglichkeit den Wärmeübergang aus der Gasphase nach Li. et. al. [123] zu berechnen. Dieser wird zusätzlich zur Wärmeleitung durch die Kontaktstelle berechnet. Aufgrund einer zu befürchtenden Interaktion mit der Wärmekonvektion aus der Gasphase wurde im DEM-Teil der Simulation der Wärmeübergang nach Batchelor nach Literatur berechnet. Es zeigte sich, dass der auf diese Weise berechnete Wärmeübergangskoeffizient um einen Faktor von vier kleiner ist, als der bei einer DEM-Simulation mit dem modifizierten Batchelor Algorithmus. Dies erlaubt die Schlussfolgerung, dass der Wärmeübergang um die laminare Grenzschicht der Kontaktstelle keinen wesentlichen Einfluss darstellt und verdeutlicht das Entwicklungspotential bei den Modellen zur Darstellung des Wärmeübergangs in gekoppelten CFD-DEM-Simulationen. Nachfolgende Arbeiten können darauf anknüpfend den Partikelkontakt als Einflussgröße implementieren. Jedoch ist es erforderlich den Einfluss der fluiden Konvektion am Wärmeübergang durch den Partikel-Partikel-Kontakt zu charakterisieren.

Im weiteren Verlauf wurde beobachtet, dass die Rückvermischung bis in den nicht begleitbeheizten Biomasse Transportbereich hineinragt und dort zu einer nachteiligen Abkühlung der Wärmeträger über die Reaktorwand führt.

Darüber hinaus konnte ein lokaler Versatz des Wasser-Verdampfungsbereichs und des Reaktionsbereichs festgestellt werden, sodass durch eine Modifikation des Auslassbereichs Einfluss auf die Produktqualität genommen werden kann.

Eine Betrachtung des Reaktionsverlaufes im Reaktor zeigt, dass die Verweilzeit nicht ausreicht, um Biomasse vollständig zu zersetzen. Weder beim thermisch stabilen Lignin, noch bei der Cellulose konnte eine wesentliche Zersetzung beobachtet werden. Eine Sensitivitätsanalyse der Reaktionskinetik, bei der Reaktionen in einem breiten Temperaturbereich betrachtet wurden, bestätigt diese Aussage. Die Reaktionsgeschwindigkeit kann demnach nur zum Teil auf den Massen- und Energieverlust bei der Übertragung von DEM zur CFD-Simulation und den geringeren Wärmeübergang zurückgeführt werden.

Unter der Voraussetzung, dass die Reaktionskinetik das reale Verhalten widerspiegelt, erfordert dies die Verlängerung der Verweilzeit im Reaktor. Aufgrund von Verblockungen im niedrigen Drehzahlbereich macht dies eine Verlängerung des Reaktors erfolgen. TG-Analysen weisen Heizraten von bis zu  $50 \text{ K min}^{-1}$  auf. Bei der Schnellpyrolyse von Biomasse können Heizraten von über  $500 \text{ K s}^{-1}$  beobachtet werden, sodass ein abweichendes Reaktionsverhalten auch auf einen Übertrag von TGA auf die Schnellpyrolyse zurückzuführen ist. Daher kann es von Interesse sein, die Kinetik basierend auf alternativen Verfahren zu bestimmen.



# Appendix

## Semi-kontinuierlicher Pyrolysereaktor

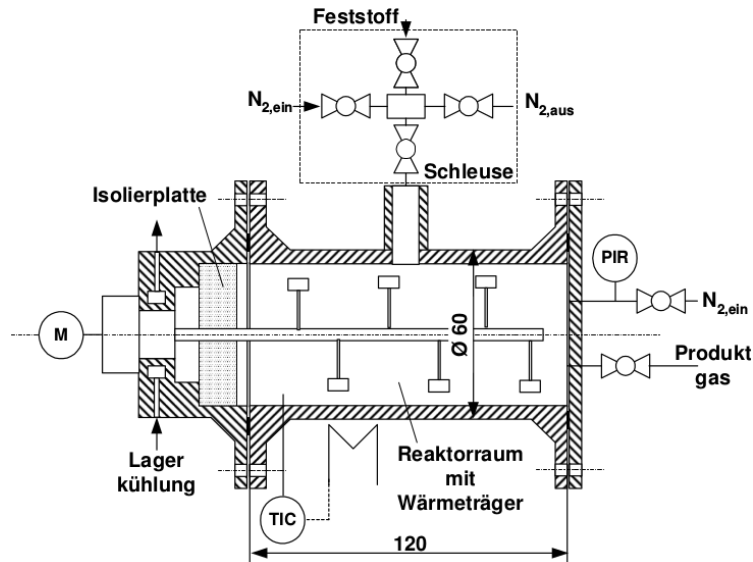


Abbildung 0.1: semi-kontinuierlicher Pyrolysereaktor

## Bestimmung der Verweilzeit und axialen Dispersion im Versuchsaufbau

Tabelle 0.1: Geometrische Daten Des semi-kontinuierlichen Pyrolysereaktors mit Oxidationsreaktor

		Daten Reaktorenteile					
		Länge L [m]	Durchmesser d [m]	Porosität $\epsilon$	V [m <sup>3</sup> ]	V <sub>OXI,ges</sub> [m <sup>3</sup> ]	V <sub>ges</sub> [m <sup>3</sup> ]
nur OXI	OXI	1	0,05	0,85	0,006675884	0,0078697	0,0095096
	Wärmeübertrager	0,4	0,02	1	0,000502655		
	Gasheizer vor OXI	0,3	0,02	1	0,000376991		
	WÜ-MS	0,15	0,02	1	0,000188496		
	Rohrleitungen, Ne	2,5	0,004	1	0,000125664		
Pyrolyse	Pyrolyse	0,12	0,06	1	0,001357168		
	Gasheizer vor PYR	0,4	0,015	1	0,000282743		

Tabelle 0.2: Versuchsbedingungen zur Bestimmung der Verweilzeit im semi-kontinuierlichen Pyrolysereaktor mit Oxidationsstufe

V(20°C)	0,000503333	m <sup>3</sup> /s
N(20°C)	0,020651698	mol/s
$\dot{V}$ (500°C)	0,001327485	m <sup>3</sup> /s

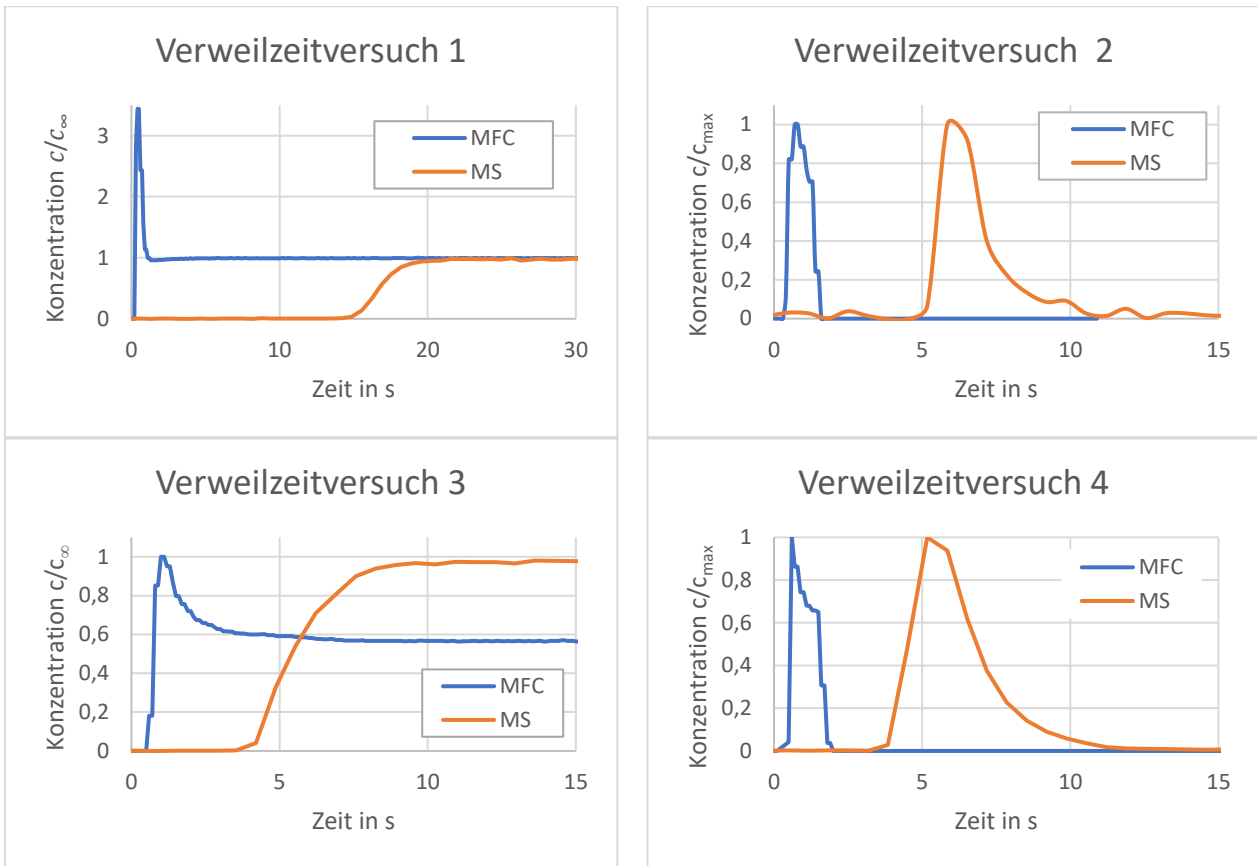
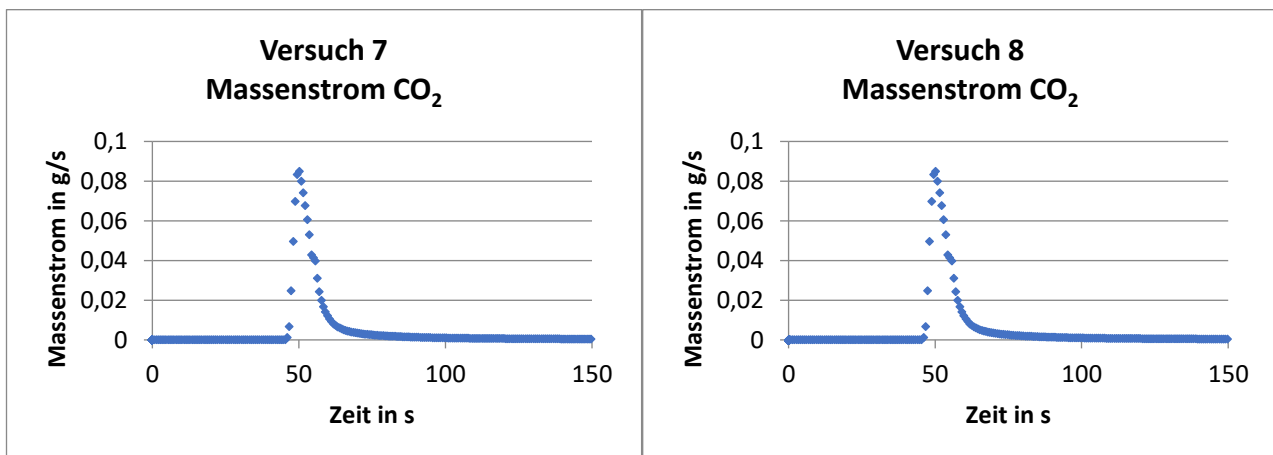
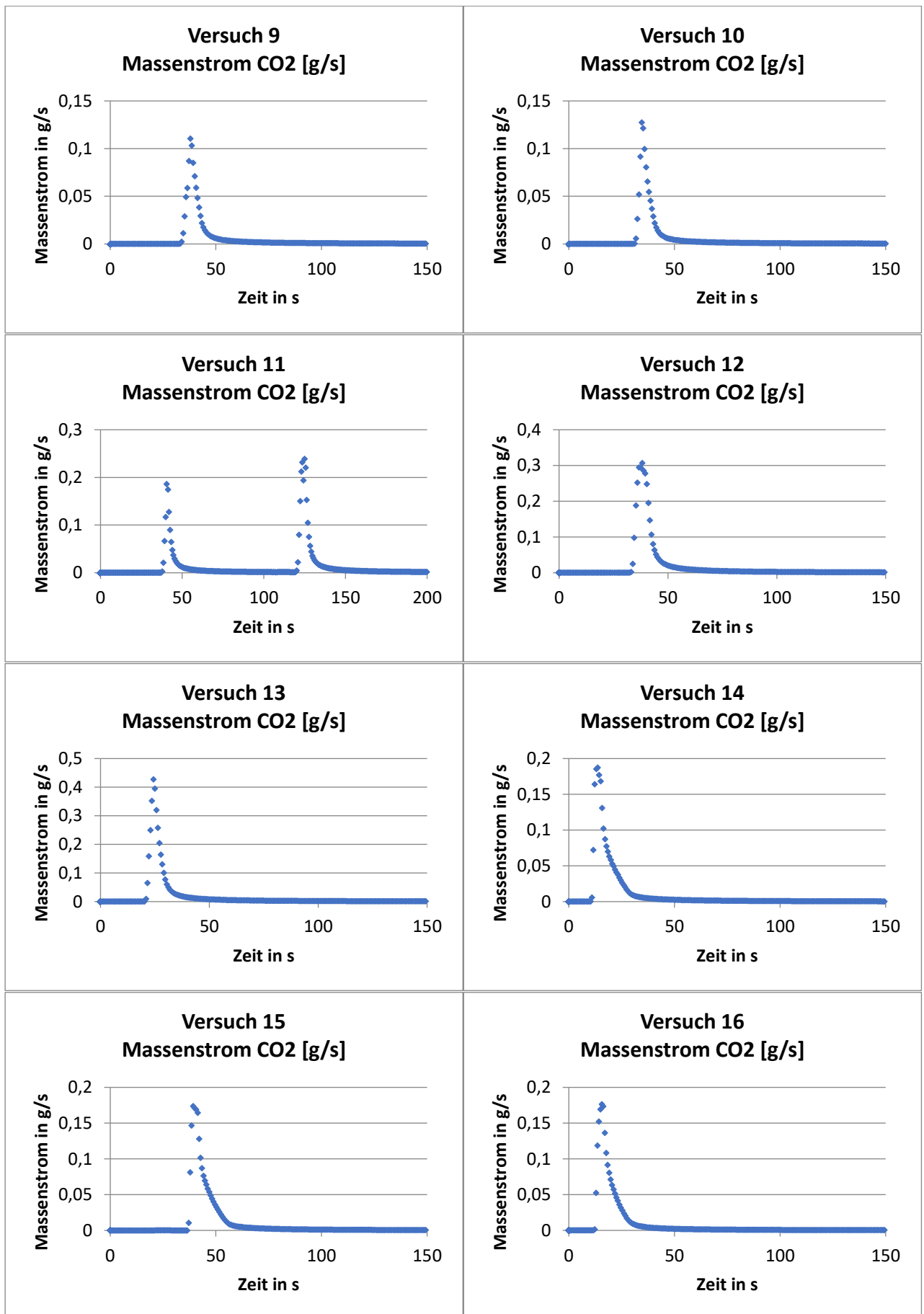


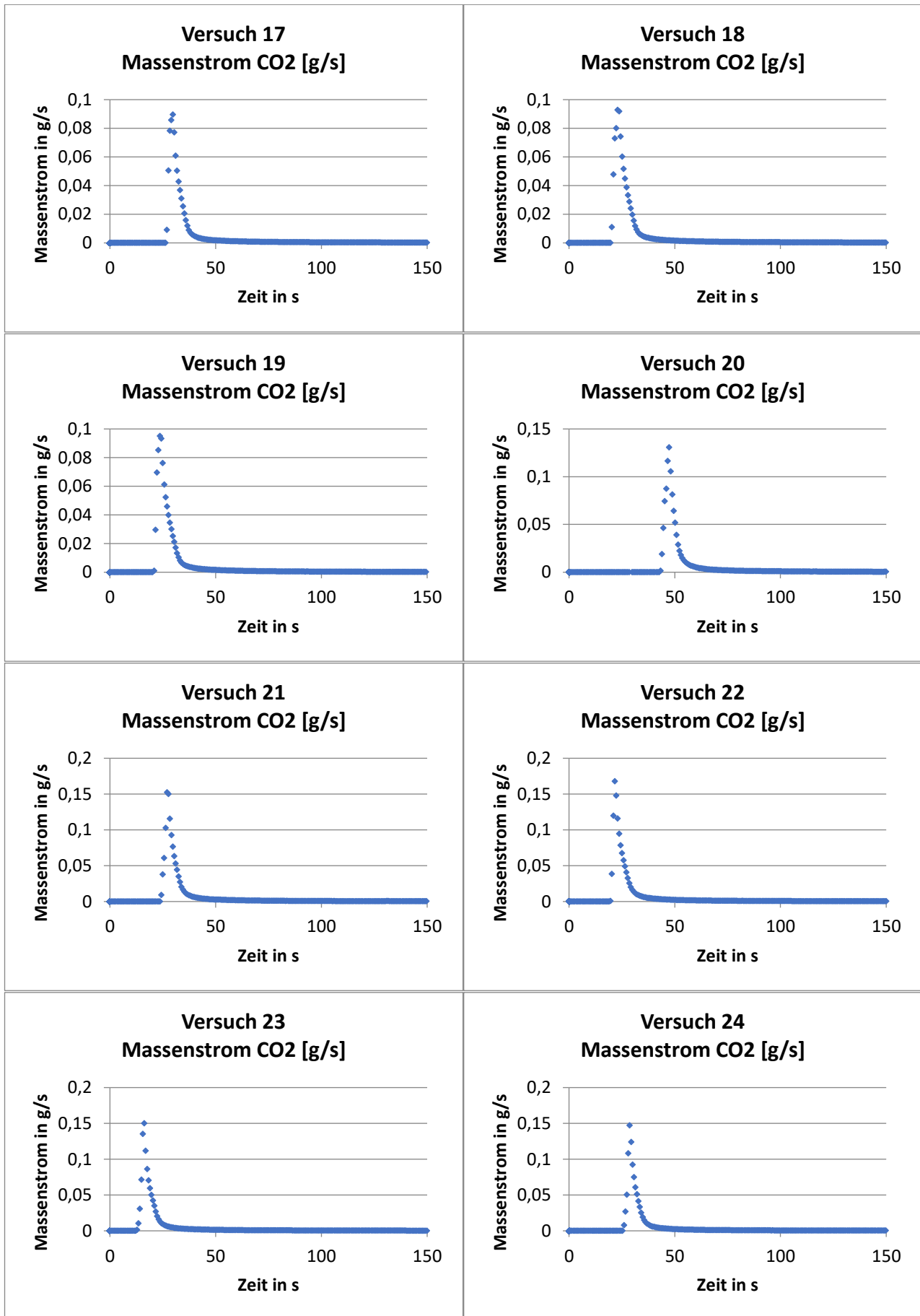
Abbildung 0.2: Eingangs- und Ausgangssignal im Versuchsaufbau. Marker: Neon. Verweilzeitversuch 1 weicht von Hydrodynamischer Verweilzeit erheblich ab. Mögliche Ursachen: Technischer defekt, Fehlbedienung

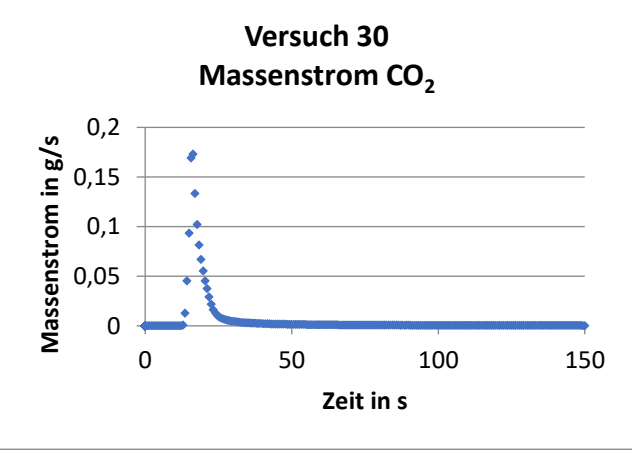
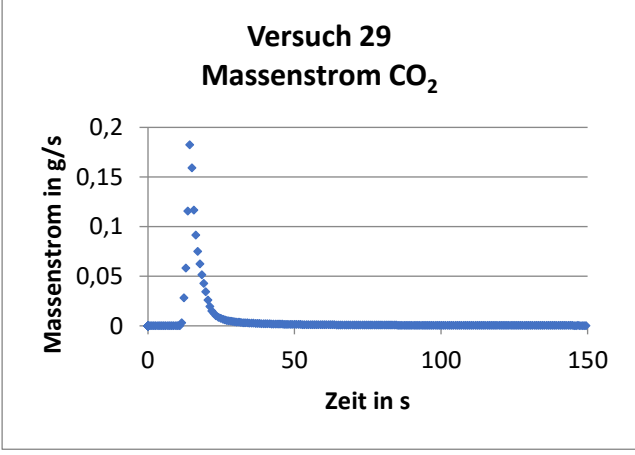
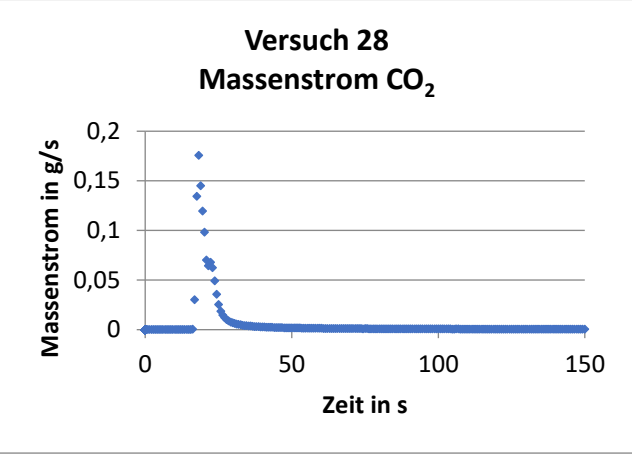
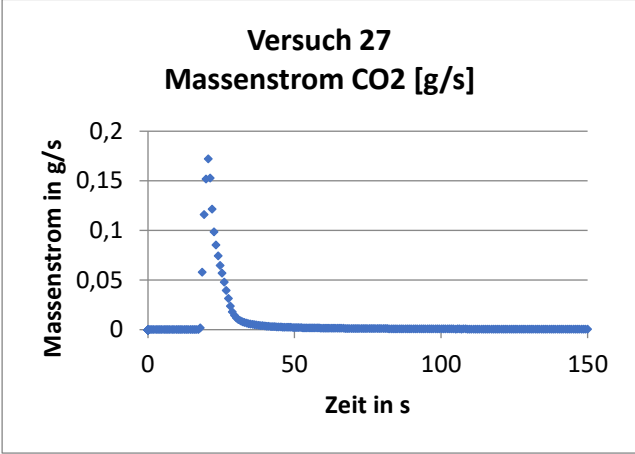
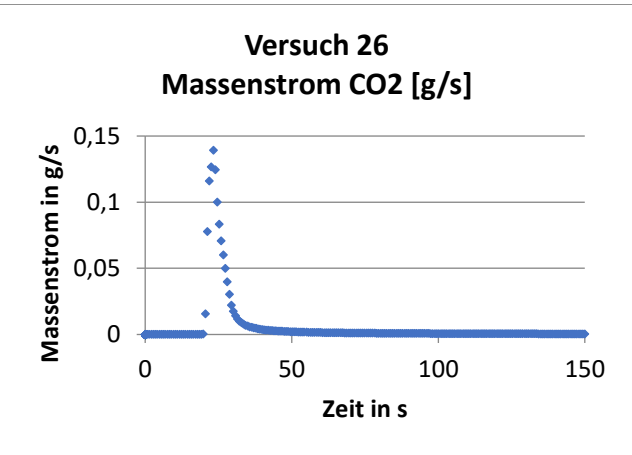
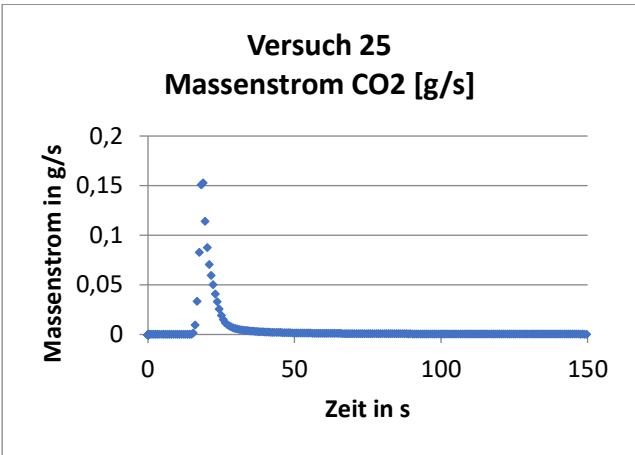
## Pyrolyse im semi-kontinuierlichen Bach-Pyrolysereaktor

Bei den folgenden Versuchen wurde 1 g Weizenstroh in der Schläufe Abbildung 0.1 inertisiert. Beim Auslösen der Schläufe gerät dies in den Bach-Pyrolysereaktor und wird dort Pyrolysiert. Das Pyrolysegas wird im Oxidationsreaktor in  $CO_2$  umgewandelt. Die Konzentration des  $CO_2$  und des Ne-Markers wird mit einem Massenspektrometer erfasst. Da die Ne-Konzentration bekannt ist, kann diese genutzt werden um auf die  $CO_2$ -Konzentration rückzuschließen. Versuche 1-6 dienten der bestimmung der optimalen versuchsbedingungen und werden daher nicht aufgeführt.









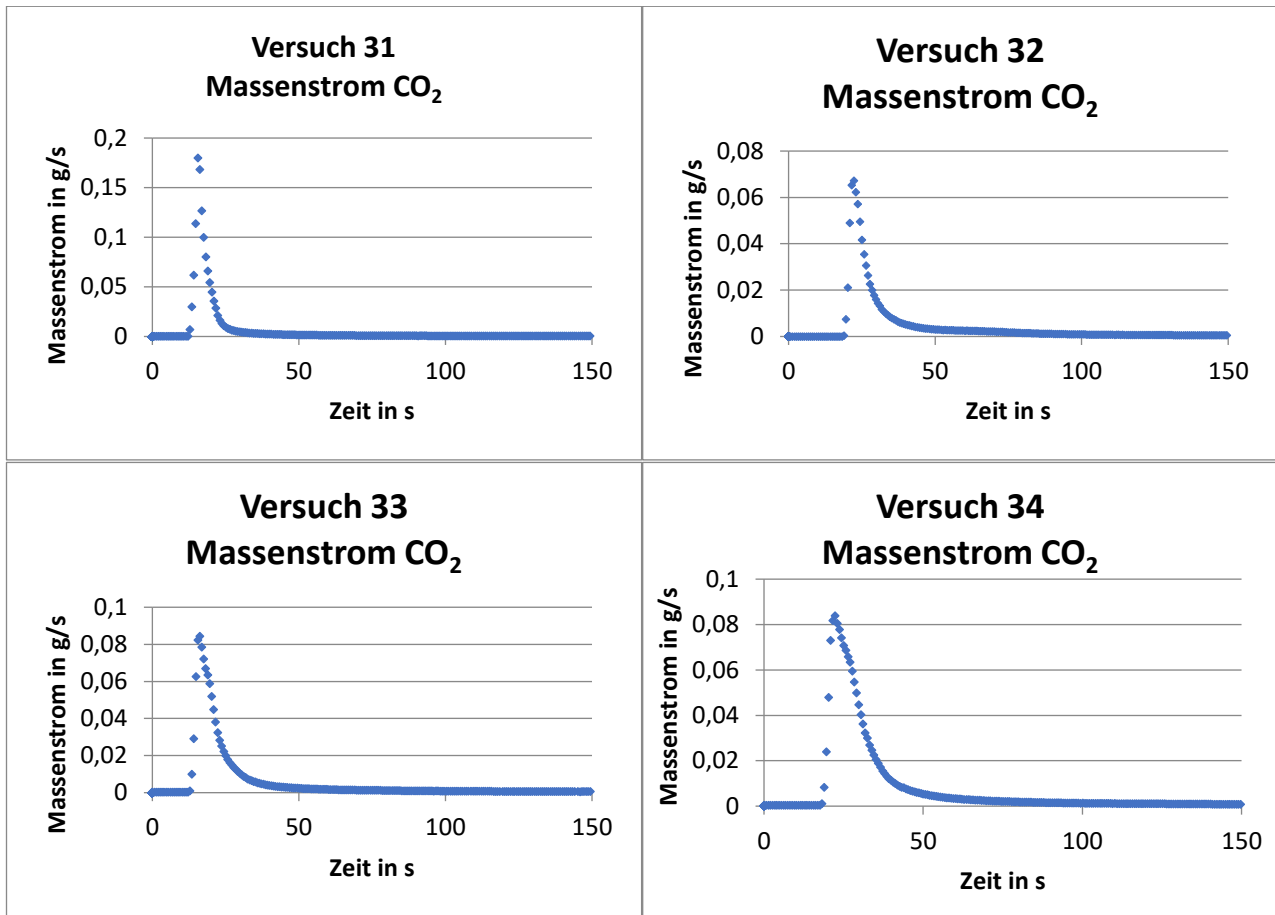


Abbildung 0.3: Versuche mit Biomasse am semi-kontinuierlichen Pyrolysereaktor.

## Umrechnung der TG-Reaktionskinetik

Für den Zerfall/Verdampfung von Wasser, Cellulose, Hemicellulose und Lignin gilt:

$$\begin{aligned}\frac{dc_X}{dt} &= -k_X \cdot c_X \\ \frac{dn_X \cdot M_X}{dt} &= -k_X \cdot n_X \cdot M_X \\ \frac{dn_X}{dt} &= -k_X \cdot n_X\end{aligned}\quad (0.1)$$

Wobei  $X$  für Wasser, Cellulose, Hemicellulose oder Lignin steht. Die Masse  $C_X$  der Komponente  $X$  wird durch die Verwendung deren molaren Masse  $M_X$  in die Stoffmenge  $n_X$  umgewandelt.

Für das Koks  $K$  gilt:

$$\begin{aligned}\frac{dc_K}{dt} &= \alpha_C \cdot k_C \cdot c_C + \alpha_H \cdot k_H \cdot c_H + \alpha_L \cdot k_L \cdot c_L \\ \frac{dn_K \cdot M_K}{dt} &= \alpha_C \cdot k_C \cdot n_C \cdot M_C + \alpha_H \cdot k_H \cdot n_H \cdot M_H + \alpha_L \cdot k_L \cdot n_L \cdot M_L \\ \frac{dn_K}{dt} &= \underbrace{\frac{\alpha_C \cdot M_C}{M_K}}_{=\psi_C} \cdot k_C \cdot n_C + \underbrace{\frac{\alpha_H \cdot M_H}{M_K}}_{=\psi_H} \cdot k_H \cdot n_H + \underbrace{\frac{\alpha_L \cdot M_L}{M_K}}_{=\psi_L} \cdot k_L \cdot n_L\end{aligned}\quad (0.2)$$

Dabei steht  $C$  für Cellulose,  $H$  für Hemicellulose und  $L$  für Lignin. Analog muss für das Gas gelten:

$$\frac{dn_G}{dt} = \underbrace{\frac{\beta_C \cdot M_C}{M_K}}_{=\omega_C} \cdot k_C \cdot n_C + \underbrace{\frac{\beta_H \cdot M_H}{M_K}}_{=\omega_H} \cdot k_H \cdot n_H + \underbrace{\frac{\beta_L \cdot M_L}{M_K}}_{=\omega_L} \cdot k_L \cdot n_L\quad (0.3)$$

mit

$$1 = \alpha_X + \beta_X \quad \forall X$$

# Programme

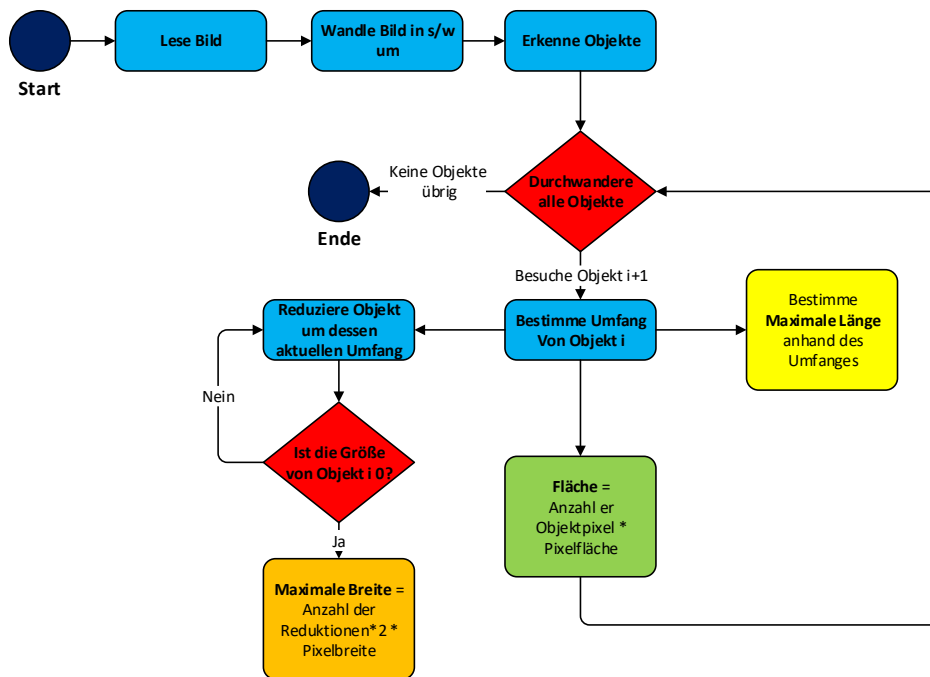


Abbildung 0.4: Programm: Weizenstrohformauswertung

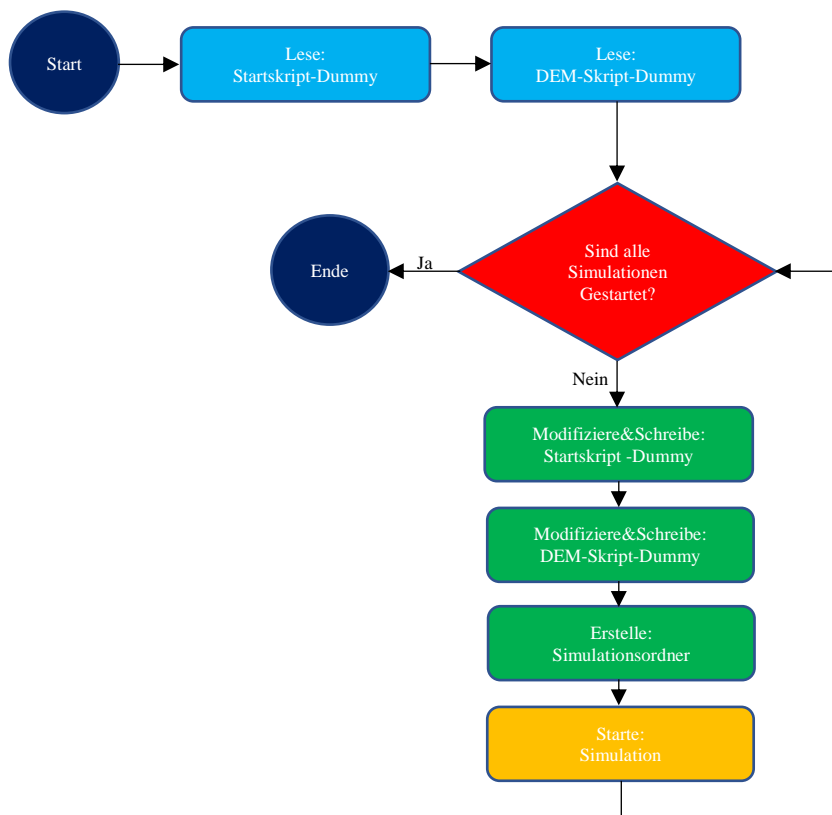


Abbildung 0.5: Programm: Start von Simulationsgruppen auf dem Cluster







```

}

penalty.update();

Ksl = particleCloud.momCoupleM(0).impMomSource();

Ksl.correctBoundaryConditions();

surfaceScalarField voidfractionf = fvc::interpolate(voidfraction);

//Force Checks
#include "forceCheckIm.H"

volScalarField voidfraction_common("voidfraction_common",voidfraction);

penalty.voidfractionUpdate(voidfraction_common);

surfaceScalarField voidfractionf_common = fvc::interpolate(voidfrac-
tion_common);

phi = voidfractionf_common*phiByVoidfraction;

#include "solverDebugInfo.H"
particleCloud.clockM().stop("Coupling");

particleCloud.clockM().start(26,"Flow");

volScalarField Fluid("Fluid", 1-penalty.Lambda());
volScalarField voidfraction_Fluid("voidfraction_Fluid",voidfraction);

forAll(Fluid,cellI)
{
    if (Fluid[cellI]<1)
    {
        Fluid[cellI]=0.0;
        voidfraction_Fluid[cellI]=1.0;
    }
}

Info<< "void gas min/max    = " << gMin(voidfraction.internalField()) << ", "
    << gMax(voidfraction.internalField()) << endl;

Info<< "void_fluid gas min/max    = " << gMin(voidfraction_Fluid.internal-
Field()) << ", "
    << gMax(voidfraction_Fluid.internalField()) << endl;

surfaceScalarField voidfractionf_Fluid = fvc::interpolate(voidfrac-
tion_Fluid);

volScalarField voidfractionRho = voidfraction_common * rho;

if(particleCloud.solveFlow())
{
#       include "chemistry.H"

#       include "rhoEqn_Eigen.H"
        voidfractionRho = voidfraction_common * rho;

        for (label ocorr=1; ocorr <= nOuterCorr; ocorr++)

```

```

#         {
#             include "UEqn.H"

//         thermo.T() = min(thermo.T(), 800.0);

#             include "YEqn.H"

#             include "hsEqn.H"

//         --- PISO loop
//         for (int corr=1; corr<=nCorr; corr++)
//         {

#                 include "pEqn3.H"

//         }

//         turbulence->correct();
//         thermo.T() = min(thermo.T(), 800.0 );

//         rho = thermo.rho();

//         if (runTime.write())
//         {
//             chemistry.dQ()().write();
//             massSource.write();
//         }

//         Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
//             << "   ClockTime = " << runTime.elapsedClockTime() << " s"
//             << nl << endl;

//         particleCloud.clockM().stop("Flow");

//         particleCloud.clockM().stop("Global");
//     }

//     Info << "Pmin=" << gMin(p.internalField()) << ", Pmax=" << gMax(p.inter-
//     nalField()) << endl;

// }

// Info<< "End\n" << endl;

// return 0;
// }

// ***** //

```

## //Speicherort: ~\foamPy\createFields.H

```
Info<< nl << "Reading thermophysicalProperties" << endl;
autoPtr<rhoChemistryModel> pChemistry
(
    rhoChemistryModel::New(mesh)
);

rhoChemistryModel& chemistry = pChemistry();

hsReactionThermo& thermo = chemistry.thermo();

basicMultiComponentMixture& composition = thermo.composition();

PtrList<volScalarField>& Y = composition.Y();

word inertSpecie(thermo.lookup("inertSpecie"));

volScalarField rho
(
    IOobject
    (
        "rho",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    thermo.rho()
);

volScalarField Cp
(
    IOobject
    (
        "Cp",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    thermo.Cp()
);

mesh.checkIn(Cp);

Info<< "Reading field U\n" << endl;
volVectorField U
(
    IOobject
    (
        "U",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);

volScalarField& p = thermo.p();
```

```

const volScalarField& psi = thermo.psi();

volScalarField& hs = thermo.hs();

const volScalarField &T = thermo.T();

volScalarField kappa
(
    IObject
    (
        "kappa",
        runTime.timeName(),
        mesh,
        IObject::NO_READ,
        IObject::AUTO_WRITE
    ),
    mesh,
    dimensionedScalar("zero", dimless, 0.0)
);

multivariateSurfaceInterpolationScheme<scalar>::fieldTable fields;

forAll(Y, i)
{
    fields.add(Y[i]);
}

fields.add(hs);

DimensionedField<scalar, volMesh> chemistrySh
(
    IObject
    (
        "chemistry:Sh",
        runTime.timeName(),
        mesh,
        IObject::NO_READ,
        IObject::NO_WRITE
    ),
    mesh,
    dimensionedScalar("chemistrySh", dimEnergy/dimTime/dimVolume, 0.0)
);

//=====
// particle interaction modelling
//=====

Info<< "\nReading momentum exchange field Ksl\n" << endl;
volScalarField Ksl
(
    IObject
    (
        "Ksl",
        runTime.timeName(),
        mesh,
        IObject::MUST_READ,
        IObject::AUTO_WRITE
    ),
    mesh,
    //dimensionedScalar("0", dimensionSet(1, -3, -1, 0, 0), 1.0)
);

```

```

    Info<< "\nReading voidfraction field voidfraction = (Vgas/Vparticle)\n" <<
endl;
    volScalarField voidfraction
    (
        IOobject
        (
            "voidfraction",
            runTime.timeName(),
            mesh,
            IOobject::MUST_READ,
            IOobject::AUTO_WRITE
        ),
        mesh
    );

#include "compressibleCreatePhi_eigen.H"

    Info<< "Reading particle velocity field Us\n" << endl;
    volVectorField Us
    (
        IOobject
        (
            "Us",
            runTime.timeName(),
            mesh,
            IOobject::MUST_READ,
            IOobject::AUTO_WRITE
        ),
        mesh
    );

    Info<< "Creating field DpDt\n" << endl;
    volScalarField DpDt =
        fvc::DDt(surfaceScalarField("phiU", phi/fvc::interpolate(rho)), p);

//=====
// scalar field modelling
//=====
/* Info<< "\nCreating dummy density field rho = 1\n" << endl;
    volScalarField T
    (
        IOobject
        (
            "T",
            runTime.timeName(),
            mesh,
            IOobject::MUST_READ,
            IOobject::AUTO_WRITE
        ),
        mesh//,
        //dimensionedScalar("0", dimensionSet(0, 0, -1, 1, 0), 273.15)
    );*/

    Info<< "\nCreating fluid-particle heat flux field\n" << endl;
    volScalarField TSource
    (
        IOobject
        (
            "TSource",
            runTime.timeName(),
            mesh,
            IOobject::NO_READ,

```

```

        IOobject::NO_WRITE
    ),
    mesh,
    dimensionedScalar("0", dimensionSet(0, 0, -1, 1, 0), 0.0)
);

volScalarField QSource
(
    IOobject
    (
        "QSource",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    mesh,
    dimensionedScalar("0", dimensionSet(1, 2, -3, 0, 0), 0.0)
);

/*IOdictionary transportProperties
(
    IOobject
    (
        "transportProperties",
        runTime.constant(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    )
);

dimensionedScalar DT
(
    transportProperties.lookup("DT")
);*/

//=====
/*
//# include "createPhi.H"
#ifdef createPhi_H
#define createPhi_H
Info<< "Reading/calculating face flux field phi\n" << endl;
surfaceScalarField phi
(
    IOobject
    (
        "phi",
        runTime.timeName(),
        mesh,
        IOobject::READ_IF_PRESENT,
        IOobject::AUTO_WRITE
    ),
    linearInterpolate(U*voidfraction) & mesh.Sf()
);
#endif
*/

Info<< "Generating interstitial face flux field phiByVoidfraction\n" << endl;
surfaceScalarField phiByVoidfraction
(
    IOobject
    (

```



```

        "phiByVoidfraction",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    fvc::interpolate(rho)*(linearInterpolate(U) & mesh.Sf())
);

    label pRefCell = 0;
    scalar pRefValue = 0.0;
//    setRefCell(p, mesh.solutionDict().subDict("PISO"), pRefCell, pRefValue);
    setRefCell(p, mesh.solutionDict().subDict("PIMPLE"), pRefCell, pRefValue);

//singlePhaseTransportModel laminarTransport(U, phi);
/*
    autoPtr<incompressible::turbulenceModel> turbulence
    (
        incompressible::turbulenceModel::New(U, phi, laminarTransport)
    );
*/

Info << "Creating turbulence model.\n" << nl;
autoPtr<compressible::turbulenceModel> turbulence
(
    compressible::turbulenceModel::New
    (
        rho,
        U,
        phi,
        thermo
    )
);

volScalarField massSource
(
    IOobject
    (
        "massSource",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    mesh,
    dimensionedScalar("zero", dimensionSet(1, -3, -1, 0, 0) , 0.0)
);

volScalarField oneByVol
(
    IOobject
    (
        "oneByVol",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),

```

```

    mesh,
    1/dimensionedScalar("zero", dimVolume, 1.0)
);
oneByVol.internalField()/=mesh.V().field();

const IOdictionary scalarTransportProperties //this is clumsy,
but effective
(
    IOobject
    (
        "scalarTransportProperties",
        mesh.time().constant(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    )
);

dictionary          simplePropsDict (scalarTransportProperties.subDict("simple-
ModelProps"));
scalar              nComponents (simplePropsDict.lookupOrDefault<scalar>("nCompo-
nents",0.0));

const IOdictionary PenaltyDict_ //this is clumsy, but effective
(
    IOobject
    (
        "PenaltyDict",
        mesh.time().constant(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    )
);

scalar              inertPenalty (PenaltyDict_.lookupOrDefault<scalar>("inertPen-
alty",1000.0));

```

**//Speicherort: ~\foamPy\chemistry.H**

```
{
  Info<< "Solving chemistry" << endl;

  chemistry.solve
  (
    runTime.value() - runTime.deltaT().value(),
    runTime.deltaT().value()
  );

  // turbulent time scale
  if (turbulentReaction)
  {
    volScalarField tk =
      Cmix*sqrt(turbulence->muEff()/rho/turbulence->epsilon());

    volScalarField tc = chemistry.tc();

    // Chalmers PaSR model
    kappa = (runTime.deltaT() + tc)/(runTime.deltaT() + tc + tk);
  }
  else
  {
    kappa = 1.0;
  }

  chemistrySh = kappa*chemistry.Sh()();
}
```



```
//Speicherort: ~\foamPy\hsEqn.H
```

```
{  
  
QSource=particleCloud.forceM(1).getScalarField("QConversion");  
  
Cp.internalField() = thermo.Cp();  
  
    Info<< "void_fluid gas min/max    = " << gMin(voidfraction_Fluid.internal-  
Field()) << ", "  
        << gMax(voidfraction_Fluid.internalField()) << endl;  
  
particleCloud.forceM(3).manipulateScalarField(TSource);  
  
TSource.correctBoundaryConditions();  
  
particleCloud.forceM(3).commToDEM();  
  
volScalarField PressureChange("PressureChange", voidfraction_Fluid*DpDt );  
volScalarField TempSource("TempSource", Fluid*Cp*rho*TSource);  
volScalarField MassTempSource("MassTempSource", Fluid*oneByVol*QSource*massCor-  
rection);  
  
Info << "\t Heatexchange Energie: \t min = " << gMin(TempSource.internalField())  
<< "\t max = " << gMax(TempSource.internalField()) << endl;  
Info << "\t Massexchange Energie: \t min = " << gMin(MassTempSource.internal-  
Field()) << "\t max = " << gMax(MassTempSource.internalField()) << endl;  
Info << "\t ChemReaction Energie: \t min = " << gMin(chemistrySh) << "\t max = "  
<< gMax(chemistrySh) << endl;  
  
dimensionedScalar T_soll("Penalty",dimensionSet(0,0,0,1,0,0,0),773.15);  
dimensionedScalar T_penalty("Penalty",dimensionSet(0,0,-1,0,0,0,0),inertPen-  
alty);  
  
volScalarField ThermalCorrection("ThermalCorrection", PressureChange + MassTemp-  
Source );  
  
forAll(ThermalCorrection, cellI)  
{  
    if ((thermo.T()[cellI]>=800.0)&&(ThermalCorrection[cellI]>0))  
    {  
        ThermalCorrection[cellI]=0;  
    }  
    else  
    {  
        ThermalCorrection[cellI]+=chemistrySh[cellI];  
    }  
}  
  
fvScalarMatrix hsEqn  
(  
    fvm::ddt(voidfraction_Fluid*rho, hs)  
    + mvConvection->fvmDiv(voidfractionf_Fluid*phiByVoidfraction, hs)  
    - fvm::laplacian(turbulence->alphaEff()*voidfraction_Fluid, hs)  
//    - fvm::laplacian(turbulence->muEff()*voidfraction_common, hs)  
==  
    ThermalCorrection  
/*  
    PressureChange  
    + chemistrySh  
    + TempSource  
    + MassTempSource
```

```

*/

);

hsEqn.relax();

hsEqn.solve();

if (runTime.write())
{
    TempSource.write();
    MassTempSource.write();
}

scalar lb = 293.15;
scalar ub = 800.0;
// thermo.limitT(lb,ub);

Info<< "hs gas min/max = " << gMin(hs.internalField()) << ", "
    << gMax(hs.internalField()) << endl;
volScalarField temp("test",hs/rho/Cp);
Info<< "hs/rho/cp gas min/max = " << gMin(temp.internalField()) << ", "
    << gMax(temp.internalField()) << endl;

thermo.correct();
Info<< "T gas min/max before correction = " << gMin(T.internalField()) <<
", "
    << gMax(T.internalField()) << endl;
thermo.limitT(lb,ub);

Info<< "T gas min/max = " << gMin(T.internalField()) << ", "
    << gMax(T.internalField()) << endl;
Info<< "hs gas min/max = " << gMin(hs.internalField()) << ", "
    << gMax(hs.internalField()) << endl;
}

```

**//Speicherort: ~\foamPy\pEqn3.H**

```
{
    rho = thermo.rho();

    //thermo.rho() -= psi*p;

    volScalarField rAU = 1.0/UEqn.A();
    U = rAU*UEqn.H();

    surfaceScalarField rAUf("1|A(U)", fvc::interpolate(rho*rAU));
    volScalarField rAUvoidfraction("(voidfraction2|A(U)", rho*rAU*voidfrac-
tion_Fluid);
    surfaceScalarField rAUfvoidfraction("(voidfraction2|A(U)F)", fvc::interpo-
late(rAUvoidfraction));
    //surfaceScalarField phiS(fvc::interpolate(parti-
cleCloud.forceM(0).rhoField()*Us) & mesh.Sf());

    phi = ( fvc::interpolate(U*rho) & mesh.Sf() );
    surfaceScalarField phiS(fvc::interpolate(Us) & mesh.Sf());
    phi += rAUf*(fvc::interpolate(Ksl) * phiS);

    for (int nonOrth=0; nonOrth<=nNonOrthCorr; nonOrth++)
    {
        // Pressure corrector
        fvScalarMatrix pEqn
        (
            fvm::ddt(voidfraction_Fluid*psi, p)
            + fvc::div(voidfractionf_Fluid*phi)
            - fvm::laplacian(rAUvoidfraction, p)
            ==
            Fluid*rho*particleCloud.ddtVoidfraction()
            + massSource
        );

        if
        (
            ocorr == nOuterCorr
            && corr == nCorr
            && nonOrth == nNonOrthCorr
        )
        {
            pEqn.solve(mesh.solutionDict().solver(p.name() + "Final"));
        }
        else
        {
            pEqn.solve();
        }

        if (nonOrth == nNonOrthCorr)
        {
            phiByVoidfraction = phi + pEqn.flux()/voidfractionf_Fluid;
        }
    }

    phi = voidfractionf_common*phiByVoidfraction;
    #include "rhoEqn_Eigen.H"
    #include "compressibleContinuityErrs.H"
    // Second part of thermodynamic density update
    //thermo.rho() += psi*p;

    p.relax();
}
```

```

rho = thermo.rho();
rho.relax();
Info<< "rho max/min : " << max(rho).value()
    << " " << min(rho).value() << endl;

if (modelType=="B" || modelType=="Bfull")
    U -= rAU*fvc::grad(p) - Fluid*Ksl*Us*rAU ;
else
    U -= voidfraction_common*rAU*fvc::grad(p) - Fluid*Ksl*Us*rAU ;

U.correctBoundaryConditions();
p.correctBoundaryConditions();

DpDt = fvc::DDt(surfaceScalarField("phiU", phi/fvc::interpolate(rho)), p);
}

```



```
//Speicherort: ~\foamPy\readChemistryProperties.H
```

```
Info<< "Reading chemistry properties\n" << endl;
```

```
IOdictionary chemistryProperties
```

```
(  
    IOobject  
    (  
        "chemistryProperties",  
        runtime.constant(),  
        mesh,  
        IOobject::MUST_READ,  
        IOobject::NO_WRITE,  
        false  
    )  
);
```

```
Switch turbulentReaction(chemistryProperties.lookup("turbulentReaction"));
```

```
dimensionedScalar Cmix("Cmix", dimless, 1.0);
```

```
if (turbulentReaction)  
{  
    chemistryProperties.lookup("Cmix") >> Cmix;  
}
```

## //Speicherort: ~\foamPy\rhoEqn\_Eigen.H

```
/*-----*\
===== |
\\      /  F ield      | foam-extend: Open Source CFD
\\      /  O peration  | Version:      3.2
\\      /  A nd        | Web:         http://www.foam-extend.org
  \\    /  M anipulation | For copyright notice see file Copyright
-----*\

License
  This file is part of foam-extend.

  foam-extend is free software: you can redistribute it and/or modify it
  under the terms of the GNU General Public License as published by the
  Free Software Foundation, either version 3 of the License, or (at your
  option) any later version.

  foam-extend is distributed in the hope that it will be useful, but
  WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
  General Public License for more details.

  You should have received a copy of the GNU General Public License
  along with foam-extend. If not, see <http://www.gnu.org/licenses/>.

Global
  rhoEqn

Description
  Solve the continuity for density.

/*-----*\
//volScalarField oneByVol=1; //("oneByVol",1/(voidfraction*dimensioned-
Scalar("dummy", dimVolume, 1.0)));
//oneByVol.internalField()/=mesh.V().field()*dimensionedScalar("dummy", dim-
Volume, 1.0);

{
    volScalarField MSource("dummy", particleCloud.forceM(1).getScalar-
Field("totalFlux"));
    scalar massFluxTotal = gSum(MSource.internalField());
    if (massFluxTotal>0)
        massCorrection =min(1, 2.7778e-3/massFluxTotal);

    massSource=Fluid*MSource*oneByVol*massCorrection;
    Info << "\t sum(totalFlux)=" << gSum(MSource.internalField()) << "\t correc-
tionFaktor=" << massCorrection << endl;

    solve(
        fvm::ddt(voidfraction_common, rho)
        + fvc::div(voidfractionF_common*phiByVoidfraction)
        ==
        massSource
    );

    Info<< "rho max/min : " << gMax(rho.internalField())
        << " " << gMin(rho.internalField()) << endl;
}

// ***** //
```

**//Speicherort: ~\foamPy\UEqn.H**

```
    volVectorField fluxPenalty("U_I",U);

    penalty.getUPenalty(fluxPenalty);

    fvVectorMatrix UEqn
    (
        fvm::ddt(voidfraction_Fluid*rho,U)
        + fvm::div(voidfractionf_Fluid*phiByVoidfraction,U)
        + particleCloud.divVoidfractionTau(U, voidfraction_Fluid)
        + rho*fluxPenalty
        ==
        - Fluid*fvm::Sp(Ksl,U)
        + massSource*U
    );

    if (runTime.write())
    {
        fluxPenalty.write();
    }
    UEqn.relax();

    if (momentumPredictor)
    {
        if (modelType=="B" || modelType=="Bfull")
            solve(UEqn == - fvc::grad(p) + Fluid*Ksl*Us );
        else
            solve(UEqn == - voidfraction_Fluid*fvc::grad(p) +
Fluid*Ksl*Us ); //
    }
```

## //Speicherort: ~\foamPy\YEqn.H

```
tmp<fv::convectionScheme<scalar> > mvConvection
(
    fv::convectionScheme<scalar>::New
    (
        mesh,
        fields,
        phi,
        mesh.schemesDict().divScheme("div(phi,Yi_h)")
    )
);

{
    label inertIndex = -1;
    volScalarField Yt("Yt", 0.0*Y[0]);
    volScalarField temp_Y("temp_Y", 0.0*Y[0]);

    for (label i=0; i<Y.size(); i++)
    {
        if ( Y[i].name() != "N2")
        {
            volScalarField& Yi = Y[i];
            //volScalarField LambdaTemp(penalty.Lambda()/dimensioned-
            Scalar("tempp", dimTime, 1.0));
            volScalarField N2_welle("N2_welle",-1*inertPenalty*Yi*rho*pen-
            alty.Lambda()/dimensionedScalar("temp", dimTime, 1.0));
            volScalarField massSource_temp(Yi.name()+"Source",Fluid*parti-
            cleCloud.forceM(1).getScalarField(Yi.name())*oneByVol*massCorrection);

            solve
            (
                fvm::ddt(voidfraction_Fluid*rho, Yi)
                + mvConvection->fvmDiv(voidfractionf_Fluid*phiByVoidfraction, Yi)
                - fvm::laplacian(turbulence->muEff()*voidfraction_Fluid, Yi)
                ==
                kappa*chemistry.RR(i)
                + massSource_temp
                + N2_welle

                mesh.solutionDict().solver("Yi")
            );

            Yi.max(0.0);
            Yt += Yi;
            temp_Y+=Yi;
            Info << "min(Yi("<< Yi.name() <<"))=" << gMin(massSource_temp.inter-
            nalField()) << ", max(Yi("<< Yi.name() <<"))=" << gMax(massSource_temp.inter-
            nalField()) << endl;

            if (runTime.write())
            {
                massSource_temp.write();
            }
        }
        else
        {
            inertIndex = i;
            temp_Y+=Y[i];
        }
    }
}
```

```

    }

    Info<< "void_fluid gas min/max   = " << gMin(voidfraction_Fluid.internal-
Field()) << ", "
        << gMax(voidfraction_Fluid.internalField()) << endl;

    Y[inertIndex] = scalar(1) - Yt;
    Y[inertIndex].max(0.0);

/*
   for (label i=0; i<Y.size(); i++)
   {
       Y[i]=Y[i]/temp_Y;
   }

   Yt /= temp_Y;
*/
}

```

```
//Speicherort: ~\foamPy\Make\files
pyFoam.C
```

```
EXE = $(FOAM_APPBIN)/foamPy
```

```
//Speicherort: ~\foamPy\Make\options
#include $(GENERAL_RULES)/mplib$(WM_MPLIB)
#include $(RULES)/mplib$(WM_MPLIB)
```

```
PFLAGS+= -DCFDEMWMPROJECTVERSION="$(CFDEM_WM_PROJECT_VERSION)"
PFLAGS+= -Dcompre
```

```
EXE_INC = \
  $(PFLAGS) \
  $(PINC) \
  $(CFDEM_ADD_COMPTURBMOD_PATHS) \
  -I$(LIB_SRC)/turbulenceModels/compressible/turbulenceModel \
  -I$(LIB_SRC)/thermophysicalModels/specie/lnInclude \
  -I$(LIB_SRC)/thermophysicalModels/reactionThermo/lnInclude \
  -I$(LIB_SRC)/thermophysicalModels/basic/lnInclude \
  -I$(LIB_SRC)/thermophysicalModels/chemistryModel/lnInclude \
  -I$(LIB_SRC)/ODE/lnInclude \
  -I$(LIB_SRC)/finiteVolume/lnInclude \
  -I$(LIB_SRC)/triSurface/lnInclude \
  -I$(LIB_SRC)/meshTools/lnInclude \
  -I$(CFDEM_SRC_DIR)/lagrangian/cfdemParticle/lnInclude \
  -I$(CFDEM_SRC_DIR)/lagrangian/cfdemParticle/cfdTools \
  -I$(CFDEM_SRC_DIR)/lagrangian/cfdemParticle/etc/addLibs_universal \
  -I./phaseExchange/lnInclude \
  -I$(CFDEM_LIGGGHTS_SRC_DIR) \
  -I$(LIB_SRC)/mesh/cfMesh/lnInclude \
  -I$(LIB_SRC)/sampling/lnInclude \
  -I./openPenalty/lnInclude
```

```
EXE_LIBS = \
  -L$(CFDEM_LIB_DIR)\
  $(CFDEM_ADD_COMPTURBMOD_LIBS) \
  -lmeshTools \
  -lcompressibleTurbulenceModel \
  -lcompressibleRASModels \
  -lcompressibleLESModels \
  -lreactionThermophysicalModels \
  -lspecie \
  -lbasicThermophysicalModels \
  -lchemistryModel \
  -lODE \
  -lfiniteVolume \
  -I$(CFDEM_LIB_COMP_NAME) \
  $(CFDEM_ADD_LIB_PATHS) \
  $(CFDEM_ADD_LIBS) \
  -llduSolvers \
  -lPenalty
```

## openPenalty (Brinkman -Strafverfahren)

//Speicherort: ~\foamPy\openPenalty\openPenalty.H

```
#ifndef openPenalty_H
#define openPenalty_H

#include "dictionary.H"
#include "coordinateSystem.H"
#include "coordinateSystems.H"
#include "wordList.H"
#include "labelList.H"
#include "dimensionedScalar.H"
#include "dimensionedTensor.H"
#include "primitiveFieldsFwd.H"
#include "volFieldsFwd.H"
#include "fvMatricesFwd.H"
#include "fvMesh.H"
#include "triSurfaceMesh.H"
#include <vector>
#include "obstacle.H"

namespace Foam
{

class openPenalty
{

private:

    enum {STATICBODY , ROTATINGBODY};

    //Basic members
    const fvMesh& mesh_;
    IOdictionary PenaltyDict_;

    //Immersed bodies objects
    std::vector< obstacle* > obstacles_; //Container for immersed bodies

    void getLambda();

    void getVoid();

    void getFlux();

    void reset();

    volScalarField *lambda_;

    volScalarField *voidfractionOld_;

    volScalarField *voidfractionNew_;

    volScalarField *rhoEqn_;

    volVectorField *Ui;

    scalar min_void_;

    scalar penalty_;

};

}

#endif
```

```

scalar T0_;

scalar thermalPenalty_;

public:

openPenalty(const Foam::fvMesh& mesh);
~openPenalty();

void update();

volScalarField Penalty();

volVectorField PenaltyFlux();

volScalarField ThermalPenalty();

volScalarField ThermalPenaltyFlux();

volVectorField PenaltyFluxP();

void voidfractionNew(volScalarField & voidfraction_);
void voidfractionUpdate(volScalarField & voidfraction_);

void fluxPenalty(volVectorField & U);

volScalarField voidfraction();

volScalarField Lambda();

volScalarField rhoEqn();

//geprüft auf Nützlichkeit
void getVoidfraction(volScalarField &voidfraction);
//Einheit(voidfraction) = -

void getUPenalty(volVectorField & UPenalty);
//Einheit(UPenalty(ein)) = m/s
//Einheit(UPenalty(aus)) = kg/(m^2*s^2)

void getDtVoidfraction(volScalarField &dtvoidfraction);
//Einheit(dtvoidfraction(ein)) = -
//Einheit(dtvoidfraction(aus)) = 1/s

void getTPenalty(volScalarField & TPenalty);
//Einheit(UPenalty(ein)) = m/s
//Einheit(UPenalty(aus)) = kg/(m^2*s^2)

};

}

#endif

```



**//Speicherort: ~\foamPy\openPenalty\openPenalty.C**

```
#include "openPenalty.H"
#include "fvMesh.H"
#include "polyMesh.H"
#include "fvCFD.H"
#include "fvMatrices.H"
#include "geometricOneField.H"
#include <cmath>
#include <algorithm>

#include "interpolationCellPoint.H"
#include "interpolationCell.H"

#define ORDER 2

using namespace Foam;

//-----//
openPenalty::openPenalty(const Foam::fvMesh& mesh)
:
mesh_(mesh),
PenaltyDict_
(
    IOobject
    (
        "PenaltyDict",
        "constant",
        mesh_,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    )
)
{
    lambda_ = new volScalarField
    (
        IOobject
        (
            "lambda",
            mesh_.time().timeName(),
            mesh_,
            IOobject::NO_READ,
            IOobject::AUTO_WRITE
        ),
        mesh_,
        dimensionedScalar("zero",dimless,0)
    );

    voidfractionOld_ = new volScalarField
    (
        IOobject
        (
            "voidfractionOld",
            mesh_.time().timeName(),
            mesh_,
            IOobject::NO_READ,
            IOobject::NO_WRITE
        ),
        mesh_,
        dimensionedScalar("zero",dimless,0)
    );
};
```

```

voidfractionNew_ = new volScalarField
(
    IOobject
    (
        "voidfractionNew",
        mesh_.time().timeName(),
        mesh_,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    mesh_,
    dimensionedScalar("zero",dimless,0)
);

rhoEqn_ = new volScalarField
(
    IOobject
    (
        "rhoEqn",
        mesh_.time().timeName(),
        mesh_,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    mesh_,
    dimensionedScalar("zero",dimless,0)
);

Ui = new volVectorField
(
    IOobject
    (
        "Ui",
        mesh_.time().timeName(),
        mesh_,
        IOobject::NO_READ,
        IOobject::NO_WRITE
    ),
    mesh_,
    dimensionSet(0,1,-1,0,0,0,0)
);

wordList stlNames( PenaltyDict_.lookup("stlNames") );
min_void_ = readScalar(PenaltyDict_.lookup("Voidfraction"));
penalty_ = readScalar(PenaltyDict_.lookup("Penalty"));
T0_ = readScalar(PenaltyDict_.lookup("T_soll"));
thermalPenalty_ = readScalar(PenaltyDict_.lookup("ThermalPenalty"));

//Generate obstacles
forAll(stlNames,name)
{
    obstacle * body_ptr = new obstacle(stlNames[name],mesh_,PenaltyDict_);
    obstacles_.push_back(body_ptr);
}

```

```

    getLambda();

    getVoid();

    getFlux();

}
//-----//
openPenalty::~openPenalty()
{
    for(unsigned int i=0;i<obstacles_.size();i++)
        delete obstacles_[i];

    obstacles_.clear();
}

void openPenalty::reset()
{
    forAll(Ui->internalField(),cellI)
    {
        Ui->internalField()[cellI]*=0.0;
        lambda_->internalField()[cellI]=0;
        voidfractionNew_->internalField()[cellI]=0;
    }
}

//-----//
void openPenalty::update()
{
    *voidfractionOld_=*voidfractionNew_;

    reset();

    min_void_ = readScalar(PenaltyDict_.lookup("Voidfraction"));
    penalty_ = readScalar(PenaltyDict_.lookup("Penalty"));

    for(unsigned int bodyId=0;bodyId<obstacles_.size();bodyId++)
        obstacles_[bodyId]->update();

    getLambda();

    getVoid();

    getFlux();
}

void openPenalty::voidfractionNew(volScalarField & voidfraction_)
{
    voidfraction_=*voidfractionNew_;

    return;
}

```

```

void openPenalty::voidfractionUpdate(volScalarField & voidfraction_)
{
    forAll(voidfraction_, cellI)
    {
        voidfraction_.internalField()[cellI]=std::min(voidfraction_.internal-
Field()[cellI],std::max(min_void_ ,1-lambda_->internalField()[cellI]));
    }
    return;
}

void openPenalty::fluxPenalty(volVectorField & U)
{
    dimensionedScalar P_("Penalty",dimensionSet(-1,3,1,0,0,0,0),penalty_);

    forAll(U,cellI)
    {
        U.internalField()[cellI]=lambda_->internalField()[cellI]*(U.internal-
Field()[cellI]-Ui->internalField()[cellI]);
    }

    if (U.dimensions()==dimensionSet(0,1,-1,0,0,0,0))
        U/=P_;
    else
        U/=penalty_;

    return;
}

volVectorField openPenalty::PenaltyFlux()
{
//    dimensionedScalar P_("Penalty",dimensionSet(0,0,1,0,0,0,0),penalty_);
    dimensionedScalar P_("Penalty",dimensionSet(1,-3,-1,0,0,0,0),1/penalty_);

    return *Ui*P_;
}

volVectorField openPenalty::PenaltyFluxP()
{
    return *Ui/penalty_;
}

volScalarField openPenalty::Penalty()
{
    dimensionedScalar P_("Penalty",dimensionSet(1,-3,-1,0,0,0,0),1/penalty_);

    return *lambda_*P_;
}

////////// Get Values //////////

void openPenalty::getLambda()
{
    for(unsigned int bodyId=0;bodyId<obstacles_.size();bodyId++)
        obstacles_[bodyId]->updateLambda(*lambda_);
}

```

```

lambda_->correctBoundaryConditions();
}

void openPenalty::getFlux()
{
    for(unsigned int bodyId=0;bodyId<obstacles_.size();bodyId++)
        obstacles_[bodyId]->updateFlux(*Ui);

    Ui->correctBoundaryConditions();
}

void openPenalty::getVoid()
{
    forAll(*voidfractionNew_, cellI)
    {
        voidfractionNew_->internalField()[cellI]=std::min(1.0, std::max(min_void_
,1-lambda_->internalField()[cellI]));
    }

    voidfractionNew_->correctBoundaryConditions();

    return;
}

////////// give Values //////////
void openPenalty::getVoidfraction(volScalarField &voidfraction)
{
    forAll(voidfraction, cellI)
    {
        voidfraction.internalField()[cellI]=std::min(voidfractionNew_->internal-
Field()[cellI]+voidfraction.internalField()[cellI], 1.0);
    }

    voidfraction.correctBoundaryConditions();

    return;
}

void openPenalty::getUPenalty(volVectorField &UPenalty)
{
    dimensionedScalar P_("Penalty", dimensionSet(0,0,1,0,0,0,0), penalty_);

    forAll(UPenalty, cellI)
    {
        UPenalty.internalField()[cellI]=lambda_->internalField()[cellI]*(UPen-
alty.internalField()[cellI]-Ui->internalField()[cellI]);
    }

    UPenalty/=P_;

    UPenalty.correctBoundaryConditions();

    return;
}

/*
void openPenalty::getTPenalty(volScalarField &TPenalty)
{

```

```

    dimensionedScalar TP_("Penalty",dimensionSet(-1,3,1,0,0,0,0),thermalPen-
alty_);

    forAll(TPenalty,cellI)
    {
        TPenalty.internalField()[cellI]=lambda_->internalField()[cellI]*(TPen-
alty.internalField()[cellI]-T_soll_);
    }

    TPenalty/=TP_;

    return;
}
*/

void openPenalty::getDtVoidfraction(volScalarField &dtvoidfraction)
{
    dimensionedScalar dt_("Penalty",dimension-
Set(0,0,1,0,0,0,0),mesh_.time().deltaT().value());

    forAll(dtvoidfraction,cellI)
    {
        dtvoidfraction.internalField()[cellI]=voidfractionOld_->internal-
Field()[cellI]-voidfractionNew_->internalField()[cellI];
    }

    dtvoidfraction/=dt_;

    dtvoidfraction.correctBoundaryConditions();

    return;
}

volScalarField openPenalty::voidfraction()
{
    return *voidfractionNew_;
}

volScalarField openPenalty::Lambda()
{
    return *lambda_;
}

volScalarField openPenalty::rhoEqn()
{
    forAll(*rhoEqn_,cellI)
    {
        rhoEqn_->internalField()[cellI]=(1+(1/voidfractionNew_->internal-
Field()[cellI]-1))*lambda_->internalField()[cellI];
    }
    return *rhoEqn_;
}

volScalarField openPenalty::ThermalPenaltyFlux()
{
    dimensionedScalar P_("Penalty",dimensionSet(1,-3,-1,1,0,0,0),1/thermalPen-
alty_);

    return *lambda_*T0_*P_;
}

```

```
}  
  
volScalarField openPenalty::ThermalPenalty()  
{  
    dimensionedScalar P_("Penalty",dimensionSet( 1, -3,-1,0,0,0,0),1/thermalPen-  
alty_);  
    return *lambda_*P_;  
}
```

```
//Speicherort: ~\foamPy\openPenalty\obstacle.H
```

```
#ifndef obstacle_H
#define obstacle_H

#include "dictionary.H"
#include "coordinateSystem.H"
#include "coordinateSystems.H"
#include "wordList.H"
#include "dimensionedScalar.H"
#include "dimensionedTensor.H"
#include "primitiveFieldsFwd.H"
#include "volFieldsFwd.H"
#include "fvMatricesFwd.H"
#include "fvMesh.H"
#include "triSurfaceMesh.H"
#include <vector>

#include "triSurfaceMesh.H"
#include "triSurfaceTools.H"
#include "triSurfaceSearch.H"

#include "DynList.H"

namespace Foam
{
    enum {STATICBODY , ROTATINGBODY};

    class obstacle
    {
    private:
        dictionary                obstacleDict_;

        //Operation to perform
        int                        bodyOperation_;

        const fvMesh&             mesh_;

        //base geometry
        triSurfaceMesh*          bodySurfMesh_;

        //surface cells
        std::vector< label >     surfCells_;

        //internal cells
        std::vector< label >     intCells_;

        volScalarField *         body_;

        volVectorField *         U_obstacle;

        //Create obstacle info
        void createObstacle();
        void updateObstacleSurface();

        point rotateVector(point old_point, scalar t);

        vector axis;

        point origin;
    };
};
```



```

scalar omega;

int Index[3];

//Reset body field
void resetObstacle();

void setObstaclePosition();

void rotateObstacle();

void createObstacleVelocityField();

//Refine body field using MC-like integration
void refineBody(triSurfaceSearch * ibTriSurfSearch, const pointField * pp);

public:

obstacle(word fileName, const Foam::fvMesh& mesh, dictionary PenaltyDict_);
~obstacle();

//Getters

const std::vector< label > * getSurfaceCellList() const {return &surf-
Cells_};

//Update obstacle info
void update();

//Update body field
void updateLambda(volScalarField & lambda);

//Update imposed vector field
void updateFlux(volVectorField & Ui);

};
}
#endif

```

**//Speicherort: ~\foamPy\openPenalty\obstacle.C**

```
#include "obstacle.H"
#include "fvMesh.H"
#include "polyMesh.H"
#include "fvCFD.H"
#include "fvMatrices.H"
#include "geometricOneField.H"
#include <cmath>
#include <math.h>
#include <algorithm>

#include "interpolationCellPoint.H"
#include "interpolationCell.H"
#include "meshSearch.H"
#include "List.H"

#include <ctime>

#define SMAL 1e-12

#define M_PI 3.14159265358979323846

using namespace Foam;

//-----//
obstacle::obstacle(word fileName, const Foam::fvMesh& mesh, dictionary PenaltyDict_ )
:
obstacleDict_(PenaltyDict_.subDict(fileName)),
mesh_(mesh)
{
    U_obstacle = new volVectorField
    (
        IOobject
        (
            "U_"+fileName,
            mesh_.time().timeName(),
            mesh_,
            IOobject::NO_READ,
            IOobject::NO_WRITE //IOobject::AUTO_WRITE
        ),
        mesh.C()*dimensionedScalar("zero", dimensionSet(0,1,-1,0,0,0,0), 0)
    );
    body_ = new volScalarField
    (
        IOobject
        (
            "body_"+fileName,
            mesh_.time().timeName(),
            mesh_,
            IOobject::NO_READ,
            IOobject::NO_WRITE //IOobject::NO_WRITE
        ),
        mesh_,
        dimensionedScalar("zero", dimless, 0)
    );
    //Read stl file from folder "constant"
    bodySurfMesh_ = new triSurfaceMesh
    (
        IOobject
```

```

    (
        fileName + ".stl",
        "constant",
        "triSurface",
        mesh_,
        IOobject::MUST_READ,
        IOobject::NO_WRITE //IOobject::AUTO_WRITE
    )
);

for (int i=0; i<3; i++)
{
    Index[i]=0;
}

Info<< "Read obstacle triSurface: " << fileName + ".stl" << endl;
bodySurfMesh_>writeStats(Info);

Info << "Start object" << endl;
//Return if declared static
if(obstacleDict_.found("staticBody") )           bodyOperation_ = STATICBODY;
else if(obstacleDict_.found("rotatingBody"))
{
    bodyOperation_ = ROTATINGBODY;
    axis   = obstacleDict_.subDict("rotatingBody").lookup("axis");
    origin = obstacleDict_.subDict("rotatingBody").lookup("origin");
    omega  = readScalar(obstacleDict_.subDict("rotatingBody").lookup("rpm")) * 2 * 3.14159265359 / 60;

    int j=0;
    int save=0;
    for (int i=0; i<3; i++)
    {
        if (axis.component(i)==0)
        {
            Index[j]=i;
            j=j+1;
        }
        else
        {
            save=i;
        }
    }

    Index[2]=save;

    createObstacleVelocityField();

    setObstaclePosition();
}
else
{
    Info << "No body operation was found for " << fileName << ". Assuming static
body.";
    bodyOperation_ = STATICBODY;
}

```

```

// U_obstacle->write();

createObstacle();

}

//-----//
obstacle::~obstacle()
{
delete bodySurfMesh_;
delete U_obstacle;
delete body_;
}

void obstacle::createObstacleVelocityField()
{

forAll(mesh_.C(), cellI)
{
U_obstacle->internalField()[cellI]=(rotateVector(mesh_.C()[cellI],SMAL)-
mesh_.C()[cellI])/SMAL;
}

}

point obstacle::rotateVector(point old_point, scalar t)
{

point newPositon(0,0,0);

vector normV=old_point-origin;

scalar r=std::sqrt(normV.component(Index[0])*normV.component(Index[0])
+normV.component(Index[1])*normV.component(In-
dex[1]));

scalar gama=atan2(normV.component(Index[0]),normV.component(Index[1]));

newPositon.component(Index[0])=sin(gama+omega*t)*r+origin.component(In-
dex[0]);
newPositon.component(Index[1])=cos(gama+omega*t)*r+origin.component(In-
dex[1]);
newPositon.component(Index[2])=old_point.component(Index[2]);

return newPositon;
}

//-----//
//Update obstcle
void obstacle::update()
{

if(bodyOperation_==STATICBODY)
{

```

```

        return;
    }
    else if(bodyOperation_==ROTATINGBODY)
    {
        Info << "openPenalty: Updating " << endl;

        resetObstacle();
        //updateObstacleSurface();
        //TODO:Very inefficient find other algorithm
        updateObstacleSurface();
    }

    //createObstacle();
}

void obstacle::resetObstacle()
{
    if (bodyOperation_==ROTATINGBODY)
        rotateObstacle();
}

void obstacle::rotateObstacle()
{
    pointField bodyPoints (bodySurfMesh_->points());

    //Move points
    forAll(bodyPoints,p)
    {
        bodyPoints[p]=rotateVector(bodyPoints[p],mesh_.time().deltaT().value());
    }

    //move mesh
    bodySurfMesh_->movePoints(bodyPoints);
}

void obstacle::setObstaclePosition()
{
    pointField bodyPoints (bodySurfMesh_->points());

    //Move points
    forAll(bodyPoints,p)
    {
        bodyPoints[p]=rotateVector(bodyPoints[p],mesh_.time().value());
    }

    //move mesh
    bodySurfMesh_->movePoints(bodyPoints);
}

//-----//
//Create obstacle info

```

```

void obstacle::createObstacle ()
{
    triSurface ibTemp( *bodySurfMesh_);
    triSurfaceSearch ibTriSurfSearch( ibTemp );
    const pointField & pp = mesh_.points();

    intCells_.clear();
    surfCells_.clear();

    boolList vertexesInside = ibTriSurfSearch.calcInside( pp );

    //Fill body_ field with first estimation
    forAll(mesh_.C(), cellI)
    {

        //Check if partially or completely inside
        const labelList& vertexLabels = mesh_.cellPoints()[cellI];

        //fraction of cell covered
        forAll(vertexLabels, verIn)
        {
            body_>internalField()[cellI] += vertexesInside[vertexLabels[verIn]] ;
        }

        body_>internalField()[cellI]/=vertexLabels.size();

        //Add to corresponding vector
        if( body_>internalField()[cellI]==1)
        {
            intCells_.push_back(cellI);
        }
        else if (body_>internalField()[cellI]>0)
        {
            surfCells_.push_back(cellI);
        }
    }

    refineBody(&ibTriSurfSearch, & pp);
}

void obstacle::updateObstacleSurface ()
{
    //Initialisation
    triSurface ibTemp( *bodySurfMesh_);
    triSurfaceSearch ibTriSurfSearch( ibTemp );
    const pointField & pp = mesh_.points();

    //möglicher Fehler feldgröße=body_.size() Werte sind 0
    //scalarField boundary(body_>internalField().size(),0);

    volScalarField boundary("ObstacleSurface",*body_*0.0 );
    //boundary*=0;
}

```

```

//generiere Oberflächenzellen + nachbarn von Oberflächenzellen + nachbarn
von nachbarn von Oberflächenzellen
for (unsigned int surfaceCellI=0; surfaceCellI<surfCells_.size(); surface-
CellI++)
{
    boundary.internalField()[surfCells_[surfaceCellI]]=1;

    for (unsigned int neighborSurfaceCellI=0;
        neighborSurfaceCellI<mesh_.cellCells()[surfCells_[surface-
CellI]].size();
        neighborSurfaceCellI++)
    {
        boundary.internalField()[mesh_.cellCells()[surfCells_[surface-
CellI]][neighborSurfaceCellI]]=1;

        for(unsigned int neighborSurfaceCell2I=0;
            neighborSurfaceCell2I<mesh_.cellCells()[mesh_.cell-
Cells()[surfCells_[surfaceCellI]][neighborSurfaceCellI]].size();
            neighborSurfaceCell2I++)
        {
            boundary.internalField()[mesh_.cellCells()[mesh_.cell-
Cells()[surfCells_[surfaceCellI]][neighborSurfaceCellI]][neighborSurface-
Cell2I]]+=0.5;
        }
    }
}

boundary.correctBoundaryConditions();

forAll ( mesh_.boundaryMesh() , ipatch )
{
    word BCtype = mesh_.boundaryMesh().types()[ipatch];

    if( BCtype == "processor" )
    {
        const UList<label> &bfaceCells = mesh_.boundaryMesh()[ipatch].face-
Cells();

        scalarField exchange = boundary.boundaryField()[ipatch].patchNeigh-
bourField();

        forAll( bfaceCells , icell )
        {
            boundary[ bfaceCells[icell] ] += exchange[icell];
        }
    }
}

surfCells_.clear();

//generiere neue (Oberflächen+eps)Zellen
std::vector< label > surfaceArea;
surfaceArea.clear();

```

```

forAll (boundary, cellI)
{
    if (boundary.internalField() [cellI]>=1.0)
        surfaceArea.push_back (cellI);
}

//Bestimme die Eckpunkte, die die (Oberflächen+eps)Zellen beinhalten
//->Mögliche fehlerquelle falsche Initialisierung
scalarField corners (mesh_.points ().size (), 0);

for (unsigned int cellI=0; cellI<surfaceArea.size (); cellI++)
{
    const labelList& vertexLabels = mesh_.cellPoints () [surfaceArea[cellI]];

    for (unsigned int vertexI=0; vertexI<vertexLabels.size (); vertexI++)
    {
        corners[vertexLabels[vertexI]]=1;
    }
}

DynList<point> surfaceCornerPoints;
DynList<label> surfaceCornerIndex;

surfaceCornerPoints.clear ();
surfaceCornerIndex.clear ();

forAll (corners, cellI)
{
    if (corners[cellI]==1)
    {
        surfaceCornerPoints.append (mesh_.points () [cellI]);
        surfaceCornerIndex.append (cellI);
    }
}

//Überprüfe pField.size()
pointField pField (surfaceCornerPoints.size (), vector::zero);

forAll (pField, cellI)
{
    pField[cellI]=surfaceCornerPoints[cellI];
}

//Suche die Eckzellen ab und gib ihnen den Wert 1/8
boolList pInside = ibTriSurfSearch.calcInside ( pField );

for (unsigned int cellI=0; cellI<surfaceCornerIndex.size (); cellI++)
{
    corners[surfaceCornerIndex[cellI]]=pInside[cellI];
}

//Update body_ und generiere aus body liste mit inneren und rand Zellen
for (unsigned int cellI=0; cellI<surfaceArea.size (); cellI++)
{
    const labelList& vertexLabels = mesh_.cellPoints () [surfaceArea[cellI]];
    body_ ->internalField () [surfaceArea[cellI]] = 0;
}

```



```

        for(unsigned int vertexI=0; vertexI<vertexLabels.size(); vertexI++)
        {
            body_>internalField()[surfaceArea[cellI]] += corners[vertexLabels[vertexI]] ;
        }
        body_>internalField()[surfaceArea[cellI]]/=vertexLabels.size();
    }

    forAll(body_>internalField(),cellI)
    {
        if(body_>internalField()[cellI]==1.0)
        {
            intCells_.push_back(cellI);
        }
        else if(body_>internalField()[cellI]>0.0)
        {
            surfCells_.push_back(cellI);
        }
    }

    //verfeinere die randzellen
    refineBody(&ibTriSurfSearch, & pp);

}

//-----//
//Refine body field for this immersed object using MC-like algorithm
//Cells are assumed to be hexahedral at the particle surface (but can have different edge length)
void obstacle::refineBody(triSurfaceSearch * ibTriSurfSearch, const pointField * pp )
{
    if(!obstacleDict_.found("refineMC"))
        return;

    scalar nPointsEdge = readScalar(obstacleDict_.lookup("refineMC"));

    //loop over all the surface cells
    for(unsigned int cell=0;cell<surfCells_.size();cell++)
    {
        label cellI = surfCells_[cell];

        scalar deltaV = 1.0/((nPointsEdge)*(nPointsEdge)*(nPointsEdge));

        //Get cell center
        point centerC = mesh_.C()[cellI];

        //Get one node
        //Check if partially or completely inside
        const labelList& vertexLabels = mesh_.cellPoints()[cellI];
        const pointField vertexPoints(*pp,vertexLabels);
        point baseNode = vertexPoints[0];

        //create vector representing 3d diagonal of the cell
        vector edgesC = 2*(centerC - baseNode);

```

```

//create list of points
DynList<point> pointsMC;

//create deltas
scalar delta_i = edgesC[0]/nPointsEdge;
scalar delta_j = edgesC[1]/nPointsEdge;
scalar delta_k = edgesC[2]/nPointsEdge;

//add points to list
for(int i=0;i<nPointsEdge;i++)
{
    //point i-coordinate
    scalar icoord = baseNode[0] + delta_i*(i+0.5);

    for(int j=0;j<nPointsEdge;j++)
    {
        //point j-coordinate
        scalar jcoord = baseNode[1] + delta_j*(j+0.5);

        for(int k=0;k<nPointsEdge;k++)
        {
            //point k-coordinate
            scalar kcoord = baseNode[2] + delta_k*(k+0.5);

            //create point
            point p(icoord,jcoord,kcoord);

            //add to list
            pointsMC.append(p);

        }
    }
}

//Check who is inside
pointField pField(pointsMC.size(),vector::zero);

//Info << "pointsMC.size()=" << pointsMC.size() << " pField.size()="<<
pField.size() << endl;

for (unsigned int cellI=0; cellI<pointsMC.size(); cellI++)
{
    pField[cellI]=pointsMC[cellI];
}

boolList pInside = ibTriSurfSearch->calcInside( pField );

//Calculate new body
scalar newbody = 0.0;

forAll(pField,p)
{
    if(pInside[p])
    {
        newbody+=deltaV;
    }
}

body_->internalField()[cellI] = newbody;

```

```

    }
}
void obstacle::updateLambda(volScalarField & lambda)
{
    forAll(lambda, cellI)
    {
        lambda.internalField()[cellI] += body_->internalField()[cellI];
    }
    return;
}
void obstacle::updateFlux(volVectorField & Ui)
{
    forAll(Ui, cellI)
    {
        Ui.internalField()[cellI] += body_->internalField()[cellI] * U_obstacle->in-
        ternalField()[cellI];
    }
    return;
}

```

```
//Speicherort: ~\foamPy\openPenalty\Make\files
obstacle.C
openPenalty.C
```

```
LIB = $(FOAM_USER_LIBBIN)/libPenalty
```

```
//Speicherort: ~\foamPy\openPenalty\Make\options
```

```
EXE_INC = \  
-I$(LIB_SRC)/triSurface/InInclude \  
-I$(LIB_SRC)/meshTools/InInclude \  
-I$(LIB_SRC)/finiteVolume/InInclude \  
-I$(LIB_SRC)/lagrangian/basic/InInclude \  
-I$(LIB_SRC)/mesh/blockMesh/InInclude \  
-I$(LIB_SRC)/mesh/cfMesh/InInclude \  
-I$(MPI_ARCH_PATH)/include \  

```

```
LIB_LIBS = \  
-lmeshTools \  
-llagrangian \  
-lblockMesh #\  
# -L$(MPI_ARCH_PATH)/lib64 \  
# -lmpi_cxx \  

```

# LIGGGHTS

## fix\_cfd\_coupling\_convection\_reaction

//Speicherort: ~\LIGGGHTS\_PUBLIC\src\fix\_cfd\_coupling\_convection\_reaction.h

```
#ifndef FIX_CLASS

FixStyle(couple/cfd/reaction,FixCfdCouplingConvectionReaction)

#else

#ifndef LMP_FIX_CFD_COUPLING_CONVECTION_REACTION_H
#define LMP_FIX_CFD_COUPLING_CONVECTION_REACTION_H

#include "fix_cfd_coupling.h"

namespace LAMMPS_NS {

class FixCfdCouplingConvectionReaction : public Fix {

public:
    FixCfdCouplingConvectionReaction(class LAMMPS *, int, char **);
    ~FixCfdCouplingConvectionReaction();
    void post_create();
    void pre_delete(bool unfixflag);

    virtual int setmask();
    virtual void init();
    virtual void post_force(int);
    void setup_pre_force(int);
    void pre_force(int);

    void delate_atoms();

    void change_settings();

protected:
    class FixCfdCoupling* fix_coupling;

    double species0;
    char speciesName_[128];
    char sourceName_[128];
    char convectiveFluxName_[128];
    char capacityName_[128];
    char steName_[128];
    char totalFluxName_[128];
    char speciesFluidName_[128];
    char speciesTransCoeffName_[128];

    int *dlist;

    double r_min;

    int atom_type;

    double integrate_fraction(double, double, double, double, double);
    double integrate_fraction(double, double, double, double, double, double);

    double heun(double, double, double, double, double);


```

```

double euler(double, double, double);
double arrhenius(double, double, double);

double getConvertArea(double, double);

class FixPropertyAtom* fix_composition[6];
class FixPropertyAtom* fix_QConversion;

class FixPropertyAtom* fix_flux;

class FixPropertyAtom* fix_cpLost;
class FixPropertyAtom* fix_ConversionFlux;
class FixPropertyGlobal* fix_conductivity_;

class FixPropertyAtom* fix_partFlux[6];
class FixPropertyAtom* fix_density;

class FixPropertyAtom* fix_temperature;

class FixPropertyAtom* fix_convectiveFlux;
class FixPropertyAtom* fix_heatFlux;
double T0;
//class FixPropertyGlobal* fix_cp;

bool delate_flag;

char speciesName[6][128];
double mp[6];
double rho[6];
double cp[6];
double A[6];
double EA[6];
double HR[6];
double lambda_;

// double SF[6];

int N_components;
};
}

#endif
#endif

```

```
//Speicherort: ~\LIGGGHTS_PUBLIC\src\fix_cfd_coupling_convection_reaction.cpp
```

```
#include <string.h>
#include <stdlib.h>
#include "atom.h"
#include "atom_vec.h"
#include "update.h"
#include "respa.h"
#include "error.h"
#include "neighbor.h"
#include "memory.h"
#include "modify.h"
#include "group.h"
#include "comm.h"
#include <cmath>
#include "vector_liggghts.h"
#include "fix_cfd_coupling_convection_reaction.h"
#include "fix_property_atom.h"
#include "fix_property_global.h"
#include "properties.h"

#include <iostream>
#include <sstream>
#include <stdio.h>
#include <iomanip>
#include <ctime>
#include <complex>

#define SMAL 1e-12
#define fuckingSMAL 1e-32

#define Aktiviate

#define PI 3.14159265359

#ifdef Aktiviate
    #include "impEuler.h"
#endif

using namespace std;
using namespace LAMMPS_NS;
using namespace FixConst;

/* ----- */
FixCfdCouplingConvectionReaction::FixCfdCouplingConvectionReaction(LAMMPS *lmp,
int narg, char **arg) : Fix(lmp, narg, arg)
{
    fix_coupling = NULL;
    //fix_temperature = NULL;
    fix_density = NULL;
    fix_flux = NULL;
    fix_cpLost = NULL;
    fix_QConversion = NULL;
    fix_ConversionFlux = NULL;

    int iarg = 3;
    N_components=atoi(arg[iarg++]);

    if (N_components>6)
        error->all(FLERR,"Fix couple/cfd/reactons: Not more than 6 components
supported");
}
```

```

double test_mp=0;

if (narg>=(iarg+N_components))
{
    for (int i=0; i<N_components; i++)
    {
        strcpy(speciesName[i],arg[iarg++]);
        const double* values;
        values=static_cast<FixPropertyGlobal*>(modify->find_fix_prop-
erty(speciesName[i],"property/global","scalar",6,0,style))->get_values();
        mp[i]=values[0];
        cp[i]=values[1];
        rho[i]=values[2];
        A[i]=values[3];
        EA[i]=values[4];
        HR[i]=values[5];
        // SF[i]=values[6];
        test_mp+=mp[i];

        fix_composition[i]= NULL;
        fix_partFlux[i]= NULL;

        cout << endl << "\t " << speciesName[i] << endl
            << "\t mp=" << mp[i] << endl
            << "\t cp=" << cp[i] << endl
            << "\t rho=" << rho[i] << endl
            << "\t A=" << A[i] << endl
            << "\t EA=" << EA[i] << endl
            << "\t HR=" << HR[i] << endl;
    }
}
else
{
    error->all(FLERR,"Fix couple/cfd/reactons: Not enough reaction compo-
nents");
}
cout<< endl;

if (test_mp!=1.0)
{
    cout << "Summ of componentfractions is "<< test_mp << ", but should be
1.0"<< endl;
    error->all(FLERR,"Fix couple/cfd/reactons: Summ of Masspercent (mp) is
not 1.0");
}

r_min=1.0;
atom_type=0;

if (narg>(iarg+3))
{
    if (strcmp(arg[iarg++],"type") != 0)
        error->all(FLERR,"Fix couple/cfd/reactons: On 4. last position should
be the type ");

    atom_type=atoi(arg[iarg++]);

    if (strcmp(arg[iarg++],"delate") != 0)
        error->all(FLERR,"Fix couple/cfd/reactons: On 2. last position should
be the minimum radius ");
}

```



```

        delate_flag=1;
        r_min=atof(arg[iarg++]);
    }

    cout << "atom_type=" << atom_type << endl << "r_min=" << r_min << endl;
}

/* ----- */
FixCfdCouplingConvectionReaction::~FixCfdCouplingConvectionReaction()
{
}

/* ----- */

void FixCfdCouplingConvectionReaction::pre_delete(bool unfixflag)
{
}

/* ----- */

int FixCfdCouplingConvectionReaction::setmask()
{
    int mask = 0;
    mask |= POST_FORCE;
    mask |= PRE_FORCE;
    return mask;
}

/* ----- */

void FixCfdCouplingConvectionReaction::post_create()
{
    for (int i=0; i<N_components; i++)
    {

        // register species concentration
        if(!fix_composition[i])
        {

            const char* fixarg[9];
            //stringstream ss;

            //ss << speciesName[i] << "_mp";

            char compositionName_[128];
            sprintf(compositionName_, "%s_mp", speciesName[i]);

            char compositionMP_[128];
            sprintf(compositionMP_, "%f", mp[i]);

            fixarg[0]=compositionName_; //ss.str().c_str();
            fixarg[1]="all";
            fixarg[2]="property/atom";
            fixarg[3]=compositionName_;//ss.str().c_str();
            fixarg[4]="scalar";
            fixarg[5]="yes";

```

```

        fixarg[6]="yes";
        fixarg[7]="yes";

        //stringstream ss2;
        //ss2 << mp[i];

        fixarg[8]=compositionMP_;//ss2.str().c_str();

        fix_composition[i] = modify->add_fix_prop-
erty_atom(9,const_cast<char**>(fixarg),style);
    }

    // register species concentration
    if(!fix_partFlux[i])
    {
        const char* fixarg[9];

        stringstream ss;

        ss << speciesName[i] << "Flux";

        char fluxName_[128];
        sprintf(fluxName_, "%sFlux",speciesName[i]);

        fixarg[0]=fluxName_;//ss.str().c_str();
        fixarg[1]="all";
        fixarg[2]="property/atom";
        fixarg[3]=fluxName_;//ss.str().c_str();
        fixarg[4]="scalar";
        fixarg[5]="no";
        fixarg[6]="yes";
        fixarg[7]="no";
        fixarg[8]="0";

        fix_partFlux[i] = modify->add_fix_prop-
erty_atom(9,const_cast<char**>(fixarg),style);
    }

}

if(!fix_flux)
{
    const char* fixarg[9];

    fixarg[0]="totalFlux";
    fixarg[1]="all";
    fixarg[2]="property/atom";
    fixarg[3]="totalFlux";
    fixarg[4]="scalar";
    fixarg[5]="no";
    fixarg[6]="yes";
    fixarg[7]="no";
    fixarg[8]="0";
    fix_flux = modify->add_fix_property_atom(9,const_cast<char**>(fix-
arg),style);
}

if(!fix_cpLost)
{
    const char* fixarg[9];

```

```

    fixarg[0]="cp_lost";
    fixarg[1]="all";
    fixarg[2]="property/atom";
    fixarg[3]="cp_lost";
    fixarg[4]="scalar";
    fixarg[5]="no";
    fixarg[6]="yes";
    fixarg[7]="no";
    fixarg[8]="0";
    fix_cpLost = modify->add_fix_property_atom(9,const_cast<char**>(fix-
arg),style);
}

if(!fix_density)
{
    const char* fixarg[9];

    fixarg[0]="density";
    fixarg[1]="all";
    fixarg[2]="property/atom";
    fixarg[3]="density";
    fixarg[4]="scalar";
    fixarg[5]="yes";
    fixarg[6]="yes";
    fixarg[7]="yes";
    fixarg[8]="0";
    fix_density = modify->add_fix_property_atom(9,const_cast<char**>(fix-
arg),style);
}

if(!fix_ConversionFlux)
{
    const char* fixarg[9];

    fixarg[0]="ConversionFlux";
    fixarg[1]="all";
    fixarg[2]="property/atom";
    fixarg[3]="ConversionFlux";
    fixarg[4]="scalar";
    fixarg[5]="no";
    fixarg[6]="yes";
    fixarg[7]="no";
    fixarg[8]="0";
    fix_ConversionFlux = modify->add_fix_prop-
erty_atom(9,const_cast<char**>(fixarg),style);
}

if(!fix_QConversion)
{
    const char* fixarg[9];

    fixarg[0]="QConversion";
    fixarg[1]="all";
    fixarg[2]="property/atom";
    fixarg[3]="Tconversion";
    fixarg[4]="scalar";
    fixarg[5]="no";
    fixarg[6]="yes";
    fixarg[7]="no";

```

```

        fixarg[8]="0";
        fix_QConversion = modify->add_fix_prop-
erty_atom(9, const_cast<char**>(fixarg), style);
    }

}

/* ----- */

void FixCfdCouplingConvectionReaction::init()
{
    // make sure there is only one fix of this style
    //if(modify->n_fixes_style(style) != 1)
    // error->fix_error(FLERR,this,"More than one fix of this style is not al-
lowed");

    // find coupling fix
    fix_coupling = static_cast<FixCfdCoupling*>(modify-
>find_fix_style_strict("couple/cfd",0));

    if(!fix_coupling)
        error->fix_error(FLERR,this,"needs a fix of type couple/cfd");

    fix_coupling->add_push_property("totalFlux","scalar-atom");

    for (int i=0; i<N_components; i++)
    {
        char name[30];

        cout << speciesName[i] << endl;
        sprintf(name,"%sFlux",speciesName[i]);

        fix_coupling->add_push_property(name,"scalar-atom");
    }

    fix_coupling->add_push_property("Tconversion","scalar-atom");

    fix_temperature = static_cast<FixPropertyAtom*>(modify->find_fix_prop-
erty("Temp","property/atom","scalar",0,0,style));
    // fix_cp = static_cast<FixPropertyGlobal*>(modify->find_fix_property("ther-
malCapacity","property/global","peratomtype",max_type,0,style));
    //static_cast<FixPropertyAtom*>(modify->find_fix_property("thermal-
Capacity","property/atom","scalar",0,0,style));

    int max_type = atom->get_properties()->max_type();
    fix_conductivity_ =
        static_cast<FixPropertyGlobal*>(modify->find_fix_property("thermalConduc-
tivity","property/global","peratomtype",max_type,0,style));

    //lambda_ = 4.*fix_conductivity_->compute_vector(atom_type-1)*fix_conductiv-
ity_->compute_vector(atom_type-1)/(fix_conductivity_->compute_vector(atom_type-
1)+fix_conductivity_->compute_vector(atom_type-1))/4.8545e-06;
}

```

```

void FixCfdCouplingConvectionReaction::setup_pre_force(int vflag)
{
    // change_settings();
}

void FixCfdCouplingConvectionReaction::pre_force(int vflag)
{
    change_settings();
}

void FixCfdCouplingConvectionReaction::change_settings()
{
    modify->clearstep_compute();

    double *radius      = atom->radius;
    double *rmass       = atom->rmass;
    double *rhoP        = fix_density->vector_atom;
    int *mask           = atom->mask;
    int *type           = atom->type;
    int nlocal          = atom->nlocal;
    double *Temp        = fix_temperature->vector_atom;
    double *QConversion = fix_QConversion->vector_atom;
    double *flux        = fix_flux->vector_atom;
    double *cpLost      = fix_cpLost->vector_atom;

#ifdef Aktivat
    impEuler integrator;
#endif
    double h=update->dt;
    double V0;

    //get massloss for every component
    for (unsigned int i=0; i<nlocal; i++)
    {
        if (Temp[i]!=Temp[i])
        {
            cout << "Temp["<<i<<"]:= -nan resetting tempeature to "<< 293.15 <<
endl;
            Temp[i]      = 293.15;
        }
        if ((type[i]==atom_type))
        {

            V0=0;

            double fm_save = 0;
            double fm = 0;
#ifdef Aktivat
            //double fm2 = 0;
#endif
            double mass_org = rmass[i];

            flux[i] = 0;

            for (int j=0; j<N_components; j++)

```

```

    {

        fm_save = fix_composition[j]->vector_atom[i]*mass_org;
//Bestimme aktueller Massenteil von i
        //fm = integrate_fraction(fm_save,h,A[j],EA[j],Temp[i]);
//Bestimme den neuen Massenteil von i

#ifdef(Aktivate)
        fm= integrator.get_Int(A[j],EA[j],Temp[i],fm_save,h);
#endif

        if (fm<SMAL) fm=0;

        fix_partFlux[j]->vector_atom[i] = (fm_save-fm);
//Bestimme den Massenverlust von i
        rmass[i] -= fix_partFlux[j]->vector_atom[i];
//Bestimme die neue Masse durch entfernen der Massenverluste
        flux[i] += fix_partFlux[j]->vector_atom[i];
        V0+=fix_partFlux[j]->vector_atom[i]/rho[j];

    }

    //fix_ConversionFlux->vector_atom[i] = lambda_*(getConvertArea(radius[i],V0)); //keine Ahnung was ich da gemacht habe. Nullstellensuche eines
    Polynoms 3. Ordnung ...

}

}

//update metadata
for (int i=0; i<nlocal; i++)
{

    if ((type[i]==atom_type))
    {
        rhoP[i] = 0;
        cpLost[i] = 0;

        double cpLostFraction=0;
        double H_Rm=0;

        double pm_org=flux[i]+rmass[i];

//Update der Partikel Daten (Verbliebene Partikel)
        for (int j=0; j<N_components; j++)
        {
            double mfraction= fix_composition[j]->vector_atom[i] *pm_org
                -fix_partFlux[j]->vector_atom[i];

            fix_composition[j]->vector_atom[i]=mfraction/rmass[i];

            cpLost[i] += cp[j]*fix_composition[j]->vector_atom[i];

            rhoP[i] += rho[j]*fix_composition[j]->vector_atom[i];

            cpLostFraction += cp[j]*fix_partFlux[j]->vector_atom[i]/flux[i];
            H_Rm += HR[j]*fix_partFlux[j]->vector_atom[i]/flux[i];
        }

        double Q_lost = flux[i]*cpLostFraction*Temp[i]+H_Rm*flux[i];
    }
}

```

```

    QConversion[i]=Q_lost/h;

    //double T_los= Q_lost/(flux[i]*cpLostFraction);

    //double Q_dt= fix_ConversionFlux->vector_atom[i] * (T_los-Temp[i]);

    //TConversion[i]=Q_lost/h - Q_dt;

    //Temp[i]=Temp[i] +Q_dt*h/cpLost[i]/rmass[i];

}

}

for (int i=0; i<nlocal; i++)
{

    if ((type[i]==atom_type))
    {

        radius[i]=pow(rmass[i]/rhoP[i]/PI*3/4,0.3333333);

        flux[i]/=h;

        //TConversion[i] = cpLost[i]*flux[i]*Temp[i];

        for (int j=0; j<N_components; j++)
        {
            fix_partFlux[j]->vector_atom[i]/=h;
        }
    }
}

//cout << "void FixCfdCouplingConvectionReaction::change_settings() - delating"
<< endl;
}

double FixCfdCouplingConvectionReaction::integrate_fraction(double c, double h,
double A, double EA, double T)
{
    double dt=h/10;
    if (c<SMAL) return 0;
    for (int i=0; i<10; i++)
    {
        c=heun(c, dt, A, EA, T);
    }

return c;
}

double FixCfdCouplingConvectionReaction::euler(double c, double h, double c_dot)
{
    return (c+h*c_dot);
}

//Sold-kinetic

```

```

double FixCfdCouplingConvectionReaction::heun(double c, double h, double A, double EA, double T)
{
    double c2=euler(c,h,-1.0*arrhenius(A,EA,T)*c);

    double temp = 0;
    double error = 1;
    for (int i=0; i<10; i++)
    {
        temp=0.5*(c+c2+h*(-1.0*arrhenius(A,EA,T)*c2));

        error = abs(temp-c2);

        c2=temp;
    };

    if (error>SMAL) printf("Fix couple/cfd/speciesConvection: Timestep to big for given kinetic\n");

    return c2;
}

double FixCfdCouplingConvectionReaction::arrhenius(double A, double EA, double T)
{
    const double R=8.3144598; //Universelle Gaskonstante

    return A*exp(-EA/(R*T));
}

void FixCfdCouplingConvectionReaction::delate_atoms()
{
    int nlocal = atom->nlocal;

    memory->create(dlist,nlocal,"delete_atoms:dlist");
    for (int i = 0; i < nlocal; i++) dlist[i] = 0;

    double *radius = atom->radius;

    for (int i = 0; i < nlocal; i++)
        if (radius[i]<=r_min) dlist[i] = 1;

    AtomVec *avec = atom->avec;
    nlocal = atom->nlocal;

    int i = 0;

    while (i < nlocal) {
        if (dlist[i]) {
            avec->copy(nlocal-1,i,1);
            dlist[i] = dlist[nlocal-1];
            nlocal--;
        } else i++;
    }

    atom->nlocal = nlocal;
    memory->destroy(dlist);
}

```



```

// if non-molecular system and compress flag set,
// reset atom tags to be contiguous
// set all atom IDs to 0, call tag_extend()

if (atom->molecular == 0) {
    int *tag = atom->tag;
    for (i = 0; i < nlocal; i++) tag[i] = 0;
    atom->tag_extend();
}

// reset atom->natoms
// reset atom->map if it exists
// set nghost to 0 so old ghosts of deleted atoms won't be mapped

bigint nblocal = atom->nlocal;
MPI_Allreduce(&nblocal, &atom->natoms, 1, MPI_LMP_BIGINT, MPI_SUM, world);
if (atom->map_style) {
    atom->nghost = 0;
    atom->map_init();
    atom->map_set();
}
}

/* ----- */

void FixCfdCouplingConvectionReaction::post_force(int)
{
}

```

```

//Speicherort: ~\LIGGGHTS_PUBLIC\src\fix_heat_gran_conduction2.h
#ifdef FIX_CLASS

FixStyle(heat/gran/conduction2,FixHeatGranCond2)
FixStyle(heat/gran,FixHeatGranCond2)

#else

#ifndef LMP_FIX_HEATGRAN_CONDUCTION2_H
#define LMP_FIX_HEATGRAN_CONDUCTION2_H

#include "fix_heat_gran.h"

namespace LAMMPS_NS {

class FixHeatGranCond2 : public FixHeatGran {
public:
    FixHeatGranCond2(class LAMMPS *, int, char **);
    ~FixHeatGranCond2();
    virtual void post_create();
    virtual void pre_delete(bool);

    int setmask();
    void init();
    virtual void post_force(int);

    virtual void cpl_evaluate(class ComputePairGranLocal *);
    void register_compute_pair_local(ComputePairGranLocal *);
    void unregister_compute_pair_local(ComputePairGranLocal *);

    virtual void updatePtrs();

protected:
    int iarg_;

    template <int,int> void post_force_eval(int,int);

    class FixPropertyGlobal* fix_conductivity_;
    double *conductivity_;

    const double* const* thermalCorrektionFaktor;
    const double* const* thermalCorrektionConstant;

    bool store_contact_data_;
    class FixPropertyAtom* fix_conduction_contact_area_;
    class FixPropertyAtom* fix_n_conduction_contacts_;
    double *conduction_contact_area_;
    double *n_conduction_contacts_;

    // model for contact area calculation
    int area_calculation_mode_;

    double fixed_contact_area_;

    // for heat transfer area correction
    int area_correction_flag_;
    double const* const* deltan_ratio_;
};

}

#endif
#endif

```

```

//Speicherort: ~\LIGGGHTS_PUBLIC\src\fix_heat_gran_conduction2.cpp
#include "fix_heat_gran_conduction2.h"

#include "atom.h"
#include "compute_pair_gran_local.h"
#include "fix_property_atom.h"
#include "fix_property_global.h"
#include "force.h"
#include "math_extra.h"
#include "math_extra_liggghts.h"
#include "properties.h"
#include "modify.h"
#include "neigh_list.h"
#include "pair_gran.h"
//#include "<iostream>"

using namespace std;

using namespace LAMMPS_NS;
using namespace FixConst;

// modes for conduction contact area calculation
// same as in fix_wall_gran.cpp

enum{ CONDUCTION_CONTACT_AREA_OVERLAP,
      CONDUCTION_CONTACT_AREA_CONSTANT,
      CONDUCTION_CONTACT_AREA_PROJECTION};

/* ----- */

FixHeatGranCond2::FixHeatGranCond2(class LAMMPS *lmp, int nargs, char **arg) :
  FixHeatGran(lmp, nargs, arg),
  fix_conductivity_(0),
  conductivity_(0),
  store_contact_data_(false),
  fix_conduction_contact_area_(0),
  fix_n_conduction_contacts_(0),
  conduction_contact_area_(0),
  n_conduction_contacts_(0),
  area_calculation_mode_(CONDUCTION_CONTACT_AREA_OVERLAP),
  fixed_contact_area_(0.),
  area_correction_flag_(0),
  deltan_ratio_(0)
{
  iarg_ = 5;

  bool hasargs = true;
  while(iarg_ < nargs && hasargs)
  {
    hasargs = false;
    if(strcmp(arg[iarg_],"contact_area") == 0) {

      if(strcmp(arg[iarg_+1],"overlap") == 0)
        area_calculation_mode_ = CONDUCTION_CONTACT_AREA_OVERLAP;
      else if(strcmp(arg[iarg_+1],"projection") == 0)
        area_calculation_mode_ = CONDUCTION_CONTACT_AREA_PROJECTION;
      else if(strcmp(arg[iarg_+1],"constant") == 0)
      {
        if (iarg_+3 > nargs)
          error->fix_error(FLERR,this,"not enough arguments for keyword 'con-
          tact_area constant'");
        area_calculation_mode_ = CONDUCTION_CONTACT_AREA_CONSTANT;
        fixed_contact_area_ = force->numeric(FLERR,arg[iarg_+2]);
      }
    }
  }
}

```

```

        if (fixed_contact_area_ <= 0.)
            error->fix_error(FLEERR,this,"'contact_area constant' value must be >
0");
        iarg_++;
    }
    else error->fix_error(FLEERR,this,"expecting 'overlap', 'projection' or
'constant' after 'contact_area'");
    iarg_ += 2;
    hasargs = true;
} else if(strcmp(arg[iarg_],"area_correction") == 0) {
    if (iarg_+2 > nargs) error->fix_error(FLEERR,this,"not enough arguments for
keyword 'area_correction'");
    if(strcmp(arg[iarg_+1],"yes") == 0)
        area_correction_flag_ = 1;
    else if(strcmp(arg[iarg_+1],"no") == 0)
        area_correction_flag_ = 0;
    else error->fix_error(FLEERR,this,"expecting 'yes' or 'no' after 'area_cor-
rection'");
    iarg_ += 2;
    hasargs = true;
} else if(strcmp(arg[iarg_],"store_contact_data") == 0) {
    if (iarg_+2 > nargs) error->fix_error(FLEERR,this,"not enough arguments for
keyword 'store_contact_data'");
    if(strcmp(arg[iarg_+1],"yes") == 0)
        store_contact_data_ = true;
    else if(strcmp(arg[iarg_+1],"no") == 0)
        store_contact_data_ = false;
    else error->fix_error(FLEERR,this,"expecting 'yes' or 'no' after
'store_contact_data'");
    iarg_ += 2;
    hasargs = true;
} else if(strcmp(style,"heat/gran/conduction") == 0)
    error->fix_error(FLEERR,this,"unknown keyword");
}

    if(CONDUCTION_CONTACT_AREA_OVERLAP != area_calculation_mode_ && 1 == area_cor-
rection_flag_)
        error->fix_error(FLEERR,this,"can use 'area_correction' only for 'con-
tact_area = overlap'");
}

/* ----- */
FixHeatGranCond2::~FixHeatGranCond2()
{
    if (conductivity_)
        delete []conductivity_;
}

/* ----- */

void FixHeatGranCond2::post_create()
{
    FixHeatGran::post_create();

    // register contact storage
    fix_conduction_contact_area_ = static_cast<FixPropertyAtom*>(modify-
>find_fix_property("contactAreaConduction","property/atom","scalar",0,0,this-
>style,false));
    if(!fix_conduction_contact_area_ && store_contact_data_)
    {
        const char* fixarg[10];

```

```

    fixarg[0]="contactAreaConduction";
    fixarg[1]="all";
    fixarg[2]="property/atom";
    fixarg[3]="contactAreaConduction";
    fixarg[4]="scalar";
    fixarg[5]="no";
    fixarg[6]="yes";
    fixarg[7]="no";
    fixarg[8]="0.";
    fix_conduction_contact_area_ = modify->add_fix_prop-
erty_atom(9,const_cast<char**>(fixarg),style);
}

    fix_n_conduction_contacts_ = static_cast<FixPropertyAtom*>(modify-
>find_fix_property("nContactsConduction","property/atom","scalar",0,0,this-
>style,false));
    if(!fix_n_conduction_contacts_ && store_contact_data_)
    {
        const char* fixarg[10];
        fixarg[0]="nContactsConduction";
        fixarg[1]="all";
        fixarg[2]="property/atom";
        fixarg[3]="nContactsConduction";
        fixarg[4]="scalar";
        fixarg[5]="no";
        fixarg[6]="yes";
        fixarg[7]="no";
        fixarg[8]="0.";
        fix_n_conduction_contacts_ = modify->add_fix_prop-
erty_atom(9,const_cast<char**>(fixarg),style);
    }

    if(store_contact_data_ && (!fix_conduction_contact_area_ || !fix_n_conduc-
tion_contacts_))
        error->one(FLERR,"internal error");
}

/* ----- */

void FixHeatGranCond2::pre_delete(bool unfixflag)
{
    // tell cpl that this fix is deleted
    if(cpl && unfixflag) cpl->reference_deleted();
}

/* ----- */

int FixHeatGranCond2::setmask()
{
    int mask = FixHeatGran::setmask();
    mask |= POST_FORCE;
    return mask;
}

/* ----- */

void FixHeatGranCond2::updatePtrs()
{
    FixHeatGran::updatePtrs();

    if(store_contact_data_)

```

```

    {
        conduction_contact_area_ = fix_conduction_contact_area_->vector_atom;
        n_conduction_contacts_ = fix_n_conduction_contacts_->vector_atom;
    }
}

/* ----- */

void FixHeatGranCond2::init()
{
    // initialize base class
    FixHeatGran::init();

    const double *Y, *nu, *Y_orig;
    double expo, Yeff_ij, Yeff_orig_ij, ratio;
    int max_type = atom->get_properties()->max_type();

    thermalCorrektionFaktor = static_cast<FixPropertyGlobal*>(modify-
>find_fix_property("thermalCorrektionFaktor","property/global","peratomtype-
pair",max_type,max_type,style))->get_array();
    thermalCorrektionConstant = static_cast<FixPropertyGlobal*>(modify-
>find_fix_property("thermalCorrektionConstant","property/global","peratomtype-
pair",max_type,max_type,style))->get_array();

    if (conductivity_) delete []conductivity_;
    conductivity_ = new double[max_type];
    fix_conductivity_ =
        static_cast<FixPropertyGlobal*>(modify->find_fix_property("thermalConductiv-
ity","property/global","peratomtype",max_type,0,style));

    // pre-calculate conductivity for possible contact material combinations
    for(int i=1;i< max_type+1; i++)
        for(int j=1;j<max_type+1;j++)
            {
                conductivity_[i-1] = fix_conductivity_->compute_vector(i-1);
                if(conductivity_[i-1] < 0.)
                    error->all(FLERR,"Fix heat/gran/conduction: Thermal conductivity
must not be < 0");
            }

    // calculate heat transfer correction

    if(area_correction_flag_)
    {
        if(!force->pair_match("gran",0))
            error->fix_error(FLERR,this,"area correction only works with using gran-
ular pair styles");

        expo = 1./pair_gran->stressStrainExponent();

        Y = static_cast<FixPropertyGlobal*>(modify->find_fix_property("youngsModu-
lus","property/global","peratomtype",max_type,0,style))->get_values();
        nu = static_cast<FixPropertyGlobal*>(modify->find_fix_property("pois-
sonsRatio","property/global","peratomtype",max_type,0,style))->get_values();
        Y_orig = static_cast<FixPropertyGlobal*>(modify->find_fix_property("youngs-
ModulusOriginal","property/global","peratomtype",max_type,0,style))->get_val-
ues();

        // allocate a new array within youngsModulusOriginal
        static_cast<FixPropertyGlobal*>(modify->find_fix_property("youngsModu-
lusOriginal","property/global","peratomtype",max_type,0,style))->new_ar-
ray(max_type,max_type);
    }
}

```

```

    // feed deltan_ratio into this array
    for(int i = 1; i < max_type+1; i++)
    {
        for(int j = 1; j < max_type+1; j++)
        {
            Yeff_ij      = 1./((1.-pow(nu[i-1],2.))/Y[i-1]      + (1.-pow(nu[j-1],2.))/Y[j-1]);
            Yeff_orig_ij = 1./((1.-pow(nu[i-1],2.))/Y_orig[i-1]+(1.-pow(nu[j-1],2.))/Y_orig[j-1]);
            ratio = pow(Yeff_ij/Yeff_orig_ij,expo);

            static_cast<FixPropertyGlobal*>(modify->find_fix_property("youngsModulusOriginal", "property/global", "peratomtype", max_type, 0, style))->array_modify(i-1, j-1, ratio);
        }
    }

    // get reference to deltan_ratio
    deltan_ratio_ = static_cast<FixPropertyGlobal*>(modify->find_fix_property("youngsModulusOriginal", "property/global", "peratomtype", max_type, 0, style))->get_array_modified();
}

updatePtrs();

// error checks on coarsegraining
}

/* ----- */

void FixHeatGranCond2::post_force(int vflag)
{
    if(history_flag == 0 && CONDUCTION_CONTACT_AREA_OVERLAP == area_calculation_mode_)
        post_force_eval<0, CONDUCTION_CONTACT_AREA_OVERLAP>(vflag, 0);
    if(history_flag == 1 && CONDUCTION_CONTACT_AREA_OVERLAP == area_calculation_mode_)
        post_force_eval<1, CONDUCTION_CONTACT_AREA_OVERLAP>(vflag, 0);

    if(history_flag == 0 && CONDUCTION_CONTACT_AREA_CONSTANT == area_calculation_mode_)
        post_force_eval<0, CONDUCTION_CONTACT_AREA_CONSTANT>(vflag, 0);
    if(history_flag == 1 && CONDUCTION_CONTACT_AREA_CONSTANT == area_calculation_mode_)
        post_force_eval<1, CONDUCTION_CONTACT_AREA_CONSTANT>(vflag, 0);

    if(history_flag == 0 && CONDUCTION_CONTACT_AREA_PROJECTION == area_calculation_mode_)
        post_force_eval<0, CONDUCTION_CONTACT_AREA_PROJECTION>(vflag, 0);
    if(history_flag == 1 && CONDUCTION_CONTACT_AREA_PROJECTION == area_calculation_mode_)
        post_force_eval<1, CONDUCTION_CONTACT_AREA_PROJECTION>(vflag, 0);
}

/* ----- */

void FixHeatGranCond2::cpl_evaluate(ComputePairGranLocal *caller)
{
    if(caller != cpl) error->all(FLERR, "Illegal situation in FixHeatGranCond2::cpl_evaluate");
}

```

```

    if(history_flag == 0 && CONDUCTION_CONTACT_AREA_OVERLAP == area_calcula-
tion_mode_)
        post_force_eval<0, CONDUCTION_CONTACT_AREA_OVERLAP>(0,1);
    if(history_flag == 1 && CONDUCTION_CONTACT_AREA_OVERLAP == area_calcula-
tion_mode_)
        post_force_eval<1, CONDUCTION_CONTACT_AREA_OVERLAP>(0,1);

    if(history_flag == 0 && CONDUCTION_CONTACT_AREA_CONSTANT == area_calcula-
tion_mode_)
        post_force_eval<0, CONDUCTION_CONTACT_AREA_CONSTANT>(0,1);
    if(history_flag == 1 && CONDUCTION_CONTACT_AREA_CONSTANT == area_calcula-
tion_mode_)
        post_force_eval<1, CONDUCTION_CONTACT_AREA_CONSTANT>(0,1);

    if(history_flag == 0 && CONDUCTION_CONTACT_AREA_PROJECTION == area_calcula-
tion_mode_)
        post_force_eval<0, CONDUCTION_CONTACT_AREA_PROJECTION>(0,1);
    if(history_flag == 1 && CONDUCTION_CONTACT_AREA_PROJECTION == area_calcula-
tion_mode_)
        post_force_eval<1, CONDUCTION_CONTACT_AREA_PROJECTION>(0,1);
}

```

```

/* ----- */

```

```

template <int HISTFLAG, int CONTACTAREA>
void FixHeatGranCond2::post_force_eval(int vflag, int cpl_flag)
{
    double hc, contactArea, delta_n, flux, dirFlux[3];
    int i, j, ii, jj, inum, jnum;
    double xtmp, ytmp, ztmp, delx, dely, delz;
    double radi, radj, radsum, rsq, r, tcoi, tcoj;
    int *ilist, *jlist, *numneigh, **firstneigh;
    int *contact_flag, **first_contact_flag;

    int newton_pair = force->newton_pair;

    if (strcmp(force->pair_style, "hybrid")==0)
        error->warning(FLERR, "Fix heat/gran/conduction implementation may not be
valid for pair style hybrid");
    if (strcmp(force->pair_style, "hybrid/overlay")==0)
        error->warning(FLERR, "Fix heat/gran/conduction implementation may not be
valid for pair style hybrid/overlay");

    inum = pair_gran->list->inum;
    ilist = pair_gran->list->ilist;
    numneigh = pair_gran->list->numneigh;
    firstneigh = pair_gran->list->firstneigh;
    if(HISTFLAG) first_contact_flag = pair_gran->listgranhistory->firstneigh;

    double *radius = atom->radius;
    double **x = atom->x;
    int *type = atom->type;
    int nlocal = atom->nlocal;
    int *mask = atom->mask;

    updatePtrs();

    if(store_contact_data_)
    {
        fix_conduction_contact_area_->set_all(0.);
        fix_n_conduction_contacts_->set_all(0.);
    }
}

```



```

// loop over neighbors of my atoms
for (ii = 0; ii < inum; ii++) {
    i = ilist[ii];
    xtmp = x[i][0];
    ytmp = x[i][1];
    ztmp = x[i][2];
    radi = radius[i];
    jlist = firstneigh[i];
    jnum = numneigh[i];
    if(HISTFLAG) contact_flag = first_contact_flag[i];

    for (jj = 0; jj < jnum; jj++) {
        j = jlist[jj];
        j &= NEIGHMASK;

        if (!(mask[i] & groupbit) && !(mask[j] & groupbit)) continue;

        if(!HISTFLAG)
        {
            delx = xtmp - x[j][0];
            dely = ytmp - x[j][1];
            delz = ztmp - x[j][2];
            rsq = delx*delx + dely*dely + delz*delz;
            radj = radius[j];
            radsum = radi + radj;
        }

        if ((HISTFLAG && contact_flag[jj]) || (!HISTFLAG && (rsq < radsum*radsum))) { //contact

            if(HISTFLAG)
            {
                delx = xtmp - x[j][0];
                dely = ytmp - x[j][1];
                delz = ztmp - x[j][2];
                rsq = delx*delx + dely*dely + delz*delz;
                radj = radius[j];
                radsum = radi + radj;
                if(rsq >= radsum*radsum) continue;
            }

            r = sqrt(rsq);

            if(CONTACTAREA == CONDUCTION_CONTACT_AREA_OVERLAP)
            {

                if(area_correction_flag_)
                {
                    delta_n = radsum - r;
                    delta_n *= deltan_ratio_[type[i]-1][type[j]-1];
                    r = radsum - delta_n;
                }

                if (r < fmax(radi, radj)) // one sphere is inside the other
                {
                    // set contact area to area of smaller sphere
                    contactArea = fmin(radi, radj);
                    contactArea *= contactArea * M_PI;
                }
                else
                    //contact area of the two spheres
                    contactArea = - M_PI/4.0 * ( (r-radi-radj)*(r+radi-radj)*(r-radi+radj)*(r+radi+radj) )/(r*r);
            }
        }
    }
}

```

```

}
else if (CONTACTAREA == CONDUCTION_CONTACT_AREA_CONSTANT)
    contactArea = fixed_contact_area_;
else if (CONTACTAREA == CONDUCTION_CONTACT_AREA_PROJECTION)
{
    double rmax = MathExtraLiggghts::max(radi, radj);
    contactArea = M_PI*rmax*rmax;
}

if(contactArea!=contactArea)
{
    fprintf(screen, "\n nan occurred. Overlapdistance is %f \n", r);
    fprintf(logfile, "\n nan occurred. Overlapdistance is %f \n", r);
    contactArea=1e-16;
    fprintf(screen, "Contactarea set to 1e-16\n");
    fprintf(logfile, "Contactarea set to 1e-16\n");
}

tcoi = conductivity_[type[i]-1];
tcoj = conductivity_[type[j]-1];
// /4.8545e-06 // *205994.43815017
// 0.00045265;
if (tcoi < SMALL_FIX_HEAT_GRAN || tcoj < SMALL_FIX_HEAT_GRAN) hc = 0.;
else hc = 4.*tcoi*tcoj/(tcoi+tcoj)*(contactArea*thermalCorrektion-
Faktor[type[i]-1][type[j]-1])+thermalCorrektionConstant[type[i]-1][type[j]-1];

flux = (Temp[j]-Temp[i])*hc;

dirFlux[0] = flux*delx;
dirFlux[1] = flux*dely;
dirFlux[2] = flux*delz;
if(!cpl_flag)
{
    //Add half of the flux (located at the contact) to each particle in
contact
    heatFlux[i] += flux;
    directionalHeatFlux[i][0] += 0.50 * dirFlux[0];
    directionalHeatFlux[i][1] += 0.50 * dirFlux[1];
    directionalHeatFlux[i][2] += 0.50 * dirFlux[2];

    if(store_contact_data_)
    {
        conduction_contact_area_[i] += contactArea;
        n_conduction_contacts_[i] += 1.;
    }
    if (newton_pair || j < nlocal)
    {
        heatFlux[j] -= flux;
        directionalHeatFlux[j][0] += 0.50 * dirFlux[0];
        directionalHeatFlux[j][1] += 0.50 * dirFlux[1];
        directionalHeatFlux[j][2] += 0.50 * dirFlux[2];

        if(store_contact_data_)
        {
            conduction_contact_area_[j] += contactArea;
            n_conduction_contacts_[j] += 1.;
        }
    }
}
}

if(cpl_flag && cpl) cpl->add_heat(i,j,flux);
}
}

```

```

}

if(newton_pair)
{
    fix_heatFlux->do_reverse_comm();
    fix_directionalHeatFlux->do_reverse_comm();
    fix_conduction_contact_area_->do_reverse_comm();
    fix_n_conduction_contacts_->do_reverse_comm();
}

if(!cpl_flag && store_contact_data_)
for(int i = 0; i < nlocal; i++)
{
    if(n_conduction_contacts_[i] > 0.5)
        conduction_contact_area_[i] /= n_conduction_contacts_[i];
}
}

/* -----
   register and unregister callback to compute
   ----- */

void FixHeatGranCond2::register_compute_pair_local(ComputePairGranLocal *ptr)
{
    if(cpl != NULL)
        error->all(FLERR,"Fix heat/gran/conduction allows only one compute of type
pair/local");
    cpl = ptr;
}

void FixHeatGranCond2::unregister_compute_pair_local(ComputePairGranLocal *ptr)
{
    if(cpl != ptr)
        error->all(FLERR,"Illegal situation in FixHeatGranCond2::unregister_com-
pute_pair_local");
    cpl = NULL;
}

```



```

    mutable double      **partDatSaturation_;           // Lagrangian
array
    mutable double      **partCoolingFlux_;

    mutable bool        validPartFlux_;               //indicator if found,
based on heat, but also used for species              //also indicates EXplicit
Coupling
    mutable bool        validPartTransCoeff_;         //indicator if found,
based on heat, but also used for species              //also indicates IMplicit
Coupling
    mutable bool        validPartFluid_;             //indicator if found,
based on heat, but also used for species

    mutable bool        haveTemperatureEqn_;         //indicator for tempera-
ture field or not

    mutable bool        useLiMason_;                 //switch to activate cal-
culation using Li-Mason
    mutable bool        useGeneralCorrelation_;       //switch to activate cal-
culation using a generalized correlation
    mutable scalarList  generalCorrelationParameters_; //parameter for general
correlation

    PtrList<volScalarField> partField;

    mutable

    label               nComponents;

    scalar              lambda_;                     // fluid thermal conduc-
tivity [W/(m*K)]
    scalar              Prandtl_;                     // Prandtl number

    //Species Word Lists
    mutable bool        useRawData_;

    mutable wordList    eulerianFieldNames_;         //List with Eulerian
fields to exchange (handed over)
    mutable wordList    fluidReactionComponents_;

    mutable scalarList  particleSpeciesValue_;       //list with scalar to in-
dicate particle property is available
    const wordList      partSpeciesNames_;
    const wordList      partSpeciesFluxNames_;
    const wordList      partSpeciesTransCoeffNames_;
    const wordList      partSpeciesFluidNames_;

    PtrList<scalarList> fraction_;

    scalarList          DMolecular_;

    //information related to external register
    mutable int         partHeatFluxPositionInRegister_;
    mutable std::vector<int> partSpeciesFluxPositionInRegister_;
    mutable int         partHeatTransCoeffPositionInRegister_;
    mutable std::vector<int> partSpeciesTransCoeffPositionInRegister_;
    mutable int         partHeatFluidPositionInRegister_;
    mutable std::vector<int> partSpeciesFluidPositionInRegister_;

    //Scalar properties

```

```

    mutable scalar      maxSource_;           // max (limited) value of
src field

    mutable scalar      scaleDia_;

    void allocateMyArrays(scalar initialValue) const;
    void setupModel() const;
    void setPointersToExternalArrays(   word nameFlux,           int position-
Flux,
                                       word nameTransCoeff,      int position-
TransCoeff,
                                       word nameFluid,           int position-
Fluid
                                       ) const;

    mutable double (scalarGeneralExchange::*Nusselt)(double Re, double Pr, dou-
ble voidfraction) const;
    double NusseltLiMason(double Re, double Pr, double voidfraction) const;
    double NusseltDeenEtAl(double Re, double Pr, double voidfraction) const;
    double NusseltGeneralCorrelation(double Re, double Pr, double voidfraction)
const;

public:

    //- Runtime type information
    TypeName("scalarGeneralExchange");

    // Constructors

    //- Construct from components
    scalarGeneralExchange
    (
        const dictionary& dict,
        cfdemCloud& sm
    );

    //- Construct from components
    scalarGeneralExchange
    (
        const dictionary& dict,
        cfdemCloud& sm,
        word dictName
    );

    // Destructor

    ~scalarGeneralExchange ();

    // Member Functions
    void setForce() const;

    void manipulateScalarField(volScalarField&, volScalarField&, int) const;

    void updateFields ();

    volScalarField getScalarField( word) const;

};

// * * * * *

```

```
} // End namespace Foam
// * * * * *
#endif
// *****
```







```

fraction_.clear();
fraction_.setSize(fluidReactionComponents_.size());

    forAll(fraction_, i)
    {
        fraction_.set(i, new scalarList(fractionList_.lookupOrDefault<scalar-
List>(fluidReactionComponents_[i], scalarList(0))));
    }
    partField.clear();

    partField.setSize(eulerianFieldNames_.size());

    forAll(partField, i)
    {
        if ((eulerianFieldNames_[i]!="T") && (eulerianFieldNames_[i]!="QConver-
sion"))
        {
            partField.set
            (
                i,
                new volScalarField
                (
                    IOobject
                    (
                        eulerianFieldNames_[i],
                        sm.mesh().time().constant(),
                        sm.mesh(),
                        IOobject::NO_READ,
                        IOobject::NO_WRITE
                    ),
                    sm.mesh(),
                    dimensionedScalar("zero", dimensionSet(1, 0, -1, 0, 0) ,
0.0)
                )
            );
        }
        else if (eulerianFieldNames_[i]=="QConversion")
        {
            partField.set
            (
                i,
                new volScalarField
                (
                    IOobject
                    (
                        eulerianFieldNames_[i],
                        sm.mesh().time().constant(),
                        sm.mesh(),
                        IOobject::NO_READ,
                        IOobject::NO_WRITE
                    ),
                    sm.mesh(),
                    dimensionedScalar("zero", dimensionSet(1, 2, -3, 0, 0) ,
0.0)
                )
            );
        }
        else if (eulerianFieldNames_[i]=="T")
        {
            partField.set
            (
                i,
                new volScalarField
                (
                    IOobject

```

```

        (
            eulerianFieldNames_[i],
            sm.mesh().time().constant(),
            sm.mesh(),
            IOobject::NO_READ,
            IOobject::NO_WRITE
        ),
        sm.mesh(),
        dimensionedScalar("zero", dimensionSet(0, 0, -1, 1, 0) ,
0.0)
    );
}
}

////////////////////////////////////
////////////////////////////////////
setForceSubModels(propsDict_);
setupModel();
if (probeIt_ && propsDict_.found("suppressProbe"))
    probeIt_=!Switch(propsDict_.lookup("suppressProbe"));
if(probeIt_)
{
    forAll(eulerianFieldNames_, fieldIt)
    {
        particleCloud_.probeM().initialize(typeName, typeName + "_" + eulerian-
FieldNames_[fieldIt] + ".logDat");
        particleCloud_.probeM().vectorFields_.append("Urel");
//first entry must the be the vector to probe
        if(eulerianFieldNames_[fieldIt]==tempFieldName_) //this is the tempera-
ture
        {
            particleCloud_.probeM().scalarFields_.append("Rep");
            particleCloud_.probeM().scalarFields_.append("Nu");
        }
        else
            particleCloud_.probeM().scalarFields_.append("Sh");
        particleCloud_.probeM().scalarFields_.append("exchangeRate");
        particleCloud_.probeM().writeHeader();
    }
}

for (int iFSub=0;iFSub<nrForceSubModels();iFSub++)
    forceSubM(iFSub).constructorCalls(typeName);

particleCloud_.setAllowCFDsubTimestep(false);

////////////////////////////////////
////////////////////////////////////

}

// *****
// Construct from components for scalarGeneralExchangePhaseChange
scalarGeneralExchange::scalarGeneralExchange
(
    const dictionary& dict,
    cfdemCloud& sm,

```

```

    word          dictName
)
:
    forceModel(dict, sm),
    propsDict_(dict.subDict(dictName + "Props")),
    scalarTransportProperties_ //this is clumsy, but effective
    (
        IObject
        (
            "scalarTransportProperties",
            sm.mesh().time().constant(),
            sm.mesh(),
            IObject::MUST_READ,
            IObject::NO_WRITE
        )
    ),
    generalPropsDict_(scalarTransportProperties_.subDict("generalManualProps")),
    simplePropsDict_(scalarTransportProperties_.subDict("simpleModelProps")),
    fractionList_(simplePropsDict_.subDict("solidFluidConversionRage")),
    voidfractionFieldName_(propsDict_.lookup("voidfractionFieldName")),
//common names/data
    velFieldName_(propsDict_.lookup("velFieldName")),
    tempFieldName_(propsDict_.lookup("tempFieldName")),
//temperature names/data
    partTempName_(propsDict_.lookup("partTempName")),
    partHeatFluxName_(propsDict_.lookupOrDefault<word>( "partHeatFluxName",
"none")),
    partHeatTransCoeffName_(propsDict_.lookupOrDefault<word>("partHeatTrans-
CoeffName", "none")),
    partHeatFluidName_(propsDict_.lookupOrDefault<word>( "partHeatFluid-
Name", "none")),
    partDat_(NULL),
    partDatFlux_(NULL),
    partDatTransCoeff_(NULL),
    partDatFluid_(NULL),
    partDatTmpExpl_(NULL),
    partDatTmpImpl_(NULL),
    validPartFlux_(false),
    validPartTransCoeff_(false),
    validPartFluid_(false),
    haveTemperatureEqn_(false),
    useLiMason_(false),
    useGeneralCorrelation_(false),
    lambda_(readScalar(propsDict_.lookup("lambda"))),
    Prandtl_(readScalar(propsDict_.lookup("Prandtl"))),
    eulerianFieldNames_( generalPropsDict_.lookup("eulerianFields")),
    partSpeciesNames_(propsDict_.lookup("partSpeciesNames")),
    partSpeciesFluxNames_(propsDict_.lookup("partSpeciesFluxNames")),
    partSpeciesTransCoeffNames_(propsDict_.lookup("partSpeciesTransCoeff-
Names")),
    partSpeciesFluidNames_(propsDict_.lookup("partSpeciesFluidNames")),
    DMolecular_(propsDict_.lookup("DMolecular")),
    fluidReactionComponents_( simplePropsDict_.lookup("fluidReactionCompo-
nents")),
    partHeatFluxPositionInRegister_(-1),
    partHeatTransCoeffPositionInRegister_(-1),
    partHeatFluidPositionInRegister_(-1),
    maxSource_(1e30),
    scaleDia_(1.),
//    partField(eulerianFieldNames_.size()),
    useRawData_(false)
{

```



```

        particleCloud_.dataExchangeM().allocateArray(partDatTmpImpl_,initVal,1);
    }
    //external particle data (e.g., fluxes, transCoeff, or fluid data) will be
    allocated in cloud,
    //just need to set pointers before access

}
// * * * * * public Member Functions * * * * *
* * //

void scalarGeneralExchange::setForce() const
{
    const_cast<scalarGeneralExchange*>(this)->updateFields();
    // do nothing
}

// * * * * * //
void scalarGeneralExchange::manipulateScalarField(volScalarField& explicitEuler-
Source,
                                                    volScalarField& implicitEuler-
Source,
                                                    int speciesID) const
{
    // reset Scalar field (== means hard reset)
    explicitEulerSource == dimensionedScalar("zero", explicitEulerSource.dimen-
sions(), 0.);
    implicitEulerSource == dimensionedScalar("zero", implicitEulerSource.dimen-
sions(), 0.);

    if(speciesID>=0 && particleSpeciesValue_[speciesID]<0.0) //skip if spe-
cies is not active
        return;

    //Set the names of the exchange fields
    word    fieldName;
    word    partDatName;
    scalar  transportParameter;

    if(speciesID<0) //this is the temperature
    {
        fieldName          = tempFieldName_;
        partDatName         = partTempName_;
        transportParameter = lambda_;

        setPointersToExternalArrays( partHeatFluxName_,      partHeatFluxPosi-
tionInRegister_,
                                    partHeatTransCoeffName_, partHeatTrans-
CoeffPositionInRegister_,
                                    partHeatFluidName_,      partHeatFluidPosi-
tionInRegister_
                                    );

        if(probeIt_)
            particleCloud_.probeM().setOutputFile(typeName+"_"+tempField-
Name_+".logDat");
    }
    else
    {
        fieldName          = eulerianFieldNames_[speciesID];
        partDatName         = partSpeciesNames_[speciesID];
        transportParameter = DMolecular_[speciesID];
    }
}

```

```

        setPointersToExternalArrays( partSpeciesFluxNames_[speciesID],
partSpeciesFluxPositionInRegister_[speciesID],
                                partSpeciesTransCoeffNames_[speciesID],
partSpeciesTransCoeffPositionInRegister_[speciesID],
                                partSpeciesFluidNames_[speciesID],
partSpeciesFluidPositionInRegister_[speciesID]
                                );
        if(probeIt_)
            particleCloud_.probeM().setOutputFile(typeName + "_" + fieldName +
".logDat");
    }

    //=====
    // get references
    const volScalarField& voidfraction_(particleCloud_.mesh().lookupOb-
ject<volScalarField> (voidfractionFieldName_)); // ref to voidfraction field
    const volVectorField& U_(particleCloud_.mesh().lookupObject<volVectorField>
(velFieldName_));
    const volScalarField& fluidScalarField_(particleCloud_.mesh().lookupOb-
ject<volScalarField> (fieldName)); // ref to scalar field
    const volScalarField& nufField = forceSubM(0).nuField();
    //=====

    if (scaleDia_ > 1)
        Info << typeName << " using scale = " << scaleDia_ << endl;
    else if (particleCloud_.cg() > 1)
    {
        scaleDia_ = particleCloud_.cg();
        Info << typeName << " using scale from liggghts cg = " << scaleDia_ <<
endl;
    }

    // realloc the arrays and get data
    allocateMyArrays(0.0);
    if(speciesID<0)
        particleCloud_.dataExchangeM().getData(partDatName, "scalar-atom",
partDat_);
    else
        if(particleSpeciesValue_[speciesID]>ALARGECONCENTRATION) //only pull if
needed
            particleCloud_.dataExchangeM().getData(partDatName, "scalar-atom",
partDat_);

    if ((useRawData_) && (speciesID>=0))
    {
        particleCloud_.averagingM().setScalarSum
        (
            explicitEulerSource,
            partDat_,
            particleCloud_.particleWeights(),
            NULL
        );
        Info << endl << endl << "Using RawData" << endl << endl;
        return;
    }

    // calc La based heat flux
    vector position(0,0,0);
    scalar voidfraction(1);
    vector Ufluid(0,0,0);
    scalar fluidValue(0);
    label cellI=0;

```

```

vector Us(0,0,0);
vector Ur(0,0,0);
scalar dscaled(0);
scalar dparcel(0);
scalar numberParticlesInParcel(1);
scalar nuf(0);
scalar magUr(0);
scalar As(0);
scalar Rep(0);
scalar Pr(0);

#include "resetVoidfractionInterpolator.H"
#include "resetUInterpolator.H"
#include "resetFluidScalarFieldInterpolator.H"

for(int index = 0; index < particleCloud_.numberOfParticles(); ++index)
{
    cellI = particleCloud_.cellIDs()[index][0];
    if(cellI >= 0)
    {
        if(forceSubM(0).interpolation())
        {
            position = particleCloud_.position(index);
            voidfraction = voidfractionInterpolator_.interpolate(position, cellI);
            Ufluid = UInterpolator_.interpolate(position, cellI);
            fluidValue = fluidScalarFieldInterpolator_.interpolate(position, cellI);
        }else
        {
            voidfraction = voidfraction_[cellI];
            Ufluid = U_[cellI];
            fluidValue = fluidScalarField_[cellI];
        }
        if(forceSubM(0).useCorrectedVoidage())
        {
            for (int iFSub=0; iFSub<nrForceSubModels(); iFSub++)
                voidfraction = forceSubM(iFSub).calculateCorrectedVoidage(voidfraction);
        }

        // calc relative velocity
        Us = particleCloud_.velocity(index);
        Ur = Ufluid-Us;
        magUr = mag(Ur);
        dscaled = 2*particleCloud_.radius(index);
        dparcel = dscaled;
        forceSubM(0).scaleDia(dscaled, index); //caution: this fct will
scale ds!

        numberParticlesInParcel = dparcel/dscaled;
        numberParticlesInParcel *= numberParticlesInParcel*numberPar-
particlesInParcel;
        As = dscaled*dscaled*M_PI*numberParticlesInParcel;
        nuf = nufField[cellI];
        Rep = dscaled*magUr*voidfraction/nuf; //MUST use superficial
velocity here!
        if(speciesID<0) //have temperature
            Pr = Prandtl_;
        else
            Pr = max(SMALL, nuf/transportParameter); //This is Sc
for species

```



```

        scalar alpha = transportParameter*(this->*Nusselt) (Rep,Pr,void-
fraction)/dscaled;

// calc convective heat flux [W]
scalar areaTimesTransferCoefficient = alpha * As;
scalar tmpPartFlux      = areaTimesTransferCoefficient
                        * (fluidValue - partDat_[index][0]);

// split implicit/explicit contribution
forceSubM(0).explicitCorrScalar( partDatTmpImpl_[index][0],
                                partDatTmpExpl_[index][0],
                                areaTimesTransferCoefficient,
                                fluidValue,
                                fluidScalarField_[cellI],
                                partDat_[index][0],
                                forceSubM(0).verbose()
                                );

    if(validPartFlux_)
        partDatFlux_[index][0] += tmpPartFlux; //MUST ADD total
source for ALL particles in parcel

    if(validPartTransCoeff_)
        partDatTransCoeff_[index][0] += alpha; //MUST ADD total
source coefficient here

    if(validPartFluid_)
        partDatFluid_[index][0]      = fluidValue; //MUST NOT add
here, since this is factor

    if( forceSubM(0).verbose())
    {
        Pout << "fieldName = " << fieldName << endl;
        Pout << "index      = " << index << endl;
        Pout << "partFlux = " << tmpPartFlux << endl;
        Pout << "magUr = " << magUr << endl;
        Pout << "As = " << As << endl;
        Pout << "r = " << particleCloud_.radius(index) << endl;
        Pout << "dscaled = " << dscaled << endl;
        Pout << "nuf = " << nuf << endl;
        Pout << "Rep = " << Rep << endl;
        Pout << "Pr/Sc = " << Pr << endl;
        Pout << "Nup/Shp = " << (this->*Nusselt) (Rep,Pr,voidfrac-
tion) << endl;

        Pout << "voidfraction = " << voidfraction << endl;
        Pout << "fluidValue = " << fluidValue << endl ;
        Pout << "partDat_[index][0] = " << partDat_[index][0] <<
endl;

        if(validPartFlux_)
            Pout << "partDatFlux: "          << partDatFlux_[index][0]
<< endl;

        if(validPartTransCoeff_)
            Pout << "partDatTransCoeff: " << partDatTransCoeff_[in-
dex][0] << endl;
    }

//Set value fields and write the probe
if(probeIt_)
{
    #include "setupProbeModelfields.H"
    // Note: for other than ext one could use vValues.append(x)
    // instead of setSize
    vValues.setSize(vValues.size()+1, Ur);

```

```

        if(speciesID<0) //this is the temperature, then also report
Rep
            sValues.setSize(sValues.size()+1, Rep);
            sValues.setSize(sValues.size()+1, (this-
>*Nusselt)(Rep,Pr,voidfraction));
            sValues.setSize(sValues.size()+1, tmpPartFlux);
            particleCloud_.probeM().writeProbe(index, sValues, vValues);
        }
    }

//Handle explicit and implicit source terms on the Euler side
//these are simple summations!
particleCloud_.averagingM().setScalarSum
(
    explicitEulerSource,
    partDatTmpExpl_,
    particleCloud_.particleWeights(),
    NULL
);

particleCloud_.averagingM().setScalarSum
(
    implicitEulerSource,
    partDatTmpImpl_,
    particleCloud_.particleWeights(),
    NULL
);

// scale with the cell volume to get (total) volume-specific source
particleCloud_.makeSpecific(explicitEulerSource);
explicitEulerSource*=-1;
particleCloud_.makeSpecific(implicitEulerSource);
implicitEulerSource*=-1;

// limit explicit source term
scalar explicitEulerSourceInCell;
forAll(explicitEulerSource,cellI)
{
    explicitEulerSourceInCell = explicitEulerSource[cellI];

    if(mag(explicitEulerSourceInCell) > maxSource_ )
    {
        explicitEulerSource[cellI] = sign(explicitEulerSourceInCell) *
maxSource_;
    }
}

//Reporting of integral quantities
//TODO: write to different file for speciesId>0
Field<scalar> writeValues; bool writeDiskNow=forceSubM(0).verboseToDisk();
//must call 'verboseToDisk()' only once since this function is incrementing a
counter!
writeValues.clear();
if( forceSubM(0).verbose() || writeDiskNow)
{
    scalar exchangeRate = gSum(-(explicitEulerSource
                                +implicitEulerSource*fluid-
ScalarField_)
                                *explicitEulerSource.mesh().V()
                                );
    // Note: for other than ext one could use writeValues.append(x)

```

```

// instead of setSize
writeValues.setSize(writeValues.size()+1, exchangeRate);

if(forceSubM(0).verbose())
{
    if(speciesID<0) //have temperature
        Info << "total convective particle-fluid heat flux [W] (Eulerian) =
"
                << exchangeRate
                << endl;
    else
        Info << "speciesID: " << speciesID
                << ": total convective particle-fluid species flux [kmol/s] (or
[kg/s]) (Eulerian) = "
                << exchangeRate
                << endl;
    }
}
if( writeDiskNow )
    for (int iFSub=0;iFSub<nrForceSubModels();iFSub++)
        forceSubM(iFSub).verboseToDiskWrite(writeValues);
}

void scalarGeneralExchange::updateFields()
{
    allocateMyArrays(0.0);
    for (int i=0; i<eulerianFieldNames_.size(); i++)
    {
        particleCloud_.dataExchangeM().getData(eulerianFieldNames_[i],"scalar-
atom", partDat_);

        particleCloud_.averagingM().setScalarSum
        (
            partField[i],
            partDat_,
            particleCloud_.particleWeights(),
            NULL
        );
    }
}

//Baustelle
volScalarField scalarGeneralExchange::getScalarField(word speciesName) const
{
    if (speciesName!="T")
    {
        bool first=false;

        volScalarField temp(speciesName+"Source",partField[0]*0.0);

        for (int i=0; i<fluidReactionComponents_.size(); i++)
        {
            if (speciesName==fluidReactionComponents_[i])
            {
                for (int j=0;j < partField.size(); j++)
                {

```

```

        if (!first && ( fraction_[i][j]>0.0))
        {

            temp.dimensions().reset(partField[j].dimensions());
            temp+= partField[j]*fraction_[i][j];
            first=true;

        }
        else if ( fraction_[i][j]>0.0)
        {
            temp+= partField[j]*fraction_[i][j];
        }
    }
    if (first==false)
    {

        temp.dimensions().reset(partField[(int) fraction_[i][frac-
tion_[i].size()-1]].dimensions());

    }

}
}
return temp;
}
else
{
    volScalarField temp(speciesName+"Source",partField[partField.size()-1]*0.0);
    volScalarField temp_impl(speciesName+"Source_impl",partField[part-
Field.size()-1]*0.0);

    manipulateScalarField(temp,temp_impl,-1);
    return temp;
}

}

// * * * * * //
double scalarGeneralExchange::NusseltLiMason(double Rep, double Pr, double void-
fraction) const
{
    double h1(0);
    double h2(0);
    double Nup(0);
    if (Rep < 200.0)
    {
        Nup = 2.0
            + 0.6
            * voidfraction*voidfraction*voidfraction*sqrt(voidfraction) //void-
fraction^3.5
            * sqrt(Rep)
            * pow(Pr,0.3333333333); //This is Sh for species
    }
    else if (Rep < 1500.0)
    {
        h1 = voidfraction*voidfraction*voidfraction*sqrt(voidfraction);
//voidfraction^3.5
        h2 = pow(Pr,0.3333333333);
        Nup = 2.0

```

```

        + 0.5 *h1*sqrt(Rep)*h2
        + 0.02*h1*pow(Rep,0.8)*h2;
    }
    else
    {
        Nup = 2.0
            + 0.000045
              * voidfraction*voidfraction*voidfraction*sqrt(voidfraction)
//voidfraction^3.5
            * pow(Rep,1.8);
    }

    return Nup;
}

// * * * * * //
double scalarGeneralExchange::NusseltDeenEtAl(double Rep, double Pr, double
voidfraction) const
{
    //WARNING: This function is fitted for Reynolds numbers between 0 and 100!!!
    double Nup(0);
    double PrPowOneThird = pow(Pr,0.3333333333) ;
    Nup =
        ( 7.0 - 10.0 * voidfraction + 5 * voidfraction * voidfraction )
        *
        (1.0 +
          0.17
          * pow(Rep,0.2)
          * PrPowOneThird
        )
        +
        ( 1.33 - 2.31 * voidfraction + 1.16 * voidfraction * voidfraction )
        * pow(Rep,0.7)
        * PrPowOneThird ;

    return Nup;
}

// * * * * * //
double scalarGeneralExchange::NusseltGeneralCorrelation(double Rep, double Pr,
double voidfraction) const
{
    //WARNING: This function my be fitted to data for a limited range of Reyn-
olds number!!
    double Nup(0);
    double PrPowOneThird = pow(Pr,0.3333333333) ;
    Nup =
        ( generalCorrelationParameters_[0]
          + generalCorrelationParameters_[1] * voidfraction
          + generalCorrelationParameters_[2] * voidfraction * voidfraction )
        *
        ( generalCorrelationParameters_[3]
          + generalCorrelationParameters_[4]
          * pow(Rep,0.2)
          * PrPowOneThird
        )
        +
        ( generalCorrelationParameters_[5]
          + generalCorrelationParameters_[6] * voidfraction
          + generalCorrelationParameters_[7] * voidfraction * voidfraction )
        * pow(Rep,0.7)
        * PrPowOneThird ;
}

```

```

    return Nup;
}

// * * * * * //
void scalarGeneralExchange::setupModel() const
{
    //just allocate arrays for internal use
    allocateMyArrays(0.0);

    //analyze dict to decide on allocation of external arrays
    //FLUXES (explicit coupling strategy)
    if(partHeatFluxName!="none")
    {
        validPartFlux_=true;
        particleCloud_.registerNamesFieldsUserCFDEMToExt(propsDict_.lookup("partHeatFluxName"), partHeatFluxPositionInRegister_);
    }
    bool validspeciesFlux =
        particleCloud_.checkAndregisterNamesFieldsUserCFDEMToExt(partSpeciesFluxNames_,
                                                                    partSpeciesFluxPositionInRegister_);

    if( (validPartFlux_ && !validspeciesFlux && partSpeciesFluxNames_.size()>0)
        || (!validPartFlux_ && validspeciesFlux ) )
        FatalError << "scalarGeneralExchange::setupModel: you have set a valid
species flux name, but a non-valid heatflux name (or vice versa). This will mess
up memory allocation. Please use both valid or non-valid flux names" <<
        abort(FatalError);

    //TRANSCOEFF and FLUID (implicit coupling strategy)
    if(partHeatTransCoeffName!="none")    {
        validPartTransCoeff_=true;
        Info << "Found a valid partHeatTransCoeffName: " << partHeatTransCoeffName_ << endl;
        particleCloud_.registerNamesFieldsUserCFDEMToExt(partHeatTransCoeffName_, partHeatTransCoeffPositionInRegister_);
    }
    if(partHeatFluidName!="none")    {
        validPartFluid_=true;
        Info << "Found a valid partHeatFluidName: " << partHeatFluidName_ <<
        endl;
        particleCloud_.registerNamesFieldsUserCFDEMToExt(partHeatFluidName_,
partHeatFluidPositionInRegister_);
    }
    if( validPartTransCoeff_ && !validPartFluid_ )
        FatalError <<"Transfer coefficient set, but and fluid name missing.
Check your entries in the couplingProperties! \n"
        << abort(FatalError);
    if( !validPartTransCoeff_ && validPartFluid_ )
        FatalError <<"Fluid name set, but transfer coefficient missing. Check
your entries in the couplingProperties! \n"
        << abort(FatalError);

    bool validspeciesTransCoeff =
        particleCloud_.checkAndregisterNamesFieldsUserCFDEMToExt(partSpeciesTransCoeffNames_,
                                                                    partSpeciesTransCoeffPositionInRegister_);
    bool validspeciesFluid =
        particleCloud_.checkAndregisterNamesFieldsUserCFDEMToExt(partSpeciesFluidNames_,

```

```

partSpe-
ciesFluidPositionInRegister_);

    if( (validPartTransCoeff_ && !validspeciesTransCoeff && partSpeciesTrans-
CoeffNames_.size())>0 ) || (!validPartTransCoeff_ && validspeciesTransCoeff )
        FatalError << "scalarGeneralExchange::setupModel: you have set a valid
species transCoeff name, but a non-valid heat trans coeff name (or vice versa).
This will mess up memory allocation. Please use both valid or non-valid trans-
Coeff names. You might want to use 'none' in partSpeciesTransCoeffNames to de-
activate the species transCoeff name!" << abort(FatalError);

    if( (validPartFluid_ && !validspeciesFluid && partSpeciesFluid-
Names_.size())>0 ) || (!validPartFluid_ && validspeciesFluid )
        FatalError << "scalarGeneralExchange::setupModel: you have set a valid
species fluid name, but a non-valid heat fluid name (or vice versa). This will
mess up memory allocation. Please use both valid or non-valid fluid names. You
might want to use 'none' in partSpeciesFluidNames to de-activate the species
fluid name!" << abort(FatalError);

//FINALLY check and Info
if(validPartFlux_ && validspeciesFlux)      {
    Info << "Found a valid partHeatFluxName and partSpeciesFluxName: " <<
partHeatFluxName_ << endl;
    Info << "scalarGeneralExchange (or derived model) will now proceed with
EXPLICIT flux coupling " << endl;
}
else if(validPartTransCoeff_ && validPartFluid_)
{
    Info << "Found a valid partHeatTransCoeffName and partHeatFluidfName (&
corresponding species names)." << endl;
    Info << "scalarGeneralExchange (or derived model) will now proceed with
IMPLICIT flux coupling " << endl;
}
else if(validPartFlux_ )      {
    Info << "Found a valid partHeatFluxName: " << partHeatFluxName_ << endl;
    Info << "scalarGeneralExchange (or derived model) will now proceed with
EXPLICIT flux coupling for heat. " << endl;
}
else
    FatalError << "scalarGeneralExchange::setupModel: you did not specify a
valid flux or transCoeff name set. Please either specify valid flux names, or
valid transCoeff and fluid names." << abort(FatalError);

//Make internal settings
if (propsDict_.found("maxSource"))
{
    maxSource_=readScalar(propsDict_.lookup ("maxSource"));
    Info << "limiting eulerian source field to: " << maxSource_ << endl;
}

if (propsDict_.found("useLiMason"))
{
    useLiMason_=readBool(propsDict_.lookup ("useLiMason"));
    Info << "setting for useLiMason: " << useLiMason_ << endl;
}
if (propsDict_.found("useRawData"))
{
    useLiMason_=readBool(propsDict_.lookup ("useRawData"));
    Info << "setting for useRawData: " << useRawData_ << endl;
}
if (propsDict_.found("useGeneralCorrelation"))

```

```

    {
        useGeneralCorrelation_ = readBool(propsDict_.lookup ("useGeneralCorrela-
tion"));
        Info << "setting for useGeneralCorrelation: " << useGeneralCorrelation_
<< endl;
    }

    if(useLiMason_ && useGeneralCorrelation_)
        FatalError <<"You cannot set 'useLiMason' AND 'useGeneralCorrelation' to
true. Just set one setting to true. \n"
        << abort(FatalError);

    if(useLiMason_)
        Nusselt=&scalarGeneralExchange::NusseltLiMason;
    else if(useGeneralCorrelation_)
    {
        //generalCorrelationParameters_ = propsDict_.lookup("generalCorrelation-
Parameters");
        if(generalCorrelationParameters_.size()<8 || generalCorrelationParame-
ters_.size())>8)
            FatalError <<"The data array specified as 'generalCorrelationParame-
ters' is too short or too long. Must specify exactly 8 values. You specified:
\n"
                << generalCorrelationParameters_ << endl
                << abort(FatalError);
        Nusselt=&scalarGeneralExchange::NusseltGeneralCorrelation;
    }
    else
        Nusselt=&scalarGeneralExchange::NusseltDeenEtAl;

    // define switches which can be read from dict
    forceSubM(0).setSwitchesList(3,true); // activate search for verbose switch
    forceSubM(0).setSwitchesList(4,true); // activate search for interpolate
switch
    forceSubM(0).setSwitchesList(8,true); // activate scalarViscosity switch
    forceSubM(0).setSwitchesList(9,true); // activate verboseToDisk switch
    forceSubM(0).setSwitchesList(10,true); // activate correctedVoidage switch

    // read those switches defined above, if provided in dict
    for (int iFSub=0;iFSub<nrForceSubModels();iFSub++)
        forceSubM(iFSub).readSwitches();

    particleCloud_.checkCG(true);

    if (propsDict_.found("scale"))
        scaleDia_ = scalar(readScalar(propsDict_.lookup("scale")));

    //check species names
    Info << "scalarGeneralExchange found the following eulerianFieldName: " <<
eulerianFieldNames_ << endl;
    int numTempEqn=0;
    for(int iEul=0;iEul<eulerianFieldNames_.size(); iEul++)
        if(eulerianFieldNames_[iEul]==tempFieldName_)
        {
            haveTemperatureEqn_ = true;
            numTempEqn = 1;
            Info << "scalarGeneralExchange identified the eulerianField '" <<
tempFieldName_
                << "' as being the temperature field" << endl;
        }

    //check if enough particle properties have been provided
    if(partSpeciesNames_.size()!=(eulerianFieldNames_.size()-numTempEqn))

```



```

FatalError <<"Not enough partSpeciesNames specified in the couplingProp-
erties file. \n"
    << abort(FatalError);
else
    Info << "Found valid partSpeciesNames: " << partSpeciesNames_ << endl;
    if(partSpeciesFluxNames_.size()!=(eulerianFieldNames_.size()-numTempEqn))
        FatalError <<"Not enough partSpeciesFluxNames specified in the cou-
plingProperties file. \n"
            << abort(FatalError);
    else
        Info << "Found valid partSpeciesFluxNames: " << partSpeciesFluxNames_ <<
endl;
        if(partSpeciesTransCoeffNames_.size()!=(eulerianFieldNames_.size()-
numTempEqn))
            FatalError <<"Not enough partSpeciesTransCoeffNames specified in the
couplingProperties file. \n"
                << abort(FatalError);
        else
            Info << "Found valid partSpeciesTransCoeffNames: " << partSpeciesTrans-
CoeffNames_ << endl;
            if(partSpeciesFluidNames_.size()!=(eulerianFieldNames_.size()-numTempEqn))
                FatalError <<"Not enough partSpeciesFluidNames specified in the cou-
plingProperties file. \n"
                    << abort(FatalError);
            else
                Info << "Found valid partSpeciesFluidNames: " << partSpeciesFluidNames_
<< endl;
                if(DMolecular_.size()!=(eulerianFieldNames_.size()-numTempEqn))
                    FatalError <<"Not enough DMolecular specified in the couplingProperties
file. \n"
                        << abort(FatalError);

                for(int iPart=0;iPart<partSpeciesNames_.size(); iPart++)
                {
                    if(partSpeciesNames_[iPart]=="none")
                        particleSpeciesValue_.setSize(particleSpeciesValue_.size()+1, -1);
//will not consider this species for coupling
                    else if(partSpeciesNames_[iPart]=="zero")
                        particleSpeciesValue_.setSize(particleSpeciesValue_.size()+1,
0.0);//will set this species to zero for coupling
                    else
                        particleSpeciesValue_.setSize(particleSpeciesValue_.size()+1,
2*ALARGECONCENTRATION);//set to a very large value to request pull
                }
            }

// * * * * *
void scalarGeneralExchange::setPointersToExternalArrays(    word nameFlux,
int positionFlux,
                                                                word nameTransCoeff,
int positionTransCoeff,
                                                                word nameFluid,
int positionFluid
                                                                ) const
{
    if(particleCloud_.particleDatFieldsUserCFDEMToExt.size() !=
        particleCloud_.namesFieldsUserCFDEMToExt.size()
        )
        FatalError <<  "\n\n****CATASTROPHIC ERROR MOST LIKELY CAUSED BY USER!!!
\n"
            <<  "particleCloud_.particleDatFieldsUserCFDEMToExt.size() is
NOT EQUAL to particleCloud_.namesFieldsUserCFDEMToExt.size()."

```



## Gleichungs- und Symbolverzeichnis

Symbol	Bezeichnung	Definition	Einheit
$t$	Zeit	SI-Einheit	$s$
$m$	Masse	SI-Einheit	$kg$
$d$	Distanz	SI-Einheit	$m$
$v$	Geschwindigkeit	$= \frac{dd}{dt}$	$\frac{m}{s}$
$a$	Beschleunigung	$= \frac{dv}{dt}$	$\frac{m}{s^2}$
$F$	Kraft	$= m \cdot a$	$\frac{kg}{m \cdot s^2}$
$x_i, x_j$	Schwerpunktposition der Partikel $i$ u. $j$	-	$(m, m, m)$
$d_{i,j}$	Schwerpunktstand zwei Partikel $i$ u. $j$	$= \ x_i - x_j\ $	$m$
$r_i, r_j$	Partikelradius der Partikel $i$ u. $j$	-	$m$
$r_{i,j}$	Effektiver Partikelradius zweier Partikel $i$ u. $j$	Gleichung 2.11	$m$
$\rho_i, \rho_j$	Dichte der Partikel $i$ u. $j$	$= \frac{m_i}{\frac{4}{3}r_i^3 \cdot \pi}$	$\frac{kg}{m^3}$
$m_i, m_j$	Masse der Partikel $i$ u. $j$	SI-Einheit	$kg$
$m_{i,j}$	Effektive Partikelmasse zweier Partikel $i$ u. $j$	Gleichung 2.14	$kg$
$\delta_{i,j}^N$	Normale Überlappung zweier Partikel $i$ u. $j$	$= d_{i,j} - r_i - r_j$	$m$
$v_{i,j}^N$	Normale Geschwindigkeit zwischen zwei Partikel $i$ u. $j$	-	$\frac{m}{s}$
$v_{i,j}^T$	Tangentiale Geschwindigkeit zwischen zwei Partikel $i$ u. $j$	-	$\frac{m}{s}$
$\delta_{i,j}^T$	Tangentiale Überlappung zweier Partikel $i$ u. $j$	$= \int_{t_0}^t v_{i,j}^T dt$	
$E_i, E_j$	E-Modul der Partikel $i$ u. $j$	-	$\frac{kg}{m \cdot s^2}$

Symbol	Bezeichnung	Definition	Einheit
$E_{i,j}$	Effektives E-Modul der Partikel $i$ u. $j$	Gleichung 2.12	
$G_{i,j}$	Schubmodul der Partikel $i$ u. $j$	Gleichung 2.13	
$F_{i,j}^{H,N}$	Herzscher normaler Kraftvektor	Gleichung 2.5	$\frac{kg}{m \cdot s^2}$
$k^N$	Normale Federkonstante	Gleichung 2.7	
$\gamma^N$	Normale Dämpfungskonstante	Gleichung 2.7	
$F_{i,j}^{H,T}$	Herzscher tangentialer Kraftvektor	Gleichung 2.5	$\frac{kg}{m \cdot s^2}$
$k^T$	Tangentiale Federkonstante	Gleichung 2.7	
$\gamma^T$	Tangentiale Dämpfungskonstante	Gleichung 2.7	
$F_{i,j}^H$	Kraft zwischen zwei Partikel $i$ u. $j$ nach Hertz	Gleichung 2.4	$\frac{kg}{m \cdot s^2}$
$\nu_i$	Poissonzahl	-	-
$COR$	Restitutionskoeffizient	-	-
$COF$	Reibungskoeffizient	-	-
$CORF$	Rollreibungskoeffizient	-	-
$dt_i$	Schrittweite	$= t_{i+1} - t_i$	s
$N_{\text{Verlet}}$	Verlettschritte	$= \frac{c}{2 \cdot v_{\text{max}} \cdot \Delta t}$	-
$t_{\text{Ray}}$	Rayleigh-Zeit	Gleichung 2.23	s
$\omega$	Winkelgeschwindigkeit	-	$\frac{\text{rad}}{s}$
$(x, y, z)$	3D Koordinate des Partikelschwerpunkt	Gleichung 2.25	$(m, m, m)$
$(x, y)$	2D Koordinate des Partikelschwerpunkt	Gleichung 2.25	$(m, m)$
$R_P$	Partikelabstand zur Rotationsachse	-	m
$R_W$	Werkzeugradius	-	m
$\varphi$	Partikelwinkel im Kreiskoordinatensystem	-	°

<b>Symbol</b>	<b>Bezeichnung</b>	<b>Definition</b>	<b>Einheit</b>
$(R_P, \varphi)$	2D Partikelpolarkoordinaten	Gleichung 2.27	$(m, ^\circ)$
$\alpha_i$	Diskretisierungswinkel	Abbildung 2.4	$^\circ$
$Abw.$	Abweichung	Gleichung 2.28	-
$\bar{\tau}$	Mittlere Verweilzeit	Gleichung 2.29	$s$
$\sigma^2$	Varianz der Verweilzeit	Gleichung 2.30	$s^2$
$\sigma_\theta^2$	dimensionslosen Varianz	Gleichung 2.31	-
$\hat{\tau}$	hydrodynamische Verweilzeit	Gleichung 2.32	$s$
$h$	Partikelhöhe	Gleichung 2.33	$m$
$\sigma_h$	Standardabweichung der Partikelhöhe	Gleichung 2.34	$m$
$V_P$	Partikelvolumen	-	$m^3$
$A_P$	Partikeloberfläche	-	$m^2$
$r_P$	Partikelradius	Gleichung 2.37	$m$
$\dot{Q}_{i,j}$	Wärmestrom von Partikel $i$ nach $j$	Gleichung 2.42	$\frac{kg \cdot m^2}{s^2}$
$\alpha_{i,j_B}$	Wärmeübergangskoeffizient von Partikel $i$ nach $j$ nach Batchelor et al. [79]	Gleichung 2.42	$\frac{kg \cdot m^2}{s^2 \cdot K}$
$\dot{Q}_i$	Wärmestromgradient von Partikel $i$	Gleichung 2.42	$\frac{kg \cdot m^2}{s^2}$
$A_{i,j}$	Kontaktfläche von Partikel $i$ und $j$	Gleichung 2.42	$m^2$
$\lambda_i$	Wärmeleitfähigkeit von Partikel $i$	-	$\frac{W}{m \cdot K}$
$T_i$	Partikeltemperatur von Partikel $i$	-	$K$
$c_{P_i}$	Spezifische Wärmekapazität von Partikel $i$	-	$\frac{J}{kg \cdot K}$
$r_{WT}$	Wärmeträgerpartikelradius	-	$m$
$r_{BM}$	Biomassepartikelradius	-	$m$
$\dot{m}_{WT}$	Wärmeträgermassenstrom	-	$\frac{kg}{s}$

Symbol	Bezeichnung	Definition	Einheit
$\dot{m}_{BM}$	Biomassemassenstrom	-	$\frac{kg}{s}$
$\alpha_{i,j_S}$	Wärmeübergangskoeffizient von Partikel $i$ nach $j$ nach Schlünder et al. [2]	Gleichung 2.43	$\frac{kg \cdot m^2}{s^2 \cdot K}$
$\overline{A}_k$	Diskrete Schlünder-Teilfläche	Abbildung 2.26	$m^2$
$\overline{A}$	Schlünderprojektionsfläche	$= \sum \overline{A}_k$	$m^2$
$\alpha_{i,j_{B \sim R}}$	Wärmeübergangskoeffizient von Partikel $i$ nach $j$ nach Batchelor et al. [79]	Gleichung 2.45	$\frac{kg \cdot m^2}{s^2 \cdot K}$
$\alpha_{i,j_{B \sim A}}$	Wärmeübergangskoeffizient von Partikel $i$ nach $j$ nach Batchelor et al. [79], direkt proportional zur Projektionsfläche	Gleichung 2.46	$\frac{kg \cdot m^2}{s^2 \cdot K}$
$\alpha_{BMod}$	Wärmeübergangskoeffizient von Partikel $i$ nach $j$ nach Batchelor et al. [79], modifiziert	Gleichung 2.47	$\frac{kg \cdot m^2}{s^2 \cdot K}$
$c_\alpha$	Korrekturfaktor	Gleichung 2.48	-
$k_\alpha$	Konstanter Korrekturfaktor	Gleichung 2.48	$\frac{kg \cdot m^2}{s^2 \cdot K}$
$D_{ax}$	Axialer Dispersionskoeffizient	Gleichung 3.1	$\frac{m^2}{s}$
$u$	Fluide Strömungsgeschwindigkeit	-	$\frac{m}{s}$
$p$	Fluider Druck	-	$\frac{kg}{m \cdot s^2}$
$\tilde{\tau}$	Spannungstensor	-	$\frac{1}{s}$
$h_r$	Enthalpie	-	$\frac{m^2}{s^2}$
$q''$	Wärmestromdichte	-	$\frac{kg}{m \cdot s^3}$
$Y$	Fluide Komponente	-	%
$k$	Reaktionsgeschwindigkeitskonstante	Gleichung 4.18	$\frac{1}{s}$
$A$	Frequenzfaktor	-	$\frac{1}{s}$
$EA$	Aktivierungsenergie	-	$\frac{kg \cdot m^2}{s^2 \cdot mol}$
$c(Y_i)$	Konzentration Komponente $Y_i$	-	$kg$

Symbol	Bezeichnung	Definition	Einheit
$\Delta h_{f,i}^0$	Reaktionsenthalpie der Komponente $i$	-	$\frac{J}{kg}$
$\dot{q}_{reaction}$	Reaktionsenthalpie summiert	$= -\sum \Delta h_{f,i}^0 \cdot \frac{\partial c(Y_i)}{\partial t}$	$\frac{kg \cdot m^2}{s^3}$
$S$	Entropie	Gleichung 4.23	$\frac{J}{K}$
$\mu$	dynamische Viskosität	-	$Pa \cdot s$
$\nu$	kinematische Viskosität	-	$\frac{m^2}{s}$
$\chi$	Diskretisierung des vereinigten Objektraum	$= \begin{cases} 1, & \text{falls } x \in \Omega_s \\ 0, & \text{sonst} \end{cases}$	-
$U_0$	Objektgeschwindigkeit	-	$\frac{m}{s}$
$\Omega_s$	Vereinigung aller Objekträume	$= \cup \Omega_{s_i}$	-
$\Omega_{s_i}$	Objektraum $i$ im CFD-Gitter	Abbildung 4.2	-
$\partial \Omega_{s_i}$	Rand des Objektraum $i$ im CFD-Gitter	Abbildung 4.2	-
$\tilde{\eta}$	Permeabilität	-	$\frac{m^3 \cdot s}{kg}$
$\tilde{\eta}_T$	Thermische Permeabilität	-	$\frac{s^3}{m^5}$
$f$	Ablösefrequenz	-	$\frac{1}{s}$
$S_p$	Quellterm zum Kräfteaustausch Partikel→Fluid	Gleichung 4.11	$\frac{kg}{m^2 \cdot s^2}$
$F_f$	Quellterm zum Kräfteaustausch Fluid→Partikel	Gleichung 4.13	$\frac{kg \cdot m}{s^2}$
$F_A$	Partikelantrieb	Gleichung 4.14	$\frac{kg \cdot m}{s^2}$
$\dot{q}_{PF}$	Wärmestrom Partikel/Fluid	Gleichung 4.15	$\frac{kg \cdot m^2}{s^3}$
$A_P$	Partikeloberfläche	-	$m^2$
$T_P$	Partikeltemperatur	-	$K$
$T_F$	Fluidtemperatur	-	$K$

Symbol	Bezeichnung	Definition	Einheit
$\alpha_{PF}$	Wärmeübergangskoeffizient Partikel/Fluid	Gleichung 4.16	$\frac{W}{m^2 \cdot K}$
$r_P$	Partikelradius	-	$m$
$c$	Konzentration	-	$mol$
$n_X$	Stoffmenge der Komponente $X$	Gleichung 4.29	$mol$
$n_G$	Gesamte Stoffmenge im Reaktor	Gleichung 4.30	$mol$
$\varepsilon_X$	Porösität nach def. Brinkmann-Strafverfahren	-	-
$\varepsilon_f$	Porösität nach def. CFDDEM-Kopplung	-	-
$\varepsilon_G$	Gesamter freier Volumenanteil	Gleichung 4.38	-
$S_\rho$	Quellterm: Massenwachstum durch Feststoff→Fluid-Reaktion	Gleichung 4.43	$\frac{kg}{m^3 \cdot s}$
$S_M$	Quellterm: Impulswachstum durch Feststoff→Fluid-Reaktion	Gleichung 4.44	$\frac{kg}{m^2 \cdot s^2}$
$S_{h_{Chem.}}$	Quellterm: Enthalpieänderung durch Fluid→Fluid-Reaktion	-	$\frac{kg \cdot m^2}{s^3}$
$S_T$	Quellterm: Enthalpieänderung durch Feststoff↔Fluid-Wärmeleitung	$= \dot{q}_{PF}$	$\frac{kg \cdot m^2}{s^3}$
$S_{h_s}$	Quellterm: Enthalpieänderung durch Feststoff→Fluid-Reaktion: Reaktionsenthalpie	Gleichung 4.45	$\frac{kg \cdot m^2}{s^3}$
$S_{h_m}$	Quellterm: Enthalpieänderung durch Feststoff→Fluid-Reaktion: Massenänderung	Gleichung 4.46	$\frac{kg \cdot m^2}{s^3}$
$S_{Chem.}$	Stoffaustauschfaktor der Fluid→Fluid-Reaktion	-	$\frac{kg \cdot m^2}{s^3}$
$S_{Y_k}$	Stoffaustauschfaktor der Feststoff→Fluid-Reaktion	Gleichung 4.47	$\frac{kg}{m^3 \cdot s}$
$dt_K$	CFD/DEM-Kopplungsintervall	Abbildung 4.16	$s$



## Konstanten

Zeichen	Bezeichnung	Definition
$\pi$	Kreiszahl	3.1415 ...
$R$	Universelle Gaskonstante	8.31446261815324 $\frac{\text{kg}\cdot\text{m}^2}{\text{s}^2\cdot\text{mol}\cdot\text{K}}$
$e$	Eulerzahl	2.71828 ...
$g$	Erdbeschleunigung	9.81 $\frac{\text{m}}{\text{s}^2}$

## Dimensionslose Kennzahlen

Kennzahl	Beschreibung	Definition
$Fr$	Froudezahl	Gleichung 2.24
$Fr_W$	Werkzeug-Froudezahl	Gleichung 2.24
$Fr_P$	Partikel-Froudezahl	Gleichung 2.24
$Bo$	Bodensteinzahl	Gleichung 2.31
$\Psi$	Sphärizität	Gleichung 2.38
$MG$	Mischgüte	Gleichung 2.39
$Bi$	Biotzahl	Gleichung 2.51
$Ma$	Machzahl	$= \frac{\text{Strömungsgeschwindigkeit}}{\text{Schallgeschwindigkeit}}$
$Re$	Reynoldszahl	Gleichung 4.8
$Sr$	Strouhalzahl	Gleichung 4.9
$\alpha$	Porosität	$= \frac{\text{Holraumvolumen}}{\text{Gesamtvolumen}}$
$Pr$	Prandtlzahl	$= \frac{\nu}{\lambda} = \frac{\text{kinematische Viskosität}}{\text{Leitfähigkeit}}$
$Nu$	Nußeltzahl	$= \frac{\alpha_{PF} \cdot L}{\lambda}$
$R^2$	Bestimmtheitsmaß	$= \frac{\text{erklärte Variation}}{\text{Gesamtvariation}}$

# Tabellenverzeichnis

Tabelle 1.1: Unterteilung der Pyrolyse nach Kaltschmitt et. al. [14].....	2
Tabelle 1.2: Ausgewählte Reaktortypen zur Schnellpyrolyse von Biomasse nach Bridgwater et. al. [20, 23].	7
Tabelle 2.1: Programmparameter der Wärmeträger-Wärmeträger Screeningversuche.....	20
Tabelle 2.2: Froude-Zahlen der Betriebsbedingungen .....	21
Tabelle 2.3: Vergleich der Werte für Reibung und Rollreibung .....	23
Tabelle 2.4: Parameter der Wand/Wärmeträger-Screeningversuche .....	23
Tabelle 2.5: Minima der Wand/Stahlkugeln für variierende Reibung und Rollreibung .....	25
Tabelle 2.6: Verweilzeitverteilung im Doppelschneckenmischreaktor nach Frey [70] .....	29
Tabelle 2.7: Simulationsrahmenbedingungen des ColdFlow-Modellreaktor.....	30
Tabelle 2.8: Bestimmung der Verweilzeit im DEM-Modellreaktor .....	31
Tabelle 2.9: Anlagentypen der Zerkleinerungslinie.....	33
Tabelle 2.10: Regressionsgerade und Standardabweichung der Modelle .....	38
Tabelle 2.11: Partikelform und Gesamtvolumenanteil der Multisphäre-Weizenstrohpartikel .....	40
Tabelle 2.12: Abschätzung der erforderlichen Sphären für die Single- und Multisphäre-Auflösung für Biomassepartikel .....	40
Tabelle 2.13: Partikelradien und Gesamtvolumenanteil des Singlesphäre-Weizenstrohpartikels .....	41
Tabelle 2.14: Sphärität der Singlesphären-Weizenstrohpartikel.....	42
Tabelle 2.15: Aufbau der Multisphäre-DEM Simulation mit Drehtrommel .....	43
Tabelle 2.16: Schüttdichte einer Weizenstroh/Stahlkugelmischung. Real vs. Simuliert.....	45
Tabelle 2.17: Rahmenbedingungen der Simulationen zur Bestimmung der Wärmeträger/Biomasse Reibung und Rollreibung .....	47
Tabelle 2.18: Experimentelle Bestimmung der Verweilzeit im Doppelschneckenmischreaktor von Weizenstroh und Biomasse [70].....	47
Tabelle 2.19: Validierung der Biomasse/Wärmeträger Kennwerte für eine 2 Hz Rotationsgeschwindigkeit der Schnecken. Bei den rot markierten Zeilen kam es zu einem Rückstau bei der Wärmeträgerzufuhr .....	48
Tabelle 2.20: Verweilzeiten in Abhängigkeit zur Biomasse/Wärmeträger COF und CORF .....	48
Tabelle 2.21: Materialkennwerte zur Bestimmung des Einflusses des Biomasse-Biomasse COF und CORF. ....	49
Tabelle 2.22: Verweilzeit im Reaktor für die Biomasse-Biomasse COF und CORF Einstellungen. ....	50
Tabelle 2.23: Materialkennwerte zur Bestimmung des Einflusses des E-Moduls und der Zeitschrittweite ..	50
Tabelle 2.24: Simulationsparameter zur Bestimmung der Verteilung der Überlappungen.....	53
Tabelle 2.25: Korrekturkoeffizienten für die Biomasse/Wärmeträger-Wärmeübergangskoeffizienten .....	59
Tabelle 2.26: Korrekturkoeffizienten für den Wärmeträger/Wärmeträger-Wärmeübergangskoeffizienten	61
Tabelle 2.27: Korrekturkoeffizienten für die Biomasse-Biomasse-Wärmeübergangskoeffizienten. ....	62
Tabelle 2.28: Biot-Zahlen für die Partikel/Partikel-Interaktionen .....	64
Tabelle 3.1: Vergleich der Verweilzeitverteilung Simulation vs. Versuch .....	66
Tabelle 4.1: Randbedingungen am Modellversuch: Karamansche Wirbelstraße (Abbildung 4.3) .....	81
Tabelle 4.2: Vergleich der Simulationen. rhoPisoFoam mit zylindrischer Aussparung. rhoPenaltyFoam 1 mit geschlossener Aussparung, rhoPenaltyFoam 2 mit homogener Gittergröße, rhoPenaltyFoam 3 (wie rhoPenaltyFoam 2) mit verfeinertem Gitterbereich .....	81
Tabelle 4.3: Maximalgeschwindigkeiten ermittelt zwischen 50-60s an einem Messpunkt, 130 mm in x-Richtung von der Rotationsachse in Strömungsrichtung .....	82
Tabelle 4.4: Startwerte der Parameterschätzung. LB:=lower Boundary, UB:=upper Boundary und X=Startwert.....	88
Tabelle 4.5: Vergleich der Weizenstrohzusammensetzung. ....	89
Tabelle 4.6: Kinetikparameter der Weizenstrohpyrolyse im TG/DTG.....	89
Tabelle 4.7: Verkohlungsrate $\gamma_i$ .....	90
Tabelle 4.8: Rahmenbedingungen zur Charakterisierung der Transportprozesse .....	91

Tabelle 4.9: Modellgüte der parallelen axialen Dispersion.....	92
Tabelle 4.10: Summenformel und Molmassen der Edukte bei der Pyrolyse von Biomasse.....	94
Tabelle 4.11: Rahmenbedingungen des Versuchsablaufs.....	95
Tabelle 4.12: Startwerte, untere (lb) und obere (ub) Grenzwerte sowie Resultat der Parameterschätzung.....	95
Tabelle 4.13: Abweichung eines Zeitschrittes zweier Integrationsalgorithmen zur analytischen Lösung.....	98
Tabelle 4.14: Reaktionen der Biomasse .....	102
Tabelle 4.15: Zusammensetzung und Stoffeigenschaften der Biomassepartikel .....	102
Tabelle 4.16: Reaktionskennwerte der Biomasse .....	102
Tabelle 4.17: Koeffizienten zur Berechnung der Wärmekapazitäten .....	103
Tabelle 4.18: Randbedingungen bei der Strömungssimulation im Doppelschneckenmischreaktor .....	104
Tabelle 0.1: Geometrische Daten Des semi-kontinuierlichen Pyrolysereaktors mit Oxidationsreaktor .....	118
Tabelle 0.2: Versuchsbedingungen zur Bestimmung der Verweilzeit im semi-kontinuierlichen Pyrolysereaktor mit Oxidationsstufe.....	118

# Abbildungsverzeichnis

Abbildung 1.1: Aufbau der Dissertation. Gelb: Kapitel 2, Rot: Kapitel 3, Grün: Kapitel 4.....	3
Abbildung 1.1: Strukturformel von Cellulose nach Suhas et. al. [18].....	3
Abbildung 1.2: Mögliche Hemicellulosestruktur nach Zheng et. al. [19] .....	3
Abbildung 1.3: Mögliche Strukturformel von Lignin nach Watkins et. al. [12] .....	4
Abbildung 1.4: Produktausbeute in Abhängigkeit der Temperatur nach Bridgewater et. al.[26] .....	5
Abbildung 1.5: Allgemeine Prozessschritte bei der Schnellpyrolyse von Biomasse.....	5
Abbildung 1.6: Schematischer Aufbau der Reaktortypen zur Schnellpyrolyse von Biomasse .....	8
Abbildung 1.7: Schematischer Aufbau der PYTHON-Technikumsanlage nach Funke [36], modifiziert um Anlagenteile und Prozessströme besser dar zu stellen. ....	10
Abbildung 2.1: Kontaktmodell nach Hertz: .....	13
Abbildung 2.2: Zustände bei einer rotierenden Trommel in Abhängigkeit der Drehgeschwindigkeit. a) 10 rpm, b) 60 rpm, c) 120 rpm und d) 200 rpm .....	18
Abbildung 2.3: Versuchsaufbau des Drehtrommel-Modellversuches .....	19
Abbildung 2.4: Auswertung des Drehtrommel-Modellversuches.....	19
Abbildung 2.5: Qualität der Simulationsergebnisse. Abstand in blau, Varianz in rot .....	21
Abbildung 2.6: Abweichung (Gleichung 2.28) der Simulationen zu den Versuchswerten in Abhängigkeit zur Reibung und Rollreibung .....	22
Abbildung 2.7: Abweichungen in Farbe und Höhe der Zustände bzgl. Rotationsgeschwindigkeit.....	24
Abbildung 2.8: Ausgewählte Minima für Wand/Wärmeträger <i>COF</i> und <i>CORF</i> .....	26
Abbildung 2.9: Versuchsaufbau des ColdFlow-Doppelschneckenmischreaktor zur Bestimmung der Verweilzeit .....	27
Abbildung 2.10: Skizze des Doppelschneckenmischreaktors. Links: PYTHON und Modell- Doppelschneckenmischreaktors, Rechts: modifizierte Geometrie, wie sie bei der DEM-Simulation zum Einsatz kommt. ....	28
Abbildung 2.11: Vergleich der mittleren ( $\tau$ ) und hydrodynamischen ( $\tau$ ) Verweilzeit der Simulationen ( $\tau_{Sim.}$ ) mit den Experimentell ermittelten Werten ( $\tau_{Exp.}$ ). Jeweils erster Eintrag der Legende stellt den X- Wert dar.....	32
Abbildung 2.12: Auswahl der untersuchten Weizenstrohpartikel. Zweifach vergrößert. ....	33
Abbildung 2.13: Histogramm in Abhängigkeit der Formparameter Länge und Breite .....	35
Abbildung 2.14: Partikelverteilung in Abhängigkeit der Partikelbreite (oben links), Partikellänge (oben rechts), und des Verhältnis aus beiden (unten) .....	36
Abbildung 2.15: Histogramm der Partikelhöhenverteilung von Weizenstrohpartikel.....	37
Abbildung 2.16: Vergleich der Soll- mit den Ist-Werten (Gleichung 2.35). Quadermodell (oben-links), Doppelkeilmodell (oben-rechts), Ellipsenmodell (unten-links) und Ellipsoidenmodell (unten-rechts).....	38
Abbildung 2.17: Diskretisierung der Partikelform .....	40
Abbildung 2.18: Multisphäre-Weizenstrohpartikel, eingesetzt bei der DEM-Simulation der Drehtrommel..	43
Abbildung 2.19: Single-/Multisphäresimulation einer Trommel mit 75 rpm, bei 13.6 s (links) und der dazugehörigen Mischgüte auf der x/y-Achse (rechts). Die Farbe der Auswertung gibt die lokale Mischgüte wieder.....	44
Abbildung 2.20: Verlauf der globalen Mischgüte in der rotierenden Trommel für Simulationen mit Multisphären (durchgezogene Linie) und Sphären (gestrichelte Linie) .....	46
Abbildung 2.21: Einfluss der Biomasse Reibung auf die Mischgüte bezogen auf die Reaktorposition .....	49
Abbildung 2.22: Einfluss des Elastizitätsmoduls auf die Mischgüte im Doppelschneckenmischreaktor .....	51
Abbildung 2.23: Verweilzeit in Abhängigkeit des E-Modul und der Schrittweite. Links: Kleine E-Moduln und großes $\Delta t$ . Rechts: Größere E-Moduln und kleines $\Delta t$ .....	52
Abbildung 2.24: Verteilung der Abstände an den Biomasse/Wärmeträger-Kontaktstellen.....	54

Abbildung 2.25: Temperaturverlauf beim Biomasse-Wärmeträgerkontakt, berechnet nach Batchelor. Wärmeträger (unten), Biomassepartikeln (oben).....	55
Abbildung 2.26: Abstandsbestimmung zwischen zwei Partikeln nach Schlünder. Die Projektionsfläche ist blau markiert, der Wärmeübergangsbereich rot mit grünen Abstandslinien <i>sk</i> .....	56
Abbildung 2.27: Temperaturverlauf beim Wärmeübergang nach Schlünder. Links der Fall der Überlappung der Partikel, rechts effektiver Abstand zwischen den Oberflächen.....	57
Abbildung 2.28: Vergleich der Wärmeübergangskoeffizienten. Links: Verlauf des Wärmeübergangskoeffizienten entlang des Abstandes, rechts: Vergleich der Wärmeübergangskoeffizienten Batchelor und Schlünder.....	58
Abbildung 2.29: Vergleich der Wärmeübergangskoeffizienten für den Biomasse/Wärmeträger-Kontakt nach Schlünder und nach Batchelor.....	59
Abbildung 2.30: Direkter Vergleich des Wärmeüberganges. Biomassepartikel oben, Wärmeträger unten. Wärmeübergang nach Schlünder links, nach korrigierten Batchelor-Modell rechts.....	60
Abbildung 2.31: Vergleich der Wärmeübergangskoeffizienten für den Wärmeträger/Wärmeträger-Kontakt nach Schlünder und nach modifizieren Batchelor-Modell.....	61
Abbildung 2.32: Vergleich der Wärmeübergangskoeffizienten für den Biomasse/Biomasse-Kontakt identischer Größe nach Schlünder und nach modifizierten Batchelor-Modell.....	62
Abbildung 3.1: Wärmeträgergeschwindigkeitsverteilung im Doppelschneckenmischreaktor in Abhängigkeit der Rotationsgeschwindigkeit. Durchgezogene Linien entsprechen der mittleren Geschwindigkeit im Segment, die gestrichelten Linien deren Standardabweichung.....	66
Abbildung 3.2: Verlauf der Mischgüte bezogen auf die Reaktorposition.....	67
Abbildung 3.3: Biomassetemperaturanstieg und Wärmeübergangskoeffizient im Doppelschneckenmischreaktor bei 1 – 4 Hz.....	68
Abbildung 3.4: Mittlere Anzahl und mittlere Projektionsfläche der Biomassepartikel.....	69
Abbildung 3.5: Mittlerer Temperaturgradient und Wärmeübergangskoeffizient bei konstantem Gesamtpartikelvolumenstrom. Biomasse- und Wärmeträgerdurchmesser $d=1$ mm. Der rote Pfeil markiert die Standardeinstellung mit 15.4%vol. Analog zu Abbildung 3.3 berechnet.....	70
Abbildung 3.6: Gegenüberstellung: Biomassepartikel mit hohem Wärmestrom ( $\geq 0.01 \text{ Wm}^2/\text{s}^3$ ). Links hoher Biomasseanteil, rechts niedriger Biomasseanteil. Wärmeträger ist ausgeblendet.....	70
Abbildung 3.7: Wärmeübergang bei konstantem Gesamtpartikelvolumen. Biomassepartikeldurchmesser nach (Tabelle 18). Wärmeträgerdurchmesser $d=1$ mm. Analog zu Abbildung 3.3 berechnet.....	71
Abbildung 3.8: Verteilung der Biomasse/Wärmeträger-Kontaktstellen. Links: Biomasse und Wärmeträgerpartikel gleicher Größe (Abbildung 3.5, roter Pfeil). Rechts: mit Biomassepartikelgrößenverteilung (Abbildung 3.7, roter Pfeil).....	72
Abbildung 3.9: Schneckengeometrien für die Zugabereihenfolge: 1. Biomasse 2. Wärmeträger (oben), 1. Wärmeträger 2. Biomasse (unten).....	72
Abbildung 3.10: Wärmeübergang im Doppelschneckenmischreaktor bei vertauschter Zugabereihenfolge in Abhängigkeit zur Schneckengeschwindigkeit. Links: Biomassetemperaturanstieg. Rechts: Biomassewärmübergangskoeffizient. Analog zu Abbildung 3.3 berechnet.....	73
Abbildung 3.11: Leichter Rückstau bei 2 Hz und vertauschter Zugabereihenfolge. Blau: Biomasse, rot: Wärmeträger.....	73
Abbildung 3.12: Ausgewählte Mischgüte bei vertauschter Zugabereihenfolge.....	74
Abbildung 4.1: Bewegungsmodelle bei der CFD-Simulation.....	78
Abbildung 4.2: Diskretisierung eines Objektes beim Brinkman-Strafverfahren.....	79
Abbildung 4.3: Geometrischer Aufbau des Kármán-schen-Wirbelstraße Testfall.....	80
Abbildung 4.4: Vergleich des pimpleDyMlbFoam-Solver (oben, FoamExtended-4.0) mit dem rhoPenaltySolver (unten).....	82
Abbildung 4.5: Reaktionsschema der Pyrolyse von Biomasse nach DiBlasi et. al. [127].....	87

Abbildung 4.6: Thermogravimetrische Analyse von Biomasse von 0.2 g Weizenstroh, bei einer Heizrate von 10 K/min in einer NETZSCH STA 409 C/CD und deren Auswertung. ....	89
Abbildung 4.7: Versuchsaufbau zur semi-kontinuierlichen Pyrolyse von Biomasse .....	91
Abbildung 4.8: Versuchsaufbau zur Bestimmung der axialen Dispersion.....	92
Abbildung 4.9: Verweilzeitverteilung des Dispersionsmodells in Abhängigkeit paralleler Stränge.....	93
Abbildung 4.10: Temperaturprofil eines Biomassepartikels im semi-kontinuierlichen Pyrolysereaktor.....	95
Abbildung 4.11: Konzentrationsverlauf am semi-kontinuierlichen Pyrolysereaktor. Gemessen vs. Berechnet. Blau: Konzentrationsverlauf am Oxidationsreaktor Ausgang. Rot: Pyrolysegas-Konversionsrate im Pyrolysereaktor.....	96
Abbildung 4.12: Hierarchischer Aufbau des foamPy-Solvers aus den Modulen. Links: Lagrange-Modul LIGGGHTS <sup>®</sup> mit Erweiterung, Rechts: Euler-Module piso-, rhoPimple-, rhoReactingFoam Module samt rhoPenaltyFoam-Erweiterung .....	97
Abbildung 4.13: Shrinking Particle Modell .....	98
Abbildung 4.14: Schemata des verwendeten Suchalgorithmus.....	101
Abbildung 4.15: Verlauf der Wärmekapazitäten.....	103
Abbildung 4.16: Datenaustausch zwischen CFD und DEM-Simulation mit Kopplungsintervall $dtK$ .....	103
Abbildung 4.17: Datenübertragung der Reaktionskomponenten. CFD ( $kSYk, i$ , Gleichung 4.47) durch DEM ( $j\partial mi, j, n + 1\partial t$ , Gleichung 4.34).....	104
Abbildung 4.18: Verwendete Geometrien bei der CFDEM <sup>®</sup> -Simulation. Oben DEM, unten CFD.....	105
Abbildung 4.19: Wärmeübergang im Doppelschneckenmischreaktor. Gestrichelte Verläufe stellen die Standardabweichung des farblich korrespondierenden Verlaufs dar.....	107
Abbildung 4.20: Allgemeiner Wärmeleistungsstrom durch die Gasphase bei der CDFDEM-Simulation der Schnellpyrolyse von Biomasse im Doppelschneckenmischreaktor.....	108
Abbildung 4.21: Temperaturverlauf von Biomasse, Wärmeträger und dem Hintergrundgas.....	108
Abbildung 4.22: Größenabhängiger Konzentrationsverlauf der Biomasse im Reaktor .....	109
Abbildung 4.23: Konzentrationsverlauf unter dem Reaktortemperaturverlauf und einer Verweilzeit im Misch- und Reaktionsbereich von 10.2 s.....	110
Abbildung 4.24: Reaktionsverlauf bei konstanten $- 600 K, + 650 K, 0 700 K, \nabla 750 K$ und $X 800 K$ ...	111
Abbildung 4.25: Produktentstehung im Doppelschneckenmischreaktor .....	111
Abbildung 4.26: Konzentrationsverlauf im Reaktor. Schnitt durch Reaktormitte. ....	112
Abbildung 0.1: semi-kontinuierlicher Pyrolysereaktor.....	118
Abbildung 0.2: Eingangs- und Ausgangssignal im Versuchsaufbau. Marker: Neon. Verweilzeitversuch 1 weicht von Hydrodynamischer Verweilzeit erheblich ab. Mögliche Ursachen: Technischer defekt, Fehlbedienung .....	119
Abbildung 0.3: Versuche mit Biomasse am semi-kontinuierlichen Pyrolysereaktor. ....	123
Abbildung 0.4: Programm: Weizenstrohformauswertung.....	125
Abbildung 0.5: Programm: Start von Simulationsgruppen auf dem Cluster.....	125

## Literaturverzeichnis

1. Kornmayer, C., *Verfahrenstechnische Untersuchungen zur Schenllpyrolyse von Lignocellulose im Doppelschneckenreaktor*, in *Chemieingenieurwesen und Verfahrenstechnik*. 2009, Universtät Fridericiana Karlsruhe. p. 289.
2. Schlünder, E.U. and E. Tsotsas, *Wärmeübertragung in Festbetten, durchmischten Schüttgütern und Wirbelschichten*. 1988: Thieme.
3. Pain, C.C., et al., *A numerical investigation of bubbling gas–solid fluidized bed dynamics in 2-D geometries*. Powder Technology, 2002. **128**(1): p. 56-77.
4. Oevermann, M., S. Gerber, and F. Behrendt, *Euler–Lagrange/DEM simulation of wood gasification in a bubbling fluidized bed reactor*. Particuology, 2009. **7**(4): p. 307-316.
5. Kafui, K.D., C. Thornton, and M.J. Adams, *Discrete particle-continuum fluid modelling of gas–solid fluidised beds*. Chemical Engineering Science, 2002. **57**(13): p. 2395-2410.
6. Sharma, A., V. Pareek, and D. Zhang, *Biomass pyrolysis—A review of modelling, process parameters and catalytic studies*. Renewable and Sustainable Energy Reviews, 2015. **50**: p. 1081-1096.
7. Zhou, H., et al., *Simulation of Coal Combustion in a Bubbling Fluidized Bed by Distinct Element Method*. Chemical Engineering Research and Design, 2003. **81**(9): p. 1144-1149.
8. Bruchmüller, J., et al., *Modeling the thermochemical degradation of biomass inside a fast pyrolysis fluidized bed reactor*. AIChE Journal, 2012. **58**(10): p. 3030-3042.
9. Scott, D.S., et al., *The role of temperature in the fast pyrolysis of cellulose and wood*. Industrial & Engineering Chemistry Research, 1988. **27**(1): p. 8-15.
10. Aramideh, S., et al., *Numerical simulation of biomass fast pyrolysis in an auger reactor*. Fuel, 2015. **156**: p. 234-242.
11. Qi, F. and M. M Wright, *A DEM modeling of biomass fast pyrolysis in a double screw reactor*. 2017.
12. Paysen, A., in *Mathematisch-Naturwissenschaftlichen Fakultät*. 2009, Christian-Albrechts-Universität: Kiel.
13. Carr, J.C.T., W., *History of British Steel Industry*. 1962: Blackwell Publishers.
14. Kaltschmitt, M., H. Hartmann, and H. Hofbauer, *Energie aus Biomasse: Grundlagen, Techniken und Verfahren*. 2016: Springer.
15. *Was die Wikinger zu gefürchteten Seefahrern machte*, in *Spiegel* 2018.
16. Dr. Diana Freudendahl, S.R., Dr. Ramona Langner. *Recycling von carbonfaserverstärkten Kunststoffen*. Werkstoffe in der Fertigung [cited 2020 24.05.]; Available from: <https://werkstoffzeitschrift.de/recycling-von-carbonfaserverstaerkten-kunststoffen/>.
17. Dahmen, N.C. *bioliq-Hauptseite*. 2019 [cited 2020 16.05.2020]; Available from: <https://www.bioliq.de/>.
18. Suhas, et al., *Cellulose: A review as natural, modified and activated carbon adsorbent*. Bioresource Technology, 2016. **216**: p. 1066-1076.
19. Zheng, Y., et al., *Chapter One - Principles and Development of Lignocellulosic Biomass Pretreatment for Biofuels*, in *Advances in Bioenergy*, Y. Li and X. Ge, Editors. 2017, Elsevier. p. 1-68.
20. H.K./P.N. *Cellulose*. [cited 2021 03.06.]; Available from: <https://www.spektrum.de/lexikon/biologie/cellulose/12670>.
21. *Lignin*. [cited 2021 03.06.2021]; Available from: <https://www.spektrum.de/lexikon/biologie/lignin/39320>.
22. Richter, G., *Stoffwechselphysiologie der Pflanzen: Physiologie und Biochemie des Primär- und Sekundärstoffwechsels*. 1997: Thieme.
23. Nover, L., E.W. Weiler, and W. Kuhn, *Allgemeine und molekulare Botanik*. 2008: Thieme.
24. Yang, H., et al., *Characteristics of hemicellulose, cellulose and lignin pyrolysis*. Fuel, 2007. **86**(12): p. 1781-1788.

25. Di Blasi, C., *Influences of physical properties on biomass devolatilization characteristics*. Fuel, 1997. **76**(10): p. 957-964.
26. Bridgwater, A.V., D. Meier, and D. Radlein, *An overview of fast pyrolysis of biomass*. Organic Geochemistry, 1999. **30**(12): p. 1479-1493.
27. Pecha, B. and M. Garcia-Perez, *Chapter 26 - Pyrolysis of Lignocellulosic Biomass: Oil, Char, and Gas*, in *Bioenergy*, A. Dahiya, Editor. 2015, Academic Press: Boston. p. 413-442.
28. Ranzi, E., P.E.A. Debiagi, and A. Frassoldati, *Mathematical Modeling of Fast Biomass Pyrolysis and Bio-Oil Formation. Note I: Kinetic Mechanism of Biomass Pyrolysis*. ACS Sustainable Chemistry & Engineering, 2017. **5**(4): p. 2867-2881.
29. Bridgwater, A.V., *Review of fast pyrolysis of biomass and product upgrading*. Biomass and Bioenergy, 2012. **38**: p. 27.
30. Meier, D., et al., *Betriebsergebnisse der ersten BtO-Anlage zur ablativen Flash-Pyrolyse von Holz mit Energiegewinnung in einem BHKW*. 2006: p. 115-120.
31. Meier, D., et al., *State-of-the-art of fast pyrolysis in IEA bioenergy member countries*. Renewable and Sustainable Energy Reviews, 2013. **20**(0): p. 619-641.
32. Jahiril, M.I., et al., *Biofuels Production through Biomass Pyrolysis —A Technological Review*. Energies, 2012. **5**(12): p. 4952-5001.
33. 16.07.2020]; Available from: <https://biochar-international.org/dynamotive/>.
34. *Ensyn*. 16.07.2020]; Available from: <http://www.ensyn.com/>.
35. Suren Wijeyekoon, K.T., Hilary Corkran and Paul Bennett, *Commercial status of direct thermochemical liquefaction technologies*, I.B.T. 34, Editor. 2020.
36. Funke, A., *Schematischer Aufbau der PYTHON-Technikumsanlage KIT*.
37. Fang, J., et al., *Improved SPH methods for simulating free surface flows of viscous fluids*. Applied Numerical Mathematics, 2009. **59**(2): p. 251-271.
38. Bernal, J.D., *The Bakerian Lecture, 1962. The Structure of Liquids*. Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences, 1964. **280**(1382): p. 299.
39. Cundall, P.A. and O.D. Strack, *A discrete numerical model for granular assemblies*. geotechnique, 1979. **29**(1): p. 47-65.
40. Hertz, H.R., *Über die Berührung fester elastischer Körper und über die Härte*. Journal für die reine und angewandte Mathematik, 1881. **92**: p. 156-171.
41. Ai, J., et al., *Assessment of rolling resistance models in discrete element simulations*. Powder Technology, 2011. **206**(3): p. 269-282.
42. Griebel, M., et al., *Numerische Simulation in der Moleküldynamik: Numerik, Algorithmen, Parallelisierung, Anwendungen*. Springer-Lehrbuch. 2003: Springer Berlin Heidelberg.
43. Li, Y., Y. Xu, and C. Thornton, *A comparison of discrete element simulations and experiments for 'sandpiles' composed of spherical particles*. Powder Technology, 2005. **160**(3): p. 219-228.
44. Kloss, C., et al., *Models, algorithms and validation for opensource DEM and CFD-DEM*. Progress in Computational Fluid Dynamics, 2012. **12**: p. 140-152.
45. Wang J, T.H., Han Z, Jensen MR. . *Discrete Element Method in LS-DYNA*. 2012 [cited 2020 18.09.]; Available from: [http://ftp.lstc.com/anonymous/outgoing/support/PRESENTATIONS/DEM\\_LS-DYNA\\_mrj\\_11\\_14\\_2012.pdf](http://ftp.lstc.com/anonymous/outgoing/support/PRESENTATIONS/DEM_LS-DYNA_mrj_11_14_2012.pdf).
46. Washino, K., et al., *Time step criteria in DEM simulation of wet particles in viscosity dominant systems*. Powder Technology, 2016. **302**: p. 100-107.
47. Verlet, L., *Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules*. Physical Review, 1967. **159**(1): p. 98-103.
48. Walther, J.H. and I.F. Sbalzarini, *Large-scale parallel discrete element simulations of granular flow*. Engineering Computations, 2009. **26**(6): p. 688-697.
49. Adrian Jensen, K.F., George Laird, *Improving the Precision of Discrete Element Simulations through Calibration Models*, in *International LS-DYNA Users Conference*, 13, Editor.



50. Parteli, E., *DEM simulation of particles of complex shapes using the multisphere method: Application for additive manufacturing*. AIP Conference Proceedings, 2013. **1542**: p. 185-188.
51. Markauskas, D., et al., *Investigation of adequacy of multi-sphere approximation of elliptical particles for DEM simulations*. Granular Matter, 2010. **12**(1): p. 107-123.
52. Podlozhnyuk, A., S. Pirker, and C. Kloss, *Efficient implementation of superquadric particles in Discrete Element Method within an open-source framework*. Computational Particle Mechanics, 2017. **4**(1): p. 101-118.
53. Beakawi Al-Hashemi, H.M. and O.S. Baghabra Al-Amoudi, *A review on the angle of repose of granular materials*. Powder Technology, 2018. **330**: p. 397-417.
54. Höhner, D., S. Wirtz, and V. Scherer, *Experimental and numerical investigation on the influence of particle shape and shape approximation on hopper discharge using the discrete element method*. Powder Technology, 2013. **235**: p. 614-627.
55. Aigner, A., *LIGGGHTS(R)/CFDEM(R)-Workshop*. 2013.
56. Frankowski, P. and M. Morgeneyer, *Calibration and Validation of DEM Rolling and Sliding Friction Coefficients in Angle of Repose and Shear Measurements*. AIP Conference Proceedings, 2013. **POWDERS AND GRAINS 2013: Proceedings of the 7th International Conference on Micromechanics of Granular Media**: p. 851-854.
57. Komossa, H., et al., *Transversal bed motion in rotating drums using spherical particles: Comparison of experiments with DEM simulations*. Powder Technology, 2014. **264**: p. 96-104.
58. Habermann, R. and P. Djifack, *Untersuchungen zum Bewegungsverhalten von Schüttgütern in horizontalen Pflugscharmischern*. 2016.
59. Fatahi, M.R. and A. Farzanegan, *DEM simulation of laboratory Knelson concentrator to study the effects of feed properties and operating parameters*. Advanced Powder Technology.
60. Roland, P.H.M.a.C.M., *Limits to Poisson's ratio in isotropic materials*. PHYSICAL REVIEW B, 2009. **80**.
61. Christine Hathaway, H.N., *The Effect of the Width of an Aluminum Plate on a Bouncing Steel Ball*. ISB Journal of Science, 2013. **7**(1): p. 4.
62. DCS Computing GmbH, J.L.a.S.C., *LIGGGHTS(R)-PUBLIC Documentation*. 2017.
63. Jahani, M., A. Farzanegan, and M. Noaparast, *Investigation of screening performance of banana screens using LIGGGHTS DEM solver*. Powder Technology, 2015. **283**: p. 32-47.
64. Kuchling, H., *Taschenbuch der Physik*. 2014: Hanser Fachbuchverlag.
65. JAYARAM, M.A., *MECHANICS OF MATERIALS: WITH PROGRAMS IN C*. 2007: PHI Learning.
66. Sondergaard, R., K. Chaney, and C. Brennen, *Measurements of Solid Spheres Bouncing Off Flat Plates*. Vol. 57. 1990.
67. Grote, K.H. and J. Feldhusen, *Dubbel: Taschenbuch für den Maschinenbau*. 2011: Springer Berlin Heidelberg.
68. Syed, Z., M. Tekeste, and D. White, *A coupled sliding and rolling friction model for DEM calibration*. Journal of Terramechanics, 2017. **72**: p. 9-20.
69. Maione, R., et al., *DEM investigation of granular flow and binary mixture segregation in a rotating tumbler: Influence of particle shape and internal baffles*. Powder Technology, 2015. **286**: p. 732-739.
70. Frey, F., *Charakterisierung eines corotierenden Doppelschneckenmischreaktors*, in *Bioverfahrenstechnik*. 2016, Karlsruher Institut für Technologie.
71. Schwister, K.L., Volker, *Verfahrenstechnik für Ingenieure, Lehr- und Übungsbuch*. 2012: Carl Hanser Verlag München.
72. Krick, B., *Untersuchung von Strohballen und Strohballenkonstruktionen hinsichtlich ihrer Anwendung für ein energiesparendes Bauen unter besonderer Berücksichtigung der lasttragenden Bauweise*. 2008: Kassel University Press.

73. Gonzalez, R.C., R.E. Woods, and S.L. Eddins, *Digital Image Processing Using MATLAB*. 2004: Pearson Education.
74. Hansjörg Wieland, T.A.u.F.-J.B., *Stroh – Renaissance eines alten Baustoffes?* LANDTECHNIK, 4/2002. **57**(LÄNDLICHES BAUEN): p. 2.
75. H. Adensam, J.B., M. Staribacher, S. Hiller, G. Unfried, E. Schwarzmüller, H. Hegedys, V. Frosch, E. Ganglberger, *Stroh kompakt - Fabrik der Zukunft als regionales Produktionsnetzwerk auf Basis nachwachsender Rohstoffe anhand eines Pilotprojektes im Bereich Dämmstoffe*, I.u.T. Bundesministerium für Verkehr, Editor. 2005: Radetzkystraße 2, 1030 Wien.
76. Wadell, H., *Volume, Shape, and Roundness of Quartz Particles*. The Journal of Geology, 1935. **43**(3): p. 250-280.
77. O'Dogherty, M.J., et al., *A Study of the Physical and Mechanical Properties of Wheat Straw*. Journal of Agricultural Engineering Research, 1995. **62**(2): p. 133-142.
78. Sliker, A., *Measuring Poisson's ratios in wood*. Experimental Mechanics, 1972. **12**(5): p. 239-242.
79. G. K. Batchelor, F.R.S.a.R.W.O.B., *Thermal or electrical conduction through a granular material*. Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 1977. **355**(1682): p. 313.
80. Chaudhuri, B., F.J. Muzzio, and M.S. Tomassone, *Modeling of heat transfer in granular flow in rotating vessels*. Chemical Engineering Science, 2006. **61**(19): p. 6348-6360.
81. Solutions, D., *EDEM 2.6 Theory Reference Guide*, D. Solutions, Editor. 2014.
82. Richter, F., *Die physikalischen Eigenschaften der Stähle „Das 100 - Stähle - Programm“ Teil I: Tafeln und Bilder*
83. Petersen, C., *Naturwissenschaften im Fokus III : Grundlagen der Elektrizität, Strahlung und relativistischen Mechanik, einschließlich stellarer Astronomie und Kosmologie*. 2017, Wiesbaden Springer Vieweg
84. Verclyte, A., *Mass and heat transfer modelling in screw reactors*. 2013, University Gent.
85. Forster, O., *Analysis 1*. 2006: Vieweg+Teubner Verlag.
86. Lédé, J., *Comparison of contact and radiant ablative pyrolysis of biomass*. Journal of Analytical and Applied Pyrolysis, 2003. **70**(2): p. 601-618.
87. Di Blasi, C., *Heat transfer mechanisms and multi-step kinetics in the ablative pyrolysis of cellulose*. Chemical Engineering Science, 1996. **51**(10): p. 2211-2220.
88. RobertG, *some bugs in LIGGGHTS* 2015.
89. Funke, A., et al., *Modelling and improvement of heat transfer coefficient in auger type reactors for fast pyrolysis application*. Chemical engineering and processing, 2018. **130**: p. 67-75.
90. Scott, D.S., et al., *A second look at fast pyrolysis of biomass—the RTI process*. Journal of Analytical and Applied Pyrolysis, 1999. **51**(1): p. 23-37.
91. Hoover, W.G., *Smooth Particle Applied Mechanics: The State Of The Art*. 2006: World Scientific Publishing Company.
92. Chen, S. and G.D. Doolen, *LATTICE BOLTZMANN METHOD FOR FLUID FLOWS*. Annual Review of Fluid Mechanics, 1998. **30**(1): p. 329-364.
93. Chorin, A.J. and J.E. Marsden, *A Mathematical Introduction to Fluid Mechanics*. 2012: Springer New York.
94. F. Moukalled, L.M., M. Darwish, *The Finite Volume Method in Computational Fluid Dynamics An Advanced Introduction with OpenFOAM® and Matlab*. Fluid Mechanics and Its Applications, ed. 113. Vol. 1. 2016: Springer International Publishing.
95. Greenshields, C. *OpenFOAM 2.3.0: Mesh Motion*. 2014; Available from: <https://openfoam.org/release/2-3-0/mesh-motion/>.
96. Greenshields, C. *OpenFOAM 2.1.0: Arbitrary Mesh Interface*. 2011; Available from: <https://openfoam.org/release/2-1-0/ami/>.

97. Miller, S., Campbell, R. , Elsworth, C. , Pitt, J. and Boger, D., *An Overset Grid Method for Fluid-Structure Interaction*. World Journal of Mechanics, 2014. **4**: p. 20.
98. Peskin, C.S., *Flow patterns around heart valves: A numerical method*. Journal of Computational Physics, 1972. **10**(2): p. 252-271.
99. Hu, X.Y., et al., *A conservative interface method for compressible flows*. Journal of Computational Physics, 2006. **219**(2): p. 553-578.
100. Kim, Y. and C. Peskin, *Penalty immersed boundary method for an elastic boundary with mass*. Physics of Fluids, 2007. **19**.
101. Glowinski, R., T.-W. Pan, and J. Periaux, *A fictitious domain method for Dirichlet problem and applications*. Computer Methods in Applied Mechanics and Engineering, 1994. **111**(3): p. 283-303.
102. Meinke, M., et al., *A cut-cell method for sharp moving boundaries in Cartesian grids*. Computers and Fluids, 2013. **85**.
103. Hylla, E.A., *Eine Immersed Boundary Methode zur Simulation von Strömungen in komplexen und bewegten Geometrien*, in Technische Universität Berlin, Fakultät V - Verkehrs- und Maschinensysteme. 2013, Universitätsverlag der TU Berlin: Berlin.
104. Brinkman, H.C., *A calculation of the viscosity and the sedimentation constant for solutions of large chain molecules taking into account the hampered flow of the solvent through these molecules*. Physica, 1947. **13**(8): p. 447-448.
105. Brinkman, H.C., *On the permeability of media consisting of closely packed porous particles*. Flow, Turbulence and Combustion, 1949. **1**(1): p. 81.
106. Darcy, H., *Les fontaines publiques de la ville de Dijon. Exposition et application des principes à suivre et des formules à employer dans les questions de distribution d'eau: ouvrage terminé par un appendice relatif aux fournitures d'eau de plusieurs villes au filtrage des eaux et à la fabrication des tuyaux de fonte, de plomb, de tole et de bitume*. 1856: Dalmont.
107. Angot, P., C.-H. Bruneau, and P. Fabrie, *A penalization method to take into account obstacles in incompressible viscous flows*. Numerische Mathematik, 1999. **81**(4): p. 497-520.
108. Brown-Dymkoski, E., N. Kasimov, and O.V. Vasilyev, *A characteristic based volume penalization method for general evolution problems applied to compressible viscous flows*. Journal of Computational Physics, 2014. **262**: p. 344-357.
109. Joel H. Ferziger, M.P., Robert L. Street, *Computational Methods for Fluid Dynamics*. 2020: Springer.
110. Das, M.K., P.P. Mukherjee, and K. Muralidhar, *Modeling Transport Phenomena in Porous Media with Applications*. 2017: Springer International Publishing.
111. Koch, D.L. and R.J. Hill, *INERTIAL EFFECTS IN SUSPENSION AND POROUS-MEDIA FLOWS*. Annual Review of Fluid Mechanics, 2001. **33**(1): p. 619-647.
112. van Buijtenen, M.S., et al., *Numerical and experimental study on multiple-spout fluidized beds*. Chemical Engineering Science, 2011. **66**(11): p. 2368-2376.
113. Hill, R.J., D.L. Koch, and A.J.C. Ladd, *Moderate-Reynolds-number flows in ordered and random arrays of spheres*. Journal of Fluid Mechanics, 2001. **448**: p. 243-278.
114. Leitz, K.H., et al., *CFDEM modelling of particle heating and acceleration in cold spraying*. International Journal of Refractory Metals and Hard Materials, 2018. **73**: p. 192-198.
115. Leitz, K.-H., et al., *CFDEM Modelling of Particle Heating and Acceleration in Cold Gas Spraying*. 2017.
116. Ergun, *Fluid flow through packed columns*. Chem. Eng, 1952. **48**.
117. Wen, C.Y. *Mechanics of fluidization*. in Chem. Eng. Prog., Symp. Ser. 1966.
118. L. Schiller, Z.N., *Über die grundlegenden Berechnungen bei der Schwerkraftaufbereitung*. Z. Vereines Deutscher Inge., 1933. **77**: p. 318-321.
119. Gidaspow, D., *Multiphase flow and fluidization: continuum and kinetic theory descriptions*. 1994: Academic press.

120. Beetstra, R., *Drag force in random arrays of mono-and bidisperse spheres*. 2005: University of Twente [Host].
121. Beetstra, R., M. Hoef, and H. Kuipers, *Drag Force of Intermediate Reynolds Number Flow Past Mono- and Bidisperse Arrays of Spheres*. *AIChE Journal*, 2007. **53**: p. 489-501.
122. Di Felice, R., *The voidage function for fluid-particle interaction systems*. *International Journal of Multiphase Flow*, 1994. **20**(1): p. 153-159.
123. Li, J. and D.J. Mason, *A computational investigation of transient heat transfer in pneumatic transport of granular particles*. *Powder Technology*, 2000. **112**(3): p. 273-282.
124. *OpenFOAM User Guide*. [cited 2020 06.09.2020]; Available from: <https://cfdirect.openfoam/user-guide/>.
125. Ranzi, E., et al., *Chemical Kinetics of Biomass Pyrolysis*. *Energy & Fuels*, 2008. **22**(6): p. 4292-4300.
126. Debiagi, P.E.A., et al., *Detailed kinetic mechanism of gas-phase reactions of volatiles released from biomass pyrolysis*. *Biomass and Bioenergy*, 2016. **93**: p. 60-71.
127. Di Blasi, C., *Analysis of Convection and Secondary Reaction Effects Within Porous Solid Fuels Undergoing Pyrolysis*. Vol. 90. 1993. 315-340.
128. Várhegyi, G., et al., *Kinetic modeling of biomass pyrolysis*. *Journal of Analytical and Applied Pyrolysis*, 1997. **42**(1): p. 73-87.
129. Órfão, J.J.M. and J.L. Figueiredo, *A simplified method for determination of lignocellulosic materials pyrolysis kinetics from isothermal thermogravimetric experiments*. *Thermochimica Acta*, 2001. **380**(1): p. 67-78.
130. Li, Z., et al., *Kinetic study of corn straw pyrolysis: Comparison of two different three-pseudocomponent models*. *Bioresource Technology*, 2008. **99**(16): p. 7616-7622.
131. Burhenne, L., et al., *The effect of the biomass components lignin, cellulose and hemicellulose on TGA and fixed bed pyrolysis*. *Journal of Analytical and Applied Pyrolysis*, 2013. **101**: p. 177-184.
132. Greenhalf, C.E., et al., *Thermochemical characterisation of straws and high yielding perennial grasses*. *Industrial Crops and Products*, 2012. **36**(1): p. 449-459.
133. Chen, Z., et al., *Characteristics and kinetic study on pyrolysis of five lignocellulosic biomass via thermogravimetric analysis*. *Bioresource Technology*, 2015. **192**: p. 441-450.
134. Cai, J., et al., *A distributed activation energy model for the pyrolysis of lignocellulosic biomass*. *Green Chemistry*, 2013. **15**(5): p. 1331-1340.
135. BRAR, G.S., *DEVELOPMENT AND COMMISSIONING OF AN OXIDATION REACTOR FOR TOTAL OXIDATION OF PYROLYSIS GAS*, in *DEPARTMENT OF CIVIL ENGINEERING*. 2015, INDIAN INSTITUTE OF TECHNOLOGY, DELHI.
136. König, J., *Die Nahrungsmittel, Genußmittel und Gebrauchsgegenstände, ihre Gewinnung, Beschaffenheit und Zusammensetzung*. 2013: Springer Berlin Heidelberg.
137. KG, C.R.G.C., *Xylan aus Maiskolben - Sicherheitsdatenblatt*.
138. Muley, P.D., et al., *A critical comparison of pyrolysis of cellulose, lignin, and pine sawdust using an induction heating reactor*. *Energy Conversion and Management*, 2016. **117**: p. 273-280.
139. Kaps, P. and G. Wanner, *A study of Rosenbrock-type methods of high order*. *Numerische Mathematik*, 1981. **38**(2): p. 279-298.
140. *boost - getting started*. [cited 2022 32.01.2022]; Available from: [https://www.boost.org/doc/libs/1\\_60\\_0/libs/numeric/odeint/doc/html/boost\\_numeric\\_odeint/getting\\_started/overview.html](https://www.boost.org/doc/libs/1_60_0/libs/numeric/odeint/doc/html/boost_numeric_odeint/getting_started/overview.html).
141. Hardy, B., J. De Wilde, and G. Winckelmans, *A penalization method for the simulation of weakly compressible reacting gas-particle flows with general boundary conditions*. *Computers & Fluids*, 2019. **190**: p. 294-307.
142. Curtis, L.J. and D.J. Miller, *Transport model with radiative heat transfer for rapid cellulose pyrolysis*. *Industrial & Engineering Chemistry Research*, 1988. **27**(10): p. 1775-1783.

143. Miller, R.S. and J. Bellan, *Analysis of Reaction Products and Conversion Time in the Pyrolysis of Cellulose and Wood Particles*. Combustion Science and Technology, 1996. **119**(1-6): p. 331-373.
  144. Papadikis, K., S. Gu, and A.V. Bridgwater, *CFD modelling of the fast pyrolysis of biomass in fluidised bed reactors: Modelling the impact of biomass shrinkage*. Computational Materials Science, 2009. **149**(1-3): p. 417-427.
  145. Koufopoulos, C.A., et al., *Modelling of the pyrolysis of biomass particles. Studies on kinetics, thermal and heat transfer effects*. The Canadian Journal of Chemical Engineering, 1991. **69**(4): p. 907-915.
  146. Marzouk, O., *Assessment of Three Databases for the NASA Seven-Coefficient Polynomial Fits for Calculating Thermodynamic Properties of Individual Species*. International Journal of Aeronautical Science & Aerospace Research, 2018: p. 150-163.
  147. Anca-Couce, A., et al., *Biomass pyrolysis TGA assessment with an international round robin*. Fuel, 2020. **276**: p. 118002.
  148. Radl, S., et al., *PARSCALE -AN OPEN-SOURCE LIBRARY FOR THE SIMULATION OF INTRA-PARTICLE HEAT AND MASS TRANSPORT PROCESSES IN COUPLED SIMULATIONS*. 2015.
-

# Veröffentlichungen

## Publikationen

- Funke, A., Grandl, R., Ernst, M., Dahmen, N.,  
**“Modelling and improvement of heat transfer coefficient in auger type reactors for fast pyrolysis application”**,  
*Chemical Engineering and Processing - Process Intensification*, 2018
- Kruse, A., Grandl, R.,  
**“Hydrothermal carbonization: 3. Kinetic model | Hydrothermale Karbonisierung: 3. Kinetisches Modell“**  
*Chemie-Ingenieur-Technik*, 2015
- Schwiderski, M., Kruse, A., Grandl, R., Dockendorf, D.  
**“Kinetics of the AlCl<sub>3</sub>catalyzed xylan hydrolysis during Methanosolv pulping of beech wood”**  
*RSC Advances*, 2014
- Schwiderski, M., Kruse, A., Grandl, R., Dockendorf, D.,  
**“Comparison of the influence of a Lewis acid AlCl<sub>3</sub> and a Brønsted acid HCl on the organosolv pulping of beech wood”**  
*Green Chemistry*, 2014
- Kruse, A., Badoux, F., Grandl, R., Wüst, D.,  
**“Hydrothermal carbonization: 2. Kinetics of draff conversion | Hydrothermale Karbonisierung: 2. Kinetik der Biertreber-Umwandlung”**  
*Chemie-Ingenieur-Technik*, 2012

## Poster

- Grandl, Robert; Funke, Axel; Dahmen, Nicolaus; Sauer, Jörg  
**“Fast pyrolysis of biomass using a twin screw mixing reactor: Simulation of the mixing behaviour for characterisation of the mass- and energy transport under respect to the insertion order”**  
*DGMK Fachbereichstagung Konversion von Biomassen und Kohlen, Rothenburg a.d. Fulda*, 9.-11.Mai 2016
- Robert Grandl, Axel Funke, Nicolaus Dahmen, Jörg Sauer  
**“Simulation of the particle flow in a twin-screw mixing reactor for fast pyrolysis of the biomass residues”**  
*ESCRE 2015 - European Symposium on Chemical Reaction Engineering*, 27. - 30. Oktober 2015
- Robert Grandl, Axel Funke, Nicolaus Dahmen, Jörg Sauer  
**“Fast pyrolysis of biomass using a twin screw mixing reactor: Simulation of the mixing behaviour of different geometries for optimization of the mass- and energy transport”**  
*CASCATBEL - Workshop on Thermochemical Lignocellulose Conversion Technologies*, 31. Mai 2016
- Robert Grandl, Axel Funke, Nicolaus Dahmen, Jörg Sauer  
**“Fast pyrolysis of biomass using a twin screw mixing reactor: Simulation of the mixing behaviour for characterisation of the mass- and energy transport”**  
*Jahrestreffen Reaktionstechnik und Mischvorgänge*, 2.-4. Mai 2016
- Robert Grandl, Axel Funke, Nicolaus Dahmen, Jörg Sauer  
**“Characterisation of the twin screw mixing reactor used for fast pyrolysis of biomass”**  
*25th European Biomass Conference and Exhibition (EUBCE), Stockholm*, 12-15 Juni 2017
- Robert Grandl, Axel Funke, Jörg Sauer, Nicolaus Dahmen  
**“Stoff- und Energietransport in einem Doppelschneckenreaktor**

**- Eine Numerische Betrachtung”**

*Tag der Fakultät (ciw), 2015*

**Vorträge**

- Robert Grandl

**“Schnellpyrolyse von Biomasse im Doppelschneckenmischreaktor: Lagrange-Euler-Simulation zur Charakterisierung von Stoff- und Energietransportvorgängen“**

*Jahrestreffen der ProcessNet-Fachgruppen Mehrphasenströmungen, Partikelmesstechnik, Zerkleinern und Klassieren, Computational Fluid Dynamics, Mischvorgänge und dem TAK Aerosoltechnologie, 14. – 17. März 2017*

**Preise**

- Posterpreis – Fachbereich Mischvorgänge: *Jahrestreffen Reaktionstechnik und Mischvorgänge*, 2.-4. Mai 2016