Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik

Diplomarbeit

Optimierung und parallele Berechnung von Strömungen in gekrümmten dreidimensionalen Koordinaten

Michael Meyer

19. April 2001

Inhaltsverzeichnis

Kapitel 1

Einleitung

Strömungsprobleme und damit verbundene Optimierungsaufgaben treten in verschiedenen technischen Bereichen auf. So ist es zum Beispiel wünschenswert, Fahr- und Fluggeräten aller Art (Schiffen, Autos, Flugzeugen etc.), die sich in einem "strömenden" Medium (Wasser oder Luft) bewegen, möglichst günstige Umströmungseigenschaften zu geben, da sich dies unmittelbar auf zum Beispiel Energieverbrauch oder Lärmentwicklung auswirkt. Auch bei der Konstruktion von Turbinenschaufeln, Kühleinrichtungen oder hydraulischen Geräten stellen sich ähnliche Fragen. Der naheliegende Weg, gewisse Strömungseigenschaften zu bestimmen, ist der Versuch, also der physikalische Aufbau der entsprechenden Konfiguration und die Messung der entsprechenden Parameter. Dieser Weg ist jedoch in den meisten Fällen nicht praktikabel, denn

- physikalische Versuche dieser Art sind in der Regel sehr teuer: man betrachte nur den Aufwand, der für den Bau eines Windkanals getrieben werden muß;
- die Messung verschiedener Parameter ist oft nicht oder nur sehr schwer möglich, zum Beispiel das Strömungsverhalten innerhalb eines Tintenstrahldruckkopfes oder hinter einem Düsenjet;
- bei der Optimierung muß man die verwendeten Modelle variieren, um eine Veränderung feststellen zu können. Dies erhöht den Aufwand noch einmal extrem.

Daher ist man bestrebt, so weit wie möglich Strömungsprobleme physikalisch zu modellieren und mit Hilfe der immer größer und billiger werdenden Rechenleistung, die durch den Computer zur Verfügung gestellt wird, zu simulieren. Erst in späteren Entwicklungsphasen, wo die Genauigkeit der Simulation noch nicht ausreichend ist, wird auf "konventionelle" Methoden zurückgegriffen.

Strömungen werden mathematisch beschrieben durch ein System partieller Differentialgleichungen, die sogenannten Navier-Stokes-Gleichungen. Obwohl diese Gleichungen schon etwa 170 Jahre bekannt sind, ist eine analytische Lösung bisher nur in wenigen, sehr einfachen Spezialfällen gelungen. Daher ist es notwendig, diese Gleichungen zu *diskretisieren*, d.h. den unendlichdimensionalen Raum der potentiellen

Lösungen durch einen endlichdimensionalen zu ersetzen. Bei dem Finite-Volumen-Verfahren, welches in dieser Arbeit benutzt wird, geschieht dies durch Zerlegung des durchströmten Gebiets in viele kleine Kontrollvolumen (Zellen). In diesem endlichdimensionalen Raum wird dann mit Computerhilfe eine Näherungslösung für die Navier-Stokes-Gleichungen bestimmt.

Dieses Verfahren zur Strömungsberechnung wird nun zur Optimierung benutzt. Das hier verwendete Optimierungsverfahren – eine Variante des SQP-Verfahrens – arbeitet iterativ und nähert sich so schrittweise der Lösung. In jedem Schritt sind dabei mehrere jeweils leicht modifizierte Strömungsprobleme zu lösen, die zu einem großen Teil *parallel* abgearbeitet werden können.

Auch für die Strömungsberechnung ist eine parallele Berechnung notwendig: Das Resultat der Diskretisierung ist ein extrem großes, mit Computerhilfe lösbares Gleichungssystem. Die Genauigkeit dieser Approximation hängt im wesentlichen von zwei Faktoren ab, sie steigt zum einen mit der Anzahl der Zellen, zum anderen mit der Qualität der Diskretisierung.

Die Anzahl der Zellen hat direkte Auswirkungen auf die Größe des resultierenden Gleichungssystems, da für jede neue Zelle eine neue Gleichung eingefügt wird. Für die realistische Simulation turbulenter Strömungen oder Strömungen in komplexen Geometrien sind mehrere Millionen Zellen notwendig. Es ist leicht einzusehen, daß ein einzelner Computer mit der Aufgabe der Lösung dieses Gleichungssystems überfordert wäre. Daher ist es notwendig das Problem auf mehrere Rechner zu verteilen, die gleichzeitig *parallel* an dem Problem rechnen und die notwendigen Daten austauschen.

Die Qualität der Diskretisierung wird einerseits durch die Art und Weise beeinflußt, wie Differentialoperatoren auf dem endlichdimensionalen Lösungsraum dargestellt werden. Dies wird mit der *Ordnung* der Diskretisierung gemessen. Andererseits spielt natürlich die Wahl dieses Raumes selbst eine wesentliche Rolle. Die einfachste Möglichkeit, das durchströmte Gebiet in Zellen zu zerlegen ist – im dreidimensionalen Fall – Würfel zu verwenden. Dies hat auch die "einfachsten" Diskretisierungen zur Folge. In der Praxis hat man es allerdings mit Geometrien zu tun, die nicht treppenförmig sondern krummlinig begrenzt sind. Die Verwendung *gekrümmter Koordinaten* er-

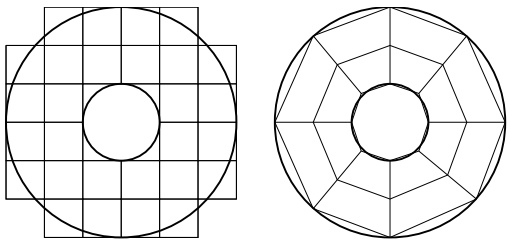

Abbildung 1.1: Diskretisierung in rechteckigen und gekrümmten Koordinaten

möglicht sowohl die genaue Approximation des Randes (Abbildung 1.1), als auch die Verwendung einer feineren Auflösung in Bereichen, in denen die Lösung stark variiert, was ihre Qualität erheblich steigern kann.

Die Berechnung von Strömungen (“computational fluid dynamics”, CFD) ist ein recht breites Forschungsfeld. Dementsprechend finden sich große Mengen an Literatur und viele verschiedene numerische Verfahren zur Behandlung dieses Problems. Einen Überblick über Verfahren zur Berechnung inkompressibler Strömungen, wie sie in dieser Arbeit behandelt werden, findet sich zum Beispiel in [Tur99]. Für Strömungsprobleme allgemeinerer Art sei auf [PT83, Fle90] verwiesen.

Es gibt bisher nicht *den* universalen Algorithmus zur effizienten Behandlung aller Strömungsprobleme. Für gewisse Problemklassen gibt es zum Teil recht effiziente Methoden. Diese sind aber in der Regel relativ komplex und daher – sollen sie flexibel genug und nicht zu speziell sein – schwierig zu implementieren. Solche Methoden sollen in dieser Arbeit nicht behandelt werden. Es soll ein relativ einfaches Verfahren implementiert werden, welches die Behandlung eines breiten Spektrums verschiedener Strömungsprobleme erlaubt und einfach erweitert werden kann.

Verschiedene Ansätze zur Optimierung von Strömungen finden sich in [GPP94, GHS93, HK98]. Auch im Optimierungsteil soll aber zunächst ein einfaches Verfahren implementiert werden, bei dem Optimierungs- und Strömungslösungsprozess getrennt voneinander behandelt werden, der Strömungslöser sozusagen als “Black Box” betrachtet wird. Ein Ansatz aus den genannten Arbeiten, der beide Prozesse miteinander verbindet, wird in Kapitel 6 diskutiert.

Ziel dieser Diplomarbeit ist die Entwicklung eines Programms zur parallelen Berechnung von Strömungen durch dreidimensionale Gebiete in gekrümmten Koordinaten und darauf aufbauend die Anwendung eines Verfahrens zur Optimierung des Randes des durchströmten Gebietes.

Schließlich möchte ich mich an dieser Stelle bei allen bedanken, die an der Entstehung dieser Arbeit mitgewirkt haben. Mein besonderer Dank gilt dabei meinen Betreuern Prof. Dr. Michael Griebel und Frank Koster für ihre kompetente Beratung und Motivation. Weiterhin möchte ich mich bei allen Korrekturlesern bedanken, die sich trotz der vielen Formeln durch den Text gearbeitet haben.

Überblick

Kapitel 2 behandelt die mathematischen Anteile der Strömungsberechnung: die Diskretisierung der Navier-Stokes-Gleichungen in Raum und Zeit, Lösungsverfahren für die entstehenden linearen Gleichungssysteme, Verfahren zur Generierung der Gitter sowie Grundlegendes zu NURBS-Flächen, die zur Parametrisierung der Ränder verwendet werden.

Kapitel 3 beschreibt zunächst Optimierungsprobleme und zugehörige Optimalitätskriterien, um dann ausgehend von Verfahren zur unrestringierten Optimierung die Grundzüge des verwendeten Verfahrens zur nichtlinearen restringierten Optimierung vorzustellen.

Kapitel 4 gibt einen Überblick über verschiedene Details der Implementierung, insbesondere die Methode der blockstrukturierten Gitter und verschiedene Aspekte der Parallelisierung. Weiterhin wird eine Übersicht über die verwendeten Softwarewerkzeuge und die Klassenstruktur der eigenen Implementierung gegeben.

Kapitel 5 enthält die Ergebnisse verschiedener numerischer Beispiele sowohl für Strömungs- als auch für Optimierungsrechnungen.

Kapitel 6 faßt die Ergebnisse dieser Arbeit zusammen und gibt einen Überblick über verschiedene Punkte, an denen eine Weiterentwicklung der Verfahren möglich ist.

Kapitel 2

Strömungsberechnung

Strömungen werden beschrieben durch ein System partieller Differentialgleichungen, die sogenannten *Navier-Stokes-Gleichungen*. In dieser Diplomarbeit werden nur instationäre Strömungen für *inkompressible* Fluide, d.h. Fluide mit konstanter Dichte, behandelt. Diese umfassen Flüssigkeiten und (näherungsweise) Gase bei niedrigen Geschwindigkeiten. Stationäre Strömungen können durch Iteration in einen stationären Zustand behandelt werden, d.h. die Strömung wird so lange instationär berechnet, bis sie sich in einen stationären Zustand entwickelt hat.

Eine Strömung in einem Gebiet $\Omega \subset \mathbb{R}^3$ ist charakterisiert durch die *Geschwindigkeit* $\boldsymbol{u}(\boldsymbol{x}, t) = (u^1, u^2, u^3, t)$ und den *Druck* $p(\boldsymbol{x}, t)$ für jeden Ort $\boldsymbol{x} \in \Omega$ und Zeit $t \in \mathbb{R}$. Die Navier-Stokes-Gleichungen ergeben sich aus den Erhaltungsgesetzen für Masse und Impuls. Diese lauten in Integralform für ein Kontrollvolumen $V \subset \Omega$ mit Rand $\Gamma = \partial V$ und äußerem Normalenvektor $\boldsymbol{n}$ (siehe [PT83, Len89])

$$\begin{aligned} \frac{d}{dt}\int_V \rho \, dV + \int_{\partial V} \rho(\boldsymbol{u}\cdot\boldsymbol{n}) \, d\Gamma &= 0, \\ \frac{d}{dt}\int_V \rho\boldsymbol{u} \, dV + \int_{\partial V} \left[(\boldsymbol{u}\cdot\boldsymbol{n})\rho\boldsymbol{u} - \boldsymbol{n}\sigma\right] d\Gamma &= \int_V \boldsymbol{f}_e \, dV. \end{aligned}$$

Hierbei ist ρ die Dichte des Fluids, die im Weiteren als konstant angenommen wird, $\boldsymbol{f}_e$ eine äußere Kraft, die als Null angenommen wird, und $\sigma = -p\boldsymbol{I} + \boldsymbol{\tau}$ der Spannungstensor mit $\boldsymbol{\tau} = \lambda(\nabla\cdot\boldsymbol{u})\boldsymbol{I} + \mu\left[\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T\right]$ und Viskositätskoeffizienten λ und μ, für die $3\lambda + 2\mu = 0$ angenommen wird. Wird schließlich noch die kinematische Zähigkeit μ des Fluids als konstant angenommen, so ergeben sich die Gleichungen (vgl. [PT83])

$$\begin{aligned} \nabla\cdot\boldsymbol{u} &= 0 \\ \rho\left[\frac{\partial\boldsymbol{u}}{\partial t} + \nabla\cdot(\boldsymbol{u}\otimes\boldsymbol{u})\right] + \nabla p - \mu\nabla^2\boldsymbol{u} &= 0. \end{aligned}$$

Um diese Gleichungen in eine dimensionslose Form zu überführen, wähle eine Referenzlänge l und eine Referenzgeschwindigkeit u. Dann lauten die Navier-Stokes-

Gleichungen im Gebiet Ω mit *Reynolds-Zahl* Re

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\nabla p + \frac{1}{\mathrm{Re}} \nabla^2 \boldsymbol{u} \tag{2.1}$$

$$\operatorname{div} \boldsymbol{u} = 0. \tag{2.2}$$

Die Reynolds-Zahl beschreibt die Zähigkeit des Fluids und ergibt sich durch $\mathrm{Re} = u \cdot l \cdot \rho/\mu$. Zu den Gleichungen (2.1) und (2.2) kommen verschiedene Randbedingungen auf $\partial\Omega$, dem Rand des Gebiets: Haft-, Gleit-, Ein- und Auslassbedingungen (siehe Kapitel 2.1.6).

Im Rahmen der Behandlung der diskretisierten Navier-Stokes-Gleichungen treten weitere Teilprobleme auf: Zum einen müssen sehr große lineare Gleichungssysteme gelöst werden, zum anderen muß das Gebiet in Zellen zerlegt, also ein Gitter über dem Gebiet generiert werden. Diese Probleme werden in den Kapiteln 2.2 und 2.3 behandelt. Zur Beschreibung der Gebietsränder selbst werden NURBS-Flächen verwendet, dazu mehr in Kapitel 2.4.

2.1 Diskretisierung der Navier-Stokes-Gleichungen in gekrümmten Koordinaten

Die Zerlegung des (i.A. nicht rechteckigen) Gebiets Ω in Zellen, also das Anlegen eines *Gitters* auf Ω, kann auf viele verschiedene Weisen erfolgen. Zunächst kann man zwei verschiedene Ansätze, nämlich strukturierte und unstrukturierte Gitter, unterscheiden. Strukturierte Gitter zeichnen sich dadurch aus, daß jeder innere Gitterpunkt Eckpunkt einer gleichen Anzahl von Zellen ist; man stelle sich im einfachsten Fall ein Schachbrett vor. Dies hat den Vorteil sehr einfacher und damit effizienter Datenstrukturen (Matrizen), während unstrukturierte Gitter ($\rightarrow$FEM) flexibler in der Verteilung der Zellen sind, die zu verwendenden Datenstrukturen aber eine größere Komplexität aufweisen (Listen, Hash-Tabellen, etc.). Ein weiterer Vorteil gekrümmter strukturierter Gitter liegt darin, daß das Gitter automatisch dem Verlauf des Randes und damit in der Regel auch dem Verlauf der Strömung angepaßt ist. Dies liefert genauere Ergebnisse.

Der hier verwendete Ansatz der *blockstrukturierten Gitter* ist eine Mischung aus beiden Formen: Das Gebiet wird zunächst in mehrere einfache Gebiete (Blöcke) zerlegt, die dann mit einem strukturierten Gitter versehen und an den Rändern miteinander verbunden werden (Details siehe Kapitel 4). In diesem Kapitel wird nur die Diskretisierung auf einem einzelnen Block behandelt.

Die hier durchgeführte Diskretisierung folgt in weiten Teilen [WSKB97] und ist eine Verallgemeinerung eines "*staggered mesh*" (siehe [HW65]).

2.1.1 Die Transformationsabbildung

Es wird von einem Gitter ausgegangen, das (topologisch) einem aus Einheitswürfeln gebauten Quader $G = [0, n^1] \times [0, n^2] \times [0, n^3]$ entspricht. Die Zellen Ω_j sind Trans-

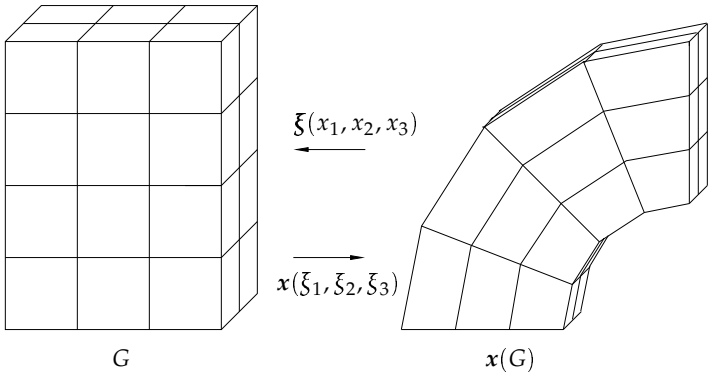

Abbildung 2.1: Transformation eines Quaders

formationen solcher Würfel (Abbildung 2.1). Die Transformation leistet die Abbildung $\boldsymbol{x} : G \to \Omega = \boldsymbol{x}(G) \subset \mathbb{R}^3$. Sie wird nur auf den Eckpunkten der Würfel vorgegeben und mittels trilinearer Interpolation auf den Rest des Gebietes fortgesetzt. Weiterhin muß die Abbildung $\boldsymbol{x}$ die folgenden drei Bedingungen erfüllen:

1. die Abbildung $\boldsymbol{x}$ ist bijektiv;
2. die Jacobideterminante J ist überall positiv, wobei
$$J := \det\Big(\frac{\partial \boldsymbol{x}}{\partial \xi^1}, \frac{\partial \boldsymbol{x}}{\partial \xi^2}, \frac{\partial \boldsymbol{x}}{\partial \xi^3}\Big) = \frac{\partial \boldsymbol{x}}{\partial \xi^1} \cdot \Big(\frac{\partial \boldsymbol{x}}{\partial \xi^2} \times \frac{\partial \boldsymbol{x}}{\partial \xi^3}\Big); \tag{2.3}$$
3. der Rand ∂G von G wird abgebildet auf den Rand $\partial\Omega$ von Ω.

Um in diesen transformierten Koordinaten Differentialrechnung betreiben zu können, werden noch einige geometrische Größen benötigt. Definiere die *kovarianten* und die *kontravarianten Basisvektoren* durch

$$\boldsymbol{a}_{(\alpha)} = \frac{\partial \boldsymbol{x}}{\partial \xi^\alpha} \quad \text{bzw.} \quad \boldsymbol{a}^{(\alpha)} = \nabla \xi^\alpha, \quad 1 \le \alpha \le 3, \tag{2.4}$$

wobei $\xi(x^1, x^2, x^3)$ die inverse Abbildung zu $\boldsymbol{x}$ ist. Es gilt (siehe [Kön93])

$$\boldsymbol{a}^{(\alpha)} \cdot \boldsymbol{a}_{(\beta)} = \delta_{\alpha\beta}. \tag{2.5}$$

Es folgt (siehe [BSMM95])

$$\boldsymbol{a}^{(\alpha)} = \frac{1}{\sqrt{g}}(\boldsymbol{a}_{(\beta)} \times \boldsymbol{a}_{(\gamma)}), \qquad \sqrt{g} = J = \boldsymbol{a}_{(\alpha)} \cdot (\boldsymbol{a}_{(\beta)} \times \boldsymbol{a}_{(\gamma)}),$$

wobei (α, β, γ) zyklische Vertauschung von $(1, 2, 3)$ und $\sqrt{g}$ die Jacobideterminante[1] (2.3) ist.

[1] Diese wird auch *Gramsche Determinante* genannt, daher die Bezeichnung $\sqrt{g}$.

Damit läßt sich die Ableitung einer Funktion $\Phi(\boldsymbol{x})$ in $\boldsymbol{x}$-Koordinaten zurückführen auf Ableitungen in ξ-Koordinaten (Beweis siehe Anhang A.1.1):

$$\frac{\partial \Phi}{\partial x^\alpha} = \sum_{\beta=1}^{3} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^\beta} (\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \Phi), \quad 1 \leq \alpha \leq 3. \tag{2.6}$$

Weiterhin definiere den *kovarianten* und *kontravarianten metrischen Tensor* g_{ij} bzw. g^{ij} durch

$$g_{ij} = \boldsymbol{a}_{(i)} \cdot \boldsymbol{a}_{(j)} \qquad \text{bzw.} \qquad g^{ij} = \boldsymbol{a}^{(i)} \cdot \boldsymbol{a}^{(j)}. \tag{2.7}$$

2.1.2 Berechnung der geometrischen Größen

Die im letzten Abschnitt definierten geometrischen Größen werden an verschiedenen Stellen einer Zelle benötigt. Daher ist es notwendig, diese Stellen genauer zu bezeichnen. Wie schon erwähnt, wird der Quader G in Einheitswürfel zerlegt. Die Eckpunkte liegen auf ganzzahligen Koordinaten, also haben die Mittelpunkte die "halben" Koordinaten $\boldsymbol{j}$, wobei

$$\boldsymbol{j} \in \left\{ \left(j^1 + \frac{1}{2}, j^2 + \frac{1}{2}, j^3 + \frac{1}{2}\right) \mid 0 \leq j^\alpha < n^\alpha, j^\alpha \in \mathbb{N}, 1 \leq \alpha \leq 3 \right\} =: G_h.$$

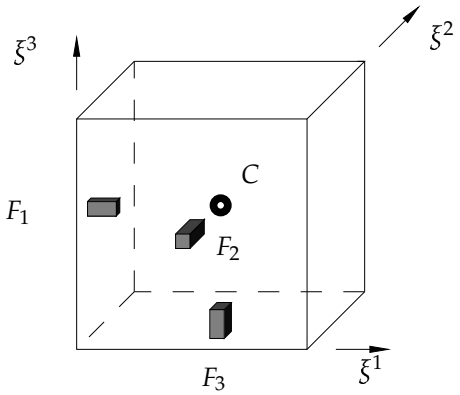

Abbildung 2.2: Zellmittelpunkt C und Flächenmittelpunkte F_1, F_2, F_3

Seien $e_1 = (\frac{1}{2}, 0, 0)$, $e_2 = (0, \frac{1}{2}, 0)$ und $e_3 = (0, 0, \frac{1}{2})$, dann befinden sich die Flächenmittelpunkte F_i in Abbildung 2.2 an den Positionen $\boldsymbol{j} - e_i$. Kantenmittelpunkte befinden sich an den Positionen $\boldsymbol{j} \pm e_i \pm e_j$, $i \neq j$, während Ecken sich in $\boldsymbol{j} \pm e_1 \pm e_2 \pm e_3$ befinden. Letztere werden für eine Zelle wie in Abbildung 2.3 numeriert. Weiterhin werden Variablen, die an bestimmten Orten ausgewertet werden sollen, mit einem entsprechenden Ortsindex versehen (z.B. $\boldsymbol{u}_{j+e_1}$, p_j).

Die Abbildung $\boldsymbol{x}$ ist nur in den Ecken $\boldsymbol{j} \pm e_1 \pm e_2 \pm e_3$, $\boldsymbol{j} \in G_h$, vorgegeben durch $\boldsymbol{x}(\boldsymbol{j} \pm e_1 \pm e_2 \pm e_3) := \boldsymbol{x}_{\boldsymbol{j} \pm e_1 \pm e_2 \pm e_3}$ und wird dazwischen mittels trilinearer Interpolation

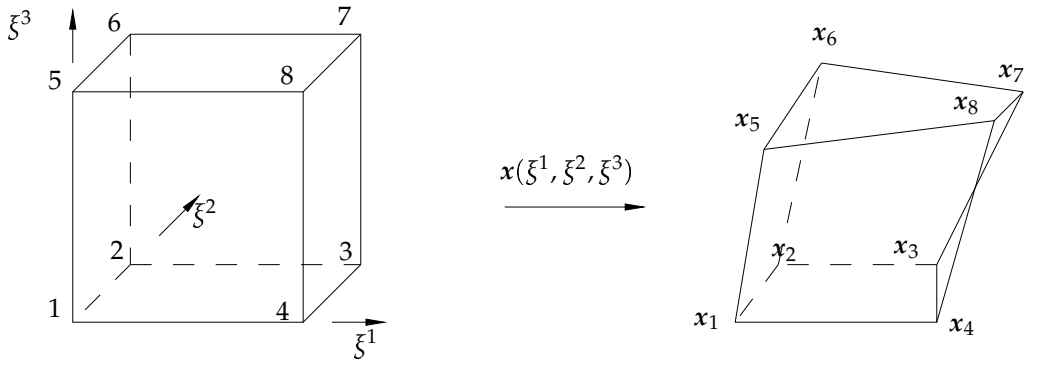

Abbildung 2.3: Eckpunkte einer Zelle

fortgesetzt. Daher ist $\boldsymbol{x}$ innerhalb einer Zelle differenzierbar (aber i.A. nicht auf dem Rand einer Zelle) und die kovarianten Basisvektoren können in Zellmittelpunkten C berechnet werden. Diese und alle weiteren Berechnungen zu diesem Abschnitt finden sich in Anhang A.1.2. Es gilt mit $\boldsymbol{x}_{ijkl} := \boldsymbol{x}_i + \boldsymbol{x}_j + \boldsymbol{x}_k + \boldsymbol{x}_l$

$$\begin{aligned} \boldsymbol{a}_{(1),C} &= \frac{1}{4}(\boldsymbol{x}_{3478} - \boldsymbol{x}_{1256}), \\ \boldsymbol{a}_{(2),C} &= \frac{1}{4}(\boldsymbol{x}_{2367} - \boldsymbol{x}_{1458}), \\ \boldsymbol{a}_{(3),C} &= \frac{1}{4}(\boldsymbol{x}_{5678} - \boldsymbol{x}_{1234}). \end{aligned} \tag{2.8}$$

Ebenso ergibt sich mit $\boldsymbol{x}_{ij} := \boldsymbol{x}_i + \boldsymbol{x}_j$

$$\begin{array}{lll} & \boldsymbol{a}_{(2),F_1} = \frac{1}{2}(\boldsymbol{x}_{26} - \boldsymbol{x}_{15}), & \boldsymbol{a}_{(3),F_1} = \frac{1}{2}(\boldsymbol{x}_{56} - \boldsymbol{x}_{12}) \\ \boldsymbol{a}_{(1),F_2} = \frac{1}{2}(\boldsymbol{x}_{48} - \boldsymbol{x}_{15}), & & \boldsymbol{a}_{(3),F_2} = \frac{1}{2}(\boldsymbol{x}_{58} - \boldsymbol{x}_{14}) \\ \boldsymbol{a}_{(1),F_3} = \frac{1}{2}(\boldsymbol{x}_{34} - \boldsymbol{x}_{12}), & \boldsymbol{a}_{(2),F_3} = \frac{1}{2}(\boldsymbol{x}_{23} - \boldsymbol{x}_{14}). & \end{array} \tag{2.9}$$

Bei den fehlenden Einträgen ist $\boldsymbol{x}$ in der entsprechenden Richtung i.A. nicht differenzierbar. Diese Ableitungen existieren zwar für eine Zelle, müssen aber für zwei benachbarte Zellen in dem gemeinsamen Punkt F_i nicht gleich sein. Da aber alle kovarianten Basisvektoren benötigt werden, um die kontravarianten Basisvektoren mittels (2.5) zu bestimmen, sind letztere in den Flächenmittelpunkten nicht eindeutig. An diesen Stellen existiert aber (eindeutig)

$$\sqrt{g}\boldsymbol{a}^{(\alpha)} = \boldsymbol{a}_{(\beta)} \times \boldsymbol{a}_{(\gamma)}$$

und kann mit $\boldsymbol{s}_{ijkl} := \frac{1}{2}(\boldsymbol{x}_j - \boldsymbol{x}_l) \times (\boldsymbol{x}_k - \boldsymbol{x}_i)$ berechnet werden durch

$$(\sqrt{g}\boldsymbol{a}^{(1)})_{F_1} = \boldsymbol{s}_{1265}, \quad (\sqrt{g}\boldsymbol{a}^{(2)})_{F_2} = \boldsymbol{s}_{1584}, \quad (\sqrt{g}\boldsymbol{a}^{(3)})_{F_3} = \boldsymbol{s}_{1432}.$$

Wegen dieser Eindeutigkeitseigenschaft sollte also statt $\boldsymbol{a}^{(\alpha)}$ die Größe $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ verwendet werden. Anschaulich ist $\boldsymbol{s}_{ijkl}$ der Normalenvektor auf der entsprechenden Fläche mit Länge gleich dem Oberflächenvolumen (Flächeninhalt) dieser Fläche.

Als weitere Größe wird noch das Volumen $|\Omega_j|$ einer Zelle Ω_j benötigt. Dieses kann berechnet werden durch

$$|\Omega_j| = \frac{1}{3}\big(\boldsymbol{b}_1 \cdot (\boldsymbol{s}_{1265} + \boldsymbol{s}_{4378}) + \boldsymbol{b}_2 \cdot (\boldsymbol{s}_{1584} + \boldsymbol{s}_{2673}) + \boldsymbol{b}_3 \cdot (\boldsymbol{s}_{1432} + \boldsymbol{s}_{8765})\big),$$

wobei

$$\begin{array}{rclcrcl} \boldsymbol{b}_1 & = & \frac{1}{8}(\boldsymbol{x}_{3478} - \boldsymbol{x}_{1256}), & \boldsymbol{b}_2 & = & \frac{1}{8}(\boldsymbol{x}_{2367} - \boldsymbol{x}_{1458}), \\ \boldsymbol{b}_3 & = & \frac{1}{8}(\boldsymbol{x}_{5678} - \boldsymbol{x}_{1234}). & & & & \end{array}$$

2.1.3 Diskretisierung der Kontinuitätsgleichung

Die Kontinuitätsgleichung (2.2) beschreibt das Prinzip der Masseerhaltung. Integriert man diese nach dem Prinzip der *finiten Volumen* über ein Kontrollvolumen Ω_j (Abbildung 2.2), so ergibt sich nach dem Gaußschen Integralsatz unter Verwendung der in diesem Kapitel gültigen Konvention $\Phi|_i^j = \Phi_j - \Phi_i$

$$0 = \int_{\Omega_j} \operatorname{div} \boldsymbol{u}\, d\Omega = \int_{\partial\Omega_j} \boldsymbol{u} \cdot \boldsymbol{n}\, d\Gamma \approx \sum_{\alpha=1}^{3} (\boldsymbol{u} \cdot \boldsymbol{s})|_{j-e_\alpha}^{j+e_\alpha} = \sum_{\alpha=1}^{3} (\boldsymbol{u} \cdot \sqrt{g}\boldsymbol{a}^{(\alpha)})|_{j-e_\alpha}^{j+e_\alpha} \tag{2.10}$$

wobei $\boldsymbol{s}$ die im letzten Abschnitt definierten Oberflächennormalenvektoren sind. Die Größe

$$V^\alpha := \sqrt{g}\boldsymbol{a}^{(\alpha)} \cdot \boldsymbol{u}$$

ist gerade der (Volumen-)fluß über die entsprechende Seitenfläche. Wählt man die V^α als die die Geschwindigkeit repräsentierenden Größen, so wird das Prinzip der Masseerhaltung exakt eingehalten. Diese Wahl entspricht genau der des "*staggered mesh*", bei dem die Geschwindigkeiten in den Seitenflächen und der Druck im Zentrum der Zelle gespeichert werden (Abbildung 2.2: Druck in C und Flüsse (Geschwindigkeiten) in F_i). Dies legt nahe, den Druck in den Zellmittelpunkten auszuwerten. Mit dieser Wahl der Variablen lautet die diskrete Kontinuitätsgleichung

$$0 = \sum_{\alpha=1}^{3} (V^\alpha_{j+e_\alpha} - V^\alpha_{j-e_\alpha}),\ \boldsymbol{j} \in G_h \quad \text{oder} \quad \boldsymbol{D}V = 0$$

mit dem dadurch definierten diskreten Divergenzoperator $\boldsymbol{D}$.

Nach (2.5) hängen $\boldsymbol{u}$ und V wie folgt zusammen:

$$V^\alpha = \sqrt{g}\boldsymbol{a}^{(\alpha)} \cdot \boldsymbol{u}, \qquad \boldsymbol{u} = \sum_{\alpha=1}^{3} \boldsymbol{a}_{(\alpha)} V^\alpha / \sqrt{g}. \tag{2.11}$$

Für die Diskretisierung der Impulsgleichung ist es notwendig, V^α und $\boldsymbol{u}$ nicht nur in Mittelpunkten der Seitenflächen sondern auch an anderen Stellen zu ermitteln. Aufgrund der oben erwähnten Uneindeutigkeit bestimmter geometrischer Größen ist bei

dieser Interpolation besondere Sorgfalt notwendig. Eine notwendige Forderung ist, daß für konstantes $\boldsymbol{u}$ dieses invariant unter der Transformation nach V^α und zurück ist, genauer: für die folgende Abbildungsfolge muß $\boldsymbol{u} = \tilde{\boldsymbol{u}}$ gelten.

$$\boldsymbol{u} \xrightarrow{(2.11)} V^\alpha_{j+e_\alpha} \xrightarrow{\text{Interpolation}} V^\alpha_\xi \xrightarrow{(2.11)} \tilde{\boldsymbol{u}}. \tag{2.12}$$

Die entsprechenden Interpolationsformeln finden sich in Anhang A.1.3. Abbildung 2.4

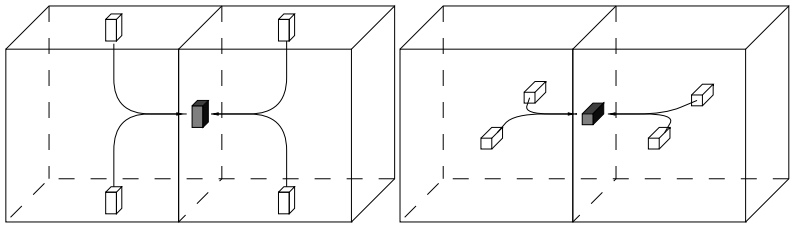

Abbildung 2.4: Interpolation in Flächenmittelpunkten

zeigt beispielhaft die einfließenden Werte für eine Interpolation in Flächenmittelpunkten. Für die Berechnung von V und $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ wird aus den umgebenden Punkten das arithmetische Mittel berechnet. Die Berechnung von $\boldsymbol{u}$ erfolgt dann im Flächenmittelpunkt mit (2.11).

2.1.4 Diskretisierung der Impulsgleichung

Auch hier erfolgt die Diskretisierung nach dem Prinzip der finiten Volumen. Zunächst wird die Impulsgleichung in eine äquivalente Form transformiert. Dann wird diese Gleichung über Kontrollvolumen Ω_{j+e_α} (Abbildung 2.5) integriert. In einem weiteren Schritt werden diese Integrale mittels der umgebenden Werte approximiert. Dies liefert für eine Zelle drei vektorwertige Größen, die in einem letzten Schritt zu drei skalaren Größen reduziert werden.

Die Impulsgleichung (2.1) kann man schreiben als

$$\frac{\partial \boldsymbol{u}}{\partial t} + \sum_{\alpha=1}^{3} \frac{\partial}{\partial x^\alpha}(u^\alpha \boldsymbol{u}) + \nabla p - \sum_{\alpha=1}^{3} \frac{1}{\mathrm{Re}} \frac{\partial \boldsymbol{e}^{(\alpha)}}{\partial x^\alpha} = 0 \tag{2.13}$$

mit

$$\boldsymbol{e}^{(\alpha)} = \begin{pmatrix} \frac{\partial u^1}{\partial x^\alpha} + \frac{\partial u^\alpha}{\partial x^1} \\ \frac{\partial u^2}{\partial x^\alpha} + \frac{\partial u^\alpha}{\partial x^2} \\ \frac{\partial u^3}{\partial x^\alpha} + \frac{\partial u^\alpha}{\partial x^3} \end{pmatrix}.$$

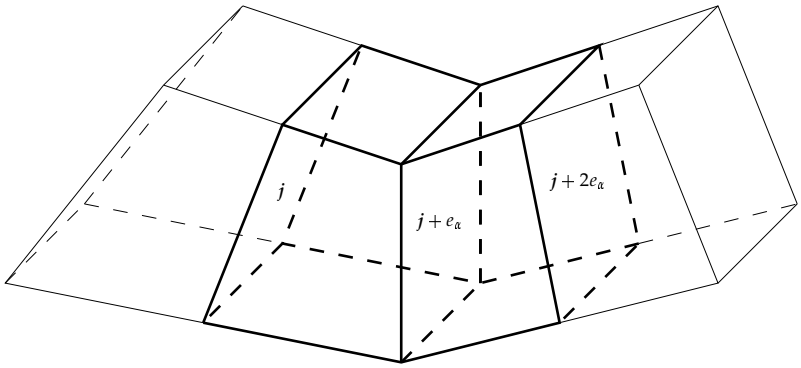

Abbildung 2.5: Verschobenes Kontrollvolumen Ω_{j+e_α}

Unter Verwendung von (2.6) werden in (2.13) Ableitungen in $\boldsymbol{x}$-Koordinaten durch Ableitungen in ξ-Koordinaten ersetzt:

$$\begin{aligned} 0 &= \frac{\partial \boldsymbol{u}}{\partial t} + \sum_{\alpha,\beta} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^\beta} (\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} u^\alpha \boldsymbol{u}) + \nabla p - \sum_{\alpha,\beta} \frac{1}{\mathrm{Re}} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^\beta} (\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \boldsymbol{e}^{(\alpha)}) \\ &= \frac{\partial \boldsymbol{u}}{\partial t} + \underbrace{\sum_{\beta} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^\beta} (V^\beta \boldsymbol{u})}_{\text{konvektiver Term}} + \nabla p - \underbrace{\sum_{\alpha,\beta} \frac{1}{\mathrm{Re}} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^\beta} (\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \boldsymbol{e}^{(\alpha)})}_{\text{diffusiver Term}} . \end{aligned} \tag{2.14}$$

Hierbei und im Weiteren wird folgende Konvention verwendet:

Wird über α, β oder γ summiert, so läuft die Summe über $\{1,2,3\}$. Die freien Variablen werden dann so gewählt, daß (α,β,γ) zyklische Vertauschung von $(1,2,3)$ ist, es ist also $(\alpha,\beta,\gamma) \in \{(1,2,3),(2,3,1),(3,1,2)\}$.

Integriere nun über ein verschobenes Kontrollvolumen (Abbildung 2.5)

$$\Omega_{j+e_\alpha} = \boldsymbol{x}\Big(\boldsymbol{j} + e_\alpha + \big[-\frac{1}{2}, \frac{1}{2}\big]^3\Big) =: \boldsymbol{x}(G_{j+e_\alpha}).$$

Für den ersten Summanden wird dieses Integral approximiert durch das Volumen multipliziert mit dem Wert im Zentrum des Kontrollvolumens:

$$\int_{\Omega_{j+e_\alpha}} \frac{\partial \boldsymbol{u}}{\partial t} \, d\Omega \approx |\Omega_{j+e_\alpha}| \frac{d}{dt} \boldsymbol{u}_{j+e_\alpha}, \quad \boldsymbol{j} \in G_h, 1 \le \alpha \le 3.$$

Dies liefert für jede Zelle drei vektorwertige Größen, für jede Zelle Ω_j jeweils aus den Kontrollvolumen Ω_{j+e_α} Diese werden nun durch Projektion auf $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ auf drei skalare Größen reduziert, also

$$\big(\sqrt{g}\boldsymbol{a}^{(\alpha)}\big)_{j+e_\alpha} \cdot |\Omega_{j+e_\alpha}| \frac{d}{dt} \boldsymbol{u}_{j+e_\alpha} = |\Omega_{j+e_\alpha}| \frac{d}{dt} V^\alpha_{j+e_\alpha}, \quad \boldsymbol{j} \in G_h, 1 \le \alpha \le 3, \text{ oder } \boldsymbol{M} \frac{d}{dt} V.$$

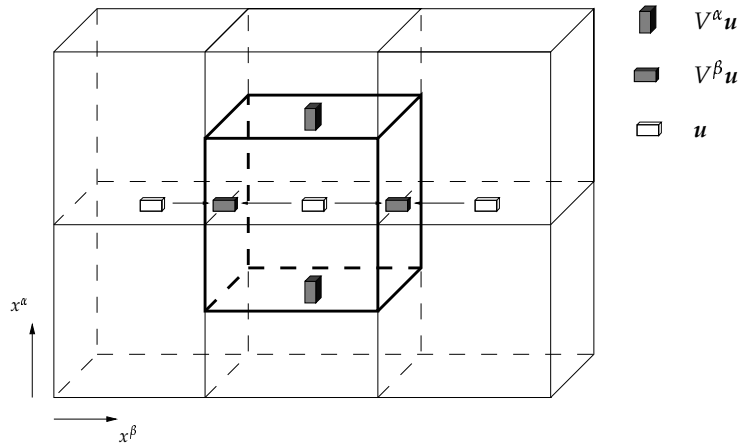

Abbildung 2.6: $\boldsymbol{u}$-Werte für den konvektiven Term

Der Operator $\boldsymbol{M}$ entspricht der punktweisen Multiplikation mit dem Volumen der Ω_{j+e_α}. Auf die Zeitdiskretisierung $\frac{d}{dt}$ wird in Abschnitt 2.1.8 eingegangen.

Die anderen Terme in (2.14) werden im folgenden auf ähnliche Weise diskretisiert, die entstehenden Terme sind allerdings etwas komplizierter. Dies liefert schließlich die folgende "*staggered mesh*"-(Orts-)Diskretisierung der Navier-Stokes-Gleichungen:

$$\begin{aligned} \boldsymbol{D}V &= 0, \\ \boldsymbol{M}\frac{d}{dt}V + \boldsymbol{A}(V)V + \boldsymbol{M}\boldsymbol{G}p - \boldsymbol{B}V &= 0. \end{aligned} \tag{2.15}$$

Diskretisierung des konvektiven Terms

Für den konvektiven Term gilt nach der Transformationsregel

$$\int_{\Omega_{j+e_\alpha}} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^i}(V^i \boldsymbol{u})\, d\Omega = \int_{G_{j+e_\alpha}} \frac{\partial}{\partial \xi^i}(V^i \boldsymbol{u})\, d\xi^1 d\xi^2 d\xi^3.$$

Das erste Integral wird approximiert durch

$$\begin{aligned} \int_{G_{j+e_\alpha}} \frac{\partial}{\partial \xi^\alpha}(V^\alpha \boldsymbol{u})\, d\xi &= \int_{j^\alpha}^{j^\alpha+1} \int_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}} \int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} \frac{\partial}{\partial \xi^\alpha}(V^\alpha \boldsymbol{u})\, d\xi^\gamma d\xi^\beta d\xi^\alpha \\ &= \int_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}} \int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} (V^\alpha \boldsymbol{u})\Big|_{j^\alpha}^{j^\alpha+1} d\xi^\gamma d\xi^\beta \\ &\approx (V^\alpha \boldsymbol{u})\Big|_{j}^{j+2e_\alpha}. \end{aligned}$$

Für die beiden anderen Integrale gilt ($\beta \neq \alpha$)

$$\begin{aligned}\int_{G_{j+e_\alpha}} \frac{\partial}{\partial \xi^\beta}(V^\beta \boldsymbol{u})\, d\xi &= \int_{j^\alpha}^{j^\alpha+1} \int_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}} \int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} \frac{\partial}{\partial \xi^\beta}(V^\beta \boldsymbol{u})\, d\xi^\gamma d\xi^\beta d\xi^\alpha \\ &= \int_{j^\alpha}^{j^\alpha+1} \int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} (V^\beta \boldsymbol{u})\Big|_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}} d\xi^\gamma d\xi^\alpha \\ &\approx (V^\beta \boldsymbol{u})\Big|_{\boldsymbol{j}+e_\alpha-e_\beta}^{\boldsymbol{j}+e_\alpha+e_\beta}.\end{aligned}$$

Bei diesen Formeln wird V^i durch die in Abschnitt A.1.3 beschriebenen Interpolationsregeln und $\boldsymbol{u}_{\boldsymbol{j}+e_\alpha \pm e_\beta}$ durch

$$\boldsymbol{u}_{\boldsymbol{j}+e_\alpha \pm e_\beta} := \frac{1}{2}(\boldsymbol{u}_{\boldsymbol{j}+e_\alpha} + \boldsymbol{u}_{\boldsymbol{j}+e_\alpha \pm 2e_\beta})$$

approximiert. Aus Stabilitätsgründen kann es, abhängig vom Gitter und der Reynolds-Zahl, notwendig sein, hier eine sogenannte Upwind-Diskretisierung zu verwenden. Die entsprechenden Werte für $\boldsymbol{u}$ lauten dann für den ersten Term (siehe Abbildung 2.6)

$$\boldsymbol{u}_{\boldsymbol{j}+e_\alpha+e_\beta} := \begin{cases} \boldsymbol{u}_{\boldsymbol{j}+e_\alpha} & \text{für } V^\beta_{\boldsymbol{j}+e_\alpha+e_\beta} > 0 \\ \boldsymbol{u}_{\boldsymbol{j}+e_\alpha+2e_\beta} & \text{für } V^\beta_{\boldsymbol{j}+e_\alpha+e_\beta} \le 0 \end{cases} \text{ bzw. } \boldsymbol{u}_j := \begin{cases} \boldsymbol{u}_{\boldsymbol{j}-e_\alpha} & \text{für } V^\alpha_{\boldsymbol{j}} > 0 \\ \boldsymbol{u}_{\boldsymbol{j}+e_\alpha} & \text{für } V^\alpha_{\boldsymbol{j}} \le 0 \end{cases}.$$

Diese Upwind-Diskretisierung ist ungenauer (nur noch von erster Ordnung), daher wird häufig eine Konvexkombination beider Diskretisierungen verwendet. Es liegt nahe, eine Upwind-Diskretisierung höherer Ordnung zu verwenden. Eine solche benötigt jedoch vier oder mehr statt der hier verwendeten zwei umgebenden $\boldsymbol{u}$-Werte. Dies würde die Anzahl der einfließenden Werte jedoch stark vergrößern, da zur Berechnung eines $\boldsymbol{u}$-Werts alle umgebenden V- und $\sqrt{g}\boldsymbol{a}^{(\alpha)}$-Werte benötigt werden. Dadurch wäre die Berechnung des konvektiven Terms am Rand des Gebietes problematisch.

Wie oben erhält man für $(\alpha, \beta, \gamma) \in \{(1,2,3),(2,3,1),(3,1,2)\}$ drei vektorwertige Größen, die wieder auf den Vektor $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ projeziert werden, um drei skalare Größen zu erhalten. So ergibt sich die diskrete Form $\boldsymbol{A}(\tilde{V})V$ des konvektiven Terms:

$$\left(\sqrt{g}\boldsymbol{a}^{(\alpha)}\right)_{\boldsymbol{j}+e_\alpha} \cdot \left[(\tilde{V}^\alpha \boldsymbol{u})\Big|_{\boldsymbol{j}}^{\boldsymbol{j}+2e_\alpha} + (\tilde{V}^\beta \boldsymbol{u})\Big|_{\boldsymbol{j}+e_\alpha-e_\beta}^{\boldsymbol{j}+e_\alpha+e_\beta} + (\tilde{V}^\gamma \boldsymbol{u})\Big|_{\boldsymbol{j}+e_\alpha-e_\gamma}^{\boldsymbol{j}+e_\alpha+e_\gamma}\right], \quad \boldsymbol{j} \in G_h, 1 \le \alpha \le 3.$$

Diskretisierung des Druckterms

Integriere nun den Druckterm ∇p:

$$\int_{\Omega_{\boldsymbol{j}+e_\alpha}} \nabla p\, d\Omega \approx \nabla p_{\boldsymbol{j}+e_\alpha} |\Omega_{\boldsymbol{j}+e_\alpha}|. \tag{2.16}$$

Zur Berechnung des Druckgradienten schreibe $\nabla p_{\boldsymbol{j}+e_\alpha}$ in Termen umgebender Druckwerte. Dazu integriere ∇p entlang von Kurven durch $\boldsymbol{j}+e_\alpha$, die jeweils zwei solche

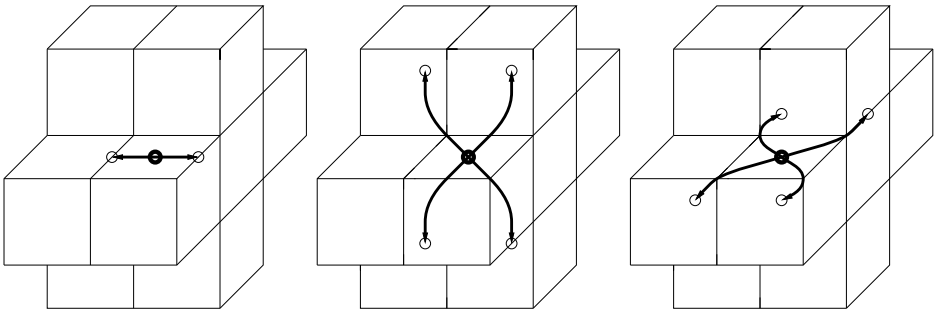

Abbildung 2.7: Integrationswege für den Druckterm

umgebenden Druckwerte verbinden (Abbildung 2.7):

$$p|_j^{j+2e_\alpha} = \int_{x_j}^{x_{j+2e_\alpha}} \nabla p \cdot d\boldsymbol{x} \approx \nabla p_{j+e_\alpha} \cdot \boldsymbol{x}|_j^{j+2e_\alpha}. \tag{2.17}$$

Um zwei weitere Gleichungen zu gewinnen, betrachte die Kurven ($\beta \neq \alpha$)

$$\begin{aligned} p|_{j-2e_\beta+2e_\alpha}^{j+2e_\beta} + p|_{j-2e_\beta}^{j+2e_\beta+2e_\alpha} &= \int_{x_{j-2e_\beta+2e_\alpha}}^{x_{j+2e_\beta}} \nabla p \cdot d\boldsymbol{x} + \int_{x_{j-2e_\beta}}^{x_{j+2e_\beta+2e_\alpha}} \nabla p \cdot d\boldsymbol{x} \\ &\approx \nabla p_{j+e_\alpha} \cdot \left(\boldsymbol{x}|_{j-2e_\beta+2e_\alpha}^{j+2e_\beta} + \boldsymbol{x}|_{j-2e_\beta}^{j+2e_\beta+2e_\alpha} \right). \end{aligned} \tag{2.18}$$

Damit erhält man drei lineare Gleichungen für ∇p_{j+e_α}, die einfach (z.B. mit GAUSSschem Eliminationsverfahren) gelöst werden können:

$$\begin{aligned} \boldsymbol{x}|_j^{j+2e_\alpha} \cdot \nabla p_{j+e_\alpha} &= p|_j^{j+2e_\alpha} \\ \left(\boldsymbol{x}|_{j-2e_\beta}^{j+2e_\beta} + \boldsymbol{x}|_{j-2e_\beta+2e_\alpha}^{j+2e_\beta+2e_\alpha} \right) \cdot \nabla p_{j+e_\alpha} &= p|_{j-2e_\beta}^{j+2e_\beta} + p|_{j-2e_\beta+2e_\alpha}^{j+2e_\beta+2e_\alpha} \\ \left(\boldsymbol{x}|_{j-2e_\gamma}^{j+2e_\gamma} + \boldsymbol{x}|_{j-2e_\gamma+2e_\alpha}^{j+2e_\gamma+2e_\alpha} \right) \cdot \nabla p_{j+e_\alpha} &= p|_{j-2e_\gamma}^{j+2e_\gamma} + p|_{j-2e_\gamma+2e_\alpha}^{j+2e_\gamma+2e_\alpha}. \end{aligned}$$

Für $1 \leq \alpha \leq 3$ erhält man drei vektorwertige Diskretisierungen für ∇p_{j+e_α}, die wieder mit $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ multipliziert werden:

$$\left(\sqrt{g}\boldsymbol{a}^{(\alpha)} \right)_{j+e_\alpha} \cdot \nabla p_{j+e_\alpha},\ \boldsymbol{j} \in G_h, 1 \leq \alpha \leq 3, \quad \text{oder} \quad \boldsymbol{G}p.$$

Diskretisierung des diffusiven Terms

Für die letzten drei Integrale, den *diffusiven Term*, gilt wieder mit der Transformationsregel

$$\int_{\Omega_{j+e_\alpha}} \frac{1}{\mathrm{Re}} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^i} (\sqrt{g}\boldsymbol{a}_k^{(i)} \boldsymbol{e}^{(k)})\, d\Omega = \frac{1}{\mathrm{Re}} \int_{G_{j+e_\alpha}} \frac{\partial}{\partial \xi^i} (\sqrt{g}\boldsymbol{a}_k^{(i)} \boldsymbol{e}^{(k)})\, d\xi^1 d\xi^2 d\xi^3.$$

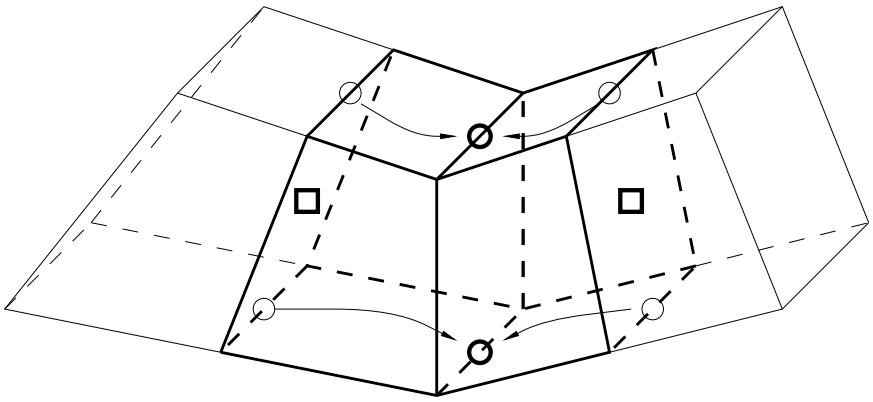

Abbildung 2.8: Berechnung des diffusiven Terms

Das erste dieser Integrale wird nun approximiert durch

$$\begin{aligned}
&\frac{1}{\mathrm{Re}}\int_{G_{j+e_\alpha}} \frac{\partial}{\partial\xi^\alpha}(\sqrt{g}\boldsymbol{a}_k^{(\alpha)}\boldsymbol{e}^{(k)})\, d\xi^1 d\xi^2 d\xi^3 \\
&\qquad = \frac{1}{\mathrm{Re}}\int_{j^\alpha}^{j^\alpha+1}\int_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}}\int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} \frac{\partial}{\partial\xi^\alpha}(\sqrt{g}\boldsymbol{a}_k^{(\alpha)}\boldsymbol{e}^{(k)})\, d\xi^\gamma d\xi^\beta d\xi^\alpha \\
&\qquad = \frac{1}{\mathrm{Re}}\int_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}}\int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} (\sqrt{g}\boldsymbol{a}_k^{(\alpha)}\boldsymbol{e}^{(k)})\big|_{j^\alpha}^{j^\alpha+1} d\xi^\gamma d\xi^\beta \\
&\qquad \approx \frac{1}{\mathrm{Re}}(\sqrt{g}\boldsymbol{a}_k^{(\alpha)}\boldsymbol{e}^{(k)})\big|_{\boldsymbol{j}}^{\boldsymbol{j}+2e_\alpha},
\end{aligned}$$

wobei für die Diskretisierung benutzt wurde, daß $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ in den Integrationsebenen, also ξ^β-ξ^γ-Ebenen in G_{j+e_α}, stetig ist (Rechtecke in Abbildung 2.8).
Für die anderen beiden Integrale gilt ($\alpha \neq \beta$)

$$\begin{aligned}
&\frac{1}{\mathrm{Re}}\int_{G_{j+e_\alpha}} \frac{\partial}{\partial\xi^\beta}(\sqrt{g}\boldsymbol{a}_k^{(\beta)}\boldsymbol{e}^{(k)})\, d\xi^1 d\xi^2 d\xi^3 \\
&\qquad = \frac{1}{\mathrm{Re}}\int_{j^\alpha}^{j^\alpha+1}\int_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}}\int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} \frac{\partial}{\partial\xi^\beta}(\sqrt{g}\boldsymbol{a}_k^{(\beta)}\boldsymbol{e}^{(k)})\, d\xi^\gamma d\xi^\beta d\xi^\alpha \\
&\qquad = \frac{1}{\mathrm{Re}}\int_{j^\alpha}^{j^\alpha+1}\int_{j^\gamma-\frac{1}{2}}^{j^\gamma+\frac{1}{2}} (\sqrt{g}\boldsymbol{a}_k^{(\beta)}\boldsymbol{e}^{(k)})\big|_{j^\beta-\frac{1}{2}}^{j^\beta+\frac{1}{2}} d\xi^\gamma d\xi^\alpha \\
&\qquad \approx \frac{1}{\mathrm{Re}}(\sqrt{g}\boldsymbol{a}_k^{(\beta)}\boldsymbol{e}^{(k)})_{\boldsymbol{j}+e_\alpha-e_\beta}^{\boldsymbol{j}+e_\alpha+e_\beta}.
\end{aligned}$$

Hier ist $\sqrt{g}\boldsymbol{a}^{(\beta)}$ nicht mehr stetig, daher berechne $(\sqrt{g}\boldsymbol{a}^{(\beta)})_{j+e_\alpha\pm e_\beta}$ nach den in Anhang A.1.3 beschriebenen Interpolationsregeln (Kreise in Abbildung 2.8).

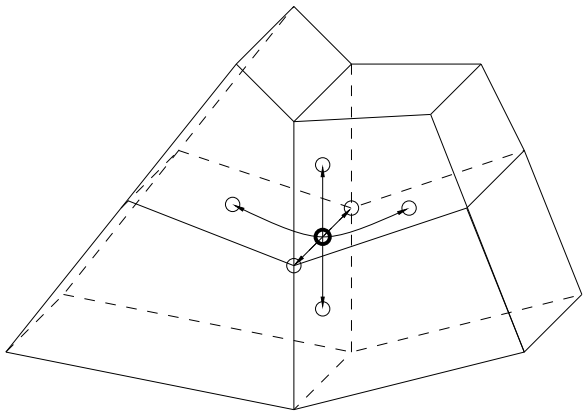

Abbildung 2.9: Integrationswege für die $\boldsymbol{u}$-Ableitungen

Für diese Diskretisierungen ist es notwendig, $\boldsymbol{e}^{(\beta)}$ und dafür alle Ableitungen von $\boldsymbol{u}$ in Zellmittelpunkten und Kantenmittelpunkten zu bestimmen. In den Zellmittelpunkten sind alle geometrischen Größen eindeutig, also folgt

$$\Big(\frac{\partial \boldsymbol{u}}{\partial x^\beta}\Big)_j = \Big(\sum_{k=1}^{3} \boldsymbol{a}_\beta^{(k)} \frac{\partial \boldsymbol{u}}{\partial \xi^k}\Big)_j \approx \sum_{k=1}^{3} (\boldsymbol{a}_\beta^{(k)})_j \, \boldsymbol{u}\big|_{j-e_k}^{j+e_k}. \tag{2.19}$$

In Kantenmittelpunkten sind bestimmte geometrische Größen nicht eindeutig. Daher verfahre ähnlich wie beim Druckgradienten und integriere über Kurven durch $\boldsymbol{x}_{j+e_\alpha+e_\beta}$, $\alpha \neq \beta$ (Abbildung 2.9). Es ist für $1 \leq \gamma \leq 3$, α, β fest,

$$\boldsymbol{u}^k\big|_{j+e_\alpha+e_\beta-e_\gamma}^{j+e_\alpha+e_\beta+e_\gamma} = \int_{\boldsymbol{x}_{j+e_\alpha+e_\beta-e_\gamma}}^{\boldsymbol{x}_{j+e_\alpha+e_\beta+e_\gamma}} \nabla \boldsymbol{u}^k \cdot \, d\boldsymbol{x} \approx (\nabla \boldsymbol{u}^k)_{j+e_\alpha+e_\beta} \cdot \boldsymbol{x}\big|_{j+e_\alpha+e_\beta-e_\gamma}^{j+e_\alpha+e_\beta+e_\gamma}. \tag{2.20}$$

Die benötigten $\boldsymbol{u}$-Werte werden mit (2.11) und den bekannten Interpolationsregeln ermittelt. Schließlich wird das lineare Gleichungssystem für $\nabla \boldsymbol{u}^k$ gelöst.
Wie oben erhält man wieder drei vektorwertige Größen, die mit $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ multipliziert werden und eine Diskretisierung $\boldsymbol{B}V$ des diffusiven Terms liefern ($\boldsymbol{j} \in G_h, 1 \leq \alpha \leq 3$):

$$\frac{1}{\mathrm{Re}}\big(\sqrt{g}\boldsymbol{a}^{(\alpha)}\big)_{j+e_\alpha} \cdot \sum_{k=1}^{3} \Big[(\sqrt{g}\boldsymbol{a}_k^{(\alpha)}\boldsymbol{e}^{(k)})\big|_{j}^{j+2e_\alpha} + (\sqrt{g}\boldsymbol{a}_k^{(\beta)}\boldsymbol{e}^{(k)})_{j+e_\alpha-e_\beta}^{j+e_\alpha+e_\beta} + (\sqrt{g}\boldsymbol{a}_k^{(\gamma)}\boldsymbol{e}^{(k)})_{j+e_\alpha-e_\gamma}^{j+e_\alpha+e_\gamma}\Big].$$

2.1.5 Diskretisierung der Transportgleichung

Für Visualisierungszwecke[2] und andere, in dieser Arbeit nicht behandelte Berechnungen von Stofftransporten, wird die *Transportgleichung*, die die Konzentrationsentwicklung $\Phi(\boldsymbol{x}, t) \in [0, 1]$ eines Stoffes mit Diffusionskonstante η beschreibt, behandelt. Diese

[2] siehe z.B. `dfg2.mpg` auf der beiliegenden CD

lautet

$$\frac{\partial \Phi}{\partial t} + \sum_{\alpha} \frac{\partial}{\partial x^\alpha}(u^\alpha \Phi) - \sum_{\alpha} \eta \frac{\partial}{\partial x^\alpha}\frac{\partial \Phi}{\partial x^\alpha} = 0 \tag{2.21}$$

in Ω und wird auf dem Rand $\partial\Omega$ mit Neumann- oder Dirichlet-Randbedingungen versehen (Kapitel 2.1.6). Transformation der Transportgleichung (2.21) wie in Abschnitt 2.1.4 liefert

$$\frac{\partial \Phi}{\partial t} + \sum_{\beta} \frac{1}{\sqrt{g}}\frac{\partial}{\partial \xi^\beta}(V^\beta \Phi) - \sum_{\alpha,\beta} \eta \frac{1}{\sqrt{g}}\frac{\partial}{\partial \xi^\beta}(\sqrt{g}\boldsymbol{a}_\alpha^{(\beta)}\frac{\partial \Phi}{\partial x^\alpha}) = 0. \tag{2.22}$$

Integriere nun über das Kontrollvolumen $\Omega_j = \boldsymbol{x}(\boldsymbol{j} + [-\frac{1}{2}, \frac{1}{2}]^3) =: \boldsymbol{x}(G_j)$. Dies ergibt für den ersten Term von (2.22)

$$\int_{\Omega_j} \frac{\partial \Phi}{\partial t}\, d\Omega \approx |\Omega_j| \frac{d}{dt}\Phi_j.$$

Für die nächsten drei Summanden von (2.22) gilt

$$\begin{aligned}
\int_{\Omega_j} \frac{1}{\sqrt{g}}\frac{\partial}{\partial \xi^\alpha}(V^\alpha \Phi)\, d\Omega &= \int_{G_j} \frac{\partial}{\partial \xi^\alpha}(V^\alpha \Phi)\, d\xi \\
&= \int_{j^\alpha - \frac{1}{2}}^{j^\alpha + \frac{1}{2}} \int_{j^\beta - \frac{1}{2}}^{j^\beta + \frac{1}{2}} \int_{j^\gamma - \frac{1}{2}}^{j^\gamma + \frac{1}{2}} \frac{\partial}{\partial \xi^\alpha}(V^\alpha \Phi)\, d\xi^\gamma d\xi^\beta d\xi^\alpha \\
&= \int_{j^\beta - \frac{1}{2}}^{j^\beta + \frac{1}{2}} \int_{j^\gamma - \frac{1}{2}}^{j^\gamma + \frac{1}{2}} (V^\alpha \Phi)|_{j^\alpha - \frac{1}{2}}^{j^\alpha + \frac{1}{2}}\, d\xi^\gamma d\xi^\beta \\
&\approx (V^\alpha \Phi)|_{\boldsymbol{j} - e_\alpha}^{\boldsymbol{j} + e_\alpha}.
\end{aligned}$$

Die Werte Φ_{j+e_α} sind hier wieder durch eine Upwind-Diskretisierung zu ermitteln. Da die Φ-Werte jedoch explizit in den Zellmittelpunkten gegeben sind, ist es hier möglich, eine Diskretisierung höherer Ordnung zu verwenden, ohne die Anzahl der einfließenden Werte massiv zu vergrößern. Setze dazu (vgl. [Zij96, p. 51])

$$\Phi_{j+e_\alpha} = \begin{cases} \Phi_j + \frac{1}{4}\left((1+\kappa)(\Phi_{j+2e_\alpha} - \Phi_j) + (1-\kappa)(\Phi_j - \Phi_{j-2e_\alpha}\right) \\ \qquad \text{für } V^\alpha_{j+e_\alpha} > 0 \\ \Phi_{j+2e_\alpha} - \frac{1}{4}\left((1+\kappa)(\Phi_{j+2e_\alpha} - \Phi_j) + (1-\kappa)(\Phi_{j+4e_\alpha} - \Phi_{j+2e_\alpha})\right) \\ \qquad \text{für } V^\alpha_{j+e_\alpha} \leq 0 \end{cases} \tag{2.23}$$

mit $\kappa \in [-1, 1]$. Für $\kappa = -1$, $\kappa = \frac{1}{2}$ und $\kappa = \frac{1}{3}$ erhält man die Schemata LUDS [Per85] und QUICK [Leo79] zweiter Ordnung bzw. CUI [BMC88] (alle nach [Zij96]) mit dritter Ordnung.

Für die letzten drei Summanden von (2.22) gilt schließlich

$$\begin{aligned}
&\eta \int_{\Omega_j} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^\beta} \big(\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \frac{\partial \Phi}{\partial x^\alpha} \big) \, d\Omega \\
&\qquad = \eta \int_{G_j} \frac{\partial}{\partial \xi^\beta} \big(\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \frac{\partial \Phi}{\partial x^\alpha} \big) \, d\xi \\
&\qquad = \eta \int_{j^\alpha - \frac{1}{2}}^{j^\alpha + \frac{1}{2}} \int_{j^\beta - \frac{1}{2}}^{j^\beta + \frac{1}{2}} \int_{j^\gamma - \frac{1}{2}}^{j^\gamma + \frac{1}{2}} \frac{\partial}{\partial \xi^\beta} \big(\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \frac{\partial \Phi}{\partial x^\alpha} \big) \, d\xi^\gamma d\xi^\beta d\xi^\alpha \\
&\qquad = \eta \int_{j^\alpha - \frac{1}{2}}^{j^\alpha + \frac{1}{2}} \int_{j^\gamma - \frac{1}{2}}^{j^\gamma + \frac{1}{2}} \big(\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \frac{\partial \Phi}{\partial x^\alpha} \big) \big|_{j^\beta - \frac{1}{2}}^{j^\beta + \frac{1}{2}} \, d\xi^\gamma d\xi^\alpha \\
&\qquad \approx \eta \big(\sqrt{g} \boldsymbol{a}_\alpha^{(\beta)} \frac{\partial \Phi}{\partial x^\alpha} \big) \big|_{j - e_\beta}^{j + e_\beta}.
\end{aligned}$$

Die Berechnung von $\nabla \Phi_{j+e_\alpha}$ erfolgt genauso wie die Berechnung des Druckgradienten. Man erhält so eine Diskretisierung der Transportgleichung (2.21):

$$\boldsymbol{M} \frac{d\Phi}{dt} + \boldsymbol{T}(V)\Phi = 0.$$

2.1.6 Randbedingungen

Die Navier-Stokes-Gleichungen werden an den Rändern mit verschiedenen Randbedingungen versehen. Im folgenden bezeichne $\boldsymbol{n}$ den nach außen gerichteten Normalenvektor mit Länge 1 auf dem Rand und $\boldsymbol{t}_1$, $\boldsymbol{t}_2$ entsprechende Tangentialvektoren. Damit ergeben sich normale und tangentiale Geschwindigkeitskomponenten $\boldsymbol{u}^n = (\boldsymbol{u} \cdot \boldsymbol{n})\boldsymbol{n}$ bzw. $\boldsymbol{u}^t = (\boldsymbol{u} \cdot \boldsymbol{t})\boldsymbol{t}$, $\boldsymbol{t} = \boldsymbol{t}_1, \boldsymbol{t}_2$. Für die Geschwindigkeit können zunächst zwei verschiedene Bedingungen formuliert werden:

- *Dirichlet-Bedingungen*, d.h. die entsprechenden Geschwindigkeitskomponenten $\boldsymbol{u}^n$ bzw. $\boldsymbol{u}^t$ wird vorgegeben;
- *Neumann-Bedingungen*, d.h. Ableitungen in Richtung $\boldsymbol{n}$ der entsprechenden Geschwindigkeitskomponenten werden (in dieser Arbeit nur mit Null) vorgegeben.

Mit diesen Randbedingungen lassen sich verschiedene physikalische Situationen modellieren:

- *Einström-Randbedingung*, d.h. Dirichlet-Bedingungen für $\boldsymbol{u}^n$ und $\boldsymbol{u}^t$, die eine Geschwindigkeit ungleich Null in Normalenrichtung vorschreiben.
- *Ausström-Bedingung*, d.h. Neumann-Randbedingungen für Richtungen $\frac{\partial}{\partial \boldsymbol{n}} \boldsymbol{u}^t = \frac{\partial}{\partial \boldsymbol{n}} \boldsymbol{u}^n = 0$.

- *Haftbedingung* (noslip-Bedingung), d.h. Dirichlet-Bedingungen $\boldsymbol{u}^n = 0$ und $\boldsymbol{u}^t = 0$. Diese Randbedingung ist an normalen Wänden zu finden.
- *Slip-Bedingung*, d.h. Dirichlet-Randbedingung $\boldsymbol{u}^n = 0$ und Neumann-Randbedingung $\frac{\partial}{\partial \boldsymbol{n}} \boldsymbol{u}^t = 0$. Diese Bedingung simuliert reibungsfreies Vorbeifließen eines Fluids an der Wand.

Diskretisierung der Randbedingungen

Die Normal- bzw. Tangentialvektoren sind gegeben durch

$$\boldsymbol{n} = \frac{\boldsymbol{a}^{(n)}}{\|\boldsymbol{a}^{(n)}\|}, \qquad \boldsymbol{t}_i = \frac{\boldsymbol{a}_{(t_i)}}{\|\boldsymbol{a}_{(t_i)}\|}, \quad 1 \leq i \leq 2.$$

Sei die Geschwindigkeit u^n in Richtung $\boldsymbol{n}$ vorgegeben, dann gilt

$$V^n = (u^n \boldsymbol{n}) \cdot \sqrt{g}\boldsymbol{a}^{(n)} = u^n \boldsymbol{n} \cdot \sqrt{g}\boldsymbol{a}^{(n)} = u^n \|\sqrt{g}\boldsymbol{a}^{(n)}\|.$$

Dieses V^n wird im entsprechenden Randpunkt gesetzt. Seien nun Tangentialgeschwindigkeiten u^{t_1}, u^{t_2} in Richtung $\boldsymbol{t}_1$ bzw. $\boldsymbol{t}_2$ vorgegeben. Damit folgt ($1 \leq i \leq 2$)

$$u^{t_i} = \boldsymbol{t}_i \cdot \boldsymbol{u} \overset{(2.11)}{=} \frac{\boldsymbol{a}_{(t_i)}}{\|\boldsymbol{a}_{(t_i)}\|} \cdot \frac{1}{\sqrt{g}} \sum_{k=1}^{3} \boldsymbol{a}_{(k)} V^k$$

und weiter

$$\begin{aligned} \sqrt{g}\|\boldsymbol{a}_{(t_i)}\| u^{t_i} &= \boldsymbol{a}_{(t_i)} \cdot \boldsymbol{a}_{(t_1)} V^{t_1} + \boldsymbol{a}_{(t_i)} \cdot \boldsymbol{a}_{(t_2)} V^{t_2} + \boldsymbol{a}_{(t_i)} \cdot \boldsymbol{a}_{(n)} V^n \\ &= g_{t_i t_1} V^{t_1} + g_{t_i t_2} V^{t_2} + g_{t_i n} V^n. \end{aligned}$$

Daraus können V^{t_1} und V^{t_2} wie folgt berechnet werden:

$$\begin{aligned} V^{t_1} &= \frac{g_{t_1 t_2}\left(\sqrt{g}\|\boldsymbol{a}_{(t_2)}\| u^{t_2} - g_{t_2 n} V^n\right) - g_{t_2 t_2}\left(\sqrt{g}\|\boldsymbol{a}_{(t_1)}\| u^{t_1} - g_{t_1 n} V^n\right)}{g_{t_1 t_2}^2 - g_{t_1 t_1} g_{t_2 t_2}}, \\ V^{t_2} &= \frac{g_{t_1 t_2}\left(\sqrt{g}\|\boldsymbol{a}_{(t_1)}\| u^{t_1} - g_{t_1 n} V^n\right) - g_{t_1 t_1}\left(\sqrt{g}\|\boldsymbol{a}_{(t_2)}\| u^{t_2} - g_{t_2 n} V^n\right)}{g_{t_1 t_2}^2 - g_{t_1 t_1} g_{t_2 t_2}}. \end{aligned}$$

Da Neumann-Randbedingungen nur bei Ausström- und Slip-Randbedingungen vorkommen und diese Randbedingungen in der Regel nicht in Bereichen mit komplizierten Geometrien auftreten, wird von einer einfachen Geometrie ausgegangen[3]. In diesem Fall können Neumann-Randbedingungen (mit Ableitung gleich Null) durch lineare Extrapolation der entsprechenden Größen implementiert werden.

[3]Genauer: Eine Randzelle und die angrenzende Ghost-Zelle sind deckungsgleich.

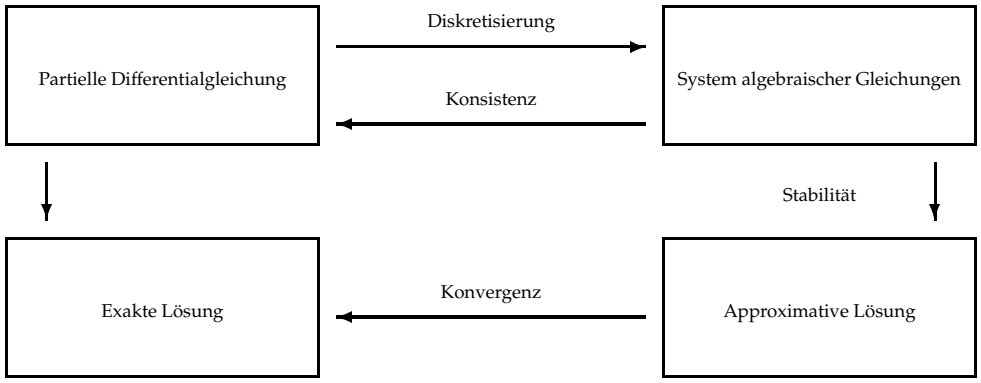

Abbildung 2.10: Zusammenhang zwischen Konsistenz, Stabilität und Konvergenz (nach [Fle91]). Die partiellen Differentialgleichungen werden diskretisiert, die Konsistenz gibt Informationen über die Qualität der Diskretisierung. Zusammen mit der Stabilität läßt sich Konvergenz der approximativen Lösung gegen die exakte Lösung erreichen.

2.1.7 Konsistenzordnung der Ortsdiskretisierungen

Um Aussagen über die Qualität einer Diskretisierung machen zu können, sind die Begriffe *Konsistenz*, *Stabilität* und *Konvergenz* von entscheidender Bedeutung. Die Zusammenhänge zwischen diesen Begriffen sind in Abbildung 2.10 dargestellt. Die folgenden Definitionen sind weitgehend aus [IK73] entnommen.

Betrachte die partielle Differentialgleichung

$$L(\boldsymbol{u})(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \quad \text{mit Randbedingungen} \quad B(\boldsymbol{u})(\boldsymbol{x}) = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma \subset \partial\Omega. \tag{2.24}$$

Sei $G_h = \Omega_\Delta \cup \Gamma_\Delta$ ein Gitter, auf dem die Variablen $\boldsymbol{x}$ des Problems (2.24) definiert sind, Ω_Δ und Γ_Δ seien innere bzw. Randpunkte. Dieses Gitter habe eine maximale Schrittweite[4] $h = \Delta\boldsymbol{x}$. Eine Diskretisierung von (2.24) auf dem Gitter G mit diskreter Lösung $\boldsymbol{U}$ lautet dann

$$L_\Delta(\boldsymbol{U})(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega_\Delta \quad \text{und} \quad B_\Delta(\boldsymbol{U})(\boldsymbol{x}) = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \Gamma_\Delta. \tag{2.25}$$

Selbstverständlich sollen die Diskretisierungen L_Δ und B_Δ punktweise gute Näherungen an die tatsächlichen Operatoren L und B liefern. Dies wird formalisiert durch den Begriff der Konsistenz.

Definition: (Konsistenz). Sei $\varphi(\boldsymbol{x})$ eine beliebige, hinreichend oft stetig partiell differenzierbare Funktion. Der *lokale Diskretisierungsfehler* ist definiert durch

$$\lambda(\varphi(\boldsymbol{x})) := L(\varphi(\boldsymbol{x})) - L_\Delta(\varphi(\boldsymbol{x})) \quad \text{bzw.} \quad \beta(\varphi(\boldsymbol{x})) := B(\varphi(\boldsymbol{x})) - B_\Delta(\varphi(\boldsymbol{x})). \tag{2.26}$$

[4] z.B. den Durchmesser der eine Zelle einschließenden Kugel

Die Diskretisierung (2.25) heißt *konsistent*, wenn mit Normen der (endlichdimensionalen) Räume $\Omega_\Delta, \Gamma_\Delta$ gilt

$$\|\lambda(\varphi)\| \to 0, \quad \|\beta(\varphi)\| \to 0 \quad \text{für } h \to 0 \text{ und alle Gitter } G_h.$$

Die Diskretisierung heißt *konsistent von Ordnung p*, wenn

$$\|\lambda(\varphi)\| + \|\beta(\varphi)\| = O(h^p) \quad \text{für alle Gitter } G_h.$$

Durch Konsistenz ist noch nicht garantiert, daß die diskrete Lösung $\boldsymbol{U}$ auch gegen die tatsächliche Lösung $\boldsymbol{u}$ konvergiert. Dies ist Inhalt der folgenden Definition.

Definition: (Konvergenz). Seien $\boldsymbol{u}$ und $\boldsymbol{U}$ die Lösungen der Probleme (2.24) bzw. (2.25). Die diskrete Lösung $\boldsymbol{U}$ heißt *konvergent mit Ordnung p* gegen die exakte Lösung, wenn

$$\|\boldsymbol{u}|_{G_h} - \boldsymbol{U}\| = O(h^p) \quad \text{für alle Gitter } G_h.$$

Da der Computer nur mit beschränkter Genauigkeit rechnen kann und auch an anderen Stellen Daten nur ungenau gegeben sein können, ist es wichtig, daß die Auswirkungen dieser Fehler auf die berechnete Lösung a priori abschätzbar sind. Eine solche Abschätzung wird durch die folgende Definition der *Stabilität* formalisiert.

Definition: (Stabilität). Eine Diskretisierung (2.25) heißt *stabil*, wenn eine Zahl K existiert, so daß für alle Schrittweiten h, alle Gitter G_h und alle auf G_h definierten Gitterfunktionen $\boldsymbol{U}$ und $\boldsymbol{V}$ gilt

$$\|\boldsymbol{U} - \boldsymbol{V}\| \leq K(\|L_\Delta(\boldsymbol{U}) - L_\Delta(\boldsymbol{V})\| + \|B_\Delta(\boldsymbol{U}) - B_\Delta(\boldsymbol{V})\|.$$

Aus Stabilität und Konsistenz ergibt sich Konvergenz:

Satz: Sei (2.25) eine stabile und konsistente Diskretisierung der Differentialgleichung (2.24). Dann konvergiert die diskrete Lösung $\boldsymbol{U}$ von (2.25) gegen die exakte Lösung $\boldsymbol{u}$ von (2.24). Dabei ist die Konvergenzordnung gleich der Konsistenzordnung.

Die Konsistenzordnung der in den vorherigen Abschnitten beschriebenen Diskretisierung soll nun berechnet werden. Für alle Beweise sei auf Anhang A.1.4 verwiesen.

Die Diskretisierung des Divergenzoperators $\boldsymbol{D}$ ist konsistent von erster Ordnung und für den Fall, daß die entsprechende Zelle ein Parallelepiped ist, d.h. alle Seitenflächen Parallelogramme sind, konsistent von zweiter Ordnung.

Die Diskretisierungen der Operatoren $\boldsymbol{M}$, $\boldsymbol{A}(\tilde{V})$ und $\boldsymbol{B}$ sind konsistent von erster Ordnung und für den Fall, daß die beiden Zellen, die das Kontrollvolumen enthalten (vgl. Abbildung 2.5), identische Parallelepipede sind, konsistent von zweiter Ordnung.

Die Diskretisierung des Operators $\boldsymbol{G}$ ist konsistent von zweiter Ordnung.

2.1.8 Die Zeitdiskretisierung

Die diskretisierten Navier-Stokes-Gleichungen schreiben sich als (vgl. (2.15))

$$\boldsymbol{M}\frac{d}{dt}V + \boldsymbol{N}(V)V + \boldsymbol{M}\boldsymbol{G}p = 0, \quad \boldsymbol{D}V = 0,$$

wobei $\boldsymbol{M}$ der (punktweisen) Multiplikation mit $|\Omega_{j+e_\alpha}|$, $\boldsymbol{N}$ dem diskretisierten konvektiven minus dem diskretisierten diffusiven Term ($\boldsymbol{N}(\tilde{V})V = \boldsymbol{A}(\tilde{V})V - \boldsymbol{B}V$), $\boldsymbol{G}$ dem diskreten Gradienten und $\boldsymbol{D}$ dem diskreten Divergenzoperator entspricht. Setzt man die einfachste Diskretisierung der Zeitableitung – ein explizites Eulerverfahren – ein, so erhält man

$$\boldsymbol{M}\frac{V^{n+1} - V^n}{\Delta t} + \boldsymbol{N}(V^n)V^n + \boldsymbol{M}\boldsymbol{G}p^{n+\frac{1}{2}} = 0, \quad \boldsymbol{D}V^n = \boldsymbol{D}V^{n+1} = 0$$

bzw.

$$\frac{V^{n+1} - V^n}{\Delta t} + \boldsymbol{M}^{-1}\boldsymbol{N}(V^n)V^n + \boldsymbol{G}p^{n+\frac{1}{2}} = 0, \quad \boldsymbol{D}V^n = \boldsymbol{D}V^{n+1} = 0. \tag{2.27}$$

Nun wird in drei Schritten eine Lösung für den Zeitschritt $n+1$ nach der *Chorin-Projektionsmethode* berechnet:

1. Berechne eine Prediktion $\tilde{V}$ für die Flüsse:

   $$\tilde{V} = V^n - \Delta t\boldsymbol{M}^{-1}\boldsymbol{N}(V^n)V^n. \tag{2.28}$$

   Dieses $\tilde{V}$ ist i.A. nicht divergenzfrei, d.h. es erfüllt nicht die Kontinuitätsgleichung (2.2).

2. Aus (2.27) folgt

   $$\tilde{V} - V^{n+1} = \Delta t\boldsymbol{G}p^{n+\frac{1}{2}}. \tag{2.29}$$

   Anwendung des diskreten Divergenzoperators $\boldsymbol{D}$ liefert unter der Annahme, daß V^{n+1} divergenzfrei sein soll,

   $$\boldsymbol{D}\tilde{V} - \boldsymbol{D}V^{n+1} = \boldsymbol{D}\tilde{V} = \Delta t\boldsymbol{D}\boldsymbol{G}p^{n+\frac{1}{2}}.$$

   Löse also das *lineare Gleichungssystem*

   $$\boldsymbol{D}\boldsymbol{G}p^{n+\frac{1}{2}} = \frac{1}{\Delta t}\boldsymbol{D}\tilde{V}. \tag{2.30}$$

3. Korrigiere die Flüsse nach (2.29):

   $$V^{n+1} = \tilde{V} - \Delta t\boldsymbol{G}p^{n+\frac{1}{2}}. \tag{2.31}$$

Dieses Schema hat den Nachteil, daß die Geschwindigkeiten nur in erster Ordnung approximiert werden, egal, welche Diskretisierung der Zeitableitung benutzt wird (siehe [Wet98]).

Alternativ kann das *Druckkorrekturverfahren* (nach van Kan [van86]) angewendet werden. Dieses unterscheidet sich von der Chorin-Projektionsmethode dadurch, daß in der Prediktion der Druckgradient berücksichtigt wird. Die entsprechenden Schritte lauten dann:

1. Berechne die Prediktion für die Flüsse

$$\tilde{V} = V^n - \Delta t\big(\boldsymbol{M}^{-1}\boldsymbol{N}(V^n)V^n + \boldsymbol{G}p^{n-\frac{1}{2}}\big). \tag{2.32}$$

2. Wieder mit (2.27) folgt

$$\tilde{V} - V^{n+1} = \Delta t\boldsymbol{G}(p^{n+\frac{1}{2}} - p^{n-\frac{1}{2}}) =: \Delta t\boldsymbol{G}q^n.$$

   Löse das lineare Gleichungssystem

$$\boldsymbol{D}\boldsymbol{G}q^n = \frac{1}{\Delta t}\boldsymbol{D}\tilde{V}. \tag{2.33}$$

3. Korrigiere die Flüsse und den Druck mittels

$$V^{n+1} = \tilde{V} - \Delta t\boldsymbol{G}q^n, \qquad p^{n+\frac{1}{2}} = p^{n-\frac{1}{2}} + q^n. \tag{2.34}$$

Eine Zeitdiskretisierung höherer Ordnung erhält man durch Verwendung der Adams-Bashforth-Formeln. Diese erhält man durch

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} P(t)\,dt,$$

wobei $P(t)$ das y an den t_i interpolierende Polynom entsprechender Ordnung ist. So lautet (2.27) für eine Diskretisierung zweiter Ordnung mit Zeitschritten Δt_n und Δt_{n-1} und Gewichten $\alpha_n^1 = 1 + \frac{\Delta t_n}{2\Delta t_{n-1}}$, $\alpha_n^2 = \frac{\Delta t_n}{2\Delta t_{n-1}}$

$$\begin{aligned}\frac{V^{n+1} - V^n}{\Delta t_n} &+ \alpha_n^1\big(\boldsymbol{M}^{-1}\boldsymbol{N}(V^n)V^n + \boldsymbol{G}p^{n+\frac{1}{2}}\big)\\ &- \alpha_n^2\big(\boldsymbol{M}^{-1}\boldsymbol{N}(V^{n-1})V^{n-1} + \boldsymbol{G}p^{n-\frac{1}{2}}\big) = 0.\end{aligned}$$

Für das Druckkorrekturverfahren ergibt sich statt (2.32)

$$\begin{aligned}\tilde{V} = V^n - \Delta t_n\big[&\alpha_n^1\big(\boldsymbol{M}^{-1}\boldsymbol{N}(V^n)V^n + \boldsymbol{G}p^{n-\frac{1}{2}}\big) -\\ &\alpha_n^2\boldsymbol{M}^{-1}\boldsymbol{N}(V^{n-1})V^{n-1}\big].\end{aligned}$$

Weiterhin ist

$$\tilde{V} - V^{n+1} = \Delta t_n \boldsymbol{G}\left(\alpha_n^1 p^{n+\frac{1}{2}} - (\alpha_n^1 + \alpha_n^2) p^{n-\frac{1}{2}}\right) =: \Delta t_n \boldsymbol{G} q^n.$$

Damit lautet die Druckkorrektur (2.34)

$$p^{n+\frac{1}{2}} = \left(q^n + (\alpha_n^1 + \alpha_n^2) p^{n-\frac{1}{2}}\right)/\alpha_n^1.$$

Für die diskretisierte Transportgleichung (2.21)

$$\boldsymbol{M}\frac{d\Phi}{dt} + \boldsymbol{T}(V)\Phi = 0$$

lautet die Adams-Bashforth-Diskretisierung zweiter Ordnung

$$\Phi^{n+1} = \Phi^n - \Delta t_n \left(\alpha_n^1 \boldsymbol{M}^{-1}\boldsymbol{T}(V^n)\Phi^n - \alpha_n^2 \boldsymbol{M}^{-1}\boldsymbol{T}(V^{n-1})\Phi^{n-1}\right).$$

Bei diesen Zeitdiskretisierungsverfahren ist die Zeitschrittweite durch die *CFL-Bedingung*[5] beschränkt. Diese Bedingung beschreibt anschaulich, daß sich Information im diskreten Modell schneller "verbreiten" muß als im kontinuierlichen Modell. Sie lautet hier

$$\Delta t \leq \min_{1\leq\alpha\leq 3, j\in G_h} \frac{|\Omega_{j+e_\alpha}|}{|V^\alpha_{j+e_\alpha}|}, \tag{2.35}$$

d.h. das verschobene Kontrollvolumen Ω_{j+e_α} muß mehr Rauminhalt haben als im Zeitschritt Δt durch den Fluß $V^\alpha_{j+e_\alpha}$ verschoben wird. Um die Stabilität dieses expliziten Verfahrens zu gewährleisten, muß die Zeitschrittweite weiterhin der Bedingung

$$\Delta t \leq \frac{1}{4}\,\mathrm{Re}\cdot\Delta x^2 \tag{2.36}$$

genügen[6] (vgl. [PT83]). Zur Sicherheit werden die Zeitschrittweiten, die sich aus (2.35) und (2.36) ergeben, jeweils noch mit einer einstellbaren Konstanten $c \in (0,1)$ multipliziert.

[5] Courant-Friedrichs-Lewy-Bedingung, vgl. [PT83]

[6] Hierbei wird Δx wie in Abschnitt 2.1.7 bestimmt.

2.2 Iterative Lösung linearer Gleichungssysteme

In Abschnitt 2.1.8 wurden Verfahren zur Zeitdiskretisierung beschrieben. Dabei waren Unterprobleme in Form von linearen Gleichungssystemen $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ zu lösen, siehe (2.30) bzw. (2.33). Diese Gleichungssysteme sind sehr groß: für jede Zelle gibt es eine Unbekannte. Einfache direkte Verfahren, zum Beispiel das GAUSSsche Eliminationsverfahren, haben eine Laufzeit der Ordnung $O(n^3)$ und einen (worst-case) Platzbedarf von $O(n^2)$, wobei n die Zahl der Unbekannten ist. Weiterhin berücksichtigen sie nicht, daß die Matrix $\boldsymbol{A}$ hier *dünn besetzt* ist. Daher sind diese Verfahren für die in dieser Arbeit auftretenden sehr großen linearen Gleichungssysteme ungeeignet. Weiterhin ist es nicht notwendig, die auftretenden linearen Gleichungssysteme exakt zu lösen, es reicht, *iterativ* eine Approximation der Lösung zu berechnen. Ein iteratives Verfahren produziert eine Folge von Vektoren $(\boldsymbol{x}_i)_{i\in\mathbb{N}}$, die gegen die Lösung $\boldsymbol{x}$ des Gleichungssystems $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ konvergiert. Eine Übersicht über verschiedene Verfahren findet sich in [BBC+94].

Hier werden nur zwei implementierten Methoden beschrieben: GMRES (Abschnitt 2.2.1) und BiCGStab(ℓ) (Abschnitt 2.2.2). Diese Methoden sind auf unsymmetrische Matrizen anwendbar, wie sie bei gekrümmten Gittern auftreten[7].

Die Konvergenzrate dieser iterativen Verfahren steht in Zusammenhang mit der *Kondition* $\|\boldsymbol{A}\| \cdot \|\boldsymbol{A}^{-1}\|$ der Matrix $\boldsymbol{A}$. Die Kondition der Matrizen $\boldsymbol{A} = \boldsymbol{D}\boldsymbol{G}$, wie sie in (2.30) bzw. (2.33) auftreten, und damit die Konvergenzrate der iterativen Verfahren verschlechtern sich mit Verfeinerung des Gitters. Daher ist es notwendig, einen *Vorkonditionierer* zu verwenden: statt des ursprünglichen Gleichungssystems $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ wird ein neues Gleichungssystem $\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{P}^{-1}\boldsymbol{b}$ gelöst, wobei $\boldsymbol{P}^{-1}\boldsymbol{A}$ eine bessere Kondition hat als $\boldsymbol{A}$. Diese Vorgehensweise ist natürlich nur effizient, wenn der Operator $\boldsymbol{P}^{-1}$ relativ einfach auszuwerten ist. In dieser Arbeit wurde ein Lifting Interpolet Vorkonditionierer verwendet (siehe [GK00, KG00]). Die Auswirkung auf die Anzahl der Iterationen ist in Abbildung 2.11 dargestellt.

Die effizientesten bekannten Methoden zur Lösung der hier auftretenden Gleichungssysteme sind *Mehrgitterverfahren*. Diese sind jedoch keine "Black-Box"-Löser, sondern müssen an jedes Problem neu angepaßt werden. Dies ist jedoch nicht Hauptthema dieser Arbeit und kann später weiterverfolgt werden.

Da die Approximation der Lösung nur bis zu einer gewissen, vom Benutzer vorgegebenen Qualität erfolgen muß, ist es notwendig, Konvergenz- bzw. Haltekriterien für die iterativen Verfahren festzulegen. In dieser Arbeit wurden die folgenden Kriterien implementiert:

- die maximale Anzahl der Iterationen (oder die Anzahl der Matrix-Vektor-Produkt-Auswertungen) n wird vorgegeben;
- die Norm des Residuums $\|\boldsymbol{r}_i\| = \|\boldsymbol{A}\boldsymbol{x}_i - \boldsymbol{b}\|$ unterschreitet einen vorgegebenen Wert ε;

[7] Aufgrund der Randbedingungen für den Druck ist es nicht in einfacher Weise möglich, für die Lösung von (2.30) bzw. (2.33) unter Verwendung eines problemangepaßten Skalarprodukts ein CG-Verfahren zu benutzen, welches weniger Rechenaufwand bedeuten würde.

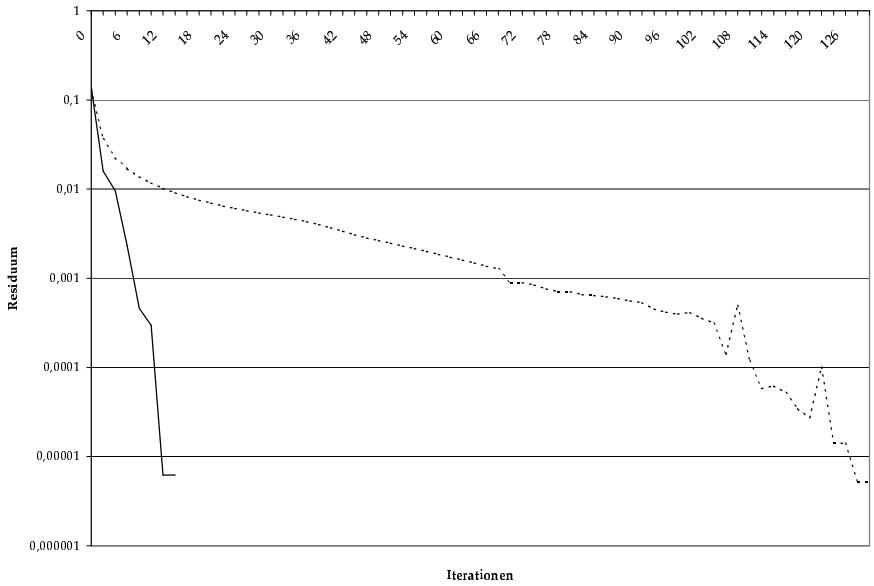

Abbildung 2.11: Residuum für die erste Iteration der Berechnung des Beispiels aus Kapitel 5.1.1 mit $63^2 \cdot 2$ Zellen, mit (durchgezogene Linie) und ohne (gestrichelte Linie) Vorkonditionierer

- die relative Größe des Residuums $\|\boldsymbol{r}_i\|/\|\boldsymbol{r}_0\|$ unterschreitet einen vorgegebenen Wert δ.

In Tabelle 2.1 sind die Rechenzeit- und Speicherkosten einiger Verfahren dargestellt.

Verfahren	Rechenzeit			Speicherbedarf
	$\boldsymbol{Ax}$	$\boldsymbol{x}+\alpha\boldsymbol{y}$	$\boldsymbol{x}^T\boldsymbol{y}$	
BiCG	2	6.5	2	7
CGS	1	3.25	1	7
BiCGStab	1	3	2	7
BiCGStab(ℓ)	1	0.75(ℓ+3)	0.25(ℓ+7)	$2\ell+5$
GMRES(m)	1	$\approx$ 0.5(m+3)	$\approx$0.5(m+1)	$m+3$

Tabelle 2.1: Kosten der verschiedenen Verfahren in Rechenoperationen per Krylov-Unterraumdimension und Speicherbedarf in Vektoren

2.2.1 GMRES

Der GMRES-Algorithmus (Algorithmus 1, siehe [BBC$^+$94]) ist eine Verallgemeinerung des MINRES-Algorithmus auf unsymmetrische Matrizen. Dieser wiederum ist eine Verallgemeinerung des CG-Verfahrens für positiv definite symmetrische Matrizen auf indefinite symmetrische Matrizen.

Das CG-Verfahren generiert Folgen von Iterierten $\boldsymbol{x}_i$ sowie die entsprechenden Residuen $\boldsymbol{r}_i = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_i$ und Suchrichtungen $\boldsymbol{p}_i$, aus welchen neue Iterierte und Residuen bestimmt werden. Die neue Iterierte $\boldsymbol{x}_i$ und das Residuum $\boldsymbol{r}_i$ werden dabei mittels

$$\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \alpha_i \boldsymbol{p}_i, \qquad \boldsymbol{r}_i = \boldsymbol{r}_{i-1} + \alpha_i \boldsymbol{A}\boldsymbol{p}_i$$

bestimmt, wobei α_{i+1} so gewählt wird, daß $\langle \boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+1} \rangle_{\boldsymbol{A}^{-1}} := \boldsymbol{r}_{i+1}^T \boldsymbol{A}^{-1} \boldsymbol{r}_{i+1}$ minimiert wird. Dies leistet $\alpha_{i+1} = \boldsymbol{r}_{i-1}^T \boldsymbol{r}_{i-1} / \boldsymbol{p}_i^T \boldsymbol{A}\boldsymbol{p}_i$. Anschließend wird die neue Suchrichtung $\boldsymbol{p}_i$ bestimmt durch

$$\boldsymbol{p}_i = \boldsymbol{r}_i + \beta_{i-1} \boldsymbol{p}_{i-1},$$

wobei $\beta_i = \boldsymbol{r}_i^T \boldsymbol{r}_i / \boldsymbol{r}_{i-1}^T \boldsymbol{r}_{i-1}$ so gewählt ist, daß $\boldsymbol{r}_i$ und $\boldsymbol{r}_{i-1}$ orthogonal zueinander sind. Es kann sogar gezeigt werden, daß mit dieser Wahl von β_i das Residuum $\boldsymbol{r}_i$ orthogonal zu allen vorherigen Residuen $\boldsymbol{r}_l$, $0 \leq l < i$, ist. Die so generierten $\boldsymbol{x}_i$ minimieren in der Menge $\boldsymbol{x}_0 + \operatorname{span}\{\boldsymbol{r}_0, \ldots, \boldsymbol{A}^{i-1}\boldsymbol{r}_0\}$ den Fehler $\|\boldsymbol{x}_i - \boldsymbol{x}\|_{\boldsymbol{A}}^2 := (\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{A} (\boldsymbol{x}_i - \boldsymbol{x})$. Der Raum $\mathcal{K}_i := \operatorname{span}\{\boldsymbol{r}_0, \ldots, \boldsymbol{A}^{i-1}\boldsymbol{r}_0\}$ wird als *Krylov-Unterraum* bezeichnet.

Ist $\boldsymbol{A}$ indefinit oder sogar unsymmetrisch, so ist $\langle \cdot, \cdot \rangle_{\boldsymbol{A}}$ kein Skalarprodukt mehr und die erwähnten Minimierungseigenschaften sind nicht mehr wohldefiniert. Daher kann die Minimierung nicht mehr bezüglich $\langle \cdot, \cdot \rangle_{\boldsymbol{A}}$ erfolgen, es muß ein anderes Skalarprodukt verwendet werden. MINRES und GMRES verwenden dazu das Skalarprodukt $\langle \cdot, \cdot \rangle_2$. Während es bei symmetrischen Matrizen $\boldsymbol{A}$ noch möglich ist, entsprechende Folgen $(\boldsymbol{x}_i)_{i \in \mathbb{N}}$ bzw. $(\boldsymbol{r}_i)_{i \in \mathbb{N}}$ zu bestimmen, ohne alle vorhergehenden Iterierten bzw. Residuen zu kennen ($\rightarrow$MINRES), ist dies bei unsymmetrischen Matrizen nicht mehr möglich. Das GMRES-Verfahren generiert explizit eine Orthogonalbasis des Krylov-Raums $\mathcal{K}_i$ und speichert diese. Daraus wird ein $\boldsymbol{x}_i$ bestimmt, welches $\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_i\|_2$ in der Menge $\boldsymbol{x}_0 + \mathcal{K}_i$ minimiert. Da diese Vorgehensweise mit wachsendem i immer mehr Speicherplatz und Rechenzeit verbraucht, werden immer nur m Schritte des Algorithmus ausgeführt, bevor dieser mit dem vorherigen Ergebnis neu gestartet wird. Wird m zu klein gewählt, kann nicht immer Konvergenz des Algorithmus gewährleistet werden, wird m zu groß gewählt, so ist der Rechen- und Speicheraufwand (der linear in m ist) unnötig groß. Es gibt keine bestimmten Regeln, wie m zu wählen ist. Bei den vorliegenden Gleichungssystemen war ein Wert von 10 bis 15 angemessen.

2.2.2 BiCGStab(ℓ)

Das BiCG-Verfahren zur Lösung linearer Gleichungssysteme $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ mit unsymmetrischem $\boldsymbol{A}$ verfolgt einen anderen Ansatz als das GMRES-Verfahren und gibt dabei die Minimierungseigenschaften auf. Statt die neue Iterierte $\boldsymbol{x}_i$ aus der orthogonalen Folge aller vorherigen $\boldsymbol{r}_i$ zu bestimmen, wird $\boldsymbol{x}_i$ nun aus zwei zueinander orthogonalen Residuen $\boldsymbol{r}_i$ und $\tilde{\boldsymbol{r}}_i$ gewonnen, wobei $\tilde{\boldsymbol{r}}_i$ ähnlich wie beim CG-Verfahren, aber mit der transponierten Matrix $\boldsymbol{A}^T$ berechnet wird. Die fehlende Minimierungseigenschaft hat zur Folge, daß das Konvergenzverhalten des CG-Verfahrens unregelmäßig wird oder das Verfahren sogar divergiert.

Algorithmus 1 Der vorkonditionierte GMRES(m)-Algorithmus

$j = 0$
repeat
  $j = j + 1$
  $\boldsymbol{r} = \boldsymbol{P}^{-1}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x})$
  $\boldsymbol{v}^{(0)} = \boldsymbol{r}/\|\boldsymbol{r}\|_2$
  $\boldsymbol{s}_0 = \|\boldsymbol{r}\|_2, s_1 = \cdots = s_m = 0$
  for $i = 0, 1, \ldots, m-1$ **do**
    $\tilde{\boldsymbol{w}} = \boldsymbol{A}\boldsymbol{v}^{(i)}$
    $\boldsymbol{w} = \boldsymbol{P}^{-1}\tilde{\boldsymbol{w}}$
    for $k = 0, 1, \ldots, i-1$ **do**
      $h_{k,i} = \langle \tilde{\boldsymbol{w}}, \boldsymbol{v}^{(k)} \rangle$
      $\boldsymbol{w} = \boldsymbol{w} - h_{k,i}\boldsymbol{v}^{(k)}$
    end for
    $h_{i+1,k} = \|\boldsymbol{w}\|_2$
    $\boldsymbol{v}^{i+1} = \boldsymbol{w}/h_{i+1,i}$
    for $k = 0, 1, \ldots, i-1$ **do**
      ApplyPlaneRotation($h_{k,i}, h_{k+1,i}, c_k, s_k$)
    end for
    GeneratePlaneRotation($h_{i,i}, h_{i+1,i}, c_i, s_i$)
    ApplyPlaneRotation($h_{i,i}, h_{i+1,i}, c_i, s_i$), ApplyPlaneRotation($s_i, s_{i+1}, c_i, s_i$)
  end for
  Update($\boldsymbol{x}, m$)
until convergence

GeneratePlaneRotation(x : IN, y : IN, c : OUT, s : OUT)
if $y = 0$ **then**
  $c = 1, s = 0$
else
  if $|y| > |x|$ **then**
    $t = x/y, s = 1/\sqrt{1+t^2}, c = t \cdot s$
  else
    $t = y/x, c = 1/\sqrt{1+t^2}, s = t \cdot c$
  end if
end if

ApplyPlaneRotation(x : INOUT, y : INOUT, c : IN, s : IN)
$t = c \cdot x + s \cdot y$
$y = -s \cdot x + c \cdot y$
$x = t$

Update($\boldsymbol{x}$: INOUT, m : IN)
for $i = m, m-1, \ldots, 0$ **do**
  $y_i = s_i/h_{i,i}$
  for $j = i-1, i-2, \ldots, 0$ **do**
    $y_j = y_j - h_{j,i} \cdot y_i$
  end for
end for
for $j = 0, 1, \ldots, m$ **do**
  $\boldsymbol{x} = \boldsymbol{x} + y_j \cdot \boldsymbol{v}^{(j)}$
end for

Die Berechnung des neuen Residuums $\boldsymbol{r}_i$ beim BiCG-Verfahren kann geschrieben werden als $\boldsymbol{r}_i = p_i(\boldsymbol{A})\boldsymbol{r}_0$ mit einem Polynom i-ten Grades p_i. Die Beobachtung, daß die Anwendung von p_i das Residuum $\boldsymbol{r}_i$ verkleinert, kann dazu verleiten, diesen Operator zweimal anzuwenden, also $p_i^2(\boldsymbol{A})\boldsymbol{r}_0$ als neues Residuum zu verwenden. Dies liefert das CGS-Verfahren, welches in manchen Fällen tatsächlich doppelt so schnell konvergiert wie das BiCG-Verfahren, in anderen Fällen aber versagt, da nicht angenommen werden kann, daß $p_i(\boldsymbol{A})$ das BiCG-Residuum $p_i(\boldsymbol{A})\boldsymbol{r}_0$ reduziert.

Um dieses Verhalten in den Griff zu bekommen, wird im BiCGStab-Verfahren das neue Residuum $\boldsymbol{r}_i$ mit einem Polynom q_i bestimmt durch $q_i(\boldsymbol{A})p_i(\boldsymbol{A})\boldsymbol{r}_0$. Dabei ist q_i Produkt aus Polynomen ersten Grades $1 - \omega_i\boldsymbol{A}$, wobei ω_i so gewählt wird, daß es das Residuum $\boldsymbol{r}_i$ minimiert. Dieses Verfahren zeigt erheblich bessere Konvergenzeigenschaften als BiCG und CGS, zeigt aber Probleme, wenn ω_k nahe Null ist. Viele dieser Fälle lassen sich durch Verwendung von Polynomen höheren Grades vermeiden. Dies ist im BiCGStab(ℓ)-Verfahren (Algorithmus 2) realisiert, bei dem q_i Produkt von Polynomen ℓ-ten Grades ist. Weitere Details zu diesem Verfahren finden sich in [SF93].

Algorithmus 2 Der vorkonditionierte BiCGStab(ℓ)-Algorithmus

$k = -l$
choose $\boldsymbol{x}_0, \tilde{\boldsymbol{r}}_0$
$\boldsymbol{r}_0 = \boldsymbol{P}^{-1}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0)$
$\boldsymbol{u}_{-1} = 0, \rho_0 = 1, \alpha = 0, \omega = 1$
repeat
  $k = k + l$
  $\hat{\boldsymbol{u}}_0 = \boldsymbol{u}_{k-1}, \hat{\boldsymbol{r}}_0 = \boldsymbol{r}_k, \hat{\boldsymbol{x}}_0 = \boldsymbol{x}_k$
  $\rho_0 = -\omega\rho_0$
  for $j = 0, 1, \ldots, l-1$ **do**
    $\rho_1 = \langle \hat{\boldsymbol{r}}_j, \tilde{\boldsymbol{r}}_0 \rangle, \beta = \beta_{k+j} = \alpha\rho_1/\rho_0, \rho_0 = \rho_1$
    for $i = 0, 1, \ldots, j$ **do**
      $\hat{\boldsymbol{u}}_i = \hat{\boldsymbol{r}}_i - \beta \cdot \hat{\boldsymbol{u}}_i$
    end for
    $\hat{\boldsymbol{u}}_{j+1} = \boldsymbol{P}^{-1}\boldsymbol{A}\hat{\boldsymbol{u}}_j$
    $\gamma = \langle \hat{\boldsymbol{u}}_{j+1}, \tilde{\boldsymbol{r}}_0 \rangle, \alpha = \alpha_{k+j} = \rho_0/\gamma$
    for $i = 0, 1, \ldots, j$ **do**
      $\hat{\boldsymbol{r}}_i = \hat{\boldsymbol{r}}_i - \alpha \cdot \hat{\boldsymbol{u}}_{i+1}$
    end for
    $\hat{\boldsymbol{r}}_{j+1} = \boldsymbol{P}^{-1}\boldsymbol{A}\hat{\boldsymbol{r}}_j, \hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0 + \alpha \cdot \hat{\boldsymbol{u}}_0$
  end for
  for $j = 1, 2, \ldots, l$ **do**
    for $i = 1, 2, \ldots, j-1$ **do**
      $\tau_{ij} = \frac{1}{\sigma_i}\langle \hat{\boldsymbol{r}}_j \hat{\boldsymbol{r}}_i \rangle$
      $\hat{\boldsymbol{r}}_j = \hat{\boldsymbol{r}}_j - \tau_{ij} \cdot \hat{\boldsymbol{r}}_i$
    end for
    $\sigma_j = \langle \hat{\boldsymbol{r}}_j, \hat{\boldsymbol{r}}_j \rangle, \gamma'_j = \frac{1}{\sigma_j}\langle \hat{\boldsymbol{r}}_0, \hat{\boldsymbol{r}}_j \rangle$
  end for
  $\gamma_l = \gamma'_l, \omega = \gamma_l$
  for $j = l-1, \ldots, 1$ **do**
    $\gamma_j = \gamma'_j - \sum_{i=j+1}^{l} \tau_{ij}\gamma_i$
  end for
  for $j = 1, \ldots, l-1$ **do**
    $\gamma''_j = \gamma_{j+1} + \sum_{i=j+1}^{l-1} \tau_{ji}\gamma_{i+1}$
  end for
  $\hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0 + \gamma_1 \cdot \hat{\boldsymbol{r}}_0, \hat{\boldsymbol{r}}_0 = \hat{\boldsymbol{r}}_0 - \gamma'_l \cdot \hat{\boldsymbol{r}}_l, \hat{\boldsymbol{u}}_0 = \hat{\boldsymbol{u}}_0 - \gamma_l \cdot \hat{\boldsymbol{u}}_l$
  for $j = 1, \ldots, l-1$ **do**
    $\hat{\boldsymbol{u}}_0 = \hat{\boldsymbol{u}}_0 - \gamma_j \cdot \hat{\boldsymbol{u}}_j$
    $\hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_0 + \gamma''_j \cdot \hat{\boldsymbol{r}}_j$
    $\hat{\boldsymbol{r}}_0 = \hat{\boldsymbol{r}}_0 - \gamma'_j \cdot \hat{\boldsymbol{r}}_j$
  end for
  $\boldsymbol{u}_{k+l-1} = \hat{\boldsymbol{u}}_0, \boldsymbol{r}_{k+l} = \hat{\boldsymbol{r}}_0, \boldsymbol{x}_{k+l} = \hat{\boldsymbol{x}}_0$
until convergence

2.3 Gittergenerierung

Zur Durchführung der Diskretisierung ist es notwendig, ein *Gitter* zu berechnen, d.h die Abbildung $\boldsymbol{x} : G \to \Omega$ zu konstruieren. Wie schon erwähnt, wird diese nur auf den Eckpunkten der Zellen vorgegeben. In diesem Abschnitt werden Methoden vorgestellt, bei vorgegebenen Punkten auf den Randflächen ein Gitter im Inneren von Ω zu generieren. Dazu wird zunächst mittels *transfiniter Interpolation* (Abschnitt 2.3.1, [Lis99]) ein einfaches, i.A. wenig glattes und nicht unbedingt im Inneren des Gebiets liegendes Gitter erstellt, welches dann mit PDE-Methoden (Abschnitt 2.3.2, [Lis99, TWM85, NB]) *geglättet* und mit bestimmten Eigenschaften versehen werden kann (Abbildung 2.12).

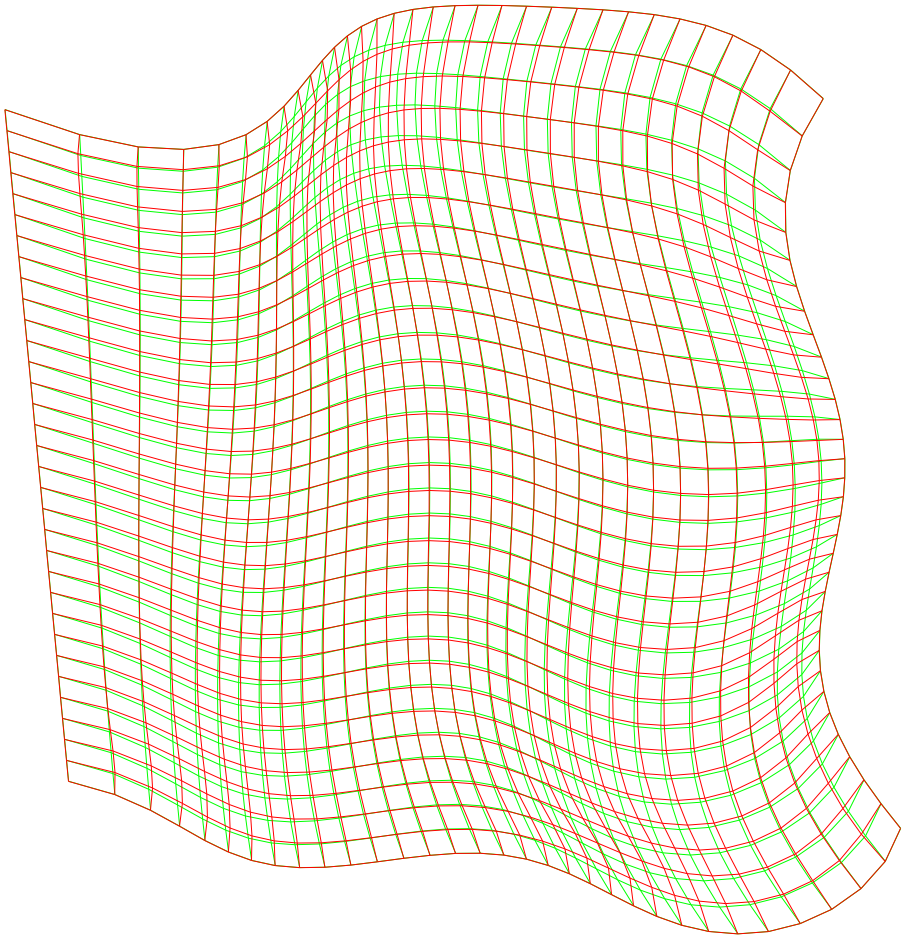

Abbildung 2.12: Gitter, generiert mit transfiniter Interpolation (grün) und anschließend geglättet mit leicht gewichteter Orthogonalität am Rand (rot)

2.3.1 Transfinite Interpolation

Unidirektionale Interpolation

Betrachte den Quader G und eine Koordinatenrichtung ξ^i, $1 \leq i \leq 3$. Sei $\boldsymbol{x}$ auf Ebenen orthogonal zu dieser Koordinatenrichtung in L_i Positionen $\bar{\xi}^i = \bar{\xi}^i_l$, $1 \leq l \leq L_i$, vorgegeben. Normalerweise ist $L_i = 2$ mit $\bar{\xi}^i_1 = 0$, $\bar{\xi}^i_2 = n^i$, d.h. auf beiden Rändern orthogonal zur Koordinatenrichtung ξ^i ist $\boldsymbol{x}$ vorgegeben. Die allgemeine Formulierung erlaubt es aber, das Gitter zum Beispiel auch in der Mitte des Quaders vorzugeben. Dies kann bei stark deformierten, insbesondere bei stark nichtkonvexen Gebieten notwendig sein, da man so sicherstellen kann, daß das erzeugte Gitter auch innerhalb des Gebiets Ω liegt. Die *unidirektionale Interpolation* $\boldsymbol{P}_i[\boldsymbol{x}](\xi) : G \to \mathbb{R}^3$ ist mit Koordinaten $\xi = (\xi^1, \xi^2, \xi^3)$ gegeben durch eine Linearkombination der vorgegebenen Werte, die an den Positionen $\bar{\xi}^i$ die vorgegebenen Werte reproduziert:

$$\boldsymbol{P}_i[\boldsymbol{x}](\xi) = \sum_{l=1}^{L_i} \alpha^i_l(\xi^i)\boldsymbol{x}(\xi|_{\xi^i=\bar{\xi}^i_l}),$$

wobei $\xi|_{\xi^i=\bar{\xi}^i_l}$ den i-ten Eintrag von ξ auf $\bar{\xi}^i_l$ festlegt, also

$$\xi|_{\xi^i=\bar{\xi}^i_l} = \begin{cases} \xi^j & \text{für } j \neq i \\ \bar{\xi}^j_l & \text{für } j = i \end{cases},$$

und $\alpha^i_l : [0, n^i] \to \mathbb{R}$ für jeden vorgegebenen Wert $\bar{\xi}^i$ vorgegebene glatte Überblendfunktionen sind, die

$$\alpha^i_l(\bar{\xi}^i_k) = \delta_{kl} \tag{2.37}$$

erfüllen, d.h. an Positionen, wo $\boldsymbol{x}$-Werte vorgegeben sind, sind nur die entsprechenden Funktionen mit Gewicht 1 aktiv. Für den Normalfall der Interpolation zwischen zwei Rändern ist dann

$$\boldsymbol{P}_i[\boldsymbol{x}](\xi) = \alpha^i_1(\xi^i)\boldsymbol{x}(\xi|_{\xi^i=\bar{\xi}^i_1}) + \alpha^i_2(\xi^i)\boldsymbol{x}(\xi|_{\xi^i=\bar{\xi}^i_2}).$$

Tensorprodukt

Die Komposition zweier unidirektionaler Interpolationen $\boldsymbol{P}_i[\boldsymbol{x}](\xi)$, $\boldsymbol{P}_j[\boldsymbol{x}](\xi)$ ist das *Tensorprodukt* $\boldsymbol{P}_i[\boldsymbol{x}] \otimes \boldsymbol{P}_j[\boldsymbol{x}](\xi)$ definiert durch

$$\begin{aligned}
\boldsymbol{P}_i[\boldsymbol{x}] \otimes \boldsymbol{P}_j[\boldsymbol{x}](\xi) &= \boldsymbol{P}_i[\boldsymbol{P}_j[\boldsymbol{x}]](\xi) = \sum_{l=1}^{L_i} \alpha^i_l(\xi^i)\boldsymbol{P}_j[\boldsymbol{x}](\xi|_{\xi^i=\bar{\xi}^i_l}) \\
&= \sum_{k=1}^{L_j}\sum_{l=1}^{L_i} \alpha^i_l(\xi^i)\alpha^j_k(\xi^j)\boldsymbol{x}(\xi|_{\xi^i=\bar{\xi}^i_l,\xi^j=\bar{\xi}^j_k}) \\
&= \sum_{k=1}^{L_j} \alpha^j_k(\xi^j)\boldsymbol{P}_i[\boldsymbol{x}](\xi|_{\xi^j=\bar{\xi}^j_k}) = \boldsymbol{P}_j[\boldsymbol{x}] \otimes \boldsymbol{P}_i[\boldsymbol{x}](\xi).
\end{aligned}$$

Bidirektionale und dreidimensionale Interpolation

Die *bidirektionale Interpolation*, die Werte in zwei Richtungen ξ^i und ξ^j interpoliert, ist definiert durch die boolesche Summe

$$\boldsymbol{P}_i[\boldsymbol{x}] \oplus \boldsymbol{P}_j[\boldsymbol{x}](\xi) = \boldsymbol{P}_i[\boldsymbol{x}](\xi) + \boldsymbol{P}_j[\boldsymbol{x}](\xi) - \boldsymbol{P}_i[\boldsymbol{x}] \otimes \boldsymbol{P}_j[\boldsymbol{x}](\xi).$$

Nachrechnen zeigt die beiden Relationen

$$\boldsymbol{P}_i[\boldsymbol{x}] \oplus \boldsymbol{P}_j[\boldsymbol{x}] = \boldsymbol{P}_j[\boldsymbol{x}] \oplus \boldsymbol{P}_i[\boldsymbol{x}] \quad \text{und} \quad \boldsymbol{P}_j[\boldsymbol{x}] - \boldsymbol{P}_i[\boldsymbol{x}] \otimes \boldsymbol{P}_j[\boldsymbol{x}] = \boldsymbol{P}_j[\boldsymbol{x} - \boldsymbol{P}_i[\boldsymbol{x}]]. \tag{2.38}$$

Eine dreidimensionale Interpolation in allen Richtungen ξ^1, ξ^2, ξ^3 ist durch die boolesche Summe aller unidirektionalen Interpolationen gegeben:

$$\boldsymbol{P}[\boldsymbol{x}] = \boldsymbol{P}_1[\boldsymbol{x}] \oplus \boldsymbol{P}_2[\boldsymbol{x}] \oplus \boldsymbol{P}_3[\boldsymbol{x}].$$

Es ist

$$\begin{aligned} \boldsymbol{P}[\boldsymbol{x}] \;=\; & \boldsymbol{P}_1[\boldsymbol{x}] + \boldsymbol{P}_2[\boldsymbol{x}] + \boldsymbol{P}_3[\boldsymbol{x}] \\ & - \boldsymbol{P}_1[\boldsymbol{x}] \otimes \boldsymbol{P}_2[\boldsymbol{x}] - \boldsymbol{P}_1[\boldsymbol{x}] \otimes \boldsymbol{P}_3[\boldsymbol{x}] - \boldsymbol{P}_2[\boldsymbol{x}] \otimes \boldsymbol{P}_3[\boldsymbol{x}] \\ & + \boldsymbol{P}_1[\boldsymbol{x}] \otimes \boldsymbol{P}_2[\boldsymbol{x}] \otimes \boldsymbol{P}_3[\boldsymbol{x}]. \end{aligned}$$

Mit (2.38) erhält man die rekursive Darstellung

$$\boldsymbol{P}[\boldsymbol{x}] = \boldsymbol{P}_1[\boldsymbol{x}] + \boldsymbol{P}_2[\boldsymbol{x} - \boldsymbol{P}_1[\boldsymbol{x}]] + \boldsymbol{P}_3[\boldsymbol{x} - \boldsymbol{P}_1[\boldsymbol{x}] - \boldsymbol{P}_2[\boldsymbol{x} - \boldsymbol{P}_1[\boldsymbol{x}]]].$$

Damit kann $\boldsymbol{P}[\boldsymbol{x}]$ durch eine Sequenz unidirektionaler Interpolationen berechnet werden:

$$\begin{aligned} \boldsymbol{F}_1[\boldsymbol{x}] &= \boldsymbol{P}_1[\boldsymbol{x}] \\ \boldsymbol{F}_2[\boldsymbol{x}] &= \boldsymbol{F}_1[\boldsymbol{x}] + \boldsymbol{P}_2[\boldsymbol{x} - \boldsymbol{F}_1[\boldsymbol{x}]] \\ \boldsymbol{P}[\boldsymbol{x}] &= \boldsymbol{F}_2[\boldsymbol{x}] + \boldsymbol{P}_3[\boldsymbol{x} - \boldsymbol{F}_2[\boldsymbol{x}]]. \end{aligned}$$

Für den Fall, daß $\boldsymbol{x}$ nur auf den Rändern vorgegeben ist, ergibt sich

$$\begin{aligned} \boldsymbol{F}_1[\boldsymbol{x}](\xi) &= \alpha_1^1(\xi^1)\boldsymbol{x}(0, \xi^2, \xi^3) + \alpha_2^1(\xi^1)\boldsymbol{x}(n^1, \xi^2, \xi^3) \\ \boldsymbol{F}_2[\boldsymbol{x}](\xi) &= \boldsymbol{F}_1[\boldsymbol{x}](\xi) + \alpha_1^2(\xi^2)\left[\boldsymbol{x}(\xi^1, 0, \xi^3) - \boldsymbol{F}_1[\boldsymbol{x}](\xi^1, 0, \xi^3)\right] \\ &\qquad + \alpha_2^2(\xi^2)\left[\boldsymbol{x}(\xi^1, n^2, \xi^3) - \boldsymbol{F}_1[\boldsymbol{x}](\xi^1, n^2, \xi^3)\right] \\ \boldsymbol{P}[\boldsymbol{x}](\xi) &= \boldsymbol{F}_2[\boldsymbol{x}](\xi) + \alpha_1^3(\xi^3)\left[\boldsymbol{x}(\xi^1, \xi^2, 0) - \boldsymbol{F}_2[\boldsymbol{x}](\xi^1, \xi^2, 0)\right] \\ &\qquad + \alpha_2^3(\xi^3)\left[\boldsymbol{x}(\xi^1, \xi^2, n^3) - \boldsymbol{F}_2[\boldsymbol{x}](\xi^1, \xi^2, n^3)\right]. \end{aligned}$$

In der vorliegenden Implementierung sind die Funktionen α_i^j als lineare Funktionen, die (2.37) erfüllen, gewählt. Weitere Manipulationen an der Verteilung und Form der Zellen sind mit anderen Überblendfunktionen und Vorgabe weiterer $\boldsymbol{x}$-Werte innerhalb des Gebiets oder mit den im nächsten Abschnitt beschriebenen Methoden möglich.

2.3.2 Glättung der Gitter

Mit der im vorherigen Abschnitt beschriebenen Methode ist es möglich, Gitter explizit zu berechnen. Die Kontrolle bestimmter wünschenswerter Eigenschaften, zum Beispiel Glattheit der Transformationsabbildung oder Orthogonalität des Gitters, sind nur schwer zu erreichen. Eine andere Möglichkeit zur Berechnung von Gittern ist es, diese als Lösung einer partiellen Differentialgleichung mit geeigneten Randbedingungen zu betrachten. Für diesen Zweck bieten sich besonders elliptische Gleichungen an, denn

1. elliptische Gleichungen erfüllen das Maximumprinzip, d.h. Extrema der Lösung befinden sich nur auf dem Rand. Dies verhindert Überlappungen des Gitters mit sich selbst;
2. Lösungen elliptischer Gleichungen sind glatt;
3. es ist möglich, auf dem gesamten Rand Randwerte vorzugeben.

Um die erste Eigenschaft nutzen zu können, ist es notwendig, die Gleichungen im physikalischen Gebiet Ω zu formulieren. Die einfachste Variante ist die Laplacegleichung

$$\nabla^2 \xi = 0,$$

wobei $\xi : \Omega \to G$ die Umkehrabbildung zu $\boldsymbol{x}$ ist. Zur Kontrolle bestimmter Gittereigenschaften ist es üblich ([Lis99, TWM85]), ein Poissonsystem mit Kontrollfunktion $\boldsymbol{P} : \Omega \to \mathbb{R}^3$ zu verwenden:

$$\nabla^2 \xi = \boldsymbol{P} = (\boldsymbol{P}^1, \boldsymbol{P}^2, \boldsymbol{P}^3). \tag{2.39}$$

Ziel ist es, Orthogonalität und Verteilung der Zellen am Rand zu beeinflussen und dabei die Glattheit des Gitters möglichst wenig zu stören. In der Praxis wird daher die Funktion $\boldsymbol{P}$ zunächst in der Nähe des Randes bestimmt und dann über das gesamte Gebiet interpoliert. Der hier verfolgte Ansatz (nach [NB]) unterscheidet sich ein wenig von dieser Vorgehensweise und erlaubt auch Kontrolle des Gitters im Inneren des Gebiets. Da die tatsächliche Berechnung des Gitters nicht im physikalischen Gebiet Ω, sondern in G erfolgt, ist es notwendig, (2.39) in $\boldsymbol{x}$-Koordinaten zu schreiben. Dies liefert die nichtlineare Gleichung (siehe [TWM85], g^{ij} ist der in (2.7) definierte kontravariante metrische Tensor)

$$\begin{aligned} 0 &= \sum_{i=1}^{3}\sum_{j=1}^{3} g^{ij}\frac{\partial^2 \boldsymbol{x}}{\partial \xi^i \partial \xi^j} + \sum_{k=1}^{3} \boldsymbol{P}^k \frac{\partial \boldsymbol{x}}{\partial \xi^k} \\ &= g^{11}(\boldsymbol{x}_{\xi^1\xi^1} + \boldsymbol{P}^1 \boldsymbol{x}_{\xi^1}) + g^{22}(\boldsymbol{x}_{\xi^2\xi^2} + \boldsymbol{P}^2 \boldsymbol{x}_{\xi^2}) + g^{33}(\boldsymbol{x}_{\xi^3\xi^3} + \boldsymbol{P}^3 \boldsymbol{x}_{\xi^3}) \\ &\quad + 2g^{12}\boldsymbol{x}_{\xi^1\xi^2} + 2g^{13}\boldsymbol{x}_{\xi^1\xi^3} + 2g^{23}\boldsymbol{x}_{\xi^2\xi^3}. \end{aligned}$$

Dieses Gleichungssystem soll nun mit einem nichtlinearen Gauß-Seidel-Verfahren gelöst werden. Anwendung einer einfachen Finite-Differenzen-Diskretisierung liefert für

den Punkt $\boldsymbol{x}_{i,j,k}$

$$\begin{aligned} 0 &= g^{11}\big[\boldsymbol{x}_{i-1,j,k} - 2\boldsymbol{x}_{i,j,k} + \boldsymbol{x}_{i+1,j,k} + \frac{1}{2}\boldsymbol{P}^1(\boldsymbol{x}_{i+1,j,k} - \boldsymbol{x}_{i-1,j,k})\big] + \\ & g^{22}\big[\boldsymbol{x}_{i,j-1,k} - 2\boldsymbol{x}_{i,j,k} + \boldsymbol{x}_{i,j+1,k} + \frac{1}{2}\boldsymbol{P}^2(\boldsymbol{x}_{i,j+1,k} - \boldsymbol{x}_{i,j-1,k})\big] + \\ & g^{33}\big[\boldsymbol{x}_{i,j,k-1} - 2\boldsymbol{x}_{i,j,k} + \boldsymbol{x}_{i,j,k+1} + \frac{1}{2}\boldsymbol{P}^3(\boldsymbol{x}_{i,j,k+1} - \boldsymbol{x}_{i,j,k-1})\big] + \\ & +2g^{12}\boldsymbol{x}_{\xi^1\xi^2} + 2g^{13}\boldsymbol{x}_{\xi^1\xi^3} + 2g^{23}\boldsymbol{x}_{\xi^2\xi^3}. \end{aligned}$$

Die Diskretisierung der gemischten Ableitungen in der letzten Zeile leistet keinen Beitrag zum Punkt $\boldsymbol{x}_{i,j,k}$. Auflösen dieser Gleichung nach $\boldsymbol{x}_{i,j,k}$ liefert

$$\boldsymbol{x}_{i,j,k} = \frac{g^{11}\boldsymbol{r}_1 + g^{22}\boldsymbol{r}_2 + g^{33}\boldsymbol{r}_3}{g^{11} + g^{22} + g^{33}}$$

mit

$$\begin{aligned} \boldsymbol{r}_1 &= \frac{1}{2}\big[(1 - \frac{1}{2}\boldsymbol{P}^1)\boldsymbol{x}_{i-1,j,k} + (1 + \frac{1}{2}\boldsymbol{P}^1)\boldsymbol{x}_{i+1,j,k}\big] + \frac{1}{2}\frac{g^{12}}{g^{11}}\boldsymbol{x}_{\xi^1\xi^2} + \frac{1}{2}\frac{g^{13}}{g^{11}}\boldsymbol{x}_{\xi^1\xi^3} \\ \boldsymbol{r}_2 &= \frac{1}{2}\big[(1 - \frac{1}{2}\boldsymbol{P}^2)\boldsymbol{x}_{i,j-1,k} + (1 + \frac{1}{2}\boldsymbol{P}^2)\boldsymbol{x}_{i,j+1,k}\big] + \frac{1}{2}\frac{g^{12}}{g^{22}}\boldsymbol{x}_{\xi^1\xi^2} + \frac{1}{2}\frac{g^{23}}{g^{22}}\boldsymbol{x}_{\xi^2\xi^3} \\ \boldsymbol{r}_3 &= \frac{1}{2}\big[(1 - \frac{1}{2}\boldsymbol{P}^3)\boldsymbol{x}_{i,j,k-1} + (1 + \frac{1}{2}\boldsymbol{P}^3)\boldsymbol{x}_{i,j,k+1}\big] + \frac{1}{2}\frac{g^{13}}{g^{33}}\boldsymbol{x}_{\xi^1\xi^3} + \frac{1}{2}\frac{g^{23}}{g^{33}}\boldsymbol{x}_{\xi^2\xi^3}. \end{aligned}$$

Dies liefert das gewünschte iterative Verfahren zur Lösung von (2.39). Man erkennt, daß die Kontrollfunktionen den Mittelwert zweier gegenüberliegender Punkte (in den eckigen Klammern) gewichten. Statt nun erst $\boldsymbol{P}$ zu bestimmen und dann (2.39) zu lösen, werden die "Zielpunkte" direkt vorgegeben, zum Beispiel

$$\overline{\boldsymbol{r}}_1 = \frac{1}{2}\big[(1 + 2\alpha_1)(\boldsymbol{x}_{i-1,j,k} + \Delta\boldsymbol{x}_i^-) + (1 - 2\alpha_1)(\boldsymbol{x}_{i+1,j,k} - \Delta\boldsymbol{x}_i^+)\big]$$

mit Gewichtsfunktionen $\alpha_i : [0, n^i] \to [-\frac{1}{2}, \frac{1}{2}]$ und Vektoren $\Delta\boldsymbol{x}_i^{\pm}$, die von $\boldsymbol{x}_{i\pm1,j,k}$ in Richtung $\boldsymbol{x}_{i,j,k}$ zeigen und den gewünschten Eigenschaften, zum Beispiel Orthogonalität oder Zellengröße, entsprechen (Abbildung 2.13). In bestimmten Bereichen des Gitters wird Kontrolle dieser Eigenschaften erwünscht sein, in manchen wird nur Glattheit gefordert werden. Daher ist es wünschenswert, dort die Laplacegleichung zu lösen. Daher wähle mit Gewichtsfunktionen $\beta_i : [0, n^i] \to [0, 1]$

$$\begin{aligned} \boldsymbol{r}_1 &= (1 - \beta_1)\cdot\frac{1}{2}(\boldsymbol{x}_{i-1,j,k} + \boldsymbol{x}_{i+1,j,k}) + \beta_1\cdot\overline{\boldsymbol{r}}_1 + \frac{1}{2}\frac{g^{12}}{g^{11}}\boldsymbol{x}_{\xi^1\xi^2} + \frac{1}{2}\frac{g^{13}}{g^{11}}\boldsymbol{x}_{\xi^1\xi^3} \\ \boldsymbol{r}_2 &= (1 - \beta_2)\cdot\frac{1}{2}(\boldsymbol{x}_{i,j-1,k} + \boldsymbol{x}_{i,j+1,k}) + \beta_2\cdot\overline{\boldsymbol{r}}_2 + \frac{1}{2}\frac{g^{12}}{g^{22}}\boldsymbol{x}_{\xi^1\xi^2} + \frac{1}{2}\frac{g^{23}}{g^{22}}\boldsymbol{x}_{\xi^2\xi^3} \\ \boldsymbol{r}_3 &= (1 - \beta_3)\cdot\frac{1}{2}(\boldsymbol{x}_{i,j,k-1} + \boldsymbol{x}_{i,j,k+1}) + \beta_3\cdot\overline{\boldsymbol{r}}_3 + \frac{1}{2}\frac{g^{13}}{g^{33}}\boldsymbol{x}_{\xi^1\xi^3} + \frac{1}{2}\frac{g^{23}}{g^{33}}\boldsymbol{x}_{\xi^2\xi^3}. \end{aligned}$$

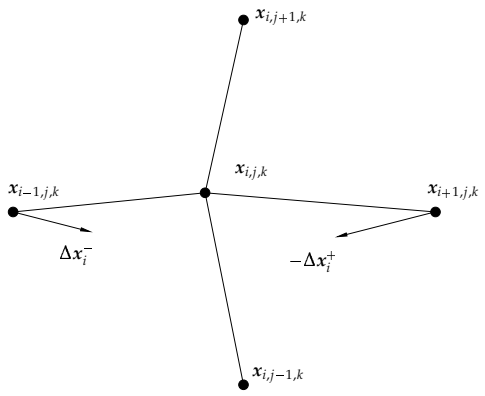

Abbildung 2.13: Gitterpunkt mit Zielpunkten

In der vorliegenden Implementierung sind die Gewichtsfunktionen gewählt als $\alpha_i(n) = \frac{n}{n^i} - \frac{1}{2}$ und $\beta_i(n) = \gamma_i(2\frac{n}{n^i} - 1)^6$ mit Konstanten $\gamma_i \in [0, 1]$.

2.4 NURBS

Im vorherigen Abschnitt wurde dargestellt, wie man aus vorgegebenen Punkten auf den Rändern ein Gitter im Inneren des Gebiets G erzeugt. Zunächst aber müssen diese Punkte auf den Rändern aus der Geometriebeschreibung generiert werden. Dazu ist eine parametrisierte Beschreibung von Flächen im dreidimensionalen Raum sehr hilfreich. NURBS (**N**on-**U**niform **R**ational **B**-**S**plines) sind in der Industrie de facto Standard für die Beschreibung geometrischer Flächen; auch hier erlaubt ihre Verwendung einen relativ einfachen Export von Geometriebeschreibungen aus CAD-Programmen in die in dieser Arbeit verwendete Darstellung. Dieser Erfolg ist in mehreren Punkten begründet:

- mit NURBS ist es möglich, sowohl "mathematische" Objekte wie Kreise, Zylinder etc. als auch "frei geformte" Flächen wie zum Beispiel Karosserieteile darzustellen;
- Konstruktion mit NURBS ist intuitiv, für elementare Designoperationen gibt es entsprechende einfache Algorithmen und umgekehrt;
- NURBS-Algorithmen sind schnell und numerisch stabil;
- NURBS-Kurven und Flächen sind invariant unter affinen Transformationen und paralleler und perspektivischer Projektion;
- NURBS sind eine Verallgemeinerung der normalen B-Splines und Bézierkurven bzw. Flächen.

Eine ausführliche Darstellung findet sich in [PT95].

Eine Bézierkurve vom Grad n ist definiert durch

$$\boldsymbol{C}(u) = \sum_{i=0}^{n} B_{i,n}(u)\boldsymbol{P}_i, \qquad 0 \leq u \leq 1.$$

Dabei sind die *Basisfunktionen* $B_{i,n}$ die n-ten Bernsteinpolynome

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!}u^i(1-u)^{n-i}.$$

Die Punkte $\boldsymbol{P}_i$ werden *Kontrollpunkte* genannt. Verbindet man die $\boldsymbol{P}_i$, so erhält man das sogenannte *Kontrollpolygon* (Abbildung 2.14). Soll eine komplexere Kurve erstellt wer-

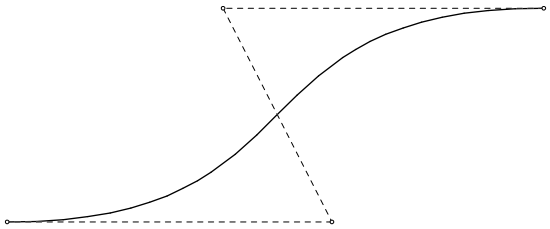

Abbildung 2.14: Bézierkurve vom Grad 3 mit Kontrollpolygon

den, so ist es ungünstig, dafür eine einzige Bézierkurve zu verwenden. Diese müßte hohen Grad haben und wäre damit nicht mehr lokal manipulierbar, d.h. lokale Änderungen würden sich auf die gesamte Länge der Kurve erstrecken. Daher geht man zu Kurven über, die stückweise aus Bézierkurven bestehen, nämlich B-Spline-Kurven (Abbildung 2.15). Diese sind definiert durch

$$\boldsymbol{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\boldsymbol{P}_i, \qquad a \leq u \leq b.$$

Dabei sind die $\boldsymbol{P}_i$ wieder die Kontrollpunkte und die $N_{i,p}$ die B-Spline-Basisfunktionen vom Grad p. Diese können auf verschiedene Arten definiert werden, eine für die Computerdarstellung effiziente ist die folgende rekursive Darstellung (mit $0/0 := 0$):

$$\begin{aligned} N_{i,0}(u) &= \begin{cases} 1 & \text{für } u_i \leq u \leq u_{i+1} \\ 0 & \text{sonst,} \end{cases} \\ N_{i,p}(u) &= \frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u). \end{aligned}$$

Die u_i stammen dabei aus dem *Knotenvektor*

$$U = (\underbrace{a,\ldots,a}_{p+1}, u_{p+1},\ldots,u_{m-p+1}, \underbrace{b,\ldots,b}_{p+1}).$$

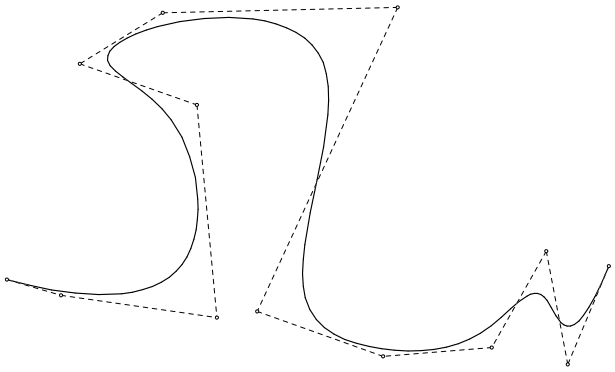

Abbildung 2.15: B-Spline-Kurve vom Grad 3 mit Kontrollpolygon

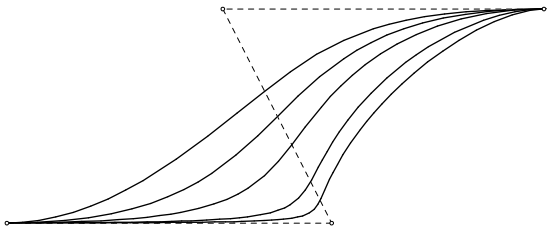

Abbildung 2.16: Einfluß der Gewichte: NURBS-Kurven mit Gewichten 0.5, 1, 2, 5 und 10 im Kontrollpunkt unten rechts

Es ist mit B-Splines nicht möglich, zum Beispiel Kreise exakt darzustellen. Dies ist jedoch mit einer Verallgemeinerung der B-Splines, den NURBS, möglich. Diese sind definiert durch

$$\boldsymbol{C}(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) w_i \boldsymbol{P}_i}{\sum_{i=0}^{n} N_{i,p}(u) w_i}, \qquad a \le u \le b$$

mit *Gewichten* $w_i > 0$. Der Einfluß der Gewichte ist in Abbildung 2.16 dargestellt. Mit

$$R_{i,p}(u) = \frac{N_{i,p}(u) w_i}{\sum_{j=0}^{n} N_{j,p}(u) w_j} \quad \text{folgt} \quad \boldsymbol{C}(u) = \sum_{i=1}^{n} R_{i,p}(u) \boldsymbol{P}_i.$$

NURBS-Kurven haben die folgenden hier interessanten Eigenschaften:

1. NURBS-Kurven sind invariant unter affinen Transformationen, d.h. eine solche angewendet auf die Kontrollpunkte liefert die gesamte transformierte Kurve. Dieser Zusammenhang gilt auch für perspektivische Projektionen.

2. Für $u \in [u_i, u_{i+1})$ liegt $\boldsymbol{C}(u)$ in der konvexen Hülle der Punkte $\boldsymbol{P}_{i-p}, \ldots, \boldsymbol{P}_i$.

3. Keine Ebene (Linie) hat mehr Schnittpunkte mit der Kurve als mit dem Kontrollpolygon. Dies heißt anschaulich, daß sich die Kurve nicht faltet.

4. Verschieben des Kontrollpunktes $\boldsymbol{P}_i$ oder des Gewichts w_i beeinflußt die Kurve nur im Bereich $u \in [u_i, u_{i+p+1}]$, d.h. die Kurve ist lokal manipulierbar.

NURBS-Flächen (Abbildung 2.17) erhält man nun aus NURBS-Kurven durch

$$\boldsymbol{S}(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_{i,j}\boldsymbol{P}_{i,j}}{\sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)w_{i,j}}, \qquad 0 \le u,v \le 1.$$

Die Kontrollpunkte $\boldsymbol{P}_{i,j}$ bilden dabei ein Kontrollnetz mit entsprechenden Gewichten $w_{i,j}$. Die für NURBS-Kurven genannten Eigenschaften übertragen sich im Wesentlichen (bis auf die dritte). Mit

$$R_{i,j}(u,v) = \frac{N_{i,p}(u)N_{j,q}(v)w_{i,j}}{\sum_{k=0}^{n}\sum_{l=0}^{m} N_{k,p}(u)N_{l,q}(v)w_{k,l}}$$

folgt wieder die Darstellung $\boldsymbol{S}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} R_{i,j}(u,v)\boldsymbol{P}_{i,j}$.

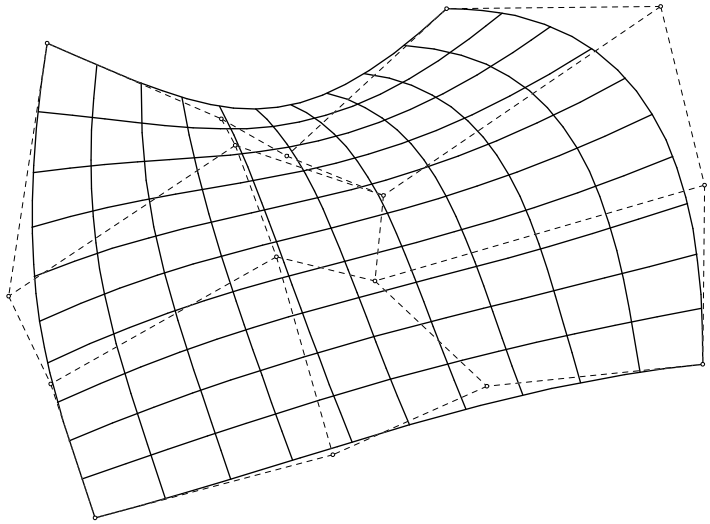

Abbildung 2.17: NURBS-Fläche mit Kontrollnetz

Kapitel 3

Optimierung

Aufgabe der Optimierung ist es, eine Anzahl Parameter so zu bestimmen, daß bei Beachtung gewisser Beschränkungen der Wert einer vorgegebenen, von den Parametern abhängigen Zielfunktion minimiert wird. Je nach Anzahl der Parameter, Charakter der beschränkenden Nebenbedingungen (z.B. Gleichungs- und Ungleichungsnebenbedingungen, Ganzzahligkeitsforderungen) oder Art der Zielfunktion haben diese Optimierungsaufgaben vollkommen unterschiedliche Strukturen.

In Abschnitt 3.1 werden die in dieser Arbeit behandelten Optimierungsprobleme formalisiert und es werden Optimalitätskriterien angegeben (vgl. [GT93]). In Abschnitt 3.2 werden Algorithmen zur Behandlung unrestringierter Optimierungsprobleme vorgestellt, die schließlich in Abschnitt 3.3 zu den Algorithmen für restringierte Optimierungsprobleme führen.

3.1 Optimierungsaufgaben und Optimalitätskriterien

Ein Optimierungsproblem läßt sich abstrakt als Problem der Form

$$f(x) \to \min! \quad \text{mit } x \in G \tag{3.1}$$

beschreiben, wobei $f : G \to \mathbb{R}$ die *Zielfunktion* und $G \subset X$ eine Teilmenge des zugrunde liegenden Raums X (z.B. $X = \mathbb{R}^n$), die *zulässige Menge* oder der *zulässige Bereich* ist. Ein Element $x \in X$ heißt *zulässig* für (3.1), falls $x \in G$ gilt. Als *optimale Lösungen* werden $\overline{x} \in G$ bezeichnet, für die

$$f(\overline{x}) \leq f(x) \qquad \text{für alle } x \in G$$

gilt. Der zugehörige Funktionswert $f_{\min} = f(\overline{x})$ wird als *optimaler Wert* bezeichnet. *Lokale Optima* sind Punkte $\overline{x} \in G$, für die eine offene Umgebung $U(\overline{x})$ existiert, so daß

$$f(\overline{x}) \leq f(x) \qquad \text{für alle } x \in G \cap U(\overline{x}).$$

Offensichtlich genügt es, Minimierungsprobleme zu betrachten (sonst gehe über zu $-f(x)$).

Wesentliche Unterschiede in der Struktur der Optimierungsprobleme ergeben sich nun aus der Wahl der zulässigen Menge G. Ganzzahligkeitsforderungen lassen sich zum Beispiel mit der diskreten Menge $G = \mathbb{Z}^n$ formalisieren. Solche Probleme gehören zur Klasse der *diskreten Optimierungsprobleme* und sind nicht Thema dieser Arbeit. Im Gegensatz zu diskreten Problemen, die mit kombinatorischen Algorithmen behandelt werden können, beruhen die Lösungsverfahren für *stetige Optimierungsprobleme* auf lokaler Differenzierbarkeit von Ziel- und Restriktionsfunktionen.

Die in dieser Arbeit behandelte Optimierung von Rändern durchströmter Gebiete fordert Einschränkungen an die Form des Randes, zum Beispiel die Vorgabe eines Gebiets, in dem sich der Rand befinden soll. Es sind aber auch kompliziertere Einschränkungen denkbar, so zum Beispiel eine Beschränkung der Krümmung des Randes. Daher sollte die Wahl des Gebietes G in recht allgemeiner Form möglich sein. Dies wird geleistet durch

$$G = \{x \in \mathbb{R}^n \mid h(x) = 0, g(x) \geq 0\}, \tag{3.2}$$

wobei $h : \mathbb{R}^n \to \mathbb{R}^p$ und $g : \mathbb{R}^n \to \mathbb{R}^m$ sowie f als zweimal stetig differenzierbar angenommen werden[1]. Die beiden Abbildungen h und g heißen *Restriktionsabbildungen* und ihre Komponenten h_i und g_j *Restriktionsfunktionen*. Die einzelnen Forderungen $h_i(x) = 0$ bzw. $g_j(x) \geq 0$ nennt man *Nebenbedingungen* oder *Restriktionen*. Um die Darstellung einfach zu halten, reduziere die Gleichungsrestriktionen $h_i(x) = 0$ auf zwei Ungleichungsrestriktionen[2]: $h_i(x) \geq 0$ und $-h_i(x) \geq 0$. Weiterhin wird angenommen, daß f eine auf G beschränkte Niveaumenge hat, d.h. es existiert ein $x_0 \in G$, so daß die Niveaumenge $\{x \in G \mid f(x) \leq f(x_0)\}$ beschränkt ist. Dann hat das Optimierungsproblem (3.1) mindestens eine Lösung (vgl. [GT93, Korollar 1.1]).

Im folgenden sollen nun notwendige und hinreichende Kriterien für eine optimale Lösung $\overline{x}$ für (3.1) mit Nebenbedingungen (3.2) angegeben werden. Aus der Differenzierbarkeit von f ergibt sich zunächst das folgende einfache Kriterium (vgl. [GT93, Lemma 1.5]):

Lemma: Sei f differenzierbar im Punkt $\overline{x} \in G$. Ist $\overline{x}$ eine lokale Lösung des Optimierungsproblems (3.1), so gilt

$$\nabla f(\overline{x})^T d \geq 0 \qquad \text{für alle } d \in Z(\overline{x}).$$

Dabei bezeichnet $Z(\overline{x})$ den *Kegel der zulässigen Richtungen*, definiert durch

$$Z(\overline{x}) := \operatorname{cone}\{d \in X \mid \forall\, t \in [0,1] : \overline{x} + td \in G\}$$

mit

$$\operatorname{cone}(S) := \{x \in X \mid \exists\, s \in S, \lambda \geq 0 : x = \lambda s\}.$$

[1] Die Relation $\geq$ auf $\mathbb{R}^m$ ist punktweise zu verstehen: $\mathbb{R}^m \ni y \geq 0 :\Leftrightarrow y_i \geq 0$, wobei $1 \leq i \leq m$.

[2] Dieses Vorgehen ist für die algorithmische Behandlung nicht geeignet, da es die Anzahl der Nebenbedingungen erhöht und die Kondition der Aufgabe verschlechtert.

Die Definition der Menge $Z(\overline{x})$ ist recht unhandlich, das Kriterium vereinfacht sich jedoch unter geeigneten Zusatzbedingungen und wird mit weiteren Forderungen an f sogar hinreichend (vgl. [GT93, Lemma 1.6, Lemma 1.7]):

Lemma: Sei die zulässige Menge G *konvex* und sei f differenzierbar. Dann ist die Erfüllung der *Variationsungleichung*

$$\nabla f(\overline{x})^T(x - \overline{x}) \geq 0 \qquad \text{für alle } x \in G \tag{3.3}$$

notwendig dafür, daß $\overline{x} \in G$ lokale Lösung des Optimierungsproblems (3.1) ist. Ist die Zielfunktion f zusätzlich *konvex*, so ist (3.3) sogar hinreichend dafür, daß $\overline{x}$ optimale Lösung von (3.1) ist.

Sind die Restriktionsfunktionen g_i differenzierbar, so läßt sich der Kegel der zulässigen Richtungen durch den *Linearisierungskegel*

$$T(\overline{x}) := \text{cone}\{d \in X \mid g(\overline{x}) + \nabla g(\overline{x})^T d \geq 0\}$$

approximieren. Ferner definiere

$$T^0(\overline{x}) := \text{cone}\{d \in X \mid g(\overline{x}) + \nabla g(\overline{x})^T d > 0\}.$$

Es gilt $T^0(\overline{x}) \subset Z(\overline{x})$ (vgl. [GT93, Lemma 1.8]). Eine äquivalente Darstellung von $T(\overline{x})$ und $T^0(\overline{x})$ erhält man durch

$$\begin{aligned} T(\overline{x}) &= \{d \in X \mid \nabla g_i(\overline{x})^T d \geq 0, i \in I_0(\overline{x})\} \quad \text{bzw.} \\ T^0(\overline{x}) &= \{d \in X \mid \nabla g_i(\overline{x})^T d > 0, i \in I_0(\overline{x})\}. \end{aligned}$$

Dabei ist $I_0(\overline{x})$ die *(Index-)menge der aktiven Restriktionen* im Punkt $\overline{x} \in G$, definiert durch

$$I_0(\overline{x}) = \big\{i \in \{1, \ldots, m\} \mid g_i(\overline{x}) = 0\big\}.$$

Mit diesen Bezeichnungen läßt sich eine dem ersten Kriterium ähnliche Bedingung formulieren (vgl. [GT93, Lemma 1.9]):

Lemma: Seien f und g differenzierbar. Ist $\overline{x} \in G$ eine lokale Lösung des Optimierungsproblems (3.1), so gilt

$$\nabla f(\overline{x})^T d \geq 0 \qquad \text{für alle } d \in T^0(\overline{x}),$$

also äquivalent formuliert

$$\nabla g_i(\overline{x})^T d > 0, i \in I_0(\overline{x}) \quad \Longrightarrow \quad \nabla f(\overline{x})^T d \geq 0.$$

Genügt der zulässige Bereich der Regularitätsforderung für den Abschluß

$$\overline{Z(\overline{x})} = T(\overline{x}), \tag{3.4}$$

so gilt

$$\nabla f(\overline{x})^T d \geq 0 \qquad \text{für alle } d \in T(\overline{x})$$

oder äquivalent

$$\nabla g_i(\overline{x})^T d \geq 0, i \in I_0(\overline{x}) \qquad \Longrightarrow \quad \nabla f(\overline{x})^T d \geq 0.$$

Die Bedingung (3.4) wird *constraint qualification* genannt. Für die praktische Überprüfung werden einfachere, hinreichende Kriterien eingesetzt:

(MFCQ) Mangasarian-Fromowitz constraint qualification. Die Gradienten $\nabla g_i(\overline{x})$, $i \in I_0(\overline{x})$, der aktiven Restriktionen sind positiv linear unabhängig, d.h.

$$\sum_{i \in I_0(\overline{x})} \alpha_i \nabla g_i(\overline{x}) = 0, \alpha_i \geq 0, i \in I_0(\overline{x}) \quad \Longrightarrow \quad \alpha_i = 0, i \in I_0(\overline{x}).$$

(SLB) Slater-Bedingung. Die Restriktionsfunktionen g_i, $1 \leq i \leq m$, sind konvex und es existiert ein $\hat{x} \in \mathbb{R}^n$ mit $g_i(\hat{x}) < 0$, $1 \leq i \leq m$.

In Verbindung mit der LAGRANGEschen Multiplikatorregel angewendet auf die Lagrange-Funktion $L(x, \lambda) = f(x) - \sum_{i \in I_0(x)} \lambda_i g_i(x)$ folgt aus diesem Lemma die Hauptaussage dieses Abschnitts (vgl. [GT93, Satz 1.2, Korollar 1.2]):

Satz: Sei $\overline{x} \in G$ eine lokale Lösung des Optimierungsproblems (3.1) und sei die constraint qualification (3.4) erfüllt. Dann gelten die *Kuhn-Tucker-Bedingungen*: Es existieren Zahlen $\overline{\lambda}_i \geq 0$, $i \in I_0(\overline{x})$, so daß

$$\begin{aligned} \nabla f(\overline{x}) - \sum_{i \in I_0(\overline{x})} \overline{\lambda}_i \nabla g_i(\overline{x}) &= 0 \\ \sum_{i \in I_0(\overline{x})} \overline{\lambda}_i g_i(\overline{x}) &= 0. \end{aligned}$$

Sind die Funktionen f und g_i, $1 \leq i \leq m$, zusätzlich konvex, so sind die Kuhn-Tucker-Bedingungen hinreichend für die Optimalität von $\overline{x}$.

Abbildung 3.1 zeigt die geometrische Interpretation der Kuhn-Tucker-Bedingungen.

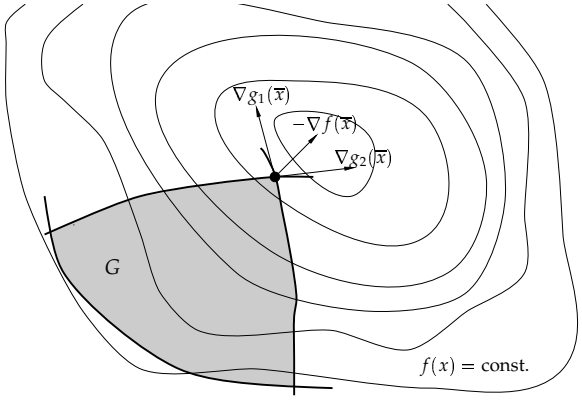

Abbildung 3.1: Geometrische Interpretation der Kuhn-Tucker-Bedingungen: $\nabla f(\overline{x})$ muß Element des Kegels sein, der durch die Gradientenvektoren $\nabla g_i(\overline{x})$ der aktiven Restriktionen aufgespannt wird (vgl. [GT93, Abbildung 1.3]).

3.2 Lösungsverfahren für unrestringierte Optimierungsprobleme

Das zu lösende Optimierungsproblem habe die Form

$$f(x) \to \min! \qquad \text{mit } x \in X. \tag{3.5}$$

Zur Lösung dieses Problems lassen sich sogenannte *Abstiegsverfahren* einsetzen. Diese sind iterative Verfahren und haben die folgende, hier am *Gradientenverfahren* verdeutlichte Struktur gemeinsam. Ausgehend von einem Startwert x^0 werden drei Schritte wiederholt ausgeführt:

1. Wähle eine *Abstiegsrichtung* d^k, zum Beispiel

$$d^k := -\nabla f(x^k).$$

2. Bestimme $\alpha_k > 0$ so, daß

$$f(x^k + \alpha_k d^k) \leq f(x^k + \alpha d^k) \qquad \text{für alle } \alpha \geq 0.$$

3. Setze

$$x^{k+1} := x^k + \alpha_k d^k.$$

Ist f Lipschitz-stetig differenzierbar, so erfüllt jeder Häufungspunkt $\overline{x}$ der Folge $(x_k)_{k\in\mathbb{N}}$ die Bedingung $\nabla f(\overline{x}) = 0$, also die Optimalitätsbedingung für (3.5) (vgl. [GT93, Satz 3.1]). Dieses mathematisch recht einfache Verfahren kann an verschiedenen Stellen verbessert und der praktischen Berechnung zugänglicher gemacht werden.

Die exakte Berechnung des Parameters α_k zur *Strahlminimierung* in Schritt 2 ist für beliebige f nicht ohne weiteres möglich, für die Konvergenz allerdings auch nicht notwendig. Anstelle des exakten Wertes kann mit dem folgenden *Armijo-Prinzip* ein ausreichender Wert berechnet werden. Sei $\delta \in (0,1)$ vorgegeben. Die *Armijo-Schrittweite* α_k zum Punkt x_k und zur Richtung d^k ist gegeben durch (vgl. Abbildung 3.2)

$$\alpha_k := \max\left\{\alpha \in (2^{-j})_{j\geq 0} \mid f(x^k + \alpha d^k) \leq f(x^k) + \alpha\delta\nabla f(x^k)^T d^k\right\}. \tag{3.6}$$

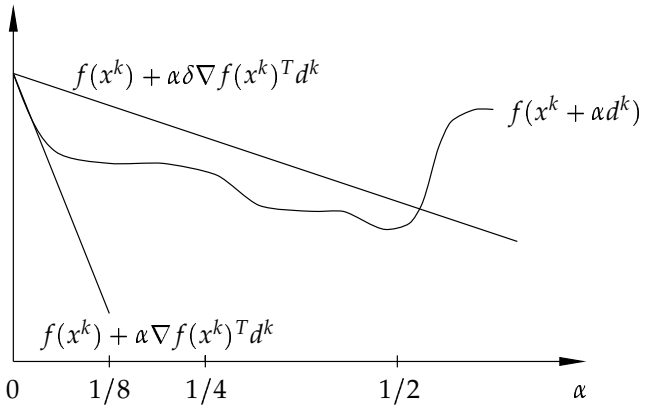

Abbildung 3.2: Geometrische Interpretation des Armijo-Prinzips: Sei $\varphi(\alpha) := f(x^k + \alpha d^k) - f(x^k)$, so gilt $\varphi'(0) = \nabla f(x^k)^T d^k$.

Legt man eine Taylorentwicklung der Zielfunktion im Iterationspunkt x^k bis zum ersten Glied zugrunde, so entspricht die Wahl von d^k in Schritt 1 der dafür stärksten Abstiegsrichtung. Es liegt nahe, statt nur des linearen Terms auch den quadratischen hinzuzunehmen. Die Taylorentwicklung hat dann die Form

$$F_k(x) = f(x^k) + \nabla f(x^k)^T(x - x^k) + \frac{1}{2}(x - x^k)^T\nabla^2 f(x^k)(x - x^k). \tag{3.7}$$

Ein x^{k+1}, welches $F_k(x)$ minimiert, genügt der Bedingung

$$\nabla f(x^k) + \nabla^2 f(x^k)(x^{k+1} - x^k) = 0.$$

Dies liefert

$$d^k = \nabla^2 f(x^k)^{-1}\nabla f(x^k).$$

Das *Newton-Verfahren*, welches sich damit ergibt, ist im Gegensatz zum Gradientenverfahren nur noch lokal (aber dafür sehr schnell) konvergent und bedarf daher einer Regularisierung. Dies kann erreicht werden, indem man $\nabla^2 f(x^k)$ ersetzt durch gleichmäßig positiv definite beschränkte Matrizen H_k, d.h. Matrizen, für die mit Konstanten $0 < m \leq M$ die Abschätzung

$$m\|z\|^2 \leq z^T H_k z \leq M\|z\|^2 \qquad \text{für alle } z \in \mathbb{R}^n \tag{3.8}$$

gilt. Mit diesen beiden Änderungen läßt sich folgendes Konvergenzresultat erzielen (vgl. [GT93, Satz 3.3]):

Satz: Es genüge die Folge der Matrizen $(H_k)_{k\in\mathbb{N}}$ der Bedingung (3.8). Dann gilt mit beliebigem Startpunkt $x_0 \in \mathbb{R}^n$ für jeden Häufungspunkt $\overline{x}$ der Folge $(x^k)_{k\in\mathbb{N}}$, die sich durch

$$d^k = -H_k^{-1}\nabla f(x^k), \qquad x^{k+1} := x^k + \alpha_k d^k$$

mit der durch (3.6) bestimmten Armijo-Schrittweite α_k ergibt, die Beziehung $\nabla f(\overline{x}) = 0$.

Die Konvergenzgeschwindigkeit eines solchen Verfahrens ist unter geeigneten Voraussetzungen *superlinear*, d.h. es existiert ein $c \in (0,1)$ mit

$$\|x^{k+1} - \overline{x}\| \leq c\|x^k - \overline{x}\|, \quad k \in \mathbb{N}.$$

Ein Problem bei diesem Verfahren ist die Bereitstellung der Matrizen H_k, die die Matrizen $\nabla^2 f(x^k)$ hinreichend gut approximieren. Es hat sich gezeigt, daß zur Sicherung der superlinearen Konvergenz lediglich eine gute Annäherung von $\nabla^2 f(x^k)$ in Richtung $s_k := x^{k+1} - x^k$ erforderlich ist. Taylorentwicklung liefert

$$\nabla^2 f(x^{k+1})s_k = \nabla f(x^{k+1}) - \nabla f(x^k) + O(\|s_k\|).$$

Unter Vernachlässigung des Restgliedes wird an die H_k daher die Forderung

$$H_{k+1}s_k = \nabla f(x^{k+1}) - \nabla f(x^k) =: q_k \tag{3.9}$$

gestellt, wobei H_0 eine beliebige symmetrische positiv definite Matrix ist. Verfahren, die dieser Forderung genügen, bezeichnet man als *Quasi-Newton-Verfahren* (siehe [Spe93b]). Die Matrix H_{k+1} soll aus der Matrix H_k durch additive Modifikation mit möglichst niedrigem Rang hervorgehen. Die einzige additive symmetrische Rang-1-Modifikation von H_k lautet

$$H_{k+1} = H_k + \frac{(q_k - H_k s_k)(q_k - H_k s_k)^T}{(q_k - H_k s_k)^T s_k}.$$

Diese wird als SR1-Formel bezeichnet, H_{k+1} ist aber nicht mehr notwendig positiv definit. Es ist also mindestens eine Modifikation vom Rang 2 erforderlich, um Symmetrie und Definitheit zu gewährleisten. Eine solche Methode erhält man in zwei Schritten. Zunächst wird die Abbildungseigenschaft von H_k in Richtung s_k annulliert, also $\tilde{H}_k s_k = 0$:

$$\tilde{H}_k := H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k}.$$

Da (3.9) erfüllt werden soll, ergibt sich notwendig die Konstruktion

$$H_{k+1} = \tilde{H}_k + \frac{q_k q_k^T}{q_k^T s_k}$$

mit der Zusatzbedingung $y_k^T s_k > 0$. Dies ergibt insgesamt die sogenannte *BFGS-Aufdatierung*[3]

$$H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \frac{q_k q_k^T}{q_k^T s_k}.$$

Die Inverse von H_{k+1} läßt sich rekursiv bestimmen durch

$$H_{k+1}^{-1} = H_k^{-1} + \Big(1 + \frac{s_k^T H_k^{-1} s_k}{q_k^T s_k}\Big)\frac{q_k q_k^T}{s_k^T q_k} - \frac{1}{s_k^T q}(q_k s_k^T H_k^{-1} + H_k^{-1} s_k q_k^T).$$

3.3 Lösungsverfahren für restringierte Optimierungsprobleme

Betrachte nun das nichtlineare restringierte Optimierungsproblem

$$f(x) \to \min!, \quad x \in G = \{x \in \mathbb{R}^n \mid g(x) \geq 0\}. \tag{3.10}$$

Lokale Lösungen von (3.10) werden (unter geeigneten Regularitätsvoraussetzungen, vgl. [GT93]) charakterisiert durch die in Abschnitt 3.1 dargestellten Kuhn-Tucker-Bedingungen. Diese haben mit der Lagrange-Funktion $L(x, \lambda) := f(x) - \lambda^T g(x)$ und der Voraussetzung *strenger Komplementarität*

$$g_i(\overline{x}) = 0 \qquad \Longleftrightarrow \qquad \overline{\lambda}_i > 0 \tag{3.11}$$

die Form

$$\nabla_x L(\overline{x}, \overline{\lambda}) = 0, \qquad \overline{\lambda}^T g(\overline{x}) = 0. \tag{3.12}$$

Sei $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n \times \mathbb{R}^m$ definiert durch

$$F(x, \lambda) := \Big(\frac{\partial}{\partial x_1} L(x, \lambda), \ldots, \frac{\partial}{\partial x_n} L(x, \lambda), \lambda_1 g_1(x), \ldots, \lambda_m g_m(x)\Big),$$

dann ist (3.12) äquivalent zu $F(\overline{x}, \overline{\lambda}) = 0$. Die Jacobimatrix

$$F'(x, \lambda) = \left(\begin{array}{c|ccc} \nabla_{xx}^2 L(x, \lambda) & -\nabla g_1(x) & \ldots & -\nabla g_m(x) \\ \hline \lambda_1 \nabla g_1(x)^T & g_1(x) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_m \nabla g_m(x)^T & 0 & \ldots & g_m(x) \end{array}\right) \tag{3.13}$$

ist im Punkt $(\overline{x}, \overline{\lambda})$ regulär ([GT93, Satz 2.3]). Dies legt nahe, zum Beispiel lokal das Newton-Verfahren zur Lösung von $F(x, \lambda) = 0$ zu verwenden. Ähnlich wie bei der

[3] BROYDEN, FLETCHER, GOLDFARB und SHANNO

Herleitung des Newton-Verfahrens im letzten Abschnitt kann man versuchen, die Zielfunktion quadratisch zu approximieren (siehe (3.7)) und unter Beibehaltung der Restriktionen einen neuen Wert x^{k+1} zu bestimmen. Dies geschieht durch Lösung des Optimierungsproblems

$$f(x^k) + \nabla f(x^k)^T(x - x^k) + \frac{1}{2}(x - x^k)^T \nabla^2 f(x^k)(x - x^k) \to \min! \qquad \text{mit } x \in G \quad (3.14)$$

und liefert das *Levitin-Poljak-Verfahren*. Da jedoch keine effektiven Lösungsalgorithmen für das Optimierungsproblem (3.14) bei beliebigen nichtlinearen Restriktionen bekannt sind, ist dieses Verfahren nicht praktikabel. Für lineare Restriktionen existieren jedoch solche Verfahren. Die sich damit ergebenden Optimierungsverfahren, die eine Folge quadratischer Optimierungsprobleme lösen, werden als *SQP-Verfahren*[4] bezeichnet. Ein einfaches Verfahren aus dieser Klasse wird nun hergeleitet.

Zunächst soll die Anwendung des Newton-Verfahrens auf das nichtlineare Gleichungssystem $F(x, \lambda) = 0$ untersucht werden. Dieses hat die Form ($k \geq 0$)

$$F(x^k, \lambda^k) + F'(x^k, \lambda^k) \begin{pmatrix} x^{k+1} - x^k \\ \lambda^{k+1} - \lambda^k \end{pmatrix} = 0. \quad (3.15)$$

Mit der Jacobimatrix (3.13) schreibt sich (3.15) als

$$\begin{aligned} \nabla_x L(x^k, \lambda^k) + \nabla^2_{xx}(x^{k+1} - x^k) + \nabla^2_{x\lambda} L(x^k, \lambda^k)(\lambda^{k+1} - \lambda^k) &= 0 \\ \lambda_i^k g_i(x^k) + \lambda_i^k \nabla g_i(x^k)^T(x^{k+1} - x^k) + g_i(x^k)(\lambda_i^{k+1} - \lambda_i^k) &= 0, \quad 1 \leq i \leq m. \end{aligned} \quad (3.16)$$

Wegen $L(x, \lambda) = f(x) - \sum_{i=1}^m \lambda_i g_i(x)$ folgt

$$\nabla^2_{x\lambda}(x^k, \lambda^k)(\lambda^{k+1} - \lambda^k) = -\sum_{i=1}^m (\lambda_i^{k+1} - \lambda_i^k) \nabla g_i(x^k)$$

und

$$\nabla_x L(x^k, \lambda^k) + \nabla^2_{x\lambda} L(x^k, \lambda^k)(\lambda^{k+1} - \lambda^k) = \nabla f(x^k) - \sum_{i=1}^m \lambda_i^{k+1} \nabla g_i(x^k).$$

Damit läßt sich (3.16) schreiben als

$$\begin{aligned} \nabla f(x^k) + \nabla^2_{xx} L(x^k, \lambda^k)(x^{k+1} - x^k) - \sum_{i=1}^m \lambda_i^{k+1} \nabla g_i(x^k) &= 0 \\ \lambda_i^{k+1} \left[g_i(x^k) + \nabla g_i(x^k)^T(x^{k+1} - x^k) \right] - (\lambda_i^{k+1} - \lambda_i^k) \nabla g_i(x^k)^T(x^{k+1} - x^k) &= 0, \quad 1 \leq i \leq m. \end{aligned} \quad (3.17)$$

Unter Vernachlässigung des quadratischen Gliedes in der zweiten Zeile von (3.17) ergibt sich die Iterationsvorschrift

$$\begin{aligned} \nabla f(x^k) + \nabla^2_{xx} L(x^k, \lambda^k)(x^{k+1} - x^k) - \sum_{i=1}^m \lambda_i^{k+1} \nabla g_i(x^k) &= 0 \\ \lambda_i^{k+1} \left[g_i(x^k) + \nabla g_i(x^k)^T(x^{k+1} - x^k) \right] &= 0, \quad 1 \leq i \leq m. \end{aligned} \quad (3.18)$$

[4] sequential quadratic programming

Sind ferner die Bedingungen

$$\lambda_i^{k+1} \quad \text{und} \quad g_i(x^k) + \nabla g_i(x^k)^T(x^{k+1} - x^k) \geq 0, \qquad 1 \leq i \leq m,$$

erfüllt, so bildet (x^{k+1}, λ^{k+1}) einen Kuhn-Tucker-Punkt (d.h. einen Punkt, der die Kuhn-Tucker-Bedingungen erfüllt) für das quadratische Optimierungsproblem

$$\begin{aligned} &f(x^k) + \nabla f(x^k)^T(x - x^k) + \frac{1}{2}(x - x^k)^T \nabla^2_{xx} L(x^k, \lambda^k)(x - x^k) \to \text{min!} \quad \text{mit} \\ &g_i(x^k) + \nabla g_i(x^k)^T(x - x^k) \geq 0, \quad 1 \leq i \leq m. \end{aligned} \tag{3.19}$$

Das Verfahren, welches ausgehend von einem Startpunkt (x^0, λ^0) die Iterierten (x^{k+1}, λ^{k+1}) durch Lösen des linear restringierten quadratischen Optimierungsproblems berechnet, wird als *Wilson-Verfahren* bezeichnet. Es ist in einer Umgebung des Optimums $(\overline{x}, \overline{\lambda})$ Q-quadratisch[5] konvergent ([GT93, Satz 6.4]).

Dieses Verfahren ist an verschiedenen Stellen verbesserungswürdig. Zum einen zeigt es nur lokale Konvergenz, ist also von einer guten Wahl des Startwertes abhängig. Daher sollte es mit geeigneten Methoden globalisiert werden. Dies kann zum Beispiel durch ähnliche Techniken wie bei unrestringierter Optimierung geschehen. Weiterhin kann, ähnlich zum Quasi-Newton-Verfahren, $\nabla^2_{xx} L(x^k, \lambda^k)$ durch entsprechende Näherungsverfahren approximiert werden. Dabei tritt in der Regel superlineare Konvergenz ein.

Da weitere Ausführungen zu diesem Themenbereich weit über den Rahmen dieser Arbeit hinausgehen würden, sei auf [Spe93b] verwiesen. Dort finden sich umfangreiche Analysen entsprechender Algorithmen. Das für diese Arbeit verwendete Verfahren enthält unter anderem diese Verbesserungen und ist eine Implementierung des in [Spe98] und [Spe93a] beschriebenen SQP-Algorithmus.

[5] d.h. für ein $c \in (0, 1)$ gilt $\|(x^{k+1} - \overline{x}, \lambda^{k+1} - \overline{\lambda})\| \leq c\|(x^k - \overline{x}, \lambda^k - \overline{\lambda})\|^2$.

Kapitel 4

Implementierung

4.1 Implementierung der Strömungsberechnung

4.1.1 Beschreibung der Algorithmen

Zur Abbildung der Geometrie wird die Methode der *blockstrukturierten Gitter* verwendet (vgl. Kapitel 2.1). Das gesamte Gebiet wird überdeckt mit an den Seitenflächen "zusammengeklebten" strukturierten Gittern (Abbildung 4.1), die als *Blöcke* bezeichnet werden. Dies bedeutet, daß eine Seitenfläche eines Blocks entweder Rand des Gebietes

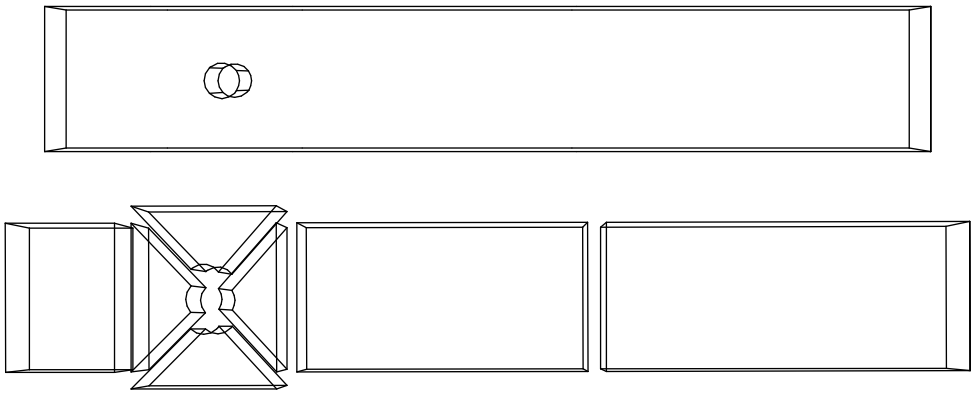

Abbildung 4.1: Geometrie des DFG-Benchmark (siehe auch Kapitel 5.1.2) zerlegt in sieben Blöcke

ist oder genau ein anderer Block ebenfalls diese Seitenfläche hat, so daß beide Seitenflächen dieselbe Gitterstruktur haben, sich also auch die Ränder der Zellen in diesen Seitenflächen genau überschneiden. Insbesondere müssen beide Blöcke in den beiden betreffenden Richtungen mit derselben Anzahl Zellen aufgelöst sein.

In der vorliegenden Implementierung muß die Seitenfläche des zweiten Blocks der Seitenfläche des ersten Blocks logisch gegenüberliegen (`IM`↔`IP`, `JM`↔`JP`, `KM`↔`KP`), und

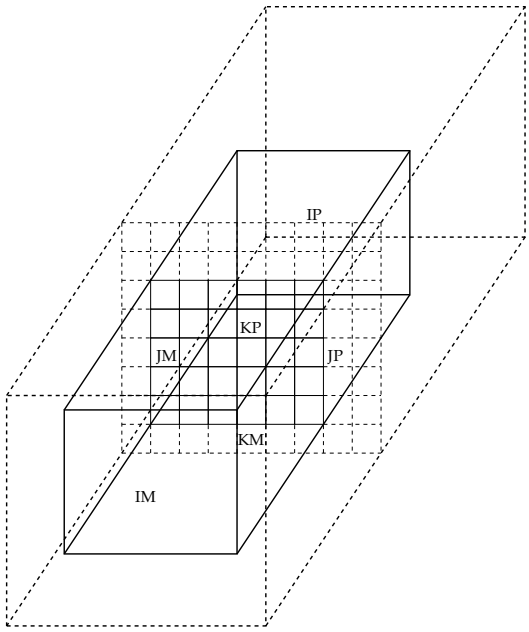

Abbildung 4.2: Block mit zwei Schichten "*ghost cells*" an den Rändern in Richtung aufsteigender Indizes (`IP`, `JP` und `KP`) und einer Schicht an den anderen Rändern (`IM`, `JM` und `KM`)

die Blöcke müssen gleich orientiert sein, d.h. die vier an die "Klebefläche" grenzenden, sich berührenden Seitenflächen der Blöcke müssen jeweils logisch dieselben sein. Dies reduziert die Anzahl notwendiger Fallunterscheidungen im Programm erheblich, ist aber keine wesentliche Einschränkung der Funktionalität. Weitere "Verklebemöglichkeiten" können später leicht hinzugefügt werden.

Zu einer Zelle Ω_j gehören die Variablen p_j und V_{j+e_α}, $1 \leq \alpha \leq 3$. Um Berechnungen wie die Auswertung von Operatoren auf den Blöcken zu vereinfachen, werden zusätzlich zu den eigentlich zu dem Gebiet des Blocks gehörenden Zellen am Rand eine oder mehrere Schichten von Zellen ("*ghost cells*") gespeichert[1] (Abbildung 4.2). Diese Zellen werden im Fall, daß die Seitenfläche zum Rand des gesamten Gebiets gehört, zur Implementierung der Randbedingungen benutzt. Andernfalls, also wenn der Block an dieser Stelle mit einem anderen Block "verklebt" ist, sind diese Zellen exakte Kopien der entsprechenden Zellen im Nachbarblock, inklusive aller dort gespeicherten Variablen (Abbildung 4.3). Dieser Nachbarblock muß nicht der sein, der mit dem ursprünglichen Block "verklebt" war, es kann sich auch um einen Block handeln, der mit dem ursprünglichen Block nur eine Kante oder sogar nur einen Punkt gemeinsam hat. Vor jeder Operatorauswertung am Rand ist also dafür zu sorgen, daß die entsprechenden Werte in den "*ghost cells*" zur Verfügung stehen. Dann kann der normale, auf dem strukturierten Gitter arbeitende Differenzenstern für den Operator angewendet werden.

[1] In der vorliegenden Implementierung werden an den Rändern `IM,JM,KM` eine Schicht und an den Rändern `IP,JP,KP` zwei Schichten benötigt.

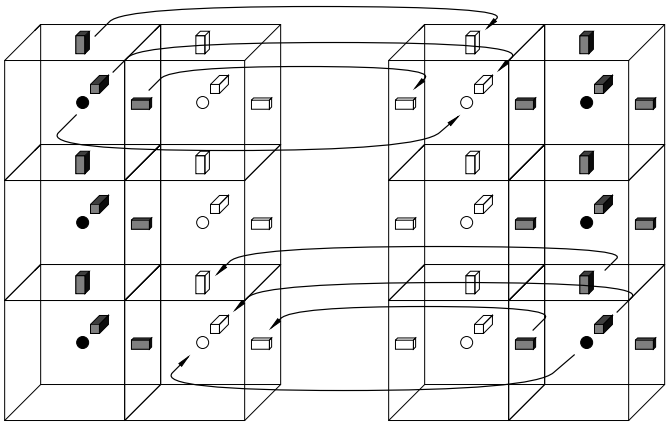

Abbildung 4.3: Zwei "verklebte" Blöcke mit "*ghost cells*" und vereinfacht für jeweils eine Zelle eingetragene Kopiervorgängen

Die Parallelisierung des Verfahrens erfolgte mit der auf fast allen Parallelrechnern und Rechnerclustern zur Verfügung stehenden Bibliothek MPI ("*message passing interface*", [Mes95]), die eine message-passing Umgebung implementiert. Die verschiedenen Prozesse kommunizieren durch Nachrichten, die sie sich gegenseitig (synchron oder asynchron) zuschicken. Zunächst werden die verschiedenen Blöcke möglichst gleichmäßig auf die zur Verfügung stehenden Prozessoren verteilt. Vor jeder Operatorauswertung wird dann das folgende asynchrone Kommunikationsschema angewendet:

1. Für alle "*ghost cells*", die Werte von anderen Prozessoren benötigen, initialisiere den Empfang von Daten (d.h. MPI wird darüber informiert, daß eine Nachricht mit den entsprechenden Daten erwartet wird und wo diese Daten zu speichern sind).

2. Für alle Kommunikationspartner von "*ghost cells*" aus dem ersten Schritt initialisiere das Senden der entsprechenden Nachrichten. Dies erfolgt asynchron, d.h. die Nachricht wird irgendwann in der Zukunft verschickt, wenn die entsprechenden Ressourcen zur Verfügung stehen.

3. Kopiere alle Werte in die "*ghost cells*", wenn Quell- und Zielblock von demselben Prozessor verwaltet werden.

4. Warte auf Beendigung des Sendens und Empfangens von Daten aus den Schritten (1) und (2).

Dieses Kommunikationsschema minimiert die Zeit, die das System auf Beendigung der Kommunikation warten muß, da – entsprechende Implementierungen von MPI vorausgesetzt – die Kommunikationsvorgänge entsprechend der zur Verfügung stehenden Ressourcen umgeordnet und gleichzeitig mit Schritt (3) ausgeführt werden können.

In dem nun skizzierten Hauptalgorithmus (Algorithmus 3) werden diese Kopiervorgänge in die "*ghost cells*" mit `copy`(*var*) bezeichnet, wobei *var* die entsprechende

zu kopierende Größe bezeichnet. Entsprechend symbolisiert `SetBC`(*var*) das Setzen von Randbedingungen in den "*ghost cells*". Für jede Operatorauswertung sind die folgenden Schritte hintereinander auszuführen: 1) `copy`(*var*) 2) `SetBC`(*var*) 3) Berechnung des Operators mit dem normalen Differenzenstern. Der Hauptalgorithmus ist eine direkte Umsetzung des in Kapitel 2.1.8 (S. 27) beschriebenen Druckkorrekturverfahrens. Für die Implementierung einer Zeitdiskretisierung höherer Ordnung sind die entsprechenden Berechnungen von $\tilde{V}$ und die Aktualisierung von p und V anzupassen. An der Struktur des Algorithmus sind jedoch keine Änderungen notwendig.

Algorithmus 3 Der Hauptalgorithmus zur Berechnung instationärer Strömungen

$t = 0, n = 0$
`copy`(V)
`SetBC`(V)
repeat
  bestimme Δt aus den Stabilitätskriterien (2.35) und (2.36)
  berechne $\tilde{V}$ mit (2.32): $\tilde{V} = V^n - \Delta t\left(\boldsymbol{M}^{-1}\boldsymbol{N}(V^n)V^n + \boldsymbol{G}p^{n-\frac{1}{2}}\right)$
  `copy`$(\tilde{V})$, setze auf dem Rand $\tilde{V} = V$
  berechne die rechte Seite $\texttt{RHS} = \frac{1}{\Delta t}\boldsymbol{D}\tilde{V}$ des Gleichungssystems (2.33)
  löse iterativ das lineare Gleichungssystem (2.33): $\boldsymbol{DG}q^n = \texttt{RHS}$
  aktualisiere p und V mit (2.34): $V^{n+1} = \tilde{V} - \Delta t\boldsymbol{G}q^n, \quad p^{n+\frac{1}{2}} = p^{n-\frac{1}{2}} + q^n$
  `copy`(V)
  `SetBC`(V)
  $t = t + \Delta t, n = n + 1$
until $t \geq t_{\text{fin}}$

4.1.2 Beschreibung der Programme

Die zur Strömungsberechnung notwendigen Arbeitsschritte sind in zwei verschiedenen Programmen implementiert. Zunächst wird mit dem Programm `nsclc` aus einer in einem Textformat (Anhang B) vorliegenden Beschreibung der Geometrie, der physikalischen Parameter wie Randbedingungen, Reynoldszahl, etc. und weiterer Parameter für die verwendeten numerischen Verfahren eine Binärdatei generiert, die als Eingabe für das Programm `nscl` dient. In diesem ist die eigentliche Strömungsberechnung implementiert, und im Gegensatz zum erstgenannten Programm liegt es in einer parallelisierten Version vor.

Die Implementierung aller Programme erfolgte unter `C++`, ein Überblick über die Klassenstruktur befindet sich weiter unten. An weiteren Hilfsmitteln und Bibliotheken wurden benutzt:

- `MPI` ("*message passing interface*", siehe [Mes95]), eine auf fast allen Parallelrechnern und Rechnerclustern verfügbare Bibliothek, die eine message-passing Umgebung implementiert.

- `ANTLR` (“*ANother Tool for Language Recognition*”, siehe [PLS98]), ein Parsergenerator für LL(k)-Grammatiken, zur Generierung der Parser für die Beschreibungs- und weiterer Eingabedateien.
- `NURBS++` (siehe [Lav99]), eine Bibliothek, die NURBS (vgl. Abschnitt 2.4) implementiert.
- `VTK` (“*visualization toolkit*”, siehe [SML97]), eine sehr umfangreiche Klassenbibliothek zur Visualisierung der anfallenden 3D-Daten.

Überblick über die Klassenstruktur

In diesem Abschnitt wird ein Überblick über die Klassenstruktur der Implementierung gegeben, siehe auch Abbildung 4.4. Neben den hier erwähnten Klassen gibt es noch einige Hilfsklassen (z.B. Exception-Klassen), die für das Verständnis der Programme weniger wichtig sind und daher hier nicht näher erklärt werden. Für weitere Informationen sei auf das mit `doxygen`[2] generierte Programmierhandbuch verwiesen.

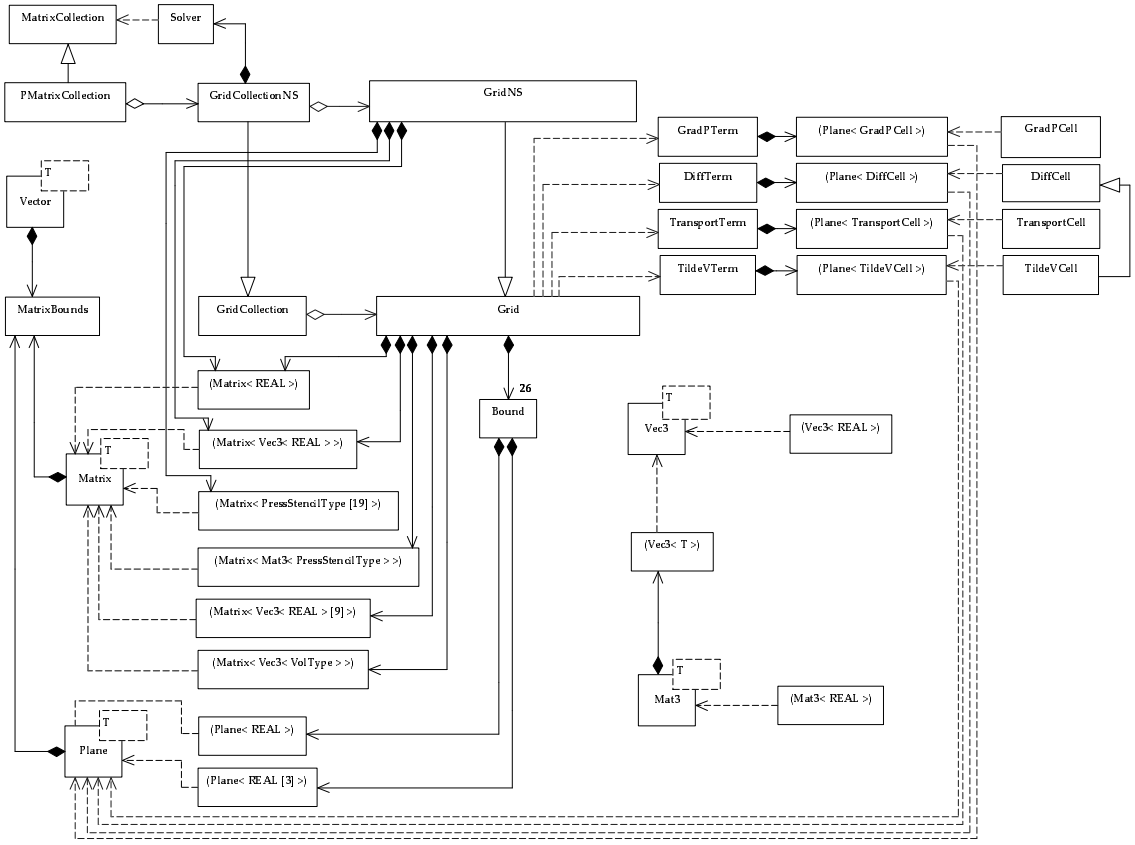

Abbildung 4.4: UML-Diagramm der wichtigsten Klassen

[2] → `http://www.doxygen.org`

Die wichtigsten Klassen ergeben sich aus der Struktur der blockstrukturierten Gitter, wie sie in Abschnitt 4.1.1 beschrieben ist. Blöcke werden durch die Klasse `Grid` modelliert. In ihr gespeichert sind in Form von dreidimensionalen Feldern (→`Matrix<T>`) die Gittertransformationsabbildung x, die Variablen V und p sowie einige geometrische Größen, die sich aus der Transformationsabbildung ergeben (Volumen der Ω_{j+e_α}, Differenzenstern für ∇p, etc.). Weiterhin erhält jede Instanz von `Grid` eine eindeutige Identifikationsnummer und die Nummer des Prozessors, auf dem sie sich befindet. Schließlich wird für jede der 26 Stellen (=6 Flächen+12 Kanten+8 Ecken), an denen ein anderer Block an den betrachteten grenzen kann, eine Instanz der Klasse `Bound` gespeichert. Die Klasse `Grid` enthält Funktionen zur Initialisierung, zur Berechnung verschiedener geometrischer und physikalischer Größen, die sich aus den gespeicherten Werten ergeben und für verschiedene Verfahren zur Strömungsberechnung relevant sind, Funktionen zur Behandlung der "*ghost cells*", also Setzen von Randwerten und Kommunikation mit anderen Blöcken, sowie einige weitere Hilfsfunktionen, die hier nicht erwähnt werden.

Von der Klasse `Grid` abgeleitet ist die Klasse `GridNS`, die im Wesentlichen die für die instationäre Strömungsberechnung nach dem Druckkorrekturverfahren (Algorithmus 3) benötigten Variablen und Funktionen auf Blockebene enthält.

Die verschiedenen Blöcke, modelliert durch `Grid` oder `GridNS` werden nun zusammengefaßt in der Klasse `GridCollection` bzw. in der von dieser abgeleiteten Klasse `GridCollectionNS`, die die gesamte Geometrie nebst aller notwendigen Parameter für das entsprechende Lösungsverfahren enthält. Diese Klassen enthalten eine Anzahl Funktionen, die einfach die entsprechenden Funktionen der einzelnen Blöcke aufrufen. Weiterhin implementieren diese Klassen die Hauptschleife der entsprechenden Lösungsalgorithmen.

Zur Modellierung der Ränder existiert die Klasse `Bound`. Diese enthält zum einen einen Verweis auf den Rand eines anderen Blocks, mit dem der Block "verklebt" ist, zu dem diese Instanz von `Bound` gehört, zum anderen Informationen darüber, welche Randbedingungen zu setzen sind, falls der Verweis auf den anderen Block nicht gesetzt ist, dieser Rand also zum Rand des physikalischen Gebiets gehört. Weiterhin enthält diese Klasse einige vorberechnete Werte, die zum Setzen der Randbedingungen benötigt werden und natürlich Funktionen, um diese vorzuberechnen.

Da die Berechnung einiger Operatoren auf Blockebene sehr speicherintensiv ist[3], ist es nicht effizient, diese durch einfaches Nacheinanderberechnen der benötigten Größen über das gesamte Gebiet des Blockes zu berechnen, da dies zum einen Auslagerung anderer benötigter Daten auf den Festplattenspeicher zur Folge haben kann und zum anderen der Cache-Architektur moderner Rechner, die auf die Lokalität von Datenzugriffen ausgelegt ist, wenig Rechnung trägt. Die Auswertung der Operatoren erfolgt nach einem Prinzip, das beide Faktoren berücksichtigt. Der Operator, der für den gesamten Quader aus Zellen, die den Block bilden, auszuwerten ist, wird schichtweise

[3]Für die Berechnung des diffusiven Terms werden pro Zelle allein 99 `REAL`-Variablen benötigt, die jeweis 4 oder 8 Byte einnehmen. Dies würde bei einer noch nicht besonders feinen Auflösung von 33^3 Zellen einen Speicherbedarf von etwa 32 MB allein für Hilfsvariablen bedeuten.

Operator	Klasse	Zellenklasse
Differenzenstern für ∇p; $\lvert\Omega_{j+\varepsilon_\alpha}\rvert$	`GradPStencil`	`GradPCell`
$\boldsymbol{M}^{-1}\boldsymbol{B}V$	`DiffTerm`	`DiffCell`
$\boldsymbol{M}^{-1}(\boldsymbol{A}(\hat{V})V - \boldsymbol{B}V)$	`TildeVTerm`	`TildeVCell`
$\boldsymbol{T}(V)\Phi$	`TransportTerm`	`TransportCell`

Tabelle 4.1: Übersicht über die Klassen zur schichtweisen Operatorberechnung mit Hilfsklassen für die Modellierung der Zellen

berechnet, wobei nur die Werte aus Zellschichten direkt neben der gerade aktuell zu berechnenden Schicht gespeichert werden. Weiterhin werden berechnete Werte, soweit möglich, sofort weiterverwendet, um Lokalität der Datenzugriffe zu gewährleisten[4]. Tabelle 4.1 gibt eine Übersicht über die auf diese Weise implementierten Operatoren.

Die iterativen linearen Gleichungslöser (vgl. Kapitel 2.2) sind in der Klasse `Solver` implementiert[5]. Die einzelnen Funktionen nehmen Instanzen von Klassen, die von der abstrakten Vektor-Basisklasse `MatrixCollection` abgeleitet sind, als Eingabe. Diese abgeleiteten Klassen müssen verschiedene arithmetische Funktionen, das Skalarprodukt, den auszuwertenden Operator und einen Vorkonditionierer implementieren. Für das Gleichungssystem (2.33) leistet dies die Klasse `PMatrixCollection`.

Weitere Datentypen sind in den Template-Klassen `Vec3<T>` (Vektoren im dreidimensionalen Raum), `Mat3<T>` (3×3-Matrizen), beide mit zahlreichen Hilfsfunktionen, `Vector<T>`, `Plane<T>` und `Matrix<T>` (ein-, zwei- und dreidimensionale Felder) implementiert.

4.1.3 Implementierung des Optimierungsverfahrens

Wie schon in Abschnitt 3.3 beschrieben, ist das verwendete Optimierungsverfahren ein SQP-Verfahren, dessen Details in [Spe98] und [Spe93a] beschrieben sind. Da die Implementierung eines solchen Verfahrens sehr aufwendig ist und eine eigene Implementierung hier keine besonderen Vorteile bietet, wurde die frei verfügbare[6] Implementierung `donlp2` von P. Spellucci entsprechend angepaßt. Eine Dokumentation dieses Programms befindet sich in [Spe95].

Die folgenden Anpassungen wurden vorgenommen:

- Zur Auswertung der Zielfunktion muß zunächst ein Strömungsproblem gelöst werden. Die entsprechenden Rechnungen sollen auf verschiedenen Knoten des Parallelrechners gestartet werden. Dazu ist eine Klasse[7] notwendig, die die verfügbaren Knoten verwaltet, Anforderungen für Funktionsauswertungen entgegen

[4] Mit diesem Verfahren konnte eine Beschleunigung der Berechnung der Operatoren um den Faktor ≈ 2 gegenüber der trivialen Methode erreicht werden.

[5] Es sind die Verfahren BiCGStab, BiCGStab(ℓ), GMRES und GMRESR implementiert.

[6] →`http://www.netlib.org`

[7] `Function` bzw. `NSCLFunction`

nimmt und auf die auf den Knoten wartenden Prozessinstanzen des Strömungslösers verteilt. Diese melden dann zurück, wenn die entsprechende Rechnung beendet ist.

Da dieses Schema nur relativ wenig Kommunikation erfordert, wird diese über das Dateisystem abgewickelt, da diese Lösung plattformunabhängig und einfach zu implementieren ist. Dazu erzeugt die Klasse `NSCLFunction` zunächst eine Eingabedatei für den Strömungslöser, indem sie in einer globalen Eingabedatei die Markierungen für die freien Parameter durch die entsprechenden Werte ersetzt. Danach wird eine weitere Datei erzeugt, auf die genau ein Strömungslöserprozess wartet. Dieser übernimmt die Eingabedatei und erzeugt nach Beendigung der Rechnung eine Datei, auf die der Optimiererprozeß wartet. Als letzter Schritt wird die Ergebnisdatei des Strömungslösers eingelesen und aus den Strömungsdaten der Wert der Zielfunktion berechnet.

Da, zum Beispiel wegen anderer Parameter für den Optimierer, ein Neustart des Optimierungsvorgangs notwendig sein kann, dieser aber viele Funktionswerte nochmals berechnen würde, werden schon berechnete Funktionswerte in einer einfachen Datenbank gespeichert und bei Bedarf abgefragt.

- Neben den Funktionswerten müssen auch Gradienten der Zielfunktion bestimmt werden. Dies geschieht durch eine schon in `donlp2` implementierte Finite-Differenzen-Approximation zweiter Ordnung. Diese mußte dahingehend angeändert werden, daß die entsprechenden Funktionsauswertungen parallel erfolgen. Für eine Auswertung des Gradienten sind $2n$ Funktionsauswertungen notwendig, wobei n die Anzahl der freien Parameter ist. Da die Geometrien dieser $2n$ Auswertungen sehr ähnlich sind, kann angenommen werden, daß auch die Strömungen ähnlich sind. Dies legt nahe, als Startwert für eine neue Auswertung eine zuvor berechnete Strömung zu verwenden, um schneller einen stationären Zustand zu erreichen. Der Effekt dieser Vorgehensweise soll in Abschnitt 5.2.3 überprüft werden.

  Weiterhin muß eine geeignete Schrittweite für die Finite-Differenzen-Approximation bestimmt werden. Dies ist in Abschnitt 5.2.1 dokumentiert.

- Zusätzlich zur Zielfunktion müssen auch die Constraints und ihre Ableitungen berechnet werden. Beide werden direkt in `donlp2` codiert.

Kapitel 5

Numerische Beispiele

5.1 Strömungsberechnung

5.1.1 Driven Cavity

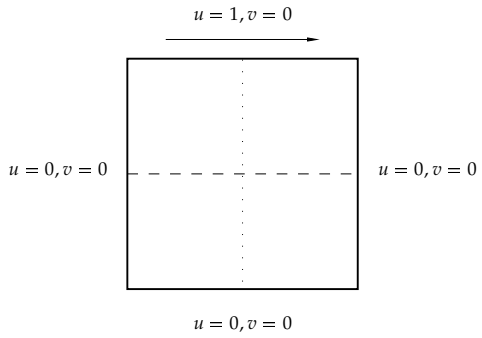

Abbildung 5.1: Geometrie der Driven Cavity

Dieses einfache Beispiel dient der Validierung des Codes zur Strömungsberechnung in einer einfachen Geometrie. Die Grundkonfiguration läßt sich Abbildung 5.1 entnehmen. Die Geometrie besteht aus einem Quadrat mit Seitenlänge 1, welches an der Oberseite angetrieben wird. Alle Ränder sind mit Dirichlet-Randbedingungen wie im Bild versehen. Da dieses Beispiel nur zweidimensional ist, wird es durch nur zwei Zellschichten im Dreidimensionalen realisiert. Die entsprechenden Seitenflächen sind mit Slip-Randbedingungen versehen. Die Rechnung wurde solange durchgeführt, bis ein stationärer Zustand erreicht wurde. Kriterium hierfür waren minimale Änderungen über mehrere Zeitschritte hinweg, genauer: $\|V_n - V_{n+1}\| < 10^{-6}$, $n_0 \leq n \leq n_0 + 5$.
Abbildung 5.2 zeigt die Geschwindigkeiten an den horizontalen und vertikalen Querschnitten durch das geometrische Zentrum der Cavity (die gestrichelte und gepunktete Linie in Abbildung 5.1) für die Reynoldszahlen 100 und 400, gerechnet auf einem 129×129-Gitter. Weiterhin sind Referenzwerte aus [GGS82] in Form von Punkten eingetragen. Es läßt sich eine weitgehende Übereinstimmung mit diesen Werten feststellen.

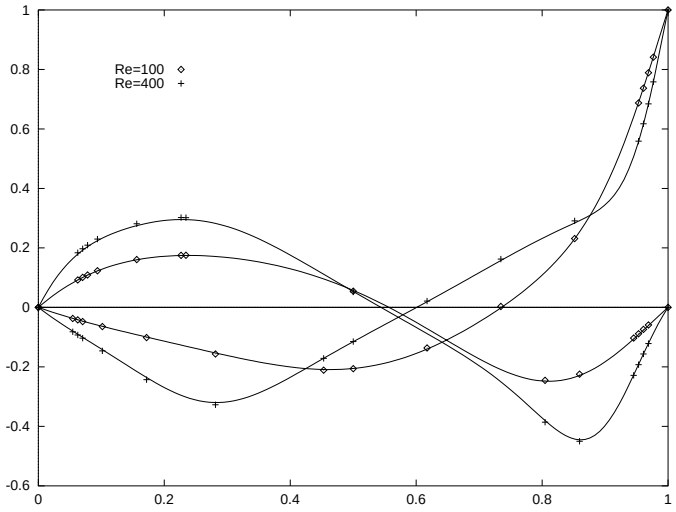

Abbildung 5.2: Geschwindigkeiten an den Querschnitten

Abbildung 5.4 zeigt die Stromlinien für diese Reynoldszahlen. Auch diese stimmen mit denen in [GGS82] gezeigten überein.

Für die Reynoldszahl 1000 finden sich genauere Ergebnisse in [BP98]. Die Stromlinien sind in Abbildung 5.4 dargestellt und zeigen gute Übereinstimmung mit der Literatur. Diese Konfiguration wurde nun für vier verschiedene Auflösungen (33x33, 65x65, 129x129 und 257x257) gerechnet. Die Geschwindigkeiten u durch den vertikalen Schnitt in Abbildung 5.1 sind in Tabelle 5.2 dargestellt, die horizontalen in 5.1. In den ersten beiden Spalten befinden sich y-Koordinate und Referenzwerte. In den folgenden vier Spalten sind die absoluten Fehler und schließlich die aus feinstem und grobstem Wert geschätzte Konvergenzrate eingetragen. In der Regel liegen diese Fehler für die höchste Auflösung unter einem Prozent. Weiterhin läßt sich Konvergenz etwas unter zweiter Ordnung beobachten, d.h. mit Halbierung der Gitterweite viertelt sich der Fehler. Dies zeigt sich besonders an Orten, wo der Fehler bei geringster Auflösungsstufe hoch ist.

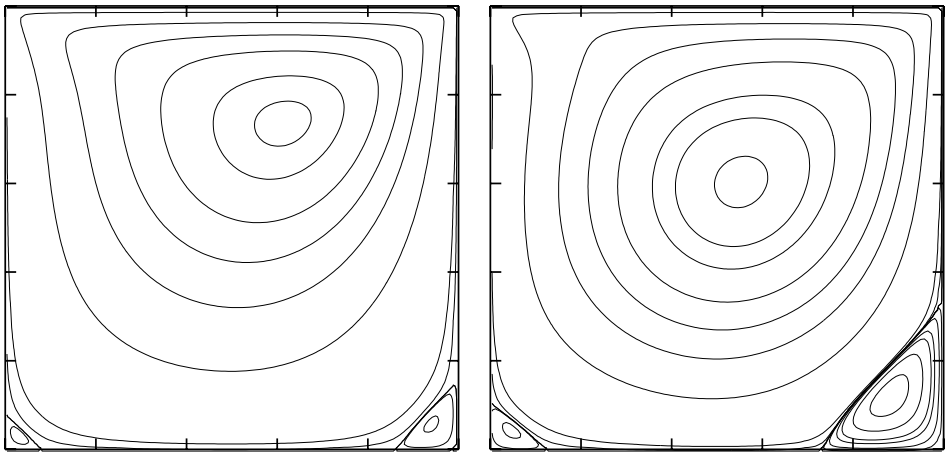

Abbildung 5.3: Stromlinien für Re = 100 und Re = 400, 129x129-Gitter

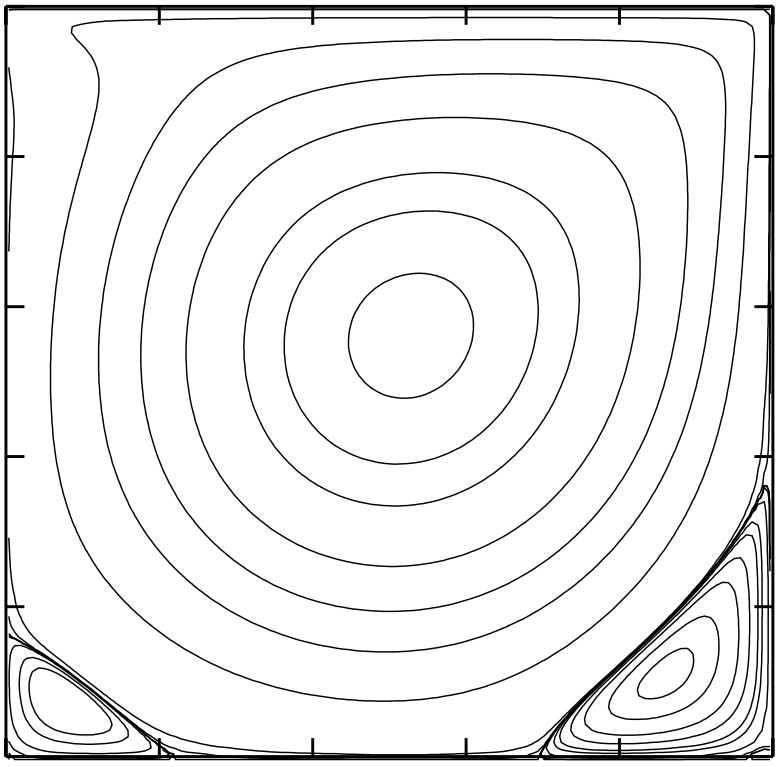

Abbildung 5.4: Stromlinien für Re = 1000, 129x129-Gitter

x	Ref. [BP98]	33x33	65x65	129x129	257x257	η
0.0625	0.280706	0.0586578	0.0190138	0.00516449	0.00135686	1.83504
0.0703	0.29627	0.0626604	0.0203636	0.00533287	0.00140181	1.85132
0.0781	0.30991	0.064739	0.0202552	0.0054933	0.00144573	1.85219
0.0938	0.333044	0.0661894	0.0214187	0.0058119	0.00153342	1.8343
0.1563	0.376919	0.0684477	0.0218692	0.00589314	0.0015759	1.83733
0.2266	0.333992	0.0445837	0.0137263	0.00363663	0.00101575	1.84245
0.2344	0.325359	0.0414282	0.0127676	0.00339272	0.000956573	1.83592
0.5	0.025799	0.0012911	0.00016625	4.28e-05	1.145e-05	2.30212
0.8047	-0.320214	0.0339507	0.0106296	0.00293604	0.000859196	1.79125
0.8594	-0.426455	0.0497158	0.0138608	0.00362416	0.000979166	1.9134
0.9063	-0.526439	0.0926892	0.0314585	0.00856421	0.00224942	1.81167
0.9453	-0.410375	0.0720428	0.0279589	0.00663327	0.00177254	1.80498
0.9531	-0.355321	0.0658264	0.0214912	0.00585479	0.00157382	1.81895
0.9609	-0.293687	0.05303	0.0195339	0.00508698	0.00137214	1.78045
0.9688	-0.227923	0.0367297	0.0153997	0.00425467	0.00113695	1.69312

Tabelle 5.1: Geschwindigkeitswerte v durch den horizontalen Schnitt in Abbildung 5.1, Referenzwerte aus [BP98], absolute Fehlern und geschätzte Konvergenzrate η für verschieden feine Gitter (Re = 1000)

y	Ref. [BP98]	33x33	65x65	129x129	257x257	η
0.0547	-0.181288	0.0352123	0.0106852	0.0028428	0.000736093	1.88437
0.0625	-0.20233	0.0390929	0.0119505	0.00326686	0.000848734	1.86593
0.0703	-0.222895	0.0446834	0.0137808	0.00371923	0.000967992	1.86699
0.1016	-0.300456	0.0652498	0.0209071	0.0056519	0.0014806	1.84441
0.1719	-0.388569	0.0752259	0.0234579	0.00628867	0.00166592	1.85627
0.2813	-0.283696	0.0282942	0.0117781	0.00559672	0.00400922	0.952006
0.4531	-0.1082	0.00882667	0.00322058	0.000859564	0.000300339	1.64702
0.5	-0.060561	0.00158235	0.0002076	0.0011583	0.00136085	0.0734697
0.6172	0.0570178	0.0108561	0.00348349	0.000956783	0.000295767	1.75532
0.7344	0.188675	0.0255084	0.00855908	0.00235428	0.000724414	1.73509
0.8516	0.337221	0.0502698	0.0169283	0.00460793	0.00130169	1.78008
0.9531	0.472333	0.00769561	0.0137626	0.0038598	0.00101711	0.985927
0.9609	0.516928	0.00358719	0.00406835	0.00327219	0.000844777	0.704508
0.9688	0.580836	0.0181678	0.0101507	0.00291032	0.000771393	1.53915
0.9766	0.664423	0.00219429	0.000465731	0.00254673	0.000663586	0.582663

Tabelle 5.2: Geschwindigkeitswerte u durch den vertikalen Schnitt in Abbildung 5.1, Referenzwerte aus [BP98], absolute Fehlern und geschätzte Konvergenzrate η für verschieden feine Gitter (Re = 1000)

5.1.2 DFG-Benchmark

Nachdem im ersten Beispiel die Strömungsberechnung in einer einfachen Geometrie überprüft wurde, soll nun die Strömungsberechnung in dreidimensionalen gekrümmten Koordinaten getestet werden. Zu diesem Zweck soll das Strömungsproblem "*The 1995-DFG benchmark: Flow around a cylinder*" (siehe [Tur99]) für den dreidimensionalen Fall gerechnet werden.

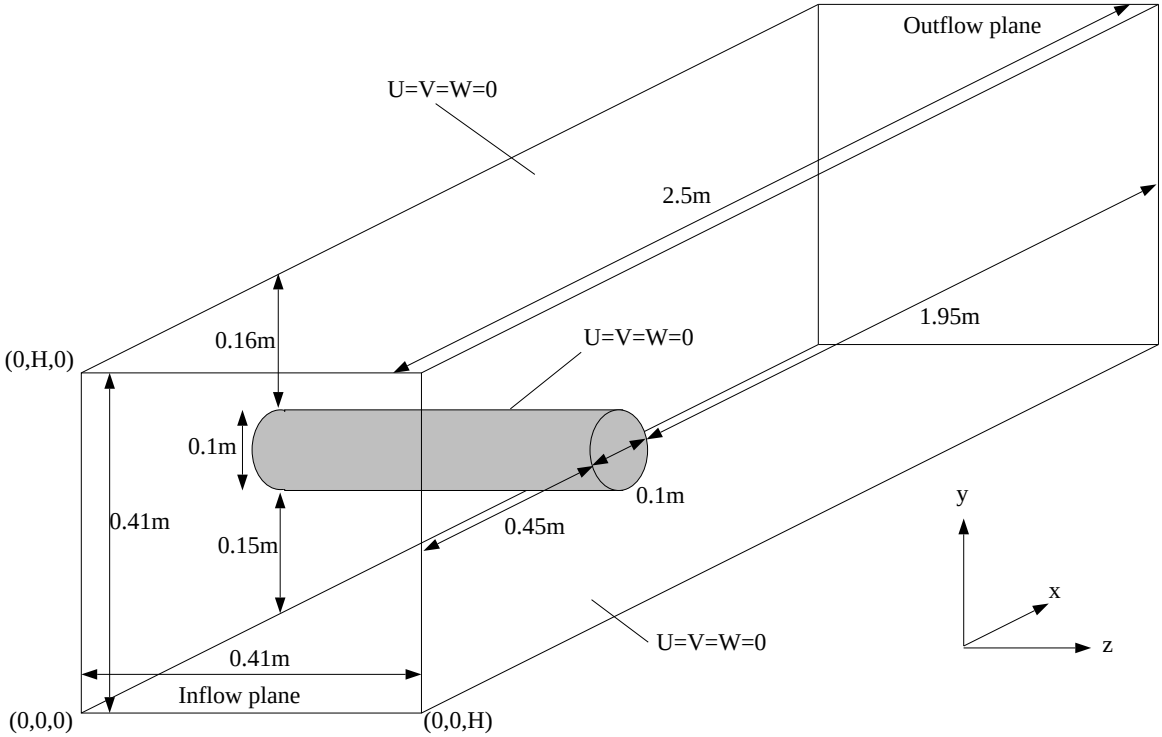

Abbildung 5.5: Geometrie und Randbedingungen für den DFG-Benchmark

Die Geometrie des DFG-Benchmarks ist in Abbildung 5.5 dargestellt. Die kinematische Zähigkeit des Fluids beträgt $\mu = 10^{-3}\, m^2/s$, die Dichte $\rho = 1\, kg/m^3$, die Höhe H beträgt $H = 0.41\, m$, der Durchmesser des Zylinders $D = 0.1\, m$ und die charakteristische Geschwindigkeit ist $\overline{U} = \frac{4}{9}\boldsymbol{u}(0, H/2, H/2)$. An der Vorderseite ("Inflow plane") sind Einström-Randbedingungen in Form eines parabolischen Geschwindigkeitsprofils gesetzt, nämlich

$$\boldsymbol{u}(0, y, z) = \big(16 U_m yz(H-y)(H-z)/H^4, 0, 0\big).$$

An der Rückseite ("Outflow plane") sind Ausström-Randbedingungen gesetzt. Alle anderen Flächen sind mit Haftrandbedingungen versehen.

Weiterhin definiere *drag*- und *lift*-Kräfte durch

$$F_D = \int_S (\rho\nu \frac{\partial \boldsymbol{u}^t}{\partial \boldsymbol{n}} \boldsymbol{n}_y - p\boldsymbol{n}_x)\, dS, \qquad F_L = -\int_S (\rho\nu \frac{\partial \boldsymbol{u}^t}{\partial \boldsymbol{n}} \boldsymbol{n}_x + p\boldsymbol{n}_y)\, dS,$$

wobei S die Oberfläche des Zylinders, $\boldsymbol{n}$ den äußeren Normalenvektor auf dieser Fläche und $\boldsymbol{u}^t$ die tangentiale Geschwindigkeit in Richtung $t = (\boldsymbol{n}_y, -\boldsymbol{n}_x, 0)$ bezeichnen. Die *drag*- und *lift*-Koeffizienten sind definiert durch

$$c_D = \frac{2F_D}{\rho \overline{U}^2 DH}, \qquad c_L = \frac{2F_L}{\rho \overline{U}^2 DH}.$$

Weiterhin sei die Druckdifferenz Δp definiert durch

$$\Delta p = p(0.45, 0.2, 0.205) - p(0.55, 0.2, 0.205).$$

Unter diesen Voraussetzungen sind zwei Rechnungen durchzuführen:

1. Berechne mit $U_m = 0.45\, m/s$ die Werte Δp, c_D und c_L. Bei dieser Rechnung wird eine stationäre Strömung erwartet.

2. Berechne mit $U_m = 2.25\, m/s$ die Werte Δp, c_D und c_L als Funktion der Zeit über $t = [t_0, t_0 + 3/f]$. Es wird eine periodische instationäre Strömung erwartet. Weiterhin berechne $c_L^{\max}$, $c_D^{\max}$ und die *Strouhal-Zahl* $St = Df/\overline{U}$, wobei f die Frequenz der Strömungsablösung ist.

Die Aufteilung des Gebiets in Blöcke wurde schon in Abbildung 4.1 (S. 55) dargestellt. Die erste Rechnung wurde in vier Auflösungen, nämlich mit $7 \cdot 9^3 = 5103$, $7 \cdot 17^3 = 34391$, $7 \cdot 33^3 = 251559$ und mit 373527 Zellen (Abbildung 5.6) durchgeführt. Für die ersten beiden Auflösungen wurde eine Upwind-Diskretisierung des konvektiven Terms verwendet, für die weiteren eine Diskretisierung mittels zentraler Differenzen. Startwert der Rechnung höherer Auflösung war jeweils das Ergebnis der Rechnung in der nächsten geringeren Auflösungsstufe. Abbildung 5.7 zeigt das Verhalten der berechneten Strömung.

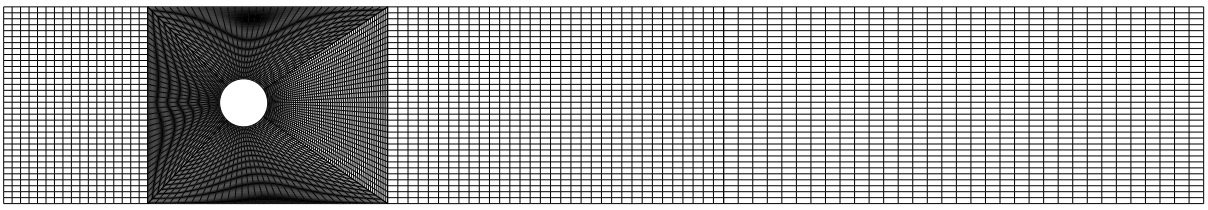

Abbildung 5.6: Senkrechter Schnitt durch das Gitter mit 373527 Zellen

Die Ergebnisse der ersten Rechnung sind in Tabelle 5.3 dargestellt. Für die verschiedenen Auflösungen wurden c_D, c_L und Δp berechnet[1]. Weiterhin sind in den letzten beiden Zeilen Referenzwerte aus [Tur99] eingetragen. Diese sind in Form eines [min,max]-Intervalls gegeben. Für die Berechnung dieser Werte wurden Diskretisierungen mit mehreren Millionen Zellen benutzt.

[1] Die dazu benötigten Größen wurden in den entsprechenden Punkten mit einem Finiten-Differenzen-Verfahren zweiter Ordnung aus Werten umgebender Punkte gewonnen.

Anzahl Zellen	c_D	c_L	Δp
5103	5.9746	0.0379	0.1521
34391	6.3949	0.0052	0.1663
251559	6.3629	0.0036	0.1702
373527	6.2952	0.0079	0.1704
Referenzwerte aus [Tur99]			
min.	6.0500	0.0080	0.1650
max.	6.2500	0.0100	0.1750

Tabelle 5.3: Ergebnisse der ersten Rechnung mit Vergleichswerten aus [Tur99] in Form eines $[\min, \max]$-Intervalls.

Zunächst läßt sich feststellen, daß die berechneten Werte nicht alle innerhalb des Referenzintervalls liegen, wohl aber die richtige Größenordnung haben. Die geringste Auflösung von 5103 Zellen ist nicht geeignet, die verlangten Größen richtig zu berechnen und wird daher in der weiteren Diskussion vernachlässigt. Für alle weiteren Auflösungen liegt Δp im Referenzintervall, für die beiden feinsten Auflösungen sogar recht genau in der Mitte des Intervalls. Der c_D-Wert wird in allen Auflösungen überschätzt, er liegt aber nur maximal 4% über der Intervallmitte, in der höchsten Auflösung halbiert sich dieser Fehler noch. Ein ähnliches Verhalten läßt sich beim c_L-Wert feststellen, der allerdings unterschätzt wird, aber in der feinsten Auflösung die untere Intervallgrenze fast erreicht[2].

Anzahl Zellen	c_D	c_L	St
251559	3.441	-0.019	0.296
Vergleichswerte aus [Tur99]			
min.	3.290	-0.011	0.290
max.	3.310	-0.008	0.350

Tabelle 5.4: Ergebnisse der zweiten Rechnung mit Vergleichswerten aus [Tur99] in Form eines $[\min, \max]$-Intervalls.

Für die zweite Rechnung sind in [Tur99] nur Vergleichswerte gegeben, da die verlangten Größen nicht mit genügender Genauigkeit berechnet werden konnten und keine Versuchsergebnisse vorlagen. Auch sind nicht für alle verlangten Größen Werte angegeben. Die vorhandenen – wieder in Form eines Intervalls gegebenen – Größen sind zusammen mit den berechneten Werten in Tabelle 5.4 eingetragen. Auf eine Angabe der Werte, für die kein Vergleichswert vorlag, wurde verzichtet.

[2]Dazu ist zu erwähnen, daß dieser Wert bei etwa der Hälfte der in [Tur99] wiedergegebenen Berechnungsergebnisse anderer Verfahren und Programme nicht korrekt berechnet wurde. Bei zwei Ergebnissen konnte sogar das Vorzeichen nicht richtig bestimmt werden.

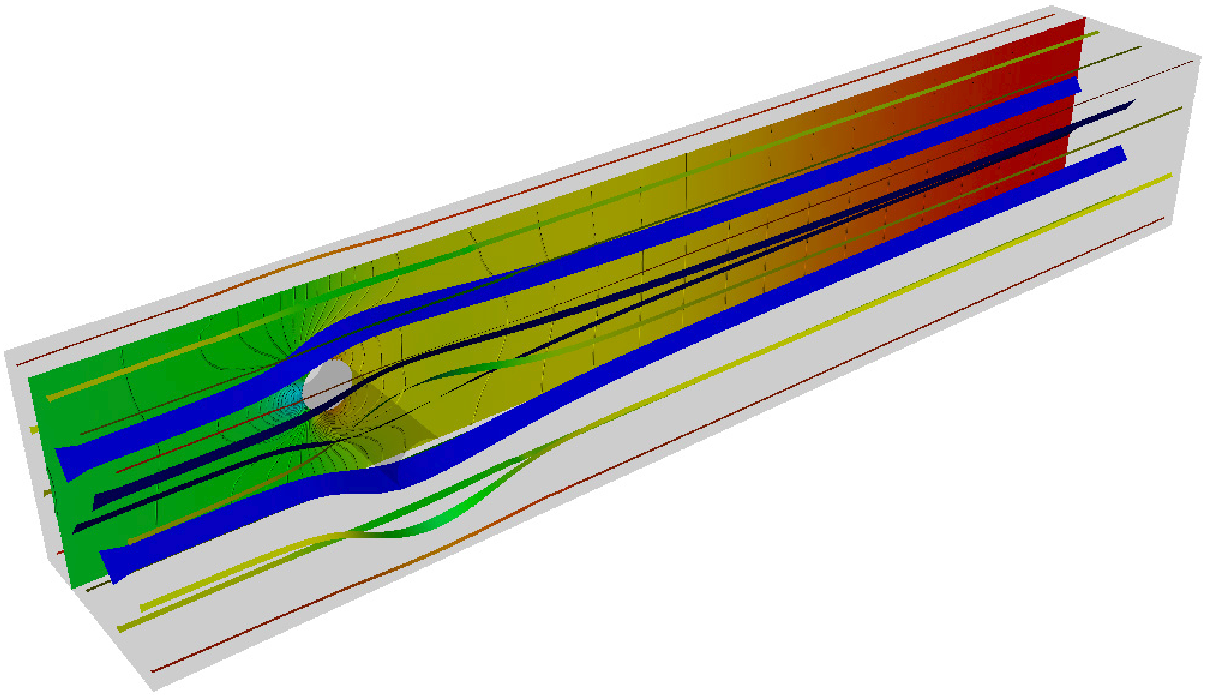

Abbildung 5.7: Visualisierung des Ergebnisses der ersten Rechnung in der dritten Auflösungsstufe ($7 \cdot 33^3$ Zellen). Neben Stromlinien, die Richtung und Geschwindigkeit der Strömung durch Farbe und Dicke anzeigen, ist auf der senkrechten Schnittfläche der Druck mit 50 äquidistanten Niveaulinien dargestellt.

Auch hier kann beobachtet werden, daß die c_D- und c_L-Werte leicht überschätzt werden. Dagegen konnte die Strouhal-Zahl St recht genau bestimmt werden.

Insgesamt läßt sich feststellen, daß mit dem verwendeten Verfahren die qualitativen Aspekte der berechneten Strömungen erfaßt wurden, wogegen die genaue quantitative Bestimmung im instationären Fall schwierig ist. Dieses Problem haben jedoch auch viele andere Verfahren (vgl. [Tur99]). Für die Anwendung in einem Optimierungsverfahren ist aber die absolute Größe solcher Werte nicht wichtig, sondern nur ihr Verhältnis zueinander für verschiedene Geometrien.

5.2 Optimierung

5.2.1 Parameteridentifikation

Dieses Beispiel einer Optimierungsrechung dient dazu, das Optimierungsverfahren zu testen und Konvergenzeigenschaften herauszufinden. Dazu ist es hilfreich, ein Problem zu konstruieren, dessen optimale Lösung bekannt ist. Hier bietet sich an, die Form eines Gebiets, welches durchströmt wird, mit einigen Formparametern zu versehen, diese einmal fest zu wählen und das resultierende Referenzströmungsfeld zu berechnen. Anschließend wird versucht, diese Formparameter durch einen Optimierungsprozeß wiederzufinden, bei dem der Unterschied des resultierenden Strömungsfelds zum Referenzströmungsfeld minimiert wird. Das Minimum ist dann erreicht, wenn das Strömungsfeld identisch zum Referenzströmungsfeld ist. Dies sollte dann der Fall sein, wenn die Formparameter identisch sind.

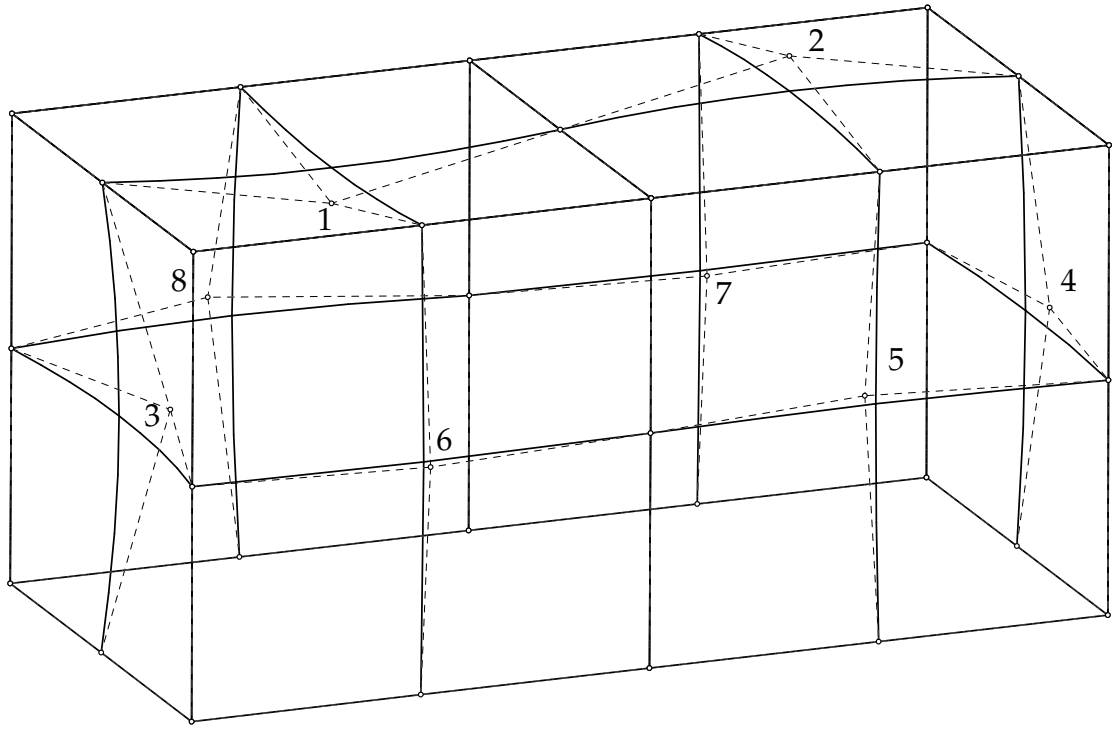

Abbildung 5.8: Geometrie für die Parameteridentifikation. Die Ablenkung der acht Kontrollpunkte in den Flächenmittelpunkten sind freie Parameter.

Die Geometrie zu dieser Rechnung ist in Abbildung 5.8 dargestellt. An der Unterseite ist eine Dirichlet-Randbedingung gesetzt, die – ähnlich wie in der Driven Cavity (siehe Abschnitt 5.1.1) – das Fluid mit Geschwindigkeit 1 nach hinten antreibt. An den übrigen Randflächen sind Haftbedingungen gesetzt. Diese acht Seitenflächen(-teile) sind, wie in Abbildung 5.8 zu sehen, mit NURBS-Flächen zweiter Ordnung parametri-

Parameter	Referenzwert	Startwert	Constraints
1	0.9	1	$0.7 \leq x_1 \leq 1.3$
2	1.1	1	$0.7 \leq x_2 \leq 1.3$
3	1.85	2	$1.7 \leq x_3 \leq 2.3$
4	-0.07	0	$-0.3 \leq x_4 \leq 0.3$
5	0.08	0	$-0.3 \leq x_5 \leq 0.3$
6	-0.05	0	$-0.3 \leq x_6 \leq 0.3$
7	0.95	1	$0.7 \leq x_7 \leq 1.3$
8	1.18	1	$0.7 \leq x_8 \leq 1.3$

Tabelle 5.5: Referenzformparameter und Startwerte für die Optimierung

siert. Als Formparameter sind die Ablenkungen der mittleren Kontrollpunkte entlang der Flächennormalen (bzw. der entsprechenden Koordinateneinträge) vorgesehen. Die Reynoldszahl beträgt 500 und es wird vom Startzustand $\boldsymbol{u} = 0$, $p = 0$ zwei Sekunden lang gerechnet. Für das Referenzströmungsfeld sind Formparameter wie in Tabelle 5.5 gewählt. Abbildung 5.8 zeigt die Referenzgeometrie, Abbildung 5.9 die resultierende Strömung. Als Startwerte für die Optimierung sind die Formparameter so gewählt, daß die resultierende Geometrie ein Quader ist. Zielfunktion ist der Abstand der diskreten Geschwindigkeitsfelder gegeben durch

$$h(V) = \sqrt{\sum_{i=1}^{3} \sum_{V^i \text{ an innerem Punkt}} \left(V^i_{2s} - V^i_{\text{ref},2s}\right)^2},$$

wobei V^i_{2s}, $V^i_{\text{ref},2s}$ die Geschwindigkeitsfelder nach zwei simulierten Sekunden sind. Weiterhin sind in Tabelle 5.5 die Constraints für jeden Parameter aufgelistet.

In der ersten Rechung wird das Gebiet mit $2 \cdot 23^3$ Zellen aufgelöst und dann die Diskretisierungsschrittweite für die Gradientenberechnung `epsfcn` im Optimierer variiert, um günstige Werte für diese Schrittweite herauszufinden. Die Auswahl des Parameters `epsfcn` wurde auf vernünftige Werte beschränkt, d.h. Rechnungen, bei deren Verlauf schon schlechte Konvergenz erkennbar war, wurden abgebrochen. Die Ergebnisse für diese Rechnung zeigt Tabelle 5.6. Zunächst sind die absoluten Fehler zu den Referenzparametern und deren Mittelwert aufgelistet. Der jeweils beste Wert ist markiert. Für den Wert $\texttt{epsfcn} = 1.5 \cdot 10^{-6}$ wird der Referenzwert am häufigsten gut approximiert, der Mittelwert bestätigt dies. Für diesen Wert von `epsfcn` zeigt sich auch eine geringe Standardabweichung, d.h. die Referenzwerte werden durchgängig gut getroffen. Weiterhin fällt auf, daß die Parameter 3 und 4 relativ schlecht getroffen werden. Dies könnte daran liegen, daß sich diese Parameter entlang der Rotationsachse des entstehenden Wirbels bewegen und daher relativ wenig Einfluß auf das Geschwindigkeitsfeld haben. Die Parameter, die am besten getroffen werden, befinden sich genau gegenüber des "Antriebs" der Strömung.

Weiterhin finden sich in der Tabelle die Werte der Zielfunktion im gefundenen Minimum. Diese korrelieren wie erwartet mit dem Mittelwert der absoluten Parameterfeh-

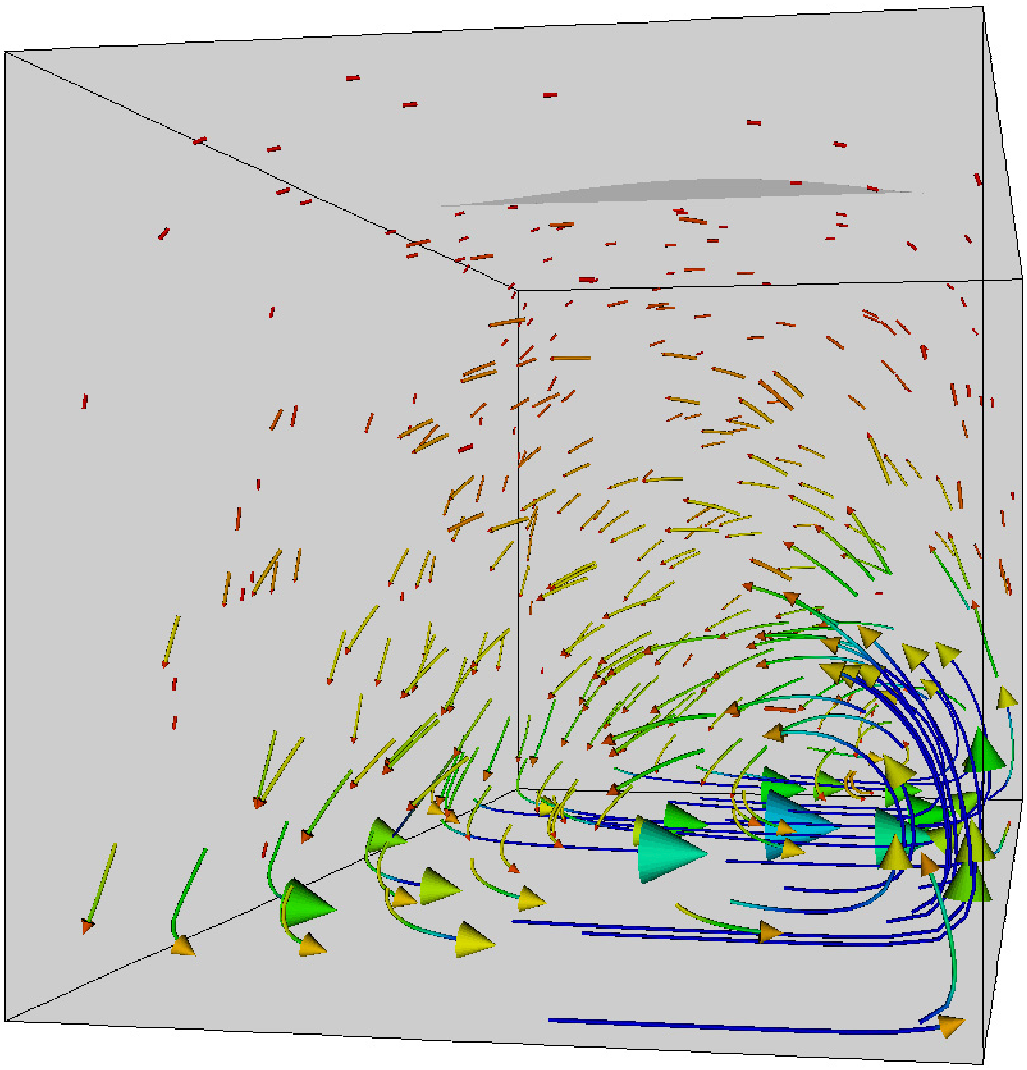

Abbildung 5.9: Visualisierung der Strömung zur Parameteridentifikation

ler. Schließlich sind die Anzahl der Funktionsauswertungen und die Anzahl der Gradientenauswertungen aufgelistet. Auch diese korrelieren leider positiv mit der Qualität der Lösung, d.h. für eine genaue Lösung müssen auch viele Auswertungen erfolgen.

In Abbildung 5.10 ist das Konvergenzverhalten für die verschiedenen Werte von `epsfcn` dargestellt. Die verschiedenen Kurven unterscheiden sich nicht wesentlich voneinander. Es läßt sich superlineare Konvergenz erkennen. In den ersten zehn Iterationsschritten vermindert sich der Wert der Zielfunktion monoton und relativ stark (Faktor ≈ 0.73, d.h. der Wert der Zielfunktion reduziert sich in jedem Schritt etwa um den Faktor 0.73), danach ist das Verhalten ungleichmäßiger und die Konvergenz langsamer (im Mittel Faktor 0.92), bis sich etwa ab dem 70. Iterationsschritt der Fehler kaum mehr verringert. Dieses Verhalten läßt sich durch die ungenaue (weil diskrete) Auswertung des

epsfcn·10^6	3	1.5	0.75	0.375
	Absoluter Fehler·10^4			
Parameter 1	2.61196315	**0.04902806**	0.28289107	2.48627108
Parameter 2	2.88548944	0.36261855	**0.03887260**	2.31403446
Parameter 3	5.56447439	1.54700773	1.44994599	**0.62236969**
Parameter 4	2.31962417	**0.88759937**	1.08117852	2.79815921
Parameter 5	2.13330073	**0.13560769**	0.46175775	1.62565721
Parameter 6	1.41182764	**0.39649840**	1.10552179	2.53483782
Parameter 7	1.25348171	**0.19157616**	0.75335615	1.17574683
Parameter 8	1.45305123	**0.11555147**	0.23805246	2.28875148
Mittelwert	2.45415156	**0.46068593**	0.67644704	1.98072847
Standardabweichung	1,39059889	0,51308080	**0,50025667**	0,76120057
$h_{\min} \cdot 10^6$	2.785057	0.9080612	1.340331	2.335925
$\sharp\,\mathrm{Eval}(h)$	134	272	260	183
$\sharp\,\mathrm{Eval}(\nabla h)$	42	86	67	53

Tabelle 5.6: Ergebnisse der ersten Rechnung – Variation von epsfcn: Absolute Fehler der Formparameter zu den Referenzparametern, Wert der Zielfunktion im gefundenen Minimum, Anzahl der Funktionsauswertungen und Gradientenberechnungen

Gradienten erklären. Ist der Wert der Zielfunktion noch relativ hoch, genügt eine grobe Richtungsangabe durch den Gradienten, um ihren Wert stark zu vermindern. Je näher man jedoch dem Optimum kommt, desto stärker weisen "falsche" Gradienten von dem Optimum weg.

Aufgrund der bisherigen Ergebnisse wird für die weiteren Rechnungen der Wert epsfcn $= 1.5 \cdot 10^{-6}$ verwendet.

Als nächstes wird ein möglicher Zusammenhang zwischen der Diskretisierungsauflösung und dem Optimierungsprozeß untersucht. Dazu wird die schon beschriebene Rechnung für verschiedene Auflösungen, nämlich für $2 \cdot 17^3$, $2 \cdot 20^3$, $2 \cdot 23^3$ und $2 \cdot 33^3$ Zellen durchgeführt. Die Ergebnisse dieser Rechnungen sind in Tabelle 5.7 dargestellt. Es ist kein offensichtlicher Zusammenhang zwischen Diskretisierungsauflösung und Optimierungsprozeß erkennbar. Die Genauigkeit, mit der die einzelnen Parameter gefunden werden, variiert sehr stark und ohne erkennbares Muster über die verschiedenen Auflösungen. Auch die Konvergenzraten sind unterschiedlich, mehr Gradientenauswertungen, also Iterationen des Optimierungsverfahrens, bedeuten nicht, daß die Parameter auch besser gefunden werden.

Zur Reduzierung der Anzahl der Optimierungsparameter soll schließlich untersucht werden, wie gut sich eine Fläche höherer Ordnung durch Flächen geringerer Ordnung approximieren läßt. In Abbildung 5.11 ist die für diese Rechnung benutzte Geometrie dargestellt. Alle Randbedingungen und weitere Parameter sind mit denen der ersten

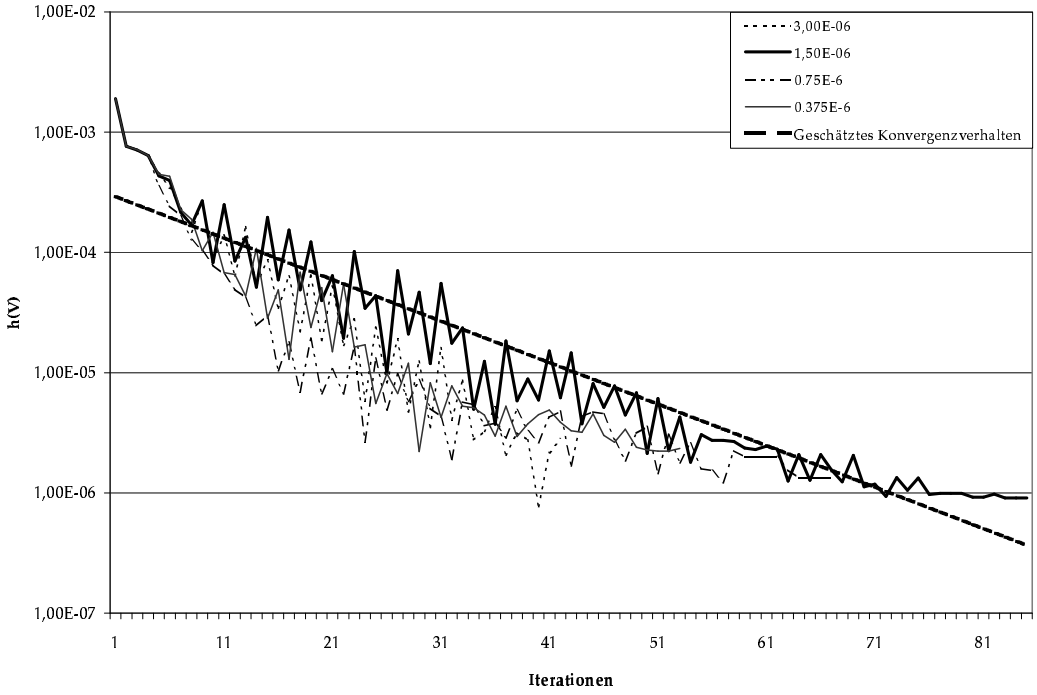

Abbildung 5.10: Konvergenzverhalten für verschiedene Werte von `epsfcn` mit logarithmischer Skala für $h(V)$ und geschätztem Verhalten für `epsfcn` $= 1.6 \cdot 10^{-6}$

beiden Rechnungen identisch. Das Gebiet wurde mit $2 \cdot 24^3$ Zellen diskretisiert. Die beiden oberen Flächen sind durch NURBS zweiter Ordnung parametrisiert. Sie sollen nun mit NURBS erster Ordnung approximiert werden. Es wurden Rechnungen mit zwei und mit 18 freien Parametern durchgeführt.

Die Ergebnisse beider Rechnungen sind in Tabelle 5.8 und graphisch in Abbildung 5.12 dargestellt. Es wurden 40 bzw. 43 Optimierungsschritte benötigt, die Startwerte für die erste Rechnung ergaben sich aus der normalen Kastenform und die der zweiten Rechnung aus dem Ergebnis der ersten Rechnung. In Abbildung 5.12 läßt sich erkennen, daß die vorgegebe Form recht gut approximiert wird. Dies wird bestätigt durch die numerischen Werte in Tabelle 5.8: Für die erste Rechnung beträgt der maximale absolute Fehler etwa 0.06, für die zweite Rechnung mit der höheren Randauflösung etwa 0.02.

♯ Zellen	$2 \cdot 17^3$	$2 \cdot 20^3$	$2 \cdot 23^3$	$2 \cdot 33^3$
	Absoluter Fehler$\cdot 10^4$			
Parameter 1	0,05571983	1,00810754	**0,04902806**	0,56023333
Parameter 2	0,54593830	0,60316867	**0,36261855**	2,98089228
Parameter 3	**0,12347082**	1,51574618	1,54700773	3,37916038
Parameter 4	6,01087976	**0,33249622**	0,88759937	3,05571765
Parameter 5	0,48459953	**0,04720828**	0,13560769	3,49541495
Parameter 6	1,16048650	0,86946092	0,39649840	**0,16975804**
Parameter 7	**0,17245216**	1,39563030	0,19157616	1,76993747
Parameter 8	1,12963358	0,51176369	**0,11555147**	1,99451489
Mittelwert	1,21039756	0,78544773	**0,46068593**	2,17570362
Standardabweichung	1,98616476	**0,50998394**	0,51308080	1,27860064
♯ Eval(h)	530	202	272	171
♯ Eval(∇h)	124	64	86	63

Tabelle 5.7: Ergebnisse der zweiten Rechnung – Variation der Diskretisierungsauflösung: Absolute Fehler der Formparameter zu den Referenzparametern, Anzahl der Funktionsauswertungen und Gradientenberechnungen

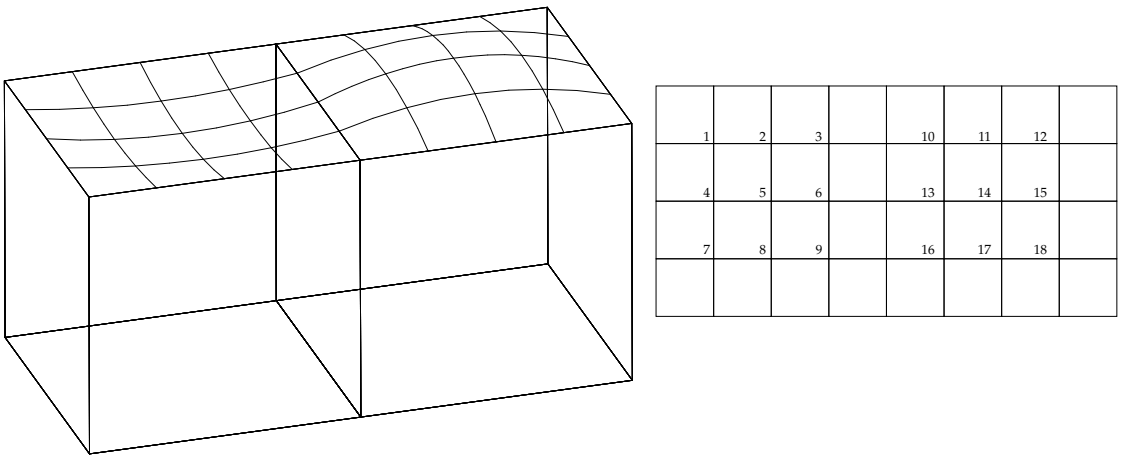

Abbildung 5.11: Geometrie für die dritte Rechnung. Die Flächen sind durch NURBS zweiter Ordnung parametrisiert. Der Antrieb der Strömung erfolgt wieder an der Unterseite in Richtung der hinteren Wand. Auf der rechten Seite sind die Positionen der 18 Parameter dargestellt (Sicht von oben auf die dargestellte Geometrie).

		2 Parameter		18 Parameter	
Parameter	Referenzwert	Wert	Fehler	Wert	Fehler
1	0,971875			0,97209697	0,00022197
2	0,9625			0,94794073	0,01455927
3	0,971875			0,97148418	0,00039082
4	0,9625			0,96355132	0,00105132
5	0,95	0,91422804	0,03577196	0,93297376	0,01702624
6	0,9625			0,96219955	0,00030045
7	0,971875			0,97224938	0,00037438
8	0,9625			0,95732644	0,00517356
9	0,971875			0,97220472	0,00032972
10	1,05625			1,06755691	0,01130691
11	1,075			1,08647461	0,01147461
12	1,05625			1,06646854	0,01021854
13	1,075			1,08608399	0,01108399
14	1,1	1,16178557	0,06178557	1,11026607	0,01026607
15	1,075			1,08537507	0,01037507
16	1,05625			1,06648438	0,01023438
17	1,075			1,08350413	0,00850413
18	1,05625			1,06578196	0,00953196

Tabelle 5.8: Ergebnisse der dritten Rechnung für zwei und 18 Parameter. Dargestellt sind für alle freien Parameter die entsprechenden Werte und die absoluten Fehler zur Referenzfläche (siehe auch Abbildung 5.12)

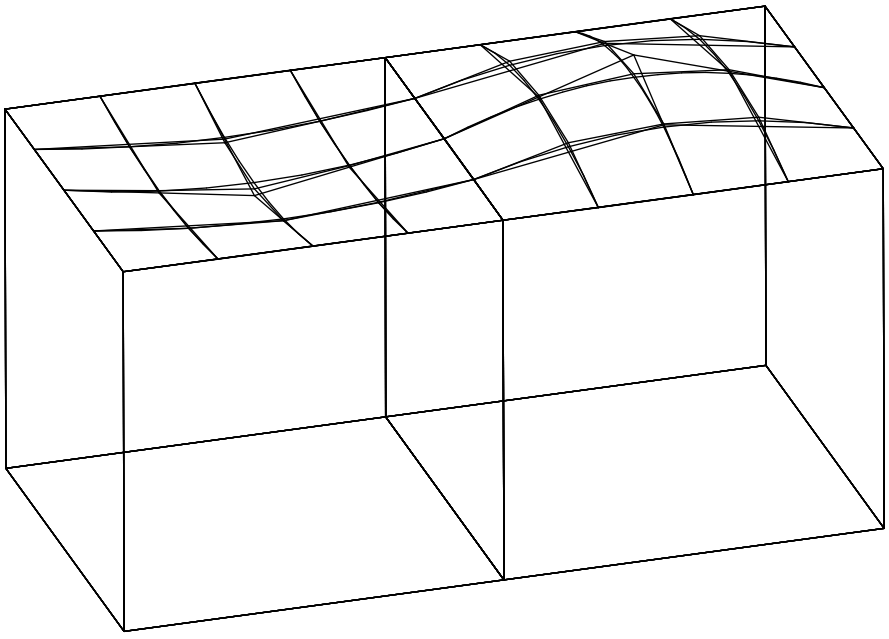

Abbildung 5.12: Ergebnisse der dritten Rechnung für zwei und 18 Parameter

5.2.2 Diffusor

Ein Diffusor ist ein strömungstechnisches Bauteil zur möglichst verlustarmen Umwandlung von kinetischer Energie in Druckenergie. Diffusoren werden technisch angewendet zur Durchsatzerhöhung in Rohrleitungen, zur Verminderung der Strömungsverluste in nachgeschalteten Leitungssystemen und zur Verkleinerung der Austrittsverluste in Maschinen (siehe [Len89]). Da die Strömung im Diffusor verzögert wird, muß seine Gestaltung sehr sorgfältig erfolgen, um Grenzschichtablösung zu vermeiden und einen hohen Wirkungsgrad zu erzielen.

Abbildung 5.13 zeigt die Geometrie des hier betrachteten Diffusors. An der Einlaßöffnung links ist ein parabolisches Geschwindigkeitsprofil mit Maximalgeschwindigkeit 1 gesetzt, an der Auslaßöffnung rechts entsprechend Ausströmrandbedingungen. Das zweidimensionale Gebiet wird durch zwei Zellschichten und Slip-Randbedingungen an den entsprechenden Seitenflächen modelliert und mit acht Blöcken zu je $17^2 \cdot 2$ Zellen aufgelöst. Die Reynoldszahl beträgt 100. Die Rechnungen wurden jeweils bis in einen stationären Zustand iteriert. Dieser ist erreicht, wenn die Änderung der folgenden Zielfunktion über 200 Zeitschritte in jedem Zeitschritt klein genug ist, genauer: $|h(p_n) - h(p_{n+1})| < 2 \cdot 10^{-6}$, $n_0 \leq n \leq n_0 + 200$.

Die Zielfunktion, die den Druckverlust quantifiziert, ist durch das arithmetische Mittel $h(p_n) = \frac{1}{30} \sum_{i=0}^{29} g(p_{n-i})$ der letzten 30 Druckdifferenzen gegeben, wobei

$$g(p) = \int_{\text{Einströmfläche}} p \, d\Gamma - \int_{\text{Ausströmfläche}} p \, d\Gamma.$$

Für die Berechnung des Gradienten der Zielfunktion wurde ein Finite-Differenzen-Schema zweiter Ordnung verwendet. Um Ungenauigkeiten bei der Gradientenberechnung zu verringern, wurde für diese immer mit derselben Startströmung begonnen.

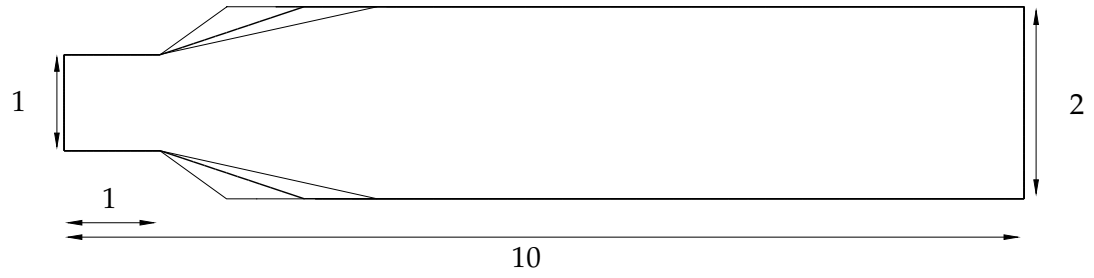

Abbildung 5.13: Geometrie des Diffusors für die erste Rechnung mit der Einströmfläche auf der linken und der Ausströmfläche auf der rechten Seite. Fett dargestellt ist die Startgeometrie, weiterhin sind die beiden Extremkonfigurationen dargestellt.

Zunächst soll eine einfache Optimierungsrechnung durchgeführt werden, bei der nur der Winkel der Aufweitung des Diffusors variiert wird. Dies wird durch Änderung

der Länge der Aufweitung erreicht, wobei die Länge der gesamten Konfiguration erhalten bleibt (Abbildung 5.13). Das Ergebnis dieser Rechnung ist in Tabelle 5.9 eingetragen und in Abbildung 5.15 visualisiert. Vom Startwert 0.5, was einem Aufweitungswinkel von 30 Grad entspricht, wurde ein optimaler Wert von 0.75, entsprechend einem Winkel von etwa 16 Grad, gefunden. Dabei konnte die Druckdifferenz um etwa 7 % gesteigert werden.

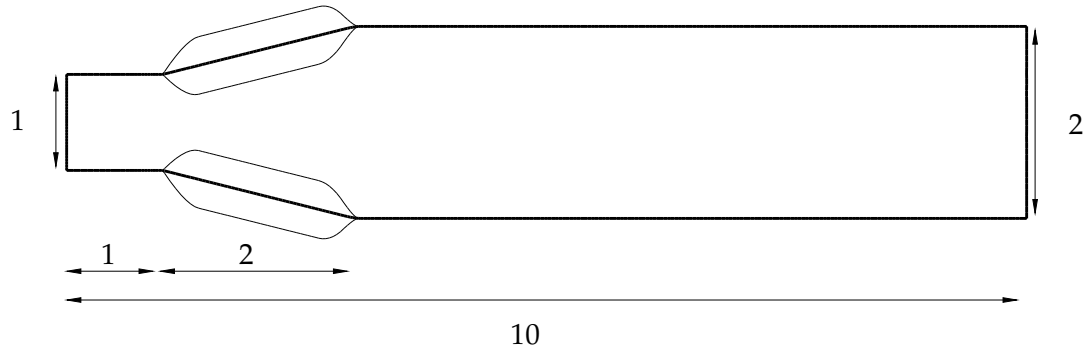

Abbildung 5.14: Geometrie des Diffusors für die zweite Rechnung mit der Einströmfläche auf der linken und der Ausströmfläche auf der rechten Seite. Fett dargestellt ist die Startgeometrie, weiterhin sind die beiden Extremkonfigurationen dargestellt. Der Rand im variablen Bereich ist mit einer NURBS-Fläche zweiter Ordnung und 7 freien Parametern parametrisiert. Das gesamte Gebiet bleibt immer symmetrisch.

Für die zweite Rechnung wurde ein Teil des Randes – wie in Abbildung 5.14 gezeigt – mit sieben freien Optimierungsparametern versehen, die die Abweichung von der geraden diagonalen Verbindung angeben. Das Ergebnis dieser Rechnung ist in Abbildung 5.16 dargestellt. Die Werte der Formparameter finden sich in Tabelle 5.9. Durch die Ausbeulung am Anfang der Aufweitung wird scheinbar ein Ablösen der Strömung verhindert bzw. auf diese Beule begrenzt. Dadurch kann der Druckverlust um etwa 30 % verringert werden. Es ist aber zu vermuten, daß dieser Effekt nur bei relativ niedriger Reynoldszahl auftritt.

Parameter	Rechnung 1		Rechnung 2	
x_1	0.5	0.75	0	0.23253
x_2			0	0.11321
x_3			0	0.10182
x_4			0	0.06857
x_5			0	0.04261
x_6			0	0.01911
x_7			0	-0.0003
Constraints	$-0.75 \leq x_i \leq 0.75$		$-0.3 \leq x_i \leq 0.3$	
h_0	-0.0208313		-0.02201	
$h_{\min}$	-0.02234		-0.03277	
$\sharp \text{Eval}(h)$	14		32	
$\sharp \text{Eval}(\nabla h)$	4		5	

Tabelle 5.9: Ergebnisse der Rechnungen, Parameter mit Start- und Optimalwert, Constraints, Zielfunktionswerten und Anzahl der Funktions- und Gradientenauswertungen

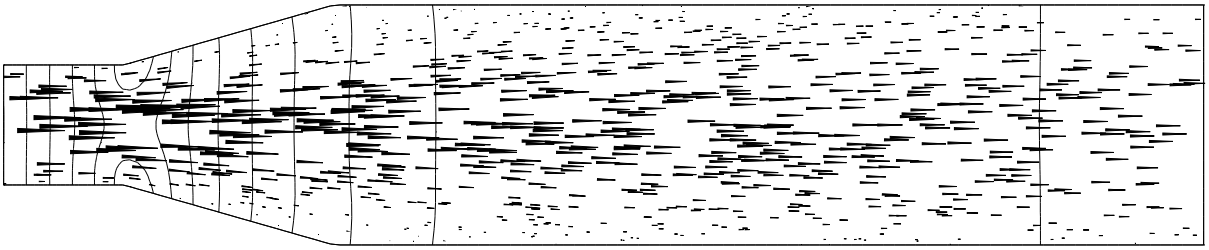

Abbildung 5.15: Ergebnis der ersten Rechnung, visualisiert mit Geschwindigkeitspfeilen und 10 äquidistanten Druckisolinien

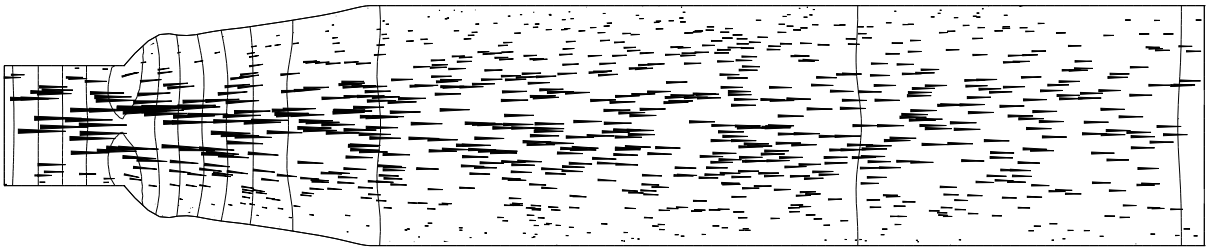

Abbildung 5.16: Ergebnis der zweiten Rechnung, visualisiert mit Geschwindigkeitspfeilen und 10 äquidistanten Druckisolinien

5.2.3 Krümmer

Als dreidimensionales Optimierungsbeispiel soll ein optimaler 180°-Krümmer berechnet werden. Die Geometrie ist in Abbildung 5.17 dargestellt. Das Rohr hat einen Radius von einer Einheit, der Krümmungsradius beträgt vier Einheiten. Auf der einen Seite ist ein parabolisches Einströmprofil mit maximaler Geschwindigkeit 1 gesetzt, auf der anderen Seite die Ausström-Randbedingung. Die Rohroberfläche ist mit der Haftrandbedingung versehen. Die Reynoldszahl beträgt 100. Das Gebiet wurde in acht Blöcke zu je $9 \cdot 9 \cdot 17$ Zellen aufgeteilt.

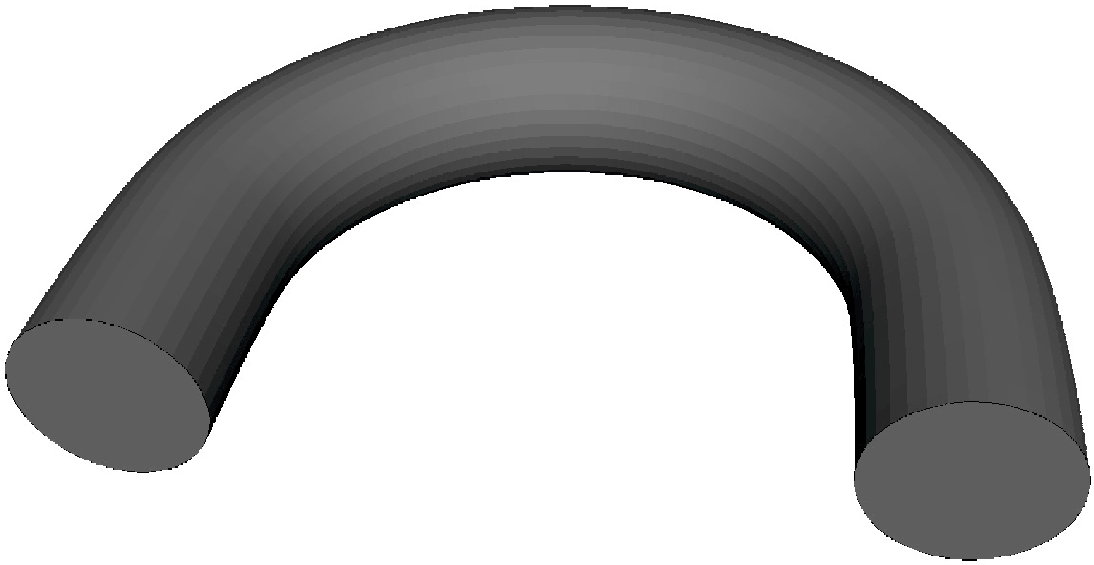

Abbildung 5.17: Grundgeometrie des Krümmers

Das Rohr soll nun so geformt werden, daß der Druckverlust minimal wird. Dies wird durch die Zielfunktion

$$h(p) = \int_{\text{Einströmfläche}} p \, d\Gamma - \int_{\text{Ausströmfläche}} p \, d\Gamma$$

formalisiert, wobei der Druck p einer stationären Strömung eingesetzt wird. Eine Strömung wird hier als stationär betrachtet, wenn die Geschwindigkeitsänderung 5 Iterationen lang klein genug ist, genauer: $\|V_n - V_{n+1}\|_2 < 10^{-6}$, $n_0 \leq n \leq n_0 + 5$.

Die Verformung des Rohrs wird durch neun freie Parameter ermöglicht, mit denen Querschnittkreise durch das Rohr in der Krümmungsebene verschoben werden können. Diese Verschiebung ist auf eine Veränderung des Radius des Krümmungskreises beschränkt (Abbildung 5.18). Es wurden Rechnungen mit unterschiedlichen Constraints, d.h. möglichen Geometrien durchgeführt. Diese sind in Abbildungen 5.18, 5.19 dargestellt, Details können Tabelle 5.10 entnommen werden. Als Startwert wurde eine halbkreisförmige Krümmung gewählt ($x_i = 0$, $1 \leq i \leq 9$). Die Approximation des Gradienten der Zielfunktion erfolgte mit einer Finite-Differenzen-Approximation sechster Ordnung.

Die Ergebnisse der Rechnungen sind ebenfalls in den Abbildungen 5.18, 5.19 und in Tabelle 5.10 eingetragen. Abbildung 5.20 zeigt die berechnete Strömung für die zweite Rechnung. Der initiale Zielfunktionswert $h_0 = 5.799$ konnte auf etwa 5.389 bzw. 5.362 verbessert werden, was einer Effizienzsteigerung von etwa 7% gleichkommt. Die Ergebnisse legen nahe, daß dies hauptsächlich durch eine Verkürzung der Länge des Krümmers erreicht wird, denn der Verlauf der optimalen Geometrie liegt nahe an der kürzesten durch die Constraints möglichen.

Weiterhin sollte in diesem Abschnitt überprüft werden, wieviel die Benutzung schon berechneter Strömungen als Startwert für neue Strömungsberechnungen zur Effizienz des Optimierungsvorgangs beiträgt. Für den Start einer neuen Strömungsberechnung wurde immer das letzte Ergebnis des ersten Prozesses verwendet. In Abbildung 5.21 sind die (physikalischen) Simulationszeiten bis in einen stationären Zustand dargestellt, wobei die Rechnungen für die Finite-Differenzen-Approximation des Gradienten der Zielfunktion unterschieden wurden von den Rechnungen für die Liniensuche, da für erstere ein größerer Effekt zu erwarten ist.

Es zeigt sich, daß die beschriebene Vorgehensweise einen großen Effekt auf die Effizienz des Verfahrens hat. Für eine Rechnung mit Startwert 0 werden etwa 10 bis 12 simulierte Sekunden benötigt. Beginnt man die Rechnungen mit schon bekannten Ergebnissen, so reduziert sich diese Zeit auf durchschnittlich etwa eine Sekunde. Damit reduziert sich auch die Rechenzeit auf etwa ein Zehntel. Dieser Effekt kann besonders bei den Gradientenberechnungen (durchschnittlich 0.6 Sekunden) beobachtet werden, da bei diesen die Geometrien recht ähnlich sind. Die ersten Liniensuchen eines Iterationsschritts benötigen etwa dieselbe Zeit wie eine mit Null gestartete Rechnung. Dies liegt daran, daß die Geometrien weit von einer die Zielfunktion vermindernden weg liegen und die resultierenden Strömungen recht unterschiedlich sind.

Parameter	Rechnung 1		Rechnung 2	
x_1	$-0.5 \leq x_1 \leq 0.5$	0.21125	$-0.01 \leq x_1 \leq 0.01$	0.01
x_2	$-0.5 \leq x_2 \leq 0.5$	0.5	$-0.5 \leq x_2 \leq 0.5$	0.48087
x_3	$-0.5 \leq x_3 \leq 0.5$	0.5	$-0.75 \leq x_3 \leq 0.75$	0.75
x_4	$-0.5 \leq x_4 \leq 0.5$	0.5	$-1 \leq x_4 \leq 1$	1
x_5	$-0.5 \leq x_5 \leq 0.5$	0.5	$-1 \leq x_5 \leq 1$	1
x_6	$-0.5 \leq x_6 \leq 0.5$	0.5	$-1 \leq x_6 \leq 1$	0.91144
x_7	$-0.5 \leq x_7 \leq 0.5$	0.5	$-0.75 \leq x_7 \leq 0.75$	0.72199
x_8	$-0.5 \leq x_8 \leq 0.5$	0.5	$-0.5 \leq x_8 \leq 0.5$	0.5
x_9	$-0.5 \leq x_9 \leq 0.5$	0.17567	$-0.01 \leq x_9 \leq 0.01$	0.01
$h_{\min}$	5.389		5.362	
$\sharp\,\mathrm{Eval}(h)$	62		163	
$\sharp\,\mathrm{Eval}(\nabla h)$	9		18	

Tabelle 5.10: Constraints und Ergebnisse der Optimierungsrechnungen

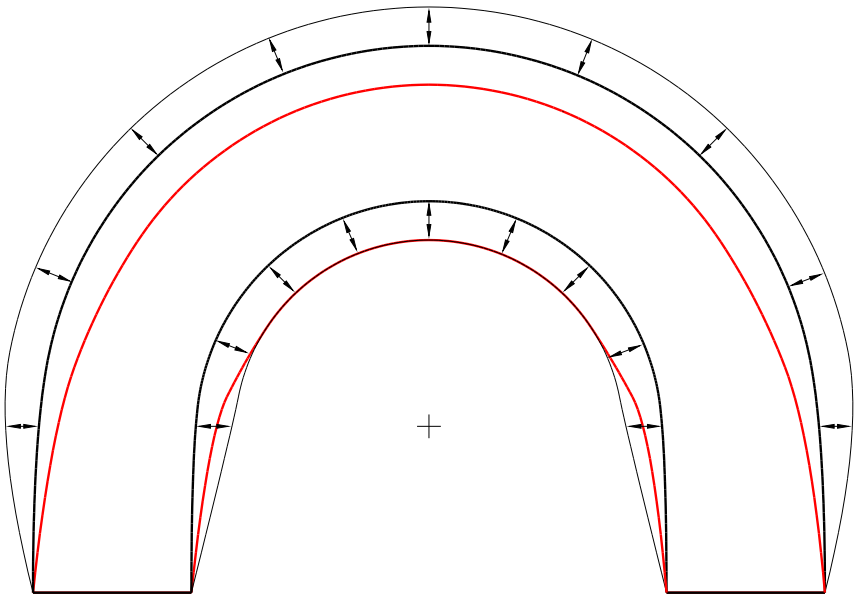

Abbildung 5.18: Erste Rechnung, Schnitt durch das Rohr in der Krümmungsebene - Startkonfiguration (schwarz, fett), maximale Verformung (schwarz) und optimale Konfiguration (rot) für die erste Rechnung. Durch die Pfeile sind die Positionen der freien Parameter angedeutet. Links ist die Einströmfläche, rechts die Ausströmfläche.

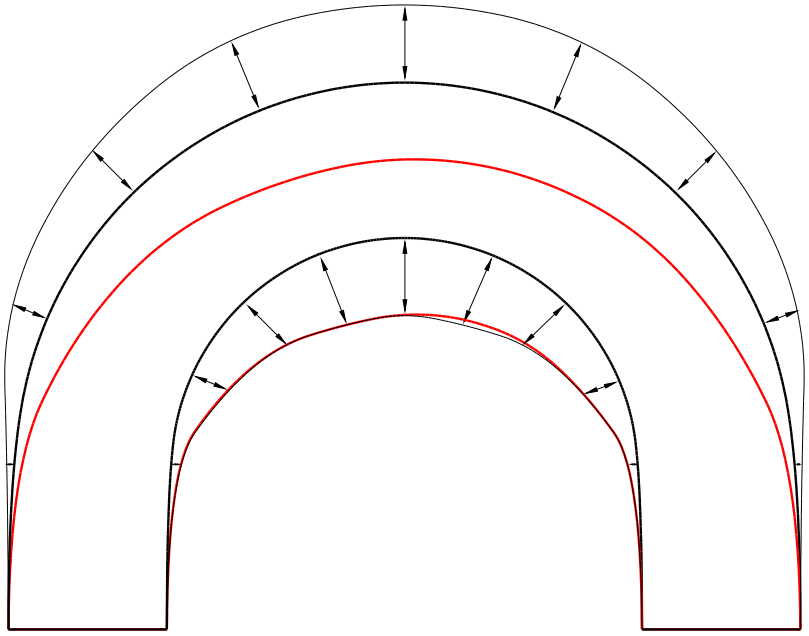

Abbildung 5.19: Zweite Rechnung, Schnitt durch das Rohr in der Krümmungsebene - Startkonfiguration (schwarz, fett), maximale Verformung (schwarz) und optimale Konfiguration (rot) für die erste Rechnung. Die freien Parameter befinden sich an denselben Stellen wie in Abbildung 5.18 angedeutet. Links ist die Einströmfläche, rechts die Ausströmfläche.

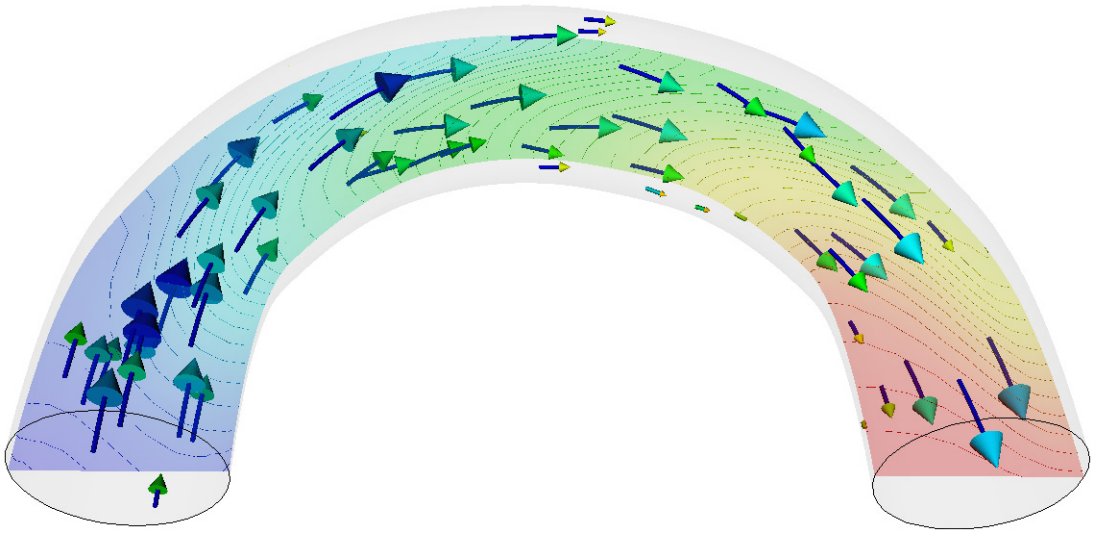

Abbildung 5.20: Visualisierung der Strömung für die zweite Rechnung. In der Schnittfläche ist der Druck durch Farbe und 50 äquidistante Isolinien dargestellt.

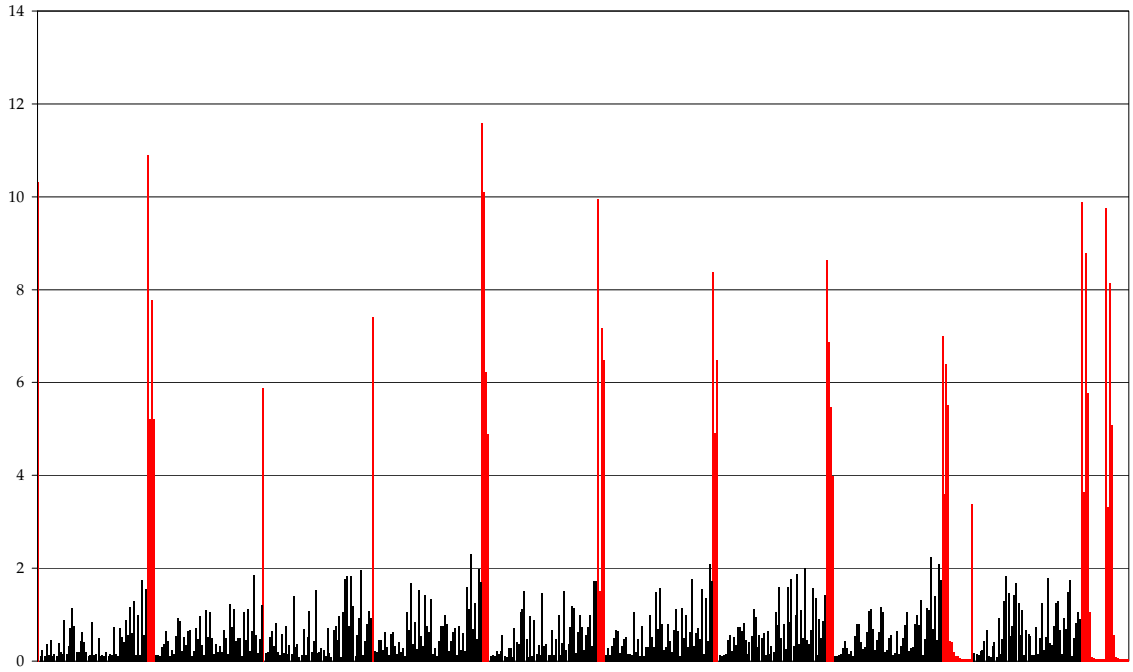

Abbildung 5.21: Simulierte Zeit bis in den stationären Zustand für die Funktionsauswertungen, schwarz für die Gradientenberechnung und rot für die Liniensuche

Kapitel 6

Zusammenfassung und Ausblick

In diesem letzten Kapitel sollen die bisherigen Ergebnisse zusammengefaßt und ein Ausblick auf weitere Entwicklungsmöglichkeiten gegeben werden.

6.1 Zusammenfassung

Zunächst wurde ein Verfahren zur Berechnung von instationären inkompressiblen Strömungen implementiert. Dazu erfolgte die Ortsdiskretisierung der Navier-Stokes-Gleichungen nach dem Prinzip der finiten Volumen und mit einer "*staggered mesh*"-Anordnung der Variablen in dreidimensionalen gekrümmten Koordinaten. Diese Diskretisierung ist im allgemeinen Fall von erster Ordnung und für "einfache" Gitter von zweiter Ordnung. Die Zeitdiskretisierung erfolgte mit dem Druckkorrekturverfahren nach van Kan und ist von zweiter Ordnung in der Zeit. Bei dieser Art der Zeitdiskretisierung sind sehr große lineare Unterprobleme zu lösen. Als geeigneter Löser hat sich das BiCGStab(ℓ)-Verfahren zusammen mit einem Lifting Interpolet Vorkonditionierer herausgestellt.

Es wurde die Methode der blockstrukturierten Gitter verwendet, die die effizienten Datenstrukturen der strukturierten Gitter mit der Flexibilität der unstrukturierten Gitter verbindet und gleichzeitig eine einfache und effiziente Parallelisierung des Verfahrens erlaubt. Zur Beschreibung der Geometrie wurden NURBS-Flächen benutzt, wie sie auch in zahlreichen CAD-Programmen Anwendung finden. Aus dieser Beschreibung wurden dann mit transfiniter Interpolation Gitter generiert, die anschließend mit einer PDE-Methode geglättet und/oder am Rand orthogonal gemacht werden können.

Die Implementierung des Verfahrens erfolgte in der Programmiersprache C++, wobei für die Parallelisierung die weit verbreitete Message-passing-Umgebung MPI verwendet wurde.

Die Verifikation der Implementierung erfolgte mittels zweier Beispiele: der Driven Cavity und des DFG-Benchmarks. Beide Rechnungen lieferten weitgehend korrekte Ergebnisse, wie sie für das implementierte Verfahren zu erwarten waren.

Dieser Strömungslöser wurde nun zur Behandlung von Gebietsoptimierungsproble-

men benutzt. Als Optimierungsverfahren kam eine Variante des SQP-Verfahrens zum Einsatz. Da eine eigene Implementierung eines solchen Verfahrens recht kompliziert ist und hier keine Vorteile bietet, wurde die Implementierung `donlp2` von P. Spellucci angepaßt.

Die für dieses Verfahren benötigten Gradienten der Zielfunktion wurden durch finite Differenzen approximiert. Dies liefert eine hohe Parallelität des Verfahrens, wobei gleichzeitig die "Ähnlichkeit" der zu lösenden Strömungsprobleme die Rechenzeit stark verringern konnte.

Das Optimierungsverfahren wurde zunächst durch Parameteridentifikations-Beispiele getestet, d.h. eine vorgegebenen Geometrie sollte wiedergefunden werden, indem die resultierende Strömung der Strömung in der Referenzgeometrie angenähert wird. Dieser Test wurde in allen Beispielen bestanden. Als praktische Beispiele wurden ein Diffusor und ein Krümmer optimiert.

6.2 Ausblick

Zunächst soll auf Verbesserungsmöglichkeiten bei der Strömungsberechnung eingegangen werden. Da, wie schon in der Einleitung erwähnt, ein relativ einfaches Verfahren zum Einsatz kommt, ist die Berechnung von bestimmten Strömungen nicht sehr effizient, dafür kann mit diesem Verfahren eine große Klasse von Strömungen behandelt werden.

Die mangelnde Effizienz betrifft zum Beispiel stationäre Strömungen, die durch Iteration in einen stationären Zustand berechnet werden. Für solche Strömungen gibt es bessere Verfahren, siehe z.B. [PT83, Tur99]. Diese würden auch die Verwendung eines anderen Optimierungsverfahrens erlauben (s.u.).

Das verwendete Verfahren könnte durch eine implizite Behandlung des diffusiven Terms verbessert werden, da man auf diese Weise die für kleine Reynoldszahlen recht restriktive Stabilitätsbedingung (2.36) los wird.

Ein weiterer Ansatzpunkt wäre die Implementierung eines Turbulenzmodells zur Behandlung turbulenter Strömungen, wie sie in vielen praktischen Anwendungen zu finden sind. Diese Modelle sind meist nicht sehr genau, für Optimierungszwecke sollte die Genauigkeit jedoch ausreichen. Die dafür benötigte Diskretisierung der Transportgleichung wurde in dieser Arbeit schon durchgeführt.

In dieser Arbeit wurde der Optimierungsprozeß getrennt vom Strömungslösungsprozeß betrachtet; dem Optimierer wurde eine Zielfunktion als "black box" übergeben. Es ist anzunehmen, daß durch Aufhebung dieser Trennung der Optimierungsprozeß erheblich effizienter wird. Dies kann dadurch realisiert werden, daß die (diskretisierten) stationären Navier-Stokes-Gleichungen als Constraints im Optimierungsproblem betrachtet werden. Das verwendete SQP-Verfahren wäre mit der großen Anzahl zusätzlicher Constraints überfordert, mit dem PRSQP-Verfahren [Sch97] sollte es aber möglich sein, dieses Problem anzugehen. Dafür ist jedoch zunächst ein effizientes Verfahren zur Lösung der stationären Navier-Stokes-Gleichung notwendig.

Abschließend kann festgestellt werden, daß mit dem in dieser Arbeit entwickelten Verfahren eine Methode zur Optimierung dreidimensionaler Strömungen vorliegt, mit der relativ einfache Probleme in vernünftiger Zeit gelöst werden können und die, vom Strömungs- als auch vom Optimierungsteil gesehen, als guter Ausgangspunkt für weitere Entwicklungen dienen kann.

Anhang A

A.1 Berechnungen zu Kapitel 2.1

A.1.1 Beweis von (2.6)

Es gilt

$$\sum_{\alpha=1}^{3} \frac{\partial}{\partial \xi^\alpha}(\sqrt{g}\boldsymbol{a}^{(\alpha)}) = 0 \qquad (*)$$

denn

$$\begin{aligned}
\sum_{\alpha=1}^{3} \frac{\partial}{\partial \xi^\alpha}(\sqrt{g}\boldsymbol{a}^{(\alpha)}) &= \frac{\partial}{\partial \xi^1}(\boldsymbol{a}_{(2)} \times \boldsymbol{a}_{(3)}) + \frac{\partial}{\partial \xi^2}(\boldsymbol{a}_{(3)} \times \boldsymbol{a}_{(1)}) + \frac{\partial}{\partial \xi^3}(\boldsymbol{a}_{(1)} \times \boldsymbol{a}_{(2)}) \\
&= \frac{\partial \boldsymbol{a}_{(2)}}{\partial \xi^1} \times \boldsymbol{a}_{(3)} + \boldsymbol{a}_{(2)} \times \frac{\partial \boldsymbol{a}_{(3)}}{\partial \xi^1} + \\
&\quad \frac{\partial \boldsymbol{a}_{(3)}}{\partial \xi^2} \times \boldsymbol{a}_{(1)} + \boldsymbol{a}_{(3)} \times \frac{\partial \boldsymbol{a}_{(1)}}{\partial \xi^2} + \\
&\quad \frac{\partial \boldsymbol{a}_{(1)}}{\partial \xi^3} \times \boldsymbol{a}_{(2)} + \boldsymbol{a}_{(1)} \times \frac{\partial \boldsymbol{a}_{(2)}}{\partial \xi^3} \\
&= \frac{\partial^2 \boldsymbol{x}}{\partial \xi^1 \partial \xi^2} \times \boldsymbol{a}_{(3)} - \frac{\partial^2 \boldsymbol{x}}{\partial \xi^1 \partial \xi^3} \times \boldsymbol{a}_{(2)} \\
&\quad \frac{\partial^2 \boldsymbol{x}}{\partial \xi^2 \partial \xi^3} \times \boldsymbol{a}_{(1)} - \frac{\partial^2 \boldsymbol{x}}{\partial \xi^2 \partial \xi^1} \times \boldsymbol{a}_{(3)} \\
&\quad \frac{\partial^2 \boldsymbol{x}}{\partial \xi^3 \partial \xi^1} \times \boldsymbol{a}_{(2)} - \frac{\partial^2 \boldsymbol{x}}{\partial \xi^3 \partial \xi^2} \times \boldsymbol{a}_{(1)} = 0.
\end{aligned}$$

Damit folgt (2.6), denn

$$\begin{aligned}\frac{\partial \Phi}{\partial x^\alpha} &= \sum_{\beta=1}^{3} \frac{\partial \xi^\beta}{\partial x^\alpha}\frac{\partial \Phi}{\partial \xi^\beta} = \sum_{\beta} \boldsymbol{a}_\alpha^{(\beta)} \frac{\partial \Phi}{\partial \xi^\beta} \\ &= \sum_{\beta=1}^{3} \frac{1}{\sqrt{g}}\Big[\underbrace{\big(\frac{\partial}{\partial \xi^\beta}(\sqrt{g}\boldsymbol{a}_\alpha^{(\beta)}) \big)}_{=0 \text{ nach } (*)} \Phi + \sqrt{g}\boldsymbol{a}_\alpha^{(\beta)} \frac{\partial \Phi}{\partial \xi^\beta}\Big] \\ &= \sum_{\beta=1}^{3} \frac{1}{\sqrt{g}} \frac{\partial}{\partial \xi^\beta}(\sqrt{g}\boldsymbol{a}_\alpha^{(\beta)} \Phi).\end{aligned}$$

A.1.2 Diskretisierung der geometrischen Größen

Die Abbildung $\boldsymbol{x}$ ist nur in den Ecken von Zellen vorgegeben und wird dazwischen mittels trilinearer Interpolation fortgesetzt. Für $\zeta = (\zeta^1, \zeta^2, \zeta^3) \in [-\frac{1}{2}, \frac{1}{2}]^3$ und $\xi = (\xi^1, \xi^2, \xi^3) = \boldsymbol{j} + \zeta$ hat diese die Form

$$\begin{aligned}\boldsymbol{x}(\xi) = \boldsymbol{x}_{j+\zeta} &= \sum_{\text{Vorzeichen}} \boldsymbol{x}_{\boldsymbol{j}\pm e_1 \pm e_2 \pm e_3} \cdot \Big(\frac{1}{2} \pm \zeta^1\Big)\Big(\frac{1}{2} \pm \zeta^2\Big)\Big(\frac{1}{2} \pm \zeta^3\Big) \\ &= \boldsymbol{x}_0 + \sum_{\alpha} \boldsymbol{c}_\alpha (\xi^\alpha - j^\alpha) + \sum_{\alpha<\beta} \boldsymbol{c}_{\alpha\beta}(\xi^\alpha - j^\alpha)(\xi^\beta - j^\beta) + \\ &\quad \boldsymbol{c}_{123}(\xi^1 - j^1)(\xi^2 - j^2)(\xi^3 - j^3)\end{aligned} \tag{A.1}$$

mit Konstanten $\boldsymbol{x}_0$ und $\boldsymbol{c}$ die hier nicht näher bestimmt werden. Zunächst gilt für das Volumen $|\Omega_j|$ nach der Transformationsregel [Kön93]

$$|\Omega_j| = \int_{\Omega_j} d\Omega = \int_{\boldsymbol{j}+[-\frac{1}{2},\frac{1}{2}]^3} |J|\, d\xi^1 d\xi^2 d\xi^3.$$

Da J positiv ist, gilt mit der Variablentransformation $\xi^\alpha = j^\alpha + \frac{1}{2}s^\alpha$

$$|\Omega_j| = \frac{1}{8}\int_{[-1,1]^3} J\, ds^1 ds^2 ds^3. \tag{A.2}$$

Für die Jacobideterminante sind die Ableitungen von (A.1) zu berechnen. Diese lauten

$$\begin{aligned}\frac{\partial \boldsymbol{x}}{\partial \xi^1} &= \boldsymbol{c}_1 + \frac{1}{2}\boldsymbol{c}_{12}s^2 + \frac{1}{2}\boldsymbol{c}_{13}s^3 + \frac{1}{4}\boldsymbol{c}_{123}s^2 s^3 \\ \frac{\partial \boldsymbol{x}}{\partial \xi^2} &= \boldsymbol{c}_2 + \frac{1}{2}\boldsymbol{c}_{12}s^1 + \frac{1}{2}\boldsymbol{c}_{23}s^3 + \frac{1}{4}\boldsymbol{c}_{123}s^1 s^3 \\ \frac{\partial \boldsymbol{x}}{\partial \xi^3} &= \boldsymbol{c}_3 + \frac{1}{2}\boldsymbol{c}_{13}s^1 + \frac{1}{2}\boldsymbol{c}_{23}s^2 + \frac{1}{4}\boldsymbol{c}_{123}s^1 s^2.\end{aligned} \tag{A.3}$$

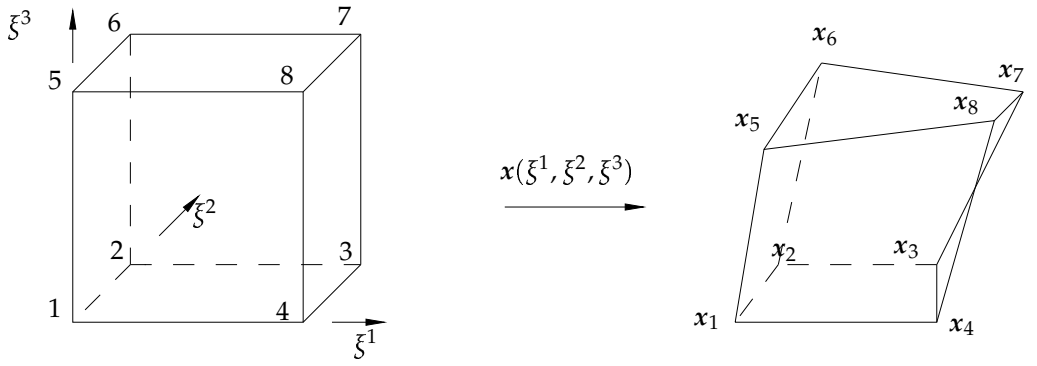

Abbildung A.1: Eckpunkte einer Zelle

Nun berechne das Integral (A.2). Da die Integrationsgrenzen symmetrisch sind, fallen Terme bei denen ein s^α nach Integration geraden Exponenten hat weg. Daher gilt

$$\begin{aligned}
|\Omega_j| &= \frac{1}{8}\int_{[-1,1]^3} \frac{\partial \boldsymbol{x}}{\partial \xi^1} \cdot \Big(\frac{\partial \boldsymbol{x}}{\partial \xi^2} \times \frac{\partial \boldsymbol{x}}{\partial \xi^3}\Big)\, ds^1 ds^2 ds^3 \\
&= \frac{1}{8}\int_{[-1,1]^3} \Big[\boldsymbol{c}_1 \cdot (\boldsymbol{c}_2 \times \boldsymbol{c}_3) + \frac{1}{4} s^1 s^1 \boldsymbol{c}_1 \cdot (\boldsymbol{c}_{12} \times \boldsymbol{c}_{13}) + \\
&\quad \frac{1}{4} s^2 s^2 \boldsymbol{c}_{12} \cdot (\boldsymbol{c}_2 \times \boldsymbol{c}_{23}) + \frac{1}{4} s^3 s^3 \boldsymbol{c}_{13} \cdot (\boldsymbol{c}_{23} \times \boldsymbol{c}_3)) + \\
&\quad \frac{1}{64} s^1 s^1 s^2 s^2 s^3 s^3 \boldsymbol{c}_{123} \cdot (\boldsymbol{c}_{123} \times \boldsymbol{c}_{123})\Big]\, ds^1 ds^2 ds^3 \\
&= \boldsymbol{c}_1 \cdot (\boldsymbol{c}_2 \times \boldsymbol{c}_3) + \frac{1}{12}\Big[\boldsymbol{c}_1 \cdot (\boldsymbol{c}_{12} \times \boldsymbol{c}_{13}) + \\
&\quad \boldsymbol{c}_{12} \cdot (\boldsymbol{c}_2 \times \boldsymbol{c}_{23}) + \boldsymbol{c}_{13} \cdot (\boldsymbol{c}_{23} \times \boldsymbol{c}_3)\Big].
\end{aligned}$$

Schreibe nun (A.1) in s-Koordinaten, also

$$\boldsymbol{x}(s^1, s^2, s^3) = \boldsymbol{x}_0 + \sum_\alpha \boldsymbol{b}_\alpha s^\alpha + \sum_{\alpha<\beta} \boldsymbol{b}_{\alpha\beta} s^\alpha s^\beta + \boldsymbol{b}_{123} s^1 s^2 s^3$$

mit $\boldsymbol{b}_\alpha = \frac{1}{2}\boldsymbol{c}_\alpha$, $\boldsymbol{b}_{\alpha\beta} = \frac{1}{4}\boldsymbol{c}_{\alpha\beta}$, $\boldsymbol{b}_{123} = \frac{1}{8}\boldsymbol{c}_{123}$. Für die acht Eckpunkte $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_8$ (Abbildung A.1) muß $y = (\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{b}_{12}, \boldsymbol{b}_{13}, \boldsymbol{b}_{23}, \boldsymbol{b}_{123}, \boldsymbol{x}_0)^T$ so gewählt werden, daß $Ay = r = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_8)^T$ mit

$$A = \begin{pmatrix}
-1 & -1 & -1 & 1 & 1 & 1 & -1 & 1 \\
-1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 \\
1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 \\
1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 \\
-1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\
-1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & -1 & 1
\end{pmatrix}.$$

Nachrechnen zeigt, daß die Spalten von A orthogonal und 8 Einheiten lang ist, also $y = \frac{1}{8}A^T r$. Mit $\boldsymbol{x}_{ijkl} = \boldsymbol{x}_i + \boldsymbol{x}_j + \boldsymbol{x}_k + \boldsymbol{x}_l$ ist dann

$$\begin{array}{rclrcl} \boldsymbol{b}_1 &=& \frac{1}{8}(\boldsymbol{x}_{3478} - \boldsymbol{x}_{1256}), & \boldsymbol{b}_2 &=& \frac{1}{8}(\boldsymbol{x}_{2367} - \boldsymbol{x}_{1458}), \\ \boldsymbol{b}_3 &=& \frac{1}{8}(\boldsymbol{x}_{5678} - \boldsymbol{x}_{1234}), & \boldsymbol{b}_{12} &=& \frac{1}{8}(\boldsymbol{x}_{1357} - \boldsymbol{x}_{2468}), \\ \boldsymbol{b}_{13} &=& \frac{1}{8}(\boldsymbol{x}_{1278} - \boldsymbol{x}_{3456}), & \boldsymbol{b}_{23} &=& \frac{1}{8}(\boldsymbol{x}_{1467} - \boldsymbol{x}_{2358}), \\ \boldsymbol{b}_{123} &=& \frac{1}{8}(\boldsymbol{x}_{2457} - \boldsymbol{x}_{1368}), & \boldsymbol{x}_0 &=& \frac{1}{8}(\boldsymbol{x}_{1234} + \boldsymbol{x}_{5678}). \end{array} \tag{A.4}$$

Damit folgt

$$\begin{aligned} |\Omega_j| = 8\boldsymbol{b}_1 \cdot (\boldsymbol{b}_2 \times \boldsymbol{b}_3) + \frac{8}{3}\big[&\boldsymbol{b}_1 \cdot (\boldsymbol{b}_{12} \times \boldsymbol{b}_{13}) + \\ &\boldsymbol{b}_{12} \cdot (\boldsymbol{b}_2 \times \boldsymbol{b}_{23}) + \boldsymbol{b}_{13} \cdot (\boldsymbol{b}_{23} \times \boldsymbol{b}_3)\big]. \end{aligned} \tag{A.5}$$

Betrachte die Oberfläche O mit $\xi^3 \equiv$ const. (in Abbildung A.1 die Oberfläche 1432). Ein Normalenvektor auf dieser Fläche mit einer Länge gleich dem Volumen dieser Fläche in der Richtung von steigendem ξ^3 ist dann gegeben durch (siehe [Kön93])

$$\begin{aligned} \boldsymbol{s}_{1432} &= \int_O \boldsymbol{n}\, d\Gamma = \int_{[-\frac{1}{2},\frac{1}{2}]^2} \frac{\partial_1\boldsymbol{x} \wedge \partial_2\boldsymbol{x}}{\|\partial_1\boldsymbol{x} \wedge \partial_2\boldsymbol{x}\|} \cdot \|\partial_1\boldsymbol{x} \wedge \partial_2\boldsymbol{x}\|\, d\xi^1 d\xi^2 \\ &= \int_{[-\frac{1}{2},\frac{1}{2}]^2} \partial_1\boldsymbol{x} \times \partial_2\boldsymbol{x}\, d\xi^1 d\xi^2. \end{aligned}$$

Schreibt man (A.1) in s-Koordinaten $s^\alpha := 2(\xi^\alpha - j^\alpha)$ mit $s^3 \equiv$ const., so ist

$$\boldsymbol{x}(s^1, s^2, s^3) = \boldsymbol{x}_0 + \frac{1}{2}\boldsymbol{d}_1 s^1 + \frac{1}{2}\boldsymbol{d}_2 s^2 + \frac{1}{4}\boldsymbol{d}_{12} s^1 s^2 \tag{A.6}$$

mit Konstanten $\boldsymbol{x}_0$ und $\boldsymbol{d}$. Für die Ableitungen gilt dann

$$\partial_1\boldsymbol{x} = 2\frac{\partial\boldsymbol{x}}{\partial s^1} = \boldsymbol{d}_1 + \frac{1}{2}\boldsymbol{d}_{12} s^2 \quad \text{bzw.} \quad \partial_2\boldsymbol{x} = 2\frac{\partial\boldsymbol{x}}{\partial s^2} = \boldsymbol{d}_2 + \frac{1}{2}\boldsymbol{d}_{12} s^1. \tag{A.7}$$

Nach Transformation in s-Koordinaten ist dann

$$\boldsymbol{s}_{1432} = \frac{1}{4}\int_{[-1,1]^2} \partial_1\boldsymbol{x} \times \partial_2\boldsymbol{x}\, ds^1 ds^2 = \frac{1}{4}\int_{[-1,1]^2} \boldsymbol{d}_1 \times \boldsymbol{d}_2\, ds^1 ds^2 = \boldsymbol{d}_1 \times \boldsymbol{d}_2,$$

denn Terme, die nach Integration ein s^α mit geradem Exponenten enthalten, fallen wieder weg. Die Gleichung (A.6) muß für $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_4$ und entsprechende s^α erfüllt sein, also gilt $Ay = r = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_4)^T$ mit $y = (\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_{12}, \boldsymbol{x}_0)^T$ und

$$A = \begin{pmatrix} -\frac{1}{2} & -\frac{1}{2} & \frac{1}{4} & 1 \\ -\frac{1}{2} & \frac{1}{2} & -\frac{1}{4} & 1 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{4} & 1 \\ \frac{1}{2} & -\frac{1}{2} & -\frac{1}{4} & 1 \end{pmatrix} \quad \Rightarrow \quad A^{-1} = \begin{pmatrix} -\frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ 1 & -1 & 1 & -1 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}.$$

Damit folgt $y = A^{-1}r$, also mit $\boldsymbol{x}_{ij} = \boldsymbol{x}_i + \boldsymbol{x}_j$ und wegen $a \times a = 0$

$$\boldsymbol{d}_1 = \frac{1}{2}(\boldsymbol{x}_{34} - \boldsymbol{x}_{12}), \quad \boldsymbol{d}_2 = \frac{1}{2}(\boldsymbol{x}_{23} - \boldsymbol{x}_{14}), \quad \boldsymbol{d}_{12} = \boldsymbol{x}_{13} - \boldsymbol{x}_{24}, \quad \boldsymbol{x}_0 = \frac{1}{4}\boldsymbol{x}_{1234} \tag{A.8}$$

und damit

$$\begin{aligned}
\boldsymbol{s}_{1432} &= \frac{1}{4}(\boldsymbol{x}_{34} - \boldsymbol{x}_{12}) \times (\boldsymbol{x}_{23} - \boldsymbol{x}_{14}) \\
&= \frac{1}{4}(\boldsymbol{x}_3 \times \boldsymbol{x}_2 - \boldsymbol{x}_3 \times \boldsymbol{x}_1 - \boldsymbol{x}_3 \times \boldsymbol{x}_4 + \boldsymbol{x}_4 \times \boldsymbol{x}_2 + \boldsymbol{x}_4 \times \boldsymbol{x}_3 - \boldsymbol{x}_4 \times \boldsymbol{x}_1 - \\
&\qquad \boldsymbol{x}_1 \times \boldsymbol{x}_2 - \boldsymbol{x}_1 \times \boldsymbol{x}_3 + \boldsymbol{x}_1 \times \boldsymbol{x}_4 - \boldsymbol{x}_2 \times \boldsymbol{x}_3 + \boldsymbol{x}_2 \times \boldsymbol{x}_1 + \boldsymbol{x}_2 \times \boldsymbol{x}_4) \\
&= \frac{1}{2}(\boldsymbol{x}_4 \times \boldsymbol{x}_3 - \boldsymbol{x}_4 \times \boldsymbol{x}_1 - \boldsymbol{x}_2 \times \boldsymbol{x}_3 + \boldsymbol{x}_2 \times \boldsymbol{x}_1) \\
&= \frac{1}{2}(\boldsymbol{x}_4 - \boldsymbol{x}_2) \times (\boldsymbol{x}_3 - \boldsymbol{x}_1).
\end{aligned}$$

Für alle Flächen lauten diese Vektoren dann $\boldsymbol{s}_{1265}$, $\boldsymbol{s}_{4378}$, $\boldsymbol{s}_{1584}$, $\boldsymbol{s}_{2673}$, $\boldsymbol{s}_{1432}$ und $\boldsymbol{s}_{8765}$ mit

$$\boldsymbol{s}_{ijkl} = \frac{1}{2}(\boldsymbol{x}_j - \boldsymbol{x}_l) \times (\boldsymbol{x}_k - \boldsymbol{x}_i).$$

Direktes Nachrechnen zeigt nun, daß das Volumen einer Zelle (A.5) als

$$\begin{aligned}
|\Omega_j| &= \frac{8}{3}\big(\boldsymbol{b}_1 \cdot (\boldsymbol{b}_2 \times \boldsymbol{b}_3 + \boldsymbol{b}_{12} \times \boldsymbol{b}_{13}) + \boldsymbol{b}_2 \cdot (\boldsymbol{b}_3 \times \boldsymbol{b}_1 + \boldsymbol{b}_{23} \times \boldsymbol{b}_{12}) + \\
&\qquad \boldsymbol{b}_3 \cdot (\boldsymbol{b}_1 \times \boldsymbol{b}_2 + \boldsymbol{b}_{13} \times \boldsymbol{b}_{23})\big).
\end{aligned}$$

und schließlich als

$$|\Omega_j| = \frac{1}{3}\big(\boldsymbol{b}_1 \cdot (\boldsymbol{s}_{1265} + \boldsymbol{s}_{4378}) + \boldsymbol{b}_2 \cdot (\boldsymbol{s}_{1584} + \boldsymbol{s}_{2673}) + \boldsymbol{b}_3 \cdot (\boldsymbol{s}_{1432} + \boldsymbol{s}_{8765})\big)$$

geschrieben werden kann.

A.1.3 Interpolationsregeln

In diesem Abschnitt werden die Interpolationsregel beschrieben, die notwendig sind, um $\boldsymbol{u}$ in verschiedenen Punkten einer Zelle zu ermitteln, so daß Forderung (2.12) erfüllt ist. In Zellmittelpunkten $\boldsymbol{x}_j$ definiere (siehe Abbildung A.2)

$$\begin{aligned}
V_j^\alpha &= \frac{1}{2}(V_{j-e_\alpha}^\alpha + V_{j+e_\alpha}^\alpha), \\
(\sqrt{g}\boldsymbol{a}^{(\alpha)})_j &= \frac{1}{2}\big((\sqrt{g}\boldsymbol{a}^{(\alpha)})_{j-e_\alpha} + (\sqrt{g}\boldsymbol{a}^{(\alpha)})_{j+e_\alpha}\big).
\end{aligned}$$

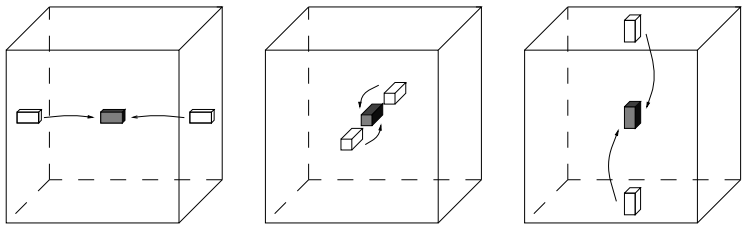

Abbildung A.2: Interpolation in Zellmittelpunkten

Die für die Berechnung von $\boldsymbol{u}$ aus V^α benötigten Größen $(\boldsymbol{a}_{(\alpha)}/\sqrt{g})_\xi$ werden an allen Stellen ξ wo sie benötigt werden mit (2.5) aus $(\sqrt{g}\boldsymbol{a}^{(\alpha)})_\xi$ berechnet bzw. an diesen Stellen wird ein entsprechendes lineares Gleichungssystem zur Bestimmung von $\boldsymbol{u}$ gelöst. Nachrechnen zeigt, daß die geforderte Eigenschaft (2.12) erfüllt ist:

$$\begin{aligned} V_j^\alpha &= \frac{1}{2}\big((\sqrt{g}\boldsymbol{a}^{(\alpha)})_{j-e^\alpha}\cdot\boldsymbol{u} + (\sqrt{g}\boldsymbol{a}^{(\alpha)})_{j+e^\alpha}\cdot\boldsymbol{u}\big) \\ &= \frac{1}{2}\big((\sqrt{g}\boldsymbol{a}^{(\alpha)})_{j-e^\alpha} + (\sqrt{g}\boldsymbol{a}^{(\alpha)})_{j+e^\alpha}\big)\cdot\boldsymbol{u} \\ &= (\sqrt{g}\boldsymbol{a}^{(\alpha)})_j\cdot\boldsymbol{u}. \end{aligned}$$

Da $(\boldsymbol{a}_{(\alpha)}/\sqrt{g})_j$ gerade aus $(\sqrt{g}\boldsymbol{a}^{(\alpha)})_j$ berechnet wird, folgt die Behauptung. Die folgenden Interpolationsregeln rechnet man genauso nach. Sei weiterhin $\Phi^\alpha \in \{V^\alpha, \sqrt{g}\boldsymbol{a}^{(\alpha)}\}$. Für Mittelpunkte der Seitenflächen $\boldsymbol{x}_{j+e_\alpha}$ definiere für $\alpha \neq \beta$ (siehe Abbildung A.3)

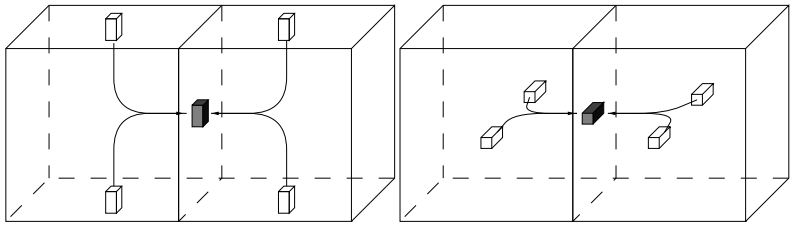

Abbildung A.3: Interpolation in Flächenmittelpunkten

$$\Phi^\beta_{j+e_\alpha} = \frac{1}{4}\Big[\Phi^\beta_{j+e_\beta} + \Phi^\beta_{j-e_\beta} + \Phi^\beta_{j+2e_\alpha+e_\beta} + \Phi^\beta_{j+2e_\alpha-e_\beta}\Big].$$

Für Kantenmittelpunkte $\boldsymbol{x}_{j+e_\alpha+e_\beta}$ definiere (siehe Abbildung A.4)

$$\Phi^\alpha_{j+e_\alpha+e_\beta} = \frac{1}{2}\Big[\Phi^\alpha_{j+e_\alpha} + \Phi^\alpha_{j+e_\alpha+2e_\beta}\Big]$$

und für $\gamma \neq \alpha, \beta$

$$\begin{aligned} \Phi^\gamma_{j+e_\alpha+e_\beta} &= \frac{1}{8}\Big[\Phi^\gamma_{j-e_\gamma} + \Phi^\gamma_{j+e_\gamma} + \Phi^\gamma_{j-e_\gamma+2e_\alpha} + \Phi^\gamma_{j+e_\gamma+2e_\alpha} + \\ &\quad \Phi^\gamma_{j-e_\gamma+2e_\beta} + \Phi^\gamma_{j+e_\gamma+2e_\beta} + \Phi^\gamma_{j-e_\gamma+2e_\alpha+2e_\beta} + \Phi^\gamma_{j+e_\gamma+2e_\alpha+2e_\beta}\Big] \end{aligned}$$

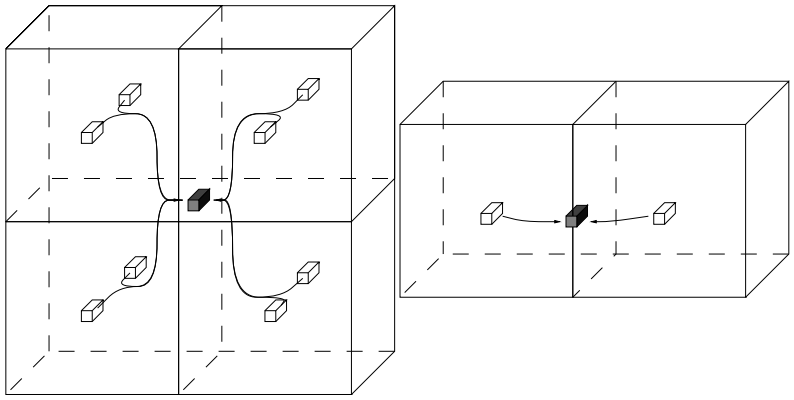

Abbildung A.4: Interpolation in Kantenmittelpunkten

Schließlich definiere für Ecken $\boldsymbol{x}_{\boldsymbol{j}+e_1+e_2+e_3}$ (siehe Abbildung A.5)

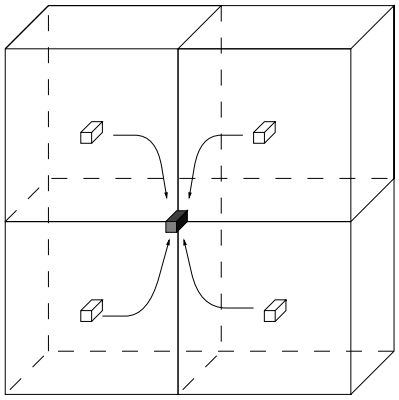

Abbildung A.5: Interpolation in Ecken

$$\Phi^{\alpha}_{\boldsymbol{j}+e_1+e_2+e_3} = \frac{1}{4}\Big[\Phi^{\alpha}_{\boldsymbol{j}+e_\alpha} + \Phi^{\alpha}_{\boldsymbol{j}+e_\alpha+2e_\beta} + \Phi^{\alpha}_{\boldsymbol{j}+e_\alpha+2e_\gamma} + \Phi^{\alpha}_{\boldsymbol{j}+e_\alpha+2e_\beta+2e_\gamma}\Big].$$

A.1.4 Beweis der Konsistenzordnungen

Ist die Lösungsfunktion $\varphi(\boldsymbol{x})$ genügend oft differenzierbar, so ist eine Diskretisierung konsistent von Ordnung p, falls sie für Polynome vom Grad $p-1$ exakt ist. Dies folgt sofort durch Einsetzen der Taylorentwicklung von φ in die Definition des lokalen Diskretisierungsfehlers (2.26).

Konsistenzordnung des Divergenzoperators D

Sei

$$\boldsymbol{u}(\xi^1,\xi^2,\xi^3) = \boldsymbol{u}_0 + \boldsymbol{u}_{11}(\xi^1 - j^1) + \boldsymbol{u}_{12}(\xi^2 - j^2) + \boldsymbol{u}_{13}(\xi^3 - j^3).$$

Es ist zu zeigen (vgl. 2.10)

$$\int_{\partial\Omega_j} \boldsymbol{u}\cdot\boldsymbol{n}\,d\Gamma = \sum_{\alpha=1}^{3} (\boldsymbol{u}\cdot\boldsymbol{s})\big|_{j-e_\alpha}^{j+e_\alpha}.$$

Betrachte zunächst nur eine Seitenfläche der Zelle. Da die Transformationsabbildung durch trilineare Interpolation zwischen den Eckpunkten definiert ist, hat die Seitenfläche mit $\xi^3 \equiv$ const. (in Abbildung A.1 die Oberfläche 1432) nach Transformation in s-Koordinaten $s^\alpha := 2(\xi^\alpha - j^\alpha)$ (vgl. (A.6)) die bilineare Form

$$\boldsymbol{x}(s^1,s^2,s^3) = \boldsymbol{x}_0 + \frac{1}{2}\boldsymbol{d}_1 s^1 + \frac{1}{2}\boldsymbol{d}_2 s^2 + \frac{1}{4}\boldsymbol{d}_{12} s^1 s^2$$

mit $\boldsymbol{x}_0, \boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_{12}$ wie in (A.8). Mit den entsprechenden Ableitungen von $\boldsymbol{x}$ (vgl. (A.7)) gilt dann für den Normalenvektor in Richtung von steigendem ξ^3

$$\begin{aligned}
\boldsymbol{n} &= \big(\boldsymbol{d}_1 + \frac{1}{2}\boldsymbol{d}_{12}s^2\big) \times \big(\boldsymbol{d}_2 + \frac{1}{2}\boldsymbol{d}_{12}s^1\big) \\
&= \boldsymbol{d}_1\times\boldsymbol{d}_2 + \frac{1}{2}\boldsymbol{d}_{12}s^2\times\boldsymbol{d}_2 + \boldsymbol{d}_1\times\frac{1}{2}\boldsymbol{d}_{12}s^1 + \frac{1}{2}\boldsymbol{d}_{12}s^1\times\frac{1}{2}\boldsymbol{d}_{12}s^2 \\
&= \boldsymbol{s} + \frac{1}{2}s^2\boldsymbol{d}_{12}\times\boldsymbol{d}_2 + \frac{1}{2}s^1\boldsymbol{d}_1\times\boldsymbol{d}_{12}.
\end{aligned}$$

Das oben definierte $\boldsymbol{u}$ hat in s-Koordinaten die Form

$$\boldsymbol{u}(s^1,s^2,s^3) = \boldsymbol{u}_0 + \frac{1}{2}\big(\boldsymbol{u}_{11}s^1 + \boldsymbol{u}_{12}s^2 + \boldsymbol{u}_{13}s^3\big).$$

Damit berechnet sich das in s-Koordinaten transformierte Integral über diese Seitenfläche ($s^3 = -1$) unter weglassen der aufgrund symmetrischer Integrationsgrenzen wegfallender Summanden als

$$\begin{aligned}
&\frac{1}{4}\int_{[-1,1]^2} \boldsymbol{u}\cdot\boldsymbol{n}\,ds^1ds^2 \\
&= \frac{1}{4}\int_{[-1,1]^2} \Big[\boldsymbol{u}_0 + \frac{1}{2}\big(\boldsymbol{u}_{11}s^1 + \boldsymbol{u}_{12}s^2 + \boldsymbol{u}_{13}s^3\big)\Big]\cdot\Big[\boldsymbol{s} + \frac{1}{2}s^2\boldsymbol{d}_{12}\times\boldsymbol{d}_2 + \frac{1}{2}s^1\boldsymbol{d}_1\times\boldsymbol{d}_{12}\Big]\,ds^1ds^2 \\
&= \frac{1}{4}\int_{[-1,1]^2} \big(\boldsymbol{u}_0 - \frac{1}{2}\boldsymbol{u}_{13}\big)\cdot\boldsymbol{s}\,ds^1ds^2 + \\
&\quad \frac{1}{4}\int_{[-1,1]^2} \boldsymbol{u}_{11}\cdot\frac{1}{2}\boldsymbol{d}_1\times\boldsymbol{d}_{12}s^1s^1 + \boldsymbol{u}_{12}\cdot\frac{1}{2}\boldsymbol{d}_{12}\times\boldsymbol{d}_2 s^2s^2\,ds^1ds^2 \\
&= \boldsymbol{u}_0\cdot\boldsymbol{s} - \frac{1}{2}\boldsymbol{u}_{13}\cdot\boldsymbol{s} + \frac{1}{6}\Big[\boldsymbol{u}_{11}\cdot(\boldsymbol{d}_1\times\boldsymbol{d}_{12}) + \boldsymbol{u}_{12}\cdot(\boldsymbol{d}_{12}\times\boldsymbol{d}_2)\Big].
\end{aligned}$$

Es ist $\boldsymbol{d}_{12} = (\boldsymbol{x}_1 - \boldsymbol{x}_2) - (\boldsymbol{x}_4 - \boldsymbol{x}_3)$, also ist $\boldsymbol{d}_{12}$ und damit die eckige Klammer gleich Null, wenn jeweils zwei gegenüberliegende Seiten der Seitenfläche parallel und gleich lang

sind, diese Seitenfläche also ein Parallelogramm ist. Gilt dies für alle Seitenflächen, so ergibt sich insgesamt

$$\int_{\partial\Omega_j} \boldsymbol{u}\cdot\boldsymbol{n}\,d\Gamma = \sum_{\alpha=1}^{3}(\boldsymbol{u}_0+\frac{1}{2}\boldsymbol{u}_{1\alpha})\cdot\boldsymbol{s}_{j+e_\alpha} - \sum_{\alpha=1}^{3}(\boldsymbol{u}_0-\frac{1}{2}\boldsymbol{u}_{1\alpha})\cdot\boldsymbol{s}_{j-e_\alpha} = \sum_{\alpha=1}^{3}(\boldsymbol{u}\cdot\boldsymbol{s})|_{j-e_\alpha}^{j+e_\alpha}.$$

Dies beweist die Aussage über die Konsistenzordnung des Divergenzoperators $\boldsymbol{D}$.

Konsistenzordnung des Operators $\boldsymbol{M}$

Es ist zu zeigen, daß

$$\int_{\Omega_{j+e_\alpha}} \varphi\,d\Omega = |\Omega_{j+e_\alpha}|\varphi_{j+e_\alpha}.$$

Sei o.B.d.A. $\alpha = 1$ und sei

$$\varphi(\xi^1,\xi^2,\xi^3) = \varphi_0 + \varphi_{11}(\xi^1 - j^1 - \frac{1}{2}) + \varphi_{12}(\xi^2 - j^2) + \varphi_{13}(\xi^3 - j^3),$$

also in s-Koordinaten wie oben

$$\varphi(s^1,s^2,s^3) = \varphi_0 + \frac{1}{2}\big(\varphi_{11}(s^1-1) + \varphi_{12}s^2 + \varphi_{13}s^3\big).$$

Dann gilt (vgl. Volumenberechnung in Abschnitt A.1.2)

$$\begin{aligned}\int_{\Omega_{j+e_1}} \varphi\,d\Omega &= \int_{\Omega_{j+e_1}} \varphi_0 + \frac{1}{2}\big(\varphi_{11}(s^1-1) + \varphi_{12}s^2 + \varphi_{13}s^3\big)\,d\Omega \\ &= |\Omega_{j+e_1}|\varphi_0 + \frac{1}{16}\underbrace{\int_{j+e_1+[-1,1]^3} \big[\varphi_{11}(s^1-1) + \varphi_{12}s^2 + \varphi_{13}s^3\big]\,|J|\,d\xi^3 d\xi^2 d\xi^1}_{I}\end{aligned}$$

Die Jacobideterminante J ist positiv und für den Fall, daß beide Zellen, in denen das Kontrollvolumen Ω_{j+e_1} liegt, identische Parallelepipede sind, auch stetig und sogar konstant. Daher wertet sich das obige Integral I zu Null aus und die Behauptung über die Konsistenzordnung von $\boldsymbol{M}$ ist bewiesen.

Konsistenzordnung des Operators $A(\tilde{V})V$

Sei o.B.d.A. $\alpha = 1$ und seien zunächst $\boldsymbol{u} = \boldsymbol{u}_0$, $\tilde{\boldsymbol{u}} = \tilde{\boldsymbol{u}}_0$. Für den ersten Summanden des konvektiven Terms ist zu zeigen, daß

$$\int_{j^2-\frac{1}{2}}^{j^2+\frac{1}{2}}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}} (\tilde{V}^1\boldsymbol{u})|_{j^1}^{j^1+1}\,d\xi^3 d\xi^2 = (V^1\boldsymbol{u})_{\boldsymbol{j}}^{j+2e_1}. \tag{A.9}$$

Es gilt

$$\begin{aligned}\int_{j^2-\frac{1}{2}}^{j^2+\frac{1}{2}}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}}(\tilde{V}^1\boldsymbol{u})_{\xi^1=j^1}\,d\xi^3d\xi^2 &= \int_{j^2-\frac{1}{2}}^{j^2+\frac{1}{2}}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}}\big[(\sqrt{g}\boldsymbol{a}^{(1)}\cdot\tilde{\boldsymbol{u}})\boldsymbol{u}\big]_{\xi^1=j^1}\,d\xi^3d\xi^2\\ &= \frac{1}{4}\int_{[-1,1]^2}\Big[\big((\frac{\partial\boldsymbol{x}}{\partial\xi^2}\times\frac{\partial\boldsymbol{x}}{\partial\xi^3})\cdot\tilde{\boldsymbol{u}}\big)\boldsymbol{u}\Big]_{s^1=0}\,d\xi^3d\xi^2\\ &\overset{(A.3)}{=} \big[(\boldsymbol{c}_2\times\boldsymbol{c}_3)\cdot\tilde{\boldsymbol{u}}_0\big]\boldsymbol{u}_0=\big[(\sqrt{g}\boldsymbol{a}^{(1)}\cdot\tilde{\boldsymbol{u}}_0)\boldsymbol{u}_0\big]_j\end{aligned}$$

und analog

$$\int_{j^2-\frac{1}{2}}^{j^2+\frac{1}{2}}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}}(\tilde{V}^1\boldsymbol{u})_{\xi^1=j^1+1}\,d\xi^3d\xi^2=\big[(\sqrt{g}\boldsymbol{a}^{(1)}\cdot\tilde{\boldsymbol{u}}_0)\boldsymbol{u}_0\big]_{j+2e_1}.$$

Dies zeigt (A.9). Für die nächsten beiden Summanden ist zu zeigen (o.B.d.A. $\beta=2$), daß

$$\int_{j^1}^{j^1+1}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}}(\tilde{V}^2\boldsymbol{u})_{j^2-\frac{1}{2}}^{j^2+\frac{1}{2}}\,d\xi^3d\xi^1=(\tilde{V}^2\boldsymbol{u})\big|_{j+e_1-e_2}^{j+e_1+e_2}. \tag{A.10}$$

Da die Integration über ξ^1 eine Zellengrenze überschreitet, ist das Integral aufzuspalten:

$$\begin{aligned}&\int_{j^1}^{j^1+1}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}}(\tilde{V}^2\boldsymbol{u})_{\xi^2=j^2-\frac{1}{2}}\,d\xi^3d\xi^1\\ &= \int_{j^1}^{j^1+\frac{1}{2}}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}}(\tilde{V}^2\boldsymbol{u})_{\xi^2=j^2-\frac{1}{2}}\,d\xi^3d\xi^1+\int_{j^1+\frac{1}{2}}^{j^1+1}\int_{j^3-\frac{1}{2}}^{j^3+\frac{1}{2}}(\tilde{V}^2\boldsymbol{u})_{\xi^2=j^2-\frac{1}{2}}\,d\xi^3d\xi^1\\ &= \frac{1}{4}\int_0^1\int_{-1}^1\Big[(\sqrt{g}\boldsymbol{a}^{(2)}\cdot\tilde{\boldsymbol{u}})\boldsymbol{u}\Big]_{s^2=1}ds^3ds^1+\frac{1}{4}\int_1^2\int_{-1}^1\Big[(\sqrt{g}\boldsymbol{a}^{(2)}\cdot\tilde{\boldsymbol{u}})\boldsymbol{u}\Big]_{s^2=1}ds^3ds^1\\ &= \frac{1}{2}\Big[\big((\sqrt{g}\boldsymbol{a}^{(2)})_{j+\frac{1}{2}e_1-e_2}+(\sqrt{g}\boldsymbol{a}^{(2)})_{j+\frac{3}{2}e_1-e_2}\big)\cdot\tilde{\boldsymbol{u}}_0\Big]\boldsymbol{u}_0\\ &= \frac{1}{4}\Big[\big((\sqrt{g}\boldsymbol{a}^{(2)})_{j-e_2}+(\sqrt{g}\boldsymbol{a}^{(2)})_{j+e_1-e_2}+(\sqrt{g}\boldsymbol{a}^{(2)})_{j+e_1-e_2}+(\sqrt{g}\boldsymbol{a}^{(2)})_{j+2e_1-e_2}\big)\cdot\tilde{\boldsymbol{u}}_0\Big]\boldsymbol{u}_0\\ &= \frac{1}{4}\Big(\tilde{V}^2_{j-e_2}+2\tilde{V}^2_{j+e_1-e_2}+\tilde{V}_{j+2e_1-e_2}\Big)\boldsymbol{u}_0=(\tilde{V}^2\boldsymbol{u})_{j+e_1-e_2}.\end{aligned}$$

Damit ist auch (A.10) bewiesen, also ist die Diskretisierung des konvektiven Terms von erster Ordnung. Seien nun die beiden Zellen, in denen das Kontrollvolumen $|\Omega_{j+e_1}|$ liegt, identische Parallelepipede. Dann ist $\sqrt{g}\boldsymbol{a}^{(\alpha)}$ konstant, $1\le\alpha\le3$. Da die Integrationsgrenzen symmetrisch sind, ist die Diskretisierung für

$$\begin{aligned}\boldsymbol{u}(\xi^1,\xi^2,\xi^3) &= \boldsymbol{u}_0+\boldsymbol{u}_{11}(\xi^1-j^1-\frac{1}{2})+\boldsymbol{u}_{12}(\xi^2-j^2)+\boldsymbol{u}_{13}(\xi^3-j^3),\\ \tilde{\boldsymbol{u}}(\xi^1,\xi^2,\xi^3) &= \tilde{\boldsymbol{u}}_0+\tilde{\boldsymbol{u}}_{11}(\xi^1-j^1-\frac{1}{2})+\tilde{\boldsymbol{u}}_{12}(\xi^2-j^2)+\tilde{\boldsymbol{u}}_{13}(\xi^3-j^3),\end{aligned}$$

exakt (vgl. Rechnung zur Konsistenzordnung von $\boldsymbol{M}$). Damit ist die Diskretisierung des konvektiven Terms in diesem Fall von zweiter Ordnung.

Konsistenzordnung des Operators G

Es wurde bereits gezeigt, daß die Diskretisierung (2.16) für konstantes ∇p exakt ist. Für konstantes ∇p sind auch die Diskretisierungen (2.17) und (2.18) exakt. Daher ist die Diskretisierung von $\boldsymbol{G}$ exakt für Polynome ersten Grades, also ist sie konsistent von zweiter Ordnung.

Konsistenzordnung des Operators B

Zunächst kann festgestellt werden, daß die Diskretisierungen (2.19) und (2.20) für Polynome ersten Grades exakt sind. Eine analoge Vorgehensweise wie bei der Berechnung der Konsistenzordnung von $\boldsymbol{A}(\tilde{V})$ liefert die behauptete Konsistenzordnung von $\boldsymbol{B}$.

Anhang B

Die Strömungsproblembeschreibungssprache

Im diesem Kapitel soll die Beschreibungssprache für die Strömungsprobleme dokumentiert werden. `Terminale` und *nichtterminale* Symbole werden durch eine entsprechende Schriftart hervorgehoben. Optionale Symbole werden in eckigen, sich wiederholende (auch 0-mal) Symbole in geschweiften Klammern { } geschrieben. Diese sind von Terminalen `{ }` zu unterscheiden. Regeln, welche kursive Terminale enthalten, dürfen in im jeweiligen Block nur einmal angewendet werden. Sind sie zusätzlich unterstrichen, so müssen sie mindestens einmal angewendet werden. Alternativen werden durch senkrechte Striche getrennt. Die Symbole *num_int* und *num_double* bezeichnen entsprechende Zahlen in C/C++-Notation, *string* bezeichnet eine Zeichenkette, die mit Anführungsstrichen (”...”) abgegrenzt ist.

Durch die folgenden, weitgehend selbsterklärenden Regeln werden mathematische Ausdrücke ohne Variablen definiert:

expr ::= *mexpr* { `+` *mexpr* | `-` *mexpr* }.
mexpr ::= *expexpr* { `*` *expexpr* | `/` *expexpr* }.
expexpr ::= *unaryexpr* [`^` *unaryexpr*].
unaryexpr ::= [+] *primexpr* | `-` *primexpr*.
primexpr ::= *number* | `(` *expr* `)` |
`acos(`*expr*`)` | `asin(`*expr*`)` | `atan(`*expr*`)` |
`box(`*expr*`,`*expr*`,`*expr*`)` | `ceil(`*expr*`)` |
`cos(`*expr*`)` | `cosh(`*expr*`)` | `exp(`*expr*`)` |
`fabs(`*expr*`)` | `floor(`*expr*`)` | `hs(`*expr*`)` |
`log(`*expr*`)` | `obox(`*expr*`,`*expr*`,`*expr*`)` |
`random(`*expr*`)` | `sin(`*expr*`)` | `sinh(`*expr*`)` |
`sqrt(`*expr*`)` | `tan(`*expr*`)` | `tanh(`*expr*`)`.
number ::= *num_int* | *num_double* | `PI`.

Ein Strömungsproblem setzt sich aus den Optionen für den Strömungslöser und der Geometriedefinition zusammen. Letztere umfaßt Punkte, Oberflächen und Volumen.

main ::= *opts* { *point* | *surface* | *volume*}.

Die Definition der Optionen gestattet Erweiterungen in Form von anderen Lösungsverfahren. Im Moment ist nur das Druckkorrekturverfahren zur Behandlung instationärer Strömungen implementiert. Sind Werte nicht angegeben, so werden Default-Werte verwendet, die dem Quellcode zu entnehmen sind.

opts ::= `options nonstat {` *opts_nonstat* `}`.

opts_nonstat ::= {

re expr ; |	Reynolds-Zahl
alpha expr ; |	Blending CDS/Upwind$\in [0,1]$
tMax expr ; |	Maximale Simulationszeit
deltMax expr ; |	Maximales Δt
convDeltFactor expr ; |	Faktor für (2.35)
diffDeltFactor expr ; |	Faktor für (2.36)
writeInterval expr ; |	Zeitintervall zum Abspeichern
outfile string ; |	Name der Binärdatei
initfile string ; |	Läd V, p aus anderer Binärdatei
psolver `{` *opts_solver* `}` |	Optionen für Gleichungslöser
chem expr ;	Stofftransport mit
}.	Diffusionskonstante η

Die Optionen für lineare Gleichungssystemlöser umfassen die Art des Verfahrens und die Haltekriterien (vgl. Kapitel 2.2)

opts_solver ::= {
eps expr ; \|
delta expr ; \|
maxIter num_int ; \|
method (`GMRES(` *num_int* `)` \| `BiCGStab[(` *num_int* `)];`
}.

Punkte werden durch drei Koordinaten im Raum angegeben und durch eine eindeutige Nummer identifiziert. Die angegebenen Koordinaten dienen nur der Orientierung und haben für die Geometriebeschreibung keine weitere Bedeutung. Die tatsächlichen Koordinaten der Eckpunkte der Volumen werden den Flächendefinitionen entnommen.

point ::= `point` *num_int* `{ coord` *expr*,*expr*,*expr* `;}`.

Die Definition der begrenzenden Flächen, wieder identifiziert durch eine eindeutige Nummer, erfolgt mit *surface* und umfaßt Randbedingungen und Parametrisierung. Mit `tBC` werden tangentiale Randbedingungen gesetzt, wobei die normale Komponente des angegebenen dreidimensionalen Vektors ignoriert wird. Normale Randbedingungen werden mit `nBC` definiert, wobei hier neben der numerischen Angabe in der *string*-Position auch ein Ausdruck ähnlich *expr* für die Dirichlet-Randbedingung angegeben werden kann, der aber zusätzlich die Variablen x, y, z enthält, für die bei der Auswertung der Randbedingung in einem Punkt die kartesischen Koordinaten dieses Punktes

eingesetzt werden. Für die periodische Randbedingung ist die Nummer der zu verbindenden Fläche anzugeben. In ähnlicher Weise werden mit `cBC` Randbedingungen für den Stofftransport gesetzt.

Die Parametrisierung der Fläche kann auf drei verschiedene Weisen definiert werden. Zunächst können mit der ersten `initX`-Regel drei Ausdrücke ähnlich *expr* mit zusätzlichen Variablen x, y die drei kartesischen Koordinaten der Fläche angegeben werden, wobei $(x, y) \in [0, 1]^2$ ein regelmäßiges Gitter durchläuft. Mit *nurbssurface* kann eine NURBS-Fläche angegeben werden (s.u.) und mit `TFI` werden die Koordinaten mittels transfiniter Interpolation ermittelt, wobei alle umgebenden Flächen schon definiert sein müssen. Durch `param` kann das regelmäßige Gitter über $[0, 1]^2$ durch Angabe zweier Ausdrücke in x, y verändert werden. Schließlich können die so definierten Punkte mit `transformation` transformiert werden, wobei die angegebenen Transformationen einfach hintereinander ausgeführt werden.

```
surface ::= surface num_int {
            [ copyFrom num_int ; ]
            {
            tBC ( NEUMANN_ZERO | DIRICHLET expr,expr,expr ) ; |
            nBC ( NEUMANN_ZERO|DIRICHLET (expr|string)|IOBC1|PERIODIC num_int) ; |
            cBC ( NEUMANN_ZERO | DIRICHLET expr ) ; |
            initX string,string,string; |
            initX nurbssurface ; |
            initX TFI ; |
            param string,string; |
            transformation {{
                rotX expr ; | rotY expr ; | rotZ expr ; |
                translate expr,expr,expr; }}
            }
            }.
```

Zur Definition einer NURBS-Fläche (vgl. Kapitel 2.4) muß zunächst deren Grad mit `degree` in beiden Richtungen angegeben werden, sei dieser p, q. Dann werden in jeder Richtung n bzw. m Kontrollpunkte in Form einer $m \times n$-Matrix vorgegeben, wobei jeder Eintrag ein vierdimensionaler Vektor ist (3 Raumkoordinaten + Gewicht). Schließlich sind Knotenvektoren mit $n + p + 1$ bzw. $m + q + 1$ Knotenpunkten anzugeben.

```
nurbssurface ::= NurbsSurface {
                 degree num_int,num_int;
                 controlPoints num_int,num_int {
                 { expr, } expr
                 }
                 knotU { expr, } expr;
                 knotV { expr, } expr;
                 }.
```

Ein Volumen, wieder identifiziert durch eine eindeutige Nummer, wird durch 6 Seitenflächen, die in der Reihenfolge IM,IP,JM,JP,KM,KP durch ihre Nummer angegeben

werden, begrenzt. Weiterhin sind die Eckpunkte in der Reihenfolge wie in Abbildung 2.3 angegeben. Sie dienen dazu, die Flächen unterschiedlicher Volumen, welche 4 Punkte gemeinsam haben, miteinander zu verkleben. Weiterhin können die inneren Punkte ähnlich wie bei den Flächen entweder direkt oder durch transfinite Interpolation definiert werden. Auch Startwerte für $\boldsymbol{u}$ und die Stoffkonzentration können so angegeben werden. Mit der Definition eines `smoother` kann schließlich die in Kapitel 2.3.2 beschriebene Methode zur Glättung des Gitters angewendet werden.

volume ::= `volume` *num_int* `{`
`{`
surfaces *num_int*,*num_int*,*num_int*,
num_int,*num_int*,*num_int*; |
points *num_int*,*num_int*,*num_int*,*num_int*,
num_int,*num_int*,*num_int*,*num_int*; |
dim *num_int*,*num_int*,*num_int*; |
initX `expr` *string*,*string*,*string*; |
initU `expr` *string*,*string*,*string*; |
initC `expr` *string*; |
initX `TFI`; |
smoother `{` *smoother* `}`
`}`
`}`.

Die Parameter für die in Kapitel 2.3.2 beschriebene Methode zur Glättung der Gitter können hier eingestellt werden. Ihre Bedeutung ist dem genannten Kapitel zu entnehmen.

smoother ::= `{`
`iter` *num_int*; |
`alpha` *string*,*string*,*string*; |
`beta` *string*,*string*,*string*; |
`}`

Literaturverzeichnis

[BBC+94] Richard Barrett, Mike Berry, Tony Chan, Jim Demmel, June Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Chuck Romine, and Henk van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1994.

[BMC88] D. Bradley, M. Missaghi, and S. Chin. A taylor series approach to numerical accuracy and a third-order scheme for strong convective flows, 1988.

[BP98] O. Botella and R. Peyret. Benchmark spectral results on the lid-driven cavity flow. *Computers & Fluids*, 27(4):421–433, 1998.

[BSMM95] Ilja N. Bronstein, Konstantin A. Semendjajew, Gerhard Musiol, and Heiner Mühlig. *Taschenbuch der Mathematik*. Verlag Harri Deutsch, Thun, Frankfurt am Main, 2. edition, 1995.

[Fle90] C. Fletcher. Computational techniques for fluid dynamics, 1990.

[Fle91] Clive A. J. Fletcher. *Computational Techniques for Fluid Dynamics, Volume I: Fundamental and General Techniques*. Springer-Verlag, Berlin, 2nd edition, 1991.

[GGS82] U. Ghia, K. N. Ghia, and C. T. Shin. High–Re solutions for incompressible Navier–Stokes equations and a multi–grid method. *J. Comput. Phys.*, 48:387–411, 1982.

[GHS93] M. GUNZBURGER, L. HOU, and T. SVOBOTNY. Optimal control and optimization of viscous, 1993.

[GK00] M. Griebel and F. Koster. Adaptive wavelet solvers for the unsteady incompressible Navier Stokes equations. In J. Malek, J. Necas, and M. Rokyta, editors, *Advances in Mathematical Fluid Mechanics*, Lecture Notes of the Sixth International School "Mathematical Theory in Fluid Mechanics", Paseky, Czech Republic, September 1999. Springer Verlag, 2000. also as Report SFB 256 No. 669, Institut für Angewandte Mathematik, Universität Bonn, 2000.

[GPP94] R. Glowinski, T. W. Pan, and J. Périaux. A fictitious domain method for external incompressible viscous flow modeled by Navier–Stokes equations. *Comp. Meth. Appl. Mech. Engng.*, 112:133–148, 1994.

[GT93] Christian Grossmann and Johannes Terno. *Numerik der Optimierung*. B.G. Teubner, Stuttgart, 1993.

[HK98] M. Hinze and K. Kunisch. On suboptimal control strategies for the navier-stokes equations, 1998.

[HW65] F.H. Harlow and J.E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface. *The Physics of Fluids*, 8:2182–2189, 1965.

[IK73] Eugene Isaacson and Herbert Bishop Keller. *Analyse numerischer Verfahren*. Verlag Harri Deutsch, Zürich und Frankfurt am Main, 1973.

[KG00] F. Koster and M. Griebel. Efficient preconditioning of linear systems for the finite difference and the collocation method on sparse grids, 2000. in preparation.

[Kön93] Konrad Königsberger. *Analysis 2*. Springer Verlag Berlin Heidelberg, 1993.

[Lav99] Philippe Lavoie. Nurbs++ 3.0 documentation. 1999.

[Len89] Richard Lenk. *Fachlexikon ABC Physik*. Verlag Harri Deutsch, 2. edition, 1989.

[Leo79] B. P. Leonard. A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Computer Methods in Applied Mechanics and Engineering*, 19:59–98, 1979.

[Lis99] V. D. Liseikin. *Grid Generation Methods*. Scientific Computation. Springer-Verlag, Berlin, 1999.

[Mes95] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard*. University of Tennessee, Knoxville, TN, June 1995.

[NB] Peter Niederdrenk and Olaf Brodersen. Elliptic grid control on a discrete level.

[Per85] M. Perić. *Finite volume method for the prediction of three-dimensional fluid flow in complex ducts*. PhD thesis, Imperial College, London, 1985.

[PLS98] Terence John Parr, John Lilley, and Scott Stanchfield. *ANTLR 2.xx Reference Manual*. 1998. ANTLR [PQ95] is a redesign and rewrite of PCCTS [PQ96] in Java. ANTLR generates parsers in C++ and Java. This book is still in preparation, and the title has not been settled.

[PQ95] Terence J. Parr and Russell W. Quong. ANTLR: A predicated-LL(k) parser generator. *Software - Practice and Experience*, 25(7):789–810, July 1995.

[PQ96] Terence J. Parr and Russell W. Quong. LL and LR translators need $k > 1$ lookahead. *ACM SIGPLAN Notices*, 31(2):27–34, February 1996.

[PT83] Roger Peyret and Thomas D. Taylor. *Computational Methods for Fluid Flow*. Springer-Verlag, New York, 1983.

[PT95] Les Piegl and Wayne Tiller. *The NURBS book*. Springer-Verlag, Berlin Heidelberg, 1995.

[Sch97] V. Schulz. Solving discretized optimization problems by partially reduced sqp methods, 1997.

[SF93] G. Sleijpen and D. Fokkema. Bicgstab(l) for linear equations involving unsymmetric matrices with complex spectrum, 1993.

[SML97] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit: An Object-Oriented Approach To 3D Graphics*. Prentice Hall, 1997.

[Spe93a] P. Spellucci. A new technique for inconsistent qp-problems in the sqp-method, 1993.

[Spe93b] Peter Spellucci. *Numerische Verfahren der nichtlinearen Optimierung*. Birkhäuser Verlag, Berlin, 1993. Internationale Schriftenreihe zur numerischen Mathematik.

[Spe95] P. Spellucci. donlp2 users guide, 1995.

[Spe98] P. Spellucci. An SQP method for general nonlinear programs using only equality constrained subproblems. *Mathematical Programming*, 82:413–448, 1998.

[Tur99] Stefan Turek. *Efficient solvers for incompressible flow problems / an algorithmic and computational approach*, volume 6 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag Inc., New York, NY, USA, 1999. Includes CD-ROM.

[TWM85] J. F. Thompson, Z. U. A. Warsi, and C. W. Mastin. *Numerical Grid Generation*. Elsevier Science Publishing, Amsterdam, 1985.

[van86] J. van Kan. A second-order-accurate pressure-correction scheme for viscous incompressible flow. *SIAM Journal on Scientific and Statistical Computing*, 7(3):870–891, July 1986.

[Wet98] Brian R. Wetton. Error analysis of pressure increment schemes, April 1998.

[WSKB97] P. Wesseling, A. Segal, C.G.M. Kassels, and H. Bijl. Computing flows on general three-dimensional nonsmooth staggered grids. Technical Report 97-23, Delft University of Technology, 1997.

[Zij96] M. Zijlema. *Computational modeling of turbulent flow in general domains*. PhD thesis, Delft University of Technology, April 1996.

Hiermit versichere ich, daß ich die vorliegende Diplomarbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet sowie Zitate kenntlich gemacht habe.