Institut für Maritime Systeme und Strömungstechnik
Fakultät für Ingenieurwissenschaften
Universität Rostock

Diplomarbeit

Entwicklung eines Panelgenerierungsverfahrens mit automatischer Panelgrößenanpassung

cand.-ing. Michael Zimmermann

Juli 2003

Selbständigkeitserklärung

Hiermit erkläre ich, Michael Zimmermann, die vorliegende Arbeit selbständig nach bestem Wissen und mit den vorliegenden Informationen erarbeitet zu haben. Alle Quellen und Hilfsmittel sind ordnungsgemäß angegeben.

Rostock im Juli 2003

Michael Zimmermann

Aufgabenstellung

this page left blank intentionally!

this page left blank intentionally!

Inhaltsverzeichnis

1 Einleitung **1**

2 Grundlagen - Hydrodynamik **3**

2.1 Problemstellung Wellenwiderstandsberechnung 3

2.2 Modellierung 4

2.3 Wellenwiderstandsberechnung mit CFD 5

2.3.1 Entwicklung eines CFD-Codes 6

2.3.2 Durchführung von CFD-Berechnungen 7

2.4 Physikalisches Modell 9

2.4.1 Koordinatensystem 9

2.4.2 Grundgleichungen der Hydrodynamik 10

2.4.3 Potentialtheorie 10

2.4.4 Einfache Elemente der Potentialtheorie 12

2.4.5 Randbedingungen 13

2.5 Mathematisches Modell 15

2.5.1 Elemente für Panelverfahren 16

2.5.2 Beispiel für ein Panel-Problem 17

2.6 Numerisches Modell 18

3 Grundlagen - Kurven und Flächen **19**

3.1 Möglichkeiten zur mathematischen Definition 19

3.1.1 Explizite Form 19

3.1.2 Implizite Form 20

3.1.3 Parametrische Definition 20

3.1.4 Rationale Darstellung 21

3.2 Anforderungen an CAG-Objekte 21

3.3 Approximation durch Polynome 21
3.4 Spline-Kurven 22
3.4.1 B-Spline-Funktionen 22
3.4.2 B-Spline-Kurven 22
3.5 NURBS-Kurven 23
3.6 NURBS-Flächen 23

4 Netzgenerierung 24
4.1 Begriffsdefinitionen 24
4.2 Anforderungen 25
4.2.1 Allgemeine Anforderungen 25
4.2.2 Oberflächennetze 25
4.3 Einordnung von Gittern 26
4.3.1 Linie, Ebene, Fläche, Volumen 27
4.3.2 Strukturierte und unstrukturierte Gitter 27
4.4 Generierung unstrukturierter Netze 28
4.4.1 Warum unstrukturierte Gitter? 29
4.4.2 Delaunay-Triangulisierung 29
4.4.3 Triangulisierung nach der *advancing front*-Methode 30
4.4.4 *quad/octree*-basierte Verfahren 30
4.4.5 Bewertung 31
4.5 Netzoptimierung 33
4.5.1 Knotenverschiebung 33
4.5.2 Variation der Elementanzahl 34

5 Der Vernetzungsalgorithmus 36
5.1 Zielsetzung und Problemdefinition 36
5.1.1 Numerische Stabilität und Robustheit 37
5.1.2 Topologie 37
5.2 Betrachtete Verfahren 38
5.2.1 *top down*-Verfahren 38
5.2.2 *bottom up*-Verfahren 38
5.2.3 Bewertung 38
5.3 Der Vernetzungsalgorithmus 39

5.3.1 Überblick 39
5.3.2 Erzeugung der Ausgangsvernetzung 40
5.3.3 Netzoptimierung 44
5.3.4 Netzadaption an die Wasserlinienkontur 51

6 Implementierung **53**
6.1 Programmiersprache und -technik 53
6.2 Code- und Datenstrukturen 54
6.2.1 Paketstruktur 54
6.2.2 Geometriebibliothek 55
6.2.3 Speicherung von Geometrieinformationen 55
6.3 Allgemeiner Programmablauf 56

7 Ergebnisse **59**
7.1 Schiffsbeschreibungen 59
7.2 Anfangsvernetzung 61
7.3 Merging 62
7.4 Netzvergröberung 64
7.5 Wasserlinie 67

8 Fazit und Ausblick **68**

Literaturverzeichnis **69**

A *Merging* **72**
A.1 Verfahren zur Topologieerkennung 72
A.1.1 *bounding box*-Verfahren 72
A.1.2 *scanline*-Verfahren 72
A.2 *Merging* 72

B CD-Inhalt **73**

C Dateiformate **74**
C.1 Wasserlinienkontur 74
C.2 Lokaler Einfluss 74
C.2.1 Lagebasiert 74

C.2.2 Quellenverteilung 75
C.3 Gitter 75
C.3.1 OFF 75
C.3.2 Shallo 76
C.3.3 Autowing 76

D The GNU General Public License 77

Abbildungsverzeichnis

Grundlagen - Hydrodynamik

2.1 Modellierung des Verfahrens zur Lösung eines ingenieurwissenschaftlichen Problems 4
2.2 Entwicklung eines Algorithmus zur Wellenwiderstandsberechnung 6
2.3 Durchführung einer CFD-Berechnung 8
2.4 Koordinatensystem und Gebietsdefinition 9
2.5 Diskretisierung eines Zylinders für potentialtheoretische Berechnung 17

Grundlagen - Kurven und Flächen

3.1 Schraubenkurve als Beispiel einer Parameterkurve in R^3 20

Netzgenerierung

4.1 Beispiele der Vernetzung von Linien, Flächen und Volumina 27
4.2 Beispiele für stukturierte und unstrukturierte Netze 27
4.3 Delaunay-Triangulisierung 30
4.4 Iterativer Prozess der Netzgenerierung mittels Delaunay-Triangulisierung . 31
4.5 Die *advancing front*-Methode 32
4.6 Vernetzung mittels *Quadtrees* 33
4.7 Reduktion der Elementanzahl durch Entfernen eines Punktes 34
4.8 Reduktion der Elementanzahl durch Kollabieren einer Kante 35
4.9 Reduktion der Elementanzahl durch Kollabieren eines Dreiecks 35

Der Vernetzungsalgorithmus

5.1 Nachbarschaftsrelationen von NUBRS-Patches 37
5.2 Der Vernetzungsalgorithmus im Überblick 39
5.3 Erzeugung der Ausgangsvernetzung 41
5.4 Methodik der *butterfly subdivision* 42
5.5 Verfahren zur Netzoptimierung durch Reduktion der Elementanzahl 45

5.6 Fallunterscheidung für den *edge collapse* 46
5.7 Abmessungen der an eine Kante angrenzenden Elemente 48

Implementierung

6.1 Allgemeiner Programmablauf 57

Ergebnisse

7.1 Korvette *DTMB 5415* 59
7.2 RoPAX-Schiff 60
7.3 Frachtschiff Mercator 60
7.4 Strukturierte Vernetzung eines Patches 61
7.5 Adaptive Netzverfeinerung 62
7.6 Zusammenführen der Dreiecksnetze für *DTMB 5415* 63
7.7 Fehler durch das *merging* im Bereich des Sonardoms 63
7.8 *Merging* am Beispiel der RoPAX-Fähre 64
7.9 Netzvergröberung mit den Qualitätskriterien q_{AR} und q_A 65
7.10 Netzvergröberung mit den Qualitätskriterien q_{AR} und q_A sowie k_K 65
7.11 Netzvergröberung mit lokaler Elementgrößenkontrolle 66
7.12 Einflussfunktion $f_x(x)$ 66
7.13 Adaption der Vernetzung an die Wasserlinienkontur 67

Kapitel 1

Einleitung

In der Schiffbauindustrie ist, wie auch in vielen anderen Bereichen der Wirtschaft, in den letzten Jahren ein steigender Wettbewerb zu verzeichnen. Aufträge werden oft an die Unternehmen vergeben, die eine äußerst kurze „time to market“, also die Ablieferung eines Schiffes innerhalb eines sehr kurzen Zeitraums, bieten können. Durch die Auftraggeber werden noch vor der Auftragsvergabe zunehmend umfangreichere Informationen über die Leistungen des projektierten Schiffes verlangt.

Um diese Forderungen zu erfüllen und hierdurch sich im Wettbewerb durchzusetzen, sind die Unternehmen gefordert, neben einer guten betriebsinternen Organisation der Arbeitsabläufe die Möglichkeiten der modernen rechnergestützten Konstruktion und Fertigung auszunutzen. Hierdurch läßt sich insbesondere in der Anfangsphase eines Projektes die Effektivität steigern sowie die Wahrscheinlichkeit von später erkennbar werdenden Entwicklungsfehlern minimieren.

So ist es für die Projektphase eines Entwurfs im Bereich der Hydrodynamik vielfach üblich, umfangreiche Informationen zu Schiffswiderstand, Wellenbild wie auch Antriebsleistung zu liefern. Eine Abschätzung der Seegangseigenschaften und der Steuerbarkeit wird ebenfalls erarbeitet. Dies ermöglicht es dem Auftraggeber, noch vor der Signierung des Bauvertrages die Güte der unterbreiteten Angebote zu analysieren und entsprechend zu entscheiden.

Im Bereich der Widerstandsprognose werden für die erforderliche Optimierung der Schiffsform wie auch für die Bestimmung der oben genannten relevanten, die Leistungsfähigkeit eines Schiffes charakterisierenden Parameter vielfach numerische Verfahren eingesetzt. Diese ermöglichen es, kostengünstig und innerhalb eines kurzen Zeitraumes Varianten zu untersuchen und hierdurch gezielt globale Formparameter wie auch lokale Details zu optimieren.

Grundlage jedes numerischen Prognoseverfahrens ist die Darstellung der Schiffsgeometrie mit Hilfe von Dreiecks- oder Viereckselementen, den sog. Panels. Durch diese Approximation wird die komplexe Schiffsform mathematisch beschreibbar; eine numerische Lösung der die Vorgänge bei der Umströmung eines Schiffs beschreibenden Gleichungen wird möglich.

Insofern ist die Erzeugung von Oberflächenbeschreibungen in Form von sog. Netzen ein

entscheidender Bestandteil des Berechnungsprozesses zur Ermittlung des Widerstandes eines Schiffes in Fahrt. Die Qualität der Netze hat hierbei entscheidenden Einfluss auf die Güte der Berechnungsergebnisse. Vielfach wird daher ein Großteil der Bearbeitungszeit für die Durchführung einer Widerstandsberechnung auf die Erzeugung von qualitativ hochwertigen Netzen verwendet. Wird das Verfahren zur Erzeugung von Netzen vereinfacht und beschleunigt, so ergibt sich ein großes Einsparpotential.

Im Rahmen dieser Diplomarbeit werden Möglichkeiten zur weitgehend automatischen Erzeugung von qualitativ hochwertigen Netzen auf der Schiffsaußenhaut entwickelt und getestet, um so das gewünschte Ziel einer Reduktion der erforderlichen Arbeitszeit für die Gittergenerierung zu erreichen. Besonderer Wert wurde darauf gelegt, eine automatische Anpassung der Netzstruktur an die Oberflächeneigenschaften zu ermöglichen.

Aufbauend auf den in Kapitel 2 beschriebenen Grundlagen der Hydrodynamik und numerischen Widerstandsberechnung sowie der computergestützten Formdefinition, siehe Kapitel 3, befasst sich Kapitel 4 mit Methoden zur Netzgenerierung.

Der durch den Autor entwickelte Algorithmus zur Gittergenerierung wird in Kapitel 5 vorgestellt. Details der Programmimplementierung sind in Kapitel 6 zu finden. In Kapitel 7 werden Ergebnisse dargelegt.

Abschließend gibt Kapitel 8 eine Zusammenfassung. Aspekte, die weiterer Untersuchungen bedürfen, werden benannt.

Kapitel 2

Grundlagen - Hydrodynamik

Insbesondere durch die Entwicklung leistungsstarker Computer wurden bei der numerischen Bestimmung der Umströmung von Schiffen deutliche Fortschritte erzielt. In diesem Kapitel wird eine grundlegende Einführung in die der Strömungsberechnung zugrunde liegende Theorie der Hydrodynamik gegeben.

In Abschnitt 2.1 wird Funktion und Rolle der Wellenwiderstandsberechnung im schiffbaulichen Entwurfsprozess näher dargelegt. Die in Abschnitt 2.2 beschriebene Methodik der ingenieurwissenschaftlichen Problemlösung wird im folgenden Abschnitt auf die Problemstellung Wellenwiderstandsberechnung angewandt. In den Abschnitten 2.4 bis 2.6 werden die einzelnen Aspekte bei der Lösung dieses Problems näher beschrieben.

2.1 Problemstellung Wellenwiderstandsberechnung

Der Wellenwiderstand eines Schiffes in Fahrt beeinflusst u. a. die erforderliche Antriebsleistung erheblich. Daher werden vielfach insbesondere in der frühen Entwurfsphase Verfahren eingesetzt, welche die Abschätzung dieses Widerstandsanteils und somit eine Verbesserung der Ökonomie im Schiffsbetrieb durch Optimierungsmaßnahmen mit nur geringem Zeitaufwand ermöglichen.

Hierfür sind Verfahren erforderlich, welche eine Berechnung oder Abschätzung der beiden wichtigsten Widerstandsanteile, nämlich Wellenwiderstand und Reibungswiderstand, ermöglichen. Da in den vergangenen Jahrzehnten eine stetige Zunahme der Schiffsgeschwindigkeiten[1] zu verzeichnen ist, kommt der Ermittlung und Optimierung des Wellenwiderstands eine zunehmend gewichtigere Rolle zu. Dementsprechend wurden und werden Methoden entwickelt, die zu einer steigenden Genauigkeit der Berechnungsergebnisse führen.

[1] Bei höheren Schiffsgeschwindigkeiten steigt der Anteil des Wellenwiderstandes am Gesamtwiderstand, während der Anteil des Reibungswiderstandes abnimmt.

2.2 Modellierung

Für die Modellierung eines Problems in den Ingenieurwissenschaften, aber auch in anderen Bereichen, hat sich vielfach ein mehrstufiges Vorgehen bewährt. Dieses ist in Abb. 2.1 dargestellt.

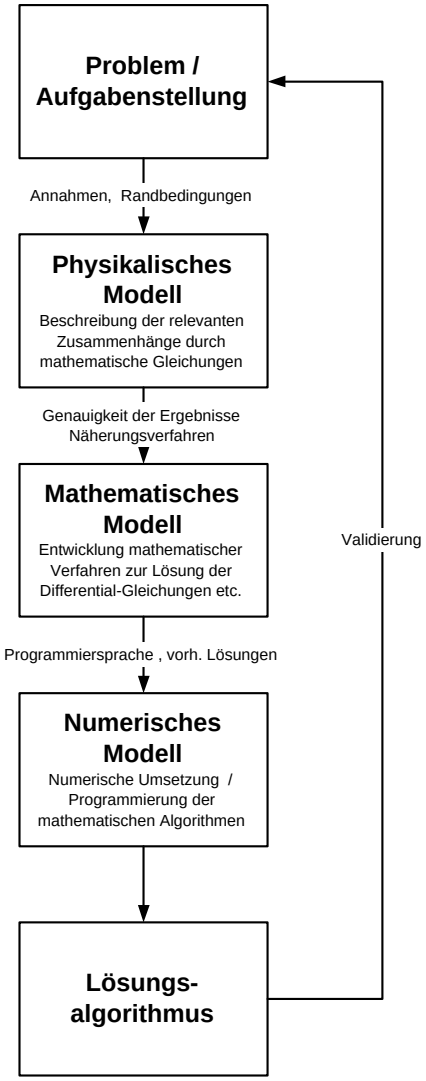

Abb. 2.1: Modellierung des Verfahrens zur Lösung eines ingenieurwissenschaftlichen Problems

Physikalisches Modell Ausgehend von der zu analysierenden Problematik werden in einem ersten Schritt alle relevanten Zusammenhänge und Effekte bestimmt. Die Güte der letztlich erzielten Ergebnisse hängt hierbei vom Umfang bzw. der Anzahl der berücksichtigten Effekte ab. Daher ist bei diesem Schritt darauf zu achten, dass, abhängig von der

angestrebten Genauigkeit, die für diese Problemstellung entscheidenden Parameter untersucht werden. Es sollte vermieden werden, dass nicht relevante bzw. die Ergebnisse nur sehr gering beeinflussende Effekte ebenfalls mit einbezogen werden, da hierdurch die Komplexität des Problems wie auch die Fehlermöglichkeiten anwachsen sowie der numerische Aufwand für die Lösung stark ansteigt.

Für die identifizierten und als relevant eingestuften Effekte wird in einem zweiten Teilschritt das Problem mit Hilfe von mathematischen Gleichungen beschrieben; die erforderlichen Anfangs- oder Randbedingungen werden erarbeitet. Sollten für einige Aspekte keine ausreichenden theoretischen Erkenntnisse verfügbar sein, sind geeignete Ersatzmodelle zu wählen. Auch hier wiederum ist, so mehrere mögliche Repräsentationen existieren, eine Entscheidung für eine geeignete Umsetzung zu treffen. Exaktheit, Anwendbarkeit sowie numerischer Aufwand sind Gründe, die diese Auswahl beeinflussen. Weiterhin sind die Zusammenhänge zwischen den einzelnen Effekten zu bestimmen; ein Gesamtsystem ist zu erarbeiten, welches die realen Zustände geeignet widerspiegelt.

Mathematisches Modell Ausgehend von diesem physikalischen Modell werden Verfahren erarbeitet, die eine numerische Lösung ermöglichen. Hierfür wird ein mathematisches Modell entwickelt, durch welches z. B. die gegebenen Differentialgleichungssysteme gelöst werden.

Numerisches Modell Im Rahmen des numerischen Modells wird die Umsetzung der erarbeiteten Verfahrensweise zu einem Programm durchgeführt. Dieser Schritt ist stark abhängig von den zur Verfügung stehenden Möglichkeiten wie z. B. spezieller mathematischer Software, Programmiersprachen oder Bibliotheken. Die bevorzugte Lösung wird weiterhin durch die Fähigkeiten des Bearbeitenden wie auch durch eventuell bereits vorhandene Teillösungen bestimmt.

Validierung Nach der vollständigen Erarbeitung des Verfahrens wird letztlich die Funktionsfähigkeit dieser Lösung mit Hilfe von geeigneten Testfällen validiert. Hierzu wird im Bereich der Hydrodynamik vielfach auf verfügbare Ergebnisse von experimentellen wie auch numerischen Untersuchungen an Beispielschiffen zurückgegriffen[2].

2.3 Wellenwiderstandsberechnung mit CFD

CFD-Methoden[3] werden vielfach zur Bestimmung des Wellenwiderstands eingesetzt und haben dementsprechend eine wichtige Rolle im Entwicklungsprozess. Die grundlegenden Schritte bei der Entwicklung einer Software[4] zur Durchführung derartiger Berechnungen

[2]Bekannt sind insbesondere das Wigley-Schiff sowie Schiffe der Serie 60.

[3]Das Akronym CFD steht für *computational fluid dynamics*.

[4]Im folgenden wird für Software zur Berechnung des Wellenwiderstandes eines Schiffes die Bezeichnung CFD-Code verwendet.

werden im folgenden Abschnitt erläutert. Hierbei wird entsprechend der in Abb. 2.1 dargestellten Methodik zur ingenieurwissenschaftlichen Problemlösung vorgegangen. Abschnitt 2.3.2 beschreibt schematisch die Anwendung von CFD-Berechnungs-Software.

2.3.1 Entwicklung eines CFD-Codes

Die Entwicklung eines CFD-Codes zur Berechnung des Wellenwiderstandes kann in drei Schritte unterteilt beschrieben werden. Dies ist in Abb. 2.2 dargestellt.

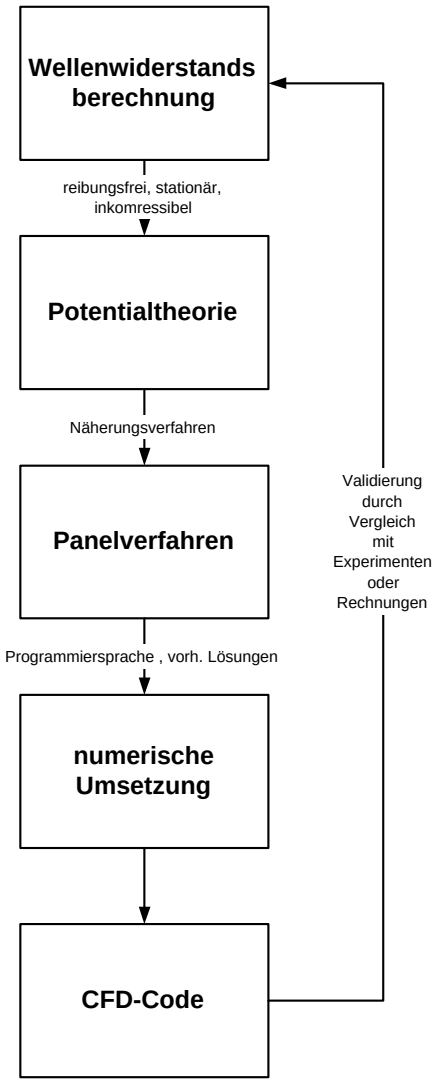

Abb. 2.2: Entwicklung eines Algorithmus zur Wellenwiderstandsberechnung

Physikalisches Modell Für die Umströmung eines Schiffes sind die Navier-Stokes-Gleichungen[5] gültig. Dieses Differentialgleichungssystem beschreibt allgemeingültig die Strömungsverhältnisse um einen Körper. Da ein Schiff in Wasser operiert, es also von einem nahezu inkompressiblen Medium umströmt wird, kann die Dichte als konstant angenommen werden. Der Wellenwiderstand wird durch reibungsbedingte Effekte[6] nur indirekt beeinflußt. Insofern ist es möglich, den Wellenwiderstand unter Vernachlässigung viskoser Effekte zu bestimmen. Wird weiterhin ein stationärer Zustand angenommen, so kann das Problem der Wellenwiderstandsberechnung durch die sog. Potentialtheorie beschrieben werden[7].

Mathematisches Modell Eine analytische Lösung der durch die Potentialtheorie für eine Problemstellung gültigen Gleichungssysteme ist nur für einfache Probleme möglich; die Umströmung eines Schiffsrumpfes kann, basierend auf der Potentialtheorie, nur durch Näherungsverfahren bestimmt werden. Hierfür wird durch die Methodik der sog. Panelverfahren[8] eine Ortsdiskretisierung durchgeführt; die potentialtheoretisch basierten Differentialgleichungen werden in numerisch lösbare algebraische Gleichungssysteme umgewandelt. Entscheidend für die erzielbare Genauigkeit der CFD-Rechnung ist hierbei, dass eine adäquate Formulierung für die Interaktion zwischen einzelnen ortsdiskretisierten Elementen gefunden wird.

Numerisches Modell Bei der abschließenden Entwicklung des numerischen Modells ist eine geeignete Umsetzung des mathematischen Modells zu einem Programm zu finden. Genauigkeit, Effizienz, Speicherplatz, Parallelisierbarkeit wie auch die Programmstruktur sind Aspekte, die hierbei besonderer Beachtung bedürfen.

2.3.2 Durchführung von CFD-Berechnungen

Besteht die Notwendigkeit, CFD-Berechnungen durchzuführen, so wird vielfach auf kommerziell verfügbare CFD-Codes zurückgegriffen; eine zeit- und kostenintensive Eigenentwicklung sowie die anschließende Validierung entfallen. Abb. 2.3 stellt aus Sicht des Nutzers die Anwendung eines CFD-Codes für die Wellenwiderstandsberechnung dar. Durch den Anwender erforderliche Interaktion ist rot dargestellt.

Ausgehend von einer meist durch moderne, auf den Schiffbau ausgerichtete CAD-Systeme bereitgestellten Geometriebeschreibung des Schiffsrumpfes wird nach einer eventuell durchzuführenden Geometrieaufbereitung die Netzgenerierung auf der Rumpf- wie auch Wasseroberfläche durchgeführt. Falls das Verhalten des Schiffes in begrenzten Fahrgewässern untersucht werden soll, ist ebenfalls eine Gittergenerierung für die Fahrbahnbegrenzungen durchzuführen.

[5] Benannt nach Claude Navier und Sir George Stokes
[6] Zu nennen sind hier beispielhaft Ablösungserscheinungen.
[7] Eine Einführung in die Potentialtheorie ist in Abschnitt 2.4 zu finden.
[8] Siehe hierzu auch Abschnitt 2.5

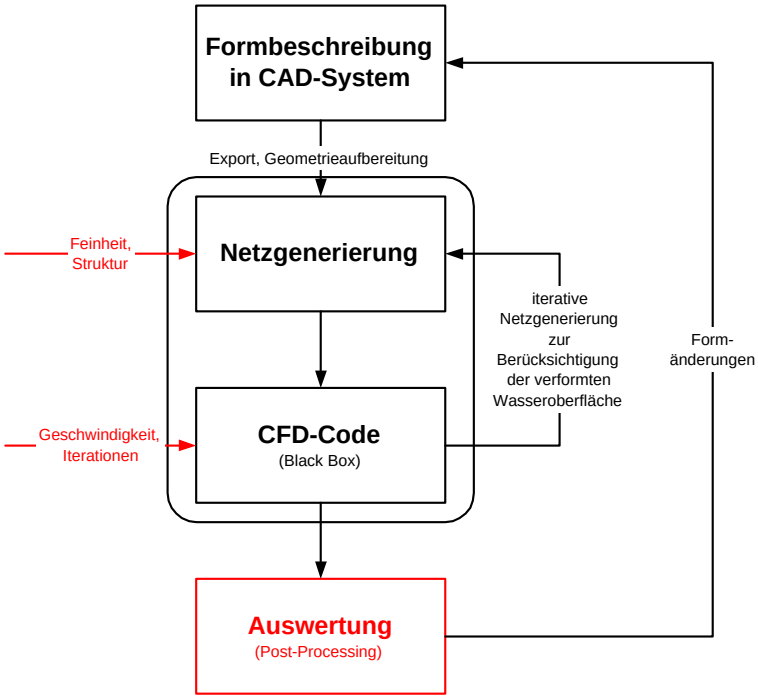

Abb. 2.3: Durchführung einer CFD-Berechnung

Nutzereingaben definieren die Feinheit und Struktur des Netzes, d. h. die Anzahl und Anordnung der zur Berechnung verwendeten Elemente und hierdurch die erzielbare Genauigkeit der Berechnung. Weiterhin hat der Anwender Sorge zu tragen, dass die Geometrie des Schiffes durch das Gitter ausreichend genau dargestellt wird. Ebenfalls zu prüfen ist die Qualität des Netzes[9], d. h. z. B. die Größenverhältnisse benachbarter Elemente etc., da diese ebenfalls gravierende Auswirkungen auf die Güte der Berechnungen haben kann.

Durch die Definition eines Netzes wird die mathematische näherungsweise Lösung der potentialtheoretischen Gleichungen festgelegt. Bei komplexen Geometrien ist insofern vielfach ein großer Aufwand zur Erstellung eines „guten“Netzes erforderlich. Erfahrung, aber auch leistungsfähige Netzgenerierungsprogramme sind hierfür unabdingbar.

In einem zweiten Schritt werden durch den Benutzer die Bedingungen für die Berechnung vorgegeben. So werden Schiffsgeschwindigkeit und Fluideigenschaften festgelegt.

Die Lösung erfolgt anschließend weitestgehend nicht interaktiv. In einem iterativen Prozess wird bei modernen CFD-Codes eine Lösung ermittelt. Da auf Grund der durch Wellenbildung ausgelenkten Wasseroberfläche eine Veränderung der Kontur der benetzten Außenhaut auftritt, ist es für jeden Iterationsschritt erforderlich, das Gitter auf der Außenhaut an die geänderten Bedingungen anzupassen. Dies erfolgt vielfach vollautomatisch. Weiterhin ist eine Anpassung der Gitterstruktur auf der Wasseroberfläche durchzuführen, um diese an die berechnete Wellenerhebung anzupassen.

[9] Anmerkungen zur Qualität von Gittern sind in Abschnitt 4.2 zu finden.

Nach Abschluß der Berechnungen werden die gewonnenen Ergebnisse ausgewertet. Es wird versucht, Mängel im Design durch Änderungen an der Formbeschreibung im CAD-System zu beseitigen oder zumindest deren Auswirkungen einzuschränken.

2.4 Physikalisches Modell

Das physikalische Modell beschreibt die für die Lösung der Aufgabenstellung erforderlichen Gegebenheiten eindeutig und bildet die Basis für die anschließende Entwicklung des mathematischen und des numerischen Modells. Im folgenden werden die Grundgleichungen der Hydrodynamik wie auch die für das hier untersuchte Problem gültigen Gleichungen dargestellt. Werden Vereinfachungen durchgeführt, so werden diese benannt. Weiterhin werden die für die Wellenwiderstandsberechnung maßgeblichen Randbedingungen beschrieben.

2.4.1 Koordinatensystem

Wie im Schiffbau allgemein üblich wird ein kartesisches Koordinatensystem verwendet. Der Ursprung ist im hinteren Lot auf Höhe der Unterkante des Kiels definiert. Die Schiffslängsachse ist x-Achse; die y-Achse verläuft orthogonal zur Mittschiffsebene. Dies ist in Abb. 2.4 dargestellt.

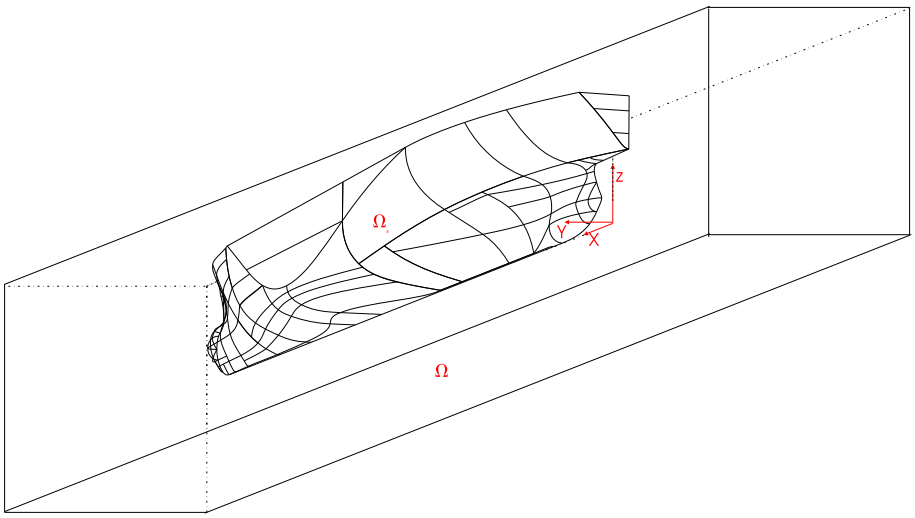

Abb. 2.4: Koordinatensystem und Gebietsdefinition am Beispiel eines Containerschiffes

Das für die Berechnungen berücksichtigte Flüssigkeitsgebiet wird mit Ω bezeichnet. Ω_S beschreibt die benetzte Oberfläche des Schiffsrumpfes. Diese variiert in Abhängigkeit von der Schiffsgeschwindigkeit, welche gleich der Anströmgeschwindigkeit U_∞ ist.

2.4.2 Grundgleichungen der Hydrodynamik

Die Kontinuitätsgleichung besagt, dass der Massenstrom über die Grenzen eines infinitesimal kleinen Kontrollvolumens dV gleich Null ist. Werden die Geschwindigkeitskomponenten in die Hauptkoordinatenachsen $\vec{x} = (x, y, z)^T$ mit $\vec{u} = (u, v, w)^T$ bezeichnet, so läßt sich dies wie folgt ausdrücken[10]:

$$\frac{\delta\left(\rho\vec{u}\right)}{\delta\vec{x}} + \nabla\left(\rho\vec{u}\right) \qquad \text{in} \qquad dV \tag{2.1}$$

Bei der Umströmung eines Schiffes treten nur geringe Drücke auf, so dass der Einfluss der Kompressibilität des umströmenden Mediums, d. h. von Wasser, vernachlässigbar gering ist. Insofern werden im folgenden nur inkompressible Strömungen betrachtet. Für die Dichte gilt $\rho = const$. Hierfür kann Gl. 2.1 überführt werden zu:

$$u + v + w = 0 \qquad \text{in} \qquad dV \tag{2.2}$$

Die Navier-Stokes-Gleichungen sind geeignet, die Umströmung eines Schiffes in seiner Gesamtheit exakt, d. h. drei-dimensional, instationär und für ein kompressibles Fluid, zu beschreiben. Diese basieren auf dem Impulserhalt der Strömung.

$$\begin{aligned}
\rho f_x - \frac{\delta p}{\delta x} + \mu\left(\frac{\delta^2 u}{\delta x^2} + \frac{\delta^2 u}{\delta y^2} + \frac{\delta^2 u}{\delta z^2}\right) &= \rho\frac{du}{dt} \\
\rho f_y - \frac{\delta p}{\delta y} + \mu\left(\frac{\delta^2 v}{\delta x^2} + \frac{\delta^2 v}{\delta y^2} + \frac{\delta^2 v}{\delta z^2}\right) &= \rho\frac{dv}{dt} \\
\rho f_z - \frac{\delta p}{\delta z} + \mu\left(\frac{\delta^2 w}{\delta x^2} + \frac{\delta^2 w}{\delta y^2} + \frac{\delta^2 w}{\delta z^2}\right) &= \rho\frac{dw}{dt}
\end{aligned} \tag{2.3}$$

Dabei beschreibt f_i eine Volumenkraft in den drei Koordinatenhauptachsen $i = x, y, z$, p den Druck, μ die Viskosität der Flüssigkeit und t die Zeit. Der Operator d/dt beschreibt Ableitungen nach der Zeit; ortsbezogene Ableitungen werden, mit i stellvertretend für die entsprechende Koordinatenrichtung, durch $\delta/\delta i$ kenntlich gemacht.

Werden Navier-Stokes-Gleichungen und die Kontinuitätsgleichung gekoppelt, so entsteht ein System nichtlinearer Differentialgleichungen, deren analytische Lösung für die komplexen Strömungsverhältnisse in der Umgebung eines Schiffsrumpfs nicht möglich ist.

2.4.3 Potentialtheorie

Werden die Reibung und somit auch Turbulenzeffekte vernachlässigt, so erhält man aus den Navier-Stokes-Gleichungen die sog. Euler-Gleichungen:

$$\begin{aligned}
\rho\left(\frac{du}{dt} + u\frac{\delta u}{\delta x} + v\frac{\delta u}{\delta y} + w\frac{\delta u}{\delta z}\right) &= \rho f_x - \frac{\delta p}{\delta x} \\
\rho\left(\frac{dv}{dt} + u\frac{\delta v}{\delta x} + v\frac{\delta v}{\delta y} + w\frac{\delta w}{\delta z}\right) &= \rho f_y - \frac{\delta p}{\delta y}
\end{aligned}$$

[10] Durch den Operator ∇ wird ausgedrückt: $\nabla = (\delta/\delta x, \delta/\delta y, \delta/\delta z)$

$$\rho\left(\frac{dw}{dt}+u\frac{\delta w}{\delta x}+v\frac{\delta w}{\delta y}+w\frac{\delta w}{\delta z}\right) = \rho f_z-\frac{\delta p}{\delta z} \tag{2.4}$$

Wird weiterhin eine drehungsfreie Strömung vorausgesetzt, d. h.

$$\nabla\times\vec{u}=0 \tag{2.5}$$

, so gelangt man zu den sog. Potentialströmungen.

Potentialströmungen zeichnen sich dadurch aus, dass die Komponenten des Geschwindigkeitsvektors nicht mehr unabhängig voneinander, sondern über das Potential ϕ gekoppelt sind. Es ist also ausreichend, eine Lösung für das Potential ϕ zu berechnen. Durch Ableitung des Potentials in die gewünschte Raumrichtung erhält man die Geschwindigkeitskomponente für diese Richtung.

$$\vec{u}=\nabla\phi \tag{2.6}$$

Die Kontinuitätsgleichung 2.2 für inkompressible Fluide kann für Potentialströmungen in die sog. Laplace-Gleichung überführt werden:

$$\Delta\phi=\frac{\delta^2\phi}{\delta x^2}+\frac{\delta^2\phi}{\delta y^2}+\frac{\delta^2\phi}{\delta z^2}=0 \tag{2.7}$$

Berücksichtigt man als Volumenkraft nur die Schwerkraft g, also

$$\vec{f}=(0,0,-g)^T \tag{2.8}$$

, so erhält man aus Gl. 2.4 die Bernoulli-Gleichung.

$$\frac{d\phi}{dt}+\frac{1}{2}\left(\nabla\phi\right)^2-gz+\frac{p}{\rho}=const \tag{2.9}$$

Für Potentialströmungen sind in den Gleichungen Druck und Geschwindigkeit entkoppelt. Dies bedeutet, dass, so z. B. die Geschwindigkeit gesucht ist, nur die Lösung der Laplace-Gleichung zu ermitteln ist.

Da die Laplace-Gleichung linear in ϕ ist, kann die Lösung des Gesamtproblems aus Einzellösungen zusammengesetzt werden. Es ist daher möglich, das für eine Lösung gesuchte Potential ϕ durch Überlagerung von n Funktionen φ_i anzunähern.

$$\phi(x,y,z,t)\approx\sum_{i=1}^{n}a_i(t)\varphi_i(x,y,z) \tag{2.10}$$

Die Faktoren $a_i(t)$ bestimmen den zeitabhängigen Einfluss der Einzellösungen $\varphi_i(x,y,z)$ auf die Gesamtlösung. Da nur eine begrenzte Anzahl von Einzellösungen für die Berechnung herangezogen werden kann, ist nur eine Annäherung an die exakte Lösung von ϕ möglich. Für $n\to\infty$ nähern sich exakte Lösung der Laplace-Gleichung und Näherungslösung an. Der Fehler verringert sich gemäß:

$$\Delta\varepsilon\to 0 \qquad \text{für} \qquad n\to\infty \tag{2.11}$$

Auf Grund der Linearität der Laplace-Gleichung ist es möglich, die Umströmung eines Schiffes durch die Verwendung einfacher Elemente[11], d. h. z. B. durch Quellen, Senken, Wirbel oder Dipole, zu konstruieren. Komplexere und numerisch aufwendigere Elemente werden eingesetzt, sofern eine höhere Genauigkeit der Ergebnisse bei konstanter oder geringerer Anzahl an Elementen erzielt werden soll, siehe Abschnitt 2.5.

[11] Siehe Abschnitt 2.4.4

2.4.4 Einfache Elemente der Potentialtheorie

Durch Superposition einfacher Funktionen φ_i, im folgenden Elemente genannt, welche die Bedingungen der Potentialtheorie erfüllen, ist es möglich, für komplexe Probleme eine Lösung zu erzielen.

Abgesehen von der Parallelströmung sind die im folgenden vorgestellten Funktionen singulär. Dies bedeutet, dass die Laplace-Gleichung nicht im Zentrum des jeweiligen Elementes erfüllt ist. Stattdessen gilt mit r als Abstand vom Mittelpunkt des jeweiligen Elementes:

$$|\vec{u}| \to \infty \qquad \text{für} \qquad r \to 0 \tag{2.12}$$

Eine Anwendung dieser sog. Singularitätenverfahren setzt daher voraus, dass die Elemente nicht im untersuchten Flüssigkeitsgebiet Ω angeordnet sind.

2.4.4.1 Parallelströmung

Bewegt sich ein Schiff relativ zum Wasser, so ist das Geschwindigkeitsfeld in der Umgebung des Schiffes durch eine parallel zur Fahrtrichtung angeordnete Parallelströmung geprägt.

Eine beliebige lineare Funktion

$$\varphi = ax + by + cz \tag{2.13}$$

erfüllt die Laplace-Gleichung. Hierdurch kann daher eine Parallelströmung beschrieben werden. Der Geschwindigkeitsvektor läßt sich nach Gl. 2.6 bestimmen.

$$\vec{u} = \nabla\varphi = (a, b, c)^T \tag{2.14}$$

Größe und Richtung des Geschwindigkeitsvektors $\vec{u}$ sind im gesamten Gebiet Ω konstant. Die Stromlinien verlaufen parallel.

2.4.4.2 Punktquelle und -senke

Eine Quellströmung ist eine Strömung, bei der Flüssigkeit in einem Quellpunkt in das Strömungsgebiet eintritt. Eine Senkenströmung ist die Umkehrung der Punktquelle. Fluid verläßt das Strömungsgebiet im Senkenpunkt.

Die Strömung ist für den dreidimensionalen Fall kugelsymmetrisch. Die Stromlinien gehen radial vom Quellpunkt aus. Auf Grund der Kugelsymmetrie tritt nur eine Geschwindigkeitskomponente $u_r(r)$ in radialer Richtung auf.

Da die austretende Flüssigkeitsmenge Q konstant ist, während die durchströmte Oberfläche mit anwachsendem Radius r vom Quell- bzw. Senkenpunkt anwächst, gilt:

$$u_r(r) \to 0 \qquad \text{für} \qquad r \to \infty \tag{2.15}$$

Das Potential einer Punktquelle bzw. -senke im dreidimensionalen Raum ist:

$$\varphi = \pm\frac{Q}{4\Pi r} \tag{2.16}$$

Für die Quellströmung ist das negative Vorzeichen maßgebend; Punktsenken werden durch das positive Vorzeichen beschrieben.

2.4.4.3 Wirbel

Potentialwirbel sind ebene Strömungen, bei denen die Stromlinien konzentrische Kreise um den Mittelpunkt des Wirbels beschreiben. Die Geschwindigkeit in radialer Richtung u_r ist Null, die Geschwindigkeit in Umfangsrichtung u_α ist umgekehrt proportional zum Abstand r vom Mittelpunkt.

$$u_\alpha(r) = \frac{\Gamma}{2\Pi r} \tag{2.17}$$

Γ beschreibt die Zirkulation um den Mittelpunkt des jeweiligen Wirbels:

$$\Gamma = \oint u_\alpha r d\alpha \tag{2.18}$$

2.4.5 Randbedingungen

Die für diese Arbeit maßgebenden Untersuchungen betrachten den stationären Zustand eines Schiffes, welches sich mit konstanter Geschwindigkeit durch das umgebende Fluid bewegt. Zeitabhängige Effekte wie z. B. Turbulenzerscheinungen oder Bewegungen auf Grund von Seegang werden nicht betrachtet. Insofern ist ein Randwertproblem zu lösen. Die zu berücksichtigenden Randbedingungen werden im folgenden benannt.

2.4.5.1 Neumann-Bedingung

Ist $\vec{n}$ der nach außen gerichtete Einheitsnormalenvektor auf der Schiffsoberfläche[12], so gilt:

$$\vec{n} \cdot \nabla\phi = 0 \tag{2.19}$$

Gl. 2.19 drückt aus, dass Fluid nicht durch die Schiffsoberfläche strömt.

Für Schiffe mit Spiegelheck wird angenommen, dass in Fahrt die Strömung an der Außenkante des Spiegels abreißt, der Spiegel also nicht benetzt ist[13]. An den Rändern des Spiegels wird daher Luftdruck erzwungen [1].

$$\frac{\delta\phi}{\delta x} = \begin{cases} -U_\infty\sqrt{1+\frac{2g}{U_\infty^2 z_T}} & \text{für} \quad z_T \leq U_\infty^2/2g \\ -U_\infty & \text{für} \quad z_T > U_\infty^2/2g \end{cases} \tag{2.20}$$

Da bei der Strömung an den Kanten des Spiegels die Geschwindigkeitskomponente in Schiffslängsrichtung dominiert, wird angenommen, dass die Geschwindigkeitsbeträge in y- und z-Richtung vernachlässigbar klein sind.

[12] Für die einzelnen Panels wird dieser derart bestimmt, dass $\vec{n}$ normal zur Schiffsoberfläche in den Kollokationspunkten ist. Die Bedeutung von Kollokationspunkten wird in Abschnitt 2.5.1 näher beschrieben.

[13] Bei Fahrzeugen mit Spiegelheck ist bei langsamer Fahrt ebenfalls eine Benetzung des Spiegels zu verzeichnen. Für diese Fälle ist die Oberfläche numerisch identisch den Schiffsseitenwänden zu behandeln.

2.4.5.2 Kinematische Bedingung

An der Wasseroberfläche ist die kinematische Bedingung zu erfüllen. Diese besagt, dass in Fahrt an der Wasseroberfläche, d. h. für $z(x, y) = \zeta(x, y)$, kein Fluid durch diese dringt.

$$\nabla\phi \cdot \nabla\zeta = \frac{\delta\phi}{\delta z} \tag{2.21}$$

2.4.5.3 Dynamische Bedingung

Wie am Spiegel herrscht an der Wasseroberfläche der Luftdruck p.

$$\frac{1}{2}\left(\nabla\phi\right)^2 - gz = \frac{1}{2}U_\infty^2 \tag{2.22}$$

Werden Gl. 2.21 und Gl. 2.22 kombiniert, so dass ζ eliminiert wird, so ergibt sich die sog. kombinierte Oberflächenbedingung [2]:

$$\frac{1}{2}\nabla\phi\nabla\left(\nabla\phi\right)^2 - g\frac{\delta\phi}{\delta z} = 0 \tag{2.23}$$

Hierdurch wird die zunächst unbekannte Wellenerhebung ζ eliminiert. Ist die Geschwindigkeitsverteilung durch Lösung von Gl. 2.23 bestimmt, so folgt aus der dynamischen Bedingung die Kontur der Wasseroberfläche. Gl. 2.23 ist allerdings für $z = \zeta$, also für die a priori unbekannte Wellenerhebung, zu erfüllen.

Einfache Verfahren verwenden eine linearisierte Oberflächenbedingung. Die Wellenerhebung ζ wird im Vergleich zur Wellenlänge als klein angenommen. Die kombinierte Oberflächenbedingung wird für die ungestörte Wasseroberfläche erfüllt. Numerisch aufwendiger, aber genauer sind nichtlineare Verfahren, die iterativ Gl. 2.23 und 2.22 lösen bis Konvergenz erreicht wird. Hierdurch werden sowohl die dynamische als auch die kinematische Randbedingung an der Oberfläche erfüllt.

2.4.5.4 Strahlungsbedingung

Da potentialtheoretische Berechnungen numerisch korrekte, aber nicht der Realität entsprechende Lösungen ergeben können, bei denen die Wellen sich stromaufwärts ausbreiten, muss durch geeignete Verfahren garantiert werden, dass sich vom Schiff erzeugte Wellen nur in einem Sektor hinter dem Schiff ausbreiten können.

Hierfür werden vielfach Finite-Differenzen-Operatoren angewendet, mit deren Hilfe die in Gl. 2.23 benötigten zweiten Ableitungen des Potentials ermittelt werden. Wird für die Finite-Differenzen-Operatoren ein Aufwind-Schema benutzt, d. h. die zweite Ableitung des Punktes P_i wird aus den ersten Ableitungen am Punkt P_i sowie n stromaufwärts gelegenen Punkten $P_{i-1} \ldots P_{i-n}$ gebildet, so beeinflussen die Strömungszustände stromabwärts den Zustand am betrachteten Punkt P_i kaum.

2.4.5.5 Bedingung des offenen Randes

Da nur ein Teil des Fluids diskretisiert werden kann, entsteht ein künstlicher Rand des Berechnungsgebietes. Vom Schiff erzeugte Wellen laufen unreflektiert durch diesen Rand.

Am vorderen Rand - dort treten Wellen nicht auf - ist zu garantieren, dass keine Störgeschwindigkeiten auftreten. Die Strömung in diesem Bereich ist eine Parallelströmung mit der Geschwindigkeit $\vec{u} = (-U_\infty, 0, 0)^T$.

Wellen, die das diskretisierte Strömungsgebiet am hinteren bzw. seitlichen Rand verlassen, werden vielfach durch spezielle Operatoren in diesem Bereich stark gedämpft. Eine Reflexion der Wellen wird dadurch unterbunden. Alternativ ist es möglich, auf der Wasseroberfläche Quell- und Kollokationspunkte verschoben zueinander anzuordnen. Für weitere Ausführungen wird hier auf die Literatur, siehe u. a. [3] und [4], verwiesen.

2.4.5.6 Dynamisches Gleichgewicht

Die Schwimmlage des Schiffes wird derart bestimmt, dass sich ein dynamisches Gleichgewicht einstellt:

$$\sum_i F_i = 0 \quad \text{und} \quad \sum_i M_i = 0 \tag{2.24}$$

Die Summe aller auf das Schiff wirkenden Volumenkräfte und auf der Rumpfoberfläche Ω_S angreifenden Kräfte F_i und Momente M_i ist Null. Trimm, Absenkung und Krängung[14] werden berechnet, so dass die dynamischen Kräfte und Momente ausgeglichen werden.

Für die Bestimmung der am Schiff angreifenden generalisierten Kräfte wird vielfach eine Integration des Drucks über die Schiffsoberfläche durchgeführt. Hierbei ist zu beachten, dass sich die benetzte Oberfläche auf Grund der Wellenbildung verändert. Die Integration ist daher über die tatsächlich benetzte Oberfläche des in Fahrt befindlichen Schiffes durchzuführen.

Die potentialtheoretisch ermittelte, am Schiffskörper in x-Richtung angreifende Kraft F_x ist gleich dem negativen Wellenwiderstand. Der Wellenwiderstandskoeffizient c_W ergibt sich nach:

$$c_W = \frac{-F_x}{\frac{1}{2}\rho U_\infty^2 S} \tag{2.25}$$

S ist die benetzte Schiffsoberfläche des Schiffes.

2.5 Mathematisches Modell

Die im Rahmen der Erstellung des physikalischen Modells entwickelte mathematische Beschreibung des Problems besteht im Bereich der Hydrodynamik vielfach aus nichtlinearen, z. T. instationären Differentialgleichungssystemen, welche nicht direkt gelöst werden

[14] Durch dynamische Effekte hervorgerufen tritt für den stationären Fall Krängung nur bei schräger Anströmung oder einer asymmetrischen Rumpfform auf.

können. Daher ist eine Umwandlung durch geeignete Verfahren in algebraische Ausdrücke erforderlich; es kommen Näherungsfunktionen zum Einsatz.

Mit höherer Rechenleistung ist es möglich, komplexere Näherungsfunktionen der Potentialtheorie für die Lösung des Wellenwiderstandsproblems der Umströmung eines Schiffes zu verwenden. Hierbei haben sich in den letzten Jahren die sogenannten Panel-Verfahren[15] durchgesetzt.

2.5.1 Elemente für Panelverfahren

Bei Panel-Methoden werden die Oberflächen, an der die in Abschnitt 2.4.5 beschriebenen Randbedingungen zu erfüllen sind, in kleine flächige Bereiche, den sog. Panels, unterteilt. Die einzelnen Panels verhalten sich dabei ähnlich einer über die Fläche des Panels verteilten Quelle bzw. Senke. Daher erfüllen Panels ohne weitere Nebenbedingungen die Laplace-Gleichung, siehe [2].

Die Form der Panels wie auch die Verteilung der Quellstärke über jedes Panel ist je nach Panel-Verfahren unterschiedlich. Gebräuchlich sind u. a. [5]:

- Eine Punktquelle bzw. -senke je Panel.
- Punkthaufen, d. h. mehrere Quellen sind gleichmäßig auf dem Panel verteilt. Alle Quellen haben gleiche Stärke.
- Konstante Quellstärke über die ebene Vierecksfläche des Panels.
- Konstante Quellstärke verteilt über ein räumliches, nicht notwendigerweise ebenes Vieleck.

Die mathematische Komplexität wie auch der für die Lösung erforderliche numerische Aufwand steigt für die aufgeführten Panel-Typen von oben nach unten. Bei Panel-Verfahren höherer Ordnung[16] ist die Verteilung der Quellstärke über die Oberfläche nicht konstant. Stattdessen wird z. B. eine lineare oder parabolische Verteilung angenommen.

Direkt auf dem Panel wird das Potential des jeweiligen Panels unendlich. Dies wird als Flächensingularität bezeichnet. Eine direkte numerische Auswertung des Quell-Potentials ist nicht möglich. Durch Umformungen[17] ist es allerdings möglich, für Punkte auf dem Panel mathematisch auswertbare Ausdrücke herzuleiten.

Auf den einzelnen Panels ist eine endliche Anzahl diskreter Kollokationspunkte angeordnet, in denen bei der numerischen Auswertung die Randbedingungen durch geeignete

[15] Der deutsche Ausdruck Paneel-Verfahren wird in dieser Arbeit nicht verwendet, da sich international die englischsprachige Bezeichnungsweise eingebürgert hat.

[16] Entwickelt unter anderem von Hughes/Bertram [6].

[17] Es ist z. B. möglich, eine einfach berechenbare Funktion mit identischer Singularität - häufig wird hierfür eine Quellverteilung konstanter Stärke auf einer Kugeloberfläche, welche das Panel im Panelmittelpunkt tangiert, verwendet - von der Quellstärke des Panels zu subtrahieren und nach der numerischen Auswertung der verbleibenden Quellstärke des Panels diesen Betrag wieder zu addieren [2].

Variation der Quellstärken exakt erfüllt werden. Ist der Abstand zwischen den Kollokationspunkten hinreichend klein, kann davon ausgegangen werden, dass zwischen diesen Punkten die geforderten Bedingungen in guter Näherung erfüllt sind. Ein zu geringer Abstand der Kollokationspunkte ist nicht wünschenswert, da es ansonsten auf Grund der sehr geringen Abstände zu Rundungsfehlern bei der Berechnung kommen kann.

2.5.2 Beispiel für ein Panel-Problem

Am Beispiel eines umströmten ebenen Zylinders in einem unendlich ausgedehnten Flüssigkeitsgebiet wird im folgenden der Lösungsweg eines Panel-Problems exemplarisch dargestellt.

Das Potential der Anströmung ϕ_∞ wird durch eine Parallelströmung mit der Geschwindigkeit $\vec{u}_\infty$ beschrieben.

$$\phi_\infty = \vec{u}_\infty \cdot \vec{x} \tag{2.26}$$

Die Körpergeometrie wird in m nicht gekrümmte Panels mit gleicher Länge l unterteilt. Das die Umströmung des Körpers bewirkende Störpotential ϕ_S wird durch die Anordnung von Quellen und Senken auf den Panels gebildet. Je Panel wird in diesem Beispiel ein Quell- bzw. Senkenpunkt mit dem Potential φ_{Q_i} angeordnet, siehe Abb. 2.5. Die Ergiebigkeit der Quelle oder Senke i beträgt Q_i

$$\phi_S = \sum_{i=1}^{m} \varphi_{Q_i} = \sum_{i=1}^{m} \frac{Q_i}{4\Pi r} \tag{2.27}$$

r beschreibt den Abstand zwischen Quelle bzw. Senke[18] und dem Punkt $\vec{x}$, für welchen

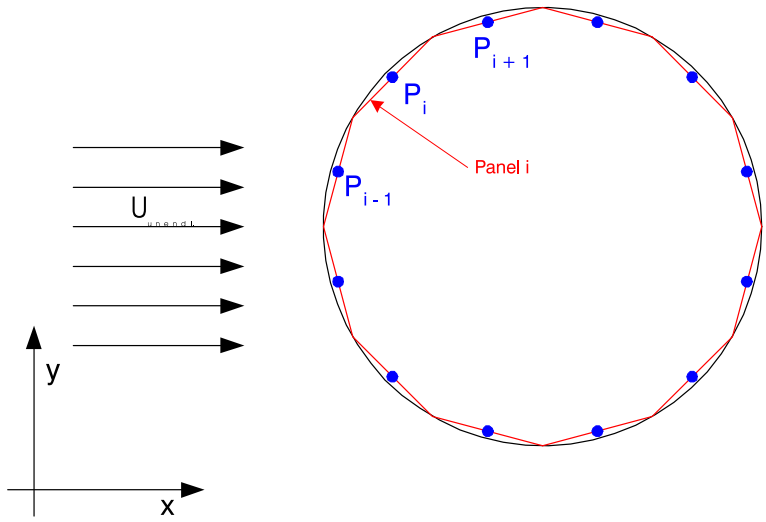

Abb. 2.5: Diskretisierung eines Zylinders für potentialtheoretische Berechnung

das Störpotential $\phi_S(\vec{x})$ bestimmt wird.

Die Potentiale der Parallelströmung sowie der Quellen, also nach Gl. 2.10 auch das Gesamtpotential

$$\phi = \phi_\infty + \phi_S \tag{2.28}$$

, erfüllen alle Randbedingungen bis auf die Neumann-Bedingung.

Die Neumann-Bedingung, siehe Gl. 2.19, drückt aus, dass keine Durchströmung der Oberfläche stattfindet. Für jedes Panel k mit dem Kollokationspunkt $\vec{P}_k$ muss daher gelten:

$$\frac{\delta\phi(\vec{P}_k)}{\delta\vec{n}_k} = 0 \tag{2.29}$$

$\vec{n}_k$ ist hierbei der Normalenvektor auf das Panel k im Kollokationspunkt $\vec{P}_k$.

Aus Gleichung 2.27, 2.28 und Gl. 2.29 ergibt sich:

$$\frac{\delta\phi}{\delta\vec{n}_k} = \frac{\delta\phi_\infty}{\delta\vec{n}_k} + \sum_{i=1}^{m} \frac{\delta\varphi_{Q_i}}{\delta\vec{n}_k} = 0 \tag{2.30}$$

Gl. 2.30 kann unter Berücksichtigung geometrischer Beziehungen der Panels zueinander in ein algebraisches Gleichungssystem überführt werden. Hierdurch werden die Quellstärken Q_i auf den Panels bestimmt. In [7] ist dies ausführlich dargelegt.

Die Neumann-Bedingung wird in den Kollokationspunkten genau erfüllt. Mit steigender Panelanzahl nähert sich die Lösung außerhalb der Kontrollpunkte den analytisch erzielbaren Ergebnissen an.

2.6 Numerisches Modell

Die Entwicklung des numerischen Modell ist, wie bereits erwähnt, stark abhängig von bereits existierenden Teillösungen. Sie wird durch Einsatzgebiet und Programmierumgebung beeinflusst. Weiterhin gilt das in Abschnitt 2.2 dargelegte.

[18] Da Quell- und Senkenpunkte sich nur im Vorzeichen der Ergiebigkeit Q unterscheiden, werden im folgenden nur die Begriffe Quellpunkt bzw. Quelle verwendet.

Kapitel 3

Grundlagen - Kurven und Flächen

Das Gebiet der *computer aided geometry*[1] beinhaltet die computergestützte Modellierung und Bearbeitung von Geometrieobjekten wie z. B. Kurven, Flächen und Volumina. Mit Hilfe von mathematischen Verfahren werden geometrische Objekte beschrieben und hierdurch für spätere Modifikationen, aber auch für die Berechnung von z. B. Länge, Volumen oder Schwerpunkt eines Objektes, bereitgestellt.

In diesem Kapitel wird, soweit für diese Arbeit relevant, ein Überblick über Möglichkeiten zur geometrischen Modellierung gegeben. Abschnitt 3.1 beschreibt abstrakt Möglichkeiten zur Definition von geometrischen Objekten. Eine Übersicht über die Anforderungen an CAG-Objekte für die Verwendung im Schiffbau wird in Abschnitt 3.2 gegeben. In den Abschnitten 3.3 bis 3.6 werden Verfahren vorgestellt, durch die Kurven und Flächen beschrieben werden können.

3.1 Möglichkeiten zur mathematischen Definition

In der analytischen Geometrie stehen für die eindeutige Definition von Flächen oder Kurven vier mögliche Ansätze zur Verfügung. Diese werden im folgenden erläutert[2].

3.1.1 Explizite Form

Die explizite Form einer Kurve oder Fläche in R^n beschreibt einen Punkt auf der Kurve bzw. Fläche eindeutig durch $n-1$ Koordinaten. So kann in R^2 eine Parabel beschrieben werden durch z. B.

$$f(x) = x^2 + 4 \cdot x + 3 \tag{3.1}$$

, d. h., wird eine x-Koordinate gegeben, so kann die zugehörige unbekannte Koordinate direkt berechnet werden.

[1] Im folgenden wird die Kurzform CAG verwendet.

[2] Auf Grund der einfacheren Darstellung werden Beispiele für ebene Probleme gegeben. Die erläuterten Verfahren sind ebenfalls im R^3 anwendbar.

3.1.2 Implizite Form

Durch die implizite Form wird eine Gleichung gegeben, die durch alle Punkte auf der Kurve oder Fläche erfüllt wird. Für einen Kreis in R^2 läßt sich schreiben:

$$f(x, y) = x^2 + y^2 - 1 = 0 \tag{3.2}$$

Hierdurch ist es möglich, geometrische Objekte wie z. B. Kreise zu definieren, die durch eine explizite Form nicht darstellbar sind.

3.1.3 Parametrische Definition

In Parameterform werden alle Koordinaten eines Punktes separat als explizite Funktion eines Parameters t definiert. So läßt sich die in Abb. 3.1 dargestellte Schraubenkurve beschreiben wie folgt:

$$\begin{aligned} x(t) &= r \cdot \cos t \\ y(t) &= r \cdot \sin t \\ z(t) &= h \cdot t \end{aligned} \tag{3.3}$$

Vielfach wird der Parameter t so gewählt, dass gilt $t \in R, 0 \leq t \leq 1$.

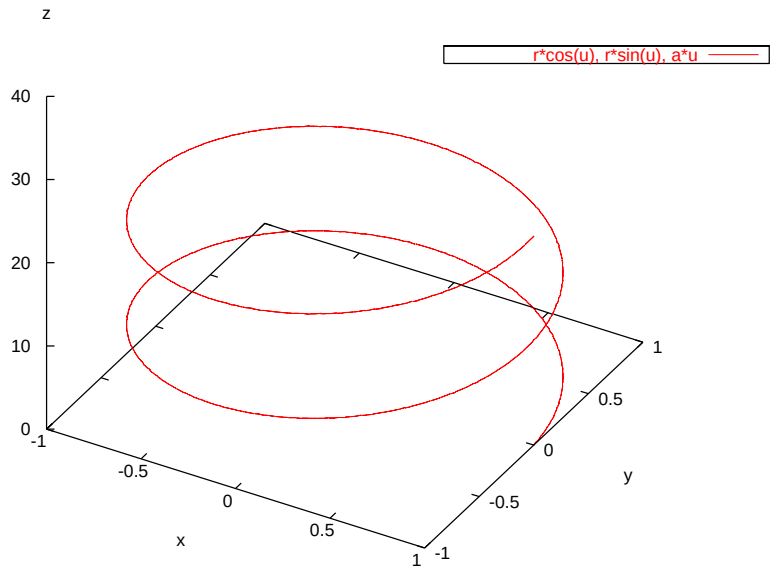

Abb. 3.1: Schraubenkurve als Beispiel einer Parameterkurve in R^3

3.1.4 Rationale Darstellung

Bei rationaler Darstellung wird ein Objekt im projizierten Raum R^n definiert. Eine Rückprojektion auf R^{n-1} wird u. a. zur Darstellung verwendet. Eine parametrisierte Kurve $R(u) = x(u), y(u)$ in R^2 ergibt sich aus der entsprechenden rationalen Funktion $R_r(u, w)$ [8].

$$R(u) = \frac{R_r(u, w)}{w(u)} = \left(\frac{x(u)}{w(u)}, \frac{y(u)}{w(u)}\right) \tag{3.4}$$

Die Gewichtsfunktion $w(u)$ ermöglicht es, komplexe Objekte zu beschreiben, die in nicht rationaler Darstellung nur schwer zu definieren wären.

3.2 Anforderungen an CAG-Objekte

Sollen durch CAG-Objekte Kurven und Flächen dargestellt werden, so werden an deren Funktionalität wie auch an die Handhabung im praktischen Einsatz diverse Anforderungen gestellt.

Um eine exakte Übereinstimmung zwischen CAD-Modell und realem Objekt zu erreichen, müssen durch CAG-Objekte bzw. durch Kombinationen von Objekten beliebige Formen dargestellt werden können. Da vielfach nur lokale Formänderungen an einem Objekt vorgenommen werden, ist eine globale Definition des Objektes, d. h. eine lokale Variation führt zu globalen, das ganze Objekt beeinflussenden Formänderungen, nicht wünschenswert [9].

Eine einfache und effiziente Erstellung und Modifikation von CAG-Objekten ist für den praktischen Einsatz erforderlich. Weiterhin soll durch die CAG-Modellierung eine Basis geschaffen werden, auf welcher weitere notwendige Bearbeitungsschritte wie z. B. die Durchführung von Berechnungsverfahren aufsetzen können. Hierfür ist es notwendig, dass umfassende Informationen über die Objekte zur Verfügung stehen bzw. auf einfachem Wege gewonnen werden können.

3.3 Approximation durch Polynome

Bei klassischen Interpolationsproblemen kommen vielfach Approximationsverfahren zur Anwendung, die eine Punktmenge mit Hilfe von Polynomfunktionen annähern. Auf Grund der Neigung zum Oszillieren bei Verwendung von Polynomen höheren Grades wie auch der globalen Auswirkungen von lokalen Änderungen sind diese Verfahren zur Modellierung geometrischer Objekte ungeeignet [8].

3.4 Spline-Kurven

Bewährt haben sich Ansätze, welche das zu beschreibende Gebiet in Teilintervalle unterteilen und diese anschließend getrennt durch z. B. Polynome niedrigen Grades modellieren. Geeignete Übergangsbedingungen zwischen Teilintervallen garantieren, dass die gewünschte Stetigkeit an den Übergängen erreicht wird. Bekannte Beispiele sind Bezier[3]- und Spline-Kurven bzw. -Flächen [10].

B-Spline-Kurven werden heute in den gängigen CAD-Systemen zur Modellierung von Kurven eingesetzt. Diese parametrisierten Kurven zeichnen sich dadurch aus, dass eine lokale Modifikation wie z. B. die Verschiebung eines Punktes sich nur auf einen Bereich der Kurve auswirkt. Die oben aufgeführte Forderung nach lokaler Definition wird also erfüllt. B-Spline-Kurven ermöglichen es weiterhin, durch entsprechende Definition der sog. Kontrollpunkte die Stetigkeit einer Kurve zu beeinflussen. Somit können C^n-stetige Kurven, also auch Kanten modelliert werden.

3.4.1 B-Spline-Funktionen

Basis der B-Spline-Kurven sind die sog. B-Spline-Funktionen. Eine gebräuchliche Definition von B-Spline-Funktionen ist mittels rekursiver Formulierung. Eine B-Spline-Funktion $N_{i,k}$ der Ordnung[4] k und dem Grad[5] $k-1$ wird beschrieben durch

$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(t) + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t) \qquad \text{für} \qquad i = 0, 1, \ldots, n \tag{3.5}$$

mit

$$N_{i,0}(t) = \begin{cases} 1 & : \quad t_i \leq t < t_{i+1} \\ 0 & : \quad t < t_i, t \geq t_{i+1} \end{cases} \tag{3.6}$$

t_i sind hierbei die bestimmenden Parameter[6] der Funktion. An Segmentübergängen sind B-Spline-Funktionen C^{k-r-1} stetig, wobei r die Anzahl der identisch definierten Knoten angibt.

3.4.2 B-Spline-Kurven

Eine B-Spline-Kurve wird für gegebene Kontrollpunkte $d_i = d_0, d_1, \ldots, d_n$ der Kurve definiert durch:

$$\vec{x}(t) = \sum_{i=0}^{n} d_i N_{i,k}(t) \tag{3.7}$$

[3]Bezier-Kurven bieten keine Möglichkeit der lokal begrenzten Formänderung. Wird ein Kontrollpunkt modifiziert, so hat dies globale Auswirkungen auf die Kurve.

[4]Die Ordnung einer B-Spline-Funktion entspricht der Anzahl der Punkte, die zur Definition der Kurve auf einem Teilintervall verwendet werden.

[5]Unter dem Grad einer B-Spline-Funktion ist der Grad des Polynoms zu verstehen, welches zur Approximation verwendet wird.

[6]Diese werden auch als Knoten bezeichnet.

$\vec{x} = (x, y, z)^T$ beschreibt einen Punkt auf der Kurve als Funktion des Parameters t; k bezeichnet wiederum die Ordnung der Kurve.

3.5 NURBS-Kurven

Wird, wie in Abschnitt 3.1.4 beschrieben, für die Definition der B-Spline-Kurven eine rationale Darstellungsweise herangezogen, so ergibt sich:

$$X(t) = \frac{\sum_{i=0}^{n} d_i N_{i,k}(t) w_i}{\sum_{i=0}^{n} N_{i,k}(t) w_i} \tag{3.8}$$

Durch die sog. Gewichte w_i ist eine weitere Möglichkeit gegeben, die Form der Kurve sowie den Einfluss einzelner Knoten auf die Gesamtkontur zu kontrollieren.

Eine B-Spline-Kurve in rationaler Darstellung wird als NURBS-Kurve[7] bezeichnet, so die Knoten nicht äquidistant sind. Neben oben genannten Eigenschaften zeichnen sich NURBS-Kurven aus durch [11]:

- *convex hull property*: Die Kurve verläuft ausschließlich innerhalb des durch die Kontrollpunkte aufgespannten Polygons.
- Jede NURBS-Kurve folgt der durch das Kontrollpolygon beschriebenen Kurve.
- Affine Transformationen der Kontrollpunkte führen zu einer affinen Transformation der NURBS-Kurve.
- *variation diminishing property*: Schneidet eine Gerade oder Ebene das Kontrollpolygon n mal, so wird die NURBS-Kurve höchstens $m \leq n$ mal geschnitten.

3.6 NURBS-Flächen

Wird eine NURBS-Kurve in eine zweite Parameterrichtung verschoben, so entsteht eine sog. Tensor-Produktfläche mit einem rechteckigen Parametergebiet, eine NURBS-Fläche. Diese ist definiert wie folgt:

$$X(u, v) = \frac{\sum_{i=0}^{n} \sum_{j=0}^{m} d_{i,j} N_{i,k}(u) N_{j,k}(v) w_{i,k}}{\sum_{i=0}^{n} \sum_{i=0}^{m} N_{i,k}(u) N_{j,k}(v) w_i} \tag{3.9}$$

Die Punkte $d_{i,j}$ sind die ähnlich einem Gitter im Raum angeordneten Kontrollpunkte der Fläche. Durch $N_{i,k}$ und $N_{j,k}$ werden Spline-Funktionen der Ordnung k in Parameterrichtung u respektive v definiert.

Analog zu NURBS-Kurven gelten für die NURBS-Flächen oben beschriebene Eigenschaften.

[7] NURBS: Non Uniform Rational B-Spline

Kapitel 4

Netzgenerierung

Bei CFD-Berechnungnen hat die Qualität der verwendeten Netze[1] erheblichen Einfluss auf die Güte der Ergebnisse, siehe Abschnitt 2.3.2. Daher ist die Gittererzeugung in der Vergangenheit Ziel umfangreicher und eingehender Untersuchungen gewesen[2].

Aufbauend auf den in Abschnitt 4.1 erläuterten Begriffsdefinitionen wird eine grundlegende Einordnung der möglichen Gittertypen in Abschnitt 4.3 gegeben. Abschnitt 4.4 beschreibt in der Praxis erprobte Algorithmen zur Gitterzeugung. Insbesondere werden Algorithmen beschrieben, die auf mathematischen Untersuchungen basieren und welche nachweislich für eine gegebene Problematik „schöne“ Netze entsprechend der Definition in Abschnitt 4.2 erzeugen. Abschnitt 4.5 erläutert abschließend Verfahren zur Optimierung eines bestehenden Netzes.

4.1 Begriffsdefinitionen

Der Begriff Netzgenerierung beschreibt die Approximation einer mit Hilfe der CAG beschriebenen Geometrie durch diskrete, meist ebene Flächenelemente, den sog. Panels. Im Zuge der Panelierung werden auf der Geometrie Elemente derart angeordnet, dass die Kontur möglichst gut angenähert sowie den im folgenden Abschnitt beschriebenen Anforderungen genügt wird.

Panels werden definiert durch die Koordinaten in den Eckpunkten, den sog. Knoten, sowie durch Relationseigenschaften wie z. B. Kanteninformationen zwischen diesen Knoten.

[1] Vergleichbar der Äquivalenz von *mesh* und *grid* im Englischen werden in dieser Arbeit die Bezeichnungen Gitter und Netz als gleichbedeutend behandelt.

[2] Für eine umfassende Einführung in und Zusammenfassung der Thematik Netzgenerierung sei an dieser Stelle z. B. auf [12] und [13] verwiesen.

4.2 Anforderungen

FEM[3] wie auch CFD-Berechnungen stellen sehr hohe Anforderungen an die Güte des verwendeten Netzes. Während z. B. im Bereich der Computergraphik einzelne degenerierte Elemente keine signifikanten Folgen haben, können diese zu deutlichen Verfälschungen der Ergebnisse führen bzw. das Konvergenzverhalten beeinflussen.

4.2.1 Allgemeine Anforderungen

Nach [14] sind durch Verfahren zur Netzgenerierung folgende Anforderungen zu erfüllen:

- Automatische Netzgenerierung: Eine explizite Definition einzelner Punkte und Kanten durch den Anwender ist nicht erforderlich. Allerdings kann der Anwender die Anordnung der Elemente mit Hilfe globaler oder lokaler Parameter steuern.
- Minimierung der Elementanzahl: Dies ist erforderlich, um die für die anschließende Berechnung erforderliche Rechenzeit zu begrenzen.
- Bedingte Dreiecksnetze: Wird eine Oberfläche bestehend aus mehreren Flächenstücken, die nur mit C^0-Stetigkeit aneinanderschließen, vernetzt, so müssen die C^0-stetigen Schnittlinien durch das Netz exakt wiedergegeben werden. Es werden keine Elemente erzeugt, die außerhalb der Berandung liegen.
- Approximation mit kontrollierbarem Fehler: Durch das Netz soll eine gute Approximation des geometrischen Objektes erreicht werden, d. h. der Wert einer für alle Elemente ermittelten Qualitätsfunktion ist kleiner als eine durch den Anwender vorgegebene Toleranzschranke.
- Netzdichtekontrolle: Eine Steuerung der Elementverteilung über den zu vernetzenden Bereich ist durch den Anwender möglich[4].
- Form der Elemente: Es dürfen keine stark verzerrten Elemente auftreten. Es ist anzustreben, dass Flächenverhältnisse benachbarter Elemente i, j nicht stark von $A_i/A_j = 1$ abweichen.

4.2.2 Oberflächennetze

Eine eindeutige und konkrete Definition der Anforderungen an Netze für auf der Potentialtheorie basierende Berechnungen ist nur schwer zu geben.

Während in der gesprochenen Sprache vielfach dem aus der Erfahrung gewonnenen Wunsch nach „schönen“ Netzen Ausdruck verliehen wird, und dies durch den Betrachter durchaus

[3] Berechnungen basierend auf Finite Elemente Methoden

[4] Hierdurch ist, so a priori eine Abschätzung der Lösung vorliegt, eine Steuerung der Vernetzung möglich. Bereiche mit besonders komplexen Strömungsverhältnissen können somit feiner vernetzt und hierdurch die Ergebnisgüte verbessert werden.

bewertet werden kann, ist diese Aussage bei der Entwicklung eines Vernetzungsalgorithmus nur bedingt verwendbar. Für die numerische Erzeugung und Bewertung eines Netzes sind stattdessen mathematisch quantifizierbare Faktoren ausschlaggebend.

Bezogen auf die Erzeugung von Oberflächennetzen für potentialtheoretisch basierte CFD-Rechnungen können aus obigen Bedingungen folgende konkrete Anforderungen abgeleitet werden, siehe [15], [16] :

- Um eine adäquate Auflösung der ausgelenkten Wasserlinienkontur zu ermitteln, sollten für eine Berechnung mit Froudezahl F_n die Abmessungen eines Elements l_x in Schiffslängsrichtung kleiner sein als:

$$l_x < F_n^2 \cdot L_{PP} \tag{4.1}$$

- Das Seitenverhältnis von Viereckselementen sollte nicht größer sein als 1 : 2. Der kleinste Winkel eines Dreieckselementes sollte nicht kleiner sein als $\alpha_{min} = 30°$ Dies unterbindet die Ausbildung von degenerierten bzw. stark verzerrten Elementen.

- Für das Flächenverhältnis benachbarter Elemente mit den Flächen A_i und A_j sollte gelten:

$$A_i/A_j < 2 \tag{4.2}$$

- Klaffungen zwischen und Überlappungen von Elementen sind auszuschließen.

- Um eine gute Repräsentation der zu approximierenden Oberfläche zu erreichen, sollte der Winkel zwischen den Normalenvektoren benachbarter Elemente möglichst gering sein. Hieraus ergibt sich die Forderung, dass in Bereichen starker Krümmung die durchschnittliche Elementfläche verringert, also die Elementdichte erhöht wird.

Da allerdings auch Netze, welche obige Anforderungen erfüllen, nicht zwingend als „schöne“ Netze eingestuft werden, ist es erforderlich, dem Anwender zusätzliche Einflussmöglichkeiten bei der Netzerzeugung bereitzustellen. Zu nennen sind u. a.:

- die Möglichkeit, die Panelfläche in Abhängigkeit lokaler Oberflächeneigenschaften zu beeinflussen,

- sowie die Option, dass durch den Benutzer Anordnung und Elementdichte gezielt in Abhängigkeit der Lage des Elements auf der Rumpfoberfläche gesteuert werden können.

4.3 Einordnung von Gittern

Wird versucht, eine Klassifizierung von Gittern und Verfahren zur Gittererzeugung zu erreichen, so bietet sich zum einen eine Einordnung auf Grund der geometrischen Eigenschaften des zu vernetzenden Objektes, zum anderen an Hand der Eigenschaften des erzeugten Netzes an.

4.3.1 Linie, Ebene, Fläche, Volumen

Zu vernetzende Objekte können sein: Kurven, ebene Flächen und Flächen im R^3 sowie Volumina. Ein Überblick über die sich hieraus ergebenden Netztypen ist in Abb. 4.1 dargestellt.

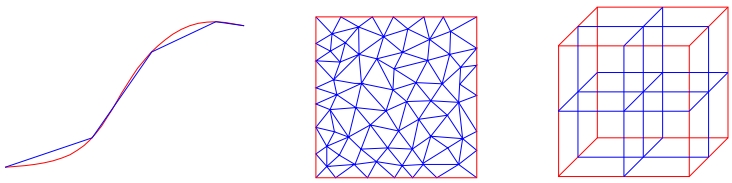

Abb. 4.1: Beispiele der Vernetzung von Linien, Flächen und Volumina

Gitter auf dreidimensionalen Oberflächen nehmen eine besondere Stellung ein, da es sich hierbei um die Vernetzung eines Körpers im R^3 durch flächige, zweidimensionale Elemente handelt. In der Literatur wird diese Problematik vielfach mit dem Begriff 2.5D-Vernetzung o. ä. beschrieben [17].

4.3.2 Strukturierte und unstrukturierte Gitter

Es können, abgesehen von Mischformen wie blockstrukturierten Gittern [18] bzw. Chimera-Netzen [19], zwei Typen von Gittern unterschieden werden, nämlich unstrukturierte und strukturierte Gitter, siehe Abb. 4.2.

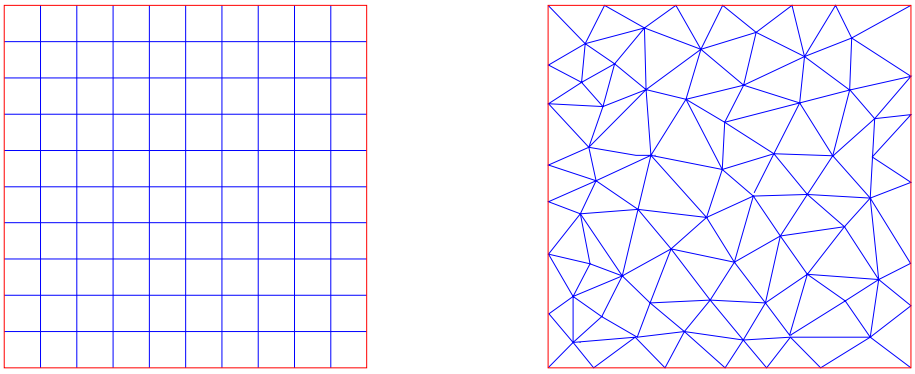

Abb. 4.2: Beispiele für stukturierte (links) und unstrukturierte (rechts) Netze

Strukturierte Netze Kennzeichnendes Merkmal strukturierter Netze ist eine regelmäßige Struktur, d. h. jedes Objekt[5] hat eine konstante, durch die Netzstruktur festgeschriebene Anzahl von Nachbarobjekten[6].

Eine einfache Netzerzeugung wie auch eine sehr effiziente Verwaltung der die Netzstruktur beschreibenden Informationen sind Vorteile dieses Typs. Als Nachteil ist zu nennen, dass auf Grund der zwingend festgelegten Regelmäßigkeit des Netzes es z. T. nicht möglich ist, für Objekte von komplexer Geometrie[7] Gitter zu erzeugen, welche höheren Qualitätsansprüchen genügen.

So können z. B. bei einer 2.5D-Netzgenerierung Knicke innerhalb der Oberfläche nur ungenügend wiedergegeben werden. Sind starke Querschnittsänderungen eines Objekts vorhanden, kann dies, so keine Unterteilung des Netzes in mehrere Blöcke durchgeführt wird, zu stark verzerrten Elementen führen.

Unstrukturierte Netze Unstrukturierte Netze sind das Gegenstück zu strukturierten Netzen. Es sind keine festgelegten Nachbarschaftsrelationen gegeben. Dies erfordert eine aufwendige Speicherung und Verwaltung der Lage eines Elementes sowie der Anordnung von Objekten in Bezug zueinander.

Da Elemente den geometrischen Eigenschaften des Objektes angepasst angeordnet werden können, ist es durch unstrukturierte Netze möglich, eine gute Approximation eines geometrischen Objektes zu erreichen. Zu beachten ist allerdings, dass die Anordnung eines Elementes nicht nur durch die Eigenschaften des zu vernetzenden Objektes beeinflusst wird, sondern in einer Weise durchgeführt wird, dass keine ungünstig geformten Elemente entstehen bzw. eine deutliche Differenz der Größe benachbarter Elemente zu verzeichnen ist. Es sind also existierende Elemente mit zu betrachten. Hierfür ist eine Beurteilung der Güte des Netzes durch geeignete Verfahren durchzuführen; eine Gittermodifikation ist bei ungenügendem Ergebnis notwendig, siehe Abschnitt 4.5.

4.4 Generierung unstrukturierter Netze

Dieser Abschnitt befasst sich mit Verfahren zur Erzeugung unstrukturierter Netze. Es werden Verfahren benannt und beschrieben, die sich für die Netzgenerierung auf einer komplexen Oberfläche in R^3 eignen. Zu Beginn wird erläutert, warum für die Lösung der gegebenen Problemstellung die Verwendung von unstrukturierten Gittern bevorzugt wird.

Verfahren[8], die vergleichbar dem Ansatz der rationalen Darstellung, Abschnitt 3.1.4, eine Vernetzung in R^2 durchführen gefolgt von einer Rückprojektion auf die Oberfläche in R^3, sind nach [12] und [23] nicht für sehr komplexe, aus Teilflächen zusammengesetzte Probleme geeignet und werden an dieser Stelle nicht beschrieben.

[5] Dies können Knoten, Elemente, aber auch Kanten sein.
[6] Abweichungen ergeben sich im Bereich der Berandung.
[7] Dies können Schiffe, aber auch andere Objekte aus dem Bereich des allgmeinen Maschinenbaus sein.
[8] Z. B. transfinite Interpolation [20], elliptische [21] und hyperbolische [22] Netzgenerierung

4.4.1 Warum unstrukturierte Gitter?

Insbesondere bei raschen Änderungen von geometrischen Eigenschaften[9] sind bei strukturierten Netzen häufig schlecht konditionierte Elemente[10] zu verzeichnen. Diese Effekte können zwar durch die Verwendung von blockstrukturierten Gittern, d h. eine Kombination mehrerer auf jeweils ein Gebiet beschränkter strukturierter Netze, minimiert werden, erfordert dann allerdings eine durch den Benutzer vorzugebende Blockstruktur; eine Vernetzung mit keiner oder nur minimaler Benutzerinteraktion ist nicht bzw. nur schwer zu realisieren [15].

Die Außenhaut eines Schiffes weist häufig starke Formänderungen innerhalb eines begrenzten Gebiets auf[11]. Durch die Verwendung von Verfahren zur Generierung unstrukturierter Gitter können Elemente entsprechend den in Abschnitt 4.2 genannten Anforderungen plaziert werden. Durch geeignete Wahl von Punkten, Kanten und Flächen[12] kann die Ausbildung degenerierter Elemente verhindert werden.

4.4.2 Delaunay-Triangulisierung

Für eine Schar P von n gegebenen Punkten[13] $p_0, p_1, \ldots, p_{n-1}$ erzeugt eine Delaunay-Triangulisierung eine optimale Triangulisierung in Hinblick auf eine Maximierung der minimalen Winkel innerhalb jedes Elements [8].

Dies wird erreicht durch das sog. Delaunay-Kriterium[14], welches besagt, dass kein Punkt p_i innerhalb eines Umkreises für den ebenen Fall bzw. einer Umkugel im R^3, jeweils gebildet durch die Eckpunkte der Elemente, liegt, siehe Abb. 4.3. Für Flächen im R^3 ist zu garantieren, dass die erzeugten Dreieckselemente die Oberfläche approximieren. Es ist zu unterbinden, dass bei nicht konvexen Flächen eine Triangulisierung außerhalb der durch die Fläche definierten Region stattfindet.

Das Delaunay-Kriterium bietet nur eine Möglichkeit, die Güte eines Netzes abzuschätzen bzw. zu untersuchen, ob eine gegebene Triangulisierung eine Delaunay-Triangulisierung ist, also dieses Kriterium erfüllt. Ausgehend von einer Punktemenge gibt es diverse Verfahren[15] zur Erzeugung einer Delaunay-Triangulisierung. Eine ausführliche Beschreibung dieser Algorithmen wird in [14] gegeben.

Da durch das Delaunay-Verfahren lediglich eine gegebene Punkteschar optimal triangulisiert wird, ist zuerst durch einen zweiten Algorithmus eine die in Abschnitt 4.2 genannten Anforderungen erfüllende Punktemenge zu erzeugen. Vielfach wird hierzu ein Verfahren

[9] Dies können Krümmungen von Kurven oder Flächen, aber auch Querschnittsänderungen bei Volumina sein.

[10] Z. B. besonders große oder kleine bzw. stark verzerrte Elemente.

[11] Als kritische Bereiche sind z. B. der Bugwulstbereich wie auch der Übergang zwischen Spiegel und angrenzender Außenhaut zu nennen.

[12] Für Volumennetze

[13] Die Punkte werden auch als Knoten bezeichnet.

[14] Im Englischen wird dieses häufig als *empty sphere property* bezeichnet.

[15] U. a. konvexe Hüllalgorithmen, Flipping-Algorithmen, inkrementelle Einfügealgorithmen, Merging-Algorithmen

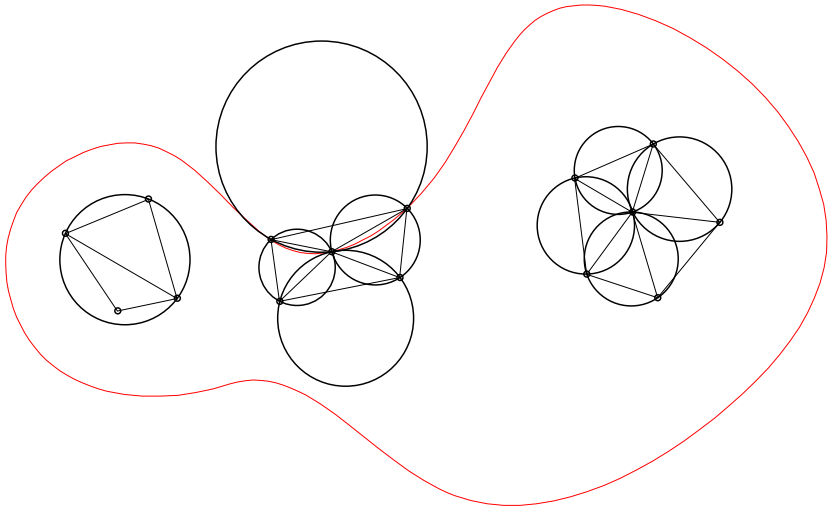

Abb. 4.3: Delaunay-Triangulisierung (links: Delaunay-Kriterium nicht erfüllt; mitte: Delaunay Kriterium erfüllt, Elemente außerhalb der Berandung; rechts: Delaunay-Kriterium erfüllt)

ähnlich dem in Abb. 4.4 dargestellten iterativen Prozess herangezogen.

4.4.3 Triangulisierung nach der *advancing front*-Methode

Kennzeichnendes Merkmal der *advancing front*-Methode ist, dass Elemente und Knoten zugleich erzeugt werden. Abb. 4.5 zeigt die grundlegende Vorgehensweise bei der Verwendung der *advancing front*-Methode [24].

Ausgehend von einer gegebenen Diskretisierung der geschlossenen Berandung werden sukzessive Segmente in die Berandung eingefügt bzw. kollabiert, wodurch neue Elemente erzeugt werden. Die Berandung umschließt dabei immer den noch nicht vernetzten Bereich, wächst also durch das Einfügen von Punkten und Elementen nach innen [25].

Es ist möglich, die Generierung der Punkte so vorzunehmen, dass eine optimale Anordnung gegeben ist. Allerdings sind bei der Ermittlung der optimalen Anordnung bereits erzeugte Punkte zu berücksichtigen. Weiterhin ist die Erzeugung neuer Punkte dann schwierig, wenn gegenüberliegende Bereiche der Berandung sich annähern. Hierbei kann es zu einer Aufteilung der Berandung in mehrere geschlossene Bereiche kommen [26]. Es können Konvergenzprobleme auftreten.

4.4.4 *quad/octree*-basierte Verfahren

Bei *quad/octree*-basierten Verfahren wird der zu vernetzende Bereich rekursiv in sog. *quads*, d. h. Rechtecke in 2D, bzw. Oktaeder in 3D, unterteilt, bis eine den Anforderungen

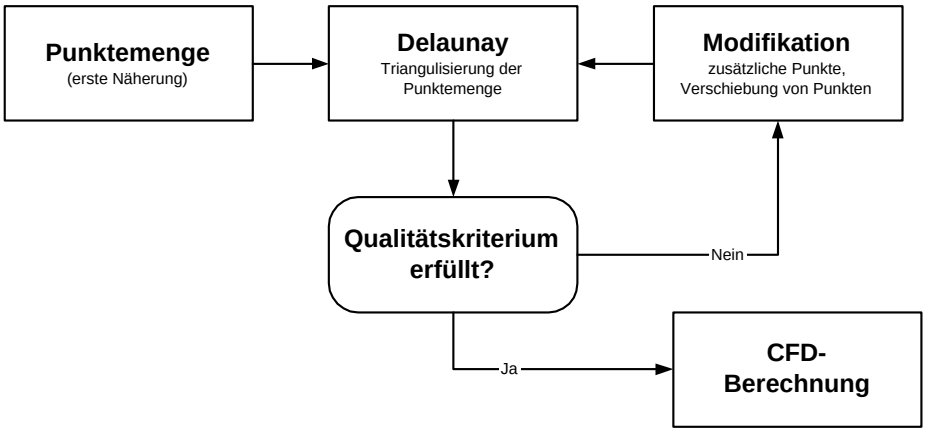

Abb. 4.4: Iterativer Prozess der Netzgenerierung mittels Delaunay-Triangulisierung

entsprechende Näherung an die vorhandenen Kontur erreicht ist. Durch Unterteilung der Teilregionen kann anschließend ein Gitter[16] erzeugt werden [27] [28].

Abb. 4.6 zeigt für den ebenen Fall die Unterteilung einer durch eine Randrepräsentation beschriebenen Fläche in *quads* sowie die anschließend erzeugte Diskretisierung der Berandung.

4.4.5 Bewertung

Zusammenfassend sind folgende Problempunkte bei der Verwendung oben genannter Verfahren zur Netzgenerierung zu nennen:

- Eine Delaunay-Triangulisierung ist nur für eine gegebe Punktemenge möglich. Diese ist separat zu erzeugen. Liegt die Form der zu vernetzenden Fläche nicht explizit vor[17], so kann nicht garantieren werden, dass die Berandung eingehalten wird.
- Bei einer Vernetzung nach der *advancing front*-Methode sind bereits erzeugte Panels zu berücksichtigen. Dies führt zu einem komplexen Algorithmus. Konvergenzprobleme bzw. schlecht konditionierte Gitter können in Bereichen auftreten, wo die durch die Front umschlossene, noch nicht vernetzte Fläche sehr klein ist. Diese müssen eventuell gesondert aufbereitet bzw. optimiert werden.
- Durch die bei *quad/octree*-basierten Verfahren durchgeführte sukzessive Unterteilung von Elementen mit nicht adäquater Auflösung ist ein deutlicher Größenunterschied benachbarter Elemente nicht auszuschließen. Es wird eines der in Abschnitt 4.2 als entscheidend für die Gitterqualität benannten Kriterien nicht erfüllt.

[16] In Bereichen mit konstanter Unterteilungsdichte wird ein strukturiertes Netz erzeugt. In Bereichen mit variierender Unterteilungsdichte ist ein unstrukturiertes Netz zu verzeichnen.

[17] Dies kann z. B. der Fall sein, falls die Punkteschar durch 3D-Scans etc. erzeugt wurde.

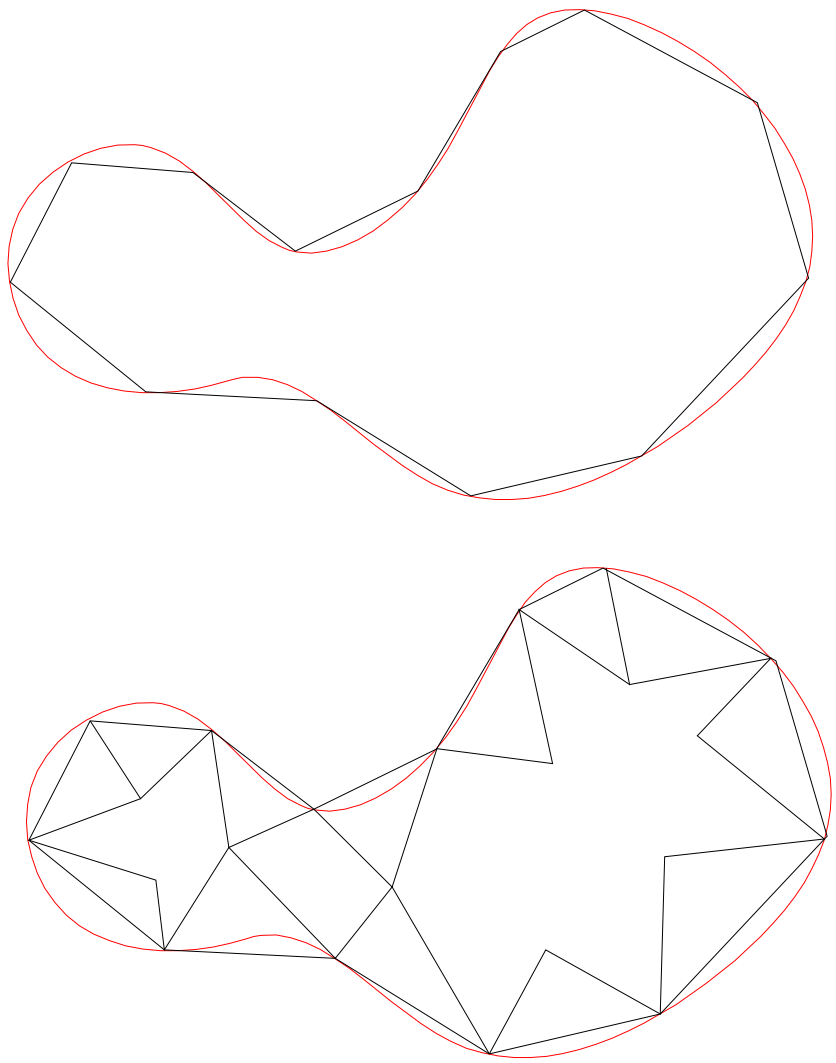

Abb. 4.5: Die *advancing front*-Methode (oben: Diskretisierung der geschlossenen Berandung; unten: Erzeugung neuer Elemente, Ausbildung von mehreren geschlossenen Bereichen)

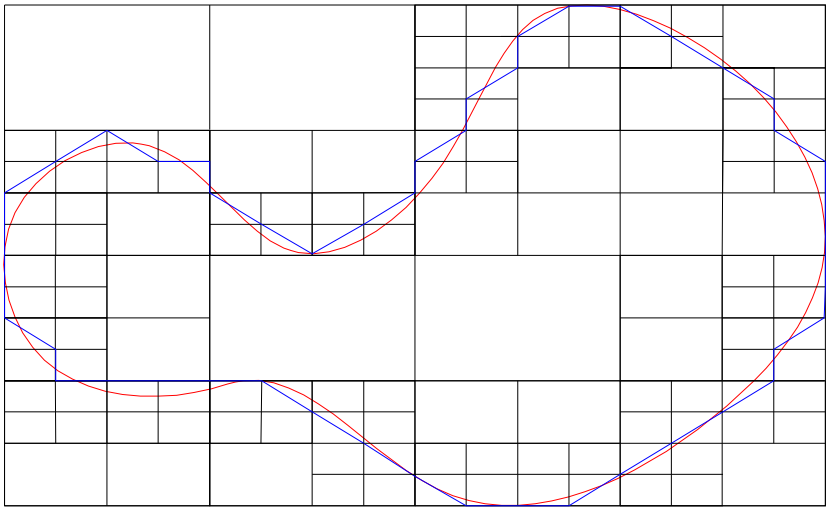

Abb. 4.6: Vernetzung mittels *quadtrees* (rot: Kontur der zu vernetzenden Fläche; schwarz: Analyse der Fläche durch *quads*; blau: diskretisierte Berandungskontur)

Es wird deutlich, dass kein Verfahren allein die Erzeugung von „schönen“, alle Anforderungen erfüllenden Netzen ermöglicht. Vielversprechend erscheint vielmehr eine Kombination dieser Verfahren bzw. die zusätzliche Verwendung z. B. eines Algorithmus zur Erzeugung und Beurteilung der Knoten und Elemente.

4.5 Netzoptimierung

Ist nach einem der in Abschnitt 4.4 beschriebenen Verfahren eine initiale Vernetzung erzeugt worden bzw. liegt ein auf anderem Wege gewonnenes Netz vor, so ist vielfach in einem zweiten Schritt eine Optimierung des Netzes hinsichtlich Netzdichte, Elementverteilung etc. erforderlich. Hierfür existieren zwei Verfahrensweisen, welche im folgenden beschrieben werden [29].

4.5.1 Knotenverschiebung

Entspricht die Dichte eines Netzes, und damit die Elementanzahl, bereits der gewünschten, so ist eine lokale Modifikation der Elemente durch die Verschiebung von einzelnen Knoten möglich. Dies bewirkt eine Formänderung aller Elemente, welche mit diesem Knoten verbunden sind.

Hierdurch ist eine Optimierung der Größenverhältnisse benachbarter Elemente sowie der Winkel innerhalb eines Panels möglich. Allerdings kann eine derartige Operation nur durchgeführt werden, wenn die Geometrie des zu vernetzenden Objektes auch außerhalb

der vorhandenen Elementknoten ermittelt werden kann. Stehen nur die Aufpunkte zur Verfügung, z. B. aus einem 3D-Scan, so kann nicht garantiert werden, dass ein modifizierter Knoten ebenfalls auf der Kontur eines Objektes angeordnet ist.

4.5.2 Variation der Elementanzahl

Wird eine Modifikation der Elementdichte und -anzahl angestrebt, so kann dies durch Einfügen bzw. Entfernen von Punkten, Kanten und Dreiecken erreicht werden. Im folgenden werden die möglichen Schritte für den Fall einer Reduzierung der Elementanzahl erläutert. Eine Netzverfeinerung kann auf umgekehrtem Wege erreicht werden.

Entfernen eines Punktes Wird zur Reduktion der Elementanzahl ein Knoten entfernt, so wird hierdurch Lage, Form und Anordnung aller mit diesem Knoten verbundenen Elemente beeinflusst, siehe Abb. 4.7.

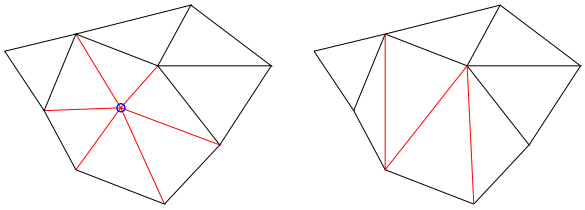

Abb. 4.7: Reduktion der Elementanzahl durch Entfernen eines Punktes (links: zu löschender Punkt (blau) und betroffene Kanten (rot); rechts: durch erneute Triangulisation erzeugte Kanten (rot))

Um weiterhin eine konsistente Triangulisierung, d. h. eine Vernetzung ohne Klaffungen und Überlappungen, beibehalten zu können, ist es erforderlich, alle betroffenen Dreiecke zu entfernen und das so entstandene Polygon erneut zu triangulisieren. Liegt der zu löschende Punkt auf der Berandung, so ist sicherzustellen, dass auch nach erneuter Triangulisation die Berandungstreue gesichert ist.

Kollabieren einer Kante In Abb. 4.8 ist die Vorgehensweise beim Kollabieren einer Kante dargestellt. Hierbei werden die Endpunkte einer Kante zu einem gemeinsamen Punkt zusammengefasst. Dreiecke, welche eine Kante mit der kollabierten Kante gemeinsam haben, werden gelöscht. Die Knotenkoordinaten von mit den Eckpunkten der kollabierten Kante verbundenen Elementen werden aktualisiert.

Liegt ein Punkt der zu kollabierenden Kante auf der Berandung, so ist, um die Berandung nicht zu verändern, darauf zu achten, dass als neuer gemeinsamer Punkt der auf der Berandung angeordnete Endpunkt der Kante gewählt wird.

Kollabieren eines Dreiecks Als dritte Variante zur Verringerung der Elementanzahl ist, wie in Abb. 4.9 gezeigt, die Entfernung eines Dreieckes möglich. Hierbei werden die

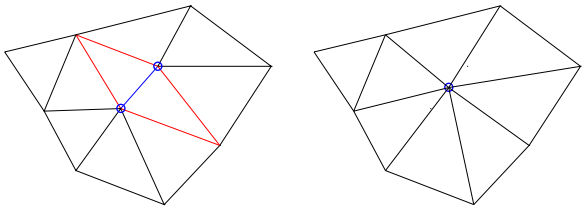

Abb. 4.8: Reduktion der Elementanzahl durch Kollabieren einer Kante (links: zu löschende Kante und betroffene Kantenendpunkte (blau), Kanten zu löschender Dreiecke (rot); rechts: Kante ist kollabiert, neu erzeugter Punkt (blau) und modifizierte Kanten (rot))

Eckpunkte des Dreieckselements zu einem gemeinsamen Punkt zusammengeführt; alle benachbarten Elemente werden entsprechend aktualisiert bzw., so sich eine Elementfläche von null ergibt, entfernt.

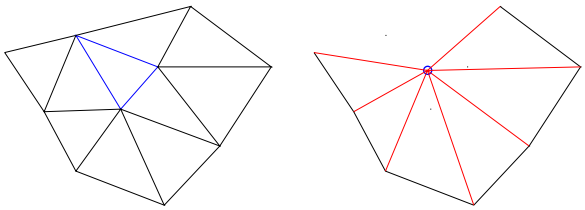

Abb. 4.9: Reduktion der Elementanzahl durch kollabieren eines Dreiecks (links: Dreieck, das kollabiert wird (blau); rechts: Punkt nach der Löschung des Dreiecks (blau) und modifizierte Kanten (rot). Es ist der Fall einer nicht korrekten Behandlung der Berandung dargestellt.)

Diese Variante führt zur stärksten Abnahme der Elementanzahl je Iterationsschritt. Schwierig ist hierbei allerdings die Wahl des für das Kollabieren herangezogenen Punktes, da zum einen ein „schönes“ Netz erzeugt bwz. beibehalten werden soll, zum anderen, da in mehreren Sonderfällen[18] eine spezielle Vorgehensweise bei der Wahl des Kollabierungspunktes erforderlich ist, um die Berandungstreue sicherstellen zu können.

[18]Zu nennen sind: ein Punkt auf Berandung; eine oder zwei Kanten auf Berandung

Kapitel 5

Der Vernetzungsalgorithmus

Aufbauend auf den in den Kapiteln 2, 3 und 4 dargelegten theoretischen Grundlagen und allgemeinen Verfahrensweisen zur Netzgenerierung wird in diesem Kapitel eine detailierte Beschreibung des im Rahmen dieser Arbeit entwickelten und implementierten Verfahrens gegeben.

Von den in Abschnitt 4.2 dargelegten Anforderungen an die zu erzeugenden Netze ausgehend, wird in Abschnitt 5.1 die gewünschte Funktionalität des Algorithmus beschrieben. In Abschnitt 5.2 werden betrachtete Algorithmen zur Netzerzeugung dargelegt und diskutiert. Eine ausführliche Beschreibung des gewählten und getesteten Algorithmus ist in Abschnitt 5.3 zu finden.

5.1 Zielsetzung und Problemdefinition

Gemäß Aufgabenstellung ist das Ziel dieser Diplomarbeit die Entwicklung eines weitestgehend automatischen Netzgenerierungsverfahrens für die Vernetzung von Schiffsoberflächen.

Als Ausgangsbasis wird eine Oberflächenbeschreibung des Schiffsrumpfes verwendet. Diese liegt als IGES-Datei vor. In dieser Datei ist die Oberfläche mathematisch exakt beschrieben durch eine oder mehrere NURBS-Flächen, siehe Abschnitt 3.6.

Unter Verwendung dieser Oberflächenbeschreibungen, den sog. *patches*, ist ein Verfahren zu entwickeln, das ohne zwingend erforderliche Benutzerinteraktion eine Panelierung der Rumpfoberfläche mit Hilfe von Dreieckselementen durchführt; es wird ein unstrukturiertes Gitter erzeugt. Entspricht das Ergebnis nicht den Erwartungen und Anforderungen, so sind dem Anwender Einflussmöglichkeiten zur Verfügung zu stellen, wodurch eine globale wie auch lokale Steuerung der Netzstruktur ermöglicht wird. Zeit- und arbeitsaufwendige Verfahren wie z. B. die nutzergesteuerte Erzeugung eines blockstrukturierten Gitters sind zu vermeiden.

Es zeigt sich, dass bei oben genannter Zielsetzung insbesondere zwei Aspekte besonderer Sorgfalt und Handhabung bedürfen.

5.1.1 Numerische Stabilität und Robustheit

Liegen ausschließlich fehlerfreie geometrische Beschreibungen von Rumpfoberflächen vor, so kann die Entwicklung und Implementierung eines Vernetzungsalgorithmus unter Annahme definierter Ausgangsbedingungen vorgenommen werden.

Erfahrungen aus der Praxis haben allerdings gezeigt, dass diese Annahme nicht immer erfüllt wird. So treten z. T. auch bei Verwendung von renommierten CAD-Systemen nach dem Geometrieexport Überlappungen von NURBS-Oberflächen auf; Klaffungen zwischen einzelnen Patches sind zu finden. Um die Nutzbarkeit des Algorithmus auch in diesen Fällen sicherzustellen, ist daher bei der Entwicklung darauf zu achten, dass auch bei fehlerbehafteten Eingabedaten eine erfolgreiche Netzgenerierung erreicht wird.

Weiterhin ist bei der Implementierung des entwickelten Verfahrens darauf zu achten, dass intern eventuell auftretende Rundungsfehler bzw. verwendete Approximationen keine störenden Einflüsse auf den Programmablauf haben. Es ist also eine ausreichend große numerische Stabilität zu garantieren.

5.1.2 Topologie

Die Geometriebeschreibung eines Schiffsrumpfes wird mittels NURBS-Flächen erreicht. Je nach Komplexität der Oberfläche und verwendetem Programmsystem wird die Rumpfoberfläche abgebildet durch einige wenige bis zu mehreren hundert NURBS-Flächenstücke.

Diese Patches können in beliebiger Reihenfolge und Anordnung definiert sein. Wie in Abb. 5.1 dargestellt, sind verschiedene Zusammenhänge benachbarter Patches möglich. Hieraus ergibt sich, dass eine fehlerfreie und eindeutige Bestimmung der Patch-Topologie, d. h. eine Identifikation von Nachbarschaftsrelationen, selbst für fehlerfrei definierte Nurbs-Patches nur sehr schwer durchgeführt werden kann. Wird oben angesprochener Aspekt der numerischen Robustheit ebenfalls berücksichtigt, d. h. es können Überlappungen und Klaffungen auftreten, so wird eine Topologieerkennung weiter erschwert.

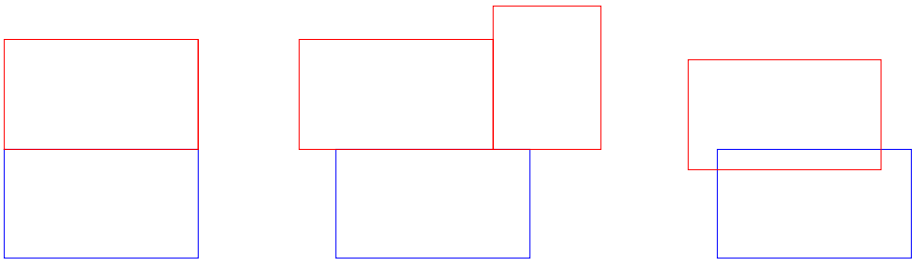

Abb. 5.1: Nachbarschaftsrelationen von NUBRS-Patches (links: Beide Patches haben genau eine identische Berandung; mitte: Die Berandung eines Patches ist mit zwei Nachbarflächen verbunden; rechts: Überlappung zweier Patches)

Diese Problematik ist ein Aspekt, der die Erzeugung eines Oberflächennetzes, welches die gesamte Schiffsoberfläche abdecken soll, also patchübergreifend generiert wird, sehr erschwert.

5.2 Betrachtete Verfahren

Da eine direkte Erzeugung eines optimalen Netzes mittels *advancing front*-Methode auf Grund der in Abschnitt 4.4 genannten Aspekte nur schwer erzielt werden kann, wurde der Fokus bei der Entwicklung eines Algorithmus auf die Erzeugung eines initialen Vernetzungszustandes mit anschließender Gitteroptimierung gelegt.

Hierbei kann die Erzeugung eines idealen Netzes als iterativer Prozess betrachtet werden. Aufbauend auf aus vorherigen Iterationsschritten gewonnenen Erkenntnissen wird die Netzstruktur durch Modifikation der Vernetzungsparameter bzw. Gitterstruktur variiert, bis ein Ergebnis ausreichender Güte erreicht wird. Denkbar sind hierfür zwei Vorgehensweisen.

5.2.1 *top down*-Verfahren

Bei einem *top down*-Verfahren wird das Netz, ausgehend von einer sehr groben Vernetzung, solange verfeinert, bis die gewünschte Genauigkeit der Oberflächenrepräsentation erreicht ist.

Für die Netzverfeinerung muss eine Möglichkeit gefunden werden, Punkte auf der Oberfläche derart zu plazieren, dass die weiteren in Abschnitt 4.2 genannten Anforderungen erfüllt werden. Dies gestaltet sich insbesondere bei der Erzeugung zusätzlicher Punkte und Elemente im Bereich der Patch-Grenzen wegen der nicht eindeutig bestimmbaren Topologie, siehe Abschnitt 5.1.2, als schwierig.

5.2.2 *bottom up*-Verfahren

Bottom up-Verfahren zeichen sich dadurch aus, dass bei einer sehr feinen initialen Vernetzung sukzessive die Zahl der Elemente reduziert wird, bis wiederum die geforderte Netzstruktur und -dichte erzielt wird.

Zwar besteht, wie auch beim *top down*-Verfahren, in diesem Fall die Problematik, eine patchübergreifende anfängliche Vernetzung zu erzeugen; bei den folgenden Schritten der Reduktion der Elementanzahl ist es allerdings nicht erforderlich, weitere, nur patchweise verfügbare Informationen über die Gestalt der Rumpfoberfläche zu verwenden.

5.2.3 Bewertung

Sowohl bei *top down*- wie auch bei *bottom up*-Verfahren besteht die Notwendigkeit, ein konsistentes Ausgangsnetz zu erzeugen, welches die gesamte Rumpfoberfläche beinhaltet. Der Vorteil von *bottom up*-Verfahren, nämlich die weitere Netzoptimierung ohne Rückgriff auf die Oberflächenbeschreibung durch NURBS-Flächen, ist Grund dafür, dass für den im folgenden Abschnitt vorgestellten Vernetzungsalgorithmus ein *bottom up*-Verfahren herangezogen wird.

5.3 Der Vernetzungsalgorithmus

Abschnitt 5.3.1 gibt einen Überblick über den Ablauf einer Netzgenerierung. Detailierte Beschreibungen der entwickelten relevanten Verfahrensweisen werden in den Abschnitten 5.3.2 bis 5.3.4 dargelegt.

5.3.1 Überblick

Das Verfahren zur Panelgenerierung kann in drei Teile gegliedert werden. Dies ist in Abb. 5.2 dargestellt. Sind Informationen durch den Anwender zu liefern, so sind diese als rote Pfeile aufgeführt.

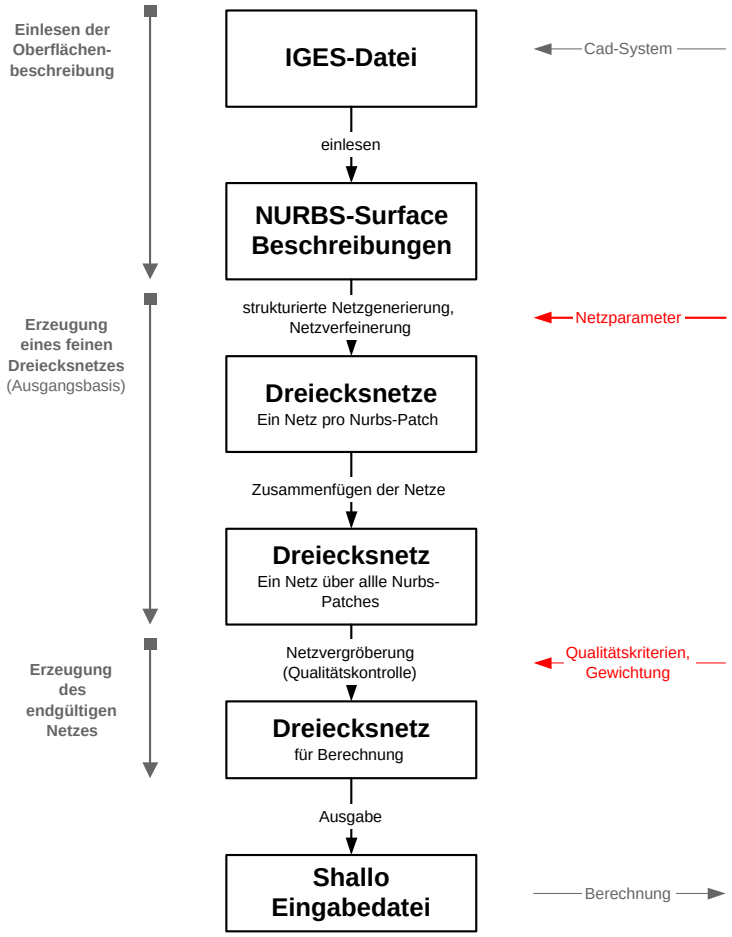

Abb. 5.2: Der Vernetzungsalgorithmus im Überblick

In einem ersten Schritt werden die in der IGES-Datei definierten Oberflächen gelesen und ausgewertet. Diese stehen anschließend zur weiteren Verarbeitung als NURBS-Oberflächen zur Verfügung. Diese Funktionalität wurde im Rahmen einer früheren Diplomarbeit am

Lehrstuhl entwickelt und durch den Autor, wenn auch mit geringfügigen Modifikationen[1], übernommen. Interessierte seien an dieser Stelle auf [15] verwiesen.

Anschließend erfolgt die Erzeugung einer Triangulierung je NURBS-Patch. Hierfür wird, von einer strukturierten Vernetzung ausgehend, eine adaptive Netzverfeinerung in Abhängigkeit der Oberflächeneigenschaften durchgeführt. Die einzelnen Gitter werden dann zu einem konsistenten, die ganze Rumpfoberfläche umfassenden Netz zusammengefügt. Dieses feine Gitter bestehend aus Dreieckselementen dient als Ausgangsbasis für die im nächsten Schritt durchgeführte Reduktion der Anzahl an Elementen.

Als Kernpunkt der Netzgenerierung wird letztlich eine Netzvergröberung durchgeführt mit dem Ziel, aus dem feinen Ausgangsnetz ein Gitter zu erzeugen, welches die in Kapitel 4.2 genannten Anforderungen an ein Netz für potentialtheoretisch basierte Berechnungen erfüllt und eine geeignete Anzahl an Elementen aufweist. Dem Anwender stehen hierbei globale wie auch lokale Modifikationsmöglichkeiten zur Verfügung, durch welche die Struktur des Gitters geeignet beeinflusst werden kann.

Eine visuelle Darstellung der Ergebnisse jedes Einzelschrittes wie auch des gesamten Vernetzungsalgorithmus ist in Kapitel 7 zu finden.

5.3.2 Erzeugung der Ausgangsvernetzung

Wie im vorhergehenden Abschnitt erläutert wird eine Ausgangsvernetzung durch eine je Patch durchgeführte strukturierte Gittererzeugung gekoppelt mit adaptiver Netzverfeinerung sowie anschließendem *merging*, d. h. einem Zusammenführen der einzelnen Netze zu einem konsistenten Gitter, erzeugt. Dies ist in Abb. 5.3 graphisch dargestellt. Durch den Anwender vorzugebende Einstellungen sind rot gezeichnet. Graue Pfeile stellen Ein- bzw. Ausgabedaten dar.

5.3.2.1 Strukturierte Vernetzung

Zweck und Ziel einer anfänglichen strukturierten Gittergenerierung ist die Erstellung einer Elementierung, welche sich auch nach der anschließenden Netzverfeinerung durch ein gewisses Maß an Homogenität, d. h. durch eine bedingt regelmäßige Anordnung von Elementen und Knoten, auszeichnet. Nach Erfahrungen des Autors ist dies der Erzeugung eines „schönen“ Gitters förderlich. Weiterhin wird vermieden, dass in flächigen Abschnitten wie z. B. im Bereich des parallelen Mittelschiffes eine zu geringe Elementdichte vorliegt.

Wird durch den Anwender eine mittlere Länge l_{mean} der Elementkanten auf der Patch-Berandung vorgegeben, so ergibt sich hieraus die Anzahl der Elemente n_u, n_v je Parameterrichtung u, v wie folgt:

$$n_u = \frac{l_u}{l_{mean}}, \quad n_v = \frac{l_v}{l_{mean}} \qquad mit \quad n_u, n_v \epsilon N \tag{5.1}$$

[1] Hierunter fallen hauptsächlich Code-Cleanup, Refactoring, javadoc-konforme Dokumentation sowie Adaption an die veränderten Schnittstellen.

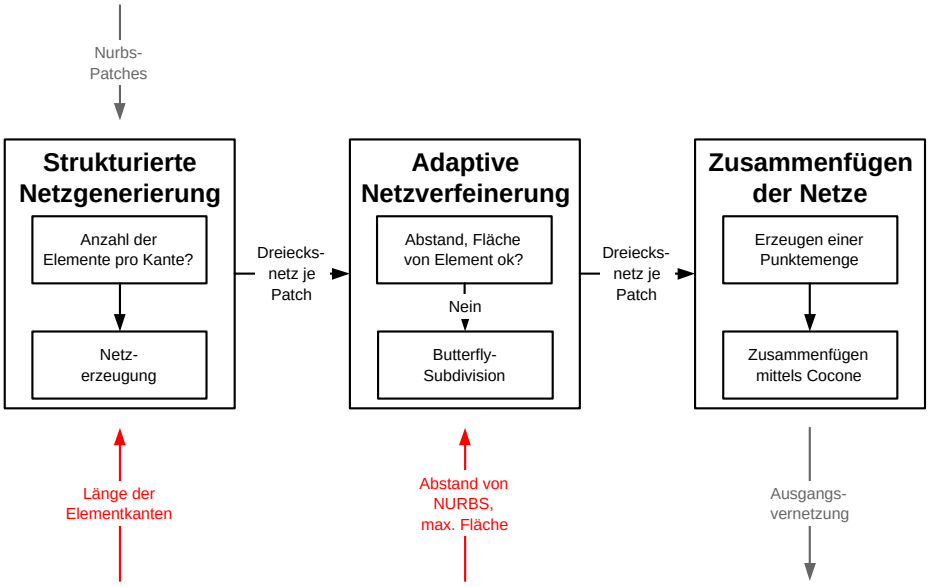

Abb. 5.3: Erzeugung der Ausgangsvernetzung

Mit l_u und l_v werden die mittlere Länge der Patchberandungen in den Parameterrichtungen u und v beschrieben. Diese werden durch stückweise lineare Approximation[2] berechnet.

Die Knoten der für das Gitter erzeugten Dreieckselemente werden gemäß dem durch n_u und n_v aufgespannten Raster erzeugt, so dass insgesamt $n_{Quad} = n_u \cdot n_v$ Viereckselemente gebildet werden. Eine Umwandlung der Viereckselemente in $n_{Trgl} = 2 \cdot n_{Quad}$ Dreieckselemente wird durch eine Aufspaltung der Viereckselemente entlang der kürzeren Diagonale erreicht.

Wird ein dreieckiges NURBS-Patch vernetzt, d. h. die Länge einer Berandungskante ist identisch Null, so werden degenerierte Dreieckselemente[3] erzeugt, welche zu entfernen sind.

5.3.2.2 Adaptive Netzverfeinerung

Insbesondere in Bereichen mit starker Oberflächenkrümmung ist eine ausreichend genaue Approximation der Oberflächenkontur durch strukturierte Vernetzung nur möglich, falls auch für Bereiche geringer Krümmung eine sehr hohe Elementdichte gewählt wird. Dies führt zu einer unerwünscht hohen Elementanzahl und ist daher nicht empfehlenswert.

Bei einer adaptiven Netzverfeinerung werden nur Bereiche verfeinert, für welche auf Grund der Geometrieeigenschaften bzw. anderer, durch den Benutzer vorgegebener Bedingun-

[2] Es wurde festgestellt, dass durch eine Approximation mit 20 Liniensegmenten eine ausreichend genaue Länge der Berandung ermittelt wird.

[3] Diese bilden eine Linie, d. h. zwei Eckpunkte des Dreiecks sind identisch.

gen eine zu geringe Elementdichte ermittelt wird. Im hier vorgestellten Algorithmus stehen dem Anwender folgende Parameter zur Beeinflussung des Vernetzungsprozesses zur Verfügung:

- Ist die Fläche eines Elementes größer als maximal zulässig, so wird dieses Element geteilt. Hierdurch wird vermieden, dass unverhältnismäßig große Elemente enstehen, welche die in Abschnitt 5.3.3 erläuterte Netzoptimierung stören können.
- Ist der Abstand zwischen mindestens einem Mittelpunkt einer Kante des betrachteten Elements zur NURBS-Fläche größer als durch den Anwender erlaubt, d. h. die Approximation der Oberfläche ist zu ungenau, so wird für dieses Element eine Netzverfeinerung durchgeführt[4].

Da die Erzeugung eines konsistenten Netzes für die anschließende Netzoptimierung erforderlich ist, kommt der in [15] entwickelte modifizierte *butterfly subdivision*-Algorithmus zur Anwendung. Wird ein Dreieck in vier Dreiecke unterteilt, wobei neue Knoten mittig auf den Kanten des Elementes erzeugt werden, so werden angrenzende Elemente ebenfalls entsprechend der neu erzeugten Knoten unterteilt, siehe Abb. 5.4, und so die geforderte Konsistenz garantiert [30]. Der Algorithmus wurde modifiziert, um, im Gegensatz zu herkömmlichen *butterfly subdivision*-Algorithmen, sicherzustellen, dass die Berandung erhalten bleibt.

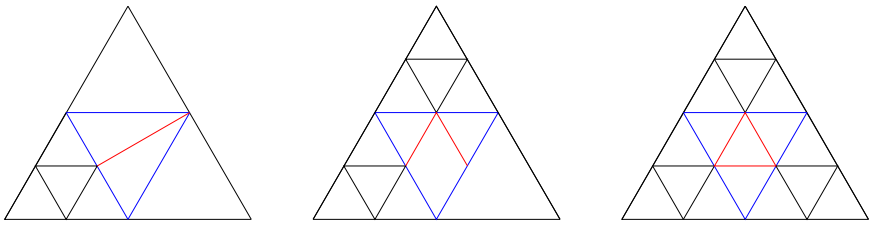

Abb. 5.4: Methodik der *butterfly subdivision* (Dargestellt ist das betrachtete Dreieck (blau) sowie die entsprechend der Unterteilung von Nachbarelementen erzeugte Unterteilung (rot) dieses Dreiecks.)

Als Pseudo-Code geschrieben, kann der implementierte Algorithmus beschrieben werden wie folgt:

```
Gegeben: Ein strukturiertes Dreiecksnetz
Gesucht: Ein adaptiv verfeinertes Dreiecksnetz

while mindestens 1 Dreieck zu gross
  for jedes Dreieck
    teste Dreieck auf Flaeche, Abstand
    if Dreieck zu gross und nicht gesperrt
       teile Dreieck in 4 Teildreiecke
       sperre Nachbardreiecke
    endif
  endfor
```

[4]Die praktische Erfahrung hat gezeigt, dass eine rechenzeitinsensivere Ermittlung des Abstands zwischen dem Schwerpunkt eines Dreieckselementes und dem korrespondierenden Punkt auf dem NURBS-Patch keine erkennbaren Vorteile mit sich bringt.

```
9:   for jedes Dreieck
10:     if Dreieck gesperrt
11:       finde Anzahl geteilter Nachbardreiecke
12:       fuehre geeignete butterfly subdivision durch
13:       entsperre Dreieck
14:     endif
15:  endfor
16: endwhile
```

Wird der oben beschriebe Algorithmus verglichen mit etwa einer rekursiven Lösung, so ist dieser unter dem Aspekt der Ausführungsgeschwindigkeit nicht optimal. Allerdings kann auf diesem Wege die Konsistenz des Netzes auf einfache Weise garantiert werden. Auf dem Entwicklungs- und Testsystem[5] des Autors wurden akzeptable Bearbeitungszeiten gemessen.

5.3.2.3 Zusammenfügen der Dreiecksnetze

Wie in Abschnitt 5.1.2 erläutert ist ein Zusammenfügen der Dreiecksnetze zu einem Netz, das sog. *merging*, auf Grund der nicht eindeutig und fehlerfrei bestimmbaren Topologie sehr schwierig.

Surface reconstruction Aus diesem Grund wie auch auf Grund der beschränkten Bearbeitungsdauer wurde eine Lösung gewählt, bei welcher die aus dem vorigen Schritt innerhalb eines NURBS-Patches verfügbaren Nachbarschafts- und Berandungsinformationen nicht genutzt werden. Stattdessen werden ausschließlich die im vorigen Schritt erzeugten Knoten verwendet.

Unter Verwendung der Koordinaten dieser Knoten wird ein sog. *surface reconstruction* Algorithmus angewendet. Hierunter ist ein Algorithmus zu verstehen, der, ausgehend von einer Punktewolke, eine Rekonstruktion der Oberfläche berechnet, indem eine Delaunay-konforme Triangulisierung dieser Punktemenge erzeugt wird. Durch eine vorgeschaltete Analyse der Punkte kann, so eine ausreichend günstige Punkteverteilung vorliegt, eine Identifikation von Knicken innerhalb der Oberfläche sowie der Berandung erreicht werden.

Im Rahmen dieser Arbeit kommt der *surface reconstruction* Algorithmus *cocone*[6] der University of Ohio[7] zur Anwendung und wird als externe Komponente eingebunden. *Cocone* ist in der Lage, bedingte Delaunay-Triangulisierungen von Punktemengen durchzuführen, d. h. eine Identifikation der Oberflächenberandung durchzuführen. Knicke werden automatisch erkannt und das Netz entsprechend erzeugt. Für eine Darstellung der theoretischen Basis dieses Algorithmus siehe [31], [32] und [33].

Das Merging lässt sich wiederum in Form von Pseudo-Code folgendermaßen beschreiben:

```
Gegeben: Ein Dreiecksnetz je NURBS–Patch
Gesucht: Ein Dreiecksnetz ueber die gesamte Aussenhaut
```

[5] Es kommt ein zu heute gängigen Systemen vergleichsweise langsames System mit folgenden Daten zum Einsatz: Duron 700, 384 MB RAM, Linux, Kernel 2.4.19. JDK 1.3.1

[6] Dieser ist für nicht kommerzielle Anwendungen frei und online verfügbar.

[7] Zu finden im Web unter: http://www.cis.ohio-state.edu/~tamaldey/

```
1: for alle Dreiecksnetze
2:   Extrahiere Knoten als Punktemenge P_i
3: endfor
4: fuege alle Punktemengen P_i zusammen zu P
5: schreibe Koordinaten der Knoten in P in Datei
6: starte cocone
7: lese generiertes Dreiecksnetz aus Datei
```

Im praktischen Einsatz dieser Lösung mußte festgestellt werden, dass insbesondere in Bereichen mit komplexer Kontur der Berandung, z. B. bei deutlich ausgeprägtem Bugwulst und stark ausfallendem Vorsteven, die Berandung nicht immer korrekt erkannt wird. Durch Wahl einer geeigneten Ausgangsvernetzung konnten derartige Artefakte zwar minimiert, aber nicht ausgeschlossen werden. Im Mittschiffsbereich werden durch cocone z. T. Punkte nicht bei der Triangulisierung berücksichtigt; es werden Dreiecke mit zu großer Fläche erzeugt. Auch hier konnte durch eine Erhöhung der Strukturierung des Gitters[8] Abhilfe geschaffen werden, wenn auch zu Lasten einer für die Netzoptimierung nicht erforderliche Erhöhung der Anzahl an Panels.

Merging durch Topologieerkennung Da der oben beschriebene *surface reconstruction*-Algorithmus *cocone* nicht immer zu zufriedenstellenden Ergebnissen führte, wurde durch den Autor versucht, einen Algorithmus zum Zusammenführen der je NURBS-Oberfläche erzeugten Dreiecksnetze zu entwickeln.

Aus Gründen der hohen Komplexität dieser Problematik wie auch der beschränkten Bearbeitungszeit konnte diese Entwicklung nicht zu Ende geführt werden; es existiert ein Prototyp, der eingeschränkt funktionsfähig ist. Der Stand der Entwicklung zum Zeitpunkt der Abgabe dieser Arbeit ist im Anhang in Kapitel A dargelegt.

5.3.3 Netzoptimierung

Kernpunkt des im Rahmen dieser Diplomarbeit entwickelten Verfahrens ist die in diesem Abschnitt beschriebene Netzoptimierung. Es werden in Abschnitt 5.3.3.1 Gründe für die Wahl des eingesetzten Verfahrens zur Netzvergröberung genannt. Die für die qualitätsbasierte Netzoptimierung herangezogenen Qualitätskriterien werden in Abschnitt 5.3.3.2 erläutert. Abschnitt 5.3.3.3 beschreibt implementierte Abbruchkriterien.

5.3.3.1 Verfahren zur Netzvergröberung

Als Verfahren zur Reduktion der Elementanzahl wird der sog. *edge collapse*, siehe Abschnitt 4.5.2, eingesetzt. Hierdurch entfällt im Gegensatz zur Methodik der Punktlöschung die Notwendigkeit einer erneuten Triangulisierung des modifizierten Bereichs. Im Gegensatz zur Kollabierung eines Dreiecks zu einem neuen gemeinsamen Knoten ist eine feinere Beeinflussung der Elementanzahl und -struktur möglich. Da beim *edge collapse* weniger

[8] Zu erreichen ist dies z. B. durch eine deutlich feinere anfängliche strukturierte Vernetzung, siehe Abschnitt 5.3.2.1.

zu betrachtende Fälle bei der Wahl des Kollabierungspunktes als beim *triangle collapse* auftreten können, ist die Ermittlung eines neuen Knotens einfacher zu bewerkstelligen.

Ein Überblick über die entwickelte Methodik der Netzvergröberung ist in Abb. 5.5 zu finden.

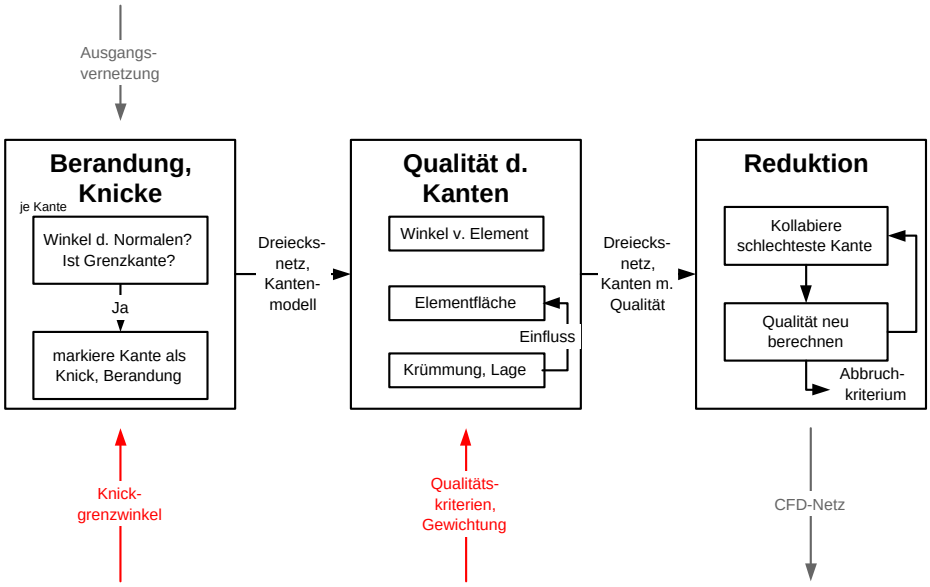

Abb. 5.5: Verfahren zur Netzoptimierung durch Reduktion der Elementanzahl

Von einer feinen Vernetzung ausgehend wird in einem ersten Schritt aus der internen Datenstruktur ein Kantenmodell berechnet. Alle Elementkanten werden analysiert. Ist eine Kante nur in einem Element enthalten, so ist diese Kante Teil der Berandung des Dreiecksnetzes. Kanten innerhalb des Gitters werden in zwei Elementen verwendet. Die Knoten von Berandungskanten werden entsprechend markiert, so dass diese bei der Netzoptimierung gesondert behandelt werden können.

Wird durch den Anwender ein Grenzwinkel vorgegeben, so werden weiterhin Kanten gesucht, die entlang eines Knicks auf der Rumpfoberfläche verlaufen. Ist der Winkel zwischen den Normalen der mit einer Kante verbundenen Elemente größer als der vorgegebene Grenzwinkel, so wird diese Kante als Knickkante betrachtet; die Knoten dieser Kante werden geeignet markiert.

Die Auswahl der zu kollabierenden Kanten erfolgt qualitätsbasiert, d. h. es werden diejenigen Kanten zuerst gelöscht, welche die Gitterqualität am stärksten negativ beeinflussen. Eine Beschreibung der Qualitätskriterien ist in Abschnitt 5.3.3.2 zu finden.

Beim Entfernen einer Kante, d. h. für das Zusammenführen der beiden Endpunkte dieser Kante zu einem gemeinsamen Punkt, sind drei Fälle zu unterscheiden, die unterschiedlich zu handhaben sind, siehe Abb. 5.6.

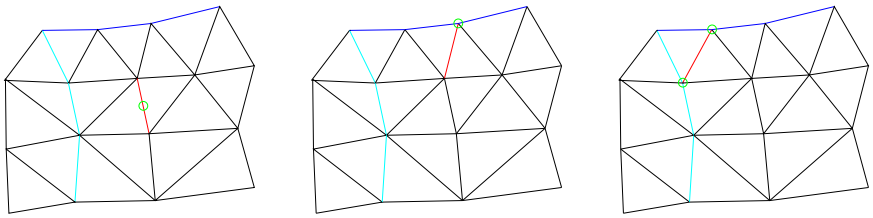

Abb. 5.6: Fallunterscheidung für den *edge collapse* (Es wird ein Netz mit Berandung (blau) und Knick (hellblau) dargestellt. links: Kante (rot) innerhalb des Gitters; mitte: ein Endpunkt der Kante auf Berandung; rechts: ein Endpunkt der Kante auf Berandung, der andere Punkt auf Knicklinie)

1. Haben beide Punkte identische topologische Eigenschaften, so wird als gemeinsamer Punkt der Mittelpunkt der entsprechenden Kante gewählt. Die Eigenschaft identischer topologischer Eigenschaften ist erfüllt, wenn beide Punkte auf der Berandung bzw. beide Punkte auf einem Knick angeordnet sind sowie für den Fall, dass die Knoten im Inneren des Netzes, aber nicht auf einem Knick, liegen.

2. Ist ein Knoten ein Berandungspunkt, während der zweite Punkt innerhalb des Gitters, aber nicht auf einem Knick angeordnet ist, so ist als neuer gemeinsamer Punkt der Berandungspunkt zu wählen.

3. Keine Kollabierung ist möglich, so ein Punkt auf der Berandung, der zweite Punkt auf einer Knicklinie angeordnet ist, da in diesem Fall die Bestimmung eines gemeinsamen Punktes zwingend dazu führt, dass der Verlauf von Berandung oder Knick modifiziert wird.

Ist der gemeinsame Punkt erzeugt, so werden die beiden mit der nun gelöschten Kante verbundenen Dreieckselemente gelöscht. Die Koordinaten der Eckpunkte, welche durch den *edge collapse* modifiziert wurden, werden angepasst; die elementweise gespeicherten Informationen werden hierzu entsprechend geändert. Für geänderte Kanten wird die Qualität neu berechnet. Ein Abbruch der Netzvergröberung erfolgt, so ein durch den Nutzer definiertes Kriterium erfüllt ist. Dies kann z. B. eine minimal zulässige Anzahl an Dreiecken bzw. eine vorgegebe mittlere Qualität der Kanten sein.

In Form von Pseudo-Code läßt sich der Algorithmus folgendermaßen beschreiben:

```
Gegeben: Fein vernetzes Ausgangsgitter
Gesucht: Netz fuer CFD–Berechnung

for alle Kanten
  berechne Qualitaet der Kante
  if Kante ist auf Berandung
    markiere Punkte der Kante: isOnBorder
  endif
  if Kante ist auf Knick
    markiere Punkte der Kante: isOnKnuckle
  endif
while Abbruchkriterium nicht erfuellt
  finde schlechteste Kante
  if Kantenpunkte p1, p2 ident. Eigenschaften
```

```
12:    neuer Punkt p als Mittelpunkt von p1, p2
13:  endif
14:  if ein Kantenpunkt auf Berandung oder auf Knick
15:   p entspricht diesem Kantenpunkt
16:  endif
17:  if p1 auf Berandung, p2 auf Knick oder umgekehrt
18:   continue
19:  endif
20:  loesche Dreiecke mit kollabierter Kante
21:  Ersetze p1, p2 in umliegenden Dreiecken durch p
22:  berechne Qualiaet aller mit p verbundenen Kanten neu
23: endwhile
```

5.3.3.2 Qualitätskriterien

Die Wahl der Kanten für den *edge collapse* erfolgt qualitätsbasiert, d. h. es wird die Kante mit schlechtester Qualität kollabiert, so diese nicht dem im vorigen Abschnitt aufgeführten Fall 3 entspricht und ein *edge collapse* daher nicht möglich ist.

Vor der Kollabierung der Kante wird nicht geprüft, ob diese Aktion zu einer Verbesserung der Gesamtqualität des Gitters führt. Untersuchungen haben aber gezeigt, dass, zumindest bei der im Rahmen dieser Arbeit angewendeten Methodik der Qualitätsberechnung, dies immer der Fall ist.

Hierfür wird ein Qualitätsmodell verwendet, das jeder Kante einen Qualitätsindex q, $q \geq 0$, zuweist. Je größer der Qualitätsindex q, desto geringer ist die Qualität der Kante; $q = 0$ entspricht einer Kante, welche alle gestellten Anforderungen vollständig erfüllt.

Der Qualitätsindex einer Kante berechnet sich aus n verschiedenen Kriterien wie folgt:

$$q = \sum_{i=0}^{n} \frac{w_i \cdot q_i}{n} \tag{5.2}$$

Eine Gewichtung des Einflusses einzelner Qualitätskriterien kann durch die Wichtungsfaktoren w_i erreicht werden; q_i bezeichnet die Qualität der Kante für ein Kriterium.

Im folgenden wird ein Überblick über die im Rahmen der Diplomarbeit entwickelten und implementierten Kriterien gegeben.

Seitenverhältnis Ein Kennzeichen von „schönen“ Netzen ist eine hohe Gleichmäßigkeit der Elementierung. Kommen Dreieckselemente zum Einsatz, so kann dies erreicht werden, indem die Ausbildung von gleichseitigen oder nahezu gleichseitigen Dreieckselementen gefördert wird. Nicht gleichseitige Dreiecke werden bevorzugt entfernt.

Wird ein Gitter durch derart angeordnete Elemente gebildet, dass der Unterschied der Länge der Elementkanten minimiert wird, so wird neben der oben erwähnten homogenen Struktur des Gitters ebenfalls die Forderung nach einem Flächenverhältnis benachbarter Elemente nahe Eins erfüllt. Weiterhin wird die Ausbildung sehr spitzer oder stumpfer Winkel unterbunden, so dass degenerierte Dreieckselemente nicht bzw. nur selten auftreten.

Die kantenbezogene Qualität für dieses Kriterium berechnet sich aus den Seitenverhältnissen der n_{NB} Nachbarelemente.

$$q_{AR} = \frac{1}{n_{NB}} \sum_{i=0}^{n_{NB}} q_{AR_i} \tag{5.3}$$

Mit den in Abb. 5.7 dargestellten Längenabmessungen wird für jedes Nachbarelement i dessen Qualität q_{AR_i} berechnet wie folgt:

$$q_{AR_i} = \frac{(e_{i1} - e) + (e_{i2} - e)}{e} \tag{5.4}$$

Die Normierung mit der Länge der betrachteten Kante wird durchgeführt, um zu er-

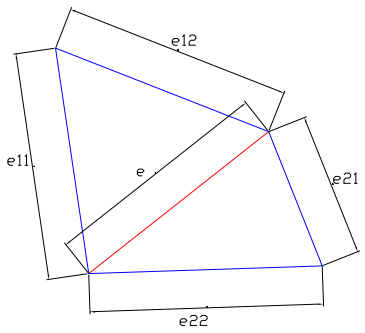

Abb. 5.7: Abmessungen der an eine Kante angrenzenden Elemente

reichen, dass ausschließlich die Längenverhältnisse der Kanten betrachtet werden. Ein Einfluss durch die Größe der an die Kante angrenzenden Elemente wird hierdurch ausgeschlossen.

Wird für ein Nachbarelement $q_{AR_i} < 0$ berechnet, d. h. ein stumpfwinkeliges Element liegt vor, so wird dies modifiziert gemäß:

$$q_{AR_i} = -\frac{1}{2} \cdot q_{AR_i} \tag{5.5}$$

Krümmung der Oberfläche Um eine gute Approximation der Schiffsaußenhaut zu erreichen, ist es erforderlich, in Bereichen mit großer Krümmung eine höhere Netzdichte zu wählen. Wie in Abschnitt 5.3.2 beschrieben besteht bei der Erzeugung des für die Netzoptimierung verwendeten Ausgangsgitters die Möglichkeit einer derartigen adaptiven Netzverfeinerung.

Um eine Steuerung der Netzdichte entsprechend der Krümmung auch bei der Reduktion der Elementanzahl zu erzielen, wird die Elementgröße durch die Krümmung beeinflusst.

Hierzu wird, ausgehend von der durch den Nutzer vorgegebenen minimal zulässigen Anzahl an Elementen n_Δ, die mittlere Elementfläche A_Δ berechnet nach

$$A_\Delta = \frac{A_{Hull}}{n_\Delta} \tag{5.6}$$

, wobei mit A_{Hull} die Gesamtfläche der Schiffsaußenhaut bezeichnet wird.

Da eine direkte Bestimmung der Krümmung in den Kantenendpunkten mit Hilfe der Oberflächenbeschreibung in Form von NURBS-Flächen auf Grund der in Abschnitt 5.1.2 beschriebenen Gründe nicht möglich ist, wird stattdessen die Krümmung aus den n mit einem Kantenendpunkt verbundenen Dreieckselementen ermittelt. Diese wird berechnet aus einem mittleren Normalenvektor n_{mean} sowie der gemittelten Summe der Winkel aller Dreiecksnormalen zu diesem Normalenvektor.

Werden die Dreiecksnormalen bezeichnet mit $\vec{n}_i$, so ergibt sich der Normalenvektor nach:

$$\vec{n}_{mean} = \sum_{i=1}^{n} \vec{n}_i \tag{5.7}$$

Hierbei wird berücksichtigt, dass alle Normalenvektor in die „gleiche“ Richtung zeigen, da ansonsten eine Berechnung von $\vec{n}_{mean}$ durch dieses Verfahren zu falschen Ergebnissen führen könnte.

Ausgedrückt durch Winkelabmessungen, symbolisiert durch $\angle$, läßt sich die Krümmung k beschreiben wie folgt:

$$k = \frac{1}{n} \sum_{i=0}^{n} \angle \left(\vec{n}_{mean}, \vec{n}_i \right) \tag{5.8}$$

Die Qualität einer Kante k_K in Hinblick auf die Krümmung wird im folgenden ermittelt aus den Krümmungswerten k_1 und k_2 der beiden Endpunkte p_1 und p_2.

$$k_K = \frac{k_1 + k_2}{2} \tag{5.9}$$

Ist $k_K \neq 0$, so wird aus der für oben ermittelten Elementfläche eine gewünschte Elementfläche A_{Δ_w} ermittelt.

$$A_{\Delta_w} = \frac{A_\Delta}{k_K + 1} \tag{5.10}$$

Im Rahmen der Netzvergröberung wird hieraus ein Qualitätskriterium einer Kante bezogen auf die Flächen benachbarter Elemente unter Berücksichtigung der Oberflächenkrümmung gebildet.

$$q_A = \frac{A_{\Delta_w}}{A_\Delta} - 1 \tag{5.11}$$

Das in diesem Abschnitt beschriebene Verfahren zur Berücksichtigung der Oberflächenkrümmung bei der Netzvergröberung eignet sich, um eine den Oberflächeneigenschaften entsprechende Verteilung der Elementdichte zu erzielen. Praktische Erfahrungen haben gezeigt, dass allerdings das hierfür eingesetzte Verfahren zu Ermittlung einer die Krümmung beschreibenden Kennzahl insbesondere für Randkanten nicht immer einwandfrei arbeitet,

da für diese Kanten nur eine geringe Anzahl von Nachbardreiecken zur Verfügung steht. Wird die Krümmung in den Endpunkten der Kanten direkt aus der NURBS-Beschreibung ermittelt, so ist diese Problematik nicht gegeben. Auf Grund der begrenzten Bearbeitungszeit sowie der ungeklärten Topologieproblematik konnte dies nicht umgesetzt werden.

Beeinflussung der Elementgröße durch Anordnung und Lage Vielfach besteht von Anwenderseite aus der Wunsch, die Netzstruktur nicht nur durch globale Parameter, sondern ebenfalls lokal begrenzt beeinflussen zu können. Dies kann durch zwei Möglichkeiten realisiert werden, nämlich

- an Hand der Lage der Kante
- mittels einer Quellenverteilung

Für eine Modifikation der erwünschten Elementfläche auf Grund der räumlichen Anordnung einer Kante werden durch den Benutzer für die Koordinatenrichtungen x, y, z die Einflussfunktionen $f_x(x)$, $f_y(y)$ und $f_z(z)$ vorgegeben. Gebildet mit dem Mittelpunkt $\vec{p}_M = (x_M, y_M, z_M)^T$ einer Kante wird hieraus der Modifikationsfaktor für diese Kante gebildet.

$$f(\vec{p}_M) = f_x(x_M) \cdot f_y(y_M) \cdot f_z(z_M) \tag{5.12}$$

Ist für $\vec{p}_M$ eine der Funktionen nicht definiert, so wird als Funktionswert Eins gesetzt.

Anstatt der gewünschten Dreiecksfläche ohne lokale Modifikation A_{Δ_w} wird für die Qualitätsberechnung die modifizierte gewünschte Dreiecksfläche $A_{\Delta_{\tilde{w}}}$ sowie der Wichtungsfaktor w_{xyz} verwendet.

$$A_{\Delta_{\tilde{w}}} = \frac{A_{\Delta_w}}{w_{xyz} \cdot f(\vec{p}_M)} \tag{5.13}$$

Weiterhin ist eine Beeinflussung der Elementierung durch die Definition einer Quellenverteilung möglich. Werden durch den Benutzer m Quellpunkte $\vec{q}_i = (x_i, y_i, z_i)^T$ der Stärke s_i vorgegeben, so wird die gewünschte Elementfläche $A_{\Delta_{\tilde{w}}}$ umgekehrt proportional dem Abstand zu diesen Quellpunkten beeinflusst. Mit dem Einfluss S aller Quellen auf $\vec{p}_M$

$$S = \sum_{i=0}^{m} \frac{s_i}{|\vec{p}_M, \vec{q}_i|} \tag{5.14}$$

, sowie dem Wichtungsfaktor w_Q berechnet sich diese nach:

$$A_{\Delta_{\tilde{w}}} = \frac{A_\Delta}{w_Q \cdot S} \tag{5.15}$$

Der Ausdruck $|\vec{p}_M, \vec{q}_i|$ bezeichnet den Abstand zwischen Mittelpunkt $\vec{p}_M$ der betrachteten Kante und dem Quellpunkt $\vec{q}_i$. Um bei sehr geringen Abständen einen zu großen Beitrag eines Quellpunktes zu vermeiden, wird für $s_i/|\vec{p}_M, \vec{q}_i| < 50$ vorgegeben.

5.3.3.3 Abbruchkriterien

Als Kriterien für einen Abbruch der Netzvergröberung sind zwei mögliche Bedingungen zu identifizieren:

- Da die mittlere Größe der Elemente, und damit die Netzfeinheit, für ein gegebenes Schiff durch die Anzahl der Elemente bestimmt ist, ist durch die Festlegung einer minimal zulässigen Zahl an Elementen eine wirkungsvolle Möglichkeit gegeben, die Gitteroptimierung zu einem gewünschten Punkt abzubrechen. Weiterhin ist vielfach die Größenordnung der für die CFD-Rechnung erforderlichen Elementanzahl bekannt, so dass hierdurch rasch Netze gewünschter Feinheit erzeugt werden können.
- Weiterhin ist ein qualitätsbasiertes Abbruchkriterium denkbar, wodurch, so die mittlere Qualität aller Kanten einen bestimmten Wert unterschreitet, die Netzoptimierung abgebrochen wird. Auf Grund der beliebig durch den Anwender definierbaren Gewichtungsfaktoren für die Qualitätsberechnung ist dieses Kriterium allerdings weniger aussagekräftig. Es kann daher nur bei iterativer Netzoptimierung wirksam eingesetzt werden.

5.3.4 Netzadaption an die Wasserlinienkontur

Weiterhin bietet der Vernetzungsalgorithmus die Möglichkeit, eine Adaption der über die gesamte Schiffsaußenhaut erzeugten Gitterstruktur an eine Kontur der Wasserlinie vorzunehmen.

So davon ausgegangen wird, dass die Wasseroberfläche symmetrisch zur Mittschiffsebene ausgebildet ist, können folgende Fälle unterschieden werden:

1. ebene Wasseroberfläche, d. h. für die Kontur der Wasserlinie gilt $\zeta = const.$
2. ausgelenkte Wasseroberfläche, d. h. es existiert ein Wellensystem. Hierfür gilt $\zeta = f(x)$.

Für die Anpassung des bestehenden Gitters an die Wasserlinienkontur werden in einem ersten Schritt alle Dreieckselemente gelöscht, die vollständig oberhalb der Wasserlinie angeordnet sind. Dreiecke, welche vollständig unterhalb der Wasserlinie liegen, werden nicht verändert.

Bei den Dreiecken, welche in der Wasserlinie angeordnet sind, d. h. zwei Kanten schneiden die Wasserlinienkontur, werden gemäß der in Abschnitt 4.5.1 beschriebenen Methodik der *point relocation* alle oberhalb der Wasserlinie liegenden Punkte $\vec{p}_i = (x_i, y_i, z_i)^T$ verschoben, so dass gilt:

$$z_i = \zeta(x_i) \tag{5.16}$$

Weiterhin ist es möglich, so z. B. nur ein geringer Teil des Dreiecks unterhalb der Wasserlinienkontur liegen sollte, alle unterhalb der Wasserlinie liegenden Punkte anzuheben.

Ebenfalls denkbar ist, die Dreiecke in mehrere Teildreiecke zu zerlegen und hierdurch eine feinere Auflösung der Wasserlinienkontur zu erreichen. Aus Zeitgründen wurde im Rahmen der Entwicklung dieser *proof of concept*-Lösung von einer Implementierung dieser beiden Verfahrensweisen Abstand genommen.

Kapitel 6

Implementierung

In Kapitel 5 wurde eine Beschreibung der Funktionsweise des erarbeiteten Vernetzungsalgorithmus gegeben. Aufbauend auf diesen Ausführungen wird in diesem Kapitel die programmiertechnische Umsetzung, d. h. die sog. Implementierung, erläutert.

In Abschnitt 6.1 werden allgemeine Informationen zur verwendeten Programmiersprache sowie zur Programmiertechnik gegeben. Gründe für die Wahl werden genannt. Einen Überblick über die entwickelte Geometriebibliothek gibt Abschnitt 6.2. Die Paketstruktur sowie interne Datenverwaltung werden beschrieben. Im folgenden Abschnitt 6.3 wird der Programmablauf aus Sicht des Entwicklers dargestellt und soll so Interessierten eine vereinfachte Einarbeitung in die bereits erzeugte Code-Basis ermöglichen.

Für die Darstellung des Programmablaufs wird, wie in der Informatik vielfach üblich, die Notation gemäß *universal markup language*[1] herangezogen. Für eine grundlegende Einführung in diese Beschreibungssprache siehe z. B. [34], [35].

6.1 Programmiersprache und -technik

In [15] ist eine grundlegende Funktionalität zum Einlesen von IGES-Dateien wie auch der Handhabung der darin beschriebenen NURBS-Flächen erarbeitet worden. Als Programmiersprache kam hierbei *java*[2] zum Einsatz.

Da demnach bereits eine solide Basis für die weitere Bearbeitung vorhanden war, wurde eine Abkehr von der Programmiersprache *java* zwar erwogen, es konnten allerdings für die erstellte *proof of concept*-Lösung keine signifikanten Vorteile beim Einsatz anderer Sprachen[3] festgestellt werden. Der nicht unerhebliche Aufwand für die Portierung der vorhandenen Funktionalität in eine andere Sprache sprachen ebenfalls gegen eine Portierung.

[1] Es wird eine Beschreibung nach *universal markup language* (*UML*) Version 1.1 verwendet.

[2] Siee http://www.java.sun.com

[3] Insbesondere die Programmiersprachen C bzw. C++ wurden betrachtet, da hierfür diverse Bibliotheken zur Handhabung von Dreiecksnetzen etc. verfügbar sind. Eine nicht der gewünschten Funktionalität entsprechende Leistungsfähigkeit dieser Bibliotheken schränkte den Nutzen dieser Komponenten allerdings stark ein.

Gründe für den Einsatz der Programiersprache *java* sind zum einen die plattformübergreifende Einsetzbarkeit sowie freie Verfügbarkeit dieser Sprache, so dass Einsatz und Entwicklung unter vielfältigsten Betriebssystemen ohne aufwendige Portierung möglich ist, sowie der konsequent objektorientierte Ansatz, der die Entwicklung eines leicht wartbaren und modifizierbaren Codes fördert. Weiterhin wird die Entwicklung durch eine umfangreiche und leistungsfähige Bibliothek, das sog. *java development kit*, unterstützt; die Einbindung von Fremdbibliotheken, eventuell verbunden mit Lizenzproblemen bzw. nur eingeschränkter Verfügbarkeit auf bestimmten Architekturen, ist im Rahmen der Implementierung nur für den Einsatz der *surface reconstruction*-Lösung *cocone* erforderlich.

Bei der Evaluierung des Vernetzungsprogramms zeigte sich, dass eine konsequent objektorientierte Programmierung zwar zu sehr gut verständlichen und handhabbaren Quelltexten führt, die interne Erzeugung und Handhabung von Objekten allerdings einen Abschlag bezüglich der Ausführungsgeschwindigkeit bedingt. Soll die entwickelte Software nicht nur als *proof of concept*-Lösung verwendet werden, so ist eine Optimierung der Objekthandhabung wie auch eine eingehende Analyse und Optimierung der implementierten Methodik zu wünschen.

Die durch den Autor erstellte Software ist unter der *General Public License* (*GLP*) frei verfügbar. Die Lizenzvereinbarungen sind im Anhang, Abschnitt D, einzusehen.

6.2 Code- und Datenstrukturen

Dieser Abschnitt gibt einen Überblick über Aufbau und Strukturierung des Quelltexts in Packete, Abschnitt 6.2.1, sowie die entwickelte Klassenbibliothek, Abschnitt 6.2.2. Informationen über die Speicherungen von Geometrieinformationen werden in Abschnitt 6.2.3 dargelegt.

6.2.1 Paketstruktur

Gemäß der Philosophie von *java* wurde bei der Code-Entwicklung auf eine sinnvolle und verständliche Strukturierung des Quelltexts Wert gelegt. Hierzu gehört neben selbsterklärenden Klassen- und Methodennamen eine Aufteilung der entwickelten Klassen in sog. Pakete[4]. Es wurde versucht, eine weitgehende Trennung von Daten, Benutzeraktivitäten und der Modellierung logischer Zusammenhänge zu erreichen[5]. Hierzu sind folgende *packages* definiert:

- **iges:** Für das Einlesen und Umwandeln von IGES-Datei (aus [15])
- **geometry:** Klassen zur Repräsentation von geometrischen Objekten
- **meshing:** Netzverfeinerung, Netzvergröberung, Adaption an eine gg. Wasserlinie, Klassen zur Ein- und Ausgabe sowie erforderliche Helferklassen

[4] Im Englischen als *packages* bezeichnet.

[5] Auf Grund des kurzen Bearbeitungszeitraums konnte dies nicht durchgängig erreicht werden. Vielmehr erfolgte die Entwicklung z. T. angelehnt an die Methodik des *extreme programming*.

- **ui:** Dialoge, Elemente zur Visualisierung
- **visualization:** Klassen zur Darstellung von Dreiecksgittern mittels *java3d*

6.2.2 Geometriebibliothek

Aufbauend auf der in [15] entwickelten Geometriebibliothek werden die für die Speicherung und Verarbeitung von Geometrieobjekten benötigten Klassen und Methoden im Package geometry bereitgestellt. Wichtige Elemente dieser Bibliothek sind:

- Die Klassen Vertex2D, Vertex3D, Vertex4D, Vector3D und Vector4D werden verwendet, um Punkt- und Vektorinformationen zu speichern. Umfangreiche Konvertierungsmöglichkeiten zwischen Objekten dieser Klassen sowie Methoden zur Manipulation stehen zur Verfügung.
- Kanten eines Dreiecks werden durch die Klassen Edge3D und Edge3DIndexed modelliert, indem die Endpunkte der jeweiligen Kante gespeichert werden. In Objekten des Typs Edge3D wird die Kante durch in diesem Objekt gespeicherte Instanzen der Klasse Vertex3D gespeichert, d. h. die Koordinaten der Endpunkte sind innerhalb des Objektes abgelegt. Bei Verwendung der Klasse Edge3DIndexed sind die Punkte einer Kante in einem separaten Vektor gespeichert. Innerhalb von Objekten der Klasse Edge3DIndexed werden ausschließlich die Indizes der Punkte in diesem Vektor abgelegt.
- Triangle3D, Triangle3DIndexed sind Klassen zur Modellierung von Dreieckselementen. Methoden zur Bestimmung der Flächennormalen, der Fläche eines Dreiecks wie auch der Berandungskanten sind implementiert.
- Mit Hilfe der Klassen NurbsSurfacePatch und TriangleSurfacePatch werden einzelne NURBS-Oberflächen bzw. konsistente Dreiecksnetze abgebildet.
- Objekte des Typs NurbsSurfaces wie auch TriangleSurfaces werden genutzt, um die im vorigen Punkt erwähnten Objekte zusammenzufassen. Sie fungieren als sog. *wrapper* um mehrere dieser Objekte und bieten die Möglichkeit, identische Operationen auf alle Objekte z. B. des Typs TriangleSurfacePatch einfach und schnell auszuführen.

Weiterhin sind verschiedene weitere Klassen implementiert, die vielfach ausschließlich als Helferklassen für oben beschriebenen Klassen eingesetzt werden. Eine ausführliche Darstellung wird aus Platzgründen nicht gegeben; Interessierte seien an dieser Stelle auf den dieser Arbeit beiliegenden Quellcode verwiesen.

6.2.3 Speicherung von Geometrieinformationen

Wie in Abschnitt 6.1 beschrieben erfolgt die Speicherung von Daten objektorientiert, d. h. geometrische Objekte werden abgebildet durch Instanzen entsprechender Objekttypen.

Sind mehrere Objekte gleichen Typs darzustellen, so werden diese als Elemente von z. B. Vektoren gespeichert.

Da die Modellierung eines konsistenten Gitters aus Dreieckselementen ein Kernstück der im Rahmen dieser Arbeit implementierten Methodik darstellt und diese von der in [15] entwickelten Datenstruktur abweicht, soll die in der Klasse TriangleSurfacePatch herangezogene Datenstruktur kurz vorgestellt werden.

Ein Gitter bestehend aus Dreieckselementen wird definiert durch n Punkte, den sog. Knoten. Diese werden in einem Vektor `points` in Form von Vertex3D Objekten gespeichert. Jedes Dreieck eines Gitters ist durch die Knoten in den Eckpunkten eindeutig beschrieben. Es wird durch ein Objekt vom Typ Triangle3DIndexed definiert, welches neben weiteren, für die Netzvergröberung gebrauchten Angaben wie z. B. `isLocked` die Indizes der drei Eckpunkte enthält, wie sie im Vektor `points` gespeichert sind.

Im Vergleich zur in [15] zur Anwendung kommenden Datenstruktur, dort werden alle Kanten der Elemente aufeinanderfolgend in einem Vektor in Form von Edge3DIndexed Elementen gespeichert, bietet diese obige Art der Speicherung den Vorteil einer größeren Datensicherheit[6] und einer einfacheren Gewinnung aller Informationen über ein bestimmtes Dreieck.

6.3 Allgemeiner Programmablauf

Von der in Abb. 5.2 dargestellten Logik des Vernetzungsalgorithmus ausgehend wurde durch den Autor unter der Verwendung der in Abschnitt 6.2 beschriebenen Geometriebibliothek der Vernetzungsalgorithmus implementiert. Eine stark vereinfache Abbildung der entwickelten Programmstruktur zeigt Abb. 6.1 und kann durch folgende Schritte beschrieben werden:

1. Erzeugung eines IgesReader Objektes. Diesem wird im Konstruktor die auszulesende Datei übergeben.

2. Auslesen und Parsen der IGES-Datei. Alle Einträge mit dem *Iges entity type* `type` werden zu NurbsSurfacePatch Objekten umgewandelt. Diese werden in einem Vektor `vecNS` gespeichert. `vecNS` wird zurückgegeben.

3. Mit `vecNS` wird ein NurbsSurfaces Objekt erzeugt, in welchem die ausgelesenen NURBS-Objekte zur Verfügung gestellt werden.

4. Durch die Methode `getStructTriangleSurfaces` wird für alle NurbsSurfacePatch Objekte eine strukturierte Vernetzung durchgeführt. Der durch den Anwender vorgegebene Wert für die Kantenlänge wird beim Aufruf der Methode übergeben. Das je Patch erzeugte TriangleSurfacePatch Objekt wird gespeichert in `trSurf` vom Typ TriangleSurfaces.

[6] Wird z. B. eine falsche Kanteninformation gespeichert oder eine Kante nicht definiert, so ist beim in dieser Arbeit verwendeten Modell nur ein Element betroffen, während bei der in [15] eingesetzte Methodik eine mögliche Kompromitierung des gesamten Netzes möglich ist.

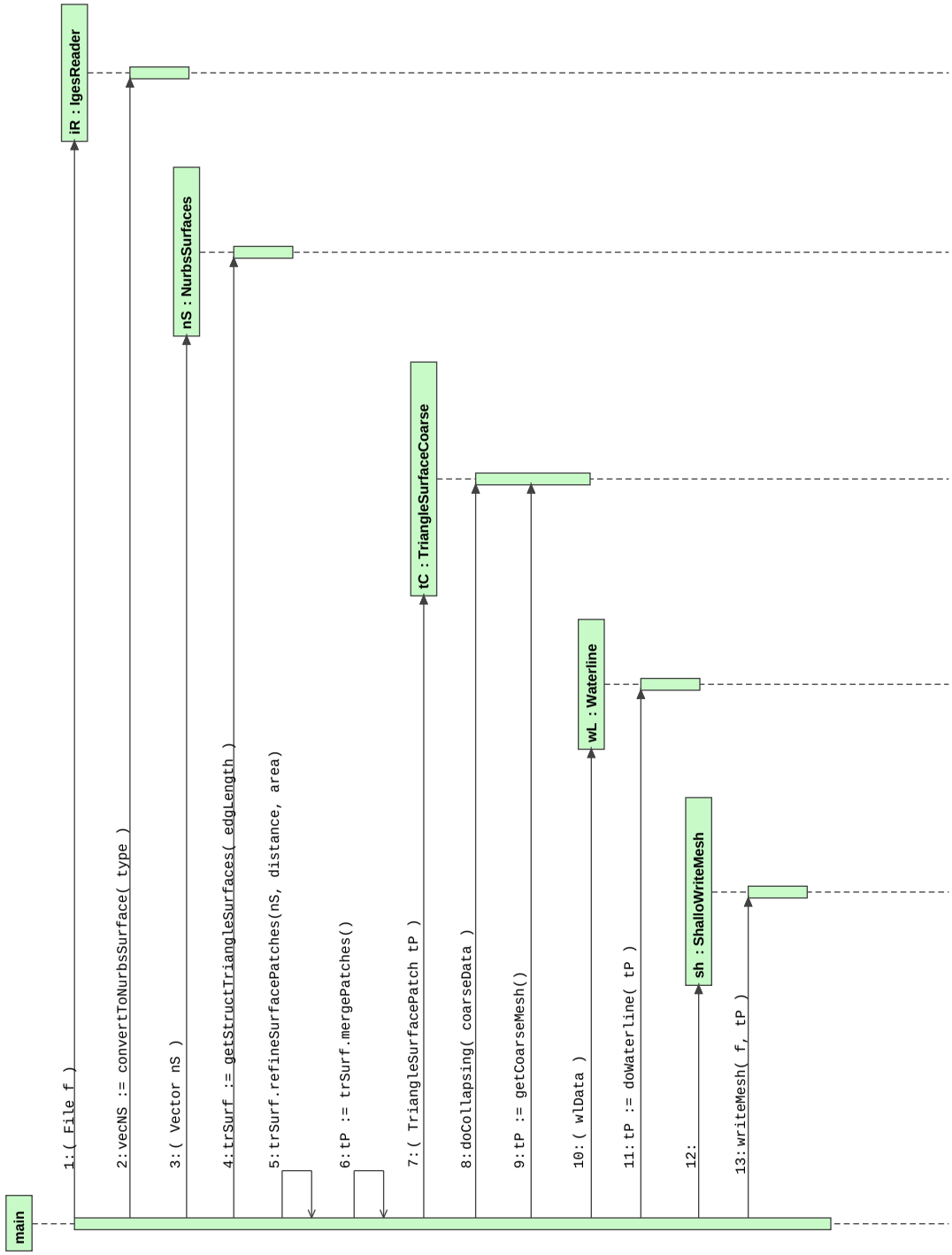

Abb. 6.1: Allgemeiner Programmablauf

5. `trSurf.refineSurfacePatches` verfeinert die Gitterstruktur jedes Dreiecksnetzes gemäß den übergebenen Parametern für maximale Fläche `area` und maximalen Abstand `distance`. Da zusätzliche Punkte auf der Oberfläche der NURBS-Patches erzeugt werden, wird das NurbsSurfaces Objekt `nS` mit den Nurbs-Oberflächenbeschreibungen ebenfalls übergeben.

6. Das *merging* der einzelnen Netze zu einem konsistenten Gitter wird durch die interne Methode `trSurf.mergePatches` erreicht. Es wird das zusammengeführte TriangleSurfacePatch `tP` zurückgegeben.

7. Für die Netzvergröberung wird ein Objekt des Typs TriangleSurfaceCoarse erzeugt. Das TriangleSurfacePatch `tP` wird im Konstruktor übergeben und steht somit innerhalb des Objektes für Modifikationen zur Verfügung.

8. Der Prozess der Netzvergröberung wird durch Aufruf der Methode `doCollapsing` gestartet. Die durch den Anwender gegebenen Einstellungen werden in einem Objekt des Typs CoarseData gespeichert und übergeben.

9. Das Gitter mit reduzierter Elementanzahl wird mittels `getCoarseMesh` ausgelesen.

10. Bei der Erzeugung des für die Behandlung der Wasserlinie zuständigen Objektes werden die Nutzerangaben in Form eines Objektes vom Typ WaterlineData übergeben.

11. Mittels `doWaterline` wird das Netz entsprechend der in `wlData` vorgegebenen Wasserlinie beschnitten und angepasst. Dieses wird als TriangleSurfacePatch Objekt zurückgegeben.

12. Für die Ausgabe des Netzes wird ein Objekt vom Typ ShalloWriteMesh erzeugt.

13. Aufruf der Methode `writeMesh` erzeugt die als Parameter übergebene Datei `f` und schreibt das im TriangleSurfacePatch `tP` definierte Netz in diese. Es wird ein Format ausgegeben, welches durch die von der *HSVA* entwickelten CFD-Codes *shallo* und *nyshallo* eingelesen werden kann.

Kapitel 7

Ergebnisse

Mit Hilfe des in Kapitel 5 und 6 beschriebenen Vernetzungsalgorithmus wurden durch den Autor beispielhaft Gitter auf Schiffsoberflächen erzeugt, um so die Funktionsweise der Netzgenerierung zu testen. Dieses Kapitel fasst die dabei gewonnenen Erfahrungen zusammen.

In Abschnitt 7.1 werden die zur Entwicklung und Validierung verwendeten Schiffsbeschreibungen vorgestellt. Für diese Beispielschiffe werden in den Abschnitten 7.2, 7.3 und 7.4 exemplarisch Oberflächennetze für die drei Schritte des Vernetzungsalgorithmus, d. h. patchweise Anfangsvernetzung, *merging* und Netzvergröberung beschrieben. Abschnitt 7.5 zeigt die Adaption eines durch die vorhergehenden Schritte erzeugten Gitters an eine durch den Anwender definierte Wasserlinienkontur.

7.1 Schiffsbeschreibungen

Im Rahmen der Entwicklung und Validierung des Vernetzungsalgorithmus werden ähnlich der Vorgehensweise in [15] primär folgende drei durch IGES-Dateien repräsentierte Schiffsformen verwendet.

5415.igs *DTMB 5415* ist ein nicht gebauter Entwurf einer kleinen Korvette, der durch das *David Taylor Model Basin* ca. 1980 entwickelt wurde. Nähere Informationen hierzu sind in [36] zu finden; Abb. 7.1 zeigt das Fahrzeug.

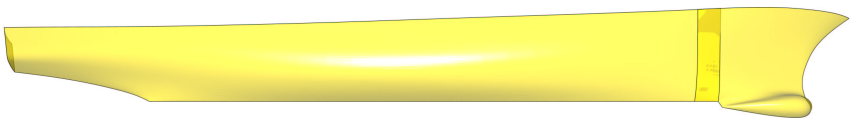

Abb. 7.1: Korvette *DTMB 5415* (Quelle: [15])

Eine Vernetzung von *DTMB 5415* stellt insbesondere im Bereich des Sonardoms und Bugwulstes hohe Anforderungen an den Vernetzungsalgorithmus, da dort sehr starke Ände-

rungen der Spantkontur wie auch des Krümmungsverlaufes auf einem Spant zu verzeichnen sind. Weiterhin ist durch den Algorithmus sicherzustellen, dass der Knick zwischen Spiegel und Schiffslängswand erkannt und gehalten wird.

Es ist zu erkennen, dass die Schiffsaußenhaut durch insgesamt drei NURBS-Patches beschrieben wird. Der Spiegel wird durch ein dreieckiges Patch modelliert. Im Vorschiffsbereich überlappen sich die beiden NURBS-Patches deutlich, so dass sich dieses Beispielschiff insbesondere dafür eignet, die numerische Robustheit des Algorithmus, siehe Abschnitt 5.1.1, zu testen.

twin.igs Hierbei handelt es sich um ein sog. RoPAX-Schiff, d. h. um eine schnelle Zweischraubenfähre mit Center-Skeg von ca. $130m$ Länge. Zur Formmodellierung werden insgesamt 335 NURBS-Patches verwendet, siehe Abb. 7.2.

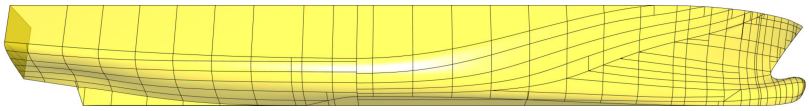

Abb. 7.2: RoPAX-Schiff Quelle: [15])

Durch den Vernetzungsalgorithmus sind insbesondere die starke Krümmung im Bereich des Skegs adäquat aufzulösen. Der stark ausgebildete Bugwulst in Kombination mit dem deutlich auskragenden Vorsteven stellt hohe Anforderungen an das *merging* der einzelnen Dreiecksnetze.

mercator.igs Das Schiff *Mercator* ist ein Handelsschiff von ca. $150m$ Länge gebaut in den 60er Jahren durch die Neptun-Werft, Rostock. Es ist in Abb. 7.3 abgebildet. Die IGES-Datei wurde mit Hilfe des CAD-Systems *Napa* erzeugt und besteht aus insgesamt 117 NURBS-Flächen.

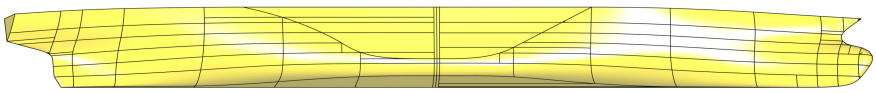

Abb. 7.3: Frachtschiff Mercator

Kennzeichnendes Merkmal dieses Schiffes ist im Vergleich zu modernen Schiffen das Fehlen eines Heckwulstes. Stattdessen ist ein nahezu senkrechter Hintersteven vorhanden. Dieser läuft nach achtern waagerecht aus und geht in den Spiegel über.

7.2 Anfangsvernetzung

In einem ersten Schritt wird im Rahmen der Erzeugung des Ausgangsgitters eine strukturierte Vernetzung durchgeführt. Abb. 7.4 zeigt für das vorderste NURBS-Patch der Korvette *DTMB 5415* zwei strukturierte Netze. Als Kantenlänge auf der Berandung wurde $0.15m$ respektive $0.05m$ vorgegeben[1]. Deutlich ist zu erkennen, wie durch eine feine-

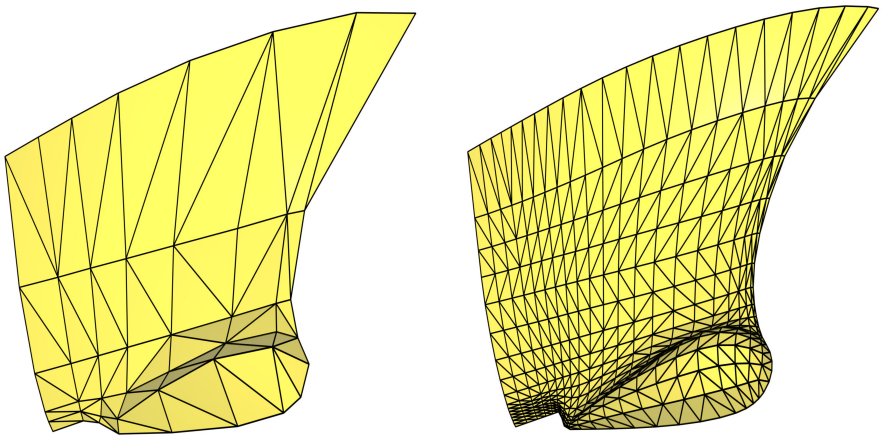

Abb. 7.4: Strukturierte Vernetzung eines Patches mit einer Kantenlänge von $0.15m$ (links) und $0.05m$ (rechts)

re strukturierte Vernetzung die Schiffsform besser durch Dreieckselemente approximiert wird.

Die Elementfläche verringert sich in Kielnähe verglichen mit der Elementgröße im Bereich des oberen Decks. Dies ist darauf zurückzuführen, dass ein nicht linearer Zusammenhang zwischen parametrischem und kartesischem Koordinatensystem vorhanden ist, und kann durch die implementiere Verfahrensweise nicht ausgeglichen werden.

Beispiele für die in einem zweiten Schritt durchgeführte Netzverfeinerung sind in Abb. 7.5 abgebildet. Als Kriterien für die Abschätzung, ob ein Dreieckselement zu verfeinern ist, wurde zum einen die maximale Fläche eines Elementes A_{max}, zum anderen der maximale Abstand $dist_{max}$ zwischen den Mittelpunkten der Elementkanten und der NURBS-Fläche verwendet.

Durch die Verwendung des Kriteriums $dist_{max}$ werden insbesondere die Bereiche verfeinert, in denen eine starke Oberflächenkrümmung vorliegt, d. h. für die betrachtete Bugsektion der Bugwulstbereich sowie Bereiche nahe dem Vorsteven bzw. oberen Deck. Auf Grund der in Abschnitt 5.3.2.2 beschriebenen Methodik der *butterfly subdivision* werden Nachbardreiecke ebenfalls unterteilt, so dass eine Netzverfeinerung z. T. auch in angrenzenden Bereichen durchgeführt wird. Hieraus resultiert die variierende Netzdichte in Gebieten ähnlicher Krümmung.

[1] Diese sehr geringen Werte ergeben sich daraus, dass vorliegende IGES-Datei eine falsche Definition des Maßstabs enthält und somit sich eine Schiffslänge von $L_{PP} \approx 6.2m$ ergibt.

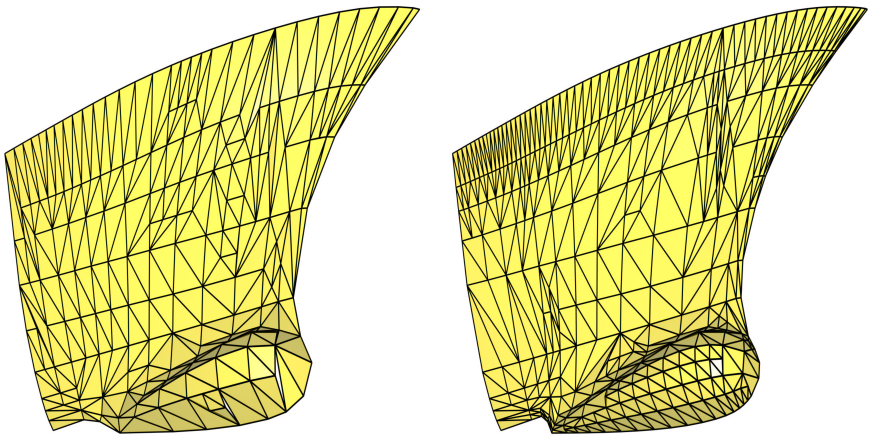

Abb. 7.5: Adaptive Netzverfeinerung durch das Kriterium $A_{max} = 0.01m$ (links) bzw. $dist_{max} = 0.01m$ (rechts)

Wird die maximale Fläche eines Dreieckselementes vorgegeben, so wird das Patch weitgehend homogen unterteilt; eine nur gering schwankende Netzdichte wird erzielt. Dies führt allerdings dazu, dass in ebenen oder nahezu ebenen Bereichen nicht benötigte zusätzliche Elemente erzeugt werden, während im Bereich des Bugwulstes und Sonardoms die Approximation der Oberfläche nur ungenügend erfolgt.

Durch eine Kombination der beiden Kriterien wird dem Anwender die Möglichkeit geboten, die Netzdichte in Gebieten mit starker Krümmung zu erhöhen, zugleich aber durch die Festsetzung einer maximalen Fläche die Schwankung der Netzdichte zu begrenzen. Hierdurch kann bereits bei der Anfangsvernetzung ein homogenes, also ein sog. „schönes“ Netz erzeugt werden.

7.3 Merging

Wie bereits in Abschnitt 5.3.2.3 erwähnt arbeitet der für das Zusammenfügen der Netze eingesetzte, extern eingebundene Algorithmus *cocone* nicht garantiert fehlerfrei. Daher werden nicht nur Beispiele einwandfreier Funktionsweise dieses *surface reconstruction*-Algorithmus dargestellt, sondern es werden ebenfalls Probleme und Fehler von *cocone* sowie eventuell mögliche Gegenmaßnahmen zu erläutert.

Für das Beispielschiff *DTMB 5415* ist in Abb. 7.6 die Vernetzung vor und nach dem Zusammenfügen der je Patch erzeugten Gitter dargestellt. Es ist zu erkennen, dass durch *cocone* überwiegend ein „schönes“ Gitter mit harmonischer Elementdichteverteilung erzeugt wird.

Die numerische Robustheit des Algorithmus ist insbesondere im Vorschiffbereich zu erkennen. Aus den beiden überlappenden Netzen, welche aus den NURBS-Flächen erzeugt wurden, wird ein konsistentes Dreiecksgitter, wenn auch von höherer Netzfeinheit, gene-

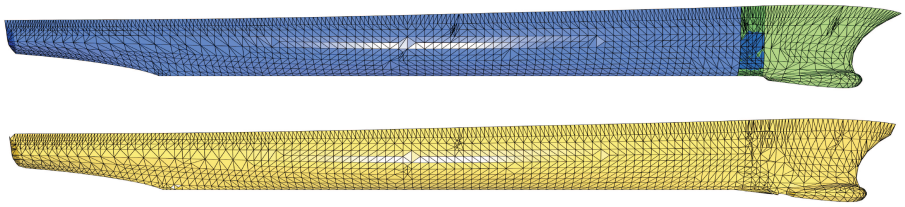

Abb. 7.6: Zusammenführen der Dreiecksnetze für *DTMB 5415*

riert.

Am Bug wie auch am Spiegel wird im Bereich des oberen Decks die Berandung nicht korrekt erkannt; es fehlt je ein Dreieckselement. Im Rahmen der Adaption an die Wasserlinienkontur werden obige Elemente entfernt, so dass diese Artefakte als unproblematisch bewertet werden können.

Am geometrisch komplexen Übergang zwischen Sonardom und Kiel sind ebenfalls Probleme bei der Erkennung der Berandung zu erkennen, siehe Abb. 7.7. Es werden zum einen zwei Dreieckselemente nicht erzeugt, zum anderen werden diverse Panels außerhalb der durch die NURBS-Flächen beschriebenen Berandung generiert.

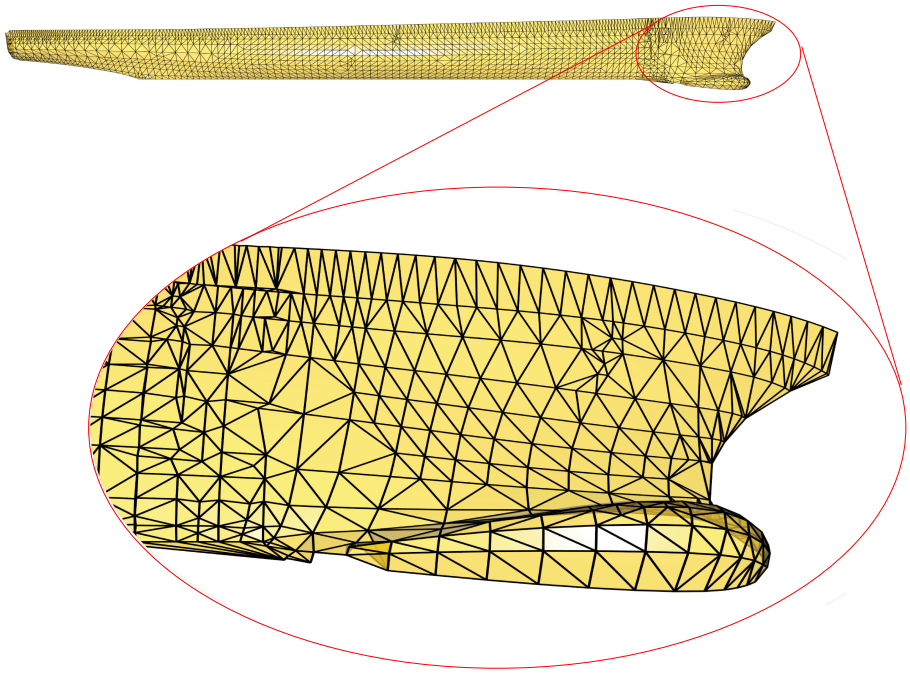

Abb. 7.7: Fehler durch das *merging* im Bereich des Sonardoms

In Abb. 7.8 ist das durch den *surface reconstruction*-Algorithmus erzeugte Netz für die Doppelschraubenfähre, IGES-Datei *twin.igs*, abgebildet. Hier sind deutliche Fehler zu erkennen. So wird, bedingt durch den stark ausgeprägten Bugwulst sowie den deutlich

geneigten Vorsteven, im Bugbereich die Berandung nicht korrekt erkannt; es sind mehrere Elemente außerhalb der Berandungskontur vorhanden. Weiterhin wird der Abschluss der Außenhaut am oberen Deck falsch triangulisiert, so dass zwischen Spiegel und Schiffslängswand horizontal angeordnete Dreiecke vorhanden sind.

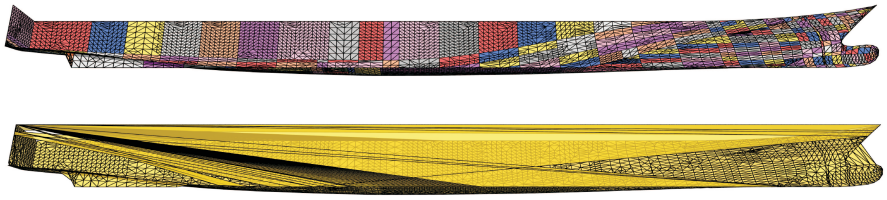

Abb. 7.8: *Merging* am Beispiel der RoPAX-Fähre

Die in Abb. 7.8 erkennbaren deutlichen Fehler lassen sich z. T. unterbinden bzw. vermindern, so eine geeignete Ausgangsvernetzung gewählt wird. Dies ist z. B. dadurch möglich, dass die strukturierte Anfangsvernetzung feiner erfolgt. Dennoch zeigt dieses Beispiel, dass die Verwendung von *cocone* bzw. von ähnlich arbeitenden Algorithmen zu Problemen führt, so die Berandung der Punktemenge nicht einfach bzw. eindeutig zu bestimmen ist.

7.4 Netzvergröberung

Für die Korvette *DTMB 5415* sind in Abb. 7.9 beispielhaft vier Gitter dargestellt, die aus einer Ausgangsvernetzung mit 15000 Dreiecken erzeugt wurden. Als Qualitätskriterien wurden das Seitenverhältnis q_{AR} sowie die Fläche der Dreieckselemente q_A verwendet.

Es ist zu erkennen, dass auf diesem Wege eine nur gering variierende Elementdichteverteilung über die Schiffsoberfläche erreicht wird. Das Seitenverhältnis der Elemente wie auch das Verhältnis der Flächen benachbarter Elemente erfüllt die in Abschnitt 4.2 genannten Anforderungen.

Kritisch kann benannt werden kann, dass insbesondere bei den Netzen mit geringer Elementanzahl die komplexe Geometrie im Vorschiffsbereich nicht ausreichend genau vernetzt wird, so nur die Qualitätskriterien q_{AR} und q_A verwendet werden. Die Fehler in der Vernetzung am achteren Ende des Sonardoms sind wiederum auf Fehler des *surface reconstruction*-Algorithmus zurückzuführen.

Wird neben oben genannten Qualitätskriterien ebenfalls die Krümmung der Oberfläche berücksichtigt, d. h. das durch den die lokale Krümmung beschreibenden Faktor k_K wird das Kriterium q_A modifiziert, so ergeben sich die in Abb. 7.10 dargestellten Oberflächennetze. Als Ausgangsbasis fand wiederum eine Vernetzung mit 15000 Dreiecken Verwendung.

Es ist zu erkennen, dass insbesondere Bereiche hoher Oberflächenkrümmung, d. h. Sonardom und Bugwulst, eine feinere Vernetzung aufweisen. Mit Ausnahme des Vorschiffsbereichs ist eine gleichmäßige Vernetzung zu verzeichnen.

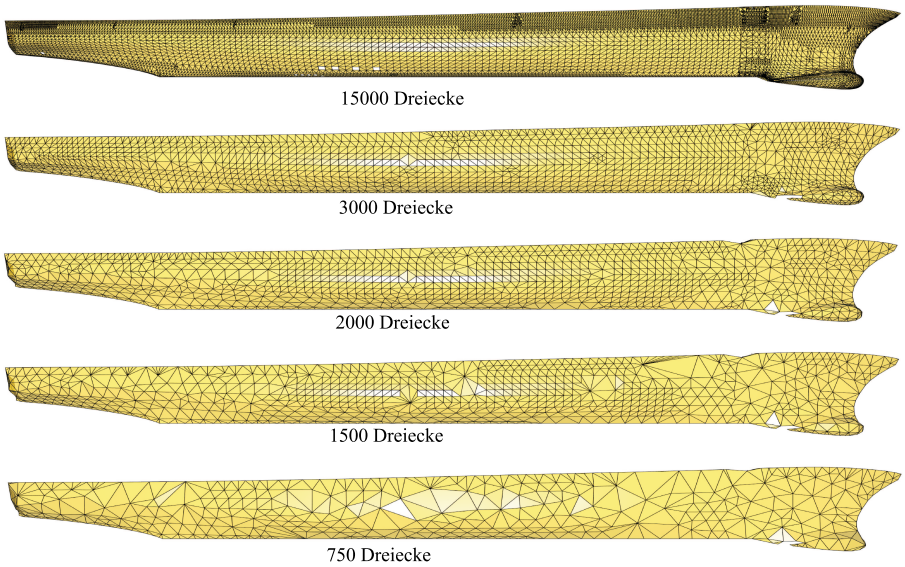

Abb. 7.9: Netzvergröberung mit den Qualitätskriterien q_{AR} und q_A

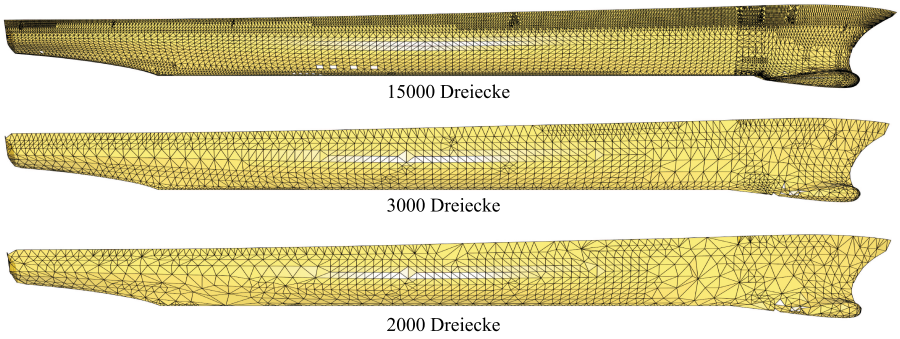

Abb. 7.10: Netzvergröberung mit den Qualitätskriterien q_{AR} und q_A sowie k_K

Die Unregelmäßigkeiten im Vorschiffsbereich sind, abgesehen von der gewünschten Elementgrößenanpassung auf Grund der Form der Außenhaut, auf die beiden überlappenden NURBS-Oberflächen zurückzuführen, welche im Rahmen des *merging* zu einem sehr feinen und unregelmäßigen Netz zusammengeführt werden. Dies kann zwar durch den Vernetzungsalgorithmus zum Teil ausgeglichen werden; ein vollständiger Ausgleich ist allerdings nicht möglich. Gründe hierfür sind die in Abschnitt 5.3.3.2 beschriebene Verfahrensweise zur Ermittlung der Oberflächenkrümmung wie auch Fehler des *surface reconstruction*-Algorithmus *cocone*, der in Gebieten mit stark variierender Elementdichte z. T. inkonsistente bzw. schlecht konditionierte, d. h. nicht Delaunay-konforme Gitter erzeugt[2].

In Abb. 7.11 sind zwei Gitter abgebildet, die, von einer Anfangsvernetzung mit 15000 Dreiecken ausgehend, mit Hilfe einer lokalen Beeinflussung der Elementgröße gemäß Gl. 5.12 erzeugt wurden. Hierbei wurde in Schiffslängsrichtung die in Abb. 7.12 abgebildete Einflussfunktion $f_x(x)$ verwendet.

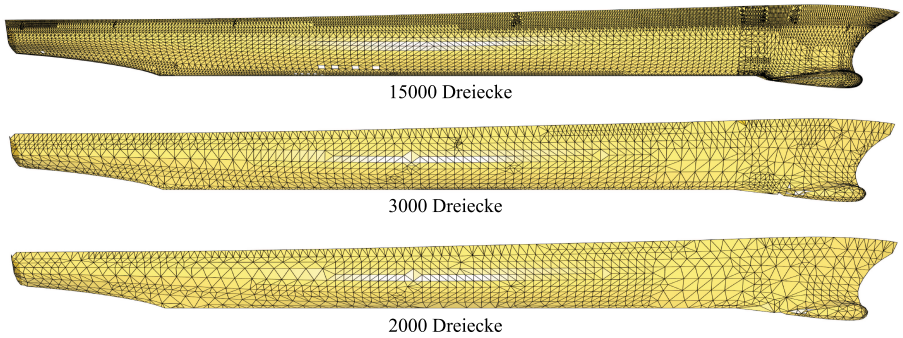

Abb. 7.11: Netzvergröberung mit lokaler Elementgrößenkontrolle

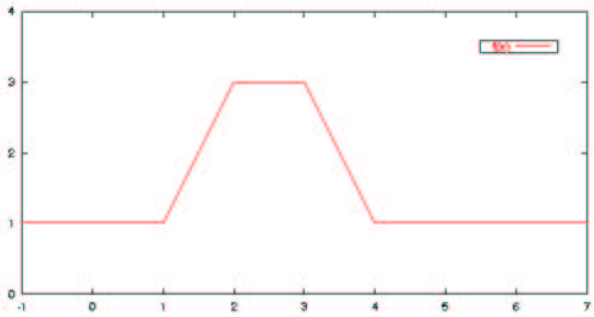

Abb. 7.12: Einflussfunktion $f_x(x)$

Es ist zu erkennen, dass, verglichen mit Abb. 7.9, im Mittschiffsbereich eine feinere Vernetzung vorliegt, d. h. dass die im Rahmen dieser Arbeit entwickelte Methodik der lo-

[2]Diese Problematik betont nochmals die Notwendigkeit zur Entwicklung eines *merging*-Verfahrens, welches direkt mit den NURBS-Oberflächen und den daraus abgeleiteten Dreiecksnetzen arbeitet.

kalen Elementgrößenanpassung wie gewünscht arbeitet. Ein sanfter Übergang zwischen den fein vernetzten Bereichen und dem gröberen Gitter ist zu verzeichnen. Dies ist zum einen auf die Definition der in Abb. 7.12 abgebildeten Einflussfunktion zurückzuführen; zum anderen fördert auch die Verfahrensweise des *edge collapse* in Kombination mit den implementierten Qualitätskriterien diese Tatsache selbst bei starken Schwankungen von $f_x(x)$.

7.5 Wasserlinie

Zum Abschluß des Vernetzungsalgorithmus wird das vergröberte Netz an die vorgegebene Wasserlinienkontur angepaßt. Dies ist in Abb. 7.13 dargestellt. Es sind Beispiele für eine ebene Wasserlinie, d. h. $z = const$, wie auch für eine ausgelenkte Wasseroberfläche mit $z = f(x)$ abgebildet.

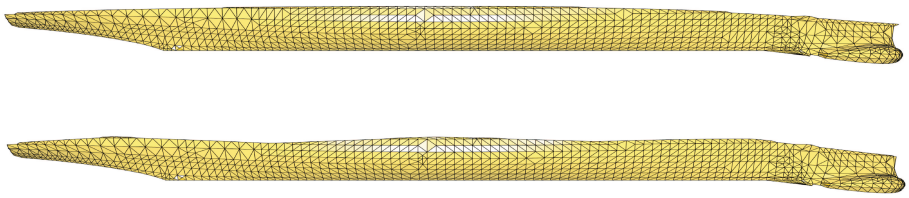

Abb. 7.13: Adaption der Vernetzung an die Wasserlinienkontur (oben: $z = const$; unten: $z = f(x)$)

In den abgebildeten Beispielen ist zu erkennen, dass die implementierte Verfahrensweise nicht optimal arbeitet. Liegen Punkte eines die Wasserlinie schneidenden Dreiecks nur geringfügig oberhalb von dieser, so entstehen durch die *point relocation* sehr kleine Dreiecke. Um auch im Bereich der Wasserlinie ein „schönes“Netz zu erzeugen, ist es daher empfehlenswert, je nach Situation entweder oberhalb der Wasserlinienkontur angeordnete Punkte nach unten zu verschieben oder unterhalb angeordnete Knoten nach oben.

Eine Implementierung dieser Methodik ist nicht mit großen Schwierigkeiten verbunden. Aus Gründen der beschränkten Bearbeitungszeit dieser Arbeit wurde diese Methodik nicht vollständig implementiert; vielmehr wurde versucht, eine Lösung für die drängendere Problemstellung des *merging* zu finden, siehe Abschnitt 5.3.2.3.

Kapitel 8

Fazit und Ausblick

Literaturverzeichnis

[1] SÖDING [u. a.]: Absenkung und Vertrimmung von Schiffen durch Fahrt in flachem Wasser. In: *Jahrbuch der Schiffbautechnischen Gesellschaft* (1989)

[2] SÖDING, H. *Numerische Berechnung inkompressibler Potentialströmungen.* 1995

[3] JENSEN, G.: Ein Programmsystem zur 2-dimensionalen Berechnung der Potentialströmung um Körper in unbeschränkter Flüssigkeit / Institut für Schiffbau der Universität Hamburg. 1994 (Nr. 442). – Forschungsbericht

[4] JENSEN, G.: Berechnung der stationären Potentialströmung um ein Schiff unter Berücksichtigung der nichtlinearen Randbedingung an der Wasseroberfläche / Institut für Schiffbau der Universität Hamburg. 1988 (Nr. 484). – Forschungsbericht

[5] BERTRAM, V.: Numerische Schiffshydrodynamik in der Praxis / Institut für Schiffbau der Universität Hamburg. 1994 (Nr. 545). – Forschungsbericht

[6] BERTRAM, V. ; HUGHES, M.: A higher-order panel method for 3-D free surface flows / Institut für Schiffbau der Universität Hamburg. 1995 (Nr. 558). – Forschungsbericht

[7] UHLENBROCK-JANSE, S.: *Untersuchungen zum dynamischen Verhalten von Steg- und Auslegersystemen*, Universität Rostock, Diplomarbeit, 2002

[8] HOSCHEK, Josef ; LASSER, Dieter: *Grundlagen der geometrischen Datenverarbeitung.* B. G. Teubner, 1989

[9] FARIN, Gerald E.: *Curves and surfaces for computer aided geometric design.* 4. Auflage. San Diego : Academic Press, 1997

[10] BARTELS, R. ; BEATTY, J. ; BARSKY, B.: *An introduction to splines for use in computer graphics & geometric modelling.* Morgan Kaufmann Publishers, 1987

[11] WATT, A.: *3D Computer Graphics.* 3. Auflage. Addison-Wesley, 2000

[12] THOMPSON, J. F. (Hrsg.): *Numerical grid generation.* North-Holland, 1982

[13] THOMPSON, J. (Hrsg.) ; SONI, B. (Hrsg.) ; WEATHERILL, N. (Hrsg.): *Handbook of grid generation.* CRC Press, 1999

[14] KLEIN, R.: *Netzgenerierung impliziter und parametrisierter Kurven und Flächen in einem objektorientierten System*, Universität Tübingen, Diplomarbeit, 1995

[15] Knieling, G.: *Entwicklung eines Panelgenerierungsverfahrens für eine Wellenwiderstandsberechnung von Schiffen*, Universität Rostock, Diplomarbeit, 2002

[16] HSVA: *The complete user's guide to shallo*

[17] Frey, P.: About surface remeshing / INRIA, Gamma project. – Forschungsbericht

[18] Liseikin, V.: *Grid generation methods.* Springer Verlag, 1999

[19] Hamann, B. ; Moorhead, R.: Survey of grid generation methodologies and scientific visualization efforts. In: Nielson, G. (Hrsg.) [u. a.]: *Scientific visualization: overwies, methodologies and techniques.* IEEE Computer Society Press, 1997, S. 59–102

[20] Smith, T.: Transfinite interpolation generation systems. In: Thompson, J. (Hrsg.) ; Soni, B. (Hrsg.) ; Weatherill, N. (Hrsg.): *Handbook of grid generation.* CRC Press, 1999

[21] Spekreisje, S.: Elliptic generation systems. In: Thompson, J. (Hrsg.) ; Soni, B. (Hrsg.) ; Weatherill, N. (Hrsg.): *Handbook of grid generation.* CRC Press, 1999

[22] Chan, W.: Hyperbolic methods for surface grid generation. In: Thompson, J. (Hrsg.) ; Soni, B. (Hrsg.) ; Weatherill, N. (Hrsg.): *Handbook of grid generation.* CRC Press, 1999

[23] Khamayseh, A ; Kuprat, A.: Surface grid generation systems. In: Thompson, J. (Hrsg.) ; Soni, B. (Hrsg.) ; Weatherill, N. (Hrsg.): *Handbook of grid generation.* CRC Press, 1999

[24] Frykestig, J.: *Advancing front mesh generation techniques with application to the finite element method.* Göteborg, Chalmers University of Technology, Diss., 1994

[25] George, P.: *Automatic mesh generation.* Wiley Publishers, 1991

[26] Moeller, P. ; Hansbo, P.: *On advancing front mesh generation in three dimensions.* – preprint

[27] Formaggia, L. ; Arienti, M.: A survey of mesh generation and adaption algorithms for computational fluid dynamics / Centro di Ricerca, Sviluppo e Studi Superiori in Sardegna. 1997 (Tech-Rep 97/33). – Forschungsbericht

[28] Owen, S.: *A survey of unstructured mesh generation topology.* – http://www.andrew.cmu.edu/user/sowen/

[29] Klein, R. *Latest advances in multiresolution surface modeling and applications to visualization, reconstruction and simulation*

[30] Cheng, B. ; Plaue, C.: Subdivision schemes for surfaces / Georgia Institute for Technology. 2002. – Forschungsbericht

[31] Amenta, N. [u. a.]: A simple algorithm for homeomorphic surface reconstruction. In: *16th ACM Symposium on Computational Geometry*, 2000

[32] Dey, T. ; Giesen, J.: Detecting undersampling in surface reconstruction. In: *17th ACM Symposium on Computational Geometry*, 2001

[33] Dey, T. ; Goswami, S.: Tight cocone: a water tight surface reconstructor / The Ohio State University. 2002. – Forschungsbericht

[34] Graham, I. ; Willis, A.: *UML - a tutorial.* Trireme International Ltd, 2000

[35] Harmon, P. ; Watson, M.: *Understanding UML.* Morgan Kaufmann, 1997

[36] Larsson, L. [u. a.]: Benchmarking of computational fluid dynamics for ship flows: the Gothenburg 2000 workshop. In: *Journal of ship research* 47 (2003), Nr. 1

Anhang A

Merging

A.1 Verfahren zur Topologieerkennung

Wie in Abschnitt 5.1.2 beschrieben ist die Aufgabe der Topologieerkennung, die Nachbarschaftsrelationen der NURBS-Oberflächen zu ermitteln und zu analysieren. Hierbei wird bestimmt, welche Patches eine gemeinsame Kante besitzen und wie benachbarte Patches aneinandergrenzen, siehe Abb. 5.1. Es ist hierbei eine Lösung zu finden, die eine ausreichende numerische Robustheit bietet, d. h. Überlappungen von bzw. Klaffungen zwischen NURBS-Oberflächen sind, so möglich, tolerant zu behandeln.

Durch den Autor wurden im Rahmen dieser Arbeit zwei Verfahren entwickelt und z. T. implementiert, die in den beiden folgenden Abschnitten vorgestellt werden.

A.1.1 *bounding box*-Verfahren

Unter dem Begriff *bounding box*-Verfahren sind Verfahren zu verstehen, die Nachbarschaftsrelationen dadurch ermitteln, dass getestet wird, ob bzw. in welchem Bereich sich geometrische Objekte umschreibende Quader, die sog. *bounding boxes*, schneiden.

A.1.2 *scanline*-Verfahren

A.2 *Merging*

Anhang B

CD-Inhalt

Auf beiliegender CD-Rom sind die im Rahmen der Diplomarbeit erstellten Dokumente, Quelltexte etc. enthalten. Die Verzeichnisstruktur ist angeordnet wie folgt:

- **cvs:** Enthält das sog. *cvs*[1]*-repository* mit den Modulen *mesh2*, den *java*-Quelltexten, und *text*, den Latex-Quelltexten für die schriftliche Ausarbeitung dieser Diplomarbeit
- **iges:** IGES-Dateien der in Abschnitt 7.1 beschriebenen Beispielschiffe
- **src:** Eine Kopie des *JBuilder*-Projektes, welches für die Code-Entwicklung verwendet wurde
- **text:** Die Latex-Dokumente und Abbildungen für die schriftliche Dokumentation

[1] concurrent versioning system

Anhang C

Dateiformate

Das erstellte und getestete Programm verwendet diverse Dateiformate, um extern verfügbare oder durch den Anwender bereitzustellende Informationen einzulesen. Weiterhin stehen diverse Möglichkeiten zur Verfügung, das erzeugte Netz abzuspeichern und, so gewünscht, in anderen Programmen weiter zu verwenden. Dieses Kapitel gibt einen Überblick über die Struktur und Bedeutung der Dateiformate.

C.1 Wasserlinienkontur

Für die Definition der Form der Wasserlinie werden in der Mittschiffsebene angeordnete Punkte verwendet. Die Kontur der Wasserlinie wird durch lineare Interpolation benachbarter Punkte ermittelt. Sind insgesamt n Punkte vorhanden, so werden diese in einer Textdatei definiert wie folgt:

```
x1 z1
...
xn zn
```

C.2 Lokaler Einfluss

Im Rahmen dieser Diplomarbeit wurden zwei Verfahren implementiert, um eine lokale Beeinflussung der Netzgenerierung zu erreichen, siehe Abschnitt 5.3.3.2, nämlich auf Grund der räumlichen Lage der Kante wie auch durch eine Quellenverteilung.

C.2.1 Lagebasiert

Für eine lagebasierte Qualitätssteuerung werden entsprechend Gl. 5.12 Einflussfunktionen in den drei Koordinatenrichtungen verwendet. Diese Funktionen werden durch Punkte in der entsprechenden Koordinatenrichtung definiert; die Funktionswerte werden durch lineare Interpolation ermittelt.

Die Einflussfunktionen werden in einer Datei definiert und anschließend durch das Programm eingelesen. Liegen in x-, y- und z-Richtung n, m bzw. o Aufpunkte für die Beschreibung der Funktionen vor, so wird dies in der Datei geschrieben als:

```
x x1 fx1
...
x xn fxn
y y1 fy1
...
y ym fym
z z1 fz1
...
z zo foz
```

In der ersten Spalte ist die Koordinatenrichtung geschrieben, für die dieser Punkt definiert wird, gefolgt von Argument und Funktionswert. Die Reihenfolge der einzelnen Zeilen ist beliebig.

C.2.2 Quellenverteilung

Für die Definition der Quellenverteilung wird ein Vorgehen ähnlich der Definition von Einflussfunktionen verwendet. Sind insgesamt n Quellen vorhanden, so werden diese in einer Datei beschrieben wie folgt:

```
x1 y1 z1 q1
...
xn yn zn qn
```

In den ersten drei Spalten sind hierbei die Koordinaten des jeweiligen Quellpunktes aufgeführt. Durch q_i wird die jeweilige Quellstärke definiert.

C.3 Gitter

Durch den Vernetzungsalgorithmus erzeugte Gitter können in verschiedenen Formaten ausgegeben werden. Diese werden im folgenden kurz beschrieben.

C.3.1 OFF

Der *surface reconstruction*-Algorithmus *cocone* verwendet das OFF-Format[1] für die Ergebnisausgabe. Es wurde eine Teilmenge der OFF-Definition implementiert, um diese Ergebnisse zu lesen. Weiterhin besteht die Möglichkeit, das erzeugte Netz in Form einer OFF-Datei auszugeben.

Für ein Netz bestehend aus nP Punkten und $nTrgl$ Dreiecken wird die OFF-Datei geschrieben entsprechend:

```
OFF

nP nTrgl 0
```

[1] Das OFF-Format findet Verwendung im Datenvisualisierungstool *geomview*. Eine umfangreiche Definition dieses Formates ist zu finden unter http://www.geomview.org

```
x1 y1 z1
...
xnP ynP znP

#JV nonBoundaryFacets TriangleSet

3  p1(1) p2(1) p3(1) 1.0 1.0 1.0 1.0
...
3  p1(nTrgl) p2(nTrgl) p3(nTrgl) 1.0 1.0 1.0 1.0
```

C.3.2 Shallo

Schließt sich an die Netzgenerierung eine Berechnung mit den durch die *HSVA* entwickelten CFD-Codes *Shallo* bzw. *nyShallo* an, so können die Netze geeignet ausgegeben werden. Die Struktur der erzeugten Dateien ist dabei wie folgt:

```
n
x1 y1 z1
...
xn yn zn
m
3 p1(1) p2(1) p3(1)
...
3 p1(m) p2(m) p3(m)
```

Hierbei wird durch n die Anzahl der Punkte gegeben; m steht für die Anzahl an Dreieckselementen. Im oberen Teil der Datei werden die Koordinaten der n Punkte definiert. Ein Dreieck i wird anschließend durch die Indizes der Eckpunkte $p1(i)$, $p2(i)$, $p3(i)$ definiert. Die Indizes verweisen auf Einträge in der Punktedefinition.

C.3.3 Autowing

Weiterhin ist eine Ausgabe der Netzstruktur in einer Form möglich, dass diese durch *Autowing* eingelesen und weiterverarbeitet werden kann. Diese hierfür verwendete Datenstruktur ist wie folgt:

```
nTrgl
x1(1),y1(1),z1(1)
x2(1),y2(1),z2(1)
x3(1),y3(1),z3(1)
...
x1(nTrgl),y1(nTrgl),z1(nTrgl)
x2(nTrgl),y2(nTrgl),z2(nTrgl)
x3(nTrgl),y3(nTrgl),z3(nTrgl)
```

Die Anzahl $nTrgl$ der Dreieckselemente steht in der ersten Zeile geschrieben. In den folgenden Zeilen werden in Dreiergruppen die Koordinaten der Eckpunkte jedes Elementes i aufgelistet.

Anhang D

The GNU General Public License

Version 2, June 1991

Copyright © 1989, 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions For Copying, Distribution and Modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program

or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

   In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

   This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

11. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

12. In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.