RaPPPS -

Resistance **a**nd **P**ropulsive **P**ower **P**rediction for **S**hips

Studienarbeit im Prüfungsfach
Schiffshydrodynamik
Thema:
„Leistungsprognose und Propelleroptimierung im Vorentwurf“

vorgelegt von:
Pascal Anschau, Matr.-Nr.: 182739

Aufgabenstellung und Betreuung:
Dr. Lothar Birk,
Institut für Land- und Seeverkehr
Fachgebiet Dynamik maritimer Systeme
TU Berlin

5. April 2006

Die selbständige und eigenhändige Anfertigung versichere ich an Eides statt.

Berlin, den 5. April 2006

Zusammenfassung

In der Phase des Vorentwurfs eines Schiffes ist u.a. eine vorläufige, aber möglichst realitätsnahe Aussage über den zu erwartenden Widerstand des Schiffes und die erforderliche Antriebsleistung notwendig. Zeitintensive Untersuchungen mittels CFD - Software fallen wegen der üblicherweise kurzen Fristen zur Angebotsabgabe als Prognosemöglichkeit aus; Modellversuche verbieten sich wegen der hohen Kosten und des Zeitaufwandes in diesem Stadium des Schiffentwurfes von selbst.
Statistische Verfahren bieten hier eine preiswerte und schnelle Alternative. Eine weit verbreitete Methode zur Widerstands- und Leistungsprognose ist das von Holtrop und Mennen in den 70er und 80er Jahren entwickelte Verfahren, das auf einer statistischen Auswertung mehrerer tausend Modell- und Großausführungsversuche basiert.
Die vorliegende Arbeit implementiert dieses Verfahren mit weitreichenden Möglichkeiten zur Visualisierung der Ergebnisse. Das Verfahren wird prinzipiell vorgestellt und das Optimierungsproblem wird detailliert formuliert. Das verwendete Optimierungsverfahren, die Struktur des Programms sowie verschiedene Programmiertechniken werden erklärt; Datenfluss und -organisation werden beleuchtet.
Das Programm ist in erster Linie für Parameterstudien in der Lehre gedacht, ist aber auch für den produktiven Einsatz geeignet.

Umfang der Arbeit

Die Arbeit besteht aus zwei Teilen:

1. **Dokumentation:** Der vorliegende Text; hier werden die Grundzüge der Methode von Holtrop und Mennen vorgestellt, sowie die bei der Implementierung verwendeten Techniken und Verfahren erläutert. Die prinzipielle Strukturierung des Programms sowie das für die Ermittlung des optimalen Propellers verwendete Optimierungsverfahren werden beschrieben.

2. **Software:** Das Programm RaPPPS ; auf der beiliegenden CD-ROM befinden sich Verzeichnisse und Dateien, die u.a. den Quellcode und ausführbare Dateien des Programms enthalten. Eine genaue Auflistung befindet sich im Anhang B.2.

RaPPPS wird unter der GNU General Public License (GPL) veröffentlicht; das Programm wird im Internet unter der URL http://www.pascal-anschau.de/rappps/ zum Download bereitgestellt.

Inhaltsverzeichnis

Einleitung **7**

1 Beschreibung des Verfahrens von Holtrop und Mennen **9**
1.1 Berechnung des Widerstandes 9
1.2 Berechnung der Propulsionseigenschaften 12
1.2.1 Ermittlung des Arbeitspunktes des Propellers 13
Berechnung von K_T und K_Q 13
Reynolds - und Rauhigkeitskorrektur 14
Berechnung von Freifahrtwirkungsgrad η_O und Drehzahl n 14
1.2.2 Berücksichtigung der Kavitationsfreiheit 17
1.3 Berechnung temperaturabhängiger Größen 20
1.3.1 Wasserdichte ρ 20
1.3.2 Kinematische Viskosität ν 20
1.3.3 Dampfdruck P_V 20

2 Das Optimierungsproblem **21**
2.1 Formulierung und Ausarbeitung im Standardformat 21
2.1.1 Praktische Einschränkungen 21
2.1.2 Annahmen 22
2.1.3 Ausarbeitung 22
2.1.4 Formulierung 23
2.2 Nebenbedingungen 24
2.3 Darstellung des Suchraums 24

3 Implementierung des Optimierungsproblems **28**
3.1 Auswahl eines Optimierungs - Verfahrens 28
3.2 Der Simplex - Algorithmus 29
3.2.1 k - te Iteration nach Nelder-Mead 30
3.3 Normierung der Wertebereiche 31
3.4 Einhaltung der Nebenbedingung und Wertebereiche 31
3.4.1 Einhaltung der Nebenbedingung 31
3.4.2 Einhaltung der Wertebereiche der Variablen 32
3.5 Abbruchkriterien 32

4 Das Programm RaPPPS **34**
4.1 Die Programmiersprache Tcl/Tk 34
4.2 Grundlegende Anforderungen an RaPPPS 35
4.3 Objektorientierung und Datenstruktur 35

4.3.1 Das Datenobjekt: Klasse RaPPPS_Data 35
4.3.2 Organisation in Projekten 37
4.3.3 Die Datenstruktur eines Projektes 38
4.3.4 Erzeugen von Design - Varianten 38
4.4 Format einer Projektdatei 41
4.5 Stringkonstanten und dynamische Kurzhilfe 42
4.6 Konfigurationsdateien 44
4.6.1 Die Standardkonfiguration 44
4.6.2 Benutzerspezifische Konfiguration 46
4.7 Namenskonventionen 47
4.8 Datenfluss 48
4.9 Ablauf einer Berechnung und Auswertung 49
4.10 Kurzbeschreibung der Programmfenster von RaPPPS 53

A Verwendete Symbole **55**
A.1 Symbole nach ITTC 55
A.2 Von Holtrop und Mennen verwendete Symbole 59

B Zum Programm RaPPPS **60**
B.1 Kurzbeschreibung der Programm - Dateien 60
B.2 Inhalt der CD-ROM 63

Tabellenverzeichnis

1 Parameterbereiche 7

1.1 Verwendete Parameter zur Widerstandsbestimmung 10
1.2 Verwendete Parameter zur Propulsionsbestimmung 13

2.1 Wertebereiche Wageninger B-Serie 22
2.2 Parameter eines ausgeführten Beispiels 24

A.1 ITTC Symbole 55
A.2 Abkürzungen von Holtrop und Mennen 59

B.1 Programmdateien - Kurzbeschreibung 60

Abbildungsverzeichnis

1.1 Freifahrtdiagramm 15
1.2 Burrill - Diagramm 18
1.3 Ablauf der Berechnung 19

2.1 Suchraum ohne Nebenbedingung 26
2.2 Suchraum mit Nebenbedingung 27

3.1 Ablauf der Optimierung 33

4.1 Datenobjekt 37
4.2 Datenstruktur eines Projektes 39
4.3 Anordnung neuer Varianten 39
4.4 Dateiformat 43
4.5 Format der Konfigurationsdatei 47
4.6 Datenfluss 48
4.7 Erstellen eines neuen Projektes 51

Einleitung

In den Jahren 1977 - 1984 wurde von J. Holtrop und G. Mennen an der Niederländischen Schiffsbauversuchsanstalt in Wageningen[1] ein Verfahren entwickelt, das eine rein numerische Beschreibung des Schiffswiderstandes, der Propulsionseigenschaften sowie der Maßstabseffekte zwischen Modell und Großausführung ermöglicht ([1]). Konstrukteuren sollte damit ein einfaches und schnelles Mittel zu einer ersten hydrodynamischen Bewertung ihrer Entwürfe an die Hand gegeben werden. Mittels multipler Regressionsanalysen wurden mehrere tausend Widerstands- und Propulsionsversuche mit Modellen sowie Versuchsfahrten mit Schiffen parametrisch ausgewertet. Allein der ersten Veröfffentlichung (1977) lagen 1707 Widerstandsversuche, 1287 Propulsionsversuche und 82 Versuchsfahrten zugrunde; weitere kamen im Lauf der folgenden Jahre hinzu.

Type of ship	Fn	C_P		L/B		Number of ships			
						single screw		twin screw	
	max.	min.	max.	min.	max.	model	full scale	model	full scale
Tankers, bulkcarriers	0.24	0.73	0.85	5.1	7.1	48	13	3	2
General cargo	0.30	0.58	0.72	5.3	8.0	21	17	3	2
Fishing vessels, tugs	0.38	0.55	0.65	3.9	6.3	35	-	3	2
Container ships, frigates	0.45	0.55	0.67	6.0	9.5	6	-	18	1
Various	0.30	0.56	0.75	6.0	7.3	7	6	3	3
Total						117	36	30	10

Tabelle 1: Parameterbereiche der von Holtrop und Mennen untersuchten Schiffe.

Die Versuche wurden zunächst mit Verdrängerschiffen durchgeführt, mit Froudezahlen im Bereich von 0.24 - 0.45. In Tabelle sind einige Hauptmerkmale der in den Untersuchungen verwendeten Schiffe und Modelle aufgelistet; die untersuchten Schiffe decken einen recht grossen Bereich von üblichen und verbreiteten Schiffstypen ab.

Die Ergebnisse der Untersuchungen waren einige Dutzend Formeln, mit deren Hilfe sich der Schiffswiderstand und die Propulsionseigenschaften auf der Basis einiger weniger Formparameter berechnen lässt, üblicherweise mittels Computer. In den nachfolgenden Veröffentlichungen ([2], [3] und [4]) wurden diese

[1] Später: MARIN – Maritime Research Institute Netherlands, Wageningen, The Netherlands

Gleichungen teilweise modifiziert, das Prinzip blieb jedoch das gleiche. Obwohl sich die Rumpfformen in den zwanzig Jahren, die seit der letzten diesbezüglichen Veröffentlichung vergangen sind, noch gewandelt haben, ist dieses Verfahren - wahrscheinlich wegen seines statistischen Charakters - nach wie vor so aussagekräftig, dass es auf Werften weit verbreitete Anwendung findet.

Hauptsächliches Ziel des Verfahrens ist die Prognose der benötigten Antriebsleistung und der dafür notwendigen Propellerabmessungen und Drehzahl. Dabei sind auch zwei Polynome zur Beschreibung der Schub - und Drehmomentenbeiwerte K_T und K_Q des freifahrenden Propellers von Bedeutung, die von Oosterveld/van Oossanen auf Basis der Wageninger B-Serie Propeller einige Jahre zuvor entwickelt worden waren ([5]). Mittels dieser Polynome lassen sich K_T und K_Q in Abhängigkeit von Propellerblattzahl, Flächenverhältnis, Steigungsverhältnis und Fortschrittsziffer beschreiben. Somit ist es möglich, ohne weitere Diagramme, nur mittels des Computers, den Freifahrtwirkungsgrad η_O zu berechnen und damit die notwendige Wellenleistung für den Schiffspropeller zu bestimmen. Auch wenn heute mit CFD - Verfahren optimierte Propeller zum Einsatz kommen, die von der Wageninger B - Serie in Form und Wirkungsgrad deutlich abweichen, ist eine Leistungsprognose mittels dieses Verfahrens in der Phase des Vorentwurfs eine lohnenswerte Alternative.

Kapitel 1

Beschreibung des Verfahrens von Holtrop und Mennen

In diesem Kapitel wird zunächst kurz die prinzipielle Vorgehensweise bei der Ermittlung des Schiffswiderstandes (d.h. des Schiffsrumpfes inklusive der Anhänge) nach Holtrop und Mennen vorgestellt; anschließend wird die Bestimmung der Propulsionseigenschaften ausführlicher beschrieben. Formeln aus den Veröffentlichungen von Holtrop/Mennen werden jedoch nur dort angegeben, wo sie für das Verständnis des Rechenweges sinnvoll erscheinen; für die vollständige Darstellung der Gleichungen sei auf die entsprechenden Veröffentlichungen verwiesen.

1.1 Berechnung des Widerstandes

Analog zum ITTC - Modell der Aufteilung des Gesamtwiderstandes ergibt sich der Gesamtwiderstand R_T aus[1]:

$$R_T = R_F(1+k) + R_{AP} + R_W + R_B + R_{TR} + R_A$$

(R_B, R_{TR} siehe Tabelle A.2).
Sämtliche Gleichungen zur Bestimmung dieser Widerstandsanteile sind Funktionen einer Menge $\vec{P}$ von Parametern, die sich aus der Teilmenge der Eingabe-Parameter $\vec{P}_{Input}$ ergeben:

$$\vec{P}_{Input} \subseteq \vec{P}$$

In Tabelle 1.1 sind alle Parameter aufgelistet, die in den Gleichungen von Holtrop und Mennen zur Widerstandsbestimmung verwendet werden[2] (alle Werte beziehen sich auf den Entwurfstiefgang).

Der Plattenreibungswiderstand R_F wird mit dem Widerstandsbeiwert c_F nach der bekannten ITTC-1957 Formel bestimmt:

$$c_F = \frac{0.075}{(\log_{10} Rn - 2)^2} \qquad \rightsquigarrow \qquad R_F = 0.5\,\rho\,V^2\,S\,(1+k)\,c_F$$

[1] Abkürzungen siehe Anhang A.1

[2] Die mit * gekennzeichneten Parameter werden aus den Eingabeparametern berechnet; mit (*) sind Werte gekennzeichnet, die eingegeben werden *können*; alternativ werden sie berechnet.

Symbol	Definition
V	Schiffsgeschwindigkeit
L_{WL}	Länge der Wasserlinie
* L_R	Länge vom vorderen Lot zum vorderen Ende des parallelen Mittelschiffs
L_{CB}	Lage des Auftriebsschwerpunktes relativ zu $L_{PP}/2$
B	max. Breite Wasserlinie
T	Tiefgang
T_F	Tiefgang vorderes Lot
T_A	Tiefgang hinteres Lot
$\forall$	Volumen
C_M	Flächenkoeffizient Hauptspant
C_{WP}	Wasserflächenkoeffizient
* C_P	Schärfegrad
t_W°	Wassertemperatur (TEWA)
ρ	Dichte des Wassers
FW/SW	Frisch - oder Salzwasser
* ν	Kinematische Viskosität des Wassers
A_T	Eingetauchte Fläche Stern
A_{BT}	Querschnittsfläche des Bugwulstes am vorderen Lot
h_B	Höhe des Flächenschwerpunkt der Querschnittsfläche des Wulstbuges über Kiel
(*) S	Benetzte Oberfläche
k_S	Rauhigkeit des Schiffsoberfläche
(*) i_ϵ	Eintrittswinkel Wasserlinie
d	Durchmesser der Öffnung für Bugstrahlruder
C_BTO	Relative Position des Bugstrahlruders
* C_B	Blockkoeffizient
C_{Stern}	Sternform
$(1 + k_2)$ - Werte	Formfaktoren für versch. Anhänge
* Fn	Froudezahl
* Rn	Reynoldszahl

Tabelle 1.1: Die in den Gleichungen von Holtrop und Mennen verwendeten Parameter zur Bestimmung des *Widerstandes*. Die mit * gekennzeichneten Parameter werden aus den Eingabeparametern berechnet; mit (*) sind Werte gekennzeichnet, die eingegeben werden *können*; alternativ werden sie berechnet.

Der Formfaktor k wird dabei maßgeblich von der Form des Sterns beeinflusst, für die – für vier veschiedene typische Formen – Koeffizienten angegeben sind.

Mit c_F wird (nach [3]) auch der Widerstand der Anhänge bestimmt:

$$R_{APP} = 0.5\ \rho\ V^2\ S_{APP}\ (1 + k_2)\ c_F$$

Dabei sind für verschiedene Arten von Anhängen mit der benetzten Gesamtoberfläche S_{APP} (Bilge-Kiele, Doppelruder, Stabilisierungsflossen, Skegs, etc.) verschiedene $(1+k_2)$ - Werte angegeben, die aus vergleichenden Widerstandsversuchen mit und ohne Anhängen gewonnen wurden. Die Widerstandserhöhung durch Bugstrahlruder-Öffnungen wird ebenfalls berücksichtigt.

Der Wellenwiderstand R_W wird (nach [4]) – in Abhängigkeit von der Froude-Zahl – mit drei unterschiedlichen Formeln berechnet. Dabei werden die Bereiche $Fn < 0.4$, $0.4 < Fn < 0.55$ und $0.55 < Fn$ unterschieden. Für den mittleren Bereich ist eine Interpolationsformel angegeben, die auf den Widerstandswerten für $Fn = 0.4$ bzw. $Fn = 0.55$ basiert:

$$R_W = R_{W,0.4} + \frac{(10\ Fn - 4)(R_{W,0.55} - R_{W,0.4})}{1.5}$$

R_B ist der Widerstandsanteil, der durch den Wulstbug nahe der Oberfläche verursacht wird. Nach [4] wird dieser mit einer auf die Querschnittsfläche des Wulstbuges am Vorderlot bezogenen, lokalen Froude-Zahl berechnet. Außerdem wird einer teilweisen Austauchung des Bugwulstes Rechnung getragen.

Der zusätzliche Widerstand R_{TR}, den ein teilweise eintauchendes Spiegelheck verursacht, wird (nach [3]) mit der eingetauchten Querschnittsfläche A_T des Hecks berechnet:

$$R_{TR} = 0.5\ \rho\ V^2\ A_T\ c_6$$

Der Koeffizient c_6 hängt dabei von einer lokalen Froude-Zahl ab, die auf die Breite und den Wasserflächenkoeffizienten bezogen ist.

Schließlich wird (nach [3]) der Rauhigkeitszuschlag berechnet, der nach der ITTC-1978 Formulierung auch an eine vom Standardwert abweichende Oberflächenrauhigkeit angepasst werden kann. Der Standardwert für die Rauhigkeit des Schiffsrumpfes wird hier mit $150\mu m$ angegeben.

1.2 Berechnung der Propulsionseigenschaften

In Tabelle 1.2 (S. 13) sind die Parameter aufgeführt, die zusätzlich zur Bestimmung der Propulsionseigenschaften benötigt werden.

Aus dem Produkt von Wellenleistung P_S, Gütegrad der Anordnung η_R, Freifahrtwirkungsgrad η_O, Wellenwirkungsgrad η_S und Schiffseinflussgrad η_H ergibt sich die effektive Schleppleistung des Propellers:

$$P_E = P_S * \eta_R \, \eta_O \, \eta_S \, \eta_H = P_S * \eta_R \, \eta_O \, \eta_S \, \frac{1-t}{1-w} = R_T * V \tag{1.1}$$

und damit die Wellenleistung zu:

$$P_S = \frac{P_E}{\eta_R \, \eta_O \, \eta_S \, \frac{1-t}{1-w}} \tag{1.2}$$

Der benötigte Schub ergibt sich unter Berücksichtigung der Sogziffer t zu:

$$T = \frac{R_T}{1-t} \tag{1.3}$$

Die Sogziffer t wird für ein- bzw. zweischraubige Schiffe nach unterschiedlichen Formeln berechnet. Dabei gilt (nach [4]):

Einschrauber: $t = f(B, L, T, D, C_P, XCB, Sternform)$
Zweischrauber: $t = f(B, T, D, C_B)$

Die Nachstromziffer w ist u.a. ebenfalls vom Propellerdurchmesser D abhängig, womit dem im Nachstrom arbeitenden Propeller Rechnung getragen wird. Es wird ebenfalls nach Ein - und Zweischraubern unterschieden:

Einschrauber: $w = f(L, B, T_A, D, S, C_V, C_P, C_M, XCB)$
Zweischrauber: $w = f(B, T, D, C_V, C_P)$

Der Freifahrtwirkungsgrad η_O ist bekanntlich:

$$\eta_O = \frac{J}{2\pi} * \frac{K_T}{K_Q} \tag{1.4}$$

Die Schub - und Drehmomentenbeiwerte K_T und K_Q werden mit den bereits erwähnten Polynomen von Oosterveld/van Oossanen berechnet; in Abschnitt 1.2.1 wird darauf ausführlich eingegangen. Die Ermittlung des optimalen Freifahrtwirkungsgrades ist das Kernstück bei der Ermittlung der optimalen Propellerparameter.
Der Gütegrad der Anordnung η_R ist das Verhältnis der Wirkungsgrade des Propellers unter Freifahrtbedingungen (η_O) und hinter dem Schiff (η_B) und ist (nach [3]) wieder für ein - und zweischraubige Schiffe unterschiedlich:

Einschrauber: $\eta_R = f(\frac{A_E}{A_O}, XCB, C_P)$
Zweischrauber: $\eta_R = f(\frac{P}{D}, XCB, C_P)$

Der Wellenwirkungsgrad wird bei Holtrop/Mennen in diesen Berechnungen mit $\eta_S = 0.99$ angenommen, ist in RaPPPS aber frei einstellbar.

Symbol	Definition
V	Schiffsgeschwindigkeit
D	Propellerdurchmesser
k_P	Rauhigkeit der Propelleroberfläche
η_S	Wellenwirkungsgrad
$\frac{A_E}{AO}$	Flächenverhältnis
$\frac{P}{D}$	Steigungsverhältnis
P_V	Dampfdruck des Wassers
Z	Anzahl Propellerblätter
N	Anzahl Propeller

Tabelle 1.2: Die in den Gleichungen von Holtrop und Mennen verwendeten Parameter zur Berechnung der *Propulsionseigenschaften*.

1.2.1 Ermittlung des Arbeitspunktes des Propellers

Berechnung von K_T und K_Q

Die von Oosterveld und van Oossanen ermittelten Polynome zur numerischen Ermittlung der Schub - und Drehmomentenbeiwerte K_T und K_Q sind Funktionen von

- Fortschrittsziffer J
- Steigungsverhältnis $\frac{P}{D}$
- Flächenverhältnis $\frac{A_E}{A_0}$
- Propellerblattzahl Z

Ausserdem werden noch Korrekturterme hinzugefügt (s. nächster Abschnitt), die eine Berücksichtigung des veränderten Widerstandes des Propellers bei anderen Reynoldszahlen als $2 * 10^6$ erlauben.
Die Polynome haben die folgende Form:

$$\begin{aligned} K_T &= f(J, \frac{P}{D}, \frac{A_E}{A_O}, Z) + \Delta f_{K_T}(\frac{P}{D}, \frac{A_E}{A_O}, Z, D, k_S) \\ &= \sum_i C_T * J^{s,i} \left(\frac{P}{D}\right)^{t,i} \left(\frac{A_E}{A_)}\right)^{u,i} Z^{v,i} + \Delta f_{K_T}, i = 1, ..., 39 \end{aligned} \tag{1.5}$$

$$\begin{aligned} K_Q &= f(J, \frac{P}{D}, \frac{A_E}{A_O}, Z) + \Delta f_{K_Q}(\frac{A_E}{A_O}, Z, D, k_S) \\ &= \sum_j C_Q * J^{s,j} \left(\frac{P}{D}\right)^{t,j} \left(\frac{A_E}{A_O}\right)^{u,j} Z^{v,j} + \Delta f_{K_Q}, j = 1, ..., 47 \end{aligned} \tag{1.6}$$

Dabei sind C_T bzw. C_Q die Koeffizienten des Polynoms; die ganzzahligen Exponenten (mit s,t,u,v gekennzeichnet) sind für K_T und K_Q verschieden.

Reynolds - und Rauhigkeitskorrektur

Da die $K_T,\ K_Q$ - Polynome für eine Reynoldszahl von $2 * 10^6$ ermittelt wurden, ist eine Anpassung für andere Reynoldszahlen bzw. andere Rauhigkeitswerte des Propellers erforderlich, die nach der ITTC78 - Methode durchgeführt wird:

$$K_{T,Ship} = K_T + \Delta f_{K_T} = K_T + \Delta C_D * 0.3 \frac{P * c_{0.75} * Z}{D^2}$$

und

$$K_{Q,Ship} = K_Q + \Delta f_{K_Q} = K_Q - \Delta C_D * 0.25 \frac{c_{0.75} * Z}{D}$$

Dabei ist ΔC_D die Differenz des Widerstandskoeffizienten des Propellerprofils:

$$\Delta C_D = \left(2 + 4\left(\frac{t}{c}\right)_{0.75}\right)\left[0.003605 - (1.89 + 1.62\log(\frac{c_{0.75}}{k_S}))^{-2.5}\right]$$

$(\frac{t}{c})_{0.75}$ das Dickenverhältnis des Propellerprofils, und $c_{0.75}$ die Profilsehnenlänge des Propellers, jeweils bei 75% des Propellerradius.
Schliesslich ist k_S die Rauhigkeit der Propelleroberfläche und wird mit einem Standardwert von $k_S = 0.00003\ m$ angenommen.

Durch die Korrekturterme Δf_{K_T} bzw. Δf_{K_Q} werden die K_T, K_Q - Polynome zu Funktionen, die zusätzlich zu $J, \frac{P}{D}, \frac{A_E}{A_0}, Z$ auch vom Propellerdurchmesser D abhängen:

$$K_T,\ K_Q := f(J, \frac{P}{D}, \frac{A_E}{A_0}, Z, D) \tag{1.7}$$

Im folgenden ist mit K_T bzw. K_Q immer der korrigierte Wert $K_{T,Ship}$ bzw. $K_{Q,Ship}$ gemeint.

Berechnung von Freifahrtwirkungsgrad η_O und Drehzahl n

Unter der Annahme von Schubidentität ist nun zunächst die Fortschrittsziffer (und damit die Drehzahl) gesucht, bei der ein Propeller mit der durch $\frac{P}{D}$, $\frac{A_E}{A_0}$, D und Z festgelegten Geometrie den benötigten Schub T erzeugt. Das geschieht mittels der (parabolischen) Schubbedarfskurve des Schiffes:

$$f(J) = \left[\frac{K_T}{J^2}\right]_P * J^2 \tag{1.8}$$

Dabei ist:

$$\left[\frac{K_T}{J^2}\right]_P = \frac{R_T}{\rho * D^2(1-t)(1-w)^2 * V_S^2} \tag{1.9}$$

Der Arbeitspunkt des Propellers liegt bei der Fortschrittsziffer, bei der sich die Kurven von K_T und Schubbedarf schneiden (in Abbildung 1.1 ist ein Freifahrtdiagramm mit Schubbedarfskurve dargestellt). Dieser Punkt wird mit dem Newton-Verfahren ermittelt, das allgemein folgendermaßen lautet:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Abbildung 1.1: Beispiel für ein Freifahrtdiagramm mit Schubbedarfsparabel des Schiffes; Ausgabe des Programms RaPPPS .

Hierbei entspricht x der Fortschrittsziffer J.
Der Schnittpunkt ergibt sich bei Gleichheit von K_T und Schubbedarf, sodass mit

$$f_1 = K_T(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z)$$

und

$$f_2 = f(J) = \left[\frac{K_T}{J^2}\right]_P * J^2$$

die Funktion f lautet:

$$\begin{aligned} f(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z) :=& f_1 - f_2 \\ =& K_T(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z) - \left[\frac{K_T}{J^2}\right]_P * J^2 \stackrel{!}{=} 0 \end{aligned} \tag{1.10}$$

Wendet man auf diese Funktion das Newton-Verfahren an, ergibt sich:

$$\begin{aligned} J_{k+1} &= J_k - \frac{f(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z)}{\frac{d}{dJ} f(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z)} \\ &= J_k - \frac{K_T(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z) - \left[\frac{K_T}{J^2}\right]_P * J^2}{\frac{d}{dJ}(K_T(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z) - \left[\frac{K_T}{J^2}\right]_P * J^2)} \\ &= J_k - \frac{K_T(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z) - \left[\frac{K_T}{J^2}\right]_P * J^2}{\frac{d}{dJ} K_T(J, \frac{P}{D}, \frac{A_E}{A_0}, D, Z) - 2 * \left[\frac{K_T}{J^2}\right]_P * J} \end{aligned} \tag{1.11}$$

Die Ableitung des K_T - Polynoms lautet dabei:

$$\frac{dK_T}{dJ} = \sum_i C_T * sJ^{(s,i)-1} t \left(\frac{P}{D}\right)^{(t,i)-1} u \left(\frac{A_E}{A_)}\right)^{(u,i)-1} vZ^{(v,i)-1} \tag{1.12}$$

Die in Abschnitt 1.2.1 erläuterten Korrekturterme für K_T und K_Q sind bezüglich J konstant und entfallen daher.
Das Verfahren bricht ab, wenn eine Konvergenzschranke ΔJ unterschritten wird, d.h. wenn gilt:

$$|J_{k+1} - J_k| \leq \Delta J$$

Eine ausreichende Genauigkeit von J wird in der vorliegenden Implementierung angenommen bei:

$$\Delta J = 0.0001$$

Mit

$$J = \frac{2\pi\eta_O K_Q}{K_T} = \frac{V_A}{nD} = \frac{V_S(1-w)}{nD} \tag{1.13}$$

ist die für diesen Propeller und die geforderte Schubleistung optimale Drehzahl damit bekannt.
Ergänzend wird in RaPPPS noch die Möglichkeit einer Einflussnahme auf den endgültigen Wert von D geboten. Da die hydrodynamischen Eigenschaften des Schiffes sich im Lauf der Zeit etwas verschlechtern werden (w wird größer) und der Propeller hinter dem Schiff nicht in einem homogenen Strömungsfeld arbeitet (was bei der Ermittlung von η_O aber angenommen werden kann), wird sich der Propellerwirkungsgrad im Betrieb verringern. Nach Gleichung 1.13 wird dabei J kleiner. Ist der Propeller zunächst auf den optimalen Freifahrtwirkungsgrad ausgelegt, wird dieser bei sich verringerndem J ebenfalls abfallen. Um im Betrieb möglichst optimal zu arbeiten, wird der Propeller daher mittels des Faktors D_{mod} etwas verkleinert und damit auf zunächst etwas größere J eingestellt (rechts vom Maximum von η_O). Mit im Betrieb anwachsendem w wird J dann kleiner und der Wirkungsgrad verbessert sich hin zum Maximum:

$$J = \frac{V_A}{nD} * \frac{1}{D_{mod}} \tag{1.14}$$

Auf diese Weise wird durch Verringerung des Faktors D_{mod} die Effizienz des Propellers im Betrieb erhöht.
Aus der so ermittelten Fortschrittsziffer J ergibt sich mit

$$n = \frac{(1-w) * V_S}{J * D * D_{mod}} \tag{1.15}$$

die Drehzahl des Propellers, die etwas höher als die optimale Drehzahl ist, wenn $D_{mod} < 1$ ist. Weiterhin ergibt sich mit diesem J und den aktuellen Werten für $\frac{A_E}{A_O}$, $\frac{P}{D}$, D und Z nun der Drehmomentenbeiwert K_Q des freifahrenden Propellers, und mit Gleichung 1.4 der Freifahrtwirkungsgrad η_O.

Damit sind alle zur Bestimmung der Wellenleistung notwendigen Werte bekannt.

1.2.2 Berücksichtigung der Kavitationsfreiheit

Eine Bedingung, die der berechnete Propeller erfüllen soll, ist Kavitationsfreiheit. Zur Bestimmung der Kavitationssicherheit wird, abweichend von dem in [4] vorgeschlagenen Verfahren, das Kriterium von Burrill herangezogen. Bei Holtrop/Mennen wird die Erhöhung der Drehzahl bzw. der Wellenleistung bestimmt, die bei kavitierendem Propeller notwendig werden würde. Da partielle Kavitation generell vermieden werden sollte und bei der Berechnung eines solchen Propellers i.A. nicht von der B - Serie ausgegangen wird, ist in RaPPPS ein modifiziertes Verfahren nach Burrill implementiert.
Nach Burrill ist die Propellerbelastung für normale Handelsschiffe:

$$\tau_{c,Burrill} = 0.3\sqrt{\sigma_{0.7R}} - 0.03 \tag{1.16}$$

mit der Kavitationszahl $\sigma_{0.7}$, die bei 70% des Propellerradius herrscht.
$\sigma_{0.7}$ ergibt sich aus der bezogenen Differenz von Umgebungsdruck P_0 und Dampfdruck P_V des Wassers, der temperaturabhängig berechnet wird. Dabei ist zu beachten, dass Burrill den Umgebungsdruck auf die Eintauchtiefe h_0 der Propellernabe bezieht und nicht auf die höchste Position des Propellerprofils

Abbildung 1.2: Burrill - Diagramm zur Kavitationssicherheit. Das im Text genannte Burrill - Kriterium gilt fr „SUGGESTED UPPER LIMIT (1943)(FOR MERCHANT SHIP PROPELLERS)“.

bei 0.7*R, wo die Kavitationsgefahr am grössten ist. Mit $P_0 = P_{atm} + \rho * g * h_0$ ergibt sich also:

$$\sigma = \frac{P_0 - P_V}{0.5 * \rho * v_{0.7R}^2}$$

Die Anströmgeschwindigkeit $v_{0.7R}$ des Profils bei 0.7*R ergibt sich mit $v_A = V_S * (1 - w)$ und der Propellerdrehzahl n zu:

$$v_{0.7R} = \sqrt{v_A^2 + (0.7 * \pi * n * D)^2}$$

Der für die aktuelle Propellergeometrie geltende Wert von τ_c wird bestimmt nach:

$$\tau_{c,vorhanden} = \frac{T}{0.5 * \rho * v_{0.7R}^2 * A_P} \tag{1.17}$$

Für die Berechnung der projizierten Propellerfläche A_P kann die folgende Näherungsformel benutzt werden:

$$A_P = A_D(1.067 - 0.229\frac{P}{D})$$

Für Propeller ohne Hang (wie z.B. die B-Serie) ist die abgewickelte Propellerfläche A_D ungefähr gleich der gestreckten Fläche A_E, sodass gilt:

$$A_D \approx A_E$$

Da das aktuelle Flächenverhältnis $\frac{A_E}{A_0}$ bekannt ist (es ist ja ein Parameter für die K_T, K_Q - Polynome), kann man mit

$$A_0 = \frac{\pi * D^2}{4}$$

annehmen:

$$\frac{A_E}{A_0} \approx \frac{A_D}{A_0} \rightsquigarrow A_D \approx A_0 * \frac{A_E}{A_0} = \frac{\pi * D^2}{4} * \frac{A_E}{A_0}$$

und damit

$$A_P = \frac{\pi * D^2}{4} * \frac{A_E}{A_0}(1.067 - 0.229\frac{P}{D})$$

Die Bedingung, die bei Kavitationsfreiheit erfüllt sein muss, ist nun:

$$\tau_{c,vorhanden} - \tau_{c,Burrill} \overset{!}{\leq} 0 \tag{1.18}$$

Zur Beeinflussung der Bedingung der Kavitationssicherheit wird der Faktor $\tau_{c,X}$ eingeführt, mit dem mit Gleichung 1.16 gilt:

$$\tau_{c,Burrill} = (0.3\sqrt{\sigma_{0.7R}} - 0.03) * \tau_{c,X} \tag{1.19}$$

D.h., mit Erhöhung des Faktors $\tau_{c,X}$ wird die Grenze zum kavitierenden Bereich „nach oben“ verschoben (s. Abb. 1.2), und der Propeller kann als weniger gefährdet durch Kavitation eingestuft werden, da $\tau_{c,vorhanden}$ entsprechend größer werden darf.

Zusammenfassend ist in Abbildung 1.3 der Verlauf der Berechnungen nach Holtrop und Mennen als Flussdiagramm dargestellt.

Abbildung 1.3: Ablauf der Berechung nach Holtrop/Mennen .

1.3 Berechnung temperaturabhängiger Größen

Ausserhalb des Verfahrens von Holtrop und Mennen werden drei physikalische Größen, deren Wert eine Funktion der eingestellten Wassertemperatur ist, berechnet: die Wasserdichte ρ, die kinematische Viskosität ν und der Dampfdruck P_V. Die Gleichungen für ρ und P_V wurden dabei mittels Regressionsanalyse nach Datenreihen (s. [19], [20]) ermittelt.

1.3.1 Wasserdichte ρ

Die Gleichung für die Wasserdichte ρ ist für Salz - bzw. Süßwasser unterschiedlich. Es gilt:

$$\begin{aligned} \rho_{Süß} &= -0.000002 * T^4 + 0.0002 * T^3 - 0.0113 * T^2 + 0.0884 * T + 999.8 \\ \rho_{Salz} &= 0.000003 * T^4 - 0.0001 * T^3 - 0.0043 * T^2 - 0.058 * T + 1028 \end{aligned}$$

1.3.2 Kinematische Viskosität ν

Die kinematische Viskosität ν wird abhängig von der Art des Wassers (Salz - oder Süßwasser) berechnet. Dabei gelten folgende Gleichungen mit der Wassertemperatur T:

$$\begin{aligned} \nu_{Süß} &= ([0.000585(T-12) - 0.03361](T-12) + 1.235) * 10^{-6} \\ \nu_{Salz} &= ([0.000659(T-1) - 0.05076](T-1) + 1.7688) * 10^{-6} \end{aligned}$$

1.3.3 Dampfdruck P_V

Der Dampfdruck berechnet sich nach folgender Formel:

$$P_V = 0.0005 * T^4 + 0.0221 * T^3 + 1.4377 * T^2 + 44.883 * T + 609.32$$

Da bei der Messung der zugrunde liegenden Daten wahrscheinlich kein Seewasser verwendet wurde, dürfte der tatsächliche Wert etwas höher liegen.

Kapitel 2

Das Optimierungsproblem

Das Problem, die minimale erforderliche Antriebsleistung zu berechnen, erweist sich als nicht geschlossen analytisch lösbar, da die Funktion (insbesondere wegen der K_T, K_Q - Polynome, s. Gleichungen 1.5 und 1.6) nichtlinear bezüglich der Variablen $\frac{A_E}{A_0}$, $\frac{P}{D}$ und D ist. Daher kommt zur Lösung ein Verfahren der sog. „Nichtlinearen Programmierung“ zum Einsatz. Im folgenden wird das Problem zunächst formal beschrieben; im nächsten Kapitel wird dann das numerische Verfahren beschrieben.

2.1 Formulierung und Ausarbeitung im Standardformat

Für das Schiff, das durch die in den Abschnitten 1.1 und 1.2 genannten Parameter definiert ist, soll der Propeller (nebst zugehöriger Drehzahl) gefunden werden, der mit minimaler Maschinenleistung den für die Dienstgeschwindigkeit erforderlichen Schub erzeugt.

2.1.1 Praktische Einschränkungen

Für die Wertebereiche, in denen die Variablen variiert werden können, bestehen folgende Ober - bzw. Untergrenzen, die durch Benutzereingaben jedoch auch weiter eingeschränkt, aber nicht erweitert werden können:

- Für Propeller der Wageninger B - Serie gelten folgende Grenzen für das Steigungsverhältnis:

$$0.6 \leq \frac{P}{D} \leq 1.4$$

- Für das Flächenverhältnis $\frac{A_E}{A_0}$ eines B - Serien - Propellers gelten die in Tabelle 2.1 aufgeführten Grenzen in Abhängigkeit von der Blattzahl.[1]

[1] Die Angaben dieser Grenzen sind in der Literatur nicht einheitlich: In [15], S. 166, Tabelle 6.4 sind z.B. für zwei- bzw. dreiflügelige Propeller keine Obergrenzen des Flächenverhältnisses angegeben; die Unter - und Obergrenzen steigen mit der Blattzahl an. In [11] werden alle Berechnungen unabhängig von der Propellerblattzahl mit $\frac{A_E}{A_0} \in [0.3; 1.05]$ durchgeführt. Ebenfalls in [15] sind für das Steigungsverhältnis Grenzen von $0.6 - 1.4$ angegeben; in [11] werden alle Propeller aber mit $\frac{P}{D} \in [0.5; 1.4]$ berechnet.

- Die Grenzen des Propellerdurchmessers sind konstruktiv bedingt und werden in RaPPPS durch Benutzereingaben festgelegt:

$$D_{llim} \leq D \leq D_{ulim}$$

Blattzahl	$\frac{A_E}{A_0\ min}$	$\frac{A_E}{A_0\ max}$
2	0.30	0.8
3	0.35	0.95
4	0.40	1.0
5	0.45	1.05
6	0.50	1.05
7	0.55	1.05

Tabelle 2.1: In RaPPPS implementierte Wertebereiche für die Flächenverhältnisse der Wageninger B - Serien - Propeller.

2.1.2 Annahmen

Die Dimensionierung des Propellers für ausreichende strukturelle Festigkeit kann in dem vorliegenden Optimierungsproblem vernachlässigt werden.
Die Anzahl der Propellerblätter Z ist durch andere konstruktive Entscheidungen (Kosten, Schwingungen, etc.) bereits festgelegt.
Die Anzahl der Propeller ist zu Beginn der Optimierung ebenfalls bereits festgelegt.
Der Gesamtwiderstand R_T des Schiffes ist bereits berechnet.

2.1.3 Ausarbeitung

Die Leistung berechnet sich nach Gleichung 1.2 zu:

$$P_S = \frac{P_E * (1-w)}{\eta_S\ \eta_O\ \eta_r\ (1-t)}$$

In den folgenden Beziehungen, die teilweise für ein - und zweischraubige Schiffe unterschiedlich sind, ist der funktionale Zusammenhang mit jeweils allen möglichen Abhängigkeiten angegeben; in den eigentlichen Formeln wird jedoch – abhängig von der Schraubenzahl – meist nur eine Teilmenge der angegebenen Parameter verwendet.
Es gelten die folgenden Beziehungen:

$$\begin{aligned}
\eta_r &:= f(\frac{A_E}{A_0}, C_P, lcb, \frac{P}{D}) \\
\eta_O &:= f(J, \frac{P}{D}, \frac{A_E}{A_0}, Z, D) \\
t &:= f(L_{WL}, B, T, D, lcb, C_P, C_{Stern}, C_B) \\
w &:= f(L_{WL}, B, T, D, lcb, C_P, C_{Stern}, C_B, S, C_M, C_V)
\end{aligned}$$

Die dem Optimierungsalgorithmus „zur Verfügung“ stehenden Variablen sind Steigungs - und Flächenverhältnis sowie der Propellerdurchmesser, sodass sich der Vektor der freien Variablen ergibt zu:

$$\vec{x} = (\frac{P}{D},\ \frac{A_E}{A_0}\ , D) = (x_1, x_2, x_3) \tag{2.1}$$

Die Fortschrittsziffer J (die in den K_T/K_Q- Polynomen als Variable erscheint) ist zwar variabel, ist aber für den Optimierungsalgorithmus nicht „verfügbar“, da sie nur intern zur Bestimmung des Arbeitspunktes und der Dregzahl des aktuellen Propellers benutzt wird.
Sogziffer t, Nachstromziffer w und die Drehzahl n sind abhängige Variablen:

$$\vec{d} = (t,\ w,\ n) = (d_1, d_2, d_3) \tag{2.2}$$

Alle anderen Werte sind vom Benutzer gemachte Eingaben, sodass sich der Vektor der Parameter ergibt zu:

$$\vec{P} = (L_{WL}, B, T, S, C_P, C_B, lcb, C_M, h_0, \\ V_S, N, Z, R_T, \eta_S, C_V, t_W^\circ, \rho, P_V, k_P, C_{Stern}) \tag{2.3}$$

Folgende Konstanten werden verwendet:

$$\text{Erdbeschleunigung } g = 9.80665\frac{m}{s^2}$$
$$\text{Atmosphï¿½ischer Druck } P_a = 101360\frac{N}{m^2}$$

2.1.4 Formulierung

Das Problem, die benötigte Leistung zu minimieren, lautet:
Minimiere unter Verwendung der Variablen $\frac{P}{D}$, $\frac{A_E}{A_0}$, D die erforderliche Wellenleistung P_S bei gegebenem Schubbedarf:

$$F(\vec{x},\ \vec{d},\ \vec{P}) = \frac{P_E * (1-w)}{\eta_S\ \eta_O\ \eta_r\ (1-t)} \overset{!}{\Longrightarrow} min \tag{2.4}$$

Folgende Relationen bestehen:

$$\begin{aligned}
R(1,2) \quad & 0.6 && \leq x_1 \leq 1.4 \\
R(3,4) \quad & \left(\frac{A_E}{A_0}\right)_{llim} && \leq x_2 \leq \left(\frac{A_E}{A_0}\right)_{ulim} \\
R(5,6) \quad & D_{llim} && \leq x_3 \leq D_{ulim} \\
R(7) \quad & && d_1 = f(L_{WL}, B, T, x_3, lcb, C_P, C_{Stern}, C_B) \\
R(8) \quad & && d_2 = f(L_{WL}, B, T, x_3, lcb, C_P, C_{Stern}, C_B, S, C_M, C_V) \\
R(9) \quad & && d_3 = \frac{(1-d_1) * V_S}{J * x_3}
\end{aligned}$$

Damit lautet die Gütefunktion:

$$F(\vec{x},\ \vec{d},\ \vec{P}) \quad = \quad \frac{R_T * V_S * (1-d_2)}{\eta_S * \eta_r * \frac{J * K_T(J,Z,x_1,x_2,x_3)}{2\pi * K_Q(J,Z,x_1,x_2,x_3)} * (1-d_1)} \overset{!}{\Longrightarrow} min \tag{2.5}$$

2.2 Nebenbedingungen

Das Minimierungsproblem

$$F(\vec{x},\ \vec{d},\ \vec{P}) = \frac{P_E * (1 - d_2)}{\eta_S\ \eta_O\ \eta_r\ (1 - d_1)} \overset{!}{\Longrightarrow} min$$

soll unter Einhaltung der Forderung nach Kavitationsfreiheit ausgeführt werden. Mit dem in Abschnitt 1.2.2 eingeführten Burrill - Kriterium ergibt sich (mit Gleichungen 1.16, 1.17 und 1.18) die folgende Ungleichheitsnebenbedingung:

$$\tau_{c,vorhanden} - \tau_{c,Burrill} \overset{!}{\leq} 0$$

$$g_1(\vec{x},\vec{d},\vec{P}) := \frac{\frac{R_T}{1-d_1}}{0.5 * \rho * [V_S^2(1-d_2)^2 + (0.7 * \pi * d_3 * x_3)^2] * x_2 \frac{\pi}{4} x_3^2}$$

$$- 0.3 * \sqrt{\frac{(P_a + \rho * g * h_0) - P_V}{0.5 * \rho * V_S^2(1-d_2)^2 + (0.7 * \pi * d_3 * x_3)^2}} - 0.03 \overset{!}{\leq} 0 \quad (2.6)$$

Solange diese Bedingung erfüllt ist, arbeitet der Propeller – nach Burrill – kavitationsfrei. Dabei kann $\tau_{c,Burrill}$ durch einen von 1 abweichenden Wert von $\tau_{c,X}$ (s. Gleichung 1.19) variiert werden, um die Grenze für die Kavitationsfreiheit „härter“ oder „weicher“ einzustellen.

2.3 Darstellung des Suchraums

In [4] ist ein ausgeführtes Beispiel für die Widerstands - und Leistungsprognose veröffentlicht. Dabei handelt es sich um ein (hypothetisches) zweischraubiges Schiff mit den folgenden Merkmalen:

Parameter	Wert	Parameter	Wert
L	50.0 m	A_{BT}	0 m^2
B	12.0 m	i_E	25°
T_F	3.1 m	C_M	0.78
T_A	3.3 m	L_{CB}	-4.5%L
∇	900 m^3	A_T	10 m^2
S_{AP}	50 m^2	$1 + k_2$	3
C_{Stern}	0	C_{WP}	0.8

Tabelle 2.2: Formparameter des Schiffes aus dem ausgeführten Beispiel in [4].

In den Abbildungen 2.1 und 2.2 (s. S. 26 und 27) ist für eine angenommene Dienstgeschwindigkeit von 30 Kn für dieses Schiff der Verlauf der Gütefunktion dargestellt, wobei in Abbildung 2.2 die Nebenbedingung der Kavitationsfreiheit berücksichtigt wurde. Die erforderliche Wellenleistung ist dabei über dem Propellerdurchmesser und dem Flächenverhältnis für 9 verschiedene Steigungsverhältnisse aufgetragen, da bei drei Variablen eine räumliche Gesamtauftragung

des Suchraums nicht mehr möglich ist. Trotzdem gewinnt man einen guten Eindruck vom möglichen Verlauf der Gütefunktion; besonders deutlich wird auch die Wirkung der Nebenbedingung (Kavitationsfreiheit), die den zulässigen Suchraum mit wachsendem Steigungsverhältnis immer weiter einschränkt, und zwar in Richtung großer Propellerdurchmesser und Flächenverhältnisse.

Der hier exemplarisch dargestellte Verlauf der Gütefunktion legt die Vermutung nahe, dass sie ein einzelnes globales Minimum hat; dieser Suchraum gilt jedoch nur für diesen *einen* Fall. Um eine allgemein gültige Aussage über die Modalität der Gütefunktion treffen zu können, müsste man nachweisen, dass die Gütefunktion konvex ist, d.h. dass die Krümmung der (Hyper-)Fläche des *zulässigen* Suchraums überall auf dem untersuchten Gebiet positiv ist; das bedeutet, dass die Hesse-Matrix der Gütefunktion positiv definit sein muss. Da das vorliegende Problem eine Nebenbedingung enthält, müsste zunächst eine neue, *un*bedingte Gütefunktion aus der Qualitätsfunktion und der Nebenbedingung erzeugt werden, z.B. in Form einer Lagrange - Funktion:

$$\mathcal{L}(\vec{x},\vec{d},\vec{P},\mu) = F(\vec{x},\vec{d},\vec{P}) + \mu g_1(\vec{x},\vec{d},\vec{P})$$

Für $\mathcal{L}(\vec{x},\vec{d},\vec{P},\mu)$ müsste dann der o.g. Nachweis über die Hesse-Matrix durchgeführt werden, d.h. alle Eigenwerte von

$$\underset{=}{\mathcal{H}}(\mathcal{L}(\vec{x},\vec{d},\vec{P},\mu)) = \left[\underset{-}{\nabla^2}\mathcal{L}(\vec{x},\vec{d},\vec{P},\mu)\right] = \begin{bmatrix} \frac{\partial^2\mathcal{L}}{\partial x_1^2} & \frac{\partial^2\mathcal{L}}{\partial x_1 x_2} & \frac{\partial^2\mathcal{L}}{\partial x_1 x_3} \\ \frac{\partial^2\mathcal{L}}{\partial x_2 x_1} & \frac{\partial^2\mathcal{L}}{\partial x_2^2} & \frac{\partial^2\mathcal{L}}{\partial x_2 x_3} \\ \frac{\partial^2\mathcal{L}}{\partial x_3 x_1} & \frac{\partial^2\mathcal{L}}{\partial x_3 x_2} & \frac{\partial^2\mathcal{L}}{\partial x_3^2} \end{bmatrix}$$

müssen gleich Null werden. Da dieser Nachweis den Rahmen der vorliegenden Arbeit sprengen würde, wird auf diesen formalen Nachweis verzichtet.

Abbildung 2.1: Exemplarische Darstellung des Verlaufes der Gütefunktion am Beispiel der ausgeführten Rechnung aus [4], *ohne* Berücksichtigung der Forderung nach Kavitationsfreiheit.

Abbildung 2.2: Exemplarische Darstellung des Verlaufes der Gütefunktion am Beispiel der ausgeführten Rechnung aus [4], *mit* Berücksichtigung der Forderung nach Kavitationsfreiheit (die Ausrichtung der Achsen ist gegenüber der vorhergehenden Abbildung gedreht).

Kapitel 3

Implementierung des Optimierungsproblems

3.1 Auswahl eines Optimierungs - Verfahrens

Es sind sehr viele numerische Optimierungsverfahren bekannt, die sich grundsätzlich in die zwei Gruppen der „deterministischen“ und der „stochastischen“ Algorithmen einteilen lassen. Die Gruppe der deterministischen Algorithmen lässt sich ausserdem in zwei Untergruppen unterteilen: Methoden, die Ableitungs - bzw. Krümmungsinformationen der Gütefunktion benötigen, und solche, die nur Funktionswerte verwenden. Erstere werden *Gradientenmethoden* genannt, letztere *Suchmethoden*.
Die Auswahl eines für das zu lösende Problem geeigneten Algorithmus wird von mehreren Faktoren bestimmt; dazu gehören u.a.:

- Anzahl der freien Variablen.
- Uni-/Multimodalität der Gütefunktion (existieren – ggf. multiple – lokale/globale Minima?).
- Anzahl der Gütefunktionen.
- Benötigte Rechenzeit für eine Evaluation der Gütefunktion.
- Verfügbare Rechenleistung.
- Sind ggf. Ableitungen der Gütefunktion bekannt, oder müssen sie (wie aufwendig?) berechnet werden?
- Handelt es sich um eine lineare/nichtlineare Gütefunktion?

Das vorliegende Problem ist ein „kleines“ Problem – sowohl in bezug auf die Anzahl der Variablen als auch auf den benötigten Rechenaufwand:

Variablen: 3

Modalität: Wahrscheinlich unimodales Problem (s. auch Abschnitt 2.3).

Gütefunktionen: 1

Rechenzeit: Auf einem Pentium II mit 256 MHz Taktrate werden ca. 0.5 Sekunden für eine Iteration benötigt.

Rechenleistung: Die Rechnung wird lokal auf einem einzelnen Rechner ausgeführt.

Ableitungsinformationen: Sind grundsätzlich bekannt. Zweifache stetige Ableitbarkeit der Gütefunktion ist hier nicht nachgewiesen. Der Rechenaufwand wäre im Vergleich zur Evaluation der Gütefunktion wahrscheinlich größer, da wegen der Existenz einer Nebenbedingung z.B. mit Lagrange - Operatoren gearbeitet werden müsste. Das hätte die Lösung eines Gleichungssystems von $n+1=4$ Gleichungen zur Folge (Dimension $n=3$, Anzahl Nebenbedingungen $k=1$).

Linearität: Das Problem ist nicht linear.

Da RaPPPS in der Skriptsprache Tcl/Tk implementiert ist und Skriptsprachen bekanntlich zur Laufzeit in Maschinencode interpretiert werden, ist die Ausführungsgeschwindigkeit im Vergleich zu kompilierten Programmen um den Faktor 10 - 20 geringer. Daher fallen rein stochastische Algorithmen aus, da für eine verlässliche Aussage über die Lage des/eines Minimums leicht einige hundert oder mehr Evaluationen notwendig werden können.
Gradientenverfahren sind i. A. aufwendiger zu implementieren. Ausserdem kann ohne weitere Untersuchung der Gütefunktion nicht garantiert werden, dass die notwendige Bedingung der Stetigkeit erfüllt ist; es könnten also numerische Probleme durch die Differentiation auftreten. Daher wird in RaPPPS eine Suchmethode zum Einsatz kommen. Eine bekannte und recht einfach zu implementierende Suchmethode stellt der Simplex - Algorithmus nach Nelder und Mead (s. [16]) aus dem Jahr 1965 dar, der ein sog. „Amöben" - Algorithmus ist (so genannt wegen der quasi „einzelligen" Struktur und der simplen Art der Bewegung durch den Suchraum). Dieses Verfahren ist in RaPPPS zur Lösung des Optimierungsproblems implementiert.

3.2 Der Simplex - Algorithmus

Ein Simplex ist ein Satz von $n+1$ Punkten im $\Re^n$. Die Koordinaten eines Punktes entsprechen den Werten der Variablen $x_1 ... x_n$ des Variablenvektors $\vec{x}$ (s. Abschnitt 2.1.3). Jedem dieser $n+1$ Punkte ist ein Funktionswert $F_i(\vec{x})$ zugeordnet. Ein Simplex im $\Re^2$ ist ein Dreieck, im $\Re^3$ ergibt sich ein Tetraeder, für höhere Dimensionen gibt es keine geometrische Entsprechung.
Das vorliegende Problem ist eine Optimierung im $\Re^3$, da drei freie Variablen (x_1, x_2, x_3) existieren. Zu Beginn wird ein Anfangspunkt v_1 festgelegt, aus dem durch eine bestimmte Änderung entlang jeweils einer Koordinate die restlichen 3 Punkte $v_{2...4}$ abgeleitet werden:

$$v_1 = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, v_2 = v_1 + \begin{pmatrix} \alpha * x_1 \\ 0 \\ 0 \end{pmatrix}, v_3 = v_1 + \begin{pmatrix} 0 \\ \alpha * x_2 \\ 0 \end{pmatrix}, v_4 = v_1 + \begin{pmatrix} 0 \\ 0 \\ \alpha * x_3 \end{pmatrix}$$

In dieser Weise entsteht also eine Art Tetraeder, wobei in RaPPPS α zu 0.1 (normierte Koordinaten, s. Abschnitt 3.3) gesetzt ist. Der Startpunkt v_1 wird

in RaPPPS ebenfalls in normierten Koordinaten festgelegt:

$$v_1 = \begin{pmatrix} 0.1 \\ 0.8 \\ 0.8 \end{pmatrix} \tag{3.1}$$

Diese Werte (kleines Steigungsverhältnis, großes Flächenverhältnis, großer Propellerdurchmesser) liefern i.A. einen kavitationsfreien Propeller, was sich zur Vermeidung von verfrühter Konvergenz oder Degeneration des Simplex als sehr geeignet erweist.
Alle Punkte werden bezüglich ihres Funktionswertes und der Einhaltung der Nebenbedingung (s. Abschnitt 3.4.1) ausgewertet. Mit diesem Startsimplex beginnt der im folgenden kurz für eine beliebige Iteration k beschriebene Algorithmus nach Nelder und Mead.

3.2.1 k - te Iteration nach Nelder-Mead

Eine beliebige Iteration des Nelder-Mead-Algorithmus im $\Re^3$ hat den hier dargestellten Verlauf.
(Der Index i, mit $i = 1 \ldots 4$, bezeichnet im Folgenden die 4 Punkte des Simplex).

1. **Ordnen:** Ordnen der 4 Punkte entsprechend ihrer Funktionswerte, mit $F_1^{(k)} \leq F_2^{(k)} \leq F_3^{(k)} \leq F_4^{(k)}$.

2. **Reflektieren:** Berechnen des Schwerpunktes $\vec{c}$ des Simplex nach:

   $$\vec{c} = \frac{1}{n} \sum_{i=1}^{n} v_i \qquad\qquad n = 3$$

   Berechnung des Reflektionspunktes v_r nach:

   $$v_r = \vec{c} + (\vec{c} - v_4)$$

   Wenn $F_1 \leq F_r \leq F_3$ gilt, fortsetzen mit nächster Iteration.

3. **Expansionsschritt:** Wenn $F_r \leq F_1$ gilt, Berechnen des Expansionspunktes v_e nach:

   $$v_e = \vec{c} + 2 * (v_r - \vec{c})$$

   Wenn $F_e \leq F_r$ gilt, setze $v_4 = v_e$, sonst $v_4 = v_r$; fortsetzen mit nächster Iteration.

4. **Kontraktionsschritt:** Wenn $F_r \geq F_3$ gilt, Berechnen des Kontraktionspunktes v_c entsprechend einer der folgenden Bedingungen:
   Äußere Kontraktion: Wenn $F_3 \leq F_r \leq F_4$ gilt:

   $$v_c = \vec{c} + 0.5 * (v_r - \vec{c})$$

   Wenn $F_c \leq F_r$ gilt, setze $v_4 = v_c$ und setze mit der nächsten Iteration fort; sonst führe einen Verkleinerungsschritt durch (Schritt 5).
   Innere Kontraktion: Wenn $F_r \geq F_4$ gilt:

   $$v_c = \vec{c} + 0.5 * (v_4 - \vec{c})$$

   Wenn $F_c \leq F_4$ gilt, setze $v_4 = v_c$ und setze mit der nächsten Iteration fort; sonst führe einen Verkleinerungsschritt durch (Schritt 5).

5. **Verkleinerungsschritt:** Definition eines neuen, kleineren Simplex, ausgehend von Punkt v_1, nach:

$$v_i = v_1 + 0.5 * (v_i - v_1), \qquad i = 2 \ldots 4$$

Allgemein hat eine Iteration dieses Algorithmus zwei mögliche Ergebnisse: Entweder ergibt sich ein neuer Punkt, der dann den bisher schlechtesten Punkt, nämlich v_4, ersetzt; oder es wird (falls kein Punkt gefunden wird, der besser ist als der bisher schlechteste) eine Verkleinerung des Simplex durchgeführt und erneut reflektiert. Auf diese Weise wird ein neu erzeugter Punkt immer besser sein als der bisher schlechteste im Simplex.

3.3 Normierung der Wertebereiche

Die Wertebereiche für die drei Variablen x_1, x_2, x_3 werden auf einen Bereich $x_i \in [0, 1]$ normiert, sodass innerhalb des Simplex - Algorithmus mit normierten Koordinaten gerechnet werden kann, die erst bei der Evaluierung der Gütefunktion in „reale“ Variablenwerte umgerechnet werden müssen. Dabei gilt:

$$\frac{x_{real} - x_{llim}}{x_{ulim} - x_{llim}} = x_{norm}$$

wobei mit den Indizes *llim* bzw. *ulim* die untere bzw. obere Grenze des „realen“ Wertebereichs gekennzeichnet ist. Bei Umrechnung einer normierten Koordinate in reale Variablenwerte gilt dann:

$$x_{real} = x_{norm} * (x_{ulim} - x_{llim}) + x_{llim}$$

Z.B. ist die Mitte des Standardbereichs (0.6 - 1.4) für das Steigungsverhältnis $\frac{P}{D}$:

$$\left(\frac{P}{D}\right)_{real} = 0.5 * (1.4 - 0.6) + 0.6 = 1$$

3.4 Einhaltung der Nebenbedingung und Wertebereiche

3.4.1 Einhaltung der Nebenbedingung

Zur Einhaltung der Nebenbedingung wird eine Straffunktionsmethode mit einer „exterior penalty“ („äußere Strafe“) verwendet. Dabei wird dem Funktionswert des aktuellen Simplexpunktes ein „Strafterm“ aufaddiert, der sich aus dem größten Funktionswert des aktuellen Simplex ergibt:

$$\mathcal{P} = \max(F(v_1), F(v_2), F(v_3), F(v_4)) \tag{3.2}$$

Da zu Beginn des Optimierungslaufes die Funktionswerte der vier Punkte des Startsimplex erst sukzessive bekannt werden, wird als Wert für den Strafterm $\mathcal{P}$ zunächst ein Wert von 1000 kW festgelegt.

Solange die Nebenbedingung nicht verletzt ist, wird kein Strafterm zum Funktionswert addiert. Erst wenn gilt:

$$g_1 \geq 0$$

(s. auch Gleichung 2.6) wird $\mathcal{P}$ addiert und der gefundene Punkt damit künstlich derart „verschlechtert“, dass er im Laufe der Auswertung in den Schritten 2 bis 4 des Nelder-Mead-Algorithmus „ausscheidet“:

$$F(v_i) = F(v_i) + \mathcal{P}$$

Auf diese Weise wird verhindert, dass die gefundene Lösung ausserhalb des zulässigen Suchraums liegt.

3.4.2 Einhaltung der Wertebereiche der Variablen

Da die Variablenwerte immer innerhalb der (vom Benutzer eingestellten) gültigen Wertebereiche liegen sollen, muss das Verfahren garantieren, dass kein normierter Wert $x_{i,norm}$ ausserhalb des Intervalls $[0, 1]$ liegt. Das wird durch eine einfache Fallunterscheidung erreicht:

$$x_{i,norm} = \begin{cases} x_{i,norm} & \text{wenn} \quad x_{i,norm} \in [0,1] \\ 1 & \text{wenn} \quad x_{i,norm} > 1 \\ 0 & \text{wenn} \quad x_{i,norm} < 0 \end{cases}$$

3.5 Abbruchkriterien

Das Optimum (bzw. Minimum) ist erreicht, wenn eine oder mehrere Bedingungen erfüllt sind. In RaPPPS sind zwei Abbruchkriterien implementiert, die beide gleichzeitig erfüllt sein müssen:

1. Es werden die Funktionswerte F_1, F_2, F_3 der Simplexpunkte v_1, v_2 und v_3 verglichen; wenn die Differenz jeweils geringer ist als ein vom Benutzer einzustellender Wert ΔP, d.h. wenn gilt:

   $$|F_2 - F_1| < \Delta P \quad \wedge \quad |F_3 - F_2| < \Delta P$$

   ist das erste Abbruchkriterium erfüllt.

2. Die letzten drei Propeller-Designs müssen kavitationsfrei sein, d.h. für die Nebenbedingung g_1 der Punkte v_1, v_2 und v_3 muss gelten:

   $$g_1(v_1) \leq 0, \quad g_1(v_2) \leq 0, \quad g_1(v_3) \leq 0$$

Sind alle diese Bedingungen erfüllt, ist ein Minimum erreicht und der Optimierungslauf wird beendet.
Mit den auf diese Weise erhaltenen Werten für $\frac{A_E}{A_0}$, $\frac{P}{D}$ und D des letzten Designs wird nun noch der Leistungsbedarf für alle anderen Geschwindigkeiten berechnet.

Abbildung 3.1 (s. S. 33) zeigt schematisch den Ablauf der Optimierung.

Abbildung 3.1: Schematische Darstellung des Optimierungslaufes.

Kapitel 4

Das Programm RaPPPS

Die Software RaPPPS dient zur Prognose des Widerstandes und des Leistungsbedarfs eines Schiffes auf Grundlage eines Satzes von Formparametern. Daraus ist der Name der Software abgeleitet: RaPPPS – **R**esistance **a**nd **P**ropulsive **P**ower **P**rediction for **S**hips.

4.1 Die Programmiersprache Tcl/Tk

Tcl steht für *Tool Command Language* und meint sowohl eine Skriptsprache wie auch die entsprechende Interpreter - Umgebung. Ursprünglich für das Betriebssystem Unix entworfen, ist die Sprache heute auf allen gängigen Plattformen verfügbar und weit verbreitet. Der einzige Datentyp, den Tcl kennt, ist der String, also eine Zeichenkette.
Tk steht für *Toolkit* und bezeichnet eine Erweiterung der Tcl - Shell, die eine leichte Programmierung von graphischen Benutzeroberflächen (GUI - Graphical User Interface) ermöglicht.
In RaPPPS werden einige Zusatzpakete verwendet, die alle auf Tcl/Tk basieren. Diese stellen fertige Elemente, sog. *Widgets*[1], als Objekte zur Verfügung, die üblicherweise in GUIs Verwendung finden: Listenfelder, Buttons, Menüs, Statusbalken, Dialogboxen, etc. Die folgenden Pakete werden in RaPPPS verwendet (die Beschreibung spiegelt nur einen Teil des jeweiligen Paketumfangs wieder):

- **BLT:** Menüs, Statusbalken, Vektoren, diverse Diagrammtypen
- **BWidget:** Rahmen, Eingabefelder, Listboxen
- **Iwidgets:** Radiobuttons, Checkbuttons, Namespace, Spzifikation von Objektklassen
- **Tktable:** Tabelle
- **csv:** *comma separated values*, zur Formatierung von Komma - getrennten Strings

[1] Widget – Mischwort aus den englischen Worten *wizard* - *Magier* und *gadget* - *Dings, Apparatus*, das als Sammelbegriff für alle Arten graphischer Elemente einer Benutzeroberfläche verwendet wird (z.B. Fensterrollbalken, Buttons, Eingabefelder, etc.)

- **math:** Mathematische Standardoperationen

4.2 Grundlegende Anforderungen an RaPPPS

Da eine formale Spezifikation der Anforderungen im Sinne des klassischen „Pflichtenheftes“ den Rahmen dieser Dokumentation übersteigen würde, werden im Folgenden überblicksartig die grundlegendsten Anforderungen an das Programm aufgelistet:

- **Zielsetzung:** Die Software soll in erster Linie für Parameterstudien zur Widerstands - und Leistungsprognose im Vorentwurf eingesetzt werden können; darüber hinaus soll ggf. ein späterer produktiver Einsatz möglich sein.
- **GUI:** Die komfortable Eingabe/Veränderung der Parameter für die Widerstands - und Propulsionsberechnung soll durch eine benutzerfreundliche, mausgesteuerte graphische Oberfläche möglich sein.
- **Graphische Auswertung:** Alle für die hydrodynamische Bewertung des Schiffes relevanten Datensätze sollen als Diagramme darstellbar sein. Der unmittelbare Vergleich verschiedener Designs zum Zwecke .o.g. Parameterstudien soll sehr einfach möglich sein.
- **Daten:** Die Auswertung soll ausserdem alle relevanten Daten in numerischer Form verfügbar machen.
- **Export der Ergebnisse:** Alle Diagramme und Daten sollen zur Weiterverarbeitung exportierbar sein.
- **Reproduzierbarkeit:** Die entwickelten Design - Varianten sollen zur späteren Reproduzierbarkeit gespeichert und vom Programm auch wieder gelesen werden können.
- **Optional:** Die Ausgabe der Ergebnisse soll an die Erfordernisse/Vorlieben der Benutzer angepasst werden können.

4.3 Objektorientierung und Datenstruktur

(Im Folgenden werden Dateinamen sans-serif geschrieben, und Prozedurnamen in `Maschinenschrift`).

4.3.1 Das Datenobjekt: Klasse RaPPPS_Data

Mit der Erweiterung [incr Tcl/Tk] ist objektorientierte Programmierung in Tcl/Tk realisierbar. In RaPPPS wird von dieser Möglichkeit Gebrauch gemacht, indem ein eigenes (Daten-)Objekt als Klasse definiert wird. Alle in RaPPPS verwendeten Ein - und Ausgabedaten werden in solchen Datenobjekten gespeichert, wobei alle Daten *einer* Design - Variante (s. Abschnitt 4.3.3) in *einem* Datenobjekt gespeichert werden. Ausserdem werden einige Klassenmethoden definiert. Das folgende Code - Fragment (Datei sans-serif) zeigt die Definition des Datenobjektes **RaPPPS_Data** als eigene Klasse:

```
::itcl::class RaPPPS_Data {
   constructor {} {}
   destructor {}
   # Class data members.
   private {
      # Array for all input variables.
      variable all_inputs
      # Array for calculated output data.
      variable all_outputs
      }
   public  {
      # Program name.
      common progname
      # Version.
      common progversion
      # Homepage.
      common homepage
      }
   # Class methods.
   private {}
   # Class methods.
   public {
      method XXX... {} {}
      # Follow some method declarations.
   }
}
```

Die als *private variable* deklarierten Variablen **all_inputs** und **all_outputs** sind Array - Variablen, deren Elemente die verschiedenen Ein - bzw. Ausgabewerte bzw. Listen derselben sind. Sie verhalten sich auf Objekt - Ebene wie statische Variablen, sind also für die Lebensdauer eines einzelnen Datenobjektes in allen Funktionen und Methoden mit dem aktuellen Wert sichtbar, in denen sie bekannt gemacht werden.
Die als *public common* deklarierten Variablen sind statische skalare Variablen, die auf Klassenebene gültig sind; d.h., ihre Werte sind in allen Instanzen des Datenobjektes jederzeit gleich. Daher brauchen diese Werte nur einmal während des Programmlaufs initialisiert werden.
Es werden keine privaten Methoden benötigt, aber einige öffentliche Methoden, die im wesentlichen dem Zugriff auf die Daten dienen, die im Objekt gespeichert sind. Abbildung 4.1 zeigt schematisch den Aufbau der Klasse.
Wie bereits weiter oben erwähnt, wurde RaPPPS in einer Skript - Sprache

Abbildung 4.1: Schematische Darstellung des zentralen Datenobjektes von RaPPPS .

geschrieben, wodurch die Verarbeitungsgeschwindigkeit im Vergleich zu kompilierten Programmen deutlich geringer ausfällt. Daher wurde bei der Programmierung erheblicher Wert auf eine möglichst performante Implementierung gelegt; objektorientierte Programmierung ist aber mit einem nicht unerheblichen Overhead verbunden, da nicht nur der Variablen - Stack vom

Betriebssystem verwaltet werden muss, sondern zusätzlich auch die Speicherbereiche der Objekte und ihrer Methoden, die ja für jedes Objekt extra angelegt werden müssen. Andererseits bestand wiederum die Notwendigkeit einer praktikablen, selbstdefinierten Datenstruktur. Um den bei objektorientierter Programmierung unvermeidlichen Overhead möglichst gering zu halten, wurden nur Funktionen als Klassenmethoden programmiert, die in irgendeiner Weise Zugriff auf die im Objekt gespeicherten Daten benötigen. Alle anderen Funktionen wurden – in klassischer „prozeduraler“ Manier – als Prozeduren ausserhalb der Datenklasse implementiert. Dadurch verringert sich der Speicherbedarf pro Objekt und die Anzahl der Zugriffe auf Klassenmethoden wird ebenfalls deutlich verringert.

4.3.2 Organisation in Projekten

Daten werden in RaPPPS in sog. *Projekten* organisiert: Der Benutzer muss zu Beginn ein neues Projekt anlegen oder ein bereits vorhandenes laden, um Parameter - Varianten berechnen zu können. Es kann immer nur ein Projekt gleichzeitig geladen sein, das maximal 99 Varianten beinhalten kann. Ein Projekt ist gekennzeichnet durch den Projektnamen und den Geschwindigkeitsbereich, für den die Berechnungen durchgeführt werden sollen; diese Angaben können, wenn ein Projekt erst existiert, nicht mehr geändert werden.
Beim Anlegen eines Projektes werden daher folgende Angaben vom Benutzer gemacht:

1. Maximale Geschwindigkeit
2. Dienstgeschwindigkeit
3. Minimale Geschwindigkeit
4. Schrittweite der Unterteilung des vorher angegebenen Geschwindigkeitsbereiches

Dabei wird die Konsistenz der Angaben von RaPPPS geprüft: Die Dienstgeschwindigkeit darf nicht über der Maximalgeschwindigkeit liegen und nicht unter der Minimalgeschwindigkeit. Ebenso dürfen Maximal - und Minimalgeschwindigkeit sich nicht „überschneiden“:

$$V_{min} \leq V_{service} \leq V_{max}$$

Die Unterteilung mittels der Schrittweitenangabe wird an der Dienstgeschwindigkeit orientiert, d.h. die Bereiche über - und unterhalb derselben werden entsprechend der Schrittweite unterteilt. Wenn eine Schrittweite angegeben wird, mit der sich mit den eingegebenen Ober - und Untergrenzen keine ganzzahlige Schrittzahl erzeugen lässt, werden die Ober - und Untergrenzen bis zum nächsthöheren bzw. - kleineren Wert verschoben, mit dem/denen eine ganzzahlige Schrittzahl möglich ist. Die ursprünglich eingegebenen Grenzen sind dabei immer in den ggf. neuen Grenzen mit enthalten (`Check_correctness_of_input`, `Calc_intermediate_speeds`, Datei `create_new_project.tcl`).
Innerhalb eines Projektes können Varianten (bis maximal 99) beliebig erzeugt oder gelöscht werden, allerdings kann die letzte vorhandene Variante nicht mehr gelöscht werden, da sonst ein leeres Projekt entstehen würde. Dieser Spezialfall wäre dann äquivalent zu einem neu angelegten Projekt.

4.3.3 Die Datenstruktur eines Projektes

In RaPPPS können insgesamt 99 Varianten gleichzeitig existieren. Die dazu nötigen Datenobjekte der Klasse **RaPPPS_Data** werden im Array **data_objects** gespeichert; die Elemente dieses Arrays werden durch Integer - Zahlen von 1 bis 99 indiziert.

Abbildung 4.2: Schematische Darstellung der internen Datenstruktur eines Projektes von RaPPPS .

Dadurch wird es auf einfache Weise möglich, auf Objekte zuzugreifen. Als Beispiel gelte folgende Zeile:

```
$data_objects($histcount) Get_output_value vknprop prop
```

Hier wird aus dem Objekt, das mit dem Wert von **histcount** im Array **data_objects** indiziert ist, mittels der Methode `Get_output_value` ein bestimmter Wert der Ausgabedaten extrahiert; die Variable **histcount** enthält dabei die Nummer des aktuellen Objekts. Es entsteht die in Abbildung 4.2 veranschaulichte Datenstruktur.

4.3.4 Erzeugen von Design - Varianten

Neue Varianten werden an der aktuellen Stelle eingefügt: wird gerade z.B. die 3. Variation angezeigt, werden alle evtl. existierenden Varianten (im Beispiel sind das die Varianten 4 - 6) mit höheren Plätzen um eine Position „nach oben“ verschoben und ein neues Datenobjekt an der Stelle 4 eingefügt (Datei: create_new_variation.tcl).
Mit der Zeile

```
set data_objects($histcount) [RaPPPS_Data #auto]
```

wird ein neues Objekt erzeugt und in das Array **data_objects** mit dem Index **histcount** eingefügt. Das neue Objekt wird mit den Daten der 3. Variante vorinitialisiert, da davon ausgegangen wird, dass das zuvor angezeigte Design variiert werden soll. Auf diese Weise kann der Benutzer ohne hin - und herwechseln zwischen „entfernt“ liegenden Varianten das gerade interessierende Design überarbeiten und bekommt nach der Berechnung die Ergebnisse zum einfachen Vergleich direkt “nebeneinander liegend“ angezeigt.

Abbildung 4.3: Beispiel für die Anordnung der Variationen im Projekt vor und nach Erzeugen einer neuen Variante: Ein Duplikat von Objekt 3 wird erzeugt; Objekte 4 - 6 werden einen Platz „nach oben“ verschoben; das Duplikat wird als neues Objekt 4 eingefügt.

Wenn zwar schon eine neue Variation erzeugt wurde (also bereits als Datenobjekt existiert), aber die Berechnungen noch nicht durchgeführt wurden, existiert die Variante als sog. „anhängige Variation“ (engl. „pending variation“). Zu diesem Zeitpunkt enthält das Datenobjekt nur Eingabedaten, aber keinerlei berechnete Ausgabedaten. Es kann nur eine *pending variation* geben; daraus folgt, dass keine weitere Variation erzeugt werden kann, solange die *pending variation* nicht berechnet wurde.
Die Eingabedaten werden während der Eingabe im Array **gui_inputs** gespeichert und erst bei Beginn der Berechnung in das eigentliche Datenobjekt übertragen. Das ist notwendig, weil die mit den Eingabefeldern assoziierten Variablen nicht zu einem Objekt gehören *können*, da dieses zum Zeitpunkt der Erzeugung der Eingabefelder noch gar nicht existiert. Ausserdem sollen verschiedene Datenobjekte die Daten der Eingabemaske aufnehmen können, d.h. die Eingabefelder müssten mit den Variablen wechselnder Objekte assoziiert werden. Das ist zwar möglich, stellt aber eine weniger robuste und dabei aufwendigere Lösung dar.

4.4 Format einer Projektdatei

Ein - und Ausgabedaten werden im ASCII - Format gespeichert; dabei werden die Daten eines ganzen Projektes in einer Datei gespeichert. Die Dateiendung ist „.rappps“. Die Datei beginnt mit einer „magic number“[2] zur eindeutigen Identifizierung einer .rappps - Datei.
Nach dieser „magic number“ folgt der Dateikopf, der (in dieser Reihenfolge) die projektspezifischen Daten enthält: Projektname, Dienst -, Maximal - und Minimalgeschwindigkeit, Schrittweite, Anzahl der vorhandenen Variationen und ein Flag, das ggf. das Vorhandensein und die Position einer *pending variation* anzeigt.
Anschliessend folgt für jede Variante ein Abschnitt, der mit [variation] eingeleitet wird. Die Position jedes Datensatzes innerhalb der Datei entspricht der Position in der Liste der Varianten, wie sie in RaPPPS angezeigt wird.
In den nachfolgenden Fragmenten einer .rappps - Datei sind überlange Zeilen zur besseren Darstellung umgebrochen.

```
RaPPPS_1511831121141838121
[project]
initial_speed_defs(projectname) 50m twinscrew
initial_speed_defs(speedservice) 30
initial_speed_defs(speedmax) 35.0
initial_speed_defs(speedmin) 25.0
initial_speed_defs(speedstep) 1
initial_speed_defs(speeds) 25.0 26.0 27.0 28.0 29.0 30.0 31.0 32.0
        33.0 34.0 35.0
number_of_variations 3
flags(pending_variation) false
[variation]
all_inputs(opts,ks) 0.00015
all_inputs(res,draftchoice) drafttfta
```

[2] Diese „magic number“ setzt sich zusammen aus dem Programmnamen RaPPPS und dem mittels Alphabetpositionen kodifizierten Namen des Autors.

Abbildung 4.4: Schematische Darstellung des Dateiformats von RaPPPS .

```
all_inputs(res,thrusterdiam) 0.000
all_inputs(opts,rho) 1025.98
...
all_inputs(res,speeds) 25.00 26.00 27.00 28.00 29.00 30.00 31.00
        32.00 33.00 34.00 35.00
all_inputs(prop,upperbladearea) 0.8000
all_outputs(prop,ps) {6389.1} {6916.2} {7415.5} {7898.1} {8376.0}
        {8927.6} {9532.9} {10168.4} {10839.9} {11550.1} {12298.8}
all_outputs(prop,dp) 3.231
...
[variation]
all_inputs(opts,ks) 0.00015
```

Der erste Eintrag in jeder Zeile ist der vollständige Variablenname der jeweiligen Variable, das können sowohl Skalare als auch Arrays sein; im letzteren Fall steht der Index in (). Auf diese Weise können die Daten beim Einlesen einer Datei durch eine einfache Evaluierung des Variableneintrags an die Variablen im Programm zugewiesen werden (**line** ist dabei die Variable, die die aktuelle Zeile enthält):

```
eval "set [lindex $line 0] [list [lrange $line 1 end]]"
```

Die Reihenfolge der Variablen ist nur im Dateikopf festgelegt; in den Abschnitten der Varianten hängt die Reihenfolge von der Anordnung der Werte in den mit Hash - Tabellen realisierten Arrays von Tcl/Tk ab. Daher kann es zu unterschiedlichen Anordnungen sogar zwischen einzelnen Varianten-Datensätzen kommen. In jedem Variantenblock erscheinen allerdings die Eingabedaten vor den auszugebenden Daten. Wenn es eine *pending variation* gibt, enthält der entsprechende Datenblock nur die Eingabedaten.
Die maximale Dateigröße, d.h. bei 99 Varianten, beträgt ca. 550 KB.

Abbildung 4.4 zeigt schematisch den Aufbau einer Datei (s. auch: Save_procs.tcl, Save_project; class_methods.tcl, Write_data_to_file).

4.5 Stringkonstanten und dynamische Kurzhilfe

In der Datei strings.rappps sind sämtliche im Programm verwendeten Stringkonstanten sowie die Texte der dynamischen Kurzhilfe hinterlegt (sie wird nach ca. 0.5 Sekunden angezeigt, wenn der Mauszeiger über einem Objekt der graphischen Oberfläche steht, dem ein Hilfetext zugeordnet ist). Stringkonstanten werden in RaPPPS im Array **textuals** gespeichert und werden z.B. verwendet als: Tabellenköpfe, Bezeichner von Eingabefeldern, Menütitel, Schaltflächentexte, Texte in Diagrammen, Legendeneinträge, Fehlermeldungen u.v.m. Durch die Auslagerung dieser Texte ist eine sehr einfache Änderung möglich, ohne dass der Quelltext selber bearbeitet werden muss. Darüberhinaus ist eine zukünftige Übertragung des Programms in andere Sprachen als Englisch sehr einfach zu realisieren.

Die Datei strings.rappps hat folgendes Format:
Einzelne Bereiche sind thematisch bzw. nach Widget - Typ getrennt und beginnen jeweils mit einem Namen in eckigen Klammern. Mögliche Abschnitte werden mit einer kurzen Erläuterung nachfolgend aufgelistet:

- **[menu]**: Texte für Menüeintrage.
- **[toolbar]**: Kurzhilfetexte für Toolbar - Buttons.
- **[toplevel]**: Titel von Toplevel - Fenstern.
- **[notebook]**: Titel der einzelnen (notebook-) Seiten von RaPPPS .
- **[titleframe]**: Titel für Gruppierungsrahmen in der GUI.
- **[labelframe]**: Bezeichner in der GUI.
- **[label]**: Bezeichner in der GUI.
- **[labelentry]**: Bezeichner in der GUI.
- **[radiobutton]**: Bezeichner und Kurzhilfen für Radio - Buttons.
- **[checkbutton]**: Bezeichner und Kurzhilfen für Check - Boxen.
- **[spinbox]**: Bezeichner und Kurzhilfen für Spinboxen.
- **[combobox]**: Bezeichner und Kurzhilfen für Comboboxen.
- **[listbox]**: Bezeichner und Kurzhilfen für Listenfelder.
- **[scrolledframe]**: Bezeichner in der GUI.
- **[button]**: Texte und Kurzhilfen für Buttons.
- **[errors]**: Texte für Fehlermeldungen.
- **[wmtitle]**: Titel für diverse Fenster.
- **[messages]**: Text für Meldungen im Logger (s. Abschnitt 4.10 und Dialogboxen.
- **[table]**: Texte für Feldbezeichner und Tabellenköpfe.
- **[colhead]**: Tabellenköpfe.
- **[graphs]**: Texte für Bezeichner und Legenden in den Diagrammen.

Jede Zeile einer Sektion besteht aus maximal 3 Einträgen:
<Index im Array **textuals**> <Text in GUI> <Kurzhilfetext>
Bei Texten für Fehlermeldungen u.ä. entfällt der letzte Eintrag, da in diesen Fällen keine Kurzhilfe benötigt wird.

4.6 Konfigurationsdateien

4.6.1 Die Standardkonfiguration

In der Datei default_config.rappps sind Konfigurationsdaten abgelegt. Diese beinhalten Angaben zu verschiedenen Pfaden, die innerhalb von RaPPPS verwendet werden, Voreinstellungen z.B. für das Standard - Debugging - Level, einige physikalische Standardwerte und Voreinstellungen für das Erscheinungsbild von Linien und anderen Elementen in den Diagrammen (Linienstärke, Linienart, Farbe, Symbol, Symbolgröße). Die Verwendung einer externen Datei zur Ablage dieser Daten vereinfacht wiederum evtl. notwendig werdende Änderungen. Nachfolgend ist der Inhalt der Datei fragmentarisch wiedergegeben, ergänzt um eine kurze Erläuterung in jeder Zeile; die meisten Bezeichner sind allerdings selbsterklärend (zu lange Zeilen sind hier umgebrochen):

```
[overall]                          # Schluesselwort fuer allg.
                                   # Angaben
progname        "RaPPPS"           # Programmname
version         "1.0"              # Versionsnummer
strings_file    "strings.rappps"   # Name der Datei mit
                                   # Stringkonstanten
pictures_dir    "../pics"          # Pfad des Verzeichnisses mit
                                   # Icons und Bildern
help_dir        "../help"          # Pfad des Verzeichnisses mit
                                   # den HTML-Hilfedateien (s.u.)
lockfile        "rappps.lck"       # Name der Lock - Datei
personal_conf   "rappps.cfg"       # Name der Datei mit benutzer-
                                   # spezifischen Einstellungen
titlepic        "title.gif"        # Name des Titelbildes
maxvars         99                 # Maximale Anzahl von
                                   # moeglichen Varianten
magicnumber     RaPPPS_1511831121141838121          # Magic Number
homepage        http://www.pascal-anschau.de/rappps # URL der
                                   # Website von RaPPPS
kt_coeff_file   "kt_coeffs.rappps" # Datei mit den KT -
                                   # Koeffizienten und
                                   # - Exponenten
kq_coeff_file   "kq_coeffs.rappps" # Datei mit den KQ -
                                   # Koeffizienten und
                                   # - Exponenten
dkt_coeff_file  "dkt_coeffs.rappps" # Datei mit den
                                   # Koeffizienten und
                                   # Exponenten der
                                   # Ableitung des
                                   # KT - Polynoms
def_debuglevel  3                  # Standard - Debugging - Level
err_debuglevel  2                  # Debugging - Level fuer
                                   # Fehlermeldungen
sound           on                 # Standard fuer PC - Ton im
                                   # Fehlerfall
[graphs]                           # Schluesselwort fuer Abschnitt
```

Abbildung 4.5: Schematische Darstellung des Formats der Konfigurationsdatei(en) von RaPPPS .

```
                                    # mit Angaben zu
                                    # Diagrammkomponenten
styles,rt red 3 1 cross 4           # Beispiel fuer die Konfiguration
                                    # der Linie fuer RT
...
[table]                             # Schluesselwort fuer Angaben zur
                                    # Tabellenausgabe
output_order_res vknres vmpsres fn rn rfo rf rv rap rw rb rtr ra rt
        cfo cf cv cw ct             # Standardreihenfolge der
                                    # Tabellenspalten
formats,vknres %-10.2f              # Formatierungsangaben fuer
                                    # Datenausgabe (hier fuer die
                                    # Geschwindigkeit in Knoten bei
                                    # Widerstandsberechnung
...
[phys]                              # Schluesselwort fuer Abschnitt
                                    # mit (physikalischen)
                                    # Konstanten
...
etas 0.99                           # Standardwert des
                                    # Wellenwirkungsgrades
...
```

Die Datei ist in mehrere Abschnitte unterteilt, die durch folgende Schlüsselwörter in eckigen Klammern eingeleitet werden: [overall], [graphs], [table] und [phys]. Diese vier Schlüsselwörter sind Indizes im Array **config_specs** und bilden in diesem Array jeweils ein Sub - Array. Der erste Eintrag in einer Zeile stellt die restlichen Indizes dieser Sub - Arrays dar; so ist z.B. ein Eintrag im Abschnitt [graphs] noch in ein Sub - Array **styles** oder **ranges** unterteilt. Die **styles** - Variablen beziehen sich meist auf den Linienstil in den Diagrammen und haben folgendes Format:
<Farbe> <Strichlänge/-abstand> <Linienstärke> <Symbol> <Symbolgröße>
z.B.:
red 3 1 cross 4
Der Zugriff z.B. auf das Linien - Symbol für den Gesamtwiderstand RT erfolgt dann folgendermaßen:

```
[lindex $config_specs(graphs,styles,rt) 3]
```

Abbildung 4.5 zeigt schematisch die Struktur der Datei default_config.rappps.

4.6.2 Benutzerspezifische Konfiguration

Einige Konfigurationsdaten können vom Benutzer innerhalb von RaPPPS geändert werden (s. auch Abschnitt 4.10). Dazu gehören sämtliche graphischen

Darstellungsoptionen (also die im vorigen Abschnitt erwähnten **styles** - Angaben), die Reihenfolge der Tabellenspalten und die variablen physikalischen Standardwerte wie z.B. die Wassertemperatur (aus der dann die Dichte und die kinematische Viskosität resultiert). Diese Daten werden beim Beenden des Programms in der Datei rappps.cfg im Home - Verzeichnis des Benutzers gespeichert (Datei: save_procs.tcl).
Beim Starten des Programms wird zunächst die Standardkonfiguration geladen und anschliessend durch das Einlesen der Werte aus rappps.cfg die entsprechenden Werte der Konfiguration mit den Benutzereinstellungen überschrieben.

4.7 Namenskonventionen

Bei einem Projekt in der Größenordnung wie RaPPPS ist es notwendig, einige Konventionen bei der Vergabe von Variablennamen und Objektbezeichnungen einzuhalten. Das trägt erheblich zur besseren Verständlichkeit des Programmcodes bei und erfordert weniger begleitende Kommentare im Quelltext.

Variablennamen Sämtliche Objekte der GUI, bei denen die Möglichkeit besteht, dass im weiteren Programmlauf ändernd auf sie zugegriffen wird, werden dem statischen Array **gui_widgets** als Element hinzugefügt; dieses ist bezüglich des Namespace **RaPPPS** „global“ sichtbar. Die Bildung des Indexnamens geschieht dabei im folgenden Format:
<Abkürzung für Widget-Typ> <Abkürzung für Notebook-Seite> <Parameter>

Die Abkürzung für den Widget-Typ ist z.B. für ein sog. „labeled entry“ (also ein Eingabefeld mit Bezeichner): *labent*. Im nachstehenden Beispiel ist die Notebook - Seite die Seite für die Eingabe der Daten zur Widerstandsberechnung, daher der zweite Bestandteil *res*; und der Parameter *lwl* , für dessen Eingabe dieses Feld vorgesehen ist, ist die Wasserlinienlänge L_{WL}:

```
set gui_widgets(labent_res_lwl) [iwidgets::entryfield
       $subfr.labentlwl -width 8 ... -labeltext [lindex
       $textuals(labent_res_lwl) 0]]
```

Widgetnamen Der Pfadname des Widgets aus dem vorigen Beispiel, **$subfr.labentlwl** wird aus den gleichen Namenselementen zusammengesetzt, und der Index des Bezeichnertextes im Array **textuals** (s. Abschnitt 4.5) ist wiederum der gleiche wie im Array **gui_widgets**. Auf diese Weise wird eine weitgehende Konsistenz in den widgetbezogenen Bezeichnungen hergestellt, die sowohl beim Erzeugen neuer Bezeichnungen wie auch bei der Analyse und dem Debuggen des Quelltextes hilfreich ist und die Fehlerrate beim Programmieren deutlich senkt. Sieht man also z.B. folgende Zeile:

```
set gui_widgets(spinbox_prop_numofblades) [iwidgets::spinner
       $helpfr.spinnumofblades ...
```

ist sofort erkennbar, dass es sich um eine Spinbox auf der Seite zur Eingabe der Propulsionsdaten handelt, mit der die Anzahl der Propellerblätter eingestellt werden kann.

Alle Eingabedaten werden im Array **gui_inputs** gespeichert. Dieses enthält zwei Sub - Arrays, **res** und **prop**, in die die Daten für die Widerstands- bzw. Propulsionsberechnungen geschrieben werden. Bei der Wahl der Array - Indizes wurden keine weiteren Konventionen berückichtigt, sondern lediglich aussagekräftige Namen benutzt.

4.8 Datenfluss

Abbildung 4.6 zeigt den Datenfluss bei der Berechnung des Widerstandes und der Propulsion. Der Ablauf wird im folgenden kurz erläutert.

Abbildung 4.6: Schematische Darstellung des Datenflusses in RaPPPS .

Die Berechnungen von Widerstand und Propulsion erfolgen nicht in einem Durchgang, vielmehr werden zunächst die Widerstandsdaten berechnet; anschliessend können beliebig viele Propulsionsberechnungen (mit verschiedenen Parametern) vom Benutzer durchgeführt werden.

- Eingabe der Form - und Propulsionsparameter; die *optionalen* Parameter sind einige physikalische Werte (Temperatur, Oberflächenrauhigkeiten, etc.), die zu jedem beliebigen Zeitpunkt verändert werden können und von diesem Zeitpunkt an gelten. Bei der Berechnung der Propulsion wird auf die im Objekt gespeicherten physikalischen Daten zurückgegriffen, da eine Propulsionsberechnung bei Umgebungsbedingungen, die von denen der Widerstandsberechnung abweichen, wenig Sinn macht. Die Eingabedaten werden im Array **gui_inputs** gespeichert, das in die drei Sub - Arrays **res**, **opts** und **prop** unterteilt ist.

- Vom Array **gui_inputs** werden die Eingabedaten in das aktuelle Datenobjekt übertragen. Jedes Datenobjekt ist eine Instanz der Klasse **RaPPPS_Data** und ein Element im Array **data_objects** (s. a. Abschnitt 4.3.1). Durch die Trennung von Eingabevariablen und Speichervariablen im Datenobjekt ist eine unabhängige Verarbeitung der in der GUI *angezeigten* Daten und der den Berechnungen zugrunde liegenden Daten möglich (s. a. Abschnitt 4.3.4). Das ist z.B. beim Wechseln zwischen Varianten wichtig.

- Mit den Daten aus **gui_inputs** werden (`Calc_fixed_res_vals`) zunächst die geschwindigkeitsunabhängigen Werte/Koeffizienten berechnet (die also nur einmalig berechnet werden müssen).

- Dann werden mit den Daten aus **gui_inputs** die Widerstandswerte berechnet (`Calc_resistance_all`) und nach **calculated_outputs** geschrieben, wobei ständig auf bereits berechnete Werte in **calculated_outputs** zurückgegriffen wird.

- Schliesslich werden die berechneten Widerstandswerte aus **calculated_output** im Datenobjekt im Array **all_outputs** gespeichert.

Abbildung 4.7: Schematische Darstellung der programminternen Abläufe beim Erzeugen eines neuen Projektes.

- Zu Beginn der Propulsionsberechnung werden ebenfalls geschwindigkeitsunabhängige Werte berechnet (`Calc_fixed_prop_vals`). Dazu wird auf Daten aus der Eingabemaske (Array **gui_inputs**) sowie auf Daten, die bereits bei der Widerstandsberechnung im Datenobjekt gespeichert wurden, zurückgegriffen.

- Die Berechnung der Propulsionseigenschaften benutzt ebenfalls zusätzlich zu den geschwindigkeitsunabhängigen Werten Eingabedaten aus dem klasseneigenen Array **all_inputs** und greift ebenfalls auf bereits nach **calculated_output** geschriebene Daten zurück.

- Schliesslich werden die berechneten Propulsionswerte aus **calculated_output** im Datenobjekt im Array **all_outputs** gespeichert.

4.9 Ablauf einer Berechnung und Auswertung

Eine vollständige Berechnung des Widerstandes und der Propulsionseigenschaften verläuft immer in den gleichen Schritten: Dateneingabe für Widerstandsberechnung, Durchführung der Berechnung, Datenausgabe; Dateneingabe für Propulsionsberechnung, Datenausgabe. Im folgenden soll der programminterne Ablauf kurz in den wesentlichen Schritten erläutert werden. Entsprechende Prozedurnamen sind, wo sinnvoll, angegeben.

Anlegen eines Projektes: Datei: create_new_project.tcl

- Ggf. altes Projekt speichern (`Show_new_project_gui`)
- Eingaben auf Konsistenz prüfen (`Check_correctness_of_input`)
- Bestehende Objekte löschen (`Delete_old_objects`)
- GUI initialisieren (`Init_gui_inputs`)
- Geschwindigkeiten aus Eingaben berechnen (`Calc_intermediate_speeds`)
- Liste mit Geschwindigkeiten an **gui_inputs** übertragen (`Assign_initial_speed_defs`)
- Alle Diagramme und Tabelle zurücksetzen (`Reset_all`)
- Diverse Zähler initialisieren

Das Diagramm in Abbildung 4.7 (s. S. 51) veranschaulicht den Ablauf.

Erzeugen einer Variation: Datei: create_new_variation.tcl

- Zähler für Anzahl der Variationen um 1 erhöhen
- Aktuelle Position um 1 erhöhen
- Alle „höheren“ Objekte um einen Platz nach „oben“ verschieben
- Neues Datenobjekt erzeugen
- Neues Objekt mit Daten aus GUI initialisieren (`public method Transfer_gui_inputs`)
- Im neuen Objekt den Programmnamen und - version setzen (`public method Set_progname_and_version`)
- Erzeugungsdatum im Objekt setzen (`public method Init_variation_date`)
- Eingabedaten in Tabelle anzeigen (`public method Show_res_input_values`)
- Flag für *pending variation* auf *true* setzen
- Flag für ungespeicherte Daten auf *true* setzen
- Alle Diagramme aktualisieren; da in der *pending variation* noch keine Daten vorhanden sind, sind die Diagramme leer.

Nun kann die Eingabe der Formparameter des Schiffes durch den Benutzer erfolgen; anschliessend wird die Berechnung des Widerstandes gestartet.

Berechnen des Widerstandes: (Dateien: calc_gen_res.tcl, calc_res.tcl)

- Eingaben auf Konsistenz prüfen; falls fehlerhaft, nicht fortsetzen (`Validate_mand_res_input`, `Validate_res_relations`, `Validate_volunt_res_input`)
- Eingaben formatieren (`Format_res_input_vals`)
- Geschwindigkeitsunabhängige Werte/Koeffizienten berechnen (`Calc_fixed_res_vals`)
- Eingabedaten aus Widerstands - GUI in das neue Objekt übertragen (`Transfer_gui_inputs`)
- Physikalische Werte zum Zeitpunkt der Berechnung in das neue Objekt übertragen (`Transfer_gui_inputs`)
- Eigentliche Berechnungen durchführen; Daten werden im Array **calculated_output** gespeichert (Datei: calc_res.tcl; `Calc_resistance_all`)
- Flag für *pending variation* auf *false* setzen
- Flag für ungespeicherte Daten auf *true* setzen

- Benutzer und Datum mit Zeitpunkt der Berechnung im Objekt setzen
- Berechnete Daten formatieren (`Format_res_output_values`)
- Übertragen der im Array **calculated_output** gespeicherten Daten in das objekteigene Array **all_outputs** (`Transfer_calculated_output`)
- Diagramme und Tabelle aktualisieren mit den neu berechneten Daten (`Display_object`)

Damit ist die Berechnung des Widerstandes abgeschlossen. Es liegen nun Daten vor, auf deren Grundlage die Berechnung der Propulsionseigenschaften des Schiffes durchgeführt werden kann.

Berechnung der Propulsionseigenschaften (Dateien: calc_gen_prop.tcl, calc_prop.tcl)
Zur Berechnung der Propulsionseigenschaften muss keine neue Variation angelegt werden; die Ein - und Ausgabedaten werden in die aktuelle Instanz der Klasse RaPPPS_Data integriert. Das trägt dem Umstand Rechnung, dass der Benutzer unbegrenzt viele Varianten zur Propulsion berechnen können soll, aber nur auf der Grundlage des einen, gerade bearbeiteten Designs, für das erst eine Widerstandsberechnung stattgefunden haben muss.
Die Berechnung der Propulsion erfolgt fast analog zur Widerstandsberechnung:

- Prüfen, ob Variation *pending* ist; falls ja, nicht fortsetzen, da keine Widerstandsdaten vorhanden
- Eingaben auf Konsistenz prüfen; falls fehlerhaft, nicht fortsetzen (`Validate_variable_intervals, Validate_prop_inputs`)
- Propulsionsbezogene Diagramme zurücksetzen
- Eingaben formatieren (`Format_prop_input_vals`)
- Eingabedaten der Widerstandsberechnung laden (`Load_data_set`)
- Eingabedaten aus Propulsions - GUI in das Datenobjekt übertragen (`Transfer_gui_inputs`)
- Geschwindigkeitsunabhängige *Widerstands* - Werte/Koeffizienten berechnen (`Calc_fixed_res_vals`)
- Geschwindigkeitsunabhängige *Propulsions* - Werte/Koeffizienten berechnen (`Calc_fixed_prop_vals`)
- Evtl. schon existierende Propulsionsdaten aus Array **calculated_output** löschen
- Eigentliche Berechnungen (ggf. Optimierung) durchführen; Daten werden im Array **calculated_output** gespeichert (Datei: calc_prop.tcl; `Calc_propulsion_all`)
- Berechnete Daten formatieren (`Format_prop_output_values`)
- Übertragen der im Array **calculated_output** gespeicherten Daten in das objekteigene Array **all_outputs** (`Transfer_calculated_output`)

- Diagramme und Tabelle aktualisieren mit den neu berechneten Daten (`Display_object`)

Wegen der Linearität des Ablaufes der Variationserzeugung, Widerstands - und Propulsionsberechnung wurde auf begleitende Diagramme verzichtet; sie würden keinen weitergehenden Einblick verschaffen.

4.10 Kurzbeschreibung der Programmfenster von RaPPPS

RaPPPS stellt insgesamt 9 sog. „tabbed notebook“ - Seiten dar, in denen die verschiedenen Funktionalitäten gruppiert sind. Diese Seiten werden nachfolgend kurz beschrieben.

- **Resistance:** Anzeige der Projektdaten; Eingabe der für die Widerstandsberechnung relevanten Parameter, thematisch gruppiert.
- **RT Overview:** Balkendiagramm, das für alle Design-Varianten die Werte der Gesamtwiderstände RT zum schnellen Vergleich anzeigt.
- **Res. Components:** Anzeige von insgesamt 7 wichtigen Widerstandsanteilen in einem nicht veränderlichen Diagramm.
- **Propulsion:** Eingabe der für die Propulsionsberechnung noch fehlenden Parameter; Auswahl der Optimierung; Anzeige des Verlaufs der Optimierung mit den Werten der Variablen, der Nebenbedingung und der Gütefunktion (Leistung in MW).
- **KT, KQ, ETA0:** Freifahrtdiagramm des endgültigen Propellers mit den Kurven für Schub- und Drehmomentenbeiwert sowie Freifahrtwirkungsgrad.
- **Freestyle:** In diesem Diagramm können beliebig viele der berechneten Wertereihen über einer Wertereihe aufgetragen dargestellt werden.
- **Data:** Tabellarische Darstellung der Ein - und Ausgabewerte.
- **Logger:** Anzeige von Informations - und Fehlermeldungen sowie der berechneten Datenreihen; Werte anderer Variablen und Koeffizienten, die in der tabellarischen Auswertung nicht angezeigt werden.
- **Help:** Englisch - sprachige Online - Hilfe im HTML - Format.

Optionale Einstellungen Wie schon in Abschnitt 4.6 kurz erwähnt, können verschiedene Werte und Darstellungsoptionen vom Benutzer an persönliche Vorlieben angepasst werden. Dazu gehören die Werte einiger physikalischer Größen, die Darstellung der Diagramme und die Anordnung der Spalten in der Tabelle. Die Änderungen können in einem kleineren „notebook“ - Feld (über Menüpunkt <Optionals>) vorgenommen werden; es können auch die Standard - Einstellungen geladen werden, sowie die zuletzt gespeicherten, vom Benutzer definierten Einstellungen. Diese werden bei Beendigung des Programms im Hintergrund in der Datei rappps.cfg im Heimatverzeichnis des Benutzers gespeichert. Falls während des Programmlaufes Änderungen vorgenommen werden, die nicht zusagen, kann so der letzte gespeicherte Status wieder hergestellt werden.

Anhang A

Verwendete Symbole

A.1 Symbole nach ITTC

Für Tabellenköpfe im Text und in RaPPPS wurden häufig schiffbautechnische Symbole und Abkürzungen verwendet, die gelegentlich nicht dem deutschen Sprachgebrauch entsprechen. Alle verwendeten Symbole sind, soweit vorhanden, aus der von der ITTC veröffentlichten Liste „ITTC Symbols and Terminology List“ ([8]) entnommen. Im folgenden werden tabellarisch sowohl die in diesem Text als auch die in RaPPPS verwendeten Symbole der ITTC mit den Erläuterungen in englischer Sprache wiedergegeben.

Tabelle A.1: Die im Text und in RaPPPS verwendeten Symbole der ITTC.

ITTC Symbol	Computer Symbol	Name	Definition	SI-Unit
A_{BT}	ABT	Area of transverse crosssection of a bulbous bow (full area port and starboard)	The cross sectional area at the fore perpendicular.	m^2
A_D	AD	Developed blade area	Developed blade area of a screw propeller outside the boss or hub	m^2
A_E	AE	Expanded blade area	Expanded blade area of a screw propeller outside the boss or hub	m^2
A_O	AO	Disc area	$\pi D^2/4$	m^2
A_P	AP	Projected blade area	Projected blade area of a screw propeller outside the boss or hub	m^2
A_{TR}	ATR	Area of transom (full area port and starboard)	Cross-sectional area of transom stern below the load waterline	m^2

ITTC Symbol	Computer Symbol	Name	Definition	SI-Unit
A_W	AW	Area of water-plane	-	m^2
B	B	Beam or breadth, moulded, of ships hull	-	m
c	LCH	Chord length	-	m
CA	CA	Incremental resistance coefficient for model ship correlation	$R_A/(Sq)$	1
C_{AA}	CAA	Air or wind resistance coefficient	$R_{AA}/(A_V * q_R)$	1
C_B	CB	Block coefficient	$\nabla/(LBT)$	1
C_D	CD	Drag coefficient	$D/(Sq)$	1
C_F	CF	Frictional resistance coefficient of a body	$R_F/(Sq)$	1
C_{FO}	CFO	Frictional resistance coefficient of a corresponding plate	$R_{FO}/(Sq)$	1
C_M	CMS	Midship section coefficient (midway between forward and aft perpendicular)	$A_M/(BT)$	1
C_P	CPL	Longitudinal prismatic coefficient	$\nabla/(A_M L)$	1
C_P	CP	Local pressure coefficient	-	1
C_R	CR	Residuary resistance coefficient	$R_R/(Sq)$	1
C_T	CT	Total resistance coefficient	$R_T/(Sq)$	1
C_V	CV	Total viscous resistance coefficient	$R_V/(Sq)$	1
C_W	CW	Wavemaking resistance coefficient	$R_W/(Sq)$	1
C_{WP}	CWP	Water-plane area coefficient	$A_W/(LB)$	1
D	DP	Propeller diameter	-	m
d, T	T	Draft, moulded, of ship hull	-	m
η_B	ETAB, EFTP	Propeller efficiency behind ship	$P_T/P_D = TV_A/(Q\omega)$	1
η_D	ETAD, EFRP	Propulsive efficiency or quasi-propulsive coefficient	$P_E/P_D = P_R/P_P$	1
η_H	ETAH, EFRT	Hull efficiency	$P_E/P_T = P_R/P_T = (1-t)/(1-w)$	1

ITTC Symbol	Computer Symbol	Name	Definition	SI-Unit
η_O	ETAO	Propeller open water efficiency	-	1
η_R	ETAR, EFRO	Relative rotative efficiency	η_B/η_O	1
η_S	ETAS, EFPS	Shafting efficiency	$P_D/P_S = P_P/P_S$	1
h_O	HO	Immersion	The depth of submergence of the propeller measured vertically from the propeller center to the free surface	m
i_ϵ	ANEN	Angle of entrance, half	Angle of waterline at the bow with reference to centerplane, neglecting local shape at stem	rad
k	K	Three dimensional form factor on flat plate friction	$(C_V - C_{FO})/C_{FO}$	1
κ_S	KS	Roughness height of propeller blade surface	-	m
K_Q	KQ	Torque coefficient of propeller determined from thrust coefficient identity	-	1
K_T	KT	Thrust coefficient	$T/(\rho n^2 D^4)$	1
L	L	Length of ship	Reference length of ship (generally length between the perpendiculars)	m
L_{PP}	LPP	Length between the perpendiculars	-	m
L_R	LRU	Length of run	From section of maximum area or after end or parallel middle body to waterline termination or other designated point of the stern	m

ITTC Symbol	Computer Symbol	Name	Definition	SI-Unit
L_{WL}	LWL	Length of waterline	-	m
n	N	Frequency, commonly rate of revolution	-	Hz
P	PDR	Pitch ratio	P/D	1
P_D, P_P	PD, PP	Delivered power, propeller power	$Q\omega$	W
P_E, P_R	PE, PR	Effective power, resistance power	$R * V$	W
P_S	PS	Shaft power	Power measured on the shaft	W
P_T	PT	Thrust power	$T * V_A$	W
q	PD, EK	Dynamic pressure, density of kinetic flow energy	$\rho V^2/2$	Pa
Q	Q	Torque	P_D/ω	Nm
R_A	RA	Model-ship correlation allowance	Incremental resistance to be added to the smooth ship resistance to to complete the model-ship prediction	N
R_{AA}	RAA	Air or wind resistance	-	N
R_{AP}	RAP	Appendage resistance	-	N
R_{AR}	RAR	Roughness resistance	-	N
R_F	RF	Frictional resistance of a body	Due to fluid friction on the surface of the body	N
R_{FO}	RFO	Frictional resistance of a flat plate	-	N
R_T	RT	Total resistance	Total towed resistance	N
R_V	RV	Total viscous resistance	$R_F + R_{VP}$	N
R_W	RW	Wavemaking resistance	Due to formation of surface waves	N
S	S, AWS	Area of wetted surface	-	m^2
S_{AP}	SAP	Appendage wetted surface area, underway	-	m^2
T, d	T	Draft, moulded, of ship hull	-	m
T_A, d_A	TA, TAP	Draft at aft perpendicular	-	m
T_F, d_F	TF, TFP	Draft at forward perpendicular	-	m
t	THDF	Thrust deduction fraction	$1 - (R_T - F_P)/T$	1

ITTC Symbol	Computer Symbol	Name	Definition	SI-Unit
∇, V	DISPVOL	Displacement volume	$\Delta/(\rho g)$	m^3
V	V	Speed of the model or the ship	-	m/s
V_A	VA	Propeller advance speed	Equivalent propeller open water speed based on thrust or torque identity	m/s
V_{KN}	VKN	Speed in knots	-	-
w	WFT	Taylor wake fraction in general	$(V - V_A)/V_A$	1
X_{CB}, L_{CB}	XCB	Longitudinal center of buoyancy (LCB)	Longitudinal distance from reference point to the center of buoyancy, B	m
Z, z	NPB	Number of propeller blades	-	1
–	FW	Fresh water	-	-
–	SW	Salt water	-	-

A.2 Von Holtrop und Mennen verwendete Symbole

Teilweise wurden in den veröffentlichungen von Holtrop und Mennen eigene Symbole verwendet, die kein Äquivalent in der Liste der Symbole der ITTC haben. Sie sind nachfolgend mit englischem Originaltext aufgelistet.
Tabelle A.2: Die in den Veröffentlichungen von Holtrop und Mennen verwendeten Symbole.

Symbol	Definition	SI-Unit
R_B	Additional pressure resistance of bulbous bow near the water surface	$[N]$
R_{TR}	Additional pressure resistance due to transom immersion	$[N]$
h_B	Vertical position of the centre of A_{BT} above the keel plane	$[N]$
d	Thruster tunnel diameter	$[m]$
C_{BTO}	Relative position of thruster tunnel openings	1

Anhang B

Zum Programm RaPPPS

B.1 Kurzbeschreibung der Programm - Dateien

Für ein leichteres Verständnis der Programmstruktur werden im folgenden die Programm-Dateien von RaPPPS aufgelistet und mit einer kurzen Inhaltsangabe versehen. Mit ca. 9000 Zeilen reinem Programmcode ist eine solche überblicksartige Orientierungshilfe z.B. für den an der Programmierung interessierten Anwender sicher von Nutzen.

Tabelle B.1: Kurzbeschreibung der Quellcode - Dateien von RaPPPS

Dateiname	Beschreibung
default_config.rappps	Allg. Konfigurationsangaben, Formatstrings, Linienstile, Programmname, Pfade, Grundgrößen
dkt_coeffs.rappps	Koeffizienten und Exponenten für die Berechnung der ersten Ableitung des K_T - Polynoms; zur Ermittlung des Schnittpunktes von Schubbedarfskurve und Schubbeiwertkurve mittels Newton-Verfahren
kq_coeffs.rappps	Koeffizienten und Exponenten zur Berechnung des Drehmomentenbeiwertes nach [5].
kt_coeffs.rappps	Koeffizienten und Exponenten zur Berechnung des Schubbeiwertes nach [5].
strings.rappps	Alle im Programm verwendeten Text-Strings.
calc_fixed_coeffs.tcl	Berechnung von Geschwindigkeitsunabhängigen Werten.
calc_gen_prop.tcl	Vor- und Nachbereitung der Propulsionsberechnung.
calc_gen_res.tcl	Vor- und Nachbereitung der Widerstandsberechnung.
calc_optimum.tcl	Optimierung der Propellerparameter mit Simplex-Algorithmus.
calc_prop.tcl	Eigentliche Berechnung der Propulsionseigenschaften; alle geschwindigkeitsabhängigen Werte.

...Fortsetzung Programmdateien	
calc_res.tcl	Eigentliche Berechnung des Widerstandes; alle geschwindigkeitsabhängigen Werte.
check_prop_inputs.tcl	Konsistenzprüfung der vom Anwender eingegebenen Daten zur Propulsionsberechnung.
check_res_inputs.tcl	Konsistenzprüfung der vom Anwender eingegebenen Daten zur Widerstansberechnung.
class_data.tcl	Klassendefinition für das Datenobjekt von RaPPPS .
class_methods.tcl	Klassenmethoden für RaPPPS .
create_new_project.tcl	GUI-Definition zur Eingabe der neuen Projektdaten; Konsistenzprüfung; Prozeduren zur Initialisierung der Datenstruktur und der Ausgabebereiche.
create_new_variation.tcl	Anlegen eines neuen Datenobjektes.
err.tcl	Fehlerbehandlung und -Ausgabe.
history.tcl	Prozeduren zum Wechseln zwischen einzelnen Variationen und zum Löschen einzelner Variationen.
images.tcl	Erstellung der verwendeten Bilder/Icons.
init_data.tcl	Laden der Definitionsdateien für die Polynome, Textstrings, Tabellenköpfe, der allgemeinen und vom Anwender definierten Konfigurationsdaten; Initialisierung aller Eingabewerte.
intro.tcl	Anzeigen des RaPPPS - Startlogos.
load_project.tcl	Laden eines bereits vorhandenen Projektes.
mainframe.tcl	Definition des Hauptrahmens des Programms, der Schaltflächen, Menüs und des Statusbalkens.
notebook_barchart.tcl	Definition der Notebook Seite „RT Overview“. Prozeduren zur Behandlung des Ausgabebereichs.
notebook_freestyle.tcl	Definition der Notebook Seite „Freestyle“. Prozeduren zum Setzen/Ändern der Liste der anzuzeigenden Diagramme und zur Behandlung des Ausgabebereichs.
notebook_help.tcl	Definition der Notebook Seite für die Online-Hilfe. Prozeduren zum Laden der HTML-Dateien.
notebook_lines.tcl	Definition der Notebook Seite „Res. Components“. Prozeduren zur Behandlung des Ausgabebereichs.
notebook_log.tcl	Definition der Notebook Seite „Logger“. Prozeduren zum Einstellen des Debug-Levels.
notebook_openwater.tcl	Definition der Notebook Seite „KT, KQ, ETAO“. Prozeduren zur Berechnung des angezeigten Freifahrtdiagramms und zur Behandlung des Ausgabebereichs.
notebook_options.tcl	Definition der Notebook Seite für die optionalen Einstellungen. Prozeduren zum Setzen/Laden der Konfigurationseinstellungen.

...Fortsetzung Programmdateien	
notebook_prop.tcl	Definition der Notebook Seite „Propulsion“. Prozeduren zur Einstellung der Eingabedaten zur Propulsionsberechnung und Behandlung der Ausgabebereiche.
notebook_res.tcl	Definition der Notebook Seite „Resistance“. Prozeduren zur Einstellung der Eingabedaten zur Widerstandsberechnung.
notebook_table.tcl	Definition der Notebook Seite „Data“. Prozeduren zum Initialisieren der Tabelle und zum Anzeigen der Tabellenköpfe und Feldnamen.
pkgIndex.tcl	Für Programmstart.
rappps.tcl	Sourcen der Programmdateien; Laden der benötigten externen Tcl/Tk-Programmpakete; Start der einzelnen Initialisierungs- und Definitionsroutinen zum Programmstart.
save_procs.tcl	Prozeduren zum Speichern der Projektdaten und zum Exportieren einzelner Variationen und Diagramme.
units.tcl	Einheitenumrechner; z. Zt. noch nicht implementiert.

B.2 Inhalt der CD-ROM

Auf der beiliegenden CD-ROM befinden sich folgende Dateien und Verzeichnisse (relativ zum Wurzelverzeichnis der CD-ROM):

1. Die vorliegende Dokumentation als Postscript- und als PDF-Datei im Verzeichnis:
   /dokumentation/rappps.pdf,rappps.ps
2. Die Veröffentlichungen, in denen das Verfahren von Holtrop und Mennen dargestellt wird, als PDF-Dateien; sowie die Veröffentlichung von Oosterveld/van Oossanen in denen u.a. die KT/KQ-Polynome dargestellt sind:
   /dokumentation/*.pdf
3. Das Programm RaPPPS als Quellcode sowie als ausführbare Datei für Windows Betriebssysteme:
   Quellcode:
   /programm/rappps/src/*.tcl,*.rappps
   /programm/rappps/help/*.html
   /programm/rappps/pics/*.gif
   Executable:
   /programm/rappps-1.0.exe
4. Zwei Projektdateien für RaPPPS :
   /programm/examples/*.rappps
5. Alle Dateien der Web-Site von RaPPPS :
   /rappps-web/*.html
   Einstiegsadresse:
   /rappps-web/index.html
6. Eine Installationsanleitung (englisch):
   /programm/install.txt
7. Der Wortlaut der GNU General Public License, unter der RaPPPS veröffentlicht wird:
   /programm/license.txt

Literaturverzeichnis

[1] Holtrop, J.: *A statistical analysis of performance test results.* International Shipbuilding Progress, 24(270): 23-28, 1977.

[2] Holtrop, J.; Mennen, G.: *A statistical power prediction method.* International Shipbuilding Progress, 25(290):253-256, 1978.

[3] Holtrop, J.; Mennen, G.: *An approximate power prediction method.* International Shipbuilding Progress, 29(335):166-170, 1982.

[4] Holtrop, J.: *A statistical re-analysis of resistance and proulsion data.* International Shipbuilding Progress, 31(363):272-276, 1984.

[5] Oosterveld M.; van Oossanen, P.: *Further computer-analyzed data of the Wageningen B-screw series.* International Shipbuilding Progress, 22:251-261, 1975.

[6] Benini, E.: *Multiobjective Design Optimization of B-Screw Series Propellers Using Evolutionary Algorithms.* Marine Technology, Band 40, Nr. 4, 10/2003, S. 229-238.

[7] Gramlich, G.; Werner, W.: *Numerische Mathematik mit Matlab.* Heidelberg, dpunkt-Verlag, 2000.

[8] Johnson, B. (Ed.): *ITTC Symbols and Terminology List, Draft Version 1999.* International Towing Tank Conference, 1999.

[9] Comstock, J.P. (ed.): *Principles of Naval Architecture.* SNAME, 1967.

[10] Saunders, H.E.: *Hydrodynamics in Ship Design.* SNAME, 1957.

[11] Bernitsas, M.M.; Ray, D.; Kinley, P.: *K_T, K_Q and Efficiency Curves for the Wageningen B-Series Propellers.* The Department of Naval Architecture and Marine Engineering, Nr. 237, 1981.

[12] Bernitsas, M.M.; Ray, D.: *Optimal Revolution B-Series Propellers.* The Department of Naval Architecture and Marine Engineering, Nr. 244, 1982.

[13] Bernitsas, M.M.; Ray, D.: *Optimal Diameters B-Series Propellers.* The Department of Naval Architecture and Marine Engineering, Nr. 245, 1982.

[14] Birk, L.; Harries, S.: OPTIMISTIC - *Optimization in Marine Design.* Proceedings of 39^{th} WEGEMT Summer School. Mensch und Buch, 2003.

[15] Carlton, J.S.: *Marine Propellers and Propulsion*. Butterworth-Heinemann Ltd, 1994.

[16] Wrigth, M.H.: *Direct search methods: once scorned, now respectable*. In D. F. Griffiths and G. A. Watson (eds.), NUMERICAL ANALYSIS 1995 *(Proceedings of the 1995 Dundee Biennial Conference in Numerical Analysis)*, 191-208, Addison Wesley Longman, Harlow, UK, 1996.

[17] Welch, B.: *Praktisches Programmieren in Tcl und Tk*. Prentice Hall, 1996.

[18] Smith, C.: *[incr Tcl/Tk] from the Ground Up*. Osborne/MacGraw-Hill, 2000.

[19] http://hyperphysics.phy-astr.gsu.edu/hbase/kinetic/watvap.html

[20] Newman, J.N.:*Marine Hydrodynamics*. 9. Auflage, 1999, Maple-Vail Inc.