A Tool to Create Hydrodynamically Optimized Hull-Forms with Geometrical Constraints from Internal Arrangements

by

Konstantinos Nestoras

Bachelor of Science in Marine Engineering
Hellenic Naval Academy, 2005

Submitted to the Department of Mechanical Engineering and the System Design and Management Program in Partial Fulfillment of the Requirements for the Degrees of

Naval Engineer
and
Master of Science in Engineering and Management

at the
Massachusetts Institute of Technology
June 2013

Signature of Author [illegible]
Department of Mechanical Engineering
System Design and Management Program
May 10, 2013

Certified by [illegible]
Chryssostomos Cryssostomidis, Director of MIT SeaGrant
Doherty Professor of Ocean Science and Engineering
Department of Mechanical Engineering
Thesis Supervisor

Certified by [illegible]
Stefano Brizzolara, Assistant Director for Research at MIT Sea Grant
Research Scientist, Department of Mechanical Engineering
Thesis Advisor

Accepted by [illegible]
Patrick Hale, Director, System Design and Management Fellows Program
Engineering Systems Division
Thesis Advisor

Accepted by [illegible]
David E. Hardt
Chairman, Department Committee on Graduate Studies
Department of Mechanical Engineering

A Tool to Create Hydrodynamically Optimized Hull-Forms with Geometrical Constraints from Internal Arrangements

by

Konstantinos Nestoras

Submitted to the Department of Mechanical Engineering and the System Design and Management Program on May 10, 2013 in Partial Fulfillment of the Requirements for the Degrees of

Naval Engineer
and
Master of Science in Engineering and Management

ABSTRACT

Internal arrangements and bulky equipment like machinery have been treated for many years as a secondary aspect of the ship design. Traditionally, in the design process, the centerpiece of the effort is the hull and its hydrodynamic performance. Once the hull of a ship has been selected, all the other systems, like propulsion and electric plants, are selected and fitted in the ship. Due to the fact that the hull is considered as the most important system of the ship, any compromises and systems trade-offs that need to be done in the design process are focused mainly on all the systems apart from the hull-form. This inherent prioritization in the traditional design process, might lead to the selection of suboptimal solutions for the other systems like the propulsion and electric plants, which in turns might lead to a global suboptimal solution for the whole ship design. Unfortunately, these decisions bound the designed ship for lifetime and, down the road, might lead to excess operational costs.

The tool developed in this thesis treats the internal arrangements and the hull-form of the ship as two systems that need to be optimized together and not on a decoupled manner. Thus, the selection of the propulsion and electric plants or even large weapon systems like VLCs becomes as important as the hull during the design process. Propulsion and electric systems can be preselected in the early stage design, based on their efficiency and then a hull can be wrapped around them. The optimization of the hull can be done either with the use of the Holtrop method or a potential flow panel method, which provides higher fidelity. The designer has the ability to utilize this tool in order to easily conduct trade-off studies between the internal arrangements and the hull-form or save time from their integration and allocate it in other important problems of the design. This could aid the decision-making process in the early stage of the design, where information is scarce, decisions are crucial and uncertainty is high.

Thesis Supervisor: Chryssostomos Chryssostomidis
Title: Doherty Professor of Ocean Science and Engineering
Professor if Mechanical and Ocean Engineering

(This Page Intentionally Left Blank)

ACKNOWLEDGEMENTS

I would like to thank the Hellenic Navy and the Greek taxpayer, who in times of great economic difficulty gave me the opportunity to attend MIT and supported financially my studies. In addition, I would like to mention this work is supported by the U.S. Department of Defense, Office of Naval Research Award Numbers N00014-08-1-0080 and N00014-11-1-0598, ESRD Consortium, and MIT Sea Grant College Program under NOAA Grant Number NA10OAR4170086, MIT SG Project Number 2008-ESRDC-01-LEV.

I would like to express my deepest gratitude and sincere appreciation to my thesis supervisor Professor Chryssostomos Chryssostomidis for his mentoring and guidance throughout the duration of our collaboration. His passion on passing the knowledge and the art of ship design to new generations of naval engineers (and not only) motivated and inspired me greatly.

To Dr. Stefano Brizzolara, I would like to say a great thank you for his valuable advice, support and knowledge he shared with me. His excitement about ship design and his work were major contributors on finishing this thesis.

Next, I would like to thank the director of the Naval Construction and Engineering program, CAPT Mark Thomas, USN and the director of the Systems Design and Management program, Patrick Hale for their support during my studies.

Last but not least, I would like to express my deepest appreciation and love towards my wife Demetra for her unprecedented support and carrying. A very special thanks to our son Nicolas for his sincere love and priceless smiles, which were dissolving any troublesome thoughts I had during the duration of this work.

This thesis is dedicated to my father, who taught me to think like an engineer and ask *why, how* and *what* and not just get everything for granted...

(This Page Intentionally Left Blank)

TABLE OF CONTENTS

(This Page Intentionally Left Blank)

LIST OF FIGURES

(This Page Intentionally Left Blank)

LIST OF TABLES

NOMENCLATURE

AP	Aft Perpendicular
ASSET	Advanced Surface Ship Evaluation Tool
CAD	Computer Aided Design
C_f	Frictional resistance coefficient
CFD	Computational Fluid Dynamics
C_W	Wave making resistance coefficient
C_{WP}	Water-plane Area Coefficient
DWL	Design Waterline
FP	Forward Perpendicular
g	Gravitational acceleration
GA	Genetic Algorithm
GUI	Graphical User Interface
IGES	Initial Graphics Exchange Specification
IPS	Integrated Power System
ITTC	International Towing Tank Conference
L	Length; most of the times associated with LBP
LBP	Length Between Perpendiculars
LCB	Longitudinal Center of Buoyancy
NSGA-II	Non-dominated Sorting Genetic Algorithm 2
NURBS	Non Uniform Rational B-Spline
RANSE	Reynolds Averaged Navier Stoke Equations
Re	Reynolds number
R_f	Frictional Resistance of a ship
R_t	Total resistance; defined here as the sum of R_f and R_W
$\tilde{R}_t$	Non-dimensional form of the total resistance
R_W	Wave making resistance of a ship
S_{Wetted}	Wetted surface of a hull
U	Speed

USN	United States Navy
VLS	Vertical Launch System
Δ	Displacement of a ship
ν	Kinematic viscosity of a fluid
ρ	Density

CHAPTER 1 INTRODUCTION

1.1. Motivation

Modern warship designs of the United States Navy (USN) have led to an increasing demand of installed power. Even though the term power for many decades was mainly used for the propulsive power, as it was the major component of the installed power, in the recent years the installed electric power has grown rapidly. The reason for this is the use of larger, higher resolution and more powerful sensors and radars. For many years the naval engineers' community has observed the need to install more energy consuming sensors on warships, as they have a big impact on the war fighting performance of the ship.

This has led to a major stress in the design process, as more non-war-fighting equipment has to be installed inside the hull in order to provide the needed energy and power for the combat suite. Adding to this problem is the advances to weapons, which have moved from the era of gunpowder to the era of electromagnetism. Today's and tomorrow's weapons are relying even more in the electric generation capabilities of the ship-host. This has led to an unprecedented new demand of installed electric power onboard the ships.

The breadth of this problem can be realized if we consider that designers already have major issues and problems to accommodate the increasing size of machinery equipment, related with the increased power demand coming from the sensors. The increasing trend of these new design requirements has been studied for many years, as a lot of historical and statistical data have been gathered by the use of these Weapons, however, which are demanding for more electrical power to be installed on board the ships

Generally, in order to cover this increased demand of power on board the ships three different approaches can be followed; improve efficiency, improve power management or install more power. The first approach is to increase the efficiency of the main components, such as engines, generators and propulsors so that more power can be delivered by using equipment of the same size. Unfortunately, this approach seems to be utopic, as all these systems have almost reached their theoretical efficiency limits. Thus, even if improvements are made to current equipment to allow for greater effective power for a given size, this will not be enough to cover the increased power demand.

Another way to deal with the problem is to manage more efficiently the installed power on board the ship. This approach has been followed a lot in the last years and spawned some very interesting concepts. The concepts included shaft generators driven by the main propulsion engines of the ship and the opposite, where electric motors were used, in a hybrid configuration, to move the ship at low speeds. The idea of integrating the propulsive and electric power on board ships yielded some very positive results in the problem of high power demand. Further, the ultimate point of integration has been reached by the Integrated Power Systems (IPS), which manage the generation and distribution of power at a higher level, by allowing it to be consumed at any part or equipment of the ship; either in the propulsor or in the radars and sensors, or even in lighting.

Even this radical concept however was not able to cover the increased power demand of modern warships, leaving the only choice to be an increase of the installed power. Unfortunately, with this comes the problem of space, existing in the design of warships for many years now. Warships by definition, tradition and need are ships that have to encapsulate and contain a large number of equipment, while at the same time be fast and small for operational and economic reasons. This has led to the design of ships with high density of equipment, where it seems that everything is compacted with each other. Sadly, introducing more equipment in order to produce more power adds more into the problem of limited space.

For these reasons the arrangement of machinery rooms in modern warships has become a very important design aspect, troubling a lot naval architects. The importance of the machinery spaces and equipment layout becomes even more critical, as they are directly connected with the extremely important survivability requirements of such ships. The needed separation and the challenges imposed to the damaged stability by their massive volume makes the task of arranging them very demanding and important.

Traditionally the arrangement of these spaces and equipment was done after the selection of the hull. The designer had to squeeze all these equipment inside the ship, while at the same time make sure that she meets the demanding stability and survivability requirements. Following a more holistic approach, similar to the concept of IPS, benefits might be realized if the arrangement of such bulky and critical equipment is considered along with the definition and design of the hull-form.

Instead of giving priority to the shape of the hull, without considering nothing more than the resistance and, to a lesser extent, sea-keeping performance, the new approach proposes that priority should be given to the machinery spaces and the distributed systems related to the new IPS architecture. This would allow the naval architect to define and select this equipment a-priori and ensure that the design and shape of the hull will be able to fit and accommodate them. The tools used currently in the design process of ships do not allow or help the designer to treat the arrangements is such a way. For instance the Advanced Surface Ship Evaluation Tool (ASSET), used by the USN, allows the definition of the machinery spaces and engine layouts well after the hull has been selected. Further, there is no way that an interaction from the machinery equipment to the hull geometry can be realized.

For this reason, a tool was developed as part of this thesis, which gives the user the ability to conduct the design of the hull in the traditional way or to implement the new approach. The user is able to use this tool in order to design a hull, optimized upon resistance estimated with a potential flow panel method or the Holtrop method. This hull can then be used during the conceptual design of a ship and also can be further studied by more sophisticated fully viscous methods, in order to ensure and further improve the hydrodynamic performance. Of course, the designer would be still challenged to make sure that in the next steps of the design the needed equipment can fit inside the hull.

Alternatively, however, the tool gives to the designer the ability to handle the equipment at the beginning of the design. After the critical equipment is laid out, the tool is capable of performing the same optimization as before, but now, the internal arrangements serve as geometric constraints for the generated and optimized hulls. By employing this holistic approach, the designer has the benefit to relax the over-constraint problem of designing navy ships, as the tool removes one of the major design problems; to design an optimal hull able to encapsulate all the critical and bulky equipment.

1.2. Parametric Hull-Form Optimization

Parametric hull-form optimization is the process where the shape of the hull is controlled by a set of parameters and optimized upon a specific objective. A general and simple parametric optimization problem would be for example the design of a rectangle that has the maximum area with the minimal perimeter. In this simple example, the design parameters that control the shape of the rectangle are its two dimensions. By changing these variables, both the area of the rectangle and its perimeter are affected. For this simple problem, the solution can be extremely easy and straight forward that can be solved if a constraint is imposed for the area or the perimeter.

By applying the same approach in ships the problem becomes highly complicated and much more difficult to solve, especially if we set a goal for the resistance, which by itself is a very sophisticated hydrodynamic problem to solve. Never the less, parameterizing the shape of a ship's hull is indeed possible, especially due to the advances in computers and software. By using a parametric representation of the ship's surfaces, we have the ability to alter them easily and rapidly, which provides us with the benefit of effortlessly generating multiple hulls and variants. This capability can be used by sophisticated optimization algorithms in order to figure out what is the optimal combination of the design parameters in order to meet the goal of resistance, sea-keeping or whatever else it was decided to be.

The ability of modern High-Performance Computing (HPC) machines to distribute calculations on a large number of CPUs has not only allowed the widespread use of Computer Aided Design (CAD) software tools, but also shortened the time needed by numerical methods to solve the hydrodynamic problems in ship design. These tools were condemned for long to a limited use for analysis, due to the long calculations associated with them. Combining the use of parallel computing in the definition of ship's hulls and the calculation of their hydrodynamic performance has opened a new field of studies on the parametric optimization of hull-forms by CFD methods.

The benefits of using computers and conducting such kind of optimization have led to the realization of considerable improvements in the performance of hulls (Biliotti, et al., 2011). This becomes very important in the design of ships, as it provides the designer with more freedom on defining and selecting the hull to be used. In the past such freedom was forbidden due to the cost of building models and testing in the towing tanks. Now this is done virtually and with the press of a button with minimal cost to the designer.

Due to the important benefits of parametric ship design and the considerable improvements in the hydrodynamic performance that can be realized, the developed tool, in this thesis, uses the concept of the parametric hull-form optimization. The user is able to utilize a set of design variables in order to vary the hull geometry and at the same time assess the resistance of the ship for a specific speed by the use of a potential flow panel method or the Holtrop method. Going a step forward, it is possible for the user to setup optimization scenarios and problems in order to attain an optimal set of variables based on an objective function governed by the resistance.

1.3. Organization of the Thesis

This thesis develops a tool by using the Friendship Framework® software, which is a CAD program with multiple capabilities like easy connectivity with other tools and native optimization functions. The tool aims to be used as a decision tool at the early stage design of the ship, where it can aid the designer on defining the geometry of the hull. The designer can use this tool to generate optimized hulls based on the resistance at a given speed, while constraints about the longitudinal center of buoyancy (LCB) and displacement can be implemented. The tool goes one step farther by giving the user the ability to model internal arrangements and automatically translate them into geometric constraints that are taken into consideration during the optimization process.

The major advantage of this tool is the geometric definition of the hull that is described in chapter 2. Consequently chapter 3 describes the modeling of the internal arrangements and how they are used in the optimization process. Chapter 4 describes the potential flow panel method and the Holtrop method that can be used for the calculation of the hydrodynamic resistance of the generated hulls. As the hull and the internal arrangements are considered a coupled problem, chapter 5 discusses how the optimization of these two systems was treated. Finally, chapter 6 contains the conclusions about the use of this tool and recommendations for future.

Apart from the main body of the thesis, the appendices contain the code for the custom functions that had to be generated during this thesis. Also, the code for Friendship Framework® functions that had to be slightly altered, have been included in the appendix. Two script files that assign the values of the modeled DDG-51 to the variables and the measured values from hulls that are to be modeled have been added as well. Last but not least, a user manual on how to use the tool through the Graphical User Interface (GUI) or in batch mode; i.e. through a command line, is provided as well.

CHAPTER 2

HULL GEOMETRY MODELER

2.1. Goals of Geometry Modeler

The parametric model of the hull geometry was defined using the Friendship Framework® software. The goal was to be able to represent mono-hull combatant type ships, while at the same time give the user the ability to alter the geometry both in a global and a local level. Furthermore, the following aspects were taken into consideration:

- The modeler had to be able to give high control over the underwater surface, as this is the main driver of the resistance.
- The above the water surfaces had to be also modeled but no high detail or flexibility on the control of the geometry had to be sought.
- The generated surfaces should be as faired surfaces in the usual sense in naval architecture in order to improve the quality of the generated hulls.
- Parameters that are commonly used and meaningful for naval architects had to be utilized, in order to make the optimization process user-friendlier.
- The process for changing the geometry should be easy and flexible, allowing the user to easily manipulate the generated geometries.

2.2. Geometric Modeling for Ship Design

The geometric definition of ships has passed from the pure 2D drawings of the body-plan used for hundreds of years to a fully 3D definition based on mathematical representations. In 2D drawings the surface of the hull is represented by curves generated from the intersection of the hull surface with a set of planes. The used planes are of specific orientation and essentially normal to the longitudinal, transverse and vertical axis of the ship.

Modern 3D CAD software enabled the generation of ships geometries in a much easier way. At the same time, the need of specially trained designers that had the craftsmanship to draw high quality (faired) lines has been shifted onto the expertise of developing a well-behaved definition of mathematical geometric elements used by these programs.

Nonetheless and even if the design world has advanced in the 3D era, the traditional geometric definitions are still utilized. This is mainly as they offer a compact, well-understood and universally accepted way to define and communicate geometric definitions of hulls. Throughout the years different ways and systems to define the geometry emerged and can be categorized by level of definition and by the use of associative geometric modeling (Lechter, 2009). An explanation and a breakdown of these categories can be seen in the next sections.

2.2.1. Level of Definition

The geometric definition can be broken down into five levels: particulars, offsets, wireframe, surface models and solid models. Each of these models carries a different quantity and quality of information. It has to be clearly stated that none of these levels or methods is bad or outdated; however each one of these is more or less suited in specific cases and needs.

2.2.1.1 Particulars

This is the preferred method when general and gross characteristics of a ship need to be conveyed. Dimensions, volumetric and capacity measures are commonly used as particulars. No specific recipe or ideal combination of parameters exists and most of the times the particulars are depended on the class of the described ship. Hence, when describing a cargo ship, a reference about the tonnage or the capacity of the carried product is common.

The following parameters and measures are commonly used as "particulars":

- Length Overall (LOA)
- Draft (T); as the draft is a variable parameter depending on the loading condition of the vessel, most of the times the design draft is given, which is defined at a specific loading condition.
- Waterline Length (LWL), defined at the design draft and waterline.
- Design Waterline (DWL), the waterline of the vessel at the design draft.
- Length Between Perpendiculars (LBP or LPP), defined at the design draft and waterline.
- Beam (B); can be distinguished between the maximum width of the hull or the maximum beam of the DWL.
- Displacement (Δ), which is the entire mass of the ship. It can be given either at a design loading condition (military ships) or at the maximum loading condition (merchant ships).
- Tonnage, which is a measurement of the cargo capacity for merchant ships.
- Form Coefficients defined at a specific loading or design condition, such as the block, prismatic and water-plane coefficients.

Due to the fact that the particulars contain little detailed information, only a general description of the hull can be made. This makes the particulars difficult to be used for detailed parametric optimization of hulls. However, it is possible to use the particulars for hull optimization in a more general level, by using an empirical method, like Holtrop, to estimate the resistance. This can be possible; as such methods use a set of particulars level parameters to estimate the ship's resistance.

2.2.1.2 Offsets

Offsets are a tabulation of points residing on the surface of a ship and are defined based on a specific, and most of the times standardized, Cartesian coordinate system. For many years offsets were the primary means of sharing the geometry of the ship between Naval Architects and shipyards, as it was a pretty straightforward and simple way to describe the geometry of the hull.

The representation of the geometry is based on defining points on the intersection curves between the surface of the ship and specific planes. These planes could be normal to the vertical, transverse or longitudinal axis of the ship. Hence the intersections could be waterlines, sections or buttocks respectively (Gilmer & Johnson, 1982).

The accuracy of the representation depends on the number of points (offsets) used and can vary from few hundreds to few thousands. This method has been proven suitable to conduct hydrostatic calculations with an accuracy level on the order of 1 percent (Lechter, 2009). A major drawback of this method is the fact that by virtue of the definition it depends on 2D elements (curves) and hence is limited on the information that can contain. Finally, the method is not ideal to conduct hull optimization, as the tabulated points of the offsets are extremely difficult to manipulate in an effective way.

2.2.1.3 Wireframe

The wireframe representation of a hull is based on 2D and 3D curves. These curves can contain the section curves or other curves of interest like the transom and the stem curves. The traditional representation of the body plan can be considered as a wireframe representation of the ship's surfaces. There is no standard combination of curves used to represent the geometry, thus waterlines, buttocks or any other intersection curve of the ship's surface with any specified plane can be used. This representation can give the same information that offsets provide, accompanied with extra information about the stem and the bow sections, where the offsets fail to provide adequate information.

Even though, the wireframe representation is more detailed about the ship's geometry definition than offsets, it is limited on the fact that it can be considered a complete representation of the ship's surfaces. A major benefit of this method compared to offsets is that it can, more easily, be used for altering ship's geometry. Offsets on the other hand, as stated above, are more an output of the specific geometry and are not considered as a suitable method to define and alter the geometry. However, even the wireframe representation might not be ideal for hull optimization, as it does not contain enough information about the actual geometry of surfaces.

2.2.1.4 Surface Modeling

Surface modeling is a method to describe ship's geometry by directly defining her surfaces. This method has emerged in the past years, mainly due to the advancements in mathematics, where mathematical models and definitions were generated for surfaces well suited for use in ships. More specifically, the advent of B-Spline and NURBS (Non-Uniform Rational B-Splines) surfaces provided a rigorous tool to represent hull geometries with superior characteristics compared to the past. The inherent tendency of both B-Splines and NURBS to generate faired enough geometric elements (curves and surfaces) gave them the edge on the usage at the representation of hulls.

For many years architects pushed their limits on their effort to generate faired lines and surfaces to be used in ships. Interestingly enough, even though fairness is playing a huge role on the hydrodynamic performance of the ship, even today there is no rigorous and mathematical way to evaluate this property. For this reason, it can be said that fairness is considered an art and more of a qualitative aspect of a geometric element, as it is left in the designer's subjective opinion about visual aesthetic of the curve or surface. Due to this, the usage of geometric elements that tend to generate faired geometries is highly important in Naval Architecture and thus can explain the huge penetration observed in the design of ships and marine structures. Another reason for this penetration is the usage of these surfaces from CAD software, mainly due to their straightforward mathematical definition, which easily can be understood by computers.

Apart from the above benefits, modeling ships by using mathematical 3D surfaces is a more accurate, precise and flexible way to define their geometry. By having mathematically defined the surfaces, all the information needed for the geometry is contained in this definition and no possibilities of errors are left. This can't be said for offsets and wireframes, as by virtue of their definition blank spaces, between points or lines are left. To put it in a different perspective, if a surface definition exists, then easily offsets and wireframes can be generated, as they are by definition points or curves on a surface. On the other hand, if we have the offsets or the wireframe of a surface, or both, there is no guaranty that the surface from which they were initially defined can be replicated.

Surface modeling can be used to represent the geometry of a ship that needs to be altered and optimized, as it gives to the designer a complete definition of the ship's geometry. Moreover, it allows the designer to effectively and precisely change the geometry by using the mathematical definition of the surface. This aspect is of high importance when an optimization procedure is executed, as a straightforward and fairly easy way should exist in order to alter the geometry in the different variants.

2.2.1.5 Solid Modeling

Solid modeling of ships is a new way to represent their geometry. It can be considered a level upwards in terms of detail and complexity compared with surface modeling. The main difference and benefit of using this method is the attribute of topology, which can be considered as an extra layer of information. Inside this layer, complete information about which surfaces form the boundaries of a solid and how surfaces are adjoined or not are included. As with surface modeling, this method can also be a good candidate when building a model for hull optimization. However, the designer should take into consideration the extra topological information that is needed for the geometry definition and how this would affect the optimization process.

2.2.2. Associative Geometric Modeling

Associative modeling is the process where, in the definition of the model, relationships exist in the definition of the different geometric entities. This approach provides the benefit of fast alteration of the geometry, due to the fact that the different geometric entities are interrelated and interconnected. Thus, the alteration of some attributes or entities leads to the alteration of the whole design.

Four different methods of associative geometry are identified; Parametric, Variational and Feature-based modeling, as well as the Relational Geometry method (Lechter, 2009). The Parametric Modeling, also known as "Dimension-Driven", is the method where a set of dimensional parameters controls the definition of the geometry by means of defined formulas. The alteration of these parameters generates a parametric family of ships with common and general characteristics.

In the Variational Modeling, a set of geometric entities is altered by a set of dimensions, formulas and constraints that are imposed. Even though it seems to be very similar with the parametric method, the main difference is the usage of the constraints that can have the form of rules. Due to this, some designs may be discarded, as they do not meet the specific constraints in the early stage of the geometry definition. This is accomplished by the implementation of the constraints a-priori and not after the geometric definition is accomplished.

The Featured-based Modeling is a method where a set of features is used to define the geometry. This method is strong where similar structures or entities are used regularly, or a model mostly consists by the combination of different elements that are used multiple times. In the ship design, this kind of features can be frames; stiffeners or plates, which when combined can represent the ship.

Finally the Relational Geometry (RG) is an object based geometric definition, where different geometric entities are highly interrelated and interconnected. In this method, a set of points can be an input to the definition of a set of curves, which can in the end define a surface. Also, the opposite can happen, where a surface is an input for a point (point on surface) or a curve (edge or intersection curve). The different connections between the elements are stored and structured in a genealogical manner, where one element has its ancestors and descendants.

The need of this rich in information bookkeeping didn't allow this method to be used until recently, where the use of powerful computers, able to track all this relationships, is more common. In this method, a change somewhere in the model can create a small change in the neighboring area or a vast global change, depending on the established relationships between the different geometric elements.

2.3. Unique Geometric Entities of Friendship Framework®

Friendship Framework® has some unique geometry entities that can be used by the user in order to define the modeled geometry. These entities are the "F-Spline" and the "Metasurface", which are explained below. Apart from these two, a description about the "Feature Definition" and the "Curve Engine" is given, as they are both entities and functions used by the metasurface. For more information the reader can consult the user's manual of Friendship Framework® or the referenced material.

2.3.1. F-Spline

The F-Spline is a fairness optimized B-Spline that can respect specific constraints set forth by the user (Harries, 1998). These constraints can be the tangent angles at the two ends of the curve, as well as the area of the curve and the corresponding centroid with respect to a selected axis. This curve is strictly defined as planar and can't be in 3 dimensions. Never the less, this specific curve type is extremely powerful, as it gives to the designer the ability to easily control the tangent angles at the two ends, without the need to deal with the control points of the underlying B-Spline.

The capability of this curve to respect area constraints imposed by the user during the definition, gives an excellent way to control its fullness, something that can't be easily controlled at the B-Spline level. Further, the control that user has over the centroid of the area of the curve with respect to a given axis, allows for an extra lever of controlling the shape and the fullness of this curve. These two properties of the curve make it very powerful when the shape of hulls is to be modeled.

2.3.2. Metasurface

The "Metasurface" is a function of the software that is used to define surfaces by using predefined curves. This particular surface gives to the designer very high flexibility, as it allows him to define his own unique or special curve definition, which then will be used to define a surface. Metasurfaces are extremely valuable tools when the geometry of an object changes a bit as we move along a specific axis. This condition is almost always valid in ships and especially mono-hulls, if we think about than the cross section at x will be slightly different than the cross-section at $x+\Delta x$.

In other words, the metasurface is an automated way of lofting a surface through a set of curves, which the user can easily control. In order for the metasurface to work, two other entities of Friendship Framework® should be used, the "Feature Definition" and the "Curve Engine". Both of them are explained below and the reader can consult the help files and the software's user guide for more information. An illustration on how a metasurface is defined can be seen in Figure 1.

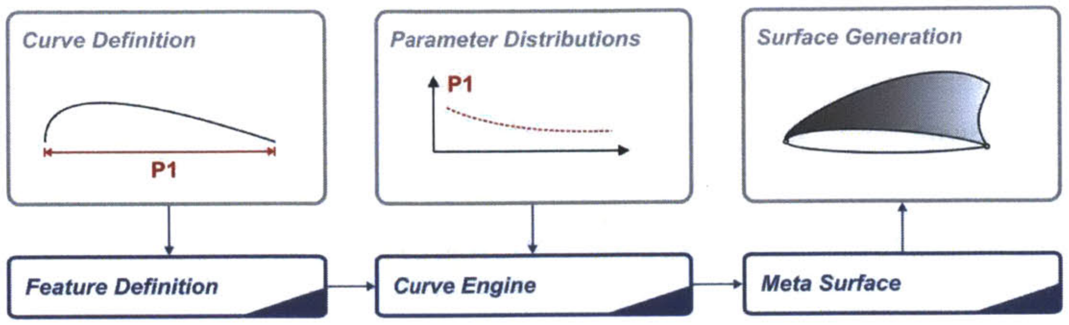


Figure 1 - Illustration of the Metasurface Definition (Friendship Framework®)

2.3.2.1 Feature Definition

The "Feature Definition" is a custom function that the user can define by using scripts and the programming language of Friendship Framework®. In here, the user defines the special curve that he wants to use, which can consist of multiple interconnected curves. The power of the "Feature Definition" is the ability of the user to describe in an abstract way a very complex parametric curve. The varying parameters can be defined in the form of curves and then used by the "Curve Engine" that builds the curves along a specific axis by using these varying parameters. This gives to the software its immense capabilities on parametric modeling, as it can be easily adapted to the user's needs and not limit him/her on pre-defined function, entities and features.

2.3.2.2 Curve Engine

As stated by the name, a "Curve Engine" is a functional element of Friendship Framework® that allows for rapid building of curves along an axis. To do that, the "Curve Engine" needs to consult an abstract definition of how the parametric curve is defined, which is provide in the form of a "Feature Definition". Apart from that, other curves that define the values of the parameters along the definition axis should be provided. These curves contain the needed information about the values of the parameters defining the shape of the generated curves along the definition axis. These generated curves are then used in the definition of a metasurface. Cross-sections of foils and ship's hulls can easily be represented and described by the use of a "Curve Engine", along with its corresponding "Feature Definition".

2.4. Definition of Geometry Modeler

As stated before the modeler should be able to represent the hull surface under the water with good detail for fast mono-hull displacement combatant type of ships without a sonar dome or a bulbous bow and at the same time allow for easy and rapid alteration of the geometry. After considering the different options on representing ship's geometries and their inherent pros and cons, as well as the limitations and advantages of the Friendship Framework® software, it was determined that the geometry modeler should be based on the Surface Modeling method by application of Relational Geometry.

The geometric definition was broken down in above and under the waterline parts. This was done, in order to better capture the different main parameters and factors that affect and control the shape of the two different surface groups. Moreover, one of the goals of this tool was to give priority on the underwater part of the hull, from where the hydrodynamic properties of the ship come from. Thus, by decoupling the definition of the above the water surfaces led to a simpler definition of them and consequently to a less complex, overall, geometric model.

2.4.1. Geometric Framework

The geometric framework adopted for the modeler is based on the definition of a set of curves, the main curves, based on which the hull surfaces are then generated. For the definition of the main curves, points are utilized, which are interrelated with the input parameters or with other curves or surfaces previously defined. The curves identified as "main" are summarized below and can be seen in Figure 2.

1. Profile Curve (buttock on the symmetry plane)
    a. Stem Curve
        i. Underwater part
        ii. Above the water part
    b. Keel
    c. Transom
    d. Edge of the weather deck
2. Design Waterline (DWL)
3. Transverse Cross Section
    a. Underwater part
    b. Above the water part

Based on the information obtained by the main curves, the transverse cross section is defined both in the underwater and above the water part. The cross section is then used to generate the corresponding surfaces.

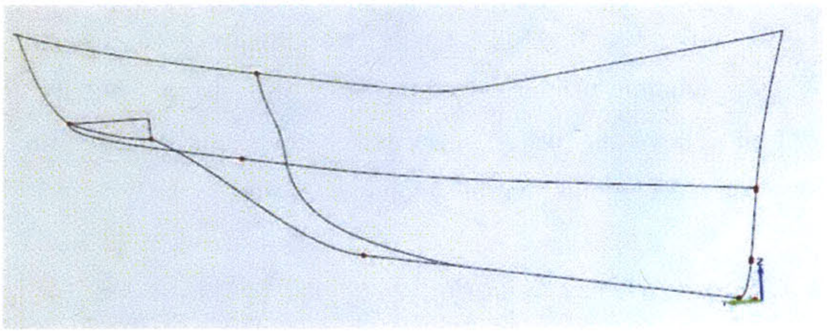


Figure 2 - Main Curves

The reference system used has its origin at the bow of the ship and the z-axis coincides with the Forward Perpendicular (FP). The stern of the ship has a positive x-abscissa, while the forward most point of the deck edge has a negative value, as it is positioned, forward of the FP. The keel of the ship lies on the x-axis and thus the z-abscissa is zero for most of it. The deck edge has always-positive z-abscissa values, as it is, by definition, above the keel. Finally, the starboard (STBD) or right side of the ship has positive y-abscissa, while the port or left negative.

2.4.2. Underwater Part (Quickwork)

The curves utilized for the definition of the underwater part of the hull surface (quickwork) are:

1. Profile Curve (buttock on the symmetry plane)
    a. Stem Curve (Underwater part)
    b. Keel
    c. Transom (Underwater part)
2. Design Waterline (DWL)
3. Transverse Cross Section (Underwater part)

2.4.2.1 Profile Curve – Underwater Part

The underwater part of the profile curve consists of the ship's keel, the stem curve and the transom. Each of these curves has a different geometric definition, due to the unique geometric characteristics that they have. Figure 3 shows the three different curves that define the underwater profile curve.

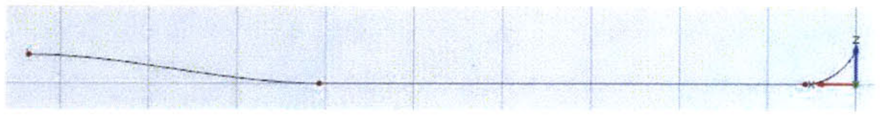


Figure 3 - Underwater Part of the Profile Curve

(1) Stem Curve

The stem part of the profile curve can be seen as the highlighted part (in green) of the underwater profile curve in Figure 4. This curve is and F-Spline and is defined by two points, forward (top) and aft (bottom), as well as two tangency constraints.

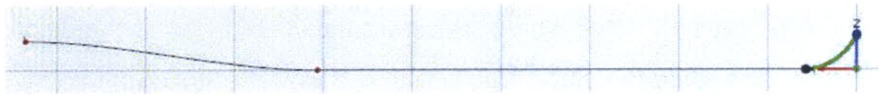


Figure 4 - Stem Curve of the Underwater Profile Curve

The forward point of the curve, which coincides with the forward point of the DWL, is defined as the intersection of the centerline with the FP. Thus, the coordinates of this point are zero for the x and y axes, while the z-coordinate equals with the draft of the ship.

$$Stem_{Fwd\ Point} = [0, \quad 0, \quad Draft]$$

The aft point of the stem lies on the baseline and centerline of the ship, thus the y and z coordinates will be zero. As this point is displaced aft wards of the FP its x-coordinate will be positive. This coordinate is determined by the use of the parameter "*D_StemRise*" located in the "*Parameters*" section of the object tree, inside the "Ship" section. This parameter is in turn defined as the product of the "*Draft*" and the "*StemRadiusOverDraft*" variables, located in the "*Variables*" section.

$$D\_StemRise = StemRadiusOverDraft * Draft$$

"*StemRadiusOverDraft*" is non-dimensional by the draft and describes what the radius of the stem will be. If the "*StemRadiusOverDraft*" equals one, then our stem curve will be a quarter of a circle with a radius equal to the draft of the ship. The coordinates of the aft point of the stem curve can be seen below.

$$Stem_{Aft\ Point} = [D\_StemRise, \quad 0, \quad 0]$$

Two tangency constraints need be imposed in the definition of the stem curve. As the stem curve, should meet the flat part of the keel, tangency continuity should be ensured. Thus, the tangent of the stem curve at the aft point is enforced to be zero with the x-axis. At the same time, at the forward point a tangency constraint is implemented, in order for the user to be able to control the angle of the stem. Thus, the tangent of the F-Spline at the forward point of the stem curve equals to the value set by the user at "*StemAngle*" variable in the "*Variables*" section of the object tree. A summary of the input parameters used in the F-Spline definition of the underwater stem curve can be seen in Table 1.

Parameter	Value
Fwd Position	[0, 0, *Draft*]
Fwd Tangent	*StemAngle*
Aft Position	[*D_StemRise*, 0, 0]
Aft Tangent	Parallel to x-axis

Table 1 - Summarized Values for the F-Spline of the Underwater Stem Curve

(2) Keel

The keel part can be seen as the highlighted part (in blue) of the underwater profile curve in Figure 5. The most forward point of the keel is connected with the underwater part of the stem curve, while the most aft point of the keel is connected with the transom.

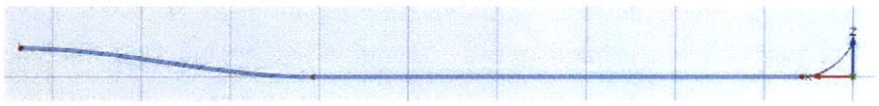


Figure 5 - Keel Curve of the Underwater Profile Curve

The keel itself is broken down into two curves. The forward part (curve 1) is a straight line, as the keel at this part of the ship is always a straight line lying on the baseline and centerline of the ship. The forward point is the aft point of the stem curve defined in the previous section, while the aft point is the point where the keel starts to rise. The "*KeelRisePoint*" variable, in the "*Variables*" section, defines the x-coordinate of the aft point. This variable is non-dimensional and is defined as a fraction of the "*LBP*" variable. The coordinates of the two points can be seen below.

$$Curve1_{Fwd\ Point} = Stem_{Aft\ Point} = [D\_StemRise, \quad 0, \quad 0]$$

$$Curve1_{Aft\ Point} = [KeelRisePoint * LBP, \quad 0, \quad 0]$$

The aft part (curve 2) of the keel curve is defined as an F-Spline. The forward point of this curve coincides with the aft point of the forward part (curve 1) of the keel. At this point, the tangent of the F-Spline is defined to be parallel with the x-axis, in order for tangency continuity to be respected. The aft point of curve 2 is defined to be at distance LBP from the origin and at an elevation equal to the draft of the ship reduced by the depth of the submerged part of the transom. This depth is controlled by the "*TransomDepth*" variable in the "*Variables*" section of the object tree.

$$Curve2_{Fwd\ Point} = Curve1_{Aft\ Point} = [KeelRisePoint * LBP, \quad 0, \quad 0]$$

$$Curve2_{Aft\ Point} = [LBP, \quad 0, \quad Draft - TransomDepth]$$

Here, it should be noted that the aft perpendicular (AP) is defined to be at the position where the keel stops and the transom starts. The user should be very careful, as most of the times the AP is defined at the aft most point of the DWL. In our case, this position is controlled by the angle of the transom. If we assume a regular transom, which is inclined aft-wards, then the aft most point of the DWL will be aft of the position where the keel stops and thus from where the AP is defined in this model.

The user has the ability to control the tangent at the aft point of the F-Spline by the variable "*KeelAngleAtTransom*". This allows the user to have more control on the shape of the aft part of the keel. A summary of the input parameters used in the F-Spline definition of curve 2 can be seen in Table 2.

Parameter	Value
Fwd Position	[*KeelRisePoint* * *LBP*, 0, 0]
Fwd Tangent	Parallel to x-axis
Aft Position	[*LBP*, 0, *Draft-TransomDepth*]
Aft Tangent	*KeelAngleAtTransom*

Table 2 - Summarized Values for the F-Spline of the Aft Part of Keel Curve

(3) Transom

The transom part of the underwater profile curve can be seen in Figure 6. This curve is defined as a straight line connecting the aft most point of the keel with the aft most point of the DWL. The angle of the transom can be controlled by the "*TrasnomAngle*" variable; positive values indicate that the transom is inclined aft-wards or in the opposite direction of the bow.

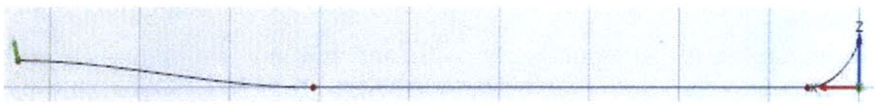


Figure 6 - Transom Curve of the Underwater Profile Curve

Only two points, the top and the bottom, define the line, with the bottom point coinciding with the aft point of the aft part of the keel (curve 2). The top point lies in the plane of the DWL and thus its z-coordinate equals the draft of the ship. The x-coordinate of this point equals the LBP, increased by a quantity coming from the inclination of the transom.

$$Transom_{Bottom\ Point} = [LBP, \quad 0, \quad Draft - TransomDepth]$$

$$Transom_{Top\ Point} = \begin{bmatrix} LBP + TransomDepth * \tan(TransomAngle), \\ 0, \\ Draft \end{bmatrix}$$

2.4.2.2 Design Waterline (DWL)

The Design Waterline is defined as a planar curve on a X-Y plane elevated (with respect to z-axis) by the draft of the ship. Two F-Spline curves are used in the definition of the DWL, with the separation into two segments done in the position of the maximum beam.

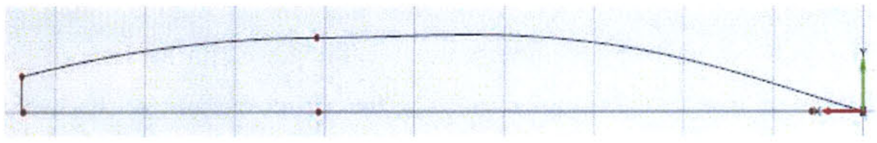


Figure 7 - Design Waterline

For the definition of these two curves, three points are needed. The forward most point is located on the centerline at elevation equal to the draft and its x-coordinate is zero as it lies on the FP. The mid-point is located at a longitudinal position controlled by the user through the "*x_Bmax*" variable, which is non-dimensional by the *LBP*. The y-coordinate of this point is the maximum half-beam of the ship and the user can control it by the "*L_Over_Beam*" variable.

$$DWL_{Fwd\ Point} = [0, \quad 0, \quad Draft]$$

$$HalfBeam = \frac{L\_Over\_Beam}{2} * LBP$$

$$DWL_{Mid\ Point} = [x\_Bmax * LBP, \quad HalfBeam, \quad Draft]$$

The aft most point is located at the same longitudinal position with the *Top Point* of the transom. The y-coordinate of this point is also the half-beam of the ship at the transom. The user can control this attribute of the ship by changing the "*H_BeamTransomOverH_Beam*" variable.

$$HalfBeamTransom = HalfBeam * H\_BeamTransomOverH\_Beam$$

$$DWL_{Aft\ Point} = \begin{bmatrix} LBP + TransomDepth * \tan(TransomAngle), \\ HalfBeamTransom, \\ Draft \end{bmatrix}$$

The aft F-Spline is a curve connecting the aft and the mid points of the DWL. The tangent of the curve at the mid-point was set to be zero, as this point was defined to be the point of maximum beam. At the aft point the user has the ability to control the tangent of the curve by the variable "*DWLaftAngle*". This gives to the user a control over the aft part of the DWL. A summary of the input parameters used in the F-Spline definition of the aft part of the DWL can be seen in Table 3.

Parameter	Value
Fwd Position	$DWL_{Mid\ Point}$
Fwd Tangent	Parallel to x-axis
Aft Position	$DWL_{Aft\ Point}$
Aft Tangent	*DWLaftAngle*

Table 3 - Summarized Values for the F-Spline of the Aft Part of DWL

The forward F-Spline of the DWL definition is a curve between the forward and the mid-point. The tangent at the forward point of the curve equals the entrance angle of the DWL and the user can control it by the "*EntranceAngle*" variable. At the mid-point the tangent was set again to be zero, or parallel to the x-axis, in order to force this point as being the one with the maximum half-breadth and also to ensure the tangent continuity between the forward and aft curves.

Apart from the tangent inputs, in the definition of the forward F-Spline, an area input has been used as well. This area input allows the user to control the water-plane area of the DWL, by means of the "*Cwp*" variable, which corresponds to the water-plane coefficient. The area of the aft F-Spline with respect to the x-axis is automatically calculated and taken into consideration in the area used as input for the forward curve.

By this, it is ensured that the total area of the DWL, with respect to the x-axis, is the one defined by the user through the C_{WP}. A summary of the input parameters used in the F-Spline definition of the forward part of the DWL can be seen in Table 4.

$$Area_{Total} = Cwp * HalfBeam * LBP$$

$$Area_{Fwd\ Part} = Area_{Total} - Area_{Aft\ Part}$$

Parameter	Value
Fwd Position	$DWL_{Fwd\ Point}$
Fwd Tangent	Entrance Angle
Aft Position	$DWL_{Mid\ Point}$
Aft Tangent	Parallel to x-axis
Area	$Area_{Fwd\ Part}$

Table 4 - Summarized Values for the F-Spline of the Forward Part of DWL

2.4.2.3 Transverse Cross Section – Underwater Part

The transverse cross section is the building block of the underwater surface of the hull. By definition this curve should extent from the keel of the ship to the DWL and the user should be able to control the deadrise (at keel) and flare (at the DWL) angles though out the length. In Friendship Framework®, this can be done with a curve engine that allows the user to generate multiple curves of the same definition by utilizing other curves and parameters as inputs. These, internally defined, curves are then used by a metasurface to generate a surface.

Because the geometry of the underwater surface of the hull changes drastically from the bow to the stern, it was decided that the best way to model this surface was by splitting it into two segments. For the most part of the ship, the cross section used for the definition of the surface was a NURBS curve. Closer to the bow, however, where the deadrise angle takes high values, an F-Spline curve was decided to be more appropriate. The user can change the point where the cross section definition changes by using the "*Cr_Section_Change*" variable, which is expressed as a fraction of the LBP.

Two "*Feature Definitions*", one for each curve (NURBS and F-Spline) are used for the definition of the cross section curves. These Friendship Framework® elements give to the users the ability to build their own unique functions or geometric elements, in a way similar to writing functions in Matlab®.

(1) NURBS

The cross section is defined as a 3rd order NURBS curve controlled by five points. The positions of the keel and the DWL are automatically identified for every longitudinal position that the cross section has to be defined. The points defining the curve were positioned so that the user can control the deadrise and flare angles for each transverse cross section, as well as the fullness. The NURBS cross section is defined in the "NURBS_Underwater" feature definition, which can be seen in appendix B.5.

Two points are positioned close to each other in the keel region, while two other points are located in the DWL region, so that the tangent of the NURBS curve in these two areas can be controlled. Further, the intersection of the two lines passing through the aforementioned set of points, defines the 5th point of the NURBS curve. The weight of this point can be varied through the length of the ship, giving the ability to the user to control the fullness of the section.

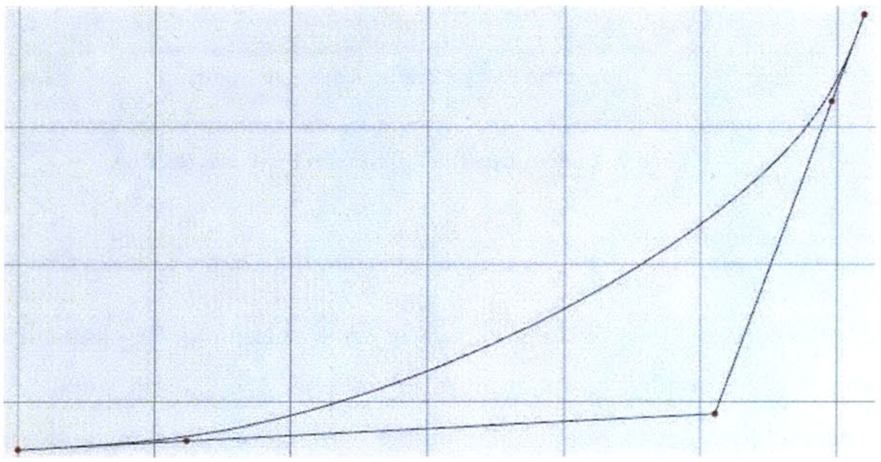


Figure 8 - NURBS Cross Section

The position of the control points and the shape of the control polygon affect a lot the end shape of the generated NURBS curve (Piegl & Tiller, 1997). After considering different arrangements of the points, it was decided that the transverse distance between the points that control the deadrise angle (2 lower points) would equal the one 5th (1/5) of the local beam at the longitudinal position where the cross section is defined. The vertical distance of the points that control the flare angle (2 upper points) was set to be one 5th (1/5) of the local vertical distance between the DWL and the keel. This arrangement allowed for smooth curves to be generated with small changes in the curvature, while the deadrise and flare angles were respected at the two ends (DWL and keel) of the cross section.

Apart from the control over these angles, the intermediate point in the definition of the NURBS curve can control the fullness of the section. This is done by allowing the user to vary the weight of the intermediate point throughout the length of the ship. The rest four points of the control polygon have a constant weight of one, at all longitudinal positions. Hence, when the weight of the intermediate point is increased above the value of one, it attracts the NURBS curve, resulting to a fuller cross section.

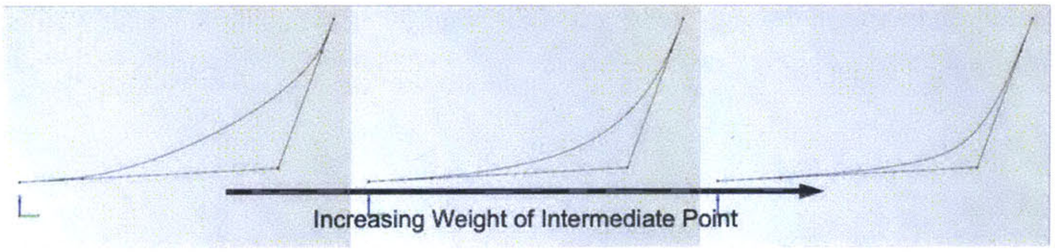


Figure 9 - Controlling the Fullness of the Cross Section

(2) F-Spline

For the forward part of the underwater cross section an F-Spline curve was used, as the NURBS definition has an inherent problem when the value of the deadrise angle is high. When this happens and depending on the value of the flare angle, the intermediate point can be located either below the baseline or above the DWL. This leads to curves that represent extremely poorly the geometry of the sections close to the bow. In Figure 10, we see an example (on the left) where the intermediate point is located below the baseline and in the opposite side (negative) of the y-axis. As stated before, the user has the ability to choose where the transition, from a NURBS curve to an F-Spline, takes place by changing the value of the "*Cr_Section_Change*" variable.

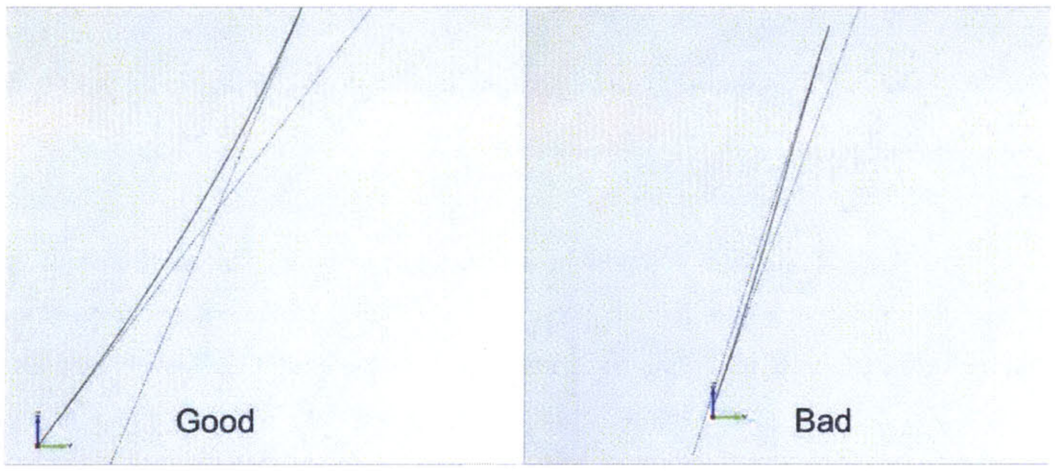


Figure 10 - Problem of the NURBS Cross Section Definition

The F-Spline curve is defined between two points, one on the keel and one on the DWL, for each longitudinal position. The tangents at the keel (deadrise angle) and at the DWL (flare angle) are also inputs in the definition of the curve. The position of the two defining points and the corresponding tangents at each of them are automatically identified and fed into the F-Spline definition at each longitudinal position. The feature definition describing the F-Spline cross-section definition can be seen in appendix B.4.

2.4.2.4 Surfaces

Four surfaces are used for the description of the total underwater part of the hull. Three of them are used for the definition of the side of the ship, while the fourth is the transom. For the most part of the side, two metasurfaces are used, utilizing the definitions of the cross sections described above. A very small lofted surface is used for the connection of the side with the transom, while the transom itself is a planar surface, defined by using the "*DeckSurf*" feature surface of Friendship Framework®.

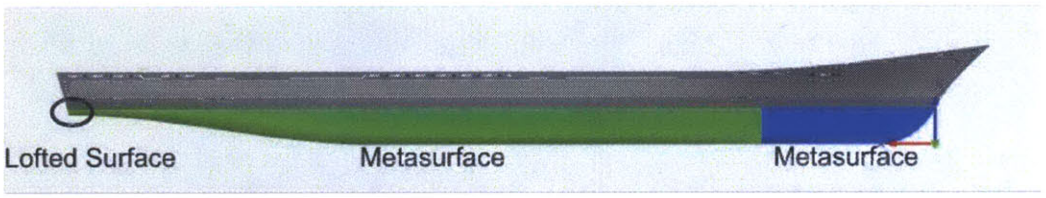


Figure 11 - Surfaces of Underwater Part of the Hull

(1) NURBS Metasurface

The largest surface component of the underwater side of the hull is the metasurface defined by the NURBS cross section. This surface extends from the point where the transition to an F-Spline section is done to the end of the keel, where we have defined our AP. The curve engine used for the generation of this surface is the one defined by the NURBS cross-section feature definition. Table 5 shows the values for the main input parameters in the definition of the NURBS metasurface. The base positions are defined with respect to the x-axis, as the cross section lies in the Y-Z plane.

Parameter	Value
Curve Engine at Beginning	Underwater_NURBS
Base Position at Beginning	*Cr_Section_Change*LBP*
Curve Engine at End	Underwater_NURBS
Base Position at End	*LBP*
Blending Method	Smooth Endings

Table 5 - Main Settings of NURBS Metasurface

(2) F-Spline Metasurface

The forward part of the underwater side of the ship is a metasurface defined by the F-Spline cross-section definition. This surface extends from the position where the cross section definition changes to the FP. In order to have a continuous and smooth side for the ship, both the NURBS and F-Spline curve engines are used in the definition of this metasurface. In the forward part, we have the F-Spline cross-section, while in the aft part we have the NURBS cross-section. An automatic blending mechanism ensures the smooth transition from the F-Spline to the NURBS geometry, thus the generated metasurface connects smoothly with the NURBS part of the side.

Parameter	Value
Curve Engine at Beginning	Underwater_FSpline
Base Position at Beginning	FP (x=0)
Curve Engine at End	Underwater_NURBS
Base Position at End	*Cr_Section_Change*LBP*
Blending Method	Smooth Endings

Table 6 - Main Settings of F-Spline Metasurface

(3) Lofted Surface

As indicated above, a lofted surface is used between the aft edge of the NURBS metasurface and the connection to the transom. This surface is needed due to the aft inclination of the transom. As the NURBS part of the side surface stops at the keel, a gap is generated between this surface and the surface of the transom. The lofted surface is built between the aft edge of the NURBS metasurface and the side edge of the underwater part of the transom.

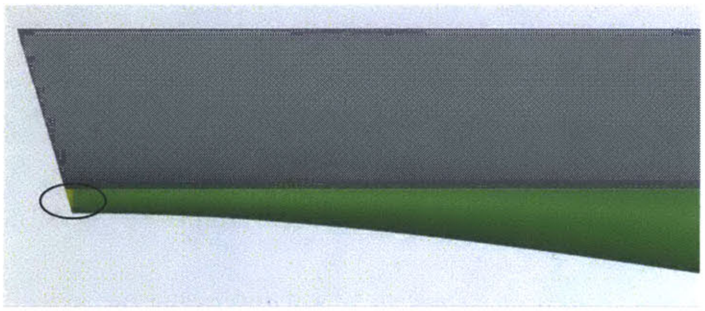


Figure 12 - Lofted Surface of Underwater Side

The edge of the underwater transom is defined using the aft edge of the NURBS metasurface. This curve is projected on the transom plane, which is defined as the plane passing through the most aft (end) point of the keel and is normal to a vector lying on the

X-Z plane and having an angle complementary to the transom angle[1]. The projection is done along the tangent vector of the DWL at the most aft point, which equals with the variable "*DWLaftAngle*".

(4) Transom Surface

The underwater transom surface is defined by using a modified version of the build-in "*DeckSurf*" Friendship Framework's feature surface and can be seen in appendix B.2. The most aft edge of the lofted surface is used as an input, in order to ensure that there are no openings in the underwater volume of the ship. Finally, the transom surface is automatically extended until it meets the centerline (X-Z) plane.

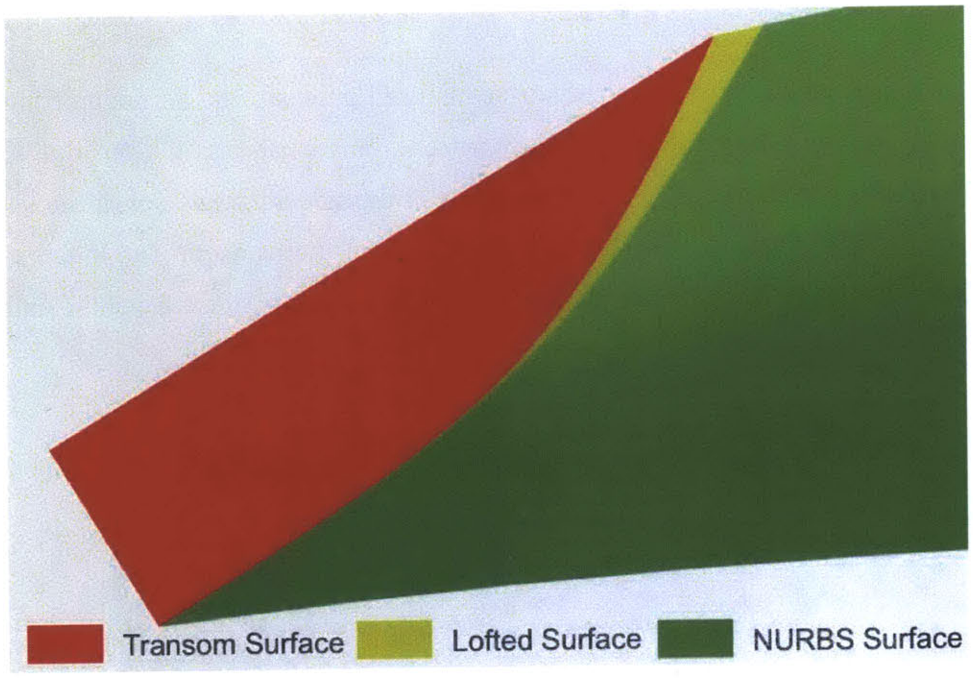


Figure 13 – Transom Area of Underwater Surfaces

[1] In the model, the transom plane is defined in a different way, as there is no capability on projecting curves on planes and thus surfaces are used.

2.4.3. Above the Water Part (Deadworks)

For the definition of the above the water part of the hull (deadworks), the following curves are used:

1. Profile Curve
    a. Stem Curve (Above the water part)
    b. Edge of Weather Deck
    c. Transom (Above the water part)
2. Transverse Cross Section (Above the water part)
3. Design Waterline (DWL)

The definition of the DWL was discussed previously in section 2.4.2.2, as it was also used in the definition of the underwater surface. As discussed in section 2.1, the above the water part of the hull was not of high importance, as it does not contribute in the calculated resistance by the panel method potential flow solver. For this reason, a simpler representation of the above the water part was selected, in order to simplify the geometric definition of the hull.

2.4.3.1 Profile Curve – Above the Water Part

The above the water part of the profile curve consists of the weather deck's edge, the above the water part of the transom and the stem. Figure 14 shows the three curves in different color.

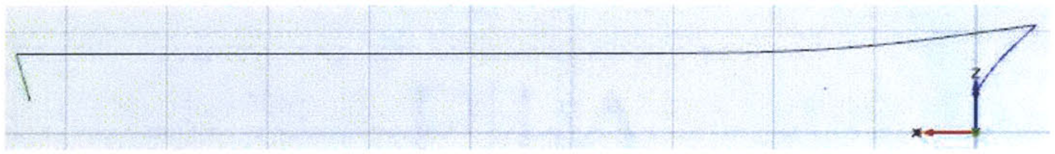


Figure 14 – Above the Water Part of the Profile Curve

(1) Transom

The above the water part of the transom is a simple straight line that extends from the point where the underwater part stops, to the aft most point of the ship. The geometry model assumes that the ship has the same freeboard at the stern as at the amidships section. For this reason, the aft most point will be elevated from the baseline by the draft of the ship, increased by any freeboard the user defines in the "*Freeboard_MS*" variable. Further, because of the angle of the transom, the x coordinate of the point should also be corrected accordingly.

$$Transom_{Bottom\ Point} = \begin{bmatrix} LBP + TransomDepth * \tan(TransomAngle), \\ 0, \\ Draft \end{bmatrix}$$

$$Transom_{Top\ Point} = \begin{bmatrix} LBP + (TransomDepth + Freeboard\_MS) * \tan(TransomAngle), \\ 0, \\ Draft + Freeboard\_MS \end{bmatrix}$$

(2) Edge of Weather Deck

The edge of the weather is the curve that dominates in the definition of the profile curve above the water. This curve is defined based on the centerline of the weather deck and the DWL. The DWL is used as a parent curve and a manual transformation is devised in order to give the final shape to the edge of the weather deck. The weather deck is assumed to have zero camber, thus the elevation at the edge of the deck is the same with the elevation at the centerline.

The centerline of the weather deck is defined by three different curves, as it can be seen in Figure 15. The largest and most aft of the three parts is a straight line extending from the top point of the transom to length 0.25 of LBP. Here it is assumed that the freeboard at the FP is larger than that at amidships and also that the elevation of the weather deck is done in the forward quarter length of the ship.

$$Point4 = Transom_{Top\ Point}$$

$$Point3 = \begin{bmatrix} 0.25 * LBP \\ 0, \\ Draft + Freeboard\_MS \end{bmatrix}$$

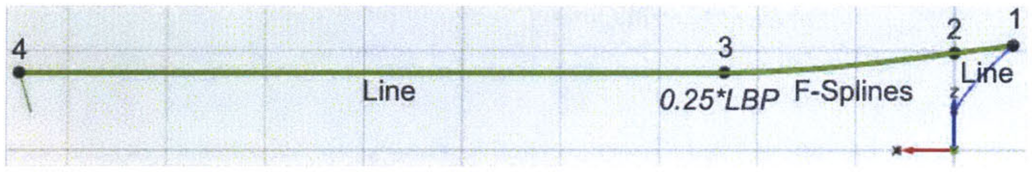


Figure 15 - Centerline of Weather Deck

The second curve used for the definition of the weather deck is an F-Spline running between a point on the centerline at 0.25 LBP and a point defined at the FP. The user can control the elevation of this point, which corresponds to the freeboard of the ship at the FP, by using the "Freeboard_FP" variable. Finally, the tangent is set to be parallel to the x-axis at the aft most point of the F-Spline, in order for this curve to connect smoothly with the straight line defined earlier.

$$Point2 = \begin{bmatrix} 0 \\ 0, \\ Draft + Freeboard\_FP \end{bmatrix}$$

Parameter	Value
Fwd Position	*Point2*
Fwd Tangent	Free – No tangent constraint
Aft Position	*Point3*
Aft Tangent	Parallel to x-axis

Table 7 - Summarized Values for the F-Spline of the Weather Deck's Centerline

The last part of the weather deck centerline is a straight line, stretching between the forward point of the F-Spline and the forward most point of the hull. The forward most point of the ships is simply defined forward of the FP, at a distance equal to the freeboard at the FP. Its elevation above the baseline is calculated by taking into consideration the tangent of the F-Spline at its forward point.

$$\alpha = tangent\ of\ FSpline\ at\ forward\ location$$

$$Point1 = ForwardMostPoint = \begin{bmatrix} -Freeboard\_FP \\ 0, \\ Draft + Freeboar\_FP \ * [1 + \tan(\alpha)] \end{bmatrix}$$

After having defined the centerline curve of the weather deck, the elevation component of the deck edge has been expressed. In order to complete the definition of the edge, we need to describe the y-component as well. The DWL is used as a parent curve and an offset transformation in the y-direction is used. This transformation is based on three offset angles that the user controls along the length of the ship and the vertical distance between the DWL and the weather deck's centerline, which corresponds to the ship's freeboard.

In the "*Variables*" section of the object tree, under "*Auxiliary*" and "*DeckEdge*", the user can find the variables "*OffsetAngleFWD*", "*OffsetAngleMidShips*" and "*OffsetAngleTransom*". These variables are used to control the y-coordinate of the points defining the edge curve of the weather deck. Eight points and three curves are used to define the deck edge curve as it can be seen in Figure 16. A major assumption made in the definition of this curve is that the width of the weather deck from 40% to 70% of LBP will be constant.

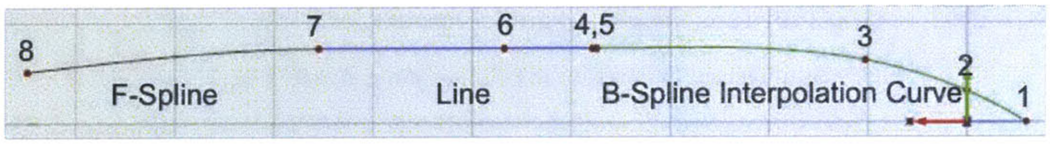


Figure 16 – Edge of Weather Deck Curve

Starting from forward, the first curve used is a B-Spline interpolation curve through points 1 to 6. Points 4 and 5 are positioned very close to each other, in order to ensure that the tangent of the B-Spline would be parallel to the x-axis. Between points 4 and 7, a straight line is drawn and these points have the same y-coordinate as they belong to the part of the weather deck with the fixed width. In the aft part, an F-Spline is used to connect points 7 and 8 with a constraint of having a tangent parallel to the x-axis at point 7, so that a smooth connection is achieved with the straight line defined earlier.

Parameter	Value
Fwd Position	*Point7*
Fwd Tangent	Parallel to x-axis
Aft Position	*Point8*
Aft Tangent	Free – No tangent constraint

Table 8 - Summarized Values for the F-Spline of the Weather Deck's Edge

Table 9 has a summary of the x and y coordinates of all points used in the definition of the edge curve of the weather deck. Point 6 is located amidships and is not used directly in the curve definition, but indirectly for visualization and calculation purposes. Point 1 coincides with the forward most point of the ship, defined earlier for the centerline curve of the weather deck. From the same curve, the z-coordinate of all points is taken, based on their longitudinal position. When the DWL's y-coordinate is needed, it is taken automatically based on the point's longitudinal position.

Point 8 is located at the same longitudinal position with the aft most point of the deck's centerline, resulting to a straight and parallel to the y-axis top edge of the transom surface. The definition of the weather deck does not allow the existence of a stepped hull, which was considered out of the scope for this tool and can be added later on in the design process.

Point	X-coordinate	Y-coordinate
1	*ForwardMostPoint*	
2	0	$Y_{p3}/2$
3	0.11**LBP*	Y_{DWL} + *(Z*$_{DeckCL}$ *- Draft)*tan(OffsetAngleFWD)*
4	0.40**LBP*	$Y_{p6}/2$
5	0.405**LBP*	$Y_{p6}/2$
6	0.50**LBP*	Y_{DWL} + *Freeboard_MS*tan(OffsetAngleMidShips)*
7	0.70**LBP*	$Y_{p6}/2$
8	$X_{AftMostPoint}$	Y_{DWL} + *Freeboard_MS*tan(OffsetAngleTransom)*

Table 9 - X and Y Coordinates of Deck Edge's Defining Points

(3) Stem Curve

The stem curve, defined as an F-Spline, can be seen with blue color in Figure 14. Two points and one tangent constraint are used in the definition of this curve. The lower point of the curve is the forward most point of the DWL, or the upper point of the lower part of the stem curve as defined in section 2.4.2.1(1). At this point the tangent of the F-Spline is set to be equal to the stem angle, controlled by the *"StemAngle"* variable, so that continuity exists between the above and under the water parts of the stem.

$$DWL_{Fwd\ Point} = UnderwaterStem_{Fwd\ Point} = [0, \quad 0, \quad Draft]$$

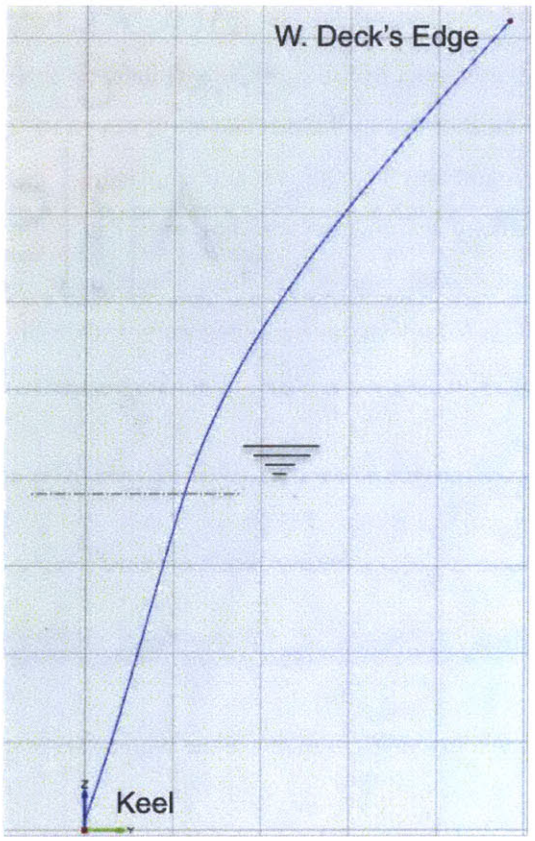
For the upper point of the stem curve, the forward most point of the weather deck's centerline is used, as defined in the last section. There is no tangent constraint enforced at this end of the curve and thus the F-Spline by definition takes the shape with the minimum change in curvature. The bow shape of the ship is highly correlated with the stem curve and is very important for the phenomenon of slamming and the loads associated to it. As the scope of this tool is to examine the hydrodynamic resistance of the hull in calm water, it was concluded that the used definition of the stem is accurate enough. A better design of the bow section and especially the part above the waterline can be done separately by the designer by using a different tool or commonly used rules and technics.

Parameter	Value
Fwd Position	*ForwardMostPoint*
Fwd Tangent	Free – No tangent constraint
Aft Position	$DWL_{Fwd\ Point}$
Aft Tangent	*StemAngle*

Table 10 - Summarized Values for the F-Spline of the Above the Water Stem Curve

2.4.3.2 Transverse Cross Section – Underwater Part

For the part of the hull that is above the water the cross section is defined by using an F-Spline. This method allows for as simple and easy definition for this part of the cross section, while at the same time the generated geometries are very close to reality. The feature definition used by the curve engine is the "*FSpline_AboveWater*" and can be seen in appendix B.3.



Two points generate the cross section, one on the DWL and the other on the edge of the weather deck. The curve engine gets these positions automatically for every longitudinal position that a curve has to be defined. Further, the tangent of the point on the DWL is set equal to the flare angle of the underwater section for every longitudinal position. This is done in order to enforce the tangency continuity between the two parts of the cross section.

Parameter	Value
Top Position	On Edge of Weather Deck
Top Tangent	Free – No tangent constraint
Lower Position	On DWL
Lower Tangent	*FlareAngle*

Table 11 - Summarized Values for the F-Spline of the Above the Water Cross Section

2.4.3.3 Surfaces

The above the water part of the hull is described by the use of four surfaces. Three of them represent the side of the hull from the stem curve to the transom of the hull and the fourth is the transom surface above the waterline. The most part of the hull's deadworks is described by a metasurface that uses the definition of the cross section described before. A coons patch is used for the connection of the hull's side with the stem curve, while in the aft; a lofted surface is used to connect the side of the hull with the transom surface. For the transom the *"DeckSurf"* feature surface of Friendship Framework ® is used.

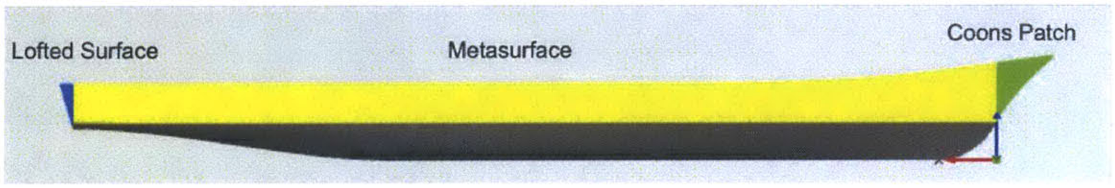


Figure 17 - Surfaces of Above the Water Part of the Hull

(1) F-Spline Metasurface

The most part of the side of the ship above the water is a metasurface defined by the F-Spline cross-section definition. This surface extends from the FP to the longitudinal position where the keel meets the transom (x=LBP). The curve engine used for the generation of this surface is the one defined by the above the water cross-section feature definition. Table 12 shows the values for the main input parameters in the definition of the metasurface. The base positions are defined with respect to the x-axis, as the cross section lies in the Y-Z plane.

Parameter	Value
Curve Engine at Beginning	AboveWater_FSpline
Base Position at Beginning	FP (x=0)
Curve Engine at End	AboveWater_FSpline
Base Position at End	*LBP*
Blending Method	Smooth Endings

Table 12 - Main Settings of F-Spline Metasurface

(2) Coons Patch

A coons patch is used for the closure of the above the water side-surface of the hull with the stem curve. The need for this surface comes from the specific geometry of the hull at that particular position, which does not allow the metasurface to efficiently describe the geometry of the hull starting from the stem curve. The coons patch, in Friendship Framework ®, is a surface that is described by four edge curves.

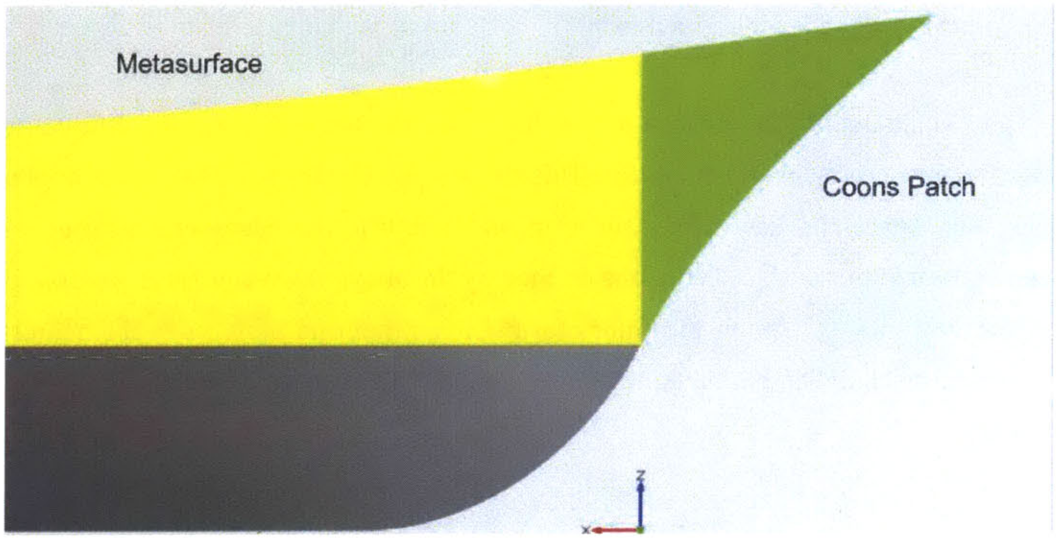


Figure 18 - Coons Patch of Above the Water Side

In this part of the hull, however, only three curves are available for the definition of the surface; the stem curve, the forward edge of the F-Spline metasurface and a part of the weather deck's edge. For this reason a pseudo-curve of zero length was created, as an image curve of the DWL, and used in the definition of the coons patch.

Curves	Value
C-1	Part of Deck Edge forward of FP
C-2	"Zero Line"
D-1	Forward Edge of F-Spline Metasurface
D-2	Stem Curve Above DWL

Table 13 - Coons Patch Defining Curves

(3) Lofted Surface

As indicated above, a lofted surface is used between the aft edge of the F-Spline metasurface and the transom. This surface is needed because the transom is inclined towards the aft and the metasurface stops at the point where the keel meets the transom (x=LBP). The lofted surface is built between the aft edge of the metasurface and the side edge (or section curve) of the underwater part of the transom.

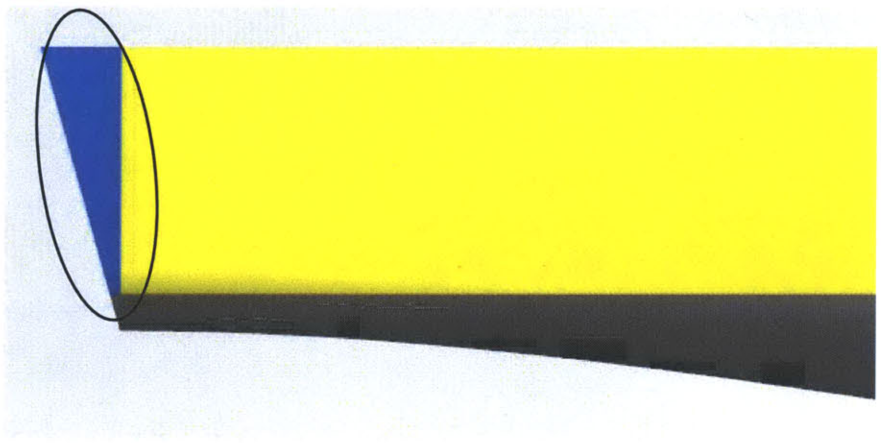


Figure 19 - Lofted Surface of Above the Water Side

The edge of the underwater transom is defined using the aft edge of the metasurface, which is projected on the transom plane. This plane is defined as the one passing through the most aft (end) point of the keel and is normal to a vector lying on the X-Z plane and having an angle complementary to the transom angle[2]. The projection is done along the tangent vector of the DWL at the most aft point, which equals with the variable "*DWLaftAngle*". This ensures that the above the DWL part of the side will connect properly with the underwater part.

Unfortunately, however, by projecting the aft edge of the metasurface on the transom plane, with an angle parallel to the angle of the DWL at the aft most point does not ensure that the top point of the transom edge will coincide with the aft most point of our weather deck's edge, defined at 2.4.3.1(2). For this reason, a correction should be done in the initial projected curve, which should be displaced in such way that the connection point with the underwater part of the transom remains the same.

[2] In the model, the transom plane is defined in a different way, as there is no capability on projecting curves on planes and thus surfaces are used.

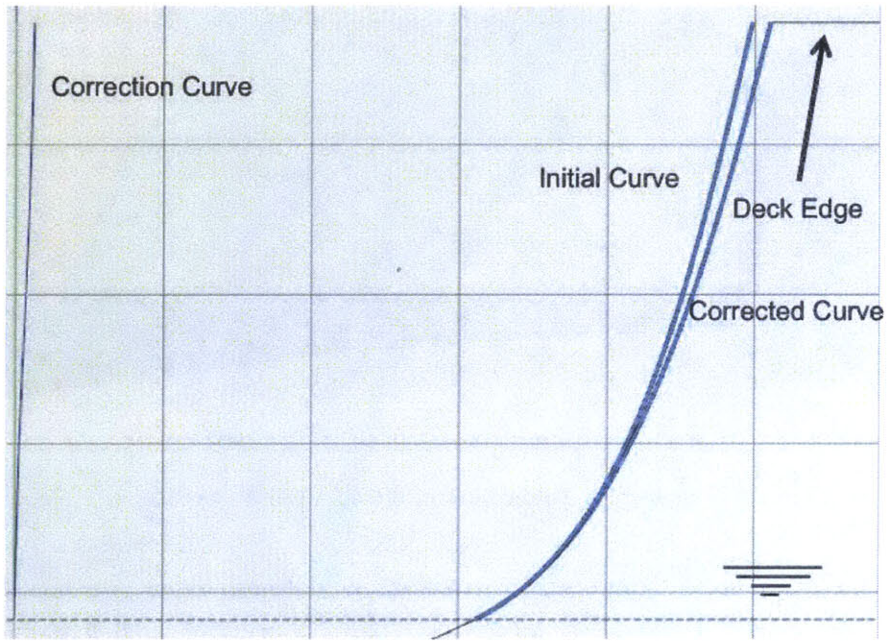


Figure 20 - Correction of Above the Water Transom Edge

The correction was done by using the "*DeltaShift*" transformation of Friendship Framework ®, which is a offset displacement of the parent curve with respect to a specific axis and by using a correction (or "delta") curve. The initial and final sections, along with the correction curve used can be seen in Figure 20. The correction curve is an F-Spline that starts with zero displacement and zero tangent and ends with the displacement needed in order for the transom edge to meet with the aft most point of the deck edge. The zero tangent in the lower part of the F-Spline was selected in order to minimize the effects of this transformation in the smooth transition between the underwater and above the water sections of the transom.

(4) Transom Surface

The underwater transom surface is defined by using a modified version of the build-in "*DeckSurf*" Friendship Framework's feature surface and can be seen in appendix B.2. The most aft edge of the lofted surface is used as an input, in order to ensure that there are no openings between the surfaces. Finally, the transom surface is automatically extended until it meets the centerline (X-Z) plane.

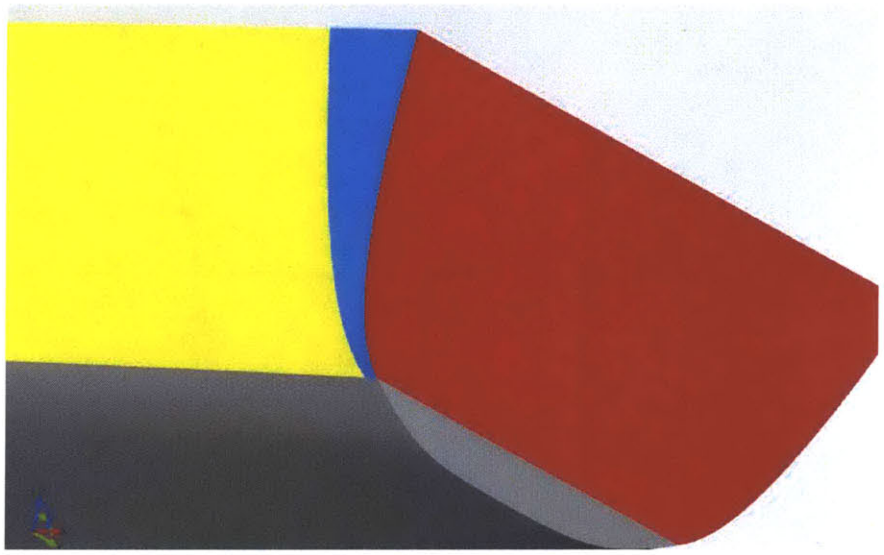


Figure 21 - Transom Area of Above the Water Surfaces

2.4.4. Control Curves

As discussed before, there are two sets of curves in the geometry definition of the hull, the main curves described before and the control curves. The control curves contain information needed by the definition of the cross-section above and below water. As seen before, the curve engines used by the metasurfaces, need as an input the deadrise and flare angles. Also for the part of the underwater portion of the ship defined as a NURBS metasurface, the weight of the intermediate point of the NURBS polygon is needed.

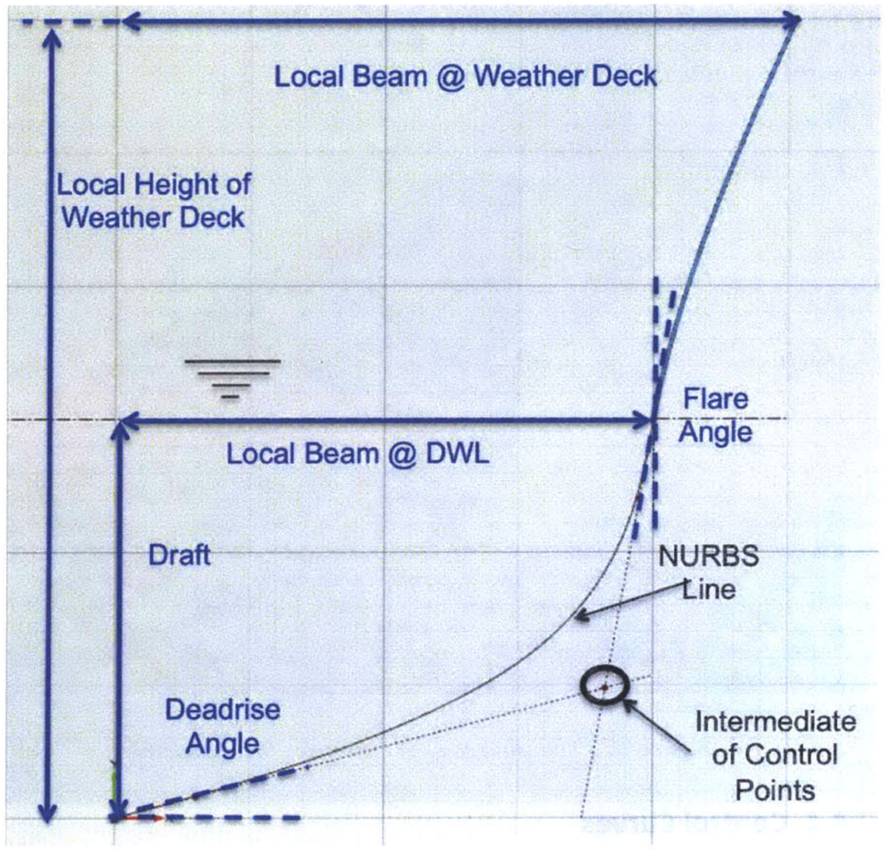

Local Beam @ Weather Deck
Local Height of Weather Deck
Flare Angle
Local Beam @ DWL
Draft
NURBS Line
Deadrise Angle
Intermediate of Control Points

Figure 22 - Total Cross Section

The control curves contain this information for the entire length of the ship and the curve engines acquire this information automatically for each longitudinal position. The flare angle and the fullness factor have been defined as interpolation B-Splines of 2nd order and the user has the ability to control the ordinate of the defining points. The deadrise angle has been defined partially as interpolation B-Spline of 2nd order and partially as a straight line, due to its shape. Figure 23 shows an example on how these curves look like for modeling the USN's DDG-51 type destroyer. The *"Fullness Factor"* refers to the weight of the intermediate of the NURBS polygon discussed in 2.4.2.3(1).

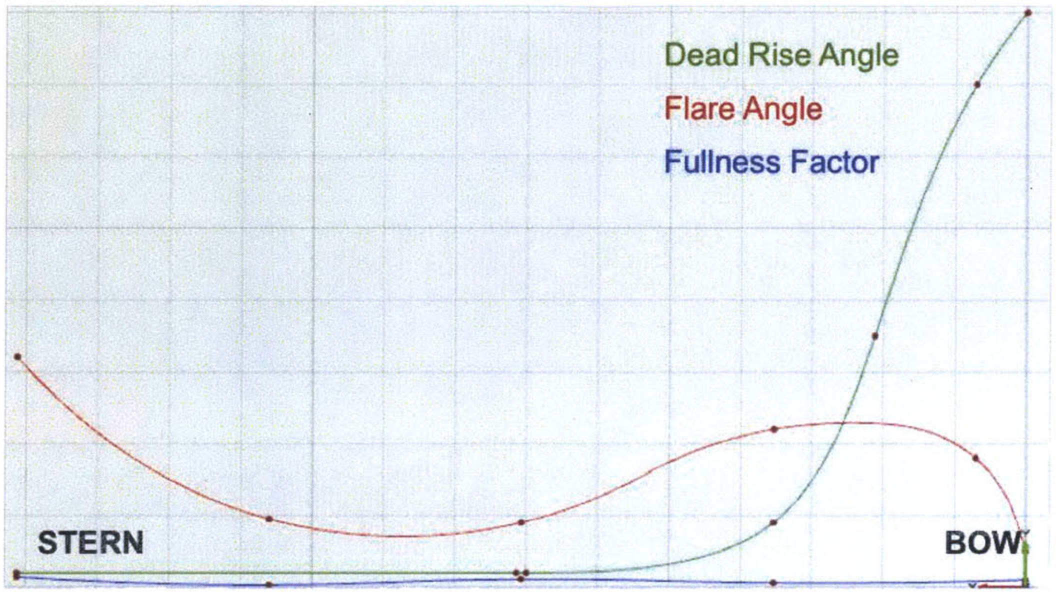


Figure 23 - Example of Control Curves

The user has the ability to control the ordinate of the defining points shown in Figure 23 by using the corresponding variables of the model. Under the "*Variables*" folder of the object tree, the "*Deadrise*", "*Flare*" and "*FullnessFactor*" folders exist, which contain the variables controlling each curve. The values are given in degrees, even though the values actually used to form the metasurfaces are non-dimensional.

The reason for using non-dimensional values as an input to the metasurfaces was because of problems occurring when the length of the ship is drastically reduced. As said before, the control curves are defined as interpolation curves, the maximum value of the deadrise and flare angles for a ship might be of the order of eighty and sixty degrees respectively. Also, both angles are varying considerably along the length of the ship and for this reason, when the length is reduced it can lead to abrupt changes in the slope of the curve, or even to situations where two values of deadrise of flare angle exist at the same longitudinal position.

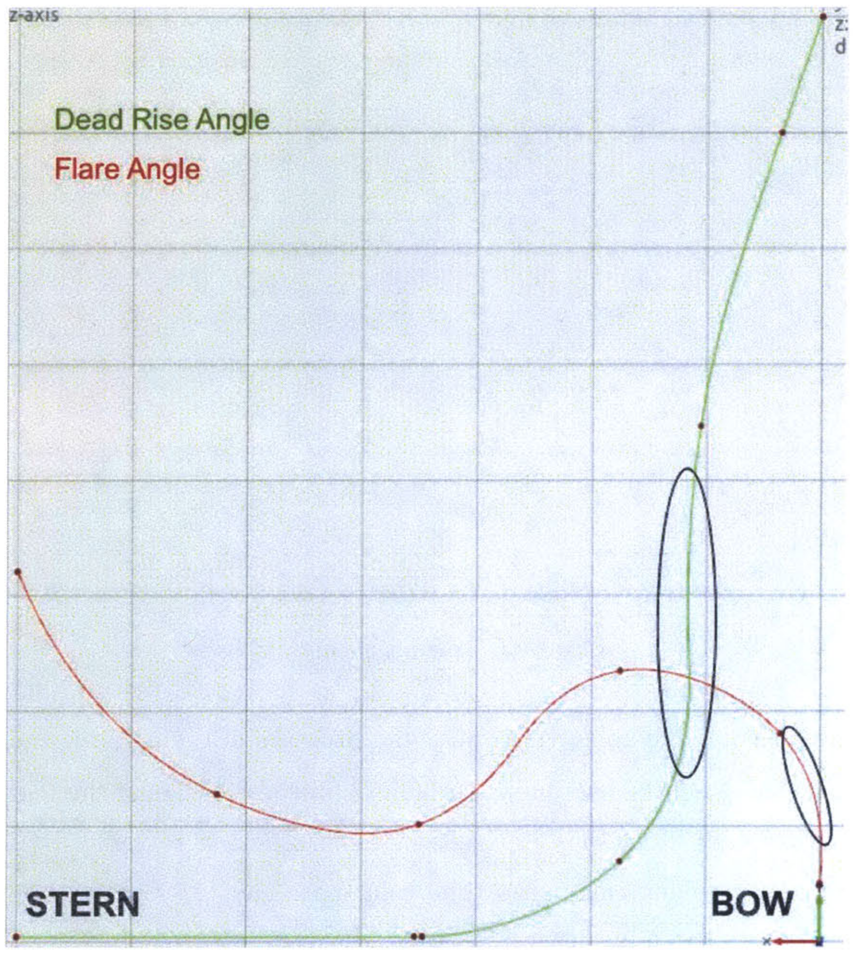


Figure 24 - Problematic Control Curves in Ships with Small Length

These problems lead to discontinuities of the generated surfaces, as the input angles are changing very quickly and abruptly for a smooth surface to be generated. The described issues, associated with the reduction of length can be observed in Figure 24, where the length of the ship was reduced to 70m. By dividing the values of the curves seen in the previous figure by 45 degrees, the problem is removed, as the maximum ordinate value for the new curves can't exceed the value of two. However, in this limiting condition, the shape of the non-dimensional control curves might be such that negative values might occur. For this reason, in the feature definitions of the cross-section that can be seen in the appendix, a check is performed and the minimum value allowed for the flare and deadrise angles is numerically zero.

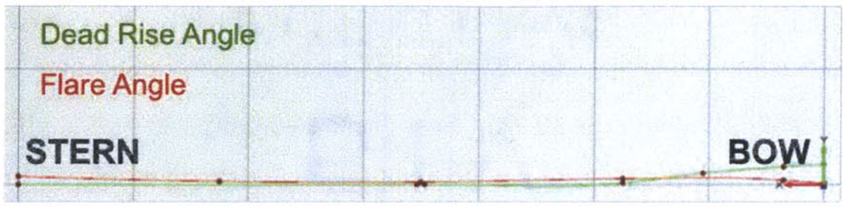


Figure 25 - Non-Dimensional Control Curves

The points defining the control curves are distributed along the length of the ship in longitudinal positions expressed as fractions of the LBP. This allows the points to move according to the changes of the length and thus reduce the complexity of the model, as no interaction of the user is needed for displacing the points forward or aft as the LBP is changing. Table 14 summarizes the longitudinal positions of these points, along with the indication on whether the ordinate of each point is fixed or controlled by the user.

% of LBP	Deadrise Angle	Flare Angle	Fullness Factor
0	Fixed	Fixed	User Controlled
0.05	User Controlled	User Controlled	-
0.15	User Controlled	-	-
0.25	User Controlled	User Controlled	User Controlled
0.495	User Controlled	-	-
0.50	-	User Controlled	User Controlled
0.505	User Controlled	-	-
0.75	-	User Controlled	User Controlled
1.00	User Controlled	User Controlled	User Controlled

Table 14 – Longitudinal Positions of Points Defining the Control Curves

2.4.4.1 Deadrise Angle Curve

The deadrise angle curve has been defined as an interpolation B-Spline for the forward part and as a straight line for the aft part of the hull. Amidships to stern the deadrise is assumed to be constant, which is a valid assumption for most mono-hull combatant ships. The user can control the elevation of the straight line or the value of the deadrise in the aft half of the ship by changing the corresponding value of the mid-ship section.

The rest of the points define the forward part of the curve and two points have been used around the mid-ship section, in order to ensure the smooth connection between the B-Spline and the straight line. These two points, at 49.5% and 50.5% of the LBP, have the value of deadrise that the user selects for the mid-ship section. The points at 5%, 15% and 25% of the LBP are controlled directly by the user through the use of the corresponding variables in the "*Variables*" folder of the object tree.

From the total of seven points used in the definition of the deadrise angle, the user directly controls four of them, as the points at 49.5%, 50.5% and 100% of LBP have the same ordinate value and the point at the FP has a fixed value. The value of the deadrise at the FP was assumed to be equal to 80 degrees for all the hulls generated by the geometry modeler. This values was selected as it gives a good representation of the bow section for most of the mono-hull combatant type ships.

2.4.4.2 Flare Angle Curve

Six points are used to define the flare angle curve and the user has control over five of them, while the sixth is fixed and positioned at the FP. The values provided by the user should be in-line with the definition of the flare as an acute angle, seen in the cross section example of Figure 22. It is assumed that at the FP the flare angle of the cross section will be 5 degrees for all the hulls generated by this geometry modeler. This was considered an acceptable assumption, as the flare angle at this position has values around the value selected for most hulls of interest.

2.4.4.3 Fullness Factor

The definition of the fullness factor curve is based on five points, which can be all controlled by the user. As discussed in section 2.4.2.4(2), the F-Spline metasurface is smoothly connected with the NURBS metasurface, as an automatic blend of the two cross section definitions takes place in the aft part of the F-Spline surface. For this reason, the fullness factor needs to be defined for the whole length of the ship, in order for the

NURBS definition of the cross section to be able to get its required values at any longitudinal position.

2.5. Importing Hull Geometries

As discussed earlier, Friendship Framework® is a very interconnected software that has CAD capabilities. Due to this, importing surfaces and lines is a straightforward task and can be done in multiple widely acceptable formats. This section discusses a tool that was created in order to measure the design parameters of imported hulls as well as a script that automatically populates the measured values in the project's variables. Finally, a simple optimization process is explained that aims to fit the DWL of the imported hull as closely as possible.

2.5.1. A Tool to Identify Design Parameters of the Imported Hull

Under the folder "*Hull_to_be_fitted*" a tool was created, in order to allow the user to read the design parameters of imported hull geometries. For this tool to work properly, the imported hull geometry should be represented by a single continuous surface for each side of the ship, without containing any part of the transom. If this is not the case, then an external CAD tool should be used in order to generate such a surface, or the poly-surface command of Friendship Framework® could bundle all the surfaces into one.

In Figure 26 an example of a preferred and a problematic import of hull geometries is demonstrated. In the problematic case, the user should remove the transom and generate one single surface before importing or a poly-surface should be created in Friendship Framework®. The latter might prove very tricky, as the orientation of the imported surfaces should be taken into consideration and aligned by using "image surfaces" before the poly-surface is defined.

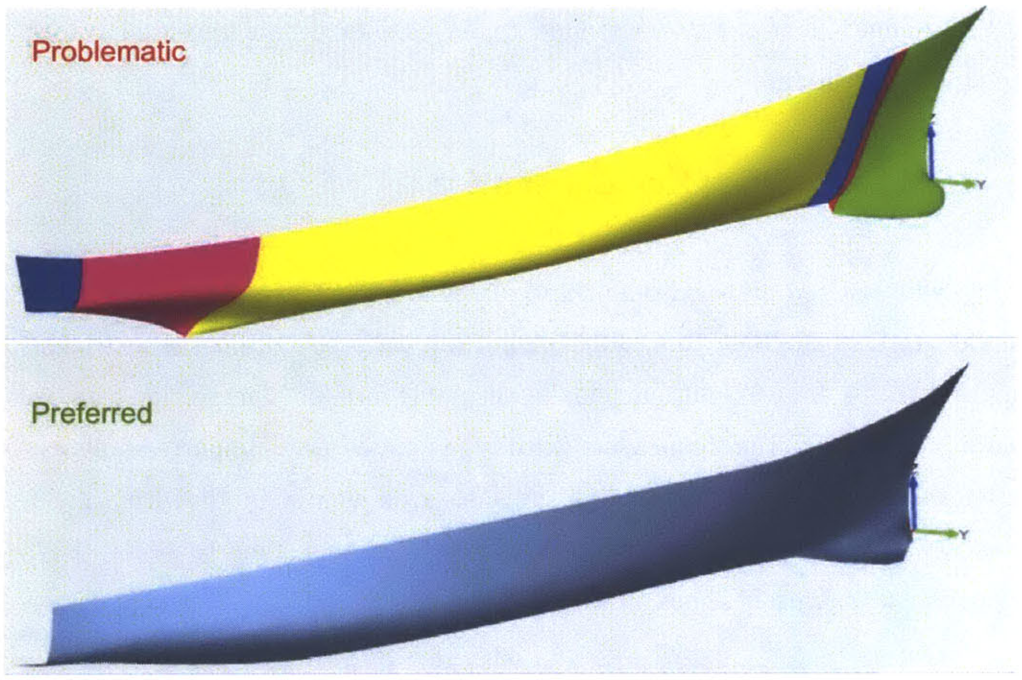


Figure 26 - Example of Imported Hull Geometries

Inside the folder "*Hull_to_be_*fitted", a subfolder with the name "Surfaces" exists; in there the user should populate the corresponding "*Source*" field of the two image-surfaces named as "*PORT*" and "*STBD*" (starboard) with the respective imported surfaces. After that, the tool will automatically calculate most of the design variables needed as measured from the imported geometry and can be found in the "*Measured"* subfolder of "*Hull_to_be_fitted*".

The user should be very careful to check that the following curves extracted automatically are the anticipated.

1. Deck Edge
2. Keel
3. Stem
4. Transom
5. Design Waterline

Depending on the orientation of the imported surface, a wrong edge or intersection surface might be wrongly identified as the DWL, for instance. For this reason, the user should access the definition of the above-mentioned curves and choose the correct edge. Finally, the tool will automatically recognize the design draft of the ship by assuming that the imported hull has its FP at a zero x-coordinate position. If this is not the case, a shift of the imported geometry should be done to make sure that the forward most point of the DWL and the FP has a zero x-coordinate.

This tool is able to identify most of the design variables needed for the definition of a hull through the geometry modeler. The design variables that can't be measured are the following:

1. Fullness Factor values for the entire length of the ship.
2. *Cr_Section_Change*, which indicates where along the length of the ship the transition from the F-Spline metasurface to the NURBS one should be done.
3. Offset angles (FWD, amidships and transom) for the definition of the edge of the weather deck.
4. *StemAngle*, which is the tangent of the stem curve at the DWL.
5. *StemRadiusOverDraf*, which controls the radius of the stem curve in the connection with the keel.

2.5.2. A Script for Populating Automatically the Design Variable

A Friendship Framework® script was created and can be seen in appendix B.10, which automatically reads the measured values and populates the corresponding design variables under the "*Variables*" folder of the object tree. The values of the deadrise at 5% and 15% of LBP were omitted from the script, as at this location a sonar dome might exist. If this is the case, the measured deadrise angles will be wrong to be used, as the geometry modeler does not support sonars.

The user should keep in mind that the measured values of the design variables are indicative and a careful examination of the final values should be done. The accuracy level in which the geometry modeler should approximate the imported geometry depends on the reason we needed to model this geometry. If a hydrodynamic optimization is to be conducted, then the accuracy level might be intentionally lower than in other cases. Whatever the accuracy level might be, it is highly recommended that the user spend some time on examining all the chosen and measured design variables.

2.5.3. FitDWL Optimization

In order to aid the user to find the best values for the variables a simple optimization process was developed to aid with the fitness of the modeled DWL with the one extracted from the imported geometry. Seventeen pairs of points are dispersed along the two DWL (real and modeled) and the transverse distance between them is calculated. All the errors are squared and finally a summation of all these errors becomes the objective function for the optimization algorithm.

The used algorithm is the Non-Sorting Genetic Algorithm II and the optimization process can be found as *"FitDWL"* under the *"Design Engines"* section of the object tree. 40 generations with 52 individuals are used, while the mutation and crossover probabilities are set at 10% and 95% respectively. The design variables for the optimization are shown below, while the only constrain used is one about the quality of the DWL. This constraint ensures that the maximum beam of the DWL will not be greater than the one defined by the user. This is very important, as the value of the design variables can be such that a very bad DWL can be created as it can be seen in Figure 27.

1. Cwp
2. EntranceAngle
3. L_Over_Beam
4. X_Bmax
5. DWLaftAngle

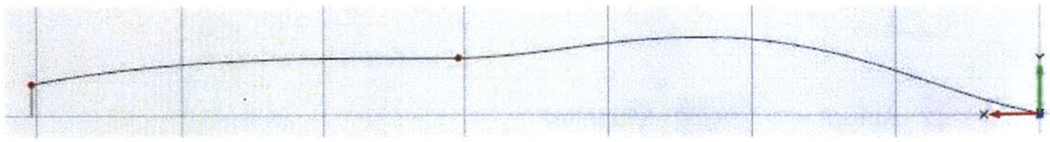


Figure 27 - Bad Design Waterline

2.6. Validation

DDG-51 was considered as a representative ship of the mono-hull combatants and thus it was used to validate that the geometry modeler can represent well such ships. The geometry of the hull was exported from USN's ASSET_v6.3 (Advanced Surface Ship Evaluation Tool) hull-form utility and imported as an IGES file in the Friendship Framework®. The values for all the design variables defining the modeled geometry can be seen in Table 15.

In Figure 28, a comparison between the actual profile curve and the one generated by the geometry modeler can be seen. The two curves match for the most part of the hull and any discrepancies are caused because of the presence of the sonar dome the shape of the bow above the waterline, which are is out of the scope for the geometry modeler. Figure 29 has a comparison of the two DWLs, where we see that the two curves match very well for all the length of the ship.

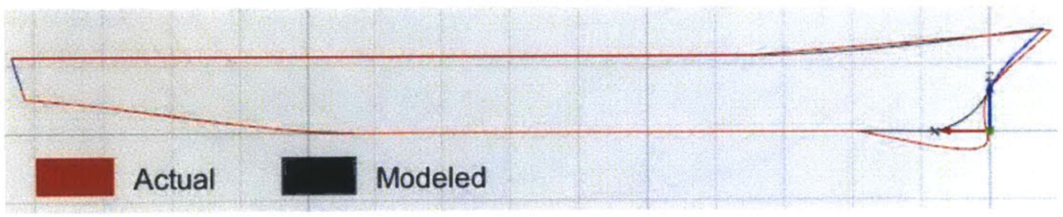


Figure 28 - Modeled vs Actual Profile Curve

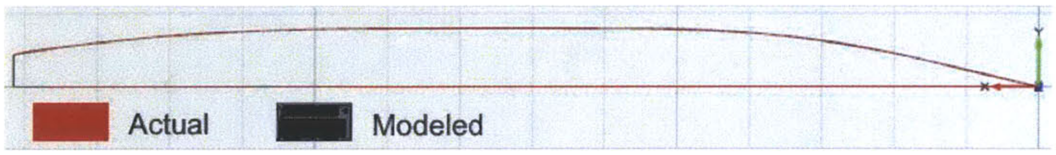


Figure 29 - Modeled vs Actual Design Waterline

In Figure 30 a comparison of the body-plan between the actual geometry and the one generated by the modeler can be seen. Even though a perfect match is not achieved, the discrepancies can be considered small and the ability of the modeler to create mono-hull combatant like geometries can be validated.

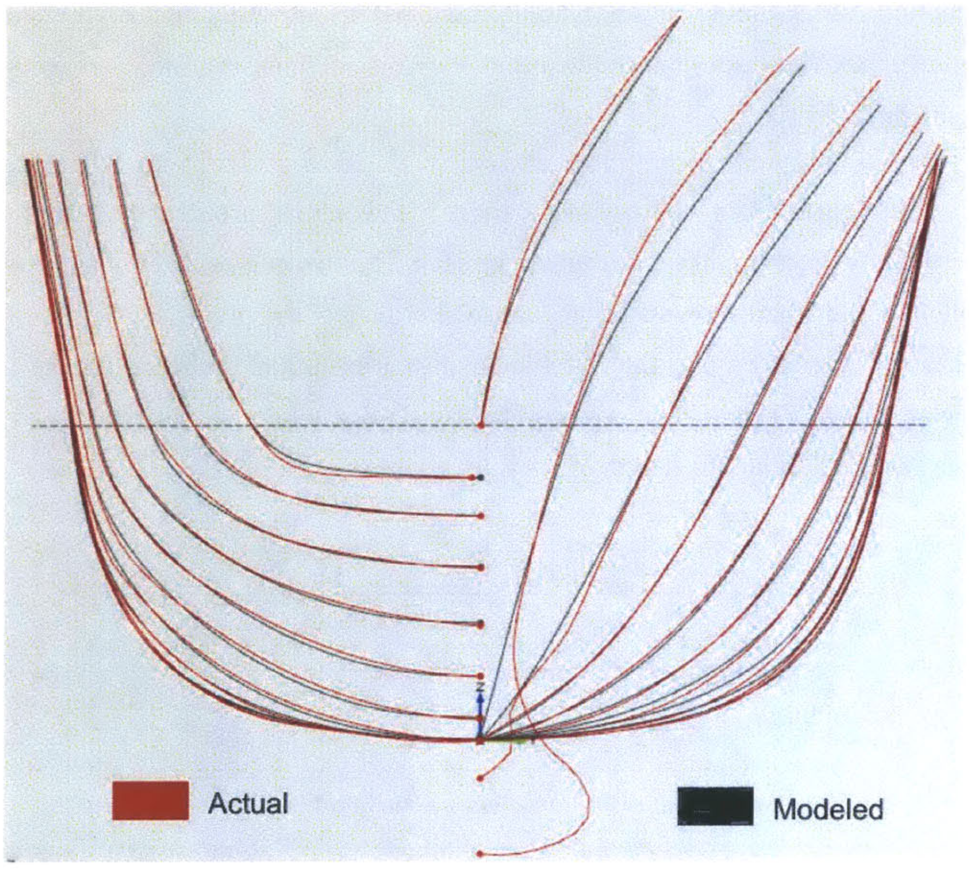


Figure 30 - Modeled vs Actual Body-plan

The general characteristics of the generated geometry are good and acceptable for the early stage design phase that this tool aims for. Furthermore, a more complex model could have generated surfaces and lines of higher quality; however, these benefits would have been lost in the resistance calculations. It is questionable though, if the first order potential flow panel method and the Holtrop method would have been able to distinguish the difference.

Variable Name	Path in Object Tree	Value
Cr_Section_Change	|Variables	0.2
Cwp	|Variables	0.8165
DWLaftAngle	|Variables	9.5608
Draft	|Variables	6.1230
EntranceAngle	|Variables	12.3628
H_BeamTransomOverH_Beam	|Variables	0.5523
KeelAngleAtTransom	|Variables	4.1009
KeelRisePoint	|Variables	0.65
LBP	|Variables	141.2770
L_Over_Beam	|Variables	8.8573
StemAngle	|Variables	63
StemRadiusOverDraft	|Variables	1.5
TransomAngle	|Variables	17.5957
TransomDepth	|Variables	1.0174
x_Bmax	|Variables	0.4520
Deadrise Anle		
0_05	|Variables|Deadrise	60
0_15	|Variables|Deadrise	30
0_25	|Variables|Deadrise	9
0_50	|Variables|Deadrise	1.5
Flare Angle		
0_05	|Variables|Flare	18
0_25	|Variables|Flare	22
0_50	|Variables|Flare	8
0_75	|Variables|Flare	13
1_00	|Variables|Flare	35
Fullness Factor		
0_00	|Variables|FullnessFactor	1
0_25	|Variables|FullnessFactor	0.6
0_50	|Variables|FullnessFactor	1
0_75	|Variables|FullnessFactor	0.4
1_00	|Variables|FullnessFactor	1.2
DeckEdge		
OffsetAngleFWD	|Variables|Auxiliary|DeckEdge	34
OffsetAngleMidShips	|Variables|Auxiliary|DeckEdge	10.2
OffsetAngleTransom	|Variables|Auxiliary|DeckEdge	20
Freeboard_FP	|Variables|Auxiliary	7.9592
Freeboard_MS	|Variables|Auxiliary	5.1770

Table 15 - Summary of Variables for Modeling DDG-51

CHAPTER 3

INTERNAL ARRANGEMENTS

Design of ships has been a process that dates back thousands of years. Even though immense technological advances have occurred throughout the history the design process remained the same. In this process, the selection of the hull geometry was and is the first major step that has to be undertaken. Figure 31 depicts the so-called *"Design Spiral"*, which demonstrates the traditional way of designing ships.

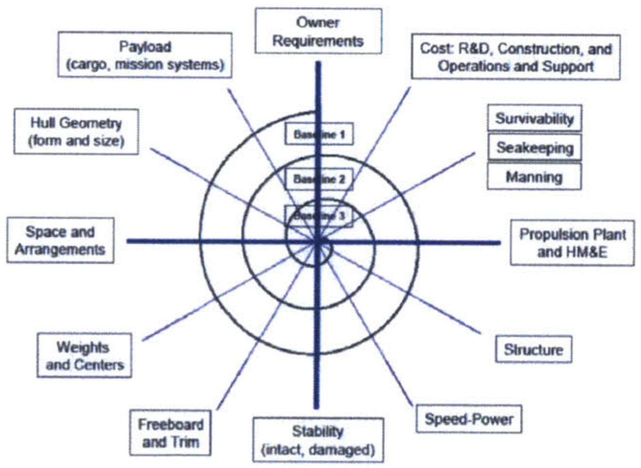

Owner Requirements
Cost: R&D, Construction, and Operations and Support
Payload (cargo, mission systems)
Hull Geometry (form and size)
Survivability
Seakeeping
Manning
Baseline 1
Baseline 2
Baseline 3
Space and Arrangements
Propulsion Plant and HM&E
Weights and Centers
Structure
Freeboard and Trim
Stability (intact, damaged)
Speed-Power

Figure 31 - Ship Design Spiral (Gale, 2003)

The process is iterative, although not necessarily serial as depicted by the spiral, and can be considered as a convergence process, where alterations in the different steps of the design aspects are done in order for the final ship to be as acceptable as possible from the customer. Even though the process predicts for changes to occur in the hull geometry during the second iteration, this most of the times can't realistically happen.

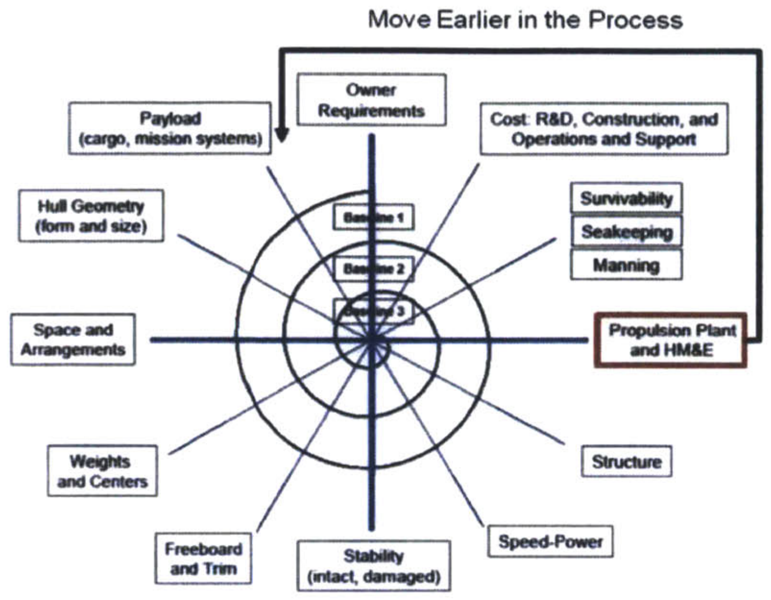
The reason is that the selected hull, most of the times, is predefined based on experimental model data or full-scale data from prior designs. Thus, altering the hull geometry is most of the times seen to be highly risky and costly, if not limited to only small areas.

3.1. A New Era in Navy Ship Designs

In recent years, the design of navy ships has been challenged by an intriguing problem of high complexity. Larger and heavier weapons and sensors need to be used and installed in the most modern ship designs. Interestingly enough, all these system have exponentially greater electric consumption requirements. This is happening mostly for two reasons; one, sensors become more powerful and able to detect threats in greater distances, thus needing more power to operate. Second and more important, weapons are transitioning from the era of gunpowder to the era of electromagnetism. The new weapons substitute gunpowder with electricity on launching projectiles (McNab, et al., 2004) and even go one step further on substituting projectile with lasers (Jason, 2013), which increase even further the electric demand.

This need of high electric loads and the concurrent need for more efficient ships led the designers to bundle propulsive and electric power demand and try to address them together and more efficiently. For this reason, the use of an Integrated Power System (IPS) in new designs becomes more and more popular, driven by the need to increase ship performance, affordability and operability (Doerry, 1994). Further, an additional benefit on survivability, which is important in navy ships, comes from the inherent ability of IPS to allow the separation and distribution of the propulsive and electrical generation equipment in the ship (Doerry & Firemann, 2006).

An extra benefit coming from this combination is the ability for more design flexibility in the early stage of the design. System-to-system tradeoffs with the combat system can be realized and conducted in order to form a better design solution, something not done in conventional designs (Doerry & Firemann, 2006). Realising the importance of the IPS and the benefits that can spawn from the adaptation of such a system, a radical modification of the traditional design spiral has been proposed. In the modified design spiral, the propulsion plant becomes the first step in the design process and now the hull and the rest of the ship adapts to it (Jurkiewicz, 2012).



Move Earlier in the Process
Owner Requirements
Payload (cargo, mission systems)
Cost: R&D, Construction, and Operations and Support
Hull Geometry (form and size)
Survivability
Seakeeping
Manning
Baseline 1
Baseline 2
Baseline 3
Space and Arrangements
Propulsion Plant and HM&E
Weights and Centers
Structure
Freeboard and Trim
Stability (intact, damaged)
Speed-Power

Figure 32 - Modified Ship Design Spiral (Jurkiewicz, 2012)

By streamlining the design of the ship around her propulsive and electrical generation equipment, a more inside-out approach can be followed. This can lead to increased efficiency of the ship, thus increase the affordability and reduce the total ownership cost, which is highly correlated with the ship's propulsive efficiency and not just the hydrodynamic efficiency. For these reasons it was deemed important for the tool to incorporate the modeling of the internal arrangements and then consider them as geometric constraints for any optimization need. An example of the machinery arrangements for a ship using extensively IPS can be seen in Figure 33.

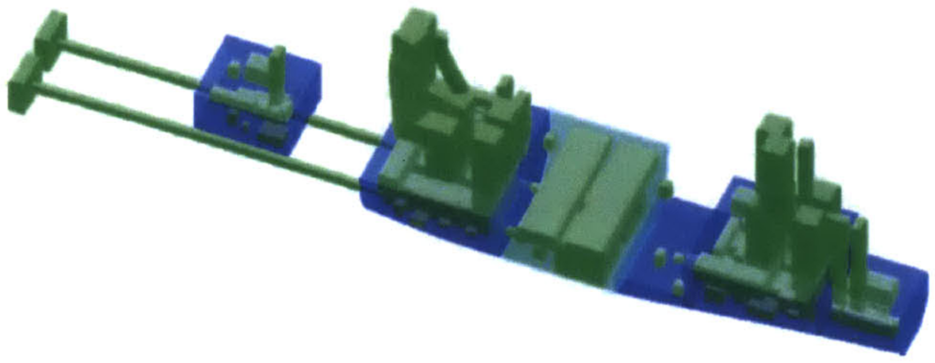


Figure 33 - Machinery Arrangements for an All-Electric Ship (Jurkiewicz, 2012)

3.2. Defining Internal Arrangements

The tool allows the user to use up to six objects, in order to model any internal arrangements of interest. These objects can be used to model engines, motors or large objects like Vertical Launch Systems (VLS), which are commonly used in navy ships. The control of these objects is done from the "*Variables*" folder of the object tree, where the subfolder "*InternalArrangements*" exists. From here the user has the ability to control how many objects to use and also to define their position and dimensions.

3.2.1. Building Equipment Boxes

The internal arrangements are defined as boxes and the user has control over the dimensions, the position and the angle of each box. A cuboid, which is a predefined Friendship Framework's element, was used to define all the equipment boxes (objects) allowing for easy and fast modeling. All the boxes are defined in the starboard (right or y>0) side of the ship and since only mono-hulls with symmetry along the centerline are modeled; all the equipment should be defined at this side. Thus any engines situated at the port (left or y<0) side of the ship and need to be modeled should be mirrored and modeled as they were at the starboard side of the hull.

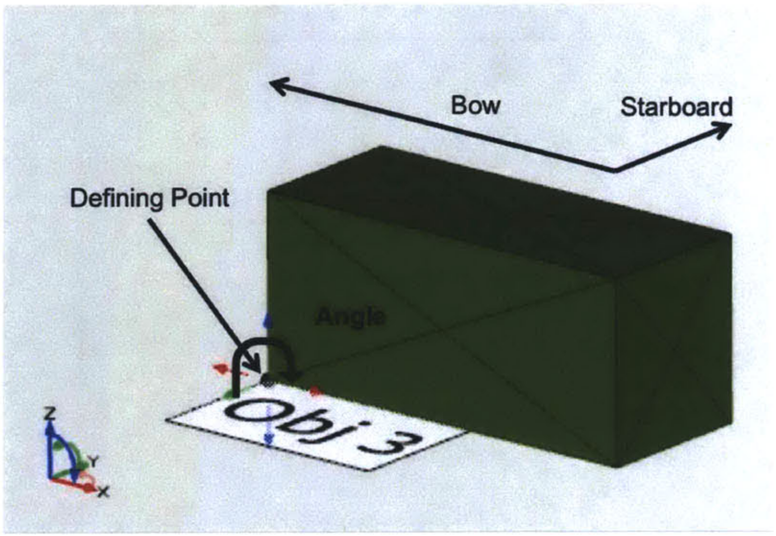


Figure 34 - Equipment Box

The length, width and height of the box are controlled through the respective variables inside the corresponding subfolder of each object in the object tree (ie "*Variables/InternalArrangements/Obj1*"). The user should indicate the dimensions of the modeled equipment increased by any needed clearance, as the tool will use these dimensions to check if there is any interference between the equipment and the hull. The units should be given in the metric system.

The forward, bottom, port point of each equipment box controls its position inside the ship and can be seen in Figure 34. In other words, the positioning of each box is done with respect to its corner closer to the bow of the ship, the keel and the centerline. All the coordinates of this point should be given in meters and are controlled by the "*FWDPortBottomPoint*" subfolder of each box (or object), located inside the "*Variables*" folder of the object tree. The coordinates of this point are defined with respect to the same reference system used for the entire ship, where positive X indicates aft, positive Y indicates starboard (right) side and positive Z indicates above the baseline (or keel).

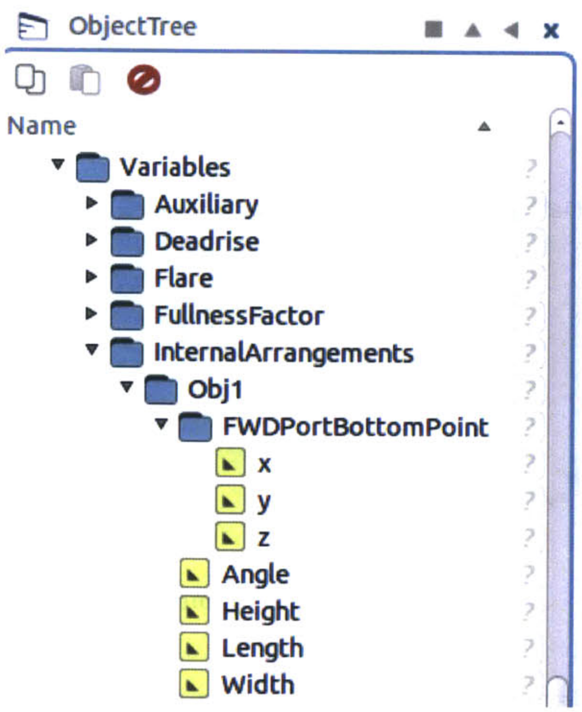


Figure 35 - Object Tree View for Each Equipment Object (Box)

Apart from the position and dimensions, the user has the ability to control the angle that each box has with respect to a transverse axis passing through the defining point of each box described before. This was deemed necessary in order to realistically model the propulsion plant of conventional ships, where the engines and the reduction gears are inclined. This angle should be provided in degrees and a positive value means a clockwise rotation with respect to the defining point and the X-Z plane as seen in Figure 34. The effect of a 10-degree angle to an equipment box (object) can be seen in Figure 36.

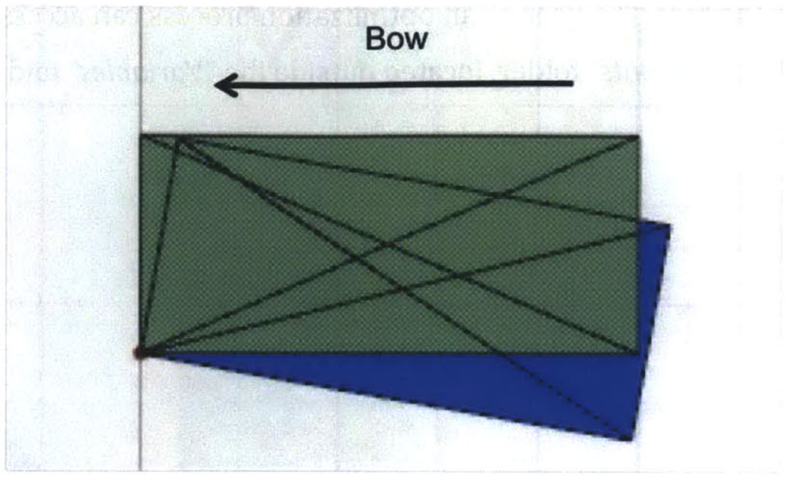


Figure 36 - Example of Inclined Equipment Object

Finally, the user can control how many equipment objects to use through the *"UseObj"* variables inside the *"InternalArrangements"* folder of the object tree. When a zero is set in one of the *"UseObj"* variables, then the corresponding equipment is not defined and not taken into consideration as a geometric constraint.

3.2.2. Interference with Hull Geometry

Modeling major internal arrangements of a ship is the first step, in the more interesting process of checking whether these object actually interfere with the hull's geometry. The tool is able to automatically check whether an object is inside or outside the hull and thus issue a warning for the user, which can also be used as a geometric constraint in any optimization.

The *"Hardpoint"* feature element of Friendship Framework is used to establish whether a box interferes or not with the hull geometry. The check is performed in the two lower and outboard corners of the cuboid representing the equipment boxes, as it can be seen in Figure 37. If any of the two corners interferes with the hull, then a warning is issued indicating this. The warnings from all boxes used, as specified by the user, are bundled together into one parameter. This parameter takes the value of one if all the used boxes are clear of the hull and zero in case any of them at any of its lower corners interferes with the hull. The user or an optimization process can access this parameter in the "*Internal_Arrangements*" folder, located outside the "*Variables*" folder.

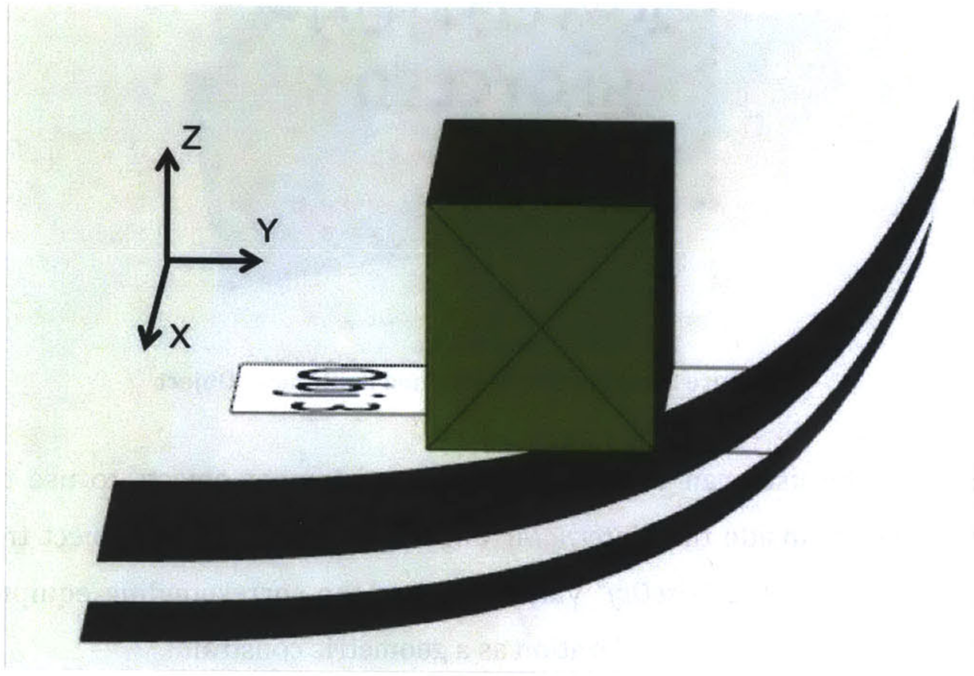


Figure 37 - Example of Hull Interference Check

3.3. Discussion on Modeling Internal Arrangements

This tool gives the ability to the user to use 6 cuboids in order to model the internal arrangements of a ship. As the number of available boxes is limited, the user should use a clever way on managing them. The goal should be to use the least amount of objects and at the same time be able to capture any possible interference of the installed equipment with the hull. Also the user should have in mind that all the equipment should be defined in the starboard (right) side of the ship, even if in reality it is not there. This comes from the fact that this tool models mono-hulls with symmetry around the center-plane and thus everything are modeled in half of the hull.

In many navy ships, the machinery rooms contain engines that are arranged in couples. In this case the user should model only the outer engines of each machinery space, as these would be the ones possible to interfere with the hull geometry. On other words, the engines closer to the centerline will not interfere with the hull geometry if the outer engines do not interfere first. Further, the user should establish which piece of equipment is most likely to interfere first with the hull geometry or is closer to the skin of the ship and model only this one, in order to save resources. Many times, the most limiting equipment is not an engine but a reduction gear or a chilled water unit.

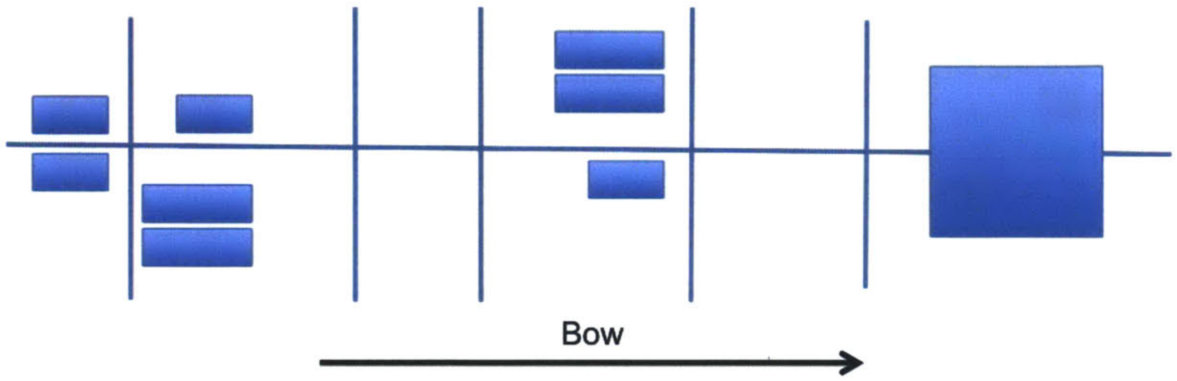


Figure 38 - Notional Arrangement of Engines Rooms and VLS

Saving resources allow us to model more pieces of equipment, as the 6 equipment boxes can be used not only to model machinery arrangements, but weapons like VLS and guns as well. Another reason for modeling only the most important (or critical) equipment is that if we manage to effectively model the arrangements by using less than the 6 boxes, then the optimization process will be sped up, as less checks and calculations will need per each variant.

In the notional example of Figure 38, a total of 9 pieces of equipment exist that need to be ensured that will not interfere with the hull. Each box represents a piece of equipment, containing any needed clearances around it. The big square in the bow is a VLS system, extending deep enough in the hull that is likely to interfere with it. The smaller rectangles represent ship service engines used for electric generation, while the larger rectangles represent the propulsion engines of the ship.

Based on their notional position (longitudinal, transverse and vertical), a sorting can be performed, in order to understand, which of them are really the most critical. The goal is to identify the minimum equipment that if monitored for interference with the hull, will guarantee that all the equipment if these objects do not interfere, then none has a problem of touching or penetrating the hull. In Figure 39, the critical equipment that can't be omitted from checking if interferes with the hull, has been marked in red.

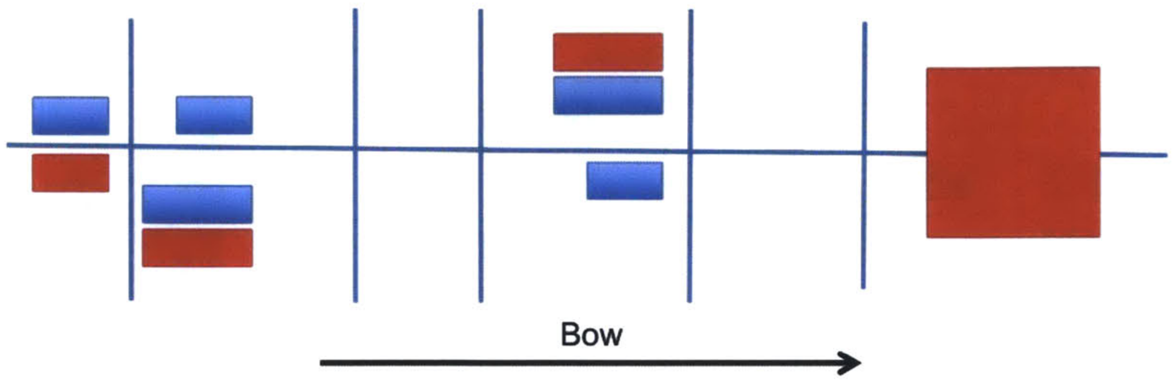


Figure 39 - Determining the Critical Equipment to Be Modeled

Now that we have indicated, which equipment is more critical and likely to penetrate or interfere with the hull, we see that we need only 4 objects to model in our tool. Because of the symmetry properties of the hulls modeled by the tool, only half the VLS needs to be modeled. Following same reasoning, the engine in red at the port (left) side will be modeled, as it was located at the starboard (right) side.

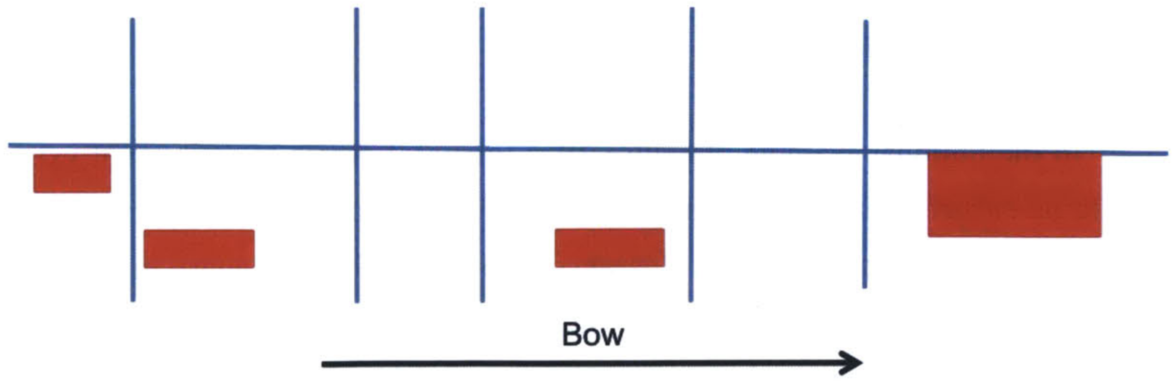


Figure 40 - Critical Equipment to Be Modeled

Figure 40 shows how the critical equipment should be modeled, while Figure 41 shows how the tool actually models and represents the critical equipment. For these, four equipment objects, the model will check automatically if any of the lower and outer corners of any box interferes with the hull.

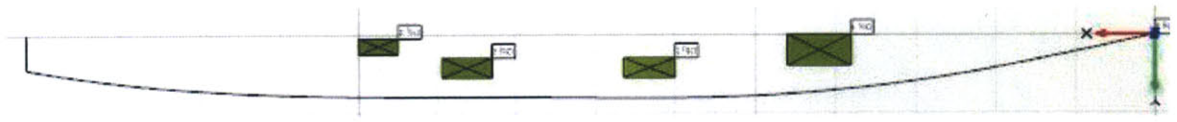


Figure 41 - Critical Equipment Modeled by the Tool

(This Page Intentionally Left Blank)

CHAPTER 4

RESISTANCE ESTIMATION

4.1. Introduction

Different methods on predicting the resistance of a ship exist and have been used for many years now. The most accurate and expensive, at the same time, is the model testing method, where a scaled model of the ship in interest is built and then tested in a towing tank. With this method, very accurate and reliable results for the ship's resistance can be attained. Due to the nature of this method, however, it can't be used to automatically and quickly provide the resistance estimation for the generated hulls during an optimization.

Fortunately, there are many numerical and computational methods that can be utilized for this task. Each method has different strong and weak points, as well as different unique characteristics that should be taken into consideration in the process of selecting one to be used for some specific application. All these methods can be broken down to four major categories; the empirical methods, the potential flow solvers, the (full) viscous flow solvers and hybrid solvers using both potential and viscous flow theories. This breakdown is very general and inside each category multiple subcategories can exist that distinguish and describe each method in a more detailed and appropriate way.

From these four categories, the solvers considering the viscous nature of the flow around a ship are the most accurate and closer to reality, when compared with the other methods. The accuracy of these codes is mainly based on the fact that they try to solve the flow by making the least assumptions possible, thus trying to capture almost all of the physical phenomena that exist in the flow. This level of accuracy however comes with a high cost in the needed calculations to be carried out. For many years the available computing capacity didn't allow these codes to be used, as the needed calculations were dramatically lengthy.

The improvement of computers in the recent years, however, allowed for these codes to be revived and be used more and more. Nowadays, the existence of computers with multiple processors able to conduct millions of computations per second has drastically decreased the needed time for a complete calculation of the flow around a hull. This allowed designers to use more easily and broadly viscous flow solvers in the design and optimization of ships. The attained results are of such a good quality that many times are substituting the model tests and there is a general trend for such codes to completely replace model tests and experiments (Marzi, circa 2006). Nevertheless however, such calculations are highly time consuming and thus such codes is difficult to be used when thousands of variants or hulls have to be generated during an optimization process.

By removing the viscous effects of the flow around the hull, the potential flow solvers are much faster when compared to the viscous solvers. The accuracy of these codes is compromised, as the viscous part of the flow is not considered. However, these codes are able to produce reliable results under specific conditions, when the predominant component of the resistance is due to the waves generated by the forward movement of the ship. The ability of these codes to estimate the resistance of the ship in about 2-5 minutes makes them a very good candidate for optimizations where multiple designs or variants are to be generated (Larsson & Raven, 2010). Further, these codes are able to produce reliable comparative results when the geometry of different hulls needs to be compared, as long as they are used in cases where the viscous phenomena of the flow are not dominating (Larsson & Raven, 2010).

The ability of the potential flow panel method codes to solve fairly accurately the free surface problem, led to the creation of hybrid codes. The main goal of these codes is to produce high quality and reliable results, while at the same time reduce the computation time needed by a viscous code. This is accomplished by using the potential flow solution for the free surface problem as an input or condition in a viscous solver. This allows for a considerable reduction in the needed computations and as long as the solution of the free surface is reliable the results are pretty accurate (Raven & Starke, 2002). Even when implementing this hybrid approach however, the time needed to conduct a complete calculation is of the order of several hours, which again might be very expensive and lengthy for an optimization process with multiple hulls and variants.

Finally and apart from the pure numerical methods, the resistance of a ship can be estimated by using empirical methods. These methods are most of the times regression and statistical models that are based on a dataset of ships or models for which the resistance has been estimated by the reliable method of model testing. In this category belong the systematic series and the Holtrop methods, which both of them have been used extensively and with respectable success in the past years. Even today, they are considered to be a good and reliable starting point for the resistance estimation in the early design stages of ships. The extremely high speed of these methods however, has its downside, as the resolution of these methods is very coarse and the parameters, used to estimate the resistance, are in the particulars level.

Due to the high speed of the Holtrop method and the potential flow panel method solvers, both of them were used in the tool. The user should be aware about the advantages and disadvantages coming with each method and use each one respectively. In the next sections, a limited and short description of these two methods is given for clarity purposes.

4.2. Holtrop Method

4.2.1. Description of the Method

This method was first introduced by Holtrop and Mennen in October 1978 and was based on a regression analysis of random model experiments and full-scale data (Holtrop & Mennen, 1978). This method was then revised and corrected in order to give more accurate predictions for high-block ships with low L/B and more slender naval ships (Holtrop & Mennen, 1982). Finally, the method was re-analyzed in order to include published data from the Series 64 hull forms and its final version was based on the analysis of test results from 334 models (Holtrop, 1984).

Parameter	Comment	Parameter	Comment
L	LBP [m]	A_{BT}	Transverse Bulb Area [m^2]
B	Beam [m]	i_E	DWL Entrance Angle [deg]
T_F	Forward Draft [m]	C_M	Mid-ship Section Coefficient
T_A	Aft Draft [m]	C_{WP}	Water-plane Area Coefficient
∇	Displaced Volume [m^3]	A_T	Immersed Transom Area [m^2]
S_{app}	Area of Appendages [m^2]	$1+k_2$	Appendages Coefficient
C_{stern}	Stern Shape Coefficient, depends on after-body form	lcb	Longitudinal Center of Buoyancy as fraction of L and with respect to $L/2$
h_B	Vertical Center of A_{BT} above keel-line [m]		

Table 16 – Input Parameters for the Holtrop Resistance Estimation Method

This method provides a set of equations defined through the regression analysis of the studied dataset. The input to the method is narrowed down to the 15 particular level parameters reported in Table 16. The information contained in this set of parameters is then translated and used in the corresponding equations of the method in order to predict the resistance of the ship in different speeds. The decomposition of the resistance in different parts can be seen in the equation below. The user has the benefit of getting an estimate for each one of these components and thus giving him a better understanding of the ship's resistance characteristics.

$$R_{TOTAL} = R_F(1+k_1)+R_{APP}+R_W+R_B+R_{TR}+R_A$$

Symbol	Comment
R_{TOTAL}	Total resistance of the ship
R_F	Frictional resistance as predicted by the ITTC-1957 correlation line
R_{APP}	Resistance of appendages
R_W	Wave-making and wave-breaking resistance
R_B	Pressure resistance due to bulbous bow
R_{TR}	Pressure resistance due to transom immersion
R_A	Model-ship correlation resistance
$1+k_1$	Form factor of the hull calculated by a proposed formula based on the particulars of the ship

Table 17 - Nomenclature of Resistance Decomposition adopted in the Holtrop & Mennen Method

A very important and interesting aspect of this method is the incorporation and prediction of the appendages resistance. Even though this resistance may greatly vary from ship to ship, the method was proactive enough to distinguish the differences on resistance coming from different types of appendages. The user should be careful of the mixture of different types of appendages when selecting the k_2 coefficient, as a weighted average based on the corresponding areas should be used. At any case, consultation of the initial papers describing the method should be sought.

4.2.2. Implementation in the Tool

The Holtrop & Mennen method was used in the tool by using a Scilab® function that was publicly available in the community's portal. Scilab® is a software very similar to Matlab® that allows the user to build function and executable scripts, which then can be called and accessed by other software in batch mode (ie without a GUI). The function found on the Internet was modified and corrected and then benchmarked against the example and the results Holtrop reported in his latest correction of the method (Holtrop, 1984).

The needed inputs for the method to work are automatically gathered from the geometric definition of the hull, converted in the needed format and then fed to the

Scilab® function. The output of the function is then automatically read and the corresponding parameters are automatically populated in the model. The user is able to find the input and output parameters in the *"/Computations/Holtrop"* subfolder of the object tree. As there are no appendages and bulbous bow in the hull geometry definition the corresponding input parameters are always zero. The user should never change or alter the definition of the input parameters, as they are automatically calculated and read by the geometry of the hull. The correct process is to alter the variables controlling the geometry that can be found in the *"Variables"* subfolder of the object tree.

4.2.3. Discussion

The user has the choice of using the Holtrop & Mennen method to estimate the resistance of a designed hull or even carry an optimization to minimize the resistance. Even if this method extremely fast to provide an estimate of the ship's resistance, it should not be forgotten that the level of accuracy and the resolution of this method is inherently small. For this reason small changes in the geometry of the hull will not be noticeable by this method, even if they might induce big changes in the total of the ship.

As the method recognizes the different hull-forms by only looking a set of their particulars, it is possible that different hull geometries can have the same or almost the same set of particulars. In this case, for the Holtrop & Mennen method, these hulls will be the same and identical and no distinction between them would be realized. Thus, it would be impossible for the designer or an optimization algorithm to draw any conclusions on how the resistance of the ship might be affected by small changes happening in the geometry of the hull. In order to capture such interactions and effects a higher resolution method should be selected. Such a method could be a viscous flow solver that is highly accurate and time consuming or a potential flow solver that is less accurate but much faster.

4.3. Potential Flow Panel Method (Totale)

4.3.1. Description of the Method

In this method, the flow around a body is treated as inviscid and irrotational, or in other words as a potential flow. For this reason the velocity potential, velocity and pressure in such a flow are governed by the following equations. In such a flow the surface of the ship and the free surface are represented by panels of specific source distribution. Appropriate boundary conditions should be used in order to model the physical problem correctly. Ultimately, the interaction of these source distributions of the panels with the flow of the fluid generates the wave and pressure field around the ship and thus the resistance can be estimated.

Velocity Field: $\vec{u} = \nabla\varphi$

Laplace Equation: $\frac{\partial^2\varphi}{\partial x^2} + \frac{\partial^2\varphi}{\partial y^2} + \frac{\partial^2\varphi}{\partial z^2} = 0$

Bernoulli Equation: $\vec{u} = \frac{1}{2}\nabla\varphi \cdot \nabla\varphi + \frac{p}{\rho} = constant$

The free surface panel method utilized in the tool is a first order, linear panel method, developed at the University of Genoa by Bruzzone (1994) and Brizzolara (2000). The code is very robust and widely validated over the years on different kind of hulls. The selection of this particular code, in fact, derived from the very good results demonstrated by this method in the particular case of high-speed displacement mono-hulls with transom stern.

The boundary element method (BEM) used for the estimation of the wave resistance is a first order Rankine sources panel method with a linearized free surface condition around the double model flow. This formulation is very similar to what initially was proposed by Dawson (1977), but reformulated (Brizzolara, 2000) to obtain a better stability at high Froude numbers, including some typical non-linearities particularly relevant for high-speed transom stern hulls.

The total velocity potential Φ is decomposed in the potential of the undisturbed free stream uniform flow and a perturbation potential ϕ, as follows.

$$\nabla\Phi = \nabla\phi + U_\infty \cdot \vec{i} \qquad \text{or} \qquad \Phi = U_\infty x + \phi$$

The boundary conditions applied to the problem are:

Impermeability of hull surface: $\vec{n}\cdot\vec{v} = \vec{n}\cdot\nabla\Phi = 0$ over S_B

Tangential flow to free surface: $\nabla\Phi\cdot\nabla\zeta = 0$ over S_F

Dynamic condition on the f.s. : $g\zeta + \frac{1}{2}\nabla\Phi\cdot\nabla\Phi = \frac{1}{2}U_\infty^2$ over S_F

Radiation condition to infinity: $\underset{x \to -\infty}{\Phi \to U_\infty \cdot x}$ & $x \to -\infty$ over S_F

The free surface S_F is defined, in general, by the unknown equation $z = \zeta(x,y)$.

A linearization method similar to the one proposed by Dawson (1977) is used at the boundary conditions. The problem is then broken down and first the double model problem is solved and its solution is consequently used in the free surface final solution. The problem is solved numerically by means of a boundary element method, which employs a distribution of sources with constant intensity over the quadrilateral panels of the body and free surface. The numerical four-point upstream derivation scheme is used in both longitudinal and transversal directions. Both the double model and the free surface problems are solved by means of sources distribution alone.

A peculiar zone of the hull is the transom area, where the flow separates for high speeds. In these cases, the potential flow can't allow for the separation to occur but the main characteristics of the flow remain the same, if the correct assumptions are made. As it can be seen in Figure 42, a smooth separation of the flow aft of the transom is generally true for "well-designed", high-speed transom stern hulls at their design speed. The free surface is then divided into two zones: one aside of the hull and one behind the transom stern.

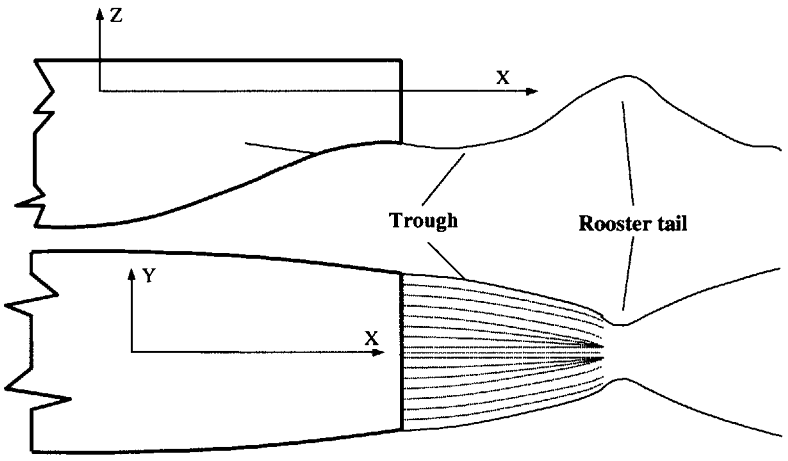


Figure 42 - Unique Shape of Flow at the Transom Area at High Speeds

The second zone is characterized by a certain inflow, which is originated by the hull's outflow at the stern. A special boundary condition must be imposed at the beginning of this area. Following the physical observations (Saunders, 1957), which indicate a sharply separated flow coming out of the transom forming a wave trough first and then a crest (rooster tail), we impose that the flow is tangent to the hull surface just before the transom edge. Finally, the pressure on the stream surface detaching from the transom is assumed to be the atmospheric.

4.3.2. Implementation in the Tool

The potential flow panel method code developed by Bruzzone and Brizzolara was used in the tool in the form of an executable Linux program. Even though the code has the ability to calculate the resistance of a free hull, all the calculations are carried out by assuming the ship to have fixed trim. This is done primarily in order to conserve computational time, which valuable when optimization processes with large number of variants are used. Before using the resistance data of the code, the user should run some investigative computations in order to decide the correct settings for the size and the number of panels. This is a very important step and should be done when the dimensions of the hull-forms will change considerably. Unfortunately, there are no clear guidelines on how to choose the setting of the panels, apart from experience (Larsson & Raven, 2010).

All the inputs that the code needs in order to operate are automatically created by the generated geometry and fed to the code. The output file containing the information about the resistance of the ship is read and the corresponding parameters in the model are automatically populated. All the input parameters, both for the geometry of the ship and the settings of the code can be found in the "*/Computations/PanelCode*" subfolder of the project tree. The output parameters along with the any other calculation that is needed but not carried out by the code can also be found in the same subfolder. Again the user should never alter the definition of any input or output parameters in this folder and changes needed should be done from the "*/Variables*" folder of the object tree.

Apart from the output file the code generates and contains information about the wave-making resistance of the ship, other files containing information about the generated meshes and the values of the velocities, displacements and pressures are also generated. The user has the ability to transform these files by using the auxiliary program "totech" in a form recognizable by the Tecplot® software. The new output files can then be imported into Tecplot® for visualization, which is an important step when the hydrodynamic performance of a hull is evaluated by the use of such a code. Unfortunately, due to difficulties on the file management of the Friendship Framework® version 2.4.8, it was opted that this post processing of the data should be carried out by the user and outside of the tool and not automatically for each generated geometry, inside the tool.

4.3.3. Discussion

Clearly the ability of the user to use a potential flow panel method code to carry out the needed resistance calculations is very important and helpful. Especially when the code has been tested and evaluated with experimental data. This method is, of course, not as fast as the Holtrop method, but it is able to produce more accurate and reliable results. At the same time, the resolution of this method is considerably higher from that of Holtrop and thus changes in the geometry can be reliably and directly translated to possible changes in the resistance of the studied hull geometry.

The user should keep in mind, however, that this method is not perfect and it should be used inside the bounds of her assumptions. The most important and major assumption done in formulation of the panel methods is the properties of the flow, which considered as potential. That been said, great care should be given that in the cases where the resistance is estimated by such a method, the viscous effects are not dominant. As the method is very accurate on the prediction of the generated waves on the free surface, predictions of the resistance at the Froude regime where the wave resistance is dominant will be more reliable and accurate.

Another serious consideration that should be taken into account is the assumption made about the flow at the transom areas. It was assumed that the flow of the water is tangent on the last panels of the hull, which leads to the generation of a trough in stern of the hull. This is valid and following the physical observation for all the transom stern ships above a certain speed, at which the transom is considered "dry". Since, we are treating the flow around of the hull as if the transom is always "dry", it is expected that in the cases where the speed is low and the transom "wet", the panel code will not generate accurate enough results, as it can't follow well the physical problem (Larsson & Raven, 2010).

If the designer needs to estimate the resistance of the ship in speeds low enough where the transom is not "dry" yet, the drawback of not modeling accurately the physical phenomenon should be seriously considered. A more detailed and careful examination of the generated wave patterns around the hull should be carried out, in order to arrive to safer conclusion. Another choice would be to use both the Holtrop and the panel methods in conjunction, in order to attain a better understanding about the hydrodynamic performance of the hull.

CHAPTER 5

SYSTEMS OPTIMIZATION

5.1. Problem Formulation

The final part of the tool is a systems optimization process between the hull geometry and the equipment of the ship. The geometry of the hull can be considered one of the most important systems of the ship, as it affects greatly her performance and capabilities. Further, the importance of the hull geometry increases, if it is studied under the scope of total ownership cost, as a hull of superior hydrodynamic performance can result to considerable fuel savings.

5.1.1. Discussion

The acquisition cost of a ship is just a part of the total cost that the ship will induce to the owner. The maintenance and operational costs are the major parts of the total ownership cost, as they continue to burden the owner for all the years the ship is in service. A considerable part of the operational cost comes from the cost of fuel to run the ship throughout her lifetime. For this reason, a better-designed ship that will need less energy to move through the water will allow for savings in the total ownership cost.

Another aspect that can greatly affect the efficiency of the ship and thus the total ownership cost is her propulsion and electrical generation plants. At these two plants the owner's money are consumed in the form of fuel to allow the ship to operate. This descriptive scheme on how these two systems are directly connected to the owner has driven the designers on trying to improve the efficiency of the propulsion and electric generation systems.

For many years these improvement efforts were focused on the two systems separately by trying to improve engines and generators. However, in the recent years, there are efforts to improve them together and collectively, as better results can be realized this way. Hybrid electric propulsion systems, where an electric motor is installed on the propulsion shaft has been designed and implemented in USN's DDG-51 destroyer class ships (McCoy, et al., 2007). At the same time the need for higher efficiency has driven the design of an all-electric ship with extensive use of IPS, as discussed in section 3.1.

Traditionally the selection of the propulsive and electrical generation equipment had to be done under the major constraint of fitting this equipment inside the hull. Trade-offs on the size and type of the engines had to be made, in order to ensure that they will fit inside the preselected hull. Also, in conventional ships, shaft lines and elevation or depression of engines were also constrained by the physical dimensions of the machinery rooms and the ship itself. Thus, many times, a sub-optimal, in terms of efficiency, solution had to be adopted and affect the operation and efficiency of the ship for lifetime.

This tool has a different approach, where the hull of the ship is adapted to the selected propulsive and electrical generation systems based on a hydrodynamic optimization. This allows the designer to quickly seek for a hull that will better match with the equipment installed inside. Going a step further, the designer would be able to conduct trade-off studies on how the hydrodynamic efficiency of the ship is affected by the selected propulsion and electrical plant. Thus, a better understanding on the interaction between these systems can be realized and a more educated decision can be made.

5.1.2. Design Variables

Treating the position and size of the major equipment inside the hull as fixed, an optimization process trying to reduce the resistance of the ship can be formulated. The resistance of the ship for a given ship is mainly controlled by the geometry of the hull and more precisely from the geometry of the underwater part of the hull (quickwork). For this reason the design variables for a problem like that would be the parameters that affect the shape of the underwater hull surfaces.

The potential design variables for an optimization can be identified by looking into the definition of the geometry modeler, of the tool, done in section 2.4. The total number of variables needed to describe the geometry of DDG-51 in Table 15 at section 2.6 were 34. These variables are all the variables used to describe the geometry of the hull. However, we should break them down into separate sets, based on their impact onto the resistance and whether they induce global or local changes in the geometry.

5.1.2.1 Auxiliary Variables

The offset angles used in the definition of the edge of the weather deck in section 2.4.3.1(2) can be considered as auxiliary variables. Even though, these 3 variables affect the general shape of the hull, they will not induce any changes in the resistance of the generated hulls. The induced changes are limited in the above-the-waterline part of the geometry and thus not contributing in the resistance generated by the underwater volume of the ship. The variables defining the freeboard of the ship at the FP and amidships are also not affecting the resistance of the ship and thus characterized as auxiliary, under the scope of resistance optimization. A summary of all the auxiliary variables can be seen in Table 18.

No	Variable Name	Path in Object Tree
1	*OffsetAngleFWD*	|Variables|Auxiliary|DeckEdge
2	*OffsetAngleMidShips*	|Variables|Auxiliary|DeckEdge
3	*OffsetAngleTransom*	|Variables|Auxiliary|DeckEdge
4	*Freeboard_FP*	|Variables|Auxiliary
5	*Freeboard_MS*	|Variables|Auxiliary

Table 18 - Auxiliary Design Variables

5.1.2.2 Global Variables

Another set of variables, from the 34, can be characterized as more global variables. These variables can be seen in Table 19 and most of them fit in the definition of the "Particulars" given in section 2.2.1.1, which as said before describe gross characteristics of the ship. The changes induced to the hull geometry from these variables are more general and in a more global level. Most of them, control main dimensions of the hull or main characteristics of the hull-form, like the entrance angle and the submerged part of the transom.

No	Variable Name	Path in Object Tree
1	*Cwp*	|Variables
2	*DWLaftAngle*	|Variables
3	*Draft*	|Variables
4	*EntranceAngle*	|Variables
5	*H_BeamTransomOverH_Beam*	|Variables
6	*KeelRisePoint*	|Variables
7	*LBP*	|Variables
8	*L_Over_Beam*	|Variables
9	*TransomDepth*	|Variables
10	*StemAngle*	|Variables
11	*StemRadiusOverDraft*	|Variables
12	*TransomAngle*	|Variables

Table 19 - Global Design Variables

5.1.2.3 Local Variables

The final set of variables that can be seen in Table 20, are characterized as local variables. These variables induce changes in the hull-form in a much more local level than the global variables. In some cases, small changes of these variables might induce so small local changes that are hard to notice if not carefully observe the lines of the hull (body-plan and/or buttocks). Apart from the longitudinal position of maximum breadth and the tangent angle of the keel at the connection with the transom, all the other variables, defining the "control curves", are not commonly used in the traditional description of hull-forms. For this reason, the user should be very diligent on understanding what these variables describe and how they affect the geometry, by consulting the geometry definition in section 2.4.

No	Variable Name	Path in Object Tree
1	*Cr_Section_Change*	|Variables
2	*KeelAngleAtTransom*	|Variables
3	*x_Bmax*	|Variables
	Deadrise Anle	
4	*0_05*	|Variables|Deadrise
5	*0_15*	|Variables|Deadrise
6	*0_25*	|Variables|Deadrise
7	*0_50*	|Variables|Deadrise
	Flare Angle	
8	*0_05*	|Variables|Flare
9	*0_25*	|Variables|Flare
10	*0_50*	|Variables|Flare
11	*0_75*	|Variables|Flare
12	*1_00*	|Variables|Flare
	Fullness Factor	
13	*0_00*	|Variables|FullnessFactor
14	*0_25*	|Variables|FullnessFactor
15	*0_50*	|Variables|FullnessFactor
16	*0_75*	|Variables|FullnessFactor
17	*1_00*	|Variables|FullnessFactor

Table 20 - Local Design Variables

5.1.2.4 Discussion

The variables used to describe the hull's geometry were broken into sets, depending on the magnitude of their effects over the geometry and hydrodynamic resistance of the ship. This separation of the variables allows the user to better understand how the variables affect the end geometry of the generated hull and thus easily decide on which variables to use and how in any optimization process.

Even though, there is a set of variables described as "Local Variables", the user should have in mind that if all these variables were altered in a considerable range of +/10% or +/-20%, the end result of the geometry would be a totally different hull. On the other hand, if one of these variables is slightly altered (+/- 5%), then the end hull will not be the same of course, but not drastically different from the original.

Finally, a completed set of 34 variables allows for the generation of a unique hull for most of the times. This would have been very difficult to accomplish, if the selected variables were in the "Particulars" (see section 2.2.1.1) level only, which are gross characteristics and parameters of a ship. Having a unique hull for each set of variables is very important for an optimization algorithm, as it allows it to understand towards where it should move in order to accomplish the requested goal. At the same time, having only variables with local effect on the geometry might have overcomplicated the model and also make difficult for the user to understand how these variables are affecting the total geometry.

5.1.3. Objective Function

As discussed before the goal of the optimization would be to minimize the hydrodynamic resistance calculated by the potential flow panel method or the Holtrop method, both described in chapter 4. The calculation of the resistance is carried out for a specific speed that the user selects by using the variable *"Speed_Kn"* in the *"Optimization"* subfolder of the "Variables" folder inside the object tree. The user should be very careful to select a speed high enough, in which the dominant part of the hydrodynamic resistance comes from the generation of waves, due to the movement of the ship.

By using the Holtrop method described in section 4.2.1, we can calculate the components of the resistance for the modeled DDG-51 hull of section 2.6. In Figure 43, we see that the wave making resistance is the dominant component for speeds greater than 22 knots. Thus, it makes sense to optimize the hull-form based on the wave-making resistance for speeds greater than this, when the panel method is used. For speeds, less than 22 knots the friction resistance is more dominant and viscous effects are more important. For this speed regime a different method, like a full viscous solver, should be used to calculate the resistance. Having this in mind, the user can select to optimize the hull by using either the Holtrop or the panel method.

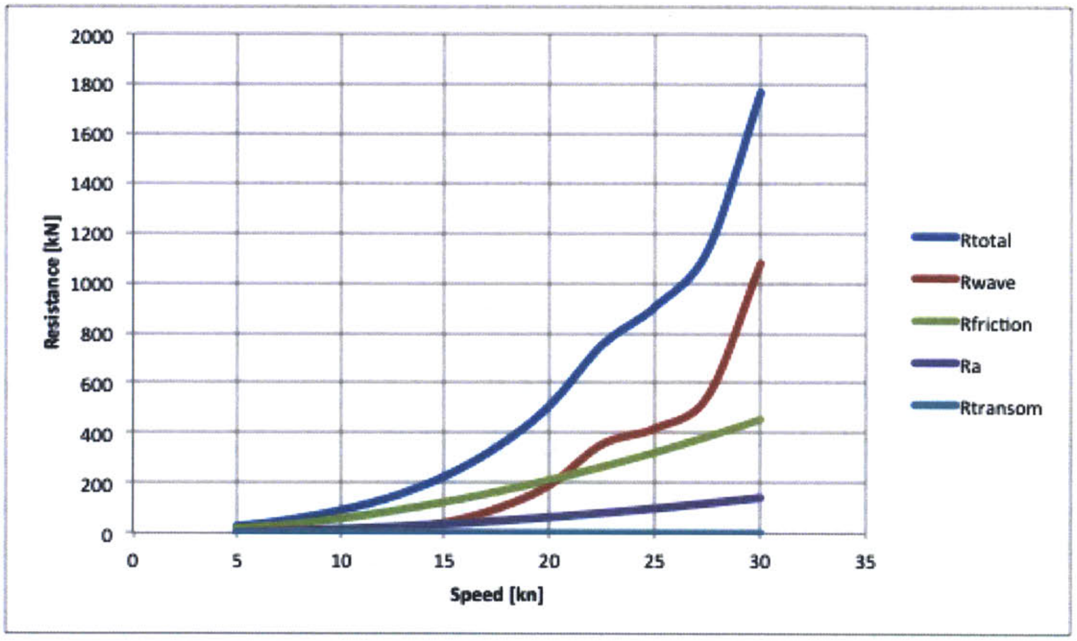


Figure 43 - Modeled DDG-51 Components of Resistance Based on the Holtrop and Mennen Method

5.1.3.1 Panel Method

For each variant, the panel method code calculates the wave-making resistance coefficient (C_W), which is a non-dimensional number. From this, the wave-making resistance (R_W) is calculated by using the wetted surface of the ship, calculated automatically by the hydrostatic computations of Friendship Framework®. Consequently, the frictional resistance (R_f) is calculated by using the ITTC-57 model ship correlation line (Larson & Raven, 2010).

$$R_W = \frac{1}{2}\rho U^2 S_{Wetted} C_W$$

$$Re = \frac{UL}{\nu} \qquad C_f = \frac{0.075}{(logRe-2)^2} \qquad R_f = \frac{1}{2}\rho U^2 S_{Wetted} C_f$$

$$R_t = R_W + R_f$$

The resistance designated as total resistance (R_t) in the above equation is not the real total resistance of the ship, as it misses the viscous pressure resistance component, which can't be calculated by the potential flow theory. For the scope of this tool, however, this can be considered the total resistance, as it is the best resistance prediction that can be done and is entirely connected with the geometry of the hull.

5.1.3.2 Holtrop Method

When the Holtrop method is used the process is easier, as by definition the Scilab® function calculates all the components of the resistance, as it was described in section 4.2.2. For our application, we will use the total resistance of the ship that is calculated in Holtrop's method (1984). All of the needed input and output information is automatically stored by the tool in the input and output Friendship Framework® parameters, located in the "*/Computations/Holtrop*" subfolder of the object tree.

5.1.3.3 Non-dimensional Resistance

Instead of using the total resistance as the objective function of the optimization, a non-dimensional version of it is utilized. The displacement (Δ) of the ship and the gravitational acceleration (g) are used to remove the dimensions from the total resistance. By doing this, we manage to remove the displacement as the primary parameter controlling the resistance. In most cases, the resistance of a displacement hull is highly and directly correlated with the displacement, thus smaller and lighter ships will most probably oppose less resistance when compared with larger and heavier ships.

$$\tilde{R}_{t=\frac{R_t}{\Delta \cdot g}}$$

For this reason, an optimizer seeking to find the smallest resistance will most probably try to minimize the displacement. However, this does not mean that the final ship has better hydrodynamic characteristics, but rather is smaller and it can move easier through the water. By using the non-dimensional version of the resistance, we change slightly the objective, so that we seek the ship that hauls the most weight and opposes the least resistance. In other words, if two ships have the same resistance at a given speed, we would like to choose the ship with the larger displacement, as by that we "pay" the same to transport more weight (and volume) through the water. The non-dimensional form of the resistance is automatically calculated for both the Holtrop and panel method and is then used in the two optimization function pre-setup in the tool.

5.1.4. Constraints

There are three sets of constraints utilized in the optimization process that are used in this tool. First, we have the geometric constraints that are imposed from the objects defining the internal arrangements of the hull. Second, there are constraint imposed directly by the user for the displacement and longitudinal location of the center of buoyancy (LCB). Finally, we have some quality constraints used by default and always, in order to ensure the goodness of the generated hulls.

5.1.4.1 Geometric Constraints

As discussed in section 3.2 the bulk equipment of the ship can be modeled in the form of objects, represented by boxes. Once the user activates these boxes by using the "*InternalArrangements*" subfolder in the "*Variables*" folder of the object tree, they can serve as geometric constraints for an optimization process. As shown in Figure 37, the tool performs automatically the check on whether the lower and exterior edges of each box interfere with the hull or not.

The *"Hardpoint"* function of Friendship Framework®, along with the logic *"IF"* and *"AND"* functions are used to monitor the intereference of a box with the hull. This is done inside the *"Internal_Arrangements"*[3] folder of the project, located outside of the *"Variables"* folder. Inside the dedicated folder for each object, there is a parameter named *"Object_OK"*. If the user does not use the object, then this parameter takes always the value one, which means that the object is clear from the hull. When, the object is actually activated by the user, an *"AND"* function is used and the parameter takes the value one only when both edges of the object are clear of the hull. The logic diagram for the check performed about the interference of the object with the hull can be seen in Figure 44.

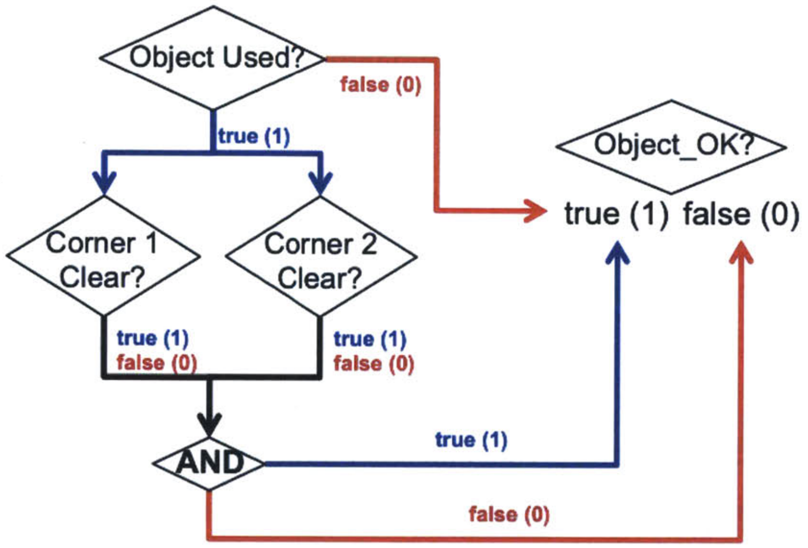


Figure 44 - Logic Diagram of Object Interfering with the Hull

Once the check has been performed for every object, the ones or zeroes from their *"Object_OK"* parameter are fed into the *"CHECK_Objects_OK"* parameter, which serves as an *"AND"* function. In there, the corresponding values from the objects are multiplied with each other and the value one is issued only when all of the used objects are clear of the hull and/or not used. Consequently, this parameter is the one monitored by the *"MachineryCheck"* constraint function of Friendship Framework®, which is located in the special *"Constraints"* part of the object tree. This constraint function is then used in the

[3] Notice the slight difference in name

optimization process, making sure that all the feasible designs fit the arranged equipment (objects).

5.1.4.2 Constraints Imposed by User

Apart from the geometric constraints coming from the presence of the modeled equipment, the user is able to set limits for the displacement and LCB of the feasible hulls. A high and low value can be set for both of these parameters, by using the related variables summarized in Table 21. Once these values are set, they are fed into the corresponding constraint functions of the software.

Constraint	Variable	Path in Object Tree	Constraint Function
High Displacement	DispHigh	|Variables ↳|Optimization ↳|Constraint	DisplacementHigh
Low Displacement	DispLow		DisplacementLow
High LCB	LCBHigh		LCB_High
Low LCB	LCBLow		LCB_Low

Table 21 - Summary of User Specified Constraints

The optimization algorithms then use these constraints to identify which of the designs are feasible and adapt accordingly. In case the user does not want to impose any constraints on these two parameters, he/she should assign values high and low enough, such that the constraints become ineffective. This is very important, as penalties, based on the state of the constraint functions, are fed into the objective functions of the optimization. Even when the optimization algorithm does not consider a constraint, the penalties associated with it are added into the objective function. For this, in the case where the constraints were just removed in the optimization functions, the algorithms would see feasible designs, having extremely high non-dimensional resistance, due to the penalties opposed in the object functions. This would most probably cause problems in the algorithms, as they would not be able to have a clear view of the trade space.

5.1.4.3 "Quality" Constraints

Finally, a set of "quality" constraints exists in the tool, which are utilized to control and ensure the "goodness" of the generated hulls. These constraints are checks done on the shape of the DWL and the fullness factor control curve described at section 2.4.4.3.

(1) Fullness Factor

A check is done on the fullness factor control curve in order to ensure that this parameter does not acquire negative values though out the length of the ship. The points controlling this curve might take values between 0 and 2, while the curve is defined for a much larger interval. For this reason, it is possible a combination of the points to exist, at which a part of the curve will have negative ordinate. In case, this happens, the NURBS cross-section becomes concave and thus it fails to represent correctly the geometry observed in mono-hulls, as it can be seen in Figure 45. The constraint function used to perform this check is the "*FullnessCheck*", located in the "*Constraints*" part of the object tree.

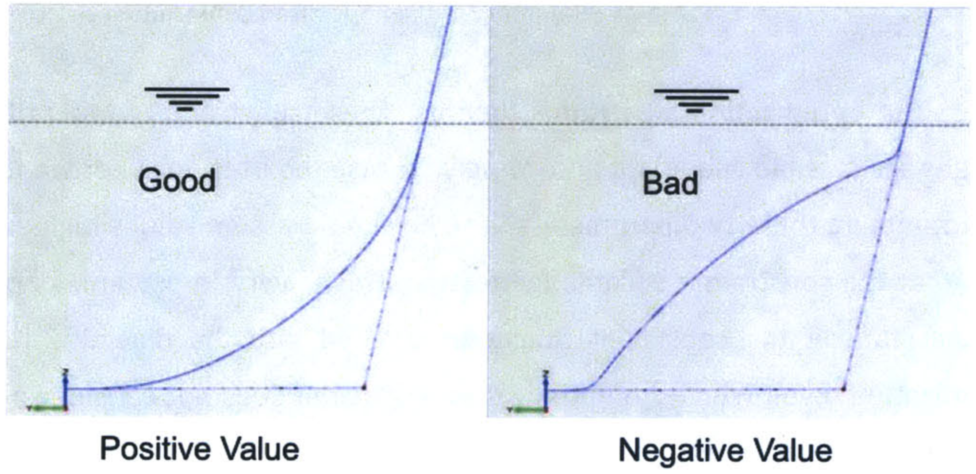


Figure 45 - Comparison of Sections with Positive and Negative Fullness Factor

(2) Design Waterline

Lastly, a quality constraint exists for the shape of the DWL ensuring that its shape is favorable. As the optimization algorithm alters the design variables, there are some combinations that can lead to a bizarre shape of the DWL. These designs might respect the rest of the constraints and thus the panel or the Holtrop method will calculate their resistance. The quality constraint for the DWL, however, eliminates these designs and thus saves computational power and time. The constraint function, used to perform this task, is named "*DWLCheck*" and can be found in the constraints section of the object tree.

For every design, the maximum ordinate of the DWL is monitored and compared with the value of the maximum breadth that the design should have had, based on the "*LBP*" and "*L_Over_Beam*" variables that the user controls. If the measured maximum value is more than 1% of the desired maximum breadth, then the design is considered as unfeasible by the algorithm. Figure 46 shows an example where the C_{WP} is so high that a problematic DWL is generated. This variant would not pass the DWL constraint and thus will be characterized as an unfeasible in the optimization.

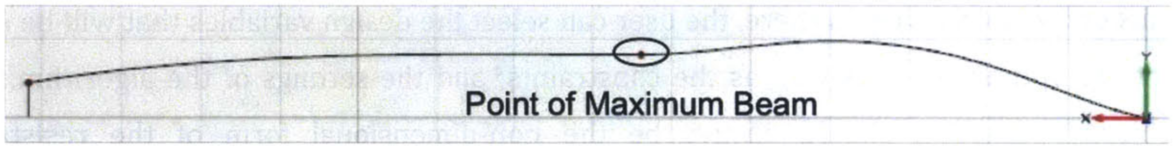


Figure 46 - Example of Problematic DWL

5.2. Optimization Algorithm

Friendship Framework® comes with a set of optimization algorithms and design engines. One of these algorithms is the Non-dominated Sorting Genetic Algorithm 2 (NSGA-II), which is an evolutionary genetic algorithm (Deb, et al., 2002). The non-dominating sorting technic allows for good diversity of the results in the optimization process, which improves the search of the trade-space and ultimately the probability to find the theoretical minimum solution. The NSGA-II is a multi-objective optimization algorithm, where the optimization process is done based on a global convergence of the studied individuals towards a minimum value of the objective function. This type of algorithms does not require calculating the gradient (Jacobian) of the objective function(s) and thus it is harder for them to be trapped in a local minimum.

The main reason for using this native algorithm was the nature of the problem. The resistance of a ship and its correlation to the specified design parameters is a very difficult and complex system that needs to be optimized. For this reason a heuristic approach like the one offered by a Genetic Algorithm (GA) was deemed to be the most suitable choice. Further, the intelligent strategy utilized by an algorithm like NSGA-II, during the exploration of the trade space, allows for better management of the computational resources, as it is faster and more efficient on minimizing the objective function(s).

The user can access the NSGA-II optimization functions of Friendship Framework® with the name "*PanelOpt*" and "*HoltropOpt*", through the "*Design Engines*" part of the object tree. In there, the user can select the design variables that will be used for the optimization, as well as the constraints[4] and the settings of the algorithm. The objective function should always be the non-dimensional form of the resistance calculated by the panel or Holtrop method, which is already pre-selected in the aforementioned optimization functions. Also, the quality constraints, concerning the shape of the DWL and the fullness factor control curve, should never be removed from the optimization process.

[4] The user should remember that if he/she wants to remove a constraint should do it in the way described at sections 5.1.4 and 5.2.2.

5.2.1. Design Variables

The user has the ability to set up each time his/her own optimization problem, by selecting which variables to consider as design variables and what should be the appropriate upper and lower bounds. Because of this, the tool provides high flexibility for the user to explore a different problem each time. He/she can choose to apply either local changes to parts of the hull, like the bow, stern or mid-ship area, or global changes in the "Particulars'" level or a combination of local and global changes. Each time, the optimization problem can be fitted into the user's needs, by selecting the needed set of design variables.

Even though, the tool allows for such flexibility, the user should always keep in mind that as complexity and number of variables increase, the needed computations and variants studied, during the optimization process, increase as well. Thus, the user should consider very carefully which variables can be fixed and what the bounds of the design variables should be. A safe approach to this problem is to use as a starting point the values of a known design. The optimization can be conducted on bounds around the values defining a specific ship. If in the end of the process the optimization converges at a design, which has reached the maximum or minimum bounds for some design variables, the bounds can be accordingly altered and a new optimization could be launched.

5.2.2. Constraints

Theoretically, the user can control which constraints the optimization function should take into consideration or not, by using the corresponding field at the "*Object Editor*". However, the tool was built, so that it always takes into consideration the state of the constraints, even when are not used by the optimization function. This was obligatory, as in the 2.4.8 version of the software; the objective function is evaluated even for the unfeasible designs. In order, to stop this waste of resources, a check is performed on the state of the constraints and only when all of them are met, the non-dimensional resistance (objective function) is evaluated.

In case the user does not consider a constraint in the optimization function of the software, the model is built to always consider it and will not calculate the resistance of the variant not meeting the constraint. Instead, it will automatically assign a very high value to the objective function, in the form of a penalty. Thus, all the constraints should be activated at all times in the optimization function. If the user does not want to use any of them, then a very large or small monitor value should be given to the constraints, as it was suggested in section 5.1.4.2. By this, the constraint becomes ineffective to the optimization procedure, as it will be always met.

5.2.3. Optimization Settings

There are four setting that the user should choose for every optimization case or problem. As in all Genetic Algorithms (GA), the number of the "Generations" and the number of the "Population Size" per generation should be selected. These two numbers are highly correlated on the number of the design variables, as well as their effect on the objective function. A simple rule of thumb is that both generation and population size, should increase when the design variables and the mapping of each combination of variables to the objective function becomes ambiguous and complicated.

Ideally, we would like each generation to have a population of 2^N (N number of design variables), which would allow for a full factorial exploration of the trade-space at each generation. This, however, would lead to lots and lots of computations and time, which can be avoided when algorithms like NSGA-II are used. These algorithms will never establish a perfect picture of the trade-space but will try to look at most of its areas, in order to find the optimal solution. NSGA-II introduces the needed diversity in the points of the trade-space that looks into, either by its characteristic non-dominated sorting of the individuals, or by using the traditional in GAs crossover and mutation probabilities.

The user can control both of these probabilities and their main purpose is to introduce the needed diversity to the variants generated during the process of the optimization. If this does not happen, then the algorithm might get "trapped" around a specific point, around which it finds good enough results and miss the even better values attained at a neighboring point or area. As their names imply, both of the probabilities are inspired by the genetics science and introduce diversity through two different mechanisms.

The crossover probability tries to mimic the biological recombination of chromosomes (Mitchell, 1998). This probability expresses the likelihood that part of the parent's chromosome will pass down to the offspring. If this probability is set to zero, then the chromosomes of the parents will just be passed down to the next generation, without being combined or mixed with the chromosomes of another parent. In other words, a zero crossover probability would mean that its generation is a clone of its immediate ancestral generation.

On the other hand, the mechanism of the mutation probability is more chaotic and introduces random changes in the genes (or bits) of the offspring chromosome. The higher this probability is, the higher the chance for part of the chromosome to be randomly changed. This mechanism takes place after the creation of the offspring chromosome by combining parts of parental chromosomes. Continuing on the extreme example of the zero percent crossover probability, if the mutation probability is also zero, then the future generation will be clones and identical of the first generation.

Thus, it is evident that ideally, we would like to have both probabilities high enough, in order to ensure that our algorithm will indeed explore a wide enough space. This, however, can't happen if the algorithm is limited to create relatively small generation, as it will leave the algorithm clueless on towards where to move in order to discover the optimal solution. For this reason, we need to establish a balance between the two aforementioned probabilities and the size of the population per generation.

All these settings should be selected by the user with care and after considering and examining the problem set forth to the optimization algorithm. For this reason, no specific recommendations on these values can be provided to the user, apart from the fact that always have to have in mind the fine balance needed between diversity, number of individuals and generations. The latter are the aspects that give to the algorithm the power to search and seek for the best optimal solution, but at the same time, are the ones that should be limited due to physical constraints in computational power and time.

5.3. Case Studies

5.3.1. Problem Formulation

Two different cases were studied in order to demonstrate the capabilities of the tool. First, a hull-form optimization is conducted without any constraints from internal arrangements and both resistance estimation methods are used. In the second case, internal arrangements of machinery equipment are modeled and the optimization is repeated, by considering the geometric constraints imposed by these arrangements. It is assumed that the engine layout of DDG-51 Flight I will be used for the internal arrangements.

Information about the machinery layout of DDG-51 Flight I was taken from ASSET version 5.3. The equipment was first modeled in Rhino®, in order to ease the process of calculating the needed parameters for each equipment box to be defined in the model. As explained in section 3.3 the equipment located in the port (left) side of the ship was modeled as if it was in the starboard (right) side of the ship. Also for each engine assembly, the required side, top and bottom clearances reported by ASSET v 5.3 were taken into consideration.

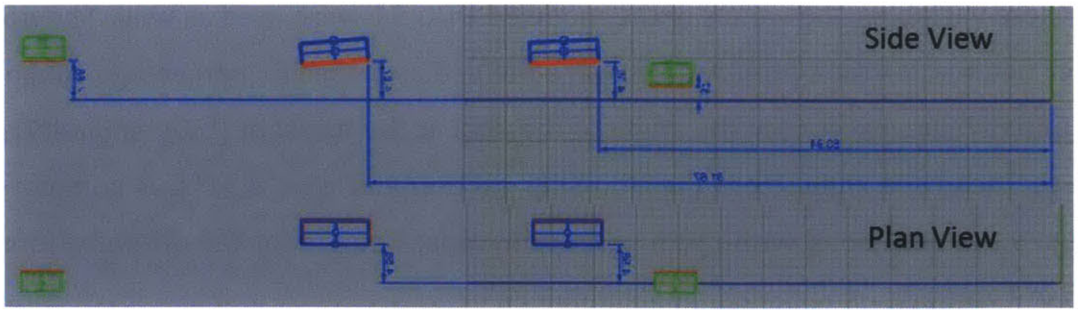


Figure 47 - Rhino Modeling of Internal Arrangements

From the total of 34 variables needed to completely define a hull in the model, 13 were considered as fixed, while the rest 21 were considered as design variables of the optimizations. The values selected, for the fixed variables, were based on some hypothetical values that the design should have. For instance a specific LBP, draft, transom angle and depth were selected to be the same for all hulls. For the variables that a similar estimation could not be done, the value used when modeling DDG-51 was used, as they were reported in Table 15.

Fixed Variables		
Variable Name	**Value**	**Comment**
Draft	6.17	
Freeboard_FP	8	
Freeboard_MS	5.2	
LBP	140	
StemAngle	63	DDG-51
StemRadiusOverDraft	1.5	DDG-51
TransomAngle	17	
TransomDepth	1	
OffsetAngleFWD	34	DDG-51
OffsetAngleMidShips	10.2	DDG-51
OffsetAngleTransom	20	DDG-51
Fullness Factor0_00	1	DDG-51

Table 22 - Summary of Fixed Variables Used in Optimizations

For some of the design variables arbitrary initial values were selected, apart from the variables controlling the fullness factor, deadrise angle and flare angle control curves.

For these variables, the values of the modeled DDG-51 were used as well. The upper and lower bounds for the optimizations were defined as a specific percentage range around the initial value, except for the angle of the keel at the transom (*"KeelAngleAtTransom"*) and the deadrise angle from the mid-ship section and aft (*"0_50"*). A summary of the design variables, along with their lower and upper bounds for the optimizations can be seen in Table 23.

Design Variables					
Variable Name	**Value**	**Comment**	**Bounds**		**+/- %**
			Lower	**Upper**	
Cwp	0.818		0.777	0.859	5.00%
DWLaftAngle	10.00		8.00	12.00	20.00%
EntranceAngle	12.00		9.60	14.40	20.00%
H_BeamTransomOverH_Beam	0.60		0.54	0.66	10.00%
KeelAngleAtTransom	2.00		0.00	5.00	-
KeelRisePoint	0.65		0.59	0.72	10.00%
L_Over_Beam	8.60		8.17	9.03	5.00%
x_Bmax	0.50		0.45	0.55	10.00%
Deadrise Anle					
0_05	60.00	DDG-51	54.00	66.00	10.00%
0_15	30.00	DDG-51	27.00	33.00	10.00%
0_25	9.00	DDG-51	8.10	9.90	10.00%
0_50	1.50	DDG-51	0.50	1.65	-
Flare Angle					
0_05	18.00	DDG-51	16.20	19.80	10.00%
0_25	22.00	DDG-51	19.80	24.20	10.00%
0_50	8.00	DDG-51	7.20	8.80	10.00%
0_75	13.00	DDG-51	11.70	14.30	10.00%
1_00	35.00	DDG-51	31.50	38.50	10.00%
Fullness Factor					
0_25	0.60	DDG-51	0.48	0.72	20.00%
0_50	1.00	DDG-51	0.80	1.20	20.00%
0_75	0.40	DDG-51	0.32	0.48	20.00%
1_00	1.20	DDG-51	0.96	1.44	20.00%

Table 23 - Summary of Design Variables Used in Optimizations

Finally for all optimizations, the same displacement and LCB constraints were imposed. Apart from these user-controlled constraints, all the "Quality" constraints, described in section 5.1.4.3 were considered. Moreover the same number of total individuals and crossover and mutation probabilities were used for all optimizations, so that a fair comparison between the two different cases and resistance estimation methods can be attained. The total number of individuals was kept low; however, better results might be realized if more generations and higher population per generation should have been used. A summary of the user constraints used and the settings of the optimization algorithms can be seen in Table 24.

Constraints	
Displacement High [mtons]	8,300
Displacement Low [mtons]	7,700
LCB High [meters]	75
LCB Low [meters]	70
Algorithm Settings	
Generations	60
Population per Generation	64
Total number of individuals	3,840
Crossover Probability	0.75
Mutation Probability	0.10

Table 24 – Constraints and Settings of Optimization Algorithm

5.3.2. Without Internal Arrangements

Both optimizations conducted, in the case with no internal arrangements (free), were successful and the convergence of the algorithm can be seen in the next sections. Clear pareto frontiers were formed for both optimizations, which indicate a good approach of the algorithm to the problem. The non-dimensional resistance, displacement and LCB for the baseline and optimal hulls are summarized in Table 25. The non-dimensional resistance was calculated for the baseline and optimal hulls by using both the Holtrop and the panel method. However, only one of the two was each time used as the objective function of the optimization. From the results, it is observed that with both methods (Holtrop and panel), improvements in excess of 5% are realized for the resistance of the hulls.

	Baseline	Holtrop Optimal	Δ %	Panel Optimal	Δ %
Resistance Holtrop [Dmnlss]	0.01466051	**0.013887852**	-5.27%	0.014966964	2.09%
Resistance Panel [Dmnlss]	0.012177773	0.012675193	4.08%	**0.011418154**	-6.24%
LCB [meters]	71.528	72.723566	1.67%	74.136777	3.65%
Displacement [mtons]	7715.62	7722.9034	0.09%	7747.8556	0.42%

Table 25 - Summary of Optimal Hulls vs Basline for the Free Optimization

Interestingly, the two methods seem to not agree when the resistance of the optimal hull is calculated with the method not used as an objective function. For instance, when the Holtrop method was used, the resistance reduction for the optimal hull was 5.27%. But, when the resistance is calculated with the panel method for both the baseline and the Holtrop optimized hull, we observe that the resistance of the optimal hull is higher (4.08%) than the baseline, indicating that the optimal hull is hydrodynamically worse than the baseline. The same observation is made when the optimization is based on the resistance estimation of the panel method and the resistance of the optimal hull is then estimated by using Holtrop (2.09% increase).

This very interesting outcome clearly indicates the difference of the two methods, especially if the displacement and LCB of the optimal hulls are taken into consideration as well. When the optimization is based on the Holtrop method, we can observe from Table 25 that the algorithm tries to minimize the displacement. The Holtrop method uses parameters in the "Particulars" level in order to estimate the resistance of the ship and the displaced volume is one of them. It seems that the correlation of the displaced volume with the resistance is very strong that forces the algorithm to adapt a strategy of minimizing the displacement. On the other hand, the panel method does not really try to reduce the displacement, but rather move the volume distribution of the hull towards the stern. This is done, as a finer bow will produce smaller waves and thus reduce the resistance of the hull.

Table 26 has a comparative summary of the design variables of the optimal hulls against the baseline. We can't really distinguish a common trend on how the algorithm moved through the optimization when the two different resistance estimation methods were used. However, we can see that for both cases, some variables have closely approached their minimum or maximum bounds (in red), as they were reported in Table 23. For this reason, it would be a good practice to start a new optimization by allowing for larger variations in these variables, or use the optimal hull for the new baseline and re-define the new bounds based on it.

	Baseline	**Holtrop Optimal**	**Difference**	**Panel Optimal**	**Difference**
Cwp	**0.818**	0.81712226	-0.11%	0.77720771	-4.99%
DWLaftAngle	**10**	8.8935073	-11.06%	8.0756237	-19.24%
EntranceAngle	**12**	9.9206592	-17.33%	10.117317	-15.69%
H_BeamTransomOverH_Beam	**0.6**	0.5856434	-2.39%	0.65581964	9.30%
KeelAngleAtTransom	**2**	4.4651713	123.26%	0.16983291	-91.51%
KeelRisePoint	**0.65**	0.70906683	9.09%	0.69626818	7.12%
x_Bmax	**0.5**	0.46672541	-6.65%	0.50123217	0.25%
Deadrise Angle					
0_05	**60**	60.540993	0.90%	64.105196	6.84%
0_15	**30**	28.412497	-5.29%	32.193042	7.31%
0_25	**9**	9.208098	2.31%	9.774342	8.60%
0_50	**1.5**	1.0224704	-31.84%	1.2484344	-16.77%
Flare Angle					
0_05	**18**	16.973559	-5.70%	16.233948	-9.81%
0_25	**22**	21.235648	-3.47%	21.528646	-2.14%
0_50	**8**	7.8190524	-2.26%	7.380032	-7.75%
0_75	**13**	12.850689	-1.15%	13.83102	6.39%
1_00	**35**	34.56511	-1.24%	31.826741	-9.07%
FullnessFactor					
0_25	**0.6**	0.57242573	-4.60%	0.48413825	-19.31%
0_50	**1**	1.1214771	12.15%	0.98050813	-1.95%
0_75	**0.4**	0.47447501	18.62%	0.33981476	-15.05%
1_00	**1.2**	1.4137935	17.82%	1.1706184	-2.45%
L_Over_Beam	**8.6**	9.0143445	4.82%	8.1780311	-4.91%

Table 26 - Comparison of Design Variables between the Optimal Hulls and Baseline

5.3.2.1 Holtrop Optimized Hull

The history of the optimization process can be seen in Figure 48. The algorithm managed to establish a pareto frontier and to follow it to a minimum value of the objective function. The diversity of the evaluated designs is large enough and shows that the algorithm made a fairly good search of the trade space. Finally, we can observe that at a point, around individual #1500, there is a step in the values of the objective function, indicating that the algorithm found some favorable combination of the design variables and then followed this path and managed to minimize the resistance even more.

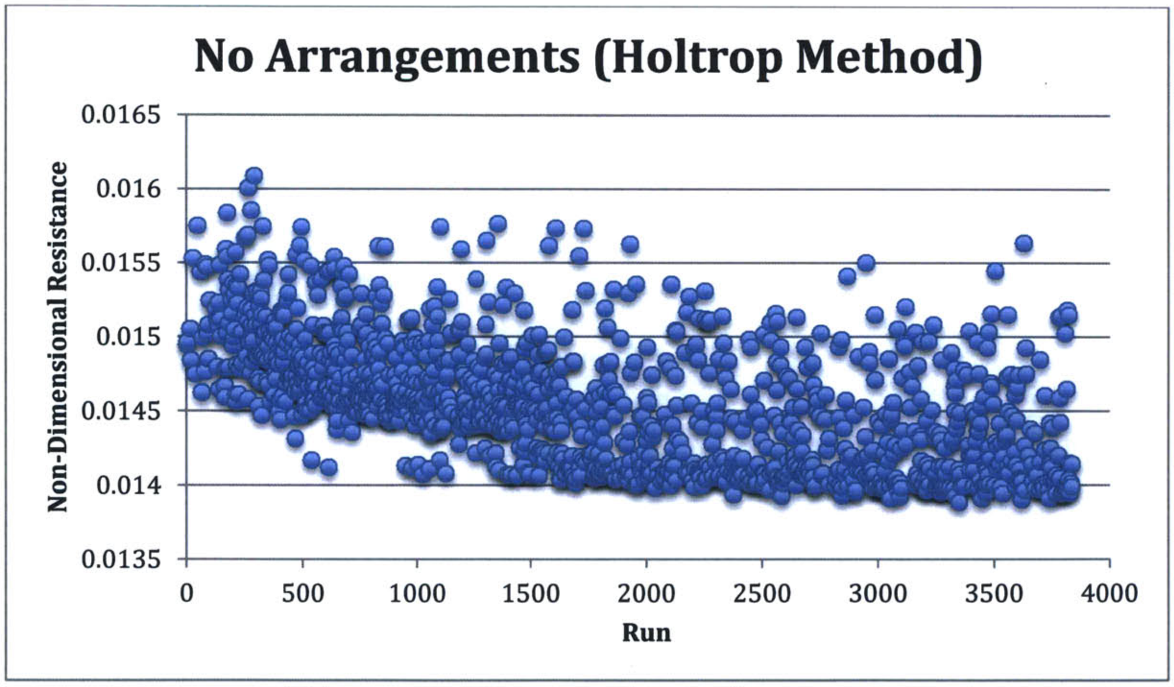


Figure 48 - History of Optimization Process

The body-plan and side view of the optimized hull, along with the Sectional Area Curve (SAC) can be seen in Figure 49 and Figure 50 respectively. From the body-plan and the data in Table 26, we can observe that the hull is more slender than the baseline. The sections are less full, but the part where the keel rises is fuller in order to balance the loss of displacement in the rest of the ship. Also, by observation of the SAC, we see that there is a slightly abrupt change in her curvature at the point where the keel starts to rise, which it might indicate a not very well faired surface at this area. Finally in Figure 51, there is a comparison of the wave field for the baseline and the Holtrop optimized hull, generated by the panel method. The only difference observed is for the bow wave, which is slightly larger for the optimized hull, causing the increased resistance indicated by the panel method.

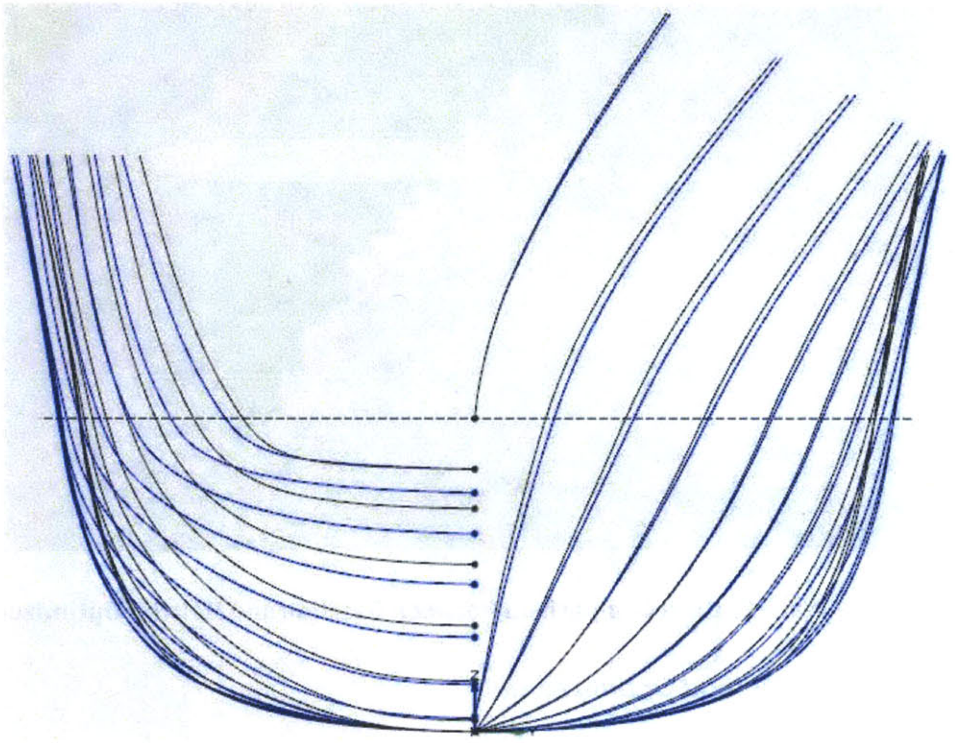


Figure 49 – Body-Plan Comparison between Baseline (blue) and Holtrop Optimized Hull (black)

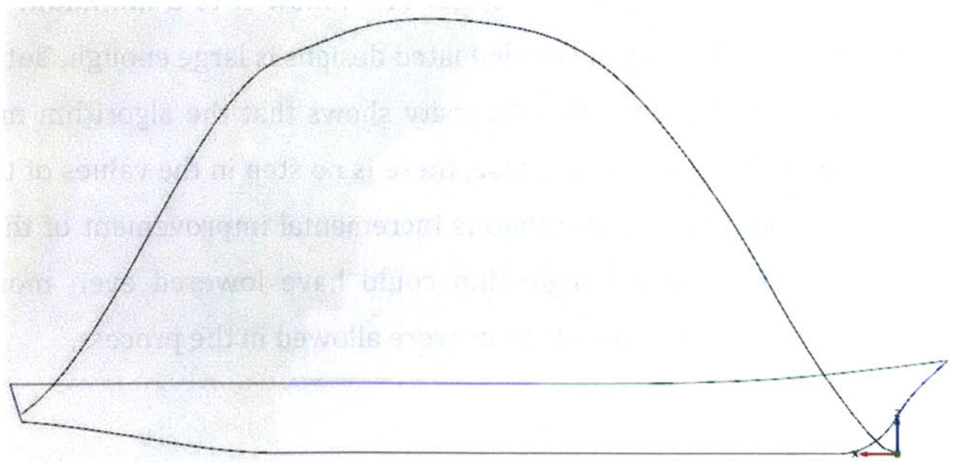


Figure 50 - Sectional Area Curve and Side View of Holtrop Optimized Hull

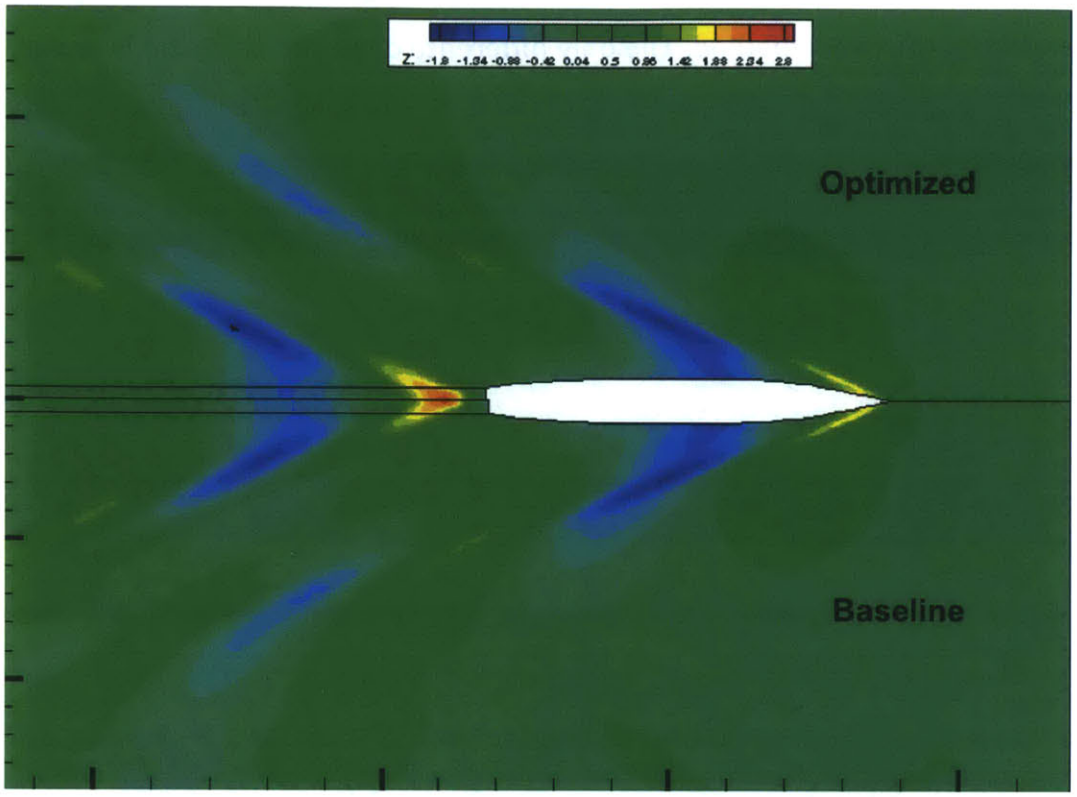


Figure 51 - Wave Pattern Comparison Between Baseline and Holtrop Optimized Hull

5.3.2.2 Panel Method Optimized Hull

The history of the optimization process can be seen in Figure 52. The algorithm, again, managed to establish a pareto frontier and follow it to a minimum value of the objective function. The diversity of the evaluated designs is large enough, but not as large as in the Holtrop case. However, this diversity shows that the algorithm made a fairly good search of the trade space. In this case, there is no step in the values of the objective function, as before, but rather a continuous incremental improvement of the generated hulls. Finally, it seems that the algorithm could have lowered even more the non-dimensional resistance, if more generations were allowed in the process.

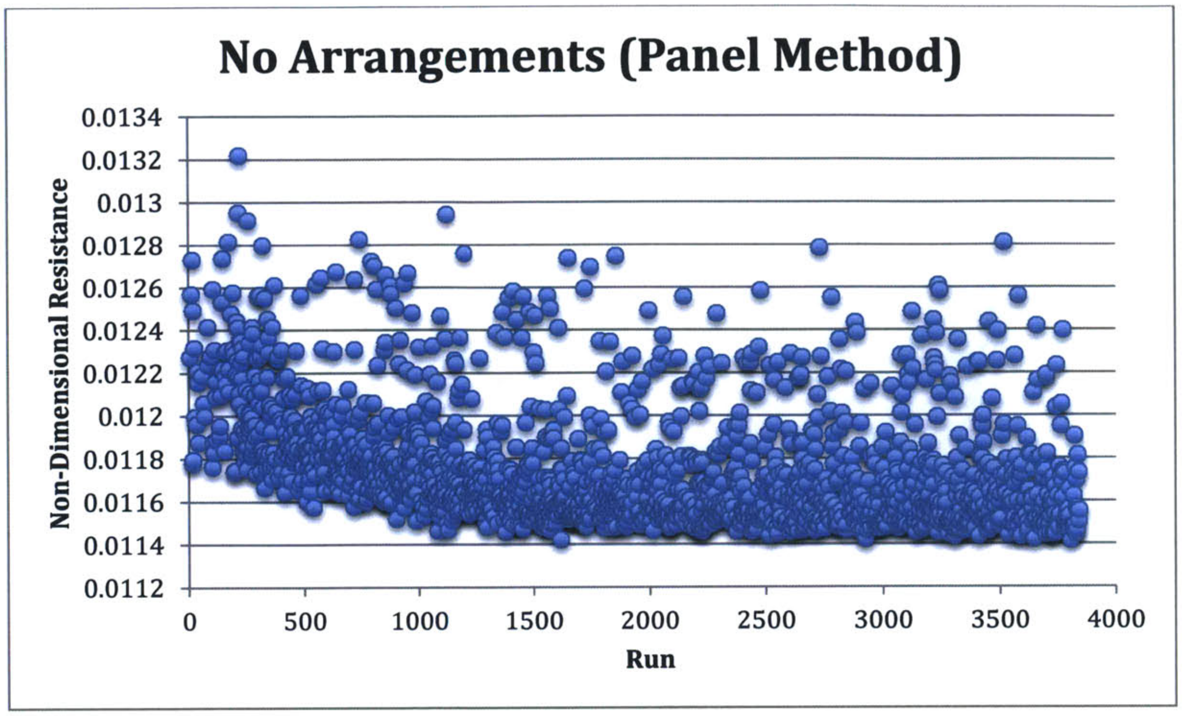


Figure 52 - History of Optimization Process

The body-plan and side view of the optimized hull, along with the Sectional Area Curve (SAC) can be seen in Figure 53 and Figure 54 respectively. From the body-plan and the data in Table 26, we can observe that this time the optimized hull is beamier than the baseline. The sections are considerable less full in the bow area and most of the hull's displacement has been moved towards the stern. Also, now we see that the part of the keel rising is smoother and not so full as in the Holtrop optimization.

The keel has almost a zero angle at her aft end, where it meets the transom and this aspect possibly comes from the way the panel method treats the transom stern, as described in section 4.3.1. The SAC again has a small hump in the area where the keel starts to rise, but it is believed that any issues because of this would be small and failry easy to be fixed. Finally in Figure 55, there is a comparison of the wave field for the baseline and the panel optimized hull, where we can see clearly that the optimized hull has a smaller bow wave than the baseline.

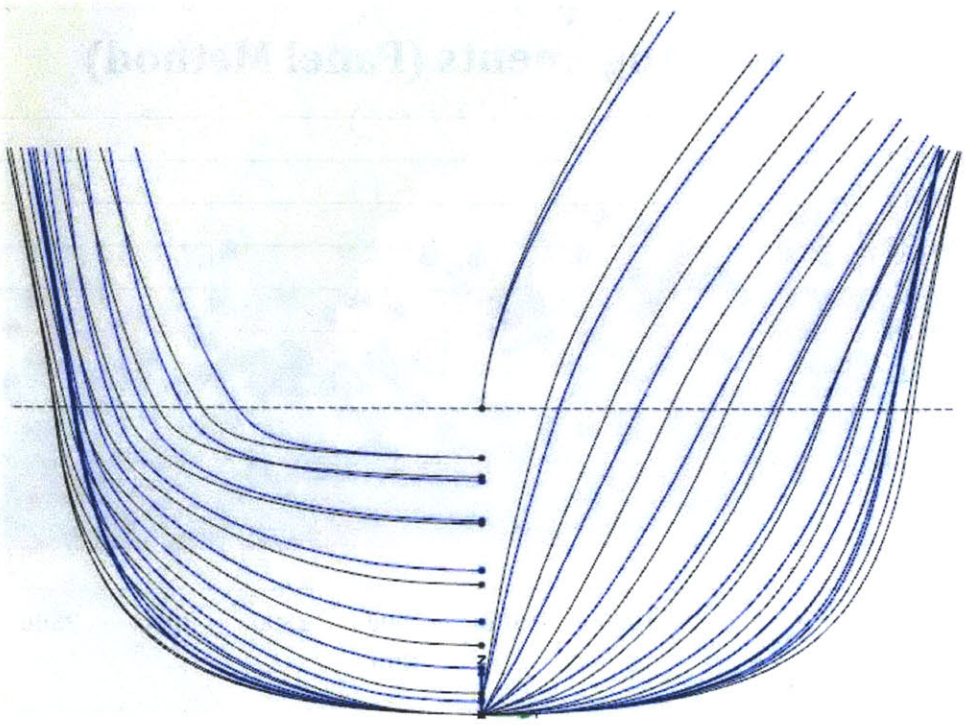


Figure 53 - Body-Plan Comparison between Baseline (blue) and Panel-Optimized Hull (black)

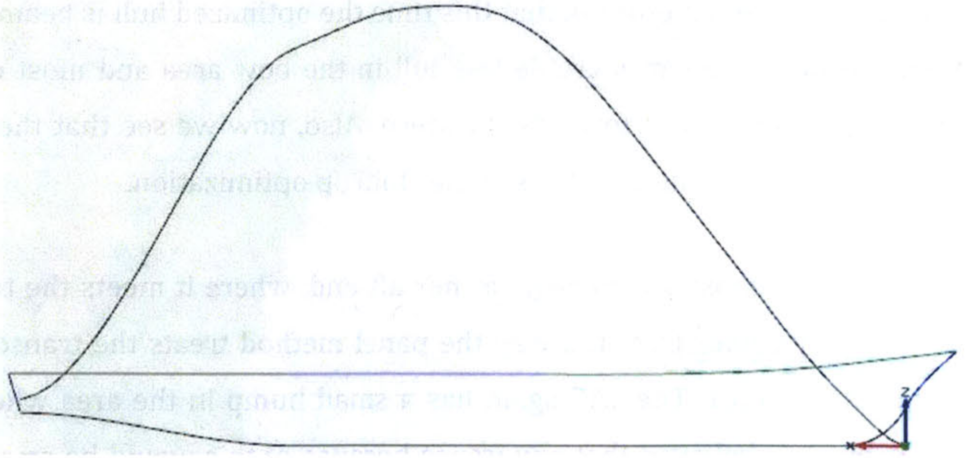


Figure 54 - Sectional Area Curve and Side View of Panel-Optimized Hull

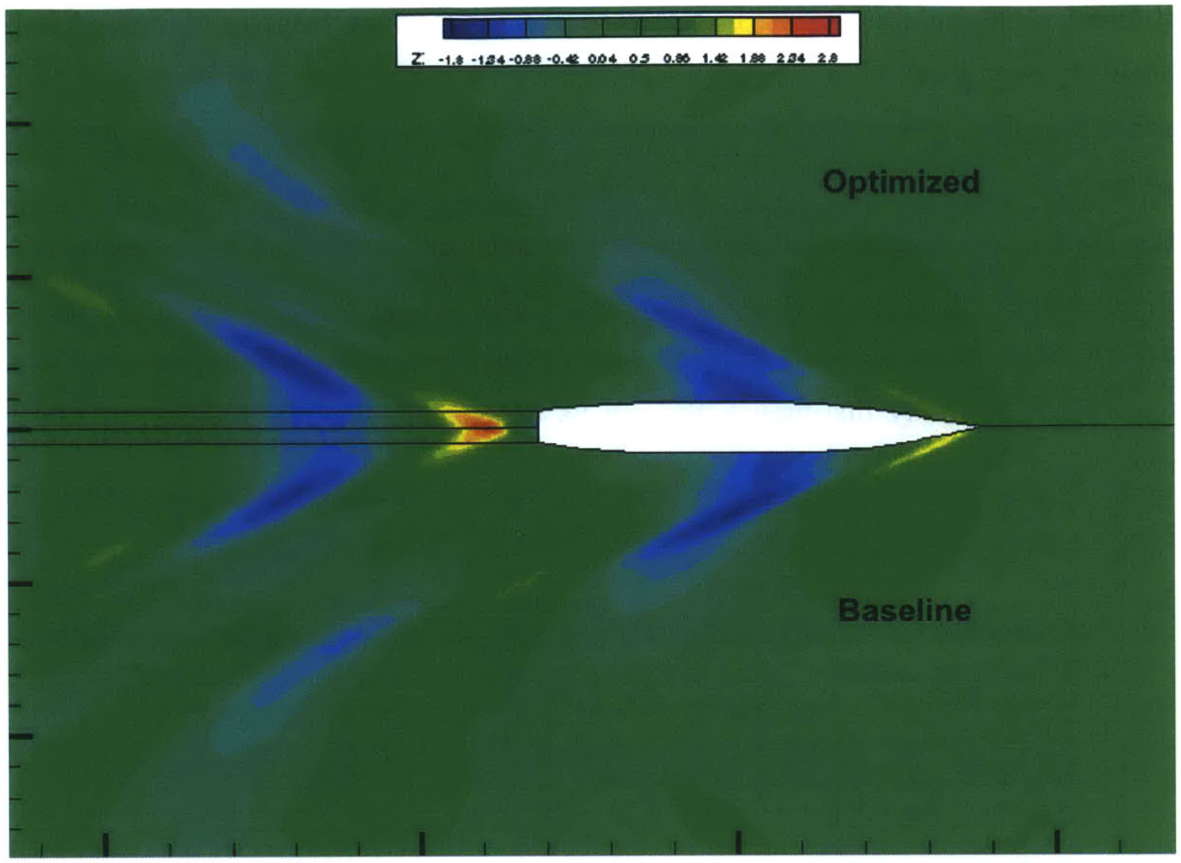


Figure 55 - Wave Pattern Comparison Between Baseline and Panel-Optimized Hull

5.3.3. Considering Internal Arrangements

In the case where the internal arrangements are modeled, both optimizations were also successful and the convergence of the algorithm can be seen in the next sections. Clear pareto frontiers were formed for both optimizations as well, indicating a good approach of the algorithm to the problem. The non-dimensional resistance, displacement and LCB for the baseline and optimal hulls are summarized in Table 27. The non-dimensional resistance was calculated for the baseline and optimal hulls by using both the Holtrop and the panel method. However, only one of the two was each time used as the objective function of the optimizations.

From the results, it is observed that Holtrop method managed to improve the resistance of the ship by 2.3%, while the panel method did twice as much, by reducing the resistance by almost 6%. In this case, we see that the Holtrop optimized hull has smaller resistance even when the resistance is calculated with the panel method, something not observed in the previous case. On the other hand, the panel-optimized hull seems to be again worst than the baseline, when the Holtrop method is used to calculate the resistance. Finally, it seems that in the most constraint problem, where we also account for the internal arrangements, the Holtrop method does not perform that well as before. This can be accounted to the smaller fidelity and resolution of the methos, not allowing it to really distinguish the effects of small geometric changes on the ship's resistance.

	Baseline	**Holtrop Optimal**	**Δ %**	**Panel Optimal**	**Δ %**
Resistance Holtrop [Dmnlss]	0.01466051	**0.014321890**	-2.31%	0.014830298	1.16%
Resistance Panel [Dmnlss]	0.012177773	0.012022024	-1.28%	**0.011449705**	-5.98%
LCB [meters]	71.528	71.961431	0.61%	73.617015	2.92%
Displacement [mtons]	7715.62	7706.124	-0.12%	7708.0266	-0.10%

Table 27 - Summary of Optimal Hulls vs Basline for the Optimization with Internal Arrangements

The strategy chosen by the algorithm in both optimizations was to minimize the displacement. Once again, the panel method moved the displaced volume of the hull towards the stern, in order to have a finer bow, generating smaller waves. Table 28 has a comparative summary of the design variables of the optimal hulls against the baseline. Again, we can't really say that the algorithm followed the same strategy when minimizing the resistance in the two cases. Once again, in both cases, some variables have closely approached their minimum or maximum bounds (in red), as they were reported in Table 23. Finally, a new optimization can be started, where larger bounds for these variables should be allowed, or we could use the optimal hull as the new baseline and re-define the new bounds based on it.

	Baseline	Holtrop Optimal	Difference	Panel Optimal	Difference
Cwp	**0.818**	0.7948114	-2.83%	0.78911951	-3.53%
DWLaftAngle	**10**	8.2592813	-17.41%	8.4624094	-15.38%
EntranceAngle	**12**	9.7540307	-18.72%	9.9380911	-17.18%
H_BeamTransomOverH_Beam	**0.6**	0.58322637	-2.80%	0.64979538	8.30%
KeelAngleAtTransom	**2**	0.76050965	-61.97%	0.39826047	-80.09%
KeelRisePoint	**0.65**	0.68374922	5.19%	0.6821603	4.95%
x_Bmax	**0.5**	0.4643328	-7.13%	0.46596094	-6.81%
Deadrise Angle					
0_05	**60**	54.525154	-9.12%	60.560952	0.93%
0_15	**30**	31.879377	6.26%	28.652369	-4.49%
0_25	**9**	9.4650172	5.17%	9.3295811	3.66%
0_50	**1.5**	0.61651789	-58.90%	1.3884314	-7.44%
Flare Angle					
0_05	**18**	17.16835	-4.62%	16.605786	-7.75%
0_25	**22**	20.300325	-7.73%	24.17113	9.87%
0_50	**8**	7.8037934	-2.45%	7.6209293	-4.74%
0_75	**13**	12.871398	-0.99%	13.522757	4.02%
1_00	**35**	37.304334	6.58%	38.476822	9.93%
FullnessFactor					
0_25	**0.6**	0.65820829	9.70%	0.49351339	-17.75%
0_50	**1**	0.97892729	-2.11%	1.0581888	5.82%
0_75	**0.4**	0.44523384	11.31%	0.32220218	-19.45%
1_00	**1.2**	1.3274617	10.62%	1.2292424	2.44%
L_Over_Beam	**8.6**	8.5990486	-0.01%	8.2764649	-3.76%

Table 28 - Comparison of Design Variables between the Optimal Hulls and Baseline

5.3.3.1 Holtrop Optimized Hull

The history of the optimization process can be seen in Figure 56. The algorithm, again, managed to establish a pareto frontier and to follow it to a minimum value for the objective function. The diversity of the evaluated designs is large enough, ensuring that the algorithm made a fairly good search of the trade space. The data points seem to follow very closely the pareto frontier and they continue to have a declining trend when the process is finished. For this reason, it would be beneficial to allow for more generations in a future optimization process, as a hull with less resistance might be generated.

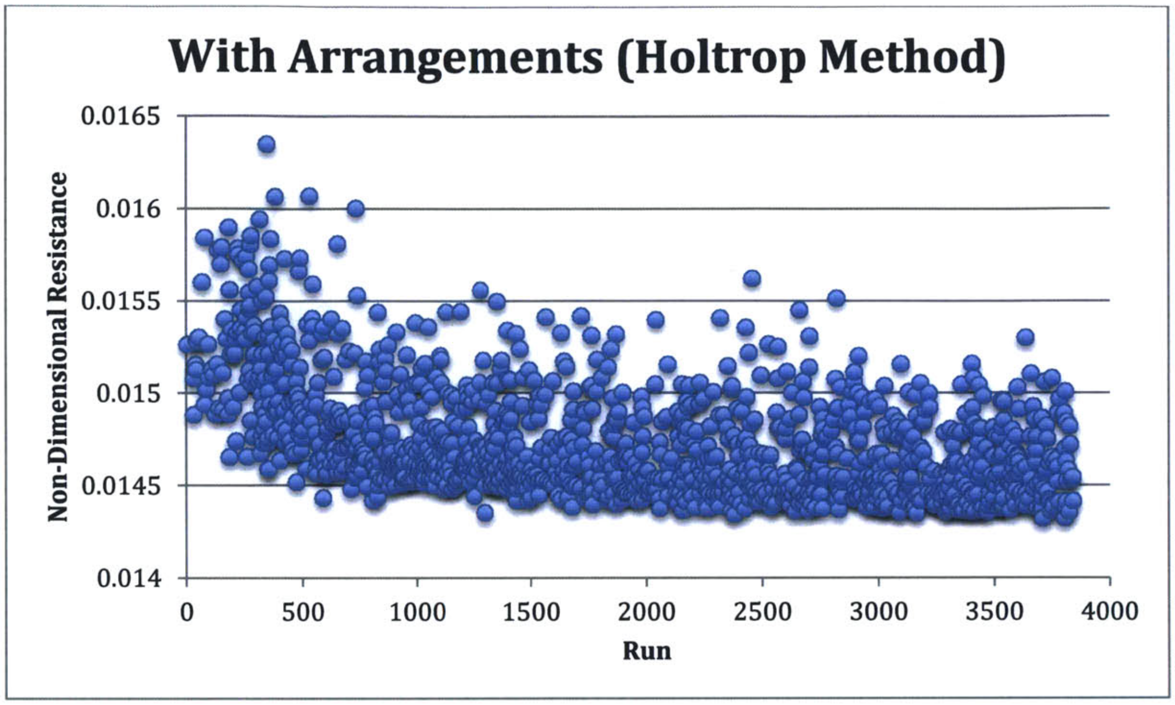


Figure 56 - History of Optimization Process

The body-plan and side view of the optimized hull, along with the Sectional Area Curve (SAC) can be seen in Figure 57 and Figure 58 respectively. From the body-plan and the data in Table 28, we can observe that the optimized hull has almost the same beam with the baseline. The sections are slender at the bow and a bit fuller in the stern, while the shape of the keel at the part where it meets the transom is almost the same with the baseline. On the other side of the keel, the keel rise point has moved a bit aft, as it can be seen in the last 5 sections. Finally, in Figure 59, we can see a comparison of the wave pattern calculated for the optimal hull by using the panel method with the wave pattern of the baseline. As the difference in the resistance is just 2%, it is very difficult to see any remarkable differences in the two patterns.

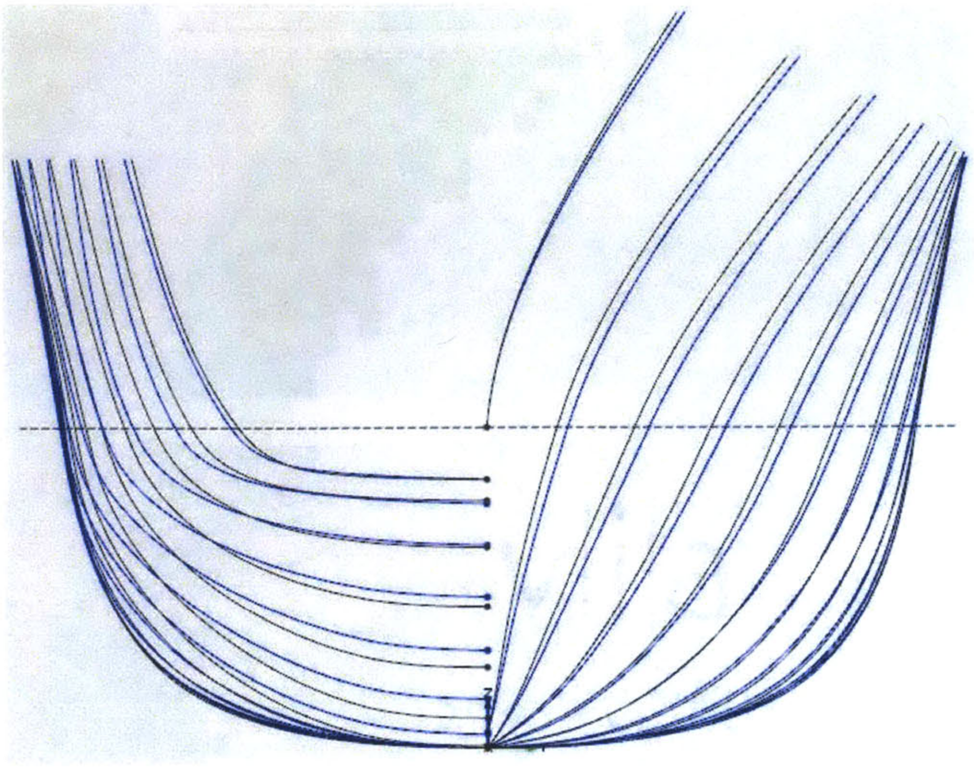


Figure 57 - Body-Plan Comparison between Baseline (blue) and Holtrop Optimized Hull (black)

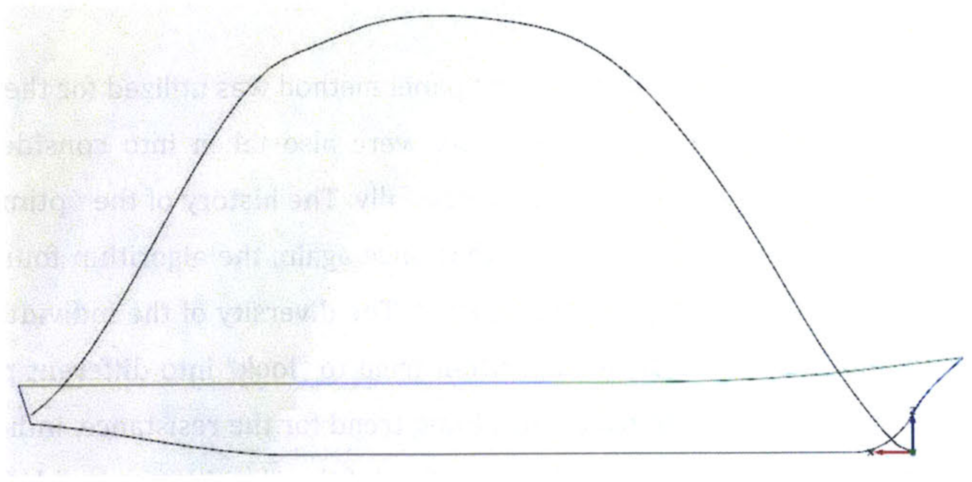


Figure 58 - Sectional Area Curve and Side View of Holtrop Optimized Hull

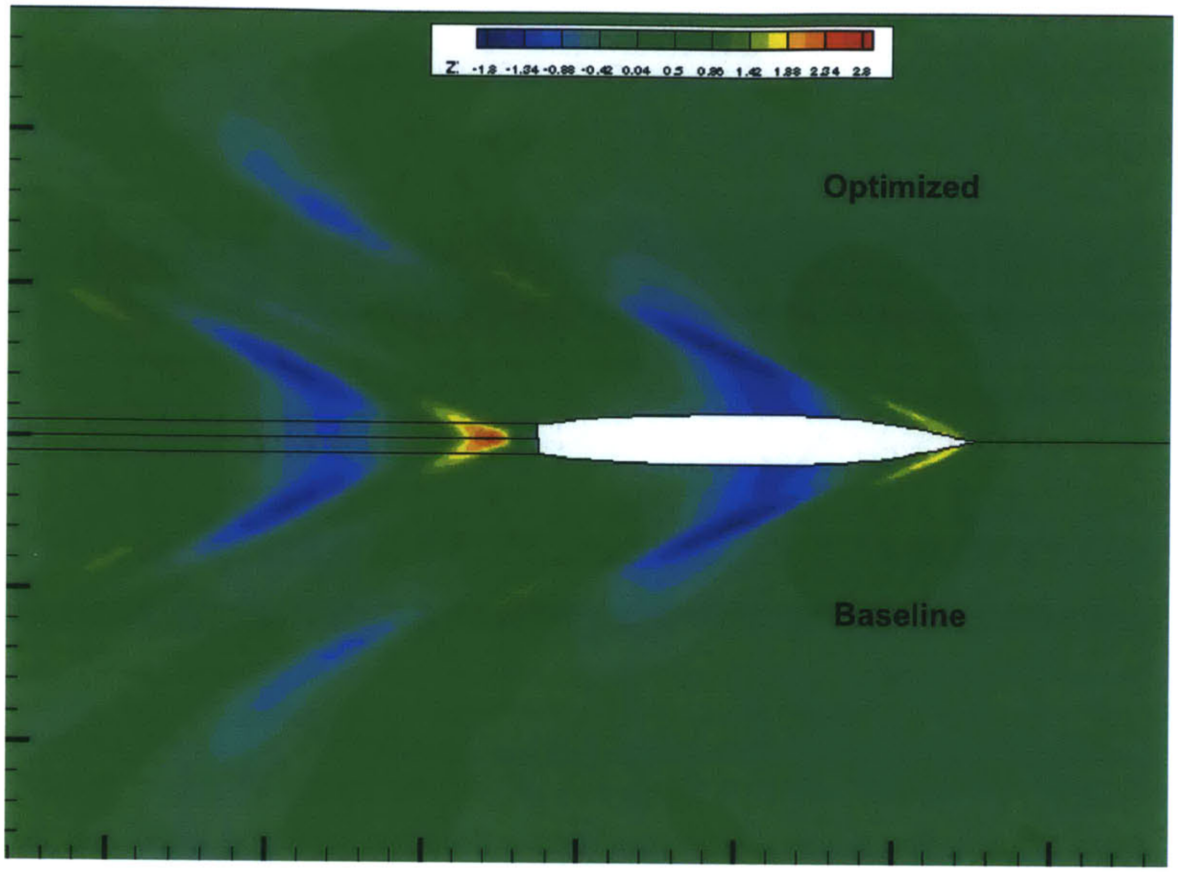


Figure 59 - Wave Pattern Comparison between Baseline and Holtrop Optimized Hull

5.3.3.2 Panel Method Optimized Hull

For the final optimization, where the panel method was utilized for the estimation of the resistance and internal arrangements were also taken into consideration, the algorithm managed to treat the problem succesfully. The history of the optimization can be seen in Figure 60, where we observe that once again, the algorithm found a pareto frontier and followed it throughout the process. The diversity of the individuals studied is very good and indicates that the algorithm tried to "look" into different parts of the trade space. The data seems to have a declining trend for the resistance, indicating once again that better results would have been realized if the algorithm was able to continue for more generations.

After examining the history diagrams of all 4 optimization processes, we can definitively say that the selection of the settings in Table 24 was successful. However, this might not be true if the constraints in the same table or the geometric constraints from the internal arrangements were different.

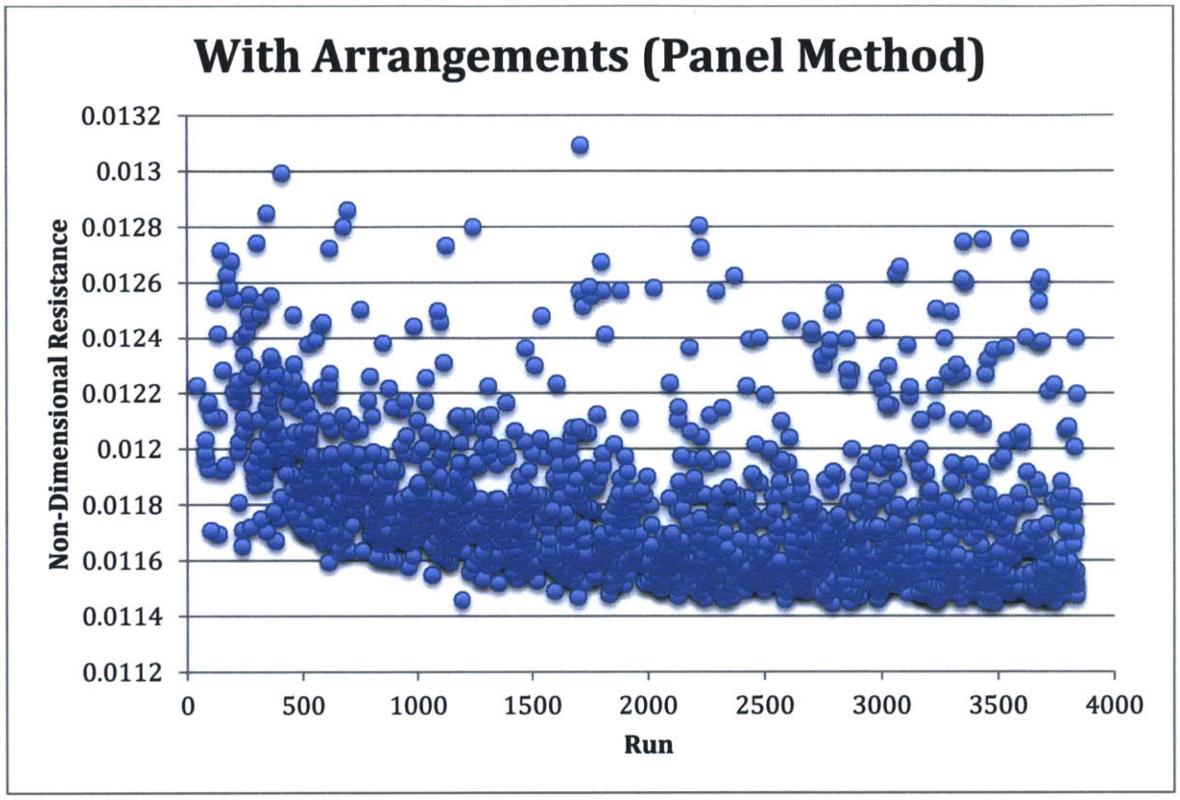


Figure 60 - History of Optimization Process

The observations from the body-plan and Sectional Area Curve (SAC) are almost the same as in 5.3.2.2, indicating that the algorithm followed a similar strategy of moving the volume aft and having a more slender bow. Finally from the comparison of the wave fields in Figure 63, there is a clear indication that the optimal hull creates smaller waves in the bow and thus has better hydrodynamic performance than the baseline.

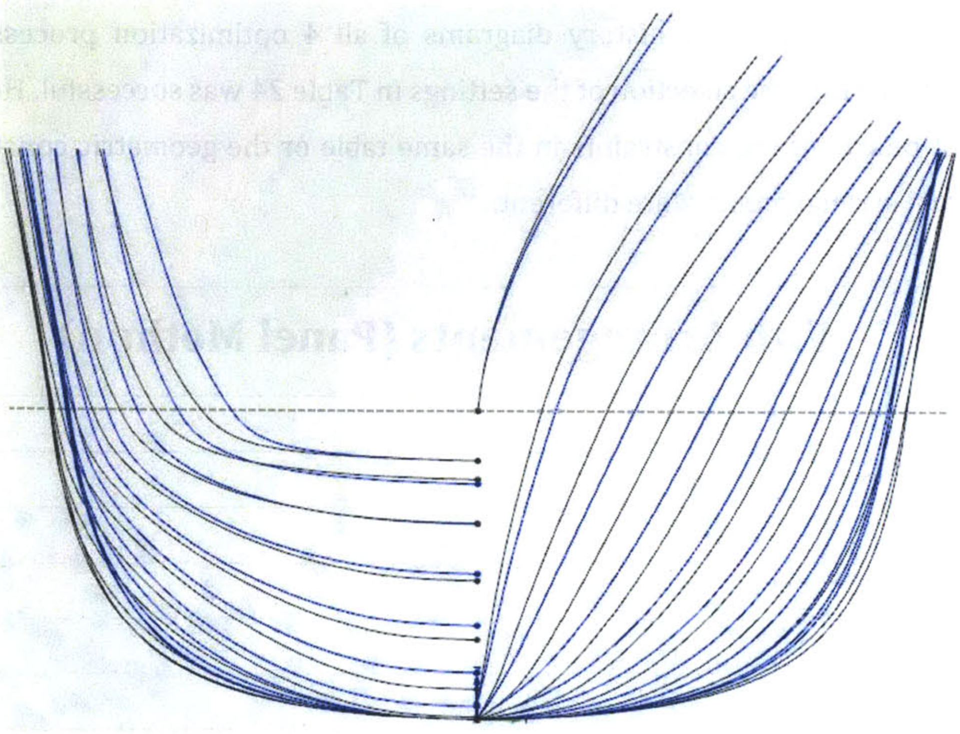


Figure 61 - Body-Plan Comparison between Baseline (blue) and Panel-Optimized Hull (black)

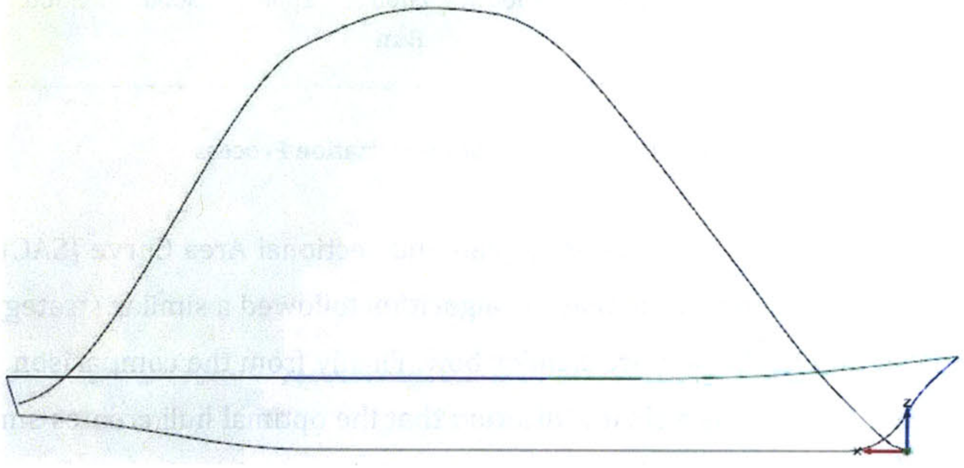


Figure 62 - Sectional Area Curve and Side View of Panel-Optimized Hull

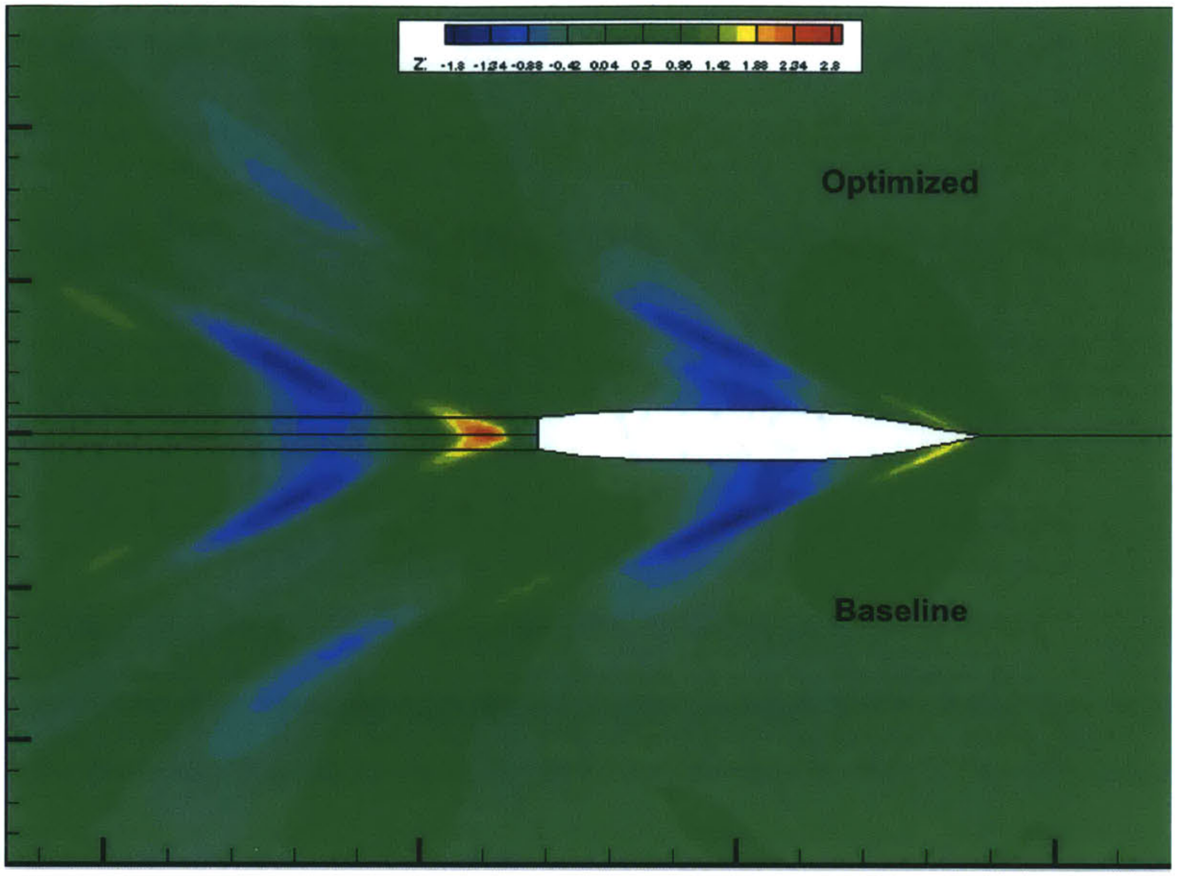


Figure 63 - Wave Pattern Comparison Between Baseline and Panel-Optimized Hull

(This Page Intentionally Left Blank)

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

6.1. Conclusions

This thesis developed a tool that parametrically optimizes hull-forms by considering geometric constraints from the internal arrangements. The targeted geometries were fast displacement mono-hull combatant-type ships. The tool is to be used in the conceptual and early stage design of the ship and aims to aid the designer on integrating the internal arrangements with the hull design.

In the traditional design process of ships, the internal arrangements are considered only after the hull has been selected. However, the large weight, space and volume of equipment like the machinery, poses a great difficulty on the designer. Navy ships, by design and need, are ships with high density of equipment and many times compromises have to be done on the selection of the machinery equipment, in order to fit it inside the belly of the ship.

The tool gives to the designer the opportunity to consider both the hull and the arranged equipment in a more holistic and integrated fashion. The designer can select the machinery layout to be used in the ship and the tool will "wrap around" a hydrodynamically optimized hull. By this, the hassle of trying to fit bulky equipment, like engines, is removed from the designer, allowing him to deal with other complex an complicated problems of the design.

The user has the ability to use either the Holtrop method or a potential flow panel method to conduct the hydrodynamic optimization of the hull-form. The two methods are very different allowing for either a very fast and low-resolution or a slower and more accurate approach to the problem. Even when the panel method is used, the optimized hulls should be further studied about the viscous effects of the flow with a CFD code like RANSE.

The hydrodynamic problem of a ship is very complex by itself and becomes even more complicated when geometric constraints are introduced. For this reason, heuristic methods, like genetic algorithms, are used in this tool. The tool is very flexible, as it allows to the designer to easily setup custom optimization problems, based on any special need of the design.

For demonstration purposes, two different case studies were demonstrated in this thesis. First the "free" problem is treated, where there are no geometric constraints. Consequently, internal arrangements are introduced and the optimization is conducted again. In both cases, both resistance estimation methods were used and the lack of fidelity of the Holtrop method was demonstrated. Moreover, the panel method managed to reduce the resistance of the optimal hulls by more than 5% in both cases, when compared with the starting point.

For many years, the design process of ships was dominated by the hull-form selection. The internal arrangements had secondary importance, as they had to be adapted and conform to the selected hull. However, in the recent years, operability, affordability and operational requirements have made the arrangement of machinery spaces and equipment more important.

With this tool, it is the first time known to the author that the two systems, hull and propulsion arrangements, can be treated together. It is believed that benefits can be realized when the optimization of these two systems is treated in a more holistic and integrated way. The ability given to the designer to easily study the interaction of the propulsion plant with the hull-form will allow for better decision to be made in the early stage design of the ship, where uncertainty is high and decision-making is difficult and risky.

6.2. Recommendations

As with any tool, user feedback will indicate areas of shortcomings and needed improvement. However, the future work can contain the optimization over a speed profile, a method to capture viscous effects, especially in the stern and multi-objective optimization. Further, more sophisticated optimization software might be used for the optimization process, as Friendship Framework® can be launched in "batch mode"; without a GUI.

Even though, Friendship Framework® comes with native optimization algorithms, like the NSGA-II used in the tool, it is not a software streamlined to conduct large and complex optimizations. Connecting the tool with a software like ModeFrontier® would allow the use of more algorithms and an easier and faster post-processing of the data. This would allow possibly for better results to be attained and provide the designer a better understanding over the treated problem.

The optimization algorithm used in the tool is capable of treating multi-objective problems. An interesting improvement would be to modify the tool, so that it can take into consideration hydrostatic parameters like the metacentric height. This trivial modification in the tool might slightly increase the optimization time needed, due to the extra complexity of multi-objective optimizations. However, it would be a considerable aid to the designer if such parameters were taken into consideration, as they are highly correlated with stability and indirectly connected with the motions of the ship.

Viscous effects of the flow are not considered in the optimization, as they are outside of the potential flow theory limits. However, the effects of viscosity, especially in the stern of the ship, are very important. Even today, with very powerful computers, the solution of RANSE for a ship might take from a day to a week, making fully viscous flow solvers prohibiting for any optimization method that needs to evaluate hundreds of hulls.

Luckily, there is a method based on potential flow that can give some good results about the viscous effects taking place in the stern of the ship. This method considers the so-called secondary flow (in y- and z- axis) and is based on an axiom that the minimization of the energy of the secondary flow leads to decreased form drag (Suzuki, et al., 2005). It is highly recommended that this method should be added in the tool, by post-processing the data of the panel method and thus allowing for a better understanding of the viscous effects in the stern of the ship.

Finally, navy ships never operate on a single speed, but rather to a range of speeds. For this reason, it would be very beneficial for the tool to be able to conduct the optimization over the operational profile of the ship and not for a single speed. With fuel cost becoming a big issue to the owner, optimized hull-forms along an operational profile would be of extreme interest, as they can drastically reduce the fuel cost associated with the operation of the ship.

References

Biliotti, I. et al., 2011. Automatic Parametric Hull Form Optimization of Fast Naval Vessels. Honolulu, s.n.

Brizzolara, S., 2000. Theory and Application of Numerical Methods for the Evaluation of High Speed Craft Wave Resistance, s.l.: s.n.

Bruzzone, D., 1994. Numerical Evaluation of the Steady Free Surface Waves. Tokyo, Japan, s.n., pp. 126-134.

Dawson, C. W., 1977. A Practical Computer Method for Solving Ship-Wave Problems. Berkeley, s.n.

Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T., 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, April, 6(2), pp. 182-197.

Doerry, N. H., 1994. Integrated power system for marine applications. Naval Engineers Journal, May.pp. 77-90.

Doerry, N. H. & Firemann, H., 2006. Designing all electic ships. Ann Arbor, MI, s.n., pp. 475-497.

Gale, P. A., 2003. The ship design process. In: Ship Design & Construction. Jersey City(NJ): The Society of Naval Architects and Marine Engineers, pp. 1-40.

Gilmer, T. C. & Johnson, B., 1982. Introduction to Naval Architecture. Annapolis(Maryland): United States Naval Institute.

Harries, S., 1998. Parametric design and hydrodynamic optimization of ship hull forms, Berlin, Germany: Technische Universitat Berlin.

Holtrop, J., 1984. A Statistical Re-Analysis of Resistance and Propulsion Data. International Shipbuilding Progress, November, Volume 31, pp. 272-276.

Holtrop, J., 1984. A Statistical Re-Analysis of Resistance and Propulsion Data. s.l., s.n., pp. 272-276.

Holtrop, J. & Mennen, G., 1982. An Approximate Power Prediction Method. Internation Shipbuilding Progress, 29(335), pp. 166-170.

Holtrop, J. & Mennen, G. G. J., 1978. A statistical power prediction method. s.l., s.n.

Jason, K., 2013. Navy Unveils Its First Laser Gun. [Online] Available at: http://navylive.dodlive.mil/2013/04/10/solid-state-laser-gun-to-be-placed-aboard-uss-ponce/
[Accessed 25 April 2013].

Jurkiewicz, D., 2012. Modular Machinery Arrangement and Its Impact in Early-Stage Naval Electric Ship Design, Cambridge: MIT.

Larson, L. & Raven, H. C., 2010. The Flow Around the Hull and the Viscous Resistance. In: The Principles of Naval Architecture Series - Ship Resistance and Flow. New Jersey: The Society of Naval Architects and Marine Engineers, pp. 51-77.

Larsson, L. & Raven, H. C., 2010. Numerical Prediction of Resistance and Flow Around the Hull . In: R. J. Paulling, ed. The Principles of Naval Architecture Series - Ship Resistance and Flow. New Jersey City: The Society of Naval Architects and Marine Engineers, pp. 107-154.

Lechter, J. S., 2009. The Principles of Naval Architecture Series: The Geometry of Ships. Jersey City(NJ): The Society of Naval Architects and Marine Engineers.

Marzi, J., circa 2006. VIRTUE – The Virtual Tank Utility in Europe Extending the scope and capabilities of maritime CFD , Hamburg: VIRTUE Project.

McCoy, T. et al., 2007. Hybrid Electric Drive for DDG-51 Class Destroyers. American Society of Naval Engineers, pp. 83-91.

McNab, I. R. et al., 2004. Development of a naval railgun. s.l., s.n., pp. 76-80.

Mitchell, M., 1998. An Introduction to Genetic Algorithms. s.l.:Massachusetts Institute of Technology.

Piegl, L. & Tiller, W., 1997. The NURBS book. 2nd ed. Germany: Springer.

Raven, H. C. & Starke, A. R., 2002. Efficient methods to compute ship viscous flow with a free surface. Fukuoka, Japan, s.n.

Saunders, H. E., 1957. Hydrodynamics in ship design. New York City: The Society of Naval Architects and Marine Engineers.

Suzuki, K., Kai, H. & Kashiwabara, S., 2005. Studies on the optimization of stern hull form based on a potential flow solver. Marine Science and Technology.

APPENDIX

A

USER GUIDE

This user guide aims to help the designer use the tool. At no way does it substitutes the user manual, help files and tutorials of Friendship Framework®. The user should never try to alter the structure of the tool if he/she is not very familiar with Friendship Framework®. The guide is designed as a tutorial, first some general information is given for the tool and then there is an explanation on how to import a hull, set values in the variables, set internal arrangements and constraints and launch an optimization. Finally, a description on how to launch the tool in batch mode and run some scripts is given as well.

A.1. General Information

In the object tree, the user can find all the information that needs to be altered in order to use the tool effectively. The major subfolders along with their indented use are described below:

1. *"Computations"* is the folder where all the needed parameters for the computations are located. There are three computations executed in this tool, the hydrostatics, which is a Friendship Framework® function, the panel method and the Holtrop method computation for resistance.
2. *"Export"* is the folder where the ship's surfaces have been manipulated in order to match their directionality and are ready in groups to be exported. The surfaces exported are the hull, the weather and the transom. The user can export them in any of the supported from the software formats (iges,

stl, etc). Further, the user can select to export the full ship, the half ship or even export the model surfaces, upon a scale that can be controlled.

3. *"Global"* contains constants and elements like mirror transformations and principle planes that are used globally in the tool.
4. *"Hull_to_be_fitted"* contains an automated mechanism that calculates different crucial parameters of a hull to be modeled.
5. *"Internal_Arrangements"* is the folder where the internal arrangements of the ship are really defined and modeled.
6. *"Miscellaneous"* contains a DWL that can be used when viewing the body-plan of the hull in the 3D View window. Also, contains cross section (NURBS and F-Spline) in the positions where there are variables to control the deadrise, fullness and flare curves. Also one NURBS and one F-Spline free cross section exists that the user can position anywhere along the length of the ship. The main purpose of these sections is to aid the user on deciding what would be the values for the variables controlling the control curves of the tool.
7. *"SHIP"* is the folder where the complete definition of all the geometric elements describing the hull geometry takes place. It is the centerpiece of the tool, as without it, no geometry definition can exist.
8. *"Ships_Library"* is a folder containing imported hull surfaces of ships, or it can also contain surfaces of hulls build in the tool.
9. *"Variables"* is the folder that the user should and will use the most, as it contains all the variables that really control the geometry of the generated hulls. Also, from this folder, the user has the ability to control the number of longitudinal and transverse panels to be used by the panel code in the resistance estimation[5].

[5] Intentionally has not given a lot control to the user to alter the settings of the panel method, as it is dangerous to do without knowing. If someone knows how to manipulate the code, then the input parameters in the *"Computation"* should be changed accordingly. Also, in case a different code is used for the resistance estimation, any variables defined here would be irrelevant.

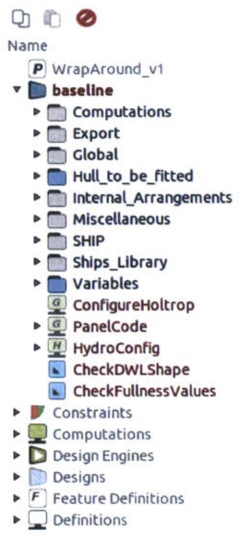
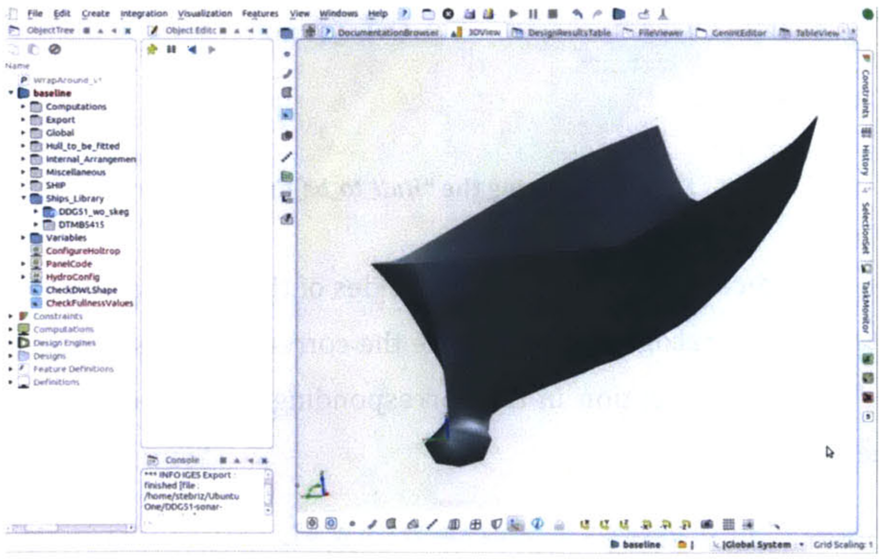

Name
- WrapAround_v1
- baseline
  - Computations
  - Export
  - Global
  - Hull_to_be_fitted
  - Internal_Arrangements
  - Miscellaneous
  - SHIP
  - Ships_Library
  - Variables
  - ConfigureHoltrop
  - PanelCode
  - HydroConfig
  - CheckDWLShape
  - CheckFullnessValues
- Constraints
- Computations
- Design Engines
- Designs
- Feature Definitions
- Definitions

Figure 64 - Object Tree

A.2. Importing a Hull Geometry

A hull surface can be imported by using the *"File-Import"* menu of the software. Once the surface is imported, it should be checked in the 3DView and then the surfaces should be moved into the *"Ships_Library"* folder and give them a meaningful name.



Name
- WrapAround_v1
- baseline
  - Computations
  - Export
  - Global
  - Hull_to_be_fitted
  - Internal_Arrangemen
  - Miscellaneous
  - SHIP
  - Ships_Library
    - DDG51_wo_skeg
    - DTMB5415
  - Variables
  - ConfigureHoltrop
  - PanelCode
  - HydroConfig
  - CheckDWLShape
  - CheckFullnessValues
- Constraints
- Computations
- Design Engines
- Designs
- Feature Definitions
- Definitions

```
*** INFO IGES Export :
finished [file :
/home/stebriz/Ubuntu
One/DDG51-sonar-
```

Figure 65 - 3DView and *"Ships_Library"*

A.3. Measuring Important Parameters of the Imported Hull

If the imported is to be used as a starting point for an optimization or if the user wants to model it in the tool, then the *"Hull_to_be_fitted"* folder should be used. Inside the *"Surfaces"* subfolder of the *"Hull_to_be_fitted"*, the user should indicate which is the starboard (STBD) or right side of the hull and which is the left side or port. If the imported sides of the hull consist of more than one surface, then a polysurface and possibly image surfaces should be used to fix the orientation and create single surface that can be used for the automated measured of key hull parameters.

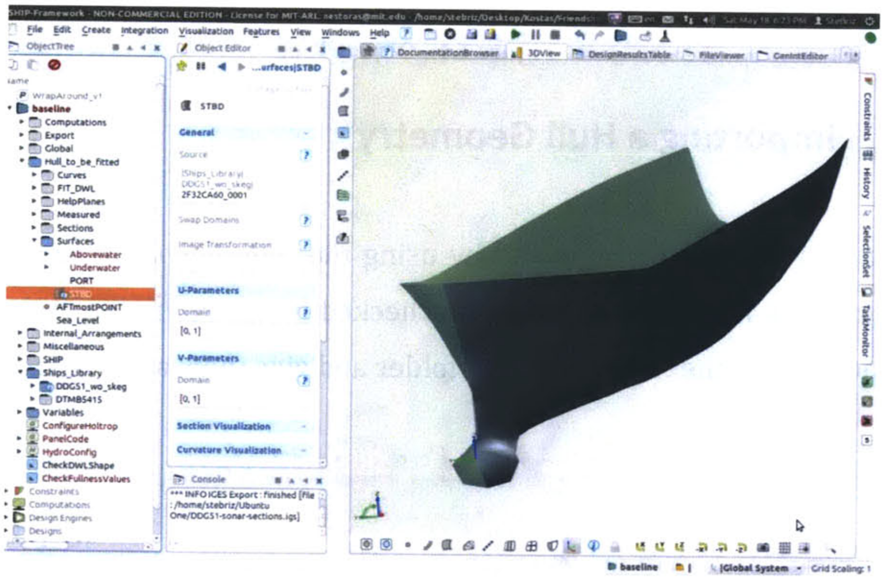
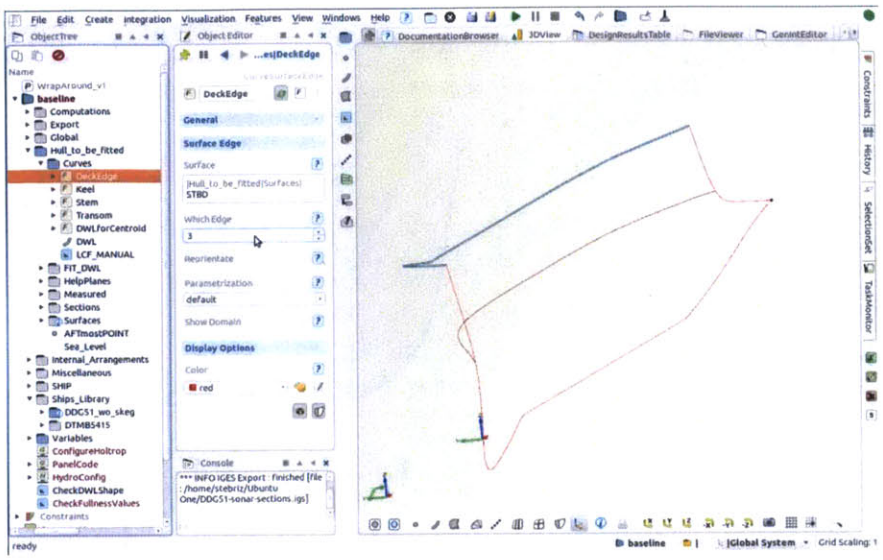


Figure 66 - Using the "*Hull_to_be_fitted*" Folder

After the indicating the port and stbd sides of the hull, the user should make sure that the automatically recognized curves are the correct. For this, the user might need to change the *"Which Edge"* option in the corresponding curves, so that the correct curve shows in the 3DView.



...es|DeckEdge

DeckEdge

General

Surface Edge

Surface: |Hull_to_be_fitted|Surfaces| STBD

Which Edge: 3

Reorientate

Parametrization: default

Show Domain

Display Options

Color: ■ red

Console

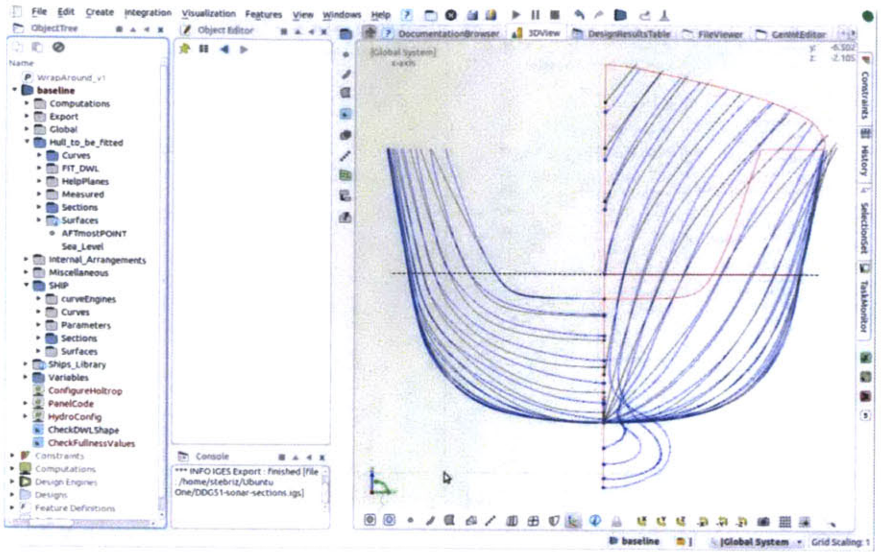
```
*** INFO IGES Export : finished [file
: /home/stebriz/Ubuntu
One/DDG51-sonar-sections.igs]
```

Figure 67 - Identifying the Correct Curves

After identifying correctly the curves, the script file "values_in_variables.fsc" can be used to automatically populate the models variables with the ones measured from the imported geometry. The "File-Execute Script" menu of the software can be used to run the script file.

A.4. Fine Tuning the Variables

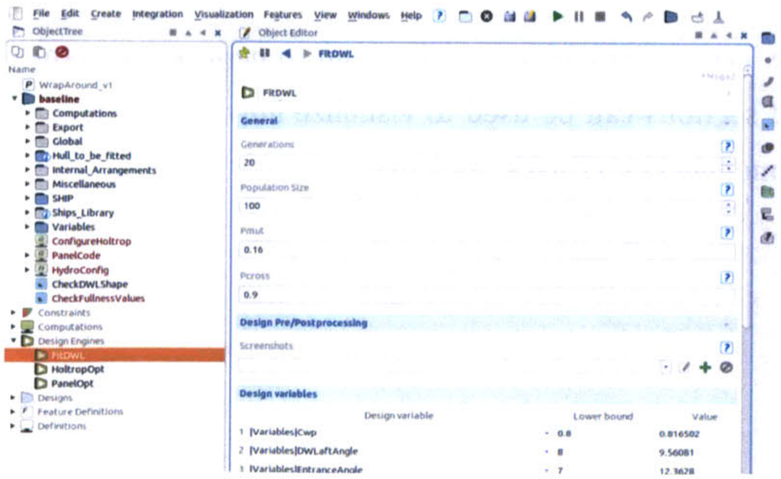
Once the script has been run, the user might want or not to fine tune the fitness of the modeled hull with the imported. For this reason, it is a good practice to turn on the visibility to the "*Section*" subfolders of the "*Hull_to_be_fitted*" and the "*SHIP*". 20 sections, along with half sections can be used to visualize how the sections of the imported hull (blue) and the modeled hull (black) look like.



```
*** INFO IGES Export : finished [File
: /home/stebriz/Ubuntu
One/DDG51-sonar-sections.igs]
```

Figure 68 - Comparison of Cross Section between the Imported and Modeled Geometries

Before the user starts to change the values of the deadrise angle, flare angle and fullness factor along the length of the ship, he/she should make sure that the DWL is close enough to the imported hull. For this reason an automatic optimization algorithm based on the NSGA-II (*"FitDWL"*), which alter the variables controlling the shape of the DWL exists in the tool. The fitting of the modeled DWL with the one from the imported geometry is evaluated by using 17 points along the length of the ship, where the deviation is calculated and in the end the sum of the squares is used as the objective function.



FitDWL

General

Generations
20

Population Size
100

Pmut
0.16

Pcross
0.9

Design variables

	Design variable	Lower bound	Value
1	[Variables]Cwp	0.8	0.816502
2	[Variables]DWLaftAngle	8	9.56081
3	[Variables]EntranceAngle	7	12.3628

Figure 69 - *"FitDWL"* Optimization

The user can select to set different set of variables to be considered in the calculation from the default. After the optimization has been completed, the user should investigate the quality of the optimized design to see which of the combinations with the lowest least square sum has the best fit to the geometry to be modeled. After this step, the user can alter the variables of the tool, in order to fir the imported geometry.

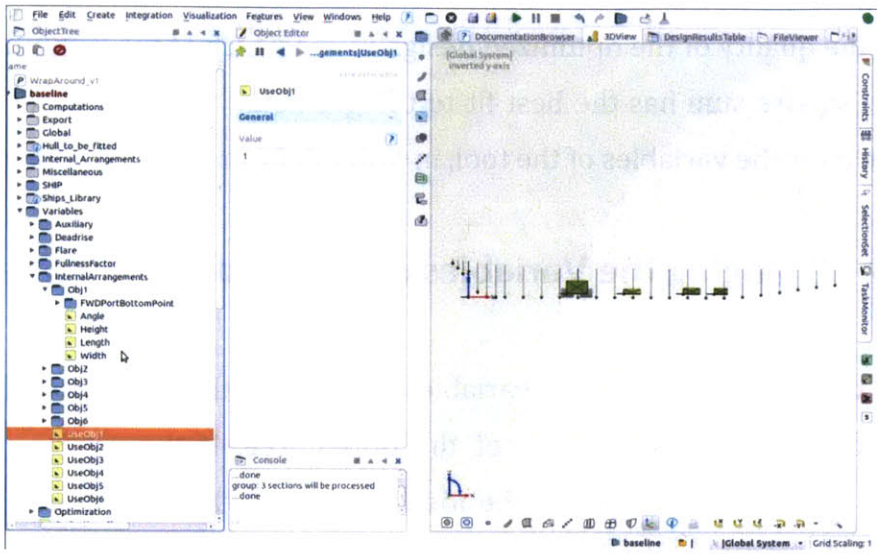
A.5. Changing the Variables of the Modeled Geometry

The user can access all the variables controlling the geometry of the generated hull through the *"Variables"* folder of the object tree. If the user wants, the folder *"Miscellaneous"* can be used to aid the identification of the best values for the control curves in the different longitudinal positions.

It is highly recommended to the user to start changing or correcting the variables from these that control the profile curve first, then the deck edge (auxiliaries) and finally the actual deadrise, flare angle and fullness factor. If the rest of the ship's lines are not close enough to the hull to be modeled, then the process of matching the geometry is extremely difficult.

A.6. Modeling Internal Arrangements

The internal arrangements of the ship can be modeled by using the corresponding subfolder of the *"Variables"*. It is a good practice to turn on the visibility of the *"Inernal_Arrangements"* folder and visualize any changes in the 3DView. The user should first decicde how many objects he wants to use and put the value 1 in the corresponding *"UseObj"* variable. The user should make sure that if an object is not needed, it has to have a zero value at its *"UseObj"* variable. The dimensions of the object and the position of the forward inboard lower corner, as well as its angle can be defined from the corresponding variables for each object.



```
..done
group: 3 sections will be processed
..done
```

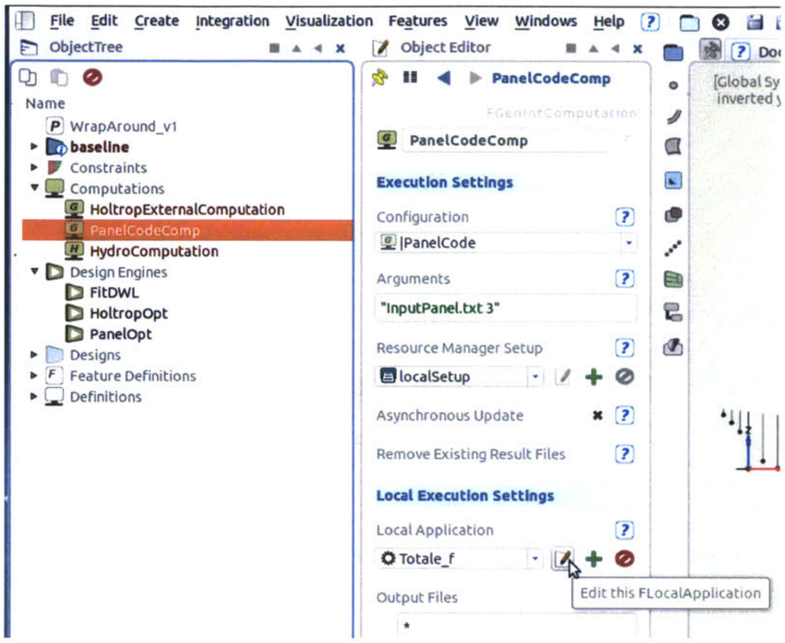
Figure 70 - Defining Internal Arrangements

A.7. Assigning Constraints

The values of the constraints that will be used in the optimizations can be controlled by the *"Optimization-Constraints"* folder inside *"Variables"*. The user should always that these values are correct. If some constraint will not be used in the optimization a very high or low value should be imported in the corresponding, so that it will make the constraint ineffective. If this is not done, then the optimization procedure will not properly executed, as penalties are assigned in the objective functions, even when typically a constraint is not taken into consideration by the algorithm.

A.8. Launching an Optimization

An optimization can be launched by accessing the *"HoltropOpt"* or *"PanelOpt"* in the design engines part of the object tree. The user can check which variables should be used in the optimization and what the bounds for each one should be. It high NOT recommended for the user to remove any of the preselected constraints. If the user wishes to optimize without any constraint for displacement and LCB, he/she should assign a very high or low value for this constraint, as it was discussed above. Before the optimization is launched the used should check in the "*Computations*" part of the object tree that the panel and Holtrop method are pointing in the correct executable files. For the Holtrop method, the executable of Scilab-cli (for Linux) should always be used. For the panel code, the executable is the totale_f.



PanelCodeComp

Configuration: PanelCode

Arguments: "InputPanel.txt 3"

Resource Manager Setup: localSetup

Asynchronous Update

Remove Existing Result Files

Local Application: Totale_f

Output Files: *

Figure 71 - Launch Optimization and Check Executable

A.9. Batch Mode

Friendship Framework® is a software that can be launched in batch mode. Once this is done in a terminal of Linux®, the user can open any project and use the native commands of the software to alter any component of this project. Apart, from that, the software allows the execution of script files in batch mode that can be done in a single command line. This enables the software to be executed from other software packages, without the need of a keystroke and thus making it ideal for optimization. In the next appendix, an example script that opens a project, makes some changes and then exports the generated geometry is given. To terminal command to execute the script in Linux® is given in appendix B.12, as well.

(This Page Intentionally Left Blank)

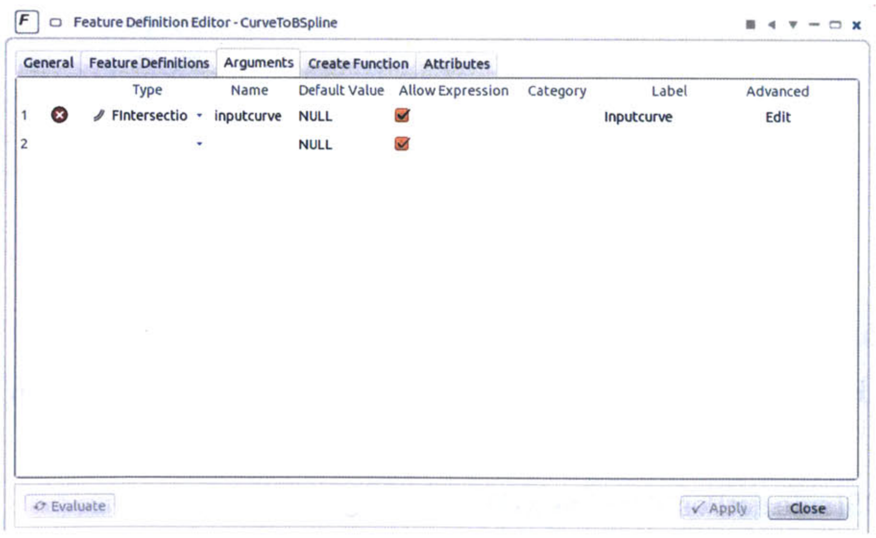
APPENDIX

B

SCRIPTS

This appendix contains the scripts and the arguments tab of the feature definitions used in the model that is described in this thesis. Some of the feature definitions are written from scratch, while others are feature definitions of Friendship Framework® that were slightly modified in order to accommodate special needs of the model. Finally, in the comments of the feature definitions are given credits to any person that helped to build any of them. Feature definitions that are native to Friendship Framework® are not contained here, as the reader can find them in the documentation of the software. Lastly, two script files are included which are used to assign values to the variables from an imported hull and to automatically assign the values of the variables for the modeled DDG-51.

B.1. CurveToBSPline

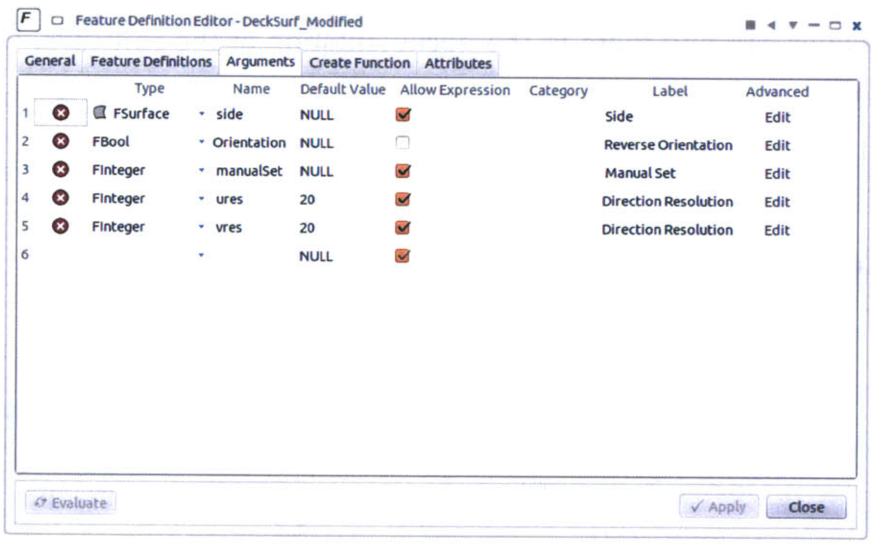


	Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	FIntersectio	inputcurve	NULL	✓		Inputcurve	Edit
2			NULL	✓			

Figure 72 - Arguments of CurveToSBpline Feature Definition

```
// Created by K. Nestoras
// Used in the model to convert an intersection curve to a B-Spline,
// so that the centroid and area of the curve with respect to an axis
// can be computed.

// inputcurve is an intersection curve

bsplinecurve c2(inputcurve)

// end
```

B.2. DeckSurf_Modified



	Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	FSurface	side	NULL	✓		Side	Edit
2	FBool	Orientation	NULL	☐		Reverse Orientation	Edit
3	FInteger	manualSet	NULL	✓		Manual Set	Edit
4	FInteger	ures	20	✓		Direction Resolution	Edit
5	FInteger	vres	20	✓		Direction Resolution	Edit
6			NULL	✓			

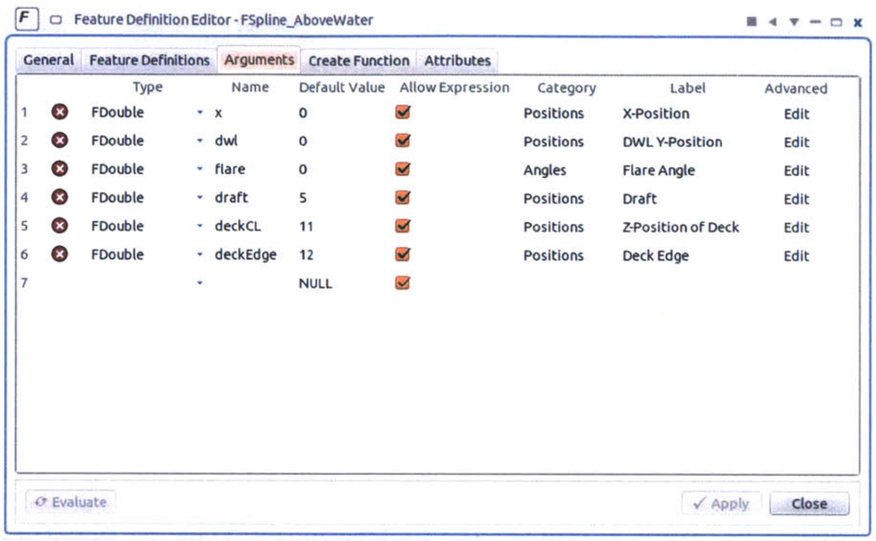
Figure 73 - Arguments of DeckSurf_Modified Feature Definition

```
// Native feature definition of Friendship Framework
// Altered by K.Nestoras in order to enable changes in the u and v resolution

double z_00_05(side.getPos(0,0.5).getz())
double z_05_00(side.getPos(0.5,0).getz())
double z_10_05(side.getPos(1,0.5).getz())
double z_05_10(side.getPos(0.5,1).getz())

double maxZ(-1000)
line domain()

if (maxZ < z_00_05)
 |domain.setStartPos([0, 0, 0])
 |domain.setEndPos([0, 1, 0])
 maxZ = z_00_05
endif
if (maxZ < z_05_00)
 |domain.setStartPos([1, 0, 0])
 |domain.setEndPos([0, 0, 0])
 maxZ = z_05_00
endif
if (maxZ < z_10_05)
 |domain.setStartPos([1, 1, 0])
 |domain.setEndPos([1, 0, 0])
 maxZ = z_10_05
endif
if (maxZ < z_05_10)
 |domain.setStartPos([0, 1, 0])
 |domain.setEndPos([1, 1, 0])
 maxZ = z_05_10
endif

if (manualSet == 1)
 |domain.setStartPos([0, 0, 0])
 |domain.setEndPos([0, 1, 0])
endif
if (manualSet == 2)
```

```
  |domain.setStartPos([1, 0, 0])
  |domain.setEndPos([0, 0, 0])
endif
if (manualSet == 3)
  |domain.setStartPos([1, 1, 0])
  |domain.setEndPos([1, 0, 0])
endif
if (manualSet == 4)
  |domain.setStartPos([0, 1, 0])
  |domain.setEndPos([1, 1, 0])
endif

surfacecurve edge()
edge.setSurface(side)
edge.setDomainCurve(domain)

imagecurve center()
center.setCurve(edge)
scaling toCenter(1,0,1)
center.setImageTransformation(toCenter)

if(reverseOrientation,domain.reverse())

ruledSurface deck(edge,center)
//Altered lines by K.Nestoras
deck.setUresolution(ures)
deck.setVresolution(vres)

//this.setName("deck00")
domain.setVisible(false)
```

// end

B.3. FSpline_AboveWater


Feature Definition Editor - FSpline_AboveWater

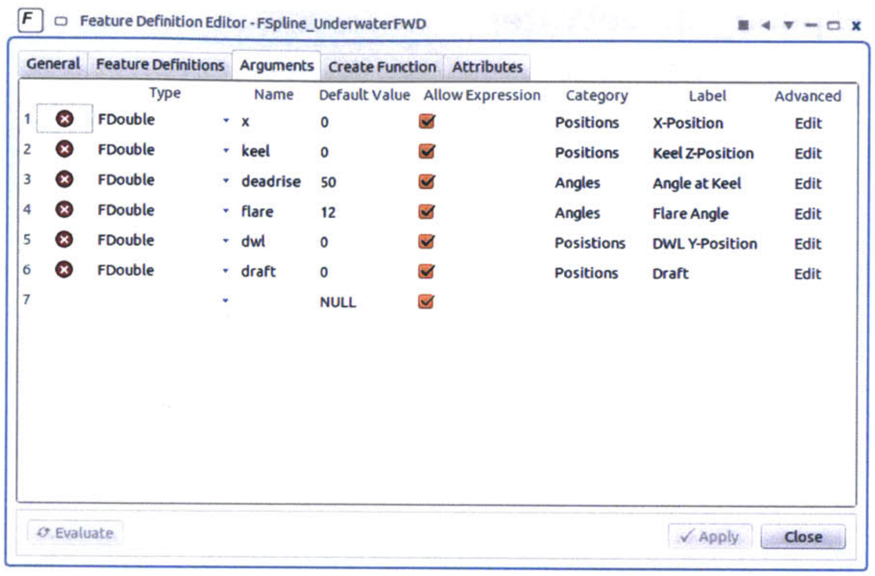
General | Feature Definitions | Arguments | Create Function | Attributes

		Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	⊗	FDouble	x	0	✓	Positions	X-Position	Edit
2	⊗	FDouble	dwl	0	✓	Positions	DWL Y-Position	Edit
3	⊗	FDouble	flare	0	✓	Angles	Flare Angle	Edit
4	⊗	FDouble	draft	5	✓	Positions	Draft	Edit
5	⊗	FDouble	deckCL	11	✓	Positions	Z-Position of Deck	Edit
6	⊗	FDouble	deckEdge	12	✓	Positions	Deck Edge	Edit
7				NULL	✓			

Figure 74 - Arguments of FSpline_AboveWater Feature Definition

```
// Created by K. Nestoras
// This feature definition is used to define the F-Spline curve used by
// the curve engine that defines the above the water side surface of the hull.

// For numerical reasons, the flare angle and the deadrise angle should not get zero values
// multiply by 45 degrees as the control curves are non-dimensional
fdouble flare0(45*max(flare,0.000001))

//flare2=min(flare+10,25)
//fsplinecurve     side([x,dwl,draft],[x,(deckCL-draft)*tan(min(max(10,flare),deckEdge))   +
dwl,deckCL])
fsplinecurve side([x,dwl,draft],[x,deckEdge,deckCL])
side.setActivePlaneYZ()
side.setStartTan(90-flare0)
//section.setEndTan(Flare)
// end
```

B.4. FSpline_UnderwaterFWD



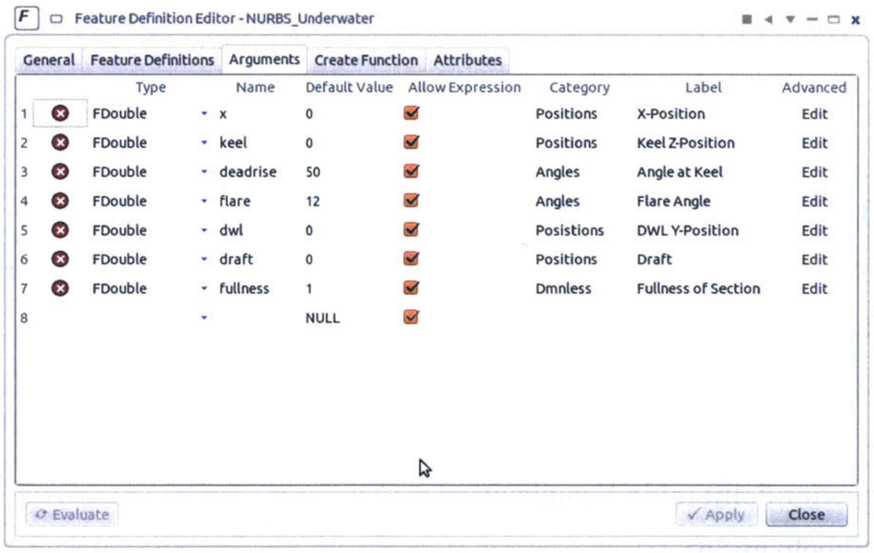
	Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	FDouble	x	0	✓	Positions	X-Position	Edit
2	FDouble	keel	0	✓	Positions	Keel Z-Position	Edit
3	FDouble	deadrise	50	✓	Angles	Angle at Keel	Edit
4	FDouble	flare	12	✓	Angles	Flare Angle	Edit
5	FDouble	dwl	0	✓	Posistions	DWL Y-Position	Edit
6	FDouble	draft	0	✓	Positions	Draft	Edit
7			NULL	✓			

Figure 75 - Arguments of FSpline_UnderwaterFWD Feature Definition

```
// Created by K. Nestoras
// This feature definition is used to define the F-Spline curve used by
// the curve engine that defines the forward part of the underwater side surface
// of the hull.
// For numerical reasons, the flare angle and the deadrise angle should not get zero values
// multiply by 45 degrees as the control curves are non-dimensional
fdouble flare0(45*max(flare,0.000001))
fdouble deadrise0(45*max(deadrise,0.000001))


// Points
point onDWL ([x,dwl,draft])
point keelp([x,0,keel])


// Section
fsplinecurve section()
```

```
// set object attributes
//X - (Y,Z)
section {
 .setPlane("X - (Y,Z)")
 .setPlane("X - (Y,Z)")
 .setStartPos(onDWL)
 .setStartTan(-90-flare0)
 .setEndPos(keelp)
}

// end
```

B.5. NURBS_Underwater



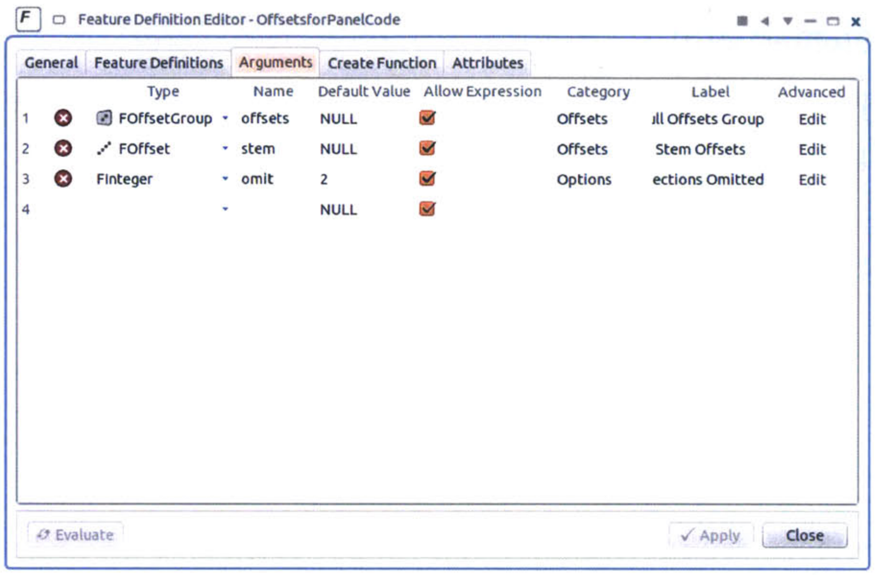
	Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	FDouble	x	0	✓	Positions	X-Position	Edit
2	FDouble	keel	0	✓	Positions	Keel Z-Position	Edit
3	FDouble	deadrise	50	✓	Angles	Angle at Keel	Edit
4	FDouble	flare	12	✓	Angles	Flare Angle	Edit
5	FDouble	dwl	0	✓	Posistions	DWL Y-Position	Edit
6	FDouble	draft	0	✓	Positions	Draft	Edit
7	FDouble	fullness	1	✓	Dmnless	Fullness of Section	Edit
8			NULL	✓			

Figure 76 - Arguments of NURBS_Underwater Feature Definition

```
// Created by K. Nestoras
// This feature definition is used to define the NURBS curve used by
// the curve engine that defines the most part of the underwater side surface
// of the hull.
// For numerical reasons, the flare angle and the deadrise angle should not get zero values
// multiply by 45 degrees as the control curves are non-dimensional
fdouble flare0(45*max(flare,0.000001))
fdouble deadrise0(45*max(deadrise,0.000001))



// Points
point onDWL ([x,dwl,draft])
point flarep([x,dwl-((draft-keel)/5)*tan(flare0),draft-(draft-keel)/5])
point deadrisep([x,(dwl/5),(dwl/5)*tan(deadrise0)+keel])
//point
midp([x,3*deadrisep.getglobaly()/2+flarep.getglobaly()/2,(flarep.getglobalz()+deadrisep.ge
tglobalz())/2])
point keelp([x,0,keel])

//Midpoint
fdouble y()
fdouble z()
z=(draft-dwl*tan(90-flare0,0)-(keel*tan(90-
flare0,0)/tan(deadrise0,0)))*tan(deadrise0,0)/(tan(deadrise0,0)-tan(90-flare0,0))
y=(z-keel)/tan(deadrise0,0)
point midp([x,y,z])

//CurveIntersectionPoint midp Visualization
line midp_line(keelp,midp)
line midp_dwl(midp,onDWL)

// Section
nurbscurve section([onDWL,flarep,midp,deadrisep,keelp])
section.setweights([1,1,fullness,1,1])
section.setdegree(3)
```

```
// end
```

B.6. OffsetsForPanelCode


Feature Definition Editor - OffsetsforPanelCode

General | Feature Definitions | Arguments | Create Function | Attributes

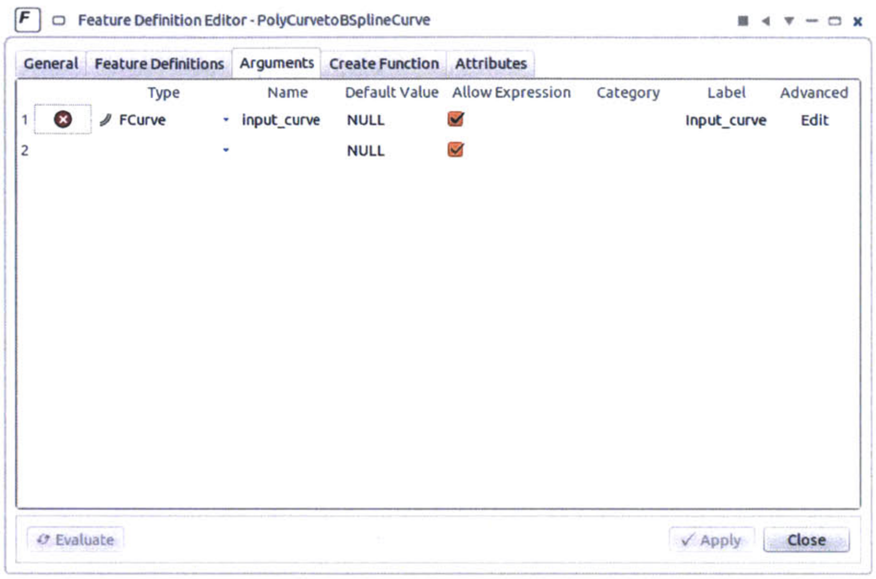
		Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	⊗	FOffsetGroup	offsets	NULL	✓	Offsets	ıll Offsets Group	Edit
2	⊗	FOffset	stem	NULL	✓	Offsets	Stem Offsets	Edit
3	⊗	Finteger	omit	2	✓	Options	ections Omitted	Edit
4				NULL	✓			

Figure 77 - Arguments of OffsetsForPanelCode Feature Definition

```
// Created by K. Nestoras
// Generates the offsets for the input file of prof Brizzolara's Panel Code


string expStr()
finteger numStemOffsets(stem.getsize())
expStr.append(stem.getSize().toString("f",0)+"\n")
loop(numStemOffsets)
 expStr.append(stem.at($$i).getX().toString("f",4)+"     "+stem.at($$i).getY().toString("f",4)+"
"+stem.at($$i).getZ().toString("f",4)+"\n")
endloop


// This subroutine was created by Tobbias from Friendship Framework Team
// Altered by K. Nestoras in order to create the needed format for the output file


unsigned numOffsets(offsets.getSize())
```

```
expStr.append((numoffsets-omit).toString("f",0)+"\n")
fdouble i(omit)
while(i<numOffsets-1)
 foffset tmpOff(offsets.at(i))
 expStr.append(tmpOff.getFirst().getX().toString("f",4)+"    "+tmpOff.getSize()+"\n")
 loop(tmpOff.getSize())
  expStr.append("   "+tmpOff.at($$i).getY().toString("f",4)+"
"+tmpOff.at($$i).getZ().toString("f",4) + "\n")
 endloop
 i=i+1
endwhile

// last offset without "return"
foffset tmpOff(offsets.at(offsets.getsize()-1))

expStr.append(tmpOff.getFirst().getX().toString("f",4)+"    "+tmpOff.getSize()+"\n")
loop(tmpOff.getSize()-1)
expStr.append("   "+tmpOff.at($$i).getY().toString("f",4)+"
"+tmpOff.at($$i).getZ().toString("f",4)+"\n")
endloop

expStr.append("   "+tmpOff.getlast().getY().toString("f",4)+"
"+tmpOff.getlast().getZ().toString("f",4))

// end
```

B.7. PolyCurvetoBSplineCurve

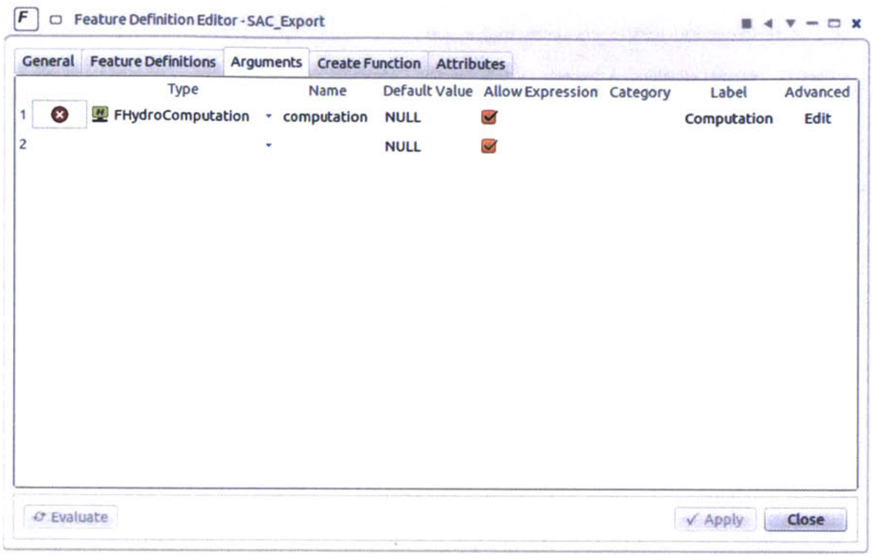


	Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	FCurve	input_curve	NULL	✓		Input_curve	Edit
2			NULL	✓			

Figure 78 - Arguments of PolyCurvetoSBpline Feature Definition

```
// Created by K.Nestoras
// Used in the model to convert a poly-curve to a B-Spline,
// so that the centroid and area of the curve with respect to an axis
// can be computed.

bsplinecurve out_curve(input_curve)

// end
```

B.8. SAC_Export

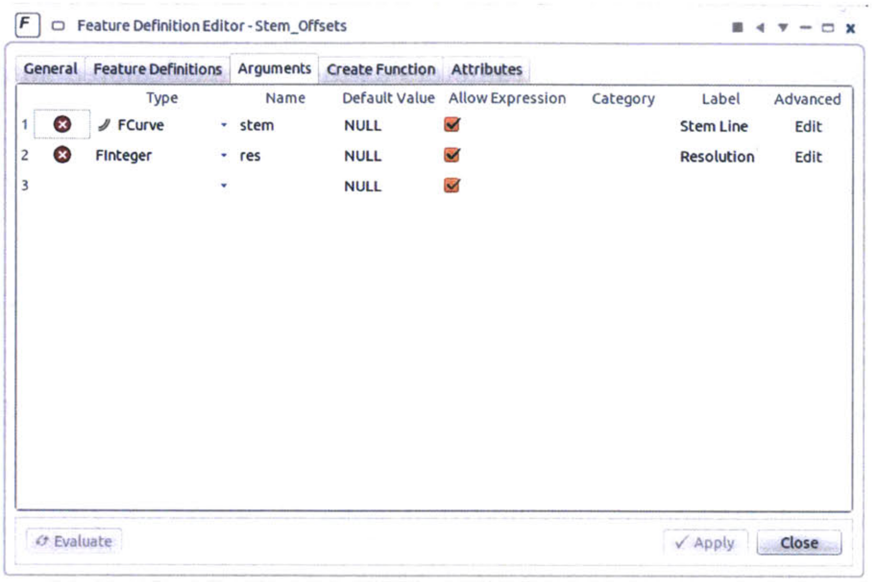


	Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	FHydroComputation	computation	NULL	✓		Computation	Edit
2			NULL	✓			

Figure 79 - Arguments of SAC_Export Feature Definition

```
// Created by K.Nestoras
// Extracts the SAC from hydrostatic computation as a BSpline

bsplinecurve SAC(computation.getresults().getsectionalareacurve())

// end
```

B.9. Stem_Offsets



		Type	Name	Default Value	Allow Expression	Category	Label	Advanced
1	⊗	FCurve	stem	NULL	✓		Stem Line	Edit
2	⊗	FInteger	res	NULL	✓		Resolution	Edit
3				NULL	✓			

Figure 80 - Arguments of CurveToSBpline Feature Definition

```
// Created by K. Nestoras
// Generates offsets for the stem curve, so they can be used by the
// panel code calculations

offset offsets(stem,res).sortbyaxis(2)
clearselection()

// end
```

B.10. values_in_variables.fsc

```
// Created by K. Nestoras
// Reads the values for the desing parameters as they were calculated automatically based on the
// hull to be fitted. Then it sets the right value in the corresponding Design Variables.

|Variables|x_Bmax.setvalue(|Hull_to_be_fitted|Measured|Parameters|x_Bmax)
|Variables|TransomDepth.setvalue(|Hull_to_be_fitted|Measured|Parameters|TransomDepth)
|Variables|TransomAngle.setvalue(|Hull_to_be_fitted|Measured|Parameters|TransomAngle)
|Variables|LBP.setvalue(|Hull_to_be_fitted|Measured|Parameters|LBP)
|Variables|KeelAngleAtTransom.setvalue(|Hull_to_be_fitted|Measured|Parameters|KeeltoTransomAngle)
|Variables|KeelRisePoint.setvalue(|Hull_to_be_fitted|Measured|Parameters|KeelRisePoint)
|Variables|H_BeamTransomOverH_Beam.setvalue(|Hull_to_be_fitted|Measured|Parameters|H_BeamTransomOverH_Beam)
|Variables|Freeboard_MS.setvalue(|Hull_to_be_fitted|Measured|Parameters|Freeboard_MS)
|Variables|Freeboard_FP.setvalue(|Hull_to_be_fitted|Measured|Parameters|FreeboardFWD)
|Variables|EntranceAngle.setvalue(|Hull_to_be_fitted|Measured|Parameters|EntranceAngle)
|Variables|DWLaftAngle.setvalue(|Hull_to_be_fitted|Measured|Parameters|DWLtoTransomAngle)
|Variables|Cwp.setvalue(|Hull_to_be_fitted|Measured|Parameters|Cwp)
|Variables|L_over_Beam.setvalue(|Hull_to_be_fitted|Measured|Parameters|L_over_Beam)
|Variables|Draft.setvalue(|Hull_to_be_fitted|Measured|Parameters|Design_Draft)


//Angles
//Deadrise - Angles at 0.05 and 0.15 have been ommited from this auto-fill because at these positions a sonar might exist
|Variables|Deadrise|0_25.setValue(|Hull_to_be_fitted|Measured|Angles|Deadrise|0_25)
|Variables|Deadrise|0_50.setValue(|Hull_to_be_fitted|Measured|Angles|Deadrise|0_50)
```

```
// Flare Angle
|Variables|Flare|0_05.setValue(|Hull_to_be_fitted|Measured|Angles|Flare|0_05)
|Variables|Flare|0_25.setValue(|Hull_to_be_fitted|Measured|Angles|Flare|0_25)
|Variables|Flare|0_50.setValue(|Hull_to_be_fitted|Measured|Angles|Flare|0_50)
|Variables|Flare|0_75.setValue(|Hull_to_be_fitted|Measured|Angles|Flare|0_75)
|Variables|Flare|1_00.setValue(|Hull_to_be_fitted|Measured|Angles|Flare|1_00)
```

B.11. DDG51-eng-variables.fsc

```
|Variables|Cr_Section_Change.setValue(0.2)
|Variables|Cwp.setValue(0.81650217)
|Variables|DWLaftAngle.setValue(9.56081483)
|Variables|Draft.setValue(6.12300886)
|Variables|EntranceAngle.setValue(12.36278325)
|Variables|H_BeamTransomOverH_Beam.setValue(0.55231841)
|Variables|KeelAngleAtTransom.setValue(4.1008982)
|Variables|KeelRisePoint.setValue(0.65)
|Variables|LBP.setValue(141.27697962)
|Variables|L_Over_Beam.setValue(8.85725154)
|Variables|StemAngle.setValue(63)
|Variables|StemRadiusOverDraft.setValue(1.5)
|Variables|TransomAngle.setValue(17.59566994)
|Variables|TransomDepth.setValue(1.01739948)
|Variables|x_Bmax.setValue(0.45200275)

|Variables|Deadrise|0_05.setValue(60)
|Variables|Deadrise|0_15.setValue(30)
|Variables|Deadrise|0_25.setValue(9)
|Variables|Deadrise|0_50.setValue(1.5)

|Variables|Flare|0_05.setValue(18)
|Variables|Flare|0_25.setValue(22)
|Variables|Flare|0_50.setValue(8)
|Variables|Flare|0_75.setValue(13)
|Variables|Flare|1_00.setValue(35)

|Variables|FullnessFactoR|0_00.setValue(1)
|Variables|FullnessFactoR|0_25.setValue(0.6)
```

```
|Variables|FullnessFactoR|0_50.setValue(1)
|Variables|FullnessFactoR|0_75.setValue(0.4)
|Variables|FullnessFactoR|1_00.setValue(1.2)

|Variables|Auxiliary|DeckEdge|OffsetAngleFWD.setValue(34)
|Variables|Auxiliary|DeckEdge|OffsetAngleMidShips.setValue(10.2)
|Variables|Auxiliary|DeckEdge|OffsetAngleTransom.setValue(20)
|Variables|Auxiliary|Freeboard_FP.setValue(7.95922998)
|Variables|Auxiliary|Freeboard_MS.setValue(5.1770332)

|Variables|InternalArrangements|UseObj1.setValue(1)
|Variables|InternalArrangements|UseObj2.setValue(1)
|Variables|InternalArrangements|UseObj3.setValue(1)
|Variables|InternalArrangements|UseObj4.setValue(1)
|Variables|InternalArrangements|UseObj5.setValue(0)
|Variables|InternalArrangements|UseObj6.setValue(0)

|Variables|InternalArrangements|Obj1|FWDPortBottomPoint|x.setValue(42.93)
|Variables|InternalArrangements|Obj1|FWDPortBottomPoint|y.setvalue(0)
|Variables|InternalArrangements|Obj1|FWDPortBottomPoint|z.setValue(1.67)
|Variables|InternalArrangements|Obj1|Angle.setvalue(0)
|Variables|InternalArrangements|Obj1|Height.setValue(2.8)
|Variables|InternalArrangements|Obj1|Length.setValue(4.87)
|Variables|InternalArrangements|Obj1|Width.setValue(1.155)

|Variables|InternalArrangements|Obj2|FWDPortBottomPoint|x.setValue(53.94)
|Variables|InternalArrangements|Obj2|FWDPortBottomPoint|y.setvalue(4.58)
|Variables|InternalArrangements|Obj2|FWDPortBottomPoint|z.setValue(4.55)
|Variables|InternalArrangements|Obj2|Angle.setvalue(2.87)
|Variables|InternalArrangements|Obj2|Height.setValue(2.74)
|Variables|InternalArrangements|Obj2|Length.setValue(8.1)
```

```
|Variables|InternalArrangements|Obj2|Width.setValue(2.95)

|Variables|InternalArrangements|Obj3|FWDPortBottomPoint|x.setValue(81.67)
|Variables|InternalArrangements|Obj3|FWDPortBottomPoint|y.setvalue(4.58)
|Variables|InternalArrangements|Obj3|FWDPortBottomPoint|z.setValue(4.67)
|Variables|InternalArrangements|Obj3|Angle.setvalue(4.52)
|Variables|InternalArrangements|Obj3|Height.setValue(2.74)
|Variables|InternalArrangements|Obj3|Length.setValue(8.1)
|Variables|InternalArrangements|Obj3|Width.setValue(2.95)

|Variables|InternalArrangements|Obj4|FWDPortBottomPoint|x.setValue(118.12)
|Variables|InternalArrangements|Obj4|FWDPortBottomPoint|y.setvalue(0)
|Variables|InternalArrangements|Obj4|FWDPortBottomPoint|z.setValue(4.68)
|Variables|InternalArrangements|Obj4|Angle.setvalue(0)
|Variables|InternalArrangements|Obj4|Height.setValue(2.8)
|Variables|InternalArrangements|Obj4|Length.setValue(4.87)
|Variables|InternalArrangements|Obj4|Width.setValue(1.155)

|Variables|InternalArrangements|Obj5|FWDPortBottomPoint|x.setValue(118.12)
|Variables|InternalArrangements|Obj5|FWDPortBottomPoint|y.setvalue(0)
|Variables|InternalArrangements|Obj5|FWDPortBottomPoint|z.setValue(4.68)
|Variables|InternalArrangements|Obj5|Angle.setvalue(0)
|Variables|InternalArrangements|Obj5|Height.setValue(2.8)
|Variables|InternalArrangements|Obj5|Length.setValue(4.87)
|Variables|InternalArrangements|Obj5|Width.setValue(1.155)

|Variables|InternalArrangements|Obj6|FWDPortBottomPoint|x.setValue(118.12)
|Variables|InternalArrangements|Obj6|FWDPortBottomPoint|y.setvalue(0)
|Variables|InternalArrangements|Obj6|FWDPortBottomPoint|z.setValue(4.68)
|Variables|InternalArrangements|Obj6|Angle.setvalue(0)
|Variables|InternalArrangements|Obj6|Height.setValue(2.8)
```

```
|Variables|InternalArrangements|Obj6|Length.setValue(4.87)
|Variables|InternalArrangements|Obj6|Width.setValue(1.155)
```

B.12. In_batch.fsc

In order to execute this script the following command should be given in the terminal and from inside the folder, where the script is contained.

Command: "friendship_crt –float In_batch.fsc"

```
// Created by K. Nestoras.
// Can be used to alter all the variables in the model, in order to generate geometries
// in batch mode.
// All values should be given in the SI system, except otherwise noted

// Open the project. The absolute path should be used!!
openproject("/home/friendship/Programs/Friendship/projects/WrapAround_v1.fdb")

|Variables|Cwp.setValue(0.81650217)
|Variables|Draft.setValue(6.12300886)
|Variables|LBP.setValue(141.27697962)
|Variables|EntranceAngle.setValue(12.36278325)
|Variables|L_Over_Beam.setValue(8.85725154)

// Position where definition of underwater cross-section changes
// from F-Spline to NURBS
|Variables|Cr_Section_Change.setValue(0.2)

// Angle of the DWL at the meeting point with the transom [deg]
|Variables|DWLaftAngle.setValue(9.56081483)

//Beam @ transom over Max Beam. Non dimensional
|Variables|H_BeamTransomOverH_Beam.setValue(0.55231841)
```

```
// Angle of the keel at the meeting point with the transom [deg]
|Variables|KeelAngleAtTransom.setValue(4.1008982)

// Point where the keel starts to rise from the baseline as % of LBP
|Variables|KeelRisePoint.setValue(0.65)

// Variables controlling the shape of the stem curve
|Variables|StemAngle.setValue(63)
|Variables|StemRadiusOverDraft.setValue(1.5)

// submerged part of transom
|Variables|TransomAngle.setValue(17.59566994)
|Variables|TransomDepth.setValue(1.01739948)

// position of max breadth, given as % of LBP
|Variables|x_Bmax.setValue(0.45200275)

// Auxiliary Variables

// Deck edge shape
|Variables|Auxiliary|DeckEdge|OffsetAngleFWD.setValue(34)
|Variables|Auxiliary|DeckEdge|OffsetAngleMidShips.setValue(10.2)
|Variables|Auxiliary|DeckEdge|OffsetAngleTransom.setValue(20)

// Freaboard at MS and at FP
|Variables|Auxiliary|Freeboard_FP.setValue(7.95922998)
|Variables|Auxiliary|Freeboard_MS.setValue(5.1770332)

// Distribution of Deadrise Angle along the length. The desired
// Deadrise angle [deg] should be given at .05, .15, .25 and .50 of LBP
|Variables|Deadrise|0_05.setValue(60)
```

```
|Variables|Deadrise|0_15.setValue(30)
|Variables|Deadrise|0_25.setValue(9)
|Variables|Deadrise|0_50.setValue(1.5)

// Distribution of Flare Angle along the length. The desired Flare
// angle [deg] should be given at .05, .25, .50, .75 and 1.0 of LBP
|Variables|Flare|0_05.setValue(18)
|Variables|Flare|0_25.setValue(22)
|Variables|Flare|0_50.setValue(8)
|Variables|Flare|0_75.setValue(13)
|Variables|Flare|1_00.setValue(35)

// Distribution of cross section fullness along the length. The
// desired fullness should be given at .25, .50, .75 and 1.0 of LBP
|Variables|FullnessFactoR|0_00.setValue(1)
|Variables|FullnessFactoR|0_25.setValue(0.6)
|Variables|FullnessFactoR|0_50.setValue(1)
|Variables|FullnessFactoR|0_75.setValue(0.4)
|Variables|FullnessFactoR|1_00.setValue(1.2)

// Definition of Internal Arrangements
// x,y and z positions are for the FWD, lower and inboard corner
// of each machinery box.
// When UseObj# is set to zero, then the object is not generated
// and the values for angle, dimensions and x,y,z positions
// are not taken into consideration

// Object 1
|Variables|InternalArrangements|UseObj1.setValue(1)
|Variables|InternalArrangements|Obj1|FWDPortBottomPoint|x.setValue(42.93)
|Variables|InternalArrangements|Obj1|FWDPortBottomPoint|y.setvalue(0)
```

```
|Variables|InternalArrangements|Obj1|FWDPortBottomPoint|z.setValue(1.67)
|Variables|InternalArrangements|Obj1|Angle.setvalue(0)
|Variables|InternalArrangements|Obj1|Height.setValue(2.8)
|Variables|InternalArrangements|Obj1|Length.setValue(4.87)
|Variables|InternalArrangements|Obj1|Width.setValue(1.155)

// Object 2
|Variables|InternalArrangements|UseObj2.setValue(1)
|Variables|InternalArrangements|Obj2|FWDPortBottomPoint|x.setValue(53.94)
|Variables|InternalArrangements|Obj2|FWDPortBottomPoint|y.setvalue(4.58)
|Variables|InternalArrangements|Obj2|FWDPortBottomPoint|z.setValue(4.55)
|Variables|InternalArrangements|Obj2|Angle.setvalue(2.87)
|Variables|InternalArrangements|Obj2|Height.setValue(2.74)
|Variables|InternalArrangements|Obj2|Length.setValue(8.1)
|Variables|InternalArrangements|Obj2|Width.setValue(2.95)

// Object 3
|Variables|InternalArrangements|UseObj3.setValue(1)
|Variables|InternalArrangements|Obj3|FWDPortBottomPoint|x.setValue(81.67)
|Variables|InternalArrangements|Obj3|FWDPortBottomPoint|y.setvalue(4.58)
|Variables|InternalArrangements|Obj3|FWDPortBottomPoint|z.setValue(4.67)
|Variables|InternalArrangements|Obj3|Angle.setvalue(4.52)
|Variables|InternalArrangements|Obj3|Height.setValue(2.74)
|Variables|InternalArrangements|Obj3|Length.setValue(8.1)
|Variables|InternalArrangements|Obj3|Width.setValue(2.95)

// Object 4
|Variables|InternalArrangements|UseObj4.setValue(1)
|Variables|InternalArrangements|Obj4|FWDPortBottomPoint|x.setValue(118.12)
|Variables|InternalArrangements|Obj4|FWDPortBottomPoint|y.setvalue(0)
|Variables|InternalArrangements|Obj4|FWDPortBottomPoint|z.setValue(4.68)
```

```
|Variables|InternalArrangements|Obj4|Angle.setvalue(0)
|Variables|InternalArrangements|Obj4|Height.setValue(2.8)
|Variables|InternalArrangements|Obj4|Length.setValue(4.87)
|Variables|InternalArrangements|Obj4|Width.setValue(1.155)

// Object 5
|Variables|InternalArrangements|UseObj5.setValue(0)
|Variables|InternalArrangements|Obj5|FWDPortBottomPoint|x.setValue(118.12)
|Variables|InternalArrangements|Obj5|FWDPortBottomPoint|y.setvalue(0)
|Variables|InternalArrangements|Obj5|FWDPortBottomPoint|z.setValue(4.68)
|Variables|InternalArrangements|Obj5|Angle.setvalue(0)
|Variables|InternalArrangements|Obj5|Height.setValue(2.8)
|Variables|InternalArrangements|Obj5|Length.setValue(4.87)
|Variables|InternalArrangements|Obj5|Width.setValue(1.155)

// Object 6
|Variables|InternalArrangements|UseObj6.setValue(0)
|Variables|InternalArrangements|Obj6|FWDPortBottomPoint|x.setValue(118.12)
|Variables|InternalArrangements|Obj6|FWDPortBottomPoint|y.setvalue(0)
|Variables|InternalArrangements|Obj6|FWDPortBottomPoint|z.setValue(4.68)
|Variables|InternalArrangements|Obj6|Angle.setvalue(0)
|Variables|InternalArrangements|Obj6|Height.setValue(2.8)
|Variables|InternalArrangements|Obj6|Length.setValue(4.87)
|Variables|InternalArrangements|Obj6|Width.setValue(1.155)

//Save
//saveproject()

//SaveAs
//saveprojectas("enterpath and name of file")
```

```
// Export surfaces of ship
// Both Sides (full ship)
|Export|Ship_exportFULLsurfaces.exportiges("/home/friendship/Programs/Friendship/pro
jects/WrapAround_v1/Ship-Full.iges")
// or
//
|Export|Ship_exportFULLsurfaces.exportstl("/home/friendship/Programs/Friendship/proje
cts/WrapAround_v1/Ship-Full.stl")

// Starboard Side (half ship)
|Export|Ship_exportHALFsurfaces.exportiges("/home/friendship/Programs/Friendship/pro
jects/WrapAround_v1/Ship-Half.iges")
// or
//|Export|Ship_exportHALFsurfaces.exportstl("/home/friendship/Programs/Friendship/pr
ojects/WrapAround_v1/Ship-Half.stl")

// Export surfaces of model

// Set the scale
|Export|Model_Scale.setvalue(30)

// ***!!! CAUTION !!!***
// The exported model is being translated to a reference system where
// x=0 is at MS, z=0 is at the DWL and y=0 is at the symmetry plane

// Both Sides (full ship)
|Export|Model_exportFULLsurfaces.exportiges("/home/friendship/Programs/Friendship/p
rojects/WrapAround_v1/Model-Full.iges")
// or
```

```
//
|Export|Model_exportFULLsurfaces.exportstl("/home/friendship/Programs/Friendship/pro
jects/WrapAround_v1/Model-Full.stl")

// Starboard Side (half ship)
|Export|Model_exportHALFsurfaces.exportiges("/home/friendship/Programs/Friendship/p
rojects/WrapAround_v1/Model-Half.iges")
// or
//|Export|Model_exportHALFsurfaces.exportstl("/home/friendship/Programs/Friendship/
projects/WrapAround_v1/Model-Half.stl")

//Close project,
closeproject()

//exit friendship
exit()
```