Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Optimal Thrust Allocation Methods for Dynamic Positioning of Ships

A thesis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

by

CHRISTIAAN DE WIT

Delft, the Netherlands
July 2009

MSc THESIS APPLIED MATHEMATICS

"Optimal Thrust Allocation Methods for Dynamic Positioning of Ships"

CHRISTIAAN DE WIT

Delft University of Technology

Daily supervisor

Dr. J.W. van der Woude

Responsible professor

Prof. dr. K.I. Aardal

Other thesis committee members

M. Gachet, Ingenieur ENSTA

Dr. R.J. Fokkink

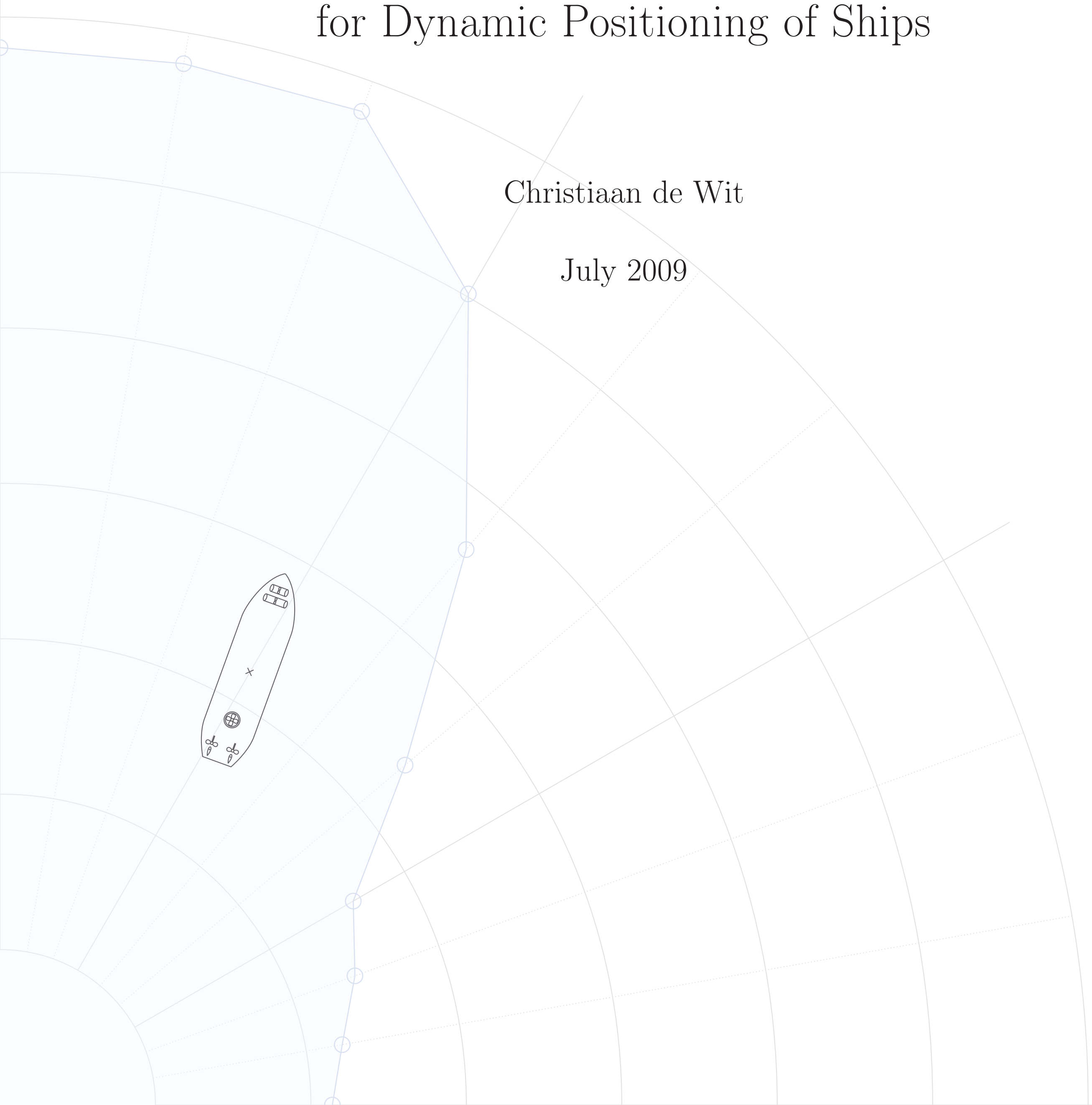
July 2009

Delft, the Netherlands

Optimal Thrust Allocation Methods for Dynamic Positioning of Ships

Christiaan de Wit

July 2009



Abstract

The first *Dynamic Positioning* (DP) systems emerged in the 1960's from the need for deep water drilling by the offshore oil and gas industry, as conventional mooring systems, like a jack-up barge or an anchored rig, can only be used in shallow waters. GustoMSC has been developing DP drill ships since the early 1970's and it is still one of their core businesses.

DP systems automatically control the position and heading of a ship subjected to environmental and external forces, using its own actuators. The thrust allocator of a DP system is responsible for the thrust distribution over the actuators of the ship. Apart from minimizing the power consumption an ideal thrust allocator can also take other aspects into account, such as forbidden/spoil zones and thruster relations. Because the Lagrange multiplier method, used inhouse by GustoMSC for thrust allocation, cannot accurately describe fullscale DP systems with rudders and forbidden/spoil zones, new methods need to be explored.

Various optimal thrust allocation methods for dynamic positioning of ships are considered and their practical use is tested with DP capability calculations and time domain simulations with online optimization routines. The shortcomings of Lagrange multiplier methods are illustrated and quadratic programming methods combined with disjunctive programming techniques are used to present a more elaborate solution to optimal thrust allocation problems. Using disjunctive programming each actuator is modeled by a finite union of convex polygons representing the attainable thrust region. This approach allows combinations of non-rotatable thrusters, rotatable azimuth thrusters with forbidden/spoil zones and main propeller/rudder pairs to be used. As a consequence the allocation problem decomposes into a finite number of subproblems that all need to be solved separately in order to find the optimal solution of the main problem. For time domain simulations the dynamic limitations of the thrusters are taken into account by adding more convex thrust regions to the problem. Briefly, the potential use of linear matrix inequalities for optimal thrust allocation problems is treated.

The obtained results are discussed and conclusions are given.

Preface

This report concludes my Master Thesis project for the MSc programme Applied Mathematics at the Delft University of Technology. This project was carried out in a period of 8 months at GustoMSC in Schiedam, Holland. During this period a thrust allocation algorithm based on quadratic programming techniques was developed for use with Dynamic Positioning systems.

I would like to thank my supervisors Dr. J.W. van der Woude from the Delft University of Technology and M. Gachet from GustoMSC who made it possible for me to graduate and who were always there to help me. Also I would like to thank all the colleagues at GustoMSC who helped me during my master thesis work. I really appreciate your dedicated and professional advise.

Furthermore I want to thank the other members from the thesis committee, Prof. dr. K.I. Aardal and Dr. R.J. Fokkink, for their involvement, time and effort.

Last but not least I would like to thank all my friends that supported me during this sometimes stressful period. I really appreciate all your support.

Christiaan de Wit

July, 2009

Contents

Chapter 1

Introduction

Offshore drilling dates back to the mid 1920's when the first subsea wells were drilled. Starting at tidal zones and piers, the first drilling activities soon occurred from concrete platforms near the shore. In the 1940's fixed drilling/production platforms allowed drilling at a water depth of 6 meters, tens of kilometers off the coast. Keeping a fixed position in these shallow waters obviously never was a problem, but when the demand for deep water drilling increased in the 1950's station keeping became a big obstacle. This resulted in different positioning solutions.

A jack-up barge can be used in water depths up to approximately 120m. When it is on location, it can raise itself clear of the sea with its three or more massive legs. Figure 1.1 shows an installed jack-up rig. Also known as a mobile offshore drilling unit (MODU) it has the benefits of a fixed platform, combined with the ease of mobility. A big advantage of this system is that the station keeping is not vulnerable to blackouts or power shortages and there is no need for a position reference system, once on location. Despite these advantages the maximum water depth at which it can operate is still very limiting compared to the other mooring solutions.

Figure 1.1: A jack-up rig showing its three enormous legs.

Spread mooring and anchor pattern like systems can be used for many different structure types in water depths exceeding 1000m. The position is controlled by fixing the vessel to the seabed using mooring lines and anchors. Positioning therefore takes up a lot of time and can be quite expensive due to the required anchor-handling tugs. Once anchored, there will always be some movement left due to the flexibility of the mooring lines, although small adjustments are possible by adjusting the line lengths. When a large position shift is needed, all or some of the anchors will need to be lifted and relaid. Then there is also the possibility of underwater hazards represented by any existing underwater installation, such as pipelines. Regarding the station keeping vulnerabilities is has the same advantages as the jack-up system. Figure 1.2 shows a spread turret mooring system.

Figure 1.2: Example of spread turret mooring.

Dynamic positioning (DP) systems are not limited to a maximal water depth as they automatically control the position and heading of a vessel by using its own propulsion system (see figure 1.3). Although this gives a lot of freedom, it also makes DP systems relatively complex. Because the propulsion system needs to react to environmental/external changes continuously, DP systems can be quite expensive. Also this online approach brings more reliability problems, as the system is more vulnerable to failures regarding the power supply, thrusters, electronics or the reference system. On the other hand, DP systems provide a solution that can be used at any water depth (only excluding some shallow waters), are very precise because they can rapidly response to environmental/external changes and they are set up very quickly and easily. No assisting tugs are required whatsoever and a DP ship can easily change to another location without a lot of extra costs. Also underwater installations form no obstacle, as it only relies on its own actuators for station keeping. Because station keeping can also be relative, DP can also be used for Dynamic Tracking (DT) purposes, where the system can follow a predefined track, or where it can maintain position relative to another moving vessel. Nowadays, almost every floating structure can be quickly fitted with a DP system (as there are also mobile DP solutions available). All these advantages results in DP systems being used in an increasing number of applications, including exploration drilling, cable/pipe laying, dredging, shuttle tanker operations, rocket launch pads, repair/maintenance support, crane vessels. Despite all the advantages, there is still room for improvements. Redundancy has already made

DP systems more reliable, but the growing demand for more fuel efficient operation and the need for more advanced control over the thrusters, calls for the development of new control techniques.

Figure 1.3: Example of a DP vessel, showing the wash of the propulsors.

Chapter 2

Dynamic positioning

In this chapter the fundamental working of a DP system will be explained, while giving an overview of its most essential components. DP systems require many different components to communicate correctly with each other in order to work and can therefore be very complex. Because DP suppliers have different philosophies regarding their design, many variations exist. Despite of these variations, certain components are essential for all DP systems. In the following sections each of these components will be described. For more detailed information om DP system components, see [Bra03].

2.1 Position/Heading reference systems

Every DP system needs to measure the state of the system, containing the current position and heading of the ship. As this is very important for the correct functioning of the DP system, there are usually multiple devices on board that can generate this information. There are also many different systems to choose from. Commonly used are satellite navigation systems (GPS and DGPS), hydro acoustic position reference systems (HPR), mechanical reference systems (taut wire), relative positioning systems (Artemis and Fanbeam/CyScan), vertical reference units (VRU) and heading reference systems (gyrocompass). More accuracy can be achieved by combining different methods. A DP system also has a mathematical model of the ship from which the position and heading can be estimated given its current state. This can help to keep the DP system on course when a reference system failure would occur.

2.2 Sensors: measuring the environment

The three most important environmental factors for a DP system are the wind, the waves and the current. From these, the wind is the most important factor because the waves are assumed to be wind driven and the current is often slowly varying. The wind is also the most easiest to measure and from this, wind-wave relations can be used to estimate the waves. Because the DP system cannot react to the high frequency wave loads, only the second order wave-drift forces are of interest. Using a Kalman filter these can be filtered from the measured motions (see also [Sni05]).

2.3 Propulsion system

Actuators can exert forces on the ship and are used by the DP system to control its position. Most propulsion devices use a rotating propeller to generate thrust, although there are also jet propulsion devices. Controlling the amount of thrust that is generated by an actuator can be a difficult task. Often the rpm is used for this, although there are many other factors that influence the amount of generated thrust. Controllable Pitch (CP) propellers (as opposed to Fixed Pitch (FP) propellers) can be used to increase the efficiency and wear/tear of the motor. By controlling the pitch angle of the blades on the propeller, the amount of thrust can be controlled, while the motor can be left at a fixed efficient state (constant rpm). Other factors that can influence the efficiency, and the generated amount of thrust, are thruster-thruster interactions (thrusters blowing against each other), thruster-hull interactions, cavitation phenomena, and strong current. The ideal DP system should be able to take all these factors into account when controlling the ship.

Apart from the correct thruster type, the actuator layout is also very important and greatly determines the DP capability of the ship. When the thrusters are badly positioned, an under-actuated ship can be the result. This is bad because the ship will not be able to maintain position under certain conditions. The aim in designing a DP system is to have a fully-actuated ship. For more critical DP applications an over-actuated ship will be preferred. In this case DP problems will have multiple feasible solutions, resulting in a more robust DP system. Figure 2.1 shows an example actuator layout. Table 2.1 lists the different thruster types and their corresponding symbol.

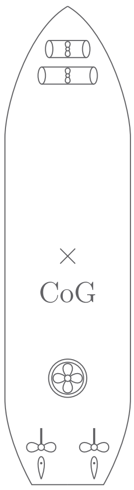


Figure 2.1: Example actuator layout.

Symbol	Description
	Tunnel thruster
	Azimuth thruster
	Main propeller
	Rudder

Table 2.1: Thruster symbol list.

Because there are a lot of different thrusters available for different application, the most basic types will be briefly discussed, explaining their key features (see also figure 2.2).

(a) Two tunnel thrusters at the bow.

(b) Close-up of a tunnel thruster.

(c) An azimuth thruster at the stern of a ship.

(d) The stern of a ship, showing the main propeller with the rudder and a tunnel thruster.

Figure 2.2: Examples of different actuator types.

2.3.1 Tunnel thrusters

Tunnel thrusters (see figure 2.2(a), 2.2(b) and 2.2(d)) can generate transverse thrust. They are usually placed at the bow or the stern of the ship, where they can generate the greatest moment. Because they are primarily used for steering purposes (and not for main propulsion purposes), they often have controllable pitch propellers as they need to be able to switch quickly from forward to reverse mode and vice versa. Multiple tunnel thrusters can be placed next to each other when more thrust is needed.

2.3.2 Azimuth thrusters

Azimuth thrusters (see figure 2.2(c)) can be used for steering and propulsion purposes. They can rotate the full 360 degrees to generate thrust in any direction. Because they are often fitted with a duct around the propeller to reduce cavitation and increase the efficiency, they are usually not used in reverse mode. Although they can be maneuvered in any direction this can sometimes also form a hazard when they are positioned near a moonpool where divers can enter the water. In these situations they will be switched off for safety reasons. Forbidden/spoil zones can be used to avoid an azimuth thruster from blowing at the skeg (a sternward extension of the keel of a ship) or to avoid thruster-thruster interactions, which will reduce the efficiency.

2.3.3 Main propellers and rudders

Main propellers are usually designed to be very efficient under certain conditions. They are not always part of the DP system, as they will be mainly used for the propulsion of the ship. Modern DP systems can use them to increase the DP capability, but estimating the exact capability of a main propeller/rudder pair can be very difficult. Often the characteristics of the rudder (which can also be seen as a wing) are given by lift and drag coefficients. These can be transformed into lift and drag curves in percentage of bollard pull, but then there are still a lot of important factors not taken into account, such as the hull shape that influences the flow and for instance lift induced thrust, generated by the rudder. Figure 2.2(d) shows a main propeller and a rudder at the stern of a ship.

2.4 Power system

The power system often consists of more than one power generator and some switchboards that distribute the power over the ship components. When a generator would fail the switchboard can switch to another generator to maintain power to certain vital components. Because DP systems are very vulnerable to power failures the power management system is very important and should be very robust.

2.5 Control system

The DP control system forms the brain of the whole DP system. Here all the information comes together from which the control signals are determined that will be sent to the actuators. Figure 2.3 shows a block diagram of a DP control system. First the observer (Kalman filter) estimates the current state from the generated thrust and the measurement data. Then the controller calculates the required forces and moment from the current state and the set target state. This information is sent to the thrust allocator, which distributes thrust over the actuators, trying to satisfy the given requirements as good as possible. While doing this the allocator also tries to minimize the power consumption and takes forbidden/spoil zones into account. It could even take thruster-thruster interactions and thruster-hull interactions into account while distributing the thrust.

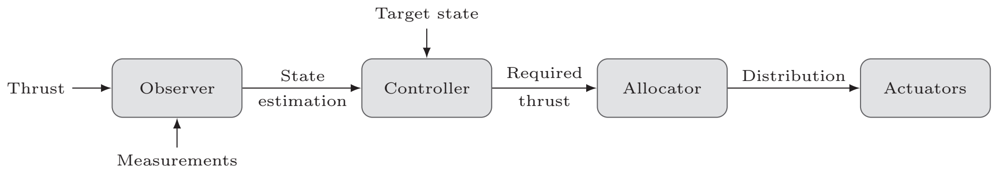


Figure 2.3: Block diagram showing how the thrust allocator is embedded in the DP system.

2.6 Thesis goal

The problem that the thrust allocator has to solve can become quite difficult and forms the main subject of this thesis. The Lagrange multiplier method used inhouse by GustoMSC for thrust allocation, is not able to do DP calculations for azimuth thrusters with forbidden/spoil zones and propeller/rudder pairs. Therefore a new and better thrust allocation algorithm needs to be developed. Given the required forces and moment and the ship layout with the needed actuator information, the new thrust allocator should be able to do full scale DP calculations, including tunnel (fixed) thrusters, azimuthing thrusters with forbidden/spoil zones and main propeller/rudder pairs. While doing this it should also minimize the power consumption.

Chapter 3

General model assumptions

In this chapter general model assumptions will be given. These will be used for the Lagrange multiplier method and the Quadratic Programming method.

3.1 Required thrust

The thrust allocation problem will be restricted to the motions in the horizontal plane of the vessel, leaving only surge, sway and yaw as degrees of freedom (these are controlled by the forces F_x, F_y and the moment M_z). Figure 3.1 shows the sign conventions for F_x, F_y and M_z (Note that the coordinate system is different from the typically used Cartesian coordinate system). The thrust allocator will be given the forces and moment required by the DP system (see also figure 2.3). These are defined as

$$\tau_{\text{ref}} = \begin{bmatrix} F_{x,\text{ref}}, & F_{y,\text{ref}}, & M_{z,\text{ref}} \end{bmatrix}^T . \tag{3.1}$$

3.2 Actuators

The state of each thruster is given by a thrust vector $u = (u_x, u_y)$. All thrusters will be modeled using Cartesian coordinates [1]. All thrusters have the same amount of state variables (two), but as each thruster type has its own thrust characteristics some of the parameters will have a slightly different meaning for different thruster types. Figure 3.2 gives a schematic overview of the different thruster types.

[1] Although it is possible to use polar coordinates for fixed non-rotatable thrusters (by incorporating the constant angle in the configuration matrix), using the same coordinate system for all the thrusters seems less confusing and will only cost one additional constraint for each fixed thruster.

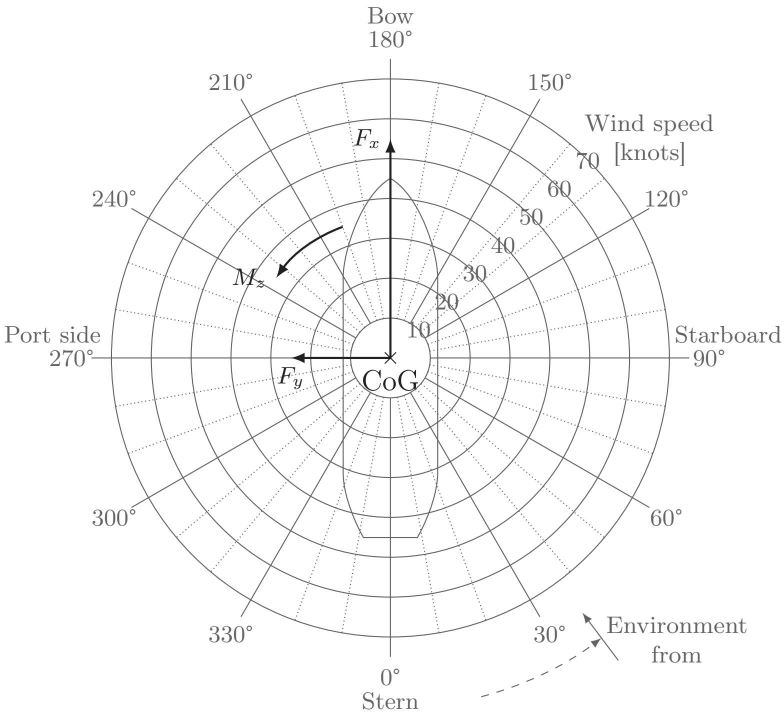
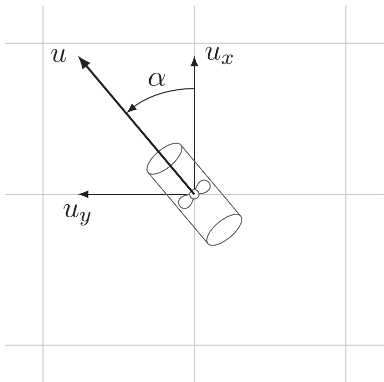
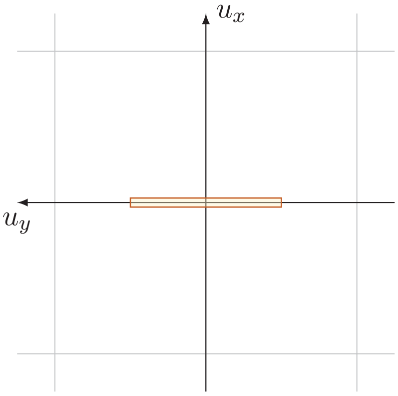


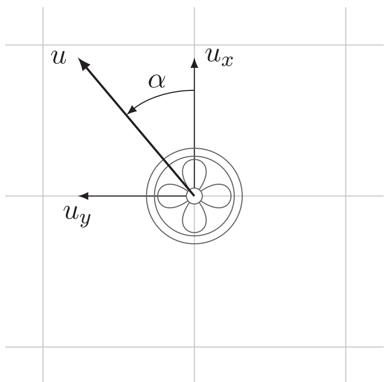
Figure 3.1: Sign conventions.



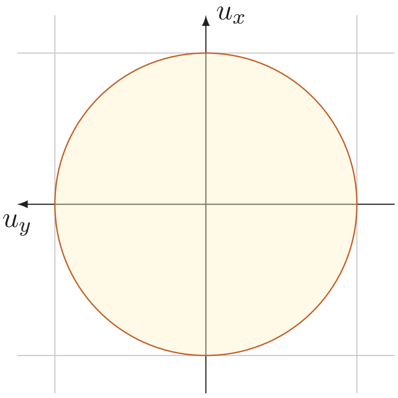
(a) Fixed thruster model.



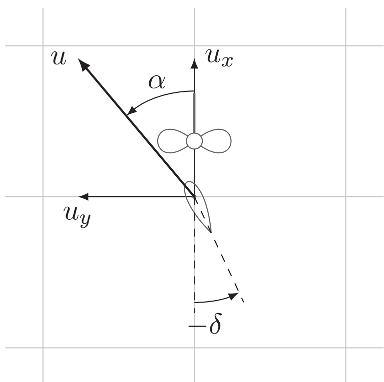
(b) Schematic thrust region for a fixed thruster.



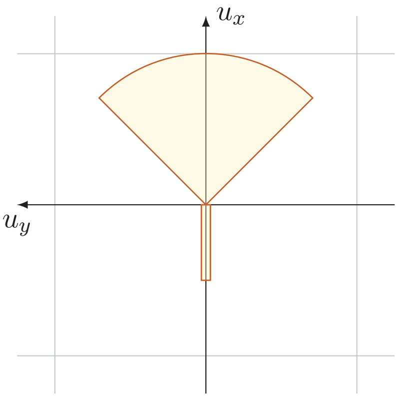
(c) Azimuth thruster model.



(d) Schematic thrust region for an azimuth thruster.



(e) Main propeller & rudder model.



(f) Schematic thrust region for a propeller/rudder pair.

Figure 3.2: Thruster models.

3.2.1 Thrust region

Each thruster has a limited working area, regarding its (local) state (u_x, u_y) (limited amount of thrust and limited directions in which to generate thrust). For each thruster we can define a set that contains all its physical realizable states (u_x, u_y). We will call this the *thrust region* of the thruster. Each thrust region forms a limited and closed subset of $\mathbb{R}^2$ that will be linearly approximated by polygons. Different thruster types will have different thrust region shapes (polygons), as will be clear at the end of this chapter (see also 3.2).

3.2.2 Fixed thruster

A fixed thruster is a non-rotatable device. Therefore the orientation angle α is fixed and cannot change. The generated thrust is thus limited to a line shaped region (the line through zero, with angle α). The maximal and minimal thrust are given respectively by $T_{\max}$ and $T_{\min}$ (in kN). These restrict the thrust region to a line segment. For a tunnel thruster α will be 90°and typically $T_{\max} = -T_{\min} = T_{\text{lim}}$ for some value of T_{lim}.

3.2.3 Azimuth thruster

An azimuth thruster is a rotatable device and can deliver thrust in any direction. The angle of the generated thrust is represented by α and is variable. For a given state (u_x, u_y), we find that $\alpha = \arctan(u_y/u_x)$. Assuming that an azimuth thruster will not be used in reverse mode, the minimal thrust $T_{\min}$ is 0. The maximal thrust $T_{\max}$ will be limited, thus creating a circular thrust region. When a forbidden zone is defined the thrust region will take a Pacman-like shape. More complex shapes are of course also possible.

3.2.4 Main propeller with rudder

The main propeller is always pointing to the stern of the ship and can provide a thrust T_{main} in the F_x direction only (see figure 3.3(a)). Because the main propeller generates no force in the F_y direction, we may shift the force T_{main} over an imaginary line (y is constant), without changing the resulting moment. This way we can model the main propeller/rudder pair as one thruster. The turning point of the rudder will function as the position of the combined propeller/rudder thruster (see figure 3.3(a)).

A rudder with a fixed propeller is defined by the bollard pull of the main propeller and the lift and drag curves of the rudder. The main propeller generates a force T_{main}. The rudder behind the main propeller is able to rotate and generates lift and drag. Lift and drag curves define the lift and drag forces in percentage of the bollard pull T_0 of the main propeller, as a function of the rudder angle δ.

It can be seen from the lift and drag curves that the lift force generated by the rudder will eventually stall at an angle around 35°. Rotating a rudder any further will decrease the generated lift force. The angle at which this happens is called the stall angle and it defines the limiting angle at which the rudder should be used, to avoid inefficiency.

The thrust region from a main propeller/rudder pair is determined by the lift and drag curves, and will have a circle sector like shape. Note that the angle of the generated thrust need not be the same as the angle of the rudder. This is due to the lift and drag characteristics of the rudder. Using the lift and drag curves, the thrust angle α can be translated to and from the rudder angle δ, by using interpolation algorithms on the quotient of the two curves in figure 3.3(b). For $T_{\max}$ the bollard pull is used, as the given lift and drag curves are assumed to be defined in percentage of bollard pull. When no lift and drag information is known, default lift and drag curves can be used. In figure 3.3(b) some lift and drag curves are given in percentage of the bollard pull, that can be used for low ship speeds.

The main propeller can be used in reverse mode. In such a case, the rudder is not able to deliver a lift force. The thrust region will reduce to a line shaped region, with about half the amount of bollard pull as the limiting thrust amount, due to inefficient water flows. For $T_{\min}$ half the bollard pull is a typical value.

3.3 Configuration matrix

The configuration matrix is used to easily calculate the forces and moment on the ship, generated by a propulsor, given its state (u_x, u_y). For this, the *Center of Gravity* (CoG) of the ship is assumed to be known and defined as $(x_{\text{CoG}}, y_{\text{CoG}})$. For each of the propulsors, its position with respect to the CoG of the ship is also assumed to be given by (x,y).From these the lever arms L_x and L_y can be calculated for each of the thrusters using

$$\begin{aligned} L_x &= x - x_{\text{CoG}}, \\ L_y &= y - y_{\text{CoG}}. \end{aligned} \tag{3.2}$$

For every thruster the forces and moment that it generates with respect to the CoG of the ship are given by (see also figure 3.4)

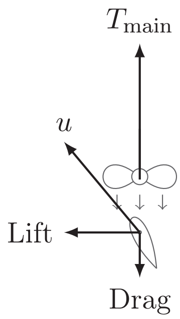
$$\begin{aligned} F_x &= u_x, \\ F_y &= u_y, \\ M_z &= L_x u_y - L_y u_x. \end{aligned} \tag{3.3}$$

From this the configuration matrix is formed by rewriting (3.3) as

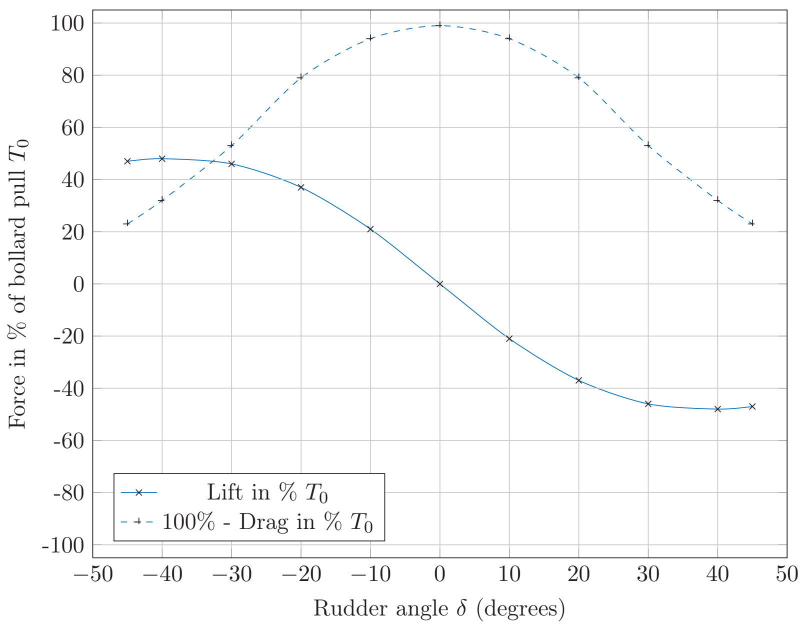
$$\begin{bmatrix} 1 & \\ & 1 \\ -L_y & L_x \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} = \begin{bmatrix} F_x \\ F_y \\ M_z \end{bmatrix} \qquad \Leftrightarrow \qquad Bu = \tau. \tag{3.4}$$

3.4 Minimizing power consumtion

The thrust allocator tries to fulfill the given τ_{ref}, while minimizing the power consumption. Because the model variables u represent thrust, we need to know how to relate this to



(a) Main propeller/rudder forces.



(b) Lift and drag curves in percentage of bollard pull of a rudder for low ship speeds.

Figure 3.3: Propeller/rudder forces.

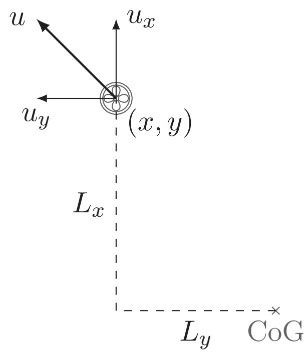


Figure 3.4: Thruster position definition.

power. The power consumption P as function of the amount of generated thrust T of a thruster, is typically defined by the non-linear relation:

$$P(T) = (P_{\max} - P_{\min}) \left(\frac{|T|}{T_{\max}} \right)^{\eta} + P_{\min} \tag{3.5}$$

In this model the parameters $P_{\max}$, $P_{\min}$, $T_{\max}$ and η totally define the relation between the thrust T in kN and the power P in kW of a thruster, where typically $1,3 < \eta < 1,7$. Using the least squares method, we can approximate this function with a quadratic function $wT^2 + c$, with $w, c \in \mathbb{R}$ (see figure 3.5).

For the modeled Cartesian decomposed thrust state (u_x, u_y) and thrust amount T the following relation holds:

$$T = \|(u_x, u_y)^T\|_2 = \sqrt{u_x^2 + u_y^2}. \tag{3.6}$$

It can be seen from figure 3.5 that approximating the function (3.5) with a function $wT^2 + c$, where $w, c \in \mathbb{R}$ are the weights matching the quadratic fit, is not too bad and matches the shape of $P(T)$ rather well. We see that

$$P(T) \approx wT^2 + c = w(u_x^2 + u_y^2) + c = (u_x, u_y) \begin{bmatrix} w & \\ & w \end{bmatrix} \begin{pmatrix} u_x \\ u_y \end{pmatrix} + c = u^T W u + c. \tag{3.7}$$

With the above approximation, the quadratic cost/weight matrix W is found, which we will use for the power minimization.

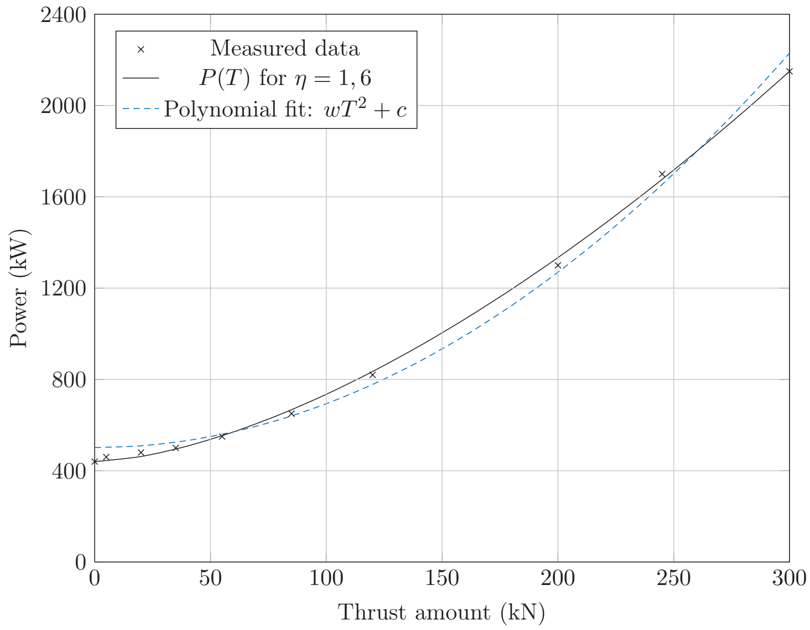


Figure 3.5: Power thrust relation for a bow thruster with $P_{\text{min}} = 440$, $P_{\text{max}} = 2150$ and $T_{\text{max}} = 300$.

Chapter 4

Lagrange multiplier method

The Lagrange multiplier method that is used inhouse by GustoMSC will be described, and its limitations are analyzed. The general model assumptions from Chapter 3 will be used.

4.1 Problem formulation

The thrust allocation problem is formulated as a quadratic equality constrained minimization problem and solved using the Lagrange multiplier method. Assume we have a ship with n thrusters and that all the information as discussed in Chapter 3 is known for every thruster. Then we know the cost/weight matrix $W_i \in \mathbb{R}^{2\times 2}$ and the configuration matrix $B_i \in \mathbb{R}^{3\times 2}$ for $i = 1, \ldots, n$. From these the thrust allocation problem can be assembled as

$$\begin{aligned} &\min_u u^T W u \\ &\text{s.t. } Bu = \tau_{\text{ref}} \end{aligned} \tag{4.1}$$

with

$$W = \begin{bmatrix} W_1 & & \\ & \ddots & \\ & & W_n \end{bmatrix}, \qquad B = \begin{bmatrix} B_1 & \cdots & B_n \end{bmatrix} \quad \text{and} \quad u = (u_1, \ldots, u_n)^T,$$

where

$$W \in \mathbb{R}^{2n\times 2n}, \quad B \in \mathbb{R}^{3\times 2n} \quad \text{and} \quad u \in \mathbb{R}^{2n}.$$

With the Lagrangian, problem (4.1) can be written as an unconstrained minimization problem

$$L(u, \lambda) = u^T W u + \lambda^T(-Bu + \tau_{\text{ref}}), \tag{4.2}$$

where $\lambda \in \mathbb{R}^3$ are the Lagrange multipliers. The Karush-Kuhn-Tucker (KKT) conditions give necessary conditions for an optimal solution:

$$\frac{\partial L}{\partial u} = 2Wu - B^T\lambda = 0 \quad \Rightarrow \quad u = \tfrac{1}{2} W^{-1} B^T \lambda \tag{4.3}$$

and

$$\frac{\partial L}{\partial u} = \tau_{\text{ref}} - Bu = 0 \quad \Rightarrow \quad Bu = \tau_{\text{ref}}. \tag{4.4}$$

From (4.3) and (4.4) it follows that

$$\tfrac{1}{2} BW^{-1}B^T\lambda = \tau_{\text{ref}} \quad \Rightarrow \quad \lambda = 2(BW^{-1}B^T)^{-1}\tau_{\text{ref}}. \tag{4.5}$$

Substitution of (4.5) in (4.3) gives the solution

$$u = W^{-1}B^T(BW^{-1}B^T)^{-1}\tau_{\text{ref}}. \tag{4.6}$$

When the matrix $BW^{-1}B^T$ is neither singular nor ill conditioned, problem (4.1) can be solved by solving the linear system of equations in (4.6).

4.2 Saturation handling

A problem with the above solution is that the thrust limits of the thrusters are not taken into account by the solver. The solutions will contain thrust values that the thrusters are not physically capable of. To solve this issue saturation handling is added to the thrust allocator algorithm. A saturation handling routine checks for every thruster if the proposed solution is saturating the thruster. If this is the case, the thruster is set to its maximum amount of thrust, and the generated forces and moment are calculated using $Bu = \tau$ for this thruster (equation (3.4)). Then the thruster is taken out of the problem, after updating τ_{ref} by subtracting τ from it. Now a smaller problem with one thruster less remains to be solved in the same fashion as before, until all the thrusters are saturated, or a valid solution is found.

An overview of the described saturation handling algorithm is given in the pseudo MATLAB code listing 4.1.

Listing 4.1: Multiplier Lagrange algorithm

```
%% Pre-processing
% Initialize the configuration matrix
% and the cost matrix for each propulsor
for each Propulsor in PropulsorList
{
    Propulsor.initConfigurationMatrix;
    Propulsor.initCostMatrix;
}

%% Lagrange optimization loop over the propulsorlist
while(1)
{
    if( PropulsorList.isEmpty || (||τ_ref|| < 10^-3) || Iteration > 20 )
    {
      % Stopping criterium
       break;
    }

    % Assemble the general configuration matrix and
```

```
    % the general inverse cost matrix
    for each Propulsor in PropulsorList
    {
        % Concatenate the configuration matrices
        B = [ B, Propulsor.getConfigurationMatrix ];
        % Concatenate the inverse weight matrices
        invW = [ invW, diag(inv(Propulsor.getCostMatrix)) ];
    }

    %% Make sure B is of full rank
    % (otherwise the inversion gives problems)
    % This is necessary when a thruster is located at the CoG.
    Tau = TargetTau( 1:rank(B) );
    B = B( 1:rank(B), : );

    %% Solve the unconstrained Lagrange problem
    % min_u u^T W u
    % subject to: τ − Bu = 0
    %
    u = W^{-1} B^T (B W^{-1} B^T)^{-1}

    % Break on errors
    if isnan(u)
    {
      % An error occurred while solving the Lagrange Multiplier system
        break;
    }

    %% Saturation handling
    % Post process the solution (for each propulsor),
    % taking into account the propulsor limits
    for each Propulsor in PropulsorList
    {
        % Calculate the thrust generated by this propulsor
        % according to the Lagrange solution
        τ_sol = Bu

        % If one of the propulsor parameters is saturated
        % (in the new Lagrange solution), the saturated parameter
        % is set according to the propulsor limits and
        % the propulsor is removed from the propulsor list
        % over which the lagrange multiplier method is applied iteratively
        % after τ_ref has been updated with τ_sol
        % (The propulsor cannot be altered anymore once is saturates).
    }
}
```

4.3 Thrust inequality constraints

In this section the Lagrange multiplier method will be extended with thrust constraints. Adding only *one* linear inequality constraint to (4.1) (representing one of the thrust constraints) will give the following problem:

$$\begin{aligned}\min_{u}\ & u^T W u \\ \text{s.t. } & Bu = \tau_{\text{ref}} \\ & Au \leq b\end{aligned}$$

with

$$b \in \mathbb{R}, \quad \text{and} \quad A \in \mathbb{R}^{1\times 2}.$$

Eliminating the inequality constraint with a slack variable $s \in \mathbb{R}$ gives:

$$\begin{aligned}\min_{u,s}\ & u^T W u \\ \text{s.t. } & Bu = \tau_{\text{ref}} \\ & Au + s^2 = b\end{aligned}$$

The Lagrangian now takes the form

$$L(u, s, \lambda, \mu) = u^T W u + \lambda(\tau_{\text{ref}} - Bu) + \mu(b - s^2 - Au).$$

Checking the KKT conditions gives

$$\begin{aligned}
\frac{\partial L}{\partial u} &= 2Wu - B^T\lambda - A^T\mu = 0 && \Rightarrow && u = \tfrac{1}{2}W^{-1}B^T\lambda + \tfrac{1}{2}W^{-1}A^T\mu, \\
\frac{\partial L}{\partial s} &= -2s\mu^T = 0 && \Rightarrow && s = 0 \text{ or } \mu = 0, \\
\frac{\partial L}{\partial \lambda} &= \tau_{\text{ref}} - Bu = 0 && \Rightarrow && Bu = \tau_{\text{ref}}, \\
\frac{\partial L}{\partial \mu} &= b - s^2 - Au = 0 && \Rightarrow && Au + s^2 = b.
\end{aligned}$$

From this it is evident that for every inequality constraint, the problem branches into two problems. So for n inequality constraints, 2^n problems will need to be solved. This indeed does indicate that a better solution method needs to be used for thrust allocation problems. In the next chapter Quadratic Programming techniques will be used to solve this shortcoming.

Chapter 5

Quadratic Programming

In this chapter Quadratic Programming (QP) methods combined with Disjunctive Programming techniques will be used to present an optimal thrust allocation algorithm, capable of doing full scale DP calculations for fixed/tunnel thrusters, azimuthing thrusters with forbidden/spoil zones and main propellers with rudder, while minimizing the power consumption. The general model assumptions formulated in Chapter 3 will be used.

Various factors led to the choice for Quadratic Programming methods. Power optimal thrust allocation required at least a quadratic object function. Also more solvers emerge for quadratic programming problems, using efficient interior point algorithms. Furthermore they can guarantee to find the optimal solution in a finite amount of time (which can be very important for online use), or they can guarantee that no solution exists, whereas with non-linear optimizations techniques these guarantees cannot be given. Still they are quite easy to use and easy to define. Because of these factors, Quadratic Programming seems a very suitable method to handle the thrust allocation problem.

5.1 Problem formulation

Using the same techniques as in Chapter 4, we can formulate the thrust allocation problem as

$$\begin{aligned} &\min_u u^T W u, \\ &\text{s.t. } Bu = \tau_{\text{ref}}, \\ &\qquad Au \leq b. \end{aligned} \tag{5.1}$$

The simplest form of the power limiting thrust allocation problem is now formulated and can be solved by a QP solver. The only thing that needs some more attention are the added inequality constraints $Au \leq b$ which contains the thrust constraints for every type of actuator.

5.2 Thrust region constraints

For each thruster, a set of inequality constraints is created, representing the thrust region of the thruster. The inequality constraints will be formed by a finite intersection of hyperplanes, resulting in a convex polygon. This polygon represents the linearized thrust region of the thruster and can take different shapes for different thruster types (as discussed in section 3.2).

5.2.1 Fixed/Tunnel thruster

The thrust region of a fixed thruster can be modeled as a line segment (see figure 3.2(a) and 3.2(b)). Given its constant angle α and its thrust limits $T_{\max}$ and $T_{\min}$ the following constraints can be added to the QP problem (5.1)

$$\begin{bmatrix}\sin(\alpha) & -\cos(\alpha)\end{bmatrix}\begin{pmatrix}u_x\\ u_y\end{pmatrix} = 0,$$
$$\begin{bmatrix}\cos(\alpha) & \sin(\alpha)\\ -\cos(\alpha) & -\sin(\alpha)\end{bmatrix}\begin{pmatrix}u_x\\ u_y\end{pmatrix} \leq \begin{bmatrix}T_{\max}\\ -T_{\min}\end{bmatrix}.$$

The constraints are illustrated in figure 5.1 for a tunnel thruster (a fixed thruster with $\alpha = 90°$). Of course the equality constraint can be written as two inequality constraints, but then they are more likely to give numerical problems for the QP solver. Therefore one equality constraint and two inequality constraints are used. The equality constraint will be added to the rest of the defined equality constraints, and forms the only exception to the inequality constraints, defining the thruster regions.

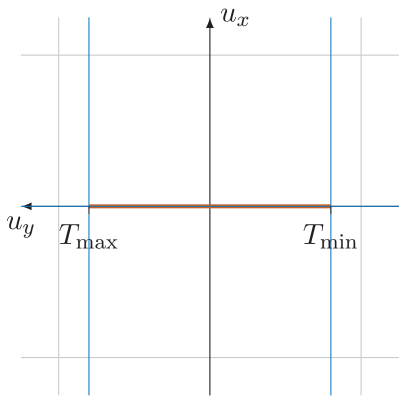


Figure 5.1: The thrust constraints for a tunnel thruster (a fixed thruster with $\alpha = 90°$).

5.2.2 Azimuth thruster

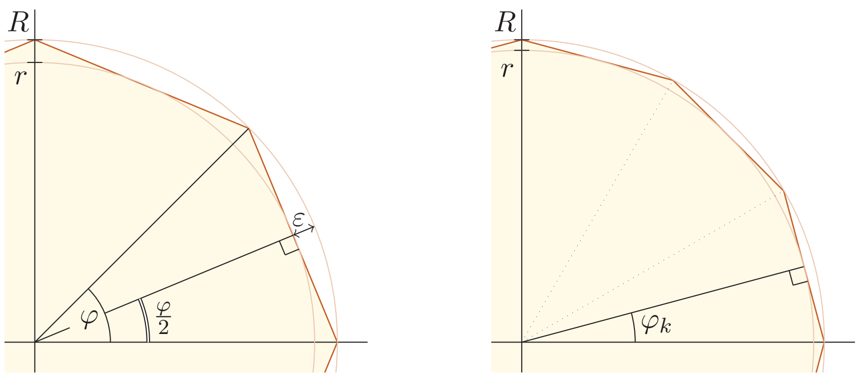
For an azimuthing thruster the inequality constraints are more complex. The thrust region for an azimuth thruster without any forbidden/spoil zones has a circular shape with radius $T_{\max}$. This can be written as

$$\sqrt{u_x^2 + u_y^2} \leq T_{\max} \quad \Leftrightarrow \quad \|(u_x, u_y)^T\|_2 \leq T_{\max}. \tag{5.2}$$

As these are non linear constraints (and can therefore not be added to (5.1)), they will need to be linearized, approximating the circular thrust region sufficiently accurately (see also [Lea08]). When doing this it is necessary to make sure that the approximation stays strictly within the given thrust region. Otherwise solutions could go beyond the maximal thrust that can be realized by the thruster, and would over evaluate the DP capability. To get the best balance between the approximation error and the number of inequality constraints, the linearization error needs to be analyzed in more detail.

Linear approximation error

A circular region with a radius $T_{\max} = R > 0$ will be approximated by a N-sided regular polygon (with $N \geq 3$). The regular polygon divides the circular region into N circular sectors, each having a central angle of $\varphi = 2\pi/N$. One such circle sector is depicted in figure 5.2(a). For a given circumradius R and a given number of sides N of the



(a) Linear approximating a regular polygon.

(b) Creating hyperplanes.

Figure 5.2: Creating a system of linear inequalities that represent a circular thrust region, by approximating it with a regular polygon.

approximating regular polygon, we find for the inradius r of the regular polygon

$$r = R\cos\left(\tfrac{\varphi}{2}\right) = R\cos\left(\tfrac{\pi}{N}\right). \tag{5.3}$$

The maximum approximation error ε is found as the difference between the circumradius R and the inradius r of the regular polygon (see also figure 5.2(a))

$$\varepsilon = R - r = R\left(1 - \cos\left(\tfrac{\pi}{N}\right)\right). \tag{5.4}$$

The maximum approximation error depends on the given circumradius R and the given number of sides N of the regular polygon. When the circumradius R and a maximum error $\varepsilon > 0$ are given, we find for the minimum number of sides of the approximating regular polygon that

$$N = \left\lceil \frac{\pi}{\arccos\left(1 - \frac{\varepsilon}{R}\right)} \right\rceil, \quad \text{with} \quad R > 0. \tag{5.5}$$

With the approximation error ε defined as 1% of $T_{\max}$ ($\varepsilon = T_{\max}/100$), the minimum number of sides required to achieve this accuracy is given by formula (5.5). Using formula (5.3) for the inradius r the circular thrust region is defined by the following system of linear inequalities (see also figure 5.2(b))

$$\begin{bmatrix} \cos\varphi_k & \sin\varphi_k \end{bmatrix} \begin{pmatrix} u_x \\ u_y \end{pmatrix} \leq r, \tag{5.6}$$

for every

$$\varphi_k = \frac{\pi}{N} + k\frac{2\pi}{N} = (2k+1)\frac{\pi}{N}, \quad \text{with} \quad k = 0, \ldots, N-1. \tag{5.7}$$

Figure 5.3 shows some examples of the linearization method described above.

Forbidden zones

Forbidden zones can be defined by taking out some portion of the thrust region. This way the solver is prevented from allocating thrust in the defined forbidden zone, because it will no longer be part of the feasible set in which the solver searches for its solution. For azimuth thrusters it is obvious to choose circle sector/pie shaped regions, as this will restrict the azimuth thruster from operating at certain angles and thrust magnitudes. To create pie shaped thrust regions, the method described in 5.2.2 can be used again to balance the accuracy and the number of inequality constraints for the thrust region. The two additional hyperplanes, limiting the sector angle (see also figure 5.4), are of the form

$$\begin{bmatrix} \sin\varphi_{\text{start}} & -\cos\varphi_{\text{start}} \\ -\sin\varphi_{\text{end}} & \cos\varphi_{\text{end}} \end{bmatrix} \begin{pmatrix} u_x \\ u_y \end{pmatrix} \leq \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{5.8}$$

Here φ_{start} and φ_{end} should be ordered counterclockwise. Figure 5.5 shows some examples of circle sector/pie shaped thrust regions, with their corresponding hyperplanes.

By defining a forbidden zone for an azimuth thruster, a Pacman like thrust region shape is left over. Because this shape is in general not convex, it must be split up into multiple (disjunct) convex thrust regions. When split, using disjunctive programming techniques the problem can still be solved at the cost of extra computation time. Intelligent splitting of the thrust region into a minimal number of convex thrust regions is necessary to be able to use forbidden zones in the problem definition and the *minimal* number of regions will minimize the computation time. With this technique quite exotic thrust regions can be used, as can be seen in figure 5.6. It could also be used to define a thrust region for an azimuth thruster, that minimizes thruster-hull interactions.

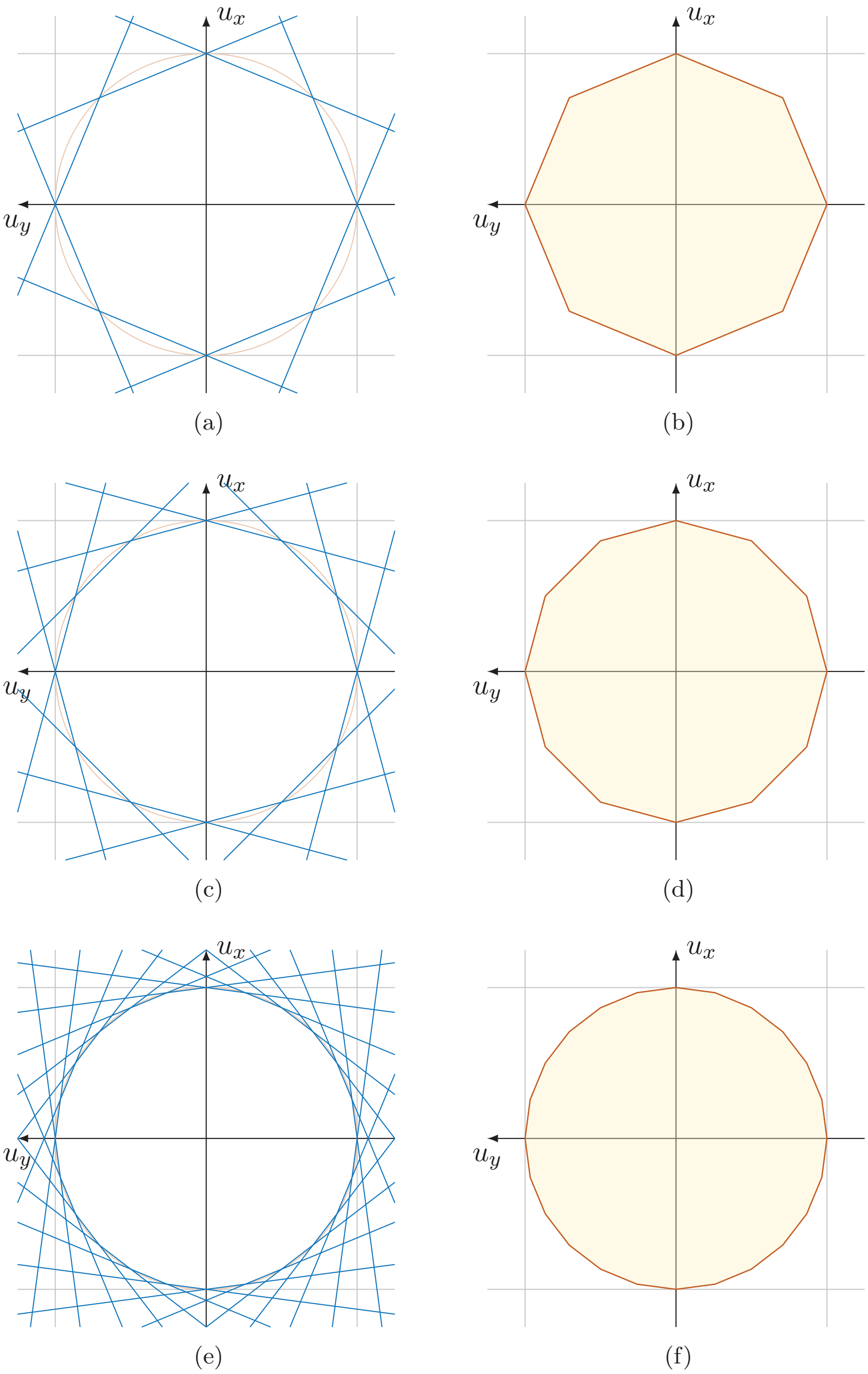


Figure 5.3: Examples of linear circle approximations showing the hyperplanes that define the regular polygon for a different number of sides/constraints ($N = 8$, $N = 12$ and $N = 24$).

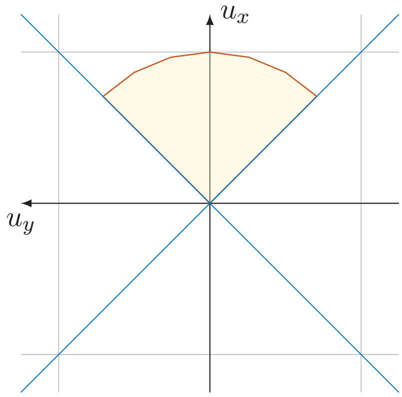


Figure 5.4: A circle sector shaped thrust region, illustrating the sector angle constraints.

5.2.3 Main propeller with rudder

The thrust region for a propeller/rudder pair operating in forward mode is derived from the lift and drag curves (see figure 3.3(a), 5.8(a) and [MT07]). These relate the rudder angle to the lift/drag forces in percentage of the bollard pull[1] T_0 (often known from a bollard pull test). From the lift and drag curves and the known bollard pull, the generated thrust in the F_x and F_y directions can be derived for every rudder angle. These points define the vertices of the polygon used to define the thrust region for the propeller/rudder pair in forward mode. To find the inequality constraint for every hyperplane, we use the fact that the line through two different points (x_1, y_1) and (x_2, y_2) is defined by

$$(y - y_1) = \frac{(y_2 - y_1)}{(x_2 - x_1)}(x - x_1).$$

Assuming that the polygon vertices are ordered counterclockwise the hyperplanes defining the thrust region are

$$\begin{bmatrix} a_{k,1} & a_{k,2} \end{bmatrix} \begin{pmatrix} u_x \\ u_y \end{pmatrix} \leq b_k, \tag{5.9}$$

with

$$\begin{aligned} a_{k,1} &= (y_{k+1} - y_k), \\ a_{k,2} &= (x_k - x_{k+1}), \\ b_k &= x_k y_{k+1} - x_{k+1} y_k, \end{aligned} \tag{5.10}$$

for every couterclockwise succeeding pair of points (x_k, y_k) and (x_{k+1}, y_{k+1}), where $k = 0, \ldots, N$ (see also figure 5.7). Note that for a cyclic/closed polygon $(x_0, y_0) = (x_N, y_N)$.

[1]In a bollard pull test the ship is tied to a bollard and the maximum pulling force is measured when the ship is at full power. This gives an indication of the maximum thrust force, generated by the propulsion system on the ship.

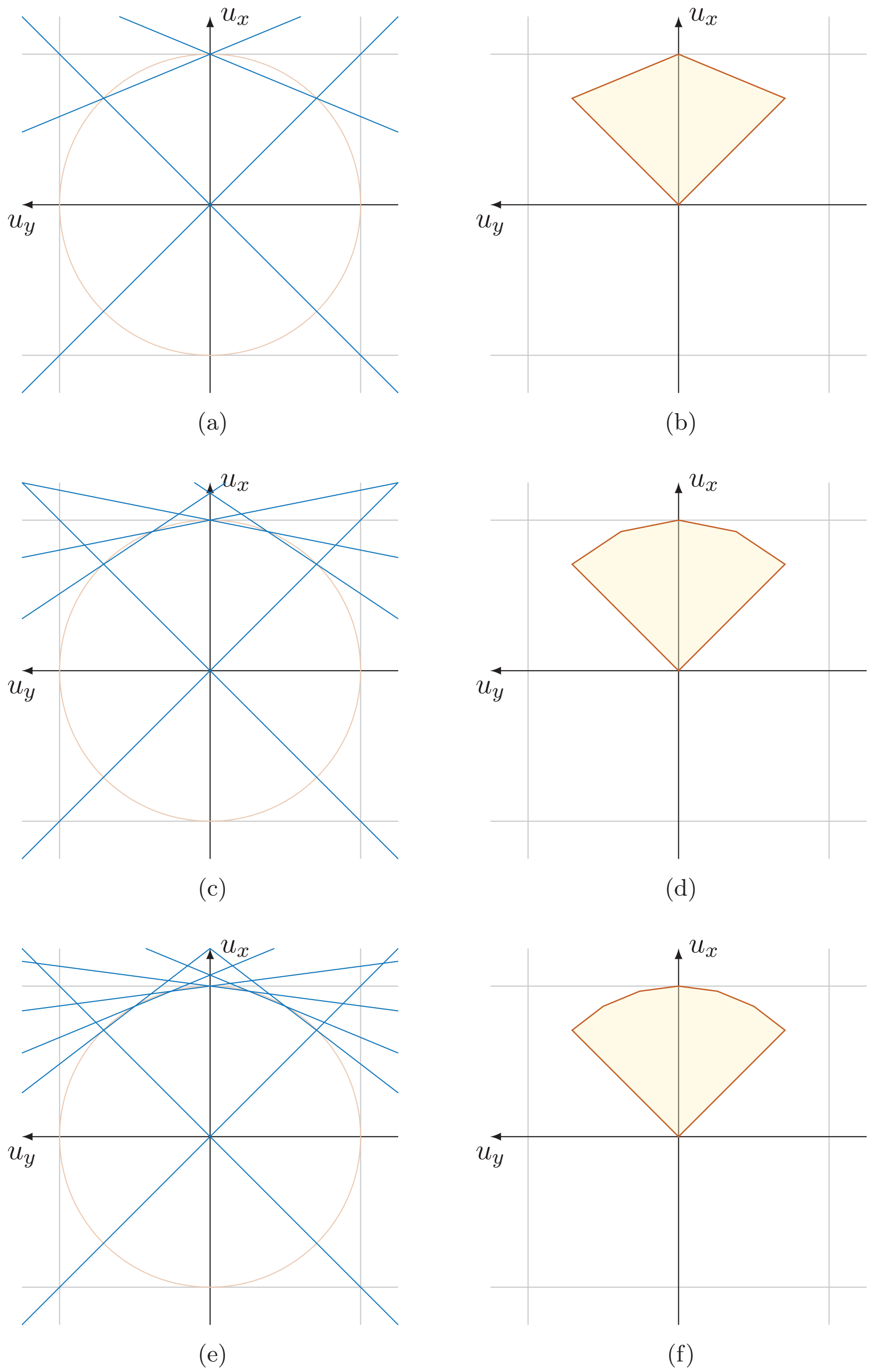
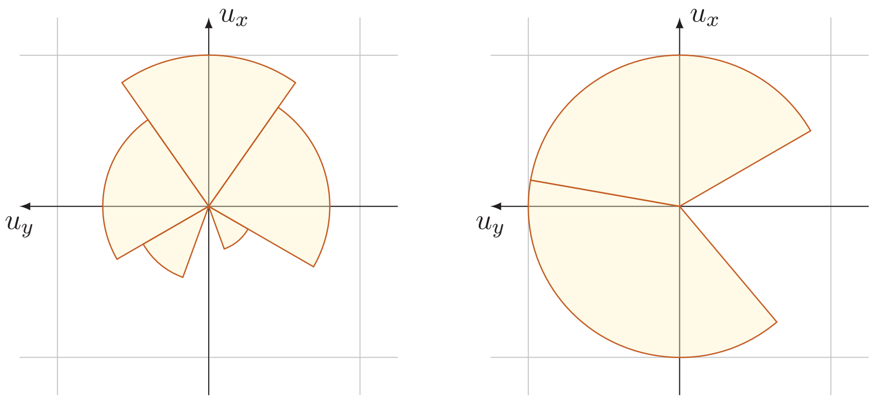


Figure 5.5: Examples of circle sector/pie shaped thrust region approximations (resulting in 4, 6 and 8 hyperplanes/inequality constraints).



(a) An exotic thrust region for an azimuth thruster.

(b) A *Pacman* shaped thrust region for an azimuth thruster, split into two disjunct convex regions.

Figure 5.6: Examples of some exotic thrust regions that can be defined using multiple disjunct thrust regions.

The convex polygon is now defined by the linear system of inequalities

$$\begin{bmatrix} a_{0,1} & a_{0,2} \\ \vdots & \vdots \\ a_{N,1} & a_{N,2} \end{bmatrix} \begin{pmatrix} u_x \\ u_y \end{pmatrix} \leq \begin{bmatrix} b_0 \\ \vdots \\ b_N \end{bmatrix} \quad \Leftrightarrow \quad Au \leq b. \tag{5.11}$$

Figure 5.8 shows the approximated thrust region for an example propeller/rudder pair.

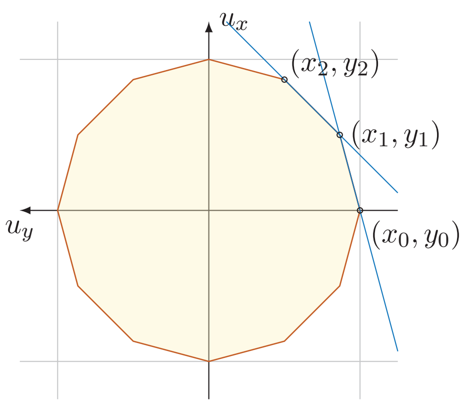
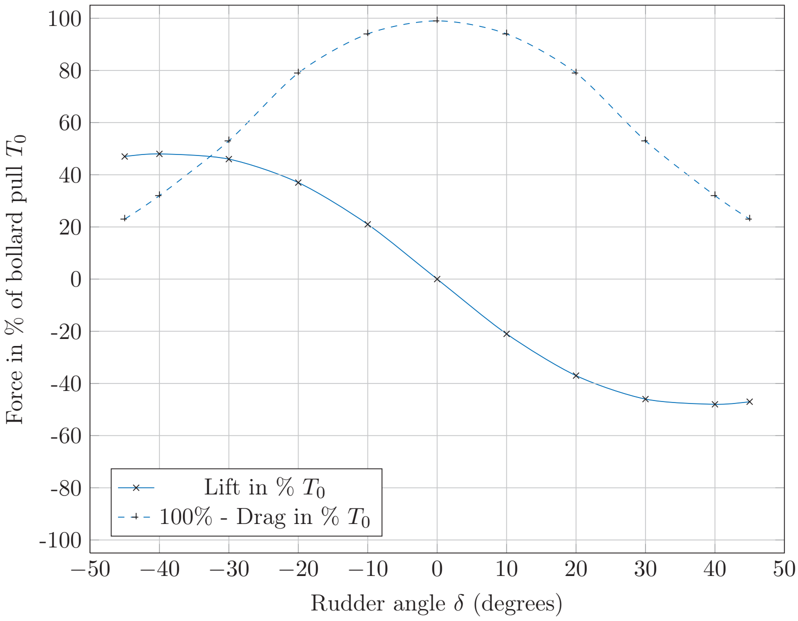


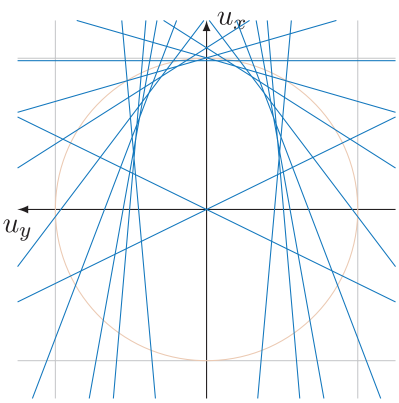
Figure 5.7: Creating hyperplanes from counterclockwise ordered polygon vertices.

When no lift and drag curves are known, a circle sector/pie shaped thrust region could also be used for the thrust region.

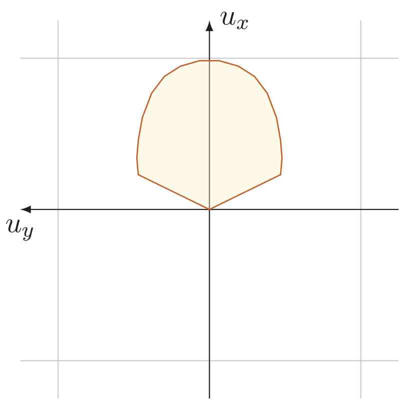
When the propeller is working in reverse mode a line shaped thrust region appears, as the rudder cannot generate lift in this situation. This is again easily defined (by 1 equality



(a)



(b)



(c)

Figure 5.8: Approximating the thrust region of a propeller/rudder pair in forward mode from the lift and drag curves.

constraint and 2 inequality constraints) and is added to the problem as a disjunct thrust region of the propeller/rudder pair, because the combined thrust region is obviously not convex when $T_{\min} < 0$.

5.2.4 Thrust relations

Additional thrust relations between different thrusters can also be added to the problem formulation. For instance, the relation $u_{y,1} = u_{y,2}$ would link the amount of thrust generated by thruster 1 in the F_y direction with the amount of thrust generated in the F_y direction by thruster 2. This can be used when two tunnel thrusters are physically controlled by one signal. With these kind of relations thrust mirroring is also possible ($u_{y,1} = -u_{y,2}$). Even thrust relations between multiple thrusters are possible, as long as the relations are linear in u.

5.3 Handling non-convex thrust regions

If all the thrust regions would have been convex, the formulated problem can be solved right away. Since some of the thrust regions are split into multiple convex thrust regions additional treatment is required (see also [STJ06]). If on the other hand one would try to define a non convex thrust region with intersecting hyperplanes, only a smaller portion of the intended thrust region will be the result. This is because one can prove that a finite number of intersecting hyperplanes always will result in a convex set. So the choice is either to split the non convex region up into multiple convex disjunct parts, or to use a different shaped and smaller thrust region that indeed *is* convex.

The trick to solving the main optimization problem when disjunct thrust regions are defined, is to first generate all the possible combinations of the thrust regions, picking *one* disjunct convex region for each thruster. This can be programmed as a backtracking routine, using recursion to go to the shrinking tree of possible combinations for each thruster. The total number of combinations can be derived by multiplying the number of disjunct thrust regions for each thruster. It is therefore wise to use as little disjunct thrust regions as possible, to minimize computation time.

When for example a ship with 2 tunnel thrusters and 2 propeller/rudder pairs is given, we know that the tunnel thrusters each have only one disjunct thrust region (line shaped) and each propeller/rudder pair will have two disjunct thrust regions (one for forward mode and one for reverse mode). All the possible thrust region combinations are schematically given by (using the notation (1$^{\text{st}}$ tunnel, 2$^{\text{nd}}$ tunnel, 1$^{\text{st}}$ rudder, 2$^{\text{nd}}$ rudder))

(1,1,1,1) The tunnel thrusters use their only defined thrust region, and the propeller/rudder pairs are both operating in forward mode;

(1,1,1,2) The tunnel thrusters use their only defined thrust region, and the first propeller/rudder pair is operating in forward mode, while the second one is operating in reverse mode;

(1,1,2,1) The tunnel thrusters use their only defined thrust region, and the second propeller/rudder pair is operating in forward mode, while the first one is operating in reverse mode;

(1,1,2,2) The tunnel thrusters use their only defined thrust region, and the propeller/rudder pairs are both operating in reverse mode.

For each of these thrust region combinations, the QP problem is formulated/assembled and solved. While this happens, the solution corresponding to each combination is stored. After solving all the QP subproblems, the best solution is chosen by comparing the objective costs and this will be the optimal solution of the main problem (see also [JFTF08]).

Obviously the number of disjunct thrust regions that can be used with this method is limited, given that the problem should be solved within a defined finite amount of time (also neglecting any possible computer problems, such as memory size etc.). However, in practice, the number of thrusters on a vessel is also limited, and no more than a few forbidden zones will be needed for most applications.

The disjunctive programming technique lets us solve the problem for non convex thrust regions, taking up extra computation time. The disjunctive programming method can be exploited more, resulting in more control on the allocator, while even reducing the main problem size. This will be discussed in the following two sections.

5.3.1 Spoil zones

To define spoil zones for certain thrust regions a spoil multiplier is introduced and defined for every disjunct thrust region, for every thruster. This coefficient can now be used in every subproblem, by using it as a multiplier in the objective/cost function (multiplying it with the corresponding thruster weights w). So setting the spoil multiplier to 1, will not change the problem whatsoever, but increasing the multiplier for a particular thrust region, increases the objective/cost function, and thus makes it a more expensive solution, making the spoil zone region less attractive for the solver to use. Because the main solution is determined by comparing the costs of the subproblems, the spoil multiplier has the effect of making certain thrust regions less likely to be used, hence the name spoil zone. It is even possible to split the thrust regions into disjunct thrust regions just to represent a spoil zone.

5.3.2 Thruster relations

Another way to exploit the disjunct programming technique is to remove some of the combinations of thrust regions from the problem. This can come in handy when for instance one is only interested in solutions where the two propeller/rudder pairs should work simultaneously in forward mode. This can be easily realized by throwing away the correct subproblems from the main problem. Even more relations can be forced by using a filter on the thrust region combinations. This also reduces the number of subproblems that need to be solved to find the solution of the main problem and thus also reduces the computation time. Again disjunct thrust regions can also be created with these kind of thruster relations in mind.

5.4 Dynamic thrust regions

For the time domain simulation the thrust limits of each thruster for each time step will be modeled by *adding* a so called *dynamic thrust region* to the problem for each thruster (see also [Rut08]). A dynamic thrust region models the physical limitations of a thruster at a given time step. For instance, an azimuth thruster can not turn 180 degrees in half a second, and without these dynamic regions that limit the thrust region, unrealistic behavior would be the result in time domain simulations. By adding the dynamic region for each thruster to the global problem, the global/static thrust allocation problem, with all its subproblems, needs to be assembled only once (at initialization). The dynamic thrust regions will be created at every time step, as these can change every time step (see also 5.9). Because the dynamic thrust region for every thruster contains no more than five vertices[2], they are computationally cheap to generate. Each dynamic thrust region will therefore add no more than five inequality constraints to the global problem. Figure 5.10 shows how the dynamic thrust region is defined, and figure 5.11 shows a created dynamic thrust region with its hyperplanes. In figure 5.12(a) the thrust region of an azimuth thruster with a forbidden zone and a dynamic thrust region can be seen. It shows how the global/static and the dynamic thrust constraints work together (being both in the problem formulation). Figure 5.12(b) shows that a small forbidden zone and a relative large dynamic thrust region can give problems.

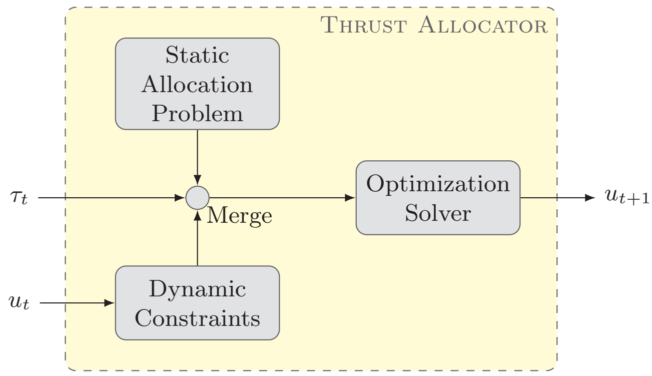


Figure 5.9: QP thrust allocator block diagram.

[2] When necessary, the dynamic thrust region can also reduce to a pie, line or point like shape. This is achieved by using some checking algorithms which can be implemented straight forward.

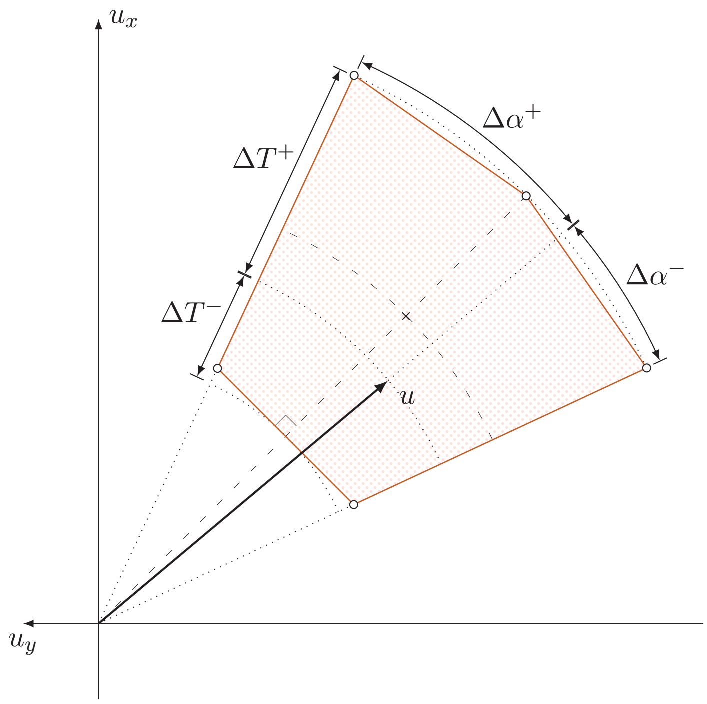


Figure 5.10: Defining a dynamic thrust region, representing the physical limits of a thruster.

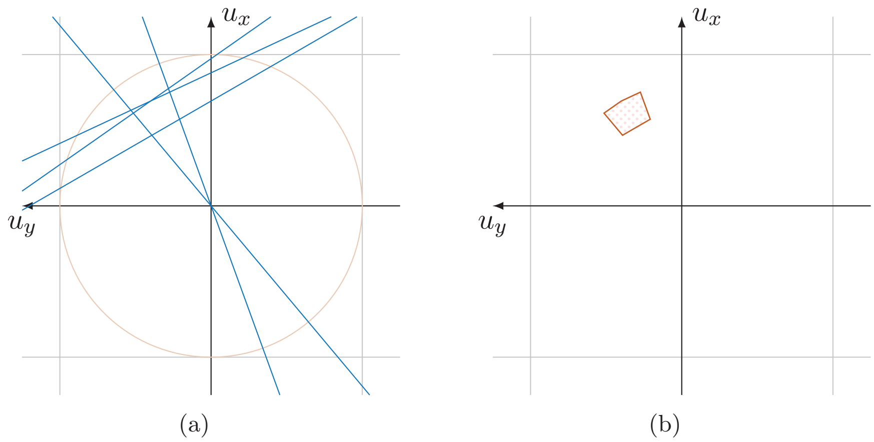
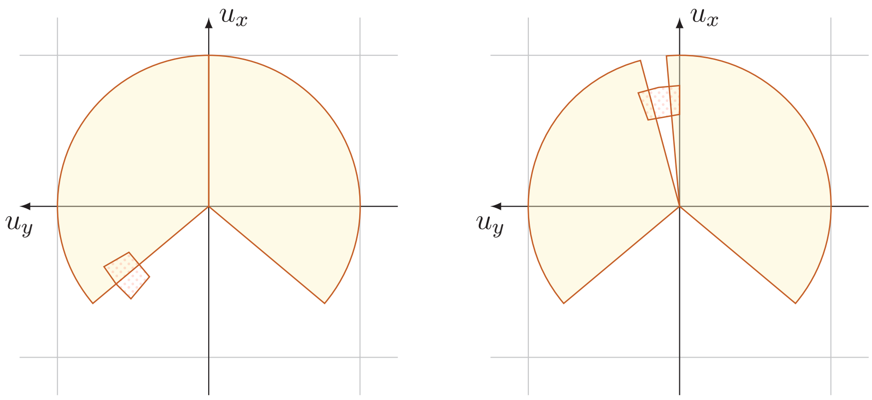


Figure 5.11: A dynamic thrust region.



(a) An azimuth thruster with a defined forbidden zone and a dynamic thrust region.

(b) An azimuth thruster with forbidden zones and a dynamic thrust region in a problematic position.

Figure 5.12: Azimuth thrusters with forbidden zones and a dynamic thrust region.

5.5 QP models

5.5.1 QP basic model

This model is used to solve the basic thrust allocation problem. It tries to solve the DP problem while minimizing the power consumption.

$$\begin{aligned} \min_u \; & u^T W u \\ \text{s.t. } & Bu = \tau \\ & Au \leq b \end{aligned} \tag{5.12}$$

with

$$W = \begin{bmatrix} W_1 & & \\ & \ddots & \\ & & W_n \end{bmatrix} \qquad B = \begin{bmatrix} B_1 & \cdots & B_n \end{bmatrix} \qquad A = \begin{bmatrix} A_1 & & \\ & \ddots & \\ & & A_n \end{bmatrix}$$
$$u = \begin{bmatrix} u_1 & \cdots & u_n \end{bmatrix}^T \qquad b = \begin{bmatrix} b_1 & \cdots & b_n \end{bmatrix}^T$$

with $W > 0$ containing the power/thrust coefficients.

5.5.2 QP minimize largest thrust force

This model is the same as the basic model, but also gives penalty to the maximum thrust force $\bar{u} \in \mathbb{R}$. Therefore this model avoids/discourages solutions with large differences in the amount of thrust, generated by the different thrusters, when other solutions with overall smaller thrust amounts exist (see also [JFT05]).

$$
\begin{aligned}
\min_{u,\bar{u}} \; & u^T W u + \beta \bar{u} \\
\text{s.t. } & Bu = \tau \\
& Au \leq b \\
& -\bar{u} \leq u \leq \bar{u}
\end{aligned}
\tag{5.13}
$$

with $W > 0$ containing the power/thrust coefficients and where $\beta \geq 0$ minimizes the largest force

$$
\bar{u} = \max_i |u_i|. \tag{5.14}
$$

Problem (5.13) can be formulated in the standard QP form:

$$
\begin{aligned}
\min_{u,\bar{u}} \; & \begin{bmatrix} u \\ \bar{u} \end{bmatrix}^T \begin{bmatrix} W & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ \bar{u} \end{bmatrix} + \begin{bmatrix} 0 & \beta \end{bmatrix} \begin{bmatrix} u \\ \bar{u} \end{bmatrix} \\
\text{s.t. } & \begin{bmatrix} B & 0 \end{bmatrix} \begin{bmatrix} u \\ \bar{u} \end{bmatrix} = \tau \\
& \begin{bmatrix} A & 0 \\ -I & -1 \\ I & -1 \end{bmatrix} \begin{bmatrix} u \\ \bar{u} \end{bmatrix} \leq \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix} \\
& \begin{bmatrix} -\infty \end{bmatrix} \leq \begin{bmatrix} \bar{u} \end{bmatrix} \leq \begin{bmatrix} \infty \end{bmatrix}
\end{aligned}
\tag{5.15}
$$

5.5.3 QP relaxed model

This model always generates a solution, even when the ship cannot hold position. This model is used for the time domain simulations, as the system still needs to know what it should do, even though it cannot always hold position. The slack variable $s \in \mathbb{R}^3$ is given a relative high cost ($Q >> W > 0$), so that the solver will first try to minimize s and will therefore first try to make sure that $Bu = \tau$. The model can be written as:

$$
\begin{aligned}
\min_{u,s} \; & u^T W u + s^T Q s \\
\text{s.t. } & Bu = \tau + s \\
& Au \leq b \\
& -\infty \leq s \leq \infty
\end{aligned}
\tag{5.16}
$$

with $Q >> W > 0$. It is also possible to give preference to position holding or heading keeping. When position keeping is more important than heading keeping, then s_1 and s_2 should be chosen such that $s_1 >> s_3$ and $s_2 >> s_3$. The standard QP form of (5.16) is given by:

$$\begin{aligned}
&\min_{u,s} \begin{bmatrix} u \\ s \end{bmatrix}^T \begin{bmatrix} W & 0 \\ 0 & Q \end{bmatrix} \begin{bmatrix} u \\ s \end{bmatrix} \\
&\text{s.t. } \begin{bmatrix} B & -I \end{bmatrix} \begin{bmatrix} u \\ s \end{bmatrix} = \tau \\
&\begin{bmatrix} A & 0 \end{bmatrix} \begin{bmatrix} u \\ s \end{bmatrix} \leq b \\
&\begin{bmatrix} -\infty \end{bmatrix} \leq \begin{bmatrix} s \end{bmatrix} \leq \begin{bmatrix} \infty \end{bmatrix}
\end{aligned} \tag{5.17}$$

5.5.4 QP relaxed model, minimizing power and largest thrust force

This model simply combines the previously discussed relaxed model 5.16 and force minimizing model 5.13 into one:

$$\begin{aligned}
&\min_{u,s,\bar{u}} u^T W u + s^T Q s + \beta \bar{u} \\
&\text{s.t. } Bu = \tau + s \\
&Au \leq b \\
&-\infty \leq s \leq \infty \\
&-\bar{u} \leq u \leq \bar{u}
\end{aligned} \tag{5.18}$$

with $Q >> W > 0$ and where $\beta \geq 0$. Where again

$$\bar{u} = max_i |u_i|. \tag{5.19}$$

Formulating problem (5.18) in standard QP form gives:

$$\begin{aligned}
&\min_{u,s,\bar{u}} \begin{bmatrix} u \\ s \\ \bar{u} \end{bmatrix}^T \begin{bmatrix} W & 0 & 0 \\ 0 & Q & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ s \\ \bar{u} \end{bmatrix} + \begin{bmatrix} 0 & 0 & \beta \end{bmatrix} \begin{bmatrix} u \\ s \\ \bar{u} \end{bmatrix} \\
&\text{s.t. } \begin{bmatrix} B & -I & 0 \end{bmatrix} \begin{bmatrix} u \\ s \\ \bar{u} \end{bmatrix} = \tau \\
&\begin{bmatrix} A & 0 & 0 \\ -I & 0 & -1 \\ I & 0 & -1 \end{bmatrix} \begin{bmatrix} u \\ s \\ \bar{u} \end{bmatrix} \leq \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix} \\
&\begin{bmatrix} -\infty \\ -\infty \end{bmatrix} \leq \begin{bmatrix} s \\ \bar{u} \end{bmatrix} \leq \begin{bmatrix} \infty \\ \infty \end{bmatrix}
\end{aligned} \tag{5.20}$$

5.6 Problem scaling

To avoid numerical problems, the QP problem can be scaled before it goes to the QP solver. After the scaled QP problem is solved, the solution is scaled back again. Especially when the quadratic weight/cost coefficients (in the matrices W and Q) are large, numerical problems can arise, therefore the scaling factor c_{scale} is chosen such that

$$c_{\text{scale}} u_{\max}^2 \approx 1, \tag{5.21}$$

where $u_{\max}$ is the largest absolute thrust value that can be generated by one of the thrusters. Then this scaling factor is also used when the thrust regions (inequality constraints) are created ($T_{\min}$ and $T_{\max}$ are pre-scaled with this factor) and on the τ_{ref} vector. After the scaled problem is solved by a QP solver the vector u containing the solution is scaled back again with $1/c_{\text{scale}}$.

5.7 QP solvers

Two external open source QP solvers were used to solve the QP problems in Matlab . They are both compiled with a c++ compiler, and used in Matlab as *mex* code. QPC and OOQP are both open source quadratic program solvers, written in c. They are based on interior-point methods, for solving convex quadratic programming problems. More information on these solvers can be found at http://pages.cs.wisc.edu/~swright/ooqp/ and http://sigpromu.org/quadprog/index.html.

Chapter 6

Results

Using the DP tools developed and used by GustoMSC inhouse the proposed QP power optimal thrust allocation algorithm is implemented in MATLAB and tested in the GustoMSC DP tools environment, testing it with DP capability plots and time domain simulations. The results will be presented in the following sections.

6.1 DP Capability Plots

A DP capability plot gives a visual impression of the maximum environment in which a ship can hold position. They are used in the development phase of a ship, to get an indication of its future DP capabilities. They are also used when analyzing the robustness of a DP system. For instance, the impact of a thruster failure on the DP capability can be analyzed with a DP capability plot.

Specifications on the production of DP capability plots are given by *The International Marine Contractors Association* (IMCAM140) [3]. These have the primary goal to standardize the production of capability plots and enable a direct comparison of DP capability plots from different vessels. There is however still some degree of freedom, and the specifications are mere guidelines for best industry practice.

A (static) DP capability plot only depends on the *existence* of solutions to DP problems, for different environmental conditions. Every point in a DP capability plot defines an environmental state, defining a wind speed and a direction from which the wind, current and waves come from. It is common practice to look at the worst case scenario and take the wind, current and waves collinear. From this information τ_{ref} can be calculated for the particular ship, and the thrust allocator tries to solve the DP problem. For each direction a point is plotted at the highest wind speed for which the DP problem has a solution. Doing this for all the directions creates what is called a DP plot and indicates the DP capability of the ship. So again, only the *existence* of a solution to a DP problem is actually needed to create a DP capability plot, the actual solution is not necessary, but can of course be used to get more insight in the DP capabilities of the ship.

In the following sections some DP capability plots are discussed.

6.1.1 Optimal thrust allocation

Figure 6.1 shows two DP capability plots for a ship containing only azimuth thrusters. Because the ship has only azimuth thrusters defined, the Lagrange multiplier method and the Quadratic Programming methods can be compared for this ship. It can be cleary seen from figure 6.1 that the Lagrange multiplier method results in less DP capability for this ship, where the QP thrust allocator is still able to find solutions for some more extreme conditions. This is not very surprising, because the Lagrange multiplier method can not generate optimal solutions. It actually solves the problem for thrusters with unlimited thrust capabilities and then the saturation handling takes into account the thrust regions. Even changing the order in which the saturation of the thrusters is handled, can give different results. There were also differences in the proposed thruster configuration for different conditions. This also is no surprise as we are comparing a non-optimal with an optimal thrust allocation method.

Another point of interest is that the Lagrange multiplier method did run into numerical difficulties for this problem. This occured because the matrix $BW^{-1}B^T$ gets ill conditioned for certain situations. Also in the time domain simulations this did happen from time to time, making it less robust than the QP thrust allocator (with problem scaling). Especially for under and fully actuated problems this can be a big problem (where in some cases the Lagrange multiplier allocator was not able to find any solution). For over actuated problems the number of numerical problems decrease, but can still pop up under certain conditions. To avoid numerical problems with the Lagrange multiplier method near zero thrust, it could be possible to add a relative large artificial force to the problem, before solving it, and then subtract it again from the found solution. This method is however not tested in this thesis.

6.1.2 Thruster relations

In this section a somewhat strange phenomena will be discussed, where two tunnel thrusters seem to be wasting energy by working in opposite directions. At first this seemed strange, because it did not look like a very optimal solution, but given the problem formulation, it all makes sense. The DP capability plot can be seen in figure 6.2.

On closer inspection the two tunnel thrusters were operating in the same direction for an environment coming from 350°. For increasing wind speeds the tunnel thrusters seemed to generate thrust in the same direction, but at a certain point the solution from the thrust allocator made the tunnel thrusters blow in opposite direction. This can however be explained by the fact that the two tunnel thrusters are a few meters apart from each other. Because of this, a small moment can be generated by letting them generate thrust in opposite direction, while the resulting force in the F_y direction remains zero (they counteract each other). For this particular environmental condition this small amount of extra moment can just make the problem solvable. Letting the tunnel thrusters blow in opposite directions, apparently forms the only possible way to solve this particular DP problem. Because it is the *only* solution to the DP problem it is automatically also the optimal solution.

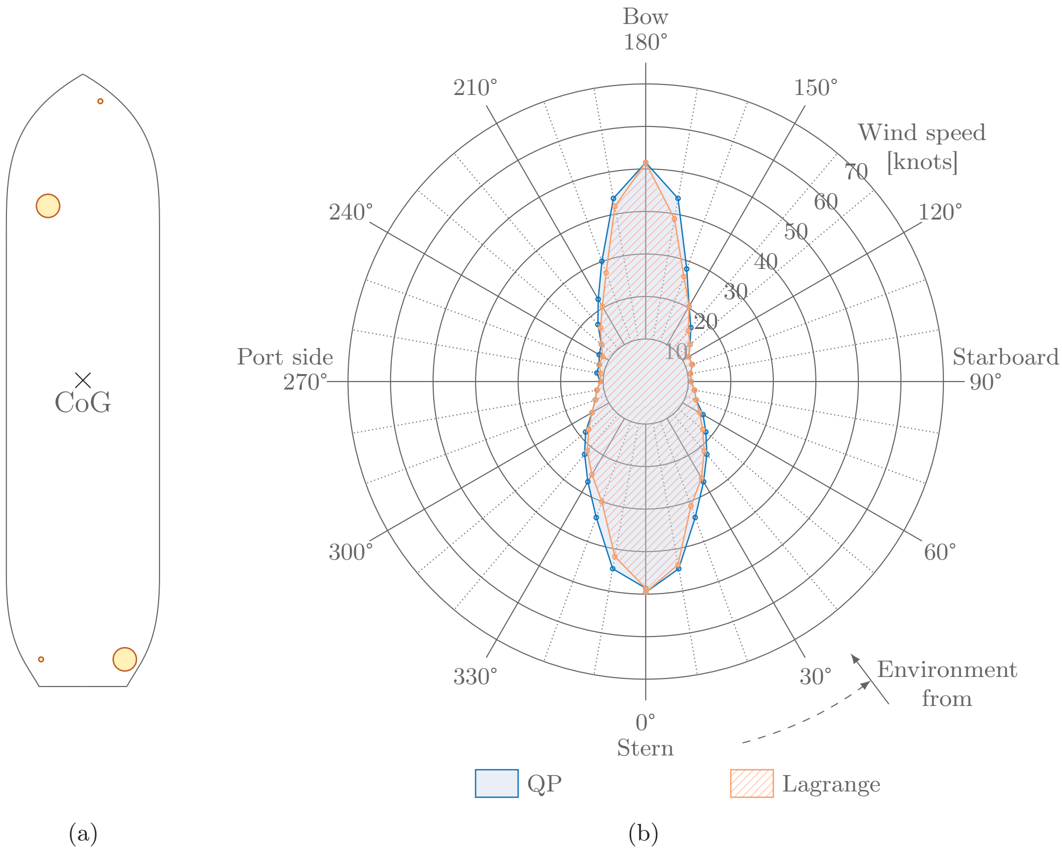


Figure 6.1: A DP Capability Plot: (a) Schematic actuator layout of the ship, showing the modeled thrust region for each thruster; (b) The corresponding DP capability plot.

Thrust linking or a thruster relation can eliminate these sort of solutions from appearing and it is therefore advised to use these methods to avoid these unwanted solutions. It also shows that the generated solutions need not change continuously, but can jump quite rapidly from one thruster configuration to another. This is good to know for the time domain simulations, where the dynamic thrust regions are necessary to reduce jumpy thruster behavior.

Figure 6.2(c) also shows that little perturbations in the environmental conditions can lead to large differences in the power optimal thrust allocation solution. Each red line shows all the solutions for a fixed environment direction and increasing wind speeds. Clearly there are occasions where a little wind speed change makes the solution jump from reverse to forward mode for the propeller/rudder pairs. This indicates that small environmental changes do not always lead to a small changes in the solution. This result also proves that the dynamic regions for the time domain simulations are really necessary.

Note that the DP capability of this ship actually has a non-smooth form, because of the two rudders, having a line shaped thrust region for their reverse mode. When the resolution of the DP plot is increased, a sharp spike will appear in the DP plot for environments coming from the stern side of the ship.

6.1.3 Extended testing

In figure 6.3 another DP capability plot is shown, testing more exotic thrust regions. In this case, $1 \cdot 1 \cdot 3 \cdot 3 \cdot 2 \cdot 1 \cdot 2 \cdot 2 = 72$ subproblems need to be solved for each DP problem (this is the number of possible convex thrust region combinations).

6.2 Time Domain Simulations

Using time domain simulations the thrust allocators were tested on their quality and practical use as online thrust allocators. For the QP thrust allocator, additional dynamic thrust regions for each thruster were added to model their physical limitations. For the Lagrange multiplier method the dynamic thruster limitations are coded next to the saturation handling. The simulation environment uses a great deal of the DP tools environment developed inhouse by GustoMSC, including JONSWAP spectra and Kalman filtering. As it is out of the scope of this thesis to explain the whole set-up simulation environment, only the results regarding the thrust allocation algorithms are discussed.

6.2.1 Optimal thrust allocation

Using the Lagrange multiplier method in the simulation environment immediately showed major problems. At some point in the simulation (using only azimuth thrusters for the ship) all the azimuth thrusters turned around 360°. This is a highly unwanted effect and could be very dangerous in real life situations. Explaining the phenomena brings us again to the saturation handling. Apparently the order in which the saturation handling is commenced, can have a huge impact on the overall behavior of the online thrust allocator.

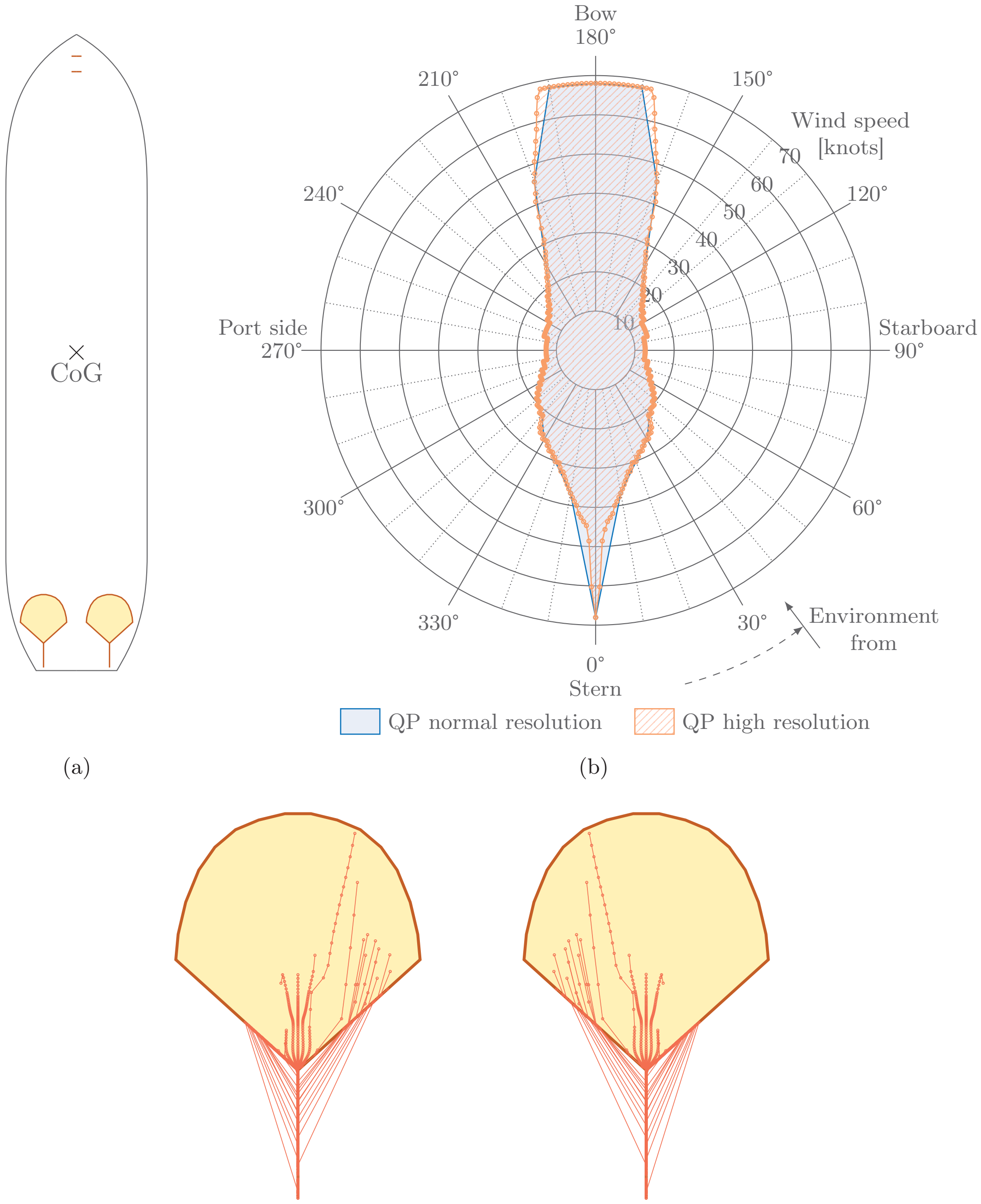
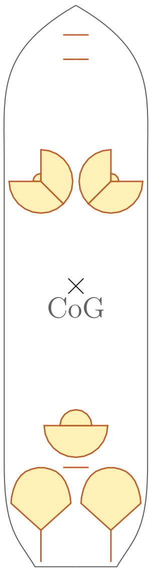
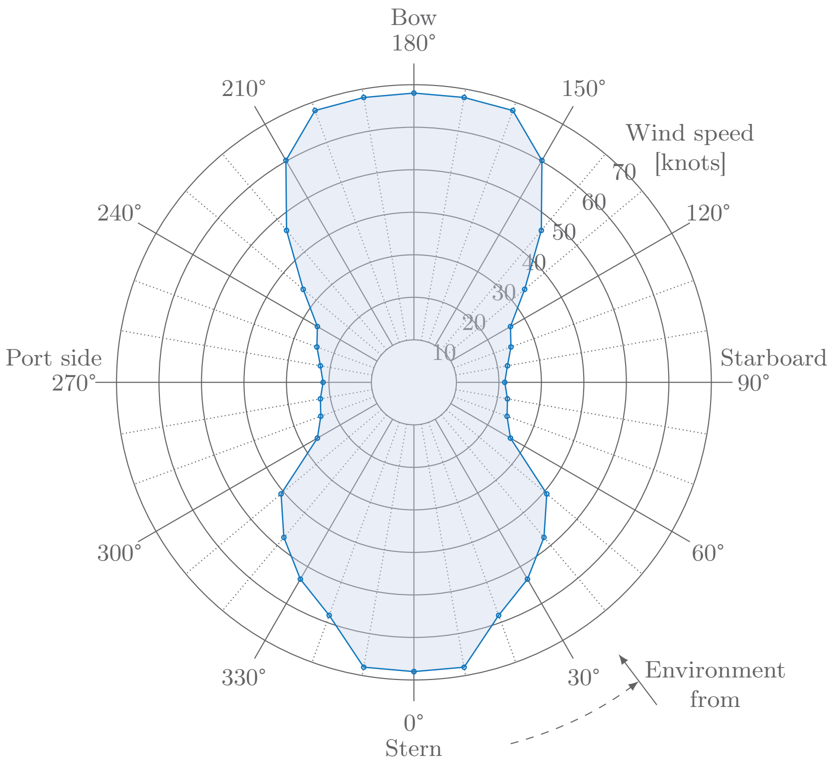


Figure 6.2: A DP Capability Plot: (a) Schematic actuator layout of the ship, showing the modeled thrust region for each thruster; (b) The corresponding DP capability plot; (c) Rudder states for a whole DP plot, where for every environment direction, increasing environments are connected.



(a)



(b)

Figure 6.3: A DP Capability Plot: (a) Schematic actuator layout of the ship, showing the modeled thrust region for each thruster; (b) The corresponding DP capability plot.

The fact that the Lagrange allocator does not generate optimal solutions, apparently results in an growing error, leading to the 360°behavior.

As the QP thrust allocator generates optimal solutions, the 360°error was not present. Although the QP thrust allocator looked as if it was reacting a little bit slower than the Lagrange allocator, it did not behave strange. Because of the linearly approximated thrust regions, the allocator prefers to be on the vertices of the thrust regions at all time, when τ_{ref} is way too large for the DP system to handle. This has a nice advantage, making the allocator less jumpy, where the Lagrange allocator constantly tries to make even the smallest control adjustment.

6.2.2 Forbidden zones

The forbidden zones did give some problems. The QP allocator seems to hang at forbidden zones, because at zero thrust the allocator will not turn the azimuth thruster. If this is really the desired behavior for a forbidden zone will depend on the application. The behavior can be easily changed by defining very small thrust regions at the forbidden zones, allowing the allocator to rotate the thruster, when it is delivering a very small amount of thrust (see also figure 6.3(a)).

6.3 QPC vs OOQP

For almost all of the thrust allocation problems, QPC outperforms OOQP when is comes to computation time. They both generate the same results (as should be the case), although it is advised to use problem scaling with both of these solvers, because numerical problems can occur when this is not done.

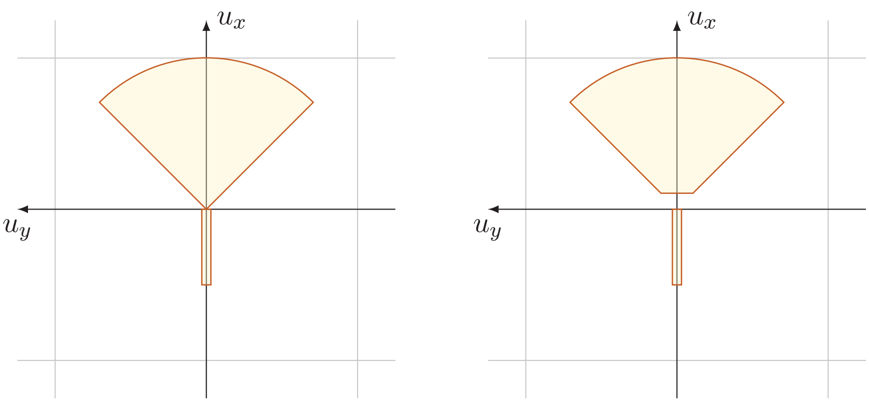
Chapter 7

Discussion

This chapter will discuss some of the problems, and ideas, regarding the described thrust allocation methods.

7.1 Rudder Flapping

Rudder flapping happens when the rudder is working at near zero thrust. Because of numerical problems near zero, the rudder can be flapping from one side to the other. A way to minimize this rudder flapping is to remove a thrust region part near zero thrust as can be seen in figure 7.1. Other solutions can be found in [Lea08].



(a) Normal main propeller & rudder thrust region.

(b) Modified main propeller & rudder thrust region.

Figure 7.1: Modified main propeller & rudder thrust region to reduce rudder flapping near zero thrust.

7.2 Thruster dynamics model

To get a more accurate description of the movement of the thrusters, a thruster dynamics model can be used to create the dynamic thrust regions (see figure 7.2. This thrust dynamics model can take into account the momentum of a thruster, resulting in a more realistic behavior of the thrusters in real time simulations.

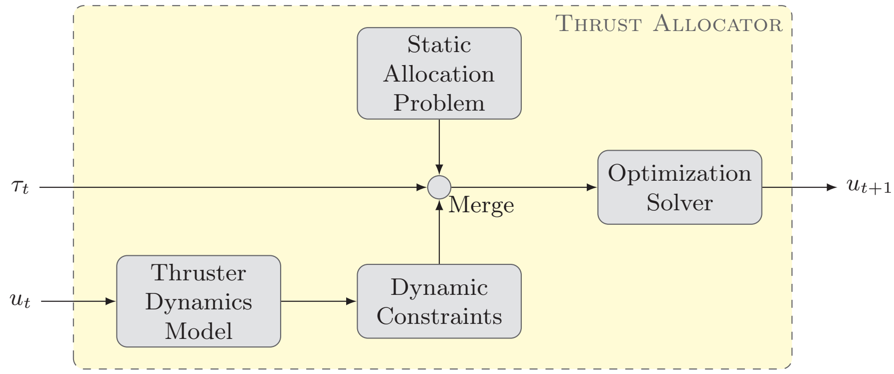


Figure 7.2: QP thrust allocator block diagram including a thruster dynamics model.

7.3 Linear Programming

The only reason for using quadratic programming techniques is because of the quadratic approximation of the power/thrust relation. This relation can of course also be approximated by a linear function. The approximation error will of course be larger, but a linear object/cost function allows us to use linear programming techniques, which are generally considered to be more mature and robust than quadratic programming techniques.

Because u can contain negative elements, the linear optimization problem has the form:

$$\begin{aligned} &\min_u w^T|u|, \\ &\text{s.t. } Bu = \tau_{\text{ref}}, \\ &\qquad Au \leq b, \\ &\qquad -\infty \leq u \leq \infty. \end{aligned} \tag{7.1}$$

Because of the absolute value in the object/cost function the problem can not be solved in this form by a linear optimization solver. The problem can however be rewritten as:

$$\begin{aligned} &\min_{u,v} w^T v, \\ &\text{s.t. } Bu = \tau_{\text{ref}}, \\ &\qquad Au \leq b, \\ &\qquad -v \leq u \leq v. \end{aligned} \tag{7.2}$$

Here u is closed in by $-v$ and $+v$, for $v > 0$ and models the absolute value of u. This form is compatible with common linear constrained optimization solvers. The problem has the following standard LP form:

$$
\begin{aligned}
&\min_{u,v} \begin{bmatrix} 0 & w \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \\
&\text{s.t. } \begin{bmatrix} B & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \tau \\
&\begin{bmatrix} A & 0 \\ -1 & -1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \leq \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix}
\end{aligned} \tag{7.3}
$$

7.4 LMI

Linear Matrix Inequalities are used to define Semi-Definite Programming (SDP) problems that form a bigger class op optimization problems. The QP problem

$$
\begin{aligned}
&\min_{x} x^T Q^T Q x + c^T x, \\
&\text{s.t. } Ax \geq b,
\end{aligned}
$$

can be written as a SDP problem, by using the Schur complement:

$$
\gamma \geq x^T Q^T Q x + c^T x \quad \Leftrightarrow \quad F_Q(\gamma, x) = \begin{bmatrix} \gamma - c^T x & x^T Q^T \\ Qx & I \end{bmatrix} \geq 0.
$$

Now the equivalent SDP problem is

$$
\begin{aligned}
&\min_{\gamma,x} \gamma, \\
&\text{s.t. } F(\gamma, x) \geq 0,
\end{aligned}
$$

where $F(\gamma, x) = \text{block diag}[F_Q(\gamma, x), A_i x - b_i]$, with A_i the ith row of A.

LMI's may have some advantages over QP problems. Especially when only the existence of a solution is needed (as is the case for creating DP capability plots) LMI's seem very promising, because this translates into checking the positive definiteness of a matrix. The 2-norm constraints seen for azimuth thrusters also should fit nice into SDP formulations. Maybe even more accurate power/thrust approximations are possible by using higher order convex polynomials.

Chapter 8

Conclusions

Although the power optimal thrust allocation problem may seem simple at first, finding a practical solution method for this problem proved to be very challenging. The amount of factors that can play a role in the problem also makes the implementation very difficult. Let alone the amount of input variables needed to define the problem. Because the problems can get quite complex, the amount of exceptions that need to be checked in the code also grows quite large. Compiling and interfacing the QP solvers also took more time than expected.

Apart from these implementation difficulties, it can be concluded that the QP thrust allocation algorithm is a big improvement on the Lagrange thrust allocator. Fixed/tunnel thrusters, azimuth thrusters with forbidden/spoil zones and propeller/rudder pairs can be used for power optimal thrust allocation. Even thrusters can be linked with each other in various way's, making it possible to model the propulsion system on a ship more accurate. Problem scaling makes the algorithm less sensitive to numerical problems, yielding in a robust solution for static and dynamic thrust allocation applications. For online use the presented dynamic thrust regions work rather well, although there is still room for improvements. Especially the rudder flapping at zero, or the forbidden zones for azimuth thrusters can be sources of problems. Altogether Quadratic Programming seems to be at the right spot for Dynamic Positioning.

Bibliography

[Bra03] David Bray. *Dynamic Positioning*, volume nine of *Oilfield Seamanship*. Oilfield Publications Limited, Ledbury, second edition, 2003. Available from: http://www.oilpubs.com.

[Fos02] Thor I. Fossen. *Marine Control Systems, Guidance, Navigation, and Control of Ships, Rigs and Underwater Vehicles*. Marine Cybernetics, Trondheim, first edition, 2002. Available from: http://www.marinecybernetics.com.

[3] The International Marine Contractors Association (IMCA). *Specification For DP Capability Plots*, June 2000. Available from: http://www.imca-int.com.

[JFB04] Tor A. Johansen, Thor I. Fossen, and Svein P. Berge. Constrained Nonlinear Control Allocation with Singularity Avoidance using Sequential Quadratic Programming. *IEEE Transactions on Control Systems Technology*, 12(1):211–216, January 2004.

[JFT05] Tor A. Johansen, Thor I. Fossen, and Petter Tøndel. Efficient Optimal Constrained Control Allocation via Multi-parametric Programming. *Journal of guidance, control, and dynamics*, 28(3):506–515, 2005.

[JFTF08] Tor A. Johansen, Thomas P. Fuglseth, Petter Tøndel, and Thor I. Fossen. Optimal constrained control allocation in marine surface vessels with rudders. *Control Engineering Practice*, 16(4):444–445, April 2008.

[Lea08] J. A. Leavitt. Optimal Thrust Allocation in a Dynamic Positioning System. Number 077. The Society of Naval Architects & Marine Engineers (SNAME), Annual Meeting & Expo: Technical Program, October 2008.

[ML06] J. Meek and G. H. G. Lagers. An Introduction to Offshore Structures and Systems. Lecture notes for the course OE4603, Delft University of Technology, Delft, November 2006.

[Mor78] Max J. Morgan. *Dynamic Positioning of Offshore Vessels*. PPC Books Division, The Petroleum Publishing Co., Tulsa, first edition, 1978.

[MT07] Anthony F. Molland and Stephen R. Turnock. *Marine Rudders and Control Surfaces, Principles, Data, Design and Applications*. Butterworth-Heinemann (An imprint of Elsevier Ltd.), Oxford, first edition, 2007. Available from: http://books.elsevier.com.

[Rut08] Eivind Ruth. *Propulsion control and thrust allocation on marine vessels*. PhD thesis, Norwegian University of Science and Technology, Trondheim, October 2008.

[Sni05] Jeroen Snijders. Wave Filtering and Thruster Allocation for Dynamic Positioned Ships. Master's thesis, Delft University of Technology, November 2005.

[STJ06] Jørgen Spjøtvold, Petter Tøndel, and Tor A. Johansen. Decomposing Linear Control Allocation Problems. In *Proceedings of the* 17^{th} *International Symposium on Mathematical Theory of Networks and Systems*, number SINTEF S1179, pages 1201–1207, July 2006.

[vD03] Ton van den Boom and Bart De Schutter. Optimization in Systems and Control. Lecture notes for the course ET4-101, Delft University of Technology, Delft, January 2003.

[WGT08] C. A. Willemse, H. T. Grimmelius, and A. R. Tjallema. Dynamic Positioning. Lecture notes for the course OE5663, Delft University of Technology, Delft, September 2008.