JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 28, No. 3, May–June 2005

Efficient Optimal Constrained Control Allocation via Multiparametric Programming

Tor A. Johansen,* Thor I. Fossen,† and Petter Tøndel‡
Norwegian University of Science and Technology, N-7491 Trondheim, Norway

Constrained control allocation is studied, and it is shown how an explicit piecewise linear representation of the optimal solution can be computed numerically using multiparametric quadratic programming. Practical benefits of the approach include simple and efficient real-time implementation that permits software verifiability. Furthermore, it is shown how to handle control deficiency, reconfigurability, and flexibility to incorporate, for example, rate constraints. The algorithm is demonstrated on several overactuated aircraft control configurations, and the computational complexity is compared to other explicit approaches from the literature. The applicability of the method is further demonstrated using overactuated marine vessel dynamic position experiments on a scale model in a basin.

I. Introduction

CONTROL allocation is an important part of many ship control systems,[1,2] flight control systems,[3,4] and other overactuated control applications, for example, Ref. 5. The control allocation module will send control commands to the individual actuators (such as thrusters, rudders, and propellers in marine vessels, or ailerons, elevators, flaps, rudders, and vector thrust devices in aircraft) to produce the required forces and moments (hereafter denoted generalized forces) commanded from a higher level control system or pilot during manual operation.

Such overactuated control allocation problems are naturally formulated as optimization problems because one usually wants to take advantage of all available degrees of freedom to minimize power consumption, drag, tear/wear, and other costs related to the use of control, subject to constraints such as actuator rate and position limitations (Refs. 3, 6, and 7, for example). In general, this leads to a constrained optimization problem that is hard to solve using state-of-the-art iterative numerical optimization software at a high sampling rate in a safety-critical real-time system with limiting processing capacity and high demands for software reliability. Still, real-time iterative optimization solutions have been proposed[6–11] for some large-scale problems[5] as well, although their worst-case computational complexity may be disadvantageous and software verification is a complicated issue. Many practical solutions have, therefore, been focused on simplified optimization problems where simpler semi-explicit solutions can be found and implemented efficiently by combining simple matrix computations, logic, and filtering.[1,6,12,13]

Here we consider overactuated systems where each of the individual linear actuators produces a bounded force in a fixed direction. We propose a computationally efficient approach to exact real-time computation of the optimal solution (in the sense of minimizing a quadratic objective function) to a class of control allocation problems. We show that constrained control allocation problems lead to a linearly constrained multiparametric quadratic program (MPQP). This means that the QP is parameterized by problem data such as the commanded generalized force and possibly also constraint limits and criterion parameters. When recently developed MPQP solver[14,15] are used, the exact solution to such problems can be precomputed off-line in the explicit form of a piecewise linear (PWL) function of the problem parameters, defined on a polyhedral partition of the space of generalized forces and possibly other parameters. In real-time implementation, the optimal constrained control allocation can, thus, be computed via the evaluation of a PWL function. These computations can be organized as a binary tree search,[16] with worst-case computational complexity that is logarithmic in the number of polyhedral regions. A considerate advantage of the approach is that the unconstrained solution coincides with the common generalized inverse. Dividing the computations into off-line and online parts in this fashion has the advantages of a very simple and efficient online implementation. The main disadvantage is that it is not straightforward to account for, in real time, changes in the control allocation problem, for example, due to actuator failure. Still, such problems can be handled with our approach, by augmenting the parameter vector, at the cost of increasing the complexity of both the off-line and online computations.

In Refs. 3, 4, and 17, the so-called direct control allocation method is suggested, based on precomputing a polyhedral representation of the attainable moment set, the set of generalized forces that can be attained using feasible controls. The advantage of this approach is that a feasible control allocation can be found if it exists. This is a property shared by the method of this paper, as well as approaches based on real-time optimization, for example, Refs. 6 and 7. The main problems of the real-time algorithms associated with direct control allocation are that the worst-case computational complexity may be large due to the combinatorial nature of the problem. Some modifications and extensions are given in Refs. 18–20, which allow the computational complexity to be reduced. The suggested approach provides a significantly more efficient algorithm for implementing the solution than the null-space intersection method,[21] which suffers from very high worst-case computational complexity because it must compare in real time a larger combinatorial number of possible combinations of active constraints (saturated actuators) to determine the optimal control allocation.

This paper is organized as follows. In Sec. II, the control allocation problem is introduced, and it is formulated as a multiparametric program in Sec. III. The method is illustrated on some aircraft control allocation problems from the existing literature in Sec. IV and marine surface vessel dynamic positioning control allocation experiments using a scale model in a basin in Sec. V. Some conclusions are given in Sec. VI.

II. Optimization Formulation of the Control Allocation Problem

Let the commanded forces in the body-fixed coordinate frame (x, y, z) be denoted (τ_x, τ_y, τ_z) and the commanded body-axis

Received 31 October 2002; accepted for publication 21 April 2004.

*Professor, Department of Engineering Cybernetics; Tor.Arne.Johansen@itk.ntnu.no.
†Professor, Department of Engineering Cybernetics. Member AIAA.
‡Postdoc, Department of Engineering Cybernetics.

moments in roll, pitch, and yaw be denoted $(\tau_\phi, \tau_\theta, \tau_\psi)$. These are stacked in a vector of commanded generalized forces $\tau = (\tau_x, \tau_y, \tau_z, \tau_\phi, \tau_\theta, \tau_\psi)^T$. Assume the system is equipped with n linear actuators with control inputs u_i. (If each actuator is characterized by a monotonic nonlinearity, it is implicitly assumed that this nonlinearity is inverted.) This leads to a relationship between the controls $u = (u_1, u_2, \ldots, u_n)^T$ and the generalized forces $\tau \in \mathbb{R}^m$ of the following form:

$$Bu = \tau \tag{1}$$

where $B \in \mathbb{R}^{m \times n}$. In many control allocation applications, not all six components of τ are specified. For example, in aerospace applications one is often only concerned with the three body-axis moments,[4] whereas in dynamic position applications involving marine surface vessels one is usually concerned only with the three horizontal plane components $\tau = (\tau_x, \tau_y, \tau_\psi)$ (Ref. 2).

When constraints are neglected, the common solution to this problem is the generalized inverse, defined as $B^+ = R^{-1}B^T(BR^{-1}B^T)^{-1}$ assuming the configuration is nonsingular such that B has full rank,[1,3,4]

$$u = B^+\tau \tag{2}$$

which solves the least-squares problem

$$\min_u u^T Ru \quad \text{subject to} \quad Bu = \tau \tag{3}$$

and $R \in \mathbb{R}^{n \times n}$, where $R > 0$ is a weighting matrix. The most important feature of this approach is that it admits an explicit solution (2) that is linear and, therefore, computationally efficient and easily implemented. It is, however, of interest to consider more advanced optimization formulations that allow more general cost indices and in particular considers the presence of constraints on u (Refs. 3, 4, 6, 7, and 17):

$$u_{\min} \le u \le u_{\max} \tag{4}$$

where $u_{\min}, u_{\max} \in \mathbb{R}^n$, where the inequalities are to be considered elementwise.

When taking constraints on u into consideration, one can in general identify two different objectives for the control allocation.[7]

First is control sufficiency. This means that there exists a feasible u that attains the desired generalized force τ. In this case, one would, in addition to satisfying Eq. (1), also desire to minimize some norm on u, to minimize the cost of control,

$$\min_u \|u\|, \quad \text{subject to} \quad Bu = \tau, \qquad u_{\min} \le u \le u_{\max} \tag{5}$$

The second is control deficiency. When a feasible u that solves Eq. (1) does not exist, the difference $Bu - \tau$ should be minimized in some sense. The direct control allocation does this by finding a solution that preserves the direction of the generalized force vector. Alternatively, for example, in Ref. 7, a norm of this difference is minimized:

$$\min_u \|Bu - \tau\|, \quad \text{subject to} \quad u_{\min} \le u \le u_{\max} \tag{6}$$

We will in this paper use weighted 2-norms in both Eqs. (5) and (6). [The problem would still be suited for solution by multiparametric programming (giving PWL explicit solutions) if ∞- or 1-norms were used in the criteria, but then the resulting problems would be multiparametric linear programs.[22]] Unlike when using the ∞-norm or 1-norm, as in Refs. 6 and 7, this gives a unique solution for a given τ. Another useful property gained by using the 2-norm is that the resulting control becomes a continuous function of τ (next section). Moreover, for many control surfaces and propulsion devices, the 2-norm is a reasonable approximation to the energy used. The reason for including weighting matrices is to provide different costs for the different controls and to prioritize among the axis in τ. In the sequel, let $\mathcal{T}$ denote the polyhedral set of τ such that there exists a feasible solution to Eq. (5). ($\mathcal{T}$ is sometimes referred to as the attainable moment set in the literature.) $\mathcal{T}$ can be obtained by a projecting the set of feasible (u, τ) in Eq. (5) into τ space.

The two objectives (5) and (6) will in this paper be combined into a single optimization problem similar to the mixed optimization problem formulated in Ref. 7,

$$\min_{u,s} \tfrac{1}{2}(s^T Qs + u^T Ru) \tag{7}$$

subject to

$$Bu = \tau + s \tag{8}$$

$$u_{\min} \le u \le u_{\max} \tag{9}$$

Here s is a vector of slack variables used to penalize $Bu - \tau$. Note that by combining the two objectives in this fashion the solution can have a nonzero s even when $\tau \in \mathcal{T}$. The weighting matrix Q should, however, be chosen much larger than R, to prioritize objective 2 to objective 1; thus, $s \approx 0$ whenever $\tau \in \mathcal{T}$. See Sec. III.B for a discussion on how one could guarantee $s = 0$ whenever $\tau \in \mathcal{T}$. Note that one can off-line compute the largest value of s for $\tau \in \mathcal{T}$ from the explicit solution by solving a linear program (LP) for each polyhedral region in the explicit solution. If this s is unacceptably large, one should increase the weighting matrix Q.

The constraints handled by the methods used in this paper can have a more general form than stated here. For instance, rate constraints or constraints involving multiple actuators can easily be incorporated. See Sec. III.C for more discussion on this.

III. Obtaining an Explicit Solution

The optimization problem (7–9) can for a given τ be considered as a QP. One could, therefore, consider solving this QP for every sample to obtain the optimal solution to problem (7–9) (Ref. 6). Here we will instead obtain an explicit solution to problem (7–9) as a function of τ.

When $z = (u^T, s^T)^T$ and $x = \tau$, it is straightforward to reformulate problem (7–9) as follows:

$$\min_z \tfrac{1}{2} z^T Hz \tag{10}$$

subject to

$$G_1 z = W_1 + S_1 x \tag{11}$$

$$G_2 z \le W_2 + S_2 x \tag{12}$$

where $H = \mathrm{diag}(R, Q)$, $G_1 = (B \mid -I_{n\times n})$, $G_2 = (I_{m\times m} | 0_{m\times n}; -I_{m\times m}\, 0_{m\times n})$, $W_1 = 0_{n\times 1}$, $W_2 = (u_{\max}^T, u_{\min}^T)^T$, $S_1 = I_{n\times n}$, and $S_2 = 0_{2m\times n}$, where diag denotes a block-diagonal matrix. Because $Q > 0$ and $R > 0$ implies $H > 0$, this defines a convex quadratic program in z parameterized by x. It has recently been found that the solution to such problems is a continuous PWL function $z^*(x)$ defined on a polyhedral partition of the parameter space.[23] Moreover, an exact representation of this PWL function on any compact polyhedral subset of the parameter space can be precomputed off-line using multiparametric quadratic programming algorithms.[16,23] Consequently, it is not necessary to solve the QP (7–9) in real time for the current value of x. It suffices to evaluate the precomputed PWL function $z^*(x)$.

Compared to solving the QP with real-time optimization software, this explicit solution has several advantages. First, one can a priori state the maximum number of arithmetic operations necessary to compute the solution in real time, which is in general much less than with real-time optimization. The online implementation can be made using only a few lines of code, which can be formally verified in a safety-critical application. Moreover, the online implementation can be made using fixed-point arithmetic, giving further advantages in terms of hardware complexity and computational speed.

The MPQP algorithm, is briefly presented in Appendix A, and an algorithm for efficient evaluation of such PWL functions using a binary search tree representation is briefly described in Appendix B.

Note that the results can be extended to more complicated situations where in addition the directions of the forces produced by the actuators may be optimized, by using the idea in Ref. 8; see Ref. 24 for an example of this.

A. Control Reconfiguration

One disadvantage of the explicit solutions approach to optimization problems presented in this paper compared to solving the optimization problem in real time is less flexibility of reconfiguration of the control problem. For instance, in the case of an actuator failure or other changed operating conditions, one could when using real-time optimization reformulate or simply change some parameters in the optimization problem to account for these changes. Such reconfigurations can also be handled by explicit solutions if the parameter vector $\boldsymbol{x}$ can be extended to include other elements that appear linearly in the constraints (or quadratically/linearly in the objective function) than the desired generalized force $\boldsymbol{\tau}$. For example, one can obtain real-time reconfigurability of the control input bounds by introducing $\boldsymbol{u}_{\min}$ and $\boldsymbol{u}_{\max}$ into the parameter vector, that is, $\boldsymbol{x} = (\boldsymbol{\tau}^T, \boldsymbol{u}_{\min}^T, \boldsymbol{u}_{\max}^T)^T$. The MPQP (10–12) would now be defined by $H = \mathrm{diag}(R, Q)$, $G_1 = (B|-I_{n\times n})$, $G_2 = (I_{m\times m}|0_{m\times n}; -I_{m\times m}0_{m\times n})$, $W_1 = 0_{n\times 1}$, $W_2 = 0_{2m,1}$, $S_1 = (I_{n\times n}|0_{n\times 2m})$, and

$$S_2 = \begin{bmatrix} 0_{m\times n} & I_{m\times m} & 0_{m\times m} \\ 0_{m\times n} & 0_{m\times m} & I_{m\times m} \end{bmatrix}$$

One would in addition require $\boldsymbol{s} \geq 0$. The optimal solution $z^*(\boldsymbol{x})$ is still computed as an explicit PWL solution. The complexity of the resulting explicit solution will typically be somewhat increased in terms of the number of polyhedral regions in the PWL solution and in a higher dimensional space. However, if it is sufficient to consider only a finite number of values of $(\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max})$, one could instead consider solving and storing different PWL solutions for the different combinations of $(\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max})$.

B. Exact Penalty Functions

As mentioned in the preceding section, QP (7–9) can give a solution with $\boldsymbol{s} \neq 0$ even if $\boldsymbol{\tau} \in \mathcal{T}$. This can be avoided, by using so-called exact penalty functions. Basically, if replacing problem (7) with

$$\min_{u,s} \left(\tfrac{1}{2}\boldsymbol{u}^T R\boldsymbol{u} + \rho 1^T \boldsymbol{s} \right) \tag{13}$$

where 1 is a vector of ones and ρ is a sufficiently large scalar one can guarantee that the solution to problem (13), subject to problems (8) and (9) is exactly the same as the solution to problem (5) (using $\|\boldsymbol{u}\|^2 = \frac{1}{2}\boldsymbol{u}^T R\boldsymbol{u}$) if $\boldsymbol{\tau} \in \mathcal{T}$, (Ref. 25, Theorem 14.3.1). (See Ref. 26 for how such a ρ can be computed.) The MPQP (13), (8), (9) is convex, but not strictly convex, and can be solved by using the method in Ref. 27.

C. Rate Constraints and Other Extensions

When the control input is included at the previous sample $\boldsymbol{u}_{\text{prev}}$ in the parameter vector, that is, $\boldsymbol{x} = (\boldsymbol{\tau}^T, \boldsymbol{u}_{\text{prev}}^T)^T$, rate constraints on the actuators can be handled by our method, by adding the constraints

$$|\boldsymbol{u}_i - \boldsymbol{u}_{\text{prev},i}| \leq \Delta\boldsymbol{u}_{\max,i} \tag{14}$$

Note that if the reconfiguration scheme of Sec. III.A is used, such that $\boldsymbol{u}_{\min}$ and $\boldsymbol{u}_{\max}$ are already in the parameter vector, rate constraints can be obtained without including $\boldsymbol{u}_{\text{prev}}$ in the parameter vector, by choosing the appropriate values for the parameters $\boldsymbol{u}_{\min}$ and $\boldsymbol{u}_{\max}$ when evaluating the PWL function:

$$\boldsymbol{u}_{\min} = \boldsymbol{u}_{\text{prev}} - \Delta\boldsymbol{u}_{\max}, \qquad \boldsymbol{u}_{\max} = \boldsymbol{u}_{\text{prev}} + \Delta\boldsymbol{u}_{\max} \tag{15}$$

One could also give further extensions to these methods, possibly including nonlinearities. In this case, the problem formulation typically leads to a multiparametric nonlinear program (MPNLP) that is harder to solve, but admits approximate solutions with the same implementation advantages as MPQPs.[28]

IV. Aircraft Control Allocation Examples

We consider three different control configurations of an F18 aircraft: 1) 5 control surfaces (left and right horizontal tails, left and right ailerons, and rudder) described in Ref. 21; 2) 7 control surfaces (left and right horizontal tails, left and right trailing-edge flaps, left and right ailerons, and rudder) described in Ref. 4; and 3) 10 control surfaces (left and right horizontal tails, left and right trailing-edge flaps, left and right aileron, three thrust-vectoring control, and rudder) described in Ref. 18. Thus, with 5 control surfaces,

$$B_5 = \begin{pmatrix} 6.47\times10^{-4} & -6.47\times10^{-4} & 6.00\times10^{-4} & -6.00\times10^{-4} & 5.83\times10^{-5} \\ -7.35\times10^{-3} & -7.35\times10^{-3} & -6.50\times10^{-4} & -6.50\times10^{-4} & 0 \\ 0 & 0 & -1.50\times10^{-4} & 1.50\times10^{-4} & -6.67\times10^{-4} \end{pmatrix}$$

$$\boldsymbol{u}_{\min,5} = (-24, -24, -25, -25, -30)^T$$

$$\boldsymbol{u}_{\max,5} = (24, 24, 25, 25, 30)^T$$

with 7 control surfaces,

$$B_7 = \begin{pmatrix} 2.38\times10^{-4} & -2.38\times10^{-4} & 1.23\times10^{-3} & -1.23\times10^{-3} & 4.18\times10^{-4} & -4.18\times10^{-4} & 3.58\times10^{-5} \\ -6.98\times10^{-3} & -6.98\times10^{-3} & 9.94\times10^{-4} & 9.94\times10^{-3} & -5.52\times10^{-4} & -5.52\times10^{-4} & 0 \\ -3.09\times10^{-4} & 3.09\times10^{-4} & 0 & 0 & -1.74\times10^{-4} & 1.74\times10^{-4} & -5.62\times10^{-4} \end{pmatrix}$$

$$\boldsymbol{u}_{\min,7} = (-24, -24, -8, -8, -30, -30, -30)^T$$

$$\boldsymbol{u}_{\max,7} = (24, 24, 8, 8, 30, 30, 30)^T$$

and finally with 10 control surfaces,

$$B_{10} = \begin{pmatrix} -7.64\times10^{-4} & 7.64\times10^{-4} & -1.02\times10^{-3} & 1.02\times10^{-3} & 2.92\times10^{-4} & -1.09\times10^{-3} & 1.09\times10^{-3} & 5.09\times10^{-4} & 1.74\times10^{-7} & 1.74\times10^{-4} \\ -9.30\times10^{-3} & -9.30\times10^{-3} & -1.13\times10^{-3} & -1.13\times10^{-3} & 0 & 1.09\times10^{-3} & 1.09\times10^{-3} & 1.74\times10^{-7} & 6.20\times10^{-3} & 1.74\times10^{-7} \\ 1.92\times10^{-4} & -1.92\times10^{-4} & 6.82\times10^{-5} & -6.82\times10^{-5} & -1.29\times10^{-3} & 0 & 0 & 5.23\times10^{-6} & 1.74\times10^{-7} & 2.59\times10^{-3} \end{pmatrix}$$

$$\boldsymbol{u}_{\min,10} = (-24, -24, -30, -30, -30, -8, -8, -30, -30, -30)^T$$

$$\boldsymbol{u}_{\max,10} = (10.5, 10.5, 30, 30, 30, 45, 45, 30, 30, 30)^T$$

Table 1 Complexity of the explicit piecewise linear solution for control allocation problems of different complexity

	Number of actuators		
Complexity	5	7	10
Number of regions	31	372	1194
Number of arithmetic operations	86	154	193
Off-line computation time, s	5	80	471
Stored numbers	1,771	33,711	137,903

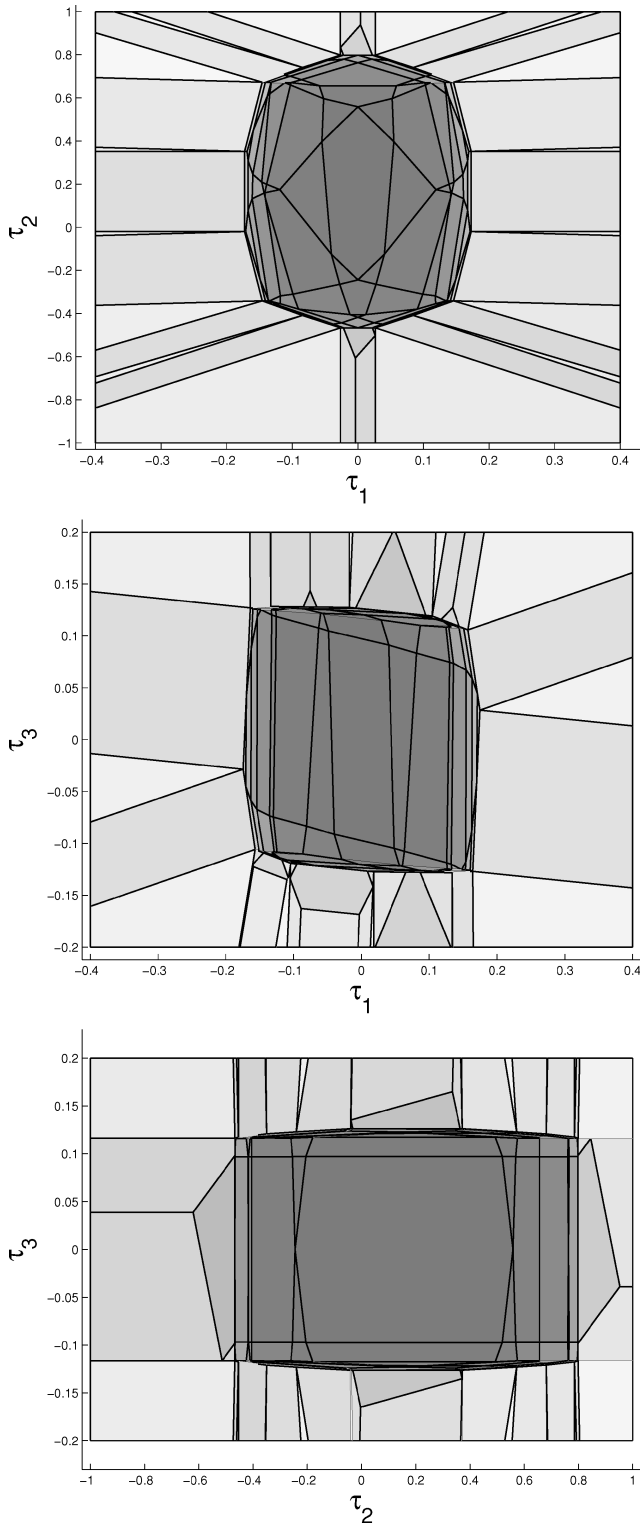

Fig. 1 Planar intersection $(\tau_1, \tau_2, 0)$, $(\tau_1, 0, \tau_3)$, and $(0, \tau_2, \tau_3)$ of the underlying partition of the PWL optimal control allocation for the F18 configuration with 10 control surfaces.

Table 2 Comparison of real-time computational complexity of constrained control allocation algorithms

	Number of real-time FLOPS		
Method	5 actuators	7 actuators	10 actuators
MPQP	86	154	193
Ref. 4	305	637	1360
Ref. 21	1,500	39,396	2,483,040

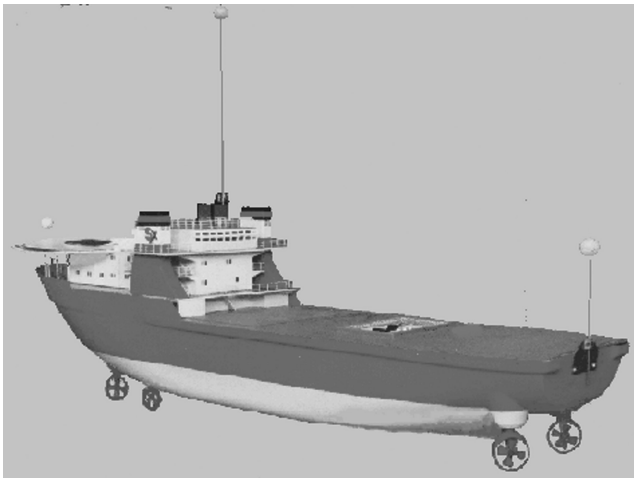

Fig. 2 Cybership 1, a 1:70 scale model of a supply ship with four azimuth thrusters.

The generalized forces in $\boldsymbol{\tau}$ are the three body-axis moments. Solving QP (7–9), the MPQP algorithm gives a PWL function $u^*(\boldsymbol{\tau})$. The complexity of the PWL control allocation functions corresponding to these three cases, and their computation are summarized in Table 1. The row denoted number of arithmetic operations refers to online/real-time computations, whereas the off-line computation times includes generating the explicit solution and its binary tree representation. We also compare the real-time computations with the methods of found in Refs. 4 and 21 in Table 2 and notice that, in addition to allowing the minimization of a quadratic cost, our approach is typically significantly more computationally efficient. The floating-point operations per second (FLOPS) given in Table 2 are the worst-case numbers. For our method, the worst-case number of FLOPS can easily be obtained as a simple function of the depth of the search tree.[16] This worst-case number does depend on the particular B matrix, but a change of this matrix would not considerably change the complexity of the off-line or online problems to be solved. Note that the implementation of the direct allocation method found in Ref. 20 probably would speed up the online evaluation considerably compared to that in Ref. 4. Moreover, as pointed out in Ref. 29, the direct allocation method can be divided into off-line and online computations using methods from computational geometry.

Figure 1 shows the underlying partition of the PWL functions $\boldsymbol{u}_i^*(\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, 0)$, $\boldsymbol{u}_i^*(\boldsymbol{\tau}_1, 0, \boldsymbol{\tau}_3)$, and $\boldsymbol{u}_i^*(0, \boldsymbol{\tau}_2, \boldsymbol{\tau}_3)$ for the F18 configuration with 10 control surfaces. Notice that three planar intersections of the three-dimensional domain are shown to simplify Fig. 1. Each region in the partition corresponds to a different optimal combination of active constraints (different actuators being saturated in each region). The boundary of the darker set of regions in $(\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \boldsymbol{\tau}_3)$ space is the feasible set $\mathcal{T}$. Although the direct allocation method[4] scales down the unique optimal control allocation at this boundary along a line to the origin, the present approach also computes the optimal active constraint combination (in terms of a quadratic cost) when the commanded generalized force is not at the boundary of the feasible set.

V. Marine Vessel Control Allocation: Experimental Results

A 1:70 scale model of a dynamically positioned supply ship is considered (Fig. 2). Its motion is controlled by using four

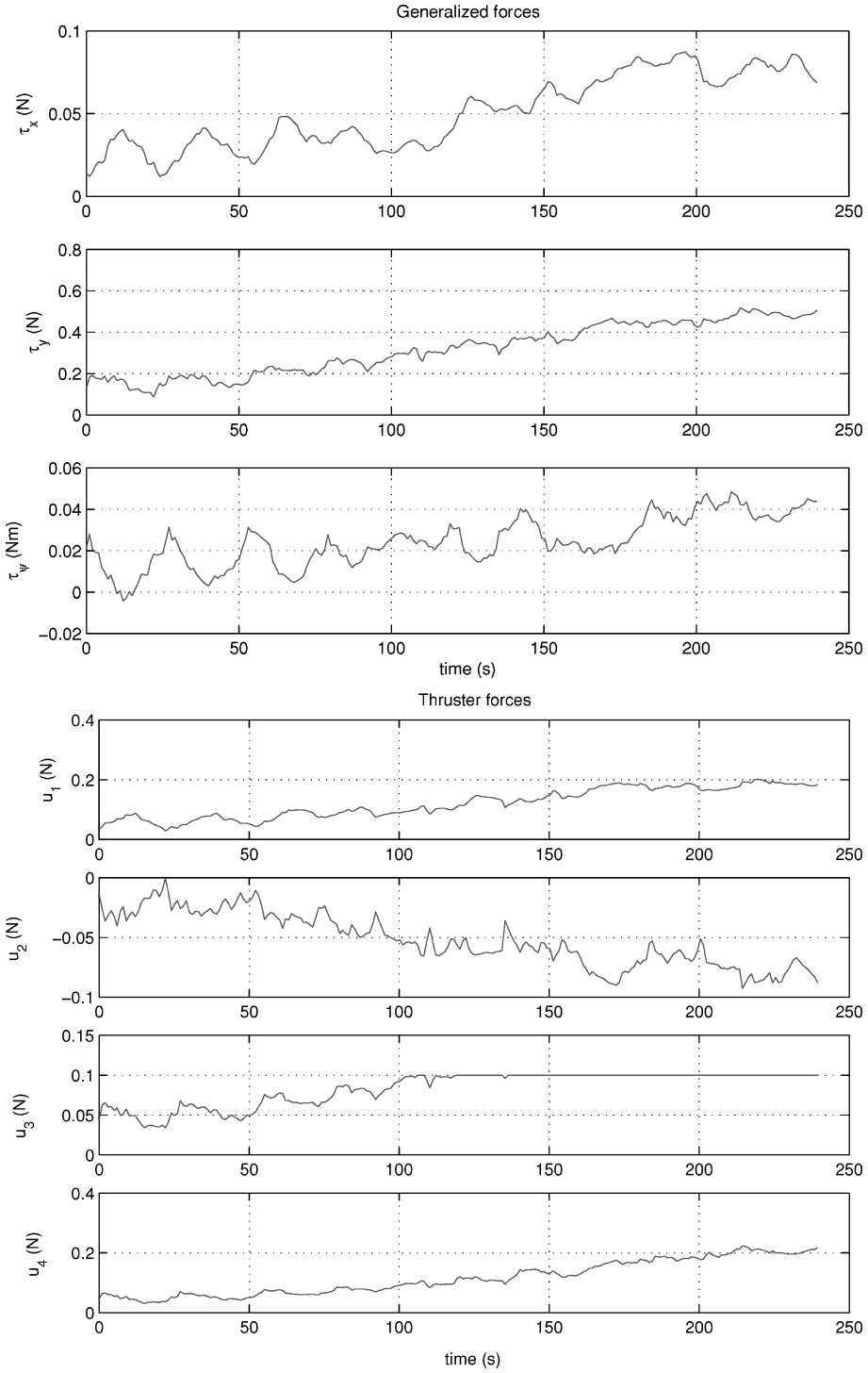

Fig. 3 Experimental results, scenario 1: desired generalized forces and thruster forces computed using the optimal constrained thrust allocation; when u_3 saturates, a compensational force is set up by u_1, u_2, and u_4.

independent azimuth thrusters. In this paper, the thruster orientations are kept fixed such that the control allocation problem is defined in terms of the forces produced by each thruster, $\boldsymbol{u} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_4)$, and the commanded generalized forces in surge, sway, and yaw $\boldsymbol{\tau} = (\boldsymbol{\tau}_x, \boldsymbol{\tau}_y, \boldsymbol{\tau}_\psi)$. (We remark that rotatable azimuth thruster problems can be formulated by letting the $\boldsymbol{u}$ vector contain the individual thruster force vector $\boldsymbol{xy}$ components, as shown in Ref. 24.) Hence, $\boldsymbol{\tau} = B\boldsymbol{u}$, with B defined as (also see Ref. 13)

$$B = \begin{pmatrix} 0.7314 & 0.7314 & 0 & 0 \\ -0.6820 & 0.6820 & 1 & 1 \\ 0.3314 & -0.3314 & 0.3400 & 0.4600 \end{pmatrix} \tag{16}$$

The maximum and minimum forces that can be generated by each thruster are limited:

$$\boldsymbol{u}_{\max} = -\boldsymbol{u}_{\min} = \begin{pmatrix} 0.8 \\ 0.8 \\ 0.1 \\ 0.8 \end{pmatrix} \tag{17}$$

We pose the problem in the form QP (7–9) with the following parameters: $R = I$ and $Q = 10^3 I$. The optimal MPQP solution is a PWL function represented by 59 polyhedral regions and affine parts. Evaluating the PWL function requires at most 83 arithmetic operations, and storing the search tree and PWL function representation requires

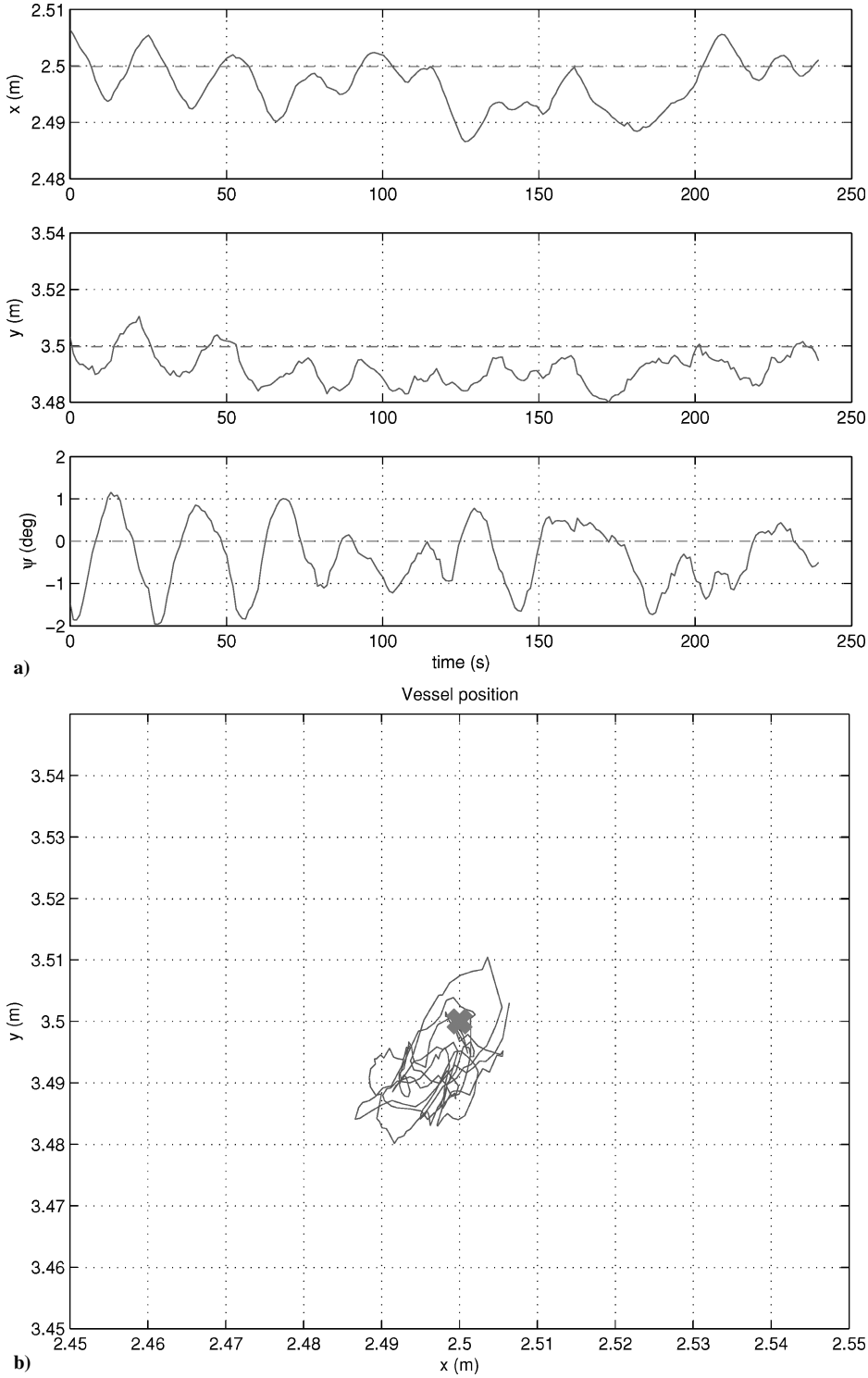

Fig. 4 Experimental results, a) scenario 1: ——, position and heading and – – –, reference; and b) dynamic positioning station-keeping performance.

1675 numbers. The maximal value of an element in $\boldsymbol{s}$ for $\boldsymbol{\tau} \in \mathcal{T}$ was in this application computed to be 0.0011. The off-line computations to obtain the PWL solution with its search tree representation was done in 1.2 s, using a Pentium M 1.6-GHz CPU.

In the experiments, the generalized forces are commanded by a generalized dynamic positioning controller that was tuned for this model ship.[30,31]

Figures 3 and 4 show experimental results from scenario 1, where the position and heading setpoints are kept constant. An external environmental force is generated using a wind generator. The force attacks the vessel along the y axis and is slowly increased throughout the experiment (Fig. 5). We observe that to compensate for this disturbance the dynamic positioning controller gradually increases the commanded generalized forces and that after $t \geq 120$ s the third thruster is saturated. The constrained control allocation handles this without any inaccuracy in the control allocation, and the tracking performance is consistently excellent throughout the experiment, with positioning errors typically less than 0.01 m and heading error less than 2 deg.

Figures 6 and 7 show experimental results from scenario 2, where the heading and position setpoints along the x axis are kept constant at $\psi = 0$, whereas the setpoint for the position along the y axis is changed from 3.5 to 4.5 m during the experiment (Fig. 5). A constant wind disturbance is present throughout the experiment. The

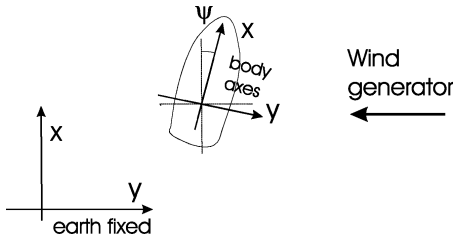

Fig. 5 Definition of coordinate systems.

third thruster is saturated at 0.1 N during a significant part of the experiment, without leading to loss of performance because the optimal constrained allocation ensures that the commanded generalized force vector $\boldsymbol{\tau}$ is still achieved with high accuracy.

We also computed an explicit solution to a real-time reconfigurable control allocation for this model ship. Here we assumed that the maximum available forces from the thrusters are symmetric, that is, $\boldsymbol{u}_{\min} = -\boldsymbol{u}_{\max}$. Then with $\boldsymbol{x} = (\boldsymbol{\tau}^T, \boldsymbol{u}_{\max}^T)^T$ used as the parameter vector, the solution was computed for all $\boldsymbol{u}_{\max}$ in the box $0_{4\times 1} \leq \boldsymbol{u}_{\max} \leq 0.8 \times 1_{4\times 1}$. The explicit PWL solution in this case

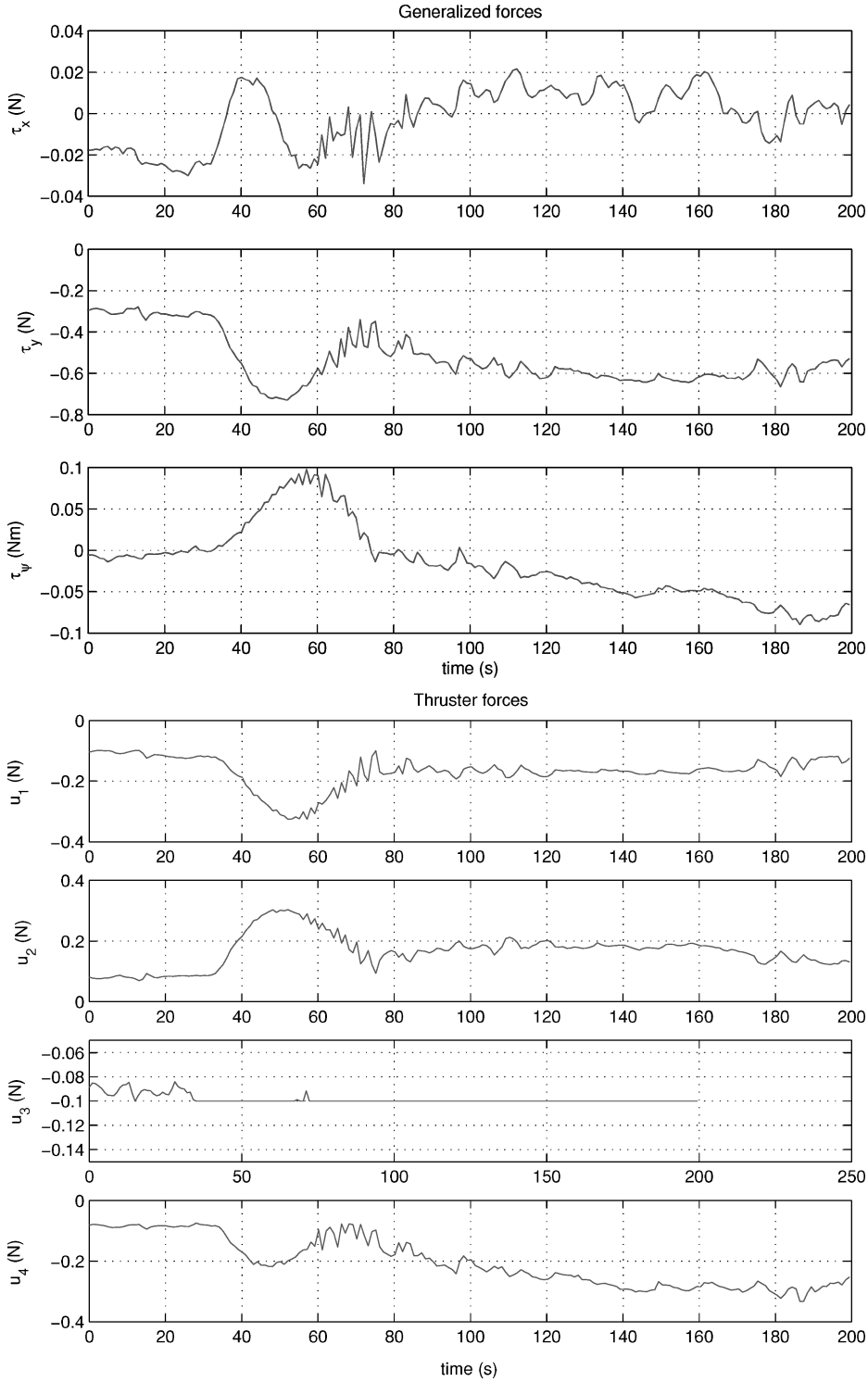

Fig. 6 Experimental results, scenario 2: desired generalized forces and thruster forces, computed using the optimal constrained thrust allocation; when u_3 saturates, u_1, u_2, and u_4 compensate.

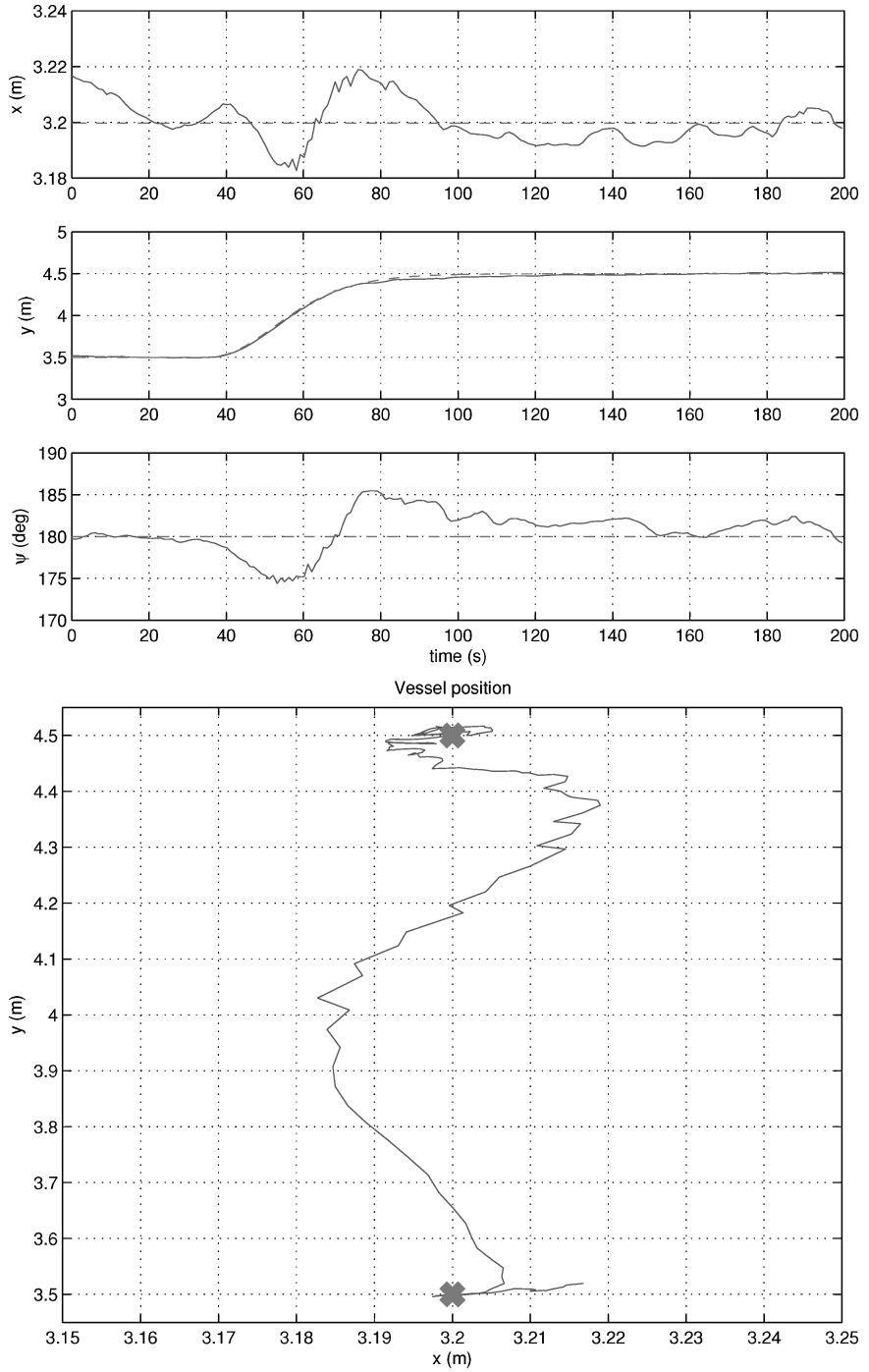

Fig. 7 Experimental results, a) scenario 2: ——, position and heading and – – –, reference; and b) dynamic positioning station-keeping performance.

consisted of 79 polyhedral regions in the seven-dimensional parameter space. Even if the number of polyhedral regions in the solution is not much larger than in the nonreconfigurable case, the corresponding search tree is larger, due to the larger dimension of the parameter space. Still, the PWL solution can be evaluated with at most 296 arithmetic operations and be stored with 13,395 numbers in real-time memory. The off-line computations needed to compute this PWL function and its binary search tree representation took 28 s.

VI. Conclusions

We have suggested a quadratic cost optimal approach to constrained control allocation. Comparison with other approaches has demonstrated that the suggested approach gives significant real-time computational advantages, having a very simple online implementation, which can formally be verified. The method takes the use of control effort explicitly into account and is a true generalization of the generalized inverse approach. Moreover, prioritizing among generalized force directions and rate constraints on the actuators can be handled. Advantages of the method includes a unique solution, which is a continuous function of the desired generalized force. We have demonstrated the use of the algorithm on ship control examples where we have shown that saturation of one out of four thrusters does not lead to loss of control performance with the constrained control allocation.

The general formulation allows several extensions compared to the mentioned explicit methods because constraint limits and certain criterion parameters may be taken as parameters to the problem so that the control allocation may be reconfigured in real time simply by changing the point at which the PWL control allocation function is evaluated. Dynamic optimal constrained control allocation in general can be solved using the same technique. In fact, the explicit

model predictive control approach based on MPQP solutions is ideally suited for this purpose. The results can be extended to more complicated situations. In this case, the problem formulation typically leads to an MPNLP that is harder to solve, but admits approximate solutions with the same implementation advantages as MPQPs.

Appendix A: MPQP

The MPQP problem in Sec. III includes both equality and inequality constraints. When the MPQP problem is solved, it is advantageous to eliminate the equality constraints from the problem formulation. This can be done by using standard methods; for example, see Ref. 32, Chapter 15.2.

As shown in Ref. 15, the MPQP problem

$$V_z(x) = \min_z \tfrac{1}{2} z^T H z \tag{A1}$$

$$\text{subject to} \qquad Gz \le W + S\boldsymbol{x} \tag{A2}$$

can be solved by applying the Karush–Kuhn–Tucker (KKT) conditions

$$Hz + G^T\lambda = 0, \qquad \lambda \in \mathbb{R}^q \tag{A3}$$

$$\lambda_i(G^i z - W^i - S^i\boldsymbol{x}) = 0, \qquad i = 1, \ldots, q \tag{A4}$$

$$\lambda \ge 0 \tag{A5}$$

$$Gz - W - S\boldsymbol{x} \le 0 \tag{A6}$$

Superscript i on some matrices denotes the ith row. When it is assumed that H has full rank, Eq. (A3) gives

$$z = -H^{-1}G^T\lambda \tag{A7}$$

Assume for the moment that we know which constraints are active at the optimum for a given $\boldsymbol{x}$, and let $\tilde{\lambda}$ be the Lagrange multipliers of the active constraints, $\tilde{\lambda} \ge 0$. We can now form matrices $\tilde{G}$, $\tilde{W}$, and $\tilde{S}$, which contain the rows G^i, W^i, and S^i corresponding to the active constraints. Assume that $\tilde{G}$ has full row rank, such that the rows of $\tilde{G}$ are linearly independent. For the active constraints, Eqs. (A4) and (A7) give $-\tilde{G}H^{-1}\tilde{G}^T\tilde{\lambda} - \tilde{W} - \tilde{S}\boldsymbol{x} = 0$, which leads to

$$\tilde{\lambda} = -(\tilde{G}H^{-1}\tilde{G}^T)^{-1}(\tilde{W} + \tilde{S}\boldsymbol{x}) \tag{A8}$$

Equation (A8) can now be substituted into Eq. (A4) to obtain

$$z = H^{-1}\tilde{G}^T(\tilde{G}H^{-1}\tilde{G}^T)^{-1}(\tilde{W} + \tilde{S}\boldsymbol{x}) \tag{A9}$$

We have now characterized the solution to problem (A1) and (A2) for a given optimal active set and a fixed $\boldsymbol{x}$. However, as long as the active set remains optimal in a neighborhood of $\boldsymbol{x}$, the solution (A9) remains optimal, when z is viewed as a function of $\boldsymbol{x}$. Next, we characterize the region where this active set remains optimal. First, $\boldsymbol{z}$ must remain feasible [inequality (A6)]:

$$GH^{-1}\tilde{G}^T(\tilde{G}H^{-1}\tilde{G}^T)^{-1}(\tilde{W} + \tilde{S}\boldsymbol{x}) \le W + S\boldsymbol{x} \tag{A10}$$

Second, the Lagrange multipliers λ must remain nonnegative [inequality (A5)]:

$$-(\tilde{G}H^{-1}\tilde{G}^T)^{-1}(\tilde{W} + \tilde{S}\boldsymbol{x}) \ge 0 \tag{A11}$$

Inequalities (A10) and (A11) describe a polyhedron in the parameter space. This region is denoted as the critical region CR_0 corresponding to the given set of active constraints. In Ref. 15, it is shown that when you pick an arbitrary $\boldsymbol{x}_0 \in X$ and let (z_0, λ_0) be the corresponding values satisfying the KKT conditions, then one can find the critical region CR_0 from inequalities (A10) and (A11). This region is a convex polyhedral set and represents the largest set of parameters $\boldsymbol{x}$ such that the combination of active constraints at the minimizer remains optimal.

An algorithm has been developed in Ref. 14 for constructing polyhedral partitions of the parameter space that explicitly defines the PWL function $z^*(\boldsymbol{x})$. Next we give a simplified description of the algorithm; a more comprehensive description and analysis that also covers degeneracy and infeasibility is found in Ref. 14.

Algorithm (off-line MPQP solver):

1) Initialize the list of unexplored active sets $\mathcal{U}$ with an arbitrary (but feasible) active set. Initialize the list of explored active sets $\mathcal{E}$ to be empty.

2) Choose an arbitrary active set in $\mathcal{U}$, compute the associated linear state feedback (A9), Lagrange multiplier (A8), and polyhedral region CR_0 defined by inequalities (A10) and (A11). Remove the active set under consideration from $\mathcal{U}$ and add it to $\mathcal{E}$.

3) If $CR_0 = \emptyset$, go to step 2, otherwise go to step 4.

4) For each facet of the corresponding polyhedral representation determine the active set in the neighboring region as described in detail in Ref. 14. For each new active set, that is, not already in $\mathcal{E} \cup \mathcal{U}$, add it to $\mathcal{U}$.

5) If $\mathcal{U}$ is nonempty, go to step 2; otherwise terminate.

Appendix B: Real-Time Computations Using Binary Search Tree

PWL functions defined on polyhedral partitions can be evaluated efficiently by searching a binary search tree constructed using the recently developed method in Ref. 16. Assume the PWL function is $z = z^*(\boldsymbol{x})$. The key problem is then to determine in which polyhedral region any given $\boldsymbol{x}$ belongs. To solve this problem efficiently, we design a binary search tree. At each level in the search tree, the linear equation corresponding to a hyperplane is evaluated. Based on the sign, one knows on which side of the hyperplane the given $\boldsymbol{x}$ is. An example of such a search tree is given in Fig. B1. The partition given at the top of Fig. B1 is a PWL function represented by six polyhedral regions X_i in a two-dimensional space. There are, however, only three different linear functions F_j in the different regions. The objective is to find which F_j to evaluate. The tree has to be traversed from the root node N_1 to one leaf node to find such a F_j. The binary tree structure (bottom of Fig. B1) is a representation of this partition. In each nonleaf node of the tree (nodes N_1, N_2, and N_3) one hyperplane is evaluated, and based on the sign of this evaluation, a subnode is selected. In this case, only two hyperplanes have to be evaluated to reach a leaf node and its corresponding linear function F_j.

If the hyperplane is such that it roughly divides the set of polyhedral regions into two parts with the same number of regions, one has effectively reduced the number of candidate polyhedral regions by half. Only in special cases can one expect to be able to find a hyperplane that splits a given set of polyhedral regions into half, but the method in Ref. 16 typically splits the polyhedral regions into three parts: one on each side of the hyperplane and one-third containing

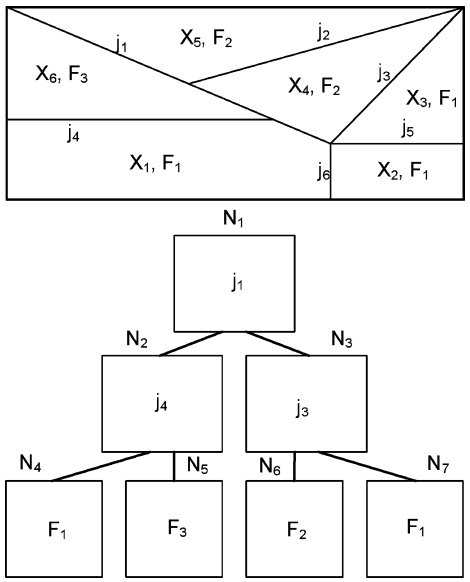

Fig. B1 Search tree generated from partition.

polyhedra that are intersected by the hyperplane. Hence, one can exclude some fraction of the polyhedral regions by evaluating a single hyperplane at each level in the search tree. The computational complexity of binary tree search is still logarithmic in the number of polyhedral regions[33] and, therefore, leads to extremely efficient evaluation of such PWL functions.

Acknowledgment

This work was partly financed by the Norwegian Research Council, Projects 134553/410 and 157804/432.

References

[1]Sørdalen, O. J., "Optimal Thrust Allocation for Marine Vessels," *Control Engineering Practice*, Vol. 5, 1997, pp. 1223–1231.

[2]Fossen, T. I., *Marine Control Systems: Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles*, Marine Cybernetics, Trondhiem, Norway, 2002.

[3]Durham, W. C., "Constrained Control Allocation," *Journal of Guidance, Control, and Dynamics*, Vol. 16, 1993, pp. 717–725.

[4]Durham, W. C., "Constrained Control Allocation: Three-Moment Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 17, 1994, pp. 330–336.

[5]Jackson, W., Fromherz, M. P. J., Biegelsen, D. K., Reisch, J., and Goldberg, D., "Constrained Optimization Based Control of Real Time Large-Scale Systems: Airjet Object Movement System," *Proceedings of the 40th IEEE Conference on Decision and Control*, Inst. of Electrical and Electronics Engineers, New York, 2001.

[6]Enns, D., "Control Allocation Approaches," *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, AIAA, Reston, VA, 1998, pp. 98–108.

[7]Bodson, M., "Evaluation of Optimization Methods for Control Allocation," *Journal of Guidance, Control, and Dynamics*, 2002, Vol. 25, 2002, pp. 703–711.

[8]Lindfors, I., "Thrust Allocation Methods for the Dynamic Positioning System," *Proceedings of the 10th International Ship Control Symposium*, 1993, pp. 3.93–3.106.

[9]Webster, W. C., and Sousa, J., "Optimum Allocation for Multiple Thrusters," *Proceedings of the International Society of Offshore and Polar Engineers Conference*, 1999.

[10]Härkegård, O., "Efficient Active Set Algorithms for Solving Constrained Least Squares Problems in Aircraft Control Allocation, *Proceedings of the 41st IEEE Conference on Decision and Control* , Inst. of Electrical and Electronics Engineers, New York, 2002.

[11]Johansen, T. A., Fossen, T. I., and Berge, S. P., "Constrained Nonlinear Control Allocation Using Quadratic Programming," *IEEE Transactions on Control Systems Technology*, Vol. 12, 2004, pp. 211–216.

[12]Lindegaard, K.-P., and Fossen, T. I., "Fuel Efficient Control Allocation for Surface Vessels with Active Rudder Usage: Experiments with a Model Ship," *IEEE Transactions on Control Systems Technology*, Vol. 11, 2003, pp. 850–862.

[13]Berge, S. P., and Fossen, T. I., "Robust Control Allocation of Overactuated Ships; Experiments with a Model Ship," IFAC Conf. on Maneuvering and Control of Marine Craft, 1997.

[14]Tøndel, P., Johansen, T. A., and Bemporad, A., "An Algorithm for Multi-Parametric Quadratic Programming and Explicit MPC Solutions," *Automatica*, Vol. 39, No. 3, 2003, pp. 489–497.

[15]Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N., "The Explicit Linear Quadratic Regulator for Constrained Systems," *Automatica*, Vol. 38, No. 1, 2002, pp. 3–20.

[16]Tøndel, P., Johansen, T. A., and Bemporad, A., "Computation of Piecewise Affine Control via Binary Search Tree," *Automatica*, Vol. 39, No. 5, 2003, pp. 945–950.

[17]Durham, W. C., "Attainable Moments for the Constrained Control Allocation Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 17, 1994, pp. 1371–1373.

[18]Bordignon, K. A., and Durham, W. C., "Closed-Form Solutions to Constrained Control Allocation Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 18, 1995, pp. 1000–1007.

[19]Durham, W. C., "Efficient, Near-Optimal Control Allocation," *Journal of Guidance, Control, and Dynamics*, Vol. 22, 1999, pp. 369–372.

[20]Petersen, J. A. M., and Bodson, M., "Fast Implementation of Direct Allocation with Extension to Coplanar Controls," *Journal of Guidance, Control, and Dynamics*, Vol. 25, 2002, pp. 464–473.

[21]Bordignon, K. A., and Durham, W. C., "Null-Space Augmented Solutions to Constrained Control Allocation Problems," *AIAA Guidance, Navigation, and Control Conference*, AIAA, Reston, VA, 1995, pp. 328–333.

[22]Gal, T., *Postoptimal Analyses, Parametric Programming, and Related Topics*, 2nd ed., de Gruyter, Berlin, 1995.

[23]Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N., "The Explicit Linear Quadratic Regulator for Constrained Systems," *Proceedings of the American Control Conference*, 2000, pp. 872–876.

[24]Johansen, T. A., Fuglseth, T. P., Tøndel, P., and Fossen, T. I.,"Optimal Constrained Control Allocation in Marine Surface Vessels with Rudders," *IFAC 6th Conference on Manoeuvring and Control of Marine Crafts*, IFAC, 2003, pp. 215–220.

[25]Fletcher, R., *Practical Methods of Optimization*, 2nd ed., Wiley-Interscience, New York, 1987.

[26]Kerrigan, E. C., and Maciejowski, J. M., "Soft Constraints and Exact Penalty Functions in Model Predictive Control," *Proceedings of the UKACC International Conference (Control 2000)*, 2000.

[27]Tøndel, P., Johansen, T. A., and Bemporad, A., "Further Results on Multi-Parametric Quadratic Programming," *Proceedings of the 42nd IEEE Conference on Decision and Control*, IEEE Publications, Piscataway, NJ, 2003.

[28]Johansen, T. A., "On Multi-Parametric Nonlinear Programming and Explicit Nonlinear Model Predictive Control," *Proceedings of the 41st IEEE Conference on Decision and Control*, Vol. 3, IEEE Publications, Piscataway, NJ, 2003, pp. 2768–2773.

[29]Durham, W. C., "Computationally Efficient Control Allocation," AIAA Paper 99-4281, 1999.

[30]Fossen, T. I., and Strand, J. P., "Passive Nonlinear Observer Design for Ships Using Lyapunov Methods: Experimental Results with a Supply Vessel," *Automatica*, Vol. 35, 1999, pp. 3–16.

[31]Strand, J. P., "Nonlinear Position Control Systems Design for Marine Vessels," Ph.D. Dissertation, Dept. of Engineering Cybernetics, Norwegian Univ. of Science and Technology, Trondheim, Norway, 1999.

[32]Nocedal, J., and Wright, S. J., *Numerical Optimization*, Springer, Berlin, 1999, Chap. 15.2.

[33]Aho, A. V., Hopcroft, J. E., and Ullman, J. D., *Data Structures and Algorithms*, Addison–Wesley, Reading, MA, 1983.