Thruster fault diagnosis and accommodation for open-frame underwater vehicles

Article *in* Control Engineering Practice · December 2004

DOI: 10.1016/j.conengprac.2003.12.014

2 authors, including:

Edin Omerdic
University of Limerick
101 PUBLICATIONS **1,019** CITATIONS

SEE PROFILE

3B2v7.51c GML4.3.1 CONPRA : 1761 Prod.Type:FTP pp.1–25(col.fig.:fig1–4,6–16,18,19) ED:Suraina PAGN: CSR SCAN: roopa

ARTICLE IN PRESS

Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Control Engineering Practice ▮ (▮▮▮▮) ▮▮▮–▮▮▮

CONTROL ENGINEERING PRACTICE

www.elsevier.com/locate/conengprac

Thruster fault diagnosis and accommodation for open-frame underwater vehicles

Edin Omerdic*, Geoff Roberts

Mechatronics Research Centre, University of Wales College, Newport Allt-yr-yn Campus, P.O. Box 180, Newport NP20 5XR, UK

Received 1 December 2003; accepted 20 December 2003

Abstract

This paper introduces a novel thruster fault diagnosis and accommodation system (FDAS) for open-frame underwater vehicles. Basically, the FDAS is a control allocator, but this primary function is enhanced with the ability of automatic thruster fault detection and accommodation. The proposed FDAS consists of two subsystems: a fault diagnosis subsystem (FDS) and a fault accommodation subsystem (FAS). The FDS uses fault detector units (FDUs), associated with each thruster, to monitor their state. Robust and reliable FDUs are based on integration of self-organising maps and fuzzy logic clustering methods. These units are able to detect internal and external faulty states of thrusters. The FAS uses information provided by the FDS to accommodate faults and perform an appropriate control reallocation. A control energy cost function is used as the optimisation criteria. The FAS uses weighted pseudo-inverse to find the solution of the control allocation problem, which minimise this criteria. Two approximations (truncation or scaling) can be used to ensure feasibility of the solution. The proposed FDS is evaluated with data obtained during test trials. The feasible region concept, related with the problem of thruster velocity saturation, is developed in order to provide geometrical interpretation of the control allocation problem. The proposed FDAS is implemented as a Simulink model (ROV simulator), in order to evaluate its performance in different faulty situations.

Keywords: Fault detection; Fault-tolerant systems; Feasible region; Optimal design; Simulators; Underwater vehicles

1. Introduction

A large number of open-frame underwater vehicles have no other actuators except thrusters. This paper introduces new approach associated with thruster fault diagnosis and accommodation for this class of underwater vehicles. The work presented herein is applicable to wide class of open-frame underwater vehicles. However, the application is focused on two remotely operated vehicles (ROVs) with different thruster configuration. The paper expands upon the work previously reported by the authors (Omerdic & Roberts 2003; Omerdic, Roberts, & Ridao, 2003).

Underwater vehicles are liable to faults or failures during underwater missions. Thrusters are one of the most common and most important sources of faults. In all but the most trivial cases the existence of a fault may lead to cancelling the mission. The implication of small faults could be very expensive and time consuming. Although good design practice tries to minimize the occurrence of faults and failures, there is a certain probability that faults will occur. Recognition that such events do occur enables system designers to develop strategies by which the effect they exert is minimised. A large number of underwater vehicles have more horizontal thrusters for the motion in the horizontal plane than controllable DOF. This paper demonstrates that, for this class of vehicle, in the case of a partial or total fault in a horizontal thruster, it is possible to reconfigure the control system in an optimal manner, in order to maintain a high level of manoeuvrability of the faulty vehicle and complete the mission.

In the literature numerous applications of fault diagnosis are reported for aeronautical and aerospace systems, automotive and traffic systems, chemical processes, electrical and electronic systems, nuclear plants, power systems and transportation systems (for

*Corresponding author. Tel.: +44-78-132-40-419; fax: +44-1633-432-442.

E-mail addresses: edin.omerdic@newport.ac.uk (E. Omerdic), geoff.roberts@newport.ac.uk (G. Roberts).

0967-0661/$ - see front matter © 2003 Published by Elsevier Ltd.
doi:10.1016/j.conengprac.2003.12.014

example, see Isermann & Ballé, 1997). Only recently attention has been devoted to fault diagnosis for unmanned underwater vehicles. The integration of fault-tolerant capabilities within the frameworks of the various control architectures for unmanned underwater vehicles is still an open problem. A comprehensive discussion of this topic for general fault-tolerant systems is reported in Blanke, Staroswiecki, and Eva Wu (2001), Blanke (2001).

Takai and Ura (1999) proposed a model-based approach for self-diagnosis of an autonomous underwater vehicle (AUV). A key element of the self-diagnosis scheme was a recurrent neural network representation of the dynamics of the AUV. The scheme was implemented on a test bed AUV, and results showed its ability to cope with sensor and actuator failures.

A fault-diagnostic system for unmanned underwater vehicles was designed and tested in real operating conditions by Alessandri, Caccia, and Veruggio (1999). They considered total and partial actuator faults. An approximate model of the vehicle was used. Fault detection and diagnosis was accomplished by evaluating any change in the normal behaviour of the system by comparing the state, the parameters and other related quantities of the observed process with those of the normal and faulty processes. On the basis of the healthy and faulty models, a bank of estimators was used for the nominal model, the left and right actuator faults. Extended Kalman filters were implemented in the process of residual generation for each actuator fault type, including the no-fault case. This scheme showed effective isolation, at the cost of greater computational efforts.

A fault-tolerant system for use in an experimental AUV was outlined in Yang, Yuh, and Choi (1999, 1998). The system was subdivided into individual fault-tolerant subsystems for dealing with thruster and sensor failures separately. The thruster subsystem consisted of a rule base for detection and isolation purposes, and an algorithm for reconfiguring the thruster control matrix by eliminating the corresponding column to accommodate the failure. Only a total fault (failure) of the thruster was considered. The authors used a constraint-based method instead of the pseudo-inverse method to compute the inverse of the thruster configuration matrix. An experimental investigation was conducted on a 6 DOF AUV, ODIN at the University of Hawaii to evaluate the performance of the proposed approaches and experimental results showed that the overall system was capable of performing effectively.

A fault detection, isolation and accommodation system, based on operationally experienced faults in ROV actuators, is proposed in Bono, Bruzzone, Bruzzone, and Caccia (1999). The authors designed a fault management system for underwater vehicles, able to satisfy the basic requirement of handling experienced faults (e.g. flooded thruster) and conventional zero output failures treated in the literature. In addition, the fault management system had to be easily integrated in the hierarchical control architectures. The authors published experience from the sea trials, when the water penetrated the thruster and modified the internal electrical connections in such a way that the actual angular speed was higher than the desired one, and current consumption was higher than normal. Fault detection was performed by monitoring the servo-amplifiers residuals, while fault isolation required the vehicle to execute steady-state manoeuvres. Actuator fault accommodation was performed by inhibiting the faulty thruster and by reconfiguring the distribution of the control actions cancelling the corresponding column in the thruster control matrix.

The problem of optimal distribution of propulsion forces for over actuated underwater vehicles is addressed in Podder, Antonelli, and Sarkar (2000). The authors investigate how to exploit the excess number of thrusters to accommodate thruster faults. First, a redundancy resolution scheme is presented, which takes into account the presence of excess number of thrusters along with any thruster faults and determines the reference thruster forces to produce the desired motion. In the next step, these reference thruster forces are utilized in the thruster controller to generate the required motion. This approach resolves the thruster redundancy in the Cartesian space and allows the AUV to track the task-space trajectories with asymptotic reduction of the task-space errors. The results from both computer simulations and experiments were provided to demonstrate the viability of the proposed scheme. The paper is a development of the preliminary concept proposed in Podder and Sarkar (1999).

If the number of control inputs is equal to or more than the number of controllable DOF, it is possible to find an "optimal" distribution of the control energy, which minimises the quadratic energy cost function i.e. a measure of the control effort (Fossen, 1995). This approach uses the generalised inverse matrix to find the optimal control vector. However, the problem of thruster velocity saturation was not considered.

The method for process condition monitoring, based on the integration of a fuzzy inference system and a self-organising map (SOM), is proposed in Cuadrado, Díaz, Díez, Obeso, and González (2001). The method identifies regions in the SOM visualisation space, corresponding to different conditions of a monitored process, by means of a fuzzy rule system, which incorporate expert knowledge about process in the region identification procedure.

Significant efforts have been undertaken in research community over last two decades to solve the control allocation problem for modern aircraft. Different methods were proposed such as direct control allocation

(Durham, 1994, 1993), optimisation-based methods using l_2 norm (Enns, 1998; Virnig & Bodden, 1994; Snell, Enns, & Garrard, 1992) and l_1 norm (Ikeda & Hood, 2000; Enns, 1998; Lindfors, 1993), fixed-point method (Burken, Lu, & Wu, 1999; Burken, Lu, Wu, & Bahm, 2001) and daisy chain control allocation (Bordignon, 1996). However, the problem of fault accommodation for underwater vehicles is closely related with the control allocation problem for aircrafts. In both cases, the control allocation problem can be defined as the determination of the actuator control values that generate a given set of desired or commanded forces and moments.

For the unconstrained control allocation problem with a control energy cost function used as optimisation criteria the optimal solution is weighted pseudo-inverse (Fossen, 1995). Pseudo-inverse is a special case of general inverse (GI). GI solutions have the advantage of being relatively simple to compute and allowing some control in distribution of control energy among available actuators. However, in real applications actuator constraints must be taken into account, which leads to a constrained control allocation problem. Handling of constrained controls is the most difficult problem for GI approach. In some cases, the solution obtained by the generalised inverse approach is not feasible, i.e. it violates actuator constraints. Durham (1993) demonstrated that, except in certain degenerate cases, a general inverse cannot allocate controls inside a constrained control subset Ω that will map to all attainable command set Φ, i.e. only a subset of Φ can be covered. Two methods are suggested to handle cases where attainable control inputs cannot be allocated. The first approach (truncation) calculates a GI solution and truncates any controls (components of control vector) which exceed their limits. The second approach (scaling) maintains the direction of the desired control input command by scaling unfeasible pseudo-inverse solution to the boundary of Ω (Bordignon, 1996). Even if the controls do not saturate, care must be taken in choosing the GI. When weighted pseudo-inverse solutions are used for problems where the actuator settings are measured in different physical dimensions, the elements of the weighting matrix must be chosen carefully if the resulting solution is desired to be invariant to changes in units and coordinate systems (Doty, Melchiorri, & Bonivento, 1993). The FDAS overcomes this problem by performing normalisation, such that all physical parameters are removed from the thruster control matrix and included in limit constraints, which are used during normalisation process to scale individual components of vectors on standard interval $[-1, 1]$.

Recently, some authors reformulated the constrained control allocation problem as a quadratic programming (QP) problem. QP generally refers to the numerical solution of the optimisation problems with an l_2 norm. An explicit solution approach is developed by Tøndel, Johansen, and Bemporad (2001). An on-line algorithm is presented in Tøndel, Johansen, and Bemporad (2003), while the application to marine vessels is given in Johansen, Fossen, and Tøndel (2002). An alternative to the explicit solution is to use an iterative solution to solve the QP problem. The drawback with the iterative solution is that several iterations may have to be performed at each sample in order to find the optimal solution. An advantage of the iterative approach is that there is more flexibility for on-line reconfiguration. Computational complexity is also greatly reduced by a "warm start", i.e. the numerical solver is initialised with the solution of the optimisation problem from the previous sample (Fossen, 2002).

Ideas from Podder et al. (2000), Yang et al. (1999), Yang et al. (1998), Bordignon (1996), Fossen (1995) were used as the initial basis for the design of the FDAS, presented herein. Two common thruster configurations are described in the second section. The third section describes a standard control architecture for ROVs (without the FDAS) and an improved architecture (with the FDAS). A general thruster model is described in the fourth section. The control allocation problem is analysed in the fifth section, which begins with the general problem formulation, followed by the decomposition of motion. A normalisation procedure is then described, followed by the problem formulation for horizontal thrusters and an analysis of the choice of weighting matrix. Finally, the section ends with the remarks about a vertical thruster. The sixth section describes the architecture of the proposed FDAS. This section begins with the short description of the overall architecture, followed by a detail description of the FDS and the FAS structure. The potential application of the FDAS is addressed in the seventh section. Finally, the eighth section summarizes the concluding remarks and contributions of the paper.

2. Thruster configuration

For underwater vehicles the most common actuators are (Fossen, 2002):

- *Azimuth thrusters*: Thruster units that can be rotated an angle α about the z-axis during the mission and produce two force components $(F_x.F_y)$ in the horizontal plane. They are attractive in dynamic positioning systems, since they can produce forces in different directions, leading to an overactuated control problem that can be optimised with respect to power and possible faulty situations.
- *Fixed direction* (*non-rotable*) *thrusters*: In contrast to azimuth thrusters, where an angle α can vary with time, fixed direction thrusters are characterised with a

fixed angle $\alpha = \alpha_0$, i.e. orientation of these thrusters is fixed in advance and cannot be changed during the mission.

- *Control surfaces*: Control surfaces can be mounted at different locations to produce lift and drag forces, like fins for diving, rolling and pitching, rudders for steering, etc.

Two underwater vehicles (FALCON, SeaEye Marine Ltd. and URIS, University of Girona, see Fig. 1) with different thruster configurations are used to demonstrate the performance of the FDAS. These ROVs have no other actuators except fixed direction thrusters, and the following discussion will be concentrated on this type of actuators, while more information about other types can be found in Fossen (2002).

The main assumption for successful control allocation in the case of a fault in a single thruster is that the control system for motion in the horizontal plane is overactuated. From this reason the original thruster configuration of URIS, with two horizontal and two vertical thrusters, is transformed into a configuration with four horizontal thrusters, without any vertical thruster, as shown in Fig. 1b. This modification was possible, because the tank for test trials and acquiring of training data at University of Girona was very shallow and there was no space and need for motion in vertical plane. Three-dimensional view of the FALCON and URIS are shown, respectively, in Fig. 1a and b. Plan views of the vehicles, with corresponding configuration of the horizontal thrusters, are shown in bottom part of Fig. 1. The origin of the body-fixed reference frame $\{B\}$ is chosen to coincide with the centre of gravity. The axes are chosen to coincide with the principal axes of inertia and they are defined as:

- x_B—*longitudinal* axes (directed to front side),
- y_B—*transversal* axes (directed to starboard),
- z_B—*normal* axes (directed from top to bottom).

The FALCON has four horizontal thrusters, denoted as ${}^{i}HT$, $i = \overline{1,4}$ and one vertical ${}^{1}VT$ (not shown in Fig. 1a—bottom). The URIS has only four horizontal thrusters, denoted in the same way. Thruster configuration of the FALCON enables direct control of 4 DOF: surge, sway, yaw and heave, as indicated in Fig. 2a. In a similar way, modified thruster configuration of the URIS allows direct control of only 3 DOF: surge, sway and yaw (Fig. 2b).

Vector $\boldsymbol{\tau}_d$ (Fig. 3) can be decomposed into two parts as $\boldsymbol{\tau}_d = [\boldsymbol{\tau}_d^{HT} \ \ \boldsymbol{\tau}_d^{VT}]^{\mathrm{T}}$, where $\boldsymbol{\tau}_d^{HT}$ represents desired surge force τ_X, sway force τ_Y and yaw moment τ_N for motion in the horizontal plane, and $\boldsymbol{\tau}_d^{VT}$ is equal to heave force τ_Z for motion in the vertical plane. The control allocation problem for motion of the FALCON in the vertical plane is straightforward, since the vector $\boldsymbol{\tau}_d^{VT}$ is a scalar, i.e. there is one-to-one correspondence between the controllable DOF (heave) and the vertical thruster. However, in the general case the vector $\boldsymbol{\tau}_d^{VT}$ can have three components (heave force τ_Z, roll moment τ_K and

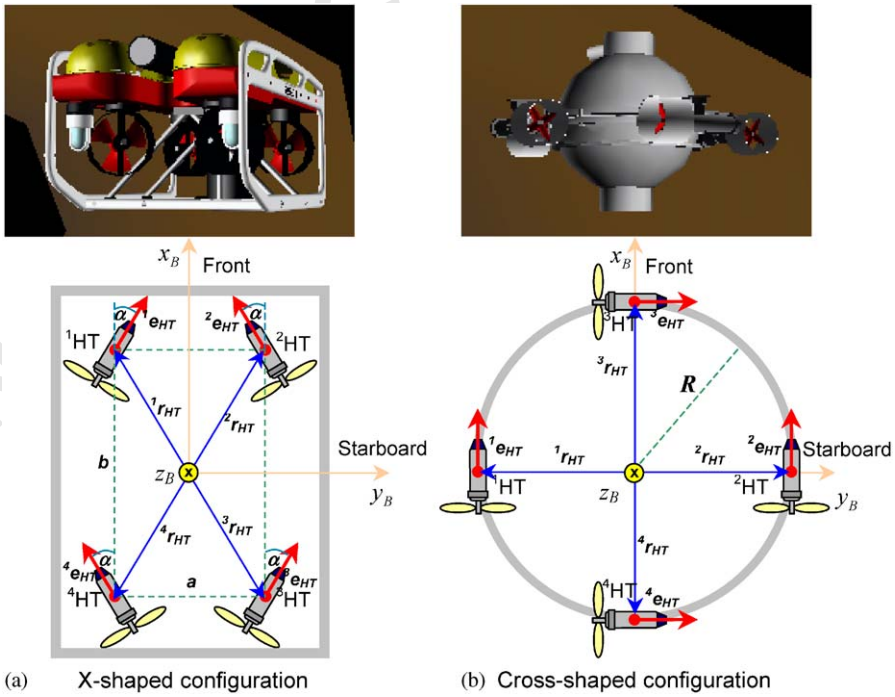

Fig. 1. Two common configurations of the horizontal thrusters: (a) FALCON. (b) URIS.

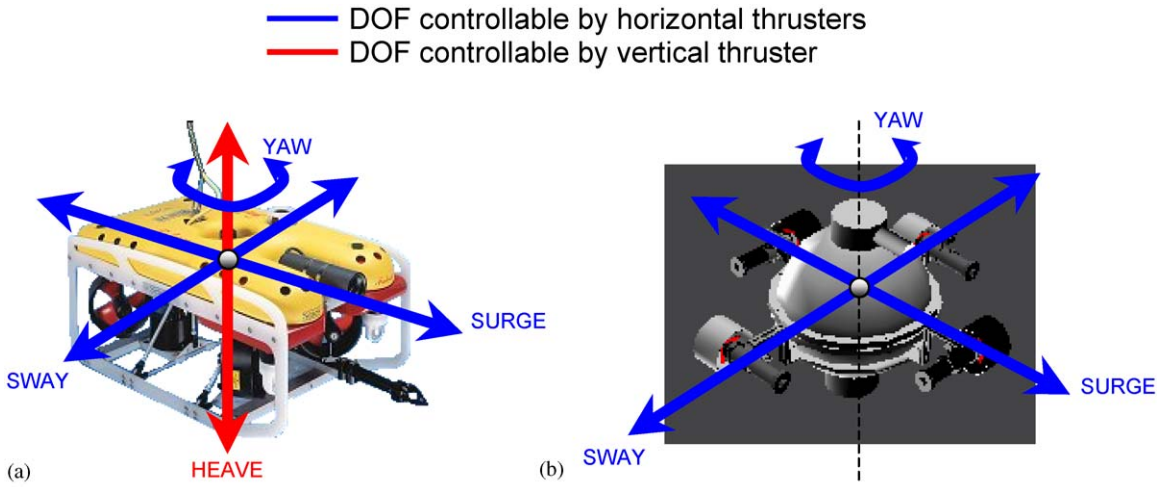

Fig. 2. Relationship between thruster configuration and controllable DOF: (a) FALCON. (b) URIS.

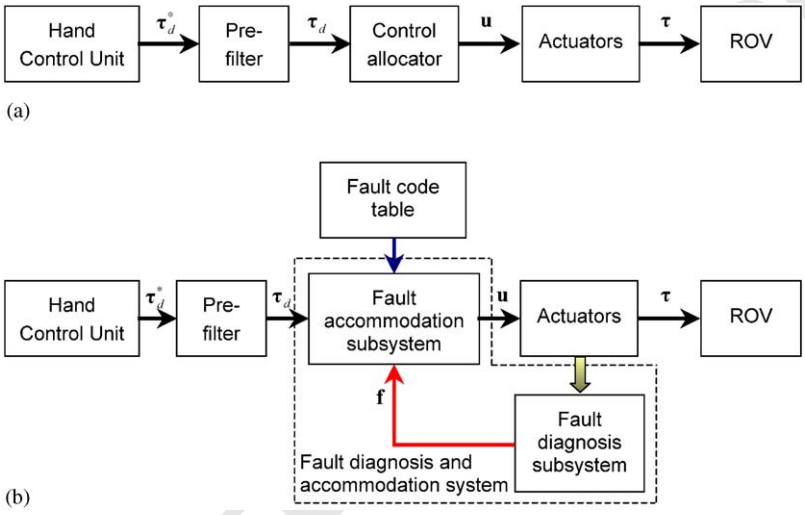

Fig. 3. Open-loop ROV control structure: (a) Standard architecture, without FDAS. (b) Improved architecture, with FDAS.

pitch moment τ_M). Typical example is ODIN (Yang et al., 1999, 1998), with four horizontal and four vertical thrusters, where each of vectors $\boldsymbol{\tau}_d^{HT}$ and $\boldsymbol{\tau}_d^{VT}$ have three components.

3. Control architecture

A standard open-loop ROV control structure is shown in Fig. 3a. The ROV pilot uses the Hand Control Unit (HCU) to generate vector $\boldsymbol{\tau}_d^*$, which can be interpreted as a desired vector of propulsion forces and moments among axes in the body-fixed frame. Raw signals from the HCU, packed in vector $\boldsymbol{\tau}_d^*$, pass through the low-pass pre-filter to smooth out the commanded input and to protect the actuators from damage caused by abrupt changes of set points. The output of the pre-filter is the desired vector of propulsion forces and moments (virtual control input) $\boldsymbol{\tau}_d$. The control allocator maps the vector $\boldsymbol{\tau}_d$ into the vector (true control input) $\mathbf{u}$, representing control settings for individual actuators. After actuation with $\mathbf{u}$, the actuators generate a vector of propulsion forces and moments (total control effect) $\boldsymbol{\tau}$, which is applied as the input to the ROV dynamics block, and determine the behaviour of the vehicle. The main objective of the control allocation is to ensure that the condition $\boldsymbol{\tau} = \boldsymbol{\tau}_d$ is satisfied for all attainable $\boldsymbol{\tau}_d$.

The control allocator in the standard structure shown in Fig. 3a is replaced by the fault accommodation subsystem (FAS) in the improved control structure, shown in Fig. 3b. The FAS performs weighted pseudo-inverse method for control allocation. The primary task of control allocation is enhanced with the fault diagnosis subsystem (FDS), able to monitor state of the thrusters and inform the FAS about any malfunctions using the total fault indicator vector $\mathbf{f}$, carrying the codes of faulty states for each thruster. The FAS uses information

Table 1
Position vectors for different thruster configurations

	Horizontal thrusters				Vertical thruster
	${}^1\boldsymbol{r}_{HT}$	${}^2\boldsymbol{r}_{HT}$	${}^3\boldsymbol{r}_{HT}$	${}^4\boldsymbol{r}_{HT}$	${}^1\boldsymbol{r}_{VT}$
X-shaped configuration (FALCON)	$\begin{bmatrix} b/2 \\ -a/2 \\ 0 \end{bmatrix}$	$\begin{bmatrix} b/2 \\ a/2 \\ 0 \end{bmatrix}$	$\begin{bmatrix} -b/2 \\ a/2 \\ 0 \end{bmatrix}$	$\begin{bmatrix} -b/2 \\ -a/2 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ l_z \end{bmatrix}$
Cross-shaped configuration (URIS)	$\begin{bmatrix} 0 \\ -R \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ R \\ 0 \end{bmatrix}$	$\begin{bmatrix} R \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} -R \\ 0 \\ 0 \end{bmatrix}$	—

Table 2
Orientation vectors for different thruster configurations

	Horizontal thrusters				Vertical thruster
	${}^1\boldsymbol{e}_{HT}$	${}^2\boldsymbol{e}_{HT}$	${}^3\boldsymbol{e}_{HT}$	${}^4\boldsymbol{e}_{HT}$	${}^1\boldsymbol{e}_{VT}$
X-shaped configuration (FALCON)	$\begin{bmatrix} \cos\alpha \\ \sin\alpha \\ 0 \end{bmatrix}$	$\begin{bmatrix} \cos\alpha \\ -\sin\alpha \\ 0 \end{bmatrix}$	$\begin{bmatrix} \cos\alpha \\ \sin\alpha \\ 0 \end{bmatrix}$	$\begin{bmatrix} \cos\alpha \\ -\sin\alpha \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$
Cross-shaped configuration (URIS)	$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$	—

provided by the FDS to accommodate faults by performing an appropriate reconfiguration, i.e. to reallocate control energy among operable thrusters. The overall fault diagnosis and accommodation process is very fast, despite the fact that in some cases it is necessary to perform iterations, due to the computational efficiency of the FDAS algorithm, where the heaviest numerical calculations are performed off-line, in advance.

4. Thruster model

In the general case an ROV has p thrusters ${}^1Th, {}^2Th, \ldots, {}^pTh$. Each thruster ${}^iTh$, $i = \overline{1,p}$ exerts thrust (force) ${}^i\mathbf{T}$ and torque (moment) ${}^i\mathbf{Q}_e$ (Fig. 4).[1] The position vector ${}^i\mathbf{r} = [{}^ir_x \ {}^ir_y \ {}^ir_z]^{\mathrm{T}}$ determines the position of the point of attack of the force ${}^i\mathbf{T}$, relative to the $\{B\}$. The force ${}^i\mathbf{T}$ also generates the moment ${}^i\mathbf{Q}_r = {}^i\mathbf{r} \times {}^i\mathbf{T}$, so that the total moment vector exerted by the thruster is given by ${}^i\mathbf{Q} = {}^i\mathbf{Q}_e + {}^i\mathbf{Q}_r$. The orientation of the thruster ${}^iTh$ relative to the $\{B\}$ is defined by the unit vector ${}^i\mathbf{e} = [{}^ie_x \ {}^ie_y \ {}^ie_z]^{\mathrm{T}}$. The vector ${}^i\mathbf{e}$ shows the positive direction of the force ${}^i\mathbf{T}$. This means that, if the propeller angular velocity is positive, it will exert the force ${}^i\mathbf{T}$ in the direction of ${}^i\mathbf{e}$. Otherwise, the force ${}^i\mathbf{T}$ is opposite to ${}^i\mathbf{e}$. Depending on propellers spin direction, vectors ${}^i\mathbf{T}$ and ${}^i\mathbf{Q}_e$ have the same direction (for clockwise rotation looking from the back of the propellers) or the opposite direction (for counterclockwise rotation). An elegant way to describe this relationship is to introduce a *spin direction coefficient* s: the value $s = +1(-1)$ means that the force vector ${}^i\mathbf{T}$ and the torque vector ${}^i\mathbf{Q}_e$ have the same (opposite) direction. Assume that propeller angular velocity is positive and the propellers spin direction is clockwise (looking from the back of the propellers, see Fig. 4a). The direction of the torque vector ${}^i\mathbf{Q}_e$ is determined using the right-hand rule. In this case, the force vector ${}^i\mathbf{T}$ and the torque vector ${}^i\mathbf{Q}_e$ have the same direction as the vector ${}^i\mathbf{e}$ and $s = +1$. If the propeller spin direction is counter clockwise (Fig. 4b), then vectors ${}^i\mathbf{T}$ and ${}^i\mathbf{e}$ have the same direction, while vectors ${}^i\mathbf{Q}_e$ and ${}^i\mathbf{e}$ have the opposite direction and $s = -1$.

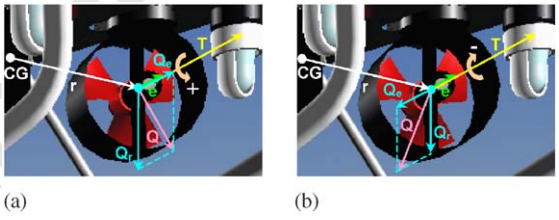

Fig. 4. Thrust and torque, exerted by a thruster, for two possible propellers spin directions: (a) Clockwise. (b) Counter clockwise.

Position vectors for different configurations are given in Table 1, while Table 2 shows orientation vectors. Parameters a, b, α and R can be obtained from the technical specifications of the vehicles.

A simplified block diagram of full thruster model, including the dynamics of the thruster control loop, is shown in Fig. 5. A DC motor drives a thruster. The input to the thruster control loop is a control variable n_d

[1] Fig. 4 displays front side of propellers.

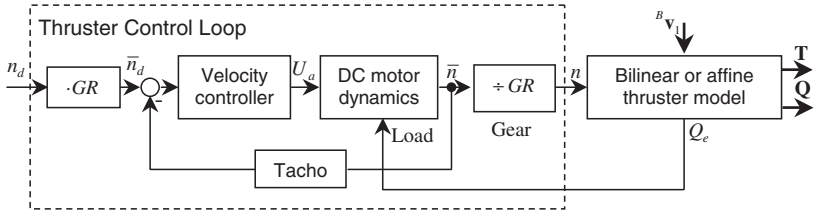

Fig. 5. Simplified diagram showing full thruster model, including thruster control loop dynamics.

(desired angular velocity). The motor is equipped with a tachometer, which measures actual angular velocity $\bar{n}$. A gearbox with the gear ratio $GR > 1$ is used to reduce the output angular velocity $\bar{n}$ ($n = (1/GR)\bar{n}$) and to enhance the output torque. This means that the input n_d must be multiplied by GR ($\bar{n}_d = GRn_d$). A typical thruster control loop is implemented as an independent device called *thruster control unit* (TCU) with integrated power amplifiers and controlled by a microcontroller. The velocity controller is usually implemented as a digital PID controller, although the other designs are possible.

In the general case, the thrust T and torque Q, exerted by a thruster, are complicated functions depending on the vehicle's velocity vector $\mathbf{v}_1 = [u\ v\ w]^{\mathrm{T}}$ and control variable n. The bilinear thruster model, described in (Fossen, 1995; Fossen & Blanke, 2000), can be used to approximate these functions. In the general case, the thrust $\mathbf{T}$ and torque $\mathbf{Q}_e$ can be calculated from

$$T(n, V_a) = \rho D^4 K_T(J_0) n|n|, \quad \mathbf{T} = T\mathbf{e},$$
$$Q_e(n, V_a) = \rho D^5 K_Q(J_0) n|n|, \quad \mathbf{Q}_e = sQ_e\mathbf{e}, \tag{1}$$

where ρ is density of water, D is propeller diameter, K_T and K_Q are non-dimensional thrust and torque coefficients, n is propeller shaft speed, s is spin direction coefficient and J_0 is advance ratio, defined as

$$J_0 = \frac{u_a}{nD}, \tag{2}$$

where u_a is ambient water velocity. Relationships between coefficients K_T, K_Q and J_0 are given by

$$K_T(J_0) = \alpha_2 - \alpha_1 J_0,$$
$$K_Q(J_0) = \beta_2 - \beta_1 J_0, \tag{3}$$

where α_i and β_i are four positive non-dimensional constants. Eqs. (3) are obtained by linear approximation of the experimental curves $K_T(J_0)$ and $K_Q(J_0)$, obtained in open water test in a cavitation tunnel or a towing tank (Fossen & Blanke, 2000). Eqs. (1) and (3) imply that the mathematical expressions for the thrust $\mathbf{T}$ and torque $\mathbf{Q}_e$ can be written as the bilinear thruster model:

$$T(n, u_a) = T_{n|n|} n|n| - T_{|n|u_a} |n| u_a, \quad \mathbf{T} = T\mathbf{e},$$
$$Q_e(n, u_a) = Q_{n|n|} n|n| - Q_{|n|u_a} |n| u_a, \quad \mathbf{Q}_e = sQ_e\mathbf{e}, \tag{4}$$

where

$$T_{n|n|} = \rho D^4 \alpha_2, \quad T_{|n|V_a} = \rho D^3 \alpha_1,$$
$$Q_{n|n|} = \rho D^5 \beta_2, \quad Q_{|n|V_a} = \rho D^4 \beta_1. \tag{5}$$

The relationship between ambient water velocity u_a around the thruster with the orientation defined with the vector $\mathbf{e}$ and the vehicle's linear velocity $\mathbf{v}_1$ is given by

$$u_a = (1 - \omega)\mathbf{v}_1^{\mathrm{T}} \cdot \mathbf{e}, \tag{6}$$

where $\omega > 0$ (typically 0.1–0.4) is the wake fraction number. It can be seen from (6) that u_a can be found as projection of the vector $\mathbf{v}_1$ on the vector $\mathbf{e}$, scaled by factor $(1 - \omega)$.

However, in practical applications, the bilinear thruster model (4) can be approximated by an *affine model* (7), where it is assumed that $u_a = 0$:

$$T(n, V_a) = T_{n|n|} n|n|, \quad \mathbf{T} = T\mathbf{e},$$
$$Q_e(n, V_a) = Q_{n|n|} n|n|, \quad \mathbf{Q}_e = sQ_e\mathbf{e}. \tag{7}$$

Initially, the control allocation problem will be formulated and solved under the following assumptions:

1. the dynamics of thruster control loop is neglected[2] (Fig. 5),
2. the relationship between propeller thrust/torque and the control variable is given by modified version of affine thruster model (7), described below.

The first assumption is realistic, since the time constants of DC motors used to drive thrusters of the FALCON and the URIS are very small and sample time, dictated by control software, is about 40 ms, long enough to ensure that all transient responses between two samples in thruster control loop disappear.

The second assumption means that:

- shaft torque Q_e is neglected, since it is small compared to Q_r, $\mathbf{Q}_r = \mathbf{r} \times \mathbf{T}$,
- the effect of the ambient water velocity u_a on propeller thrust T is neglected,

[2] One of the most important difficulties for successful control allocation is actuator dynamics. Fortunately, in many cases actuator dynamics is much faster than the dynamics of the other parts of the system. In these cases, the most common solution is that actuator dynamics are simply neglected. This will work as long as the closed-loop system is designed to be substantially slower than the actuator servo systems.

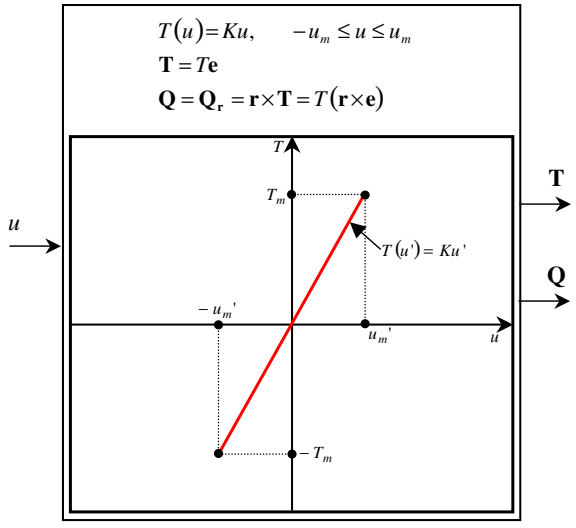

Fig. 6. Thruster model used in control allocation for underwater vehicles.

- the symmetrical relationship between thrust T and the control variable $u=n|n|$ is used (see Fig. 6).

Under these assumptions, the control allocation problem for open-frame underwater vehicles (in particular, FALCON and URIS) is linear and can be solved using techniques described in the following. The influence of the neglected factors on the performance of the ROV control system can be investigated using an ROV simulator incorporating the bilinear thruster model (4) and the dynamics of the thruster control loop.

5. Control allocation

5.1. General problem formulation

In the following it is assumed that the fixed direction (non-rotable) thrusters are only available actuators for control allocation. Under the assumptions mentioned above, a general thruster ${}^iTh$, $i=\overline{1,p}$ is modelled by a modified affine model shown in Fig. 6. Vector of forces and moments, exerted by thruster ${}^iTh$, can be written as

$$ {}^i\boldsymbol{\tau}=\begin{bmatrix}{}^i\mathbf{T}\\ {}^i\mathbf{Q}\end{bmatrix}=\begin{bmatrix}{}^iT{}^i\mathbf{e}\\ {}^iT({}^i\mathbf{r}\times{}^i\mathbf{e})\end{bmatrix}=\begin{bmatrix}{}^ie_x\\ {}^ie_y\\ {}^ie_z\\ ({}^i\mathbf{r}\times{}^i\mathbf{e})_x\\ ({}^i\mathbf{r}\times{}^i\mathbf{e})_y\\ ({}^i\mathbf{r}\times{}^i\mathbf{e})_z\end{bmatrix}{}^iT. \tag{8}$$

Superposition of the individual contributions ${}^i\boldsymbol{\tau}$, $i=\overline{1,p}$ leads to total vector of propulsion forces and moments $\boldsymbol{\tau}$:

$$ \boldsymbol{\tau}=\begin{bmatrix}\tau_X\\ \tau_Y\\ \tau_Z\\ \tau_K\\ \tau_M\\ \tau_N\end{bmatrix}=\sum_{i=1}^{p}{}^i\boldsymbol{\tau}=\sum_{i=1}^{p}\begin{bmatrix}{}^i\mathbf{e}\\ ({}^i\mathbf{r}\times{}^i\mathbf{e})\end{bmatrix}{}^iT $$

$$ =\underbrace{\begin{bmatrix}{}^1e_x & \cdots & {}^ie_x & \cdots & {}^pe_x\\ {}^1e_y & & {}^ie_y & & {}^pe_y\\ {}^1e_z & & {}^ie_z & & {}^pe_z\\ ({}^1\mathbf{r}\times{}^1\mathbf{e})_x & & ({}^i\mathbf{r}\times{}^i\mathbf{e})_x & & ({}^p\mathbf{r}\times{}^p\mathbf{e})_x\\ ({}^1\mathbf{r}\times{}^1\mathbf{e})_y & \cdots & ({}^i\mathbf{r}\times{}^i\mathbf{e})_y & \cdots & ({}^p\mathbf{r}\times{}^p\mathbf{e})_y\\ ({}^1\mathbf{r}\times{}^1\mathbf{e})_z & & ({}^i\mathbf{r}\times{}^i\mathbf{e})_z & & ({}^p\mathbf{r}\times{}^p\mathbf{e})_z\end{bmatrix}}_{\mathbf{T}}\underbrace{\begin{bmatrix}{}^1T\\ \vdots\\ {}^iT\\ \vdots\\ {}^pT\end{bmatrix}}_{\mathbf{f}}=\mathbf{Tf}, \tag{9}$$

where $\mathbf{T}\in\Re^{6\times p}$ is the *thruster configuration matrix* and $\mathbf{f}\in\Re^p$ is vector of *control forces*. For azimuth thrusters ${}^i\mathbf{e}={}^i\mathbf{e}(\alpha)$ and ${}^i\mathbf{r}={}^i\mathbf{r}(\alpha)$, which means that ${}^i\boldsymbol{\tau}={}^i\boldsymbol{\tau}(\alpha)$ and $\mathbf{T}=\mathbf{T}(\alpha)$. However, for fixed direction thrusters $\alpha=\alpha_0=\text{const.}$ and $\mathbf{T}=\mathbf{T}(\alpha_0)=\text{const.}$

Substituting ${}^iT={}^iK{}^iu$ in (9) yields

$$ \boldsymbol{\tau}=\underbrace{\begin{bmatrix}{}^1e_x & & {}^ie_x & & {}^pe_x\\ {}^1e_y & \cdots & {}^ie_y & \cdots & {}^pe_y\\ {}^1e_z & & {}^ie_z & & {}^pe_z\\ ({}^1\mathbf{r}\times{}^1\mathbf{e})_x & & ({}^i\mathbf{r}\times{}^i\mathbf{e})_x & & ({}^p\mathbf{r}\times{}^p\mathbf{e})_x\\ ({}^1\mathbf{r}\times{}^1\mathbf{e})_y & \cdots & ({}^i\mathbf{r}\times{}^i\mathbf{e})_y & \cdots & ({}^p\mathbf{r}\times{}^p\mathbf{e})_y\\ ({}^1\mathbf{r}\times{}^1\mathbf{e})_z & & ({}^i\mathbf{r}\times{}^i\mathbf{e})_z & & ({}^p\mathbf{r}\times{}^p\mathbf{e})_z\end{bmatrix}}_{\mathbf{T}}\times\underbrace{\begin{bmatrix}{}^1K & & 0 & & 0\\ & \ddots & & \cdots & \\ 0 & & {}^iK & & 0\\ & \cdots & & \ddots & \\ 0 & & 0 & & {}^pK\end{bmatrix}}_{\mathbf{K}}\underbrace{\begin{bmatrix}{}^1u\\ \vdots\\ {}^iu\\ \vdots\\ {}^pu\end{bmatrix}}_{\mathbf{u}'}=\mathbf{TKu}, \tag{10}$$

where $\mathbf{K}\in\Re^{p\times p}$ is the *force coefficient matrix* and $\mathbf{u}\in\Re^p$ is the *control vector*. Introducing substitution

$$ \mathbf{B}=\mathbf{TK}, \tag{11}$$

where $\mathbf{B}\in\Re^{6\times p}$ is the *thruster control matrix*, (10) can be rewritten as

$$ \boldsymbol{\tau}=\mathbf{Bu}. \tag{12}$$

Zero-row in $\mathbf{B}$ means that the corresponding DOF is not directly controllable with the particular thruster configuration.

Assuming that thrusters are identical, the coefficients ${}^iK$ are the same for all thrusters,

$$ {}^1K=\cdots={}^pK=K \tag{13}$$

Eq. (11) can be simplified as

$$ \mathbf{B}=\mathbf{TK}=\mathbf{T}(K\mathbf{I}_p)=K(\mathbf{TI}_p)=K\mathbf{T}. \tag{14}$$

Each component ${}^iu$ of the control vector $\mathbf{u}$ is limited by

constraint

$$-{}^{i}u_m \leqslant {}^{i}u \leqslant {}^{i}u_m, \quad i = \overline{1,p}. \tag{15}$$

Constraint (15) represents thruster velocity saturation, i.e. the physical construction of the thruster ${}^{i}Th$ imposes velocity limitations because the thruster cannot rotate faster than its maximum velocity. For the control vector $\mathbf{u}$ set of constraints (15) can be written in compact vector form as

$$-\mathbf{u}_m \leqslant \mathbf{u} \leqslant \mathbf{u}_m, \tag{16}$$

where

$$\mathbf{u}_m = [{}^{1}u_m \ \cdots \ {}^{i}u_m \ \cdots \ {}^{p}u_m]^{\mathrm{T}}. \tag{17}$$

For identical thrusters, ${}^{1}u_m = \cdots = {}^{p}u_m = u_m$ and

$$\mathbf{u}_m = u_m \underbrace{\begin{bmatrix} 1 & \cdots & 1 & \cdots & 1 \end{bmatrix}}_{p}^{\mathrm{T}}.$$

The constrained control subset Ω is defined as a set of all control vectors $\mathbf{u}$ which satisfy (16).

Finally, the general constrained control allocation problem for open-frame underwater vehicles can be formulated as

For given $\boldsymbol{\tau}$, find $\mathbf{u} \in \Omega$ such that $\mathbf{B}\mathbf{u} = \boldsymbol{\tau}$.

If condition $\mathbf{u} \in \Omega$ is removed, the problem becomes unconstrained.

Thruster configuration matrices for the FALCON and the URIS, shown in Table 3, are obtained from (14), assuming that all thrusters are identical. Parameter A is given as $A = (b/2)\sin\alpha + (a/2)\cos\alpha$.

It can be seen that the uncontrollable DOF for the FALCON are *roll* and *pitch*, since the fourth and the fifth row of $\mathbf{B}$ are zero-rows. In a similar way, uncontrollable DOF for the URIS are *heave*, *roll* and *pitch*.

Table 3
Thruster control matrix for different thruster configurations

FALCON	URIS
$\mathbf{B} \quad K \underbrace{\begin{bmatrix} \overbrace{\begin{matrix} \cos\alpha & \cos\alpha & \cos\alpha & \cos\alpha \\ \sin\alpha & -\sin\alpha & \sin\alpha & -\sin\alpha \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ A & -A & -A & A \end{matrix}}^{HT} & \overbrace{\begin{matrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{matrix}}^{VT} \end{bmatrix}}_{\mathbf{T}}$	$K \underbrace{\begin{bmatrix} \overbrace{\begin{matrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ R & -R & R & -R \end{matrix}}^{HT} \end{bmatrix}}_{\mathbf{T}}$

5.2. Decomposition of motion

In the following, the general control allocation problem will be separated into two subproblems and each will be treated individually. The first subproblem is related with the motion in the horizontal plane, and the second with the motion in the vertical plane. Decomposition of the motion is given in Table 4 for the FALCON and in Table 5 for the URIS.

Table 4
Decomposition of the FALCON motion

	FALCON
Horizontal thrusters (motion in horizontal plane)	Controllable DOF: Surge, Sway, Yaw $\boldsymbol{\tau}^{HT} = \begin{bmatrix} \tau_X \\ \tau_Y \\ \tau_N \end{bmatrix} = \underbrace{K \begin{bmatrix} \overbrace{\cos\alpha}^{HT_1} & \overbrace{\cos\alpha}^{HT_2} & \overbrace{\cos\alpha}^{HT_3} & \overbrace{\cos\alpha}^{HT_4} \\ \sin\alpha & -\sin\alpha & \sin\alpha & -\sin\alpha \\ A & -A & -A & A \end{bmatrix}}_{\mathbf{B}^{HT}} \underbrace{\begin{bmatrix} {}^{1}u^{HT} \\ {}^{2}u^{HT} \\ {}^{3}u^{HT} \\ {}^{4}u^{HT} \end{bmatrix}}_{u^{HT}}$
Vertical thruster (motion in vertical plane)	Controllable DOF: Heave $\boldsymbol{\tau}^{VT} = [\tau_Z] = \underbrace{K}_{\mathbf{B}^{VT}} = \underbrace{u^{VT}}_{\mathbf{u}^{VT}}$

Table 5
Decomposition of the URIS motion

	URIS
Horizontal thrusters (motion in horizontal plane)	Controllable DOF: Surge, Sway, Yaw $\boldsymbol{\tau}^{HT} = \begin{bmatrix} \tau_X \\ \tau_Y \\ \tau_N \end{bmatrix} = \underbrace{K \begin{bmatrix} \overbrace{1}^{HT_1} & \overbrace{1}^{HT_2} & \overbrace{0}^{HT_3} & \overbrace{0}^{HT_4} \\ 0 & 0 & 1 & 1 \\ R & -R & R & -R \end{bmatrix}}_{\mathbf{B}^{HT}} \underbrace{\begin{bmatrix} {}^{1}u^{HT} \\ {}^{2}u^{HT} \\ {}^{3}u^{HT} \\ {}^{4}u^{HT} \end{bmatrix}}_{\mathbf{u}^{HT}}$
Vertical thruster (motion in vertical plane)	Controllable DOF: no motion in the vertical plane

It can be seen that the motion of the FALCON in the vertical plane is determined by vertical thruster and heave force is directly proportional to the value of control signal. In the case of a partial fault in a vertical thruster, the only available solution is to limit angular velocity of the thruster. In the case of total fault (failure), the thruster must be switched off and the vehicle must be recovered for repair.

The situation is different for motion in the horizontal plane, where the number of horizontal thrusters is four and the number of controllable DOF is three. In this case inherent redundancy in thruster configuration enables successful control allocation in the case of partial or even total fault in a horizontal thruster.

The control allocation of horizontal thrusters is the topic of the discussion in the following sections. Before the full problem is formulated, relevant vectors and matrices will be normalised, in order to make problem more understandable and easier to visualise and solve.

5.3. Normalisation

Normalisation means that vector components are divided by their maximum values, such that each component is dimensionless number that lies between -1 and $+1$. Normalised vectors and matrices are underlined, in order to distinguish them from the standard nomenclature. Normalisation procedure will be explained on the example of the X-shaped thruster configuration (the FALCON). Recall from Fig. 6 that maximum thruster force is given by

$$T_m = Ku_m. \tag{18}$$

The first step is to find maximum values (modules) of the surge and sway forces and the yaw moment. Three characteristic cases are indicated in Fig. 7. It can be seen that

$$\tau_{Xm} = 4T_m \cos\alpha = 4Ku_m \cos\alpha \;\Rightarrow\; K\cos\alpha = \frac{\tau_{Xm}}{4u_m}, \tag{19}$$

$$\tau_{Ym} = 4T_m \sin\alpha = 4Ku_m \sin\alpha \;\Rightarrow\; K\sin\alpha = \frac{\tau_{Ym}}{4u_m}, \tag{20}$$

$$\tau_{Nm} = 4T_m A = 4Ku_m A \;\Rightarrow\; KA = \frac{\tau_{Nm}}{4u_m}. \tag{21}$$

The second step is to substitute expressions (19)–(21) in the standard relation $\boldsymbol{\tau}^{HT} = \mathbf{B}^{HT}\mathbf{u}^{HT}$ as follows:

$$\boldsymbol{\tau}^{HT} = \begin{bmatrix} \tau_X \\ \tau_Y \\ \tau_N \end{bmatrix} = \underbrace{\begin{bmatrix} K\cos\alpha & K\cos\alpha & K\cos\alpha & K\cos\alpha \\ K\sin\alpha & -K\sin\alpha & K\sin\alpha & -K\sin\alpha \\ KA & -KA & -KA & KA \end{bmatrix}}_{\mathbf{B}^{HT}} \underbrace{\begin{bmatrix} {}^1u^{HT} \\ {}^2u^{HT} \\ {}^3u^{HT} \\ {}^4u^{HT} \end{bmatrix}}_{\mathbf{u}^{HT}}$$

$$= \begin{bmatrix} \frac{\tau_{Xm}}{4u_m} & \frac{\tau_{Xm}}{4u_m} & \frac{\tau_{Xm}}{4u_m} & \frac{\tau_{Xm}}{4u_m} \\ \frac{\tau_{Ym}}{4u_m} & -\frac{\tau_{Ym}}{4u_m} & \frac{\tau_{Ym}}{4u_m} & -\frac{\tau_{Ym}}{4u_m} \\ \frac{\tau_{Nm}}{4u_m} & -\frac{\tau_{Nm}}{4u_m} & -\frac{\tau_{Nm}}{4u_m} & \frac{\tau_{Nm}}{4u_m} \end{bmatrix} \begin{bmatrix} {}^1u^{HT} \\ {}^2u^{HT} \\ {}^3u^{HT} \\ {}^4u^{HT} \end{bmatrix}. \tag{22}$$

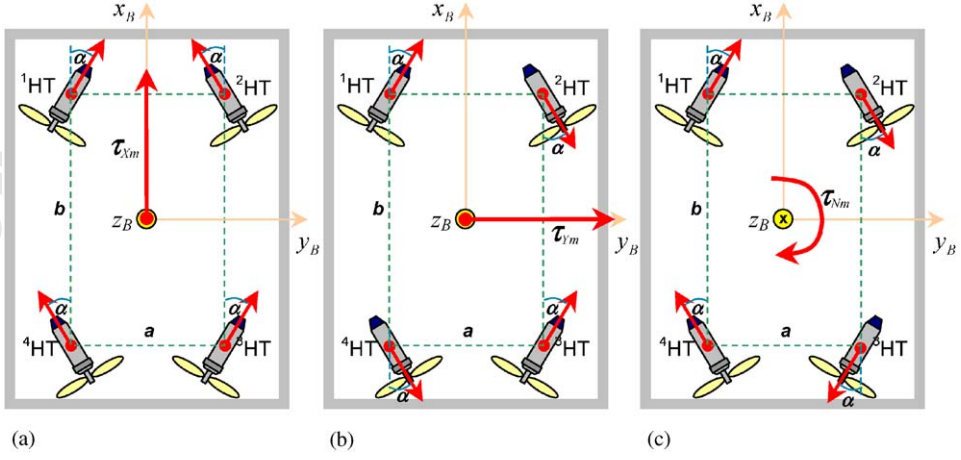

Fig. 7. Three cases for finding the maximum modules of force and moment vectors (X-shaped thruster configuration): (a) Max. surge force τ_{Xm}. (b) Max. sway force τ_{Ym}. (c) Max. yaw moment τ_{Nm}.

Finally, the last step is to rewrite (22) in the normalised form as follows:

$$\underbrace{\begin{bmatrix} \frac{\tau_X}{\tau_{Xm}} \\ \frac{\tau_Y}{\tau_{Ym}} \\ \frac{\tau_N}{\tau_{Nm}} \end{bmatrix}}_{\underline{\boldsymbol{\tau}}^{HT}} = \underbrace{\begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & -\frac{1}{4} & \frac{1}{4} & -\frac{1}{4} \\ \frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} & \frac{1}{4} \end{bmatrix}}_{\underline{\mathbf{B}}^{HT}} \underbrace{\begin{bmatrix} \frac{^1u^{HT}}{u_m} \\ \frac{^2u^{HT}}{u_m} \\ \frac{^3u^{HT}}{u_m} \\ \frac{^4u^{HT}}{u_m} \end{bmatrix}}_{\underline{\mathbf{u}}^{HT}} \Leftrightarrow \underline{\boldsymbol{\tau}}^{HT} = \underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT}. \tag{23}$$

The normalised form (23) has a number of advantages compared to the standard form $\boldsymbol{\tau}^{HT} = \mathbf{B}^{HT}\mathbf{u}^{HT}$ as follows:

1. Components of the vectors $\underline{\boldsymbol{\tau}}^{HT}$ and $\underline{\mathbf{u}}^{HT}$ are dimensionless number, restricted to the standard interval $[-1, +1]$. This enables better understanding and easier visualisation of the problem.
2. All physical parameters are removed from the matrix $\underline{\mathbf{B}}^{HT}$ during the normalisation process. The compact form of $\underline{\mathbf{B}}^{HT}$ simplifies calculations and, as will be shown later, leads to the very simple representation of the weighted pseudo-inverse solution and a clear geometric interpretation of the control allocation problem.

Normalisation for the cross-shaped thruster configuration (URIS) can be performed in a similar way. The final expression is given by

$$\underbrace{\begin{bmatrix} \frac{\tau_X}{\tau_{Xm}} \\ \frac{\tau_Y}{\tau_{Ym}} \\ \frac{\tau_N}{\tau_{Nm}} \end{bmatrix}}_{\underline{\boldsymbol{\tau}}^{HT}} = \underbrace{\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & -\frac{1}{4} & \frac{1}{4} & -\frac{1}{4} \end{bmatrix}}_{\underline{\mathbf{B}}^{HT}} \underbrace{\begin{bmatrix} \frac{^1u^{HT}}{u_m} \\ \frac{^2u^{HT}}{u_m} \\ \frac{^3u^{HT}}{u_m} \\ \frac{^4u^{HT}}{u_m} \end{bmatrix}}_{\underline{\mathbf{u}}'^{HT}} \Leftrightarrow \underline{\boldsymbol{\tau}}^{HT} = \underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT}. \tag{24}$$

5.4. Problem formulation for horizontal thrusters

The control allocation problem for the motion in the horizontal plane can be formulated using normalised variables as follows:

For given $\underline{\boldsymbol{\tau}}^{HT}$, find $\underline{\mathbf{u}}^{HT} \in \underline{\Omega}^{HT}$ such that $\underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT} = \boldsymbol{\tau}^{HT}$.

In the following, the problem is analysed in more details from the general control allocation perspective.

- The true control input is

$$\underline{\mathbf{u}}^{HT} = \begin{bmatrix} \underline{u}_1^{HT} \\ \underline{u}_2^{HT} \\ \underline{u}_3^{HT} \\ \underline{u}_4^{HT} \end{bmatrix} = \begin{bmatrix} \frac{^1u^{HT}}{u_m} \\ \frac{^2u^{HT}}{u_m} \\ \frac{^3u^{HT}}{u_m} \\ \frac{^4u^{HT}}{u_m} \end{bmatrix} \in \Re^4 \quad (m = 4).$$

- The virtual control input is

$$\underline{\boldsymbol{\tau}}^{HT} = \begin{bmatrix} \underline{\tau}_X \\ \underline{\tau}_Y \\ \underline{\tau}_N \end{bmatrix} = \begin{bmatrix} \frac{\tau_X}{\tau_{Xm}} \\ \frac{\tau_Y}{\tau_{Ym}} \\ \frac{\tau_N}{\tau_{Nm}} \end{bmatrix} \in \Re^3 \quad (k = 3).$$

- Control effectiveness matrix $\underline{\mathbf{B}}^{HT}$ is given by

$$\text{FALCON}: \quad \underline{\mathbf{B}}^{HT} = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & -\frac{1}{4} & \frac{1}{4} & -\frac{1}{4} \\ \frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} & \frac{1}{4} \end{bmatrix}. \tag{25}$$

$$\text{URIS}: \quad \underline{\mathbf{B}}^{HT} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & -\frac{1}{4} & \frac{1}{4} & -\frac{1}{4} \end{bmatrix}. \tag{26}$$

- Actuator position constraints are $\begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \leqslant \begin{bmatrix} \underline{u}_1^{HT} \\ \underline{u}_2^{HT} \\ \underline{u}_3^{HT} \\ \underline{u}_4^{HT} \end{bmatrix} \leqslant \begin{bmatrix} +1 \\ +1 \\ +1 \\ +1 \end{bmatrix}.$

Equation $\underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT} = \underline{\boldsymbol{\tau}}^{HT}$ represents the system of equations

$$\text{FALCON}: \quad \begin{aligned} \frac{1}{4}\underline{u}_1^{HT} + \frac{1}{4}\underline{u}_2^{HT} + \frac{1}{4}\underline{u}_3^{HT} + \frac{1}{4}\underline{u}_4^{HT} &= \underline{\tau}_X, \\ \frac{1}{4}\underline{u}_1^{HT} - \frac{1}{4}\underline{u}_2^{HT} + \frac{1}{4}\underline{u}_3^{HT} - \frac{1}{4}\underline{u}_4^{HT} &= \underline{\tau}_Y, \\ \frac{1}{4}\underline{u}_1^{HT} - \frac{1}{4}\underline{u}_2^{HT} - \frac{1}{4}\underline{u}_3^{HT} + \frac{1}{4}\underline{u}_4^{HT} &= \underline{\tau}_N, \end{aligned} \tag{27}$$

$$\text{URIS}: \quad \begin{aligned} \frac{1}{2}\underline{u}_1^{HT} + \frac{1}{2}\underline{u}_2^{HT} &= \underline{\tau}_X. \\ \frac{1}{2}\underline{u}_3^{HT} + \frac{1}{2}\underline{u}_4^{HT} &= \underline{\tau}_Y, \\ \frac{1}{4}\underline{u}_1^{HT} - \frac{1}{4}\underline{u}_2^{HT} + \frac{1}{4}\underline{u}_3^{HT} - \frac{1}{4}\underline{u}_4^{HT} &= \underline{\tau}_N. \end{aligned} \tag{28}$$

Each equation in (27) and (28) represents a hyperplane in $\Re^4$. Consequently, (27) and (28) can be rewritten as

$$\begin{aligned} \pi_X&: \quad \mathbf{N}_X^{\mathrm{T}} \cdot \underline{\mathbf{u}}^{HT} = \underline{\tau}_X, \\ \pi_Y&: \quad \mathbf{N}_Y^{\mathrm{T}} \cdot \underline{\mathbf{u}}^{HT} = \underline{\tau}_Y, \\ \pi_N&: \quad \mathbf{N}_N^{\mathrm{T}} \cdot \underline{\mathbf{u}}^{HT} = \underline{\tau}_N, \end{aligned} \tag{29}$$

where normal vectors $\mathbf{N}_X$, $\mathbf{N}_Y$ and $\mathbf{N}_N$, defined as

$$\text{FALCON}: \quad \begin{aligned} \mathbf{N}_X^{\mathrm{T}} &= \left[\frac{1}{4}\ \frac{1}{4}\ \frac{1}{4}\ \frac{1}{4}\right]^{\mathrm{T}}, \\ \mathbf{N}_Y^{\mathrm{T}} &= \left[\frac{1}{4}\ -\frac{1}{4}\ \frac{1}{4}\ -\frac{1}{4}\right]^{\mathrm{T}}, \\ \mathbf{N}_N^{\mathrm{T}} &= \left[\frac{1}{4}\ -\frac{1}{4}\ -\frac{1}{4}\ \frac{1}{4}\right]^{\mathrm{T}}. \end{aligned} \tag{30}$$

$$\text{URIS}: \quad \begin{aligned} \mathbf{N}_X^{\mathrm{T}} &= \left[\frac{1}{2}\ \frac{1}{2}\ 0\ 0\right]^{\mathrm{T}}, \\ \mathbf{N}_Y^{\mathrm{T}} &= \left[0\ 0\ \frac{1}{2}\ \frac{1}{2}\right]^{\mathrm{T}}, \\ \mathbf{N}_N^{\mathrm{T}} &= \left[\frac{1}{4}\ -\frac{1}{4}\ \frac{1}{4}\ -\frac{1}{4}\right]^{\mathrm{T}} \end{aligned} \tag{31}$$

are orthogonal on the hyperplanes π_X, π_Y and π_N, respectively. The intersection of these hyperplanes is a convex set, denoted by $\underline{\aleph}^{HT}$, which represent the set of all points $\underline{\mathbf{u}}^{HT}$ that satisfy $\underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT} = \underline{\tau}^{HT}$.

The thruster velocity saturation constraints determine the constrained control subset $\underline{\Omega}^{HT}$, that is, the unit four-dimensional hypercube in $\Re^4$:

$$\underline{\Omega}^{HT} = \{\underline{\mathbf{u}}^{HT} \in \Re^4 \mid \|\underline{\mathbf{u}}^{HT}\|_\infty \leqslant 1\} \subset \Re^4. \tag{32}$$

Intersection of $\underline{\aleph}^{HT}$ and $\underline{\Omega}^{HT}$ is a solution set, denoted by $\underline{\Im}^{HT}$. Geometric interpretation of the control allocation problem for the motion in the horizontal plane using normalised variables is given by

For a given $\underline{\tau}^{HT}$, find intersection $\underline{\Im}^{HT} = \underline{\Re}^{HT} \bigcup \underline{\Omega}^{HT}$.

The control effectiveness matrix $\underline{\mathbf{B}}^{HT}$ performs a linear transformation from the true control space $\Re^3$ to the virtual control space $\Re^4$. The image of $\underline{\Omega}^{HT} \subset \Re^4$ is called the attainable command set and denoted by $\underline{\Phi}^{HT}$. The boundary of $\underline{\Phi}^{HT}$ is denoted by $\partial(\underline{\Phi}^{HT})$. The character of the solution is closely related with the position of the vector $\underline{\tau}^{HT}$ relative to $\underline{\Phi}^{HT}$. Three cases are possible:

- If $\underline{\tau}^{HT} \notin \underline{\Phi}^{HT}$, then $\underline{\Im}^{HT}$ is empty (i.e. no exact solution exists),
- If $\underline{\tau}^{HT} \in \partial(\underline{\Phi}^{HT})$, then $\underline{\Im}^{HT}$ has exactly one element (i.e. there is one, unique solution[3]),
- $\underline{\Im}^{HT}$ has more than one element (i.e. there are many solutions).

In order to extract a unique, "best" solution from a solution set, it is necessary to introduce criteria, which is minimised by the chosen solution. The most suitable criteria for underwater applications is a control energy cost function.

The optimal control input $\underline{\mathbf{u}}^{HT}$ is given as a solution to a two-step optimisation problem:

$$\underline{\mathbf{u}}^{HT} = \arg \min_{\underline{\mathbf{u}}^{HT} \in \underline{\Psi}^{HT}} \|\mathbf{W}_u^{HT}\underline{\mathbf{u}}^{HT}\|_2, \tag{33}$$

$$\underline{\Psi}^{HT} = \arg \min_{\underline{\mathbf{u}}^{HT} \in \underline{\Omega}^{HT}} \|\underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT} - \underline{\tau}^{HT}\|_2. \tag{34}$$

Problem (33)–(34) can be interpreted as follows:

Given $\underline{\Psi}^{HT}$, the set of feasible control inputs that minimise $\|\underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT} - \underline{\tau}^{HT}\|_2$, find the control input $\underline{u}^{HT}$ that minimises $\|\mathbf{W}_u^{HT}\underline{\mathbf{u}}^{HT}\|_2$.

The design parameter $\mathbf{W}_u^{HT}$ is a positive definite weighting matrix, weighting the control energy, and can be used for thruster prioritisation, i.e. to decide which thruster should be used primarily. The weighting matrix $\mathbf{W}_u^{HT}$ is usually chosen to be a diagonal matrix

$$\mathbf{W}_u^{HT} = \begin{bmatrix} w_1^{HT} & 0 & 0 & 0 \\ 0 & w_2^{HT} & 0 & 0 \\ 0 & 0 & w_3^{HT} & 0 \\ 0 & 0 & 0 & w_4^{HT} \end{bmatrix}, \tag{35}$$

where $w_i^{HT} > 0$ is the weight associated with the thruster ${}^iHT$, $i = \overline{1,4}$. Using $\mathbf{W}_u^{HT}$, a faulty thruster is penalised by increasing its weight, as explained in the following.

5.5. Weighting matrix $\mathbf{W}_u^{HT}$ for fault-free case

In the fault-free case, all horizontal thrusters have the same priority and $\mathbf{W}_u^{HT}$ is chosen to be equal to identity matrix

$$\mathbf{W}_u^{HT} = \mathbf{I}_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{36}$$

5.6. Weighting matrix $\mathbf{W}_u^{HT}$ for faulty situations

Two faulty situations are possible: a *partial* fault and a *total* fault (failure).

In the case of a partial fault in ${}^iHT$, the thruster is typically allowed to continue operation with the restricted usage, i.e. the new constraint (saturation) bounds are

$$-s_i^{HT} \leqslant \underline{u}_i^{HT} \leqslant s_i^{HT}, \tag{37}$$

[3] The matrix $\underline{\mathbf{B}}^{HT}$ satisfies the *linear independency condition* (non-complanar controls), since every 3×3 partition of $\underline{\mathbf{B}}^{HT}$ is non-singular. This leads to a unique solution of the control allocation problem for the case when virtual control input lies on the boundary $\partial(\underline{\Phi}^{HT})$.

where $0<s_i^{HT}<1$. The numerical value of s_i^{HT} depends on the type of the fault and is selected in advance for each particular fault type. For example, restricted constraint bound $s_i^{HT}=0.75$ can be selected for a faulty state "Jammed propeller" of ${}^iHT$, which means that the thruster's operating range is restricted to 75% of its nominal range. In addition to the change of the constraint bounds, the weight w_i^{HT} of the faulty thruster is increased using

$$w_i^{HT}=1+\Delta w_i^{HT}, \tag{38}$$

where

$$\Delta w_i^{HT}=2\left(\frac{1}{s_i^{HT}}-1\right). \tag{39}$$

The weight update (39) is introduced to penalise the faulty thruster, prioritise healthy thrusters and to compensate restricted usage of the faulty thruster in an optimal way.

In the case of a total fault in ${}^iHT$, the thruster is switched off and removed from the allocation process. The same effect can be achieved using formulation (33) by allowing $w_i^{HT}\to\infty$. In this way, the redundancy is eliminated from the system of equation $\underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT}=\underline{\boldsymbol{\tau}}^{HT}$, which can now be solved in a standard way.

5.7. Remarks on vertical thrusters

For the motion of the FALCON in the vertical plane normalisation yields

$$\left.\begin{aligned}\boldsymbol{\tau}^{VT}=[\tau_Z]&=\underbrace{K}_{\mathbf{B}^{VT}}\underbrace{u^{VT}}_{\mathbf{u}^{VT}}\\ K&=\frac{\tau_{Zm}}{u_m}\end{aligned}\right\}\Rightarrow\underbrace{\begin{bmatrix}\tau_Z\\ \tau_{Zm}\end{bmatrix}}_{\underline{\boldsymbol{\tau}}^{VT}}=\underbrace{[1]}_{\underline{\mathbf{B}}^{VT}}\underbrace{\begin{bmatrix}{}^1u^{VT}\\ u_m\end{bmatrix}}_{\underline{\mathbf{u}}^{VT}}$$

$$\Leftrightarrow\underline{\boldsymbol{\tau}}^{VT}=\underline{\mathbf{B}}^{VT}\underline{\mathbf{u}}^{VT}. \tag{40}$$

The weighting matrix for vertical thruster $\mathbf{W}_u^{VT}$ becomes a scalar, defined as

$$\mathbf{W}_u^{VT}=w_1^{VT}, \tag{41}$$

where $w_1^{VT}>0$ is the weight associated with the thruster ${}^1VT$.

In the fault-free case, $w_1^{VT}=1$.

In the case of a partial fault, the new restricted constraint (saturation) bounds are

$$-s_1^{VT}\leqslant\underline{u}_1^{VT}\leqslant s_1^{VT}, \tag{42}$$

where $0<s_1^{VT}<1$. Since $\underline{\mathbf{B}}^{VT}=1$, weighted pseudo-inverse solution $\underline{\mathbf{B}}^{+VT}_{W_u^{VT}}$ does not depends on w_1^{VT}. Hence, restriction of constraint bounds (42) is the only action that is undertaken in the case of a partial fault in the vertical thruster.

In the case of a total fault, the thruster ${}^1VT$ must be switched off and removed from the allocation process. In this case, heave becomes uncontrollable DOF.

6. Fault diagnosis and accommodation system

6.1. Architecture

The overall functional architecture of the proposed FDAS, shown in Fig. 8, represents the expanded version of the improved architecture shown in Fig. 2b. The description of the architecture will be given in an hierarchical way, such that the general description and the main idea of the method are presented in this section, while more details about individual components can be found in the following subsections.

The input to the FDAS is the vector $\underline{\boldsymbol{\tau}}_d$, obtained by filtering (smoothing) the desired vector of propulsion forces and moments $\underline{\boldsymbol{\tau}}_d^*$, generated by the HCU. The output of the FDAS is the vector of desired thruster velocities $\mathbf{n}$, transformed into the form that is compatible (acceptable) by the TCUs. A short description of the individual components is given in the following.

6.1.1. FDS

FDU: The FDS uses FDUs to monitor the state of the thrusters. The FDU is a software module associated with the thruster, able to detect internal faults (for example, temperature of the windings exceeds limits) and external faults (for example, jammed propeller). The output of the FDU is a fault indicator f_i, the code of the fault.

Integration: The fault indicators f_i are integrated into the *total fault indicator vector* $\mathbf{f}$ inside this block. The vector $\mathbf{f}$ is a carrier of thrusters' states.

6.1.2. FAS

Demux: In accordance with the decomposition of the motion, shown in Tables 4 and 5, this block symbolically indicates separation of the vector $\underline{\boldsymbol{\tau}}_d$ into two parts: $\underline{\boldsymbol{\tau}}_d^{HT}$, representing the DOF (surge, sway and yaw) controllable by horizontal thrusters, and $\underline{\boldsymbol{\tau}}_d^{VT}$, representing the DOF (heave) controllable by vertical thruster.

Pseudo-inverse: This block finds weighted pseudo-inverse solution of the control allocation problem, separately for horizontal and vertical thrusters. For horizontal thrusters, the solution $\underline{\mathbf{u}}^{HT}$ is given by (49). For vertical thruster, the solution $\underline{\mathbf{u}}^{VT}$ is given by (51).

Approximation: The weighted pseudo-inverse solution $\underline{\mathbf{u}}^{HT}$ can be feasible (satisfies all constraints) or unfeasible (violates some constraint(s)). The output $\underline{\mathbf{u}}^{*HT}$ of this block must be feasible solution all the time. Hence, if $\underline{\mathbf{u}}^{HT}$ is feasible, then $\underline{\mathbf{u}}^{*HT}=\underline{\mathbf{u}}^{HT}$. Otherwise, T-approximation (truncation) or S-approximation (scaling) is perform to find feasible approximation $\underline{\mathbf{u}}^{*HT}$.

Co-ordinator: The role of this block is to undertake remedial actions in accordance to the context of the total fault indicator vector $\mathbf{f}$ and the instructions, stored in the fault code table. For each possible fault type the

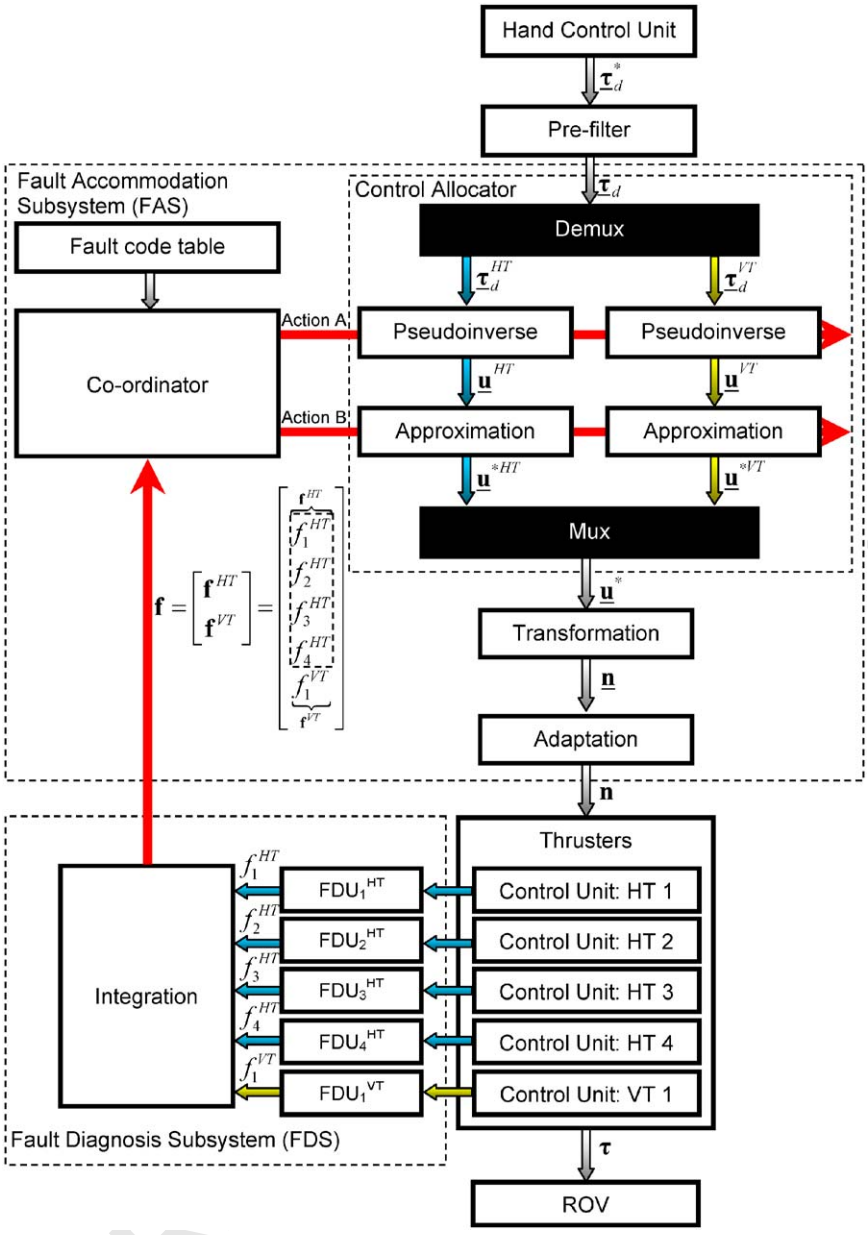

Fig. 8. Overall functional architecture of the proposed FDAS.

fault code table has stored corresponding actions A and B. The action A is related with the weight updates of weighting matrices, used to find the weighted pseudo-inverse solution. The action B is related with change of constraint bounds, in accordance to fault type.

Mux: This block performs opposite role of Demux block, i.e. it merges feasible solutions $\underline{\mathbf{u}}^{*HT}$ and $\underline{\mathbf{u}}^{*VT}$ into composite solution vector $\underline{\mathbf{u}}^{*}$.

Transformation: The vector $\underline{\mathbf{u}}^{*}$ cannot be directly applied to drive thrusters. It must be transformed into the vector of desired thruster velocities $\underline{\mathbf{n}}$. This block performs this function, using transformation $\underline{n}_i = \operatorname{sgn} \underline{u}_i \sqrt{|\underline{u}_i|}$ for each component.

Adaptation: Different TCUs accept data in different format. For example, the desired angular velocity for the TCU of the FALCON must be represented as an integer number between -100 and $+100$. In contrast, the same variable must be converted into the voltage in order to be applied to drive the thruster of the URIS. This block transforms the vector $\underline{\mathbf{n}}$ into the vector $\mathbf{n}$, which has the form adapted for the particular TCU.

Vector $\mathbf{n}$ is used to drive the thrusters, which generate a vector of propulsion forces and moments $\boldsymbol{\tau}$. The proposed FDAS guarantees that the condition $\boldsymbol{\tau} = \boldsymbol{\tau}_d$ is satisfied for all $\underline{\boldsymbol{\tau}}_d^{HT}$ that lies inside the convex polyhedron $\underline{\Phi}_p^{HT}$ (feasible region for pseudo-inverse),

which is a subset of the attainable command set Φ^{HT}. That is, if $\underline{\tau}_d^{HT} \in \underline{\Phi}_p^{HT}$, the FDAS will find the exact solution of the control allocation problem, optimal in l_2 sense. Otherwise, the solution obtained by the FDAS is a very good approximation that lies on the boundary $\partial(\Omega^{HT})$, which depends on design parameters (weighting matrices) and type of approximation.

6.2. Fault diagnosis subsystem

6.2.1. Fault classification

Thrusters are liable to different fault types during the underwater mission. Some of these faults (partial faults) are not critical and the thruster is able to continue operation in the presence of a fault with the restricted usage, i.e. reduced maximum velocity. In other cases (total faults—failures) the thruster must be switched off and the mission has to be continued with the remaining operable thrusters. Thruster faults are classified into two main classes (Omerdic et al., 2003):

- *Internal faults* (e.g. temperature of the windings is out of range, lost communication between the TCU and main processor, drop in bus voltage, etc.)
- *External faults* (e.g. jammed or broken propeller).

6.2.2. Fault code table

Relationships between thruster states, fault types and remedial actions are stored in the fault code table (Table 6). It must be emphasized, at this point, that this fault code table is just a suggestion, intended to reveal the main ideas of the proposed FDAS. New states (rows) can be added, and the existing relationships can be changed, in order to cover intermediate cases and meet specific requirements.

6.2.3. Fault detection unit

The FDU is used for monitoring the thruster states and reporting any faulty situation. The FDU is a software module, able to detect internal and external faults. Connections between the FDU and the TCU for arbitrary thruster ${}^{i}Th$ are indicated in Fig. 9.

Signals $Int_1, Int_2, \ldots$ for detection of internal faults are already available in existing TCUs for both vehicles. In particular, the TCU for the URIS, based on Maxon Servoamplifier ADS 50/5, has status-reading signal Int_1 = "Ready", which can be used to report internal faults (excess temperature or excess current). Similarly, communication protocol for FALCON provides monitoring of the winding temperature (Int_1) and bus voltage (Int_2) of each thruster. In order to build a universal FDU, capable of detecting both internal and external faults, it is necessary to augment the existing

Table 6
The fault code table

Thruster state	Class	Type	Indicator f_i	Constraint bound s_i
Normal (Fault-free)	—	—	1	1.00
Jammed propeller	Ext.	Partial	2	0.75
Heavy jammed propeller	Ext.	Partial	3	0.50
Broken propeller	Ext.	Total	4	0.00
Unknown fault	Ext.	Partial	5	0.25
Internal fault	Int.	Total	6	0.00

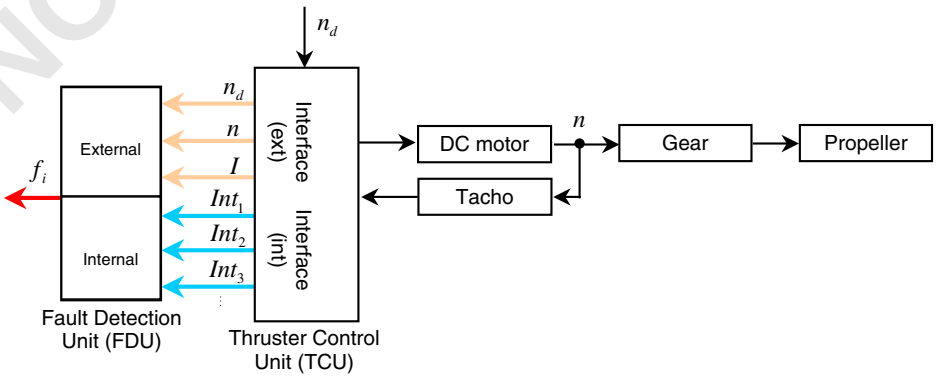

Fig. 9. Block diagram showing connections between the FDU and the TCU for thruster ${}^{i}Th$.

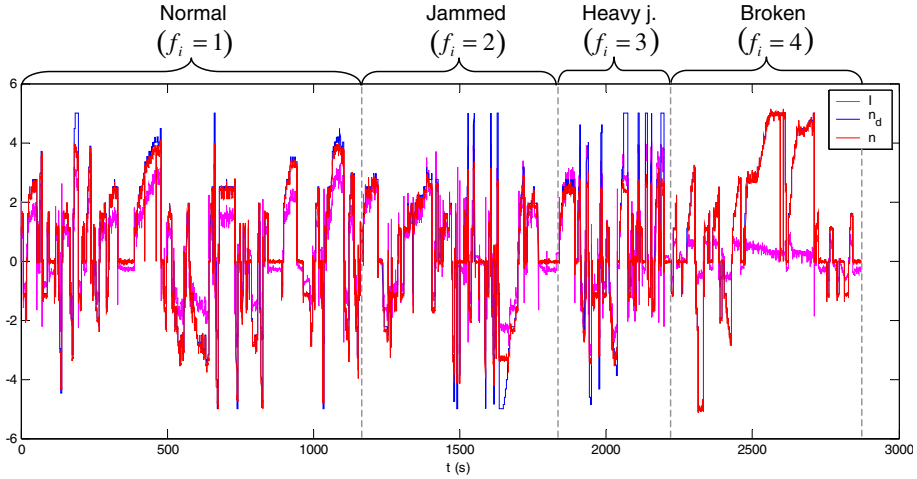

Fig. 10. Time diagrams of raw training data.

internal protection with a software module for fast and reliable detection of external faults.

For detection of external faults available signals are actual velocity of the motor shaft n and current consumption I of the thruster. For the URIS, these signals are called "Monitor n" and "Monitor I", respectively; for the FALCON, the communication protocol enables output speed and winding current to be read. By monitoring n and I, together with desired speed n_d obtained as output of the FDAS, the FDU must be able to detect external thruster fault.

Finally, the universal FDU integrates both parts (internal and external) into one unit, which is able to detect internal and external faults (Fig. 9). Integration is performed using a priority scheme, where total faults have higher priority than partial faults. Indicator f_i, the output of the FDU, is the code of the fault.

Implementation of the FDU involves two phases: *off-line training* and *on-line fault detection*.

Off-line training phase: The first stage in the training phase is acquisition of training data. Test trials were performed with the URIS at University of Girona in July 2002, and training data were saved in files. Normal state and three different fault cases were considered (jammed, heavy jammed and broken propeller).[4] A jammed propeller was simulated by attaching an object to it. When the thruster is actuated, the propeller and the object rotate together, representing additional load for the motor. Heavy-jammed propeller was simulated with two objects attached. In order to simulate broken propeller, blades were removed from the shaft. Each record in file consists of acquired data from the TCU (n_d, n and I) and associated fault code f_i. Sampling time was 0.1 s, long enough to ensure that all transient

[4] However, in real applications the number of faulty cases can be higher. In addition, partially broken or damaged propeller blades can be used to cover intermediate cases.

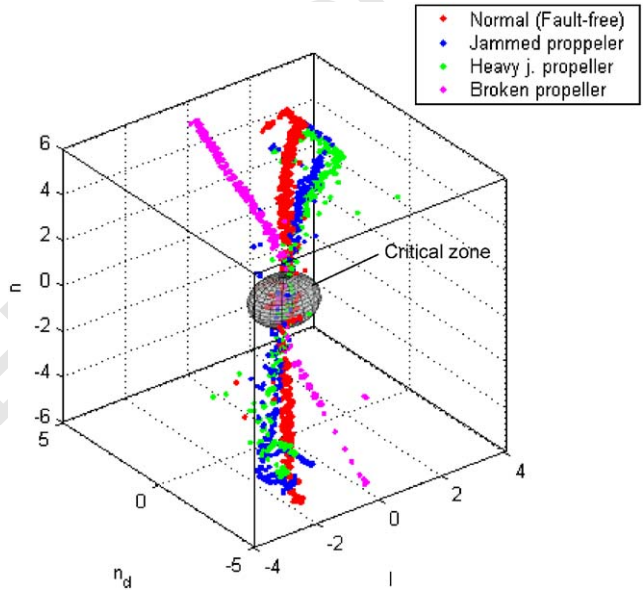

Fig. 11. Pre-processed training data in the 3D space.

responses in thruster control loop disappear. The motion of the URIS was controlled by a joystick, such that all range of possible thruster velocities was covered with enough data points. The real-time experiments were undertaken during the development stage of the URIS, what meant that the inadequate signal conditioning, wiring and shielding resulted in noisy data.

The time diagrams of the raw training data are shown in Fig. 10. Signals from different fault types are connected next to each other in order to make easier their comparison. Noise and *outliers* (data items that lie very far from the main body of the data) are particularly noticeable in current response.

Data pre-processing filters the raw training data in order to remove outliers and reject noise. Pre-processed

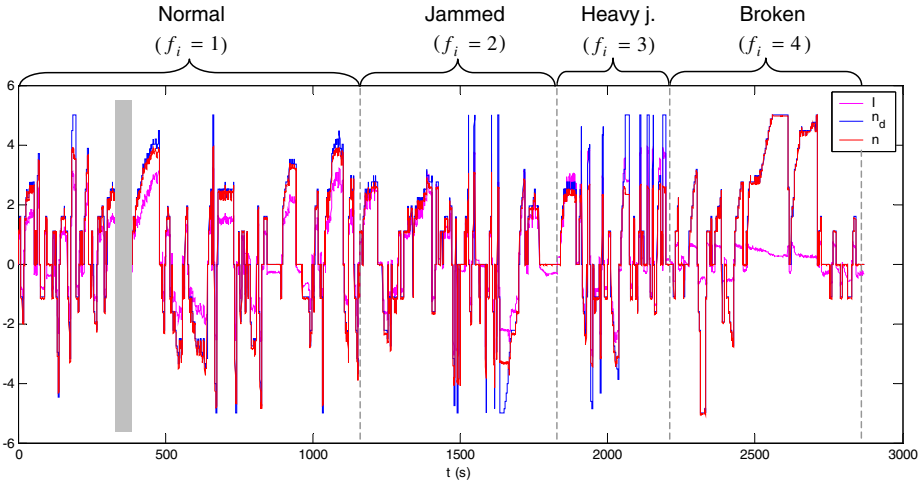

Fig. 12. Time diagrams of pre-processed training data. One of the zero-velocity segments is highlighted. These segments are excluded from the training process.

training data in 3D space are shown in Fig. 11.[5] Fig. 12 displays time diagrams of pre-processed data. Normalisation step in pre-processing is optional, since all variables have the same range and no one is dominant.

Analysing the distribution of the training data in Fig. 11, the *first* feature that can be noticed is that each fault-type creates certain pattern. In the ideal case, these patterns should be well defined curves. However, the presence of the noise and outliers in training data results that patterns shown in Fig. 11 are "cloudy". The *second* feature is that the zone around $n_d \approx 0$ (called *the critical zone*) is filled with data from different fault types in such a way that it is very hard to distinguish individual fault types. Geometrically, the critical zone represents intersection of different fault-type patterns. This makes successful fault detection in the critical zone difficult to achieve. In particular, for zero-velocity case $n_d = 0$ thruster does not rotate. Successful and reliable fault detection in this case is impossible, since external faults cannot be detected without shaft rotation. The solution for this problem is an exclusion of the zero-velocity segments from the training process, i.e. the training is performed considering data records with $n_d \neq 0$, as shown in Fig. 12. However, the same exclusion is performed during the on-line fault detection phase.

The problem of thruster fault detection for underwater vehicles has special features, due to environmental conditions in which the vehicle operates. The most important requirements that the FDU should fulfil are:

- reliable and fast fault detection, without false alarms,
- easy integration with the existing control system,
- on-line learning and adaptation to the new types of faults,

[5] Matlab function `medfilt1` was used to filter data and remove outliers at the same time.

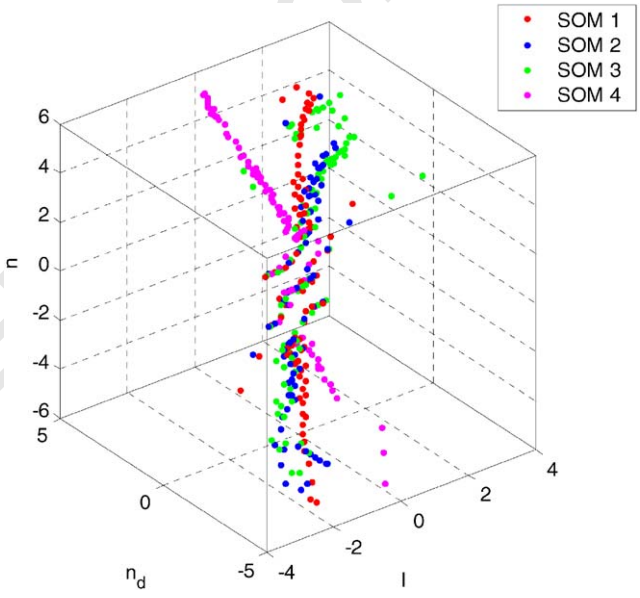

Fig. 13. The second stage in off-line training phase: different fault types (patterns) are replaced by SOM representatives.

- cost efficient i.e. the FDU should use resources already available, without introducing new hardware,
- easy transfer to and implementation in other vehicles.

By carefully examination of the available resources in the existing TCUs, the model-free approach, based on integration of an SOM and fuzzy clustering techniques, is chosen as the best candidate for FDU to fulfil all these requirements (Omerdic et al., 2003). The main idea of the second stage in the training phase is to replace each fault type (pattern) in Fig. 11 with an SOM as shown in Fig. 13, which serves as a representative of the particular fault type. A fault type with code $f_i = k$ is replaced with

SOM k. Each SOM k is an one-dimensional array of 100 neurons. Each of these neurons has an associated prototype vector with three coordinates. The distribution of prototype vectors (Fig. 13) in the input space was found using fuzzy c-means clustering and approximately 80% data from each fault type. Each prototype vector is cluster centre and representative of all data from its cluster.

Finally, in the third and the last stage in the training phase the structure of the SOM representatives is saved on hard disk for future use. In this way, heavy demanding calculations are performed off-line, during the training phase, which enables fast and efficient fault detection during the on-line phase.

On-line fault detection phase: During the initialisation stage of the on-line fault detection phase, the main processor reads the structure of the SOM representatives, saved on hard disk during the training phase, and stores it in the working memory for fast access.

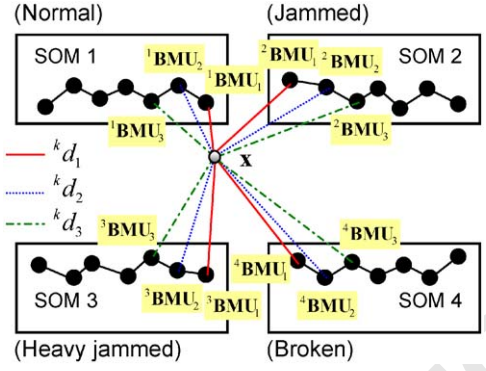

Fig. 14. On-line fault detection phase: position of the feature vector is determined relative to SOM representatives by finding three closest BMUs in each SOM.

After the initialisation is finished, the fault detection is performed by repeating the following steps at each program cycle:

Three closest codebook vectors (Best Matching Units—BMUs) from each map to feature vector $\mathbf{x}$ (which consists of pre-processed, actual measurements of n_d, n and I) are computed, together with corresponding distances (Fig. 14), where ${}^k\mathbf{BMU}_j$ means jth BMU in SOM k, while ${}^kd_j$ means Euclidian distance between $\mathbf{x}$ and ${}^k\mathbf{BMU}_j$, $k=\overline{1,4}$, $j=\overline{1,3}$. In the next step matrix $\mathbf{M}=[{}^kd_j]_{4\times 3}$ is created. Minimum values of each column of $\mathbf{M}$ are found and the indices of the minimum values are stored in row vector $\mathbf{b}$. For example, $\mathbf{b}=[1\ 3\ 2]$ means that the closest first BMU is in SOM 1, second in SOM 3 and third in SOM 2. In order to avoid false detection, the final decision about faulty state is accomplished using present and past vectors $\mathbf{b}$, which are stored in the buffer with size $s\times 3$. If all buffer elements have the same value, then the fault indicator f_i, output of the FDU, is set to this value. Finally, outputs of each FDU are integrated to form the total fault indicator vector $\mathbf{f}$:

$$\mathbf{f}=[f_1\ f_2\ f_3\ f_4]^{\mathrm{T}}. \tag{43}$$

In a fault-free case all four indicators f_i are equal to 1, in accordance to the fault code table. It is assumed that at any time instance at most one horizontal thruster can be faulty i.e. at most one of the indicators f_i can be different than 1. Simultaneous faults in multiple thrusters are very rare in practical operations and in these cases unavoidable loss in controllability of some DOF will occur. Only faults in a single thruster are considered in this paper.

6.2.4. Evaluation

As stated before, a large data set was acquired during test trials and only a part of this data was used for

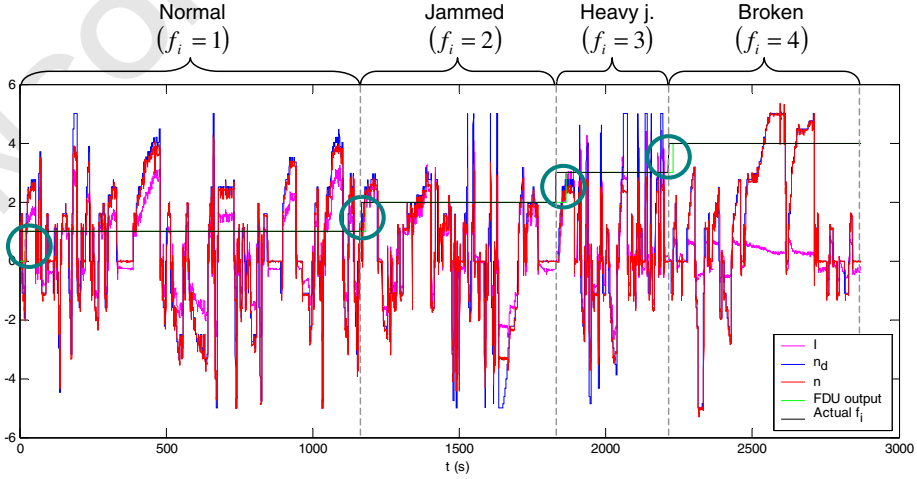

Fig. 15. Evaluation of the FDU.

training. The capability of the proposed FDU to detect external faults is evaluated using the entire data set. Only signals n_d, n and I are presented as inputs to the FDU, which must estimate the state of the thruster using only these inputs. Fig. 15 displays actual fault code and FDU output, together with training data. It can be seen that the FDU identifies the new thruster state correctly in a short time after the change in state, as highlighted in Fig. 15. This delay is unavoidable, because the thruster must spend some time in a faulty state before the fault can be identified. Delay is proportional to the buffer size s. A conservative value $s = 25$ was used in Fig. 15, in order to prevent a wrong detection. It is expected that the buffer size and delay will be reduced in future similar experiments with the FALCON, due to advanced signal conditioning and better quality of measured signals.

6.3. Fault accommodation subsystem

6.3.1. Introduction

An ROV pilot uses the HCU (Fig. 8) to generate the input command vector $\underline{\boldsymbol{\tau}}_d$. The FDS finds the total fault indicator vector $\mathbf{f}$ with information about healthy state of each thruster. The FAS uses these two vectors and relationships in the fault code table to solve the control allocation problem separately for motion in horizontal and vertical plane.

The solution method adopted in the FAS relies on the fact that explicit solution to the unconstrained control allocation problem:

$$\min_{\mathbf{u}} \|\mathbf{W}\mathbf{u}\|_2 \tag{44}$$

subject to

$$\mathbf{B}\mathbf{u} = \boldsymbol{\tau}_d \tag{45}$$

is given by (Fossen, 1995)

$$\mathbf{u} = \mathbf{B}_{\mathbf{W}}^{+}\boldsymbol{\tau}_d = (\mathbf{W}^{-1}\mathbf{B}^{\mathrm{T}}(\mathbf{B}\mathbf{W}^{-1}\mathbf{B}^{\mathrm{T}})^{-1})\boldsymbol{\tau}_d, \tag{46}$$

where $\mathbf{B}_{\mathbf{W}}^{+}$ is weighted pseudo-inverse of $\mathbf{B}$. Solution (46) is obtained using Lagrange multipliers.

Horizontal plane: For motion in the horizontal plane the weighted pseudo-inverse matrix $\underline{\mathbf{B}}_{\mathbf{W}_u^{HT}}^{+HT}$ is given by

$$\text{FALCON}: \quad \underline{\mathbf{B}}_{\mathbf{W}_u^{HT}}^{+HT} = \frac{1}{\sum_{i=1}^{4} w_i^{HT}} \times \begin{bmatrix} 2(w_3^{HT}+w_4^{HT}) & 2(w_2^{HT}+w_3^{HT}) & 2(w_2^{HT}+w_4^{HT}) \\ 2(w_3^{HT}+w_4^{HT}) & -2(w_1^{HT}+w_4^{HT}) & -2(w_1^{HT}+w_3^{HT}) \\ 2(w_1^{HT}+w_2^{HT}) & 2(w_1^{HT}+w_4^{HT}) & -2(w_2^{HT}+w_4^{HT}) \\ 2(w_1^{HT}+w_2^{HT}) & -2(w_2^{HT}+w_3^{HT}) & 2(w_1^{HT}+w_3^{HT}) \end{bmatrix}, \tag{47}$$

$$\text{URIS}: \quad \underline{\mathbf{B}}_{\mathbf{W}_u^{HT}}^{+HT} = \frac{1}{\sum_{i=1}^{4} w_i^{HT}} \times \begin{bmatrix} (2w_2^{HT}+w_3^{HT}+w_4^{HT}) & (w_3^{HT}-w_4^{HT}) & 2(w_3^{HT}+w_4^{HT}) \\ (2w_1^{HT}+w_3^{HT}+w_4^{HT}) & (w_4^{HT}-w_3^{HT}) & -2(w_3^{HT}+w_4^{HT}) \\ (w_1^{HT}-w_2^{HT}) & (w_1^{HT}+w_2^{HT}+2w_4^{HT}) & 2(w_1^{HT}+w_2^{HT}) \\ (w_2^{HT}-w_1^{HT}) & (w_1^{HT}+w_2^{HT}+2w_3^{HT}) & -2(w_1^{HT}+w_2^{HT}) \end{bmatrix}. \tag{48}$$

These expressions are obtained by combining (25), (26) and (46).

The weighted pseudo-inverse solution of the unconstrained control problem for motion in the horizontal plane is given by

$$\underline{\mathbf{u}}^{HT} = \underline{\mathbf{B}}_{\mathbf{W}_u^{HT}}^{+HT}\underline{\boldsymbol{\tau}}_d^{HT}. \tag{49}$$

Vertical plane: Combining (40), (41) and (46), the weighted pseudo-inverse matrix $\underline{\mathbf{B}}_{\mathbf{W}_u^{VT}}^{+VT}$ for motion in the vertical plane becomes scalar

$$\text{FALCON}: \quad \underline{\mathbf{B}}_{\mathbf{W}_u^{VT}}^{+VT} = 1. \tag{50}$$

The weighted pseudo-inverse solution of the unconstrained control problem for motion in the vertical plane is given by

$$\underline{\mathbf{u}}^{VT} = \underline{\mathbf{B}}_{\mathbf{W}_u^{VT}}^{+VT}\underline{\boldsymbol{\tau}}_d^{VT} = \underline{\boldsymbol{\tau}}_d^{VT}. \tag{51}$$

6.3.2. Feasibility of the weighted pseudo-inverse solution

The input command vector $\underline{\boldsymbol{\tau}}_d^{HT} = [\tau_X \ \tau_Y \ \tau_N]^{\mathrm{T}}$ for the motion in the horizontal plane, generated by the HCU, belongs to the virtual control space $\underline{\Phi}_v^{HT}$, the unit cube in $\Re^3$:

$$\underline{\Phi}_v^{HT} = \{\underline{\boldsymbol{\tau}}^{HT} \in \Re^3 \mid \|\underline{\boldsymbol{\tau}}^{HT}\|_\infty \leqslant 1\} \subset \Re^3. \tag{52}$$

For the constrained control allocation problem, where the constraint $\underline{\mathbf{u}}^{HT} \in \underline{\Omega}^{HT}$ is required to be satisfied, solution (49) may become unfeasible, depending on the position of $\underline{\boldsymbol{\tau}}_d^{HT}$ inside $\underline{\Phi}_v^{HT}$. The virtual control space $\underline{\Phi}_v^{HT}$ can be partitioned into characteristic regions, as indicated in Fig. 16. The two characteristic regions inside $\underline{\Phi}_v^{HT}$ are $\underline{\Phi}_p^{HT}$ (feasible region for pseudo-inverse) and $\underline{\Phi}^{HT} \supset \underline{\Phi}_p^{HT}$ (attainable command set). The shape of $\underline{\Phi}_p^{HT}$ is found from condition $\underline{\mathbf{u}}^{HT} = \underline{\mathbf{B}}_{\mathbf{W}_u^{HT}}^{+HT}\underline{\boldsymbol{\tau}}_d^{HT} \in \underline{\Omega}^{HT}$. It should be emphasised that, for the general constrained control allocation problem, there is an infinite number of exact solutions for $\underline{\boldsymbol{\tau}}_d^{HT} \in \underline{\Phi}^{HT}$, while no exact solution exists for $\underline{\boldsymbol{\tau}}_d^{HT} \in \underline{\Phi}_v^{HT} \backslash \underline{\Phi}^{HT}$. The weighted pseudo-inverse is able to find the exact feasible solution of the control allocation problem, optimal in the l_2 sense, only if $\underline{\boldsymbol{\tau}}_d^{HT} \in \underline{\Phi}_p^{HT}$. Otherwise, for $\underline{\tau}_d^{HT} \in \underline{\Phi}_v^{HT} \backslash \underline{\Phi}_p^{HT}$ the solution obtained by pseudo-inverse is unfeasible and cannot be directly applied to the thrusters. An unfeasible solution means that some components (controls) of the control vector $\underline{\mathbf{u}}$ violate constraints. In this case the unfeasible pseudo-inverse solution is approximated in order to get

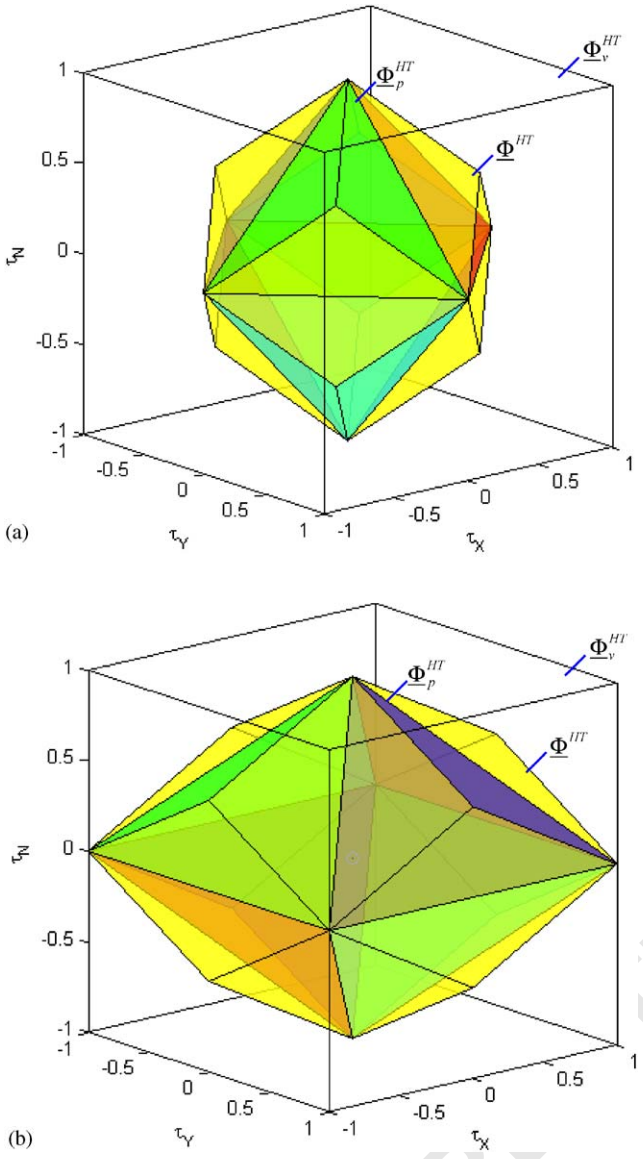

Fig. 16. Partitions of the virtual control space for motion in the horizontal plane: (a) FALCON. (b) URIS.

a feasible approximation. However, some other methods, such as direct control allocation or fixed-point method, are able to find the exact, unique, feasible solution for all attainable $\underline{\tau}_d^{HT} \in \underline{\Phi}^{HT}$.

6.3.3. Approximation of unfeasible solution

In the literature, two common approximations are used to the approximate unfeasible pseudo-inverse solution $\underline{\mathbf{u}}^{HT}$: T-approximation (truncation) and S-approximation (scaling). In case of T-approximation, the approximation $\underline{\mathbf{u}}_t^{*HT} \in \partial(\underline{\Omega}^{HT})$ is obtained from $\underline{\mathbf{u}}^{HT}$ by truncating (clipping) all controls which exceed their control constraints. In contrast, the S-approximation $\underline{\mathbf{u}}_s^{*HT}$ is obtained by scaling unfeasible solution $\underline{\mathbf{u}}^{HT}$ to the boundary $\underline{\mathbf{u}}_s^{*HT} \in \partial(\underline{\Omega}^{HT})$ by factor f:

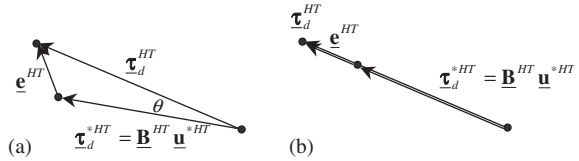

Fig. 17. Approximation error.

$$\underline{\mathbf{u}}_s^{*HT} = f\underline{\mathbf{u}}^{HT}, \tag{53}$$

where

$$f = \min\left(\frac{1}{\max_{i\in\{1,2,3,4\}}\left|\frac{u_i^{HT}}{s_i^{HT}}\right|}, 1\right). \tag{54}$$

The approximation error is defined as $\underline{\mathbf{e}}^{HT} = \underline{\tau}_d^{HT} - \underline{\tau}_d^{*HT}$, where $\underline{\tau}_d^{*HT} = \underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{*HT}$ (see Fig. 17). In order to be able to compare different approximations, two scalar errors are introduced: *direction error* $\theta = \operatorname{acos} \underline{\tau}_d^{HT} \cdot \underline{\tau}_d^{*HT} / \|\underline{\tau}_d^{HT}\|_2 \|\underline{\tau}_d^{*HT}\|_2$ and *magnitude error* $\|\underline{\mathbf{e}}^{HT}\|_2 = \|\underline{\tau}_d^{HT} - \underline{\tau}_d^{*HT}\|_2$. The direction error represents the angle between $\underline{\tau}_d^{HT}$ and $\underline{\tau}_d^{*HT}$, while the magnitude error represents the module of the approximation error vector $\underline{\mathbf{e}}^{HT}$ (Fig. 17). In the case when $\theta = 0$, the approximation $\underline{\tau}_d^{*HT}$ preserves the direction of the original vector $\underline{\tau}_d^{HT}$.

6.3.4. Example (Approximation of unfeasible pseudo-inverse solution)

Let $\underline{\tau}_d^{HT} = [0.70\ 0.20\ 0.25]^{\mathrm{T}}$ for X-shaped thruster configuration in fault-free case (Fig. 18). It can be seen that $\underline{\tau}_d^{HT} \in \underline{\Phi}^{HT} \backslash \underline{\Phi}_p^{HT}$. The weighted pseudo-inverse solution $\underline{\mathbf{u}}^{HT} = [1.15\ 0.25\ 0.65\ 0.75]^{\mathrm{T}}$ is obtained combining (36) and (49). This solution is unfeasible, since $u_1^{HT} > 1$.

The T-approximation is given by $\underline{\mathbf{u}}_t^{*HT} = [1.00\ 0.25\ 0.65\ 0.75]$ and this is feasible solution that lies on the boundary $\underline{\mathbf{u}}_t^{*HT} \in \partial(\underline{\Omega}^{HT})$. This solution leads to approximate $\underline{\tau}_{dt}^{*HT} = \underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}_t^{*HT} = [0.6625\ 0.1625\ 0.2125]^{\mathrm{T}}$ that lies outside $\underline{\Phi}_p^{HT}$ (see Fig. 18). The magnitude error is $\|\underline{\mathbf{e}}_t\|_2 = 0.0650$ and direction error is $\theta = 2.6362^\circ$.

The first step for the S-approximation is to use (54) to find the scaling factor $f = 0.8696$. Then the approximation $\underline{\mathbf{u}}_s^{*HT} = [1.0000\ 0.2174\ 0.5652\ 0.6522] \in \partial(\underline{\Omega}^{HT})$ is found using (53). This solution leads to an approximation $\underline{\tau}_{ds}^{*HT} = \underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}_s^{*HT} = [0.6087\ 0.1739\ 0.2174]^{\mathrm{T}}$ that lies on boundary $\partial(\underline{\Phi}_p^{HT})$ and represents intersection of $\underline{\tau}_d^{*HT}$ and $\partial(\underline{\Phi}_p^{HT})$ (see Fig. 18). The magnitude error is $\|\underline{\mathbf{e}}_s\|_2 = 0.1004$ and direction error is $\theta = 0$, i.e. the S-approximation $\underline{\tau}_{ds}^{*HT}$ preserves the direction of the original vector $\underline{\tau}_d^{HT}$. Comparing errors it can be seen that T-approximation leads to feasible approximation

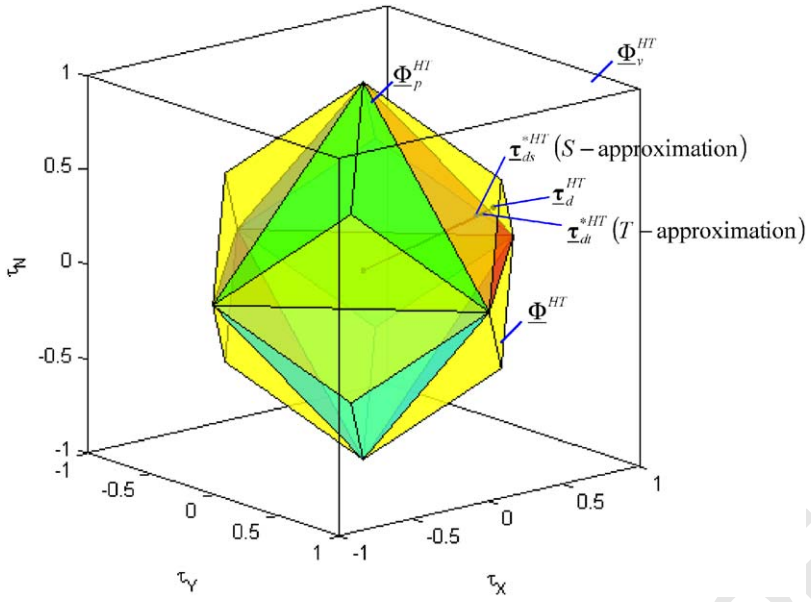

Fig. 18. Approximation of unfeasible pseudo-inverse solution.

with lower magnitude error, while *S*-approximation finds a feasible approximation that has the same direction as $\underline{\tau}_d^{HT}$, i.e. with direction error equal to zero.

6.3.5. FAS algorithm

The fault accommodation process involves these steps:

1. FDS detects thrusters' states and generates total fault indicator vector $\mathbf{f} = [\underbrace{f_1^{HT}\ f_2^{HT}\ f_3^{HT}\ f_4^{HT}}_{\mathbf{f}^{HT}}\ \underbrace{f_1^{VT}}_{\mathbf{f}^{VT}}]^{\mathrm{T}}$.
2. Using Table 6, "Co-ordinator" (Fig. 8) transforms vector $\mathbf{f}$ into the association vector $\mathbf{s} = [\underbrace{s_1^{HT}\ s_2^{HT}\ s_3^{HT}\ s_4^{HT}}_{\mathbf{s}^{HT}}\ \underbrace{s_1^{VT}}_{\mathbf{s}^{VT}}]^{\mathrm{T}}$, which relates thruster states and saturation bounds of each thruster. This transformation is called action B. For example, $\mathbf{f} = [2\ 1\ 1\ 1\ 1]^{\mathrm{T}}$ means "Jammed propeller" state in ${}^1HT$ and fault-free states in others. This vector is mapped to the association vector $\mathbf{s} = [0.75\ 1\ 1\ 1\ 1]^{\mathrm{T}}$ i.e. ${}^1HT$ should operate at 75%, while the others should work at full power (see Table 6).
3. If a faulty thruster is horizontal, then "Co-ordinator" updates its weight in the weighting matrix $\mathbf{W}_u^{HT}$ using (38) and (39), and set the other weights to unity. If a faulty thruster is vertical, then there is no need for weight update. The procedure of weighting update is called action A. New weights for the given example are $w_1^{HT} = 5/3$, $w_2^{HT} = w_3^{HT} = w_4^{HT} = w_1^{VT} = 1$.
4. "Pseudo-inverse" and "Approximation" find the new, feasible control vector for the given input and actual state of thrusters. "Pseudo-inverse" uses (47)–(49) to find the pseudo-inverse solution for motion in the horizontal plane, and (51)—for vertical plane. If the pseudo-inverse solution is unfeasible, "Approximation" block makes it feasible utilising *T*- or *S*-approximation.
5. Feasible control vectors for motion in the horizontal and vertical plane are merged, transformed into the form determined by the TCUs and used to actuate thrusters.

In the case of a fault in a single thruster, the feasible region for pseudo-inverse $\underline{\Phi}_p^{HT}$ and attainable command set $\underline{\Phi}^{HT}$ shrink, as shown in Fig. 19. In particular, Fig. 19a and c display the partitions of the virtual control space $\underline{\Phi}_v^{HT}$ (regions $\underline{\Phi}_p^{HT}$ and $\underline{\Phi}^{HT}$) for the case of a partial fault ("Heavy-jammed propeller") in ${}^2HT$. In this case, ${}^2HT$ is penalised by increasing its weight ($w_2^{HT} = 3$, Eqs. (38) and (39)) and corresponding saturation bound s_2^{HT} is changed to $s_2^{HT} = 0.5$. This guarantees equality between desired and actual vectors ($\underline{\tau}_d^{HT}$ and $\underline{\tau}^{HT}$) inside $\underline{\Phi}_p^{HT}$ shown in Fig. 19a and c and, at the same time, the value of $|\underline{u}_2^{HT}|$ will never be greater than 0.5. Similarly, Fig. 19b and d show the feasible region for the case of a total breakdown in ${}^2HT$. In this case, ${}^2HT$ is switched off and $w_2^{HT} \to \infty$. In this way, the redundancy is eliminated from the system of equation $\underline{\mathbf{B}}^{HT}\underline{\mathbf{u}}^{HT} = \underline{\tau}^{HT}$, which can now be solved in a standard way. From this reason $\underline{\Phi}_p^{HT}$ and $\underline{\Phi}^{HT}$ coincide in Fig. 19b and d. Only three remaining thrusters are capable to track the desired vector $\underline{\tau}_d^{HT}$ without any error inside $\underline{\Phi}_p^{HT}$ shown in Fig. 19b and d. This means that, in the case of a total failure in a single thruster, mission can be continued and the control allocation will be successful if the desired vector $\underline{\tau}_d^{HT}$ stays inside $\underline{\Phi}_p^{HT}$ for particular fault case.

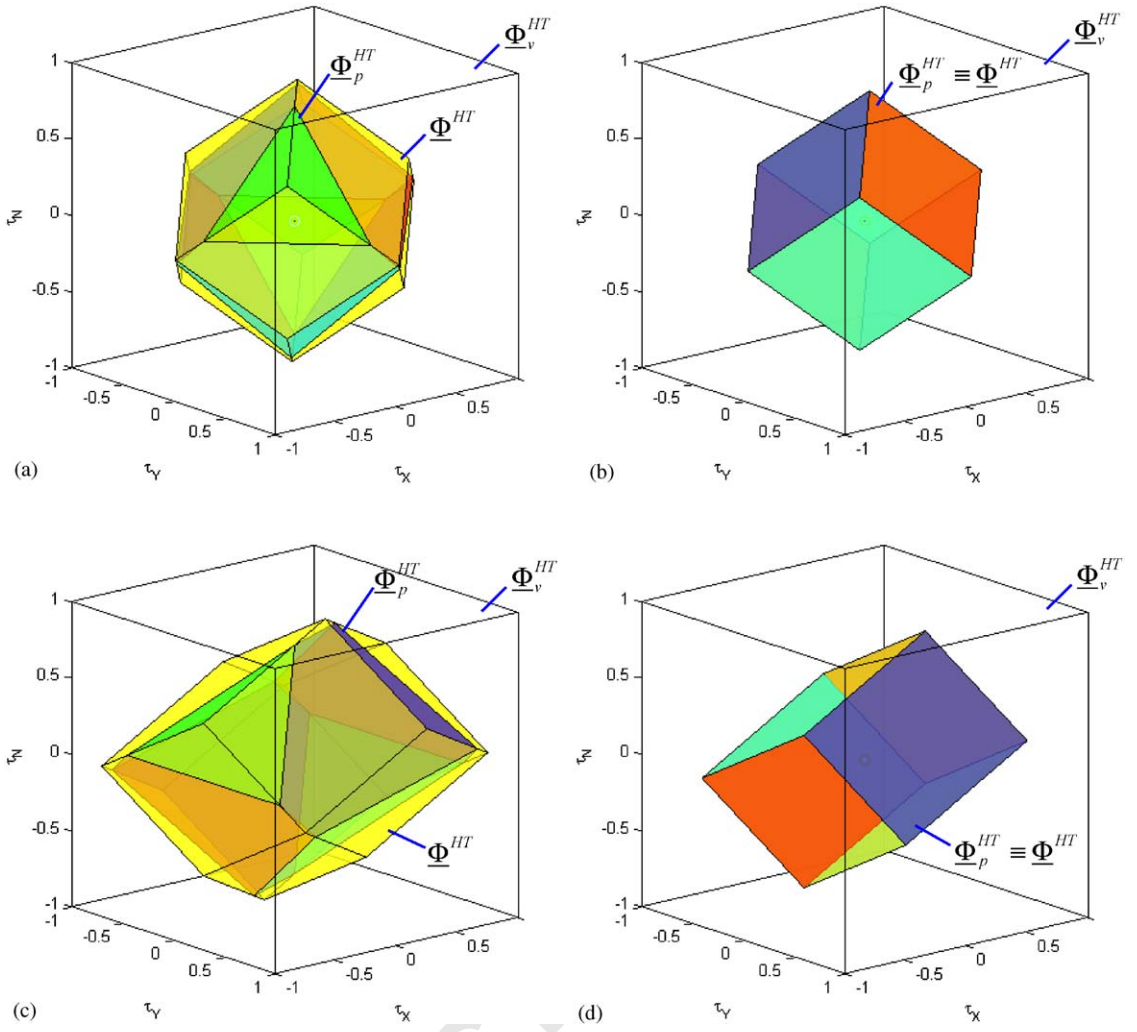

Fig. 19. Partitions of the virtual control space for different faulty situations: (a) FALCON: "Heavy j. propeller" ($s_2^{HT} = 0.5$). (b) FALCON: "Broken propeller" ($s_2^{HT} = 0$). (c) URIS: "Heavy j. propeller" ($s_2^{HT} = 0.5$). (d) URIS: "Broken propeller" ($s_2^{HT} = 0$).

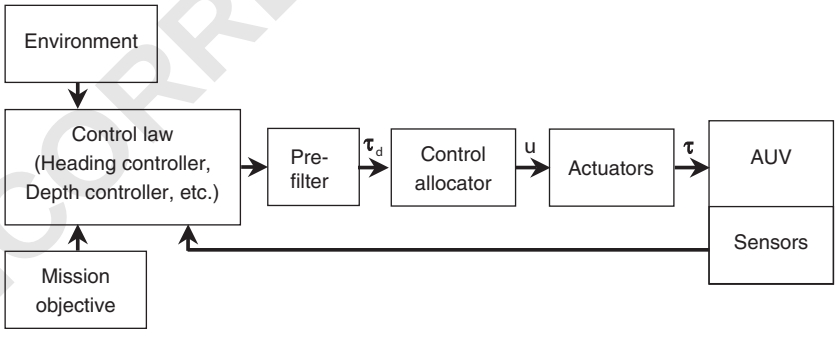

Fig. 20. Typical closed-loop AUV control structure.

7. Potential applications

In standard open-loop ROV control structure (Fig. 3a), vector τ_d is generated by an ROV pilot using HCU. In contrast, Fig. 20 displays a typical, closed-loop AUV control structure. In order to achieve mission objectives, the control law uses actual knowledge about the environment and sensor measurements to find a reference inputs for a set of controllers (heading controller, depth controller, etc.). The outputs of the controllers are integrated into a vector that is similar to the output of the HCU in Fig. 3a. The control allocator performs in exactly the same way as in the previous case. Hence, from the control allocator point of view, it does

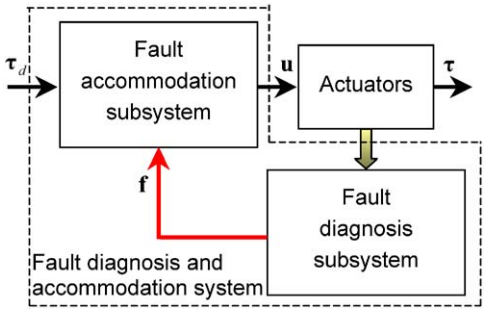

Fig. 21. Relationship between the FDAS and a typical control structure for open-frame underwater vehicles (ROVs or AUVs).

not matter how the virtual control input τ_d is generated (by the ROV pilot or the control law). The control allocation algorithm is the same for both structures. The task of the control allocator in both cases is to determine appropriate control settings for individual actuators, which produce the desired set of forces and moments.

It is useful at this point to consider the role (potential application) of the FDAS in the control structures shown in Figs. 3a and 20. As indicated in Fig. 21, the FDAS performs the control allocation task, but this primary task is enhanced with the ability to monitor the state of the thrusters and, if necessary, perform automatic reconfiguration, i.e. redistribution of propulsion forces among the operable thrusters, as explained in previous sections. In this way, using control allocation the actuator selection task is separated from the regulation task in the control design. That is, the control law, which maps the desired response to a set of commands (objectives), is not dependent on the design of the control allocation system, which relates these commands with settings and positions of individual actuators.

Treating control allocation independently of the control law is convenient because of the following:

- *Actuator constraints can be taken into account*: In real applications actuator saturation always exists. If one actuator saturates, some of methods for control allocation are able to redistribute control energy among other available actuators to compensate for the inability of a saturated thruster to produce its nominal control effect. In this way, available control resources are fully exploited before the closed-loop performance is degraded.
- *Reconfiguration can be performed*: If the effectiveness of the actuators change over time, or in the case of an actuator total or partial fault, reconfiguration i.e. redistribution of control energy among a set of available actuators can be performed, without having to redesign the control law.

8. Concluding remarks

A novel thruster fault detection and accommodation system for overactuated open-frame underwater vehicles has been presented. The FDAS includes two subsystems: FDS and FAS. The FDS is a hybrid, on-line, model-free approach, based on integration of SOM and fuzzy clustering methods. In the training phase, the FDS uses data obtained during test trial to find SOM representatives for each fault type. In the detection phase, the FDS makes decision about fault type by comparing the position of feature vector relative to these maps. The results demonstrate efficiency and robustness of the FDS. The FAS uses the output of the FDS to accommodate faults and perform reconfiguration by updating weights used in the optimisation criteria and thruster velocity saturation bounds. This paper demonstrates that, for open-frame underwater vehicles with four horizontal thrusters, in the case of a partial or total fault in a horizontal thruster, it is possible to reconfigure the control system in an optimal manner, in order to maintain a high level of manoeuvrability and mission completion.

Important contributions of the paper are:

- normalisation of the control allocation problem, which leads to easier understanding and visualisation of the problem,
- design of enhanced control allocator, able to reallocate control energy among operable thrusters in an optimal way and continue the mission in the presence of thruster velocity saturation and a fault in a single thruster.
- visualisation of the feasible region for pseudo-inverse, which provides a framework to visualise thruster velocity saturation bounds and to incorporate knowledge about saturation margins into control law. This is a very important enhancement, which improves existing controllers and provides better performance of the control system in real-world applications.

The most important issue is that every control law should generate input vectors that lie inside the feasible region for pseudo-inverse. Knowledge about position of the input vector inside the feasible region and its distance from the saturation bounds can be used to improve existing control laws and to avoid discrepancy between predicted and real behaviour of the vehicle caused by thruster velocity saturation. Currently, most ROVs' energy is provided, via the umbilical cable, from power supplies located on the mother ship. AUVs and some ROVs utilise batteries for the energy supply and excessive power consumption adversely affects available time on mission. The proposed FDAS provides an optimal solution for the distribution of propulsion

forces for fault-free case and fault in a single thruster, that minimises a control energy cost function. The importance of an optimal solution can be summarised as minimum control energy means maximum operational time from battery, and minimum usage of thruster means maximisation of thruster life.

However, the FDAS is able to find the exact solution of the control allocation problem only on a subset of attainable command set, i.e. in small zones around the feasible region, where the pseudo-inverse solution is unfeasible, still it is possible to find the exact solution by applying other techniques, such as direct control allocation or fixed-point methods. An extension of the FDAS to cover these zones is described in (Omerdic, Roberts, & Toal, 2004).

Underwater vehicles, which use other types of actuators (like control surfaces) beside thrusters, are not covered in this paper, but the same fault diagnosis and accommodation concept can be extended to cover this class of underwater vehicles by including new actuators into control architecture and reformulating the control allocation problem.

The proposed FDAS is a part of the low-level control layer and its modular design enables easy integration into existing control laws. Future work will include implementation of the proposed approach and its integration into existing control architecture for both vehicles. Special attention will be devoted to the design of a universal controller, robust to a partial/total fault in a single thruster. An important part of this work will be the integration of feasible region with real-time video presented to the ROV pilot from the on-board camera.

References

Alessandri, A., Caccia, M., & Veruggio, G. (1999). Fault detection of actuator faults in unmanned underwater vehicles. *Control Engineering Practice*, *7*, 357–368.

Blanke, M. (2001). Enhanced maritime safety through diagnosis and fault tolerant control. *IFAC conference on control applications and marine systems*, *CAMS 2001*, Glasgow (Invited plenary).

Blanke, M., Staroswiecki, M., & Eva Wu, N. (2001). Concepts and methods in fault-tolerant control. *Proceedings of the American control conference*, Washington (Invited tutorial).

Bono, R., Bruzzone, Ga., Bruzzone, Gi., & Caccia, M. (1999). ROV actuator fault diagnosis through servo-amplifiers' monitoring: An operational experience. *MTS/IEEE Oceans' 99*, Vol. 3 (pp. 1318–1324), Seattle, USA.

Bordignon, K. A. (1996). *Constrained control allocation for systems with redundant control effectors*. Ph.D. thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

Burken, J., Lu, P., & Wu, Z. (1999). *Reconfigurable flight control designs with applications to the X-33 vehicle*. NASA/TM-1999-206582.

Burken, J., Lu, P., Wu, Z., & Bahm, C. (2001). Two reconfigurable flight-control design methods: Robust servomechanism and control allocation. *Journal of Guidance, Control and Dynamics*, *24*(3), 482–493.

Cuadrado, A. A., Díaz, I., Díez, A. B., Obeso, F., & González, J. A. (2001). Fuzzy inference maps for condition monitoring with self-organising maps. *International conference in fuzzy logic and technology* (pp. 55–58), Leicester, UK.

Doty, K. L., Melchiorri, C., & Bonivento, C. (1993). A theory of generalised inverses applied to robotics. *The International Journal of Robotics Research*, *12*(1), 1–19.

Durham, W. C. (1993). Constrained control allocation. *Journal of Guidance, Control, and Dynamics*, *16*(4), 717–725.

Durham, W. C. (1994). Constrained control allocation: Three moment problem. *Journal of Guidance, Control, and Dynamics*, *17*(2), 330–336.

Enns, D. (1998). Control allocation approaches. In *AIAA guidance, navigation, and control conference and exhibit* (pp. 98–108), Boston.

Fossen, T. I. (1995). *Guidance and control of ocean vehicles*. Chichester: Wiley.

Fossen, T. I. (2002). *Marine control systems*. Trondheim, Norway: Marine Cybernetics AS.

Fossen, T. I., & Blanke, M. (2000). Nonlinear output feedback control of underwater vehicle propellers using feedback form estimated axial flow velocity. *IEEE Journal of Oceaning Engineering*, *25*(2), 241–255.

Ikeda, Y., & Hood, M. (2000). An application of l_1 optimisation to control allocation. In *AIAA guidance, navigation and control conference and exhibit*, Denver.

Isermann, R., & Ballé, P. (1997). Trends in the application of model-based fault detection and diagnosis of technical process. *Control Engineering Practice*, *5*(5), 709–719.

Johansen, T. A., Fossen, T. I., & Tøndel, P. (2002). Efficient optimal constrained control allocation via multi-parametric programming. *AIAA Journal of Guidance, Control and Dynamics*, submitted for publication.

Lindfors, I. (1993). Thrust allocation method for the dynamic positioning system. In *Proceedings of the 10th international ship control systems symposium* (*SCSS' 93*) (pp. 3.93–3.106), Otawa.

Omerdic, E., & Roberts, G. N. (2003). Thruster fault accommodation for underwater vehicles. *First IFAC workshop on guidance and control of underwater vehicles GCUV' 03* (pp. 221–226), 9–11 April 2003, Newport, South Wales, UK.

Omerdic, E., Roberts, G. N., & Ridao, P. (2003). Fault detection and accommodation for ROVs. *Sixth IFAC conference on manoeuvring and control of marine craft* (*MCMC 2003*), Girona, Spain.

Omerdic, E., Roberts, G. N., & Toal, D. (2004). Extension of feasible region of control allocation for open-frame underwater vehicles. *IFAC conference on control applications in marine systems* (*CAMS 2004*), Ancona, Italy, submitted for publication.

Podder, T. K., Antonelli, G., & Sarkar, N. (2000). Fault tolerant control of an autonomous underwater vehicle under thruster redundancy: Simulations and experiments. *Proceeding of the 2000 IEEE international conference on robotics & automation*, San Francisco, USA.

Podder, T. K., & Sarkar, N. (1999). Fault tolerant decomposition of thruster forces of an autonomous underwater vehicle. *Proceeding of the 1999 IEEE international conference on robotics & automation*, Detroit, USA.

Snell, S. A., Enns, D. F., & Garrard, W. L. (1992). Nonlinear inversion flight control for a supermaneuverable aircraft. *Journal of Guidance, Control and Dynamics*, *15*(4), 976–984.

Takai, M., & Ura, T. (1999). Development of a system to diagnose autonomous underwater vehicle. *International Journal of Systems Science*, *30*(9), 981–988.

Tøndel, P., Johansen, T. A., & Bemporad, A. (2001). An algorithm for multi-parameter quadratic programming and explicit MPC solutions. In *Proceedings of the IEEE conference on decision and control* (*CDC' 01*) (pp. TuP11-4).

Tøndel, P., Johansen, T. A., & Bemporad, A. (2003). Evaluation of piecewise affine control via binary search tree. *Automatika*, *39*, 743–749.

Virnig, J. C., & Bodden, D. S. (1994). Multivariable control allocation and control law conditioning when control effectors limit. In *AIAA guidance*, *navigation and control conference and exhibit*, Scottsdale.

Yang, K. C., Yuh, J., & Choi, S. K. (1998). Experimental study of fault-tolerant system design for underwater robots. *Proceeding of the 1998 IEEE international conference on robotics & automation*, Leuven, Belgium.

Yang, K. C. H., Yuh, J., & Choi, S. K. (1999). Fault-tolerant system design of an autonomous underwater vehicle—ODIN: An experimental study. *International Journal of Systems Science*, *30*(9), 1011–1019.