A Fault-tolerable Control Scheme for an Open-frame Underwater Vehicle

Regular Paper

Huang Hai1,*, Wan Lei1, Chang Wen-tian1, Pang Yong-jie1 and Jiang Shu-qiang2

1 National Key Laboratory of Science and Technology on Underwater Vehicle, Harbin Engineering University, Harbin, China
2 College of Automation, Harbin Engineering University, Harbin, China
* Corresponding author E-mail: haihus@163.com

Received 08 Oct 2012; Accepted 07 Apr 2014

DOI: 10.5772/58578

© 2014 The Author(s). Licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Open-frame is one of the major types of structures of Remote Operated Vehicles (ROV) because it is easy to place sensors and operations equipment onboard. Firstly, this paper designed a petri-based recurrent neural network (PRFNN) to improve the robustness with response to nonlinear characteristics and strong disturbance of an open-frame underwater vehicle. A threshold has been set in the third layer to reduce the amount of calculations and regulate the training process. The whole network convergence is guaranteed with the selection of learning rate parameters. Secondly, a fault tolerance control (FTC) scheme is established with the optimal allocation of thrust. Infinity-norm optimization has been combined with 2-norm optimization to construct a bi-criteria primal-dual neural network FTC scheme. In the experiments and simulation, PRFNN outperformed fuzzy neural networks in motion control, while bi-criteria optimization outperformed 2-norm optimization in FTC, which demonstrates that the FTC controller can improve computational efficiency, reduce control errors, and implement fault tolerable thrust allocation.

Keywords Open-frame Underwater Vehicle, Recurrent Neural Network, Bi-criteria Control, Fault Tolerance Control

1. Introduction

As a cost-effective solution for performing complex tasks in an underwater environment, underwater vehicles are attracting significant interest. However, the problem of controlling them is particularly challenging, since they are expected to operate in uncertain underwater environments. In such a context, fault tolerable control (FTC) is crucial for overcoming the possible occurrence of faults in the vehicle’s thrusters [1].

Currently, there are two major approaches of FTC for underwater vehicles, namely passive FTC and active FTC [2]. The passive approach relies mainly on systematic reliability and adaptability without fault detection [3,4]. The active approach either applies a compensation method on the basis of fault detection results or reconstructs a new control law [5]. For ROV manoeuvrability in complicated environments, thrusters are usually redundantly arranged. These are operated under an effective control scheme or by hand manipulation at the beginning. Once the thruster fault has been detected, a novel switching approach or a tuning methodology will be applied.
Podder, K. et al. exploited the excess number of thrusters for propulsion force distribution to accommodate thruster faults [6]. Daqi Z. et al. used a neural network to accommodate faults and perform an appropriate control reallocation on the basis of the on-line fault identification [7]. However, they only took the total fault of the thruster into consideration, which was different from the actual fault situation of the thrusters.

On the other hand, based on self-organizing maps [8], the fault accommodation subsystem of reference [9] used a weighted pseudo inverse system to find a 2-norm optimal solution to the control allocation problem. The thruster fault is differentiated into jammed, heavy jammed and broken propellers. It is typically allowed to continue operating in the case of a partial fault [10]. However, a pseudo inverse solution may generate an unequal distribution of propulsion forces and cause a loss of maneuverability because it does not necessarily minimize the magnitudes of the individual thrusters [11,12]. Though infinity-norm optimization enables a better direct monitoring and control of the magnitude for thrust minimization [12], it may encounter a discontinuity problem because of the non-uniqueness of such a solution and the separation of two successive solution-sets [13].

For the automatic control of ROV motions, different approaches have been applied in the existing literature, for example, $H_{\infty}$ control [14], adaptive control [15], sliding mode control [16] and neural network control [17]. Because neural networks can approximate linear or nonlinear mapping through learning, they have been widely used in the study of underwater robot motion controllers to compensate for the effects of nonlinearity and system uncertainties [17-21]. Furthermore, fuzzy neural networks (FNN) have been employed for ROV due to their possibility of expressing human experience in an algorithmic manner [22-25]. However, it is very difficult for the forward neural network to reflect the time series influence between system input and output variables, the weights of the neural network would take a long time to converge, especially when the underwater vehicle operates under disturbances or the target trajectory undergoes an abrupt change. This will lead to a poor transition process and even uncontrolled output [26,27]. Recently, the recurrent fuzzy neural network (RFNN) has been extensively presented since it is superior in its dynamic response and information storing [28-30]. However, RFNN has a low learning efficiency, which makes it very difficult to meet the high-precision trajectory tracking control of underwater robots [27]. As a powerful tool for modelling, analysis, control and optimization, petri-net can select and train the most effective weight to improve its training and control efficiency when combined with FNN control [31].

This paper will construct a petri-based RFNN (PRFNN) controller for the motion control of open-frame underwater vehicles to improve computational efficiency, and combine infinity-norm optimization with 2-norm optimization in a primal-dual neural network to implement fault tolerance (bi-criteria FTC) in thrust allocation.

2. Design of PRFNN controller

2.1 Model of an Open-frame Underwater Vehicle

If we establish an absolute reference frame $E-x\xi y\eta z\zeta$ and relative reference frame $G-xyz$ (Figure 1) and neglect roll, a 5-DOF motion model can be obtained as:

$$m \begin{bmatrix} (\ddot{u} - vr + wq) - x_q(q^2 + r^2) + y_q^r + z_qq \dot{q} \end{bmatrix} = X$$
$$m \begin{bmatrix} (\ddot{v} + ur) - y_q^r + z_qgr + x_qr \dot{r} \end{bmatrix} = Y$$
$$m \begin{bmatrix} (\ddot{w} - uq) - z_q^r - x_qr + y_qr \dot{q} \end{bmatrix} = Z$$
$$m \begin{bmatrix} I_q \dot{q} + m \begin{bmatrix} x_qr (\ddot{u} + wq - vr) - x_qr (\ddot{w} - uq) \end{bmatrix} = M$$
$$I_r \dot{r} + m \begin{bmatrix} x_qr (\ddot{v} + ur) - y_qr (\ddot{u} - vr) \end{bmatrix} = N$$

where $m$ is the mass of the vehicle, $x_q, y_q, z_q$ are the gravity centres of the vehicle; $I_x, I_y, I_z$ are the moments of inertia in the associated area with axes $Gx, Gy, Gz$ respectively; $u, v, w, q, r$ are velocities and angular velocities of the 5 DOF respectively; $X, Y, Z, M, N$ are applied forces of the 5-DOF respectively. In this way, the hydrodynamic model is obtained:

$$M(\ddot{\xi}) + C(\dot{\xi}) + D(\xi) + g(\xi) = \tau$$

Figure 1. ROV Coordinate system

where $\xi = [u\ v\ w\ q\ r]^T$, $M$ is the $5 \times 5$ inertia matrix. $C(\dot{\xi})$ is the $5 \times 1$ vector of centrifugal and Coriolis terms, $D(\dot{\xi})$ is the resistance matrix, $g(\xi)$ is gravity, $\tau$ is the input force vector of the thrusters and torques from the controller.
2.2 PRFNN Controller

The PRFNN Controller is the 5-layer block diagram of the petri-net based RFNN (Figure 2) designed to handle the control and trajectory tracking problems for open frame underwater vehicles. It includes an input layer, a membership layer, a petri layer, a rule layer and an output layer. The feedback connection is implemented using recurrent feedback at the membership layer. The propagation function in each layer is issued as follows.

1. The first layer is the input layer. Its output node is expressed as:

   \[ y_i^{(1)} = x_i^{(1)}, \quad i = 1, 2, 3, \ldots, n \]

where \( x_i^{(1)} \) and \( y_i^{(1)} \) are the input and output of the ith node in this layer, and \( n \) is the number of input variables.

2. The second layer is the membership layer. Each node in this layer acts as a membership function. The input of this layer is expressed as:

   \[ x_i^{(2)}(n) = x_i^{(1)}(n) + \eta_p (n - 1) \theta_y \]

where \( \theta_y \) denotes the weight of the self-feedback loop, \( n \) is the number of iterations, \( \eta_p (n - 1) \) represents the last time of the output of the 2nd layer. The output of the 2nd layer is defined with Gaussian membership functions as:

   \[ y_j^{(2)} = \exp \left( -\frac{(x_i^{(2)} - m_j)^2}{\sigma_j^2} \right), \quad j = 1, 2, \ldots, n \]

where \( m_j \) and \( \sigma_j \) are the mean and standard deviation of the Gaussian function, respectively. \( n \) is the number of the linguistic variables associated with each input.

3. The third layer is the petri layer, the purpose of which is to produce a threshold by using competition laws to determine the training requirements:

   \[
   S_j = \begin{cases} 
   1 & y_j^{(2)} \geq T_H \\
   0 & y_j^{(2)} < T_H
   \end{cases}
   \]

where \( S_j \) is the switch and \( T_H \) is a dynamic threshold value, corresponding with errors to be determined in equation (9).

4. The fourth layer is the rule layer, which multiplies the input variables to produce the output:

   \[
   y_r^{(4)} = \begin{cases} 
   0 & T_H = 0 \\
   \prod_{j=1}^{n} W_{jr} y_j^{(2)} & T_H = 1
   \end{cases}
   \]

where \( W_{jr} \) is the weights between the third layer and the fourth layer, \( r = 1, 2, \ldots, n \) is the number of the total rule.
(5) The fifth layer is the output layer:
\[ y^{(5)}_o = \sum_{i=1}^{n} A^i_o y^{(4)}_i \]  
where \( A^i_o \) is the output action’s strength with respect to
the \( i \)-th rule.

2.3. Strategy for Online Learning

The core of the learning strategy is to recursively obtain a gradient vector so that each element can be defined as the derivative of an energy function. The strategy is generally referred to as the back propagation learning rule, because the gradient vector is calculated in direction opposition to the flow of the output of each node. Firstly, we define the energy function \( E \) as:
\[ E = \frac{1}{2}[\left( x_d - x \right)^2 + (\dot{x}_d - \dot{x})^2 + (p_d - p)^2 + (\dot{p}_d - \dot{p})^2 + (z_d - z)^2 + (\dot{z}_d - \dot{z})^2] \]
\[ = \frac{1}{2}(x_{dc}^2 + \dot{x}_{dc}^2 + e_{dp}^2 + e_{dp}^2 + e_{pz}^2 + \dot{e}_{pz}^2) \]
where \( x_d \) is the desired X-axes position, \( p_d \) is the desired heading, \( z_d \) is the desired diving depth. \( x \) is the real X-axes position, \( p \) is the real heading, \( z \) is the real diving depth, \( e_{dx} = x_d - x \) is the current X-axes position error, \( e_{dp} = p_d - p \) is the current heading error, \( e_{dz} = z_d - z \) is the current depth error, \( \dot{x}_d, \dot{x}, \dot{p}, \dot{p}_d, \dot{z}_d, \dot{z}, \dot{e}_d, \dot{e}_p \) and \( \dot{e}_z \) are their derivatives. \( T_H \) of equation (5) is defined with the following equation:
\[ T_H = \frac{\partial \exp(-\Delta E)}{1 + \exp(-\Delta E)} \]
where \( \partial \) and \( \delta \) are positive constants to coordinate \( T_H \) which will be larger if the error is smaller.

In the output layer, the error term to be propagated is:
\[ \beta^* = \left( \frac{\partial E}{\partial x} \frac{\partial e_x}{\partial y^{(5)}} + \frac{\partial E}{\partial p} \frac{\partial e_p}{\partial y^{(5)}} + \frac{\partial E}{\partial z} \frac{\partial e_z}{\partial y^{(5)}} + \right) e_{dx} + \frac{\partial \dot{e}_x}{\partial y^{(5)}} + \frac{\partial \ddot{e}_x}{\partial y^{(5)}} + \frac{\partial \dot{e}_p}{\partial y^{(5)}} + \frac{\partial \ddot{e}_p}{\partial y^{(5)}} + \frac{\partial \dot{e}_z}{\partial y^{(5)}} + \frac{\partial \ddot{e}_z}{\partial y^{(5)}} \]

The weight of the output layer is updated by the equation (10):
\[ \Delta A^o = -\mu_o \frac{\partial E}{\partial A^o} = -\mu_o \left[ \frac{\partial E}{\partial y^{(5)}} \frac{\partial y^{(5)}}{\partial A^o} \right] = \mu_o \beta^* A^o \]
where \( \mu_o \) is the learning-rate parameter of the connecting weights. The propagating error term can be calculated as:
\[ \beta^* = \left( \frac{\partial E}{\partial A^o} \right) \]

where \( r^o = (x^{(2)} - m_o)^2 \cdot A^o(n), m_o(n), \sigma_o(n) \) and \( \theta_o(n) \) are updated as \((\bullet)(n+1) = (\bullet)(n) + \Delta(\bullet)(n)\). Hence the update laws for \( m_o, \sigma_o, \theta_o \) are:
\[ \Delta m_o = -\mu_o \frac{\partial \Delta E}{\partial m_o} = -\mu_o \left[ \frac{\partial E}{\partial y^{(5)}} \frac{\partial y^{(5)}}{\partial m_o} \right] = \mu_o \beta^{(3)} \frac{2(x^{(2)} - m_o)^2}{(\sigma_o)^3} y^{(2)}(n-1) \]
\[ \Delta \sigma_o = -\mu_o \frac{\partial \Delta E}{\partial \sigma_o} = \mu_o \beta^{(3)} \frac{2(x^{(2)} - m_o)^2}{(\sigma_o)^5} \]
\[ \Delta \theta_o = -\mu_o \frac{\partial \Delta E}{\partial \theta_o} = -\mu_o \beta^{(3)} \frac{2(x^{(2)} - m_o)^2}{(\sigma_o)^5} y^{(2)}(n-1) \]
where \( \mu_m \) and \( \mu_\sigma \) are the learning-rate parameters for the mean and the standard deviation of the Gaussian function respectively, and \( \mu_\theta \) is the learning-rate parameter for the self-feedback loop. In order to improve the learning speed, \( x, p, z, \dot{x}, \dot{p} \) and \( \dot{z} \) can be approximated by their sign functions \( \text{sgn}(\cdot) \):
\[ \frac{\partial (\bullet)(n)}{\partial y^{(5)}(n)} \approx \text{sgn} \left( \frac{\partial (\bullet)(n)}{\partial y^{(5)}(n-1)} \right) \]
2.4. Analysis for Convergence

The values selected for the learning-rate parameters have significant effects on the network’s performance. In the following section, the learning-rate parameters will be analysed in relation to the convergence of the network.

The discrete-type of (8) is:

\[ E(n+1) = E(n) + \Delta E(n) \]

\[ = E(n) + \frac{1}{4} \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \alpha_i} \Delta \alpha_i \right]^2 + \frac{1}{4} \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \sigma_i} \Delta \sigma_i \right]^2 + \frac{1}{4} \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \theta_i} \Delta \theta_i \right]^2 \]

\[ = E(n) - \mu \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \alpha_i} \Delta \alpha_i \right]^2 + \frac{1}{4} E(n) \]

\[ = E(n) - \mu \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \sigma_i} \Delta \sigma_i \right]^2 + \frac{1}{4} E(n) \]

\[ = E(n) - \mu \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \theta_i} \Delta \theta_i \right]^2 + \frac{1}{4} E(n) \]

Thus \( \mu_\alpha, \mu_\sigma, \mu_\sigma \) and \( \mu_\theta \) are set as:

where \( \varepsilon \) is a positive constant. Thus (14) can be written as:

\[ E(n+1) = E(n) \left[ \frac{\mu_\alpha + \mu_\sigma + \mu_\sigma + \mu_\theta}{4} \right] \]

\[ < E(n) \left[ \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} \right] = E(n) \]

According to (8) and (14), the tracking errors for the proposed petri-net-based dynamic RFNN controller will converge to zero gradually.

### Table 1. Values of learning rate parameters

<table>
<thead>
<tr>
<th>Learning rate</th>
<th>Parameters values</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mu_\alpha )</td>
<td>[ E(n) + \frac{1}{4} \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \alpha_i} \Delta \alpha_i \right]^2 + \varepsilon ]</td>
</tr>
<tr>
<td>( \mu_\sigma )</td>
<td>[ E(n) + \frac{1}{4} \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \sigma_i} \Delta \sigma_i \right]^2 + \varepsilon ]</td>
</tr>
<tr>
<td>( \mu_\theta )</td>
<td>[ E(n) + \frac{1}{4} \sum_{i=1}^{N} \left[ \frac{\partial E(n)}{\partial \theta_i} \Delta \theta_i \right]^2 + \varepsilon ]</td>
</tr>
</tbody>
</table>

### Table 2. The position for the thruster configurations

<table>
<thead>
<tr>
<th></th>
<th>1HT</th>
<th>2HT</th>
<th>3HT</th>
<th>4HT</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-shaped configuration</td>
<td>[ \frac{L}{2} ]</td>
<td>[ \frac{L}{2} ]</td>
<td>[ \frac{L}{2} ]</td>
<td>[ \frac{L}{2} ]</td>
</tr>
<tr>
<td>Cross-shaped configuration</td>
<td>[ \frac{W}{2} ]</td>
<td>[ \frac{W}{2} ]</td>
<td>[ \frac{W}{2} ]</td>
<td>[ \frac{W}{2} ]</td>
</tr>
</tbody>
</table>

### Table 3. The orientation for the thruster configurations

<table>
<thead>
<tr>
<th></th>
<th>e1</th>
<th>e2</th>
<th>e3</th>
<th>e4</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-shaped configuration</td>
<td>[\cos \beta, \sin \beta]</td>
<td>[\cos \beta, -\sin \beta]</td>
<td>[\cos \beta, \sin \beta]</td>
<td>[\cos \beta, -\sin \beta]</td>
</tr>
<tr>
<td>Cross-shaped configuration</td>
<td>[1, 0]</td>
<td>[1, 0]</td>
<td>[1, 0]</td>
<td>[1, 0]</td>
</tr>
</tbody>
</table>

3. Fault Tolerable Thrust Allocation

3.1 Model of Thrust Control Allocation

The underwater vehicle has four horizontal thrusters, denoted HTi, i=1,2,3,4. There are two common configurations of the horizontal thrusters, e.g., X-shaped and Cross-shaped configurations (Figure 3) [10]. Their position and orientation are illustrated in Table 2 and Table 3.

![Figure 3. Two configurations of the horizontal thrusters](image)

Therefore, the vectors of forces and moments, exerted by horizontal thrusters, e.g., X-shaped and Cross-shaped configurations, can be written as:

\[ \tau = \tau \begin{bmatrix} \dot{F} \, \vec{x} \times \vec{e} \varepsilon_i \\ \dot{M} \end{bmatrix} \]

\[ = \begin{bmatrix} e_x & e_y & (\vec{r} \times \vec{e} \varepsilon_i) \, \vec{r} \times \vec{e} \varepsilon_i \end{bmatrix} \begin{bmatrix} \dot{F} \end{bmatrix} \]
where \( \vec{F} \) is the force exerted by thruster \( i \) on the ROV, \( \vec{M} = \vec{F} (\vec{e} \times \vec{e}) \) is the moment deduced from \( \vec{F} \). The direction of \( \vec{F} \) is decided by \( \vec{e}_i = \left[ e_{ix}, e_{iy}, e_{iz} \right] \). We obtain:

\[
\vec{F}^i_{\text{tr}} = \sum_{i=1}^{4} \vec{F}^i_{\text{tr}} = \sum_{i=1}^{4} \left[ \vec{F}^i \right] \tag{15}
\]

\[
\begin{bmatrix}
  e_{ix} & e_{iy} & \ldots & e_{iz} \\
  e_{ix} & e_{iy} & \ldots & e_{iz} \\
  (\vec{r} \times \vec{e}_i)_x & (\vec{r} \times \vec{e}_i)_y & \ldots & (\vec{r} \times \vec{e}_i)_z \\
  (\vec{r} \times \vec{e}_i)_x & (\vec{r} \times \vec{e}_i)_y & \ldots & (\vec{r} \times \vec{e}_i)_z \\
\end{bmatrix}
= \begin{bmatrix}
  \vec{F} \\
  \vec{F} \\
  \vec{F} \\
  \vec{F} \\
\end{bmatrix} = \begin{bmatrix}
  \vec{F} \\
  \vec{F} \\
  \vec{F} \\
  \vec{F} \\
\end{bmatrix}
\]

where \( \vec{R} \in \mathbb{R}^{4 \times 4} \) is the thrusters' configuration matrix.

**3.2 Fault Tolerant Scheme for the Allocation of Thrusters**

There are two faulty situations for thrusters: partial fault and total fault. Given this, we use a weighting matrix \( W = \text{diag}(W_1, W_2, \ldots, W_4) \) to describe the thrusters' situations. The diagonal elements of the weighting matrix demands that the thrust cannot exceed the available capacity of the faulty thrusters:

\[
\begin{cases}
  W_i = 1 & \text{represents thrusters being free of fault} \\
  0 < W_i < 1 & \text{represents thrusters having partial fault} \\
  W_i = 0 & \text{represents thrusters having total fault}
\end{cases}
\]

If we set \( \bar{F} = \begin{bmatrix} F_x & F_y & F_z \end{bmatrix} \), then \( \bar{F} = W \bar{F} \) is obtained with a fault tolerant property, and the same for \( \bar{F} = \begin{bmatrix} F_x & F_y & F_z \end{bmatrix} \) versus \( \bar{F} = W \bar{F} \), when \( \bar{F} = W \bar{F} \), \( \bar{F} > W \bar{F} \) or \( \bar{F} < W \bar{F} \), \( \bar{F} = W \bar{F} \), respectively.

Thus, the 2-norm of the thrusters' force manifold is

\[
\| \bar{F} \| = \frac{\bar{F}^T \bar{F}}{2}
\]

where the infinity norm of the thrusters' force manifold is defined as:

\[
\| \bar{F} \| = W \max \left\{ \| \vec{F} \|, \| \vec{F} \|, \| \vec{F} \|, \| \vec{F} \| \right\} = 1
\]

Therefore, the thrust allocation problem can be addressed as the following constrained optimization problem:

\[
\begin{align*}
& \text{Minimize} & & a \| \bar{F} \| + \frac{1}{2} (1 - a) \| \bar{F} \| \\
& \text{subject to} & & \bar{F}^T \bar{F} = \bar{R}^T \bar{F}, \bar{F} W_i \leq \bar{F} \leq \bar{F} W_i
\end{align*}
\]

where \( a \in (0,1) \) is a weighting factor. This problem can also be written in a matrix form:

\[
\begin{align*}
& \text{Minimize} & & \frac{S^T Q S}{2} \\
& \text{subject to} & & B_0 S = \bar{d}_i \\
& & & B_1 S \leq \bar{d}_i, \quad \bar{S} \leq \bar{S}^* 
\end{align*}
\]

where \( \bar{S} = \begin{bmatrix} \bar{F} \end{bmatrix} \) is the weighting factor. This problem can be solved with a convex due to the positive definition of \( Q \).

(1) When \( 0 < a < 1 \), the objective function (18) is strictly convex due to the positive definition of \( Q \). Provided that the feasible region of linear constraints (19) is a closed convex set, the constrained optimal solution to the optimization problem (17) is unique [32]. In light of its uniqueness, the continuity of the solution could thus be guaranteed [13,33].

(2) When \( a = 1 \), the proposed scheme is reduced to the standard 2-norm scheme. The uniqueness and continuity of such schemes are both guaranteed.

(3) When \( a = 0 \), the proposed scheme is reduced to the standard infinity-norm scheme. According to [13], the infinity-norm scheme may have non-unique solutions at successive time instants, which may result in a discontinuity phenomenon. So, we set \( 0 < a \leq 1 \) to remedy such a discontinuity problem.

If we define \( u \in \mathbb{R}^4 \), corresponding to the equality constraints of (19), \( v \in \mathbb{R}^3 \), corresponding to the inequality constraints of (20), the primal-dual decision variable vector, \( g \) and its bounds \( g^+ \), could be constituted as:

\[
\begin{bmatrix}
  g \\
  v
\end{bmatrix} = \begin{bmatrix}
  S & \bar{S}^* & 1 \sigma & 1 \sigma \\
  -I & -1 \sigma & 0
\end{bmatrix} \in \mathbb{R}^{12 \times 4}
\]

where \( g \) and \( g^+ \) are the primal and dual decision variables, respectively.
Theorem 1

There exists at least one optimal solution, \( S^* \in \mathbb{R}^5 \), so that the dual problem (18)-(21) can be converted into the linear variation inequalities (LVI) problem:

\[
(g - g^*)^T (Hg + P) \geq 0
\]

(23)

where \( H = \begin{bmatrix} Q & -B_2^T & B_2^T \\ B_2 & 0 & 0 \\ B_1 & 0 & 0 \end{bmatrix} \) and \( P = \begin{bmatrix} 0 \\ -\bar{d}_1 \end{bmatrix} \).

Proof

It follows from [31] that the dual problem of (18)–(21) can be derived as:

\[
\text{Maximize } -\frac{\bar{S}}{2} Q \bar{S} + \bar{d}_2^T u - \bar{d}_1^T v + \bar{S}^T \chi - \bar{S}^T \chi^* \\
\text{subject to } Q \bar{S} - B_1 u + B_2 v - \chi + \chi^* = 0
\]

(24)

(25)

where \( \chi \in \mathbb{R}^4 \) are the corresponding dual decision variables, \( \chi^* \geq 0 \). Then, a necessary and sufficient condition for the optimum value \( (\bar{S}^*, \bar{g}, \bar{v}, \bar{v}^*) \) of the primal problem (18)-(21) and its dual problem (24)-(25) are the following:

primal feasibilities:

\[
B_2 \bar{S} = \bar{d}_2
\]

(27)

\[
B_1 \bar{S} \leq \bar{d}_1
\]

(28)

\[
\bar{S} \leq \bar{S}^* \leq \bar{S}^+
\]

(29)

and complementarities [34]:

\[
\bar{v}^T (-B_1 \bar{S}^* + \bar{d}_1) = 0
\]

(30)

\[
\bar{\chi}^T (-\bar{S}^* + \bar{S}^+) = 0
\]

(31)

\[
\bar{\chi}^* T (-\bar{S}^* + \bar{S}^+)= 0
\]

(32)

To correspond with the constraints of (21) and to simplify the above necessary and sufficient formulation, constraints

\[
\begin{cases}
S^*_i = S_i^+ & \text{if } \chi^*_i > 0, \chi^* = 0 \\
S^-_i < S^*_i < S_i^+ & \text{if } \chi^*_i = 0, \chi^* = 0 \\
S^*_i = S^-_i & \text{if } \chi^*_i = 0, \chi^* > 0
\end{cases}
\]

(33)

are obtained from (31) and (32).

If we define \( \chi^* = \chi^+ - \chi^- \), the dual feasibility constraint (25) will be transformed into

\[
Q \bar{S}^* - B_1^T u + B_2^T v^* = \bar{\chi}^*
\]

(34)

which is equal to the following linear variational inequality [35]:

\[
(\bar{S} - \bar{S}^*)^T (Q \bar{S}^* - B_1^T u^* + B_2^T v^*) \geq 0
\]

(35)

Similarly, we obtain: \( (\bar{v} - v^*)^T (-B_1 \bar{S}^* + \bar{d}_1) \geq 0 \) and \( (u - u^*)^T (B_2 \bar{S}^* - \bar{d}_1) \geq 0 \), where \( v^* \in \{ v \geq 0 \} \), \( u \in \mathbb{R}^4 \). Therefore:

\[
\begin{pmatrix}
\bar{S} \\
\bar{u} \\
\bar{v}
\end{pmatrix}
\begin{pmatrix}
\frac{Q}{2} & -B_2^T & B_2^T \\
B_2 & 0 & 0 \\
0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
\bar{S}^* \\
\bar{u}^* \\
\bar{v}^*
\end{pmatrix}
\begin{pmatrix}
0 \\
-\bar{d}_2 \\
\bar{d}_1
\end{pmatrix}
\geq 0
\]

(36)

According to [31,36], the LVI problem (23) is equivalent to the following system of piecewise-linear equations.

\[
K_{u} (g - (Hg + P)) - g = 0
\]

(37)

where \( K_{u} (\cdot) : R^{124+41} \rightarrow \Omega \) is a piecewise linear projection operator from space \( R^{124+41} \) onto set \( \Omega \) with the ith element of \( K_{u} (g) \) defined as:

\[
\begin{cases}
g_i & \text{if } g_i < g_i^- \\
g_i & \text{if } g_i \leq g_i^* \leq g_i^+ \quad \forall i \in \{1,\ldots,(124+41)\} \\
g_i^* & \text{if } g_i > g_i^*
\end{cases}
\]

(38)

Therefore, a primal-dual neural network solver could be developed as the following dynamic equation:

\[
\dot{g} = \gamma (I + H^T) K_{u} (g - (Hg + P)) - g
\]

(39)

where \( \gamma > 0 \) is a parameter with which to scale the network convergence, and this should be set as large as possible in the implementation.

Expression (36) can be further written as:

\[
\dot{g}_i = \sum_{j=1}^{30} c_i \left( K_{u} (\sum_{k=i}^{124} t_{k} g - P_i) - g_i \right)
\]

(40)

where \( c_i \) denotes the ith entry of the scaling matrix \( C = I + H^T \), and \( t_{k} \) denotes the ikth entry of the matrix, \( S = I - H \). On the basis of neuron expression (37), a more detailed architecture of the neural network is shown in Figure 5.
Theorem 2
With the existence of at least one optimal solution $\hat{S}$ to (18)-(21), starting from any initial state $\mathbf{g}(0) \in \mathbb{R}^{12+4i}$, the state vector $\mathbf{g}(t)$ of (13) converges to an equilibrium point $\mathbf{g}^*$, of which the first (4+1) elements constitute the optimal solution $\hat{S} \in \mathbb{R}^{4i}$. Moreover, if there exists a constant $\rho > 0$ so that
\[
(g^* - K_\alpha(g - (Hg + P)))^{T}(-Hg^* + P) \geq 0
\]then the exponential convergence can be achieved for (13) with a rate proportional to $\gamma \rho$.

Proof
First, it follows from the projection inequality:
\[
(g^* - K_\alpha(g - (Hg + P)))^{T} \times \left( (Hg + P) - g + K_\alpha(g - (Hg + P)) \right) \geq 0
\]Second, it follows from the projection-equation formulation of the linear variational inequalities (34):
\[
(g^* - g - K_\alpha(g - (Hg + P)))^{T} \times (H(g - g^*) - g + K_\alpha(g - (Hg + P))) \geq 0
\]
Then extending (40) gives
\[
(g^* - g)^T[I + H^T](g - K_\alpha(g - (Hg + P))) \geq \\
\|g - K_\alpha(g - (Hg + P))\| + \|g - g^*\|H(g - g^*)
\]
Noting that $H$ is positive semi-definite in terms of:
\[
g^T H = g^T[I + H^T] \geq \frac{1}{2} \|Q\| \leq \|g - g^*\|
\]
We have
\[
(g^* - g)^T[I + H^T](g - K_\alpha(g - (Hg + P))) \geq \\
\|g - K_\alpha(g - (Hg + P))\| + \|g - g^*\|H
\]
Define the Lyapunov function $L(g) = \|g - g^*\|^2 \geq 0$. Its time derivative along the primal-dual neural network (36) is
\[
\frac{dL(g)}{dt} = \frac{dL(g)}{dt} \times \frac{dy}{dt} = \\
(g^* - g)\gamma(I + H^T)(K_\alpha(g - (Hg + P)) - g)
\]
Using the Lyapunov theory, the network state $\mathbf{g}(t)$ is stable and globally convergent to an equilibrium point $\mathbf{g}^*$ in view of $\hat{L} = 0$, $\hat{g} = 0$ and $\mathbf{g} = \mathbf{g}^*$. According to Theorem 1 and Eq. (34), $\mathbf{g}^*$ is the solution to the linear variational inequality problem (23) and the first five elements of $\mathbf{g}^*$ constitute the optimal solution $\hat{S}$ to the primal problem (18)-(21).

For the exponential convergence, if $\rho > 0$ , then
\[
\|g - K_\alpha(g - (Hg + P))\| \geq \rho \|g - g^*\| , \text{ we obtain}
\]
\[
\frac{dL(g)}{dt} \leq \gamma \|g - K_\alpha(g - (Hg + P))\|^2 + \|g - g^*\|^2 H
\]
\[
\leq \gamma \|g - g^*\|^2 \leq -\gamma \|g - g^*\|^2 H
\]
where $\rho \gamma > 0$ is the convergence rate. Thus, the exponential convergence condition can be justified in practice, by considering the equivalence $g - K_\alpha(g - (Hg + P)) = 0$ and $\mathbf{g} = \mathbf{g}^*$. Therefore, we
obtain \( L(g) = O(e^{-rt(t)}) \), \( \forall t \geq t_0 \), which completes the exponential convergence property of this primal-dual network.

4. Simulation and Experimental Results

In order to verify and analyse the FTC scheme of this paper, experiments and a simulation were performed on a SY-II Open-frame underwater vehicle. The control commands are sent through network communication between a surface computer and a PC/104 embedded system. SY-II is equipped with a depth gauge, a magnetic compass, six thrusters, including two main thrusters, two side thrusters and two vertical thrusters. The horizontal thrusters are cross configuration (Figure 2(b)). The hydrodynamic and inertial parameters are illustrated in Table 4 and Table 5. Experiments have been conducted in a 50×30×10 meter tank at the Key Laboratory of Science and Technology on Underwater Vehicles, Harbin Engineering University.

In these depth and heading control experiments, shown in Figure 6 and 7, we performed comparisons between the PRFNN and FNN control. The learning-rate parameter is obtained through tank experiments: \( \mu_a = 167.5, \mu_u = 0.15, \mu_d = 97.7, \mu_e = 14.8 \). The depth control results and a comparison of Figure 6 and 7 illustrate that the PRFNN controller has a higher convergence speed and fewer errors than FNN. Operating the heading control of SY-II is much harder because of the length-width ratio of the open-frame of the underwater vehicle and its strong non-linear and random disturbance from the tether. In the heading control experiments of Figure 7, we compared the systems in four situations: (1) PRFNN controller of section 2 without thrusters fault; (2) PRFNN controller with bi-criteria FTC \( (a = 0.5, \gamma = 10^5) \), where \( a \) has been selected from heading control simulations and experiments, for the two criteria of convergence speed and control errors, \( \gamma \) should be set as large as possible to allow for computer simulations, the left main thruster (1HT in Figure 3.(b)) 70% fault and tail side thruster (4HT in Figure 3.(b)) 100% fault; (3) PRFNN controller with 2-norm FTC \( (a = 1, \gamma = 10^5) \); the left main thruster (1HT in Figure 3.(b)) 70% fault and tail side thruster (4HT in Figure 3.(b)) 100% fault), which is similar to a pseudo inverse solution in [11] and [12]; (4) FNN controller

<table>
<thead>
<tr>
<th>mass(kg)</th>
<th>( I_x ) (Nms(^2))</th>
<th>( I_y ) (Nms(^2))</th>
<th>( I_z ) (Nms(^2))</th>
<th>( I_{xy} ) (Nms(^2))</th>
<th>( I_{xz} ) (Nms(^2))</th>
<th>( I_{yz} ) (Nms(^2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>111.9</td>
<td>97.3</td>
<td>26.1</td>
<td>56.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4. SY-II inertial parameters

<table>
<thead>
<tr>
<th>Dimensionless coefficient</th>
<th>( x )</th>
<th>( y )</th>
<th>( z )</th>
<th>( r )</th>
<th>( q )</th>
<th>( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td>First order terms</td>
<td>27.65</td>
<td>47.26</td>
<td>44.86</td>
<td>62.63</td>
<td>57.55</td>
<td>141.33</td>
</tr>
</tbody>
</table>

Table 5. SY-II Hydrodynamic parameters

![Figure 6. Comparisons of depth control experiments](image-url)
without any faulty thrusters. At the beginning of the heading control experiments, the acceleration for the heading control is great, and fluctuation results from the counter force of the tether and the self inertia. From the heading control results, RFNN works better in a nonlinear system dynamic response setting and the petri-net alleviated training inefficiency. Thus, the PRFNN controller exhibited a faster convergence speed in comparison with the FNN controller because it incorporated a petri-net into a conventional recurrent FNN framework. Thrusts are better optimized for allocation when faults have been detected. When an infinity-norm optimal solution is introduced through weighting factor $a$, the FTC is transformed into a bi-criteria FTC scheme. The bi-criteria FTC scheme behaves with better convergence results in comparison with the 2-norm FTC scheme during the heading control experiments, because infinity-norm optimization enables a better thrust magnitude optimization and a self continuity solution.

Cruising simulations and experiments have been carried out with the precondition that the left main thruster (1HT in Figure 3,(b)) has 40% fault and the tail side thruster (4HT in Figure 3,(b)) has 100% fault. In the cruising simulation under or without a current effect (eastward 0.2m/s) in Figure 8 (a) and (b), the desired motion trajectory is (0,0)-(70,200)-(0,300)-(120,300)-(120,200)-(0,0). Furthermore, cruising experiments have been conducted for a SY-II equipped with an Ultrasonic Doppler Velocity Meter in Figure 6 (a) and 8 (c), the desired motion trajectory is (-7,-20)-(-2,-2)-(5,-2)-(5,-22)-(-7,-20). From Figure (8), the bi-criteria FTC outperforms the 2-norm FTC for fault tolerance control in reducing overshoot during yawing, approaching the desired trajectory and cruising against the current.

**Figure 7.** Heading control experiments of fault tolerant optimal allocation of thrusters

**Figure 8.** Cruising experiments and simulation of fault tolerant optimal allocation of thrusters
5. Conclusions

The main contributions of this paper are two twofold:

Firstly, a PRFNN controller was designed. We have incorporated a petri-net into a conventional recurrent FNN framework in the third layer. A threshold was used in this layer to regulate training and learning, according to controller errors. The computational inefficiency was reduced and a faster convergence speed was obtained by using a petri-net layer. Thus, the amount of calculations performed by the whole network was reduced. Moreover, an online training algorithm was developed based on a gradient descent method. The selection of learning rate parameters guaranteed the whole network convergence.

Secondly, a bi-criteria neural network optimization scheme was presented to implement fault tolerant control for thrusters. When thruster faults are detected, the proposed bi-criteria neural network optimization scheme combines the 2-norm optimization scheme and the infinity-norm optimization scheme via a weighting factor. The fault and saturation limit of the thrusters are considered at the same time. A primal-dual neural network, based on linear variation inequalities, is developed to implement the optimal allocation of thrust.

Experiments and simulations have further demonstrated the performance, characteristics and efficacy of the proposed PRFNN, with a bi-criteria neural network optimization fault tolerant control scheme.

6. Acknowledgements

This project is supported by the National Science Foundation of China (No.51209050, 51179035, 51279221 and 61100006), the Doctoral Fund of the Ministry of Education for Young Scholars (20122304120003), the pre-research fund of state key of science and technology of underwater vehicle (9140C270306130C27102), the State Key Laboratory of Robotics and Systems (Harbin Institute of Technology) (No. SKLRS-2012- ZD-03).

7. References


