Ship docking and undocking control with adaptive-mutation beetle swarm prediction algorithm

Wang, Le; Li, Shijie; Liu, Jialun; Wu, Qing; Negenborn, Rudy R.

DOI
10.1016/j.oceaneng.2022.111021

Publication date
2022

Document Version
Final published version

Published in
Ocean Engineering

Ocean Engineering 251 (2022) 111021

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng

Ship docking and undocking control with adaptive-mutation beetle swarm prediction algorithm ☆

Le Wang [a], Shijie Li [a], Jialun Liu [b,c,*], Qing Wu [a], Rudy R. Negenborn [d,b]

[a] School of Transportation and Logistics Engineering, Wuhan University of Technology, Wuhan, 430063, PR China
[b] Intelligent Transportation Systems Research Center, Wuhan University of Technology, Wuhan, 430063, PR China
[c] National Engineering Research Center for Water Transport Safety, Wuhan, 430063, PR China
[d] Department of Maritime and Transport Technology, Delft University of Technology, Delft, 2628 CD, The Netherlands

ARTICLE INFO

Keywords:
Ship
Docking
Undocking
Intelligent optimization algorithm
Predictive control

ABSTRACT

Autonomous docking and undocking control is an important part of intelligent ship motion control. In this study, the adaptive-mutation beetle swarm prediction (AMBS-P) algorithm is used to propose a control approach for autonomous docking and undocking. Firstly, this paper introduces the principle of the AMBS-P algorithm, then the convergence is proved. Secondly, the "Tito-Neri" model ship is introduced as a case study, and the thrust allocation process is described. Finally, the effect of docking and undocking is verified in multiple scenarios starting from different angles. In the verification, first of all, when designing the docking and undocking controllers, the correctness of the algorithm and the practicality of the control are verified by whether there is ship drag or not. Secondly, by analyzing the parameters of the algorithm, the optimal parameters of it are determined and verified in the real environment. Thirdly, compared with typical proportion–integral–derivative (PID) algorithm and nonlinear model predictive control (NMPC) algorithm, the AMBS-P algorithm has better results for autonomous docking and undocking control, no matter in long-distance or short-distance. The research shows that the AMBS-P algorithm has a fast response and good effect for the ship autonomous docking and undocking, and does not rely too much on the system model.

1. Introduction

For a long time, large ships usually rely on tugboats or crew for docking, so the self-docking of ships is one of the research hot-spots of its motion control. Whether for tugboats or autonomous docking research, the commonly used algorithms are neural network (NN) (Shuai et al., 2019), proportion–integral–derivative (PID) (Bui et al., 2009), Fuzzy logic control (Liao et al., 2019), or the combination and improvement of these algorithms. Among them, PID is the most commonly used control algorithm, which is often combined with the NN algorithm for ship autonomous docking (Fang et al., 2010; Ahmed, 2012). Besides, it can resist the interference of high wind and waves. This good effect has been verified in the subsequent research from simulation to experiment (Ahmed and Hasegawa, 2015, 2013). The Fuzzy logic control algorithm can realize the adaptive control of the imprecise mathematical model, but it has a strong subjectivity. Therefore, in the detailed judgment stage, it needs to be combined learning strategies to make up for the shortcomings. NN theory can imitate the behavior of the human brain activity through training in the stage of the ship docking. So it is considered as one of the most effective theories to study autonomous docking (Namkyun, 2007; Tran and Im, 2012). However, there are too many iterations in the training process. NN algorithm is one of the metaheuristic optimization algorithms. Metaheuristic optimization algorithms include the following categories (Hashim et al., 2019): (1) nature-inspired algorithms (NIAs), including swarm-intelligence-based algorithms (SIs) and bio-inspired algorithms (BIAs); (2) natural science-based algorithms (NSAs); (3) natural phenomena-based algorithms (NPAs).

The nature-inspired algorithm taking this as an example has the following application research in the field of ship control. First of all, it often combines the nature-inspired algorithm with the basic algorithm, such as the combination of a NN algorithm and a PID controller. In addition, the combination of nature-inspired algorithms and a basic algorithm is also used to optimize the parameters of the ship controller. For example, Ant Colony Optimization (Tomera, 2014),

☆ Supported by National Natural Science Foundation of China (62003250), Southern Marine Science and Engineering Guangdong Laboratory (Zhuhai) (SML2021SP101).

* Corresponding author at: Intelligent Transportation Systems Research Center, Wuhan University of Technology, Wuhan, 430063, PR China.
E-mail address: jialunliu@whut.edu.cn (J. Liu).

https://doi.org/10.1016/j.oceaneng.2022.111021
Received 25 June 2020; Received in revised form 28 February 2022; Accepted 2 March 2022
Available online 30 March 2022

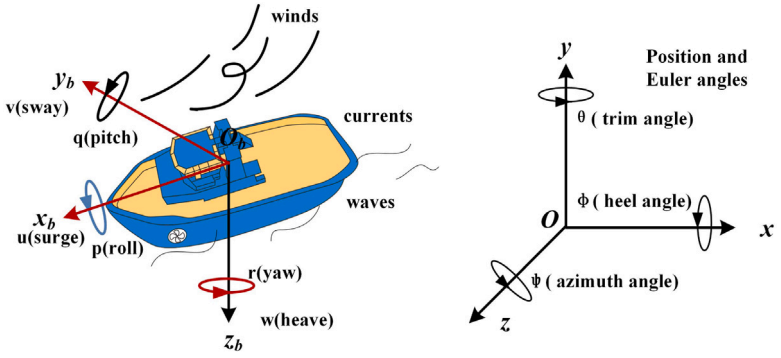


Fig. 1. Ship and coordinate systems.

Genetic Algorithm (Liu et al., 2017; Larrazabal and Peñas, 2016) and Bacterial Foraging Algorithm (Dong et al., 2019) are applied to modify the PID controller parameters to adapt to the different states of the ship. In addition to controller parameter modification for basic algorithm design, Fruit Fly Optimization Algorithm (Wang and Jian, 2012) and Particle Swarm Optimization (PSO) algorithm (Tomera, 2015; Shin et al., 2017) are applied to identification of ship maneuverability response model. Bacterial Foraging Algorithm and PSO can be combined to optimize ship collision avoidance path (Liu et al., 2016). Artificial Fish Swarm Algorithm (Li and Ma, 2016; Chen et al., 2018) is used to optimize the collision avoidance algorithm. However, some nature-inspired algorithms, such as Gray Wolf Optimizer, Krill Herd, Spider Monkey Optimization, Whale Optimization Algorithm, Ant Lion Optimization Algorithm, Lion Optimization Algorithm, have not been applied to the ship field.

To sum up, the nature-inspired algorithm with its advantages of intelligence provides new ideas and methods for solving various complex problems. It has also been successfully applied in the field of the ship. Based on this, this study considers whether the nature-inspired algorithm can be directly applied to the docking and undocking control of the ship. Among them, Beetle Antennae Search (BAS) is a relatively nature-inspired algorithm in 2017 (Jiang and Li, 2017). Now it is gradually applied in the field of ship control for its simple principle and fast optimization velocity outstanding features, such as PID controller parameters optimization (Wang et al., 2019). In this study, based on the BAS algorithm, the PSO swarm idea and mutation factor are introduced. Combined with the idea of a predictive control algorithm, the AMBS-P algorithm is designed to study the autonomous docking and undocking of ships.

The main contributions of this paper are as follows:

- A controller based on the AMBS-P algorithm is designed and its convergence is proved.
- Firstly, a motion controller without ship drag is designed to verify the correctness of the algorithm. Then, a controller with ship drag is designed to obtain the control effect in line with the real situation.
- The influence of the prediction idea in the auxiliary nature-inspired algorithm to realize the ship control is analyzed.
- The AMBS-P algorithm is suitable for ship autonomous docking and undocking control with a wide application prospect for other aspects of intelligent ships.

This paper is organized as follows: Section 2 describes the AMBS-P control algorithm. Section 3 introduces the ship model. Section 4 designs the ship docking and undocking motion controllers. In Section 5, the effect of controllers is verified and the result is analyzed. Section 6 summarizes the results and prospects the future research.

2. An overview of AMBS-P algorithm and its convergence proof

2.1. An overview of AMBS-P algorithm

BAS algorithm is a nature-inspired algorithm and developed based on the beetle foraging principle. This paper extends it from single to swarm to improve search efficiency. Besides, a mutation factor is introduced to reduce the probability of getting local optimum. Based on these, this algorithm is named AMBS-P algorithm. The specific design process of the algorithm is as follows.

Suppose there are n beetles $B = (B_1, B_2, \dots, B_n)$ in N dimensional space. The position of the ith beetle is described as $B_i = (b_{i1}, b_{i2}, \dots, b_{iN})$ and the velocity is expressed as $V_i = (v_{i1}, v_{i2}, \dots, v_{iN})$. The individual and global extreme points of beetles are respectively assumed as: $Pb_i = (Pb_{i1}, Pb_{i2}, \dots, Pb_{iN})$ and $Gb_i = (Gb_{i1}, Gb_{i2}, \dots, Gb_{iN})$.

So the velocity V_i and original position B_i of the beetle are as Eqs. (1a) and (1b).

$$V_i^k = \omega^k \cdot V_i^{k-1} + c_1 \cdot r_1 \cdot Pb_i^{k-1} - B_i^{k-1} + c_2 \cdot r_2 \cdot (Gb_i^{k-1} - B_i^{k-1}), \tag{1a}$$

$$B_i^k = B_i^{k-1} + V_i^k, \tag{1b}$$

Among them, k is the current number of iterations. The "Linearly Decreasing Weight (LDW)" is used to set ω, which makes adaptive adjustment according to the number of iterations, that is,

$$\omega^k = \omega_{min} + (\omega_{max} - \omega_{min}) \cdot (K - k)/K, \tag{2}$$

where, ω_{max} and ω_{min} are the maximum and minimum weight coefficients, K represents the maximum number of iterations.

The antennae of beetle is considered as two optional directions. The motion path can be further determined according to this. In this paper, B_L and B_R are defined as the coordinates of left and right antennae respectively, shown in Eq. (3). Besides, D_0 is initial antennae length.

$$\left.\begin{aligned} B_{Li}^k &= B_i^k + D_0^k \cdot \tilde{D}^k/2 \\ B_{Ri}^k &= B_i^k - D_0^k \cdot \tilde{D}^k/2 \end{aligned}\right\}. \tag{3}$$

Antennae orientation can be described as $Dir^k = rand(N, 1)$, and define $\tilde{D}^k$ as:

$$\tilde{D}^k = Dir^k / norm(Dir^k), \tag{4}$$

B_i represents the beetle centroid coordinates. The calculation formula of B_i is shown in Eq. (5), where L_{step} is beetle step length. Then the cost function value $F(B_i^k)$ is calculated according to B_i.

$$B_i^k = B_i^{k-1} - L_{step}^k \cdot \tilde{D}^k \cdot sign(F(B_{Li}^k) - F(B_{Ri}^k)). \tag{5}$$

The optimal cost function is calculated based on the B_i^k in each iteration cycle. Then the current Pb_i^k and F_{best} are compared and

(a) Structure-1 (b) Structure-2 (c) Ship model

Fig. 2. "Tito-Neri" ship model. ((a) and (b) are cited from Ref. Bruggink et al. (2018)).

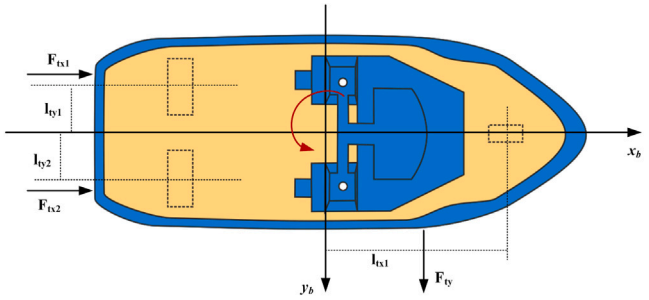


Fig. 3. Thruster structure of the ship.

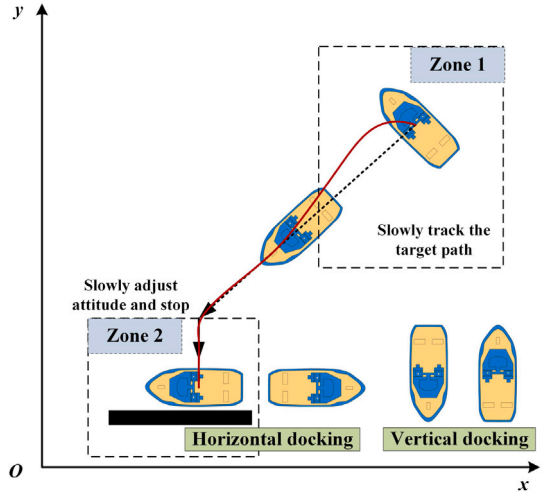


Fig. 4. Analysis of docking modes and process.

updated according to Eq. (6).

$$\left.\begin{aligned} F_{best}^k &= F(B_i^k) \\ Pb_i^k &= B_i^k \end{aligned}\right\}, F(B_i^k) < F_{best}. \tag{6}$$

Update L_{step} and D_0. L_{step} and D_0 require to be updated in time, and proportion control method is selected. The condition for L_{step} attenuation is that the optimal fitness value does not decrease. Because this means that the current L_{step} and D_0 can no longer obtain a better position. They require to be updated until the set optimal cost function value is met or the control is closed. On the contrary, only the B_i^k needs to be updated. Update rules are shown in Eq. (7).

$$\left.\begin{aligned} L_{step}^k &= bl_1 \cdot L_{step}^{k-1} \\ D_0^k &= L_{step}^{k-1}/bl_2 \end{aligned}\right\}, F(B_i^k) < F_{best}. \tag{7}$$

The prediction idea is introduced, and the total cost function is set as F.

$$minF(x|j) = \sum_{j=1}^{N_p} q_j \cdot F_{best}^j, \tag{8}$$

where, q is the weight factor, N_p is the number of prediction steps and F_{best} is the single step cost function value.

With the progress of iteration, the concentration degree of beetles increased, and the difference between them decreased gradually. In this case, on the one hand, the algorithm may find the global optimal value, on the other hand, it may fall into the local optimum. In this paper, we first calculate the variance of the average cost function and the spatial location aggregation degree of the beetle swarm, and then judge whether the threshold selection needs adaptive mutation to increase the diversity of beetle swarm, to further reduce the probability of the algorithm falling into the local optimum.

At the kth, $F_i^k = F(B_i^k)$ is the cost function, and F_{ave}^k is the average cost function. The fitness variance Θ^k is used to reflect the aggregation degree of beetle swarm, as shown in Eq. (9).

$$\Theta^k = \sum_{i=1}^{n} \left(\frac{F_i^k - F_{avg}^k}{\tilde{F}^k} \right)^2, \tag{9}$$

where, $\tilde{F}^k = max\{max\{|F_i^k - F_{avg}^k|\}, 1\}, i \in [1, n]$.

Set the Φ^k to indicate the spatial location aggregation degree between beetle individuals, as described in Eq. (10). The smaller Φ^k is, the more concentrated the beetle is, the higher the probability of mutation is.

$$\Phi^k = \frac{max_{1\le i\le n}\left\{\|B_i^k - Pb_i^k\|\right\}}{max_k\left\{max_{1\le i\le n}\left\{\|B_i^k - Pb_i^k\|\right\}\right\}}. \tag{10}$$

Define mutation probability p_m^k as Eq. (11). The algorithm may fall into local convergence when Θ^k is less than the set threshold and Φ^k decreases, and Eq. (11) needs to be executed.

$$p_m^k = \begin{cases} e^{-\Phi^k}/5, & \Theta^k < \Theta_0 \\ 0, & \Theta^k \ge \Theta_0 \end{cases}. \tag{11}$$

Θ_0 is a threshold set according to the actual research object. Finally, further judgment is made according to this probability, and then the individual extreme value of each beetle is adjusted, such as Eq. (12). Among them, p_{rand} is a random number, and η is a random vector obeying the standard normal distribution.

$$Pb_i^{k+1} = \begin{cases} Pb_i^k \cdot (1 + 0.5\eta), & p_{rand}^k < p_m^k \\ Pb_i^k, & p_{rand}^k \ge p_m^k \end{cases}. \tag{12}$$

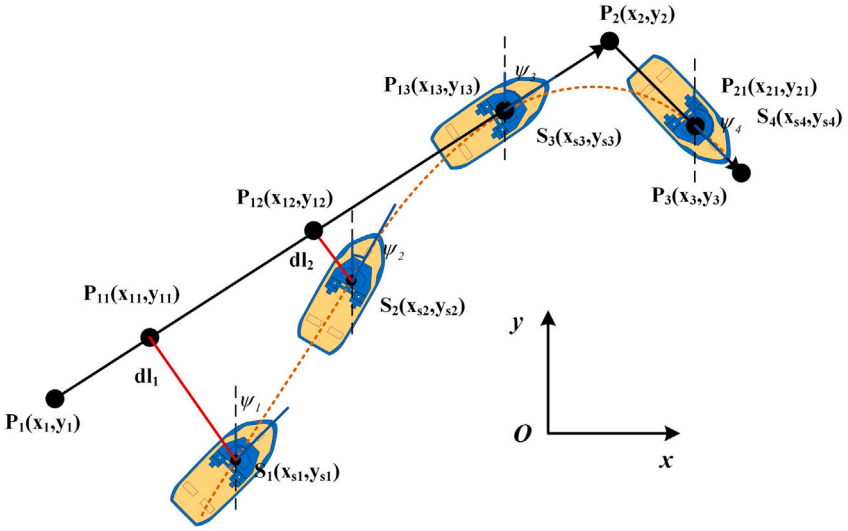


Fig. 5. Schematic diagram of ship trajectory control.

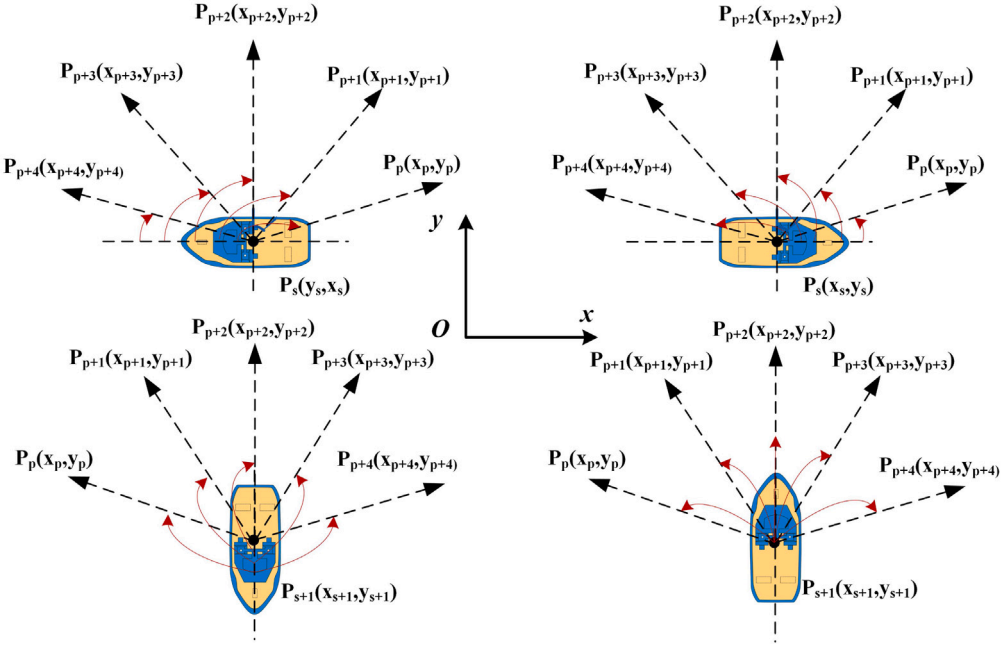


Fig. 6. Schematic diagram of ship undocking control.

2.2. Convergence proof of AMBS-P algorithm

The convergence of this algorithm is the basis to achieve the purpose of motion control. It means that the algorithm tends to a certain value after the multi-step iteration. According to the calculation process of the AMBS-P algorithm proposed in Section 2.1, the convergence of the AMBS-P algorithm is proved as follows.

Definition 1. In the kth iteration of AMBS-P algorithm, the state set of beetle swarm is $B^k = \{B_1^k, B_2^k, \ldots, B_n^k\}$. Among them, $B_n^k \in R$, $0 < n < +\infty$, $0 < k < +\infty$, B and R represent beetle position and real numbers space respectively, n is the number of beetles. $\{B^k, k > 0\}$ constitutes a discrete stochastic process.

Definition 2. Define the optimal solution as:

$$BE^* = \{B^* | \nexists B_i^k \neq B^*, F(B_i^k) \leq F(B^*), i = 1, 2, \ldots, n\}, \tag{13}$$

Minimum value of F is $F_{best} = F(B^*)$. Let $N(B^k) = |B^k \cap BE^*|$, which represents the number of optimal solutions contained in the beetle swarm.

Definition 3. The convergence of AMBS-P algorithm to the optimal value with probability 1 means that there is result for any initial state B^0 according to Eq. (14).

$$\lim_{k \to +\infty} P(N(B^k) > 0 | B(0) = B^0) = 1. \tag{14}$$

Definition 4. The predicted total fitness value is expressed as F. This is the jth prediction,

$$F(j : j + N_p) = Fmin, j = 1, 2, \ldots, +\infty, Fmin \in R, \tag{15}$$

If the result is in accordance with this Eq. (15) it is said that every prediction of F converges to an unfixed minimum value $Fmin$.

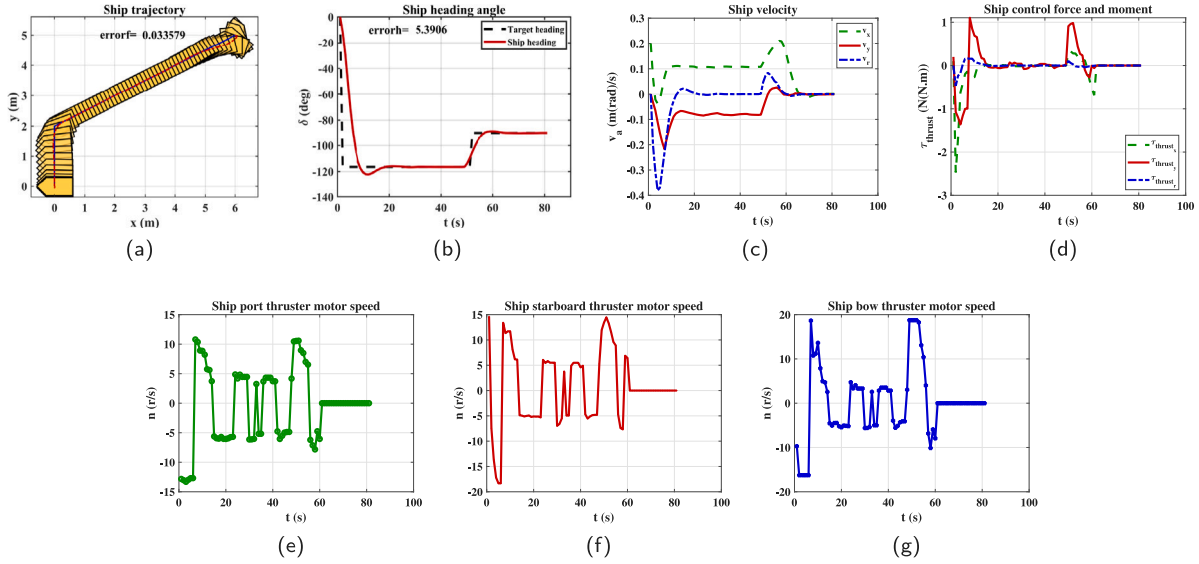


Fig. 7. Horizontal docking (port docking) without ship drag. ((a): the trajectory of ship docking; (b): the comparison of ship calculated and target headings; (c) the change of ship velocity; (d) the change of ship input force and moment; (e) the RPS of ship port thruster motor; (f) the RPS of ship starboard thruster motor; (g) the RPS of ship bow thruster motor.)

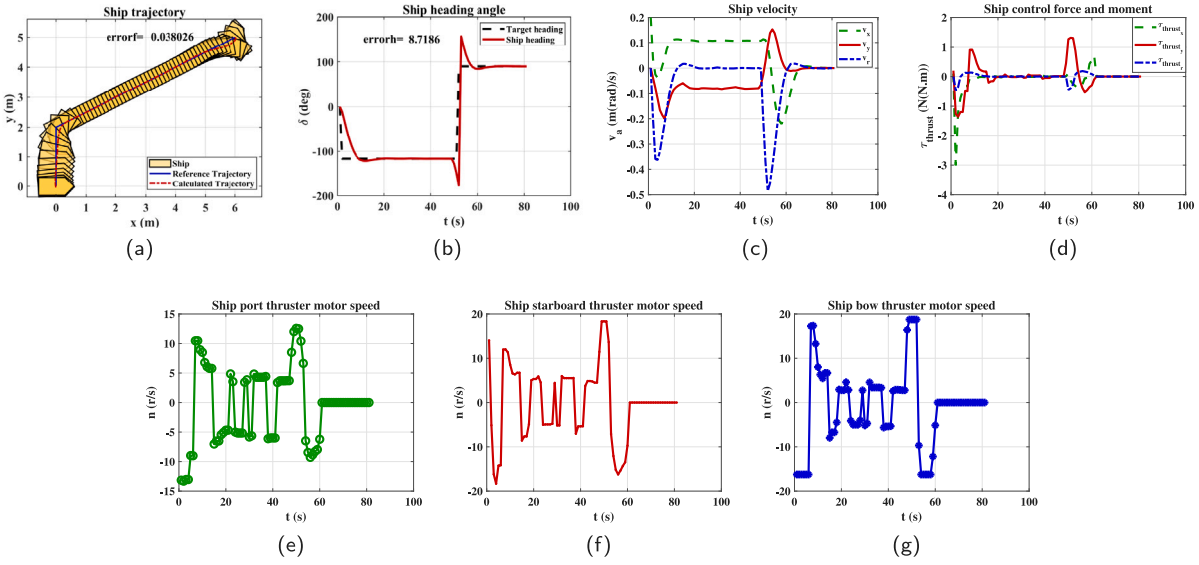


Fig. 8. Horizontal docking (starboard docking) without ship drag. (The meanings of (a)–(g) are the same with Fig. 7).

Theorem 1. *The optimization process of AMBS-P algorithm $B_k, k > 0$ is a finite homogeneous Markov chain.*

Proof. At the beginning $k = 0$, the state $B(0) = \{B_1^0, B_2^0, \ldots, B_n^0\}$ is a series of randomly generated values. In the following iteration, based on the current kth beetle swarm state $B^k = \{B_1^k, B_2^k, \ldots, B_n^k\}$, it search the solution space and update the beetle swarm according to the useful information in the search process. Finally, we obtain that the next $(k+1)th$ beetle swarm state B^{k+1}. Substituting Eq. (3) into (5):

$$B_i^{k+1} = B_i^k - L_{step}^k \cdot \tilde{D}^k \cdot sign(F(B_i^k + D_0 \cdot \tilde{D}^k/2) - F(B_i^k - D_0 \cdot \tilde{D}^k/2)), \tag{16}$$

It indicates that the probability $P(B^{k+1}|B^k)$ of the next state of the beetle swarm just depends on the current state B^k and is a constant independent of time. The formula is:

$$P(B^{k+1} = B(i)|B^k = B(j)) = \zeta, \zeta > 0, \tag{17}$$

where, $B(i)$ and $B(j)$ are two arbitrary states. It can be seen that $\{B^k, k > 0\}$ is a homogeneous Markov chain. Because the size of the beetle swarm is limited, if the search space is limited, then the beetle swarm is limited, and the state space of Markov process composed of such beetle group set is limited. Therefore, the search process $\{B^k, k > 0\}$ constructed by the AMBS-P algorithm is a finite homogeneous Markov chain.

Theorem 2. *$F_{best}(B_i)$ is not increased in the AMBS-P algorithm.*

Proof. It can be seen from the algorithm that at any k, If $F(B_i^k) < F_{best}(B_i)$, then $F_{best}(B_i) = F(B_i^k)$.

Theorem 3. *For $\forall k \geq 0$, there is:*

$$P(N(B^{k+1}) < \xi | N(B^k) = \xi) = 0, \xi \geq 0, \tag{18}$$

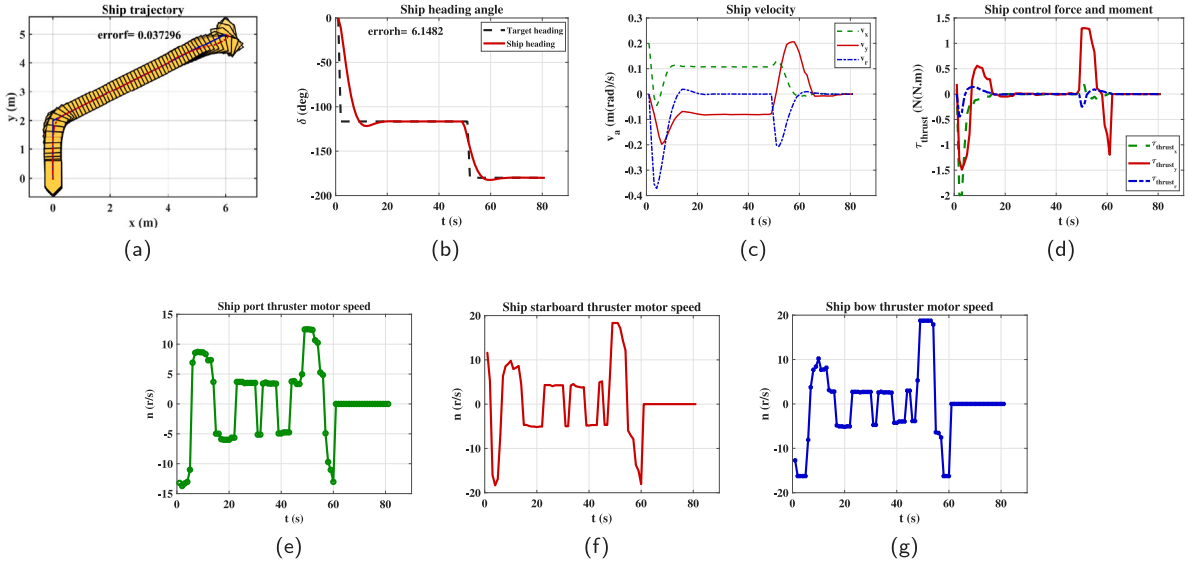


Fig. 9. Vertical docking (bow docking) without ship drag. (The meanings of (a)–(g) are the same with Fig. 7).

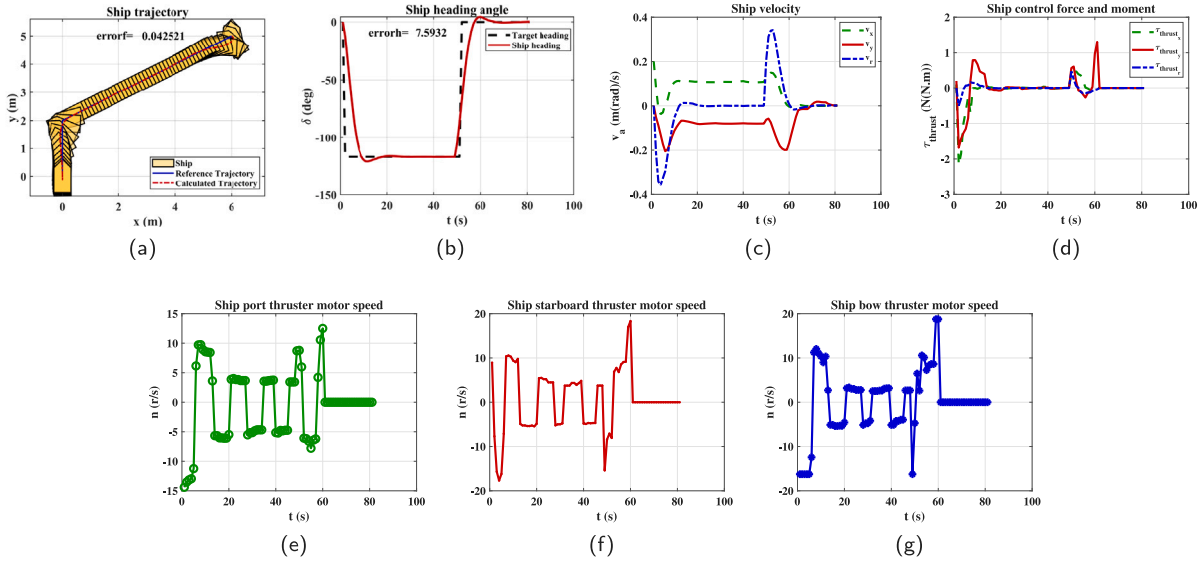


Fig. 10. Vertical docking (stern docking) without ship drag. (The meanings of (a)–(g) are the same with Fig. 7).

Proof. Although it is known from Theorem 2 that the adaptive value F of each iteration is constantly updating, as $F(B_i^k) < F_{best}(B_i)$, the selection strategy of AMBS-P algorithm is to retain the better beetle particles and fitness value of each iteration. Therefore, at any k, under the condition that the number of optimal solutions in the beetle swarm is $\xi(\xi \geq 0)$, the number of optimal solutions in the beetle swarm at $k+1$ cannot be less than ξ.

Theorem 4. *The global optimal solution of AMBS-P algorithm can be find at any time, that is,*

$$P(N(B^{k+1}) > 0 | N(B^k) = 0) > 0, \forall k \geq 0, \tag{19}$$

Proof. It can be seen from the AMBS-P algorithm that the initial $B(0)$ is a series of randomly generated values, so the probability of B^k being any possible solution at any time is not 0, and the probability of appearing the global optimal solution at the same time is not 0. Therefore, under the condition of $N(B^k) = 0$, the probability of $N(B^{k+1}) \neq 0$ is greater than 0.

Theorem 5. *AMBS-P algorithm converges to the optimal solution with probability* 1.

$$\lim_{k \to +\infty} P(N(B^k) > 0) = 1, \tag{20}$$

Proof. Set the probability that the number of optimal solutions is ι in the beetle swarm at k time be $P_\iota(k) = P(N(B^k) = \iota)$. From the Bayesian conditional probability formula, it can obtain:

$$\begin{aligned} P_0(k+1) = P(N(B^{k+1}) = 0) &= P(N(B^{k+1}) = 0 | N(B^k) = 0) \\ &\times P(N(B^k) = 0) \\ &+ P(N(B^{k+1}) = 0 | N(B^k) \neq 0) \times P(N(B^k) \neq 0), \end{aligned} \tag{21}$$

According to Theorem 3, $P(N(B^{k+1}) = 0 | N(B^k) \neq 0) = 0$, so,

$$\begin{aligned} P_0(k+1) &= P(N(B^{k+1}) = 0 | N(B^k) = 0) \times P(N(B^k) = 0) \\ &= P(N(B^{k+1}) = 0 | N(B^k) = 0) \times P_0(k), \end{aligned} \tag{22}$$

According to Theorem 4, $P(N(B^{k+1}) > 0 | N(B^k) = 0) > 0$. Let

$$C = min(P(N(B^{k+1}) > 0 | N(B^k) = 0), k = 0, 1, \ldots, +\infty), \tag{23}$$

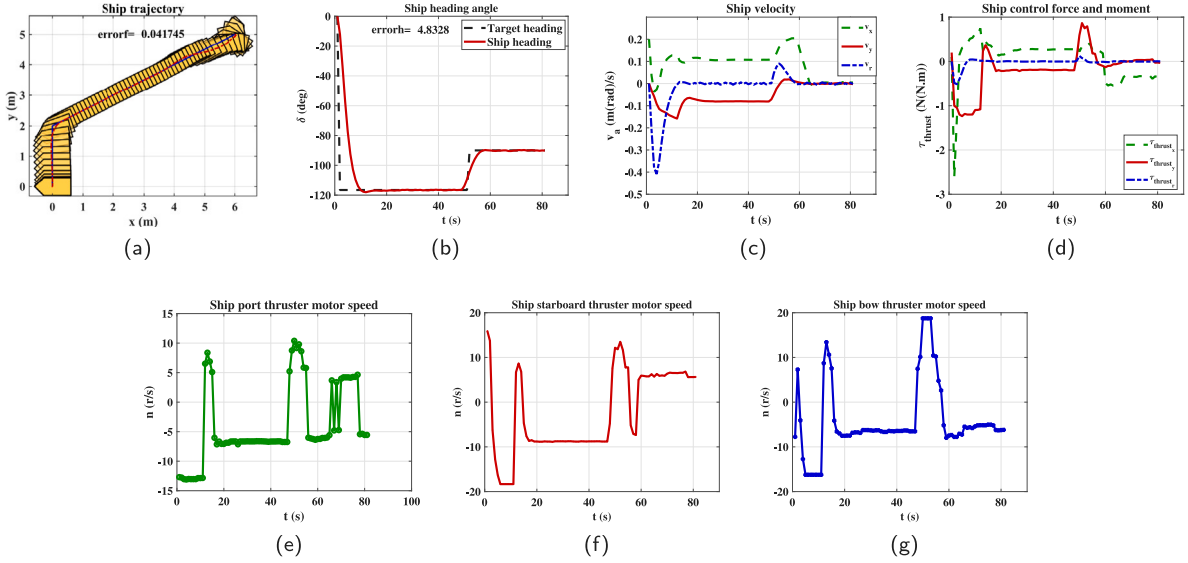


Fig. 11. Horizontal docking (port docking) with ship drag. (The meanings of (a)–(g) are the same with Fig. 7).

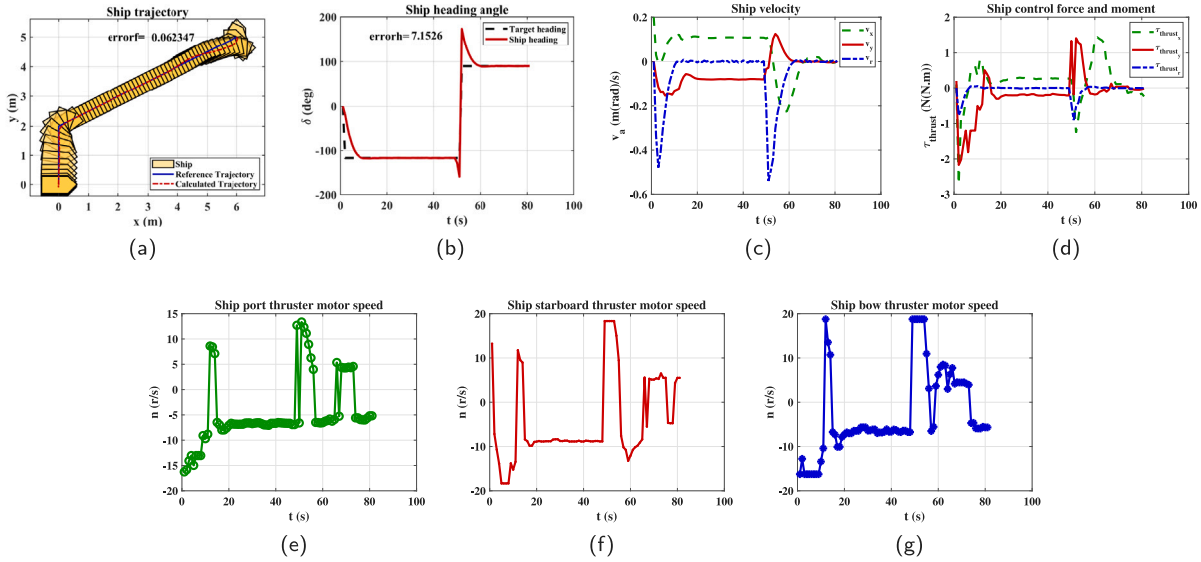


Fig. 12. Horizontal docking (starboard docking) with ship drag. (The meanings of (a)–(g) are the same with Fig. 7).

that is

$$P(N(B^{k+1})>0|N(B^k)=0)\geq C>0, \tag{24}$$

then

$$\begin{aligned}&P(N(B^{k+1})=0|N(B^k)=0)=1-P(N(B^{k+1})\neq 0|N(B^k)=0)\\&=1-P(N(B^{k+1})>0|N(B^k)=0)\leq 1-C<1,(k=0,1,\ldots,+\infty),\end{aligned} \tag{25}$$

So,

$$P_0(k+1)\leq(1-C)P_0(k)\leq\cdots\leq(1-C)^{k+1}P_0(0), \tag{26}$$

Because $\lim_{k\to+\infty}(1-C)^{k+1}=0$ and $0\leq P_0(0)\leq 1$ (for any probability), when $k\to+\infty$,

$$0\leq P_0(k+1)\leq(1-C)^{k+1}\times P_0(0)=0, \tag{27}$$

so, $\lim_{k\to+\infty}P_0(k+1)=0$.

To sum up,

$$\begin{aligned}\lim_{k\to+\infty}P(N(B^{k+1})>0)&=1-\lim_{k\to+\infty}P(N(B^{k+1})=0)\\&=1-\lim_{k\to+\infty}P_0(k+1)=1,\end{aligned} \tag{28}$$

that is, $\lim_{k\to+\infty}P(N(B^k)>0)=1$.

Theorem 6. *If* $\lim_{k\to+\infty}P(N(B^k)>0)=1$, *the* F *of the prediction* N_p *step converges to the unfixed minimum value* $Fmin$ *of the* j*th prediction.*

If $\lim_{k\to+\infty}P(N(B^k)>0)=1$, then the global optimal solution is B^*, the optimal adaptive value is $F_{best}(B^*)$, any particle except the optimal solution is set as B, and the corresponding adaptive value is $F(B)$. According to Theorem 2 and Eq. (8), $F_{best}(B^*)<F(B)$, so,

$$\begin{aligned}F(j:j+N_p)&=\sum_{j=1}^{N_p}q_j\cdot F_{best}^j=q_j\times(F_{best}^j(B^*)+F_{best}^{j+1}(B^*),\ldots,F_{best}^{j+N_p}(B^*))\\&<q_j\times(F^j(B)+F^{j+1}(B),\ldots,F^{j+N_p}(B)),\end{aligned} \tag{29}$$

That is to say, the F always converges to the minimum value $Fmin$ of the jth prediction. Because of the change of practical application, $Fmin$ is not a fixed value, but $Fmin\in R$.

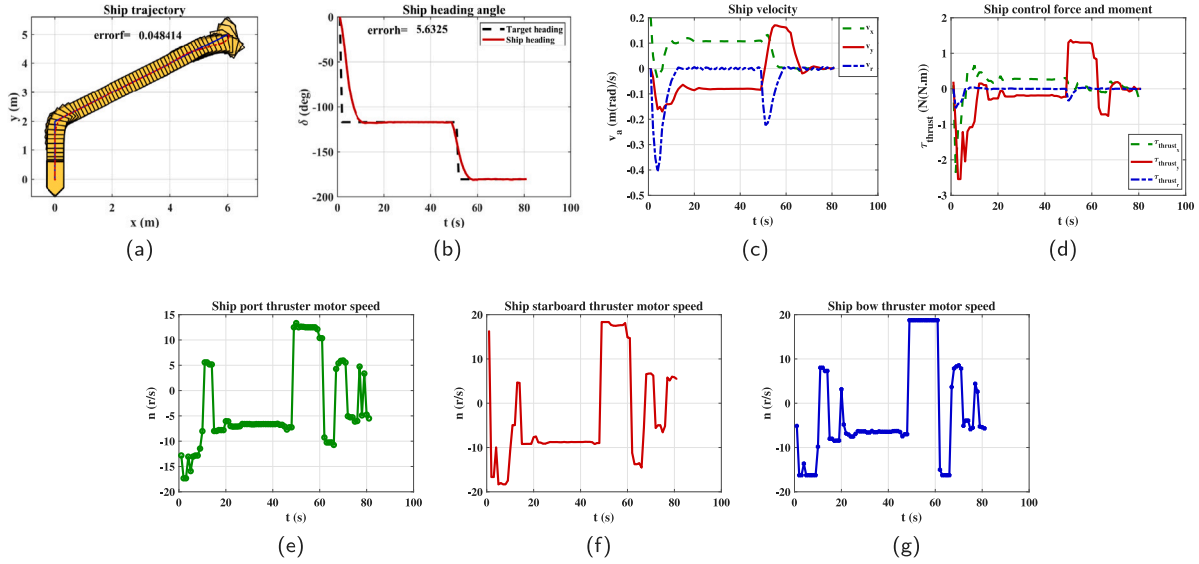


Fig. 13. Vertical docking (bow docking) with ship drag. (The meanings of (a)–(g) are the same with Fig. 7).

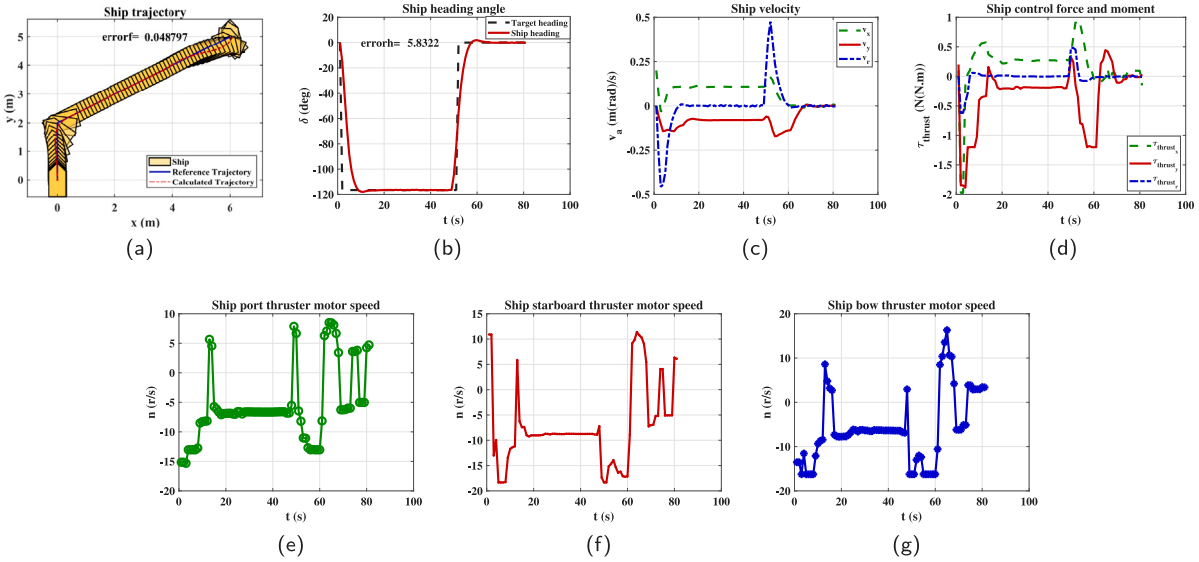


Fig. 14. Vertical docking (stern docking) with ship drag. (The meanings of (a)–(g) are the same with Fig. 7).

To sum up, AMBS-P algorithm has convergence, which indicates that a definite result can be obtained after calculation, so this algorithm can be used for the design and application of motion controllers.

3. Mathematical model of the ship

The foundation of ship motion control is to build a better maneuverability model. This paper chooses the ship space "Tito-Neri" as the research object. The coordinate system and degrees of freedom used in this study are shown in Fig. 1. Among them, $O_b - x_b y_b z_b$ is the body-fixed coordinate system, and $O - xyz$ can be approximately regarded as the inertial coordinate system of ships sailing only in a local areas. Fig. 1 shows that the three degrees of freedom (3-DOF) model is adopted (swing, surge, and yaw direction).

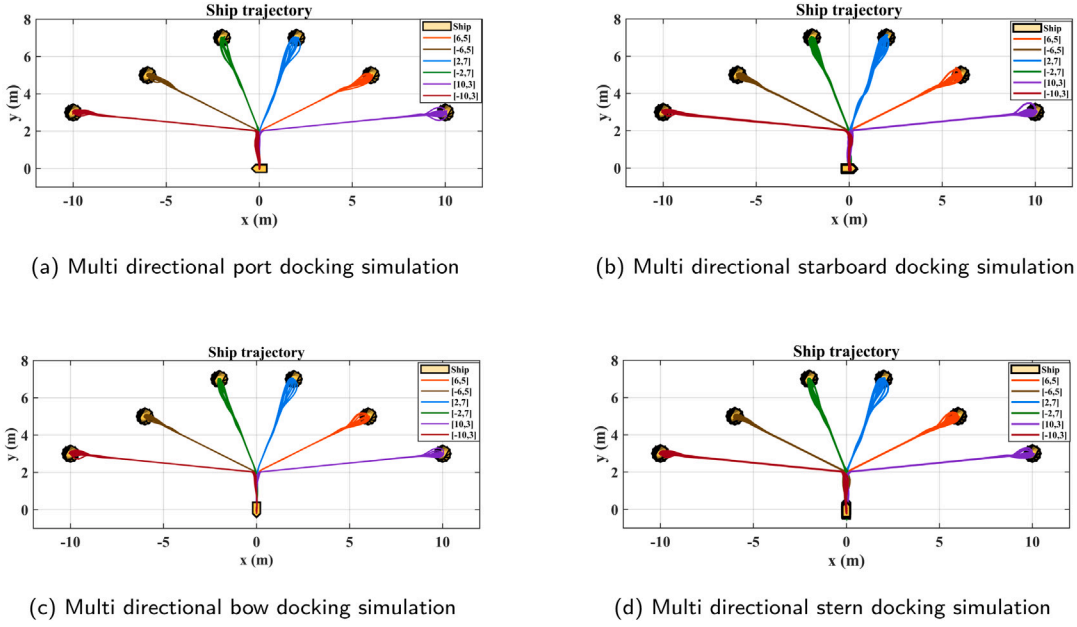
"Tito-Neri" ship is a scaled model ship developed by Delft University of Technology, with a length of 0.97 m and a width of 0.3 m. The structure (Bruggink et al., 2018) and ship model are shown in Fig. 2. There are two stern thrusters and one bow thruster. This structure makes the model ship very flexible. The ship motion maneuvering model is selected in the form Eq. (30) (Fossen, 2002). The meanings of ship symbols are shown in Table 10.

$$\begin{aligned} &\dot{\eta} = R(\eta)\upsilon_a \\ &(M_{RB} + M_A)\dot{\upsilon}_a + C_{RB}(\upsilon_a)\upsilon_a + C_A(\upsilon_a)\upsilon_a = \tau + \tau_{distrub} \end{aligned} \tag{30}$$

The input of the model is the force and moment produced by thrusters, and the output is the distance (in meters) in X and Y directions and the heading angle (in radians) of the ship.

According to the Eq. (30), the position and heading are $\eta = [x, y, \psi]^{\mathrm{T}}$, $R = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$, the velocity is $\upsilon_a = [\upsilon_x, \upsilon_y, \upsilon_r]^{\mathrm{T}}$, $\tau = \tau_{thrust} + \tau_{drag}$, C_A is omitted in this article, $C_{RB} = \begin{bmatrix} 0 & 0 & -m_b\upsilon_y \\ 0 & 0 & m_b\upsilon_x \\ m_b\upsilon_y & -m_b\upsilon_x & 0 \end{bmatrix}$. The meaning of ship basic variables is shown in Appendix A.

Because the ship is small, the resistance caused by the water has effect on the ship motion. The ship drag needs to be calculated in



(a) Multi directional port docking simulation

(b) Multi directional starboard docking simulation

(c) Multi directional bow docking simulation

(d) Multi directional stern docking simulation

Fig. 15. Multi directional horizontal and vertical docking simulation.

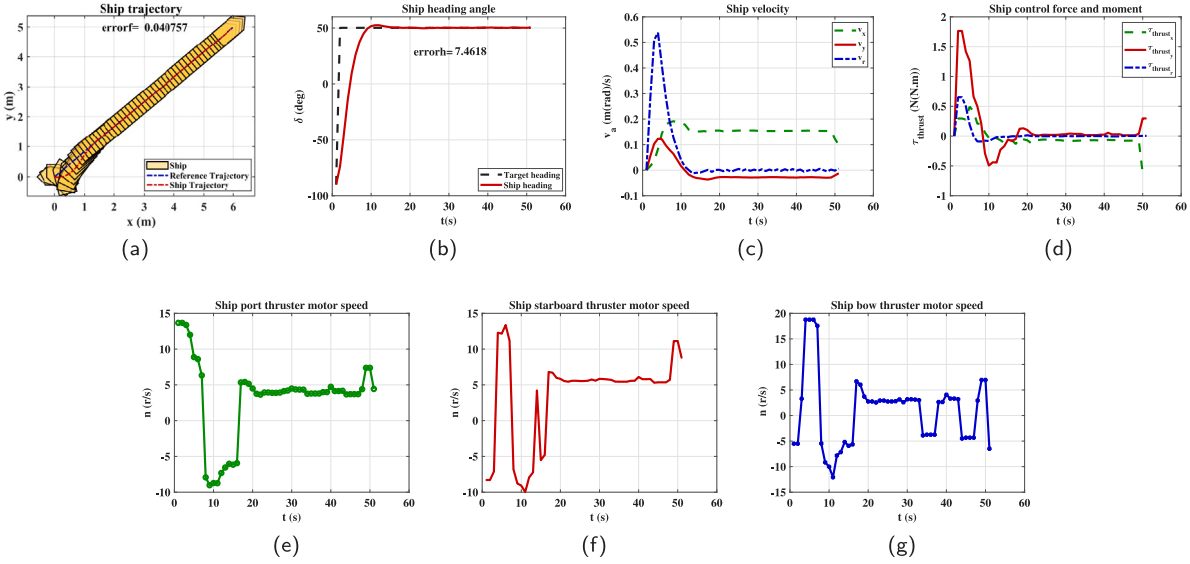


Fig. 16. Horizontal undocking (port undocking) simulation. ((a): the trajectory of ship undocking; (b): the comparison of ship calculated and target headings; (c) the change of ship velocity; (d) the change of ship input force and moment; (e) the RPS of ship port thruster motor; (f) the RPS of ship starboard thruster motor; (g) the RPS of ship bow thruster motor.)

the actual motion control. Setting $\tau_{drag} = [\tau_{dragx}; \tau_{dragy}; \tau_{dragr}]$. The estimates results are shown in Appendix B. Besides, τ_{dragr} is shown in Eq. (31):

$$\tau_{dragr} = \frac{1}{3}\tau_{dragr}(\frac{pi}{2}, \frac{2v_r}{3}). \tag{31}$$

For more information about the "Tito-Neri" model, see Damen Shipyards (2018) and Bruggink et al. (2018). In the following part of algorithm verification, this study will take two cases: with adding ship drag and without drag.

It is necessary to convert the force and moment into the RPS (Rounds Per Second) of the motor driving thrusters for the actual navigation. First of all, the problem of the ship's thrust allocation requires to be solved. $\tau_{thruster}$ is defined as $\tau_{thruster} = [\tau_{thrusterx}; \tau_{thrustery}; \tau_{thrusterr}]$, where $\tau_{thrusterx}$ and $\tau_{thrustery}$ are represent surge and sway forces respectively, $\tau_{thrusterr}$ represents yaw moment. The thruster structure of the ship is shown in Fig. 3. According to the (Haseltalab and Negenborn, 2019):

$$\tau_{thruster} = \tilde{T}_{3\times m}[g_1(n_1); ...; g_m(n_m)]^{\mathrm{T}}. \tag{32}$$

Among them, thrust configuration matrix $\tilde{T} = [\beta_1, \ldots, \beta_m]$, where β is the actuator vector column. m is the number of actuators, $g_1, \ldots, g_m$ are actuator dynamics, $n_1, \ldots, n_m$ are the actuator shaft velocity. Assuming this study controls the RPS of the motor, then for the stern thruster:

$$\beta_i = [1; 0; -l_y]^{\mathrm{T}}. \tag{33}$$

For bow thruster:

$$\beta_j = [0; 1; l_x]^{\mathrm{T}}, \tag{34}$$

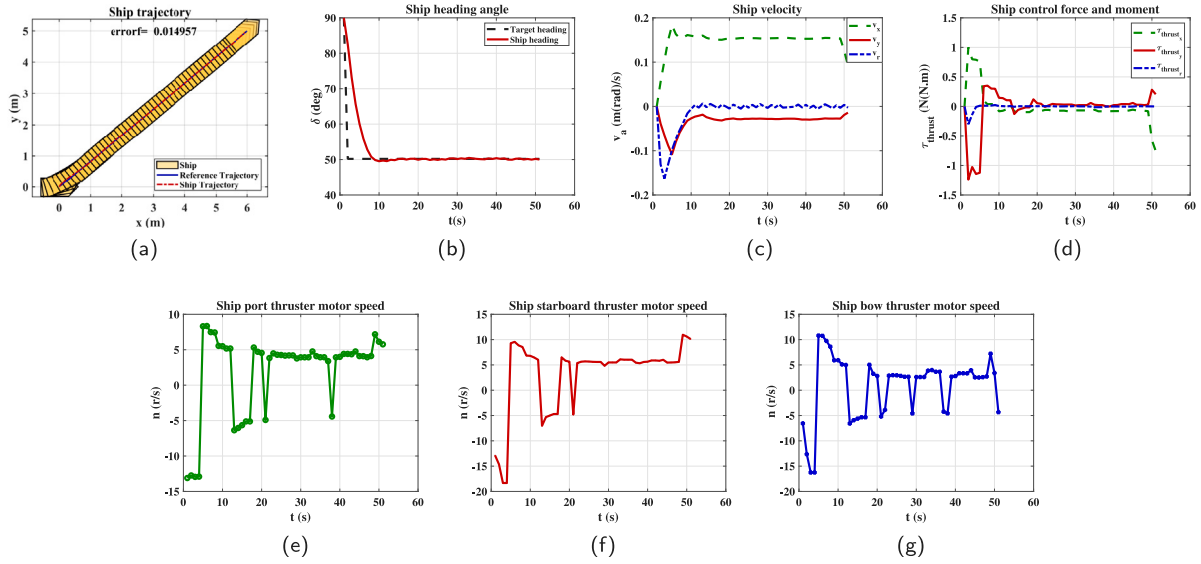


Fig. 17. Horizontal undocking (starboard undocking) simulation. (The meanings of (a)–(g) are the same with Fig. 16).

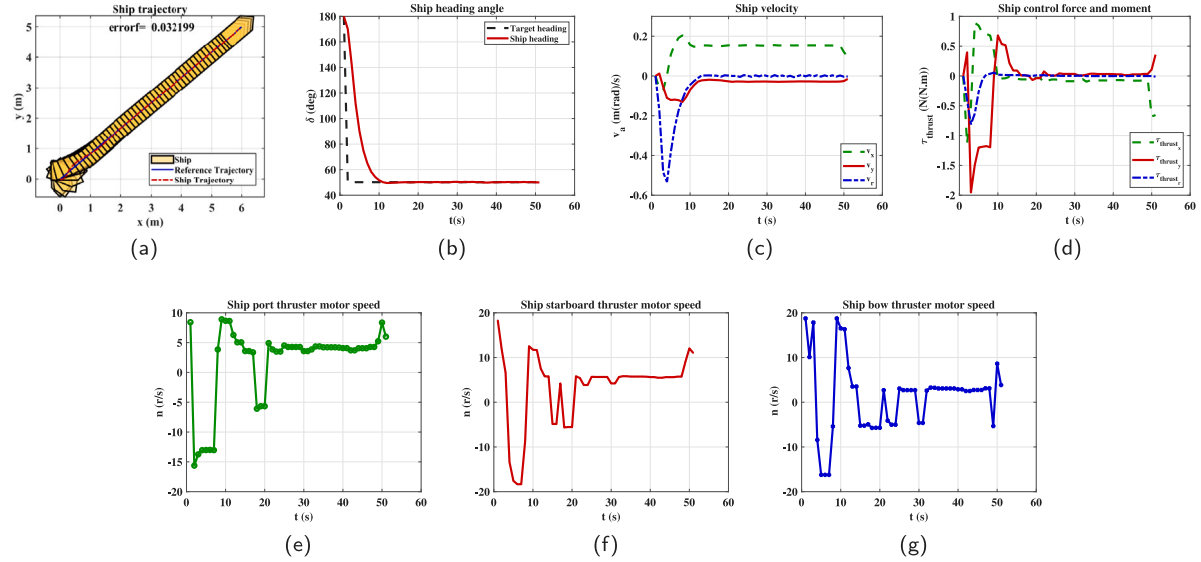


Fig. 18. Vertical undocking (bow undocking) simulation. (The meanings of (a)–(g) are the same with Fig. 16).

where l_y and l_x represent actuator position coordinates. Therefore, the RPS calculation formula of a single thruster is:

$$\begin{bmatrix} g_1(n_1) \\ \dots \\ g_m(n_m) \end{bmatrix} = (\tilde{T}_{3\times m})^{-1} \cdot \tau_{thruster}. \tag{35}$$

There are three thrusters in this study. $\tilde{T}$ is a square matrix, and its inverse matrix can be calculated directly. Otherwise, it is necessary to find its pseudo inverse matrix for ships with the number of thrusters not equal to three:

$$\tilde{T} = \tilde{T}^{\mathrm{T}}(\tilde{T}\tilde{T}^{\mathrm{T}})^{\dagger}. \tag{36}$$

4. Design of autonomous docking and undocking controller

4.1. Docking analysis

On account of the actual demand for cargo handling, there are three basic modes of docking: (1) horizontal docking (port docking and starboard docking); (2) vertical docking (bow docking and stern docking); (3) docking between horizontal and vertical directions. Since the (1) and (2) are limit positions, the first two modes can be analyzed to include all other orientations in this study. No matter which way of docking is adopted, this study divides the design of ship docking process into two parts: the first step is that the ship directly moves from the current position to the docking area, and the second step is that the ship gradually stops at the docking area by adjusting its attitude at a low velocity, as shown in Fig. 4.

Taking the case shown in Fig. 4 as an example, in zone 1, the ship follows the path and sails to zone 2 in a straight line. The starting point of zone 2 is set at λ times the length of the ship directly above the final docking position. After arriving at zone 2, the attitude of the ship is adjusted until the velocity is reduced to zero, and the whole docking process is completed as expected.

In the first stage of docking, the control goal is to synchronously correct the ship's heading and trajectory by calculating the motor RPS for every step, so as to gradually track it to the planned target path. The principle of ship path following is shown in Fig. 5. In this paper, each path $P_p(x_p, y_p), p = 1, 2, \dots, m$ is divided into several path points $P_{p+i}(x_{p+i}, y_{p+i}), i = 1, 2, \dots, n$, and each path has a

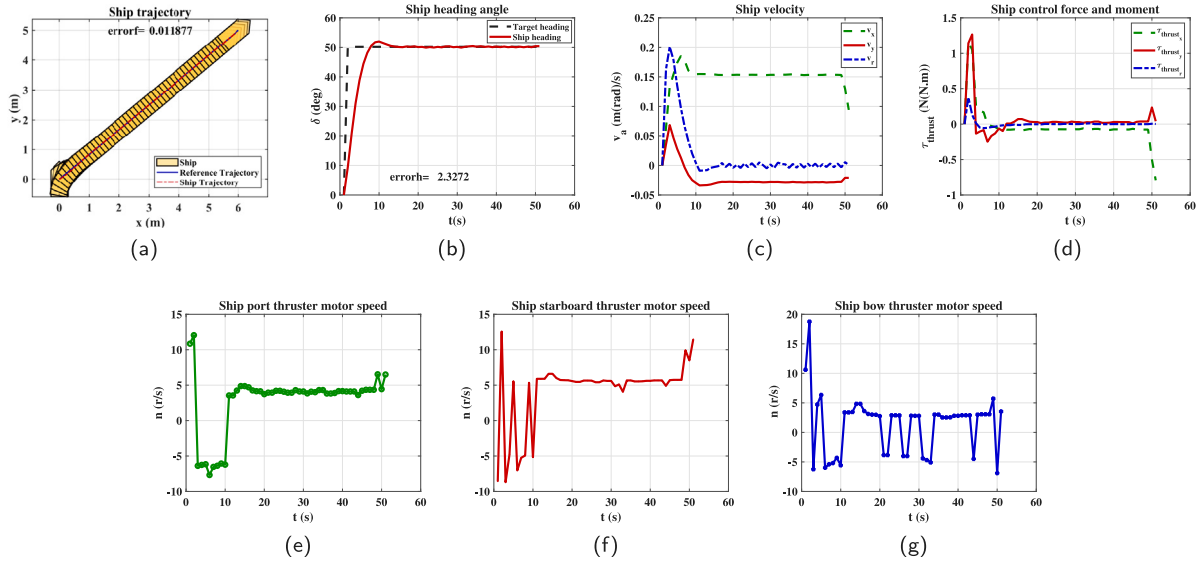


Fig. 19. Vertical undocking (stern undocking) simulation. (The meanings of (a)–(g) are the same with Fig. 16).

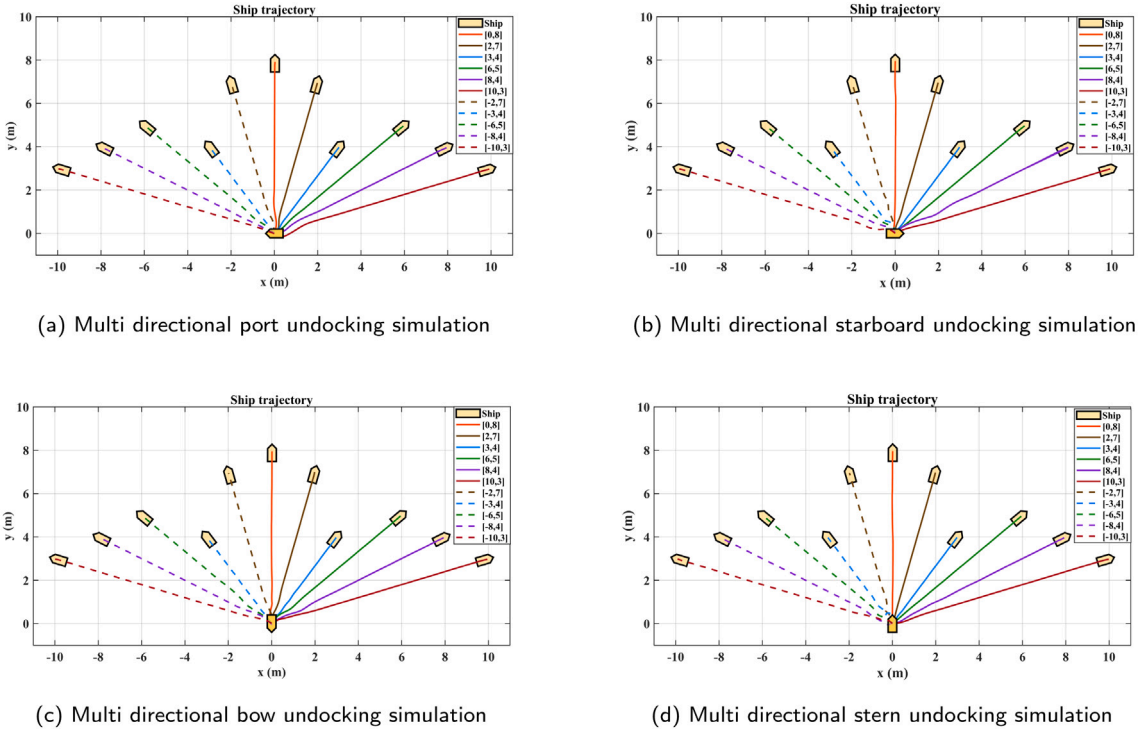


Fig. 20. Multi directional horizontal and vertical undocking simulation.

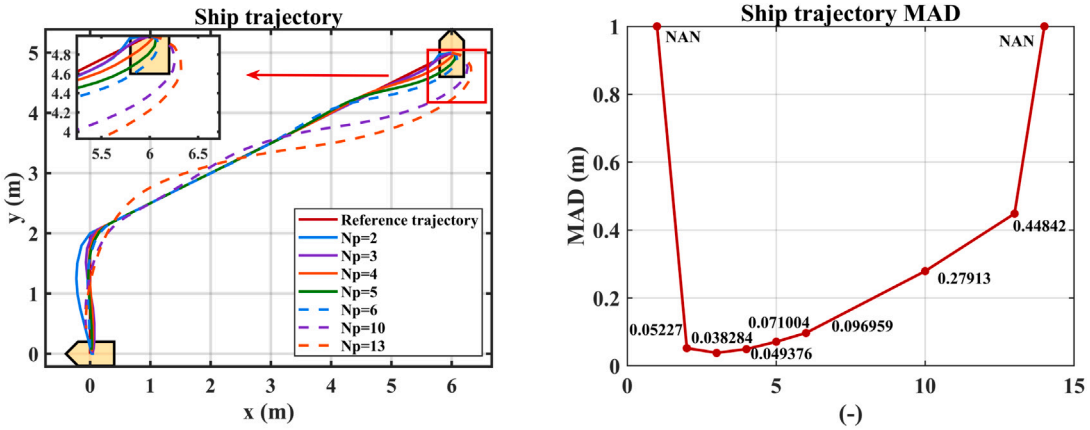
fixed direction ψ_p, and the corresponding reference velocity and rate are $[u_p, v_p, r_p]^{\mathrm{T}} = R^{-1}(\eta_p)\dot{\eta}_p, \eta_p = [x_p, y_p, \psi_p]^{\mathrm{T}}$. The ship position is $S_i(x_{s+i}, y_{s+i}), i = 1, 2, \ldots, n+m$. The control input of path following are motor RPS and output is ship state. The reference target is $R_{ref}(i) = [x_{p+i}, y_{p+i}, \psi_p, u_p, v_p, r_p]^{\mathrm{T}}, p = 1, 2, \ldots, m; i = 1, 2, \ldots, n$.

In the second stage of docking, the ship heading range is $[-180°, 180°]$, and the state of the ship is $S_{state} = [\eta, v_a] = [x, y, \psi, v_x, v_y, v_r]^{\mathrm{T}}$. The condition is that the position and heading reach the expected object when docking, that is, the position is $[0, 0]$, the heading is $-90°$, $90°$, $0°$ or $180°$, and the velocity is reduced to zero. That is, $v_x = 0; v_y = 0; v_r = 0$.

The advantage of this design is that the docking path can be set and executed adaptively according to this rule. It avoids the problem of path setting and selection caused by the uncertainty of ship heading when the docking command is executed.

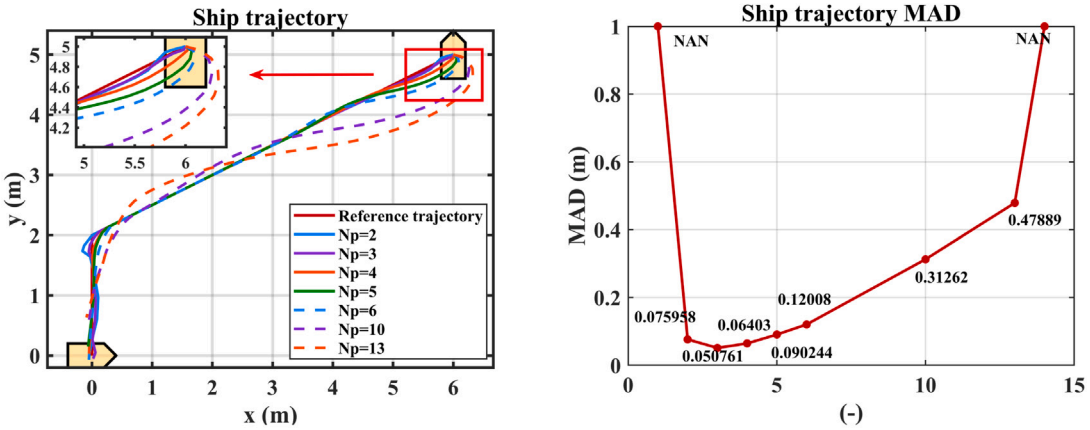
4.2. Undocking analysis

Ship undocking is to control the ship from docking to the starting state. Due to the different target headings of the ship, it is required that the ship can quickly switch from any docking mode to the target attitude, as shown in Fig. 6.



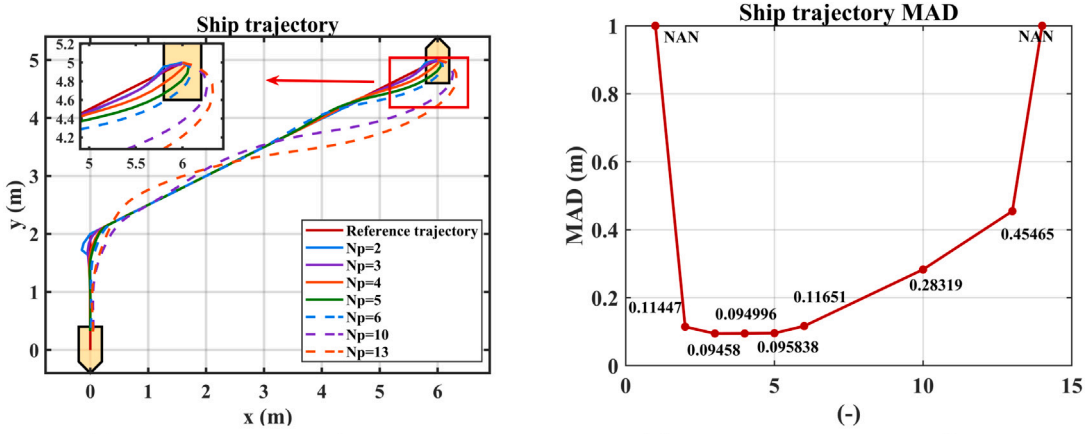
(a) Port docking under different N_p (b) Port docking error under different N_p

Fig. 21. Horizontal docking (port docking) analysis under different N_p.



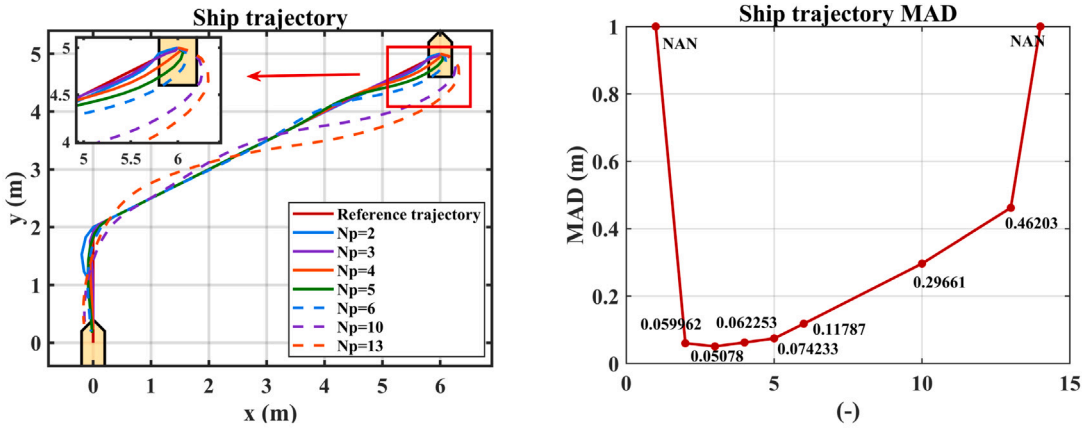
(a) Starboard docking under different N_p (b) Starboard docking error under different N_p

Fig. 22. Horizontal docking (starboard docking) analysis under different N_p.



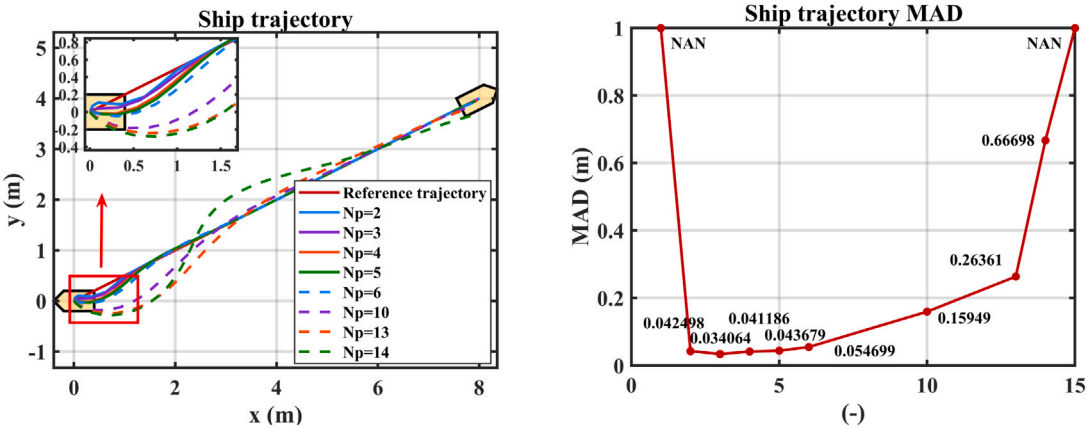
(a) Bow docking under different N_p (b) Bow docking error under different N_p

Fig. 23. Vertical docking (bow docking) analysis under different N_p.



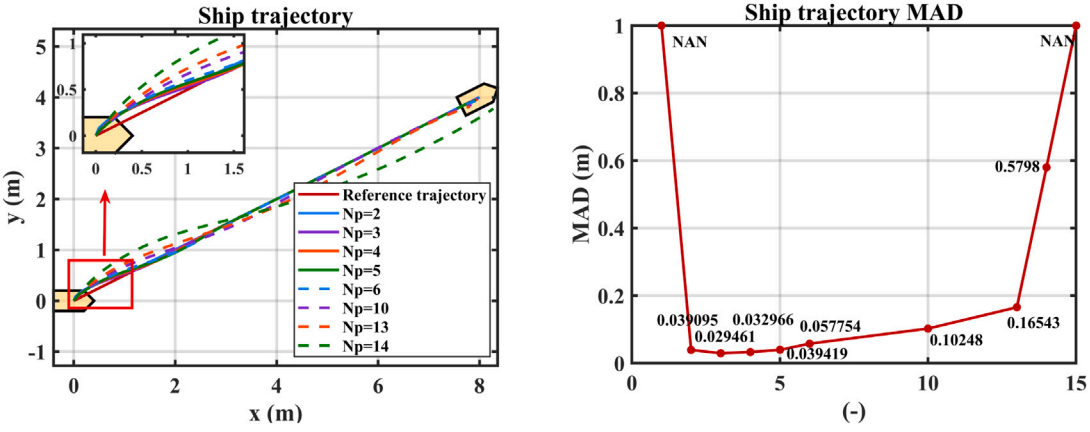
(a) Stern docking under different N_p (b) Stern docking error under different N_p

Fig. 24. Vertical docking (stern docking) analysis under different N_p.



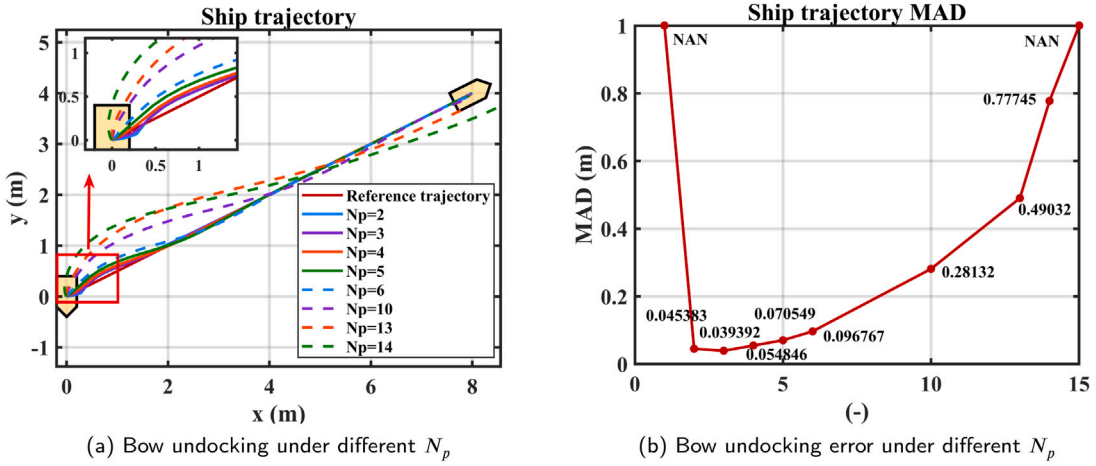
(a) Port undocking under different N_p (b) Port undocking error under different N_p

Fig. 25. Horizontal undocking (port undocking) analysis under different N_p.



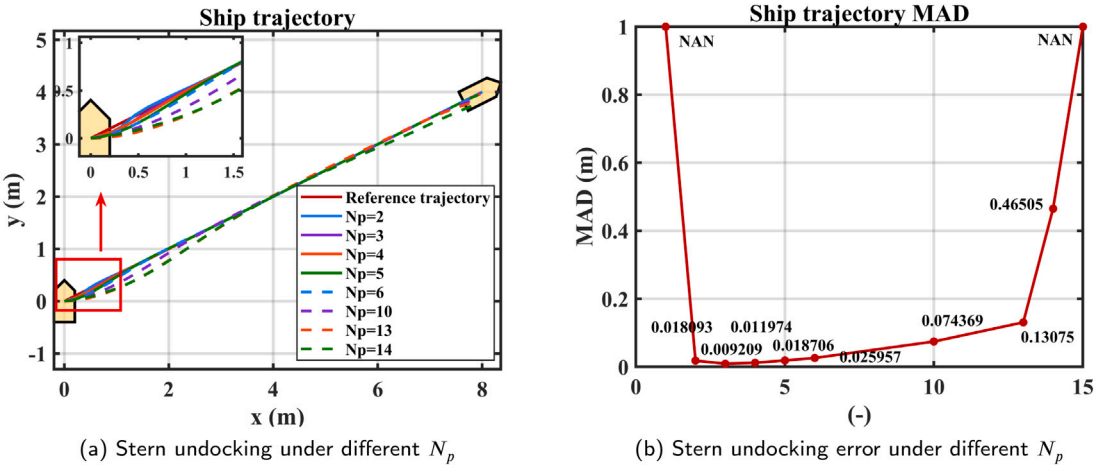
(a) Starboard undocking under different N_p (b) Starboard undocking error under different N_p

Fig. 26. Horizontal undocking (starboard undocking) analysis under different N_p.



(a) Bow undocking under different N_p (b) Bow undocking error under different N_p

Fig. 27. Vertical undocking (bow undocking) analysis under different N_p.



(a) Stern undocking under different N_p (b) Stern undocking error under different N_p

Fig. 28. Vertical undocking (stern undocking) analysis under different N_p.

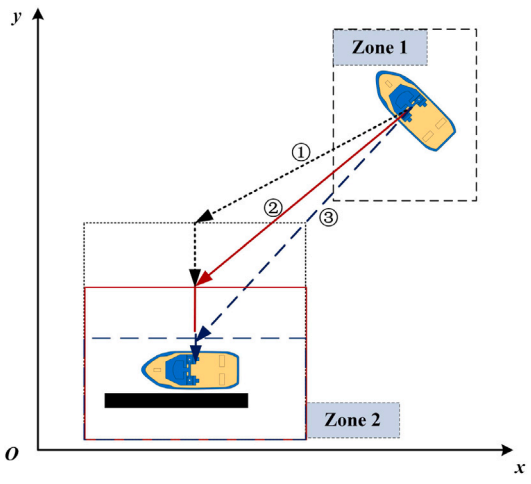


Fig. 29. Schematic diagram of path location.

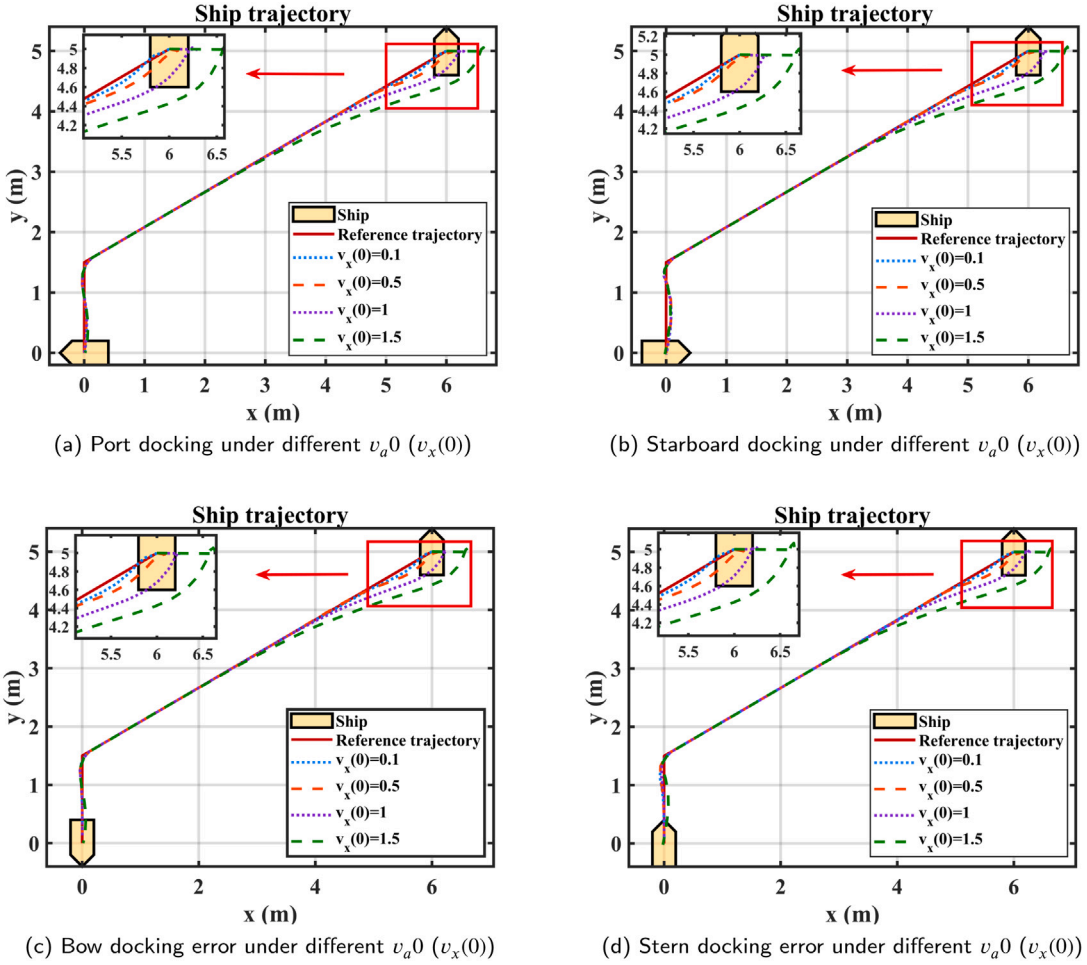
Whether docking or undocking, combined with Section 2, the Eq. (8) is detailed:

$$\begin{aligned} minF(B|j) &= \sum_{j=1}^{N_p} q_j \cdot F_{best}^j = \sum_{j=1}^{N_p} q_j \cdot (R_{ref}^j - S_{state}^j) \\ &= Q \cdot (R_{REF} - S_{STATE}), \end{aligned} \tag{37}$$

where Q is the total weight factor, $Q = diag[q_1, q_2, \ldots, q_{N_p}]$. R_{REF} is target path, $R_{REF} = [R_{ref}(j), R_{ref}(j+1), \ldots, R_{ref}(j+N_p)]^{\mathrm{T}}$. S_{STATE} is ship path, $S_{STATE} = [S_{state}(j), S_{state}(j+1), \ldots, S_{state}(j+N_p)]^{\mathrm{T}}$.

4.3. The pseudo-code of AMBS-P algorithm

According to Sections 2 and 4.2, the pseudo-code of whole algorithm is given in Algorithm 1. In this paper, the docking and undocking control of ships will be realized according to this algorithm pseudo-code.



(a) Port docking under different v_a0 ($v_x(0)$)

(b) Starboard docking under different v_a0 ($v_x(0)$)

(c) Bow docking error under different v_a0 ($v_x(0)$)

(d) Stern docking error under different v_a0 ($v_x(0)$)

Fig. 30. Docking analysis under different v_a0 ($v_x(0)$).

Algorithm 1: AMBS-P algorithm

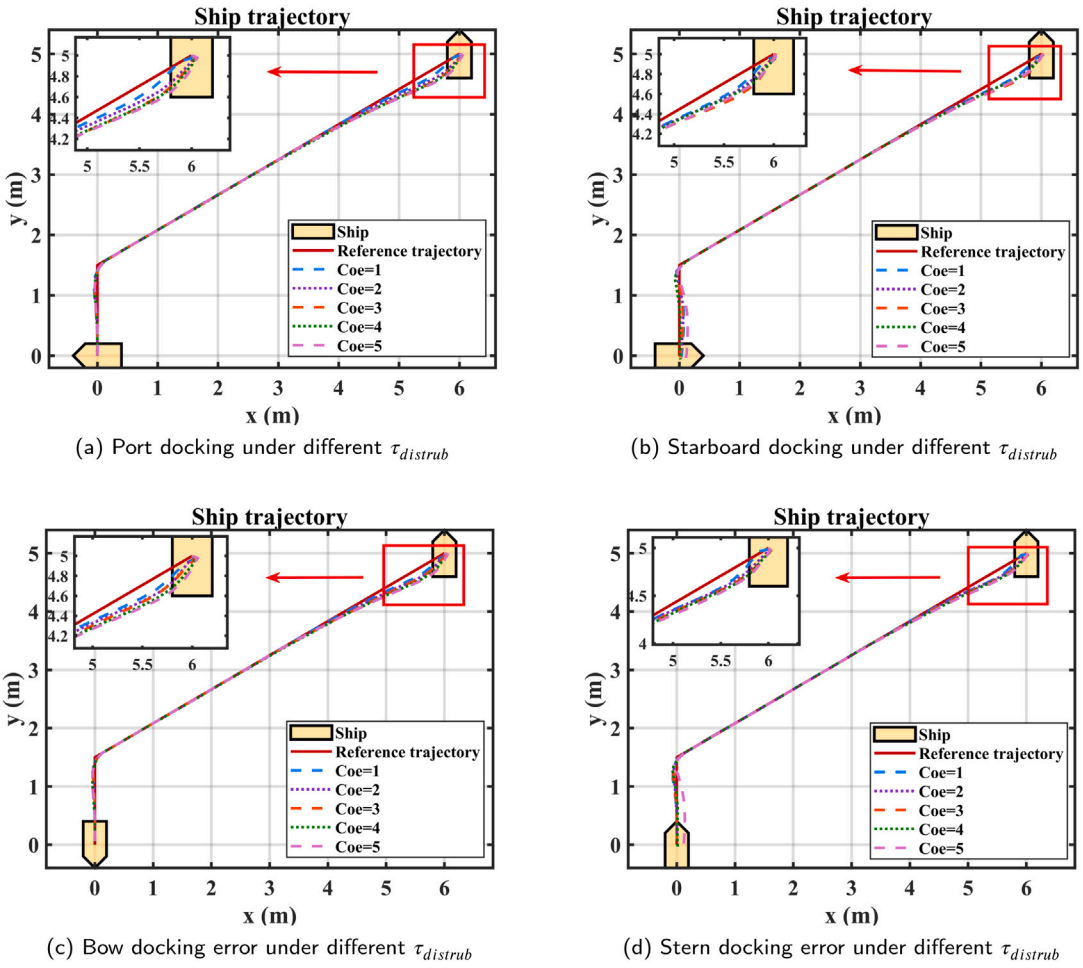
Initialize: $B(0), V(0), D_0^0, L_{step}^0, bl_1, bl_2, Pb^0 = Gb^0 = B(0), F_{best} = F(B(0))$;
while $k < K$ **do**
  for $i = 1 \rightarrow n$ **do**
    Calculate ω^k according to Eq. (2);
    Calculate the V_i^k and B_i^k of bettle according to Eqs. (1a) and (1b);
    Generate $\tilde{D}^k$ according to Eq. (4);
    Calculate $B_L i^k$ and $B_R i^k$ according to Eq. (3);
    Calculate $F(B_L i^k)$ and $F(B_R i^k)$ according to Eq. (37);
    Update the B_i^k according to Eq. (5);
    Calculate the $F(B_i^k)$ according to Eq. (3);
    if $F(B_i^k) < F_{best}(B_i^*)$ **then**
      $Pb_i^k = B_i^k$;
      $F_{best}(B_i^*) = F(B_i^k)$;
  Update the D_0^k and L_{step}^k of bettle according to Eq. (7);
  $Gb^k = min(Pb^k)$;
  $F_{best} = min(F_{best}(B^k))$;
  Calculate Θ^k and Φ^k separately according to Eqs. (9) and (10);
  Calculate p_m^k according to Eq. (11);
  Calculate the Pb^{k+1}according to Eq. (12);

Table 1
Ship motion control parameter values.

"Tito Neri" ship parameter values			AMBS-P algorithm parameter values		
Symbols	Value	Unit	Symbols	Value	Unit
Bt	[0.345,0]	[m,m]	bl_1	0.95	–
COG	[0,0]	[m,m]	bl_2	0.99	–
L	0.97	m	D_0	0.99	–
M_A	$\begin{bmatrix}1.2 & 0 & 0\\ 0 & 15 & 0\\ 0 & 0 & 1.8\end{bmatrix}$	$\begin{bmatrix}\text{kg}\\ \text{kg}\\ \text{kg m}^2\end{bmatrix}$	E_y	[7,7,7,1,1,1]	–
m_b	16.9	kg	K	40	–
M_{RB}	$\begin{bmatrix}16.9 & 0 & 0\\ 0 & 16.9 & 0\\ 0 & 0 & 0.51\end{bmatrix}$	$\begin{bmatrix}\text{kg}\\ \text{kg}\\ \text{kg m}^2\end{bmatrix}$	L_{step}	1	–
Pt	[−0.42,−0.08]	[m,m]	n	5	–
St	[−0.42,0.08]	m	N	3	–
$v_a(0)$	[0.1;0;0]	m/s	N_p	4	–
w_{boat}	0.3	m	λ	2	–
			Θ	1	–

5. Simulation of autonomous docking and undocking control of ships

The basic parameters of the ship and the AMBS-P algorithm are shown in Table 1. According to the basic parameters and different initial conditions, the feasibility and effect of the application of the AMBS-P algorithm in ship docking and undocking are verified and analyzed.



(a) Port docking under different $\tau_{distrub}$

(b) Starboard docking under different $\tau_{distrub}$

(c) Bow docking error under different $\tau_{distrub}$

(d) Stern docking error under different $\tau_{distrub}$

Fig. 31. Docking analysis under different $\tau_{distrub}$.

5.1. Autonomous docking verification

5.1.1. Docking control in ideal situation

Horizontal docking and vertical docking means the final heading are $\psi = -90°$, $\psi = 90°$, $\psi = 0°$ or $\psi = 180°$ respectively. First of all, ideally, it is assumed that the influence of water resistance on the ship is ignored, that is $\tau_{drag} = [0; 0; 0]$. The position and heading are optional, such as $[x, y, \psi] = [6, 5, 0]$. The simulation effect is shown in Figs. 7, 8, 9 and 10. In this paper, the Mean Absolute Deviation (MAD) is selected as the standard to evaluate the algorithm.

MAD can avoid the offset of positive and negative errors so that its value can better reflect the simulation effect. Figs. 7(a), 8(a), 9(a) and 10(a) respectively show the trajectory of horizontal docking and vertical docking. Figs. 7(b)–(g), 8(b)–(g), 9(b)–(g) and 10(b)–(g) respectively show the change of ship heading, the ship velocity, the ship input force and moment, and the RPS of the stern and bow motors. It can be seen from Figs. 7, 8, 9 and 10 that due to the ignorance of the ship drag, the velocity can be better maintained at zero at the end of docking. Following the velocity, the thrust and moment generated by the propeller can gradually tend to zero.

5.1.2. Docking control in actual situation

The hull in this paper is small, in the actual process, a small velocity can also cause the drag. Because the ship needs to stop at a fixed position and the ship itself has inertia, at the end of docking, the actual ship velocity will gradually tend to zero and fluctuates around it. Therefore, the force and moment produced by thrusters are close to but not kept at zero at the end of docking to resist the ship drag. The simulation results are shown in Figs. 11(d), 12(d), 13(d) and 14(d).

By analogy, set six positions and twelve different initial headings within $[-180°, 180°]$, as shown in Table 2. At each position, twelve different initial headings are tested. Taking the case of ship drag as an example, autonomous docking is realized for multiple directions and positions to verify the generality of the algorithm. It can be seen from Fig. 15 that the AMBS-P algorithm has a good application effect for docking in any direction.

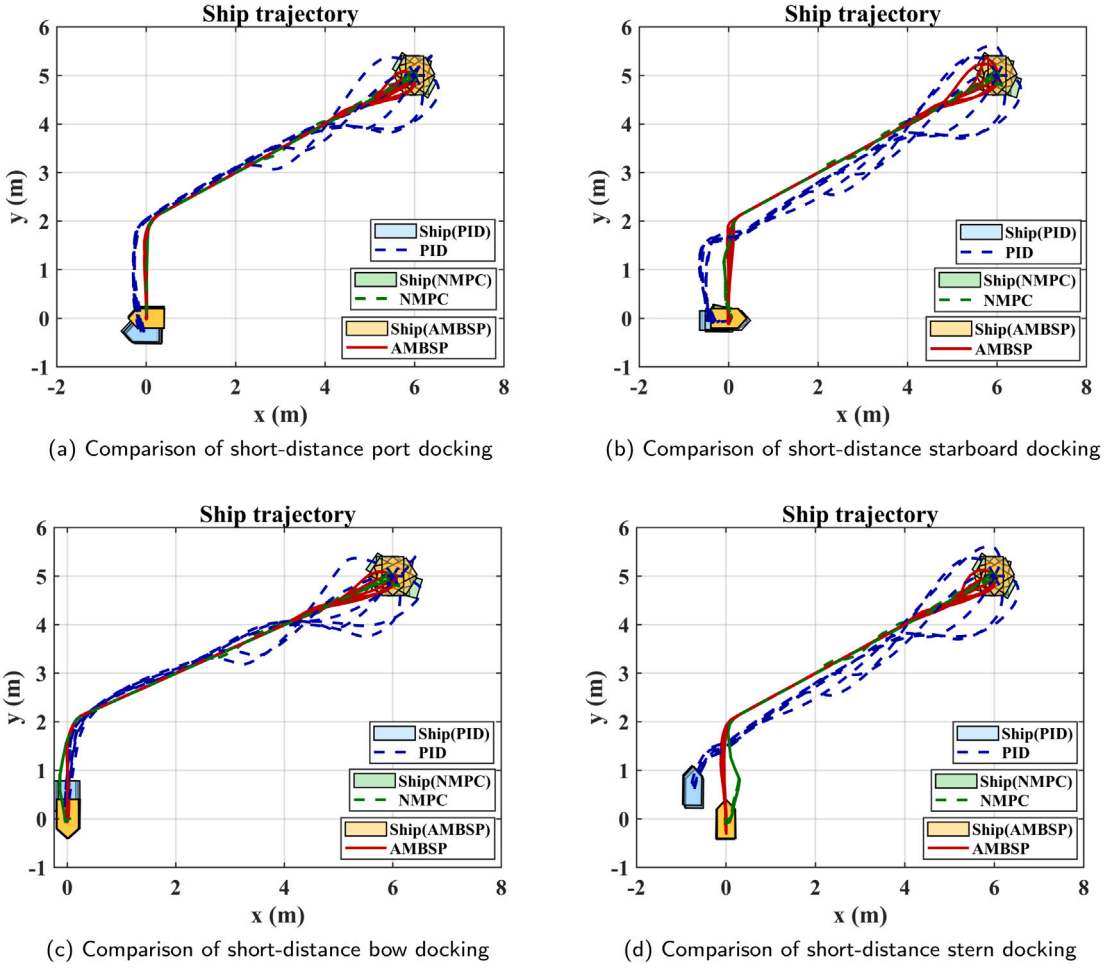
5.2. Autonomous undocking verification

Following the docking control, the current docked ship is located at a position vertical or horizontal to the shore. The ship will move from the initial docking position to any desired course and position when the undocking control is carried out. The simulation results are shown in Figs. 16, 17, 18 and 19. The multi-directional tests are shown in Fig. 20. Fig. 20 shows no matter what direction the ship destination is, it can follow the target trajectory quickly. The above simulation shows that the AMBS-P algorithm can achieve autonomous undocking better.

5.3. Results analysis and comparison

5.3.1. Results analysis

According to the principle of the AMBS-P algorithm, the main control variables that affect the results of docking and undocking are: N_p, λ, $v_a(0)$, $\tau_{distrub}$ and calculation efficiency. In addition, bl_1, bl_2 and D_0 are less sensitive to the algorithm, please refer to the paper (Wang et al., 2019) for details, so the analysis will not be repeated here. For each variable value to be analyzed, it will be repeated 100 times to calculate the average MAD and trajectory. Then, one by one analysis.



(a) Comparison of short-distance port docking

(b) Comparison of short-distance starboard docking

(c) Comparison of short-distance bow docking

(d) Comparison of short-distance stern docking

Fig. 32. Comparison of short-distance docking effect between PID, NMPC and AMBS-P algorithm.

Table 2
Set different position and heading for docking.

Initial position ([x (m), y (m)])	Initial heading (degree)
[6, 5], [−6, 5], [2, 7], [−2, 7], [10, 3], [−10, 3]	0°, 30°, 60°, 90°, 120°, 150°, −30°, −60°, −90°, −120°, −150°, −180°

(1) Analysis of N_p on the results.

According to the principle of ship motion control, the N_p is an important factor on the effect of ship docking and undocking. The simulation under different N_p are shown in Figs. 21, 22, 23 and 24.

Every step of optimization needs to ensure efficient operation, so the optimal value within a certain error range is needed. This cause the results of the algorithm are slightly different. The trajectories shown in Figs. 21, 22, 23 and 24 are the average trajectories of 100 times of simulation to reflect its regularity. It can be seen from Figs. 21(a), 22(a), 23(a), and 24(a), for example, from $[6, 5]$ to $[0, 0]$, when $N_p = 3$, the tracking effect is the best, and the MAD value of the tracking is the smallest, which is 0.0383 m, 0.0508 m, 0.0946 m and 0.0508 m, as shown in Figs. 21(b), 22(b), 23(b), and 24(b). Conversely, the path control error is too large or calculated path deviates from the normal track when $N_p = 1$ or greater than 13. NAN is used to indicate failure.

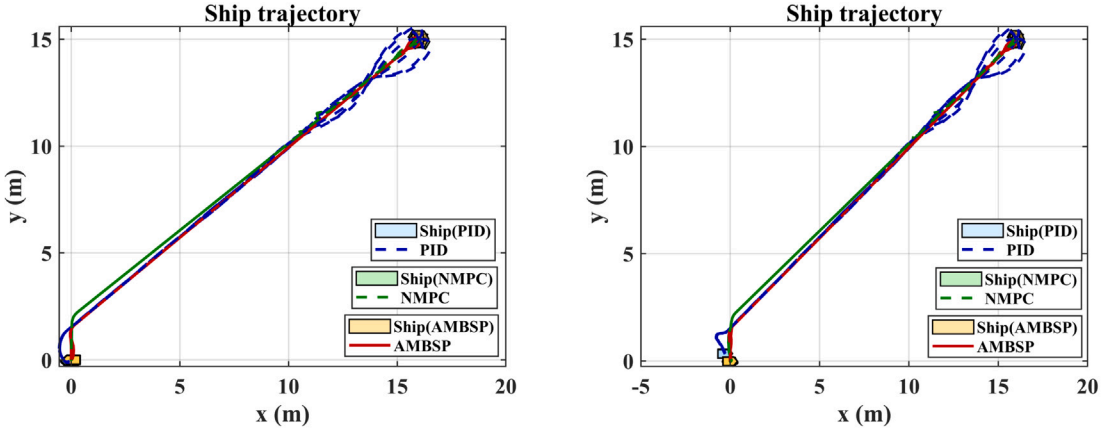
For undocking, take one of the paths as an example, such as from $[0, 0]$ to $[8, 4]$. The test results are as shown in Figs. 25, 26, 27 and 28. It can also be concluded that when $N_p = 3$, the MAD is the smallest, that is 0.0341 m, 0.0295 m, 0.0394 m and 0.0092 m. Similarly, path control is unsuccessful when $N_p = 1$ or greater than 14.

It can be seen from the above situation that the introduction of the prediction idea has a great effect on the ship docking and undocking control. It is not that the larger the prediction steps, the more effective. When the algorithm has no prediction or low degree of prediction, the system cannot control the future situation. Therefore, in this case, it is impossible to predict the control value in advance, and large error or deviation will occur. If the prediction degree is too high, it will have a great impact on the current control of the ship, which will also lead to inaccurate judgment. Through the simulation test, it is concluded that when the number of prediction steps is around three steps, the effect of prediction and control is balanced, and the error is minimal.

(2) Analysis of λ on the results.

The design of this study is divided into two stages for docking control. In the first stage, The starting point of the ship is the current position point, and the target point of the first stage is $\lambda \cdot L$ above the endpoint. This path is the closest distance compared with other paths.

In the second stage, the position of $\lambda \cdot L$ is selected to adjust the attitude. The main purpose is to ensure the ship has enough



(a) Comparison of long-distance port docking

(b) Comparison of long-distance starboard docking

(c) Comparison of long-distance bow docking

(d) Comparison of long-distance stern docking

Fig. 33. Comparison of long-distance docking effect between PID, NMPC and AMBS-P algorithm with long-distance.

Table 3
Set different path for docking.

D_{path}	Port docking MAD (m)	Starboard docking MAD (m)	Bow docking MAD (m)	Stern docking MAD (m)
$2L$	0.0383	0.0508	0.0946	0.0508
$1.5L$	0.0242	0.0466	0.0430	0.0344
L	0.0350	0.0523	0.0512	0.0429

Table 4
Set different $v_a(0)$ ($v_x(0)$) for docking.

$v_x(0)$ (m/s)	Port docking MAD (m)	Starboard docking MAD (m)	Bow docking MAD (m)	Stern docking MAD (m)
0.1	0.0300	0.0512	0.0591	0.0345
0.5	0.0479	0.0612	0.0717	0.057
1	0.1095	0.1447	0.1429	0.1140
1.5	0.2531	0.2508	0.2739	0.2929

space for attitude change. During docking, there are many divisions of ship zone 2, as shown in Fig. 29. Suppose the length of path ② in Fig. 29 is the $2L$. When $\lambda > 2$, such as the path ①, is not the shortest path, so it is not considered from the perspective of efficiency. The distance of path ③ is shorter than that of path ②, which is more efficient in principle, but there may be insufficient space to adjust the ship attitude to the predetermined position.

On the basis of setting $N_p = 3$, the error of different paths is calculated, and the trajectory MAD is shown in Table 3. Table 3 shows that when the path setting is $1.5L$, the ship attitude adjustment effect is better. Therefore, it is a safe value for the ship to adopt $\lambda = 1.5$.

(3) Analysis of $v_a(0)$ on the results

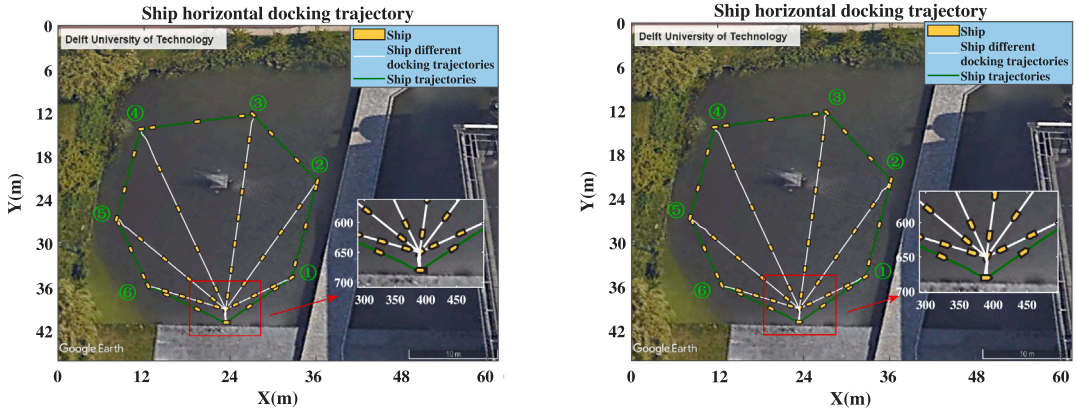
The $v_a(0)$ represents the velocity when the ship executes the docking order, which is not zero in general. Take setting v_x direction with value as an example, besides $N_p = 3$, $D_{path} = 1.5L$, the simulation are shown in Fig. 30 and Table 4. According to the test results, with the increase of the initial ship velocity, the autonomous docking effect is still good. This shows that the algorithm has certain robustness to the change of velocity.

(4) Analysis of $\tau_{distrub}$ on the result

In this paper, white noise is used to simulate environmental disturbances, that is $\tau_{distrub} = Coe \cdot randn(3,1)$. Among it, Coe is disturbances coefficient, $randn(\cdot)$ is the pseudo-random white noise of standard normal distribution. According to the above analysis, the simulation is under the condition of $N_p = 3$, $D_{path} = 1.5L$, $v_a(0) = [0.2; 0; 0]$. It can be seen from Fig. 31 and Table 5 that with the increase of disturbance, the calculation effect of the algorithm is good. It shows that the algorithm has certain robustness against the change of environment.

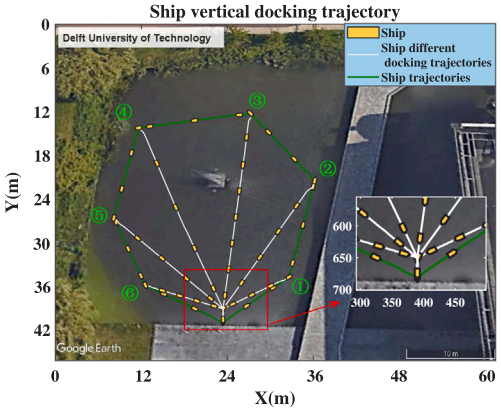
(5) Analysis of calculation efficiency on the results

In practical application, under the premise of pursuing calculation efficiency, a certain degree of accuracy needs to be guaranteed. The control variable method is used to test the efficiency and error of the algorithm, as shown in Table 6. From the comprehensive comparison of the Table 6 and actual experience, it can be seen that the average calculation velocity within

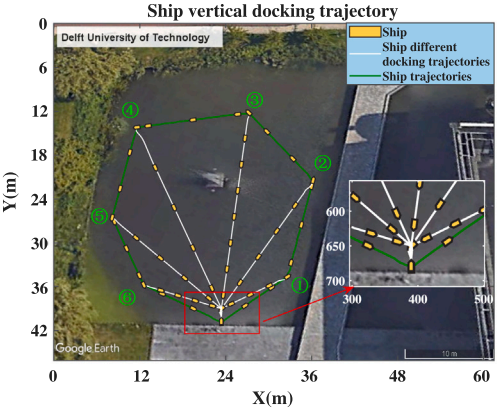


(a) Port docking in the real environment

(b) Starboard docking in the real environment



(c) Bow docking in the real environment



(d) Stern docking in the real environment

Fig. 34. Docking in the real environment. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
Set different $\tau_{distrub}$ for docking.

Coe	Port docking MAD (m)	Starboard docking MAD (m)	Bow docking MAD (m)	Stern docking MAD (m)
1	0.0510	0.0749	0.0840	0.0641
2	0.0668	0.0821	0.1003	0.0676
3	0.0866	0.0967	0.1016	0.0724
4	0.0920	0.0990	0.1083	0.0761
5	0.0962	0.1153	0.1122	0.0948

$0.1s - 0.2s$ is appropriate. At this time, the search parameter is that the $n = 5$ and $K = 40$.

5.3.2. Results comparison

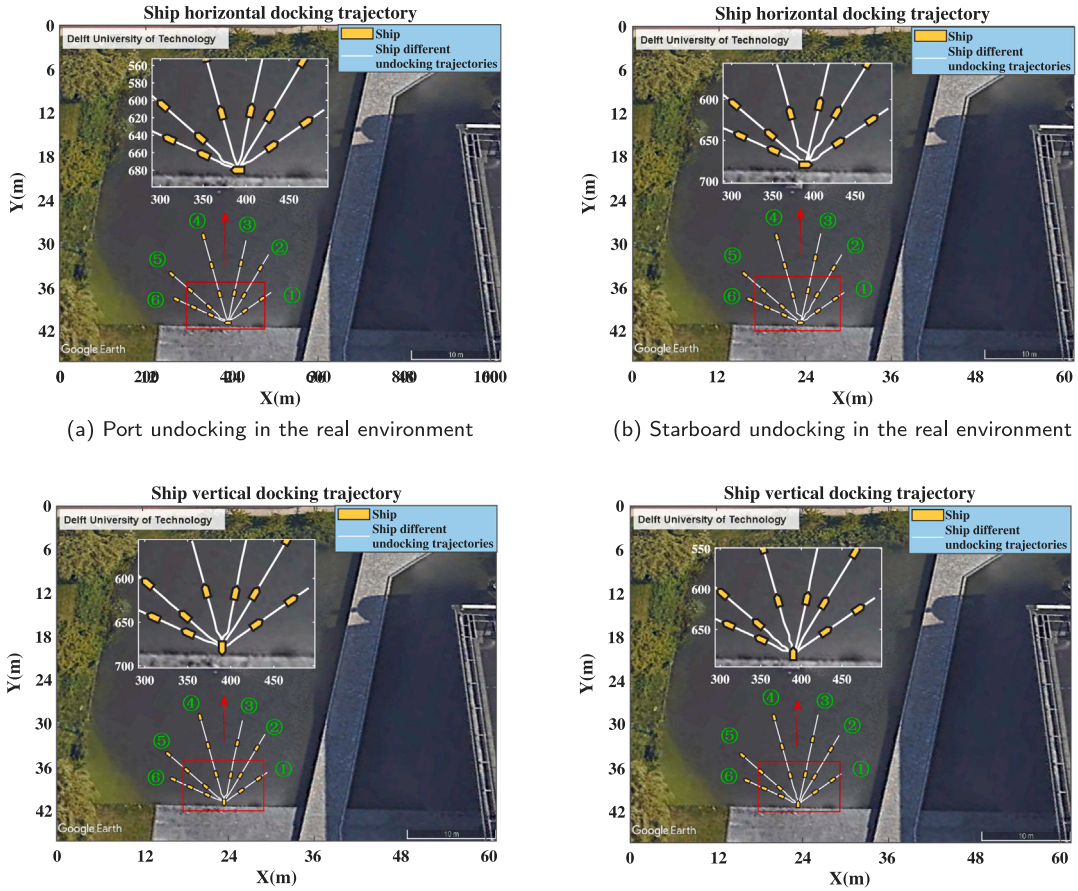
This algorithm is different from the NN algorithm in training, so it is not compared with it. PID is the simplest and most common algorithm in practice. The nonlinear model predictive control (NMPC) is similar to this control algorithm. In this study, the three algorithms are compared and analyzed. The simulation is carried out under the same initial state and constraints. For PID control, several groups of PID are tried, and the parameters of better PID effect is $[K_P, K_i, K_D] = [0.61, 0, 0.3]$. For NMPC control, we adopt the "active set" method. The force constraint range of each thruster is obtained through maneuverability experiments, that is, $[f_p; f_s; f_b] = [-2.8940, 3.2373; -2.7959, 2.7959; -1.1772, 1.2753]$.

The comparison results of short-distance docking are shown in Fig. 32. In Fig. 32, the initial trajectory is $[6, 5]$ and the heading angle are $[0°, 60°, 120°, -60°, -120°, -180°]$. If the initial trajectory is long-distance docking, such as $[16, 15]$, the comparison is shown in Fig. 33.

All kinds of situations of short-distance and long-distance docking are summarized. The average values are used and the comparison is shown in Table 7. In short-distance docking case, AMBS-P has better response velocity and smaller error than PID, but PID calculation rate is faster. For long-distance docking, although the stability of PID is improved and the error is reduced, the AMBS-P algorithm is still better than the PID and NMPC algorithm. NMPC algorithm has a small deviation for the initial position change of the ship, but the MAD is not as small as AMBS-P, and the calculation rate is slower than AMBS-P. In conclusion, the AMBS-P algorithm is very good for both short-distance and long-distance docking, and calculation rate is appropriate. It has the advantages of quick response and prediction for short-distance docking and large turning.

5.3.3. Application effect of simulation in the real environment

The lake of the Delft University of Technology (TUDelft) is selected as the test environment, and the optimal control parameter $N_p = 3, \lambda = 1.5, n = 5, K = 40$. The docking and undocking simulation effects under different conditions are shown in Figs. 34 and 35. Among them, the latitude and longitude coordinates of docking and undocking are shown in Table 8. Fig. 34 shows the situation of autonomous docking of the ship, in which the green line represents the ship trajectory, and white lines represent the execution of docking instructions at each path point. Besides, Fig. 35 shows the situation of autonomous undocking of the ship. The tracking error is shown in Table 9. The average error of docking is within 0.035 m and the average error of undocking is within 0.065 m. As can be seen from Figs. 34, 35 and Table 9, the tracking effect is well.



(a) Port undocking in the real environment

(b) Starboard undocking in the real environment

(c) Bow undocking in the real environment

(d) Stern undocking in the real environment

Fig. 35. Undocking in the real environment. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 6
Different calculation efficiency for docking.

n	K	Port docking		Starboard docking		n	K	Bow docking		Stern docking	
		Times (s)	MAD (m)	Times (s)	MAD (m)			Times (s)	MAD (m)	Times (s)	MAD (m)
1	40	0.0394	0.1003	0.0395	0.0955	1	40	0.0343	0.1165	0.0397	0.0854
5	40	0.1298	0.0341	0.1340	0.0400	5	40	0.1145	0.0571	0.1281	0.0490
10	40	0.2400	0.0322	0.2559	0.0375	10	40	0.2521	0.0544	0.2626	0.0360
15	40	0.3566	0.0300	0.339	0.0362	15	40	0.4095	0.0534	0.3610	0.0349
20	40	0.4553	0.0304	0.4268	0.0272	20	40	0.5431	0.0497	0.4586	0.0276
5	10	0.0049	0.0955	0.0435	0.1517	5	10	0.0415	0.1123	0.0687	0.0979
5	20	0.0654	0.0898	0.0746	0.0887	5	20	0.0711	0.0777	0.0755	0.0683
5	30	0.0941	0.0446	0.0960	0.0583	5	30	0.0946	0.0599	0.1015	0.0562
5	40	0.1346	0.0370	0.1219	0.0541	5	40	0.1299	0.0586	0.1232	0.0411
5	50	0.1455	0.0377	0.1875	0.0378	5	50	0.1711	0.0569	0.1762	0.0343
5	60	0.1676	0.0284	0.1962	0.0278	5	60	0.2143	0.0552	0.1946	0.0301

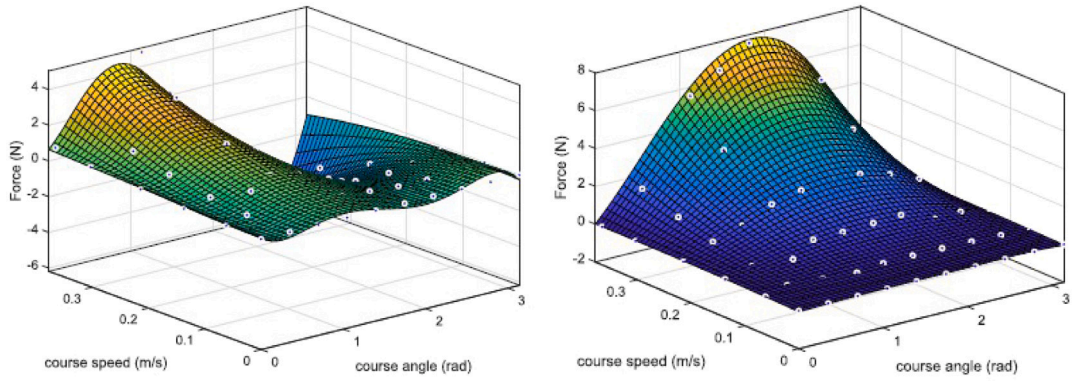
Table 7
Comparison docking results based on PID, NMPC and AMBS-P algorithm.

Comparison metrics	Short-distance docking			Long-distance docking		
	PID	NMPC	AMBS-P	PID	NMPC	AMBS-P
MAD (m)	0.5029	0.1440	0.0333	0.4440	0.1824	0.0382
Maximum overshoot (m)	1.0602	0.3176	0.2312	1.0116	0.3353	0.3216
Per-step calculation time (s)	0.0188	1.0723	0.1179	0.0014	1.9452	0.1108

6. Conclusions and future research

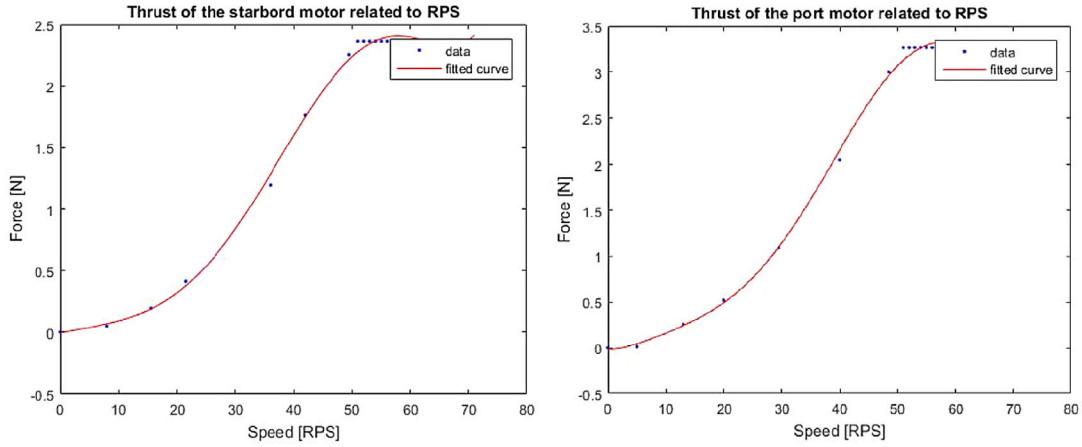
This study focuses on the application of a nature-inspired algorithm (AMBS-P) in the autonomous docking and undocking of ships. Firstly, the AMBS-P algorithm is proposed, and then the control effect of this algorithm on ship docking and undocking is verified. Based on the above analysis, the following points can be summarized:

(1) The simulation without ship drag shows that the AMBS-P algorithm can meet the requirements of autonomous docking under ideal conditions, which proves the correctness of the algorithm. The test with ship drag shows that the AMBS-P algorithm applies



(a) Longitudinal drag forces in the body-fixed frame. (b) Lateral drag forces in the body-fixed frame.

Fig. 36. Tito-Neri drag force.



(a) Characteristic curve of the starboard side motor (b) Characteristic curve of the port side motor

Fig. 37. Characteristic curve of the motor.

Table 8
Latitude and longitude coordinates of docking and undocking.

Docking	Coordinates	Undocking	Coordinates
Starting ①	52.002115, 4.371308	Starting point	52.00216, 4.371196
Starting ②	52.00213, 4.371473	Terminal ①	52.002128, 4.371268
Starting ③	52.002218, 4.371534	Terminal ②	52.002147, 4.371328
Starting ④	52.00232, 4.371437	Terminal ③	52.002175, 4.371347
Starting ⑤	52.002308, 4.371283	Terminal ④	52.002216, 4.371315
Starting ⑥	52.00225, 4.37120	Terminal ⑤	52.002233, 4.371237
Terminal point	52.00216, 4.371196	Terminal ⑥	52.002219, 4.371199

Table 9
The docking and undocking errors in the real environment.

Motion type	The docking and undocking RMSE with different starting (ending) poses			
	Port	Starboard	Bow	Stern
Docking (m)	0.0245	0.0221	0.0674	0.0254
Undocking (m)	0.0869	0.0258	0.1036	0.0418

to the autonomous docking of ships in practice, which shows the applicability of the algorithm.

(2) The analysis of the AMBS-P algorithm shows that the algorithm has convergence.

(3) The introduction of prediction theory is of great significance to ship docking control, which can make up for the lack of prediction in the algorithm.

(4) The advantage of the AMBS-P algorithm is that it can be well realized no matter how far or near it is, especially in the large turning.

(5) The application of AMBS-P algorithm is not limited to the object model, and the control rate does not depend on the constraints set by the controlled object model, so it has certain universality. It can not only be used in the independent docking and undocking of ships but also can be used in other control aspects such as ship track tracking, course control, and other fields.

In the future, the AMBS-P algorithm itself could be further improved. Other algorithms may be combined with AMBS-P algorithm to obtain more rapid and accurate results, which can be used in ship motion control research or other fields.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Ship symbols

See Table 10.

Appendix B. Tito-Neri drag force and thruster of motors related to RPS

See Figs. 36 and 37

Table 10
Ship symbols.

Symbol	Meaning	Symbol	Meaning
Bt	Bow thruster coordinates	m_b	Mass of the ship
C_{RB}	Coreolis centripedal matrix	τ_{thrust}	Force vector of the thrusters
C_A	Added coreolis centripedal matrix	τ_{drag}	Drag vector of ship
CoG	Center of gravity	$\dot{v}_a$	Acceleration vector
L	Length of ship	Pt	Port thruster coordinates
M_{RB}	Mass matrix	St	Starboard thruster coordinates
M_A	Added mass matrix	w	Width of ship

References

Ahmed, Y.A., 2012. Automatic ship berthing using artificial neural network based on virtual window concept in wind condition. In: IFAC Symposium on Control in Transportation Systems. pp. 286–291.

Ahmed, Y.A., Hasegawa, K., 2013. Implementation of automatic ship berthing using artificial neural network for free running experiment. IFAC Proc. Vol. 46, 25–30.

Ahmed, Y.A., Hasegawa, K., 2015. Artificial neural network based automatic ship berthing combining pd controlled side thrusters — a combined controller for final approaching to berth. In: International Conference on Control Automation Robotics & Vision. pp. 1304–1309.

Bruggink, D.J.J.B., Cremer, Q.C., Groenewegen, R.R., Klokgieters, A.G.C., 2018. Differentiation of Maneuvering Coefficients for Scaled Model Vessels. Technical Report.

Bui, V.P., Kim, Y.B., Yong, W.C., Kawai, H., 2009. A study on automatic ship berthing system design. In: International Conference on Networking, Sensing and Control. pp. 181–184.

Chen, P., Shi, G.Y., Liu, S., Zhang, Y.Q., 2018. Decision support based on artificial fish swarm for ship collision avoidance from ais data. In: 2018 International Conference on Machine Learning and Cybernetics. pp. 31–36.

Damen Shipyards, 2018. Stan-tug-3011. https://products.damen.com/fr-fr/ranges/stan-tug%5C/stan-tug-3011.

Dong, Z., Bao, T., Zheng, M., Yang, X., Song, L., Mao, Y., 2019. Heading control of unmanned marine vehicles based on an improved robust adaptive fuzzy neural network control algorithm. IEEE Access 7, 9704–9713.

Fang, M.C., Zhuo, Y.Z., Lee, Z.Y., 2010. The application of the self-tuning neural network pid controller on the ship roll reduction in random waves. Ocean Eng. 37, 529–538.

Fossen, T.I., 2002. Marine control systems: guidance, navigation and control of ships, rigs and underwater vehicles. In: Marine Cybernetics. Trondheim, Norway.

Haseltalab, A., Negenborn, R.R., 2019. Model predictive maneuvering control and energy management for all-electric autonomous ships. Appl. Energy 251, 1–27.

Hashim, F.A., Houssein, E.H., Mabrouk, M.S., Al-Atabany, W., Mirjalili, S., 2019. Henry gas solubility optimization: A novel physics-based algorithm. Future Gener. Comput. Syst. 101, 646–667.

Jiang, X., Li, S., 2017. Bas: Beetle antennae search algorithm for optimization problems. arXiv:arXiv:1710.10724.

Larrazabal, J.M., Peñas, M.S., 2016. Intelligent rudder control of an unmanned surface vessel. Expert Syst. Appl. 55, 106–117.

Li, W., Ma, W., 2016. Simulation on vessel intelligent collision avoidance based on artificial fish swarm algorithm. Pol. Marit. Res. 23, 138–143.

Liao, Y., Jia, Z., Zhang, W., Jia, Q., Li, Y., 2019. Layered berthing method and experiment of unmanned surface vehicle based on multiple constraints analysis. Appl. Ocean Res. 86, 47–60.

Liu, G.Y., Hou, Y.B., Luo, Y., Li, D., 2017. Genetic algorithm's application for optimization of pid parameters in dynamic positioning vessel. In: MATEC Web of Conferences. p. 00153.

Liu, H.D., Rong, D., Zhang, L.Y., 2016. The application research for ship collision avoidance with hybrid optimization algorithm. In: IEEE International Conference on Information & Automation. pp. 760–767.

Namkyun, I.M., 2007. All direction approach automatic berthing controller using ann. IFAC Proc. Vol. 40, 300–304.

Shin, J., Dong, J.K., Lee, Y., 2017. Adaptive path-following control for an unmanned surface vessel using an identified dynamic model. IEEE/ASME Trans. Mechatronics 22, 1143–1153.

Shuai, Y., Li, G., Cheng, X., Skulstad, R., Xu, J., Liu, H., Zhang, H., 2019. An efficient neural-network based approach to automatic ship docking. Ocean Eng. 191, 106514.

Tomera, Miroslaw, 2014. Ant colony optimization algorithm applied to ship steering control. Procedia Comput. Sci. 35, 83–92.

Tomera, M., 2015. Swarm intelligence applied to identification of nonlinear ship steering model. In: IEEE International Conference on Cybernetics. pp. 133–139.

Tran, V.L., Im, N., 2012. A study on ship automatic berthing with assistance of auxiliary devices. Int. J. Nav. Archit. Ocean Eng. 4, 199–210.

Wang, X.G., Jian, Z.Z., 2012. Identification of ship manoeuvring response model based on fruit fly optimization algorithm. J. Dalian Marit. Univ. 38, 1–4.

Wang, L., Wu, Q., Liu, Jialun, e.a., 2019. Ship motion control based on ambps-pid algorithm. IEEE Access 7, 183656–183671.