Hindawi
Complexity
Volume 2020, Article ID 3095426, 15 pages
https://doi.org/10.1155/2020/3095426

Research Article

Local Path Planning for Unmanned Surface Vehicle Collision Avoidance Based on Modified Quantum Particle Swarm Optimization

Guoqing Xia, Zhiwei Han, Bo Zhao, and Xinwei Wang

College of Automation, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Zhiwei Han; hanzhiwei@hrbeu.edu.cn

Received 5 November 2019; Revised 30 January 2020; Accepted 5 March 2020; Published 13 April 2020

Academic Editor: Matilde Santos

An unmanned surface vehicle (USV) plans its global path before the mission starts. When dynamic obstacles appear during sailing, the planned global path must be adjusted locally to avoid collision. This study proposes a local path planning algorithm based on the velocity obstacle (VO) method and modified quantum particle swarm optimization (MQPSO) for USV collision avoidance. The collision avoidance model based on VO not only considers the velocity and course of the USV but also handles the variable velocity and course of an obstacle. According to the collision avoidance model, the USV needs to adjust its velocity and course simultaneously to avoid collision. Due to the kinematic constraints of the USV, the velocity window and course window of the USV are determined by the dynamic window approach (DWA). In summary, local path planning is transformed into a multiobjective optimization problem with multiple constraints in a continuous search space. The optimization problem is to obtain the USV's optimal velocity variation and course variation to avoid collision and minimize its energy consumption under the rules of the International Regulations for Preventing Collisions at Sea (COLREGs) and the kinematic constraints of the USV. Since USV local path planning is completed in a short time, it is essential that the optimization algorithm can quickly obtain the optimal value. MQPSO is primarily proposed to meet that requirement. In MQPSO, the efficiency of quantum encoding in quantum computing and the optimization ability of representing the motion states of the particles with wave functions to cover the whole feasible solution space are combined. Simulation results show that the proposed algorithm can obtain the optimal values of the benchmark functions and effectively plan a collision-free path for a USV.

1. Introduction

An unmanned surface vehicle (USV) is an autonomous marine vehicle that has emerged as a viable tool to perform tasks that are dangerous or unsuitable for manned vessels. Planning the path of a USV is important as regards safety and efficiency [1]. USV path planning can be divided into two stages: global path planning based on prior environmental information and local path planning based on sensor information [2]. Since unexpected conditions can occur during USV sailing, such as other vessels and obstacles on its path, the global path must be adjusted locally to avoid collision. As part of local path planning, USV collision avoidance for moving obstacles is carried out online based on real-time information of shipborne sensors such as radars. This paper studies local path planning for USV collision avoidance.

All USVs must have the ability to autonomously avoid obstacles, including land masses, watercraft, low-hanging obstacles, submerged shallow obstacles, interface obstacles, and submerged obstacles for USVs that tow systems [3].

There are some results for local path planning in the literature. An obstacle-avoidance solution based on rules and criteria was proposed, in which the collision avoidance rules and criteria are determined according to the target's collision priority [4]. A method to calculate the collision probabilities of vessels was established under the constraints of mission space and number of vessels [5]. An alternative method assessed the collision risk for surface ships in close-range encounters that is compliant with the International Regulations for Preventing Collisions at Sea (COLREGs) [6].

The velocity obstacle (VO) method is a local path planning method that considers the velocity of obstacles [7]. The method has been used to study the case of a single vehicle avoiding some obstacles moving along known linear trajectories. Using the relative velocity between the vehicle and each obstacle, the dynamic problem is transformed into a number of static problems. These are converted to a single problem by vector transformation, and a set of velocity vectors are calculated to avoid obstacles. To avoid collisions, both the current position and the velocities of vehicles are used to compute their future trajectories [8]. An improved VO method was proposed to choose an optimal velocity by introducing a cost function, which consists of a pass time and clearance [9]. A conflict-free maneuver that aims to avoid obstacles while limiting the deviation from the original route was generated by the selective VO method [10]. A finite-time velocity obstacle (FVO) method was used to plan the motion of a mobile robot and complete a crossing through an unknown environment [11]. Based on the nonconvex nature of the FVO constraints, a parabolic function and the approximated function were used to calculate the optimal next-step velocity of a mobile robot to ensure collision-free motion.

In [12, 13], considering the kinematic constraints of the robot, a dynamic window approach (DWA) was proposed, which calculates the velocity that the robot can reach within a given time interval. The reachable velocities constitute a set of velocity space, known as dynamic window. The path planner selects the optimal feasible velocity vector from the dynamic window by the objective functions. DWA was proposed to plan an energy efficient local path for navigation of omnidirectional battery-powered mobile robots in dynamic environments with incorporating a cost function based on energy consumption [14].

The COLREGs are maritime traffic regulations established by the International Maritime Organization (IMO) to prevent collisions between vessels. The COLREGs specify several collision situations during a voyage: head-on, crossing, and overtaking [15]. It is reasonable for a USV to follow COLREGs during navigation. Some researchers have applied COLREGs to actual collision avoidance. When a ship was detected, the range of its position during a given time frame was estimated, and the path planning system, considering COLREGs, produced a new, safe, and smooth path in real time [16]. Based on the motion velocity model and COLREGs, a reverse eccentric expansion method was designed to deal with dynamic obstacles [17]. A multi-objective optimization approach for path planning of an autonomous surface vehicle combined COLREGs with good seaman's practice and stratified the objectives [18].

Some intelligent optimization algorithms have been applied to the path planning of vehicles. These include ant colony optimization [19], genetic algorithms [20], fuzzy algorithms [21], and particle swarm optimization (PSO) [22]. The algorithms have global optimization ability and overcome the shortcomings of many traditional path planning algorithms. As a method to solve optimization problems, PSO has been developed rapidly in recent years and widely used in many fields, such as combinatorial optimization [23], scheduling problems [24], and neural networks [25].

With the development of quantum theory, to combine quantum computing with intelligent optimization algorithms has been proposed. Quantum-behaved particle swarm optimization (QBPSO) was first proposed to improve the global search ability of the original PSO [26]. The position of a particle in PSO was sampled by the quantum delta potential well model around the mean best position [27]. The basic principle of QBPSO is to regard the optimization process as a moving process of particles in the potential field of quantum mechanics to the lowest point of potential energy (the center of the potential well) [28]. Compared with PSO, the iterative equation of QBPSO needs no velocity vectors of particles, is easier to tune, and requires fewer parameters to implement [29].

Quantum mechanics principles and evolutionary computing methods were first combined in [30]. A quantum bit and superposition of states were used to solve the knapsack problem by a quantum-inspired evolutionary algorithm (QEA) [31]. Based on the QEA with a quantum rotation gate strategy, an adaptive evolution-based quantum-inspired evolutionary algorithm (AEQEA) introduced an adaptive evolution mechanism [32]. A quantum ant colony algorithm was used to plan the global path of a USV [33, 34]. A quantum evolutionary algorithm was integrated into particle swarm optimization [35]. The positions of particles were encoded by probability amplitudes of quantum bits, and the movements of particles were performed by quantum rotation gates to achieve particle searching. Quantum-inspired particle swarm optimization (QIPSO) has a stronger search ability and quicker convergence speed, benefiting from quantum computing theory, self-adaptive probability selection, and chaotic sequence mutation [36].

This study proposes a local path planning algorithm for USV collision avoidance based on modified quantum particle swarm optimization (MQPSO). The main contributions are as follows:

(1) Based on the model proposed in [7, 17], a USV collision avoidance model is established by the VO method considering the constraints of USV kinematic model, the rules of COLREGs, and the uncertainty of obstacle's velocity. The collision avoidance model not only considers the velocity and course of the USV but also handles the variable velocity and course of an obstacle. According to the collision avoidance model, the USV needs to adjust its velocity and course simultaneously to avoid collision and minimize its energy consumption. Due to the kinematic constraints of the USV, the velocity window and course window of the USV are determined by DWA. Therefore, the local path planning for USV collision avoidance is transformed into a multiobjective optimization problem with multiple constraints in a continuous search space.

(2) In order to solve the optimal value in a continuous search space that satisfies a series of constraints, an optimization algorithm is needed. Because the particles in QBPSO are encoded by real numbers in [26–29], the search efficiency is low. The motion states of particles are represented by a vector of velocity and position [36]. To avoid premature convergence and local optima, a new optimization algorithm called modified quantum particle swarm optimization (MQPSO), combining QBPSO and quantum computing, is proposed in this study. The quantum bits (Q-bits) are used to encode the positions of particles, and the motion states of particles are determined by wave functions. A quantum nongate is introduced to increase the population's diversity. Since USV local path planning is completed in a short time, it is essential that the optimization algorithm can quickly obtain the optimal value. MQPSO meets that requirement.

The remainder of this study is organized as follows. In Section 2, the model of USV local path planning is established, and the USV kinematic model, the rules of COLREGs, the VO method, the velocity window and course window of the USV, the collision avoidance model with considering uncertainties, and the cost function of path planning are described. In Section 3, the principles of MQPSO are provided, and it is applied to plan a USV local path. In Section 4, simulations for testing the performance of MQPSO and USV local path planning using MQPSO are presented. Conclusions are provided in Section 5.

2. Problem Statement

In this section, the USV kinematic model, rules of COLREGs, velocity obstacle method, and cost function of path planning are introduced.

2.1. USV Kinematic Model. A three-degree-of-freedom model is usually adopted for USV path planning. The kinematic equation can be expressed according as [16, 37]

$$\dot{\eta} = \mathbf{R}(\psi)\mathbf{v} + \mathbf{V}_c, \tag{1}$$

where $\boldsymbol{\eta} = [x, y, \psi]^{\mathrm{T}} \in \mathbb{R}^3$ is the vector including the positions and heading angle of the USV in the north-east coordinate system. $\mathbf{v} = [u_r, v_r, r]^{\mathrm{T}}$ is the relative velocity between the USV and the ocean current in the body-fixed reference frame, where $u_r = u - u_c$ and $v_r = v - v_c$. u and v are the velocities in surge and sway, respectively, of the USV. Let $\mathbf{V}_c = [V_{cx}, V_{cy}, 0]^{\mathrm{T}} \in \mathbb{R}^3$ represent the velocity of the ocean current in the north-east coordinate system. The body-fixed ocean current velocities (u_c, v_c) and north-east current velocities (V_{cx}, V_{cy}) satisfy $[u_c, v_c]^{\mathrm{T}} = \mathbf{R}^{\mathrm{T}}(\psi)[V_{cx}, V_{cy}]^{\mathrm{T}}$, and the rotation matrix $\mathbf{R}(\psi)$ is

$$\mathbf{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

Therefore, the USV moving in surge, sway, and yaw can be described as

$$\begin{cases} \dot{x} = u_r\cos\psi - v_r\sin\psi + V_{cx}, \\ \dot{y} = u_r\sin\psi - v_r\cos\psi + V_{cy}, \\ \dot{\psi} = r. \end{cases} \tag{3}$$

The relative velocity of the USV is

$$U = \sqrt{u_r^2 + v_r^2}. \tag{4}$$

The USV at position (x, y) moving in the sea has velocity

$$v_{\mathrm{usv}} = \sqrt{\dot{x}^2 + \dot{y}^2}. \tag{5}$$

2.2. Collision Avoidance Model

2.2.1. VO-Based Modeling. The collision avoidance model is established as follows. Figure 1 shows the schematic of obstacle-avoidance modeling. Take the current position of the USV, that is, point V, whose coordinates are (x_v, y_v), as the origin of the north-east coordinate system, and point O, whose coordinates are (x_o, y_o), as the current position of the obstacle. The USV and the obstacle are moving. $\mathbf{v}_{\mathrm{usv}}$ and $\mathbf{v}_{\mathrm{obs}}$ are the velocities of the USV and obstacle, respectively. Thus, $\Delta\mathbf{v} = \mathbf{v}_{\mathrm{usv}} - \mathbf{v}_{\mathrm{obs}}$ is the velocity difference between the USV and the obstacle. α is the angle from the X-axis to $\mathbf{v}_{\mathrm{usv}}$, which is represented as the course angle of the USV. β is the angle from the X-axis to $\mathbf{v}_{\mathrm{obs}}$. θ is the angle from the X-axis to the line VO. ϕ is the angle from $\Delta\mathbf{v}$ to $\mathbf{v}_{\mathrm{usv}}$. Although the shape of the obstacle is unknown, the obstacle's collision area can be expanded to a circle. With the point O as the center and r_{obs} as the radius, the circle of the collision area is constructed and represented by ⊙O. The lines VM and VN are two tangents of the collision area ⊙O. γ is the angle from the line VO to $\Delta\mathbf{v}$. μ is the angle from the line VO to VM, which is called the minimum safety angle.

Based on the current velocity of the obstacle, to avoid collision, the vector $\Delta\mathbf{v}$ cannot be in the polygon VMON in the next time interval. Therefore, $\gamma + \Delta\gamma > \mu$ is the collision avoidance criterion.

The angle γ is related to $\Delta\mathbf{v}$. $\Delta\mathbf{v}$ is decomposed into velocity components $\Delta\mathbf{v}_o$, directed to the obstacle, and $\Delta\mathbf{v}_e$, perpendicular to $\Delta\mathbf{v}_o$. Thus, Δv_o and Δv_e are represented as

$$\begin{cases} \Delta v_e = v_{\mathrm{usv}}\sin(\alpha - \theta) - v_{\mathrm{obs}}\sin(\beta - \theta), \\ \Delta v_o = v_{\mathrm{usv}}\cos(\alpha - \theta) - v_{\mathrm{obs}}\cos(\beta - \theta). \end{cases} \tag{6}$$

As shown in Figure 1, the angle γ is represented as

$$\gamma = \arctan\left(\frac{v_{\mathrm{usv}}\sin(\alpha - \theta) - v_{\mathrm{obs}}\sin(\beta - \theta)}{v_{\mathrm{usv}}\cos(\alpha - \theta) - v_{\mathrm{obs}}\cos(\beta - \theta)}\right). \tag{7}$$

The difference of γ in the time interval Δt can be calculated as

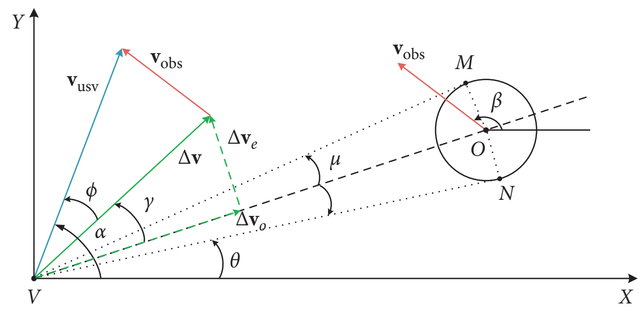

Figure 1: Schematic of collision avoidance modeling.

$$\Delta\gamma = \frac{-v_{\text{obs}}\sin(\alpha-\beta)}{v_{\text{usv}}^2+v_{\text{obs}}^2+2v_{\text{usv}}v_{\text{obs}}\cos(\alpha-\beta)}\Delta v_{\text{usv}} + \frac{v_{\text{usv}}^2-v_{\text{usv}}v_{\text{obs}}\cos(\alpha-\beta)}{v_{\text{usv}}^2+v_{\text{obs}}^2+2v_{\text{usv}}v_{\text{obs}}\cos(\alpha-\beta)}\Delta\alpha + \frac{-v_{\text{usv}}\sin(\alpha-\beta)}{v_{\text{usv}}^2+v_{\text{obs}}^2+2v_{\text{usv}}v_{\text{obs}}\cos(\alpha-\beta)}\Delta v_{\text{obs}} + \frac{v_{\text{obs}}^2-v_{\text{usv}}v_{\text{obs}}\cos(\alpha-\beta)}{v_{\text{usv}}^2+v_{\text{obs}}^2+2v_{\text{usv}}v_{\text{obs}}\cos(\alpha-\beta)}\Delta\beta. \tag{8}$$

Since there is a relationship between $\mathbf{v}_{\text{usv}}$, $\mathbf{v}_{\text{obs}}$, and $\Delta\mathbf{v}$, as shown in Figure 2, the following equations hold:

$$\begin{cases} \dfrac{v_{\text{obs}}}{\sin\phi} = \dfrac{\Delta v}{\sin|\alpha-\beta|}, \\ \dfrac{v_{\text{obs}}}{\sin\phi} = \dfrac{v_{\text{usv}}}{\sin(|\alpha-\beta|+\phi)}, \\ v_{\text{usv}} = v_{\text{obs}}\cos(\alpha-\beta)+\Delta v\cos\phi, \\ v_{\text{obs}} = v_{\text{usv}}\cos(\alpha-\beta)-\Delta v\cos(|\alpha-\beta|+\phi), \\ \Delta v^2 = v_{\text{usv}}^2+v_{\text{obs}}^2-2v_{\text{usv}}v_{\text{obs}}\cos(\alpha-\beta). \end{cases} \tag{9}$$

Substituting (9) in (8) yields

$$\Delta\gamma = -\frac{\sin\phi}{\Delta v}\Delta v_{\text{usv}} + \frac{v_{\text{usv}}\cos\phi}{\Delta v}\Delta\alpha + \frac{\sin(|\alpha-\beta|+\phi)}{\Delta v}\Delta v_{\text{obs}} - \frac{v_{\text{obs}}\cos(|\alpha-\beta|+\phi)}{\Delta v}\Delta\beta, \tag{10}$$

where α, β, and v_{obs} are measured by the shipborne sensors. $\Delta\mathbf{v}_{\text{obs}}$ and $\Delta\beta$ can also be measured. μ and ϕ are calculated as follows:

$$\mu = \arcsin\left(\frac{r_{\text{obs}}}{\sqrt{(x_v-x_o)^2+(y_v-y_o)^2}}\right), \tag{11}$$

$$\phi = \arcsin\left(\frac{v_{\text{obs}}\sin|\alpha-\beta|}{|v_{\text{usv}}-v_{\text{obs}}|}\right). \tag{12}$$

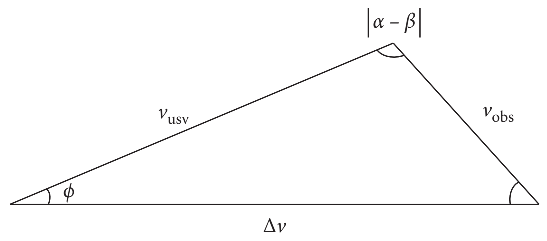

Figure 2: Positional relationship between $\mathbf{v}_{\text{usv}}$, $\mathbf{v}_{\text{obs}}$, and $\Delta\mathbf{v}$.

As shown in Figure 1, if $\gamma<\mu$, then the USV will collide with the obstacle, so $\Delta\gamma$ must be adjusted to satisfy $\gamma>\mu$, that is,

$$|\gamma+\Delta\gamma|>\mu. \tag{13}$$

Therefore, the adjustment range of $\Delta\gamma$ is

$$\begin{cases} \Delta\gamma>\mu-\gamma, & \gamma>0, \\ \Delta\gamma<-\mu-\gamma, & \gamma\le 0. \end{cases} \tag{14}$$

It can be concluded from (8) and (10) that $\Delta\gamma$ is related to Δv_{usv}, $\Delta\alpha$, Δv_{obs}, and $\Delta\beta$. But only Δv_{usv} and $\Delta\alpha$ are control variables of the USV. Thus, the local path planning for USV collision avoidance is transformed to solve the velocity variation Δv_{usv} and course variation $\Delta\alpha$ of the USV.

As mentioned above, the USV adjusts its velocity v_{usv} and the course α within a given time interval Δt to avoid collision. However, due to the constraints of USV kinematics, the velocity and course of the USV will not change much in a given time interval. DWA was introduced to handle the kinematic constraints of the USV. The DWA considers the kinematics and calculates the velocity and course that the USV can achieve within a given time window, which is called the dynamic window V_{DW}, respectively:

$$V_{\text{DW}} = \left\{(v_{\text{usv}},\alpha) \mid v_{\text{usv}} \in [v_{\text{cur}}-\Delta v_{\text{usv}}\Delta t, v_{\text{cur}}+\Delta v_{\text{usv}}\Delta t], \alpha \in [\alpha_{\text{cur}}-\Delta\alpha\Delta t, \alpha_{\text{cur}}+\Delta\alpha\Delta t]\right\}, \tag{15}$$

where v_{cur} is the current velocity of the USV and α_{cur} is the current course of the USV [12].

The velocity v_{usv} and course α reached by USV within a given Δt can be determined by (15). The elements in the set V_{DW} are continuous. Therefore, the local path planner needs to use optimization algorithm to obtain the optimized Δv_{usv} and $\Delta\alpha$ that meet the optimization objectives and constraints within the feasible continuous space of velocity and course, and then a new USV velocity vector $\mathbf{v}_{\text{usv}}$ can be obtained to avoid collision.

2.2.2. Uncertainty of the Obstacle. In actual collision avoidance, the path planner must consider various types of uncertainties. The uncertainty in USV local path planning is mainly divided into two categories: the uncertainty of sensor detection and the uncertainty of obstacle motion.

It is assumed that moving obstacles are detected and tracked by shipborne sensors. Performance characteristics of the sensors in turn affect the noise and state estimation errors for the tracked obstacle. Another uncertainty is from the imprecise motion of the moving obstacle [38]. The velocities of moving obstacle are assumed to be constant in traditional VO, but in reality they do not move with constant velocities. Considering the uncertainty of the obstacle, the velocity of the obstacle is calculated as

$$\mathbf{v}_{\text{obs}} = \overline{\mathbf{v}}_{\text{obs}} + \boldsymbol{\delta}_{\text{obs}}, \tag{16}$$

where $\overline{\mathbf{v}}_{\text{obs}}$ is the nominal velocity (i.e., the expected velocity that is estimated by the obstacle tracker) and $\boldsymbol{\delta}_{\text{obs}}$ represents the uncertainties of the obstacle's velocity [38]. We assumed that the uncertain component of the velocity lies in a set $\boldsymbol{\delta}_{\text{obs}} \in \odot\mathbf{w}_{\text{obs}}$, where $\odot\mathbf{w}_{\text{obs}}$ is a bound set and treated as a constant. Since $\Delta\mathbf{v} = \mathbf{v}_{\text{usv}} - \mathbf{v}_{\text{obs}}$, the vector $\Delta\mathbf{v}$ is also affected by the uncertainty and cannot be in the polygon VMON in the next time interval. Due to the uncertainty of $\mathbf{v}_{\text{obs}}$, the collision area $\odot O$ with the worst case uncertainty $\odot$WO is defined as

$$\odot \text{WO} = \odot O \oplus \odot \mathbf{w}_{\text{obs}}, \tag{17}$$

where $\oplus$ is the Minkowski sum operation.

Figure 3 shows the same situation as shown in Figure 2 and takes into account the uncertainties. The circle drawn with a dotted line is represented as $\odot$WO. The lines VM′ and VN′ are two tangents of the area $\odot$WO. Therefore, considering the uncertainties, the vector $\Delta\mathbf{v}$ will not be in the polygon VM′ON′ to avoid collision in the next time interval.

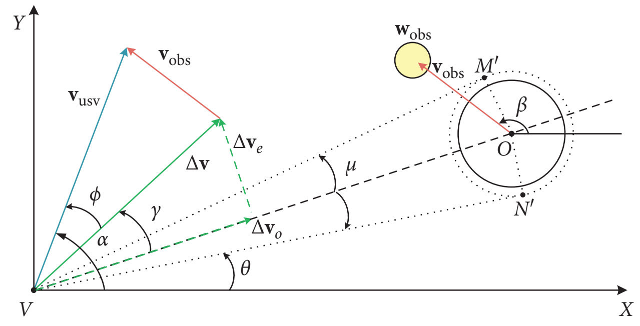

FIGURE 3: Schematic of collision avoidance modeling considering uncertainty.

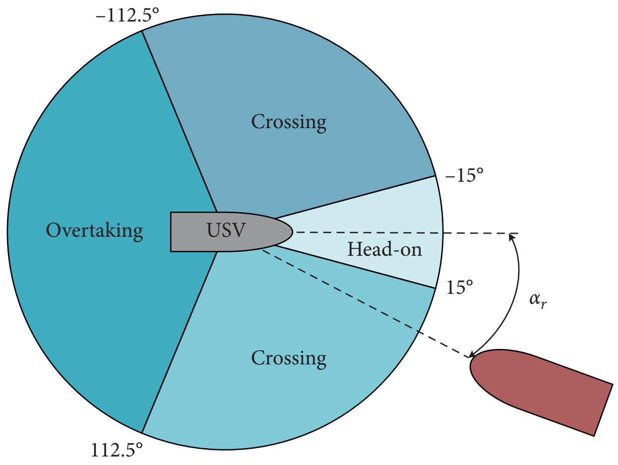

FIGURE 4: Possible encounter situations between USV and moving obstacle.

2.3. Collision Avoidance Rules Based on COLREGs. Rules 13, 14, and 15 of the COLREGs [15] make provisions for three types of collisions that may occur during the sailing of a vessel, that is, head-on, crossing, and overtaking, respectively. The rules are as follows:

(1) Overtaking: any vessel overtaking any other shall keep out of the way of the vessel being overtaken.

(2) Head-on: when two power-driven vessels are meeting on reciprocal or nearly reciprocal courses so as to involve risk of collision, each shall alter her course to starboard so that each shall pass on the port side of the other.

(3) Crossing: when two power-driven vessels are crossing so as to involve risk of collision, the vessel which has the other on her own starboard side shall keep out of the way and shall, if the circumstances of the case admit, avoid crossing ahead of the other vessel.

However, the rules are only behavioral constraints on operation and do not explicitly state the angle and scope in practical applications. In this study, we assume that there is no communication between the USV and the obstacle. Due to the special nature of the unmanned operation, the USV is regarded as the give-way vessel and the obstacle is regarded as the stand-on vessel. Therefore, the USV must avoid the obstacle [38]. According to the empirical data of collision avoidance, the angle ranges of the four encounters of overtaking, head-on, left crossing, and right crossing are determined [39], as shown in Figure 4.

(1) Overtaking rule: when the relative course angle between the USV and the obstacle is within $[112.5^{\circ}, 180^{\circ}) \cup [-180^{\circ}, -112.5^{\circ})$, that is,

$$(\alpha - \beta) \in [112.5^{\circ}, 180^{\circ}) \cup [-180^{\circ}, -112.5^{\circ}), \tag{18}$$

the USV will pass along the port side of the obstacle, as shown in Figure 5(a).

(2) Head-on rule: when the relative course angle between the USV and the obstacle satisfies $[-15^{\circ}, 15^{\circ})$, that is,

$$(180^{\circ} - (\alpha - \beta)) \in [-15^{\circ}, 15^{\circ}), \tag{19}$$

the USV will sail along the starboard side of the obstacle, as shown in Figure 5(b).

(3) Crossing from left rule: when the relative course angle between the USV and the obstacle satisfies $[-112.5^{\circ}, -15^{\circ})$, that is,

$$(\alpha - \beta) \in [-112.5^{\circ}, -15^{\circ}), \tag{20}$$

the USV will adjust the course to its port side and sail along the tail of the obstacle, as shown in Figure 5(c).

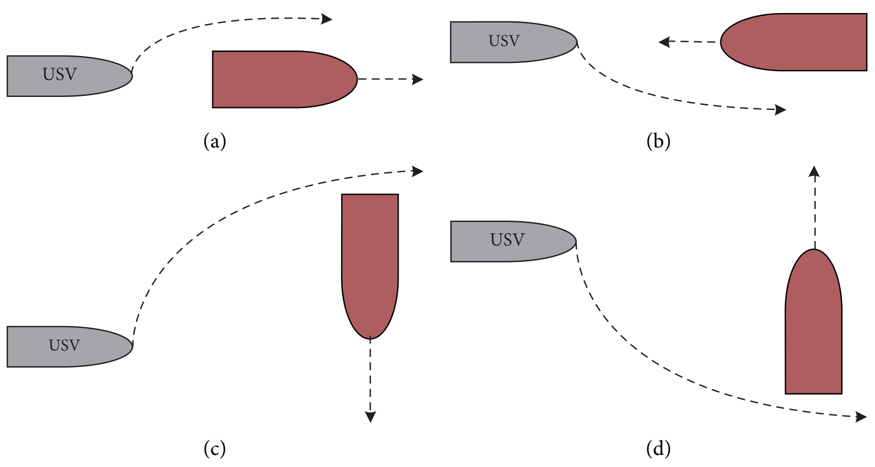

Figure 5: Possible encounter situations and collision avoidance paths between the USV and the obstacle. (a) Overtaking. (b) Head-on. (c) Crossing from left. (d) Crossing from right.

(4) Crossing from right rule: when the relative course angle between the USV and the obstacle satisfies $[15^\circ, 112.5^\circ)$, that is,

$$(\alpha - \beta) \in [15^\circ, 112.5^\circ), \tag{21}$$

the USV will adjust the course to its starboard side and sail along the tail of the obstacle, as shown in Figure 5(d).

2.4. Optimization Model of Local Path Planning

2.4. Optimization Model of Local Path Planning. The mathematical model of the optimization objectives and constraints of local path planning for USV collision avoidance is constructed.

2.4.1. Objective Functions. According to Section 2.2, the USV must adjust its velocity and course to avoid collisions. To adjust these simultaneously, it is necessary to obtain the optimal values of the two variables under constraints. Hence, the local path planning for USV collision avoidance is transformed to a multiobjective optimization problem. The energy consumption while sailing determines the USV's endurance and duration [34]. Therefore, the energy consumption of the USV is considered as one of the objectives of local path planning.

A USV sails on a preplanned global path until it detects unsafe obstacles, such that local adjustment must be made to the global path to avoid collision. To ensure the efficiency of the USV's task execution, it is expected that the motion of the USV should be changed as little as possible during local path planning, that is, the variations of the USV's velocity and course should be small.

In local path planning for USV collision avoidance, the velocity variation, course variation, and energy consumption of the USV during time interval Δt are used as the optimization objectives.

Suppose the velocity of the USV at time t is $v_{\text{usv}}(t)$, and after time interval Δt, its velocity is $v_{\text{usv}}(t+\Delta t)$. The velocity variation J_v is defined as

$$J_v = |\Delta v_{\text{usv}}| = |v_{\text{usv}}(t+\Delta t) - v_{\text{usv}}(t)|. \tag{22}$$

Suppose the course of the USV at time t is $\alpha(t)$, and after time interval Δt, its course is $\alpha(t+\Delta t)$. Consequently, the course variation J_α is

$$J_\alpha = |\Delta\alpha| = |\alpha(t+\Delta t) - \alpha(t)|. \tag{23}$$

The energy consumption of the USV while sailing is derived from the propulsion system. The resistance determines the effective power of the vessel [40]. The hydrodynamic drag F_d is calculated as

$$F_d = 0.5\rho|U|^2 C_D A, \tag{24}$$

where ρ is the density of the water, C_D is the drag coefficient, A is the reference area, and U is obtained by (4).

Suppose the USV's energy consumption after time interval Δt is

$$J_e = F_d|U|\Delta t = 0.5\rho|U|^3 C_D A\Delta t. \tag{25}$$

In summary, the cost function of USV local path planning is established as

$$J = \min(w_1 J_v + w_2 J_\alpha + w_3 J_e), \tag{26}$$

where w_1, w_2, and w_3 are the weights of J_v, J_α, and J_e, respectively.

2.4.2. Constraints of Local Path Planning. To plan an optimal local path for a USV, some constraints, such as kinematic constraints, COLREGs rules, and obstacle-avoidance constraints, must be met. The kinematic constraints are the limitations of the USV's velocity and course variation. The obstacle-avoidance constraints require the

USV to be outside the collision area, and these are obtained from (14).

Therefore, the constraints of USV local path planning are as follows:

$$0 < v_{\min} \le v_{\mathrm{usv}} \le v_{\max}, \tag{27}$$

$$\left|\Delta v_{\mathrm{usv}}\right| \le a, \tag{28}$$

$$|\Delta\alpha| \le r \cdot \Delta t, \tag{29}$$

$$\begin{aligned} &-\frac{\sin\phi}{\Delta v}\Delta v_{\mathrm{usv}} + \frac{v_{\mathrm{usv}}\cos\phi}{\Delta v}\Delta\alpha + \frac{\sin(|\alpha-\beta|+\phi)}{\Delta v}\Delta v_{\mathrm{obs}} \\ &-\frac{v_{\mathrm{obs}}\cos(|\alpha-\beta|+\phi)}{\Delta v}\Delta\beta > \mu - \gamma, \quad \gamma > 0, \end{aligned} \tag{30}$$

$$\begin{aligned} &-\frac{\sin\phi}{\Delta v}\Delta v_{\mathrm{usv}} + \frac{v_{\mathrm{usv}}\cos\phi}{\Delta v}\Delta\alpha + \frac{\sin(|\alpha-\beta|+\phi)}{\Delta v}\Delta v_{\mathrm{obs}} \\ &-\frac{v_{\mathrm{obs}}\cos(|\alpha-\beta|+\phi)}{\Delta v}\Delta\beta < -\mu - \gamma, \quad \gamma \le 0, \end{aligned} \tag{31}$$

where $v_{\max}$ and $v_{\min}$ are, respectively, the upper and lower bounds of v_{usv} and a is the limit of the size of Δv_{usv}. (27), (28), and (29) are constraints of USV kinematics. (30) and (31) are collision avoidance constraints. In addition, (18)–(21) are the constraints based on COLREGs.

In summary, the local path planning for USV collision avoidance is transformed into a multiobjective optimization problem with multiple constraints in a continuous search space.

3. Optimization Algorithm

As mentioned above, in order to obtain the velocity variation Δv_{usv} and course variation $\Delta\alpha$ in a continuous search space that satisfies a series of constraints, an optimization algorithm is needed. In this section, we introduce the MQPSO algorithm, which is used to plan the local path for USV collision avoidance.

3.1. Particle Swarm Optimization. PSO is an important method in swarm intelligence. The basic algorithm is as follows [41]. In the n-dimensional space, a population of m particles is represented as $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$. The position and velocity vectors of particle i can be represented as $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^{\mathrm{T}}$ and $v_i = [v_{i1}, v_{i2}, \ldots, v_{in}]^{\mathrm{T}}$, respectively. The n-dimensional best previous position vector is $\mathbf{p}_i = [p_{i1}, p_{i2}, \ldots, p_{in}]^{\mathrm{T}}$, and the position vector of the best among all particles in the population is $\mathbf{p}_g = [p_{g1}, p_{g2}, \ldots, p_{gn}]^{\mathrm{T}}$. We substitute $\mathbf{x}_i$ in the objective function to calculate its fitness. Particle i changes its velocity and position according to the following equations:

$$\begin{aligned} \mathbf{v}_{id}(t+1) = \mathbf{v}_{id}(t) &+ c_1 r_1 (\mathbf{p}_{id}(t) - \mathbf{x}_{id}(t)) \\ &+ c_2 r_2 \big(\mathbf{p}_{gd}(t) - \mathbf{x}_{id}(t)\big), \end{aligned} \tag{32}$$

$$\mathbf{x}_{id}(t+1) = \mathbf{x}_{id}(t) + \mathbf{v}_{id}(t+1), \tag{33}$$

where $d = 1, 2, \ldots, n$; $i = 1, 2, \ldots, m$, and t is the current iteration number. r_1 and r_2 are random numbers in $[0, 1]$, and c_1 and c_2 are positive constants. By the iteration of (32) and (33), each particle in the population gradually approaches the global optimal solution.

3.2. Quantum-Behaved Particle Swarm Optimization. In PSO, the velocity obtained by (32) is limited. In (33), the particles' movement range is also limited, such that particles cannot cover the whole feasible solution space. Therefore, PSO cannot guarantee a global optimal solution [42].

QBPSO uses the principle of quantum mechanics to determine the states of particles by wave functions, and the particles are bound by the center of the attractive potential generated by the delta potential well. In this way, particles in a quantum bound state can appear anywhere in the search space with a certain probability density. Therefore, the particles in QBPSO can be searched in the whole feasible solution space and approach global optimality.

In QBPSO, each particle is treated as spinless and moving in quantum space, and an individual particle is assumed to move in a delta potential well. In quantum mechanics space, the motion state of a particle is represented by a wave function ψ, which replaces the method described by the velocity vector and position vector of the particle. The probability density that a particle appears at a certain point in the search space is represented by $|\psi^2|$. After determining the probability distribution function of the particles, a particle's position is determined by the Monte Carlo method. In the n-dimensional space, a population is composed of m particles, and QBPSO updates related parameters through the following equations:

$$\begin{aligned} &p_{ij}(t) = \varphi_j(t) p\mathrm{best}_{ij}(t) + \big(1 - \varphi_j(t)\big) g\mathrm{best}_j(t), \\ &\varphi_j(t) \sim U(0, 1), \end{aligned} \tag{34}$$

where $1 \le i \le m$ and $1 \le j \le n$; p_{ij} is the center position of a delta potential well; $p\mathrm{best}_{ij}$ is the best previous position of particle i and is called the personal best ($p$best) position; and $g\mathrm{best}_j$ is the position of the best particle among all particles and is called the global best ($g$best) position [26, 43].

The mean best position $C(t)$ defined by the average of the best previous positions of all the particles is

$$\begin{aligned} \mathbf{C}(t) &= (C_1(t), C_2(t), \ldots, C_n(t)) = \frac{1}{m}\sum_{i=1}^{m} P_i(t) \\ &= \left(\frac{1}{m}\sum_{i=1}^{m} P_{i1}(t), \frac{1}{m}\sum_{i=1}^{m} P_{i2}(t), \ldots, \frac{1}{m}\sum_{i=1}^{m} P_{in}(t)\right). \end{aligned} \tag{35}$$

Thus, the position of particle i updates as follows:

$$\begin{aligned} &X_{ij}(t+1) = p_{ij}(t) \pm \alpha\left|C_j(t) - X_{ij}(t)\right| \left[\ln\left(\frac{1}{u_{ij}(t)}\right)\right], \\ &u_{ij}(t) \sim U(0, 1), \end{aligned} \tag{36}$$

where α is a contraction-expansion coefficient to adjust the algorithm.

3.3. Modified Quantum Particle Swarm Optimization. In QBPSO, particles with quantum behavior search in the whole feasible solution space. However, since the particles in QBPSO are encoded by real numbers, the search is only in one-dimensional space, and the search efficiency is low. Quantum bits are introduced to encode the positions of particles to improve the performance of QBPSO. Since the USV local path planning must be completed in real time, it is essential that the optimization algorithm can quickly obtain the optimal value. Therefore, MQPSO is proposed.

3.3.1. Quantum Code. The quantum bit (Q-bit) is the basic unit in quantum computing. A Q-bit is a system with two possible states, $|0\rangle$ and $|1\rangle$. The state of a Q-bit $|\varphi\rangle$ is expressed as $|\varphi\rangle = \cos\theta\cdot|0\rangle + \sin\theta\cdot|1\rangle$, where $\cos\theta$ and $\sin\theta$ are the probability amplitudes and θ is the phase angle of $|\varphi\rangle$. $|\cos\theta|^2$ and $|\sin\theta|^2$ are the probabilities in state $|0\rangle$ and $|1\rangle$, respectively. The state of $|\varphi\rangle$ collapses to state $|0\rangle$ with probability $|\cos\theta|^2$ and to state $|1\rangle$ with probability $|\sin\theta|^2$. Thus, the state of $|\varphi\rangle$ is an uncertain superposition state between $|0\rangle$ and $|1\rangle$. When the dimension of an individual $\mathbf{X}_i$ is n, $\mathbf{X}_i$ is expressed as

$$\mathbf{X}_i = \left[\begin{array}{c|c|c|c|} \cos\theta_{i1} & \cos\theta_{i2} & \cdots & \cos\theta_{in} \\ \sin\theta_{i1} & \sin\theta_{i2} & \cdots & \sin\theta_{in} \end{array}\right], \tag{37}$$

where $\mathbf{X}_{ic} = (\cos\theta_{i1}, \cos\theta_{i2}, \ldots, \cos\theta_{in})$ and $\mathbf{X}_{is} = (\sin\theta_{i1}, \sin\theta_{i2}, \ldots, \sin\theta_{in})$ are two sets of solutions for $\mathbf{X}_i$. Therefore, after quantum coding, in each iteration, every individual has two sets of solutions and the search space is doubled [35]. In this way, the ergodicity of the search space can be enhanced and the optimization process can be accelerated when the population size is same. In MQPSO, quantum coding is combined with quantum-behaved particle swarm optimization, that is, the positions of the particles are encoded by Q-bits according to (37).

3.3.2. Solution Space Transformation. In MQPSO, each particle in the population contains $2n$ Q-bits' probability amplitudes. Using linear transformation, these probability amplitudes can be mapped from n-dimensional unit space to the solution space of the optimization problem. Suppose the Q-bit of particle i on the jth dimension is $[\cos\theta_i^j, \sin\theta_i^j]^{\mathrm{T}}$. The corresponding solution space is

$$\begin{cases} X_{ic}^j = \dfrac{\left[b_i\left(1+\cos\theta_i^j\right) + a_i\left(1-\cos\theta_i^j\right)\right]}{2}, \\ X_{is}^j = \dfrac{\left[b_i\left(1+\sin\theta_i^j\right) + a_i\left(1-\sin\theta_i^j\right)\right]}{2}, \end{cases} \tag{38}$$

where the probability amplitude $\cos\theta_i^j$ of quantum state $|0\rangle$ corresponds to X_{ic}^j and the probability amplitude $\sin\theta_i^j$ of quantum state $|1\rangle$ corresponds to X_{is}^j [35].

3.3.3. Particle Evaluation. The two sets of solutions (X_{ic}, X_{is}) corresponding to particle i are substituted in the cost function to calculate its cost value. For the minimum problem, the smaller the cost value of the particle, the better the corresponding fitness value. The best previous position of particle i on the jth dimension is determined by

$$p\text{best}_{ij}(t) = \begin{cases} X_{ij}(t), & \text{if } f\left[X_{ij}(t)\right] < f\left[p\text{best}_{ij}(t-1)\right], \\ p\text{best}_{ij}(t-1), & \text{if } f\left[X_{ij}(t)\right] \geq f\left[p\text{best}_{ij}(t-1)\right], \end{cases} \tag{39}$$

where X_{ij} is the current position of particle i on the jth dimension.

The best particle's position $g\text{best}_j$ among all particles in the population on the jth dimension is

$$g = \arg\min_{1\leq i\leq m}\left\{f\left[p\text{best}_{ij}(t)\right]\right\}, \tag{40}$$

$$g\text{best}_j(t) = P_g(t), \tag{41}$$

where g is the label of the particle in the global best position.

3.3.4. Particle State Update. In MQPSO, the update of the particle's position is converted to the update of the Q-bit probability amplitude's phase angle. In this paper, the particle-updating mechanism in QBPSO is used to adjust the phase angle θ of the Q-bit to update the particle's position. In the n-dimensional space, a population is composed of m particles. At the tth iteration, the best previous position of particle i and the position of the best among all particles in the population are expressed by probability amplitudes as

$$p\text{best}_{ij}(t) = \left[\cos\left(\theta_{ij}^p(t)\right), \sin\left(\theta_{ij}^p(t)\right)\right]^{\mathrm{T}}, \tag{42}$$

$$g\text{best}_j(t) = \left[\cos\left(\theta_{gj}(t)\right), \sin\left(\theta_{gj}(t)\right)\right]^{\mathrm{T}}, \tag{43}$$

where $1\leq i\leq m$ and $1\leq j\leq n$, respectively.

The phase angle of the Q-bit in the delta potential well center position C_{ij} is

$$\begin{aligned} &\theta_{ij}^c(t) = \varphi_j(t)\theta_{ij}^p(t) + \left(1-\varphi_j(t)\right)\theta_{gj}(t) \\ &\varphi_j(t) \sim U(0,1), \end{aligned} \tag{44}$$

where $\theta_{ij}^p(t)$ and $\theta_{gj}(t)$ are, respectively, the phase angles of the best previous position of particle i and the position of the best among all particles.

Therefore, at the $(t+1)$th iteration, the phase angle of the Q-bit of the position of the ith particle is

$$\begin{aligned} &\theta_{ij}(t+1) = \theta_{ij}^c(t) \pm \alpha\left|\frac{1}{m}\sum_{i=1}^{m}\left(\theta_{ij}^p(t) - \theta_{ij}(t)\right)\right|\ln\left[\frac{1}{u_{ij}(t)}\right] \\ &u_{ij}(t) \sim U(0,1), \end{aligned} \tag{45}$$

where m is the number of the particles.

In summary, the position of particle i on the jth dimension at the $(t+1)$th iteration is expressed as

$$\mathbf{X}_{ij}(t+1) = \begin{bmatrix} \cos\left(\theta_{ij}(t+1)\right) \\ \sin\left(\theta_{ij}(t+1)\right) \end{bmatrix}. \tag{46}$$

3.3.5. Mutation Operation. Because the population loses its diversity during optimization [27], the PSO algorithm tends to fall into local extreme values. So, the population mutation factor, that is, quantum nongate, is introduced to increase the population's diversity [35]. First, the mutation probability P_i must be determined, and we choose a random number Rnd_i in $(0,1)$ for each particle. If $\mathrm{Rnd}_i < P_i$, then we randomly select $n/2$ Q-bits and switch the probability amplitudes of Q-bits by the quantum non-gate, such as

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \cos\left(\theta_{ij}\right) \\ \sin\left(\theta_{ij}\right) \end{bmatrix} = \begin{bmatrix} \sin\left(\theta_{ij}\right) \\ \cos\left(\theta_{ij}\right) \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\pi}{2} - \theta_{ij}\right) \\ \sin\left(\frac{\pi}{2} - \theta_{ij}\right) \end{bmatrix}. \tag{47}$$

Since the mutation operation is independent of the best previous position of the particle and the position of the best among all particles, it can increase the diversity of the population.

3.3.6. Pseudocodes of MQPSO. Based on the above, the main steps of MQPSO to obtain the minimum value of the optimization problem are shown as Algorithm 1.

3.4. Local Path Planning Based on MQPSO. As described in Section 2, the local path planning for USV collision avoidance is transformed to a multiobjective optimization problem with multiple constraints. First, we formulate the optimization model of USV local path planning according to (18)–(21) and (22)–(31). Without loss of generality, it is assumed that the USV detects the obstacle that will affect its safe navigation at the waypoint P_1 when $t = 0$. According to the optimization model, MQPSO can obtain the velocity variation $\Delta v_{\mathrm{usv}}(t)$ and course variation $\Delta\alpha(t)$. Then, the USV sails at velocity $v_{\mathrm{usv}}(t+\Delta t) = v_{\mathrm{usv}}(t) + \Delta v_{\mathrm{usv}}(t)$ and course $\alpha(t+\Delta t) = \alpha(t) + \Delta\alpha(t)$ to reach the waypoint P_2. We repeat this process until the USV has completed collision avoidance. The waypoints are connected to form a local path for USV collision avoidance.

4. Evaluation and Simulation Studies

4.1. Performance Evaluation of MQPSO. To examine the effectiveness and efficiency of MQPSO, its performance is compared with that of PSO and QBPSO on five different kinds of test benchmark functions. The five benchmark functions are listed in Table 1. They are the Ackley, Griewank, Rastrigin, Rosenbrock, and Schaffer functions as defined as follows.

$f_1(x)$ is Ackley function. Ackley function is an n-dimensional function with one narrow global optimum basin and many minor local optima [44]. The function gets the minimum value of 0 at $(x_1, x_2, \ldots, x_n) = (0, 0, \ldots, 0)$.

$f_2(x)$ is Griewank function. Griewank function has a $\prod_{i=1}^{N} \cos(x_i/\sqrt{i})$ component causing linkages among variables, thereby making it difficult to reach the global optimum [44]. The function gets the minimum value of 0 at $(x_1, x_2, \ldots, x_n) = (0, 0, \ldots, 0)$.

$f_3(x)$ is Rastrigin function. Rastrigin function is a complex multimodel problem with a large number of local optima. There are about $10n$ local optimum in the range $x_i \in [-5.12, 5.12]$. When attempting to solve Rastrigin function, algorithms may easily fall into a local optimum [44]. The function gets the minimum value of 0 at $(x_1, x_2, \ldots, x_n) = (0, 0, \ldots, 0)$.

$f_4(x)$ is Rosenbrock function. Because Rosenbrock function provides little information for the optimization algorithm, it is difficult for the algorithm to identify the search direction and find the optimal solution [28]. The function gets the minimum value of 0 at $(x_1, x_2, \ldots, x_n) = (1, 1, \ldots, 1)$.

$f_5(x_1, x_2)$ is Schaffer function. Schaffer function is a two-dimensional complex function. Because the function has strong oscillation characteristics, it is difficult to find the global optimal value [28]. The function gets the minimum value of 0 at $(x_1, x_2) = (1, 1)$.

In the test, the maximum number of iterations is 300, the number of particles is 50, and the dimensions of the benchmark functions are 30. The parameters of PSO, QBPSO, and MQPSO are consistent.

Since all the above test benchmark functions are minimization problems, the smaller the final value of these functions, the better the performance of the corresponding algorithm. The test results, which contain optimal value, the number of iterations for optimization algorithm to converge to the minimum, the calculation time, and the average calculation time for per iteration, are listed in Table 2. The 1st, 5th, and 9th rows of Table 2, respectively, show the optimal values obtained by PSO, QBPSO, and MQPSO for benchmark functions f_1 to f_5. The 2nd, 6th, and 10th rows show the number of iterations required for PSO, QBPSO, and MQPSO to converge to the minimum. The 3rd, 7th, and 11th rows, respectively, show the calculation time required for the benchmark functions f_1 to f_5 to perform the entire optimization calculation based on PSO, QBPSO, and MQPSO. The 4th, 8th, and 12th rows, respectively, show the average calculation time taken by each iteration of benchmark functions f_1 to f_5 based on PSO, QBPSO, and MQPSO.

The data in Table 2 show that MQPSO is superior to PSO and QBPSO in both the optimal value and the number of iterations that converge to the minimum.

The optimal values obtained by MQPSO and QBPSO are lower than obtained by PSO. Because the motion states of particles in MQPSO and QBPSO are determined by wave functions, the particles can cover the whole feasible solution space. MQPSO uses the above methods to avoid falling into local extreme values. And MQPSO further avoids falling into local extremes by using a quantum nongate to increase the population's diversity. Therefore, the optimal values

```
        **Input**: the number of the particles $m$,
          the dimension of the problem $n$,
          the maximum of iterations $T_{\max}$;
        **Output**: optimal solution $gbest$;
 (1) Initialize the position of each particle, $pbest$, and $gbest$ according to (37), (42), and (43), respectively;
 (2) $t \leftarrow 0$;
 (3) **if** $t \le T_{\max}$, **then**
 (4)     $t \leftarrow t + 1$;
 (5)     **for** each particle $i$, **do**
 (6)         **for** each dimension $j$, **do**
 (7)             Calculate the particle's fitness value according to (26);
 (8)             Update $pbest_{ij}(t)$ according to (39);
 (9)             Update $gbest(t)$ according to (40) and (41);
(10)             Calculate $\theta_{ij}^{c}(t)$ according to (44);
(11)             Update $\theta_{ij}(t+1)$ according to (45);
(12)         Determine the new position of particle $i$ according to (46);
(13) Output the optimal solution $gbest$.
```

Algorithm 1: MQPSO.

Table 1: Benchmark functions.

Benchmark function	Search limit
$f_1(x) = -20\exp\left(-0.2\sqrt{(1/N)\sum_{i=1}^{N} x_i^2}\right) - \exp\left((1/N)\sum_{i=1}^{N}\cos(2\pi x_i)\right) + 20 + \exp(1)$	$x_i \in [-8, 8]$
$f_2(x) = \sum_{i=1}^{N} (x_i^2/4000) - \prod_{i=1}^{N}\cos(x_i/\sqrt{i}) + 1$	$x_i \in [-600, 600]$
$f_3(x) = \sum_{i=1}^{N} [x_i^2 - 10\cos(2\pi x_i) + 10]$	$x_i \in [-5.12, 5.12]$
$f_4(x) = \sum_{i=1}^{N-1} [100(x_i^2 - x_{i+1}) + (x_i - 1)^2]$	$x_i \in [-8, 8]$
$f_5(x_1, x_2) = 0.5 + (((\sin\sqrt{x_1^2 + x_2^2})^2 - 0.5)/(1 + 0.001(x_1^2 + x_2^2))^2)$	$x_1, x_2 \in [-10, 10]$

Table 2: Results of three algorithms on benchmark functions.

Algorithm	Performance	$f_1(x)$	$f_2(x)$	$f_3(x)$	$f_4(x)$	$f_5(x_1, x_2)$
PSO	Optimal value	1.55×10^{-3}	5.56×10^{-3}	5.61×10^{-2}	2.03×10^{-3}	6.32×10^{-3}
	Iteration	228	245	258	206	261
	Time (s)	1.8619	1.8428	1.9653	1.5378	1.6795
	Average time per iteration (s)	0.0062	0.0061	0.0066	0.0051	0.0056
QBPSO	Optimal value	9.37×10^{-4}	2.05×10^{-3}	2.56×10^{-2}	1.25×10^{-3}	3.12×10^{-3}
	Iteration	215	231	254	197	263
	Time (s)	1.4844	1.4688	1.4531	1.1563	1.2031
	Average time per iteration (s)	0.0049	0.0049	0.0048	0.0039	0.0040
MQPSO	Optimal value	9.15×10^{-4}	1.86×10^{-3}	1.96×10^{-2}	1.12×10^{-3}	2.89×10^{-3}
	Iteration	149	158	136	132	142
	Time (s)	1.5271	1.5163	1.5534	1.3123	1.4808
	Average time per iteration (s)	0.0051	0.0051	0.0052	0.0044	0.0049

obtained by MQPSO are slightly lower than those obtained by QBPSO and PSO.

The number of iterations required for MQPSO to converge to the minimum is obviously lower than for QBPSO and PSO. Because the positions of the particles are encoded by Q-bits, the search space is doubled, and the convergence speed is faster.

In PSO, the strategy of first calculating the particle's velocity at the next moment and then updating the positions of the particles is adopted, as shown in (32) and (33). So, the calculation time is long. QBPSO uses a method of directly updating the particles' positions, which reduces the calculation time. The particle's position update strategy of MQPSO is based on QBPSO. However, if the number of encoded Q-bits is n, each particle is updated n times in MQPSO. Therefore, the increasing calculation will lead to the longer calculation time. As the calculation time increases, the number of particle's position updates is also increased, improving the optimization ability of the algorithm.

Simulation results show that the optimization capability of MQPSO is indeed better than QBPSO and PSO.

TABLE 3: Initial motion information for USV and obstacle.

	Case 1	Case 2	Case 3
Initial location of USV	$(0, 0)$	$(0, 0)$	$(0, 0)$
Location of goal	$(500, 500)$	$(500, 500)$	$(500, 500)$
$v_{usv}(0)$	18 m/s	18 m/s	18 m/s
$\alpha(0)$	45°	45°	45°
Initial location of obstacle	$(150, 150)$	$(500, 400)$	$(450, 40)$
$v_{obs}(0)$	5 m/s	10 m/s	10 m/s
$\beta(0)$	45°	-145°	125°
Δv_{obs}	0.6 m/s	0.8 m/s	0.8 m/s
$\Delta\beta$	0.8594°	0.8594°	-0.8594°
Encounter situation	Overtaking	Head-on	Crossing from right
Whether to follow COLREGs	Yes	Yes	Yes
Expected time of collision	20s	20s	20s

4.2. Simulations of USV Local Path Planning. In this section, some simulations of USV local path planning based on MQPSO, QBPSO, and PSO considering the uncertainty of obstacle's velocity are discussed.

It is assumed that the USV can detect the motion states of the obstacle by the shipborne sensors and estimate its motion states, that is, Δv_{obs} and $\Delta\beta$ in (10) can be measured. The initial motion information for the USV and the obstacles in Cases 1–3 is listed in Table 3. If the USV does not avoid collision when it detects an obstacle that affects its safe navigation at $t = 0$ s, then the USV will collide with the obstacle at $t = 20$ s. The time interval ΔT is 2 s. In this way, the USV will plan a local path every 2 seconds, that is, 10 calculations will be performed during the entire collision avoidance.

The radius of the collision area $\odot O$ is 30 m, and the radius of $\odot \mathbf{w}_{obs}$ is 9 m. The parameters in (25) are set as $C_D = 1$ and $A = 300$. In Cases 1–3, the weights $[w_1, w_2, w_3]$ are $[1000, 100, 1]$ in (26). The constraints in (27), (28), and (29), respectively, are $0\ \text{m/s} < v_{usv} \leq 22\ \text{m/s}$, $|\Delta v_{usv}| \leq 0.25\ \text{m/s}$, and $|\Delta\alpha|/\Delta t \leq 2^\circ$.

In the simulations, the maximum number of iterations is 200 and the number of particles is 50. A particle represents a candidate solution, so the particles are two-dimensional, where one dimension represents Δv_{usv} and the other represents $\Delta\alpha$. The parameters of MQPSO, QBPSO, and PSO are consistent.

Figures 6–8 show the planned local paths for USV collision avoidance based on MQPSO considering uncertainty in Cases 1–3, respectively. The location of the USV is represented by a blue pentagon every two seconds. The location of the obstacle is represented by a red triangle every two seconds. The location of the USV is represented by a black pentagon at $t = 20$ s. The collision area is represented as a dotted red line at $t = 20$ s, and the USV global path is represented by a dotted green line.

Table 4 shows the cost values in each calculation, respectively, obtained by MQPSO, QBPSO, and PSO in Cases 1–3, where $J(1)$ represents the first calculation performed by local path planner when $t = 0$ and so on. From the data in Table 4, we can conclude that the cost value obtained by the USV local path planner based on MQPSO is lower than

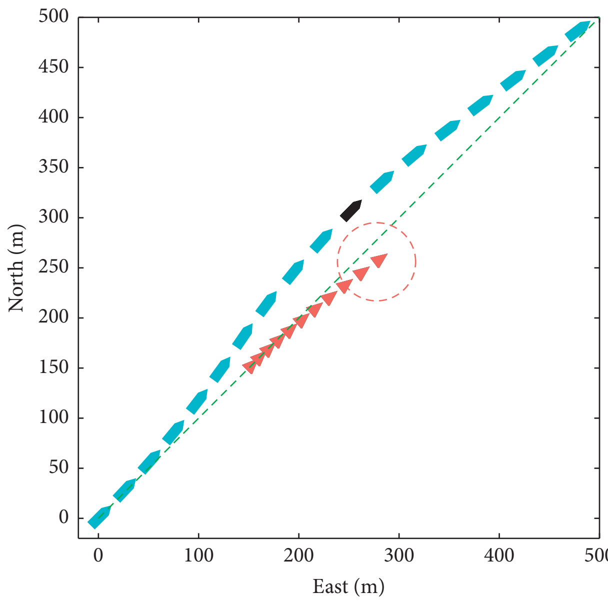

FIGURE 6: Local path for USV collision avoidance based on MQPSO in Case 1.

those obtained by QBPSO and PSO. According to the evaluation criteria of (25), the paths obtained by MQPSO are better than the paths obtained by QBPSO and PSO. Therefore, it is reasonable to choose MQPSO as the optimization algorithm of USV local path planning in this study.

Figures 6–8 show the paths obtained by MQPSO can avoid the moving obstacle in Cases 1–3. After $t = 20$ s, the USV returns to the global path as soon as possible. In Cases 1–3, the strategy of simultaneously adjusting the velocity and course of the USV is adopted.

As shown in Table 3, the encounter situation of Case 1 between the USV and the obstacle is overtaking. According to the rules in COLREGs and Section 2.3, the USV should pass along the port side of the obstacle. As shown in Figure 6, the local path planner based on MQPSO can obtain a collision-free path that meets the rules of COLREGs on

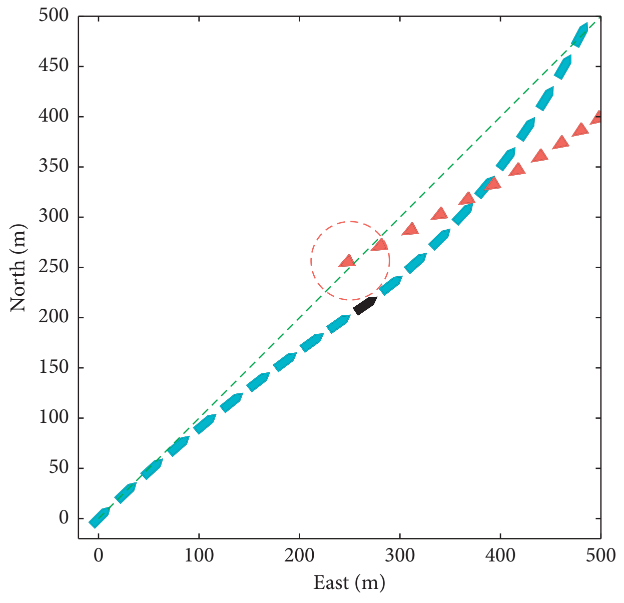

Figure 7: Local path for USV collision avoidance based on MQPSO in Case 2.

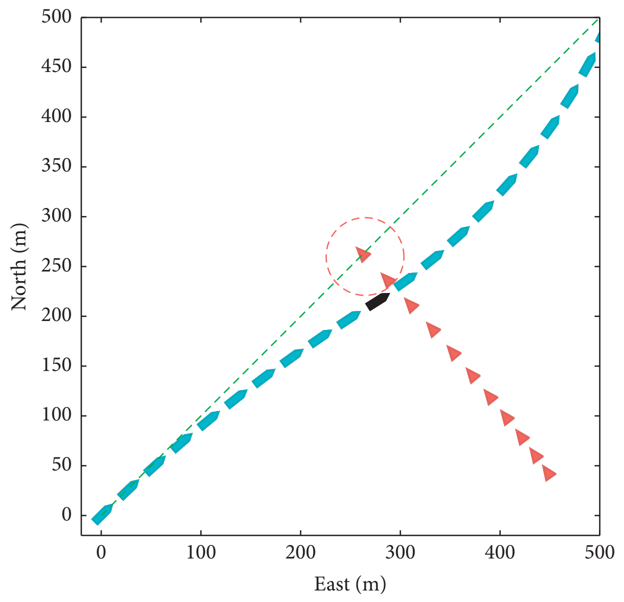

Figure 8: Local path for USV collision avoidance based on MQPSO in Case 3.

Table 4: Cost values in each calculation.

	Algorithm	$J(1)$	$J(2)$	$J(3)$	$J(4)$	$J(5)$	$J(6)$	$J(7)$	$J(8)$	$J(9)$	$J(10)$
Case 1	MQPSO	1319.75	1305.67	1328.42	1323.27	1322.91	1325.21	1525.87	1595.71	1698.57	1784.21
	QBPSO	1406.68	1433.61	1407.88	1383.07	1357.21	1358.81	1600.38	1692.81	1784.61	1844.95
	PSO	1420.98	1430.80	1443.26	1424.96	1413.08	1398.57	1634.82	1774.58	1877.89	1918.35
Case 2	MQPSO	1048.58	1037.63	1013.48	1007.50	967.91	937.69	917.59	890.68	845.15	836.10
	QBPSO	1138.62	1130.76	1097.33	1071.18	1037.98	1006.69	979.31	952.58	908.22	888.58
	PSO	1186.66	1167.49	1162.48	1118.84	1086.32	1058.32	1040.08	1005.78	967.77	949.83
Case 3	MQPSO	992.27	980.16	988.98	967.42	965.07	961.52	948.09	953.24	936.98	932.10
	QBPSO	1017.63	1020.35	1022.21	1006.77	1018.66	996.09	984.16	986.49	986.19	972.12
	PSO	1097.36	1083.25	1069.46	1063.22	1050.69	1055.83	1053.99	1041.82	1020.96	1021.23

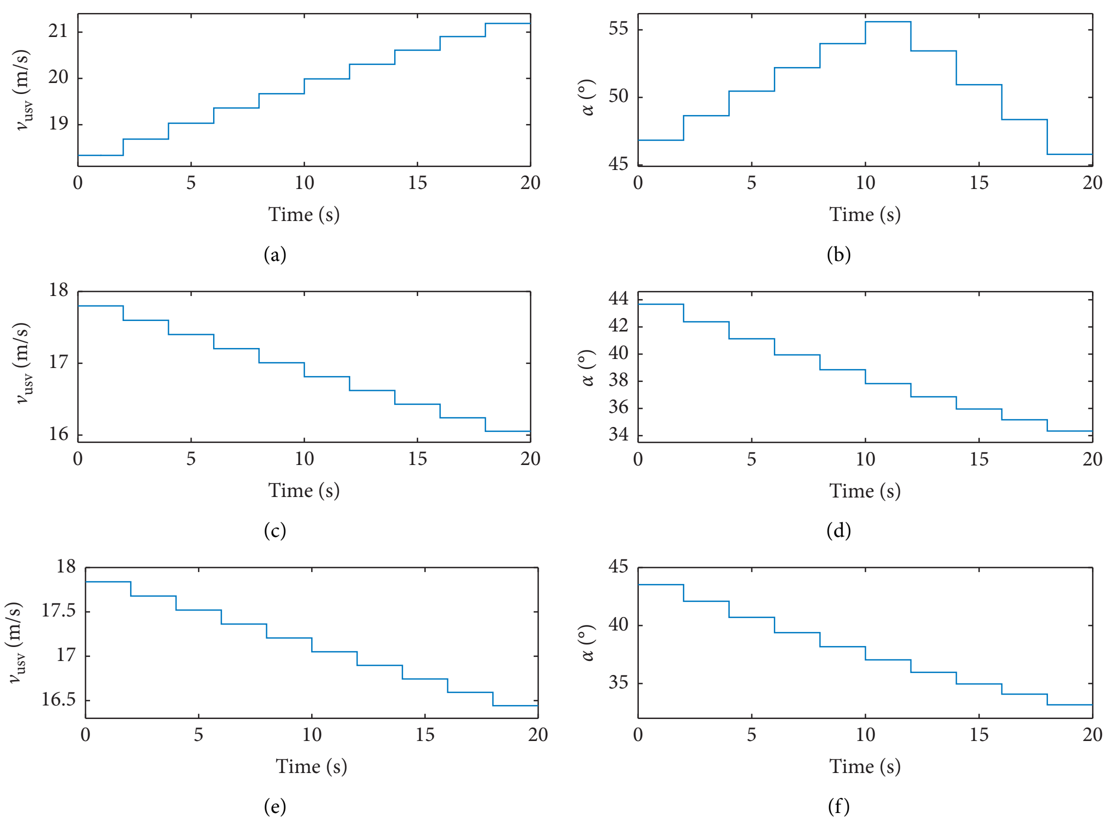

Figure 9: Adjusted velocity and course based on MQPSO in Cases 1–3.

overtaking. The adjusted velocity and course of the USV within 0–20 s in Case 1 are shown in Figure 9(a) and 9(b), respectively. As shown in Table 3, the encounter situation of Case 2 between the USV and the obstacle is head-on. According to the rules in COLREGs and Section 2.3, the USV should sail along the starboard side of the obstacle. As shown in Figure 7, the local path planner based on MQPSO can obtain a collision-free path that meets the rules of COLREGs on head-on. The adjusted velocity and course of the USV within 0–20 s in Case 2 are shown in Figure 9(c) and 9(d), respectively. As shown in Table 3, the encounter situation of Case 3 between the USV and the obstacle is crossing from right. According to the rules in COLREGs and Section 2.3, the USV should adjust the course to its starboard side and sail along the tail of the obstacle. As shown in Figure 8, the local path planner based on MQPSO can obtain a collision-free path that meets the rules of COLREGs on crossing from right. The adjusted velocity and course of the USV within 0–20 s in Case 3 are shown in Figure 9(e) and 9(f), respectively.

The results verify that the proposed algorithm can effectively plan the local path for USV collision avoidance.

5. Conclusions

This study proposes a local path planning algorithm for USV collision avoidance based on MQPSO.

First, a USV collision avoidance model is established, which is composed of USV kinematics, COLREGs rules, the uncertainty of the obstacle's velocity, and a VO method, which not only considers the velocity and course of the USV but also handles variable velocities and courses of an obstacle. Due to the kinematic constraints of the USV, the velocity window and course window of the USV are determined by DWA. In this way, local path planning for USV collision avoidance is transformed into a multiobjective optimization problem with multiple constraints in a continuous search space. Thus, a USV can avoid collision by simultaneously adjusting its optimized velocity and course.

Second, to solve the optimization problem, MQPSO is primarily proposed. MQPSO is an optimization algorithm combining quantum computing with QBPSO. It benefits from the high efficiency of quantum computing and the optimization ability of QBPSO. Simulation results show that MQPSO converges more quickly than PSO and QBPSO and avoids falling into local extreme values. The effectiveness of the proposed collision avoidance model is verified. Performance evaluation tests on benchmark functions show that MQPSO is superior to PSO and QBPSO in both the optimal value and the number of iterations required for convergence, and the proposed algorithm based on MQPSO can effectively and efficiently plan a collision-free USV path.

In future work, the collision avoidance model proposed in this study should consider multiple obstacles that a USV may encounter during navigation. Moreover, the correlation between multiple objectives should be calculated, and MQPSO should accommodate multiobjective processing.

The uncertainties of the obstacle's velocity and course are also considered in future works.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the 7th Generation Ultra Deep Water Drilling Platform (Ship) Innovation Project and the National Natural Science Foundation of China (grant no. 51879049).

References

[1] H. Kim, S.-H. Kim, M. Jeon, J. Kim, S. Song, and K.-J. Paik, "A study on path optimization method of an unmanned surface vehicle under environmental loads using genetic algorithm," *Ocean Engineering*, vol. 142, pp. 616–624, 2017.

[2] M. G. Park, J. H. Jeon, and M. C. Lee, "Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing," in *Proceedings of the ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No.01TH8570)*, vol. 3, June 2001.

[3] U. S. Navy, *The Navy Unmanned Surface Vehicle (USV) Master Plan*, Department of the Navy, Washington DC, USA, 2007.

[4] A. Tan and C. Wee Wong, T. J. Tan, Criteria and rule based obstacle avoidance for usvs," in *Proceedings of the 2010 International WaterSide Security Conference*, November 2010.

[5] Z. Wu, J. Li, J. Zuo, and S. Li, "Path planning of UAVs based on collision probability and Kalman filter," *IEEE Access*, vol. 6, pp. 34237–34245, 2018.

[6] C. Tam and R. Bucknall, "Collision risk assessment for ships," *Journal of Marine Science and Technology*, vol. 15, no. 3, pp. 257–270, 2010.

[7] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[8] J. Snape, J. v. d. Berg, S. J. Guy, and D. Manocha, "The hybrid reciprocal velocity obstacle," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 696–706, 2011.

[9] M. Kim and J.-H. Oh, "Study on optimal velocity selection using velocity obstacle (OVVO) in dynamic and crowded environment," *Autonomous Robots*, vol. 40, no. 8, pp. 1459–1470, 2016.

[10] Y. I. Jenie, E.-J. v. Kampen, C. C. de Visser, J. Ellerbroek, and J. M. Hoekstra, "Selective velocity obstacle method for deconflicting maneuvers applied to unmanned aerial vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 6, pp. 1140–1146, 2015.

[11] S. Samavati, M. Zarei, and M. T. Masouleh, "An optimal motion planning and obstacle avoidance algorithm based on the finite time velocity obstacle approach," in *Proceedings of the 2017 Artificial Intelligence and Signal Processing Conference (AISP)*, October 2017.

[12] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[13] O. Brock and O. Khatib, "High-speed navigation using the global dynamic window approach," in *Proceedings of the 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 1, May 1999.

[14] C. Henkel, A. Bubeck, and W. Xu, "Energy efficient dynamic window approach for local path planning in mobile service Robotics∗∗This work was conducted at the university of auckland, auckland, New Zealand," *IFAC-PapersOnLine*, vol. 49, no. 15, pp. 32–37, 2016.

[15] COLREGs, *Convention on the International Regulations for Preventing Collisions at Sea*, International Maritime Organization, London, UK., 1972.

[16] M. Candeloro, A. M. Lekkas, and A. J. Sørensen, "A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels," *Control Engineering Practice*, vol. 61, pp. 41–54, 2017.

[17] H. Wang, F. Guo, H. Yao, S. He, and X. Xu, "Collision avoidance planning method of USV based on improved ant colony optimization algorithm," *IEEE Access*, vol. 7, pp. 52964–52975, 2019.

[18] L. Hu, W. Naeem, E. Rajabally et al., "A multiobjective optimization approach for COLREGs-compliant path planning of autonomous surface vehicles verified on networked bridge simulators," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1167–1179, 2019.

[19] C. Huang, Y. Lan, Y. Liu et al., "A new dynamic path planning approach for unmanned aerial vehicles," *Complexity*, vol. 2018, Article ID 8420294, 17 pages, 2018.

[20] Q. Xu, C. Zhang, and N. Wang, "Multiobjective optimization based vessel collision avoidance strategy optimization," *Mathematical Problems in Engineering*, vol. 2014, Article ID 914689, 9 pages, 2014.

[21] San Juan, Víctor, M. Santos, and J. M. Andújar, "Intelligent UAV map generation and discrete path planning for search and rescue operations," *Complexity*, vol. 2018, Article ID 6879419, 17 pages, 2018.

[22] Y. Liu, X. Zhang, X. Guan et al., "Potential odor intensity grid based UAV path planning algorithm with particle swarm optimization approach," *Mathematical Problems in Engineering*, vol. 2016, Article ID 7802798, 16 pages, 2016.

[23] X. Yu, W.-N. Chen, T. Gu et al., "Set-based discrete particle swarm optimization based on decomposition for permutation-based multiobjective combinatorial optimization problems," *IEEE Transactions on Cybernetics*, vol. 48, no. 7, pp. 2139–2153, 2018.

[24] A. Wu and Z.-L. Yang, "An elitist transposon quantum-based particle swarm optimization algorithm for economic dispatch problems," *Complexity*, vol. 2018, Article ID 7276585, 15 pages, 2018.

[25] J. Yu, S. Wang, and L. Xi, "Evolving artificial neural networks using an improved PSO and DPSO," *Neurocomputing*, vol. 71, no. 4–6, pp. 1054–1060, 2008.

[26] J. Sun, W. Xu, and B. Feng, "A global search strategy of quantum-behaved particle swarm optimization," in *Proceedings of the IEEE Conference on Cybernetics and Intelligent Systems*, vol. 1, December 2004.

[27] J. Sun, W. Fang, X. Wu, V. Palade, and W. Xu, "Quantum-behaved particle swarm optimization: analysis of individual particle behavior and parameter selection," *Evolutionary Computation*, vol. 20, no. 3, pp. 349–393, 2012.

[28] W. Fang, J. Sun, H. Chen, and X. Wu, "A decentralized quantum-inspired particle swarm optimization algorithm with cellular structured population," *Information Sciences*, vol. 330, pp. 19–48, 2016.

[29] J. Sun, X. Wu, W. Fang, Y. Ding, H. Long, and W. Xu, "Multiple sequence alignment using the Hidden Markov Model trained by an improved quantum-behaved particle swarm optimization," *Information Sciences*, vol. 182, no. 1, pp. 93–114, 2012.

[30] A. Narayanan and M. Moore, "Quantum-inspired genetic algorithms," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, May 1996.

[31] K.-H. Han and J.-H. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 580–593, 2002.

[32] H. Xing, Y. Ji, L. Bai, X. Liu, Z. Qu, and X. Wang, "An adaptive-evolution-based quantum-inspired evolutionary algorithm for QoS multicasting in IP/DWDM networks," *Computer Communications*, vol. 32, no. 6, pp. 1086–1094, 2009.

[33] G. Xia, Z. Han, B. Zhao, and Y. Yang, "Unmanned surface vessel path planning based on quantum ant colony algorithm," *Journal of Harbin Engineering University*, vol. 40, no. 7, pp. 880–885, 2019.

[34] G. Xia, Z. Han, Bo Zhao, C. Liu, and X. Wang, "Global path planning for unmanned surface vehicle based on improved quantum ant colony algorithm," *Mathematical Problems in Engineering*, vol. 2019, Article ID 2902170, 10 pages, 2019.

[35] S. Y. Li and P. C. Li, "Quantum particle swarms algorithm for continuous space optimization," *Chinese Journal of Quantum Electronics*, vol. 24, no. 5, pp. 569–574, 2007.

[36] Ke Meng et al., "Quantum-inspired particle swarm optimization for valve-point economic load dispatch," *IEEE Transactions on Power Systems*, vol. 25, no. 1, pp. 215–222, 2009.

[37] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*, John Wiley & Sons, New York, NY, USA, 2011.

[38] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, "Safe maritime autonomous navigation with COLREGS, using velocity obstacles," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 110–119, 2013.

[39] G. P. Smeaton and F. P. Coenen, "Developing an intelligent marine navigation system," *Computing & Control Engineering Journal*, vol. 1, no. 2, pp. 95–103, 1990.

[40] H. Niu, Y. Lu, A. Savvaris, and A. Tsourdos, "An energy-efficient path planning algorithm for unmanned surface vehicles," *Ocean Engineering*, vol. 161, pp. 308–321, 2018.

[41] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*, pp. 760–766, Springer, Boston, MA, USA, 2011.

[42] F. Van Den Bergh, *An analysis of particle swarm optimizers*, PhD Dissertations University of Pretoria, Pretoria, South Africa, 2001.

[43] X. Su, W. Fang, Q. Shen, and X. Hao, "An image enhancement method using the quantum-behaved particle swarm optimization with an adaptive strategy," *Mathematical Problems in Engineering*, vol. 2013, Article ID 824787, 14 pages, 2013.

[44] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.