Generalized velocity obstacle algorithm for preventing ship collisions at sea

Huang, Yamin; Chen, Linying; van Gelder, P. H.A.J.M.

DOI
10.1016/j.oceaneng.2018.12.053

Publication date
2019

Document Version
Accepted author manuscript

Published in
Ocean Engineering

Citation (APA)
Huang, Y., Chen, L., & van Gelder, P. H. A. J. M. (2019). Generalized velocity obstacle algorithm for preventing ship collisions at sea. *Ocean Engineering*, *173*, 142-156. https://doi.org/10.1016/j.oceaneng.2018.12.053

Generalized Velocity Obstacle Algorithm for Preventing Ship Collisions at Sea

Yamin Huang[1], Linying Chen[2], P. H. A. J. M. van Gelder[1]

1. *Safety and Security Science Group, Faculty of Technology, Policy and Management, Delft University of Technology, Delft, the Netherlands*
2. *Department of Maritime and Transport Technology, Delft University of Technology, Delft, the Netherlands*

Abstract: Numerous methods have been developed for ship collision prevention over the past decades. However, most studies are based on strong assumptions, such as the need for a constant velocity of the target-ship, the limitation to two-ship scenarios, the simplification of ships' dynamics, etc. Generalized Velocity Obstacle (GVO) algorithm can bridge these gaps. This paper presents a GVO algorithm for ship collision avoidance and designs a collision avoidance system (GVO-CAS). The proposed system visualizes the changes of one ship's course and speed resulting in collisions, which can be used not only for supporting the officer on watch to prevent collisions, but also for collision prevention of Autonomous Surface Vessels (ASVs) and for human operators taking over the control of ASVs. Simulation experiments show that the proposed collision avoidance system can work properly in various maritime environments. Compared to the original Velocity Obstacle algorithm, the GVO algorithm is more reliable and suitable for close range ship collision avoidance. Moreover, the GVO-CAS can offer rule-compliant evasive actions with a minimum number of required actions for ships. These results show the great potential to use the GVO algorithm in both manned and unmanned ships at sea.

Keywords: Collision prevention; Generalized velocity obstacle; Ship dynamics model; COLREGs Compliance

1. Introduction

Ship collision is of critical and fundamental concern for the maritime community, due to its high frequency and severe consequence. Many techniques have been developed for preventing collisions at sea. In the body of literature, most research has many assumptions and are applied to specific scenarios, such as relatively simple dynamics of the ship (e.g. holonomic vehicle), conflicts involving only a pair of ships, and constant-velocity of target-ships. Those assumptions may make the methods not suitable for collision prevention in more general collision avoidance cases. Moreover, most methods only provide one solution. If those methods function as navigation assistance for human-operated ships, the operators have no information about the decision processes. Consequently, they do not have choices but to accept the proposed solution.

A family of Velocity Obstacle (VO) algorithms bridges some of these gaps: VO algorithms enable to resolve conflicts with multiples moving obstacles (static and/or dynamic); the algorithms collect all the velocities that result in collisions and present a set of collision-free velocities for human/machine, which facilitates human/machine to search for the best option. Today, these algorithms have been widely used for collisions prevention of various vehicles, e.g. car-like robots (Alonso-Mora et al., 2012), airplanes (Velasco et al., 2015), unmanned underwater vehicles (Zhang et al., 2017), etc.

The advantages of the VO algorithms have been noticed by researchers in maritime engineering. The idea of VO algorithms had appeared in the 1980s, named as Collision Threat Parameter Area (CTPA) (Degre and Lefevre, 1981; Lenart, 1983). Subsequently, Pedersen et al. (2003) showed that this method can provide a better support for the Officer On Watch (OOW) in collision prevention comparing with traditional Automatic Radar Plotting Aid (ARPA). Later on, a series of studies proposed to use VO/CTPA algorithms for collision avoidance in various scenarios, e.g. restricted waters (Szlapczynski and Szlapczynska, 2017), multiple-ship (Szlapczynski, 2008), incorporating with regulations (Zhao et al., 2016), and unmanned ship (Kuwata et al., 2014). In (Huang et al., 2018), the algorithms which presume the target-ship keeps constant velocity, are concluded as a special case of the VO algorithm, called linear VO (LVO) algorithm. Details about the existing applications of VO algorithms in the maritime domain are addressed in Section 2.3.

The main challenge of using VO algorithms in ship collision prevention is the violation of the holonomic assumption in ship dynamics, i.e. the ship can change speed and course simultaneously. This assumption might not influence the performance of the VO algorithm when the distances between ships are large enough. However, in close range scenarios, this assumption might result in failure in collision prevention (as shown in Section 5.2.1). Thus, this article aims at finding a suitable VO algorithm which discards the holonomic assumption and is capable to handle the ship's dynamics. We used a generalized velocity obstacle (GVO) algorithm on the basis of (Bareiss and van den Berg, 2015) and modified it for collision prevention in the maritime environment. This method can be used for

unmanned ships and for supporting the OOW on board. The main contributions of our paper are as follows:

(1) A modified GVO algorithm considering the dynamics of ships has been used for collision prevention in the maritime environment, which can be used to support collision prevention with multiple dynamic obstacles in close range;
(2) We propose a Collision Avoidance System based on GVO algorithm (GVO-CAS), which visualizes the changes in course and speed resulting in collisions and facilitates collision prevention for both unmanned and manned ships.
(3) The navigation regulations are integrated into the proposed collision avoidance system. The collision avoidance actions can have a better compliance with Convention on the International Regulations for Preventing Collisions at Sea (COLREGs) (IMO, 1972).

The remaining parts of this paper are organized as follows. In Section 2, the studies on ship collision prevention, VO algorithms and their application to the maritime fields are reviewed. Section 3 provides details about the VO algorithm and modified GVO algorithm. The proposed collision avoidance system is presented in Section 4, followed by simulation experiments in Section 5. Section 6 discusses the major findings and Section 7 concludes this paper.

2. Related Work

2.1. Ship collision prevention techniques

In traditional maritime studies, many techniques have been developed for supporting collision prevention for the OOW, particularly in determining the evasive actions and the timing to apply these actions. Two main categories of collision prevention techniques have been identified in the literature.

One is using various indicators to measure a collision risk and find associated evasive actions that keep the risk at an acceptable level. Two widely used indicators, Distance to CPA (DCPA) and Time to CPA (TCPA), are based on the concept of closest point of approach (CPA) (Kearon, 1979). For the indicator-based methods, the threshold of risk is usually determined by the OOW, experts, and regulations (Goerlandt et al., 2015; Hilgert and Baldauf, 1997). Then, the OOW can try different evasive actions to modify the risk levels. The actions that can keep the risk at an acceptable level are collision-free solutions. A typical application of the methods in this category is called ARPA. However, these methods usually ignored the dynamics of the ship and assume that the target-ships sail with constant speed (Tam et al., 2009), which can easily result in failure when the target ships are within a close range. Recently, some improvements have been made by considering a simple ship dynamics (Wang et al., 2017; Zhang et al., 2015). In (Zhang et al., 2015), the authors provided a method to estimate ship's trajectory with discretized maneuvers of the ship. By checking the

trajectories, one collision-free maneuver is determined. However, the checking process is time-consuming, which limits the potential of the method for online collision avoidance.

The other category of collision prevention techniques is characterized by a warning ring surrounding the Own Ship (OS). They use extreme evasive actions, e.g. hard starboard, to determine the last moment/distance to apply the actions. Two well-known warning rings are ship domain and Minimum Distance to Collision (Montewka et al., 2014; Szlapczynski and Szlapczynska, 2016). In these studies, ship dynamics are employed to generate trajectories of the OS with given extreme actions (He et al., 2017; Zhang et al., 2012). By changing the initial conditions, the boundary conditions that result in collisions can be found. These boundary conditions are defined as action lines (Szlapczynski et al., 2018) or the last line of defend (Baldauf et al., 2017). These boundary conditions are used to trigger a collision alarm and to remind the OOW to take/not to take certain evasive actions. However, since they are not designed for general conflict resolution, they have some limitations: firstly, only two-ship encounter scenarios are considered; secondly, the target-ship is relatively ideal with unchanged speed and course. The performance of these methods in multiple-ship scenarios still needs further research.

Additionally, the development of Autonomous Surface Vehicle (ASV) motivates a number of new collision prevention methods which take the dynamics of ships into account. However, most existing algorithms only deal with a simple encounter scenario in which the static or semi-dynamic[1] obstacles are considered (Campbell et al., 2012; Liu et al., 2016). Moreover, they only offer one (sub-)optimal solution regarding cost function, such as optimization method (Johansen et al., 2016), artificial potential field (Lyu and Yin, 2017), which are less effective when they are used for supporting the OOW's decision making in various encountering scenarios. For instance, consider the encounter of two human-supervised ASVs, problems arise because the operators have no information about the decision process of the methods: how does human operators judge the "(sub)optimal solution" is acceptable or not; how can human operators take over control of an ASV without making the situation go worse; how can human operators know the alternative maneuvers are safe or not; which maneuvers should not be accepted in the current situation, etc.

In summary, as we can see from the literature, existing methods are usually suffering from the following limitations: (1) ignorance of ship dynamics or relative ideal ship dynamics model (Tam et al., 2009); (2) simple encounter scenarios (Campbell et al., 2012), e.g. encounter of two ships; (3) semi-dynamic target-ship (Liu et al., 2016; Tam et al., 2009); (4) only offer one evasive action and limited information.

[1] Semi-dynamic obstacle means the obstacle moves with constant course.

2.2. Velocity Obstacle Algorithm

VO algorithms bridge some of these gaps. The VO family is a successful velocity-based approach for collision avoidance with moving obstacles. Fiorini and Shiller formulated the basic idea of VO algorithms in (Fiorini and Shiller (1998). As this VO algorithm presumes the moving obstacle keeps its velocity, it is also named as LVO algorithm. To generalize this algorithm, Nonlinear VO (NLVO) algorithm is proposed (Large et al., 2005). Since then, a family of VO algorithms has been developed for preventing collisions in different scenarios. Probabilistic VO (PVO) algorithm is proposed to consider the uncertainty of obstacles' movement and it has applied in an experimental car to avoid collisions with pedestrians(Coue et al., 2006). Reciprocal VO (RVO) eliminates the oscillatory motion of the agents and the proposed method perform successfully in a crowd simulation with 1000 agents (van den Berg et al., 2011). Ellipse-Based VO (EBVO) (Lee et al., 2016) developed a VO algorithm deal with ellipse-shaped agents, which can apply to the human-shaped robot in a narrow environment. Moreover, Cooperative Collision Avoidance (CCA) offers a solution to handle non-holonomic constraint for the vehicle (Alonso-Mora et al., 2018). In (Wilkie et al., 2009), a GVO algorithm is proposed, which accept a relatively simple non-holonomic constraint on the vehicle. Based on that, Bareiss and van den Berg (2015) developed a general framework of GVO algorithm considering dynamics for unmanned vehicles, e.g. the wheel-based robot, quadrotors, (Bareiss and van den Berg, 2013; Bareiss and van den Berg, 2015), etc. The family of VO algorithms expands its potential in numerous applications.

2.3. Development of VO algorithm in maritime research

In maritime research, many methods have the same idea as VO algorithm but with various names such as CTPA in (Lenart, 1983), Collision Danger Sector in (Pedersen et al., 2003), and methods proposed in (Degre and Lefevre, 1981) and (Benjamin et al., 2006). These methods share the same characteristics with LVO algorithm (Huang et al., 2018): firstly, they all collect velocities which satisfy certain conditions (leading to collisions), i.e. DCPA is smaller than a threshold; secondly, the moving obstacles keep constant velocity; thirdly, the own-ship can adopt new course and speed simultaneously.

LVO algorithm is popular in maritime, due to its advantages over other methods. For one reason, LVO algorithm provides identical conflict detection results as DCPA/TCPA method. Moreover, it also provides the operators the conflict resolution. In particular, the LVO set can be integrated into a Radar system, which could make collision prevention more intuitive for the OOW (Pedersen et al., 2003). To make this method feasible in practice, many researchers considered various factors in the basic model, e.g. regulations (Kuwata et al., 2014), ship domains (Szlapczynski and Szlapczynska, 2015), restricted waters (Szlapczynski and Szlapczynska, 2017), etc. However, the application is still restricted due to its assumptions on the motion of obstacles. In Literature (Huang et al., 2018), the authors introduce

NLVO and PVO in the maritime environment in which expand its applications in the maritime field, which allows the ship to avoid a collision in more general cases. Moreover, NLVO also can be used to detect collision candidates from historical data for risk analysis (Chen et al., 2018b).

VO algorithms also attract much attention in the studies on ASVs. Many researchers have used the VO algorithms and tested them with ASV prototypes. Benjamin et al. (2006) used this method to eliminate dangerous solutions and search an optimal collision-free velocity for ASVs. In (Kuwata et al., 2014), the VO algorithm incorporated with COLREGs is proposed to obtain rule-compliant solutions. Additionally, Zhao et al. (2016) combined the VO algorithm with the experts' judgment system to meet regulations and seamanship.

In these studies, the dynamics of ships are usually simplified or ignored. Consequently, the obtained collision-free solution might not be feasible in close range. Therefore, in this paper, we proposed a ship collision avoidance system using GVO algorithm which considers the dynamics of the ship.

3. Generalized Velocity Obstacle Algorithm

In this section, the basic VO algorithm and GVO algorithm are introduced, respectively.

Notations are presented as follows: Sub-index $\cdot_i$ indicates the own-ship (OS) and Sub-index $\cdot_j$ refers to the target-ship (TS). The state of the system is denoted as a vector **x** which is in bold. The position and velocity are denoted as vectors **P** and **v**. The superscript index $\cdot^k$ indicates the value at time *k*. For example, $\mathbf{x}_i^k$ is the state of the OS at time *k*.

In this paper, ship collision is defined as follow:

Definition (1): Ship collision is an event in which two ships overlap each other at the same time.

According to Definition (1), a collision at time *t* is interpreted as the OS is located at a set of positions that results in overlap of two ships:

$$\mathbf{P}_i(t) \in \mathbf{P}_j(t) \oplus ConfP, \tag{1}$$

here, *ConfP* , which is an abbreviation of "conflict position", is a set of positions leading to an overlap of two ships; $\oplus$ is the Minkowski addition and $\mathbf{P}_j(t) \oplus ConfP$ means add all the elements in *ConfP* with $\mathbf{P}_j(t)$. Two illustrations of *ConfP* are shown in Fig. 1. The shape of *ConfP* is depending on the representation of the ship. If the ship is seen a circle, *ConfP* is a set of positions that lead to the overlap of two circles (the upper panel in Fig. 1), which is shaped as a circle, as well. Similarly, the *ConfP* of two ellipse-shaped ships is shown in the lower panel in Fig. 1. In this paper, we use the circular-shaped ship and *ConfP*.

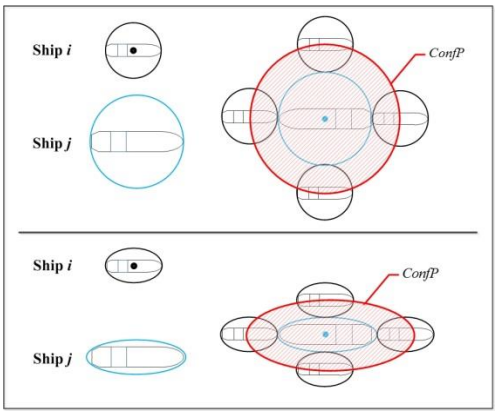

Fig. 1 Two representations of *ConfP*

3.1. Velocity obstacle algorithm

If the OS is regarded as a holonomic vehicle, i.e., it can change its velocity immediately, its position at time *t*, i.e. $\mathbf{P}_i(t)$, is calculated as:

$$\mathbf{P}_i(t) = \mathbf{P}_i(0) + \mathbf{v}_i \cdot t \cdot \tag{2}$$

Thus, a collision occurs at time *t*, if and only if $\mathbf{P}_i(t)$ is inside $\mathbf{P}_j(t) \oplus ConfP$. An illustration is shown in Fig. 2. If the OS keeps its velocity, a collision will happen at time *t*, i.e. Equation (1) is satisfied. Since there is still time before the collision, the OS can apply a new $\mathbf{v}_i$ to change its position at time *t* and prevent the collision. The relation of new $\mathbf{v}_i$ and new position $\mathbf{P}_i(t)$ is described in Equation (2). The idea of VO algorithm is using this relation to find all the new $\mathbf{v}_i$ leading to the position of the OS falling in $\mathbf{P}_j(t) \oplus ConfP$ and the rest of velocities are safe for the OS at time *t*.

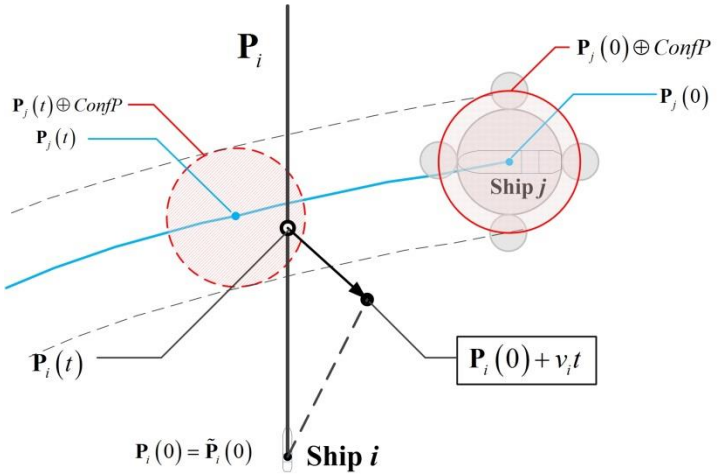

Fig. 2 Schematic sketch of the VO algorithm

Those $\mathbf{v}_i$ leading to a collision at time *t* can be formulated by substituting Equation (2) to Equation (1), which consist of a sub-VO set, denoted as $\mathrm{sVO}(t)$:

$$\mathbf{v}_i \in \frac{1}{t}\left[\left(\mathbf{P}_j(t) - \mathbf{P}_i(0)\right) \oplus ConfP\right] = \mathrm{sVO}(t) \cdot \tag{3}$$

Moreover, the set of velocities leading to a collision at any time in the future, i.e. VO set, is a union of all the sub-VO sets in the future:

$$\mathrm{VO} = \bigcup_{t}^{\infty} \mathrm{sVO}(t) \cdot \tag{4}$$

If the OS keeps a velocity in the VO set, a collision will happen in the future.

3.2. Generalized velocity obstacle algorithm

GVO algorithm is one of the VO algorithms, which is proposed to consider vehicle dynamics and generalize the basic VO algorithm. The GVO algorithm presented in this section is based on the original GVO from (Bareiss and van den Berg, 2015), while we assume the trajectory of other agents are known. The difference between the original GVO and the modified GVO in this paper can be found in Section 6.2.

Although the holonomic assumption is no longer hold, the derivation of GVO algorithm is presented as the same line of thinking as that of VO algorithm presented in Section 3.1. The algorithm firstly formulates the position of the OS at time *t* with respect to control inputs and the dynamics of ships. This step is the most challenging part due to the nonlinearity of ship dynamics. Subsequently, a set of controls leading to a collision at time *t* is found, named as a sub-UO set, i.e. sUO set. Then, the union of sUO sets is the UO set, i.e., a set collecting all the controls of the OS resulting in collisions.

Let the dynamics of the OS being described as a nonlinear ordinary differential equation (ODE):

$$\dot{\mathbf{x}}_i = \mathbf{f}_i\left(\mathbf{x}_i, \mathbf{u}_i\right), \tag{5}$$

where $\mathbf{u}_i$ is control input, $\mathbf{f}_i$ is a continuous-time equation of motion and $\mathbf{x}$ the system state containing ship position $\mathbf{P}$, velocity, etc. Thus, we obtain the position of the ship from the system state via:

$$\mathbf{P}_i(t) = C \cdot \mathbf{x}_i(t), \tag{6}$$

where $C = [\mathbf{I}^{2\times 2}, \mathbf{0}^{2\times 4}]$ contains a 2-by-2 identical matrix and a 2-by-4 zero matrix.

The GVO algorithm consists of the following steps:

Firstly, the mathematical expression of the system $\mathbf{x}_i(t)$ in terms of control input $\mathbf{u}_i$ is needed. Since this system is nonlinear, a linearization of this system about an estimated trajectory is needed. The estimated trajectory is a trajectory of the system without additional inputs, noted as $\tilde{\mathbf{x}}_i$, which is calculated via Runge-Kutta Integration with known initial state $\mathbf{x}^0$ and input $\mathbf{u}^0$. Fig. 3 shows the estimated trajectory of the OS ($\tilde{P}_i$). In this way, the state of the system can be formulated as:

$$\mathbf{x}_i(t) \approx \int_0^t \mathbf{f}_i\left(\mathbf{x}^0, \mathbf{u}^0\right) d\tau + \int_0^t \Delta\dot{\mathbf{x}}_i(\tau) d\tau = \tilde{\mathbf{x}}_i(t) + G(t)\Delta\mathbf{u}_i, \tag{7}$$

where $\Delta\mathbf{u}_i$ is the difference between $\mathbf{u}^0$ and real input, i.e. $\Delta\mathbf{u}_i = \mathbf{u}_i - \mathbf{u}_i(0)$, and $G(t) = \int_0^t e^{A(t-\tau)} B d\tau$, with $A = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}^0, \mathbf{u}^0}$ and $B = \left.\frac{\partial \mathbf{f}}{\partial u}\right|_{\mathbf{x}^0, \mathbf{u}^0}$. When we keep the initial input $\mathbf{u}^0$, i.e. $\Delta\mathbf{u}_i = \mathbf{0}$, the trajectory of the OS is equal to the estimated trajectory ($\tilde{P}_i$ in Fig. 3). When $\Delta\mathbf{u}_i \neq \mathbf{0}$, the position of the OS is pushed away from $\tilde{P}_i$, calculated with $\mathbf{x}_i(t)$ in Equation (7), $\tilde{P}_i + CG(t)\Delta\mathbf{u}_i$.

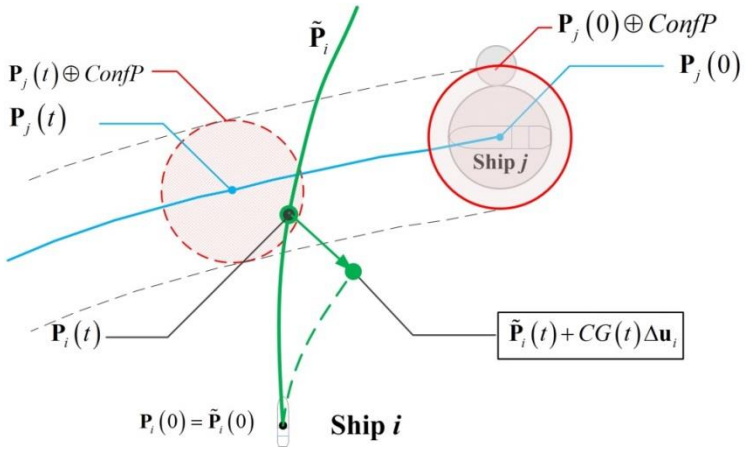

Fig. 3 Schematic sketch of the GVO algorithm

Secondly, sUO set is formulated. sUO set collects all the $\Delta\mathbf{u}_i$ resulting in the OS inside the $\mathbf{P}_j(t) \oplus ConfP$. Specifically, by substituting Equations (6) and (7) back into Equation (1), we have:

$$C\tilde{\mathbf{x}}_i(t) + CG(t)\Delta\mathbf{u}_i \in \mathbf{P}_j(t) \oplus ConfP. \tag{8}$$

Solving this equation, regarding $\Delta\mathbf{u}_i$, we obtain:

$$\Delta\mathbf{u}_i \in \left(CG(t)\right)^{-1} \cdot \left[-\left(\tilde{\mathbf{P}}_i(t) - \mathbf{P}_j(t)\right) \oplus ConfP\right] = \mathrm{sUO}(t). \tag{9}$$

This equation collects all $\Delta\mathbf{u}_i$ leading to a collision at time *t*. It means that Equation (1) will be true, i.e. a collision happens at time *t*, if $\Delta\mathbf{u}_i$ of the OS's control fall in this set and the ship keeps the control $\left(\mathbf{u}_i^{\,0} + \Delta\mathbf{u}_i\right)$ till time *t*.

Lastly, the UO set is constructed. The UO set is a generalized VO set, which contains the changes of control leading to collisions. Therefore, any $\Delta\mathbf{u}_i$ outside this set is collision-free. By combining all these sUO sets, we obtain the UO set:

$$\mathrm{UO} = \bigcup_t^{\infty} \mathrm{sUO}(t). \tag{10}$$

4. Collision Avoidance System Using GVO Algorithm

4.1. The framework of collision avoidance system

With the above-mentioned GVO algorithm, we design a ship collision avoidance system (GVO-CAS). We consider the situation in which the OS follows a sequence of waypoints and takes evasive actions using GVO-CAS when encountered with moving obstacles. Fig. 4 shows the GVO-CAS consisting of three main modules:

Global Planner generates the planned waypoint for the system which guides the ship to the destination, noted as *wp*.

Local Planner generates a desired velocity considering planned waypoints, observed the states of ships (include the OS and the TS) and regulations, noted as $\mathbf{u}^*$ (e.g. speed and course). In this module, the GVO algorithm is applied.

Controller calculates the control forces $\boldsymbol{\tau}$ required to follow the desired velocity.

Details about the settings of the modules are introduced in following subsections.

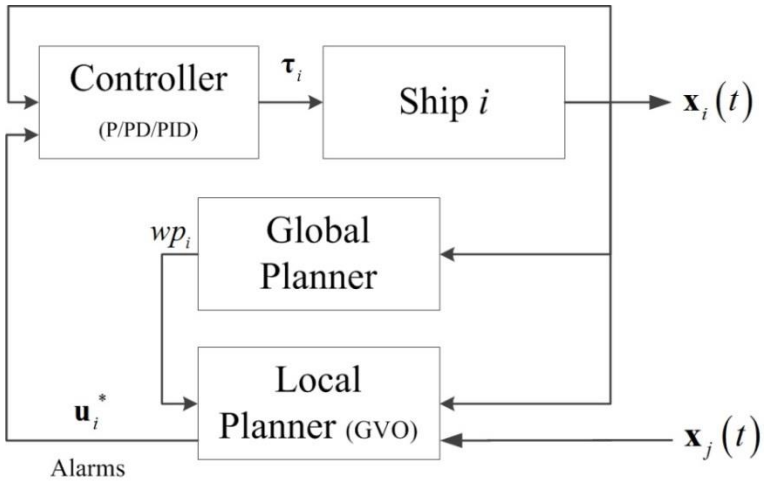

Fig. 4 The framework of GVO-CAS

As shown in Fig. 4, collision avoidance using the GVO-CAS consists of following steps:

Step 1: The states of the OS and the TS ($x_i(t)$ and $x_j(t)$) are collected and send to the global planner and local planner.

Step 2: Global planner compares the states of the own-ship with a series of waypoints and decides its active waypoint wp_i.

Step 3: Local planner searches a collision-free velocity $\mathbf{u}^*$ for the OS via using GVO algorithm.

Step 4: A feedback controller calculates the forces $\boldsymbol{\tau}$ to reach the desired velocity.

Step 5: The actuators of OS are assigned to generate the planned forces.

During the collision avoidance process, the following assumptions on information are made:

(1) The planned waypoints are known in advance;

(2) The dynamics model of the own-ship is completely known;

(3) The ship can obtain the trajectory of obstacles (target-ships);

(4) The shape of the ship is represented as a circle with a diameter equal to its length.

4.2. Ship dynamics model

A ship dynamics model with 3 degrees of freedom (DOF) in (Fossen, 2002) is employed:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \dot{\mathbf{x}} = \begin{bmatrix} \mathbf{R}(\psi)\mathbf{v} \\ -\mathbf{C}(\mathbf{v})\mathbf{v} - \mathbf{D}(\mathbf{v})\mathbf{v} \end{bmatrix} + \mathrm{B}\boldsymbol{\tau}, \tag{11}$$

where $\mathbf{x}$ is system state contains coordinates (x, y), heading angle ψ, and vector $\mathbf{v}$ consisting of linear velocities (u, v) and angular velocity r. $\boldsymbol{\tau}$ is force vector. **M**, **C(v)**, **D(v)**, and $\mathbf{R}(\psi)$ are inertia matrix, Coriolis–centripetal matrix, damping matrix, and rotation matrix, respectively. Details refer to (Fossen, 2002). $\mathrm{B}=\begin{bmatrix} \mathbf{0}^{3\times 3} & \mathbf{I}^{3\times 3} \end{bmatrix}^{\mathrm{T}}$ consists of a 3-by-3 identical matrix and a 3-by-3 zero matrix.

4.3. Controller

A controller is designed to control the state of the system to reference velocity. Well known controllers are the Proportional-Integral-Derivative (PID) controllers and robust H_∞ controllers (Van Gelder, 1991). In this paper, we apply a Proportional-Derivative (PD) controller. Control input is formulated as:

$$\boldsymbol{\tau} = \mathrm{K_p}\left(\mathbf{u}^* - \mathrm{V}\mathbf{x}\right) - \mathrm{K_d}\mathrm{V}\dot{\mathbf{x}}, \tag{12}$$

where $\mathrm{K_p}$ and $\mathrm{K_d}$ are non-negative feedback gains; V is the observe matrix and $\mathbf{u}^*$ is the desired velocity:

$$\mathbf{u}^* = \begin{bmatrix} u^* \\ v^* \\ \psi^* \end{bmatrix}, \quad \mathrm{V} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

By substituting Equation (12) into (11), the control input is changed from the force $\boldsymbol{\tau}$ to the desired velocity $\mathbf{u}^*$:

$$\left(\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} + \mathrm{B}\mathrm{K_d}\mathrm{V} \right) \dot{\mathbf{x}} = \begin{bmatrix} \mathbf{R}(\psi)\mathbf{v} \\ -\mathbf{C}(\mathbf{v})\mathbf{v} - \mathbf{D}(\mathbf{v})\mathbf{v} - \mathrm{K_p}\mathrm{V}\mathbf{x} \end{bmatrix} + \mathrm{B}\mathrm{K_p}\mathbf{u}^*. \tag{13}$$

4.4. Local planner

Local planner module is the core of GVO-CAS. It has four sub-components, see Fig. 5.

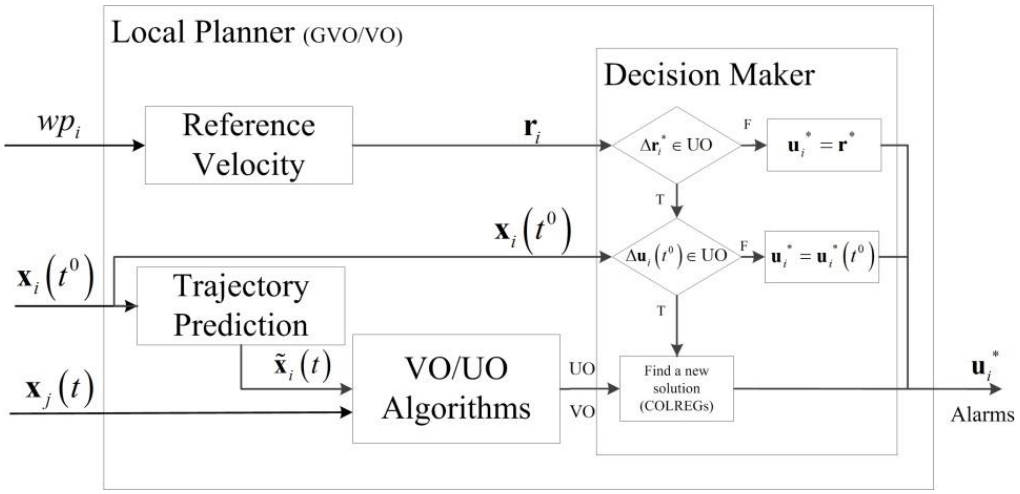

Fig. 5 The framework of a Local Planner

4.4.1 Reference velocity module

"Reference Velocity" module receives waypoints and produces a reference velocity. In this paper, the reference velocity is set as:

$$\mathbf{r}_i(t^0) = \left[u_{eco}, 0, \psi_{ref} + \psi\right]^{\mathrm{T}}, \tag{14}$$

where u_{eco} is a constant sway speed which is the economical speed of the OS, and ψ_{ref} is the relative bearing of the waypoints.

4.4.2 Trajectory prediction module

"Trajectory Prediction" module calculates the estimated trajectory of the OS regarding its initial desired velocity ($\mathbf{u}^0$) and initial state ($\mathbf{x}^0$), i.e. $\tilde{\mathbf{x}}_i(t)$. Here, Runge-Kutta Integration is employed.

4.4.3 UO algorithm module

This module generates UO sets via Equation (5)-(10). Since control input is changed to the desired velocity $\mathbf{u}^*$ via Equation (13), the change of control refers to the difference between the chosen desired velocity and the initial desired velocity $\mathbf{u}^0$, i.e. $\Delta\mathbf{u}^* = \mathbf{u}^* - \mathbf{u}^0$. Moreover, the space where UO set is presented is named as ΔU space.

Since $\mathbf{u}^*$ has three variables, the ΔU space has 3 dimensions, which might not be easy for the OOW to interpret. Thus, we introduce two Remarks to facilitate the OOW to read the UO set:

> **Remark 1:** The desired sway speed v^* is always set to be 0 and the desired velocity $\mathbf{u}^*$ remains with two degrees of freedom: u^* and ψ^*.

With Remark 1, ΔU space is reduced to 2 dimensions and can be displayed in the two-dimensional Cartesian coordinate system (e.g. Fig. 8 (1)). Moreover, we denote the horizontal axis as the difference in the heading ($\Delta\psi^*$) and the vertical axis as the difference in surge speed (Δu^*).

Remark 2: The origin of the space which UO set is presented is moved from the initial desired velocity $\mathbf{u}^0$ to the real velocity at the present time, i.e. V$\mathbf{x}$.

Remark 2 moves the origin of ΔU space from the initial desired velocity to the real velocity at present time. As a result, the negative half-plane always means the port-side turns regarding the real velocity. Additionally, any point in this space represents the difference between the alternative desired velocity and the real velocity at present time, i.e. $\Delta\mathbf{u}^* = \mathbf{u}^* - \mathrm{V}\mathbf{x}$.

Realisations of the modified ΔU space and UO set are shown in Fig. 8.

4.4.4 Decision maker module

"Decision-maker" module is designed to find a collision-free solution for the OS. Some vectors used are defined as: the vector $\Delta\mathbf{r}$ is the reference velocity in ΔU space which is formulated as: $\Delta\mathbf{r} = \mathbf{r} - \mathrm{V}\mathbf{x}$; the vector $\Delta\mathbf{u}^0$ is the initial desired velocity in ΔU space, i.e. $\Delta\mathbf{u}^0 = \mathbf{u}^0 - \mathrm{V}\mathbf{x}$; the vector $\Delta\mathbf{u}$ is alternative desired velocity in ΔU space, i.e. $\Delta\mathbf{u} = \mathbf{u}^* - \mathrm{V}\mathbf{x}$.

The following rules are employed to choose a new desired velocity and they are presented in order of priority.

Rule 1: The OS is expected to choose reference velocity $\mathbf{r}$ if $\Delta\mathbf{r} \notin \mathrm{UO}$;

Rule 2: The OS prefers to continue with its initial desired velocity $\mathbf{u}^0$ when $\Delta\mathbf{u}^0 \notin \mathrm{UO}$;

Rule 3: A new $\mathbf{u}^*$ should keep its current speed, avoid port-side turn and satisfy $\Delta\mathbf{u} \notin \mathrm{UO}$;

Rule 4: A new $\mathbf{u}^*$ is close to its current velocity and satisfies $\Delta\mathbf{u} \notin \mathrm{UO}$;

Rule 1 means when the reference velocity is collision-free, the ship will choose the reference velocity. This rule is with the highest priority.

Rule 2 indicates that if the current desired velocity is collision-free and reference velocity is not, the ship's controller will still implement the current desired velocity.

Rule 3 shows the principle of the OS to find a collision-free solution compliant with COLREGs when the current desired velocity and reference velocity are both unsafe. The expression "keep its current speed" follows Rule 8 in COLREGs, which encourages the ship to find a collision-free solution via

turning the course. Besides, “avoid port-side turn” is introduced to comply with Rule 14, 15 and 17 in COLREGs (the relevant rules are listed in Appendix B).

Rule 4 is introduced when there is no COLREGs-compliant and collision-free solution within the feasible range. In this urgent case, we will find a collision-free control which has the lowest cost, i.e. the control closest to the current system’s state. The feasible range is framed by kinematic constraints, e.g. maximum speed etc.

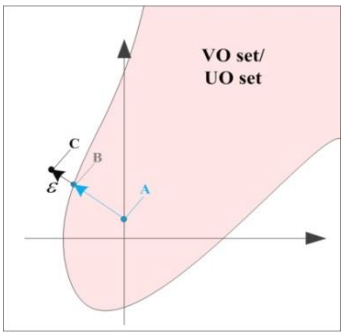

Fig. 6 Optimal solution $\mathrm{AC} = (1+\varepsilon)\mathrm{AB}$ in a feasible range by UO/VO set

In Rule 3 and 4, we utilize UO set to find the best collision-free solution. The solution is on the boundary of the UO set. However, when we choose the solution on the boundary, two ships will get infinite close to each other. In fact, two ships will be tangent to each other. To avoid this situation, we add a small value ε to the best solution. As shown in Fig. 6, Point A indicates the initial desired velocity, which is inside the UO set; Point B is the closest point to A on the UO set, and it is the suggested collision-free desired velocity; Point C is the final solution we adopt, which can be formulated as: $\mathrm{AC} = (1+\varepsilon)\mathrm{AB}$.

5. Case Study

In this section, simulation experiments are carried out to show the effectiveness of the proposed GVO-CAS in various encounter scenarios.

5.1. Setup

In this section, a scale model of the ship, “CyberShip II” (Skjetne et al., 2004), is employed. Details of this model ship are attached in Appendix A. The ship is seen as a circle. Accordingly, a set of positions leading to collisions (*ConfP*) is shaped like a circle with radius R and is defined as:

$$ConfP(\mathrm{R}) = \left\{ \mathbf{P} \middle| \left\| \mathbf{P} - \mathbf{O} \right\|_2 \le \mathrm{R} \right\}, \tag{15}$$

where R is the sum of two ships’ radius; $\mathbf{O}$ is the origin and $\mathbf{P}$ is a position vector.

Feedback gains K_p and K_d in Equation (13) are set as: K_p = diag([200, 10, 10]) and K_d = diag([2, 2, 2]). The economical speed in Equation (14) is $u_{eco} = 0.5$ [m/s]. That means the ship prefers to sail 0.5 [m/s]

in the scaled world, i.e. 8 [knot] in the real-size world. The small value ε used in Rule 3 and Rule 4 is set as 0.01.

Some kinematic constraints are considered to determine a feasible range. Specifically, the maximum of turning is $\pm 90^{\circ}$, the minimal speed of the ship is 0 [m/s], and maximal speed is 1 [m/s].

A simulator, which is employed to simulate the encountering scenarios, is developed in Matlab 2016b platform. The frequency of the control sequence is set as 1 Hz. The prediction horizon is 80[s]. Runge-Kutta integration uses 0.1 [s] time-step. The *ConfP* is approximated as a 15-sided polygon.

5.2. Two-ship case

A series of heading scenarios is designed to compare VO and GVO algorithms. Additionally, a crossing scenario is used to show how the proposed GVO-CAS find a solution that complies with navigational regulations.

Two ships are involved in the following scenarios, i.e., OS and TS. Both ships are "CyberShip II" with a length of 1.255 [m]. Thus, the radius of *ConfP* and safety distance are both 1.255 [m]. The OS is placed at origin heading to the North and its waypoint is set as (0, 28) [m].

5.2.1 Heading scenarios

The performance of VO algorithm and GVO algorithm in ship collision scenarios are compared in this section. We simulate a series of heading scenarios in which the relative distances between OS and TS are different. When applying the VO algorithm, we use the VO set to find a collision-free solution instead of UO set. Correspondingly, Rule 1~4 in Section 4.4 is modified as:

(1) The OS is expected to choose its reference velocity, if $\mathbf{r} \notin \mathrm{VO}$;

(2) The OS prefers to continue with its initial desired velocity $\mathbf{u}^0$, when $\mathbf{u}^0 \notin \mathrm{VO}$;

(3) A new $\mathbf{u}^*$ is chosen, which keeps current speed, avoids port-side turn and satisfies $\mathbf{u}^* \notin \mathrm{VO}$;

(4) A new $\mathbf{u}^*$ is chosen, which is close to its current velocity and satisfies $\mathbf{u}^* \notin \mathrm{VO}$;

The TSs in these scenarios have an identical motion feature but different initial positions. The sketch diagram is shown in Fig. 7. The distance between the OS and TS increases from $5L_i$ (5 times of the OS's length) to $16L_i$. In total, 12 scenarios are considered. The details are shown in Table 2.

The results are shown in Table 3. If we apply the VO algorithm in these scenarios, the safety constraint might not always be satisfied. Particularly, when the distance is smaller than $9L_i$, we cannot avoid collision using the VO algorithm. On the contrary, the GVO algorithm performs properly in these scenarios.

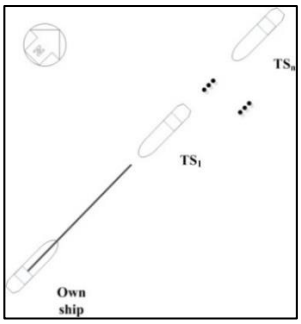

Fig. 7 A series of heading scenarios

Table 2 Setting of heading scenarios

	Position [m]	Speed [m/s]	Course [°]
OS	[0,0]	0.5	000
TS_k^*	$\left[0,\left(k+4\right)L_i\right]$	0.5	180

Note *: k = 1, 2...12. L_i is the length of the OS.

Table 3 Results of Collision Avoidance by VO/GVO algorithms

	Target Ship	Initial Dist. [L_i]	VO Algorithm			GVO algorithm		
			Minimal Dist. [m]	Successful Avoidance	No. of $\mathbf{u}^*$	Minimal Dist. [m]	Successful Avoidance	No. of $\mathbf{u}^*$
Heading 1	TS_1	5	0.992	×	5	1.730	√	1
Heading 2	TS_2	6	1.134	×	6	1.631	√	1
Heading 3	TS_3	7	1.154	×	8	1.434	√	1
Heading 4	TS_4	8	1.169	×	7	1.408	√	1
Heading 5	TS_5	9	1.276	√	6	1.429	√	1
Heading 6	TS_6	10	1.225	×	6	1.339	√	1
Heading 7	TS_7	11	1.265	√	4	1.327	√	1
Heading 8	TS_8	12	1.357	√	4	1.355	√	1
Heading 9	TS_9	13	1.325	√	4	1.349	√	1
Heading 10	TS_{10}	14	1.2547	×	2	1.294	√	1
Heading 11	TS_{11}	15	1.298	√	2	1.322	√	1
Heading 12	TS_{12}	16	1.395	√	1	1.389	√	1

Note: “Dist.” is an abbreviation of “distance” and the value represents the multiplication factor of the OS’s length $L_{i.}$ Safety distance is 1.255 [m].

Moreover, in the scenarios (Heading 9, 11, 12, 13, 15) in which the OS and the TS successfully avoid a collision, the VO algorithms require the OS to take more evasive actions than the GVO algorithm. The main reason is that the dynamics of the ship is neglected in the VO algorithm. When the VO algorithm offers one collision-free velocity ($\mathbf{u}^*$) to the OS, the OS needs time to reach the desired velocity and this process needs time and space. After the OS reaches $\mathbf{u}^*$, the encountering situation has been changed. In the new situation, the previous $\mathbf{u}^*$ might become unsafe and a new collision-free velocity is needed. However, the GVO algorithm considers the dynamics of the ship, i.e., the algorithm takes the action time and space into consideration. Following the solution provides by the GVO algorithm, the OS only needs one control to avoid a collision.

From these scenarios, we can see that the GVO algorithm can perform better in close range scenarios. Moreover, when applying the GVO algorithm, we can avoid the succession of small actions, which is advocated by regulations.

5.2.2 Crossing scenario

A crossing scenario between two ships is simulated and the encounter situation is shown in the left panel of Fig. 8. The settings are listed in Table 4.

Fig 8 (1)-(3) show the UO sets at different time slices. In these figures, x-axis and y-axis represent the changes in heading and speed, respectively. According to Remark 2, the origin refers to no changes in heading or speed, i.e. $\Delta\mathbf{u} = (0,0)$, which implies the OS keeps its velocity at present. The "*" in red is the reference velocity shown in ΔU space and "o" in blue is the suggested control by the proposed system, i.e. new desired velocity $\mathbf{u}^*$.

Table 4 Settings of crossing scenario

	Position [m]	Speed [m/s]	Course [°]
OS	[0,0]	0.5	000
TS	$\left[5L_i, 5\left(1+\sqrt{2}\right)L_i\right]$	0.5	-145

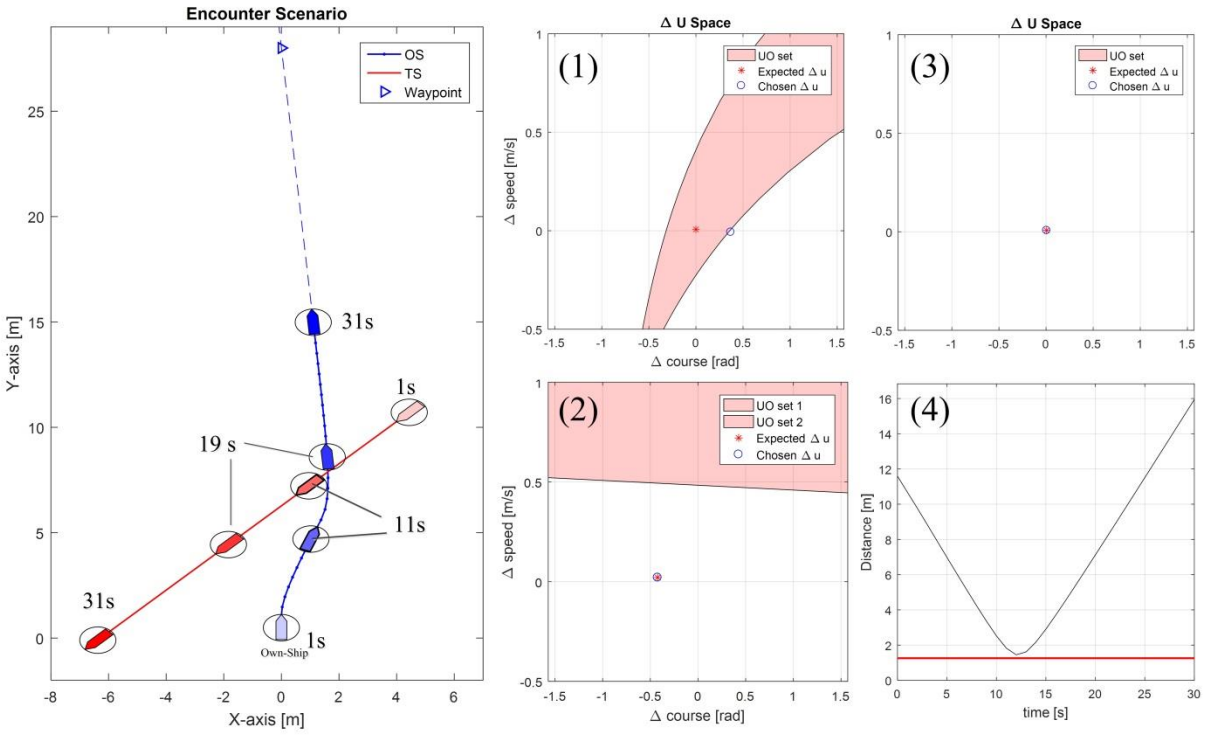

Fig. 8 Encounter Situation, UO set and relative distance at different time slices

* Note: (1) UO set at 1 [s]; (2) UO set at 11 [s]; (3) UO set at 19 [s]; (4) Relative Distance over time (details at each time slice are shown in Appendix C).

Fig. 8 (1) shows ΔU space at t = 1 [s]. At this moment, the OS is sailing at its reference velocity and this velocity is its initial desired velocity. However, GVO-CAS detects that Δr ("*" in red) and Δu^0 are both inside the UO set. That implies Rule 1-2 are not held and a collision danger is in the near

future. Therefore, Rule 3 is activated, and the system searches a new solution in the positive direction of the x-axis. Consequently, a solution $\Delta u = (0, 0.3688)$ is found ("o" in blue): the OS is asked to keep its speed and turn for 0.3688 [rad] (or 21 [$^\circ$]). The desired velocity (u^*) then is the sum of velocity at present Vx and Δu .

Fig. 8 (2) shows ΔU space at t = 11 [s]. Before this moment, the OS keeps the desired velocity obtained at 1 [s]. However, then, GVO-CAS detects Δr is out of the UO set, which activates Rule 1 in Section 4.4.4. Though the TS is just located on the straight-line between the OS and its waypoint, the ship chooses to head to its waypoint. This solution requests the OS to turn port-side and keep speed close to current speed.

In the following time, if there are no new collision conflict with other ship, the OS will keep its reference velocity r and heading to the waypoint. In Fig. 8 (3), we present ΔU space at 19 [s]. Since there is no UO set and the desired velocity (also reference velocity) is collision-free, the OS is suggested to keep its desired velocity. The relative distance over time in Fig. 8 (4) shows that the GVO-CAS works properly and prevents a collision in this case.

5.3. Multiple-ship case

A multiple-ship scenario is designed to show the potential of the GVO-CAS in a more complicated case. Three "CyberShip II" ships are employed, and they all have GVO-CAS on board to determine their evasive action. The encounter situation is displayed in Fig. 9. The safety distance is set as 1.255 [m]. Other parameters are shown in Table 5.

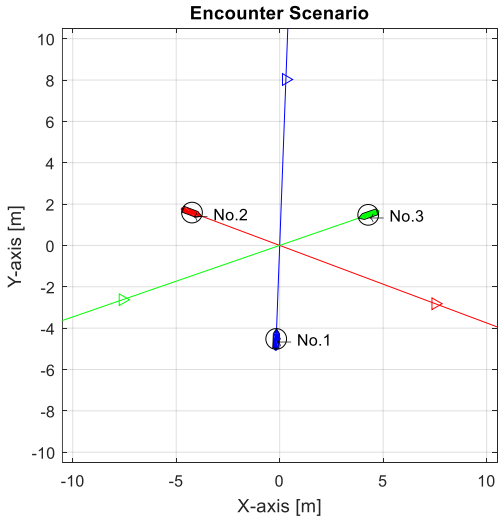

Fig. 9. Encounter Situation at 0 [s]

In GVO-CAS, we presume each ship is capable to know the trajectories of other ships. Thus, in the simulation, ships broadcast their trajectories sequentially: when one ship choses an evasive action, the updated trajectory will be broadcasted to the others; then, the next ship finds a collision-free solution according to the updated trajectory and broadcasts it; this process continues until all the ships find

their solutions. Additionally, if one ship could not find an available solution, it will keep its current state and other ships will try to find evasive actions. These setting results in a cooperative collision scenario.

Table 5 The setting of ships in Multiple-ship Scenario

	Position [m]	Speed [m/s]	Course [°]	Destination [m]
Ship 1	$[-0.19, -5.02]$	0.5	002	$[0.30, 8.03]$
Ship 2	$[-4.70, 1.76]$	0.5	200	$[7.52, -2.82]$
Ship 3	$[4.74, 1.64]$	0.5	251	$[-7.59, -2.62]$

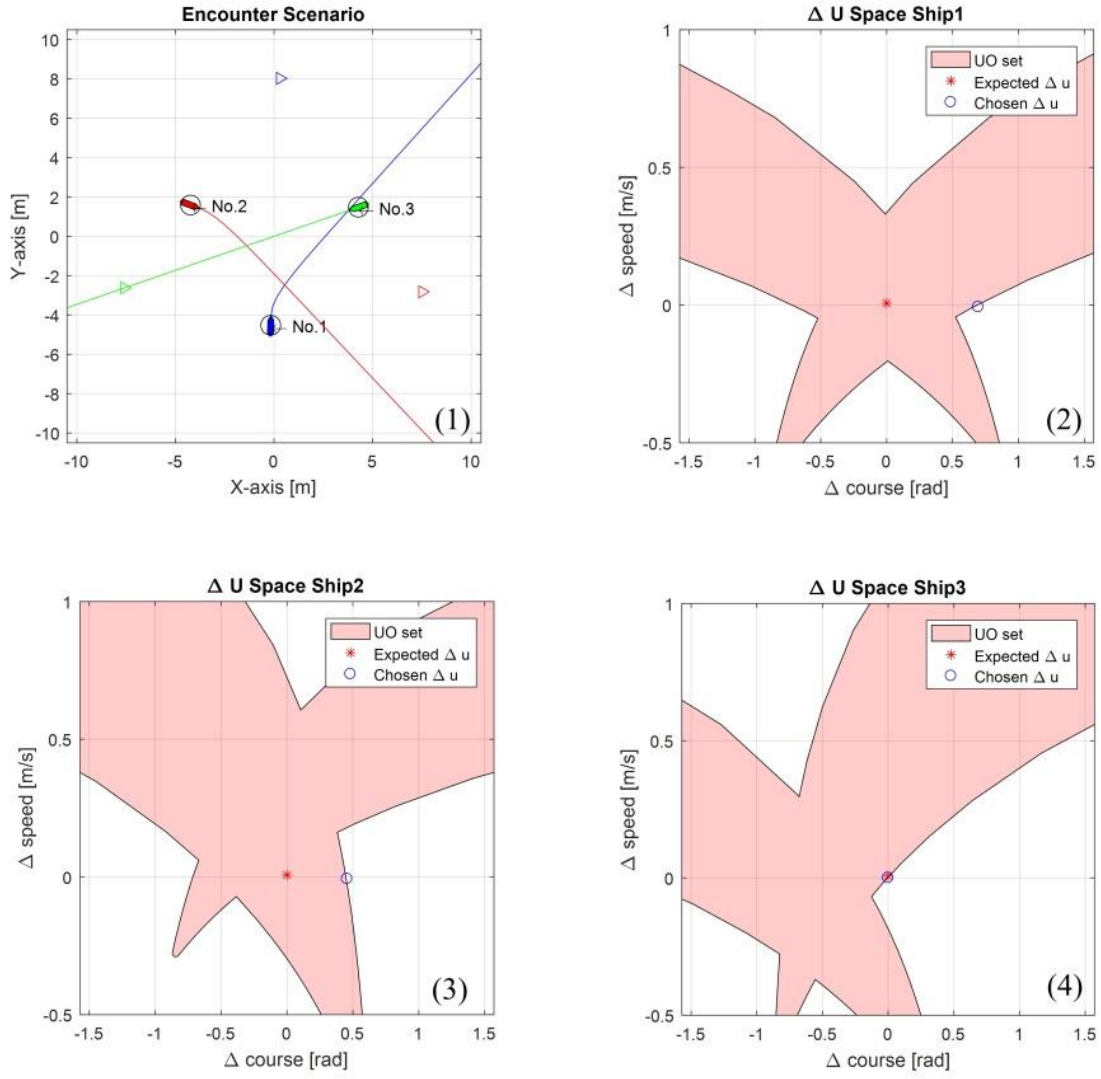

Fig. 10. Encounter Situation and UO sets from the perspective of different ships at 1 [s]

Fig. 10 shows the system states and ΔU spaces of ships at time 1 [s]. Fig. 10 (1) shows the encounter scenario at 1 [s] and all the ships have found their collision-free solutions. Fig. 10 (2)-(4) show how Ship 1, Ship 2, and Ship 3 choose a collision-free solution based on UO sets, respectively. Ship 1 is

the first ship to search its evasive action. As shown in Fig. 10 (2), a starboard turn solution was found, and the ship updated its trajectory and broadcasted the changes to the others, see the blue line in Fig. 10 (1). Then, Ship 2 determines its evasive action according to the initial trajectory of Ship 3 and the updated trajectory of Ship 1. In the end, the Ship 3 search its evasive action based on the updated information from Ship 1 and Ship 2.

In Fig. 11, we show the historical trajectory of the ship at time 25 [s] and the relative distance over time. From the right panel in Fig. 11, all ships successfully avoid collision using GVO-CAS.

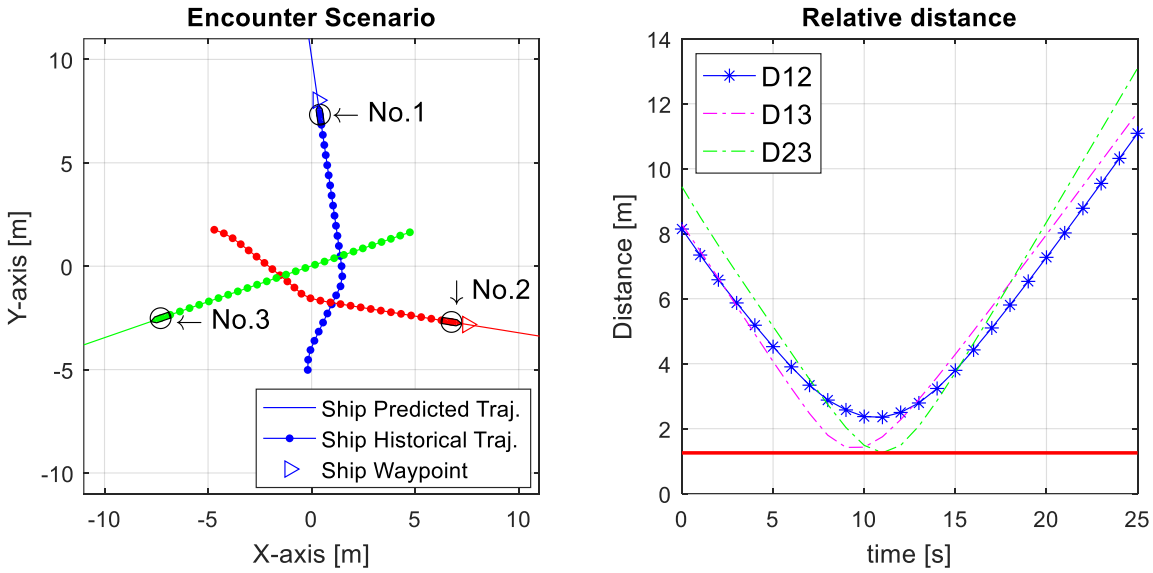

Fig. 11. Encounter Situation and relative distance over time [s]

*Note: 'D12' means the distance between Ship 1 and Ship2, 'D13' is the distance between Ship 1 and 3, and 'D23' is the distance between Ship 2 and 3.

5.4. Discussion of case study

Heading scenarios compare the performance of the VO and GVO algorithm. When two ships have sufficient initial relative distance, both the VO algorithm and the GVO algorithm can find collision-free solutions. However, when the initial distance decreases, the VO algorithm fails to find a collision-free solution. Moreover, the VO algorithm usually offers a series of small maneuvers, which is not encouraged by COLREGs (Rule 8(b)). From these perspectives, the GVO algorithm performs better in conflict resolution than the VO algorithm under the same condition.

The crossing scenario shows the capability of the proposed system in offering regulation-compliant solutions. Here, we demonstrate how the GVO-CAS searches a collision-free solution automatically. Nevertheless, this method also can be used for supporting the OOWs. Specifically, the OOWs can either choose the solution suggested by GVO-CAS or choose any collision-free solutions outside the UO set themselves.

The multiple-ship scenario shows the potential of the GVO-CAS in compliant encounter situations. Using the proposed system, the ships can cooperate with each other and search for the collision-free solution automatically. In the simulation, the order of ships finding solutions are predetermined: Ship 1 finds its solution firstly; then Ship 2 starts searching; and, Ship 3 is the last one. This arrangement, in fact, gives Ship 3 a higher priority than Ship 2 and Ship 1. As we shown in Fig. 11, the routes of Ship 1 and Ship 2 are longer than the route of Ship 3. Ship 3 might even not need to take evasive action if the actions Ship 1 and Ship 2 took have solved the collision conflicts. As cooperation is not the focus of this paper, we only introduce this mode with priority as an example of cooperation. For the reader interested in this theme, more cooperation modes are found in (Chen et al., 2018a).

6. Discussion

In this section, we carry out a comparison between the GVO algorithm and the VO algorithm in the maritime environment. We also illustrate the differences between the modified GVO algorithm in this work and other related works. Besides, the potential of the UO set and GVO-CAS are also discussed.

6.1. Comparison between the GVO and the VO algorithms for ship collision prevention

This paper shows that both VO and GVO algorithm can support collision prevention for the ship. However, the GVO algorithm considered the dynamics of ships. Thus, the GVO algorithm has better performance in providing a more reliable solution, and fewer control actions are needed for collision avoidance than the VO algorithm.

From a theoretical perspective, the GVO algorithm is more suitable for ships in the dynamic environment than the VO algorithm. The derivations of the basic VO and GVO algorithms both start from the same collision condition, i.e. Equation (1): $\mathbf{P}_i(t) \in \mathbf{P}_j(t) \oplus ConfP$. The disparity starts from the calculation of the OS's trajectories $\mathbf{P}_i(t)$. The VO algorithm assumes that the ship can change its velocity immediately, while in GVO algorithm, the calculation of $\mathbf{P}_i(t)$ depends on the ship's dynamics. Therefore, $\mathbf{P}_i(t)$ calculated by the second method is closer to the real ship's trajectory. Thus, the GVO algorithm is more reliable than the VO algorithm in conflict detection and resolution.

Since the GVO algorithm is a generalized version of the VO algorithm, some developments of the VO algorithms can be directly applied to the GVO algorithm. For example, in the Multiple-ship case in Section 5.3, the GVO algorithm is capable to deal with non-linear trajectories of ships, which was learned from the non-linear VO algorithm. Thus, it is possible to integrate other VO algorithms in GVO algorithm such as PVO algorithm(Coue et al., 2006), EBVO algorithm(Lee et al., 2016), RVO algorithm(van den Berg et al., 2011), etc., which can expand its application at sea.

6.2. Comparison of GVO algorithm with related works

The GVO algorithm in this paper is based on the original GVO in (Bareiss and van den Berg, 2015). The original GVO algorithm is proposed for unmanned vehicles, such as wheel-based robots, drones, etc., in a lab environment. Due to the differences of operating environments and application purposes, some modifications have been made to accommodate to the maritime environment.

First of all, the existing GVO method assumes that each participant has complete information about itself and others in advance. However, this is not realistic in the maritime environment. Therefore, in the current method, the dynamics of the ship and the controls are not necessary to be completely known or observable to each other. Instead, we assume that the ships can exchange their predicted trajectories, which is more realistic in the maritime practice.

Secondly, although the existing GVO method can find collision-free options, they only choose one optimal solution for unmanned vehicles. Readers also can see (Best et al., 2017; Zuhaib et al., 2017) for more cases. In this paper, we apply GVO as a decision supporting tool for human operators. In GVO-CAS, the visualization of UO set and ΔU spaces make the decision process of GVO clear for human operators. This characteristic makes that the GVO-CAS not only can be used to control ASVs, but can also support decision making of the OOW on manned ships.

Thirdly, the original GVO methods find an optimal solution via minimizing the changes in control, irrespective of the rule of the road. However, in the maritime environment, the evasive actions should be taken in compliance with navigation regulations. Thus, this article finds a collision-free solution incorporating COLREGs instead of only minimizing changes in controls.

6.3. The feature of the UO set and its potential

Different from many popular conflict resolution methods, such as potential fields (Daily and Bevly, 2008), model predictive control (Johansen et al., 2016), the GVO algorithm does not seek for one optimal control or one route for the ship. Instead, it collects the controls leading to a collision, i.e. the UO set, and the controls outside of this set, which are collision-free for the ship. This characteristic is useful in the maritime environment.

In this paper, we have demonstrated two possible applications of the UO set. Firstly, it is used for ASV to find one optimal solution minimizing the changes in controls w.r.t. COLREGs. In fact, we can define a different objective function considering more factors, e.g. fuel consumption, etc. Secondly, it can be used as a navigational assistance tool. As we show, the visualized UO set supports the OOW in decision making, in particular, to trigger a collision alarm if current velocity is in the UO set, to check the feasibility of alternative maneuvers, to eliminate the dangerous maneuvers, etc.

The UO set and GVO algorithm not only can be used to control ASVs and to facilitate the OOW, but it can also support human operators to (remotely) take over the control of ASVs when necessary. For instance, when an ASV is implementing one optimal control according to the defined utility/cost function, the optimal solution might not meet the requirements of officers, e.g. violation of regulations, etc. The officers need to find an alternative solution which has to be safe. Here, UO set plays a role and human can find an alternative maneuver outside of the UO set. The visualized UO set, in fact, allows human to track the decision-making process of the ASV and understand how an optimal solution is chosen by GVO-CAS. This also can help human to manipulate an ASV with GVO-CAS.

6.4. Limitations of the proposed GVO-CAS

The GVO-CAS is based on the linearization of the ship's dynamics via Taylor expansion, see Section 3.2. As a result, there are errors between the predicted trajectory calculated by Equation (7) and the actual nonlinear trajectory of the ship. When the difference between alternative control and initial control is too big, the solution which the algorithm finds is not guaranteed to be collision-free.

To overcome this problem, four methods have been considered in this paper, i.e. successive linearization, enlarged buffer, increasing sampling rate and reducing fluctuation. Firstly, we use the successive linearization technique to reduce the errors. Usually, the linearization of a system is around its initial state at the first stage. Hence, the errors are accumulated. In this paper, we linearize the system always around the estimated trajectory at each time step. Thus, the errors are less than the errors in the usual approach by linearizing only at the first stage. Secondly, the UO set is enlarged by a positive small factor ε when the GVO-CAS finds a collision-free solution, which can avoid the collision caused by linearization errors under most conditions. Thirdly, we increase the sampling rate in the GVO-CAS to ensure that a collision-free solution can be found in time. Lastly, Rule 4 in GVO-CAS is introduced to prevent the fluctuations and dramatic changes of control inputs, e.g. from hard-port turn to starboard turn. Although these methods can work in most cases, a rigorous proof is needed in the future research.

6.5. Compliance with regulations

Rules of the road in COLREGs are considered in the GVO-CAS with Rule 3 and Rule 4 in Section 4.4.4. The experiment results show these rules can help the system to find a COLREGs-compliant solution. Moreover, though we did not intend to let the system minimize the number of control actions, the system automatically found one solution, which avoids a succession of small changes in course and/or speed.

Some rules related to certain special cases at sea are not discussed in this paper, such as narrow channels, traffic separation, etc. However, extensions in GVO-CAS can be made to deal with these situations in future research.

7. Conclusion

This paper applies a Generalized Velocity Obstacle (GVO) algorithm for ship collision prevention considering the dynamics of ships. We propose a GVO-based Collision Avoidance System (GVO-CAS), for both manned and unmanned ship in a dynamic maritime environment. Three main contributions of this paper are: (1) GVO algorithm considering ship dynamics is introduced to handle multiple-ship scenario in close range at sea; (2) A novel collision avoidance system based on GVO algorithm is proposed which visualizes collision-free solutions for the Officer On Watch (OOW) and supports both manned and unmanned for collision prevention; (3) Navigation regulations are integrated in the GVO-CAS to make it more suitable for maritime practice.

Case studies of two-ship encounter scenarios and multiple-ship encounter scenarios are presented to show the performance of the GVO algorithm and the GVO-CAS. The two-ship encounter scenarios show that the GVO algorithm is more reliable than the Velocity Obstacle (VO) algorithm: it is capable to find rule-compliant evasive actions for the ship; and fewer actions are needed to avoid collisions. In the multiple-ship encounter case, GVO-CAS is capable to support multiple ship collision avoidance problems. These scenarios show that the proposed GVO algorithm has great potential for collision prevention at sea.

In the future, some research directions are considered to improve the GVO algorithm and GVO-CAS. Firstly, the influence of model errors and environmental disturbances on the performance of the GVO algorithm needs further research. Secondly, the errors due to the linearization of nonlinear dynamics need be studied. Thirdly, the shape of the ship, more constraints on dynamics (e.g. maximal control forces) and cooperation between ships also need further research.

List of Abbreviations

ARPA	Automatic Radar Plotting Aid	OS	Own Ship
ASV	Autonomous Surface Vehicle	PD	Proportional-derivative
CAS	Collision Avoidance System	PID	Proportional-integral-derivative
CCA	Cooperative collision avoidance	sUO	Sub-UO set
COLREGs	Convention on the International Regulations for Preventing Collisions at Sea	sVO	Sub-VO set
		TS	Target Ship
ConfP	Conflict Positions	VO	Velocity obstacle
CPA	Closest point of approach	EBVO	Ellipse Based VO
DCPA	Distance to CPA	GVO	Generalized VO
TCPA	Time to CPA	LVO	Linear VO
CTPA	Collision threat parameter area	NLVO	Nonlinear VO
DOF	Degree of freedom	PVO	Probabilistic VO
OOW	Officer on watch	RVO	Reciprocal VO

Acknowledgment

This work is supported by the China Scholarship Council under Grant: 201406950010 and 201406950041.

Reference

Alonso-Mora, J., Beardsley, P., Siegwar, R., 2018. Cooperative Collision Avoidance for Nonholonomic Robots. Ieee Transactions on Robotics 34 (2), 404-420.

Alonso-Mora, J., Breitenmoser, A., Beardsley, P., Siegwart, R., 2012. Reciprocal Collision Avoidance for Multiple Car-like Robots. 2012 Ieee International Conference on Robotics and Automation (ICRA), 360-366.

Baldauf, M., Mehdi, R., Fischer, S., Gluch, M., 2017. A perfect warning to avoid collisions at sea? Scientific Journals of the Maritime University of Szczecin 49 (121), 53-64.

Bareiss, D., van den Berg, J., 2013. Reciprocal Collision Avoidance for Robots with Linear Dynamics using LQR-Obstacles. 2013 Ieee International Conference on Robotics and Automation (Icra), 3847-3853.

Bareiss, D., van den Berg, J., 2015. Generalized reciprocal collision avoidance. The International Journal of Robotics Research 34 (12), 1501-1514.

Benjamin, M.R., Leonard, J.J., Curcio, J.A., Newman, P.M., 2006. A method for protocol-based collision avoidance between autonomous marine surface craft. Journal of Field Robotics 23 (5), 333-346.

Best, A., Narang, S., Barber, D., Manocha, D., 2017. AutonoVi: Autonomous Vehicle Planning with Dynamic Maneuvers and Traffic Constraints, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, Vancouver, BC, Canada.

Campbell, S., Naeem, W., Irwin, G.W., 2012. A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres. Annual Reviews in Control 36 (2), 267-283.

Chen, L., Hopman, H., Negenborn, R.R., 2018a. Distributed model predictive control for vessel train formations of cooperative multi-vessel systems. Transportation Research Part C: Emerging Technologies 92, 101-118.

Chen, P., Huang, Y., Mou, J., van Gelder, P.H.A.J.M., 2018b. Ship collision candidate detection method: A velocity obstacle approach. Ocean Engineering 170, 186-198.

Coue, C., Pradalier, C., Laugier, C., Fraichard, T., Bessiere, P., 2006. Bayesian Occupancy Filtering for Multitarget Tracking: An Automotive Application. The International Journal of Robotics Research 25 (1), 19-30.

Daily, R., Bevly, D.M., 2008. Harmonic potential field path planning for high speed vehicles, 2008 American Control Conference. IEEE, Westin Seattle Hotel, Seattle, Washington,, pp. 4609-4614.

Degre, T., Lefevre, X., 1981. A Collision Avoidance System. Journal of Navigation 34 (2), 294-302.

Fiorini, P., Shiller, Z., 1998. Motion Planning in Dynamic Environments using Velocity Obstacle. The International Journal of Robotics Research 17 (7), 760-772.

Fossen, T.I., 2002. Marine Control Systems: Guidance, Navigation, and Control of Ships, Rigs and Underwater Vehicles. Marine Cybernetics, Trondheim, Norway.

Goerlandt, F., Montewka, J., Kuzmin, V., Kujala, P., 2015. A risk-informed ship collision alert system: Framework and application. Safety Science 77, 182-204.

He, Y., Jin, Y., Huang, L., Xiong, Y., Chen, P., Mou, J., 2017. Quantitative analysis of COLREG rules and seamanship for autonomous collision avoidance at open sea. Ocean Engineering 140, 281-291.

Hilgert, H., Baldauf, M., 1997. A Common Risk Model for the Assessment of Encounter Situations on Board Ships German Journal of Hydrography 49 (4), 531-542.

Huang, Y., van Gelder, P.H.A.J.M., Wen, Y., 2018. Velocity Obstacle Algorithms for Collision Prevention at Sea. Ocean Engineering 151, 308-321.

IMO, 1972. Convention on the International Regulations for Preventing Collisions at Sea, 1972 (COLREGs), in: Organization, I.M. (Ed.).

Johansen, T.A., Perez, T., Cristofaro, A., 2016. Ship Collision Avoidance and COLREGS Compliance Using Simulation-Based Control Behavior SelectionWith Predictive Hazard Assessment. IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATIONSYSTEMS 17 (12), 3407-3422.

Kearon, J., 1979. Computer programs for collision avoidance and track keeping, in: Hollingdale, S.H. (Ed.), Mathematical Aspects of Marine Traffic. Academic Press INC. LTD., London, UK.

Kuwata, Y., Wolf, M.T., Zarzhitsky, D., Huntsberger, T.L., 2014. Safe maritime autonomous navigation with COLREGS, using velocity obstacles. IEEE Journal of Oceanic Engineering 39 (1), 110-119.

Large, F., Laugier, C., Shiller, Z., 2005. Navigation among moving obstacles using the NLVO: Principles and applications to intelligent vehicles. Autonomous Robots 19 (2), 159-171.

Lee, B.H., Jeon, J.D., Oh, J.H., 2016. Velocity obstacle based local collision avoidance for a holonomic elliptic robot. Autonomous Robots 41 (6), 1347-1363.

Lenart, A.S., 1983. Collision Threat Parameters for a new Radar Display and Plot Technique. Journal of Navigation 36 (03), 404-410.

Liu, Z., Zhang, Y., Yu, X., Yuan, C., 2016. Unmanned surface vehicles: An overview of developments and challenges. Annual Reviews in Control 41, 71-93.

Lyu, H.G., Yin, Y., 2017. Ship's Trajectory Planning for Collision Avoidance at Sea Based on Modified Artificial Potential Field. 2017 2nd International Conference on Robotics and Automation Engineering (Icrae), 351-357.

Montewka, J., Ehlers, S., Goerlandt, F., Hinz, T., Tabri, K., Kujala, P., 2014. A framework for risk assessment for maritime transportation systems—A case study for open sea collisions involving RoPax vessels. Reliability Engineering & System Safety 124, 142-157.

Pedersen, E., Inoue, K., Tsugane, M., 2003. Simulator Studies on a Collision Avoidance Display that Facilitates Efficient and Precise Assessment of Evasive Manoeuvres in Congested Waterways. The Journal of Navigation 56 (3), 411-427.

Skjetne, R., Smogeli, Ø.N., Fossen, T.I., 2004. A Nonlinear Ship Manoeuvering Model: Identification and adaptive control with experiments for a model ship. Modeling, Identification and Control: A Norwegian Research Bulletin 25 (1), 3-27.

Szlapczynski, R., 2008. Planning Emergency Manoeuvres. Journal of Navigation 62 (01), 79.

Szlapczynski, R., Krata, P., Szlapczynska, J., 2018. Ship domain applied to determining distances for collision avoidance manoeuvres in give-way situations. Ocean Engineering 165, 43-54.

Szlapczynski, R., Szlapczynska, J., 2015. A Target Information Display for Visualising Collision Avoidance Manoeuvres in Various Visibility Conditions. Journal of Navigation 68 (06), 1041-1055.

Szlapczynski, R., Szlapczynska, J., 2016. An analysis of domain-based ship collision risk parameters. Ocean Engineering 126, 47-56.

Szlapczynski, R., Szlapczynska, J., 2017. A method of determining and visualizing safe motion parameters of a ship navigating in restricted waters. Ocean Engineering 129, 363-373.

Tam, C., Bucknall, R., Greig, A., 2009. Review of Collision Avoidance and Path Planning Methods for Ships in Close Range Encounters. Journal of Navigation 62 (03), 455-476.

van den Berg, J., Guy, S.J., Lin, M., Manocha, D., 2011. Reciprocal n-Body Collision Avoidance. Robotics Research 70, 3-19.

Van Gelder, P.H.A.J.M., 1991. H-infinity-control design on a spraydryer process. Department of Mathematics and Computer Science, Department of Mathematics and Computer Science. TU Eindhoven, Eindhoven, NL.

Velasco, G.A.M., Borst, C., Ellerbroek, J., van Paassen, M.M., Mulder, M., 2015. The Use of Intent Information in Conflict Detection and Resolution Models Based on Dynamic Velocity Obstacles. Ieee Transactions on Intelligent Transportation Systems 16 (4), 2297-2302.

Wang, X., Liu, Z., Cai, Y., 2017. The ship maneuverability based collision avoidance dynamic support system in close-quarters situation. Ocean Engineering 146, 486-497.

Wilkie, D., Berg, J.v.d., Manocha, D., 2009. Generalized velocity obstacles, 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5573-5578.

Zhang, J., Yan, X., Chen, X., Sang, L., Zhang, D., 2012. A novel approach for assistance with anti-collision decision making based on the International Regulations for Preventing Collisions at Sea. Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment 226 (3), 250-259.

Zhang, J., Zhang, D., Yan, X., Haugen, S., Guedes Soares, C., 2015. A distributed anti-collision decision support formulation in multi-ship encounter situations under COLREGs. Ocean Engineering 105, 336-348.

Zhang, W., Wei, S., Teng, Y., Zhang, J., Wang, X., Yan, Z., 2017. Dynamic Obstacle Avoidance for Unmanned Underwater Vehicles Based on an Improved Velocity Obstacle Method. Sensors (Basel) 17 (12).

Zhao, Y., Li, W., Shi, P., 2016. A real-time collision avoidance learning system for Unmanned Surface Vessels. Neurocomputing 182, 255-266.

Zuhaib, K., Khan, A., Iqbal, J., Ali, M., Usman, M., Ali, A., Yaqub, S., Lee, J., Han, C., 2017. Collision Avoidance from Multiple Passive Agents with Partially Predictable Behavior. Applied Sciences 7 (9), 903.

Appendix A: Parameters of CyberShip II

CyberShip II is a scaled ship model with m = 23.8 kg and L_m = 1.255 m. The settings of these parameters are presented in Table A.

Table A The parameters of CyberShip II (Skjetne et al., 2004)

m	*23.800*	Y_v	*-0.88965*	N_v	*0.03130*
I_z	*1.760*	$Y_{\dot{v}}$	*-10.0*	N_r	*-1.900*
x_g	*0.046*	Y_r	*-7.250*	$N_{\dot{v}}$	*-0.0*
X_u	*-0.72253*	$Y_{\dot{r}}$	*-0.0*	$N_{\dot{r}}$	*-1.0*
$X_{\dot{u}}$	*-2.0*	$Y_{\|v\|r}$	*-0.845*	$N_{\|v\|r}$	*0.08*
$X_{\|u\|u}$	*-1.32742*	$Y_{\|v\|v}$	*-36.47287*	$N_{\|r\|r}$	*-0.750*
X_{uuu}	*-5.86643*	$Y_{\|r\|v}$	*-0.805*	$N_{\|r\|v}$	*0.130*
		$Y_{\|r\|r}$	*-3.450*	$N_{\|v\|v}$	*3.95645*

We use Froude scaling law in the simulation to link the scaled model and the real physical world, see Table 1. The gravity keeps the same in scale model and the real world. The scale of length is $\alpha = 1/70$. That means 1 meter in scaled model equals to 70 meters in the real world. The scale of velocity and time are $\sqrt{\alpha}$. Thus, the speed of the scale ship is 0.5 [m/s] is approximately equal to the full-scale ship speed 4.2 [m/s] (roughly 8 [knot]). In the following text, the units are all in scale system, if not specifically specified.

Table 1. The scale relations between model and real world ($\alpha = 1/70$)

	Scale Model	Real world	Relation
Gravity	g_m [m/s^2]	g_s [m/s^2]	$g_m = g_s$
Length	L_m [m]	L_s [m]	$L_m = \alpha L_s$
Speed	V_m [m/s]	V_s [m/s]	$V_m = \sqrt{\alpha} V_s$
Time	T_m [s]	T_s [s]	$T_m = \sqrt{\alpha} T_s$

Appendix B: COLREGs regulations

A brief overview of the main operational requirements of COLREGs(IMO, 1972) related to our research is listed in this section:

Rule 8: Action to avoid collision. (a) Any action taken to avoid collision shall be made in ample time. (b) Any alteration of course and/or speed shall be large enough and a succession of small alterations of course and/or speed should be avoided. (c) If there is sufficient sea room, alteration of course alone may be the most effective. (d) Action taken to avoid collision with another vessel shall be such as to result in passing at a safe distance. (e) If it is necessary to avoid collision, a vessel shall slacken her speed or take all way off by stopping or reversing her means of propulsion.

Rule 13: Overtaking. (a) A vessel shall be deemed to be overtaking when coming up with another vessel from a direction more than 22.5 degrees abaft her beam. (b) Any subsequent alteration of the bearing between the two vessels shall not make the overtaking vessel relieve her of the duty of keeping clear of the overtaken vessel.

Rule 14: Head-on situation. When two power-driven vessels are meeting on 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.

Rule 15: Crossing situation. When two power-driven vessels are crossing, 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.

Rule 16: Action by give-way vessel. Every vessel keeping out of the way of another vessel shall, so far as possible, take early and substantial action to keep well clear.

Rule 17: Action by stand-on vessel. The stand-on vessel shall keep her course and speed, except two cases: (i) the vessel required to keep out of the way is not taking appropriate action in compliance with these Rules; (ii) the vessels are so close that collision cannot be avoided by the action of the give-way vessel alone. The vessel which takes action in a crossing situation in accordance with case (i) of this Rule to avoid collision with another vessel shall not alter course to the portside of a vessel on her own portside.

Appendix C: Details of the simulation results

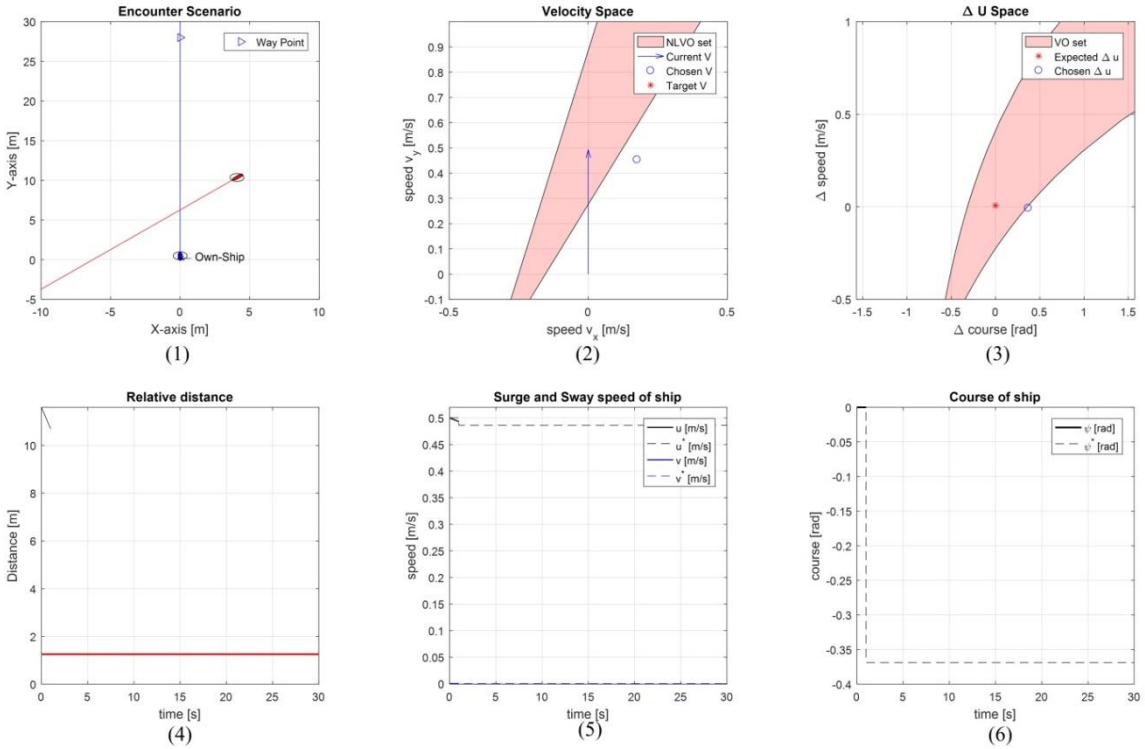

Fig. C1. The state of the OS at time 1[s]

(note: (1) is encounter situation; (2) is VO set in this situation; (3) is the UO set; (4) relative distance; (5) speed of the OS; (6) course of the OS)

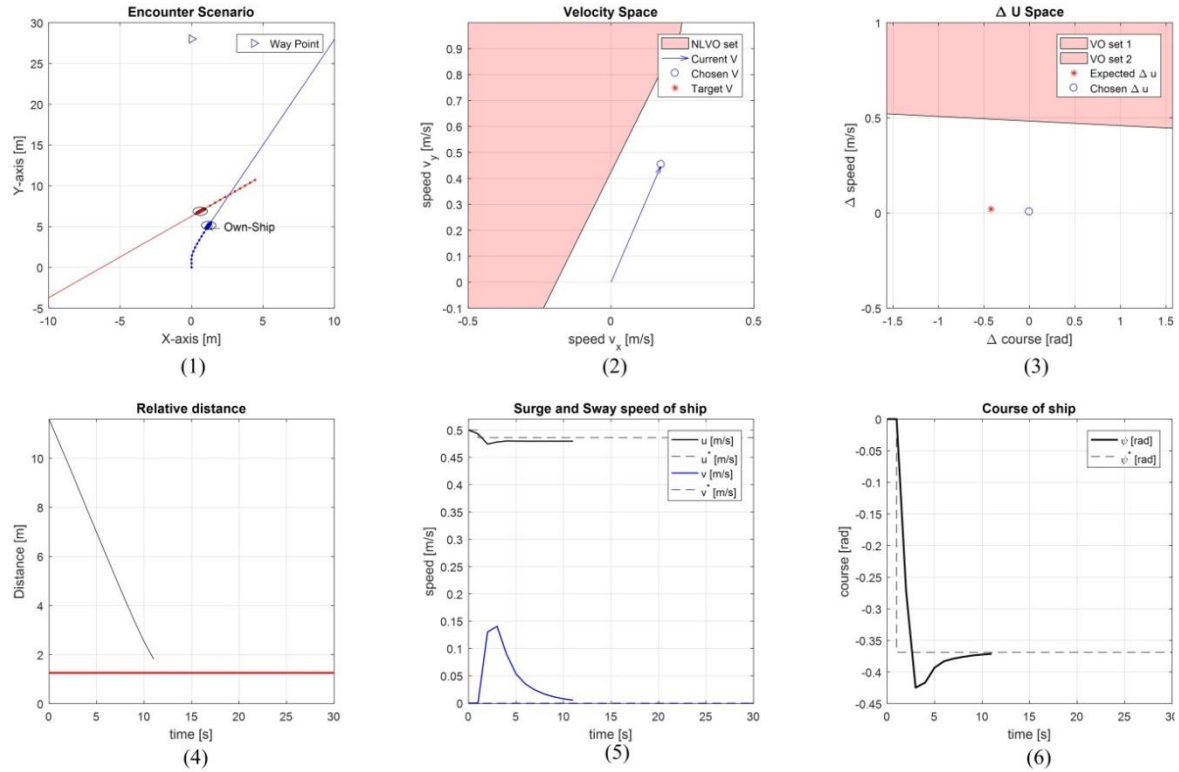

Fig. C2. The state of the OS at time 11[s]

(note: (1) is encounter situation; (2) is VO set in this situation; (3) is the UO set; (4) relative distance; (5) speed of the OS; (6) course of the OS)

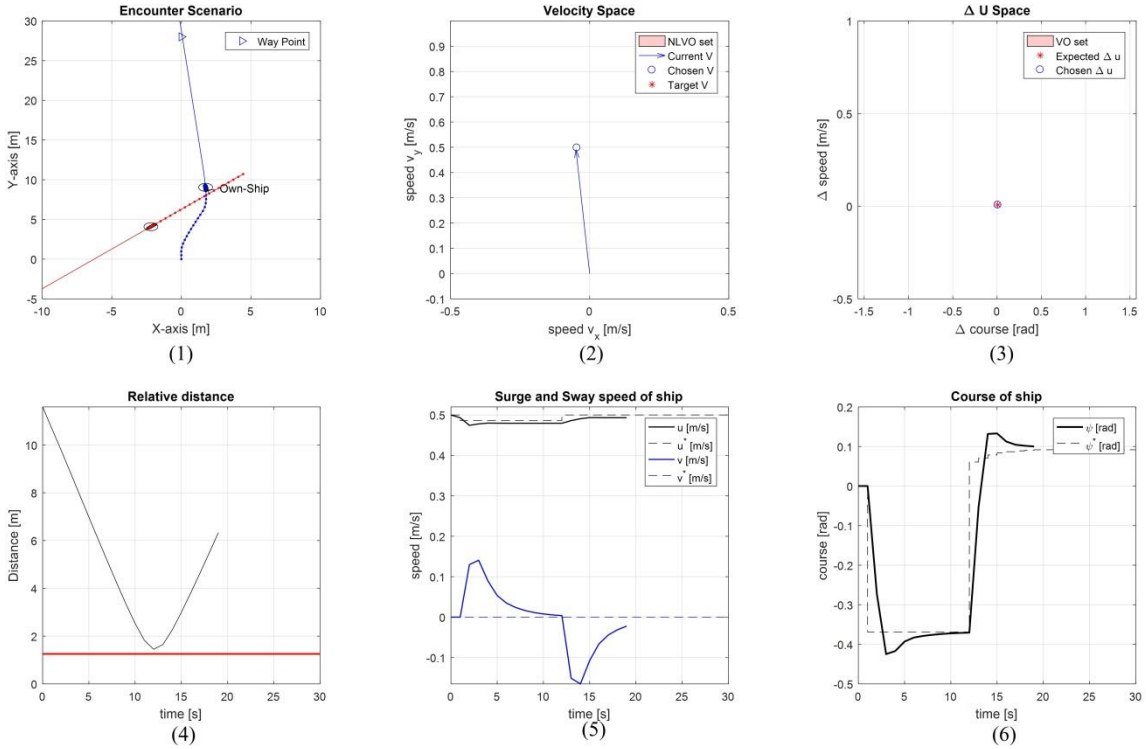

Fig. C3. The state of the OS at time 19 [s]

(note: (1) is encounter situation; (2) is VO set in this situation; (3) is the UO set; (4) relative distance; (5) speed of the OS; (6) course of the OS)