Journal of Marine Science and Technology

Volume 29 | Issue 4 Article 3

Differential Evolution Algorithm for Ship Path Planning in Open Waters

Yu-Tao Kang
College of Ocean Science and Engineering, Shanghai Maritime University, Shanghai, China, ytkang@shmtu.edu.cn

Wei-Jiong Chen
College of Ocean Science and Engineering, Shanghai Maritime University, Shanghai, China

Jin-Hui Wang
College of Ocean Science and Engineering, Shanghai Maritime University, Shanghai, China

Recommended Citation
Kang, Yu-Tao; Chen, Wei-Jiong; and Wang, Jin-Hui (2021) "Differential Evolution Algorithm for Ship Path Planning in Open Waters," *Journal of Marine Science and Technology*: Vol. 29: Iss. 4, Article 3.
DOI: 10.51400/2709-6998.1582
Available at: https://jmstt.ntou.edu.tw/journal/vol29/iss4/3

RESEARCH ARTICLE

Differential Evolution Algorithm for Ship Path Planning in Open Waters

Yu-Tao Kang*, Wei-Jiong Chen, Jin-Hui Wang

College of Ocean Science and Engineering, Shanghai Maritime University, No.1550, Hai-Gang Rd., Pudong New Area, Shanghai, China

Abstract

A basic problem in the design of a decision support system for ship collision avoidance is path optimization in a complex and dynamic navigational environment. This paper introduces a new path planning method based on the differential evolution (DE) algorithm to calculate a safe, optimal path for a ship. The algorithm was tested on a set of traffic scenarios typically encountered in open waters. The simulation test results prove the method's ability to solve a path planning problem for ships. We also discuss the optimality, consistency, and performance of the algorithm and provide a comparison of this algorithm with the particle swarm optimization (PSO) algorithm. The comparison results clearly show a significant advantage of the DE algorithm over the PSO algorithm in the areas of output optimality, algorithm consistency, and execution efficiency.

Keywords: Collision avoidance, Path planning, Differential evolution algorithm

1. Introduction

Ship collision avoidance remains a key consideration in maritime navigation. Ships must contend with path planning problems still present in some applications, posing problems for decision support systems and unmanned vehicles. Path planning is a growing research area in ship collision avoidance. However, ship path planning for collision avoidance is a multiobjective nonlinear optimization problem in a complex and dynamic marine environment, and it must balance navigational safety and economic performance under some constraints, such as human factors, ship maneuverability, environmental conditions, and COLREGS (the Convention on International Regulations for Preventing Collisions at Sea) compliance. To solve this problem, experts and scholars have proposed and developed many methods over the past few decades. These methods can be approximately divided into deterministic and the heuristic approaches [13].

The deterministic approach determines solutions by following a set of rigorously defined steps; these approaches include fuzzy set theory [3,4], the maze-routing algorithm [1,10], the dynamic programming method [8], and the trajectory base algorithm [7]. The deterministic approaches determine collision avoidance maneuvers or calculate optimal paths by treating each target ship (TS) in stages, resulting in a suboptimal final output. Furthermore, most studies have focused solely on determining the shortest collision-free path within the solution space without considering COLREGS compliance and environmental conditions.

The heuristic approach does not involve searching for the optimal solution in the search space; instead, it involves developing an acceptable solution that satisfies design requirements. Hence, the execution efficiency of a heuristic algorithm is generally much higher than that of a deterministic algorithm. More representative studies use the evolutionary algorithm [11,12,14] or the genetic algorithm [2,16,17] to compute the collision avoidance navigation path.

Received 21 August 2019; revised 16 October 2020; accepted 18 January 2021.
Available online 3 September 2021.

* Corresponding author.
E-mail address: ytkang@shmtu.edu.cn (Y.-T. Kang).

https://doi.org/10.514/2709-6998.1582

Other researchers have adopted the ant colony algorithm [6,15] to plan a safe and economical collision avoidance path in dynamic environments. Recently, the current authors presented a modified particle swarm optimization (PSO) algorithm [5] to solve the path planning problem in real-time navigation environments. In this 2018 study, an improved dynamic ship domain model was used to assess collision risks in close-range encounters. However, some limitations must be remedied in these algorithms, such as coding, parameter setting, high dimension problem, and premature convergence problems. In the majority of the reviewed studies, researchers have determined a navigation path without explicit consideration of the path compatibility with other ships as well as the algorithm's consistency and completeness.

This paper presents a new path planning approach based on the differential evolution (DE) algorithm as an attempt to produce a collision-free and optimal path in open waters. In this new approach, ship domain is regarded as an assessment criterion for collision risk. In this paper, we discuss the optimality of the algorithm output and the consistency and efficiency of the algorithm; we also compare the performance of the algorithm with that of the PSO algorithm. The rest of this paper is organized as follows. Section II describes a new path planning method based on the DE algorithm. Section III presents the results of simulation studies for verifying the proposed algorithm. Section IV provides a discussion of the performance of the proposed algorithm and a comparison with the PSO algorithm. Section V conclude the paper.

2. Path planning method

To simplify the path planning problem, several assumptions are made as follows:

1) All ships can be regarded as moving points in open waters because the distance between ships is much greater than the size of ships. The ship under direct control is denoted as the OS, whereas any ship other than the OS is denoted as a TS.
2) All ship operators can obtain real-time collision avoidance information about their surroundings. Static and dynamic information about ships in real-life navigation can be provided by numerous navigational aids (e.g., Automatic Identification System and Automatic Radar Plotting Aid).
3) A two-dimensional space was adopted in this study, where the real-time data for the OS and TS were defined.

Generally, path planning for a ship at a certain time in navigation involves searching for an optimal path from the initial position to the target position while simultaneously avoiding other ships and their ship domains.

2.1. Environmental map

The domain of interest in this paper is limited to the 6 nautical mile (nm) radius of the OS. Construction of the environment map for path planning is shown in Fig. 1. The environment map is built with the initial position of the OS as the origin and the bow orientation of the OS as the Y-axis of the coordinate system, where the area within the red solid line is the domain of the TS, S is the initial point of the OS, and F is the target point.

The navigation path can be discretized into several linear segments divided equally from the start to the destination. A safety domain defined by the ship domain model around the TS must be provided at each segment. It is assumed that the velocity vector of the TS is constant due to the instantaneous navigation path in this study. Path planning involves searching for a set of waypoints in the environmental map to obtain the shortest path, and this enables the adjacent points and their connecting lines to avoid the TS and its safety domain. Because of the equidistant selection of the longitudinal axis, the vector X composed of the abscissa

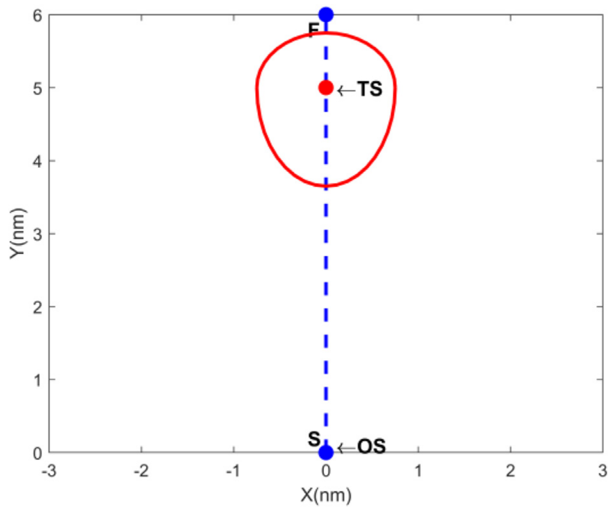

Fig. 1. Environment map.

coordinates of the path points can determine a unique path, which is defined as follows:

$$\left\{x \middle| x_j^L \le x_j \le x_j^U, j = 1, 2, \cdots, D\right\} \tag{1}$$

where x_j is the component of X, and D is the dimension of the solution space. x_j^L and x_j^U are the upper and lower bounds of the value range of x_j, respectively. Ship path planning is transformed into a process of searching for a set of path points in the environmental map. This helps the adjacent path points and their links avoid other ships and their ship domains, and it makes the total path length the shortest possible.

2.2. DE algorithm

The DE algorithm is a stochastic optimization algorithm based on the genetic evolution of the population. Similar to other evolutionary algorithms, it includes mutation, crossover, and selection. The algorithm mutates individuals using differential information among individuals in a population. It then performs crossover operations based on a probability mechanism and finally updates the population through a greedy selection mechanism. At the beginning of the evolution, due to the larger disturbing quantity from larger individual variations in the population, the algorithm has a stronger exploration capability than at other time points, allowing it to search in a larger range. However, when the algorithm tends to converge in the later stages, it searches near the individual with a stronger local search ability because of the smaller individual variation of the population. Thus, the DE algorithm has superior performance to other evolutionary algorithms as a result of its ability to learn from individual populations [9].

The population of the DE algorithm is driven by mutation and selection processes. Mutation processes, including mutation and crossover operations, are designed to exploit or explore the search space, and selection processes are used to ensure that information about promising individuals can be further used. As shown in Fig. 2, the algorithm is composed of the following steps:

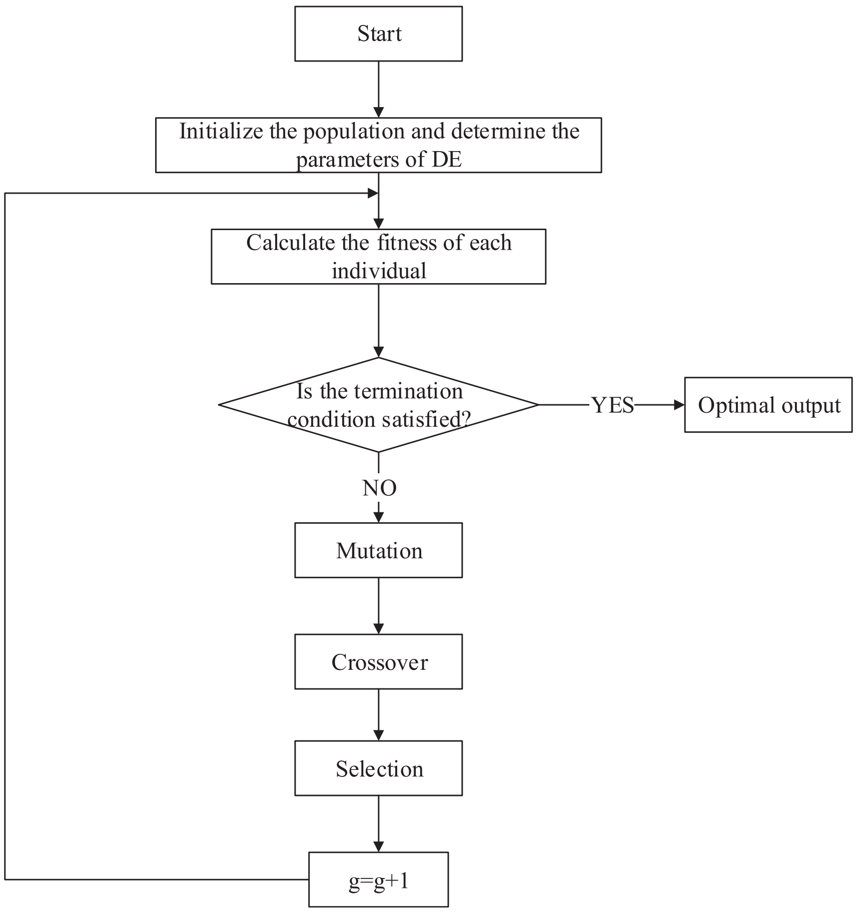
Start
Initialize the population and determine the parameters of DE
Calculate the fitness of each individual
Is the termination condition satisfied?
YES
Optimal output
NO
Mutation
Crossover
Selection
g=g+1

Fig. 2. DE algorithm calculation process.

1) Initialization: The DE algorithm contains a population of many individuals, each of which can be regarded as a solution in the search space. Assuming that the dimension of the current population is NP, the initial population is generated randomly.

$$\left\{x_i(0)\middle|x_{j,i}^L \le x_{j,i}(0) \le x_{j,i}^U, i=1,2,\cdots,NP; j=1,2,\cdots,D\right\} \tag{2}$$

$$x_{j,i}(0)=x_{j,i}^L+rand(0,1)\cdot\left(x_{j,i}^U-x_{j,i}^L\right) \tag{3}$$

where $x_i(0)$ is the chromosome of the 0th generation of the population, $x_{j,i}(0)$ is the gene of the chromosome, and $x_{j,i}^L$ and $x_{j,i}^U$ are the upper and lower bounds of the value range of x_j, respectively.

2) Mutation: In biology, this refers to change in genes at a certain location on a chromosome. In evolutionary computation, it refers to changes in values at a certain location through random destabilization. The DE algorithm achieves individual mutation through a differential strategy, an important difference between this algorithm and genetic algorithms. In the DE algorithm, the usual differential strategy is to randomly select two different individuals in a population, whose vector differences are scaled, and synthesize the vectors with the individuals to be mutated. The mutation process is defined as follows:

$$v_i(g+1)=x_{r1}(g)+F\cdot(x_{r2}(g)-x_{r3}(g)) \tag{4}$$

$$i \neq r_1 \neq r_2 \neq r_3 \tag{5}$$

where $x_i(g)$ is the individual of the population and F is the scaling factor. To ensure the validity of the solution, it is necessary to judge whether the genes in the chromosome satisfy the boundary conditions in the evolutionary process. If the boundary conditions are not satisfied, the "genes" are regenerated randomly (the same as the generation method of the initial population). The g-generation population $x_i(g)$ produces an intermediate $v_i(g+1)$ after mutation.

3) Crossover: The crossover between individuals of $x_i(g)$ and $v_i(g+1)$ is defined as follows:

$$u_{j,i}(g+1)=\begin{cases} v_{j,i}(g+1), & rand(0,1) \le CRorj=j_{rand} \\ x_{j,i}(g), & rand(0,1)>CRandj\neq j_{rand} \end{cases} \tag{6}$$

where CR is the crossover probability and j_{rand} is the random integer in $[1,2,\ \cdots,\ D]$. Introducing the crossover operation provides more potential solutions for the algorithm, which promotes information sharing within the population and improves the diversity of the population.

4) Selection: To maintain the current population dimension during population iteration, the DE algorithm uses a greedy algorithm to select the g-generation population $x_i(g)$ and the intermediate population $u_i(g+1)$ generated by mutation and crossover operations, and individuals that exhibit better fitness are selected for the next generation. The greedy algorithm is defined as follows:

$$x_i(g+1)=\begin{cases} u_i(g+1), & f(u_i(g+1)) \le f(x_i(g)) \\ x_i(g), & f(u_i(g+1))>f(x_i(g)) \end{cases} \tag{7}$$

where $f(u_i(g+1))$ and $f(x_i(g))$ are the fitness of $x_i(g)$ and $u_i(g+1)$, respectively.

2.3. Fitness function

Because the TS is always moving, it is necessary to determine the possibility of collision at every segment of the path. It is assumed, for calculation simplicity, that no collision risk arises in this segment as long as the waypoint is not in the ship domain. Thus, the calculation process for the fitness of each "chromosome" is shown in Fig. 3. To shorten the length of the path and avoid any potential collisions with other ships, the fitness function of each "chromosome" is defined as follows:

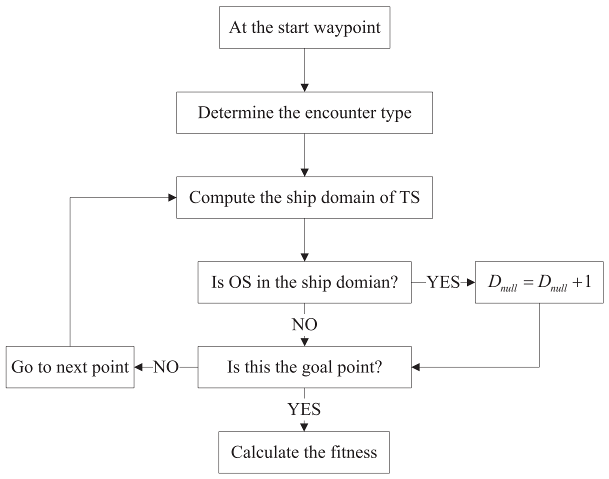

Fig. 3. Calculation process for the fitness of each "chromosome".

Table 1. Traffic scenarios.

Scenario	OS		TS		Encounter type
	Initial position(nm)	velocity (nm/min)	Initial position(nm)	velocity (nm/min)	
1	[0,0]	[0,0.5]	[0,6]	[0,-0.5]	Head-on
2	[0.25,0]	[0,0.5]	[-0.25,6]	[0,-0.5]	Port to Port
3	[-0.25,0]	[0,0.5]	[0.25,6]	[0,-0.5]	Starboard to Starboard
4	[0,0]	[0,0.6]	[0,2]	[0,0.2]	Overtaking
5	[0,0]	[0,0.5]	[3,3]	[-0.5,0]	Crossing
6	[0,0]	[0,0.5]	[3,4]	[-0.5,-0.25]	Small-angle Crossing
7	[0,0]	[0,0.5]	[3,1]	[-0.75,0.25]	Large-angle Crossing
8	[0,0]	[0,0.5]	[0,3]	[0,0]	Static

$$l = \sqrt{(x_1 - x_S)^2 + \left(\frac{y_F - y_S}{D+1}\right)^2} + \sum_{i=1}^{D-1}\sqrt{(x_{i+1} - x_i)^2 + \left(\frac{y_F - y_S}{D+1}\right)^2} + \sqrt{(x_F - x_D)^2 + \left(\frac{y_F - y_S}{D+1}\right)^2} \tag{8}$$

$$f = l + G_D \cdot D_{null} \tag{9}$$

where l is the length of the path, and (x_S, y_S) and (x_F, y_F) are the coordinates of the start point and the goal point, respectively. G_D is the range of the environment map, which was set to 6 nm in this study. D_{null} is the number of invalid path segments (i.e., where the OS collides with or is in the safety domain of the TS). f is the fitness of the "chromosome." The smaller the fitness value is (i.e., a shorter path length and fewer intersections with other ships or their ship domains), the better the solution is.

3. Simulation

The path planning algorithm was tested using several traffic scenarios based on different ship encounters. These scenarios were established to examine the practicality of the algorithm output as well as evaluate the algorithm performance. In addition, the algorithm was simulated in MATLAB and run on an Intel core i7 processor (3.40 GHz [8 cores] with 8 GB of RAM) and Windows 7 operating system; the parameters were set as follows:

$$NP = 20, \quad D = 19, \quad F = 0.5, \quad CR = 0.3, \quad G = 5000$$

3.1. Traffic scenarios

A series of traffic scenarios were constructed for almost all encounters between two ships in the real-time marine traffic environment, as shown in Table 1. The position and velocity of the TS were measured relative to the initial position of the OS, which was set to be at point (0,0) in the coordinate system. To verify the applicability of collision avoidance, some traffic scenarios (i.e., 1, 4–8) were set up with convergent bearing such that the OS would collide with the TS if the OS did not change its course. Different from other scenarios (i.e., 2–3) with nonconvergent bearing, a collision risk would occur between an OS and TS if their courses are maintained.

Scenarios 1–3 were based on the head-on encounters, where two ships meet on reciprocal or nearly reciprocal courses so as to involve a risk of collision. The aim of using scenario 4 was to explain overtaking encounters between the OS and TS on the same course but at different speeds. Scenarios 5–7 were set up to test crossing encounters, including square crossing, small-angle crossing, and large-angle crossing. Finally, for scenario 8, a stationary TS was introduced to examine the OS's avoidance of a static obstacle. Figure 4 illustrates the initial states of the OS and TS for all traffic scenarios.

3.2. Simulation results

The previously mentioned path planning algorithm was used to simulate all traffic scenarios, and the results are illustrated in Fig. 5. To verify the compatibility of the output, the simulation results present the optimal paths of the OS and the path selection of the TS simultaneously.

In scenarios 1 and 2, each ship altered its course to starboard so that each would pass on the port side of the other, complying with COLREGS rule 14. However, each altered its course to port to obtain the shortest path in scenario 3. The path of each TS was obtained by rotating the path of the OS under the TS scenario because the OS and TS had reverse roles under the same traffic scenario. In scenario 4, the OS stayed out of the path of the TS according to COLREGS rule 13, whereas the TS always maintained its course to avoid confusion with the overtaking party. In scenarios 5–7, the OS also stayed out of the path of the TS and avoided passing ahead

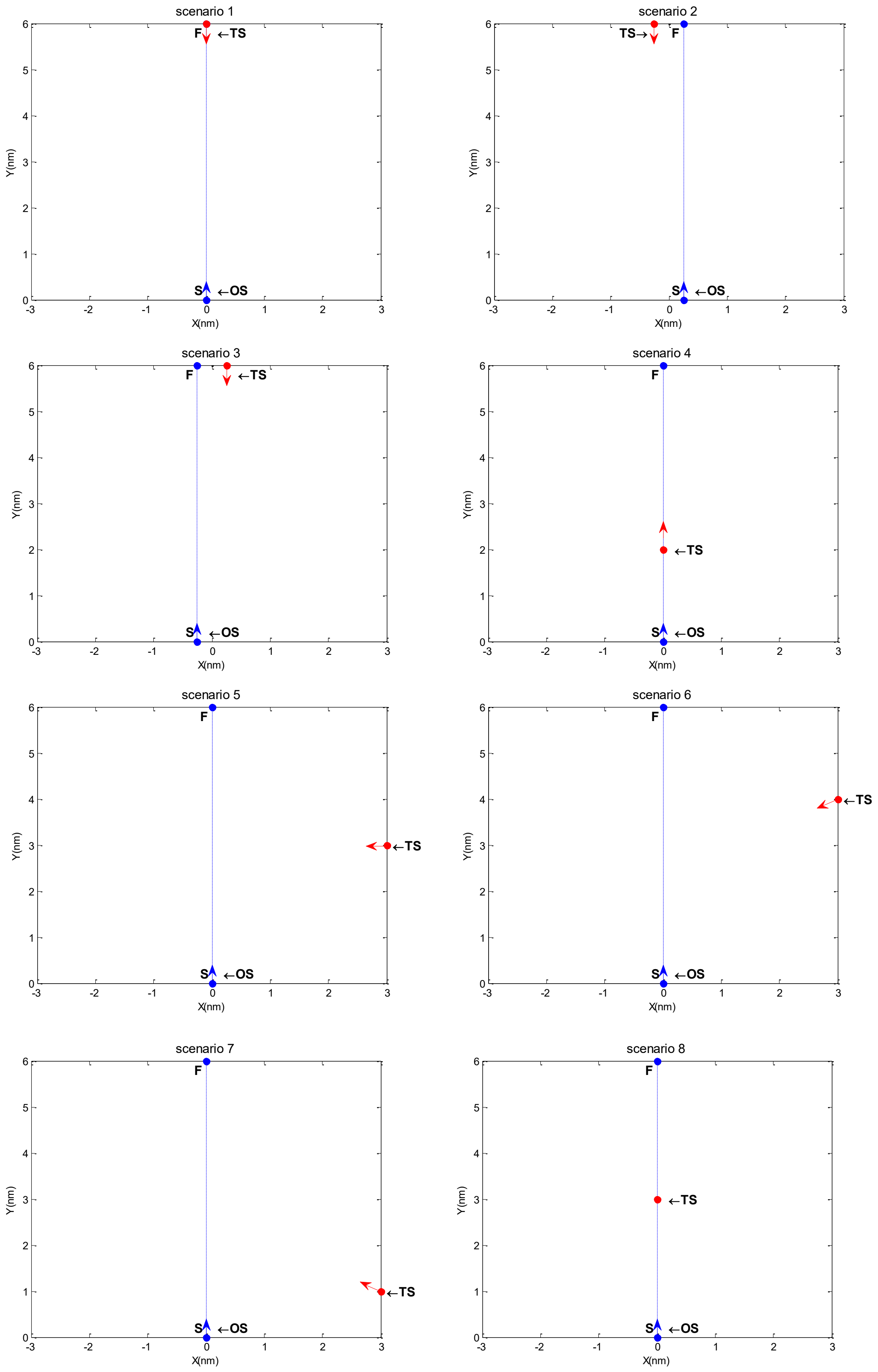

Fig. 4. Traffic scenarios.

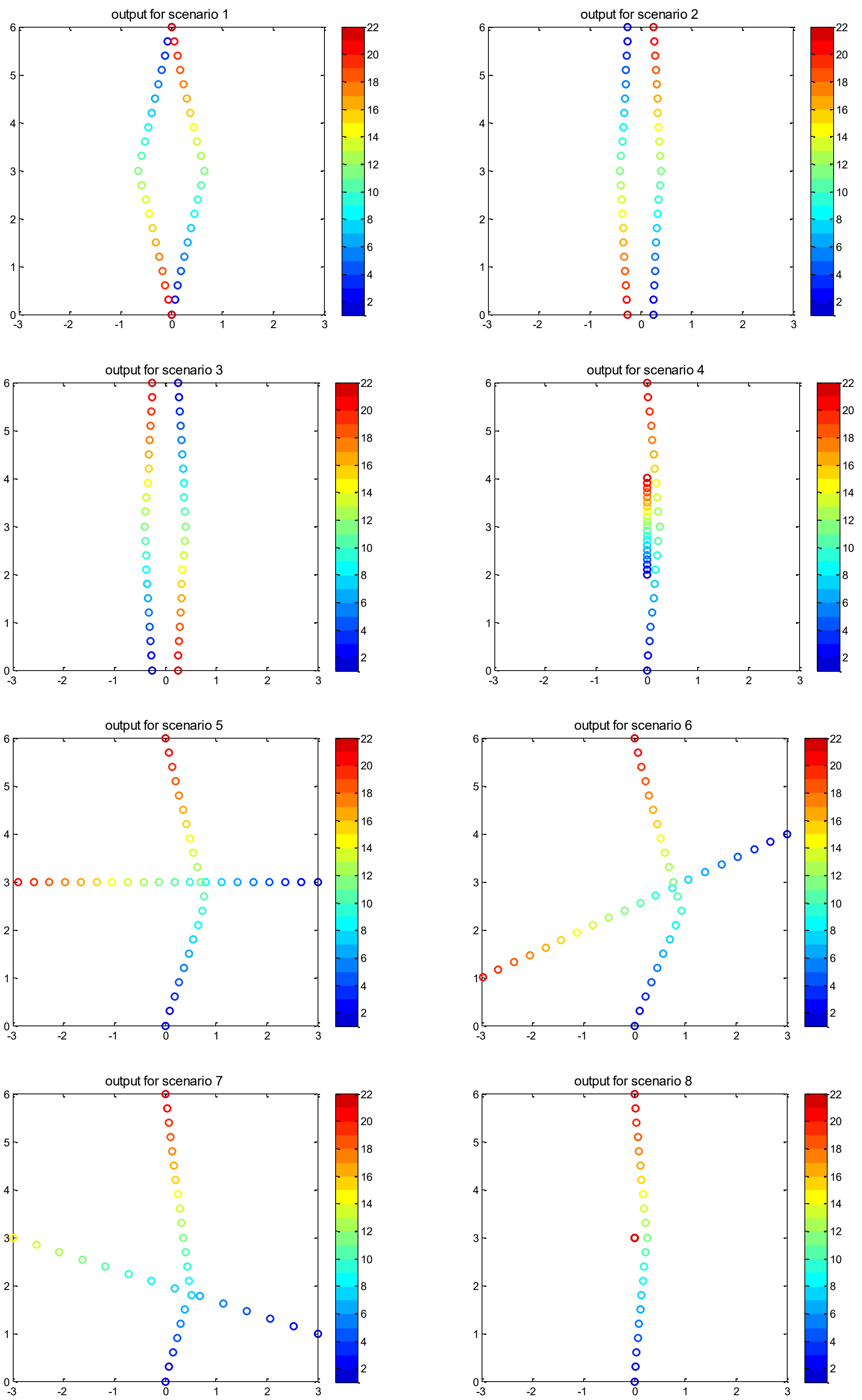

Fig. 5. Algorithm outputs for traffic scenarios.

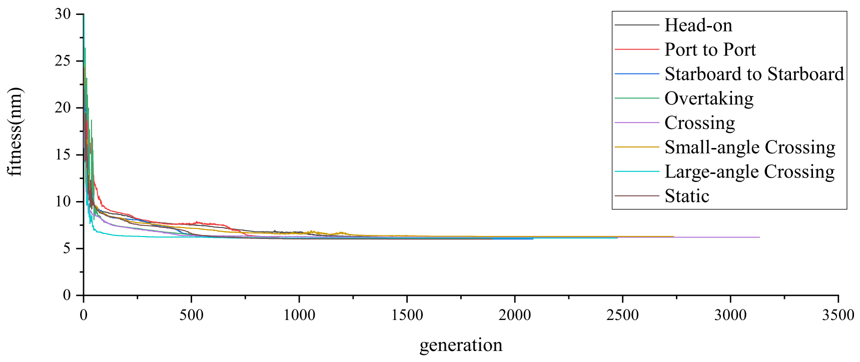

Fig. 6. Convergence process of the DE algorithm for all scenarios.

of the TS; these actions conformed to COLREGS rule 15. The OS avoided the TS as a static obstacle in scenario 8.

The convergence process of the DE algorithm for all scenarios are shown in Fig. 6. Generally, the algorithm can converge rapidly within approximately 1000 generations to determine the optimal solution, where the fitness value fluctuated greatly before 500 generations and gradually stabilized after 1000. The earliest convergence of the fitness value occurred in scenario 8 (static obstacle), whereas the latest convergence of fitness occurred in scenario 6 (small-angle crossing) because of the increased complexity of the solution space.

Numerical results for the simulation of all scenarios are listed in Table 2. Different from the fitness convergence process, the algorithm simulated the fewest generations (1,787) in scenario 4 (overtaking) and the most (3,136) in scenario 5 (crossing). This is because alteration of the fitness value in the next generation is affected by uncertainty and randomness.

The relative distance between ships in all scenarios are shown in Fig. 7. The shortest relative distance (0.25 nm) was observed in scenarios 4 and 8; this was the minimum safe distance set in advance. All other scenarios had higher minimum values. Because ship domain is used to assess collision risk, the dimensions are calculated on the basis of the minimum safe distance between ships. In general, the algorithm can generate an accurate optimal path that maintains a safe distance from obstacles.

4. Discussion

As demonstrated in the previous section, the path planning algorithm can produce a satisfactory output for different encounters involving dynamic and static ships. Of the eight scenarios, seven complied with COLREGS; the exception was scenario 3. The algorithm could generate navigation paths with immediate maneuvers that avoid collision with other ships in certain scenarios. In addition, the algorithm generated the corresponding path of the TS by reversing the roles of the OS and TS under the same traffic scenario, proving the usefulness of this algorithm when employed for multiple ships. Because the algorithm involved stochastic processes, each scenario was set to run 50 times. Then, the performance of the algorithm was analyzed and compared with that of the PSO algorithm under the same scenario. The version of the PSO algorithm followed the format of a previous study (Kang et al., 2018). Thus, details of the PSO algorithm are not discussed further.

4.1. Optimality

To compare the convergence process of the two algorithms, scenario 1 was respectively calculated by DE and PSO algorithms, as shown in Fig. 8. The fitness value of the PSO algorithm fluctuates greatly

Table 2. Simulation results.

Scenario	Encounter type	The optimal path length (nm)	DCPA(nm)	generation
1	Head-on	6.1379	1.2927	2477
2	Port to Port	6.0077	0.8000	2063
3	Starboard to Starboard	6.0077	0.8000	2086
4	Overtaking	6.0209	0.2500	1787
5	Crossing	6.1996	0.6500	3136
6	Small-angle Crossing	6.2910	0.5910	2738
7	Large-angle Crossing	6.1081	0.3594	2472
8	Static	6.0209	0.2500	1895

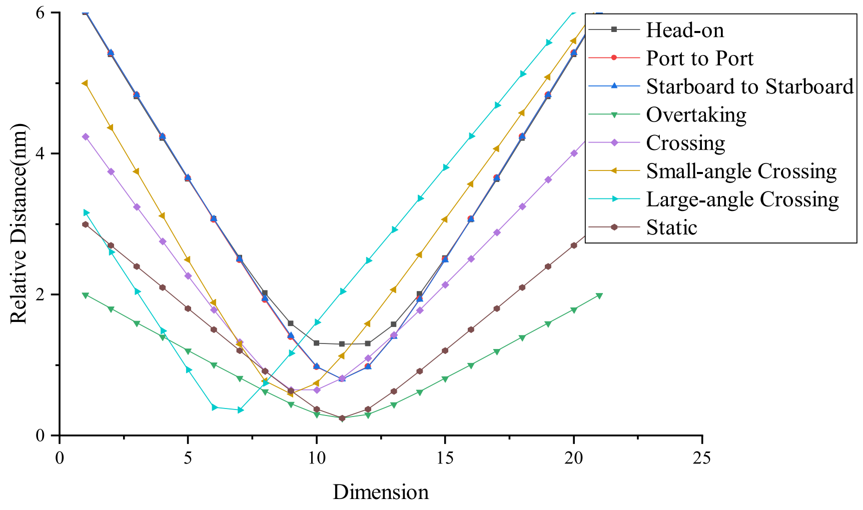

Fig. 7. Relative distance between ships in all scenarios.

before 2500 generations and converges near the optimal solution at approximately 3000 generations, whereas the overall fluctuation of the DE algorithm is small and gradually converges after 1000 generations. In other words, the DE algorithm can converge to the optimal solution earlier than the PSO algorithm can for that particular path optimization problem.

A comparison of simulation results between DE and PSO for all scenarios is provided in Table 3. Little difference exists between the optimization effect of the DE and PSO algorithms; the navigation path generated by the DE algorithm was slightly more optimal in all scenarios. Furthermore, the DE algorithm can generate the optimal path more quickly because it has fewer generations than the PSO algorithm in all scenarios.

4.2. Consistency

The simulation results shown in the previous section are the best results selected from repeated computations of each scenario, and they cannot confirm the consistency of the algorithm. To focus on testing consistency, the optimal path length and distance to the closest point of approach (DCPA) of all simulations were recorded and their mean and standard deviation were calculated; the results were compared with those for the PSO algorithm.

Figure 9 shows the comparison of the optimal path length calculated by the DE and PSO algorithms for all scenarios. The mean of the optimal path length calculated by the DE algorithm was only approximately 0.0001 nm longer than that calculated by the PSO algorithm in scenario 1, whereas the average lengths of the optimal paths returned by both algorithms were the same in scenario 3. As for other scenarios, the optimal DE algorithm paths were slightly shorter than those of the PSO algorithm. However, the standard deviation difference was even more meaningful than to that of the mean. The standard deviation of the DE algorithm is smaller than that of the PSO algorithm in each scenario.

Figure 10 shows a comparison of the DCPA calculated by the DE and PSO algorithms for all scenarios. Except for subtle differences in scenario

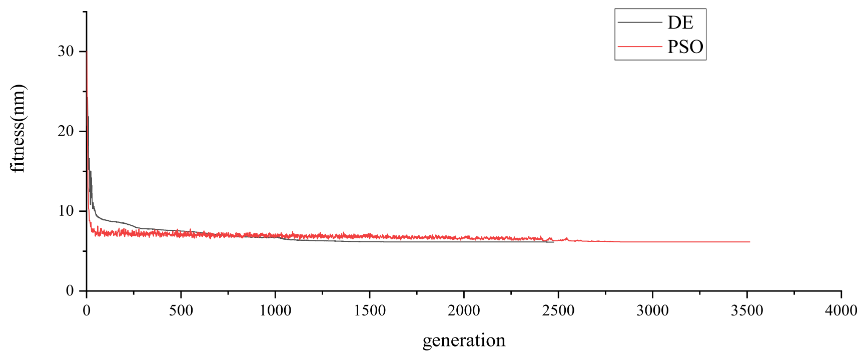

Fig. 8. Convergence process of DE and PSO algorithms for scenario 1.

Table 3. Comparison of simulation results between DE and PSO algorithms.

Scenario	Encounter type	The optimal path length (nm)		generation	
		DE	PSO	DE	PSO
1	Head-on	6.1379	6.1380	2477	3516
2	Port to Port	6.0077	6.0078	2063	3588
3	Starboard to Starboard	6.0077	6.0079	2086	3097
4	Overtaking	6.0209	6.0210	1787	3493
5	Crossing	6.1996	6.1999	3136	3719
6	Small-angle Crossing	6.2910	6.2915	2738	3391
7	Large-angle Crossing	6.1081	6.1086	2472	4155
8	Static	6.0209	6.0211	1895	3055

2–3 and 7, the DCPA means of both algorithms were the same. The standard deviations of DCPA received with the use of both algorithms are almost negligible in the majority of scenarios. In general, the DE algorithm slightly outperforms the PSO algorithm in terms of consistency.

4.3. Efficiency

The statistical data on computational time are presented and compared in Fig. 11. For both algorithms, the simulation of scenario 8 needed the shortest computational time, while scenario 7 required the longest. Furthermore, the mean computation time of the DE algorithm was significantly shorter than that of the PSO algorithm in each scenario. Hence, the DE algorithm offers better performance in terms of execution speed.

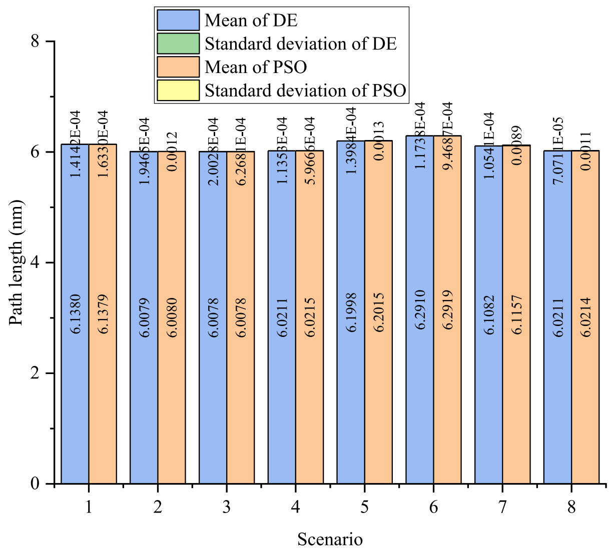

Fig. 9. Comparison of the optimal path length calculated by the DE and PSO algorithm for all scenarios.

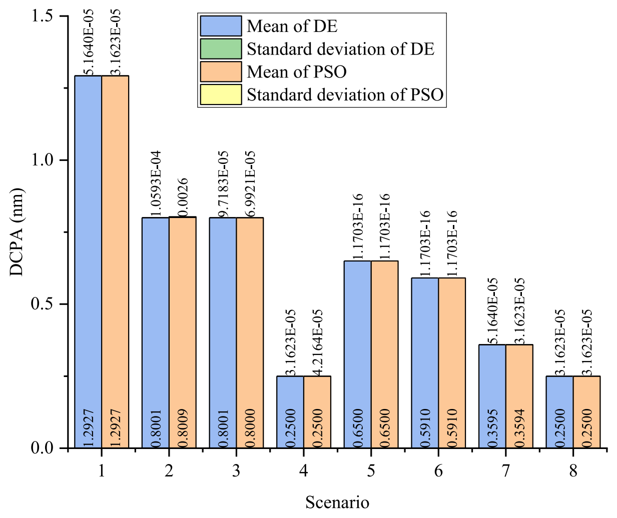

Fig. 10. Comparison of DCPA calculated by the DE and PSO algorithms for all scenarios.

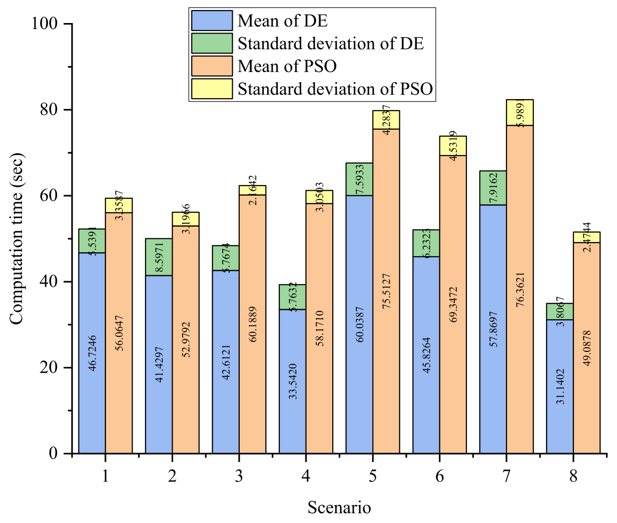

Fig. 11. Comparison of the DE and PSO algorithm computational time for all scenarios.

Nevertheless, the advantage in terms of the PSO computational time is its stability for every run of the calculations because the PSO algorithm had lower standard deviations of computational time in all scenarios.

5. Conclusion

A DE-based algorithm for path planning of ships in open waters is proposed in this paper. The design of this algorithm had two steps. The first step involves assessing the collision risk for each obstacle at a fixed space interval based on the relative position and heading with respect to OS, which was previously presented in Kang et al. (2018). The second step involves planning an optimal path for a ship in a complex navigation environment where static and dynamic obstacles occur. Several traffic scenarios based on typical encounters have been constructed to verify the practicability of the algorithm. The simulation results illustrate the successful application of the proposed algorithm.

The results of the new algorithm were compared with the solutions provided by the PSO algorithm. The DE algorithm achieved better performance with regard to algorithm output optimality, algorithm consistency, and computational time. In summary, the proposed algorithm is capable of consistently returning a safe, optimal, COLREGs-compliant, and real-time navigation path for all traffic scenarios conceived. Future research must focus on path planning environments involving multiple obstacles as well as cooperative path planning for multiple ships within an environment. The main difficulty in this research is achieving cooperation between multiple ships for collision avoidance. A possible solution is to decompose a multi-ship encounter into several two-ship encounters from the separate perspectives of ships to evaluate the danger of a collision between two ships at the same time. Another direction is the development of the path planning algorithm based on improvements in or combinations of existing lgorithms.

Acknowledgements

This work was supported by the National Natural Science Foundation of China, China (Grant No. 71503166) and the Natural Science Foundation of Shanghai, China (Grant No. 16ZR1414600). This manuscript was edited by Wallace Academic Editing.

Conflict of interest statement

The authors declared that they have no conflicts of interest to this work. We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

References

[1] Chang KY, Jan GE, Parberry I. A method for searching optimal routes with collision avoidance on raster charts. J Navig 2003;56:371–84.
[2] Cheng XD, Liu ZY. Study on ship collision avoidance trajectory optimization in inland waterways with genetic algorithm. Navigation of China 2006;67:38–46.
[3] Hwang CN, Yang JM, Chiang CY. The design of fuzzy collision avoidance expert system implemented by h1-autopilot. J Mar Sci Technol 2001;9:25–37.
[4] Hwang CN. The integrated design of fuzzy collision-avoidance and $H\infty$-Autopilots on ships. J Navig 2002;55:117–36.
[5] Kang YT, Chen WJ, Zhu DQ, Wang JH, Xie QM. Collision avoidance path planning for ships by particle swarm optimization. J Mar Sci Technol 2018;26(6):777–86.
[6] Lazarowska A. Ship's trajectory planning for collision avoidance at sea based on ant colony optimisation. J Navig 2015;68:291–307.
[7] Lazarowska A. A new deterministic approach in a decision support system for ship's trajectory planning. Expert Syst Appl 2017;71:469–78.
[8] Lisowski J. Computational intelligence methods of a safe ship control. Procedia Computer Science 2014;35:634–43.
[9] Storn R, Price K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 1997;11(4):341–59.
[10] Szlapcynski R. A new method of ship routing on raster grids, with turn penalties and collision avoidance. J Navig 2006a;59:27–42.
[11] Szlapcynski R. Evolutionary sets of safe ship trajectories within traffic separation schemes. J Navig 2013;66:65–81.
[12] Szlapcynski R. Evolutionary planning of safe ship tracks in restricted visibility. J Navig 2015;68:39–51.
[13] Tam CK, Bucknall R, Greig A. Review of collision avoidance and path planning methods for ships in close range encounters. J Navig 2009;62:455–76.
[14] Tam CK, Bucknall R. Path-planning algorithm for ships in close-range encounters. J Mar Sci Technol 2010;15:395–407.
[15] Tsou MC, Hsueh CK. The study of ship collision avoidance route planning by ant colony algorithm. J Mar Sci Technol 2010a;18:746–56.
[16] Tsou MC, Kao SL, Su CM. Decision support from genetic algorithms for ship collision avoidance route planning and alerts. J Navig 2010b;63:167–82.
[17] Zeng XM. Evolution of the safe path for ship navigation. Appl Artif Intell 2003;17:87–104.