VERSITA

Int. J. Appl. Math. Comput. Sci., 2012, Vol. 22, No. 4, 999–1009
DOI: 10.2478/v10006-012-0074-x

amcs

CUSTOMIZED CROSSOVER IN EVOLUTIONARY SETS OF SAFE SHIP TRAJECTORIES

RAFAŁ SZŁAPCZYŃSKI *, JOANNA SZŁAPCZYŃSKA **

* Department of Marine Mechatronics, Faculty of Ocean Engineering and Ship Technology
Gdańsk University of Technology, Narutowicza 11-12, 80-233 Gdańsk, Poland
e-mail: rafal@pg.gda.pl

**Department of Navigation, Faculty of Navigation
Gdynia Maritime University, Morska 81-87, 81-225 Gdynia, Poland
e-mail: asiasz@am.gdynia.pl

The paper presents selected aspects of evolutionary sets of safe ship trajectories—a method which applies evolutionary algorithms and some of the assumptions of game theory to solving ship encounter situations. For given positions and motion parameters of the ships, the method finds a near optimal set of safe trajectories of all ships involved in an encounter. The method works in real time and the solutions must be returned within one minute, which enforces speeding up the optimisation process. During the development of the method the authors tested various problem-dedicated crossover operators to obtain the best performance. The results of that research are given here. The paper includes a detailed description of these operators as well as statistical simulation results and examples of experiment results.

Keywords: evolutionary algorithms, ship collision avoidance, decision support systems.

1. Introduction

There is a number of approaches to solving multi-ship encounter situations. One of the most successful and flexible methods is searching for the ship trajectory by evolutionary algorithms. The method has been first proposed by Śmierzchalski and Michalewicz (2000), and since then similar approaches have been attempted by other researchers: the evolutionary method may be applied for finding an optimal path (Zeng, 2003) as well as the optimal collision avoidance manoeuvre (Tsou *et al.*, 2010). In short, these methods use genetic or evolutionary algorithms, which, for a given set of pre-determined input trajectories, find a solution that is optimal according to a given fitness function. A similar approach based on Ant Colony Optimisation (ACO) of course alteration manoeuvres was presented by Tsou and Hsueh (2010). Apart from these, automatic collision avoidance of ships using an artificial potential field and speed vector (Xue *et al.*, 2009) is also employed, which is an adaptation of the Potential Field Method (PFM) widely used for navigating mobile robots (Pradhan *et al.*, 2006). The ACO and PFM based methods face the same limitations as the general evolutionary approach mentioned before, that is, focusing on one's own ship only and assuming that target motion parameters not to change.

The authors have proposed a new approach where, instead of finding an optimal ship trajectory for the unchanged courses and speeds of targets, an optimal set of safe trajectories of all ships involved in an encounter is sought. The method is called *evolutionary sets of safe trajectories* and its earlier version has been presented by one of the authors (Szłapczyński, 2011). Optimising a set of trajectories instead of a single trajectory drastically magnifies the searching space. At the same time, working in a restricted water environment and trying to obey the international collision avoidance rules (also known as COLREGS) produces multiple constraints, which make it more difficult to find any acceptable solution. All these factors contribute to an atypical optimisation problem, often unsolvable by pure genetic algorithms in a given time, strictly limited because of operating while the ships are approaching each other or the landmasses. To meet all the requirements, the authors had to apply a number of special problem-dedicated operators, modify the traditional evolutionary order of operations and experiments with crossover methods. The

paper presents a description and discussion of the choices and modifications made in the crossover phase of the evolutionary cycle, as well as its implications for other phases.

The rest of the paper is organised as follows. In the next section the task of finding sets of safe trajectories is presented as an optimisation problem. In Section 3 the evaluation of individuals is described, including its consequences for an evolutionary cycle. The investigated crossover operators are presented in Section 4 and their test results and conclusions drawn from the experiments in Section 5. Finally, a summary is provided in Section 6.

2. Optimisation problem

For a given multi-ship encounter, a set of safe ship trajectories which minimises an average way loss is sought. It is assumed that we are given the following data:

- stationary constraints (defined by a map including navigational obstacles),
- positions, courses and speeds of all ships involved,
- ship domain defined for each ship considered (a domain is an area around the ship free from other ships, obstacles, etc.),
- times necessary for accepting and executing the proposed manoeuvres.

Knowing all the above mentioned parameters, the goal is to find a set of trajectories which minimises the average way loss spent on manoeuvring, while fulfilling the following conditions:

- key COLREGS rules (Rules 13–17) are obeyed,
- none of the stationary constraints is violated,
- none of the ship domains is violated,
- the minimal acceptable course alteration should not be too small (the minimal alteration of 15 degrees has been assumed here),
- the maximal acceptable course alteration should not be too large (the maximal alteration of 60 degrees has been assumed here),
- speed alteration is not to be applied unless necessary (collision cannot be avoided by course alteration up to 60 degrees),
- a ship only manoeuvres when obliged to,
- in the case of head-on and crossing encounters, manoeuvres to starboard are favoured over manoeuvres to port board.

The conditions are mostly either imposed by COLREGS (Cockroft and Lameijer, 1993) and good marine practise or by economics. In particular, course alterations less than 15 degrees might be misleading for ARPA systems (problems with detection), and course alterations larger than 60 degrees are inefficient. Also, ships should only manoeuvre when necessary, since each manoeuvre makes it harder to track its motion parameters for the ARPA systems of other ships. An additional computational constraint results from the fact that due to the optimisation being performed in real time (with the ships approaching each other and the obstacles), the solution should be returned within a short time specified by the operator of the system (by default, one minute is assumed).

3. Evaluation of individuals and its consequences

In an evolutionary method all individuals (here sets of trajectories) are evaluated by a specially designed fitness function, which should reflect optimisation criteria and constraints (Michalewicz and Fogel, 2004). In this section we show how this fitness function is formulated.

3.1. Basic criterion: Minimizing way loss. The basic criterion is the economic one—minimizing way losses of trajectories in a set. For each of the trajectories, a trajectory economy factor, tef, is computed according to the following formula:

$$tef_i = \frac{tr\_length_i - tr\_way\_loss_i}{tr\_length_i}, \qquad (1)$$

where: i is the index of the current ship [/], *tef* is the economy factor of the i-th trajectory [/], *tr_length* is the total length of the i-th ship's trajectory [nautical miles], *tr_way_loss* is the total way loss of the i-th ship's trajectory [nautical miles] computed as a difference between the trajectory length and the length of a segment joining the trajectory's start and endpoints. As can be seen, *tef* is always a number from the interval (0,1].

3.2. Detecting and penalizing static constraint violations. The method uses a vector map of a given area. The authors have decided not to process the vector map directly for constraint violation detection (due to an extremely high time complexity of such an approach), but to use the vector map for generating the bitmap of an area. Although it is a time-consuming operation, fortunately, it is enough to generate such bitmaps off-line and only once for each area. Then, when the method runs in real time, each bitmap cell, traversed by the trajectory of a ship, is read and checked for belonging to the landmass or a safety isobate. For a bitmap whose detail level

reflects that of a given vector map, the computational time would be proportional to the number of traversed cells and thus acceptable. This approach is also flexible in terms of implementation of bathymetry: for a cell containing information on the water depth, it is easy to check whether or not it is passable for a particular ship. It is also possible to apply the bitmap strategy for ECDIS software systems. A bitmap for the area of interest only (not the whole map) would be generated on-line prior to the land crossing checkup. An example of such a bitmap-based ECDIS solution is NaviWeather by NavSim with weather routing software. A description of the software is intended to be published by Szłapczyńska (2012).

After the trajectory economy factor (1) has been computed, the static constraints are handled by introducing penalties for violating them. For each trajectory its static constraints factor, *scf*, is computed. The static constraints are always valid and their violations must be avoided at all cost. Therefore, penalties applied here are the most severe—hence the square in the following formula:

$$scf_i = \left(\frac{tr\_length_i - tr\_cross\_length_i}{tr\_length_i} \right)^2, \quad (2)$$

where *scf* is the static constraints violation factor of the i-th ship's trajectory [/], *tr_cross_length* is the total length of the parts of the i-th ship's trajectory which violate stationary constraints [nautical miles].

The static constraint factor is a number from the interval $[0, 1]$, where the '1' value means no static constraint violation (no landmasses or other obstacles are crossed) and the '0' value is for trajectories crossing landmasses on their whole length.

3.3. Detecting and penalising collisions with other ships. The algorithm for detecting ship-to-ship collisions is as follows. Each ship's trajectory is checked against those of all other ships. For each pair of ships, the start and end times of each trajectory's segments are computed. If two segments of the two trajectories overlap in time, they are checked for geometrical crossing. In the case of a crossing, a special collision risk measure—the approach factor value (Szłapczyński, 2006) is computed. Then, if the approach factor value indicates a collision, the type of encounter (head-on, crossing or overtaking) is determined on the basis of the ships' courses, and it is decided which ship is to give way (both ships in case of a head-on encounter). The collision is only registered for the give way ship and the information on the collision is stored in the corresponding trajectory data structure.

Analogically to the static constraint factor, the collision avoidance factor, *caf*, is computed to reflect the ship's collisions with all other privileged ships as shown by (3):

$$caf_i = \prod_{j=1, j\neq i}^{n} \left(\min \left(f \min_{ij}, 1 \right) \right), \quad (3)$$

where *caf* is the collision avoidance factor of the i-th ship's trajectory [/], n is the number of ships [/], j is the index of a target ship [/], *fmin* is the approach factor value (Szłapczyński, 2006) for an encounter of ships i and j; if the i-th ship is the privileged one, the potential collision is ignored and the approach factor value is equal to '1' by definition. [/]

The collision avoidance factor is a number from the interval $[0, 1]$ where '1' means no ship domain violation and '0' means a crash with at least one of the targets.

3.4. Detecting and penalising COLREGS violations. The three most common types of COLREGS violations are as follows:

- a ship does not give way when it should,
- a ship gives way when it should not (making unexpected and misleading manoeuvres),
- a ship manoeuvres to port board when it should manoeuvre to starboard.

Each of these three situations may happen on either open or restricted waters, which gives us a total of six cases to handle. The difficulty with deciding whether or not a ship has acted lawfully lies in the nature of evolutionary algorithms as well as in the nature of the problem itself: COLREGS specify only the procedures for ship-to-ship encounters. When looking at a set of ship trajectories for a multi-target encounter, it is sometimes impossible to tell what was the reason for a particular manoeuvre: which ship was given way intentionally, and which one benefited from it only accidentally. Therefore the final COLREGS violations detection rules applied in the method include the following.

1. On open waters:
   (a) if a ship is not obliged to give way to any other ship, any manoeuvre it performs is registered as a COLREGS violation,
   (b) if a ship is obliged to give way and does not perform a manoeuvre, it is registered as a COLREGS violation,
   (c) all manoeuvres to port board are registered as COLREGS violations.
2. On restricted waters: Here, as explained before, every trajectory node, which is a part of a manoeuvre, contains information on the reason why this

particular node has been inserted or shifted—land or other stationary obstacle avoidance, target avoidance or accidental manoeuvre generated by evolutionary mechanisms. Based on this, COLREGS violations are registered as follows:

(a) if a ship does not initially have to give way to any target and its first manoeuvre has a reason other than static constraint violation avoidance, it is registered as a COLREGS violation,

(b) any manoeuvre to port board for a reason other than static constraint violation avoidance is registered as a COLREGS violation.

The COLREGS violations are secondary to static constraint violations and to collisions with other ships, and therefore the authors have decided to penalize them moderately, to make sure that constraints from the previous two points are met first. The COLREGS compliance factor, *ccfi* is computed according to the following formula:

$$ccf_i = 1 - \sum_{k=1}^{m} [COLREGS\_violation\_penalty_k], \quad (4)$$

where *ccf* is the COLREGS violation factor of the i-th ship's trajectory [/], m is the number of COLREGS violations registered for the current ship [/], k is the index of a registered violation [/], *COLREGS_violation_penalty* is the penalty for the k-th of the registered COLREGS violation [/].

The penalty values for all registered COLREGS violations are configurable in the method and are set to 0.05 by default.

3.5. Fitness function value. Once all aforementioned factors have been computed, the fitness function value is calculated. The function is normalised and has been designed to keep a relatively high resolution of evaluation: minor stationary constraints violations are penalised similarly as major collisions with other ships and minor collisions with other ships are penalised similarly as multiple COLREGS violations. The final fitness function is as follows:

$$fitness = \frac{1}{n} \sum_{i=1}^{n} tr\_fitness_i, \quad (5)$$

where

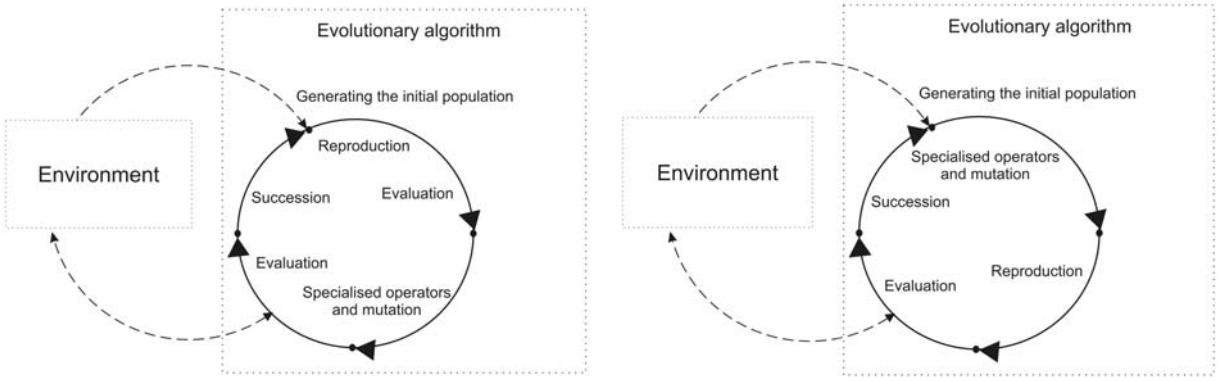
$$tr\_fitness_i = tef_i \cdot scf_i \cdot caf_i \cdot ccf_i. \quad (6)$$

The final fitness function value assigned to an individual is an arithmetical average of fitness function values computed for all trajectories. It is disputable whether all trajectories should have the same impact on the final fitness function value (as is done here) or the trajectory fitness function values should be taken with weights proportional to the trajectory lengths. When combined with the formula for the trajectory economy factor, the current approach means that we are trying to minimise the average relative way loss computed over all trajectories, instead of the total absolute way loss (with weights being used). However, experiments have shown that minimising the total absolute way loss leads to discrimination of ships whose basic trajectories are shorter and to their large relative way losses.

3.6. Changes in the evolutionary cycle. Due to the complex violation detection process, the evaluation is the most time-consuming phase of the evolutionary algorithm, with the computational times of other stages being relatively insignificant. Combined with the fact that the evolutionary process is executed in real time, this seriously limits the number of generations we can afford. For the most complex scenarios including several ships of various dynamics, complex ship domain models and restricted waters with multiple obstacles, only up to 200 generations can be processed for a population of 100 members, even if more generations could cause a further increase in the fitness function values. Thus, an obvious conclusion is that it is necessary to achieve as much progress as possible in each of the generations.

The authors have done this by investing more computational time in specialised operators which fix constraint violations (Szłapczyński and Szłapczyńska, 2011). However, these operators need the violations to be detected and registered first, which means that this phase must be preceded by evaluation. Therefore a new evolutionary scheme has been introduced, which is shown in Fig. 1(a). The obvious disadvantage of the modified evolutionary algorithm shown above is doubling the time-costly evaluation phase. It would be performed first after reproduction and then again after specialised operators/mutation phase. Doubling the evaluation phase in a cycle increases the total computational time approximately 1.5 times (the extra evaluation after reproduction is done for a population half the size of the one after mutation).

To shorten the process, the authors have decided to try a radical change in the order of operations within the algorithm. The reproduction phase and the specialised operations mutation phase have switched places so that the evaluation could be done only once for each cycle—directly preceding succession. The result is shown in Fig. 1(b). Unfortunately, using this scheme means that the crossover has to be limited to operators which do not need evaluation data. Thus we are facing the question whether it is better to use an extended set of crossover



(a) (b)

Fig. 1. Changed evolutionary cycle: with a doubled evaluation (a), with a single evaluation and an amended order of evolutionary operations (b).

operators (at the cost of doubling the evaluation phase) or a limited set of operators (and have a single evaluation phase). This issue is addressed in Section 4.2.

4. Crossover operators

The majority of contemporary AI systems using Evolutionary Algorithms (EAs) are hybrid ones. EAs may be successfully combined with neural networks (Styrcz *et al.*, 2011), reinforcement learning (Krawiec *et al.*, 2011), ensemble machine learning methods (Troć and Unold, 2010) or heuristics which reduce the search space (Belter and Skrzypczyński, 2010). Hybrid approaches may also be utilised at various stages of EAs, including crossover (Jóźwiak and Postula, 2002).

4.1. Crossover in EAs. Eiben and Schoenauer (2002) summarise that in GAs crossover is considered to be the essential variation operator while mutation is only a background necessity. On the other hand, in the past many researchers did not use any crossover at all. Today the general opinion is that the answer is problem-dependent. If there exists an appropriate crossover for the problem at hand, it should be tried (otherwise mutation alone might be sufficient to find good solutions and the resulting algorithm can still be called an evolutionary algorithm). Crossover operators can be divided into categories in several ways. The most commonly used classifications are

- one point crossover and multi-point (including two-point) crossover,
- intermediate recombination and discrete recombination.

One-point crossover is strongly supported by the so-called building block hypothesis. It states (Beyera *et al.*, 2002) that such crossover often allows putting together good parts of one parental bit string with other good parts of the second parent delivering an even better combination of both in an offspring. In two-point crossover, the bit strings of two parents are cut at two random positions (instead of one) and put together by exchanging the innermost parts between the parents, thus creating two offspring at a time. A disadvantage of two-point crossover (Jóźwiak and Postula, 2002) is a high chance of producing children identical or very similar to their parents by converging populations (i.e., if the parent chromosomes are similar to each other). This may result in premature convergence of the algorithm. However, multi-point crossover has been successfully used in problems where individuals are complex and may be easily decomposed, e.g., the spanning tree problem (Julstrom, 2004).

In the case of intermediate recombination (Beyera *et al.*, 2002) the average of the parental variable values is transferred to the offspring, whereas discrete recombination chooses each component from one of the parents at random. No checking is imposed whether the parents involved are all different, and there is no mating selection—all parents have the same chance to be chosen.

Apart from the basic types, many researchers apply more customized crossover, an example of which is the merge operator (Jóźwiak and Postula, 2002)—a deterministic crossover operator which, instead of combining two population members at random, tries to combine two population members in such a way that their good features are combined and the bad ones are eliminated. The choice of crossover operators has an

impact on faster or slower convergence (Kowalczuk and Białaszewski, 2006) and therefore sometimes, e.g., in parallel genetic algorithms for continuous optimisation (Alba *et al.*, 2004), special crossover operators promoting exploration or exploitation are introduced.

Additionally, in EAs, elements of traditional GA crossover operators (which do not express the characteristics of the selected individuals within a population) may be combined with Estimation of Distribution Algorithms (EDAs), where these characteristics are taken into account by considering the interdependencies between the different variables that form an individual (Miquelez *et al.*, 2004). For selected problems, it is also possible to design crossover operators that guarantee generating legal offspring, e.g., for job scheduling problems (Mesghouni *et al.*, 2004). Clearly, the choice of crossover operators to be applied (or the lack of them) may be essential for the results returned by EAs and therefore this issue should be investigated for the problem at hand.

4.2. Crossover in the ESoSST method. Contrary to genetic algorithms, where the individuals are coded (e.g., binary), here the individuals are implemented directly using objects, structures and arrays of real numbers. In the crossover phase, pairs of individuals (parents) are crossed to generate new individuals (offspring).

4.2.1. Mating selection: Methods and probabilities. Four types of parent selection have been taken into account by the authors, with the total probability of being selected as a parent established as follows.

(a) Threshold:

$$P_k = \begin{cases} 0 & \text{if} \quad \mathit{fitness}_k < \mathit{threshold}, \\ 1 & \text{if} \quad \mathit{fitness}_k \geq \mathit{threshold}. \end{cases} \tag{7}$$

(b) Random proportional (roulette wheel):

$$P_k = 1 - \left(1 - \frac{\mathit{fitness}_k}{\sum_{i=1}^{N} [\mathit{fitness}_i]} \right)^n . \tag{8}$$

(c) Modified random proportional (roulette wheel with scaled fitness function values):

$$P_k = 1 - \left(1 - \frac{\mathit{fitness}_k - \mathit{min\_fitness}}{\sum_{i=1}^{N} [\mathit{fitness}_i] - \mathit{min\_fitness}} \right)^n , \tag{9}$$

where *min_fitness* is the minimal fitness function value over all population. The uniform version is

$$P_k = 1 - \left(1 - \frac{1}{n} \right)^n , \tag{10}$$

where n is the population size.

All four types of mating selection were tested and the differences in results for various methods were insignificant. Therefore uniform selection was chosen, which does not need the evaluation data and enables the authors to apply the evolutionary scheme from Fig. 2(b).

4.2.2. Crossover operators applied. Four types of crossover operators have been designed and implemented:

(a) fitness-based trajectory exchange: an offspring inherits whole trajectories from both parents and the higher-valued of the two possible trajectories is chosen (Fig. 2),

(b) random trajectory exchange: an offspring inherits whole trajectories from both parents and the choice of a particular trajectory (from the first or the second parent) is done randomly (again Fig. 2),

(c) one-point trajectory crossover: each of the trajectories of the offspring is a crossover of the appropriate trajectories of the parents (Fig. 3),

(d) intermediate recombination of nodes: each node of a trajectory is a crossover of the nodes in the parents' trajectories (Fig. 4).

Another possible operator, multi-point trajectory crossover, has not been applied due to relatively short trajectories and the high probability of generating identical offspring. Of the operators applied, the first one (fitness-based trajectory exchange) was designed to combine the best features of two parent individuals and it can be classified as the "merge operator" mentioned in the previous section. It is the only one of the operators applied in the paper which should statistically produce the offspring higher valued than the parents. Unfortunately, using this operator has to be preceded by the evaluation phase, which enforces applying the evolutionary scheme with a doubled evaluation phase, cf. Fig. 1(a). Therefore, during experiments, it has been tested whether its advantages compensate for the additional computational time, which results in a reduced number of possible generations.

As for the other operators, there is no guarantee or even a high probability that offspring of two highly valued parents will be highly valued itself. For example,

in the case of the random trajectory exchange (b), the resulting trajectories may not fit other trajectories (collisions between ships). Therefore, to make sure that the best individuals will not be lost (the parents might be better fitted than their offspring), the overlapping populations are used. As a result, reproduction doubles the temporary population size.

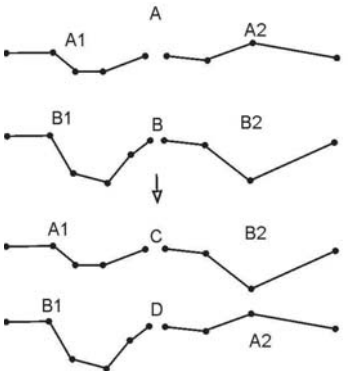


Fig. 2. Crossover inheriting entire trajectories utilised by the fitness-based trajectory exchange and the random trajectory exchange.

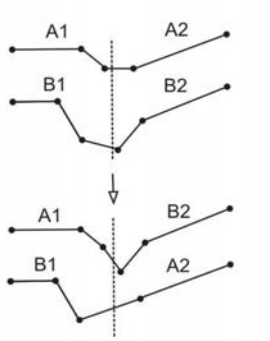


Fig. 3. Crossover of trajectories utilised by one-point trajectory crossover.

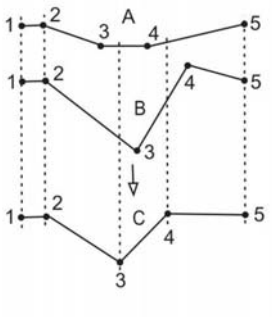


Fig. 4. Crossover of nodes utilised by intermediate recombination of nodes.

5. Comparison of crossover approaches and customized operators

5.1. Compared approaches. The simulation experiments were aimed at comparing competitive versions of the method, namely, those having different combinations of crossover operators and different evolutionary schemes applied. The basic test parameters are gathered in Table 1. The 91 randomly generated test scenarios cover open and restricted waters, all typical combinations of courses with the number of ships ranging from 2 to 6.

Table 1. Test parameters.

Test scenarios	91
Runs	10
Generations	100 or 200
Population size	100
Basic mutation probability	0.05
Specialised operators probability	1.00

Test results for various settings are provided in Tables 2 and 3. For each of the sets of crossover operators considered both evolutionary schemes (with single evaluation as in Fig. 1(a) and with doubled evaluation as in Fig. 1(b)) were tried. It is assumed that the method would normally run for 100 generations, but the tests were also run for 200 generations to find out what further progress is possible.

5.2. Experiments results and conclusions. The results led the authors to the following conclusions:

1. The differences in average fitness function values between the tested versions are insignificant when compared with the differences obtained when testing the method with various fixing operators and mutation settings (Szłapczyński and Szłapczyńska, 2011). When combined with generally high fitness function values, this suggests that the designed crossover operators (random trajectory exchange, one-point trajectory crossover and intermediate recombination of nodes) are effective enough and it is unlikely to improve the method's effectiveness by experimenting with new ones.

2. The extended set of crossover operators (fitness-based trajectory exchange, random trajectory exchange, one-point trajectory crossover and intermediate recombination of nodes) causes only a minor increase in the fitness function and is not worth the additional computational time spent on the extra evaluation phase preceding the crossover phase.

Table 2. Statistical test results obtained for the method with a single evaluation phase and mutation/fixing operators preceding crossover (Fig. 1(a)).

Crossover operators used	Number of generations	Average fitness function values
Random trajectory exchange and one-point trajectory crossover	100 200	0.9756 0.9764
Random trajectory exchange, one-point trajectory crossover and intermediate recombination of nodes	100 200	0.9768 0.9773

Table 3. Statistical test results obtained for the method with the evaluation phase doubled and crossover preceding mutation/fixing operators (Fig. 1(b)).

Crossover operators used	Number of generations	Average fitness function values
Random trajectory exchange and one-point trajectory crossover	100 200	0.9743 0.9749
Random trajectory exchange, one-point trajectory crossover and intermediate recombination of nodes	100 200	0.9754 0.9762
Fitness-based trajectory exchange, random trajectory exchange, one-point trajectory crossover and intermediate recombination of nodes	100 200	0.9770 0.9772

3. The set of three operators causes minor progress when compared with a set of two operators only, but it is obtained at no additional cost (the same computational time), so it might be considered an improvement over the basic set.

4. The experimental evolutionary scheme with mutation and fixing operators preceding the crossover phase (Fig. 1(b)) returned better average results for all combinations of crossover operators and maximum generation numbers. Thus this scheme of the evolutionary algorithm may be considered not only more efficient (saving computational time due to a single valuation phase) but also more effective and generally better suited for the method.

5. As expected, the average fitness function values were always higher for 200 generations than for 100 generations but the difference was always below 0.1% of the average fitness function values. This shows that the progress is still possible with a growing number of generations, but the solutions returned for 100 generations are close enough to the optimal ones to be accepted and recommended by the system.

6. The favourable version of the method is the one with a set of three crossover operators and a modified evolutionary scheme (the last row of Table 2).

The example results returned by the selected version of the method for 100 generations are provided below. In both scenarios, apart from the landmass, a safety isobate (marked as a dotted area in Figs. 5 and 6) also has to be taken into account when manoeuvring on restricted waters.

The first scenario is an overtaking encounter of two ships in a narrow channel. The ship motion parameters are given in Table 4 (Scenario 1). The result, i.e., a set of two trajectories, is shown in Fig. 5.

As can be seen in Fig. 5, the overtaking ship manoeuvres to starboard. The manoeuvre of the overtaking ship is large enough to pass safely ahead of the other ship, even though the other ship also manoeuvres to its starboard to avoid violating the landmass's safety isobate. The fitness function value for this solution is 0.9911, which means that the way loss spent on manoeuvring is less than 1% of the distance covered by the ships.

The second scenario is a more complex one: a crossing encounter of five ships on restricted waters. The ship motion parameters are given in Table 4 (Scenario 2). The resulting set of five trajectories is shown in Fig. 6.

In the solution depicted in Fig. 6 all ships act as recommended by COLREGS. The course alterations are to starboard and each ship passes astern of the ships on its starboard. The fitness function value for this solution is 0.9698, which means about 3% of an average way loss computed over all trajectories.

6. Summary

In the paper a method of solving encounter situations, evolutionary sets of safe trajectories, has been presented with a focus on specialised crossover operators. The

Table 4. Ship parameters for Scenarios 1 and 2.

	Scenario 1		Scenario 2				
	Ship 1	Ship 2	Ship 1	Ship 2	Ship 3	Ship 4	Ship 5
Initial position	107.1;20.1	102.5;20.0	46.9;154.3	39.3;178.6	33.8;170.9	36.5;156.3	55.1;159.4
Goal position	138.3;21.1	142.9;21.3	44.6;179.9	52.2;155.7	57.8;163.4	54.9;178.0	36.4;174.9
Velocity [kn]	15.64	26.12	12.84	13.17	12.60	14.24	12.17

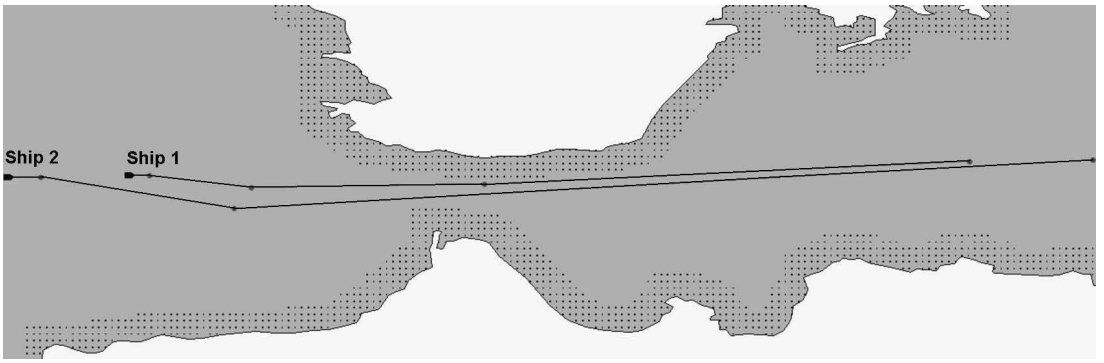


Fig. 5. Resulting set of trajectories for the scenario of an overtaking encounter of two ships in a narrow channel.

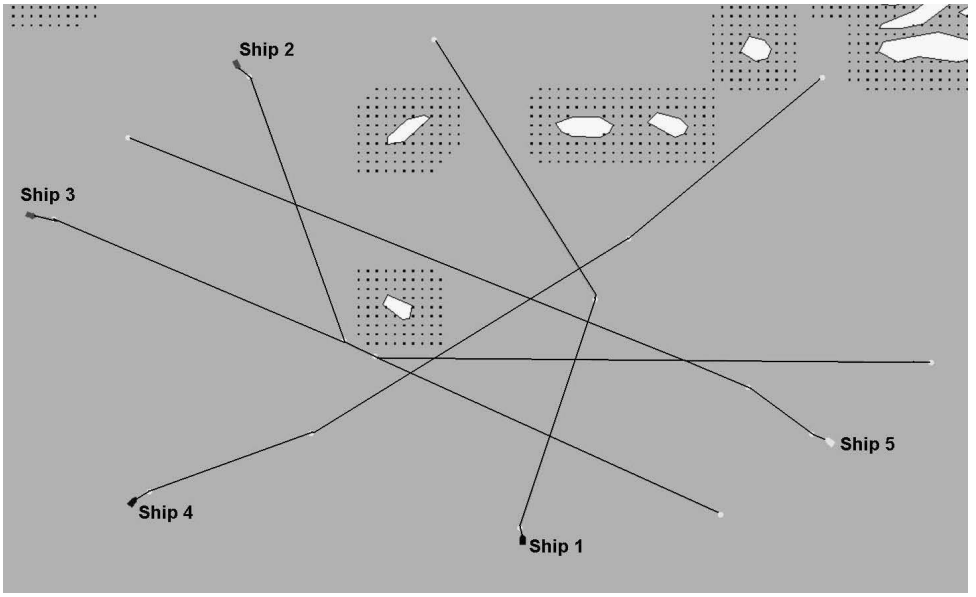


Fig. 6. Resulting set of trajectories for the scenario of a crossing encounter of five ships.

method is a generalisation of evolutionary trajectory determining. A set of trajectories of all ships involved, instead of just ones' own trajectory, is determination. The method avoids violating ship domains and stationary constraints while obeying COLREGS and minimising the average way loss computed over all trajectories. The use of specialised operators, which fix the constraint violations, combined with strict time limits (one minute by default) led to the necessary changes in the evolutionary cycle. The authors have faced an alternative of applying an additional, more conscious crossover operator at the cost of doubling the evolutionary phase or relying on a basic set of two or three random crossover operators which do not need additional evaluation. Altogether, six combinations of crossover operators and evolutionary schemes have been compared in a series

of tests. The results have shown that the method with additional deterministic crossover returns insignificantly better results and its advantages do not compensate for the computational time which has to be spent on the extra evaluation. Another conclusion is that the modified evolutionary cycle which was introduced to save computational time appeared to be slightly more effective than the traditional one for all combinations of crossover operators. The authors' work on the method and its optimal parameters will be continued. The practical implementation of the method requires applying a more precise model of the ship's dynamics and handling Rule 10 of COLREGS, which concerns traffic separation schemes. Both issues are currently being researched by the authors and the results will be presented in the upcoming papers.

Acknowledgment

This research has been funded by the Polish Ministry of Science and Higher Education under the grant no. N N516 186737.

References

Alba, E., Luna, F. and Nebro, A.J. (2004). Advances in parallel heterogeneous genetic algorithms for continuous optimisation, *International Journal of Applied Mathematics and Computer Science* **14**(3): 317–333.

Belter, D. and Skrzypczyński, P. (2010). A biologically inspired approach to feasible gait learning for a hexapod robot, *International Journal of Applied Mathematics and Computer Science* **20**(1): 69–84, DOI: 10.2478/v10006-010-0005-7.

Beyera, H., Schwefela, H. and Wegenerb, I. (2002). How to analyse evolutionary algorithms, *Theoretical Computer Science* **287**(1): 101–130.

Cockroft, A. and Lameijer, J. (1993). *A Guide to Collision Avoidance Rules*, Butterworth-Heinemann Ltd., Oxford.

Eiben, A. and Schoenauer, M. (2002). Evolutionary computing, *Information Processing Letters* **82**(1): 1–6.

Jóźwiak, L. and Postula, A. (2002). Genetic engineering versus natural evolution: Genetic algorithms with deterministic operators, *Journal of Systems Architecture* **48**(1–3): 99–112.

Julstrom, B.A. (2004). Codings and operators in two genetic algorithms for the leaf-constrained minimum spanning tree problem, *International Journal of Applied Mathematics and Computer Science* **14**(3): 385–396.

Kowalczuk, Z. and Białaszewski, T. (2006). Niching mechanisms in evolutionary computations, *International Journal of Applied Mathematics and Computer Science* **16**(1): 59–84.

Krawiec, K., Jaskowski, W. and Szubert, M. (2011). Evolving small-board Go players using coevolutionary temporal difference learning with archives, *International Journal of Applied Mathematics and Computer Science* **21**(4): 717–731, DOI: 10.2478/v10006-011-0057-3.

Mesghouni, K., Hammadi, S. and Borne, P. (2004). Evolutionary algorithms for job-shop scheduling, *International Journal of Applied Mathematics and Computer Science* **14**(1): 91–103.

Michalewicz, Z. and Fogel, D. (2004). *How to Solve It: Modern Heuristics*, Springer-Verlag, Berlin.

Miquélez, T., Bengoetxea, E. and Larrañaga, P. (2004). Evolutionary computation based on Bayesian classifiers, *International Journal of Applied Mathematics and Computer Science* **14**(3): 335–349.

Pradhan, S., Parhi, D., Panda, A. and Behera, R. (2006). Potential field method to navigate several mobile robots, *Applied Intelligence* **25**(1): 321–333.

Śmierzchalski, R. and Michalewicz, Z. (2000). Modeling of a ship trajectory in collision situations at sea by evolutionary algorithm, *IEEE Transactions on Evolutionary Computation* **4**(3): 227–241.

Styrcz, A., Mrozek, J. and Mazur, G. (2011). A neural-network controlled dynamic evolutionary scheme for global molecular geometry optimization, *International Journal of Applied Mathematics and Computer Science* **21**(3): 559–566, DOI: 10.2478/v10006-011-0044-8.

Szłapczyńska, J. (2012). Multicriteria weather routing algorithm (MEWRA) applied to marine weather forecast and analysis tool—NaviWeather by NavSim^{TM}, *European Navigational Conference (ENC), Gdynia, Poland,* (submitted).

Szłapczyński, R. (2006). A unified measure of collision risk derived from the concept of a ship domain, *The Journal of Navigation* **59**(3): 477–490.

Szłapczyński, R. (2011). Evolutionary sets of safe ship trajectories: A new approach to collision avoidance, *The Journal of Navigation* **64**(1): 169–181.

Szłapczyński, R. and Szłapczyńska, J. (2011). Evolutionary sets of safe ship trajectories: Problem dedicated operators, *in* P. Jędrzejowicz, N.T. Nguyen and K. Hoang (Eds.), *Computational Collective Intelligence: Technologies and Applications*, Part II, Lecture Notes in Artificial Intelligence, Vol. 6923, Springer-Verlag, Berlin/Heidelberg, pp. 231–240.

Troć, M. and Unold, O. (2010). Self-adaptation of parameters in a learning classifier system ensemble machine, *International Journal of Applied Mathematics and Computer Science* **20**(1): 157–174, DOI: 10.2478/v10006-010-0012-8.

Tsou, M.C. and Hsueh, C.K. (2010). The study of ship collision avoidance route planning by ant colony algorithm, *Journal of Marine Science and Technology* **18**(5): 746–756.

Tsou, M.C., Kao, S.L. and Su, C.M. (2010). Decision support from genetic algorithms for ship collision avoidance route planning and alerts, *The Journal of Navigation* **63**(1): 167–182.

Xue, Y., Lee, B. and Han, D. (2009). Automatic collision avoidance of ships, *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* **223**(1): 33–46.

Zeng, X. (2003). Evolution of the safe path for ship navigation, *Applied Artificial Intelligence* **17**(2): 87–104.

Rafał Szłapczyński, M.Sc.Eng. in computer science, holds a Ph.D. in technical sciences. Currently he works as an assistant professor at the Gdańsk University of Technology, Faculty of Ocean Engineering and Ship Technology. His scientific research focuses mostly on application of various AI methods and heuristics to solving navigational problems, mostly in ship collision avoidance. In the years 2009–2011 he was the head of the research project for the Polish Ministry of Science and Higher Education entitled *Evolutionary sets of safe ship trajectories in solving multiple ship collision situations*.

Joanna Szłapczyńska, M.Sc.Eng. in computer science, holds a Ph.D. in technical sciences. Currently, she works as an assistant professor at Gdynia Maritime University, Faculty of Navigation. Her scientific research has been focused on multicriteria optimisation of ship routes (weather routing) and application of AI methods to various navigational problems.

Received: 8 December 2011
Revised: 28 May 2012