A random sampling based algorithm for ship path planning with obstacles

R. Zaccone* & M. Martelli*

** Dept. of Electrical, Electronic, Telecommunications, Naval Architecture and Marine Engineering (DITEN), Polytechnic School of Genoa University, Genova, Italy.*
* Corresponding Author. Email: michele.martelli@unige.it

Synopsis

The paper presents a path planning algorithm for ship guidance in presence of obstacles, based on an *ad hoc* modified version of the Rapidly-exploring Random Tree (RRT*) algorithm. The proposed approach is designed to be part of a decision support system for the bridge operators, in order to enhance traditional navigation. Focusing on the maritime field, a review of the scientific literature dealing with motion planning is presented, showing potential benefits and weaknesses of the different approaches. Among the several methods, details on RRT and RRT* algorithms are given. The ship path planning problem is introduced and discussed, formulating suitable cost functions and taking into account both topological and kinematic constraints. Eventually, an existing time domain ship simulator is used to test the effectiveness of the proposed algorithm over a number of realistic operation scenarios. The obtained results are presented and critically discussed.

Keywords: Collision Avoidance, Ship Simulator, Path Planning, RRT*

1.Introduction

Nowadays, major attention in crew training and certification program, onboard sensors, emergency procedures are present to reduce to risk of accidents at sea. However, as reported by the European Maritime Safety Agency (EMSA) annual report, the total number of marine accidents is continuously increasing, as shown in Figure 1. This can partially be charged to the increase of both the number and the vessel sizes. New ship designs should take into account design criteria not yet covered in past, and the designer should exploit the help available nowadays, coming from the progress in information technologies. Figure 2 highlights that the severe normative framework in structure and stability that has been issued in the last decades has almost nullified the hull failure and the capsizing casualties.

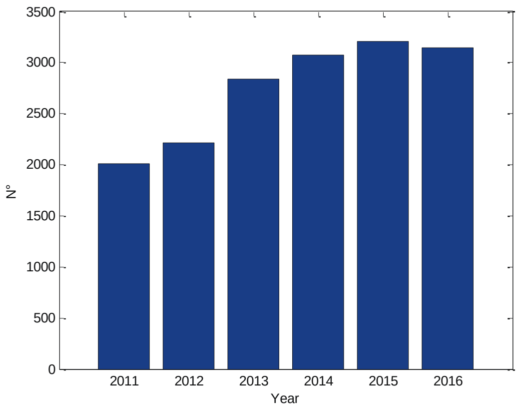

Figure 1: Marine incidents 2011-2016.

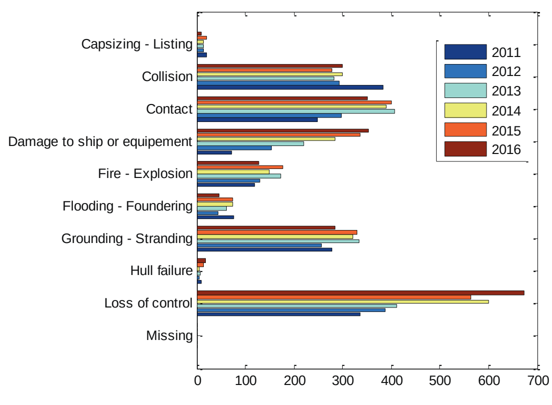

Figure 2: Detail of casualties.

This paper aims to propose a methodology to improve the ship safety, preventing grounding and collision which still represent the 31% of the total sea accidents involving ships. The standards to prevent collisions are clearly reported in COLREGS, however, the effectiveness of the escape manoeuvre is affected by two important factors: environmental and human. Poor visibility (fog or darkness), rough weather, joint with tiredness, stress, carelessness of the bridge operators can result in severe consequences. The main idea is indeed to use the computational power available in every ICT device to help the bridge operators taking a decision in case of potentially dangerous situations, without taking any action, yet acting as a virtual bridge assistant that suggests

http://doi.org/10.24868/issn.2631-8741.2018.018

the correct manoeuvre. Of course, the collision avoidance is not a new topic in the marine scientific community, the work by Skjong, & Mjelde (1982) presents an interesting differential formulation of the collision avoidance problem, together with an algorithm to compute the solution by numeric integration. Most of the marine applications are based on classic heuristic algorithms, such as genetic algorithms (GA) (Ito et al., 1999; Smierzchalski & Michalewicz, 2000; Alvarez et al., 2004). Hasegawa et al. (2012) present a different approach based on potential field methods, that are widely used in robotics. Fang et al. (2017) present an iterative methodology to set up the minimum rudder angle to avoid a collision. Montewka et al., (2009) suggest a new criterion for collision avoidance based on the minimum distance to the collision. The autonomous navigation, or better, the real-time autonomous path planning, is a research topic in several areas that are often a step forward compared with the marine field: aerospace, automotive, robotics, ground unmanned vehicles, video-games. An analysis of the scientific papers coming from different research fields can give a wider overview of the current state of the art of motion planning. It turns out, for example, that evolutionary algorithms are not so widely used in the recent applications. Other proposed methods are dynamic programming (Barraquand & Ferbach, 1994; Shin & McKay, 1986) and mixed integer linear programming (Richards & How, 2002).

Potential field methods are also widely used for various unmanned vehicles (Barraquand et al., 1992; Hwang & Ahuja, 1992; Shimoda et al., 2005). The potential field methods are based on the idea of associating a potential with the elements of the scenario: for example, sources are associated with the obstacles, while the target is modelled by a sink. The trajectory is found by following the minimum potential path. The most recent works, finally, are based on random sampling algorithms, which conjugate a relative simplicity and flexibility with great computational performances. The random sampling philosophy inspired a huge number of different algorithms: Karman & Frazzoli (2010) present a substantial and exhaustive analysis of this type of algorithms, and state that the introduction of this type of algorithms has been the most significant step forward in the development of motion planning algorithms for autonomous objects. Later on, the authors describe the rapidly-exploring random tree (RRT) algorithm, and its optimizing version RRT*, and compare them in detail.

RRT algorithm has been introduced by LaValle & Kuffner (2000), who analyze its performances on various applications, for instance, vehicles, robotic manipulators, motion planning of cumbersome objects in restricted spaces, space aircraft, and virtual human characters. LaValle (2003) compares RRT to dynamic programming in planning feasible (non-optimal) trajectories. The heuristic variant RRT* is presented by Karaman et al. (2011), and applied to an autonomous forklift for cargo handling.

The advantages and the effectiveness of these algorithms claimed in the previously mentioned publications, pushed the authors to try to develop a new "ad hoc" path planning methodology. Albeit in the scientific literature several procedures are present, the reason that leads to developing a new methodology is due to the fact that a ship is a rigid body, working in a six degree of freedom environment a free surface between two fluids, with limited manoeuvring capability and a mass hundreds of time higher than other vehicles', that results in a peculiar dynamic behaviour. All these factors make the ship a unique and particular vehicle, which needs to dedicate studies.

2. RRT and RRT* algorithm

In this section, a brief overview of the RRT algorithm will is going to be reported. The RRT algorithm (LaValle & Kuffner, 2000; Karman & Frazzoli 2010; 2011) aims to explore the state space by generating feasible trajectories subject to constraints, which can be in principle can be holonomic or non-holonomic.

For example, consider a particle moving in a two-dimensional space with obstacles, i.e. forbidden regions: the RRT generates a tree of paths that explore the space trying to reach all the feasible configurations from a starting point.

The schematic representation of RRT and RRT* algorithm flows are reported in Figure 5, where the differences between the two variants branch from a common stream in red and blue respectively. The algorithm builds a tree G that is composed by a set of nodes V and a set of edges E, growing it for a fixed number of iterations. First, the algorithm generates a new random node x_{rand} in the domain, then finds the node $x_{nearest} \in G$ that is nearest to x_{rand}.

A kinematic model generates a new node x_{new} from x_{rand} and $x_{nearest}$, in accordance with the constraints, moving by a fixed $step$. In the RRT case, the edge linking $x_{nearest}$ and x_{new} is checked to be free from collisions: if it is, both the node and the edge are added to the tree.

Note that the RRT algorithm does not perform any optimization and does not compute any cost or return function, i.e. it does not provide any ranking criterion for the solutions. It only generates trajectories compliant with the constraints, which can in principle be winding and illogic. This allows the RRT to run incredibly fast: in fact, it is useful when feasibility and speed of computation are the only two main issues. Note also that an

 http://doi.org/10.24868/issn.2631-8741.2018.018

increase in the number of iterations will result only in a tree with more nodes and edge, but not in “better” trajectories.

However, most of the times, an optimization criterion can be set, and this is when the heuristic variant of the RRT, named RRT*, comes in handy. The RRT* is based on the RRT algorithm, and branches from it just after having added the node x_{new} , adding some additional passages in order to heuristically drive the tree growth to minimize a cost function. In particular, two separate actions, both aiming to locally drive the growth through optimality, take place:

1. The nodes that are within a neighbourhood of the assigned radius are selected, and the parent of x_{new} is selected as the one that minimizes the cost function in x_{new}, in compliance with the constraints.
2. The actual cost in each neighbour node is compared to the one achievable by reaching it from x_{new}: in case this last is better, x_{new} is set as the new parent for the neighbour node. (this action is usually called rewiring)

The selection of a proper cost function allows controlling the trajectories generated by the algorithm. Note that, unlike in the RRT, an increase in the number of iterations in RRT* leads to better solutions.

3. Marine application

As previously mentioned, the RRT* algorithm was developed and used for different vehicle types, completely different from ships in terms of dynamics and degrees of freedom. The purpose of a marine automatic collision avoidance system is to detect the obstacles and, in case a collision is possible, plan and actuate the evasive manoeuvre. Thus, a detection system, able to identify and provide information about the presence of obstacles in the environment is needed. In case a collision is possible, a motion planning system is asked to generate an evasive manoeuvre, and to provide a set of waypoints to safely guide the ship through the target. The diagram presented in Figure 3 schematizes the described concepts.

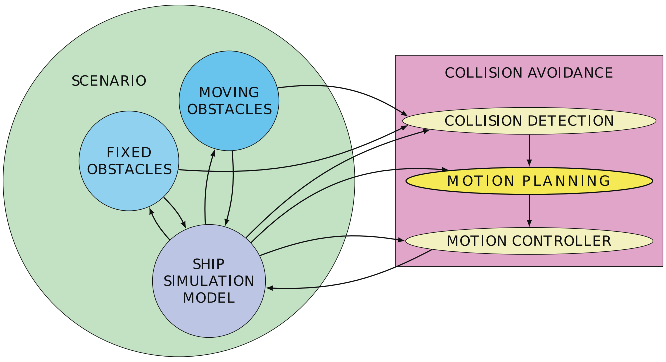

Figure 3: Proposed simulation framework.

The detection aspects are not in the aim of this study: a simple geometrical approach can be used under the hypothesis that both the bathymetry of the navigation area and the data of the surrounding ships coming from AIS are available. A possible collision is detected if the minimum distance between the ship and the obstacles lowers under a threshold value, under which the risk of collision is too high.

In Figure 4 an example of the detection module is reported: the module, after receiving the data coming from the sensors, first evaluates the bearing of the target, and then evaluates the Closest Point of Approach (CPA), the time needed to reach the CPA, and the relative distance at CPA, and provides the computed information in a message dialog box. In the figure, the green points represent the positions of the two ships when the distance equals the safety threshold value (green circle), while the red points identify the two ships when the relative distance is minimum: in the presented case, the minimum distance is expected to be zero, i.e. a real collision is going to occur.

http://doi.org/10.24868/issn.2631-8741.2018.018

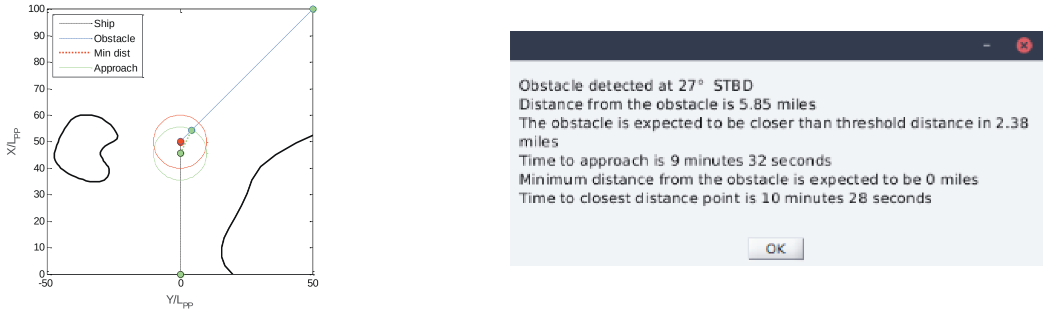

Figure 4: Detection module output.

In case a possible collision is detected, an evasive manoeuvre should be planned, and it is represented in the following form:

$$S = \{\underline{x}_k\}, \qquad k = 0, \dots, N \tag{1}$$

Where: $\underline{x}_k = (X_k, Y_k)$ is the k^th waypoint, and $\underline{x}_0$, $\underline{x}_N$ represent the start and the target point, respectively.

The computed path should be as smooth and regular as possible, in order to allow the control system to successfully follow the waypoints, and should drive the ship back to its original route when there are no more hazards. In particular, the motion planning system needs to take into account the manoeuvring performance of the ship, in order to provide a feasible evasive path. In addition, some optimization logic is needed in order to provide "good" avoidance manoeuvres. In this study, a kinematic model, acting through a set of constraints, tackle the first issue, while the second is managed through a proper definition of the cost function. In addition, the ship speed is supposed constant during the manoeuvre.

3.1. Ship Simulator

The simulator used in the present work to realistically set the problem constraints is a multi-domain platform, able to represent the dynamics of a twin screw ship in three degrees of freedom, taking into account the complete propulsion system with automation effects.

The ship simulator consists of a set of differential equations, algebraic equations, and tables that represent the various elements of propulsion and control systems, ship manoeuvrability and the mutual interactions among them. From the mathematical point of view, the problem can be summarized by the following set of differential equations:

$$M\dot{\nu} + C(\nu)\nu = \tau_H + \tau_P + \tau_R \tag{2}$$

$$2\pi I \frac{dn(t)}{dt} = Q_{eng}(t) + Q_{fric}(t) + Q_P(t) \tag{3}$$

$$S_i(t) = K_{P\_i}\, e_i(t) + K_{I_i}\int_0^t e_i(t)dt + + K_{D\_i}\frac{d}{dt}e_i(t) \tag{4}$$

For the sake of compactness, the equation of ship motions (2)is here expressed in vectorial form as, for instance, in Fossen (2002). The propulsion plant dynamics is described through the differential equation of the shaft line (3); solving this equation over the time domain for each shaft allows to obtain the propulsion plant behaviour in terms of shaft line revolution regime, $n(t)$. Shaft line acceleration depends on the engine, friction and propeller torques, Q_{eng}, $Q_{fric}$and Q_p, respectively, and on the total polar inertia I of the drive line. In the case of a twin screw ship, where the two shaft lines can be used independently from each other or locked together, this is a crucial modelling aspect: in fact, in the case of tight manoeuvres, strong asymmetries in terms of shaft loads can be experienced.

The local engine governor control system is represented by a set of equations, whose form in many cases describes a PID controller, like the one presented in Equation (4), describing.

Detailed information about each sub-module of the simulation platform is reported in (Alessandri et al., 2015). Moreover, the validation of the simulator model, based on a dedicate sea trials campaign, is presented in Donnarumma et al. (2017).

http://doi.org/10.24868/issn.2631-8741.2018.018

Such a detailed platform is needed both to evaluate the constraints and to check if the outcomes of the motion planning can be really actuated by the ship, as described in the next section.

3.2. Mathematical modelling of the problem constraints

The distance from the obstacles during the navigation is required to be greater than a threshold value:

$$d_{min} - \min_i d\big(\underline{x}(t), O_i(t)\big) < 0, \quad t \in [t_i, t_f] \tag{5}$$

Where d_{min} is the threshold, set on the basis of the common practice, $\underline{x}(t)$ is the ship position at time t along the path, O_i is the position of the i^{th} obstacle (which can be function a of the time), and t_i, t_f are the initial and final instants respectively.

Note that the threshold value should also take into account the uncertainties coming from the obstacle localization system, as well as the possible overshoots from the waypoints due to the control system.

Two more constraints take into account the turning capabilities of the ship: the maximum course variation between two consecutive segments and the curvature of the circumference connecting the midpoints of the segments are constrained to be less than their respective thresholds.

Let $l_k = d\big(\underline{x}_k, \underline{x}_{k+1}\big)$ be the length of the k^{th} segment and χ_k its orientation. The constraints take the form:

$$|\chi_k - \chi_{k+1}| - \Delta\chi_{max} < 0, \qquad k = 0, \dots, N \tag{6}$$

$$\frac{2\,tan(|\chi_k - \chi_{k+1}|)}{min(l_k, l_{k+1})} - \rho_{max} < 0, \qquad k = 0, \dots, N-1 \tag{7}$$

Where $\Delta\chi_{max}$ and ρ_{max} are the threshold values of the course variation and curvature respectively. Both those constraints are evaluated by a dynamic simulation model capable to take into account the real ship behavior, the system non-linearity, and the mutual interaction among all the involved elements, as reported in the previous section.

3.3. Cost function

The definition of the cost function is the key-point when using an RRT* algorithm, due to its optimizing action: the cost function drives the growth of the tree, thus controls the shape of the solution. Mimicking the possible bridge operator behaviour, the desirable features of an evasive manoeuvre can be summarized as short path elongation, a limited number of manoeuvring actions and as much distance as possible from the obstacles. These features can be conflicting in most of the scenarios, in these cases, a trade-off solution should be chosen, to this end, the proposed cost functions assume the following form.

The first, c_l is related to the path length, and penalizes the path elongation:

$$c_l(\underline{x}_p) = \sum_{k=0}^{p-1} d\,(\underline{x}_{k-1}, \underline{x}_k) \tag{8}$$

The second is related to the curvature: it has the twofold benefit of keeping the path straight and with smooth turns, and distributing the waypoints at regular distances:

$$c_\rho\big(\underline{x}_p\big) = \max_{k \in \{0,\dots,p\}} \rho_k + \frac{1}{p}\sum_{k=0}^{p} \rho_k \qquad 9)$$

Where:

$$\rho_k = \frac{2\,tan(|\chi_k - \chi_{k+1}|)}{min(l_k, l_{k+1})} \qquad 10)$$

The third aims to model a repulsive effect of the obstacles in analogy with potential field methods, in order to maximize the margin from the obstacles:

$$c_r(\underline{x}_p) = \sum_{k=0}^{p} \frac{1}{\min_i d\big(\underline{x}_k, O_i(t_k)\big)} \qquad 11)$$

http://doi.org/10.24868/issn.2631-8741.2018.018

The proposed cost functions can produce significantly different tree topologies and path shapes in function of the setting of the weights: Figure 6, Figure 7, and Figure 8 show the effects of the three contributions on the tree growth. The weights must be properly calibrated in order to get the desired results.

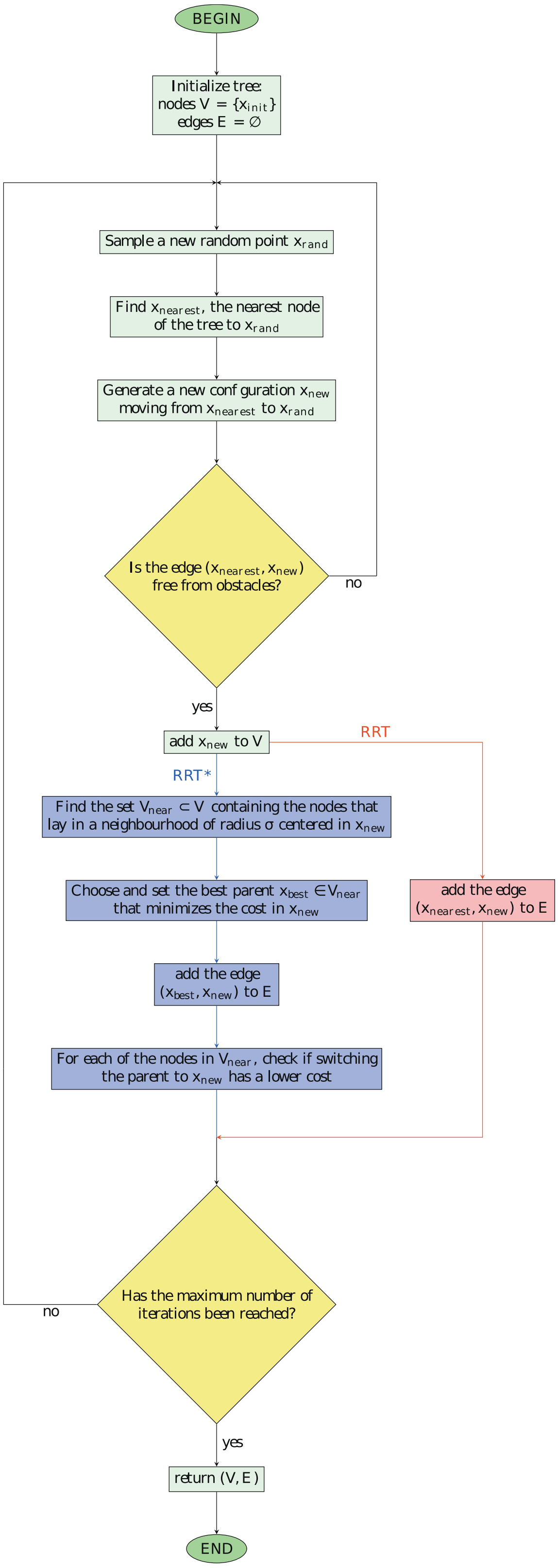

Figure 5: RRT/RRT* Flowchart

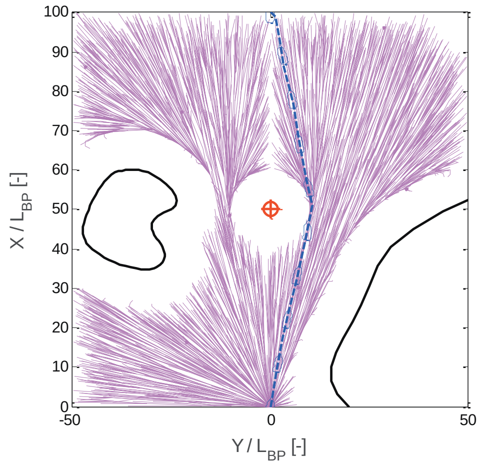

Figure 6: Minimum distance.

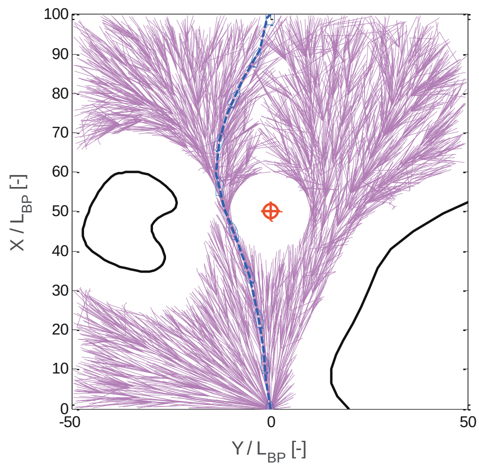

Figure 7: Minimum curvature cost.

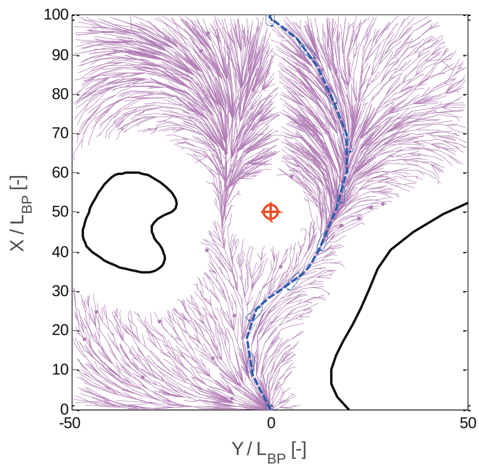

Figure 8: Minimum repulsion potential.

http://doi.org/10.24868/issn.2631-8741.2018.018

4. Results & Conclusion

In order to test the effectiveness of the proposed methodology, a realistic benchmark has been developed and tested. The scenario, presented in Figure 9 is composed of the own ship (blue), two islands (black line) and two moving obstacle ships (red). The extension of the map is 100x100 ship lengths, around five square miles. The starting point is located in [0,0] and the goal is to reach safely the point $[0, 100]$ through a straight course. It is assumed that at $t = 0$, the two moving obstacles (ships) have been identified and the collision avoidance algorithm start to compute the evasive manoeuvre.

The manoeuvre has been represented in nine sequential screenshots (one every two minutes and a half). During the navigation, the detection system advice at the initial time that two collision events are possible. The automatic path planning system assesses an alternative path, generating a set of waypoints identified by light blue circles, that are strictly followed by the track keeping system. In this case, all the three costs function components have been used with suitable weighs. After 22 minutes of simulation (few seconds in CPU time), it is possible to see that the ship reaches the target point, navigating always respecting the safety threshold (dotted green circle).

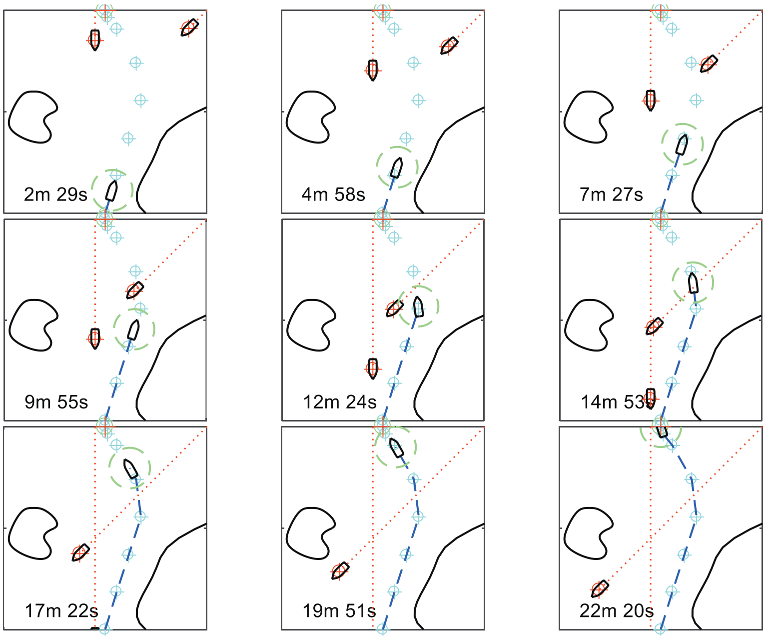

Figure 9: Example of evasive manoeuvre.

The results show that the proposed avoidance methodology works very well in case of complex scenarios with both fixed and moving obstacles. For sake of shortness only one manoeuvre is reported, however, more complex scenarios up to 10 moving obstacles with different speed and course have been simulated, and the system successfully found a suitable solution. In all cases, the algorithm is able to compute the new path in less than 1 second running on an entry-level laptop. Such a computation time is fully compatible with the relative slow ship dynamics, and with the distance, in terms of nautical miles, in which the others ship can be identified in terms of speed and course. The return to the original course is also ensured, with the bridge operator only supervising the process.

5. Acknowledgements

This study is part of the Project POLAR, a Research and Innovation project coordinated by Fincantieri S.p.A. with the participation of the National Research Council (CNR); the project receives grants from the Italian Ministry of Infrastructures and Transport (MIT).

6. References

Alessandri, A., Donnarumma, S., Luria, G., Martelli, M., Vignolo, S., Chiti, R., Sebastiani, L., Dynamic positioning system of a vessel with conventional propulsion configuration: Modeling and simulation, (2015) Maritime Technology and Engineering - Proceedings of MARTECH 2014: 2nd International Conference on Maritime Technology and Engineering, pp. 725-734.

http://doi.org/10.24868/issn.2631-8741.2018.018

Altosole, M., Figari, M., Martelli, M., Time-domain simulation for marine propulsion applications, (2012) Proceedings of the 2012 - Summer Computer Simulation Conference, SCSC 2012, Part of SummerSim 2012 Multiconference, 44 (10), pp. 36-43.

Altosole, M., Martelli, M, Propulsion control strategies for ship emergency manoeuvres, (2017) Ocean Engineering, 137, pp. 99-109.

Alvarez, A., Caiti, A., & Onken, R. (2004). Evolutionary path planning for autonomous underwater vehicles in a variable ocean. IEEE Journal of Oceanic Engineering, 29(2), 418-429.

Barraquand, J., & Ferbach, P. (1994). Path planning through variational dynamic programming. In Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on (pp. 1839-1846). IEEE.

Barraquand, J., Langlois, B., & Latombe, J. C. (1992). Numerical potential field techniques for robot path planning. IEEE Transactions on Systems, Man, and Cybernetics, 22(2), 224-241.

COLREGS - International Regulations for Preventing Collisions at Sea 1972

Donnarumma, S., Figari, M., Martelli, M., Vignolo, S., Viviani, M., Design and Validation of Dynamic Positioning for Marine Systems: A Case Study, (2017) IEEE Journal of Oceanic Engineering,

EMSA, 2017 Annual overview of marine casualties and incidents 2017

Fang, M.-C., Tsai, K.-Y., Fang, C.-C. A Simplified Simulation Model of Ship Navigation for Safety and Collision Avoidance in Heavy Traffic Areas, (2017) Journal of Navigation, pp. 1-24.

Fossen TI. Handbook of marine craft hydrodynamics and motion control. Chichester, UK: John Wiley & Sons Ltd, 2011.

Hasegawa, K., Fukuto, J., Miyake, R., & Yamazaki, M. (2012). An intelligent ship handling simulator with automatic collision avoidance function of target ships. Proc. of INSLC, 17.

Hwang, Y. K., & Ahuja, N. (1992). A potential field approach to path planning. IEEE Transactions on Robotics and Automation, 8(1), 23-32.

Ito, M., Zhnng, F., & Yoshida, N. (1999). Collision avoidance control of the ship with a genetic algorithm. In Control Applications, 1999. Proceedings of the 1999 IEEE International Conference on (Vol. 2, pp. 1791-1796). IEEE.

Karaman, S., & Frazzoli, E. (2010). Incremental sampling-based algorithms for optimal motion planning. Robotics Science and Systems VI, 104.

Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. The international journal of robotics research, 30(7), 846-894.

Karaman, S., Walter, M. R., Perez, A., Frazzoli, E., & Teller, S. (2011, May). Anytime motion planning using the RRT. In Robotics and Automation (ICRA), 2011 IEEE International Conference on (pp. 1478-1483). IEEE.

LaValle, S. M. (2003). From dynamic programming to RRTs: Algorithmic design of feasible trajectories. In Control Problems in Robotics (pp. 19-37). Springer, Berlin, Heidelberg.

LaValle, S. M., & Kuffner Jr, J. J. (2000). Rapidly-exploring random trees: Progress and prospects.

Montewka, J., Goerlandt, F., Kujala, P. Determination of collision criteria and causation factors appropriate to a model for estimating the probability of maritime accidents (2012) Ocean Engineering, 40, pp. 50-61.

Richards, A., & How, J. P. (2002). Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In American Control Conference, 2002. Proceedings of the 2002 (Vol. 3, pp. 1936-1941). IEEE.

Shimoda, S., Kuroda, Y., & Iagnemma, K. (2005). Potential field navigation of high-speed unmanned ground vehicles on uneven terrain. In Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on (pp. 2828-2833). IEEE.

Shin, K., & McKay, N. (1986). A dynamic programming approach to trajectory planning of robotic manipulators. IEEE Transactions on Automatic Control, 31(6), 491-500.

Skjong, R., & Mjelde, K. M. (1982). Optimal Evasive Maneuver for a Ship in an Environment of Fixed Installations and Other Ships.

 http://doi.org/10.24868/issn.2631-8741.2018.018

Smierzchalski, R., & Michalewicz, Z. (2000). Modelling of ship trajectory in collision situations by an evolutionary algorithm. IEEE Transactions on Evolutionary Computation, 4(3), 227-241.

http://doi.org/10.24868/issn.2631-8741.2018.018