Article

Not peer-reviewed version

# A Dynamic Programming Approach to Collision Avoidance of Autonomous Ships

Raphael Zaccone *

Posted Date: 29 March 2024

doi: 10.20944/preprints202403.1831.v1

Keywords: Dynamic Programming; Autonomous Ship; Path Planning; Collision Avoidance; Optimization.

 doi:10.20944/preprints202403.1831.v1



*Article*

# A Dynamic Programming Approach to Collision Avoidance of Autonomous Ships

**Raphael Zaccone**

Department of Naval, Electrical, Electronics and Telecommunications Engineering (DITEN), Polytechnic School, University of Genoa; raphael.zaccone@unige.it

**Abstract:** The advancement of autonomous capabilities in maritime navigation has gained significant attention, with a trajectory moving from decision support systems to full autonomy. This push towards autonomy has led to extensive research focusing on collision avoidance, a critical aspect of safe navigation. Among the various possible approaches, Dynamic Programming is a promising tool for optimizing collision avoidance maneuvers. This paper presents a DP formulation for collision avoidance of autonomous vessels. We set up the problem framework, formulate it as a multi-stage decision process, define cost functions and constraints focusing on the actual requirements a marine maneuver must comply with, and propose a solution algorithm leveraging parallel computing. Additionally, we present a greedy approximation to reduce algorithm complexity. Through case studies, we demonstrate the effectiveness of the proposed approach in navigating complex scenarios, contributing to the future of autonomous maritime navigation. Through case studies, we show the efficacy of our approach in navigating complex scenarios.

**Keywords:** dynamic programming; autonomous ship; path planning; collision avoidance; optimization

## 1. Introduction

The future of navigation points toward increasing the autonomous capabilities of ships [1,2], from decision support systems to fully autonomous navigation. The framework that will guide the future of the maritime world is hinged on the classification released by the International Maritime Organization (IMO) [3], in which four incremental autonomy levels are described for the classification of autonomous ships. The push toward the development of autonomous technologies in the maritime field has created fertile ground for the research community, which has identified numerous scientific gaps in various areas typical of autonomous navigation and marine robotics, such as the development of complex guidance and control algorithms [4–6], collaborative control [7–9], situational awareness [10], path planning, and collision avoidance.

The problem of collision avoidance is to develop algorithms capable of reacting to the presence of obstacles, fixed or moving in the surrounding environment, and generating trajectories or maneuvers capable of avoiding collision, maintaining an adequate safety distance, and, when applicable, complying with the COLREG [11,12], which set the "rules of the road a ship must follow when interacting with other ships. This requirement is essential in scenarios where autonomous systems interact with human-controlled systems [13]. The scientific literature proposed various approaches to collision avoidance of marine vessels, including A* [14,15], Dijkstra's algorithm [16,17], visibility graphs [18], rapidly-exploring random trees [19–21], and various population-based heuristics [22–25].

Dynamic Programming (DP) is an effective approach for solving multi-stage optimization problems. From its introduction by Bellman [26,27], DP has been successfully generalized and formulated to describe path planning problems [28]. The requirements of a collision avoidance system are the ability to effectively and efficiently represent obstacles and the surrounding environment and to determine in real-time, with low computational cost, evasive maneuvers that onboard control systems can actuate. These requirements perfectly match the potential of DP, which, in this framework, offers the possibility of a simple and effective problem description, implementation of constraints, and efficient solution schemes capable of exploiting parallelism. Despite the inherent recursive formulation of DP, very efficient solution strategies based on function memoization or tabulation [29] and leveraging

 doi:10.20944/preprints202403.1831.v1



parallel computation [30–32] have been proposed to optimize the computation efficiency. DP-based algorithms are ideal for scenarios where resources are limited and real-time computation is crucial, such as autonomous vehicles, robotics, and embedded systems, and have found application in a broad range of industrial fields, including railway transportation [33], robotics [34,35], and maritime transport. The maritime literature features various applications of DP, primarily to weather routing problems [36–39], where the objective is to determine the optimal trajectory across a space-time domain to reach the destination subject to various constraints (e.g., ship motions, sea sickness), by effectively navigating through dynamic weather conditions in compliance with various constraints.

In this article, we introduce a DP formulation of the collision avoidance problem of autonomous vessels. In Section 2, we set up the framework to describe the ship collision avoidance problem; In Section 3, we formulate the optimal path planning problem as a multi-stage decision process with a recursive definition based on Bellman's equation; in Section 4, we describe the cost function and constraints based on the actual requirements of a collision avoidance maneuver for marine application, taking into account maneuvering limits and regulations to obtain smooth, collision-free, and COLREG compliant maneuvers. In Section 5, we propose a bottom-up solution scheme for the problem, leveraging parallel computing, while in Section 6, we further reduce the algorithm time complexity by proposing a greedy approximation of the DP problem formulated in the previous sections. Eventually, in Section 7, we test and compare the proposed algorithms against case study navigational scenarios to assess the potential of the proposed approach.

## 2. Collision Avoidance Framework Definition

Despite the Earth's curvature being relevant when planning long routes, the horizon of a maneuver in a collision avoidance context is usually such that we can approximate the ship's state space $X$ with an Euclidean plane. At the beginning of the collision avoidance maneuver, the ship is located in $\mathbf{x}_{start}$. We introduce the orthogonal unit vectors $\mathbf{e}_x$ and $\mathbf{e}_y$ to represent the cardinal axes of $X$: the unit vector $\mathbf{e}_x$ is aligned with the ship's course at the beginning of the collision avoidance maneuver, while $\mathbf{e}_y$ points towards the ship's starboard (right) side, so that $\mathbf{e}_z = \mathbf{e}_x \times \mathbf{e}_y$ points downwards, as per the standard convention for ship maneuverability. We will refer to any generic position $\mathbf{x} \in S$ as a waypoint. We can represent a generic maneuver or route $R$ as a sequence of consecutive waypoints:

$$R = (\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_N) = (\mathbf{x}_i)_{i=0}^{N} \tag{1}$$

A sequence of two consecutive waypoints $s = (\mathbf{x}, \mathbf{y})$ is called route leg. If $s_i = (\mathbf{x}_{i-1}, \mathbf{x}_i)$, and we denote with "$\oplus$" the sequence concatenation operator such that $(\mathbf{x}, \mathbf{y}) \oplus (\mathbf{y}, \mathbf{z}) = (\mathbf{x}, \mathbf{y}, \mathbf{z})$, We can represent the route $R$ as:

$$R = (\mathbf{x}_0, \mathbf{x}_1) \oplus (\mathbf{x}_1, \mathbf{x}_2) \oplus \dots \oplus (\mathbf{x}_{N-1}, \mathbf{x}_N) = s_1 \oplus s_2 \oplus \dots \oplus s_N \tag{2}$$

We introduce the notation $\vec{s} = \mathbf{y} - \mathbf{x}$ to represent the vector connecting the start point of the leg to the endpoint. We can obtain the course change between two consecutive legs $s_i \in S_i$, $s_j \in S_j$, $S_i, S_j \subseteq \{(\mathbf{x}, \mathbf{y}) | \mathbf{x}, \mathbf{y} \in X\}$, using the function $\theta : S_i \times S_j \to [0, \pi]$ defined as follows:

$$\theta(s_i, s_j) = \arccos\left(\frac{\vec{s_i} \cdot \vec{s_j}}{|\vec{s_i}||\vec{s_j}|}\right) \tag{3}$$

Some considerations about the ship's kinematics are also needed. Firstly, we assume the ship is moving at constant speed $u$. Such an assumption is reasonable since, in maritime practice, course alterations are preferred to speed reductions due to the long transients a speed alteration takes. In particular, it is common practice to avoid obstacles by altering the ship's course and trying to keep the speed, or reducing the speed by a limited amount in the first part of the maneuver and then keeping it constant, to and slowly regaining the cruise speed after the collision risk is mitigated. Leveraging the constant speed assumption, we can easily map $R$ to a sequence of time instants $T = (t_i)_{i=0}^{N}$. The instant $t_i$ at which it engages the waypoint $\mathbf{x}_i$ is given by:

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 March 2024 doi:10.20944/preprints202403.1831.v1



$$\begin{cases} t_i = t_{i-1} + \dfrac{|\vec{s_i}|}{u} \\ t_0 = 0 \end{cases} \tag{4}$$

Therefore, the instantaneous vessel position $\mathbf{x}(t)$ at time $t = t_i + \Delta t$ is given by:

$$\mathbf{x}(t_i + \Delta t) = \mathbf{x}_i + u\frac{\vec{s_i}}{|\vec{s_i}|}\Delta t \tag{5}$$

Moreover, since collision avoidance is about the interaction with obstacles, we need to introduce a notation and some hypotheses to describe them. First, we assume there are $T$ obstacles in the scenario with known kinematics. We denote with $\mathbf{a}_m(t) \in S$ the instantaneous position of the $m^{th}$ obstacle. For this study, we will approximate the motion of the obstacles as a straight line, constant speed motion:

$$\mathbf{a}_m(t) = \mathbf{a}_m(t_0) + \mathbf{w}_m t \tag{6}$$

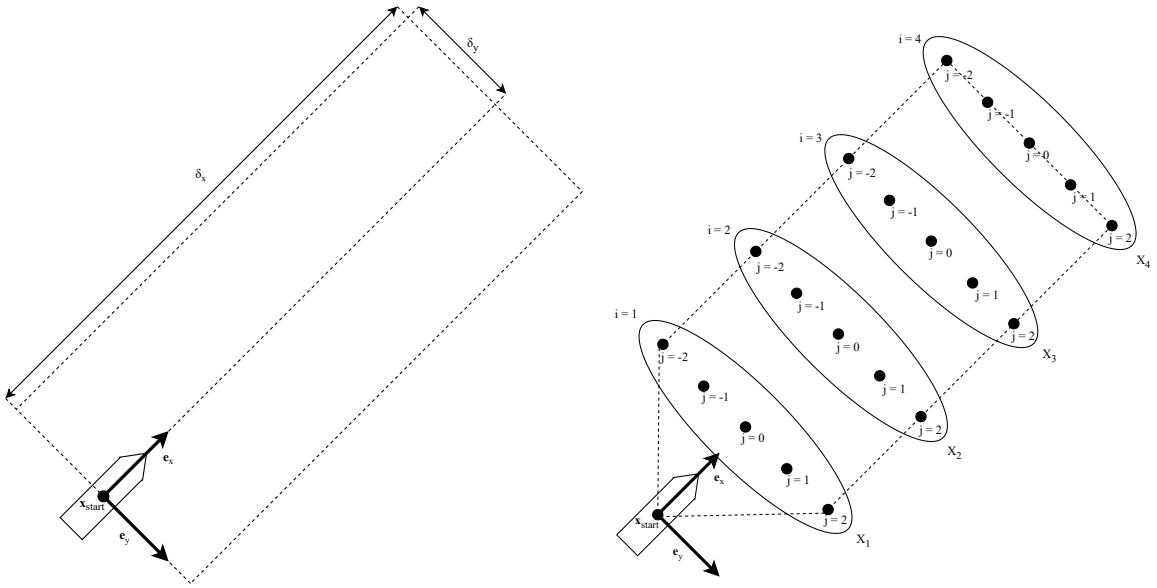
Where $\mathbf{w}_m$ represents the speed vector of the $m^{th}$ obstacle. The proposed formulation could be generalized to any known motion law.

## 3. Dynamic Programming Formulation

Concerning Figure 1a, let $\mathbf{x}_{start} \in S$ be the initial position of the ship, the unit vector $\mathbf{e}_x$ aligned with the ship's initial course, and $\mathbf{e}_y$ a unit vector orthogonal to $\mathbf{e}_x$ and such that $\mathbf{e}_z = \mathbf{e}_x \times \mathbf{e}_y$ points downwards. Let $\delta_x$ and $\delta_y$ be the dimensions of the region in which the ship can maneuver. We introduce a discretization of the domain by defining $X_0 = \{\mathbf{x}_{start}\}$, and $X_i \subset X, i \in \{1, ..., N\}$ as follows:

$$X_i = \left\{ \mathbf{x} \in X | \mathbf{x} = \mathbf{x}_{start} + \frac{i}{N}\delta_x \mathbf{e}_x + \frac{j}{D}\delta_y \mathbf{e}_y,\ j \in \{-D, \dots D\} \right\} \tag{7}$$

Where $N, D \in \mathbb{N}$, control the fineness of the discretization along $\mathbf{e}_x$ and $\mathbf{e}_y$ respectively, and $i, j$ are the indices of the discretization. Figure 1b exemplifies a graphical representation. A route $R = (x_i)_{i=0}^{N}$ is such that $x_i \in X_i$.




**(a)** Domain region. **(b)** Discretized domain region.

**Figure 1.** A representation of the domain of the collision avoidance problem (1a), and an example discretization of the domain, with $N = 4$ and $D = 2$1b

 doi:10.20944/preprints202403.1831.v1



We can imagine a route $R = s_1 \oplus s_2 \oplus ... \oplus s_N$ as the result of a sequence of transitions between consecutive route legs $s_i = (\mathbf{x}_{i-1}, \mathbf{x}_i) \in S_i$, where $S_i = \{s = (\mathbf{x}, \mathbf{y}) | \mathbf{x} \in X_{i-1}, \mathbf{y} \in X_i\}$. In this framework, optimal route planning can be represented as finding an optimal policy in a multi-stage decision process in which the route legs represent the states. Let $J : \prod_{i=1}^{N} S_i \to \mathbb{R}$ be an additively separable function expressing the cost of a route, i.e., $J$ is such that there exists a function $c : S_{i-1} \times S_i \to \mathbb{R}$ such that:

$$J(s_1, s_2, ..., s_i) = J(s_1, s_2, ..., s_{i-1}) + c(s_{i-1}, s_i) \tag{8}$$

Let us denote $F(s_i) : S_i \to \mathbb{R}$ be the route's cost up to the state $s_i$ following an optimal policy. We can leverage the Bellman's equation to define $F$ recursively:

$$\begin{cases} F(s_i) = \min\limits_{z \in S_{i-1} | t(z, s_i)} c(z, s_i) + F(z) \\ F(s_1) = c(s_0, s_1) \end{cases} \tag{9}$$

Where $s_0$ represents the state of the ship before the collision avoidance maneuver begins, and $t : S_{i-1} \times S_i \to \{T, F\}$ is a function such that $t(s_{i-1}, s_i) = T$ if the transition from $s_{i-1}$ to $s_i$ is feasible, $F$ otherwise.

We can analogously define the optimal predecessor function, $p : S_i \to S_{i-1}$, returning the optimal predecessor state in $S_{i-1}$ for the state $s_i \in S_i$:

$$p(s_i) = \operatorname*{argmin}_{z \in S_{i-1} | t(z, s_i)} c(z, s_i) + F(z) \tag{10}$$

Let the recursive function $B : S_i \to \{(\mathbf{x}_k)_{k=0}^{i} | x_k \in S_k\}$ have the following form:

$$\begin{cases} B(s_i) = B(p(s_i)) \oplus s_i \\ B(s_1) = s_1 \end{cases} \tag{11}$$

Once we know $p(\mathbf{x}_i)$ for all $i \in \{1, ..., N\}$, the optimal solution $R$ can be obtained backtracking the optimal decisions:

$$R = B(s_N) \tag{12}$$

where:

$$s_N = \operatorname*{argmin}_{s \in S_N} F(s) \tag{13}$$

## 4. Constraints and Cost Function

The constraints are needed to formulate the function $t : S_{i-1} \times S_i \to \{T, F\}$, identifying whether a transition is feasible. For the transition to be feasible, the next leg must begin at the end of the previous. Moreover, each transition must be such that the final result, i.e., the route, is compatible with the ship's maneuvering capabilities, respects the COLREG, and is collision-safe. Therefore, the function $t$ takes the following form:

$$t(s_{i-1}, s_i) = t_{link}(s_{i-1}, s_i) \wedge t_{\theta}(s_{i-1}, s_i) \wedge t_{COLREG}(s_i) \wedge t_{collision}(s_i) \tag{14}$$

In particular, $t_{link}$ ensures the two route legs can be connected; the course change constraint function $t_{\theta}$ ensures the course changes along the path are compatible with ship maneuvering capabilities and good seamanship, the COLREG constraint function $t_{COLREG}$ implements a set of rules to ensure the COLREG compliance of the own ship's kinematics relative the other vessels', and the collision avoidance constraint function $t_{collision}$ aims to guarantee an appropriate distance from the obstacles to avoid collisions safely.

 doi:10.20944/preprints202403.1831.v1



### 4.1. Route Leg Connection

The feasibility of the transition from $s_{i-1} \in S_{i-1}$ to $s_i \in S_i$ is expressed by the function $t_{link} : S_{i-1} \times S_i \to \{T, F\}$. The transition is possible if the end point of the first leg is the start point of the second:

$$t_{link}(s_{i-1} = (\cdot, \mathbf{x}), s_i = (\mathbf{y}, \cdot)) = (\mathbf{x} \stackrel{?}{=} \mathbf{y}) \tag{15}$$

Where the symbol "$\cdot$" is a placeholder for any point in the proper domain, and the operator "$\stackrel{?}{=}$" returning values in $\{T, F\}$ is such that $A \stackrel{?}{=} B$ returns $T$ if $A = B$, $F$ otherwise.

### 4.2. Course Change Constraint

We need to define a minimum and maximum threshold value for the ship's course changes, denoted as $\theta_{\min}$ and $\theta_{\max}$, respectively. The introduction of $\theta_{\max}$ is motivated by the ship's maneuvering capabilities and the performance of the motion control system: during a course change, the ship goes off the predefined track by a certain distance which is related to the magnitude of the course change. A large course change will push the ship too much off track, increasing the risk of collision. The reason to $\theta_{\min}$ can be found in COLREG Rule 8(b), which requires that "*Any alteration in course and/or speed to avoid collision must, if the circumstances of the case permit, be large enough to be readily apparent to another vessel observing visually or by radar; a succession of small alterations in course and/or speed must be avoided.*". We thus want to ensure that we either have course changes large enough to be apparent to an observer but not as large that the control and steering system cannot actuate them or no course change between legs. Thus, we can define $t_\theta : S_{i-1} \times S_i \to \{T, F\}$:

$$t_\theta(s_{i-1}, s_i) = (\theta_{\min} \leq \theta(s_{i-1}, s_i) \leq \theta_{\max}) \vee \left(\theta(s_{i-1}, s_i) \stackrel{?}{=} 0\right) \tag{16}$$

### 4.3. COLREG Compliance Constraint

COLREG is an international convention regulating several aspects of navigation, including visibility, navigation lights, and the behaviors that each ship must have in encounter situations. This last part is a set of "rules of the road" each ship has to follow when encountering other ships. The COLREG-compliant behavior can be identified by analyzing the kinematics of the ships engaging the scenario and relying on a set of if-then-else rules to assess which behavior each ship has to keep relative to the others. This action is usually called COLREG classification. Various approaches and algorithms for COLREG classification have been proposed in literature [40,41].

For this study, we will suppose we are able to determine the COLREG-compliant behavior $\beta_i$ the own ship must keep relative to the $i^{th}$ dynamic obstacle in the scenario, and that $\beta_i \in CB$, where $CB$ is the enumeration of all the possible behaviors:

$$CB = \{SO, GW, HO, AA\} \tag{17}$$

In particular, we describe the possible behaviors hereinafter:

- *SO* (Stand On): COLREG requires the target ship to maneuver to avoid a collision, so the own ship must keep its course and speed: in this case, we will neglect the presence of the target ship;
- *GW* (Give Way): The own ship must maneuver to avoid collision with the target ship, letting the latter pass ahead;
- *HO* (Head On): The own ship and the target ship are sailing on parallel and opposite routes, and the own ship must avoid the collision by turning to starboard;
- *AA* (Any Action): The own ship must take any appropriate action to avoid collision; this behavior is adopted in emergencies, such as when the target ship is expected to maneuver to avoid the collision but does not seem to initiate the evasive maneuver.

These behaviors are ensured in the solution by imposing constraints on the leg in which the own ship and the target ship cross each other's route.

 doi:10.20944/preprints202403.1831.v1



Let us assume that the ship and the target **a** cross in leg $s_i = (\mathbf{x}_{i-1}, \mathbf{x}_i)$, i.e., for $t \in [t_{i-1}, t_i]$, and let us denote with $\mathbf{x}_c$ the intersection point. If $\beta = GW$, COLREG requires the target ship to engage $\mathbf{x}_c$ before the own ship; thus, we need to impose that if the own ship engages $\mathbf{x}_c$ during leg $s_i$, it does after the target has:

$$\begin{aligned} &t_{GW}(s_i = (\mathbf{x}_{i-1}, \mathbf{x}_i)) = \\ &\quad = (\mathbf{x}_c \in \{\mathbf{x}(t) \in S | t \in [t_{i-1}, t_i]\}) \wedge \left( \frac{|\mathbf{x}_c - \mathbf{x}_{i-1}|}{u} - \frac{|\mathbf{x}_c - \mathbf{a}(t_{i-1})|}{|\mathbf{w}|} > 0 \right) \end{aligned} \tag{18}$$

If $\beta = HO$, the correct behavior is ensured by imposing that the computed path keeps the target ship on the port side of the own ship. The latter is ensured by the constraint below:

$$t_{HO}(s_i) = (\forall t \in [t_{i-1}, t_i] : \ (\mathbf{a}_m(t) - \mathbf{x}(t)) \times \vec{s_i} \cdot \mathbf{e}_z > 0) \vee \tag{19}$$

Eventually, the COLREG constraint function takes the following form:

$$\begin{aligned} t_{COLREG}(s_i) = &\forall m \in \{1, ..., T\} : \ (\beta_m \stackrel{?}{=} GW \wedge t_{GW}(s_i)) \vee \\ &\vee \ (\beta_m \stackrel{?}{=} HO \wedge t_{HO}(s_i)) \vee \\ &\vee \ (\beta_m \in \{AA, SO\}) \end{aligned} \tag{20}$$

### *4.4. Collision Avoidance Constraint*

The obstacle avoidance constraint aims to guarantee that the evasive maneuver is collision-safe. We introduce the concepts of safety distance $d_{safety}$ and the Closest Point of Approach, $CPA$. The $d_{safety}$ represents the distance the own ship must keep from all obstacles during an evasive maneuver to ensure safety with enough margin to account for ship dimensions and uncertainties related to the environment, control, and measurement systems. The $CPA$ is the minimum distance between the center points of the own ship $\mathbf{x}(t)$ and an obstacle, whose position over time is denoted by $\mathbf{a}_m(t)$, for $t \in [t_0, t_N]$. In particular, we define the $CPA_m(s_i) : S_i \to \mathbb{R}$ as follows:

$$CPA_m(s_i) = \min_{t \in [t_i - 1, t_i]} |\mathbf{a}_m(t) - \mathbf{x}(t)| \tag{21}$$

Thus, if there are $T \in \mathbb{N}$ obstacles, the function $t_{collision} : S_i \to \{T, F\}$ checks whether a transition is collision-safe:

$$t_{collision}(s_i) = \forall m \in \{1, \ldots, T\} \mid \beta_m \neq SO : \ CPA_m(s_i) \geq d_{safety} \tag{22}$$

### *4.5. Cost Function*

The definition of the cost function plays an essential role because it directly influences the geometric characteristics of the optimal path. Various approaches can be used; the proposed approach seeks the minimum control energy maneuver. In the context of this paper, control energy is related to the amplitude of the maneuvers needed to follow a path. In other words, a path with high control energy requires the ship to perform large course changes over time. To this end, we can define the transition cost $c$ as follows:

$$c(s_{i-1}, s_i) = \theta^2(s_{i-1}, s_i) \tag{23}$$

## 5. Solution Scheme

This section presents a bottom-up solution scheme for the problem proposed in Section 3 based on a tabulation approach. We can divide the algorithm into two phases: the tabulation phase and the backtracking phase.

Algorithm 1 illustrates the tabulation phase. We sequentially generate the feasible states at each stage by leveraging Equation 7, while keeping track of the values of the partial optimum $F$ and the

 doi:10.20944/preprints202403.1831.v1



backward link to the optimal predecessor $p$ into proper data structures. If $c$ and $t$ run in constant time, the time complexity of Algorithm 1 is $O(ND^3)$. Notice that the inner for-loop at line 6 of Algorithm 1 can be run in parallel since the loop does not mutate shared data.

**Algorithm 1:** Bottom-up solution scheme

1 $S_1 = \emptyset$
2 **for** $j \in [-D, ..., D]$ **:**
3 $\quad s_j = (\mathbf{x}_{start}, \mathbf{x}_{start} + \frac{1}{N}\delta_x \mathbf{e}_x + \frac{j}{D}\delta_y \mathbf{e}_y)$
4 $\quad S_1 = S_1 \cup \{s_j\}$
5 $\quad F(s_j) = c((\mathbf{x}_{start} - \mathbf{e}_x, \mathbf{x}_{start}), s_j)$
6 $\quad p(s_j) = None$
7
8 **for** $i \in [2, \ldots, N]$ **:**
9 $\quad S_i = \emptyset$
10 $\quad$ **parallel for** $j \in [-D, \ldots, D]$ **:**
11 $\quad\quad \mathbf{x}_j = \mathbf{x}_{start} + \frac{i}{N}\delta_x \mathbf{e}_x + \frac{j}{D}\delta_y \mathbf{e}_y$
12 $\quad\quad$ **for** $k \in [-D, \ldots, D]$ **:**
13 $\quad\quad\quad \mathbf{x}_k = \mathbf{x}_{start} + \frac{i-1}{N}\delta_x \mathbf{e}_x + \frac{k}{D}\delta_y \mathbf{e}_y$
14 $\quad\quad\quad s_j = (\mathbf{x}_k, \mathbf{x}_j)$
15 $\quad\quad\quad T = \{z \in S_{i-1} | t(z, s_j)\}$
16 $\quad\quad\quad$ **if** $T \neq \emptyset$ **:**
17 $\quad\quad\quad\quad S_i = S_i \cup \{s_j\}$
18 $\quad\quad\quad\quad F(s_j) = \min_{z \in T} c(z, s_i) + F(z)$
19 $\quad\quad\quad\quad p(z_j) = \operatorname{argmin}_{z \in T} c(z, x_i) + F(z)$

The backtracking phase, described in Algorithm 2, reconstructs the optimal solution based on the previously tabulated backlinks returned by the best predecessor function $p$.

**Algorithm 2:** Backtracking of the optimal solution

1 $R = (\ )$
2 $s = \min_{z \in S_N} F(z)$
3 **while** $s \neq None$ **:**
4 $\quad R = s \oplus R$
5 $\quad s = p(s)$
6 **return** $R$

The approach described in Algorithms 1 and 2 can be implemented leveraging a tree structure, where, for each stage $i$ and for each state $j$, each node $n_{i,j}$ stores the state segment $s_j = (\mathbf{x}, \mathbf{y}) \in S_i$, the optimal cost $F(s_j)$, and a back-link to its optimal predecessor (e.g., a pointer to its optimal predecessor node). Firstly, we create and complete the nodes of the tree according to Algorithm 1. Figure 2 shows the tree structure after Algorithm 1 is completed: each node contains the state, $s_j$, the optimal cost $F$, and points to its optimal predecessor. Secondly, we backtrack the optimal solution by selecting the node with the minimum value of $F$ at the last stage and following the chain of backlinks, pushing the last element of $s_j$ to the front of a list until we reach the initial state, where we push the starting point to complete the list of waypoints to be returned.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 29 March 2024 doi:10.20944/preprints202403.1831.v1



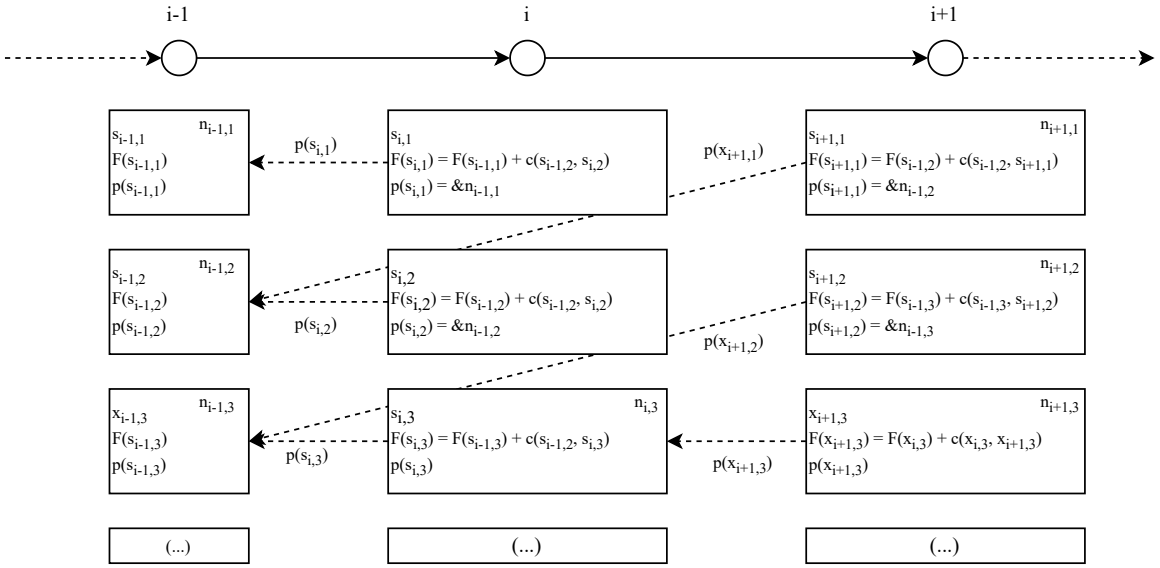



**Figure 2.** Tabular representation of the partial solutions in a tree structure. For each stage $i$ and for each state $j$, each node $n_{i,j}$ stores the state $s_j \in S_i$, the optimal cost $F(s_j)$, and a back-link to its optimal predecessor.

Concerning Algorithm 1, we can estimate the number of stages at each state as $(2D+1)^2$, i.e., the total number of transitions to be evaluated at each stage is $(2D+1)^4$. Since the only transitions to be evaluated are those whose initial state ends in the starting point of the final state, a proper implementation allows accessing them directly in constant time; thus, the number of evaluated transitions can be reduced to $(2D+1)^3$ in constant time, leading to a time complexity of $O(ND^3)$ for the overall process if the cost and constraints have constant time complexity.

## 6. Greedy Approximation

From Equation 14, we can notice that only part of the conditions for transition feasibility depend on the state $s_{i-1}$, as the transition cost $c$ defined in Equation 23 does. In this section, we propose a greedy approximate dynamic programming (GADP) scheme that makes greedy decisions to reduce the number of evaluated transitions. The proposed scheme loses the capacity to find the globally optimal policy, yet it allows for reducing the time complexity of the algorithm. The basic idea is to reformulate the route $R = (\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_N)$ as a sequence of transitions between consecutive states $\mathbf{x}_i \in X_i$, $i \in \{0, 1, ..., N\}$. In other words, we use the waypoints to represent the ship's state rather than representing it with a leg connecting two consecutive waypoints. The greedy minimum cost at stage $i$ is expressed by the function $F_g : X_i \to \mathbb{R}$, while the greedy optimal predecessor is represented by the function $p_g : X_i \to X_{i-1}$:

$$F_g(\mathbf{x}_i) = \min_{\mathbf{z} \in S_{i-1} | t_g(\mathbf{z}, \mathbf{x}_i)} c_g(\mathbf{z}, \mathbf{x}_i) + F_g(\mathbf{z}) \tag{24}$$

$$p_g(\mathbf{x}_i) = \operatorname*{argmin}_{\mathbf{z} \in S_{i-1} | t_g(\mathbf{z}, \mathbf{x}_i)} c_g(\mathbf{z}, \mathbf{x}_i) + F_g(\mathbf{z}) \tag{25}$$

Where $t_g : X_{i-i} \times X_i \to \{T, F\}$ and $c_g : X_{i-i} \times X_i \to \mathbb{R}$ greedily compute the result based on $p_g : X_i \to X_{i-1}$:

$$t_g(\mathbf{x}, \mathbf{y}) = t((p_g(\mathbf{x}), \mathbf{x}), (\mathbf{x}, \mathbf{y})) \tag{26}$$

$$c_g(\mathbf{x}, \mathbf{y}) = c((p_g(\mathbf{x}), \mathbf{x}), (\mathbf{x}, \mathbf{y})) \tag{27}$$

Algorithm 3 shows the proposed approach. Since a whole cycle over $[-D, ...D]$ disappears in the solution scheme, we now evaluate only $(2D+1)^2$ transition at each stage, and the time complexity required to run the algorithm drops from $O(ND^3)$ to $O(ND^2)$.

 doi:10.20944/preprints202403.1831.v1



**Algorithm 3:** Approximate dynamic programming bottom-up solution scheme

$S_0 = \{\mathbf{x}_{start}\}$
$F(\mathbf{x}_{start}) = 0$
$p(\mathbf{x}_{start}) = None$
**for** $i \in [1, \dots, N]$ **:**
    $S_i = \emptyset$
    **parallel for** $j \in [-D, \dots, D]$ **:**
        $\mathbf{x}_j = \mathbf{x}_{start} + \frac{i}{N}\delta_x \mathbf{e}_x + \frac{j}{D}\delta_y \mathbf{e}_y$
        $T = \{\mathbf{z} \in S_{i-1} | t_g(\mathbf{z}, \mathbf{x}_j)\}$
        **if** $T \neq \emptyset$ **:**
            $S_i = S_i \cup \{\mathbf{x}_j\}$
            $F_g(\mathbf{x}_i) = \min_{\mathbf{z} \in T} c_g(\mathbf{z}, \mathbf{x}_i) + F_g(\mathbf{z})$
            $p_g(\mathbf{x}_i) = \operatorname{argmin}_{\mathbf{z} \in T} c_g(\mathbf{z}, \mathbf{x}_i) + F_g(\mathbf{z})$

The backtracking phase works similarly to the previous case.

## 7. Case Study

An implementation of the two proposed algorithms has been developed for testing and comparison purposes, and some test case scenarios have been designed to compare and evaluate the algorithms' application in autonomous navigation contexts. The algorithms have been implemented in Rust language, relying on the Rayon library for parallelization. All scenarios take place in a square domain $D = [0, 10] \times [0, 10]$ nautical miles, in which the own ship starts from point $(0, 0)$ with heading $0°$ to reach the opposite side of the domain, i.e., any point $(10, y) \in D$.

- Scenario 1, shown in Figure 3a, has two barriers placed at $x = 5$ and $x = 9$ nautical miles respectively, ranging in $[-5, 2.5]$ and $[-2.5, 5]$. These obstacles force the own ship to maneuver between the barriers.
- Scenario 2, shown in Figure 3b, features two sailboat vessels, the first starting in $(8.0, 4.5)$ with a speed of 5.0 knots, and a heading of $270^{circ}$, the second positioned at $(4, -2)$, with speed 6 knots and heading $90^{circ}$. For both target vessels, the COLREG imposes a "give-way" behavior.
- Scenario 3, shown in Figure 3c, features a double "head-on" with two target vessels starting from $(9, 1)$ and $(10, 0)$ and heading $180^{circ}$ and speeds of 9 and 8 knots, respectively. In addition, two fixed side barriers form a channel parallel to the $x$ axis and 8 nautical miles wide.

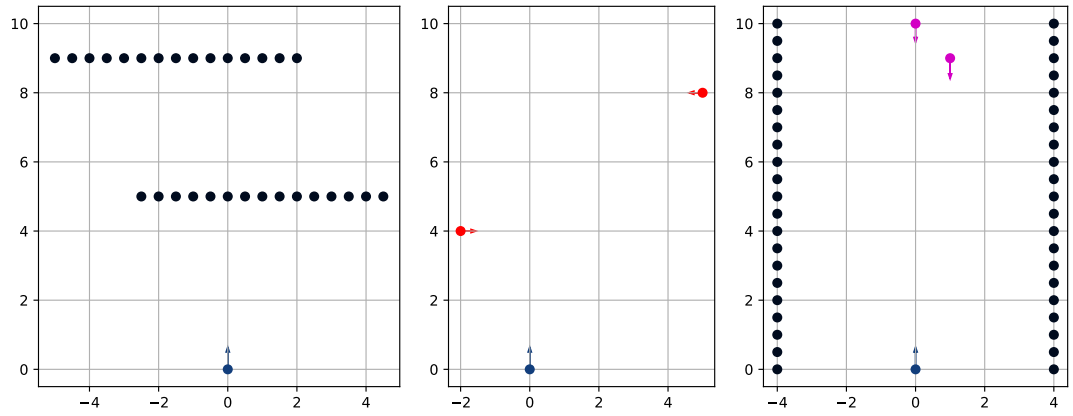
To comply with the COLREG, the own ship must make only visible heading alterations, with a minimum angle of $15^{circ}$, while a maximum of $60^{circ}$ turn is accepted. The algorithms perform the path optimization on a discrete computation grid defined as per Equation 7, where $N = 10$ and $D = 20$.

Figure 4 shows the solutions found by the two algorithms in the three proposed test scenarios. In scenarios 1 and 2 (Figures 4a and 4b), DP and GADP propose dissimilar trajectories; in particular, the GADP solution features more delayed direction changes. In scenario number 3 (Figure4c), on the contrary, the solution computed by the two approaches is the same, i.e., the greedy optimum computed by GADP corresponds to the global optimum of the DP.

Eventually, Figures 5a and 5b present the value of the cost function and the computation time required to determine the solution, respectively. We can note that, at the price of a higher cost function value, the solutions determined by the GADP algorithm require less computation time.

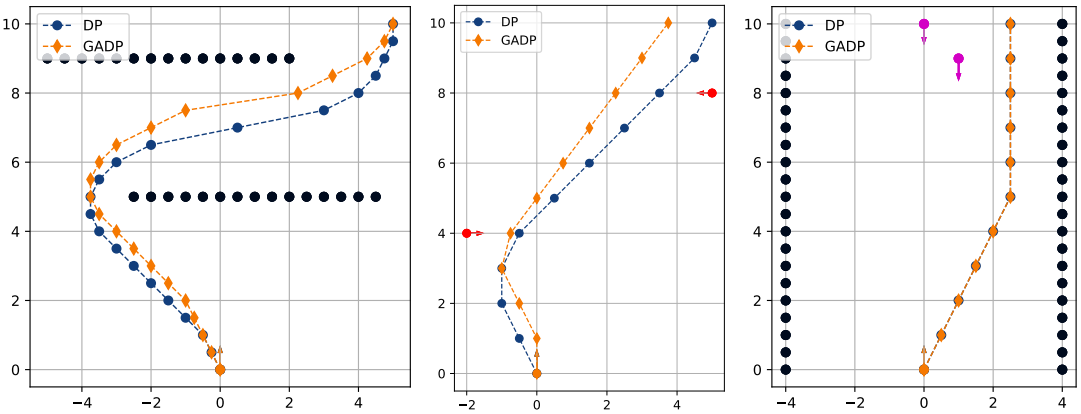
 doi:10.20944/preprints202403.1831.v1





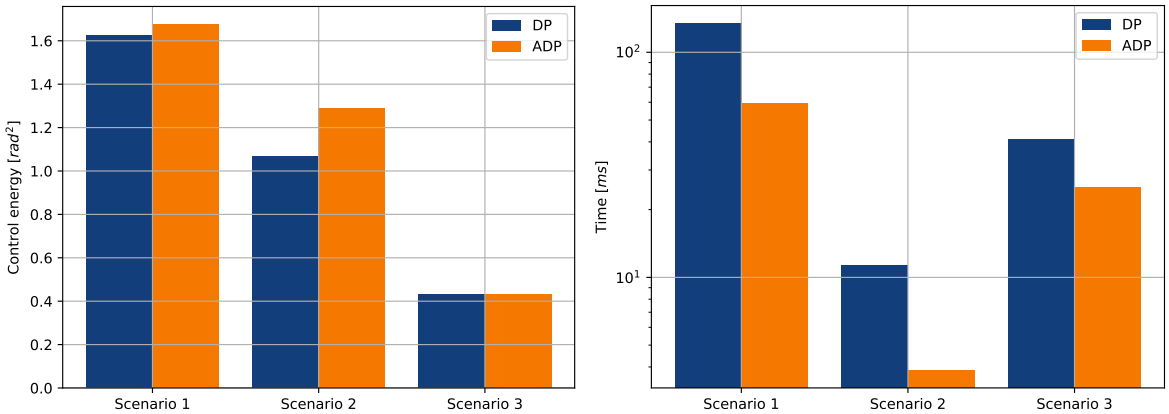
**(a)** Scenario 1 **(b)** Scenario 2 **(c)** Scenario 3

**Figure 3.** Application case scenarios. Scenario 1 (3a) includes fixed obstacles only; Scenario 2 (3b) features two sailboats (in red), both requiring "give-way" behavior to the own ship (blue); Scenario 3(3b) features two head-on ships (magenta) in a narrow channel.




**(a)** Scenario 1 **(b)** Scenario 2 **(c)** Scenario 3

**Figure 4.** Application case scenarios solved using DP (blue) and GADP (red).




**(a)** Cost function (Control energy). **(b)** Computation time.

**Figure 5.** Cost function value (Figure 5a) and computation time (Figure 5b) comparison of the two presented algorithms.

 doi:10.20944/preprints202403.1831.v1



## 8. Conclusions

This paper described a dynamic programming scheme for calculating evasive maneuvers of autonomous ships. We showed how the collision-free route calculation for a ship can be described by leveraging Bellman's equation, and we described appropriate constraints to obtain a collision-safe route with features compatible with the maneuvering capabilities of a ship and compliant with collision regulations. We also proposed a greedy approximate dynamic programming (GADP) scheme that allows, using a greedy approach, to reduce the number of transitions to be evaluated for each step, consequently reducing the algorithm's time complexity. Finally, we compared the proposed algorithms on three scenarios relevant to autonomous navigation.

**Funding:** This research was partially funded by European Union's Horizon Europe under the call HORIZON-CL5-2022-D6-01 (Safe, Resilient Transport and Smart Mobility services for passengers and goods), grant number 101077026, project name SafeNav. However, the views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or Executive Agency (CINEA). Neither the European Union nor the granting authority can be held responsible.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| DP | Dynamic Programming |
| COLREG | Convention on the International Regulations for Preventing Collisions at Sea |
| CPA | Closes Point of Approach |
| GADP | Greedy Approximate Dynamic Programming |
| IMO | International Maritime Organization |

## References

1. Martelli, M.; Virdis, A.; Gotta, A.; Cassarà, P.; Di Summa, M. An outlook on the future marine traffic management system for autonomous ships. *IEEE Access* **2021**, *9*, 157316–157328.
2. Tran, H.A.; Johansen, T.A.; Negenborn, R.R. Collision avoidance of autonomous ships in inland waterways–A survey and open research problems. In Proceedings of the Journal of Physics: Conference Series. IOP Publishing, 2023, Vol. 2618, p. 012004.
3. IMO, M. Outcome of the Regulatory Scoping Exercise for the Use of Maritime Autonomous Surface Ships (MASS), 2021.
4. 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. *Maritime Technology and Engineering, Proceedings of the MARTECH* **2014**.
5. Alessandri, A.; Donnarumma, S.; Vignolo, S.; Figari, M.; Martelli, M.; Chiti, R.; Sebastiani, L. System control design of autopilot and speed pilot for a patrol vessel by using LMIs. *Towards green marine technology and transport* **2015**, pp. 577–583.
6. Singh, Y.; Bibuli, M.; Zereik, E.; Sharma, S.; Khan, A.; Sutton, R. A novel double layered hybrid multi-robot framework for guidance and navigation of unmanned surface vehicles in a practical maritime environment. *Journal of Marine science and Engineering* **2020**, *8*, 624.
7. Bruzzone, G.; Bibuli, M.; Caccia, M.; Zereik, E. Cooperative robotic maneuvers for emergency ship towing operations. In Proceedings of the 2013 MTS/IEEE OCEANS-Bergen. IEEE, 2013, pp. 1–7.
8. Chen, L.; Huang, Y.; Zheng, H.; Hopman, H.; Negenborn, R. Cooperative multi-vessel systems in urban waterway networks. *IEEE Transactions on Intelligent Transportation Systems* **2019**, *21*, 3294–3307.
9. Chen, L.; Haseltalab, A.; Garofano, V.; Negenborn, R.R. Eco-VTF: Fuel-efficient vessel train formations for all-electric autonomous ships. In Proceedings of the 2019 18th European Control Conference (ECC). IEEE, 2019, pp. 2543–2550.
10. Faggioni, N.; Ponzini, F.; Martelli, M. Multi-obstacle detection and tracking algorithms for the marine environment based on unsupervised learning. *Ocean Engineering* **2022**, *266*, 113034.
11. Organization, I.M. Convention on the International Regulations for Preventing Collisions at Sea, 1972.

 doi:10.20944/preprints202403.1831.v1




12. Burmeister, H.C.; Constapel, M. Autonomous collision avoidance at sea: A survey. *Frontiers in Robotics and AI* **2021**, *8*, 739013.
13. Zaccone, R.; Martelli, M. Interaction between COLREG-compliant collision avoidance systems in a multiple MASS scenario. In Proceedings of the Journal of Physics: Conference Series. IOP Publishing, 2023, Vol. 2618, p. 012006.
14. Singh, Y.; Sharma, S.; Sutton, R.; Hatton, D.; Khan, A. A constrained A* approach towards optimal path planning for an unmanned surface vehicle in a maritime environment containing dynamic obstacles and ocean currents. *Ocean Engineering* **2018**, *169*, 187–201.
15. Seo, C.; Noh, Y.; Abebe, M.; Kang, Y.J.; Park, S.; Kwon, C. Ship collision avoidance route planning using CRI-based A* algorithm. *International Journal of Naval Architecture and Ocean Engineering* **2023**, *15*, 100551.
16. Singh, Y.; Sharma, S.; Sutton, R.; Hatton, D.; Khan, A. Feasibility study of a constrained Dijkstra approach for optimal path planning of an unmanned surface vehicle in a dynamic maritime environment. In Proceedings of the 2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). IEEE, 2018, pp. 117–122.
17. Singh, Y.; Sharma, S.; Sutton, R.; Hatton, D. Towards use of Dijkstra algorithm for optimal navigation of an unmanned surface vehicle in a real-time marine environment with results from artificial potential field **2018**.
18. D'Amato, E.; Nardi, V.A.; Notaro, I.; Scordamaglia, V. A Visibility Graph approach for path planning and real-time collision avoidance on maritime unmanned systems. In Proceedings of the 2021 International Workshop on Metrology for the Sea; Learning to Measure Sea Health Parameters (MetroSea). IEEE, 2021, pp. 400–405.
19. Chiang, H.T.L.; Tapia, L. COLREG-RRT: An RRT-based COLREGS-compliant motion planner for surface vehicle navigation. *IEEE Robotics and Automation Letters* **2018**, *3*, 2024–2031.
20. Zaccone, R.; Martelli, M. A collision avoidance algorithm for ship guidance applications. *Journal of Marine Engineering & Technology* **2020**, *19*, 62–75.
21. Enevoldsen, T.T.; Reinartz, C.; Galeazzi, R. COLREGs-Informed RRT* for collision avoidance of marine crafts. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 8083–8089.
22. Ito, M.; Zhnng, F.; Yoshida, N. Collision avoidance control of ship with genetic algorithm. In Proceedings of the Proceedings of the 1999 IEEE International Conference on Control Applications (Cat. No. 99CH36328). IEEE, 1999, Vol. 2, pp. 1791–1796.
23. Kang, Y.T.; Chen, W.J.; Zhu, D.Q.; Wang, J.H.; Xie, Q.M. Collision avoidance path planning for ships by particle swarm optimization. *Journal of Marine Science and Technology* **2018**, *26*, 3.
24. Ning, J.; Chen, H.; Li, T.; Li, W.; Li, C. COLREGs-Compliant unmanned surface vehicles collision avoidance based on multi-objective genetic algorithm. *Ieee Access* **2020**, *8*, 190367–190377.
25. Gao, P.; Zhou, L.; Zhao, X.; Shao, B. Research on ship collision avoidance path planning based on modified potential field ant colony algorithm. *Ocean & Coastal Management* **2023**, *235*, 106482.
26. Bellman, R. The theory of dynamic programming. *Bulletin of the American Mathematical Society* **1954**, *60*, 503–515.
27. Bellman, R. Dynamic programming. *Science* **1966**, *153*, 34–37.
28. Jones, M.; Peet, M.M. A generalization of Bellman's equation with application to path planning, obstacle avoidance and invariant set estimation. *Automatica* **2021**, *127*, 109510.
29. Ayyappan, B.; Gopalan, S. A Performance and Power Characterization study of Memoization and Tabulation methods in Graph Neural Networks by assessing Dynamic Programming Workloads. In Proceedings of the 2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE). IEEE, 2022, pp. 1–6.
30. Rytter, W. On efficient parallel computations for some dynamic programming problems. *Theoretical Computer Science* **1988**, *59*, 297–307.
31. González, D.; Almeida, F.; Roda, J.; Rodriguez, C. From the theory to the tools: parallel dynamic programming. *Concurrency: practice and experience* **2000**, *12*, 21–34.
32. Guo, Q.; Li, Z.; Song, W.; Fu, W. Parallel computing based dynamic programming algorithm of track-before-detect. *Symmetry* **2018**, *11*, 29.
33. Mazzarello, M.; Ottaviani, E. A traffic management system for real-time traffic optimisation in railways. *Transportation Research Part B: Methodological* **2007**, *41*, 246–274.

 doi:10.20944/preprints202403.1831.v1




34. Shin, K.; McKay, N. A dynamic programming approach to trajectory planning of robotic manipulators. *IEEE Transactions on Automatic Control* **1986**, *31*, 491–500.
35. Li, X.; Wang, L.; An, Y.; Huang, Q.L.; Cui, Y.H.; Hu, H.S. Dynamic path planning of mobile robots using adaptive dynamic programming. *Expert Systems with Applications* **2024**, *235*, 121112.
36. Zaccone, R.; Ottaviani, E.; Figari, M.; Altosole, M. Ship voyage optimization for safe and energy-efficient navigation: A dynamic programming approach. *Ocean engineering* **2018**, *153*, 215–224.
37. Zaccone, R.; Figari, M.; Martelli, M. An optimization tool for ship route planning in real weather scenarios. In Proceedings of the ISOPE International Ocean and Polar Engineering Conference. ISOPE, 2018, pp. ISOPE–I.
38. Zhengping, T.; Chao, J.; Xiaohai, W. Design of ship route through waterway based on dynamic programming. In Proceedings of the Journal of Physics: Conference Series. IOP Publishing, 2021, Vol. 1906, p. 012001.
39. Choi, G.H.; Lee, W.; Kim, T.w. Voyage optimization using dynamic programming with initial quadtree based route. *Journal of Computational Design and Engineering* **2023**, p. qwad055.
40. Zaccone, R.; Martelli, M.; Figari, M. A colreg-compliant ship collision avoidance algorithm. In Proceedings of the 2019 18th European Control Conference (ECC). IEEE, 2019, pp. 2530–2535.
41. Martelli, M.; Žuškin, S.; Zaccone, R.; Rudan, I. A COLREGs-compliant decision support tool to prevent collisions at sea. *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation* **2023**, *17*.


## Short Biography of Authors

**Raphael Zaccone** received his MSc in naval architecture and marine engineering in 2013 from the University of Genoa. Subsequently, he received his PhD in 2017 with research on ship optimal weather routing, then continued his research in autonomous navigation and collision avoidance as a Post Doc. He joined the Department of Naval, Electrical, Electronic, and Telecommunications Engineering at the University of Genoa in 2019 as a Research Fellow and, from 2021, Assistant Professor. His research activities cover various areas of ship digitization and are mainly focused on developing and applying motion planning and collision avoidance algorithms for enhanced navigation, decision support, and autonomous marine vehicles. He has authored more than 35 scientific publications in international journals and conferences.