Developing Decision-Making Algorithm for Unmanned Vessel Navigation Using Markov Processes

Ruolan Zhang [a,*], and Masao Furusho[b]

Graduate School of Maritime Sciences, Kobe University, Kobe, Japan
a. anco1437@gmail.com, b. furusho@maritime.kobe-u.ac.jp
**corresponding author*

Keywords: Markov decision processes; optimization algorithm; path planning; risk perception; unmanned vessel

Abstract: In this study, the autonomous decision-making architecture of unmanned vessel navigation has been formulated. The aim of this study is the advancement of mathematical methods in the ship transportation field with relevance to collision avoidance scenario applications. The process of seafarers safely navigating a vessel at sea entails enacting appropriate decision-making at the appropriate time. In our model, we do not input the appropriate action order based on a seafarer's experience. The model scores each step's reward by its action behaviour and learns how to avoid obstacles by itself. By deploying decision timing, state, reward, and digitizing the seafarer's decision, we establish a reinforcement learning algorithm based on Markov decision processes. In the model training, under a single factor influence, the vessel tends to change course with the best appropriate action behaviour, which is almost consistent with decision-making behaviour based on actual experience at sea.

1. Introduction

In the past few decades, increasing attention has been paid to Markov decision process (MDP) algorithms, partly due to the success of self-driving car research using reinforcement learning methods [1]. With the rapid development of automatic control, the Internet of Things (IoT), big data, state awareness, telecommunication, and other navigation technologies, the technical feasibility of smart ships has increased broadly [2]. In particular, the unmanned vessel yet to be christened, Yara Birkeland, is expected to begin its voyage in 2018 [3]. A large-scale merchant transportation of unmanned vessels has obvious advantages. For example, a fleet of unmanned container ships would maximize the capacity of storage space through the exclusion of the bridge and living area for the on-board seafarers; thereby maximizing the cargo volume and improving the transportation efficiency.

Moreover, many of the facilities on board serve the on-board seafarers, such as life-saving equipment, firefighting apparatus, pollution prevention, and living facilities. In the absence of seafarers, such equipment will not be required, reducing the weight of the ship and energy consumption, lowering the construction and operating costs, and increasing the ship's cargo capacity. Moreover, the main causes of maritime accidents are human-based factors, such as inadequate decision-making, operational negligence, deficient emergency response, and other seafarer factors. In an unmanned vessel, ship maneuvering is conducted primarily through automatic decision-making and remote monitoring by personnel working under better working conditions at a shore side control station [4]. Thus, the impact of human-based factors is reduced.

However, the challenges are still many. One of the key problems of maritime navigation is the current trend towards human-centered decision-making systems. The process of seafarers maneuvering a ship at sea entails enacting the appropriate decision-making procedures at the appropriate time. Reinforcement learning has proven capable of developing learning models that are effective in planning [5]. In the prior research, the authors used the Robot Operating System as a tool, extended the Markov Decision-making (MDM) and supported the decision-making methodologies based on MDPs [6]. The aim of the MDPs is to provide an action set of decision-making for the on-board cycle.

Based on the complex maritime environment, it is observed that predetermination is the basic requirement to achieve safe navigation of unmanned vessels. This study consists of three sections. The first section clarifies the concept of MDPs, especially focusing on how to use MDPs to optimize the decision-making procedure under the situation of no seafarer on the bridge. In the second section, using MDPs to formulate a mathematical model, we classify the different elements of the navigation behaviour state. The third section concludes this study by presenting an algorithm for a very simple demonstration. The purpose of this paper is to build an automatic decision cycle model to increase the decision-making efficiency and safety performance of unmanned ships, the experiments conducted in this article also proved the performance of the autonomous decision-making can be improved. Under the single factor influence, with the model training, this paper also will provide the result that the vessel trends to change the course with the best appropriate maneuver behaviour, which is almost consistent with decision-making behaviour based on actual expert's experience at sea.

2. Mathematical Description of the Navigation State

Generally, in a vessel navigating at sea, the officer of the watch learns to recognize the state of the navigation environment through equipment and his look-out experience. The scenario in this study describes a case without any seafarers on the bridge; therefore, the vessel only recognizes the surrounding environment with devices such as radar, camera, and automatic identification system (AIS). Within the scope of this study, we assume that the data obtained by all other devices has been integrated into a similar state framework (radar screen or electronic chart display). As shown in Fig. 1, a cross encounter situation has been constructed between the vessel at the port side and the own ship. According to the International Regulations for Preventing Collisions at Sea (COLREGS), the own ship has no obligation to take an action to avoid the target vessel, but it should always pay attention to the changing state. Further, there is an obstacle (oil rig) in front of the starboard side of the own ship, and it needs to pass beyond a safe distance. This constitutes one of the most common navigational environments at sea.

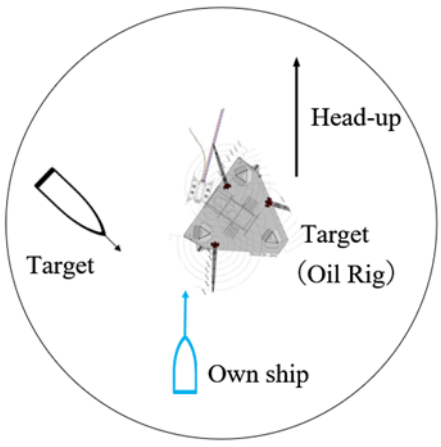

Figure 1: Example of the own ship encounters an obstacle state

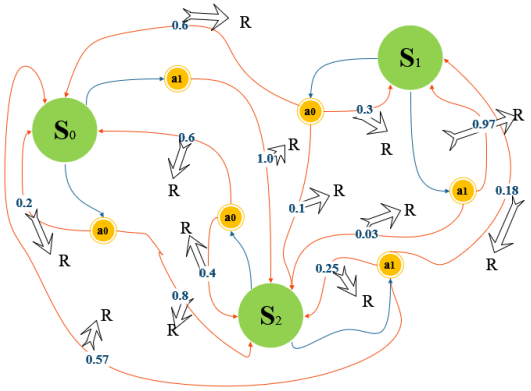

Figure 2: Example of the own ship encounters an obstacle state

This state changes, based on the surrounding environment and the change of the navigational state of the own ship. Each state change of the model requires a decision that weighs the next safe action behaviour and the risk probability of exercising the action behaviour. Therefore, the own ship must comprehensively predict and decide for the action change of each state to determine the optimal maneuver policy. There are two main ways to change the action behaviour of the own ship: change the course by steering (turn to port side 'P' or turn to starboard side 'S') or change speed (accelerate 'A' or decelerate 'D'). Due to the different amplitude of each action change, it can be regarded as a discrete vector set, as well as the entire control tensor consisting of the free combination of the two action sets: T_s= {A,D,P,S,AP,AS,DP,DS}.

3. MDPs

MDPs are usually advantageous for approaching a wide range of optimization problems solved through dynamic programming and reinforcement learning. It is a class of stochastic sequential decision processes, in which the reward and transition functions depend only on the current state of the model and the current action [7]. With the mathematical framework of the navigation states constructed, we can approach a simple model of completely autonomous decision-making based on Markov processes, as shown in Fig. 2. The green and orange circles represent the different states and actions, respectively. MDPs show that there may be more than one result per action in different states. For example, after the state S1 passes the action a0, it may return to the previous state S0 or change to the state S2. Moreover, it is also possible to keep the current state S1 with nothing to change. The number above the vector arrow represents the probability that the previous state will shift to another state after the action. R, corresponding to the vector arrow, is the reward we need to observe.

3.1.State Space

The decision state space, S, is the state of the unmanned navigation environment, which is the combination of a vector set of the state, P_((x)), and a vector set of the dynamic change of external context, Q_((y)). P_((x))=P_1, 〖 P〗 _2,⋯,P_m describes the set of actions, from act 1 to act m; while Q_((y))=Q_1,Q_2,⋯,Q_n is the set of vectors whose model takes a specific decision procedure to change the navigation state. Therefore, we obtain the formula S=(■(P_((x)),&Q_((y)),& ρ)), where ρ is the factorial of the set, i!, that describes the total summation of the most recent situation. The decision state space can be described as

$$\bar{\theta} = \left\{ (P_{(x)}, \quad Q_{(y)}, \quad \rho) \middle| \begin{matrix} \sum_{m=0}^{i-1} x_i \le x \\ \sum_{n=0}^{i} y_i \le y \\ x_i, y_i \ge 0 \\ r = 0,1,2,\dots,i \\ m = 0,1,2,\dots,i-1 \\ n = 0,1,2,\dots,i \end{matrix} \right\}, \quad (1)$$

where y > 0 is the model within a time period (which may be a very short period of time) for the amount of operations permitted.

$$0 \le \mathrm{y} \le \lim_{x \to \infty} \sum P_{(x)} \quad (2)$$

The following four equations can be obtained from $P_{(x)}$, $Q_{(y)}$, and r:

$$\theta_1 = \left\{ \left((P_{(x)}, \quad Q_{(y)}, \quad 0) \middle| \begin{matrix} \sum_{m=0}^{i-1} x_m = 0 \\ \sum_{n=0}^{i} y_n = 0 \end{matrix} \right) \right\} \in \bar{\theta} \quad (3)$$

θ_1 describes the model with no change in the unmanned vessel's maneuvers and no change in the dynamic state of the external context for all the possible states. Here, r = 0.

$$\theta_2 = \left\{ \left((P_{(x)}, \quad Q_{(y)}, \quad 0) \middle| \begin{matrix} \sum_{m=0}^{i-1} x_m > 0 \\ \sum_{n=0}^{i} y_n = 0 \end{matrix} \right) \right\} \in \bar{\theta} \quad (4)$$

θ_2 describes a single change in the unmanned vessel's maneuvers, but no change in the external context. This situation rarely occurs. Thus, the index weight will be relatively small.

$$\theta_3 = \left\{ \left((P_{(x)}, \quad Q_{(y)}, \quad i) \middle| \begin{matrix} \sum_{m=0}^{i-1} x_m = 0 \\ \sum_{n=0}^{i} y_n > 0 \end{matrix} \right) \right\} \in \bar{\theta} \quad (5)$$

θ_3 describes a single external context change. For instance, it may have detected a new target that could have an impact on the own ship's safety. The model needs to continue to follow up the situation, so r = i.

$$\theta_4 = \left\{ \left((P_{(x)}, \quad Q_{(y)}, \quad i) \middle| \begin{matrix} \sum_{m=0}^{i-1} x_m > 0 \\ \sum_{n=0}^{i} y_n > 0 \end{matrix} \right) \right\} \in \bar{\theta} \quad (6)$$

θ_4 describes the situation where a change occurs in both the unmanned vessel's maneuvers and the external context. This situation commonly occurs in complex waters, and the model needs a continuous decision-making process. Thus, the index weight will be relatively large.

Subsequently, $\bar{\theta} = \theta_1 \cup \theta_2 \cup \theta_3 \cup \theta_4$. When the unmanned boat's performance limit y is a constant, θ represents all the possible operational decision states.

$$\mathrm{S} = \sum_{m=0}^{i-1} x_i \quad (7)$$

3.2. Action Set

For an unmanned vessel in the actual navigation process, from point A to point B, the system's navigation environment variables may occur in the entire navigation time interval [a, b]. In a safe navigation condition, the unmanned vessel does not have to make any maneuvers to change the model's navigational state; the actual decision-making time occurs only when the unmanned vessel encounters something affecting the safety of the navigation. Thus, the number of valid environment variables is the number of the MDP model's decision moments.

For the embedded MDP, it follows from the previous discussion that the effective decision time can only be generated in the θ_2, θ_4 decision state space, and the set of ship operations that can be selected belongs to set T_s.

$$T_s = \begin{cases} \{0\}, & s \in \theta_1 \\ \{\mathrm{A, D, P, S, AP, AS, DP, DS}\}, & s \in \theta_2 \\ \{0\}, & s \in \theta_3 \\ \{\mathrm{A,D,P,S,AP,AS,DP,DS}\}, & s \in \theta_4 \end{cases} \quad (8)$$

3.3. Reward Function

It is assumed that the operating time interval of the unmanned vessel decision model is a time series that follows an exponential distribution. For the state S transferring to state S' by action A, the mathematical expression is:

$$\mathrm{r(s'|s, a)} = \begin{cases} 0, & s \in \theta_1 \cup \theta_2 \\ \int_a^b r_{\delta\omega} td(1 - e^{-p_{\delta\omega}}), & s \in \theta_3 \cup \theta_4 \end{cases}, \quad (9)$$

where $r_{\delta\omega}$ is the reward generated by the traffic event occurrence of the action ω^{th}, when the model is under the decision operation δ^{th}, and $p_{\delta\omega}$ is the probability that the unmanned vessel in the δ^{th} decision-making operation needs to act upon the ω^{th} operation.

The MDP decision model based on a continuous timeline is now complete.

3.4. Environment Formulation

Maritime long-distance transportation vessels have great inertia, resulting in difficult control. In the case of deceleration, the ship's power system may cause damage, while frequent steering using the rudder could reduce the service life of the steering gear and propeller. Therefore, to optimize maneuvering, this study introduces the Markov process, with safe navigation constraints, as an unmanned vessel decision model. It integrates all the navigation data in a continuous decision-making timeline for the action set *T*. In an avoidance scenario, the local reward function not only needs to obtain the maximum value, but also needs to satisfy the global process to get the maximum reward value. Thus, it needs to achieve the purpose of optimizing navigation with less manipulation and obtain a larger reward function value. We can continue to simplify the MDPs model, through the Python computer programming language, to build a simple marine collision avoidance environment for simulation and authentication.

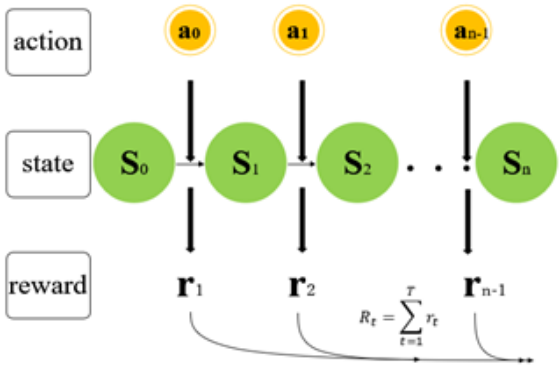

Figure 3:Simplified single-target, single-task Markov chain process

4. Decision-Making Training and Analysis

4.1 Simplified Decision Algorithm

The process shown in Fig. 3 is a navigation situation usually encountered in a seaborne vessel. The own ship belongs to the original state from S0. According to the changes in the surrounding environment, the model tries to bring up a series of simple action behavior a0. The model keeps changing the status of this ship until a complete collision avoidance has been achieved and gives the state Sn. The purpose of the decision-making algorithm is to try to give manipulation in the process. It does not directly tell the model manipulation of right or wrong. Thus, the model only scores a reward value by random action and makes the final global reward Rt approach an optimal value.

Some specific algorithm ideas are as follows:

Algorithm:

Initialize state S,

$MP(S,a) \leftarrow MP(S,a) + \alpha[R + \gamma max_{a'} MP(S',a') - MP(S,a)]$,

$S \leftarrow S'$,

until S is under the safe situation.

MP(S,a) for all possible values of the composition in S_A_table,

Pick a random action A from the original state and use the policy derived from S_A_table,

S_A_table is State_Action_table,

Take action A, observer, S'.

In the above algorithm, α is the learning efficiency, γ is the decay rate, $MP(S', a')$ is the true value, and $MP(S, a)$ is the estimated value.

In the model training of this study, we assume that the encountered scenario at sea is: according to the navigation experience and collision avoidance rules (COLREGS), the appropriate action behavior of an experienced officer of the watch is that if the "starboard five" is held for 10 s, it can safely avoid obstacles at sea. Therefore, to simplify the flow of demonstration algorithm, we can make a single-threaded simulation, where every second there is an action behavior taken. The action behavior can be the same as before the manipulation of this state, and only "turn to port five" or "starboard five" types of steering action can be chosen. After setting the reward value gained by different manipulative behaviors, the whole training process cannot interfere with human factors. If

the model is trained, it can perform the avoidance and steering operation around 10 actions, indicating that this method can be applied to the unmanned vessel collision avoidance operations in a real offshore environment.

Figure 4:Training model results (approach 10 steps to avoid obstacles)

4.2 Training Result

After 20 iterations of model training, we obtain the training results, as shown in Fig. 4. In the first training process, the model did not know which one was suitable and could avoid the manipulation of obstacles. Therefore, it took 72 steps to achieve the aim of obstacle avoidance, which is not suitable for the requirement of collision avoidance in a practical situation at sea. However, with the increase in training times, especially after training 5 times, the model quickly learned that if "starboard five" is the most effective action to avoid the obstacles in front of it, the number of subsequent rudder steps is obviously cut back. Finally, the model can be stabilized in about 10 steps, to complete the collision avoidance manipulation.

4.3 Steering Action Data Analysis

Moreover, as the training times increase, we can also obtain results as shown in Fig. 5. The abscissa axis in the figure is the number of steps performed in one state, and the ordinate axis is the average of the probabilities for decision-making. With the change of state, the decision made by the algorithm at the beginning does not know whether turn to port or starboard can be obtained a desirable outcome, but statistics of the results of 20 times training can be observed. Generally, after five steps, the "starboard five" action has significantly improved its probability compared to the "port five" decision, making it quicker to learn how to avoid the obstacles in front of it. Therefore, it also proves that the vessel is controlled to navigate safely, and the Markov processes-based autonomous decision-making model is fully applicable to the unmanned vessel navigation at sea.

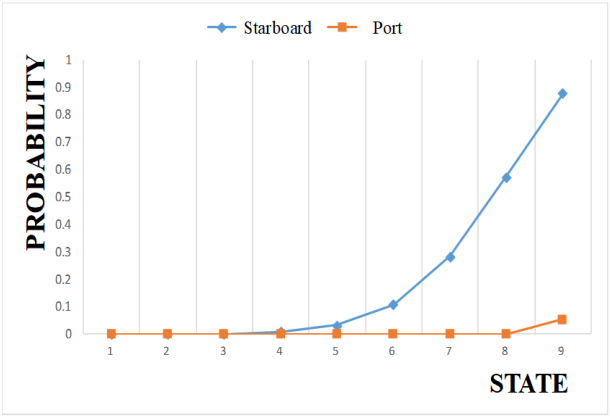

Figure 5: Another perspective proves the ability of MDPs to make efficient decisions

5. Conclusion

This study formulated a Markov processes-based autonomous decision-making model to help an unmanned vessel achieve collision avoidance by methods learned by itself. Further, it constructed a simple common encounter environment, through the Python programming language, for a "starboard five" action, while keeping a 10 s rudder order to complete the obstacle avoidance action behavior that has training, which is known by officers of a watch's experience. No human-based factors control the model to avoid the obstacles. The model explores on its own, and according to each step, gets the appropriate reward for choosing the best appropriate solution to achieve the collision avoidance goal.

References

[1] *M.L. Puterman, Markov games as a framework for multi-agent reinforcement learning. In Machine Learning Proceedings pp. 157-163. 1994*

[2] *J. H. Ang, C. Goh, and Y. Li. "Smart design for ships in a smart product through-life and industry 4.0 environment," in Evolutionary Computation (CEC), 2016 IEEE Congress on. IEEE, 2016: 5301-5308.*

[3] *K. Maritime, "Autonomous ship project, key facts about the YARA Birkeland," URL https://www.km.kongsberg.com/ks/web/nokbg0240.nsf/AllWeb/4B8113B707A50A4FC125811D00407045?OpenDocument.*

[4] *T. Zarsky, "The trouble with algorithmic decisions: an analytic road map to examine efficiency and fairness in automated and opaque decision making," Sci. Technol. Hum. Values, vol. 41(1), pp. 118–132, 2016.*

[5] *D. Silver, H. V. Hasselt, M. Hessel, et al. "The predictron: End-to-end learning and planning." arXiv preprint arXiv:1612.08810, 2016.*

[6] *RL. Zhang, & M Furusho, Constructing a Deci-sion-Support System for Safe Ship-Navigation Using a Bayesian Network. International Conference on Human-Computer Interaction. Springer International Publishing, 616-628. 2016.*

[7] *M.L. Puterman, "Markov decision processes." Handbooks Oper. Res. Management Sci., vol. 2, pp. 331–434, 1990.*