Consideration of State Representation for Semi-autonomous Reinforcement Learning of Sailing Within a Navigable Area

Hideaki Manabe and Kanta Tachibana

Abstract To sail quickly to a goal within a navigable area, complex control of the rudder and sail is required. Sailors must determine the current action with consideration of the time series of states; i.e., both current and future states. Reinforcement learning is an appropriate method for learning a complex problem, such as sailing. In this paper, we apply the navigable area such that a robotic sailor must avoid touching a boundary. To realise a higher layer of sailing architecture, the action space is simplified and discretised to the degree of the sailboat direction change. Moreover, we utilize semi-autonomous reinforcement learning, also known as imitation learning, in which a human selects an action and a robot updates its Q-values to evaluate pairs of states and actions until the robot's action selection is equivalent to the human's. For semi-autonomous learning, as well as for normal reinforcement learning, a representation of the state space is important. The state representation should be defined so that the state space is discretised to specify a desirable action, thereby removing any redundancy if possible. In this paper, we verify and investigate the possibility of state representation.

1 Introduction

Reinforcement learning is a machine learning method proposed by Sutton and Barto [6]. Agents of learning recognize states and select actions. They learn how each state and action pair contributes to the rewards. The objectives of the World Robotic Sailing Championship consist of sub-tasks to move the sailboat to the desired area.

H. Manabe (✉)
Laboratory for Intelligence, Kogakuin University, Nishi-Shinjuku, Shinjuku, Tokyo, Japan
e-mail: em14012@ns.kogakuin.ac.jp

K. Tachibana (✉)
Laboratory for Intelligence, Faculty of Informatics, Kogakuin University, Nishi-Shinjuku, Shinjuku, Tokyo, Japan
e-mail: kanta@cc.kogakuin.ac.jp

A. Friebe and F. Haug (eds.), *Robotic Sailing 2015*,
DOI 10.1007/978-3-319-23335-2_7

The robot must recognize continuous state variables, position (x, y) and velocity $(\dot{x}, \dot{y})$, and determine the appropriate action to the destination. However, unlike the control of driverless cars, the acceleration of a sailboat depends on the apparent wind. Therefore, we make the agents learn to navigate to the goal using reinforcement learning. In addition, we consider that state is better represented by the directions of the apparent wind, goal areas, and obstacles in the boat coordinate system than by the sailboat position and velocity in the global coordinate system. The master's dissertation of Sterne [5] realised this observation by sailing to a desired direction through reinforcement learning. However, neither the task of avoiding obstacles nor route finding in navigable areas was realised.

Konidaris and Barto [1] produced successful results of reinforcement learning experiments in the pinball domain in which agents learned the route to a goal using bounces to fixed elastic walls. In the experiment, the state vector of continuous values $(x, y, \dot{x}, \dot{y})$ was used. They employed agents to choose an action out of five possible actions ($\dot{x} \leftarrow \dot{x} \pm \Delta, \dot{y} \leftarrow \dot{y} \pm \Delta$ or no accelerations, where Δ is a small fixed value). In our sailing task, we make sailboats circumnavigate a target area without touching the navigable area boundary by using continuous state spaces. State spaces should be represented in the sailboat coordinate system because the danger of colliding with a boundary of the navigable area depends on the apparent wind, even if the relative position and velocity moving toward the boundary of the navigable area are the same.

The number of state variables becomes too large if all measured variables are used when the agent is made to recognize the direction and distance to obstacles and the target area. The number of state variables can be reduced if we use the Fourier transform of measured distance data. Kuhl [2] showed that the original distance signal can be restored by inverse Fourier transformation with a relatively small number of Fourier coefficients.

In this paper, we simplify the action spaces, utilize semi-autonomous learning, and arrange state representation to make the agent circumnavigate target areas within the navigable area. Action space is discretized to $\Delta\theta \in \{0°, \pm 1°\}$, not to continuous values, as was done in Sterne's dissertation. We adopt semi-autonomous reinforcement learning which is also known as imitation learning [3, 4]. Semi-autonomous reinforcement learning is the method of learning actions that are selected by the human's operation at the beginning stage of learning, and the robot learns from the human's actions. For state representation, distances to the target area and boundaries to the navigable area at respective directions are measured in the sailboat coordinate system. They are then Fourier-transformed to reduce the number of state variables. The state variables consist of Fourier coefficients of distance to the target area and to obstacles, and a two-dimensional vector of apparent wind. We define the relative direction θ with respect to the sailboat. Then, we calculate distances to the target area $[d_+(\theta)]$ for all angles $\theta = 0°, 1°, \ldots, 359°$ around the sailboat. We calculate the distances to the boundaries of the navigable area $[d_-(\theta)]$ in the same way. If a ray to angle θ does not cross the target area, $d_+(\theta)$ becomes ∞. We define $\ell = \exp(-d)$ so that ℓ is within the range from 0 to 1. The functions $\ell_+(\theta)$ and $\ell_-(\theta)$ are naturally periodic; therefore, they are

Fourier-transformed. In this paper, we verify whether a reduced number of Fourier coefficients effectively represent the state space combined with the apparent wind vector.

The remainder of this paper is organized into six sections. In Sect. 2, sailing and our sailing simulator are explained. In Sect. 3, semi-autonomous learning and Q-learning are described. In Sect. 4, state space is defined and the methods of performing Fourier transformation of state representation and the dividing of state space are respectively described. In Sect. 5, our experiment and parameters, experimental scenario, and results are presented. In Sect. 6, we discuss our experiment. Our research conclusions and future work are discussed in Sect. 7.

2 Sailing Simulator

The main propulsive force of the sailboat is captured by the sail. This force can be reasonably approximated as being proportional to the sail area facing the wind and the squared wind speed. The propulsive component of this force pushes the sailboat forward, whereas the lateral component moves the sailboat sideways. Let W be the apparent wind speed, ϕ be the direction of the apparent wind, and φ' be the direction of the sail. Then, the sail gains force such that

$$F \propto W^2 \sin(\phi - \varphi'),$$

and its propulsive and lateral components are

$$F_x \propto W^2 \sin(\phi - \varphi') \sin \varphi',$$
$$F_y \propto W^2 \sin(\phi - \varphi') \cos \varphi',$$

respectively. For a given W and ϕ, the half angle $\varphi' = \phi/2$ maximizes the propulsive force, F_x. Therefore, $F_x \propto W^2(1 - \cos \phi)$ and $F_y \propto W^2 \sin \phi$

Figure 1 presents a screenshot of the sailing simulator used in this study. The wind direction, φ, is set to East-North-East; i.e., at $\varphi = -22.5°$ in the world coordinate system. The 'dead zone' is indicated by a striped pattern. When steering in the dead zone, the direction of the apparent wind becomes almost $\phi \approx 0°$; therefore, the propulsive force component is lost, and it is difficult for the sailboat to gain propulsion. Accordingly, sailors heading upwind cannot move directly and must choose a zigzag course to arrive at their goal in a shorter time. The direction perpendicular to the wind is called the 'abeam', in which the sailboat can move fast by maintaining a high apparent wind speed, W, and a good propulsive component $(1 - \cos \theta)$. Finally, the downwind direction is called the 'running', in which the

Fig. 1 Screenshot of the sailing simulator

sailboat cannot go faster than the abeam because the true wind is cancelled by the speed of the sailboat, and the apparent wind becomes approximately $W \approx 0$. Therefore, to arrive at the goal in a shorter time, the sailboat must avoid the dead zone and running directions and maintain an abeam direction as much as possible.

In the sailing simulator, it is assumed that the resistance force from the water is proportional to the square of the boat velocity. The leeway effect is also simulated with consideration of the lateral motion equation. Moreover, if the sailboat touches a boundary of the navigable area, its velocity is immediately changed to zero. The frame rate of the sailing simulator is set to 40 frames per second.

3 Semi-autonomous Reinforcement Learning

In Sect. 3.1, we explain reinforcement learning, especially Q-learning. We explain the method to renew Q-values and to select the action. In Sect. 3.2, we propose the method of semi-autonomous reinforcement learning.

3.1 Reinforcement Learning

Q-learning is a reinforcement learning method. An agent has a finite set of states S and a finite set of actions A. At each time frame, t, an agent recognizes its state, $s_t \in S$, and selection action, $a_t \in A$. The efficacy of a pair of states and actions is

quantified as the 'Q-value', $\hat{Q}(s_t, a_t)$. The Q-value is renewed with obtained reward r_t and Q-values of the next state s_{t+1} caused by action a_t as the following:

$$\hat{Q}(s_t, a_t) \leftarrow (1-\alpha)\hat{Q}(s_t, a_t) + \alpha\left\{r_t + \gamma \max_a \hat{Q}(s_{t+1}, a)\right\},$$

where $\alpha \in [0, 1]$ and $\gamma \in [0, 1]$ are the learning rate and discount rate, respectively. The method to select the action with the highest Q-value for the current state is called the greedy method. The method to select the action according to the greedy method with the probability of $(1-\varepsilon) \in (0, 1)$, and to otherwise randomly select the action, is called the ε-greedy method. The soft-max method is used to select action a with a probability that is proportional to $\exp\left(\beta\hat{Q}(s_t, a)\right)$, the exponential of its Q-value, where β is a constant. Q-value is updated for each step. Q-values are set at 0 initially.

3.2 Semi-autonomous Reinforcement Learning

We propose a semi-autonomous reinforcement learning technique. In semi-autonomous learning, a human operator teaches robots the best route to the target area. Robots share human decision-making through Q-values, which are renewed with each frame in the same way as in conventional Q-learning.

4 State Representation

To find an appropriate route to a target area within a navigable area, the robot must recognize its state and properly select the action. Recognition of the state is important; it depends on representation of the state space. We investigate the following state representation consisting of Fourier coefficients of distances to the target area, Fourier coefficients of distances to the boundaries of the navigable area, and the two-dimensional vector of apparent wind.

We define the relative direction, θ, with respect to the sailboat. We assume that agents can precisely measure their Euclidean distances to the target area $[d_+(\theta)|\theta = 0°, 1°, \ldots, 359°]$ and the boundaries of the navigable area $[d_-(\theta)|\theta = 0°, 1°, \ldots, 359°]$ for each degree. If a ray to angle θ does not cross the target area, $d_+(\theta)$ is set to ∞. The distance functions are transformed to 'nearness' functions, $\ell_+(\theta) = \left[e^{-d_+(\theta)}\right]$ and $\ell_-(\theta) = \left[e^{-d_-(\theta)}\right]$, respectively. Then, the periodic nearness functions are Fourier-transformed, respectively. The number of original nearness signals is 360 for each function. After Fourier transformation:

$$\ell(\theta) = a_0 + a_1 \cos\theta + b_1 \sin\theta + a_2 \cos 2\theta + b_2 \sin 2\theta + \cdots + a_N \cos N\theta + b_N \sin N\theta + \ldots,$$

$$a_0 = \frac{1}{360}\sum_{\theta=0^\circ}^{359^\circ} f(\theta)$$

$$a_k = \frac{1}{720}\sum_{\theta=0^\circ}^{359^\circ} \ell(\theta)\cos k\theta, b_k = \frac{1}{720}\sum_{\theta=0^\circ}^{359^\circ} \ell(\theta)\sin k\theta, k \in \{1, \ \ldots \ , N\},$$

we use $(2N+1)$ coefficients of N lowest wave numbers, i.e., $a_0, a_1, b_1, \ \ldots \ , a_N, b_N$, as state variables. In addition, we assume that the sailboat precisely measures the apparent wind. The x, y components of the apparent wind in the sailboat coordinate system are used as other state variables. We investigate the state space of $(2N_+ + 1) + (2N_- + 1) + 2$ dimensions in total, where N_+ and N_- are wave numbers for $\ell_+(\theta)$ and $\ell_-(\theta)$, respectively. We scale apparent wind and a_0 of both distances by 0.5, a_k and $b_k, k \in \{1, \ \ldots \ , N_+\}$ and $k \in \{1, \ \ldots \ , N_-\}$ by 2 to adjust Euclidean distance in the state space.

Figure 2 shows an example of sailboat state. The target area is from 45 degrees left to 45 degrees right in the front direction of the sailboat. The shortest distance is $d_+(0^\circ) = 300$. Also, the shortest distance to the boundary of navigable area is $d_-(270^\circ) = 300$. Figure 3 shows $d_+(\theta)$, $d_-(\theta)$, $\ell_+(\theta)$ and $\ell_-(\theta)$ respectively.

The continuous state space is partitioned to m regions. Before semi-autonomous learning, a human operator makes the sailboat travel in the navigable area and stores state data for each frame. Then, m state data are randomly picked up and used as generators of Voronoi division. While learning, the robot must discretize its continuous state vector. For each frame, the robot calculates a state vector and recognizes it as one of the m states corresponding to the nearest generator in the Euclidean distance.

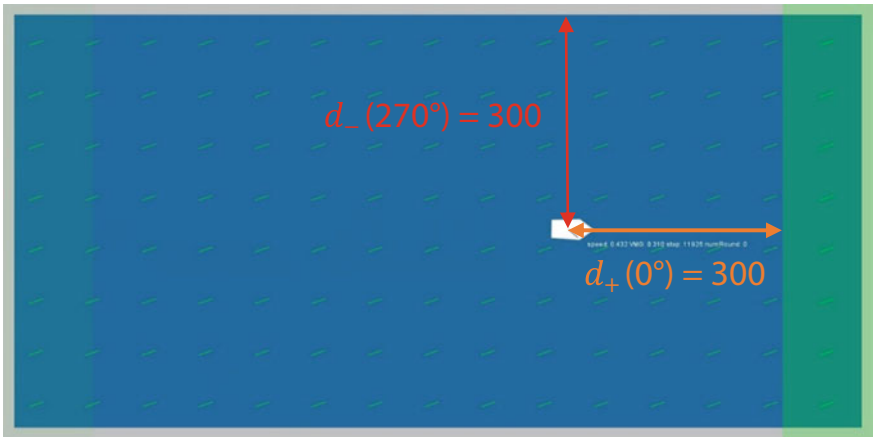

Fig. 2 An example of sailboat state

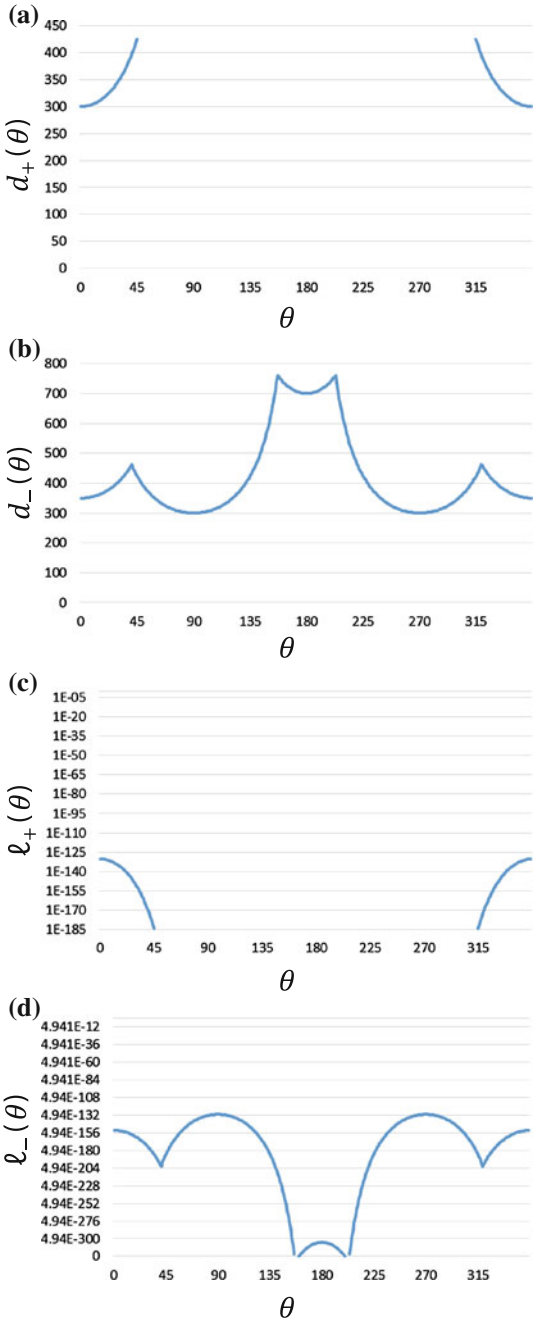

Fig. 3 The distances to boundaries of navigable area $d_+(\theta)$ and $d_-(\theta)$, the nearness $\ell_+(\theta)$ and $\ell_-(\theta)$ **a** The distance to target area $d_+(\theta)$ **b** The distance to boundaries of navigable area $d_-(\theta)$ **c** The nearness $\ell_+(\theta)$ **d** The nearness $\ell_-(\theta)$

5 Experiment and Results

We set the parameters of reinforcement learning as $\alpha = 0.001$ and $\gamma = 0.999$. A reward given to the agent is $r_t = 1,000$ when the agent reaches the target area. In addition, the reward is $r_t = -10,000$ when the agent touches a boundary of the navigable area; $r_t = -2,000$ when the agent stops after heading to the dead zone; and $r_t = -1$ for other frames. We set the wave numbers for Fourier transformation, $N_+ = N_- = 17$, so that the dimension of the state space becomes 72, which is one-tenth of 722, the number of originally measured variables. We execute the semi-autonomous learning procedure by changing $m = |S|$, the number of discretized state subspaces. The learning procedure is repeated twice for each $m = 10, 20, 50, 100, 200, 500$. The action set is $A = \{a|\Delta\theta = \pm 1°, 0°\}$; i.e., turning right or left, or staying straight. The action is selected by the mixture of ϵ-greedy and soft-max methods. The action with the highest Q-value is selected with the probability of $(1-\varepsilon)$; otherwise, the soft-max method with $\beta = 0.1$ is applied.

The green area on the right side of the display shown in Fig. 1—for example, the east side—is the target area through which the sailboat must traverse. When the sailboat enters the eastern target area, the western-most area becomes the next target area. The task for the robotic sailboat is to navigate through both target areas.

Before semi-autonomous learning, a human operator controls a sailboat until it finishes the first round. Then, the state space is partitioned, as described in the previous section. The other two robot sailboats appear in the display and start learning. Three sailboats share one set of Q-values. Therefore, the Q-values are renewed three times per frame. The human operator continues to control the sailboat for the first ten rounds. The average of elapsed steps per human-operated round is 1,424.8 (±36.1) steps. From that point in our procedure, the sailboat turns to 'anchor mode' for 500,000 steps. In anchor mode, the human-operated sailboat stops updating the Q-values, and the other two sailboats continue Q-learning. Then, the sailboat starts moving according to human operation for ten more rounds. This loop continues for up to approximately 3,000,000 steps (Fig. 4).

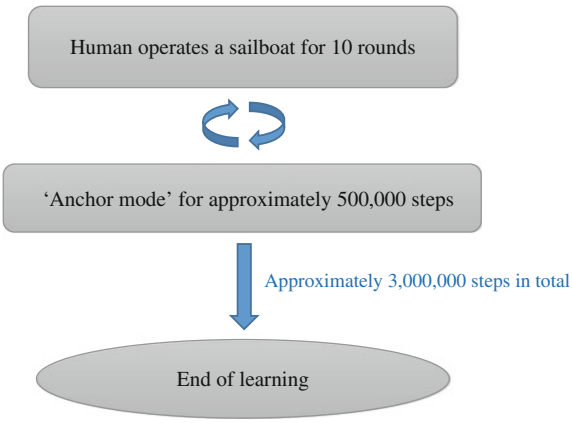

Fig. 4 Process of learning with the sailing simulator

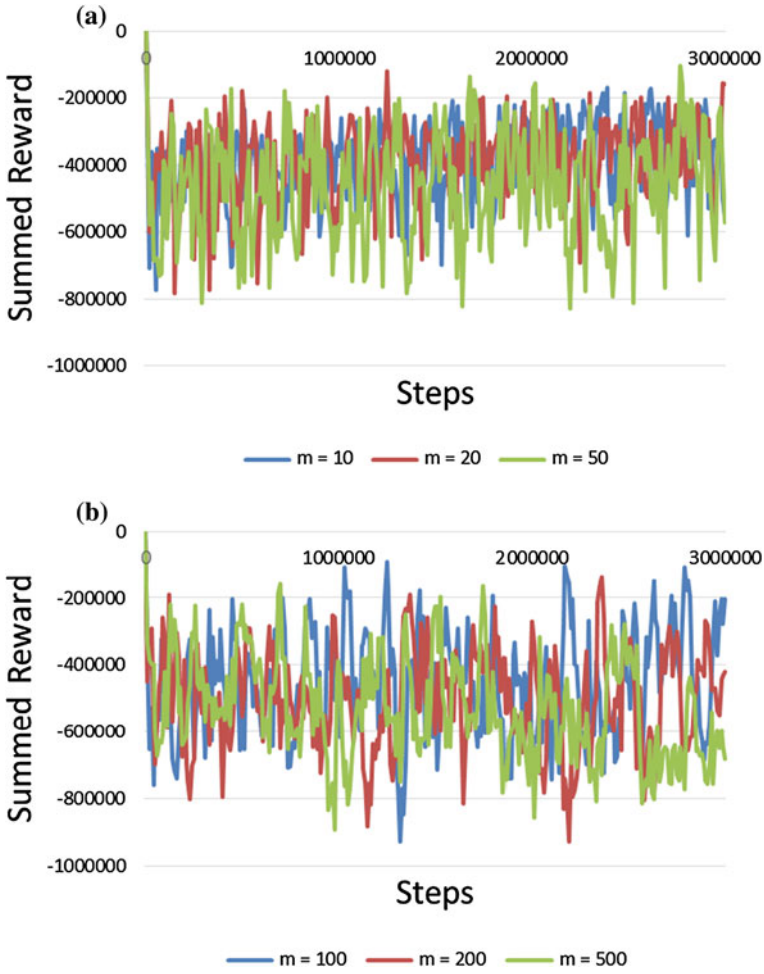

Fig. 5 Steps and summed reward **a** Reward in the case in which $m = 10, 20, 50$ **b** Reward in the case in which $m = 100, 200, 500$

Figure 5 shows the summed rewards in the cases where $m = 10, 20, 50$, and $m = 100, 200, 500$. 'Summed reward' means the summation that rewards the three sailboats, which are one's own sailboat and the agents' two sailboats 10,000 step. The X axis is the number of steps in the learning; the Y axis is the summed reward in the learning.

6 Discussion

Even after semi-autonomous learning, the robotic sailboats could not circumnavigate the target areas without touching a boundary of the navigable area in any case of m. Linear regression of summed reward y_{10} with step x_{10} was $y_{10} = 0.02580x_{10} - 435,126$ for the first experiment of $m = 10$, and $y_{10} = 0.06634x_{10} - 514,768$ for the second experiment of $m = 10$. In the summary of the two experiments, the average ± standard deviations of the slope and intersect were 0.04607 ± 0.02027 and $-474,947 \pm 39,821$, respectively. In the same way, the linear regressions of summed reward y_m with step x_m in cases of $m = 20, 50, 100, 200, 500$ were:

$$y_{10} = 0.04607(\pm 0.02027)x_{10} - 474,947(\pm 39,821)$$

$$y_{20} = 0.02899(\pm 0.00251)x_{20} - 430,284(\pm 99,984)$$

$$y_{50} = 0.01975(\pm 0.00019)x_{50} - 480,563(\pm 83,760),$$

$$y_{100} = 0.02331(\pm 0.02868)x_{100} - 489,586(\pm 73,007)$$

$$y_{200} = -0.00047(\pm 0.03219)x_{200} - 500,849(\pm 78,099)$$

$$y_{500} = -0.06096(\pm 0.02676)x_{500} - 440,724(\pm 80,421)$$

Figure 6 shows the relationship between m and slope.

The results show that the increase of summed rewards during learning was greater with a smaller m. In the case of $m = 500$, the summed reward decreased during learning in both experiments. It may occur that a cluster in the state space is

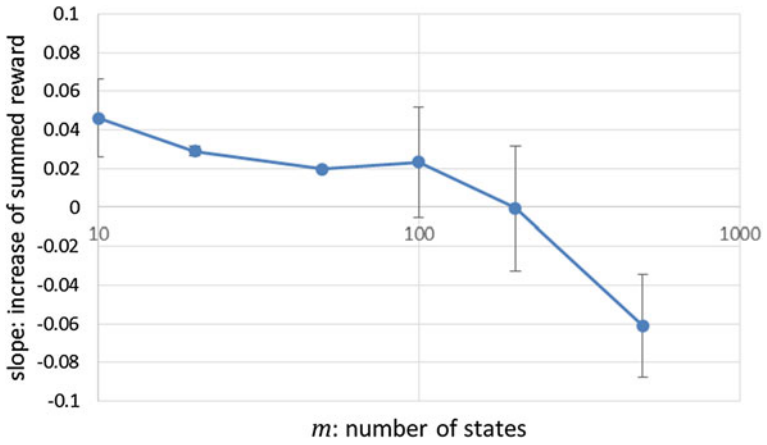

Fig. 6 Graph of approximations to steps and all rewards in Fig. 5

subdivided to too many partitions in the cases of larger m. If similar states are discretized to different state subspace and recognized as different discrete state, more updates of Q-values are needed than appropriate number of partitions. We compare progress of learning by action selection entropy between the cases of $m = 10$ and $m = 500$.

We calculate action selection entropy for each state. If action selection entropy is small for a state, a robot always selects the same action in the state. The minimum of action selection entropy is 0, where the probability of an action is selected is 1 and other actions are not selected. And if action selection entropy is large, a robot selects actions at random. The maximum is $\log_2 3$ in our case, where each of three possible actions is selected with the probability of one third. Action selection entropy for a state s is:

$$H(s) = -\sum_{a \in A} P(a|s),$$

$$P(a|s) = \frac{\exp\left(\beta \hat{Q}(s,a)\right)}{\sum_{a' \in A} \exp\left(\beta \hat{Q}(s,a')\right)},$$

We denote the set of states chosen for the experiment of $m = 10$ as S_{10}. We calculate action selection entropy for each state in S_{10} to find the state s^* with the minimal entropy after the learning procedure. Then, we find subset of states:

$$S^*_{500} = \left\{ s \in S_{500} | s^* = \underset{s' \in S_{10}}{\operatorname{argmin}}\, d(s, s') \right\},$$

here S_{500} is the set of states chosen in the experiment of $m = 500$, and $d(s, s')$ is the Euclidean distance between states s and s'. The subset S^*_{500} consists of states that are in the Voronoi region of s^*. Action selection entropy is calculated for each state of S^*_{500}.

Four combinations are evaluated, i.e. s^* is found for each experiment of $m = 10$, and S^*_{500} is detected for each experiment of $m = 500$. $H(s^*)$ was 1.200 for the first experiment of $m = 10$. $\left|S^*_{500}\right|$ was 64 and 76 for the first and the second experiments, respectively. $H(s^*)$ was 1.268 for the second experiment of $m = 10$. $\left|S^*_{500}\right|$ was 76 and 76 for the first and the second experiments, respectively. Figure 7 shows histograms of entropy in S^*_{500} for each experiment. $H(s^*)$ are shown by red arrows for each experiment. Majority of S^*_{500} have larger entropy than s^* and action is selected almost at random in such states.

However, the robotic sailboats learned the action of avoiding the obstacles. For example, we showed state 39 as one of 100 states. The red dot in Fig. 8 denotes where the sailboat recognized its state as state 39. The agent selected the action to turn left in state 39. Figure 9 shows the distance to the goal area and to the boundaries of the navigable area restored by inverse Fourier transformation from the state vector of state 39. The solid and dotted lines show restored $\ell_+(\theta)$ and $\ell_-(\theta)$, respectively. The direction of apparent wind is shown by the arrow.

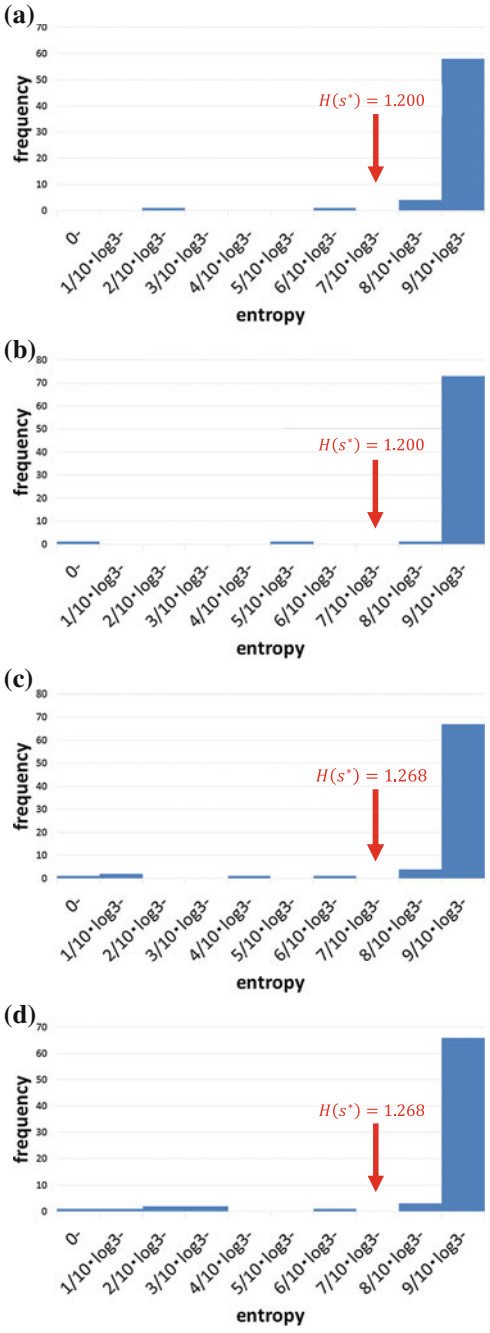

Fig. 7 Histograms of entropy **a** Histogram of entropy in S^*_{500} for the first experiment and $H(s^*)$ for the first experiment **b** Histogram of entropy in S^*_{500} for the second experiment and $H(s^*)$ for the first experiment **c** Histogram of entropy in S^*_{500} for the first experiment and $H(s^*)$ for the second experiment **d** Histogram of entropy in S^*_{500} for the second experiment and $H(s^*)$ for the second experiment

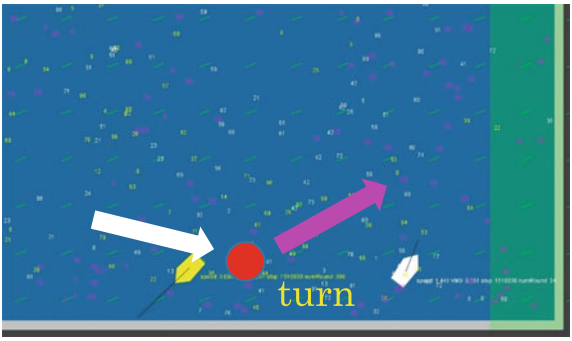

Fig. 8 Screenshot in state 39 at the number of 100 state spaces

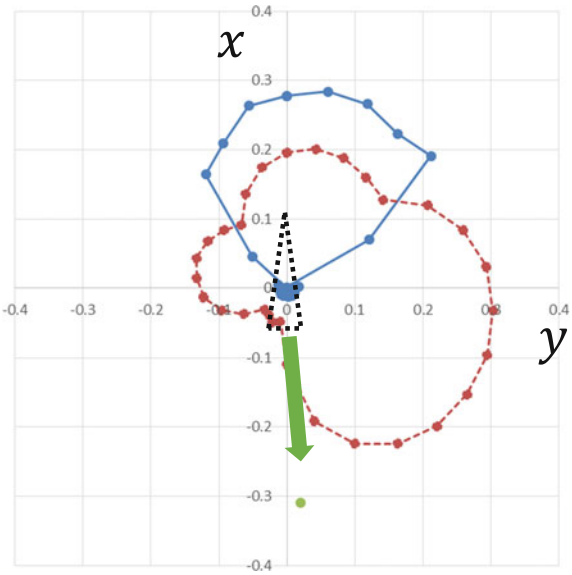

Fig. 9 Value of $\ell_+(\theta)$ and $\ell_-(\theta)$ and wind direction at state 39

The target area was in the forward direction of the sailboats. The nearest boundary of the navigable area was on the right side of the sailboats. The direction of apparent wind was forward and somewhat left in state 39. In addition, the goal area was located ahead or somewhat to the left, and the boundaries of the navigable area were located on the right of the sailboats. The direction of wind was forward in state 39 of Fig. 6; the state was certainly recognized in the results. The Q-values at state 39 were -711.9 (turn left), -717.9 (go straight), and -720.0 (turn right); therefore, the sailboats certainly learned to avoid the boundary by turning left. From this result, we can see that the Fourier coefficients of $N_- = 17$ of the lowest wave numbers stored sufficient information to avoid touching the boundaries.

7 Conclusion

In this paper, we described previous research of reinforcement learning for control in continuous state space in introduction. We explained our sailing simulator, and proposed a semi-autonomous reinforcement learning technique. We did the experiments at each number of divisions of the state space. As the result, the sailboat did not navigate to the goal area in these experiments. Nevertheless, the agent employed the action that avoided the boundaries of the navigable area when they were approached. We calculated the value of $\ell_+(\theta)$ and $\ell_-(\theta)$ to use inverse Fourier transformation from Fourier coefficients. We confirmed that the states were certainly recognized.

The state representations were calculated as the paths of the sailboats first approaching the navigable area, and they were chosen at random. In this division method, multiple divisions of state spaces were assigned to a cluster or, inversely, the division of state spaces was assigned to multiple clusters. It likely occurred that very similar state vectors—i.e., in the same cluster in the state space—were recognized as states that differ from each other, or, inversely, that very different state vectors—i.e., belonging to different clusters in the state space—were recognized as the same state. In other words, the recognized state was not a one-to-one correspondence with the cluster structure in the state space. Therefore, in future work, we will apply *k*-means clustering to the preparation phase for learning for the robotic sailboats to accurately circumnavigate the target areas within the navigable area.

References

1. Konidaris G, Barto A (2008) Skill discovery in continuous reinforcement learning domains using skill chaining, advances in neural information processing systems 22 (NIPS 2009), Computer Science Department, University of Massachusetts Amherst
2. Kuhl FP, Giardina CR (1982) Elliptic fourier features of a closed contour, Fairleigh Dickinson University, U.S. Army Armament Research and Development Command. In: Computer graphics and image processing, vol 18, pp 236–258
3. Ross S, Gordon GJ, Bagnell JA (2011) A reduction of imitation learning and structured prediction to no-regret online learning. In: Appearing in proceedings of the 14th international conference on artificial intelligence and statistics (AISTATS), JMLR: W&CP 15, vol 15. Fort Lauderdale, FL, USA
4. Shon AP, Verma D, Rao RPN (2007) Active imitation learning, AAAI conference on artificial intelligence, Department of Computer Science and Engineering, pp 756–762
5. Sterne PJ (2004) Reinforcement sailing [master's thesis], Master of science. School of Informatics, University of Edinburgh
6. Sutton RS, Barto AG (1998) Reinforcement learning, a bradford book. The MIT Press, Cambridge