IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 22 February 2023, accepted 12 March 2023, date of publication 16 March 2023, date of current version 21 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3257849

RESEARCH ARTICLE

Deep Reinforcement Learning-Based Air-to-Air Combat Maneuver Generation in a Realistic Environment

JUNG HO BAE, HOSEONG JUNG, SEOGBONG KIM, SUNGHO KIM, AND YONG-DUK KIM
Agency for Defense Development, Daejeon 34186, Republic of Korea
Corresponding author: Jung Ho Bae (deawith@gmail.com)

ABSTRACT Artificial intelligence is becoming increasingly important in the air combat domain. Most air combat research now assumes that all aircraft information is known. In practical applications, however, some aircraft information, such as their position, attitude, velocity, etc., can be incorrect or impossible to obtain due to realistic limitations and sensor errors. In this paper, we propose a deep reinforcement learning-based framework for developing a model capable of performing within visual range (WVR) air-to-air combat under the conditions of a partially observable Markov decision process (POMDP) with insufficient information. To deal robustly with such a situation, we use recurrent neural networks and apply a soft actor-critic (SAC) algorithm to cope effectively with realistic limitations and sensor errors. Additionally, to raise the efficiency and effectiveness of learning, we apply the curriculum learning technique to restrict the scope of exploration in state space. Finally, simulations and experiments show that the proposed techniques can deal with practical problems caused by sensor limitations and errors in a noisy environment while also being efficient and effective in reducing the training time for learning.

INDEX TERMS Air-to-air combat, limitation and error of sensors, recurrent neural network, reinforcement learning, soft actor-critic.

I. INTRODUCTION

The necessity of unmanned combat air vehicles (UCAVs) in various countries is increasing with the development of artificial intelligence (AI), integrated sensors, and communication technologies. Hence, the importance of the air combat model, which plays a key role in UCAVs, is increasing. Future warfare is shifting to ''mosaic warfare,'' which utilizes low-cost unmanned aerial vehicles (UAVs). For example, if the classical manned fighting formation can be improved into a manned-unmanned complex fighting formation, each well-trained human pilot can operate the command fighter while simultaneously controlling several unmanned fighters operated by the combat models. Thus, well-trained human pilots can operate several unmanned fighters, including their ownship (which refers to one's own aircraft), eventually contributing to the effective operation of a fighting formation. Hence, the Defense Advanced Research Projects Agency (DARPA) is developing a human-level air combat model to advance and build trust for air combat through the Air Combat Evolution (ACE) program.

Various kinds of research to develop an air combat model have been performed using minmax tree search, behavior tree, a virtual pursuit point technique based on a genetic fuzzy tree, etc. [1], [2], [3]. However, there are various kinds of limitations in the case of such rule-based air combat algorithms. In the functional aspect, aircraft characteristics such as rules and gain should be defined to develop an air combat model that can perform any specified task. In the performance aspect, developing the rules for optimal performance in a high-difficulty mission such as air combat is challenging because such rules are based on the appropriate hypothesis of the algorithm and the expert knowledge of the developer.

Recently, with the development of deep reinforcement learning (DRL), research using this has been applied to various challenging fields and has resulted in significant

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Liu.

successes, such as AlphaGo [4], AlphaStar [5], and Falco [6]. In particular, several studies use DRL to generate aircraft maneuvers for air combat [7], [8], [9]. In some cases, some studies simplify the complex air combat environment and then learn the maneuver to avoid obstacles using reinforcement learning [10]. In contrast to this approach, our work aims to develop a model that is good enough to chase and shoot down the fighter, not to avoid obstacles, at the same level of mechanical properties and control input/output of the fighter in a within visual range (WVR) engagement environment.

Most aircraft maneuver generation for air combat research assumes that all information about the ownship and target is known [11], [12]. However, in practical applications, some information, such as the position, attitude, and velocity of the ownship and target, can be incorrect because of sensor performance limitations, errors, and weather conditions. Furthermore, because of the failure to pursue the opponent, it is sometimes impossible to obtain information about the target. Therefore, to apply the air combat model to the actual aircraft, it should be necessary to consider the limitations and errors of the integrated sensors with which the aircraft is equipped.

Some studies have been conducted on sensor errors [13]. The 3-dimensional environment was considered; however, the fidelity of the control model remains at the level of kinematics and does not reflect practical characteristics, such as sensor limitations and errors, by adapting the methodology to consider noise in the state value.

In this paper, a DRL framework for developing an air combat model with a high-fidelity physical model capable of performing air-to-air combat under partially observable Markov decision process (POMDP) environments while considering the limitations and errors of sensors is proposed. We consider two kinds of sensors: radar and visual sensors. We assume that the radar system can detect the opponent precisely in the forward direction of the ownship, and the visual sensor can detect the opponent in all directions in a close combat situation. To robustly deal with the situation where there is no information about the opponent because it exists beyond the scope of sensors, we utilize long short-term memory (LSTM) and apply the soft actor-critic (SAC) [14] algorithm to cope with sensor errors effectively. To evaluate our proposed combat model, we call it SAC-LSTM, which follows the network architecture mentioned in Section IV, in contrast to SAC-FC, which only builds on fully connected layers.

Moreover, to increase the efficiency and effectiveness of learning, we apply the turn circle-based curriculum learning technique, which restricts the scope of exploration in state space and is based on the idea that humans learn gradually from simple to complex concepts. It has been proven to be effective in reducing training time. In addition, it induces feedback of a reward signal in an environment where the reward function is rare, helping to balance between exploitation and exploration.

Finally, various simulations and experiments show that the proposed algorithms can deal with practical problems caused by sensor limitations and errors while also being efficient and effective at learning.

In summary, our paper provides the following contributions:

- The introduction of a novel air combat maneuver generation framework. The model generated using the framework effectively performs air combat in a WVR engagement environment that considers the limitations and errors of sensors.
- An evaluation of SAC-FC and SAC-LSTM in an engaging environment that takes account of the limitations and errors of sensors. Experiments confirmed that SAC-LSTM has better performance.
- Validation of our proposed turn circle-based curriculum. Models learned using the curriculum recorded higher performance and were learned faster.

The rest of the sections of the paper are organized as follows: In Section II, backgrounds are explained, and the air combat simulation environment is described in Section III. The air combat DRL framework under POMDP circumstances is proposed in Section IV. Section V describes the experimental training and test results for the performance of the combat model through simulation analyses. Finally, Section VI discusses the results and describes further work.

II. BACKGROUNDS

This section describes the background information required for this study.

A. REINFORCEMENT LEARNING

Markov decision process (MDP) refers to a sequential decision process to satisfy the property of the state function of the next step, which depends only on the current state and action and is affected by the reward function [15]. According to the standard notation, the MDP can be expressed as a 4-tuple $< S, A, P, R >$. Each component means a state $s \in S$, an action $a \in A$, the transition probability $P(s'|s, a)$ when an action is performed in the state, the probability of moving on to the next state, and the reward function $r(s, a)$.

Reinforcement learning (RL) refers to the process in which an agent performs an action in a given MDP environment based on the trial-and-error learning process and receives a reward based on the performance of the action [16]. Algorithms for learning agents are divided into on-policy and off-policy algorithms. In on-policy algorithms, policy functions are learned directly; in off-policy algorithms, they are not, and most of them are learned by estimating action-value functions (the Q Function). The optimal Q Function $Q^*(s, a)$ for the current state and action is calculated as the expectation value of the sum of the optimal Q Function considering the action a' obtained by reward $r(s, a)$, next state s', and the transition probability distribution P. In this case, the max function of the Bellman equation cannot be dealt with in a problem related

to the control of continuous space. To solve this problem, the actor-critic structure is designed to learn Q functions through the critic network and policy functions through the actor network simultaneously.

The SAC is known as the state-of-the-art actor-critic structured reinforcement learning algorithm. The deep deterministic policy gradient (DDPG) is a deep reinforcement learning algorithm that deals with control in continuous space. However, this algorithm has a problem of overestimation specific to the actor-critic structure [17]. Twin delayed DDPG (TD3) introduces two critic networks, Q_1 and Q_2, and alleviates the above-mentioned problem by bootstrapping the Q value with an algorithm that selects the minimum value among them [18]. However, the above two algorithms make it difficult to challenge control of continuous state and action spaces because the action value corresponding to the state value is mapped deterministically. To solve this problem, the action value needs to be estimated stochastically. Moreover, the SAC algorithm adapted the objective function to include a term that induces entropy to be maximized, as shown in (1).

$$J(\pi) = \sum_{t=0}^{T} E_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t) + \alpha H(\pi(a_t|s_t))], \quad (1)$$

where H is the desired minimum expected entropy and α is the temperature function.

If the learning proceeds by adding terms that maximize entropy, it was confirmed that the agent could become better at exploration and stronger against noise [19]. In addition, it has the advantage of being able to learn several state–action pairs close to the optimal policy function. Therefore, SAC is known to create policies in the POMDP environment that can strongly respond to unobserved reward functions or adversarial variations in reward functions.

The goal of RL is to find the state-action pairs that produce the maximum values of the reward function. To achieve this, agents must strike a good balance between exploiting previously acquired knowledge and exploring state and action spaces they have not visited. However, if the environment gives rare feedback on the reward signal, the magnitude of the agent's update will become weak, making it difficult to induce effective learning. To solve this problem, several papers have created and applied curriculum using knowledge of the domain of RL, solving sparse reward problems, or leading to the stabilization of learning by balancing exploration and exploitation [20]. Since the state space and action space of the air combat environment used in our study are large, and the reward signal received by chasing an enemy fighter is rare, we designed the curriculum mentioned in Section IV to make learning effective.

B. REINFORCEMENT LEARNING UNDER POMDP ENVIRONMENT

In a real-world environment, it is rare for an agent to receive full information on the system state. Additionally, certain elements may affect the following states and multiple timesteps. In other words, the Markov property does not work well in this environment. Therefore, it is necessary to design a framework that applies a POMDP environment to provide only partial information about the state of the system [21]. According to the standard notation of POMDP, it can be represented as a 6-tuple $< S, A, P, R, \Omega, O >$. S, A, P, and R are the same state, action, transition, and reward functions as MDP. In the POMDP environment, the agent can no longer access the entire system state; instead, only the observation function $o \in \Omega$ is allowed to be accessed. This observation function is generated by the probability distribution $o \sim O(s)$ for the entire state function.

Various research studies have created a POMDP environment by imposing restrictions on the environment used in the existing MDP environment [22], [23], [24]. To make POMDPs, [25] proposed a method that makes states partially masked, [22] proposed to drop random frames, and [26] proposed to add random noise to states. Because of POMDP's non-stationary and suboptimal characteristics, it is difficult for agents to learn in a POMDP environment. Therefore, the recurrent neural network was adapted into a structure that can store information from a previous point in time in memory and utilize it to solve this limitation of POMDP.

Among the methods using recurrent layers, LSTM [27] and gated recurrent units (GRU) [28] are generally used. These methods increase the complexity of the network and simultaneously increase the error-proneness of DRL algorithms. Because of the nature of the LSTM, which controls the ability to remember or forget depending on the need for information, it has been utilized in environments requiring recurrence. Yang and Nguyen [29] compared performance by adding a vanilla recurrent neural network, LSTM, and GRU as recurrent modules to DDPG, TD3, and SAC. As a result, models built on LSTM and SAC showed the highest reliability and best performance.

The deep recurrent Q-network (DRQN) [22] extends the deep Q-network (DQN) algorithm and explains how to sample episodes and hidden states from the experience replay buffer to adapt the LSTM layer. Additionally, the recurrent replay distributed DQN (R2D2) [30] algorithm provides an approach to sampling sequences in the replay buffer, initializing hidden states, and solving the problem of blurring information over time. The recurrent deep policy gradient (RDPG) [25] manages the replay buffer efficiently and presents an approach to how the recurrent layer is utilized in continuous space. Meng et al. [26] implement an approach to attaching the LSTM layer to the TD3 algorithm as a method of concatenating the LSTM layer individually with the fully connected (FC) layer.

There is also an approach to utilizing Transformers for reinforcement learning [31], which has shown remarkable success in Natural Language Processing and Computer Vision [32]. However, it is unsuitable for our task because of the inherent computation complexity that extends quadratically to the length of the input sequence and the instability

that arises when applied to the problem of controlling continuous actions [33].

III. AIR COMBAT SIMULATION ENVIRONMENT

In this research, we have trained air combat agents in the F-16 engagement environment [34]. For the F-16 control physical model, a high-fidelity open-source aerodynamic model called JSBSim [35] is used. A Python class that provides the OpenAI Gym Env API wraps the simulation environment, which includes the two JSBSim models [36].

In the MDP environment, available features consist of JSBSim outputs, including simulation time and health points (HP) for both aircraft. The physical model output consists of position, posture, speed, acceleration, previous control values, aircraft information, etc. Control inputs are four continuous values that drive roll, pitch, rudder, and throttle. The physics engine is simulated at 60 Hz, while the learning input is limited to 10 Hz in consideration of the sensor, network, and mission computer performance in real-world operation. In other words, an agent's input values were equally input to the physical model six times.

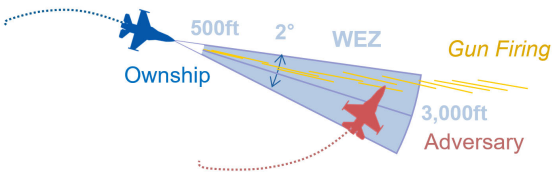


FIGURE 1. Weapon Engagement Zone.

We consider only a machine gun as a fighter-mounted weapon because the research goal is to train a combat model that satisfies advanced maneuvering performance, excluding the performance of sensors and/or weapons. Fig. 1 shows the weapon engagement zone (WEZ). The WEZ is defined as an effective range within 500 ft to 3,000 ft and an effective angle within 2° in the same way as previous research [34]. When the aircraft is located within the WEZ, HP is reduced in proportion to time according to equation (2) in consideration of the probability of being hit according to the distance.

$$d_{wez} = \begin{cases} 0 & r > 3000ft; \\ \dfrac{3000 - r}{2500} & 500ft \leq r \leq 3000ft; \\ 0 & r < 500ft, \end{cases} \tag{2}$$

where r is the distance between the ownship and the target. For example, damage at 500 ft is 1/s, and our system evaluates damage every 100 ms; thus, 0.1 damage is accumulated at every step.

There are three engagement termination conditions: first, when any aircraft is hit and its HP becomes 0; second, when the altitude of any aircraft drops below 1,000 ft; and third, when the engagement time exceeds 300 seconds. The victory conditions are when the enemy is shot down or goes down, or when the HP prevails after 300 seconds have elapsed. In the opposite case, it is judged a loss, and if the HP is the same after 300 seconds, it is judged a draw.

IV. AIR COMBAT REINFORCEMENT LEARNING FRAMEWORK

In this section, the overall learning environment and processes applied to learning aerial combat models based on reinforcement learning will be described in detail.

Figure 2 shows the overview of the air combat model learning framework proposed in this study, which consists of vectorized air combat simulation environments and a recurrent SAC module, including a replay buffer. The environment has two dynamic models: ownship and target. They get an action a_t from the actor of the SAC module and a_{target} from the rule-based behavior model, and they output the aircraft states $s_{ownship}$ and s_{target}, respectively. The simulator generates a reward r_t and an observation o_t using the states, considering the configured sensor characteristics. The trajectories (o_t, a_t, r_t) are stored in the replay buffer, and fixed-length sequences of trajectories are sampled for the critic.

A. OBSERVATION AND ACTION SPACES

The design of the state space is important in terms of learning efficiency and effectiveness. In the vision-based reinforcement learning used in Atari games, DM-30, etc., the network parameters that extract key features from downscaled screen images are trained simultaneously with vision processing models such as convolutional neural networks (CNN) [37]. However, in the case of air-to-air engagement, it is not necessary to consider complex surrounding environment information because it is a wide 3D space of up to 20 km or more. On the contrary, it may be effective to use the feature values preprocessed through various sensors.

The JSBSim for each aircraft outputs features such as position, attitude, and velocity in the global coordinate system. The ownship information at timestep t includes altitude (al_t), attitude $(\psi_t, \theta_t, \phi_t)$, speed $(\vec{V}_t = [u_t, v_t, w_t])$, acceleration $(\vec{A}_t = [Ax_t, Ay_t, Az_t])$, remaining fuel (f_t), previous four control inputs (a_{t-1}), and ownship HP (oHP_t):

$$o_{ownship,t} = [al_t, \psi_t, \theta_t, \phi_t, \vec{V}_t, \vec{A}_t, f_t, a_{t-1}, oHP_t]. \tag{3}$$

The relative target information at timestep t includes aspect angle (AA_t), antenna train angle (ATA_t), heading crossing angle (HCA_t), relative distance (d_t), relative speed $(d\vec{V}_t = [du_t, dv_t, dw_t])$, relative acceleration $(d\vec{A}_t = [dAx_t, dAy_t, dAz_t])$, and target HP (tHP_t):

$$o_{rel,t} = [AA_t, ATA_t, HCA_t, d_t, d\vec{V}_t, d\vec{A}_t, tHP_t]. \tag{4}$$

The learning network to convert the absolute coordinates of the two aircraft into relative coordinates can be considered; however, this can cause a significant decrease in learning efficiency. In this study, the basic geometry information was converted to the relative coordinate of the ownship based on the combat manual. Fig. 3 shows the basic geometry between two aircraft and indicates components of relative target information. According to the geometry, we constructed

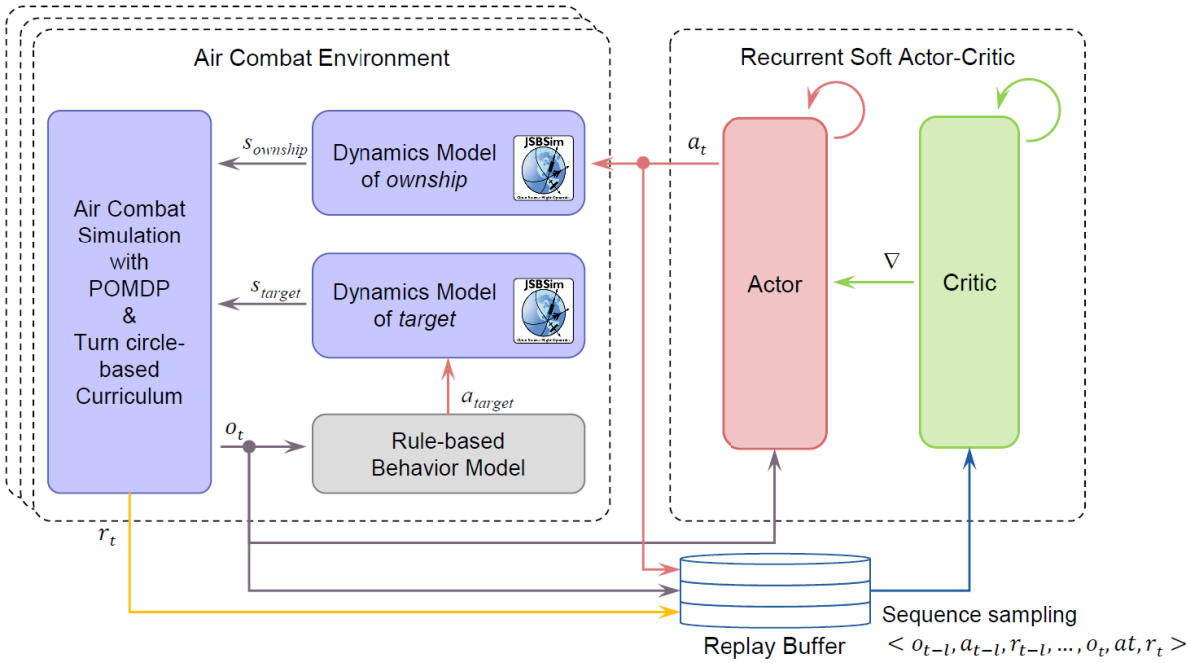


FIGURE 2. Air combat model learning framework.

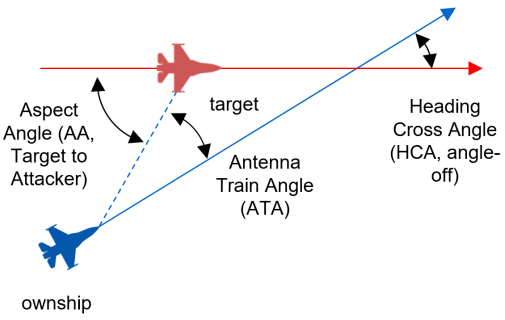


FIGURE 3. Geometry of fighters.

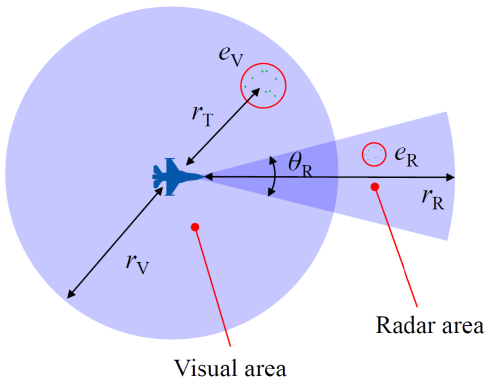


FIGURE 4. POMDP environment design.

the observation by concatenating ownship information and relative target information:

$$o_t = [o_{ownship,t} || o_{rel,t}], \tag{5}$$

where || is the concatenation operator. The action space is the maneuvering commands of the aircraft:

$$a_t = [Aileron_t, Elevator_t, Rudder_t, Throttle_t]. \tag{6}$$

Note that, in a POMDP situation, if the target is out of the sensor's range, data related to the target will not be provided.

B. POMDP ENVIRONMENT DESIGN

In a real-world engagement, it is difficult to accurately measure target information, such as position, posture, and speed, due to sensor errors. In addition, when the target is beyond the detection area, there is also a situation in which the target's position must be predicted. We designed a POMDP WVR engagement environment to reflect these real-world characteristics of sensors.

Figure 4 illustrates the POMDP environment with two types of sensors: a visual sensor capable of detecting short distances, such as a camera, and a radar sensor capable of detecting relatively narrow but long-range signals in the forward direction. r_V and e_V represent the detection range and error rates of the visual sensor, and r_R, θ_R, and e_R represent the detection distance, angle, and error rate of the radar sensor, respectively. The detection ranges of the visual and radar sensors were set to vary between values known from the sensor's specification and assuming more severe conditions. When a target is detected in the detection area, an error proportional to the target's range r_T is injected into the target-related observation. When the detection areas of the

two sensors overlap, a smaller error is used. If there is a target beyond the detection area, the target-related information is set to 0.

C. REWARD FUNCTION

The sparse reward problem is the major performance degradation factor in reinforcement learning. We additionally designed dense rewards as guides to achieve the goals efficiently. The reward functions applied in this study can be divided into five types: shooting down, WEZ, control zone, Crash, and control stabilization. The explanations for each are as follows:

1) SHOOTING DOWN

Shooting down reward function $R_{shooting\_down}(HP_{ownship}, HP_{target})$ rewards +500 if ownship shoots down the target, and -500 if ownship shot down.

2) WEZ

WEZ rewards induce the target to enter the ownship's WEZ and the ownship not to enter the target's WEZ. We defined two WEZ rewards: $R_{shoot}(range, ATA)$ is the effective shooting reward, and $R_{beHit}(range, AA)$ is a penalty when the target shoots the ownship. The agent gets rewards for every step according to equations (7) and (8).

$$R_{shoot} = \begin{cases} (5+5\times\dfrac{3000-r}{2500}) & r<3,000ft \text{ and } ATA \le 1^\circ; \\ 0 & otherwise. \end{cases} \tag{7}$$

$$R_{beHit} = \begin{cases} -5 & r<3,000ft \text{ and } AA \ge 179^\circ; \\ 0 & otherwise. \end{cases} \tag{8}$$

Because this shooting reward is sparse and difficult to get in early learning phases, we added another continuous WEZ relative reward, $R_{wez\_dot}(range, \Delta wez\_to\_target) \propto \frac{\Delta wez\_to\_target}{range}$ to be rewarded every step. There are two features of R_{wez_dot}. The first is to adjust the amount of reward received adaptively according to the distance by using a single function, and the second is to use the distance between the WEZ and the target instead of the distance between the aircraft bodies. In many studies related to WVR engagement, distance and/or ATA reduction rates are used for WEZ induction. In our best experience, when the rewards are composed only of these items, the ''overshoot'' problem continuously occurs. We solved the minimum and maximum WEZ range and overshoot issues simultaneously using this reward function.

3) CONTROL ZONE

The reward functions relative to the control zone consist of two sparse and dense rewards for occupying the target's tail and gaining the upper hand. Fig. 5 shows the process for attaining control zone rewards.

The two sparse rewards are $R_{CZ\_ATA}(ATA, range)$, which is intended to allow the ownship to look at the target, and

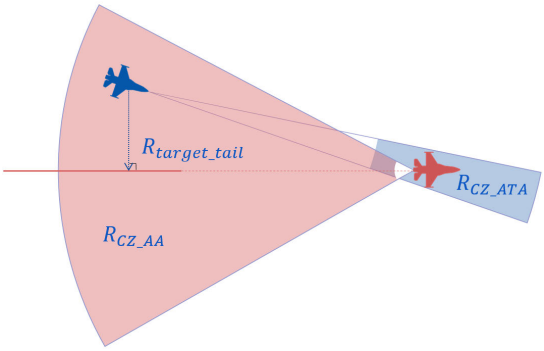


FIGURE 5. Control zone rewards.

$R_{CZ\_AA}(AA, range)$ to catch the target's tail. These rewards are designed such that the agent can receive them as follows:

$$R_{CZ\_ATA} = \begin{cases} 0.1 & ATA \le 1^\circ \text{ and } 3,000ft \le range \le 5,000ft; \\ 0. & \text{otherwise}, \end{cases} \tag{9}$$

$$R_{CZ\_AA} = \begin{cases} 0.1 & AA \le 15^\circ \text{ and } 500ft \le range \le 5,000ft; \\ 0. & \text{otherwise}. \end{cases} \tag{10}$$

Dense rewards for attaining control zone consisted of $R_{delta\_angle}(\Delta ATA, \Delta AA)$ for preempting an advantageous position compared to the target and $R_{target\_tail}(range, \Delta target\_tail\_to\_ownship)$ for biting the target's tail. R_{delta_angle} compares the ATA reduction rate with the AA reduction rate at every step and gives a reward if the ATA reduction rate is greater and a penalty if it is vice versa. A virtual stick tail was created from 3,000 ft to 5,000 ft behind the target to bite the target's tail. Additionally, if the ownship was closer to or farther from the tail, rewards or penalties, depending on the distance, were given, respectively.

4) CRASH

We also configure the reward functions for the fall with sparse and dense rewards. There are two sparse rewards: 1,000 points as a penalty when the ownship falls and 10 rewards when the target crashes. The target falling down reward is set to be small because the agent can get rewards easily regardless of their maneuvering performance if the target has low control performance. The dense reward for fall prevention is designed such that when the ownship is below 2,000 ft in altitude, the closer the aircraft gets to 1,000 ft, the greater the penalties.

5) CONTROL STABILIZATION

In the consideration of the application for the actual aircraft, the reward function of control value stabilization is applied to minimize the mechanical or electronic load of the actual aircraft. When considering the dynamic characteristics of fighters, the greater the change in the roll command, the greater the penalty becomes.

D. NETWORK ARCHITECTURE

In this study, there are two major problems to be solved by applying recurrent networks. The first is a multi-task problem. The feedforward network infers only the fragmentary observation values at the current step. Thus, it may not be possible to know the various patterns that can be known only by looking at the previous trajectory. The second is the POMDP problem. In an actual combat situation, due to the performance limitations of onboard sensors such as radar, the target information may not be obtainable or errors may be included in the measured information. We tried to solve these problems by learning to memorize important observed patterns among previously received observations through the recurrent network.

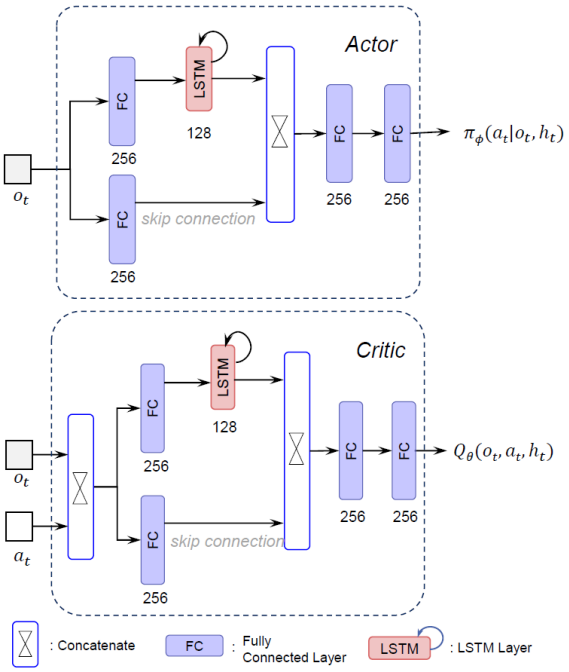


FIGURE 6. Network architecture.

A schematic illustration of the policy and value networks is shown in Fig. 6. The actor network receives an observation o_t and outputs the next action a_t at time t. a_t and o_t are input to the critic network to obtain a Q-value. At this time, the internal memory of each network is updated at every step. Each network consists of a recurrent branch in charge of internal memory and a feedforward branch. The former branch is modeled as a layer of LSTM units to analyze maneuver patterns and cope with situations where the target was missed.

E. TURN CIRCLE-BASED CURRICULUM LEARNING

We applied a curriculum learning technique to improve learning performance. In deep reinforcement learning, effective learning can be performed by successively approximating the state space using deep neural networks. However, the process for finding effective maneuvers against a target fighter with high-level maneuvers in the nearly infinite observation and action spaces does not converge despite applying the naive deep reinforcement learning technique. In this study, the observation space at the early stages is limited such that the agent can learn the maneuver to shoot down the target within a limited time. The learning proceeds step by step, gradually releasing the limit of the observation space as the learning progresses. For the agent to experience more diverse observations, the curriculum consists of two steps: 1) difficulty level determination; and 2) formation determination.

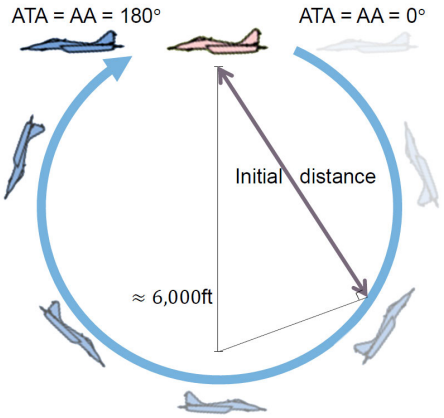


FIGURE 7. Turn circle-based curriculum learning.

1) DIFFICULTY LEVEL DETERMINATION

The difficulty of engagement can be defined by how easily the agent can shoot down the target, and we determined it by using the attitude between the two fighters, especially ATA and AA.

Figure 7 visually shows the proposed turn circle-based curriculum learning. At first, ATA and AA were started at 0°, and when the success conditions were satisfied, the two angles were increased, and finally, when they rotated to 180°, learning was terminated.

Algorithm 1 Turn Circle-Based Curriculum Learning

for angle $\alpha \leftarrow 0°$ **to** $180°$ **do**
  if $15° \leq \alpha \leq 165°$ **then**
    range $r \leftarrow$ 6,000ft $\times\ cos(90° - \alpha)+$ **noise**(0∼1,000ft)
  else $r \leftarrow$ **random**(500ft 1,000ft)
    Set environment aircrafts with $(ATA \leftarrow \alpha, AA \leftarrow \alpha, r)$
  end if
  if $\alpha \leq 30°$ **then**
    Add environment done condition $(AA > 90°)$
  end if
  $win\_ratio \leftarrow 0\%$
  while $win\_ratio < 70\%$ **do**
    Collect set of trajectories D_k with max sequence length
    Replay buffer $D \leftarrow D \cup D_k$
    Update the networks using D with the recurrent SAC.
    Update win_ratio with 30 episodes.
  end while
end for

Algorithm 1 summarizes the curriculum procedure. A typical fourth-generation fighter is known to form a turn circle

with a maximum maneuverability of 6,000 ft. We designed the curriculum such that the agent sequentially learns how to win in the most difficult defensive basic fighter maneuvers (BFM) formation with ATA and AA of 180° starting with the least difficult offensive BFM formation where the agent has ATA and AA of 0°. The observation space is limited to improve learning performance. At the beginning of learning, when the angle is less than or equal to 30°, the episode ends and starts anew if the ownship is located in front of the target's 3/9 line.

2) FORMATION DETERMINATION

When the distance and angle are determined in the stage of difficulty level determination, the detailed positions and postures of fighters are taken to enable the agent to explore the observation space to the extent possible. The JSBSim dynamic model is also modeled to consider the influence of the environment, such as gravity and altitude. Therefore, even when the two fighters have the same ATA and AA, the difficulty and maneuvering performance can be different depending on whether they are level on the ground, ascending, or descending. In consideration of this, one of the three formations of forward, upside, and downside is arbitrarily determined as the initial formation, as shown in Fig. 8. Furthermore, even if ATA and AA are the same, rolls can be set differently. We arbitrarily set the rolls of the two fighters to allow them to start in all possible postures, each corresponding to a specific ATA and AA.

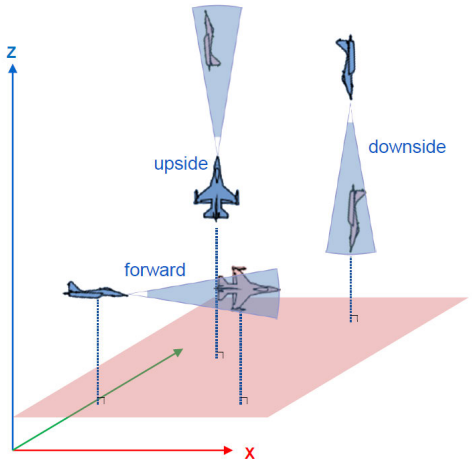


FIGURE 8. Formation determination.

3) TARGET MANEUVERING

The maneuvering performance of the target is important because it becomes the baseline of the learning agent's performance. We adopted a rule-based aerial combat model composed of decision and guidance modules as a target model [38]. Fig. 9 depicts the top-level maneuver decision flow chart developed by analyzing the BFM.

The decision module process input values based on the positional relationship between ownship and target, and

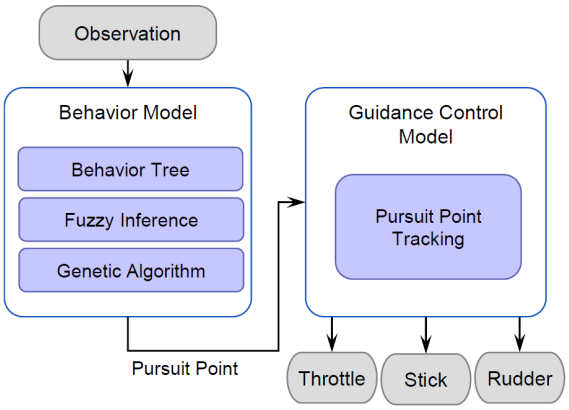


FIGURE 9. Target maneuver decision flow chart.

informs what kind of combat situation the current state is. Then, it determines the optimal maneuver using the behavior tree made based on the tactical instructors and generates a pursuit point. And the guidance module generates four inputs to move the aircraft to the generated pursuit point. Similar to previous studies, these modules were created using general flight control techniques [1], [3].

The performance of the rule-based air combat model has been verified through engagements with humans [39]. Two Air Force pilots played five simulation battles each, and the rule-based model won 10:0. Afterwards, vulnerabilities, including DBFM and altitude drop, discovered through the engagement with learning-based agents were supplemented. Currently, the performance has been upgraded, and the latest model shows a winning rate of 100% compared to the model that faced the human pilots.

V. EXPERIMENT

This section discusses the results of the learning progress and engagement performance of agents who have learned air combat maneuvers in the POMDP environment.

A. EXPERIMENTAL SETTINGS

In this experiment, we compare the learning effectiveness and engagement capabilities of three types of agents: SAC-FC learned with the SAC algorithm by applying only a 2-layer fully connected network, SAC-LSTM learned with the SAC algorithm by applying LSTM, and SAC-LSTM-no-curriculum excluded curriculum from SAC-LSTM. Table 1 shows the details about the values or ranges for parameters.

The specification of the machine used for learning is an Intel Xeon Silver 4210 (10 cores, 2.2 GHz) processor and Nvidia RTX 2080Ti GPUs. The results are confirmed by learning up to 5×10^5 updates per agent. The learning time for the two types of agents is similar, at about 21 days for SAC-FC agents and 18 days for SAC-LSTM agents. The number of samples used for one update is 64 times higher in SAC-LSTM than in SAC-FC. For a fairer comparison

TABLE 1. Hyperparameters.

Parameters	SAC-FC[a]	SAC-LSTM
Update interval	100 env steps	640 env steps
Minibatch size	256	64×256
Sequence length	-	64
Number of rollout workers	10	
Replay buffer size	1×10^6	
Discount factor	0.99	
Optimizer	Adam	
Optimizer settings	Actor/critic/entropy $\lambda = 3 \times 10^{-4}$	
r_V	[10, 5, 2, 1]km	
e_V	Gaussian Noise $N(\sigma = [0, 0.01, 0.1, 0.2])$[b]	
r_R	30km	
θ_R	60°	
e_R	Gaussian Noise $N(\sigma = [0, 0.015])$[b]	

[a] Network architecture of SAC-FC is simply two fully connected hidden states, FC(256)-FC(256)
[b] Mean value of the Gaussian Noise is set to 0 ($\mu = 0$)

between the two agents, additional experiments were conducted while increasing the minibatch size of SAC-FC, but the performance ended up being degraded. It is analyzed that the SAC-FC has a longer total learning time than the SAC-LSTM due to the greater number of times it is stored in the replay buffer. SAC-LSTM collects and stores every 64 steps, whereas SAC-FC stores them individually. The SAC-LSTM agents accumulated approximately 500 hours of flight experience per day and a total of 9,000 hours of flight experience over about 18 days. To maintain the target's performance, the rule-based model used as a target in learning is designed by considering the situation in which all information is known without error even in learning under the POMDP environment. It helps to distinguish whether the agent can learn to satisfy the engagement performance in a more severe engaging situation.

The experiment results were summarized by engaging each final learning agent versus the rule-based model in win-draw-lose counts for 300 matches under the learning environment. Winning rates were calculated using the following equation:

$$WinRatio = \frac{n_w}{n_w + n_d + n_l} \times 300, \tag{11}$$

where n_w, n_d, and n_l are the numbers of wins, draws, and losses, respectively. The engagement between the two aircraft was conducted by varying the heading angle of the fighter at the starting point with random altitude, distance, and speed, followed by a 300-second time-out. In detail, 100 matches were conducted in DBFM, OBFM, and neutral situations, respectively.

B. EXPERIMENTAL RESULTS

The experiment was conducted in two categories. First, the detection area of the visual sensor was gradually reduced; second, the noise of the visual sensor was gradually increased; and third, the existence of a turn circle-based curriculum was changed.

1) RANGE LIMITATION RESULTS

This experiment confirmed whether each agent was successfully trained when the range of the visual sensor was limited from 10 km to 5 km, 2 km, and 1 km. Fig. 10 shows the experimental results of the visual range limitation. The graphs show the achievement level of curriculum learning. The goal of the curriculum is to progress to the final 180°.

The results show that all SAC-LSTM models achieved the final goal within a limited time. This means that even when starting from the most unfavorable defensive BFM situation, where ATA and AA are 180° apart, all models with limited sensing range have a winning rate of over 70% against the rule-based air combat model that uses all information. In the cases of the 10 km and 5 km limits, it was confirmed that the angle continuously increased up to 180°. In WVR engagements, the range between two fighters is rarely greater than 10 km. Therefore, in the case of the 10 km limit, it can be seen that the situation is similar to the MDP situation. Also, considering that the F-16's turn circle is about 6 km in diameter, even if the sensor range is limited to 5 km, it will be possible to engage in combat without missing the target through selective maneuvering.

Conversely, in 2 km and 1 km experiments, the agents inevitably miss the target. In the experiment with a 2 km range constraint, it was confirmed that it progressed rapidly up to 180°, and in the case of the 1 km constraint, learning progressed slowly from 50°. However, it was found that the levels increased rapidly from about 3×10^5 updates, which indicated that the agent had acquired maneuvers to beat the rule-based model in DBFM situations.

The SAC-FC agents showed a winning rate of more than 70% in OBFM and neutral situations in 10 km and 5 km constraint experiments but did not progress the curriculum to 180° in a limited time.

Figure 11 shows a comparison of win-draw-lose counts obtained through the neutral engagement of the eight agents versus the rule-based model (RM) in each POMDP environment learned. All four agents learned with SAC-LSTM in all POMDP environments and obtained a win ratio higher than 80%. In particular, the agents trained in 10 km and 5 km visible environments obtained 87% and 85% win ratios, respectively. On the other hand, the SAC-FC agents achieved an average 55.2% win ratio. The three agents learned in 2 km or more visible environments got win ratios similar to or higher than 50%; however, the agent learned in the 1 km POMDP environment showed a 27.3% win ratio.

2) ERROR INJECTION RESULTS

In this experiment, we examine whether the proposed method can complete learning even for sensor values that include noise. To examine the effect of learning according to the noise value, POMDP environments were produced in which r_V was fixed at 5 km, which is a realistic visual range obtained from pilots, and e_V was set to $\sigma = 0.01$, 0.1, and 0.2, respectively. Fig. 12 shows the learning results of SAC-FC

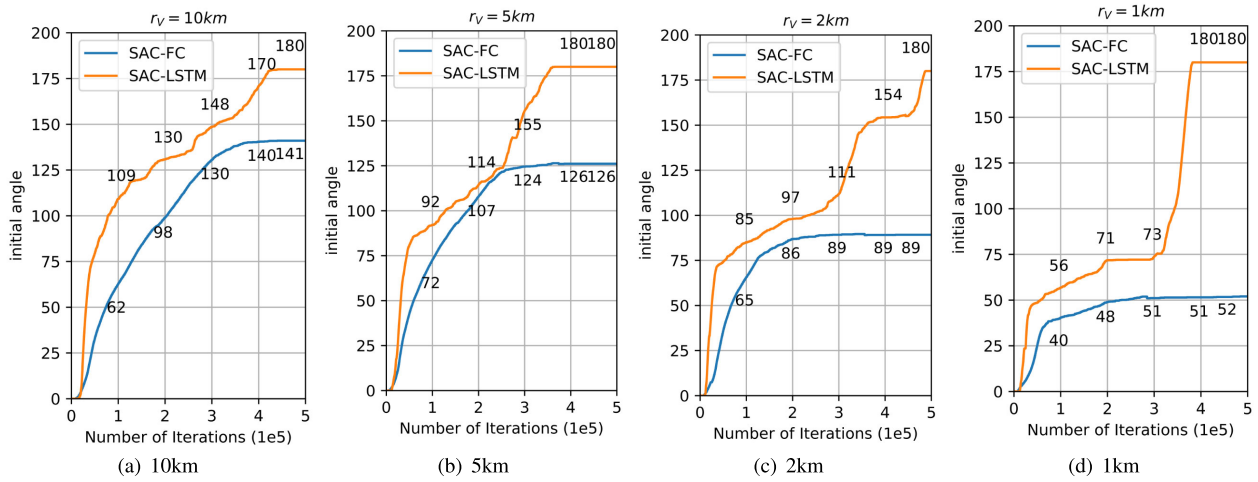


FIGURE 10. Range limitation experiment results.

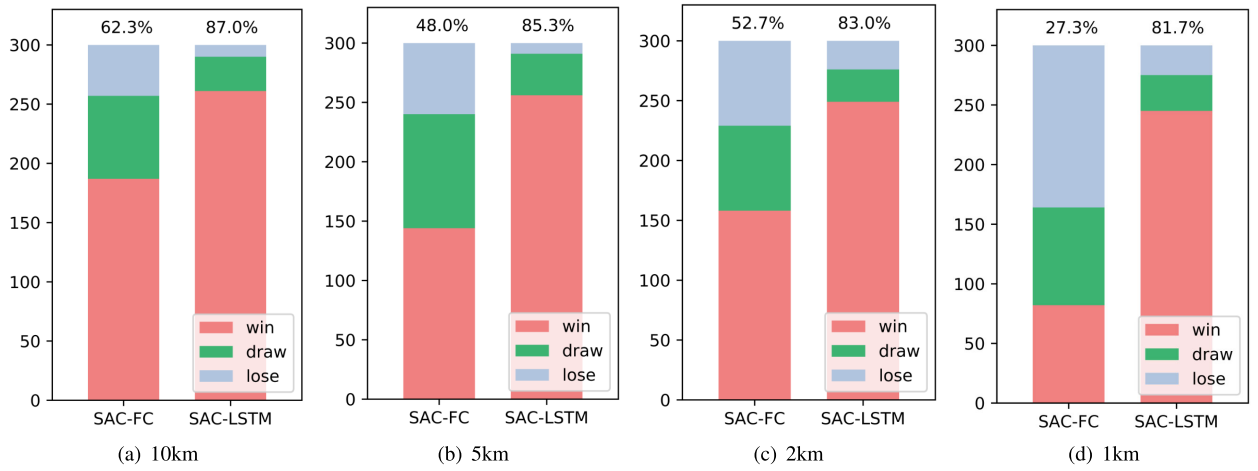


FIGURE 11. Air combat evaluation results on range limitation environment.

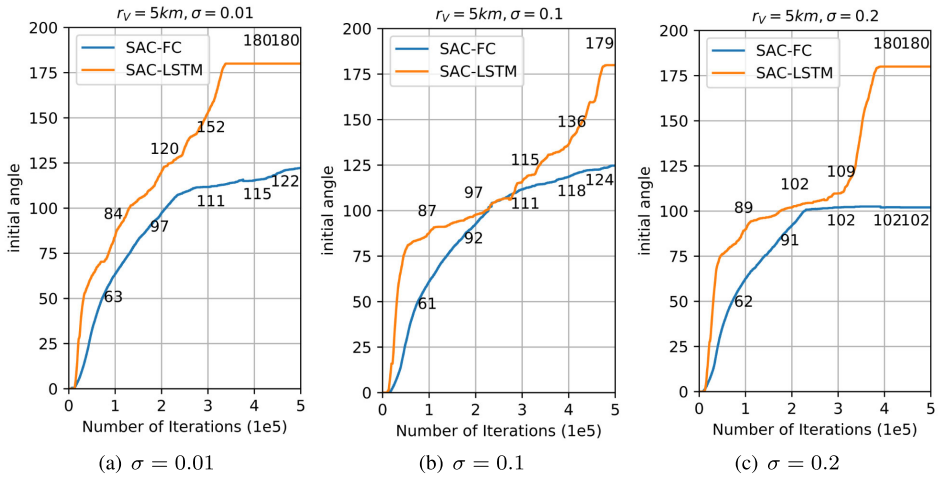


FIGURE 12. Error injection results at 5km range limitation environment.

and SAC-LSTM agents in each POMDP environment. The experimental results show that all SAC-LSTM agents were trained up to the final 180°. In the 0.01 and 0.1 POMDP environments, it reached 180° without blockage, whereas in

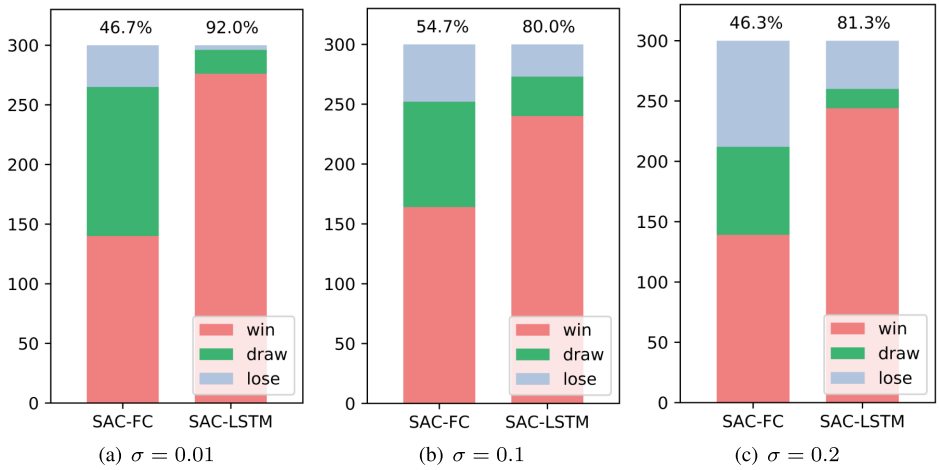


FIGURE 13. Air combat evaluation results on error injection environment.

the 0.2 POMDP environment, learning did not progress well for a long time in a neutral formation near 100° and showed a steep rise to 180°, similar to the 2 km and 1 km POMDP environments.

None of the SAC-FC agents could complete the curriculum within the update limit. The final angles are similar in the three environments. It seems that noise is less affected than the range limitation, according to the characteristics of the SAC algorithm, which is robust to noise.

The result of comparing win ratios indicates that the combat models trained with the proposed framework are robust to the errors of sensor measurements. As shown in Fig. 13, all agents trained with SAC-LSTM showed higher winning rates compared to agents trained with SAC-FC. The average win ratio of the LSTM-based agents is 84.4%, whereas that of the FC-based agents is 49.2%.

3) CURRICULUM LEARNING EFFECT EVALUATION

In this experiment, we investigated whether applying our proposed curriculum improves efficiency and performance in learning procedures. We compared SAC-LSTM and SAC-LSTM-no-curriculum with the conditions of the POMDP environment fixed. r_V was set to 5 km, e_V was set to $\sigma = 0.01$. The SAC-LSTM-no-curriculum was learned through episodes of neutral engagement situations starting at random positions. Figure 14 shows the results of the evaluation engagement in the learning process.

SAC-LSTM's winning rate against RM rose rapidly and eventually showed a higher winning rate compared to SAC-LSTM-no-curriculum. This results support that our turn circle-based curriculum robustly worked in various WVR engagement situations.

4) MANEUVER ANALYSIS

We analyzed the engagement logs stored during learning with the maneuver visualization tool and examined whether the learned agent performed maneuvers that could be used

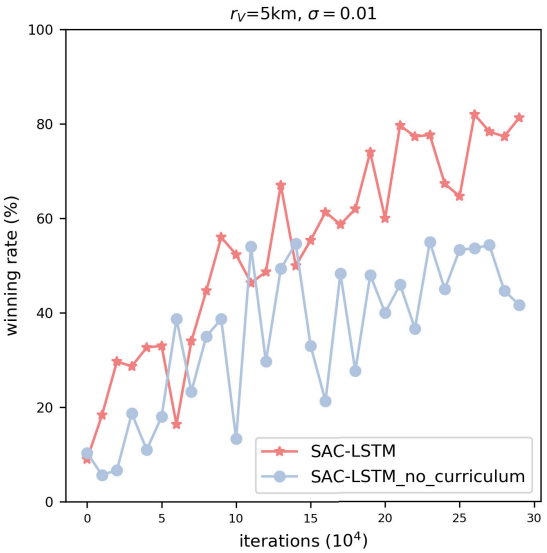
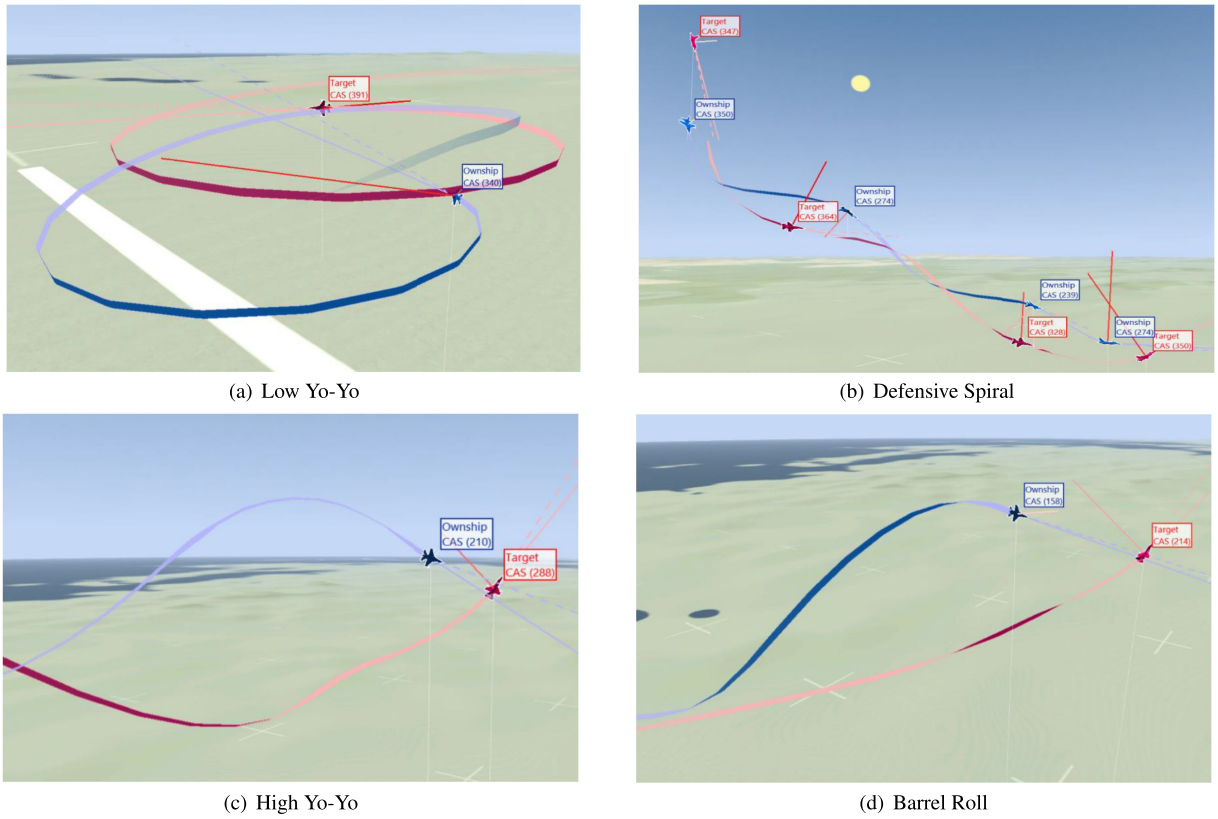


FIGURE 14. Turn circle-based curriculum learning effect evaluation.

in practice. According to the BFM manual, High/Low Yo-Yo, Scissors, Lag/Barrel Roll, Defensive Spiral, etc., are explained as maneuvers for WVR engagement. High Yo-Yo is a maneuver in which the speed of the ownship is faster in an offensive situation. Thus, if there is a risk of overshooting, it slows down by rising and then descending. Low Yo-Yo is a maneuver performed when the target is relatively far away in an offensive situation. It increases speed through descent and, simultaneously, reduces the size of the turn circle to enable the ownship to get closer to the target's tail. The Scissors are rotational maneuvers to induce or prevent overshoot.

We discovered that the learning agents performed the basic maneuvers according to the situation. Fig. 15 shows the trajectories of some maneuvers. In the figure, the blue aircraft is the training model, and the red aircraft is the rule-based model.



(a) Low Yo-Yo (b) Defensive Spiral

(c) High Yo-Yo (d) Barrel Roll

FIGURE 15. Basic maneuvers of the combat model.

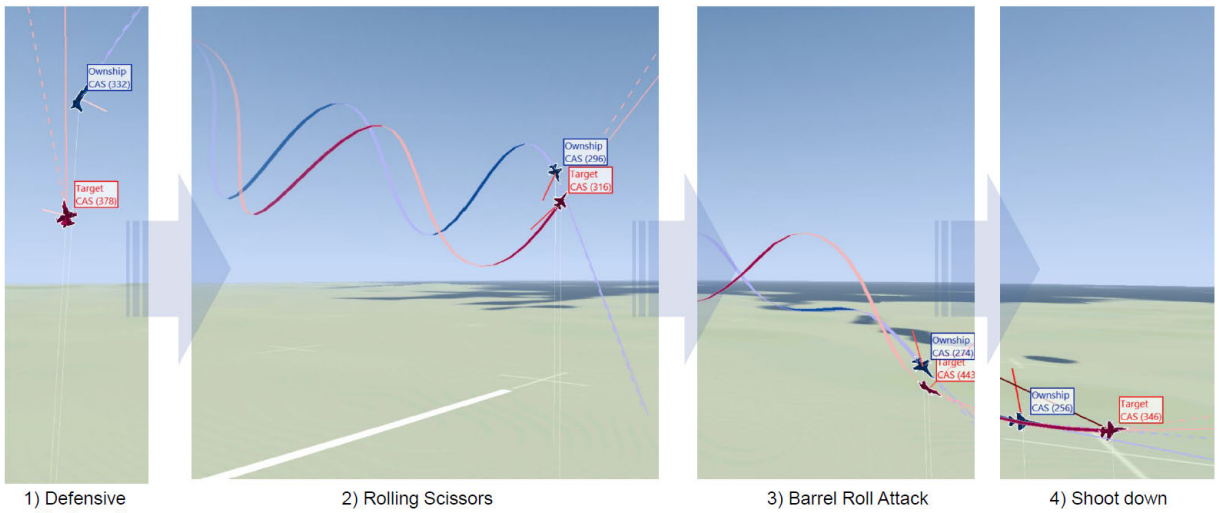


FIGURE 16. One of winning patterns at defensive upside formation.

In addition, we examined the pattern in which the learning model wins in defensive formations. Fig. 16 shows, as snapshots, the process of winning in the situation of the initial defensive upside formation. It showed a winning pattern after inducing the overshoot of the target through Rolling Scissors and Barrel Roll Attack maneuvers.

VI. CONCLUSION

In this study, we proposed a framework for developing RL-based air combat models in engagement environments with realistic limitations and errors. The partially observable environments were designed with radar-like and visual sensors. The SAC algorithm was used as the learning algorithm,

and a network architecture including LSTM was applied. In addition, the curriculum learning was proposed to increase the learning effectiveness by limiting the observation space. Learning was conducted against the BFM manual-based model. In POMDP environments, the ranges of the visual sensor were limited to 10 km, 5 km, 2 km, or 1 km, and the error-injected environments were produced with Gaussian noises $\sigma = 0.01$, 0.1, or 0.2. As the results of the experiments show, winning rates of 75% or more were achieved against the rule-based model in all POMDP environments. In addition, the validity of the curriculum we applied was confirmed by the high winning rate difference in the experimental results. In future work, we plan to examine applicability to real environments rather than simulation environments.

REFERENCES

[1] X. Ma, L. Xia, and Q. Zhao, "Air-combat strategy using deep Q-learning," in *Proc. Chin. Autom. Congr. (CAC)*, Nov. 2018, pp. 3952–3957.

[2] B. Vlahov, E. Squires, L. Strickland, and C. Pippin, "On developing a UAV pursuit-evasion policy using reinforcement learning," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 859–864.

[3] N. Ernest, K. Cohen, E. Kivelevitch, C. Schumacher, and D. Casbeer, "Genetic fuzzy trees and their application towards autonomous training and control of a squadron of unmanned combat aerial vehicles," *Unmanned Syst.*, vol. 3, no. 3, pp. 185–204, Jul. 2015.

[4] D. Silver, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[5] O. Vinyals, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[6] H. S. Inc. (Sep. 25, 2020). *Heron Systems at DARPA Alpha Dogfight Trials*. Accessed: Dec. 6, 2022. [Online]. Available: https://www.youtube.com/watch?v=lldE5XFTA88

[7] Y. Chen, J. Zhang, Q. Yang, Y. Zhou, G. Shi, and Y. Wu, "Design and verification of UAV maneuver decision simulation system based on deep Q-learning network," in *Proc. 16th Int. Conf. Control, Autom., Robot. Vis. (ICARCV)*, Dec. 2020, pp. 817–823.

[8] J. Xu, Q. Guo, L. Xiao, Z. Li, and G. Zhang, "Autonomous decision-making method for combat mission of UAV based on deep reinforcement learning," in *Proc. IEEE 4th Adv. Inf. Technol., Electron. Autom. Control Conf. (IAEAC)*, vol. 1, Dec. 2019, pp. 538–544.

[9] Q. Yang, J. Zhang, G. Shi, J. Hu, and Y. Wu, "Maneuver decision of UAV in short-range air combat based on deep reinforcement learning," *IEEE Access*, vol. 8, pp. 363–378, 2019.

[10] M. M. Ozbek and E. Koyuncu, "Reinforcement learning based air combat maneuver generation," 2022, *arXiv:2201.05528*.

[11] J. Xianyong, M. Hou, G. Wu, Z. Ma, and Z. Tao, "Research on maneuvering decision algorithm based on improved deep deterministic policy gradient," *IEEE Access*, vol. 10, pp. 92426–92445, 2022.

[12] D. Hu, R. Yang, J. Zuo, Z. Zhang, J. Wu, and Y. Wang, "Application of deep reinforcement learning in maneuver planning of beyond-visual-range air combat," *IEEE Access*, vol. 9, pp. 32282–32297, 2021.

[13] W. Kong, D. Zhou, Z. Yang, Y. Zhao, and K. Zhang, "UAV autonomous aerial combat maneuver strategy generation with observation error based on state-adversarial deep deterministic policy gradient and inverse reinforcement learning," *Electronics*, vol. 9, no. 7, p. 1121, Jul. 2020.

[14] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.

[15] R. Bellman, "A Markovian decision process," *Indiana Univ. Math. J.*, vol. 6, no. 4, pp. 679–684, 1957.

[16] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[17] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[18] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 1587–1596.

[19] B. Eysenbach and S. Levine, "If MaxEnt RL is the answer, what is the question?" 2019, *arXiv:1910.01913*.

[20] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *J. Mach. Learn. Res.*, vol. 21, pp. 1–50, Jul. 2020.

[21] K. J. Åström, "Optimal control of Markov processes with incomplete state information," *J. Math. Anal. Appl.*, vol. 10, no. 1, pp. 174–205, 1965.

[22] M. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," in *Proc. Assoc. Adv. Artif. Intell. (AAAI) Fall Symp. Ser.*, 2015, pp. 1–52.

[23] Y. Shao, Q. Kong, T. Matsumura, T. Fuji, K. Ito, and H. Mizuno, "Mask Atari for deep reinforcement learning as POMDP benchmarks," 2022, *arXiv:2203.16777*.

[24] C. Romac and V. Béraud, "Deep recurrent Q-learning vs deep Q-learning on a simple partially observable Markov decision process with minecraft," 2019, *arXiv:1903.04311*.

[25] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, "Memory-based control with recurrent neural networks," 2015, *arXiv:1512.04455*.

[26] L. Meng, R. Gorbet, and D. Kulic, "Memory-based deep reinforcement learning for POMDPs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 5619–5626.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," 2014, *arXiv:1409.1259*.

[29] Z. Yang and H. Nguyen, "Recurrent off-policy baselines for memory-based continuous control," in *Proc. Deep RL Workshop, Neural Inf. Process. Syst. (NeurIPS)*, 2021, pp. 1–14.

[30] S. Kapturowski, G. Ostrovski, J. Quan, R. Munos, and W. Dabney, "Recurrent experience replay in distributed reinforcement learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–19.

[31] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 15084–15097.

[32] I. Vaswani, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5998–6008.

[33] M. Siebenborn, B. Belousov, J. Huang, and J. Peters, "How crucial is transformer in decision transformer?" in *Proc. Neural Inf. Process. Syst. (NeurIPS) Found. Modeling Decis. Making Workshop*, 2022, pp. 1–9.

[34] A. P. Pope, J. S. Ide, D. Micovic, H. Diaz, D. Rosenbluth, L. Ritholtz, J. C. Twedt, T. T. Walker, K. Alcedo, and D. Javorsek, "Hierarchical reinforcement learning for air-to-air combat," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2021, pp. 275–284.

[35] J. Berndt, "JSBSim: An open source flight dynamics model in C++," in *Proc. AIAA Model. Simul. Technol. Conf. Exhib.*, Aug. 2004, p. 4923.

[36] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI gym," 2016, *arXiv:1606.01540*.

[37] V. Mnih, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[38] J. Oh, C. Kim, S. H. Ro, W. C. Choi, and Y. Kim, "Air-to-air BFM engagement simulator for AI engagement model," in *Proc. Korea Inst. Mil. Sci. Technol. Conf.*, 2022, pp. 1753–1754.

[39] M. Lee, J. Oh, C. Kim, J. Bae, Y. Kim, and C. Jee, "The development of rule-based AI engagement model for air-to-air combat simulation," *J. Korea Inst. Mil. Sci. Technol.*, vol. 25, no. 6, pp. 637–647, Dec. 2022.

JUNG HO BAE received the B.S., M.S., and Ph.D. degrees in computer engineering from Pusan National University (PNU), Pusan, South Korea, in 2007, 2009, and 2014, respectively. Since 2014, he has been with the Agency for Defense Development, South Korea. His current research interests include M&S, robot control, and reinforcement learning.

HOSEONG JUNG received the B.S. degree in electrical and computer engineering from Seoul National University, Seoul, South Korea, in 2021. He is currently an active Naval Officer and has been with the Agency for Defense Development (ADD), since 2021. His current research interests include reinforcement learning and computer vision.

SEOGBONG KIM received the B.S., M.S., and Ph.D. degrees in mechanical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 1993, 1995, and 2009, respectively. Since 1996, he has been with the Agency for Defense Development (ADD), South Korea. His research interests include the development of swarm intelligence, AI pilots, and modeling and simulation using AI.

SUNGHO KIM received the B.S. and M.S. degrees in information engineering from Korea University, in 1994 and 1996, respectively. He is currently pursuing the Ph.D. degree in industrial engineering with Seoul National University, Republic of Korea. Since 1996, he has been with the Agency for Defense Development (ADD). During this period, he joined several unmanned systems projects. He was the Project Manager of object detection using EO/SAR satellite imagery, swarm robotics, and AI-based pilot development. His research interests include swarm robotics, sim-to-real domain adaptation, and biomimetic robotics.

YONG-DUK KIM received the B.S., M.S., and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2000, 2002, and 2008, respectively. Since 2008, he has been with the Agency for Defense Development, South Korea. His current research interests include computer vision, robot control, and machine learning.