Beyond-Visual-Range Air Combat Tactics Auto-Generation by Reinforcement Learning

Haiyin Piao
School of Electronics and Information
Northwestern Polytechnical University
Xian, China
haiyinpiao@mail.nwpu.edu.cn

Zhixiao Sun
Unmanned System Research Institute
Northwestern Polytechnical University
Xian, China
zhixiaosun@mail.nwpu.edu.cn

Guanglei Meng
School of Automation
Shenyang Aerospace University
Shenyang, China
mengguanglei@sau.edu.cn

Hechang Chen
School of Artificial Intelligence
Jilin University
Changchun, China
chenhc@jlu.edu.cn

Bohao Qu
School of Artificial Intelligence
Jilin University
Changchun, China
qubh19@mails.jlu.edu.cn

Kuijun Lang
Department of Electronics System
SADRI Institute
Shenyang, China
jordan_lang@foxmail.com

Yang Sun
Department of Electronics System
SADRI Institute
Shenyang, China
yang.sun2010@gmail.com

Shengqi Yang
Department of Electronics System
SADRI Institute
Shenyang, China
shengqiyang@foxmail.com

Xuanqi Peng
Department of Electronics System
SADRI Institute
Shenyang, China
xuanqi519@foxmail.com

***Abstract*—For quite a long time, effective Beyond-Visual-Range (BVR) air combat tactics can only be discovered by human pilots in the actual combat process. However, due to the lack of actual combat opportunities, making new air combat tactics innovation was generally considered quite difficult. To address this challenge, we first introduced a solely end-to-end Reinforcement Learning (RL) approach for training competitive air combat agents with adversarial self-play from scratch in a high fidelity air combat simulation environment during training. Furthermore, a Key Air Combat Event Reward Shaping (KAERS) mechanism was proposed to provide sparse but objective shaped rewards beyond episodic win/lose signal to accelerate the initial machine learning process. Experimental results showed that multiple valuable air combat tactical behaviors emerged progressively. We hope this study could be extended to the future of air combat machine intelligence research.**

***Index Terms*—air combat, reinforcement learning**

I. Introduction

BVR air combat has several fascinating key features: Engagements involve multiple aircrafts, operating in a contested airspace, strategic maneuvers and weapon operations within a long-term decision horizon. Therefore, BVR air combat AI has always been a hot topic in research. The existing AI air combat methods include rule-based system methods [1, 2], probabilistic model/fuzzy logic and computational intelligence hybrid methods [3–5], machine learning, especially reinforcement learning methods [6–8]. Significant progress and lots of achievements have been made by the aforementioned approaches. However, all existing methods depend on prior knowledge of human beings more or less. Rule-based methods rely on pilots to pre-define air combat rule databases [9]. Probabilistic model/fuzzy logic and computational intelligence hybrid methods require experts to establish a probabilistic reasoning network or to design a particular heuristic objective function [3, 5, 10]. Machine learning methods require a large number of data samples marked by human experts, while existing reinforcement learning air combat methods rely on dense reward functions designed by human experts [7]. Moreover, expert data sets are often expensive, unreliable or simply unavailable. Even when reliable data sets are available, they may impose a ceiling on the performance of systems trained in this manner [11].

In this paper, we aim to address this challenge. The main contributions of this work are summarized as follows:

- We first introduced an end-to-end RL approach to BVR air combat tactics auto-generation, which was solely based on self-play RL training without any supervision from human. Furthermore, a Key Air combat Event Reward Shaping (KAERS) mechanism was proposed to solve the cold start problem in the early stage of air combat neural network training [12]. Compare with previous solutions based on human knowledge, our method exceeds the human congnition limitation, thus prevents AI agents from overfitting to prior knowledge of human beings.
- It was observed that plenty of complex and meaningful

All authors contributed equally.
Corresponding author: Haiyin Piao.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

air combat tactical behaviors have been progressively generated during the training process. To the best of our knowledge, this is the first air combat AI with tactical creativity and evolution ability.

All these accomplishments inspire confidence that RL could eventually enable air combat agents to acquire an unbounded number of expert human-level tactical skills.

II. Related Works

A. Reinforcement Learning

Recently, many RL approaches have been proposed to scale to interactive decision-making problems that were previously intractable, i.e., settings with high-dimensional state and action spaces. Amongst recent work in the field of RL, there is one story with outstanding success-AlphaGo. that defeated a human world champion in Go [13], paralleling the historic achievement of IBM Deep Blue in chess two decades earlier [14]. Unlike the hand crafted rules that have dominated chess-playing systems, AlphaGo was composed of neural networks that were trained using both supervised learning (SL) and RL, in combination with a traditional heuristic search algorithm. A few months later, AlphaGo Zero was announced to defeat AlphaGo without any human prior knowledge or trained from scratch by only self-play [11]. After this, several breakthroughs in AI have been made in these domains by combining deep reinforcement learning (DRL) with self-play, achieving superhuman performance. Challenging collaborative-competitive multi-agent environments have only recently been addressed using end-to-end RL by Jaderberg et al. [15], which learns visually complex first-person 2v2 (2 players versus 2 players) video games to human level, as well as in continuous real-time domains e.g., Dota2 [16] and Starcraft2 [17].

B. AI Air Combat

Beyond the aforementioned traditional AI aircombat methods [3–8], RL based AI air combat algorithms present the potential of simplifying the development and maintenance of complex autonomous systems by learning optimal behaviors from continuous simulations. Such learning can be done without supervision and simply need processor time during learning. Towards this particular air combat AI solution, Vinberg first introduced guided reinforcement learning applied to air combat simulation [6]. McGrew used approximate dynamic programming to solve a fixed velocity, one-on-one air combat maneuvering problem in two-dimensional space [7]. In addition, Kurniawan et al. proposed an empirical study of reward structures for actor-critic reinforcement learning in air combat maneuvering simulation [8].

All these aforementioned works that used RL focused on human expert-designed dense reward approaches. By contrast, our research is different from previous works as it trains two adversarial air combat agents with solely self-play in end-to-end RL manner with highly sparse and objective key event (shown in Table II) based rewards. All training results are accomplished without any human handcrafted rules. Moreover, this particular approach induced a sequence of challenges for the adaptive process that was proved to emerge plenty of complex and meaningful air combat tactical behaviors progressively.

III. Proposed Method

A. Problem Formulation

One-on-one BVR air combat, a.k.a., controlling one aircraft to fight against its opponent in the air. Each air combat scenario can be considered as a competitive game between two adversarial agents. In this paper, we formulate it as a multi-agent extension of Markov Decision Processes(MDPs) called Marcov Games for air combat tactics auto-generation [18]. A Markov game for N agents is defined by a set of states $\left\{\boldsymbol{S}^{i}\right\}_{N}^{i=0}$ and a set of joint actions $\left\{\boldsymbol{A}^{i}\right\}_{N}^{i=0}$ for each agent. Each agent i observes a private system state $\boldsymbol{S}^{i}$ and selects actions through its policy $\pi^{i}:\boldsymbol{S}^{i}\times\boldsymbol{A}^{i}\rightarrow[0,1]$, producing the next state according to the state transition function $\boldsymbol{P}:\boldsymbol{S}\times\boldsymbol{A}^{1}\times\cdots\times\boldsymbol{A}^{N}\times\boldsymbol{S}\rightarrow[0,1]$. Each agent i aims to maximize its won total expected sum of rewards $R_t=E\left\{\sum_{T}^{t=0}\gamma^{t}r_{i}^{t}\right\}$, where γ is a discount factor and T is the time horizon, r^{t} is reward received t steps into the future. The agents' joint policy induces a state-value function, i.e., an expectation over R_t, $V^{\pi_i}\left(s_t\right)=E\left[R_t\mid s_t\right]$, while the action-value function is defined as $Q^{\pi_i}\left(s_t,a_t^i\right)=E\left[R_t^i\mid s_t,a_t^i\right]$. The advantage function $A^{\pi_i}\left(s_t;a_t^i\right)=Q^{\pi_i}\left(s_t,a_t^i\right)-V^{\pi_i}\left(s_t\right)$ describes whether taking action a_t^i is better or worse for agent i when in state s_t than average action of policy π_i, which is V^{π_i} actually. Especially, a Markov Game is called two player zero-sum Markov Game when all two players acts fully-competitive and the sum of reward is zero for these agents.

B. Air Combat States, Actions and Rewards Definition

System states are composed as follows: Adversarial aircrafts' global position x and x_b, velocity v and v_b, attitude ψ, θ, ϕ, normal load factor n_n, range r, closing rate $\dot{r}$ and height advantage Δh. Furthermore, to be successful in air combat, the agent's aircraft needs to be in specific relative aircraft geometry with the opponent. We adopted McGrew's geometry annotation as a baseline [7], and defined the relative state observation calculation as follows: The aircraft centers of mass are connected by the line of sight (LOS) line, which is also used to calculate the range between the two aircraft. The aspect angle (AA) is the angle between the LOS line and the tail of the opponent aircraft. The antenna train angle (ATA) is the angle between the nose of the agent aircraft and the LOS line. The elevation angle (EL) is the vertical angle between the LOS line and the horizontal plane. AA, ATA and EL help the pilot to make maneuvering decisions. By convention, angles to the right side of the aircraft are considered positive and angles to the left negative. Then we added-up radar lock on signal lo, RWR status $warn$, mid-range missile left m_{left} and time to missile seeker activation T_{go} to the end of the

whole state vector, the total air combat state definition can be described as below:

$$S^i = [x, x_b, v, v_b, \psi, \theta, \phi, n_n, r, \dot{r}, \Delta h, AA, ATA, EL, lo, warn, m_{left}, T_{go}]^T \quad (1)$$

We then carefully designed 14 offensive and defensive Basic Fighter Maneuvering (BFM) macro actions for this particular full BVR game, as summarized in Table I. BFMs have been described as the art of maneuvering a combat aircraft in order to obtain a position from which an attack can be made on another aircraft and representing the primary elements that can be viewed as the building blocks for air combat maneuvers [19]. They are composed of accelerations/decelerations, climbs/descents, and turns that can be performed in combination relative to other aircraft. Therefore the total combat strategy consists of composition or series of atomic BFM operations. With such an abstraction in action space, it becomes easier for agents to learn a high-level strategy for the full game becomes easier. Thus RL driven agents could use their training and experience of combat tactics, along with their knowledge of aircraft capabilities, to determine which BFM macro actions to be performed at the proper time [20]. We supposed the combination of BFM macro actions could emerge rich and varied BVR air combat tactics, which have not been realized by human beings. Moreover, maneuvering and shooting operations are all considered as discrete softmax actions described in Figure 1.

TABLE I
BASIC FIGHTER MANEUVERING (BFM) MACRO ACTIONS

Category	Serial	Macro Action
Offense	1	Guided Level Flight
	2	$+30^\circ$ Climbing and Accelerating
	3	$+60^\circ$ Climbing and Accelerating
	4	-30° Offensive Descending
	5	-60° Offensive Descending
	6	$\pm 30^\circ$ Single Side Off(SSO)[1]
	7	$\pm 60^\circ$ Single Side Off(SSO)
	8	Horizontal Snake Maneuvering
Defense	9	Split-S
	10	Turn $\pm 90^\circ$ from LOS
Retreat	11	Level Turning
	12	Fast Turning
	13	Descending -30° after Turning
	14	Descending -60° after Turning

Due to previous works relied on per step dense reward signals tuned by human experts [7, 8], We proposed a method to solve the prohibitively hard credit assignment problem of learning from sparse and delayed episodic win/loss signal (Optimizing hundreds of actions based on a single final reward with some sparse and objective key events). Refer to Table II. We believe this assumption will lead to a bias less solution to the true wining policy rather than over fitted to some human expert crafted potential functions. In these solutions, the causal relationship with the true combat result is mathematically ambiguous.

[1]The $\pm$ symbol denotes aircraft will automatically maneuvering to the smaller angle offset direction.

C. Policy Optimization

The agent's objective is to learn a policy that maximizes the expected sum of discounted rewards, Conversely, the opponent's joint policy is to minimize the expected sum. Correspondingly, we have the following minimax zero-sum Markov Game:

$$Q^*\left(s_t, a_t^i, a_t^{-i}\right) = r\left(s_t, a_t^i, a_t^{-i}\right) + \max_{a^i \in \pi^i} \min_{a^{-i} \in \pi^{-i}} Q^*\left(s_{t+1}, \left\langle a_{t+1}^i, a_{t+1}^{-i} \right\rangle\right) \quad (2)$$

Where $Q^*\left(s_t, a_t^i, a_t^{-i}\right)$ is the optimal action-state value function, which follows the Bellman Optimal Equation, For solving such kinds of games, agent policies are trained against each other by self-play, which means fictitious players choose the best responses to their opponents' average behavior. The average strategies of fictitious players were proven converging to Nash equilibrium in this particular kind of game [21–23], furthermore, self-play acts as a natural curriculum as agents always play opponents of an appropriate level [24].

We utilized decentralized execution and centralized training paradigm [25]. At execution time, each agent acts given only its own observations and memory state. At optimization time, we used a centralized global value function for each agent, which has access to the full environment state without any imperfect information due to visibility [25–27]. In this training paradigm, maximizing/minimizing the agent's/opponent's expected rewards simultaneously is equivalent to maximizing all separate agents' expected rewards, which leads our adversarial training into a minimax pattern. Agent policies are approximated by one unified neural network with shared parameters θ. The neural network constructed an approximation to both the policy distribution $\pi\left(a|s,\theta\right)$ and a value function $V\left(s,\theta\right)$ which predicts the discounted future returns. The parameters of the policy are learned by Proximal Policy Optimization (PPO) algorithm.

PPO that utilized by us is a state-of-the-art synchronous policy gradient method [28], Policy gradient techniques aim to estimate the gradient of expected returns with respect to the parameters of its policy $\nabla_\theta J\left(\pi_\theta\right) = E_{a\sim\pi_\theta}\left[\nabla_\theta \log\left(\pi_\theta\left(a_t|s_t\right)\right) V\left(s_t\right)\right]$. where $J(\pi_\theta)$ is the accumulated expected return. The aim of PPO is to prevent training instabilities by penalizing large changes to the policy. The policy network parameters θ could be updated according the gradient of PPO loss function listed below:

$$Q\left(s_t, a_t\right) = E\left[r_{t+1} + \gamma V\left(s_{t+1}\right)\right] \quad (3)$$

$$A\left(s_t, a_t\right) = Q\left(s_t, a_t\right) - V\left(s_t\right) = r_{t+1} + \gamma V_\theta\left(s_{t+1}\right) - V_\theta\left(s_t\right) \quad (4)$$

$$J\left(\pi_\theta\right) = E_{a\sim\pi_\theta}\left[\min\left(l_t\left(\theta\right) A\left(s_t, a_t\right), clip\left(l_t\left(\theta\right), 1-\varepsilon, 1+\varepsilon\right) A\left(s_t, a_t\right)\right)\right] - E_{s\sim\pi_\theta}\left[\alpha H\left(s_t, \cdot, \theta\right)\right] \quad (5)$$

Where $l_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)}$ denotes the likelihood ratio between new and old policies and $clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)$ clips $l_t(\theta)$ in the interval $[1-\varepsilon, 1+\varepsilon]$. $\alpha H(s_t, \cdot, \theta)$ is an entropy regularization penalty for encouraging policy exploration and α is scaling factor. Both policy and value are adjusted towards an lookahead value, thus value function loss could be defined as follows and value network could be updated according the loss:

$$L_v = E_{s\sim\pi_\theta}\left[\left(r_{t+1} + \gamma^n V(s_{t+1}, \theta) - V(s_t, \theta)\right)^2\right] \quad (6)$$

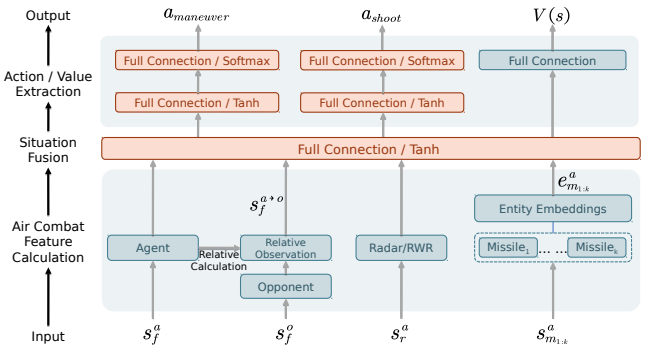

Fig. 1. Agent neural network architecture

The details of network structure is shown in Figure 1. The input of our neural network could be splited into 3 semantical parts:

- Agent and Radar/RWR input layers contain fully-connected hidden units aligned with agent and Radar/RWR state input vector s_f^a and s_r^a, taking Tanh as activation function.
- Opponent state input s_f^o then be calculated as relative observation based on agent state [7], which has the same network structure with agent layer.
- For the 4 missiles mounted, each one is embedded with a unified entity embeddings layer where parameters are shared between missiles.

Next, we concatenated all outputs together to a centralized hidden layer with 512 units and activated also via Tanh function. Finally, we treated policy and value function as joint distribution by pulling out $a_{maneuver}$, a_{shoot} and $V(s)$ function heads separately. For maneuvering and shoot policy heads, we represented the overall policy in an auto-regressive manner, utilising the chain rule $\pi_\theta(a|s) = \prod_{l=0}^{L} \pi_\theta\left(a^l|a^{<l}, s\right)$ [29] where l represents different kinds of actions. This representation is arguably simpler as it transforms the problem of choosing a full action a to a sequence of decisions for each argument a^l. For all discrete actions, an exponential softmax distribution $\pi_\theta(a|s) = \frac{e_\theta^{h(s,a)}}{\sum_b e_\theta^{h(s,b)}}$ was adopted in action selection, where $h(s,a)$ and $h(s,b)$ indicates previous layer output logits.

D. Key Air combat Event Reward Shaping (KAERS)

The main purpose of one-on-one BVR air combat is to kill or drive out opponents within a specified time. Effective killing depends on a series of precise operations, including aiming, locking, firing, guidance, and avoiding missiles fired by enemy [20]. In case of only win/lose signal as an episodic reward, the probability of these series of effective actions occurring simultaneously is obviously close to zero. However, the success of agents in the competitive games requires the agents to occasionally solve the task (i.e. win the competition) by random actions. In RL theory, we called this kind of problem a long-term sparse reward time credit assignment problem. In such a scenario, since the neural network of the agent is initialized randomly by Xavior method [30], and the basic skills related to air combat at the very beginning are deficient, the problem of cold start could impact seriously [12]. In order to address this problem, we proposed a method called Key Air combat Event Reward Shaping (KAERS). The main idea is providing relatively sufficient signal stimulation at beginning of agent training by discrete key events during air combat as described in Table II. We can use these eventually shaped rewards to allow the agents to learn basic air combat skills initially.

$$r_t = \lambda e_t^T w + r_{episodic} \quad (7)$$

Where r_t is the t step total reward, e_t is key event, w is reward scale weight according to the corresponding event, as described in Table II. The key event shaped reward is gradually annealed to zero linearly after 100th iteration, in favor of the true end game episodic reward $r_{episodic}$, to allow the agents to train for the majority of the training using the sparse episodic reward. This is achieved using a linear annealing factor of λ. Follow-up experiments will show that the KAERS method can effectively solve the cold start problem in the early stage of air combat agent training, thereby sufficiently accelerating the total training process.

TABLE II
KEY AIR COMBAT EVENT REWARD SHAPING (KAERS)

Category	Event Name (e_t)	Polarity	Weight (w)
Episodic Rewards	Kill	+	500
	Be killed	-	500
	Crash	-	500
	Drive Out	+	500
	Be driven out	-	500
Key Event based Rewards	Shoot	-	25
	Stall	-	50
	Out	-	10
	Lockon	+	2
	Be Locked	-	2
	Missile Lock	+	5
	Missile Alert	-	5
	Lock Escaped	+	5
	Missile Escaped	+	10

E. Training Process

In self-play training, we initialized both red and blue agents' neural networks by Xavier initialization method for producing brand-new networks [30], which means no human air combat knowledge is adopted by our approach. In the adversarial

training scenario, we truncated the trajectory and calculated advantage function $A(s,a)$ after $n = 30$ forward steps of a network or if a terminal signal is received. The optimization process runs 16 asynchronous processes using shared Adam. For each parallel process, we ran experiments till the total episode replay buffer(overcomed the correlation of empirical data and improved data utilization) gathered a specified batch size of samples [31]. The learning rate was set to $1e^{-4}$. We used an independent entropy penalty of $4e^{-5}$ for the action heads. The environment performed a fixed updating rate as one step per second, we acted every 4 game steps, which is equivalent to 4 seconds per action.

IV. Experiments

A. WUKONG: Air Combat Simulation

The experimental environment used in this paper is WUKONG[1], a RL oriented air combat simulation framework currently under development by Northwestern Polytechnical University(NPU). WUKONG is designed for simulating teams of aircraft in BVR n-versus-m style air combat. The environment is designed for air combat operation and AI-driven combat behaviors research.

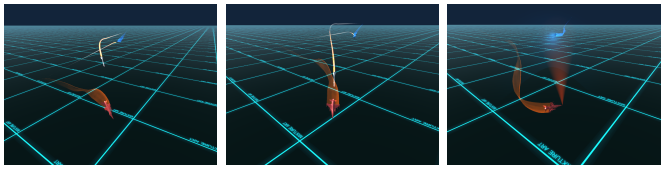

Fig. 2. Dive and retreat tactics emerged

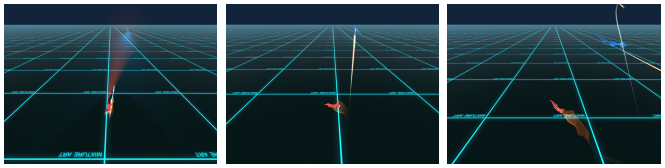

Fig. 3. Loft shoot and crank tactics emerged

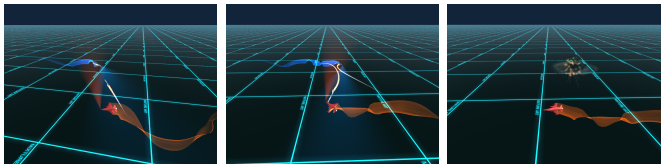

Fig. 4. Drag and re-engage tactics emerged

The scenario we considered in this paper consists of one red and one blue fighter[2] aircraft. Each aircraft consists of four components-aircraft flight dynamics, radar and Radar Warning Receiver(RWR), mid-range missiles and expert level built-in AI bot. The only purpose of our BVR air combat scenario is to shoot the enemy down or to drive away from them. The environment also offers the state of both aircrafts' radar/RWR sights which includes information on the orientation of entities list that radar/RWR has detected and could track. We leave the radar target searching procedure out of the discussion as we assuming perfect observation of opponent's states. Thus, radar is only used for locking on the opponent before a shoot and performing missile mid-guidance procedures.

[1]WUKONG means "Zen of Aerospace" in Chinese.

[2]Also denoted as agent and opponent for convenience.

B. Auro-Generated Air Combat Tactics

With the progress of continuous air combat agent training in WUKONG simulation environment, it can be observed that adversarial agents had gradually produced a series of brand-new tactical behaviors. Although these behaviors are not hard-coded by human experts, they are highly consistent with the performance of expert-level jet pilot [20, 32]. The agent learned from nothing but only episodic rewards and sparse key air combat event based shaped rewards. Therefore, adversarial agents actually constructed an adaptive course learning mechanism through self-play. Such mechanism gradually generated brand-new air combat tactical behaviors as training progresses thus forces opponents to adapt to accordingly, which indicates for the first time, we have endowed air combat AI with the ability of continuous evolution.

At the very beginning, agents just spun around without showing any meaningful tactical behavior. As agents gradually enhanced the accuracy of aiming and shooting skills through self-play, we noticed that adversarial agents began to shoot down each other. Once shooting down key events occurred, the opponent was enforced to evade incoming missiles by exploring and mastering defense dragging tactics. The auto-generated tactics could be classified into three main categories:

- Dive and retreat tactics, refer to Figure 2.
- Loft shoot and crank tactics, refer to Figure 3.
- Drag and re-engage tactics, refer to Figure 4.

Each category is also consist of some concrete-tactics which will be discussed in Table III. This interaction between agents along with the training process effectively promoted the continuous progress of adversarial evolution.

As described in Table III, we found that training batch size adopted in each iteration performed a critical impact on behavior emergence. When batch size was set to smaller than 16000, agents could not learn any semantically tactical behaviors by simply increasing the training iteration. When batch size reached 16000, agents could only learn basic shooting and turning behaviors through self-play since 210th iterations, but from empirical study, learned behaviors always showed up with a non-ignorable uncertainty, rather than converging to specific behaviors for particular situations. Finally, when batch size achieved 32000, agents could spontaneously emerge all above tactical behaviors since 75th iterations. Due to computation resource limitation, we did not conduct larger-scale distributed sampling-training iterative experiments.

TABLE III
PROGRESSIVELY EMERGENCE OF AIR COMBAT TACTICS

Tactics Category	Concrete-Tactics	Iteration	
		Batch 16000	Batch 32000
Dive and Retreat	Turn-Out Retreat	210	75
	Precise Retreat Timing	360	120
	Turn-Out and Dive	——	230
Loft Shoot and Crank	Precise Shoot Timing	290	110
	Climb and Shoot	——	270
	Shoot and Crank	——	380
Drag and Re-Engag	Precise Drag Timing	——	200
	Re-Engage Timing	——	430

C. Comparison with Baseline Algorithms

To show the effectiveness of our approach, we compared our proposed PPO with KAERS architecture with four state-of-the-art RL algorithms with aforementioned expert-crafted shaped dense rewards including PPO with McGrew Score, vanilla PPO, A2C with McGrew Score, and TRPO with McGrew Score, in which actor-critic manner RL with McGrew Score is a widely-used paradigm in air combat agent training [7, 8].

The experiment was set to collect 960000 self-play samples for training five independent air combat agents, and then reveal the average rewards training effectiveness and the battle result statistics between our approach with other algorithms.

Comparison of the training results are shown in Figure 5 (The shaded region denotes a standard deviation of average evaluation over five trials). With the training procedure processed, all algorithms achieved relatively high scores from large negative starting points, which demonstrated that all algorithms had learned some air combat knowledge solely from self-play. Compared with the other four state-of-the-art baselines, the average reward of PPO with KAERS mechanism performed lowest during training. On the contrary, PPO and TRPO with McGraw score performed much better than the others, vanilla PPO and A2C with McGraw score performed mediocrely.

Since McGraw score is a dense human-crafted per step shaped reward mechanism [7], it was easier to get higher training scores via this approach. PPO and TRPO both utilized importance-sampling and experienced more training epochs than A2C thus performed better than it. From air combat replay visualization, it was demonstrated that agents trained by McGrew score quickly converged to approaching opponent's 6 o'clock aspect, which was exactly consistent with the McGraw score principle. Therefore, it could be determined that although this approach gained more rewards but also overfitted to some type of human-crafted strategy. Moreover, vanilla PPO achieved higher scores than our approach. By using replay visualization, it was revealed that in the early stage of training, vanilla PPO only learned conservative spinning around strategy. Such behaviors ensured both sides gained approximately 0 scores, but obviously, it was helpless to discover new air combat tactics. In the later period of training, as the adversaries slowly began to launch some missiles randomly with ± 500 rewards after mastering simple flight skills, which induced larger reward variance as demonstrated.

Although the training reward curve of our method looks lower than the others, it was revealed in replay visualization that KAERS mechanism encouraged the willingness of adversaries to fight each other at the early stage of training. e.g., a successful radar lock-on operation gained $+2$ score, which drove the agent to lock opponents more incentively. Moreover, since successful radar lock-on is the premise of a missile launch, it enlarged the probability of successful shots, which produced more opportunities for adversaries to discover the necessity of evading incoming missiles launched by opponents in the early stage of training. Therefore, we could infer that our PPO with KAERS approach revealed the causality of specific tactical behavior with more objective end-game results, which refrained the proposed method from overfitting to some certain kind of expert knowledge.

In order to prove this conclusion, we carried out 100 air combat battles between agents trained by PPO with KAERS approach with other algorithms right after training ended. The experimental results were shown in figure 6. From the confrontation result, it could be clearly found that our method achieved the highest winning rate even if the draw game ratio is ignored. Consequently, PPO with KAERS method could objectively and efficiently accelerate the self-play training process.

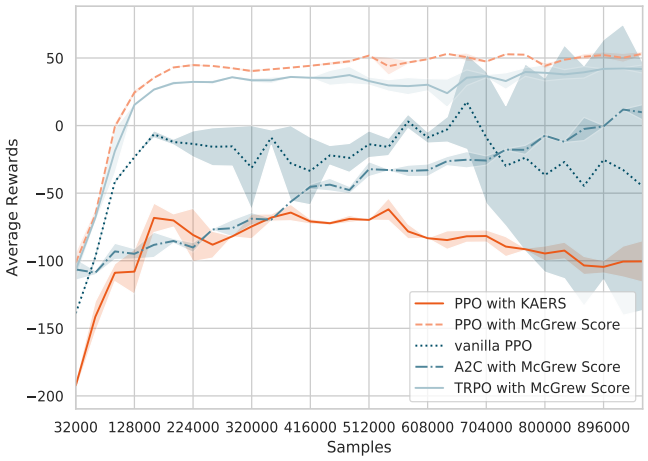

Fig. 5. Training rewards comparison

V. CONCLUSION

In this paper, we focused on solving a fully autonomous BVR air combat problem. Firstly, we took the lead in introducing an end-to-end RL air combat AI approach without human prior knowledge. Moreover, we also proposed a mechanism named KAERS for accelerating the initial air combat training process by providing objective key air combat event based shaped rewards. Consequently, experimental results showed that multiple valuable air combat tactical behaviors emerged by self-play manner training process spontaneously. In a sense,

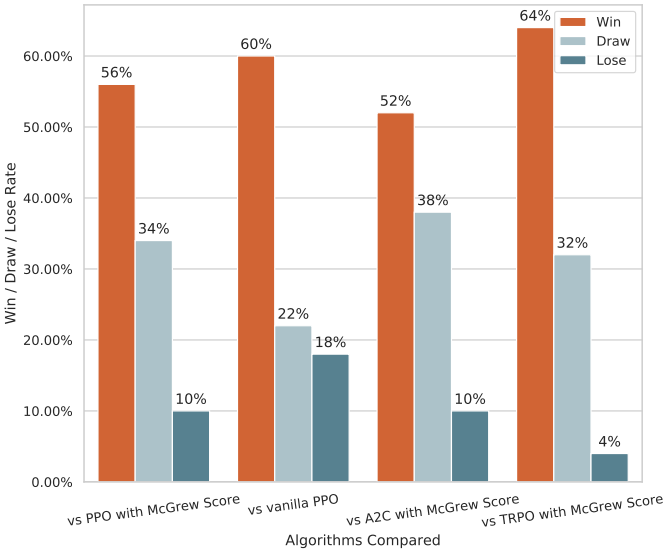

Fig. 6. Air combat competition results between algorithms

we endowed air combat AI with continuous tactics evolution capacity.

However, this work has preliminarily assumed that fully-observed information about all WUKONG air combat environment could be acquired accurately. In the future, we will work towards a Partially-Observed Markov Decision Process (POMDP) version of one-on-one BVR air combat simulation and trying to investigate on solutions for highly uncertainty nature of the BVR air combat.

REFERENCES

[1] H. Shin, J. Lee, H. Kim, and D. H. Shim, "An autonomous aerial combat framework for two-on-two engagements based on basic fighter maneuvers," *Aerospace Science and Technology*, vol. 72, pp. 305 – 315, 2018.

[2] M. Wang, L. Wang, T. Yue, and H. Liu, "Influence of unmanned combat aerial vehicle agility on short-range aerial combat effectiveness," *Aerospace Science and Technology*, vol. 96, p. 105534, 2020.

[3] K. Virtanen, T. Raivio, and R. P. Hämäläinen, "An influence diagram approach to one-on-one air combat," in *Proceedings of the 10th International Symposium on Differential Games and Applications, St. Petersburg, Russia*, vol. 2, 2002, pp. 859–864.

[4] K. Virtanen, J. Karelahti, and T. Raivio, "Modeling air combat by a moving horizon influence diagram game," *Journal of guidance, control, and dynamics*, vol. 29, no. 5, pp. 1080–1091, 2006.

[5] N. Ernest, D. Carroll, C. Schumacher, M. Clark, K. Cohen, and G. Lee, "Genetic fuzzy based artificial intelligence for unmanned combat aerial vehicle control in simulated air combat missions," *Journal of Defense Management*, vol. 6, no. 1, pp. 2167–0374, 2016.

[6] D. Vinberg, *Guided Reinforcement Learning applied to Air-Combat Simulation*. Citeseer, 2009.

[7] J. S. McGrew, J. P. How, B. Williams, and N. Roy, "Air-combat strategy using approximate dynamic programming," *Journal of guidance, control, and dynamics*, vol. 33, no. 5, pp. 1641–1654, 2010.

[8] B. Kurniawan, P. Vamplew, M. Papasimeon, R. Dazeley, and C. Foale, "An empirical study of reward structures for actor-critic reinforcement learning in air combat manoeuvring simulation," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2019, pp. 54–65.

[9] K. Goodrich and J. McManus, "Development of a tactical guidance research and evaluation system (tgres)," in *Flight Simulation Technologies Conference and Exhibit*, 1989, p. 3312.

[10] T.-H. Teng, A.-H. Tan, Y.-S. Tan, and A. Yeo, "Self-organizing neural networks for learning air combat maneuvers," in *The 2012 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2012, pp. 1–8.

[11] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[12] N. Ding and R. Soricut, "Cold-start reinforcement learning with softmax policy gradient," in *Advances in Neural Information Processing Systems*, 2017, pp. 2817–2826.

[13] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[14] M. Campbell, A. J. Hoane Jr, and F.-h. Hsu, "Deep blue," *Artificial intelligence*, vol. 134, no. 1-2, pp. 57–83, 2002.

[15] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castaneda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman *et al.*, "Human-level performance in first-person multiplayer games with population-based deep reinforcement learning," *arXiv preprint arXiv:1807.01281*, 2018.

[16] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dkebiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[17] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[18] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine Learning Proceedings 1994*, W. W. Cohen and H. Hirsh, Eds. San Francisco (CA): Morgan Kaufmann, 1994, pp. 157 – 163.

[19] "Air combat maneuvering," 2019. [Online]. Available: https://en.wikipedia.org/wiki/Air_combat_manoeuvring

[20] P. Bonanni, *The art of the kill*. Spectrum HoloByte, 1993.

[21] D. S. Leslie and E. J. Collins, "Generalised weakened

fictitious play," *Games and Economic Behavior*, vol. 56, no. 2, pp. 285–298, 2006.

[22] J. Heinrich, M. Lanctot, and D. Silver, "Fictitious self-play in extensive-form games," in *International Conference on Machine Learning*, 2015, pp. 805–813.

[23] J. Heinrich and D. Silver, "Deep reinforcement learning from self-play in imperfect-information games," *arXiv preprint arXiv:1603.01121*, 2016.

[24] T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, and I. Mordatch, "Emergent complexity via multi-agent competition," *arXiv preprint arXiv:1710.03748*, 2017.

[25] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, pp. 6379–6390.

[26] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," *arXiv preprint arXiv:1710.06542*, 2017.

[27] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[29] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser *et al.*, "Starcraft ii: A new challenge for reinforcement learning," *arXiv preprint arXiv:1708.04782*, 2017.

[30] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[32] R. L. Shaw, *Fighter Combat*. Naval Institute Press Annapolis, 1985.