Are We Machine Learning Yet? Computer Generated Forces with Learning Capabilities in Military Simulation

Joost van Oijen, Armon Toubman
NLR – Royal Netherlands Aerospace Centre
Amsterdam, the Netherlands
Joost.van.Oijen@nlr.nl, Armon.Toubman@nlr.nl

ABSTRACT

In military modeling and simulation (M&S) there is an increasing need for Computer Generated Forces (CGFs) with machine learning capabilities for use in training or decision support applications. Machine learning based CGFs have benefits for the implementation of adaptive enemies for human-in-the-loop training; the development and evaluation of TTPs (tactics, techniques and procedures) for military platforms; or supporting course of action (CoA) planning and analysis in mission simulation.

Machine learning introduces a radical new paradigm for behavior modeling for CGFs. When modeling a behavioral task for a CGF, rather than handcrafting individual decisions and actions, learning algorithms allow for mere specifying the underlying goal of the task, and leaving it up to the algorithm to learn how to achieve this goal. In this paper we explore the implications of this paradigm shift for the future development of CGFs. Underlying the paper is our vision that within five to ten years, military operators can independently employ machine learning for CGFs to enhance their real-world operations.

Our contribution to the field is threefold. First, we describe the benefits of CGFs with learning capabilities by reviewing several application areas in the M&S domain. Second, we describe the primary challenges to the exploitation of machine learning capabilities for CGFs in terms of gaps in industry practices, impacts on the process of behavior modeling and deployment in M&S ecosystems. Finally, we report on our experiences with taking a pragmatic approach to overcoming some of these challenges by implementing and demonstrating trainable and reusable CGF behaviors within a conventional CGF behavior modeling tool. The ideas in this paper have been implemented in a military simulation system geared towards the air-to-air combat domain. Based on this implementation, challenges, lessons learned and future directions are discussed.

ABOUT THE AUTHORS

Joost van Oijen, PhD is a Senior Scientist at the Royal Netherlands Aerospace Centre (NLR). Having a background in Computer Science and Artificial Intelligence, he has over ten years of experience in AI for modelling & simulation, both in the industry and academia. At his current position, Joost leads several R&D projects focused on human behavior modeling for training and decision support. Having a strong software engineering background, he is actively involved in the development of multi-agent systems and behavior modeling tools for military simulation systems.

Armon Toubman, PhD is an R&D engineer at the Royal Netherlands Aerospace Centre (NLR). He designs and tests new modeling and simulation concepts for defense applications. His work focuses on the use of Artificial Intelligence in training simulations, including behavior modeling, adaptive training, and performance assessment.

Are We Machine Learning Yet? Computer Generated Forces with Learning Capabilities in Military Simulation

Joost van Oijen, Armon Toubman
NLR – Royal Netherlands Aerospace Centre
Amsterdam, the Netherlands
Joost.van.Oijen@nlr.nl, Armon.Toubman@nlr.nl

INTRODUCTION

As Artificial Intelligence (AI) pioneer Andrew Ng noted in an interview, "All of AI […] has a proof-of-concept-to-production gap" (Perry, 2021). The application of machine learning (ML) to military modeling and simulation (M&S) has been a 'future work' item in many publications for many years. The academic world continuously delivers new machine learning techniques, each more powerful than the previous one. These techniques can benefit the application of simulated forces, yet looking at current products and tools, machine learning is practically nowhere to be found. Why is this the case, and how can we advance the field in this area?

In earlier work (Roessingh et al., 2017; Toubman et al., 2015; van Oijen et al., 2019), we have summarized the art and science that is behavior modelling for Computer Generated Forces (CGF). Furthermore, we have outlined the benefits of including machine learning capabilities into the modeling process. The current paper focuses on types of applications that may benefit from ML-empowered CGFs, and discusses the practical challenges that must be solved before these CGFs can be capitalized on in a structural manner.

The paper is based on the vision that within five to ten years, military operators can independently employ machine learning for CGFs to enhance their real-world operations.

- We aim for five to ten years, because while research into the machine learning algorithms has boomed in the last decade, the research into the human-machine interfaces for these algorithms has lagged behind (see, e.g., Gibson et al., 2020).
- We aim for empowering a wide range of operators, beyond simulation and IT experts. The algorithms should be usable by the people who directly require the information provided by the algorithms, be it at a desk or in the field. Essentially, this is a military take on democratizing AI (see, e.g., Allen et al., 2019).
- We aim for algorithms that produce results that are based on real-world information, and therefore are applicable to the real-world. The results may be (partially) based on assumptions, as long as the assumptions are clear to the end user (see also Evensen & Bentsen, 2016).

The three contributions of this paper are as follows:

1. We review the two areas of application that may benefit from the use of machine learning in CGFs: training and decision support. We review studies of military machine learning applications, and investigate the more general questions that these applications can answer. Furthermore, we discuss the obstacles to bringing the proposed applications to operational environments.
2. We present our view on the current challenges for applying machine learning to CGFs in military M&S.
3. We discuss some practical implications of exploiting machine learning algorithms in military M&S, based on our experience in addressing the identified challenges.

With these contributions, we aim to advance the practical application of machine learning. We conclude the paper with a summary of the contributions and an outlook on future work.

APPLICATIONS AND OPPORTUNITIES

In this section, we review the state of the art regarding research on ML-empowered CGFs intended for military applications. In this paper we specifically focus on reinforcement learning (RL). RL is a branch of machine learning that is used to train agents to learn behaviors (also known as policies) in some environment. RL enables agents to explore the environment in order to optimize their behavior according to certain user-defined performance metrics (Sutton & Barto, 2018). In recent years, deep RL (a neural network based approach) has seen major advances in the academic field, allowing agents to solve complex problems in intricate environments (Baker et al., 2020; Berner et al., 2019; Mnih et al., 2015).

Below, we discuss two areas of application for the military domain: training and decision support.

Training

Training scenarios, especially for training on the tactical and strategic level, often include non-trainee roles such as opponents or non-combatants. These roles exist to interact with the trainee, or to enrich the training scenario with background activity. While the non-trainee roles may be filled by human experts, it makes economic sense to fill these roles with virtual agents. This is especially the case in environments such as Live, Virtual, Constructive (LVC), Mission Training through Distributed Simulation (MTDS), or embedded training (ET), where constructive entities such as CGFs may replace live players in real-world platforms (cf. Aronsson et al., 2020; Keuning, 2010; Lee et al., 2020).

Reinforcement learning provides several opportunities for enhancing the development of training scenarios, and the training experiences that follow. For example, CGFs may be pre-trained with relevant behaviors and stored in a library, and then be selected for use by a virtual agent in a newly developed training scenario. The pre-trained behavior models can express full mission behavior (e.g. in the air domain: take-off, combat air patrol, engagement, disengagement, and landing for a fighter jet CGF), or partial behaviors (e.g., target selection or an evasive maneuver). Furthermore, reinforcement learning enables the live adaptation of CGF behaviors to anything that can be measured, e.g. in-simulation performance by the trainee, or the cognitive load of the trainee. Given the right measures, the CGFs may adapt their behavior in such a way to optimize the training value for the trainees. The effectiveness of adaptive agents in training is being investigated (Toubman, 2019; van den Bosch et al., 2020).

As an instance of research in this area, Pope et al. (2021) outline the use of reinforcement learning to develop an air combat CGF for participation in a competition held by DARPA, as part of the Air Combat Evolution program. The goal of the program is to advance autonomous air-to-air combat capabilities, but the developed techniques may also be used for training purposes. Additionally, by structurally demonstrating these capabilities, DARPA aims to build trust in human-machine teaming solutions (*Virtual AlphaDogfight Trials Finals (Archived)*, n.d.).

Decision support

For decision support we expect the problem/product owners to benefit from RL-empowered CGFs in simulation experiments aimed at answering complex questions, for instance to support Concept Development & Experimentation (CD&E) activities in Battle Lab environments or to support the Military Decision Making Process (MDMP) during mission planning.

In the context of CD&E, since RL is an optimization process, it may aid in the discovery of new optimal configurations of a set of known variables, such as the development of new tactics for an aircraft with novel capabilities (for a classical example see, e.g., Smith et al., 2001). For the same reason, RL can be applied in the context of *red teaming*, e.g., simulating a smart and adaptive opponent to find the deficiencies in one's own (weapon) systems, tactics or courses of action. As an example of CD&E enabled by reinforcement learning, Zaroukian et al. (2021) study the behavior of a virtual entity that learns to evade multiple pursuers. Here, Zaroukian et al. focus on the interpretation of the learned behavior, thereby extracting high-level strategies from the sequences of specific actions that the entity demonstrates in simulations. A similar study was performed by Luotsinen et al. (2016).

In the context of supporting the mission planning process, RL-empowered CGFs can benefit AI-enabled wargaming and course of action (CoA) development (Evensen et al., 2019; Schwartz et al., 2020). Wargaming by means of simulation provides an automated manner of trying out various courses of action, and distilling the simulation results into a strategy for use in the real world. Wargaming simulations can potentially also be used during mission execution where current executed friendly CoAs can be continuously evaluated and adapted in response to current enemy behavior and predictions of possible enemy CoAs (thereby forming a decision support solution). The introduction of self-optimizing CGFs to wargames may enable novel features, such as forces that adapt their behavior over time in consecutive games. This may provide more realistic modeling of opponent behavior.

From showcase to real-world M&S applications

Above, we presented examples of studies into reinforcement learning applications, and briefly expanded upon the different areas of application where reinforcement learning might enhance operations. Why, then, are the showcased systems so rarely brought to real-world operations?

One major reason is that the showcased systems are often developed using custom simulated worlds, and custom representations of those worlds. This can be observed in the many different papers that employ RL for CGFs. Models developed for each of these worlds, including reinforcement learning models, are not easily transferrable across worlds.

We conjecture that the reason for the many custom simulated worlds is that the commercial/government off-the-shelf (COTS/GOTS) CGF packages do not (directly) support machine learning capabilities. In a brief survey of the websites of eight CGF packages, we found no mention of machine learning (see Table 1). Artificial intelligence was sometimes mentioned, often in combination with each package's custom (non-learning) behavior editor. All but one of the websites did mention the presence of an application programming interface (API), meaning that machine learning is likely possible through these interfaces (depending on the level of simulation control offered by the interfaces), but also meaning that each machine learning solution has to be developed from scratch. The effort going into the development of these solutions is effort that cannot be put into the research of actual applications.

Related to CGF packages are the standalone behavior editors, which provide alternative behavior editing capabilities for the CGF generated by the CGF packages. These editors function by connecting to specific CGFs by means of the provided APIs. In the realm of standalone CGF behavior editors, the only editor we are aware of that provides built-in machine learning capabilities is SimBionic, developed by Stottler Henke Associates, Inc. (Ludwig & Presnell, 2019). SimBionic incorporates the dynamic scripting algorithm (Spronck et al., 2006), which uses an adaptive rule-selection mechanism.

Table 1. Mention of "machine learning" (ML), "artificial intelligence" (AI), and "API" on the websites of nine COTS/GOTS CGF packages (in no particular order).

Product Name	***Mention of ML***	*Mention of AI*	*Mention of an API*	*Reference*
Presagis STAGE	**no**	yes	yes	(Presagis, 2021)
VR-Forces - MAK Technologies	**no**	yes	yes	(MAK Technologies, 2021)
MASA SWORD	**no**	yes	yes	(MASA Group, 2020)
VBS4 | BISim	**no**	yes	yes	(BISim, 2021)
One Semi-Automated Forces (OneSAF)	**no**	no	yes	(Leidos, 2019)
DirectCGF	**no**	yes	yes	(Diginext, 2021)
FLAMES	**no**	no	yes	(Ternion, 2021)
Steel Beasts Pro	**no**	no	no	(eSim Games, 2021)
Next Generation Threat System (NGTS)	**no**	yes	yes	(Naval Air Warfare Center Aircraft Division (NAWCAD), 2018)

CHALLENGES

In this section an overview is given of current challenges that are seen as potential obstacles for more widespread use of RL technology in CGFs for military simulations. We categorize them into three perspectives. The first perspective takes an industry view on the software requirements for incorporating RL techniques. The second one focuses on the conceptual design of CGFs with learning capabilities. The third one addresses the deployment in M&S (eco-) systems.

Perspective 1: Paradigm Shift in CGF Behavior Modeling

In current industry practice, CGF behaviors are mostly developed using traditional software engineering, using hand-crafted, rule-based techniques such as scripts, state machines or behavior trees. The introduction of RL techniques calls for a different approach towards problem solving and leads to a paradigm shift in the manner in which CGF behavior models can be developed.

Wan et al. (2019) highlighted the significant differences between the development of ML and non-ML systems regarding software engineering aspects and work practices such as requirement acquisition, software design, testing, required engineering skills and specialized tooling. Where in current industry CGF tools, techniques and user-friendly tools for traditional behavior modeling have been maturing in the last decade or so, in the case of RL, standardized techniques and end-user tools are currently non-existent. As in many other domains, in the military M&S domain, the field of RL has not yet proven to be mature enough for applications, or commercially viable. Below two crucial challenges are identified that accompany the paradigm shift in CGF behavior engineering.

Challenge: The need for a training system

In contrast to traditional engineering where we program *how* to achieve a behavior by manually crafting the rules of decision-making (viz. imperative programming), in RL we specify *what* the behavior is that we want to achieve and let the machine come up with a solution through training. Training a CGF to exhibit certain behaviors (tasks, knowledge, skills or abilities) requires a training system. Figure 1 provides a high-level conceptual view of such a training system. It shows the core functions that are applicable for training any learning algorithm.

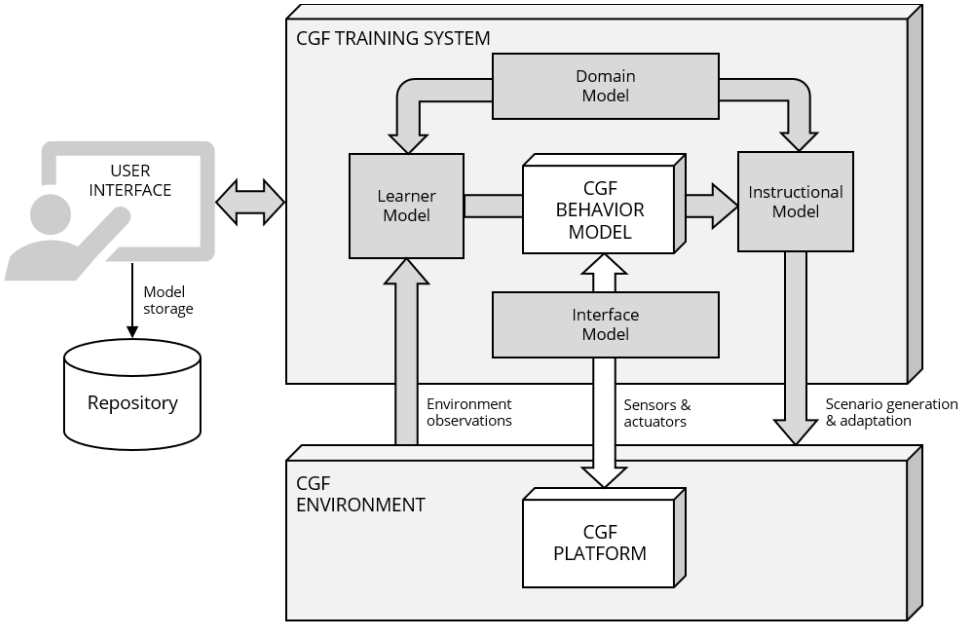

Figure 1. Training system for training a CGF behavior model (top center) using a reinforcement learning algorithm to control a CGF in an environment.

In defining the training system we borrow concepts from the field of adaptive instructional systems (AIS). An AIS is a computer-based system to train (human) learners by guiding learning and tailoring instruction in the context of

certain learning objectives (Sottilare, 2019). Although AISs are originally intended for human learners, their underlying concepts are applicable to computational learners as well (van Oijen et al., 2021).

The four core functions of a training system are:

- Domain model: Encompasses all (expert) knowledge about the behavior we want the CGF to exhibit. It offers a performance standard that can be used to judge the 'correctness' of a behavior. In RL, these include performance indicators to assess behavior as 'good' or 'bad'.
- Learning model: Keeps track of the learning progress. It is responsible for measuring the real-time performance during training and evaluating the behavior by comparing the current performance with the performance standard. As an external critic, it requires observing the learner's behavior in the environment to make objective judgements.
- Instructional model: Continuously processes the evaluations and applies instructional interventions. Instructional actions include providing direct feedback to the learning algorithm (the reward signal in RL terminology), or adapting the training environment to control the learning experiences offered to the learner.
- Interface model: Defines the required inputs and outputs for the learning algorithm. These correspond to the sensors and actuators of a CGF platform that are necessary to successfully accomplish a behavior.

The user interface shown in Figure 1 is a hypothetical interface of a tool by which users can design and run training experiments to teach CGFs specific behaviors, and then store learned behaviors in a repository for later use.

Challenge: The changing role of the environment

As is commonly said, machine learning requires a large amount of data and computational power. In RL, the data comes from a runtime simulation environment where a CGF can be exposed to learning experiences. The environment therefore becomes an integral part of the modeling (training) process. This is reflected by the interactions between the training system and environment shown in Figure 1. For the environment, it is paramount that it supports high performance computing which is generally required for any type of machine learning: (1) The environment should support fast-time simulation in order to train behaviors within a reasonable time span; (2) The environment should allow for external control regarding generating, initiating, adapting and restarting scenarios (i.e. controlling learning experiences); (3) One would benefit from environments that are easily replicable and support parallel execution in order to make use of distributed RL (Samsami & Alimadad, 2020).

In the current state of the M&S industry, we see that many simulation systems are not yet prepared to meet the computing requirements required for RL. From practical experience it was experienced that certain CGF platforms can become inaccurate in their simulation when performing in faster-than-real time mode (e.g. deviations in the environment state are observed when running in different time modes, as described by Fiedler (2004)); or platforms were found unsuited because of their inability to speed up time or to disable computationally expensive 3D rendering.

Perspective 2: CGF Development

Learning algorithms can be used by CGFs to learn specific (tactical) tasks or missions that are representative for the military platform they represent (e.g., a fighter aircraft engaging an enemy, a frigate escorting a civilian vessel or soldiers setting up an ambush). For the modeler of a CGF there are two primary design concerns: (1) what learning algorithms are best suited (or fit-for-purpose) to learn specific tasks or behaviors and (2) how to apply the algorithm to the CGF domain and incorporate it into a specific CGF model for execution.

Challenge: The selection of fit-for-purpose learning algorithms

In machine learning, there is no one-size-fits-all algorithm. Choosing a suitable algorithm entails understanding the cognitive abilities that underly the task to be learned. In Figure 2, a rough categorization is shown of behavioral abilities, divided between perception and decision-making, and further divided into low-level and high-level cognition. Different algorithms suit different abilities. For instance, a neural network based RL algorithm may be better suited for lower-level tactical execution, such a platform maneuvering and control (Zhang et al., 2018); and a rule-based RL algorithm can be suited for more high-level strategic and tactical decision-making. If the task is related to obtaining good situational awareness or understanding, other algorithms come into play such as Deep Learning or Bayesian networks (Chakraborty et al., 2017). It has also been demonstrated that certain algorithms can excel at both high-level and low-level cognition. For instance, Berner et al. (2019) use a deep RL model is used to optimize real-time strategy games, requiring both long term strategic planning and short-term tactical and reactive behaviors.

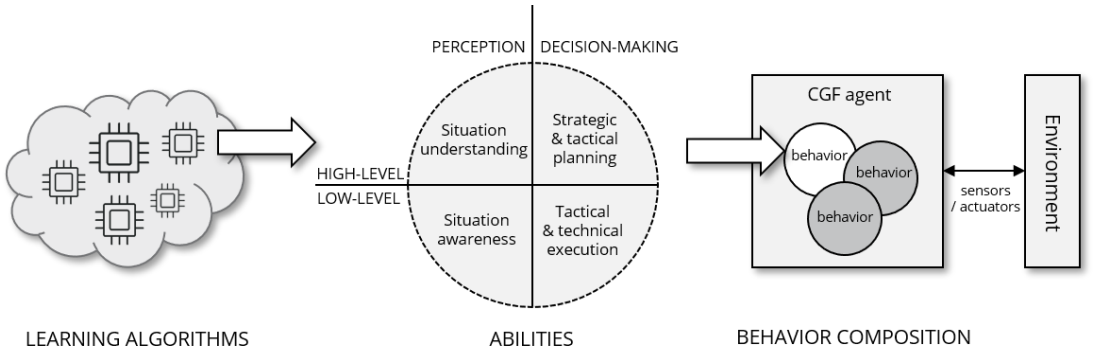

Figure 2. Conceptual CGF Design: Learning algorithms (left) fulfill behavioral abilities (center) that are composed to form a CGF (right).

Besides algorithmic capabilities there are other considerations that can affect the choice for a particular algorithm. For instance, should the algorithm allow the encoding of specific expert knowledge rules such as Tactics, Techniques and Procedures (TTP) or Rules of Engagement (RoE)? Or should the algorithm offer some level of explainability and transparency for its reasoning process?

Furthermore, it is important to recognize that a ML approach towards CGF behavior modeling is not the go-to answer for all modeling problems. The benefits of ML can also be seen as drawbacks for specific applications. For instance, consider a training simulation where an instructor wants to precisely control a CGFs behavior to achieve a specific learning situation and ensure consistency in training; or where CGFs need to follow human-designed procedures and tactics 'according to the book'. The more one needs to constrain behaviors to adhere to specific (human) rules, the less freedom there is for machine learning to explore alternative or creative solutions. It is expected that future CGF models will benefit from hybrid models that combine both symbolic and sub-symbolic approaches to find the right balance between human control on the one hand, and automated behavior modeling on the other hand.

Challenge: The composition of a CGF model

In the case where a single learning algorithm is not able to cover the full scope of a CGF's behaviors, some form of behavior composition is necessary. In (human) behavior modelling, agents are often decomposed to cope with modeling complex behaviors. Different paradigms for composition exist: *Hierarchical composition* is used to break down an agent's decision-making process into sub-behaviors, based on notions such as goals, sub-goals, plans or action sequences. This paradigm is found in many control techniques such as (hierarchical) state machines, behavior trees or belief-desire-intention (BDI) approaches. In *functional composition*, an agent is composed of specific cognitive functions, such as perception, memory, reasoning or motor control. For instance in (Lewis et al., 2019), a reference architecture is proposed for human behavior models from a functional point of view. Finally, in *technological composition*, an agent is able to combine and bridge the gap between different programming languages or software platforms. E.g. in (Warwick & Rodgers, 2019) a composition approach is described that allows combining different formalisms such as models developed in ACT-R, MATLAB or R.

Perspective 3: Deployment in M&S ecosystems

In order to benefit from trainable CGFs in actual defense applications, they need to be integrated in broader M&S ecosystems. Ideally, such CGFs can be easily trained, reused and deployed alongside other M&S components in environments such as LVC, MTDS, or Battle Labs. In current practices, it is seen that RL practitioners often work in isolated environments and that their algorithms and learned behaviors cannot easily be shared and reused by others. This prevents the establishment of a collaborative community where practitioners can complement each other, compare and evaluate each other's models and build upon previous successes.

Challenge: Service-oriented CGF models

In the M&S industry, the paradigm of Modeling & Simulation as a Service (MSaaS) has been found to be a good fit for promoting sharing and reuse of simulation models (Siegfried & van den Berg, 2015). In alignment with such service-oriented approaches, when considering CGFs as-a-service, different "service levels" can be considered. These are illustrated in Figure 3 and explained further below.

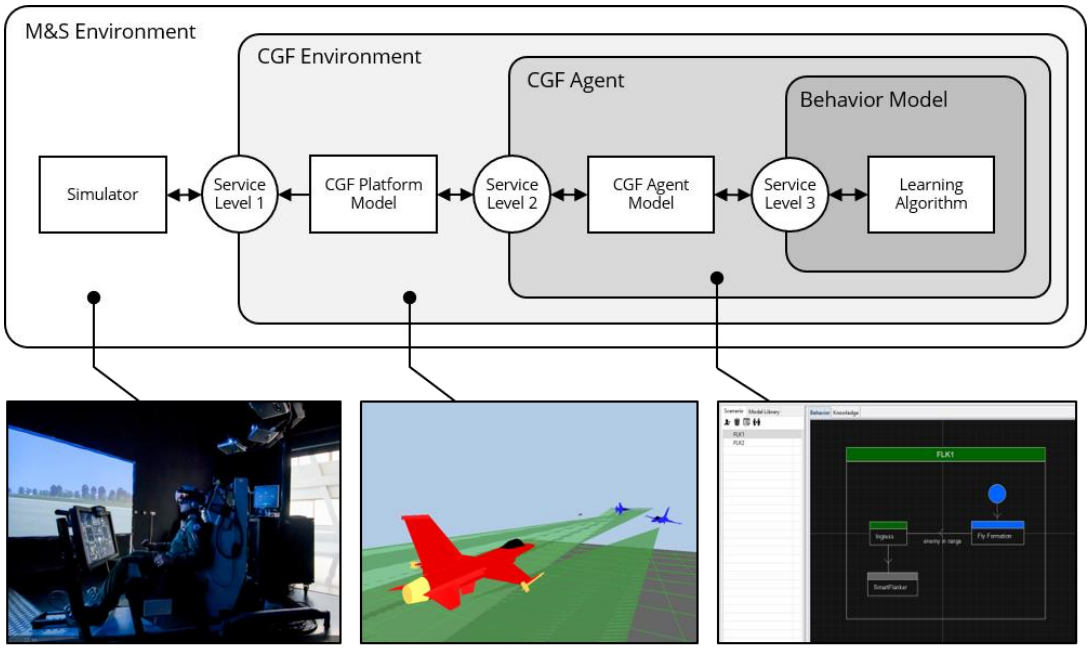

Figure 3. Service-oriented approaches for CGFs. The three screenshots at the bottom show examples of an M&S environment as a human-in-the-loop simulator (left), a CGF environment as a CGF package (middle) and a CGF agent modeled in a specialized behavior modeling tool (right).

We identify three service levels:

- Service level 1: At this level, CGFs are fully 'plug'n'play' and can be injected into M&S environments using common interoperability standards such as DIS or HLA. This is similar to how current COTS/GOTS CGF platforms provide their CGFs to external M&S systems.
- Service level 2: At this level, we separate the CGF agent from its physical embodiment in the CGF environment. This is a common abstraction for specialized behavior modeling tools that can be coupled to different CGF environments. An agent model is provided as a service that encompasses all decision-making and that can interface with a CGF's sensors and actuator models (such as navigation, sensor or weapon models). Standards for such an interface are scarce, but two examples are LLBML (Alstad et al., 2013) and its successor NETN-ETR, modelled as an HLA FOM module.
- Service level 3: At this level, individual learning algorithms themselves can be offered as a service and integrated into a CGF agent, operating alongside other behavioral processes through one or more composition paradigms (as described earlier). Minimal interfaces to any learning algorithm include e.g. input (observations) and output (actions) streams and feedback signals to learn from. A standard for such a service would allow abstracting away from specific RL algorithms and allow the replacement of algorithms (e.g. improved algorithms have become available) without impacting the CGF agent design. Specific technical solutions exist for common interfaces between RL algorithms and environments (such as OpenAI Gym).

Challenge: Interoperability & transfer of models

The service-oriented approaches illustrate how reusability can be achieved for individual behaviors, agents or CGFs. In reference to different levels of interoperability, the associated interfaces of these services can ensure interoperability at the *syntactic level* (agreement on message formats) and *semantic level* (agreement on the meaning of messages). However, specifically for RL-based CGFs, there is the danger of model transfer issues that prevents interoperability at the *pragmatic level* (consistency in the *effects* of exchanged messages).

In RL, transfer issues arise when a learned model behaves differently (i.e. leads to different effects) when it is deployed in an environment that is different from the one that was used during training. Differences could be caused by the specific implementation of a CGF's navigation, sensor or weapon models, or externally caused by the environment, such as different fidelity of dynamics simulations. I.e., there is the danger of 'overfitting' a learning algorithm to a specific training environment, which can lead to unexpected behaviors when applied to other environments.

Environment transfer is a common problem in RL when transferring from a source environment (where a model is trained) to a target environment (where a model is deployed). It is often recognized in systems where simulations are used to train behaviors that are to be deployed in the real-world (such as autonomous vehicles or robots). In the robotics domain, a proposed approach is the use of domain randomization (Peng et al., 2018). It is based on the idea of varying environment physics *during* training, leading to better generalization of a behavior.

It is hard to predict whether transfer issues will arise in a specific CGF simulation and this may only be detected through thorough testing and evaluation in the desired target environment. However, when transfer issues do arise and lead to wrong or non-optimal behaviors, there should be the ability to retrain a behavior using the desired target environment as the new training environment.

PRACTICAL IMPLICATIONS

In a research context, we have experimented with practical solutions for the challenges mentioned above. In this section, we describe the ongoing work. The research question guiding the experiments is: how can we exploit RL technology in a CGF tool, and demonstrate several RL behaviors that are reusable in M&S systems for training or decision-support applications? We restricted our experiments to the air-to-air combat domain.

In this domain, the focus has been on learning two different CGF behavior tasks. The first task is a formation flying task that requires an agent's ability to assume and keep some (configurable) relative position to another (leading) aircraft. When applied to multiple agents, a wide variety of formation types can be created with varying numbers of aircraft. The motivation for using machine learning for these tasks is to eliminate the effort of scenario developers to manually hand-craft the required maneuvers, which has been found to be tedious when building training scenarios. The second task is an engagement task which is about an agent's ability to engage and eliminate a target enemy aircraft. It includes offensive and defensive tactical maneuvers that are to be optimized by a RL algorithm. We envision the use of such a task in a decision support application for tactics development. When applied to blue forces, it can be used to explore one's own new tactics against an enemy, or when applied to red forces, it can be used to test the quality of existing blue tactics against an enemy trying to find weaknesses in those tactics.

In the remainder of this section we shortly revisit each challenge that was identified in the previous section and report on our experiences in addressing them.

Implementing a training system

First we developed a rudimentary training system for training CGF tasks, based on the design presented in Figure 1. In RL algorithms, most functions of the training system are captured by the *reward function* (see Figure 4). This function evaluates a CGF's behavior by comparing metrics of desired performance and measured performance. For instance, for the 'formation' task, performance is judged by a distance and bearing measure; or for the 'engagement' task, a variety of measures are used such as win/loss, time spend or missiles left. The output value of the reward function comprises a judgement on the strength of the positive or negative feedback, which is signaled to the learning algorithm. In RL the result of the reward function is called the *reward signal*, and is often a numerical value between -1 and 1.

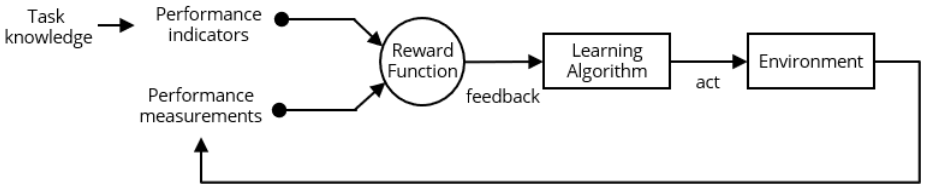

Figure 4. Reward function for reinforcement learning

When relating the RL operation to the core functions of a training system (Figure 1), the performance indicators can be seen as a representation of the *domain model*, the measurements and evaluation in the reward function as the *learning model* and the final judgement on the feedback value as the *instructional model*. In addition (and part of the instructional model), we added conditions that can be defined for a task to trigger environment adaptations. This feature proved to be crucial for fast learning as it enabled generating new scenarios to ensure a continuous stream of

relevant learning experiences (e.g., there is no use in continuing learning an engagement task when there are no more missiles left).

In our experience, many functions of the training system could be developed independently of a specific RL algorithm. However, only the part of the reward function that calculates the precise feedback value to send to the algorithm can be highly dependent on the algorithm used. In RL, a strategy for feedback design is commonly tailored to the specific type of algorithm. Still, we were able to put in place a general training system to train tasks, while minimizing dependencies on too specific algorithmic learning technology.

The environment in support of machine learning

To study the new role of the environment (i.e., becoming an environment for learning, as well as for execution), we revamped an existing in-house developed, lightweight simulation environment (called LWACS – light-weight air combat simulator) and optimized it for machine learning. Important programming 'hooks' were implemented that allowed us to efficiently start, reset and adapt scenarios (as required by the training system). Furthermore, the environment supports faster-than-real-time (and as fast as possible) and can run in the background without visual rendering. The added flexibility and efficiency enabled quick iteration upon experiments, as opposed to earlier efforts where we were dependent on COTS CGF platforms that lacked some of these capabilities. In the future our goal is to implement additional deployment strategies involving parallelization of the environment, as well as containerization.

Selecting fit-for-purpose learning algorithms

Driving the experiments was the desire to train CGF behaviors by means of two distinct RL techniques applied to different tasks. The first technique, deep reinforcement learning (DRL), is a neural network based technique capable of discovering highly effective sequences of actions. It was found suited for low-level platform maneuvering and was used for the 'formation' task. The second technique, dynamic scripting (DS), is a rule-based learning technique that recombines a set of pre-written rules into useful scripts. It was found suited for tasks that require more tactical decision-making (as opposed to more low-level maneuvering) and was used for the 'engagement' task. Where DRL commonly requires long training times, DS is computationally lightweight, but the creativity of DS is restricted by the variety in the pre-written rules.

The composition of a CGF model

The behaviors that were trained with RL algorithms were integrated into a larger structure consisting of a CGF agent that uses a rule-based method to orchestrate top-level decision-making (using a hierarchical finite-state machine). A description of this composition approach falls outside the scope of this paper and has been described in (van Oijen et al., 2019).

Service-oriented CGFs

An important goal in our experiments was the ability to deploy pre-trained behavior models in broader M&S eco-systems. In the previous section, three service-oriented approaches were discussed. These were all addressed:

Service level 1 (CGF integration)

The LWACS environment that was used for training the CGF behaviors included a DIS-adapter. This allowed us to inject CGFs with pre-trained behaviors to any external DIS-capable system.

Service level 2 (agent integration)

We implemented a standard interface for controlling fighter aircraft. It defines messages to exchange sensors data and control instructions between a CGF fighter platform in an environment and a CGF agent controlling the platform. These messages relate to systems such as the navigation systems (e.g. positioning information and flight instructions); weapon systems (e.g. munition availability and launch instructions) and radar systems (e.g. radar contacts and target locking). Several CGF package connections were available that supported this interface, including our own LWACS, but also STAGE and VR-Forces (see Table 1). With these connections we were able to train RL behavior in one environment (the LWACS), and still deploy them in other platforms that might not be suited for ML training.

Service level 3 (algorithm integration)

We developed a generic interface protocol for RL algorithms such that different algorithms could be integrated in a more service-oriented manner. The goal was that the consumer side of the algorithm (i.e. the CGF agent) merely has

to select which learning algorithm to use for a specific task, without requiring knowledge about the algorithm's internal workings. Training the algorithm would then be handled by the training system described earlier.

The developed protocol was implemented using gRPC, a remote procedure call system that supports a wide variety of client programming languages. The advantage was that it easily allowed us to integrate different algorithms written in different languages (Python for the DRL algorithm and C# for the DS algorithm). However, the design of the interface raised some technical questions regarding a format for the input (features) and output (action) space for RL algorithms. For example, neural networks operate with a fixed length input which makes it problematic to represent variable information (such as varying number of radar contacts). In DS, the consumption of input is only restricted by the rules defined in the algorithm's scripting language. Similar for the action space, some algorithms operate with discrete actions, while other algorithms can select multiple, continuous actions at once. Other protocol messages were found to be generally useable, such as the current mode of operation (training mode or execution mode), feedback signals and model saving and loading facilities.

Transfer of models

The potential issues of model transfer have not been explored in detail. Transfer issues relate to the possible unexpected behavior that may arise when deploying a pre-trained CGF behavior in an environment that is different from the one it was trained in. In our experiments, transfer issues did not arise since our ML-optimized training environment was able to deploy CGFs directly using DIS (service level 1). The investigation of transfer issues for service level 2 is left for future work (e.g. training a behavior for a CGF platform in the LWACS environment and deploying it to a CGF platform in a STAGE or VR-Forces environment).

CONCLUSION

In this paper we have sketched the potential (future) benefits of CGFs with learning capabilities for M&S applications such as training and decision support. We addressed the question of why machine learning technologies (in particular reinforcement learning) for CGFs are currently not yet more widely available in the M&S industry and why research demonstrations beyond toy examples still seem to be scarce.

Six challenges were identified, which we can formulate as requirements:

- *The need for a training system*: the ability of CGF tools to provide a RL training infrastructure to train new CGF behaviors (defining training objectives, measuring performance indicators and implementing instructional strategies).
- *Environment requirements:* the ability of simulation environments to support high performance computing requirements for efficient RL training (e.g. high magnitudes of faster-than-real-time simulation updates).
- *Fit-for-purpose algorithms:* the ability to fit a suited learning algorithm to the specific task to be learned, that is, an algorithm that is qualified to master the behavioral abilities that underly the task.
- *CGF behavior compositionality:* the ability of a learning algorithm to operate alongside other behaviors in a larger whole that represents a complete agent.
- *Service-oriented deployment*: the ability to access CGFs with pre-trained (or trainable) behaviors in a service-oriented manner such that they can be deployed within broader M&S eco-systems.
- *Model transfer:* the ability to successfully transfer pre-trained behaviors to new environments.

Lessons learned were gathered for the above challenges based on a pragmatic goal and implementation of CGFs that can be trained to learn different tasks using different learning algorithms, and are deployable into different M&S applications.

The insights of this paper are specifically geared towards the *exploitation* of RL technology in CGF simulations, rather than on the quality and performance of specific RL algorithms. It has to be recognized that reinforcement learning is still in its infancy when applied to solving complex tasks in comprehensive environments. Current state-of-the-art algorithms still lack in many regards when compared to human intelligence and the efficiency of human learning (Lake et al., 2017). However, the field has gained much traction in the last decade and new and improved algorithms continue to arise. When they do, the industry needs to be ready to exploit them as soon as they become available. The insights given in this paper provide directions towards this goal.

REFERENCES

Allen, B., Agarwal, S., Kalpathy-Cramer, J., & Dreyer, K. (2019). Democratizing AI. *Journal of the American College of Radiology*, *16*(7), 961–963.

Alstad, A., Mevassvik, O. M., Nielsen, M. N., Løvlid, R., Henderson, H., Jansen, R. E. J., & de Reus, N. M. (2013). Low-level battle management language. *Proceedings of the 2013 Spring Simulation Interoperability Workshop, No. 13S-SIW-032*.

Aronsson, S., Artman, H., Mitchell, M., Ramberg, R., & Woltjer, R. (2020). LVC Allocator: Aligning training value with scenario design for envisioned LVC training of fast-jet pilots. *The Journal of Defense Modeling and Simulation*, 1548512920958079.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., & Mordatch, I. (2020). *Emergent Tool Use From Multi-Agent Autocurricula*.

Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., & others. (2019). Dota 2 with large scale deep reinforcement learning. *ArXiv Preprint ArXiv:1912.06680*.

BISim. (2021). *VBS4 | BISim*. Bohemia Interactive Simulations. https://bisimulations.com/products/vbs4

Chakraborty, S., Preece, A., Alzantot, M., Xing, T., Braines, D., & Srivastava, M. (2017). Deep learning for situational understanding. *2017 20th International Conference on Information Fusion (Fusion)*, 1–8.

Diginext. (2021). *DirectCGF - Home*. https://www.directcgf.com/

eSim Games. (2021). *ESim Games – Vehicle-centric Combined Arms Combat Tactics and Gunnery Simulation*. https://www.esimgames.com/

Evensen, P.-I., & Bentsen, D. H. (2016). Simulation of land force operations—A survey of methods and tools. In *2015/01579*. https://ffi-publikasjoner.archive.knowledgearc.net//handle/20.500.12242/1104

Evensen, P.-I., Martinussen, S. E., Halsør, M., & Bentsen, D. H. (2019). Wargaming Evolved: Methodology and Best Practices for Simulation-Supported Wargaming. In *0970977808*. https://ffi-publikasjoner.archive.knowledgearc.net//handle/20.500.12242/2742

Fiedler, G. (2004, June 10). *Fix Your Timestep!* Gaffer On Games. https://gafferongames.com/post/fix_your_timestep/

Gibson, Z., Butterfield, J., Ferguson, R. S., Rafferty, K., Yu, W., & Casement, A. (2020). Assessing Current HMI Designs and Exploring AI Potential for Future Air-Defence System Development. In S. Yamamoto & H. Mori (Eds.), *Human Interface and the Management of Information. Interacting with Information* (Vol. 12185, pp. 305–323). Springer International Publishing. https://doi.org/10.1007/978-3-030-50017-7_22

Keuning, M. F. R. (2010). *Embedded Training and LVC*. National Aerospace Laboratory NLR. https://reports.nlr.nl/handle/10921/185

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, *40*.

Lee, K., Lee, G., & Rabelo, L. (2020). A Systematic Review of the Multi-Resolution Modeling (MRM) for Integration of Live, Virtual, and Constructive Systems. *Information*, *11*(10), 480.

Leidos. (2019). *One Semi-Automated Forces (OneSAF®)*. One Semi-Automated Forces (OneSAF) Factsheet. https://www.leidos.com/sites/g/files/zoouby166/files/2019-10/FS-OneSAF-Overview-Leidos.pdf

Lewis, M., Alexander, T., Huiskamp, W., & Blais, C. L. (2019). *A Reference Architecture for Human Behaviour Representation* (MSG-127). NATO STO.

Ludwig, J., & Presnell, B. (2019). Developing an Adaptive Opponent for Tactical Training. In R. A. Sottilare & J. Schwarz (Eds.), *Adaptive Instructional Systems* (pp. 532–541). Springer International Publishing. https://doi.org/10.1007/978-3-030-22341-0_42

Luotsinen, L. J., Kamrani, F., Hammar, P., Jändel, M., & Løvlid, R. A. (2016). Evolved creative intelligence for computer generated forces. *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 3063–3070. https://doi.org/10.1109/SMC.2016.7844707

MAK Technologies. (2021). *VR-Forces—MAK Technologies*. https://www.mak.com/products/simulate/vr-forces

MASA Group. (2020, May 29). *DEFENSE*. MASA Group. https://masasim.com/en/notre-metier/defense/

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., & others. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Naval Air Warfare Center Aircraft Division (NAWCAD). (2018). *Next Generation Threat System*. https://www.navair.navy.mil/nawctsd/sites/g/files/jejdrs596/files/2018-11/2018-ngts.pdf

Peng, X. B., Andrychowicz, M., Zaremba, W., & Abbeel, P. (2018). Sim-to-Real Transfer of Robotic Control with Dynamics Randomization. *2018 IEEE International Conference on Robotics and Automation (ICRA)*. https://doi.org/10.1109/icra.2018.8460528

Perry, T. S. (2021, May 3). *Andrew Ng X-Rays the AI Hype—IEEE Spectrum*. IEEE Spectrum: Technology, Engineering, and Science News. https://spectrum.ieee.org/view-from-the-valley/artificial-intelligence/machine-learning/andrew-ng-xrays-the-ai-hype

Pope, A. P., Ide, J. S., Micovic, D., Diaz, H., Rosenbluth, D., Ritholtz, L., Twedt, J. C., Walker, T. T., Alcedo, K., & Javorsek, D. (2021). Hierarchical Reinforcement Learning for Air-to-Air Combat. *ArXiv:2105.00990 [Cs]*. http://arxiv.org/abs/2105.00990

Presagis. (2021). *Presagis STAGE*. https://www.presagis.com/en/product/stage/

Roessingh, J. J., Toubman, A., Oijen, J. van, Poppinga, G., Løvlid, R. A., Hou, M., & Luotsinen, L. J. (2017). Machine learning techniques for autonomous agents in military simulations—Multum in parvo. *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 3445–3450. https://doi.org/10.1109/smc.2017.8123163

Samsami, M. R., & Alimadad, H. (2020). Distributed Deep Reinforcement Learning: An Overview. *ArXiv*, *abs/2011.11012*.

Schwartz, P. J., O'Neill, D. V., Bentz, M. E., Brown, A., Doyle, B. S., Liepa, O. C., Lawrence, R., & Hull, R. D. (2020). AI-enabled wargaming in the military decision making process. In T. Pham, L. Solomon, & K. Rainey (Eds.), *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II* (Vol. 11413, pp. 118 – 134). SPIE. https://doi.org/10.1117/12.2560494

Siegfried, R., & van den Berg, T. (2015). M&S as a Service: Paradigm for Future Simulation Environments. *Proceedings of the 2015 Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*.

Smith, R. E., Dike, B. A., Ravichandran, B., El-Fallah, A., & Mehra, R. K. (2001). Two-Sided, Genetics-Based Learning to Discover Novel Fighter Combat Maneuvers. *Applications of Evolutionary Computing, EvoWorkshops 2001: EvoCOP, EvoFlight, EvoIASP, EvoLearn, and EvoSTIM, Como, Italy, April 18-20, 2001, Proceedings*, 233–242. https://doi.org/10.1007/3-540-45365-2_24

Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I., & Postma, E. (2006). Adaptive game AI with dynamic scripting. *Machine Learning*, *63*(3), 217–248. https://doi.org/10.1007/s10994-006-6205-6

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Ternion. (2021). *FLAMES Modeling & Simulation software by Ternion*. FLAMES Modeling & Simulation Software by Ternion. https://www.ternion.com/

Toubman, A. (2019). Validating Air Combat Behaviour Models for Adaptive Training of Teams. In R. A. Sottilare & J. Schwarz (Eds.), *Adaptive Instructional Systems* (pp. 557–571). Springer International Publishing. https://doi.org/10.1007/978-3-030-22341-0_44

Toubman, A., Poppinga, G., Roessingh, J. J., Hou, M., Luotsinen, L., Løvlid, R. A., Meyer, C., Rijken, R., & Turčaník, M. (2015). Modeling CGF Behavior with Machine Learning Techniques: Requirements and Future Directions. *Proceedings of the 2015 Interservice/Industry Training, Simulation, and Education Conference*, 2637–2647.

van den Bosch, K., Blankendaal, R., Boonekamp, R., & Schoonderwoerd, T. (2020). Adaptive Agents for Fit-for-Purpose Training. In C. Stephanidis, D. Harris, W.-C. Li, D. D. Schmorrow, C. M. Fidopiastis, P. Zaphiris, A. Ioannou, X. Fang, R. A. Sottilare, & J. Schwarz (Eds.), *HCI International 2020 – Late Breaking Papers: Cognition, Learning and Games* (Vol. 12425, pp. 586–604). Springer International Publishing. https://doi.org/10.1007/978-3-030-60128-7_43

van Oijen, J., Toubman, A., & Claessen, O. (2021). Teching Reinforcement Learning Agents with Adaptive Instructional Systems. In R. A. Sottilare & J. Schwarz (Eds.), *Adaptive Instructional Systems. Design and Evaluation* (pp. 1–17). Springer International Publishing.

van Oijen, J., Toubman, A., & Poppinga, G. (2019). Effective Behaviour Modelling for Computer Generated Forces. *Interservice/Industry Training, Simulation and Education Conference (I/ITSEC)*.

Virtual AlphaDogfight Trials Finals (Archived). (n.d.). Retrieved May 11, 2021, from https://www.darpa.mil/news-events/virtual-alphadogfight-trials-finals

Wan, Z., Xia, X., Lo, D., & Murphy, G. C. (2019). How does Machine Learning Change Software Development Practices? *IEEE Transactions on Software Engineering*, 1–1. https://doi.org/10.1109/TSE.2019.2937083

Warwick, W., & Rodgers, S. (2019). Wrong in the right way: Balancing realism against other constraints in simulation-based training. *International Conference on Human-Computer Interaction*, 379–388.

Zaroukian, E., Basak, A., Sharma, P. K., Fernandez, R., & Asher, D. E. (2021). Emergent reinforcement learning behaviors through novel testing conditions. *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III*, *11746*, 117460T. https://doi.org/10.1117/12.2585627

Zhang, X., Liu, G., Yang, C., & Wu, J. (2018). Research on air confrontation maneuver decision-making method based on reinforcement learning. *Electronics*, *7*(11), 279.