Proceedings of the 2023 Winter Simulation Conference
C. G. Corlu, S. R. Hunter, H. Lam, B. S. Onggo, J. Shortle, and B. Biller, eds.

INCORPORATION OF MILITARY DOCTRINES AND OBJECTIVES INTO AN AI AGENT VIA NATURAL LANGUAGE AND REWARD IN REINFORCEMENT LEARNING

Michael Möbius
Daniel Kallfass
Matthias Flock

Airbus Defence and Space GmbH
Dept. Operational Analysis and Studies
Claude-Dornier-Strasse
88090 Immenstaad, GERMANY

Thomas Doll

Joint Support Service Command
Abt Plg, UAbt PlgStrg/Int
Fontainengraben 150
53123 Bonn, GERMANY

Dietmar Kunde

German Army Headquarters
von-Hardenberg-Kaserne
Prötzeler Chaussee 25
15344 Strausberg, GERMANY

ABSTRACT

This paper emphasizes the integration of sound tactical behavior in the generation of realistic military simulations, which includes the definition of combat tactics, doctrine, rules of engagement, and concepts of operations. Recent advances in reinforcement learning (RL) enable RL agents to generate a wide range of tactical actions. A multi-agent ground combat scenario is used in this paper to demonstrate how a machine learning (ML) application generates strategies and issues commands while following a given objective. Natural language is used to issue doctrines and objectives to improve communication between the human advisor and the ML agent. This allows us to embed objectives and existing doctrines into the reasoning of an artificial intelligence (AI). The research demonstrates the successful integration of natural language to enable an agent to achieve different objectives. This groundwork will enhance RL agents' ability in the future to uphold the doctrines and rules of military operations.

1 INTRODUCTION

In the past years, deep reinforcement learning (Sutton et al. 1998) has become a popular technique in the field of machine learning for solving complex tasks. Testing the usability of AI systems in a realistic environment has been crucial since their development. The gaming industry has a significant interest in utilizing AI in commercial games to attract more buyers. As games become more sophisticated and demanding, they also become more challenging to play, especially for a single player. Therefore, games have become an essential test environment to evaluate the performance of AI systems.

Extensive research has been conducted to achieve superhuman performance in-game environments. The incorporation of deep learning models into RL by Mnih et al. (2013) was a significant breakthrough in this field. This approach enabled AI models to play Atari games and handle high-dimensional input. Recent advances in training reinforcement learning agents to play classic board games like Go (Silver et al. 2016) and complex games such as StarCraft II (Vinyals et al. 2019) have demonstrated the success of deep RL.

In parallel, RL has also been applied to the military domain in several ways. RL can be used to train agents to perform various tasks in military simulations and has been applied to use cases such as:

- Battlefield decision-making: RL can be used to train agents to make decisions in complex military scenarios (Doll et al. 2021). The agents can learn to take actions that maximize a reward signal (e.g., completing a mission objective, minimizing casualties), based on their observations of the environment (e.g., enemy positions, terrain).
- Autonomous systems / vehicle control: RL can be used to train agents to control military vehicles (e.g., drones, tanks) in simulations (Möbius et al. 2022). The agents can learn to navigate the vehicle in the environment and perform various tasks (e.g., reconnaissance, and target acquisition).
- Logistics planning: RL can be used to optimize logistics planning in military simulations (Yan et al. 2021). The agents can learn to allocate resources (e.g., troops, supplies) to different areas of the battlefield to achieve mission objectives while minimizing losses.
- Cybersecurity: RL can be used to train agents to detect and respond to cyber-attacks in military simulations (Vyas et al. 2023). The agents can learn to identify and mitigate threats to military networks and systems.
- Training and evaluation: RL can be used to train and evaluate military personnel in simulations (Salas et al. 2003). The agents can simulate different scenarios and provide feedback on the actions taken by the trainees.

Overall, RL can be applied to military simulations to enhance the training and evaluation of military personnel, optimize resource allocation and decision-making, and improve overall military effectiveness. However, there are several reasons why units trained with RL may not behave realistically in military simulations and do not completely fulfill constraints given by common tactics, techniques, and procedures (TTPs) or rules of engagement:

- Limited training data: RL requires a large amount of training data to learn complex behaviors. However, in military simulations, generating enough training data to capture the full range of realistic behaviors that units should exhibit may be difficult. As a result, the trained agent controlling the units may exhibit only a subset of realistic behaviors.
- Limited exploration: RL agents explore the environment to learn about the consequences of different actions. However, in military simulations, there may be constraints on the actions that units can take, which limits the exploration of the RL agent. As a result, the agent may not learn about all the possible actions that units can take, leading to suboptimal behavior.
- Ill-defined rewards: RL agents are typically trained to maximize a reward signal, which is defined by the designer of the simulation. If the reward signal is oversimplified or does not capture all the relevant aspects of the task, the trained units may exhibit unrealistic behavior. For example, if only reward is assigned to units for eliminating enemies but not for avoiding casualties or completing objectives, the trained units may prioritize eliminating enemies over other important tasks.
- Model bias: RL agents learn from the data that they are trained on, and if the training data is biased, the agent may learn to exhibit biased behavior. In military simulations, the training data may be biased towards certain behaviors, strategies, or capabilities, leading to unrealistic behavior in the trained units.

It is important to carefully design the RL training process, including the reward signal, exploration strategies, and training data to address these issues. Additionally, it may be necessary to supplement RL with other techniques, such as expert knowledge or rule-based systems, to ensure that the trained units exhibit realistic behavior.

This paper demonstrates the successful integration of natural language into the process of reinforced learning to enable an agent to achieve different objectives. By specifying a concrete objective, we limit the

errors in the reward function as it can be optimized for the given goal. This groundwork will improve AI agents in the future to uphold the doctrines and rules of military operations.

2 RELATED WORK

Dynamically changing the reward function in RL based on different objectives or tasks has already been used in some research papers and different methodologies have been applied.

Reward shaping or *engineering* is the process of modifying the reward function to guide the RL agent toward a specific objective (Gupta et al. 2022). This can be useful when the original reward function does not directly correspond to the desired objective, or when the agent needs to learn multiple objectives simultaneously. One way to dynamically change the reward function is to use a *composite reward function* that includes multiple terms, each corresponding to a different objective (Hu et al. 2020). For example, suppose an RL agent is learning to navigate a maze and its original reward function only provides a positive reward for reaching the goal. If we want to encourage the agent to explore the maze more thoroughly, we could add a term to the reward function that gives a small reward for every new state visited. By adjusting the weights assigned to each term, we can switch the focus of the agent between exploration and goal-reaching.

Another approach is to use *meta-learning* or *multi-task learning techniques* to learn a set of reward functions that correspond to different objectives (Wang et al. 2021). This can be useful when the objectives are distinct but related, or when the agent needs to switch between objectives frequently. The agent can then select the appropriate reward function based on the current task or objective. Multi-task learning techniques in RL refer to methods that enable an agent to simultaneously learn and accomplish multiple related tasks. The idea behind multi-task learning is to share knowledge across tasks so that the agent can learn more efficiently and generalize better to new tasks.

One approach to multi-task RL is to learn a single policy that can solve multiple tasks. This can be done by using a shared neural network architecture that processes the observations from the environment and outputs a set of actions. Each task has its own set of output nodes that correspond to the specific actions required for that task. During training, the agent learns to optimize a joint objective function that includes all the tasks. This approach is often referred to as *multi-head learning* (Kim et al. 2022).

If the reward function is difficult to define and we do have expert knowledge it is possible to use Inverse Reinforcement Learning (IRL) to learn a reward function based on expert data, therefore it would also be possible to learn a reward function based on different objectives.

Another approach is to use a hierarchical RL framework, where the agent learns a set of sub-policies that can be combined to solve different tasks (Jiang et al. 2019). In this approach, the sub-policies are trained independently to solve different tasks and then combined in a top-level policy that selects the appropriate sub-policy for the current task. The top-level policy can be trained using a meta-learning approach, where it learns to select the appropriate sub-policy based on the observed reward and state.

Multi-task RL can also be achieved by learning a shared representation of the state space (Vithayathil and Mahmoud 2020). This approach involves training a neural network to encode the state information into a low-dimensional vector representation that can be used for multiple tasks. The agent then learns a set of task-specific policies that operate on the shared state representation. Overall, multi-task RL techniques aim to improve the agent's learning efficiency and generalization by leveraging the shared structure and knowledge across multiple related tasks.

However, dynamically changing the reward function can be challenging. If the new objective is significantly different from the previous one, the agent may need to re-learn its policy from scratch, which can be time-consuming and computationally expensive. Additionally, switching objectives too frequently can make it difficult for the agent to learn a coherent policy, as it may not have enough time to explore the environment and learn from experience.

In summary, dynamically changing the reward function can be a powerful tool for guiding RL agents toward different objectives. However, it requires careful consideration of the trade-offs between exploration and exploitation, and the potential costs of switching objectives too frequently.

3 RELEGS – REINFORCEMENT LEARNING FOR COMPLEX COMBAT SITUATIONS

3.1 The "ReLeGSim" Simulation Environment

ReLeGSim is a turn-based 2D tactical game, simulating military battalion-level battles in which companies (aggregated entities) are led by an RL agent. The simulation was first introduced by (Doll et al. 2021) and supports the deployment of up to 10 companies per player. Thereby 8 combat companies and 2 reconnaissance units are usually provided. Combat units consist of different kinds of tanks. Reconnaissance units can be ground or air forces. ReLeGSim is specifically designed to train an AI using reinforcement learning. But equally, the control by a human is intended, to be able to play against a trained AI-Agent. ReLeGSim was introduced with an OpenAI Gym Interface (Brockman et al. 2016) but after its ended support we switched to the successor "gymnasium".

In the first version of ReLeGSim an attack scenario BLUE against RED is represented as a turn-based game. Here, two players (AI and or human) compete against each other with their companies to fulfill their respective objectives. One player takes the role of the attacker, who must try to take a certain target area from the enemy. The other player is the defender of the area, which he must hold for the entire duration of the game. Players have different companies with different capabilities at their disposal. A company consists of several platoons, which in turn consist of individual units and their capabilities. Players must be able to put themselves in the enemy's shoes during the game, know the abilities of the different companies and master the characteristics of the terrain to win the wargame. In particular, the attacker must use his companies and combat support synchronized in space and time to win against the defender.

To simulate the use cases, a 3D terrain base of a defined delimited area can be generated from vector, elevation, and satellite data. An example is shown in Figure 1. A specific 3D terrain, e.g., of a real military training area, is generated using a database. The generated 3D terrain is rasterized for the AI and a fixed field type (e.g., forest, road...) is assigned to each raster. This rasterized information is used for AI planning within ReLeGSim.

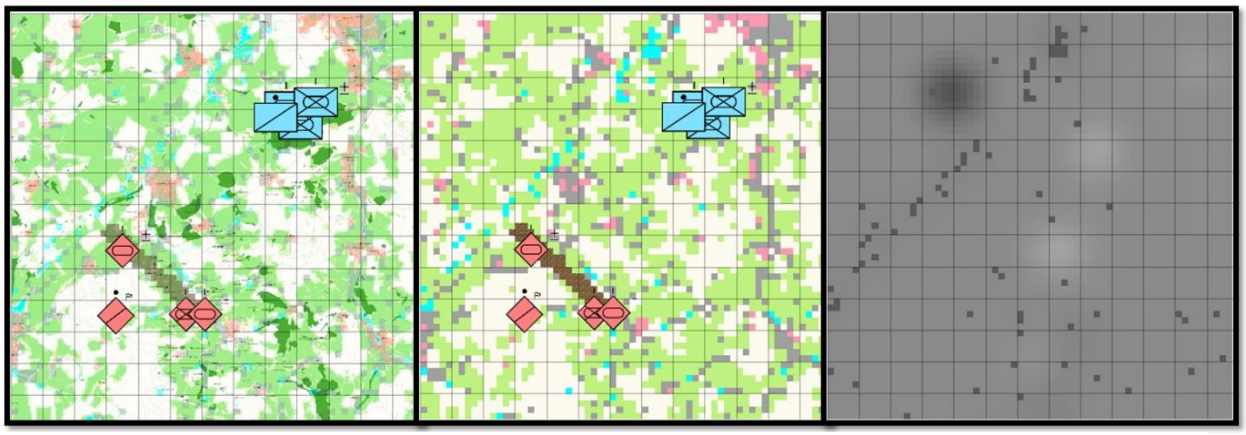


Figure 1: ReLeGSim map.

In the future, AI should not act on its own but work together with a military operator. For this concept, an Interface between the human and the AI-agent must be designed. In this paper, we propose a natural langue interface, which will be fed into the AIs observation.

3.2 ReLeGs AI Architecture Overview

The AI architecture is based on (Doll et al. 2021) and written with the TensorFlow 2.12 API and designed to support multiple use-cases. Therefore the model is split into different parts, like the visual input processing module, the scalar processing unit, the objective, the core network, and the action generation.

As needed we can exchange parts of the Modelto compare the usage of MobileNetV3 (Howard et al. 2019) and VisionTransformer (Dosovitskiy et al. 2021) as a visual input encoders, or to experiment with a different core network, as the LSTM (Hochreiter et al. 1997) is getting outdated, for example.

In addition to the optimization of the network (see Figure 2), however, the main test object in this paper is to allow the operator to intervene in the reasoning of the AI. For this purpose, special reward functions, action space, and observation space have to be defined. The structures, options for action, and principles of operation should reflect the current conditions within the military.

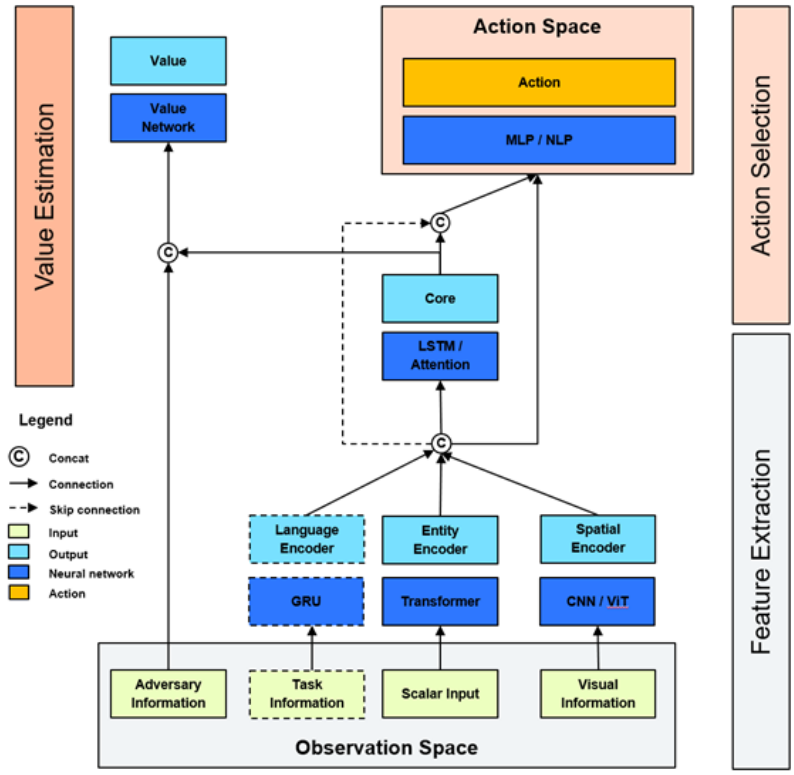


Figure 2: AI-Model architecture.

It is necessary to investigate how the decisions of the operator can be integrated into the machine learning models. There are several ways to do this. First, the AI must be informed of the operator's command. This should be done in natural language. The idea to use natural language is based on (Eloff et al. 2021). On the one hand , to teach the agent to follow commands, the reward can be controlled. On the other hand, there is also the possibility to pre-train such principles through supervised learning. Changes to the simulation can also be helpful to intervene harder in the behavior. This is especially necessary when changes are made to fixed boundaries in the simulation, like no-go areas. A combination of the various options was tested and is likely the best solution to achieve a generic interface.

3.3 Action Space

The agent’s action space is set up using natural langue and is strongly based on real-world commands. The action space was first tested by (Möbius et al. 2022) and extended in this work. In our experiment, a special vocabulary for the natural language interface is defined. We observed in our initial trials that a large space of unused vocabulary is disadvantageous and results in really slow training.
Therefore we use a small, but effective vocabulary. The vocabulary contains only the following tokens:

`"0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "attack", "move", "observe", "x", "y", "own_entity", "enemy_entity", "artillery", "attack_helicopter", "close_air_support", ";"` and `","`.

The token ";" splits the whole resulting text into multiple actions, while the "." token is used to end or pad the results. The reduction of tokens and the optimization was done manually and corresponds directly to the execution of the resulting behavior in the simulation. To tokenize the actions, we use one-hot encoding, as this allows us to use stochastic sampling over the given actions and can be easily integrated into any given RL framework through a multi-discrete representation.

Another setting we can define is the maximum length of the text. In our initial experiments, we used a maximum length of ten tokens. However, to support multiple actions in one text we need to increase it to around forty tokens.

3.4 Execution of Natural Langue Commands

The generated commands in natural langue are processed via a language parser inside the simulation. The parsed actions are passed through a generic factory to the corresponding rule-based implementation to execute the given command. The implementation of the execution of actions is crucial for the AI training process and links directly to the quality of the agents. Therefore these executors need to be carefully written with expert knowledge about the simulation and military operations.

3.5 Masking with complex NLP Action Space

This flexible action space allows many invalid sentences, which need to be learned by the AI and can slow down the initial training of the AI. To reduce the number of errors we introduced action masking. Masking the NLP action space is not as straightforward as it is with a typical multi-discrete action space, where every value maps to a single action that can be masked. With the NLP action space, every value can have multiple different meanings in the sentence. For example, a number can be used in many ways. This results in a tree-like structure. Our novel idea is based on the tree-based masking in RL of Bamford et al. (2021).

Our first step was to implement masking purely based on the semantics of the vocabulary, by creating a mask for every word based on the previous word. This would prevent e.g., two tasks (attack, move, and observe) right behind each other, or ensure that an entity must be followed by a number. Whilst this already greatly reduced possible errors, the AI could still produce a lot of sentences that, while being semantically correct, did not reflect the status of the simulation. For example, it could try to task units that were already destroyed or not part of the simulation to begin with. Thus, the masking was extended to also exclude unavailable resources.

3.6 Observation-Space

The agent's observation space consists of eight distinct components. There are three different ways to process the observations based on their input format.

```
prev_action: Box(0.0, 1.0, (10, 75), float32)
image: Box(0.0, 1.0, (64, 64, 3), float32)
core_information: Box(0.0, 1.0, (6,), float32)
own_forces_table: Box(0.0, 1.0, (10, 50), float32)
enemy_forces_table: Box(0.0, 1.0, (10, 50), float32)
combat_support_table: Box(0.0, 1.0, (4, 10), float32)
objective: Box(0.0, 1.0, (10, 75), float32)
action_mask: Box(0.0, 1.0, (76, 75), float32)
```

Where the first two entries represent the normalization range, while the third represents the observation's dimension. On the one hand, a graphical image will depict the location and terrain properties.

The image is processed via ViT (Dosovitskiy et al. 2021) and a custom CNN architecture. On the other hand, scalar values like the `own_forces_table` are used to provide accurate information on friendly forces positions. These scalar inputs are processed by a custom entity encoder transformer network. Other inputs like `enemy_forces_table`, `combat_support_table` and `core_information` are processed the same way. In addition to these inputs, the current command of the operator is given via natural language allowing it to fully comprehend the intent of the operator's instructions. All the extracted features are concatenated to an internal state representation and fed into the core network.

3.7 Reward

The given reward is based on the current objective of the agent. Therefore, the reward design will change depending on the current task. This leads to better exploration but can also lead to an increase in initial training time. To compensate this, we propose the use of curriculum learning. In the early stages of the training, a static objective is given and some helping rewards are added. Late in the training multiple objectives are sampled randomly and used for the reward generation. To support multiple objectives at the same time the individual rewards from the objectives are weighted. To boost the initial training of the natural language generation, we use a sentence-based reward. This reward guides the agent to produce meaningful sentences.

4 EXPERIMENTS & RESULTS

4.1 Experiment Setup

To enable human intervention in the prioritization of the AI-agent, the human-made command must be understood and learned by the AI. This cannot be retrofitted without changing the AI. For this reason, this capability is deeply integrated into our AI-model. Figure 2 shows our AI-Model with the added objective as task information in the observation space. Therefore, it is possible to change the current objective at any given time. It is possible to start into a scenario with an objective like “move hidden to a certain area” and if an enemy interaction occurs, the operator can change the prioritization to destroy all enemy forces.

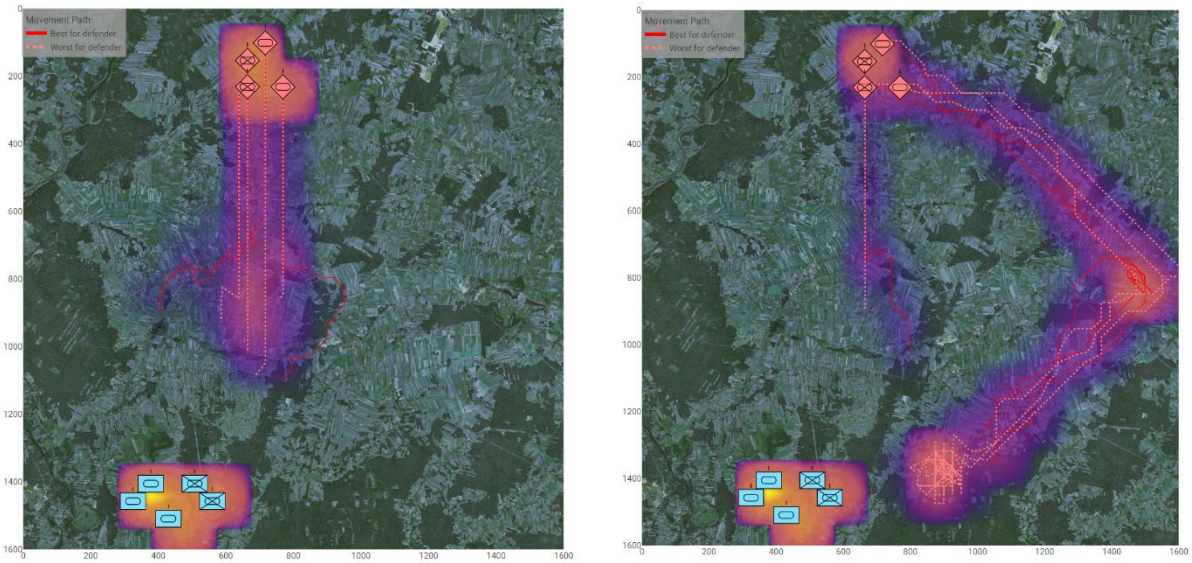
To train this complex behavior we integrated a curriculum learning strategy. This involves adding different priorities to the training and combining them randomly as input to the AI. To keep the priorities human-readable, they are expressed via natural language and given to the AI from the graphical user interface. For each priority, a reward is created that teaches the AI to adhere to the corresponding rule.

To run the experiments, we use a server with an AMD Threadripper (64 CPU-Cores) and two Nvidia RTX 4090 GPUs which allows us to train a model within 7 days (1 billion simulation steps with 24.000 training iterations). The model is implemented in PyTorch and TensorFlow. For the training we use RLlib to distribute workloads across the available hardware. As training algorithm, we use an asynchronous implementation of PPO.

4.2 Results

We can show the first implementation of an AI model with several priorities. The user can set the selected priority and the AI adjusts its behavior accordingly. Figure 3 shows an example from the ReLeGSim visualization dashboard to showcase a battalion given two different priorities. Figure 3 shows two different objectives executed with the same policy. Depending on the given objective the AI-Agent changes its behavior. On the left side (a), the AI tries to attack as fast as possible, but still tries to win the battle. On the right side (b) the AI tries to minimize own losses, which in most cases is the better strategy. These screenshots show our decision support tool (based on the idea of Horne et al. 2014) that can run a given scenario multiple times, with different objectives or combat support units. The results of multiple episodes are aggregated and automatically analyzed. This enables the operator to simulate, validate and optimize

tactics before conducting the mission. Another use-case of our decision support tool is in the evaluation of AI-models. It can be used to compare different AI-models and validate the used tactics (Möbius et al. 2022).



(a) Attack as quickly as possible. (b) As few own losses as possible.

Figure 3: Comparing company movements based on the given priority (a) and (b).

Figure 4: Curriculum learning in level 5 to 77% chance of destroying all enemies (Scenario win rate over number of simulation steps).

Another advantage of this approach is the increased level of exploration achieved by the AI agent. With varying goals and priorities, the agent is encouraged to experiment with multiple strategies to determine the most effective course of action, thereby facilitating faster and more comprehensive training. This approach encourages the agent to develop a broad repertoire of skills, enabling it to excel in diverse scenarios and adapt to unforeseen circumstances. By promoting versatile and adaptive decision-making, the agent is better equipped to optimize its performance and achieve its objectives efficiently.

Figure 4 shows a clear difference in AI training performance between training with only one priority and training with three different priorities. Adding the three different priorities initially shows an increased need for training time in the beginning, after a certain point the increased exploration shows beneficial.

These successful explorations will be adapted and extended to a drone use case to give the operator the possibility to intervene in the priorities of the AI at runtime.

Another function, which has already been successfully demonstrated with UAS (see Figure 5), is that individual UAS can be detached from the composite and controlled individually by the operator. In doing so, the AI optimizes itself to continue executing its mission in the best possible way without the extracted UAS. This feature could be also very interesting for human training and tactics evaluation. Overall, the ability to detach and control individual units represents a significant advancement in controlling AI agents and holds potential for both training and operational use.

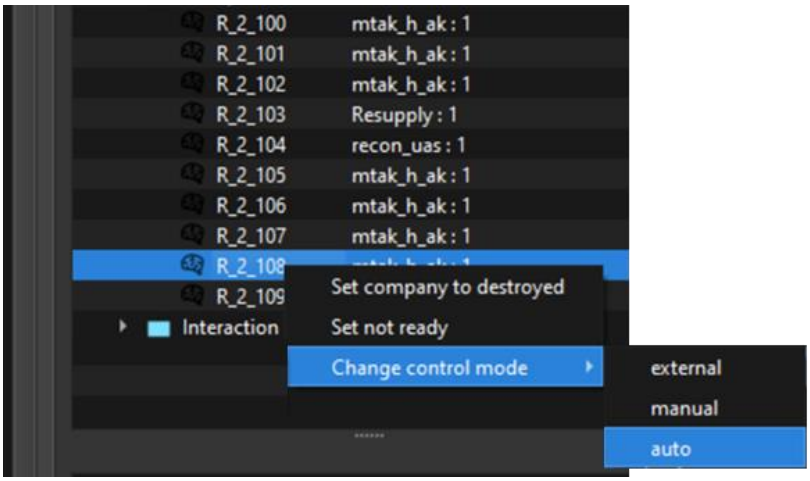


Figure 5: ReLeGSim GUI to detach UAS from composite.

5 CONCLUSION

This paper presents a novel approach to embedding objectives and doctrines into a military AI agent trained with reinforcement learning. It allows a human operator to change the behavior of the AI agent according to current needs. Simultaneously, the goal is to integrate an understandable interface for the AI forces to allow a comprehensive understanding of the AI's decisions. The AI's decision based on the given objects are easier to understand and allow a human to judge given commands. Although this approach is still in the research stage, we anticipate that it holds great potential for real-world military applications. The language-based action space is straightforward to understand and can be extended to support different needs and can be tailored to meet different requirements.

The decision support tool (shown in Figure 3) is successfully designed and implemented through an interactive application and dashboard. It has been shown that this application significantly improves the ability to understand agent strategies and the performance of various combinations of capabilities. This process has helped identify weaknesses and possible areas of improvement for the simulation model by visualizing agent behavior that was not previously observed or understood.

This approach provides a comprehensive understanding of the AI's decision-making process, allowing humans to assess and judge the given commands. As such, this research can serve as a foundation for the development of effective and efficient military AI agents that can operate alongside human counterparts in complex environments for decision support.

6 THE WAY FORWARD

Research on natural language-based military decision support is still ongoing and far from reaching its limits. While we are currently working with scripted subordinates, we want to introduce hierarchical multi-agent training to increase the capabilities of the agents. This will push the strategies to the next level. The

other open topic is the limited vocabulary. In future iterations, the vocabulary should be increased to enable more flexibility in the command and objective structure. This imposes new challenges with scripted behavior, rewards, and the planned hierarchical multi-agent training.

The given objectives are currently used for this proof-of-concept evaluation. After the current evaluation of the training, we will add more different and flexible objectives to increase the potential of this approach. Furthermore, future work will also focus on extending this approach to various military scenarios and developing a robust model that can address different objectives and goals. Additionally, the application of this method to other domains, such as autonomous UAS systems, can further enhance the utility of reinforcement learning-based AI systems.

ACKNOWLEDGMENTS

We would like to thank the entire ReLeGs and KITU study group — and especially LTC Jan Wilhelm Brendecke, MAJ Stefan Göricke and LTC Silvio Püschel from the Army Concepts and Capabilities Development Center — for their highly valuable contributions to our studies. In addition, we want to thank Kai Fischer for his valuable support in improving the AI-Model architecture.

REFERENCES

Bamford, C., and A. Ovalle. 2021. "Generalising Discrete Action Spaces with Conditional Action Trees". In *Proceedings of the 2021 Conference on Games (CoG),* August 17-20*,* Copenhagen, Denmark: Institute of Electrical and Electronics Engineers, Inc.

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

Doll, T., J.-W. Brendecke, M. Behm, and D. Kallfass. 2021. "From the Game Map to the Battlefield – Using DeepMind's Advanced AlphaStar Techniques to Support Military Decision-Makers". In *Proceedings of the NATO MSG-184 Symposium,* October 21-22, Amsterdam, Netherlands.

Dosovitskiy, A., L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. 2020. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". *arXiv*:2010.11929.

Eloff, K. M., and H. A. Engelbrecht. 2021. "Toward Collaborative Reinforcement Learning Agents that Communicate Through Text-Based Natural Language". In *2021 Southern African Universities Power Engineering Conference/Robotics and Mechatronics/Pattern Recognition Association of South Africa (SAUPEC/RobMech/PRASA), January 27-29*, Potchefstroom, South Africa: Institute of Electrical and Electronics Engineers, Inc.

Gupta, A., A. Pacchiano, Y. Zhai, S. M. Kakade, and S. Levine. 2022. "Unpacking Reward Shaping: Understanding the Benefits of Reward Engineering on Sample Complexity". *arXiv*:2210.09579.

Hochreiter, S., and J. Schmidhuber. 1997. "Long Short-Term Memory". *Neural Computation* 9(8):1735–1780.

Horne, G., and T. Meyer. 2005. "Data Farming: Discovering Surprise". In *Proceedings of the 2005 Winter Simulation Conference,* edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 1082–1087. Orlando, Florida: Institute of Electrical and Electronics Engineers, Inc.

Howard, A., M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le. 2019. "Searching for MobileNetV3". In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, October 27 – November 2, Seoul, South Korea, 1314–1324: Institute of Electrical and Electronics Engineers, Inc.

Hu, J., Y. Cheng, Z. Gan, J. Liu, J. Gao, and G. Neubig. 2019. "What Makes A Good Story? Designing Composite Rewards for Visual Storytelling". *arXiv*:1909.05316.

Jiang, Y., S. Gu, K. Murphy, and C. Finn. 2019. "Language as an Abstraction for Hierarchical Deep Reinforcement Learning". *arXiv*:1906.07343: Red Hook, Curran Associates, Inc.

Kim, Gyuhak and Ke, Zixuan and Liu, Bing 2022. "A Multi-Head Model for Continual Learning via Out-of-Distribution Replay". *arXiv*:2208.09734.

Mnih, V. and Kavukcuoglu, K. and Silver, D. and Graves, A. and Antonoglou, I. and Wierstra, D. and Riedmiller, M. "Playing Atari with Deep Reinforcement Learning". *arXiv*:1312.5602.

Möbius, M., D. Kallfass, T. Doll, and D. Kunde. 2022. "AI-based Military Decision Support Using Natural Language". In *Proceedings of the 2022 Winter Simulation Conference,* edited by B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C.G. Corlu, L.H. Lee, E.P. Chew, T. Roeder, and P. Lendermann, 2082-2093, Singapore: Institute of Electrical and Electronics Engineers, Inc.

Salas, E., L. Milham, and C. Bowers. "Training Evaluation in the Military: Misconceptions, Opportunities, and Challenges". *Military Psychology*. 15:3–16.

Silver, D., A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. 2016. "Mastering the game of Go with deep neural networks and tree search". *Nature* 529(7587):484–489.

Sutton, R., and A. Barto. 1998. *Reinforcement Learning: An Introduction*. 2nd ed. London: The MIT Press.

Varghese, N. V., and Q. H. Mahmoud. 2020. "A Survey of Multi-Task Deep Reinforcement Learning". *Electronics* 9(9):1363.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. "Attention Is All You Need". In *Proceedings of 31st Conference on Neural Information Processing Systems*, December 4 – 9, Long Beach, California.

Vinyals, O., I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver. 2019. "Grandmaster level in StarCraft II using multi-agent reinforcement learning". *Nature* 575(7782):350–354.

Vyas, S., J. Hannay, A. Bolton, and P. P. Burnap. 2023. "Automated Cyber Defence: A Review". *arXiv*:2303.04926.

Wang, Haoxiang and Zhao, Han and Li, Bo 2021. "Bridging Multi-Task Learning and Meta-Learning: Towards Efficient Training and Effective Adaptation". *arXiv*:2106.09017.

Yan, Y., A. H. Chow, C. P. Ho, Y.-H. Kuo, Q. Wu, and C. Ying. 2021. "Reinforcement Learning for Logistics and Supply Chain Management: Methodologies, State of the Art, and Future Opportunities". *SSRN Electronic Journal* 162:102712.

AUTHOR BIOGRAPHIES

MICHAEL MÖBIUS is a system engineer for AI and software applications working in the department "Operational Analysis and Studies" at Airbus Defence and Space in Germany as technical lead for simulation and AI-related studies and projects. He also loves to share and teach his software engineering knowledge to young students. He started developing software at 10 years old and sold his first commercial software at 16. As of today, his main research interests include large language model (LLM), stochastic simulation design and analysis, autonomous systems, reinforcement learning, and operational analysis. His email address is michael.moebius@airbus.com.

DANIEL KALLFASS is an expert in operational analysis in the department "Operational Analysis and Studies" at Airbus Defence and Space in Germany. He has more than 17 years of experience in leading national and international defense and security-related studies and research projects in the field of simulation-based operational analysis, distributed simulations, and artificial intelligence. His focus is on the development and application of simulations such as the 3D agent-based simulation PAXSEM in combination with data farming and the latest artificial and deep learning techniques, such as deep reinforcement learning simulations. His email address is daniel.kallfass@airbus.com.

MATTHIAS FLOCK is a systems engineer for simulation software and AI in the department "Operational Analysis and Studies" at Airbus Defence and Space in Germany, working on a broad spectrum of simulation tools and AI-related studies. He started his career at Airbus in 2009 and has since worked in a multitude of different areas, from C2 applications, mobile apps, and satellite ground stations to AI and simulation. His current focus is on the development of simulation tools to enable reinforcement learning. His email address is matthias.flock@airbus.com.

LTC THOMAS DOLL is a German Army officer currently working for the German Joint Support Service Command, where he is responsible for conducting military analyses and studies. His main areas of interest are modeling and simulation, the development and deployment of unmanned systems, and artificial intelligence. From 1990 to 1994, he studied electrical engineering at the University of the German Armed Forces in Munich. He received his Master of Science degree in 2004 from the Naval Postgraduate School in Monterey, California, USA. His email address is thomasmanfreddoll@bundeswehr.org.

LTC DR. DIETMAR KUNDE is a German Army officer who completed the Modeling, Virtual Environments, and Simulation (MOVES) Ph.D. program at the MOVES Institute of the Naval Postgraduate School (NPS) in Monterey, California in December 2005. He received his M.S. in Surveying from the German Armed Forces University in Munich in 1985. In 1997, he received his M.S. in Operations Research at the NPS. His research covers military decision support, human performance modeling, cognitive modeling, and artificial intelligence. For the past 10 years, he has been working as a branch head for operations research and modeling and simulation for the German Army Headquarters. His email address is dietmarkunde@bundeswehr.org.