Improving Reinforcement Learning Experiments in Unity through Waypoint Utilization

Caleb Koresh[1], Volkan Ustun[2], Rajay Kumar[2], Tim Aris[3]
[1] University of Florida, Gainesville, FL
[2]University of Southern California, Institute for Creative Technologies, Playa Vista, CA
[3]US Army Soldier Center, Orlando, Florida
calebkoresh@ufl.edu, ustun,kumar@ict.usc.edu, timjaris@gmail.com

Abstract

Multi-agent Reinforcement Learning (MARL) models teams of agents that learn by dynamically interacting with an environment and each other, presenting opportunities to train adaptive models for team-based scenarios. However, MARL algorithms pose substantial challenges due to their immense computational requirements. This paper introduces an automatically generated waypoint-based movement system to abstract and simplify complex environments in Unity while allowing agents to learn strategic cooperation. To demonstrate the effectiveness of our approach, we utilized a simple scenario with heterogeneous roles in each team. We trained this scenario on variations of realistic terrains and compared learning between fine-grained (almost) continuous and waypoint-based movement systems. Our results indicate efficiency in learning and improved performance with waypoint-based navigation. Furthermore, our results show that waypoint-based movement systems can effectively learn differentiated behavior policies for heterogeneous roles in these experiments. These early exploratory results point out the potential of waypoint-based navigation for reducing the computational costs of developing and training MARL models in complex environments. The complete project with all scenarios and results is available on GitHub: https://github.com/HATS-ICT/ml-agents-dodgeball-env-ICT.

Introduction

The development of autonomous synthetic characters presents opportunities to provide safe, replicable, and cost-efficient virtual training. These simulations require intelligent virtual agents that can achieve human-level performance and strategy to be effective. Reinforcement Learning (RL) strives to find optimal policies in game-like environments and can produce adaptive behavior in virtual environments. Multi-agent Reinforcement Learning (MARL), on the other hand, models multiple agents that learn by dynamically interacting with an environment and each other, providing a framework for evaluating competitive and collaborative dynamics between these agents (Buşoniu, Babuška, and De Schutter 2010). MARL algorithms have shown the ability to learn policies demonstrating cooperative and strategic team-oriented behaviors (Gronauer and Diepold 2022). State-of-the-art simulations utilize MARL algorithms to generate behavior for dynamic and adaptive agents.

Even though MARL can assist in behavior generation for military training simulations (Ustun et al. 2021), the task of enabling machine learning to make effective decisions for synthetic characters, including commander roles, is particularly challenging given that these simulations unfold in complex, multi-objective, continuous, stochastic, partially observable, non-stationary, and doctrine-based environments involving multiple collaborating or competing players. Furthermore, military training environments have heterogeneous entities with different roles. For example, some team members may have weaponry that warrants other behavior. As a result, leveraging MARL algorithms, even when only considering homogeneous entities, requires immense computational power to achieve convergence.

Waypoint-based navigation has shown the potential to address the computational requirements of MARL experiments by replacing the fine-grained action space with a more abstracted, navmesh-based waypoint movement system in single-agent reinforcement learning (Aris, Ustun, and Kumar 2023). Such a movement system discretizes the movement of agents and can increase the generality and success rate of the models. This paper augments the waypoint-based navigation to heterogeneous multi-agent systems and shows that leveraging waypoints can decrease the computational requirements of MARL training with superior performance compared to fine-grained action spaces. Furthermore, the waypoint-based models retain performance even when translated back onto the simulation environment with fine-grained action spaces.

For the scenarios in this paper, we utilized Unity's ML-Agents framework (Juliani et al. 2018) and a modified version of Unity's dodgeball environment (Berges et al. 2021), which acts as a simple proxy for military simulations. The changes to the dodgeball environment allowed us to test our main argument: a waypoint-based navigation system can assist in speeding up MARL experiments with heterogeneous entities without compromising performance in military simulation-like environments. Waypoints allow for generating abstractions of an environment accurately rep-

resentative of the terrain, which helps narrow the search space, thus allowing for faster learning. We also introduce roles within a team in the modified scenario: each team has short-range and long-range units; the long-range units could represent snipers or simply units with longer-range weaponry. The multi-agent Posthumous Credit Assignment (MA-POCA) algorithm (Cohen et al. 2021) runs our MARL experiments with functionality for self-play, enabling the agents to learn against themselves as equally skilled opponents without any human intervention. Our experiments demonstrate the effectiveness of waypoint-based models by running head-to-head matches against models trained with fine-grained movement systems. In these matches, waypoint-based models performed significantly better while learning visibly distinct policies for each role in a team.

After providing a short background, we introduce our waypoint-based movement system. We then give the details of our proof-of-concept scenarios and discuss our experimentation results and findings. We conclude with the potential implications of waypoint-based movement systems for military training simulations.

Background

Many state-of-the-art MARL algorithms utilize an actor-critic approach (Konda and Tsitsiklis 1999). In this approach, during training time, a central critic can observe all the actors (agents) and their rewards. As a result, it can inform individual agent policies, potentially yielding a learned consensus in cooperative tasks (Lowe et al. 2017). For example, modifications to the popular on-policy single agent Proximal Policy Optimization (PPO) (Schulman et al. 2017) algorithm for multi-agent settings under actor-critic paradigm can be surprisingly effective (Yu et al. 2022). Multi-agent Deep Deterministic Policy Gradient (MADDPG)(Lowe et al. 2017), which is a multi-agent algorithm by design, delivers excellent results for toy problems like predator-prey, cooperative navigation, and physical deception. Counterfactual multi-agent policy gradient (COMA) (Foerster et al. 2018) is another actor-critic architecture that tackles the challenge of multi-agent credit assignment in cooperative settings with a unique shared reward through counterfactuals.

Multi-agent Posthumous Credit Assignment (MA-POCA)(Cohen et al. 2021) extends COMA via attention to better handle the credit assignment with terminated agents in training episodes. Before MA-POCA, the most common solution for eliminated agents was to account for the maximum number of agents and place the inactive agents in an absorbing state, allowing for reward to propagate back to eliminated agents. MA-POCA approaches the posthumous credit assignment problem via attention rather than a fully connected neural network, avoiding the need for any absorbing state while accurately quantifying an individual's contribution to the team's outcome. MA-POCA performed better than PPO and COMA in various cooperative environments, and the margin was most significant in scenarios that added or removed agents during a simulation. These results are auspicious for MARL experiments with military training scenarios, where agent elimination is prevalent and self-sacrificing behavior can benefit the team's outcome.

Waypoints

We developed an automatic waypoint generation system to set up the waypoint-based movement graph for the terrain used in a scenario. This system utilizes a connected mesh of waypoints deployed on geo-specific and basic Unity terrains, even though the examples used in this paper are all simple Unity terrains. Before deploying the waypoints, we create a NavMesh for the terrain with the constraints such that the agent cannot ascend steep slopes. The action space used in our experiments includes choosing which of 8 adjacent waypoints (cardinal directions and diagonals) to move to. We laid out a grid one waypoint at a time, moving from the southwest corner to the northeast corner and assigning connections to any waypoints in the southeast, south, southwest, and east directions. When a waypoint is visited, the environment attempts to create adjacent waypoints in any direction that does not have one at a parameterized distance. Even though we generate waypoints for the entire grid, each generated waypoint and edge is marked either valid or invalid. A waypoint is marked invalid if there is no proper navmesh position at that location. An edge is marked invalid if there is no good path on the navmesh to get to that waypoint, the waypoints are too far apart vertically, or the path is far longer than the Euclidean distance between the waypoints. Our RL experiments will utilize the generated waypoint-based movement graph, and if an edge or waypoint is marked invalid, the action masking will prune the associated action. Our experiments treat all edges as equidistant, although the diagonal edges are slightly longer. Figure 1 depicts the waypoint placement.

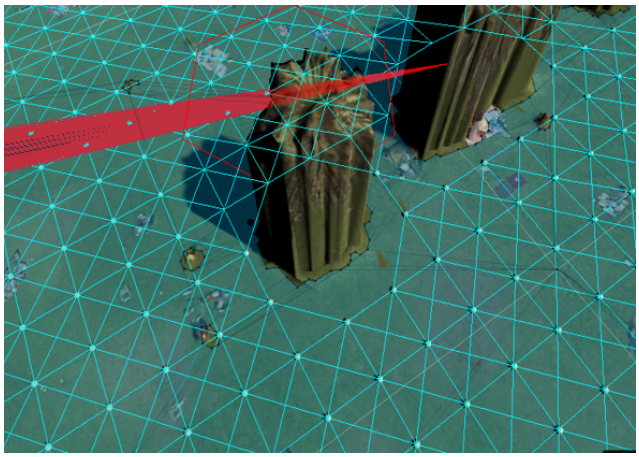

Figure 1: Waypoint placement

We compared the performance of the waypoint and non-waypoint agents in this paper, and the main difference between them is their action spaces. Non-waypoint agents use fine-grained actions along with a branch to adjust their rotation as defined in the original dodgeball scenario. However, we removed the dash mechanic, and dashes are not

part of the action spaces for either agent type. For movements, the non-waypoint-based agents use the original implementation, which features two fine-grained branches representing movement along the X and Z directions. On the other hand, our waypoint-based agents use one discrete action branch consisting of 9 possible actions. Eight of these actions correspond to the eight cardinal directions, and the ninth is for standing still. As stated earlier, movement actions are masked between waypoints when one or more directions are inaccessible, reducing the search space without affecting gameplay.

Environment

Unity's original dodgeball environment features a flat terrain (Berges et al. 2021) where agents cooperatively battle against the opposing team by picking up and throwing dodgeballs to eliminate opponents. To make this environment akin to a military simulation, we iteratively adapted it to fit our needs. First, we added hills and obstacles for similarity to geo-specific terrains. Such a change required us to build functionality for shooting in three dimensions. To address this, we added additional ray casts to expand from two to three dimensions of observations. We also incorporated infinite ammunition so agents would not be required to locate and pick up dodgeballs, only focusing on the combat aspects of the dodgeball scenario. We generated two sizes of environments to introduce varying levels of spatial complexity. The smaller of the two is an approximately 45x50 meter environment with a hill in the middle. The larger of the two is around 45x100 meters, with the same hill in the center of the arena and two additional smaller but taller hills on either end. Furthermore, we introduced two different obstacle densities for each environment size: sparse and dense. Figure 2 shows these four environment combinations with and without waypoint placements.

Simple Combat Scenarios with Heterogeneous Units

Military scenarios often include role-based strategy; snipers are more advantageous from long range, infantry prefer close range, and other units could have different advantages and disadvantages. Standard MARL experiments dictate homogeneous units for efficient training, but training heterogeneous units is essential for practical virtual military training scenarios to yield different types of behavior based on each agent's available weaponry. In this effort, we implemented long and short-range units with identical policy architectures to train long and short-range units simultaneously. Such a change increases training time and complexity since it enables a more complex strategy, and each policy receives half as much training as it would have if all agents used the same policy. Nevertheless, we found it worthwhile to explore these possibilities, so we tuned the parameters to fit each type of unit's intended role. In our setup, long-range units have increased ray cast range, targeting range, and projectile velocity, but they could shoot less frequently than their short-range comrades in the case of the large arena.

Our modified dodgeball scenarios consist of two teams of four units, each comprised of two short-range and two long-range units. We used two environments of different sizes and terrains and tested each with two different obstacle densities, as seen in Figure 2. The agents' observation spaces include: (1) The cooldown time before they can use another projectile; (2) The proportion of remaining hit points relative to their initial two hit points; (3) The agent's current direction and velocity; and (4) Raycasts which detect obstacles or other agents spanning 200 degrees horizontally and 60 degrees vertically. Furthermore, each agent can be hit twice before being eliminated.

This limited information and lack of deterministic behavior ensure all behavior is learned and based only on local observations. To encourage positional strategies and reduce the complexity of the simulation, we implemented automatic targeting so that the models need not learn the complex task of accurately shooting their opponents. Our implementation only targets units less than 45 degrees horizontally from the direction the agent is facing. When multiple enemies are in this range, the target is the one that is closest to the agent's facing direction. We found this to significantly reduce training time compared to allowing agents to pick the angle of their shots. Parameters that differed between roles and environment sizes included ray-cast length, maximum targeting distance, fire rate, and projectile velocity.

Experiment Design and Results

We set up simple four vs. four training symmetrical scenarios with our waypoint-based movement system and the standard Unity dodgeball fine-grained movement system. We ran each scenario on four different environment configurations, as shown in Figure 2. We allowed all experiments to run for 20 million training steps to compare their learning directly. The long-range units' parameters were representative of a sniper-like role. Both long-range and short-range units were trained using the same reward system. The reward for landing a hit on an enemy unit was 0.1, and a linearly decreasing reward was given to the winning team, reflecting the proportion of steps remaining to encourage faster victories. The losing team was penalized a -1 reward regardless of time.

Furthermore, we tracked the agents' ELO scores as a learning metric popularized in (Silver et al. 2017). ELO score represents the relative skill of the agents compared to previous versions of themselves. A 100-point difference in ELO would signify a 64% win rate, and a 200-point difference in ELO represents a 76% win rate when measured against the less skilled version of the agent. Self-play makes ELO a good indicator of learning in MARL environments. As can be seen in Figures 3 and 4, ELO scores for the waypoint-based movement systems increase more rapidly and converge at a significantly higher value for both long-range and short-range units.

We also tested the waypoint-based agents against their fine-grained moving counterparts for a more direct comparison in head-to-head matches. To have both types of agents move in the same environment, we lifted the waypoint constraints from the models trained with a waypoint-

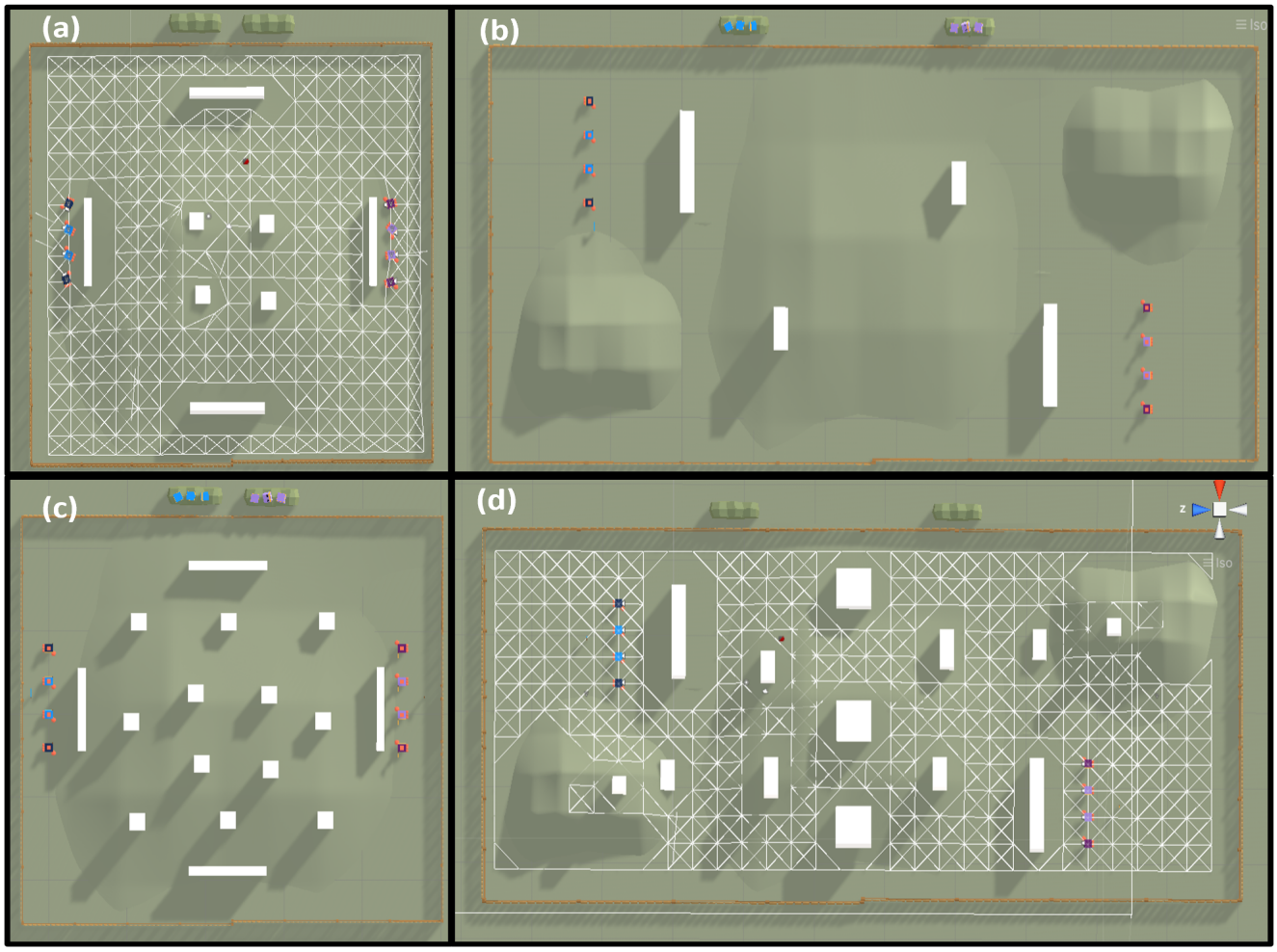

Figure 2: Environments used in our experiments:(a) small environment with sparse obstacles; (b) large environment with sparse obstacles; (c) small environment with dense obstacles; (d) large environment with dense obstacles. (a) and (d) show the waypoint placements for small and large environments.

based movement system, leaving them with nine discrete movement actions and the ability to change directions at any point. Since we removed the assumptions of turn-based waypoint movements, the head-to-head evaluation setting was slightly different than the training setting for the waypoint-based movement models. Such a difference is a disadvantage for waypoint-based models. Still, as shown in Figure 5, waypoint-based systems performed better than their fine-grained movement counterparts over 100 evaluations in each of the four scenarios. So, no matter the evaluation condition, the waypoint-based movement systems learned robust and high-performing behavior models for our test scenarios.

Discussion

Complex military environments present significant challenges for MARL methods since they can be stochastic, partially observable, non-stationary, doctrine-based, and role-based. MARL experiments require heavy computation, demanding substantial computational resources and making extensive MARL experiments elusive for many researchers. This paper shows how waypoint-based movement systems could address some of these challenges. A waypoint-based movement system could help speed up learning in MARL experiments, and learned policies are robust enough even when they are transferred back to the original action space. Our automatic waypoint generation works with geo-specific terrains, making MARL experiments on realistic terrains more accessible. Furthermore, it creates a foundation for further assumptions to be made based on a project's unique design and needs. For example, the waypoint-based movement graphs could provide a foundation for leveraging simpler graph-based environments, e.g., in Python, giving new opportunities for even faster training of behavior policies and transferring them back to higher fidelity environments for fine-tuning.

We found that, on average, waypoint movement systems progress in ELO rating with fewer steps than their fine-grained movement counterparts. ELO rating is a powerful

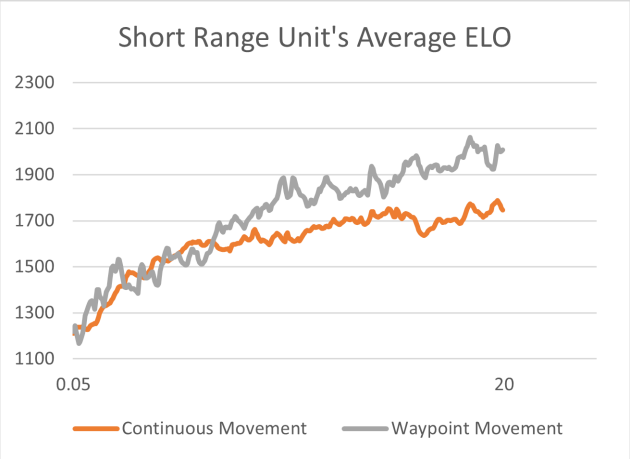

Figure 3: Short-range units ELO score progression during training in large environment for waypoint-based and fine-grained (continuous) settings

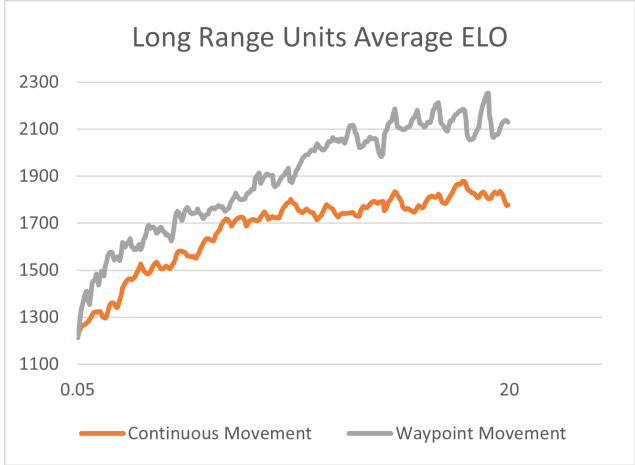

Figure 4: Long-range units ELO score progression during training in large arena for waypoint-based and fine-grained (continuous) settings

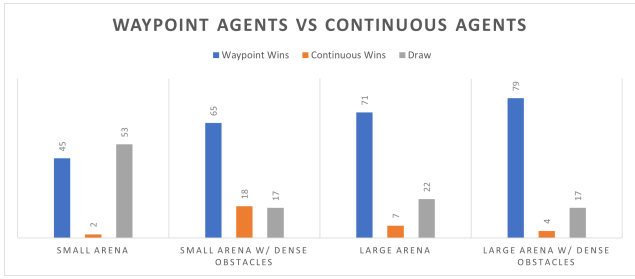

Figure 5: Waypoint-based vs fine-grained (continuous) wins in head-to-head matches

metric for learning, but considering the differences between the two movement systems, it may be slightly biased. Consequently, we also tested the waypoint system in a Unity environment to see whether or not a simplified movement system would learn sufficiently good policies to compete with agents using standard fine-grained movement. The ability to transfer policies back into the original action space could be critical for specific applications, and our results showed that despite competing under conditions different from those during training, the waypoint-based agents could easily outcompete the agents trained using fine-grained movement.

Training units with different capabilities is a critical part of military simulations. Waypoint-based movement systems also assist in speeding up the convergence of MARL experiments with heterogeneous units. Our experiments show that we can learn effective policies simultaneously for units with varying capabilities, e.g., long- and short-range units, in our proof-of-concept scenarios. Unity's ML-Agents framework makes the implementation of training two separate policies simultaneously simple and effective. We found that parameter changes were sufficient to yield different behaviors, and no added functionality was necessary to differentiate between the roles. The behavioral differences between the two roles were consistent between the waypoint and non-waypoint environments. In our experiments, long-range units tended to hold back and engage in long-range combat, whereas the short-range units utilized a more aggressive short-range style. We believe adding different functionality between the two roles could further exacerbate the differences in learned behavior.

Even though our results are promising, it is worth noting that this paper did not delve into further abstractions for even faster experiments, which will largely depend on the nuances of each project. Our focus was primarily on speeding up experiments in realistic environments, potentially with geo-specific terrains. However, our results lead us to believe that such abstractions could significantly reduce computational requirements to run similar experiments, and waypoints could provide a strong foundation for creating high-speed abstract simulation environments. Further research could investigate such abstractions in military simulations and seek to understand what degrees of abstraction retain the ability to learn policies that function in their non-abstract counterpart. For example, equation-based calculations could be leveraged rather than firing projectiles, or graph-based representations could help integrate recent architectures, such as graph transformers, into machine learning for multi-agent systems.

In future work, we look forward to investigating adding additional functionality and units, such as vehicles or medical units, to create a broader range of military scenarios. More importantly, we plan to continue our ongoing efforts on the effectiveness of abstracted environments for performance and their connections to waypoint-based representations. Such abstractions offer an avenue for faster reinforcement learning, which is crucial for complex systems such as military training simulations. Reasonable abstractions with pathways for transfer learning should help narrow the search space without removing critical strategic components like cooperative positioning while still allowing the generation of intelligent behavior in high-fidelity environments.

Conclusion

MARL algorithms offer the opportunity to develop intelligent adaptive teams of synthetic characters, but the challenges associated with military training scenarios have limited their development. This paper introduced some simple combat environments to serve as a proxy for high-fidelity training scenarios. We found that a waypoint-based system can narrow the search space without sacrificing the performance of the original fine-grained movement environments. This system is scalable, geo-specific, and accessible. These preliminary experiments suggest that graph-based abstractions, represented by the nodes and edges of the automatically generated waypoints, could make complex environments run fast. An abstract graph-based version of a military training environment would be less computationally expensive while retaining the crucial positional strategy required for adaptable intelligent agents. Further research in this direction should make training complex scenarios with reinforcement learning more accessible.

Acknowledgments

Research was sponsored by the Army Research Office and was accomplished under Cooperative Agreement Number W911NF-20-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation.

References

Aris, T.; Ustun, V.; and Kumar, R. 2023. Learning to take cover with navigation-based waypoints via reinforcement learning. In *The International FLAIRS Conference Proceedings*, volume 36.

Berges, V.-P.; Teng, E.; Cohen, A.; and Henry, H. 2021. Ml-agents plays dodgeball. *https://blog.unity.com/engine-platform/ml-agents-plays-dodgeball*.

Buşoniu, L.; Babuška, R.; and De Schutter, B. 2010. Multi-agent reinforcement learning: An overview. *Innovations in multi-agent systems and applications-1* 183–221.

Cohen, A.; Teng, E.; Berges, V.-P.; Dong, R.-P.; Henry, H.; Mattar, M.; Zook, A.; and Ganguly, S. 2021. On the use and misuse of absorbing states in multi-agent reinforcement learning. *arXiv preprint arXiv:2111.05992*.

Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Gronauer, S., and Diepold, K. 2022. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review* 1–49.

Juliani, A.; Berges, V.-P.; Teng, E.; Cohen, A.; Harper, J.; Elion, C.; Goy, C.; Gao, Y.; Henry, H.; Mattar, M.; et al. 2018. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*.

Konda, V., and Tsitsiklis, J. 1999. Actor-critic algorithms. *Advances in neural information processing systems* 12.

Lowe, R.; Wu, Y. I.; Tamar, A.; Harb, J.; Pieter Abbeel, O.; and Mordatch, I. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* 30.

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

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. 2017. Mastering the game of go without human knowledge. *nature* 550(7676):354–359.

Ustun, V.; Kumar, R.; Reilly, A.; Sajjadi, S.; and Miller, A. 2021. Adaptive synthetic characters for military training. *arXiv preprint arXiv:2101.02185*.

Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; and Wu, Y. 2022. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems* 35:24611–24624.