Creating Interactive Crowds with Reinforcement Learning

Ariel Kwiatkowski

LIX, École Polytechnique/CNRS, Institut Polytechnique de Paris, France
Route de Saclay, 91128 Palaiseau, France
ariel.kwiatkowski@polytechnique.edu

Abstract

The entertainment industry, as well as the field of Computer Graphics, frequently faces the issue of creating large virtual crowds that would populate a scene. One of the ways to achieve that, particularly with modern rendering techniques, is by using simulation – this, however, is nontrivial to design and control. The main goal of my PhD work is working towards the creation of a tool enabling the creation of virtual crowds that one can interact with, and we believe the best way to that is through Multiagent Reinforcement Learning techniques. These animated crowds can then be used both in movies and video games. Especially for the latter, it is highly desirable that both the crowd as a whole, as well as the individual characters, can react to the user's input in real time.

Introduction

The dominating approach for crowd simulation in Computer Graphics (CG) is using classical hand-crafted algorithms, and adapting them to the specific scenario (Toll and Pettré 2021). These algorithms typically use some rule that considers their goal (e.g. towards the destination), the positions and velocities of obstacles and other nearby agents, and based on that output a collision-free velocity. This can then be combined with a global path-planning algorithm to generate the complete trajectories of each agent.

There are several issues with this approach. Firstly, agents obtained this way exhibit very simple and homogeneous behaviors, which makes them seem unrealistic. Secondly, it is in general difficult to encode additional preferences for their behavior, like changing the size of their personal space, arranging them in groups, or modeling persons with disabilities. Finally, the agents have no actual notion of other participants, especially a potential human, and will only treat them as another potential obstacle.

For this reason, we propose using Reinforcement Learning (RL) as the mechanism driving the behavior of agents in the crowd. Over the recent years, RL proved itself in many challenging domains such as the Atari games (Mnih et al. 2015), the ancient game of Go (Silver et al. 2017), or StarCraft II (Vinyals et al. 2017). These high-profile achievements proved the method's viability in solving difficult problems, and drew significant attention to the field.

In particular, we intend to focus on the multiagent perspective, taking advantage of the fact that a crowd consists of multiple similar (or even identical) agents, each of them having analogous goals (although not necessarily shared). While MARL algorithms are relatively newer compared to their single agent equivalent, they have also achieved success in beating the previous best solutions in the game of Hanabi (Hu and Foerster 2019). In this case, a major difficulty lies in the coordination between agents, each of which only has a partial view of the game.

Existing Work

While a certain body of research on the topic of applying RL to Crowd Simulation already exists, the field is far from being solved. The main paradigm of MARL, which can be described as independent learning with single-agent algorithms, seems to be struggle with the basic task of navigation combined with collision avoidance in the presence of multiple agents. Here we present a subset of the relevant papers.

(Lee, Won, and Lee 2018) discuss this exact task – given a set of agents, make them move to their respective destinations, while avoiding colliding with each other and the environment. They use a relatively simple approach with the independent learning method, basing the agents' observations on raycasting. The resulting agents, while navigating to their goals correctly, still collide quite frequently, making the resulting animations implausible.

(Long et al. 2018) approach a similar problem, but from a robotics point of view. In that case, any collisions between robots can have disastrous consequences, so it was accordingly resultant in the "death" of an agent in their training procedure. They also used a two-stage method, firstly pretraining the agents on an unstructured environment, and then fine-tuning them to a specific scenario. While the resultant behaviors do avoid collisions, this is still not entirely accurate, as humans do sometimes collide without major consequences. They are also very robotic, which is not what one expects from a human crowd.

(Xu and Karamouzas 2021) take a step towards using real data in the policy learning. By using knowledge distillation on recorded trajectories, they generate an additional reward signal which is included as an auxiliary task. Some of their findings are surprising, particularly the fact that including that auxiliary task improves the performance on the primary tasks as well. Ultimately, this approach still suffers the problem of homogeneous behaviors and a lack of interactivity.

Main Challenges

Environment Dynamics

One of the technical details with a nontrivial impact turns out to be the environment dynamics, both for the agents' actions and observations. Actions can be either discrete (simpler) or continuous (more expressive). They can directly control either the velocities or the accelerations. Finally, they can operate either in Cartesian or polar coordinates. There is no single correct choice, and different options can impact both the performance, as well as the qualitative properties of movement, which still affect the realism of the simulation.

Optimization Algorithm

Our current approach is using the Unity3D engine with ML-Agents to build the environment, and a custom implementation of the PPO algorithm (Schulman et al. 2017) for policy optimization. We consider many potential improvements to the algorithm, most notably, introducing multiagent hierarchies could improve the coordination between agents to decrease the collision chance. Furthermore, treating the problem as multitask learning could allow better separation between the mutually conflicting goals.

Reward Design

We expect that the design of the reward function might prove difficult. Even in the simplest scenario only considering navigation and collision avoidance, there can be several parameters controlling the sparsity of the navigation signal, the balance between the two separate reward sources, etc. It gets even more complex when introducing components meant to encourage human-like behavior – maintaining a comfortable speed, avoiding others' personal space, and similar features.

Performance and Realism

While RL is effective at optimizing a given reward function, humans are not, so if we want the trained agents to behave similarly to humans, it might be worth moving away from the classic optimization paradigm. This may be circumvented by designing a reward function whose intention is "act like a human", for example by basing it on real-world data, or on perceptual studies of what elements of a crowd simulation matter.

Methods and Progress

At the time of writing, I am in the middle of a research collaboration with Ubisoft La Forge, where I work on a related project involving multiple agents navigating through a shared environment, trying to satisfy in some sense the subjective preferences of a user. This shares both the multitask and multiagent aspects of the regular crowd simulation setup. The first objective is introducing a new method for multitask learning through disentangling the different tasks with separate policies, which can be combined afterwards. Later, I intend to introduce a form of soft centralization to introduce coordination between agents, without the inherent problems of a centralized policy. To this end, I am developing a custom implementation of PPO that is then trained on simulations built in Unity ML-Agents.

By the time of the workshop date, the collaboration with Ubisoft will be over, and ideally will result in a submitted publication, or a situation where most development is done, and some data is collected, so that a publication could be made soon. I also aim to make sufficient progress on the main Crowd Simulation project, so that it is possible to see some concrete results with one of the aforementioned improvements to the baseline model.

References

Hu, H.; and Foerster, J. N. 2019. Simplified Action Decoder for Deep Multi-Agent Reinforcement Learning. *arXiv:1912.02288 [cs].*

Lee, J.; Won, J.; and Lee, J. 2018. Crowd simulation by deep reinforcement learning. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, 1–7. Limassol Cyprus: ACM. ISBN 978-1-4503-6015-9.

Long, P.; Fan, T.; Liao, X.; Liu, W.; Zhang, H.; and Pan, J. 2018. Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, 6252–6259. IEEE.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *arXiv:1707.06347 [cs].*

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; Lillicrap, T.; Simonyan, K.; and Hassabis, D. 2017. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *arXiv:1712.01815 [cs].*

Toll, W.; and Pettré, J. 2021. Algorithms for Microscopic Crowd Simulation: Advancements in the 2010s. *Computer Graphics Forum*, 40(2): 731–754.

Vinyals, O.; Ewalds, T.; Bartunov, S.; Georgiev, P.; Vezhnevets, A. S.; Yeo, M.; Makhzani, A.; Küttler, H.; Agapiou, J.; Schrittwieser, J.; Quan, J.; Gaffney, S.; Petersen, S.; Simonyan, K.; Schaul, T.; van Hasselt, H.; Silver, D.; Lillicrap, T.; Calderone, K.; Keet, P.; Brunasso, A.; Lawrence, D.; Ekermo, A.; Repp, J.; and Tsing, R. 2017. StarCraft II: A New Challenge for Reinforcement Learning. *arXiv:1708.04782 [cs].*

Xu, P.; and Karamouzas, I. 2021. Human-Inspired Multi-Agent Navigation using Knowledge Distillation. *arXiv:2103.10000 [cs].* ArXiv: 2103.10000.