Acta Polytechnica CTU Proceedings **57**:323–329, 2026
https://doi.org/10.14311/APP.2026.57.0323

Published by the Czech Technical University in Prague

RL-GODOT PEDESTRIAN SIMULATION: CURRICULUM-BASED REINFORCEMENT LEARNING FOR PEDESTRIAN SIMULATION

Giuseppe Vizzari*, Andrea Falbo, Ruben Tenderini, Daniela Briola

University of Milano-Bicocca, Department of Informatics, Systems, and Communication, viale Sarca 336 – Building 14, 20126 Milano, Italy

* corresponding author: giuseppe.vizzari@unimib.it

Abstract. The paper describes a research effort aimed at developing RL-Godot, a reinforcement learning (RL) based open source software system supporting the study of pedestrian dynamics. We first introduce the curriculum based RL approach to pedestrian simulation adopted for RL-Godot, then describe its system architecture and report a preliminary experimental application that evaluates the framework's ability to qualitatively reproduce low-to-medium density pedestrian behaviors reported in the literature.

Keywords: Reinforcement learning, curriculum learning, pedestrian simulation.

1. Introduction

Reinforcement Learning (RL) [1] is a machine learning approach in which an agent learns how to behave in in its environment by receiving a scalar reward signal that reflects the consequence of each action. The RL algorithms explores the effects of agents' actions, and gradually learn how to select an action given the current situation, through a trial and error process. This feedback received by the agent from the environment, as a consequence of its action, is a simple scalar value: it is much less informative than the ground-truth based loss function value, typical of a supervised training process Nonetheless, it is applicable in cases where there are difficulties in achieving sufficiently large and varied data sets for supervised training, but an evaluation of the quality of an action can be computed.

Pedestrian simulation, despite being still a very active field of research, has provided sufficient empirical evidences allowing for such an evaluation. Recent works have shown that Deep Reinforcement Learning (DRL) (i.e. RL techniques employing deep neural networks in their functioning) can produce policies producing plausible pedestrian behaviors. This has been attempted previously in [2, 3] with promising results but room for improvement. Within this line of research, in [4] we developed a curriculum based RL approach and a simulation system built on Unity [5] and the ML-Agents [6] toolkit as underlying enabling technologies.

The present work aims at substituting proprietary closed source technologies with open source libraries and tools, improving accessibility and reproducibility of the achieved results. The work follows however an analogous approach, that we will now briefly introduce, focusing on the fundamental elements of the curriculum based RL training process.

2. Reinforcement learning and pedestrian simulation

First of all, RL algorithms exploit experiences in the form of tuples $(s_t,\ a_t,\ r_{t+1},\ s_{t+1})$, where s_t is the agent's perception (state) at time t, a_t is the action taken, r_{t+1} is the reward signal received from the environment after executing a_t, and s_{t+1} is the resulting next state. Given a sufficiently rich set of such experience tuples, and under standard assumptions (e.g., the Markov property and sufficient exploration), RL algorithms learn a *policy* $\pi(s)$ – a mapping from states to actions – that maximises the expected cumulative (discounted) reward. In our case, the state of the environment is not completely accessible to the agent, so it is actually substituted by its perception by the agent, as a form of belief about the environmental state. Moreover, the Markov property (stating that the future state of the environment depends only on the current state and action taken by the agent, not on any earlier history) is a sufficient condition for RL to succeed but not a necessary one: when it does not hold, the environment is said to be partially observable, and the problem falls in the category of Partially Observable Markovian Decision Processes (POMPD). In these cases, RL methods can still learn useful policies, although the learning problem is harder and often requires additional mechanisms and concepts, as in our case.

Pedestrians are represented as agents embedded in a continuous environment. First, we define a perception model: each agent receives a set of *projectors* that emit rays, providing information about nearby obstacles and goals; in addition, *proximal areas* allow the agent to perceive other agents and obstacles within a limited radius.

Agents' actions consist of continuously regulating their velocity vector, i.e., adjusting walking speed and heading. This regulation takes place three times per

Score	Event
+6	Reached final target
+0.5	Reached intermediate target (first time)
−1	Reached intermediate target (other times)
−0.5	No target in sight
−0.5	Wall or agent quite near (within 0.6 m)
−0.005	Agent near (within 1 m)
−0.001	Agent in social area (within 1.4 m)
−0.0001	End of each timestep
−6	Episode length reached

Table 1. Contributions to the reward function computation.

second. Each agent is assigned a maximum walking speed (which may differ across agents). For the experiments reported in this paper we fix two actuation limits: an agent can change its heading by at most 25° per update, and it can accelerate from rest to full speed (or decelerate to a stop) in two updates. These limits are chosen to reflect normal walking behavior, rather than jogging or running.

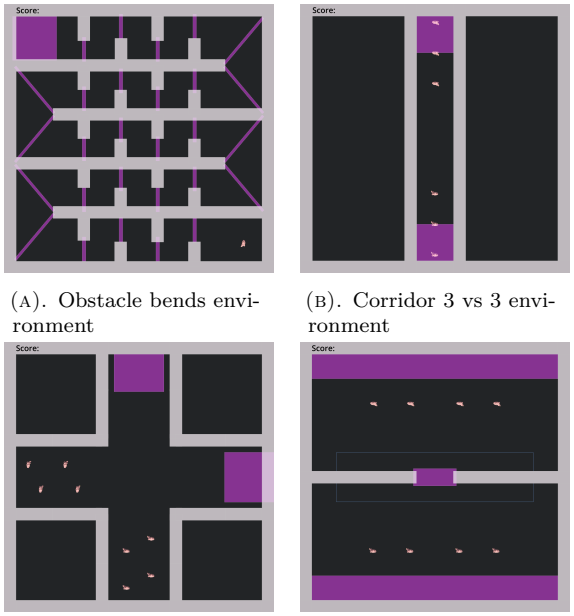
Perception informs the reward function, which evaluates each action taken by a pedestrian agent. Table 1 lists the individual contributions that events make to the reward at every decision step.

Most reinforcement-learning (RL) systems are, however, task-specific: in addition to the reward function, an environment model is supplied and a training process produces *a policy* (i.e., a decision function mapping the agent's perception to the action that should be taken) tailored to that particular setting. The policy is tuned to the exact state space, environment, and reward structure, so it usually does not generalise to anything that differs. In our case this would mean a separate training process for every environment.

The **curriculum learning** [7] approach, when applied to RL, allows the acquisition of a general policy capable of producing plausible pedestrian behaviours in environments that were not encountered during training. A curriculum is a set of scenarios (or environments) of increasing difficulty that support incremental skill acquisition. In our work the skills are: orienting toward a target, walking while maintaining an appropriate distance from walls and obstacles, and navigating around other pedestrians. Agents experience scenarios sequentially, moving to the next one only when the average accumulated reward of a set of agents over a given time period exceeds a scenario-specific threshold. While curriculum learning is a general technique, not specific to RL, it has proven to be an effective *transfer-learning* strategy for RL (see [8]). The structure of the proposed curriculum is summarized in Table 2: first, agents learn the effects of their actions (e.g., turning to locate a target is beneficial; walking too close to a wall is harmful). Next, they navigate environments of progressively greater geometric complexity, featuring bends and bottlenecks. Finally, they learn to coordinate in scenarios with relatively simple conflicts (e.g., agents moving toward the same target) and then in increasingly challenging settings, such as intersections and counter-flows. Representative environments used in the curriculum are shown in Figure 1.

High level behaviour	Environment
Walk towards target	StartEz Start
Turn towards target	Observe
Navigate narrow corridors	Start corridor
Make bends and avoid walls	Bends Obstacle bends
Avoid agents walking in the same direction	Narrow door
Avoid agents walking in conflicting directions	Corridor 3v3 T junction 4 way intersection
Combine all the above	Double narrow door

Table 2. Curriculum structure.



(a). Obstacle bends environment

(b). Corridor 3 vs 3 environment

(c). 4 way intersection environment

(d). 4 way intersection environment

Figure 1. Sample environments from the training curriculum.

A final step of the training process is characterized by a parallel retraining of the agent in a selected number of scenarios before the end of the overall training, to refresh skills acquired in earlier steps.

3. System architecture

In prior work we used the Unity game engine coupled with the ML-Agents library for agent training, simulation, and data export to external analysis

tools. Unity also enables the deployment of trained models in games, interactive applications, and even virtual-reality experiments to collect empirical data on pedestrian behaviour.

However, Unity is not an open-source platform and ML-Agents does not provide a comprehensive set of deep RL algorithms; it is primarily an ancillary tool that extends Unity for the development of non-player characters or autonomous agents based on a specific AI framework. From an open-science perspective, with the aim to improve the reproducibility of our simulation results and to support a broader range of RL algorithms, we therefore developed an analogous platform that relies solely on open-source technologies.

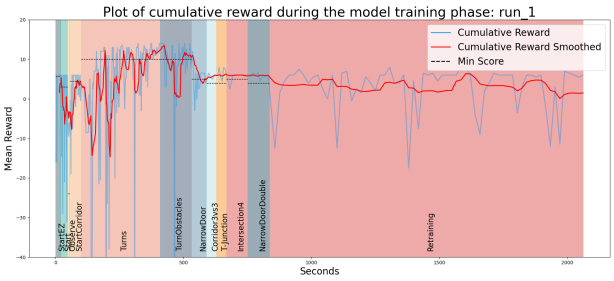
First, we identified the **Godot game engine** [9] as an open-source alternative to Unity. Godot is a free, open-source engine that supports 2D and 3D development on multiple platforms (desktop and mobile). It offers a scene system similar to Unity, an integrated scripting language (GDScript), and bindings for C#, C++, and Python. The Python bindings, in particular, enable tight integration with Python-based RL libraries.

To expose Godot's simulation state to external reinforcement-learning frameworks, we used **GodotRL Agents** [10]. This lightweight Python library implements a simple client–server protocol over WebSockets, allowing agents written in Python to interact with Godot scenes in real time.

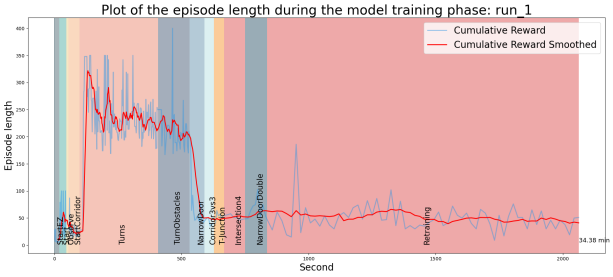
Our first RL framework was **Stable Baselines 3** [11]. Built on PyTorch, it implements a wide range of well-tested DRL algorithms (e.g., PPO, DQN, SAC, TD3). The library is designed for both research and production use, providing reproducible baselines and a consistent API. These design choices support source-level reproducibility, as all algorithmic details (hyperparameters, network architectures) are publicly documented, thereby facilitating community-driven benchmarks and comparisons.

Finally, we transitioned from a collection of ad-hoc Python data-visualization scripts to **PedPy** [12], a Python library for pedestrian-movement analysis and visualization. PedPy provides tools to compute metrics such as speed distributions, density maps, collision rates, and trajectory clustering. This choice promotes the use of standardized metrics, allowing users to generate identical quantitative measures and thereby facilitating cross-study comparisons. Moreover, all visualizations are produced programmatically, eliminating manual intervention that could introduce bias. Examples of PedPy visualizations will be shown in the next section, where we describe example applications of RL-Godot.

The only visualizations that we still produce outside of PedPy are those related to the dynamics of the training process. In particular, we generate two plots: (1) the mean cumulative reward per episode and (2) the episode duration. The training dynamics can be logged by instructing Stable Baselines to store the



(A). Mean cumulative reward during training



(B). Episode length during training

FIGURE 2. Diagrams showing the mean cumulative reward and episode duration during the training process.

desired metrics. Technically, the logging component is **TensorBoard** [13], which can also generate visualizations; however, we use it solely as a logger. We created a dedicated Python notebook that consumes the logged data and additional curriculum metadata, enriching the plotted trends with colors corresponding to the different environments in the curriculum for improved readability. The resulting plots are shown in Figure 2. In this example, the training employed Stable Baselines' implementation of Proximal Policy Optimization [14]. Note that each curriculum step has a distinct duration and mean cumulative-reward threshold. Different environments are characterized by varying numbers of intermediate targets for agent movement and different plausible path lengths from start to goal. Consequently, the reward and duration curves do not always improve monotonically; instead, they exhibit sudden drops that typically correspond to exploratory policy updates when agents try alternative actions that yield negative outcomes, prompting further exploration of the state space.

The outcome of this work is **RL-Godot Pedestrian Simulation** [15], an open-source system that allows users to (i) run simulations in planar environments (multi-floor environments are not yet supported), (ii) employ existing agent behavioral models, and (iii) customize the training pipeline to generate new models. The overall system architecture is illustrated in Figure 3.

4. Experimental applications

The policy obtained from the training process described earlier was evaluated in environments that

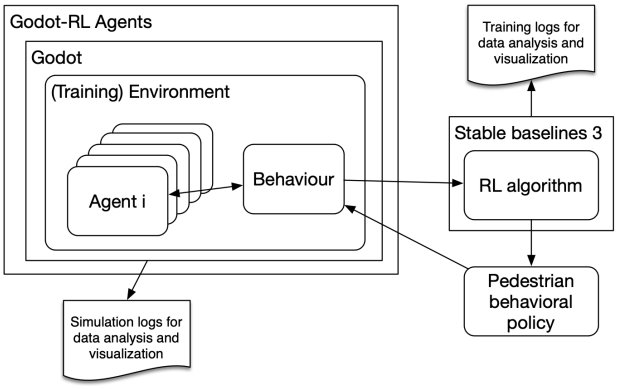


Figure 3. High level schema of the RL-Godot system architecture.

had not been presented to the agents during training. Two such test scenarios are illustrated in Figure 4a and Figure 4c.

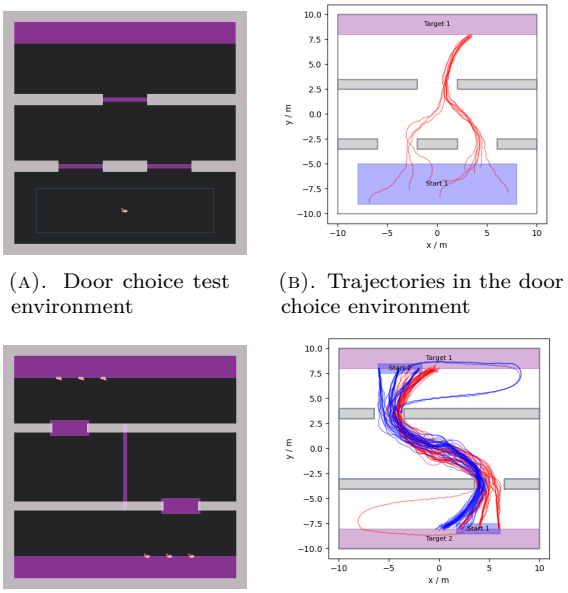
In the first scenario, a single pedestrian agent is initialized at random in the southern room. The agent can exit through two doors: one leads to a central corridor, and the other opens onto a northern room that contains the final target. Figure 4b (generated using PedPy to process logs from RL-Godot) shows the trajectories of ten independent runs. The agent's motion is plausible and largely conforms to the "least-effort principle", with smooth paths.

The second scenario depicted in Figure 4c involves six agents – three in the southern room and three in the northern room. All must pass through a single door into the central corridor, thereby avoiding collisions with teammates from their own room. Once inside, they encounter counter-flow pedestrians moving in the opposite direction. Figure 4d displays ten runs, yielding a total of 60 trajectories. A few outliers exhibit unusually long paths as the agents avoid others, and a mild turbulence is observable in the central corridor.

These tests demonstrate that the training process yields a behavioral policy capable of generalizing to novel situations, confirming that the learned rules are not over-fitted to the training environments. In general, the policy produces plausible trajectories in low-density settings but struggles to maintain consistent social interactions when space becomes contested.

While the use of game engines such as Godot or Unity, together with their physics engines, can impose intrinsic limits on handling medium-high densities, we wanted to investigate whether these limits were already reached in the test environments or whether the policy could be improved by altering the training process. To increase the agents' exposure to social interactions and conflicts, we modified several training scenarios:

- In some environments where the agent previously moved alone, additional pedestrians were introduced (e.g., obstacle bends).



(A). Door choice test environment

(B). Trajectories in the door choice environment

(C). Double door counter flow environment

(D). Trajectories in the Double door counter flow environment

Figure 4. Diagrams showing the mean cumulative reward and episode duration during the training process.

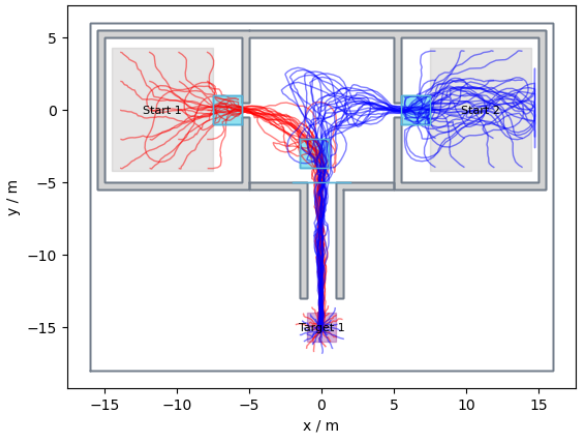
- Selected bottlenecks and passages were narrowed (e.g., narrow door, double-narrow door, T-junction, 4-way intersection).
- Some environments featured corner passages between rooms.

The reward function was also slightly altered: we retained the penalty for being within 0.6m of another pedestrian, but removed the smaller penalties applied at distances of 1 m and 1.4 m.

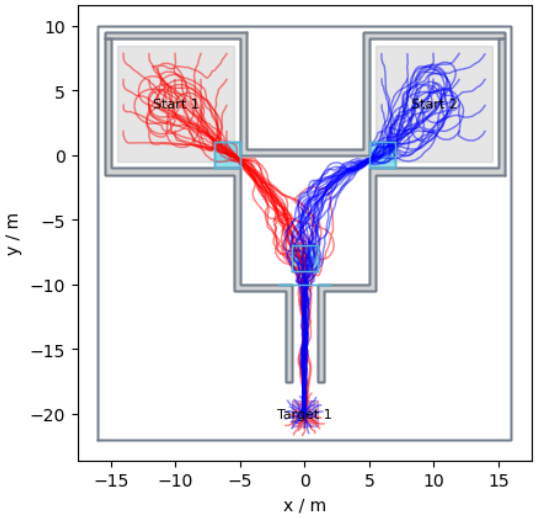
The decision to add corner passages was motivated by the results reported in [16]. That study examined how changing door positions between rooms affects evacuation dynamics using a floor-field model [17]. It considered two rooms with an intermediate "entrance hall" leading to the final exit. Two door-placement scenarios were compared: (i) doors placed in the middle of a wall, and (ii) doors placed in a corner. Simulations showed that corner-placed doors facilitated smoother, faster evacuations of the starting rooms but increased congestion in the middle room near the shared exit. Consequently, overall evacuation time was higher when doors were positioned in corners.

The new training process we carried out allowed us to achieve improvements in the pedestrian behaviors in moderate levels of density, such as those locally experienced by pedestrians in the double door counter flow scenario.

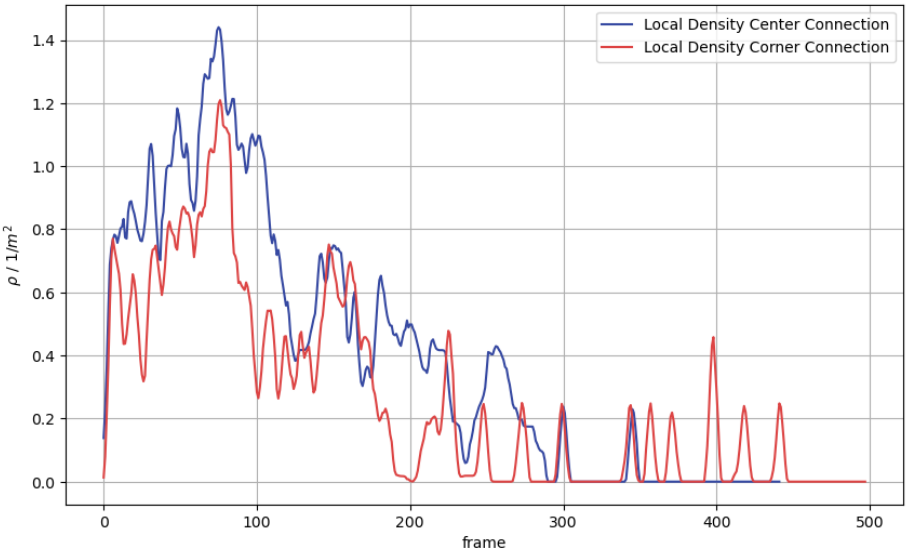
We were able to qualitatively reproduce the scenario discussed in [16], as shown in Figure 5. In particular, Figure 5a and Figure 5b respectively show trajectories in the two considered spatial configurations of the



(A). Trajectories generated by agents in the center door connection



(B). Trajectories generated by agents in the corner door connection



(C). Overall measured density in the central room

FIGURE 5. Results achieved in the scenario discussed in [16].

environment. A proper queuing behavior at the bottleneck is still not uniformly present: the trajectories show that agents sometimes exhibit a sort of movement bias, following unnecessarily long paths to keep moving rather than stopping and waiting. The curriculum, however, does not include any situation in which agents must stand still and wait for the possibility to move, so this could probably be improved adding such a situation; the reward function could also be reconsidered to penalize final trajectories excessively long compared to the optimal one.

Nonetheless, the central stylized fact discussed by the original study, i.e. that the corner door positioning leads to an overall higher evacuation time in this layout, was correctly reproduced: Figure 5c, in fact, shows the level of density in the monitored areas (highlighted as cyan squares in the previous figures) throughout the simulation. The red line is associated to the corner door positioning, and it is initially lower than the blue one. In the initial phases, in fact, pedestrians are able to more smoothly vacate the starting rooms, with corner door positioning. Nonetheless, they meet in the shared entrance hall where the congestion level leads to a longer queuing process: the blue line, associated to the mid wall door positioning, in fact, gets to zero earlier, which implies that the associated simulation ends earlier than the case of the corner positioning. Trajectory graphs were generated by PedPy, that processed logs produced by RL-Godot; analyses such as the one in Figure 5c, computing the density level dynamics in given areas of the environment throughout a simulation are also made possible and quite easy by PedPy, that represent an extremely useful complement to any pedestrian simulator.

5. Conclusions and future developments

The paper has presented RL-Godot, a Reinforcement Learning based open source software system supporting the simulation of pedestrian dynamics. After briefly introducing the curriculum based RL approach to pedestrian simulation, we introduced the RL-Godot system architecture. Finally, we described a first experimental application aimed at qualitatively reproducing results from the literature, to evaluate its adequacy to simulate low to medium density situations. The results we have achieved are promising: the curriculum based approach yields policies that generalize well the experienced situations, avoiding overfitting while building on training scenarios. Nonetheless, further research is necessary to evaluate to which extent the proposed curriculum and reward function are adequate to cover a sufficiently large set of situations. Preliminary results suggest, however, that it is possible to *fine tune* the achieved policy by adding further scenarios to the curriculum, to grant the agents additional competences.

However, the fine tuning approach has structural limitations: changing the perception model or reward function requires a new training run. An example of this process regards providing agents with the ability to perceive signs and passage information, indicating whether a path leads toward an exit or intermediate goal. This kind of setting enables training autonomous wayfinding policies. Preliminary results appear in [18], and we are working to replicate them within RL-Godot.

In conclusion, RL-Godot is still in its early stage of development, like most the research lines on RL based pedestrian simulation approaches, yet it already shows promise, not just for pedestrian simulation, but also as a tool for creating believable non-player characters (NPCs) in VR applications [19].

Acknowledgements

This work was partly developed within the Spoke 8 – "Mobility-as-a-Service (MaaS) and and Innovative services" of the National Center for Sustainable Mobility (MOST), set up by the "Piano nazionale di ripresa e resilienza (PNRR) – M4C2, investimento 1.4, Potenziamento strutture di ricerca e creazione di campioni nazionali di R&S su alcune Key Enabling Technologies" funded by the European Union. Project code CN00000023, CUP: D93C22000410001. This work was also partially supported by the MUR under the grant "Dipartimenti di Eccellenza 2023–2027" of the Department of Informatics, Systems and Communication of the University of Milano-Bicocca, Italy.

References

[1] R. S. Sutton, A. G. Barto. *Reinforcement Learning: an Introduction (2nd ed.)*. MIT press Cambridge, 2018.

[2] R. Junges, F. Klügl. Programming agent behavior by learning is simulation models. *Applied Artificial Intelligence* **26**(4):349–375, 2012. https://doi.org/10.1080/08839514.2012.652906

[3] F. Martinez-Gil, M. Lozano, F. Fernández. Emergent behaviors and scalability for multi-agent reinforcement learning-based pedestrian models. *Simulation Modelling Practice and Theory* **74**:117–133, 2017. https://doi.org/10.1016/j.simpat.2017.03.003

[4] G. Vizzari, T. Cecconello. Pedestrian simulation with reinforcement learning: A curriculum-based approach. *Future Internet* **15**(1):12, 2023. https://doi.org/10.3390/FI15010012

[5] Unity. Go create. [2026-02-12]. https://unity.com/

[6] Unity. Unity ML-Agents Toolkit. [2026-02-12]. https://github.com/Unity-Technologies/ml-agents

[7] Y. Bengio, J. Louradour, R. Collobert, J. Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48. Association for Computing Machinery, New York, NY, USA, 2009. https://doi.org/10.1145/1553374.1553380

[8] F. L. Da Silva, A. H. R. Costa. A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research* **64**:645–703, 2019. https://doi.org/10.1613/jair.1.11396

[9] The Godot Foundation. Godot game engine. [2026-02-12]. https://godotengine.org/

[10] Godot RL Agents. [2026-02-12]. https://github.com/edbeeching/godot_rl_agents

[11] Stable Baselines3. [2026-02-12]. https://github.com/DLR-RM/stable-baselines3

[12] PedPy: Pedestrian Trajectory Analyzer. [2026-02-12]. https://pedpy.readthedocs.io/

[13] TensorBoard: TensorFlow's visualization toolkit. [2026-02-12]. https://www.tensorflow.org/tensorboard

[14] J. Schulman, F. Wolski, P. Dhariwal, et al. Proximal policy optimization algorithms, 2017. ArXiv:1707.06347. https://doi.org/10.48550/arXiv.1707.06347

[15] RL-Godot-Pedestrian-Simulation. [2026-02-12]. https://github.com/Ruben-2828/RL-Godot-Pedestrian-Simulation

[16] T. Ezaki, D. Yanagisawa, K. Nishinari. Pedestrian flow through multiple bottlenecks. *Physical Review E* **86**:026118, 2012. https://doi.org/10.1103/PhysRevE.86.026118

[17] C. Burstedde, K. Klauck, A. Schadschneider, J. Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications* **295**(3):507–525, 2001. https://doi.org/10.1016/S0378-4371(01)00141-8

[18] G. Vizzari, D. Briola, T. Cecconello. Curriculum-based reinforcement learning for pedestrian simulation: Towards an explainable training process? In R. Falcone, C. Castelfranchi, A. Sapienza, F. Cantucci (eds.), *Proceedings of the 24th Workshop "From Objects to Agents", Roma, Italy, November 6-8, 2023*, vol. 3579 of *CEUR Workshop Proceedings*, pp. 32–48. CEUR-WS.org, 2023. [2023-12-14]. https://ceur-ws.org/Vol-3579/paper3.pdf

[19] D. Briola, F. Tinti, G. Vizzari. Creating virtual reality scenarios for pedestrian experiments focusing on social interactions. In M. Alderighi, M. Baldoni, C. Baroglio, et al. (eds.), *Proceedings of the 25th Workshop "From Objects to Agents", Bard (Aosta), Italy, July 8-10, 2024*, vol. 3735 of *CEUR Workshop Proceedings*, pp. 161–176. CEUR-WS.org, 2024. [2024-07-26]. https://ceur-ws.org/Vol-3735/paper_13.pdf