PANSim: Visualization Tool for Planning and Acting against Nature

Erol Medenčević[1,2], Jakub Med[1,2], Lukáš Chrpa[2]

[1]Faculty of Electrical Engineering, Czech Technical University in Prague
[2]Czech Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague
medenero@fel.cvut.cz, jakub.med@cvut.cz, chrpaluk@cvut.cz

Abstract

The demo presents a tool that visualizes the acting of planning agents in dynamic environments that might be modified by "acts of nature", The purpose of this tool is to better understand the behavior of the agent, debug agent's behavior, and for making the underlying planning concepts accessible to wider audience.

Code & Examples —
https://gitlab.com/automated-planning/tools/pansim

Introduction & Motivation

Automated planning is a fundamental concept in AI. It plays a critical role in domains where decision-making must be carried out based on a known model of the environment and a set of defined actions and constraints. The environment is often described formally using state variables and actions with preconditions and effects, allowing the planner to construct a valid sequence of actions, or a plan, from the initial state to the goal state.

At its core, planning assumes a predictable world, one in which the results of actions are known in advance, and the environment is fully observable. However, real-world scenarios often go beyond these ideal conditions. In practice, an agent may operate in environments where uncertainty is introduced, for example, by external forces beyond the agent's control, which are referred to as *exogenous events* or *acts of nature*. These events can affect the state of the world independently of the agent's actions, making planning more complex and dynamic.

This demo introduces a visualization tool that transforms planning execution into a clear and accessible graphical form. Its primary purpose is to help researchers understand, analyze, and debug the behavior of planning agents in structured domains. In addition, it supports the communication of planning concepts to broader audiences, including those unfamiliar with the underlying logic. The tool can also be used for PR purposes to advertise the "planning and acting" research for students or potential industry partners. The visualization tool currently implements two distinct planning domains, each with its own dynamic properties, such as shrinking platforms or moving obstacles influenced by external events. The tool is built with reusability in mind, allowing simulations to be executed repeatedly, compared across different problem instances, and presented in a consistent and visually coherent way. The tool currently implements the "linear plan" based agents (see the following Section), yet, it is planned to be extended to consider agent based on MDP techniques or Reinforcement learning.

Planning Against Nature

Planning against nature is a recent concept that formalizes planning tasks in the presence of exogenous events as multi-agent planning tasks for two agents, the acting *agent*, and *nature* that (randomly) trigger events (Chrpa and Karpas 2024b). The concept of planning with exogenous events has been studied for some time (Dean and Wellman 1990; Musliner, Durfee, and Shin 1993; Iocchi, Nardi, and Rosati 2000). However, addressing such planning tasks usually requires reasoning with a large portion of the state space. Methods that reason with Markov Decision Process (MDP) models (Mausam and Kolobov 2012) are a good example. These methods generate policies, assigning the most promising action in each (reachable) state. Policies, however, might not be easy to interpret and explain to human operators, who oversee the acting agent.

Focusing on *linear plans* can improve interpretability and explainability of an agent's actions, yet the agent has to consider exogenous (or nature's) events while planning and acting. A naive approach that replans (e.g., by FF-replan (Yoon, Fern, and Givan 2007)) whenever events change the environment or disrupt the plan might not be safe. Chrpa, Gemrot, and Pilát (2020) extended the planning and acting loop by adapting the notion of *safe states* (Cserna et al. 2018) in which nature cannot cause a dead-end for the agent and proposed a technique that iteratively generates *robust plans*, guaranteed to succeed regardless of nature, connecting safe states until the goal is achieved. A subsequent work of Chrpa, Pilát, and Med (2021) introduced a technique to generate *eventually applicable plans* that are guaranteed to be eventually executed. In both works, nature can execute any set of independent events (that do not interfere with each other) between actions of the agent. The most recent works consider that the nature can apply any valid and finite sequence of events between the agent's actions (Chrpa and Karpas 2024b,a).

Related Work

Techniques that tackle *planning and acting* under uncertainty were evaluated, for example, by using MDPSim or RDDLSim tools that were developed for the purpose of evaluating planners in the probabilistic tracks of the International Planning Competition. These tools implemented a client-server architecture, where the server was responsible for action execution (in an uncertain environment) and communicate the result (e.g. a current state of the environment) to the client that was responsible for providing actions (to the server). A recent tool (Chrpa, Pilát, and Med 2021) was inspired by MDPSim and RDDLSim and implemented a similar client-server architecture for *planning against nature*. That tool was implemented in C++ and accepted domain and problem definition in a modified PDDL that supported definitions of events (by using `:event` keyword instead of the `:action` keyword).

Although plain textual information may be sufficient to verify the correctness or evaluate performance, it often becomes difficult to interpret, especially as the complexity of the problem increases. For this reason, a visualization tool can be a highly valuable addition to the planning workflow.

There are several tools that can be used for classical plan visualization such as vPlanSim (Roberts et al. 2021), Planimation (Chen et al. 2020), or, the most recent PDSim (Pellegrin and Petrick 2024), which we will describe in more details (as PDSim is the main inspiration for our tool).

PDSim (Pellegrin and Petrick 2024) is a *Unity* package that enables users to create custom animations for planning domains without having to interact with the implementation of the underlying planning. Its primary goal is to facilitate the visual presentation of agent behavior by allowing users to focus on designing animations rather than solving or implementing the planning problem itself.

The core idea behind *PDSim* is to allow the user to simply provide the PDDL files describing the domain and a specific planning problem. *PDSim* then processes the input, generates a plan, and automatically creates all necessary `GameObjects` corresponding to the elements of the environment. The only task left to the user is to supply the visual assets to be assigned to the generated `GameObjects` and to create animations using *Unity's* built-in node-based system.

PDSim Backend The separation of concerns between visualization and planning logic is enabled by *PDSim's* backend component, which abstracts away the underlying planning implementation. The backend, implemented in *Python*, is responsible for processing the planning logic, computing plans from the provided PDDL files, and providing state information to the visualization layer. It utilizes the *Unified Planning Library* (Micheli et al. 2025), a modern *Python* framework that simplifies the representation and solving of planning problems through a unified interface to various planning engines. As a result, users interact exclusively with the *Unity* Editor, where they can focus on the graphical representation of the domain without having to deal with the complexities of plan generation.

Limitations of PDSim Initially, the intention was to utilize *PDSim* to visualize the domains used in this project by designing the required models and animations, while relying on the internal mechanisms of *PDSim* to handle the planning logic. However, during the process, it became clear that *PDSim* presents several limitations when applied to more complex or dynamic environments. In particular, the back-end component of *PDSim* does not support the inclusion of events of nature, which are essential in our domains.

Furthermore, *PDSim* does not offer a way to modify or extend the communication between the simulation back-end and external agents. It is tightly coupled with the *Unified Planning Library* and does not support the integration of agents implemented outside of this framework. As a result, it is not possible to connect custom agents, such as those written in a different language or using alternative reasoning models, which significantly limits the flexibility of the simulation. These limitations ultimately make *PDSim* unsuitable for the requirements of this project.

Technical Solution

Inspired by *PDSim* (Pellegrin and Petrick 2024) and the existing textual simulator (Chrpa, Pilát, and Med 2021), the visualization tool was developed for visualizing plan execution in *Planning against Nature* tasks. Currently, the tool can visualise two domains – AUV, which describes AUV operations in the area with passing ships that might endanger AUVs, and Perestroika, which is inspired by a retro game Perestroika (or Toppler)[1]. Both domains are described in the work of Chrpa, Pilát, and Med (2021).

The architecture of our tool follows a structure similar to the existing simulator; however, in this implementation, the server is written in *Python* using the *Unified Planning Library* (Micheli et al. 2025), and a front-end component was developed in *Unity*. The user interacts exclusively with this front-end.

The front-end allows the user to select the domain and problem to be simulated, start or stop the simulation, and adjust its speed. It generates game objects based on the corresponding PDDL problem file and communicates with the server. Upon receiving a state from the back-end, the front-end displays it to the user by triggering the corresponding animations of the objects. This happens twice in each turn: once to display the state after applying the agent's action and once to display the state after events of nature are applied.

Another component of the tool deals with plan generation and implements "planning against nature" (e.g. generating *eventually applicable plans* (Chrpa, Pilát, and Med 2021)).

A demonstration video, which is located in the code repository, describes the main functionality of the tool, its architecture, and demonstrates how the tool works on two examples. The video shows agents that generate *eventually applicable plans* (Chrpa, Pilát, and Med 2021) that are followed during the execution. The agent sometimes has to wait for nature to modify the environment to a desirable state (e.g. moving the ship far enough so the AUV can pass its corridor).

[1] https://en.wikipedia.org/wiki/Perestroika_(video_game)

Acknowledgements

This research is supported by the Czech Science Foundation (project no. 24-13337L), by the European Union under the project ROBOPROX (reg. no. CZ.02.01.01/00/22_008/0004590) and by the Grant Agency of Czech Technical University (project no. SGS24/140/OHK3/3T/13).

References

Chen, G.; Ding, Y.; Edwards, H.; Chau, C. H.; Hou, S.; Johnson, G.; Syed, M. S.; Tang, H.; Wu, Y.; Yan, Y.; Tidhar, G.; and Lipovetzky, N. 2020. Planimation. *CoRR*, abs/2008.04600.

Chrpa, L.; Gemrot, J.; and Pilát, M. 2020. Planning and Acting with Non-Deterministic Events: Navigating between Safe States. In *AAAI*, 9802–9809. AAAI Press.

Chrpa, L.; and Karpas, E. 2024a. On Verifying and Generating Robust Plans for Planning Tasks with Exogenous Events. In *KR*, 273–283.

Chrpa, L.; and Karpas, E. 2024b. On Verifying Linear Execution Strategies in Planning Against Nature. In Bernardini, S.; and Muise, C., eds., *Proceedings of the Thirty-Fourth International Conference on Automated Planning and Scheduling, ICAPS 2024, Banff, Alberta, Canada, June 1-6, 2024*, 86–94. AAAI Press.

Chrpa, L.; Pilát, M.; and Med, J. 2021. On Eventual Applicability of Plans in Dynamic Environments with Cyclic Phenomena. In *KR*, 184–193.

Cserna, B.; Doyle, W. J.; Ramsdell, J. S.; and Ruml, W. 2018. Avoiding Dead Ends in Real-Time Heuristic Search. In *AAAI*.

Dean, T.; and Wellman, M. 1990. *Planning and Control*. Morgan Kaufmann Publishers.

Iocchi, L.; Nardi, D.; and Rosati, R. 2000. Planning with sensing, concurrency, and exogenous events: logical framework and implementation. In *KR*, 678–689.

Mausam; and Kolobov, A. 2012. *Planning with Markov Decision Processes: An AI Perspective*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Micheli, A.; Bit-Monnot, A.; Röger, G.; Scala, E.; Valentini, A.; Framba, L.; Rovetta, A.; Trapasso, A.; Bonassi, L.; Gerevini, A. E.; Iocchi, L.; Ingrand, F.; Köckemann, U.; Patrizi, F.; Saetti, A.; Serina, I.; and Stock, S. 2025. Unified Planning: Modeling, manipulating and solving AI planning problems in Python. *SoftwareX*, 29: 102012.

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. 1993. CIRCA: a cooperative intelligent real-time control architecture. *IEEE Trans. Systems, Man, and Cybernetics*, 23(6): 1561–1574.

Pellegrin, E. D.; and Petrick, R. P. A. 2024. Planning Domain Simulation: An Interactive System for Plan Visualisation. In Bernardini, S.; and Muise, C., eds., *Proceedings of the Thirty-Fourth International Conference on Automated Planning and Scheduling, ICAPS 2024, Banff, Alberta, Canada, June 1-6, 2024*, 133–141. AAAI Press.

Roberts, J. O.; Mastorakis, G.; Lazaruk, B.; Franco, S.; Stokes, A. A.; and Bernardini, S. 2021. vPlanSim: An Open Source Graphical Interface for the Visualisation and Simulation of AI Systems. In *Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling, ICAPS 2021, Guangzhou, China (virtual), August 2-13, 2021*, 486–490. AAAI Press.

Yoon, S. W.; Fern, A.; and Givan, R. 2007. FF-Replan: A Baseline for Probabilistic Planning. In *ICAPS 2007*, 352–359.