Plan Visualisation with PDSim

Emanuele De Pellegrin, Ronald P. A. Petrick,

Edinburgh Centre for Robotics
Heriot-Watt University
Edinburgh, Scotland, United Kingdom
ed50@hw.ac.uk, R.Petrick@hw.ac.uk

Abstract

This paper presents updates on the development of the Planning Domain Simulation (PDSim) project, a plugin for the Unity game engine to simulate planning domains and plans. PDSim aims to fill the gap in planning visualisation and validation using a novel approach for translating the output of a plan into 3D or 2D animations and graphic effects. Simulating a planning problem can help users evaluate the quality of a plan and improve domain and problem modelling. In this system demonstration, we present the latest version of PDSim with new additions that aim to further help non-expert users approaching a planning problem for the first time.

Introduction

Modelling planning domains and verifying plan correctness can be challenging tasks, especially for real-world problems. Although plans could be valid, relying solely on planner output will not always help the user catch domain modelling errors that may be apparent when displayed in a more intuitive form such as a 2D or 3D visualisation (Chen et al. 2020).

While systems that use visualisation methods to illustrate generated plans do exist (Vrakas and Vlahavas 2005; Vaquero et al. 2007; Chen et al. 2020; Tapia, San Segundo, and Artieda 2015; Muise 2016; Magnaguagno et al. 2017; Le Bras et al. 2020; Roberts et al. 2021; Shah et al. 2021), PDSim (De Pellegrin 2020; De Pellegrin and Petrick 2021) offers a different approach by using the framework and components of the Unity game engine (Unity Technologies 2020) to deliver a 3D or 2D virtual visualisation of the planning problem, which can help users spot errors or incorrect definitions in the planning domain. With PDSim, the user is able to customise the model that represents the objects of the planning problem and define animations for such objects using Unity-based features, such as basic transformation (translation, rotation, scale) of an object, path following between two points on a map, playing a sound when a particular condition is met, spawning a particle system, etc. The user is able to create real-world scenes that reflect the execution environment of the planning problem, exploiting the functionality of the Unity game engine as an enhancement to existing automated planning tools.

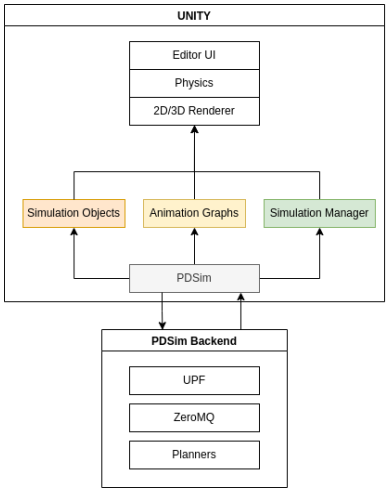


Figure 1: PDSim System Architecture

Planning Domain Simulation (PDSim)

PDSim (De Pellegrin 2020; De Pellegrin and Petrick 2021) has been developed as a plugin for the Unity game engine, a 2D and 3D game environment widely used in the video game industry. Unity offers an intuitive graphical user interface (GUI) editor and many available components, such as a built-in physics engine, realistic shaders and materials, and a path-planning library. Thanks to its modularity, it is possible to extend the interface to fit user needs, for instance, by defining custom animations for PDDL objects or by extending existing animations to reuse models from different simulations without creating them from scratch every time.

An overview of the system architecture is given in Figure 1. The front-end visual component provides the interface for users to actively interact with the system to build planning simulations, while the back-end is responsible for parsing planning files and generating plans.

A simulation is initialised and handled by the back-end server, which is integrated with the AIPlan4EU's Unified Planning Framework (UPF),[1] a recent addition to PDSim which is responsible for parsing and building a JSON representation of the planning model. UPF also handles calls

[1] https://github.com/aiplan4eu/unified-planning

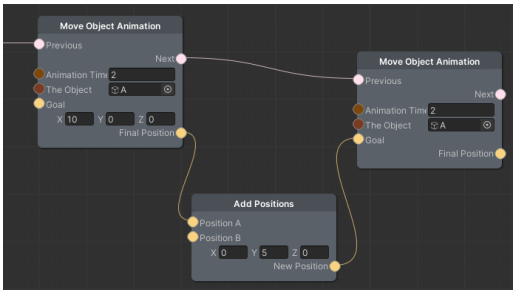


Figure 2: Node-based Animation System

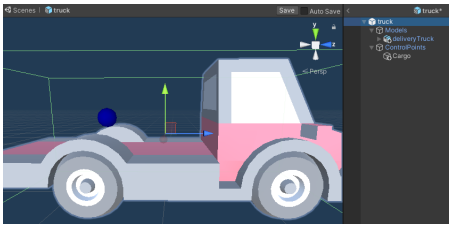


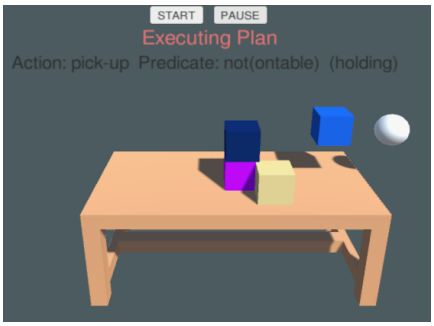
Figure 3: Simulation Object Customisation

to external planners to generate plans. UPF is a planner-agnostic framework for Python, which increases PDSim's modularity and lets users select their preferred planner implementation, separating it from the simulation stage itself which is handled later in the process. PDDL files are translated into a JSON map of the components needed for simulation. PDSim uses components of the domain such as the definitions of actions, types, and predicates to set up the core simulation in Unity. Later in the Unity editor, the user can configure multiple problems for the same domain, and thus multiple simulations for different plans.

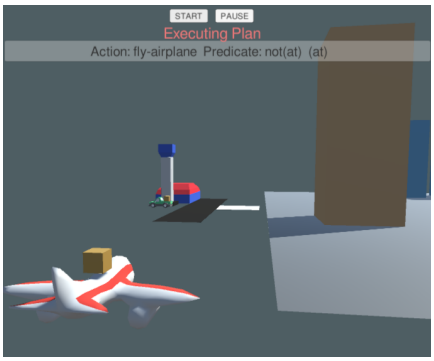
Recent changes to PDSim's animation system have also aimed to improve the ease in which animations and simulated objects are defined and customised. For instance, Figure 2 shows PDSim's new animation system which is much more intuitive. Using the system, animations can be created by adding nodes and connecting animations between them. Simulation objects are editable and more customisable. Figure 3 shows the interface to set up a planning object, where the user can insert labelled points to be used later with animations. All these new features require less interaction with the Unity editor. These features will be presented in the system demonstration, along with the core components and tools that are part of the PDSim system.

Example Simulations

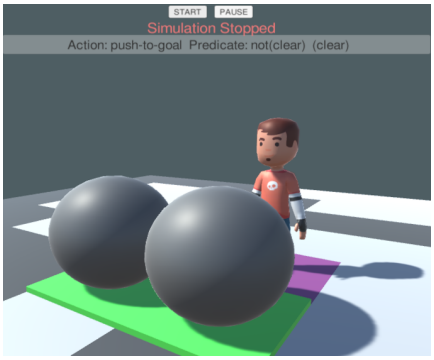
The PDSim package also includes several predefined simulations ready to be executed by the user, focusing on standard benchmarks from the International Planning Competition (IPC), as shown in Figure 4. For instance, Figure 4a shows the Blocks World domain and the block stack animation during the execution of the plan. Figure 4c shows the animation of the plan for the Sokoban domain, where the player pushes the grey spheres represent the stones to the goal tiles. Figure 4b shows the animation of the plan for



(a) Blocks World



(b) Logistics



(c) Sokoban

Figure 4: Planning Domain Simulations

the logistics domain with custom models of cities, airports, trucks, and aeroplanes. Examples from these domains will be featured in the system demonstration.

Conclusion

In this paper, we describe the features of the PDSim system, an extension to the Unity game engine to simulate planning domains and plans. PDSim focuses on user interaction to specify the animations and models relative to a given PDDL domain and problem. In this system demonstration, we will present the main PDSim front-end, integrated with Unity, and the important components of the system, such as creating custom animations from scratch and how to customise simulations. The demo will also highlight future work, such as additional tools to help simulate robotic environments.

References

Chen, G.; Ding, Y.; Edwards, H.; Chau, C. H.; Hou, S.; Johnson, G.; Sharukh Syed, M.; Tang, H.; Wu, Y.; Yan, Y.; Tidhar, G.; and Lipovetzky, N. 2020. Planimation. In *ICAPS System Demonstration System*. ArXiv:2008.04600.

De Pellegrin, E. 2020. PDSim: Planning Domain Simulation with the Unity Game Engine. In *Proceedings of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

De Pellegrin, E.; and Petrick, R. P. 2021. Automated Planning and Robotics Simulation with PDSim. In *Proceedings of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Le Bras, P.; Carreno, Y.; Lindsay, A.; Petrick, R. P. A.; and Chantler, M. J. 2020. PlanCurves: An Interface for End-Users to Visualise Multi-Agent Temporal Plans. In *Proceedings of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Magnaguagno, M. C.; Fraga Pereira, R.; Móre, M. D.; and Meneguzzi, F. R. 2017. Web planner: A tool to develop classical planning domains and visualize heuristic state-space search. In *ICAPS Workshop on User Interfaces and Scheduling and Planning*.

Muise, C. 2016. Planning.domains. In *ICAPS System Demonstration Session*.

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 International Conference on Automated Planning and Scheduling (ICAPS)*, 486–490.

Shah, N.; Verma, P.; Angle, T.; and Srivastava, S. 2021. JEDAI: A System for Skill-Aligned Explainable Robot Planning. In *Proceedings of the Demonstration Track, International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. ArXiv:2111.00585.

Tapia, C.; San Segundo, P.; and Artieda, J. 2015. A PDDL-based simulation system. In *Proceedings of the International Conference on Intelligent Systems and Agents (ISA)*.

Unity Technologies. 2020. Unity.

Vaquero, T. S.; Romero, V.; Tonidandel, F.; and Silva, J. R. 2007. itSIMPLE2.0: An Integrated Tool for Designing Planning Domains. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 336–343.

Vrakas, D.; and Vlahavas, I. 2005. A Visualization Environment for Planning. *International Journal of Artificial Intelligence Tools*, 14(6): 975–998.