When Prolog Meets Generative Models: a New Approach for Managing Knowledge and Planning in Robotic Applications – Abstract

Enrico Saccon[1,*], Ahmet Tikna[1], Davide De Martini[1], Edoardo Lamon[1], Luigi Palopoli[1] and Marco Roveri[1]

[1]*University of Trento, Trento, Italy*

Introduction. Robotics applications require robust mechanisms to represent and reason about knowledge in dynamic environments. Prolog has long been considered an effective symbolic language for robotics due to its declarative nature and support for reasoning. However, creating structured KBs manually is challenging. Recent advances in LLMsprovide new opportunities for extracting structured knowledge from unstructured input. Our research investigates how Prolog and LLMs can be combined into a unified framework for multi-agent plan generation and execution with Behavior Trees (BT).
Framework Overview. The proposed architecture follows three main steps:

(1) *Knowledge Base Generation with LLM Support*. The KB encodes facts and actions in Prolog. States are described using predicates such as `on(b1,b2,pos1)` or `av(ag1)`. Actions are defined through preconditions and effects. LLMs are employed to translate natural language queries into Prolog clauses describing initial and goal states (see Fig. 2 in the full paper).
(2) *Task Planning*. It is divided into two stages: (a) A total-order plan is computed using depth-first search in Prolog. (b) A partial-order plan and Simple Temporal Network (STN) are extracted.
(3) *BT Extraction*. The STN is compiled into a BT, enabling resilient execution on robotic platforms.

Evaluation. We evaluated the approach with a block-stacking scenario in Gazebo, using two robotic arms performing coordinated pick-and-place tasks. The system successfully generated executable BTs from KBs partially populated by LLMs. In ten scenarios, GPT-4 achieved a success rate of 0.8 compared to 0.4 for GPT-3.5 and Bard. The most common errors were misordered block stacking predicates.
Conclusions. This work shows the synergy between logic programming and generative models. Prolog ensures structured symbolic reasoning, while LLMs enhance usability by easing the KB construction. Future work will extend LLM support to action generation, integrate probabilistic reasoning, and improve search heuristics for planning.
Declaration on Generative AI: The author(s) have not employed any Generative AI tools.

Joint Proceedings of the Workshops and Doctoral Consortium of the 41st International Conference on Logic Programming, September 9–13, 2025, Rende, Italy

[*]Corresponding author.

[**]Co-funded by the European Union under NextGenerationEU (FAIR - Future AI Research - PE00000013), under project INVERSE (Grant Agreement No. 101136067), under project MAGICIAN (Grant Agree- ment n. 101120731) and by the project MUR PRIN 2020 (RIPER - Resilient AI-Based Self-Programming and Strategic Reasoning - CUP E63C22000400001).

enrico.saccon@unitn.it (E. Saccon); ahmet.tikna@unitn.it (A. Tikna); davide.demartinig@studenti.unitn.it (D. D. Martini); edoardo.lamon@unitn.it (E. Lamon); luigi.palopoli@unitn.it (L. Palopoli); marco.roveri@unitn.it (M. Roveri)
0000-0002-2418-6618 (E. Saccon); 0009-0003-4428-8376 (A. Tikna); 0009-0002-7084-5706 (D. D. Martini); 0000-0002-5526-2337 (E. Lamon); 0000-0001-8813-8685 (L. Palopoli); 0000-0001-9483-3940 (M. Roveri)