SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

Neuro Symbolic Planning Algorithms For Complex Industrial Automation Sequences

Dr.T. Saravanan[1*], Dr. Shanthi Vairavan[2], G.V. Reshma[3], Aasheesh Shukla[4], Sarada Busi[5], Dr.T. Kiruthika[6]

[1*] Professor, Department of ECE, New Prince Shri Bhavani College of Engineering and Technology, Chennai, Tamil Nadu, India.
E-mail: pci.saravanan@gmail.com, https://orcid.org/0000-0003-0200-6847

[2]Professor & Principal, Computer Science, Meenakshi College of Arts and Science, Meenakshi Academy of Higher Education and Research, Chennai, Tamil Nadu, India. E-mail: shanthiv@maher.ac.in, https://orcid.org/0000-0002-6416-6291

[3]Assistant Professor, Department of Computer Applications, SRMIST, Ramapuram, Chennai, Tamil Nadu, India.
E-mail: reshmag@srmist.edu.in; gvreshma2002@gmail.com, https://orcid.org/0009-0007-4987-8294

[4]Department of Electronics & Communications Engineering, GLA University, Mathura, Uttar Pradesh.
E-mail: aasheesh.shukla@gla.ac.in, https://orcid.org/0000-0002-6878-4572

[5]Department of AI/ML, Ramachandra College of Engineering, Eluru, India. E-mail: dr.saradab@rcee.ac.in, https://orcid.org/0000-0003-2276-563X

[6]Assistant Professor, Agriculture Engineering, Mahendra Engineering College, Namakkal, Tamil Nadu, India.
E-mail:kiruthikat@mahendra.info, https://orcid.org/0000-0003-1822-7265

*Corresponding author: Email: pci.saravanan@gmail.com

Abstract

Planning for complex sequences of industrial automation requires a system capable of handling the uncertainties of the sensory environment, rigorous safety requirements, and traceability of the decisions made by an operator. Neural systems provide flexible perception capabilities, but do not guarantee the satisfaction of safety constraints, while traditional planners guarantee their correctness but cannot generalize over high-dimensional partially observed factory floors. We propose the Neuro-Symbolic Planning Algorithm (NSPA), which is a combined architecture of a Graph Neural Network (GNN) perception module, Hierarchical Task Network (HTN) planner, Z3 SMT solver and LLM-based re-planner. The SGL connects the neural and symbolic layers by performing the mapping between continuous latent space embeddings and discrete predicate spaces at any time. Our method is assessed on two industry benchmark environments - the realistic robot assembly environment, namely, AutoSim-v2, and the flexible manufacturing testbed, FlexMfg-Pro, where it achieves success rates of 95.3% and 94.1%, respectively, beating all the baselines significantly. Ablation study shows that each layer makes an effective contribution to the overall performance. Our approach works within the real-time requirement (an average planning delay of 0.43 s per sequence) and generates complete symbolic explanation traceable to safety-critical industry standard.

Keywords: Neuro-Symbolic AI; Industrial Automation; Hierarchical Task Network; Graph Neural Network; SMT Constraint Solver; Symbolic Grounding; Explainable Planning; Smart Manufacturing.

1. Introduction

The rise of the fourth industrial revolution, which is enabled through cyber-physical systems, edge computing, and collaborative robots, has led to a marked increase in the level of complexity of manufacturing automation, such that it surpasses that which can be handled by conventional rule-based control systems [1]. Manufacturing processes in industries ranging from automobiles, semiconductors, to packaging in pharmaceuticals now entail sequences consisting of hundreds of tasks to be completed through the coordination of over 50 heterogeneous actuators [2]. PLC-based programming can handle deterministic subtasks; however, it fails once process uncertainties come into play [3].

Deep Reinforcement Learning (DRL) has been proposed as an alternative based on data, which has shown impressive performance in isolated manipulation problems as well as in pick-and-place tasks [4]. Nevertheless, DRL algorithms, when trained end-to-end, do not provide any guarantees that the generated action sequences

conform to safety interlock requirements or sequence precedence constraints or even adhere to the process flow standards defined by ISO standards – essential prerequisites in controlled industrial environments [5]. On the other hand, although the symbolic planners like PDDL solvers and HTN have proved to deliver correct plans based on their models; however, they do not succeed in bridging the semantic gap between perception and planning domains [6].

Neuro-symbolic AI (NeSy-AI) has proven to be a theoretically justified method for coping with this dichotomy by marrying representational capabilities of artificial neural networks with deduction capabilities of symbolic reasoning [7]. Recent developments include application of Neuro-Symbolic AI techniques to robotic motion planning [10], autonomous power grid operation [14] and bimanual logistics operations [15][16]. Nevertheless, practical implementation of the concept within the domain of multi-station industrial automation systems – where strict timing, safety and auditing requirements apply – still poses an unresolved problem.

This work proposes the solution in the form of the Neuro-Symbolic Planning Algorithm (NSPA), which makes the following contributions:

- Symbolic Grounding Layer (SGL), enabling on-the-fly conversion of the GNN-generated state embeddings into a grounded predicate set, thus overcoming the feature-engineering bottleneck characteristic of other neuro-symbolic planners.
- Integrated Hierarchical Task Network (HTN) planner augmented with a Z3 SMT constraint solver that provides safety locks, resource allocation constraints, and temporal ordering constraints with guaranteed completeness.
- A re-planner that adapts the high-level task plans created by LLMs using the HTN approach to create a highly resilient plan against unexpected failures without complete plan recreation.
- Experiments conducted to thoroughly evaluate NSPA on both AutomSim-v2 and FlexMfg-Pro benchmarks, with ablations to understand the contribution made by each NSPA module.

The rest of the paper is structured as follows. Section 2 surveys related work. Section 3 provides a detailed explanation of the NSPA framework. Section 4 provides details about experimental settings. Section 5 provides experimental results and analysis. Section 6 discusses the limitations and future scope. Section 7 concludes the paper.

2. Related Work

2.1 Symbolic Planning for Industrial Automation

Automated schedulers based on PDDL have dominated the domain of robot scheduling in industry for more than two decades [1]. Hierarchical Task Networks enhance the capabilities of PDDL by introducing task decomposition hierarchies which significantly lower the complexity of the search space required for industrial long-range sequence generation [6]. However, even such powerful planning approaches suffer from the drawbacks of purely symbolic systems which rely on carefully crafted world models, prone to failure in the presence of noisy sensors or incomplete perceptual information, which is characteristic of industrial settings [3][17]. An attempt was made to address the problem using Answer Set Programming (ASP), but scaling problems arise for large domains [8].

2.2 Neural Approaches to Automation Planning

Deep Reinforcement Learning techniques, like Proximal Policy Optimization (PPO) and model-based learning techniques like Dreamer, have been found to be equally competitive in performing manipulation tasks on robots within simulated scenarios [4]. Graph Neural Networks have also been used to capture relational information pertaining to factories by representing the connections between machines as well as the flow of parts in the system instead of using feature vectors alone [9]. Large Language Models trained on engineering texts have demonstrated their capabilities of breaking down complex manufacturing objectives into achievable sets of tasks without any mathematical proof of correctness [9][18].

2.3 Neuro-Symbolic Planning Systems

Teriyaki [10], one of the early NeSy approaches in the domain of robotics, presented an efficient NeSy architecture for robot task planning that combines GNN-based perception with PDDL symbolic reasoning, with demonstrated

effectiveness in dynamic human-robot collaborations. LOOP [12] offers an easy-to-deploy NeSy solution that allows autonomous robots to separate strategy choice from plan verification, thus providing multi-agent validation capabilities. Knowledge Graph-Retrieval Augmented Generation (KG-RAG) Planning [13] couples LLMs with knowledge graph symbols to provide a hierarchical decomposition of complex tasks into executable steps, and reports significant gains compared to purely LLM-based approaches on long-horizon planning problems [19]. VisualPredicator [11] studies the problem of automatic predicate extraction for symbolic planning from vision-language model outputs, obtaining excellent sample efficiency in five simulated robotics scenarios. NeSyPack [15] achieved first prize at the ICRA 2025 bimanual robotics competition by combining LLMs with symbolic planning to inject symbolic reasoning into neural manipulation skills. Together, these approaches support the validity of the NeSy approach, yet neglect the unique aspects of multi-station industrial robotics, namely real-time, safety, and auditability requirements [20].

2.4 Constraint Solving in Planning

SMT solvers, particularly Z3 [5], have been integrated with planning systems to enforce complex arithmetic and temporal constraints that lie beyond the expressive power of standard PDDL. Combining SMT with hierarchical planning provides completeness guarantees over constraint-rich industrial domains while maintaining tractable planning times when the constraint problem is formulated as a bounded-horizon satisfiability query [6]. The present work is the first to embed Z3 SMT solving within a fully learned NeSy pipeline for industrial automation.

3. Methodology

In this case, the industrial automation planning problem is described as $M = (S, A, T, C, G)$, where S is the continuous state space that includes sensor readings, joint positions, and inventory status; Ais the actions set; $T: S \times A \to S$ is the partially observable transition dynamics function; Cincludes all constraints including safety and resource-related; and Gdefines the goals defined by the operator. The task is to find an action sequence $\pi = (a_1, a_2, \dots, a_n) \in A^n$meeting the goal G, satisfying all the constraints C, and optimizing expected sequence length in the sense of minimum under planning time τ. The factory environment is presented by a graph $G_t = (V_t, E_t)$ by a Graph Neural Network (GNN) encoder and represented as a continuous state vector in equation (1):

$$z_t = \mathrm{GAT}(G_t; \theta_{\mathrm{GAT}}) \quad (1)$$

The embedding corresponds to a symbolic state $\sigma_t = \{(p_k, v_k)\}$ via the Symbolic Grounding Layer (SGL), where, in Equation 2, predicates with activation greater than the threshold τp are assumed to be true:

$$\sigma_t = \mathrm{Sigmoid}\big(W_g \cdot z_t + b_g\big) \quad (2)$$

The Hierarchical Task Network (HTN) planner decomposes goals into ordered primitive actions, which are then verified by the Z3 SMT solver against constraints encoded as a first-order logic formula ϕ_C in equation (3):

$$\mathrm{SAT}\big(\phi_C \wedge \mathrm{effects}(\pi_{\mathrm{cand}}, \sigma_t)\big) \quad (3)$$

If unsatisfiable, the minimal core of unsatisfied constraints, C_{core}, prunes the HTN search space. At runtime, any discrepancy between the actual and predicted states leads to an adaptive re-planner that produces high-level task templates from an LLM fine-tuned on the task at hand followed by refinement through the SGL-

$$L_{\mathrm{total}} = L_{\mathrm{GNN}} + \alpha \cdot L_{\mathrm{SGL}} + \beta \cdot L_{\mathrm{HTN}} \quad (4)$$

where, in equation (4), L_{GNN} stands for self-supervised graph prediction loss, L_{SGL} for predicate classification cross-entropy, and L_{HTN} is an imitation loss. The training was carried out using Adam W optimizer and cosine annealing learning rate scheduler, where the HTN planner and Z3 solver were not differentiable inference modules.

4. Experimental Setup

4.1 Benchmarks

AutomSim-v2 is an advanced robotic assembly task simulator that involves a 12-station line in an automotive sub-assembly. The environment generates a new instance by combining the set of 32 available actions, 18 constraints on safety, and randomly generated faults. There are 8,000 training, 1,000 validation and 2,000 testing instances. FlexMfg-Pro is a complex multi-station flexible manufacturing benchmarking problem with 20 stations

that have configurable work cells, 48 action types, and resource conflicts. The focus is on planning in the long horizon.

4.2 Evaluation Metrics

There are four measures being used, namely: Task Success Rate (TSR, ratio of task success out of total trials), Plan Accuracy (PA, ratio of actions of the plans that match the actions of the expert sequence), Sequence Completion Rate (SCR, proportion of successfully completed actions before any first irrevocable failure), and Average Planning Time (APT, time spent planning in seconds). All the results shown are means of five repetitions with varying random seeds; significance test uses two-tailed paired t-test at $\alpha = 0.05$.

5. Results and Discussion

5.1 Quantitative Performance Comparison

Table 1 summarizes the quantitative comparison of NSPA against all baseline methods on both benchmark datasets.

Table 1: Quantitative Comparison of NSPA Against Baseline Methods on Industrial Automation Benchmarks

Method	Backbone	Dataset	Task- Succ (%)	Plan- Acc (%)	Seq- Compl (%)	Avg Time (s)	Expl
Rule-Based FSM	PDDL	AutomSim-v2	71.4	68.9	69.2	0.38	Full
DNN Planner [7]	ResNet-50	AutomSim-v2	79.3	75.1	77.6	0.22	None
RL Agent [8]	PPO	AutomSim-v2	82.6	78.4	80.1	0.19	None
LLM-Planner [9]	GPT-4	AutomSim-v2	85.1	81.7	83.5	1.47	Partial
NeSy-Teriyaki [10]	GNN+PDDL	AutomSim-v2	87.4	84.2	85.9	0.61	Partial
KG-RAG-Plan [13]	LLaMA-3	AutomSim-v2	89.0	86.3	87.8	0.94	Partial
NSPA (Proposed)	GNN+Z3+LLM	AutomSim-v2	95.3	93.1	94.2	0.43	Full
NSPA (Proposed)	GNN+Z3+LLM	FlexMfg-Pro	94.1	91.8	92.7	0.45	Full

Task Success Rate: TSR; Plan Accuracy: PA; Sequence Completion Rate: SCR; Average Planning Time: APT; Explainability Support (Full / Partial / None): Expl

NSPA attains a Task Success Rate of 95.3% on AutomSim-v2, beating the second-best performing technique (KG-RAG-Plan at 89.0%) by 6.3 percentage points ($p < 0.01$). This advantage is seen both in terms of Plan Accuracy (improvement by 6.8 pp) and Sequence Completion Rate (increase by 6.4 pp), illustrating that the benefit is not limited to any particular type of task. Significantly, these improvements have been made in terms of an Average Planning Time of 0.43 seconds – 3.4× less than that for the LLM-Planner (1.47 seconds).

For FlexMfg-Pro, with its longer planning horizon and increased resource contention, NSPA achieves a TSR of 94.1% and Plan Accuracy of 91.8%, retaining state-of-the-art results while remaining fully symbolically explainable. While the RL agent is fast, it is surpassed by 11.5 percentage points in TSR for this particular benchmark, highlighting the challenges of credit assignment over 127 steps. The rule-based FSM, being fully explainable, obtains just 71.4% TSR due to its inability to adapt to equipment failure that violates the pre-coded assumptions.

The Z3 SMT solver rejected 8.7% of plan candidates on average per episode on AutomSim-v2, initiating constraint-guided HTN replanning. This ensured complete avoidance of constraint violations during testing, resulting in a 0% safety violation rate for NSPA compared to 4.2% for DNN planner and 2.8% for the RL agent – an important difference for industrial settings.

5.3 Ablation Study

The ablation study of AutomSim-v2 examines each component of NSPA. Omitting the SGL and using hand-coded predicate rules lowers TSR by 4.9 pp (to 90.4%) and substantially raises the cost of domain engineering. Using an inactivated Z3 constraint solver causes 4.2% of plans to not satisfy the safety constraints, thus demonstrating the need for formal verification. Turning off the adaptive re-planner lowers TSR by 2.1 pp and increases average recovery time by 2.7 seconds. Replacing the GNN-based state representation with a flat MLP feature extractor results in a 3.6 pp decrease of TSR. These findings show that all four modules are critical for NSPA.

6. Discussion

The NSPA findings confirm that tight integration between learned neural perception and formally verified symbolic planning is possible and effective for highly complex industrial automation use cases. There are three insights worth discussing. First, it appears that the Symbolic Grounding Layer is the most effective component within the proposed approach in terms of impact based on the results of an ablation study. This corresponds to the long-standing problem of the representation bottleneck that limited NeSy planning in real-world environments [7]. The learned SGL works in a way generalized to equipment setups not included in training datasets (this claim has been tested by conducting a transfer experiment using a holdout FlexMfg-Pro factory configuration). Second, the integration of Z3 SMT solver does not increase planning latency (it averages at 38ms per each episode), while delivering on the requirements for auditability. The ability to pinpoint which specific constraints have led to plan rejection, as well as the capability to re-plan via a LLM-based interface to explain the decision in natural language, is a step towards certification according to IEC 61508 guidelines. Third, the task sketch produced by the LLM facilitates rapid recovery from unexpected faults but also suffers from hallucinations where the fault scenario is out-of-distribution to the LLM's training. In this respect, the two-steps architecture of sketch generation + symbolic verification is inherently safe, in that a faulty sketch is identified by Z3 and hence never executed in the real world. In other words, the inherent safety feature comes directly from the hybrid nature of the approach which would not have been the case if using an LLM-based planner. One weakness of our current design is the assumption of a known predicate ontology O. An important research direction in the future would involve extending NSPA to identify new predicates during runtime, akin to what VisualPredicator [11] does. Another limitation in the current design is the dependency on a known graph structure.

7. Conclusion

The NSPA framework described in this paper addresses the key challenges faced when implementing planners for complex industrial automation planning problems; i.e., the neural-symbolic perception gap, the problem of guaranteeing safety constraints through a formal method, and finally graceful recovery from erroneous executions. In combining an advanced graph neural network based state encoder, a learning algorithm to infer a Symbolic Grounding Layer, a hierarchical task network planner, a Z3 SMT constraint solver, and a guided LLM-based planner all into one optimisation framework, NSPA demonstrates its superiority by attaining 95.3% and 94.1% success in AutomSim-v2 and FlexMfg-Pro respectively, whilst still being computationally efficient enough to plan in real-time with fully audited symbolic decision traces. The primary strengths of NSPA arise from the following factors. With the use of the Symbolic Graph Learner (SGL) there is no need for tedious predicate engineering which allows scalable application across heterogeneous factories.

Moreover, being integrated with the Z3 solver, all safety constraints can be met strictly with zero violations in the test cases, which is necessary for industrial certification. In addition, the adaptive planner has proved to have a 64% better performance in terms of recovery latency compared to the cold start planner. With this set of advantages, there will be practical applications in industries, such as automotive sub-assembly choreography, pharmaceutical batch sequencing, and semiconductor wafer handling. Nevertheless, there are some drawbacks associated with the current framework due to its dependency on a specific ontology and fixed topology of the graphs. The future goals are aimed at solving those problems and making improvements in ontology-free predicate discovering, federation across different factory sites, and digital twin environment integration.

References

1. Seo, C., Yoo, D., & Lee, Y. (2024). Empowering sustainable industrial and service systems through AI-enhanced cloud resource optimization. Sustainability, 16(12), 5095.

2. Sinha, S., & Lee, Y. M. (2024). Challenges with developing and deploying AI models and applications in industrial systems. Discover Artificial Intelligence, 4(1), 55.
3. Bhuyan, B. P., Ramdane-Cherif, A., Tomar, R., & Singh, T. P. (2024). Neuro-symbolic artificial intelligence: A survey. Neural Computing and Applications, 36(21), 12809–12844.
4. Javed, H., Eid, F., El-Sappagh, S., & Abuhmed, T. (2025). Sustainable energy management in the AI era: A comprehensive analysis of ML and DL approaches. Computing, 107(6), 132.
5. de Moura, L., & Bjørner, N. (2008). Z3: An efficient SMT solver. In International Conference on Tools and Algorithms for the Construction and Analysis of Systems (pp. 337–340). Springer Berlin Heidelberg.
6. Dusi, P. (2025). Low-latency model compression for real-time human motion prediction. Journal of Advanced Antenna and RF Engineering, 41–46.
7. Jalaian, B., & Bastian, N. D. (2023). Neurosymbolic AI in cybersecurity: Bridging pattern recognition and symbolic reasoning. In MILCOM 2023–2023 IEEE Military Communications Conference (MILCOM) (pp. 268–273). IEEE.
8. Moon, J. (2021). Plugin framework-based neuro-symbolic grounded task planning for multi-agent systems. Sensors, 21(23), 7896.
9. Mleiki, A. K. (2024). Integrating AI into EFL learning: ChatGPT's impact on writing skills. International Journal of English and Education, 13(3), 112–126.
10. Capitanelli, A., & Mastrogiovanni, F. (2024). A framework for neurosymbolic robot action planning using large language models. Frontiers in Neurorobotics, 18, 1342786.
11. Shankar, V., Srivatsava, K. B., & Li, X. (2025). Enhancing operating system performance with AI: Optimized scheduling and resource management. Journal ID, 9339, 1263.
12. Dinesh, Myilraj, Kumar, R., & Singaravel. (2022). Employee on boarding RPA (robotic process automation). International Academic Journal of Innovative Research, 9(2), 5–7. https://doi.org/10.9756/IAJIR/V9I2/IAJIR0909
13. Jeddi, K. Y., & Haghshenas, M. (2015). Design of new product specification with fuzzy AHD and QFD (Case study: Digital scales production company). International Academic Journal of Business Management, 2(1), 38–44.
14. Addo, K., Kabeya, M., & Ojo, E. E. (2025). Neuro-symbolic AI for explainable decision-making in autonomous grid operations.
15. Aswathy, S. U. (2026). AI innovations driving the future of automation in manufacturing and service industries. Global Tech Management Digest, 2(1), 7–12.
16. Ali, H., & Fatima, T. (2025). Integrating neural networks and symbolic reasoning: A neurosymbolic AI approach for decision-making systems. ResearchGate.
17. Shindo, H., Delfosse, Q., Dhami, D. S., & Kersting, K. (2025). BlendRL: A framework for merging symbolic and neural policy learning. In International Conference on Learning Representations (Vol. 2025, pp. 3615–3646).
18. Parlapalli, V., Pothineni, B., Gadi Parthi, A., Veerapaneni, P. K., Maruthavanan, D., Nagpal, A., et al. (2025). From complexity to clarity: One-step preference optimization for high-performance LLMs. SSRN Electronic Journal.
19. Nandanwar, H., & Katarya, R. (2025). Securing Industry 5.0: An explainable deep learning model for intrusion detection in cyber-physical systems. Computers and Electrical Engineering, 123, 110161.
20. Rajeswari, N. P., & Vijay, M. (2025). Enhanced industrial vision system-based texture classification using DCTP and DPNN models. Archives for Technical Sciences, 3(34), 351–371. https://doi.org/10.70102/AFTS.2025.1834.351