

# LLM-Assisted Tool for Joint Generation of Formulas and Functions in Rule-Based Verification of Map Transformations

Ruidi He

*Institute for Software and Systems Engineering  
Technische Universität Clausthal  
Clausthal-Zellerfeld, Germany  
ruidi.he@tu-clausthal.de*

Yu Zhang

*Institute for Software and Systems Engineering  
Technische Universität Clausthal  
Clausthal-Zellerfeld, Germany  
yu.zhang.1@tu-clausthal.de*

Meng Zhang

*Institute for Software and Systems Engineering  
Technische Universität Clausthal  
Clausthal-Zellerfeld, Germany  
meng.zhang@tu-clausthal.de*

Andreas Rausch

*Institute for Software and Systems Engineering  
Technische Universität Clausthal  
Clausthal-Zellerfeld, Germany  
andreas.rausch@tu-clausthal.de*

**Abstract**—High-definition map transformations are essential in autonomous driving systems, enabling interoperability across tools. Ensuring their semantic correctness is challenging, since existing rule-based frameworks rely on manually written formulas and domain-specific functions, limiting scalability.

In this paper, We present an LLM-assisted pipeline that jointly generates logical formulas and corresponding executable predicates within a computational FOL framework, extending the map verifier in CommonRoad scenario designer with elevation support. The pipeline leverages prompt-based LLM generation to produce grammar-compliant rules and predicates that integrate directly into the existing system.

We implemented a prototype and evaluated it on synthetic bridge and slope scenarios. The results indicate reduced manual engineering effort while preserving correctness, demonstrating the feasibility of a scalable, semi-automated human-in-the-loop approach to map-transformation verification.

**Index Terms**—Autonomous driving, map transformation, rule-based verification, large language models, first-order logic.

## I. INTRODUCTION

Autonomous driving systems rely on high-quality digital maps to support perception, planning, and control tasks [1], [2]. These maps are available in different formats, such as OpenDRIVE [3] and CommonRoad [4], each offering a specific geometric and semantic abstraction of the real world. During system development and simulation, it is often necessary to transform maps from one representation to another for compatibility across tools and platforms.

Such transformations go beyond mere syntactic changes: they involve semantic transformations that may require complex computations, potentially introducing information loss or approximation errors [5], [6]. For example, geographic coordinates (latitude/longitude) may be converted to Cartesian coordinates with elevation, or roads may be represented as polylines in one format and as polynomial curves in another.

Ensuring that such transformations preserve critical semantic and geometric information is essential for safe autonomous driving.

The correctness of these transformations is crucial, yet there is no simple universal criterion to verify them. While transformation tools could be tested with extensive test suites, such tests are costly to construct and cannot cover the full range of transformation cases due to the complexity of map data [6].

An alternative is to validate the transformation result itself using rule-based verification frameworks, such as the CommonRoad verification framework proposed by Althoff et al. [7]. These systems check for structural and semantic correctness using formalised rules. However, writing such rules manually is time-consuming, error-prone, and inevitably incomplete, especially as new semantic domains are introduced.

Recent work has shown that large language models can generate Boolean logic, propositional logic, and even more expressive systems like predicate logic (first-order logic) [8], [9]. However, producing complex logical structures remains challenging and prone to inaccuracies [10]. Consequently, many existing LLM-based verification approaches target computational first-order logic (FOL), where formulas are expressed symbolically, and the atomic symbols (predicates and functions) are implemented in imperative code and invoked during formula evaluation [11]. This paradigm has proven highly effective in rule-based verification, for example in the context of SMT solving (e.g., SMT-LIB) [12].

Yet, when verification must be extended to new semantic domains, such as validating complex map transformations, it is insufficient merely to generate new FOL formulas. In these cases, the LLM must also propose the corresponding

functions that can be invoked within formulas to access and process domain-specific information [9]. In effect, formulas and functions are inherently interdependent, and both must be generated in tandem to form an executable ruleset.

To the best of our knowledge, no existing research has explored using LLM for the joint generation of logical formulas and corresponding functions in computational FOL to enable scalable verification across new semantic domains. In particular, combining LLM-generated formulas with semantically defined function implementations within an integrated framework for map transformation validation remains an open and unexplored research direction [5], [7].

**Research Challenge:** The central challenge of this work is to investigate whether large language models (LLMs) can be leveraged to generate both formulas and their corresponding functions for use in computational first-order logic (FOL) within the context of rule-based verification of map transformations. Unlike prior approaches that only target symbolic rule generation, this requires the LLMs to co-generate logical rules and the executable functions they depend on, ensuring that the extended ruleset is directly integrable into existing verification frameworks such as CommonRoad verification framework.

#### Research Questions:

- **RQ1:** What should the architecture of a pipeline look like that enables an LLM-assisted tool to generate executable rules and functions for rule-based verification of map transformations?
- **RQ2:** Which prompting strategies and LLM approaches are most effective for jointly generating logical formulas and corresponding function definitions in this context?
- **RQ3:** How can the effectiveness of the proposed approach be demonstrated through case studies, and which evaluation metrics best capture its utility in real-world transformation verification scenarios?

As a proof of concept, we build on the CommonRoad Scenario Designer [5], a tool capable of transforming maps between formats such as OpenDRIVE and CommonRoad. While the tool supports various geometric and semantic conversions, it currently lacks functionality for handling elevation data. At the same time, a computational FOL-based verification pipeline for transformation correctness already exists [7]. In this work, we extend the verification domain to include elevation and develop an LLM-assisted prototype tool pipeline capable of jointly generating formulas and functions for this new domain, which we evaluate in a targeted case study.

The contributions of this paper are threefold.

- 1) **Conceptual foundation for tool support in computational FOL verification:** We propose an architecture for an LLM-assisted tool pipeline that jointly generates logical formulas and their corresponding function definitions, thereby answering *RQ1* on how such a pipeline should be structured.
- 2) **Tool-oriented case study with CommonRoad:** We extend the CommonRoad Scenario Designer [7] and its verification pipeline to include elevation as a new

semantic domain. This demonstrates the effectiveness of prompting strategies and LLM-based approaches for jointly producing rules and functions, directly engaging with *RQ2*.

- 3) **Prototype tool implementation and evaluation:** We implement a semi-automated, human-in-the-loop prototype tool and evaluate it on synthetic scenarios with systematically injected errors. This evaluation demonstrates feasibility and provides evidence toward *RQ3*, concerning how the proposed approach can be assessed in practice.

## II. RELATED WORK

### A. Rule-Based Map Checking and Verification

Several works apply rule-based verification to vector-based HD maps. [13], [14] define checks from national specifications (e.g., topology consistency, format compliance), but these remain checklist-style validations tied to local standards. [15] detects per-element changes in lanelets or crosswalks, yet such element-wise detection lacks the abstraction needed for reasoning about higher-level semantics. [16] validates logical and structural integrity across formats, but relies on predefined templates that limit flexibility. By contrast, the CommonRoad verifier [7] applies computational FOL for composable, explainable rules, offering more general and rigorous reasoning. However, it still requires manual domain-specific predicates, limiting scalability.

### B. Natural Language to Formal Logic with Language Models

Recent work has applied LLMs to generate formal logic from natural language, typically in FOL or MTL. Examples include ChatRule [17] (FOL via prompting), TR2MTL [18] (traffic rules to MTL with chain-of-thought), SafePlan [19] (robotic safety as LTL invariants), and GenAI [20] (OCL from structured requirements). Other recent efforts include nl2spec [21], which translates natural language into LTL with interactive user feedback, and AutoSpec [22], which synthesizes program specifications (pre/postconditions, invariants) via LLMs combined with static analysis.

While these approaches show the potential of symbolic rule generation, they all assume that predicates are predefined and fixed. This rigidity limits adaptability to new domains, where both formulas and executable functions are required. Our work bridges this gap by combining logic generation with automatic synthesis of predicate functions, enabling scalable extension of computational FOL verification frameworks.

## III. TOOL APPROACH

### A. Pipeline Overview

As illustrated in Fig. 1, this design directly addresses *RQ1* by defining the architecture of an LLM-assisted tool pipeline that integrates rule generation and function synthesis into the CommonRoad verification framework. The overall workflow consists of two components:

- 1) **CommonRoad verification framework (existing system):** Contains formal rules, the ANTLR-based rule

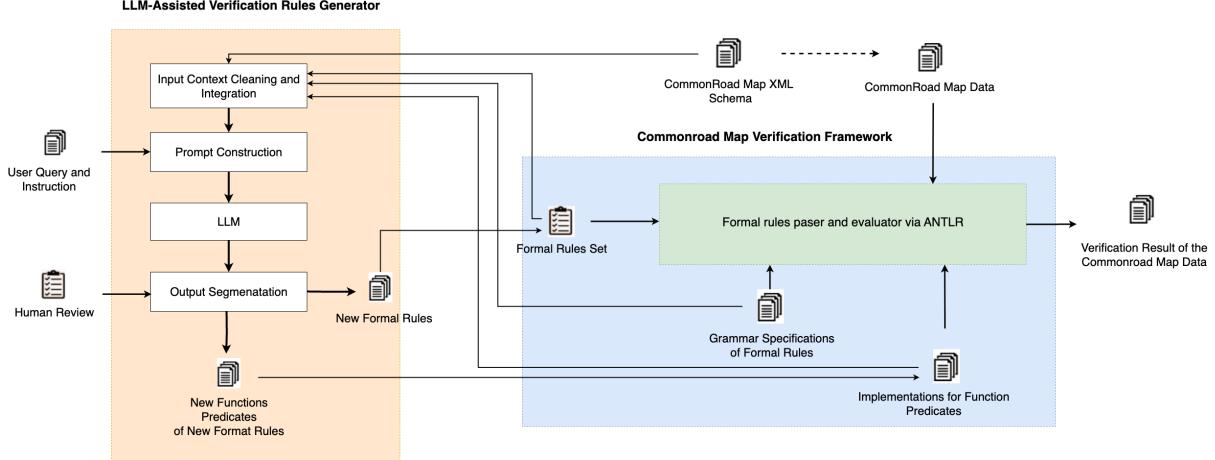


Fig. 1. Architecture of the LLM-assisted verification rules generator based on CommonRoad verification framework [7].

parser and evaluation engine, and existing predicate implementations in Python.

- 2) **LLM-assisted verification rules generator:** Uses structured context (grammar, rules, predicates, XML schema) to generate new rules and predicates that integrate directly into the framework. Outputs are designed for grammar compliance and executability, and are reviewed by humans before integration to ensure correctness.

#### B. Prompt Construction

To generate executable rules and corresponding predicates, we guide the LLM with structured context and strict output constraints. Each prompt includes: (i) context (cleaned inputs; comments removed), (ii) instructional constraints (roles, insertion boundaries), and (iii) a rule specification reflecting domain semantics. This structure addresses *RQ2*, defining strategies and constraints that steer the LLM to produce grammar-compliant rules and executable predicates, with an output contract requiring the LLM to return:

- 1) a grammar-compliant formal rule,
- 2) a Python-implementable function predicate (name and signature),
- 3) a brief semantic explanation.

more details will be introduced in Section IV. This ensures outputs are executable, integrable, and auditable by human reviewers. Potential issues such as syntactically correct but semantically invalid predicates are explicitly handled during the human review stage before integration into the framework.

#### C. Rule and Predicate Synthesis

To extend computational FOL to new semantic domains, the LLM jointly generates symbolic FOL formulas and corresponding executable predicates defined by a name, Python implementation, and semantic description. These predicates are then automatically registered in the framework's handler, replacing manual insertion and ensuring consistent expansion of the function library.

## IV. CASE STUDY AND RESULTS

To evaluate our approach, we applied the proposed pipeline to generate elevation-related verification rules, focusing on a domain not previously addressed by existing rule sets. The full case study examples and code are available online<sup>1</sup>.

**Experiment Setup.** We designed three representative categories of elevation-related defects involving scenarios like bridges and slopes: (1) *Excessive slope*, where road segments exhibit unrealistic grade values; (2) *Abrupt elevation step*, i.e., sudden height discontinuities between consecutive lanelets; and (3) *Insufficient clearance in overlapping segments*, where two roads overlap in the *z*-dimension but the vertical distance between them is below a safe threshold.

We constructed 40 synthetic scenarios with RoadRunner [23]: 30 scenarios contained up to two of the above defects and 10 scenarios served as defect-free baselines. All scenarios were exported in OpenDRIVE and converted into CommonRoad with elevation, covering typical elevation-related issues.

**Prompt-based Rule Generation.** GPT-4o was employed in a prompt-driven configuration to synthesise formal rules and corresponding function predicates tailored to the above three defect types. To ensure integration feasibility, the user query explicitly instructed the model including above request also as follows: “*Do not modify the ANTLR grammar. Output must comply with the existing syntax and directly integrate with the verification pipeline.*”

**Provided Context Artifacts:** The ANTLR grammar definitions, existing predicate implementations, the CommonRoad XML schema and the current rule collection are provided as part of the prompt.

**Generated Outputs.** The pipeline produced formal rules and new predicates aligned with the three defect categories, for example:

- `Is_grade_within_limit(l) || l in L`

<sup>1</sup><https://github.com/ZombiesYard/CRUCIBLE>

These predicates enforce, respectively, reasonable elevation differences, slope bounds, and minimum clearance in overlapping segments.

**Generated Rule Integration.** Each output underwent human review to validate both its logical correctness and its Python implementation. Once approved, the formal rules were added to the rule collection, and the predicates were implemented and registered in the verification system.

**Results.** The effectiveness of the proposed pipeline was evaluated using both qualitative and quantitative criteria.

*Qualitative assessment:* The pipeline eliminates manual grammar analysis, constraint translation, and function implementation (time-consuming and error-prone), yields syntactically correct and semantically consistent rules, and integrates seamlessly (parsed, linked, executed) without post-processing.

*Quantitative assessment:* Across 40 synthetic maps with three mentioned elevation-defect categories, the framework detected all defect types and produced no false positives on defect-free maps, indicating high precision and robustness.

This evaluation directly targets *RQ3*, combining qualitative and quantitative metrics to demonstrate effectiveness in realistic transformation scenarios.

## V. CONCLUSION AND FUTURE WORK

In revisiting our research questions, *RQ1* was addressed by designing the tool architecture, *RQ2* by developing and testing prompting strategies for rule-predicate synthesis, and *RQ3* by evaluating the approach in a controlled case study. These results confirm feasibility and establish a basis for extension.

We presented a semi-automated, human-in-the-loop prototype that leverages LLMs to generate executable verification rules for map transformations, demonstrated with elevation constraints in CommonRoad. The prototype integrates without grammar or framework changes, reduces manual effort, and preserves correctness under human review.

These synthetic experiments provide an initial step but cannot capture the diversity of real-world HD maps. Thus, our evaluation should be viewed as a feasibility study, with future validation on larger and more heterogeneous datasets. Beyond map transformations, the joint generation of formulas and predicates can generalize to domains such as software requirements and cyber-physical system safety. Scaling to broader rule sets and richer logics remains future work.

A key limitation is reliance on GPT-4o, a proprietary model, which raises concerns about reproducibility and semantic errors. We plan to investigate open-source LLMs and automated consistency checks to strengthen robustness and reproducibility.

## REFERENCES

- [1] J. Levinson et al., "Towards fully autonomous driving: Systems and algorithms," 2011 IEEE Intelligent Vehicles Symposium (IV), Baden-Baden, Germany, 2011, pp. 163-168, doi: 10.1109/IVS.2011.5940562.
- [2] A. Armand, J. Ibanez-Guzman, and C. Zinoune, "Digital maps for driving assistance systems and autonomous driving," in *Intelligent Vehicles: Enabling Technologies and Future Developments*, Springer, 2017, doi: 10.1007/978-3-319-31895-0\_9.
- [3] OpenDRIVE Specification, ASAM e.V., 2023.
- [4] M. Althoff, M. Koschi and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 2017, pp. 719-726, doi: 10.1109/IVS.2017.7995802.
- [5] S. Maierhofer, M. Klischat and M. Althoff, "CommonRoad Scenario Designer: An Open-Source Toolbox for Map Conversion and Scenario Creation for Autonomous Vehicles," 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 2021, pp. 3176-3182, doi: 10.1109/ITSC48978.2021.9564885.
- [6] M. Althoff, S. Urban and M. Koschi, "Automatic Conversion of Road Networks from OpenDRIVE to Lanelets," 2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), Singapore, 2018, pp. 157-162, doi: 10.1109/SOLI.2018.8476801.
- [7] S. Maierhofer, Y. Ballnath and M. Althoff, "Map Verification and Repairing Using Formalized Map Specifications," 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), Bilbao, Spain, 2023, pp. 1277-1284, doi: 10.1109/ITSC57777.2023.10422044.
- [8] L. Pan, A. Albalak, X. Wang, and W. Y. Wang, "Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning," arXiv preprint arXiv:2305.12295, 2023, doi: 10.48550/arXiv.2305.12295.
- [9] Zhe Hou, Neural-Symbolic Reasoning: Towards the Integration of Logical Reasoning with Large Language Models. TechRxiv. April 01, 2025. doi: 10.36227/techrxiv.174352068.81462574/v1
- [10] Lam, Long, Shareghi, Ehsan. (2024). A Closer Look at Logical Reasoning with LLMs: The Choice of Tool Matters. 10.48550/arXiv.2406.00284.
- [11] Z. Zhu, Y. Xue, X. Chen, D. Zhou, J. Tang, D. Schuurmans, and H. Dai, "Large language models can learn rules," OpenReview preprint, 2024. [Online]. Available: <https://openreview.net/forum?id=tAmfM1sORP> unpublished
- [12] Clark Barrett, Pascal Fontaine, Cesare Tinelli, "The SMT-LIB Standard: Version 2.6," 2017.
- [13] Chiang, et al. (2022). Verification and validation procedure for high-definition maps in Taiwan. Urban Informatics. 1. 10.1007/s44212-022-00014-0.
- [14] K. -W. Chiang, J. -C. Zeng, M. -L. Tsai, H. Darweesh, P. -X. Chen and C. -K. Wang, "Bending the Curve of HD Maps Production for Autonomous Vehicle Applications in Taiwan," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 15, pp. 8346-8359, 2022, doi: 10.1109/JSTARS.2022.3204306.
- [15] Wild, L., Ericson, L., Valencia, R., Jensfelt, P. (2024). ExelMap: Explainable Element-based HD-Map Change Detection and Update. ArXiv. <https://arxiv.org/abs/2409.10178>
- [16] Chiang, et al. (2023). Establishment of HD Maps Verification and Validation Procedure with OpenDIRVE and Autoware (lanelet2) Format. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. X-1/W1-2023. 621-627. 10.5194/isprs-annals-X-1-W1-2023-621-2023.
- [17] Luo, L., Ju, J., Xiong, B., Li, Y., Haffari, G., Pan, S. (2023). ChatRule: Mining Logical Rules with Large Language Models for Knowledge Graph Reasoning. ArXiv. <https://arxiv.org/abs/2309.01538>
- [18] Manas, K., Zwicklbauer, S., Paschke, A. (2024). TR2MTL: LLM based framework for Metric Temporal Logic Formalization of Traffic Rules. ArXiv. <https://doi.org/10.1109/IV55156.2024.10588650>
- [19] Obi, .et al. (2025). SafePlan: Leveraging Formal Logic and Chain-of-Thought Reasoning for Enhanced Safety in LLM-based Robotic Task Planning. ArXiv. <https://arxiv.org/abs/2503.06892>
- [20] Petrovic, .et al. (2025). GenAI for Automotive Software Development: From Requirements to Wheels. ArXiv. <https://arxiv.org/abs/2507.18223>
- [21] Cosler, M., Hahn, C., Mendoza, D., Schmitt, F., Trippel, C. (2023). NI2spec: Interactively Translating Unstructured Natural Language to Temporal Logics with Large Language Models. ArXiv. <https://arxiv.org/abs/2303.04864>
- [22] Wen, .et al. (2024). Enchanting Program Specification Synthesis by Large Language Models using Static Analysis and Program Verification. ArXiv. <https://arxiv.org/abs/2404.00762>
- [23] MathWorks, RoadRunner: Design 3D Scenes for Automated Driving Simulation. Natick, MA, USA: MathWorks. [Online]. Available: <https://www.mathworks.com/products/roadrunner.html>. Accessed: Aug. 24, 2025.