# A Case Study on Defining Traceable Machine Learning Safety Requirements for an Automotive Perception Component

Sepeedeh Shahbeigi
*Department of Computer Science*
*University of York*
York, UK
Sepeedeh.shahbeigi@york.ac.uk

Richard Hawkins
*Department of Computer Science*
*University of York*
York, UK
Richard.hawkins@york.ac.uk

Simon Burton
*Department of Computer Science*
*University of York*
York, UK
Simon.burton@york.ac.uk

Victoria Hodge
*Department of Computer Science*
*University of York*
York, UK
Victoria.hodge@york.ac.uk

Colin Paterson
*Department of Computer Science*
*University of York*
York, UK
Colin.paterson@york.ac.uk

Ibrahim Habli
*Department of Computer Science*
*University of York*
York, UK
Ibrahim.habli@york.ac.uk

***Abstract*—Integrating machine learning (ML) into safety-critical systems introduces significant safety assurance challenges, particularly as these systems become increasingly autonomous and operate in more open and complex environments. One of the most significant of these challenges is how to systematically specify traceable ML safety requirements. In this paper, we explore the challenges of specifying safety requirements for ML components through a case study of a vehicle Automated Lane Centering function, in which an ML model performs lane detection in a highway scenario. We show how safety concerns propagate from system-level hazards, and explore specific issues that arise in defining meaningful and traceable ML-level requirements, including specifying ML behaviour and robustness. The paper provides the first detailed case study showing how effective and traceable ML safety requirements can be specified for an ML component.**

***Index Terms*—ML Assurance, ML Safety Requirements, Specification, Automotive perception**

## I. Introduction

Getting requirements right is critical in safety-critical systems such as autonomous vehicles, where incomplete or ambiguous specifications can lead to hazardous behaviour [1]. While not sufficient alone, requirements engineering (RE) supports the foundation of through-life safety assurance.

The integration of machine learning (ML) complicates this process. ML components behave probabilistically and depend on data, introducing epistemic uncertainty and challenging conventional specification techniques. This raises a key question: how can traceable safety requirements be defined for ML components embedded within complex autonomous systems?

Existing frameworks such as AMLAS [2] acknowledge this need but provide only high-level guidance without prescribing how to specify such requirements in practice. Most approaches either treat ML as a black box or assume requirements are externally defined.

In this paper, we demonstrate, through a detailed case study, how safety requirements for an ML component can be specified and traced from system-level requirements. Using an Automated Lane Centering (ALC) function, where an ML model performs lane detection on highways, we show how functional safety concerns can be decomposed and linked across abstraction levels. Furthermore, we provide preliminary guidance on the structure and formulation of safety requirements allocated to ML components, and outline how these can be traceably decomposed to requirements at the ML implementation level.

Our work complements AMLAS and previous efforts on partial specification [3], robustness [4], and traceability [5] by demonstrating how these aspects can be addressed in practice. Section II details the case study; Section III discusses next steps, including future work to formalise this into a process.

## II. The Case Study

We illustrate our approach using a case study of a vehicle equipped with a camera and an ALC function operating on a US highway. The case study demonstrates how our structured syntax enables traceable specification of safety requirements from system-level objectives down to ML components. Owing to space constraints, we focus on the practical challenges of developing traceable requirements and the discussions needed to refine them from higher levels. Full technical derivations and the development of the general methodology are deferred to follow-up work.

In this case study, we assume the ego vehicle is a standard passenger car ($1.8\,\mathrm{m}\times4.5\,\mathrm{m}$) on $3.7\,\mathrm{m}$-wide lanes, travelling at constant $v = 31\,\mathrm{m/s}$ with $a_{\max} = 3\,\mathrm{m/s^2}$; overtaking and other manoeuvres are out of scope. The ALC architecture (Fig. 1) comprises camera perception (ML lane-marking

detection), lane tracking estimating $Y_{wv}$ (lateral offset to the lane centre), and a steering controller; traceability points are annotated $[J\#]$ and cross-referenced in Table I.

### A. Operational Design Domain

Safety requirements must be specified with respect to the system's intended operational context, formalised as the Operational Design Domain (ODD). The ODD defines environmental attributes such as scenery, weather, and dynamic elements that influence system behaviour [6]. We represent the ODD as a structured, hierarchical taxonomy to support traceable requirement derivation. Each attribute is annotated with metadata indicating whether it is fixed (e.g., regulatory constraints), variable (e.g., weather), or relevant to ML behaviour, each of which plays a role in subsequent analysis.

To manage complexity, we define Partial Operational Domains (PODs) as subsets of the ODD that describe specific operational scenarios. PODs enable incremental and context-specific SR development and provide a foundation for linking requirements to concrete operational conditions. Fig. 3 illustrates examples of the ODD taxonomy. Specification of PODs are beyond the scope of this paper and is elaborated in future work.

### B. Limitations of Traditional Safety Requirements for ML Components

To highlight the limitations of conventional SRs for ML, we begin with a standard example using the *Easy Approach to Requirements Syntax* (EARS) [7], which structures SRs via conditions, responses, and constraints. A typical SR for lane detection might be:

> **Allocated SR (EARS):** When lane markings are visible, the lane detector SHALL output the detected lane positions to the lane tracker.

While EARS improves clarity, it is insufficient for ML components, which are probabilistic, data-driven, and context-sensitive. This introduces dimensions such as performance, robustness, and dataset dependence that traditional templates cannot capture. To be meaningful, such requirements must also state their underlying assumptions, which themselves become additional requirements for rigorous ML assurance.

To address this, we introduce a structured syntax incorporating ML-specific elements, detailed later in the paper. In this work, our focus is on ML component and model-level requirements, which demonstrate SR traceability from system-level to ML.

### C. System-Level and Perception Safety Requirements

To support traceability to the ML component, we briefly summarise the relevant higher-level safety requirements from the system and perception layers that adapt to our proposed MLC- and ML-level SRs. Due to space constraints, full details are deferred to a longer version of this work. The top-level system SR is to prevent unintended lane departure or collision hazard. This is formalised as:

**S-SR001**: The system SHALL ensure that the ego vehicle's bounding box remains within the [safe drivable area] on a motorway, under all conditions defined in the ODD, with a failure probability of less than $10^{-8}$ per hour.

From **S-SR001**, the following requirements are allocated to the perception component. The first specifies limits on $Y_{wv}$; the second addresses the extrapolation of lane markings from previous detections when current observations are unavailable. This is a common approach in tracking systems.

**P-SR001**: The maximum lateral displacement error $Y_{wv}$ SHALL not exceed 0.66 m for more than 143 consecutive frames, with a failure probability of $4.7 \times 10^{-9}$ per hour under all ODD conditions.

**P-SR002**: Extrapolation SHALL not persist for more than 0.94 seconds.

The $Y_{wv}$ value is estimated from detected lane positions, lane width, vehicle dimensions, and road curvature. Risk budgets follow [8], with failure probabilities allocated using data from [9].

### D. ML Component Safety Requirement Translation

The perception function is decomposed into three subcomponents. Here we focus only on the ML-based Lane detection component shown in figure 1. Before specifying ML model requirements, we distinguish between ML component safety requirements (MLC-SRs) and ML model safety requirements (ML-SRs). In our architecture, the lane detection module is fully implemented as the ML component, so MLC-SRs represent its allocated functional safety requirements, while ML-SRs define the model-level behaviours needed to fulfil them. We express MLC-SRs using the following structured syntax:

**MLC-SR#:** [{context}] [condition] [metric] SHALL [relation] [acceptance criteria] [input number] [probability of failure] [confidence]

This format captures ML-specific properties such as probabilistic failure, metric, based constraints, and temporal bounds—that are not expressible using traditional templates like EARS [7]. Key elements include:

- **Metric:** A measurable quantity linked to ML output (e.g., lateral displacement).
- **Acceptance Criteria:** A quantitative threshold on the metric.
- **Input Number:** The window over which performance is evaluated (e.g., per frame or sequence).
- **Probability of Failure:** A bound on the likelihood of requirement violation.
- **confidence:** When the condition must hold (e.g., always, or $\geq$90% of the time). This reflects the uncertainty on the requirement (including the probability of failure). However, it is not the focus of this work and will be elaborated in a follow-up work.
- **{context}:** A subdomain of the system's ODD which defines the scope of validity of the requirement.

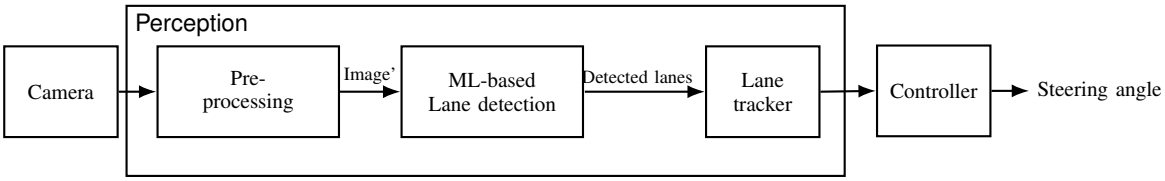


Fig. 1: Functional architecture of the ALC function.

Our syntax extends the FRET [10] template, designed for deterministic components, by explicitly incorporating probabilistic and data-dependent elements required for ML assurance. Owing to space constraints, we concentrate on the essential elements of our requirement formulation. Full detail, comparison with FRET, and examples are provided in our follow-up paper. This syntax supports traceable, quantitative specification of ML behaviours beyond the scope of EARS and FRET. If we apply this formulation to our example, assuming context, we can define the following **MLC-SR**:

**MLC-SR001**: In the {context} and the event of the correct detection, *(Context + Condition)* the sum of left and right errors of the detected lanes *(Metric)* SHALL not exceed 0.66 m *(Acceptance Criteria)* for more than 143 consecutive frames, *(Input range)* with failure probability of $6 \times 10^{-10}$. *(Probability of failure)*

**MLC-SR001** directly reflects the tolerance in lateral displacement defined in **P-SR001**, accounting for both left and right lane boundaries. The 0.66 m bound is derived from the safe drivable area calculation. The failure probability is based on statistical analysis of component-level failures in autonomous systems [9]. The MLC-SR explicitly encodes measurable safety constraints and their statistical context which are essential for developing and assuring ML models.

**MLC-SR001** addresses the behaviour of the ML component in terms of true positives (TPs) **[J1]**; However, to comprehensively capture its failure modes, additional considerations are required. In particular, SRs must account for the impact of false positives (FPs), false negatives (FNs), and the compounding effect of consecutive misclassifications over time.

This need arises from **P-SR001**, which constrains the allowable lateral displacement error $Y_{wv}$. The value of $Y_{wv}$ is computed from the positions of the lane markings detected by the ML-based lane detection component. The details of this estimation can be found in [11]. Consequently, its accuracy is highly sensitive to the correctness of the ML outputs. Any false negative or false positive directly skews the inferred lane boundaries and, in turn, the estimated lateral position of the vehicle. For example, consistently missing one lane side or detecting artefacts as valid markings can displace the computed lane centre, leading to errors in $Y_{wv}$ estimate. Therefore, the MLC-SRs defined here establish explicit constraints on detection accuracy, false detection rates, and error persistence, ensuring that the ML component operates within the tolerances required to maintain compliance with **P-SR001**.

These failure modes were previously identified during hazard analysis and system decomposition, and are already reflected in **P-SR001**. The MLC-SRs that follow refine those failure modes to specify the conditions under which FPs and FNs must be tolerated or constrained at the ML component level.

To ensure completeness, these failure modes should be refined in collaboration with domain experts, who can provide contextual insight into their operational implications. For instance, a missed lane marking (FN) or the incorrect detection of a non-lane feature as a lane (FP) may lead to significantly different hazards depending on road context, traffic density, and control strategies.

In transport systems, such as autonomous driving, isolated failures may be tolerable, but sustained misclassifications over an interval of inputs can result in hazardous outcomes. Therefore, the derived **MLC-SRs** must reflect not only individual failure events but also their temporal persistence and combinations **[J2]**. The following requirements explicitly address FP, FN, and their compounding effects:

**MLC-SR002**: If lane markings of lengths 0.3 m or longer *(Condition)* are not present, *(Metric (binary))* they SHALL not be detected *(Acceptance Criteria)* (false positive).

**MLC-SR003**: If a lane marking of lengths 0.3 m or longer *(Condition)* is present, it *(Metric (binary))* SHALL be detected *(Acceptance Criteria)* (false negative).

**MLC-SR004**: True and false negative frames *(Metric)* SHALL not continue for *(Acceptance Criteria)* more than 94 frames. *(Input range)*

The 30 cm threshold in MLC-SR002 is chosen to exclude artefacts such as road debris or worn paint fragments that could otherwise be misidentified as lanes. This value corresponds to the minimum length of a standard intermittent lane marking and will be subject to refinement based on empirical testing **[J3]**.

MLC-SR003 ensures safety-critical lane markings are reliably detected. MLC-SR004 addresses compound failure modes, where prolonged FN or TN behaviour may propagate downstream to the control system, leading to hazardous deviations **[J4]**.

A key challenge in the safety analysis of ML components is that identical failure modes may result in significantly different hazard severities depending on the operational context. For example, missing a pedestrian walking on a sidewalk might

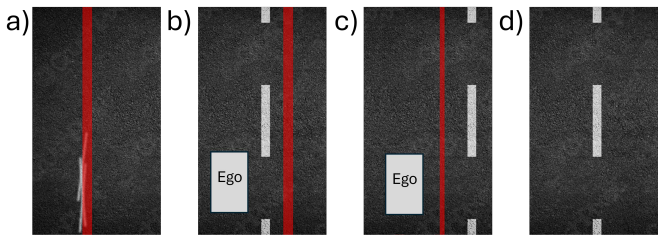


Fig. 2: False outputs of the ML component: a) FP1: the lane marking is detected when it does not exist, b) FP2: the lane marking is detected when it does not exist-further, c) FP3: the lane marking is detected when it does not exist-closer, and d) FN: the lane marking is not detected when it exist

pose negligible risk to the ego vehicle, whereas failing to detect a pedestrian on the vehicle's trajectory could result in a catastrophic outcome. Therefore, hazard analysis must account not only for the type of ML failure but also for its context-specific consequences. Such weighting informs the prioritisation of failure modes in the ML safety specification process outlined in the next subsection.

Figure 2 illustrates typical error modes in ML-based lane detection. FP1 corresponds to spurious/non-existent lane detection, while FP2 and FP3 involve the misclassification of non-lane features. FN denotes a missed detection where a lane marking is present. Among these, FP1, FP3, and FN are the most safety-critical, as they may lead to lateral deviation or unintended road departure. Hence, while SRs distinguish between false positives (FPs) and false negatives (FNs), their weighted hazard potential should guide both performance requirement thresholds and the focus of validation efforts.

In this section, we have demonstrated that the conventional approach to formalising safety requirements for system components must be adapted when applied to components incorporating ML. In the next section, we demonstrate key considerations and challenges in decomposing these MLC-SRs into ML-specific safety requirements.

### *E. ML safety Requirements*

In the previous subsection, we identified **MLC-SR**s for context. The **MLC-SR**s specify the safety requirements on the component, but are not yet presented in a form that can be used by an ML developer to create and verify an ML model. To do this requires that the **MLC-SR**s be translated into ML-specific requirements, while maintaining traceability to the component safety requirement.

ML safety requirements encompass two distinct aspects: *performance* and *robustness*. Performance SRs specify how well the ML model must behave under expected conditions, while robustness SRs ensure the model remains safe under operational variability across the ODD. In this subsection, we focus exclusively on deriving the performance-related SRs for an example POD, $\text{POD}_0$. The next subsection elaborated on robustness SRs.

A direct decomposition of the **MLC-SR**s involves specifying both classification and localisation accuracy requirements. Classification accuracy relates to the ML model's ability to correctly classify true positives (TP) and true negatives (TN). localisation accuracy refers to the spatial alignment between TP detections and their corresponding ground truth. localisation accuracy ensures that localisation error remains within an acceptable bound. We begin by deriving requirements related to classification accuracy and associated detection metrics, before addressing localisation accuracy at the end of this subsection, as it follows more directly from prior constraints.

classification accuracy is sensitive to dataset imbalance, which may obscure model limitations if not properly accounted for. While we assume the dataset has been collected following rigorous safety requirements (**SRs**), we acknowledge that achieving perfect balance is impractical in real-world perception tasks. The effect of these imperfections are reflected in the [confidence] element of the requirement syntax and are studied in a follow-up research.

Building on **MLC-SR001**, we assume that safety-relevant failures may result from misclassifications, false positives (FPs) or false negatives (FNs), sustained across up to 143 consecutive frames. This motivates the following requirement:

- **ML-SR001:** The classification accuracy of lane detection SHALL not be less than 0.889.

This threshold reflects the need to tolerate occasional misclassifications while preventing prolonged failures that could lead to a lane departure. In the worst case, systematic failures—triggered by specific inputs, blind spots, or dataset gaps can persist with certainty ($P_{fail} = 1$) until mitigated. Such failures cannot be addressed through probabilistic guarantees alone and must be eliminated during development. Any reproducible failure observed in validation invalidates prior assumptions and requires re-executing the safety process.

To bound risk from semi-systematic failures, which may persist due to temporal correlation in input data, we model error propagation using a conservative exponential growth model. Assuming an initial failure rate $P_{\text{fail}}$ and a temporal amplification factor $k = 1.05$, the cumulative failure probability over 143 frames must remain below $6 \times 10^{-10}$. Solving this yields an upper bound of $P_{\text{fail}} < 0.027$, corresponding to a required accuracy of at least 0.973 in quasi-systematic conditions. In practice, this is balanced against architectural mitigations and offline validation results, leading to the selected threshold of 0.889 as a practical lower bound that supports the system's safety target.

This formulation makes traceability from MLC-SR001 to ML-SR001 explicit and highlights how probabilistic accuracy targets are grounded in the safety assumptions and temporal risk models of the system.

To mitigate dataset bias and support robust validation of accuracy-related SRs, we adopt complementary performance metrics rather than relying on single detection thresholds. Notice that FP and FN are defined using **MLC-SR002** and **MLC-SR003 [J5]**.

TABLE I: Traceability Between Allocated, MLC and ML-Level Safety Requirements. [J#] points to the paragraphs within the subsections that include the justifications.

| System-level SR | Perception SR | MLC-SR | Justification (SR→MLC-SR) | ML-SR | Justification (MLC-SR→ML-SR) |
|---|---|---|---|---|---|
| S-SR001 | P-SR001<br>P-SR002 | MLC-SR001 | II-D: **[J1]** | ML-SR001 | II-E: **[J1]** |
| | | | | ML-SR002 | II-E: **[J6]** |
| | | | | ML-SR003 | II-E: **[J6]** |
| | | | | ML-SR004 | II-E: **[J7]** |
| | | MLC-SR002 | II-D: **[J3]** | ML-SR002 | II-E: **[J5]** |
| | | MLC-SR003 | II-D: **[J4]** | ML-SR003 | |
| | | MLC-SR004 | II-D: **[J2]** | ML-SR002 | II-E: **[J6]** |
| | | | | ML-SR003 | II-E: **[J6]** |
| | | | | ML-SR004 | II-E: **[J7]** |

The per-frame failure probability is:

$$P_{\text{fail}} = \frac{FP + FN}{TP + FP + FN + TN} = 1 - \text{PAccuracy}$$

To satisfy **ML-SR001**, the model's accuracy must exceed 0.889.

**MLC-SR004** constrains undetected frames (FN + TN) to fewer than 94 consecutive inputs, giving $P_{\text{FN+TN}} \leq 0.798$. **MLC-SR001** limits total detection errors to $P_{\text{FN+FP}} \leq 0.027$. Exploring valid FN, FP, and TN combinations under these constraints, we compute the resulting bounds:

- **ML-SR002:** Recall SHALL be greater than 0.910.
- **ML-SR003:** Precision SHALL be greater than 0.995.

These thresholds ensure safe performance across both omission and commission errors, supporting traceability from MLC-SR001 and MLC-SR004 to measurable ML model behaviour **[J6]**.

We define the localisation accuracy requirement derived from **MLC-SR001**, which limits localisation error in true positive (TP) lane detections **[J7]**. We use mean squared error (MSE) to quantify this, as it penalises large deviations relevant to lateral safety.

- **ML-SR004:** The average MSE of TP lane detections (left and right, per image) SHALL be less than 0.66 m.

This threshold aligns with the lateral error bound in **P-SR001**, maintaining consistency across the requirement chain.

Table I summarises the ML safety requirements derived in our case study, showing traceability from the allocated system-level requirement (SR-001) to ML component (MLC-SRs) and model-level (ML-SRs) requirements. Each link is supported by justification references [J#], pointing to the relevant rationale. This structured decomposition ensures that all ML-level requirements are grounded in system safety. Future work will formalise these justifications into reusable argument patterns for integration into safety cases.

### *F. Robustness Safety Requirements Through POD-Based Decomposition*

This subsection extends the previously defined **ML-SR**s, developed from the allocated SRs which are referred to as performance SRs, by introducing robustness **ML-SR**s. We define robustness as the ability of the model to behave safely in variations of the OD defined by the attributes in the ODD. Our approach takes a different path to AMLAS [12] where performance and robustness SRs are suggested to be developed separately. We define the ML SRs as a set of performance SRs contextualised in associated range of PODs (robustness). The union of these PODs spans the entire ODD. The purpose of this decomposition is to ensure that the ML component continues to satisfy its performance-related SRs as critical operational attributes vary within their defined bounds. By incrementally introducing variability across PODs, the robustness requirements capture the model's ability to generalise across the ODD, and additionally are helpful to expose potentially undiscovered systematic failures. Ongoing research focuses on formalising the integration of PODs with requirements specification to enable scalable and rigorous assurance.

We have shown how to specify a set of ML-SRs which we denote by $\textbf{ML-SRs}_1$ for $\text{POD}_0$. However, to ensure traceability we have to ensure that the whole context is covered. Therefore, we propose decomposing context in a systematic manner into $[\text{POD}_j, \ldots, \text{POD}_k]$, which captures all the attributes under which the ML component is expected to operate safely. Defining PODs requires careful coordination between ML engineers, safety experts, and domain specialists to identify relevant attributes and appropriate granularity [13]. For example, road surface quality affects lane detection and warrants inclusion, while ambient temperature may not. Excluding irrelevant factors prevents unnecessary combinatorial growth, streamlines evaluation, and provides a principled stopping point for POD refinement. These eliminated attributes should be documented and tagged in the ODD for the completeness coverage checks explained in the following paragraphs.

To manage complexity and support traceability, we represent PODs as a tree, where each node corresponds to an operational attribute, and distinct PODs differ by at least one such attribute. Starting from $\text{POD}_0$, new $\text{POD}_i$ instances are formed by incrementally varying attributes that affect ML behaviour. Nodes are annotated with metadata (e.g., fixed/variable, tree depth, included/excluded), enabling consistent terminology,

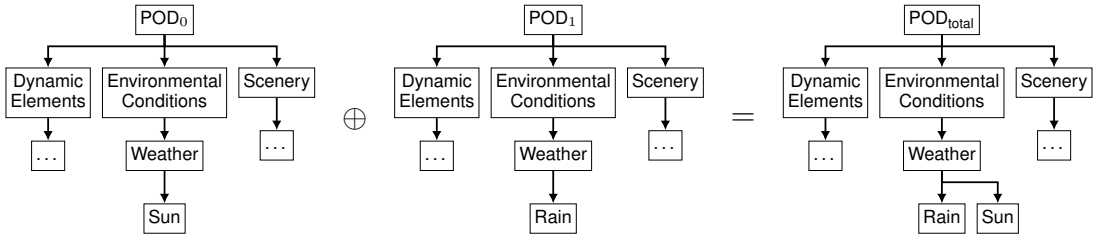


Fig. 3: Illustration of PODs merging operation: $\text{POD}_0 \oplus \text{POD}_1 =$ **POD**. Each POD corresponds to a partial operational context; their union incrementally covers full ODD.

completeness checks, and automated POD merging during iterative refinement. Fig. 3 illustrates a sample merge process.

PODs are introduced incrementally and merged through a summation process to construct a global POD structure denoted by **POD** in Fig. 3. This process continues until the union of all defined PODs fully covers the context that is relevant to ML performance. As new PODs are introduced, previously unrecognised operational attributes may be identified as relevant to ML behaviour. When this occurs, the POD structure and corresponding ML robustness SRs must be updated accordingly. Attributes that are part of the ODD but deemed not relevant to ML should still be explicitly tagged and documented, to ensure traceability and completeness when comparing the POD union to the full ODD specification. Fig. 3 illustrates this iterative expansion and merging process. In this way, we define a set of ML performance SRs (**ML-SRs**) and associate each with one or more PODs, resulting in a structured requirement set of the form:

$$\{(\textbf{ML-SRs}_i, \{\text{POD}_j, \ldots, \text{POD}_k\})\}_{i=1}^{N}$$

Here, we decomposed the MLC-SRs into a structured set of ML-SRs that explicitly account for model behaviour across the full ODD. By deriving quantitative bounds on classification accuracy, recall, and precision from system-level hazard constraints, we ensured that each ML-SR is measurable and traceable to higher-level safety objectives. Additionally, we included a localisation accuracy requirement for true positives to address the quality, not just quantity, of correct detections.

This formulation captures robustness via POD coverage, supports systematic derivation of dataset requirements (e.g., completeness, relevance, and balance as defined in AMLAS), and lays the foundation for ML assurance within the safety case. Future work will extend this to derive dataset-level requirements from these ML-SRs.

## III. Conclusion and Future Work

This work presented a case-study–driven methodology for specifying **ML-SRs** in an autonomous driving systems application, combining performance and robustness through PODs to enhance ODD-wide traceability beyond AMLAS. Future work will extend the approach to dataset-level requirements and formalise it into an application-agnostic framework for scalable ML assurance.


## Acknowledgment

This work was supported by the Centre for Assuring Autonomy, a partnership between Lloyd's Register Foundation and the University of York (https://www.york.ac.uk/assuring-autonomy/)


## References


[1] Z. Pei, L. Liu, C. Wang, and J. Wang, "Requirements engineering for machine learning: A review and reflection," in *2022 IEEE 30th International Requirements Engineering Conference Workshops (REW)*, pp. 166–175, IEEE, 2022.

[2] R. Hawkins, C. Paterson, C. Picardi, Y. Jia, R. Calinescu, and I. Habli, "Guidance on the assurance of machine learning in autonomous systems (amlas)," *arXiv preprint arXiv:2102.01564*, 2021.

[3] R. Salay and K. Czarnecki, "Improving ml safety with partial specifications," in *Computer Safety, Reliability, and Security: SAFECOMP 2019 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE, Turku, Finland, September 10, 2019, Proceedings 38*, pp. 288–300, Springer, 2019.

[4] H. Kuwajima, H. Yasuoka, and T. Nakae, "Engineering problems in machine learning systems," *Machine Learning*, vol. 109, no. 5, pp. 1103–1126, 2020.

[5] J. H. Husen, H. Washizaki, H. T. Tun, N. Yoshioka, Y. Fukazawa, and H. Takeuchi, "Traceable business-to-safety analysis framework for safety-critical machine learning systems," in *Proceedings of the 1st International Conference on AI Engineering: Software Engineering for AI*, pp. 50–51, 2022.

[6] British Standards Institution, "Operational design domain (odd)," 2025. Publicly Available Specification.

[7] A. Mavin, P. Wilkinson, A. Harwood, and M. Novak, "Easy approach to requirements syntax (ears)," in *2009 17th IEEE international requirements engineering conference*, pp. 317–322, IEEE, 2009.

[8] S. Shalev-Shwartz, "On a formal model of safe and scalable self-driving cars," *arXiv preprint arXiv:1708.06374*, 2017.

[9] A. Richter, T. P. Walz, M. Dhanani, I. Häring, G. Vogelbacher, F. Höflinger, J. Finger, and A. Stolz, "Components and their failure rates in autonomous driving," in *Proceeding of the 33rd European Safety and Reliability Conference*, pp. 233–240, 2023.

[10] D. Giannakopoulou, A. Mavridou, J. Rhein, T. Pressburger, J. Schumann, and N. Shi, "Formal requirements elicitation with fret," in *International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ-2020)*, no. ARC-E-DAA-TN77785, 2020.

[11] M. Tian, F. Liu, and Z. Hu, "Single camera 3d lane detection and tracking based on ekf for urban intelligent vehicle," in *2006 IEEE International conference on vehicular electronics and safety*, pp. 413–418, IEEE, 2006.

[12] R. Hawkins, M. Osborne, M. Parsons, M. Nicholson, J. McDermid, and I. Habli, "Guidance on the safety assurance of autonomous systems in complex environments (sace)," *arXiv preprint arXiv:2208.00853*, 2022.

[13] S. Shahbeigi, N. M. Proma, V. Hodge, R. Hawkins, B. Li, and V. Donzella, "Robustness requirement coverage using a situation coverage approach for vision-based ai systems," *arXiv preprint arXiv:2507.12986*, 2025.