WHITEPAPER

Ensuring Trustworthiness of AI-Enhanced Embedded Systems

Illustrated by AI-Based Motor Control

Daniel Scholz (Infineon Technologies)
Senior Engineer
Jürgen Schäfer (Infineon Technologies)
Distinguished Engineer
Shanza Ali Zafar (Fraunhofer IKS)
Senior Researcher
Dr.-Ing. Núria Mata (Fraunhofer IKS)
Department Head

www.infineon.com
www.iks.fraunhofer.de

v1.0
10/2025

Table of Contents

Abstract

Artificial Intelligence (AI) is unlocking new capabilities in safety-critical systems, from enhanced motor control to autonomous driving. However, integrating AI safely remains a significant challenge due to its data-driven nature and operation in open and real-world variability. While established standards such as ISO 26262 and ISO 21448 provide a foundation for functional safety and intended functionality, they do not fully address AI-specific properties like robustness, resilience, and transparency. Emerging standards such as ISO/PAS 8800 begin to close this gap by introducing AI-specific safety lifecycles and properties.

This white paper presents a unified safety lifecycle that integrates these standards. Using AI-Based Motor Control Unit (AI-MC) as an illustrative use case, it shows how robustness and resilience can be systematically realized through concrete development-time measures and operational-time measures (e.g., Out-of-Distribution). Together, these enable safe and reliable deployment of AI-based systems on real-time high integrity platforms. The recommendations presented aim to guide practitioners in systematically integrating AI into safety-critical systems without compromising safety, availability, or trustworthiness.

1 Introduction

AI is rapidly becoming a driving force for innovation across safety-critical domains, from battery management systems and motor control to autonomous vehicles. These systems rely on AI's ability to make intelligent, data-driven decisions, unlocking unprecedented efficiency, performance, and user experience. However, the very capabilities that empower these systems also introduce significant challenges in ensuring that the systems are trustworthy in all circumstances. Trustworthiness encompasses several key dimensions: safety, security, reliability, privacy, transparency, and resilience [4]. These elements are tightly linked; for example, a lack of transparency can undermine reliability, just as a security breach can directly compromise safety. In safety-critical systems, where failure can result in severe or catastrophic consequences, building AI that is not only accurate but also demonstrably trustworthy is essential [3, 13, 25].

> *In safety-critical AI, trust isn't a feature — it's a requirement*

Unlike traditional rule-based systems, AI solutions that rely on machine learning models learn their behaviour from data. This data-driven nature makes it inherently more difficult to predict and validate system behavior in previously unseen scenarios, introducing unique risks in safety-critical contexts. Addressing AI safety thus demands approaches that extend beyond conventional safety engineering. While established standards such as ISO 26262 (functional safety) [7] and ISO 21448 (safety of the intended functionality, SOTIF) [8] lay the foundation for system reliability, they do not fully encompass the data-driven characteristics of AI systems.

Emerging standards, notably ISO PAS 8800 [19], begin to bridge this gap by introducing an AI-specific safety lifecycle and outlining key safety properties, such as robustness, resilience, transparency, and explainability, that AI systems should achieve for safety-critical domains.

> *AI Safety $\neq$ System Accuracy: Safety depends on how a system handles the unknown, not just how it performs on known data.*

Among these, robustness and resilience play a central role in safety-critical environments. Robustness ensures that AI systems maintain acceptable performance despite noisy inputs, environmental variability, and operational disturbances, whereas resilience allows the AI system to degrade gracefully. However, robustness or resilience alone cannot ensure the overall safety of an AI system. Other essential properties, such as transparency, alignment, and explainability, should also be integrated into the overall safety framework and addressed from design to deployment and operation.

This white paper presents a road-vehicle standard-based unified AI safety lifecycle and provides a blueprint for integrating both robustness and resilience into AI systems deployed in safety-critical domains. Although the discussion spans multiple application areas, AI-MC is employed as an illustrative use case to demonstrate how these principles can be systematically achieved through both development-time and operational-time measures. The paper aims to:

- Explore emerging safety standards, such as ISO PAS 8800, and illustrate how these, alongside established standards (ISO 26262 and ISO 21448), form a unified AI safety lifecycle.
- Recommend robustness and resilience as critical pillars within the broader AI safety framework.
- Demonstrate the practical implementation through real-time capable AI-MC on ASIL D compliant AURIX as a use case.

The following section provides an overview of this use case, which will serve as a running example throughout the paper.

2 Use Case: AI-based Motor Control in Safety-Critical Systems

Electric motor control is a fundamental function in automotive applications, where precision and reliability are paramount. The AI-MC system discussed in this report is a control unit designed to manage electric motors by performing closed-loop control of motor speed and torque. It achieves this by dynamically regulating power delivery from the vehicle's battery based on torque or speed commands received from the Vehicle Control Unit (VCU).

A key aspect of motor control is accurately estimating the rotor's position and speed to enable Field Oriented Control (FOC). Traditionally, these measurements are provided by physical sensors. However, to reduce cost, complexity, and space requirements, the AI-MC system adopts a sensorless approach. Instead of relying on dedicated hardware sensors, it employs a Neural Network (NN) model to predict the rotor position in real-time for higher speeds.

Technically, the system processes motor currents and voltages to derive meaningful information about the motor's state. Using techniques like the Clarke transformation, the three-phase currents and voltages $(i_a, v_a, i_b, v_b, i_c, v_c)$ are converted into a two-dimensional representation $(i_\alpha, v_\alpha, i_\beta, v_\beta)$. This transformation captures the dynamics of the rotating magnetic field, which is critical for FOC. The resulting data serves as input to the NN model that estimates the sine and cosine of the rotor's position angle, providing precise feedback necessary for optimal motor control.

Figure 1 illustrates the overall architecture of the AI-MC system, showing how the traditional control components are integrated with the AI-based position sensor. This blend of conventional motor control techniques with advanced AI-driven estimation not only simplifies the system design but also enhances performance and predictive maintenance capabilities.

The current implementation was evaluated on a 6000 Rotations per Minute (RPM) test motor to validate the core AI-

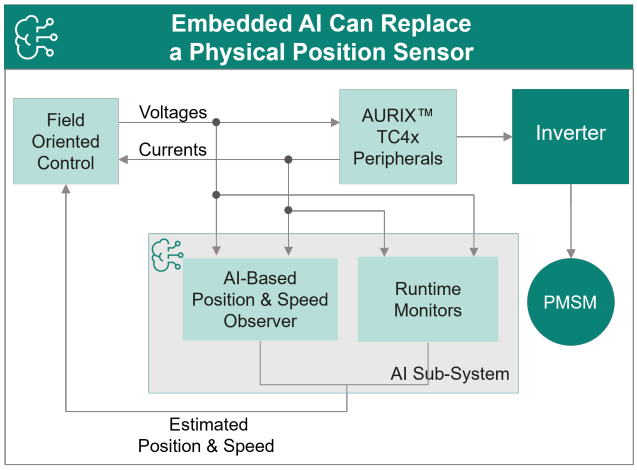


Figure 1: Overview of overall Motor Control System (AI-MC) including AI subsystem.

based position estimation approach in a controlled setting. Although this does not represent full automotive-scale systems, the same methodology can be extended to higher-performance applications with appropriate retraining and validation.

By combining established control strategies with innovative AI techniques, the AI-MC system offers a promising solution for enhancing the reliability, efficiency, and safety of electric motor control in automotive applications.

3 Towards a Unified Safety Lifecycle for AI-Based Systems

Industry safety standards are foundational to the engineering of safety-critical systems. They provide structured processes to manage hazards, guide verification, and demonstrate assurance. ISO 26262 [7] defines processes for managing functional safety related to hardware faults and software errors. However, it does not fully cover risks arising from system functional limitations or the particular characteristics of AI components. ISO 21448 [8] introduces Safety of the Intended Functionality (Safety of the Intended Functionality (SOTIF)), addressing hazards from functional insufficiencies and performance limitations such as sensor errors or environmental factors, areas outside the scope of ISO 26262 but relevant to system safety. ISO/PAS 8800 [19] complements these by focusing explicitly on AI-specific safety challenges, including robustness, resilience, transparency, and data-related risks inherent in AI systems. Other related standards and guidelines are also emerging to address AI system safety-related challenges. These include ISO/IEC TR 24028:2020 [17] for trustworthiness and risk management of AI systems, ISO/IEC 24029-1:2021 [18] for AI robustness verification, IEEE 7001-2021 [14] for transparency in autonomous systems, and UL 4600 [21] which specifies safety principles for autonomous products including AI-based systems.

Although the analyzed road-vehicle safety standards differ in scope, maturity, and methodological emphasis, they offer complementary safety perspectives. But they remain largely isolated, each valuable in its own domain, yet lacking a common framework that integrates them across the AI product lifecycle. This section introduces such a unified lifecycle, using the traditional V model from ISO 26262 as its backbone, combined with ISO 21448 and ISO PAS 8800. It tackles the critical challenge: How to integrate these standards into a seamless, lifecycle-spanning framework that captures the unique risks, hazards, and safety considerations of AI, from concept to deployment and operation.

3.1 Safety Standards for AI-based System in Automotive Domain

ISO 26262 [7], ISO 21448 [8], and ISO PAS 8800 [19] each address a different slice of the safety challenge. Viewed together, they form a more complete picture of what it means to design, develop, and assure a safe AI-based automotive system. Below is a brief description of each of the considered safety standards.

3.1.1 ISO 26262: Road Vehicles - Functional Safety

ISO 26262 [7] is the foundational standard for ensuring functional safety in automotive systems. It addresses risks arising from system malfunctions caused by both random hardware faults and systematic software/hardware failures, focusing primarily on traditional software and hardware systems rather than AI or ML components. At its core, ISO 26262 introduces a structured safety lifecycle that spans hazard identification, risk assessment, derivation of safety goals, and the implementation of risk reduction measures tailored to the severity of potential failures. A key part of the standard is the assignment of Automotive Safety Integrity Levels (ASILs), A through D, which reflect the level of risk associated. Each ASIL dictates the rigor of required development processes and safety mechanisms. For example, higher ASILs demand stronger requirements for hardware fault tolerance, including quantitative thresholds like failure rate limits and fault coverage metrics. For software, ISO 26262 does not mandate specific failure rate targets but instead prescribes development and verification methods, such as coding guidelines, static analysis, and fault injection, based on the ASIL classification. Though not designed for AI-based systems, ISO 26262 remains relevant in AI contexts. It governs the underlying hardware and software infrastructure that supports AI models (e.g., inference platforms or preprocessing components). However, the standard does not address the unique failure modes of AI models themselves such as failures due to inadequate data.

3.1.2 ISO 21448: Road Vehicles - Safety of the Intended Functionality

ISO 21448 [8] focuses on ensuring that road vehicles perform safely even when no system malfunctions are present. It addresses hazards that may arise from functional insufficiencies or performance limitations, particularly in advanced driver assistance systems (ADAS) and autonomous driving systems. A key concept introduced by SOTIF is the identification and mitigation of "triggering conditions", i.e., specific scenarios or environmental factors that could expose

these functional insufficiencies, potentially leading to hazardous situations. The standard emphasizes a scenario-based framework to systematically identify, evaluate, and mitigate such risks, ensuring that the system's intended functionality remains safe across its Operating Design Domain (ODD).

3.1.3 ISO/PAS 8800: Road Vehicles - Safety and Artificial Intelligence

ISO/PAS 8800 [19] bridges critical gaps left by ISO 26262 and SOTIF by addressing the unique safety challenges posed by AI components in road vehicles. It introduces a conceptual model that connects AI development artifacts, such as training data, model architecture, and evaluation results, to the overall system safety argument. Central to this model are safety-related properties that must be addressed in every stage of the AI safety lifecycle, from AI design to deployment, ensuring that any potential risks or insufficiencies during development and operation are identified and mitigated. Additionally, ISO PAS 8800 provides comprehensive guidance on the dataset lifecycle, verification and validation of AI systems, operational safety measures, and the development of assurance arguments tailored to AI-based systems.

*ISO PAS 8800 defines essential **AI safety properties** like robustness, resilience, and more.*

3.2 Unifying the Standards in a Lifecycle

This section delivers on one of the core contributions outlined in the introduction: proposing a unified safety lifecycle that integrates ISO 26262, ISO 21448, and ISO PAS 8800 into a cohesive framework for AI-based systems in safety-critical domains.

Despite their complementary nature, there is limited guidance on the integration of these standards, leaving practitioners without a cohesive framework for AI system development across the lifecycle. As a result, practitioners may face fragmented safety processes, duplicated work artifacts, and inconsistencies in safety practices across the lifecycle. This fragmentation not only complicates regulatory compliance but also increases development cost, prolongs validation cycles, and introduces uncertainty into safety assurance.

To resolve this, we introduced a unified lifecycle model that integrates the aforementioned standards into a cohesive process in [26]. It uses the well-established V-model from ISO 26262 as a structural backbone while embedding AI-specific phases such as AI system design and verification, AI component design and verification, and learning assurance, in the V-cycle. This combined V cycle is shown in Figure 2. On this V-cycle for AI systems, the clauses from ISO 21448 and ISO PAS 8800 are mapped. This integrated view provides practical clarity on where and how the standards intersect. It also helps teams identify which lifecycle phases are governed by which standards and where dedicated AI-focused processes are needed.

The Key lifecycle stages in the combined V cycle include:

- **Concept Phase:** The lifecycle begins with the definition of the intended functionality, hazard identification, and high-level ODD specification. ISO 21448 complements this by analysing hazards from functional insufficiencies or misuse and establishing the acceptance criteria, as opposed to ASILs in ISO 26262. For AI system, these can include model misclassifications or insufficient performance under rare conditions. Preliminary ODD boundaries and assumptions are established here but remain coarse, to be refined in later phases.
- **System Design Phase:** This phase encompasses both the *functional safety concept* and the *technical safety concept*, as shown in ISO 26262 activities in Figure 2. First, system-level safety goals are translated into functional safety requirements defining what must be achieved to mitigate hazards (functional safety concept). Next, the technical safety concept specifies how these goals are realised via architectural choices, safety mechanisms, and redundancies.For AI systems, this include ISO/PAS 8800's guidance on AI-specific safety requirement and design considerations. Similarly, ODD is refined for AI and non-AI subsystems.
- **AI Component Design and Implementation:** Following system design, detailed design and implementation occur at the component level. For conventional software, this includes low-level software design and coding based on ISO 26262. For AI components, this includes model training, dataset specification, and model selection—activities addressed by ISO/PAS 8800 under the umbrella of *AI learning assurance*. This is also the point where refined hazards, performance limitations, and system assumptions dictate training data selection and evaluation procedures.
- **Verification and Validation:** This phase combines traditional software verification with AI-specific testing and assurance. ISO 26262 defines structural test coverage and fault injection; ISO/PAS 8800 complements this with model robustness testing, out-of-distribution detection, and validation against the intended ODD. Here, lifecycle traceability becomes essential as safety requirements and ODD constraints must be testable and measurable.
- **Operation and Maintenance:** After deployment, the system must continue to operate safely under evolving conditions. ISO 21448 and ISO/PAS 8800 emphasize the importance of monitoring AI performance post-deployment and managing distribution shift. Updates to models or system configurations must trigger reassessment of safety assumptions. Maintaining traceability to earlier lifecycle decisions is vital during updates and re-validation.

3.3 Lifecycle-Spanning Safety Enablers

While lifecycle phases presented in the previous section provide structure, they cannot fully capture the cross-cutting nature of some key AI safety concepts. In this section, we present three such safety concepts that operate across lifecycle phases and unify the application of the standards.

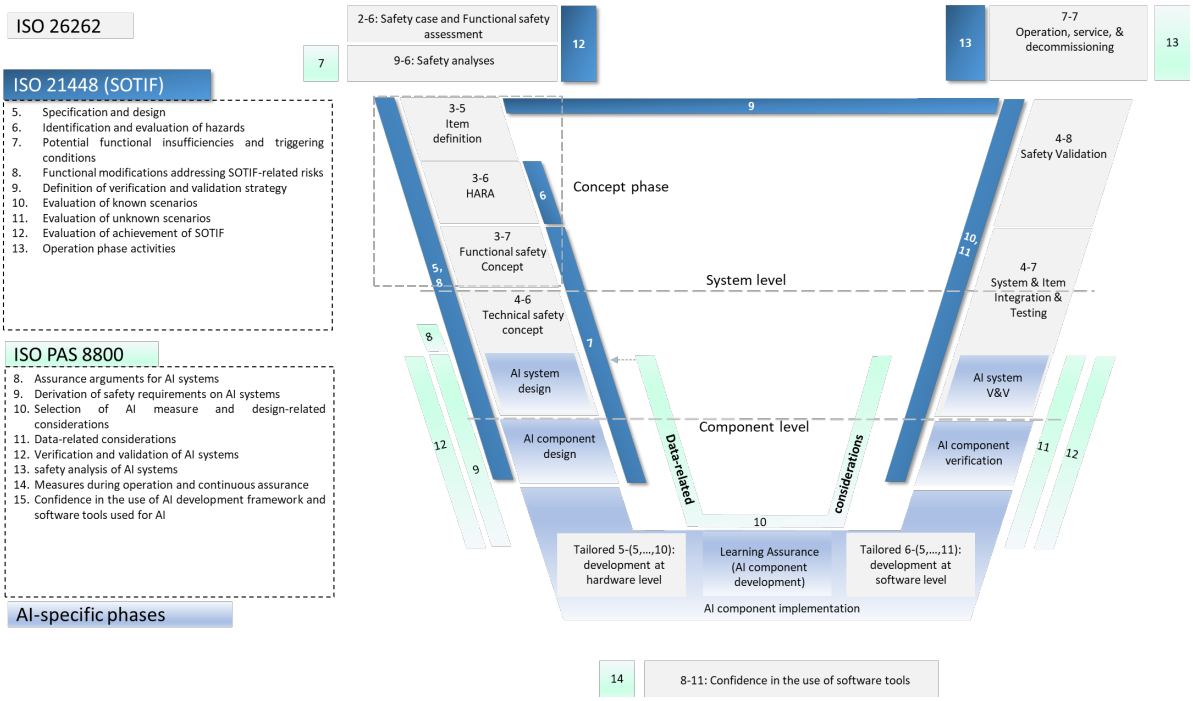


Figure 2: Reference AI-based product lifecycle for road vehicles based on ISO 26262 series and its possible interaction with ISO 21448 and ISO PAS 8800 [26].

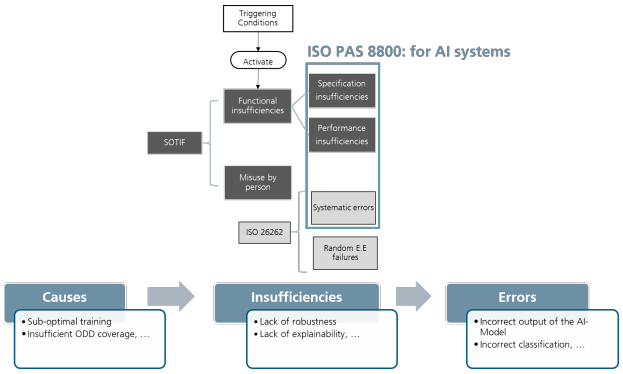


Figure 3: Hazardous model for AI systems.

3.3.1 Unified Hazardous Event Model

Hazard analysis remains a foundational activity in any safety lifecycle. In the context of AI-based systems, its scope must be broadened to account for both traditional and AI-specific failure modes. ISO 26262 identifies hazards stemming from systematic errors and hardware failures. These remain highly relevant for embedded AI systems in safety-critical domains.

ISO 21448 extends the hazard landscape to include functional insufficiencies, performance and specification limitations, as well as potential misuse by users. It introduces triggering conditions that expose unsafe system behavior not due to component failure, but due to limitations in the design or assumptions.

For AI systems, this challenge is compounded by the fact that such insufficiencies often lack a clear deterministic cause. ISO PAS 8800 addresses this by refining the notion of performance insufficiencies through the lens of *safety-related AI properties*, such as AI robustness, AI resilience, and AI explainability. Rather than direct causation, AI errors often emerge from the absence or insufficiency of these properties. For instance, incomplete data coverage during training can lead to reduced robustness at runtime, ultimately resulting in misclassifications or failures. As shown in Figure 3, these insufficiencies form a causal chain: sub-optimal development processes can compromise key AI safety-related properties, which in turn lead to erroneous outputs. Hazards in AI systems must be traced and mitigated across phases, from incomplete ODD specification in the concept phase to degraded robustness under real-world conditions during operation.

3.3.2 Hierarchical ODD Refinement

A unified AI safety lifecycle begins with a strong system specification, anchored in explicit safety goals, operational constraints, and assumptions about the intended environment. Central to this is the ODD, defined in ISO 21448 (based on SAE J3016) as the specific set of conditions in which a system or feature is intended to operate safely. In AI-based systems, the ODD not only bounds system behavior but also guides data collection, model training, and runtime monitoring.

In early concept phases, the system-level ODD serves as a high-level constraint space, defining where the system is intended to operate safely. As design progresses, it must be systematically refined across the architecture: from system to subsystems, components, and individual AI models. Each level introduces new assumptions and constraints that must remain consistent with the higher-level ODD definition. This hierarchical refinement supports realistic training, evaluation, and deployment strategies, allowing AI models to generalize within the ODD while gracefully handling out-of-

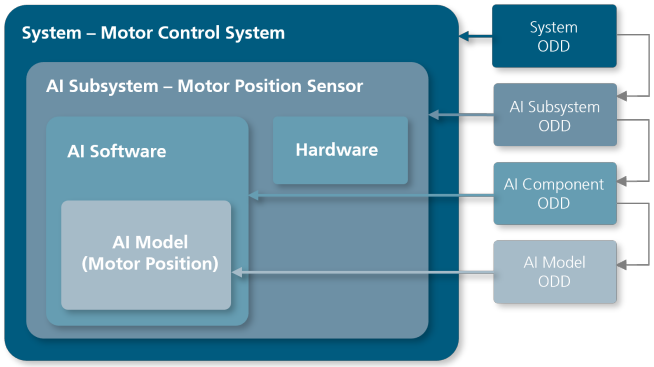


Figure 4: Hierarchical refinement of the Operational Design Domain (ODD) for AI-based Motor Control.

domain conditions.

Figure 4 illustrates this process in the AI-MC use case: system-level torque or speed limits are refined into AI model-specific constraints such as current and voltage sensor noise characteristics for AI-based motor position prediction. Clear ODD allocation across levels supports modular development and reduces assumption mismatches.

Importantly, the ODD continues to evolve beyond development. As the system is exposed to operational data, real-world edge cases, and potential distribution shifts, its understanding of safe operational conditions must adapt. AI systems must monitor runtime conditions to ensure they operate within their validated ODD and trigger fallback strategies when faced with unfamiliar or out-of-domain inputs.

> *The ODD is Alive — In AI systems, it's a safety envelope that evolves with every mile and every edge case.*

3.3.3 Multi-Level Operational Monitoring

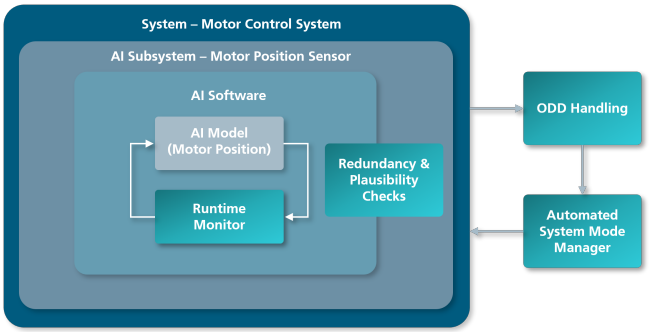


Figure 5: Hierarchical Operational monitoring.

Design-time efforts alone cannot eliminate all risks in AI systems. Residual insufficiencies, such as unforeseen inputs, model drift, or sensor noise, can only be fully addressed during operation. To detect, contain, and respond to such issues before they escalate, a *multi-layered operational monitoring* concept is required. Insufficiencies can surface at any stage, for example, at the level of the model, in interactions between components, or due to external factors beyond system control. No single layer of monitoring can guarantee safety across all contexts. A distributed monitoring strategy increases coverage and enables faster, more targeted mitigation. The operation time monitoring can be at the following levels:

- **Component Level:** AI models are monitored in real time for anomalies in input and performance. Techniques like Out-of-Distribution (OOD) detection, sensor plausibility checks, or confidence estimation help catch failures close to the source.
- **System Level:** Architectural redundancies (e.g., backup controllers or rule-based logic) safeguard the system when AI components fail. Runtime monitors compare AI outputs against expected behaviors to trigger safe fallbacks.
- **Vehicle Level:** At this level, managers such as the *ODD Handler* Unit and the *Automated System Mode Manager* enforce operation within validated boundaries and coordinate transitions to safe modes when conditions fall outside the ODD. Additionally, field data is continuously collected to capture edge cases and refine system behavior post-deployment.

4 The Role of Robustness and Resilience in AI Systems

Ensuring robustness and resilience is critical to maintaining safety in AI-driven applications throughout their lifecycle. Robustness ensures that an AI system performs reliably under nominal and slightly adverse conditions, such as sensor noise or input variability. When these limits are exceeded, resilience becomes essential. It preserves overall system functionality through higher-level safety mechanisms like runtime monitoring or fallback strategies [5]. Robustness and resilience describe an AI system's ability to handle variations in input conditions:

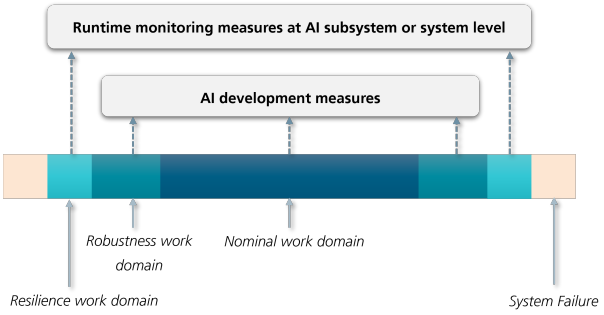


Figure 6: Robustness and resilience across operational domains (*adapted from [5]*).

- Nominal Work Domain: The scenarios the AI component is specifically designed for. These scenarios often correspond to the defined ODD.
- Robustness Domain: This extends beyond the nominal domain to include minor perturbations such as noise or mild data distribution shifts, ensuring the system maintains acceptable performance.

- Resilience Domain: When robustness limits are exceeded, higher-level mechanisms such as runtime monitoring (OOD detection and Uncertainty Quantification (UQ)) prevent complete system failure.
- Failure Domain: Scenarios beyond both robustness and resilience capabilities, where system-wide failures may occur, potentially causing hazards.

This layered understanding highlights the necessity of validating robustness during AI model development and complementing it with runtime measures such as monitoring, fallback procedures, and out-of-distribution detection. Together, these ensure the AI system maintains operational integrity in dynamic and uncertain real-world conditions.

Robustness vs. Resilience

Robustness extends the AI system's nominal domain, ensuring reliability in the face of minor adversities. Resilience maintains system-level functionality when robustness limits are exceeded.

In the following sections, we discuss the development and operation time measures for an AI-based Motor Control to enable robustness and resilience.

5 Development-Time Robustness Measures for AI-based Motor Control

5.1 Functional Robustness

5.1.1 Definition

Robustness generally refers to a model's ability to sustain an acceptable level of performance despite semantically insignificant yet reasonably expected variations in the input [19]. For regression tasks, a more formal definition is:

"The **robustness target** is said to be robust to the **robustness modifier** if relevant interventions in the modifier, as specified by the **robustness domain**, do not lead to greater changes in the target than specified by the **target tolerance** [9]."

To ground this definition in the context of our use case, we map its components to the AI-Based Position Sensor (AI-PS) as follows:

- **Robustness Target:** Output of the NN, i.e., the sine and cosine outputs of the AI-PS or the predicted motor position itself.
- **Robustness Modifier:** Inputs to the NN, i.e., outputs of the Clarke transform (i_α and i_β) and of the Inverse Park transform ($v_{\alpha\text{Ref}}$ and $v_{\beta\text{Ref}}$).
- **Robustness Domain:** Application-specific input perturbations that are considered plausible or expected under normal operating conditions or fault scenarios.
- **Target Tolerance:** Acceptable deviation in the predicted rotor position under such perturbations.

To enable formal robustness assessment, we adopt a probabilistic framing based on Pearl's do-operator [24], which captures the notion of applying explicit interventions to the system's input rather than passively observing it. Let T and M be random variables for the robustness **target** and the robustness **modifier**, respectively. Let $\mathcal{D}_M \subseteq M$ denote the **domain** of plausible perturbations, and let $\alpha_T \in \mathbb{R}_{\geq 0}$ be the **target tolerance**. Then we define robustness as:

$$\forall m \in \mathcal{D}_M, \quad Pr(d_T(T, T_{|\text{do}(M)=m}) \leq \alpha_T) \geq p, \quad (1)$$

where d_T describes a distance function on the output space of T. This expression states that for all perturbations m in $\mathcal{D}_M \subseteq M$, the probability that the output deviates from its unperturbed value by no more than α_T must be at least p. For this analysis, we assume $p = 0.95$.

5.1.2 Target Tolerance

Defining the target tolerance is a crucial part of assessing model robustness. In the AI-PS use case, it is derived by analyzing the impact of AI-based position sensor errors at the system level of the AI-MC. Through simulations, bias and Gaussian noise perturbations were introduced, and their effects on motor speed deviation were evaluated against predefined tolerance bounds (2% speed deviation and 10 ms persistence). The analysis showed that for loads between 0.0025 Nm and 0.004 Nm, a bias range of -10° to -5° and Gaussian noise variance below 0.001 met the 99.5% acceptability threshold. Although higher position errors are often acceptable in safety-critical systems, a conservative absolute tolerance of $\pm 3°$ is proposed to ensure a safety margin for closed-loop errors and maintain stable performance.

5.1.3 Robustness Domain

Robustness is typically evaluated by introducing input alterations within the robustness domain to the robustness modifier and measuring their impact on model performance. A crucial first step in this process is identifying realistic input variations to ensure a meaningful assessment. Figure 7 shows the input perturbations assessed for the AI-PS use case.

The magnitude of the applied perturbations for bias, gain, and Gaussian noise is derived from typical specifications of current sensors used in similar applications (e.g., [15]), reflecting realistic input variations. Values for series shift, short perturbation, omission, and constant fault are assumed based on plausible operational disturbances. These ranges for the perturbation magnitudes are summarized in Table 1.

Table 1: Robustness evaluation ranges.

Type	Robustness Domain (m)	Combination of Robustness Domain with Modifier
Gain	$[0.95, 1.05]$	Multiply input with m
Bias	$[-0.667, 0.667]$	Add m to input
Gaussian	$[0, 0.0133]$	Add perturbation with standard deviation m to input
Series Shift	$[0, 8]$	Shift currents/ voltages by m time steps
Short	$[-0.1, 0.1]$	Add m to input
Omission	$[1, 4]$	Set input to zero for m time steps on 1% of the dataset
Constant Fault	$[1, 8]$	Keep input fixed for m time steps on 1% of the dataset

Robustness is then assessed by applying these perturbations to the model inputs, as described in Table 1, and monitor-

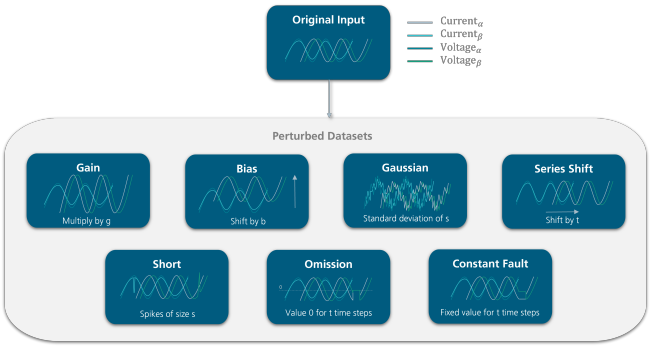


Figure 7: Perturbations applied to the currents and voltages.

ing the error of the AI-PS predictions. The percentages of samples within the acceptable position error bounds are visualized in Figure 8. The yellow region indicates the defined robustness domain; however, we also include perturbations beyond this region to analyze the model's behavior under OOD conditions in Section 6.2.

The results demonstrate that the AI-PS is robust against various common perturbations, including gain variations, Gaussian noise, constant faults, omissions, and short faults. For these input perturbations, over 95% of the model's predictions fall within the acceptable error margin of $3°$ specified at the system level. Series shifts were analyzed within the context of sampling intervals that exceed what is typically encountered in real-world systems. Bias perturbations may impact model performance. Functional robustness can be improved through several AI model development measures: augmenting training data with systematic input offsets to simulate sensor drifts, applying adversarial training with worst-case expected bias scenarios, using ensemble models to reduce prediction variance [23], and applying regularisation techniques [11]. Residual biases that remain despite these efforts can then be effectively detected and managed at the system level through runtime monitoring and fault isolation mechanisms.

5.2 Tool Robustness

Ensuring that the safety goals of an AI-based system are upheld in its target execution environment is explicitly emphasized in ISO/PAS 8800. This requirement poses a critical challenge for safety-critical AI systems, where the transformation from trained models to deployable code often involves a complex toolchain with multiple error-inducing steps. We extend the concept of development-time robustness by focusing on this transformation pipeline and its associated uncertainties.

5.2.1 Error Propagation Across Deployment Steps

The process of preparing a trained neural network for deployment on hardware such as Infineon's AURIX™ TC4x PPU™ can include multiple transformation steps, each of which may introduce approximation, as illustrated in Figure 9. Core steps may include:

- Model Importation: Floating-point (FP) models from frameworks like TensorFlow or ONNX are converted to an intermediate representation suitable for tooling. This may introduce an import error, denoted by E_{Imp}, as a result, potentially simplifying graph structures or altering layer-level fidelity.
- Quantization: Models are converted into Fixed Point (FXP) representations, where accuracy loss due to precision reduction introduces an error, representational errors in parameters and activations E_{Quant}.
- Optimization: Advanced optimization techniques refine the FXP model for performance and resource utilization, such as removing drop-out layers or scaling layers where the scale is 1. The error introduced due to this step is represented by E_{Opt}.
- Code Generation: Converts the optimized model into deployable code or binaries, typically via hardware-specific libraries or code templates, introducing error E_{Code}. Although this step can also introduce errors due to the approximation of functions and libraries used, we don't further discuss this error and assume functional correctness and no unintended behavior in this step.

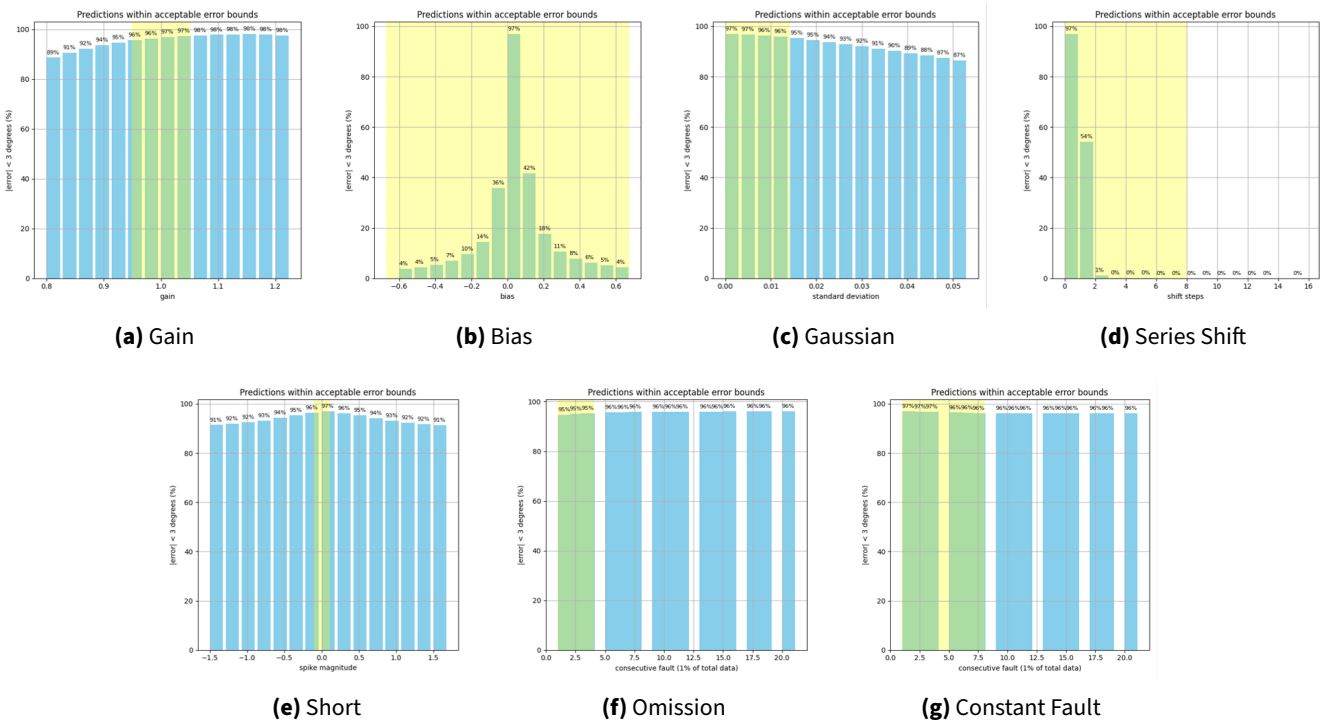
To maintain robustness and guarantee functional equivalence or bounded deviation from the trained model, we define an aggregate requirement:

$$E_{\text{Imp}} + E_{\text{Quant}} + E_{\text{Opt}} + E_{\text{Code}} \leq E_{\text{Req}}, \tag{2}$$

where E_{Req} is the maximum allowable error for this stage of the process. This inequality formalizes our goal: the cumulative transformation error must not exceed the system's acceptable error margin, as derived from the safety requirements. For our use case of AI-MC, we set the maximum allowable $E_{\text{Req}} \leq 1\,\text{deg}$. Since the network outputs sine and cosine components, this translates to a maximum tolerable vector error of approximately 0.017. Similarly, Mean Squared Error (MSE) can be set to 5×10^{-3}.

5.2.2 Empirical Evaluation of Toolchain Robustness

To validate whether practical toolchain transformations respect this bound, we conducted a full pipeline evaluation: importation, quantization, optimization, and code-level simulation. The NN layers and blobs were quantized between the range of 9 to 14 fractional bits. Figure 10 depicts the comparison of sine and cosine outputs between the original and quantized models. The MSE was measured at 0.0000029, which is well below our threshold. Figure 10 further confirms that over 95% of prediction errors lie within the range $[-0.005, 0.005]$, well below the derived threshold of 0.017. These results empirically validate that the transformation process remains within the defined safety envelope under nominal conditions. Nonetheless, further robustness margins should be reserved for scenarios involving perturbations like noise, gain shifts, or unmodeled operational conditions.



(a) Gain **(b)** Bias **(c)** Gaussian **(d)** Series Shift

(e) Short **(f)** Omission **(g)** Constant Fault

Figure 8: Percentage of data with a position error $< 3°$ for all types and magnitudes of perturbations. Yellow areas mark the defined robustness domain.

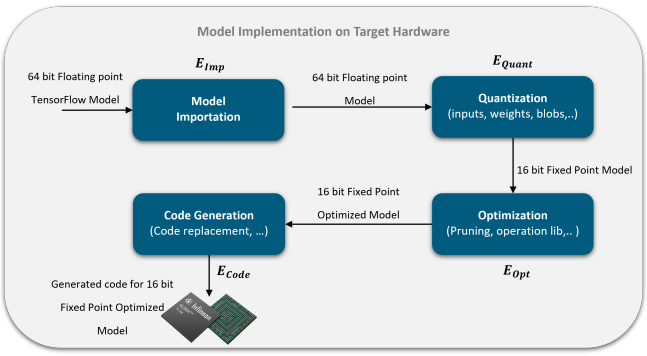


Figure 9: Cumulative error sources across the model deployment pipeline.

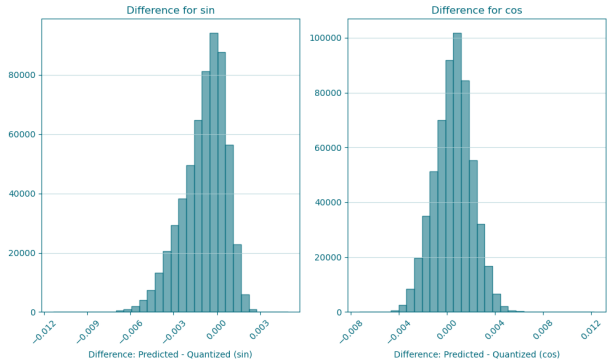


Figure 10: Comparison between the transformed and untransformed models.

6 Operation-Time Measures for AI-based Motor Control

6.1 Runtime Monitoring

6.1.1 Definition

Runtime monitoring plays a critical role in ensuring AI system resilience during operation. It becomes especially relevant when the system encounters conditions beyond its nominal and robustness work domain (cf. Table 1) [22]. While robustness is addressed during the design phase of the AI component, runtime monitoring mechanisms provide a dynamic safety net to detect and respond to deviations in real time. These mechanisms, such as OOD detection, are vital in managing scenarios where the AI component begins to fail, preventing broader system-level breakdowns.

This process is critical for identifying potential issues, such as uncertainty in predictions, OOD inputs, or performance degradation due to shifts in data patterns over time. Typically, two complementary components can be used for runtime monitoring: OOD detection and UQ [1].

6.1.2 Types of Runtime Monitoring

Definition: Runtime Monitoring [12]

"Runtime monitoring involves continuously tracking the behavior of the deployed AI model to ensure it operates reliably and delivers accurate predictions."

Runtime monitoring is critical for identifying potential issues, such as high uncertainty in predictions, OOD inputs,

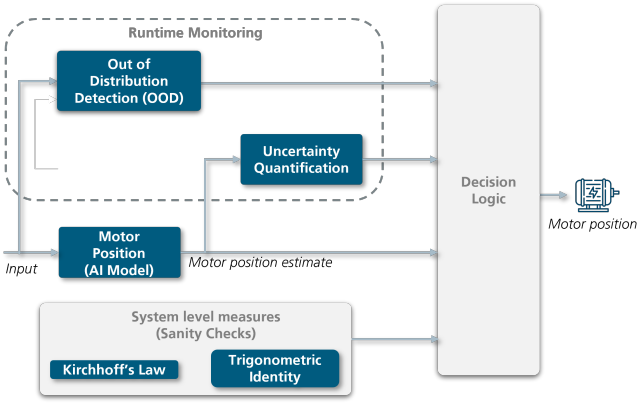


Figure 11: Overview of the runtime monitoring measures.

or performance degradation due to shifts in data patterns over time. Typically, two complementary components can be used for runtime monitoring: OOD detection and UQ [1].

- ***Out-Of-Distribution Detection:*** This focuses on identifying inputs that deviate significantly from the data distribution seen during training. Such inputs may originate from environmental changes, sensor noise, or operational shifts, and often lead to unreliable predictions.
- ***Uncertainty Quantification:*** While OOD detection identifies anomalous inputs, UQ assesses the confidence of the model's predictions. High uncertainty can for example indicate OOD inputs or poorly understood regions of the input space. Although an overview of UQ methods is provided, their implementation lies outside the scope of this report.

6.2 Out-of-Distribution Detection

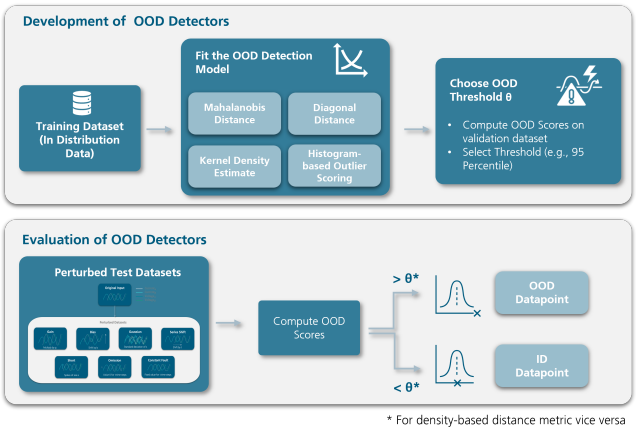


Figure 12: Development and evaluation pipeline for out-of-distribution detectors.

AI systems deployed in real-world environments often encounter data that differs from their training distribution. Such Out-of-Distribution (OOD) inputs may cause unpredictable or unsafe behavior if not properly identified. OOD detection aims to flag these inputs before they impact the system's operation.

This section presents a condensed version of our earlier work in [2] and considers four representative OOD detection methods, two distance-based and two density-based, which provide a numerical score for each input, compared against a threshold to determine whether it is In-Distribution (ID) or OOD. The methods are:

- Mahalanobis Distance: Measures how far an input deviates from the multivariate mean using a full covariance estimate [20].
- Diagonal Distance: A lightweight variant using diagonal distance to the distance based on use-case specific correlation of voltage and currents.
- Kernel Density Estimation (KDE): Estimates the data distribution non-parametrically, assigning low scores to sparse regions [6].
- Histogram-Based Outlier Scores (HBOS): Uses histograms of input features to assign anomaly scores based on observed density [10].

These methods vary not only in their principles but also in their practical characteristics. Mahalanobis and Diagonal Distance are simpler in their representational power but offer relatively lower computational cost and memory footprint, making them suitable for deployment. KDE and HBOS, while more expressive in modeling feature distributions, require substantial computational and memory resources, especially during training, where KDE must fit kernel functions and HBOS constructs detailed histograms. Thus, the choice of method may depend on the operational context, balancing detection performance with implementation constraints. Detailed description of the setup can be found in [2].

All four OOD detection methods follow a common development and evaluation pipeline, illustrated in Figure 12; they are trained using training data (assumed to be ID data)and tested across various perturbation types applied to test data, where extreme values represent the OOD data. Thresholds τ are tuned per method and perturbation to balance false positives and detection rates. This setup allows consistent benchmarking across methods and fault types.

The ideal desired behavior for OOD detectors is derived from the robustness analysis of the AI-PS, as illustrated in Figure 9. For perturbations where the AI system maintains functional robustness across all applied perturbation levels; such as Constant Fault, Omission, Short Fault, and Gain the detector should ideally avoid triggering false positives. This is captured through the condition of $\min(\text{OOD})$ at both nominal levels (e.g., $P = 0$) and the highest perturbation level P_{max}. For the higher perturbations level, we usually allow a small increase (typically up to 10%) in OOD detection rates. Here, P represents the applied perturbation level and OOD indicates the percentage of samples flagged as out-of-distribution.

Conversely, for perturbations such as Gain, Gaussian, or Load, where the AI system's performance degrades with increasing perturbation, the detector is expected to raise OOD flags more frequently. In these cases, the desired behavior includes low detection rates near the nominal condition (e.g., $\min(\text{OOD})$ at $P = 1$ for Gain), and a rise toward $\max(\text{OOD})$ as $P \to P_{\text{max}}$, reflecting the growing divergence from the training distribution.

Certain perturbations, such as Bias or Series-Shift, may lead to performance degradation even within the boundaries of the designated robustness domain. Ideally, in such situations, the functional robustness of the AI-PS should be improved to tolerate these perturbations by measures such as extending training data for within the robustness domain. For the purpose of current analysis, the detector should begin flagging OOD samples at those perturbation levels within the robustness domain where safety requirement is being violated.

Table 2 summarizes the performance of each method against these expectations. The Omission, Constant Fault, and Short Fault perturbations are not detected by all the evaluated OOD detectors, and it is considered that the neural network is robust against these perturbations. This is because only a small percentage of the test dataset (approximately 1%) is perturbed, resulting in minimal changes to the distribution of distances or densities, making it challenging to separate ID and OOD samples effectively. However, since the AI-PS component is maintains performance under these perturbations, this limitation is acceptable.

For Gain, Gaussian, and Load perturbations, the Diagonal method performs best, but none of the detectors fully meet the desired performance criteria. Mahalanobis is the most effective for Series Shift, though it still falls short of requirements. For Bias perturbations, HBOS demonstrates the best performance, but still exhibits inadequate performance.

Overall, these results indicate that no single OOD detection method alone can detect all perturbation types. While some detectors show relatively better performance for specific perturbations (e.g., Diagonal Distance for Gain, Mahalanobis for Series-Shift, HBOS for Bias and Gaussian), their suitability needs to be evaluated on a case-to-case basis. For perturbations where the AI system is inherently robust (Omission, Constant Fault, Short Fault), all OOD detectors performance is acceptable.

Therefore, for system owners aiming to ensure operational safety, these results indicate that OOD detection alone may not be sufficient. A robust safety strategy should combine OOD detection with complementary system-level monitoring measures, tailored to the nature and criticality of each perturbation. For instance, perturbations such as Gain, Bias, and Series-Shift could also be addressed through system-level sanity checks. Ultimately, the selection of detection and mitigation strategies should be systematically driven by the specific perturbation type, its impact on system performance, and the overall safety requirements.

7 Lesson Learned and Recommendations

This section presents practical insights into integrating AI-based components into established engineering processes, while maintaining the level of rigor expected in safety-relevant domains.

- **Define the ODD in a semi-formal, structured way.** The ODD is the foundation of any data-driven safety concept, defining where the system can safely operate, what data to collect, and expected behavior. A vague ODD ("urban driving") may be readable but not actionable; fully formal models are often impractical. Instead, a structured, semi-formal ODD, which is machine-readable and organized by parameters or categories, enables refinement across system levels, traceability to training data, and a consistent safety argument from design through implementation.
- **Extend proven system architectures for AI components.** In established applications with validated system designs, AI components can be added as modular enhancements without a full redesign. Integration must align with development lifecycles like the V-model and comply with standards such as ISO 26262, ISO 21448, and ISO/PAS 8800, including implementing safety mechanisms for the AI components to maintain overall system safety and reliability.
- **Define new roles and responsibilities.** AI-based systems require roles beyond traditional development and V&V. Roles such as data owner, data steward, ML developer, model integrator, and AI V&V engineer must be clearly defined, with explicit working instructions and integration into existing processes. This ensure that data sets and models are managed with the same level of traceability, accountability, and safety assurance as conventional software components.
- **AI components as Safety Element out of Context (SEooC)** SEooC principles can be applied to modules that rely on data-driven approaches. To do so effectively, suppliers must explicitly state assumptions such as input distributions, ODD boundaries, and performance expectations. These assumptions must be thoroughly validated during integration and continuously monitored at runtime, with interfaces provided by supplier to detect violations or degradations at runtime.
- **Aim for grey-box understanding, not black-box trust.** Even when internal model logic is complex, partial insight into its behavior can often be achieved. Tools for explainability, testing under constraints, and monitoring can support system-level safety reasoning without requiring full interpretability.
- **Design for real-time operation including hardware fit.** For tasks involving control or timely decision-making, real-time processing is critical. Suitable hardware platforms, such as Infineon AURIX™ with ASIL-D certification [16], enable high-integrity execution of AI-based components. AI model architecture should be chosen in alignment with these platform constraints to ensure reliable operation.

8 Conclusion

Achieving trustworthy embedded AI requires a harmonized approach ensuring safety is upheld throughout all phases of the product lifecycle, from development to operation. In this whitepaper, we present a unified lifecycle which harmonizes guidance from ISO 26262 [7], ISO PAS 21448 [8], and ISO PAS 8800 [19]. In particular, we highlight three activities which benefit from harmonization: the hazard and risk anal-

Table 2: Summary of the suitability of OOD methods for different perturbation types. *Olive* indicates that the desired behavior is fully met, meaning the method performs as expected for the given perturbation; *Orange* signifies that the method is the best among the compared methods, but it does not fully satisfy the requirements; *Red* indicates that the desired behavior is not met, meaning the method fails to adequately detect OOD samples as required by the perturbation.

Perturbation Method	Desired Behaviour	Mahalanobis Distance	Diagonal Distance	KDE-based	HBOS
Bias	$\min(\text{OOD})$ at $P = 0$ AND $\max(\text{OOD})$ at $P = P_{max}$	$\sim 3\%$ at $P = 0$ $\sim 20\%$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 5\%$ at $P = P_{\max}$	$\sim 3\%$ at $P = 0$ $\sim 50\%$ at $P = P_{\max}$	$\sim 3\%$ at $P = 0$ $\sim 65\%$ at $P = P_{\max}$
Constant Fault, Omission, Short	$\min(\text{OOD})$ at $P = 0$ AND $\min(\text{OOD})$ at $P = P_{\max}$	$\sim 3\%$ at $P = 0$ $\sim 3\%$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 0\%$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 0\%$ at $P = P_{\max}$	$\sim 1\%$ at $P = 0$ $\sim 1\%$ at $P = P_{\max}$
Gain	$\min(\text{OOD})$ at $P = 1$ AND $\min(\text{OOD})$ at $P = P_{\max}$	$\sim 3\%$ at $P = 1$ $\sim 60\%$ at $P = P_{\max}$	$\sim 2\%$ at $P = 1$ $\sim 18\%$ at $P = P_{\max}$	$\sim 3\%$ at $P = 1$ $\sim 78\%$ at $P = P_{\max}$	$\sim 1\%$ at $P = 1$ $\sim 83\%$ at $P = P_{\max}$
Gaussian	$\min(\text{OOD})$ at $P = 0$ AND $\max(\text{OOD})$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 90\%$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 42\%$ at $P = P_{\max}$	$\sim 1\%$ at $P = 0$ $\sim 95\%$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 100\%$ at $P = P_{\max}$
Series-Shift	$\min(\text{OOD})$ at $P = 0$ AND $\max(\text{OOD})$ at $P = P_{\max}$	$\sim 3\%$ at $P = 0$ $\sim 78\%$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 5\%$ at $P = P_{\max}$	$\sim 0\%$ at $P = 0$ $\sim 75\%$ at $P = P_{\max}$	$\sim 1\%$ at $P = 0$ $\sim 65\%$ at $P = P_{\max}$
Load	$\min(\text{OOD})$ for $P \leq 0.005$ AND $\max(\text{OOD})$ at $P > 0.01$	$< 45\%$ for $P \leq 0.005$ $\sim 100\%$ at $P > 0.01$	$< 35\%$ for $P \leq 0.005$ $\sim 100\%$ at $P > 0.03$	$< 75\%$ for $P \leq 0.005$ 100% at $P > 0.01$	$< 90\%$ for $P \leq 0.005$ $\sim 100\%$ at $P > 0.01$

ysis, the refinement of the ODD throughout the lifecycle, and the definition of operational monitoring at various levels of hierarchy (i.e. system, sub-system, and component levels). We make a particular focus on the role of robustness and resilience in AI systems and discuss how these properties might look in the context of an AI-MC use case. In doing so, we highlight runtime measures for robustness like OOD detection and uncertainty quantification, providing a quantitative analysis of the former. Through our work, we aim for narrowing the gap between groundbreaking AI capability and the uncompromising safety expectations of critical embedded systems.

References

[1] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[2] Shayari Bhattacharjee, Clarissa Heinemann, Leopold Mareis, Felippe Schmoeller da Roza, Shanza Ali Zafar, and Daniel Scholz. Development of runtime monitors for AI-based motor control with safety-critical applications. In *Proceedings of the FISITA World Mobility Conference (WMC) 2025 – Digitalisation & AI*. FISITA, June 2025.

[3] CSET Policy Brief. AI Accidents: An Emerging Threat, 2021.

[4] European Commission, Content Directorate-General for Communications Networks, Technology, and Grupa ekspertów wysokiego szczebla ds. sztucznej inteligencji. *Ethics guidelines for trustworthy AI*. Publications Office, 2019.

[5] Hervé Delseny, Christophe Gabreau, Adrien Gauffriau, Bernard Beaudouin, Ludovic Ponsolle, Lucian Alecu, Hugues Bonnin, Brice Beltran, Didier Duchel, Jean-Brice Ginestet, et al. White paper machine learning in certified systems. *arXiv preprint arXiv:2103.10529*, 2021.

[6] Ertunc Erdil, Krishna Chaitanya, and Ender Konukoglu. Unsupervised out-of-distribution detection using kernel density estimation. *arXiv preprint arXiv:2006.10712*, 2020.

[7] International Organization for Standardization. ISO 26262:2018 - Road vehicles – Functional safety, 2nd Edition, 2018.

[8] International Organization for Standardization. ISO/PAS 21448:2022 - Safety Of The Intended Functionality – SOTIF, 2022.

[9] Timo Freiesleben and Thomas Grote. Beyond generalization: a theory of robustness in machine learning. *Synthese*, 202(4):109, 2023.

[10] Markus Goldstein and Andreas Dengel. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track*, 1:59–63, 2012.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Regularization for deep learning. *Deep learning*, 1:224–270, 2016.

[12] Yuning He and Johann Schumann. Statistical analysis and runtime monitoring for an ai-based autonomous centerline tracking system. *International Journal of Prognostics and Health Management*, 15(3), 2024.

[13] Dan Hendrycks, Mantas Mazeika, and Thomas Woodside. An overview of catastrophic ai risks. *arXiv preprint arXiv:2306.12001*, 2023.

[14] IEEE. IEEE 7001-2021 - IEEE Standard for Transparency of Autonomous Systems, 2021.

[15] Infineon Technologies AG. TLI4971 - High-precision coreless magnetic current sensor for industrial applica-

tions. https://www.infineon.com/part/TLI4971-A120T5-E0001, 2023. Accessed: 2025-08-13.

[16] Infineon Technologies AG. 32-bit TriCore™ AURIX™ – TC4x. https://www.infineon.com/products/microcontroller/32-bit-tricore/aurix-tc4x#about, 2025. Accessed: 2025-07-30.

[17] ISO. ISO/IEC TR 24028:2020 Information technology — Artificial intelligence — Overview of trustworthiness in artificial intelligence, 2020.

[18] ISO. ISO/IEC 24029-1:2021: Assessment of the robustness of neural networks Part 1 - Overview, 2021.

[19] ISO. ISO/CD PAS 8800 - Road Vehicles – Safety and artificial intelligence, 2024.

[20] Ryo Kamoi and Kei Kobayashi. Why is the mahalanobis distance effective for anomaly detection? *arXiv preprint arXiv:2003.00402*, 2020.

[21] Underwriters' Laboratories. Ul 4600: Standard for evaluation of autonomous products, 3rd edition, 2023.

[22] Hira Naveed, John Grundy, Chetan Arora, Hourieh Khalajzadeh, and Omar Haggag. Towards runtime monitoring for responsible machine learning using model-driven engineering. In *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems*, pages 195–202, 2024.

[23] Wendy S Parker. Ensemble modeling, uncertainty and robust predictions. *Wiley interdisciplinary reviews: Climate change*, 4(3):213–223, 2013.

[24] Judea Pearl. *Causality*. Cambridge university press, 2009.

[25] John M Scanlon, Kristofer D Kusano, Tom Daniel, Christopher Alderson, Alexander Ogle, and Trent Victor. Waymo simulated driving behavior in reconstructed fatal crashes within an autonomous vehicle operating domain. *Accident Analysis & Prevention*, 163:106454, 2021.

[26] Shanza Ali Zafar, Jessica Kelly, and Núria Mata. Unified ai-product lifecycle based on road-vehicle safety standards. In *2025 20th European Dependable Computing Conference Companion Proceedings (EDCC-C)*, pages 189–194. IEEE, 2025.

Published by
Infineon Technologies AG
Am Campeon 1-15, 85579 Neubiberg
Germany

Public

Version: V1.0_EN
Date: 08/2025

Stay connected!

Scan QR code and explore offering
www.infineon.com

Please note!
This Document is for information purposes only and any information given herein shall in no event be regarded as a warranty, guarantee or description of any functionality, conditions and/or quality of our products or any suitability for a particular purpose. With regard to the technical specifications of our products, we kindly ask you to refer to the relevant product data sheets provided by us. Our customers and their technical departments are required to evaluate the suitability of our products for the intended application.

We reserve the right to change this document and/or the information given herein at any time.

Additional information
For further information on technologies, our products, the application of our products, delivery terms and conditions and/or prices, please contact your nearest Infineon Technologies office (www.infineon.com).

Warnings
Due to technical requirements, our products may contain dangerous substances. For information on the types in question, please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by us in a written document signed by authorized representatives of Infineon Technologies, our products may not be used in any life-endangering applications, including but not limited to medical, nuclear, military, life-critical or any other applications where a failure of the product or any consequences of the use thereof can result in personal injury.